LISTEN.
THINK.
SOLVE.SM

INTEGRATED PRODUCTION & PERFORMANCE SUITE

Data Management

FactoryTalk® Historian SE

HIGH AVAILABILITY AND REPLICATION GUIDE

PUBLICATION HSEHA-RM020A-EN-E–September 2007

ALLEN-BRADLEY • ROCKWELL SOFTWARE

Rockwell Automation

Contact Rockwell

Customer Support Telephone — 1.440.646.3434

Online Support — http://support.rockwellautomation.com

Warranty

This product is warranted in accordance with the product license. The product's performance may be affected by system configuration, the application being performed, operator control, maintenance, and other related factors. Rockwell Automation is not responsible for these intervening factors. The instructions in this document do not cover all the details or variations in the equipment, procedure, or process described, nor do they provide directions for meeting every possible contingency during installation, operation, or maintenance. This product's implementation may vary among users.

This document is current as of the time of release of the product; however, the accompanying software may have changed since the release. Rockwell Automation, Inc. reserves the right to change any information contained in this document or the software at anytime without prior notice. It is your responsibility to obtain the most current information available from Rockwell when installing or using this product.

Version: 9.00.05

Contents

Chapter 1. Introduction

About this Book

High Availability and PI Server Replication is part of the Rockwell Automation *PI Server Documentation Set*. High Availability is accomplished using replicated PI Server machines, which provide redundant sources of time-series data to PI clients. This guide presents options for configuring server replication, and guides you through configuration. See the *Introduction to PI Server System Management* and *PI Server System Management Guide* for more information regarding PI Server administration tools discussed in this guide.

PI Server Documentation Set is included on the FactoryTalk Historian SE installation CD under Redist > Docs.

PI Server High Availability (HA)

PI Server High Availability (HA) enhances the reliability of the PI Server by providing alternate sources of the same time-series data for users. PI Server Replication enables alternate data sources by synchronizing the configuration of multiple servers. This allows interfaces to buffer data to multiple servers with the same point configuration. PI SDK clients can retrieve data from any of the servers without changing any data reference.

Motivating Conditions

While the PI System is a robust platform, single points of failure do exist, and routine maintenance and upgrades sometimes require restarting the PI Server software or rebooting the server operating system. These restarts make the server unavailable for some time.

The PI System is becoming more important for operations in many industries. A single server cannot always provide data to all classes of users. Some industry regulations require more than one server. The needs of some users grow beyond the capability of a single server to give timely responses.

This section describes the conditions that could cause a non-replicated PI Server to be unavailable. It also lists other motivations for adding multiple servers.

Operating System Updates

The PI Server typically runs on a general-purpose server-class computer running the Microsoft Windows operating system. To counteract increased security risks, it is often necessary to apply recommended patches or updates to keep the operating system current. Some patches require rebooting the computer. Even patches that are advertised to require no

reboot carry the risk that they require rebooting the computer because of unanticipated dependencies.

It is difficult to schedule server downtime in many industries. The risk of making a PI Server unavailable drives PI administrators to delay applying patches or to apply patches during times of non-peak usage.

Software Upgrades

Rockwell Automation is constantly improving the PI System. In order to install new versions of the software, the PI Server must be shut down. During this time, users cannot access historical or current data. Because of the architecture of the PI System, the interface computers continue to buffer so that no data is lost.

The problem is similar to the one caused by operating system updates. The loss of availability of the PI Server during the upgrade drives administrators to delay upgrades or to only apply upgrades during times of non-peak usage.

Software Failure

The PI Server software is widely tested in a variety of configurations prior to release. Because there is so much configuration flexibility and customer creativity inspired by need, it is possible for customers to establish system configurations that are outside of design specifications or tested scenarios. Recovery from the rare occurrence of PI System software failure often requires restarting the PI System.

Some administrators avoid additional server configuration risk by allowing changes only at certain times, often during times of non-peak usage.

Hardware Failure

Computing hardware has become more reliable. With the increase in reliability, organizations do not stock as many spare parts and do not test plans for rapid repair or restoration. When a hardware failure occurs, the PI Server is unavailable until the hardware is replaced or a backup system is activated. Although the incidence of failure has decreased, the experience of some organizations has been an increase in the duration of each hardware outage.

Hardware Migration

Many organizations schedule server hardware replacement in an attempt to avoid the higher costs of unplanned hardware failure or the anticipated resource limitation of old hardware. During some period of the migration to new hardware, the PI Server is not available.

Network Failure

Network communication between interface computers, server computers, and client computers is required for users to obtain data. Networks can fail because a piece of network hardware fails or some network software configuration is improperly modified. Because the network is distributed, the network can also fail because a cable is destroyed or a service provider experiences problems.

Human Error

Systems are becoming more integrated, with more connections between each one. System administrators have more systems with greater diversity to manage and less time to become familiar with any one system. These factors increase the probability of an authorized person making a change that has unexpected negative consequences. Recovery from these situations often requires a PI Server restart or operating system reboot. Sometimes the PI Server or the operating system must be restored from a backup

Load Distribution

Load distribution attempts to balance server traffic among a group of servers by distributing user connections to the servers. Ideally, a new connection would be directed at a server in the group that had the lowest load.

Segregation of Users by Class of Service

Users have different needs for service. Some users, such as process operators, have immediate needs for data and must have access to any available server including some reserved for them exclusively. Some users have moderate needs for data, and should have access to any server except those reserved for process operators. Some users run intensive data mining operations that consume a large amount of server resources but can run slowly or be deferred. These users must have access only to servers that do not impact the needs of the process operators and moderate need users.

Segregation of users by class of service directs user connections to an available server that meets their needs but does not impact higher class services. When the server that they are connected to becomes unavailable, their connection should be re-established to another server if possible.

Geographic Separation

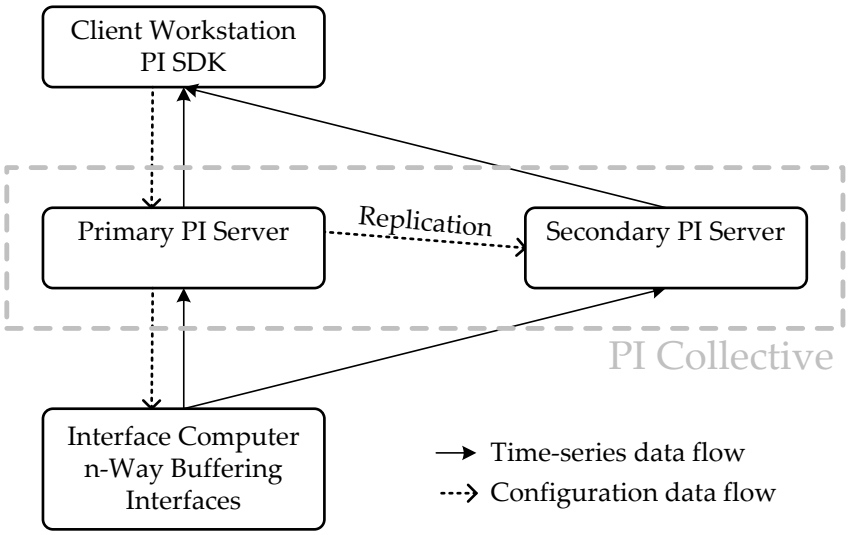
Regulatory agencies require some operating companies to establish a large geographic separation between the redundant servers. For example FERC requires power distribution companies to maintain separation between their normal operations center and a backup operations center. Some industries are planning for geographic diversity as an outcome of risk management and disaster recovery planning.

Overall Architecture

Multiple PI Servers

The overall design provides multiple PI Servers that each act as an independent source for time-series data. These PI Servers are considered to be members of a *PI Collective*. The following figure shows the architecture of a collective consisting of two PI Servers.

Configuration changes are allowed on only one PI Server, which is designated the *primary*. Other PI Servers that do not allow configuration changes are designated as *secondary*.



Collective Architecture

PI Server Replication keeps the configuration databases consistent on the servers. Any change made to the primary PI Server configuration moves to the secondary PI Servers in real time, according to a fixed schedule or on demand.

Interface N-Way Buffering

The interfaces send time-series data to a buffering service. This buffering service can be configured for any interface that uses the PI API. The buffering service queues the time-series data independently to each PI Server in a process called *n-way buffering*. Each server receives the time-series data from the interfaces and performs its functions independently.

Enhanced PI SDK for Client Connections

The PI SDK is enhanced to treat the set of equivalent servers in the Collective as one logical data source. When the server providing data is shut down normally through the `pisrvstop.bat` batch file, the PI SDK receives an early notification of shutdown and immediately connects to an alternate server. When the server providing data becomes unresponsive because of hardware or software problems, or unreachable because of network problems, the PI SDK connects to an alternate server after a short time-out period.

Applications that use the PI SDK can take advantage of the failover behavior on any computer where the PI SDK version 1.3.4 or later is installed.

Older applications that use the PI SDK 1.3.3 or earlier continue to function as before. You can use PI SDK 1.3.4 or later with both old and new applications with the PI Servers in the collective. Older applications using the PI SDK 1.3.3 or earlier do not benefit from the failover behavior.

PI Server High Availability Concepts and Features

PI Server Collective

The PI Server Collective consists of PI Servers that have the same configuration database. This provides the same association between the key values in the PI tables on all of the servers. This also ensures that the archive data files have the same structure on all of the servers.

For example, each PI Server in the PI Server Collective maintains the same tag configuration. The PI Point table has the same association between Tag and PointID on every server in the Collective. This association allows applications such as FactoryTalk Historian ProcessBook to refer to the same Tag on all of the servers, even if they translate the Tag to PointID and cache the resulting PointID.

Another example of the association of key values is the association between PointID and RecNo in the PI point table and archive files. Maintaining this association on different servers in the collective keeps the archive structure the same on all of the servers. This allows archives to be moved around and reprocessed as if they all came from the same PI Server.

Maintaining identical configuration is central to the ease of use for the PI user and ease of maintenance for the PI system administrator.

PI SDK Automatic Client Failover

The PI SDK 1.3.4 or later enables client applications to take advantage of the alternate sources of data by automatically failing over between servers in a collective. The **PI Connection Manager** dialog box shows the server in the collective that is supplying the data and allows setting the priority of the server that is used for the connection.

Client applications that use the PI API continue to work by connecting to each server individually.

See *PI SDK Server Connection and Automatic Client Failover* (page 13) for more details.

PI Server Replication

PI Server Replication on each secondary PI Server signs up for a configuration change log (similar to the transaction log of a relational database) from the primary PI Server. Configuration changes can be entered only on the primary PI Server. This keeps the configuration database tables synchronized on all of the servers.

PI Server Replication is started by configuring two new tables. See *Configure PICollective and PIServer Tables* (page 36) for more details.

N-Way Buffering

PI API buffering is configured on most computers with PI interfaces. The buffering service stores time-series values to disk when the interface computer cannot communicate with the PI Server.

The term *n-way buffering* denotes the extension to this behavior where buffers are maintained for each target server in the collective. In this release of the PI Server, two versions of n-way buffering are available.

BufServ provides n-way buffering for PI API interfaces on the widest variety of operating systems, but lets the servers run the compression algorithm independently of each other. This can result in different time-series events being selected for archiving. In order for configuration changes to BufServ to take effect, you must stop the interfaces, stop BufServ, restart BufServ, and finally restart the interfaces.

The *PI Buffer Subsystem*, or **pibufss**, provides a simpler option for managing n-way buffering. The Buffer Subsystem can react to collective configuration changes without requiring that the interfaces shut down. For example, buffering will start to a server added to the collective without having to shut down the interfaces.

The PI Buffer Subsystem runs the compression algorithm before the time-series data is sent to the PI Server. It marks time series values so that the servers store and serve identical time-series events to and from the archive files.

See the PI Buffer Subsystem help in `PIHOME\Help\pibufss.chm` (typically `C:\Program Files\Rockwell Software\FactoryTalk Historian\Server\PIPC\HELP`) for details on **pibufss**.

See *Upgrade and Configure Interfaces and Buffering on Interface Computers* (page 31) for more details on n-way buffering.

PI Interface Failover

Most standard PI Interfaces based on UniInt can potentially support standard interface failover. UniInt is the Rockwell Automation template for interfaces to the PI System.

Depending on the data source, an interface built with UniInt 4.2.3 or later can automatically switch between redundant copies of the interface run on separate interface computers. This provides uninterrupted collection of process data even when one of the interfaces is unable to collect data for any reason. When maintenance, hardware failure, or network failure causes one interface to become unavailable, the redundant interface computer automatically starts collecting, buffering, and sending data to the PI Server.

See *Interface Level Failover* in the *UniInt End User Guide* for more information.

PI Interface Restart without Server Connection

Most standard PI Interfaces based on UniInt can restart without a connection to the PI Server.

The first time an interface starts, it must obtain the list of points from the specified PI Server (home node) in order to scan and collect information from the data source. An interface built with UniInt 4.3 or later caches this information locally.

As the interface runs, it receives updates to the list of points and their parameters and writes the information—including the point scan list—to a local disk file. Subsequent starts of the interface can use the local copy of the point information to start up without a connection to the PI Server. See *Interface Restart without Server Connection* in the *UniInt End User Guide* for details.

PI to PI

The PI to PI interface allows data to be sent from one PI Server to another PI Server. This can be used to augment n-way buffering.

You can configure PI to PI to send time-series data from one PI Server to other PI Servers in the collective when you do not want every secondary PI Server to receive time-series data from every interface. This can be useful when firewalls are involved. It is also useful if you want to send manually entered time-series data to the other servers in the collective. *Routing Time-Series Data through a Firewall* (page 9) shows how to use PI to PI to send data through a firewall so that you do not have to configure n-way buffering on the interface nodes.

For systems receiving data from non-buffered sources, the PI to PI interface can be useful in moving the time-series data among the members of the collective.

See the *PI to PI TCP/IP Interface* manual for more information.

Online Backup

The PI Server can be backed up while it is running. In PI Server version 3.4.370 and later, the backup does not even disrupt end users. The backup subsystem takes advantage of Microsoft Volume Shadow Copy Services (VSS) if it is available.

Even though PI Server Replication provides a second server for client data retrieval, it is still important to back up the PI Server. It is most important to back up the primary PI Server, because that is where configuration changes are entered. It is often a good idea to back up the secondary server as well.

See *Backing up the PI Server* in the *PI Server System Management Guide* for more information.

Redundant Hardware

The HA features described here are compatible with the use of redundant hardware. Any of the servers in the collective or the PI interface computers can be run on redundant hardware. This provides an excellent solution for local availability.

Redundant hardware manufacturers include Stratus Technologies (http://www.stratus.com/) and Marathon Technologies (http://www.marathontechnologies.com/). For more information about using redundant hardware with the PI Server, search for "PI System Sizing and Configuration" on the Rockwell Automation technical support web site.

Operating System Clustering

The HA features described here are compatible with the use of clustering provided by Microsoft Cluster Services. Any of the servers in the PI Collective or the PI interface computers can be clustered. This provides an alternate solution for local availability.

The Collective Manager released with PR1 cannot be used to manage PI Servers running in a clustered environment. It can be used to view the current state and edit many parameters of PI Servers in a collective running in a clustered environment. You must use command-line tools to initialize a primary PI Server, initialize a secondary PI Server, or re-initialize a secondary PI Server running in a clustered environment.

For more information about installing PI in a Microsoft Cluster Services environment, see *PI Server Cluster Installation* in the *PI Server Installation and Upgrade Guide*. For more information about using Microsoft Cluster Services with the PI Server, search for "cluster" on the Rockwell Automation technical support web site.

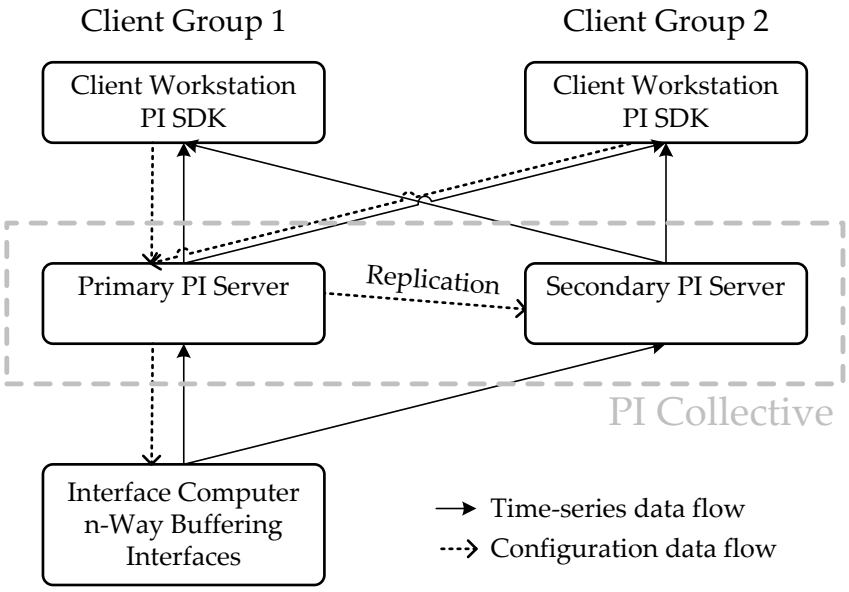
Comment [PHG1]: Hidden pending resolution discussion between PI3 Team and others

PI Server HA Deployment Patterns

The PI Server provides many options for configuration. The HA features extend this flexibility. Since there are many possible configurations, this section presents some deployment patterns that can inspire the starting point for your system layout. The deployment patterns are neither all-inclusive nor limiting, and can be combined and extended.

Client Failover and Load Distribution

The following figure shows the simplest HA deployment pattern that employs replication. The primary PI Server receives all configuration changes. PI Server Replication moves these changes to a single secondary PI Server.



Clients Connect to any PI Server in the Collective

The figure shows only one interface and one interface computer, but there are usually several. Each of the interfaces could employ interface level failover and interface restart without a PI connection. All interfaces send time-series data to both PI Servers via n-way buffering. The interfaces receive configuration changes from one PI Server configured in the startup file. In most cases you will want to configure the interface so that it receives configuration data from the primary server.

Clients connect to the PI Servers through the PI SDK, which treats the servers as one logical server and provides for failover between the servers. When an application requests data, the PI SDK connects initially to one of the servers. Individual workstations can be configured to connect to PI Servers in a preferred order. This provides a simple form of static load distribution.

In the figure, the **Client Group 1** workstations could be configured to connect to the primary PI Server first, then the secondary PI Server. The **Client Group 2** workstations could be configured to connect to the secondary PI Server first, then the primary PI Server. As long as both servers are working, the load is distributed between them.

When the PI Server is shut down normally with the `pisrvstop.bat` command file, the Network Manager Subsystem notifies the PI SDK, which immediately attempts to connect to another PI Server. If a PI Server stops responding, the PI SDK eventually times out. The loss of communication could occur due to network interruption or hardware problem. Regardless of the cause of the time-out, the PI SDK attempts to connect to another PI Server.

