Honeywell

Experion PKS
TPN Server User's Guide

EPDOC-X143-en-431A
February 2015

Release 431

Honeywell

Document	Release	Issue	Date
EPDOC-X143-en-431A	431	0	February 2015

Disclaimer

Contents

1 About This Document

This document describes how to use the TotalPlant\ Network (TPN) Server, primarily from the server perspective. It also covers information about server name forms and data types from the client perspective.

The primary user tasks are installation, configuration, and operation (startup, checkpoint and shutdown), of the TPN Server. You can access and view the System Management Status Display as part of TPN Server configuration and operation.

Attention

This document assumes that you have already installed System Management components. Refer to the *System Management Configuration Guide* if you need to install the System Management Display.

Revision	Date	Description
A	February 2015	Initial release of the document.

2 Introduction

Related topics

“Overview” on page 10
“TPN Server introduction” on page 11

2.1 Overview

2.1.1 Purpose of this guide

This guide describes how to use the TotalPlant\ Network (TPN) Server, primarily from a server perspective, primarily when operating on the Experion APP node. It also describes server name forms and data types from a client perspective. The primary user tasks are installation, configuration, and operation (startup, checkpoint and shutdown), of the TPN Server. You can access and view the System Management Status Display as part of TPN Server configuration and operation. In addition, refer to the following documents.

- *Integrated Experion-TPS User's Guide*
- *System Management Configuration Guide*
- *Experion Windows Domain/Workgroup Implementation Guide*

Prerequisite skills

This guide assumes that you have experience in Microsoft Windows 2003, 2008, and 7 Professional administrative tasks and a working knowledge of Experion system concepts and TPN concepts and requirements.

How to use this guide

This guide is organized into major sections that represent major tasks.

Task	Go to section
Verify TPN Server component installation	“Verifying the TPN Server installation” on page 13: “Verifying the TPN Server installation” on page 13
Configure TPN Server as a component	“Verifying the TPN Server installation” on page 13: “Configuring the TPN Server” on page 17
Configure AutoCheckpointing	“Managing the TPN Server cache” on page 53: “Managing the TPN Server cache” on page 53
Copy cache between TPN Servers	“Managing the TPN Server cache” on page 53: “Managing the TPN Server cache” on page 53
Start, stop and view status of TPN Server	“Monitoring TPN Server status” on page 59: “Monitoring TPN Server status” on page 59
Create a client application or interpret server name forms and data types for TPN Client	“Accessing TPN data using the TPN Server” on page 87: “Accessing TPN data using the TPN Server” on page 87

2.2 TPN Server introduction

Related topics

“TPN Server functional description” on page 11
“Functionality supported in this guide” on page 11

2.2.1 TPN Server functional description

The TPN Server is Honeywell's method of allowing the OLE for Process Control (OPC) client to access the TPN parameter data. The TPN Server provides data access to Honeywell Communications Interface (HCI)/OPC clients. This data is transferred over open interfaces defined by the OPC specification. The sole purpose of the TPN Server is to access TPN data that can be read, written, or scanned in an OPC-compliant manner. Developers with proficiency in OPC and HCI can use the TPN Server to access TPN (LCN) data for their applications.

2.2.2 Functionality supported in this guide

This guide provides information to do the following:

- Verify the TPN Server installation
- Configure the TPN Server
- Manage the TPN Server cache
- Operate the TPN Server to start up, view status displays, checkpoint, and shut down
- Create a TPN Server client

3 Verifying the TPN Server installation

Related topics

“How to verify a TPN Server installation” on page 14

3.1 How to verify a TPN Server installation

Related topics

“Who should use this section” on page 14
“What the TPN Server package represents” on page 14
“Verification of a successful installation” on page 14

3.1.1 Who should use this section

This section is intended for the system administrator or others who must verify installation of the TPN Server software package on their system. If TPN Server is already installed and functioning on your system, you may skip this section.

3.1.2 What the TPN Server package represents

The TPN Server is included within the Experion software DVD media. Installation of any TPN connected Experion Node (ES-T, ESVT, ACE-T, Experion APP) automatically includes and configures TPN Server.

The separate TPN Server connectivity Client package Experion APP TPS Client must be installed on all external third-party nodes that need to access the TPN Server. The TPN Servers loaded will run only on nodes that contain an installed LCNP board. Currently, there can be only one TPN Server per LCNP board.

3.1.3 Verification of a successful installation

Upon successful installation of the Experion TPN connected node, your selected drive directory should include a folder named '\Program Files\Honeywell\TPS\TPNServer' with contents similar to the items shown in Figure 1 that follows. The Program Files\Honeywell\HCI subdirectory contains the HCI Runtime software.

Attention

With Experion R410, 64-bit OS is supported. All 32-bit components are installed in 'C:\Program Files (x86)'. The default path for installation of an Experion TPN connected node is 'C:\Program Files (x86)\Honeywell\TPS \TPNServer'.

