User Manual

ACM5000 Remote Site Managers

ACM5500 Management Gateways

ACM7000 Resilience Gateways

IM7200 & IM4200 Infrastructure Managers

CM7100 Console Servers

SD4000 Device Servers

Rev: 4.18p0

Jun 17th 2015

Safety

Please take care to follow the safety precautions below when installing and operating the console server:

- Do not remove the metal covers. There are no operator serviceable components inside. Opening or removing the cover may expose you to dangerous voltage which may cause fire or electric shock. Refer all service to Opengear qualified personnel.
- To avoid electric shock the power cord protective grounding conductor must be connected through to ground.
- Always pull on the plug, not the cable, when disconnecting the power cord from the socket.

Do not connect or disconnect the console server during an electrical storm. Also it is recommended you use a surge suppressor or UPS to protect the equipment from transients.

FCC Warning Statement

This device complies with Part 15 of the FCC rules. Operation of this device is subject to the following conditions: (1) This device may not cause harmful interference, and (2) this device must accept any interference that may cause undesired operation.

Proper back-up systems and necessary safety devices should be utilized to protect against injury, death or property damage due to system failure. Such protection is the responsibility of the user.

This console server device is not approved for use as a life-support or medical system.

Any changes or modifications made to this console server device without the explicit approval or consent of Opengear will void Opengear of any liability or responsibility of injury or loss caused by any malfunction.

This equipment is for indoor use and all the communication wirings are limited to inside of the building.

Publishing history

Date	Revision	Update details
Jan 2010	3.8.4	SD4001 product
Mar 2010	3.8.5	ACM5004-G, fixed Failover details and added DDNS
June 2010	3.9	V3.1 (shadow password, deg F, SNMP, SMS gateway) and ACM5004-I
Aug 2010	3.9.1	V3.2 (OpenVPN, Zenoss, config commit, Call Home)
Dec 2010	4.0	V3.3 (Firewall router, Web Terminal, SNMP updates)
June 2011	4.1	V3.4 (GPS support, SNMP traffic monitoring and IPv6, 32 port models, SMS over cellular)
Oct 2011	4.2	V3.5 (Auto-Response)
Nov 2011	4.3	V3.5.2 (PPTP, GRE, Groups, FTP server, multiple dial-in, pmshell). Add IM4216-34
Feb 2012	4.4	V3.5.2u3 (Kerberos, Cisco RJ in SD4000, Add ACM5500, Remove KCS)
April 2012	4.5	V3.5.2u14 (Cellular redial)
July 2012	4.6	V3.5.3 (SMS ARM, simple key, Services page) and SD4001 Rev-01, EoL CM4001/4008
Dec 2012	4.7	V3.6 (Authenticated NTP), ACM5504-5-G-W-I release and EoL IM4004-5
April 2013	4.8	V3.7, 4G LTE support
Sept 2013	4.9	V3.8, IM7200
Oct 2013	4.10	IM7208 and DDC models
Jan 2014	4.11	V3.9 (Dual SIM, SNMP DIO, bulk provisioning), WEEE
Mar 2104	4.12	V3.10 (Connection Manager network management backend, Auto-Response extensions)
July 2014	4.13	V3.11 (New SNMP MIB, OpenLDAP, LDAPS)
Sep 2014	4.14	V3.12 (New Manage Devices UI, Brute Force Protection)
Oct 2014	4.15	CM7100
Jan 2015	4.16	V3.15 (IP Passthrough)
Feb 2015	4.17	V3.15.1 (ZTP), EoL CM4100
Jun 2015	4.18	V3.16, ACM7000

Copyright

© Opengear Inc. 2015. All Rights Reserved.

Information in this document is subject to change without notice and does not represent a commitment on the part of Opengear. Opengear provides this document “as is,” without warranty of any kind, expressed or implied, including, but not limited to, the implied warranties of fitness or merchantability for a particular purpose.

Opengear may make improvements and/or changes in this manual or in the product(s) and/or the program(s) described in this manual at any time. This product could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes may be incorporated in new editions of the publication.

TABLE OF CONTENTS

THIS MANUAL 12
INSTALLATION 16
2.1 Models 16
2.1.1 ACM5000 kit components 17
2.1.2 ACM5500 kit components 18
2.1.3 IM4208-2, IM4216-2, IM4232-2, IM4248-2 and IM4216-34 kit components 18
2.1.4 IM7208-2, IM7216-2, IM7232-2 and IM7248-2 kit components 19
2.1.5 CM7116-2, CM7132-2 and CM7148-2 kit components 19
2.1.6 SD4002 kit components 20
2.1.7 SD4001 kit components 20
2.2 Power Connection 21
2.2.1 All IM7200, IM4200 and CM7100 -DAC models 21
2.2.2 SD4002 and SD4001 power 21
2.2.3 All ACM5000 models 22
2.2.4 All ACM5500 models 22
2.2.5 IM7200-DDC and IM4200-DDC power 23
2.3 Network Connection 24
2.4 Serial Port Connection 24
2.5 USB Port Connection 26
2.6 Fitting Cellular SIM and Antennas 27
2.6.1 ACM5004-G models 28
2.6.3 ACM5500-L models 29
2.6.4 IM4200-G models 30
2.6.5 All IM7200 models 30
2.6.6 IM7200-L models 30
2.7 Digital I/O and Environmental Sensors 31
SYSTEM CONFIGURATION 32
3.1 Management Console Connection 32
3.1.1 Connected computer set up 32
3.1.2 Browser connection 33
3.2 Administrator Set up 34
3.2.1 Change default root System Password 34
3.2.2 Set up a new Administrator 35
3.2.3 Name the System 35
3.3 Network Configuration 36
3.3.1 IPv6 configuration 37
3.3.2 Dynamic DNS (DDNS) configuration 37
3.4 Services and Service Access 39
3.4.1 Brute Force Protection 42
3.5 Communications Software 43
3.5.1 SDT Connector 43
3.5.2 PuTTY 43
3.5.3 SSHTerm 44
3.6 Management Network Configuration 44
3.6.1 Enable the Management LAN 44
3.6.2 Configure the DHCP server 46
3.6.3 Select Failover or broadband OOB 48
3.6.4 Aggregating the network ports 49
3.6.5 Wi-Fi Wireless LAN 50
3.6.7 Static routes 53
SERIAL PORT, HOST, DEVICE & USER CONFIGURATION 56
4.1 Configure Serial Ports 56
4.1.1 Common Settings 57

4.1.2 Console Server Mode 58
4.1.3 SDT Mode 62
4.1.4 Device (RPC, UPS, EMD) Mode 63
4.1.5 Terminal Server Mode 63
4.1.6 Serial Bridging Mode 63
4.1.7 Syslog 64
4.1.8 NMEA Streaming 65
4.1.9 Cisco USB console connection 66
4.2 Add / Edit Users 67
4.2.1 Set up new Group 68
4.2.2 Set up new Users 69
4.3 Authentication 70
4.4 Network Hosts 70
4.5 Trusted Networks 72
4.6 Serial Port Cascading 73
4.6.1 Automatically generate and upload SSH keys 74
4.6.2 Manually generate and upload SSH keys 75
4.6.3 Configure the slaves and their serial ports 76
4.6.4 Managing the slaves 77
4.7 Serial Port Redirection (PortShare) 77
4.8 Managed Devices 78
4.9 IPsec VPN 80
4.9.1 Enable the VPN gateway 81
4.10 OpenVPN 83
4.10.1 Enable the OpenVPN 83
4.10.2 Configure as Server or Client 85
4.10.3 Windows OpenVPN Client and Server set up 87
4.11 PPTP VPN 92
4.11.1 Enable the PPTP VPN server 92
4.11.2 Add a PPTP user 94
4.11.3 Set up a remote PPTP client 94
4.12 Call Home 95
4.12.1 Set up Call Home candidate 96
4.12.2 Accept Call Home candidate as Managed Console Server on CMS 97
4.12.3 Calling Home to a generic central SSH server 98
4.13 IP Passthrough 99
4.13.1 Downstream Router Setup 99
4.13.1 IP Passthrough Pre-Configuration 99
4.13.2 IP Passthrough Configuration 100
4.13.3 Service Intercepts 100
4.13.4 IP Passthrough Status 100
4.13.5 Caveats 101
FIREWALL, FAILOVER & OOB ACCESS 102
5.1 Dialup Modem Connection 102
5.2 OOB Dial-In Access 102
5.2.1 Configure Dial-In PPP 102
5.2.2 Using SDT Connector client 105
5.2.3 Set up Windows XP or later client 105
5.2.4 Set up earlier Windows clients 105
5.2.5 Set up Linux clients 106
5.3 Dial-Out Access 106
5.3.1 Always-on dial-out 106
5.3.2 Failover dial-out 107
5.4 OOB Broadband Ethernet Access 109
5.5 Broadband Ethernet Failover 110

5.6 Cellular Modem Connection 111
5.6.1 Connecting to a GSM HSUPA/UMTS carrier network 111
5.6.2 Connecting to a CDMA EV-DO carrier network 113
5.6.3 Connecting to a 4G LTE carrier network 115
5.6.4 Verifying the cellular connection 116
5.6.5 Cellular modem watchdog 117
5.6.6 Dual SIM failover 117
5.7 Cellular Operation 119
5.7.1 OOB access set up 119
5.7.2 Cellular failover setup 120
5.7.3 Cellular routing 121
5.7.4 Cellular CSD dial-in setup 122
5.8 Firewall & Forwarding 123
5.8.1 Configuring network forwarding and IP masquerading 124
5.8.2 Configuring client devices 125
5.8.3 Port / Protocol forwarding 127
5.8.4 Firewall rules 128
SSH TUNNELS & SDT CONNECTOR 132
6.1 Configuring for SSH Tunneling to Hosts 133
6.2 SDT Connector Client Configuration 133
6.2.1 SDT Connector client installation 133
6.2.2 Configuring a new gateway in the SDT Connector client 134
6.2.3 Auto-configure SDT Connector client with the user's access privileges 136
6.2.4 Make an SDT connection through the gateway to a host 136
6.2.5 Manually adding hosts to the SDT Connector gateway 137
6.2.6 Manually adding new services to the new hosts 138
6.2.7 Adding a client program to be started for the new service 140
6.2.8 Dial in configuration 142
6.3 SDT Connector to Management Console 142
6.4 SDT Connector - telnet or SSH connect to serially attached devices 143
6.5 Using SDT Connector for out-of-band connection to the gateway 144
6.6 Importing (and exporting) preferences 145
6.7 SDT Connector Public Key Authentication 146
6.8 Setting up SDT for Remote Desktop access 146
6.8.1 Enable Remote Desktop on the target Windows computer to be accessed 147
6.8.2 Configure the Remote Desktop Connection client 148
6.9 SDT SSH Tunnel for VNC 150
6.9.1 Install and configure the VNC Server on the computer to be accessed 150
6.9.2 Install, configure and connect the VNC Viewer 152
6.10 Using SDT to IP connect to hosts that are serially attached to the gateway 153
6.10.1 Establish a PPP connection between the host COM port and console server 154
6.10.2 Set up SDT Serial Ports on console server 156
6.10.3 Set up SDT Connector to ssh port forward over the console server Serial Port 157
6.11 SSH Tunneling using other SSH clients (e.g. PuTTY) 157
ALERTS, AUTO-RESPONSE AND LOGGING 162
7.1 Configure Auto-Response 162
7.2 Check Conditions 164
7.2.1 Environmental 164
7.2.2 Alarms and Digital Inputs 165
7.2.3 UPS / Power Supply 165
7.2.4 UPS Status 166
7.2.5 Serial Login, Signal or Pattern 166
7.2.6 ICMP Ping 167
7.2.7 Cellular Data 168
7.2.8 Custom Check 168
7.2.9 SMS Command 169

7.2.10 Log In/Out Check 170
7.2.11 Network Interface Event 170
7.2.12 Routed Data Usage Check 171
7.3 Trigger Actions 172
7.3.1 Send Email 172
7.3.2 Send SMS 172
7.3.3 Perform RPC Action 173
7.3.4 Run Custom Script 173
7.3.5 Send SNMP Trap 173
7.3.6 Send Nagios Event 173
7.3.7 Perform Interface Action 173
7.4 Resolve Actions 174
7.5 Configure SMTP, SMS, SNMP and/or Nagios service for alert notifications 174
7.5.1 Send Email alerts 174
7.5.2 Send SMS alerts 175
7.5.3 Send SNMP Trap alerts 177
7.5.4 Send Nagios Event alerts 179
7.6 Logging 179
7.6.1 Log storage 179
7.6.2 Serial port logging 180
7.6.3 Network TCP and UDP port logging 181
7.6.4 Auto-Response event logging 181
7.6.5 Power device logging 182
POWER, ENVIRONMENT & DIGITAL I/O 184
8.1 Remote Power Control (RPC) 184
8.1.1 RPC connection 184
8.1.2 RPC access privileges and alerts 188
8.1.3 User power management 188
8.1.4 RPC status 188
8.2 Uninterruptible Power Supply Control (UPS) 190
8.2.1 Managed UPS connections 190
8.2.2 Remote UPS management 193
8.2.3 Controlling UPS powered computers 194
8.2.4 UPS alerts 195
8.2.5 UPS status 195
8.2.6 Overview of Network UPS Tools (NUT) 197
8.3 Environmental Monitoring 199
8.3.1 Connecting the EMD and its sensors 199
8.3.2 Connecting sensors to ACM5000 and ACM5500s 200
8.3.3 Adding EMDs and configuring the sensors 202
8.3.4 Environmental alerts 204
8.3.5 Environmental status 204
8.4 Digital I/O Ports 205
8.4.1 Digital I/O Output Configuration 205
8.4.2 Digital I/O Input Configuration 206
8.4.3 High Voltage Outputs 206
8.4.4 DIO SNMP status 206
AUTHENTICATION 208
9.1 Authentication Configuration 208
9.1.1 Local authentication 208
9.1.2 TACACS authentication 208
9.1.3 RADIUS authentication 210
9.1.4 LDAP authentication 211
9.1.5 RADIUS/TACACS user configuration 214
9.1.6 Group support with remote authentication 215

9.1.7 Remote groups with RADIUS authentication *215*
9.1.8 Remote groups with LDAP authentication *217*
9.1.9 Remote groups with TACACS+ authentication *218*
9.1.10 Idle timeout *219*
9.1.11 Kerberos authentication *219*
9.1.12 Authentication testing *219*
9.2 PAM (Pluggable Authentication Modules) 220
9.3 SSL Certificate 221
NAGIOS INTEGRATION 224
10.1 Nagios Overview 224
10.2 Configuring Nagios distributed monitoring 224
10.2.1 Enable Nagios on the console server *225*
10.2.2 Enable NRPE monitoring *226*
10.2.3 Enable NSCA monitoring *226*
10.2.4 Configure selected Serial Ports for Nagios monitoring *227*
10.2.5 Configure selected Network Hosts for Nagios monitoring *228*
10.2.6 Configure the upstream Nagios monitoring host *229*
10.3 Advanced Distributed Monitoring Configuration 229
10.3.1 Sample Nagios configuration *229*
10.3.2 Basic Nagios plug-ins *232*
10.3.3 Additional plug-ins *232*
10.3.4 Number of supported devices *234*
10.3.5 Distributed Monitoring Usage Scenarios *235*
SYSTEM MANAGEMENT 238
11.1 System Administration and Reset 238
11.2 Upgrade Firmware 239
11.3 Configure Date and Time 240
11.4 Configuration Backup 241
11.5 Delayed Configuration Commit 243
11.6 FIPS Mode 245
STATUS REPORTS 246
12.1 Port Access and Active Users 246
12.2 Statistics 247
12.3 Support Reports 248
12.4 Syslog 249
12.5 Dashboard 249
12.5.1 Configuring the Dashboard *250*
12.5.2 Creating custom widgets for the Dashboard *251*
MANAGEMENT 254
13.1 Device Management 254
13.2 Port and Host Logs 255
13.3 Terminal Connection 255
13.3.1 Web Terminal *255*
13.3.2 SDT Connector access *257*
13.4 Power Management 257
CONFIGURATION FROM THE COMMAND LINE 260
14.1 Accessing *config* from the command line 260
14.1.1 Serial Port configuration *262*
14.1.2 Adding and removing Users *265*
14.1.3 Adding and removing user Groups *266*
14.1.4 Authentication *267*
14.1.5 Network Hosts *267*
14.1.6 Trusted Networks *269*
14.1.7 Cascaded Ports *269*
14.1.8 UPS Connections *270*
14.1.9 RPC Connections *271*

14.1.10 Environmental 272
14.1.11 Managed Devices 272
14.11.12 Port Log 273
14.1.13 Alerts 273
14.1.14 SMTP & SMS 276
14.1.15 SNMP 276
14.1.16 Administration 277
14.1.17 IP settings 277
14.1.18 Date & Time settings 278
14.1.19 Dial-in settings 278
14.1.20 DHCP server 279
14.1.21 Services 280
14.1.22 NAGIOS 280
ADVANCED CONFIGURATION 282
15.1 Custom Scripting 282
15.1.1 Custom script to run when booting 282
15.1.2 Running custom scripts when alerts are triggered 282
15.1.3 Example script - Power cycling on pattern match 283
15.1.4 Example script - Multiple email notifications on each alert 284
15.1.5 Deleting configuration values from the CLI 284
15.1.6 Power cycle any device upon a ping request failure 286
15.1.8 Backing-up the configuration and restoring using a local USB stick 288
15.1.9 Backing-up the configuration off-box 289
15.2 Advanced Portmanager 290
15.2.1 Portmanager commands 290
15.2.2 External Scripts and Alerts 293
15.3 Raw Access to Serial Ports 294
15.3.1 Access to serial ports 294
15.3.2 Accessing the console/modem port 295
15.4 IP Filtering 295
15.5 SNMP Status Reporting 296
15.5.1 Retrieving status information using SNMP 296
15.5.2 Check firewall rules 296
15.5.3 Enable SNMP Service 296
15.5.4 Adding multiple remote SNMP managers 301
15.6 Secure Shell (SSH) Public Key Authentication 302
15.6.1 SSH Overview 302
15.6.2 Generating Public Keys (Linux) 303
15.6.3 Installing the SSH Public/Private Keys (Clustering) 303
15.6.4 Installing SSH Public Key Authentication (Linux) 304
15.6.5 Generating public/private keys for SSH (Windows) 305
15.6.6 Fingerprinting 307
15.6.7 SSH tunneled serial bridging 307
15.6.8 SDT Connector Public Key Authentication 309
15.7 Secure Sockets Layer (SSL) Support 310
15.8 HTTPS 310
15.8.1 Generating an encryption key 310
15.8.2 Generating a self-signed certificate with OpenSSL 311
15.8.3 Installing the key and certificate 311
15.8.4 Launching the HTTPS Server 311
15.9 Power Strip Control 312
15.9.1 The PowerMan tool 312
15.9.2 The pmpower tool 313
15.9.3 Adding new RPC devices 313
15.10 IPMItool 314

15.11 Custom Development Kit (CDK) 317
15.12 Scripts for Managing Slaves 317
15.13 SMS Server Tools 318
15.14 Multicast 318
15.15 Bulk Provisioning 319
15.16 Zero Touch Provisioning 319
15.16.1 Preparation 319
15.16.2 Example ISC DHCP server configuration 320
15.16.3 Setup for an untrusted LAN 320
15.16.4 How it works 320
APPENDIX A: Linux Commands & Source Code 324
APPENDIX B: Hardware Specification 330
APPENDIX C: Safety & Certifications 332
APPENDIX D: Connectivity, TCP Ports & Serial I/O 334
APPENDIX E: TERMINOLOGY 344
APPENDIX F: END USER LICENSE AGREEMENTS 348
APPENDIX G: SERVICE & STANDARD WARRANTY 354

THIS MANUAL

This User Manual describes the features and capabilities of the following Opengear product lines, and provides you with instructions to best take advantage of them:

- ACM5504-5-G/GV-W-I, ACM5504-5-G/GV-I, ACM5504-5-LA/LR/LV-I, ACM5504-2-P, ACM5508-2, ACM5508-2-M and ACM5008-2-P Remote Management Gateways
- ACM5002-F-E, ACM5003-M-F-E, ACM5004-F-E & ACM5004-2-I Remote Site Managers (and -G/GV/GS models with cellular support)
- IM7248-2-DAC, IM7232-2-DAC and IM7216-2-DAC Infrastructure Managers (and -LA/LR/LV models with 4G LTE)
- IM4248-2-DAC (or DDC), IM4232-2-DAC (DDC), IM4216-2-DAC (DDC), IM4216-34-DAC (DDC) & IM4208-2-DAC Infrastructure Managers (and -G/GV models)
- CM7116-2-DAC, CM7132-2-DAC & CM7148-2-DAC Console Servers
- SD4001 & SD4002 Secure Device Servers

Each of these products is referred to generically in this manual as a *console server.* Where appropriate product groups may be referred to as *console servers, gateways* or by specific product line name or product group (e.g. *IM4200 family*, *ACM5500*).

Who should read this guide?

You should read this manual if you are responsible for evaluating, installing, operating, or managing an Opengear appliance. This manual assumes you are familiar with the internal network of your organization, and are familiar with the Internet and IP networks, HTTP, FTP and basic security operations.

Manual Organization

This manual contains the following chapters:

1. Introduction	An overview of the features of the *console server* and information on this manual
2. Installation	Physical installation of the *console server* and the interconnecting of managed devices
3. System Configuration	Covers initial installation and configuration of the *console server* on the network and the services that will be supported
4. Serial & Network	Covers configuring serial ports and connected network hosts, and setting up users
5. Firewall, Failover & OOB	Describes setting up the firewall router functions and the high availability access features of the *console server*
6. Secure Tunneling	Covers secure remote access using SSH and configuring for RDP, VNC, HTTP, HTTPS etc. access to network and serially connected devices
7. Auto-Response and Logs	Explains the setting up of local and remote event / data logs and configuring auto-response actions to trigger events
8. Power & Environment	Management of USB, serial and network attached power strips and UPS supplies. EMD environmental sensor configuration
9. Authentication	All access to the *console server* requires usernames and passwords which are locally or externally authenticated
10. Nagios Integration	Setting Nagios central management and configuring the *console server* as a distributed Nagios server
11. System Management	Covers access to and configuration of services to be run on the *console server*

12. Status Reports	View a dashboard summary and detailed status and logs of serial and network connected devices (ports, hosts, power and environment)
13. Management	Includes port controls and reports that can accessed by *Users*
14. Basic Configuration	Command line installation and configuration using the *config* command
15. Advanced Config	Advanced command line configuration activities using Linux commands

The latest update of this manual can be found online at *www.opengear.com/download.html*

Types of users

The *console server* supports two classes of users:

I. Firstly there are the administrative users who have unlimited configuration and management privileges over the *console server*; and all the connected devices. These administrative users will be set up as members of the **admin** user group and any user in this class is referred to generically in this manual as the *Administrator*. An *Administrator* can access and control the *console server* using the *config* utility, the Linux command line or the browser based Management Console. By default the *Administrator* has access to all services and ports to control all the serial connected devices and network connected devices (*hosts*).

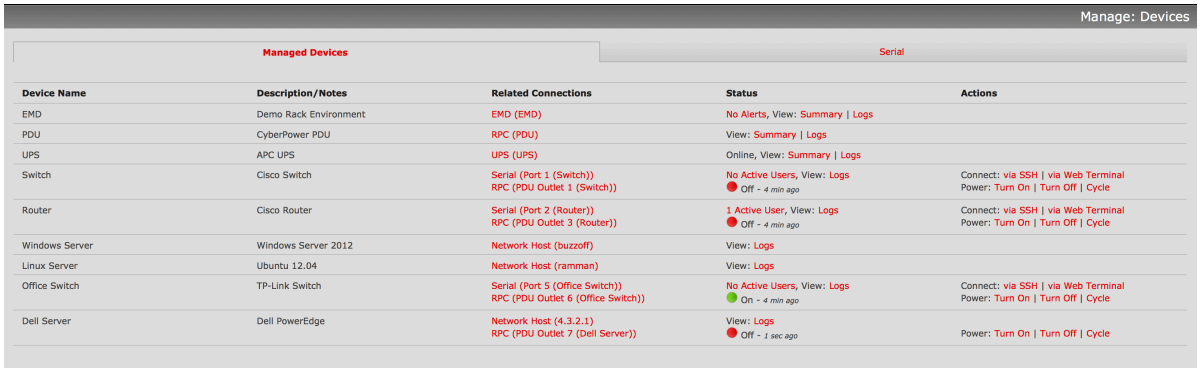
II. The second class of users embraces those who have been set up by the *Administrator* with specific limits of their access and control authority. These users are set up as members of the one of the pre-configured user groups (*pptpd, dialin, ftp, pmshell* or *users*) - or some other user groups the *Administrator* may have added. They are only authorized to perform specified controls on specific connected devices are referred to as *Users*. These *Users* (when authorized) can access serial or network connected devices; and control these devices using the specified services (e.g. Telnet, HHTPS, RDP, IPMI, Serial over LAN, Power Control). An authorized *User* also has a limited view the Management Console and can only access authorized configured devices and review port logs.

In this manual, when the term *user* (lower case) is used, it is referring to both the above classes of users. This document also uses the term *remote users* to describe users who are not on the same LAN segment as the *console server*. These remote users may be *Users,* who are on the road connecting to managed devices over the public Internet, or it may be an *Administrator* in another office connecting to the *console server* itself over the enterprise VPN, or the remote user may be in the same room or the same office but connected on a separate VLAN to the *console server*.

Management Console

The features of your *console server* are configured and monitored using the Opengear Management Console. When you first browse to the Management Console, you can use the menu displayed on the left side to configure the *console server*. Once you have completed the initial configuration, you can continue to use the Management Console runs in a browser and provides a view of the *console server* and all the connected devices.

Administrators can use the Management Console, either locally or from a remote location, to configure and manage the console server, users, ports, hosts, power devices and associated logs and alerts. *Users* can also use the Management Console, but have limited menu access to control select devices, review their logs and access them using the in-built Web terminal or control power to them.



The *console server* runs an embedded Linux operating system, and experienced Linux and UNIX users may prefer to undertake configuration at the command line. You can command line access by cellular / dial-in or directly connecting to the *console server's* serial console/modem port; or by using *ssh* or *Telnet* to connect to the *console server* over the LAN (or connecting with PPTP, IPsec or OpenVPN).

Manual Conventions

This manual uses different fonts and typefaces to show specific actions:

Note Text presented like this indicates issues to take note of

Text presented like this highlights important issues and it is essential you read and take heed of these warnings

- Text presented with an arrow head indent indicates an action you should take as part of the procedure

Bold text indicates text that you type, or the name of a screen object (*e.g.* a menu or button) on the Management Console.

Italic text is also used to indicate a text command to be entered at the command line level.

Where to find additional information

The following table contains related documentation and additional sources for information.

Document	Description
Quick Start Guide	Leads you through your initial Opengear configuration
Knowledge Base	The *https://opengear.zendesk.com /* web site contains a knowledge base with technical how to articles, tech tips, FAQs and important notifications

INSTALLATION

This chapter describes how to install the *console server* hardware, and connect it to controlled devices.

2.1 Models

There are multiple families and models, each with a different number of network / serial /USB ports or power supply and wireless configurations:

Model	Serial	USB	Network	Flash	Console Port	Modem (V.92)	Wireless	Environment Sensors	RJ Pinout	Power
ACM5002-F-E	2	1	1	4GB	-	-	-	Temp/probes	02	Ext AC/DC
ACM5004-F-E	4	1	1	4GB	-	-	-	Temp/probes	02	Ext AC/DC
ACM5003-M-F-E	3	1	1	4GB	-	Internal	-	Temp/probes	02	Ext AC/DC
ACM5004-G(S/V)-E	4	1	1	-	-	-	3G	Temp/probes	02	Ext AC/DC
ACM5004-G(S/V)-I	4*	1	1	-	-	-	3G	Temp & DI/O	02	Ext AC/DC
ACM5004-2-I	4*	2	2	-	-	-	-	Temp & DI/O	02	Ext AC/DC
ACM5504-2-P	4	2	2	4GB	-	-	-	External	02	PoE
ACM5504-5-G(V)-I	4*	2	5	4GB	-	-	3G	Temp & DI/O	02	Ext AC/DC
ACM5504-5-G-W-I	4*	2	5	4GB	-	-	WAP, 3G	Temp & DI/O	02	Ext AC/DC
ACM5504-5-Lx-I	4*	2	5	4GB	-	-	4G	Temp & DI/O	02	Ext AC/DC
ACM5508-2	8	2	2	4GB	-	-	-	-	02	Ext AC/DC
ACM5508-2-I	8*	2	2	4GB	-	-	-	Temp & DI/O	02	Ext AC/DC
ACM5508-2-M	8	2	2	4GB	-	Internal	-	-	02	Ext AC/DC
ACM5508-2-L-I	8	2	2	4GB	-	-	4G	Temp & DI/O	02	Ext AC/DC
IM4248-2-DAC/DC	48	3**	2	16GB	1	Internal	3G opt****	External	00/01/02	Dual AC/DC
IM4232-2-DAC/DC	32	3**	2	16GB	1	Internal	3G opt****	External	00/01/02	Dual AC/DC
IM4216-2-DAC/DC	16	3**	2	16GB	1	Internal	3G opt****	External	00/01/02	Dual AC/DC
IM4208-2-DAC/DC	8	3**	2	16GB	1	Internal	3G opt****	External	00/01/02	Dual AC/DC
IM4216-34-DAC/DC	16	3**	34	16GB	1	Internal	-	External	02	Dual AC/DC
IM7208-2-DAC	8	2	2	16GB	1	Internal	WAP, 4G opt	External	01 & 02	Dual AC/DC
IM7216-2-DAC	16	2	2	16GB	1	Internal	WAP, 4G opt	External	01 & 02	Dual AC/DC
IM7232-2-DAC	32	2	2	16GB	1	Internal	WAP, 4G opt	External	01 & 02	Dual AC/DC
IM7248-2-DAC	48	2	2	16GB	1	Internal	WAP, 4G opt	External	01 & 02	Dual AC/DC
CM7148-2-SAC	48	2	2	4GB	1	-	-	External	02	Single AC
CM7116-2-SAC	16	2	2	4GB	1	-	-	External	02	Single AC
CM7148-2-DAC	48	2	2	4GB	1	-	-	External	02	Dual AC
CM7132-2-DAC	32	2	2	4GB	1	-	-	External	02	Dual AC
CM7116-2-DAC	16	2	2	4GB	1	-	-	External	02	Dual AC
SD4001	1*	-	1	-	-	-	-	-	DB9***	Ext AC/DC
SD4002	2*	-	1	-	-	-	-	-	DB9***	Ext AC/DC

Network ports are all 10/100 except IM7200 which has dual 10/100/1000 LAN ports (with 2 x RJ45 copper and 2 x SFP fiber module slots) and CM7100 which has dual 10/100/100 LAN ports
The serial pin-out 02 is Cisco Straight, 01 is Cisco Rolled and 00 is Opengear Classic. IM7200
* These models have RS4232/422/485 serial. All others are RS232 only
** The IM7200 has USB3.0 ports. All other models have USB2.0 ports except those marked ** which have 2x USB2.0 and 1xUSB1.1 port.
*** These models also include Cisco RJ adapters
***** IM4200 cellular models have only 1 x USB 2.0 and 1 x USB 1.1 port and no internal V.92 modem

The various product families support different software features:

SOFTWARE FEATURES	DHCP	DDNS	Mgt LAN	Cell or WiFi	OOB Failover	Auto-Response	Flash (FTP & TFTP)	FIPS	IPsec, PPTP & OpenVPN
ACM5000	yes	yes	yes**	if -W	yes	yes	yes*	yes	yes
ACM5500	yes	yes	yes**	yes***	yes	yes	yes	yes	yes
ACM7000	yes	yes	yes	yes***	yes	yes	yes	yes	yes
IM4200	yes	yes	yes	yes***	yes	yes	yes	yes	yes
IM7200	yes	yes	yes	yes***	yes	yes	yes	yes	yes
CM7100	yes	yes	yes	no	yes	yes	yes	yes	yes
SD4000	no	no	no	no	no	yes	no	no	no

* ACM5002-F-E, ACM5003-M-F-E and ACM5004-F-E models only
** ACM500x-2, ACM550x-2, ACM5504-5 models only
*** Selected models have 3G/4G cellular and/or Wi-Fi WAP

To avoid physical and electrical hazard please read Appendix C on Safety

The sections below show the components shipped with each of these models.

2.1.1 ACM5000 kit components

ACM5002-F-E, ACM5003-M-F-E, ACM5004-F-E & ACM5004-2-I Remote Site Manager (plus –SDC options and -G/GV/GS models with cellular support)

Part # 440016 — 2 x Cable UTP Cat5 blue

Part # 3190014 and 3190015 — Cisco Connector DB9F-RJ45 straight and DB9F-RJ45 cross-over

Part # 4500XX — Power Supply 12VDC 1.0A Wall mount

Part #539000 — Quick Start Guide

- Unpack your ACM5000 kit and verify you have all the parts shown above, and that they all appear in good working order. The ACM5004-G has an external 3G aerial to be attached.

- Proceed to connect your ACM5000 to the network, the serial ports of the controlled servers and AC power as shown below

2.1.2 ACM5500 kit components

ACM5504-5-G/GV-W-I, ACM5504-5-G/GV-I, ACM5504-5-LA/LR/LV-I, ACM5504-2-P, ACM5508-2, ACM5508-2-L-I, ACM5508-2-M and ACM5008-2-P Remote Management Gateway

Part # 440016 — 2 x Cable UTP Cat5 blue

Part # 3190014 and 3190015 — Cisco Connector DB9F-RJ45 straight and DB9F-RJ45 cross-over

Part # 4500-- — Power Supply 12VDC 1.0A Wall mount

Part #539000 — Quick Start Guide

- Unpack your ACM5500 kit and verify you have all the parts shown above, and that they all appear in good working order
- The ACM5004-5-G(V)-I and ACM5504-5-LA/LR/LV-I models come with an external cellular aerial to be attached. The ACM5004-5-G(V)-W-I also has an external 802.11 wireless aerial to be attached
- Proceed to connect your ACM5500 to the network, serial and USB ports of the controlled devices, environmental monitors and AC power as shown below

2.1.3 IM4208-2, IM4216-2, IM4232-2, IM4248-2 and IM4216-34 kit components

Part # 509006 — IM4216-2 Infrastructure Manager
Part # 509007 — IM4248-2 Infrastructure Manager
Part # 509008 — IM4208-2 Infrastructure Manager
Part # 509009 — IM4216-34 Management Gateway

Part # 440016 — 2 x Cable UTP Cat5 blue

Part # 319000 and 319001 — Connector DB9F-RJ45S straight and DB9F-RJ45S cross-over

Part # 440001 — Dual IEC AC power cord (DAC models only)

Part # 539001 — Quick Start Guide

- Unpack your IM4200 (IM4208-2, IM4216-2, IM4232-2, IM4248-2 Infrastructure Manager or IM4216-34 Management Gateway) kit and verify you have all the parts shown above, and that they all appear in good working order
- If you are installing your IM4200 in a rack you will need to attach the rack mounting brackets supplied with the unit, and install the unit in the rack. Take care to head the Safety Precautions listed in Appendix C
- Proceed to connect your IM4200 to the network, to the serial ports of the controlled devices, and to power as outlined below

Note The IM4216-2-DDC, IM4232-2-DDC, IM4248-2-DDC and IM4216-34-DDC products are DC powered and the kits do not include an IEC AC power cord

2.1.4 IM7208-2, IM7216-2, IM7232-2 and IM7248-2 kit components

IM7248-2-DAC, IM7232-2-DAC, IM7216-2-DAC and IM7208-2-DAC Infrastructure Managers (and -LA/LR/LV models with 4G LTE)

Part # 440016 2 x Cable UTP Cat5 blue

Part # 319014, 319015, 319016, 319017, 319018 and 319019 Cisco Straight and Rolled Connectors DB9M/F -RJ45

Part # 440001 Dual IEC AC power cord (DAC models only)

Part # 539001 Quick Start Guide

- Unpack your IM7200 kit and verify you have all the parts shown above, and that they all appear in good working order
- If you are installing your IM7200 in a rack you will need to attach the rack mounting brackets supplied with the unit, and install the unit in the rack. Take care to head the Safety Precautions listed in Appendix C
- Proceed to connect your IM7200 to the network, to the serial ports of the controlled devices, and to power as outlined below

Note The IM7208-2-DDC, IM7216-2-DDC, IM7232-2-DDC and IM7248-2-DDC products are DC powered and the kits do not include an IEC AC power cord

2.1.5 CM7116-2, CM7132-2 and CM7148-2 kit components

CM7116-2 Console Server, CM7132-2 Console Server and CM7148-2 Console Server

Part # 440016 2 x Cable UTP Cat5 blue

Part # 319014 and 319015	Cisco Connector DB9F-RJ45 straight and DB9F-RJ45 cross-over
Part # 440001	Dual IEC AC power cord
Part # 539001	Quick Start Guide

- Unpack your CM7116-2 (or CM7132-2/CM7148-2) kit and verify you have all the parts shown above, and that they all appear in good working order
- If you are installing your CM7116-2 (or CM7132-2/CM7148-2) in a rack you will need to attach the rack mounting brackets supplied with the unit, and install the unit in the rack. Take care to head the Safety Precautions listed in Appendix C
- Proceed to connect your CM7116-2 (or CM7132-2/CM7148-2) to the network, to the serial ports of the controlled devices, and to power as outlined below

2.1.6 SD4002 kit components

Part # 509005	SD4002 Device Server
Part # 440016	2 x Cable UTP Cat5 blue
Part # 319017 and 319018	Connector DB9F-RJ45S straight and DB9F-RJ45S cross-over
Part # 4500XX	Power Supply 12VDC 1.0A Wall mount
Part # 539000	Quick Start Guide

- Unpack your SD4002 and verify you have all the parts shown above, and that they all appear in good working order
- Proceed to connect your SD4002 to the network, to the serial port of the controlled device and to power as outlined below

2.1.7 SD4001 kit components

Part # 509068	SD4001 Serial Device Server
Part # 319018	Connector DB9F to RJ45 crossover
Part # 450026	Universal Input 12 VDC Wall mount Power Supply

Part # 539000 Quick Start Guide

- Unpack your SD4001 and verify you have all the parts shown above, and that they all appear in good working order
- Proceed to connect your SD4001 to the network, to the serial port of the controlled device and to power as outlined below

2.2 Power Connection

2.2.1 All IM7200, IM4200 and CM7100 -DAC models

These standard IM7200, IM4200 and CM7100 *console servers* all have dual universal AC power supplies with auto failover built in. These power supplies each accept AC input voltage between 100 and 240 VAC with a frequency of 50 or 60 Hz and the total power consumption per *console server* is less than 30W.

Two IEC AC power sockets are located at the rear of the metal case, and these IEC power inlets use conventional IEC AC power cords. Power cords for various regions are available, although the North American power cord is provided by default. There is a warning notice printed on the back of each unit.

To avoid electrical shock the power cord grounding conductor must be connected to ground

2.2.2 SD4002 and SD4001 power

The SD4002 and SD4001 models are each supplied with an external DC wall mount power supply.

A specific power supply models for each region will have been supplied (as specified by the –US, -EU, -UK –JP or –AU extension to the part number)

The 12V DC connector from the power supply unit plugs into the DC power socket on the side of the console server casing

- Plug in the power supply AC power cable and the DC power cable
- Turn on the AC power and confirm the console server Power LED (*PWR*) is lit.

Note: When you first apply power to the SD4002 you will observe the Local and Serial LEDs flashing alternately)

The SD4002 can also be powered directly from any +9V DC to +48V DC power source by connecting the DC power lines to the IN-GND and IN-VIN+ screw jacks.

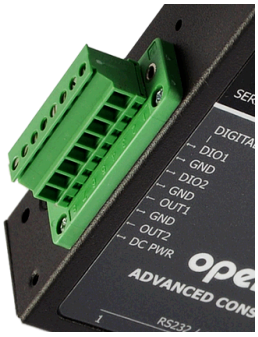
2.2.3 All ACM5000 and ACM7000 models

All the ACM5000 models are supplied with an external AC-12VDC wall mount power supply. This comes with a selection of wall socket adapters for each geographic region (North American, Europe, UK, Japan or Australia). The 12V DC connector from the power supply unit plugs into the 12VDC (*PWR*) power jack on the side of the console server casing

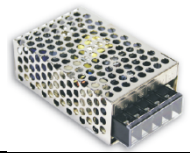
- Plug in the power supply AC power cable and the DC power cable
- Turn on the AC power and confirm the *console server* Power LED (*PWR*) is lit

The ACM5000 models can also be powered from an external +9V DC to +30V DC power source - by connecting the DC power lines to a power plug that plugs into the 12VDC *(PWR*) jack.

The industrial ACM5004-2-I model and ACM7000 models also can be powered externally by connecting a +9 to +30V DC power source to the *DC PWR* and *GND* connectors on the green screw terminal block on the side of the unit.



Note



All ACM5000 models can also be ordered with the -SDC option. These units are supplied with an external DC-DC power converter. This converter has an integrated power cable/connector that plugs into the *12VDC (PWR*) connector on the ACM5000. The input voltage for the DC-DC converter is plus or minus 36V DC to 72V DC

2.2.4 All ACM5500 models

All the ACM5500 models are supplied with an external AC-12VDC wall mount power supply. This comes with a selection of wall socket adapters for each geographic region (North American, Europe, UK, Japan or Australia). The 12V DC connector from the power supply unit plugs into the 12VDC (*PWR*) power jack on the side of the console server casing

- Plug in the power supply AC power cable and the DC power cable
- Turn on the AC power and confirm the console server Power LED (*PWR*) is lit

The ACM5500 models can also be powered from an external +9V DC to +30V DC power source - by connecting the DC power lines to a power plug that plugs into the 12VDC *(PWR*) jack.

Similarly the ACM5500 can be powered by connecting an external 9V AC to 24V AC power source to this jack.

The industrial ACM5508-2-I and ACM5504-5-G-I models also can be powered externally by connecting a +9 to +30V DC power source to the *EXT 9-30V DC* and *GND* connectors on the green screw terminal block on the side of the unit.

The ACM5504-2-P can be PoE powered using 802.3af compliant power sources.

Note

An external DC-DC power converter can be ordered as an accessory with any ACM5500 remote management gateway. This converter has an integrated power cable/connector that plugs into the *12VDC (PWR*) connector on the ACM5500. The input voltage for the DC-DC converter is plus or minus 36V DC to 72V DC

2.2.5 IM7200-DDC and IM4200-DDC power

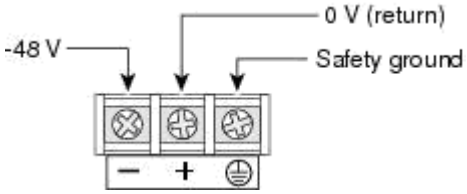
The -DDC model *console servers* all have dual DC power supplies with auto failover built in. To connect to the DC input supply:

- Strip the DC wire insulation to expose approximately 0.4 inch (10 mm) of conductor
- Connect the safety ground wire to the '**E**' safety ground terminal on the terminal block first. The DDC is floating (w.r.t. Earth), however the safety terminal on the three way screw terminal block connects to Earth or Chassis Ground
- Connect the power wires to the appropriate terminals of the terminal block:

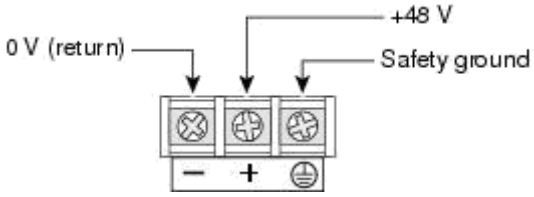
  The '**+**' terminal on the four way screw terminal block should always be connect to the more positive voltage (from 0V to +48 V)

  The '**-**' terminal on the four way screw terminal block should connect to the more negative voltage (from -48V to 0V)

  So the connections for -48 Volt DC input power are:



  The connections for -48 Volt DC input power are:



- Tighten the terminal screw to a torque of 8.0 ± 0.5 in-lb (0.93 ± 0.05 N-m)
- Repeat the connection steps above for the second power supply
- Turn on the DC power

The safety covers are an integral part of the DDC product. Do not operate the unit without the safety cover installed.

Any exposed wire lead from a DC-input power source can conduct harmful levels of electricity. So ensure that no exposed portion of the DC-input power source wire extends from the terminal block plug and safety cover

2.3 Network Connection

With the exception of the IM7200 family, all console servers have Ethernet LAN ports with RJ45 connectors. These RJ45 ports are located on the front panel of the rack-mount IM4200 units, the rear panel of the rack-mount CM7100 units, and on the side of the smaller ACM7000, ACM5500, ACM5000 and SD4001/2 units. All physical connections are made using industry standard Cat5 cabling and connectors. Ensure you only connect the LAN port to an Ethernet network that supports 10/100, or 10/100/1000 (IM7200, CM7100, ACM7000 only).

The IM7200 has four physical input ports that are logically bundled as two ports (*NET1 & NET2*). Each logical port consists of a copper 10/100/1000 copper port and a fiber-optic small form-factor pluggable (SFP) module slot. Within each port, you can use only one of the two physical ports, either the SFP module port or the 10/100/1000 port e.g. either the*NET1 (SFP)* module port or the 10/100/1000 port *NET1 (C)*. The default operation is that if the SFP module is plugged in, the fiber-optic medium has the priority over copper medium. If the SFP module is not plugged in, then the copper medium becomes active. If the SFP module is plugged in later (even after the copper medium establishes the link), then the link of the copper medium will be disconnected and the fiber-optic medium will become active.

For the initial configuration of the *console server* you must connect a computer to the *console server*'s principal network port. This port is labeled *NET1* (on IM7200 and CM7100), *NETWORK1* (on IM4200), *LAN1* (on ACM7000, ACM5500 and SD4000) *and LAN USB1* (on ACM5000).

2.4 Serial Port Connection

Console servers all come with one to forty eight serial ports, marked *SERIAL* or *SERIAL PORTS*. These ports connect to serially Managed Devices. Each *console server* also has either a dedicated Local Console (or modem) port marked *LOCAL* or *CONSOLE,* or one or its SERIAL ports can be software configured in Local Console mode. This Local Console port can be used for local command line access (or external serial modem out of band connection).

- All *console server* models except the SD4001, ACM5000, ACM5500 and ACM7000 have a dedicated local RS232 Console port. This is a DB9 connector located on the front of the IM4200 models, and a RJ45 connector (Cisco Straight) located on the front of the IM7200 and CM7100 models
- ACM5000, ACM5500 and ACM7000 models have has two, three, four or eight SERIAL PORTS presented as RJ45 ports 1-8. Port 1 on all these models by default is configured in Local Console mode
- The SD4002 has two DB9 serial ports (Ports 1-2). By default Port 1 is configured in Local Console (modem) mode. Similarly the SD4001 has one DB9 serial port and by default it is configured in Local Console (modem) mode

Conventional Cat5 cabling with RJ45 jacks is generally used for serial connections. Opengear supplies a range of cables and adapters that may be required to connect to the more popular servers and network appliances.

These are also overviewed in *Appendix D - Connectivity and Serial I/O*. More detailed information is available online at http://www.opengear.com/cabling.html

Before connecting the console port of an external device to the *console server* serial port, confirm that the device does support the standard RS-232C (EIA-232).

The *console servers* come with one to forty eight serial connectors for the RS232 serial ports:

- The SD4001 and SD4002 models have DB9 serial port connectors. All other models have RJ45 serial port connectors
- The RJ45 serial ports are located on the front face of the ACM5000 and ACM5500; on the rear of the ACM7000; on the front panel of the rack mount IM4200; and on the rear panel of the rack mount IM7200

- The ACM5000, ACM5500, ACM7000, CM7100 and IM4216-34 models have Cisco Straight serial pinouts on the RJ45 connectors (refer 2.4.3 below)
- All serial ports on the IM7200 are RJ45 and are software selectable for Cisco Straight or Cisco Rolled pinout
- The IM4200 family is available with a selection of alternate RJ45 pinouts e.g. the IM4208-2, IM4216-2 and IM4248-2 *console servers* have three RJ45 pinout configurations available - Opengear Classic, Cisco Straight or Cyclades/Cisco Rolled (refer 2.4.1)

  These alternate pinouts need to be specified in the part number at the time of order e.g. to order an IM4248-2 dual power supply AC USA model, specify:

  - *IM4248-2-DAC-X2-US* for a unit equipped with Cisco Straight RJ pinouts (straight through cable)
  - *IM4248-2-DAC-X1-US* for a unit equipped with Cyclades/Cisco Rolled RJ pinouts (rolled cable connection)
  - *IM4248-2-DAC-X0-US* for a unit equipped with Opengear Classic RJ pinouts

Some *console server* models support RS-422 and RS-485 as well as RS-232:

- The four RJ45 serial ports on the ACM5004-2-I and ACM5504-5-G-I are each RS-232/422/485 software selectable – as are the eight RJ45 serial ports on the ACM5508-2-I
- The SD4002 has one DB9 RS-232 serial port (Port 1) and one DB9/connector block RS-232/422/485 software selectable serial port (Port 2)
- Similarly the SD4001 has one DB9 RS-232 serial port which can be hardware selected to be RS-232 or RS422/485
- Refer *Appendix D - Connectivity and Serial I/O* for RS422/485 pinout and connection details

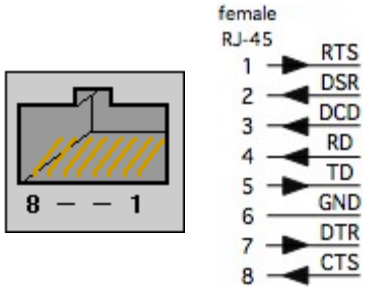
So in summary:

Model	Serial Port					Dedicated Console / Modem port
	#	Connectors	Pinout	RS232	RS422/485	
ACM500x	2,3,4	RJ	X2 Cisco	Y	N	N*
ACM5004-I	4	RJ	X2 Cisco	Y	Y	N*
ACM550x	4,8	RJ	X2 Cisco	Y	N	N*
ACM550x-I	4,8	RJ	X2 Cisco	Y	Y	N*
ACM700x	4	RJ	X2 Cisco	Y	N	N*
IM72xx-2	16,32,48	RJ	X2 Cisco or (SW config) Cisco Rolled	Y	N	Y
IM42xx-2	8,16,32,48	RJ	X0 Classic or X1 Cisco Rolled or X2 Cisco	Y	N	Y
IM4216-34	16	RJ	X2 Cisco	Y	N	Y
CM71xx	16,32,48	RJ	X2 Cisco	Y	N	Y
SD4001	1	DB9	DB9	Y	Y	N*
SD4002	2	DB9	DB9	Y	Y(1 port)	N*

*The first serial port can be reassigned to be a console/modem port

2.4.1 Opengear Classic RJ45 pinout (option –X0)

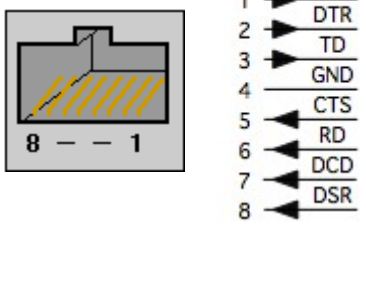
The IM4200 *console servers* are also available with this RJ45 pinout as an option:



PIN	SIGNAL	DEFINITION	DIRECTION
1	RTS	Request To Send	Output
2	DSR	Data Set Ready	Input
3	DCD	Data Carrier Detect	Input
4	RXD	Receive Data	Input
5	TXD	Transmit Data	Output
6	GND	Signal Ground	NA
7	DTR	Data Terminal Ready	Output
8	CTS	Clear To Send	Input

2.4.2 Cisco Rolled (Cyclades) RJ45 pinout (option -X1)

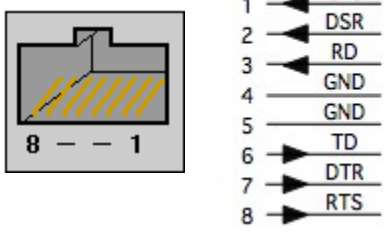
The IM4200 *console servers* are available with this RJ45 pinout option. The IM7200 *console servers* can select this pinout. This makes it easy to replace Avocent Cyclades products, and is convenient for use with rolled RJ45 cable:



PIN	SIGNAL	DEFINITION	DIRECTION
1	RTS	Request To Send	Output
2	DTR	Data Terminal Ready	Output
3	TXD	Transmit Data	Output
4	GND	Signal Ground	NA
5	CTS	Clear To Send	Input
6	RXD	Receive Data	Input
7	DCD	Data Carrier Detect	Input
8	DSR	Data Set Ready	Input

2.4.3 Cisco RJ45 pinout (option -X2)

The ACM5000, ACM5500, ACM7000, CM7100 and IM4216-34 models have *Cisco* serial pinouts on its RJ45 connectors. The IM4200 *console servers* are also available with this RJ45 pinout. The IM7200 *console servers* can select this pinout (it is the default). This provides straight through RJ45 cable to equipment such as Cisco, Juniper, Sun, and many more:



PIN	SIGNAL	DEFINITION	DIRECTION
1	CTS	Clear To Send	Input
2	DSR	Data Set Ready	Input
3	RXD	Receive Data	Input
4	GND	Signal Ground	NA
5	GND	Signal Ground	NA
6	TXD	Transmit Data	Output
7	DTR	Data Terminal Ready	Output
8	RTS	Request To Send	Output

2.5 USB Port Connection

Most *console server* models have external USB ports. For the IM7200 these are USB3.0 and for others and these ports are mostly USB2.0. They can be used for:

- connecting to UPS or PDU managed devices (e.g. for managing UPS supplies)
- connecting an external USB memory stick
- connecting to Cisco USB consoles

Some *console server* models also have a USB1.1 port and this is best reserved for use with an external USB memory stick dedicated to recovery firmware boot images / extended log file storage etc.

- All the IM4200-X models with internal cellular have one USB1.1 port on the front face and one USB 2.0 port at the rear face.

  This USB2.0 port uses a micro-AB USB connector so an adapter cable is also included. These models also have 16GB flash installed internally via a USB 2.0 flash drive for improved logging
- All the other models in the IM4200-X family (IM42xx-2-DxC-Xx models such as IM4208-2-DAC-X0, IM4248-2-DDC-X2 and IM4216-34-DAC-X2) have one USB1.1 port on the front face and two additional USB 2.0 ports at the rear face (adjacent to modem jack). These IM4200-X models also have an internal 16GB flash drive

Some *console server* models also internal USB connections to cellular modem and/or flash memory.

- The ACM5500 models all have an internal 4GB USB flash drive as well as two unallocated external USB2.0 ports
- The ACM7000 models all have an internal 4GB USB flash drive as well as four unallocated external USB2.0 ports
- The ACM5000 models have two USB2.0 ports. However one or both of these may be pre-allocated internally. For example the ACM5004-G has one internal USB committed for the cellular modem adapter, so there is only one external USB port free. Similarly with ACM5004-F-E model an internal USB flash is fitted, using up one of the two USB2.0 ports

2.6 Fitting Cellular SIM and Antennas

The ACM5504-5-G-W-I, ACM5504-5-G-I, ACM5004-G-E and ACM5004-G-I models each have an internal 3G cellular modem that requires at least one (or more) SIM cards to be installed and at least one external cellular antenna to be attached. The ACM5000-GV/GS and ACM5500-GV/GS models also have an internal cellular modem requiring external antenna connection. However the Verizon and Sprint 3G networks do not require a SIM card.

Similarly the IM4200-2-DAC-X2-G and IM4216-34-DAC-X2-G models have an internal 3G cellular modem that requires a SIM card and external antenna. The IM4200-2-DAC-X2-GV/GS and IM4216-34-DAC-X2-GV/GS models also have an internal cellular modem requiring external antenna connection however they do not require a SIM card.

The ACM5504-5-LA/LR/LV-I, ACM7004-2-LA/LR/LV and IM7200-LA/LR/LV-I models all have an internal 4G LTE cellular modem that requires at least one SIM card to be installed and two external cellular antennas to be attached.

The ACM5504-5-G-W-I and all IM7200 models also have an internal 802.11 wireless modem that requires at least one external WiFi antenna to be attached.

For more detail:

2.6.1 ACM5004 -G models

The ACM5000 -GV/-GS/-G work with Verizon USA, Sprint USA and global GSM carriers including AT&T USA, respectively. Your carrier will provide you with a SIM card for activating you data plan (-G models only). You must install the SIM card **before powering on the device**.

Unscrew the cover plate on the side of the insert the SIM into the SIM garage then screw the cover plate back on.

Take care to inset with contacts facing upwards as shown.

Screw the provided antenna on to the *MAIN* SMA antenna connector on the rear of the unit. Then place the unit and/or aerial in a location that will ensure the best signal.

These models come with dual SMA antenna connectors. The AUX connector can be used for receive diversity. This requires an external antenna (accessory Part# 569006) and cable (Part# 449041).

With -I models, the AUX connector can also be used for GPS. An external GPS passive antenna with magnetic base, SMA connector and 2 meter cable is available (Part # 569008).

Note These models has two cellular status LEDs. The SIM LED on top of unit should go on solid when powered and a SIM card has been inserted and detected.

The WWAN LED on top of unit should go on at a fast blink once a radio connection has been established with your cellular carrier (i.e. after an APN has been properly configured). WWAN LED Status:

Off:	In reset mode or not powered.
Slow blink:	Searching for service.
Solid Green:	Active service with no traffic detected.
Fast Blink:	Active service with traffic (blink rate is proportional to traffic detected)

2.6.2 ACM5500 -G models

The ACM5500 -GV/-GS/-G work with Verizon USA, Sprint USA and global GSM carriers including AT&T USA, respectively. Your carrier will provide you with a SIM card for activating you data plan (-G models only). You must install the SIM card **before powering on the device**.

-G-I models can hold two SIM cards from alternate carriers, however only requires one SIM to operate. Unscrew the SIM card access panel and insert the first carrier SIM card in the **bottom** SIM slot. A second carrier SIM can also be installed in the slot above the first. Screw the cover plate back on.

Take care to insert the SIM cards with contacts facing downward and the notch to RHS.

Screw the provided cellular antenna on to the main *Cell (M)* connector on the rear of the ACM5504-G-I.

Then place the unit and/or aerial in a location that will ensure the best signal. The ACM5504-5-G-I has a second SMA antenna connector. This *Cell (A)* connector can be used for receive diversity. This requires an external antenna (accessory Part# 569006) and cable (Part# 449041).

The ACM5504-5-G-I has a second SMA antenna connector. This *Cell (A)* connector can be used for receive diversity. This requires an external antenna (accessory Part# 569006) and cable (Part# 449041).

Alternately, the *Cell (A)* connector can be used for GPS. An external GPS passive antenna with magnetic base, SMA connector and 2 meter cable is available (Part # 569008).

The ACM5504-5-G(V)-W-I models have an internal 802.11 WiFi adapter and come with an external WiFi antenna. Screw wireless antenna on to the main *WIFI (M)* connector.

The ACM5504-5-G(V)-W-I has a second WiFi antenna connector. This *WIFI (A)* connector can be used for diversity and requires an external antenna (part # 569011).

2.6.3 ACM5500 -L models

The ACM5500 -LA/-LV/-LR models work with AT&T USA, Verizon USA, or global 4G LTE carriers respectively. Your carrier will provide you with a SIM card for activating you data plan. -LR models can hold two SIM cards from alternate carriers, however only one SIM is required.

You must install the SIM card before powering on the device.

Unscrew the SIM card access panel and insert the first carrier SIM card in the **bottom** SIM slot. A second carrier SIM can also be installed in the slot above the first.

Double check you **inserted the SIM card in the bottom SIM slot with contacts facing downward** and the notch to RHS. Then replace the SIM card access panel.

ACM5500-L models are supplied with two external 7-band cellular antennas. Screw the provided antennas on to the main *Cell (M)* and diversity *Cell (A)* SMA connectors on the rear panel.

An external GPS passive antenna with magnetic base, SMA connector and 2 meter cable is available (Part # 569008). It is screwed on to the *GPS* SMA connector on the rear panel.

2.6.4 ACM7000 -L models

The ACM7004-2-LA/LV/LR models work with AT&T USA, Verizon USA, and global 4G LTE carriers respectively. Your carrier will provide you with a SIM card for activating you data plan.

You must install the SIM card before powering on the device.

Unscrew the SIM card access panel and insert the first carrier SIM card in the **bottom** SIM slot. A second carrier SIM can also be installed in the slot above the first.

Double check you **inserted the SIM card in the bottom SIM slot with contacts facing downward** and the notch to RHS. Then replace the SIM card access panel.

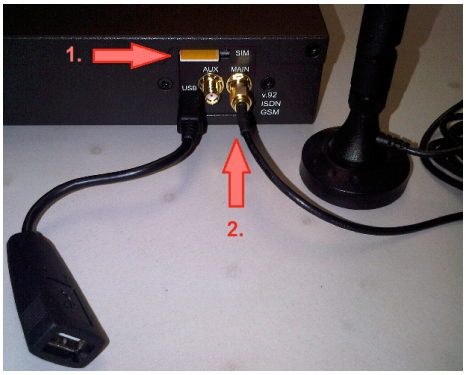
ACM7000 -L models are supplied with two external 7-band cellular antennas. Screw the provided antennas on to the main *Cell (M)* and diversity *Cell (A)* SMA connectors on the rear panel.

An external GPS passive antenna with magnetic base, SMA connector and 2 meter cable is available (Part # 569008). It is screwed on to the *GPS* SMA connector on the rear panel.

2.6.5 IM4200 -G models

The IM4200-2-DAC-X2/X0-G and IM4216-34-DAC-X2-G models have an internal 3G-GSM HSUPA/UMTS cellular modem (and an internal 16GB flash memory and an additional USB port at the rear). They are also supplied with an external antenna with extension cable, and a USB adapter cable. They work with global GSM carriers.

Before powering on the *console server*:



- Your carrier will provide you with a SIM card. Insert the SIM card (1). It will lock into place

  Take care to insert SIM with contacts facing downward

- Screw the external antenna coax cable onto the *MAIN* screw mount SMA connector on the rear of the *console server (2)*

- The *AUX* connector can be used either for receive diversity (requires external antenna Part# 569006 and cable Part# 449041) or for GPS (requires external GPS passive antenna with cable Part# 569008).

The IM4200-2-DAC-X2/X0-GV/GS and IM4216-34-DAC-X2-GV/GS models also have an internal cellular modem (and an internal 16GB flash memory and an additional USB port at the rear). They do not require a SIM card, but the supplied external antenna is installed as above. These models work with Verizon USA and Sprint USA respectively.

2.6.6 All IM7200 models

All the IM7200 models have an internal 802.11 WiFi adapter and come with an external WiFi antenna.

Before powering on the IM7200:

- Screw wireless antenna on to the *WIFI (MAIN)* SMA connector

- The IM7200 has a second WiFi antenna connector. This *WIFI (AUX)* connector can be used for diversity and requires an external antenna (part # 569022)

2.6.7 IM7200 -L models

The IM7200 -LA/-LV/-LR has a SIM card slot and three SMA cellular antenna connectors (for cellular with receive diversity and GPS). These models work with AT&T USA, Verizon USA and global 4G LTE carriers respectively. Included in the kit are two cellular antennas (with one 10 foot coaxial cable and magnetic antenna screw mount base for mounting outside the rack). If cellular signal strength is an issue, higher gain and directional antennas can be sourced.

Before powering on the IM7200 -LA/-LV/-LR:

- Screw one antenna (or antenna cable) onto the *CELL (MAIN)* screw mount (**1**) and the diversity antenna, onto the *CELL (AUX)* connector.

Note: If you have purchased a GPS antenna, screw it on to *GPS*

- Your carrier will provide you with a SIM card. Insert the card into the *SIM CARD* slot and it will lock into place (**2**)

  Take care to insert SIM card with contacts facing downwards

2.7 Digital I/O and Environmental Sensors

Any ACM5000 or ACM5500 model with an -I in the model number, or any ACM5000 with the -E option, or any ACM7000, all ship with an external green connector block for attaching environmental sensors and digital I/O devices.

Plug in this block and screw in any external devices.

On the ACM5508-2-I, ACM5504-5-G-I, ACM5504-5-LA/LR/LV-I, ACM5004-2-I, ACM5004-G-I and ACM7000 models this block can also be used for connecting the external DC power source.

Refer Chapter 8 for further details.

SYSTEM CONFIGURATION

This chapter provides step-by-step instructions for the initial configuration of your *console server*, and connecting it to the Management or Operational LAN. This involves the *Administrator*:

- activating the Management Console
- changing the Administrator password
- setting the IP address *console server*'s principal LAN port
- selecting the services to be enabled and access privileges

This chapter also discusses the communications software tools that the *Administrator* may use in accessing the *console server*, and the configuration of the additional LAN ports.

Note For guidance on configuring large numbers of Opengear appliances and/or automating provisioning, please consult the sections entitled *Bulk Provisioning* and *Zero Touch Provisioning*.

3.1 Management Console Connection

Your *console server* comes configured with a default IP Address 192.168.0.1 Subnet Mask 255.255.255.0

- Directly connect a Computer to the *console server*

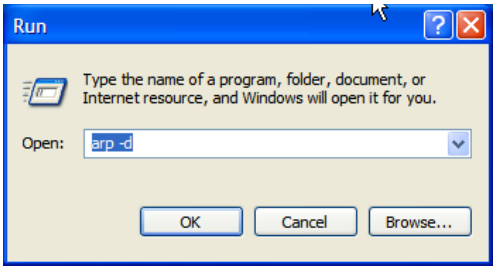
Note For initial configuration it is recommended that the *console server* be connected directly to a single Computer. However, if you choose to connect your LAN before completing the initial setup steps, it is important that:

- you ensure there are no other devices on the LAN with an **address of 192.168.0.1**
- the *console server* and the computer are on the same LAN segment, with no interposed router appliances

3.1.1 Connected computer set up

To configure the *console server* with a browser, the connected PC/workstation should have an IP address in the same range as the *console server* (for example, 192.168.0.100):

- To configure the IP Address of your Linux or Unix computer simply run *ifconfig*
- For Windows PCs:
  - Click **Start** -> (**Settings** ->) **Control Panel** and double click **Network Connections** (for 95/98/Me, double click **Network**).
  - Right click on **Local Area Connection** and select **Properties.**
  - Select **Internet Protocol (TCP/IP)** and click **Properties.**
  - Select **Use the following IP address** and enter the following details:
    - IP address: **192.168.0.100**
    - Subnet mask: **255.255.255.0**
  - If you want to retain your existing IP settings for this network connection, click **Advanced** and **Add** the above as a secondary IP connection.
- If it is not convenient to change your computer network address, you can use the *ARP-Ping* command to reset the *console server* IP address. To do this from a Windows PC:
  - Click **S**tart -> R**un** (or select **All Programs** then **Accessories** then **Run**).
  - Type *cmd* and click **OK** to bring up the command line.
  - Type *arp –d* to flush the ARP cache.
  - Type *arp –a* to view the current ARP cache (this should be empty).

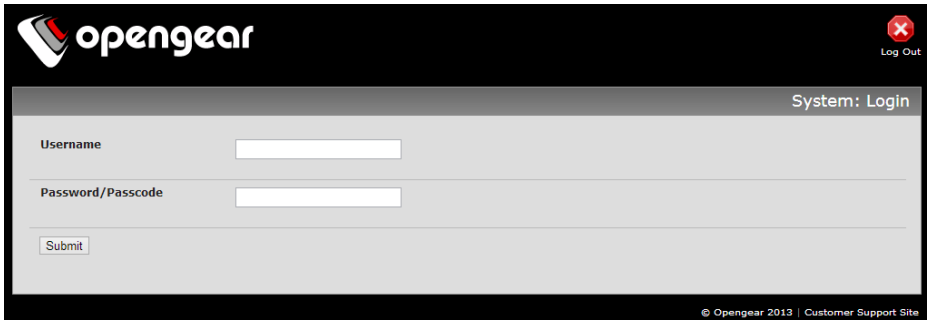


Now add a static entry to the ARP table and *ping* the *console server* to assign the IP address to the console server. In the example below, a *console server* has a MAC Address 00:13:C6:00:02:0F (designated on the label on the bottom of the unit) and we are setting its IP address to 192.168.100.23. Also the computer issuing the *arp* command must be on the same network segment as the *console server* (that is, have an IP address of 192.168.100.xxx)

- Type *arp -s 192.168.100.23 00-13-C6-00-02-0F* (Note for UNIX the syntax is: *arp -s 192.168.100.23 00:13:C6:00:02:0F*).
- Type *ping -t 192.18.100.23* to start a continuous ping to the new IP Address.
- Turn on the *console server* and wait for it to configure itself with the new IP address. It will start replying to the ping at this point.
- Type *arp –d* to flush the ARP cache again.

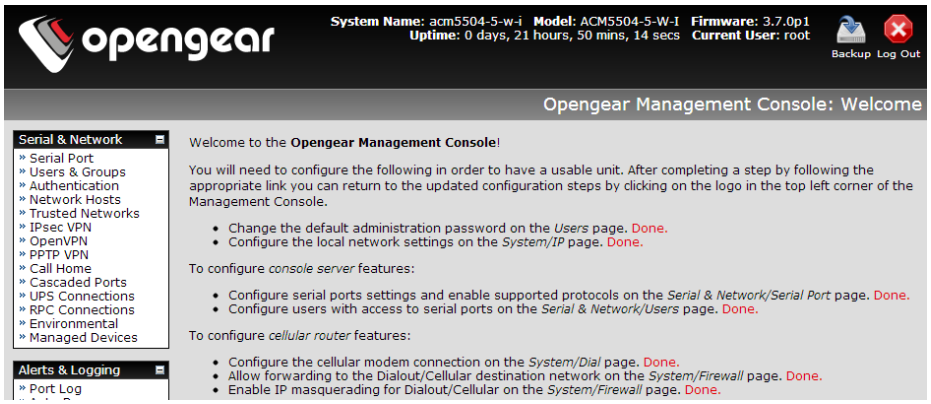
3.1.2 Browser connection

➢ Activate your preferred browser on the connected PC / workstation and enter **https://192.168.0.1** The Management Console supports all current versions of the popular browsers (Internet Explorer, Mozilla Firefox, Google Chrome, Apple Safari and more)



➢ You will be prompted to log in. Enter the default administration username and administration password (Username: **root** Password: **default**)

Note *Console server*s are factory configured with HTTPS access enabled and HTTP access disabled.



A **Welcome** screen, which lists initial installation configuration steps, will be displayed. These steps are:

Change default administration password (Users page. Refer Chapter 3.2)

- Configure the local network settings (System/IP page. Refer Chapter 3.3)

To configure console server features:

- Configure serial ports settings (Serial & Network/Serial Port page. Refer Chapter 4)
- Configure user port access (Serial & Network/Users page. Refer Chapter 4)

If your system has a cellular modem you will also be given the steps to configure the cellular router features:

- Configure the cellular modem connection (System/Dial page. Refer Chapter 5)
- Allow forwarding to the cellular destination network (System/Firewall page. Refer Chapter 5)
- Enable IP masquerading for cellular connection (System/Firewall page. Refer Chapter 5)

After completing each of the above steps, you can return to the configuration list by clicking the Opengear logo in the top left corner of the screen.

Note If you are not able to connect to the Management Console at 192.168.0.1 or if the default Username / Password were not accepted then reset your *console server* (refer *Chapter 10*)

3.2 Administrator Set up

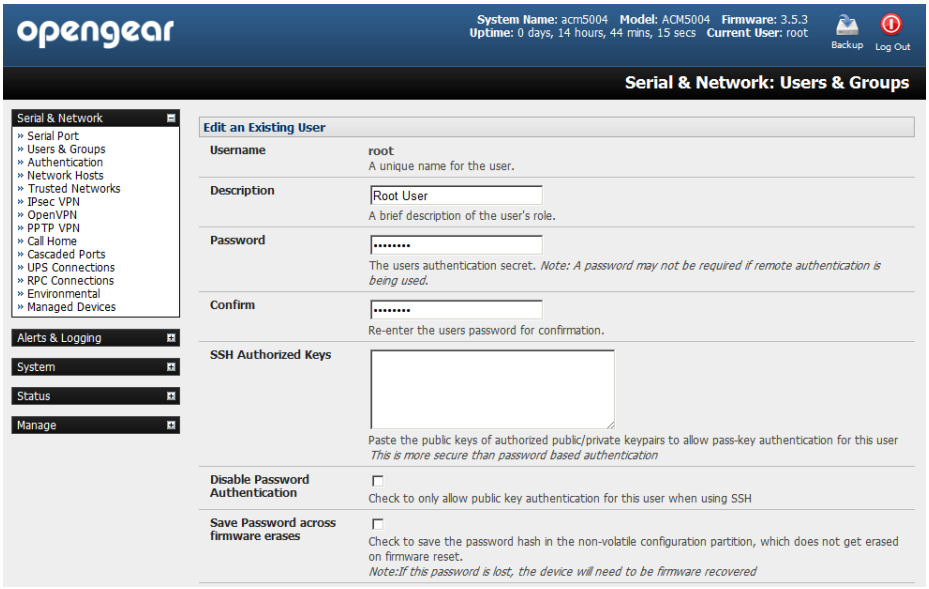
3.2.1 Change default root System Password

For security reasons, only the administration user named **root** can initially log into your *console server*. So only those people who know the root password can access and reconfigure the *console server* itself.

The corollary is that anyone who correctly guesses the root password could gain access (and the default root password is **default**). So it is essential that you enter and confirm a new password before giving the *console server* any access to, or control of, your computers and network appliances.

- Select ***Change default administration password*** from the **Welcome** page will take you to **Serial & Network: Users & Groups** where you can add a new confirmed **Password** for the user *root*

Note There are no restrictions on the characters that can be used in the user Password (which each can contain up to 254 characters). However only the first eight Password characters are used to make the *password hash*.



Note If the *console server* has flash memory (such as IM7200) you will be given the option to **Save Password across firmware erases**. Checking this will save the password hash in the non-volatile configuration partition, which does not get erased on firmware reset. However take care as if this password is lost, the device will need to be firmware recovered

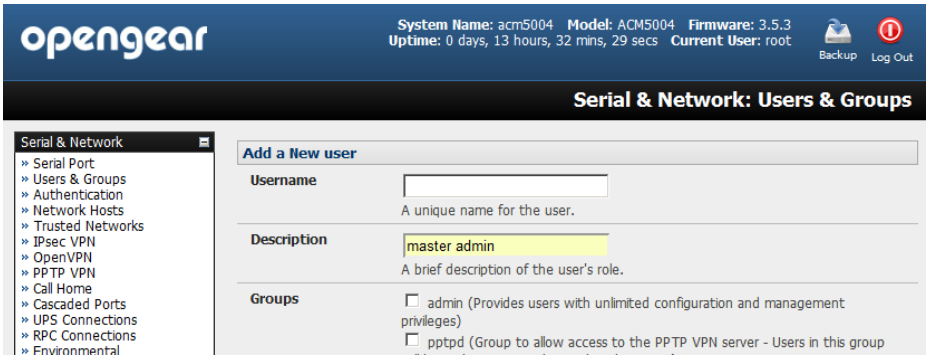
- Click **Apply**. As you have changed the password you will be prompted to log in again. This time use the new password

Note If you are not confident your *console server* has been supplied with the current release of firmware, you can upgrade. Refer *Upgrade Firmware - Chapter 11*

3.2.2 Set up a new Administrator

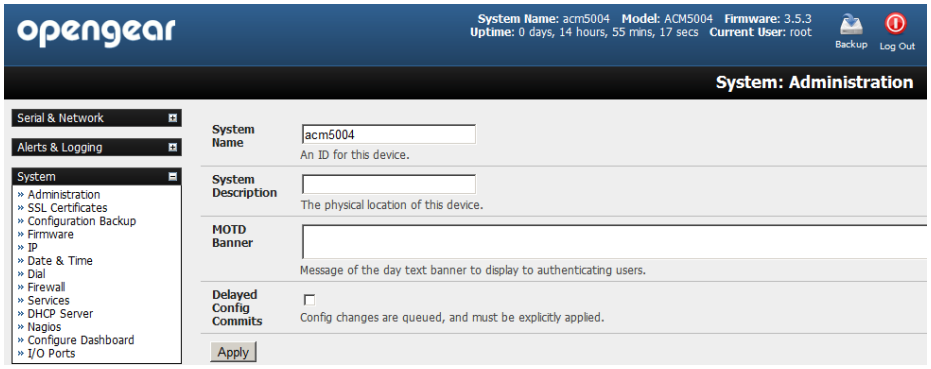
It is also recommended that you set up a new *Administrator* user as soon as convenient and log-in as this new user for all ongoing administration functions (rather than *root*).

This *Administrator* can be configured in the *admin* group with full access privileges by selecting **Add a New User** in the **Serial & Network: Users & Groups** menu (refer Ch*apter 4.2* for details)



3.2.3 Name the System

- Select **System: Administration**
- Enter a **System Name** and **System Description** for the *console server* to give it a unique ID and make it simple to identify



Note The System Name can contain from 1 to 64 alphanumeric characters (however you can also use the special characters "-" "_" and "."). There are no restrictions on the characters that can be used in the System Description (which can contain up to 254 characters).

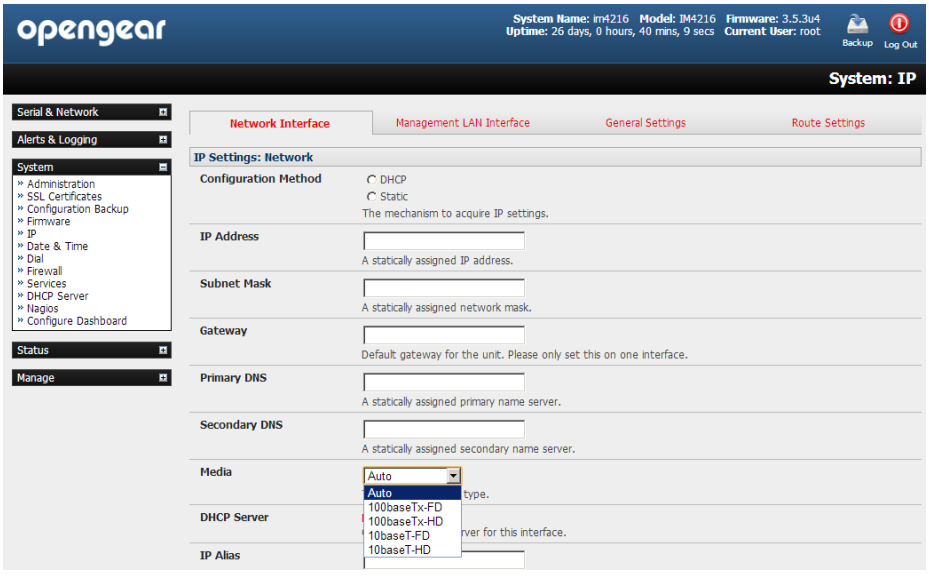
- The **MOTD Banner** can be used to display a "message of the day" text to users
- Click **Apply**

Note If you are not confident your *console server* has been supplied with the current release of firmware, you can upgrade. Refer *Upgrade Firmware - Chapter 11*

3.3 Network Configuration

The next step is to enter an IP address for the principal Ethernet (*LAN/Network/Network1*) port on the *console server*; or enable its DHCP client so that it automatically obtains an IP address from a DHCP server on the network it is to be connected to.

- On the **System: IP** menu select the **Network Interface** page then check **DHCP** or **Static** for the **Configuration Method**
- If you selected **Static** you must manually enter the new **IP Address**, **Subnet Mask**, **Gateway** and **DNS** server details. This selection automatically disables the DHCP client



- By default the *console server* LAN port auto detects the Ethernet connection speed. However you can use the **Media** menu to lock the Ethernet to 10 Mb/s or 100Mb/s and to Full Duplex (FD) or Half Duplex (HD)

Note If you encounter packet loss or poor network performance with the default auto-negotiation setting, try manually setting the Ethernet Media settings on the Opengear, and the device it is connected to. In most cases, select *100baseTx-FD* (100 megabits, full duplex). Make sure both sides are set identically.

- If you selected **DHCP** the *console server* will look for configuration details from a DHCP server. This selection automatically disables any static address. The *console server* MAC address can be found on a label on the base plate

Note In its factory default state (with no Configuration Method selected) the *console server* has its DHCP client enabled, so it automatically accepts any network IP address assigned by a DHCP server on your network. In this initial state, the *console server* will then respond to both its Static address (192.168.0.1) and its newly assigned DHCP address

- You may also enter a secondary address or comma-separated list of addresses in CIDR notation, e.g. *192.168.1.1/24* as an **IP Alias**

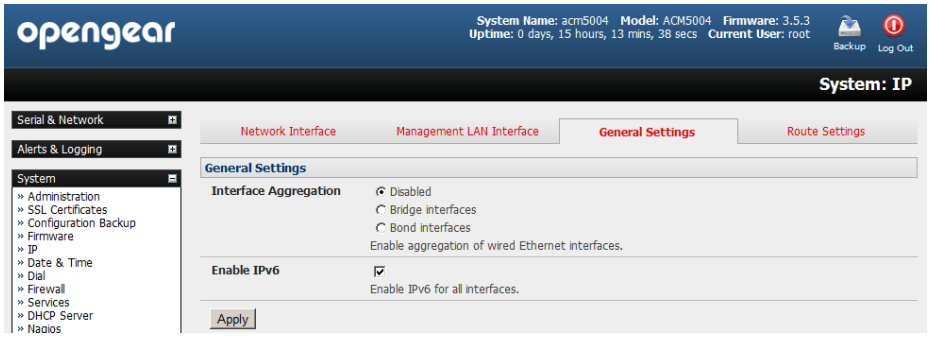
Note If you have changed the *console server* IP address, you may need to reconfigure your computer so it has an IP address that is in the same network range as this new address (as detailed in an earlier note in this chapter)

- Click **Apply**
- You will need to reconnect the browser on the computer that is connected to the *console server* by entering **http://new IP address**

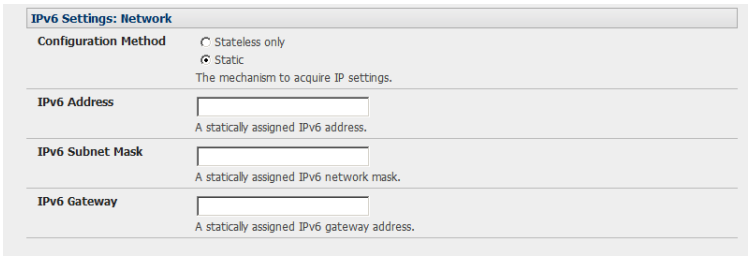
3.3.1 IPv6 configuration

By default, the *console server* Ethernet interfaces support IPv4; however, they can also be configured for IPv6 operation:

- On the **System: IP** menu select **General Settings** page and check **Enable IPv6**



- You will then need to configure the IPv6 parameters on each interface page



3.3.2 Dynamic DNS (DDNS) configuration

With Dynamic DNS (DDNS) an advanced *console server* whose IP address is dynamically assigned (and that may change from time to time) can be located using a fixed host or domain name.

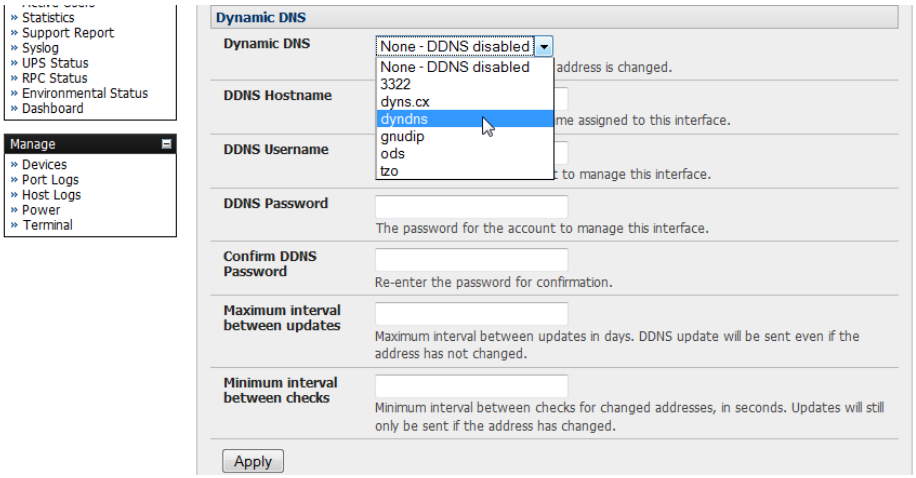
- The first step in enabling DDNS is to create an account with the supported DDNS service provider of your choice. Supported DDNS providers include:
  - DyNS www.dyns.cx
  - dyndns.org www.dyndns.org
  - GNUDip gnudip.cheapnet.net
  - ODS www.ods.org
  - TZO www.tzo.com
  - 3322.org (Chinese provider) www.3322.org

  Upon registering with the DDNS service provider, you will select a username and password, as well as a hostname that you will use as the DNS name (to allow external access to your machine using a URL).

  The Dynamic DNS service providers allow the user to choose a hostname URL and set an initial IP address to correspond to that hostname URL. Many Dynamic DNS providers offer a selection of URL hostnames available for free use with their service. However, with a paid plan, any URL hostname (including your own registered domain name) can be used.

You can now enable and configure DDNS on any of the Ethernet or cellular network connections on the *console server* (by default DDNS is disabled on all ports):

- Select the DDNS service provider from the drop down **Dynamic DNS** list on the **System:IP** or **System:Dial** menu



- In **DDNS Hostname** enter the fully qualified DNS hostname for your console server e.g. *your-hostname.dyndns.org*
- Enter the **DDNS Username** and **DDNS Password** for the DDNS service provider account
- Specify the **Maximum interval between updates** - in days. A DDNS update will be sent even if the address has not changed
- Specify the **Minimum interval between checks** for changed addresses - in seconds. Updates will still only be sent if the address has changed
- Specify the **Maximum attempts per update** i.e. the number of times to attempt an update before giving up (defaults to 3)

3.4 Services and Service Access

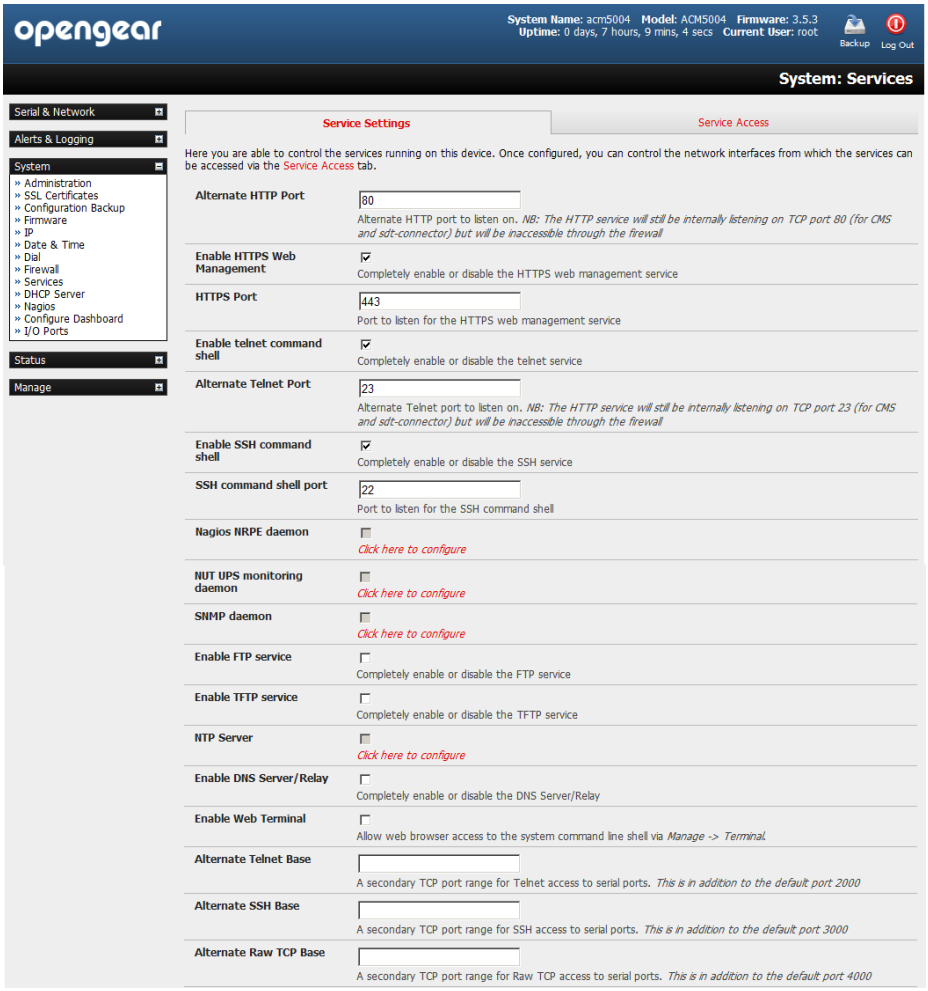
The *Administrator* can access the *console server*, and connected serial ports and managed devices, using a range of access protocols/services. For each such access:

- the particular service must first be configured and enabled to run on the *console server*
- then access through the firewall must be enabled for each network connection

To enable and configure a service:

➢ Select the **Service Settings** tab on the **System: Services** page

Note With firmware releases pre 3.5.3 services are enabled and configured using the **Service Access** tab on the **System: Firewall** page



➢ Enable and configure basic services:

HTTP By default the HTTP service is running and it cannot be fully disabled. However by default HTTP access is disabled on all interfaces and it is recommended this access remains disabled, if the console server is to be remotely accessed over the Internet.

Alternate HTTP also enables you to configure an alternate HTTP port to listen on. However the HTTP service will continue internally listening on TCP port 80 (for CMS and sdt-connector communications) but will be inaccessible through the firewall.

HTTPS By default the HTTPS service is running and this service is enabled on all network interfaces. It is recommended that only HTTPS access be used if the console server is to be managed over any public network (e.g. the Internet). This ensures the *Administrato*r has secure browser access to all the menus on the *console server*. It also allows appropriately configured *Users* secure browser access to selected *Manage* menus. For information on certificate and user client software configuration refer *Chapter 9 - Authentication*.

The HTTPS service can be completely disabled (or re-enabled) by checking **HTTPS Web Management** and an alternate port specified (default port is 443).

Telnet By default the Telnet service is running. However by default the service is disabled on all network interfaces.

Telnet can be used to give the *Administrator* access to the system command line shell. While this may be suitable for a local direct connection over a management LAN, it is recommended this service be disabled if the *console server* is to be remotely administered. This service may also be useful for local *Administrato*r and the *User* access to selected serial consoles.

The **Enable telnet command shell** checkbox will completely enable or disable the telnet service. An alternate telnet port to listen on can be specified in **Alternate Telnet Port** (default port is 23).

SSH This service provides secure SSH access to the console server and attached devices – and by default the SSH service is running and enabled on all interfaces. It is recommended you choose SSH as the protocol where the *Administrator* connects to the *console server* over the Internet or any other public network. This will provide authenticated communications between the SSH client program on the remote computer and the SSH sever in the *console server*. For more information on SSH configuration refer *Chapter 9 - Authentication*.

The **Enable SSH command shell** checkbox will completely enable or disable this service. An alternate SSH port to listen on can be specified in **SSH command shell port** (default port is 22).

- Enable and configure other services:

TFTP/FTP If a USB flash card or internal flash is detected on an advanced *console server* (ACM5000, ACM5500, ACM7000, CM7100, IM7200 or IM4200) then checking **Enable TFTP (FTP) service** will enable this service and set up default *tftp* and *ftp* server on the USB flash. These servers are used to store config files, maintain access and transaction logs etc. Files transferred using tftp and ftp will be stored under */var/tmp/usbdisk/tftpboot.* Unchecking **Enable TFTP (FTP) service** will completely disable the TFTP (FTP) service.

DNS Relay Checking **Enable DNS Server/Relay** will enable the DNS relay feature so clients can be configured with the *console server's* IP for their DNS server setting, and the *console server* will forward the DNS queries to the real DNS server.

Web Terminal Checking **Enable Web Terminal** will allow web browser access to the system command line shell via **Manage -> Terminal**.

- Specify alternate port numbers for Raw TCP, direct Telnet/SSH and unauthenticated Telnet services. The *console server* uses specific default ranges for the TCP/IP ports for the various access services that *Users* and *Administrators* can use to access devices attached to serial ports (as covered in *Chapter 4 – Configuring Serial Ports*). The *Administrator* can also set alternate ranges for these services, and these secondary ports will then be used in addition to the defaults.

  The default TCP/IP **base** port address for *telnet* access is 2000, and the range for *telnet* is IP Address: Port (2000 + serial port #) *i.e.* 2001 – 2048. So if the *Administrator* were to set 8000 as a secondary base for telnet then serial port #2 on the *console server* can be telnet accessed at IP Address:2002 and at IP Address:8002. The default base for SSH is 3000; for Raw TCP is 4000; and for RFC2217 it is 5000

- A number of other services can be enabled and configured indirectly from this menu by selecting *Click here to configure:*

Nagios Access to the Nagios NRPE monitoring daemons (refer *Chapter 8)*

NUT Access to the NUT UPS monitoring daemon (refer *Chapter 10*)

SNMP This will enable *netsnmp* in the *console server*. SNMP is disabled by default (refer *Chapter 7* and *Chapter 15.5*)

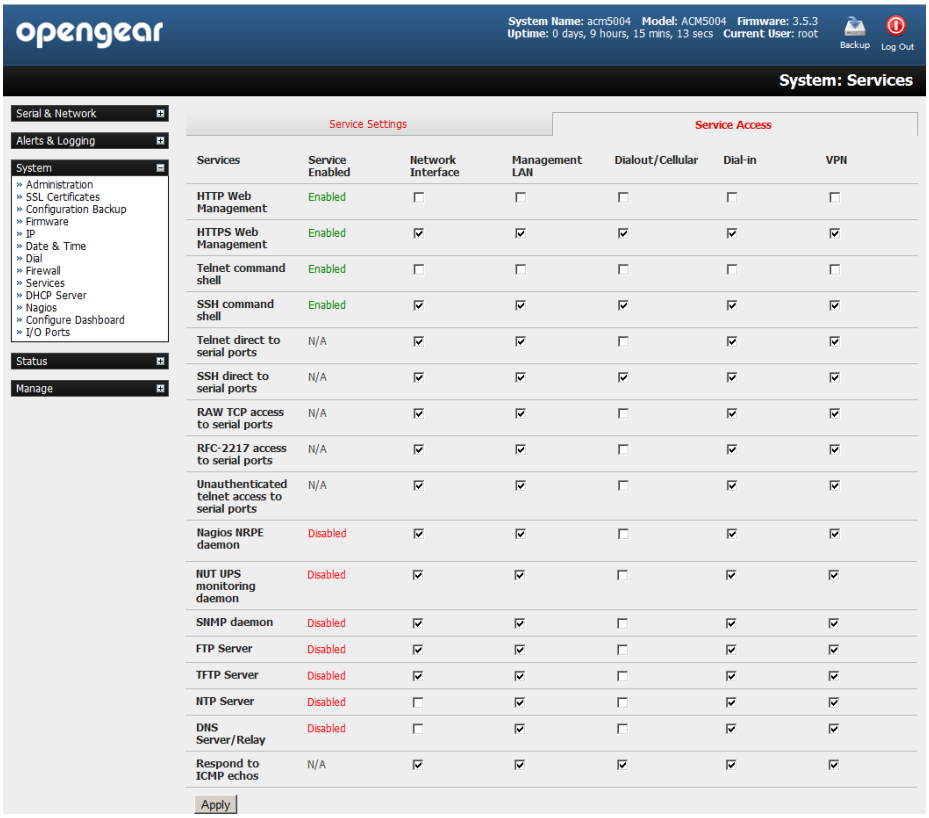
NTP Refer *Chapter 11*

- Click **Apply**. As you apply your services selections, the screen will be updated with a confirmation message: ***Message Changes to configuration succeeded***

The Services Access settings can now be set to allow or block access. This specifies which (enabled) services the *Administrator* can use over each network interface - to connect to the *console server* and through the *console server* to attached serial and network connected devices.

- Select the **Service Access** tab on the **System: Services** page.

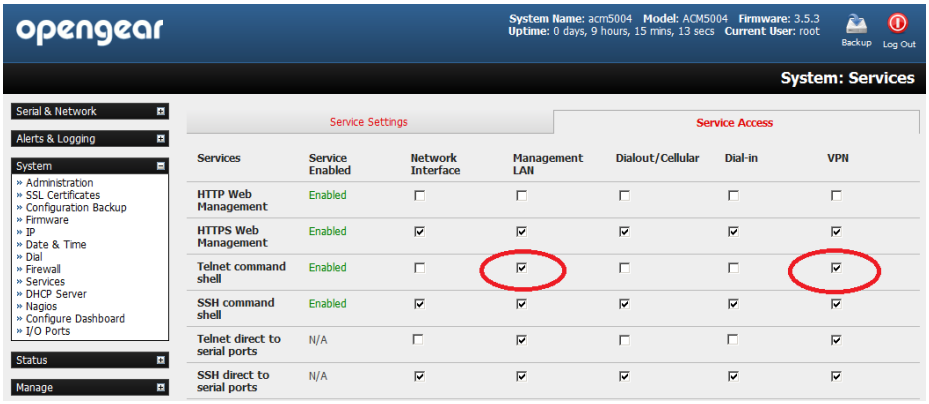
Note With firmware releases pre 3.5.3 the **Service Access** tab is found on the **System: Firewall** page



- This will display the services currently enabled for the *console server's* network interfaces. Depending on the particular *console server* model the interfaces displayed may include :
  - Network interface (for the principal Ethernet connection)
  - Management LAN / OOB Failover (second Ethernet connections)
  - Dialout/Cellular (V90 and 3G modem)
  - Dial-in (internal or external V90 modem)
  - Wi-Fi (802.11 wireless)
  - VPN (IPsec or Open VPN connection over any network interface)

- Check/uncheck for each network which service access is to be enabled /disabled

  In the example shown below local administrators on local Management LAN have Telnet access direct to the *console server* (and attached serial ports) while remote administrators using Dial In or Cellular have no such Telnet access (unless they set up a VPN).



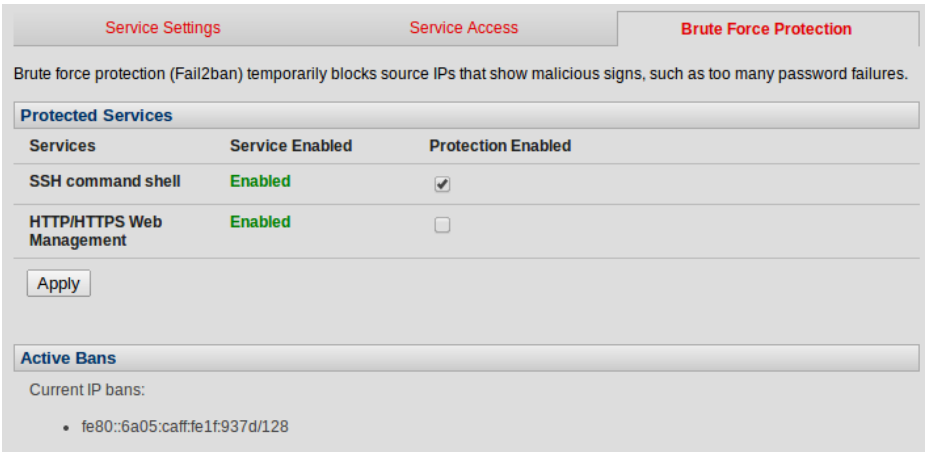
- The **Respond to ICMP echos** (i.e. *ping*) service access options that can be configured at this stage. This allows the *console server* to respond to incoming ICMP echo requests. Ping is enabled by default, however for security reasons this service should generally be disabled post initial configuration

- You can also configure to allow serial port devices to be accessed from nominated network interfaces using Raw TCP, direct Telnet/SSH, unauthenticated Telnet services etc

- Click **Apply** to apply your services access selections

3.4.1 Brute Force Protection

Brute force protection (Micro Fail2ban) temporarily blocks source IPs that show malicious signs, such as too many password failures. This may help mitigate scenarios where the Opengear device's network services are exposed to an untrusted network such as the public WAN, and scripted attacks or software worms are attempting to guess (brute force) user credentials and gain unauthorized access.



Brute Force Protection may be enabled for the listed services. Once protection is enabled, 3 or more failed connection attempts within 60 seconds from a specific source IP trigger it to be banned from connecting for the next 60 seconds. Active Bans are also listed and may be refreshed by reloading the page.

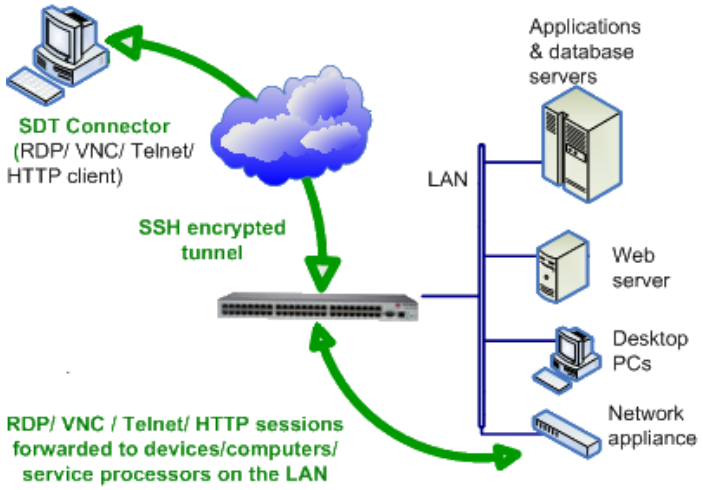
Note When an Opengear device is running on an untrusted network, it is recommended that a variety of strategies are used to lock down remote access. This includes strong passwords (or even better, SSH public key authentication), VPN, and using Firewall Rules to whitelist remote access from trusted source networks only. Please refer to the Knowledge Base for details.

3.5 Communications Software

You have configured access protocols for the *Administrator* client to use when connecting to the *console server*. *User* clients (who you may set up later) will also use these protocols when accessing *console server* serial attached devices and network attached hosts. So you will need to have appropriate communications software tools set up on the *Administrator* (and User) client's computer. Opengear provides the *SDT Connector* as the recommended client software tool, however other generic tools such as PuTTY and SSHTerm may be used, and these are all described below.

3.5.1 SDT Connector

SDT Connector is a lightweight tool that enables *Users* and *Administrators* to securely access the *Console server*, and the various computers, network devices and appliances that may be serially or network connected to the *console server*.



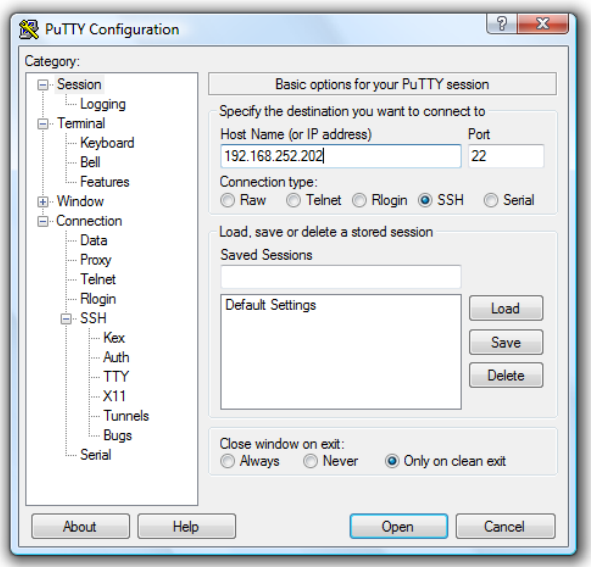
SDT Connector is a Java client program that couples the trusted SSH tunneling protocol with popular access tools such as Telnet, SSH, HTTP, HTTPS, VNC, RDP to provide point-and-click secure remote management access to all the systems and devices being managed.

Information on using *SDT Connector* for browser access to the *console server*'s Management Console, Telnet/SSH access to the *console server* command line, and TCP/UDP connecting to hosts that are network connected to the *console server* can be found in *Chapter 6 - Secure Tunneling*

SDT Connector can be installed on Windows PCs, Mac OS X and on most Linux, UNIX and Solaris systems.

3.5.2 PuTTY

Communications packages like *PuTTY* can be also used to connect to the *Console server* command line (and to connect serially attached devices as covered in *Chapter 4*). *PuTTY* is a freeware implementation of Telnet and SSH for Win32 and UNIX platforms. It runs as an executable application without needing to be installed onto your system. *PuTTY* (the Telnet and SSH client itself) can be downloaded at http://www.tucows.com/preview/195286.html

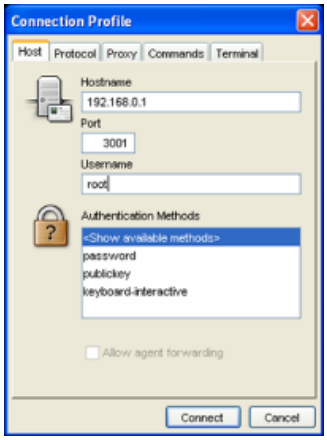


- To use PuTTY for an SSH terminal session from a Windows client, you enter the *console server*'s IP address as the 'Host Name (or IP address)'
- To access the *console server* command line you select 'SSH' as the protocol, and use the default IP Port 22
- Click 'Open' and you will be presented with the *console server* login prompt. (You may also receive a 'Security Alert' that the host's key is not cached, you will need to choose 'yes' to continue.)
- Using the Telnet protocol is similarly simple - but you use the default port 23

3.5.3 SSHTerm

Another common communications package that may be useful is *SSHTerm,* an open source package that can be downloaded from http://sourceforge.net/projects/sshtools:

- To use *SSHTerm* for an SSH terminal session from a Windows Client you simply Select the 'File' option and click on 'New Connection'



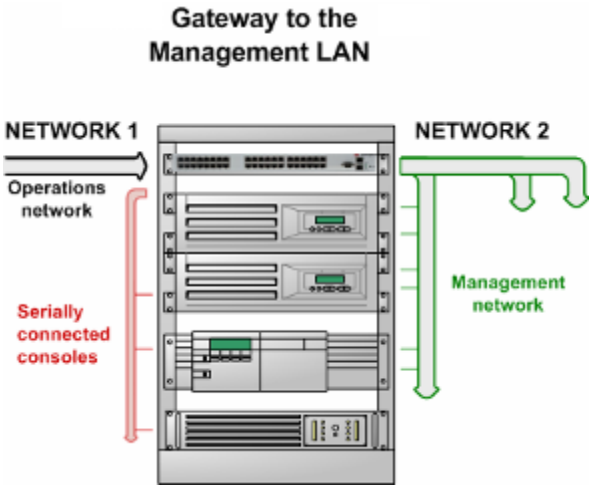
- A new dialog box will appear for your 'Connection Profile' where you can type in the host name or IP address (for the *console server* unit) and the TCP port that the SSH session will use (port 22). Then type in your username and choose password authentication and click connect.
- You may receive a message about the host key fingerprint, and you will need to select 'yes' or 'always' to continue.
- The next step is password authentication and you will be prompted for your username and password from the remote system. You will then be logged on to the *console server*

3.6 Management Network Configuration

The IM4200, IM7200, CM7100, ACM5500, ACM7000 and ACM5004-2 *console servers* have additional network ports that can be configured to provide management LAN access and/or failover or out-of-band access.

3.6.1 Enable the Management LAN

The IM4200, IM7200, CM7100, ACM7000, ACM5508-2-I/M and ACM5004-2 *console servers* can be configured so the second Ethernet port provides a management LAN gateway. The gateway has firewall, router and DHCP server features. However you need to connect an external LAN switch to *Network/LAN 2* to attach hosts to this management LAN:

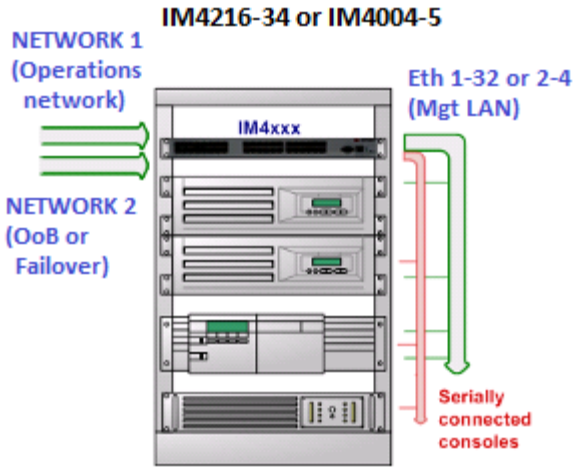


Note The second Ethernet port (Network/LAN2) on the IM4200, IM7200, CM7100, ACM7000, ACM5508-2-I/M and ACM5004-2 can be configured as either a Management LAN gateway port or it can be configured as an OOB/Failover port. It cannot be both. So ensure you did not allocate **Network/LAN 2** as the **Failover Interface** when you configured the principal **Network** connection on the **System: IP** menu.

The ACM5504-5-G-I, ACM5504-5-LA/LR/LV-I, ACM5504-5-G-W-I and IM4216-34 *console server* models have an integrated four or thirty-two port management LAN switches (with firewall, router, DHCP server and switch functions).

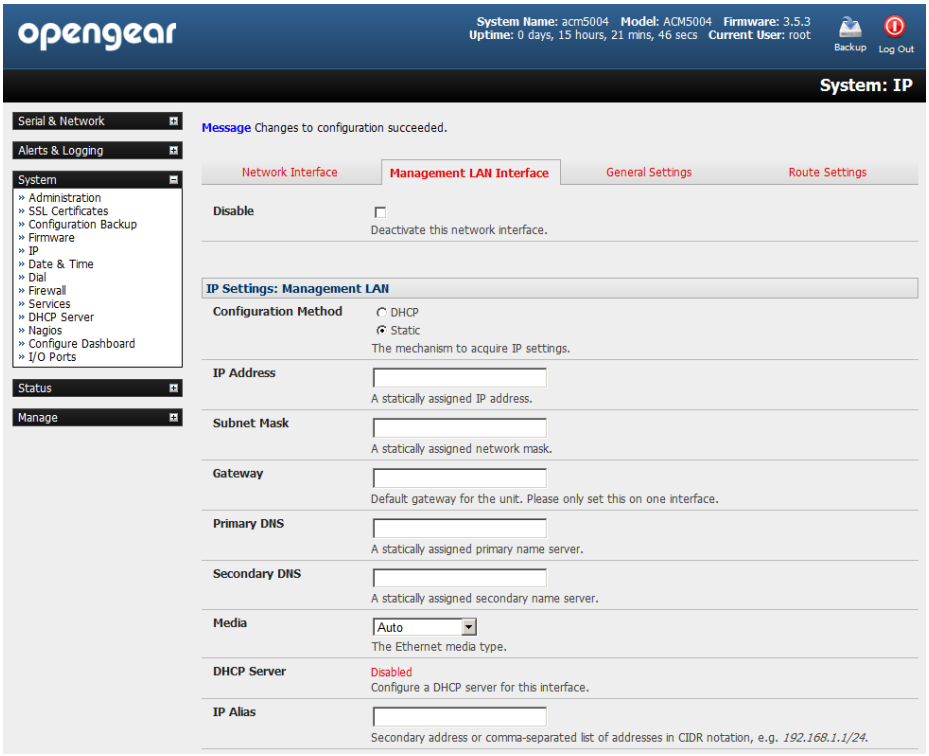
- The IM4216-34 is normally configured to have an active 32 port Management LAN (Ethernet 1-32) switch plus have Network 2 configured for OOB or Failover

- The ACM5504-5-G-W-I and ACM5504-5-G-I similarly is normally be configured with an active Management LAN. This can be a 4 port (ETH1-4) Management LAN switch, or a 3 port (ETH2-4) switch with ETH 1 configured for OOB/Failover



The above Management LAN features are all disabled by default. To configure the Management LAN gateway:

- Select the **Management LAN Interface** page on the **System: IP** menu and uncheck **Disable**
- Configure the **IP Address** and **Subnet Mask** for the Management LAN (but leave the **DNS** fields blank)
- Click **Apply**



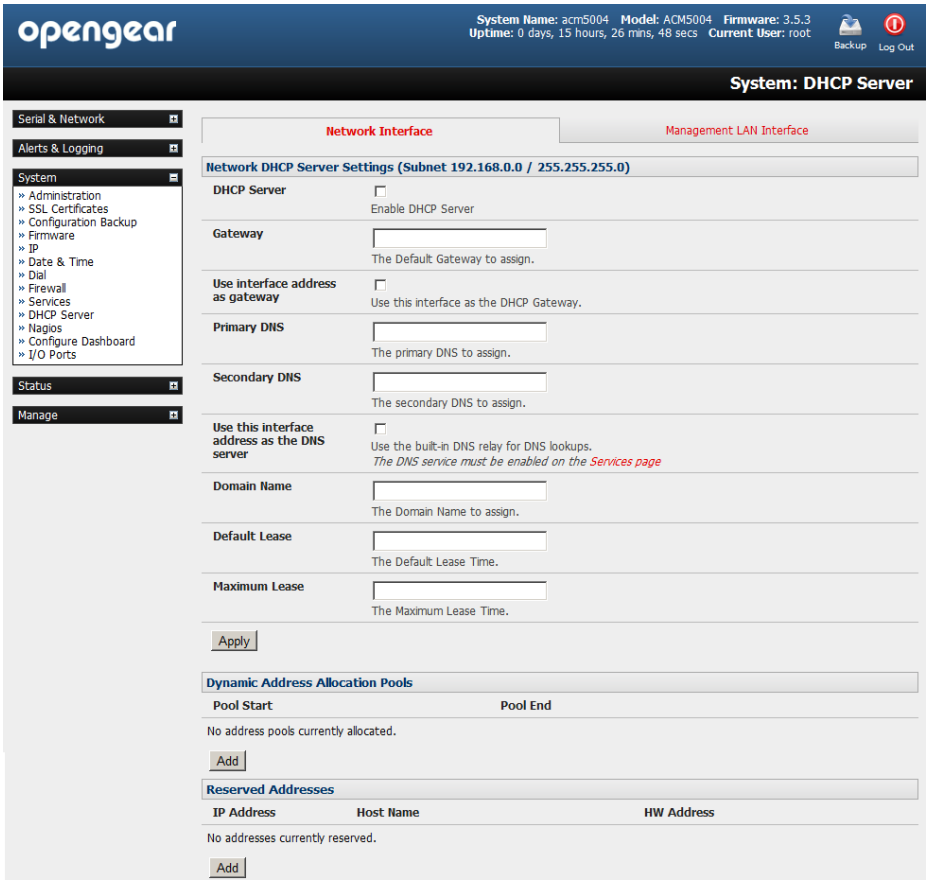
The management gateway function is now enabled with default firewall and router rules. By default these rules are configured so the Management LAN can only be accessible by SSH port forwarding. This ensures the remote and local connections to Managed Devices on the Management LAN are secure. The LAN ports can also be configured in bridged or bonded mode (as described later in this chapter) or they can be manually configured from the command line.

3.6.2 Configure the DHCP server

All IM and ACM family devices host a DHCP server, however by default it is disabled. The DHCP server enables the automatic distribution of IP addresses to devices on the Management LAN that are running DHCP clients. To enable the DHCP server:

- On the **System: IP** menu select the **Management LAN** page and click the **Disabled** label in the **DHCP Server** field (or go directly to the **System: DHCP Server** menu)

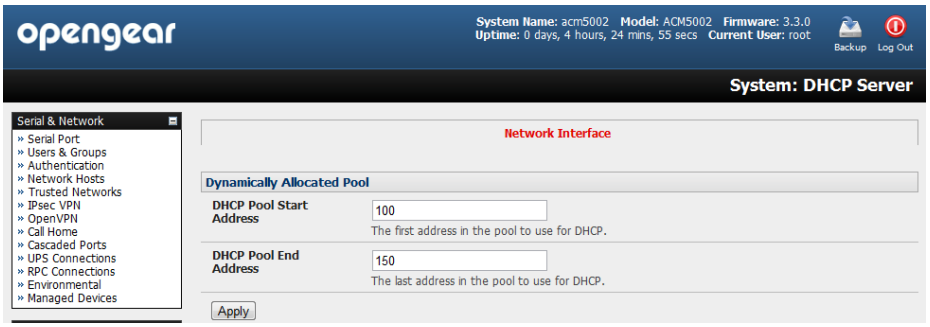
- Check **Enable DHCP Server**



- Enter the **Gateway** address that is to be issued to the DHCP clients. If this field is left blank, the *console server*'s IP address will be used
- Enter the **Primary DNS** and **Secondary DNS** address to issue the DHCP clients. Again if this field is left blank, *console server*'s IP address is used, so leave this field blank for automatic DNS server assignment
- Optionally enter a **Domain Name** suffix to issue DHCP clients
- Enter the **Default Lease** time and **Maximum Lease** time in seconds. The lease time is the time that a dynamically assigned IP address is valid before the client must request it again
- Click **Apply**

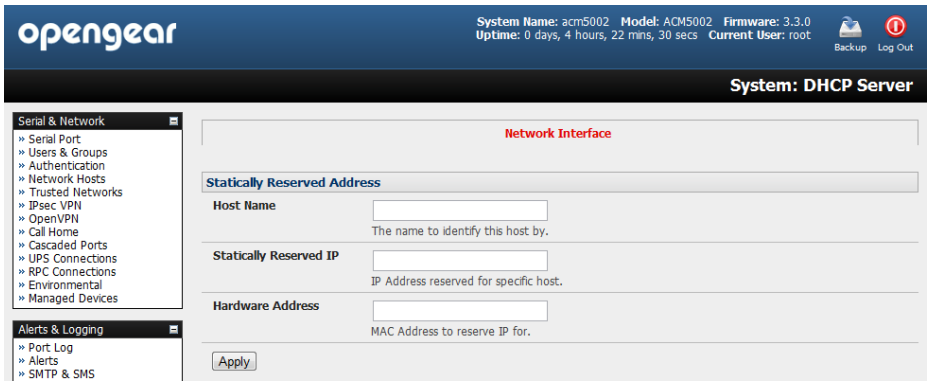
The DHCP server will sequentially issue IP addresses from a specified address pool(s):

- Click **Add** in the **Dynamic Address Allocation Pools** field
- Enter the **DHCP Pool Start Address** and **End Address** and click **Apply**



The DHCP server also supports pre-assigning IP addresses to be allocated only to specific MAC addresses and reserving IP addresses to be used by connected hosts with fixed IP addresses. To reserve an IP addresses for a particular host:

- Click **Add** in the **Reserved Addresses** field
- Enter the **Hostname,** the **Hardware Address** (MAC) and the **Statically Reserved IP** address for the DHCP client and click **Apply**



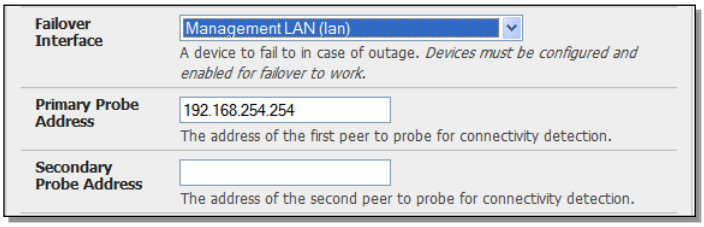
When DHCP has initially allocated hosts addresses it is recommended to copy these into the pre-assigned list so the same IP address will be reallocated in the event of a reboot.