Routing Time-Series Data through a Firewall

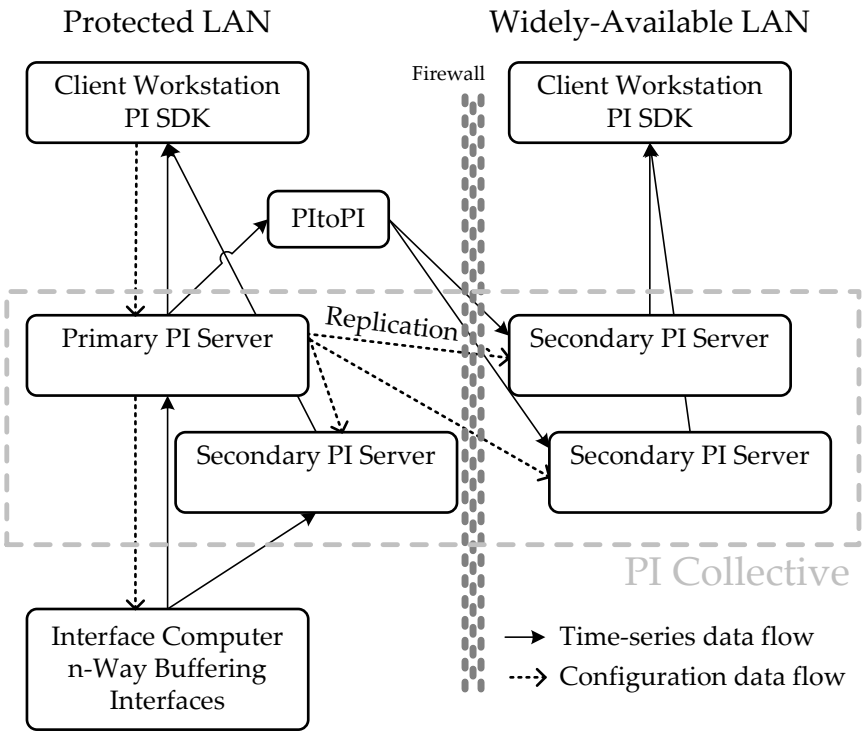
The following figure shows a more complex HA deployment pattern that employs replication. The main design feature is a firewall that isolates the PI Servers on a protected LAN from a large number of users on a widely available LAN. In this case there is one primary PI Server and one secondary PI Server on the protected LAN inside the firewall. There are two secondary servers and a large group of users outside the firewall on the widely available LAN.

The client workstations on the widely available LAN are configured to connect only to the PI Servers on the widely available LAN. The client workstations on the protected LAN are configured to connect to the PI Servers on the protected LAN, and could be configured to connect to the PI Servers on the widely available LAN as well.

As with the previous pattern, the primary PI Server receives all configuration changes from client applications. PI Server Replication running on the secondary PI Servers moves these changes to all three secondary PI Servers.

The figure shows only one interface and one interface computer, but there are usually several. The interfaces receive configuration information from the primary PI Server. The interfaces send time-series data to the PI Servers on the protected LAN via n-way buffering.

One or more PI to PI interfaces running on the protected LAN forwards time-series data from the primary server to the PI Servers on the widely available LAN. This reduces the diversity of traffic and number of data sources that would flow through the firewall from the protected LAN to the widely available LAN if n-way buffering sent time-series data to all of the secondary PI Servers. It is possible to employ PI to PI to forward time-series data in other configurations.



Time-Series Data Flows from Protected LAN through Firewall to PI Servers on a Widely Available LAN

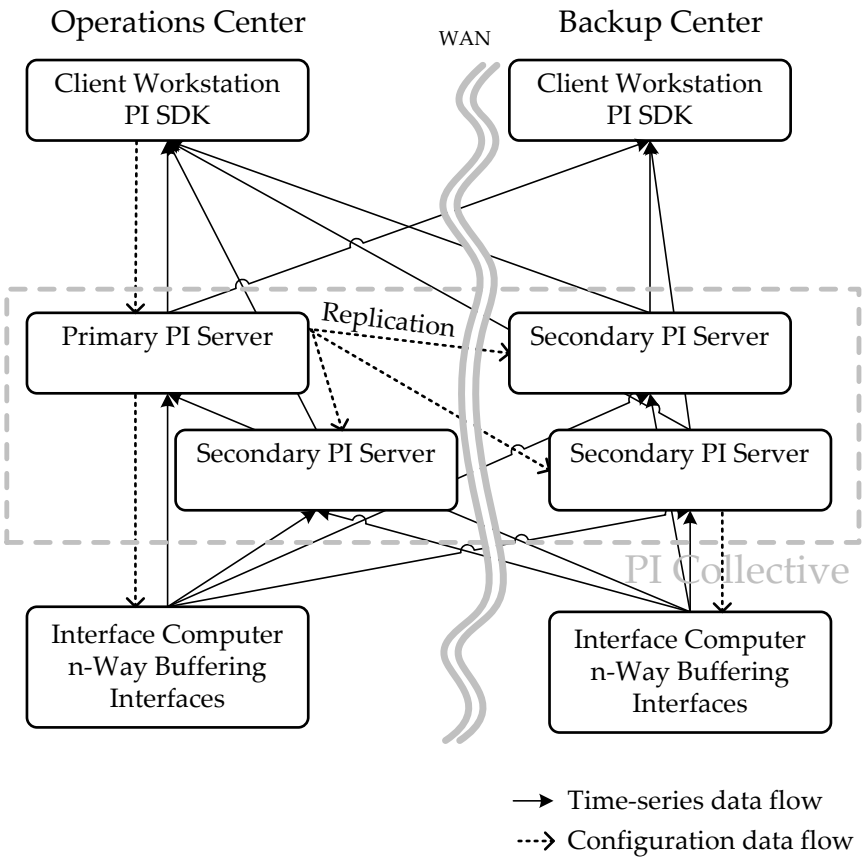
Backup Data Center

The following figure shows a more sophisticated HA deployment pattern that employs replication. This meets the requirements of some industries that operation centers be geographically separated by large distances. In this case there is a primary PI Server and one secondary PI Server locally available in an operations center. There are two secondary servers located remotely in a backup operations center.

Individual workstations are configured to connect preferentially to the local PI Servers before the remote PI Servers. Some workstations can be configured to connect to only the local PI Servers. This provides static load distribution and some separation of functions between the two operations centers.

As with all of the previous cases, the primary PI Server receives all configuration changes. The PI Server Replication moves these changes to all three secondary PI Servers.

There are interfaces in both the operations center and backup center. Each interface sends time-series data to all PI Servers via n-way buffering. Interfaces in the operations center receive configuration information from the primary PI Server. Interfaces in the backup center receive configuration from one of the secondary PI Servers in the backup center.



PI Servers in Geographically Separated Operations Centers

Chapter 2. Replication Basics

PI Server Collectives

The PI Server Collective is a contract between servers that they will keep their configuration databases identical. This ensures that the interface computers can collect time-series data and send it to the servers where it is forwarded through the snapshot into the archives for storage. This also ensures that the client applications can connect to any servers in the collective to receive time-series and historical data.

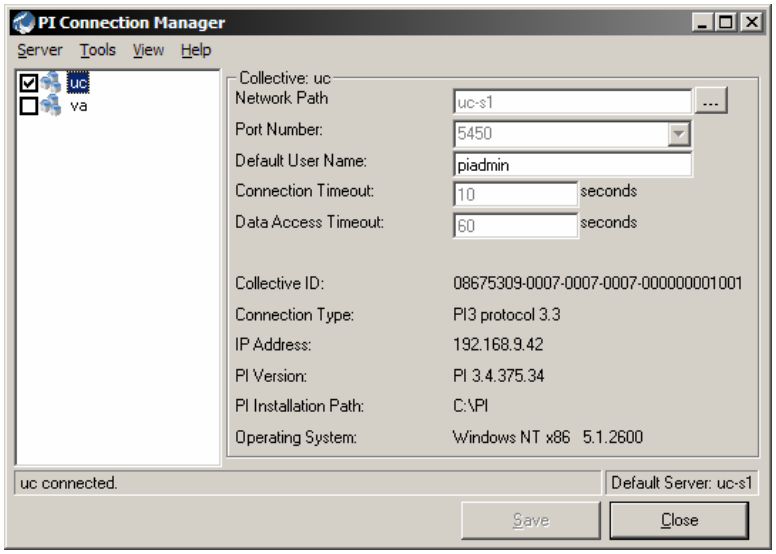
The contract is created by the procedures that establish a collective and initialize the secondary PI Servers with copies of the primary PI Server configuration database. Additional procedures verify operation of the collective to ensure that the contract is followed.

After the PI Servers are initialized with duplicate configuration data, PI Server Replication moves configuration changes between servers in a collective so that the configuration remains consistent between all of the PI Servers. PI Server Replication does not move time-series data between servers in the collective.

PI SDK Server Connection and Automatic Client Failover

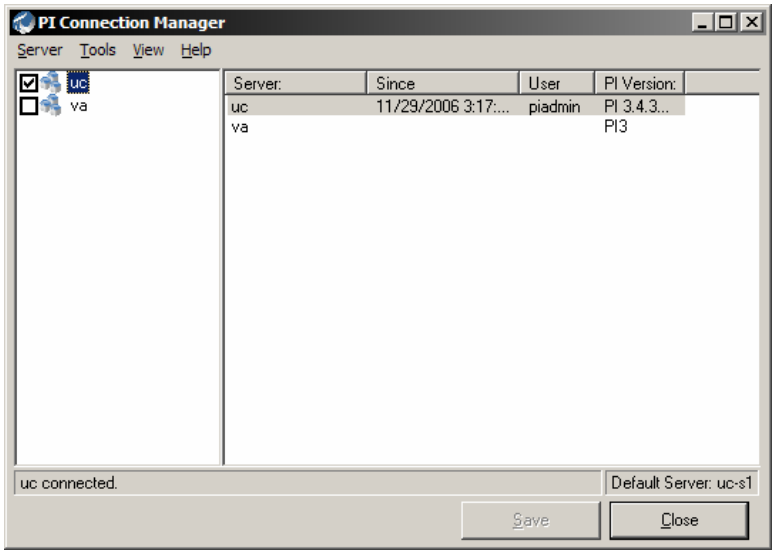
In most circumstances, it does not matter which server in the collective provides data to your application. If the server currently providing data shuts down or times out, you want your application to be connected to an alternative server.

The PI SDK selects the server in the collective that is used to supply data. The **Settings** view of the **PI Connection Manager** dialog box shows which server in the collective is currently supplying data. From the **Server** menu in the dialog box, you can force a failover to the next server in the collective or connect to the primary PI Server.



PI Connection Manager, Settings view

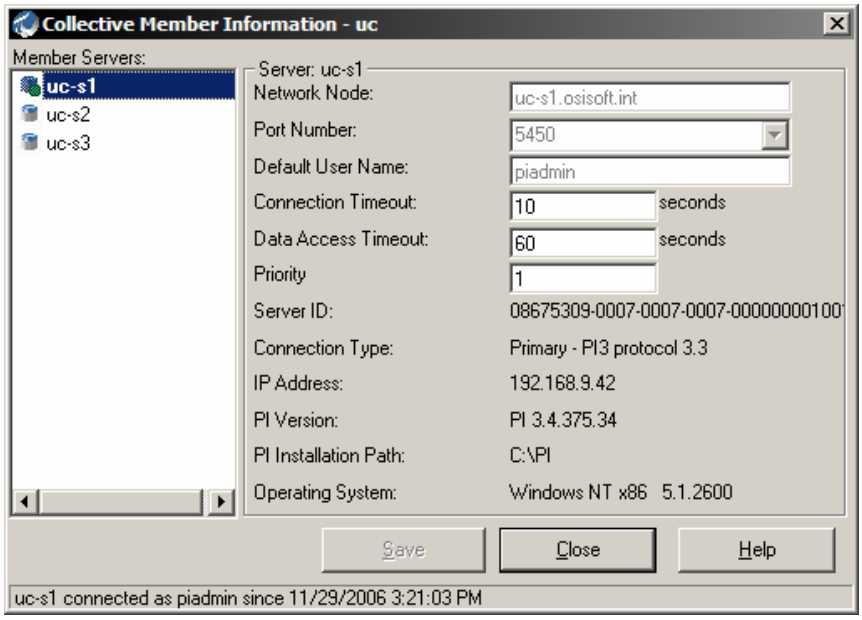
The **Summary** view of the **PI Connection Manager** dialog box displays information about the connected servers and collectives. Each server and collective in the Known Server Table (KST) has an entry that lists the connected user, PI Server version, and the time that the PI SDK connected to the server.



PI Connection Manager Summary

In the PI SDK 1.3.4 or later, priority entries stored in the local KST control the order of server connection in the collective. You can access and edit these entries from the **Collective Member Information** dialog box.

By default, the priority of the primary PI Server is set to 1. The PI SDK attempts to connect to an available server with the lowest value of the priority. The PI SDK never tries to connect to a server with a priority of -1. Changing the values of the priority on different sets of workstations provides static load distribution among the servers.



PI Connection Manager Collective Member Information

PI Server Replication Requirements

Server Computer Requirements

PI Server Replication requires one primary and one or more additional computers to act as secondary PI Servers. The *System Sizing* section in the *PI Server Installation and Upgrade Guide* contains essential information about hardware requirements for PI Servers. The following table lists the requirements for replicated PI Servers. The computers must have adequate resources to work as PI Servers, but do not need to have identical specifications.

Requirements for Replicated PI Servers

Aspect	Description and Details
Operating System	Microsoft Windows Server 2003, Service Pack 1 or Microsoft Windows Server 2000, Service Pack 4. You can use Microsoft Windows XP and Microsoft Windows Vista for testing and evaluation, but do not use them for production environments.
Computer Hardware	Adequate speed, number of processors, and disk capacity for the desired number of archive files
PI Server	Version 3.4.375 or later release
PI SDK	Version 1.3.4 or later release recommended to run SDK-based management tools PI SMT and PI ICU locally
Network	TCP/IP between interfaces and servers, and between servers (PINet protocol uses TCP port 5450 by default).
Initial Bulk Copy	Shared folder, FTP, HTTP, or removable media (CD, DVD, Flash drive, tape) moved between servers

Note: The PI Server can be run in Virtual Machine (VM) environments provided by Microsoft Virtual Server or VMware. Incorrect use of virtualization may result in a single point of failure. If you decide to run the PI Server in a VM, we highly recommend running the servers in a PI Collective on different VM host hardware.

The PI Server computers must be able to communicate with each other using the PINet protocol based on TCP/IP. The network must be configured to allow packets to move reliably between the servers. In many cases, the primary PI Server has been running for some time and is being upgraded to a Collective by adding one or more new secondary PI Servers. This might require configuring the network to allow PINet communication between the primary and secondary PI Servers, especially if the secondary servers are located remotely from the primary server.

Note: Network configuration (switches, routers, gateways) between the replicated servers in a collective will require the coordination of several IT groups in large organizations.

The process of creating a collective requires backing up the primary PI Server and moving the backup files to the computer running the new secondary PI Server. You can do the initial bulk copy of these large files in any way that is convenient. You can employ removable media to move the files in order to conserve bandwidth between the computers.

Management of a collective is easiest when the PI Server is installed in the same way on all the server machines. This includes the paths for the `PI` directory, the `PIPC` directory, the archives, and the queues. Otherwise, PI system administrators are more likely to make errors as they log into the various servers in the collective to perform administrative tasks. Consistent installation is more important for replicated PI Servers than it is for non-replicated PI Servers installed in different environments.

Interface Computer Requirements

The following tables list the requirements for interface computers that use n-way buffering. BufServ is available for the largest number of computer environments, but requires shutting

interfaces down to change configuration. The Buffer Subsystem is available for a smaller number of computer environments, but can accept configuration changes without shutting down, and ensures that the time-series data is compressed identically by all of the PI Servers. For details on Buffer Subsystem features, see *How N-Way Buffering Affects Server Compression* (page 64).

Requirements for Interface Computers with BufServ

Aspect	Description and Details
Operating System	Any that supports PI API 1.6. This includes 32-bit Microsoft Windows for Intel and many Unix variants including HP-UX, HP Tru64 (OSF1), Sun Solaris, IBM AIX, and Linux for Intel.
Buffer	BufServ 1.6 or later
PI API	Version 1.6.0.4 or later
PI SDK	Version 1.3.4 or later release recommended to run PI SDK-based management tools PI SMT and PI ICU locally on Windows computers
Network	PINet 1 (PI API) from interface computers to PI Servers (usually TCP port 5450)

Requirements for Interface Computers with the Buffer Subsystem

Aspect	Description and Details
Operating System	Any of the following: ▪ Microsoft Windows XP Service Pack 2 ▪ Microsoft Windows Server 2003 Service Pack 1 ▪ Microsoft Windows Server 2000 Service Pack 4 ▪ Microsoft Windows 2000 Professional Service Pack 4
Buffer	The PI Buffer Subsystem distributed with PI SDK 1.3.4 or later
PI SDK	Version 1.3.4 or later recommended to run PI SDK-based management tools PI SMT and PI ICU locally
Network	PINet 3 from interface computers to PI Servers (usually TCP port 5450)

Client Computer Requirements

Existing PI SDK applications running on client computers continue to work with the primary PI Server. The PI SDK 1.3.4 enables these existing applications to take advantage of failover behavior. Most Rockwell Automation client applications are enhanced to take advantage of HA features and services. For details, see the *High Availability User Guide* and the *High Availability Advanced User Guide*. Existing PI API applications will still connect directly to named PI Servers. The PI API does not provide failover behavior.

Clients that use Web-based products only, such as RtWebParts, have no specific requirements for HA. Following are the requirements for the most common client applications.

Requirements for Client Computers working with PI Collectives

Aspect	Description and Details
PI SDK	Version 1.3.4 or later required to take advantage of automatic failover
FactoryTalk Historian ProcessBook	PI SDK-based version 3.0 or later required for automatic failover. PI API-based version 2.35 or earlier requires manually selecting alternate collective members.
FactoryTalk Historian DataLink	PI SDK-based version 3 or later release

PI Server Replication Limitations

The PI Server Platform Release 1 has some limitations. Many of these limitations will be addressed in subsequent releases.

PI SDK

No PI SDK N-Way Buffering

The PI SDK does not send data to all members of the collective. Data goes only to the server to which the PI SDK is currently connected. In the default configuration, the PI SDK can post data only to the primary PI Server.

The behavior of PI SDK 1.3.4 or later is configurable with a PITimeOut table parameter. You can decide per secondary server if the PI SDK can post new time-series data. Earlier versions of the PI SDK do not recognize this parameter. For details, see *PITimeOut Table Parameters* (page 62).

Although the PI SDK does not support buffering, custom applications can be written to send time-series data to all the members of a collective. An alternative is to configure PI to PI to move time-series data for selected points from the primary PI Server to secondary PI Servers in the collective.

PI SDK Manual Data Entry

Manual data entry applications or laboratory data entry applications that use the PI SDK to send time-series data to the PI Server inherit the behavior of the PI SDK. This includes applications that are built on the FactoryTalk Historian DataLink 3.x add-in, and applications that use custom VBA in FactoryTalk Historian ProcessBook.

Configuring PI to PI to move time-series data for the points is one possible solution for manual data entry applications or lab data entry applications.

Configuration Changes using the PI SDK

Configuration changes must be directed to the primary PI Server so that they can be replicated to the other servers in the collective.

PI API

PI API N-Way Buffering

BufServ provides n-way buffering to PI Servers that compress time-series data independently of each other. It is available for a wide range of platforms supported by the PI API. The trends of time-series data from all of the PI Servers are within CompDev of the original data. However, the compression running independently on the PI Servers can result in the selection of different time-series values to be stored in the individual archives. For more details, see *How N-Way Buffering Affects Server Compression* (page 64).

The BufServ process reads configuration parameters only on startup, and must start up before all of the interfaces for which it buffers time-series data. If you want a change to the BufServ configuration parameters to take effect, you must stop all of the interfaces on the interface computer, stop buffering, then start buffering, and finally start the interfaces.

The PI Buffer Subsystem provides n-way buffering and compression to PI Servers and provides many enhancements over BufServ. The trends of time-series data from all of the PI Servers are identical except for shutdown events which might be inserted by the individual PI Servers. The Buffer Subsystem has been tested on Microsoft Windows Server 2003 and Microsoft Windows XP operating systems.

The Buffer Subsystem process reads configuration parameters on startup, and is aware of changes to the PIcollective and PIserver tables. When a new server is added to the PIserver table, the PI Buffer Subsystem starts queuing time-series data for the new PI Server.