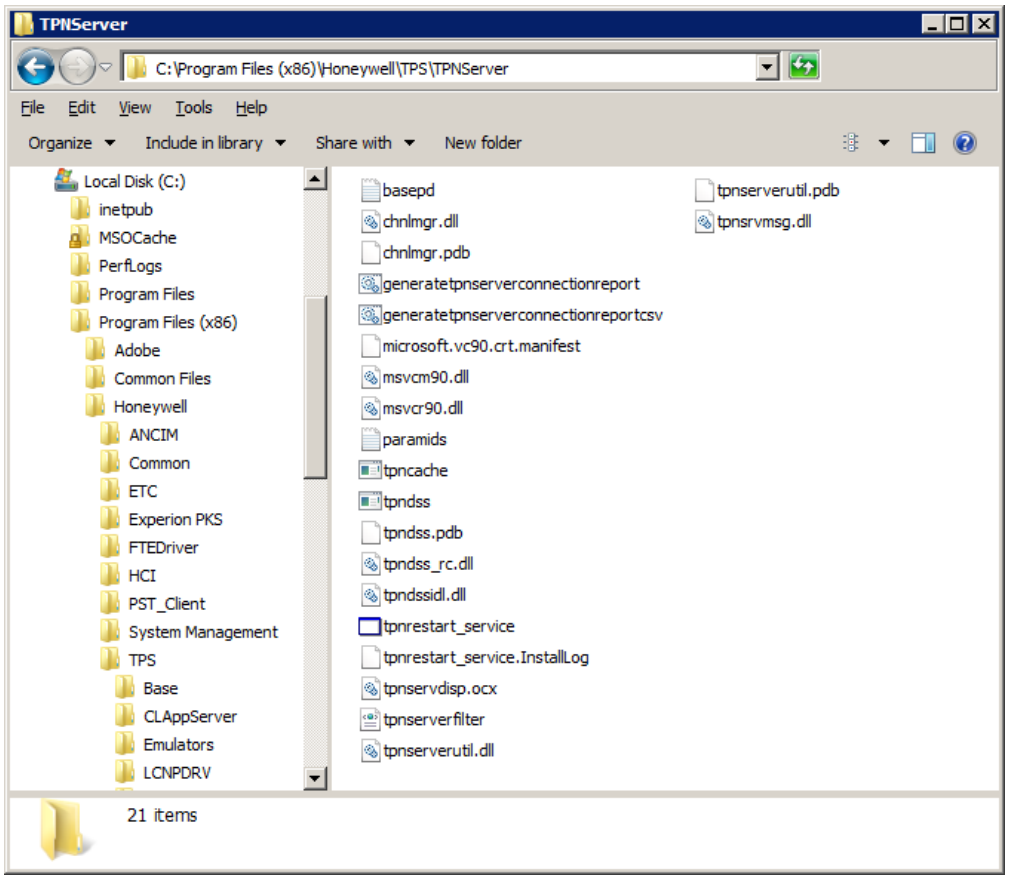


Figure 1: Example TPN Server Folder Content

4 Configuring the TPN Server

Related topics

4.1 Accessing the HCI Configure Component page

4.1.1 Procedure overview

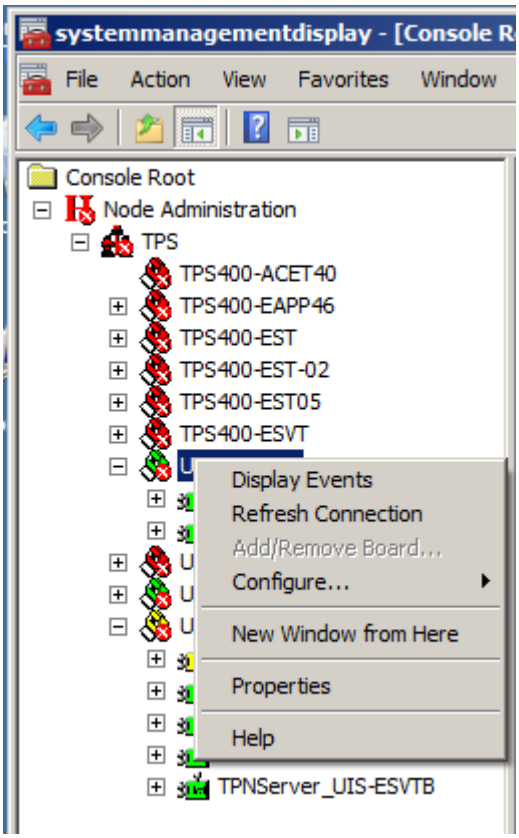
Attention

TPN Servers on LCN-connected Experion nodes (including Experion APP) are installed with a default configuration —no user configuration is required. The procedure below is included in cases when default configuration needs to be modified.

The HCI Configure Component page operates within the System Management display that is implemented as a Microsoft Management Console (MMC) snap-in. The HCI Configure Component page is also accessible from the Configuration Utility. Use either of the following procedures to access the HCI Configure Component page.

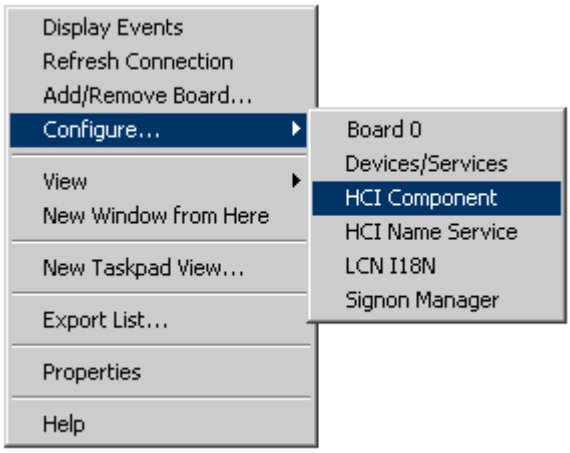
4.1.2 Procedure using System Management display

1 On the **System Management** display, right-click the computer item that represents the node containing the installed TPN Server package.



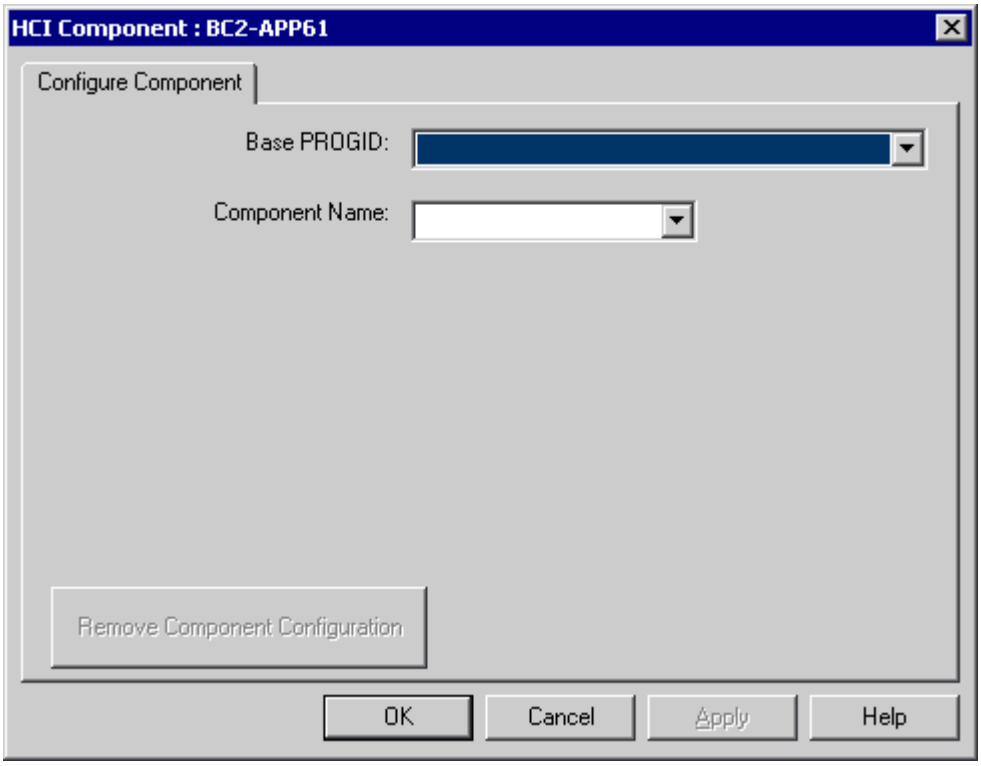
A shortcut menu appears.

2 Click **Configure** > **HCI Component**.



Note: A dimmed menu item indicates that component is not available on that node.

The **HCI Configure Component** page appears from which you can configure managed components such as the TPN Server.

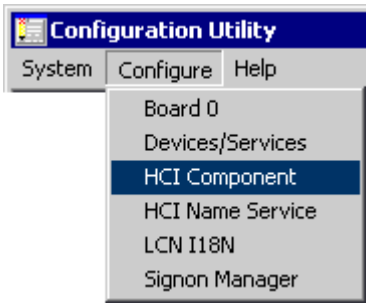


4.1.3 Procedure using Configuration Utility

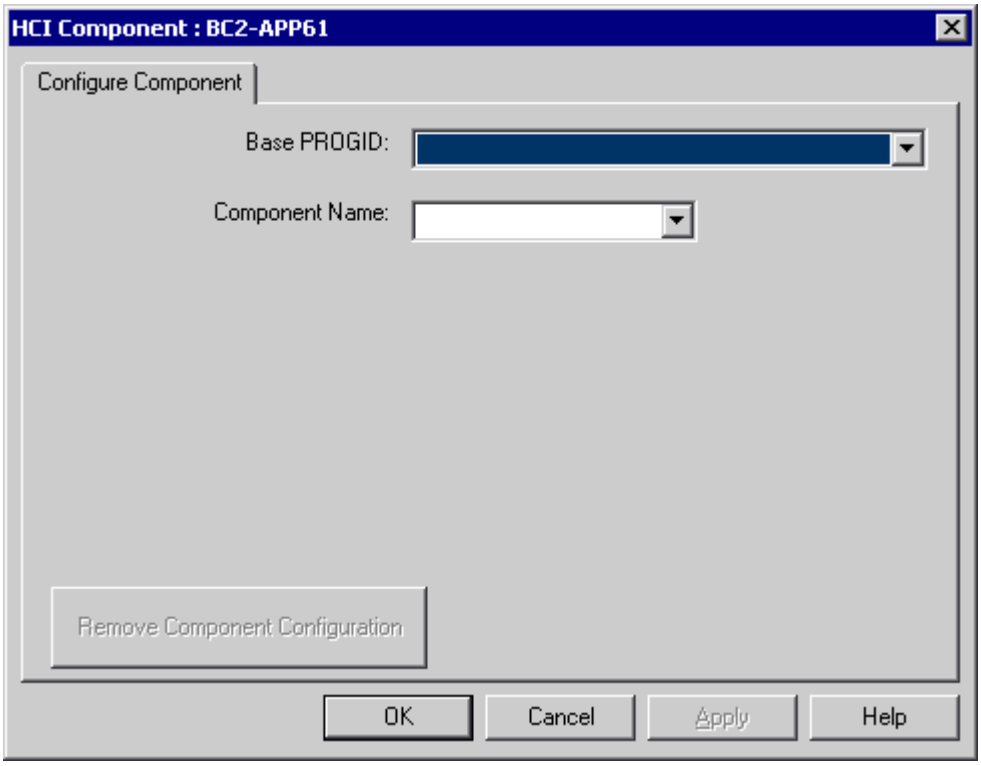
1 Choose **Start Menu** > **All Programs** > **Honeywell Experion PKS** > **System Management** > **Configuration Utility**.