3.6.3 Select Failover or broadband OOB

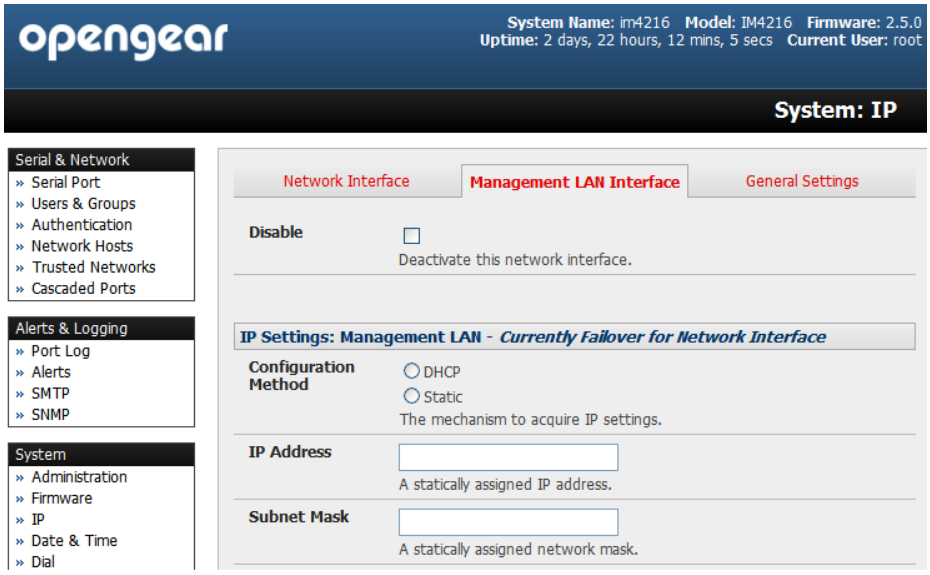
The IM4200, IM7200, CM7100, ACM7000, ACM5508-2-I/M, ACM5504-5-G-W-I, ACM5504-5-G-I, ACM5504-5-LA/LR/LV-I, and ACM5004-2 *console servers* provide a failover option so in the event of a problem using the main LAN connection for accessing the *console server*; an alternate access path is used.

By default the failover is not enabled. To enable:

- Select the **Network** page on the **System: IP** menu
- Now select the **Failover Interface** to be used in the event of an outage on the main network. This can be:
  - an alternate broadband Ethernet connection (e.g. this could be the Network/LAN2 port on most models) or
  - the IM4200 or IM7200 family internal modem or
  - an external serial modem device connected to the IM4200 or IM7200 Console port (for out-dialing to an ISP or the remote management office)



- Click **Apply**. You have selected the failover method however it is not active until you have specified the external sites to be probed to trigger failover, and set up the failover ports themselves. This is covered in *Chapter 5*.

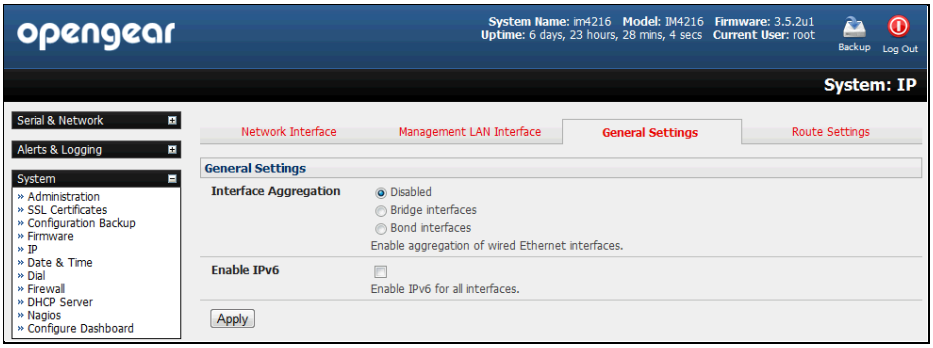


Note The ACM5504-5-G(-W)-I and IM4216-34 can be configured with an active Management LAN/gateway and with one of the switched Ethernet ports configured for OOB/Failover (ETH 1 on the ACM5504-5-G(-W)-I or NETOWRK 2 on the IM4216-34). However with the other IM4200, IM7200, CM7100, ACM7000, ACM5508-2 and ACM5004-2 models, the second Ethernet port can be configured as either a gateway port or as an OOB/Failover port, but not both. So ensure you did not enable the Management LAN function on **Network/LAN 2**

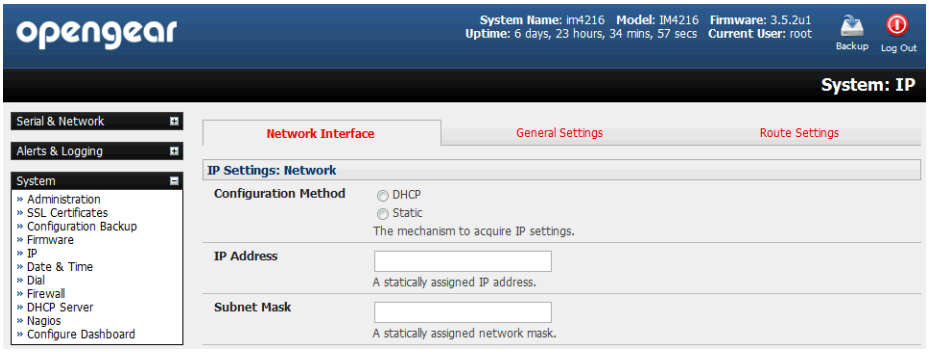
3.6.4 Aggregating the network ports

By default the *console server*'s Management LAN network ports can only be accessed using SSH tunneling /port forwarding or by establishing an IPsec VPN tunnel to the *console server*.

However all the wired network ports on the *console servers* can be aggregated by being bridged or bonded.



- By default **Interface Aggregation** is *Disabled* on the **System: IP General Settings** menu
- Select **Bridge Interfaces** or **Bond Interfaces**
  - When bridging is enabled, network traffic is forwarded across all Ethernet ports with no firewall restrictions. All the Ethernet ports are all transparently connected at the data link layer (layer 2) so they do retain their unique MAC addresses
  - With bonding he network traffic is carried between the ports but they present with one MAC address
  - Both modes remove all the **Management LAN Interface** and **Out-of-Band/Failover Interface** functions and disable the **DHCP Server**
- In *aggregation* mode all the Ethernet ports are configured collectively using the **Network Interface** menu

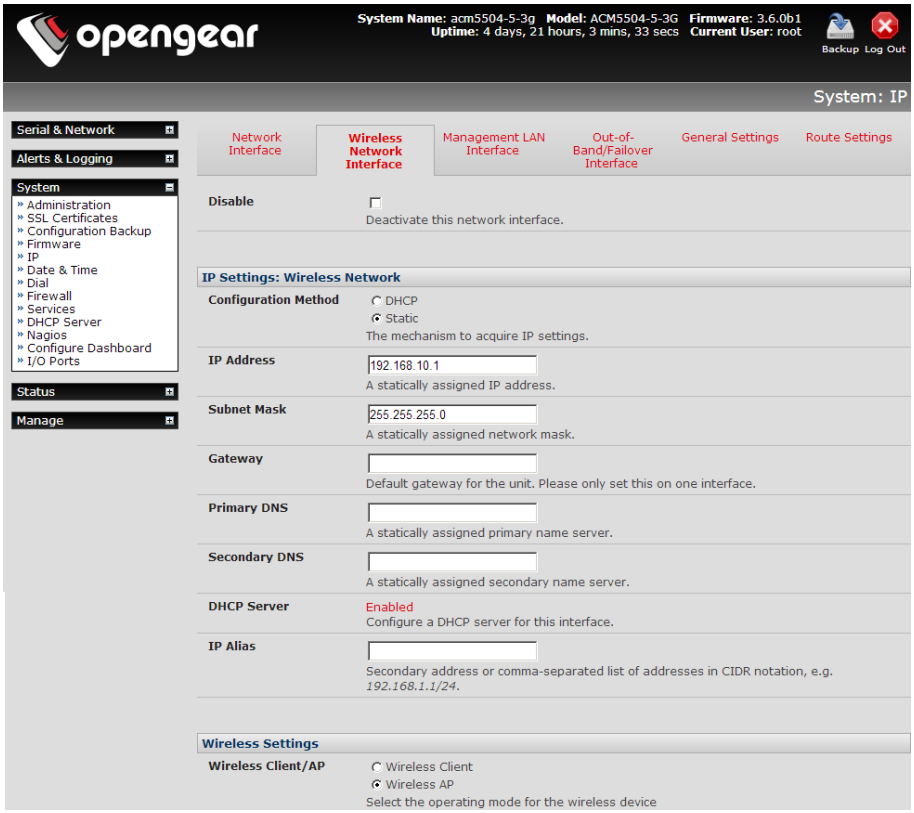


3.6.5 Wi-Fi Wireless LAN

All the IM7200 models and the ACM5504-5-G-W-I have an internal 802.11 WiFi adapter and come with an external WiFi antenna. The WiFi can be configured as a Wi-Fi Wireless Access Point (WAP) or as a Wi-Fi client. The inbuilt WiFi is inactive by default. If you wish to use the WiFi facility you will need to attach the WiFi antenna (and any auxiliary WiFi antenna you may have ordered).

Note: The custom ACM5003-W model also has an internal wireless adapter, however this can only operate as a client.

- To activate and configure the Wireless Access Point functionality, navigate to the **System: IP** page and then click the **Wireless Network Interface** tab



- Un-tick the **Disable** box

WAP configuration:

- Configure the **IP Settings** for the Wireless Network. Generally, if the device is being used as a Wireless AP, a static address is set here in the IP Settings. In this example, 192.168.10.1 is used. Set the IP address, and the netmask (in this case, 255.255.255.0 to give 254 unique network addresses in subnet), but do not fill in the Gateway, Primary DNS and Secondary DNS. These settings are used if the interface is to be the primary network link to the outside world, or if it will be used for failover.

- Select **Wireless AP**, which will make the **Wireless AP Settings** section visible:

  Country: Select the correct country from the list. If the country does not appear, select the World Regulatory Domain

  SSID: Select an SSID for the network. It should be unique.

  Broadcast SSID: Tick this to broadcast the SSID. This should generally done, disabling broadcast is not a security measure

  Network Channel: Select the network channel. 6 is most commonly used, so it is best to do a site survey and pick another channel if the unit is being deployed into an office environment

  Hardware Mode: The unit supports 802.11b,g and single band 802.11n. In most cases, selection 802.11b/g/n will provide for the best interoperability with other hardware.

  Supported Authentication Methods: Select the authentication method for the AP. If the client equipment supports it, it is always best to selecte WPA/WPA2 and AES encryption. WEP and WPA with TKIP have been proven vulnerable to cryptanalysis.

  If WEP is selected:

  WEP Mode: Select Open System or Shared System. Open System is more secure than Shared, due to the way encryption keys are used.

  WEP Key Length: Select the WEP key length. 128 bit keys offer more security, but are not supported on all devices. WEP Keys must be entered in Hexidecimal.

  WEP Key 1-4: Up to 4 WEP keys can be used on a single network.

  Default Transmit Key: This selects the default transmit key for the network

  If WPA/WPA2 is selected:

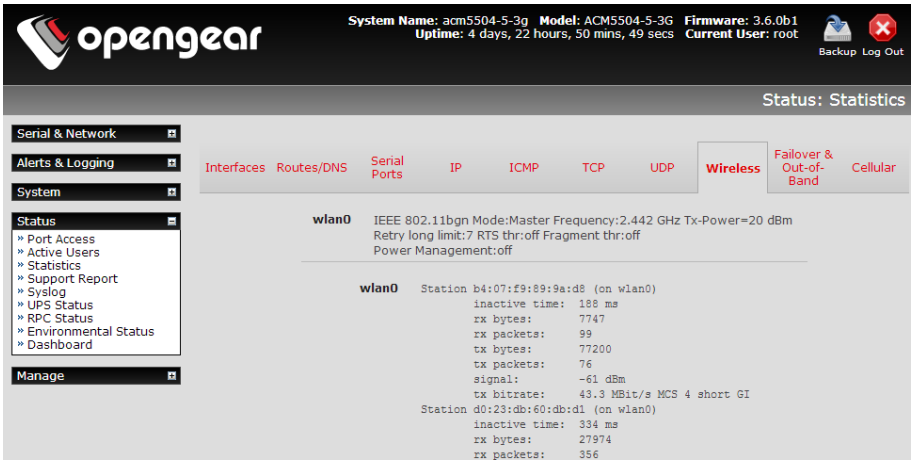
  WPA/WPA2 Encryption Methods: Select one or both of TKIP or AES for encrypting WPA/WPA2 connections. AES is more secure, and is required for the AP to advertise itself as 802.11n if that hardware mode is selected

  WPA Password: The password that clients will use to connect to the AP.

- Once the Wireless AP Settings have been filled out, click **Apply**, then wait for the page to refresh.

- The next step is to set up a DHCP server for the wireless clients. Click the link next to *DHCP Server* in the IP settings section, or go to **System: DHCP Server** page. More information on configuring DHCP can be found in Chapter 3.6.2

Note The *Wireless* screen on the *Status: Statistics* page shows the list of clients that are connected to the WAP



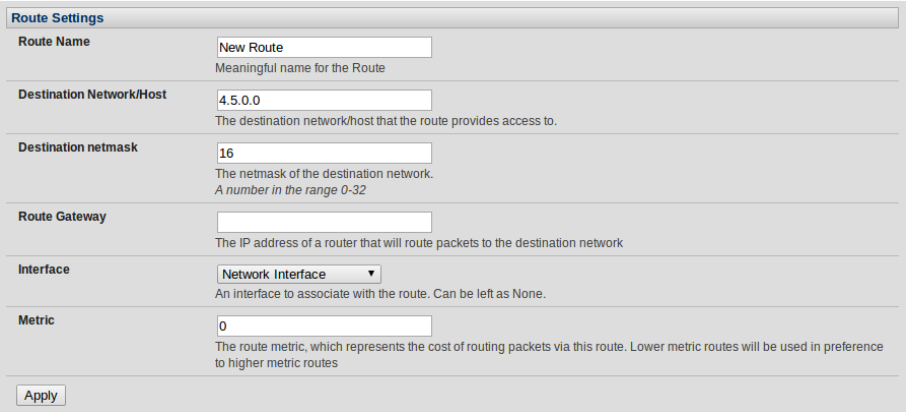
Wireless Client configuration:

- Select **Wireless Client** in the **Wireless Settings** section - which will make the **Wireless Client Settings** section visible
- Select **DHCP** or **Static** for the **Configuration Method**
    - If you selected **Static** then manually enter the new **IP Address**, **Subnet Mask**, **Gateway** and **DNS** server details. This selection automatically disables the DHCP client
    - If you selected **DHCP** the device will look for configuration details from a DHCP server on your management LAN. This selection automatically disables any static address. The device MAC address can be found on a label on the base plate
- The wireless LAN when enabled in client mode will operate as the main network connection to the device so failover is available (though it not *enabled* by default). Use **Failover Interface** to select the device to failover to in case of wireless outage and specify **Probe Addresses** of the peers to probed for connectivity detection
- Configure the Wireless Client to select the local wireless network which will serve as the main network connection to the *console server.*
    - Select the **Country** the device is to operate in
    - Enter the appropriate **SSID** (Set Service Identifier) of the wireless access point to connect to
    - Select the **Wireless Network Type** where **Infrastructure** is used to connect to an access point and **Ad-hoc** to connect directly to a computer
    - Select the **Wireless Security** mode of the wireless network (WEP, WPA etc.) and enter the required Key / Authentication / Encryption settings

Note: The *Wireless* screen in *Status: Statistics* will display all the locally accessible wireless LANs (with SSID and Encryption/Authentication settings). You can also use this screen to confirm you have successfully connected to the selected access point - refer Chapter 12

3.6.7 Static routes

Firmware 3.4 and later support *static routes* which provide a very quick way to route data from one subnet to different subnet. So you can hard code a path that specifies to the console server/router to get to a certain subnet by using a certain path. This may be useful for remotely accessing various subnets at a remote site when being accessed using the cellular OOB connection.



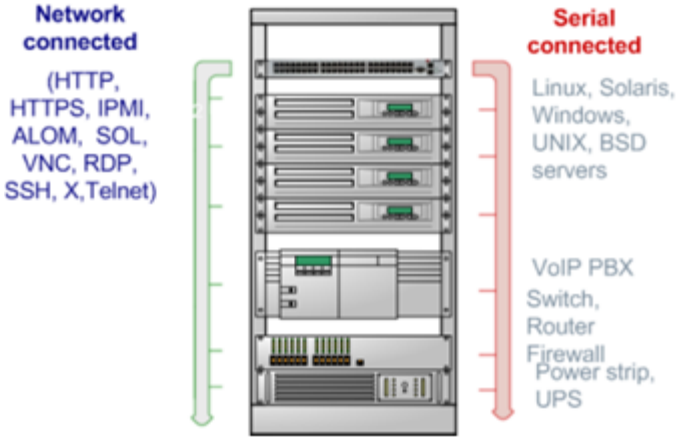
To add to the static route to the route table of the system:

- Select the **Route Settings** tab on the **System: IP General Settings** menu
- Enter a meaningful **Route Name** for the route
- In the **Destination Network/Host** field enter the IP address of the destination network/host that the route provides access to
- Enter a value in the **Destination netmask** field that identifies the destination network or host. Any number between 0 and 32. A subnet mask of 32 identifies a host route.
- Enter **Route Gateway** with the IP address of a router that will route packets to the destination network, may be left blank
- Select the **Interface** to use to reach the destination, may be left as *None*
- Enter a value in the **Metric** field that represents the metric of this connection. This generally only has to be set if two or more routes conflict or have overlapping targets. Any number equal to or greater than 0
- Click **Apply**

Note: The route details page provides a list of network interfaces and modems to which a route can be bound. In the case of a modem, the route will be attached to any dialup session which is establish via that device. A route can be specified with a gateway, an interface or both. If the specified interface is not active for whatever reason, then routes configured for that interface will not be active.

SERIAL PORT, HOST, DEVICE & USER CONFIGURATION

The *console server* enables access and control of serially-attached devices and network-attached devices (*hosts*). The *Administrator* must configure access privileges for each of these devices, and specify the services that can be used to control the devices. The *Administrator* can also set up new users and specify each user's individual access and control privileges.



This chapter covers each of the steps in configuring network connected and serially attached devices:

- *Serial Ports* – setting up protocols used serially connected devices
- *Users & Groups* – setting up users and defining the access permissions for each of these users
- *Authentication* – this is covered in more detail in Chapter 9
- *Network Hosts* – configuring access to local network connected computers or appliances (*hosts*)
- *Configuring Trusted Networks* - nominate specific IP addresses that trusted users access from
- *Cascading and Redirection of Serial Console Ports*
- *Connecting to Power (UPS PDU and IPMI) and Environmental Monitoring (EMD) devices*
- *Serial Port Redirection* – using the PortShare windows and Linux clients
- *Managed Devices* - presents a consolidated view of all the connections
- *IPSec* – enabling VPN connection
- *OpenVPN*
- *PPTP*

4.1 Configure Serial Ports

The first step in configuring a serial port is to set the **Common Settings** such as the protocols and the RS232 parameters that are to be used for the data connection to that port (e.g. baud rate).

Then you select what mode the port is to operate in. Each port can be set to support one of these operating modes:

i. *Disabled mode* is the default, the serial port is inactive

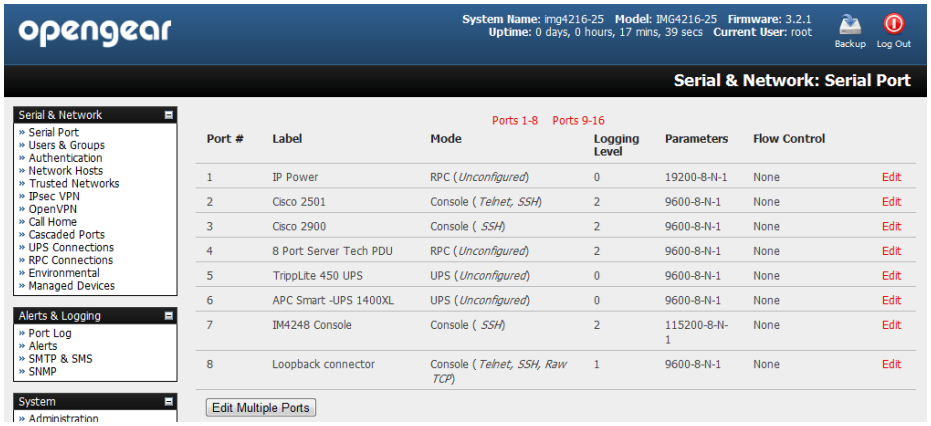
ii. *Console Server mode* enables general access to serial console port on the serially attached devices

iii. *Device mode* sets the serial port up to communicate with an intelligent serial controlled PDU, UPS or Environmental Monitor Devices (EMD)

iv. *SDT mode* enables graphical console access (with RDP, VNC, HTTPS etc.) to hosts that are serially connected

v. *Terminal Server mode* sets the serial port to await an incoming terminal login session

vi. *Serial Bridge mode* enables the transparent interconnection of two serial port devices over a network



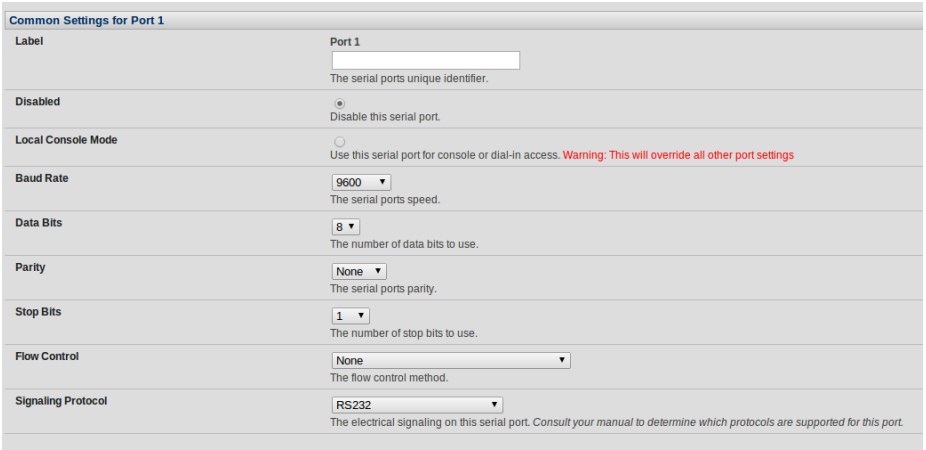
- Select **Serial & Network: Serial Port** and you will see details of the serial ports that are currently set up
- By default each serial port is set in *Console Server mode*. For the port to be reconfigured click **Edit**
- When you have reconfigured the common settings (*Chapter 4.1.1*) and the mode (*Chapters 4.1.2 - 4.1.6*) for each port, you set up any remote syslog (*Chapter 4.1.7*), then click **Apply**

Note If you wish to set the same protocol options for multiple serial ports at once click **Edit Multiple Ports** and select which ports you wish to configure as a group

- If the *console server* has been configured with distributed Nagios monitoring enabled then you will also be presented with **Nagios Settings** options to enable nominated services on the Host to be monitored (refer *Chapter 10 – Nagios Integration*)

4.1.1 Common Settings

There are a number of common settings that can be set for each serial port. These are independent of the mode in which the port is being used. These serial port parameters must be set so they match the serial port parameters on the device you attach to that port:



- Specify a label for the port
- Select the appropriate **Baud Rate**, **Parity**, **Data Bits**, **Stop Bits** and **Flow Control** for each port

- Set the **Signaling Protocol**. This menu item only presents in ports with RS422/485 options (i.e. Port 1 on SD4002 and SD4001, and all ports on ACM5004-2-I, ACM5508-2-I, ACM5504-5-LA/LR/LV-I and ACM5504-5-G-I). The options available are RS232, RS422, RS485 and RS485 Echo mode

- Set the **Port Pinout**. This menu item only presents for IM7200 ports where pin-out for each RJ45 serial port can be set as either X2 (Cisco Straight) or X1 (Cisco Rolled)

- Before proceeding with further serial port configuration, you should connect the ports to the serial devices they will be controlling, and ensure they have matching settings

4.1.2 Console Server Mode

- Select ***Console Server* Mode** to enable remote management access to the serial console that is attached to this serial port:

Logging Level This specifies the level of information to be logged and monitored (refer *Chapter 7 - Alerts and Logging)*

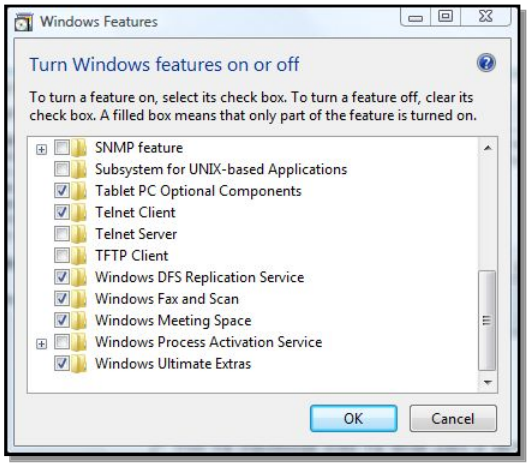
Telnet When the Telnet service is enabled on the *console server*, a Telnet client on a *User's* or *Administrator's* computer can connect to a serial device attached to this serial port on the *console server*. The Telnet communications are unencrypted so this protocol is generally recommended only for local or VPN tunneled connections.

With Win2000/XP/NT you can run *telnet* from the command prompt (*cmd.exe*). Windows Vista and later ships with a Telnet client but it is not enabled by default. You can install it by following the simple steps below.

- Click the **Start** button, click **Control Panel**, click **Programs**, and then click **Turn Windows features on or off**. If you are prompted for an administrator password or confirmation, type the password or provide confirmation.

- In the **Windows Features** dialog box, select the **Telnet Client** check box.

- Click **OK**. The installation might take several minutes.



If the remote communications are being tunneled with *SDT Connector*, then Telnet can be used for securely accessing these attached devices (refer Note below).

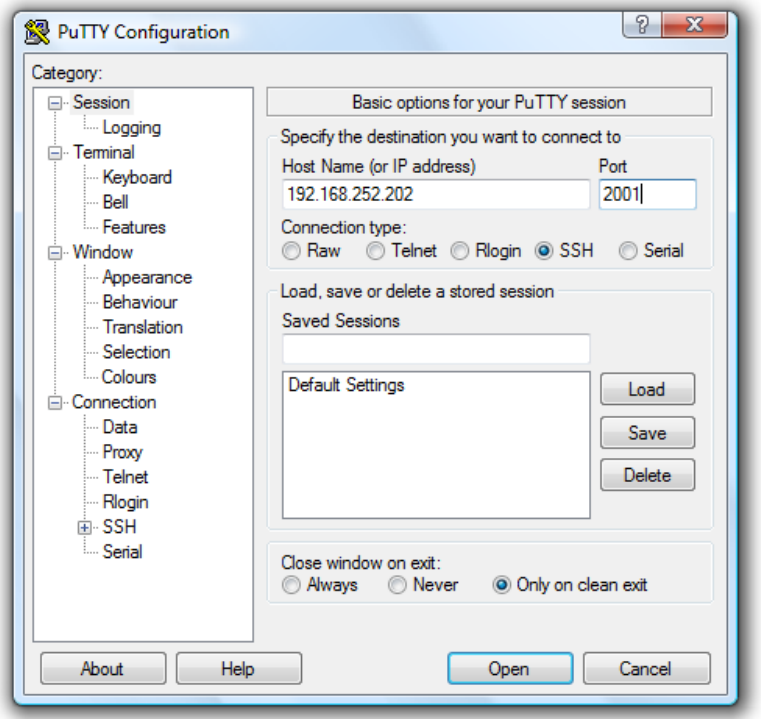
Note In *Console Server* mode, *Users* and *Administrators* can use *SDT Connector* to set up secure Telnet connections that are SSH tunneled from their client computers to the serial port on the *console server*. *SDT Connector* can be installed on Windows PCs and on most Linux platforms and it enables secure Telnet connections to be selected with a simple point-and-click.

To use *SDT Connector* to access consoles on the *console server* serial ports, you configure *SDT Connector* with the *console server* as a *gateway*, then as a *host*, and you enable Telnet service on Port (2000 + serial port #) *i.e.* 2001–2048. Refer *Chapter 6* for more details on using *SDT Connector* for Telnet and SSH access to devices that are attached to the *console server* serial ports.

You can also use standard communications packages like *PuTTY* to set a direct Telnet (or SSH) connection to the serial ports (refer Note below):

Note *PuTTY* also supports Telnet (and SSH) and the procedure to set up a Telnet session is simple. Enter the *console server's* IP address as the 'Host Name (or IP address)'. Select 'Telnet' as the protocol and set the 'TCP port' to 2000 plus the physical serial port number (*i.e.* 2001 to 2048).

Click the 'Open' button. You may then receive a 'Security Alert' that the host's key is not cached, you will need to choose 'yes' to continue. You will then be presented with the login prompt of the remote system connected to the serial port chosen on the *console server*. You can login as normal and use the host serial console screen.



PuTTY can be downloaded at http://www.tucows.com/preview/195286.html

Note In *Console Server* mode, when you connect through to a serial port you connect via *pmshell*. To will generate a BREAK on the serial port you need to type the character sequence *~b* (and if you're doing this over OpenSSH you'll need to type *~~b*)

SSH

It is recommended that you use SSH as the protocol where the *User* or *Administrator* connects to the *console server* (or connects through the *console server* to the attached serial consoles) over the Internet or any other public network. This will provide authenticated SSH communications between the SSH client program on the remote user's computer and the *console server*, so the user's communication with the serial device attached to the *console server* is secure

For SSH access to the consoles on devices attached to the *console server* serial ports, you can use *SDT Connector*. You configure *SDT Connector* with the *console server* as a *gateway*, then as a *host*, and you enable SSH service on Port (3000 + serial port #) *i.e.* 3001-3048. *Chapter 6 - Secure Tunneling* has more information on using *SDT Connector* for SSH access to devices that are attached to the *console server* serial ports.

Also you can use common communications packages, like *PuTTY* or *SSHTerm* to SSH connect directly to port address IP Address _ Port (3000 + serial port #) *i.e.* 3001–3048

Alternately SSH connections can be configured using the standard SSH port 22. The serial port being accessed is then identified by appending a descriptor to the username. This syntax supports any of:

<username>:<portXX>

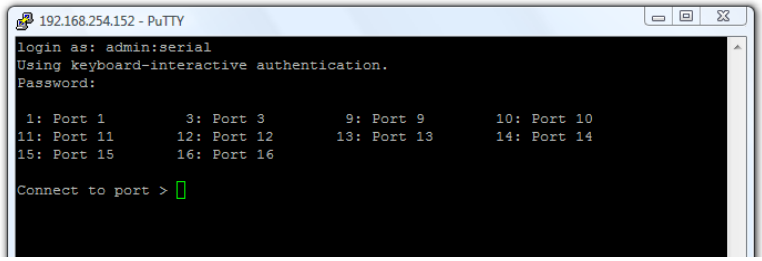
<username>:<port label>

<username>:<ttySX>

<username>:<serial>

So for a *User* named 'fred' to access serial port 2, when setting up the SSHTerm or the PuTTY SSH client, instead of typing *username = fred* and *ssh port = 3002*, the alternate is to type *username = fred:port02 (*or *username = fred:ttyS1)* and *ssh port = 22.*

Or, by typing *username=fred:serial* and *ssh port = 22,* the *User* is presented with a port selection option:



This syntax enables *Users* to set up SSH tunnels to all serial ports with only a single IP port 22 having to be opened in their firewall/gateway

Note In *Console Server* mode, when you connect through to a serial port you connect via *pmshell*. To will generate a BREAK on the serial port if you're connected over SSH, you'll need to type the character sequence "~~b"

TCP RAW TCP allows connections directly to a TCP socket. However while communications programs like *PuTTY* also supports RAW TCP, this protocol would usually be used by a custom application

For RAW TCP, the default port address is IP Address _ Port (4000 + serial port #) *i.e.* 4001 – 4048

RAW TCP also enables the serial port to be tunneled to a remote *console server*, so two serial port devices can be transparently interconnect over a network (see *Chapter 4.1.6 – Serial Bridging*)

RFC2217 Selecting *RFC2217* enables serial port redirection on that port. For RFC2217, the default port address is IP Address _ Port (5000 + serial port #) *i.e.* 5001 – 5048

Special client software is available for Windows UNIX and Linux that supports RFC2217 virtual com ports, so a remote host can monitor and manage remote serially attached devices, as though they were connected to the local serial port (see *Chapter 4.6 – Serial Port Redirection* for details)

RFC2217 also enables the serial port to be tunneled to a remote *console server*, so two serial port devices can be transparently interconnect over a network (see *Chapter 4.1.6 – Serial Bridging*)

Unauthenticated Telnet Selecting *Unauthenticated Telnet* enables telnet access to the serial port without requiring the user to provide credentials. When a user accesses the *console server* to telnet to a serial port they normally are given a login prompt. However with unauthenticated telnet they connect directly through to port with any *console server* login at all. This mode is mainly used when you have an external system (such as *conserver*) managing user authentication and access privileges at the serial device level.

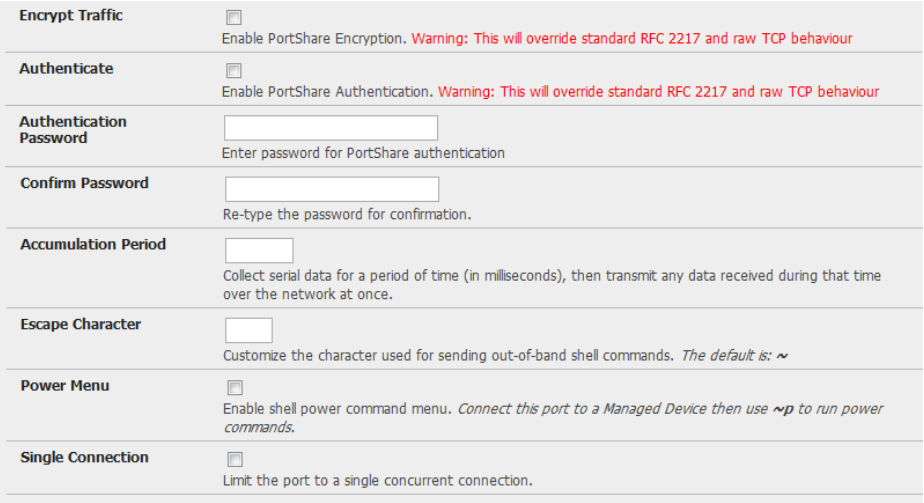
For Unauthenticated Telnet the default port address is IP Address _ Port (6000 + serial port #) i.e. 6001 – 6048

Web Terminal Selecting Web Terminal enables web browser access to the serial port via **Manage: Devices: Serial** using the Management Console's built in AJAX terminal. Web Terminal connects as the currently authenticated Management Console user and does not re-authenticate. See section 13.3 for more details.

IP Alias Enable access to the serial port using a specific IP address, specified in CIDR format. Each serial port can one or more IP aliases configured on a per-interface basis. These IP addresses can only be used to access the specific serial port, accessible using the standard protocol TCP port numbers of the console server services. For example, SSH on serial port 3 would be accessible on port 22 of a serial port IP alias (whereas on the console server's primary address it is available on port 2003).

This feature can also be configured via the multiple port edit page. In this case the IP addresses are applied sequentially, with the first selected port getting the IP entered and subsequent ones getting incremented, with numbers being skipped for any unselected ports. For example if ports 2, 3 and 5 are selected and the IP alias 10.0.0.1/24 is entered for the Network Interface, the following addresses will be asssigned:

Port 2: 10.0.0.1/24 Port 3: 10.0.0.2/24 Port 5: 10.0.0.4/24



Encrypt Traffic / Authenticate Enable trivial encryption and authentication of RFC2217 serial communications using Portshare (for strong encryption use VPN).

Accumulation Period By default once a connection has been established for a particular serial port (such as a RFC2217 redirection or Telnet connection to a remote computer) then any incoming characters on that port are forwarded over the network on a character by character basis. The accumulation period changes this by specifying a period of time that incoming characters will be collected before then being sent as a packet over the network

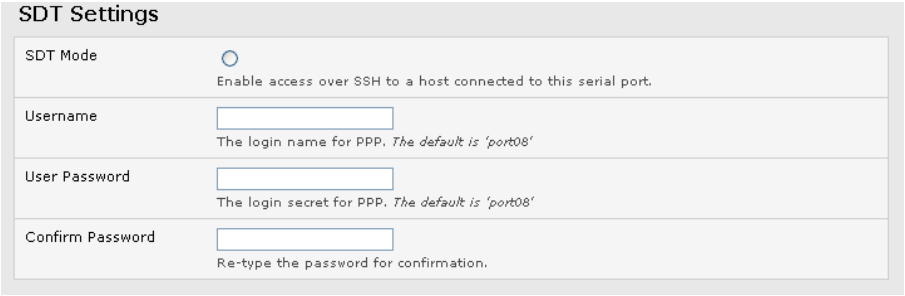
Escape Character This enables you to change the character used for sending escape characters. The default is **~.**

Power Menu This setting enables the shell power command so a user can control the power connection to a Managed Device from command line when they are telnet or ssh connected to the device. To operate the Managed Device must be set up with both its Serial port connection and Power connection configured. The command to bring up the power menu is **~p**

Single Connection This setting limits the port to a single connection so if multiple users have access privileges for a particular port only one user at a time can be accessing that port (i.e. port “snooping” is not permitted)

4.1.3 SDT Mode

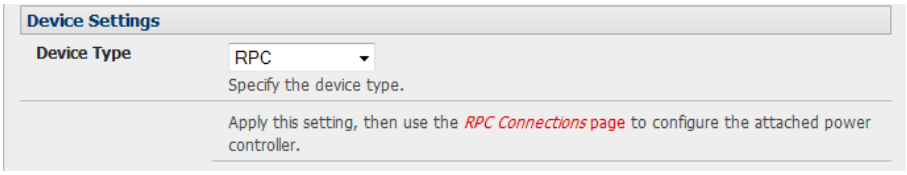
This Secure Tunneling setting allows port forwarding of RDP, VNC, HTPP, HTTPS, SSH, Telnet and other LAN protocols through to computers which are locally connected to the *console server* by their serial COM port. However such port forwarding requires a PPP link to be set up over this serial port.



For configuration details refer to Chapter *6.6 - Using SDT Connector to Telnet or SSH connect to devices that are serially attached to the console server*

4.1.4 Device (RPC, UPS, EMD) Mode

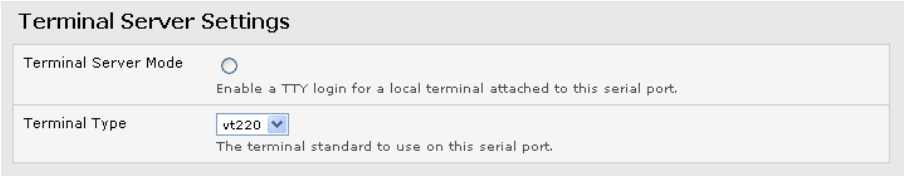
This mode configures the selected serial port to communicate with a serial controlled Uninterruptable Power Supply (UPS), Remote Power Controller / Power Distribution Units (RPC) or Environmental Monitoring Device (EMD)



- Select the desired **Device Type** (UPS, RPC or EMD)
- Proceed to the appropriate device configuration page (**Serial & Network: UPS Connections, RPC Connection** or **Environmental**) as detailed in *Chapter 8 - Power & Environmental Management*

4.1.5 Terminal Server Mode

- Select **Terminal Server Mode** and the **Terminal Type** (vt220, vt102, vt100, Linux or ANSI) to enable a *getty* on the selected serial port



The *getty* will then configure the port and wait for a connection to be made. An active connection on a serial device is usually indicated by the Data Carrier Detect (DCD) pin on the serial device being raised. When a connection is detected, the *getty* program issues a login: prompt, and then invokes the login program to handle the actual system login.

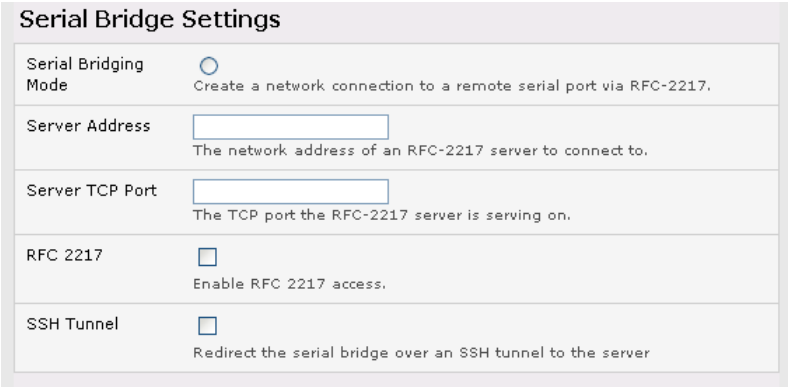
Note Selecting Terminal Server mode will disable Port Manager for that serial port, so data is no longer logged for alerts etc.

4.1.6 Serial Bridging Mode

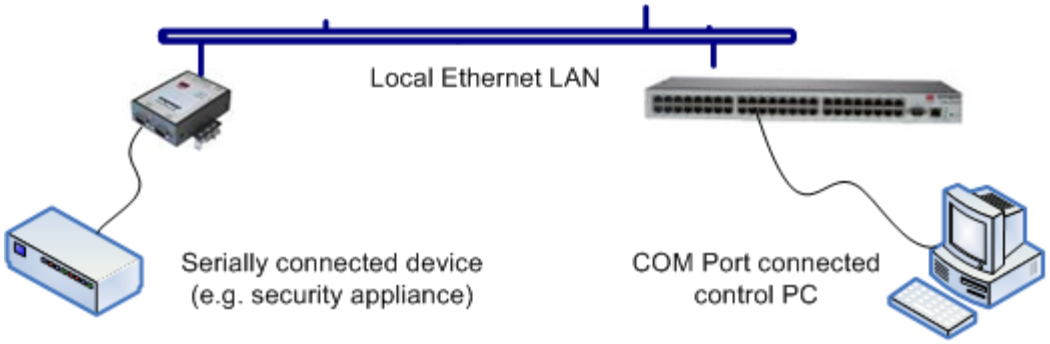
With serial bridging, the serial data on a nominated serial port on one *console server* is encapsulated into network packets and then transported over a network to a second *console server* where is then represented as serial data. So the two *console servers* effectively act as a virtual serial cable over an IP network.

One *console server* is configured to be the *Server*. The *Server* serial port to be bridged is set in *Console Server mode* with either RFC2217 or RAW enabled (as described in *Chapter 4.1.2 – Console Server Mode*).

For the *Client console server*, the serial port to be bridged must be set in Bridging Mode:



- Select **Serial Bridging Mode** and specify the IP address of the *Server console server* and the TCP port address of the remote serial port (for RFC2217 bridging this will be 5001-5048)
- By default the bridging client will use RAW TCP so you must select RFC2217 if this is the *console Server* mode you have specified on the server *console server*

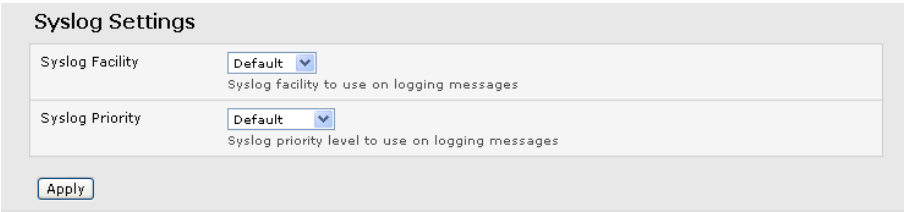


- You may secure the communications over the local Ethernet by enabling SSH however you will need to generate and upload keys (refer *Chapter 14 – Advanced Configuration*)

4.1.7 Syslog

In addition to inbuilt logging and monitoring (which can be applied to serial-attached and network-attached management accesses, as covered in *Chapter 7 - Alerts and Logging*) the *console server* can also be configured to support the remote syslog protocol on a per serial port basis:

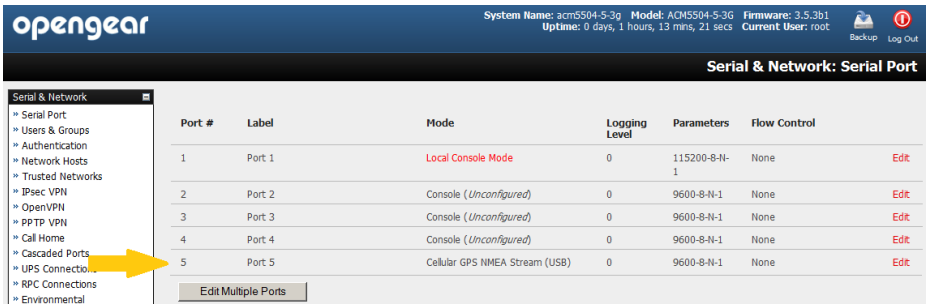
- Select the **Syslog Facility/Priority** fields to enable logging of traffic on the selected serial port to a syslog server; and to appropriately sort and action those logged messages (i.e. redirect them / send alert email etc.)



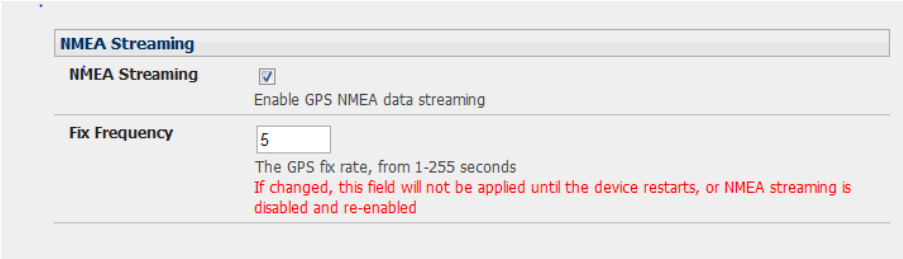
For example if the computer attached to serial port 3 should never send anything out on its serial console port, the *Administrator* can set the **Facility** for that port to *local0* (*local0 .. local7* are meant for site local values), and the **Priority** to *critical*. At this priority, if the *console server* syslog server does receive a message, it will automatically raise an alert. Refer to *Chapter 7 - Alerts & Logging*

4.1.8 NMEA Streaming

The ACM5004-G-I, ACM5504-5-G(-W)-I, ACM5504-5-LA/LR/LV-I, ACM7000-L and IM4200-G can provide GPS NMEA data streaming from the internal GPS /cellular modem. This data stream presents as a serial data steam on port 5 on the ACM models. For the IM4200-G with an internal cellular modem, the NMEA data stream presents on ports 9/17/33/49 for the IM4208/16/32/48 models.



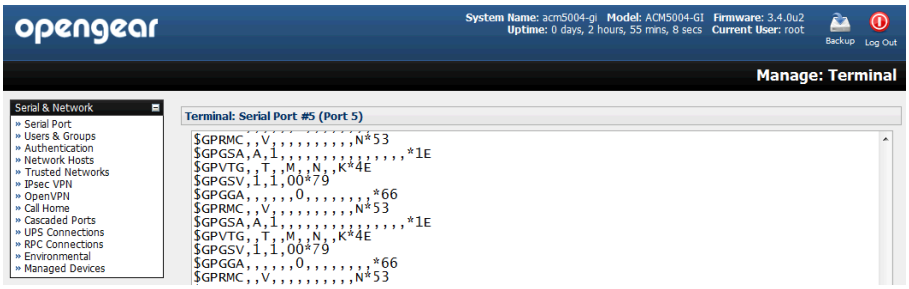
The Common Settings (baud rate etc.) are ignored when configuring the NMEA "serial port". However you can specify the **Fix Frequency** (i.e. this GPS fix rate determines how often GPS fixes are obtained). You can also apply all the Console Server Mode, Syslog and Serial Bridging settings to this port.



Note: The *NMEA Streaming* menu item should display on the *Serial & Network: Serial Port* menu. However for earlier revision ACM5004-G-I units you may need to update the *setfset* settings from the command line: ***setfset -r*** lists all of the current feature set variables. You look for the *factory_opts* variable, and then add 3g-gps to it. For example, *factory_opts=rs485,3g,ind*. To update it to 3g-gps, you do the following: *setfset -u factory_opts=rs485,3g-gps,ind.* Then run *setfset -r* again, and make sure you can see the update

You can use *pmshell*, *webshell*, *SSH, RFC2217 or RawTCP* to get at the stream:

For example using the Web Terminal:

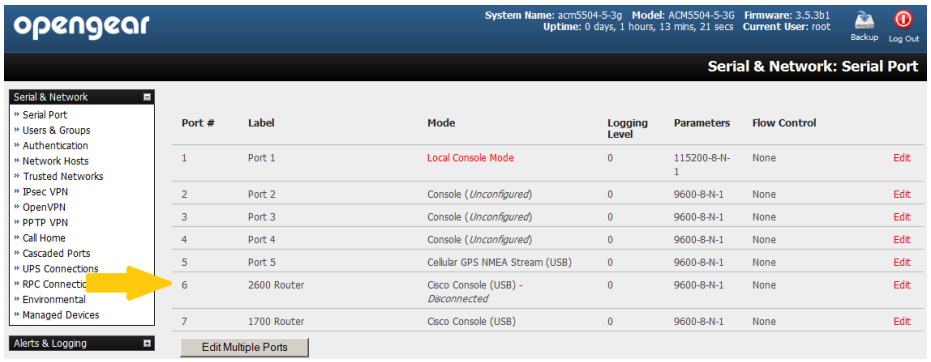


Note: This GPS support is also available for IM4200-G with an internal cellular modem. The NMEA data stream presents on ports 9/17/33/49 for the IM4208/16/32/48 models.

4.1.9 Cisco USB console connection

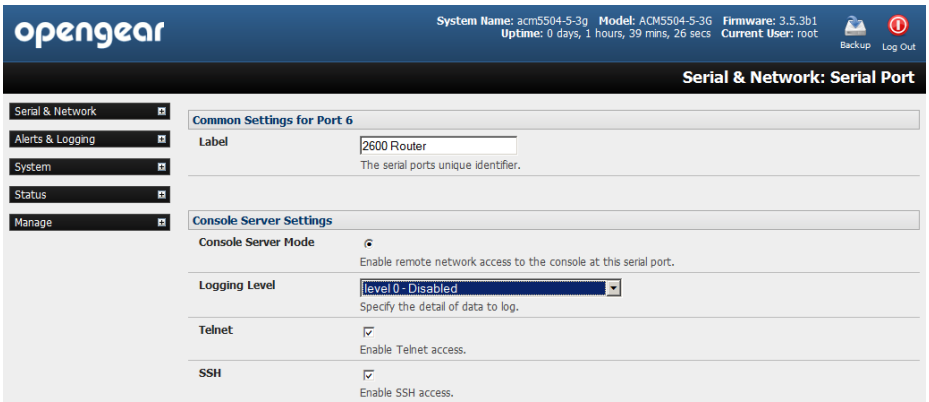
The ACM5000, ACM5500, ACM7000, CM7100, IM7200 and IM4200 family *console servers* support direct USB2.0 connection to one or two Cisco USB console ports (in addition to the traditional RS-232 serial console port connections).

With such a USB console connection users can send IOS commands through the USB console port remotely (using a browser and the *console server's* built-in AJAX terminal) or monitor messages from the Cisco USB console ports and take rule book actions (using the *console server's* built-in Auto-Response capabilities).



For configuration and control these USB consoles are presented as new "serial ports". So for an ACM5504-5-G with cellular GPS configured on Port 5 (as shown above) any Cisco USB console ports would present as Port 6 and 7. For the IM4200 (without any internal GPS modem functions) any configured Cisco USB console ports would presents on ports 9&10/17&18/33&34/49&50 for the IM4208/16/32/48 models.

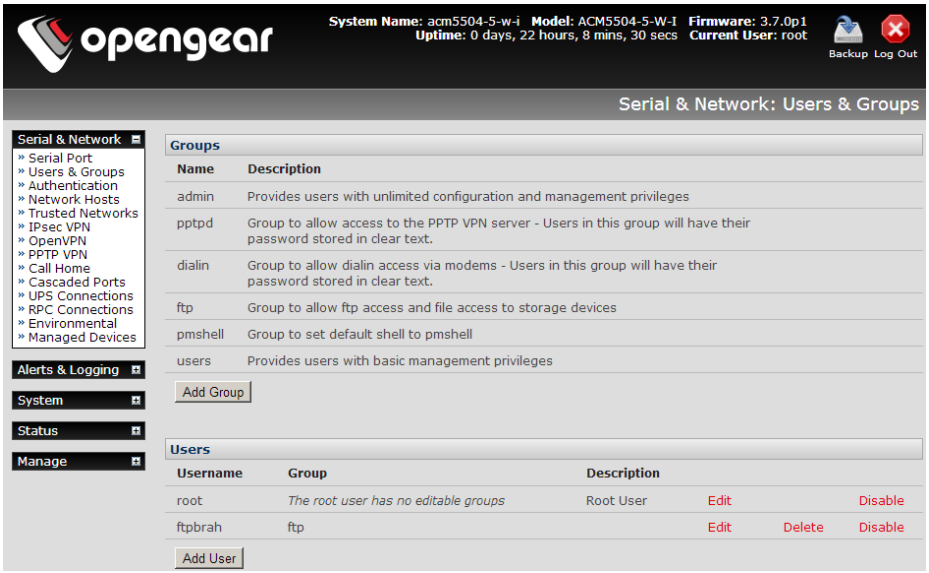
The Common Settings (baud rate etc.) are ignored when configuring the Cisco USB "serial port". However you can apply all the Console Server Mode, Syslog and Serial Bridging settings to this port.



Note: The Cisco USB console must be manually configured on initial connection. However any USB console disconnection is auto-detected. USB console re-connection on the same physical USB port will also be auto-detected, but only if the *console server* has been power cycled.

4.2 Add / Edit Users

The *Administrator* uses this menu selection to set up, edit and delete users and to define the access permissions for each of these users.



Users can be authorized to access specified services, serial ports, power devices and specified network-attached hosts. These users can also be given full *Administrator* status (with full configuration and management and access privileges).

To simplify user set up, they can be configured as members of Groups. With firmware V3.5.2 and later there are six Groups set up by default (where earlier versions only had *admin* and *user* by default):

admin	Provides users with unlimited configuration and management privileges
pptpd	Group to allow access to the PPTP VPN server. Users in this group will have their password stored in clear text.
dialin	Group to allow dialin access via modems. Users in this group will have their password stored in clear text.
ftp	Group to allow ftp access and file access to storage devices
pmshell	Group to set default shell to pmshell
users	Provides users with basic management privileges

Note: 1. Membership of the ***admin*** group provides the user with full *Administrator* privileges. The *admin* user (*Administrator*) can access the *console server* using any of the services which have been enabled in *System: Services* e.g. if only HTTPS has been enabled then the *Administrator* can only access the *console server* using HTTPS. However once logged in they can reconfigure the *console server* settings (e.g. to enabled HTTP/Telnet for future access). They can also access any of the connected Hosts or serial port devices using any of the services that have been enabled for these connections. But again the *Administrator* can reconfigure the access services for any Host or serial port. So only trusted users should have *Administrator* access

2. Membership of the **user** group provides the user with limited access to the *console server* and connected Hosts and serial devices. These *Users* can access only the Management section of the Management Console menu and they have no command line access to the *console server*. They also can only access those Hosts and serial devices that have been checked for them, using services that have been enabled

3. If a user is set up with **pptd, dialin, ftp** or **pmshell** group membership they will have restricted user shell access to the nominated managed devices but they will not have any direct access to the console server itself. To add this the users must also be a member of the "users" or "admin" groups

4. The *Administrator* can also set up additional Groups with specific power device, serial port and host access permissions. However users in these additional groups don't have any access to the Management Console menu nor do they have any command line access to the *console server* itself.

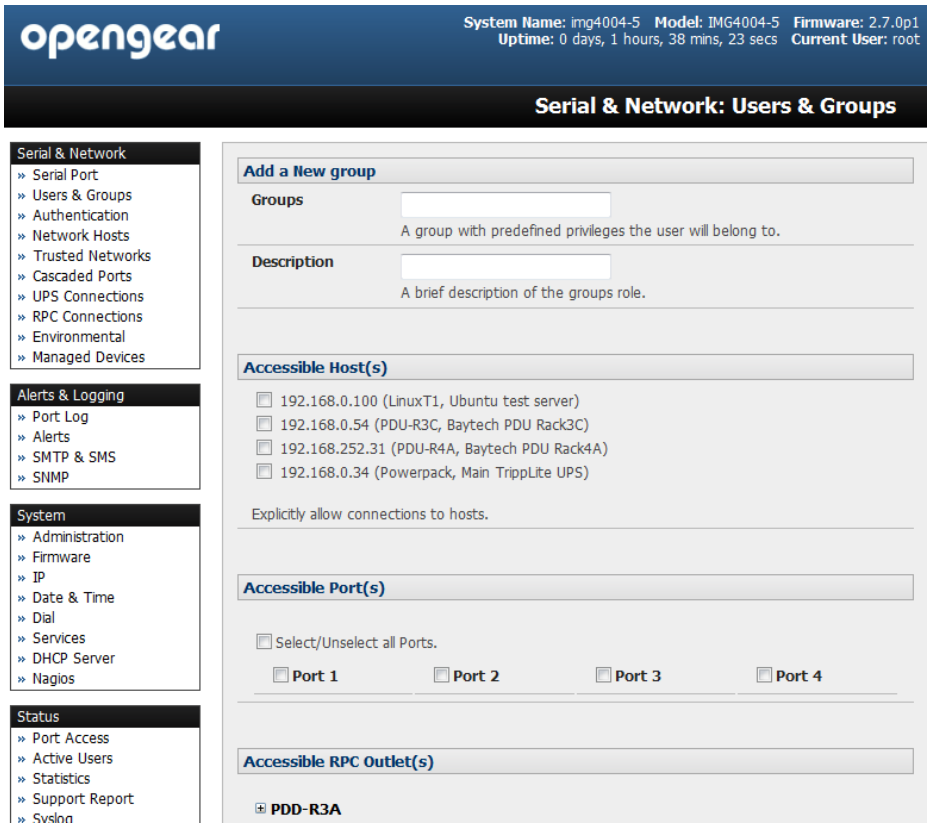
5. The *Administrator* can also set up users with specific power device, serial port and host access permissions, who are not a member of any Groups. Similarly these users don't have any access to the Management Console menu nor do they have any command line access to the *console server* itself.

6. For convenience the SDT Connector "Retrieve Hosts" function retrieves and auto-configures checked serial ports and checked hosts only, even for admin group users

4.2.1 Set up new Group

To set up new Groups and new users, and to classify users as members of particular Groups:

- Select **Serial & Network: Users & Groups** to display the configured Groups and Users
- Click **Add Group** to add a new Group



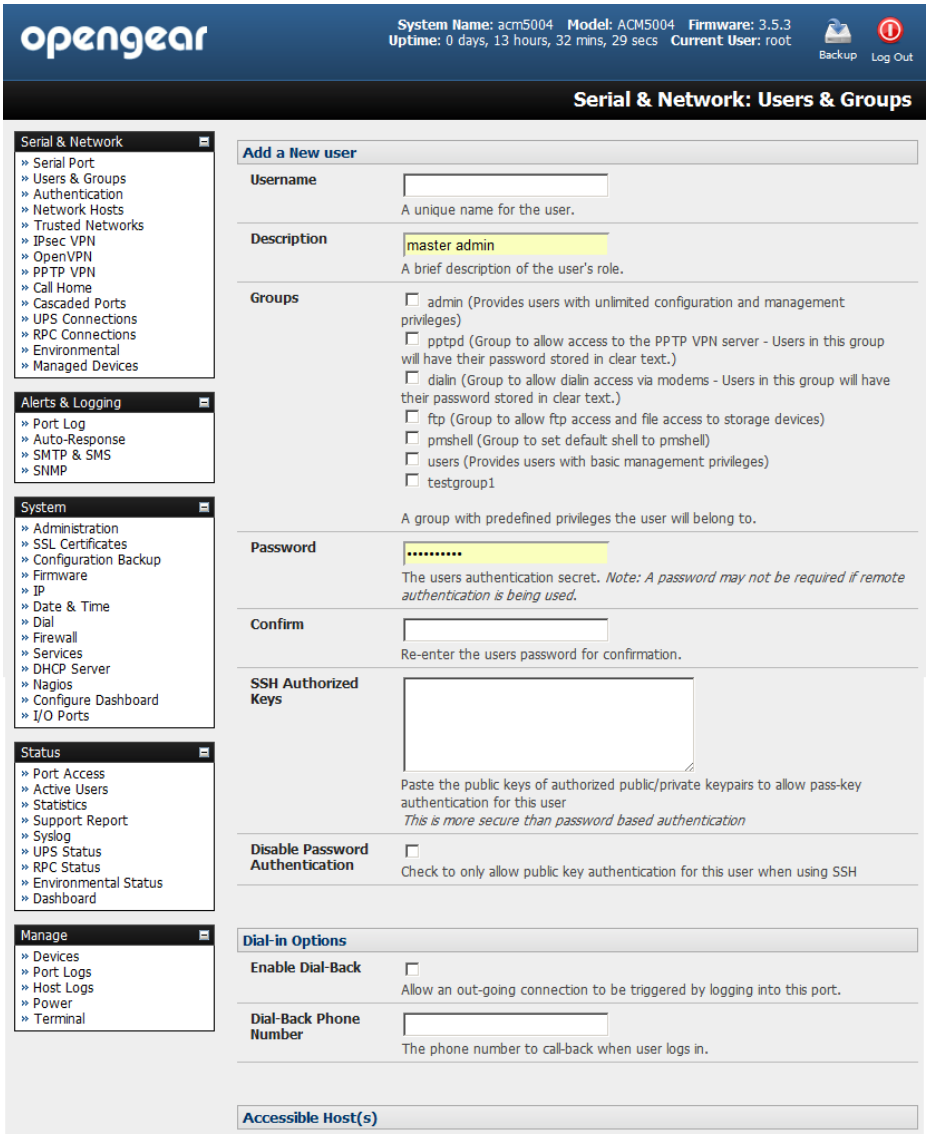
- Add a **Group** name and **Description** for each new Group, then nominate the **Accessible Hosts, Accessible Ports** and **Accessible RPC Outlet(s)** that you wish any users in this new Group to be able to access

- Click **Apply**
- The *Administrator* can **Edit** or **Delete** any added group

4.2.2 Set up new Users

To set up new users, and to classify users as members of particular Groups:

- Select **Serial & Network: Users & Groups** to display the configured Groups and User
- Click **Add User** to add a new user



- Add a **Username** for each new user. You may also include information related to the user (*e.g.* contact details) in the **Description** field

Note The User Name can contain from 1 to 127 alphanumeric characters (however you can also use the special characters "-" "_" and ".")

- Specify which **Group** (or Groups) you wish the user to be a member of
- Add a confirmed **Password** for each new user

Note There are no restrictions on the characters that can be used in the user Password (which each can contain up to 254 characters). However only the first eight Password characters are used to make the *password hash*.

- SSH pass-key authentication can be used. This is more secure than password based authentication. Paste the public keys of authorized public/private keypairs for this user in the **Authorized SSH Key**s field
- Check **Disable Password Authentication** if you wish to only allow public key authentication for this user when using SSH
- Check **Enable Dial-Back** in the **Dial-in Options** menu to allow an out-going dial-back connection to be triggered by logging into this port. Enter the **Dial-Back Phone Number** with the phone number to call-back when user logs in
- Check specific **Accessible Hosts** and/or **Accessible Ports** to nominate the serial ports and network connected hosts you wish the user to have access privileges to
- If there are configured RPCs you can check **Accessible RPC Outlets** to specify which outlets the user is able to control (i.e. Power On/Off)
- Click **Apply**. The new user will now be able to access the Network Devices, Ports and RPC Outlets you nominated as accessible plus, if the user is a Group member they can also access any other device/port/outlet that was set up as accessible to the Group

Note There are no specific limits on the number of users you can set up; nor on the number of users per serial port or host. So multiple users (*Users* and *Administrators*) can control /monitor the one port or host. Similarly there are no specific limits on the number of Groups and each user can be a member of a number of Groups (in which case they take on the cumulative access privileges of each of those Groups). A user does not have to be a member of any Groups (but if the *User* is not even a member of the default *user* group then they will not be able to use the Management Console to manage ports).

Note that while there are no specific limits the time to re-configure does increase as the number and complexity increases so we recommend the aggregate number if users and groups be kept under 250

The *Administrator* can also edit the access settings for any existing users:

- Select **Serial & Network: Users & Groups** and click **Edit** to modify the *User* access privileges
- Alternately click **Delete** to remove the *User* or click **Disable** to temporarily block any access privileges

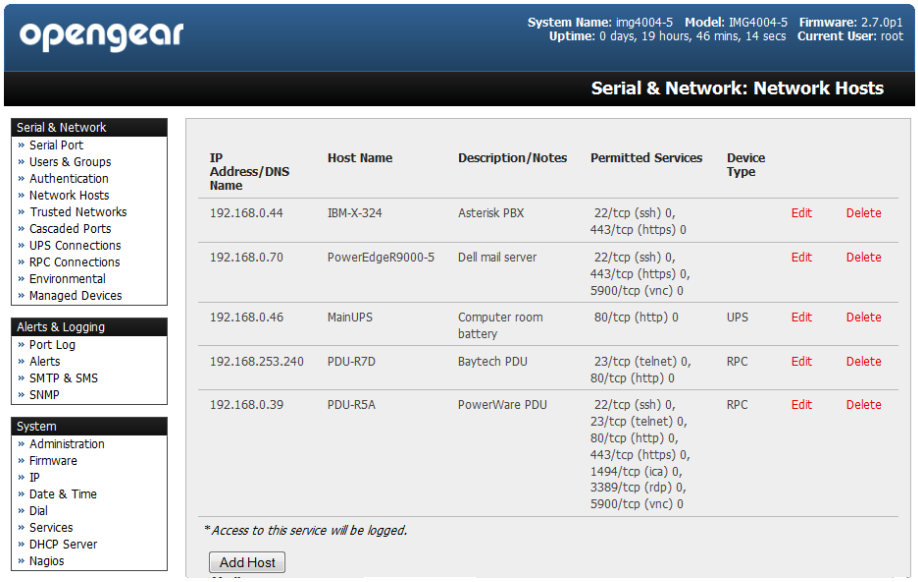
Note For more information on enabling the SDT Connector so each user has secure tunneled remote RPD/VNC/Telnet/HHTP/HTTPS/SoL access to the network connected hosts refer *Chapter 6*.

4.3 Authentication

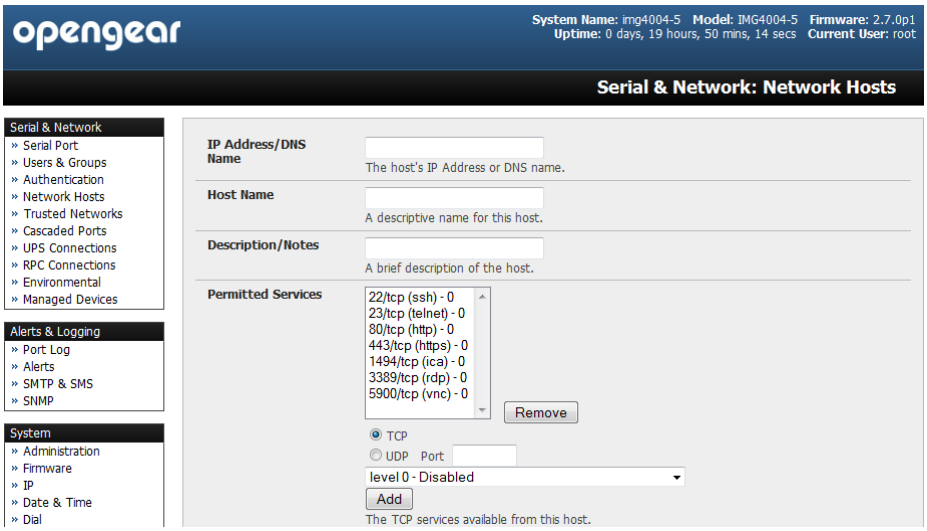
Refer to *Chapter 9.1 - Remote Authentication Configuration* for authentication configuration details

4.4 Network Hosts

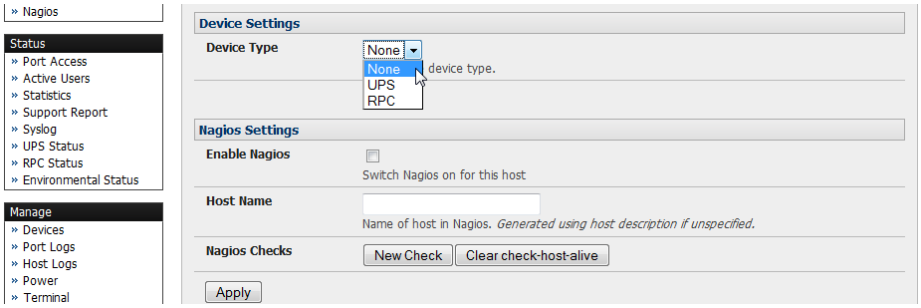
To monitor and remotely access a locally networked computer or device (referred to as a *Host*) you must identify the Host and specify the TCP or UDP ports/services that will be used to control that Host:



- Selecting **Serial & Network: Network Hosts** presents all the network connected Hosts that have been enabled for access, and the related access TCP ports/services
- Click **Add Host** to enable access to a new Host (or select **Edit** to update the settings for existing Host)



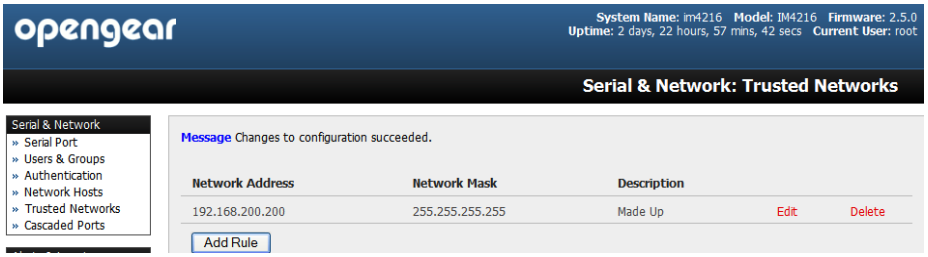
- Enter the **IP Address** or **DNS Name** and a **Host Name** (up to 254 alphanumeric characters) for the new network connected Host (and optionally enter a **Description** -up to characters)
- Add or edit the **Permitted Services** (or TCP/UDP port numbers) that are authorized to be used in controlling this host. Only these *permitted services* will be forwarded through by SDT to the Host. All other services (TCP/UDP ports) will be blocked.
- The **Logging Level** specifies the level of information to be logged and monitored for each Host access (refer *Chapter 7 - Alerts and Logging*)
- If the Host is a PDU or UPS power device or a server with IPMI power control, then specify **RPC** (for IPMI and PDU) or **UPS** and the **Device Type**. The *Administrator* can then configure these devices and enable which users have permissions to remotely cycle power etc. (refer *Chapter 8*). Otherwise leave the Device Type set to None



- If the *console server* has been configured with distributed Nagios monitoring enabled then you will also be presented with **Nagios Settings** options to enable nominated services on the Host to be monitored (refer *Chapter 10 – Nagios Integration*)
- Click **Apply.** This will create the new Host and also create a new Managed Device (with the same name)

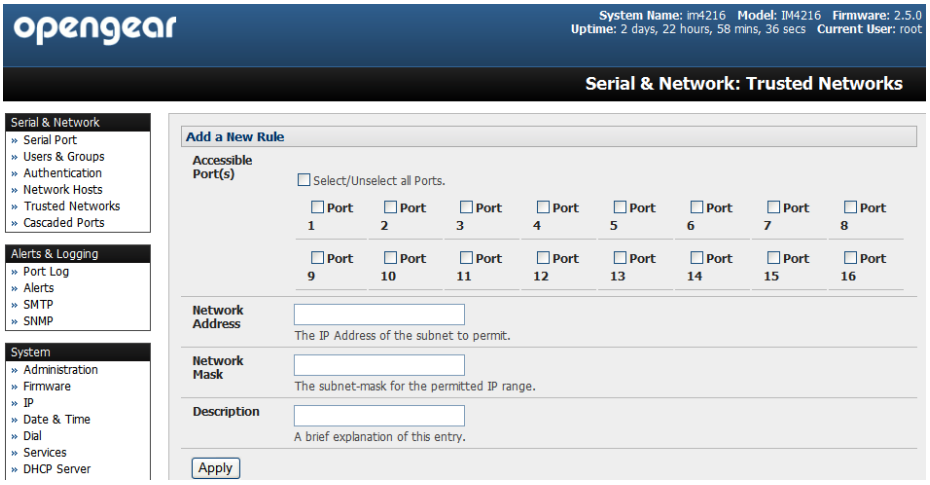
4.5 Trusted Networks

The **Trusted Networks** facility gives you an option to nominate specific IP addresses that users (*Administrators* and *Users*) must be located at, to have access to *console server* serial ports:



- Select **Serial & Network: Trusted Networks**
- To add a new trusted network, select **Add Rule**

Note In the absence of *Rules*, there are no access limitations as to the IP address at which *Users* or *Administrators* can be located.



- Select the **Accessible Port(s)** that the new rule is to be applied to
- Then enter the **Network Address** of the subnet to be permitted access

- Then specify the range of addresses that are to be permitted by entering a **Network Mask** for that permitted IP range *e.g.*
  - To permit all the users located with a particular Class C network (204.15.5.0 say) connection to the nominated port then you would add the following Trusted Network New Rule:

Network IP Address	204.15.5.0
Subnet Mask	255.255.255.0

  - If you want to permit only the one user who is located at a specific IP address (204.15.5.13 say) to connect:

Network IP Address	204.15.5.13
Subnet Mask	255.255.255.255

  - If however you want to allow all the users operating from within a specific range of IP addresses (say any of the thirty addresses from 204.15.5.129 to 204.15.5.158) to be permitted connection to the nominated port:

Host /Subnet Address	204.15.5.128
Subnet Mask	255.255.255.224

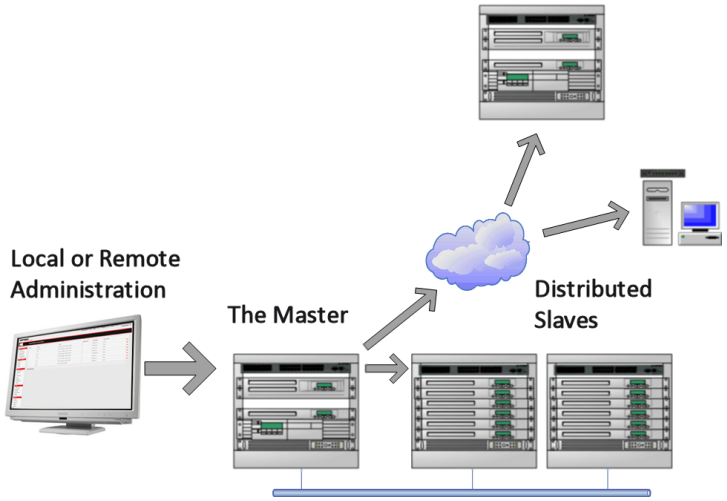
- Click **Apply**

Note The above Trusted Networks will limit access by *Users* and *Administrators* to the console serial ports. However they do not restrict access by the *Administrator* to the *console server* itself or to attached hosts. To change the default settings for this access, you will to need to edit the *IPtables* rules as described in the *Chapter 14 - Advanced.*

4.6 Serial Port Cascading

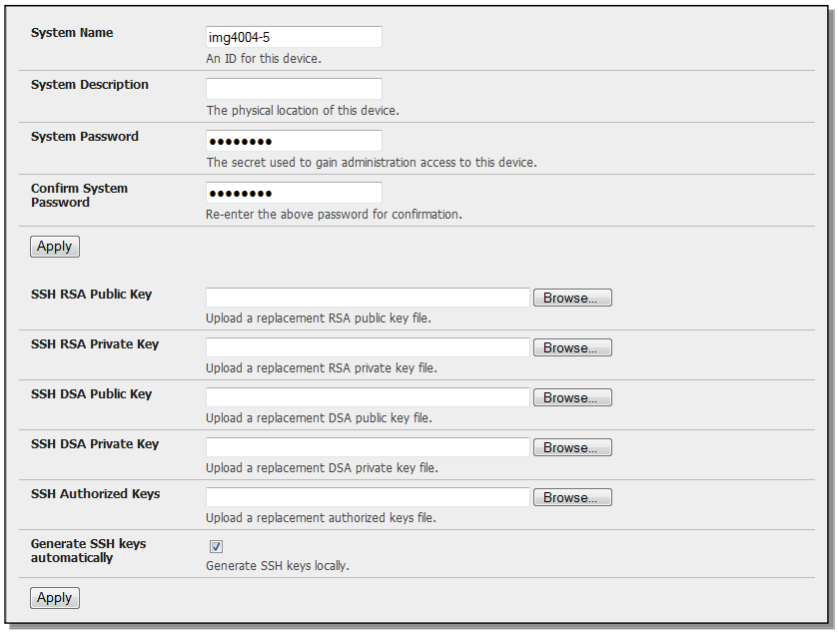
Cascaded Ports enables you to cluster distributed *console servers* so a large number of serial ports (up to 1000) can be configured and accessed through one IP address and managed through the one Management Console. One *console server*, the Master, controls other *console servers* as Slave units and all the serial ports on the Slave units appear as if they are part of the Master.

Opengear's clustering connects each Slave to the Master with an SSH connection. This is done using public key authentication so the Master can access each Slave using the SSH key pair (rather than using passwords). This ensures secure authenticated communications between Master and Slaves enabling the Slave *console server* units to be distributed locally on a LAN or remotely around the world.

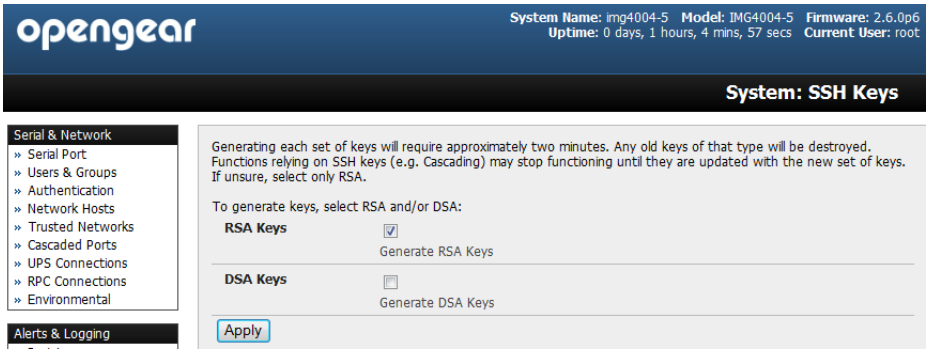


4.6.1 Automatically generate and upload SSH keys

To set up public key authentication you must first generate an RSA or DSA key pair and upload them into the Master and Slave *console servers*. This can all be done automatically from the Master:

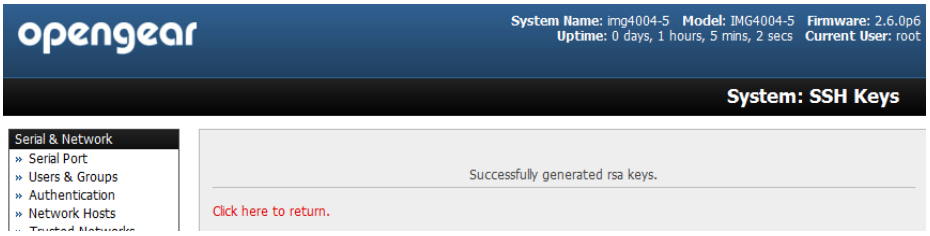


- Select **System: Administration** on Master's Management Console
- Check **Generate SSH keys automatically** and click **Apply**



Next you must select whether to generate keys using RSA and/or DSA (if unsure, select only RSA). Generating each set of keys will require approximately two minutes and the new keys will destroy any old keys of that type that may previously been uploaded. Also while the new generation is underway on the master functions relying on SSH keys (e.g. cascading) may stop functioning until they are updated with the new set of keys. To generate keys:

- Select **RSA Keys** and/or **DSA Keys**
- Click **Apply**



- Once the new keys have been successfully generated simply **Click here to return** and the keys will automatically be uploaded to the Master and connected Slaves

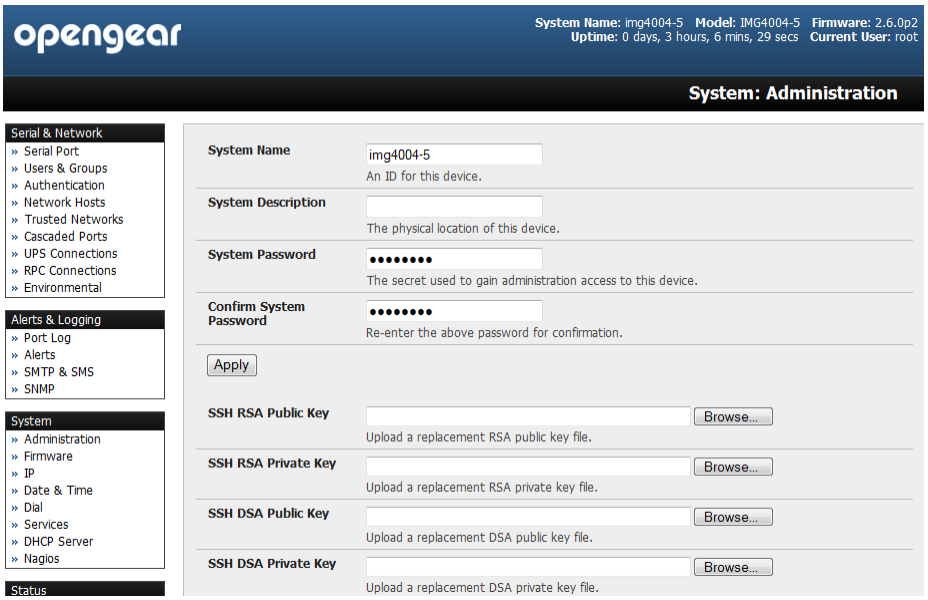
4.6.2 Manually generate and upload SSH keys

Alternately if you have a RSA or DSA key pair you can manually upload them to the Master and Slave *console servers*.

Note If you do not already have RSA or DSA key pair and you do not wish to use you will need to create a key pair using *ssh-keygen, PuTTYgen* or a similar tool as detailed in Chapter 15.6

To manually upload the key public and private key pair to the Master *console server*:

- Select **System: Administration** on Master's Management Console
- Browse to the location you have stored RSA (or DSA) Public Key and upload it to **SSH RSA (DSA) Public Key**
- Browse to the stored RSA (or DSA) Private Key and upload it to **SSH RSA (DSA) Private Key**
- Click **Apply**



Next, you must register the Public Key as an Authorized Key on the Slave. In the simple case with only one Master with multiple Slaves, you need only upload the one RSA or DSA public key for each Slave.

Note The use of key pairs can be confusing as in many cases one file (Public Key) fulfills two roles – Public Key and Authorized Key. For a more detailed explanation refer the *Authorized Keys* section of Chapter 15.6. Also refer to this chapter if you need to use more than one set of Authorized Keys in the Slave

- Select **System: Administration** on the Slave's Management Console
- Browse again to the stored RSA (or DSA) Public Key and upload it to Slave's **SSH Authorized Key**
- Click **Apply**

The next step is to *Fingerprint* each new Slave-Master connection. This once-off step will validate that you are establishing an SSH session to who you think you are. On the first connection the Slave will receive a *fingerprint* from the Master which will be used on all future connections:

- To establish the fingerprint first log in the Master server as *root* and establish an SSH connection to the Slave remote host:

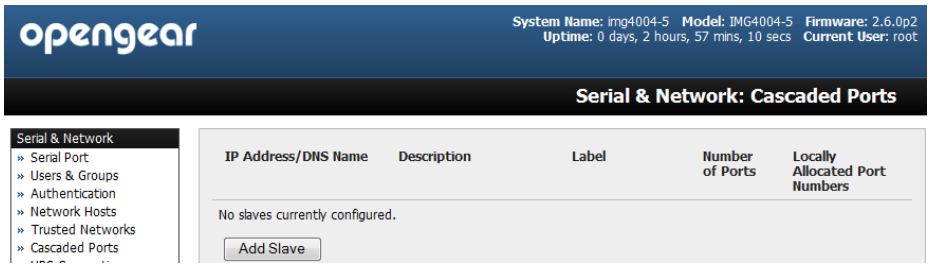
 # ssh remhost

Once the SSH connection has been established you will be asked to accept the key. Answer *yes* and the *fingerprint* will be added to the list of known hosts. For more details on Fingerprinting refer Chapter 15.6

- If you are asked to supply a password, then there has been a problem with uploading keys. The keys should remove any need to supply a password

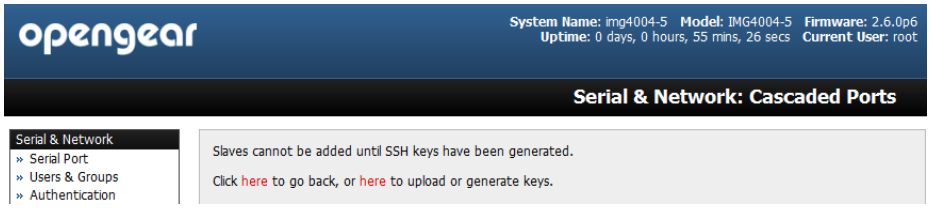
4.6.3 Configure the slaves and their serial ports

You can now begin setting up the Slaves and configuring Slave serial ports from the Master *console server*:



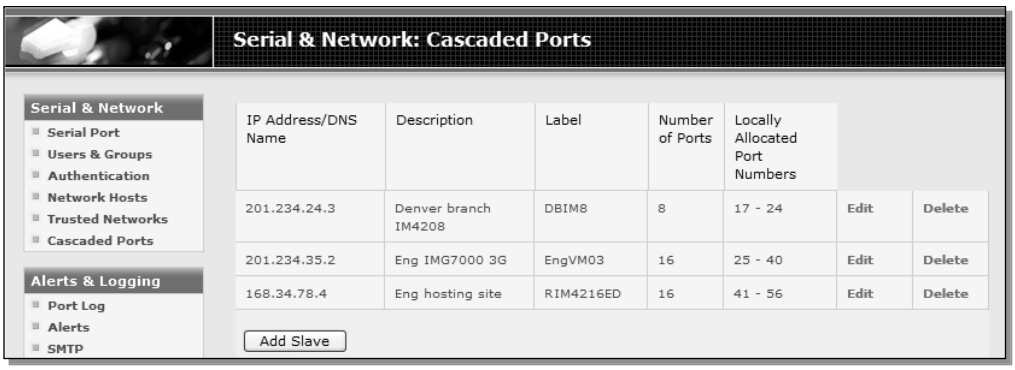
- Select **Serial & Network: Cascaded Ports** on the Master's Management Console:
- To add clustering support select **Add Slave**

Note You will be prevented from adding any Slaves until you have automatically or manually generated SSH keys:



To define and configure a Slave:

- Enter the remote **IP Address** (or DNS Name) for the Slave *console server*
- Enter a brief **Description** and a short **Label** for the Slave (use a convention here that enables effective management of large networks of clustered *console servers* and the connected devices)
- Enter the full number of serial ports on the Slave unit in **Number of Ports**
- Click **Apply**. This will establish the SSH tunnel between the Master and the new Slave



The **Serial & Network: Cascaded Ports** menu displays all the Slaves and the port numbers that have been allocated on the Master. If the Master *console server* has 16 ports of its own then ports 1-16 are pre- allocated to the Master, so the first Slave added will be assigned port number 17 onwards.

Once you have added all the Slave *console servers*, the Slave serial ports and the connected devices are configurable and accessible from the Master's Management Console menu; and accessible through the Master's IP address e.g.

- Select the appropriate **Serial & Network: Serial Port** and **Edit** to configure the serial ports on the Slave
- Select the appropriate **Serial & Network: Users & Groups** to add new users with access privileges to the Slave serial ports (or to extend existing users access privileges)
- Select the appropriate **Serial & Network: Trusted Networks** to specify network addresses that can access nominated Slave serial ports
- Select the appropriate **Alerts & Logging: Alerts** to configure Slave port Connection, State Change or Pattern Match alerts
- The configuration changes made on the Master are propagated out to all the Slaves when you click **Apply**.

4.6.4 Managing the slaves

The Master is in control of the Slave serial ports. So for example if change a *User* access privileges or edit any serial port setting on the Master, the updated configuration files will be sent out to each Slave in parallel. Each Slave will then automatically make changes to their local configurations (and only make those changes that relate to its particular serial ports).

You can still use the local Slave Management Console to change the settings on any Slave serial port (such as alter the baud rates). However these changes will be overwritten next time the Master sends out a configuration file update.

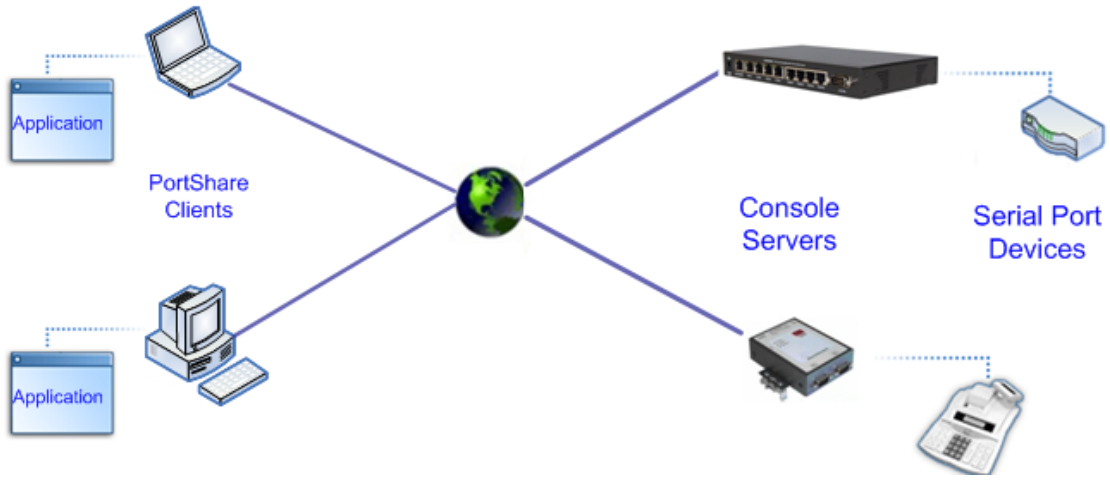
Also while the Master is in control of all Slave serial port related functions, it is not master over the Slave network host connections or over the Slave *console server* system itself.

So Slave functions such as IP, SMTP & SNMP Settings, Date &Time, DHCP server must be managed by accessing each Slave directly and these functions are not over written when configuration changes are propagated from the Master. Similarly the Slaves Network Host and IPMI settings have to be configured at each Slave.

Also the Master's Management Console provides a consolidated view of the settings for its own and the entire Slave's serial ports, however the Master does not provide a fully consolidated view. For example if you want to find out who's logged in to cascaded serial ports from the master, you'll see that *Status: Active Users* only displays those users active on the Master's ports, so you may need to write custom scripts to provide this view. This is covered in Chapter 11.

4.7 Serial Port Redirection (PortShare)

Opengear's Port Share software delivers the virtual serial port technology your Windows and Linux applications need to open remote serial ports and read the data from serial devices that are connected to your *console server*.



PortShare is supplied free with each *console server* and you are licensed to install *PortShare* on one or more computers for accessing any serial device connected to a *console server* port.

***PortShare* for Windows**

The *portshare_setup.exe* program is included on the CD supplied with your *console server.* A copy can be freely downloaded from the ftp site. Refer to the PortShare User Manual and Quick Start for details on installation and operation

***PortShare* for Linux**

The *PortShare* driver for Linux maps the console server serial port to a host *try* port. Opengear has released the *portshare-serial-client* as an open source utility for Linux, AIX, HPUX, SCO, Solaris and UnixWare. This utility can be freely downloaded from the ftp site.

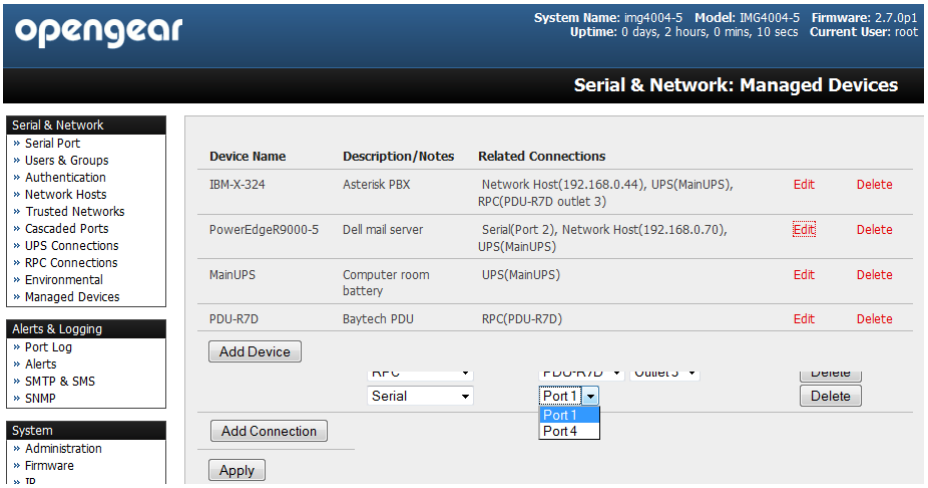
This PortShare serial port redirector allows you to use a serial device connected to the remote *console server* as if it were connected to your local serial port. The *portshare-serial-client* creates a pseudo tty port, connects the serial application to the pseudo tty port, receives data from the pseudo tty port, transmits it to the *console server* through network and receives data from the *console server* through network and transmits it to the pseudo-tty port.

The *.tar* file can be freely downloaded from the ftp site. Refer to the PortShare User Manual and Quick Start for details on installation and operation.

4.8 Managed Devices

Managed Devices presents a consolidated view of all the connections to a device that can be accessed and monitored through the *console server*. To view the connections to the devices:

- Select **Serial&Network: Managed Device**s



This screen displays all the Managed Device with their Description/Notes and lists of all the configured Connections:

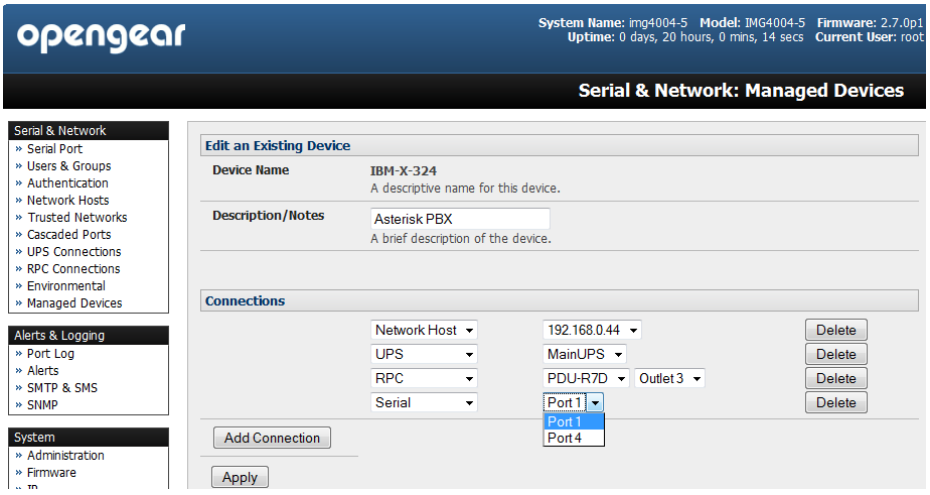
- *Serial Port #* (if serially connected) or
- *USB* (if USB connected)
- IP Address (if network connected)
- Power PDU/outlet details (if applicable) and any UPS connections

Devices such as servers will commonly have more than one power connections (e.g. dual power supplied) and more than one network connection (e.g. for BMC/service processor).

All users can view (but not edit) these Managed Device connections by selecting Manage: Devices. Whereas the *Administrator* can edit and add/delete these Managed Devices and their connections.

To edit an existing device and add a new connection:

- Select **Edit** on the **Serial&Network: Managed Device**s and click **Add Connection**
- Select the connection type for the new connection (Serial, Network Host, UPS or RPC) and then select the specific connection from the presented list of configured unallocated hosts/ports/outlets



To add a new network connected Managed Device:

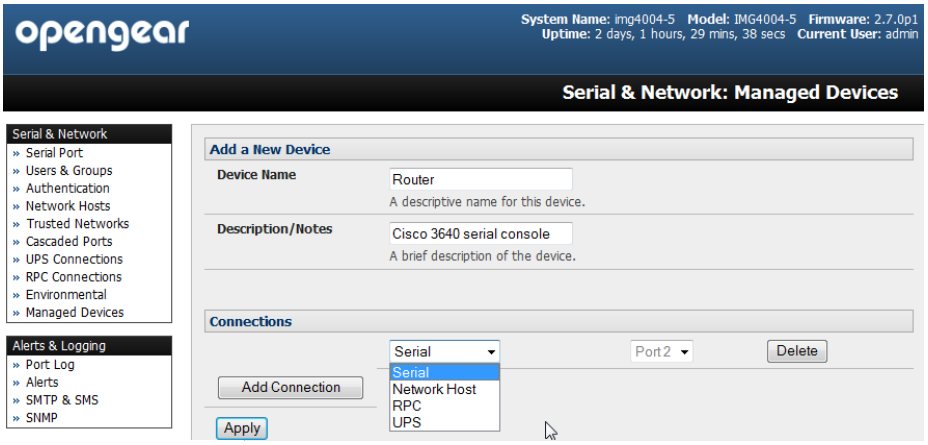
- The *Administrator* adds a new network connected Managed Device using **Add Host** on the **Serial&Network: Network Host** menu. This automatically creates a corresponding new Managed Device (as covered in *Section 4.4 - Network Hosts)*
- When adding a new network connected RPC or UPS power device, you set up a Network Host, designate it as RPC or UPS, then go to **RPC Connections** (or **UPS Connections**) to configure the relevant connection. Again

corresponding new Managed Device (with the same Name /Description as the RPC/UPS Host) is not created until this connection step is completed (refer *Chapter8 - Power and Environment)*

Note The outlet names on this newly created PDU will by default be “Outlet 1” “Outlet 2”. When you connect an particular Managed Device (that draws power from the outlet) they the outlet will then take up the name of the powered Managed Device

To add a new serially connected Managed Device:

- Configure the serial port using the **Serial&Network: Serial Port** menu (refer *Section 4.1 -Configure Serial Port)*
- Select **Serial&Network: Managed Devices** and click **Add Device**
- Enter a **Device Name** and **Description** for the Managed Device



- Click **Add Connection** and select **Serial** and the **Port** that connects to the Managed Device
- To add a UPS/RPC power connection or network connection or another serial connection click **Add Connection**
- Click **Apply**

Note To set up a new serially connected RPC UPS or EMD device, you configure the serial port, designate it as a Device then enter a Name and Description for that device in the **Serial & Network: RPC Connections** (or **UPS Connections** or **Environmental**). When applied, this will automatically create a corresponding new Managed Device with the same Name /Description as the RPC/UPS Host (refer *Chapter8 - Power and Environment)*

Also all the outlet names on the PDU will by default be “Outlet 1” “Outlet 2”. When you connect a particular Managed Device (that draws power from the outlet) then the outlet will then take up the name of the powered Managed Device

4.9 IPsec VPN

The ACM7000, ACM5500, ACM5000, CM7100, IM7200 and IM4200 family of advanced *console servers* include Openswan, a Linux implementation of the IPsec (IP Security) protocols, which can be used to configure a Virtual Private Network (VPN). The VPN allows multiple sites or remote administrators to access the Opengear advanced *console server* (and Managed Devices) securely over the Internet.

The administrator can establish an encrypted authenticated VPN connections between *advanced console serves* distributed at remote sites and a VPN gateway (such as Cisco router running *IOS IPsec)* on their central office network:

- Users and administrators at the central office can then securely access the remote console servers and connected serial console devices and machines on the Management LAN subnet at the remote location as though they were local

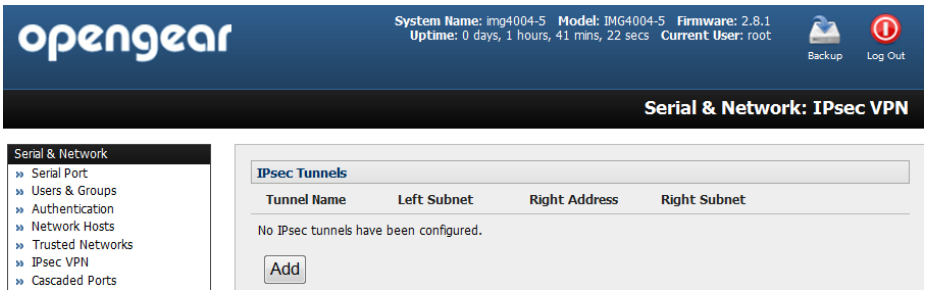
- All these remote console servers can then be monitored with a CMS6000 on the central network
- With serial bridging, serial data from controller at the central office machine can be securely connected to the serially controlled devices at the remote sites (refer Chapter 4.1)

The road warrior administrator can use a VPN IPsec software client such as TheGreenBow (www.thegreenbow.com/vpn_gateway.html) or Shrew Soft (www.shrew.net/support) to remotely access the advanced *console server* and every machine on the Management LAN subnet at the remote location

Configuration of IPsec is quite complex so Opengear provides a simple GUI interface for basic set up as described below. However for more detailed information on configuring Openswan IPsec at the command line and interconnecting with other IPsec VPN gateways and road warrior IPsec software refer **http://wiki.openswan.org** and **http://opengear.com/faq.html**

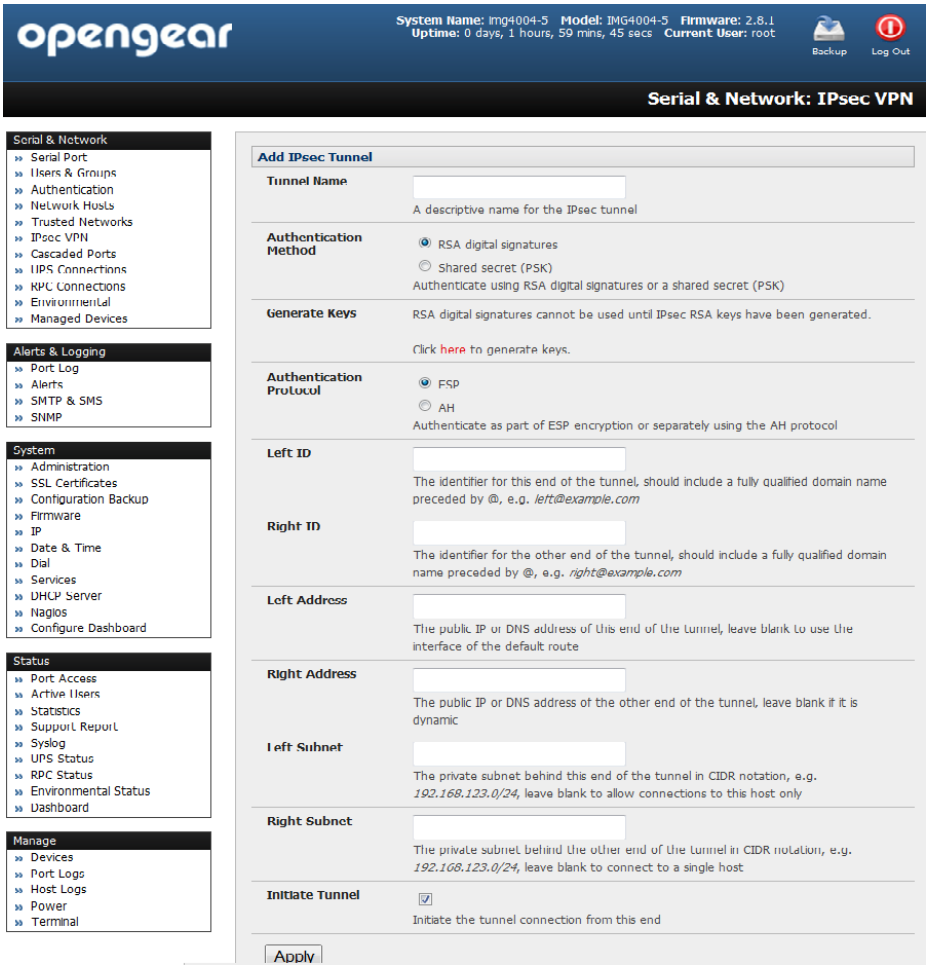
4.9.1 Enable the VPN gateway

- Select **IPsec VPN** on the **Serial & Networks** menu

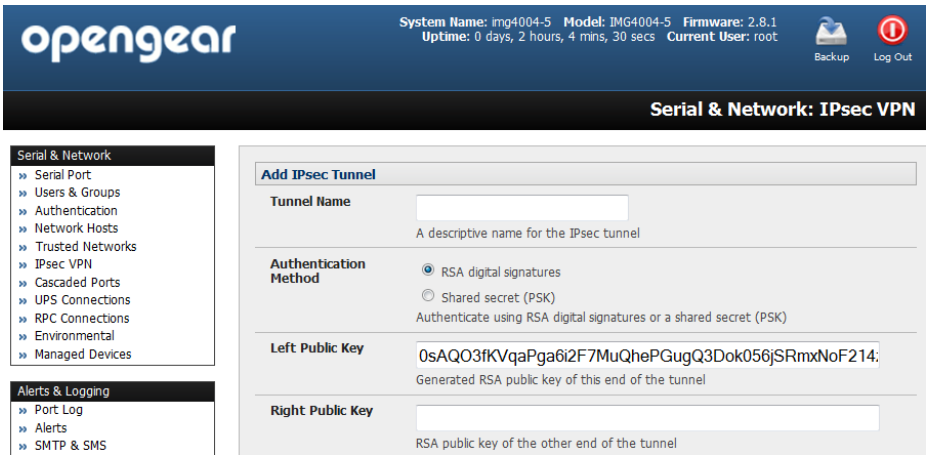


- Click **Add** and complete the *Add IPsec Tunnel* screen

- Enter any descriptive name you wish to identify the IPsec Tunnel you are adding such as *WestStOutlet-VPN*



- Select the **Authentication Method** to be used, either *RSA digital signatures* or a *Shared secret (PSK)*
  - If you select *RSA* you will asked to *click here to generate keys*. This will generate an RSA public key for the console server (the *Left Public Key*). You will need to find out the key to be used on the remote gateway, then cut and paste it into the *Right Public Key*



  - If you select *Shared secret* you will need to enter a Pre-shared secret (PSK). The PSK must match the PSK configured at the other end of the tunnel

- In **Authentication Protocol** select the authentication protocol to be used. Either authenticate as part of *ESP* (Encapsulating Security Payload) encryption or separately using the *AH* (Authentication Header) protocol.

- Enter a **Left ID** and **Right ID**. This is the identifier that the Local host/gateway and remote host/gateway use for IPsec negotiation and authentication. Each ID must include an '@' and can include a fully qualified domain name preceded by '@' (e.g. *left@example.com*)

- Enter the public IP or DNS address of this Opengear VPN gateway (or if not an ACM5004-G or ACM5504-5-G-I enter the address of the gateway device connecting it to the Internet) as the **Left Address**. You can leave this blank to use the interface of the default route

- In **Right Address** enter the public IP or DNS address of the remote end of the tunnel (only if the remote end has a static or dyndns address). Otherwise leave this blank

- If the Opengear VPN gateway is serving as a VPN gateway to a local subnet (e.g. the *console server* has a Management LAN configured) enter the private subnet details in **Left Subnet.** Use the CIDR notation (where the IP address number is followed by a slash and the number of 'one' bits in the binary notation of the netmask). For example 192.168.0.0/24 indicates an IP address where the first 24 bits are used as the network address. This is the same as 255.255.255.0. If the VPN access is only to the console server itself and to its attached serial console devices then leave **Left Subnet** blank

- If there is a VPN gateway at the remote end, enter the private subnet details in **Right Subnet**. Again use the CIDR notation and leave blank if there is only a remote host

- Select **Initiate Tunnel** if the tunnel connection is to be initiated from the Left console server end. This can only be initiated from the VPN gateway (Left) if the remote end was configured with a static (or dyndns) IP address

- Click **Apply** to save changes

Note It is essential the configuration details set up on the advanced *console server* (referred to as the Left or Local host) exactly matches the set up entered when configuring the Remote (Right) host/gateway or software client. Refer to the *http://www.opengear.com/faq.html* for details on configuring these remote ends

4.10 OpenVPN

The ACM7000, ACM5500, ACM5000, CM7100, IM7200 and IM4200 family of advanced *console servers* with Firmware V3.2 and later, include OpenVPN which is based on TSL (Transport Layer Security) and SSL (Secure Socket Layer). With OpenVPN, it is easy to build cross-platform, point-to-point VPNs using x509 PKI (Public Key Infrastructure) or custom configuration files.

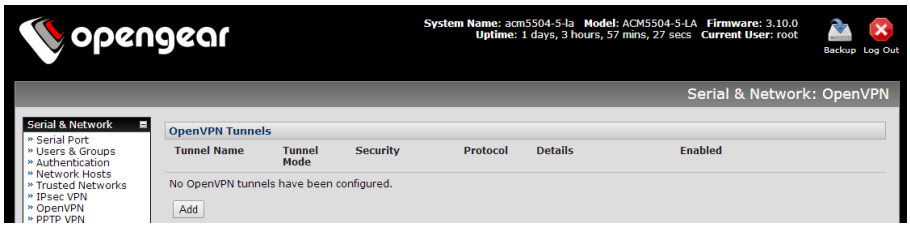
OpenVPN allows secure tunneling of data through a single TCP/UDP port over an unsecured network, thus providing secure access to multiple sites and secure remote administration to a console server over the Internet.

OpenVPN also allows the use of Dynamic IP addresses by both the server and client thus providing client mobility. For example, an OpenVPN tunnel may be established between a roaming windows client and an Opengear advanced *console server* within a data center.

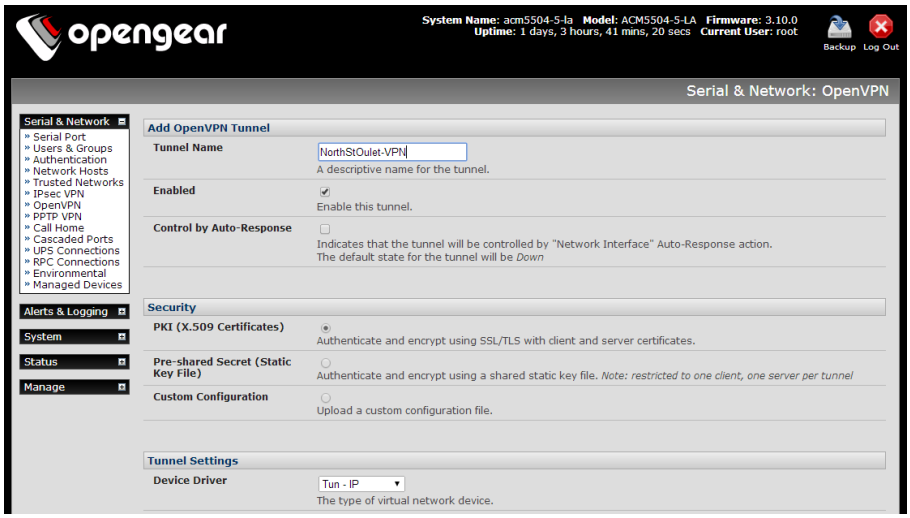
Configuration of OpenVPN can be complex so Opengear provides a simple GUI interface for basic set up as described below. However for more detailed information on configuring OpenVPN Access server or client refer to the HOW TO and FAQs at **http://www.openvpn.net**

4.10.1 Enable the OpenVPN

- Select **OpenVPN** on the **Serial & Networks** menu



- Click **Add** and complete the *Add OpenVPN Tunnel* screen
- Enter any descriptive name you wish to identify the OpenVPN Tunnel you are adding, for example *NorthStOutlet-VPN*



-
- Select the authentication method to be used. To authenticate using certificates select *PKI (X.509 Certificates)* or select *Custom Configuration* to upload custom configuration files. Custom configurations must be stored in /etc/config.

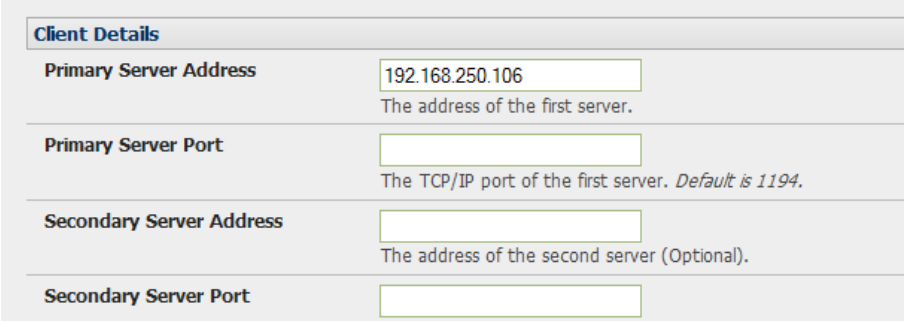
Note: If you select PKI (public key infrastructure) you will need to establish:

- Separate certificate (also known as a public key). This *Certificate File* will be a **.crt* file type
- Private Key for the server and each client. This *Private Key File* will be a **.key* file type
- Master Certificate Authority (CA) certificate and key which is used to sign each of the server and client certificates. This *Root CA Certificate* will be a **.crt* file type

For a server you may also need dh1024.pem (*Diffie Hellman* parameters). Refer http://openvpn.net/easyrsa.html for a guide to basic RSA key management. For alternative authentication methods see http://openvpn.net/index.php/documentation/howto.html#auth. For more information also see http://openvpn.net/howto.html

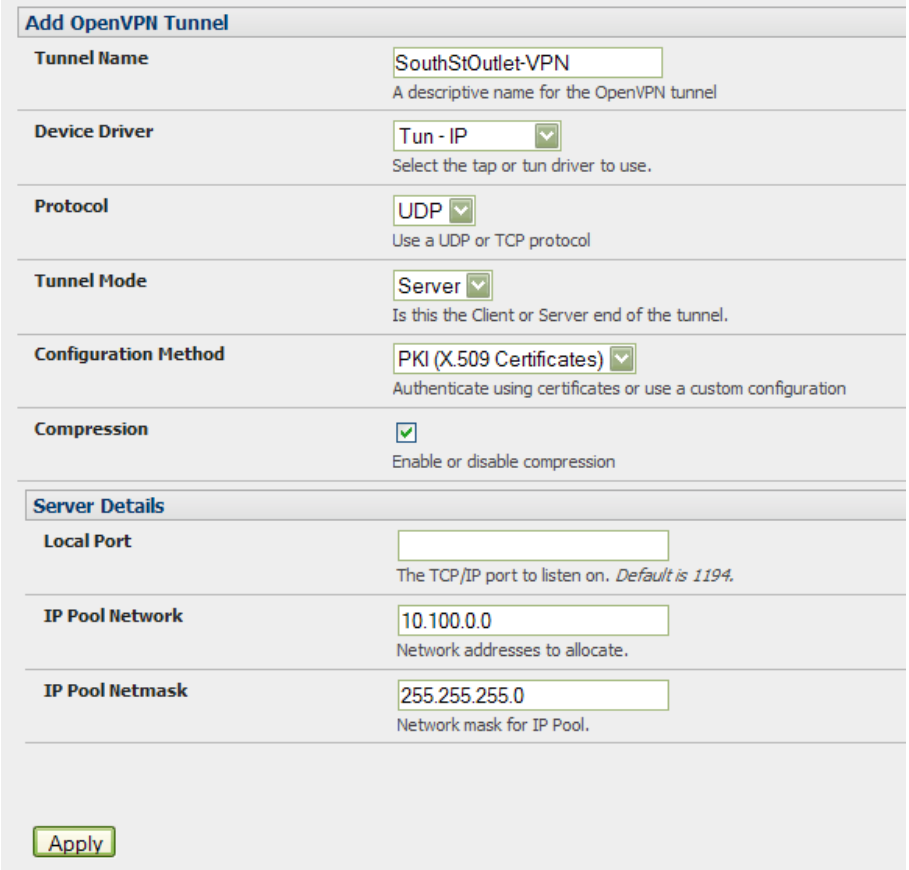
---

- Select the **Device Driver** to be used, either *Tun-IP* or *Tap-Ethernet*. The TUN (network tunnel) and TAP (network tap) drivers are virtual network drivers that support IP tunneling and Ethernet tunneling, respectively. TUN and TAP are part of the Linux kernel.
- Select either *UDP* or *TCP* as the **Protocol.** UDP is the default and preferred protocol for OpenVPN.
- In **Tunnel Mode,** nominate whether this is the *Client* or *Server* end of the tunnel. When running as a server, the advanced *console server* supports multiple clients connecting to the VPN server over the same port.
- Check or uncheck the **Compression** button to enable or disable compression, respectively

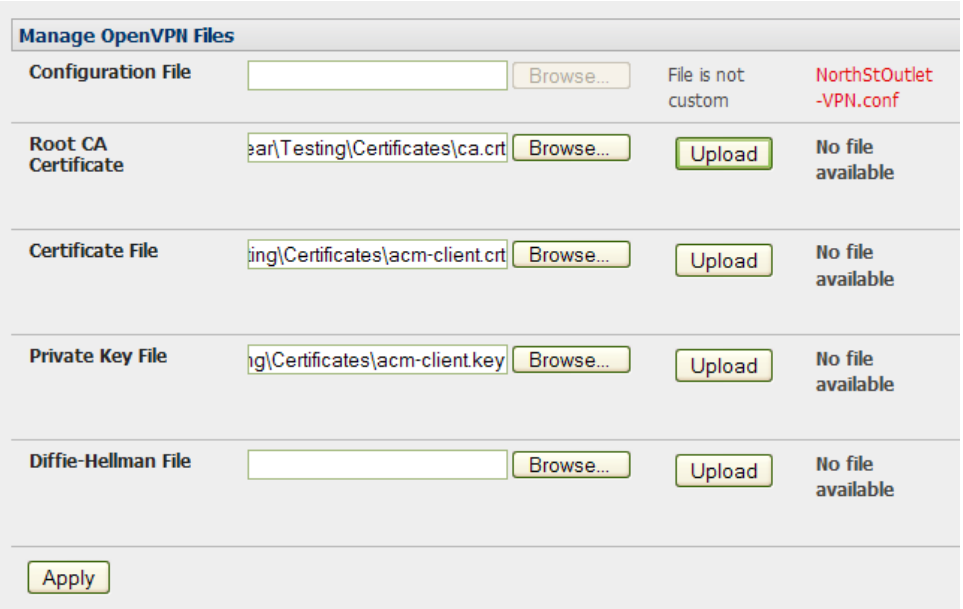


4.10.2 Configure as Server or Client

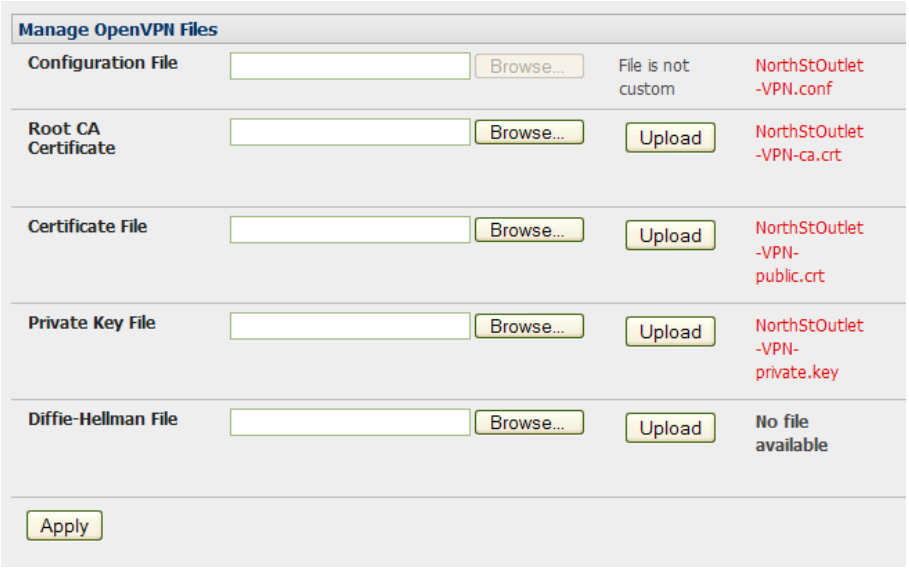
- Complete the **Client Details** or **Server Details** depending on the Tunnel Mode selected.
  - If *Client* has been selected, the *Primary Server Address* will be the address of the OpenVPN Server.
  - If *Server* has been selected, enter the IP Pool Network address and the IP Pool Network mask for the IP Pool. The network defined by the IP Pool Network address/mask is used to provide the addresses for connecting clients.
- Click **Apply** to save changes



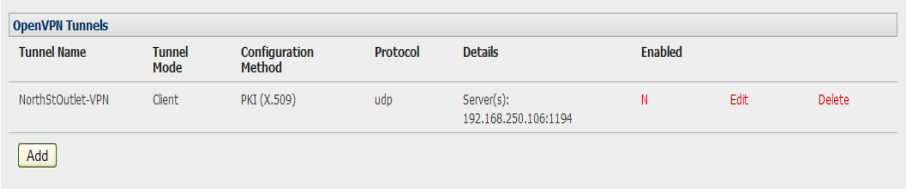
- To enter authentication certificates and files, **Edit** the OpenVPN tunnel.
- Select the **Manage OpenVPN Files** tab. Upload or browse to relevant authentication certificates and files.