You can install buffering on PI Servers so that time-series data collected locally is sent to all members of the collective. If you do this, the PI Server software must not be running on that computer when buffering is installed. An important workaround is that the server named *localhost* is omitted from n-way buffering so that local copies of the Random, Ramp Soak, and Performance Equation interfaces continue to send time-series data only to the local PI Server. For more details, see *Upgrade and Configure Interfaces and Buffering on Interface Computers* (page 51).

Note: The Buffer Subsystem 3.4.375 cannot run on a PI Server computer. BufServ 1.6 must be used for n-way buffering among servers.

PI API Manual Data Entry

Manual data entry applications that use the PI API send time-series data to all of the servers only if n-way buffering is configured on the client computer that hosts the application.

PI API Failover to Alternate Server

The PI API does not provide any method for failing over data collection to an alternate server in a collective.

Archive Editing

Time-series event edits, deletes, or additions applied to one server do not automatically propagate to the other servers in the collective.

Archive Annotations

Archive annotation edits, deletes, or additions applied to one server do not automatically propagate to the other servers in the collective.

Batch Database

The batch database configuration is stored in the Module Database. Units are created as modules in the Module Database. Units created on the primary server are moved to all of the secondary servers.

In order to prevent confusion with batch applications, the secondary PI Server default behavior is to reject Batch queries from clients. This can be overridden by a PITimeOut table parameter that enables the Batch queries for cases where customers periodically move a backup of the archives from the primary PI Server to the secondary PI Server. For details, see *PITimeOut Table Parameters* (page 62).

PIBaGen

PIBaGen must be run only on the primary PI Server because the unique archive information that is stored and updated in the archive for each batch cannot be sent to the secondary PI Server. If PIBaGen were run on multiple servers in the collective, each would create a different identifier for the same batch. This would not cause problems for PIBaGen itself, but would cause problems for applications like PI BatchView and RtReports when they fail over from one PI Server to another expecting the same batch identifier.

Batch Subsystem

The batch subsystem runs on all servers in the collective. The batches are generated independently on each server in the collective. However, the batch metadata such as units and aliases are not replicated between the PI Servers.

Alarm Subsystem

The alarm subsystem can be run on all servers in the collective. If the alarm subsystem is run on all of the servers, the PI alarms must be acknowledged at all of the PI Servers.

We recommend running the alarm subsystem on the primary PI Server only. An alternative is that the alarm subsystem be run on all PI Servers, but the alarms are acknowledged only on the primary PI Server.

Performance Equations and Totalizer

Performance equations and Totalizer calculations are computed independently on all servers in the collective. The inputs to the calculations run on the PI Servers are within the exception specifications of each other. The outputs from the calculations are compressed on each of the PI Servers.

It is possible that the results for the same points will be slightly different on each server in a PI Collective. The possible reasons include:

- The time-series data arrives at each server at different times, and the calculation is periodically scheduled according to the clock, not the timestamp of the data.
- The configuration is changed on the primary PI Server. The PI Server applications on each PI Server may receive the changes at different times.
- Internal latency within each PI Server application results in the same calculation being performed at different times.
- Even if the server applications generate identical time-series data, the results may be different due to the PI Server application restart (which may affect the exception reporting results) and the PI Server restart (which may insert Shutdown events or affect the compression results).

COM Connector Points

If COM connector points are used, the COM connector must be configured on all servers in the collective. This may require additional configuration on the system that is providing data so that it accepts connections from the secondary PI Servers as well as the primary PI Server.

PI APS

PI APS can be run on any computer, but must direct configuration changes such as point creation, editing, and deletion to the primary PI Server.

Designating a New Primary Server

Promoting a secondary server to become the new primary server is a manual operation. For details, see *Designating a New Primary PI Server* (page 48).

Load Distribution

This release of the PI Server offers static load distribution only. User workstations can be configured so that the PI SDK connects to PI Servers in a preferential order controlled by a priority entry for each server in a PI Collective. To prevent the PI SDK from connecting to a PI Server, set the priority of that PI Server to -1.

PI ACE Scheduler

The PR1 PI ACE (version 1.2.24/2.1.8) fully supports HA servers by running in redundant mode. You must configure a pair of PI ACE schedulers to run per collective. The computer hosting the PI ACE Scheduler must be set up for n-way buffering of PI API data to all members in the collective and must be different from the computer hosting any of the collective members.

If you modify configurations within your ACE calculations, ensure that you are connecting to the primary PI Server so that the changes can be replicated to the secondary PI Servers. If you bypass the ACE data-sending mechanism (PIACEPoint class) and use the PI SDK to send time-series data directly to the PI Server, you must send the data to all of the servers in the collective.

Interface Output Points

Each interface monitors writes to interface output points on the host PI Server referenced in the interface startup command file. Writes to output points on other servers are ignored. You may first notice the change on a secondary PI Server. This is because the interface can scan the time-series data back, and n-way buffering may send the time-series data to the secondary server before the primary server.

Chapter 3. Replication Operating Procedures

This chapter guides you through the steps required for various operating procedures. Each section describes what must be done, and where appropriate provides an example of the activity. In this chapter most of the activity is described in terms of command-line tools with additional overview of the GUI tools as appropriate.

Collective Formation Overview

Before going any further, install the PI SDK 1.3.4 or later on at least one computer that will be used to manage the servers in the collective. We also recommend that you install the Collective Manager on the same computer. Later you will be guided to install or update the PI SDK, PI Server, and interfaces as necessary.

Following are the steps required to initiate PI Server replication, whether you use the command-line or GUI tools to configure the PI Servers. The steps assume that the PI Server that you plan to promote to a primary server in a PI Collective is already running. Details on each of these items are covered in the following sections.

1. Verify that the Existing PI Server Is Healthy
2. Verify that the Snapshot Event Queue Is Empty
3. Upgrade Existing PI Server Software
4. Install Secondary PI Server Software
5. Select Names and IDs for the Collective and the Secondary PI Servers
6. Force an Archive Shift
7. Upgrade and Configure Interfaces and Buffering on Interface Computers
8. Limit Database Changes on Primary PI Server
9. Verify that the Snapshot Event Queue Is Still Empty
10. Flush the Archive Write Cache
11. Configure PICollective and PIServer Tables
12. Back Up PI Configuration Database and PI Archives
13. Allow Database Changes on Primary PI Server
14. Restore Configuration and Archive Files on Secondary PI Server
15. Start PI on Secondary PI Server
16. Test Replication Behavior

Collective Formation

This section describes how to configure PI Server Replication. You can do this using either the command-line utilities or the Collective Manager tool.

This section assumes that you are starting with an existing PI Server, which becomes the primary server in the collective. If you do not have an existing PI Server, your first installation of a PI Server becomes the primary PI Server in your collective.

If you have been using PI for some time, you probably have a library of FactoryTalk Historian ProcessBook displays that refer to your existing PI Server. You may also have FactoryTalk Historian DataLink spreadsheets, RtPortal Web pages, PI ACE calculations, or custom applications that refer to the existing PI Server. In the following procedures, it is important to select the Collective Name and Collective ID properly so that these displays can be used without modification on the various computers that are used for the client applications.

Note: The Collective Manager selects the proper Collective Name and ID when you choose the **An existing PI Server that contains historical data** option.

If you have been using the PI System for some time, you already have interfaces on one or more interface computers that send time-series data to the PI Server. We recommend setting up and configuring BufServ for n-way buffering so that these interfaces send the data to all of the servers in the collective. This requires shutting down the interfaces for a short time in order to reconfigure BufServ. All of these steps are included in the following procedures.

Example: Initial PI Server Configuration

As an example of the configuration steps that are necessary, this section follows a typical non-replicated PI Server as it is promoted to the primary PI Server in a collective.

The following table shows information about this PI Server. This is the information that you would see on a PI Server upgraded to version 3.4.375 before it is configured to be a member of a collective. For details on how to view this information, see *Example: Newly Upgraded PI Server Configuration* (page 27).

Example Non-Replicated PI Server Configuration

Name	Description	Collective	FQDN	Role	Server ID
uc-s1			uc-s1.osisoft.int	0	08675309-0007-0007-0007-000000001001

The Name `uc-s1` is the hostname of the computer on which the PI Server runs. The FQDN `uc-s1.osisoft.int` is the Fully Qualified Domain Name of that computer. In your configuration, replace these with the name and FQDN of the computer running your PI Server.

The Server ID is the UID assigned to the PI Server. In your configuration, replace this with the Server ID of your PI Server.

The role 0 corresponds to a non-replicated PI Server. The Description and Collective name are blank because they have not been set.

Example: Initial PI Interface Configuration

The following table shows the configuration of interfaces for the example. There are interfaces on one interface computer `uc-i1` in addition to the server `uc-s1`. The interface name is the name of the Windows service under which the interface runs. The Host PI Server corresponds to the `/host=` parameter in the bat file that starts the interface.

The type of interface corresponds to the executable that runs the interface. The Buffering column indicates whether buffering is enabled for this interface. The PointSource parameter, entered as the `ptsrc` parameter in the PI Point database, corresponds to the `/ps=` parameter in the bat file that starts the interface. The loc1 parameter corresponds to the `/id=` parameter, while the Event Counter corresponds to the `/ec=` parameter in the bat files that starts the interface. The event counter is used with the `iorates.dat` file to control summary statistics sent by UniInt interfaces to the PI System.

In this example, the interface computer `uc-i1` has copies of the `random` and `rampsoak` interface with different pointsource than the default `random` and `rampsoak` interface that are run on the server `uc-s1`. These interfaces simulate the collection of data that occurs on real interface computers. The `pipeschd` interface which drives the performance equation calculations is listed here because it is shares many characteristics with interfaces including installation, maintenance, and buffering configuration.

Example PI Interface Configuration

Computer	Interface Name	Host PI Server (/host= parameter)	Type	Buffering	PtSrc (/ps= parameter)	loc1 (/id= parameter)	Event Counter (/ec= parameter)
uc-i1	random90	uc-s1	random	yes	S	n/a	90
uc-i1	rmp_sk91	uc-s1	rampsoak	yes	8	n/a	91
uc-s1	random	localhost	random	no	r	n/a	
uc-s1	rmp_sk	localhost	rampsoak	no	9	n/a	
uc-s1	pipesched	localhost	pipesched	no	C	n/a	

Verify that the Existing PI Server Is Healthy

Before you incorporate n-way buffering and activate the replication service, verify that the server that will be initialized as the primary PI Server is performing normally.

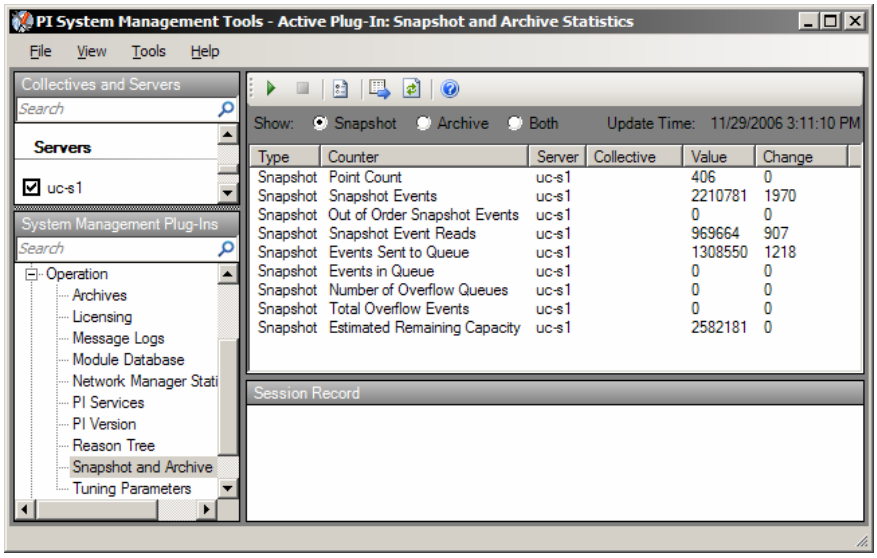
You can use most of the verification steps in *Verifying PI Server Collective Operations* (page 40). The exceptions are that you follow the steps in *Verify N-Way Buffering to PI Server* (page 41) in the context of buffering to a single PI Server. Since there is no secondary PI Server, do not follow the steps in *Configuration Data Changes Move from Primary to Secondary* (page 44) and *Collective PI Servers Have the Same Configuration Data* (page 45). The most important step is to verify that the interfaces are collecting data and sending it to the PI Server archive.

You should develop an inventory of interfaces that send data to the PI System. This is useful when you have to update interface computers to support n-way buffering. The result of the inventory should be a table containing the information in Example PI Interface Configuration

Verify that the Snapshot Event Queue Is Empty

Be sure that the PI Server snapshot event queue is empty. Do this once to make certain that the PI System is performing normally. You will do this again (page 35) just before you back up the files on the primary PI Server. This is important because a standard backup does not include the snapshot event queue. Making certain that the snapshot event queue is empty ensures that the PI archive subsystem has received all of the time-series events queued by the PI snapshot subsystem.

There are several ways to view the state of the snapshot event queue. From the PI SMT, select the server you want in the **Collectives and Servers** pane. Expand the **Operation** list and select the **Snapshot and Archive Statistics** module. In the right pane, select the **Snapshot** button. The parameter **events in queue** must be zero, as shown:



PI SMT Operation - Snapshot Statistics

Alternatively, at a windows command prompt, change to the `PI\adm` directory and enter:

```
piartool -ss
```

The output will look something like this:

```
Counters for 8-Sep-06 11:51:44
                    Point Count:          364       0
                 Snapshot Events:       518157       0
  Out of Order Snapshot Events:            0       0
           Snapshot Event Reads:       154276       0
           Events Sent to Queue:       308873       0
                  Events in Queue:          0       0
     Number of Overflow Queues:            0       0
         Total Overflow Events:            0       0
  Estimated Remaining Capacity:      2590182       0
```

As you monitor the parameter `Events in Queue`, you may see the value grow greater than 0 and then become 0 occasionally. This indicates that the queue is receiving time-series events from the snapshot subsystem, and that the archive subsystem is able to send the data to the archives.

You can also use the Microsoft Windows Performance window to view the **PI Snapshot Subsystem** performance counter **Events in Primary Snapshot**. If you have a Performance Monitor (PerfMon) interface point configured for this performance counter, you can view a trend of that point in FactoryTalk Historian ProcessBook.

Upgrade Existing PI Server Software

Upgrade the existing PI Server software to the latest release that supports replication (PI Server 3.4.375 or later).

After you upgrade the software make certain that the PI System is still performing normally. You can repeat the checks that you performed in *Verify that the Existing PI Server Is Healthy* (page 25) and *Verify that the Snapshot Event Queue Is Empty* (page 26).

If you plan to install interfaces on the PI Servers, install the PI ICU separately. Make sure that PI SDK 1.3.4 or later is installed.

Example: Newly Upgraded PI Server Configuration

After you have upgraded the software on the existing PI Server, change to the `PI\adm` directory and enter:

```
piconfig < pisysdump.dif
```

The file `pisysdump.dif` causes `piconfig` to display the contents of the tables that control collective configuration. Typical output before initializing as the primary PI Server in a collective looks similar to this:

```
Collective Configuration
Name, CollectiveID, Description
------------------------------------------------------------------

Member Server Configuration
Name, IsCurrentServer, ServerID, Collective, Description, FQDN,
Role
------------------------------------------------------------------
uc-s1,1,08675309-0007-0007-0007-000000001001,,UC 2006 Demo Server
1,uc-s1.osisoft.int,0

Collective Status
Name, Status
------------------------------------------------------------------

Member Server Status
Name,IsAvailable,CommStatus,SyncStatus,LastSyncRecordID,LastCommTi
me,LastSyncTime,SyncFailReason,UnavailableReason
------------------------------------------------------------------
uc-s1,1,0,0,0,6-Nov-06 17:17:18,31-Dec-69 16:00:00,,
```

Note that the PI Server name, Server ID and FQDN are the same as listed in Example Non-Replicated PI Server Configuration (page 24).

Install Secondary PI Server Software

Install PI Server software on the new secondary servers. It is good practice to start each secondary PI Server to allow run-once activities to complete and verify that the PI Server runs properly before making it part of a collective. You can also check file and directory access permissions especially where archives will be stored. Once you have done this, shut down each secondary PI Server.

You can test the newly installed PI Server software without affecting the eventual operation of the collective. When the secondary PI Server is initialized from the primary PI Server, the configuration database and archives are copied from the primary PI Server to the secondary PI Server, and all of the changes that you made on the non-replicated PI Server are overwritten.

Computer-Specific License Files

If you have computer-specific license files, you will need a different license file for testing the non-replicated PI Server than you will eventually use with the secondary PI Server. A computer-specific license file for a PI Collective does not allow the secondary PI Server to run as a non-replicated PI Server.

Select Names and IDs for the Collective and the Secondary PI Servers

A collective is identified by a Collective Name and Collective ID. A PI Server has a Server Name (the hostname) and an associated Server ID.

When initializing a non-replicated PI Server to be the primary PI Server in a collective, we highly recommend making the Collective Name the same as the existing PI Server Name and the Collective ID the same as the existing PI Server ID. This allows the PI SDK 1.3.4 or later to present information to client applications in a way that requires no changes to existing ProcessBook displays or datasheets, and allows the PI SDK 1.3.3 or earlier to continue to use the display or datasheet.

Warning: If you choose to make the Collective Name and Collective ID different from the existing PI Server Name and Server ID, you cannot share the displays between clients running the PI SDK 1.3.3 or earlier and the PI SDK 1.3.4 or later. Client applications interacting with PI SDK 1.3.4 prompt you to select a new PI Server as the data source for each display or datasheet. You must either save the display or do this every time you open the display. This must be repeated on each workstation that uses the display, or the saved display must be copied to each workstation that uses the display.

In an organization that has multiple collectives, the Server Names, Server IDs, Collective IDs, and Collective Names must be unique relative to each other. Otherwise, the PI SDK may generate error and warning messages on each client computer where duplicates occur in the KST.

When creating a new collective with new PI Servers, we recommend making the Collective Names different from the PI Server Names, and the Collective IDs different from the Server IDs.

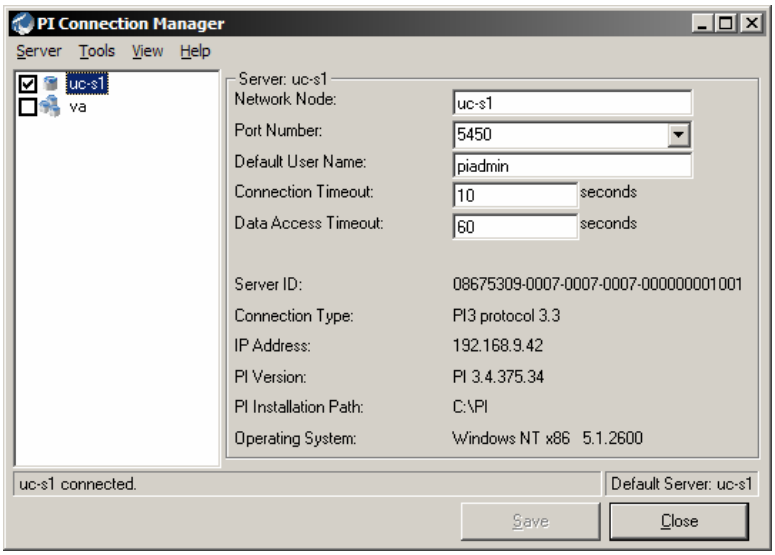
Note: Selecting the Collective Name and Collective ID to be the same as the existing PI Server Name and Server ID will ease your transition from using a non-replicated PI Server to using a PI Collective.

There are several ways to view the existing Server ID (a UID):

1. After upgrading to PR1, run the `piconfig < pisysdump.dif` command shown in *Example: Newly Upgraded PI Server Configuration* (page 27).
2. Launch **AboutPI SDK**.

   Click **Connections**.