The **Configuration Utility** menu appears.

2 Click **Configure** > **HCI Component**.



The **HCI Configure Component** page appears from where you can configure managed components such as the TPN Server.



4.2 Configuring TPN Server from HCI Component page

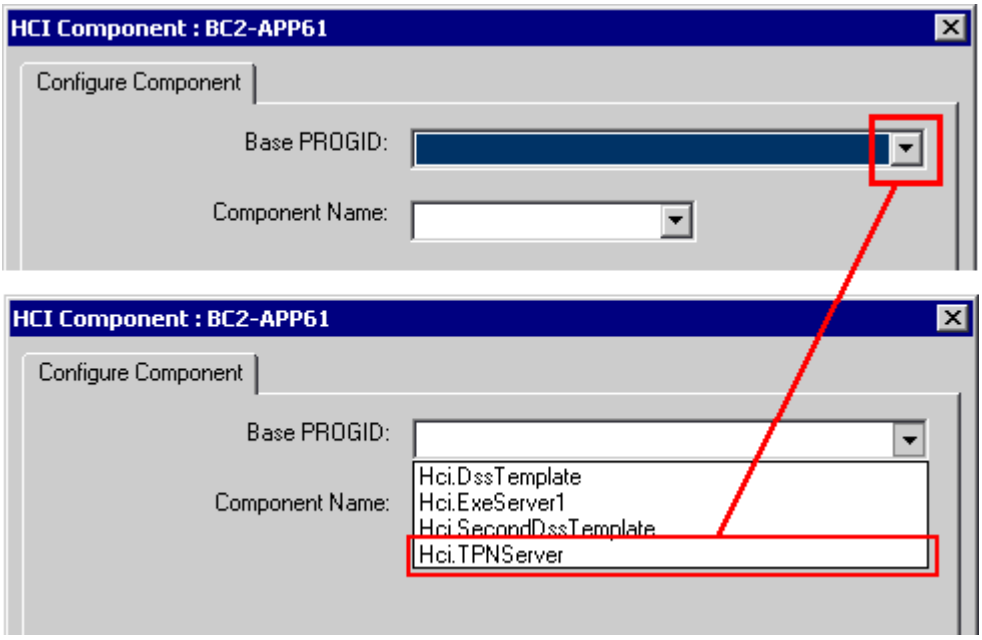
Related topics

“Procedure for using the HCI Configure Component page” on page 21
“Procedure for setting Data Access Options for a TPN Server” on page 22

4.2.1 Procedure for using the HCI Configure Component page

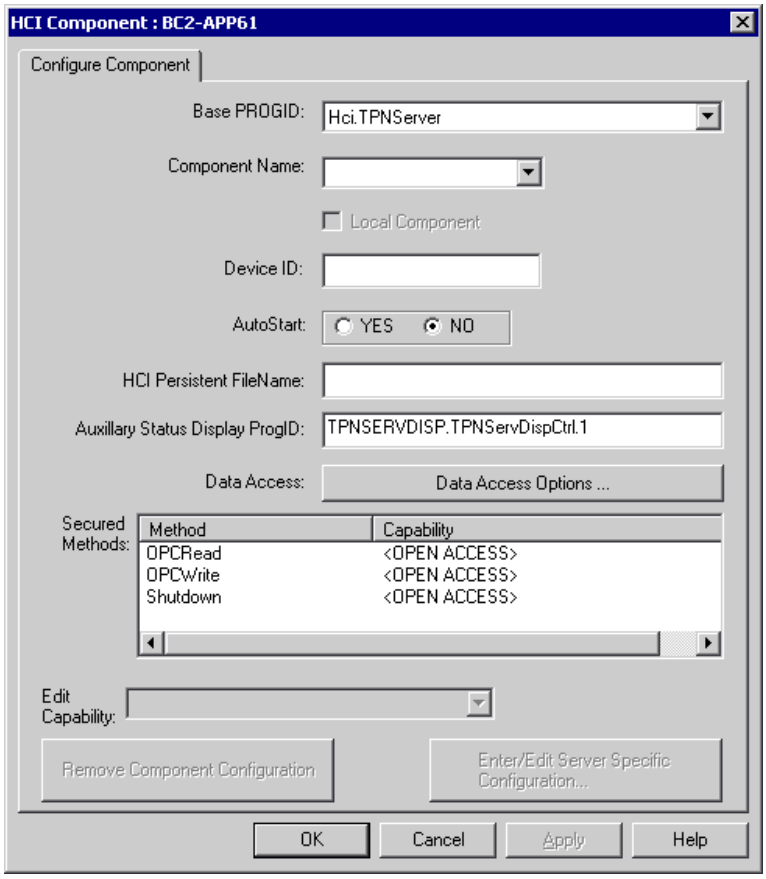
Use the **HCI Configure Component** page to configure a TPN Server.

1 From the **HCI Configure Component** page, click the down arrow in the **Base PROGID** field to open a list and then click **Hci.TPNServer**.

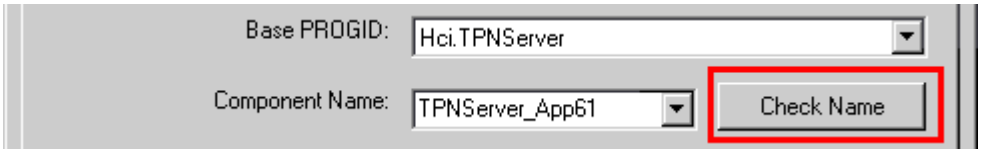


The configuration page displays TPN Server properties.

2 For a new configuration, type the TPN Server's name in the **Component Name** field.



A **Check Name** button appears next to the **Component Name**.



3 For a new configuration, click **Check Name** to validate the new **Component Name** for your TPN Server.
- Select the **AutoStart NO** option if the TPN Server is to be manually started at every system restart.
- Select the **AutoStart YES** option if the TPN Server is to be automatically started at every system restart.

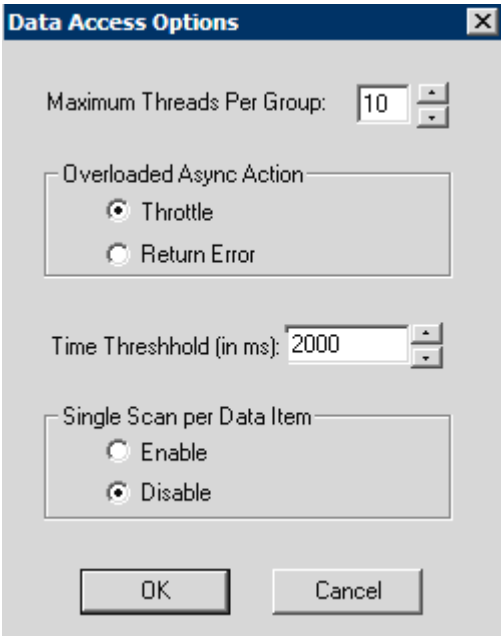
4 Accept the default **HCI Persistent Filename** if you do not need to change the filename. The file represents a checkpoint file for your TPN Server.