- **Apply** to save changes. Saved files will be displayed in red on the right-hand side of the Upload button.

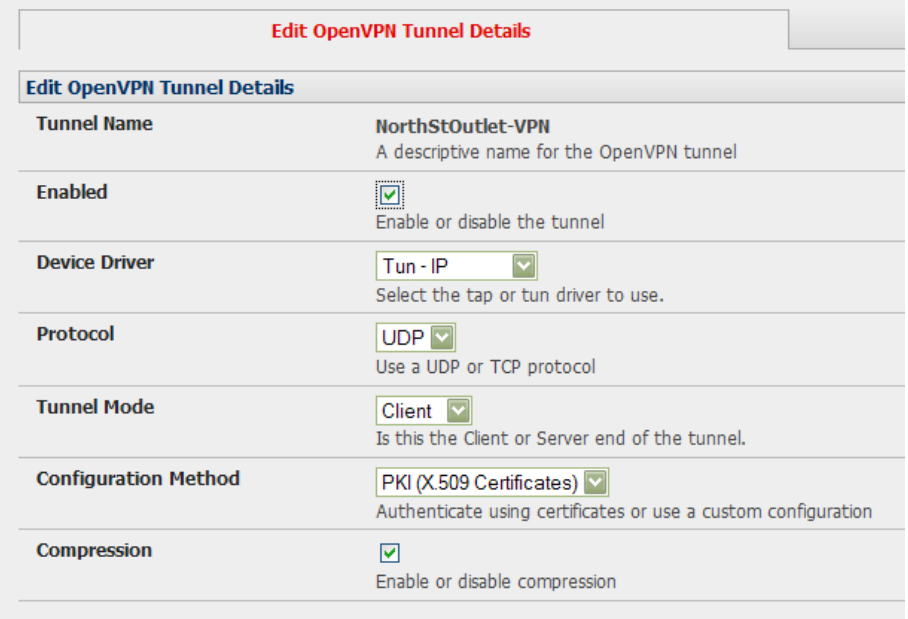


- To enable OpenVPN, **Edit** the OpenVPN tunnel

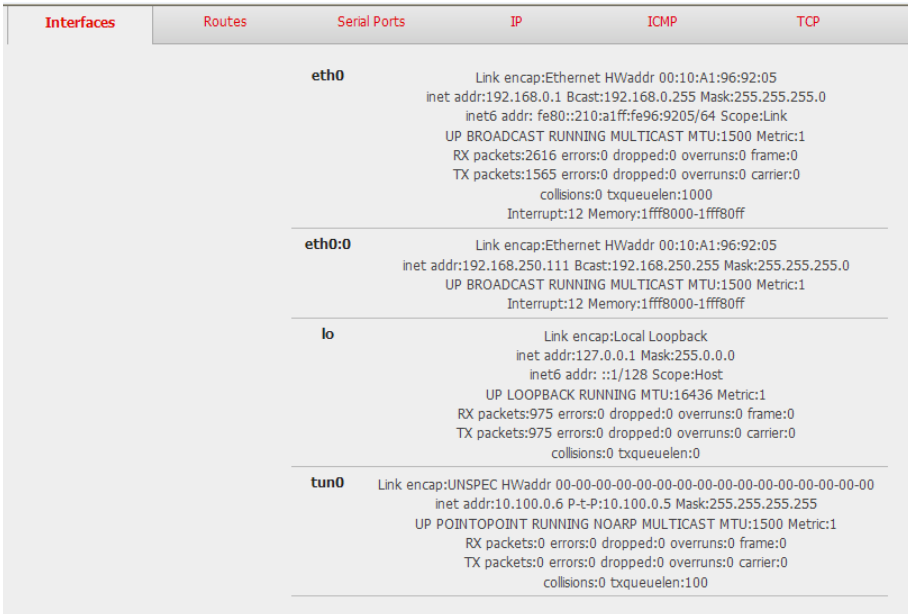


- Check the **Enabled** button.
- **Apply** to save changes

Note: Please make sure that the console server system time is correct when working with OpenVPN. Otherwise authentication issues may arise



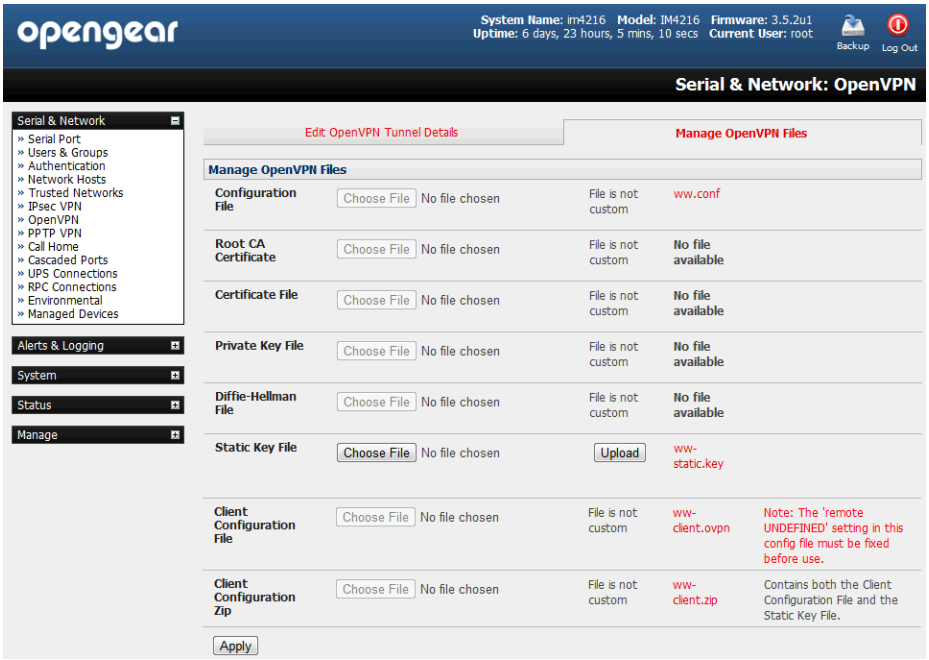
- Select **Statistics** on the **Status** menu to verify that the tunnel is operational.



4.10.3 Windows OpenVPN Client and Server set up

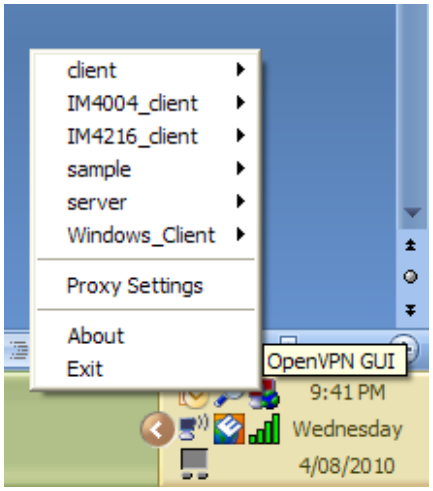
Windows does not come standard with any OpenVPN server or client. This section outlines the installation and configuration of a Windows OpenVPN client or a Windows OpenVPN server and setting up a VPN connection to a console server.

Console servers with firmware V3.5.2 and later will generate Windows client config automatically from the GUI – for **Pre-shared Secret (Static Key File)** configurations.



Alternately *OpenVPN GUI for Windows* software (which includes the standard OpenVPN package plus a Windows GUI) can be downloaded from *http://openvpn.se/download.html*.

- Once installed on the Windows machine, an OpenVPN icon will have been created in the Notification Area located in the right side of the taskbar. Right click on this icon to start (and stop) VPN connections, and to edit configurations and view logs



When the OpenVPN software is started, the *C:\Program Files\OpenVPN\config* folder will be scanned for "*.opvn*" files. This folder will be rechecked for new configuration files whenever the OpenVPN GUI icon is right-clicked. So once OpenVPN is installed, a configuration file will need to be created:

- Using a text editor, create an *xxxx.ovpn* file and save *in C:\Program Files\OpenVPN\config*. For example, *C:\Program Files\OpenVPN\config\client.ovpn*

  An example of an OpenVPN Windows client configuration file is shown below:

  # description: IM4216_client
  client
  proto udp
  verb 3
  dev tun
  remote 192.168.250.152

port 1194
ca c:\\openvpnkeys\\ca.crt
cert c:\\openvpnkeys\\client.crt
key c:\\openvpnkeys\\client.key
nobind
persist-key
persist-tun
comp-lzo

An example of an OpenVPN Windows Server configuration file is shown below:

server 10.100.10.0 255.255.255.0
port 1194
keepalive 10 120
proto udp
mssfix 1400
persist-key
persist-tun
dev tun
ca c:\\openvpnkeys\\ca.crt
cert c:\\openvpnkeys\\server.crt
key c:\\openvpnkeys\\server.key
dh c:\\openvpnkeys\\dh.pem
comp-lzo
verb 1
syslog IM4216_OpenVPN_Server

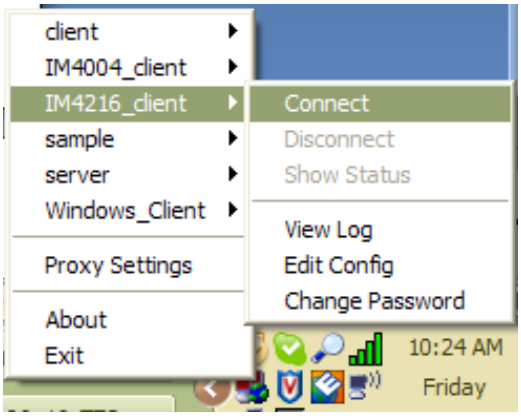
The Windows client/server configuration file options are:

Options	*Description*
#description:	This is a comment describing the configuration. Comment lines start with'#' and are ignored by OpenVPN.
Client server	Specify whether this will be a client or server configuration file. In the server configuration file, define the IP address pool and netmask. For example, server 10.100.10.0 255.255.255.0
proto udp proto tcp	Set the protocol to UDP or TCP. The client and server must use the same settings.
mssfix <max. size>	Mssfix sets the maximum size of the packet. This is only useful for UDP if problems occur.
verb <level>	Set log file verbosity level. Log verbosity level can be set from 0 (minimum) to 15 (maximum). For example, 0 = silent except for fatal errors 3 = medium output, good for general usage 5 = helps with debugging connection problems 9 = extremely verbose, excellent for troubleshooting
dev tun dev tap	Select 'dev tun' to create a routed IP tunnel or 'dev tap' to create an Ethernet tunnel. The client and server must use the same settings.
remote <host>	The hostname/IP of OpenVPN server when operating as a client. Enter either the DNS hostname or the static IP address of the server.
Port	The UDP/TCP port of the server.
Keepalive	Keepalive uses ping to keep the OpenVPN session alive. 'Keepalive 10 120' pings every 10 seconds and assumes the remote peer is down if no ping has been received over a 120 second time period.
http-proxy <proxy server> <proxy port #>	If a proxy is required to access the server, enter the proxy server DNS name or IP and port number.
ca <file name>	Enter the CA certificate file name and location. The same CA certificate file can be used by the server and all clients. Note: Ensure each '\' in the directory path is replaced with ' \\'. For example, c:\openvpnkeys\ca.crt will become c:\\openvpnkeys\\ca.crt
cert <file name>	Enter the client's or server's certificate file name and location. Each client should have its own certificate and key files. Note: Ensure each '\' in the directory path is replaced with ' \\'.

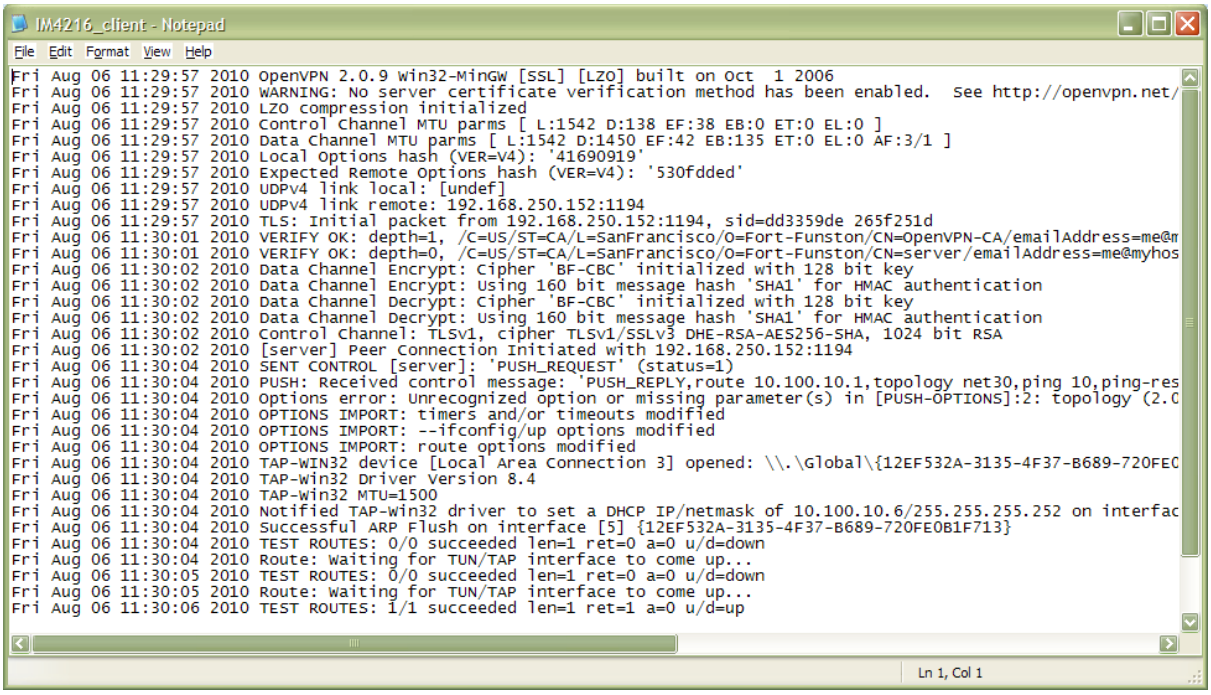
key <file name>	Enter the file name and location of the client's or server's key. Each client should have its own certificate and key files. Note: Ensure each '\' in the directory path is replaced with ' \\'.
dh <file name>	This is used by the server only. Enter the path to the key with the Diffie-Hellman parameters.
Nobind	'Nobind' is used when clients do not need to bind to a local address or specific local port number. This is the case in most client configurations.
persist-key	This option prevents the reloading of keys across restarts.
persist-tun	This option prevents the close and reopen of TUN/TAP devices across restarts.
cipher BF-CBC Blowfish (default) cipher AES-128-CBC AES cipher DES-EDE3-CBC Triple-DES	Select a cryptographic cipher. The client and server must use the same settings.
comp-lzo	Enable compression on the OpenVPN link. This must be enabled on both the client and the server.
syslog	By default, logs are located in syslog or, if running as a service on Window, in \Program Files\OpenVPN\log directory.

To initiate the OpenVPN tunnel following the creation of the client/server configuration files:

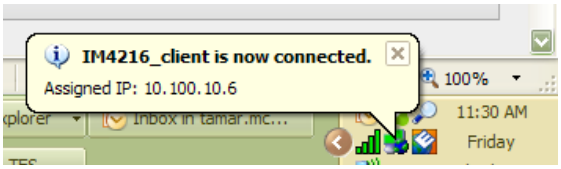
- Right click on the OpenVPN icon in the Notification Area
- Select the newly created client or server configuration. For example, IM4216_client
- Click 'Connect' as shown below



- The log file will be displayed as the connection is established



- Once established, the OpenVPN icon will display a message notifying of the successful connection and assigned IP. This information, as well as the time the connection was established, is available anytime by scrolling over the OpenVPN icon.



Note: An alternate OpenVPN Windows client can be downloaded from *http://www.openvpn.net/index.php/openvpn-client/downloads.html*. Refer to *http://www.openvpn.net/index.php/openvpn-client/howto-openvpn-client.html* for help

4.11 PPTP VPN

The ACM7000, ACM5500, ACM5000, CM7100, IM7200 and IM4200 family of advanced *console servers* with Firmware V3.5.2 and later, include a PPTP (Point-to-Point Tunneling Protocol) server. PPTP is typically used for communications over a physical or virtual serial link. The PPP endpoints define a virtual IP address to themselves. Routes to networks can then be defined with these IP addresses as the gateway, which results in traffic being sent across the tunnel. PPTP establishes a tunnel between the physical PPP endpoints and securely transports data across the tunnel.

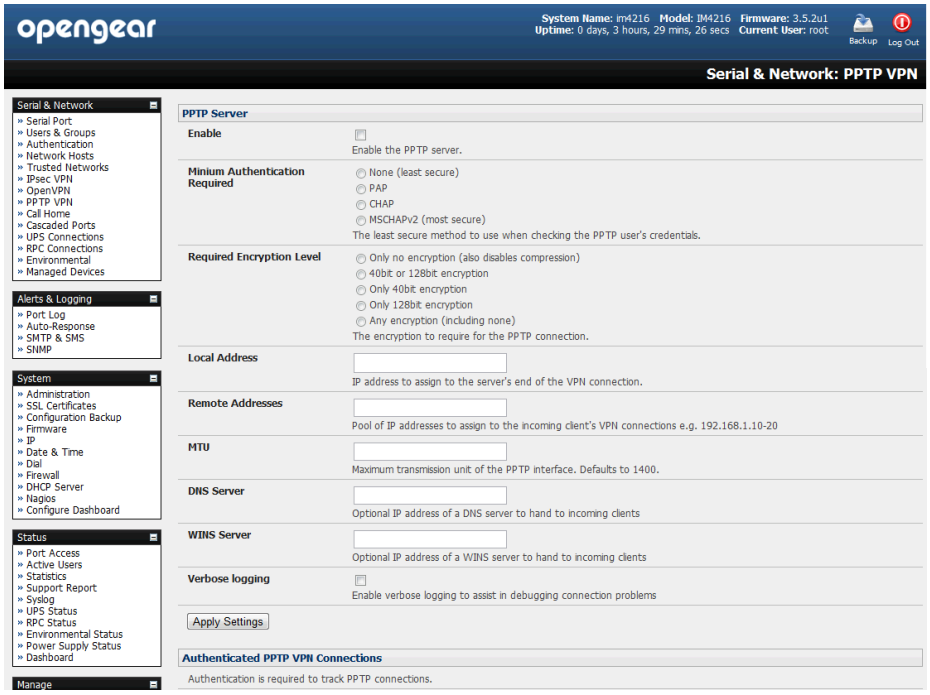
The strength of PPTP is its ease of configuration and integration into existing Microsoft infrastructure. It is generally used for connecting single remote Windows clients. If you take your portable computer on a business trip, you can dial a local number to connect to your Internet access service provider (ISP) and then create a second connection (*tunnel*) into your office network across the Internet and have the same access to your corporate network as if you were connected directly from your office. Similarly, telecommuters can also set up a VPN tunnel over their cable modem or DSL links to their local ISP.

To set up a PPTP connection from a remote Windows client to your Opengear appliance and local network:

1. Enable and configure the PPTP VPN server on your Opengear appliance
2. Set up VPN user accounts on the Opengear appliance and enable the appropriate authentication
3. Configure the VPN clients at the remote sites. The client does not require special software as the PPTP Server supports the standard PPTP client software included with Windows NT and later
4. Connect to the remote VPN

4.11.1 Enable the PPTP VPN server

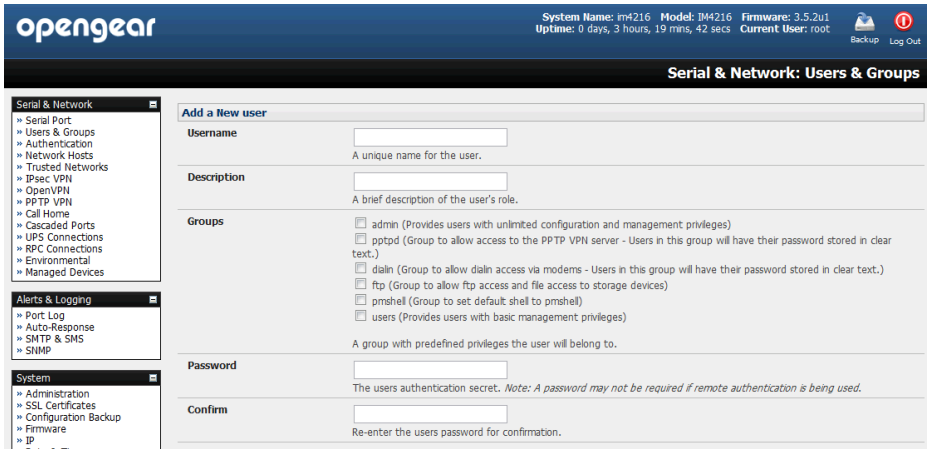
- Select **PPTP VPN** on the **Serial & Networks** menu



- Select the **Enable** check box to enable the PPTP Server
- Select the **Minimum Authentication Required**. Access is denied to remote users attempting to connect using an authentication scheme weaker than the selected scheme. The schemes are described below, from strongest to weakest.
  - **Encrypted Authentication (MS-CHAP v2):** The strongest type of authentication to use; this is the recommended option
  - **Weakly Encrypted Authentication (CHAP):** This is the weakest type of encrypted password authentication to use. It is not recommended that clients connect using this as it provides very little password protection. Also note that clients connecting using CHAP are unable to encrypt traffic
  - **Unencrypted Authentication (PAP):** This is plain text password authentication. When using this type of authentication, the client password is transmitted unencrypted.
  - **None**
- Select the **Required Encryption Level**. Access is denied to remote users attempting to connect not using this encryption level. Strong **40 bit or 128 bit encryption** is recommended
- In **Local Address** enter IP address to assign to the server's end of the VPN connection
- In **Remote Addresses** enter the pool of IP addresses to assign to the incoming client's VPN connections (e.g. 192.168.1.10-20). This must be a free IP address (or a range of free IP addresses), from the network (typically the LAN) that remote users are assigned while connected to the Opengear appliance
- Enter the desired value of the Maximum Transmission Unit (MTU) for the PPTP interfaces into the **MTU** field (defaults to 1400)
- In the **DNS Server** field, enter the IP address of the DNS server that assigns IP addresses to connecting PPTP clients
- In the **WINS Server** field, enter the IP address of the WINS server that assigns IP addresses to connecting PPTP client
- Enable **Verbose Logging** to assist in debugging connection problems
- Click **Apply Settings**

4.11.2 Add a PPTP user

- Select **Users & Groups** on the **Serial & Networks** menu and complete the fields as covered in section 4.2.
- Ensure the *pptpd* **Group** has been checked, to allow access to the PPTP VPN server. Note - users in this group will have their password stored in clear text.
- Keep note of the username and password for when you need to connect to the VPN connection
- Click **Apply**

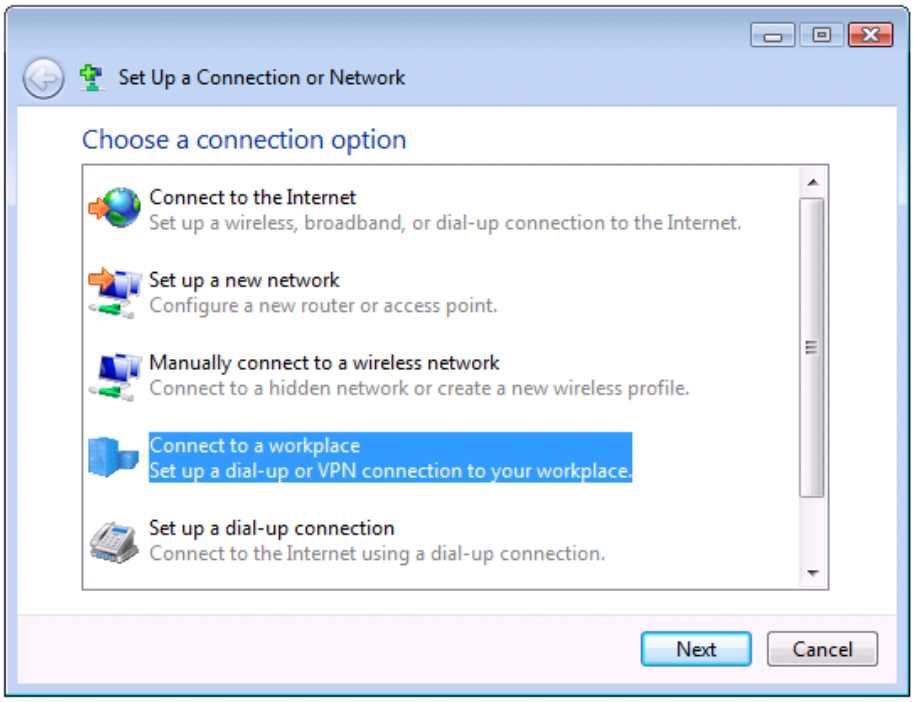


4.11.3 Set up a remote PPTP client

Ensure the remote VPN client PC has Internet connectivity. To create a VPN connection across the Internet, you must set up two networking connections. One connection is for the ISP, and the other connection is for the VPN tunnel to the Opengear appliance.

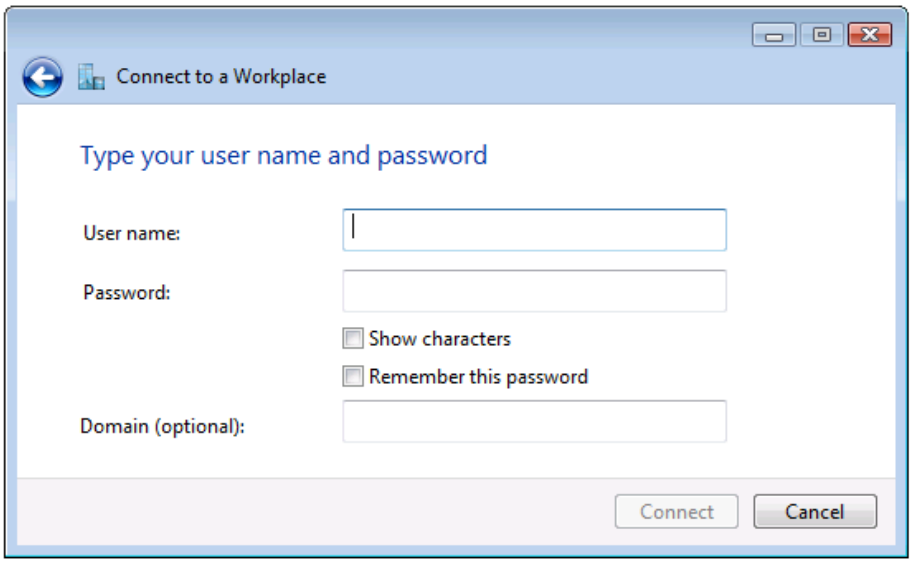
Note: This procedure sets up a PPTP client in the Windows 7 Professional operating system. The steps may vary slightly depending on your network access or if you are using an alternate version of Windows. More detailed instructions are available from the Microsoft web site.

- Login to your Windows client with administrator privileges
- From the **Network & Sharing Center** on the **Control Panel** select **Network Connections** and create a new connection



- Select **Use My Internet Connection (VPN)** and enter the IP Address of the Opengear appliance

Note: To connect remote VPN clients to the local network, you need to know the user name and password for the PPTP account you added, as well as the Internet IP address of the Opengear appliance. If your ISP has not allocated you a static IP address, consider using a dynamic DNS service. Otherwise you must modify the PPTP client configuration each time your Internet IP address changes.



4.12 Call Home

All *console servers* with Firmware V3.2 and later, include the *Call Home* feature which initiates the setup of a secure SSH tunnel from the *console server* to a centralized Lighthouse VM, Lighthouse Standard, Lighthouse Enterprise, CMS6100 or

VCMS server (referred to herein as *CMS*). The *console server* then registers as a "candidate" on the *CMS* - and once accepted there it becomes a **Managed Console Server**.

The CMS will then monitor the Managed Console Server, and administrators can access the remote Managed Console Server, through the *CMS.* This access is available even when the remote *console server* is behind a third party firewall or has a private non-routable IP addresses (which is often the case when the *console server* is connected via a cellular modem connection).

Note *CMS* maintains public key authenticated SSH connections to each of its Managed Console Servers. These connections are used for monitoring, commanding and accessing the Managed Console Servers and the Managed Devices connected to the Managed Console Server.

To manage Local Console Servers, or console servers that are reachable from the *CMS*, the SSH connections are initiated by *CMS*.

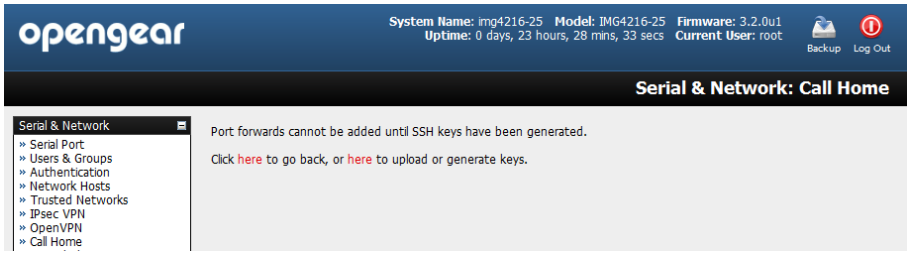
To manage Remote Console Servers, or console servers that are firewalled, not routable, or otherwise unreachable from the *CMS*, the SSH connections are initiated by the Managed Console Server via an initial Call Home connection.

This ensures secure, authenticated communications and enables Managed Console Servers units to be distributed locally on a LAN, or remotely around the world.

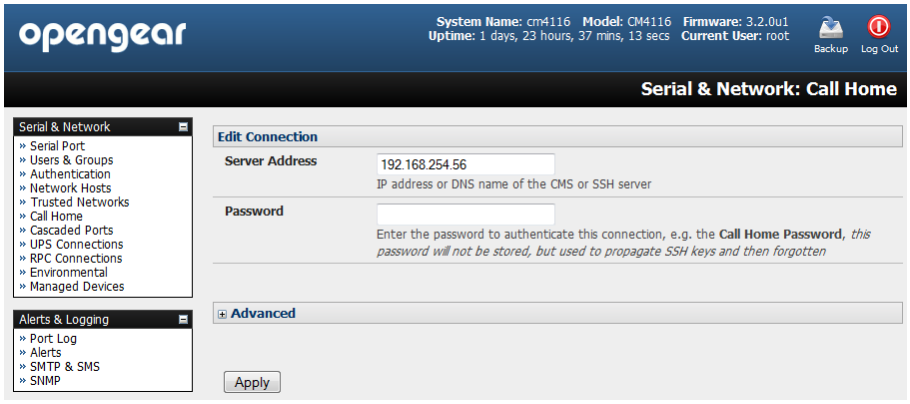
4.12.1 Set up Call Home candidate

To set up the *console server* as a Call Home management candidate on the *CMS*:

- Select **Call Home** on the **Serial & Network** menu



- If you have not already generated or uploaded an SSH key pair for this *console server,* you will need to do so before proceeding (refer Chapter 3)
- Click **Add**



- Enter the IP address or DNS name (e.g. the dynamic DNS address) of the *CMS*
- Enter the Password that you configured on the CMS as the **Call Home Password**
- Click **Apply**

These steps initiate the Call Home connection from the *console server* to the *CMS*. This creates an SSH listening port on the *CMS*, and sets the *console server* up as a candidate.

Once the candidate has been accepted on the *CMS* (as outlined in the next section) an SSH tunnel to the *console server* is then redirected back across the Call Home connection. The *console server* has now become a Managed Console Server and the *CMS* can connect to and monitor it through this tunnel.

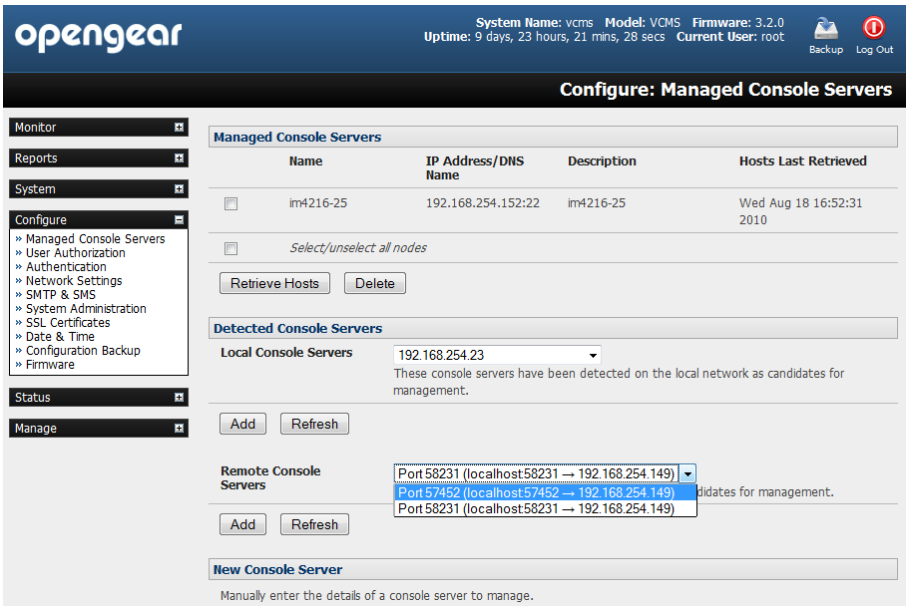
4.12.2 Accept Call Home candidate as Managed Console Server on CMS

This section gives an overview on configuring the *CMS* to monitor *console servers* that are connected via Call Home. For more details refer to the Lighthouse CMS User Manual:

1. You first must enter a new **Call Home Password** on the *CMS*. This password is used solely for accepting Call Home connections from candidate *console servers*

2. So the *CMS* can be contacted by the *console server* it must either have a static IP address or, if using DHCP, be configured to use a dynamic DNS service

3. The **Configure: Managed Console Servers** screen on the *CMS* shows the status of local and remote Managed Console Servers and candidates.



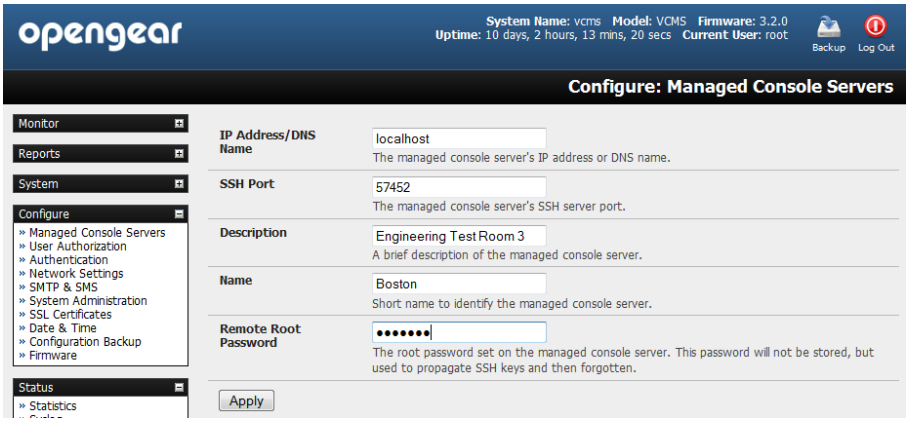
The *Managed Console Server* section shows the console servers currently being monitored by the CMS.

The *Detected Console Servers* section:

- The **Local Console Servers** drop down list in lists all the console servers which are on the same subnet as the *CMS*, and are not currently being monitored

- The **Remote Console Servers** drop down list in the *Detected Console Servers* section lists all the console servers that have established a Call Home connection, and are not currently being monitored (i.e. candidates). You can click **Refresh** to update

4. To add a *console server* candidate to the **Managed Console Server** list:
   - Select it from the *Remote Console Servers* drop down list, and click **Add**
   - Enter IP Address and SSH Port (if these fields have not been auto-completed) and enter a **Description** and unique **Name** for the *Managed Console Server* you are adding



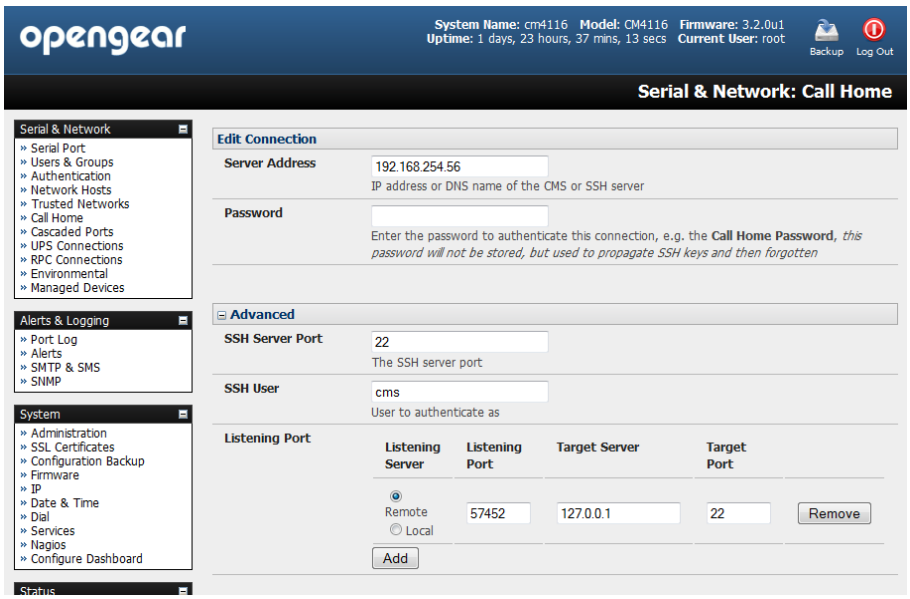
   - Enter the **Remote *Root* Password** (i.e. System Password that has been set on this *Managed Console Server).* This password is used by the *CMS* to propagate auto generated SSH keys and then forgotten. It will not be stored

   - Click **Apply**. The *CMS* will now set up secure SSH connections to and from the Managed Console Server and will retrieve its Managed Devices, user account details and configured alerts

4.12.3 Calling Home to a generic central SSH server

If you are connecting to a generic SSH server (not a Lighthouse CMS) you may configure *Advanced* settings:

- Enter the **SSH Server Port** and SSH User to authenticate as
- Enter the details for the SSH port forward(s) to create



By selecting *Listening Server*, you may create a **Remote** port forward from the Server to this unit, or a **Local** port forward from this unit to the Server:

- Specify a Listening Port to forward from, leave this field blank to allocate an unused port
- Enter the Target Server and Target Port that will be the recipient of forwarded connections

4.13 IP Passthrough

IP Passthrough is used to make a modem connection (e.g. the Opengear's internal cellular modem) appear like a regular Ethernet connection to a third-party downstream router, allowing the downstream router to use the Opengear's modem connection as a primary or backup WAN interface.

The Opengear provides the modem IP address and DNS details to the downstream device over DHCP and transparently passes network traffic to and from the modem and router.

While IP Passthrough essentially turns an Opengear into a modem-to-Ethernet half bridge, some specific layer 4 services (HTTP/HTTPS/SSH) may still be terminated at the Opengear (*Service Intercepts*). Also, services running on the Opengear can initiate outbound cellular connections independent of the downstream router.

This allows the Opengear to continue to be used for out-of-band management and alerting and also be managed via Lighthouse, while in IP Passthrough mode.

4.13.1 Downstream Router Setup

To use failover connectivity on the downstream router (aka *Failover to Cellular* or *F2C*), it must have two or more WAN interfaces.

Note *Failover* in IP Passthrough context is performed entirely by the downstream router, and the built-in out-of-band failover logic on the Opengear itself is not available while in IP Passthrough mode.

Connect an Ethernet WAN interface on the downstream router to the Opengear's *Network Interface* or *Management LAN* port with an Ethernet cable.

Configure this interface on the downstream router to receive its network settings via DHCP. If failover is required, configure the downstream router for failover between its primary interface and the Ethernet port connected to the Opengear.

4.13.1 IP Passthrough Pre-Configuration

Prerequisite steps to enable IP Passthrough are:

- Configure the *Network Interface* and where applicable *Management LAN* interfaces with static network settings
    - Click **Serial & Network: IP**
    - For **Network Interface** and where applicable **Management LAN**, select **Static** for the **Configuration Method** and enter the network settings (see the section entitled *Network Configuration* for detailed instructions)
    - For the interface connected to the downstream router, you may choose any dedicated private network – this network will only exist between the Opengear and downstream router and will not normally be accessible
    - For the other interface, configure it as you would per normal on the local network
    - For both interfaces, leave **Gateway** blank
- Configure the Opengear modem in *Always On Out-of-band* mode

- For a cellular connection, click **System: Dial: Internal Cellular Modem**
- Select **Enable Dial-Out** and enter carrier details such as **APN** (see the section entitled *Cellular Modem Connection* for detailed instructions)

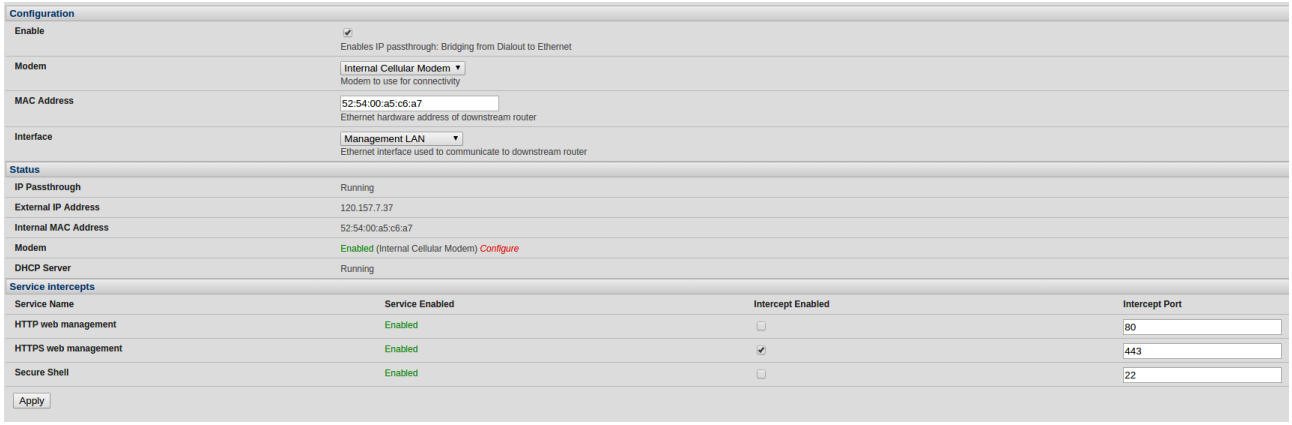
4.13.2 IP Passthrough Configuration

To configure IP Passthrough:

- Click **Serial & Network: IP Passthrough** and check **Enable**
- Select the Opengear **Modem** to use for upstream connectivity
- Optionally, enter the **MAC Address** of downstream router's connected interface

Note If MAC address is not specified, the Opengear will passthrough to the first downstream device requesting a DHCP address.

- Select the Opengear Ethernet **Interface** to use for connectivity to the downstream router
- Click **Apply**



4.13.3 Service Intercepts

These allow the Opengear to continue to provide services for e.g. out-of-band management when in IP Passthrough mode. Connections to the modem address on the specified intercept port(s) will be handled by the Opengear, rather than being passed through to the downstream router.

- For the required service of **HTTP**, **HTTPS** or **SSH**, check **Enable**
- Optionally modify the **Intercept Port** to an alternate port (e.g. 8443 for HTTPS), this is useful if you want to continue to allow the downstream router to remain accessible via its regular port

4.13.4 IP Passthrough Status

Refresh the page to view the **Status** section. It displays the modem's **External IP Address** being passed through, the **Internal MAC Address** of the downstream router (only populated when the downstream router accepts the DHCP lease), and the overall running status of the **IP Passthrough** service.

Additionally, you may be alerted to the failover status of the downstream router by configuring a **Routed Data Usage Check** under **Alerts & Logging: Auto-Response**.

4.13.5 Caveats

Some downstream routers may be incompatible with the gateway route. This may happen when IP Passthrough is bridging a 3G cellular network where the gateway address is a point-to-point destination address and no subnet information is available. The Opengear sends a DHCP netmask of 255.255.255.255. Devices will normally correctly construe this as a "single host route" on the interface, but as this is an unusual setting for Ethernet, some older downstream devices may have issues.

Intercepts for local services will not work if the Opengear is using a default route other than the modem. As per normal operation, they will also not work unless the service is enabled and access to the service is enabled (see **System: Services: Service Access: Dialout/Cellular**).

Outbound connections originating from Opengear to remote services are supported (e.g. sending SMTP email alerts, SNMP traps, getting NTP time, IPSec tunnels), however there is a miniscule risk of connection failure should both the Opengear and the downstream device try to access the same UDP or TCP port on the same remote host at the same time where they have randomly chosen the same originating local port number.

FIREWALL, FAILOVER & OOB ACCESS

The *console server* has a number of out-of-band access capabilities and transparent fail-over features, to ensure high availability. So if there's difficulty in accessing the *console server* through the main network path, all *console server* models provide out-of-band (OOB) access and the *Administrator* can still access it (and its Managed Devices) from a remote location.

- All *console server* models support serially attaching an external dial-up modem and configuring dial-in OOB access. Some models with USB ports support attaching an external USB modem. Some models also come standard with an internal modem. These modems can also be configured for dial-in OOB access
- All *console server* models with an internal or externally attached modem (and V3.4 firmware or later) can be configured for out-dial to be permanently connected
- The advanced *console server* models can also be configured for transparent out-dial failover. So in the event of a disruption in the principal management network, an external dial-up ppp connection is automatically established
- These *advanced console server* models can also be accessed out-of-band using an alternate broadband link and also offer transparent broadband failover
- Models with an internal cellular modem can be configured for OOB cellular access or for cellular transparent failover or can be configured as a cellular router

5.1 Dialup Modem Connection

To enable dial-in or dial-out you must first ensure there is a modem attached to the *console server*.

- All IM4200 and IM7200 models, ACM5508-2-M and ACM5003-M come with an internal modem which can provide for OOB dial-in access. These models will display an *Internal Modem Port* tab under *System -> Dial* (as well as the *Serial DB9 Port* tab)
- The other CM7100, ACM7000, ACM5500 and ACM5000 models also support external USB modems. The USB modem will be auto-detected and an *External USB Modem Port* tab will come up under *System -> Dial* (in addition to the *Serial DB9 Port* tab). All *console server* models supports an external modem (any brand) attached via a serial cable to the console/modem port for OOB dial-in access.
- SD4000 *console servers* need to have an external modem attached via a serial cable to their DB9 port. This port is marked *Local* and is located on the back of the SD4002 units.
- The serial ports on the ACM7000, ACM5500, ACM5000 and SD4001 are by default all configured as RJ serial *Console Server* ports. However Port 1 can be configured to be the *Local Console/Modem* port

5.2 OOB Dial-In Access

Once a modem has been attached to the *console server* you can configure the *console server* for dial-in PPP access. The *console server* will then await an incoming connection from a dial-in at remote site. Next the remote client dial-in software needs to be configured to establish the connection between the *Administrator's* client modem to the dial in modem on the *console server*.

5.2.1 Configure Dial-In PPP

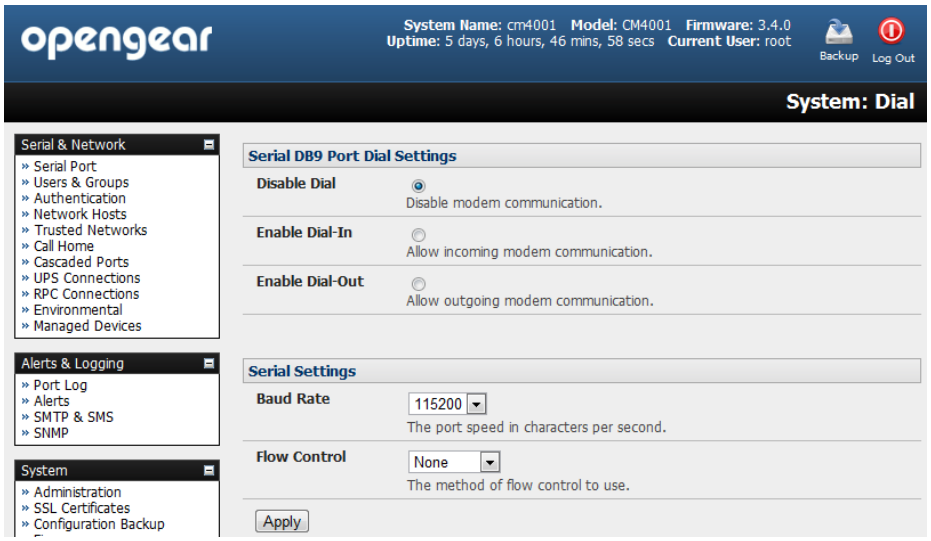
Enable PPP access on the internal or externally attached modem:

- Select the **System: Dial** menu option and the port to be configured (**Serial DB9 Port** or **Internal Modem Port** or **External USB Port**)
- Select the **Baud Rate** and **Flow Control** that will communicate with the modem

Note By default the modem port on all Opengear console servers is set with software flow control and the baud rate is set at:

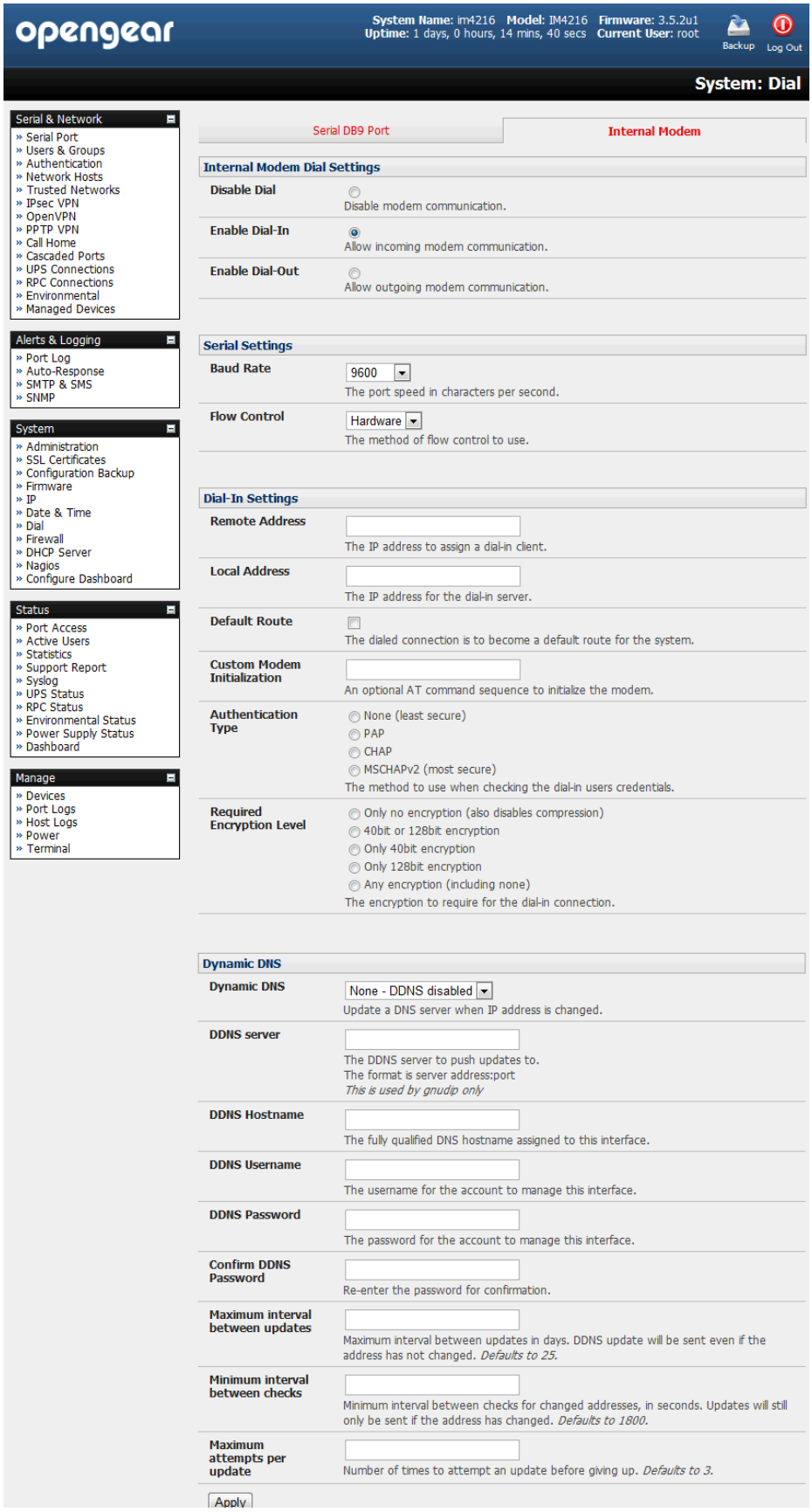
- 115200 baud for external modems connected to the local console port on CM7100, IM7200 and IM4200 console servers
- 9600 baud for the internal modem or external USB modem and for external modems connected to the Console serial ports which have been reassigned for dial-in access (on SD4001, SD4002, ACM5000, ACM5500 and ACM7000)

When enabling OOB dial-in it is recommended that the Serial Setting be changed to 38400 baud with Hardware Flow Control.



Note You can further configure the console/modem port (*e.g.* to include *modem init* strings) by editing */etc/mgetty.config* files as described in the *Chapter 14 - Advanced.*

- Check the **Enable Dial-In Access** box
- In the **Remote Address** field, enter the IP address to be assigned to the dial-in client. You can select any address for the Remote IP Address. However it must be in the same network range as the Local IP Address (*e.g.* 200.100.1.12 and 200.100.1.67)
- In the **Local Address** field enter the IP address for the Dial-In PPP Server. This is the IP address that will be used by the remote client to access *console server* once the modem connection is established. Again you can select any address for the Local IP Address but it must both be in the same network range as the Remote IP Address
- The **Default Route** option enables the dialed PPP connection to become the default route for the *console server*
- The **Custom Modem Initialization** option allows a custom AT string modem initialization string to be entered (*e.g.* AT&C1&D3&K3)


opengear
System Name: im4216 Model: IM4216 Firmware: 3.5.2u1
Uptime: 1 days, 0 hours, 14 mins, 40 secs Current User: root
Backup
Log Out
System: Dial
Serial & Network
» Serial Port
» Users & Groups
» Authentication
» Network Hosts
» Trusted Networks
» IPsec VPN
» OpenVPN
» PPTP VPN
» Call Home
» Cascaded Ports
» UPS Connections
» RPC Connections
» Environmental
» Managed Devices
Alerts & Logging
» Port Log
» Auto-Response
» SMTP & SMS
» SNMP
System
» Administration
» SSL Certificates
» Configuration Backup
» Firmware
» IP
» Date & Time
» Dial
» Firewall
» DHCP Server
» Nagios
» Configure Dashboard
Status
» Port Access
» Active Users
» Statistics
» Support Report
» Syslog
» UPS Status
» RPC Status
» Environmental Status
» Power Supply Status
» Dashboard
Manage
» Devices
» Port Logs
» Host Logs
» Power
» Terminal
Serial DB9 Port
Internal Modem
Internal Modem Dial Settings
Disable Dial
Disable modem communication.
Enable Dial-In
Allow incoming modem communication.
Enable Dial-Out
Allow outgoing modem communication.
Serial Settings
Baud Rate
9600
The port speed in characters per second.
Flow Control
Hardware
The method of flow control to use.
Dial-In Settings
Remote Address
The IP address to assign a dial-in client.
Local Address
The IP address for the dial-in server.
Default Route
The dialed connection is to become a default route for the system.
Custom Modem Initialization
An optional AT command sequence to initialize the modem.
Authentication Type
None (least secure)
PAP
CHAP
MSCHAPv2 (most secure)
The method to use when checking the dial-in users credentials.
Required Encryption Level
Only no encryption (also disables compression)
40bit or 128bit encryption
Only 40bit encryption
Only 128bit encryption
Any encryption (including none)
The encryption to require for the dial-in connection.
Dynamic DNS
Dynamic DNS
None - DDNS disabled
Update a DNS server when IP address is changed.
DDNS server
The DDNS server to push updates to.
The format is server address:port
This is used by gnudip only
DDNS Hostname
The fully qualified DNS hostname assigned to this interface.
DDNS Username
The username for the account to manage this interface.
DDNS Password
The password for the account to manage this interface.
Confirm DDNS Password
Re-enter the password for confirmation.
Maximum interval between updates
Maximum interval between updates in days. DDNS update will be sent even if the address has not changed. Defaults to 25.
Minimum interval between checks
Minimum interval between checks for changed addresses, in seconds. Updates will still only be sent if the address has changed. Defaults to 1800.
Maximum attempts per update
Number of times to attempt an update before giving up. Defaults to 3.
Apply

- Select the **Authentication Type** required. Access is denied to remote users attempting to connect using an authentication scheme weaker than the selected scheme. The schemes are described below, from strongest to weakest.
  - **Encrypted Authentication (MS-CHAP v2):** The strongest type of authentication to use; this is the recommended option
  - **Weakly Encrypted Authentication (CHAP):** This is the weakest type of encrypted password authentication to use. It is not recommended that clients connect using this as it provides very little password protection. Also note that clients connecting using CHAP are unable to encrypt traffic
  - **Unencrypted Authentication (PAP):** This is plain text password authentication. When using this type of authentication, the client password is transmitted unencrypted.
  - **None**
- Select the **Required Encryption Level**. Access is denied to remote users attempting to connect not using this encryption level. Strong **40 bit or 128 bit encryption** is recommended

Note: Firmware V3.5.2 and beyond support multiple dial-in users, who are setup with *dialin* Group membership. The **User name** and **Password** to be used for the dial-in PPP link, and any dial-back phone numbers are configured when the User is set up. Earlier firmware only supported one PPP dial-in account

Note Chapter 13 (Advanced Configurations) has examples of Linux commands that can be used to control the modem port operation at the command line level

5.2.2 Using SDT Connector client

Administrators can use their *SDT Connector* client to set up secure OOB dial-in access to remote *console servers*. The *SDT Connector* Java client software provides point-and-click secure remote access. OOB access uses an alternate path for connecting to the *console server* to that used for regular data traffic.

Starting an OOB connection in *SDT Connector* may be achieved by initiating a dial up connection, or adding an alternate route to the *console server*. *SDT Connector* allows for maximum flexibility is this regard, by allowing you to provide your own scripts or commands for starting and stopping the OOB connection. Refer *Chapter 6.5*

5.2.3 Set up Windows XP or later client

- Open **Network Connections** in Control Panel and click the **New Connection Wizard**
- Select **Connect to the Internet** and click **Next**
- On the **Getting Ready** screen select **Set up my connection manually** and click **Next**
- On the **Internet Connection** screen select **Connect using a dial-up modem** and click **Next**
- Enter a **Connection Name** (any name you choose) and the dial-up **Phone number** that will connect thru to the *console server* modem
- Enter the PPP **User name** and **Password** for have set up for the *console server*

5.2.4 Set up earlier Windows clients

- For Windows 2000, the PPP client set up procedure is the same as above, except you get to the **Dial-Up Networking Folder** by clicking the **Start** button and selecting **Settings.** Then click **Network and Dial-up Connections** and click **Make New Connection**
- Similarly for Windows 98 you double click **My Computer** on the Desktop, then open **Dial-Up Networking** and double click **Make New Connection** and proceed as above

5.2.5 Set up Linux clients

The online tutorial http://www.yolinux.com/TUTORIALS/LinuxTutorialPPP.html presents a selection of methods for establishing a dial up PPP connection:

- Command line PPP and manual configuration (which works with any Linux distribution)
- Using the *Linuxconf* configuration tool (for Red Hat compatible distributions). This configures the scripts *ifup/ifdown* to start and stop a PPP connection
- Using the Gnome control panel configuration tool -
- WVDIAL and the Redhat "Dialup configuration tool"
- GUI dial program X-isp. Download/Installation/Configuration

Note For all PPP clients:
- Set the PPP link up with TCP/IP as the only protocol enabled
- Specify that the Server will assign IP address and do DNS
- Do not set up the *console server* PPP link as the default for Internet connection

5.3 Dial-Out Access

The internal or externally attached modem on the *console server* can be set up either

- in *Failover mode* where a dial-out connection is only established in event of a *ping* failure, or
- with the dial-out connection always on

In both of the above cases in the event of a disruption in the dial-out connection, the console server will endeavor to re-establish the connection.

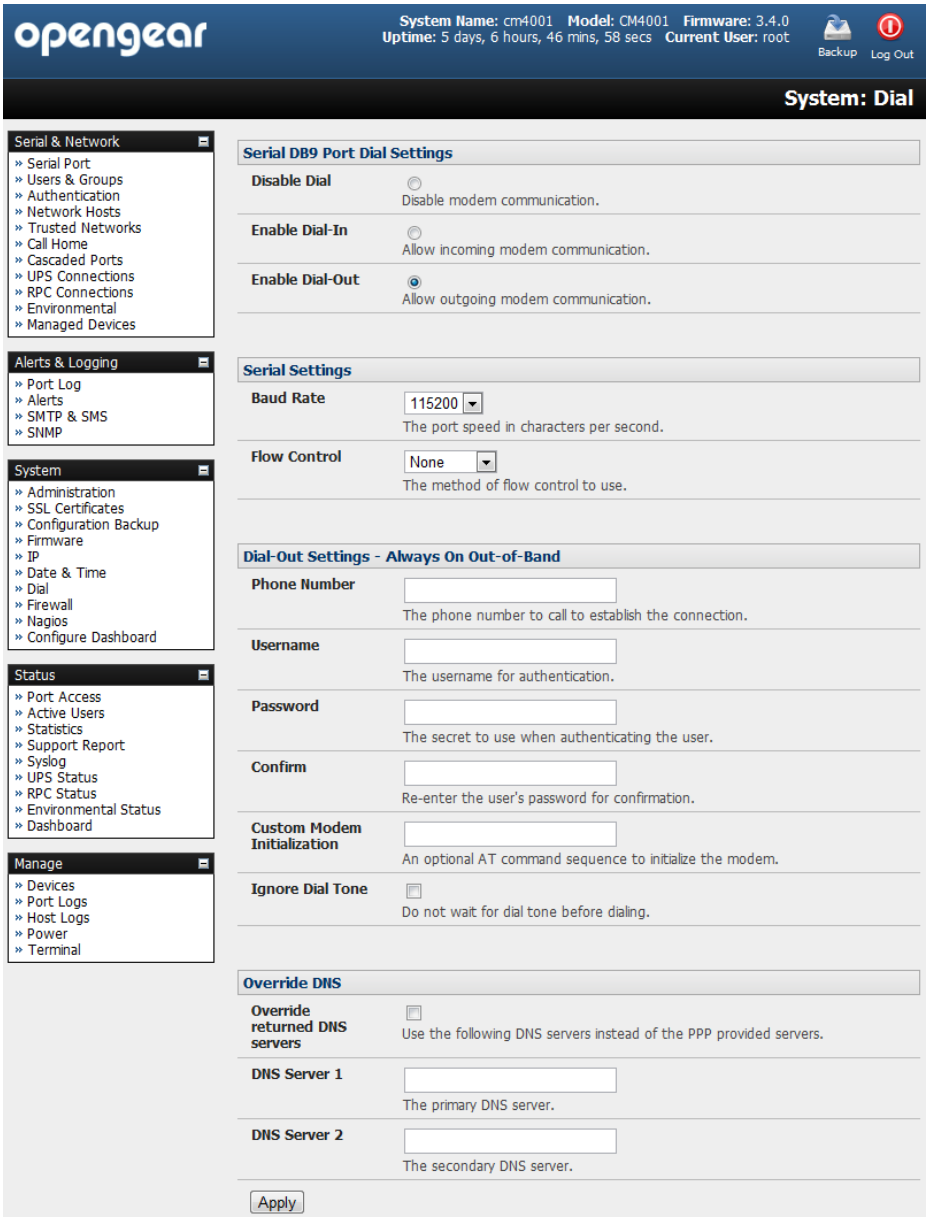
5.3.1 Always-on dial-out

With V3.4 firmware (and later) the *console server* modem can be configured for out-dial to be always on, with a permanent external dial-up ppp connection.

- Select the **System: Dial** menu option and check **Enable Dial-Out** to allow outgoing modem communications
- Select the **Baud Rate** and **Flow Control** that will communicate with the modem
- In the **Dial-Out Settings - Always On Out-of-Band** field enter the access details for the remote PPP server to be called

Override DNS is available for PPP Devices such as modems. Override DNS allows the use of alternate DNS servers from those provided by your ISP. For example, an alternative DNS may be required for OpenDNS used for content filtering.

- To enable **Override DNS**, check the Override returned DNS Servers box. Enter the IP of the DNS servers into the spaces provided.

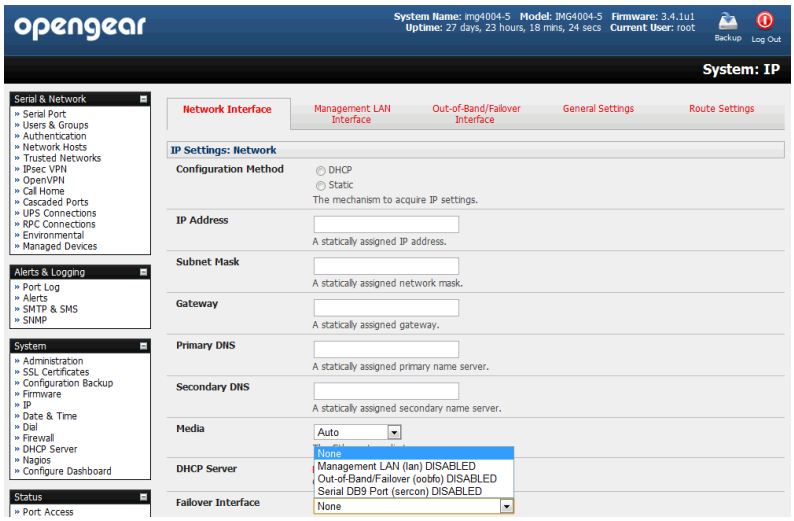


5.3.2 Failover dial-out

The ACM7000, ACM5500, ACM5000, CM7100, IM7200 and IM4200 family of advanced *console servers* can be configured so a dial-out PPP connection is automatically set up in the event of a disruption in the principal management network.

Note: Only SSH access is enabled on the failover connection. However in firmware versions later than 3.0.2 HTTPS access is also enabled. So the administrator can then SSH (or HTTPS) connect to the console server and fix the problem

- When configuring the principal network connection in **System: IP** specify the **Failover Interface** that will be used when a fault has been detected with Network / Network1 (eth0). This can be either **Internal Modem** or the **Dial Serial DB9** (if you are using an external modem on the Console port) or **USB Modem** (if you are using a plug-on USB modem on an ACM7000, ACM5500 or ACM5000)



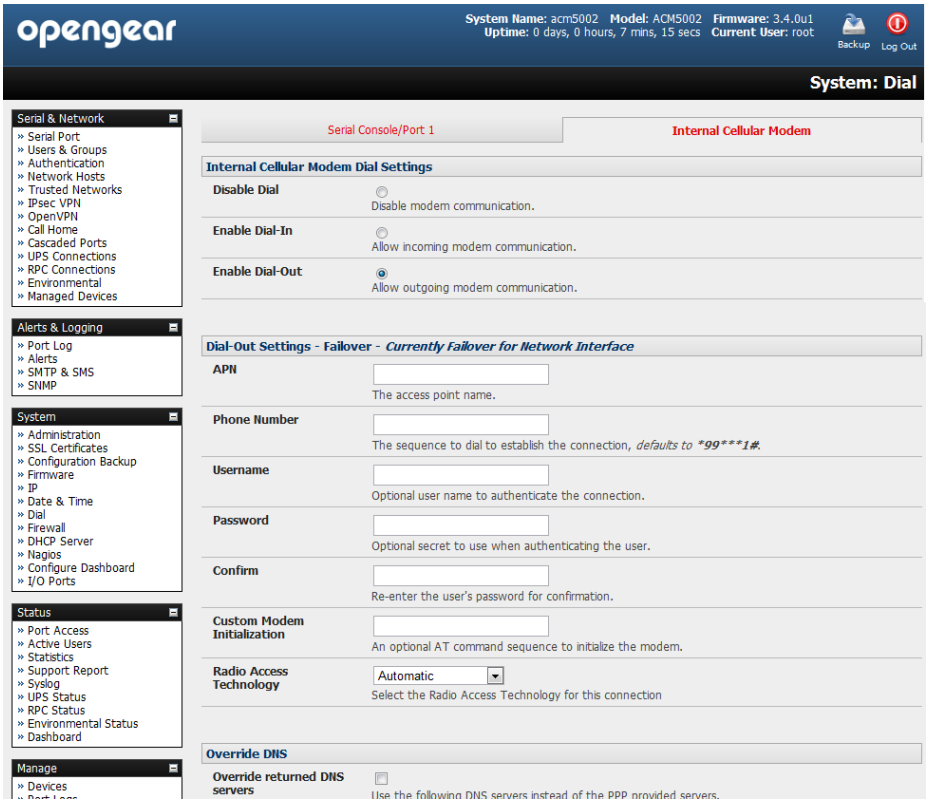
- Specify the **Probe Addresses** of two sites (the **Primary** and **Secondary**) that the IM *console server* is to *ping* to determine if Network / Network1 is still operational
- Select the **System: Dial** menu option and the port to be configured (**Serial DB9 Port** or **PC Card** or **Internal Modem Port**)
- Select the **Baud Rate** and **Flow Control** that will communicate with the modem

Note You can further configure the console/modem port (*e.g.* to include *modem init* strings) by editing */etc/mgetty.config* files as described in the Chapter 13 - Advanced.

- Check the **Enable Dial-Out Access** box and enter the access details for the remote PPP server to be called

Override DNS is available for PPP Devices such as modems. Override DNS allows the use of alternate DNS servers from those provided by your ISP. For example, an alternative DNS may be required for OpenDNS used for content filtering.

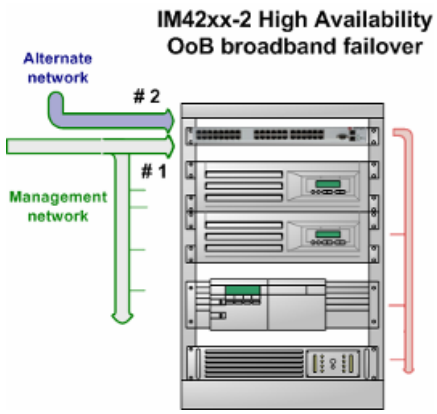
- To enable **Override DNS**, check the Override returned DNS Servers box. Enter the IP of the DNS servers into the spaces provided.



Note: By default, the advanced *console server* supports automatic failure-recovery back to the original state prior to failover (V3.1.0 firmware and later). The advanced *console server* continually pings probe addresses whilst in original and failover states. The original state will automatically be set as a priority and reestablished following three successful pings of the probe addresses during failover. The failover state will be removed once the original state has been re-established.

5.4 OOB Broadband Ethernet Access

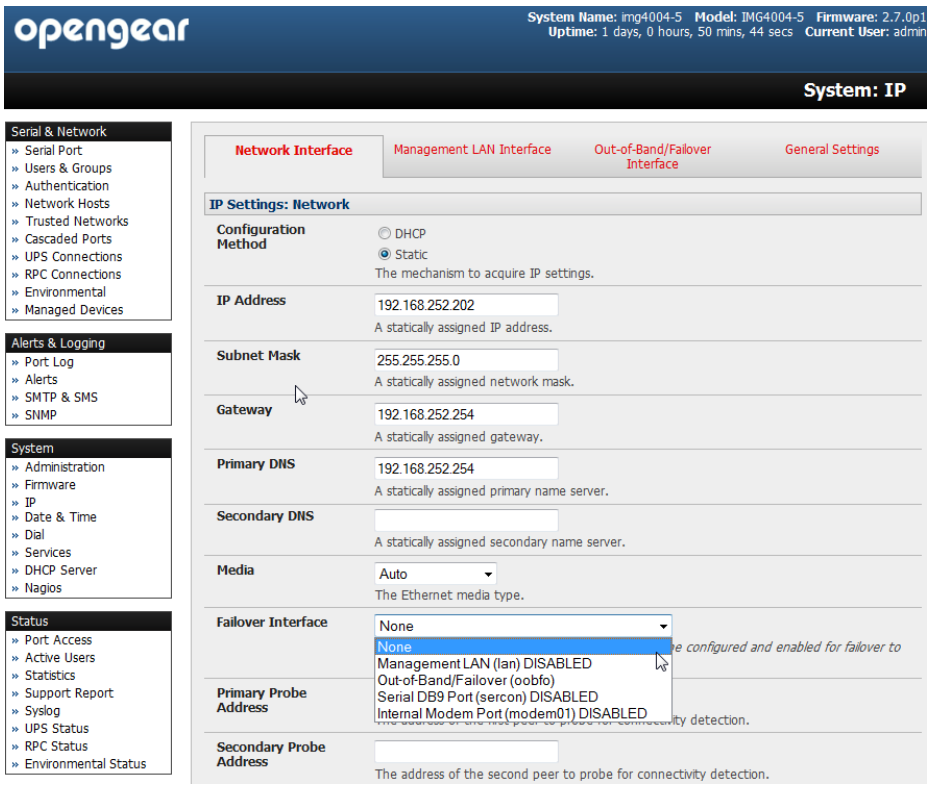
The ACM7000, ACM5500, ACM5000, CM7100, IM7200 and IM4200 family of advanced *console servers* have a second Ethernet port (*LAN2* on the ACM5004-2, ACM5508-2-I/M and ACM5504-3-P, *NET2* on the CM7100 and ACM7000, *Network 2* on the IM4200-2 or ETH-1 on the IM4216-34 and ACM5504-5-G(-W)-I) that can be configured for alternate and OOB (out-of-band) broadband access. With two active broadband access paths to these advanced *console servers*, in the event you are unable to access through the primary management network (*LAN1, Network or Network1*) you can still access it through the alternate broadband path



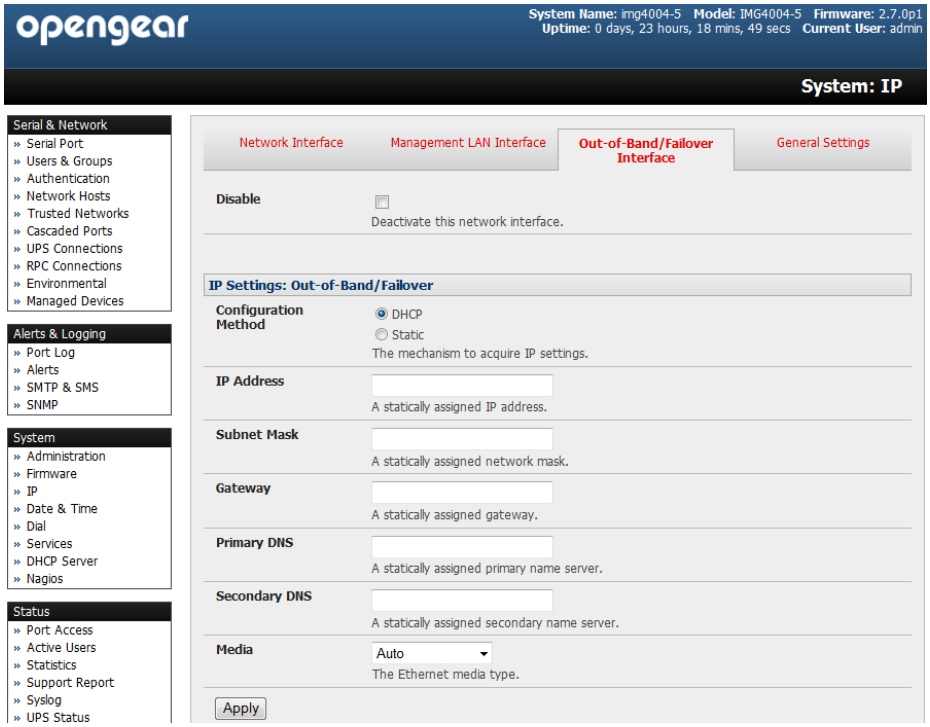
- On the **System: IP** menu select **Management LAN Interface** (ACM5004-2, CM7100, IM7200 and IM4200) and configure the **IP Address, Subnet Mask**, **Gateway** and **DNS** with the access settings that relate to the alternate link
- Ensure when configuring the principal **Network Interface** connection, the **Failover Interface** is set to *None*

5.5 Broadband Ethernet Failover

The second Ethernet port on the ACM7000, ACM5500, ACM5000, CM7100, IM7200 and IM4200 family of advanced *console servers* can also be configured for failover to ensure transparent high availability.



- When configuring the principal network connection, specify **Management LAN Interface** as the **Failover Interface** to be used when a fault has been detected with **Network Interface**
- Specify the **Probe Addresses** of two sites (the **Primary** and **Secondary**) that the advanced *console server* is to *ping* to determine if **Network Interface** is still operational
- Then on the **Management LAN Interface** (CM7100, IM4200, IM7200 or ACM5004-2) configure the **IP Address**, **Subnet Mask** and **Gateway** the same as you used for **Network Interface**



In this mode, **Management LAN Interface** is available as the transparent back-up port to **Network Interface** for accessing the management network. **Management LAN Interface** will automatically and transparently take over the work of **Network Interface**, in the event **Network Interface** becomes unavailable for any reason.

Note: Only SSH access is enabled on the failover connection. However in firmware versions later than 3.0.2 HTTPS access is also enabled. So the administrator can then SSH (or HTTPS) connect to the console server and fix the problem

By default, the advanced *console server* supports automatic failure-recovery back to the original state prior to failover (V3.1.0 firmware and later). The *advanced* console server continually pings probe addresses whilst in original and failover states. The original state will automatically be set as a priority and reestablished following three successful pings of the probe addresses during failover. The failover state will be removed once the original state has been re-established.

Note: For firmware pre V3.1.0 the advanced *console server* does not support automatic failure-recovery back to the original state prior to the failover. So to restore networking to a recovered state the following command then needs to be run:

rm -f /var/run/-failed-over && config -r ipconfig*

If required, you can run a custom bash script when the device fails over. It is possible to use this script to implement automatic failure recovery, depending on your network setup. The script to create is:

/etc/config/scripts/interface-failover-alert

5.6 Cellular Modem Connection

The ACM7000, ACM5500, ACM5000, IM7200 and IM4200 family of advanced *console servers* support internal cellular modems. These modems first need to be installed (as described below in 5.6.1, 5.6.2 or 5.6.3) and then set up to validate they can connect to the carrier network (as described below in 5.6.4 and 5.6.5). They then can be configured for operation in Always- on cellular router or OOB mode, or in Failover mode (as detailed in next section 5.7).

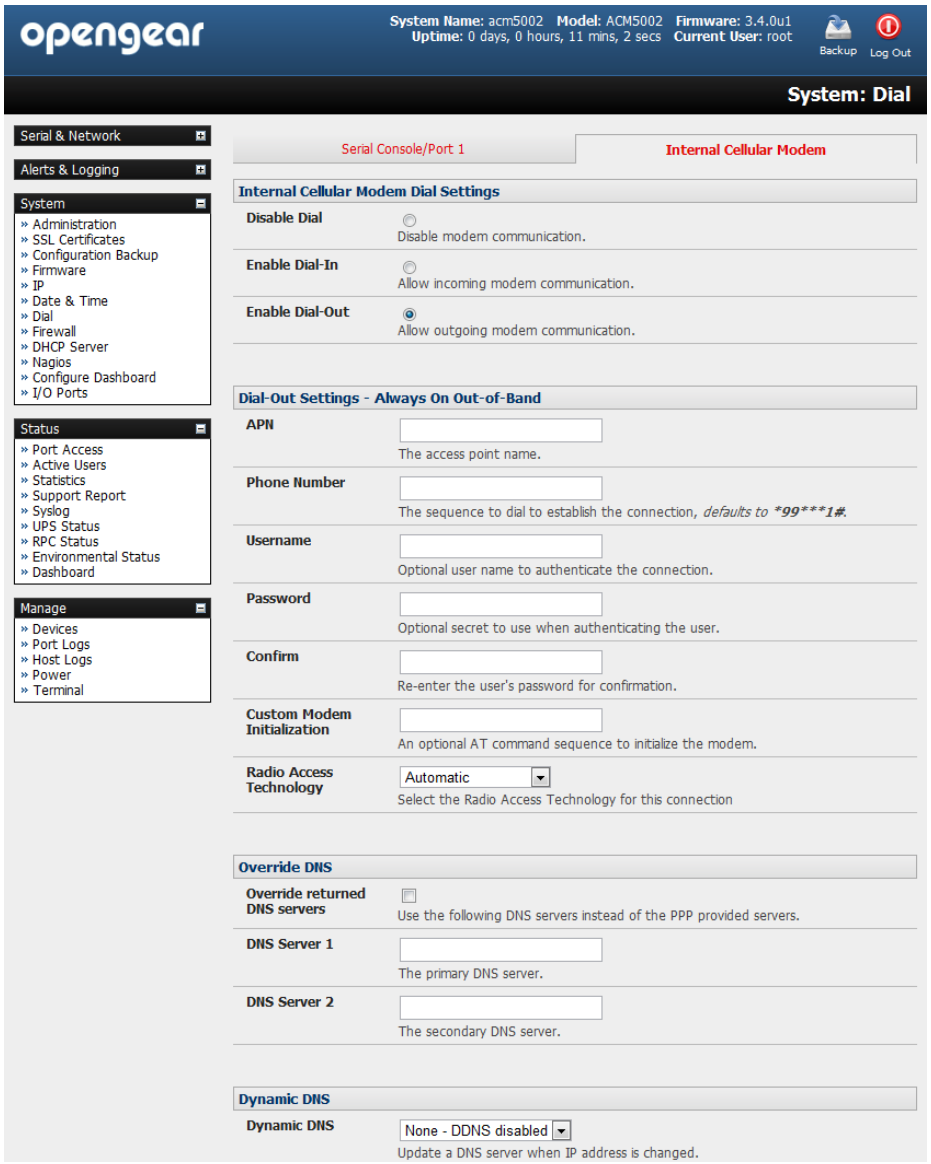
5.6.1 Connecting to a *GSM HSUPA/UMTS* carrier network

-G models have an internal GSM modem that will connect to any major GSM carrier globally.

- Before powering on the ACM5004-G(-I), ACM55044-5-G-I or IM4200-X2-G you must install the SIM card provided by your cellular carrier, and attach the external aerial

Note: The ACM5004-G(-I) and ACM55044-5-G-I each has two cellular status LEDs. The SIM LED on top of the unit should go on solid when a SIM card has been inserted and detected.

- Select **Internal Cellular Modem** panel on the **System: Dial** menu
- Check **Enable Dial-Out Settings**



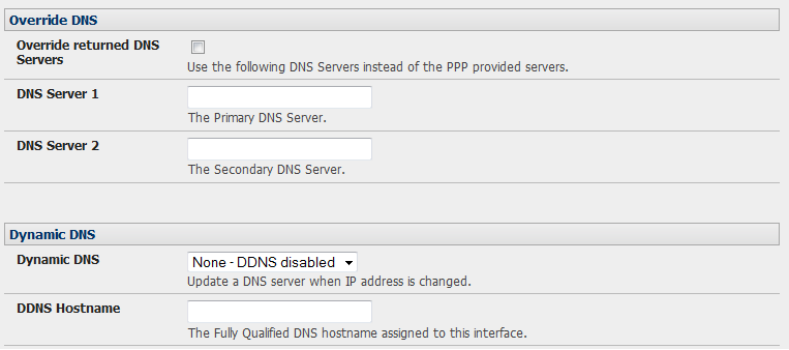
Note: Your 3G carrier may have provided you with details for configuring the connection including APN (Access Point Name), Pin Code (optional PIN code which may be required to unlock the SIM card), Phone Number (the sequence to dial to establish the connection, defaults to *99***1#), Username / Password (optional) and Dial string (optional AT commands). However you generally will only need to enter your provider's APN and leave the other fields blank

- Enter the carrier's **APN** e.g. for AT&T (USA) simply enter *i2gold*, for T-Mobile (USA) enter *epc.tmobile.com*, for InterNode (Aust) enter *internode* and for Telstra (Aust) enter *telstra.internet*

- If the SIM Card is configured with a PIN Code, you will be required to unlock the Card by entering the PIN Code. If the PIN Code is entered incorrectly three times, then the PUK Code will be required to unlock the Card.

You may also need to set Override DNS to use alternate DNS servers from those provided by your carrier.

- To enable **Override DNS**, check the Override returned DNS Servers box. Enter the IP of the DNS servers into the spaces provided.



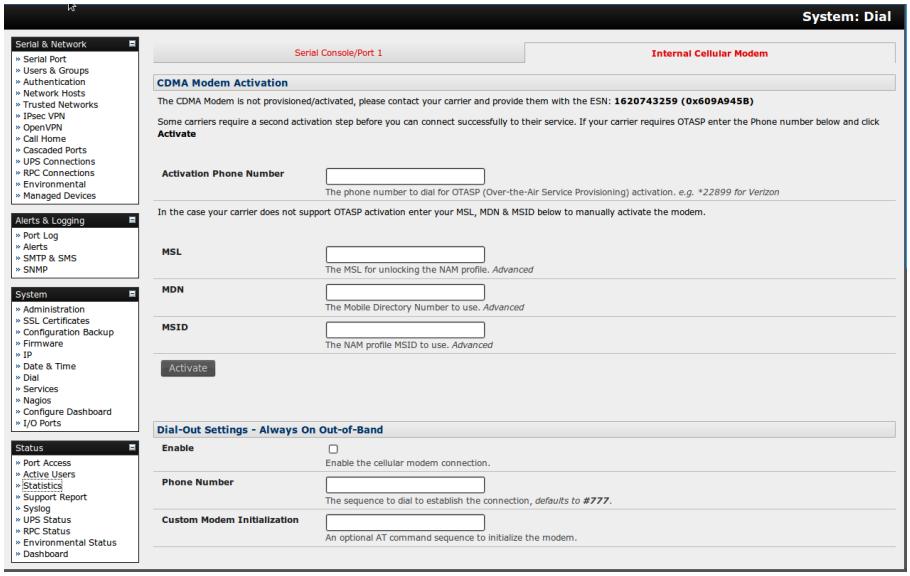
- Check **Apply** and a radio connection will be established with your cellular carrier

5.6.2 Connecting to a CDMA *EV-DO* carrier network

-GV and -GS models have an internal CDMA modem. Both will connect to the Verizon network in North America.

After creating an account with the CDMA carrier some carriers require an additional step to provision the **Internal Cellular Modem**, referred to as *Provisioning*. The ACM5004-GV (and IM4200-DAC-X2) supports:

- Over-the-Air Service Provisioning (*OTASP*) where modem specific parameters can be retrieved via a voice call to a special phone number, and
- a manual process where the phone number and other parameters can be entered manually

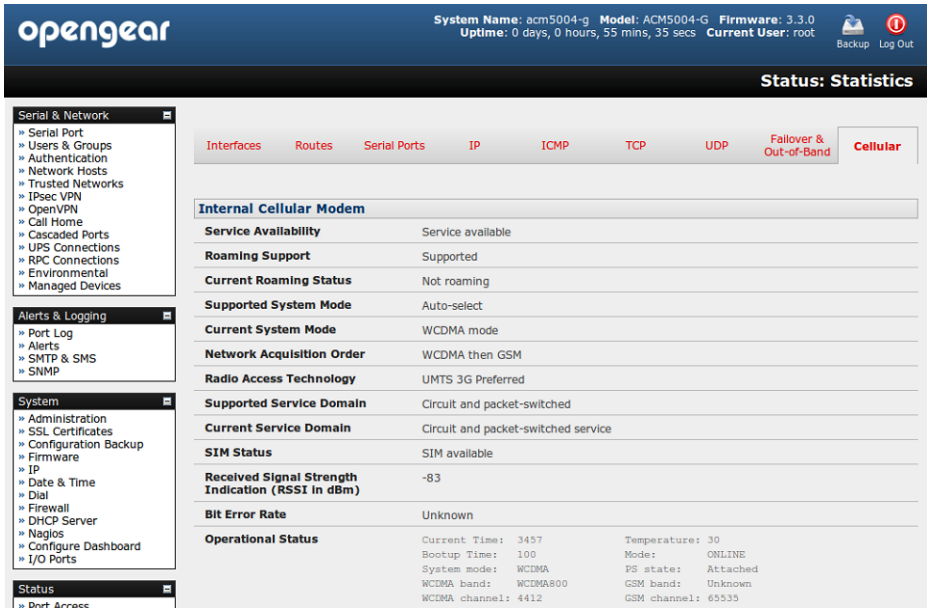


OTASP Activation:

Before this can be achieved you need both a working account and an activated device in that the Opengear's ESN (Electronic Serial Number) needs to be registered with an appropriate plan on your Carriers account

- Select **Internal Cellular Modem** panel on the **System: Dial** menu

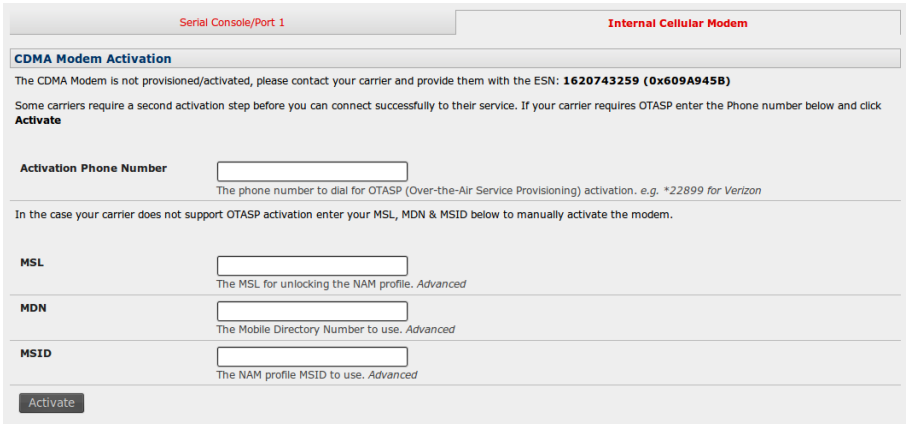
- A particular phone number will need to be dialed to complete *OTASP* e.g. Verizon uses **22899*, Telus uses **22886*
- Click **Activate** to initiate the *OTASP* call. The process is successful if no errors are displayed and you no longer see the *CDMA Modem Activation* form. (If *OTASP* is unsuccessful you can consult the System Logs for clues to what went wrong at **Status**: **Syslog**)
- When **OTASP** has completed successfully you can proceed to enabling the **Internal Cellular Modem** by entering the carriers phone number (which defaults to **#777**) and clicking **Apply**
- The **Cellular** statistics page on **Status**: **Statistics** will display the current state of the modem



- **OTASP** success will result in a valid phone number being placed in the **NAM Profile Account MDN** field

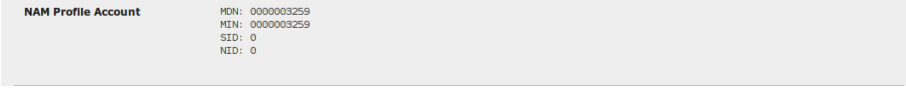
Manual Activation:
Some carriers may not support **OTASP** in which case it may be necessary to manually provision the modem.

- Select **Internal Cellular Modem** panel on the **System: Dial** menu



- Enter the **MSL**, **MDN** and **MSID** values. These values are specific to your carrier and for manual activation you will have to investigate what values your carrier uses in each field. For example **Verizon** have been known to use an **MSL** of **000000** and the phone number assigned to the Opengear device as both the **MDN** and **MSID** with no spaces or hyphens e.g. “5551231234” for “555-123-1234”

- Click **Activate**. If no errors occur you will see the new values entered into the **NAM Profile** at the **Cellular** page on **Status: Statistics**



- Navigate to the **Internal Cellular Modem** tab on **System: Dial**. To connect to your carriers 3G network enter the appropriate phone number (usually **#777**) and a **Username** and **Password** if directed to by your account/plan documentation
- Select **Enable** and then click **Apply** to initiate the **Always On Out-of-Band** connection

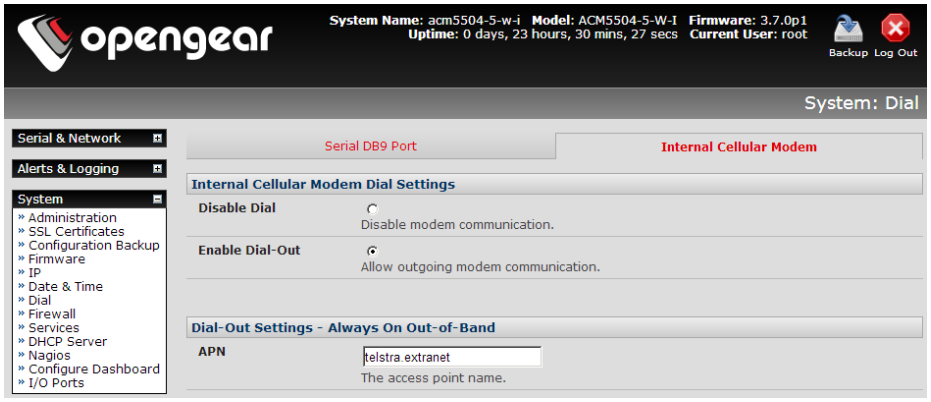
5.6.3 Connecting to a *4G LTE* carrier network

-LV, -LA and -LR models have an internal modem that will connect to any major 4G LTE carrier globally.

- Before powering on, you must install the SIM card provided by your cellular carrier, and attach the external aerial

Note: The ACM5504-5-LA/R/V-I each has two cellular status LEDs. The SIM LED on top of the unit should go on solid when a SIM card has been inserted and detected.

- Select **Internal Cellular Modem** panel on the **System: Dial** menu
- Check **Enable Dial-Out Settings**

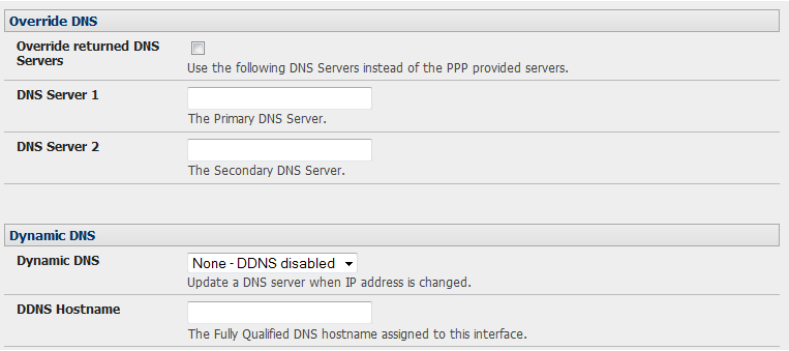


Note: Your 4G LTE carrier may have provided you with details for configuring the connection including APN (Access Point Name), Pin Code (optional PIN code which may be required to unlock the SIM card), Username / Password etc. However you generally will only need to enter your provider's APN and the other fields can remain blank.

- Enter the carrier's **APN**
- If the SIM Card is configured with a PIN Code, you will be required to unlock the Card by entering the PIN Code.

You may also need to set Override DNS to use alternate DNS servers from those provided by your carrier.

- To enable **Override DNS**, check the Override returned DNS Servers box. Enter the IP of the DNS servers into the spaces provided.

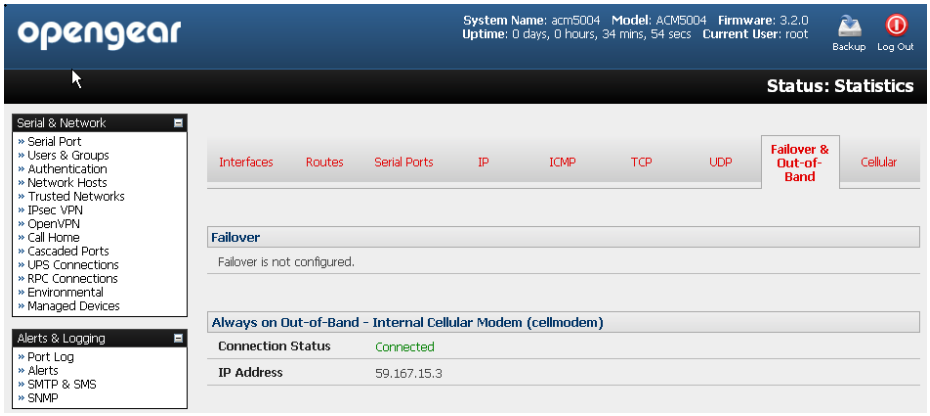


- Check **Apply** and a radio connection will be established with your cellular carrier

5.6.4 Verifying the cellular connection

Out-of-band access is enabled by default so the cellular modem connection should now be on.

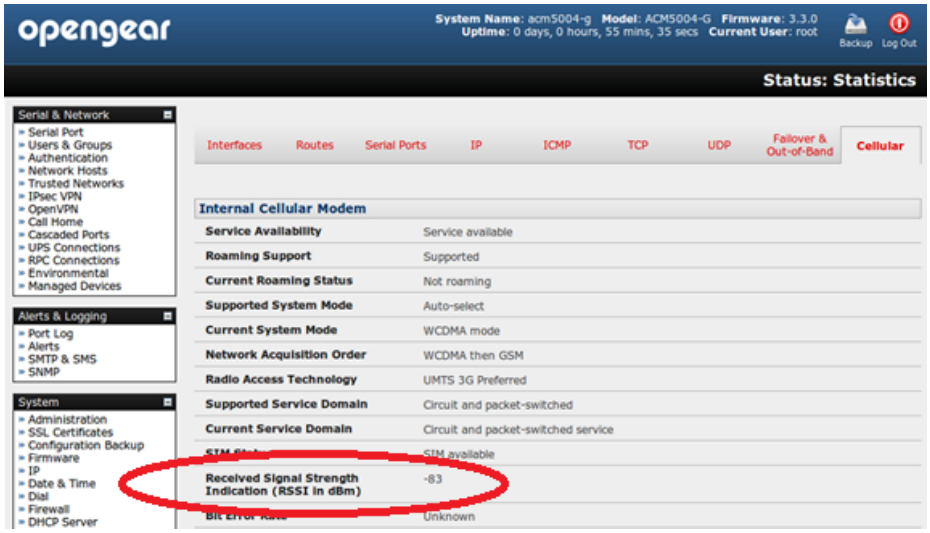
- You can verify the connection status from the **Status: Statistics**
  - Select the **Cellular** tab and in *Service Availability* verify *Mode* is set to *Online*
  - Select **Failover& Out-of-Band** and the Connection Status reads *Connected*
  - You can check your allocated *IP address*



- You can measure the received signal strength from the **Cellular Statistics** page on the **Status: Statistics** screen. This will display the current state of the cellular modem including the Received Signal Strength Indicator (**RSSI**)

Note: Received Signal Strength Indicator (**RSSI**) is a measurement of the Radio Frequency (RF) power present in a received radio signal at the mobile device. It is generally expressed in *dBm* and the best throughput comes from placing the device in an area with the highest RSSI.

-100 dbm or less = Unacceptable coverage
-99 dbm to –90 dbm = Weak Coverage
-89 dbm to – 70 dbm = Medium to High Coverage
-69 dbm or greater = Strong Coverage

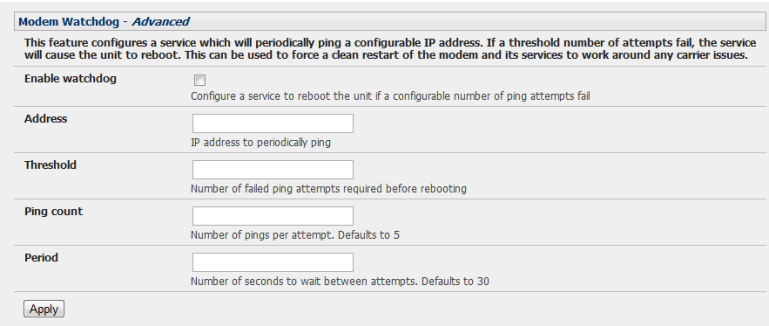


- With the cellular modem connection on you can also see the connection status from the LEDs on top of unit

Note: The ACM5004-G(-I), ACM5504-5-G(-W)-I and ACM5504-5-LA/R/V-I each has two cellular status LEDs. The WWAN LED is OFF when in reset mode or not powered. When powered it will go ON and while searching for service it will flash off briefly every 5sec.

5.6.5 Cellular modem watchdog

When you select **Enable Dial-Out** on the **System: Dial** menu you will be given the option to configure a cellar modem watchdog service (with firmware V3.5.2u13 and later). This service will periodically ping a configurable IP address. If a threshold number of consecutive attempts fail, the service will cause the unit to reboot. This can be used to force a clean restart of the modem and its services to work around any carrier issues.

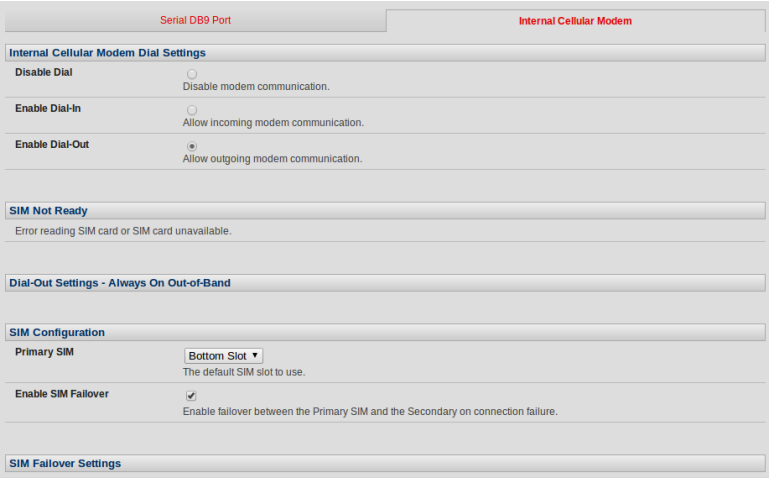


5.6.6 Dual SIM failover

Some console server models (currently ACM5500-Gx and ACM5500–Lx families) allow you to insert two SIM cards (so you can connect selectively to two carrier networks). The dual SIM failover feature allows the cell modem to selectively failover to the secondary SIM when communications over the primary SIM fails.

To configure dual SIM failover, you need to:

- Choose which of the SIMs is to be the *Primary*, and the other SIM will be the secondary/failover. Select **Internal Cellular Modem** panel on the **System: Dial** menu nominate which slot (**Top** or **Bottom**) contains the *Primary*
- Check **Enable SIM Failover**

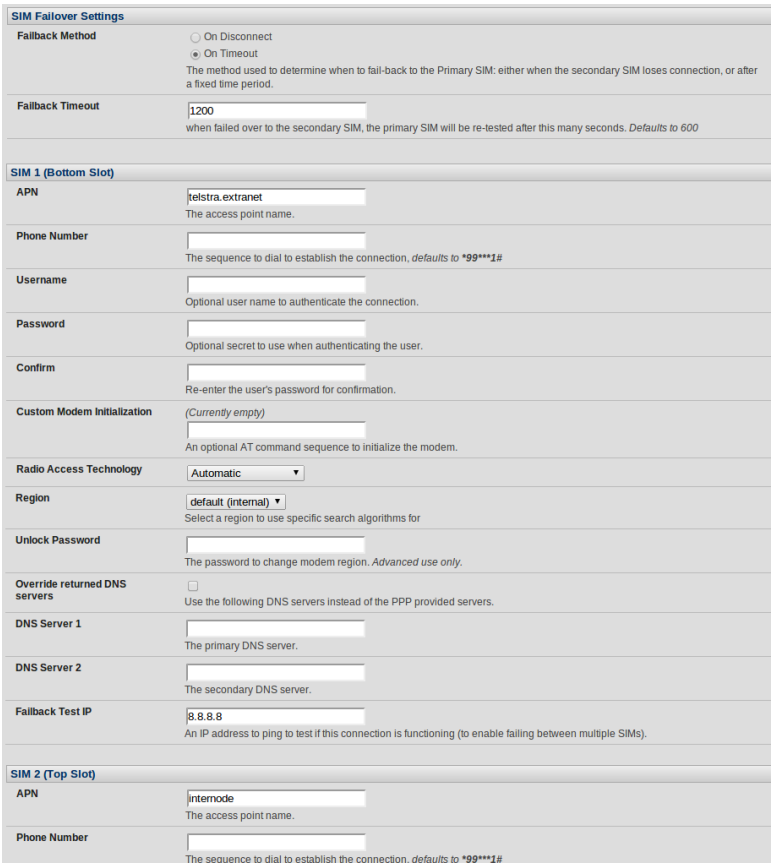


- Specify how the device will *Failback* from the failover SIM to the Primary SIM. There are two options
  - The **'On Disconnect'** failback option will failback to the Primary SIM only after the connection on the failover SIM has failed its ping test
  - The '**On Timeout'** failback option will failback to the Primary SIM after the connection on the failover SIM has been up for the timeout period. The timeout period is either the default value of 600 seconds if the **Failback Timeout** field is left empty or the number of seconds you have specified in the **Failback Timeout** field

- Next you will need to configure each SIM connection with as much information (eg. APN etc) such that it will make a successful connection assuming sufficient signal strength from the cell service provider

- Enter a Failback Test IP address for each SIM. This IP address is used to ping test the status of the cell modem connection and to determine if SIM failover or failback is to take place.

- Configuring DDNS and the Modem Watchdog are optional. DDNS, when configured, will be applied to the cell modem dial out connection regardless of which SIM is currently in use. Dual SIM failover is for *dial out* connections only



Note: Dual SIM failover still applies to the cell modem interface when the cell modem itself is used as the console server's failover interface. Be aware that when the console server is failing over to the cell modem interface and the primary SIM fails, total time to fail over to the cell modem and then for the cell modem to failover to its secondary SIM can take several minutes - be patient.

5.7 Cellular Operation

When set up as a *console server* the 3G cellular modem can be set up to connect to the carrier in either:

- *Cellular router mode*. In this case the dial-out connection to the carrier cellular network is always on, and IP traffic is routed between the cellular connected network and the console server's local network ports. This is the default mode of operation for ACM5000-G and ACM5500-L and ACM5500-G models.
- *OOB mode*. As above in this mode the dial-out connection to the carrier cellular network is always on - awaiting any incoming access (from a remote site wanting to access to the console server or attached serial consoles/network hosts)
- *Failover mode*. In this case a dial-out cellular connection is only established in event of a *ping* failure
- *Circuit Switched Data (CSD) mode*. In this dial-in mode the cellular modem can receive incoming calls from remote modems who dial a special Data Terminating number. This is a 3G mode only.

5.7.1 OOB access set up

In this mode the dial-out connection to the carrier cellular network is always on, awaiting any incoming traffic. By default the only traffic enabled are incoming SSH access to the *console server* and its serial ports, and incoming HTTPS access to the *console server*. There is a low level of keep alive and management traffic going over the cellular network, however

generally the status reports and alerts etc from the site can be carried over the main network.

This mode is used typically for out of band access to remote sites, and as above to be directly accessed the appliance needs to have a Public IP address (and it must not have SSH access firewalled). This OOB mode is the default for IM7200 and IM4200 appliances with internal cellular modems. Out-of-band access is enabled by default and the cellular modem connection is always on.

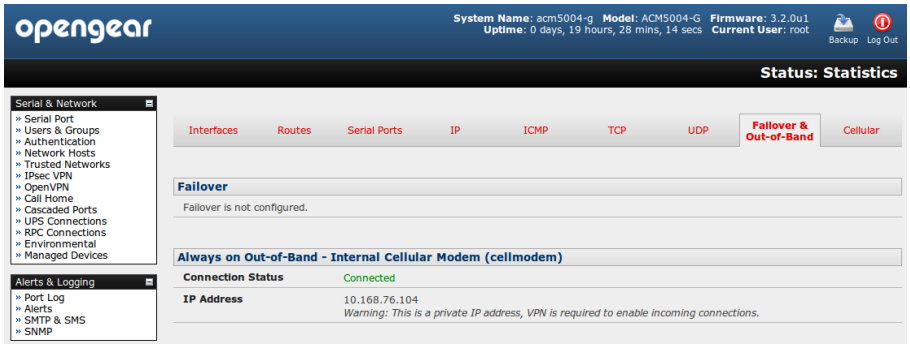
To be directly accessed the *console server* needs to have a Public IP address and it must not have SSH access firewalled.

Almost all carriers offer corporate mobile data service/plans with a Public (static or dynamic) IP address. These plans often have a service fee attached.

- If you have such a static Public IP address plan you can also now try accessing the *console server* using the Public IP Address provided by the carrier. However by default only HTTPS and SSH access is enabled on the OOB connection. So you can browse to the *console server*, but you cannot *ping* it
- If you have a dynamic Public IP address plan then a DDNS service will need to be configured to enable the remote administrator to initiate incoming access. Once this is done you can then also try accessing the *console server* using the allocated domain name

By default most providers offer a consumer grade service which provides dynamic Private IP address assignments to 3G devices. This IP address is not visible across the Internet but generally it is adequate for home and general business use.

- With such a plan the **Failover& Out-of-Band** tab on the **Status: Statistics** shows will identify that your carrier has allocated you a Private *IP Address* (i.e. in the range 10.0.0.0 – 10.255.255.255, 172.16.0.0 – 172.31.255.255 or 192.168.0.0 – 192.168.255.255)



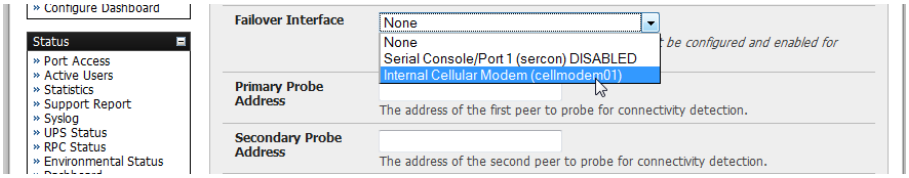
- For inbound OOB connection with such a plan you will need to use Call Home with a Lighthouse/VCMS/CMS6110 or set up a VPN

In out of band access mode the internal cellular modem will continually stay connected. The alternative is to set up Failover mode on the *console server* as detailed in the next section.

5.7.2 Cellular failover setup

In this mode a dial-out cellular connection is only established in event of disruption to the main network. The cellular connection normally remains idle - in a low power state - and is only activated in event of a ping failure. This standby mode can suit remote sites with expensive power or very high cellular traffic costs.
In this mode, the appliance continually *pings* nominated probe addresses over the main network connection and in the event of *ping* failure it dials out and sets up a dial-out *ppp* over the cellular modem and access is switched transparently to this network connection. Then when the main network connection is restored, access is switched back.
Once you have configured carrier connection, the cellular modem can be configured for failover.

This will tell the cellular connection to remain idle in a low power state. If the primary and secondary probe addresses are not available it will bring up the cellular connection and connect back to the cellular carrier.



- Navigate back to the **Network Interface** on the **System:IP** menu specify **Internal Cellular modem (cell modem 01**) as the **Failover Interface** to be used when a fault has been detected
- Specify the **Probe Addresses** of two sites (the **Primary** and **Secondary**) that the *console server* is to *ping* to determine if the principal network is still operational
- In event of a failure of the principal network the 3G network connection is activated as the access path to the console server (and Managed Devices). Only HTTPS and SSH access is enabled on the failover connection (which should enable the administrator to connect and fix the problem)

Note: By default, the advanced *console server* supports automatic failure-recovery back to the original state prior to failover (V3.1.0 firmware and later). The advanced *console server* continually pings probe addresses whilst in original and failover states. The original state will automatically be set as a priority and reestablished following three successful pings of the probe addresses during failover. The failover state will be removed once the original state has been re-established.

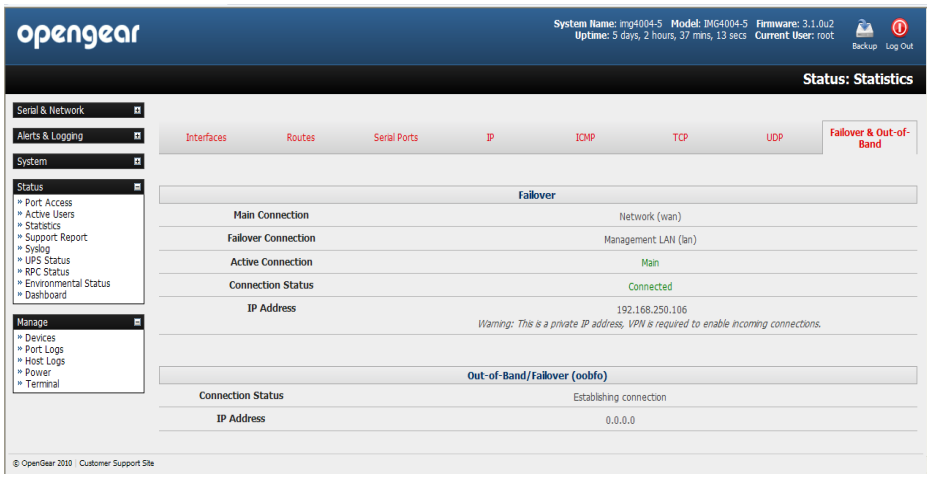
For earlier firmware that does not support automatic failure-recovery, to restore networking to a recovered state the following command then needs to be run:

rm -f /var/run/-failed-over && config -r ipconfig*

If required, you can run a custom bash script when the device fails over. It is possible to use this script to implement automatic failure recovery, depending on your network setup. The script to create is:

/etc/config/scripts/interface-failover-alert

- You can check the connection status by selecting the *Cellular* panel on the **Status: Statistics** menu



- The *Operational Status* will change as the cellular modem finds a channel and connects to the network
- The **Failover & Out-of-Band** screen will display information relating to a configured Failover/OOB interface and the status of that connection. The IP Address of the Failover / OOB interface will be presented in the **Failover & Out-of-Band** screen once the Failover/OOB interface has been triggered

5.7.3 Cellular routing

Once you have configured carrier connection, the cellular modem can be configured to route traffic through the *console server*. This requires setting up *forwarding* and *masquerading* - as detailed in Chapter 5.8.

5.7.4 Cellular CSD dial-in setup

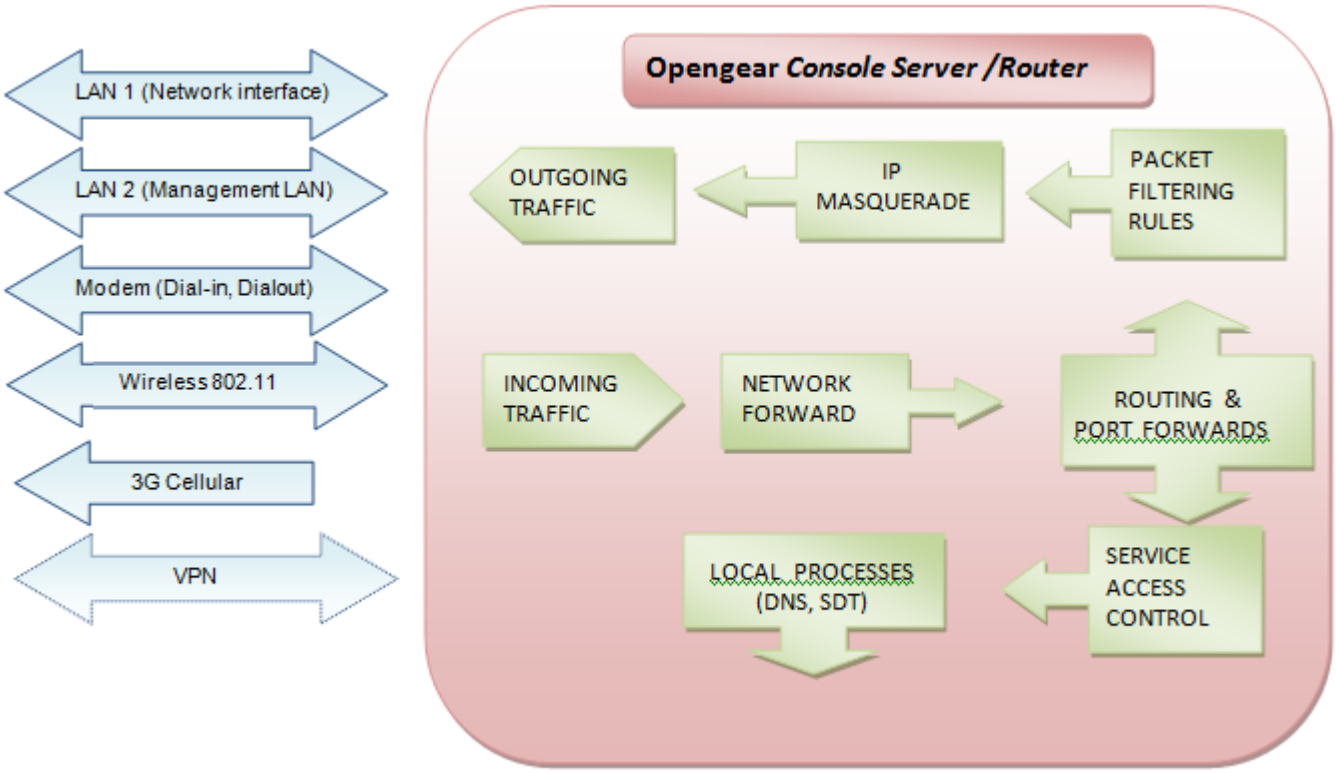
Once you have configured carrier connection, the cellular modem can be configured to receive Circuit Switched Data (CSD) calls.

Note: CSD is a legacy form of data transmission developed for the TDMA based mobile phone systems like GSM. CSD uses a single radio time slot to deliver 9.6kb/s data transmission to the GSM Network and Switching Subsystem where it could be connected through the equivalent of a normal modem to the Public Switched Telephone Network (PSTN) allowing direct calls to any dial-up service. CSD is provided selectively by carriers and it is important you receive a *Data Terminating number* as part of the mobile service your carrier provides. This is the number which external modems will call to access the console server

- Select the **Cellular Modem** panel on the **System: Dial** menu
- Check **Enable Dial-In** and configure the **Dial-In Settings**

5.8 Firewall & Forwarding

Opengear *console servers* with Version 3.3 firmware (and beyond) have basic routing, NAT (Network Address Translation), packet filtering and port forwarding support on all network interfaces.