   In the **PI Connection Manager** dialog box, select the existing PI Server. The PI Server ID appears in the dialog box:



3. Enter the following command from the `PI\adm` directory:

   ```
   PIDiag -did
   ```

Note: With PI Server 3.4.375 or later, this works only when the PI Base Subsystem is not running. With PI Server 3.4.370 or earlier, this works at any time.

If you do not specify UIDs for the new secondary PI Server or the collective, they are created automatically for the collective and each PI Server when entries are added to the tables.

Example: PI Collective Configuration

The following table shows information about the example collective. For the example shown here, we assume that an existing PI Server is being upgraded. In order that the existing displays will work against any server in the collective, the name of the collective is selected

as the existing PI Server name, and the Collective ID is selected as the Server ID of the existing PI Server that will become primary.

Example Collective Configuration

Name	Description	Collective ID
uc-s1	UC 2006 Demo Collective	08675309-0007-0007-0007-000000001001

The table Example PI Server Configuration in a PI Collective shows information about the example servers in a PI Collective. Each row contains information about one server in the collective. Note that each row has the value of Collective set to the Name in the Collective Configuration table. In this example the collective name is `uc-s1`.

Most of the information about the existing server `uc-s1` has not changed. The description has been entered, but can be left blank. The most important change is that the role has become 1 corresponding to the primary PI Server. The following table shows the possible values of the Role attribute.

PIServer Role Attributes

Role	Meaning
0	Non-Replicated
1	Primary
2	Secondary

There is a new row corresponding to the new secondary server `uc-s2`. The role is 2 for this and all other secondary PI Servers. In this case the name that is the hostname of the secondary PI Server was chosen to be semantically related to the hostname of the primary, but this is not necessary. The FQDN of the secondary PI Server must be entered according to the network environment of the server.

In this example, the Server IDs and Collective IDs have been chosen systematically to make the example easier to follow. The Server ID is always unique relative to the Server ID of all other servers, and the Collective ID is always unique relative to the Collective ID of all other PI Collectives.

Example PI Server Configuration in a PI Collective

Name	Description	Collective	FQDN	Role	Server ID
uc-s1	UC 2006 Demo Server 1	uc-s1	uc-s1.osisoft.int	1	08675309-0007-0007-0007-000000001001
uc-s2	UC 2006 Demo Server 2	uc-s1	uc-s2.osisoft.int	2	08675309-0007-0007-0007-000000001002

Creating UIDs Manually

This section discusses options that most people will not need.

If you do not want to use the default UIDs created automatically for the collective and each server, you can use `pidiag -uid` to generate the UIDs. From the `PI\adm` directory, enter:

```
PIDiag -uuid n
```

where *n* is the number of UIDs that you want to create. You can add these to a text file which you use to configure the collective and servers.

Force an Archive Shift

This is an optional step.

Before making changes that start the n-way buffering and the replication service, it is advantageous to force an archive shift on the primary PI Server. The new primary archive will be fairly empty, will not shift for some time, and will have a start time roughly coinciding with buffering to the new server in the PI Collective.

This step is most important to do this if the primary archive is very full and will shift soon. This step is less important to do if the primary archive is fairly empty and will not shift for some time.

When the newly shifted archive is duplicated and moved to the secondary PI Server, the new archive receives all of the duplicate time-series events that might occur. These duplicate events are due to the secondary server being initialized by a backup of the primary server. The time at which that backup is taken is typically later than the time at which n-way buffering starts on interface computers. Once restored on the secondary server, archive files are likely to contain some overlapping time-series data with the Interface buffers, resulting in a small number of duplicate event errors (-109) in the PI Server message log.

In order to force an archive shift, enter the following command from the `PI\adm` directory:

```
piartool -fs
```

Upgrade and Configure Interfaces and Buffering on Interface Computers

Configuring interfaces for a PI collective requires some additional decisions that were not necessary for non-replicated PI Servers. Consider all the servers in the collective when selecting values for the `pointsource` and `location1` parameters for interfaces that run on the interface computers and interfaces that run on the PI Servers.

This is the time to plan upgrading your interfaces and BufServ to the latest version. Interfaces must be compatible with the versions of BufServ, Buffer Subsystem, PI API, and PI SDK listed in the tables Requirements for Interface Computers with BufServ (page 17) and Requirements for Interface Computers with the Buffer Subsystem (page 17) to support PI Server Replication. Decide if you want to employ *UniInt Interface Level Failover* (page 33).

Make these decisions before starting to change configuration. Most changes require stopping the interfaces for some time, so you must schedule time for data collection to be out of operation.

Choosing the PI Interface Home Node

Decide which PI Server provides configuration data to the interfaces, as well as output points if the interface supports outputs from the PI Server. Each interface receives configuration data from only one server in the PI Collective. This is designated as the `/host=` parameter in the interface startup command file.

Buffering on PI Interface Computers

Buffering is provided by the BufServ process, which is documented in the PI SDK help file `PIPC\HELP\pisdk.chm`. To access this help from the About PI SDK window, choose **Help > PI SDK Help**.

Configure BufServ on each interface computer to support n-way buffering to each of the servers in the collective. To do this, edit the `piclient.ini` file, or use PI SMT or PI ICU.

For BufServ, all of the servers in the collective must be in the buffered server list and in the replicated server list, so the interfaces send time-series data to all members of the server. The PI Buffer Subsystem requires only that the PI Servers be in the buffered server list. Changes to `piclient.ini` take effect only when BufServ starts, while The PI Buffer Subsystem reads the file periodically.

Note: BufServ must start *before* all interfaces for buffering to work properly.

Once configuration changes are complete, stop all interfaces on the interface computer, stop BufServ, restart BufServ, and then start the interfaces. Do this as quickly as possible, because many interfaces cannot retrieve history from the external device to avoid data loss corresponding to the time that they were not running.

Verify that interface buffering is working properly (for example, using `bufutil.exe`). For details, see the verification steps in the PI SDK help. For more details on these steps, see *Verify N-Way Buffering to PI Servers* (page 41).

The most common issues with BufServ include:

Issue	Solution
BufServ starts after one or more interfaces start	Stop all interfaces, stop BufServ; then start BufServ followed by the interfaces
The `piclient.ini` or `pilogin.ini` files are not configured properly	Correct the configuration. In order to have the new configuration read, stop all interfaces and BufServ; then start BufServ followed by the interfaces.
The `/host=` parameter in the interface startup command file is not set properly	Correct the startup command file parameter, stop the interface and then restart the interface

Set Up Windows Service Dependencies

On interface computers running Microsoft Windows, configure the interface service to depend on BufServ or pibufss so that BufServ or pibufss cannot be stopped before interfaces and the interfaces cannot be started before BufServ or pibufss. In Microsoft Windows, you can view dependencies using the **Services** Control Panel. You can change dependencies only when the interface is installed as a service, by using the PI ICU or the **sc** command.

For more information about specifying the dependencies when you install the server, enter the interface executable at the command line with the **`-?`** parameter. The PI ICU run locally on the computer can also be used to specify the dependencies for the interfaces services.

In Microsoft Windows Server 2003 or Microsoft Windows XP, the **sc** command lists and configures services. To view the dependencies of an interface, enter at the command prompt:

```
sc \\Server qc InterfaceService
```

where *Server* is the optional server name and *InterfaceService* is the name of the interface service.

For example, typical output from the **sc** command looks like:

```
C:\Program Files\Rockwell Software\FactoryTalk
Historian\Server\PI\adm>sc \\uc-i1 qc random90

[SC] GetServiceConfig SUCCESS

SERVICE_NAME: random90
        TYPE               : 10  WIN32_OWN_PROCESS
        START_TYPE         : 2   AUTO_START
        ERROR_CONTROL      : 1   NORMAL
        BINARY_PATH_NAME   : C:\Program Files\Rockwell
Software\FactoryTalk
Historian\Server\PIPC\InterfacesRandom\random.exe 90
        LOAD_ORDER_GROUP   :
        TAG                : 0
        DISPLAY_NAME       : PI-random90
        DEPENDENCIES       : tcpip
                           : bufserv
        SERVICE_START_NAME : LocalSystem
```

This shows that the service `random90` running on server `uc-i1` has dependencies on `tcpip` and `bufserv`.

To add a dependency on the bufserv and tcpip services, use the **sc** command with the **config** option:

```
sc \\Server config InterfaceService depend= bufserv/tcpip
```

Note: The space between `depend=` and the names of the services is required.

For more information about the **sc** command, enter at the command prompt:

```
sc -?
```

UniInt Interface Level Failover

Configure UniInt interface level failover at this time. UniInt failover involves installing redundant copies of an interface on separate interface computers to provide uninterrupted data collection even when one of the interfaces fails to collect data for any reason. When maintenance, hardware failure, or network failure causes one interface to become unavailable, the redundant interface automatically collects and sends data to the PI Server. For more details, see *Interface Level Failover* in the *UniInt End User Guide*.

Example: PI Interface Configuration for the PI Collective

The table Example PI Interface Configuration (page 25) lists the interfaces that run on the interface computer and the PI Server before it is promoted to the primary PI Server in the collective. The following table lists the interface configuration on the interface computer and the servers in the collective that must be set up for the PI Server Collective. The interfaces on the interface computer are set up for n-way buffering to all servers in the collective, while the local interfaces on the PI Servers are not set up to buffer.

Example Interface Configuration for a Collective

Computer	Interface Name	Host PI Server (/host= parameter)	Type	Buffering	PtSrc (/ps= parameter)	loc1 (/id= parameter)	Event Counter (/ec= parameter)
uc-i1	random90	uc-s1	random	n-way	S	n/a	90
uc-i1	rmp_sk91	uc-s1	rampsoak	n-way	8	n/a	91
uc-s1	random	localhost	random	no	r	n/a	
uc-s1	rmp_sk	localhost	rampsoak	no	9	n/a	
uc-s1	pipesched	localhost	pipesched	no	C	n/a	
uc-s2	random	localhost	random	no	r	n/a	
uc-s2	rmp_sk	localhost	rampsoak	no	9	n/a	
uc-s2	pipesched	localhost	pipesched	no	C	n/a	

In order to set up interfaces to send time-series data via n-way buffering to the servers in the collective, the `piclient.ini` or `pilogin.ini` files must be changed. They can be edited with a text editor or with the PI ICU.

Following is an example of the random90 interface startup command file named `random90.bat` installed in `C:\Program Files\Rockwell Software\FactoryTalk Historian\Server\PIPC\Interfaces\Random` on interface computer `uc-i1`:

```
"C:\Program Files\Rockwell Software\FactoryTalk
Historian\Server\PIPC\Interfaces\Random\random.exe" 90 /PS=S
/ID=90 /host=uc-s1:5450 /ec=90 /stopstat /maxstoptime=120
/pisdktimeout=60 /sio /perf=8 /pisdkConTimeout=15 /f=00:00:30
/f=00:01:00
```

Following is an example of the rmp_sk91 interface startup command file named `rmp_sk91.bat` installed in `C:\Program Files\Rockwell Software\FactoryTalk Historian\Server\PIPC\Interfaces\rmp_sk` on interface computer `uc-i1`:

```
"C:\Program Files\Rockwell Software\FactoryTalk
Historian\Server\PIPC\Interfaces\rmp_sk\rmp_sk.exe" 91 /PS=8
/ID=91 /host=uc-s1:5450 /ec=91 /stopstat /maxstoptime=120
/pisdktimeout=60 /sio /perf=8 /pisdkConTimeout=15 /f=00:00:30
/f=00:01:00
```

Note that both of these interface startup command files specify `host=uc-s1:5450`.

If you are using **pibufss**, following is the `piclient.ini` file in `C:\Program Files\Rockwell Software\FactoryTalk Historian\Server\PIPC\dat` on interface computer `uc-i1`:

```
[APIBUFFER]
BUFFERING=1
[BUFFEREDSERVERLIST]
BUFSERV1=uc-s1
```

If you are using BufServ 1.6, following is the `piclient.ini` file in `C:\Program Files\Rockwell Software\FactoryTalk Historian\Server\PIPC\dat` on interface computer `uc-i1`:

```
[APIBUFFER]
BUFFERING=1
[BUFFEREDSERVERLIST]
BUFSERV1=uc-s1
```

```
BUFSERV2=uc-s2
[REPLICATEDSERVERLIST]
REPSERV1=uc-s1
REPSERV2=uc-s2
```

Note that both of the servers in the collective are in the `BufferedServerList` and the `ReplicatedServerList`. This file provides for a buffer file size of 10,000 KB. The size of the memory buffers is not specified, so it defaults to 32768 bytes.

Following is the `pilogin.ini` file in `C:\Program Files\Rockwell Software\FactoryTalk Historian\Server\PIPC\dat` on interface computer `uc-i1`:

```
[Defaults]
HELPFILE=C:\Program Files\Rockwell Software\FactoryTalk
Historian\Server\PIPC\HELP\pilogin.hlp
PIServer=uc-s1
PI1USER=piadmin
PI2USER=piadmin
[Services]
PI1=PI
[PINODEIDENTIFIERS]
PI1=uc-s1,56779,5450
PI2=uc-s2,56780,5450
```

Limit Database Changes on Primary PI Server

Limit configuration changes (such as point creation, point editing, point deletion, and MDB changes) between the time that the collective is configured and the time the configuration database is copied. In small organizations with one or a few people authorized to make changes, you can do this with a procedure that informs people not to make changes.

Alternatively, to place the PI Server in standalone mode, enter the following command locally from the `PI\adm` directory:

```
piartool -standalone on
```

Placing a PI Server in standalone mode has high impact on the PI Server. In standalone mode, the PI Server closes all existing connections from clients and interfaces, and prevents new connections. While the PI Server is in standalone mode, GUI tools such as PI SMT do not work locally or remotely. The PI Server is essentially shut down. The command based tools work locally but not from a remote server. Restarting the PI Server does not bring it out of standalone mode.

You can query the standalone mode with the following command entered locally from the `PI\adm` directory:

```
piartool -standalone query
```

Verify that the Snapshot Event Queue Is Still Empty

Repeat the check in *Verify that the Snapshot Event Queue Is Empty* (page 26). Doing this ensures that the PI archive subsystem has received all of the time-series events queued by the PI snapshot subsystem.

Comment [PHG2]: This feature not available for PR1. When it does become available, this will be the correct place in the procedure to do this.
Eliminate reference to this section from the list of tasks to perform in the overview.
Hide this section.

Flush the Archive Write Cache

Normally, the PI System flushes the archive cache periodically. If there has been a significant amount of time since the archive cache was flushed, there could be data in memory that has not yet been committed to disk.

Flush the archive cache to make sure all of the archive data is written to disk. Enter the following commands from the `PI\adm` directory:

```
piartool –flush –start
piartool -flush -stop
```

Configure PICollective and PIServer Tables

Configure the PISys, PICollective table to contain the Collective Name of the new collective, and the associated Collective ID (a UID). Then configure the PISys, PIServer table to have the names of the PI Servers, their Server IDs, host names, FQDNs, Collective Name, and roles as primary or secondary. Once a PI Server detects it has the role of primary, it starts recording changes made to database tables into a persistent queue for each of the secondary servers.

To make entries in the table, use the Collective Manager or the piconfig command-line tool. The command-line tool has the advantage that the tedious task of entering the Collective ID or the primary PI Server ID can be done by editing text in a file.

Example: Setting Up Replication Using Command-Line Tools

This example shows how to enter the values listed in the tables Example Collective Configuration (page 30) and Example PI Server Configuration in a PI Collective (page 30) into the PI configuration database.

You can make or change entries in the PICollective and PIServer tables with piconfig commands. In order to configure your PI Server, we recommend that you copy the following text to a text file (such as `collective_create_uc.txt`) in the `PI\adm` directory:

```
*    Collective information
*
@tabl pisys,picollective
@mode create,t
@istr name, Description, CollectiveID
uc-s1,UC 2006 Demo Collective,08675309-0007-0007-0007-000000001001
*
*    Individual server member information
*
*    valid values for Role include:
*    0    NotReplicated
*    1    Primary
*    2    Secondary
*
@tabl pisys,piserver
@mode create,t
@istr name,Description,Collective,FQDN,Role,ServerID
uc-s1,UC 2006 Demo Server 1,uc-s1,uc-s1.osisoft.int,1,08675309-
0007-0007-0007-000000001001
uc-s2,UC 2006 Demo Server 2,uc-s1,uc-s2.osisoft.int,2,08675309-
0007-0007-0007-000000001002
```

This file has PI Database configuration entries corresponding to the tables Example Collective Configuration (page 30) and Example PI Server Configuration in a PI Collective (page 30). Edit the text file to contain the Collective name, Collective ID, and description; and the PI Server Name, Description, Collective Name, FQDN, Role, and Server ID of your collective. When you have edited the file the way that you want, enter the following command from the `PI\adm` directory:

```
piconfig < collective_create_uc.txt
```

There are other parameters in the PICollective and PIServer tables that can be changed for special situations. Table PICollective Attributes (page 58) and Table PIServer Attributes (page 58) show definitions of all of the configurable and dynamic PICollective table fields and PIServer table fields.

Setting Up Replication Using GUI Tools

The Collective Manager can be used to verify the configuration of the collective. You can also use this tool to check the status of the collective. See the Collective Manager help for details.

Restrictions to the PIServer and PICollective Tables

The following edits to the PIServer and PICollective tables are not allowed:

- Renaming and removing the current server
- Renaming and removing a server marked as available. This prevents removing a secondary server while it is online.
- Removing the current collective from the PICollective table while the server is marked available
- Promoting another server to primary on the primary server while the primary is available
- Removing the primary PI Server from a collective while the primary is available

Back Up PI Configuration Database and PI Archives

Make a copy of the PI Server database using the `PIBackup.bat` command file. You can enter the command with options to specify the number of archive files to back up; otherwise it selects three archives. On Windows 2003 Server and Windows XP Pro this uses `ntbackup.exe` based on VSS to initiate a copy of the configuration files and one or more archives into a compressed backup `.BKF` file.

To back up the PI Server database, enter the following command from the `PI\adm` directory:

```
pibackup c:\temp\pibackup 9999 "01-jan-70"
```

In this example, the backup of the PI Server database together with up to 9999 archives is stored in the `C:\temp\pibackup` directory. Note that this does not back up empty archives.

When using Windows Server 2003, the backup subsystem takes advantage of VSS so you should see a message in the PI message log that looks like:

```
0 pibackup 22-Nov-06 14:38:08
 >> VSS backup requested
```

When using a windows operating system other than Windows Server 2003, you should see am message in the PI message log that looks like:

```
0 pibackup 22-Nov-06 14:37:56
 >> Non-VSS backup requested
0 pibackup 22-Nov-06 14:37:56
 >> Backup operation started. Backup type INCREMENTAL, NON-VSS
```

While the backup completes, you can continue with other activities. If the backup is successful, you will see a message in the message log that looks like:

```
0 pibackup 22-Nov-06 14:38:41
 >> Backup operation completed successfully
```

See *Backing up the PI Server* in the *PI Server System Management Guide* for details about the `PIBackup.bat` command file. The command file prints a short usage summary if no arguments are supplied.

Allow Database Changes on Primary PI Server

Comment [PHG3]: This feature not available for PR1. When it does become available, this will be the correct place in the procedure to do this.
Eliminate reference to this section from the list of tasks to perform in the overview.
Hide this section.

If the PI Server is in standalone mode, you can take it out of standalone mode with the command:

```
piartool -standalone off
```

At this point, if you used a procedure to restrict changes on the system, you can tell users that they are free to make changes.

Restore Configuration and Archive Files on Secondary PI Server

Once the backup finishes, move the backup files to the secondary PI Server. If you used the backup subsystem to initiate the backup, you may want to compress the backup directory to reduce transfer time to the new PI Server computer. You must move the files to the correct places on the secondary PI Server while the PI Server software on the secondary PI Server is not running.