5 Do not change the **Auxiliary Status Display ProgID** from the default.
The entry represents a status display used to monitor your TPN Server.

6 Continue to the next procedure.

4.2.2 Procedure for setting Data Access Options for a TPN Server

1 From the **HCI Configure Component** page, click **Data Access Options**.
The **Data Access Options** window appears.



The **Data Access Options** window is used to set data access options on HCI OPC Servers. For details regarding the runtime functionality related to these options, see the *System Management Configuration Guide*.

2 The **Maximum Threads Per Group** option sets the number of outstanding asynchronous threads per group. This is used to prevent any one OPC group from using all the system resources.

Use the up and down arrows to adjust the maximum threads per group, or type an appropriate value.

3 The **Overloaded Async Action** option specifies the action to be taken when the maximum number of threads is exhausted.

- Select **Throttle** to wait until a thread completes before processing the max+1 asynchronous thread.
- Select **Return Error** to receive an error condition from the asynchronous call.

4 The **Time Threshold** option specifies the time in milliseconds to wait before a background device read is performed. It is the time that will pass after the items are set to active. If the polling thread has not made an update within the specified time period, then a background device read is issued.

Use the up and down arrows to adjust the time, or type an appropriate value between '500' and '10,000' milliseconds.

5 The **Single Scan per Data Item** option (if enabled) ensures that a data item is scanned into a server cache only once per scan cycle independently of the number of active groups in which it is a member, using the Active List functionality.

Select the **Enable** or **Disable** option as appropriate. This option should be applicable on Experion APP nodes only; do not enable **Single Scan** on other Experion T nodes (ES-T, ESVT, and ACE-T).

For more information about this option, see ' “Disabling/enabling the Single Scan per Data Item data access option” on page 88.'

6 After you have made your selections, click **OK**.

7 Continue to the next section.

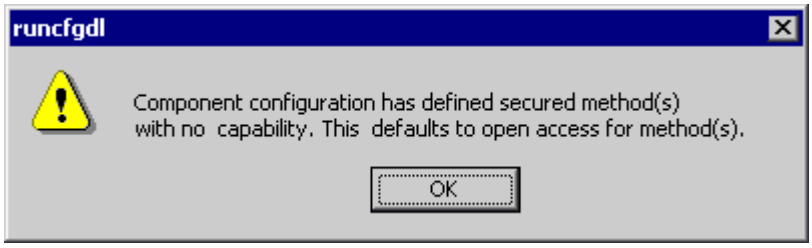
4.3 TPN Server secured method access

To change the capability file (proxy file) associated with a secured method, select the appropriate method in the **Secured Methods** list box on the **HCI Configure Component** page.

Attention

Installation of the Experion LCN connected nodes sets default access permissions that are appropriate for normal operation of the TPN Server on these nodes.

The capability field can be set to blank. A secured method without a defined capability is interpreted at runtime to mean that everyone has access to use the method. Upon detection of undefined (or blank) capability names associated with one or more secured methods, the configuration page issues the following warning dialog during any write operation that secured method(s) have open access.



4.3.1 Assumptions about method descriptions

Several assumptions are made in the following descriptions of secured method access:

- As a member of the Product Administrators group, you have read the complete discussion in the 'Customizing HCI/OPC Server Capability (Proxy) Files' section of the *Experion System Administration Guide*.
- The standard TPS capability file's Access Control Lists (ACLs) have not been altered.
- The standard TPS capability files are used to determine access levels. The access level section of the TPN Server Security configuration page is shown in “Figure 2: Standard TPS Capability Files”.

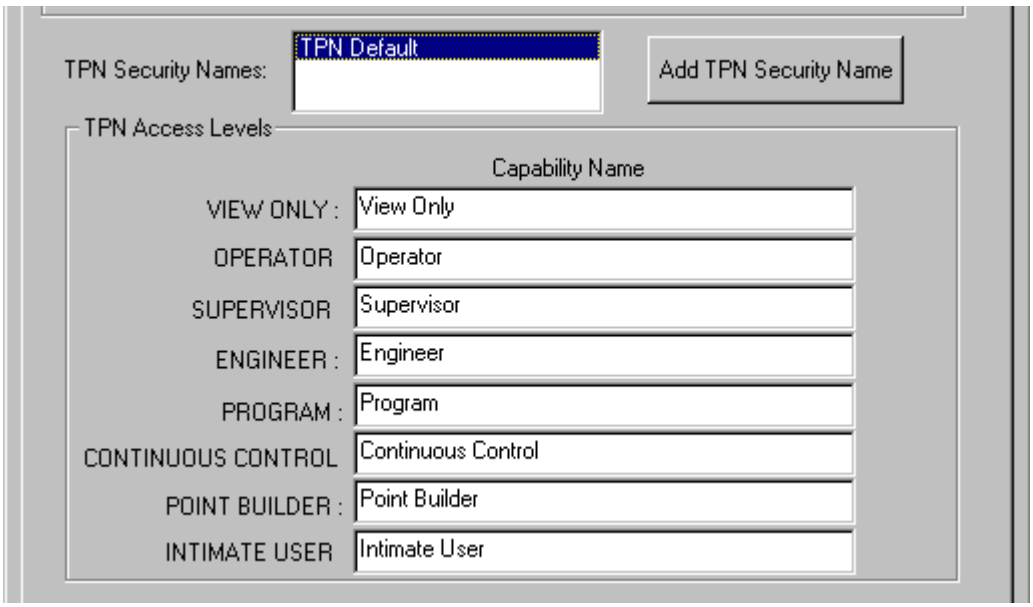


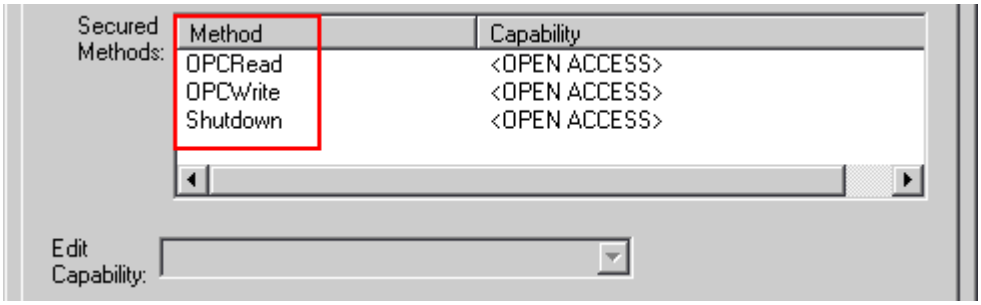
Figure 2: Standard TPS Capability Files

4.3.2 Procedure for assigning capability file to secured method

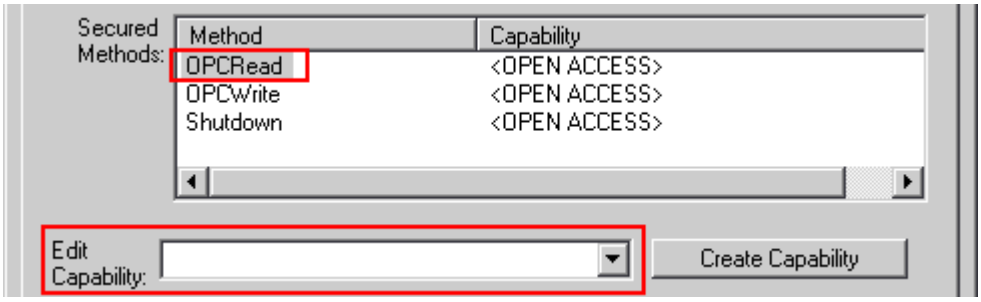
This procedure shows how to assign a capability to a secured method. Refer to the subsequent sections for details and example scenarios for secured method access.