This enables the console server to function as an Internet or external network gateway, via cellular connections or via other Ethernet networks on two Ethernet port models:

- **Network Forwarding** allows the network packets on one network interface (i.e. LAN1 / eth0) to be forwarded to another network interface (i.e. LAN2/eth1 or dial-out/cellular). So locally networked devices can IP connect through the console server to devices on remote networks
- **IP Masquerading** is used to allow all the devices on your local private network to hide behind and share the one public IP address when connecting to a public network. This type of translation is only used for connections originating within the private network destined for the outside public network, and each outbound connection is maintained by using a different source IP port number
- When using IP Masquerading, devices on the external network cannot initiate connections to devices on the internal network. **Port Forwards** allows external users to connect to a specific port on the external interface of the *console server* and be redirected to a specified internal address for a device on the internal network
- With **Firewall Rules**, packet filtering inspects each packet passing through the firewall and accepts or rejects it based on user-defined rules
- Then **Service Access Rules** can be set for connecting to the console server/router itself

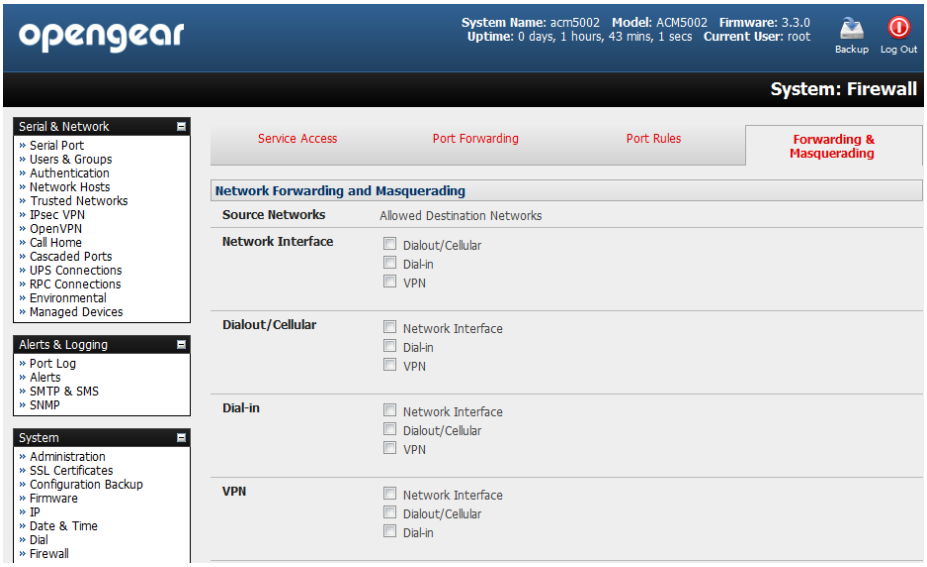
5.8.1 Configuring network forwarding and IP masquerading

To use a *console server* as an Internet or external network gateway requires establishing an external network connection (e.g. for the ACM5004-G setting up the 3G cellular link as detailed in Chapter 5) and then setting up *forwarding* and *masquerading.*

Note: Network *forwarding* allows the network packets on one network interface (i.e. LAN1 / eth0) to be forwarded to another network interface (i.e. LAN2/eth1 or dial-out/cellular). So locally networked devices can IP connect through the *console server* to devices on a remote network. IP *masquerading* is used to allow all the devices on your local private network to hide behind and share the one public IP address when connecting to a public network. This type of translation is only used for connections originating within the private network destined for the outside public network, and each outbound connection is maintained by using a different source IP port number.

By default, all *console server* models are configured so that they will not route traffic between networks. To use the *console server* as an Internet or external network gateway, *forwarding* must be enabled so that traffic can be routed from the internal network to the Internet/external network:

- Navigate to the **System: Firewall** page, and then click on the **Forwarding &Masquerading** tab



- Find the **Source Network** to be routed, and then tick the relevant **Destination Network** to enable Forwarding

For example to configure a single Ethernet device such as an ACM5004-G as a cellular router:

- The **Source Network** would the **Network Interface** and the **Destination Network** would be **Dialout/Cellular**)

IP Masquerading is generally required if the *console server* will be routing to the Internet, or if the external network being routed to does not have routing information about the internal network behind the *console server*.

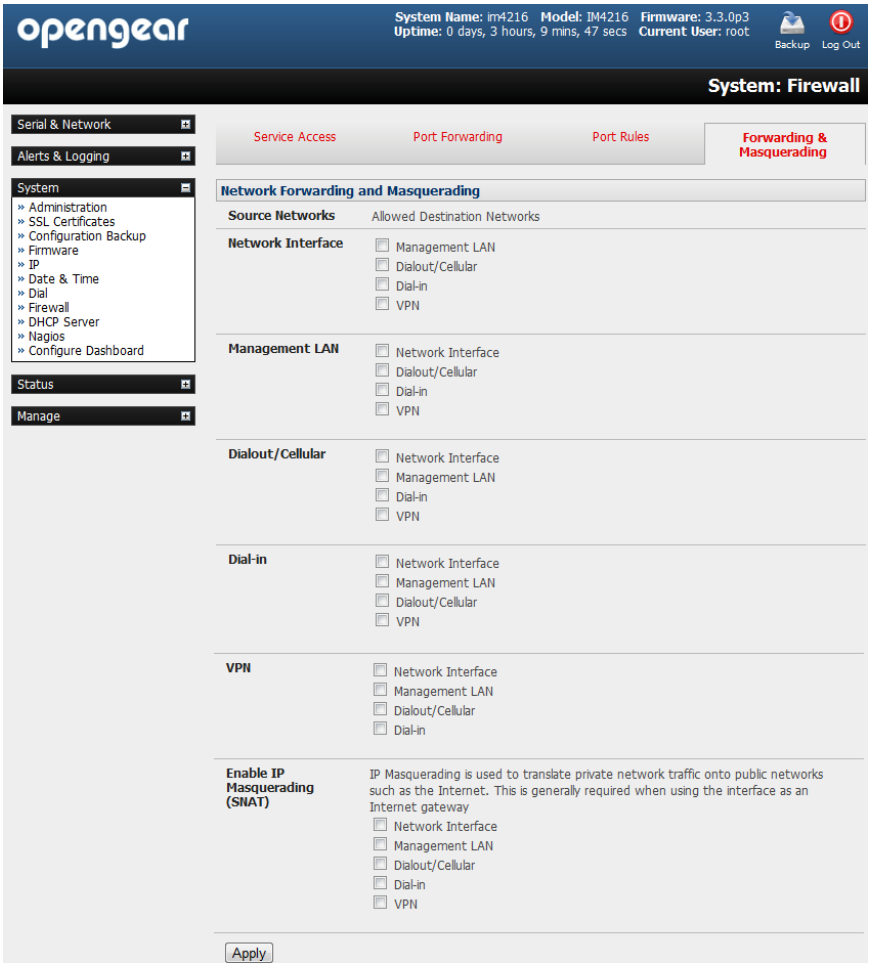
IP Masquerading performs Source Network Address Translation (SNAT) on outgoing packets, to make them appear like they've come from the *console server* (rather than devices on the internal network). When response packets come back devices on the external network, the *console server* will translate the packet address back to the internal IP, so that it is routed correctly. This allows the *console server* to provide full outgoing connectivity for internal devices using a single IP Address on the external network.

By default IP Masquerading is disabled for all networks. To enable masquerading:

- Select **Forwarding & Masquerading** panel on the **System: Firewall** menu

- Check **Enable IP Masquerading (SNAT)** on the network interfaces where masquerading is be enabled

Generally this masquerading would be applied to any interface that is connecting with a public network such as the Internet (e.g. for the ACM5004-G the IP masquerading would be enabled on **Dialout/Cellular**)



5.8.2 Configuring client devices

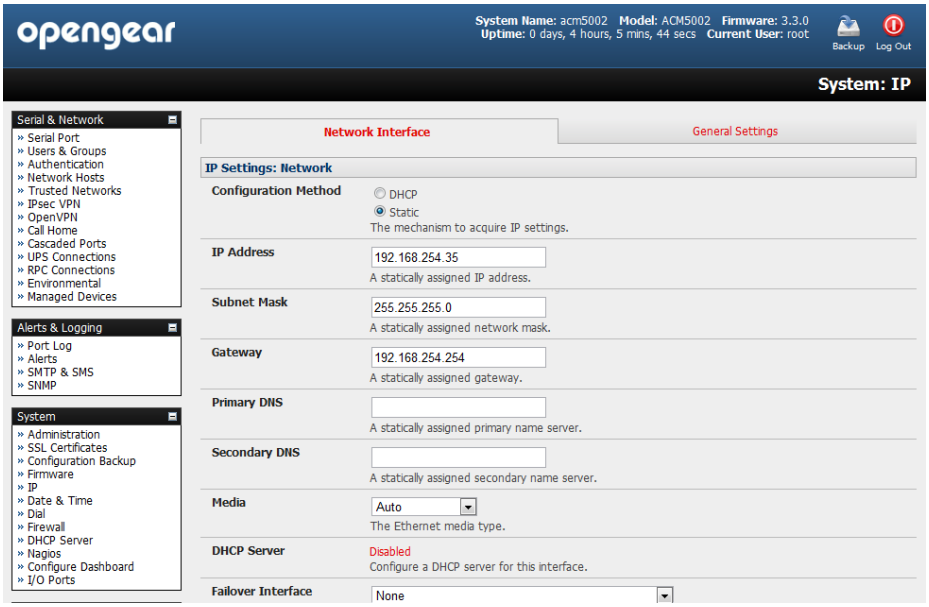
Client devices on the local network must be configured with *Gateway* and *DNS* settings. This can be done statically on each device, or using DHCP (on IM and ACM models).

Manual Configuration:

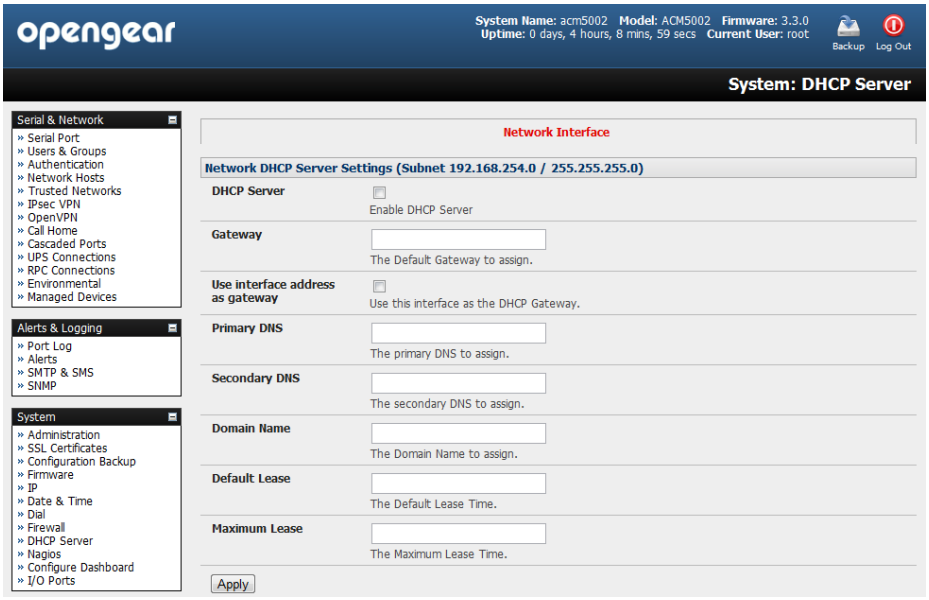
Manually set a static gateway address (being the address of the *console server*) and set the DNS server address to be the same as used on the external network i.e. if the *console server* is acting as an internet gateway or a cellular router, then use the ISP provided DNS server address.

DHCP Configuration (IM/ACM families only):

- Navigate to the **System:IP** page
- Click the tab of the interface connected to the internal network. To use DHCP, a static address must be set; check that the static IP and subnet mask fields are set.



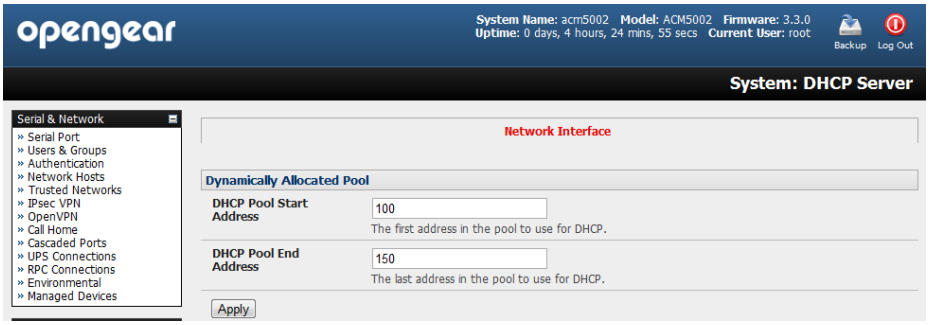
- Click on the **Disabled** link next to **DHCP Server** which will bring up the System: DHCP Server page



- Check **Enable DHCP Server**
- To configure the DHCP server, tick the **Use interface address as gateway check box**
- Set the DNS server address(es) to be the same as used on the external network i.e. if the *console server* is acting as an internet gateway or a cellular router, then use the ISP provided DNS server address
- Enter the **Default Lease** time and **Maximum Lease** time in seconds. The lease time is the time that a dynamically assigned IP address is valid before the client must request it again
- Click **Apply**

The DHCP server will sequentially issue IP addresses from a specified address pool(s):

- Click **Add** in the **Dynamic Address Allocation Pools** field
- Enter the **DHCP Pool Start Address** and **End Address** and click **Apply**



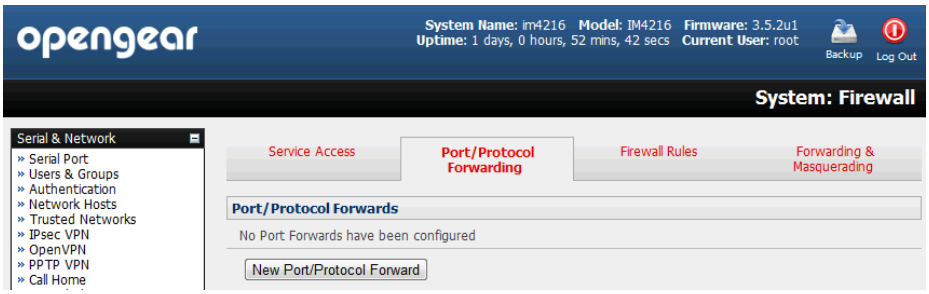
The DHCP server also supports pre-assigning IP addresses to be allocated only to specific MAC addresses and reserving IP addresses to be used by connected hosts with fixed IP addresses. To reserve an IP addresses for a particular host.

Once applied, devices on the internal network will be able to access resources on the external network.

5.8.3 Port / Protocol forwarding

When using IP Masquerading, devices on the external network cannot initiate connections to devices on the internal network.

To work around this, *Port Forwards* can be set up to allow external users to connect to a specific port, or range of ports on the external interface of the *console server/cellular router* , and have the *console server/cellular router* redirect the data to a specified internal address and port range.



To setup a port/protocol forward:

- Navigate to the **System: Firewall** page, and click on the **Port Forwarding** tab
- Click **Add New Port Forward**
- Fill in the following fields:

Name: Name for the port forward. This should describe the target and the service that the port forward is used to access

Input Interface: This allows the user to only forward the port from a specific interface. In most cases, this should be left as "Any"

Source Address/*Address Range*: This allows the user to restrict access to a port forward to a specific source IP address or IP address range of the data. This may be left blank. IP address ranges use the format *ip/netmask* (where netmask is in bits 1-32)

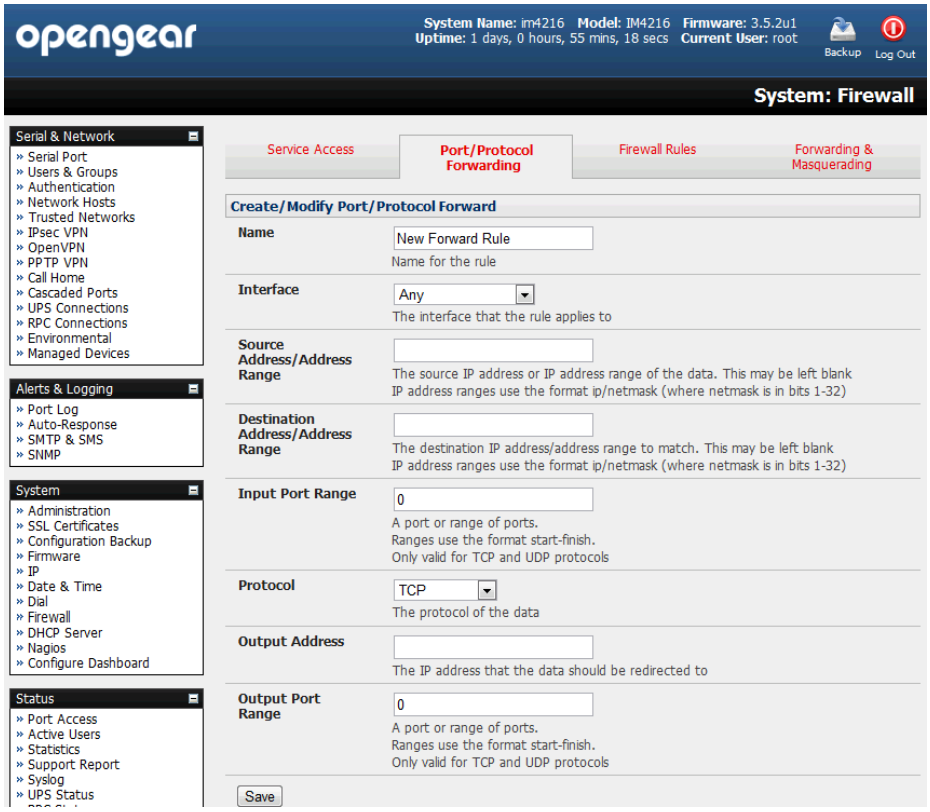
Destination Address/*Address Range*: The destination IP address/address range to match. This may be left blank IP address ranges use the format ip/netmask (where netmask is in bits 1-32)

Input Port Range: The range of ports to forward to the destination IP. These will be the port(s) specified when accessing the port forward. These ports need not be the same as the output port range.

Protocol: The protocol of the data being forwarded. The options are *TCP* or *UDP* or *"TCP and UDP"* or *ICMP* or *ESP* or *GRE* or *Any*

Output Address: The target of the port forward. This is an address on the internal network where packets sent to the Input Interface on the input port range are sent.

Output Port Range: The port or range of ports that the packets will be redirected to on the Output Address. Ranges use the format start-finish. Only valid for TCP and UDP protocols



For example, to forward port 8443 to an internal HTTPS server on 192.168.10.2, the following settings would be used:

Input Interface: Any

Input Port Range: 8443

Protocol: TCP

Output Address: 192.168.10.2

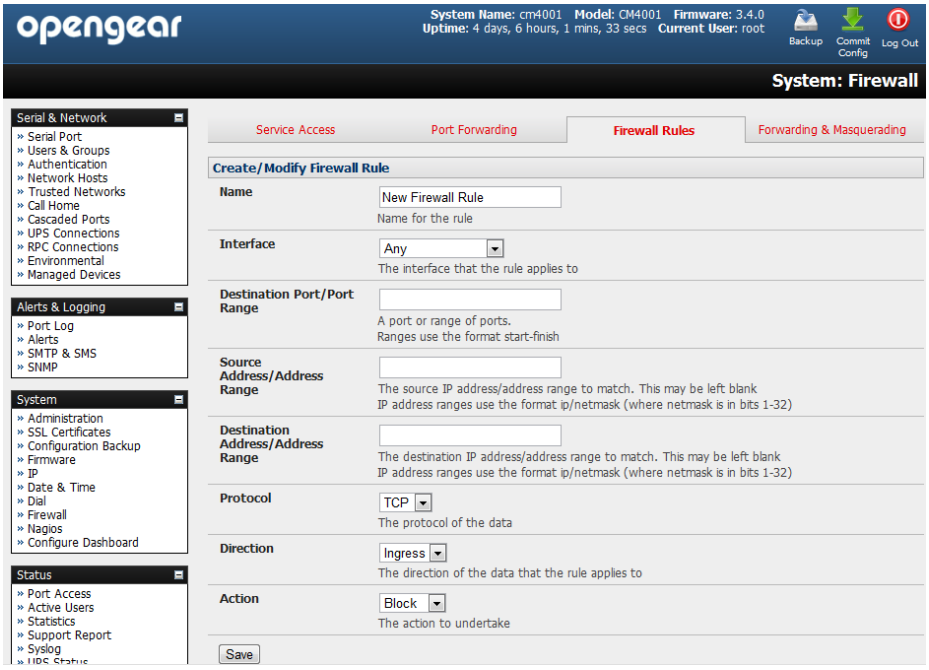
Output Port Range: 443

5.8.4 Firewall rules

Firewall rules can be used to block or allow traffic through an interface based on port number, the source and/or destination IP address (range), the direction (ingress or egress) and the protocol. This can be used to allow custom on-box services, or block traffic based on policy.

To setup a firewall rule:

- Navigate to the **System: Firewall** page, and click on the **Firewall Rules** tab



Note Prior to firmware V3.4 this tab was labeled **Port Rules** and fewer firewall rules could be configured

- Click **New Firewall Rule**
- Fill in the following fields:

Name:	Name the rule. This name should describe the policy the firewall rule is being used to implement (e.g. *block ftp, Allow Tony)*
Interface:	Select the interface that the firewall rule will be applied to (i.e. *Any, Dialout/Cellular, VPN, Network Interface, Dial-in* etc)
Port Range:	Specify the Port or range of Ports (e.g. 1000 – 1500) that the rule will apply to. This may be left blank for Any
Source MAC address	Specify the source MAC address to be matched. This may be left blank for any. MAC addresses use the format XX:XX:XX:XX:XX:XX, where XX are hex digits
Source Address Range:	Specify the source IP address (or address range) to match. IP address ranges use the format ip/netmask (where netmask is in bits 1-32). This may be left blank for Any
Destination Range:	Specify the destination IP address/address range to match. IP address ranges use the format ip/netmask (where netmask is in bits 1-32). This may be left blank.
Protocol:	Select if the firewall rule will apply to *TCP* or *UDP* or *"TCP and UDP"* or *ICMP* or *ESP* or *GRE* or *Any*
Direction:	Select the traffic direction that the firewall rule will apply to (*Ingress* = incoming or *Egress*)
Action:	Select the action (*Accept* or *Block*) that will be applied to the packets detected that match the Interface+ Port Range+ Source/destination Address Range+ Protocol+ Direction

For example, to block all SSH traffic from leaving Dialout Interface, the following settings can be used:

Interface: Dialout/Cellular

Port Range: 22

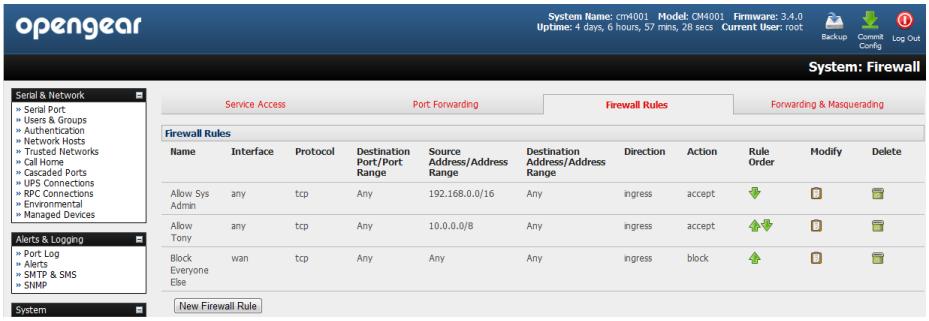
Protocol: TCP

Direction: Egress

Action: Block

The firewall rules are processed in a set order- from top to bottom. So rule placement is important. For example with the following rules, all traffic coming in over the *Network Interface* is blocked except when it comes from two nominated IP addresses (*SysAdmin* and *Tony*):

	To allow all incoming traffic on all interfaces from the SysAdmin:	To allow all incoming traffic from Tony:	To block all incoming traffic from the Network Interface:
Interface	*Any*	*Any*	*Network Interface*
Port Range	*Any*	*Any*	*Any*
Source MAC	*Any*	*Any*	*Any*
Source IP	*IP address of SysAdmin*	*IP address of Tony*	*Any*
Destination IP	*Any*	*Any*	*Any*
Protocol	*TCP*	*TCP*	*TCP*
Direction	*Ingress*	*Ingress*	*Ingress*
Action	*Accept*	*Accept*	*Block*

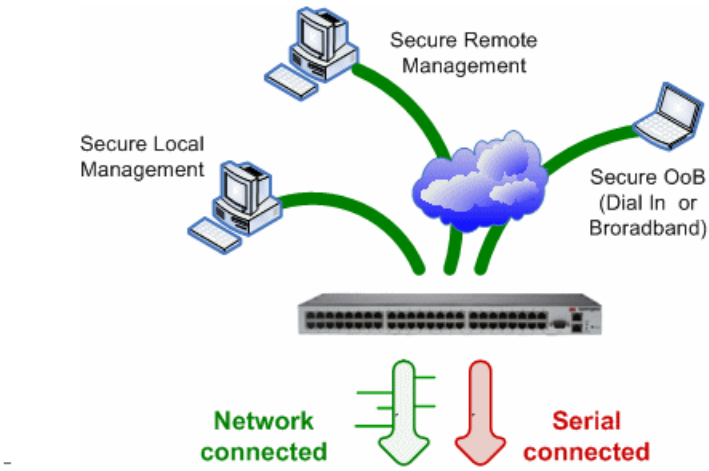


However if the **Rule Order** above was to be changed so the "*Block Everyone Else*" rule was second on the list then the traffic coming in over the *Network Interface* from *Tony* would be blocked.

SSH TUNNELS & SDT CONNECTOR

Each Opengear *console server* has an embedded SSH server and uses SSH tunneling so remote users can securely connect through the *console server* to Managed Devices - using text-based console tools (such as SSH, telnet, SoL) or graphical tools (such VNC, RDP, HTTPS, HTTP, X11, VMware, DRAC, iLO).

The Managed Devices being accessed can be located on the same local network as the *console server* or they can be attached to the *console server* via a serial port. The remote *User/Administrator* connects to the *console server* thru an SSH tunnel via dial-up, wireless or ISDN modem; a broadband Internet connection; the enterprise VPN network or the local network:



To set up the secure SSH tunnel from the Client PC to the *console server*, you must install and launch SSH client software on the User/*Administrator*'s PC. Opengear recommends you use the *SDT Connector* client software that is supplied with the *console server* for this. *SDT Connector* is simple to install and auto-configure and it will provides all your users with point-and-click access to all the systems and devices in the secure network. With one click *SDT Connector* sets up a secure SSH tunnel from the client to the selected *console server*, then establishes a port forward connection to the target network connected host or serial connected device, then executes the client application that will be used in communicating with the host.

This chapter details the basic SDT Connector operations:

- Configuring the *console server* for SSH tunneled access to network attached hosts and setting up permitted Services and user access (*Section 6.1*)
- Setting up the SDT Connector client with gateway, host, service and client application details and making connections between the Client PC and hosts connected to the *console server* (*Section 6.2*)
- Using SDT Connector to browser access the Management Console (*Section 6.3*)
- Using SDT Connector to Telnet or SSH connect to devices that are serially attached to the *console server* (*Section 6.4*)

The chapter then covers more advanced SDT Connector and SSH tunneling topics:

- Using SDT Connector for out of band access(*Section 6.5*)
- Automatic importing and exporting of configurations (*Section 6.6*)
- Configuring Public Key Authentication (*Section 6.7*)
- Setting up a SDT Secure Tunnel for Remote Desktop (*Section 6.8*)
- Setting up a SDT Secure Tunnel for VNC (*Section 6.9*)
- Using SDT to IP connect to hosts that are serially attached to the *console server* (*Section 6.10*)

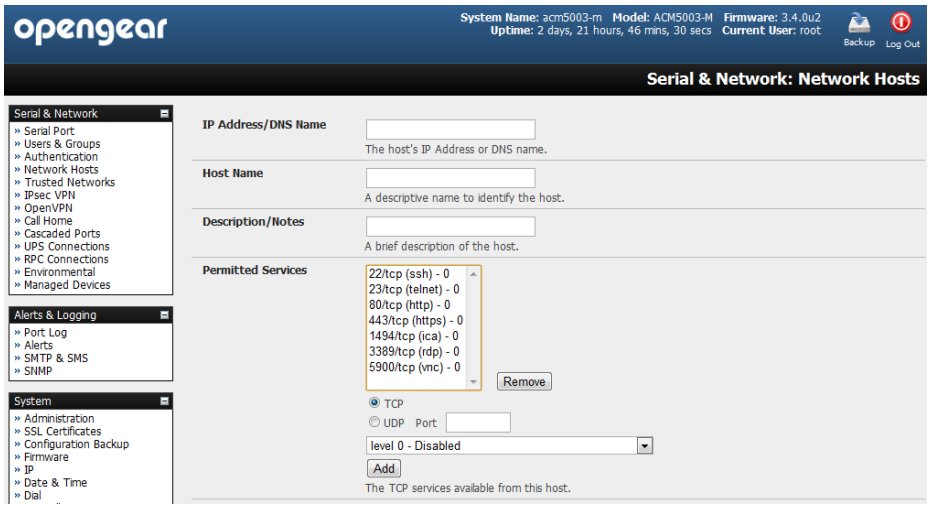
6.1 Configuring for SSH Tunneling to Hosts

To set up the *console server* for SSH tunneled access a network attached *host*:

- Add the new *host* and the *permitted services* using the **Serial & Network: Network Hosts** menu as detailed in *Network Hosts (Chapter 4.4).* Only these *permitted services* will be forwarded through by SSH to the *host*. All other services (TCP/UDP ports) will be blocked.

Note Following are some of the TCP Ports used by SDT in the *console server*:

22	SSH (All SDT Tunneled connections)
23	Telnet on local LAN (forwarded inside tunnel)
80	HTTP on local LAN (forwarded inside tunnel)
3389	RDP on local LAN (forwarded inside tunnel)
5900	VNC on local LAN (forwarded inside tunnel)
73XX	RDP over serial from local LAN – where XX is the serial port number (i.e. 7301to 7348 on a 48 port *console server*)
79XX	VNC over serial from local LAN – where XX is the serial port number



- Add the new *Users* using **Serial & Network: Users & Groups** menu as detailed in *Network Hosts (Chapter 4.4). Users* can be authorized to access the *console server* ports and specified network-attached hosts. To simplify configuration, the *Administrator* can first set up *Groups* with group access permissions, then *Users* can be classified as members of particular *Groups*.

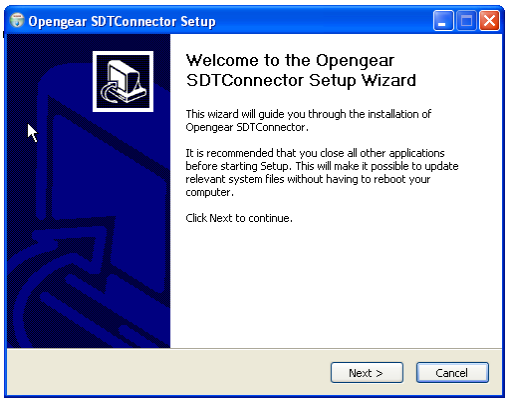
6.2 SDT Connector Client Configuration

The *SDT Connector* client works with all Opengear *console servers*. Each of these remote *console servers* have an embedded OpenSSH based server which can be configured to *port forward* connections from the *SDT Connector* client to hosts on their local network as detailed in the previous chapter. The *SDT Connector* can also be pre-configured with the access tools and applications that will be available to be run when access to a particular host has been established.

SDT Connector can connect to the *console server* using an alternate OOB access. It can also access the *console server* itself and access devices connected to serial ports on the *console server*.

6.2.1 SDT Connector client installation

- The *SDT Connector* set up program (***SDTConnector Setup-1.n.exe*** or ***sdtcon-1.n.tar.gz***) is included on the CD supplied with your Opengear *console server* product (or a copy can be freely download from Opengear's website)
- Run the set-up program:



Note For Windows clients, the *SDTConnectorSetup-1.n.exe* application will install the *SDT Connector 1.n.exe* and the config file *defaults.xml.* If there is already a config file on the Windows PC then it will not be overwritten. To remove earlier config file run the *regedit* command and search for "SDT Connector" then remove the directory with this name.

For Linux and other Unix clients, *SDTConnector.tar.gz* application will install the *sdtcon-1.n.jar* and the config file *defaults.xml*

Once the installer completes you will have a working *SDT Connector* client installed on your machine and an icon on your desktop:

- Click the *SDT Connector* icon on your desktop to start the client

Note *SDT Connector* is a Java application so it must have a Java Runtime Environment (JRE) installed. It will install on Windows 2000 and later and on most Linux platforms. Solaris platforms are also supported however they must have Firefox installed. *SDT Connector* can run on any system with Java 1.4.2 and above installed, but it assumes the web browser is Firefox, and that *xterm -e telnet* opens a telnet window

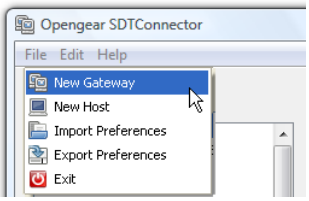
To operate *SDT Connector,* you first need to add new gateways to the client software by entering the access details for each *console server* (refer *Section 6.2.2*) then let the client auto-configure with all host and serial port connections from each *console server* (refer *Section 6.2.3*) then point-and-click to connect to the Hosts and serial devices(refer *Section 6.2.4*)

Alternately you can manually add network connected hosts (refer *Section 6.2.5*) and manually configure new services to be used in accessing the *console server* and the hosts (refer *Section 6.2.6*) then manually configuring clients to run on the PC that will use the service to connect to the hosts and serial port devices (refer *Section 6.2.7 and 6.2.9*). *SDT Connector* can also be set up to make an out-of-band connection to the *console server* (refer *Section 6.2.9*)

6.2.2 Configuring a new gateway in the SDT Connector client

To create a secure SSH tunnel to a new *console server*:

- Click the *New Gateway* icon or select the **File: New Gateway** menu option

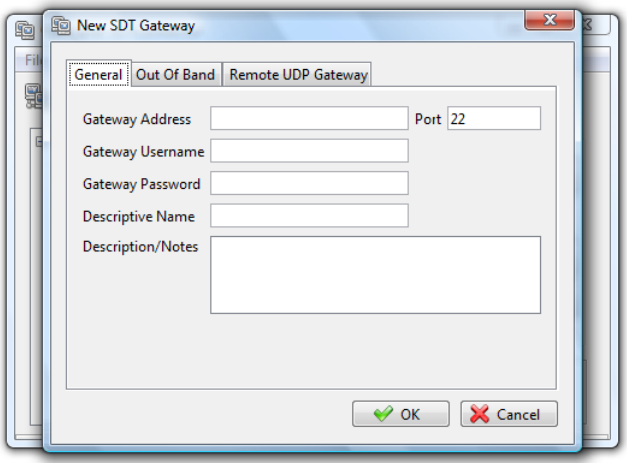


- Enter the IP or DNS **Address** of the *console server* and the SSH port that will be used (typically 22)

Note If *SDT Connector* is connecting to a remote *console server* through the public Internet or routed network you will need to:

- Determine *the public IP address* of the *console server* (or of the router / firewall that connects the *console server* to the Internet) as assigned by the ISP. One way to find the public IP address is to access / or / from a computer on the same network as the *console server* and note the reported IP address
- Set port forwarding for TCP port 22 through any firewall/NAT/router that is located between *SDT Connector* and the *console server* so it points to the *console server*. http://www.portforward.com has port forwarding instructions for a range of routers. Also you can use the Open Port Check tool from http://www.canyouseeme.org to check if port forwarding through local firewall/NAT/router devices has been properly configured

- Enter the **Username** and **Password** of a user on the gateway that has been enabled to connect via SSH and/or create SSH port redirections

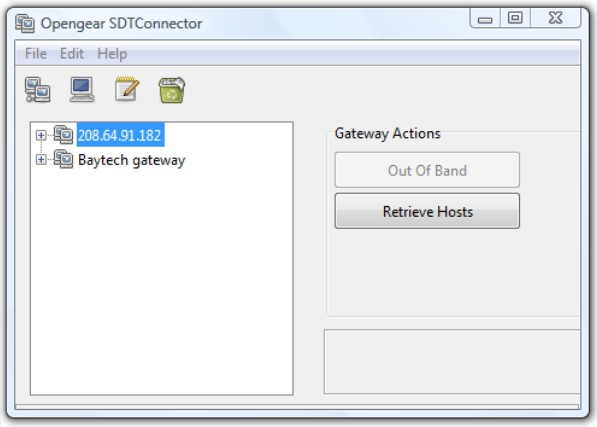


- Optionally, enter a **Descriptive Name** to display instead of the IP or DNS address, and any **Notes** or a **Description** of this gateway (such as its firmware version, site location or anything special about its network configuration).
- Click **OK** and an icon for the new gateway will now appear in the *SDT Connector* home page

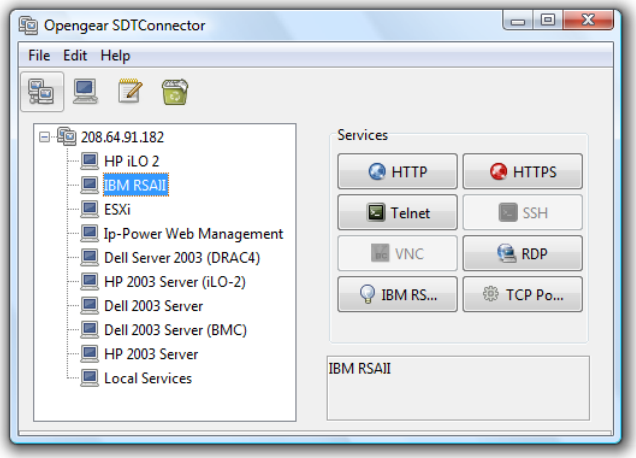
Note For an *SDT Connector* user to access a *console server* (and then access specific hosts or serial devices connected to that *console server*), that user must first be setup on the *console server*, and must be authorized to access the specific ports / hosts (refer Chapter 5) and only these *permitted services* will be forwarded through by SSH to the Host. All other services (TCP/UDP ports) will be blocked.

6.2.3 Auto-configure SDT Connector client with the user's access privileges

Each user on the *console server* has an access profile which has been configured with those specific connected hosts and serial port devices the user has authority to access, and a specific set of the enabled services for each of these. This configuration can be auto-uploaded into the SDT Connector client:



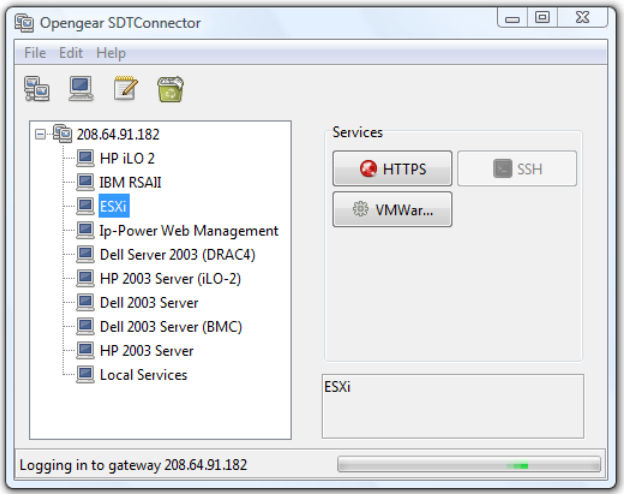
- Click on the new gateway icon and select **Retrieve Hosts**. This will:
  - configure access to network connected Hosts that the user is authorized to access and set up (for each of these Hosts) the services (e.g. HTTPS, IPMI2.0) and the related IP ports being redirected
  - configure access to the *console server* itself (this is shown as a *Local Services* host)
  - configure access with the enabled services for the serial port devices connected to the *console server*



Note The Retrieve Hosts function will auto-configure all classes of user (i.e. they can be members of *user* or *admin* or some other group or no group) however SDT Connector will not auto-configure the *root* (and it recommended that this account is only used for initial config and for adding an initial *admin* account to the *console server*)

6.2.4 Make an SDT connection through the gateway to a host

- Simply ***point*** at the host to be accessed ***and click*** on the service to be used in accessing that host. The SSH tunnel to the gateway is then automatically established, the appropriate ports redirected through to the host, and the appropriate local client application is launched pointing at the local endpoint of the redirection:



Note The SDT Connector client can be configured with unlimited number of Gateways. Each Gateway can be configured to port forward to an unlimited number of locally networked Hosts. Similarly there is no limit on the number of SDT Connector clients who can be configured to access the one Gateway. Nor are there limits on the number of Host connections that an SDT Connector client can concurrently have open through the one Gateway tunnel.

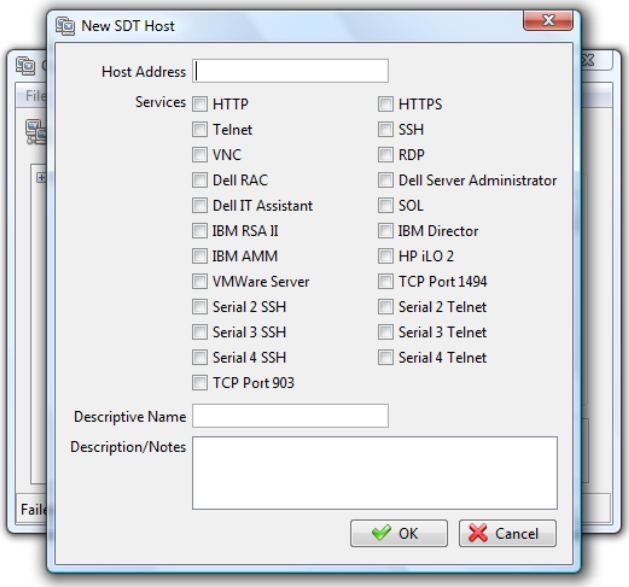
However there is a limit on the number of SDT Connector SSH tunnels that can be open at the one time on a particular Gateway. SD4001/4002 devices support at least 10 simultaneous client tunnels; IM4200, ACM5000 and ACM5500 models each support at least 50 such concurrent connections. So for a site with a IM4200 gateway you can have, at any time up to 50 users securely controlling an unlimited number of network attached computers and appliances (servers, routers, etc.) at that site.

ACM7000, IM7200 and CM7100 support many hundreds of simultaneous client tunnels.

6.2.5 Manually adding hosts to the SDT Connector gateway

For each gateway, you can manually specify the network connected hosts that will be accessed through that *console server*; and for each host, specify the services that will used in communicating with the host

- Select the newly added gateway and click the *Host* icon to create a host that will be accessible via this gateway. (Alternatively select **File: New Host**)

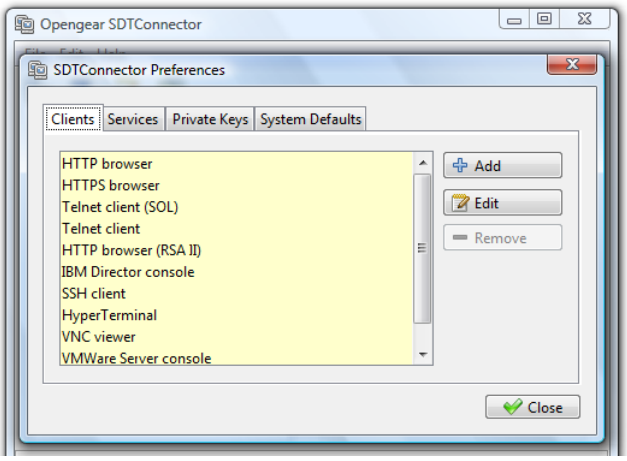


- Enter the IP or DNS **Host Address** of the host (if this is a DNS address, it must be resolvable by the gateway)
- Select which **Services** are to be used in accessing the new host. A range of service options are pre-configured in the default *SDT Connector* client (RDP, VNC, HTTP, HTTPS, Dell RAC, VMware etc). However if you wish to add new services the range then proceed to the next section (**Adding a new service**) then return here
- Optionally, enter a **Descriptive Name** for the host, to display instead of the IP or DNS address, and any **Notes** or a **Description** of this host (such as its operating system/release, or anything special about its configuration)
- Click **OK**

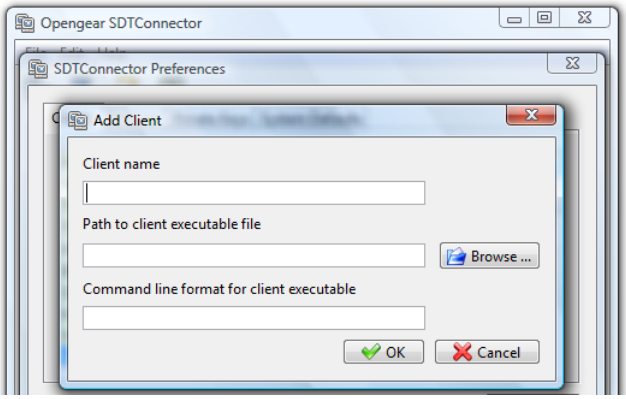
6.2.6 Manually adding new services to the new hosts

To extend the range of services that can be used when accessing hosts with *SDT Connector*:

- Select **Edit: Preferences** and click the **Services** tab. Click **Add**
- Enter a **Service Name** and click **Add**
- Under the **General** tab, enter the TCP Port that this service runs on (e.g. 80 for HTTP). Optionally, select the client to use to access the local endpoint of the redirection



- Select which **Client** application is associated with the new service. A range of client application options are pre-configured in the default *SDT Connector* (RDP client, VNC client, HTTP browser, HTTPS browser, Telnet client etc). However if you wish to add new client applications to this range then proceed to the next section (**Adding a new client**) then return here

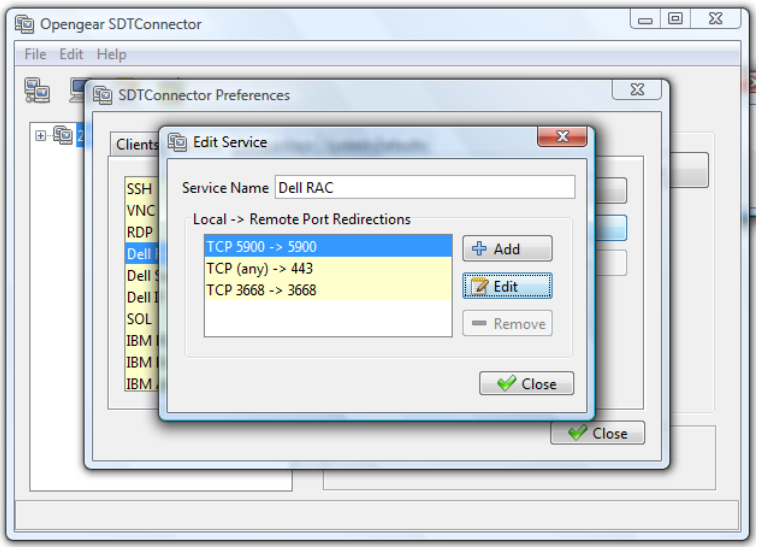


- Click **OK**, then **Close**

A service typically consists of a single SSH port redirection and a local client to access it. However it may consist of several redirections; some or all of which may have clients associated with them.

An example is the Dell RAC service. The first redirection is for the HTTPS connection to the RAC server - it has a client associated with it (web browser) that is launched immediately upon clicking the button for this service.

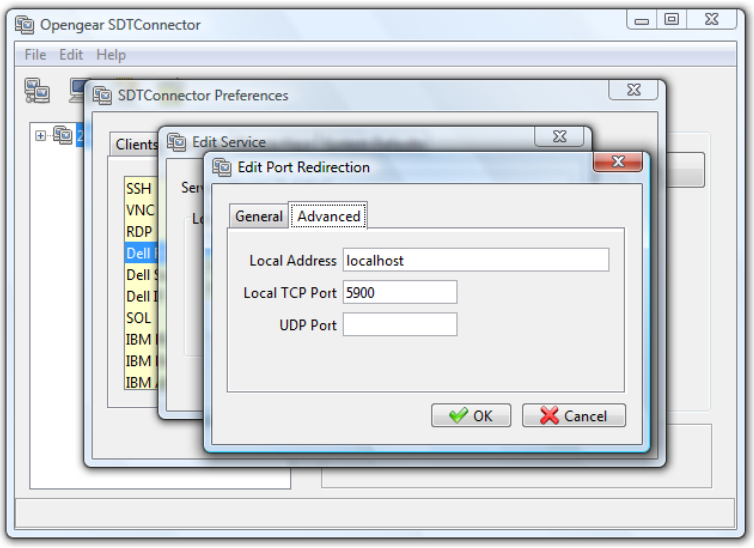
The second redirection is for the VNC service that the user may choose to later launch from the RAC web console. It is automatically loads in a Java client served through the web browser, so it does not need a local client associated with it.



- On the Add Service screen you can click **Add** as many times as needed to add multiple new port redirections and associated clients

You may also specify **Advanced** port redirection options:

- Enter the local address to bind to when creating the local endpoint of the redirection. It is not usually necessary to change this from "localhost".
- Enter a local TCP port to bind to when creating the local endpoint of the redirection. If this is left blank, a random port will be selected.



Note *SDT Connector* can also tunnel UDP services. *SDT Connector* tunnels the UDP traffic through the TCP SSH redirection, so in effect it is a tunnel within a tunnel.

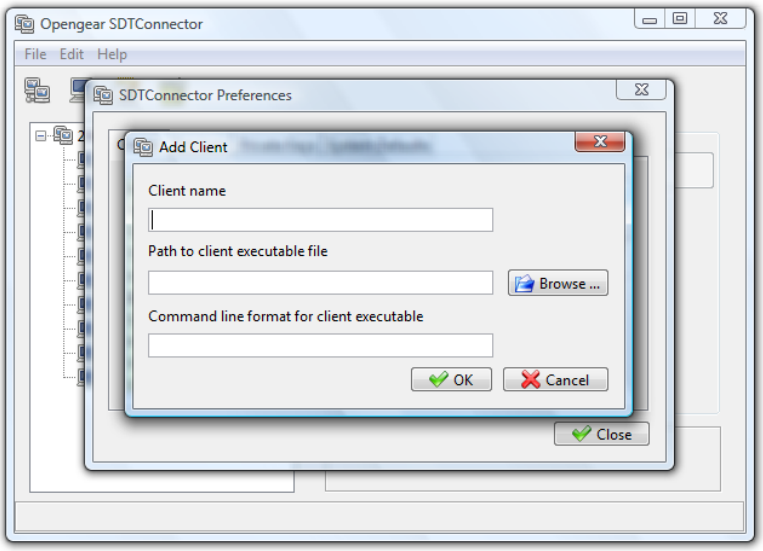
Enter the UDP port on which the service is running on the host. This will also be the local UDP port that *SDT Connector* binds as the local endpoint of the tunnel.

Note that for UDP services, you still need to specify a TCP port under General. This will be an arbitrary TCP port that is not in use on the gateway. An example of this is the SOL Proxy service. It redirects local UDP port 623 to remote UDP port 623 over the arbitrary TCP port 6667

6.2.7 Adding a client program to be started for the new service

Clients are local applications that may be launched when a related service is clicked. To add to the pool of client programs:

- Select **Edit: Preferences** and click the **Client** tab. Click **Add**



- Enter a **Name** for the client. Enter the **Path** to the executable file for the client (or click **Browse** to locate the executable)
- Enter a **Command Line** associated with launching the client application. *SDT Connector* typically launches a client using command line arguments to point it at the local endpoint of the redirection. There are three special

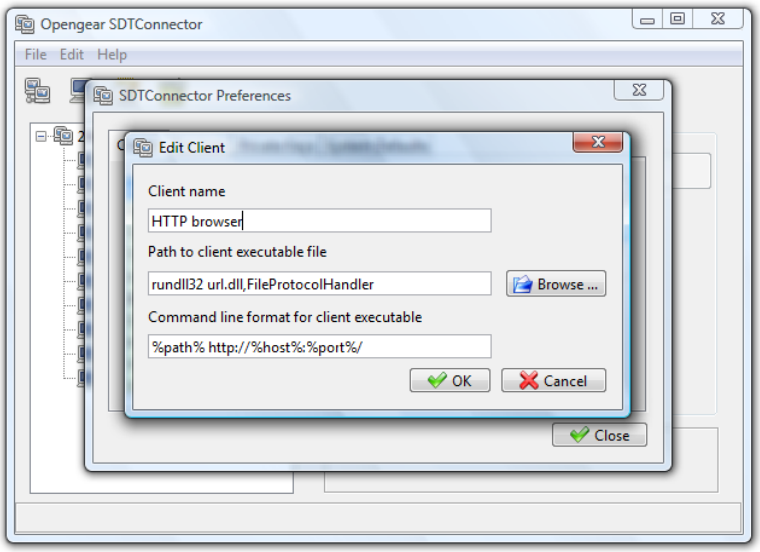
keywords for specifying the command line format. When launching the client, *SDT Connector* substitutes these keywords with the appropriate values:

%path% is path to the executable file, i.e. the previous field.

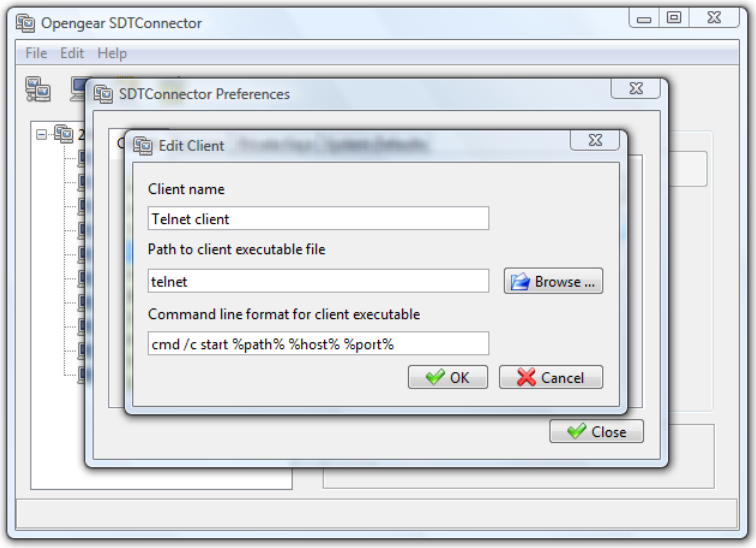
%host% is the local address to which the local endpoint of the redirection is bound, i.e. the Local Address field for the Service redirection Advanced options.

%port% is the local port to which the local endpoint of the redirection is bound, i.e. the Local TCP Port field for the Service redirection Advanced options. If this port is unspecified (i.e. "Any"), the appropriate randomly selected port will be substituted.

For example *SDT Connector* is preconfigured for Windows installations with a HTTP service client that will connect with whichever local browser the local Windows user has configured as the default. Otherwise the default browser used is Firefox:



Also some clients are launched in a command line or terminal window. The Telnet client is an example of this so the "Path to client executable file" is *telnet* and the "Command line format for client executable" is *cmd /c start %path% %host% %port%* :



- Click **OK**

6.2.8 Dial in configuration

If the client PC is dialing into *Local/Console* port on the *console server* you will need to set up a dial-in PPP link:

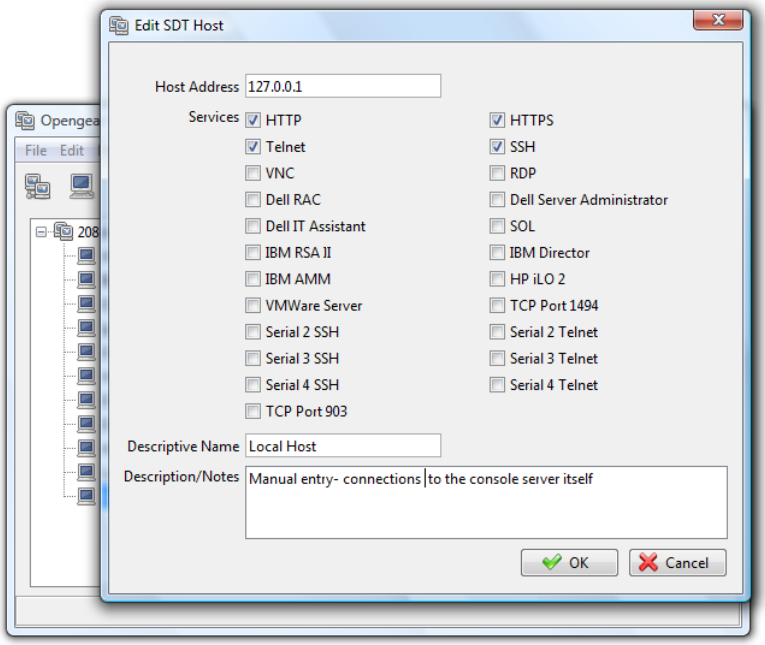
- Configure the *console server* for dial-in access (following the steps in the ***Configuring for Dial-In PPP Access*** section in *Chapter 5*, *Configuring Dial In Access*)
- Set up the PPP client software at the remote *User* PC (following the ***Set up the remote Client*** section in *Chapter 5*)

Once you have a dial-in PPP connection established, you then can set up the secure SSH tunnel from the remote Client PC to the *console server*.

6.3 SDT Connector to Management Console

SDT Connector can also be configured for browser access the gateway's Management Console – and for Telnet or SSH access to the gateway command line. For these connections to the gateway itself, you must configure *SDT Connector* to access the gateway (itself) by setting the *Console server* up as a *host*, and then configuring the appropriate services:

- Launch *SDT Connector* on your PC. Assuming you have already set up the *console server* as a *Gateway* in your *SDT Connector* client (with *username / password* etc) select this newly added *Gateway* and click the Host icon to create a host. Alternatively, select **File: New Host**
- Enter 127.0.0.1 as the **Host Address** and give some details in **Descriptive Name/Notes**. Click OK



- Click the **HTTP** or **HTTPS** Services icon to access the gateway's Management Console, and/or click **SSH** or **Telnet** to access the gateway command line console

---

Note: To enable SDT access to the gateway console, you must configure the *console server* to allow port forwarded network access to itself.

With V3.3 firmware and later, this can be done using the *console server* Management Console. Simply browse to the *console server* and select the **Service Access** tab on the **System: Firewall** menu. Ensure **SSH Command Shell** is enabled on the Network interface and any out of band interfaces.

With earlier firmware:

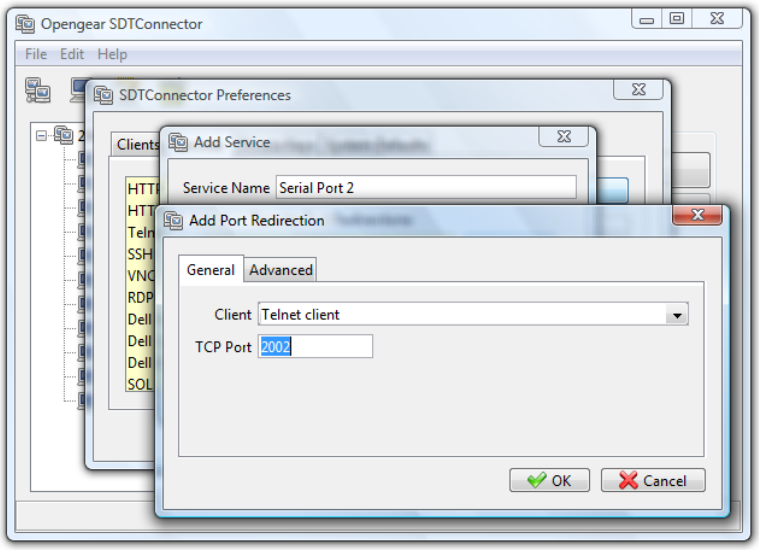
- Browse to the *console server* and select **Network Hosts** from **Serial & Network**, click **Add Host** and in the **IP Address/DNS Name** field enter 127.0.0.1 (this is the Opengear's network loopback address) and enter *Loopback* in **Description**

- Remove all entries under **Permitted Services** except for those that will be used in accessing the Management Console (80/http or 443/https) or the command line (22/ssh or 23/telnet) then scroll to the bottom and click **Apply**
- *Administrators* by default have gateway access privileges, however for *Users* to access the gateway Management Console you will need to give those *Users* the required access privileges. Select **Users & Groups** from **Serial & Network**. Click **Add User**. Enter a **Username**, **Description** and **Password/Confirm**. Select 127.0.0.1 from **Accessible Host**(s) and click **Apply**

6.4 SDT Connector - telnet or SSH connect to serially attached devices

SDT Connector can also be used to access text consoles on devices that are attached to the *console server* serial ports. For these connections, you must configure the *SDT Connector* client software with a Service that will access the target gateway serial port, and then set the gateway up as a host:

- Launch *SDT Connector* on your PC. Select **Edit: Preferences** and click the **Services** tab. Click **Add**
- Enter *"Serial Port 2"* in **Service Name** and click **Add**
- Select **Telnet** client as the Client. Enter 2002 in **TCP Port**. Click **OK**, then **Close** and **Close** again



- Assuming you have already set up the target *console server* as a *gateway* in your *SDT Connector* client (with *username / password* etc), select this *gateway* and click the **Host** icon to create a host. Alternatively, select **File: New Host**.
- Enter 127.0.0.1 as the **Host Address** and select **Serial Port 2** for Service. In **Descriptive Name**, enter something along the lines of Loopback ports, or Local serial ports. Click **OK.**
- Click *Serial Port 2* icon for Telnet access to the serial console on the device attached to serial port #2 on the gateway

To enable *SDT Connector* to access to devices connected to the gateway's serial ports, you must also configure the *Console server* itself to allow port forwarded network access to itself, and enable access to the nominated serial port:

- Browse to the *Console server* and select **Serial Port** from **Serial & Network**
- Click **Edit** next to selected Port # (e.g. Port 2 if the target device is attached to the second serial port). Ensure the port's serial configuration is appropriate for the attached device
- Scroll down to **Console Server Setting** and select **Console Server Mode**. Check **Telnet** (or **SSH**) and scroll to the bottom and click **Apply**
- Select **Network Hosts** from **Serial & Network** and click **Add Host**

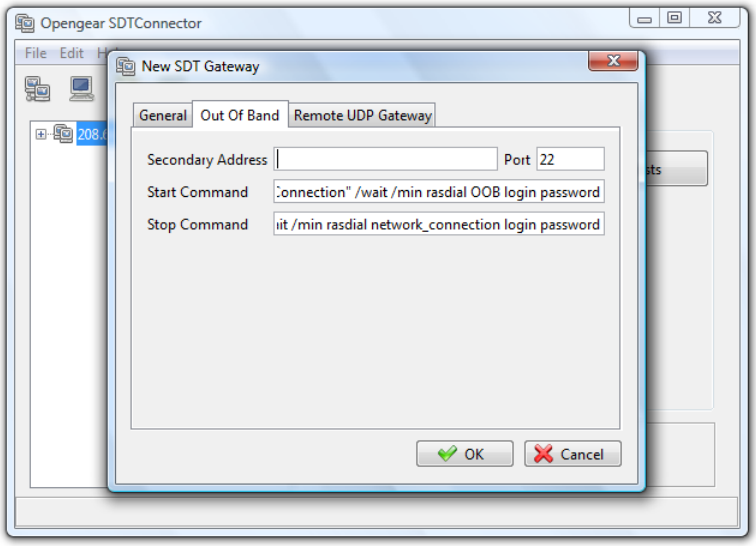
- In the **IP Address/DNS Name** field enter *127.0.0.1* (this is the Opengear's network loopback address) and enter *Loopback* in **Description**
- Remove all entries under **Permitted Services** and select **TCP** and enter *200n* in **Port**. (This configures the Telnet port enabled in the previous step, so for Port 2 you would enter *2002*)
- Click **Add** then scroll to the bottom and click **Apply**
- *Administrator*s by default have gateway and serial port access privileges; however for *Users* to access the gateway and the serial port, you will need to give those *Users* the required access privileges. Select **Users & Groups** from **Serial & Network**. Click **Add User**. Enter a **Username**, **Description** and **Password/Confirm**. Select 127.0.0.1 from **Accessible Host**(s) and select Port 2 from Accessible Port(s). Click **Apply.**

6.5 Using SDT Connector for out-of-band connection to the gateway

SDT Connector can also be set up to connect to the *console server* (gateway) out-of-band (OOB). OOB access uses an alternate path for connecting to the gateway to that used for regular data traffic. OOB access is useful for when the primary link into the gateway is unavailable or unreliable.

Typically a gateway's primary link is a broadband Internet connection or Internet connection via a LAN or VPN, and the secondary out-of-band connectivity is provided by a dial-up or wireless modem directly attached to the gateway. So out-of-band access enables you to access the hosts and serial devices on the network, diagnose any connectivity issues, and restore the gateway's primary link.

In *SDT Connector*, OOB access is configured by providing the secondary IP address of the gateway, and telling *SDT Connector* how to start and stop the OOB connection. Starting an OOB connection may be achieved by initiating a dial up connection, or adding an alternate route to the gateway. *SDT Connector* allows for maximum flexibility is this regard, by allowing you to provide your own scripts or commands for starting and stopping the OOB connection.



To configure *SDT Connector* for OOB access:

- When adding a new gateway or editing an existing gateway select the **Out Of Band** tab
- Enter the secondary, OOB IP address of the gateway (e.g. the IP address it is accessible using when dialed in directly). You also may modify the gateway's SSH port if it's not using the default of 22
- Enter the command or path to a script to start the OOB connection in **Start Command**
  - To initiate a pre-configured dial-up connection under Windows, use the following Start Command:

    cmd /c start "Starting Out of Band Connection" /wait /min rasdial network_connection login password

    where *network_connection* is the name of the network connection as displayed in *Control Panel -> Network Connections*, *login* is the dial-in username, and *password* is the dial-in password for the connection.

- To initiate a pre-configured dial-up connection under Linux, use the following Start Command:

  pon network_connection

  where *network_connection* is the name of the connection.

➢ Enter the command or path to a script to stop the OOB connection in **Stop Command**

- To stop a pre-configured dial-up connection under Windows, use the following Stop Command:

  cmd /c start "Stopping Out of Band Connection" /wait /min rasdial network_connection /disconnect

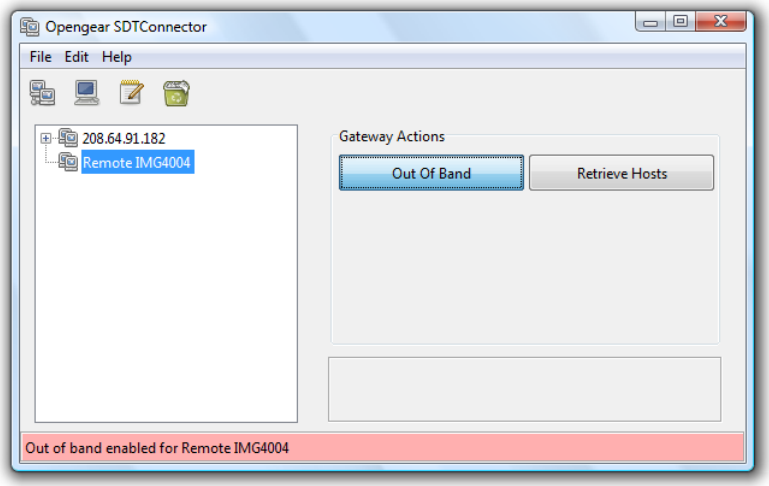
  where *network connection* is the name of the network connection as displayed in *Control Panel -> Network Connections*.

- To stop a pre-configured dial-up connection under Linux, use the following Stop Command:

  poff network_connection

To make the OOB connection using *SDT Connector:*

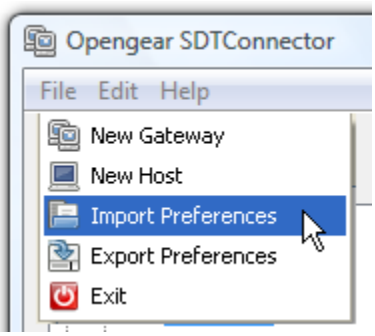
➢ Select the gateway and click Out Of Band. The status bar will change color to indicate this gateway is now being access using the OOB link rather than the primary link



When you connect to a service on a host behind the gateway, or to the *console server* gateway itself, *SDT Connector* will initiate the OOB connection using the provided Start Command. The OOB connection isn't stopped (using the provided Stop Command) until Out Of Band under Gateway Actions is clicked off, at which point the status bar will return to its normal color.

6.6 Importing (and exporting) preferences

To enable the distribution of pre-configured client config files, *SDT Connector* has an *Export/Import* facility:



- To save a configuration .xml file (for backup or for importing into other *SDT Connector* clients) select **File: Export Preferences** and select the location to save the configuration file
- To import a configuration select **File: Import Preferences** and select the .xml configuration file to be installed

6.7 SDT Connector Public Key Authentication

SDT Connector can authenticate against an SSH gateway using your SSH key pair rather than requiring your to enter your password. This is known as public key authentication.

To use public key authentication with SDT Connector, first you must add the public part of your SSH key pair to your SSH gateway:

- Ensure the SSH gateway allows public key authentication, this is typically the default behavior
- If you do not already have a public/private key pair for your client PC (the one running SDT Connector on) generate them now using *ssh-keygen, PuTTYgen* or a similar tool. You may use RSA or DSA, however it is important that you leave the passphrase field blank:
  - PuTTYgen: http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html
  - OpenSSH: http://www.openssh.org/
  - OpenSSH (Windows): http://sshwindows.sourceforge.net/download/
- Upload the public part of your SSH key pair (this file is typically named *id_rsa.pub* or *id_dsa.pub*) to the SSH gateway, or otherwise add to *.ssh/authorized keys* in your home directory on the SSH gateway
- Next, add the private part of your SSH key pair (this file is typically named *id_rsa* or *id_dsa*) to SDT Connector. Click **Edit: Preferences: Private Keys: Add**, locate the private key file and click **OK**

You do not have to add the public part of your SSH key pair, it is calculated using the private key.

SDT Connector will now use public key authentication when connecting through the SSH gateway (*console server*). You may have to restart SDT Connector to shut down any existing tunnels that were established using password authentication.

Also if you have a host behind the *console server* that you connect to by clicking the SSH button in SDT Connector you may also wish to configure access to it for public key authentication as well. This configuration is entirely independent of SDT Connector and the SSH gateway. You must configure the SSH client that SDT Connector launches (e.g. Putty, OpenSSH) and the host's SSH server for public key authentication. Essentially what you are using is SSH over SSH, and the two SSH connections are entirely separate.

6.8 Setting up SDT for Remote Desktop access

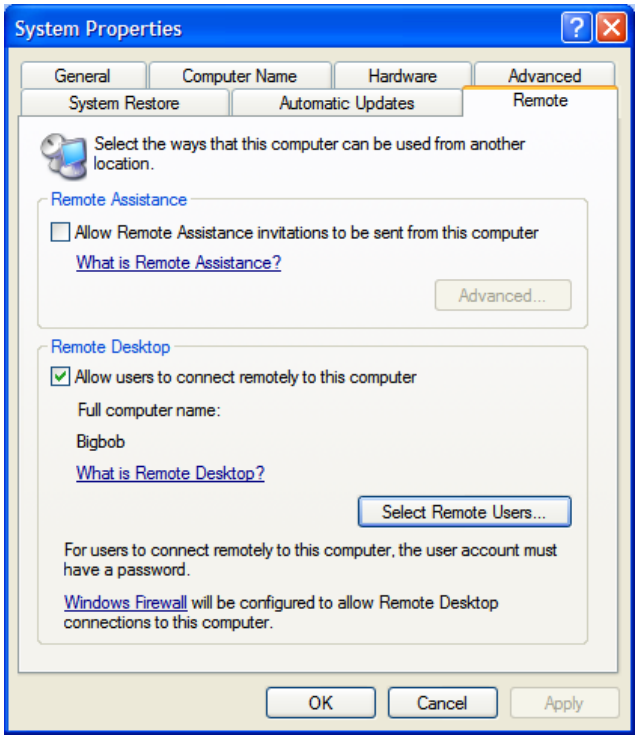
Microsoft's Remote Desktop Protocol (RDP) enables the system manager to securely access and manages remote Windows computers – to reconfigure applications and user profiles, upgrade the server's operating system, reboot the machine etc. Opengear's Secure Tunneling uses SSH tunneling, so this RDP traffic is securely transferred through an authenticated and encrypted tunnel.

SDT with RDP also allows remote *Users* to connect to Windows XP and later computers and to Windows 2000 Terminal Servers; and to have access to all of the applications, files, and network resources (with full graphical interface just as though they were in front of the computer screen at work). To set up a secure Remote Desktop connection you must enable Remote Desktop on the target Windows computer that is to be accessed and configure the RPD client software on the client PC.

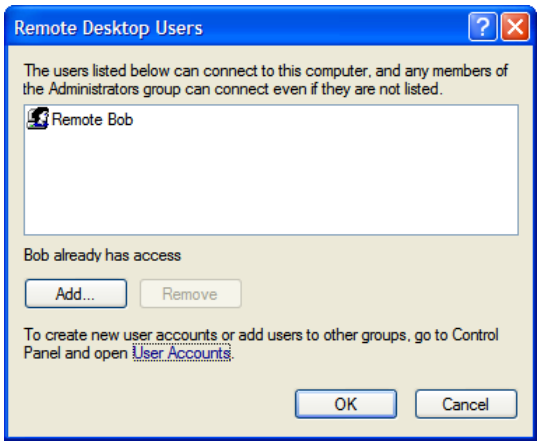
6.8.1 Enable Remote Desktop on the target Windows computer to be accessed

To enable **Remote Desktop** on the Windows computer being accessed:

- Open **System** in the Control Panel and click the **Remote** tab



- Check **Allow users to connect remotely to this computer**
- Click **Select Remote Users**



- To set the user(s) who can remotely access the system with RDP click **Add** on the **Remote Desktop Users** dialog box

Note If you need to set up new users for Remote Desktop access, open **User Accounts** in the Control Panel and proceed through the steps to nominate the new user's name, password and account type (*Administrator* or Limited)

Note With Windows XP Professional and Vista, you have only one Remote Desktop session and it connects directly to the Windows root console. With Windows Server 2008 you can have multiple sessions (and with Server 2003 you have three sessions - the console session and two other general sessions). So more than one user can have active sessions on a single computer.

When the remote user connects to the accessed computer on the console session, Remote Desktop automatically locks that computer (so no other user can access the applications and files). When you come back to your computer at work, you can unlock it by typing CTRL+ALT+DEL.

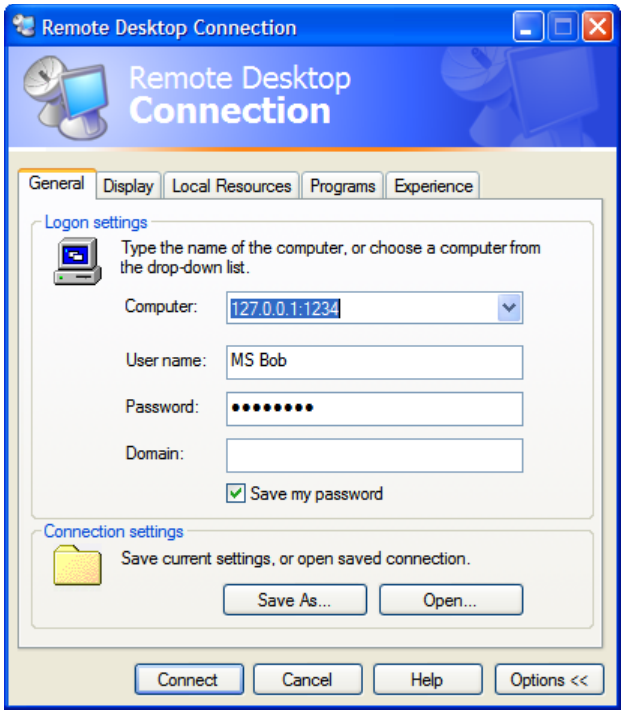
6.8.2 Configure the Remote Desktop Connection client

Now you have the Client PC securely connected to the *console server* (either locally, or remotely - thru the enterprise VPN, or a secure SSH internet tunnel or a dial-in SSH tunnel) you can establish the Remote Desktop connection from the Client. To do this you simply enable the **Remote Desktop Connection** on the remote client PC then point it to the SDT Secure Tunnel port in the *console server*:

A. On a Windows client PC

- Click **Start**. Point to **Programs**, then to **Accessories**, then **Communications**, and click **Remote Desktop Connection**

- In **Computer**, enter the appropriate IP Address and Port Number:
  - Where there is a direct local or enterprise VPN connection, enter the IP Address of the *console server*, and the Port Number of the SDT Secure Tunnel for the *console server* serial port that is attached to the Windows computer to be controlled e.g. if the Windows computer is connected to serial Port 3 on a *console server* located at 192.168.0.50 then you would enter *192.168.0.50:7303*
  - Where there is an SSH tunnel (over a dial up PPP connection or over a public internet connection or private network connection) simply enter the *localhost* as the IP address i.e. *127.0.0.1* For Port Number, enter the *source port* you created when setting SSH tunneling /port forwarding (in Section 6.1.6) e.g. *:1234*
- Click **Option.** In the **Display** section specify an appropriate color depth (e.g. for a modem connection it is recommended you not use over 256 colors). In **Local Resources** specify the peripherals on the remote Windows computer that are to be controlled (printer, serial port etc)



➢ Click **Connect**

Note The Remote Desktop Connection software is pre-installed with Windows XP and later, however for earlier Windows PCs you will need to download the RDP client:

- Go to the Microsoft Download Center site http://www.microsoft.com/downloads/details.aspx?familyid=80111F21-D48D-426E-96C2-08AA2BD23A49&displaylang=en and click the **Download** button

  This software package will install the client portion of Remote Desktop on Windows 95, Windows 98 and 98 Second Edition, Windows Me, Windows NT 4.0 and Windows 2000. When run, this software allows these older Windows platforms to remotely connect to a computer running current Windows.

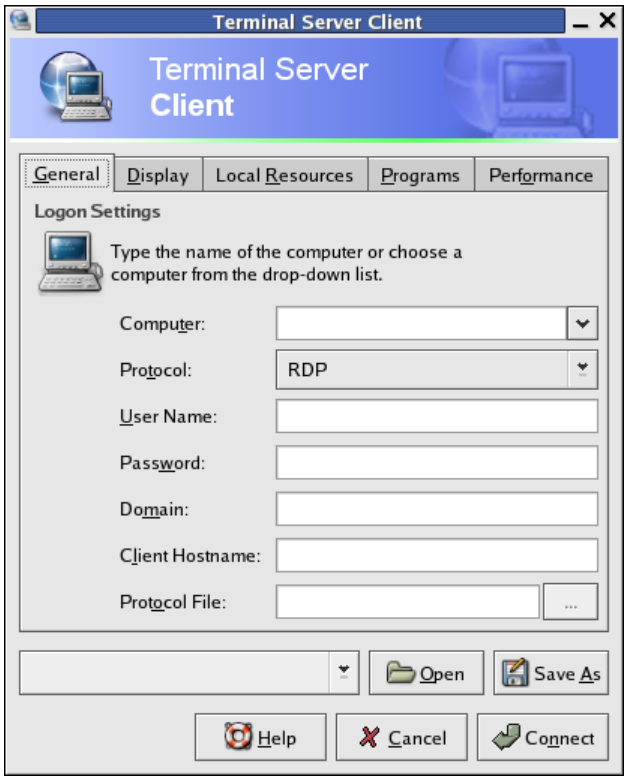
B. On a Linux or UNIX client PC:

➢ Launch the open source *rdesktop* client:

rdesktop -u *windows-user-id* -p *windows-password* -g 1200x950 *ms-windows-terminal-server-host-name*

option	description
-a	Color depth: 8, 16, 24
-r	Device redirection. i.e. Redirect sound on remote machine to local device i.e. -0 -r sound (MS/Windows 2003)
-g	Geometry: *widthxheight* or 70% screen percentage.
-p	Use -p - to receive password prompt.

➢ You can use GUI front end tools like the GNOME Terminal Services Client *tsclient* to configure and launch the *rdesktop* client. (Using *tsclient* also enables you to store multiple configurations of *rdesktop* for connection to many servers)



Note The *rdesktop* client is supplied with Red Hat 9.0:

- rpm -ivh rdesktop-1.2.0-1.i386.rpm

For Red Hat 8.0 or other distributions of Linux; download source, untar, configure, make, make then install.

rdesktop currently runs on most UNIX based platforms with the X Window System and can be downloaded from /

C. On a Macintosh client:

➢ Download Microsoft's free Remote Desktop Connection client for Mac OS X
http://www.microsoft.com/mac/otherproducts/otherproducts.aspx?pid=remotedesktopclient

6.9 SDT SSH Tunnel for VNC

Alternately, with SDT and Virtual Network Computing (VNC), *Users* and *Administrators* can securely access and control Windows, Linux, Macintosh, Solaris and UNIX computers. There's a range of popular VNC software available (UltraVNC, RealVNC, TightVNC) - freely and commercially. To set up a secure VNC connection you must install and configure the VNC Server software on the computer to be accessed, then install and configure the VNC Viewer software on the Viewer PC.

6.9.1 Install and configure the VNC Server on the computer to be accessed

Virtual Network Computing (VNC) software enables users to remotely access computers running Linux, Macintosh, Solaris, UNIX, all versions of Windows and most other operating systems.

A. For Microsoft Windows servers (and clients):

Windows does not include VNC software, so you will need to download, install and activate a third party VNC Server software package:

RealVNC http://www.realvnc.com is fully cross-platform, so a desktop running on a Linux machine may be displayed on a Windows PC, on a Solaris machine, or on any number of other architectures. There is a Windows server, allowing you to view the desktop of a remote Windows machine on any of these platforms using exactly the same viewer. RealVNC was founded by members of the AT&T team who originally developed VNC.

TightVNC http://www.tightvnc.com is an enhanced version of VNC. It has added features such as file transfer, performance improvements, and read-only password support. They have just recently included a video drive much like UltraVNC. TightVNC is still free, cross-platform (Windows Unix and Linux) and compatible with the standard (Real) VNC.

UltraVNC http://ultravnc.com is easy to use, fast and free VNC software that has pioneered and perfected features that the other flavors have consistently refused or been very slow to implement for cross platform and minimalist reasons. UltraVNC runs under Windows operating systems (95, 98, Me, NT4, 2000, XP, 2003) Download UltraVNC from Sourceforge's UltraVNC file list

B. For Linux servers (and clients):

Most Linux distributions now include VNC Servers and Viewers and they are generally can be launched from the (Gnome/KDE etc) front end e.g. with Red Hat Enterprise Linux 4 there's VNC Server software and a choice of Viewer client software, and to launch:

- Select the **Remote Desktop** entry in the **Main Menu: Preferences** menu
- Click the **Allow other users…** checkbox to allow remote users to view and control your desktop

- To set up a persistent VNC server on Red Hat Enterprise Linux 4:
  - Set a password using **vncpasswd**
  - Edit **/etc/sysconfig/vncservers**
  - Enable the service with **chkconfig vncserver on**
  - Start the service with **service vncserver start**
  - Edit **/home/*username*/.vnc/xstartup** if you want a more advanced session than just *twm* and an xterm

C. For Macintosh servers (and clients):

OSXvnc http://www.redstonesoftware.com/vnc.html is a robust, full-featured VNC server for Mac OS X that allows any VNC client to remotely view and/or control the Mac OS X machine. OSXvnc is supported by Redstone Software

D. Most other operating systems (Solaris, HPUX, PalmOS etc) either come with VNC bundled, or have third party VNC software that you can download

6.9.2 Install, configure and connect the VNC Viewer

VNC is truly *platform-independent* so a VNC Viewer on any operating system can connect to a VNC Server on any other operating system. There are Viewers (and Servers) from a wide selection of sources (e.g. UltraVNC TightVNC or RealVNC) for most operating systems. There are also a wealth of Java viewers available so that any desktop can be viewed with any Java-capable browser (http://en.wikipedia.org/wiki/VNC lists many of the VNC Viewers sources).

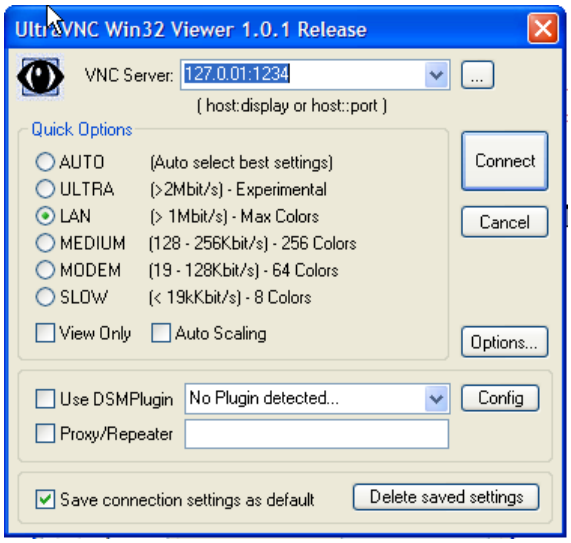
- Install the VNC Viewer software and set it up for the appropriate speed connection

Note To make VNC faster, when you set up the Viewer:
- Set encoding to ZRLE (if you have a fast enough CPU)
- Decrease color level (e.g. 64 bit)
- Disable the background transmission on the Server or use a plain wallpaper

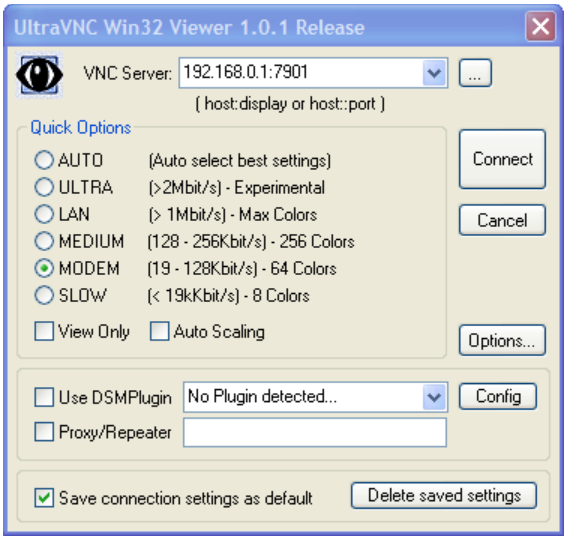
(Refer to http://doc.uvnc.com for detailed configuration instructions)

- To establish the VNC connection, first configure the VNC Viewer, entering the VNC Server IP address

A. When the Viewer PC is connected to the *console server* thru a SSH tunnel (over the public Internet, or a dial-in connection, or private network connection), enter *localhost* (or 127.0.0.1) as the IP VNC Server IP address; and *the source port* you entered when setting SSH tunneling /port forwarding (in Section 6.2.6) e.g. *:1234*



B. When the Viewer PC is connected directly to the *console server* (i.e. locally or remotely through a VPN or dial in connection); and the VNC Host computer is serially connected to the *console server*; enter the IP address of the *console server* unit with the TCP port that the SDT tunnel will use. The TCP port will be 7900 plus the physical serial port number (*i.e.* 7901 to 7948, so all traffic directed to port 79xx on the *console server* is tunneled thru to port 5900 on the PPP connection on serial Port xx) e.g. for a Windows Viewer PC using UltraVNC connecting to a VNC Server which is attached to Port 1 on a *console server* located 192.168.0.1



- You can then establish the VNC connection by simply activating the VNC Viewer software on the Viewer PC and entering the password

Note For general background reading on Remote Desktop and VNC access we recommend the following:

- *The Microsoft Remote Desktop How-To*
  http://www.microsoft.com/windowsxp/using/mobility/getstarted/remoteintro.mspx
- *The Illustrated Network Remote Desktop* help page
  http://theillustratednetwork.mvps.org/RemoteDesktop/RemoteDesktopSetupandTroubleshooting.html
- *What is Remote Desktop in Windows XP and Windows Server 2003?* by Daniel Petri
  http://www.petri.co.il/what's_remote_desktop.htm
- *Frequently Asked Questions about Remote Desktop*
  http://www.microsoft.com/windowsxp/using/mobility/rdfaq.mspx
- *Secure remote access of a home network using SSH, Remote Desktop and VNC for the home user*
  http://theillustratednetwork.mvps.org/RemoteDesktop/SSH-RDP-VNC/RemoteDesktopVNCandSSH.html
- *Taking your desktop virtual with VNC,* Red Hat magazine / and /
- *Wikipedia* general background on VNC http://en.wikipedia.org/wiki/VNC

6.10 Using SDT to IP connect to hosts that are serially attached to the gateway

Network (IP) protocols like RDP, VNC and HTTP can also be used for connecting to host devices that are serially connected through their COM port to the *console server*. To do this you must:

- establish a PPP connection (Section 6.7.1) between the host and the gateway, then
- set up Secure Tunneling - Ports on the *console server* (Section 6.7.2), then
- configure *SDT Connector* to use the appropriate network protocol to access IP consoles on the host devices that are attached to the *Console server* serial ports (Section 6.7.3)

6.10.1 Establish a PPP connection between the host COM port and console server

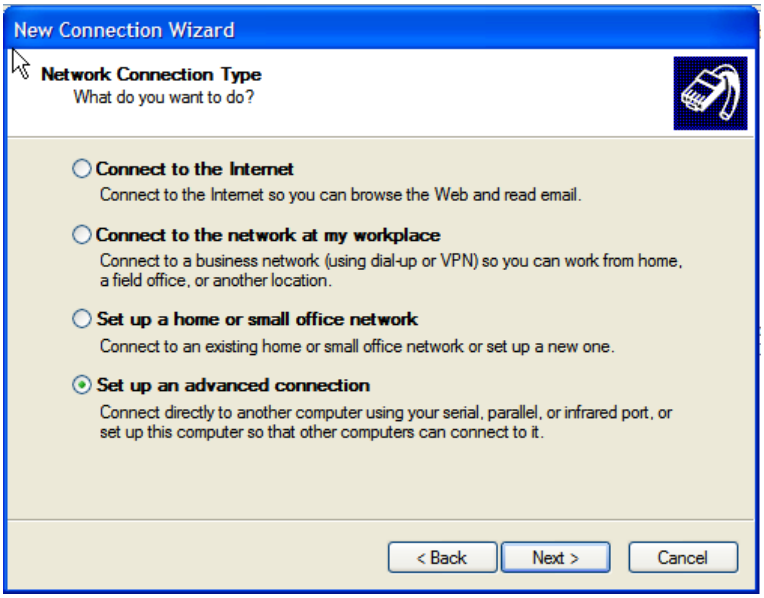
(This step is only necessary for serially connected computers)

Firstly, physically connect the COM port on the host computer that is to be accessed, to the serial port on the *console server* then:

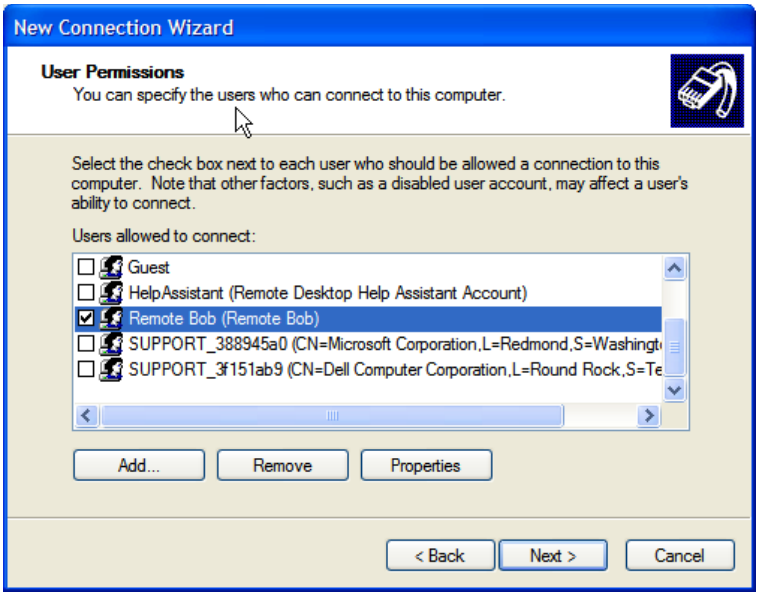
A. For non-Windows (Linux, UNIX, Solaris etc) computers establish a PPP connection over the serial port. The online tutorial http://www.yolinux.com/TUTORIALS/LinuxTutorialPPP.html presents a selection of methods for establishing a PPP connection for Linux

B. For Windows XP and 2003 computers follow the steps below to set up an advanced network connection between the Windows computer, through its COM port to the *console server*. Both Windows 2003 and Windows XP Professional allow you to create a *simple dial in service* which can be used for the Remote Desktop/VNC/HTTP/X connection to the *console server*:

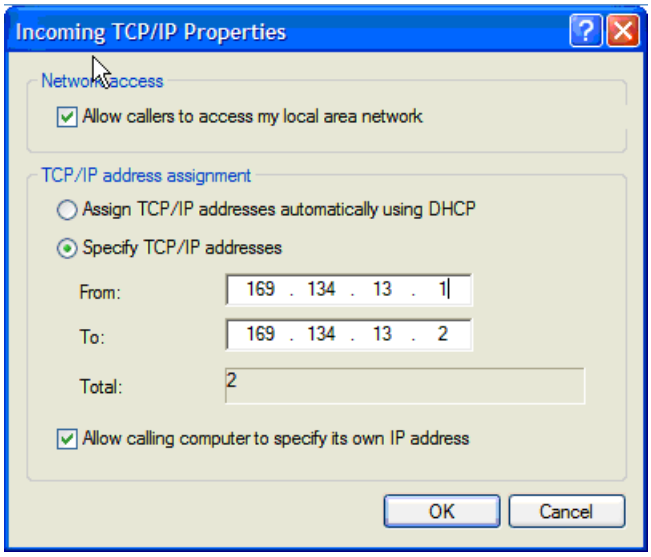
- Open **Network Connections** in Control Panel and click the **New Connection Wizard**



- Select **Set up an advanced connection** and click **Next**
- On the **Advanced Connection Options** screen select **Accept Incoming Connections** and click **Next**
- Select the **Connection Device** (i.e. the serial COM port on the Windows computer that you cabled through to the *console server*). By default select **COM1**. The COM port on the Windows computer should be configured to its maximum baud rate. Click **Next**
- On the **Incoming VPN Connection Options** screen select **Do not allow virtual private connections** and click **Next**



- Specify which *Users* will be allowed to use this connection. This should be the same *Users* who were given Remote Desktop access privileges in the earlier step. Click **Next**
- On the **Network Connection** screen select **TCP/IP** and click **Properties**



- Select **Specify TCP/IP addresses** on the **Incoming TCP/IP Properties** screen select **TCP/IP.** Nominate a *From:* and a *To:* TCP/IP address and click **Next**

Note You can choose any TCP/IP addresses so long as they are addresses which are not used anywhere else on your network. The *From:* address will be assigned to the Windows XP/2003 computer and the *To:* address will be used by the *console server*. For simplicity use the IP address as shown in the illustration above:

From: 169.134.13.1
To: 169.134.13.2

Alternately you can set the advanced connection and access on the Windows computer to use the *console server* defaults:

- Specify 10.233.111.254 as the *From:* address
- Select *Allow calling computer to specify its own address*

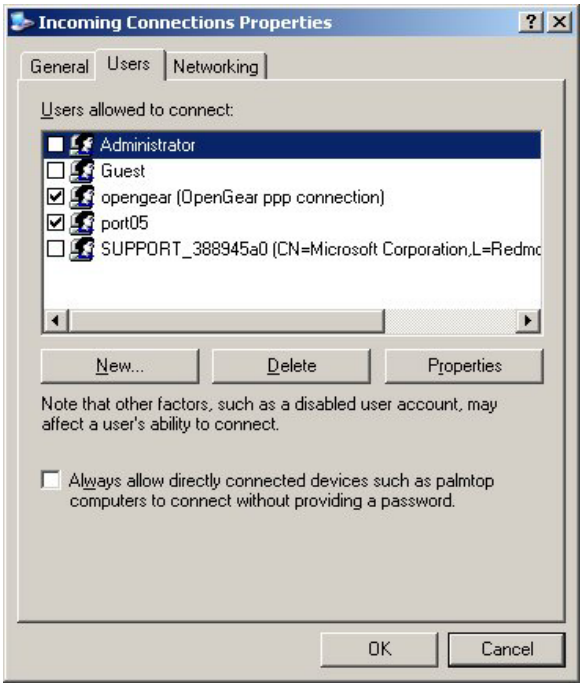
Also you could use the *console server* default username and password when you set up the new Remote Desktop *User* and gave this *User* permission to use the advance connection to access the Windows computer:

- The *console server* default *Username is portXX* where XX is the serial port number on the *console server*.
- The default *Password is portXX*

So to use the defaults for a RDP connection to the serial port 2 on the *console server*, you would have set up a Windows user named port02

➢ When the PPP connection has been set up, a network icon will appear in the Windows task bar

Note The above notes describe setting up an incoming connection for Windows XP. The steps are similar for later versions however the set up screens present slightly differently:



You need to put a check in the box for *Always allow directly connected devices such as palmtop…..*

Also the option for to **Set up an advanced connection** is not available in Windows 2003 if RRAS is configured. If RRAS has been configured it is a simply task to enable the null modem connection for the dial-in configuration.

C. For earlier version Windows computers again follow the steps in Section B. above, however to get to the **Make New Connection** button:

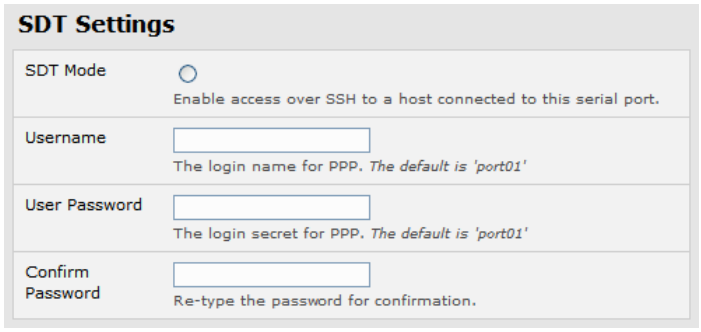
- For Windows 2000, click **Start** and select **Settings** then at the **Dial-Up Networking Folder** click **Network and Dial-up Connections** and click **Make New Connection.** Note you may need to first set up connection over the COM port using **Connect directly to another computer** before proceeding to **Set up an advanced connection**
- For Windows 98 you double click **My Computer** on the Desktop, then open **Dial-Up Networking** and double click

6.10.2 Set up SDT Serial Ports on console server

To set up *RDP (and VNC) forwarding* on the *console server* Serial Port that is connected to the Windows computer COM port:

➢ Select the **Serial & Network: Serial Port** menu option and click **Edit** (for the particular Serial Port that is connected to the Windows computer COM port)

➢ On the SDT Settings menu select **SDT Mode** (which will enable port forwarding and SSH tunneling) and enter a **Username** and **User Password.**



Note When you enable SDT, this will override all other Configuration protocols on that port

Note If you leave the *Username* and *User Password* fields blank, they default to *portXX* and *portXX* where XX is the serial port number. So the default username and password for Secure RDP over Port 2 is *port02*

- Ensure the *console server* **Common Settings** (Baud Rate, Flow Control) are the same as were set up on the Windows computer COM port and click **Apply**
- RDP and VNC forwarding over serial ports is enabled on a Port basis. You can add *Users* who can have access to these ports (or reconfigure *User* profiles) by selecting **Serial & Network :User & Groups** menu tag - as described earlier in Chapter 4 *Configuring Serial Ports*

6.10.3 Set up SDT Connector to ssh port forward over the console server Serial Port

In the *SDT Connector* software running on your remote computer specify the gateway IP address of your *console server* and a username/password for a user you have setup on the *console server* that has access to the desired port.

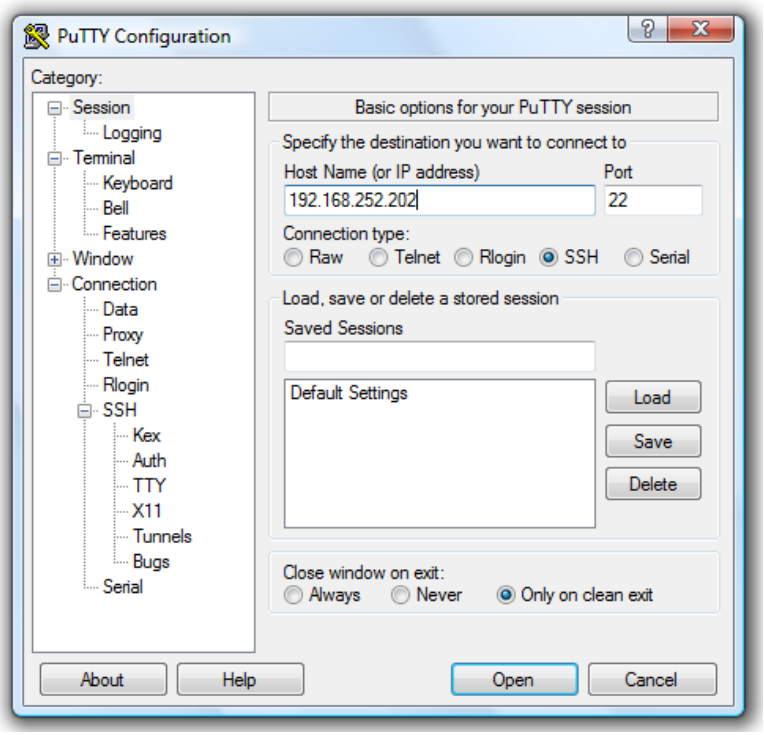
Next you need to add a New SDT Host. In the Host address you need to put portxx where xx = the port you are connecting to. Example for port 3 you would have a Host Address of: port03 and then select the RDP Service check box.

6.11 SSH Tunneling using other SSH clients (e.g. PuTTY)

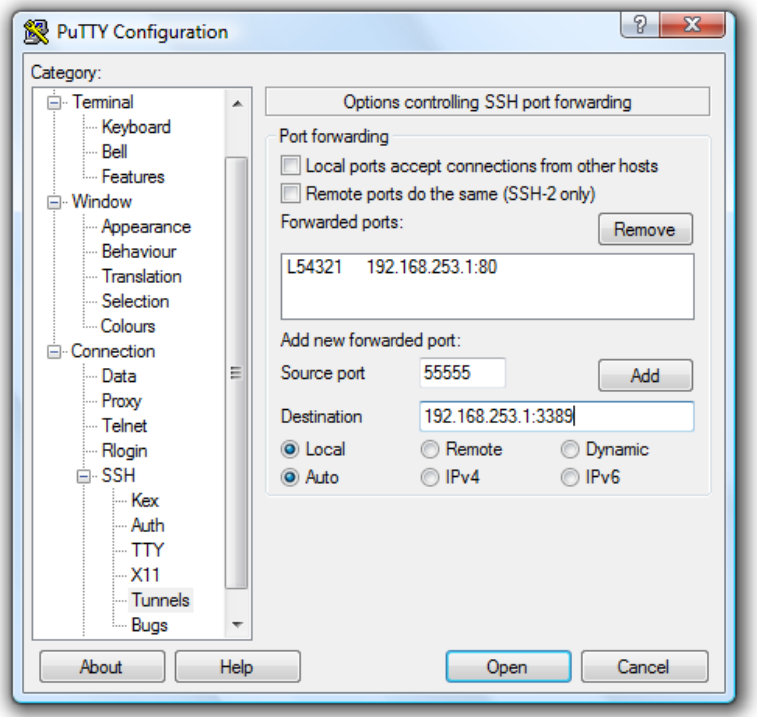
As covered in the previous sections of this chapter we recommend you use the *SDT Connector* client software that is supplied with the *console server*. However there's also a wide selection of commercial and free SSH client programs that can also provide the secure SSH connections to the *console servers* and secure tunnels to connected devices:

- PuTTY is a complete (though not very user friendly:) freeware implementation of SSH for Win32 and UNIX platforms
- SSHTerm is a useful open source SSH communications package
- SSH Tectia is leading end-to-end commercial communications security solution for the enterprise
- Reflection for Secure IT (formerly F-Secure SSH) is another good commercial SSH-based security solution

By way of example the steps below show the establishment of an SSH tunneled connection to a network connected device using the PuTTY client software.



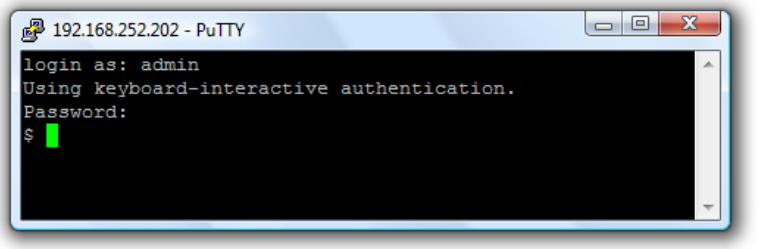
- In the **Session** menu enter the IP address of the *console server* in the **Host Name or IP address** field
  - For dial-in connections, this IP address will be the **Local** Address that you assigned to the *console server* when you set it up as the Dial-In PPP Server
  - For Internet (or local/VPN connections) connections this will be the public IP address of the *console server*
- Select the **SSH Protocol**, and the **Port** will be set as 22
- Go to the **SSH: Tunnels** menu and in *Add new forwarded port* enter any high unused port number for the **Source port** e.g. *54321*
- Set the **Destination:** IP details
  - If your destination device is network connected to the *console server* and you are connecting using RDP, set the Destination as *<Managed Device IP address/DNS Name>:3389* e.g. if when setting up the Managed Device as *Network Host* on the *console server* you specified its IP address to be 192.168.253.1 (or its DNS Name was *accounts.myco.intranet.com*) then specify the Destination as *192.168.523.1:3389* (or *accounts.myco.intranet.com:3389).* Only devices which have been configured as networked Hosts can be accessed using SSH tunneling (except by the “root” user who can tunnel to any IP address the *console server* can route to.



- If your destination computer is serially connected to the *console server*, set the *Destination* as *<port label>:3389* e.g. if the **Label** you specified on the serial port on the *console server* is *win2k3*, then specify the remote host as *win2k3:3389* . Alternative you can set the *Destination* as *portXX:3389* where XX is the SDT enabled serial port number e.g. if port 4 is on the *console server* is to carry the RDP traffic then specify *port04:3389*

Note http://www.jfitz.com/tips/putty_config.html has useful examples on configuring PuTTY for SSH tunneling

- Select **Local** and click the **Add** button
- Click **Open** to SSH connect the Client PC to the *console server*. You will now be prompted for the Username/Password for the *console server* user



- If you are connecting as a *User* in the “users” group then you can only SSH tunnel to Hosts and Serial Ports where you have specific access permissions
- If you are connecting as an *Administrator* (in the “admin” group) then you can connect to any configured Host or Serial Ports (which has SDT enabled)

To set up the secure SSH tunnel for a HTTP browser connection to the Managed Device specify port 80 (rather than port 3389 as was used for RDP) in the Destination IP address.

To set up the secure SSH tunnel from the Client (Viewer) PC to the *console server* for VNC follow the steps above, however when configuring the VNC port redirection specify port 5900 in the Destination IP address.

Note How secure is VNC? VNC access generally allows access to your whole computer, so security is very important. VNC uses a random challenge-response system to provide the basic authentication that allows you to connect to a VNC server. This is reasonably secure and the password is not sent over the network.

However, once connected, all subsequent VNC traffic is unencrypted. So a malicious user could snoop your VNC session. Also there are VNC scanning programs available, which will scan a subnet looking for PCs which are listening on one of the ports which VNC uses.

Tunneling VNC over a SSH connection ensures all traffic is strongly encrypted. Also no VNC port is ever open to the internet, so anyone scanning for open VNC ports will not be able to find your computers. When tunneling VNC over a SSH connection, the only port which you're opening on your *console server* the SDT port 22.

So sometimes it may be prudent to tunnel VNC through SSH even when the Viewer PC and the *console server* are both on the same local network.

ALERTS, AUTO-RESPONSE AND LOGGING

This chapter describes the automated response, alert generation and logging features of the *console server*.

The Auto-Response facility extends on the basic Alert facility available in earlier (pre V3.5) firmware revisions. With Auto-Response the *console server* monitors selected serial ports, logins, the power status and environmental monitors and probes for Check Condition triggers. The *console server* will then initiate a sequence of actions in response to these triggers. To configure Auto-Response you:

- set the general parameters (*Section 7.1*), then
- select and configure the *Check Conditions* (*Section 7.2*) i.e. the conditions that will trigger the response, then
- specify the *Trigger Actions* (*Section 7.3*) i.e. sequence of actions initiated in event of the trigger condition, then
- specify the *Resolve Actions* (*Section 7.4*) i.e. actions performed when trigger conditions have been resolved.

Also all *console server* models can maintain log records of all access and communications with the *console server* and with the attached serial devices. A log of all system activity is also maintained as is a history of the status of any attached environmental monitors.

Some models can also log access and communications with network attached hosts and maintain a history of the UPS and PDU power status.

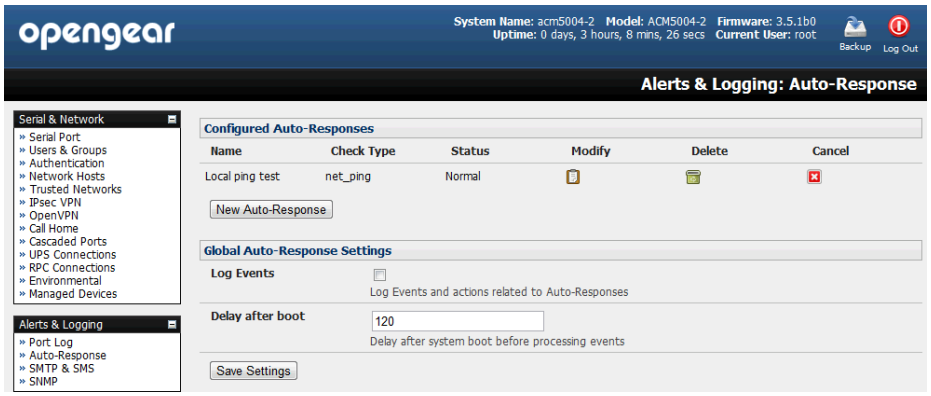
- If port logs are to be maintained on a remote server, then the access path to this location need to be configured Then you need to activate and set the desired levels of logging for each serial and/or network port *(Section 7.6)* and/or power and environment UPS (refer *Chapter 8*)

7.1 Configure Auto-Response

With the Auto-Response facility, a sequence of *Trigger Actions* is initiated in the event of a specified trigger condition *(Check Condition)*. Subsequent *Resolve Actions* can also be performed when the trigger condition has been resolved.

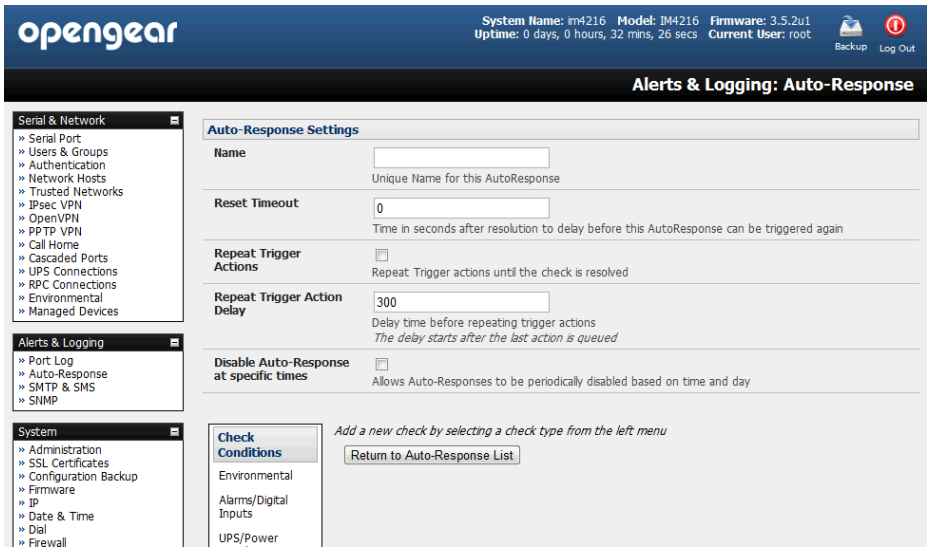
To configure, first set the general parameters that will be applied to all Auto-Responses:

- Check **Log Events** on **Alerts & Logging: Auto-Response** to enable logging all Auto-Response activities
- Check **Delay after Boot** to set any general delay to be applied after console server system boot, before processing events

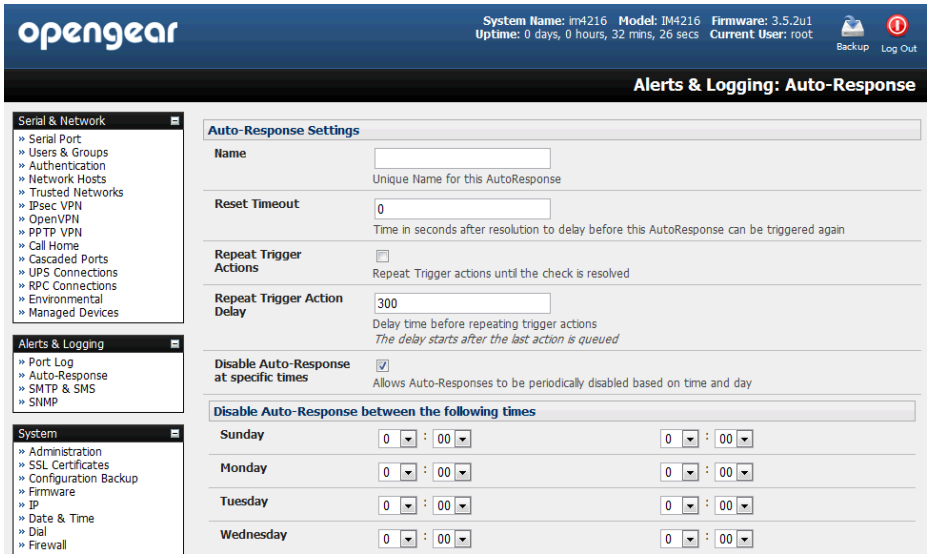


To configure a new Auto-Response:

- Select **New Auto-Response** in the **Configured Auto-Response** field. You will be presented with a new **Auto-Response Settings** menu
- Enter a unique **Name** for the new Auto-Response
- Specify the **Reset Timeout** for the time in seconds after resolution to delay before this Auto-Response can be triggered again
- Check **Repeat Trigger Actions** to continue to repeat trigger action sequences until the check is resolved
- Enter any required delay time before repeating trigger actions in **Repeat Trigger Action Delay**. This delay starts after the last action is queued



- Check **Disable Auto-Response at specific times** and you will be able to periodically disable auto-Responses between specified times of day



7.2 Check Conditions

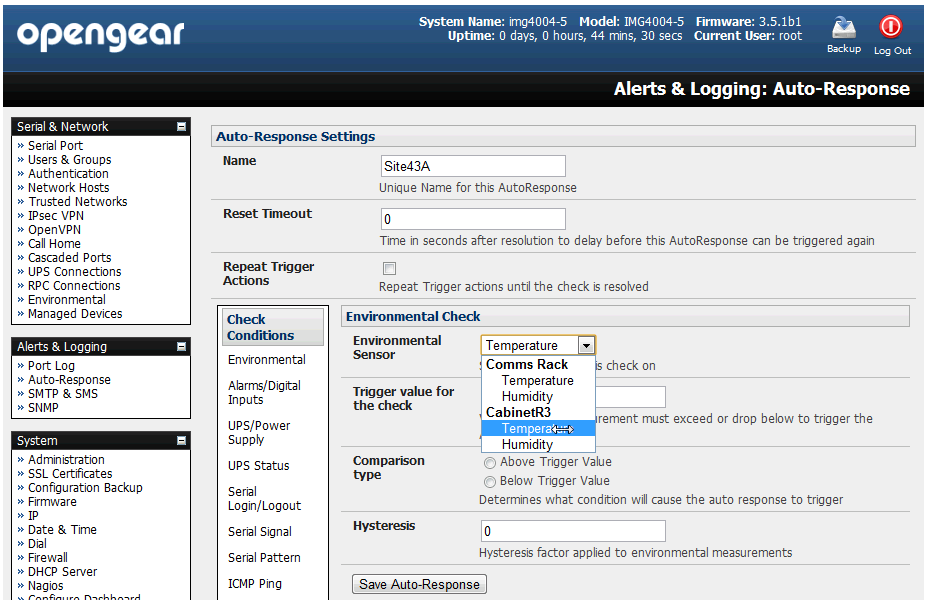
To configure the condition that will trigger the Auto-Response:

- Click on the **Check Condition** type (e.g. *Environmental*, *UPS Status* or *ICMP ping*) to be configured as the trigger for this new Auto-Response in the **Auto-Response Settings** menu

7.2.1 Environmental

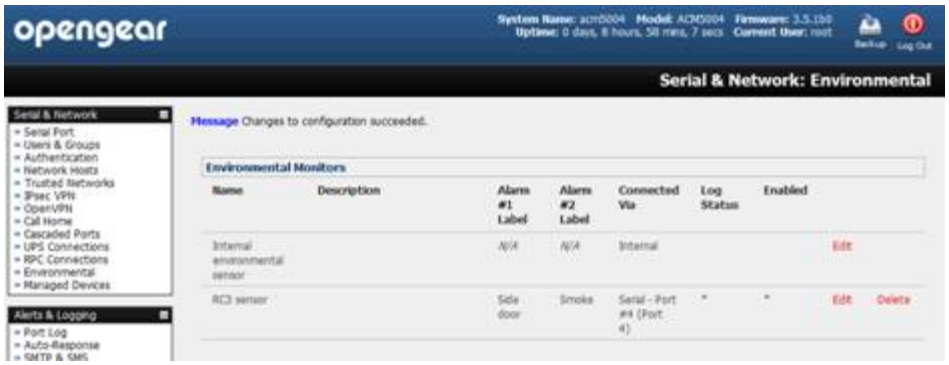
To configure Humidity or Temperature levels as the trigger event:

- Click on the **Environmental** as the **Check Condition**



- In the **Environmental Check** menu, select the specific **Environmental Sensor** to be checked for the trigger
- Specify the **Trigger value** (in °C / °F for Temp and % for Humidity) that the check measurement must exceed or drop below to trigger the AutoResponse
- Select **Comparison type** as being Above Trigger Value or Below Trigger Value to trigger
- Specify any **Hysteresis** factor that is to be applied to environmental measurements (e.g. if an Auto-Response was set up with a trigger event of a temp reading above 49°C with a Hysteresis of 4 then the trigger condition would not be seen as having been resolved till the temp reading was below 45°C)
- Check **Save Auto-Response**

Note: Before configuring Environmental Checks as the trigger in Auto-Response you will need first to configure the Temp and/or Humidity sensors on your ACM5000 or attached EMD



7.2.2 Alarms and Digital Inputs

To set the status of any attached Smoke or Water sensors or digital inputs as the trigger event:

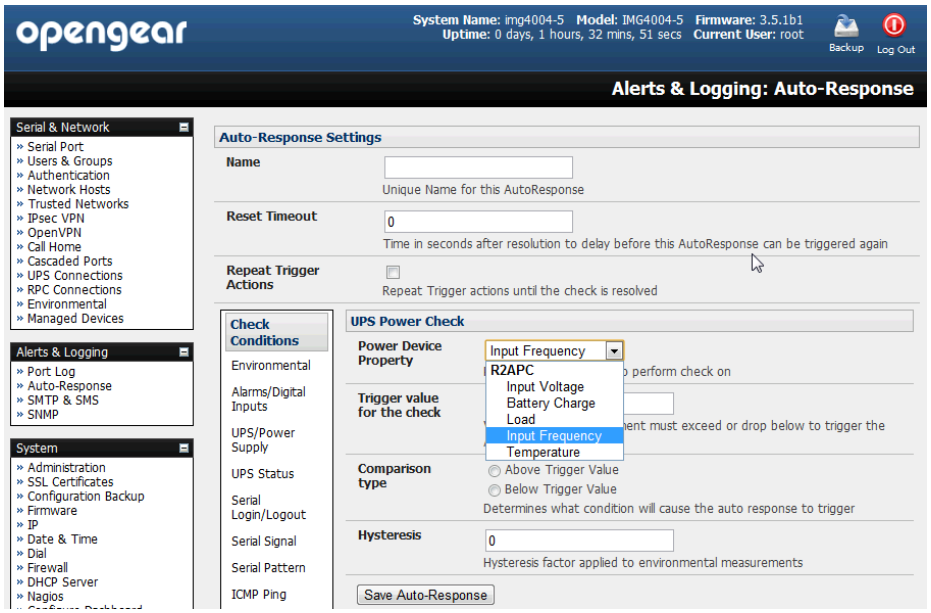
- Click on **Alarms / Digital Inputs** as the **Check Condition**
- In the **Alarms / Digital Inputs Check** menu, select the specific **Alarm/Digital IO Pin** that will trigger the Auto-Response
- Select **Trigger on Change** to trigger when alarm signal changes, or select to trigger when the alarm signal state changes to either a **Trigger Value** of *Open (0)* or *Closed (1)*
- Check **Save Auto-Response**

Note: Before configuring Alarms / Digital Inputs checks in Auto-Response you first must configure the sensor/DIO that is to be attached to your EMD or ACM5000

7.2.3 UPS / Power Supply

To use the properties of any attached UPS as the trigger event:

- Click on **UPS / Power Supply** as the **Check Condition**
- Select UPS **Power Device Property** (Input Voltage, Battery Charge %, Load %, Input Frequency Hz or Temperature in °C) that will checked for the trigger
- Specify the **Trigger value** that the check measurement must exceed or drop below to trigger the AutoResponse
- Select **Comparison type** as being Above Trigger Value or Below Trigger Value to trigger
- Specify any **Hysteresis** factor that is to be applied to environmental measurements (e.g. if an Auto-Response was set up with a trigger event of a battery charge below 20% with a Hysteresis of 5 then the trigger condition would not be seen as having been resolved till the battery charge was above 25%)
- Check **Save Auto-Response**



Note: Before configuring UPS checks in Auto-Response you first must configure the attached UPS

7.2.4 UPS Status

To use the alert state of any attached UPS as the Auto-Response trigger event:

- Click on **UPS Status** as the **Check Condition**
- Select the reported **UPS State** to trigger the Auto-Response (either *On Battery* or *Low Battery*)**.** The Auto-Response will resolve when the UPS state returns to the "Online" state
- Select which connected **UPS Device** to monitor and check **Save Auto-Response**

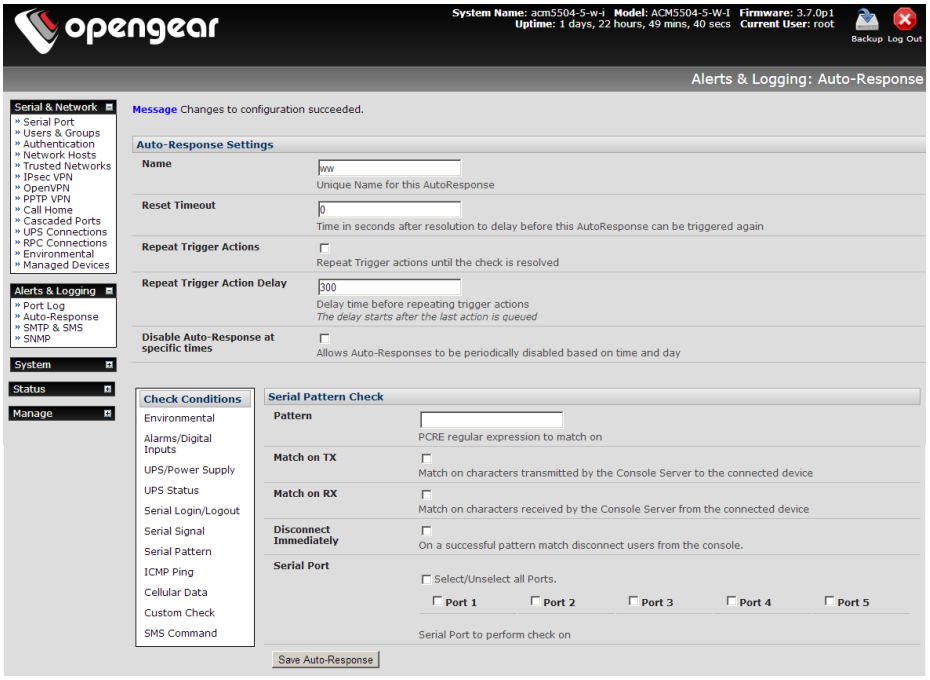
Note: Before configuring UPS state checks in Auto-Response you first must configure the attached UPS

7.2.5 Serial Login, Signal or Pattern

To monitor serial ports and check for login/logout or pattern matches for Auto-Response triggers events:

- Click on **Serial Login/Logout** as the **Check Condition**. Then in the **Serial Login/Logout Check** menu select **Trigger on Login** (to trigger when any user logs into the serial port) or **Trigger on Logout** and specify **Serial Port** to perform check on, and/or
- Click on **Serial Signal** as the **Check Condition**. Then in the **Serial Signal Check** menu select the **Signal** (CTS, DCD, DSR) to trigger on, the **Trigger** condition (either on serial signal change, or check level) and specify **Serial Port** to perform check on, and/or
- Click on **Serial Pattern** as the **Check Condition**. Then in the **Serial Pattern Check** menu select the **PCRE** pattern to trigger on and the serial line (**TX** or **RX**) and **Serial Port** to pattern check on

Note: With Serial Pattern checks you can nominate to “Disconnect Immediately” all users from the serial being monitored in event of a successful pattern match.