You can use the windows GUI to move the files to the correct directories. Alternatively, you can use a command file to move the files. You must recreate any empty archives that were not backed up and moved.

If the secondary PI Server has a different directory structure than the primary PI Server, you may have to edit the `pisubsys.cfg` file. If the secondary PI Server stores the primary archive in a different directory than the primary PI Server, you must register the primary archive in the new location.

At this time, make sure the new secondary PI Server has the correct license file. The license file should have been copied from the primary PI Server to the backup directory and from the backup directory to the secondary PI Server. This is especially important if you have a computer-specific license files for the PI collective and used a different license file for testing the computer as a non-replicated PI Server.

Computer-Specific License Files

If you have computer-specific license files for your PI collective, you may need to perform some extra actions. If the computer had previously run as a non-replicated PI Server during

testing, the PI System will not start as a secondary PI Server with a valid computer-specific license file for the PI collective until you change a registry key and delete a file.

To change the registry key, use the following to create a text file with a `.reg` extension:

Contents of file PI_Set_ServerRole_2.reg:

```
Windows Registry Editor Version 5.00

[HKEY_LOCAL_MACHINE\SOFTWARE\PISystem\PI]
"ServerRole"="2"
```

Double-click the file to invoke `regedit32.exe` and click **Yes** in the resulting dialog box. This sets the registry key used by the license manager so the base subsystem can start and read the configuration database.

You must also delete the `pilicense.ud` file from the `PI\dat` directory. Enter the following command from the `PI\adm` directory:

```
del ..\dat\pilicense.ud
```

This allows the license manager to immediately use the new computer-specific license file for the PI collective.

Start PI on Secondary PI Server

Once the files are in the correct place on the secondary PI Server, you can start the new secondary PI Server. When a secondary PI Server starts, it attempts to connect to the primary PI Server to report its status, receive the status of the other servers in the collective, and retrieve configuration changes that were made on the primary PI Server.

If the secondary PI Server stores the archives in different directories than the primary PI Server, you must register the archives.

If the secondary PI Server needs different time-out table or fire-wall table parameters that the primary PI Server, you can change them now.

Test Replication Behavior

To verify that the servers in the collective are behaving normally, follow the steps in *Verifying PI Server Collective Operations* (page 40).

The most important checks are that interfaces are able to collect time-series data and that interface buffering sends time-series data to all servers in the collective. This verifies that the configuration changes for n-way Buffering have not caused any time-series data loss.

To keep configuration consistent on the servers, it is important that PI Server replication moves configuration changes made on the primary PI Server to all secondary PI Servers in the collective. Test the movement of configuration data by making a trivial change to a test point on the primary PI Server and verifying that the change is moved to the secondary PI Server.

Verifying PI Server Collective Operations

When a new collective is formed or you suspect trouble during normal operations, you must verify that the PI Systems are performing as a PI collective. This section provides a list of steps that ensure that the PI collective is working correctly.

Even though a PI Server collective provides alternate servers for the time-series data, you should periodically check to make sure the PI Servers are working properly. The clients using the PI SDK 1.3.4 or later automatically failover to an alternate server, and users might not notice that any one server has failed. If you do not periodically check the health of the servers in the collective, you may only receive feedback from end-users after all servers fail.

Verification Principles

You must trace propagation of two types of data to verify operation of non-replicated PI Servers. Time-series data propagates from the data source through the interfaces, buffering, the PI Server, and to the clients. Configuration data propagates from the client application to the PI Server and then to the interfaces.

You must trace the two types of data over additional paths to verify operation of Replicated PI Servers. In replicated PI Servers, time-series data propagates from the data source through n-way buffering to the servers in the collective. Each of the paths from the data source through the interface to n-way buffering to the associated server must be verified separately. Configuration data still propagates from the client application to the PI Server and the interface, but also is retrieved by the secondary PI Server, which replicates the configuration data and forwards it to interfaces that specify it in the interface startup **/host** parameter. The replication of configuration data from the primary PI Server to the secondary PI Servers must be verified.

Replicated PI Servers keep track of some information about the status of the other members of the collective. Periodically, the secondary PI Servers send their status to the primary PI Server and receive the status of all of the servers in the collective. The result of this status exchange is available in the PIserver and PIcollective tables viewed using **PIConfig** or the Collective Manager.

The following sections detail the steps to verify that each part of the collective behaves as intended.

Interface Data Collection from Data Source

Do not overlook the first link of collecting time-series data from the data source (such as DCS, PLC, MMS, EMS, and network device). If the interface does not collect data from the source, the data can never be sent to the PI System.

Examine messages stored in the `pipc\pipc.log` file to make sure the interface starts properly. Some interfaces provide performance counters that record the number of values that have been retrieved from a data source. In many cases, it is easiest to infer that data collection is occurring by monitoring buffering.

See *Managing Interfaces* in the *PI Server System Management Guide* for more information.

Verify N-Way Buffering to PI Servers

Verify n-way buffering is working for data sent from each interface to each server. Perform these checks also on the interfaces such as PIPerfMon running on PI Servers.

Examine messages stored in the `pipc\pipc.log` file on each computer that hosts an interface, and on the PI Servers that are running interfaces such as PerfMon or PerfMon_Basic. Make certain that the buffering starts properly and is able to connect to each server.

The PI Buffer Subsystem

The PI Buffer Subsystem is closely integrated with the PI3 Server. It receives information from the PI Server about the PIcollective and PIserver tables and uses this information to control its behavior.

To monitor the activity of the Buffer Subsystem, use the command **pibufss –ss** or Performance Counters

For details on the PI Buffer Subsystem, see the PI Buffer Subsystem help in `PIHOME\Help\pibufss.chm` (typically `C:\Program Files\Rockwell Software\FactoryTalk Historian\Server\PIPC\HELP`).

BufServ

BufServ primarily employs command-line tools to verify the status of buffering. This is a result of the necessity of deploying on a diverse set of environments.

Run `bufutil.exe` from each computer that hosts an interface for collecting data. Using the `bufutil.exe` choice **`1, 10, 1`** allows monitoring the primary buffer for 10 times at one-second intervals.

The write location, read locations, and unprocessed entries change as data flows from the interface to BufServ and from BufServ to the server. As you monitor the parameter `Unprocessed entries`, you should see the value grow greater than 0 and then become 0 occasionally. This indicates that the buffering service is receiving time-series events from an interface, and that it is able to send the data to the selected destination PI Server. Repeat this after using `bufutil.exe` choice **`6`** to select each destination PI Server in turn.

The following shows a command-line session with `bufutil.exe`. Extraneous and repeated lines are omitted. In this example, buffering to PI Server **uc-s1** has a backlog that exceeds the memory buffers and extends onto disk, and buffering to PI Server **uc-s2** is current, so PI Server **uc-s1** has not received all of the data that is already on PI Server **uc-s2**:

```
C:\Program Files\Rockwell Software\FactoryTalk
Historian\Server\PIPC\bin>bufutil
Connected to server: UC-S1
Choices 1,2,3 may take optional arguments of [ #repeats #seconds]
( 1) Show Primary buffer header
( 2) Show Secondary buffer header
( 3) Show file buffer header
( 4) Kill server and quit
( 5) Quit
( 6) Change server
( 7) Display this menu
        Enter choice: 1
   Primary Buffer Header Data
```

```
-------------------------------------
  Version:                1
  Mode:                   Dual Memory and File
  Server status:          Disconnected
-------------------------------------
  Size:                   32768
  Next write location:    36
  Next read location:     36
  Write ptr before wrap: 32760
  Unprocessed entries:    1323
-------------------------------------

        Enter choice: 2
   Secondary Buffer Header Data
-------------------------------------
  Version:                1
  Mode:                   Dual Memory and File
  Server status:          Disconnected
-------------------------------------
  Size:                   32768
  Next write location:    24255
  Next read location:     36
  Write ptr before wrap: 0
  Unprocessed entries:    897
-------------------------------------

        Enter choice: 3
      File Header Data
-------------------------------------
Version:               2
Count  :               86052
Current read position: 16
-------------------------------------

        Enter choice: 6
Enter server name: uc-s2
        Enter choice: 1
   Primary Buffer Header Data
-------------------------------------
  Version:                1
  Mode:                   Single
  Server status:          Connected
-------------------------------------
  Size:                   32768
  Next write location:    36
  Next read location:     36
  Write ptr before wrap: 0
  Unprocessed entries:    0
-------------------------------------
        Enter choice: 5
```

For details on BufServ, see the API Buffering help in `PIHOME\Help\pisdk.chm`.

Data Received at PI Server

Verify that time-series data is received at the server computers. Use PI SMT on each server to verify that time-series data for a small set of points from each interface arrives at the PI Server. An alternative is to use `APISnap.exe` on each server to verify that some points from each interface are receiving time-series data. Also verify that once on the PI Server, the time-series data flows from the snapshot to the archive subsystem.

PI Server Collective Status

Verify that the PISys, PICollective and PISys, PIServer tables show similar results on each server. You can use the Collective Manager. With the servers in the collective selected in the PI Servers window, compare the parameters on each server. The attributes listed as editable in Table PICollective Attributes (page 58) and Table PIServer Attributes (page 58) are the same when replication is working. The PIServer attributes LastCommTime, LastGoodSyncTime, LastRecordID, and LastSyncTime must be the same as well.

You can also use piconfig to dump the tables. Enter the following command from the `PI\adm` directory:

```
piconfig < pisysdump.dif
```

The output for the primary server will look similar to this:

```
Collective Configuration
Name, CollectiveID, Description
-----------------------------------------------------------------
uc-s1,08675309-0007-0007-0007-000000001001,UC 2006 Demo Collective

Member Server Configuration
Name, IsCurrentServer, ServerID, Collective, Description, FQDN,
Role
-----------------------------------------------------------------
uc-s1,1,08675309-0007-0007-0007-000000001001,uc-s1,UC 2006 Demo
Server 1,uc-s1.osisoft.int,1
uc-s2,0,08675309-0007-0007-0007-000000001002,uc-s1,UC 2006 Demo
Server 2,uc-s2.osisoft.int,2

Collective Status
Name, Status
-----------------------------------------------------------------
uc-s1,0

Member Server Status
Name,IsAvailable,CommStatus,SyncStatus,LastSyncRecordID,LastCommTi
me,LastSyncTime,SyncFailReason,UnavailableReason
-----------------------------------------------------------------
uc-s1,1,0,0,468,6-Nov-06 17:17:18,25-Oct-06 17:33:42,,
uc-s2,1,0,0,468,6-Nov-06 17:17:14,6-Nov-06 17:17:04,,
```

The output for the secondary server will look similar to this:

```
Collective Configuration
Name, CollectiveID, Description
-----------------------------------------------------------------
uc-i1,08675309-0007-0007-0007-000000001001,UC 2006 Demo Collective

Member Server Configuration
Name, IsCurrentServer, ServerID, Collective, Description, FQDN,
Role
-----------------------------------------------------------------
uc-s1,0,08675309-0007-0007-0007-000000001001,uc-i1,UC 2006 Demo
Server 1,uc-s1.osisoft.int,1
uc-s2,1,08675309-0007-0007-0007-000000001002,uc-i1,UC 2006 Demo
Server 2,uc-s2.osisoft.int,2

Collective Status
Name, Status
-----------------------------------------------------------------
uc,0
```

```
Member Server Status
Name,IsAvailable,CommStatus,SyncStatus,LastSyncRecordID,LastCommTi
me,LastSyncTime,SyncFailReason,UnavailableReason
-------------------------------------------------------------
uc-s1,1,0,0,468,6-Nov-06 17:23:08,25-Oct-06 17:33:42,,
uc-s2,1,0,0,468,6-Nov-06 17:23:14,6-Nov-06 17:23:04,,
```

In this example, server **uc-s1** is the primary (Role is 1), while server **uc-s2** is the secondary (Role is 2). In the primary server output, the `IsCurrentServer` field is set to 1 for **uc-s1** and 0 for **uc-s2**. In the secondary server output, the values are reversed (**uc-s2** is the current server).

For more details on interpreting these attributes, see *Server Tables Supporting Replication* (page 57).

You can also compare the state of the servers in the collective with the Collective Manager.

Configuration Data Changes Move from Primary to Secondary

Verify that configuration data moves from the primary PI Server to the secondary PI Server.

The PIServer parameter SyncPeriod measured in seconds controls the synchronization schedule. The secondary server retrieves pending changes every SyncPeriod seconds. If this parameter is zero, the synchronization occurs every time you use the **Force Sync** command on the Collective Manager (right-click a secondary server and choose **Force Sync**). Alternatively, from the command prompt on the secondary server, enter the following from the `PI\adm` directory:

```
piartool -sys –sync
```

The easiest way to check that replication is in place is to make a configuration change to a point on the primary server using PI SMT, TagConfigurator Excel add-in, or PIConfig. Then monitor the propagation of the change to the secondary servers using any tool that shows point attributes.

If you do not see a change to the secondary servers right away, use the `pilistupd` utility to verify the pending changes. Enter the following command from the `PI\adm` directory on the primary PI Server:

```
pilistupd
```

The output is similar to this:

```
Producer          Consumer          Qual.   Flags   Pending
---------         ---------         ------  ------  --------
PIChangeRecordUpdates    uc-s2*            0       0       0
snapshots         pialarm|68      483     0       0
snapshots         pialarm|68      484     0       0
snapshots         pialarm|68      486     0       0
snapshots         pialarm|68      488     0       0
snapshots         pialarm|68      490     0       0
ptupdates         pialarm|68      0       0       0
snapshots         pitotal|69      483     0       0
ptupdates         pitotal|69      0       0       0
ptupdates         pibatch|70      0       0       0
ptupdates         PipeE|31|71     0       0       0
ptupdates         RmpSE|42|72     0       0       0
ptupdates         RandE|43|73     0       0       0
ptupdates         PIPeE|46|74     0       0       0
ptupdates         PIrsE|49|75     0       0       0
ptupdates         PI_RE|47|76     0       0       0
```

A `PIChangeRecordUpdates` entry appears for each secondary PI Server. The column `Pending` shows 0 if all of the pending updates were collected by each secondary PI Server.

Collective PI Servers Have the Same Configuration Data

Verify that configuration databases are the same on all secondary PI Servers. This verification for replicated PI Servers checks for something that does not happen with single PI Servers. You can use the PI base subsystem in offline mode to compare one or more configuration tables. Alternately, you can use PIConfig, Tag Configurator, MDB configurator to dump all or all of the configuration data to text that you can compare. You can also use PI SMT to compare configuration. However using PI SMT can becomes tedious if you need to compare many entries.

Using PIBasess to Compare PI Server Configuration

You can use the **pibasess -compare** option to verify that replicated PI Servers have the same configuration database. This has a similar use as the offline archive utility.

To compare databases, enter the following at a command prompt from the `PI\adm` directory of the primary PI Server:

```
pibasess -compare [-refdb path] -compdb path [-tbl tblname]
```

The **–refdb** and **–compdb** flags specify the paths to the reference and comparison databases respectively. If you omit the **–refdb** option, the installed PI directory is used by default.

The **-tbl** option specifies the table to compare. If you omit the **-tbl** option, all tables are compared. The *tblname* parameter can be one of the following:

```
attribset
collective
context
dbsec
digset
digstate
group
module
point
ptclass
server
trust
user
```

You do not have to stop the PI Base Subsystem to run this command, but the database files must be offline (not currently accessed). You can use the **pibackup** command to create an offline copy of the online databases, so that you do not have to stop the PI Base Subsystem.

Note: You can use PIConfig, TagConfigurator, or the MDB Builder to perform a manual comparison.

Manual Comparison of the Point Database

You can use a command-line script which calls **piconfig** to dump each table on a PI Server to a text file. The resulting files for the primary server can be compared to the files from the secondary servers.

Comment [RP4]: Maybe add a comment that in the release there will be a complete tool for configuration comparison without all the need to dump tables to text files.

Comment [PHG5R4]: People in regulated industries WANT and NEED the text dumps. Then you just have to validate that the database dumps correctly to text, and use any number of tools to compare the resulting text.
Otherwise, you have to validate the comparison tool.
If you build a PIGS comparator, you had better design the test protocol first

Alternatively, use TagConfigurator to retrieve configuration data from each PI Server. The data can be exported to text which can be compared.

Manual Comparison of the MDB

The Module Database (MDB) is used to store configuration data with a history of changes including display and model abstraction, and batch storage. Changes to the MDB data made on the primary server are replicated to the other servers in the collective.

You can use the MDB builder to retrieve module information from each PI Server. The data can be exported to text which can be compared.

List Collective Server Configuration Tables

Entries

To list entries in the PICollective and PIServer tables, enter the following command from the `PI\adm` directory:

```
piconfig < pisysdump.dif
```

Attributes

To list attributes in the PICollective and PIServer tables, enter the following PIConfig commands from the `PI\adm` directory:

```
*    dump both views of the whole table
*
@tabl pisys,picollective
@?atr
*
@tabl pisys,piserver
@?atr
```

Re-initializing a Secondary PI Server

When there is a chance that the PI Server configuration database could be different on some servers in the collective, re-initialize the secondary PI Servers from the primary PI Servers. If you do not re-initialize the secondary servers and they become mismatched, diagnosing problems will be difficult. The effort to diagnose problems with mismatched servers exceeds the effort to re-initialize the servers.

Note: We recommend using the Collective Manager tool to re-initialize a secondary PI Server. The following is an overview of the steps to manually re-initialize the secondary PI Server.

After you shut down the secondary PI Server perform the following steps which were already done once as part of *Collective Formation* (page 24):

1. Verify that the Existing PI Server Is Healthy
2. Verify that the Snapshot Event Queue Is Empty
3. Force an Archive Shift

4. Limit Database Changes on Primary PI Server
5. Verify that the Snapshot Event Queue Is Still Empty
6. Flush the Archive Write Cache
7. Back Up PI Configuration Database and PI Archives
8. Allow Database Changes on Primary PI Server
9. Restore Configuration and Archive Files on Secondary PI Server
10. Start PI on Secondary PI Server

The **pibackup** command supplied with PI Server 3.4.370 or later allows you to back up a PI Server while it is running.