1 On the **HCI Configure Component** page, select a **Method** to assign a **Capability** file.

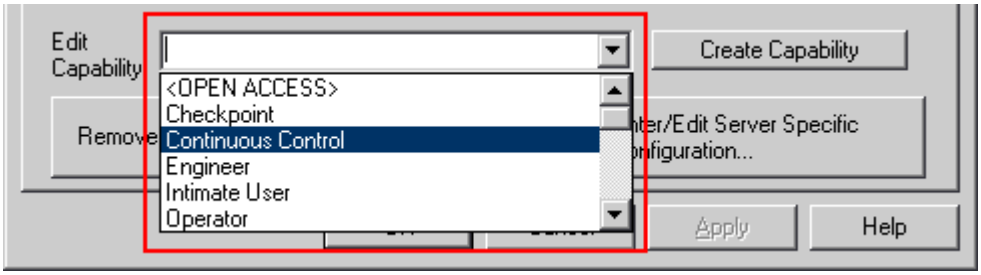
The following example shows several **Methods** that are unsecured because **<OPEN ACCESS>** appears as the **Capability**.



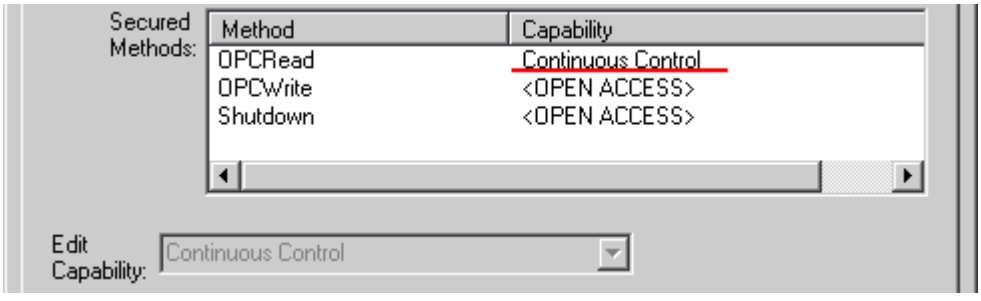
The **Edit Capability** field and menu are enabled.



2 Select a **Capability** file to secure the selected **Method**.



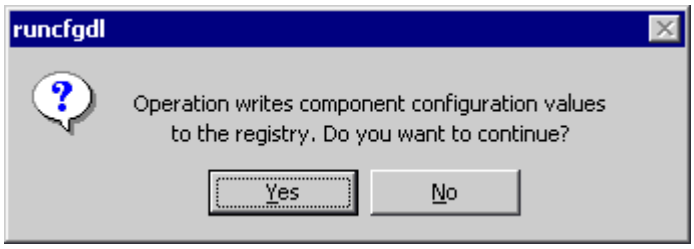
The secured **Method** displays the assigned **Capability** file.



3 Repeat the previous steps for remaining secured methods that require capability files.

4 Click **OK** or **Apply**.

A dialog appears stating that the node's registry will be updated.



5 Click **Yes**.

6 Continue to the next section.

4.3.3 What is TPN Server method access security configuration?

The HCI/OPC server capability files are used to determine who has the capability to use secured TPN Server methods. Product administrators use the **HCI Configure Component** page shown in "Figure 3: Secured Methods and Capability Selection" to secure methods by selecting one of the capability files. The files are located in the C:\ProgramData\Honeywell\ProductConfig\Security folder of the node that hosts the TPN Server.

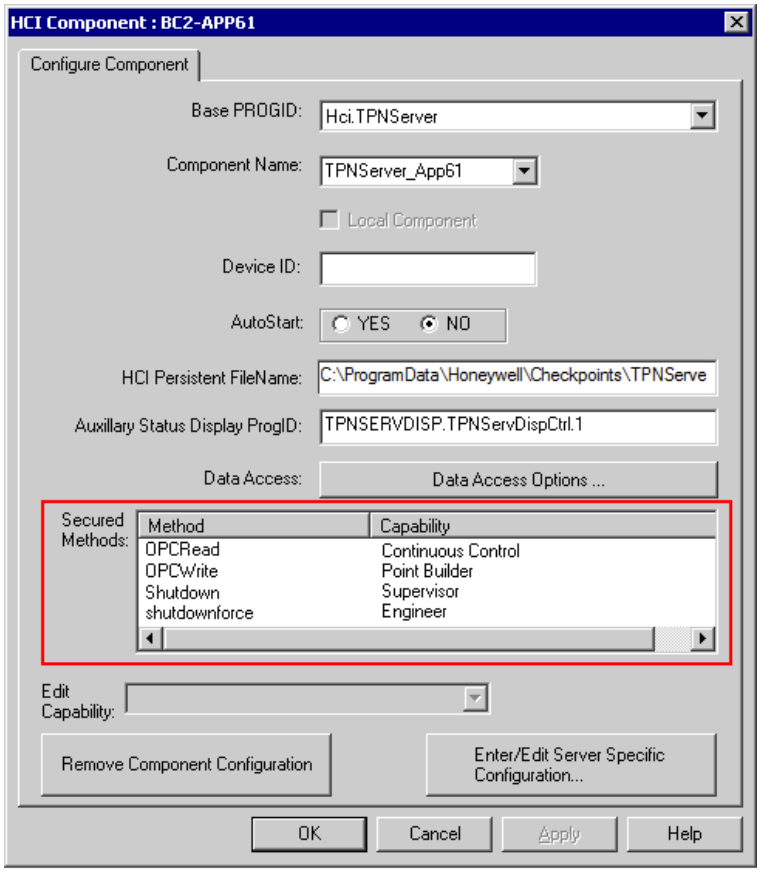


Figure 3: Secured Methods and Capability Selection

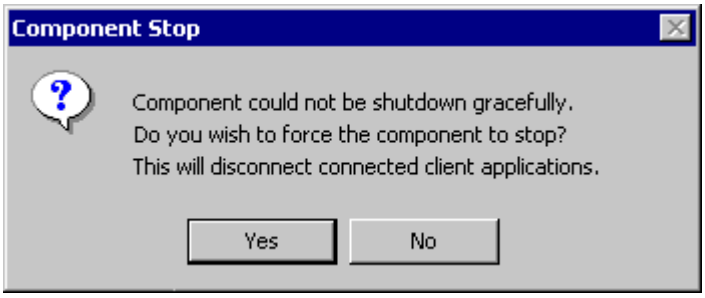
Note that in "Figure 3: Secured Methods and Capability Selection", the Product administrator secured the TPN Server methods as follows:

- The **Continuous Control** capability file secured the **OPCRead** method.
- The **Point Builder** capability file secured the **OPCWrite** method.

- The **Supervisor** capability file secured the **Shutdown** method.
- The **Engineer** capability file secured the **shutdownforce** method.

4.3.4 Shutdown and Shutdownforce

If a user attempts to manually shut down a component that has connected client applications, the following dialog appears:



Clicking **YES** to confirm the dialog causes a forced shutdown ('Shutdownforce') to be attempted. If a user is not permitted to force a shutdown, the System Management Display will ignore that user's shutdown command. If a user has permissions assigned through the capability file, the forced shutdown will occur. If no client applications are running and a shutdown is attempted, members of the capability file assigned to the shutdown method will be permitted to shut down the TPN Server.

4.3.5 TPN Server method access security per standard capability files

Refer to the 'HCI/OPC Server Capability Files' section in the *Experion System Administration Guide*.

4.3.6 TPN Server method access security per customized capability files

Refer to the 'HCI/OPC Server Capability File Customization' section in the *Experion System Administration Guide*.

4.4 OPCRead and Shutdown method access

The following is an overview of OPCRead and Shutdown method access:

- The OPCRead and Shutdown methods are secured with a capability file.
- Each capability file has a defined Access Security Level that is determined by the file's Access Control List.
- An HCI client can access an OPCRead or Shutdown method if the account running the HCI client has RX access permission to the capability file that is securing the method.
- If the account does not have RX access permission to the capability file, an access denied error is returned to the HCI client.