Note: For devices with an inbuilt cellular modem with GPS enabled the GPS will be displayed as an additional port and it can be monitored for trigger events eg with an ACM5504-5-G-I with 4 x serial ports the GPS will be shown as Port 5

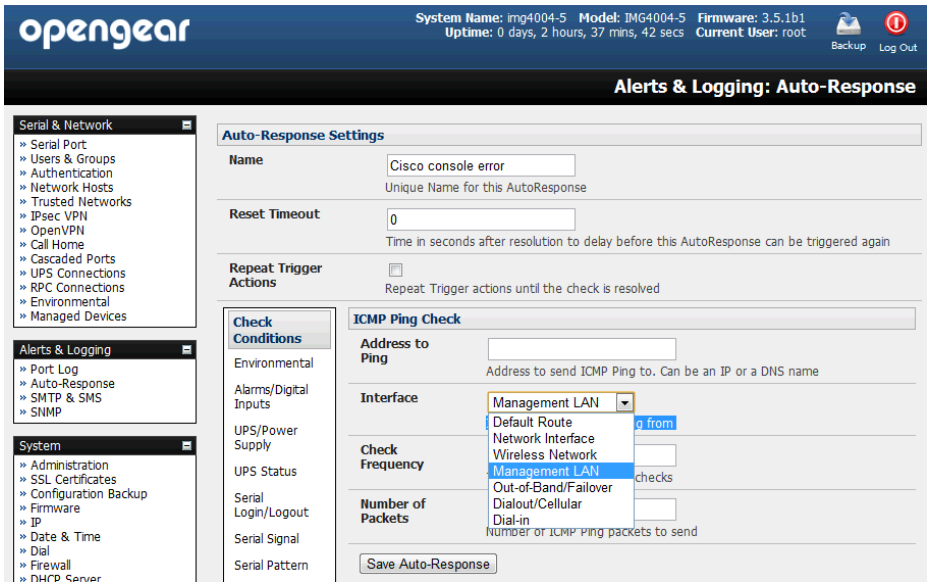
- Check **Save Auto-Response**

Note: Before configuring serial port checks in Auto-Response you first must configure the serial port in Console server mode. Also most serial port checks are not resolvable so resolve actions will not be run

7.2.6 ICMP Ping

To use a *ping* result as the Auto-Response trigger event:

- Click on **ICMP Ping** as the **Check Condition**
- Specify which **Address to Ping** (i.e. IP address or DNS name to send ICMP Ping to) and which **Interface** to send ICMP Ping from (e.g. Management LAN or Wireless network)
- Set the **Check Frequency** (i.e. the time in seconds between checks) and the **Number** of ICMP Ping packets to send
- Check **Save Auto-Response**



7.2.7 Cellular Data

This check monitors the aggregate data traffic inbound and outbound through the cellular modem as an Auto-Response trigger event.

- Click on **Cellular Data** as the **Check Condition**

Note: Before configuring cellular data checks in Auto-Response the internal cellular modem must be configured and detected by the *console server*

7.2.8 Custom Check

This check allows users to run a nominated custom script with nominated arguments whose return value is used as an Auto-Response trigger event:

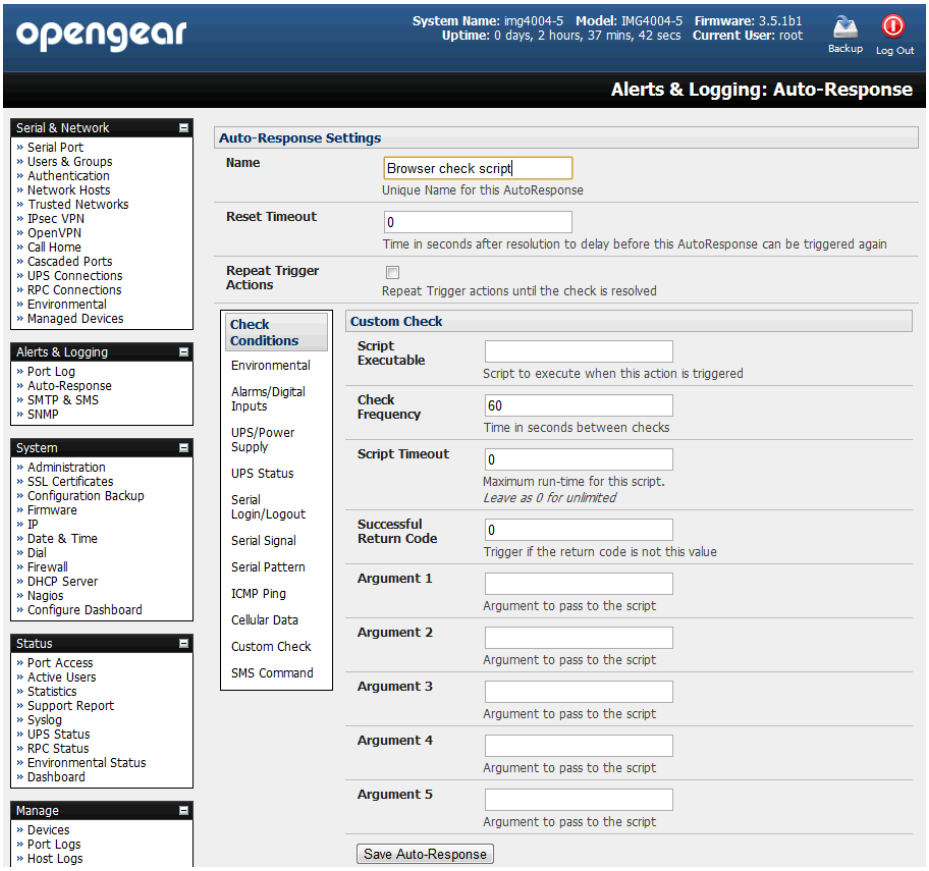
- Click on **Custom Check** as the **Check Condition**
- Create an executable trigger check script file e.g. */etc/config/test.sh*

```
#!/bin/sh
logger "A test script"
logger Argument1 = $1
logger Argument2 = $2
logger Argument3 = $3
logger Argument4 = $4
if [ -f /etc/config/customscript.0 ]; then
      rm /etc/config/customscript.0
      exit 7
fi
touch /etc/config/customscript.0
exit 1
```

Refer online FAQ for a sample web page html check and other script file templates

- Enter the **Script Executable** file name (e.g. */etc/config/test.sh*)
- Set the **Check Frequency** (i.e. the time in seconds between re-running the script) and the **Script Timeout** (i.e. the maximum run-time for the script)
- Specify the **Successful Return Code**. An Auto-Response is triggered if the return code from the script is not this value

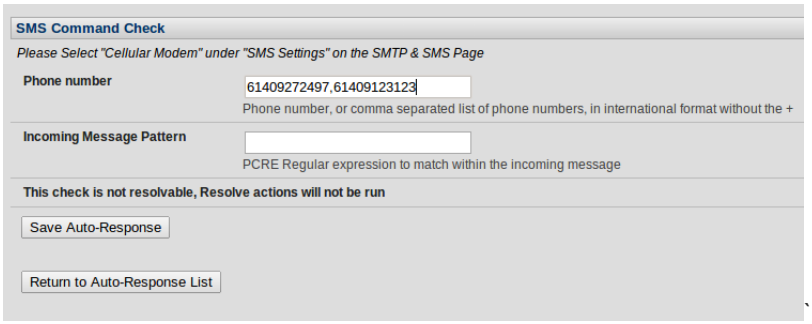
- Enter **Arguments** that are to be passed to the script (e.g. with a web page html check script, these Arguments might specify the web page address/DNS and user logins)
- Check **Save Auto-Response**



7.2.9 SMS Command

An incoming SMS command from a nominated caller can trigger an Auto-Response:

- Click on **SMS Command** as the **Check Condition**
- Specify which **Phone Number** (in international format) of the phone sending the SMS message. For multiple trusted SMS sources separate the numbers with a comma
- Set the **Incoming Message Pattern** (PCRE regular expression) to match to create trigger event

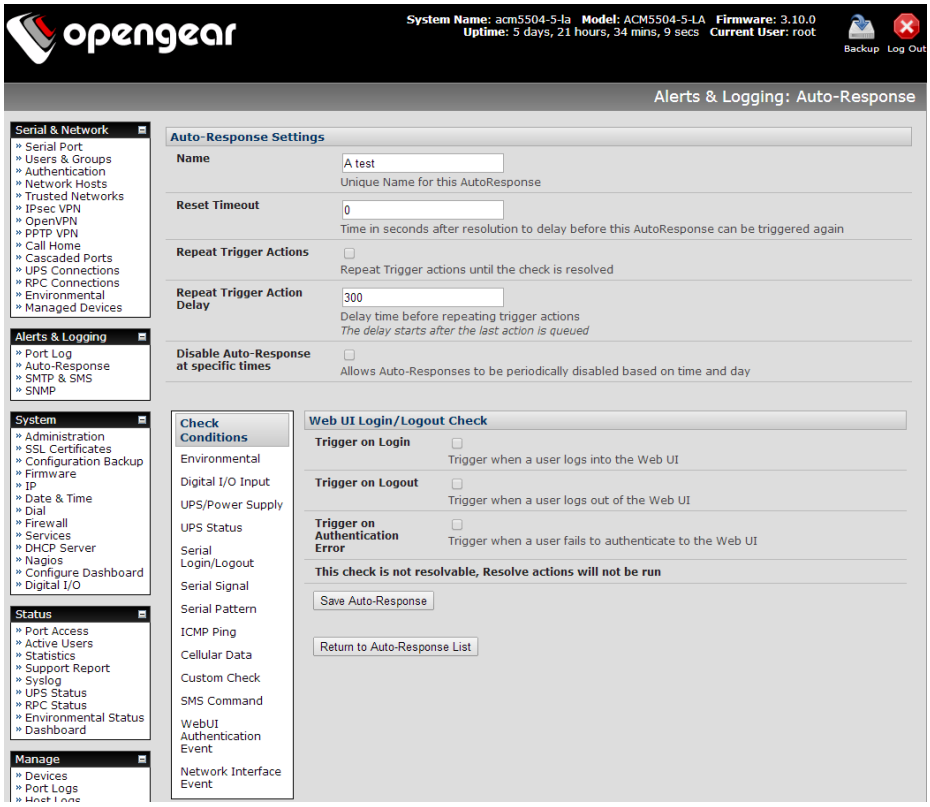


Note: The SMS command trigger condition can only be set if there is an internal cellular modem detected

7.2.10 Log In/Out Check

To configure Web Log In/Out as the trigger event:

- Click on the **Web UI Authentication** as the **Check Condition**



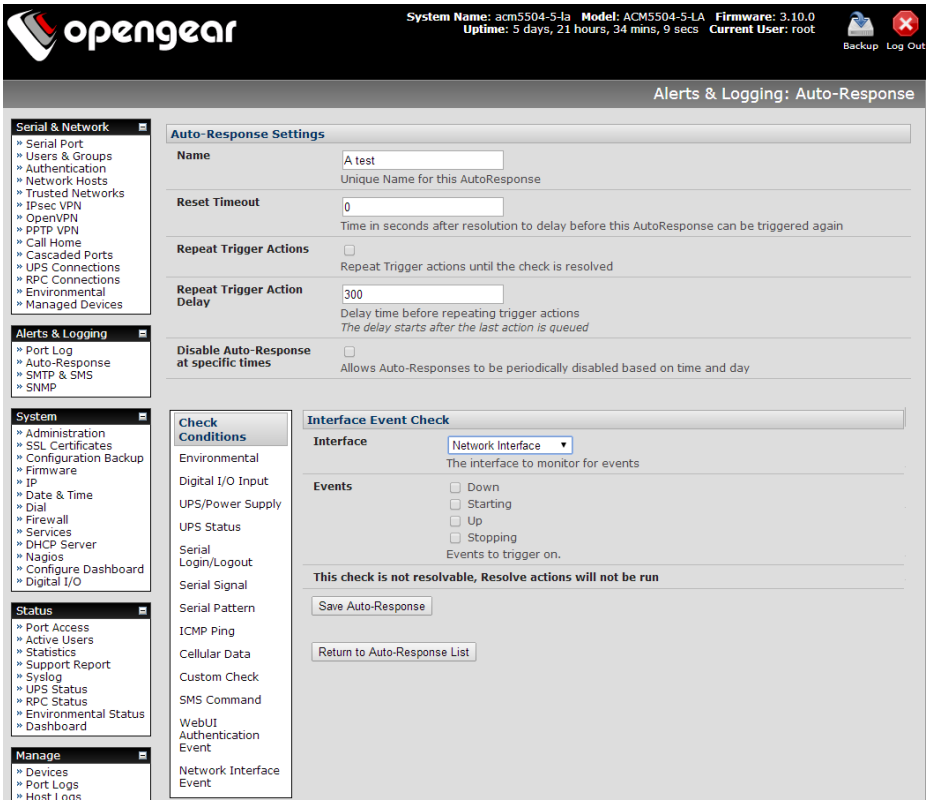
- Check **Trigger on Login (Logout)** to trigger when a user logs into (or out of) the Web UI
- Check **Trigger on Authentication Error** to trigger when a user fails to authenticate to the Web UI

Note: This check is not resolvable so Resolve actions will not be run

7.2.11 Network Interface Event

You may wish to configure a change in the network status as the trigger event (e.g. to send an alert or restart a VPN tunnel connection):

- Click on **Network Interface** as the **Check Condition**



- Select the **Interface** (Ethernet /Failover OOB Interface or Modem or VPN) to monitor
- Check what type of network interface **Event** to trigger on (interface Down, Starting, Up or Stopping)

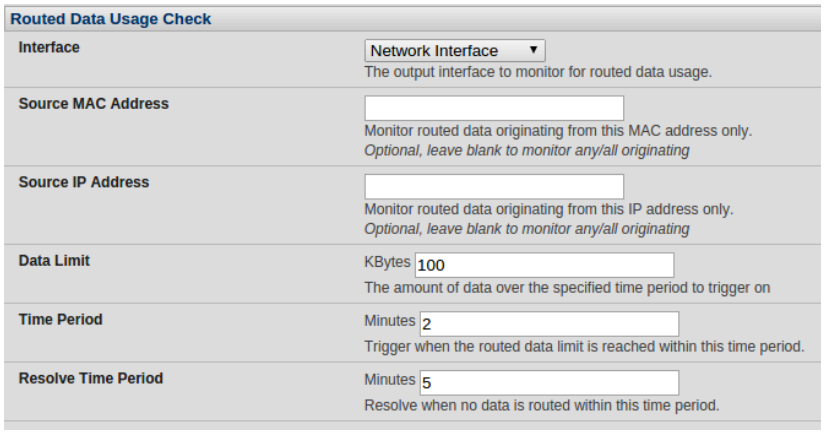
Note: This check is not resolvable so Resolve actions will not be run.

7.2.12 Routed Data Usage Check

This check monitors the specified input interface for data usage that is being routed through the Opengear and out another interface such as the Internal Cellular Modem.

It is particularly useful in IP Passthrough mode, to detect when the downstream router has failed over and is now routing via the Opengear's modem as a backup connection.

This check may be configured with these parameters:



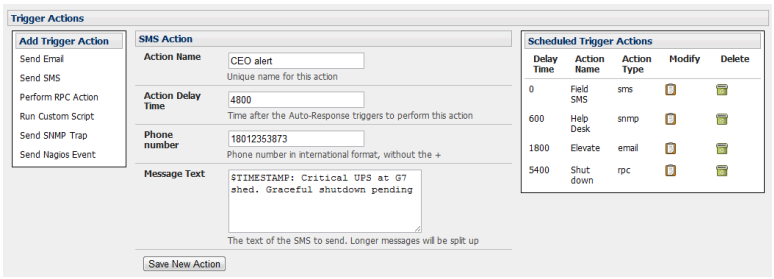
- The Opengear's incoming **Interface** to monitor
- An optional **Source MAC/IP Address**, to monitor traffic from a specific host (e.g. the downstream router)

- A **Data Limit** threshold, the Auto-Response will trigger when this is hit in the specified **Time Period**
- The Auto-Response will resolve if no matching data is routed for the **Resolve Period**

7.3 Trigger Actions

To configure the sequence of actions that is to be taken in the event of the trigger condition:

- For a nominated Auto-Response - with a defined Check Condition - click on **Add Trigger Action** (e.g. *Send Email* or *Run Custom Script)* to select the action type to be taken. Then configure the selected action (as detailed in the following sections)
- Each action is configured with a nominated **Action Delay Time** which specifies how long (in seconds) after the Auto-Response trigger event to wait before performing the action. So you can add follow-on actions to create a sequence of actions that will be taken in the event of the one trigger condition
- To edit (or delete) an existing action, click the **Modify** (or **Delete**) icon in the **Scheduled Trigger Action** table



Note: A message text can be sent with Email, SMS and Nagios actions. This configurable message can include selected values:

$AR_TRIGGER_VAL = the trigger value for the check e.g. for UPS Status, it could be *onbatt* or *battlow*
$AR_VAL = the value returned by the check e.g. for ups status, it could be *online/onbatt/battlow*
$AR_CHECK_DEV = the device name of the device being checked e.g. for Alarm, the alarm name
$TIMESTAMP = the current timestamp
$HOSTNAME = the hostname of the *console server*

The default message text is: *$TIMESTAMP: This action was run - Check details: value $AR_VAL vs trigger value $AR_TRIGGER_VAL*

7.3.1 Send Email

- Click on **Send Email** as the **Add Trigger Action**. Enter a unique **Action Name** and set the **Action Delay Time**
- Specify the **Recipient Email Address** to send this email to and the **Subject** of the email. For multiple recipients you can enter comma separated addresses
- Edit the **Email Text** message to send and click **Save New Action**

Note An SMS alert can also be sent via an SMTP (email) gateway. You will need to specify the **Recipient Email Address** in the format specified by the gateway provider (e.g. for T-Mobile it is *phonenumber* @tmomail.net)

7.3.2 Send SMS

- Click on **Send SMS** as the **Add Trigger Action**. Enter a unique **Action Name** and set the **Action Delay Time**
- Specify the **Phone number** that the SMS will be sent to in international format (without the +)

- Edit the **Message Text** to send and click **Save New Action**

Note: The SMS alert can only be sent if there is an internal cellular modem attached. However an SMS alert can also be sent via a SMTP SMS gateway as described above.

7.3.3 Perform RPC Action

- Click on **Perform RPC Action** as the **Add Trigger Action**. Enter a unique **Action Name** and set the **Action Delay Time**
- Select a power **Outlet** and specify the **Action** to be performed (power On, OFF or Cycle)
- Click **Save New Action**

7.3.4 Run Custom Script

- Click on **Run Custom Script** as the **Add Trigger Action**. Enter a unique **Action Name** and set the **Action Delay Time**
- Create a script file to execute when this action is triggered and enter the **Script Executable** file name e.g. */etc/config/action.sh*
- Set the **Script Timeout** (i.e. the maximum run-time for the script). Leave as 0 for unlimited.
- Enter any **Arguments** that are to be passed to the script and click **Save New Action**

7.3.5 Send SNMP Trap

- Click on **Send SNMP Trap** as the **Add Trigger Action**. Enter a unique **Action Name** and set the **Action Delay Time**

Note: The SNMP Trap actions are valid for Serial, Environmental, UPS and Cellular data triggers

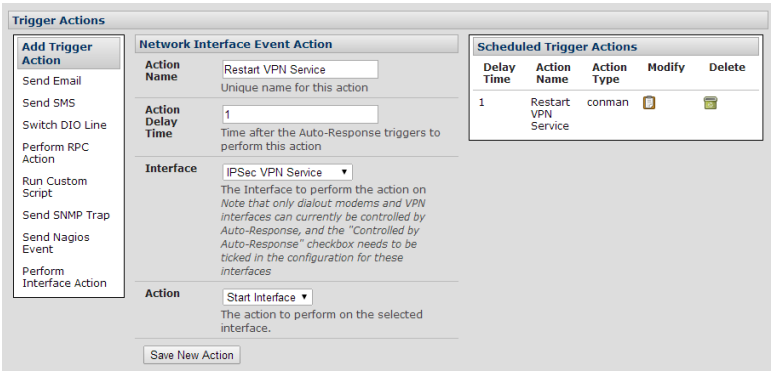
7.3.6 Send Nagios Event

- Click on **Send Nagios Event** as the **Add Trigger Action**. Enter a unique **Action Name** and set the **Action Delay Time**
- Edit the **Nagios Event Message** text to display on the Nagios status screen for the service
- Specify the **Nagios Event State** (*OK, Warning, Critical* or *Unknown*) to return to Nagios for this service
- Click **Save New Action**

Note: To notify the central Nagios server of Alerts, NSCA must be enabled under System: Nagios and Nagios must be enabled for each applicable host or port

7.3.7 Perform Interface Action

- Click on **Perform Interface Action** as the **Add Trigger Action**. Enter a unique **Action Name** and set the **Action Delay Time**
- **Select the Interface** (Modem or VPN service) and the **Action** (Start or Stop Interface) to be taken e.g. you may wish to start an IPsec VPN service in response to an incoming SMS - or set up an OpenVPN tunnel whenever your Opengear device fails over to use the cellular connection.



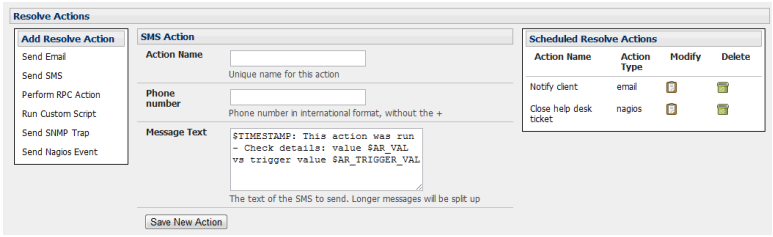
Note: If any IPsec service or OpenVPN tunnel is to be controlled by the Network Interface Event Action, you will need to have checked the *Control by Auto-Response* box when configuring that service - and, if selected, the default state for the VPN tunnel / service will be Down

7.4 Resolve Actions

Actions can also be scheduled to be taken a trigger condition has been resolved:

- For a nominated Auto-Response - with a defined trigger Check Condition - click on **Add Resolve Action** (e.g. *Send Email* or *Run Custom Script)* to select the action type to be taken

Note: Resolve Actions are configured exactly the same as Trigger Actions except the designated Resolve Actions are all executed on resolution of the trigger condition and there are no Action Delay Times set



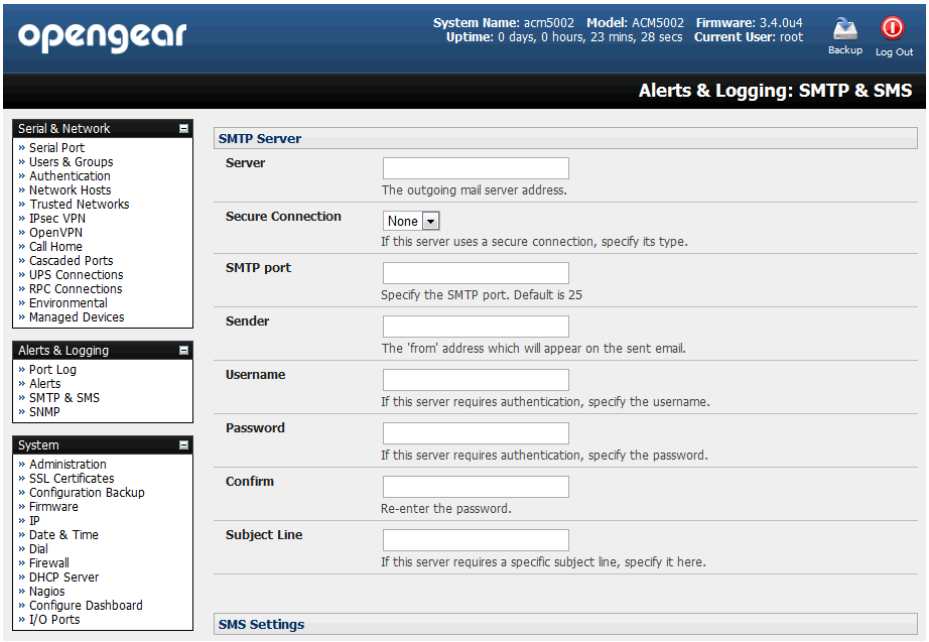
7.5 Configure SMTP, SMS, SNMP and/or Nagios service for alert notifications

The Auto-Response facility enables remote alerts to be sent as Trigger and Resolve Actions. Before such alert notifications can be sent, you must configure the nominated alert service.

7.5.1 Send Email alerts

The *console server* uses SMTP (Simple Mail Transfer Protocol) for sending the email alert notifications. To use SMTP, the *Administrator* must configure a valid SMTP server for sending the email:

- Select **Alerts & Logging: SMTP &SMS**



- In the **SMTP Server** field enter the IP address of the outgoing mail **Server**
- If this mail server uses a **Secure Connection**, specify its type. You may also specify the IP port to use for SMTP. The default **SMTP Port** is 25.
- You may enter a **Sender** email address which will appear as the "*from"* address in all email notifications sent from this *console server*. Many SMTP servers check the sender's email address with the host domain name to verify the address as authentic. So it may be useful to assign an email address for the console server such as consoleserver2@mydomian.com
- You may also enter a **Username** and **Password** if the SMTP server requires authentication
- Similarly can specify the specific **Subject Line** that will be sent with the email
- Click **Apply** to activate SMTP

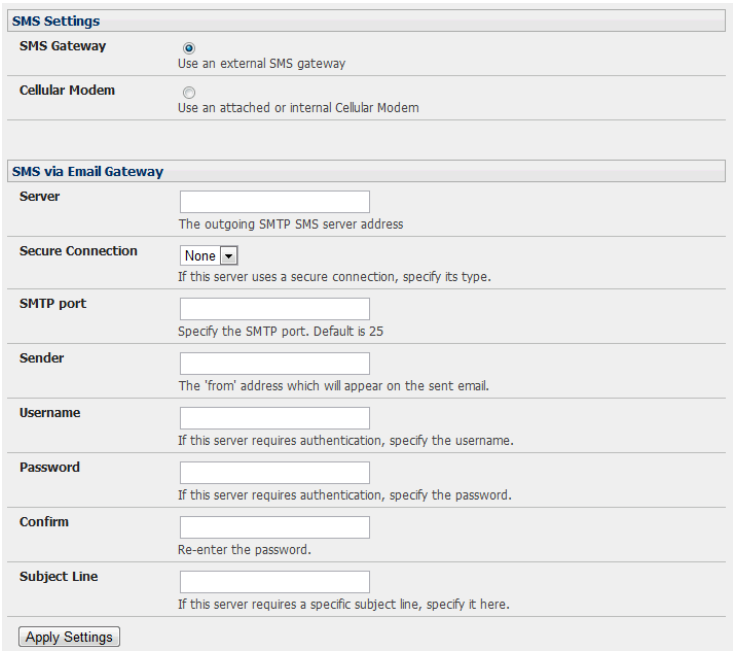
7.5.2 Send SMS alerts

With any model *console server* you can use email-to-SMS services to send SMS alert notifications to mobile devices. Almost all mobile phone carriers provide an SMS gateway service that forwards email to mobile phones on their networks. There's also a wide selection of SMS gateway aggregators who provide email to SMS forwarding to phones on any carriers.

Alternately if your *console server* has an embedded or externally attached cellular modem you will be given the option to send the SMS directly over the carrier connection.

SMS via Email Gateway

To use SMTP SMS the *Administrator* must configure a valid SMTP server for sending the email:



- In the **SMTP Settings** field in the **Alerts & Logging: SMTP &SMS** menu select **SMS Gateway**. An **SMS via Email Gateway** field will appear
- Enter the IP address of the outgoing mail **Server** SMS gateway
- Select a **Secure Connection** (if applicable) and specify the **SMTP port** to be used (if other than the default port 25)
- You may also enter a **Sender** email address which will appear as the "*from"* address in all email notifications sent from this *console server*. Some SMS gateway service providers only forward email to SMS when the email has been received from authorized senders. So you may need to assign a specific authorized email address for the *console server*
- You may also enter a **Username** and **Password** as some SMS gateway service providers use SMTP servers which require authentication
- Similarly you can specify the specific **Subject Line** that will be sent with the email. Generally the email subject will contain a truncated version of the alert notification message (which is contained in full in the body of the email). However some SMS gateway service providers require blank subjects or require specific authentication headers to be included in the subject line
- Click **Apply Settings** to activate SMS-SMTP connection.

SMS via Cellular Modem

To use an attached or internal cellular modem for SMS the *Administrator* must enable SMS:

- Select **Cellular Modem** In the **SMS Settings** field

- Check **Receive Messages** to enable incoming SMS messages to be received. A custom script will be called on receipt of incoming SMS messages
- You may need to enter the phone number of the carrier's **SMS Message Centre** (only if advised by your carrier or Support)
- Click **Apply Settings** to activate SMS-SMTP connection

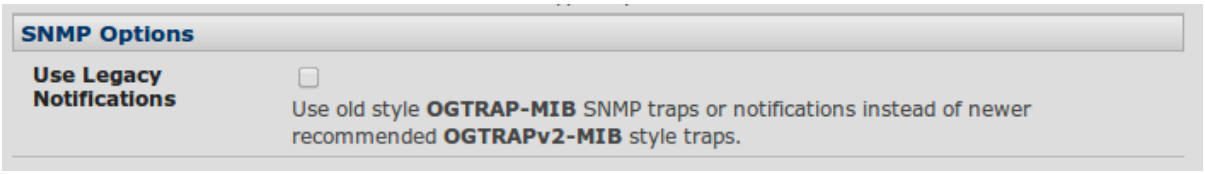
Note The option to directly send SMS alerts via the cellular modem was included in the Management GUI in V3.4. Advanced *console servers* have had the gateway software (*SMS Server Tools 3)* embedded since V3.1 however you this could only be accessed from the command line to send SMS messages (refer online FAQ).

7.5.3 Send SNMP Trap alerts

The *Administrator* can configure the Simple Network Management Protocol (SNMP) agent that resides on the *console server* to send SNMP trap alerts to an NMS management application:

- Select **Alerts & Logging: SNMP**

Note In firmware versions 3.10.2 and above, new SNMP status and trap MIBS were created to provide more and better structured SNMP status and traps from *console servers*. There is an option in the SNMP menu to **Use Legacy Notifications** for the SNMP traps. Setting this option means the console server will send SNMP traps that are compatible with those sent in older firmware before the new MIBS were added ensuring that the firmware upgrade won't upset existing SNMP management a user will already have in place.

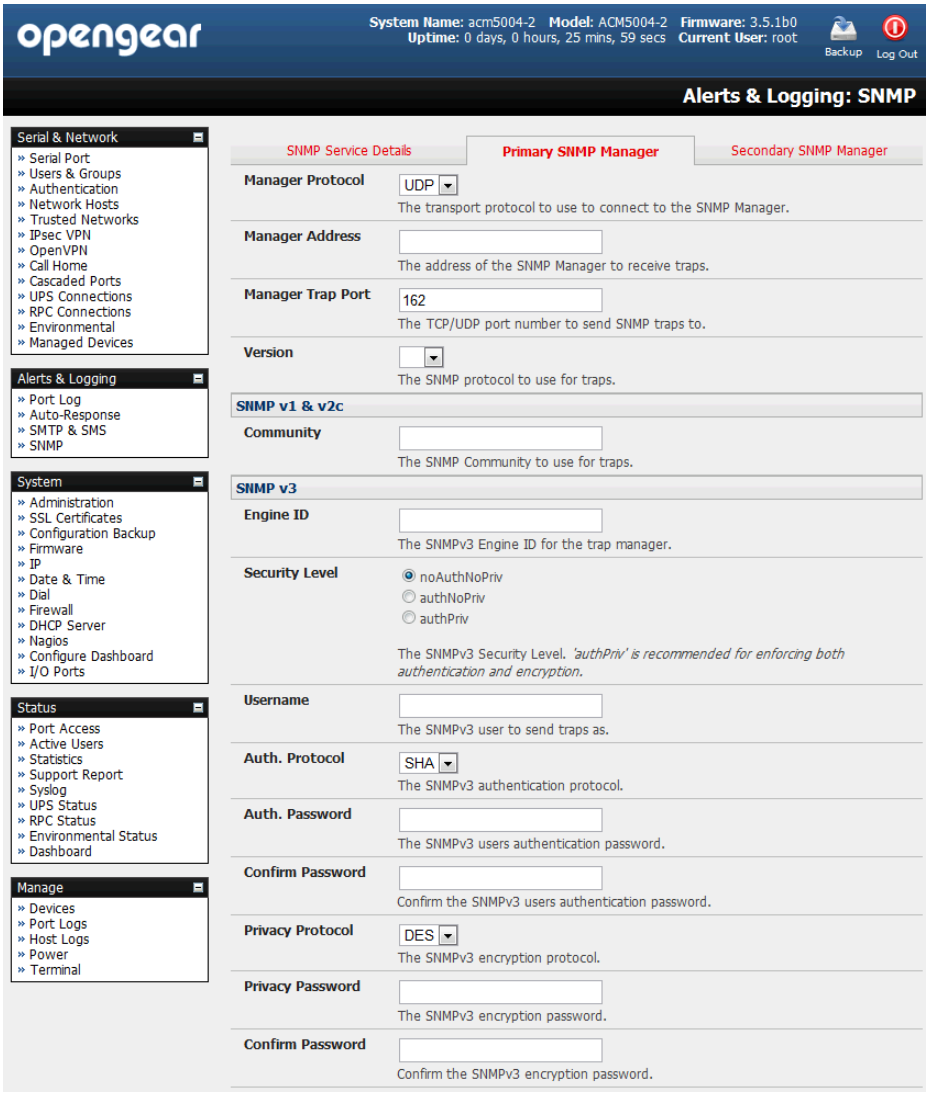


When upgrading from old firmware which does not support the newer SNMP MIBs/traps (versions before 3.10.2) to firmware that does support the new MIBs/traps:

- If the SNMP service was enabled and an SNMP manager was configured before upgrading the firmware, the console server will be configured to use the legacy traps after upgrading
- If the SNMP service was not enabled or no SNMP manger was configured before the upgrade, then the console server will be configured to use the new SNMP traps after the upgrade (but note that this won't have any effect until the SNMP service is turned on and an SNMP manager is configured).
- When starting up in the new firmware after a config erase, the console server will be configured to use the new SNMP traps.
- When upgrading from a version of firmware that supports the new traps to a newer version that supports the new traps, the 'use legacy traps' setting should just be kept the same - no checking of snmp service/manager configuration is needed.

- Select **Primary SNMP Manager** tab. The Primary and Secondary SNMP Manager tabs are used to configure where and how outgoing SNMP alerts and notifications are sent. If you require your *console server* to send alerts via SNMP then, at a minimum, a Primary SNMP Manager must be configured. Optionally, a second SNMP Network Manager with its own SNMP settings can be specified on the **Secondary SNMP Manager** tab

Note All console servers can also be configured to provide status information on demand using *snmpd*. This SNMP agent is configured using the *SNMP Service Detail* on *Alerts & Logging: SNMP* - as described in Chapter 15



- Select the **Manager Protocol.** SNMP is generally a **UDP**-based protocol though infrequently it uses **TCP** instead.
- Enter the host address of the SNMP Network Manager into the **Manager Address** field.
- Enter the TCP/IP port number into the **Manager Trap Port** field (default =162).
- Select the **Version** to be used. The *console server* SNMP agent supports SNMP v1, v2 and v3
- Enter the **Community** name for SNMP v1 or SNMP v2c. At a minimum, a community needs to be set for either SNMP v1 or v2c traps to work. An SNMP community is the group to which devices and management stations running SNMP belong. It helps define where information is sent. SNMP default communities are ***private*** for Write and ***public*** for Read.
- Configure **SNMP v3** if required. For SNMP v3 messages, the user's details and security level must match what the receiving SNMP Network Manager is expecting. SNMP v3 mandates that the message will be rejected unless the SNMPv3 user sending the trap already exists in the user database on the SNMP Manager. The user database in a SNMP v3 application is actually referenced by a combination of the Username and the Engine ID for the given SNMP application you are talking to.
    - Enter the **Engine ID** for the user sending messages as a hex number e.g. 0x8000000001020304.
    - Specify the **Security Level**. The level of security has to be compatible with the settings of the remote SNMP Network Manager.

      noAuthNoPriv No authentication or encryption.

authNoPriv	Authentication only. An authentication protocol (SHA or MD5) and password will be required.
authPriv	Uses both authentication and encryption. This is the highest level of security and requires an encryption protocol (DES or AES) and password in addition to the authentication protocol and password.

- Complete the **Username.** This is the Security Name of the SNMPv3 user sending the message. This field is mandatory and must be completed when configuring the *console server* for SNMPv3.
- An **Authentication Protocol (SHA** or **MD5**) and **Authentication Password** must be given for a Security Level of either **authNoPriv** or **authPriv**. The password must contain at least 8 characters to be valid.
- A **Privacy Protocol** (**DES** or **AES**) must be specified for the **authPriv** level of security to be used as the encryption algorithm. AES is recommended for stronger security. A password of at least 8 characters must be provided for encryption to work.

➢ Click **Apply**

Note *Console servers* with V3.0 firmware (and later) also embed the *net-snmpd* daemon which can accept SNMP requests from remote SNMP management servers and provides information on alert / serial / device status (refer *Chapter 15.5* for more details). *Console servers* with firmware earlier than V3.3 could only configure a Primary SNMP server from the Management Console. Refer *Chapter 15.5 f*or details on configuring the *snmptrap* daemon to send traps/notifications to multiple remote SNMP servers.

7.5.4 Send Nagios Event alerts

To notify the central Nagios server of Alerts, NSCA must be enabled under **System: Nagios** and Nagios must be enabled for each applicable host or port under **Serial & Network: Network Hosts** or **Serial & Network: Serial Ports** *(refer Chapter 10).*

Note: In a Lighthouse CMS centrally managed environment you can check the Nagios alert option. On the trigger condition (for matched patterns, logins, power events and signal changes) an NSCA check "warning" result will be sent to the central Nagios server. This condition is displayed on the Nagios status screen and triggers a notification, which can then cause the Nagios central server itself to send out an email or an SMS, page, etc.

7.6 Logging

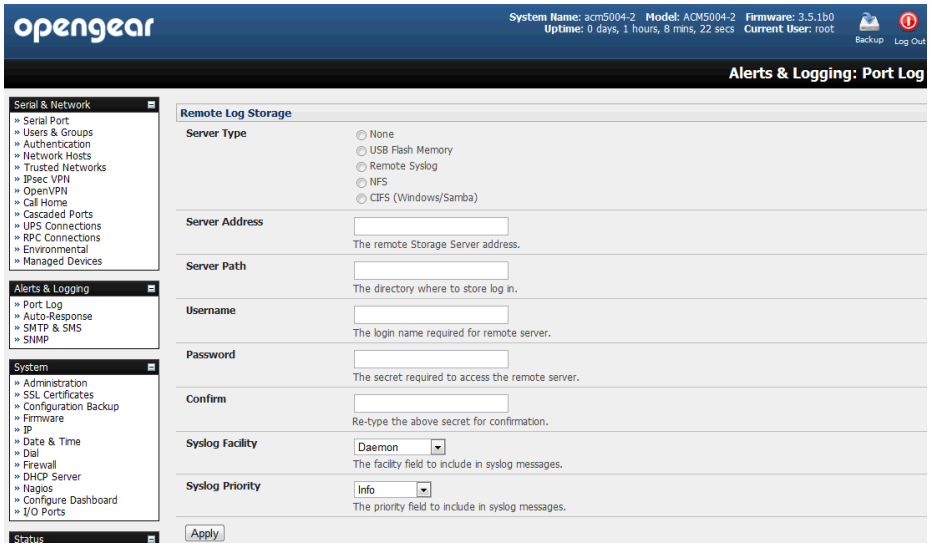
The *console server* can maintain log records of auto-response events and log records of all access and communications events (with the *console server* and with the attached serial, network and power devices).

A log of all system activity is also maintained by default, as is a history of the status of any attached environmental monitors.

7.6.1 Log storage

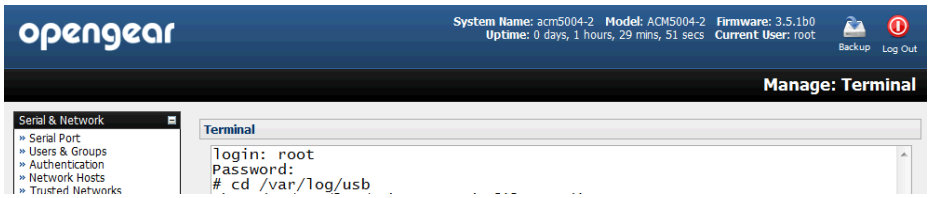
Before activating any Event, Serial, Network or UPS logging, you must specify where those logs are to be saved. These records are stored off-server or in the ACM/IM gateway USB flash memory.

➢ Select the **Alerts & Logging: Port Log** menu option and specify the **Server Type** to be used, and the details to enable log server access



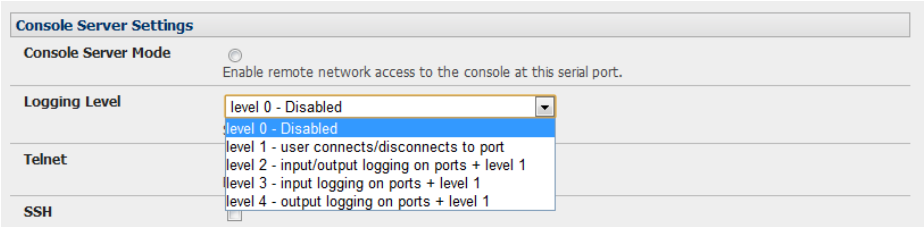
From the **Manage: Devices** menu the *Administrator* will can view serial, network and power device logs stored in the *console reserve* memory (or flash USB). The *User* will only see logs for the Managed Devices they (or their Group) have been given access privileges for (Refer Chapter 13).

Event logs on the USB can be viewed using the web terminal or by ssh/telnet connecting to the console server.



7.6.2 Serial port logging

In *Console Server* mode, activity logs can be maintained of all serial port activity. To specify which serial ports are to have activities recorded and to what level data is to be logged:



- Select **Serial & Network: Serial Port** and **Edit** the port to be logged
- Specify the **Logging Level** of for each port as:

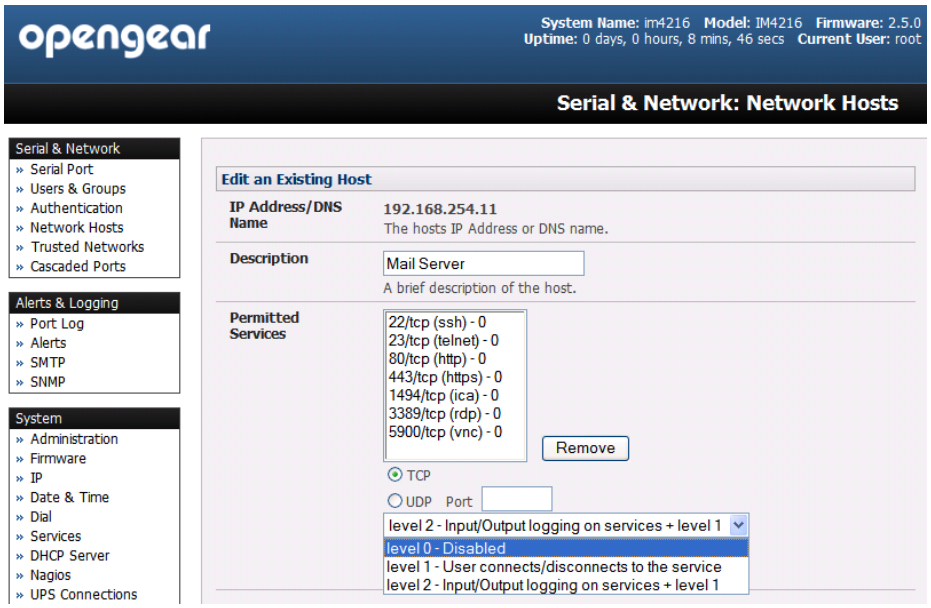
  Level 0 Turns off logging for the selected port
  Level 1 Logs all *User* connection events to the port
  Level 2 Logs all data transferred to and from the port and all changes in hardware flow control status and all *User* connection events
  Level 3 Logs all data transferred from the port and all changes in hardware flow control status and all *User* connection events
  Level 4 Logs all data transferred to the port and all changes in hardware flow control status and all *User* connection events
- Click **Apply**

Note A cache of the most recent 8K of logged data per serial port is maintained locally (in addition to the Logs which are transmitted for remote/USB flash storage). To view the local cache of logged serial port data select **Manage: Port Logs**

7.6.3 Network TCP and UDP port logging

The *console server* support optional logging of access to and communications with network attached Hosts.

- For each Host, when you set up the Permitted Services which are authorized to be used, you also must set up the level of logging that is to be maintained for each service



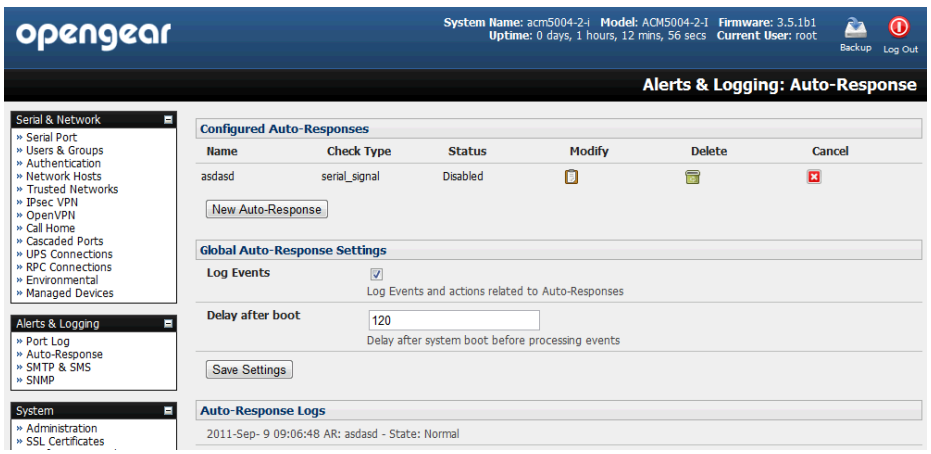
- Specify the logging level that is to be maintained for that particular TDC/UDP port/service, on that particular Host:

  Level 0 Turns off logging for the selected TDC/UDP port to the selected Host
  Level 1 Logs all connection events to the port
  Level 2 Logs all data transferred to and from the port

- Click **Add** then click **Apply**

7.6.4 Auto-Response event logging

- Check **Log Events** on **Alerts & Logging: Auto-Response** to enable logging all Auto-Response activities



7.6.5 Power device logging

The *console server* also logs access and communications with network attached hosts and maintain a history of the UPS and PDU power status.

To activate and set the desired levels of logging for each serial *(Section 7.4)* and/or network port *(Section 7.5)* and/or power and environment UPS (refer *Chapter 8*)

POWER, ENVIRONMENT & DIGITAL I/O

Opengear *console servers* manage Remote Power Control devices (RPCs including PDUs and IPMI devices) and Uninterruptible Power Supplies (UPSes). They also monitor remote operating environments using Environmental Monitoring Devices (EMDs) and sensors, and can provide digital I/O control.

8.1 Remote Power Control (RPC)

The *console server* Management Console monitors and controls Remote Power Control (RPC) devices using the embedded PowerMan and Network UPS Tools open source management tools and Opengear's power management software. RPCs include power distribution units (PDUs) and IPMI power devices.

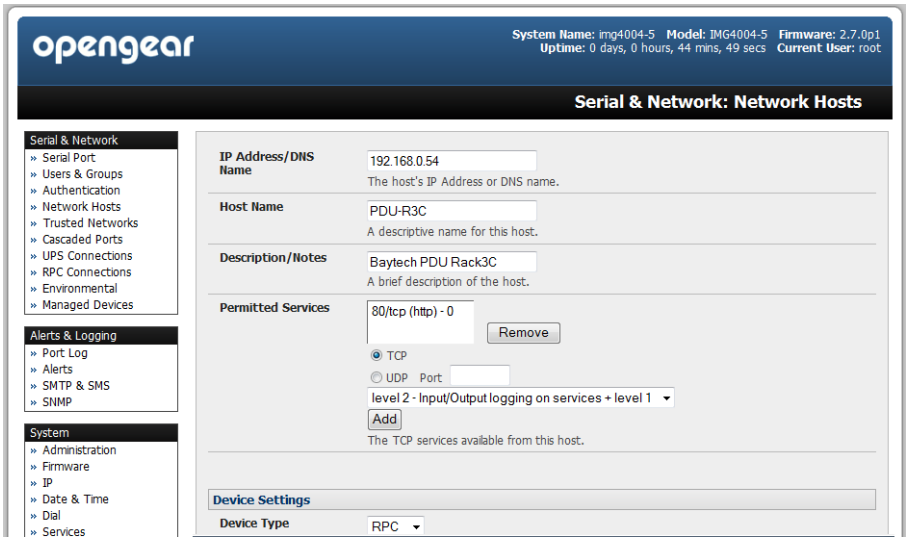
Serial PDUs invariably can be controlled using their command line console, so you could manage the PDU through the *console server* using a remote Telnet client. Also you could use proprietary software tools no doubt supplied by the vendor. This generally runs on a remote Windows PC and you could configure the *console server* serial port to operate with a serial COM port redirector in the PC (as detailed in *Chapter 4).* Similarly network-attached PDUs can be controlled with a browser (e.g. with SDT as detailed in *Chapter 6.3)* or an SNMP management package or using the vendor supplied control software. Also servers and network-attached appliances with embedded IPMI service processors or BMCs invariably are supplied with their own management tools (like SoL) that will provide secure management when connected using with SDT Connector.

However for simplicity all these devices can now all be controlled through the one window using the Management Console's RPC remote power control tools.

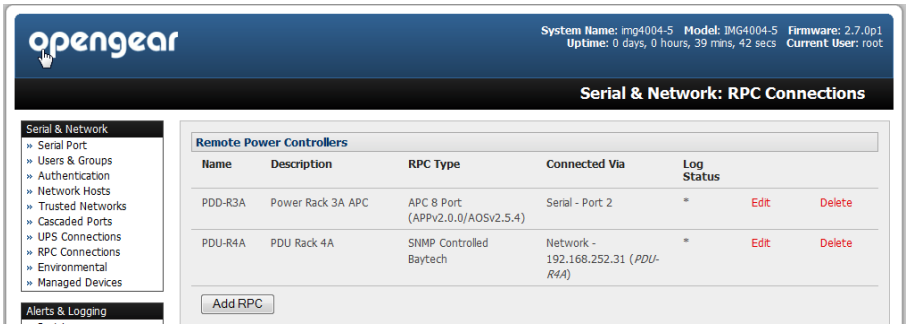
8.1.1 RPC connection

Serial and network connected RPCs must first be connected to, and configured to communicate with the *console server*:

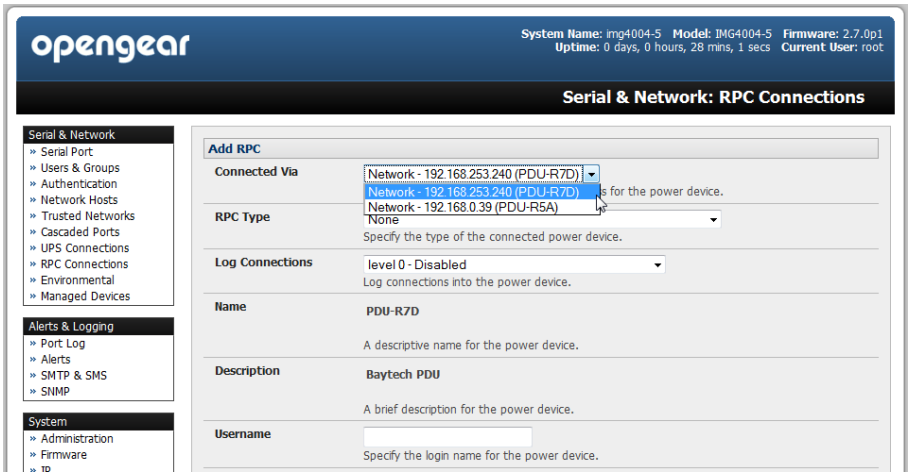
- For serial RPCs connect the PDU to the selected serial port on the *console server* and from the **Serial and Network: Serial Port** menu configure the **Common Settings** of that port with the RS232 properties etc required by the PDU (refer *Chapter 4.1.1 Common Settings*). Then select **RPC** as the **Device Type**
- Similarly for each network connected RPC go to **Serial & Network: Network Hosts** menu and configure the RPC as a connected Host by specifying it as **Device Type: RPC** and clicking **Apply** (refer *Chapter 4.4 Network Hosts*)



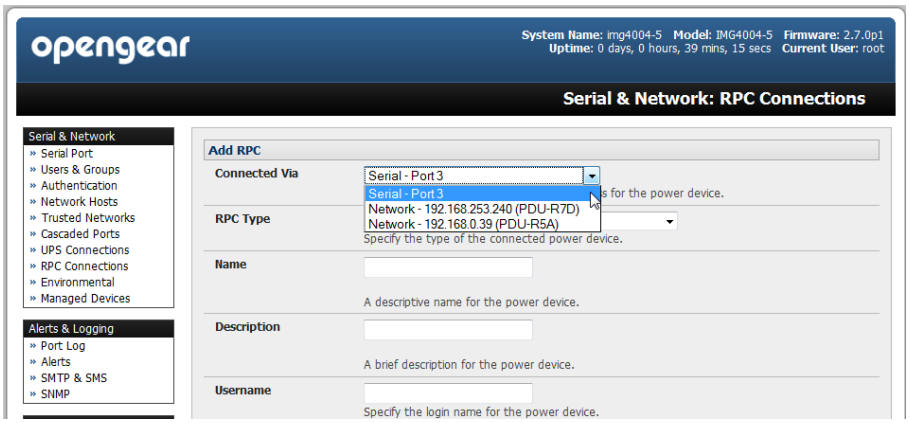
- Select the **Serial & Network: RPC Connections** menu. This will display all the RPC connections that have already been configured



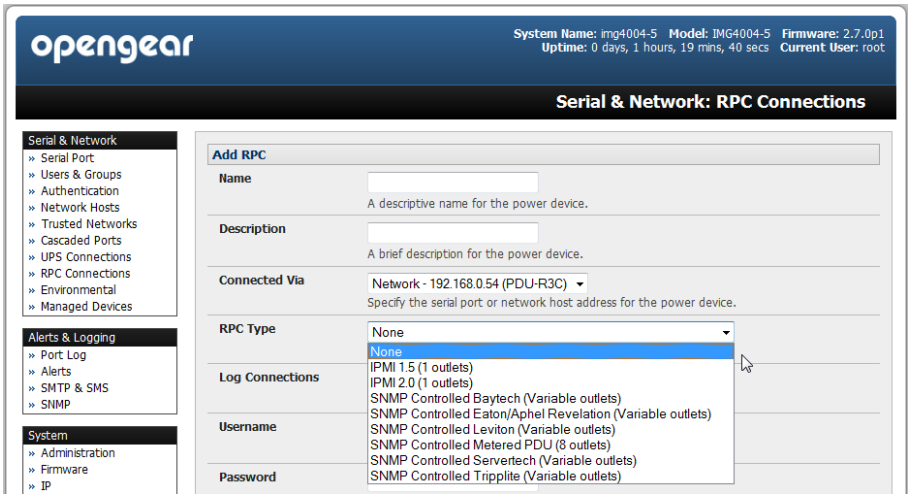
- Click **Add RPC**
- **Connected Via** presents a list of serial ports and network Host connections that you have set up with device type RPC (but have yet to connect to a specific RPC device):



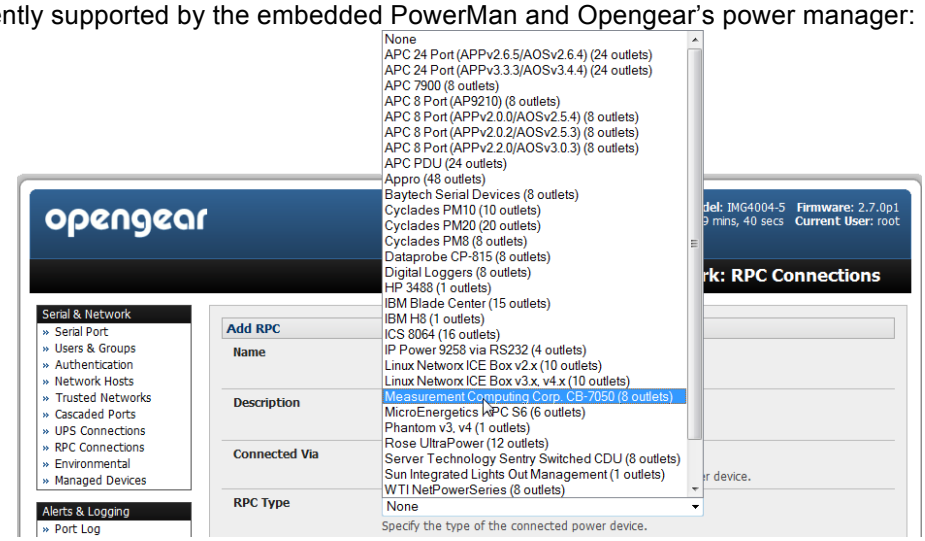
  - When you select **Connect Via** for a Network RPC connection then the corresponding Host Name/Description that you set up for that connection will be entered as the **Name** and **Description** for the power device
  - Alternately if you select to **Connect Via** a Serial connection then you will need to enter a **Name** and **Description** for the power device



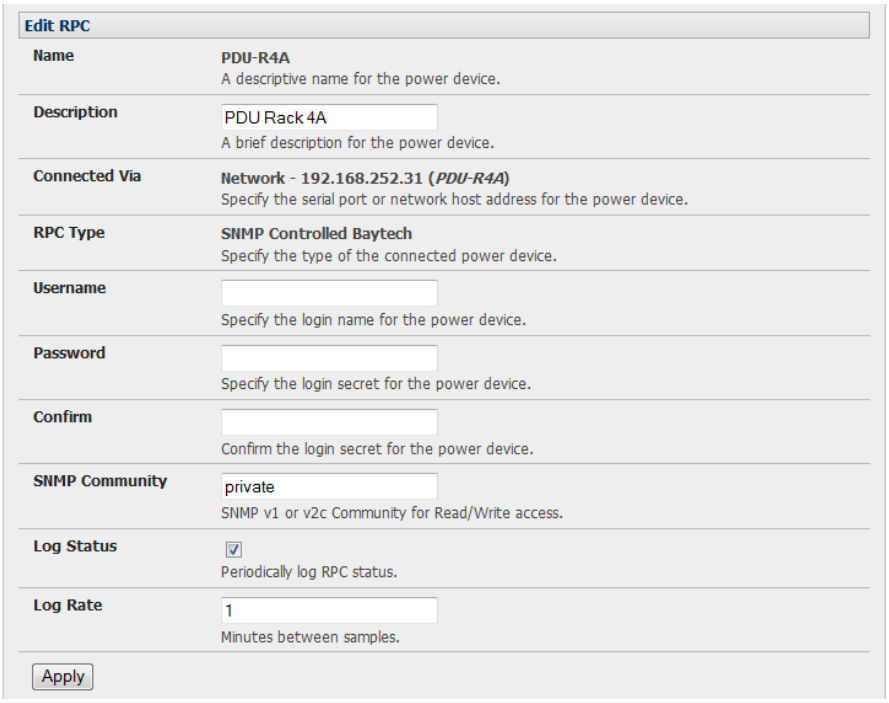
- Select the appropriate **RPC Type** for the PDU (or IPMI) being connected:
  - If you are connecting to the RPC via the network you will be presented with the IPMI protocol options and the SNMP RPC Types currently supported by the embedded Network UPS Tools



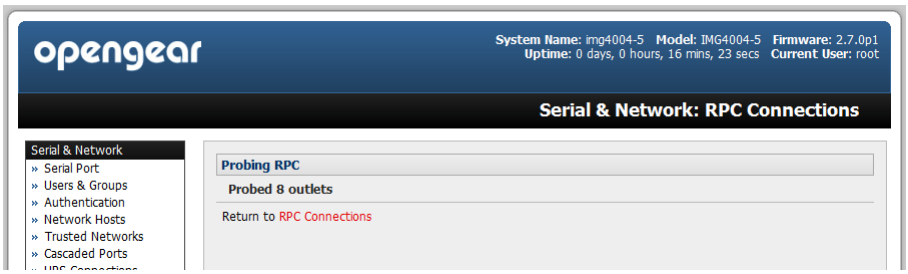
- If you are connecting to the RPC by a serial port you will be presented with all the serial RPC types currently supported by the embedded PowerMan and Opengear's power manager:



- Enter the **Username** and **Password** used to login into the RPC (Note that these login credentials are not related the *Users* and access privileges you will have configured in *Serial & Networks: Users & Groups*)
- If you selected SNMP protocol you will need to enter the SNMP v1 or v2c Community for Read/Write access (by default this would be "private")



- Check **Log Status** and specify the **Log Rate** (minutes between samples) if you wish the status from this RPC to be logged. These logs can be views from the **Status: RPC Status** screen
- Click **Apply**
- For SNMP PDUs the *console server* will now probe the configured RPC to confirm the RPC Type matches and will report the number of outlets it finds that can be controlled. If unsuccessful it will report **Unable to probe outlets** and you'll need to check the RPC settings or network/serial connection



- For serially connected RPC devices, a new Managed Device (with the same name as given to the RPC) will be created. The *console server* will then configure the RPC with the number of outlets specified in the selected RPC Type or will query the RPC itself for this information

Note Opengear's *console servers* support the majority of the popular network and serial PDUs. If your PDU is not on the default list then support can be added directly (as covered in Chapter 14 - Advanced Configurations) or by having the PDU supported added to either the Network UPS Tools or PowerMan open source projects.

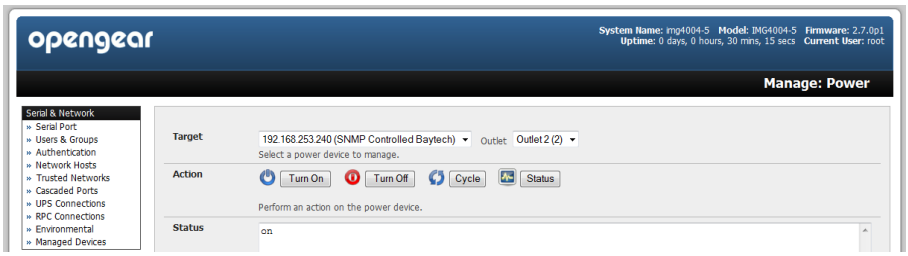
IPMI service processors and BMCs can be configured so all authorized users can use the Management Console to remotely cycle power and reboot computers, even when their operating system is unresponsive. To set up IPMI power control, the *Administrator* first enters the IP address/domain name of the BMC or service processor (e.g. a Dell DRAC) in **Serial & Network: Network Hosts,** then in **Serial & Network: RPC Connections** specifies the **RPC Type** to be IPMI1.5 or 2.0

8.1.2 RPC access privileges and alerts

You can now set PDU and IPMI alerts using **Alerts & Logging: Alerts** (refer *Chapter 7*). You can also assign which user can access and control which particular outlet on each RPC using **Serial &Network: User &Groups** (refer *Chapter 4*)

8.1.3 User power management

The Power Manager enables both *Users* and *Administrator*s to access and control the configured serial and network attached PDU power strips, and servers with embedded IPMI service processors or BMCs:



- Select the **Manage: Power** and the particular **Target** power device to be controlled (and the Outlet to be controlled if the RPC supports outlet level control)
- The outlet status is displayed and you can initiate the desired **Action** to be taken by selecting the appropriate icon:

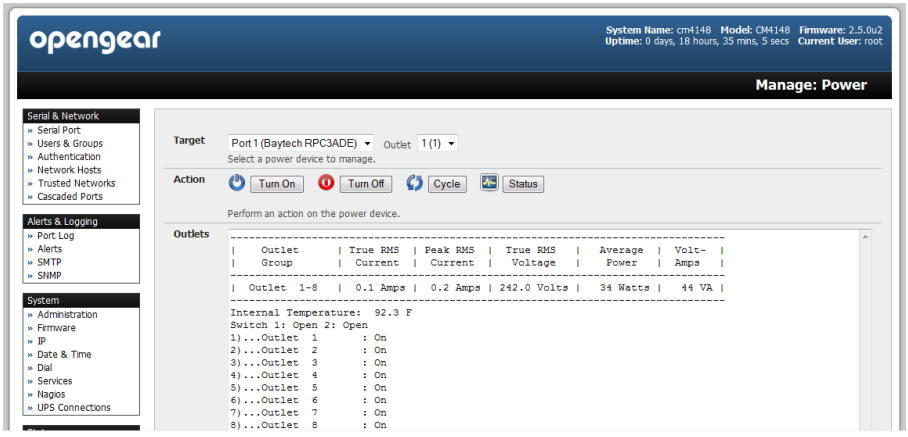
Turn ON

Turn OFF

Cycle

Status

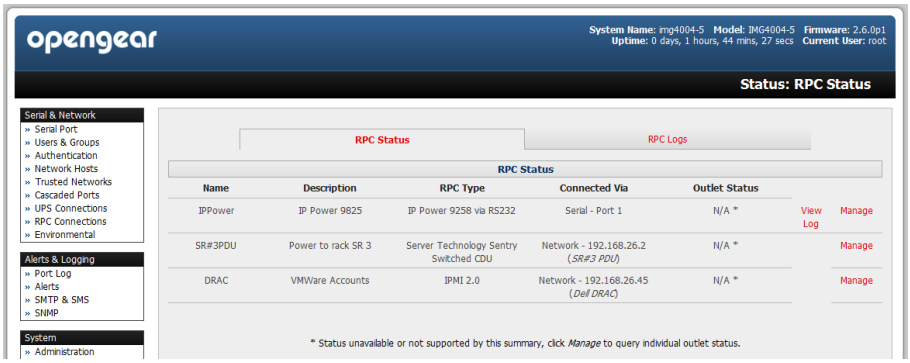
You will only be presented with icons for those operations that are supported by the **Target** you have selected.



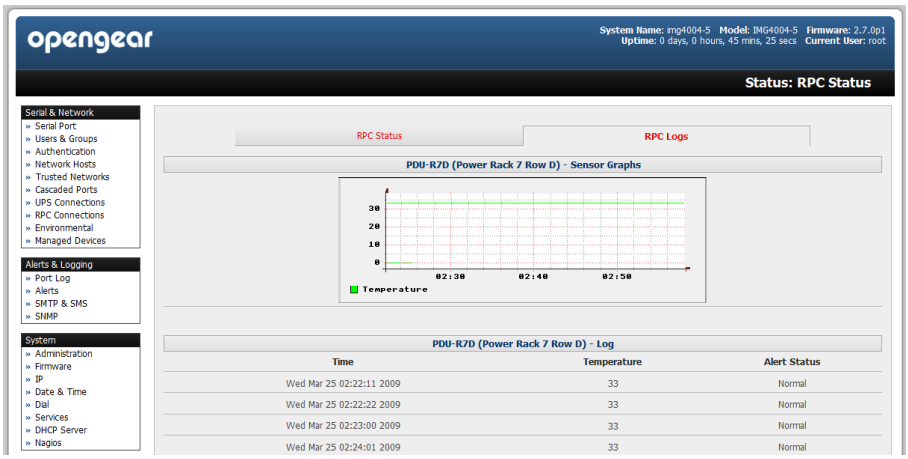
8.1.4 RPC status

You can monitor the current status of your network and serially connected PDUs and IPMI RPCs

- Select the **Status: RPC Status** menu and a table with the summary status of all connected RPC hardware will be displayed



- Click on **View Log** or select the **RPCLogs** menu and you will be presented with a table of the history and detailed graphical information on the selected RPC

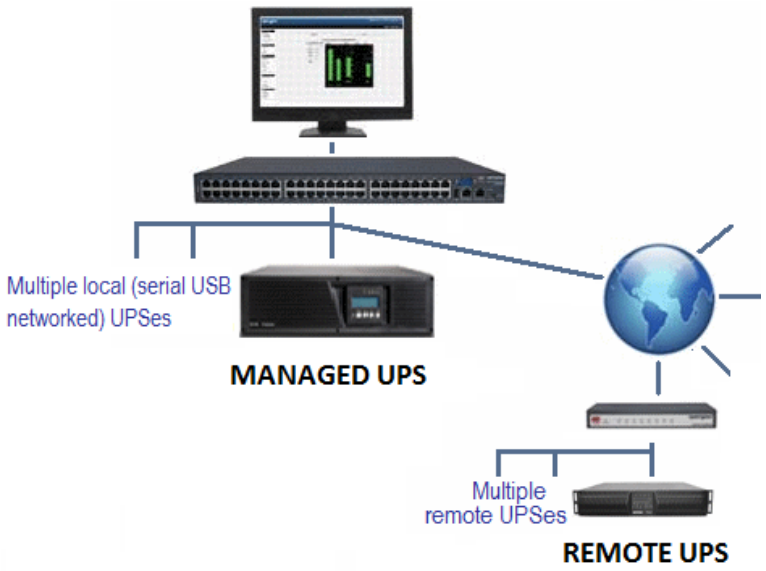


- Click **Manage** to query or control the individual power outlet. This will take you to the **Manage: Power** screen

8.2 Uninterruptible Power Supply Control (UPS)

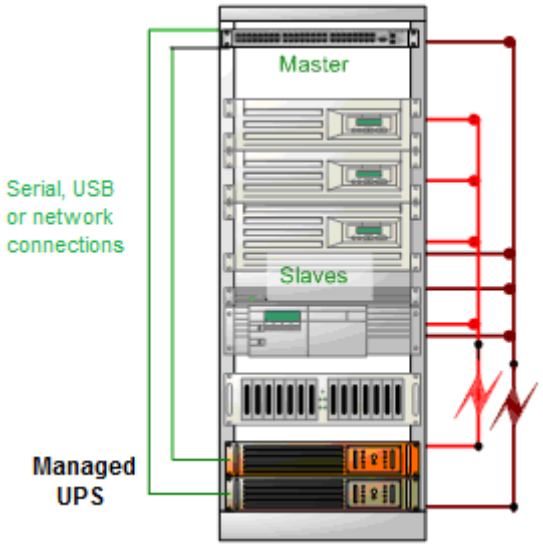
All Opengear *console servers* can be configured to manage locally and remotely connected UPS hardware using Network UPS Tools.

Network UPS Tools (NUT) is a group of open source programs that provide a common interface for monitoring and administering UPS hardware; and ensuring safe shutdowns of the systems which are connected. NUT is built on a networked model with a layered scheme of drivers, server and clients (covered in some detail in *Chapter 8.2.6*)



8.2.1 Managed UPS connections

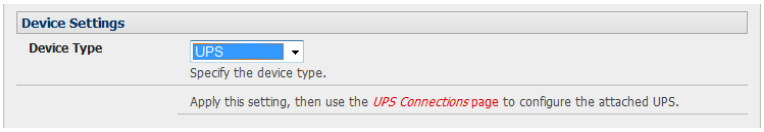
A ***Managed UPS*** is a UPS that is directly connected as a Managed Device to the *console server*. It can be connected by serial or USB cable or by the network. The *console server* becomes the *master* of this UPS, and runs a *upsd* server to allow other computers that are drawing power through the UPS (*slaves*) to monitor the UPS status and take appropriate action such as shutdown in event of low UPS battery.



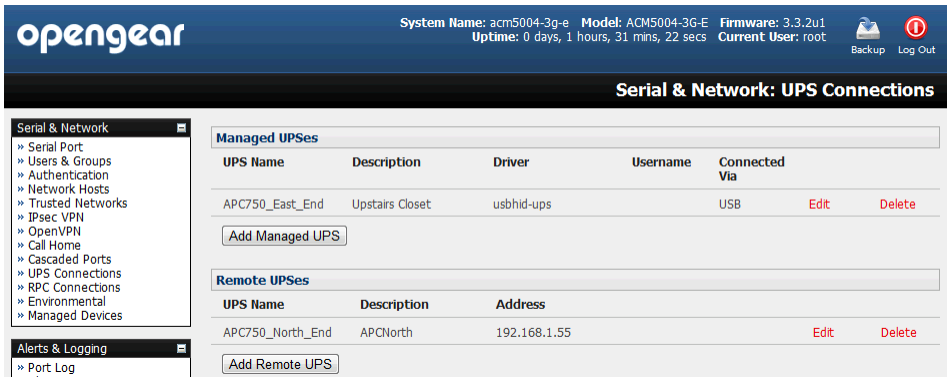
The *console server* may or may not be drawing power itself through the Managed UPS. When the UPS's battery power reaches critical, the *console server* signals and waits for *slaves* to shut down, then powers off the UPS.

Serial and network connected UPSes must first be connected to, and configured to communicate with the *console server*:

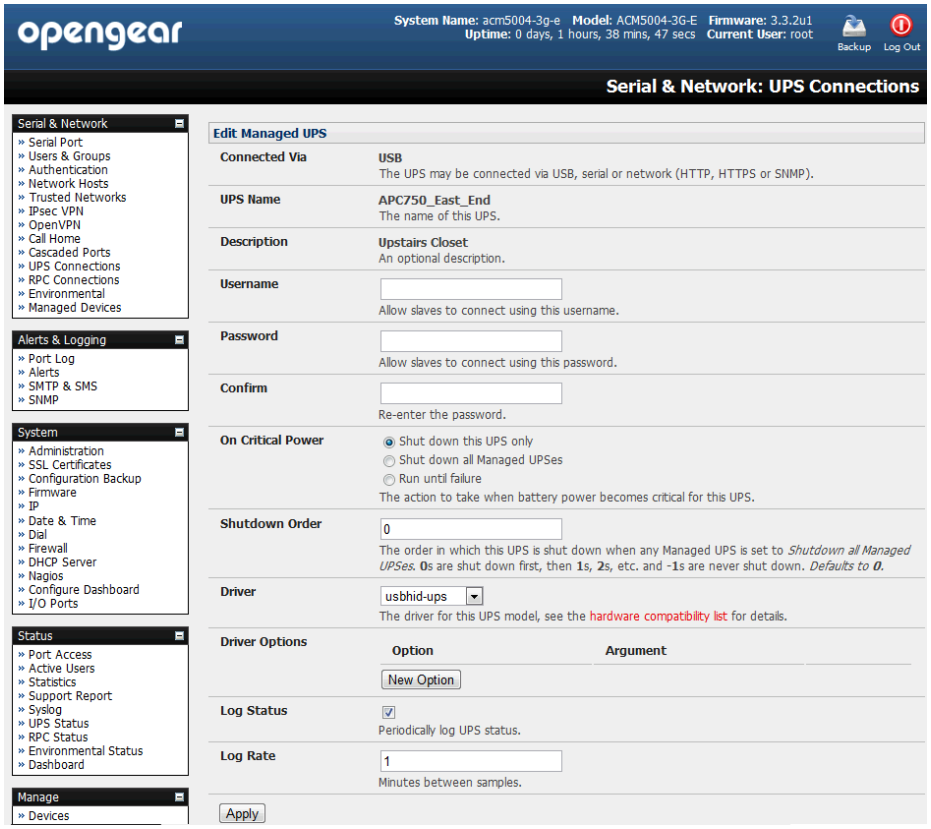
- For serial UPSes attach the UPS to the selected serial port on the *console server*. From the **Serial and Network: Serial Port** menu, configure the **Common Settings** of that port with the RS232 properties etc required by the UPS (refer *Chapter 4.1.1 Common Settings*). Then select **UPS** as the **Device Type**
- Similarly for each network connected UPS go to **Serial & Network: Network Hosts** menu and configure the UPS as a connected Host by specifying it as **Device Type: UPS** and clicking **Apply**



- No such configuration is required for USB connected UPS hardware



- Select the **Serial & Network: UPS Connections** menu. The **Managed UPSes** section will display all the UPS connections that have already been configured.
- Click **Add Managed UPS**



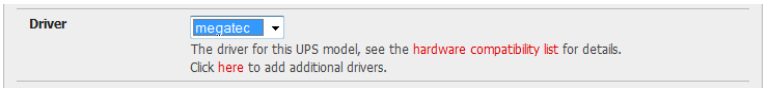
- Select if the UPS will be **Connected Via** USB or over pre-configured serial port or via SNMP/HTTP/HTTPS over the preconfigured network Host connection
- When you select a network UPS connection then the corresponding Host Name/Description that you set up for that connection will be entered as the **Name** and **Description** for the power device. Alternately if you selected to **Connect Via** a USB or serial connection then you will need to enter a **Name** and **Description** for the power device (and these details will also be used to create a new Managed Device entry for the serial/USB connected UPS devices)
- Enter the login details. This **Username** and **Password** is used by *slaves* of this UPS (i.e. other computers that are drawing power through this UPS) to connect to the *console server* to monitor the UPS status so they can shut themselves down when battery power is low. Monitoring will typically be performed using the *upsmon* client running on the slave server (refer *section 8.2.3*)

Note: These login credentials are not related the *Users* and access privileges you will have configured in *Serial & Networks: Users & Groups*

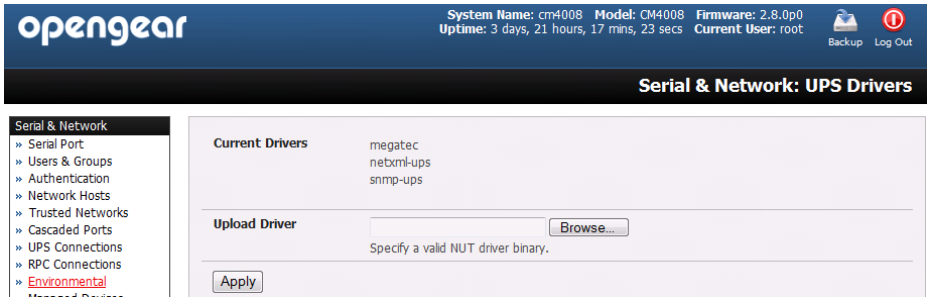
- Select the action to take when UPS battery power becomes critical i.e. Shut down the UPS (or Shut down all Managed UPSes) or simply Run until failure

Note: The shutdown script */etc/scripts/ups-shutdown* can be customized so, in the event of a critical power failure (when the UPS battery runs out) you can perform program the console server to perform "last gasp" actions using before power is lost! Refer online FAQ for details. However it generally is much simpler to perform such "last gasp" actions by triggering Auto-Response on the UPS hitting *batt* or *lowbatt.* Refer Chapter 7

- If you have multiple UPSes and require them to be shut down in a specific order, specify the **Shutdown Order** for this UPS. This is a whole positive number, or *-1*. *0s* are shut down first, then *1s*, *2s*, etc. *-1s* are not shut down at all. Defaults to *0*
- Select the **Driver** that will be used to communicate with the UPS. Most *console servers* are preconfigured so the drop down menu presents full selection of drivers from the latest Network UPS Tools (NUT version 2.4)

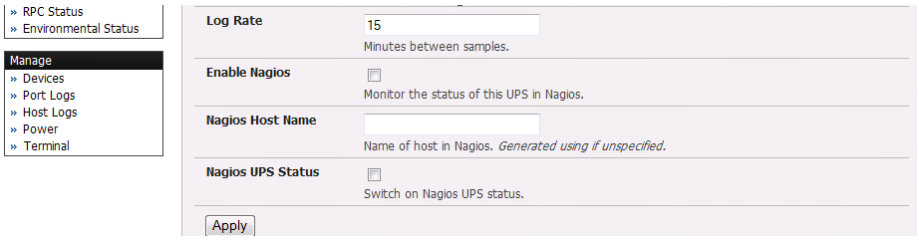


However for the SD4001/4002 models you will need to upload the driver you need from *www.opengear.com/download*



- Click **New Options** in **Driver Options** if you need to set driver-specific options for your selected NUT driver and hardware combination (more details at http://www.networkupstools.org/doc)

- Check **Log Status** and specify the **Log Rate** (minutes between samples) if you wish the status from this UPS to be logged. These logs can then be viewed from the **Status: UPS Status** screen
- If you have enabled Nagios services then you will presented with an option for Nagios monitoring. Check **Enable Nagios** to enable this UPS to be monitored using Nagios central management



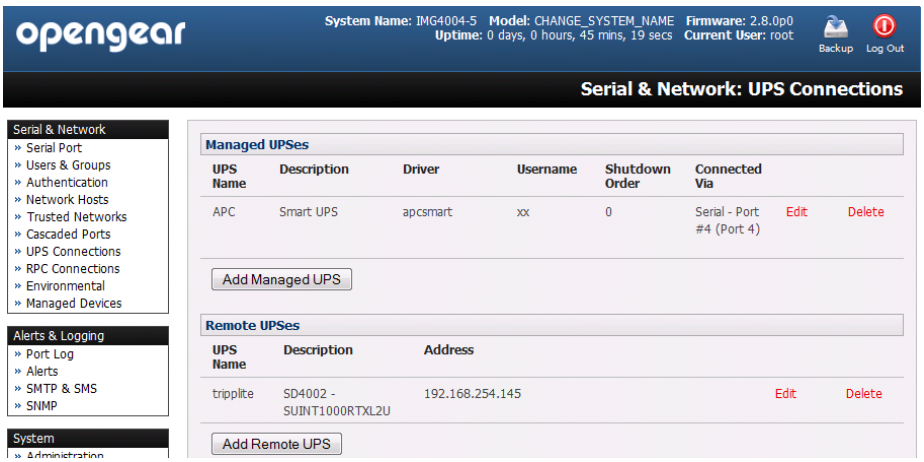
- Check **Enable Shutdown Script** if this is the UPS providing power to the *console server* itself and in the event of a critical power failure you can perform any "*last gasp*" actions on the *console server* before power is lost. This is achieved by placing a custom script in */etc/config/scripts/ups-shutdown* (you may use the provided */etc/scripts/ups-shutdown* as a template). This script is only run when then UPS reaches critical battery status
- Click **Apply**

Note: You can also customize the *upsmon*, *upsd* and *upsc* settings for this UPS hardware directly from the command line

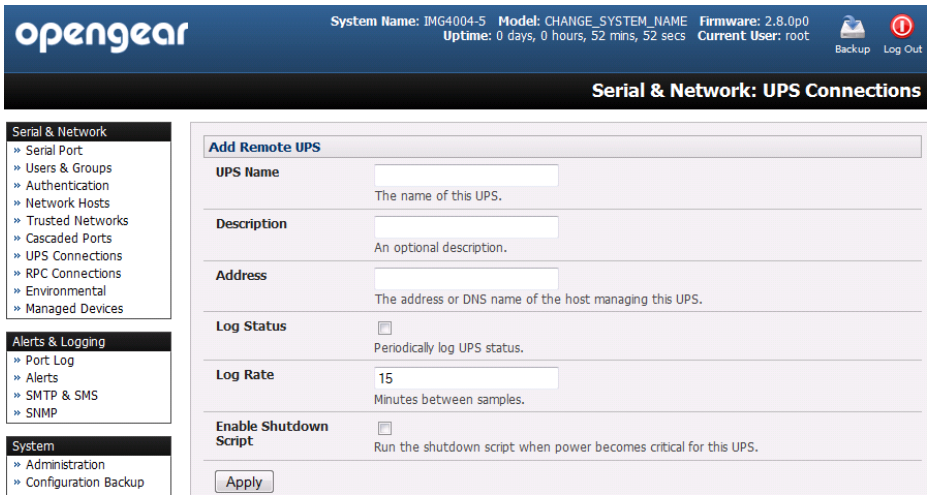
8.2.2 Remote UPS management

A ***Remote UPS*** is a UPS that is connected as a Managed Device to some remote *console server* which is being monitored (but not managed) by your *console server*.

The *upsc* and *upslog* clients in the Opengear *console server* can configured to monitor remote servers that are running Network UPS Tools managing their locally connected UPSes. These remote servers might be other Opengear *console servers* or generic Linux servers running NUT. So all these distributed UPSes (which may be spread in a row in a data center, or around a campus property or across the country) can be centrally monitored through the one central *console server* window. To add a Remote UPS:



- Select the **Serial & Network: UPS Connections** menu. The **Remote UPSes** section will display all the remote UPS devices being monitored
- Click **Add Remote UPS**



- Enter the **Name** of the particular remote UPS to be remotely monitored. This name must be the name that the remote UPS was configured with on the remote *console server* (as the remote *console server* may itself have multiple UPSes attached that it is managing locally with NUT). Optionally enter a **Description**
- Enter the IP **Address** or DNS name of the remote *console server** that is managing the remote UPS. (*This may be another Opengear *console server* or it may be a generic Linux server running Network UPS Tools)

Note An example where centrally monitor remotely distributed UPSes is useful is a campus or large business site where there's a multitude of computer and other equipment sites spread afar, each with their own UPS supply … and many of these (particularly the smaller sites) will be USB or serially connected.

Having a SD4001/2, ACM5000 or ACM5500 at these remote sites would enable the system manager to centrally monitor the status of the power supplies at all sites, and centralize alarms. So he/she can be warned to initiate a call-out or take shut down actions

- Check **Log Status** and specify the **Log Rate** (minutes between samples) if you wish the status from this UPS to be logged. These logs can then be viewed from the **Status: UPS Status** screen
- Check **Enable Shutdown Script** if this remote UPS is the UPS providing power to the *console server* itself. In the event the UPS reaches critical battery status the custom script in */etc/config/scripts/ups-shutdown* is run enabling you to perform any "*last gasp*" actions
- Click **Apply**

Note: The Remote UPS feature is supported on all *console servers* with V2.8 firmware and later. Earlier versions supported a single remote "Monitored UPS" which could be set to trigger the *console server shutdown script*

8.2.3 Controlling UPS powered computers

One of the advantages of having a Managed UPS is that you can configure computers that draw power through that UPS to be shut down gracefully in the event of UPS problems.

For Linux computers this can be done by setting up *upsmon* on each computer and directing them to monitor the *console server* that is managing their UPS. This will set the specific conditions that will be used to initiate a power down of the computer. Non-critical servers may be powered down some second after the UPS starts running on battery. Whereas more critical servers may not be shut down till a low battery warning is received). Refer to the online NUT documentation for details on how this is done:

http://eu1.networkupstools.org/doc/2.2.0/INSTALL.html

http://linux.die.net/man/5/upsmon.conf

http://linux.die.net/man/8/upsmon

An example upsmon.conf entry might look like:

MONITOR managedups@192.168.0.1 1 username password slave

- *managedups* is the UPS Name of the Managed UPS

- *192.168.0.1* is the IP address of the Opengear *console server*

- *1* indicates the server has a single power supply attached to this UPS

- *username* is the Username of the Managed UPS

- password is the Password of the Manager UPS

There are NUT monitoring clients available for Windows computers (WinNUT).

If you have an RPC (PDU) it is also possible to shut down UPS powered computers and other equipment without them have a client running (e.g. communications and surveillance gear). Set up a UPS alert and using this to trigger a script which control a PDU to shut off the power (refer *Chapter 15*).

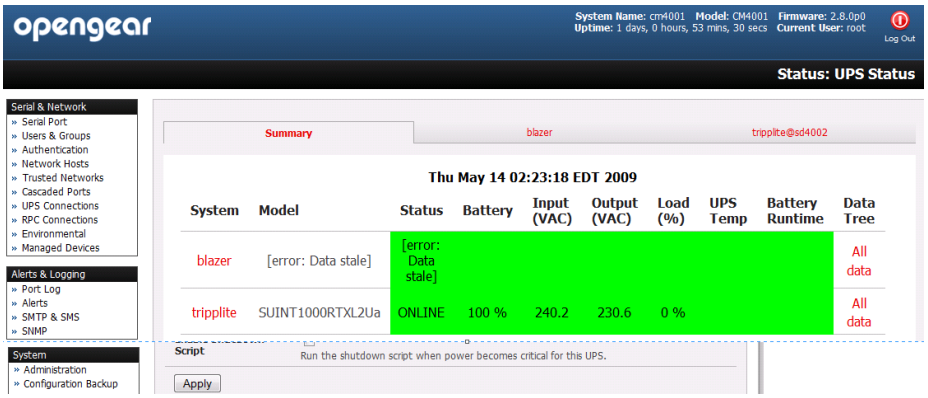
8.2.4 UPS alerts

You can set UPS alerts using **Alerts & Logging: Alerts** (refer *Chapter 7- Alerts & Logging*)

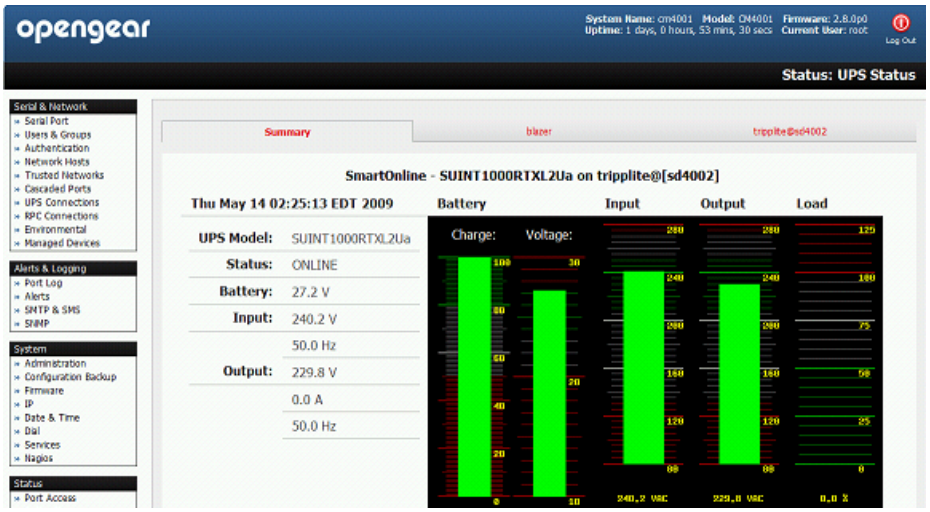
8.2.5 UPS status

You can monitor the current status of your network, serially or USB connected Managed UPSes and any configured Remote UPSes

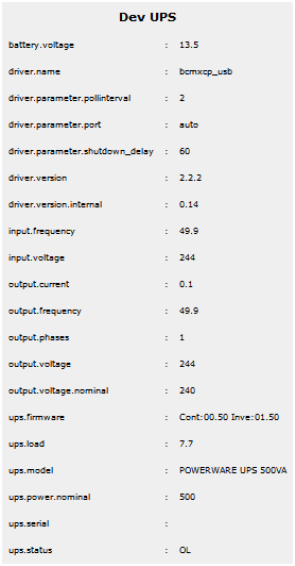
- Select the **Status: UPS Status** menu and a table with the summary status of all connected UPS hardware will be displayed



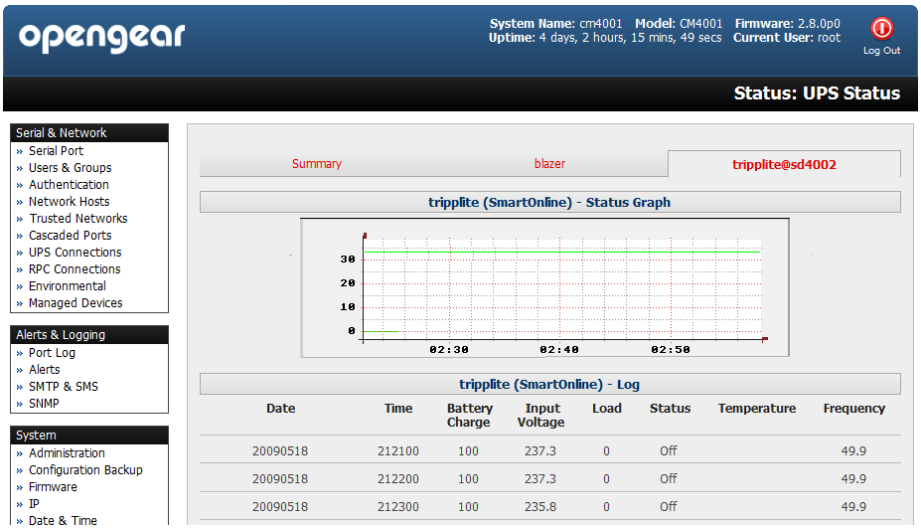
- Click on any particular UPS **System** name in the table and you will be presented with a more detailed graphical information on the select UPS System



- Click on any particular **All Data** for any UPS System in the table for more status and configuration information on the select UPS System

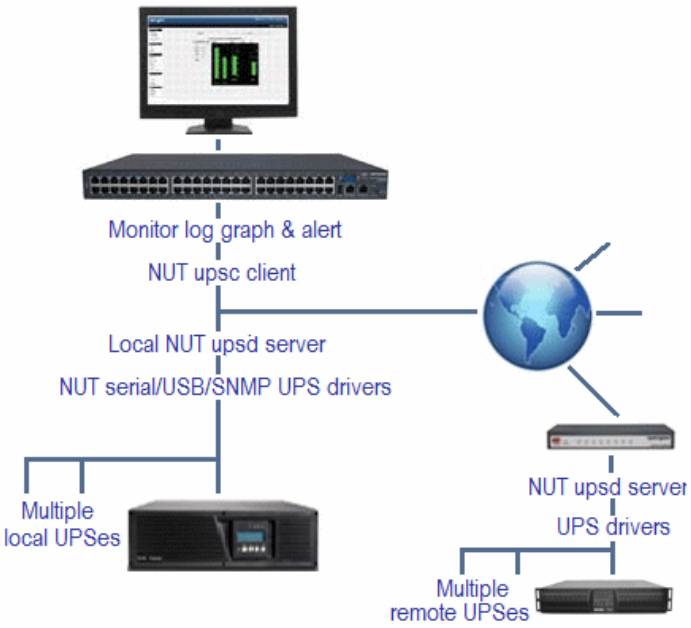


- Select **UPS Logs** and you will be presented with the log table of the load, battery charge level, temperature and other status information from all the Managed and Monitored UPS systems. This information will be logged for all UPSes which were configured with **Log Status** checked. The information is also presented graphically



8.2.6 Overview of Network UPS Tools (NUT)

NUT is built on a networked model with a layered scheme of drivers, server and clients. NUT can be configured using the Management Console as described above, or you can configure the tools and manage the UPSes directly from the command line. This section provides an overview of NUT however you can find full documentation at *http://www.networkupstools.org/doc*.



NUT is built on a networked model with a layered scheme of drivers, server and clients:

- The **driver** programs talk directly to the UPS equipment and run on the same host as the NUT network server (*upsd*). Drivers are provided for a wide assortment of equipment from most of the popular UPS vendors and understand the specific language of each UPS. They communicate to serial, USB and SNMP network connected UPS hardware and map the communications back to a compatibility layer. This means both an expensive "smart" protocol UPS and a simple "power strip" model can be handled transparently

- The NUT network **server** program *upsd* is responsible for passing status data from the drivers to the client programs via the network. *upsd* can cache the status from multiple UPSes and then serve this status data to

many clients. *upsd* also contains access control features to limit the abilities of the clients (e.g. so only authorized hosts may monitor or control the UPS hardware)

- There are a number of NUT **clients** that connect to *upsd* to check on the status of the UPS hardware and do things based on the status. These clients can run on the same host as the NUT server or they can communicate with the NUT server over the network (enabling them to monitor any UPS anywhere):
  - The *upsc* client provides a quick way to poll the status of a UPS server. It can be used inside shell scripts and other programs that need UPS data but don't want to include the full interface
  - The *upsmon* client enables servers that draw power through the UPS to shutdown gracefully when the battery power reaches critical
  - There are also logging clients (*upslog*) and third party interface clients (Big Sister, Cacti, Nagios, Windows and more)
- The latest release of NUT (2.4) also controls PDU systems. It can do this either natively using SNMP or through a *binding* to Powerman (open source software from Livermore Labs that also is embedded in Opengear *console servers*)

These NUT clients and servers all are embedded in each Opengear *console server* (with a Management Console presentation layer added) … and they also are run remotely on distributed *console servers* and other remote NUT monitoring systems. This layered distributed NUT architecture enables:

- Multiple manufacturer support: NUT can monitor UPS models from 79 different manufacturers - and PDUs from a growing number of vendors - with a unified interface
- Multiple architecture support: NUT can manage serial and USB connected UPS models with the same common interface. Network connected USB and PDU equipment can also be monitored using SNMP
- Multiple clients monitoring the one UPS: Multiple systems may monitor a single UPS using only their network connections and there's a wide selection of client programs which support monitoring UPS hardware via NUT (Big Sister, Cacti, Nagios and more)
- Central management of multiple NUT servers: A central NUT client can monitor multiple NUT servers that may be distributed throughout the data center, across a campus or around the world

NUT supports the more complex power architectures found in data centers, communications centers and distributed office environments where many UPSes from many vendors power many systems with many clients - and each of the larger UPSes power multiple devices - and many of these devices are in turn dual powered

8.3 Environmental Monitoring

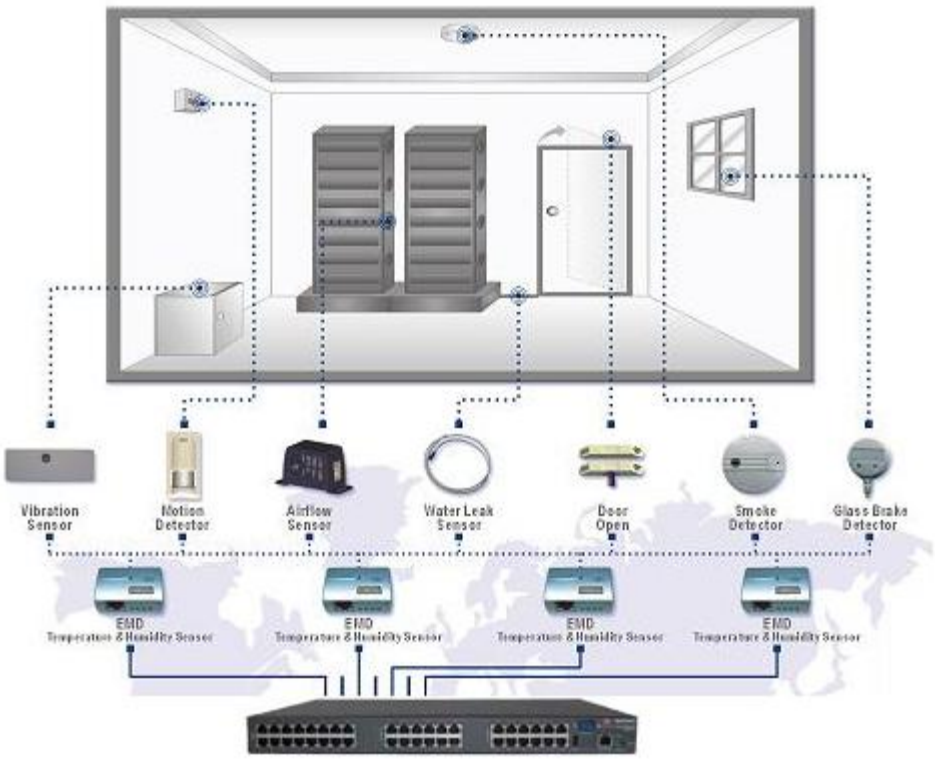
All Opengear console servers can be configured to monitor their operating environment.

External Environmental Monitor Devices (EMDs) can be connected to any Opengear *console server* serial port. Each *console server* can support multiple EMDs.

Each EMD device has an internal temperature and humidity sensor plus one or two general purpose status sensor ports which can be connected to smoke detectors, water detectors, vibration sensors or open-door sensors.

The ACM5000 and ACM5500 advanced *console server* models also each have internal temperature sensor and can optionally be configured to have up to four general purpose status sensor ports (which can be connected smoke or water detector and vibration or open-door sensors) directly connected.

Using the Management Console, *Administrator*s can view the ambient temperature (in °C or °F) and humidity (percentage) and configure alerts to monitor the status and sensors to automatically send alarms progressively from warning levels to critical.



8.3.1 Connecting the EMD and its sensors

The Environmental Monitor Device (EMD) connects to any serial port on the *console server* via a special EMD Adapter and standard CAT5 cable. The sensors then screw into the EMD:

EMD

- The EMD is powered over the serial port connection and communicates using a custom handshake protocol. It is not an RS232 device and should not be connected without the adapter

EMD Adapter

- Plug the male RJ plug on the EMD Adapter into the EMD. Then connect the Adapter to the *console server* serial port using the provided UTP cable. If the 6 foot (2 meter) UTP cable provided with the EMD is not long enough it can be replaced with a standard Cat5 UTP cable up to 33 feet (10meters) in length

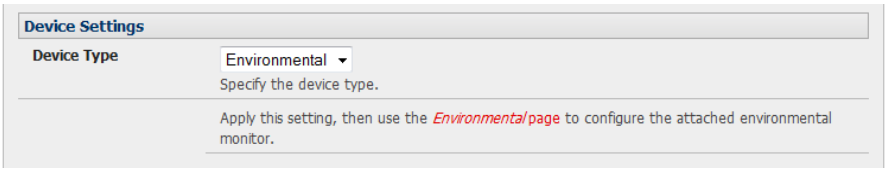
EMD sensor

- Screw the bare wires on any smoke detector, water detector, vibration sensor, open-door sensor or general purpose open/close status sensors into the terminals on the EMD

Note: You can attach two sensors onto the terminals on EMDs that are connected to *console servers* with Opengear Classic pinouts. However *console servers* with -01 and -02 pinouts only support attaching a single sensor to each EMD

The EMD can be used only with an Opengear *console server* and cannot be connected to standard RS232 serial ports on other appliances.

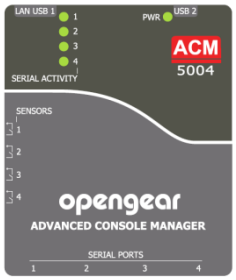
- Select **Environmental** as the **Device Type** in the **Serial & Network: Serial Port** menu for the port to which the EMD is to be attached. No particular Common Settings are required.
- Click **Apply**



8.3.2 Connecting sensors to ACM5000 and ACM5500s

You can connect EMDs (and their attached environmental sensors) to the serial ports on your ACM5000, as detailed in the previous section. However the ACM5000 can also support direct connection of environmental sensors.

All the ACM5000 models (except ACM5004-2-I) can be configured with the environmental option –E. Models with this option have a green connector block on the side (marked *SENSORS 1 -4)* and up to four environmental sensors can be directly attached to this block.



The ACM5004-2-I model is supplied with a green connector block on the side by default. The first two connectors on this block (marked *DIO1* and *DIO2*) can be configured to have external environmental sensors attached.

ACM7000, ACM5508-2-I and ACM5504-5-G-I models are also supplied with a green connector block on the side by default. The first two connectors on this block (marked *DIO1* and *DIO2*) can be configured to have external environmental sensors attached.

Note: The ACM5000-E Sensor 'inputs' are four 'dry contact' inputs which are normally open (NO). When open these are sensed as a TTL high or digital '1'. When activated the external devices (door close, vibration, water, smoke) present a short circuit and the contact 'closes to ground' which is read as a TTL low or a digital '0'.

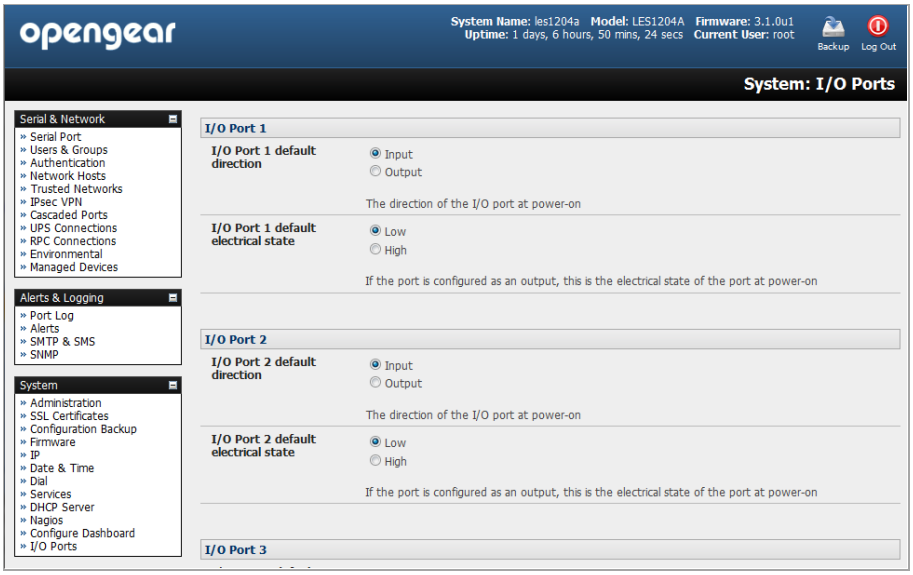
For custom applications a user can sense the state (closed or open) of non-Opengear dry contact sensors through the UI or command line.

It is also possible to control the sensor pins as 'outputs'. The user can set the pins as TTL high (1) or low (0) as required for their low voltage/low current application.

The ACM7000, ACM5004-2-I, ACM5508-2-I and ACM5504-5-G-I models have specific dedicated I/O (DIO1 & DIO2) and output only pins (OUT1 & OUT2), the later having inverting outputs with higher voltage/current transistor

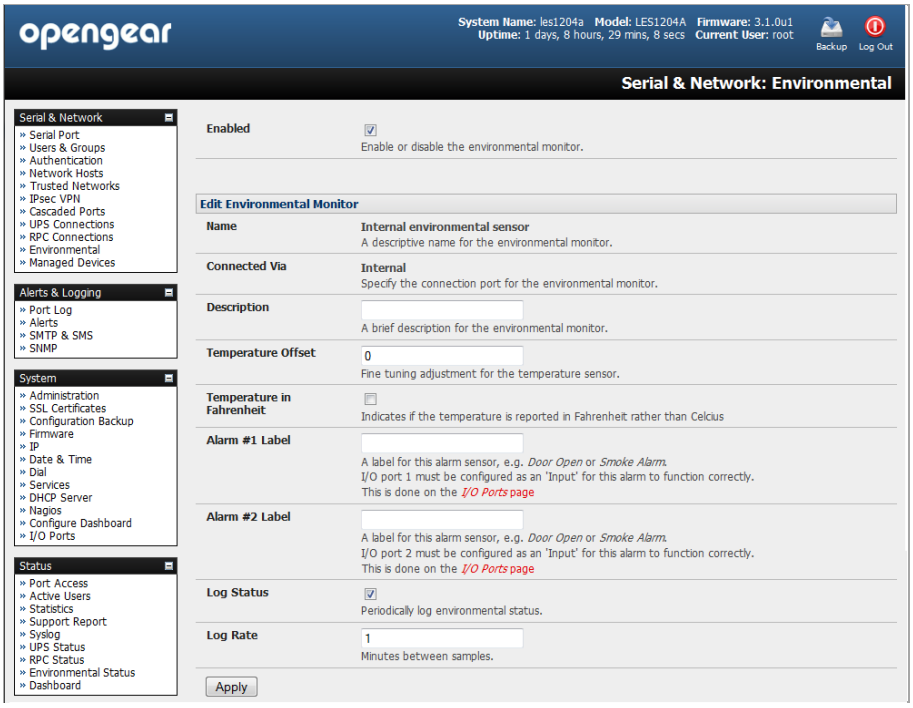
By default on the ACM5000 and ACM500 each *SENSOR* and *DIO* port is configured as an *Input,* so they are available to be used with external environmental sensors attached

- To confirm the direction and state configurations for these ports you can select the **System: I/O Ports** menu and a table with the summary status of the four digital I/O ports will be displayed. *I/O Port1* = *DIO1* or *SENSOR1*, *I/O Port2* = *DIO2* or *SENSOR2*, *I/O Port3* = *SENSOR3* and *I/O Port4* = *SENSOR 4)*



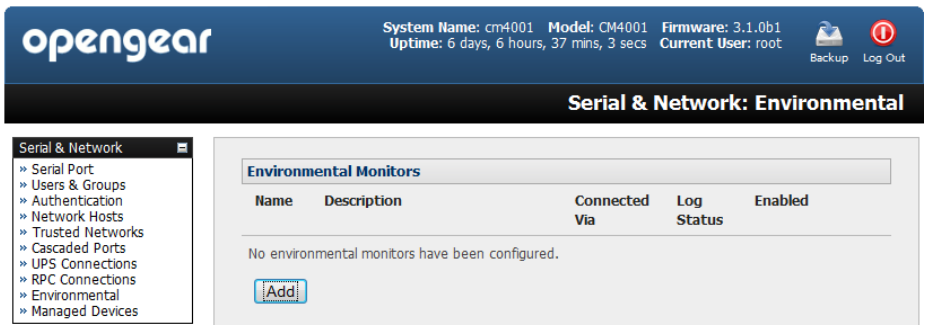
- Screw the bare wires on any smoke detector, water detector, vibration sensor, open-door sensor or general purpose open/close status sensors into the *SENSOR* or *DIO* terminals on the green connector block

- When configured as *Inputs*, the *SENSOR* and *DIO* ports are notionally attached to the internal EMD. So go to the **Serial & Network: Environmental** page and enable the **Internal EMD.** Then configure the attached sensors as alarms as covered in the next section

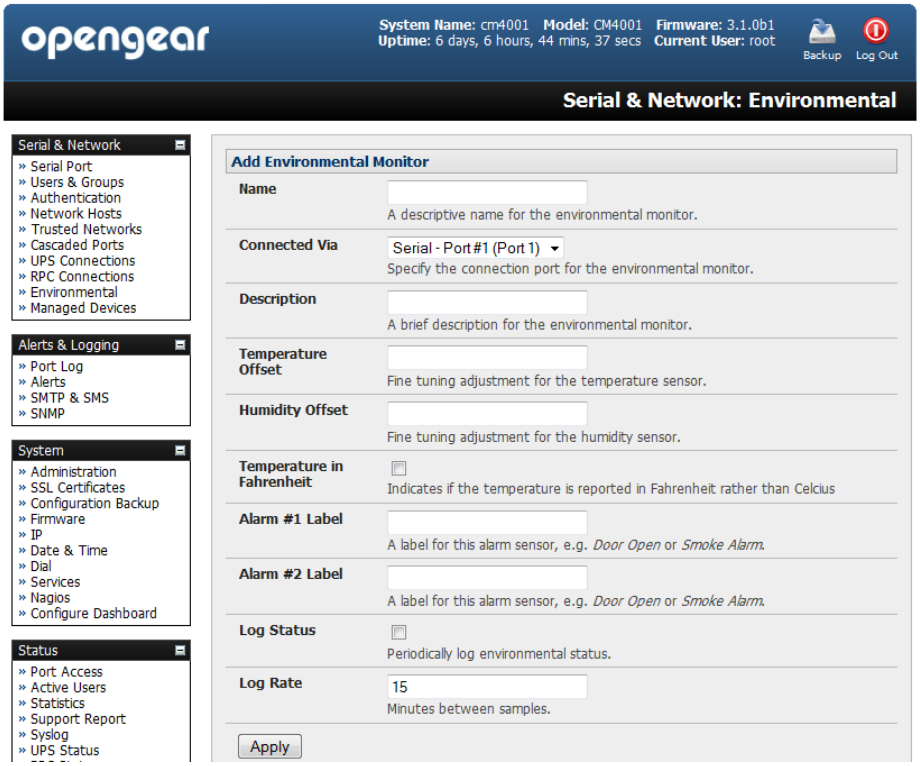


8.3.3 Adding EMDs and configuring the sensors

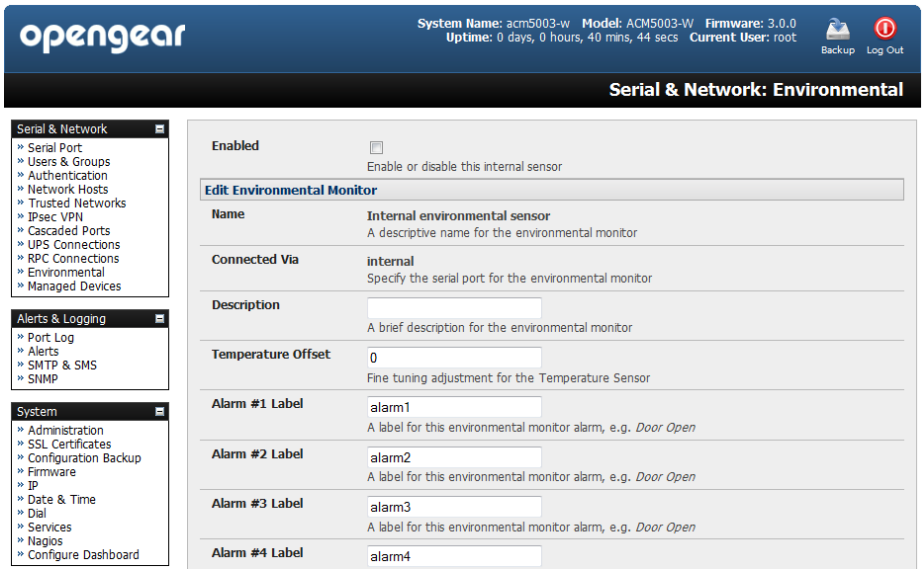
- Select the **Serial & Network: Environmental** menu. This will display any external EMDs or any "internal EMD" (i.e. sensors that may be directly attached to an ACM) that have already been configured



- To add a new EMD click **Add and** configure an external EMD enter a **Name** and optionally a **Description** and select the pre-configured serial port that the EMD will be **Connected Via**



- You may optionally calibrate the EMD with a Temperature Offset (+ or - °C) or Humidity Offset (+ or percent). Also if you check **Temperature in Fahrenheit** then the temperature will be reported in Fahrenheit. Otherwise it will be reported in degrees Celsius
- Provide **Labels** for each of the alarm sensors you will used e.g. Door Open or Smoke Alarm.
- Check **Log Status** and specify the **Log Rate** (minutes between samples) if you wish the status from this EMD to be logged. These logs can be views from the **Status: Environmental Status** screen
- Click **Apply**. This will also create a new Managed Device (with the same name)
- For the ACM5000-E select the **Serial & Network: Environmental** menu and check Enabled. You will then need set any temperature offsets and label the sensors as described above



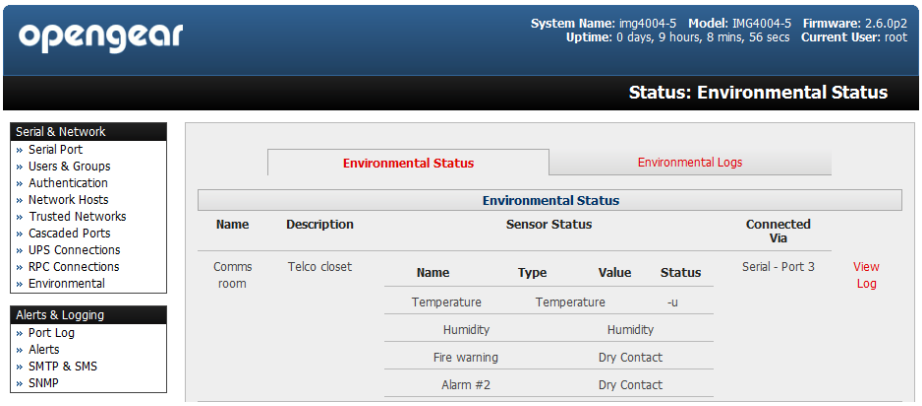
8.3.4 Environmental alerts

You can now set temperature, humidity and probe status alerts using **Alerts & Logging: Alerts** (refer *Chapter 7*)

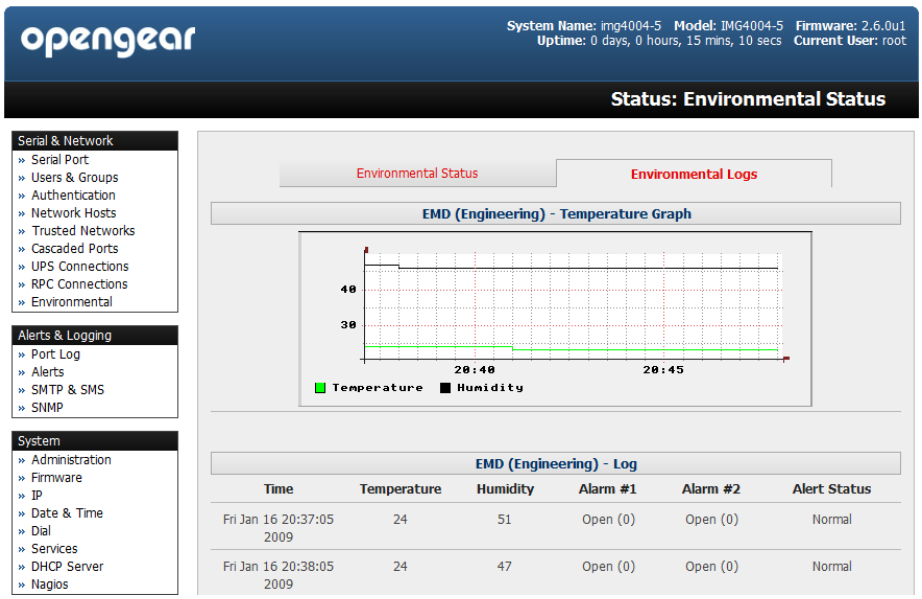
8.3.5 Environmental status

You can monitor the current status of all any configured external EMDs and their sensors, and any internal or directly attached sensors

- Select the **Status: Environmental Status** menu and a table with the summary status of all connected EMD hardware will be displayed



- Click on **View Log** or select the **Environmental Logs** menu and you will be presented with a table and graphical plot of the log history of the select EMD



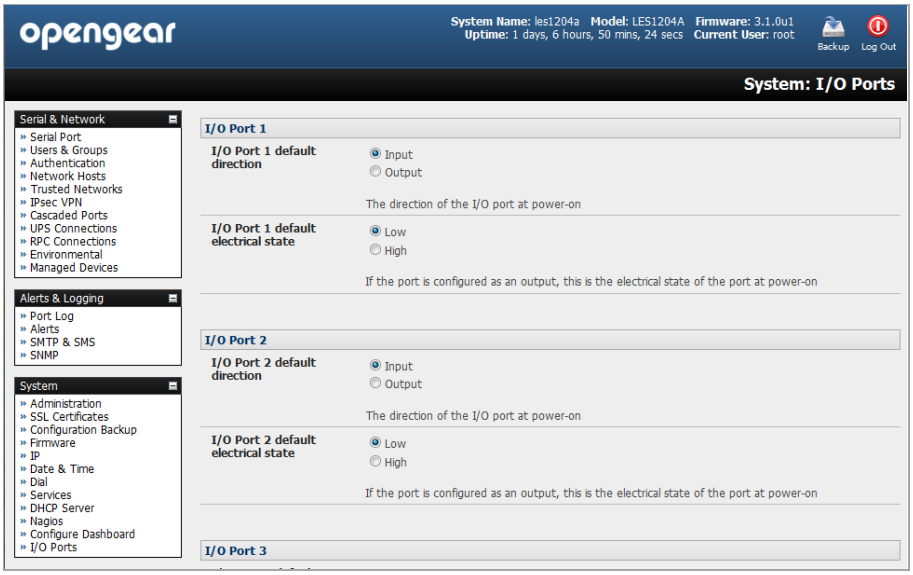
8.4 Digital I/O Ports

The ACM7000, ACM5004-2-I, ACM5508-2-I and ACM5504-5-G-I models have four digital interface ports which present on a green connector block on the side of the unit:

- *DIO1* and *DIO2* are two TTL level digital I/O ports (5V max @ 20mA)
- *OUT1* and *OUT2* are two "High-Voltage" digital Output ports (>5V to <= 30V @100mA)

The I/O ports are configured via the I/O port page which is found under the system menu. Each port can be configured with a default direction and state.