Restoring Configuration and Archive Files using a Command File

You can use a command file to restore files to a secondary PI Server as part of initialization or re-initialization of a secondary PI Server. The following sample command file illustrates one method that can be used:

Contents of file pirestore.bat:

```
@rem Restore PI files
@rem $Workfile: pirestore.bat $ $Revision: 1 $
@rem
@setlocal
@rem default source: current directory
@set pi_s_dir=%cd%
@rem default destination based on PISERVER symbol
@set pi_d_dir=%PISERVER%
@rem default archive destination set later based on pi_d_dir
@set pi_arc=
@
@if [%1] == [] (goto usage)
@goto loop
@:shift3_loop
@shift
@:shift2_loop
@shift
@:shift1_loop
@shift
@:loop
@if [%1] == [-source]   set pi_s_dir=%2%
@if [%1] == [-source]   goto shift2_loop
@if [%1] == [-dest]     set pi_d_dir=%2%
@if [%1] == [-dest]     goto shift2_loop
@if [%1] == [-arc]      set pi_arc=%2%
@if [%1] == [-arc]      goto shift2_loop
@if [%1] == [-go]       goto shift1_loop
@if [%1] == [-?]        goto usage
@if [%1] == [?]         goto usage
@if [%1] == []          goto loop_end
@echo Unrecognized argument "%1%"
@goto usage
@
@:loop_end
@if [%pi_d_dir%] == [] echo Specify argument -dest or set
environment variable PISERVER
@if [%pi_d_dir%] == [] (goto usage)
```

```
@
@set pi_adm=%pi_d_dir%\adm
@set pi_bin=%pi_d_dir%\bin
@set pi_dat=%pi_d_dir%\dat
@set pi_log=%pi_d_dir%\log
@
@if [%pi_arc%] == []    set pi_arc=%pi_d_dir%\dat
@
@echo Deleting message log and audit files from directory:
"%pi_log%"
@del /f /s %pi_log%\pimsg_*.dat
@del /f /s %pi_log%\pi*ssAudit.dat
@
@echo Copying the files to the target directories
xcopy /r /y "%pi_s_dir%\adm\*.*" "%pi_adm%"
xcopy /r /y "%pi_s_dir%\bin\*.*" "%pi_bin%"
xcopy /r /y "%pi_s_dir%\dat\*.*" "%pi_dat%"
xcopy /r /y "%pi_s_dir%\log\*.*" "%pi_log%"
xcopy /r /y "%pi_s_dir%\arc\*.*" "%pi_arc%"
@
@goto bat_end
@
@:usage
@echo. usage: pirestore.bat [-source s_dir][-dest d_dir][-arc
a_dir][-go]
@echo.
@echo.      Delete from d_dir\log\
@echo.            message log files pimsg_*.dat
@echo.            audit files       pi*ssAudit.dat
@echo.      copy archive files from s_dir\arc to a_dir
@echo.      copy other   files from s_dir\*   to d_dir
@echo.
@echo.      s_dir source directory.        default %%cd%%
@echo.      d_dir destination directory.   default %%PISERVER%%
@echo.      a_dir archive destination directory. default d_dir\dat
@echo.      -go   prevents accidental execution with no arguments
@echo.
@:bat_end
```

This command file would be invoked on the secondary computer by entering the following at a command prompt:

```
pirestore -source \\uc-s1\c$\temp\pibackup
```

In this example `\\uc-s1\c$\` refers to an administrative share of the `C:` drive on the `uc-s1` server. The message log files and audit files would be deleted from the `%PISERVER%\log` directory, and then the files would be copied from the `\temp\backup` directory on the administrative share to the `%PISERVER%\*` directories on the secondary computer.

Designating a New Primary PI Server

If the primary PI Server is damaged or unavailable for an extended period of time, you may need to designate a new computer to become the primary PI Server. If the new computer will have the same name, and the current set of data files is available, then the change is fairly simple. However, if the new computer will have a different name, or the current set of data files is not available, or an existing secondary PI Server will be promoted to the primary PI Server, then more work is involved.

Creating a Primary PI Server with the Same Name

To change the computer hardware but not the name of a PI Server, install the operating system and PI Server software as usual, and then move the current set of data files, queue files, and archive files to the new environment.

It is important that you do not restore an old set of backup files, because after a backup is made changes may have been made to the primary PI Server that were replicated to at least one secondary PI Server. These changes are not replicated back from the secondary PI Server to the new primary PI Server, so there will be a difference in the configuration database between the servers in the collective.

If you restore the primary PI Server from an old backup, re-initialize all secondary PI Servers from the primary PI Server. If you restore the primary PI Server from one secondary PI Server, you only need to re-initialize the other secondary PI Servers. Do this to avoid any inconsistency between PI Server databases.

Promoting a Secondary PI Server to be the Primary

To promote a secondary PI Server to become the primary PI Server:

1. Use one of the following methods to verify that there are no updates pending for the secondary PI Server:
   - In the Collective Manager, make sure that LastSyncRecordID is the same for both the existing primary server and the secondary PI Server.
   - On the existing primary PI Server in the `PI\adm` directory, enter:
     ```
     pilistupd
     ```
2. To verify that the LastSyncRecordID is the same on the existing primary PI Server and the secondary PI Server that will be promoted, enter the following on the existing primary PI Server in the `PI\adm` directory:
   ```
   piconfig < pisysdump.dif
   ```
3. Shut down the existing primary PI Server.
4. On the secondary PI Server that will be promoted to be the new primary PI Server, enter the command from the `PI\adm` directory:
   ```
   piartool –sys –promote PIServerName
   ```
   By default the current server is promoted.
5. Restart the PIbasess process on the new primary PI Server.
6. Re-initialize all of the secondary PI Servers with the database from the new primary PI Server. This is especially important for the old primary PI Server, as it prevents any inconsistency between PI Server databases.

This change requires reviewing buffering configuration on the interfaces and the PI Servers, and the specification of the host server for each interface.

Updating the License File

When you designate a new primary PI Server by either using a new computer or promoting a secondary PI Server, your existing license file may only allow the new primary PI Server to run for a short time. If you obtain a new license file from Rockwell Automation, you must update the license file in all of the servers in the collective.

Removing a PI Server from the Collective

To remove a PI Server from a collective, enter from the `PI\adm` directory on the primary PI Server:

```
piartool -sys -drop PIServerName
```

Note: You can remove secondary servers only. If you attempt this operation on a primary server, the server's role changes from primary to non-replicated.

Comment [CF6]: I got this rewrite from Stefan: "If you wish to drop the primary server you must close the server first before dropping it from the collective" But I believe I discussed this with Denis and he asked me to document it this way. Clarify.

Editing the Secondary PI Server Configuration for Special Situations

Under normal conditions, the secondary server configuration cannot and should not be changed. During setup of a collective, after re-imaging, and when recovering from extraordinary events, it may be necessary to edit some entries in tables on the secondary server. PI Table Replication (page 67) shows the tables that you are allowed to change on the secondary.

The most likely things that you will have to change are the PITimeOut table parameters. For example, some parameters apply only to secondary PI Servers and not to primary PI Servers.

The PIFireWall table parameters are not replicated between servers in the collective. If networks change, you must change these parameters on all members of the collective to compensate.

You must change the PIServer table parameters when you change a secondary PI Server to be a primary PI Server as described in *Promoting a Secondary PI Server to be the Primary* (page 49).

Managing Archives on Replicated Servers

Secondary PI Servers that have archives in the same directory structure as the primary PI Server are easy to manage. The registration of the archives moves with the files that are part of the initializing or re-initializing. If a secondary PI Server has archives in a different location, you must register them so that they are available.

It is possible to place archive files on a Storage-Area Network (SAN) drive. If multiple PI Servers mount the archives, they should be on a read-only partition. This can be useful if you have many old archives and you wish to allow multiple servers in the collective access them.

Archive shifts on different PI Servers are not synchronized. They can and do happen at different times on each PI Server in the collective. This increases the availability of the archive data because a shift that takes a long time or fails on one PI Server still leaves other

servers available to receive and serve data. One side effect of this is that if an archive has to be moved from one server to another server, it must be reprocessed to change the start and end times to match the destination.

For more details on the offline archive utility, see *Using the Offline Archive Utility (piarchss)* in the *PI Server System Management Guide*.

Updating PI on Replicated Servers

A design goal of HA is that you can upgrade the operating system and server software on one PI Server in the collective without shutting down the other PI Servers. During the upgrade process on one server, the other servers can provide data to clients. For most reliable replication performance, it is best to keep all of the servers in the collective at the same operating system version and the same PI System software version.

Most people will want to upgrade a secondary PI Server before the primary PI Server. If something goes wrong with the upgrade process, it is easier to create a new secondary PI Server than to recreate the primary and reconstitute the collective. However, it is a matter of choice whether to upgrade the primary PI Server or a secondary PI Server first.

Updating the Operating System on Replicated PI Servers

Organizations that use Windows Update to push upgrades to server computers must be careful to place the servers in the collective in different upgrade groups. Otherwise, a Windows Update could cause all of the servers in the collective to update at the same time. This is especially important if you configure Windows Update to allow unattended, automatic reboots of the host server operating systems.

We recommend that you employ diversity in your upgrade process by configuring the servers to be in different upgrade groups and/or configuring some of the servers in the collective to update manually. You might want to disallow unattended or automatic reboots of the operating system.

Configure Interfaces and Buffering on PI Servers in the Collective

Configuring interfaces on server in a PI Collective requires some additional decisions that are not necessary for non-replicated PI Servers. Each interface can send the time-series data either only to the local server or to all members of the collective via n-way buffering. As with interfaces on interface computers, the interface can be configured to receive configuration data from any server in the collective.

N-Way Buffering

Configure BufServ on the PI Servers to buffer time-series data to the names of each PI Server in the collective. Then add the names of all the servers in the collective to the replicated server list, If you choose to buffer time-series data to localhost, omit that name from the list of replicated servers.

Note: The PI Buffer Subsystem is not supported on PI Server computers.

Local Calculation and Test Interfaces

We recommend that you reserve `pointsource="R"` for a local copy of the Random interface that can run on each PI Server to help with diagnosing problems. We also recommend that you reserve `pointsource="9"` for a local copy of the RampSoak interface that can run on each PI Server. The Performance Equation Scheduler should run locally too.

Configure the interfaces Random, RampSoak, and Performance Equation Scheduler PIPEschd on the PI Servers to have the host server as localhost. If you configure the generic hostname localhost for buffering, keep it off the replicated server list. This prevents the time-series data sent to localhost from being sent to all of the members of the collective.

Computer Monitoring Interfaces

Computer Monitoring interfaces include Performance Monitoring PIPerfMon or PIPerfMon_basic, network host monitoring PIPing or PIPing_Basic, and SNMP recording PISNMP or PISNMP_Basic. When these interfaces are run locally on the PI Servers they should have the interface startup `.bat` file specify the host server as the canonical hostname (not localhost) of the local server. This allows these interfaces to retrieve configuration data from the local server, but causes them to send time-series data to each of the servers in the collective via n-way buffering. If one server fails, the performance data prior to the failure is already on the other servers. This allows system managers to diagnose problems with a computer when it is still down.

Using PerfMon with Replicated Servers

The *Performance Monitor Interface to the PI System* manual describes the PerfMon and PerfMon_Basic interface in detail. It is installed in the `PIPC/Interfaces/piperfmon_basic` directory as part of the PI Server installation.

It is important to configure the PerfMon interface points so that the composite of `pointsource` and `location1` is uniquely allocated to a particular PerfMon interface running on a PI Server or interface computer. If you use PerfMon_Basic on the PI Server computers, an additional constraint is that each installed PerfMon_Basic interface should have a unique `pointsource`.

You can configure PerfMon interfaces on computers that have buffering installed to send the time-series data to all PI Servers by n-way Buffering. We recommend that you enable buffering on the collective PI Servers to support the PerfMon or PerfMon_basic, Ping, Ping_basic, and SNMP, SNMP_basic interfaces.

Configure PerfMon Points

Each PI Server in the collective should have its own set of PIPerfMon points to monitor PI Server performance. We recommend that you use a different `pointsource` for each copy of PIPerfMon_Basic that is run on different servers in the collective. If you are using the full PIPerfMon application, you may use the same `pointsource` for all of the PIPerfMon

interfaces on the PI Servers and assign a different value of `location1` to the interfaces on each server.

You can use any tool that creates PI points to create and edit the PI PIPerfMon points. The PI SMT Performance Monitoring plug-in and the PIPerfCreator application, however, create points for Microsoft Windows performance counters that conform to the ITMonitor tag name convention.

You may find it convenient to create a set of PI PIPerfMon points for the primary PI Server, and then use PI SMT PointBuilder running in Microsoft Excel to duplicate those points for the secondary servers. You would import the point configuration into Microsoft Excel, use Excel editing to change the important configuration parameters, and then export the result to the PI Server.

The most important PIPerfMon point configuration parameters are tag name and extended descriptor ExDesc, which must correspond to the server. With PIPerfMon, the values of {`pointsource`, `location1`, `location4`} are usually {#, 1, 1} for the primary server, and {#, 2, 1} for the secondary server. With PIPerfMon_Basic, the values of {`pointsource`, `location1`, `location4`} are usually {1, 1, 1} for the primary server, and {2, 2, 1} for the secondary server. Once the configuration data is in the Excel spreadsheet, you can use PointBuilder export to create the points.

Example: Perfmon_Basic Interface Configuration

The following table shows a typical configuration of the Perfmon_Basic interface for PI Servers in a collective. Note that the data is sent to all servers in the collective via n-way buffering. Note also that there is a unique pointsource for each installed copy of PerfMon_Basic.

Example Perfmon_Basic Interface Configuration

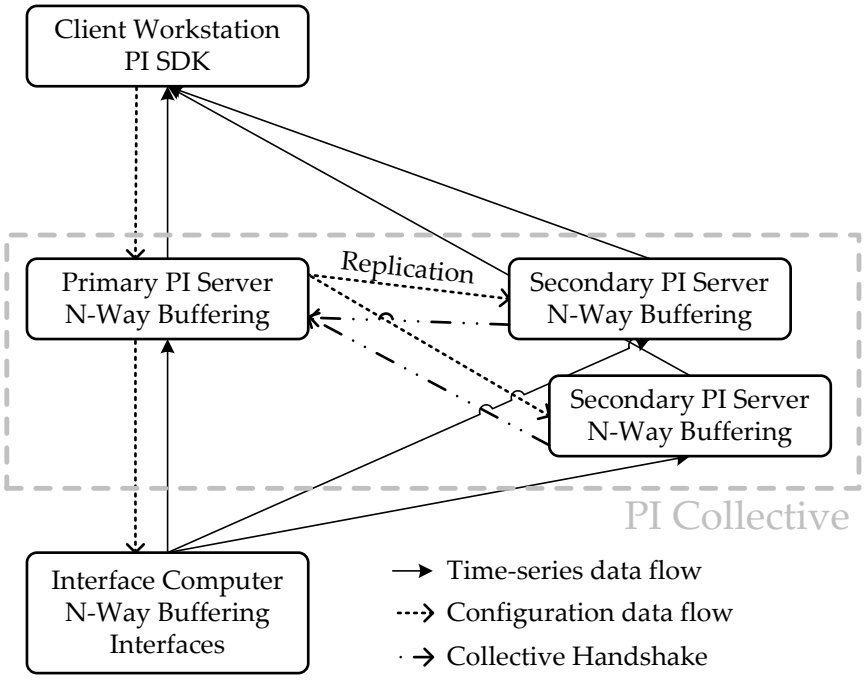
Computer	Interface Name	Host PI Server (/host= parameter)	Type	Buffering	PtSrc (/ps= parameter)	loc1 (/id= parameter)	Event Counter (/ec= parameter)
uc-s1	piperfmon_basic1	uc-s1	perfmon_basic	n-way	1	1	4
uc-s2	piperfmon_basic2	uc-s2	perfmon_basic	n-way	2	2	5

Chapter 4. Replication Reference

How the PI Server Replication Works

New functions in the Base Subsystem (**pibasess**) become active when a server detects that it is the primary PI Server of a collective. The Base Subsystem creates a change log through the update manager for each secondary PI Server to receive configuration changes. The primary PI Server becomes the producer of this change queue, and the secondary PI Server becomes the consumer. The functions also keep track of the status of the secondary servers and provide this status when contacted by the secondary PI Server.

New functions in the Base Subsystem become active when a server detects that it is a secondary server in a collective. The secondary server tries to contact the primary server to report its status and retrieve the status of the other members of the collective. The secondary server retrieves configuration changes updates made to tables on the primary server.



Collective Handshake, N-Way Buffering on Server

In order to keep the point database and MDB identical on all of the servers in the collective, the Base Subsystem includes some new functions. A significant feature is that the secondary PI Server creates points with specified PointID, and modules with specified ModuleID when it receives changes from the primary PI Server. This is not possible using client tools to incrementally duplicate database changes on different PI Systems.

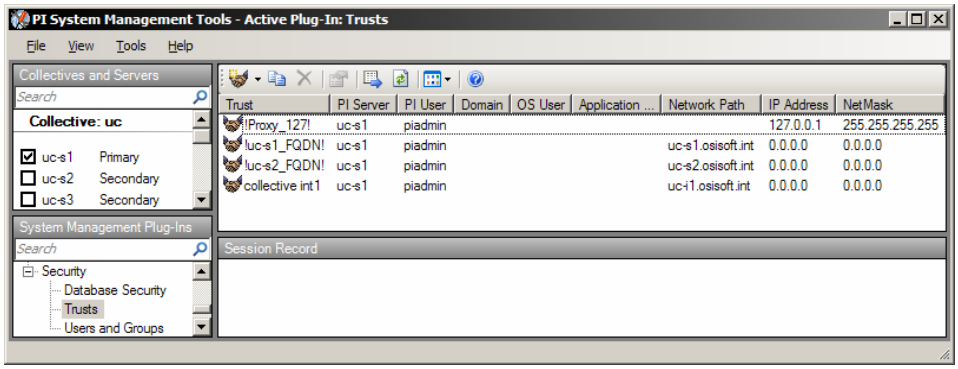
For example, it was always possible to create the same points on two systems. However, if point creation failed on one system, points created later on the two systems would have different PointIDs. It was also possible to use the PI SDK to duplicate modules from one system to another. However, the modules on the second system would have different ModuleIDs.

Trusts and the Collective

Subsystems and applications running on the servers in the PI Collective need trusted access to all servers in the collective. This is provided by trusts on each PI Server that allow trusted access from each PI Server in the collective.

Applications that may connect between the servers include PI SMT, PI ICU, and command-line tools **piconfig**, **piartool**, and **pilistupd**. In this release of the PI Server, only the PIbasess subsystem connects between the replicated PI Servers. If interfaces run on the servers and are set up to distribute time-series data to all of the collective servers using n-way buffering, they must have write access as well.

Each server has the locally generated trust entry `!server_FQDN!` that references each server's FQDN. Each member of a PI Server Collective also has a trust entry `!server_FQDN!` for each of the other servers in the collective. These trusts are not created if a trust already exists that differs by only the trust name and PI user. The following figure shows the trusts entries in the PI SMT Trusts plug-in for the collective **uc**, which includes the servers **uc-s1** and **uc-s2**.



The preceding discussion shows the default behavior. The following table shows all of the other possible trusts that can be created, and illustrates some new behavior in the creation of new trusts, and removal of old trusts during upgrades.

In the PI Server 3.4.370 and earlier, four trusts are created on a PI Server every time the server starts. In PI Server 3.4.375 and later, two of those trusts corresponding to

`!Default_ServerIP!` and `!Default_ServerMachine!` are removed. The loopback trust `!Proxy_127!` is still created. The trust `!Proxy_localhost!` is only created if enabled by a flag in the TimeOut table. The trust `!`*`server`*`_FQDN!` where *`server`* is created for each PI Server in the Server table for the same collective. Non-replicated PI Servers only have one of these trusts created.

Trust Creation

Trust	Example		Trust Creation Behavior		Auto Trust Config Mask
	IP Host	IP Address	Version 3.4.370	Version 3.4.375/PR1	
!Proxy_127!		127.0.0.1	Created	Default	1
!Proxy_localhost!	localhost		Created	Optional	2
!<server>_IPAddr!		192.168.9.25		Optional	4
!<server>_Name!	<server>			Optional	8
!<server>_FQDN!	<server>.osisoft.int			Default	16
!Default_ServerIP!		192.168.9.25	Created	Removed	
!Default_ServerMachine!	<server>		Created	Removed	

The servers in the collective also need trusts for the computers that host interfaces. If these trust entries are made using IP address, only one entry must exist for each interface computer.

For more information about configuring trusts, see:

- *Managing PI Security* in the *Introduction to PI Server System Management*
- *Managing Security* in the *PI Server System Management Guid*e

Server Tables Supporting Replication

Replication of PI Server configuration requires new configuration data to control the replication. The data resides as entries in the new PICollective and PIServer tables stored in the `picollective.dat` and `piserver.dat` files.

The following table shows the attributes of the PICollective table. This table has one row corresponding to the collective. The collective name is used to link the entries in the PIServer table to the PICollective table. Every PI Server presents the Collective ID as the Server ID to each application using the PI SDK to connect to a server in the collective. This allows all displays and applications that depend on the Server ID to connect to any PI Server in the collective without change.

The Status attribute shows the overall status of the collective. This is a PI error code that can be looked up using the **pidiag –e** command.