Attention

The Client Access Level described later in the ' “OPCWrite method access” on page 29' section is not used to determine access to the OPCRead and Shutdown methods—only the account running the HCI client is used to determine access to these methods.

4.4.1 Access approved example

The HCI client is running under an account that is a member of the 'Local Supervisors' group.

The HCI client attempts to access the Shutdown method, which is secured with the Operator capability file.

The TPN Server checks if the account running the HCI client (a member of the 'Local Supervisors' group) has RX access permission to the capability file that secures the method (the Operator capability file).

The 'Standard Capability File ACL Entries' table shows that the 'Local Supervisors' group *does* have RX access permission to the Operator capability file.

The HCI client is allowed to access the Shutdown method.

4.4.2 Access denied example

The HCI client is running under an account that is a member of the 'Local Supervisors' group.

The HCI client attempts to access the Shutdown method, which is secured with the Point Builder capability file.

The TPN Server checks if the account running the HCI client (a member of the 'Local Supervisors' group) has RX access permission to the capability file that secures the method (the Point Builder capability file).

The 'Standard Capability File ACL Entries' table shows that the 'Local Supervisors' group *does not* have RX access permission to the Point Builder capability file.

The HCI client is denied access to the Shutdown method and an access denied error is returned to the HCI client.

4.5 OPCWrite method access

Attention
The Client Access Level is only used to determine OPCWrite method access—it is not used for OPCRead and Shutdown method access.

4.5.1 Inheriting a default HCI Client Access Level

All HCI client applications inherit the Default Access Level as their Client Access Level when they first bind to the TPN Server. See the ' “Assigning default access and priority levels” on page 34' section for information about setting the Default Access Level.

4.5.2 Possible default access level settings

“Table 1: Default Access Level Settings” describes the default access level settings.

Table 1: Default Access Level Settings

Client access level	Resulting method access level
Point Builder	Highest Available Method Access (Client Access Level 7)
Continuous Control	Client Access Level 6
Program	Client Access Level 5
Engineer	Client Access Level 4
Supervisor	Client Access Level 3
Operator	Client Access Level 2
View Only	Lowest Method Access (Client Access Level 1)

Attention
Note that the highest Client Access Level (Intimate User Level 8) cannot be chosen as the default.

Tip
Any HCI client of the TPN Server, such as Experion Server, OPC Gateway or GUS HCI Client, has the capability to override the default access level when performing write operations.

4.5.3 Changing from the inherited Client Access Level

All clients initially inherit the Default Access Level as their Client Access Level when they first bind to a TPN Server. Any HCI client of the TPN Server, such as Experion Server, OPCGateway or GUS HCI Client, has the capability to, and does override the default access level when performing write operations.

A Client Access Level change request is approved if the account running the GUS HCI client has RX access permission to the capability file that represents the newly requested level. If the account does not have RX access permission, an access denied error is returned to the GUS HCI client.

4.5.4 Accessing the OPCWrite method

The OPCWrite method access is achieved after passing the following access checks:

- **Access Check 1**—The TPN Server checks if the account running the HCI client has RX access permission to the capability file that secures the OPCWrite method. If the account *does not* have RX permission, then an access denied error is returned to the HCI client. If the account *does* have RX permission, then Access Check 2 will be performed.
- **Access Check 2**—The OPCWrite request and the HCI client's current Client Access Level are sent from the TPN Server to the data owner (ex. the NIM). If the data owner determines that the HCI client's Client Access Level is adequate, then the HCI client will be allowed to write data. If the data owner determines that the HCI client's Client Access Level is inadequate, then an access denied error would be returned to the HCI client.

Write approved example

The HCI client is running under an account that is a member of the 'Local Supervisors' group.

The HCI client attempts to access the OPCWrite method, which is secured with the Supervisor capability file.

The TPN Server checks to see if the account running the HCI client (a member of the 'Local Supervisors' group) has RX access permission to the capability file that secures the method (the Supervisor capability file).

The 'Standard Capability File ACL Entries' table shows that the 'Local Supervisors' group *does* have RX access permission to the Supervisor capability file.

Access Check 1 has been passed, so execution will proceed to Access Check 2.

The TPN Server Default Access Level is set to 'Engineer.'

The HCI client has bound to the TPN Server and inherited 'Engineer' as its Client Access Level.

The OPCWrite request and the HCI client's current Client Access Level ('Engineer') is sent from the TPN Server to the data owner (such as the NIM).

Assume that the NIM allows HCI clients with an 'Engineer' Client Access Level to write data.

The data is written to the NIM.

Write denied example

The HCI client is running under an account that is a member of the 'Local Supervisors' group.

The HCI client attempts to access the OPCWrite method, which is secured with the Supervisor capability file.

The TPN Server checks if the account running the HCI client (a member of the 'Local Supervisors' group) has RX access permission to the capability file that secures the method (the Supervisor capability file).

The 'Standard Capability File ACL Entries' table shows that the 'Local Supervisors' group *does* have RX access permission to the Supervisor capability file.

Access Check 1 has been passed, so execution will proceed to Access Check 2.

The TPN Server Default Access Level is set to 'Engineer.'

The HCI client has bound to the TPN Server and inherited 'Engineer' as its Client Access Level.

The OPCWrite request and the HCI client's current Client Access Level ('Engineer') is sent from the TPN Server to the data owner (such as the NIM).

Assume that the NIM does not allow HCI clients with an 'Engineer' Client Access Level to write data.

An access denied error is returned to the HCI client.

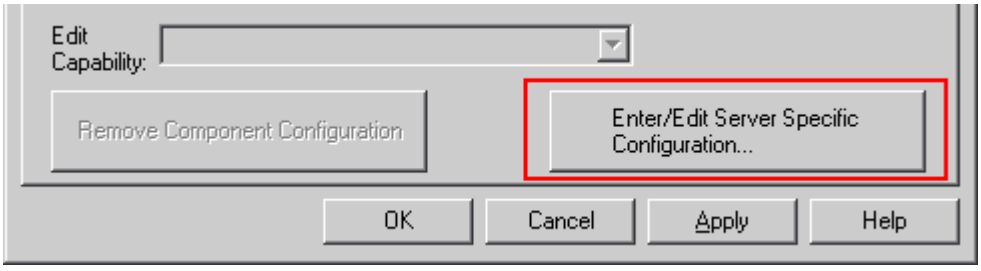
4.6 Editing the server-specific configuration

Related topics

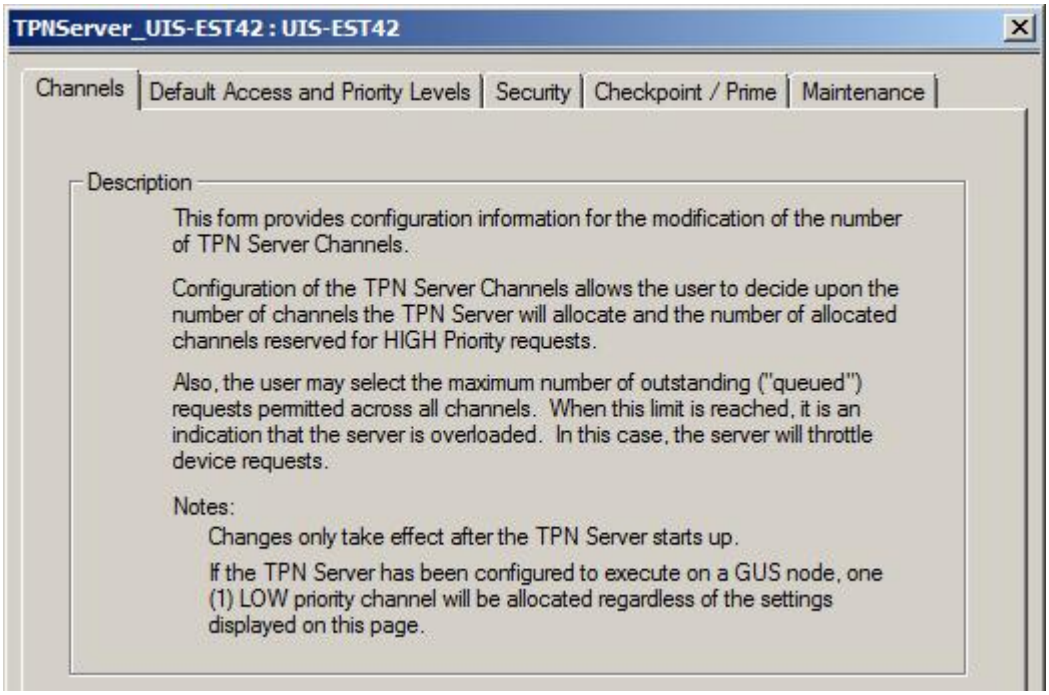
"Procedure for accessing TPN Server configuration pages" on page 31

4.6.1 Procedure for accessing TPN Server configuration pages

1 From the **HCI Configure Component** page, click **Enter/Edit Server Specific Configuration**.



Result: The server-specific configuration page appears.