➢ Select the **System: I/O Ports** menu



8.4.1 Digital I/O Output Configuration

Each of the two digital I/O ports (DIO1 and DIO2) can be configured as an *Input* or *Output* port. To use them as digital outputs first configure the port direction on the **System: I/O Ports** menu page.

The DIO1 and DIO2 pins are current limited by the chip to 20mA and accept 5V levels – so they cannot drive a relay etc.

Alternately you can change the output states using the *ioc* command line utility. The following text is the usage message from the *ioc* usage:

ioc: digital io-port controller:

-p pin_num pin number (1 to 4)

-d	pin_dir	pin direction (0 = output 1 = input)
-v	pin_val	pin electrical value in output mode (0 = low 1 = high)
-r		reset pins to all inputs and low
-g		displays the pin directions and current values
-l		load pin configuration from *configlity*

For example, to set pin 1 to a low output, type:

ioc -p 1 -d 0 -v 0

To pulse one of these outputs, use a script like the following:

ioc -p 1 -d 0 -v 1
sleep 1
ioc -p 1 -d 0 -v 0
This will set the output high for 1 second, then return it to low (assuming the initial state is low)

8.4.2 Digital I/O Input Configuration

When either of the two digital I/O (DIO1 & DIO2) outlets is configured as an *Input* on the **System: I/O Ports**, it can be used to monitor the current status of any attached sensor.

When configured as inputs (and this is the factory default) these first two ports are notionally attached to an internal EMD. So to configure them as alarms, go to the Environmental page and edit and enable the Internal EMD.

Also the low voltage circuits in DIO1 and DIO2 should not be wired to voltages greater than 5V DC.

Alternately these input ports can be monitored using the *ioc* command line utility (as detailed in the previous section

8.4.3 High Voltage Outputs

OUT1 and OUT2 (internally DIO3 & DIO4) outlets are wired as high voltage outputs. The way these outputs are expected to be used is to pull a power connected line to ground (i.e. the OUT1 and OUT2 transistors are open collector).

The I/O port header includes a 12v reference line (VIN) which can be used to detect the line state change.

For example, to light a 12v LED using the high voltage outputs, connect the positive leg of the LED to the 12v reference, and the negative leg to output pin 4. Due to the way that the I/O port is connected internally, the output has to be set "high" to pull the output to ground.

The following command will switch on the led:

ioc -p 4 -d 0 -v 1

OUT1 and OUT2 transistors can operate with a supply of >5V to <= 30V @100mA. This means to drive a relay circuit you must guarantee it doesn't provide more than 100mA when set to 1.

8.4.4 DIO SNMP status

There is a SNMP status table (with V3.9 and later) which reports on the status of the digital IO ports. The table OID is OG-STATUSv2-MIB::ogEmdDioTable. Performing an *snmpwalk* on this table on a *console server* with DIO produces something like (will vary depending on device status):

```
$ snmpwalk -v2c -c public -M $MIBSDIR -m ALL t5:161 1.3.6.1.4.1.25049.16.5
OG-STATUS-MIB::ogDioStatusName.1 = STRING: DIO 1
OG-STATUS-MIB::ogDioStatusName.2 = STRING: DIO 2
OG-STATUS-MIB::ogDioStatusName.3 = STRING: DIO 3
OG-STATUS-MIB::ogDioStatusName.4 = STRING: DIO 4
OG-STATUS-MIB::ogDioStatusType.1 = INTEGER: ttlInputOutput(0)
OG-STATUS-MIB::ogDioStatusType.2 = INTEGER: ttlInputOutput(0)
OG-STATUS-MIB::ogDioStatusType.3 = INTEGER: highVoltageOutput(1)
OG-STATUS-MIB::ogDioStatusType.4 = INTEGER: highVoltageOutput(1)
```

```
OG-STATUS-MIB::ogDioStatusDirection.1 = INTEGER: input(1)
OG-STATUS-MIB::ogDioStatusDirection.2 = INTEGER: input(1)
OG-STATUS-MIB::ogDioStatusDirection.3 = INTEGER: input(1)
OG-STATUS-MIB::ogDioStatusDirection.4 = INTEGER: input(1)
OG-STATUS-MIB::ogDioStatusState.1 = INTEGER: low(0)
OG-STATUS-MIB::ogDioStatusState.2 = INTEGER: high(1)
OG-STATUS-MIB::ogDioStatusState.3 = INTEGER: high(1)
OG-STATUS-MIB::ogDioStatusState.4 = INTEGER: high(1)
OG-STATUS-MIB::ogDioStatusCounter.1 = Counter64: 0
OG-STATUS-MIB::ogDioStatusCounter.2 = Counter64: 0
OG-STATUS-MIB::ogDioStatusCounter.3 = Counter64: 0
OG-STATUS-MIB::ogDioStatusCounter.4 = Counter64: 0
OG-STATUS-MIB::ogDioStatusTriggerMode.1 = INTEGER: risingFallingEdge(3)
OG-STATUS-MIB::ogDioStatusTriggerMode.2 = INTEGER: risingFallingEdge(3)
OG-STATUS-MIB::ogDioStatusTriggerMode.3 = INTEGER: risingFallingEdge(3)
OG-STATUS-MIB::ogDioStatusTriggerMode.4 = INTEGER: risingFallingEdge(3)
```

AUTHENTICATION

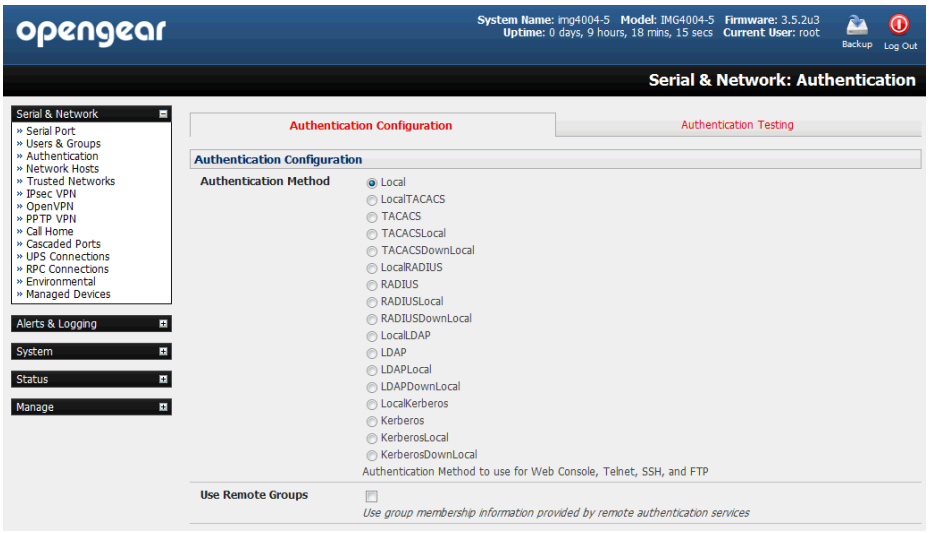
The *console server* platform is a dedicated Linux computer, and it embodies a myriad of popular and proven Linux software modules for networking, secure access (OpenSSH) and communications (OpenSSL) and sophisticated user authentication (PAM, RADIUS, TACACS+, Kerberos and LDAP).

- This chapter details how the *Administrator* can use the Management Console to establish remote AAA authentication for all connections to the *console server* and attached serial and network host devices
- This chapter also covers establishing a secure link to the Management Console using HTTPS and using OpenSSL and OpenSSH for establishing secure Administration connection to the *console server*

More details on RSA SecurID and working with Windows IAS can be found on the online FAQs.

9.1 Authentication Configuration

Authentication can be performed locally, or remotely using an LDAP, Radius, Kerberos or TACACS+ authentication server. The default authentication method for the *console server* is Local.



Any authentication method that is configured will be used for authentication of any user who attempts to log in through Telnet, SSH or the Web Manager to the *console server* and any connected serial port or network host devices.

The *console server* can be configured to the default (**Local**) or an alternate authentication method (**TACACS**, **RADIUS**, **LDAP** or **Kerberos**) with the option of a selected order in which local and remote authentication is to be used:

Local *TACACS /RADIUS/LDAP/Kerberos*: Tries local authentication first, falling back to remote if local fails

***TACACS /RADIUS/LDAP/Kerberos* Local**: Tries remote authentication first, falling back to local if remote fails

***TACACS /RADIUS/LDAP/Kerberos* Down Local**: Tries remote authentication first, falling back to local if the remote authentication returns an error condition (e.g. the remote authentication server is down or inaccessible)

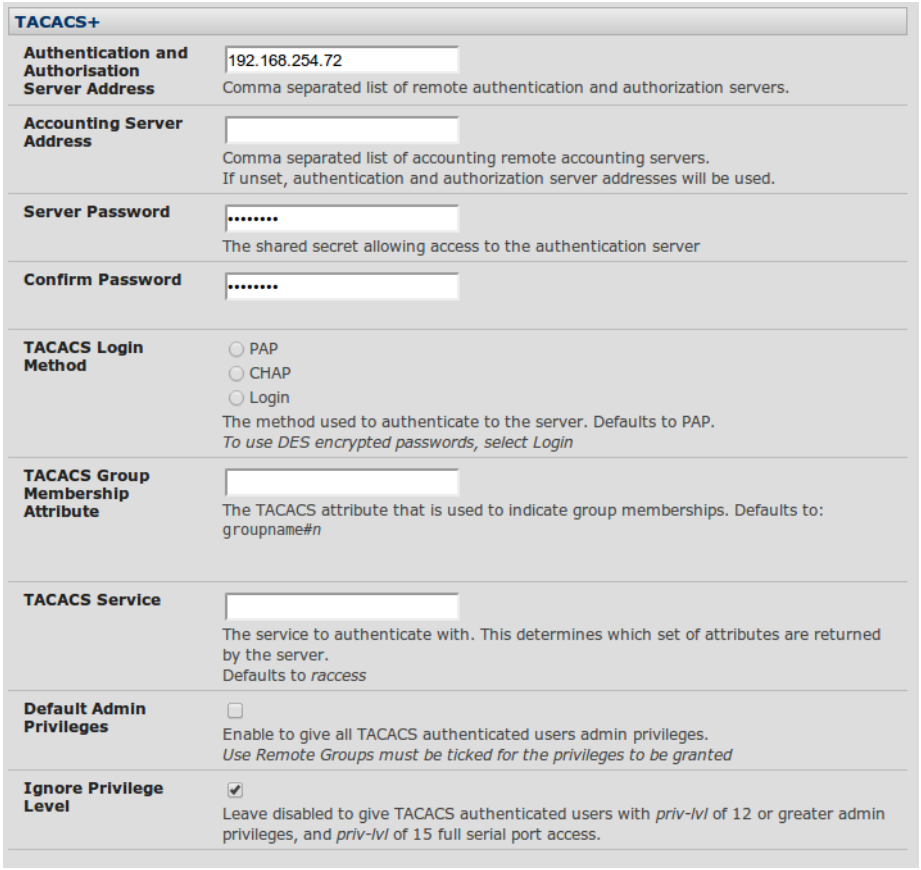
9.1.1 Local authentication

- Select **Serial and Network: Authentication** and check **Local**
- Click **Apply**

9.1.2 TACACS authentication

Perform the following procedure to configure the TACACS+ authentication method to be used whenever the *console server* or any of its serial ports or hosts is accessed:

- Select **Serial and Network: Authentication** and check **TACAS** or **LocalTACACS** or **TACACSLocal** or **TACACSDownLocal**



- Enter the **Server Address** (IP or host name) of the remote Authentication/Authorization server. Multiple remote servers may be specified in a comma separated list. Each server is tried in succession.
- In addition to multiple remote servers you can also enter for separate lists of Authentication/Authorization servers and Accounting servers. If no Accounting servers are specified, the Authentication/Authorization servers are used instead.
- Enter and confirm the **Server Password**. Then select the method to be used to authenticate to the server (defaults to **PAP**). To use DES encrypted passwords, select **Login**
- If required enter the **TACACS Group Membership Attribute** that is to be used to indicate group memberships (defaults to *groupname#n)*
- If required, specify **TACACS Service** to authenticate with. This determines which set of attributes are returned by the server (defaults to *raccess*)
- If required, check **Default Admin Privileges** to give all TACAS+ authenticated users *admin* privileges. **Use Remote Groups** must also be ticked for these privileges to be granted
- The TACACS **Privilege Level** feature only applies to TACACS remote authentication. When **Ignore Privilege Level** is enabled, the *priv-lvl* setting for all of the users defined on the TACACS AAA server will be ignored

Note: An Opengear device normally interprets a user with a TACACS *priv-lvl* of 12 or above as an admin user. There is a special case where a user with a *priv-lvl* of 15 is also given access to all configured serial ports. When the **Ignore Privilege Level** option is enabled (i.e. checked in the UI) there are no escalations of privileges based on the *priv-lvl* value from the TACACS server.

Also note that if the only thing configured for one or more TACACS users is the *priv-lvl* (e.g. no specific port access or group memberships set), then enabling this feature will revoke access to the console server for those users as the they won't be a member of any groups, even if the Retrieve Remote groups option in the Authentication menu is enabled.

- Click **Apply.** TACAS+ remote authentication will now be used for all user access to *console server* and serially or network attached devices

TACACS+ The Terminal Access Controller Access Control System (TACACS+) security protocol is a recent protocol developed by Cisco. It provides detailed accounting information and flexible administrative control over the authentication and authorization processes. TACACS+ allows for a single access control server (the TACACS+ daemon) to provide authentication, authorization, and accounting services independently. Each service can be tied into its own database to take advantage of other services available on that server or on the network, depending on the capabilities of the daemon. There is a draft RFC detailing this protocol. Further information on configuring remote TACACS+ servers can be found at the following sites:

http://www.cisco.com/en/US/tech/tk59/technologies_tech_note09186a0080094e99.shtml

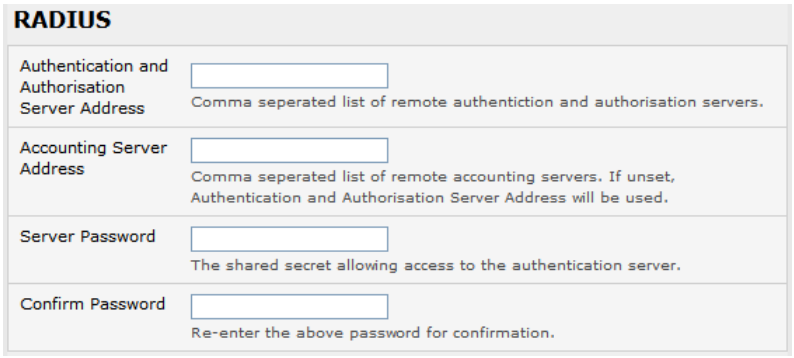
http://www.cisco.com/en/US/products/sw/secursw/ps4911/products_user_guide_chapter09186a00800eb6d6.html

http://cio.cisco.com/univercd/cc/td/doc/product/software/ios113ed/113ed_cr/secur_c/scprt2/sctplus.htm

9.1.3 RADIUS authentication

Perform the following procedure to configure the RADIUS authentication method to be used whenever the *console server* or any of its serial ports or hosts is accessed:

- Select **Serial and Network: Authentication** and check **RADIUS** or **LocalRADIUS** or **RADIUSLocal** or **RADIUSDownLocal**



- Enter the **Server Address** (IP or host name) of the remote Authentication / Authorization server. Multiple remote servers may be specified in a comma separated list. Each server is tried in succession
- In addition to multiple remote servers you can also enter for separate lists of Authentication/Authorization servers and Accounting servers. If no Accounting servers are specified, the Authentication/Authorization servers are used instead
- Enter the **Server Password**
- Click **Apply.** RADIUS remote authentication will now be used for all user access to *console server* and serially or network attached devices

RADIUS The Remote Authentication Dial-In User Service (RADIUS) protocol was developed by Livingston Enterprises as an access server authentication and accounting protocol. The RADIUS server can support a

variety of methods to authenticate a user. When it is provided with the username and original password given by the user, it can support PPP, PAP or CHAP, UNIX login, and other authentication mechanisms. Further information on configuring remote RADIUS servers can be found at the following sites:

http://www.microsoft.com/technet/prodtechnol/windowsserver2003/library/DepKit/d4fe8248-eecd-49e4-88f6-9e304f97fefc.mspx

http://www.cisco.com/en/US/tech/tk59/technologies_tech_note09186a00800945cc.shtml

http://www.freeradius.org/

9.1.4 LDAP authentication

With firmware V.3.11 and later LDAP authentication now supports OpenLDAP servers, using the Posix style schema for user and group definitions.

Performing simple authentication against any LDAP server (AD or OpenLDAP) is fairly straight forward, as they both follow the common LDAP standards and protocols. The harder part is configuring how to get the extra data about the users (the groups they are in, etc).

On an Opengear device, we may be configured to look at group information from an LDAP server for authentication and authorization. This group information is potentially stored in a number of different ways. Active Directory has one method, and OpenLDAP has two other methods:

- Active Directory: Each entry for a user will have multiple 'memberOf' attributes. Each 'memberOf' value is the full DN of the group they belong to. (The entry for the user will be of objectClass "user")
- OpenLDAP / Posix: Each entry for a user must have a 'gidNumber' attribute. This will be an integer value, which is the user's primary group (eg. mapping to the /etc/passwd file, with the group ID field). To determine which group this is, we must search for an entry in the directory that has that group ID, which will give us the group name. (The users are of objectClass "posixAccount", and the groups are of objectClass "posixGroup").
- OpenLDAP / Posix: Each group entry in the group tree (of objectClass 'posixGroup') may have multiple 'memberUid' attributes. These represent secondary groups (eg. mapping to the /etc/groups file). Each attribute would contain a username.

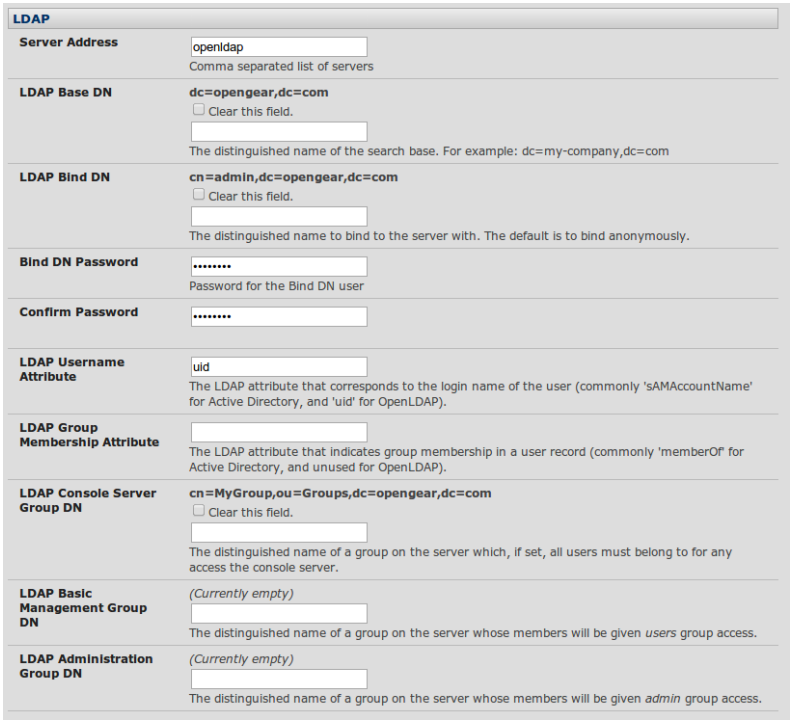
To cater for all these possibilities, the *pam_ldap* module has been modified to do group lookups for each of these three styles. This allows us to have a relatively 'generic' configuration, and not be concerned with how the LDAP directory is set up.

There are only two parameters that need to be configured based on what the user wishes to look up: these are the LDAP username and group membership attributes.

To clarify to the user what parameters to use, the descriptions for these fields have been updated to prompt the user for common or likely attributes. For example, the two configuration fields have descriptions as follows:

LDAP Username Attribute: The LDAP attribute that corresponds to the login name of the user (commonly 'sAMAccountName' for Active Directory, and 'uid' for OpenLDAP).

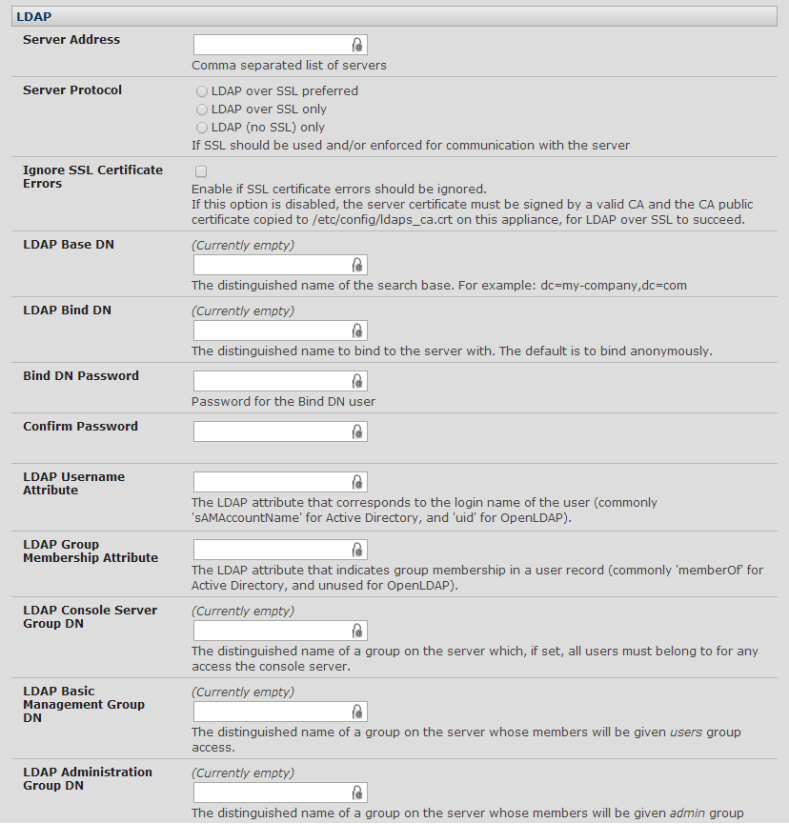
LDAP Group Membership Attribute: The LDAP attribute that indicates group membership in a user record (commonly 'memberOf' for Active Directory, and unused for OpenLDAP).



Note The *libldap* library is particularly fussy about ensuring SSL connections are using certificates signed by a trusted CA so it is often not easy to set up a connection to an LDAP server using SSL. Refer to" https://opengear.zendesk.com/entries/29959515-LDAP-over-SSL

Perform the following procedure to configure the LDAP authentication method to be used whenever the *console server* or any of its serial ports or hosts is accessed:

- Select **Serial and Network: Authentication** and check **LDAP** or **LocalLDAP** or **LDAPLocal** or **LDAPDownLocal**



- Enter the **Server Address** (IP or host name) of the remote Authentication server. Multiple remote servers may be specified in a comma separated list. Each server is tried in succession.
- Check the **Server Protocol** box to select if SSL is to be used and/or enforced for communications with the LDAP server. Console servers running firmware v3.11 and above offer three options for LDAPS (LDAP over SSL):
  - **LDAP over SSL preferred** will attempt to use SSL for authentication, but if it fails it will fall back to LDAP without SSL. As an example LDAP over SSL may fail due to certificate errors or the LDAP server not be contactable on the LDAPS port etc
  - **LDAP over SSL only**: this setting will configure the Opengear device to only accept LDAP over SSL. If LDAP over SSL fails you will only be able to log into the console server as root
  - **LDAP (no SSL) only**: this setting will configure the Opengear device to only accept LDAP without SSL. If LDAP without SSL fails you will only be able to log into the console server as root
- The '**Ignore SSL Certificate Error**' check box enables you to ignore SSL certificate errors - allowing LDAP over SSL to work regardless of these errors. This allows you to use any certificate, self-signed or otherwise, on the LDAP server without having to install any certificates on the *console server*. If this setting is not checked, then you must install the CA (certificate authority) certificate with which the LDAP server's certificate was signed, onto the *console server*. For example, the LDAP server is serving with a certificate singed using the certificate 'myCA.crt'

Note: The certificate needs to be in CRT format and *myCA.crt* needs to be installed onto *console server* at '*/etc/config/ldaps_ca.crt*'. Also the file name must be *'ldaps_ca.crt*'. You need to copy the file to this location and file name manually using 'scp' or the like e.g.

scp /local/path/to/myCA.c
rt root@console_server:/etc/config/ldaps_ca.crt

- Enter the **Server Password**

- Click **Apply.** LDAP remote authentication will now be used for all user access to *console server* and serially or network attached devices

LDAP The Lightweight Directory Access Protocol (LDAP) is based on the X.500 standard, but significantly simpler and more readily adapted to meet custom needs. The core LDAP specifications are all defined in RFCs. LDAP is a protocol used to access information stored in an LDAP server. Further information on configuring remote RADIUS servers can be found at the following sites:

http://www.ldapman.org/articles/intro_to_ldap.html

http://www.ldapman.org/servers.html

http://www.linuxplanet.com/linuxplanet/tutorials/5050/1/

http://www.linuxplanet.com/linuxplanet/tutorials/5074/4/

9.1.5 RADIUS/TACACS user configuration

Users may be added to the local *console server* appliance. If they are not added and they log in via remote AAA, a user will be added for them. This user will not show up in the Opengear configurators unless they are specifically added, at which point they are transformed into a completely local user. The newly added user must authenticate off of the remote AAA server, and will have no access if it is down.

If a local user logs in, they may be authenticated / authorized from the remote AAA server, depending on the chosen priority of the remote AAA. A local user's authorization is the union of local and remote privileges.

Example 1:

User Tim is locally added, and has access to ports 1 and 2. He is also defined on a remote TACACS server, which says he has access to ports 3 and 4. Tim may log in with either his local or TACACS password, and will have access to ports 1 through 4. If TACACS is down, he will need to use his local password, and will only be able to access ports 1 and 2.

Example 2:

User Ben is only defined on the TACACS server, which says he has access to ports 5 and 6. When he attempts to log in a new user will be created for him, and he will be able to access ports 5 and 6. If the TACACS server is down he will have no access.

Example 3:

User Paul is defined on a RADIUS server only. He has access to all serial ports and network hosts.

Example 4:

User Don is locally defined on an appliance using RADIUS for AAA. Even if Don is also defined on the RADIUS server he will only have access to those serial ports and network hosts he has been authorized to use on the appliance.

If a “no local AAA” option is selected, then root will still be authenticated locally.

Remote users may be added to the admin group via either RADIUS or TACACS. Users may have a set of authorizations set on the remote TACACS server. Users automatically added by RADIUS will have authorization for all resources, whereas those added locally will still need their authorizations specified.

LDAP has not been modified, and will still need locally defined users.

Note To interact with RADIUS, TACACS+ and LDAP with *console server* firmware pre 2.4.2 you must also set up the user accounts on the local *console server*. All resource authorizations must be added to the local appliance. With this release if remote AAA is selected, it is used for password checking only. Root is always authenticated locally. Any changes to PAM configurations will be destroyed next time the authentication configurator is run

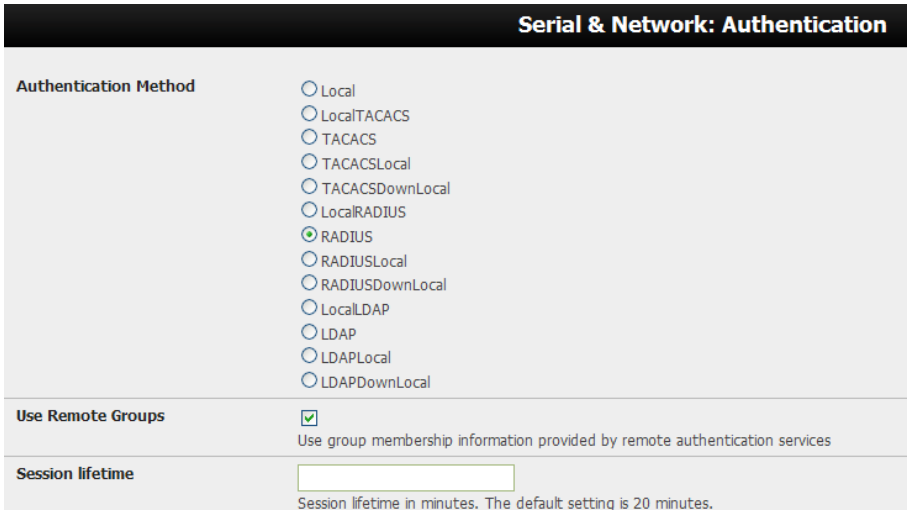
9.1.6 Group support with remote authentication

All *console servers* allow remote authentication via RADIUS, LDAP and TACACS+. With Firmware V3.2 and later, RADIUS and LDAP can provide additional restrictions on user access based on group information or membership. For example, with remote group support, users can belong to a local group that has been setup to have restricted access to serial ports, network hosts and managed devices.

Remote authentication with group support works by matching a local group name with a remote group name provided by the authentication service. If the list of remote group names returned by the authentication service matches any local group names, the user is given permissions as configured in the local groups.

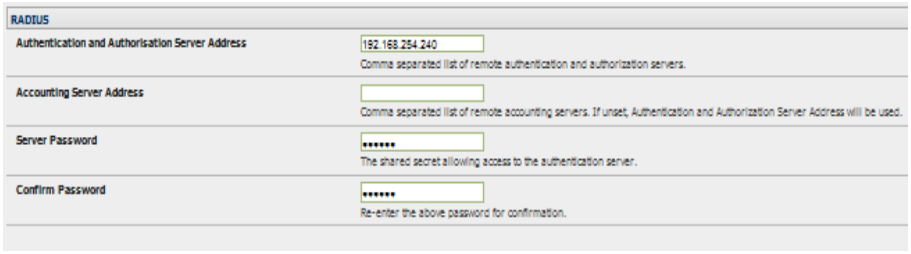
To enable group support to be used by remote authentication services:

- Select **Serial & Network: Authentication**
- Select the relevant **Authentication Method**
- Check the **Use Remote Groups** button



9.1.7 Remote groups with RADIUS authentication

- Enter the RADIUS **Authentication and Authorization Server Address** and **Server Password**
- Click Apply.



- Edit the Radius user's file to include group information and restart the Radius server

When using RADIUS authentication, group names are provided to the *console server* using the Framed-Filter-Id attribute. This is a standard RADIUS attribute, and may be used by other devices that authenticate via RADIUS.

To interoperate with other devices using this field, the group names can be added to the end of any existing content in the attribute, in the following format:

:group_name=testgroup1,users:

The above example sets the remote user as a member of testgroup1 and users if groups with those names exist on the *console server*. Any groups which do not exist on the *console server* are ignored.

When setting the Framed-Filter-Id, the system may also remove the leading colon for an empty field. To work around this, add some dummy text to the start of the string. For example:

dummy:group_name=testgroup1,users:

- If no group is specified for a user, for example AmandaJones, then the user will have no User Interface and serial port access but limited console access
- Default groups available on the *console server* include '*admin'* for administrator access and 'users' for general user access

TomFraser	Cleartext-Password := "FraTom70"
	Framed-Filter-Id=":group_name=admin:"
AmandaJones	Cleartext-Password := "JonAma83"
FredWhite	Cleartext-Password := "WhiFre62"
	Framed-Filter-Id=":group_name=testgroup1,users:"
JanetLong	Cleartext-Password := "LonJan57"
	Framed-Filter-Id=":group_name=admin:"

- Additional local groups such as testgroup1 can be added via **Users & Groups: Serial & Network**

9.1.8 Remote groups with LDAP authentication

Unlike RADIUS, LDAP has built in support for group provisioning, which makes setting up remote groups easier. The console server will retrieve a list of all the remote groups that the user is a direct member of, and compare their names with local groups on the *console server*.

Note: Any spaces in the group name will be converted to underscores.

For example, in an existing Active Directory setup, a group of users may be part of the "*UPS Admin*" and "*Router Admin*" groups. On the *console server*, these users will be required to have access to a group "*Router_Admin*", with access to port 1 (connected to the router), and another group "*UPS_Admin*", with access to port 2 (connected to the UPS). Once LDAP is setup, users that are members of each group will have the appropriate permissions to access the router and UPS.

Currently, the only LDAP directory service that supports group provisioning is Microsoft Active Directory. Support is planned for OpenLDAP at a later time.

To enable group information to be used with an LDAP server:

- Complete the fields for standard LDAP authentication including LDAP Server Address, Server Password, LDAP Base DN, LDAP Bind DN and LDAP User Name Attribute
- Enter memberOf for **LDAP Group Membership Attribute** as group membership is currently only supported on Active Directory servers
- If required, enter the group information for **LDAP Console Server Group DN** and/or **LDAP Administration Group DN**

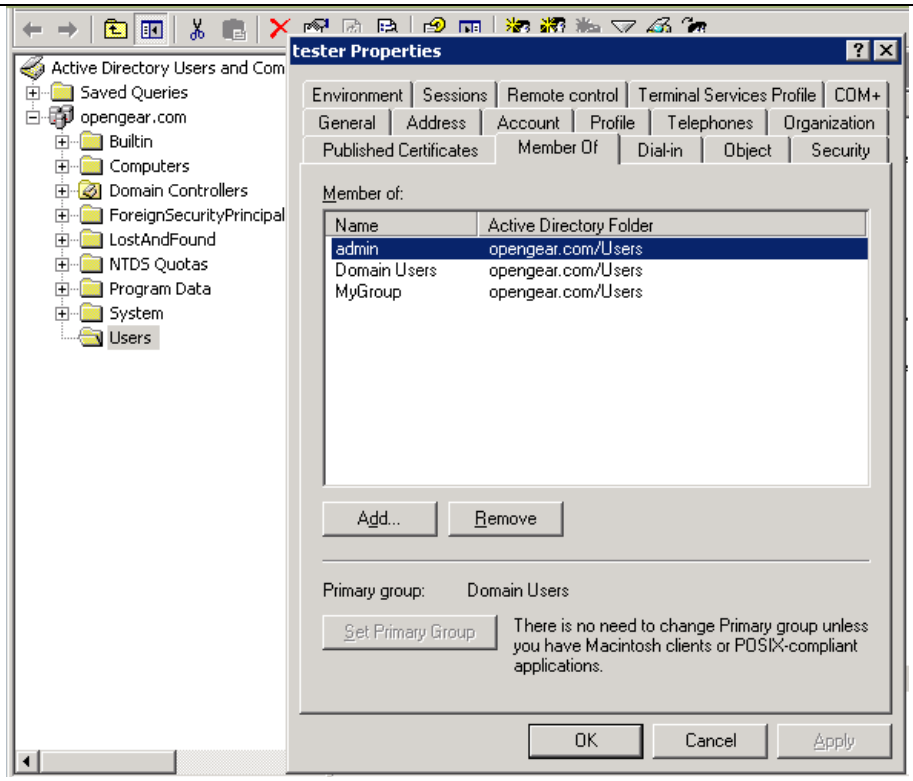
A user must be a member of the LDAP Console Server Group DN group in order to gain access to the console and user interface. For example, the user must be a member of 'MyGroup' on the Active Server to gain access to the *console server.*

Additionally, a user must be a member of the LDAP Administration Group DN in order to gain administrator access to the *console server*. For example, the user must be a member of 'AdminGroup' on the Active Server to receive administration privileges on the *console server*.

- Click Apply.

- Ensure the LDAP service is operational and group names are correct within the Active Directory

Note When you are using remote groups with LDAP remote auth, you need to have corresponding local groups on the console server BUT where the LDAP group names can contain upper case and space characters the local group name on the console server must be all lower case and the spaces replaced with underscrores. For example, a remote group on the LDAP server may be **'My Ldap Access Group**' needs a corresponding local group on the console server called **'my_ldap_access_group**' (both without the single quotes). The local group on the console server must specify what the group member is granted access to for any group membership to be effective.



9.1.9 Remote groups with TACACS+ authentication

When using TACACS+ authentication, there are two ways to grant a remotely authenticated user privileges. The first is to set the priv-lvl and port attributes of the raccess service to 12, this is discussed further in section 9.2 of this document. Additionally or alternatively, group names can be provided to the console server using the groupname custom attribute of the raccess service.

An example Linux tac-plus config snippet might look like:

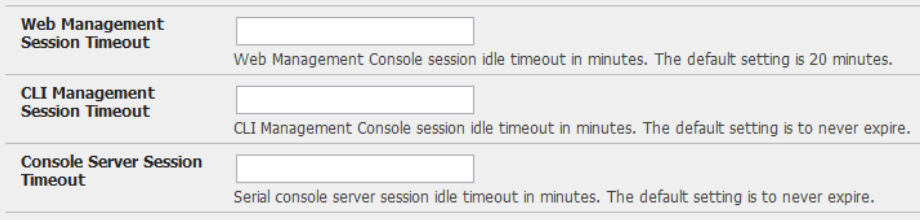
```
user = myuser {
        service = raccess {
                groupname="users"
                groupname1="routers"
                groupname2="dracs"
        }
}
```

You may also specify multiple groups in one comma-delimited, e.g. *groupname="users,routers,dracs"* but be aware that the maximum length of the attribute value string is 255 characters.

To use an attribute name other than *"groupname",* set Authentication -> TACACS+ -> TACACS Group Membership Attribute.

9.1.10 Idle timeout

You can specify amount of time in minutes the *console server* waits before it terminates an idle ssh, pmshell or web connection.



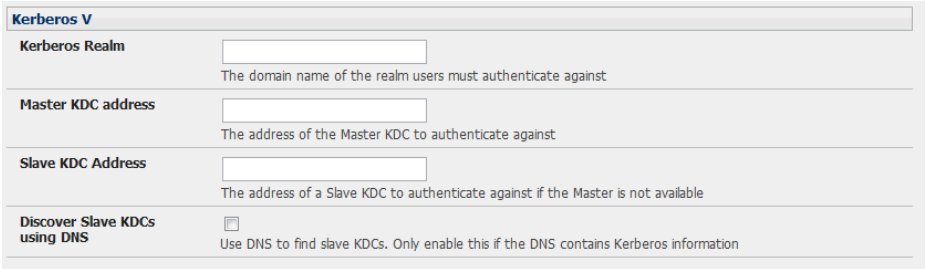
- Select **Serial and Network: Authentication**
- **Web Management Session Timeout** specifies the browser console session idle timeout in minutes. The default setting is 20 minutes
- **CLI Management Session Timeout** specifies the ssh console session idle timeout in minutes. The default setting is to never expire
- **Console Server Session Timeout** specifies the pmshell serial console server session idle timeout in minutes. The default setting is to never expire

9.1.11 Kerberos authentication

The Kerberos authentication can be used with UNIX and Windows (Active Directory) Kerberos servers. This form of authentication does not provide group information, so a local user with the same username must be created, and permissions set.

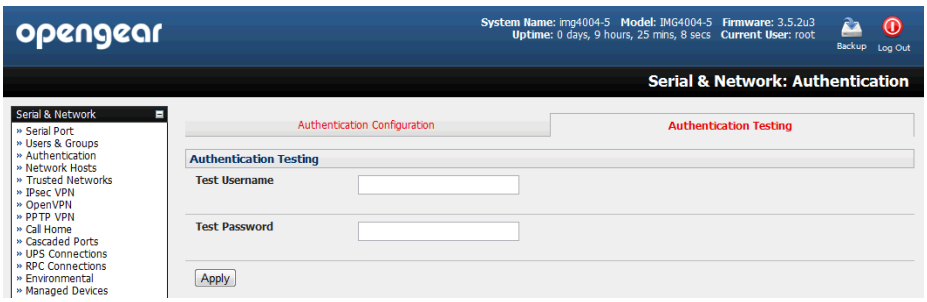
Note: Kerberos is very sensitive to time differences between the Key Distribution Center (KDC) authentication server and the client device. Please make sure that NTP is enabled, and the time zone is set correctly on the *console server*.

When authenticating against Active Directory, the Kerberos Realm will be the domain name, and the Master KDC will be the address of the primary domain controller.



9.1.12 Authentication testing

The Authentication Testing tab (firmware V3.5.2u3 and later) enables the connection to the remote authentication server to be tested.



9.2 PAM (Pluggable Authentication Modules)

The *console server* supports RADIUS, TACACS+ and LDAP for two-factor authentication *via* PAM (Pluggable Authentication Modules). PAM is a flexible mechanism for authenticating users. Nowadays a number of new ways of authenticating users have become popular. The challenge is that each time a new authentication scheme is developed; it requires all the necessary programs (login, ftpd *etc.*) to be rewritten to support it.

PAM provides a way to develop programs that are independent of authentication scheme. These programs need "authentication modules" to be attached to them at run-time in order to work. Which authentication module is to be attached is dependent upon the local system setup and is at the discretion of the local *Administrator*.

The *console server* family supports PAM to which we have added the following modules for remote authentication:

RADIUS - pam_radius_auth (http://www.freeradius.org/pam_radius_auth/)

TACACS+ - pam_tacplus (http://echelon.pl/pubs/pam_tacplus.html)

LDAP - pam_ldap (http://www.padl.com/OSS/pam_ldap.html)

Further modules can be added as required.

Changes may be made to files in /etc/config/pam.d / which will persist, even if the authentication configurator is run.

- Users added on demand:

  When a user attempts to log in, but does not already have an account on the *console server*, a new user account will be created. This account will have no rights, and no password set. They will not appear in the Opengear configuration tools.

Automatically added accounts will not be able to log in if the remote servers are unavailable
.

- Admin rights granted over AAA:

  Users may be granted *Administrator* rights via networked AAA. For TACACS a priv-lvl of 12 of above indicates an administrator. For RADIUS, administrators are indicated via the Framed Filter ID. (See the example configuration files below for example)

- Authorization via TACACS, LDAP or RADIUS for using remote groups:

  See section 9.1.6 of this document

- Authorization via TACACS for both serial ports and host access:

  Permission to access resources may be granted via TACACS by indicating an Opengear Appliance and a port or networked host the user may access. (See the example configuration files below for example.)

TACACS Example:

```
user = tim {
  service = raccess {
    priv-lvl = 11
```

```
        port1 = sd4001/port02
    }
    global = cleartext mit
}
```

RADIUS Example:

```
paul    Cleartext-Password := "luap"
        Service-Type = Framed-User,
        Fall-Through = No,
        Framed-Filter-Id=":group_name=admin:"
```

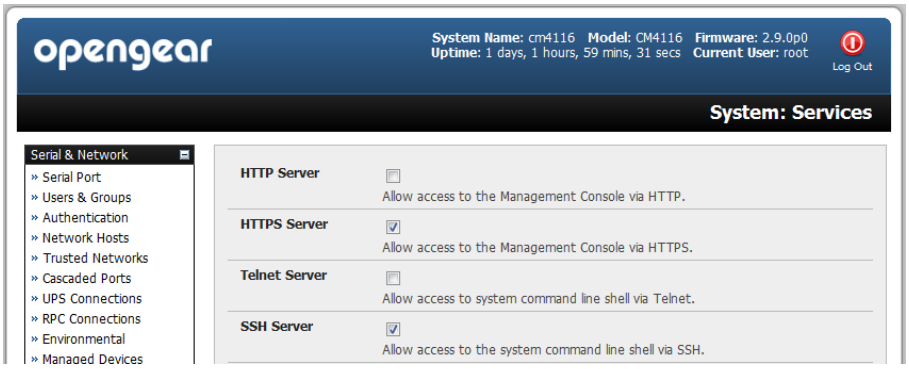
The list of groups may include any number of entries separated by a comma. If the admin group is included, the user will be made an *Administrator*.

If there is already a Framed-Filter-Id simply add the list of *group_names* after the existing entries, including the separating colon ":".

9.3 SSL Certificate

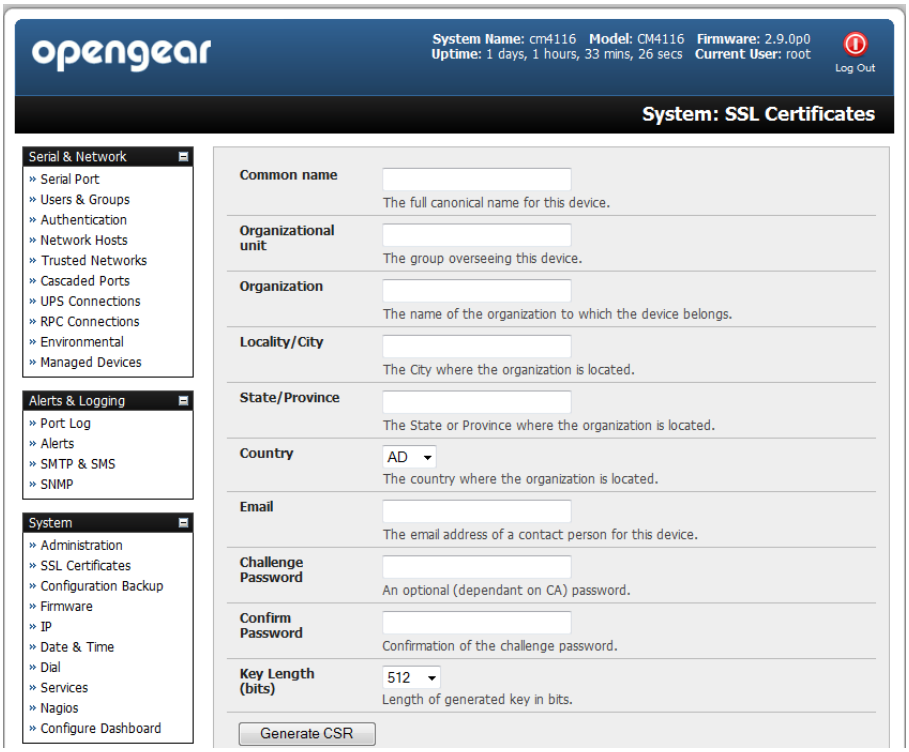
The *console server* uses the Secure Socket Layer (SSL) protocol for encrypted network traffic between itself and a connected user. During the connection establishment the *console server* has to expose its identity to the user's browser using a cryptographic certificate. The default certificate that comes with the *console server* device upon delivery is for testing purpose only and should not be relied on for secured global access.

The System Administrator should not rely on the default certificate as the secured global access mechanism for use through Internet



- Activate your preferred browser and enter https:/ / *IP address.* Your browser may respond with a message that verifies the security certificate is valid but notes that it is not necessarily verified by a certifying authority. To proceed you need to click *yes* if you are using Internet Explorer or select *accept this certificate permanently* (or *temporarily*) if you are using Mozilla Firefox.
- You will then be prompted for the *Administrator* account and password as normal.

However it is recommended you generate and install a new base64 X.509 certificate that is unique for a particular *console server*.



To do this the *console server* must be enabled to generate a new cryptographic key and the associated Certificate Signing Request (CSR) that needs to be certified by a Certification Authority (CA). A certification authority verifies that you are the person who you claim you are, and signs and issues a SSL certificate to you. To create and install a SSL certificate for the *console server*:

- Select **System: SSL Certificate** and fill out the fields as explained below:

  Common name This is the network name of the *console server* once it is installed in the network (usually the fully qualified domain name). It is identical to the name that is used to access the *console server* with a web browser (without the "http://" prefix). In case the name given here and the actual network name differ, the browser will pop up a security warning when the *console server* is accessed using HTTPS

  Organizational Unit This field is used for specifying to which department within an organization the *console server* belongs

  Organization The name of the organization to which the *console server* belongs

  Locality/City The city where the organization is located

  State/Province The state or province where the organization is located

  Country The country where the organization is located. This is the two-letter ISO code, e.g. DE for Germany, or US for the USA. (Note: the country code has to be entered in CAPITAL LETTERS)

  Email The email address of a contact person that is responsible for the *console server* and its security

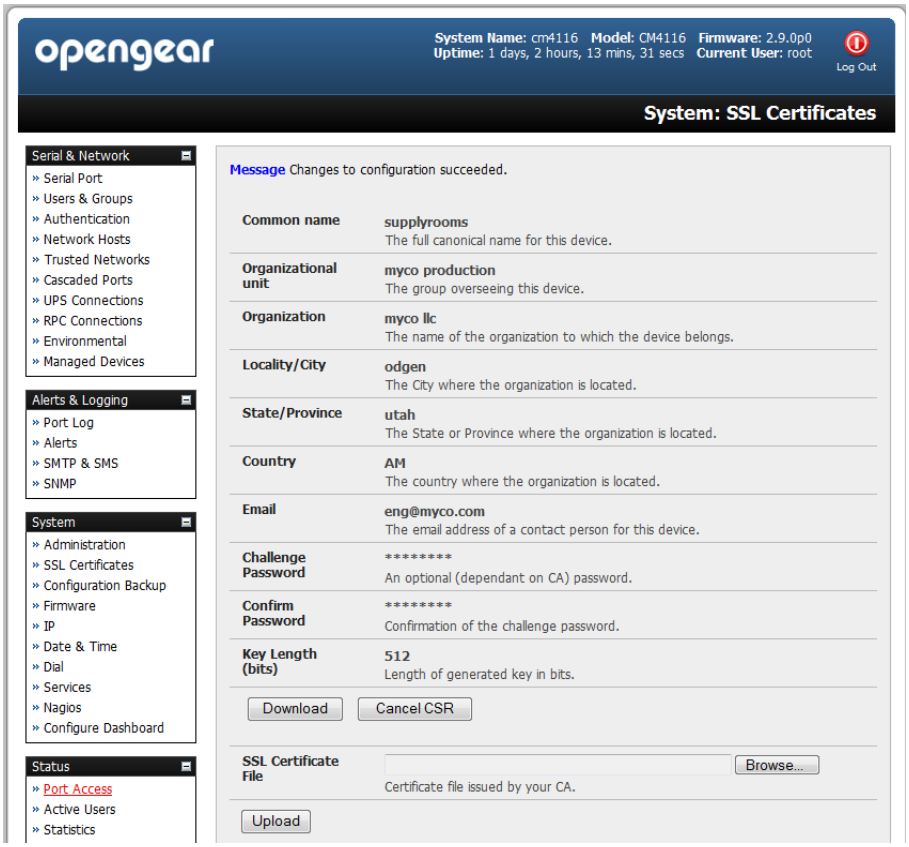
  Challenge Password Some certification authorities require a challenge password to authorize later changes on the certificate (e.g. revocation of the certificate). The minimal length of this password is 4 characters

  Confirm Challenge Password Confirmation of the Challenge Password

  Key length This is the length of the generated key in bits. 1024 Bits are supposed to be sufficient for most cases. Longer keys may result in slower response time of the *console server* during connection establishment

- Once this is done, click on the button **Generate CSR** which will initiate the Certificate Signing Request generation. The CSR can be downloaded to your administration machine with the **Download** button

- Send the saved CSR string to a Certification Authority (CA). for certification. You will get the new certificate from the CA after a more or less complicated traditional authentication process (depending on the CA)
- Upload the certificate to the *console server* using the **Upload** button as shown below



After completing these steps the *console server* has its own certificate that is used for identifying the *console server* to its users.

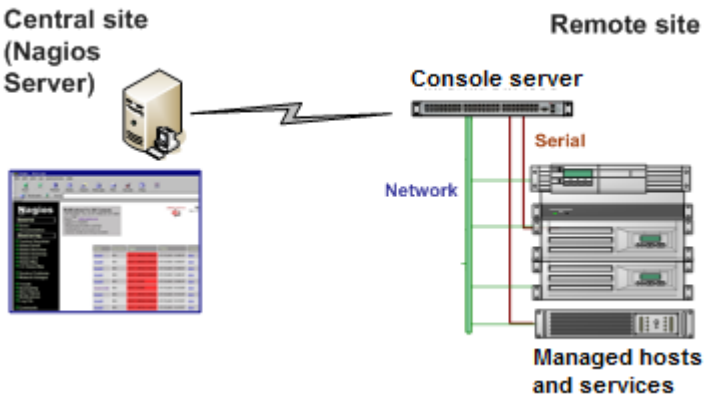
Note Information on issuing certificates and configuring HTTPS from the command line can be found in Chapter 15 - Advanced

NAGIOS INTEGRATION

Nagios is a powerful, highly extensible open source tool for monitoring network hosts and services. The core Nagios software package will typically be installed on a server or virtual server, the central Nagios server.

Console servers operate in conjunction with a central/upstream Nagios server to provide distributing monitoring of attached network hosts and serial devices. They embed the NSCA (Nagios Service Checks Acceptor) and NRPE (Nagios Remote Plug-in Executor) add-ons – this allows them to communicate with the central Nagios server, eliminating the need for a dedicated slave Nagios server at remote sites.

The *console server* products all support distributed monitoring. Even if distributed monitoring is not required, the *Console servers* can be deployed locally alongside the Nagios monitoring host server, to provide additional diagnostics and points of access to managed devices.



Note If you have an existing Nagios deployment, you may wish to use the *console server* gateways in a distributed monitoring server capacity only. If this case and you are already familiar with Nagios, skip ahead to section 10.3.

10.1 Nagios Overview

Nagios provides central monitoring of the hosts and services in your distributed network. Nagios is freely downloadable, open source software. This section offers a quick background of Nagios and its capabilities. A complete overview, FAQ and comprehensive documentation are available at: *http://www.nagios.org*

Nagios forms the core of many leading commercial system management solutions such as GroundWork: *http://www.groundworkopensource.com*

Nagios does take some time to install and configure however once it is up and running, it provides an outstanding network monitoring system. With Nagios you can:

- Display tables showing the status of each monitored server and network service in real time
- Use a wide range of freely available plug-ins to make detailed checks of specific services – e.g. don't just check a database is accepting network connections, check that it can actually validate requests and return real data
- Display warnings and send warning e-mails, pager or SMS alerts when a service failure or degradation is detected
- Assign contact groups who are responsible for specific services in specific time frames

10.2 Configuring Nagios distributed monitoring

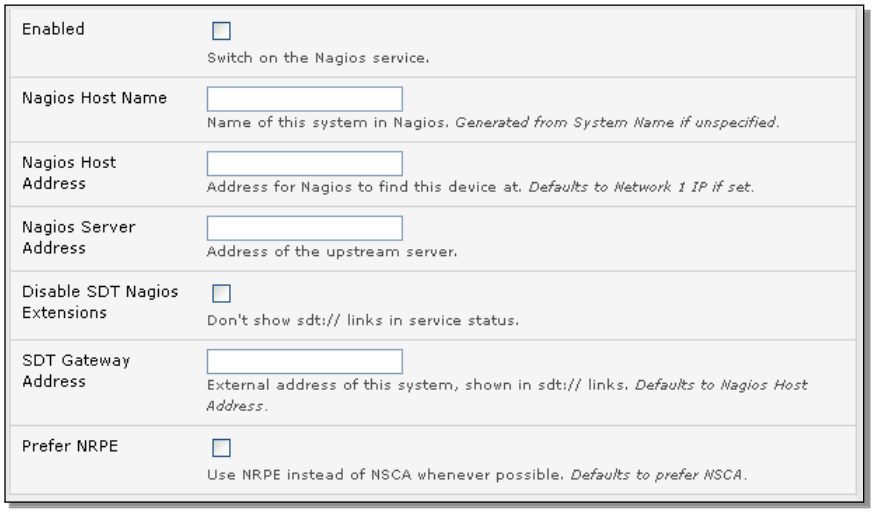
To activate the *console server* Nagios distributed monitoring:

- Nagios integration must be enabled and a path established to the central/upstream Nagios server

- If the *console server* is to periodically report on Nagios monitored services, then the NSCA client embedded in the *console server* must be configured – the NSCA program enables scheduled check-ins with the remote Nagios server and is used to send passive check results across the network to the remote server
- If the Nagios server is to actively request status updates from the *console server*, then the NRPE server embedded in the *console server* must be configured – the NRPE server is the Nagios daemon for executing plug-ins on remote hosts
- Each of the Serial Ports and each of the Hosts connected to the *console server* which are to be monitored must have Nagios enabled and any specific Nagios checks configured
- Lastly the central/upstream Nagios monitoring host must be configured

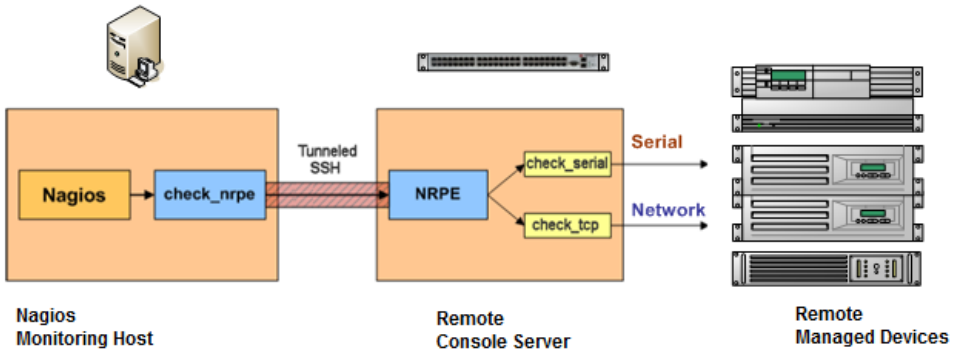
10.2.1 Enable Nagios on the console server

- Select **System: Nagios** on the *console server* Management Console and tick the Nagios service **Enabled**

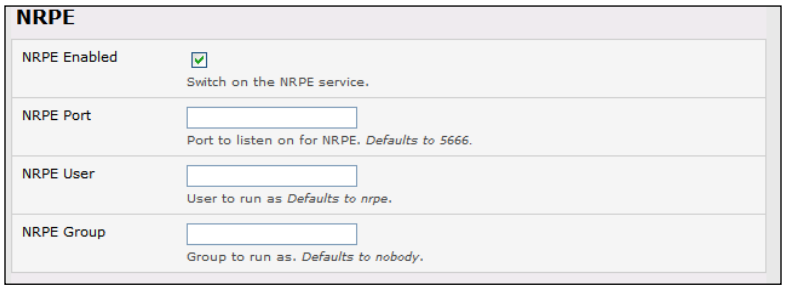


- Enter the **Nagios Host Name** that the *Console server* will be referred to in the Nagios central server – this will be generated from local System Name (entered in **System: Administration**) if unspecified
- In **Nagios Host Address** enter the IP address or DNS name that the upstream Nagios server will use to reach the *console server* – if unspecified this will default to the first network port's IP (*Network (1)* as entered in **System: IP**)
- In **Nagios Server Address** enter the IP address or DNS name that the *console server* will use to reach the upstream Nagios monitoring server
- Check the **Disable SDT Nagios Extensions** option if you wish to disable the SDT Connector integration with your Nagios server at the head end – this would only be checked if you want to run a vanilla Nagios monitoring
- If not, enter the IP address or DNS name the SDT Nagios clients will use to reach the *console server* in **SDT Gateway Address**
- When NRPE and NSCA are both enabled, NSCA is preferred method for communicating with the upstream Nagios server – check **Prefer NRPE** to use NRPE whenever possible (i.e. for all communication except for alerts)

10.2.2 Enable NRPE monitoring



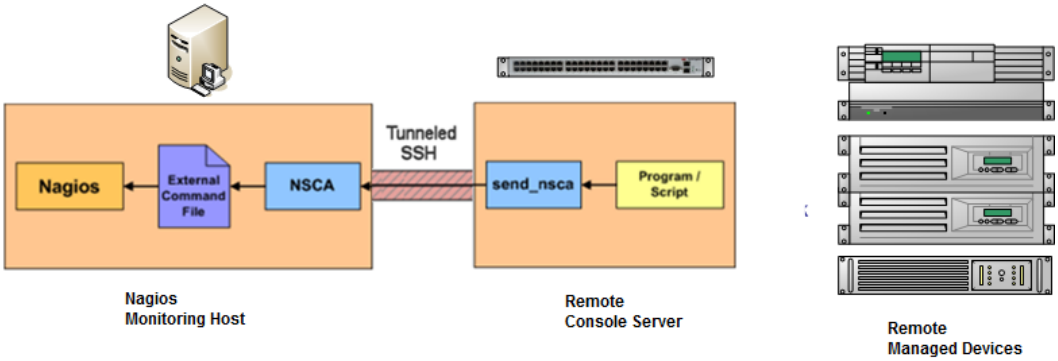
Enabling NRPE allows you to execute plug-ins (such as *check_tcp* and *check_ping*) on the remote *Console server* to monitor serial or network attached remote servers. This will offload CPU load from the upstream Nagios monitoring machine which is especially valuable if you are monitoring hundreds or thousands of hosts. To enable NRPE:



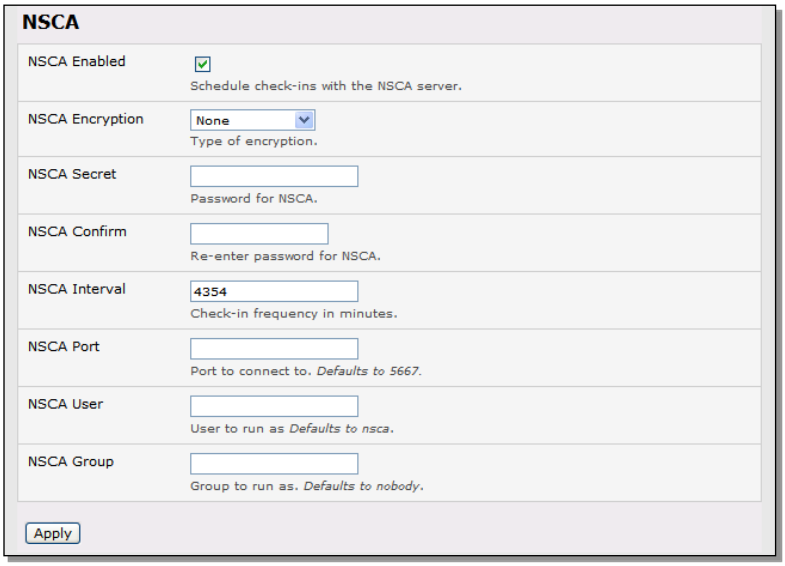
- Select **System: Nagios** and check **NRPE Enabled**
- Enter the details the user connection to the upstream Nagios monitoring server and again refer the sample Nagios configuration example below for details of configuring specific NRPE checks

By default the *console server* will accept a connection between the upstream Nagios monitoring server and the NRPE server with SSL encryption, without SSL, or tunneled through SSH. The security for the connection is configured at the Nagios server.

10.2.3 Enable NSCA monitoring



NSCA is the mechanism that allows you to send passive check results from the remote *console server* to the Nagios daemon running on the monitoring server. To enable NSCA:

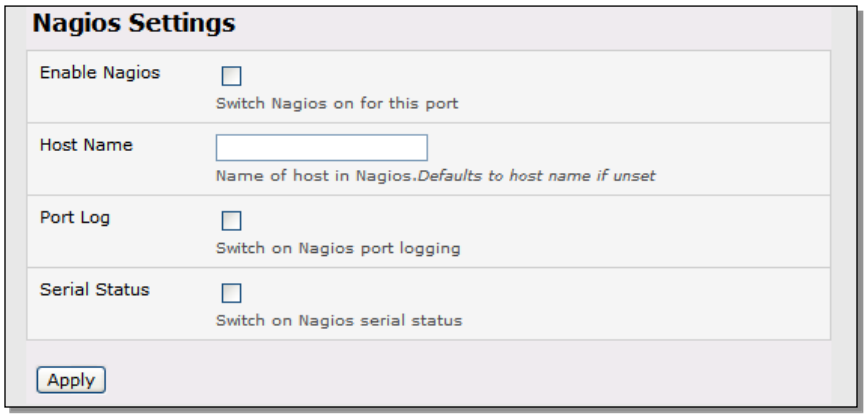


- Select **System: Nagios** and check **NSCA Enabled**
- Select the **Encryption** to be used from the drop down menu, then enter a **Secret** password and specify a check **Interval**
- Refer the sample Nagios configuration section below for some examples of configuring specific NSCA checks

10.2.4 Configure selected Serial Ports for Nagios monitoring

The individual Serial Ports connected to the *console server* to be monitored must be configured for Nagios checks. Refer *Chapter 4.4 – Network Host Configuration* for details on enabling Nagios monitoring for Hosts that are network connected to the *console server*. To enable Nagios to monitor on a device connected to the *console server* serial port:

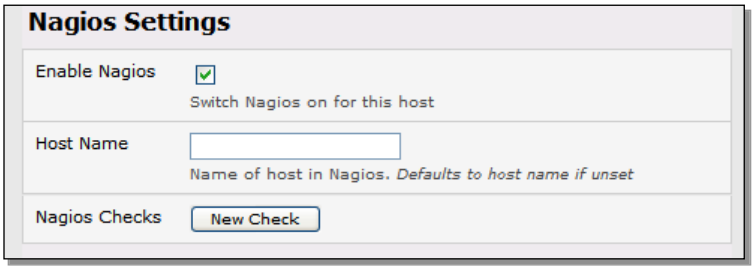
- Select **Serial&Network: Serial Port** and click **Edit** on the serial Port # to be monitored
- Select **Enable Nagios,** specify the name of the device on the upstream server and determine the check to be run on this port. **Serial Status** monitors the handshaking lines on the serial port and **Check Port** monitors the data logged for the serial port



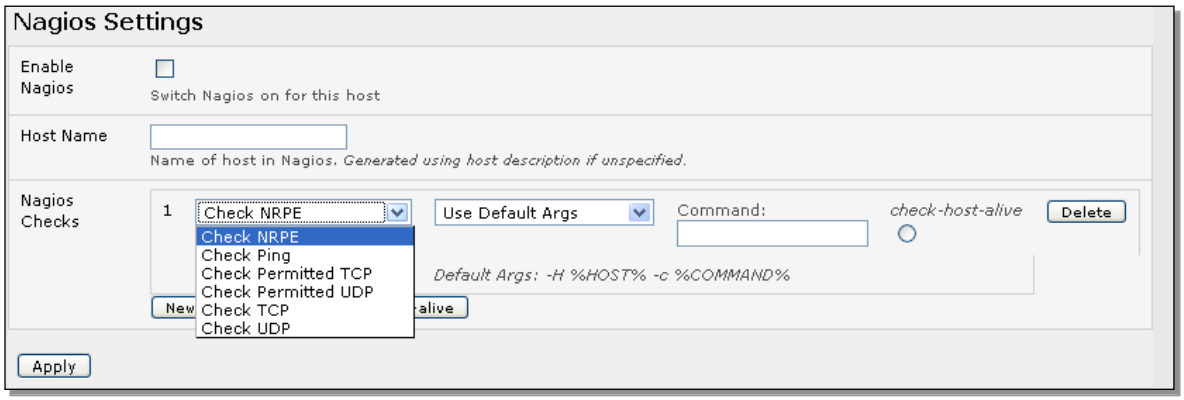
10.2.5 Configure selected Network Hosts for Nagios monitoring

The individual Network Hosts connected to the *console server* to be monitored must also be configured for Nagios checks:

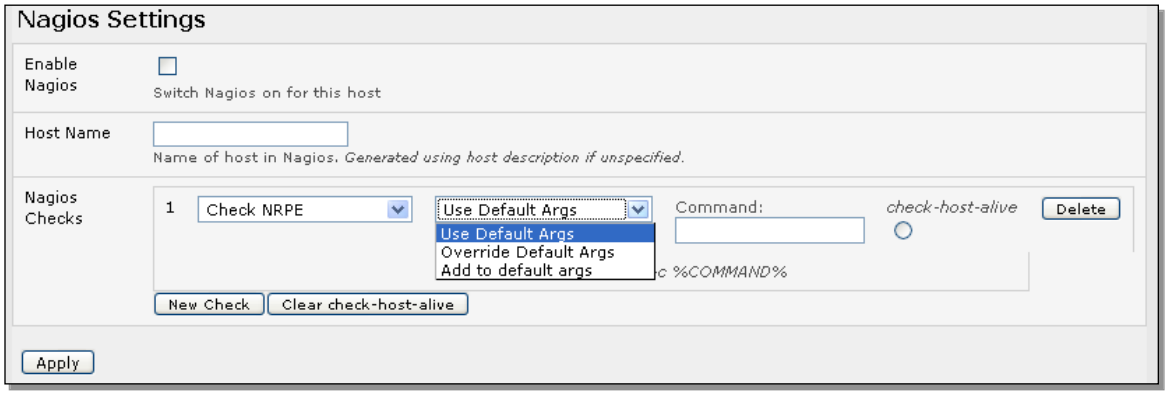
- Select **Serial&Network: Network Port** and click **Edit** on the Network Host to be monitored



- Select **Enable Nagios,** specify the name of the device as it will appear on the upstream Nagios server
- Click **New Check** to add a specific check which will be run on this host
- Select **Check Permitted TCP/UDP** to monitor a service that you have previously added as a **Permitted Service**
- Select **Check TCP/UDP** to specify a service port that you wish to monitor, but do not wish to allow external (SDT Connector) access to
- Select **Check TCP** to monitor



- The **Nagios Check** nominated as the ***check-host-alive*** check is the check used to determine whether the network host itself is up or down
- Typically this will be *Check Ping* – although in some cases the host will be configured not to respond to pings
- If no ***check-host-alive*** check is selected, the host will always be assumed to be up
- You may deselect ***check-host-alive*** by clicking **Clear *check-host-alive***
- If required, customize the selected **Nagios Checks** to use custom arguments
- Click **Apply**



10.2.6 Configure the upstream Nagios monitoring host

Refer to the Nagios documentation (http://www.nagios.org/docs/) for configuring the upstream server:

- The section entitled *Distributed Monitoring* steps through what you need to do to configure NSCA on the upstream server (under *Central Server Configuration*)
- *NRPE Documentation* has recently been added which steps through configuring NRPE on the upstream server http://nagios.sourceforge.net/docs/nrpe/NRPE.pdf

At this stage, Nagios at the upstream monitoring server has been configured, and individual serial port and network host connections on the *console server* configured for Nagios monitoring. If NSCA is enabled, each selected check will be executed once over the period of the check interval. If NRPE is enabled, then the upstream server will be able to request status updates under its own scheduling.

10.3 Advanced Distributed Monitoring Configuration

10.3.1 Sample Nagios configuration

An example configuration for Nagios is listed below. It shows how to set up a remote *Console server* to monitor a single host, with both network and serial connections. For each check it has two configurations, one each for NRPE and NSCA. In practice, these would be combined into a single check which used NSCA as a primary method, falling back to NRPE if a check was late – for details see the Nagios documentation (http://www.nagios.org/docs/) on *Service and Host Freshness Checks*

```
; Host definitions
;
; Opengear Console server
define host{
        use             generic-host
        host_name       opengear
        alias           Console server
        address         192.168.254.147
        }

; Managed Host
define host{
        use             generic-host
        host_name       server
        alias           server
        address         192.168.254.227
        }
```

```
; NRPE daemon on gateway
define command {
        command_name        check_nrpe_daemon
        command_line   $USER1$/check_nrpe -H 192.168.254.147 -p 5666
        }

define service {
        service_description     NRPE Daemon
        host_name               opengear
        use                     generic-service
        check_command                   check_nrpe_daemon
        }

; Serial Status
define command {
        command_name        check_serial_status
        command_line   $USER1$/check_nrpe -H 192.168.254.147 -p 5666 -c check_serial_$HOSTNAME$
        }

define service {
        service_description     Serial Status
        host_name               server
        use                     generic-service
        check_command                   check_serial_status
        }

define service {
        service_description     serial-signals-server
        host_name               server
        use                     generic-service
        check_command                   check_serial_status
        active_checks_enabled 0
        passive_checks_enabled          1
        }

define servicedependency{
        name                            opengear_nrpe_daemon_dep
        host_name                       opengear
        dependent_host_name             server
        dependent_service_description   Serial Status
        service_description             NRPE Daemon
        execution_failure_criteria      w,u,c
        }

; Port Log
define command{
        command_name    check_port_log
        command_line    $USER1$/check_nrpe -H 192.168.254.147 -p 5666 -c port_log_$HOSTNAME$
    }

define service {
        service_description     Port Log
        host_name               server
        use                     generic-service
        check_command                   check_port_log
        }
```

```
define service {
        service_description     port-log-server
        host_name               server
        use                     generic-service
        check_command                   check_port_log
        active_checks_enabled 0
        passive_checks_enabled          1
        }

define servicedependency{
        name                            opengear_nrpe_daemon_dep
        host_name                       opengear
        dependent_host_name             server
        dependent_service_description   Port Log
        service_description             NRPE Daemon
        execution_failure_criteria      w,u,c
        }

; Ping
define command{
        command_name    check_ping_via_opengear
        command_line    $USER1$/check_nrpe -H 192.168.254.147 -p 5666 -c host_ping_$HOSTNAME$
    }

define service {
        service_description     Host Ping
        host_name               server
        use                     generic-service
        check_command                   check_ping_via_opengear
        }

define service {
        service_description     host-ping-server
        host_name               server
        use                     generic-service
        check_command                   check_ping_via_opengear
        active_checks_enabled 0
        passive_checks_enabled          1
        }

define servicedependency{
        name                            opengear_nrpe_daemon_dep
        host_name                       opengear
        dependent_host_name             server
        dependent_service_description   Host Ping
        service_description             NRPE Daemon
        execution_failure_criteria      w,u,c
        }

; SSH Port
define command{
    command_name    check_conn_via_opengear
    command_line    $USER1$/check_nrpe -H 192.168.254.147 -p 5666 -c host_$HOSTNAME$_$ARG1$_$ARG2$
    }

define service {
        service_description     SSH Port
```

```
        host_name               server
        use                     generic-service
        check_command                   check_conn_via_opengear!tcp!22
        }

define service {
        service_description     host-port-tcp-22-server
                                ; host-port-<protocol>-<port>-<host>
        host_name               server
        use                     generic-service
        check_command                   check_conn_via_opengear!tcp!22
        active_checks_enabled   0
        passive_checks_enabled          1
        }

define servicedependency{
        name                            opengear_nrpe_daemon_dep
        host_name                       opengear
        dependent_host_name             server
        dependent_service_description   SSH Port
        service_description             NRPE Daemon
        execution_failure_criteria      w,u,c
        }
```

10.3.2 Basic Nagios plug-ins

Plug-ins are compiled executables or scripts that can be scheduled to be run on the *console server* to check the status of a connected host or service. This status is then communicated to the upstream Nagios server which uses the results to monitor the current status of the distributed network. Each *console server* is preconfigured with a selection of the checks that are part of the Nagios plug-ins package:

check_tcp and *check_udp* are used to check open ports on network hosts

check_ping is used to check network host availability

check_nrpe is used to execute arbitrary plug-ins in other devices

Each *console server* is preconfigured with two checks that are specific to Opengear:

check_serial_signals is used to monitor the handshaking lines on the serial ports

check_port_log is used to monitor the data logged for a serial port.

10.3.3 Additional plug-ins

Additional Nagios plug-ins (listed below) are available for all the CM7100, IM7200 and IM4200 products:

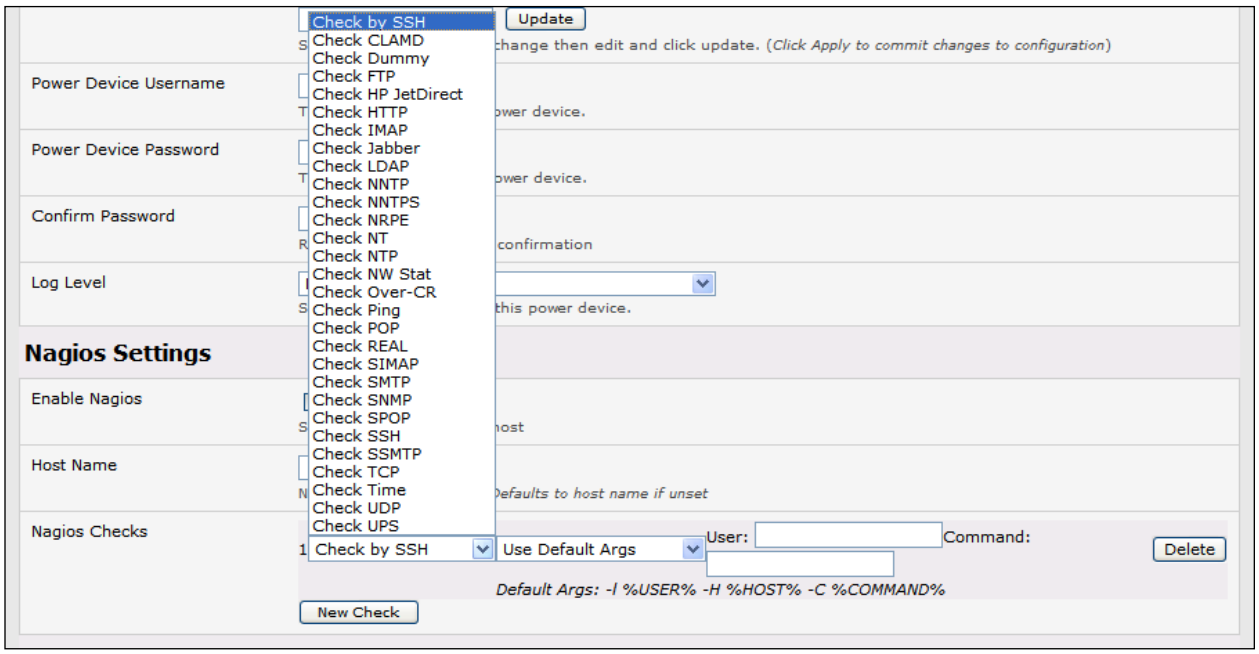
check_apt
check_by_ssh
check_clamd
check_dig
check_dns
check_dummy
check_fping
check_ftp
check_game
check_hpjd
check_http

check_imap
check_jabber
check_ldap
check_load
check_mrtg
check_mrtgtraf
check_nagios
check_nntp
check_nntps
check_nt
check_ntp
check_nwstat
check_overcr
check_ping
check_pop
check_procs
check_real
check_simap
check_smtp
check_snmp
check_spop
check_ssh
check_ssmtp
check_swap
check_tcp
check_time
check_udp
check_ups
check_users

These plug-ins from the Nagios plug-ins package can be downloaded from ftp.opengear.com.

There also are *bash* scripts which can be downloaded and run (primarily *check_log.sh).*

- To configure additional checks the downloaded plug-in program must be saved in the tftp *addins* directory on the USB flash and the downloaded text plug-in file saved in */etc/config*
- To enable these new additional checks you select **Serial&Network: Network Port**, then **Edit** the Network Host to be monitored, and select **New Checks**. The additional check option will have been included in the updated **Nagios Checks** list, and you can again customize the arguments



If you need other plug-ins to be loaded into the CM7100, IM7200 or IM4200 firmware:

- If the plug-in in a Perl script, it must be rewritten as the *console server* does not support Perl at this point. However, if you do require Perl support, please make a feature request to support@opengear.com
- Individual compiled programs may be generated using *gcc* for ARM. Again contact support@opengear.com for details

10.3.4 Number of supported devices

Ultimately the number of devices that can be supported by any particular *console server* is a function of the number of checks being made, and how often they are performed. Access method will also play a part. The table below shows the performance of three of the *console server* models (1/2 port, 8 port and 16/48 port) tabulating:

Time	No encryption	3DES	SSH tunnel
NSCA for single check	~ ½ second	~ ½ second	~ ½ second
NSCA for 100 sequential checks	100 seconds	100 seconds	100 seconds
NSCA for 10 sequential checks, batched upload	1 ½ seconds	2 seconds	1 second
NSCA for 100 sequential checks, batched upload	7 seconds	11 seconds	6 seconds

	No encryption	SSL	no encryption - tunneled over existing SSH session
NRPE time to service 1 check	1/10th second	1/3rd second	1/8th second
NRPE time to service 10 simultaneous checks	1 second	3 seconds	1 ¼ seconds
Maximum number of simultaneous checks before timeouts	30	20 (1,2 and 8) or 25 (16 and 48 port)	25 (1,2 and 8 port), 35 (16 and 48 port)

The results were from running tests 5 times in succession with no timeouts on any runs. However there are a number of ways to increase the number of checks you can do:

Usually when using NRPE checks, an individual request will need to set up and tear down an SSL connection. This overhead can be avoided by setting up an SSH session to the *console server* and tunneling the NRPE port. This allows the NRPE daemon to be run securely without SSL encryption, as SSH will take care of the security.

When the *console server* submits NSCA results it staggers them over a certain time period (e.g. 20 checks over 10 minutes will result in two check results every minute). Staggering the results like this means that in the event of a power failure or other incident that causes multiple problems, the individual freshness checks will be staggered too.

NSCA checks are also batched. So in the previous example the two checks per minute will be sent through in a single transaction.

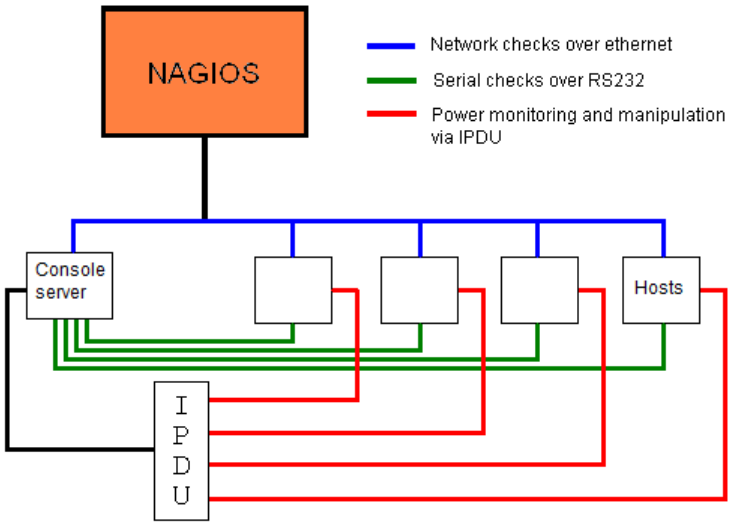
10.3.5 Distributed Monitoring Usage Scenarios

Below are a number of distributed monitoring Nagios scenarios:

I. Local office

In this scenario, the *console server* is set up to monitor the console of each managed device. It can be configured to make a number of checks, either actively at the Nagios server's request, or passively at preset intervals, and submit the results to the Nagios server in a batch.

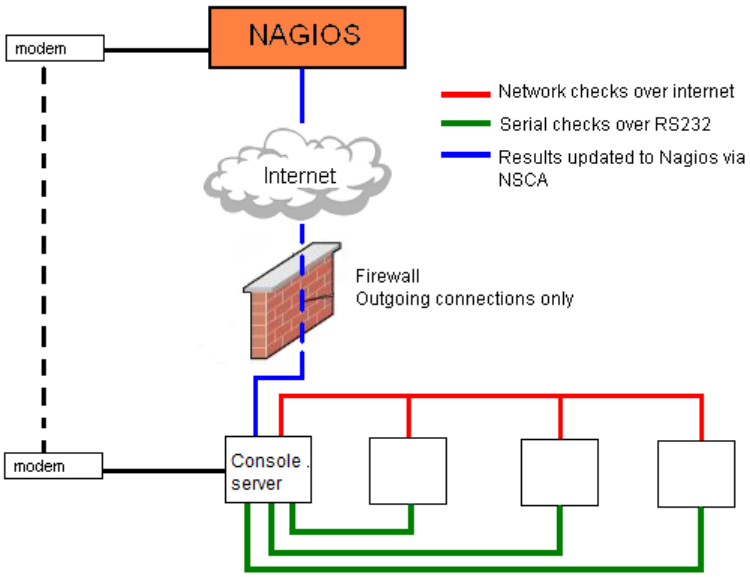
The *console server* may be augmented at the local office site by one or more Intelligent Power Distribution Units (IPDUs) to remotely control the power supply to the managed devices.



II. Remote site

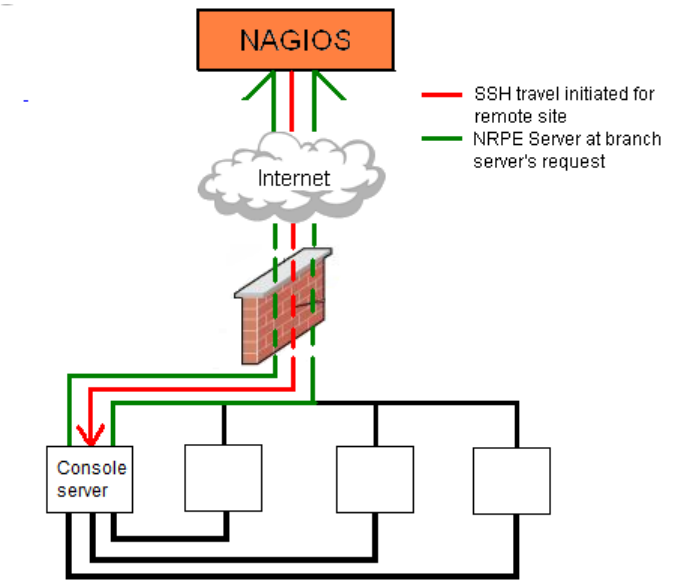
In this scenario the *console server* NRPE server or NSCA client can be configured to make active checks of configured services and upload to the Nagios server waiting passively. It can also be configured to service NRPE commands to perform checks on demand.

In this situation, the *console server* will perform checks based on both serial and network access.



Remote site with restrictive firewall

In this scenario the role of the *console server* will vary. One aspect may be to upload check results through NSCA. Another may be to provide an SSH tunnel to allow the Nagios server to run NRPE commands.



Remote site with no network access

In this scenario the *console server* allows dial-in access for the Nagios server. Periodically, the Nagios server will establish a connection to the *console server* and execute any NRPE commands, before dropping the connection

SYSTEM MANAGEMENT

This chapter describes how the *Administrator* can perform a range of general *console server* system administration and configuration tasks such as:

- Applying *Soft* and *Hard* Resets to the gateway
- Re-flashing the Firmware
- Configuring the Date, Time and NTP
- Setting up Backup of the configuration files
- Delayed configuration commits
- Configuring the console server in FIPS mode

System administration and configuration tasks that are covered elsewhere include:

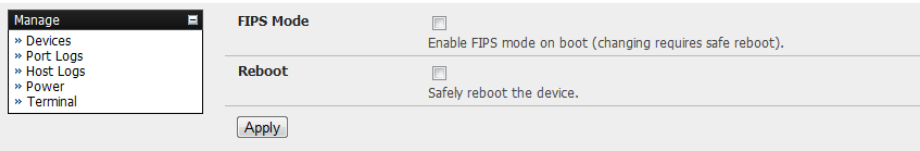
- Resetting System Password and entering new System Name / Description for the *console server* (*Chapter 3.2*)
- Setting the *console server's* System IP Address (*Chapter 3. 3)*
- Setting the permitted Services by which to access the *console server* (*Chapter 3.4*)
- Setting up OOB Dial-in (*Chapter 5*)
- Configuring the Dashboard (*Chapter 12*)

11.1 System Administration and Reset

The *Administrator* can reboot or reset the gateway to default settings.

A *soft* reset is affected by:

- Selecting **Reboot** in the **System: Administration** menu and clicking **Apply**



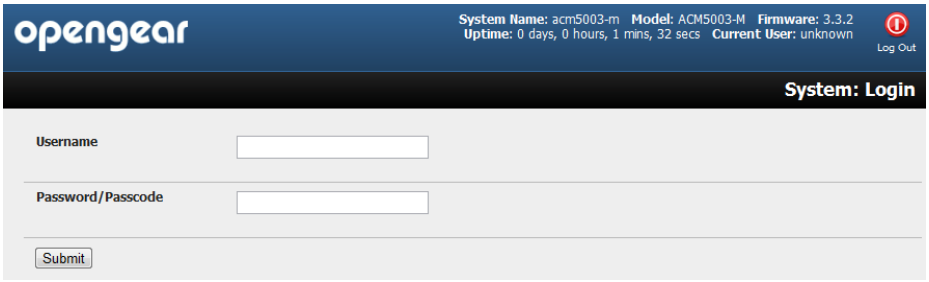
The *console server* reboots with all settings (*e.g.* the assigned network IP address) preserved. However this *soft* reset does disconnect all users and ends any SSH sessions that had been established.

A *soft* reset will also be affected when you switch OFF power from the *console server*, and then switch the power back ON. However if you cycle the power and the unit is writing to flash you could corrupt or lose data, so the software reboot is the safer option.

A *hard* erase (*hard reset*) is effected by:

- Pushing the *Erase* button on the rear panel **twice**. A ball point pen or bent paper clip is a suitable tool for performing this procedure. Do not use a graphite pencil. Depress the button gently **twice** (within a couple of second period) while the unit is powered ON.

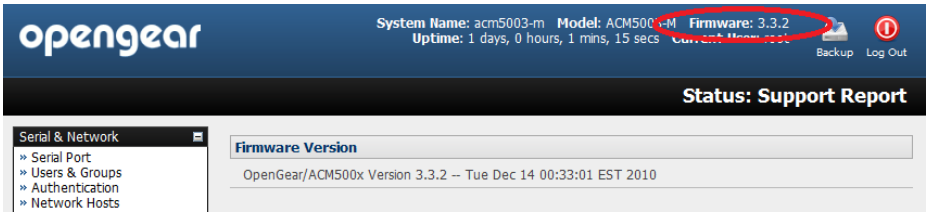
This will reset the *console server* back to its factory default settings and clear the *console server*'s stored configuration information (*i.e.* the IP address will be reset to 192.168.0.1). You will be prompted to log in and must enter the default administration username and administration password (Username: **root** Password: **default**).



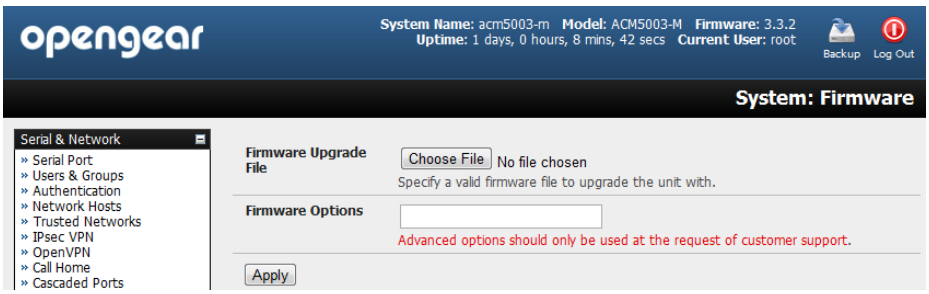
11.2 Upgrade Firmware

Before upgrading you should ascertain if you are already running the most current firmware in your Opengear device. Your Opengear device will not allow you to upgrade to the same or an earlier version.

- The **Firmware** version is displayed in the header of each page
- Alternately selecting **Status: Support Report** reports the **Firmware Version**



- To upgrade, you first must download the latest firmware image from *ftp://ftp.opengear.com* or *http://opengear.com/firmware/*:
  - For ACM5000 family download **acm500x.flash**
  - For ACM5500 family download **acm500x.flash**
  - For ACM7000 family download **acm700x.flash**
  - For CM7116/7132/7148-2 download **cm71xx.flash**
  - For IM4216-34 and IM4208/16/32/48-2 download **im42xx.flash**
  - For IM7208/16/32/48-2 download **im72xx.flash**
  - For SD4001/4002 download **sd4002.flash**
- Save this downloaded firmware image file on to a system on the same subnet as the Opengear device
- Also download and read the *Release Notes* file for the latest information
- To up-load the firmware image file, select **System: Firmware**



- Specify the address and name of the downloaded Firmware Upgrade File, or **Browse** the local subnet and locate the downloaded file

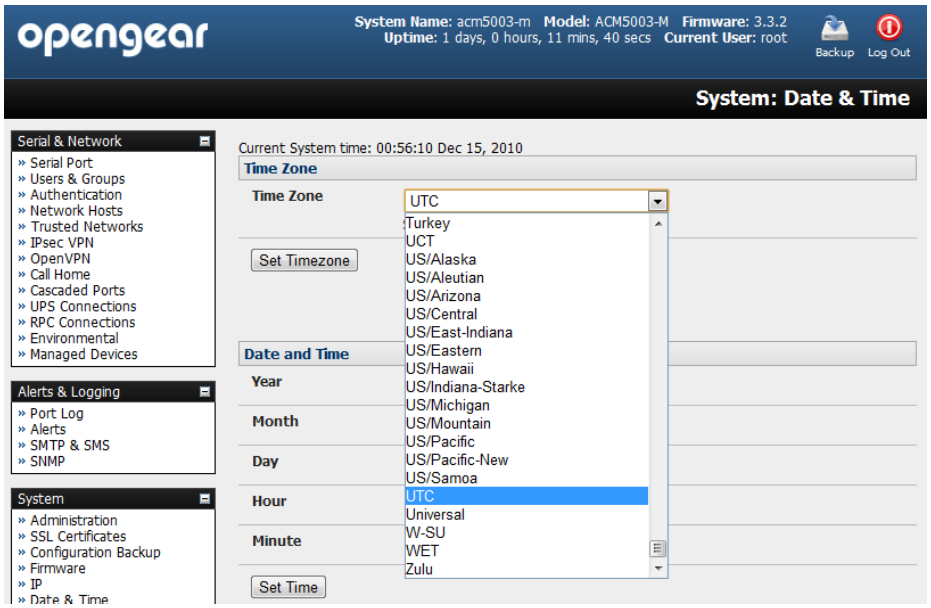
- Click **Apply** and the Opengear device will undertake a soft reboot and commence upgrading the firmware. This process will take several minutes
- After the firmware upgrade has completed, click **here** to return to the Management Console. Your Opengear device will have retained all its pre-upgrade configuration information

11.3 Configure Date and Time

It is important to set the local Date and Time in your Opengear appliance as soon as it is configured. Features such as Syslog and NFS logging use the system time for time-stamping log entries, while certificate generation depends on a correct Timestamp to check the validity period of the certificate.

Your Opengear appliance can synchronize its system time with a remote Network Time Protocol (NTP) server. NTP uses Coordinated Universal Time (UCT) for all time synchronizations so it is not affected by different time zones. However you do need to specify your local time zone so the system clock shows correct local time:

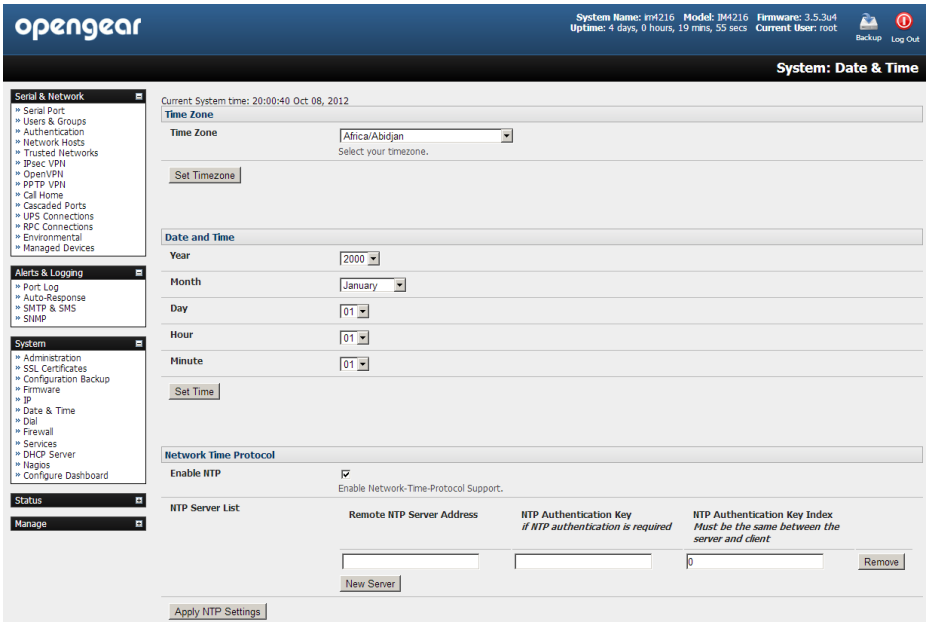
- Set your appropriate region/locality in the **Time Zone** selection box and click **Set Timezone**



Note: With Version 3.2.0 firmware the Time Zone can also be set to UCT which replaced Greenwich Mean Time as the World standard for time in 1986.

Configuring NTP ensures the Opengear appliance clock is kept extremely accurate (once Internet connection has been established).

- Select the **Enable NTP** checkbox in the **Network Time Protocol** section of the **System: Date & Time** page
- Enter the IP address of the remote **NTP Server**
- If your external NTP server requires authentication, you need to specify the **NTP Authentication Key** and the **Key Index** to use when authenticating with the NTP server
- Click **Apply NTP Settings**

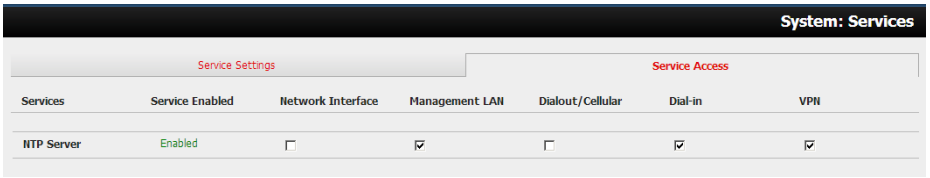


If remote NTP is not used, the time can be set manually:

- Enter the **Year**, **Month**, **Day**, **Hour** and **Minute** using the **Date** and **Time** selection boxes
- Check **Set Time**

Note: All Opengear appliances, except the SD4001/2 models, have an internal battery-backed hardware clock. When the time and date is set manually through the management console, or retrieved from an NTP server, the hardware clock of the Opengear appliance is automatically updated. The hardware clock uses a battery to allow the current time and date to be maintained across reboots, and after the appliance has been powered down for longer periods of time.

Note: With the NTP peering model, the Opengear appliance can share its time information with other devices connected to it, so all devices can be time synchronized. To do this, tick Enable NTP on the Time and Date page, and ensure that the appropriate networks are selected on the Service Access page.



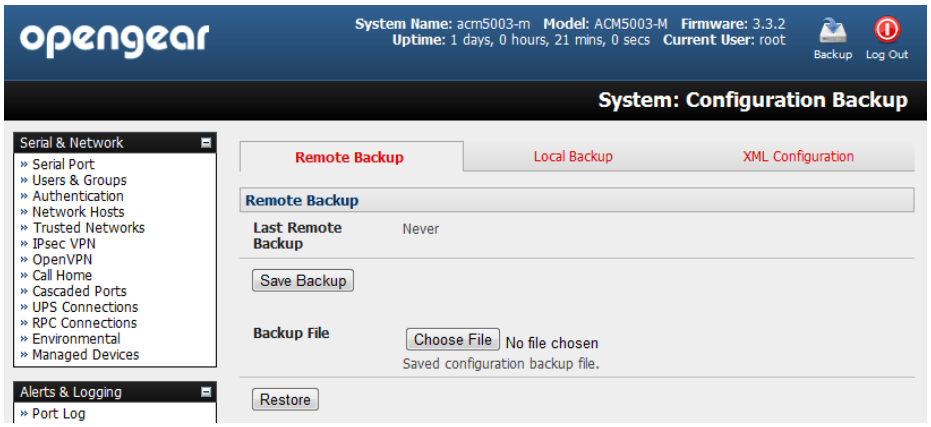
11.4 Configuration Backup

It is recommended that you back up the *console server* configuration whenever you make significant changes (such as adding new Users or Managed Devices) or before performing a firmware upgrade.

- Select the **System: Configuration Backup** menu option or click the

icon

Note The configuration files can also be backed up from the command line (refer *Chapter 14*)



With all *console servers* you can save the backup file remotely on your PC and you can restore configurations from remote locations:

- Click **Save Backup** in the Remote Configuration Backup menu
- The config backup file (*System Name_date_config.opg*) will be downloaded to your PC and saved in the location you nominate

To restore a remote backup:

- Click **Browse** in the Remote Configuration Backup menu and select the **Backup File** you wish to restore
- Click **Restore** and click **OK.** This will overwrite all the current configuration settings in your *console server*

Alternately with some *console servers* you can save the backup file locally onto the USB storage. To do this your *console server* must support USB and you must have an internal or external USB flash drive installed.

To backup and restore using USB:

- Ensure the USB flash is the only USB device attached to the *console server*
- Select the **Local Backup** tab and **click here to proceed**. This will set a Volume Label on the USB storage device. This preparation step is only necessary the first time, and will not affect any other information you have saved onto the USB storage device. However it is recommended that you back up any critical data from the USB storage device before using it with your *console server*. If there are multiple USB devices installed you will be warned to remove them

- To back up to the USB enter a brief **Description** of the backup in the Local Backup menu and select **Save Backup**
- The Local Backup menu will display all the configuration backup files you have stored onto the USB flash
- To restore a backup from the USB simply select **Restore** on the particular backup you wish to restore and click **Apply**

After saving a local configuration backup, you may choose to use it as the alternate default configuration. When the *console server* is reset to factory defaults, it will then load your alternate default configuration instead of its factory settings:

- To set an alternate default configuration, check **Load On Erase** and click **Apply**

Note: Before selecting *Load On Erase* please ensure you have tested your alternate default configuration by clicking Restore

If for some reason your alternate default configuration causes the *console server* to become unbootable recover your unit to factory settings using the following steps:

- If the configuration is stored on an external USB storage device, unplug the storage device and reset to factory defaults as per section 11.1 of the user manual
- If the configuration is stored on an internal USB storage device reset to factory defaults using a specially prepared USB storage device:
  - The USB storage device must be formatted with a Windows FAT32/VFAT file system on the first partition or the entire disk, most USB thumb drives are already formatted this way
  - The file system must have the volume label: OPG_DEFAULT
  - Insert this USB storage device into an external USB port on the *console server* and reset to factory defaults as per section 11.1

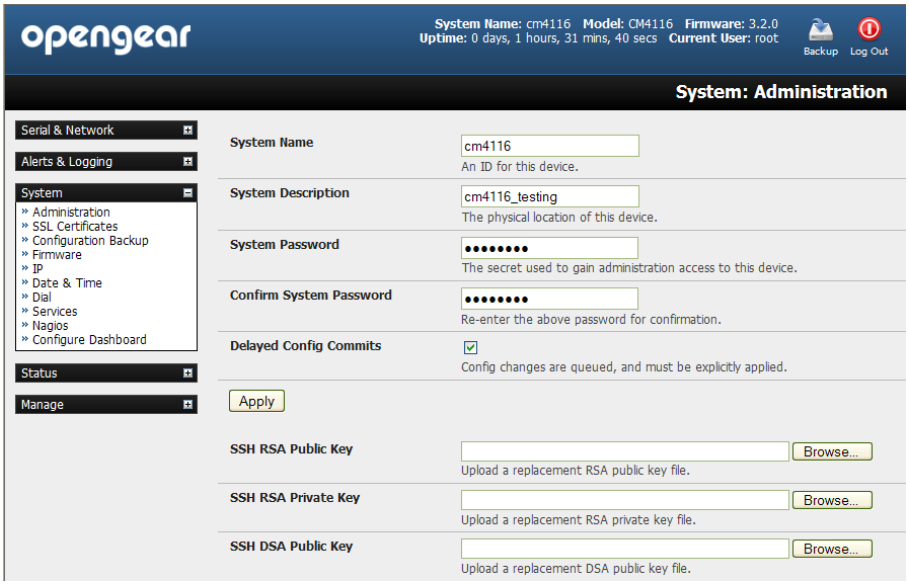
After recovering your *console server*, ensure the problematic configuration is no longer selected for Load On Erase

---

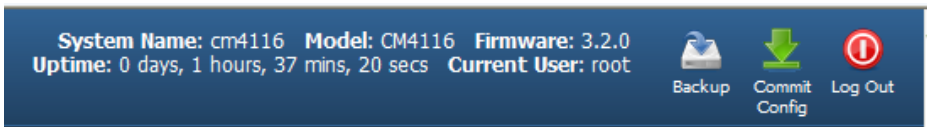
11.5 Delayed Configuration Commit

This mode allows the grouping or queuing of configuration changes and the simultaneous application of these changes to a specific device. For example, changes to authentication methods or user accounts may be grouped and run once to minimize system downtime. To enable:

- Check the **Delayed Config Commits** button under **System: Administration**
- Click **Apply**



- The Commit Config icon will be displayed in top right-hand corner of the screen between the Backup and Log Out icons



To queue then run configuration changes:

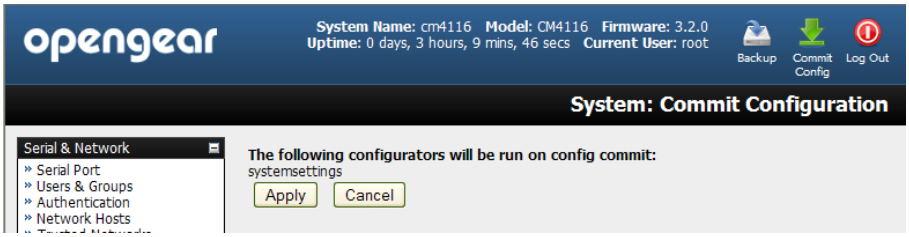
- Firstly apply all the required changes to the configuration e.g. modify user accounts, amend authentication method, enable OpenVPN tunnel or modify system time
- Click the **Commit Config** button. This will generate the **System: Commit Configuration** screen displaying all the configurators to be run

- Click **Apply** to run all the configurators in the queue
- Alternately click **Cancel** and this will discard all the delayed configuration changes

Note All the queued configuration changes will be lost if Cancel is selected

To disable the Delayed Configuration Commits mode:

- Uncheck the **Delayed Config Commits** button under **System: Administration** and click **Apply**
- Click the **Commit Config** button in top right-hand corner of the screen to display the **System: Commit Configuration** screen
- Click **Apply** to run the *systemsettings* configurator



The **Commit Config** button will no longer be displayed in the top right-hand corner of the screen and configurations will no longer be queued.

11.6 FIPS Mode

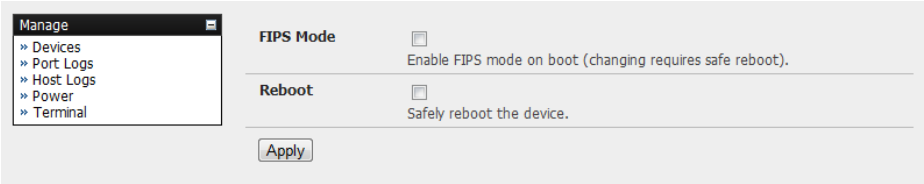
The ACM7000, ACM5500, ACM5000, CM7100, IM7200 and IM4200 family of advanced *console server* families all use an embedded cryptographic module that has been validated to meet the FIPS 140-2 standards.

Note The US National Institute of Standards and Technology (NIST) publishes the FIPS (Federal Information Processing Standard) series of standards. FIPS 140-1 and FIPS 140-2 are both technical standards and worldwide de-facto standards for the implementation of cryptographic modules. These standards and guidelines are issued by NIST for use government-wide. NIST develops FIPS when there are compelling Federal government requirements such as for security and interoperability and there are no acceptable industry standards or solutions.

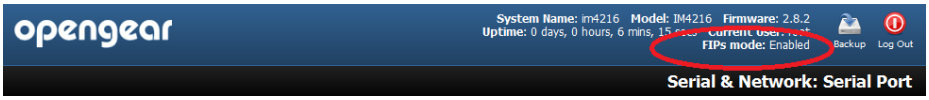
Opengear advance console servers use an embedded OpenSSL cryptographic module that has been validated to meet the FIPS 140-2 standards and has received Certificate #1051

When configured in FIPs mode all SSH, HTTPS and SDT Connector access to all services on the advanced console *servers* will use the embedded FIPS compliant cryptographic module. To connect you must also be using cryptographic algorithms that are FIPs approved in your browser or client or the connection will fail.

- Select the **System: Administration** menu option
- Check **FIPS Mode** to enable FIPS mode on boot, and check **Reboot** to safely reboot the console server



- Click **Apply** and the console server will now reboot. It will take several minutes to reconnect as secure communications with your browser are validated, and when reconnected it will display "*FIPs mode: Enabled*" in the banner



Note: To enable FIPS mode from the command line, login and run these commands:

config -s config.system.fips=on
touch /etc/config/FIPS
chmod 444 /etc/config/FIPS
flatfsd -b

The final command saves to flash and reboots the unit. The unit will take a few minutes to boot into FIPS mode. To disable FIPS mode:

config -d config.system.fips
rm /etc/config/FIPS
flatfsd –b

STATUS REPORTS

This chapter describes the dashboard feature and the status reports that are available:

- Port Access and Active Users
- Statistics
- Support Reports
- Syslog
- Dashboard

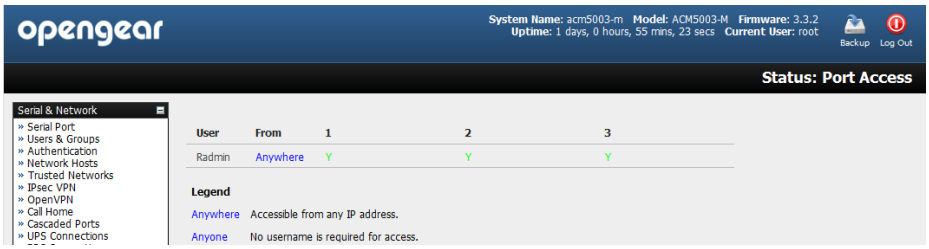
Other status reports that are covered elsewhere include:

- UPS Status (*Chapter 8.2*)
- RPC Status (*Chapter 8.1*)
- Environmental Status (*Chapter 8.3*)

12.1 Port Access and Active Users

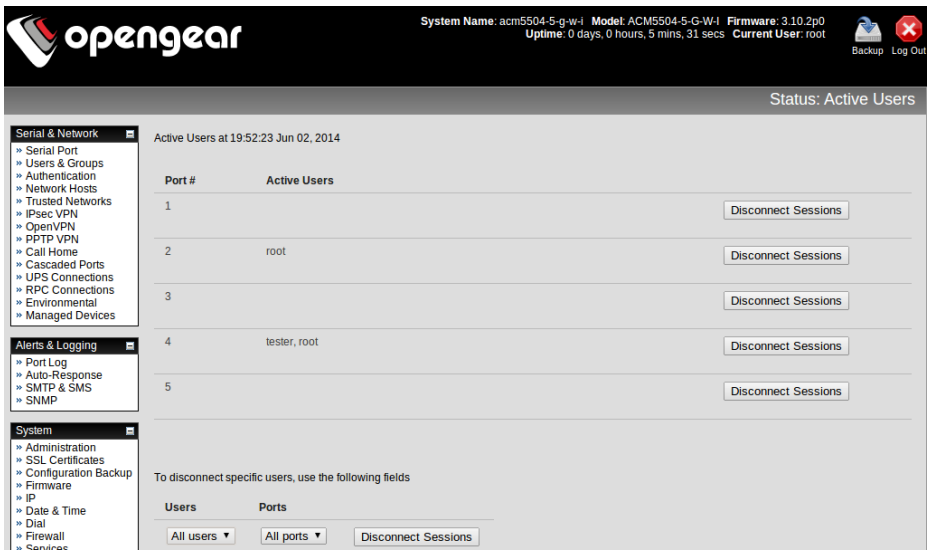
The *Administrator* can see which *Users* have access privileges with which serial ports:

➢ Select the **Status: Port Access**



The *Administrator* can also see the current status as to *Users* who have active sessions on those ports:

➢ Select the **Status: Active Users**



With Firmware V3.11 and later the **Status: Active Users** menu has been extended to enable *Administrators* to selectively terminate serial sessions. Connection types *telnet, SSH, raw TCP* and *unauthenticated telnet* can be disconnected. However you cannot disconnect an RFC2217 session

The *root* user, or any user in the *admin* group, can access the **Active Users** page, which shows a snapshot of the connected sessions, at the time indicated by the timestamp displayed at the top of the page. Note that this page only shows the local console ports, and does not include any cascaded ports.

There are "*Disconnect Sessions*" buttons along the right hand side of the table listing active users. These buttons disconnect all sessions from the *Port* they correspond to. If the port is not set up in *Console Server* mode, the user will see a pop up error informing them that they need to configure the port as *Console Server* mode before they can connect and disconnect.

After the buttons have been pressed, the selected sessions will be disconnected, and the number of disconnect sessions will be displayed to the user.

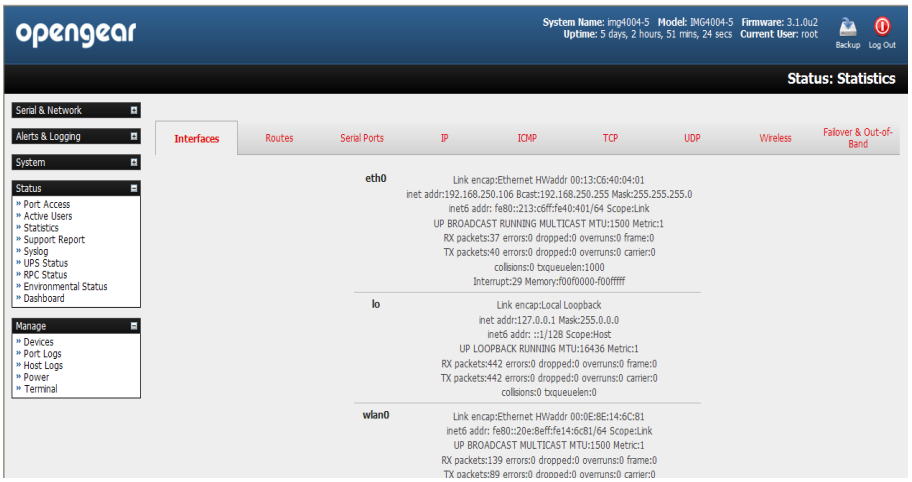
To allow more detailed control of who to disconnect, there is a table at the bottom of the page with drop down lists for all connected users and all connected ports that allow the user to choose who do disconnect.So if you wish to disconnect the user '*tester*' from all ports, choose '*teste*r' in the *Users* box, and *All ports* in the *Ports* box then hit the *Disconnect Sessions* button.

Note: You can also disconnect serial sessions from the command line using the *--disconnect* option with the *pmusers* command.