Table PICollective Attributes

Attribute	Type	Editable	Example	Comments
Name	String	Primary	"uc-s1"	Must match collective name defined in PISystem table
CollectiveID	String	Primary	"08675309-0007-0007-0007-000000001001"	a UID
Description	String	Primary	"UC 2006 Demo Collective"	
LastCollectiveConfigChangeTime	TimeStamp	No	12-Apr-06 14:00:17	
Status	Int32	No	0	0 = good

PIServer Table Attributes

The following table shows the attributes of the PIServer table. This table must have one row for each server in the collective. The Name is the computer hostname. The name must uniquely identify the row in the table. The Collective attribute in the PIServer table must match the Name attribute in the collective table.

In both tables, some of the fields can be edited to specify the PI collective configuration. Other fields are read-only and reflect statistics or run-time status. Once the database is moved from the primary to the secondary, additional changes to the attributes listed as editable=primary are propagated by replication to the secondary so that the values remain the same on all servers. Attributes listed as editable=yes can be changed on all of the PI Servers.

Attributes that are not edited are computed locally on each server, propagated from primary to secondary, or from secondary to primary. For example, the value of the IsCurrentServer attribute is computed locally on each server and identifies which server is providing the information from the table.

The value of the Status attribute is propagated from the primary to the secondary for all but the local server, and tells you the status of each server. The status attribute is a PI error code that can be translated to text using the **pidiag –e** command.

Table PIServer Attributes

Attribute	Type	Editable	Example	Comments
Name	String	Primary	"uc-s2"	The computer hostname (non-qualified). Unique key in the PI Server table. Each server uses this to find its own entry in the table.
Collective	String	Primary	"uc-s1"	Must match collective name defined in PIcollective table

Attribute	Type	Editable	Example	Comments
CommPeriod	Int32	Primary	20	Period for secondary PI Server to send status to primary, receive status of collective members, and retrieve configuration changes. Set on primary PI Server to affect behavior of secondary PI Server.
CommStatus	Int32	no	0	Status of the last secondary PI Server communication with the primary server (0 is good)
Description	String	Primary	"UC 2006 Demo Server 2"	
FQDN	String	Primary	"uc-s2.osisoft.int"	FQDN or IP address used to connect to collective servers
IsAvailable	BYTE	no	1	1 if available for client access, 0 otherwise. Derived from all other status fields in the table.
IsConnectedToPrimary	BYTE	no	1	1 Indicates that the secondary PI Server is connected to the primary PI Server, Always 1 on a primary PI Server.
IsCurrentServer	BYTE	no	1	1 on the responding PI Server, 0 for all others
IsTCPListenerOpen	BYTE	no	1	1 indicates this PI Server TCP listener is open
LastCommTime	TimeStamp	no	12-Apr-06 14:00:17	Last time the secondary PI Server communicated status to the primary PI Server
LastSyncRecordID	Uint64	no	68	Number of changes each PI Server applied to the replicated tables
LastSyncTime	TimeStamp	no	12-Apr-06 14:00:17	Last time synchronization succeeded on secondary PI Servers
NumConnections	Uint32	no	11	Total number of connections on the specific PI Server
PIVersion	String	no	3.4.375.29	
Port	Uint32	Primary	5450	
Role	Int32	Primary	2	0 for non-replicated; 1 for primary; 2 for secondary
ServerID	String	Primary	"08675309-0007-0007-0007-000000001002"	A UID representing the unique PI Server identification
SyncFailReason	String	No		Reason that synchronization did not succeed

Attribute	Type	Editable	Example	Comments
SyncPeriod	Int32	Primary	10	Synchronization period in seconds. 0 means synchronize on demand only. Set on primary PI Server to affect behavior of secondary PI Server.
SyncStatus	Int32	no	0	On secondary servers: the status the last time that synchronization was attempted (0 is good)
UnavailableReason	String	no		

Typical Status Values

Typical Status Values

Status	Error Text	Description
0	Success	
-10420	Direct write operations disallowed on secondary server of collective	Cannot change configuration on secondary
-16030	Batch database access disabled on secondary server of collective	The Batch database on the secondary server does not have batches unless archives are moved
-18000	Server has not communicated its status	The primary server never connected to this secondary
-18001	Unable to process further change records, missing update in sequence	Change record sequence error
-18002	Error processing update	Change record processing error
-18003	No producer found on the primary server	The primary server does not produce change records
-18004	Unable to register change producer	The primary server cannot produce change records
-18005	Server misconfigured, no local server defined	The server cannot find its own hostname in table
-18006	Server misconfigured, no primary server defined	The server cannot find the primary server in table
-18007	Unable to connect to primary server	The primary server is not reachable
-18008	Unable to sign up for changes on the primary server	The primary server is not responding
-18009	Unable to unsignup for changes on the primary server	The primary server does not acknowledge unsignup
-18010	Unable to disconnect from the primary server	The primary server does not acknowledge disconnect
-18011	Server misconfigured, collective not found	The server cannot find the collective name
-18012	Invalid role defined for this server	The server role is not valid

Status	Error Text	Description
-18013	Member server is in error	The collective status shows this when a server is in error
-18014	Error reading change record from the primary server	Change record read error
-18015	Unable to send updates to secondary servers	
-18016	Unable to send status update to the primary server	
-18017	Unable to get change records from the primary server	
-18018	Server has not communicated	
-18019	Member has been removed from the PI Sys Database	
-18020	Member clock as drifted too far from the primary's clock	
-18021	Failed to enable replication	
-18022	Server has failed to keep synch	
-18023	Unable to save unprocessed config changes to disk	
-18024	Unable to load unprocessed config changes from disk	
-18025	Server is not a member of a collective	
-18026	Promotion is not supported on the primary server	
-18027	Server is not ready to process change records	
-18028	Unable to sign up all secondary servers to receive config changes	
-18029	Unable to sign up all secondary servers to receive config changes	
-18030	Name cannot contain spaces or other invalid characters	
-18031	Localhost is not a valid name	
-18032	FQDN is invalid	
-18033	Two servers have the same FQDN or they resolve to the same IP address	
-18034	Unable to produce update for secondary servers	
-18035	Error reading update	
-18036	Unable to rename the current server	
-18037	Unable to rename a server while it's online	

Status	Error Text	Description
-18038	Unable to remove the current server	
-18039	Unable to remove an server while it's online	
-18040	Unsupported replication change record version; upgrade may be required	
-18041	Two servers have the same Server ID	
-18045	License Has Expired	
-18046	Unable to Resolve the FQDN	
-18047	Unable to Resolve the IP Address	
-18048	Unable to remove collective	
-18049	Unable to get change records from the primary server	

Message Log Entries

The following table lists some of the messages sent to the message log by the replication process. In this table *process* refers to the first text field in the message log that can be searched with the **pigetmsg -pn "process"** command. The **Server** column indicates which PI Server in the collective displays the message.

Message Log Entries

Process	Server	Message	Description
pirepl	Primary, Secondary	Secondary server uc-s2 is online and connected	Primary server reports that secondary server connected
pirepl	Secondary	Secondary server uc-s2 is online, but disconnected from the primary server	Secondary server was connected, now is not
pirepl	Primary, Secondary	Unable to establish this server's identity in the PISys,PIserver table	Hostname did not match name for any entry in PIserver table
Secondary server name	Primary	Primary Queue successfully initialized (64MB/1024KB)	Primary has created event queue for secondary
Secondary server name	Primary	Primary Queue successfully loaded (0 events)	Primary has loaded event queue for secondary

PITimeOut Table Parameters

The following table lists the PITimeOut table parameters that control aspects of the replication process.

PITimeOut Table Parameters for Replication

Subsystem	Name	Min – Max (Default)	Read	Description
PIBasess	AutoTrustConfig	0 – 255 17		Bitmask that controls automatic trust configuration 0 = NONE 1 = Loopback 2 = Localhost 4 = IPaddr 8 = Hostname 16 = FQDN 127 = v3.4.370 Compatible 255 = All 17 = LoopBack + FQDN
PIBasess	Replication_EnableBDBAccess	0 – 1	Every one minute	0 or missing: secondary computers respond to Batch RPCs with error. 1 enables Batch RPCs on secondary computers.
PIBasess	Replication_EnableSDKWriteValues	0 – 1	Every one minute	0 or missing on secondary computers prevents the PI SDK 1.3.4 or later from writing data. 1 allows the PI SDK 1.3.4 or later to write data. The PI SDK 1.3.3 or earlier is not affected by this parameter.
PIBasess	Replication_ClockDiffLimit	10 – 600 (300 sec)		Secondary allows this difference in server clocks before raising syncfailure
PIBasess	Replication_CommFailureGracePeriod	0 – 10000 (60 sec)		Amount of time that the primary server allows a secondary to be late in communicating before marking the server as unavailable
PIBasess	Replication_PeriodicTaskInterval	1 – 10000 (20 sec)	startup only	Secondary server period for sending status to primary, receiving status of collective members, retrieving configuration changes
PIBasess	Replication_SimulateChgRecVersion	1 – 65535	startup only	
PIBasess	Replication_SyncFailureGracePeriod	0 – 10000000 (0 sec)		Primary server allows a secondary to be late in synchronizing before marking the server as unavailable

Subsystem	Name	Min – Max (Default)	Read	Description
PIBasess	Replication_ SyncGiveupGracePeriod	0 – 10000000 (0 sec)		Primary allows the sync to be broken (or not syncing at all), before the primary marks the secondary as unavailable and stops sending configuration changes to that secondary. Ignored if the secondary's syncperiod is 0.
PIUpdMgr	Update_PersistentConsumerTimeout	0 – (86400 sec)	startup only	Amount of time that the persistent consumer is retained after last communication. This does not apply to the replication consumer, which never times out.

How N-Way Buffering Affects Server Compression

BufServ

For complete details on the BufServ features, click **View Help** in the About PI SDK application supplied with the PI SDK.

Each PI Server's snapshot subsystem receives the same time-series data from the interfaces. Compression occurs independently on each PI Server in the collective. The trend line drawn through the archive values from each server is within CompDev of all of the time-series values received from the interface for that point. However, because the compression algorithm for each point initializes at different time on the servers, the algorithm might select different time-series values for storage in the archive.

Points that do not change, or that change slowly with constant slope, can have different archive values on different servers. However, the resulting trend line drawn through the archive values is identical.

The following discussion assumes an understanding of the compression algorithm as described in *Exception Reporting and Compression Testing* in the *PI Server Reference Guide*.

The compression algorithm is initialized after several actions, including selecting a value to be sent to the archive, editing the compression parameters for a point, and rebuilding the snapshot table. Initializing the compression algorithm resets the CompMax counter and causes the current snapshot value to be selected to be sent to the archive. Since the editing of compression parameters is applied at different times to different servers in the collective, each Snapshot process selects a value to be sent to the archive and starts the counter for the edited points at different times.

When the compression parameters for a point are edited on the primary PI Server, the secondary PI Servers receive the change from PI Server replication and apply the change at some time later. Since buffering to any PI Server might be ahead or behind another server, the new compression parameters might be applied to newer or older time-series data than has been processed on the primary PI Server.

If the time-series value does not change, or if the slope of the trend does not change enough for a point to exceed the CompDev parameter, the next value is selected for the archive only

after CompMax has been exceeded. If CompMax for a point is set to the default 2880 seconds (8 hours), the compression algorithm selects a value for the archive every 8 hours even if the slope of the trend does not change.

The PI Buffer Subsystem

The PI Buffer Subsystem is an enhancement to the PI System that allows the buffering to be more aware of the members of the collective, and to take full advantage of the PI 3 protocol. These enhancements make the initialization of collective servers easier to manage, and allows more time-series data to flow from the interface to the server than is possible with BufServ.

The PI Buffer Subsystem causes all of the PI Servers to store the same time-series events in the archive. No PI Server selects more or fewer time-series events than another PI Server. This occurs because the PI Buffer Subsystem runs the compression algorithm before the time-series data is sent to the PI Servers. This results in identical time-series events in the archives on all PI Servers.

For details on the differences between the Buffer Subsystem and BufServ, see the Buffer Subsystem help in `PIHOME\Help\pibufss.chm` (typically `C:\Program Files\Rockwell Software\FactoryTalk Historian\Server\PIPC\HELP`).

How the PI SDK Works with the Collective

The PI SDK 1.3.4 or later includes features that enhance the behavior of client tools with the collective. Workstations that have earlier versions of the PI SDK can still be used to connect to individual servers in the collective, but cannot take advantage of these features.

Automatic Failover between Collective PI Servers

Normal shutdown via the `pisrvstop.bat` batch command issues the `PIartool -sys -close` command to close the TCP listener and send a signal to the client applications that are connected to the server via the PI SDK. When the PI SDK on a computer receives this signal, it attempts to reset the connection to another available server in the collective.

During abnormal shutdown of the PI Server, or interruption of communications with the PI Server, the PI SDK does not know that the server is no longer available until it tries to communicate with the server. After a communications timeout occurs, the PI SDK checks the connection and attempts to connect to another server in the collective. The initial timeout is set on the workstation to 60 seconds by default. It can be set to different values on each workstation.

Configuration of PI SDK KST

When the PI SDK first connects to a server in a collective, it receives a list of all of the servers in the collective. It uses this to build a list of the servers in the KST.

The PI SDK receives a periodic update of connection statistics from PI net manager (**pinetmgr**) running on the connected PI Server. The update contains information about the health of all of the servers in the collective.

Periodic receipt of health information allows the PI SDK to be more efficient during failover. The PI SDK connects to a server that is available, and avoids connecting to a server that unavailable. In collectives with more than two PI Servers, this avoids a timeout waiting for a known unavailable server.

Specifying PI Server Priority

The PI SDK Connection Manager dialog box allows specifying the priority for connecting among the servers in the collective. When the PI SDK needs to connect to a PI Server in the collective, it selects the server with the lowest priority from the available servers. The PI SDK does not select any server with a priority set to -1.

Manual Selection of a PI Server in a Collective

The PI SDK Connection Manager is used to specify the collective or the non-replicated PI Server for connection. If your application is connected to the collective, the PI SDK selects the member of the collective to use for the connection.

The PI SDK Connection Manager also allows you to force the PI SDK to select a different PI Server in the collective. This selection is controlled by the priority of the server entered in the local KST. If only one PI Server in the collective is available, this has the effect of re-connecting to the same server.

For details, see *PI SDK Server Connection and Automatic Client Failover* (page 13).

Backward Compatibility with Earlier Versions of the PI SDK

The PI SDK 1.3.3 or earlier can be used with PI Servers in a collective with the following limitations:

- They are completely unaware of PI Server Collectives.
- They do not automatically fail over between servers in the collective.
- They do not respect the `Replication_EnableSDKWriteValues` parameter. If they are used to write time-series data to a secondary PI Server, the data is not n-way buffered to the other servers in the collective. To protect against unintentional entry of time-series data, the KST should contain entries for the primary PI Servers in Collectives and Non-Replicated PI Servers only. **Do not add entries for secondary PI Servers to the KST.**

Replication Behavior

PI Table Replication (page 67) lists the PI tables, the key values in the tables, and important aspects of replication behavior. If a PI table replicates to the secondary, the key values are kept the same on the PI Servers, and referential integrity between rows in different tables on each server is preserved. In general, for tables that replicate, changes are allowed only at the primary and no changes are allowed at the secondary.

The table titled PI Table Replication shows the exceptions to the general rule that no changes are made to tables on the secondary server. Changes to the PITimeOut tables are allowed at the secondary PI Server in order to allow configuring the secondary PI Server differently than

the primary PI Server to accommodate special situations or recover from primary PI Server failure.

Association of Key Values

The utility of the association of the keys values between servers is best illustrated by the point database. Each PI Server in the collective shares the same association between Tag, PointID, and RecNo. The association between Tag and PointID allows the interface computers to employ simple logic to send time-series data to the servers without keeping track of different PointIDs for each server. This association also allows clients to efficiently retrieve historical time-series data from the archive on different server members of the collective, without changing, re-compiling, or re-linking the display configuration.

PI Table Replication

PI Table	Keys			Replicates to Secondary?	Can Configure on Secondary?
	Primary, Unique	Identity	Foreign		
PIPoint	Tag	PointID, RecNo	SetNo, UserID, GroupID	yes	no
PIDS	Set	SetNo		yes	no
PIUser	User	UserID	GroupID	yes	no
PIGroup	Group	GroupID		yes	no
DBSecurity	DBName		UserID, GroupID	yes	no
PIModules	UniqueID		PointID, UserID, GroupID	yes	no
PI_GEN, PITimeOut	name			no	yes
PI_GEN, PIFirewall	hostmask			no	yes
PITrust	Trust		UserID, GroupID	yes	no
PICollective	name			yes	no
PIServer	name		PICollective.name	yes	no
PIBAUNIT	UnitName	UnitID	PointID, UserID, GroupID	no	yes, but not recommended
PIBAALIAS	Alias			no	yes, but not recommended
PIATRSET	Set			no	no
PIPTCLS	Class			no	no

Each PI Server in the collective also shares the same association between Digital State Set and SetNo; User and UserID; and Group and GroupID. Since PI points refer to SetNo, UserID, and GroupID as foreign keys, sharing this association maintains referential integrity between the points and the digital state sets, users, and groups. This also maintains referential integrity between entries in other tables which use these identifiers as foreign keys. The result

is that applications do not have to translate the keys depending on which PI Server in the collective with which they happen to communicate.

Tables that do not Replicate

The PIPtCls table, which contains information about point classes, and the PIAtrSet table, which contains information about attribute sets, are not replicated. Entries in these tables contain meta-information about the attributes of point configuration stored in the PIPoint table. It is rare to make changes to these tables on a production system. Changing entries in these tables requires placing the PI Server in standalone mode, which is the single-user mode of the PI Server. Because of these factors, changes to these tables are not replicated to the secondary server. After you make changes to these tables on the primary PI Server, you must re-initialize all secondary PI Servers in the collective.

The PIAtrSet table also contains default values of attributes. These may be changed on the primary PI Server, but the changes are not replicated to the secondary PI Server. These default values are applied only during point creation and only when values are not supplied. Since replication creates identical points on the secondary, the fact that these default values are not replicated causes no problems for operation of a collective. Because of this, it is not necessary to immediately re-initialize the secondary PI Servers if you only change the attribute default values. However, if you ever promote a secondary PI Server to a primary PI Server, the collective will revert to the default values that were present on the secondary PI Server.

The Batch subsystem maintains the PIBaUnit and PIBaAlias tables, which do not replicate to the secondary server. See the *Batch Subsystem* chapter in the *PI Server Applications User Guide* for more details. We recommend the Batch Database (BDB) for new batch applications because it builds on the Module Database (MDB), which is replicated.

Note: You can only add and query BDB batches on the primary PI Server unless you set the `Replication_EnableBDBAccess` parameter to 1 on the secondary PI Server. BDB batches are not replicated among collective members.

Machine-specific configuration tables, PI_GEN, PITimeOut and PI_GEN, PIFirewall tables must be changed locally on each PI Server through the use of PI SMT or PIConfig. These tables only have to be changed to accommodate hardware-specific or network-specific conditions.

Replication Performance Points

PI Replication exposes some Windows Performance Monitoring points. These can be viewed in the Windows Performance Control Panel, or can be scanned by the PI PerfMon interface. The following table lists the performance points that are available:

PI Replication Performance Counters

Performance Object	Counter	Description
PICollective		PI Collective Statistics
	Is Running Normally	Is the status normal for all members of the collective?
	Last Config Change Time	Last time the configuration of the collective was modified.
	Current Server	The index of the current server of the collective.
	Number of Servers	The number of member servers in the collective.
PIServer		PI Server Statistics
	Is Communicating	Is the server communicating to the other members of the collective?
	Is In Sync	Is the server in sync with the other members of the collective?
	Is Available	Is the server available?
	Is Current Server	Is the server the member of the collective sending this information?
	Last Sync Record ID	Last sync record processed.
	Role	The role this server plays in a collective.
	Sync Records/sec	Sync records processed/Sec.
	Communication Period	The frequency the server is configured to communicate with the collective.
	Sync Period	The frequency this server is configured to sync with the collective.
	Last Communication Time	Last time the server communicated with the collective.
	Last Sync Time	Last time the server synced with the collective.
	Server Index	The index of the server in the list of members of the collective.