2 Continue to the next section.

4.7 Configuring TPN Server channels

Related topics

“TPN Server Channels page” on page 32
“Channels To Be Allocated setting” on page 32
“High Priority Channels setting” on page 33
“Outstanding Requests setting” on page 33

4.7.1 TPN Server Channels page

From the server-specific configuration page, click the **Channels** tab to open the view shown in “Figure 4: TPN Server Channels Page”.

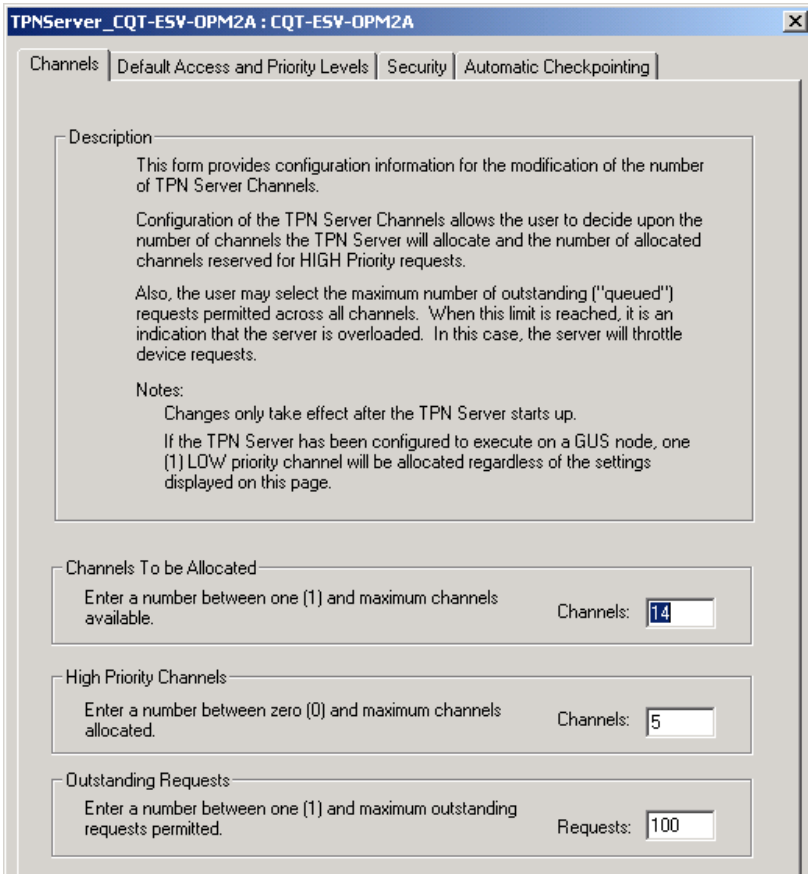


Figure 4: TPN Server Channels Page

4.7.2 Channels To Be Allocated setting

The number of channels allocated for use by the TPN Server and the number of allocated channels reserved for HIGH priority requests is user configurable.

A maximum of 30 TPN channels are available to all Windows applications. The number of channels that can be allocated by the TPN Server can be set from '1' to the maximum number of channels currently available. Note that setting changes take effect only after the TPN Server is restarted.

Task: Type the appropriate number in the **Channels To Be Allocated** text box and then continue to the next section.

4.7.3 High Priority Channels setting

The number of channels allocated is divided into two priority levels:

- HIGH for high priority requests
- LOW for low priority requests

The **High Priority Channels** setting uses the following formula to determine the priority channel allocation:

Number of low priority channels = (Channels To Be Allocated setting) - (High Priority Channels setting)

The following should be noted:

- A setting of '0' HIGH priority channels specifies that all of the allocated channels will be used for both HIGH and LOW priority requests.
- If all allocated channels are configured as HIGH priority channels, then only HIGH priority requests will be serviced.
- The number of HIGH priority channels must be equal to or less than the number of allocated channels.
- You must configure at least one LOW priority channel and one HIGH priority channel (recommended).
- Setting changes take effect only after the TPN Server is restarted.

Task: Type the appropriate number in the **High Priority Channels** text box and then continue to the next section.

4.7.4 Outstanding Requests setting

The number of Outstanding Requests permitted, across all channels, is the maximum number of requests that can be queued on both the High and Low Priority Channels at a given time.

When 80% of the Outstanding Requests setting is reached, the 'Overloaded' message will start flashing on the auxiliary status display next to the **Number of Queued Requests** (above the # High and # Low Priority requests).

When 100% of the Outstanding Requests setting is reached, 'EFAIL' will be returned to any client requesting data from the server. Then, that server will throttle device requests until the sum of the queued requests drops below the number of Outstanding Requests permitted.

The number of Outstanding Requests has a default value of '100,' but the value can range from 1 to 500.

- Higher Outstanding Requests values may result in a client having to wait for data due to the large number of queued requests.
- Lower Outstanding Requests values will cause the maximum number of outstanding requests to be reached sooner, so the client will get the 'EFAIL' sooner and then continue to other tasks.

You should set the number of outstanding requests to '100.' If the server reaches an overload status, you should install another TPN Server on a different APP node. Note that setting changes take effect only after the TPN Server is restarted.

Task: Type the appropriate number in the **Outstanding Requests** text box and then continue to the next section.

4.8 Assigning default access and priority levels

Related topics

“Default Access and Priority Levels page” on page 34
“Default Access Level setting” on page 34
“Default Server Priority Level setting” on page 35
“Default Behavior for Uncertain Quality option” on page 35

4.8.1 Default Access and Priority Levels page

From the server-specific configuration page, click the **Default Access and Priority Levels** tab to open the view shown in “Figure 5: Default Access and Priority Levels Page”.

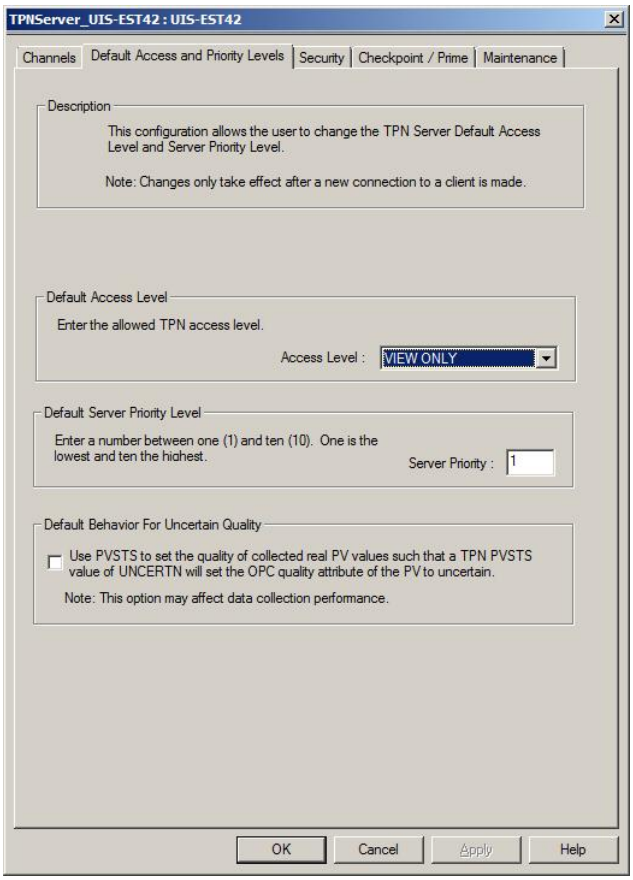


Figure 5: Default Access and Priority Levels Page

4.8.2 Default Access Level setting

The **Default Access Level** setting has **View Only** as the default level. This level will be used by an HCI or OPC client application that does not explicitly set the TPN Server access level.