12.2 Statistics

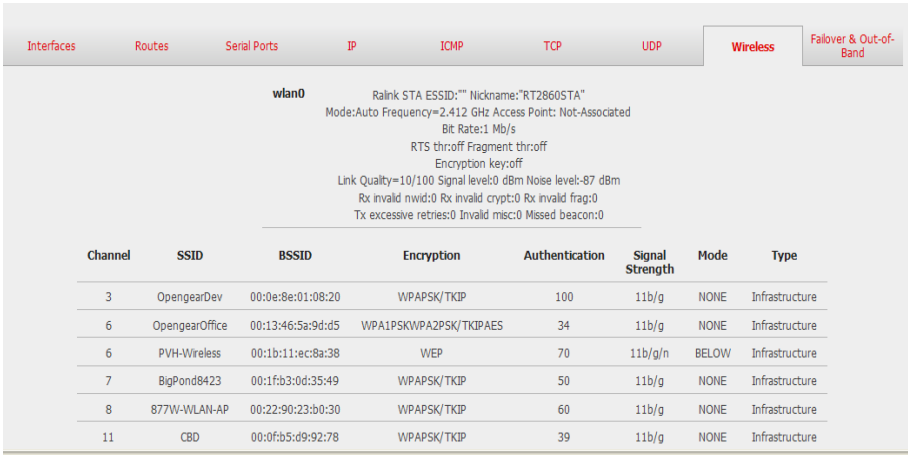
The Statistics report provides a snapshot of the status, current traffic and other activities and operations of your *console server*:

- Select the **Status: Statistics**



- Detailed statistics reports can be found by selecting the various submenus.

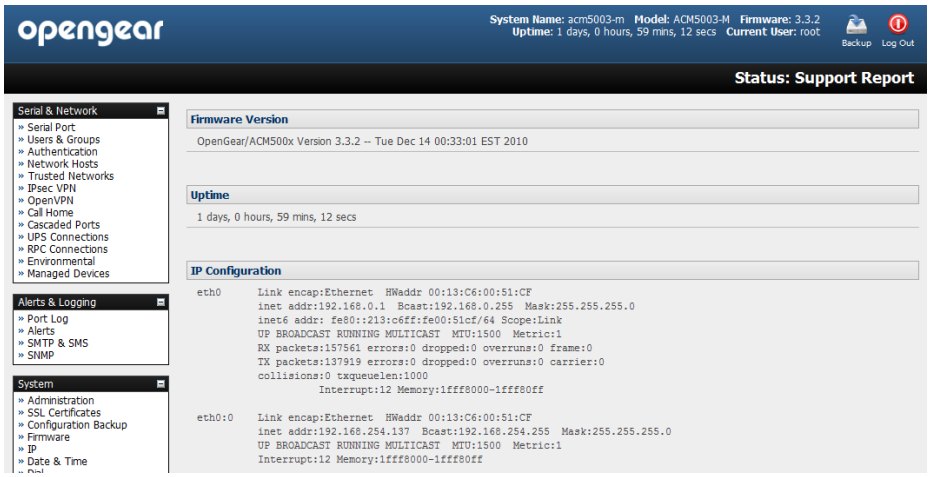
  For example if you have an ACM5504-5-G-W-I configured with a wireless LAN connection the **Wireless** screen will display all the locally accessible wireless LANs. So you can see the SSID and the Encryption / Authentication settings to use for the particular access point you wish to connect to. Also when you have successfully connected the SSID of this access point will then be shown in the Wireless ESSID filed of **ra0** (shows below as “” which is not connected)



12.3 Support Reports

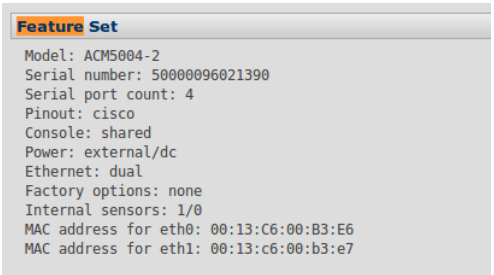
The Support Report provides useful status information that will assist the Opengear technical support team to solve any problems you may experience with your *console server*.

If you do experience a problem and have to contact support, ensure you include the Support Report with your email support request. The Support Report should be generated when the issue is occurring, and attached in plain text format.



- Select **Status: Support Report** and you will be presented with a status snapshot
- Save the file as a text file and attach it to your support email

Note: For console servers running firmware v3.11 and above, for devices where the serial number can be retrieved, there is now a Feature Set section in the support report where the serial number is amongst the information displayed. Devices for which the serial number can be displayed are ACM5000, ACM5500, ACM7000, CM7100 and IM7200. For devices not supporting this feature there is no change to the support report.



There is also a new cli command on all devices called *showserial* which does nothing more than return the serial number if it is available or *'No serial number information available*' for devices that can't retrieve their serial numbers. The reason the command exists on all devices is so that Lighthouse bulk commands can be run on many console servers to run an audit serial numbers and obtain as many serial numbers as possible in one operation.

12.4 Syslog

The Linux System Logger in the *console server* maintains a record of all system messages and errors:

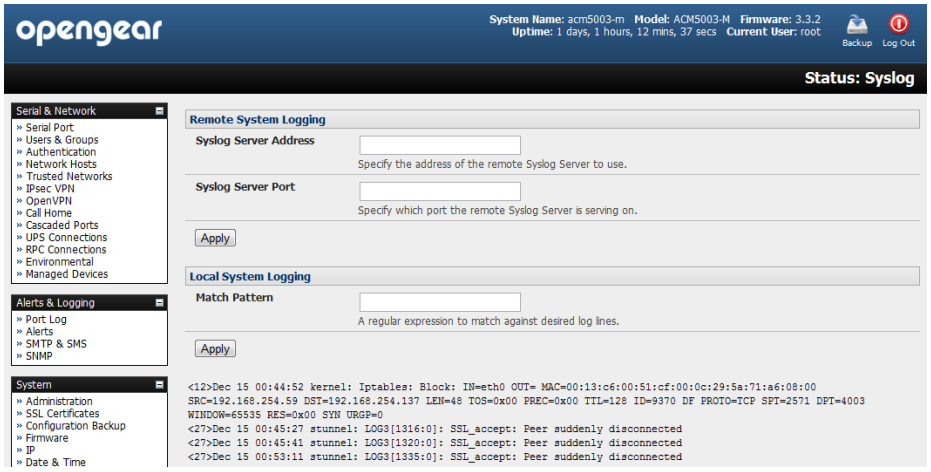
- Select **Status: Syslog**

The syslog record can be redirected to a remote Syslog Server:

- Enter the remote **Syslog Server Address** and **Syslog Server Port** details and click **Apply**

The console maintains a local Syslog. To view the local Syslog file:

- Select **Status: Syslog**

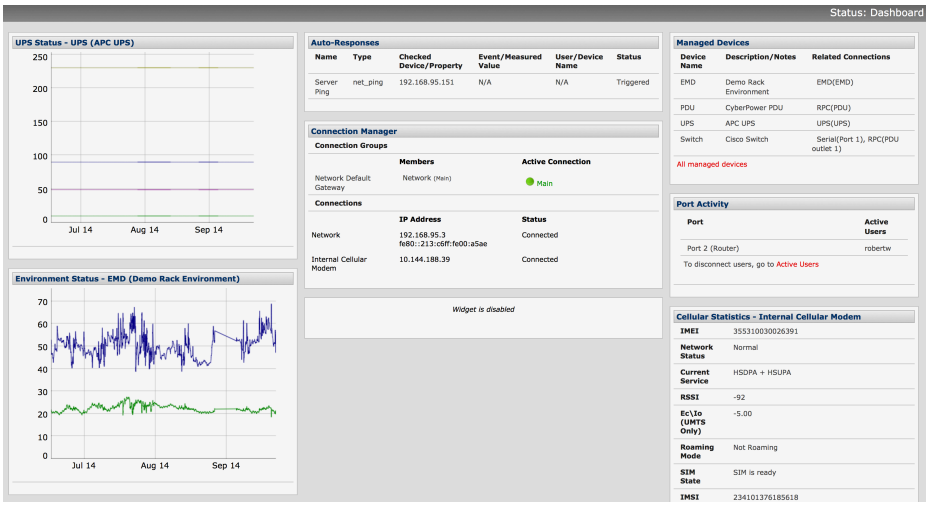


To make it easier to find information in the local Syslog file, a pattern matching filter tool is provided.

- Specify the **Match Pattern** that is to be searched for (*e.g.* the search for *mount* is shown below) and click **Apply.** The Syslog will then be represented with only those entries that actually include the specified pattern

12.5 Dashboard

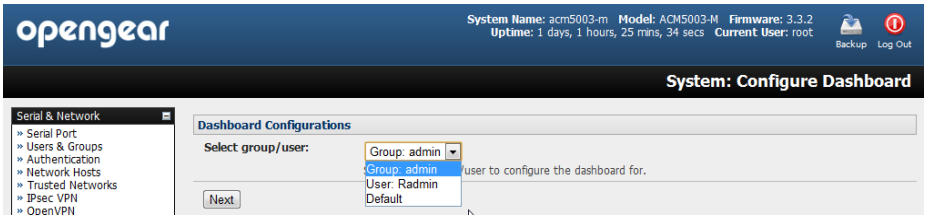
The Dashboard provides the administrator with a summary of the status of the *console server* and its Managed Devices. Custom dashboards can be configured for each user groups.



12.5.1 Configuring the Dashboard

Only users who are members of the *admin* group (and the *root* user) can configure and access the dashboard. To configure a custom dashboard:

- Select **System: Configure Dashboard** and select the user (or group) you are configuring this custom dashboard layout for



Note: You can configure a custom dashboard for any *admin* user or for the *admin* group or you can reconfigure the default dashboard

The *Status:Dashboard* screen is the first screen displayed when *admin* users (other than *root*) log into the console manager. If you log in as "*John*", and John is member of the *admin* group and there is a dashboard layout configured for John, then you will see the dashboard for John (on log-in and each time you click on the *Status:Dashboard* menu item.

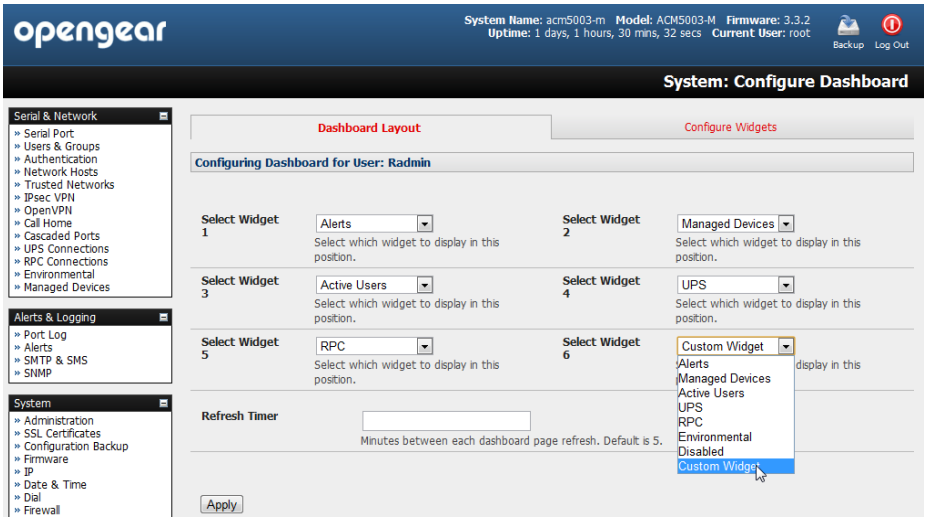
If there is no dashboard layout configured for John but there is an *admin* group dashboard configured then you will see the admin group dashboard instead. If there is no user dashboard or admin group dashboard configured, then you will see the default dashboard.

The *root* user does not have its own dashboard.

The above configuration options are intended to enable admin users to setup their own custom dashboards

The Dashboard displays a configurable number of *widgets*. These widgets include status for major subsystems such as *conma*, *Auto-Response*, *Managed Devices* and cellular. The *admin* user can configure which of these widgets is to be displayed where:

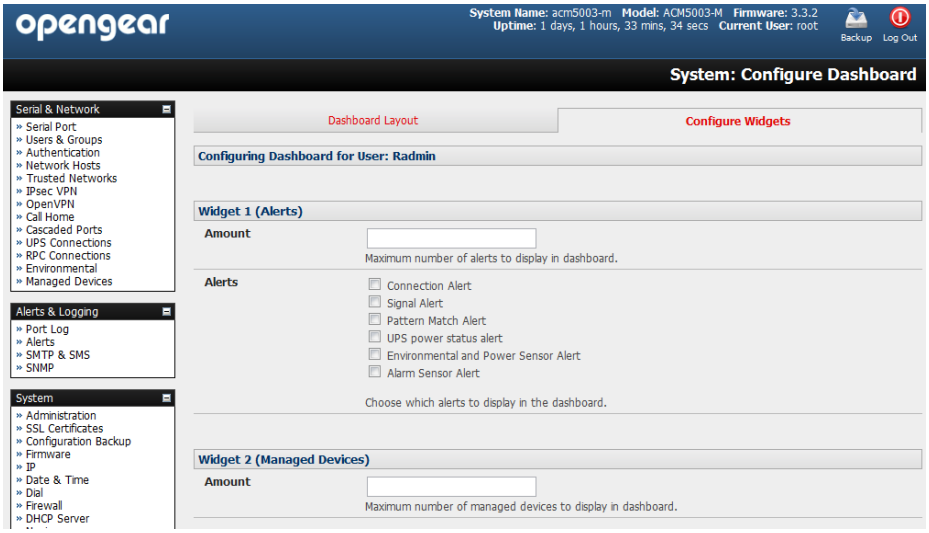
- Go to the **Dashboard layout** panel and select which widget is to be displayed in each of the **Widget Slots**
- Click Apply



Note: The Alerts widget is a new screen that shows the current alerts status. When an alert gets triggered, a corresponding .XML file is created in */var/run/alerts/.* The dashboard scans all these files and displays a summary status in the alerts widget. When an alert is deleted the corresponding .XML files that belong to that alert are also deleted.

To configure what is to be displayed by each widget:

- Go to the **Configure widgets** panel and configure each selected widget (e.g. specify which UPS status is to be displayed on the *ups widget* or the maximum number of Managed Devices to be displayed in the *devices widget*
- Click Apply



Note: Dashboard configuration is stored in the /etc/config/config.xml file. Each configured dashboard will increase the config file. If this file gets too big, you can run out of memory space on the *console server*.

12.5.2 Creating custom widgets for the Dashboard

To run a custom script inside a dashboard widget:

Create a file called "*widget-<name>.sh*" in the folder */etc/config/scripts /* where <*name*> can be anything. You can have as many custom dashboard files as you want.

Inside this file you can put any code you wish. When configuring the dashboard, choose "*widget-<name>.sh*" in the dropdown list. The dashboard will run the script and display the output of the script commands directly on the screen, inside the specific widget.

The best way to format the output would be to send HTML commands back to the browser by adding echo commands in the script:

echo '<table>'

You can of course run any command and its output will be displayed in the widget window directly.
Below is an example script which writes the current date to a file, and then echo's HTML code back to the browser. The HTML code gets an image from a specific URL and displays it in the widget.

```
#!/bin/sh

date >> /tmp/test
echo '<table>'
echo '<tr><td> This is my custom script running </td></tr>'
echo '<tr><td>'
echo '<img src="http://www.vinras.com/images/linux-online-inc.jpg">'
echo '</td></tr>'
echo '</table>'

exit 0
```

MANAGEMENT

The *console server* has a small number of **Manage** reports and tools that are available to both *Administrator*s and *Users*:

- Access and control authorized devices
- View serial port logs and host logs for those devices
- Use SSH or the Web Terminal to access serially attached consoles
- Control of power devices (where authorized)

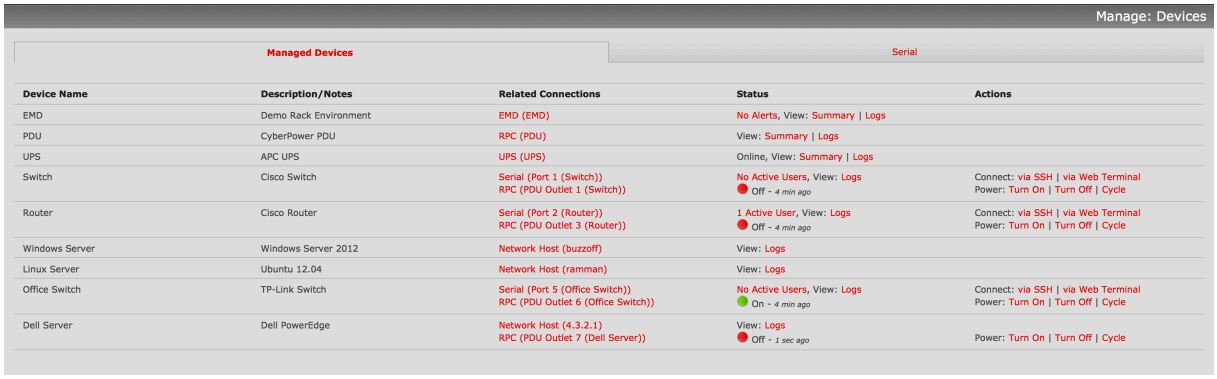
All other Management Console menu items are available to *Administrators* only.

13.1 Device Management

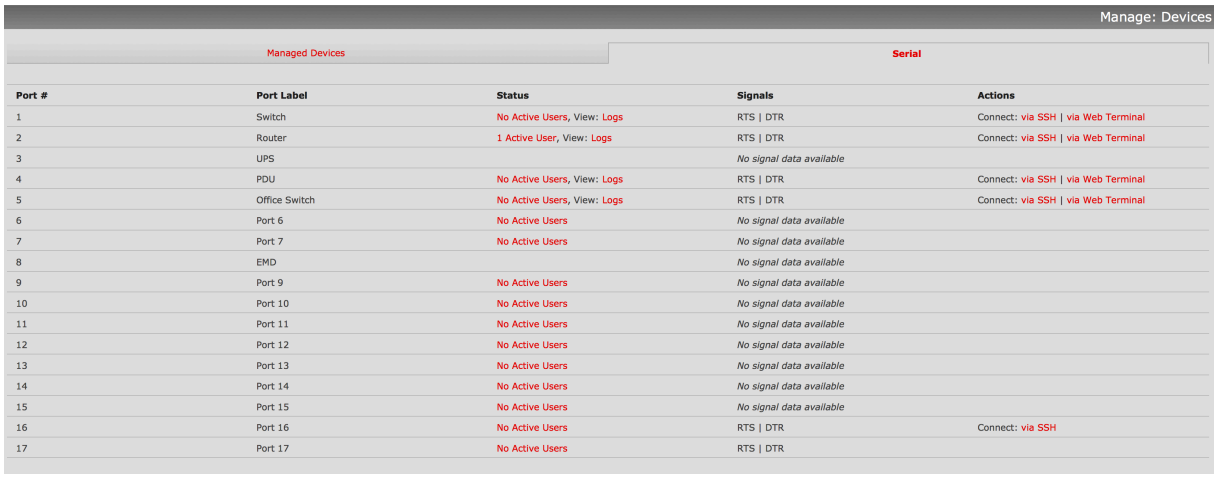
Note: The Manage Devices UI has been significantly updated in firmware version 3.12.

To display Managed Devices and their grouped serial, network and power connections:

- Select **Manage: Devices** or click the **Manage Devices** icon in the top right of the UI
- *admin*-group users are presented with a list of all configured Managed Devices and their constituent connections, *user*-group users only see the Managed Devices where for each Related Connection, they have been explicitly permitted access



- The **Status** column displays the current most salient status for each Related Connection (e.g. Active Users for serial connections, and power status for RPC outlet connections) with links to detailed status
- The links in the **Actions** column are used to control the Managed Device (e.g. connect to a console session or power cycle – power actions are not performed until the action has been confirmed via pop-up message)
- The *Administrator* will be presented with a list of all configured Managed Devices whereas the *User* will only see the Managed Devices they (or their Group) has been given access privileges for
- Alternatively, select the **Serial** tab for an ungrouped view of permitted serial port connections for the current user



- An additional **Signals** column displays the current state of the serial pins

Note: To use the Connect: via SSH links, your computer's operating system must recognize the *ssh://* URI scheme and have a protocol handler configured (e.g. an SSH client like SecureCRT).

13.2 Port and Host Logs

Administrators and *Users* can view and download logs of data transferred to and from connected devices.

- Select **Manage: Port Logs** and the serial Port # to be displayed
- To display Host logs select **Manage: Host Logs** and the Host to be displayed

This will display logs stored locally on the *console server* memory or USB flash.

13.3 Terminal Connection

There are two methods available for accessing the *console server* command line and devices attached to the *console server* serial ports, directly from a web browser:

- The Web Terminal service uses AJAX to enable the web browser to connect to the console server using HTTP or HTTPS, as a terminal - without the need for additional client installation on the user's PC
- The SDT Connector service launches a pre-installed SDT Connector client on the user's PC to establish secure SSH access, then uses pre-installed client software on the client PC to connect to the console server

Web browser access is available to users who are a member of the admin or users groups.

13.3.1 Web Terminal

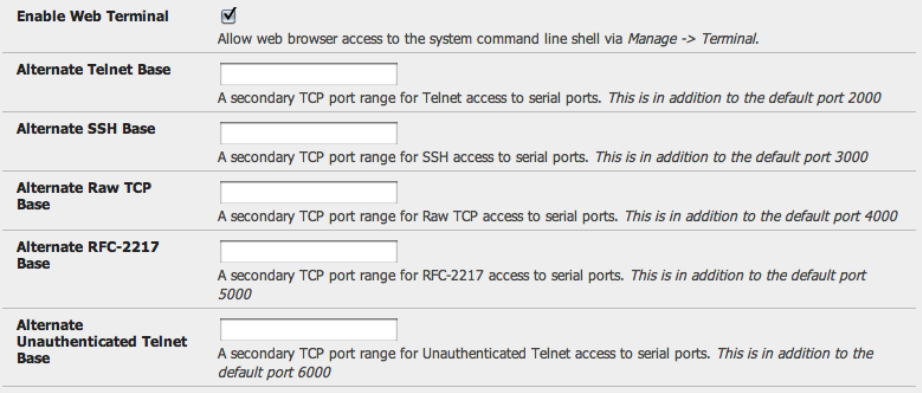
The AJAX based Web Terminal service may be used to access the console server command line or attached serial devices.

Note: Any communication using the Web Terminal service using HTTP is unencrypted and not secure. The Web Terminal connects to the command line or serial device using the same protocol that is being used to browse to the Opengear Management Console, i.e. if you are browsing using an https:/ / URL (this is the default), the Web Terminal connects using HTTPS.

13.3.1.1 Web Terminal to Command Line

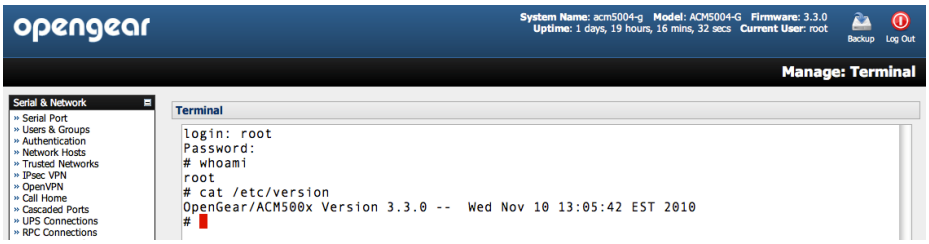
To enable the Web Terminal service for the console server

- Select **System: Firewall**
- Check **Enable Web Terminal** and click **Apply**



*Administrator*s can now communicate directly with the *console server* command line from their browser:

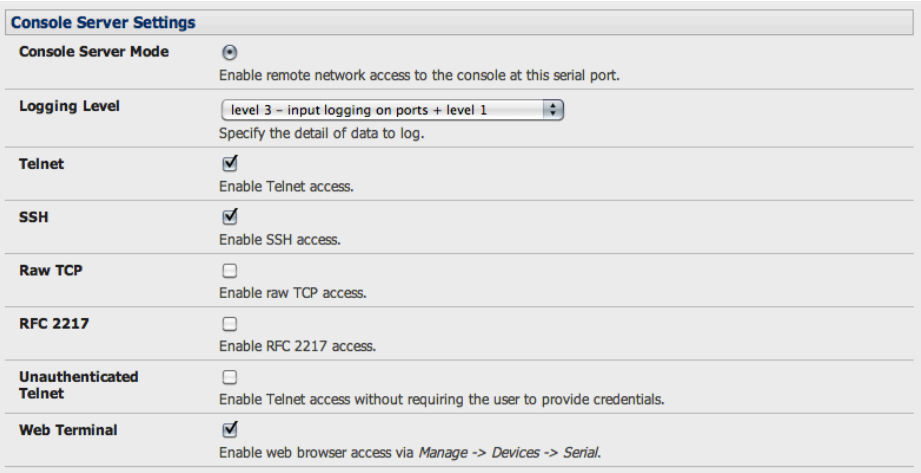
- Select **Manage: Terminal** to display the Web Terminal from which you can log in to the *console server* command line



13.3.1.2 Web Terminal to Serial Device

To enable the Web Terminal service for each serial port you want to access:

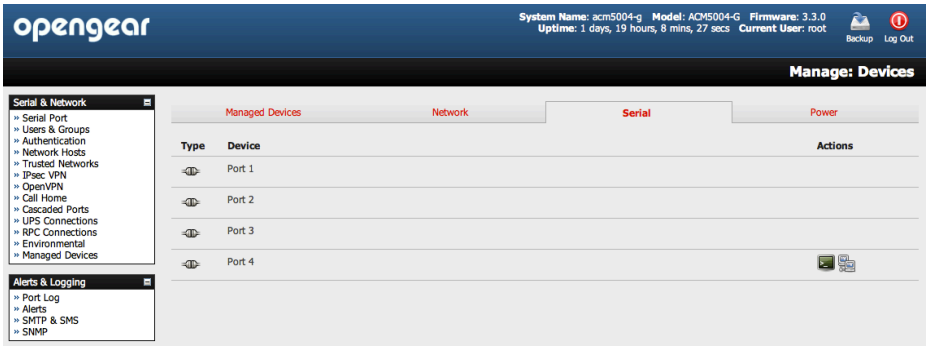
- Select **Serial & Network: Serial Port** and click **Edit**. Ensure the serial port is in *Console Server Mode*
- Check **Web Terminal** and click **Apply**



Administrator and *Users* can communicate directly with serial port attached devices from their browser:

- Select the **Serial** tab on the **Manage: Devices** menu

- Under the *Action* column, click the **Web Terminal** icon to display the Web Terminal, connected directly to the attached serial device

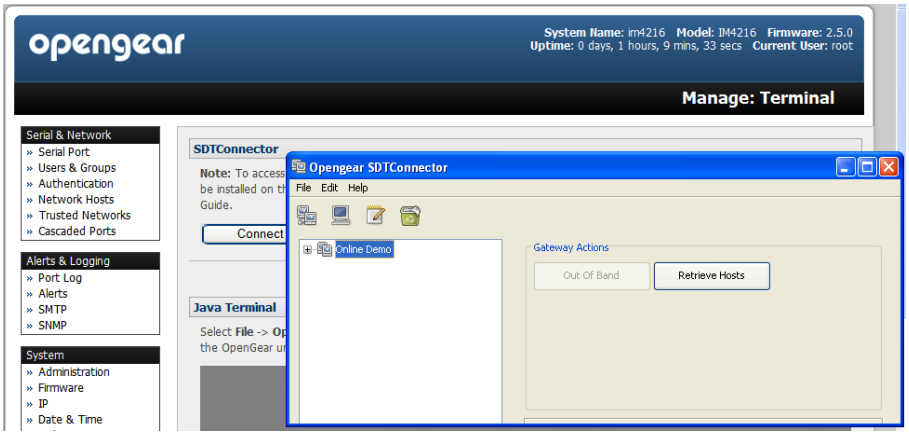


Note The Web Terminal feature was introduced in firmware V3.3. Earlier releases had an open source *jcterm* java terminal applet which could be downloaded into your browser to connect to the *console server* and attached serial port devices. However *jcterm* had some JRE compatibility issues and is no longer supported

13.3.2 SDT Connector access

Administrator and *Users* can communicate directly with the *console server* command line and with devices attached to the *console server* serial ports using SDT Connector and their local tenet client, or using a Web terminal and their browser

- Select **Manage: Terminal**
- Click **Connect to SDT Connector**. This will to activate the SDT Connector client on the computer you are browsing and load your local telnet client to connect to the command line or serial port using SSH

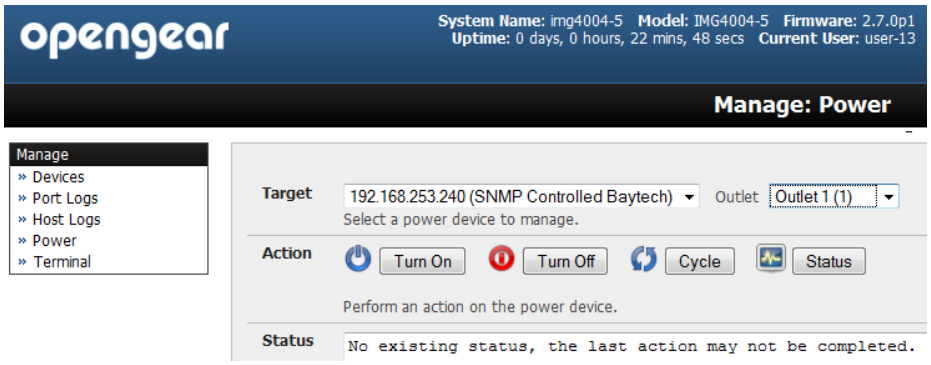


Note SDT Connector must be installed on the computer you are browsing from and the *console server* must be added as a gateway - as detailed in Chapter 6

13.4 Power Management

Administrators and *Users* can access and manage the connected power devices.

- Select **Manage: Power**. This enables the user to power Off/On/Cycle any power outlet on any PDU the user has been given access privileges to (refer *Chapter 8* for details)


opengear
System Name: img4004-5 Model: IMG4004-5 Firmware: 2.7.0p1
Uptime: 0 days, 0 hours, 22 mins, 48 secs Current User: user-13
Manage: Power
Manage
» Devices
» Port Logs
» Host Logs
» Power
» Terminal
Target
192.168.253.240 (SNMP Controlled Baytech)
Outlet
Outlet 1 (1)
Select a power device to manage.
Action
Turn On
Turn Off
Cycle
Status
Perform an action on the power device.
Status
No existing status, the last action may not be completed.

CONFIGURATION FROM THE COMMAND LINE

For those who prefer to configure their *console server* at the Linux command line level (rather than use a browser and the Management Console), this chapter describes using command line access and the ***config*** tool to manage the *console server* and configure the ports etc.

This *config* documentation in this chapter walks thru command line configuration to deliver the functions provided otherwise using the Management Console GUI.

For advanced and custom configurations and for details using other tools and commands refer to the next chapter

When displaying a command, the convention used in the rest of this chapter is to use single quotes ('') for user defined values (e.g. descriptions and names). Element values without single quotes must be typed exactly as shown.

After the initial section on accessing the *config* command the menu items in this document follow the same structure as the menu items in the web GUI.

14.1 Accessing *config* from the command line

The *console server* runs a standard Linux kernel and embeds a suite of open source applications. So if you do not want to use a browser and the Management Console tools, you are free to configure the *console server* and to manage connected devices from the command line using standard Linux and Busybox commands and applications such as *ifconfig, gettyd, stty, powerman, nut etc*. However without care these configurations may not withstand a *power-cycle-reset* or *reconfigure*.

So Opengear provides a number of custom command line utilities and scripts to make it simple to configure the *console server* and ensure the changes are stored in the *console server*'s flash memory etc.

In particular the **config** utility allows manipulation of the system configuration from the command line. With *config* a new configuration can be activated by running the relevant configurator, which performs the action necessary to make the configuration changes live.

To access *config* from the command line:

- Power up the *console server* and connect the “terminal” device:
  - If you are connecting using the serial line, plug a serial cable between the *console server* local DB-9 console port and terminal device. Configure the serial connection of the terminal device you are using to 115200bps, 8 data bits, no parity and one stop bit
  - If you are connecting over the LAN then you will need to interconnect the Ethernet ports and direct your terminal emulator program to the IP address of the *console server* (192.168.0.1 by default)
- Log on to the *console server* by pressing ‘return’ a few times. The *console server* will request a username and password. Enter the username *root* and the password *default*. You should now see the command line prompt which is a hash (#)

This chapter is not intended to teach you Linux. We assume you already have a certain level of understanding before you execute Linux kernel level commands.

The *config* tool

Syntax

config [-ahv] [-d id] [-g id] [-p path] [-r configurator] [-s id=value] [-P id]

Description

The *config* tool is designed to perform multiple actions from one command if need be, so if necessary options can be chained together.

The *config* tool allows manipulation and querying of the system configuration from the command line. Using *config* the new configuration can be activated by running the relevant *configurator* which performs the action necessary to make the configuration changes live.

The custom user configuration is saved in the */etc/config/config.xml* file. This file is transparently accessed and edited when configuring the device using the Management Console browser GUI. Only the user *'root'* can configure from the shell.

By default, the *config* elements are separated by a '.' character. The root of the *config* tree is called <*config*>. To address a specific element place a '.' between each node/branch e.g. to access and display the description of *user1* type:

 # config -g config.users.user1.description

The root node of the *config* tree is <config>. To display the entire *config* tree, type:

 # config -g config

To display the help text for the *config* command, type:

 # config -h

The *config* application resides in the */bin* directory. The environmental variable called *PATH* contains a route to the */bin* directory. This allows a user to simply type *config* at the command prompt instead of the full path */bin/config*.

Options

-a –run-all	Run all registered configurators. This performs every configuration synchronization action pushing all changes to the live system
-h –help	Display a brief usage message
-v –verbose	Log extra debug information
-d –del=id	Remove the given configuration element specified by a '.' separated identifier
-g –get=id	Display the value of a configuration element
-p –path=file	Specify an alternate configuration file to use. The default file is located at */etc/config/config.xml*
-r –run=configurator	Run the specified registered configurator. Registered configurators are listed below.
-s --set=id=value	Change the value of configuration element specified by a '.' separated identifier
-e --export=file	Save active configuration to file
-i --import=file	Load configuration from file
-t --test-import=file	Pretend to load configuration from file
-S --separator=char	The pattern to separate fields with, default is '.'
-P --password=id	Prompt user for a value. Hash the value, then save it in id

The registered configurators are:

alerts
auth
cascade
console
dhcp
dialin
eventlog
hosts
ipaccess
ipconfig
nagios
power
serialconfig
services
slave
systemsettings
time
ups
users

There are three ways to delete a config element value. The simplest way is use the *delete-node* script detailed later in Chapter 15. You can also assign the config element to "", or delete the entire config node using *-d*:

 # /bin/config -d 'element name'

All passwords are saved in plaintext *except* the user passwords and the system passwords, which are encrypted.

Note: The *config* command does not verify whether the nodes edited/added by the user are valid. This means that any node may be added to the tree. If a user were to run the following command:

 # /bin/config -s config.fruit.apple=sweet

The configurator will not complain, but this command is clearly useless. When the configurators are run (to turn the config.xml file into live config) they will simply ignore this <fruit> node. *Administrators* must make sure of the spelling when typing config commands. Incorrect spelling for a node will not be flagged.

Most configurations made to the XML file will be immediately active. To make sure that *all* configuration changes are active, especially when editing user passwords, run all the configurators:

 # /bin/config -a

For information on backing up and restoring the configuration file refer *Chapter 15 Advanced Configuration*.

14.1.1 Serial Port configuration

The first set of configurations that needs to be made to any serial port are the RS232 common settings. For example to setup serial port 5 to use the following properties:

Baud Rate	*9600*
Parity	*None*
Data Bits	*8*
Stop Bits	*1*
label	*Myport*
log level	*0*
protocol	*RS232*
flow control	*None*

To do this use the following commands:

 # config -s config.ports.port5.speed=9600
 # config -s config.ports.port5.parity=None
 # config -s config.ports.port5.charsize=8
 # config -s config.ports.port5.stop=1
 # config -s config.ports.port5.label=myport
 # config -s config.ports.port5.loglevel=0

 # config -s config.ports.port5.protocol=RS232
 # config -s config.ports.port5.flowcontrol=None

The following command will synchronize the live system with the new configuration:

 # config -r serialconfig

Note: Supported serial port baud-rates are ‘50’, ‘75’, ‘110’, ‘134’, ‘150’, ‘200’, ‘300’, ‘600’, ‘1200’, ‘1800’, ‘2400’, ‘4800’, ‘9600’, '19200', '38400', '57600', '115200', and '230400'.
Supported parity values are 'None', 'Odd', 'Even', 'Mark' and 'Space'.
Supported data-bits values are '8', '7', '6' and '5'.
Supported stop-bits values are '1', '1.5' and '2'.
Supported flow-control values are 'Hardware', 'Software' and 'None'.

Additionally, before any port can function properly, the mode of the port needs to be set. Any port can be set to run in one of the five possible modes (refer *Chapter 4* for details): [*Console Server* mode | Device mode | SDT mode | Terminal server mode | Serial bridge mode]. All these modes are mutually exclusive.

Console Server mode

The command to set the port in *portmanager* mode:

 # config -s config.ports.port5.mode=portmanager

To set the following optional config elements for this mode:

Data accumulation period	*100 ms*
Escape character	*% (default is ~)*
log level	*2 (default is 0)*
Shell power command menu	*Enabled*
RFC2217 access	*Enabled*
Limit port to 1 connection	*Enabled*
SSH access	*Enabled*
TCP access	*Enabled*
telnet access	*Disabled*
Unauthorized telnet access	*Disabled*

 # config -s config.ports.port5.delay=100
 # config -s config.ports.port5.escapechar=%
 # config -s config.ports.port5.loglevel=2
 # config -s config.ports.port5.powermenu=on
 # config -s config.ports.port5.rfc2217=on
 # config -s config.ports.port5.singleconn=on
 # config -s config.ports.port5.ssh=on
 # config -s config.ports.port5.tcp=on
 # config -d config.ports.port5.telnet
 # config -d config.ports.port5.unauthtel

Device Mode

For a device mode port, set the port type to either *ups, rpc,* or *enviro:*

 # config -s config.ports.port5.device.type=[ups | rpc | enviro]

For port 5 as a UPS port:

 # config -s config.ports.port5.mode=reserved

For port 5 as an RPC port:

 # config -s config.ports.port5.mode=powerman

For port 5 as an Environmental port:

 # config -s config.ports.port5.mode=reserved

SDT mode

To enable access over SSH to a host connected to serial port 5:

 # config -s config.ports.port5.mode=sdt
 # config -s config.ports.port5.sdt.ssh=on

To configure a username and password when accessing this port with Username = user1 and Password = secret:

 # config -s config.ports.port#.sdt.username=user1
 # config -s config.ports.port#.sdt.password=secret

Terminal server mode

Enable a TTY login for a local terminal attached to serial port 5:

 # config -s config.ports.port5.mode=terminal
 # config -s config.ports.port5.terminal=[vt220 | vt102 | vt100 | linux | ansi]

The default terminal is vt220

Serial bridge mode

Create a network connection to a remote serial port via RFC-2217 on port 5:

 # config -s config.ports.port5.mode=bridge

Optional configurations for the network address of RFC-2217 server of 192.168.3.3 and TCP port used by the RFC-2217 service = 2500:

 # config -s config.ports.port5.bridge.address=192.168.3.3
 # config -s config.ports.port5.bridge.port=2500

To enable RFC-2217 access: *# config -s config.ports.port5.bridge.rfc2217=on*

To redirect the serial bridge over an SSH tunnel to the server: *# config -s config.ports.port5.bridge.ssh.enabled=on*

Syslog settings

Additionally, the global system log settings can be set for any specific port, in any mode:

 # config -s config.ports.port#.syslog.facility='facility'
'facility' can be:
- *Default*
- *local 0-7*
- *auth*
- *authpriv*
- *cron*
- *daemon*
- *ftp*
- *kern*
- *lpr*
- *mail*
- *news*
- *user*
- *uucp*

 # config -s config.ports.port#.syslog.priority='priority'
'priority' can be:
- *Default*
- *warning*
- *notice*
- *Info*
- *error*

emergency
debug
critical
alert

14.1.2 Adding and removing Users

Firstly, determine the total number of existing Users (if you have no existing Users you can assume this is 0):

 # config -g config.users.total

This command should display *config.users.total 1*. Note that if you see *config.users.total* this means you have 0 Users configured.

Your new User will be the existing total plus 1. So if the previous command gave you 0 then you start with user number 1, if you already have 1 user your new user will be number 2 etc.

To add a user (with Username=John, Password=secret and Description =mySecondUser) issue the commands:

 # config -s config.users.total=2 (assuming we already have 1 user configured)
 # config -s config.users.user2.username=John
 # config -s config.users.user2.description=mySecondUser
 # config -P config.users.user2.password

NOTE: The -P parameter will prompt the user for a password, and encrypt it. In fact, the value of any config element can be encrypted using the -P parameter, but only encrypted user passwords and system passwords are supported. If any other element value were to be encrypted, the value will become inaccessible and will have to be re-set.

To add this user to specific groups (admin/users):

 # config -s config.users.user2.groups.group1='groupname'
 # config -s config.users.user2.groups.group2='groupname2'
etc...

To give this user access to a specific port:

 # config -s config.users.user2.port1=on
 # config -s config.users.user2.port2=on
 # config -s config.users.user2.port5=on
etc...

To remove port access:

 # config -s config.users.user2.port1='' (the value is left blank)
or simply:
 # config -d config.users.user2.port1

The port number can be anything from 1 to 48, depending on the available ports on the specific *console server*.

For example assume we have an RPC device connected to port 1 on the *console server* and the RPC is configured. To give this user access to RPC outlet number 3 on the RPC device, run the 2 commands below:

 # config -s config.ports.port1.power.outlet3.users.user2=John
 # config -s config.ports.port1.power.outlet3.users.total=2 (total number of users that have access to this outlet)

If more users are given access to this power outlet, then increment the *'config.ports.port1.power.outlet3.users.total'* element accordingly.

To give this user access to network host 5 (assuming the host is configured):

 # config -s config.sdt.hosts.host5.users.user1=John
 # config -s config.sdt.hosts.host5.users.total=1 (total number of users having access to host)

To give another user called 'Peter' access to the same host:

 # config -s config.sdt.hosts.host5.users.user2=Peter

 # config -s config.sdt.hosts.host5.users.total=2 (total number of users having access to host)

To edit any of the user element values, use the same approach as when adding user elements i.e. use the '-s' parameter. If any of the config elements do not exist, they will automatically be created.

To delete the user called John, use the delete-node script:

 # ./delete-node config.users.user2

The following command will synchronize the live system with the new configuration:

 # config -r users

14.1.3 Adding and removing user Groups

The *console server* is configured with a few default user groups (even though only two of these groups are visible in the Management Console GUI). To find out how many groups are already present:

 # config -g config.groups.total

Assume this value is six. Make sure to number any new groups you create from seven onwards.

To add a custom group to the configuration with Group name=Group7, Group description=MyGroup and Port access= 1,5 you'd issue the commands:

 # config -s config.groups.group7.name=Group7
 # config -s config.groups.group7.description=MyGroup
 # config -s config.groups.total=7
 # config -s config.groups.group7.port1=on
 # config -s config.groups.group7.port5=on

Assume we have an RPC device connected to port 1 on the console manager, and the RPC is configured. To give this group access to RPC outlet number 3 on the RPC device, run the two commands below:

 # config -s config.ports.port1.power.outlet3.groups.group1=Group7
 # config -s config.ports.port1.power.outlet3.groups.total=1 (total number of groups that have access to this outlet)

If more groups are given access to this power outlet, then increment the *'config.ports.port1.power.outlet3.groups.total'* element accordingly.

To give this group access to network host 5:

 # config -s config.sdt.hosts.host5.groups.group1=Group7
 # config -s config.sdt.hosts.host5.groups.total=1 (total number of groups having access to host)

To give another group called 'Group8' access to the same host:

 # config -s config.sdt.hosts.host5.groups.group2=Group8
 # config -s config.sdt.hosts.host5.groups.total=2 (total number of users having access to host)

To delete the group called Group7, use the following command:

 # rmuser Group7

Attention: The *rmuser* script is a generic script to remove any config element from config.xml correctly. However, any dependencies or references to this group will not be affected. Only the group details are deleted. The administrator is responsible for going through *config.xml* and removing group dependencies and references manually, specifically if the group had access to a host or RPC device.

The following command will synchronize the live system with the new configuration:

 # config -a

14.1.4 Authentication

To change the type of authentication for the *console server*:

 # config -s config.auth.type='authtype'

'authtype' can be:

Local
LocalTACACS
TACACS
TACACSLocal
TACACSDownLocal
LocalRADIUS
RADIUS
RADIUSLocal
RADIUSDownLocal
LocalLDAP
LDAP
LDAPLocal
LDAPDownLocal

To configure TACACS authentication:

 # config -s config.auth.tacacs.auth_server='comma separated list' (list of remote authentiction and authorization servers.)
 # config -s config.auth.tacacs.acct_server='comma separated list' (list of remote accounting servers. If unset, Authentication and Authorization Server Address will be used.)
 # config -s config.auth.tacacs.password='password'

To configure RADIUS authentication:

 # config -s config.auth.radius.auth_server='comma separated list' (list of remote authentiction and authorization servers.)
 # config -s config.auth.radius.acct_server='comma separated list' (list of remote accounting servers. If unset, Authentication and Authorization Server Address will be used.)
 # config -s config.auth.radius.password='password'

To configure LDAP authentication:

 # config -s config.auth.ldap.server='comma separated list' (list of remote servers.)
 # config -s config.auth.ldap.basedn='name' (The distinguished name of the search base. For example: dc=my-company,dc=com)
 # config -s config.auth.ldap.binddn='name' (The distinguished name to bind to the server with. The default is to bind anonymously.)
 # config -s config.auth.radius.password='password'

The following command will synchronize the live system with the new configuration:

 # config -r auth

14.1.5 Network Hosts

To determine the total number of currently configured hosts:

 # config -g config.sdt.hosts.total

Assume this value is equal to 3. If you add another host, make sure to increment the total number of hosts from 3 to 4:

 # config -s config.sdt.hosts.total=4

If the output is *config.sdt.hosts.total* then assume 0 hosts are configured.

Add power device host

To add a UPS/RPC network host with the following details:

IP address / DNS name	192.168.2.5
Host name	remoteUPS
Description	UPSroom3
Type	UPS
Allowed services	ssh port 22 and https port 443
Log level for services	0

Issue the commands below:

 # config -s config.sdt.hosts.host4.address=192.168.2.5
 # config -s config.sdt.hosts.host4.name=remoteUPS
 # config -s config.sdt.hosts.host4.description=UPSroom3
 # config -s config.sdt.hosts.host4.device.type=ups
 # config -s config.sdt.hosts.host4.tcpports.tcpport1=22
 # config -s config.sdt.hosts.host4.tcpports.tcpport1.loglevel=0
 # config -s config.sdt.hosts.host4.udpports.udpport2=443
 # config -s config.sdt.hosts.host4.udpports.udpport2.loglevel=0

The *loglevel* can have a value of 0 or 1.

The default services that should be configured are: *22/tcp (ssh), 23/tcp (telnet), 80/tcp (http), 443/tcp (https), 1494/tcp (ica), 3389/tcp (rdp), 5900/tcp (vnc)*

Add other network host

To add any other type of network host with the following details:

IP address / DNS name	192.168.3.10
Host name	OfficePC
Description	MyPC
Allowed sevices	ssh port 22,https port 443
log level for services	1

Issue the commands below. If the Host is not a PDU or UPS power device or a server with IPMI power control then leave the device type blank:

 # config -s config.sdt.hosts.host4.address=192.168.3.10
 # config -s config.sdt.hosts.host4.description=MyPC
 # config -s config.sdt.hosts.host4.name=OfficePC
 # config -s config.sdt.hosts.host4.device.type='' (leave this value blank)
 # config -s config.sdt.hosts.host4.tcpports.tcpport1=22
 # config -s config.sdt.hosts.host4.tcpports.tcpport1.loglevel=1
 # config -s config.sdt.hosts.host4.udpports.tcppport2=443
 # config -s config.sdt.hosts.host4.udpports.tcpport2.loglevel=1

If you want to add the new host as a managed device, make sure to use the current total number of managed devices + 1, for the new device number.

To get the current number of managed devices:

 # config -g config.devices.total

Assuming we already have one managed device, our new device will be device 2. Issue the following commands:

 # config -s config. devices.device2.connections.connection1.name=192.168.3.10
 # config -s config. devices.device2.connections.connection1.type=Host
 # config -s config. devices.device2.name=OfficePC
 # config -s config. devices.device2.description=MyPC

 # config -s config.devices.total=2

The following command will synchronize the live system with the new configuration:

 # config -hosts

14.1.6 Trusted Networks

You can further restrict remote access to serial ports based on the source IP address. To configure this via the command line you need to do the following:

Determine the total number of existing trusted network rules (if you have no existing rules) you can assume this is 0

 # config -g config.portaccess.total

This command should display *config.portaccess.total 1*

Note that if you *see config.portaccess.total* this means you have 0 rules configured.

Your new rule will be the existing total plus 1. So if the previous command gave you 0 then you start with rule number 1. If you already have 1 rule your new rule will be number 2 etc.

If you want to restrict access to serial port 5 to computers from a single class C network (*192.168.5.0* say) you need to issue the following commands (assuming you have a previous rule in place).

Add a trusted network:

 # config -s config.portaccess.rule2.address=192.168.5.0
 # config -s "config.portaccess.rule2.description=foo bar"
 # config -s config.portaccess.rule2.netmask=255.255.255.0
 # config -s config.portaccess.rule2.port5=on
 # config -s config.portaccess.total=2

The following command will synchronize the live system with the new configuration:

 # config -r serialconfig

14.1.7 Cascaded Ports

To add a new slave device with the following settings:

IP address/DNS name	192.168.0.153
Description	CM in office 42
Label	cm7116-5
Number of ports	16

The following commands must be issued:

 # config -s config.cascade.slaves.slave1.address=192.168.0.153
 # config -s "config.cascade.slaves.slave1.description=CM in office 42"
 # config -s config.cascade.slaves.slave1.label=cm7116-5
 # config -s config.cascade.slaves.slave1.ports=16

The total number of slaves must also be incremented. If this is the first slave being added, type:

 # config -s config.cascade.slaves.total=1

Increment this value when adding more slaves.

NOTE: If a slave is added using the CLI, then the master SSH public key will need to be manually copied to every slave device before cascaded ports will work (refer *Chapter 4*)

The following command will synchronize the live system with the new configuration:

 # config -r cascade

14.1.8 UPS Connections

Managed UPSes

Before adding a managed UPS, make sure that at least 1 port has been configured to run in 'device mode', and that the device is set to 'ups'.

To add a managed UPS with the following values:

Connected via	Port 1
UPS name	My UPS
Description	UPS in room 5
Username to connect to UPS	User2
Password to connect to UPS	secret
shutdown order	2 (0 shuts down first)
Driver	genericups
Driver option - option	option
Driver option - argument	argument
Logging	Enabled
Log interval	2 minutes
Run script when power is critical	Enabled

 # config -s config.ups.monitors.monitor1.port=/dev/port01
If the port number is higher than 9, eg port 13, enter:
 # config -s config.ups.monitors.monitor1.port=/dev/port13

 # config -s "config.ups.monitors.monitor1.name=My UPS"
 # config -s "config.ups.monitors.monitor1.description=UPS in room 5"
 # config -s config.ups.monitors.monitor1.username=User2
 # config -s config.ups.monitors.monitor1.password=secret
 # config -s config.ups.monitors.monitor1.sdorder=2
 # config -s config.ups.monitors.monitor1.driver=genericups
 # config -s config.ups.monitors.monitor1.options.option1.opt=option
 # config -s config.ups.monitors.monitor1.options.option1.arg=argument
 # config -s config.ups.monitors.monitor1.options.total=1
 # config -s config.ups.monitors.monitor1.log.enabled=on
 # config -s config.ups.monitors.monitor1.log.interval=2
 # config -s config.ups.monitors.monitor1.script.enabled=on

Make sure to increment the total monitors:

 # config -s config.ups.monitors.total=1

The 5 commands below will add the UPS to 'Managed devices. Assuming there are already 2 managed devices configured:

 # config -s "config.devices.device3.connections.connection1.name=My UPS"
 # config -s "config.devices.device3.connections.connection1.type=UPS Unit"
 # config -s "config.devices.device3.name=My UPS"
 # config -s "config.devices.device3.description=UPS in toom 5"
 # config -s config.devices.total=3

To delete this managed UPS:

 # config -d config.ups.monitors.monitor1

Decrement *monitors.total* when deleting a managed UPS

Remote UPSes

To add a remote UPS with the following details (assuming this is our first remote UPS):

UPS name	oldUPS
Description	UPS in room 2
Address	192.168.50.50
Log status	Disabled
Log rate	240 seconds
Run shutdown script	Enabled

 # config -s config.ups.remotes.remote1.name=oldUPS
 # config -s "config.ups.remotes.remote1.description=UPS in room 2"
 # config -s config.ups.remotes.remote1.address=192.168.50.50
 # config -d config.ups.remotes.remote1.log.enabled
 # config -s config.ups.remotes.remote1.log.interval=240
 # config -s config.ups.remotes.remote1.script.enabled=on
 # config -s config.ups.remotes.total=1

The following command will synchronize the live system with the new configuration:

 # config -a

14.1.9 RPC Connections

You can add an RPC connection from the command line but it is not recommended that you do so because of dependency issues.

However FYI before adding an RPC the Management Console GUI code makes sure that at least 1 port has been configured to run in 'device mode', and that the device is set to 'rpc'.

To add an RPC with the following values:

RPC type	APC 7900
Connected via	Port 2
UPS name	MyRPC
Description	RPC in room 5
Login name for device	rpclogin
Login password for device	secret
SNMP community	v1 or v2c
Logging	Enabled
Log interval	600 second
Number of power outlets	4 (depends on the type/model of the RPC)

 # config -s config.ports.port2.power.type=APC 7900
 # config -s config.ports.port2.power.name=MyRPC
 # config -s "config.ports.port2.power.description=RPC in room 5"
 # config -s config.ports.port2.power.username=rpclogin
 # config -s config.ports.port2.power.password=secret
 # config -s config.ports.port2.power.snmp.community=v1
 # config -s config.ports.port2.power.log.enabled=on
 # config -s config.ports.port2.power.log.interval=600
 # config -s config.ports.port2.power.outlets=4

The following five commands are used by the Management Console to add the RPC to 'Managed Devices':

 # config -s config.devices.device3.connections.connection1.name=myRPC
 # config -s "config.devices.device3.connections.connection1.type=RPC Unit"
 # config -s config.devices.device3.name=myRPC
 # config -s "config.devices.device3.description=RPC in room 5"

 # config -s config.devices.total=3

The following command will synchronize the live system with the new configuration:

 # config -a

14.1.10 Environmental

To configure an environmental monitor with the following details:

Monitor name	Envi4
Monitor Description	Monitor in room 5
Temperature offset	2
Humidity offset	5
Enable alarm 1 ?	yes
Alarm 1 label	door alarm
Enable alarm 2 ?	yes
Alarm 2 label	window alarm
Logging enabled ?	yes
Log interval	120 seconds

 # config -s config.ports.port3.enviro.name=Envi4
 # config -s "config.ports.port3.enviro.description=Monitor in room 5"
 # config -s config.ports.port3.enviro.offsets.temp=2
 # config -s config.ports.port3.enviro.offsets.humid=5
 # config -s config.ports.port3.enviro.alarms.alarm1.alarmstate=on
 # config -s config.ports.port3.enviro.alarms.alarm1.label=door alarm
 # config -s config.ports.port3.enviro.alarms.alarm2.alarmstate=on
 # config -s config.ports.port3.enviro.alarms.alarm2.label=window alarm
 # config -s config.ports.port3.enviro.alarms.total=2
 # config -s config.ports.port3.enviro.log.enabled=on
 # config -s config.ports.port3.enviro.log.interval=120

It is important to assign alarms.total=2 even if they are off.

The following 5 commands will add the environmental monitor to 'Managed devices':

To get the total number of managed devices:

 # config -g config.devices.total

Make sure to use the total + 1 for the new device below:

 # config -s config. devices.device5.connections.connection1.name=Envi4
 # config -s "config. devices.device5.connections.connection1.type=EMD Unit"
 # config -s config. devices.device5.name=Envi4
 # config -s "config. devices.device5.description=Monitor in room 5"
 # config -s config.devices.total=5

The following command will synchronize the live system with the new configuration:

 # config -a

14.1.11 Managed Devices

To add a managed device: (also see UPS, RPC connections and Environmental)

 # config -s "config.devices.device8.name=my device"
 # config -s "config.devices.device8.description=The eighth device"
 # config -s "config.devices.device8.connections.connection1.name=my device"

 # config -s config.devices.device8.connections.connection1.type=[serial | Host | UPS | RPC]
 # config -s config.devices.total=8 (decrement this value when deleting a managed device)

To delete the above managed device:

 # config -d config.devices.device8

The following command will synchronize the live system with the new configuration:

 # config -a

14.11.12 Port Log

To configure serial/network port logging:

 # config -s config.eventlog.server.address='remote server ip address'
 # config -s config.eventlog.server.logfacility='facility'

'facility' can be:

Daemon
Local 0-7
Authentication
Kernel
User
Syslog
Mail
News
UUCP

config -s config.eventlog.server.logpriority='priority'

'*priority*' can be:

Info
Alert
Critical
Debug
Emergency
Error
Notice
Warning

Assume the remote log server needs a username 'name1' and password 'secret':

 # config -s config.eventlog.server.username=name1
 # config -s config.eventlog.server.password=secret

To set the remote path as '/opengear/logs' to save logged data:

 # config -s config.eventlog.server.path=/opengear/logs
 # config -s config.eventlog.server.type=[none | syslog | nfs | cifs | usb]

If the server type is set to usb, none of the other values need to be set. The mount point for storing on a remote USB device is */var/run/portmanager/logdir*

The following command will synchronize the live system with the new configuration:

 # config -a

14.1.13 Alerts

You can add an email, SNMP or NAGIOS alert by following the steps below.

The general settings for all alerts

Assume this is our second alert, and we want to send alert emails to john@opengear.com and sms's to peter@opengear.com:

```
# config -s config.alerts.alert2.description=MySecondAlert
# config -s config.alerts.alert2.email=john@opengear.com
# config -s config.alerts.alert2.email2=peter@opengear.com
```

To use NAGIOS to notify of this alert

```
# config -s config.alerts.alert2.nsca.enabled=on
```

To use SNMP to notify of this alert

```
# config -s config.alerts.alert2.snmp.enabled=on
```

Increment the total alerts:

```
# config -s config.alerts.total=2
```

Below are the specific settings depending on the type of alert required:

Connection Alert

To trigger an alert when a user connects to serial port 5 or network host 3:

```
# config -s config.alerts.alert2.host3='host name'
# config -s config.alerts.alert2.port5=on
# config -s config.alerts.alert2.sensor=temp
# config -s config.alerts.alert2.signal=DSR
# config -s config.alerts.alert2.type=login
```

Signal Alert

To trigger an alert when a signal changes state on port 1:

```
# config -s config.alerts.alert2.port1=on
# config -s config.alerts.alert2.sensor=temp
# config -s config.alerts.alert2.signal=[ DSR | DCD | CTS ]
# config -s config.alerts.alert2.type=signal
```

Pattern Match Alert

To trigger an alert if the regular expression '.*0.0% id' is found in serial port 10's character stream.

```
# config -s "config.alerts.alert2.pattern=.*0.0% id"
# config -s config.alerts.alert2.port10=on
# config -s config.alerts.alert2.sensor=temp
# config -s config.alerts.alert2.signal=DSR
# config -s config.alerts.alert2.type=pattern
```

UPS Power Status Alert

To trigger an alert when *myUPS* (on localhost) or *thatUPS* (on remote host *192.168.0.50*) power status changes between on line, on battery and low battery.

```
# config -s config.alerts.alert2.sensor=temp
# config -s config.alerts.alert2.signal=DSR
# config -s config.alerts.alert2.type=ups
# config -s config.alerts.alert2.ups1=myUPS@localhost
# config -s config.alerts.alert2.ups2=thatUPS@192.168.0.50
```

Environmental and Power Sensor Alert

 # config -s config.alerts.alert2.enviro.high.critical='critical value'
 # config -s config.alerts.alert2.enviro.high.warning='warning value'
 # config -s config.alerts.alert2.enviro.hysteresis='value'
 # config -s config.alerts.alert2.enviro.low.critical='critical value'
 # config -s config.alerts.alert2.enviro.low.warning='warning value'
 # config -s config.alerts.alert2.enviro1='Enviro sensor name'
 # config -s config.alerts.alert2.outlet#='RPCname'.outlet#
'alert2.outlet#' increments sequentially with each added outlet. The second 'outlet#' refers to the specific RPC power outlets.
 # config -s config.alerts.alert2.rpc#='RPC name'
 # config -s config.alerts.alert2.sensor=[temp | humid | load | charge]
 # config -s config.alerts.alert2.signal=DSR
 # config -s config.alerts.alert2.type=enviro
 # config -s config.alerts.alert2.ups1='UPSname@hostname'

Example1: To configure a temperature sensor alert for a sensor called 'SensorInRoom42':

 # config -s config.alerts.alert2.sensor=temp
 # config -s config.alerts.alert2.enviro.high.critical=60
 # config -s config.alerts.alert2.enviro.high.warning=50
 # config -s config.alerts.alert2.enviro.hysteresis=2
 # config -s config.alerts.alert2.enviro.low.critical=5
 # config -s config.alerts.alert2.enviro.low.warning=10
 # config -s config.alerts.alert2.enviro1=SensorInRoom42
 # config -s config.alerts.alert2.signal=DSR
 # config -s config.alerts.alert2.type=enviro

Example2: To configure a load sensor alert for outlets 2 and 4 for an RPC called 'RPCInRoom20':

 # config -s config.alerts.alert2.outlet1='RPCname'.outlet2
 # config -s config.alerts.alert2.outlet2='RPCname'.outlet4
 # config -s config.alerts.alert2.enviro.high.critical=300
 # config -s config.alerts.alert2.enviro.high.warning=280
 # config -s config.alerts.alert2.enviro.hysteresis=20
 # config -s config.alerts.alert2.enviro.low.critical=50
 # config -s config.alerts.alert2.enviro.low.warning=70
 # config -s config.alerts.alert2.rpc1=RPCInRoom20
 # config -s config.alerts.alert2.sensor=load
 # config -s config.alerts.alert2.signal=DSR
 # config -s config.alerts.alert2.type=enviro

Alarm Sensor Alert

To set an alert for 'doorAlarm' and 'windowAlarm' which are two alarms connected to an environmental sensor called 'SensorInRoom3'. Both alarms are disabled on Mondays from 8:15am to 2:30pm:

 # config -s config.alerts.alert2.alarm1=SensorInRoom3.alarm1 (doorAlarm)
 # config -s config.alerts.alert2.alarm1=SensorInRoom3.alarm2 (windowAlarm)
 # config -s config.alerts.alert2.alarmrange.mon.from.hour=8
 # config -s config.alerts.alert2.alarmrange.mon.from.min=15
 # config -s config.alerts.alert2.alarmrange.mon.until.hour=14
 # config -s config.alerts.alert2.alarmrange.mon.until.min=30
 # config -s config.alerts.alert2.description='description'
 # config -s config.alerts.alert2.sensor=temp
 # config -s config.alerts.alert2.signal=DSR
 # config -s config.alerts.alert2.type=alarm

To enable an alarm for the entire day:

 # config -s config.alerts.alert2.alarmrange.mon.from.hour=0
 # config -s config.alerts.alert2.alarmrange.mon.from.min=0
 # config -s config.alerts.alert2.alarmrange.mon.until.hour=0
 # config -s config.alerts.alert2.alarmrange.mon.until.min=0

The following command will synchronize the live system with the new configuration:

 # config -r alerts

14.1.14 SMTP & SMS

To set-up an SMTP mail or SMS server with the following details:

Outgoing server address	mail.opengear.com
Secure connection type	SSL
Sender	John@opengear.com
Server username	john
Server password	secret
Subject line	SMTP alerts

 # config -s config.system.smtp.server=mail.opengear.com
 # config -s config.system.smtp.encryption=SSL (can also be TLS or None)
 # config -s config.system.smtp.sender=John@opengear.com
 # config -s config.system.smtp.username=john
 # config -s config.system.smtp.password=secret
 # config -s config.system.smtp.subject=SMTP alerts

To set-up an SMTP SMS server with the same details as above:

 # config -s config.system.smtp.server2=mail.opengear.com
 # config -s config.system.smtp.encryption2=SSL (can also be TLS or None)
 # config -s config.system.smtp.sender2=John@opengear.com
 # config -s config.system.smtp.username2=john
 # config -s config.system.smtp.password2=secret
 # config -s config.system.smtp.subject2=SMTP alerts

The following command will synchronize the live system with the new configuration:

config -a

14.1.15 SNMP

To set-up the SNMP agent on the device:

 # config -s config.system.snmp.protocol=[UDP | TCP]
 # config -s config.system.snmp.trapport='port number' (default is 162)
 # config -s config.system.snmp.address='NMS IP network address'
 # config -s config.system.snmp.commnity='community name' (v1 and v2c only)
 # config -s config.system.snmp.engineid='ID' (v3 only)
 # config -s config.system.snmp.username='username' (v3 only)
 # config -s config.system.snmp.password='password' (v3 only)
 # config -s config.system.snmp.version=[1 | 2c | 3]

The following command will synchronize the live system with the new configuration:

 # config -a

14.1.16 Administration

To change the administration settings to:

System Name	og.mydomain.com
System Password (root account)	secret
Description	Device in office 2

 # config -s config.system.name=og.mydomain.com
 # config -P config.system.password (will prompt user for a password)
 # config -s "config.system.location=Device in office 2"

NOTE: The -P parameter will prompt the user for a password, and encrypt it. In fact, the value of any config element can be encrypted using the -P parameter, but only encrypted user passwords and system passwords are supported. If any other element value were to be encrypted, the value will become inaccessible and will have to be re-set.

The following command will synchronize the live system with the new configuration:

 # config –a

14.1.17 IP settings

To configure the primary network interface with static settings:

IP address	192.168.0.23
Netmask	255.255.255.0
Default gateway	192.168.0.1
DNS server 1	192.168.0.1
DNS server 2	192.168.0.2

 # config -s config.interfaces.wan.address=192.168.0.23
 # config -s config.interfaces.wan.netmask=255.255.255.0
 # config -s config.interfaces.wan.gateway=192.168.0.1
 # config -s config.interfaces.wan.dns1=192.168.0.1
 # config -s config.interfaces.wan.dns2=192.168.0.2
 # config -s config.interfaces.wan.mode=static
 # config -s config.interfaces.wan.media=[Auto | 100baseTx-FD | 100baseTx-HD | 10baseT-HD] 10baseT-FD

To enable bridging between all interfaces:

 # config -s config.system.bridge.enabled=on

To enable IPv6 for all interfaces

 # config -s config.system.ipv6.enabled=on

To configure the management lan interface, use the same commands as above but replace:

config.interfaces.wan, with config.interfaces.lan

To enable the management lan interface run the following command:

config -d config.interfaces.lan.disabled
config -r ipconfig

Note: Not all devices have a management LAN interface.

To configure a failover device in case of an outage:

 # config -s config.interfaces.wan.failover.address1='ip address'
 # config -s config.interfaces.wan.failover.address2='ip address'
 # config -s config.interfaces.wan.failover.interface=[eth1 | console | modem]

The network interfaces can also be configured automatically:

 # config -s config.interfaces.wan.mode=dhcp
 # config -s config.interfaces.lan.mode=dhcp

The following command will synchronize the live system with the new configuration:

 # /bin/config –-run=ipconfig

The following command will synchronize the live system with the new configuration:

 # config -r ipconfig

14.1.18 Date & Time settings

To enable NTP using a server at pool.ntp.org issue the following commands:

 # config -s config.ntp.enabled=on
 # config -s config.ntp.server=pool.ntp.org

Alternatively, you can manually change the clock settings:

To change running system time:

 # date 092216452005.05 Format is MMDDhhmm[[CC]YY][.ss]

Then the following command will save this new system time to the hardware clock:

/bin/hwclock --systohc

Alternatively, to change the hardware clock:

 # /bin/hwclock --set --date=092216452005.05 Format is MMDDhhmm[[CC]YY][.ss]

Then the following command will save this new hardware clock time as the system time:

 # /bin/hwclock --hctosys

To change the timezone:

 # config -s config.system.timezone=US/Eastern

The following command will synchronize the live system with the new configuration:

 # config -r time

14.1.19 Dial-in settings

To enable dial-in access on the DB9 serial port from the command line with the following attributes:

Local IP Address	172.24.1.1
Remote IP Address	172.24.1.2
Authentication Type:	MSCHAPv2
Serial Port Baud Rate:	115200
Serial Port Flow Control:	Hardware
Custom Modem Initialization:	ATQ0V1H0
Callback phone	0800223665
User to dial as	user1
Password for user	secret

Run the following commands:

 # config -s config.console.ppp.localip=172.24.1.1
 # config -s config.console.ppp.remoteip=172.24.1.2
 # config -s config.console.ppp.auth=MSCHAPv2
 # config -s config.console.speed=115200

 # config -s config.console.flow=Hardware
 # config -s config.console.initstring=ATQ0V1H0
 # config -s config.console.ppp.enabled=on
 # config -s config.console.ppp.callback.enabled=on
 # config -s config.console.ppp.callback.phone1=0800223665
 # config -s config.console.ppp.username=user1
 # config -s config.console.ppp.password=secret

To make the dialed connection the default route:

 # config -s config.console.ppp.defaultroute=on

Please note that supported authentication types are 'None', 'PAP', 'CHAP' and 'MSCHAPv2'.
Supported serial port baud-rates are '9600', '19200', '38400', '57600', '115200', and '230400'.
Supported parity values are 'None', 'Odd', 'Even', 'Mark' and 'Space'.
Supported data-bits values are '8', '7', '6' and '5'.
Supported stop-bits values are '1', '1.5' and '2'.
Supported flow-control values are 'Hardware', 'Software' and 'None'.

If you do not wish to use out-of-band dial-in access please note that the procedure for enabling start-up messages on the console port is covered in Chapter 15 - Accessing the Console Port.

The following command will synchronize the live system with the new configuration:

 # config –a

14.1.20 DHCP server

To enable the DHCP server on the console management LAN, with settings:

Default lease time	200000 seconds
Maximum lease time	300000 seconds
DNS server1	192.168.2.3
DNS server2	192.168.2.4
Domain name	company.com
Default gateway	192.168.0.1
IP pool 1 start address	192.168.0.20
IP pool 1 end address	192.168.0.100
Reserved IP address	192.168.0.50
MAC to reserve IP for	00:1e:67:82:72:d9
Name to identify this host	John-PC

Issue the commands:

 # config -s config.interfaces.lan.dhcpd.enabled=on
 # config -s config.interfaces.lan.dhcpd.defaultlease=200000
 # config -s config.interfaces.lan.dhcpd.maxlease=300000
 # config -s config.interfaces.lan.dhcpd.dns1=192.168.2.3
 # config -s config.interfaces.lan.dhcpd.dns2=192.168.2.4
 # config -s config.interfaces.lan.dhcpd.domain=company.com
 # config -s config.interfaces.lan.dhcpd.gateway=192.168.0.1
 # config -s config.interfaces.lan.dhcpd.pools.pool1.start=192.168.0.20
 # config -s config.interfaces.lan.dhcpd.pools.pool1.end=192.168.0.100
 # config -s config.interfaces.lan.dhcpd.pools.total=1
 # config -s config.interfaces.lan.dhcpd.staticips.staticip1.ip=192.168.0.50
 # config -s config.interfaces.lan.dhcpd.staticips.staticip1.mac=00:1e:67:82:72:d9
 # config -s config.interfaces.lan.dhcpd.staticips.staticip1.host=John-PC
 # config -s config.interfaces.lan.dhcpd.staticips.total=1

The following command will synchronize the live system with the new configuration:

 # config –a

14.1.21 Services

You can manually enable or disable network servers from the command line. For example if you wanted to guarantee the following server configuration:

HTTP Server	Enabled
HTTPS Server	Disabled
Telnet Server	Disabled
SSH Server	Enabled
SNMP Server	Disabled
Ping Replies (Respond to ICMP echo requests)	Disabled
TFTP server	Enabled

 # config -s config.services.http.enabled=on
 # config -d config.services.https.enabled
 # config -d config.services.telnet.enabled
 # config -s config.services.ssh.enabled=on
 # config -d config.services.snmp.enabled
 # config -d config.services.pingreply.enabled
 # config -s config.services.tftp.enabled=on

To set secondary port ranges for any service

 # config -s config.services.telnet.portbase='port base number' Default: 2000
 # config -s config.services.ssh.portbase='port base number' Default: 3000
 # config -s config.services.tcp.portbase='port base number' Default: 4000
 # config -s config.services.rfc2217.portbase='port base number' Default: 5000
 # config -s config.services.unauthtel.portbase='port base number Default: 6000

The following command will synchronize the live system with the new configuration:

config -a

14.1.22 NAGIOS

To configure NAGIOS with the following settings:

NAGIOS host name	cm7116 (Name of this system)
NAGIOS host address	192.168.0.1 (IP to find this device at)
NAGIOS server address	192.168.0.10 (upstream NAGIOS server)
Enable SDT for NAGIOS ext.	Enabled
SDT gateway address	192.168.0.1 (defaults to host address)
Prefer NRPE over NSCA	Disabled (defaults to Disabled)

 # config -s config.system.nagios.enabled=on
 # config -s config.system.nagios.name=cm7116
 # config -s config.system.nagios.address=192.168.0.1
 # config -s config.system.nagios.server.address=192.168.0.10
 # config -s config.system.nagios.sdt.disabled=on (disables SDT for nagios extensions)
 # config -s config.system.nagios.sdt.address=192.168.0.1
 # config -s config.system.nagios.nrpe.prefer=''

To configure NRPE with following settings:

NRPE port	5600 (port to listen on for nrpe. Defaults to 5666)
NRPE user	user1 (User to run as. Defaults to nrpe)

NRPE group	group1 (Group to run as. Defaults to nobody)
Allow command arguments	Enabled

 # config -s config.system.nagios.nrpe.enabled=on
 # config -s config.system.nagios.nrpe.port=5600
 # config -s config.system.nagios.user=user1
 # config -s config.system.nagios.nrpe.group=group1
 # config -s config.system.nagios.nrpe.cmdargs=on

To configure NSCA with the following settings:

NSCA encryption	BLOWFISH (can be: [None | XOR | DES | TRPLEDES | CAST-256 | BLOWFISH | TWOFISH | RIJNDAEL-256 | SERPENT | GOST]
NSCA password	secret
NSCA check-in interval	5 minutes
NSCA port	5650 (defaults to 5667)
user to run as	User1 (defaults to nsca)
group to run as	Group1 (defaults to nobody)

 # config -s config.system.nagios.nsca.enabled=on
 # config -s config.system.nagios.nsca.encryption=BLOWFISH
 # config -s config.system.nagios.nsca.secret=secret
 # config -s config.system.nagios.nsca.interval=2
 # config -s config.system.nagios.nsca.port=5650
 # config -s config.system.nagios.nsca.user=User1
 # config -s config.system.nagios.nsca.group=Group1

The following command will synchronize the live system with the new configuration:

config –a

ADVANCED CONFIGURATION

Opengear *console servers* run the embedded Linux operating system. So *Administrator* class users can configure the *console server* and monitor and manage attached serial console and host devices from the command line using Linux commands and the *config* utility (as described in *Chapter 14*).

The Linux kernel in the *console server* also supports GNU *bash* shell script enabling the *Administrator* to run custom scripts. This chapter presents a number of useful scripts and scripting tools including

- *delete-node* which is a general script for deleting users, groups, hosts, UPS's etc
- *ping-detect* which will run specified commands when a specific host stops responding to ping requests

This chapter then details how to perform advanced and custom management tasks using Opengear commands, Linux commands and the open source tools embedded in the *console server*:

- *portmanager* serial port management
- raw data access to the ports and modems
- *iptables* modifications and updating IP filtering rules
- retrieving status information using SNMP and modifying SNMP with *net-snmpd*
- public key authenticated SSH communications
- SSL, configuring HTTPS and issuing certificates
- using *pmpower* for *NUT* and *PowerMan* power device management
- using *IPMItools*
- CDK custom development kit
- sms server tools
- disable multicasting

15.1 Custom Scripting

The *console server* supports GNU *bash* shell commands (refer *Appendix A*) enabling the *Administrator* to run custom scripts.

15.1.1 Custom script to run when booting

The */etc/config/rc.local* script runs whenever the system boots. By default this script file is empty. You can add any commands to this file if you want them to be run at boot time e.g. if you wanted to display *hello world*:

```
#!/bin/sh
echo "Hello World!"
```

If this script has been copied from a Windows machine you may need to run the following command on the script before *bash* can run it successfully:

```
# dos2unix /etc/config/rc.local
```

Another scenario would be to call another custom script from the */etc/config/rc.local* file, ensuring that your custom script will run whenever the system is booted.

15.1.2 Running custom scripts when alerts are triggered

Whenever an alert gets triggered, specific scripts get called. These scripts all reside in */etc/scripts/*. Below is a list of the default scripts that get run for each applicable alert:

- For a connection alert (when a user connects or disconnects from a port or network host): */etc/scripts/portmanager-user-alert* (for port connections) or */etc/scripts/sdt-user-alert* (for host connections)
- For a signal alert (when a signal on a port changes state): */etc/scripts/portmanager-signal-alert*
- For a pattern match alert (when a specific regular expression is found in the serial ports character stream): */etc/scripts/portmanager-pattern-alert*
- For a UPS status alert (when the UPS power status changes between on line, on battery, and low battery): */etc/scripts/ups-status-alert*
- For an environmental, power and alarm sensor alerts(temperature, humidity, power load and battery charge alerts): */etc/scripts/environmental-alert*
- For an interface failover alert: */etc/scripts/interface-failover-alert*

All of these scripts do a check to see whether you have created a custom script to run instead. The code that does this check is shown below (an extract from the file /etc/scripts/portmanager-pattern-alert):

```
# If there's a user-configured script, run it instead
scripts[0]="/etc/config/scripts/pattern-alert.${ALERT_PORTNAME}"
scripts[1]="/etc/config/scripts/portmanager-pattern-alert"
for (( i=0 ; i < ${#scripts[@]} ; i++ )); do
        if [ -f "${scripts[$i]}" ]; then
                exec /bin/sh "${scripts[$i]}"
        fi
done
```

This code shows that there are two alternative scripts that can be run instead of the default one. This code first checks whether a file "*/etc/config/scripts/pattern-alert.${ALERT_PORTNAME}"* exists. The variable ${ALERT_PORTNAME} must be replaced with "port01" or "port13" or whichever port the alert should run for. If this file cannot be found, the script checks whether the file "*/etc/config/scripts/portmanager-pattern-alert"* exists. If either of these files exists the script calls the *exec* command on the first file that it finds and runs that custom file/script instead.

As an example, you can copy the */etc/scripts/portmanager-pattern-alert* script file to */etc/config/scripts/portmanager-pattern-alert*:

```
# cd /
# mkdir /etc/config/scripts   (if the directory does not already exist)
# cp /etc/scripts/portmanager-pattern-alert /etc/config/scripts/portmanager-pattern-alert
```

The next step will be to edit the new script file. Firstly, open the file */etc/config/scripts/portmanager-pattern-alert* using vi (or any other editor), and remove the lines that check for a custom script (the code from above) - this will prevent the new custom script from repeatedly calling itself. After these lines have been removed, edit the file, or add any additional scripting to the file.

15.1.3 Example script - Power cycling on pattern match

If for example we had an RPC (PDU) connected to port 1 on a *console server* and also have some telecommunications device connected to port 2 and which is powered by the RPC outlet 3. Now assume the telecom device transmits a character stream "EMERGENCY" out on its serial console port every time that it encounters some specific error, and the only way to fix this error is to power cycle the telecom device.

The first step is to setup a pattern-match alert on port 2 to check for the pattern "EMERGENCY".

Next we need to create a custom script to deal with this alert:

```
# cd /
# mkdir /etc/config/scripts   (if the directory does not already exist)
# cp /etc/scripts/portmanager-pattern-alert /etc/config/scripts/portmanager-pattern-alert
```

Note: Make sure to remove the *if* statement (which checks for a custom script) from the new script, in order to prevent an infinite loop.

The *pmpower* utility is used to send power commands to RPC device in order to power cycle our telecom device:

 # pmpower -l port01 -o 3 cycle (The RPC is on serial port 1. The telecom device is powered by RPC outlet 3)

We can now append this command to our custom script. This will guarantee that our telecom device will be power cycled every time the console reads the "EMERGENCY" character stream on port 2.

15.1.4 Example script - Multiple email notifications on each alert

If you desire to send more than one email when an alert triggers, you have to create a replacement script using the method described above and add the appropriate lines to your new script.

Currently, there is a script */etc/scripts/alert-email* which gets run from within all the alert scripts (e.g. *portmanager-user-alert or environmental-alert*). The alert-email script is responsible for sending the email. The line which invokes the email script looks as follows:

/bin/sh /etc/scripts/alert-email $suffix &

If you wish to send another email to a single address or the same email to many recipients, edit the custom script appropriately. You can follow the examples in any of the seven alert scripts listed above. In particular let's consider the *portmanager-user-alert* script. If you need to send the same alert email to more than one email address, find the lines in the script responsible for invoking the alert-email script, then add the following lines below the existing lines:

export TOADDR="emailaddress@domain.com"
/bin/sh /etc/scripts/alert-email $suffix &

These two lines assign a new email address to TOADDR and invoke the alert-email script in the background.

15.1.5 Deleting configuration values from the CLI

The *delete-node* script is provided to help with deleting nodes from the command line. The *"delete-node"* script takes one argument, the node name you want to delete (e.g. *"config.users.user1"* or *"config.sdt.hosts.host1"*).

So *delete-node* is a general script for deleting any node you desire (users, groups, hosts, UPS's etc) from the command line. The script deletes the specified node and shuffles the remainder of the node values.

For example if we have five users configured and we use the script to delete user 3, then user 4 will become user 3, and user 5 will become user 4.

This creates an obvious complication as this script does NOT check for any other dependencies that the node being deleted may have had. So you are responsible for making sure that any references and dependencies connected to the deleted node are removed or corrected in the config.xml file.

The script treats all nodes the same. The syntax to run the script is *# ./delete-node {node name}* so to remove user 3:

 # ./delete-node config.users.user3

The *delete-node* script

```
#!/bin/bash
#User must provide the node to be removed. e.g. "config.users.user1"
# Usage: delete-node {full node path}

if [ $# != 1 ]
then
        echo "Wrong number of arguments"
        echo "Usage: delnode {full '.' delimited node path}"
        exit 2
fi

# test for spaces
TEMP=`echo "$1" | sed 's/.* .*/N/'`
if [ "$TEMP" = "N" ]
then
```

```
        echo "Wrong input format"
        echo "Usage: delnode {full '.' delimited node path}"
        exit 2
fi

# testing if node exists
TEMP=`config -g config | grep "$1"`
if [ -z "$TEMP" ]
then
        echo "Node $1 not found"
        exit 0
fi

# LASTFIELD is the last field in the node path e.g. "user1"
# ROOTNODE is the upper level of the node e.g. "config.users"
# NUMBER is the integer value extracted from LASTFIELD e.g. "1"
# TOTALNODE is the node name for the total e.g. "config.users.total"
# TOTAL is the value of the total number of items before deleting e.g. "3"
# NEWTOTAL is the modified total i.e. TOTAL-1
# CHECKTOTAL checks if TOTAL is the actual total items in .xml

LASTFIELD=${1##*.}
ROOTNODE=${1%.*}
NUMBER=`echo $LASTFIELD | sed 's/^[a-zA-Z]*//g'`
TOTALNODE=`echo ${1%.*} | sed 's/\(.*\)/\1.total/'`
TOTAL=`config -g $TOTALNODE | sed 's/.* //'`
NEWTOTAL=$[ $TOTAL -1 ]

# Make backup copy of config file
cp /etc/config/config.xml /etc/config/config.bak
echo "backup of /etc/config/config.xml saved in /etc/config/config.bak"


if [ -z $NUMBER ] # test whether a singular node is being \
#deleted e.g. config.sdt.hosts
then

        echo "deleting $1"
        config -d "$1"

        echo Done
        exit 0

elif [ $NUMBER = $TOTAL ] # Test if only one item exists
then
        echo "only one item exists"
        # Deleting node
        echo "Deleting $1"
        config -d "$1"

        # Modifying item total.
        config -s "$TOTALNODE=0"

        echo Done
        exit 0
```

```
elif [ $NUMBER -lt $TOTAL ] # more than one item exists
then

        # Modify the users list so user numbers are sequential
        # by shifting the users into the gap one at a time...

        echo "Deleting $1"

        LASTFIELDTEXT=`echo $LASTFIELD | sed 's/[0-9]//g'`
        CHECKTOTAL=`config -g $ROOTNODE.$LASTFIELDTEXT$TOTAL`

        if [ -z "$CHECKTOTAL" ]
        then
                echo "WARNING: "$TOTALNODE" greater than number of items"
        fi

        COUNTER=1
        while [ $COUNTER != $((TOTAL-NUMBER+1)) ]
        do

                config -g $ROOTNODE.$LASTFIELDTEXT$((NUMBER+COUNTER)) \
                | while read LINE
                do
                        config -s \
                        "`echo "$LINE" | sed -e "s/$LASTFIELDTEXT$((NUMBER+ \
                        COUNTER))/$LASTFIELDTEXT$((NUMBER+COUNTER-1))/" \
                        -e 's / /=/'`"

                done

                let COUNTER++
        done

        # deleting last user
        config -d $ROOTNODE.$LASTFIELDTEXT$TOTAL

        # Modifying item total.
        config -s "$TOTALNODE=$NEWTOTAL"

        echo Done
        exit 0
else
        echo "error: item being deleted has an index greater than total items. Increase the total count variable."
        exit 0
fi
```

15.1.6 Power cycle any device upon a ping request failure

The *ping-detect* script is designed to run specified commands when a monitored host stops responding to ping requests.

The first parameter taken by the *ping-detect* script is the hostname / IP address of the device to ping. Any other parameters are then regarded as a command to run whenever the ping to the host fails. *ping-detect* can run any number of commands.

Below is an example using *ping-detect* to power cycle an RPC (PDU) outlet whenever a specific host fails to respond to a ping request. The *ping-detect* is run from */etc/config/rc.local* to make sure that the monitoring starts whenever the system boots.

So if we assume we have a serially controlled RPC connected to port01 on a *console server* and have a router powered by outlet 3 on the RPC (and the router has an internal IP address of 192.168.22.2). The following instructions will show you how to continuously ping the router and when the router fails to respond to a series of pings, the *console server* will send a command to RPC outlet 3 to power cycle the router, and write the current date/time to a file:

- Copy the *ping-detect* script to */etc/config/scripts /* on the *console server*
- Open */etc/config/rc.local* using vi
- Add the following line to *rc.local*:

  /etc/config/scripts/ping-detect 192.168.22.2 /bin/bash -c "pmpower -l port01 -o 3 cycle && date" > /tmp/output.log &

The above command will cause the *ping-detect* script to continuously ping the host at 192.168.22.2 which is the router. If the router crashes it will no longer respond to ping requests. If this happens, the two commands *pmpower* and *date* will run. The output from these commands is sent to the file */tmp/output.log* so that we have some kind of record. The *ping-detect* is also run in the background using the *"&"*.

Remember the *rc.local* script is only run by default when the system boots. You can manually run the *rc.local* script or the *ping-detect* script if desired.

The *ping-detect* script

The above is just one example of using the *ping-detect* script. The idea of the script is to run any number of commands when a specific host stops responding to ping requests. Here are details of the *ping-detect* script itself:

```
#!/bin/sh
# Usage: ping-detect HOST [COMMANDS...]
# This script takes 2 types of arguments: hostname/IPaddress to ping, and the commands to
#  run if the ping fails 5 times in a row. This script can only take one host/IPaddress per
# instance. Multiple independent commands can be sent to the script. The commands will be
# run one after the other.
#
# PINGREP is the entire reply from the ping command
# LOSS is the percentage loss from the ping command
# $1 must be the hostname/IPaddress of device to ping
# $2... must be the commands to run when the pings fail.
COUNTER=0
TARGET="$1"
shift
# loop indefinitely:
while true
do
        # ping the device 10 times
        PINGREP=`ping -c 10 -i 1 "$TARGET" `
        #get the packet loss percentage
        LOSS=`echo "$PINGREP" | grep "%" | sed -e 's/.* \([0-9]*\)% .*/\1/'`
        if [ "$LOSS" -eq "100" ]
        then
                COUNTER=`expr $COUNTER + 1`
        else
                COUNTER=0
                sleep 30s
        fi
        if [ "$COUNTER" -eq 5 ]
        then
                COUNTER=0
```

```
                "$@"
                sleep 2s
        fi
done
```

15.1.7 Running custom scripts when a configurator is invoked

A configurator is responsible for reading the values in */etc/config/config.xml* and making the appropriate changes live. Some changes made by the configurators are part of the Linux configuration itself such as user passwords or *ipconfig*.

Currently there are nineteen configurators each one responsible for a specific group of config e.g. the *"users"* configurator makes the user configurations in the *config.xml file* live. To see all the available configurators type the following from a command line prompt:

```
# config
```

When a change is made using the Management Console web GUI the appropriate configurator is automatically run. This can be problematic as if another user/administrator makes a change using the Management Console the configurator could possibly overwrite any custom CLI/linux configurations you may have set.

The solution is to create a custom script that runs after each configurator has run. So after each configurator runs it will check whether that appropriate custom script exists. You can then add any commands to the custom script and they will be invoked after the configurator runs.

The custom scripts must be in the correct location:

```
/etc/config/scripts/config-post-
```

To create an alerts custom script:

```
# cd /etc/config/scripts
# touch config-post-alerts
# vi config-post-alerts
```

This script could be used to recover a specific backup config or overwrite a config or make copies of config files etc.

15.1.8 Backing-up the configuration and restoring using a local USB stick

The ***/etc/scripts/backup-usb*** script has been written to save and load custom configuration using a USB flash disk. Before saving configuration locally, you must prepare the USB storage device for use. To do this, disconnect all USB storage devices except for the storage device you wish to use.

Usage: */etc/scripts/backup-usb* COMMAND [FILE]

COMMAND:

```
check-magic -- check volume label
set-magic -- set volume label
save [FILE] -- save configuration to USB
delete [FILE] -- delete a configuration tarbal from USB
list -- list available config backups on USB
load [FILE] -- load a specific config from USB
load-default -- load the default configuration
set-default [FILE] -- set which file becomes the default
```

The first thing to do is to check if the USB disk has a label:

```
# /etc/scripts/backup-usb check-magic
```

If this command returns "Magic volume not found", then run the following command:

```
# /etc/scripts/backup-usb set-magic
```

To save the configuration:

 # /etc/scripts/backup-usb save config-20May

To check if the backup was saved correctly:

 # /etc/scripts/backup-usb list

If this command does not display *"* config-20May"* then there was an error saving the configuration.

The set-default command takes an input file as an argument and renames it to "default.opg". This default configuration remains stored on the USB disk. The next time you want to load the default config, it will be sourced from the new default.opg file. To set a config file as the default:

 # /etc/scripts/backup-usb set-default config-20May

To load this default:

 # /etc/scripts/backup-usb load-default

To load any other config file:

 # /etc/scripts/backup-usb load {filename}

The */etc/scripts/backup-usb* script can be executed directly with various *COMMANDS* or called from other custom scripts you may create. However it is recommended that you do not customize the */etc/scripts/backup-usb* script itself at all.

15.1.9 Backing-up the configuration off-box

If you do not have a USB on your *console server* you can back up the configuration to an off-box file. Before backing up you need to arrange a way to transfer the backup off-box. This could be via an NFS share, a Samba (Windows) share to USB storage or copied off-box via the network. If backing up directly to off-box storage, make sure it is mounted.

/tmp is not a good location for the backup except as a temporary location before transferring it off-box. The */tmp* directory will not survive a reboot. The */etc/config* directory is not a good place either, as it will not survive a restore.

Backup and restore should be done by the root user to ensure correct file permissions are set. The config command is used to create a backup tarball:

config -e <Output File>

The tarball will be saved to the indicated location. It will contain the contents of the */etc/config /* directory in an uncompressed and unencrypted form.

Example nfs storage:

 # mount -t nfs 192.168.0.2:/backups /mnt # config -e /mnt/cm7116.config

 # umount/mnt/

Example transfer off-box via scp:

 # config -e /tmp/cm7116.config
 # scp /tmp/cm7116.config username@192.168.0.2:/backups

The config command is also used to restore a backup:

config -i <Input File>

This will extract the contents of the previously created backup to */tmp*, and then synchronize the */etc/config* directory with the copy in */tmp*.

One problem that can crop up here is that there is not enough room in */tmp* to extract files to. The following command will temporarily increase the size of */tmp*:

mount -t tmpfs -o remount,size=2048k tmpfs /var

If restoring to either a new unit or one that has been factory defaulted, it is important to make sure that the process generating SSH keys is either stopped or completed before restoring configuration. If this is not done, then a mix of old and new keys may be put in place.

As SSH uses these keys to avoid man-in-the-middle attacks, logging in may be disrupted.

15.2 Advanced Portmanager

Opengear's *portmanger* program manages the *console server* serial ports. It routes network connection to serial ports, checks permissions, and monitors and logs all the data flowing to/from the ports.

15.2.1 Portmanager commands

pmshell

The *pmshell* command acts similar to the standard *tip* or *cu* commands, but all serial port access is directed *via* the portmanager.

Example: To connect to port 8 *via* the portmanager:

 # pmshell -l port08

pmshell Commands:

Once connected, the pmshell command supports a subset of the '~' escape commands that tip/cu support. For SSH you must prefix the escape with an additional '~' command (i.e. use the '~~' escape)

Send Break: Typing the character sequence '~b' will generate a BREAK on the serial port (if you're doing this over ssh, you'll need to type "~~b")

History: Typing the character sequence '~h' will generate a history on the serial port.

Quit pmshell: Typing the character sequence '~.' will exit from pmshell.

Set RTS to 1 run the command: *pmshell --rts=1*

Show all signals: *# pmshell –signals*

DSR=1 DTR=1 CTS=1 RTS=1 DCD=0

Read a line of text from the serial port: *# pmshell –getline*

Note: V3.5.2 and later firmware has includes *pmshell* chooser escape command so you can now hit *~m* from connected serial port to drop back to *pmshell*

Note: For console servers running firmware V3.11.0 and above *pmshell* has a set of key sequences built in to access things like the power menu, return to the serial port selection menu and so on. Extra controls (key sequences) can be added to the built in set of key sequences and can be configured per serial port. You can have all ports behave the same or selectively add control sequences to ports. The controls can be different from port to port for the same function.

For example, you could configure pmshell such that when you are using serial port 2, pressing *Ctrl+p* would take you straight to the power menu for that port.

The *pmshell* control commands are configured only via the command line.

There is a helper script which will configure a control command on a range of serial ports to eliminate the cumbersome task of entering the configuration command for every port. You will still need to use this script once per control function (see below) but there are only six of these.

pmshell control functions and their built in key sequences:

~*b* - Generate BREAK - send a break to the console

~*h* - View history - see the traffic logs for the port - must have port logging enabled

~*p* - Power menu - open the power menu for the port - port must be configured for an RPC

~m - Connect to port menu - go back to the serial port selection menu

~. - Exit pmshell - exit pmshell completely

~*?* - Show help message - shows the help message

Per Port Control Command Config Parameters:

config.ports.portX.ctrlcode.break - Generate BREAK

config.ports.portX.ctrlcode.portlog - View History

config.ports.portX.ctrlcode.power - Power menu

config.ports.portX.ctrlcode.chooser - Connect to port menu

config.ports.portX.ctrlcode.quit - Exit pmshell

config.ports.portX.ctrlcode.help - Show help message

The *pmshell* help message is NOT updated with the extra control command keys that may be configured. As an example, to configure *Ctrl+p* to open the power menu when using serial port 3, enter the following in the console server's command shell:

config -s config.ports.port3.power=16

killall -HUP portmanager

The first command sets the power menu command to listen for Ctrl+p (where decimal 16 is the character code sent when you press Ctrl+p in the serial port session - see the range of control codes below). The second command (killall -HUP portmanager) tells portmanager to reload the configuration so that the new control code will take affect - rebooting the device would also work.

There is a script to set serial control codes on a range of ports so that bulk port configuration can be performed more easily. For example to set the power menu control code to CTRL-P (keycode 16) on ports 4 to 10 inclusive, enter the following at the command line:

/etc/scripts/set-serial-control-codes 4 10 power 16

This sets the power menu control key to Ctrl+p (see the range of control codes below). NOTE: If you've not configured anything on a particular serial port in the included range, configuration for that port will be skipped.

Control Codes (Ctrl+a=1 ... Ctrl+z=26):

Ctrl+a = 1

Ctrl+b = 2

Ctrl+c = 3

Ctrl+d = 4

Ctrl+e = 5

Ctrl+f = 6

Ctrl+g = 7

Ctrl+h = 8

Ctrl+i = 9

Ctrl+j = 10

Ctrl+k = 11

Ctrl+l = 12

Ctrl+m = 13

Ctrl+n = 14

Ctrl+o = 15

Ctrl+p = 16

Ctrl+q = 17

Ctrl+r = 18

Ctrl+s = 19

Ctrl+t = 20

Ctrl+u = 21

Ctrl+v = 22

Ctrl+w = 23

Ctrl+x = 24

Ctrl+y = 25

Ctrl+z = 26

pmchat

The *pmchat* command acts similar to the standard *chat* command, but all serial port access is directed *via* the portmanager.

Example: To run a chat script *via* the portmanager:

 # pmchat -v -f /etc/config/scripts/port08.chat < /dev/port08

For more information on using *chat* (and *pmchat*) you should consult the UNIX man pages:

http://techpubs.sgi.com/library/tpl/cgibin/getdoc.cgi?coll=linux&db=man&fname=/usr/share/catman/man8/chat.8.html

pmusers

The *pmusers* command is used to query the portmanager for active user sessions.

Example: To detect which users are currently active on which serial ports:

 # pmusers

This command will output nothing if there are no active users currently connected to any ports, otherwise it will respond with a sorted list of usernames per active port:

Port 1:
user1
user2
Port 2:
user1
Port 8:
user2

The above output indicates that a user named "*user1*" is actively connected to ports 1 and 2, while "*user2*" is connected to both ports 1 and 8

Note: With V3.11 firmware and later the *pmusers* command is extended with the *--disconnect* option, which allows an *admin* user or *root* to disconnect console server sessions from the command line. The following connection types can be disconnected:

telnet
SSH
Raw TCP
Unauth'ed Telnet

You cannot disconnect an *RFC2217* session.

If the *--disconnect* option is specified, the *pmusers* command goes into disconnect mode where you can specify the users with *-u*, the ports with *-l* (by *label*) or *-n* (by *name*).

By default the command will prompt the user before actually disconnecting the matching sessions. This can be overriden with the *--no-prompt* argument.

Example: *pmuser* sessions:

```
# pmusers --disconnect
Disconnect all users from all ports? (y/n)
y
5 sessions were disconnected

# pmusers --disconnect -u robertw
Disconnect user robertw from all ports? (y/n)
y
1 session was disconnected

# pmusers --disconnect -u robertw -n 5
Disconnect user robertw from port 5 (BranchRouter01)? (y/n)
y
No sessions were disconnected

# pmusers --disconnect -n 5
Disconnect all users from port 5 (BranchRouter01)? (y/n)
y
2 sessions were disconnected

# pmusers --disconnect -u robertw -u pchunt -n 4 -n 6
Disconnect users robertw, pchunt from ports 4, 6? (y/n)
y
10 sessions were disconnected

# pmusers --disconnect -u tester --no-prompt
No sessions were disconnected
```

portmanager daemon

There is normally no need to stop and restart the daemon. To restart the daemon normally, just run the command:

 # portmanager

Supported command line options are:

Force portmanager to run in the foreground: *--nodaemon*

Set the level of debug logging: *--loglevel={debug,info,warn,error,alert}*

Change which configuration file it uses: *-c /etc/config/portmanager.conf*

Signals

Sending a SIGHUP signal to the portmanager will cause it to re-read its configuration file

15.2.2 External Scripts and Alerts

The portmanager has the ability to execute external scripts on certain events.

When a port is opened by the portmanager:

- When the portmanager opens a port, it attempts to execute */etc/config/scripts/portXX.init* (where XX is the number of the port, *e.g.* 08). The script is run with STDIN and STDOUT both connected to the serial port.
- If the script cannot be executed, then portmanager will execute */etc/config/scripts/portXX.chat via* the chat command on the serial port.

When an alert occurs on a port:

- When an alert occurs on a port, the portmanager will attempt to execute */etc/config/scripts/portXX.alert* (where XX is the port number, e.g. 08)
- The script is run with STDIN containing the data which triggered the alert, and STDOUT redirected to /dev/null, NOT to the serial port. If you wish to communicate with the port, use *pmshell* or *pmchat* from within the script.
- If the script cannot be executed, then the alert will be mailed to the address configured in the system administration section.

When a user connects to any port:

- If a file called */etc/config/pmshell-start.sh* exists it is run when a user connects to a port. It is provided 2 arguments, the "Port number" and the "Username". Here is a simple example:

```
</etc/config/pmshell-start.sh >
#!/bin/sh
PORT="$1"
USER="$2"
echo "Welcome to port $PORT $USER"
< /etc/config/pmshell-start.sh>
```

- The return value from the script controls whether the user is accepted or not, if 0 is returned (or nothing is done on exit as in the above script) the user is permitted, otherwise the user is denied access.
- Here is a more complex script which reads from configuration to display the port label if available and denies access to the root user:

```
</etc/config/pmshell-start.sh>
#!/bin/sh
PORT="$1"
USER="$2"
LABEL=$(config -g config.ports.port$PORT.label | cut -f2- -d' ')
if [ "$USER" == "root" ]; then
        echo "Permission denied for Super User"
        exit 1
fi
if [  -z  "$LABEL" ]; then
echo "Welcome $USER, you are connected to Port $PORT"
else
echo "Welcome $USER, you are connected to Port $PORT ($LABEL)"
fi
</etc/config/pmshell-start.sh>
```

15.3 Raw Access to Serial Ports

15.3.1 Access to serial ports

You can use *tip* and *stty* to completely bypass the *portmanager* and have raw access to the serial ports.

When you run *tip* on a *portmanager* controlled port, *portmanager* closes that port, and stops monitoring it until *tip* releases control of it.

With *stty*, the changes made to the port only "stick" until that port is closed and opened again. So it is doubtful that people will want to use *stty* for more than initial debugging of the serial connection.

If you want to use *stty* to configure the port, you can put *stty* commands in */etc/config/scripts/portXX.init* which gets run whenever portmanager opens the port.

Otherwise, any setup you do with *stty* will get lost when the portmanager opens the port. (the reason that portmanager sets things back to its *config* rather than using whatever is on the port, is so the port is in a known good state, and will work, no matter what things are done to the serial port outside of portmanager).

15.3.2 Accessing the console/modem port

The console dial-in is handled by *mgetty*, with automatic PPP login extensions. *mgetty* is a smart *getty* replacement, designed to be used with Hayes compatible data and data/fax modems. *mgetty* knows about modem initialization, manual modem answering (so your modem doesn't answer if the machine isn't ready), UUCP locking (so you can use the same device for dial-in and dial-out). *mgetty* provides very extensive logging facilities. All standard *mgetty* options are supported.

Modem initialization strings:

- To override the standard modem initialization string either use the Management Console (refer *Chapter 5*) or the command line config tool (refer *Dial-In Configuration Chapter 14*).

Enabling Boot Messages on the Console:

- If you are not using a modem on the DB9 console port and instead wish to connect to it directly via a Null Modem cable you may want to enable verbose mode allowing you to see the standard linux start-up messages. This can be achieved with the following commands:

  # /bin/config --set=config.console.debug=on # /bin/config --run=console # reboot

- If at some point in the future you chose to connect a modem for dial-in out-of-band access the procedure can be reversed with the following commands.

  # /bin/config --del=config.console.debug # /bin/config --run=console # reboot

15.4 IP Filtering

The *console server* uses the *iptables* utility to provide a stateful firewall of LAN traffic. By default rules are automatically inserted to allow access to enabled services, and serial port access *via* enabled protocols. The commands which add these rules are contained in configuration files:

/etc/config/fw.rules

This is an executable shell script which is run whenever the LAN interface is brought up and whenever modifications are made to the *iptables* configuration as a result of CGI actions or the *config* command line tool.

The basic steps performed are as follows:

- Running *iptables* configuration is erased, per-interface and other standard system chains are installed
- Fall through *Block* rules (default deny) are installed
- **Serial & Network: Services** policies are installed in per-interface chains
- Custom **Serial & Network: Firewall** rules are inserted at the top of the rule sets, taking priority over any other configuration

If you require further firewall customization, extra rules can be persisted by creating a file at **/etc/config/scripts/firewall-post** containing *iptables* commands to amend the firewall policy.

There's good documentation about using the *iptables* command at the Linux *netfilter* website *http://netfilter.org/documentation/index.html*.There are also many high-quality tutorials and HOWTOs available *via* the *netfilter* website, in particular peruse the tutorials listed on the *netfilter* HOWTO page.

15.5 SNMP Status Reporting

All console servers contain an SNMP Service (*snmpd*) which can provide status information on demand. *snmpd* is an SNMP agent which binds to a port and awaits requests from SNMP management software. Upon receiving a request, it processes the request(s), collects the requested information and/or performs the requested operation(s) and returns the information to the sender.

Note: Initially only advanced *console server* models were equipped with an SNMP Service. With V3.0 (and later) firmware this support was extended to all *console servers*. Also the MIBS were extended (and renamed for compliance) with this firmware release.

All console servers can also be configured to send SNMP traps/messages to multiple remote SNMP Network Managers on defined trigger events. Refer Chapter 7 for configuration details

15.5.1 Retrieving status information using SNMP

Console servers can provide serial and device status information through SNMP. This includes

- Serial port status
- Active users
- Remote Power Control (RPC) and Power Distribution Unit (PDU) status
- Environmental Monitoring Device (EMD) status
- Signal alert status
- Environmental alert status and
- UPS alert status

The MIBs in your *console server* are located in */etc/snmp/mibs.* You also can view the current MIBs online at / and they include:

OG-STATUS-MIB	This MIB contains serial and connected device status information (for *snmpstatusd* & *snmpalertd*)
OG-STATUSv2-MIB	This new MIB contains extended status and alert
OG-SMI-MIB	Enterprise structure of management information
OGTRAP-MIB	SMIv1 traps from old MIBS (as *smilin*t will not let SMIv1 structures coexist with SMIv2)
OGTRAPv2-MIB	Updated traps

15.5.2 Check firewall rules

- Select **System: Services** and ensure the *SNMP daemon* box has been checked for the interface required. This will allow SNMP requests through the firewall for the specified interface.

15.5.3 Enable SNMP Service

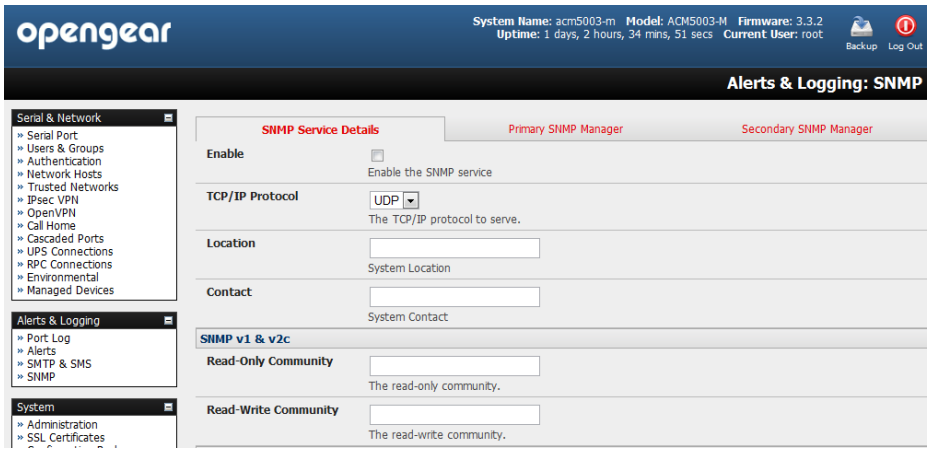
The *console server* supports different versions of SNMP including SNMPv1, SNMPv2c and SNMPv3.

SNMP, although an industry standard, brings with it a variety of security concerns. For example, SNMPv1 and SNMPv2c offer no inherent privacy, while SNMPv3 is susceptible to man-in-the-middle attacks. Recent IETF developments suggests tunnelling SNMP over widely accepted technologies such as SSH (Secure Shell) or TLS (Transport Layer Security) rather than relying on a less mature security systems such as SNMPv3's USM (User-based Security Model).

Additional information regarding SNMP security issues and SNMPv3 can be found at:
http://net-snmp.sourceforge.net/wiki/index.php/TUT:Security

http://www.ietf.org/html.charters/snmpv3-charter.html.

- Select **Alerts & Logging: SNMP**



- The **SNMP Service Details** tab is shown by default. The SNMP Service Details tab controls aspects of the SNMP Service including Security Level. It manages requests from external agents for Opengear status information.
- Check the **Enable the SNMP Service** box to start the SNMP Service. The Service is disabled by default.
- Select either **UDP** or **TCP** for the TCP/IP Protocol. UDP is the recommended protocol and is selected by default. TCP should only be used in special cases such as when Port Forwarding SNMP requests/responses to or from the Opengear device is required.
- Complete the **Location** and **Contact** fields. The Location field should describe the physical location of the Opengear and will be used in response to requests for the SNMPv2-MIB::sysLocation.0 of the device. The Contact field refers to the person responsible for the Opengear such as the System Administrator and will be used in response to requests as follows: SNMPv2-MIB::sysContact.0.
- Enter the **Read-Only Community** and **Read-Write Community.** This is required for **SNMP v1 & v2c** only. The Read-Only Community field is used to specify the SNMPv1 or SNMPv2c community that will be allowed read-only (GET and GETNEXT) access. This must be specified in order for both versions to become enabled. The Read-Write Community field is used to specify the SNMPv1 or SNMPv2c community that will be allowed read-write (GET, GETNEXT and SET) access.
- Configure **SNMP v3,** if required. SNMP v3 provides secure SNMP operations through the use of USM (User-based Security Model). It offers various levels of security including user-based authentication and basic encryption.
  - The **Engine ID** is used to localize the SNMPv3 user. It will be automatically generated from a Network Interface (eth0) hardware address, if left blank, or must be entered as a hex value e.g. 0x01020304.
  - Specify the **Security Level**:

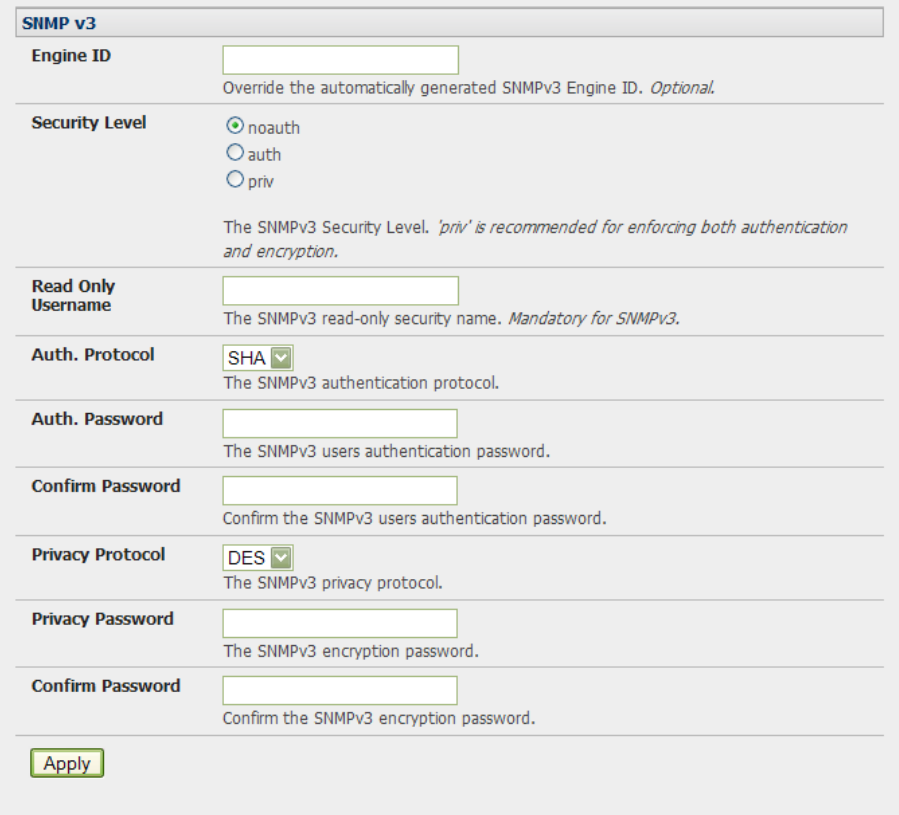
noauth No authentication or encryption is required. This is the minimum level of security.

auth Authentication will be required but encryption is not enforced. An authentication protocol (SHA or MD5) and password will be required.

priv Enforces the use of encryption. This is the highest level of security and requires an encryption protocol (DES or AES) and password in addition to the authentication protocol and password.

- Complete the **Read Only Username.** Enter the read only security name. This field is mandatory and must be completed when configuring the *console server* for SNMPv3.
- For a **Security Level** of **auth**, select the **Auth. Protocol (SHA** or **MD5)** and the **Auth. Password**. A password of at least 8 characters is required.
- For a **Security Level** of **priv**, select the **Privacy Protocol (DES** or **AES)** and the **Privacy Password**. **AES** is recommended as it provides stronger privacy but requires more intense calculations. A password of at least 8 characters is required.

➢ Click **Apply**



➢ Setup serial ports and devices as per operational requirements such as UPS, RPC/PDU and EMD

➢ Copy the mibs from /etc/snmp/mibs on the Opengear product to a local directory using *scp* or W*inscp*. For example:

*scp root@im4004:/etc/snmp/mibs/**

➢ Using the *snmpwalk* and *snmpget* commands, the status information can be retrieved from any console server. For example:

snmpwalk -Oa -v1 -M .:/usr/share/snmp/mibs -c public im4004 OG-STATUS-MIB::ogStatus

```
OG-STATUS-MIB::ogSerialPortStatusPort.1 = INTEGER: 2
OG-STATUS-MIB::ogSerialPortStatusPort.2 = INTEGER: 3
OG-STATUS-MIB::ogSerialPortStatusPort.3 = INTEGER: 4
OG-STATUS-MIB::ogSerialPortStatusSpeed.0 = INTEGER: 9600
OG-STATUS-MIB::ogSerialPortStatusSpeed.1 = INTEGER: 9600
OG-STATUS-MIB::ogSerialPortStatusSpeed.2 = INTEGER: 19200
OG-STATUS-MIB::ogSerialPortStatusSpeed.3 = INTEGER: 9600
OG-STATUS-MIB::ogSerialPortStatusDCD.0 = INTEGER: off(0)
OG-STATUS-MIB::ogSerialPortStatusDCD.1 = INTEGER: off(0)
OG-STATUS-MIB::ogSerialPortStatusDCD.2 = INTEGER: off(0)
OG-STATUS-MIB::ogSerialPortStatusDCD.3 = INTEGER: off(0)
OG-STATUS-MIB::ogSerialPortStatusDTR.0 = INTEGER: off(0)
OG-STATUS-MIB::ogSerialPortStatusDTR.1 = INTEGER: off(0)
OG-STATUS-MIB::ogSerialPortStatusDTR.2 = INTEGER: on(1)
OG-STATUS-MIB::ogSerialPortStatusDTR.3 = INTEGER: on(1)
OG-STATUS-MIB::ogSerialPortStatusDSR.0 = INTEGER: off(0)
OG-STATUS-MIB::ogSerialPortStatusDSR.1 = INTEGER: off(0)
OG-STATUS-MIB::ogSerialPortStatusDSR.2 = INTEGER: off(0)
OG-STATUS-MIB::ogSerialPortStatusDSR.3 = INTEGER: off(0)
OG-STATUS-MIB::ogSerialPortStatusCTS.0 = INTEGER: off(0)
OG-STATUS-MIB::ogSerialPortStatusCTS.1 = INTEGER: off(0)
OG-STATUS-MIB::ogSerialPortStatusCTS.2 = INTEGER: off(0)
OG-STATUS-MIB::ogSerialPortStatusCTS.3 = INTEGER: off(0)
OG-STATUS-MIB::ogSerialPortStatusRTS.0 = INTEGER: off(0)
OG-STATUS-MIB::ogSerialPortStatusRTS.1 = INTEGER: off(0)
OG-STATUS-MIB::ogSerialPortStatusRTS.2 = INTEGER: on(1)
OG-STATUS-MIB::ogSerialPortStatusRTS.3 = INTEGER: on(1)
OG-STATUS-MIB::ogRpcStatusName.0 = STRING: baytech
OG-STATUS-MIB::ogRpcStatusMaxTemp.0 = INTEGER: 0
OG-STATUS-MIB::ogRpcStatusAlertCount.0 = INTEGER: 0
OG-STATUS-MIB::ogEmdStatusName.0 = STRING: EMD_test
OG-STATUS-MIB::ogEmdStatusTemp.0 = INTEGER: 0
OG-STATUS-MIB::ogEmdStatusHumidity.0 = INTEGER: 0
OG-STATUS-MIB::ogEmdStatusAlertCount.0 = INTEGER: 0
OG-STATUS-MIB::ogSignalAlertStatusPort.0 = INTEGER: 4
OG-STATUS-MIB::ogSignalAlertStatusLabel.0 = STRING: port04
OG-STATUS-MIB::ogSignalAlertStatusSignalName.0 = STRING: DSR
OG-STATUS-MIB::ogSignalAlertStatusState.0 = INTEGER: on(1)
OG-STATUS-MIB::ogEnvAlertStatusDevice.0 = STRING: EMD_test
OG-STATUS-MIB::ogEnvAlertStatusDevice.1 = STRING: EMD_test
OG-STATUS-MIB::ogEnvAlertStatusSensor.0 = STRING: a2
OG-STATUS-MIB::ogEnvAlertStatusSensor.1 = STRING: temp
OG-STATUS-MIB::ogEnvAlertStatusOutlet.0 = INTEGER: 0
OG-STATUS-MIB::ogEnvAlertStatusOutlet.1 = INTEGER: 0
OG-STATUS-MIB::ogEnvAlertStatusValue.0 = INTEGER: 1
OG-STATUS-MIB::ogEnvAlertStatusValue.1 = INTEGER: 21
OG-STATUS-MIB::ogEnvAlertStatusOldValue.0 = INTEGER: 0
OG-STATUS-MIB::ogEnvAlertStatusOldValue.1 = INTEGER: 3
OG-STATUS-MIB::ogEnvAlertStatusStatus.0 = INTEGER: 1
OG-STATUS-MIB::ogEnvAlertStatusStatus.1 = INTEGER: 5
```

snmpget -Oa -v1 -M .:/usr/share/snmp/mibs -c public im4004 OG-STATUSMIB:: ogSerialPortStatusSpeed.2

```
OG-STATUS-MIB::ogSerialPortStatusSpeed.2 = INTEGER: 19200
```

noauth

snmpwalk -Oa –v3 –l noAuthNoPriv –u readonlyusername -M .:/usr/share/snmp/mibs im4004 OG-STATUS-MIB::ogStatus

auth

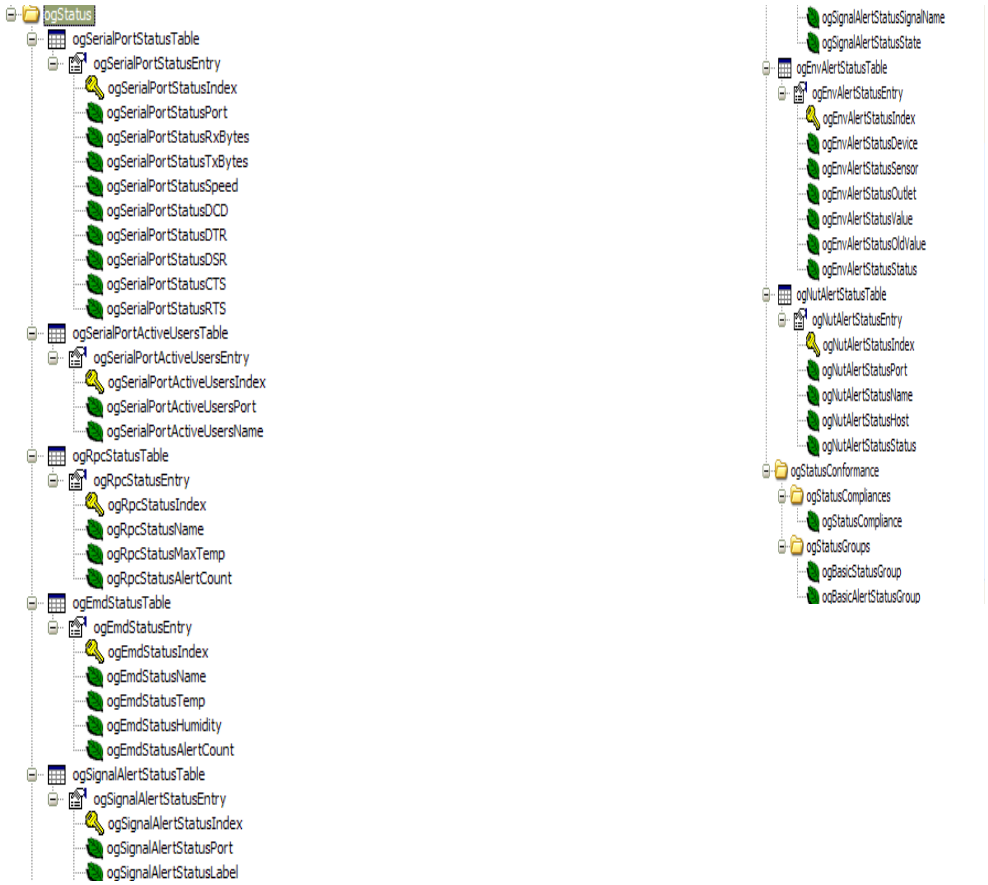
snmpwalk -Oa –v3 –l authNoPriv –u readonlyusername –a SHA –A “authpassword” -M .:/usr/share/snmp/mibs im4004 OG-STATUS-MIB::ogStatus

priv

snmpwalk -Oa –v3 –l authNoPriv –u readonlyusername –a SHA –A “authpassword” –x DES –X “privpassword” -M .:/usr/share/snmp/mibs im4004 OG-STATUS-MIB::ogStatus

-l	Security Level
-u	Security Name or Read Only Username
-a	Authentication Protocol – SHA or MD5
-A	Authentication Password
-x	Privacy Protocol – DES or AES
-X	Privacy Password

A mib browser may be used to explore the Opengear enterprise MIB structure. For example, the *ogStatus* tree is shown below:



15.5.4 Adding multiple remote SNMP managers

You can add multiple SNMP servers for alert traps add the first and second SNMP servers using the Management Console (refer Chapter 7) or the command line *config* tool. Further SNMP servers must be added manually using *config.*

Log in to the *console server*'s command line shell as root or an admin user. Refer back to the Management Console UI or user documentation for descriptions of each field.