PIsysID.dat

The `PIsysID.dat` file does not exist in the PI Server PR1 release. When the PI Server is upgraded from prior versions, it transfers the Server ID to the PIServer table and then automatically deletes it.

The **pidiag -rcsid** command is new in the PI Server PR1 release. If you are downgrading to a non-HA version of the PI Server, this command recreates the `PISysID.dat` file.

The **pidiag -did** command works only when the server is not running. Use the Collective Manager or **piconfig** to view the Server ID for a specific server.

The **pdiag -cid** command does not exist in PI Server PR1.

Updates in the DBSecurity Table

The DBSecurity table includes a new entry, PIReplication, for managing the PIServer and PICollective tables. The PIReplication entry controls permissions for:

- Adding, editing, renaming, or deleting entries in the PIServer and PICollectives tables
- Forcing a sync
- Opening or closing a server
- Putting a PI Server into—or taking a server out of—standalone mode
- Promoting a secondary server to primary and removing a server from a collective

Chapter 5. Backup Addendum

This chapter is a supplement to the *PI Server Backup* chapter in the *PI Server System Management Guide* version 3.4.370. We include it here because the backup process is integral to the initialization and re-initialization of secondary PI Servers from primary PI Servers.

Backup Behavior in PI Server 3.4.375

This section discusses changes to the PI Backup Subsystem since version 3.4.370 and earlier.

New backup features in PI Server 3.4.375

If you are upgrading from PI Server 3.4.370, your backup procedure will not change and the following features do not apply. However, if you have upgraded from a version prior to 3.4.370 or if PI Server 3.4.375 is installed, then the new backup scheme is in effect.

The following has changed since version 3.4.370:

- VSS backups launched with **pibackup.bat** do not use **NTBackup.exe** by default to perform VSS backups. The PI Backup Subsystem is now a VSS Requestor as well as a VSS Writer. **NTBackup.exe** is not delivered with Windows Vista and Windows Longhorn. You can change the **pibackup.bat** default behavior by creating a custom `pintbackup.bat` command file.
- Since **NTBackup.exe** is not used for VSS backups, the files in the destination directory are not packed into a `.bkf` file. The actual files are visible in the backup directory. This strategy allows files to accumulate in the backup directory. If a third-party backup application is used to backup the PI backup directory, it is better to back up individual files instead of backing up a `.bkf` file. For example, if a `.bkf` file is backed up, both the third-party backup application and **NTBackup.exe** are required to restore the backup.
- Backups initiated by **pibackup.bat** copy archives to the *`backupdir`*`\arc` directory by default for both VSS and non-VSS backups.
- Backups initiated by **pibackup.bat** by default are full backups on Monday and incremental backups on every other day. If there are no files in the backup directory, the incremental backups are the same as full backups. During an incremental backup, source files are not copied if a file of the same name and same last modified date exists in the backup directory. The default behavior can be changed by creating a custom **pintbackup.bat** command file.

- The **piartool -backup** command now launches a VSS backup on Windows Server 2003 or later, unless the **-nonvss** flag is specified. Previously the **piartool -backup** command could only be used to launch non-VSS backups.
- The **piartool -backup** command can be used to select individual archives for backup with the **-archive** flag. This flag was not previously available.

Upgrade Considerations

Backup Behavior after Upgrade

If you upgrade from PI Server 3.4.370 to PI Server 3.4.375, the behavior of your backups will not change as long as your nightly backups were installed with the **pibackup.bat** *backupdir* **-install** command. This command installs a scheduled task that launches a backup via the **pibackuptask.bat** script. When this script runs, it looks for the **pibackup_3.4.370.bat** script. If it exists, **pibackup_3.4.370.bat** runs instead of **pibackup.bat**. The PI Server setup kit installs **pibackup_3.4.370.bat** only if you upgrade from version 3.4.370 to version 3.4.375. The **pibackup_3.4.370.bat** script preserves the version 3.4.370 behavior of using **NTBackup.exe** to perform its VSS backups. If you delete the **pibackup_3.4.370.bat** file, then the version 3.4.375 backup behavior will be in effect.

If it exists, the **pibackup_3.4.370.bat** file is included in the PI Server Backups. This means that if you want to create a PI Server Collective and you follow the default procedure for initializing or re-initializing the secondary PI Server, **pibackup_3.4.370.bat** is copied to the secondary PI Server.

Converting PI Server 3.4.370 Backups to PI Server 3.4.375 Backups

If you have upgraded to PI Server 3.4.75 from version 3.4.370, perform these steps:

1. Delete `pibackup_3.4.370.bat` from the `C:\Program Files\Rockwell Software\FactoryTalk Historian\Server\PI\adm` directory. If you have a PI Server collective, delete the file from all nodes in your collective.
2. If necessary, create a new `pisitebackup.bat` file. See *Customizing Your Backup* (page 72) for details.

Converting Backups Prior to PI Server 3.4.370 to PI Server 3.4.375 Backups

If you have upgraded to PI Server 3.4.75 from a PI Server version earlier than 3.4.370, perform these steps:

1. Re-install your backup scheduled tasks with the **pibackup.bat** *backupdir* **-install** command. See the *PI Server System Management Guide* from 3.4.370 for details.
2. If necessary, create a new `pisitebackup.bat` file. See *Customizing Your Backup* (page 72) for details.

Customizing Your Backup

Backups are customized by creating a custom `pisitebackup.bat` and/or `pintbackup.bat` file in the `PI\adm` directory. These files do not exist by default. Do not

edit the `pibackup.bat` or `pibackuptask.bat` file, as these files are overwritten during an installation or upgrade.

The pisitebackup.bat File

If the `pisitebackup.bat` file exists, then the `pibackup.bat` backup script calls it immediately before exiting. If you want `pibackup.bat` to perform any tasks each day after the backup, add these tasks to a file called `pisitebackup.bat` in the `PI\adm` directory.

You can use the `adm\pisitebackup.bat` file to move the backup directory to tape or other offline media for safekeeping. Alternatively, you can copy the backup to a remote computer.

An example site backup script, `pisitebackup.bat.example`, is delivered with the PI Server setup kit. This script copies the files that were backed up by `pibackup.bat` to a remote server, and copies the backup files to a new backup directory so that the previous backup is not overwritten. To use the example script, rename it to `pisitebackup.bat` and edit the script, specifying the UNC backup path to the remote computer. The example script includes instructions for editing it.

The pintbackup.bat File

See the *PI Server System Management Guide* for details on creating a `pintbackup.bat` file to customize your backup.

PIBackup.bat Behavior in PI Server 3.4.375

The **pibackup.bat** script performs VSS backups on Windows Server 2003 or later. On Windows XP and Windows 2000 Server, **pibackup.bat** performs a non-VSS backup. The actual files that are backed up and the directory structure that is created are identical for VSS and non-VSS backups. The **pibackup.bat** script no longer relies on **ntbackup.exe** to perform its VSS backups. This means that the files are no longer packaged in to a `PI_backup.bkf` file.

The PI Backup Subsystem copies the individual files during the backup and none of the files are compressed into `.zip`, `.bkf`, or other format. This allows the files to accumulate over time in the backup directory. If a third-party backup application is used to backup the PI Backup directory, the actual files are backed up. This is better than backing up a `.bkf` file with the third party backup application.

By default, the **pibackup.bat** *backupdir* command backs up files to the following directories:

Directory	Files Backed Up
backupdir\dat	Files from `\PI\dat` except archive and annotation files
backupdir\arc	Archive and annotation files
backupdir\log	Files from `\PI\log`
backupdir\bin	Files from `\PI\bin`

You can change the default behavior of **pibackup.bat** by creating a custom **pintbackup.bat** command file. See the *PI Server System Management Guide* for details.

The **pibackup.bat** *backupdir* command performs a full backup on Monday, and an incremental backup every other day of the week. If there are no files in the target backup directory, then incremental backups are the same as full backups. During an incremental backup, source files are not copied if a file of the same name and same last modified date exists in the backup directory.

The command-line arguments to **pibackup.bat** have not changed since version 3.4.370. See to the *PI Server System Management Guide* for details on command-line arguments.

Plartool Backup Command Summary

You can use the **piartool –backup** commands to either launch a backup or issue some other auxiliary backup command. The syntax for launching a backup versus issuing an auxiliary backup command is described in the following sections.

Note that the **pibackup.bat** script uses the **piartool –backup** command to launch the backups that it performs.

Launching Backups with piartool

The syntax of the **`piartool –backup`** command for starting a backup is:

```
piartool –backup path [-component comp] [-archive N] [-numarch N]
    [-cutoff date][-wait [sec]] [-nonvss] [-incremental] [-arcdir]
```

In this syntax, text in *italic* type indicates a placeholder. Text in brackets `[]` indicates an optional parameter.

Argument	Description
path	Complete file path (`C:\temp\pibackup`) or UNC path (`\\myserver\c$\temp\pibackup`) to a directory with sufficient space for the entire backup

-component *comp*	Back up only the component specified by *comp*.
	For example:
	`piartool –backup C:\Program Files\Rockwell Software\FactoryTalk Historian\Server\PI\backup -component pibasess`
	backs up only the files that belong to the PI Base Subsystem. To see a full list of the components, enter:
	`piartool –backup –identify –verbose`
	The **-component** flag takes precedence over the **-numarch** and **-cutoff** flags, which are used only to restrict the number of archive components that are backed up. If the **-component** flag is not specified, all components are backed up except for the archive components that are restricted from backup by the **-numarch** and **-cutoff** flags.
	The **-component** flag also takes precedence over the **–archive** flag.

-archive *N*	Back up only the specified archive number *N* and its associated annotation file. Archive numbers begin at 0 for the primary archive. This flag is ignored if the **-component** flag is specified. The **–archive** flag takes precedence over the **-numarch** and **-cutoff** flags.

Argument	Description
-numarch *N*	Back up *N* archives. For example, specifying **-numarch 2** backs up the primary archive and archive 1, provided that the primary archive and archive 1 contain data. Empty archives are not identified for backup. The default number of archives for backup is 3, unless otherwise specified by the **Backup_NumArchives** timeout parameter. This flag is ignored if the **–component** or **–archive** flag is specified.
-cutoff *date*	Cutoff date in PI time format. For example, **-cutoff *-10d** restricts the backup to archives that contain data between 10 days prior to current time and current time. The more restrictive of **-numarch** *N* and **-cutoff** *date* takes precedence. The default cutoff date is 1-Jan-1970 00:00:00, unless otherwise specified by the **Backup_ArchiveCutoffDate** timeout parameter. This flag is ignored if the **–component** or **–archive** flag is specified.
-wait [*sec*]	Wait up to *sec* seconds for the non-VSS backup to complete before returning from the **piartool -backup** command. The progress of the backup is reported every 15 seconds and when the backup is complete, the status of the backup is reported via **piartool -backup -query**. If the **-wait** flag is used without specifying the optional *sec* parameter, then the **piartool -backup** command waits up to 1 day for the backup to complete. If the **-wait** flag is not specified, then **piartool -backup** returns immediately. In this case, the progress of the backup can be monitored with the **piartool -backup -query** command. The **-wait** option is used by the backup script for non-VSS backups because it is important for the backup to complete before the site-specific backup scripts are called.
-nonvss	Perform a non-VSS backup on Windows Server 2003 or greater. This option has no effect on Windows XP or Windows 2000.
-incremental	Files are not be backed up if a file in the backup directory has the same name and modified date as in the source directory.
-arcdir	Back up archives and annotation files to the *backupdir*`\arc` directory. If this flag is not specified, archives are backed up to a directory depending upon their current location. For example, if an archive is in the `C:\Program Files\Rockwell Software\FactoryTalk Historian\Server\PI\archives` directory, then it is backed up to *backupdir*`\archives`. Similarly, if an archive is in the `C:\PI\dat` directory, then it is backed up to *backupdir*`\dat`.

Secondary Backup Commands for piartool

The secondary **piartool –backup** commands are typically used for troubleshooting and monitoring the course of a backup. The syntax is:

```
piartool –backup Arg1 [Arg2] [Arg3] […]
```

In this syntax, text in *italic* type indicates a placeholder. Text in brackets `[]` indicates an optional parameter.

If *Arg1* does not begin with a hyphen (-), then it is assumed to be the destination directory for the backup. See *Launching Backups with piartool* (page 74). If *Arg1* begins with a hyphen, then it must be one of the secondary commands in the following table.

Secondary Backup Arguments	Description
-abort	Abort a current running backup. For example: `piartool –backup –abort`
-query [-verbose]	▪ Reports a list of subsystems that are currently registered for backup. ▪ If a backup is not in progress, reports the status of the last backup. ▪ If a backup is in progress, reports the type of backup and the status of the backup. Example: `piartool –backup –query`
-identify [-numarch *N***] [-cutoff** *date***] [-verbose]**	Report the list of files that the PI Server will back up. If the **-verbose** flag is specified, the PI Server reports a list of files and components. A component is a logical grouping of files. For example, all of the files for the base subsystem are grouped under the **pibasess** component. The purpose of a component is to identify a group of files for backup. The **-numarch** and **-cutoff** flags have the same meaning as the corresponding **piartool –backup** *backupdir* command that is used to start backups. The identify command creates a `\PI\dat\pibackupfiles.bks` file. This file is used in the **pibackup.bat** script to specify the list of files to back up using **NTBackup.exe**. **NTBackup.exe** is used by the backup script for performing VSS backups on Windows 2003 Server. Example: `piartool –backup –identify –verbose`
-SimulateVSS *simulate_command*	Control writes to PI database files as if an actual VSS backup is taking place. The simulate commands put the PI Server into a frozen state before the snapshot is taken, and unfreeze them afterwards. This is useful for some third-party backup applications that can take snapshots but do not communicate to PI via the Volume Shadow Copy Service API. **-SimulateVSS -BackupStart2Freeze** suspends database writes. The simulated backup is aborted if it is not it is not ended within 60 seconds. You can use **-SimulateVSS -Thaw2PostSnapshot** in conjunction with **-SimulateVSS –BackupShutdown** to end the simulated VSS backup.
-test –freeze *component*	Freeze the specified component. For example, **piartool –backup -test –freeze -pibasess** prevents any modification to the point database, module database, digital states, and so on. A complete list of components is available with the **piartool -backup –identify -verbose** command.

Secondary Backup Arguments	Description
-trace *level*	Control writing of debug messages to the log file. By default, the trace level is 0, resulting in no trace messages. Higher values of trace level result in more trace messages. Trace levels higher than 100 currently result in no more messages than a trace level of 100. Normally, tracing should be off to avoid unnecessary messages in the log file. If the trace level is non-zero, the trace level is displayed in the **piartool –backup –query** command. Example: `piartool –backup –trace 1`

Restoring Configuration and Archive Files using NTBackup.exe

NTbackup is available for Microsoft Windows Server 2003 and Microsoft Windows XP. By default the PI backup subsystem does not use **NTBackup** to back up files. You can use **NTBackup.exe** to restore the files that were backed up using **NTBackup**.

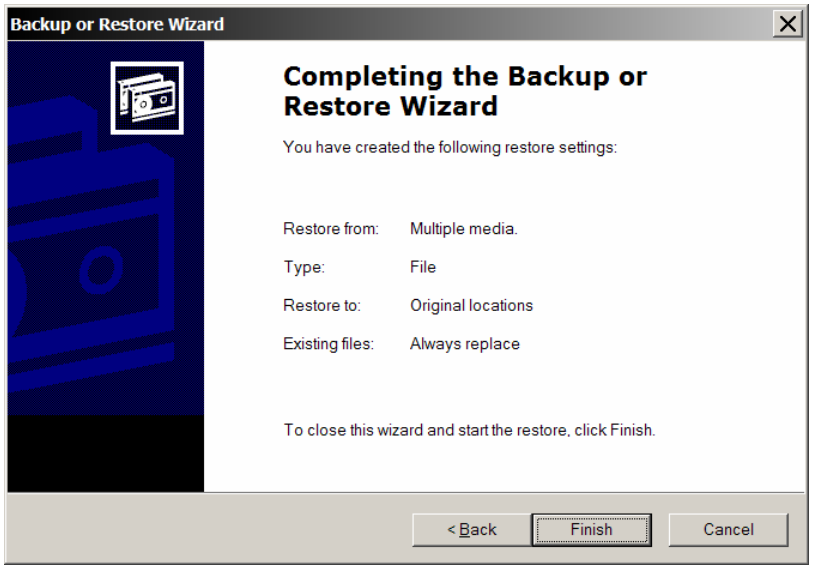
Start the **NTBackup.exe** restore wizard by entering the command:

```
ntbackup
```

or double-click the `.bkf` file that was created by **NTBackup.exe** on the primary PI Server and moved to the secondary PI Server.

The first time you start **NTBackup.exe** by double-clicking a `.bkf` file, it opens the file as expected. On subsequent times that you start **NTBackup.exe** by double-clicking a `.bkf` file, you may have to click **Browse** to browse to the file.

The **NTBackup.exe** wizard has some default behaviors that must be overridden. By default, **NTBackup.exe** does not overwrite files. If the file structure on the secondary PI Server is consistent with the file structure on the primary PI Server, click **Advanced**. In the dialog box that appears, select options to unpack the `.BKF` file created by the `PIBackup.bat` command file and overlay the files on top of the respective PI directories for `.DAT` files, event queues, and archive files.



Advanced NTBackup.exe Settings which Restore PI Files to Original Locations

If the directory structure is different on the destination computer than the one that was used to create the .bkf file, use NTBackup.exe to unpack the .BKF file to a temporary directory, and then copy the files to the appropriate directories using the windows GUI or a command file. This will be needed to initialize or re-initialize secondary PI Servers in a collective that have different directory structures than the primary PI Server.

Technical Support and Resources

Contact Rockwell Automation Technical Support at the following:

- Customer Support Telephone — 1-440-646-3434
- Online Support — http://support.rockwellautomation.com

Knowledge Center

The Knowledge Center provides a searchable library of documentation and technical data, as well as a special collection of resources for system managers. For these options, click **Knowledge Center** in the Technical Support Web site.

- ❑ The Search feature allows you to search Support Solutions, Bulletins, Support Pages, Known Issues, Enhancements, and Documentation (including user manuals, release notes, and white papers).
- ❑ System Manager Resources include tools and instructions that help you manage: archive sizing, backup scripts, daily health checks, daylight savings time configuration, PI Server security, PI System sizing and configuration, PI trusts for interface nodes, and more.

Before You Call or Write for Help

When you contact Rockwell Automation Technical Support, please provide:

- ❑ Product name, version, and/or build numbers
- ❑ Computer platform (CPU type, operating system, and version number)
- ❑ The time that the difficulty started
- ❑ The message log(s) at that time

Find the Version and Build Numbers

To find version and build numbers for each PI System subsystem (which vary depending on installed upgrades, updates or patches) use either of the following methods:

- ❑ If you have PI System Management Tools (PI SMT) installed, choose **Start > Programs > PI System > PI System Management Tools**. In PI SMT, select the server name, then under **System Management Plug-Ins**, open **Operation > PI Version**. The PI Version tree lists all versions.

- If you do not have PI SMT installed, open a command prompt, change to the `PI\adm` directory, and enter **`piversion -v`**. To see individual version numbers for each subsystem, change to the `PI\bin` directory and type the subsystem name followed by the option **`-v`** (for example, **`piarchss.exe –v`**).

View Computer Platform Information

To view platform specifications:

- In Windows, right-click **My Computer** and choose **Properties**. For more detailed information, choose **Start > Run,** and enter **msinfo32.exe**
- In UNIX, use the command, **`uname -a`**

Index