To set the SNMP Manager Address field:
config –set="config.system.snmp.address3=w.x.y.z"

.. replacing *w.x.y.z* with the IP address or DNS name.

To set the Manager Trap Port field
config --set="config.system.snmp.trapport3=162"
.. replacing 162 with the TCP/UDP port number

To set the SNMP Manager Protocol field:
config --set="config.system.snmp.protocol3=UDP" or
config --set="config.system.snmp.protocol3=TCP"

To set the SNMP Manager Version field:
config --set="config.system.snmp.version3=3"

To set the SNMP Manager v1 & v2c community field:
config --set="config.system.snmp.community3=public"

To set the SNMP Manager v3 Engine ID field:
config –set="config.system.snmp.engineid3=0x8000000001020304"

.. replacing 0x8000000001020304 with the hex Engine-ID

To set the SNMP Manager v3 Security Level field:
config --set="config.system.snmp.seclevel3=noAuthNoPriv" or
config --set="config.system.snmp.seclevel3=authNoPriv" or
config --set="config.system.snmp.seclevel3=authPriv"

To set the SNMP Manager v3 Username field:
config --set="config.system.snmp.username3=username"

To set the SNMP Manager v3 Auth. Protocol and password fields:
config –set="config.system.snmp.authprotocol3=SHA" or

config --set="config.system.snmp.authprotocol3=MD5"
config --set="config.system.snmp.authpassword3=password 1"

To set the SNMP Manager v3 Privacy Protocol and password fields:
config –set="config.system.snmp.privprotocol3=AES" or

config –set="config.system.snmp.privprotocol3=DES"

config --set="config.system.snmp.privpassword3=password 2"

Once the fields are set, apply the configuration with the following command:
config --run snmp

You can add a third or more SNMP servers by incrementing the "2" in the above commands, e.g. config.system.snmp.protocol3, config.system.snmp.address3, etc

15.6 Secure Shell (SSH) Public Key Authentication

This section covers the generation of public and private keys in a Linux and Windows environment and configuring SSH for public key authentication. The steps to use in a Clustering environment are:

- Generate a new public and private key pair
- Upload the keys to the Master and to each Slave *console server*
- Fingerprint each connection to validate

15.6.1 SSH Overview

Popular TCP/IP applications such as telnet, rlogin, ftp, and others transmit their passwords unencrypted. Doing this across pubic networks like the Internet can have catastrophic consequences. It leaves the door open for eavesdropping, connection hijacking, and other network-level attacks.

Secure Shell (SSH) is a program to log into another computer over a network, to execute commands in a remote machine, and to move files from one machine to another. It provides strong authentication and secure communications over insecure channels.

OpenSSH, the de facto open source SSH application, encrypts all traffic (including passwords) to effectively eliminate these risks. Additionally, OpenSSH provides a myriad of secure tunneling capabilities, as well as a variety of authentication methods.

OpenSSH is the port of OpenBSD's excellent OpenSSH[0] to Linux and other versions of Unix. OpenSSH is based on the last free version of Tatu Ylonen's sample implementation with all patent-encumbered algorithms removed (to external libraries), all known security bugs fixed, new features reintroduced and many other clean-ups. / The only changes in the Opengear SSH implementation are:

- PAM support

- EGD[1]/PRNGD[2] support and replacements for OpenBSD library functions that are absent from other versions of UNIX
- The config files are now in */etc/config. e.g.*
  - */etc/config/sshd_config* instead of */etc/sshd_config*
  - */etc/config/ssh_config* instead of */etc/ssh_config*
  - */etc/config/users/<username>/.ssh* / instead of */home/<username>/.ssh/*

15.6.2 Generating Public Keys (Linux)

To generate new SSH key pairs use the Linux *ssh-keygen* command. This will produce an RSA or DSA public/private key pair and you will be prompted for a path to store the two key files e.g. *id_dsa.pub* (the public key) and *id_dsa* (the private key). For example:

```
$ ssh-keygen -t [rsa|dsa]
Generating public/private [rsa|dsa] key pair.
Enter file in which to save the key (/home/user/.ssh/id_[rsa|dsa]):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/user/.ssh/id_[rsa|dsa].
Your public key has been saved in /home/user/.ssh/id_[rsa|dsa].pub.
The key fingerprint is:
28:aa:29:38:ba:40:f4:11:5e:3f:d4:fa:e5:36:14:d6 user@server
$
```

It is advisable to create a new directory to store your generated keys. It is also possible to name the files after the device they will be used for. For example:

```
$ mkdir keys
$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/user/.ssh/id_rsa): /home/user/keys/control_room
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/user/keys/control_room
Your public key has been saved in /home/user/keys/control_room.pub.
The key fingerprint is:
28:aa:29:38:ba:40:f4:11:5e:3f:d4:fa:e5:36:14:d6 user@server
$
```

You must ensure there is no password associated with the keys. If there is a password, then the Opengear devices will have no way to supply it as runtime.

Full documentation for the *ssh-keygen* command can be found at *http://www.openbsd.org/cgi-bin/man.cgi?query=ssh-keygen*

15.6.3 Installing the SSH Public/Private Keys (Clustering)

For Opengear *console servers* the keys can be simply uploaded through the web interface, on the **System: Administration** page. This enables you to upload stored RSA or DSA Public Key pairs to the Master and apply the Authorized key to the slave and is described in Chapter 4. Once complete you then proceed to Fingerprinting as described below.

15.6.4 Installing SSH Public Key Authentication (Linux)

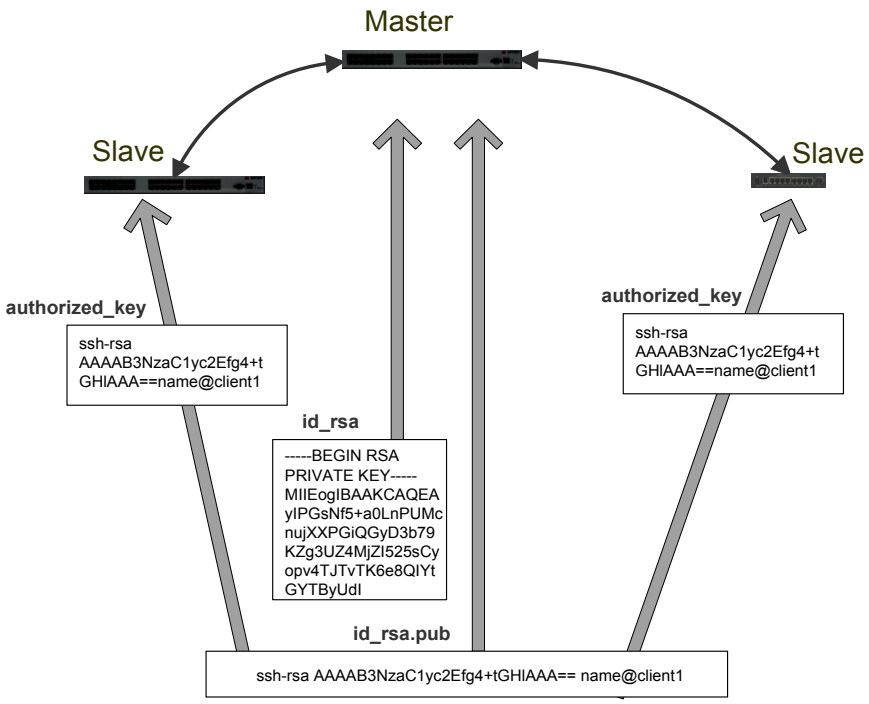
Alternately the public key can be installed on the unit remotely from the linux host with the *scp* utility as follows.

Assuming the user on the Management Console is called "fred"; the IP address of the *console server* is 192.168.0.1 (default); and the public key is on the *linux/unix* computer in *~/.ssh/id_dsa.pub*. Execute the following command on the *linux/unix* computer:

*scp ~/.ssh/id_dsa.pub *
root@192.168.0.1:/etc/config/users/fred/.ssh/authorized_keys

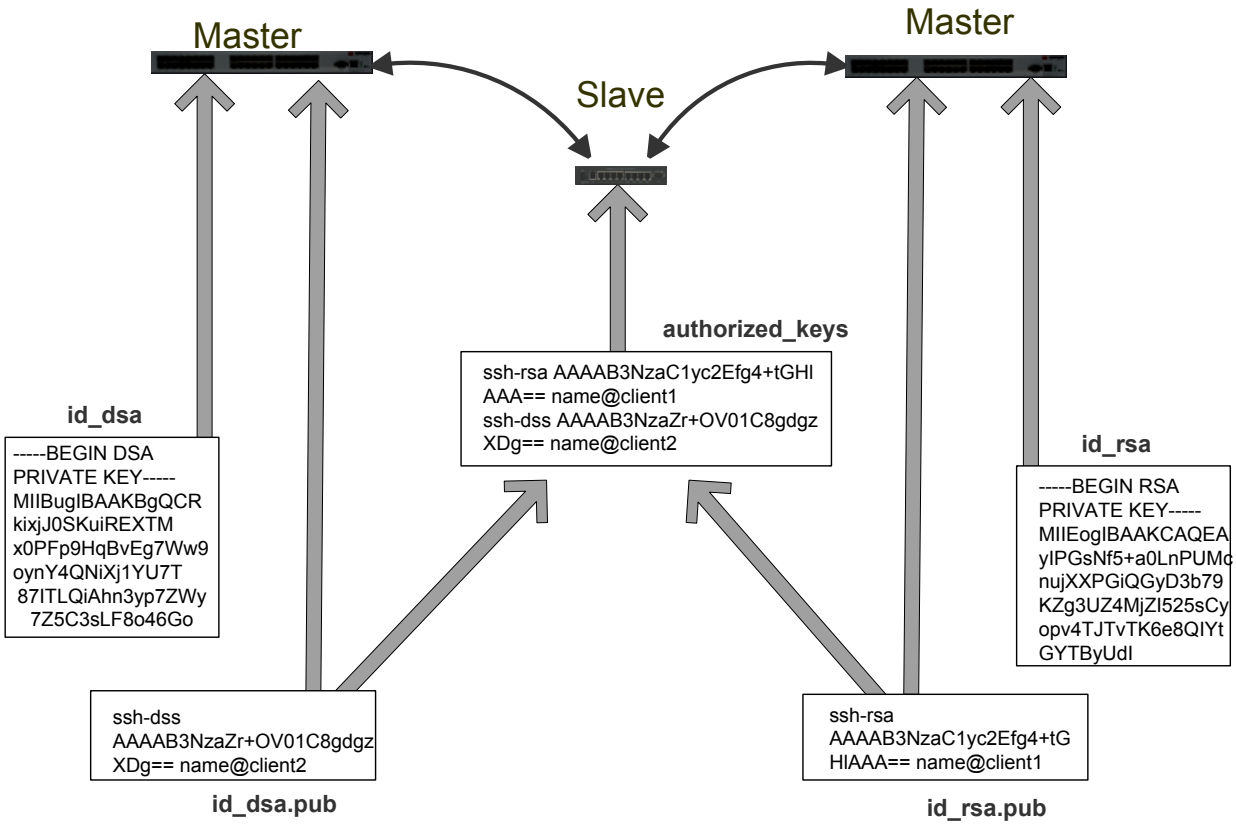
The authorized_keys file on the *console server* needs to be owned by "fred", so login to the Management Console as **root** and type:

chown fred /etc/config/users/fred/.ssh/authorized_keys



If the Opengear device selected to be the server will only have one client device, then the *authorized_keys* file is simply a copy of the public key for that device. If one or more devices will be clients of the server, then the *authorized_keys* file will contain a copy of all of the public keys. RSA and DSA keys may be freely mixed in the authorized_keys file. For example,

assume we already have one server, called bridge_server, and two sets of keys, for the control_room and the plant_entrance:
$ ls /home/user/keys control_room control_room.pub plant_entrance plant_entrance.pub $ cat /home/user/keys/control_room.pub /home/user/keys/plant_entrance.pub > /home/user/keys/authorized_keys_bridge_server



More documentation on OpenSSH can be found at:
http://openssh.org/portable.html
http://www.openbsd.org/cgi-bin/man.cgi?query=ssh&sektion=1
http://www openbsd.org/cgi-bin/man.cgi?query=sshd.

15.6.5 Generating public/private keys for SSH (Windows)

This section describes how to generate and configure SSH keys using Windows.

First create a new user from the Opengear Management (the following example uses a user called "testuser") making sure it is a member of the "users" group.

If you do not already have a public/private key pair you can generate them now using *ssh-keygen*, *PuTTYgen* or a similar tool:

PuTTYgen: *http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html*

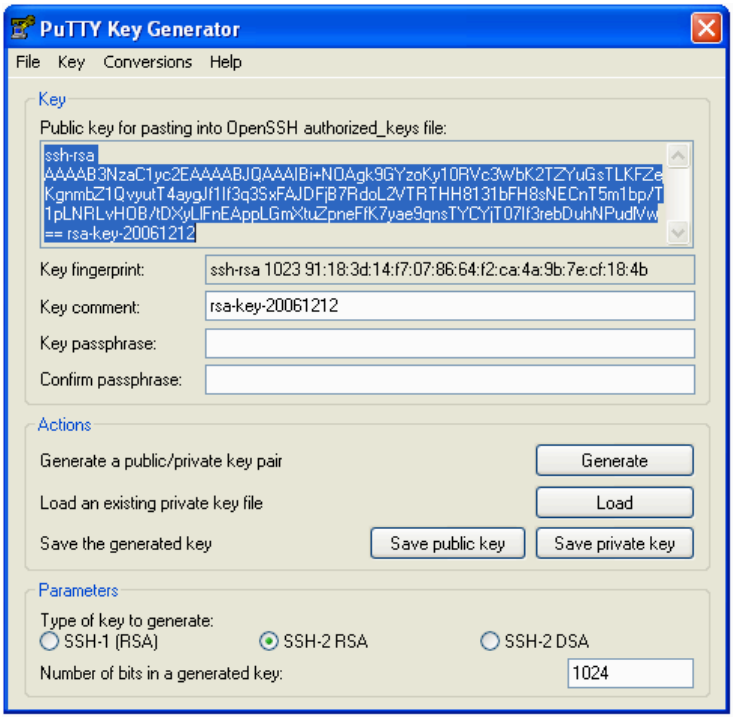
OpenSSH: *http://www.openssh.org/*

OpenSSH (Windows): *http://sshwindows.sourceforge.net/download/*

For example using PuTTYgen, make sure you have a recent version of the *puttygen.exe* (available from *http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html)* Make sure you have a recent version of WinSCP (available from *http://winscp.net/eng/download.php*)

To generate a SSH key using PuTTY *http://sourceforge.net/docs/F02/#clients:*

- Execute the PUTTYGEN.EXE program
- Select the desired key type *SSH2 DSA* (you may use RSA or DSA) within the *Parameters* section
- It is important that you leave the passphrase field blank
- Click on the *Generate* button
- Follow the instruction to move the mouse over the blank area of the program in order to create random data used by PUTTYGEN to generate secure keys. Key generation will occur once PUTTYGEN has collected sufficient random data



- Create a new file " *authorized_keys* " (with notepad) and copy your public key data from the "Public key for pasting into OpenSSH authorized_keys file" section of the PuTTY Key Generator, and paste the key data to the "authorized_keys" file. Make sure there is only one line of text in this file
- Use WinSCP to copy this "authorized_keys" file into the user's home directory: eg. */etc/config/users/testuser/.ssh/authorized_keys* of the Opengear gateway which will be the SSH server. You will need to make sure this file is in the correct format with the correct permissions with the following commands:

  *# dos2unix *
  */etc/config/users/testuser/.ssh/authorized_keys && chown testuser *
  /etc/config/users/testuser/.ssh/authorized_keys

- Using WinSCP copy the attached sshd_config over */etc/config/sshd_config* on the server (Makes sure public key authentication is enabled)
- Test the Public Key by logging in as "testuser" Test the Public Key by logging in as "testuser" to the client Opengear device and typing (you should not need to enter anything): # ssh -o StrictHostKeyChecking=no <server-ip>

To automate connection of the SSH tunnel from the client on every power-up you need to make the *clients /etc/config/rc.local* look like the following:

#!/bin/sh
ssh -L9001:127.0.0.1:4001 -N -o StrictHostKeyChecking=no testuser@<server-ip> &

This will run the tunnel redirecting local port 9001 to the server port 4001.

15.6.6 Fingerprinting

Fingerprints are used to ensure you are establishing an SSH session to who you think you are. On the first connection to a remote server you will receive a fingerprint which you can use on future connections.

This fingerprint is related to the host key of the remote server. Fingerprints are stored in *~/.ssh/known_hosts*.

To receive the fingerprint from the remote server, log in to the client as the required user (usually root) and establish a connection to the remote host:

 # ssh remhost

The authenticity of host 'remhost (192.168.0.1)' can't be established.
RSA key fingerprint is 8d:11:e0:7e:8a:6f:ad:f1:94:0f:93:fc:7c:e6:ef:56.
Are you sure you want to continue connecting (yes/no)?

At this stage, answer yes to accept the key. You should get the following message:

Warning: Permanently added 'remhost,192.168.0.1' (RSA) to the list of known hosts.

You may be prompted for a password, but there is no need to log in - you have received the fingerprint and can Ctrl-C to cancel the connection.If the host key changes you will receive the following warning, and not be allowed to connect to the remote host:

@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@ *WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!* @
@ *IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!* @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

Someone could be eavesdropping on you right now (man-in-the-middle attack)!

It is also possible that the RSA host key has just been changed.

The fingerprint for the RSA key sent by the remote host is

ab:7e:33:bd:85:50:5a:43:0b:e0:bd:43:3f:1c:a5:f8.

Please contact your system administrator.

Add correct host key in */.ssh/known_hosts* to get rid of this message.

Offending key in */.ssh/known_hosts:1*

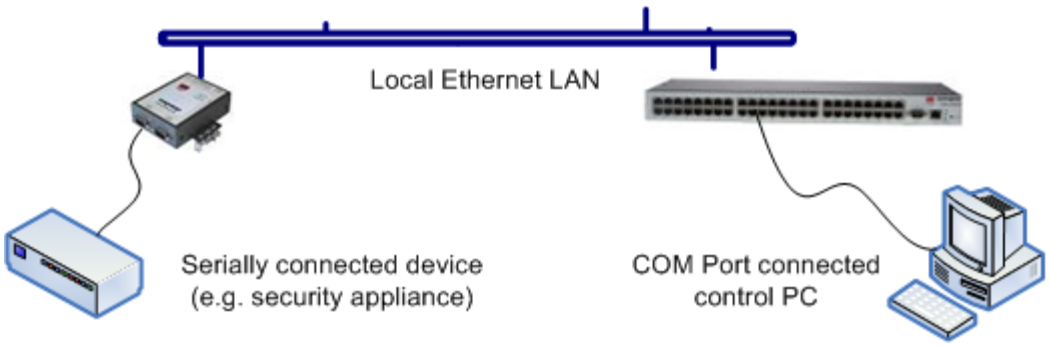
RSA host key for *remhost* has changed and you have requested strict checking.

Host key verification failed.

If the host key has been legitimately changed, it can be removed from the *~/.ssh/known_hosts* file and the new fingerprint added. If it has not changed, this indicates a serious problem that should be investigated immediately.

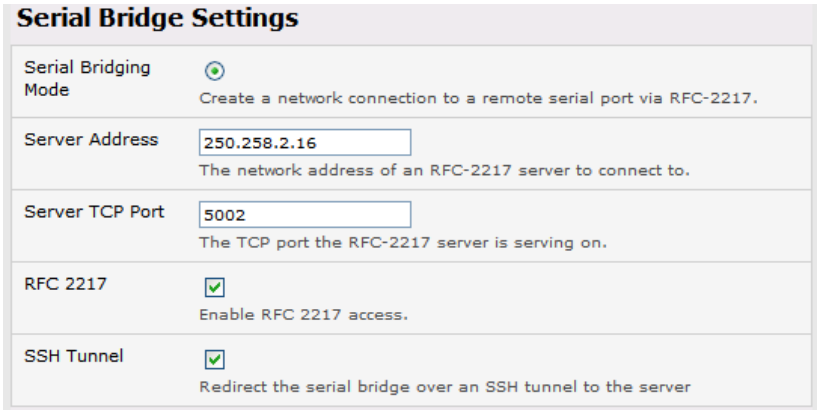
15.6.7 SSH tunneled serial bridging

You have the option to apply SSH tunneling when two Black Box console servers are configured for serial bridging.



As detailed in *Chapter 4*, the *Server* console server is setup in *Console Server* mode with either RAW or RFC2217 enabled and the *Client* console server is set up in Serial Bridging Mode with the Server Address, and Server TCP Port (4000 + port for RAW or 5000 + port # for RFC2217) specified:

- Select **SSH Tunnel** when configuring the **Serial Bridging Setting**



Next you will need to set up SSH keys for each end of the tunnel and upload these keys to the *Server* and *Client* console servers.

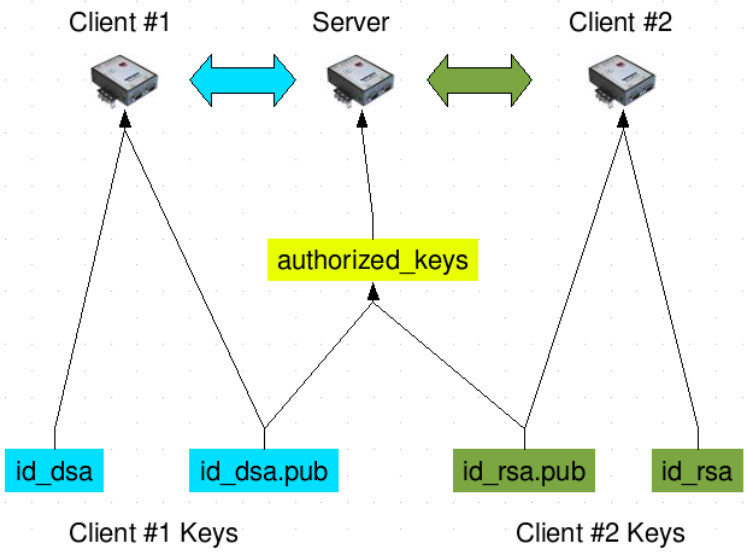
Client Keys:

The first step in setting up ssh tunnels is to generate keys. Ideally, you will use a separate, secure, machine to generate and store all keys to be used on the *console servers*. However, if this is not ideal to your situation, keys may be generated on the *console servers* themselves.

It is possible to generate only one set of keys, and reuse them for every SSH session. While this is not recommended, each organization will need to balance the security of separate keys against the additional administration they bring.

Generated keys may be one of two types - RSA or DSA (and it is beyond the scope of this document to recommend one over the other). RSA keys will go into the files *id_rsa* and *id_rsa.pub*. DSA keys will be stored in the files *id_dsa* and *id_dsa.pub*.

For simplicity going forward the term *private key* will be used to refer to either *id_rsa* or *id_dsa* and *public key* to refer to either *id_rsa.pub* or *id_dsa.pub.*



To generate the keys using OpenBSD's OpenSSH suite, we use the *ssh-keygen* program:

$ ssh-keygen -t [rsa|dsa]
Generating public/private [rsa|dsa] key pair.
Enter file in which to save the key *(/home/user/.ssh/id_[rsa|dsa]):*
Enter *passphrase* (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in */home/user/.ssh/id_[rsa|dsa].*
Your public key has been saved in */home/user/.ssh/id_[rsa|dsa].pub.*
The key fingerprint is:
28:aa:29:38:ba:40:f4:11:5e:3f:d4:fa:e5:36:14:d6 user@server
$

It is advisable to create a new directory to store your generated keys. It is also possible to name the files after the device they will be used for. For example:

$ mkdir keys
$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/user/.ssh/id_rsa): /*home/user/keys/control_room*
Enter *passphrase* (empty for no passphrase):
Enter same *passphrase* again:
Your identification has been saved in */home/user/keys/control_room*
Your public key has been saved in */home/user/keys/control_room.pub*.
The key fingerprint is:
28:aa:29:38:ba:40:f4:11:5e:3f:d4:fa:e5:36:14:d6 user@server
$

You should ensure there is no password associated with the keys. If there is a password, then the *console servers* will have no way to supply it as runtime.

Authorized Keys:

If the *console server* selected to be the server will only have one client device, then the *authorized_keys* file is simply a copy of the public key for that device. If one or more devices will be clients of the server, then the *authorized_keys* file will contain a copy of all of the public keys. RSA and DSA keys may be freely mixed in the *authorized_keys* file.

For example, assume we already have one server, called *bridge_server*, and two sets of keys, for the *control_room* and the *plant_entrance*:

$ ls /home/user/keys

control_room control_room.pub plant_entrance plant_entrance.pub

$ cat /home/user/keys/control_room.pub

/home/user/keys/plant_entrance.pub >

/home/user/keys/authorized_keys_bridge_server

Uploading Keys:

The keys for the server can be uploaded through the web interface, on the **System: Administration** page as detailed earlier. If only one client will be connecting, then simply upload the appropriate public key as the authorized keys file. Otherwise, upload the authorized keys file constructed in the previous step.

Each client will then need its own set of keys uploaded through the same page. Take care to ensure that the correct type of keys (DSA or RSA) goes in the correct spots, and that the public and private keys are in the correct spot.

15.6.8 SDT Connector Public Key Authentication

SDT Connector can authenticate against a *console server* using your SSH key pair rather than requiring your to enter your password (i.e. public key authentication).

- To use public key authentication with SDT Connector, first you must first create an RSA or DSA key pair (using *ssh-keygen, PuTTYgen* or a similar tool) and add the public part of your SSH key pair to the *console server* – as described in the earlier section.
- Next, add the private part of your SSH key pair (this file is typically named *id_rsa* or *id_dsa*) to SDT Connector client. Click **Edit: Preferences: Private Keys: Add**, locate the private key file and click **OK**. You do not have to add the public part of your SSH key pair, it is calculated using the private key.

SDT Connector will now use public key authentication when SSH connecting through the *console server*. You may have to restart SDT Connector to shut down any existing tunnels that were established using password authentication.

If you have a host behind the *console server* that you connect to by clicking the SSH button in SDT Connector, you can also configure it for public key authentication. Essentially what you are using is SSH over SSH, and the two SSH connections are entirely separate, and the host configuration is entirely independent of SDT Connector and the *console server*. You must configure the SSH client that SDT Connector launches (e.g. Putty, OpenSSH) and the host's SSH server for public key authentication.

15.7 Secure Sockets Layer (SSL) Support

Secure Sockets Layer (SSL) is a protocol developed by Netscape for transmitting private documents *via* the Internet. SSL works by using a private key to encrypt data that's transferred over the SSL connection.

The *console server* includes OpenSSL. The OpenSSL Project is a collaborative effort to develop a robust, commercial-grade, full-featured, and Open Source toolkit implementing the Secure Sockets Layer (SSL v2/v3) and Transport Layer Security (TLS v1) protocols as well as a full-strength general purpose cryptography library. The project is managed by a worldwide community of volunteers that use the Internet to communicate, plan, and develop the OpenSSL toolkit and its related documentation.

OpenSSL is based on the excellent Slay library developed by Eric A. Young and Tim J. Hudson. The OpenSSL toolkit is licensed under an Apache-style license, which basically means that you are free to get and use it for commercial and non-commercial purposes subject to some simple license conditions. In the *console server* OpenSSL is used primarily in conjunction with 'http' in order to have secure browser access to the GUI management console across insecure networks.

More documentation on OpenSSL is available from:

http://www.openssl.org/docs/apps/openssl.html

http://www.openssl.org/docs/HOWTO/certificates.txt

15.8 HTTPS

The Management Console can be served using HTTPS by running the webserver *via stunnel*. The server can be launched on request using *inetd*.

The HTTP server is a *lighttpd* server (early versions used *fnord-httpd).*

The SSL implementation is provided by *stunnel* (early versions used *sslwrap* compiled with OpenSSL support).

If your default network address is changed or the unit is to be accessed *via* a known Domain Name you can use the following steps to replace the default SSL Certificate and Private Key with ones tailored for your new address.

15.8.1 Generating an encryption key

To create a 1024 bit RSA key with a password issue the following command on the command line of a linux host with the *openssl* utility installed:

openssl genrsa -des3 -out ssl_key.pem 1024

15.8.2 Generating a self-signed certificate with OpenSSL

This example shows how to use OpenSSL to create a self-signed certificate. OpenSSL is available for most Linux distributions *via* the default package management mechanism. (Windows users can check *http://www.openssl.org/related/binaries.html*)

To create a 1024 bit RSA key and a self-signed certificate issue the following *openssl* command from the host you have *openssl* installed on:

openssl req -x509 -nodes -days 1000 \

-newkey rsa:1024 -keyout ssl_key.pem -out ssl_cert.pem

You will be prompted to enter a lot of information. Most of it doesn't matter, but the "Common Name" should be the domain name of your computer (*e.g.* test.opengear.com). When you have entered everything, the certificate will be created in a file called *ssl_cert.pem*.

15.8.3 Installing the key and certificate

The recommended method for copying files securely to the *console server* unit is with an SCP (Secure Copying Protocol) client. The *scp* utility is distributed with OpenSSH for most Unix distributions while Windows users can use something like the PSCP command line utility available with PuTTY.

The files created in the steps above can be installed remotely with the *scp* utility as follows:

scp ssl_key.pem root@<address of unit>:/etc/config/
scp ssl_cert.pem root@<address of unit>:/etc/config/

or using PSCP:

pscp -scp ssl_key.pem root@<address of unit>:/etc/config/
pscp -scp ssl_cert.pem root@<address of unit>:/etc/config/

PuTTY and the PSCP utility can be downloaded from: *http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html*

More detailed documentation on the PSCP can be found:
http://the.earth.li/~sgtatham/putty/0.58/htmldoc/Chapter5.html#pscp

15.8.4 Launching the HTTPS Server

Note that the easiest way to enable the HTTPS server is from the web Management Console. Simply click the appropriate checkbox in **Network: Services: HTTPS Server** and the HTTPS server will be activated (assuming the *ssl_key.pem* & *ssl_cert.pem* files exist in the */etc/config* directory).

Alternatively *inetd* can be configured to launch the secure *fnord* server from the command line of the unit as follows.

Edit the *inetd* configuration file. From the unit command line:

vi /etc/config/inetd.conf

Append a line:

443 stream tcp nowait root sslwrap -cert /etc/config/ssl_cert.pem -key /etc/config/ssl_key.pem -exec /bin/httpd /home/httpd"

Save the file and signal *inetd* of the configuration change.

kill -HUP `cat /var/run/inetd.pid`

The HTTPS server should be accessible from a web client at a URL similar to this: *https://<common name of unit>*

More detailed documentation about the *openssl* utility can be found at the website: *http://www.openssl.org/*

15.9 Power Strip Control

The *console server* supports a growing list of remote power-control devices (RPCs) which can be configured using the Management Console as described in Chapter 8. These RPCs are controlled using the open source *PowerMan* and *Network UPS Tools* and with Opengear's *pmpower* utility.

15.9.1 The PowerMan tool

PowerMan provides power management in a data center or compute cluster environment. It performs operations such as power on, power off, and power cycle via remote power controller (RPC) devices.

Synopsis
powerman [-option] [targets]
pm [-option] [targets]

Options
-1, --on Power ON targets.
-0, --off Power OFF targets.
-c, --cycle Power cycle targets.
-r, --reset Assert hardware reset for targets (if implemented by RPC).
-f, --flash Turn beacon ON for targets (if implemented by RPC).
-u, --unflash Turn beacon OFF for targets (if implemented by RPC).
-l, --list List available targets. If possible, output will be compressed into a host range (see TARGET SPECIFICATION below).
-q, --query Query plug status of targets. If none specified, query all targets. Status is not cached; each time this option is used, powerman queries the appropriate RPC's. Targets connected to RPC's that could not be contacted (e.g. due to network failure) are reported as status "unknown". If possible, output will be compressed into host ranges.
-n, --node Query node power status of targets (if implemented by RPC). If no targets specified, query all targets. In this context, a node in the OFF state could be ON at the plug but operating in standby power mode.
-b, --beacon Query beacon status (if implemented by RPC). If no targets are specified, query all targets.
-t, --temp Query node temperature (if implemented by RPC). If no targets are specified, query all targets. Temperature information is not interpreted by powerman and is reported as received from the RPC on one line per target, prefixed by target name.
-h, --help Display option summary.
-L, --license Show powerman license information.
-d, --destination host[:port] Connect to a powerman daemon on non-default host and optionally port.
-V, --version Display the powerman version number and exit.
-D, --device Displays RPC status information. If targets are specified, only RPC's matching the target list is displayed.
-T, --telemetry Causes RPC telemetry information to be displayed as commands are processed. Useful for debugging device scripts.
-x, --exprange Expand host ranges in query responses.

For more details refer *http://linux.die.net/man/1/powerman*
Also refer *powermand* (*http://linux.die.net/man/1/powermand)* documentation and *powerman.conf* (*http://linux.die.net/man/5/powerman.conf*)

Target Specification
powerman target hostnames may be specified as comma separated or space separated hostnames or host ranges. Host ranges are of the general form: prefix[n-m,l-k,...], where n < m and l < k, etc., This form should not be confused with regular expression character classes (also denoted by "[]"). For example, foo[19] does not represent foo1 or foo9, but rather represents a degenerate range: foo19.

This range syntax is meant only as a convenience on clusters with a prefix NN naming convention and specification of ranges should not be considered necessary -- the list foo1,foo9 could be specified as such, or by the range foo[1,9].

Some examples of powerman targets follow.

Power on hosts bar,baz,foo01,foo02,...,foo05: *powerman --on bar baz foo[01-05]*

Power on hosts bar,foo7,foo9,foo10: *powerman --on bar,foo[7,9-10]*

Power on foo0,foo4,foo5: *powerman --on foo[0,4-5]*

As a reminder to the reader, some shells will interpret brackets ([and]) for pattern matching. Depending on your shell, it may be necessary to enclose ranged lists within quotes. For example, in tcsh, the last example above should be executed as:

powerman --on "foo[0,4-5]"

15.9.2 The pmpower tool

The *pmpower* utility is a high level tool for manipulating remote preconfigured power devices connected to the *console server* either via a serial or network connection. The PDU UPS and IPMI power devices are variously controlled using the open source *PowerMan, IPMItool* or *Network UPS Tools* and Opengear's *pmpower* utility arches over these tools so the devices can be controlled through the one command line:

pmpower [-?h] [-l device | -r host] [-o outlet] [-u username] [-p password] action

-?/-h This help message.
-l The serial port to use.
-o The outlet on the power target to apply to
-r The remote host address for the power target
-u Override the configured username
-p Override the configured password
on This *action* switches the specified device or outlet(s) on
off This *action* switches the specified device or outlet(s) off
cycle This *action* switches the specified device or outlet(s) off and on again
status This *action* retrieves the current status of the device or outlet

Examples:

To turn outlet 4 of the power device connected to serial port 2 on: *# pmpower -l port02 -o 4 on*

To turn an IPMI device off located at IP address 192.168.1.100 (where username is 'root' and password is 'calvin': *# pmpower -r 192.168.1.100 -u root -p calvin off*

Default system Power Device actions are specified in */etc/powerstrips.xml*. Custom Power Devices can be added in */etc/config/powerstrips.xml*. If an action is attempted which has not been configured for a specific Power Device *pmpower* will exit with an error.

15.9.3 Adding new RPC devices

There are a number of simple paths to adding support for new RPC devices.

The first is to have scripts to support the particular RPC included in either the open source *PowerMan* project *(http://sourceforge.net/projects/powerman)* or the open source *NUT UPS Tools* project. The *PowerMan* device specifications are rather weird and it is suggested that you leave the actual writing of these scripts to the PowerMan authors. However documentation on how they work can be found at *http://linux.die.net/man/5/powerman.dev*. The *Network UPS Tools(NUT)* project has recently moved on from its UPS management origins to also cover SNMP PDUs (and embrace PowerMan). Opengear progressively includes the updated *PowerMan* and *NUT* build into the *console server* firmware releases.

The second path is to directly add support for the new RPC devices (or to customize the existing RPC device support) on your particular *console server*. The **Manage: Power** page uses information contained in */etc/powerstrips.xml* to configure and control devices attached to a serial port. The configuration also looks for (and loads) */etc/config/powerstrips.xml* if it exists.

The user can add their own support for more devices by putting definitions for them into */etc/config/powerstrips.xml*. This file can be created on a host system and copied to the Management Console device using *scp*. Alternatively, login to the Management Console and use *ftp* or *wget* to transfer files.

Here is a brief description of the elements of the XML entries in */etc/config/powerstrips.xml*.

```
<powerstrip>
        <id>Name or ID of the device support</id>
        <outlet port="port-id-1">Display Port 1 in menu</outlet>
        <outlet port="port-id-2">Display Port 2 in menu</outlet>
        ...
        <on>script to turn power on</on>
        <off>script to power off</off>
        <cycle>script to cycle power</cycle>
        <status>script to write power status to /var/run/power-status</status>
        <speed>baud rate</speed>
        <charsize>character size</charsize>
        <stop>stop bits</stop>
        <parity>parity setting</parity>
</powerstrip>
```

The *id* appears on the web page in the list of available devices types to configure.

The outlets describe targets that the scripts can control. For example a power control board may control several different outlets. The port-id is the native name for identifying the outlet. This value will be passed to the scripts in the environment variable *outlet*, allowing the script to address the correct outlet.

There are four possible scripts: *on, off, cycle* and *status*.

When a script is run, it's standard input and output is redirected to the appropriate serial port. The script receives the outlet and port in the *outlet* and *port* environment variables respectively.

The script can be anything that can be executed within the shell.

All of the existing scripts in */etc/powerstrips.xml* use the *pmchat* utility.

pmchat works just like the standard unix "chat" program, only it ensures interoperation with the port manager.

The final options, *speed, charsize, stop* and *parity* define the recommended or default settings for the attached device.

15.10 IPMItool

The *console server* includes the *ipmitool* utility for managing and configuring devices that support the Intelligent Platform Management Interface (IPMI) version 1.5 and version 2.0 specifications.

IPMI is an open standard for monitoring, logging, recovery, inventory, and control of hardware that is implemented independent of the main CPU, BIOS, and OS. The service processor (or Baseboard Management Controller, BMC) is the brain behind platform management and its primary purpose is to handle the autonomous sensor monitoring and event logging features.

The *ipmitool* program provides a simple command-line interface to this BMC. It features the ability to read the sensor data repository (SDR) and print sensor values, display the contents of the System Event Log (SEL), print Field Replaceable Unit (FRU) inventory information, read and set LAN configuration parameters, and perform remote chassis power control.

SYNOPSIS

ipmitool [**-c**|**-h**|**-v**|**-V**] **-I** *open* <*command*>

ipmitool [**-c**|**-h**|**-v**|**-V**] **-I** *lan* **-H** <*hostname*>
[**-p** <*port*>]
[**-U** <*username*>]
[**-A** <*authtype*>]

[**-L** <*privlvl*>]
[**-a**|**-E**|**-P**|**-f** <*password*>]
[**-o** <*oemtype*>]
<*command*>

ipmitool [**-c**|**-h**|**-v**|**-V**] **-I** *lanplus* **-H** <*hostname*>
[**-p** <*port*>]
[**-U** <*username*>]
[**-L** <*privlvl*>]
[**-a**|**-E**|**-P**|**-f** <*password*>]
[**-o** <*oemtype*>]
[**-C** <*ciphersuite*>]
<*command*>

DESCRIPTION

This program lets you manage Intelligent Platform Management Interface (IPMI) functions of either the local system, via a kernel device driver, or a remote system, using IPMI V1.5 and IPMI v2.0. These functions include printing FRU information, LAN configuration, sensor readings, and remote chassis power control.

IPMI management of a local system interface requires a compatible IPMI kernel driver to be installed and configured. On Linux this driver is called *OpenIPMI* and it is included in standard distributions. On Solaris this driver is called *BMC* and is included in Solaris 10. Management of a remote station requires the IPMI-over-LAN interface to be enabled and configured. Depending on the particular requirements of each system it may be possible to enable the LAN interface using *ipmitool* over the system interface.

OPTIONS

-a Prompt for the remote server password.

-A <*authtype*>
Specify an authentication type to use during IPMIv1.5 *lan* session activation. Supported types are NONE, PASSWORD, MD5, or OEM.

-c Present output in CSV (comma separated variable) format. This is not available with all commands.

-C <*ciphersuite*>
The remote server authentication, integrity, and encryption algorithms to use for IPMIv2 *lanplus* connections. See table 22-19 in the IPMIv2 specification. The default is 3 which specifies RAKP-HMAC-SHA1 authentication, HMAC-SHA1-96 integrity, and AES-CBC-128 encryption algorithms.

-E The remote server password is specified by the environment variable *IPMI_PASSWORD*.

-f <*password_file*>
Specifies a file containing the remote server password. If this option is absent, or if password_file is empty, the password will default to NULL.

-h Get basic usage help from the command line.

-H <*address*>
Remote server address, can be IP address or hostname. This option is required for *lan* and *lanplus* interfaces.

-I <*interface*>
Selects IPMI interface to use. Supported interfaces that are compiled in are visible in the usage help output.

-L <*privlvl*>
Force session privilege level. Can be CALLBACK, USER, OPERATOR, and ADMIN. Default is ADMIN.

-m <*local_address*>
Set the local IPMB address. The default is 0x20 and there should be no need to change it for normal operation.

-o <*oemtype*>
Select OEM type to support. This usually involves minor hacks in place in the code to work around quirks in various BMCs from various manufacturers. Use *-o list* to see a list of current supported OEM types.

-p <*port*>
Remote server UDP port to connect to. Default is 623.

-P <*password*>
Remote server password is specified on the command line. If supported it will be obscured in the process list.
Note! Specifying the password as a command line option is not recommended.

-t <*target_address*>

Bridge IPMI requests to the remote target address.

-U <*username*>
Remote server username, default is NULL user.

-v Increase verbose output level. This option may be specified multiple times to increase the level of debug output. If given three times you will get hexdumps of all incoming and outgoing packets.

-V Display version information.

If no password method is specified then *ipmitool* will prompt the user for a password. If no password is entered at the prompt, the remote server password will default to NULL.

SECURITY

The *ipmitool* documentation highlights that there are several security issues to be considered before enabling the IPMI LAN interface. A remote station has the ability to control a system's power state as well as being able to gather certain platform information. To reduce vulnerability it is strongly advised that the IPMI LAN interface only be enabled in 'trusted' environments where system security is not an issue or where there is a dedicated secure 'management network' or access has been provided through an *console server*.

Further it is strongly advised that you should not enable IPMI for remote access without setting a password, and that that password should not be the same as any other password on that system.

When an IPMI password is changed on a remote machine with the IPMIv1.5 *lan* interface the new password is sent across the network as clear text. This could be observed and then used to attack the remote system. It is thus recommended that IPMI password management only be done over IPMIv2.0 *lanplus* interface or the system interface on the local station.

For IPMI v1.5, the maximum password length is 16 characters. Passwords longer than 16 characters will be truncated.

For IPMI v2.0, the maximum password length is 20 characters; longer passwords are truncated.

COMMANDS

help

This can be used to get command-line help on *ipmitool* commands. It may also be placed at the end of commands to get option usage help.

ipmitool help

Commands:

raw	Send a RAW IPMI request and print response
lan	Configure LAN Channels
chassis	Get chassis status and set power state
event	Send pre-defined events to MC
mc	Management Controller status and global enables
sdr	Print Sensor Data Repository entries and readings
sensor	Print detailed sensor information
fru	Print built-in FRU and scan SDR for FRU locators
sel	Print System Event Log (SEL)
pef	Configure Platform Event Filtering (PEF)
sol	Configure IPMIv2.0 Serial-over-LAN
isol	Configure IPMIv1.5 Serial-over-LAN
user	Configure Management Controller users

channel Configure Management Controller channels
session Print session information
exec Run list of commands from file
set Set runtime variable for shell and exec

ipmitool chassis help

Chassis Commands: status, power, identify, policy, restart_cause, poh, bootdev

ipmitool chassis power help

chassis power Commands: status, on, off, cycle, reset, diag, soft

You will find more details on *ipmitools* at http://*ipmitool*.sourceforge.net/manpage.html.

15.11 Custom Development Kit (CDK)

As detailed in this manual customers can copy scripts, binaries and configuration files directly to the *console server*.

Opengear also freely provides a development kit which allows changes to be made to the software in *console server* firmware image. The customer can use the CDK to:

- generate a firmware image without certain programs, such as telnet, which may be banned by company policy
- generate an image with new programs, such as custom Nagios plug-in binaries or company specific binary utilities
- generate an image with custom defaults e.g. it may be required that the *console server* be configured to have a specific default serial port profile which is reverted to even in event of a factory reset
- place configuration files into the firmware image, which cannot then be modified e.g. # /bin/config –-set= tools update the configuration files in /etc/config which are read/write, whereas the files in /etc are read only and cannot be modified

The CDK essentially provides a snapshot of the Opengear build process (taken after the programs have been compiled and copied to a temporary directory *romfs*) just before the compressed file systems are generated. You can obtain a copy of the Opengear CDK for the particular appliance you are working with from *ftp://ftp.opengear.com/cdk* and find further information online at *http://www.opengear.com/faq284.html*

Note The CDK is free, however Opengear does not provide free technical support for systems modified using the CDK and any changes are the responsibility of the user.

15.12 Scripts for Managing Slaves

When the *console servers* are cascaded the Master is in control of the serial ports on the Slaves, and the Master's Management Console provides a consolidated view of the settings for its own and all the Slave's serial ports. However the Master does not provide a fully consolidated view e.g. *Status: Active Users* only displays those users active on the Master's ports and you will need to write a custom bash script that parses the port logs if you want to find out who's logged in to cascaded serial ports from the master.

You will probably also want to enable remote or USB logging, as local logs only buffer 8K of data and don't persist between reboots.

This script would e.g. parse each port log file line by line, each time it sees *'LOGIN: username'*, it adds username to the list of connected users for that port, each time it sees *'LOGOUT: username'* it removes it from the list. The list can then be nicely formatted and displayed. It's also possible to run this as a CGI script on the remote log server.

To enable log storage and connection logging:
- Select *Alerts & Logging: Port Log*
- *Configure* log storage
- Select *Serial & Network: Serial Port*, *Edit* the serial port(s)
- Under *Console server*, select *Logging Level 1* and click Apply

There's a useful tutorial on creating a bash script CGI at
http://www.yolinux.com/TUTORIALS/LinuxTutorialCgiShellScript.html

Similarly the Master does maintain a view of the status of the slaves:
- Select *Status: Support Report*
- Scroll down to *Processes*
- Look for: */bin/ssh -MN -o ControlPath=/var/run/cascade/%h slavename*
- These are the slaves that are connected
- Note the end of the Slaves' names will be truncated, so the first 5 characters must be unique

Alternatively, you can write a custom CGI script as described above. The currently connected Slaves can be determined by running: *ls /var/run/cascade* and the configured slaves can be displayed by running: *config -g config.cascade.slaves*

15.13 SMS Server Tools

Firmware releases V3.1 and later include the *SMS Server Tools software* which provides an SMS Gateway which can send and receive short messages through GSM modems and mobile phones.

You can send short messages by simply storing text files into a special spool directory. The program monitors this directory and sends new files automatically. It also stores received short messages into another directory as text files. Binary messages (including Unicode text) are also supported, for example ring tone messages. It's also possible to send a WAP Push message to the WAP / MMS capable mobile phone.

The program can be run as a SMS daemon which can be started automatically when the operating system starts. High availability can be ensured by using multiple GSM devices (currently up to 64, this limit is easily changeable).

The program can run other external programs or scripts after events like reception of a new message, successful sending and also when the program detects a problem. These programs can inspect the related text files and perform automatic actions

The SMS Server Tools software needs a GSM modem (or mobile phone) with SMS command set according to the European specifications GSM 07.05 (=ETSI TS 300 585) and GSM 03.38 (=ETSI TS 100 900). AT command set is supported. Devices can be connected with serial port, infrared or USB.

For more information refer *http://smstools3.kekekasvi.com* or the online Opengear *faq.html*

15.14 Multicast

By default, all Opengear console servers come with Multicasting enabled. Multicasting provides Opengear products with the ability to simultaneously transmit information from a single device to a select group of hosts.

Multicasting can be disabled and re-enabled from the command line (Firmware releases V3.1 and later). To disable multicasting type:

ifconfig eth0 -multicast

To re-enable multicasting from the command line type:

ifconfig eth0 multicast

IPv6 may need to be restarted when toggling between multicast states.

15.15 Bulk Provisioning

Opengear appliances include wizard scripts to facilitate configuration and deployment en masse. These wizards operate at the command line level, so knowledge of the Linux command line and shell scripting is useful, but not necessary – they aim to be user-friendly enough for remote hands to manage. This *bulk provisioning* feature is supported by firmware version 3.9.1 or later, and Lighthouse version 4.4.0 and later (optional).

Both the bulk provisioning of Opengear appliances and bulk enrollment of these appliances into Lighthouse central management system(s) is supported. These features may be used separately or in conjunction.

Using this method, an Opengear appliance can be fully configured and enrolled into Lighthouse with minimal interaction, in under 5 minutes. The basic steps are:

1. Configure an individual "*golden master*" appliance with the baseline configuration shared by all Opengear appliances. This may be a minimal configuration if the installs are quite diverse, or a complete configuration when dealing with replicated installs.

2. Use make-template to turn the golden master's active configuration into a template configuration that may be applied to other appliances.

3. Create an OPG backup of the templated golden master appliance.

4. Restore this configuration to each target devices via the CLI, web UI or using a USB thumb drive.

5. Login via the CLI to complete configuration using *setup-wizard*.

6. (Optional) On Lighthouse, use *enrollment-wizard* to automatically place appliances under management. This may be local/routable appliances, or remote appliances that have automatically Call Home using *callhome-wizard*.

Note: Steps 5 and 6 may be reversed for remote setup via Lighthouse

Note: Full details for the above steps can be found in the Knowledge Base

15.16 Zero Touch Provisioning

Zero Touch Provisioning (ZTP) was introduced with firmware release 3.15.1 to allow Opengear appliances to be provisioned during their initial boot from a DHCP server.

15.16.1 Preparation

These are typical steps for configuration over a trusted network:

1. Configure a same-model Opengear device.
2. Optionally use the *Bulk Provisioning* wizard scripts to remove any appliance-specific settings (i.e. create a template configuration) and/or prepare the configuration for automated Lighthouse enrollment, see the section entitled *Bulk Provisioning* in this document.
3. Save the configuration as an Opengear backup (.opg) file under *System: Configuration Backup* in the web UI, or via *config -e* in the CLI. Alternatively, you can save the XML configuration as a file ending in .xml.
4. Publish the .opg or.xml file on a fileserver that understands one of the HTTPS, HTTP, FTP or TFTP protocols.
5. Configure your DHCP server to include a "vendor specific" option for Opengear devices. The option text should be a URL to the location of the .opg or .xml file. The option text should not exceed 250 characters in length. It must end in either .opg or .xml.

6. Connect a new Opengear device (either at defaults from the factory, or config erased) to the network and apply power.
7. It may take up to 5 minutes for the device to find the .opg or .xml file via DHCP, download, install the file and reboot itself.

15.16.2 Example ISC DHCP server configuration

The following is an example of an ISC DHCP server configuration fragment for serving an .opg configuration image:

```
option space opengear  code width 1  length width 1;
option opengear.config-url code 1 = text;

class "opengear-ztp" {
  match if option vendor-class-identifier ~~ "^Opengear/";
  vendor-option-space opengear;
  option opengear.config-url "https://example.com/opg/${class}.opg";
}
```

For other DHCP servers, please consult their documentation on specifying "Vendor Specific" option fields. We use sub-option 1 to hold the URL text.

15.16.3 Setup for an untrusted LAN

If network security is a concern, and you can have remote hands insert a trusted USB flash drive into the Opengear device during provisioning, then follows are a summary of the steps required for deploying configuration in an untrusted network:

1. Generate an X.509 certificate for the client. Place it and its private key file onto a USB flash drive (concatenated as a single file, client.pem).
2. Set up a HTTPS server that restricts access to the .opg or .xml file for HTTPS onnections providing the client certificate.
3. Put a copy of the CA cert (that signed the HTTP server's certificate) onto the USB flash drive as well (ca-bundle.crt).
4. Insert the USB flash drive into the Opengear device **before attaching power or network**.
5. Continue with the steps above, but using only a https URL.
6. A detailed step-by-step document for preparing a USB flash drive and using OpenSSL to create keys is at Howto: set up a USB key for authenticated restore

15.16.4 How it works

This section explains in detail how the Opengear device uses DHCP to obtain its initial configuration.

First, an Opengear console manager is either configured or unconfigured. ZTP needs it to be in an unconfigured state, which is only obtained in the following ways:

- Firmware programming at factory
- Pressing the Config Erase button twice during operation
- Selecting *Config Erase* under *System: Administration* in the web UI, and rebooting
- Creating the file /etc/config/.init and then rebooting (command-line)

When an unconfigured Opengear boots, it performs these steps to find a configuration:

- The Opengear device transmits a DHCP DISCOVER request onto its primary Network Interface (wan). This DHCP request will carry a Vendor Class Identifier of the form Opengear/model-name (for example, Opengear/ACM5003-M) and its parameter request list will include option 43 (Vendor-Specific Information).
- On receipt of a DHCP OFFER, the device will use the information in the offer to assign an IPv4 address to its primary Network Interface, add a default route, and prepare its DNS resolver.
- If the offer also contained an option 43 with sub-option 1, the device interprets the sub-option as a whitespace-separated list of URLs to configuration files to try to restore.
- If an NTP server option was provided in the DHCP offer, the system clock is (quickly) synchronized with the NTP server.
- The system now searches all attached USB storage devices for two optional certificate files. The first file is named ca-bundle.crt and the second one is whichever one of the following filenames is found first:
    - client-AABBCCDDEEFF.pem (where AABBCCDDEEFF is the MAC address of the primary network interace); or
    - client-MODEL.pem (where MODEL is the (vendor class) model name in lowercase, truncated to before the first hyphen); or
    - client.pem
- If both files are found (ca-bundle.crt and a client.pem), then secure mode is enabled for the next section.
- Each URL in the list obtained from option 43 sub-option 1 is tried in sequence until one succeeds:
    - The URL undergoes substring replacement from the following table:

Substring	Replaced by	Example
${mac}	the 12-digit MAC address of the device, lowercase	0013b600b669
${model}	the full model name, in lowercase	acm5504-5-g-w-i
${class}	the firmware hardware class	ACM550x
${version}	the firmware version number	3.15.1

    - The resulting URL must end in .opg or .xml (an optional ?query-string is permitted). It is doesn't, then it is skipped and the next URL is tried.
    - In secure mode, the URL must use the https scheme or it is skipped.
    - Otherwise the available schemes are: http https tftp ftp ftps
    - The curl program is used to download the URL.
    - In secure mode, the server's certificate must validate against the ca-bundle.crt. The (reqiured) client.pem file is provided to authenticate the client to the server. Please see the curl documentation for the format of these files.
- The URL is downloaded. For .opg files its header is checked to see if it is compatible with the current device. For .xml files, a parse check is made. If the check fails, the downloaded file is abandoned and the next URL is tried.
- The file is imported into the current configuration.
- The system checks to see if a hostname has been set in the config. If not, it is set to ${model}-${mac}.
- The system checks to see if it is still in an unconfigured state. If it is, then the network interface mode is set to DHCP. This effectively forces the system into a configured state, preventing a future reboot loop.
- The system reboots

Note: If all the URLs were skipped or failed, the system will wait for 30 seconds before retrying again. It will retry all the

URLs up to 10 times. After the 10th retry, the system reboots. If the system has been manually configured in the meantime, the retries stop and ZTP is disabled.

Note: If no option 43 is received over DHCP, no URLs are downloaded and no reboots occur: the system must be manually configured. Once configured (manually or by ZTP), an Opengear will no longer request option 43 from the DHCP server, and it will ignore any option 43 configuration URLs presented to it.

APPENDIX A: Linux Commands & Source Code

The *console server* platform is a dedicated Linux computer, optimized to provide monitoring and secure access to serial and network consoles of critical server systems and their supporting power and networking infrastructure.

Opengear *console servers* are built on the 2.6 uCLinux kernel as developed by the uCLinux project (except for SD4001/4002 which have less flash and use 2.4 uCLinux kernel). This is GPL code and source can be found at *http://cvs.uclinux.org*.

Some uCLinux commands have config files that can be altered (e.g. *portmanager, inetd*, *init*, *ssh/sshd/scp/sshkeygen, ucd-snmpd, samba, fnord, sslwrap).*

Other commands you can run and do neat stuff with (e.g. *loopback*, *bash (shell)*, *ftp, hwclock*, *iproute*, *iptables*, *netcat*, *ifconfig*, *mii-tool*, *netstat*, *route*, *ping*, *portmap*, *pppd*, *routed*, *setserial*, *smtpclient*, *stty*, *stunel*, *tcpdump*, *tftp*, *tip*, *traceroute)*

Below are most of the standard uCLinux and Busybox commands (and some custom Opengear commands) that are in the default build tree. The *Administrator* can use these to configure the *console server*, and monitor and manage attached serial console and host devices:

addgroup *	Add a group or add an user to a group
adduser *	Add an user
agetty	alternative Linux getty
arp	Manipulate the system ARP cache
arping	Send ARP requests/replies
bash	GNU Bourne-Again Shell
busybox	Swiss army knife of embedded Linux commands
cat *	Concatenate FILE(s) and print them to stdout
chat	Useful for interacting with a modem connected to stdin/stdout
chgrp *	Change file access permissions
chmod *	Change file access permissions
chown *	Change file owner and group
config	Opengear tool to manipulate and query the system configuration from the command line
cp *	Copy files and directories
date *	Print or set the system date and time
dd *	Convert and copy a file
deluser *	Delete USER from the system
df *	Report file system disk space usage
dhcpd	Dynamic Host Configuration Protocol server
discard	Network utility that listens on the discard port
dmesg *	Print or control the kernel ring buffer
echo *	Print the specified ARGs to stdout
erase	Tool for erasing MTD partitions
eraseall	Tool for erasing entire MTD partitions
false *	Do nothing, unsuccessful
find	Search for files
flashw	Write data to individual flash devices
flatfsd	Daemon to save RAM file systems back to FLASH
ftp	Internet file transfer program
gen-keys	SSH key generation program

getopt *	Parses command options
gettyd	Getty daemon
grep *	Print lines matching a pattern
gunzip *	Compress or expand files
gzip *	Compress or expand files
hd	ASCII, decimal, hexadecimal, octal dump
hostname *	Get or set hostname or DNS domain name
httpd	Listen for incoming HTTP requests
hwclock	Query and set hardware clock (RTC)
inetd	Network super-server daemon
inetd-echo	Network echo utility
init	Process control initialization
ip	Show or manipulate routing, devices, policy routing and tunnels
ipmitool	Linux IPMI manager
iptables	Administration tool for IPv4 packet filtering and NAT
ip6tables	Administration tool for IPv6 packet filtering
iptables-restore	Restore IP Tables
iptables-save	Save IP Tables
kill *	Send a signal to a process to end gracefully
ln *	Make links between files
login	Begin session on the system
loopback	Opengear loopback diagnostic command
loopback1	Opengear loopback diagnostic command
loopback2	Opengear loopback diagnostic command
loopback8	Opengear loopback diagnostic command
loopback16	Opengear loopback diagnostic command
loopback48	Opengear loopback diagnostic command
ls *	List directory contents
mail	Send and receive mail
mkdir *	Make directories
mkfs.jffs2	Create an MS-DOS file system under Linux
mknod *	Make block or character special files
more *	File perusal filter for crt viewing
mount *	Mount a file system
msmtp	SMTP mail client
mv *	Move (rename) files
nc	TCP/IP Swiss army knife
netflash	Upgrade firmware on ucLinux platforms using the blkmem interface
netstat	Print network connections, routing tables, interface statistics etc
ntpd	Network Time Protocol (NTP) daemon
pgrep	Display process(es) selected by regex pattern
pidof	Find the process ID of a running program
ping	Send ICMP ECHO_REQUEST packets to network hosts
ping6	IPv6 ping
pkill	Sends a signal to process(es) selected by regex pattern
pmchat	Opengear command similar to the standard chat command (via portmanager)
pmdeny	
pminetd	

pmloggerd	
pmshell	Opengear command similar to the standard *tip* or *cu* but all serial port access is directed via the portmanager.
pmusers	Opengear command to query portmanager for active user sessions
portmanager	Opengear command that handles all serial port access
portmap	DARPA port to RPC program number mapper
pppd	Point-to-Point protocol daemon
ps *	Report a snapshot of the current processes
pwd *	Print name of current/working directory
reboot *	*Soft* reboot
rm *	Remove files or directories
rmdir *	Remove empty directories
routed	Show or manipulate the IP routing table
routed	Show or manipulate the IP routing table
routef	IP Route tool to flush IPv4 routes
routel	IP Route tool to list routes
rtacct	Applet printing /proc/net/rt_acct
rtmon	RTnetlink listener
scp	Secure copy (remote file copy program)
sed *	Text stream editor
setmac	Sets the MAC address
setserial	Sets and reports serial port configuration
sh	Shell
showmac	Shows MAC address
sleep *	Delay for a specified amount of time
smbmnt	Helper utility for mounting SMB file systems
smbmount	Mount an SMBFS file system
smbumount	SMBFS umount for normal users
snmpd	SNMP daemon
snmptrap	Sends an SNMP notification to a manager
sredird	RFC 2217 compliant serial port redirector
ssh	OpenSSH SSH client (remote login program)
ssh-keygen	Authentication key generation, management, and conversion
sshd	OpenSSH SSH daemon
sslwrap	Program that allows plain services to be accessed via SSL
stty	Change and print terminal line settings
stunnel	Universal SSL tunnel
sync *	Flush file system buffers
sysctl	Configure kernel parameters at runtime
syslogd	System logging utility
tar *	The tar archiving utility
tc	Show traffic control settings
tcpdump	Dump traffic on a network
telnetd	Telnet protocol server
tftp	Client to transfer a file from/to tftp server
tftpd	Trivial file Transfer Protocol (tftp) server
tip	Simple terminal emulator/cu program for connecting to modems and serial devices
top	Provide a view of process activity in real time
touch *	Change file timestamps

traceroute	Print the route packets take to network host
traceroute6	Traceroute for IPv6
true *	Returns an exit code of TRUE (0)
umount *	Unmounts file systems
uname *	Print system information
usleep *	Delay for a specified amount of time
vconfig *	Create and remove virtual Ethernet devices
vi *	Busybox clone of the VI text editor
w	Show who is logged on and what they are doing
zcat *	Identical to gunzip -c

Commands above which are appended with '*' come from Busybox (the Swiss Army Knife of embedded Linux*) http://www.busybox.net/downloads/BusyBox.html*.

Others are generic Linux commands and most commands the ***-h*** or ***--help*** argument to provide a terse runtime description of their behavior. More details on the generic Linux commands can found online at *http://en.tldp.org/HOWTO/HOWTO-INDEX/howtos.html* and *http://www.faqs.org/docs/Linux-HOWTO/Remote-Serial-Console-HOWTO.html*

An updated list of the commands in the latest *console server* build can be found at *http://www.opengear.com/faq233.html*. However it may be worth using ***ls*** command to view all the commands actually available in the */bin* directory in your *console server*.

There were a number of Opengear tools listed above that make it simple to configure the *console server* and ensure the changes are stored in the *console server*'s flash memory etc. These commands are covered in the previous chapters and include:

- **config** which allows manipulation and querying of the system configuration from the command line. With *config* a new configuration can be activated by running the relevant configurator, which performs the action necessary to make the configuration changes live
- **portmanager** which provides a buffered interface to each serial port. It is supported by the pmchat and pmshell commands which ensure all serial port access is directed via the *portmanager*
- **pmpower** is a configurable tool for manipulating remote power devices that are serially or network connected to the *console server*
- **SDT Connector** is a java client applet that provides point-and-click SSH tunneled connections to the *console server* and Managed Devices

There are also a number of other CLI commands related to other open source tools embedded in the *console server* including:

- **PowerMan** provides power management for many preconfigured remote power controller (RPC) devices. For CLI details refer *http://linux.die.net/man/1/powerman*
- **Network UPS Tools (NUT)** provides reliable monitoring of UPS and PDU hardware and ensure safe shutdowns of the systems which are connected - with a goal to monitor every kind of UPS and PDU. For CLI details refer *http://www.networkupstools.org*
- **Nagios** is a popular enterprise-class management tool that provides central monitoring of the hosts and services in distributed networks. For CLI details refer http://www.nagios.org

Many components of the *console server* software are licensed under the GNU General Public License (version 2), which Opengear supports. You may obtain a copy of the GNU General Public License at ***http://www.fsf.org/copyleft/gpl.html.*** Opengear will provide source code for any of the components of the software licensed under the GNU General Public License upon request.

Note: The software included in each Opengear console server contains copyrighted software that is licensed under the GPL (refer Appendix F for a copy of the GPL license). You may obtain the latest snapshot source code package on a CD by sending a money order or check for $5 to:
Opengear Support
630 West 9560 South, Suite A
Sandy, UT 84070, USA

Alternately the complete source code corresponding to each released version is available from us for a period of three years after its last shipment. If you would like the source code for an earlier release than the latest current release please write “source for firmware Version x.xx ” in the memo line of your payment.

This offer is valid to anyone in receipt of this information.

The *console server* also embodies the *okvm* console management software. This is GPL code and the full source is available from *http://okvm.sourceforge.net.*

The *console server* BIOS (boot loader code) is a port of *uboot* which is also a GPL package with source openly available.

The *console server* CGIs (the html code, xml code and web config tools for the Management Console) are proprietary to Opengear, however the code will be provided to customers, under NDA.

Also inbuilt in the *console server* is a Port Manager application and Configuration tools as described in *Chapters 14* and *15*. These both are proprietary to Opengear, but open to customers (as above).

The *console server* also supports GNU *bash* shell script enabling the *Administrator* to run custom scripts. GNU *bash*, version 2.05.0(1)-release (arm-OpenGear-linux-gnu) offers the following shell commands:

```
alias [-p] [name[=value] ... ]
bg [job_spec]
bind [-lpvsPVS] [-m keymap] [-f fi
break [n]
builtin [shell-builtin [arg ...]]
case WORD in [PATTERN [|
PATTERN]
cd [-PL] [dir]
command [-pVv]
command [arg ...]
compgen [-abcdefjkvu] [-o option]
complete [-abcdefjkvu] [-pr] [-o o]
continue [n]
declare [-afFrxi] [-p] name[=value]
dirs [-clpv] [+N] [-N]
disown [-h] [-ar] [jobspec ...]
echo [-neE] [arg ...]
enable [-pnds] [-a] [-f filename]
eval [arg ...]
exec [-cl] [-a name] file [redirec]
exit [n]
export [-nf] [name ...] or export
false
fc [-e ename] [-nlr] [first] [last]
fg [job_spec]
for NAME [in WORDS ... ;] do
local name[=value] ...
logout
popd [+N | -N] [-n]
printf format [arguments]
pushd [dir | +N | -N] [-n]
pwd [-PL]
read [-ers] [-t timeout] [-p promp]
readonly [-anf] [name ...] or read return
[n]
select NAME [in WORDS ... ;] do
COMMANDS
set [--abefhkmnptuvxBCHP] [-o opti]
shift [n]
shopt [-pqsu] [-o long-option] opt
source filename
suspend [-f]
test [expr]
time [-p] PIPELINE
times
trap [arg] [signal_spec ...]
true
type [-apt] name [name ...]
typeset [-afFrxi] [-p] name[=value
ulimit [-SHacdflmnpstuv] [limit]
umask [-p] [-S] [mode]
unalias [-a] [name ...]
```

COMMA
function NAME { COMMANDS ; } or NA
getopts optstring name [arg]
hash [-r] [-p pathname] [name ...]
help [-s] [pattern ...]
history [-c] [-d offset] [n] or hi
if COMMANDS; then COMMANDS; [
elif jobs [-lnprs] [jobspec ...] or *job kill [-s sigspec | -n signum | -si let arg [arg ...]*

unset [-f] [-v] [name ...]
until COMMANDS; do COMMANDS; done
variables - Some variable names an
wait [n]
while COMMANDS; do COMMANDS; done { COMMANDS ; }

APPENDIX B: Hardware Specification

FEATURE	VALUE
Dimensions	ACM5002/3/4(-2) (-M/W/G): 4.1x3.4x1.1 in (10.3 x 8.7 x 2.8 cm) ACM5504/8-2/5(-M/G/W/I): 6.5 x 4 x 1.4 in (16.6 x 10.2 x 2.8 cm) ACM7004-2-L(V/A/R): 5 1/8 x 4 3/4 x 1 3/8 in (13 x 12 x 3.5 cm) IM7216/32/48: 17 x 10 x 1.75 in (44 x 25.4 x 4.5 cm) IM4208/16/32/48: 17 x 12 x 1.75 in (43.2 x 31.3. x 4.5 cm) IM4216-34: 17 x 12 x 1.75 in (43.2 x 31.3. x 4.5 cm) CM7116/32/48: 17 x 6.9 x 1.75 in (44 x 17 x 4.5 cm) SD4002: 3.9 x 2.8 x 1.0 in (10 x 7.2 x 2.5 cm) SD4001: 4 x 1.75 x 1.0 in (10.2 x 4.4 x 2.5 cm)
Weight	ACM5002/3/4(M/W/G)(-2): 1.0 kg (2.2 lbs) ACM5504/8-2/5(-M/G/W/I): 1.8 kg (4 lbs) ACM7004-2-L(V/A/R): 0.6kg (1.3 lbs) IM7216/32/48: 4.5 kg (10 lbs) IM4208/16/32/48: 5.4 kg (11.8 lbs) IM4216-34: 5.4 kg (11.8 lbs) CM7116/32/48: 4 kg (9 lbs) SD4002: 1.1 kg (2.5 lbs)
Ambient operating temperature	5°C to 50°C (41°F to 122°F) Higher for -I models
Non-operating storage temp	-30°C to +60°C (-20°F to +140°F)
Humidity	5% to 90%
Power	Refer Chapter 2 for various models
Power Consumption	All less than 30W
CPU	IM7200: 1GHz ARM SoC (Marvell 88F6283) CM7100: 800MHz ARM SoC (Marvell 88F6W11) ACM7000: 800MHz ARM SoC (Marvell 88F6W11) ACM5000 & ACM5500: Micrel KSZ8692 ARM9 Others: Micrel KS8695P controller
Memory	ACM5002/3/4(M/W/G)(-2): 32MB SDRAM, 16MB Flash ACM5504/8-2/5(-M/G/W/I): 64MB SDRAM, 16MB + 4GB Flash ACM7004-2-L(V/A/R): 254MB SDRAM, 256MB + 4GB Flash IM7216/32/48: 256MB SDRAM, 64MB + 16 GB Flash IM4208/16/32/48: 64MB SDRAM, 16MB + 16GB Flash IM4216-34: 64MB SDRAM, 16MB Flash 16GB Flash CM7116/32/48: 256MB SDRAM, 32MB + 4GB Flash SD4001/2: 16MB SDRAM, 8MB Flash
USB ports	ACM5002/3/4(M/W/G)(-2): 2 internal/external USB2.0 ACM5504/8-2/5(-M/G/W/I): 2 external USB2.0 ACM7004-2-L(V/A/R): 4 external USB2.0 IM7216/32/48: 2 external USB3.0 IM4208/16/32/48 & IM4216-34: 3 external USB2.0 CM7116/32/48: 2 external USB2.0
Serial Connectors	ACM5002: 2 RJ-45 RS-232 serial ports ACM5003-M/W: 3 RJ-45 RS-232 serial ports ACM5004(G)(-2): 4 RJ-45 RS-232 serial ports ACM5004-2(G)-I: 4 RJ-45 selectable RS-232/422/485 serial ports ACM5504-(2/5)-G(-W)-I(-P): 4 RJ-45 RS-232 serial ports ACM5508-2-I/M: 8 RJ-45 selectable RS-232/422/485 serial ports

	ACM7004-2-L(V/A/R): 4 RJ-45 RS-232 serial ports IM7216-2: 16 RJ-45 RS-232 serial ports ** IM7232-2: 32 RJ-45 RS-232 serial ports ** IM7248-2: 48 RJ-45 RS-232 serial ports** IM4208-2: 8 RJ-45 RS-232 serial ports * IM4216-2 & IM4216-34: 16 RJ-45 RS-232 serial ports * IM4232-2: 32 RJ-45 RS-232 serial ports * IM4248-2: 48 RJ-45 RS-232 serial ports * CM7116-2: 16 RJ-45 RS-232 serial ports ** CM7132-2: 32 RJ-45 RS-232 serial ports ** CM7148-2: 48 RJ-45 RS-232 serial ports ** SD4002: 2 DB-9 RS-232 serial port (one selectable RS-232/422/485) SD4001: 1 DB-9 selectable RS-232/422/485 serial port * models also have 1 DB-9 RS-232 console / modem serial port ** models also have 1 RJ45 console port
Serial Baud Rates	RJ45 ports - 50 to 230,400bps DB9 port - 2400 to 115,200 bps
Ethernet Connectors	ACM5002/3/4(M/W/G): One RJ-45 10/100Base-T Ethernet ports ACM5004-2, ACM5504-2 and ACM5508-2: Two RJ-45 10/100Base-T Ethernet ports IM7216/32/48: Two 10/100/1000 GbE copper or SFP fiber ports IM4208/16/32/48-2: Two RJ-45 10/100Base-T Ethernet ports IM4216-34: Two RJ-45 10/100Base-T Ethernet ports and 32x RJ-45 10/100Base-T management LAN switched ports ACM5504-5-G-W-I & ACM5004-5-G-I: One RJ-45 10/100Base-T primary Ethernet port and 4x RJ-45 10/100Base-T management LAN switched ports ACM7004-2-L(V/A/R): Two 10/100/1000 GbE ports CM7116/32/48: Two 10/100/1000 GbE ports IM7216/32/48: Two 10/100/1000 GbE ports

APPENDIX C: Safety & Certifications

Please take care to follow the safety precautions below when installing and operating the *console server*:

- Do not remove the metal covers. There are no operator serviceable components inside. Opening or removing the cover may expose you to dangerous voltage which may cause fire or electric shock. Refer all service to Opengear qualified personnel

- To avoid electric shock the power cord protective grounding conductor must be connected through to ground.

- Always pull on the plug, not the cable, when disconnecting the power cord from the socket.

Do not connect or disconnect the *console server* during an electrical storm. Also it is recommended you use a surge suppressor or UPS to protect the equipment from transients.

FCC Warning Statement

This device complies with Part 15 of the FCC rules. Operation of this device is subject to the following conditions: (1) This device may not cause harmful interference, and (2) this device must accept any interference that may cause undesired operation.

WEEE Statement

The symbol on the product or its packaging indicates that this product must not be disposed of with your other household waste. Instead, it is your responsibility to dispose of your waste equipment by handing it over to a designated collection point for the recycling of waste electrical and electronic equipment. The separate collection and recycling of your waste equipment at the time of disposal will help conserve natural resources and ensure that it is recycled in a manner that protects human health and the environment. For more information about where you can drop off your waste for recycling, please contact your local authority, or where you purchased your product.

APPENDIX D: Connectivity, TCP Ports & Serial I/O

Pin-out standards exist for both DB9 and DB25 connectors; however there are not pin-out standards for serial connectivity using RJ45 connectors. Most *console servers* and serially managed servers / router / switches / power devices have adopted their own unique pin-out; so custom connectors and cables may be required to interconnect your *console server.*

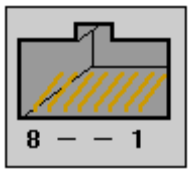
Serial Port Pinout

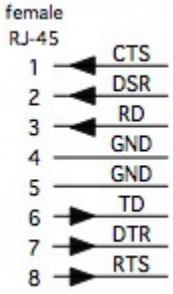
Opengear's *console servers* come with one to forty eight serial connectors (notated *SERIAL* or *SERIAL PORTS*) for the RS232 serial ports:

- The SD4001 and SD4002 models have DB9 serial port connectors, all other models have RJ45 serial port connectors
- The RJ45 serial ports are located on the front face of the ACM5000 and ACM5500; on the front panel of the rack mount IM4200; and on the rear panel of the rack mount IM7200 and CM7100
- The CM7100, ACM5000, ACM5500 and ACM7000 models and the IM4216-34 have Cisco Straight serial pinouts on its RJ45 connectors
- The other IM4200 *console servers* are available with a selection of alternate RJ45 pinouts (which must be specified in the part number at the time of order). The IM4208-2, IM4216-2, IM4232-2 and IM4248-2 *console servers* have three RJ45 pinout configurations available – Opengear Classic, Cisco Straight or Cyclades/Cisco Rolled
- The IM7200 has software selectable Cisco Straight or Cisco Rolled RJ45

Cisco Straight RJ45 pinout (option -X2)

Straight through RJ-45 cable to equipment such as Cisco, Juniper, SUN, and more...

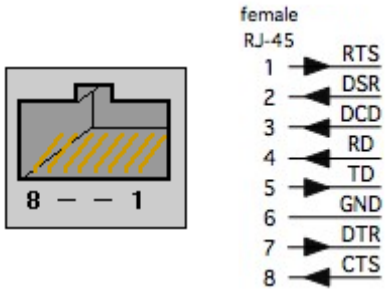




PIN	SIGNAL	DEFINITION	DIRECTION
1	CTS	Clear To Send	Input
2	DSR	Data Set Ready	Input
3	RXD	Receive Data	Input
4	GND	Signal Ground	NA
5	GND	Signal Ground	NA
6	TXD	Transmit Data	Output
7	DTR	Data Terminal Ready	Output
8	RTS	Request To Send	Output

Opengear Classic (X0) RJ45 pinout

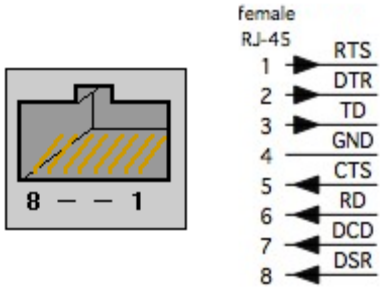
This is the same RJ45 pinout as the Avocent /Equinox brand *console server:*



PIN	SIGNAL	DEFINITION	DIRECTION
1	RTS	Request To Send	Output
2	DSR	Data Set Ready	Input
3	DCD	Data Carrier Detect	Input
4	RXD	Receive Data	Input
5	TXD	Transmit DataCTS	Output
6	GND	Signal Ground	NA
7	DTR	Data Terminal Ready	Output
8	CTS	Clear To Send	Input

Cisco Rolled RJ45 pinout (option -X1)

Easy to replace Avocent/Cyclades products, for use with rolled RJ-45 cable:



PIN	SIGNAL	DEFINITION	DIRECTION
1	RTS	Request To Send	Output
2	DTR	Data Terminal Ready	Output
3	TXD	Transmit Data	Output
4	GND	Signal Ground	NA
5	CTS	Clear To Send	Input
6	RXD	Receive Data	Input
7	DCD	Data Carrier Detect	Input
8	DSR	Data Set Ready	Input

Local Console Port

Console servers with a dedicated LOCAL console/modem port use a standard DB9 connector for this port.

To connect to the LOCAL modem/console port on the *console servers* using a computer or terminal device use the 319001 or 319003 adaptors with standard UTP Cat 5 cable.

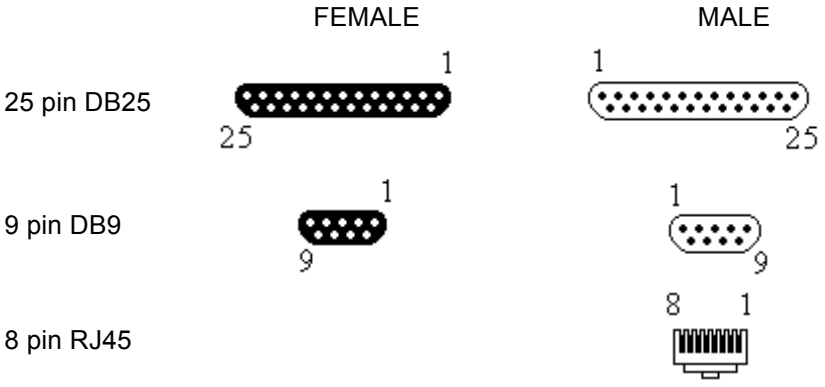
To connect the LOCAL console ports to modems (for out of band access) use the 319004 adaptor with standard UTP Cat 5 cable.

Each Opengear *console server* is supplied with UTP Cat 5 cables.

RS232 Standard Pinouts

The RS232 pinout standards for the DB9 (and DB25) connectors are tabled below:

DB25	SIGNAL	DB9	DEFINITION
1			Protective Ground
2	TXD	3	Transmitted Data
3	RXD	2	Received Data
4	RTS	7	Request To Send
5	CTS	8	Clear To Send
6	DSR	6	Data Set Ready
7	GND	5	Signal Ground
8	CD	1	Received Line Signal Detector
9			Reserved for data set testing
10			Reserved for data set testing
11			Unassigned
12	SCF		Secondary Rcvd Line Signal Detector
13	SCB		Secondary Clear to Send
14	SBA		Secondary Transmitted Data
15	DB		Transmission Signal Timing
16	SBB		Secondary Received Data
17	DD		Receiver Signal Element Timing
18			Unassigned
19	SCA		Secondary Request to Send
20	DTR	4	Data Terminal Ready
21	CG		Signal Quality Detector
22		9	Ring Indicator
23	CH/CI		Data Signal Rate Selector
24	DA		Transmit Signal Element Timing
25			Unassigned



Connectors included in *console server*

The ACM5000, ACM5500, ACM7000, CM7100 and IM7200 families, and the IM4208/16/32/48-X2 and IM4216-34-X2, have the Cisco pinout by default and ship with “cross-over” / “straight” RJ45-DB9 connectors:

Part # 319014

DB9F-RJ45S straight connector

WIRING TABLE

RJ-45		DB9 F
1 CTS	----------	8 CTS
2 DCD	----------	1 DCD
3 RXD	----------	2 RXD
4 N/C		
5 GND	----------	5 GND
6 TXD	----------	3 TXD
7 DTR	----------	4 DTR
8 RTS	----------	7 RTS

Part # 319015

DB9F-RJ45S cross-over connector

WIRING TABLE

RJ-45		DB9 F
1 CTS	----------	7 RTS
2 DCD	----------	4 DTR
3 RXD	----------	3 TXD
4 N/C		
5 GND	----------	5 GND
6 TXD	----------	2 RXD
7 DTR	----------|----	1 DCD
	|----	6 DSR
8 RTS	----------	8 CTS

The IM4208/16/32/48-X0 all have the Opengear Classic pinout and ship with a “cross-over” and a “straight” RJ45-DB9 connector for connecting to other vendor’s products:

Part # 319000

DB9F-RJ45S straight connector

WIRING TABLE

	DB9F		RJ45	
RTS	7	————	1	RTS
DSR	6	————	2	DSR
DCD	1	————	3	DCD
RXD	2	————	4	RXD
TXD	3	————	5	TXD
GND	5	————	6	GND
DTR	4	————	7	DTR
CTS	8	————	8	CTS
RI	9			

Part # 319001

DB9F-RJ45S cross-over connector

WIRING TABLE

	DB9F		RJ45	
CTS	8	——	1	RTS
DTR	4	——	2	DSR
DTR	4	——	3	DCD
TXD	3	——	4	RXD
RXD	2	——	5	TXD
GND	5	——	6	GND
DSR	6	——	7	DTR
DCD	1	——	7	DTR
RTS	7	——	8	CTS
RI	9			

Other available connectors and adapters

Opengear also supplies a range of cables and adapters that will enable you to easily connect to the more popular servers and network appliances. More detailed information can be found online at http://www.opengear.com/cabling.html

For Local/Console connection:

These adapters connect the *console server* LOCAL/Console port (via standard UTP Cat 5 cable) to modem devices (for out-of-band access):

319000 DB9F to RJ45 straight console server LOCAL Console Port to Modem

319002 DB25M to RJ45 straight console server LOCAL Console Port to Modem

For *console server* Serial Port connection, the Opengear connectors and adapters detailed below are specified to work with standard UTP Cat 5 cable.

For console servers with Cisco pinouts:

319014 DB9F to RJ45 straight Console server with Cisco pinout to IP Power and other serial device

319015 DB9F to RJ45 crossover DCE Adapter - Console server with Cisco pinout to X86 and other

319016 DB9M to RJ45 straight DTE Adapter - Console server with Cisco pinout to Netscreen and Dell

319004 DB9M to RJ45 straight DTE Adapter - Console server OOB modem connection

For console servers with Opengear Classic pinouts:

319000 DB9F to RJ45 straight Console server with Opengear classic pinout to IP Power and other serial device

319001 DB9F to RJ45 crossover DCE Adapter - Console server with Opengear classic pinout to X86 and other

319002 DB25M to RJ45 straight DTE Adapter for console server with Opengear classic pinout

319003 DB25M to RJ45 crossover DCE Adapter - Console server with Opengear classic pinout to Sun and other

319004 DB9M to RJ45 straight DTE Adapter - Console server with Opengear classic pinout to Netscreen and Dell; and OOB modem connection

319005 DB25F to RJ45 crossover DCE Adapter - Console server with Opengear classic pinout to Cisco 7200 AUX

440016 5ft Cat5 RJ-45 to RJ-45 cables

Extension cables

449016 RJ-45 plug to RJ-45 jack Adapter for console server with Opengear classic pinout to Cisco console (and to Netscreen with reversing cable)

449017 RJ-45 plug to RJ-45 jack Adapter for console server with Opengear classic pinout to Rackable Systems console

TCP/UDP Port Numbers

Port numbers are divided into three ranges: *Well Known Ports, Registered Ports* and *Dynamic* and/or *Private Ports*. Well Known Ports are those from 0 through 1023. Registered Ports are those from 1024 through 49151. Dynamic and/or Private Ports are those from 49152 through 65535.

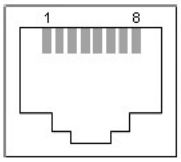
Well Known Ports are assigned by IANA, and on most systems, can only be used by system processes or by programs executed by privileged users. Table below shows some of the well-known port numbers. For more details, please visit the IANA website: http://www.iana.org/assignments/port-numbers

Port Number	Protocol	TCP/UDP
21	FTP (File Transfer Protocol)	TCP
22	SSH (Secure Shell)	TCP
23	Telnet	TCP
25	SMTP (Simple Mail Transfer Protocol)	TCP
37	Time	TCP, UCP
39	RLP (Resource Location Protocol)	UDP
49	TACACS, TACACS+	UDP
53	DNS	UDP
67	BOOTP server	UDP
68	BOOTP client	UDP
v69	TFTP	UDP
70	Gopher	TCP
79	Finger	TCP
80	HTTP	TCP
110	POP3	TCP
119	NNTP (Network News Transfer Protocol)	TCP
161/162	SNMP	UDP
443	HTTPS	TCP

Serial Port Pinouts – ACM5004-2-I, ACM5504-5-G-I and ACM5508-2-I

Each serial RJ-45 ports on these models can be software selected to be RS-232, RS-422 or RS-485.

- For RS232 they have the Cisco pinout
- For RS-422 mode it's 4-wire full duplex transmit on TX+/TX- pair, receive on RX+/RX- pair with the following pinout



Pin	Signal	Direction	RS422 Signal Description
1	RX+	Input	Receive Data
2	N/C		Receive Data
3	RX-	Input	
4	GND		
5	GND		
6	TX+	Output	Transmit Data
7	N/C		
8	TX-	Output	Transmit Data

- For RS-485 it's 2-wire half duplex

  For the RS-485 option, to provide half duplex 'party-line' communications over a 2-wire bus (D+/D-), two short cable loops are required between the RX+/TX+ pins (pins 1 and 6) and RX-/TX- pins (pins 3 and 8) on the serial RJ-45 cable connector. This is because the -I model uses universal differential transceivers that support 4-wire (RS-422) and 2-wire (RS-485) operation. In RS-485 mode, the -I model listens on the 2-wire bus for receive data until it is required to send data. In RS-485 send mode it stops receiving, enables its transmitters when there is data to be sent, transmits the data and returns to receive mode. This eliminates the possibility of collisions with other devices which share the RS-485 bus and avoids receiving bogus stale echoed data.

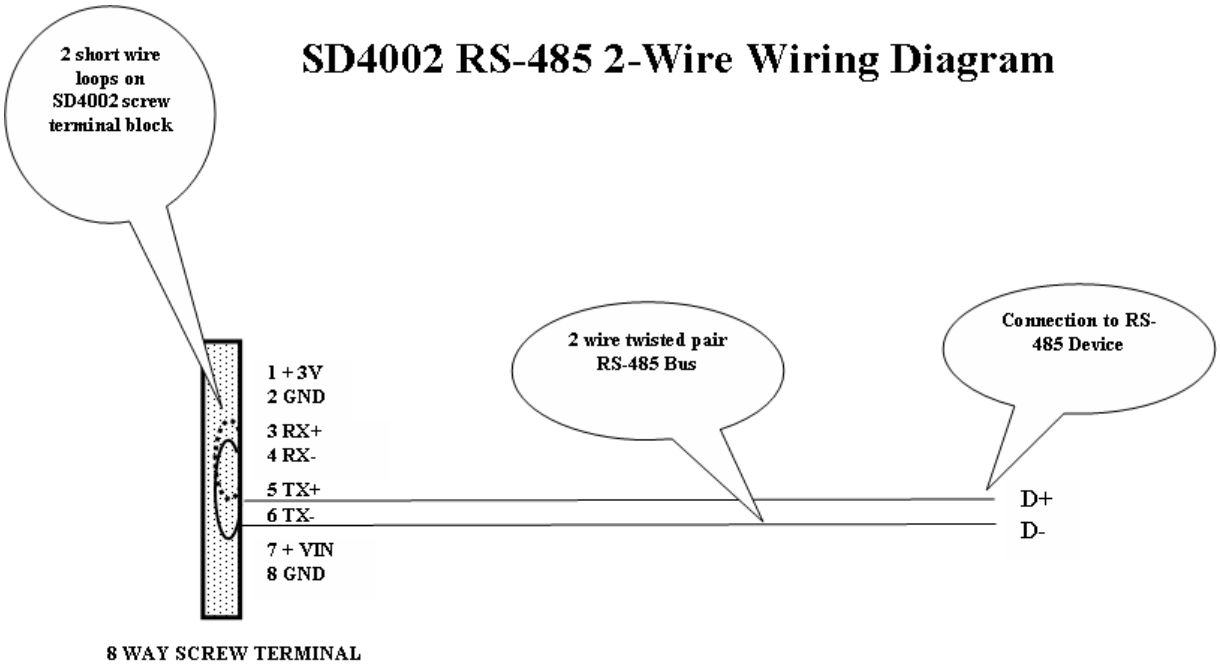
Serial Port Pinouts – SD4002

The SD4002 supports (by default) two RS232 ports on Port 1 and Port 2 DB9 connectors. Port 2 on the SD4002 can also be software selected to be an RS485 or RS422 port connected through the screw terminal block (shown below):

1	+V DC IN
2	GND
3	RX+
4	RX-
5	TX+
6	TX-
7	+3.3V DC OUT
8	GND

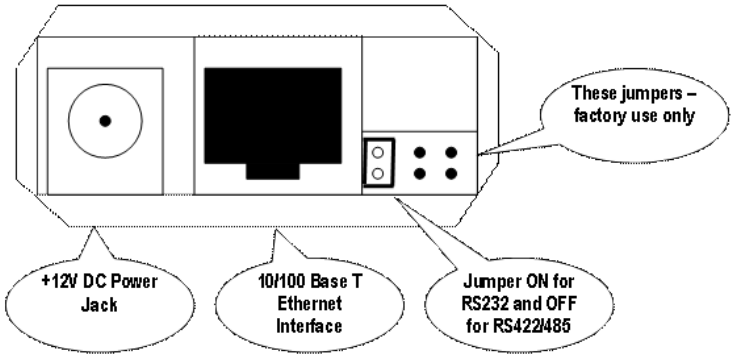
- RS-422 uses a full duplex transmit on TX+/TX- pair, receive on RX+/RX- pair

- RS-485 uses half duplex over single pair. The SD4002 supports half duplex 'party-line' communications over a 2-wire RS-485 bus (D+/D-). This is enabled by choosing the RS-485 option (instead of RS-232 or RS-422) for "Signaling Protocol" from the "Serial Port: Configuration" link on the Web management console. In addition two short cable loops are required between the RX+/TX+ pins and RX-/TX- pins. This is because the SD4002 uses universal differential transceivers that support 4-wire (RS-422) and 2-wire (RS-485) operation. In RS-485 mode, Port2 on the SD4002 listens on the 2-wire bus for receive data until it is required to send data. In RS-485 *send mode* it stops receiving, enables its transmitters when there is data to be sent, transmits the data and returns to receive mode. This eliminates the possibility of collisions with other devices which share the RS-485 bus and avoids receiving stale echoed data.



Serial Port Pinouts –SD4001

The SD4001 has one DB9 serial port that can selected to be an RS232, RS485 or RS422 port. By default the SD4001 is configured in RS232 mode (with a vertical jumper in place on the left hand **SEL** pins).



To set the port in RS422 or RS485 mode you must remove the SEL jumper and then configure the *Signaling Protocol* using the Management Console.

The DB9 pin-out is:

Pin:	RS232	RS422	RS485
1	DCD	DCD+	-
2	RXD	RX -	-
3	TXD	TX +	D+
4	DTR	DTR+	-
5	GND	GND	GND
6	DSR	RX +	-
7	RTS	TX -	D-
8	CTS	DCD-	-
9	-	DTR-	-

RS-422 uses a full duplex transmit on TX+ (Transmit Data +) / TX- (Transmit Data -) pair, receive on RX+ (Receive Data +) / RX- (Receive Data –) pair.

RS-485 uses half duplex over single pair. For RS-485 which is a 2-wire bus that drives D+ and D- from a native 4-wire interface you need to loop 3-6 and 2-7 on the DB-9.

APPENDIX E: TERMINOLOGY

TERM	MEANING
3G	Third-generation cellular technology. The standards that determine 3G call for greater bandwidth and higher speeds for cellular networks
AES	The Advanced Encryption Standard (AES) is a new block cipher standard to replace DES, developed by NIST, the US National Institute of Standards and Technology. AES ciphers use a 128-bit block and 128-, 192-, or 256-bit keys. The larger block size helps resist birthday attacks while the large key size prevents brute force attacks.
APN	Access Point Name (APN) is used by carriers to identify an IP packet data network that a mobile data user wants to communicate with and the type of wireless service
Authentication	Authentication is the technique by which a process verifies that its communication partner is who it is supposed to be and not an imposter. Authentication confirms that data is sent to the intended recipient and assures the recipient that the data originated from the expected sender and has not been altered on route
BIOS	Basic Input/Output System is the built-in software in a computer that are executed on startup (boot) and that determine what the computer can do without accessing programs from a disk. On PCs, the BIOS contains all the code required to control the keyboard, display screen, disk drives, serial communications, and a number of miscellaneous functions
Bonding	Ethernet Bonding or Failover is the ability to detect communication failure transparently, and switch from one LAN connection to another.
BOOTP	Bootstrap Protocol. A protocol that allows a network user to automatically receive an IP address and have an operating system boot without user interaction. BOOTP is the basis for the more advanced DHCP
Certificates	A digitally signed statement that contains information about an entity and the entity's public key, thus binding these two pieces of information together. A certificate is issued by a trusted organization (or entity) called a Certification Authority (CA) after the CA has verified that the entity is who it says it is.
Certificate Authority	A Certificate Authority is a trusted third party, which certifies public key's to truly belong to their claimed owners. It is a key part of any Public Key Infrastructure, since it allows users to trust that a given public key is the one they wish to use, either to send a private message to its owner or to verify the signature on a message sent by that owner.
Certificate Revocation List	A list of certificates that have been revoked by the CA before they expired. This may be necessary if the private key certificate has been compromised or if the holder of the certificate is to be denied the ability to establish a connection to the *console server*.
CHAP	Challenge-Handshake Authentication Protocol (CHAP) is used to verify a user's name and password for PPP Internet connections. It is more secure than PAP, the other main authentication protocol.
DES	The Data Encryption Standard is a block cipher with 64-bit blocks and a 56-bit key.
DHCP	Dynamic Host Configuration Protocol. A communications protocol that assigns IP addresses to computers when they are connected to the network.
DNS	Domain Name System that allocates Internet domain names and translates them into IP addresses. A domain name is a meaningful and easy to remember name for an IP address.
DUN	Dial Up Networking

Encryption	The technique for converting a readable message (plaintext) into apparently random material (ciphertext) which cannot be read if intercepted. The proper decryption key is required to read the message.
Ethernet	A physical layer protocol based upon IEEE standards
Firewall	A network gateway device that protects a private network from users on other networks. A firewall is usually installed to allow users on an intranet access to the public Internet without allowing public Internet users access to the intranet.
Gateway	A machine that provides a route (or pathway) to the outside world.
Hub	A network device that allows more than one computer to be connected as a LAN, usually using UTP cabling.
Internet	A worldwide system of computer networks - a public, cooperative, and self-sustaining network of networks accessible to hundreds of millions of people worldwide. The Internet is technically distinguished because it uses the TCP/IP set of protocols.
Intranet	A private TCP/IP network within an enterprise.
IPMI	Intelligent Platform Management Interface (IPMI) is a set of common interfaces to a computer system which system administrators can use to monitor system health and manage the system. The IPMI standard defines the protocols for interfacing with a service processor embedded into a server platform.
Key lifetimes	The length of time before keys are renegotiated
LAN	Local Area Network
LDAP	The Lightweight Directory Access Protocol (LDAP) is based on the X.500 standard, but significantly simpler and more readily adapted to meet custom needs. The core LDAP specifications are all defined in RFCs. LDAP is a protocol used to access information stored in an LDAP server.
LED	Light-Emitting Diode
MAC address	Every piece of Ethernet hardware has a unique number assigned to it called its MAC address. Ethernet is used locally to connect the *console server* to the Internet, and it may share the local network with many other appliances. The MAC address is used by the local Internet router in order to direct *console server* traffic to it rather than somebody else in the local area. It is a 48-bit number usually written as a series of 6 hexadecimal octets, *e.g.* 00:d0:cf:00:5b:da. A *console server* has a MAC address listed on a label underneath the device.
MSCHAP	Microsoft Challenge Handshake Authentication Protocol (MSCHAP) is authentication for PPP connections between a computer using a Microsoft Windows operating system and a network access server. It is more secure than PAP or CHAP, and is the only option that also supports data encryption.
NAT	Network Address Translation. The translation of an IP address used on one network to an IP address on another network. Masquerading is one particular form of NAT.
Net mask	The way that computers know which part of a TCP/IP address refers to the network, and which part refers to the host range.
NFS	Network File System is a protocol that allows file sharing across a network. Users can view, store, and update files on a remote computer.
NTP	Network Time Protocol (NTP) used to synchronize clock times in a network of computers
OUT OF BAND	Out-of-Band (OOB) management is any management done over channels and interfaces that are separate from those used for user/customer data. Examples would include a serial console interface or a network interface connected to a

	dedicated management network that is not used to carry customer traffic, or to a BMC/service processor. Any management done over the same channels and interfaces used for user/customer data is In Band.
PAP	Password Authentication Protocol (PAP) is the usual method of user authentication used on the internet: sending a username and password to a server where they are compared with a table of authorized users. Whilst most common, PAP is the least secure of the authentication options.
PPP	Point-to-Point Protocol. A networking protocol for establishing simple links between two peers.
RADIUS	The Remote Authentication Dial-In User Service (RADIUS) protocol was developed by Livingston Enterprises as an access server authentication and accounting protocol. The RADIUS server can support a variety of methods to authenticate a user. When it is provided with the username and original password given by the user, it can support PPP, PAP or CHAP, UNIX login, and other authentication mechanisms.
Router	A network device that moves packets of data. A router differs from hubs and switches because it is "intelligent" and can route packets to their final destination.
SIM	Subscriber Identity Module (SIM) card stores unique serial numbers and security authentication used to identify a subscriber on mobile telephony devices
SMASH	Systems Management Architecture for Server Hardware is a standards-based protocols aimed at increasing productivity of the management of a data center. The SMASH Command Line Protocol (SMASH CLP) specification provides an intuitive interface to heterogeneous servers independent of machine state, operating system or OS state, system topology or access method. It is a standard method for local and remote management of server hardware using out-of-band communication
SMTP	Simple Mail Transfer Protocol. *console server* includes, SMTPclient, a minimal SMTP client that takes an email message body and passes it on to a SMTP server (default is the MTA on the local host).
SOL	Serial Over LAN (SOL) enables servers to transparently redirect the serial character stream from the baseboard universal asynchronous receiver/transmitter (UART) to and from the remote-client system over a LAN. With SOL support and BIOS redirection (to serial) remote managers can view the BIOS/POST output during power on, and reconfigured.
SSH	Secure Shell is secure transport protocol based on public-key cryptography.
SSL	Secure Sockets Layer is a protocol that provides authentication and encryption services between a web server and a web browser.
TACACS+	The Terminal Access Controller Access Control System (TACACS+) security protocol is a more recent protocol developed by Cisco. It provides detailed accounting information and flexible administrative control over the authentication and authorization processes. TACACS+ allows for a single access control server (the TACACS+ daemon) to provide authentication, authorization, and accounting services independently. Each service can be tied into its own database to take advantage of other services available on that server or on the network, depending on the capabilities of the daemon. There is a draft RFC detailing this protocol.
TCP/IP	Transmission Control Protocol/Internet Protocol. The basic protocol for Internet communication.
TCP/IP address	Fundamental Internet addressing method that uses the form nnn.nnn.nnn.nnn.
Telnet	Telnet is a terminal protocol that provides an easy-to-use method of creating terminal connections to a network.
UDP	User Datagram Protocol

UTC	Coordinated Universal Time.
UTP	Unshielded Twisted Pair cabling. A type of Ethernet cable that can operate up to 100Mb/s. Also known as Category 5 or CAT 5.
VNC	Virtual Network Computing (VNC) is a desktop protocol to remotely control another computer. It transmits the keyboard presses and mouse clicks from one computer to another relaying the screen updates back in the other direction, over a network.
VPN	Virtual Private Network (VPN) a network that uses a public telecommunication infrastructure and Internet, to provide remote offices or individual users with secure access to their organization's network
WAN	Wide Area Network
WINS	Windows Internet Naming Service (WINS) that manages the association of workstation names and locations with IP addresses

APPENDIX F: END USER LICENSE AGREEMENTS

READ BEFORE USING THE ACCOMPANYING SOFTWARE

YOU SHOULD CAREFULLY READ THE FOLLOWING TERMS AND CONDITIONS BEFORE USING THE ACCOMPANYING SOFTWARE, THE USE OF WHICH IS LICENSED FOR USE ONLY AS SET FORTH BELOW. IF YOU DO NOT AGREE TO THE TERMS AND CONDITIONS OF THIS AGREEMENT, DO NOT USE THE SOFTWARE. IF YOU USE ANY PART OF THE SOFTWARE, SUCH USE WILL INDICATE THAT YOU ACCEPT THESE TERMS.

You have acquired a product that includes Opengear ("Opengear") proprietary software and/or proprietary software licensed to Opengear. This Opengear End User License Agreement ("EULA") is a legal agreement between you (either an individual or a single entity) and Opengear for the installed software product of Opengear origin, as well as associated media, printed materials, and "online" or electronic documentation ("Software"). By installing, copying, downloading, accessing, or otherwise using the Software, you agree to be bound by the terms of this EULA. If you do not agree to the terms of this EULA, Opengear is not willing to license the Software to you. In such event, do not use or install the Software. If you have purchased the Software, promptly return the Software and all accompanying materials with proof of purchase for a refund.

Products with separate end user license agreements that may be provided along with the Software are licensed to you under the terms of those separate end user license agreements.

LICENSE GRANT. Subject to the terms and conditions of this EULA, Opengear grants you a nonexclusive right and license to install and use the Software on a single CPU, provided that, (1) you may not rent, lease, sell, sublicense or lend the Software; (2) you may not reverse engineer, decompile, disassemble or modify the Software, except and only to the extent that such activity is expressly permitted by applicable law notwithstanding this limitation; and (3) you may not transfer rights under this EULA unless such transfer is part of a permanent sale or transfer of the Product, you transfer at the same time all copies of the Software to the same party or destroy such materials not transferred, and the recipient agrees to this EULA.

No license is granted in any of the Software's proprietary source code. This license does not grant you any rights to patents, copyright, trade secrets, trademarks or any other rights with respect to the Software.

You may make a reasonable number of copies of the electronic documentation accompanying the Software for each Software license you acquire, provided that, you must reproduce and include all copyright notices and any other proprietary rights notices appearing on the electronic documentation. Opengear reserves all rights not expressly granted herein.

INTELLECTUAL PROPERTY RIGHTS. The Software is protected by copyright laws, international copyright treaties, and other intellectual property laws and treaties. Opengear and its suppliers retain all ownership of, and intellectual property rights in (including copyright), the Software components and all copies thereof, provided however, that (1) certain components of the Software, including *SDT Connector*, are components licensed under the GNU General Public License Version 2, which Opengear supports, and (2) the *SDT Connector* includes code from JSch, a pure Java implementation of SSH2 which is licensed under BSD style license. Copies of these licenses are detailed below and Opengear will provide source code for any of the components of the Software licensed under the GNU General Public License upon request.

EXPORT RESTRICTIONS. You agree that you will not export or re-export the Software, any part thereof, or any process or service that is the direct product of the Software in violation of any applicable laws or regulations of the United States or the country in which you obtained them.

U.S. GOVERNMENT RESTRICTED RIGHTS. The Software and related documentation are provided with Restricted Rights. Use, duplication, or disclosure by the Government is subject to restrictions set forth in subparagraph (c) (1) (ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013 or subparagraphs (c) (1) and (2) of the Commercial Computer Software – Restricted Rights at 48 C.F.R. 52.227-19, as applicable, or any successor regulations.

TERM AND TERMINATION. This EULA is effective until terminated. The EULA terminates immediately if you fail to comply with any term or condition. In such an event, you must destroy all copies of the Software. You may also terminate this EULA at any time by destroying the Software.

GOVERNING LAW AND ATTORNEY'S FEES. This EULA is governed by the laws of the State of Utah, USA, excluding its conflict of law rules. You agree that the United Nations Convention on Contracts for the International Sale of Goods is hereby excluded in its entirety and does not apply to this EULA. If you acquired this Software in a country outside of the United States, that country's laws may apply. In any action or suit to enforce any right or remedy under this EULA or to interpret any provision of this EULA, the prevailing party will be entitled to recover its costs, including reasonable attorneys' fees.

ENTIRE AGREEMENT. This EULA constitutes the entire agreement between you and Opengear with respect to the Software, and supersedes all other agreements or representations, whether written or oral. The terms of this EULA can only be modified by express written consent of both parties. If any part of this EULA is held to be unenforceable as written, it will be enforced to the maximum extent allowed by applicable law, and will not affect the enforceability of any other part.

Should you have any questions concerning this EULA, or if you desire to contact Opengear for any reason, please contact the Opengear representative serving your company.

THE FOLLOWING DISCLAIMER OF WARRANTY AND LIMITATION OF LIABILITY IS INCORPORATED INTO THIS EULA BY REFERENCE. THE SOFTWARE IS NOT FAULT TOLERANT. YOU HAVE INDEPENDENTLY DETERMINED HOW TO USE THE SOFTWARE IN THE DEVICE, AND OPENGEAR HAS RELIED UPON YOU TO CONDUCT SUFFICIENT TESTING TO DETERMINE THAT THE SOFTWARE IS SUITABLE FOR SUCH USE.

LIMITED WARRANTY Opengear warrants the media containing the Software for a period of ninety (90) days from the date of original purchase from Opengear or its authorized retailer. Proof of date of purchase will be required. Any updates to the Software provided by Opengear (which may be provided by Opengear at its sole discretion) shall be governed by the terms of this EULA. In the event the product fails to perform as warranted, Opengear's sole obligation shall be, at Opengear's discretion, to refund the purchase price paid by you for the Software on the defective media, or to replace the Software on new media. Opengear makes no warranty or representation that its Software will meet your requirements, will work in combination with any hardware or application software products provided by third parties, that the operation of the software products will be uninterrupted or error free, or that all defects in the Software will be corrected.

OPENGEAR DISCLAIMS ANY AND ALL OTHER WARRANTIES, WHETHER EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. OTHER THAN AS STATED HEREIN, THE ENTIRE RISK AS TO SATISFACTORY QUALITY, PERFORMANCE, ACCURACY, AND EFFORT IS WITH YOU. ALSO, THERE IS NO WARRANTY AGAINST INTERFERENCE WITH YOUR ENJOYMENT OF THE SOFTWARE OR AGAINST INFRINGEMENT. IF YOU HAVE RECEIVED ANY WARRANTIES REGARDING THE DEVICE OR THE SOFTWARE, THOSE WARRANTIES DO NOT ORIGINATE FROM, AND ARE NOT BINDING ON, OPENGEAR.

NO LIABILITY FOR CERTAIN DAMAGES. EXCEPT AS PROHIBITED BY LAW, OPENGEAR SHALL HAVE NO LIABILITY FOR COSTS, LOSS, DAMAGES OR LOST OPPORTUNITY OF ANY TYPE WHATSOEVER, INCLUDING BUT NOT LIMITED TO, LOST OR ANTICIPATED PROFITS, LOSS OF USE, LOSS OF DATA, OR ANY INCIDENTAL, EXEMPLARY SPECIAL OR CONSEQUENTIAL DAMAGES, WHETHER UNDER CONTRACT, TORT, WARRANTY OR OTHERWISE ARISING FROM OR IN CONNECTION WITH THIS EULA OR THE USE OR PERFORMANCE OF THE SOFTWARE. IN NO EVENT SHALL OPENGEAR BE LIABLE FOR ANY AMOUNT IN EXCESS OF THE LICENSE FEE PAID TO OPENGEAR UNDER THIS EULA. SOME STATES AND COUNTRIES DO NOT ALLOW THE LIMITATION OR EXCLUSION OF LIABILITY FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO THIS LIMITATION MAY NOT APPLY TO YOU.

JSch License

SDT Connector includes code from JSch, a pure Java implementation of SSH2. JSch is licensed under BSD style license and it is:

Copyright (c) 2002, 2003, 2004 Atsuhiko Yamanaka, JCraft,Inc. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. The names of the authors may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED ``AS IS'' AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL JCRAFT, INC. OR ANY CONTRIBUTORS TO THIS SOFTWARE BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

GNU GENERAL PUBLIC LICENSE
TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under
the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by

the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and
of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

Wireless Driver License

The Opengear firmware includes 802.11 driver code which is used in various console server models. This code is:

Copyright (c) 2007, Ralink Technology Corporation All rights reserved.

Redistribution and use in binary form, without modification, are permitted provided that the following conditions are met:

- Redistributions must reproduce the above copyright notice and the following disclaimer in the documentation and/or other materials provided with the distribution
- Neither the name of Ralink Technology Corporation nor the names of its suppliers may be used to endorse or promote products derived from this software without specific prior written permission
- No reverse engineering, decompilation, or disassembly of this software is permitted

Ralink Technology Corporation grants a world-wide, royalty-free, non-exclusive license under patents it now or hereafter owns or controls to make, have made, use, import, offer to sell and sell ("Utilize") this software, but solely to the extent that any such patent is necessary to Utilize the software alone, or in combination with an operating system licensed under an approved Open Source license as listed by the Open Source Initiative at http://opensource.org/licenses. The patent license shall not apply to any other combinations which include this software. No hardware per se is licensed hereunder.

DISCLAIMER. THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY

AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT,
INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE
USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH

APPENDIX G: SERVICE & STANDARD WARRANTY

STANDARD WARRANTY

Opengear, Inc., its parent, affiliates and subsidiaries, (collectively, "Opengear") warrant your Opengear product to be in good working order and to be free from defects in workmanship and material (except in those cases where the materials are supplied by the Purchaser) under normal and proper use and service for the period of four (4) years from the date of original purchase from an Authorized Opengear reseller. In the event that this product fails to meet this warranty within the applicable warranty period, and provided that Opengear confirms the specified defects, Purchaser's sole remedy is to have Opengear, in Opengear's sole discretion, repair or replace such product at the place of manufacture, at no additional charge other than the cost of freight of the defective product to and from the Purchaser. Repair parts and replacement products will be provided on an exchange basis and will be either new or reconditioned. Opengear will retain, as its property, all replaced parts and products. Notwithstanding the foregoing, this hardware warranty does not include service to replace or repair damage to the product resulting from accident, disaster, abuse, misuse, electrical stress, negligence, any non- Opengear modification of the product except as provided or explicitly recommended by Opengear, or other cause not arising out of defects in material or workmanship. This hardware warranty also does not include service to replace or repair damage to the product if the serial number or seal or any part thereof has been altered, defaced or removed. If Opengear does not find the product to be defective, the Purchaser will be invoiced for said inspection and testing at Opengear's then current rates, regardless of whether the product is under warranty.

RMA RETURN PROCEDURE

If this product requires service during the applicable warranty period, a Return Materials Authorization (RMA) number must first be obtained from Opengear. Product that is returned to Opengear for service or repair without an RMA number will be returned to the sender unexamined. Product should be returned, freight prepaid, in its original or equivalent packaging, to:

Opengear Service Center
Suite A, 630 West 9560 South
Sandy, Utah 84070

Proof of purchase date must accompany the returned product and the Purchaser shall agree to insure the product or assume the risk of loss of damage in transit. Contact Opengear by emailing support@opengear.com for further information.

TECHNICAL SUPPORT

Purchaser is entitled to thirty (30) days free telephone support and twelve (12) months free e-mail support (worldwide) from date of purchase provided that the Purchaser first register their product(s) with Opengear by filling in the on-line form http://www.opengear.com/registration.html.

Direct telephone, help-desk and e-mail support is available from 9:00 AM to 5:00 PM, Mountain Time. http://www.opengear.com/support.htm.l

Opengear's standard warranty includes free access to Opengear's Knowledge Base as well as any application notes, white papers and other on-line resources that may become available from time to time.

Opengear reserves the right to discontinue all support for products that are no longer covered by warranty.

LIMITATION OF LIABILITY

No action, regardless of form, arising from this warranty may be brought by either party more than two (2) years after the cause of action has occurred. Purchaser expressly agrees that Opengear's liability, if any, shall be limited solely to the replacement or repair of the product in accordance with the warranties specifically and expressly set forth herein. The remedies of the Purchaser are the exclusive and sole remedies available, and, in the event of a breach or repudiation of any provision of this agreement by Opengear, the Purchaser shall not be entitled to receive any incidental damages as that term is defined in Section 2-715 of the Uniform Commercial Code. Opengear waives the benefit of any rule that disclaimer of warranty shall be construed against Opengear and agrees that such disclaimers herein shall be construed liberally in favor of Opengear.

THE FOREGOING WARRANTIES ARE THE SOLE ANDEXCLUSIVE WARRANTIES GIVEN IN CONNECTION WITH THE PRODUCT AND THE HARDWARE. OPENGEAR DISCLAIMS ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION, ANY WARRANTIES AS TO THE SUITABILITY OR MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. OPENGEAR DOES NOT PROMISE THAT THE PRODUCT IS ERROR-FREE OR WILL OPERATE WITHOUT INTERRUPTION. IN NO EVENT SHALL OPENGEAR BE LIABLE FOR ANY LOST OR ANTICIPATED PROFITS, OR ANY INCIDENTAL, EXEMPLARY, SPECIAL OR CONSEQUENTIAL DAMAGES, REGARDLESS OF WHETHER OPENGEAR WAS ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.