All except the **Intimate User** access level can be configured as the default TPN Server access level. If an HCI or OPC client application must use the Intimate User access level, then the client application must set the TPN access level. For more information, see the ' “TPN Server secured method access” on page 24' section.

On an ESVT node, the OPC Integrator sets the access level to **PROGRAM** (default). If the client is writing to cascaded control points on the LCN then the default must be set to **Continuous Control**.

The **Default Access Level** setting on the TPN server is used by the ESVT as the security level for the writes from the OPC clients that are connected to the Experion OPC server.

Task: Select the appropriate **Default Access Level** and then continue to the next section.

4.8.3 Default Server Priority Level setting

The **Default Server Priority Level** setting has '1' as the default priority. The values range between 1 and 10, with '1' as the lowest priority level and '10' as the highest. This level will be used by an HCI or OPC client that does not explicitly set the TPN Server priority level.

Task: Type an appropriate number between 1 and 10 in the **Default Server Priority** text box and then continue to the next section.

4.8.4 Default Behavior for Uncertain Quality option

The **Default Behavior for Uncertain Quality** functionality is a new client option for TPN Servers. It requires LCN/TPN Release 680 or later.

Configuring the Uncertain Quality functionality

The two ways to configure the new functionality are *server-specific* and *client-specific*.

- **Server-specific configuration**—On the **Default Access and Priority Levels** page, selecting the **Default Behavior for Uncertain Quality** option enables the **Uncertain** quality of PV in the TPN Server to be returned when the PVSTS parameter is **UNCERTN**. In an LCN, the PVSTS parameter has three possible states—**NORMAL**, **UNCERTN**, and **BAD**. Until now, the TPN Server could only return two qualities of PV—**Good** or **Bad** (which are independent of the PVSTS value).

  This client option is *not* selected and current operation of the TPN Server, PVSTS parameter calculation, and error handling remains unchanged. Since the new functionality is optional, users can retain current functionality at any time. However, any change to enable or disable this option requires a restart of the TPN Server to activate the change, which is then propagated to all connected OPC clients.
- **Client-specific configuration**—An HCI client can explicitly set the Default Access Level and Default Server Priority Level by using the SetAttributes method of the IHciAttributes interface. The HCI client can use the same function call to set the new additional parameter. The TPN Server checks for this additional parameter and then acts according to the parameter provided by the client application. Upon successful operation of the SetAttributes call, the new settings will only be applied to the current client connection and the subsequent requests. Conversely, these settings will not be propagated to other connected clients.

The following sections further describe server-specific configuration. For details about client-specific configuration, see the ' “Client-specific configuration”' section.

Current functionality (option not selected)

Client applications, connected to the LCN through a TPN Server, can access LCN data points and their associated parameters. When client applications request a PV value, the LCN side displays the PV quality as **NORMAL**, **BAD**, or **UNCERTN**, as represented by the PVSTS parameter. However, the TPN Server is capable of returning only **Good** or **Bad**. Consequently, if the PV quality of a TPN point is **UNCERTN**, the TPN Server can only return **Good** for those LCN-connected applications. The new functionality addresses this inconsistency.

New functionality (option selected)

On startup, the TPN Server verifies if the LCN/TPN interface is at Release 680 or later. If verified, the **Uncertain** quality is activated; if not verified, the **Uncertain** quality is not activated and an error is returned.

The release number and status information is shown on “Figure 10: TPN Server Status Display—Main Page” as follows:

- **TPN Release** field shows the current release number of the LCN/TPN interface
- **PVSTS Quality Option Default** field shows the status of the **Uncertain** functionality

For OPC applications accessing the TPN Server, the new functionality returns **Uncertain** (instead of **Good**) as the OPC quality of a PV when PVSTS is **UNCERTN**. In operation, The TPN Server reads the real PV and PVSTS value (both contained in a blind record PVSTSREC) in one parameter fetch. The TPN Server uses the returned values to set the OPC value and OPC quality for the PV.

Any changes made to enable or disable this functionality are propagated to all connected OPC clients, but *only* after restarting the TPN Server. However, an OPC client can enable or disable this functionality by using the SetAttribute() and TPN_PROPERTY methods, as described later in the ' “Client-specific configuration”' section.

PVSTS parameter states

The PVSTS parameter can return **NORMAL**, **BAD**, or **UNCERTN** as follows:

- PVSTS as **NORMAL**:
  - PVSOURCE = Auto, PVAUTOST = Normal, and the PV value is within the range defined by PVEULO and PVEUHI
- PVSTS is **BAD** when the PV value becomes NaN for the following conditions:
  - PVSOURCE = Auto and PVAUTO = NaN
  - PVSOURCE = Auto, the value in PVAUTO is out-of-range, and it not clamped
  - PVSOURCE = Sub or Man and the PV is stored as NaN
- PVSTS is **UNCERTN** for the following conditions:
  - PVSOURCE = Man or Sub, and the PV value does not equal NaN (that is, a valid real number)
  - PVSOURCE = Auto, and PVAUTOST = UNCERTN (note that PVAUTOST contains UNCERTN if at least one of the required algorithm inputs is UNCERTN and none of the required algorithm inputs are BAD)
  - PVSOURCE = Auto, and the value in PVAUTO is clamped and outside the engineering-units range

Applicability of new functionality

The new functionality supports specific LCN points as identified in “Table 2: Supported LCN Points that have a PVSTS Parameter”.

Table 2: Supported LCN Points that have a PVSTS Parameter

Node	Box	Point Type
NIM	xPM	REGPV
NIM	xPM	REGCTRL w/CTLALGID of PID, PIDFF, PIDERFB, RATIOCTL, PIDPOSPROP, POSPROP, RAMPSOAK (exceptions NULL, INCRSUM, AUTOMAN, ORSEL, SWITCH, MULDIV, SUMMER, OPSELECT)
NIM	xPM	AI
AM	AM	REGCTRL
HG	MC	REGCTRL, AI, AC, COUNTER
HG	HLPIU	AI, AC
HG	CB	REGCTRL, AI
HG	LLPIU	AI
HG	LEPIU	AI

Node	Box	Point Type
HG	DHP	AC, AI
HG	EC	REGCTL

Client-specific configuration

Clients can enable or disable the new functionality according to their requirements by using either of the following methods—*SetAttributes* or *TPN_PROPERTY*.

- **SetAttributes method**

In the SetAttributes method, the user can control the new functionality by setting an additional attribute (as is currently done when setting the TPN access level and TPN priority level attributes). Clients must add this additional parameter; they will use the GetAttribute method to fetch the current flag status. The TPN Server checks this parameter when fetching the PVSTS value. The Attribute can be passed with other attributes in the IHciAttributes SetAttributes() function.

For the SetAttributes method,

STDMETHODIMP SetAttributes (DWORD dwNumAttr,

LPCWSTR *pszcAttrNames,

VARIANT *pvAttrValues,

HRESULT **ppErrors,

);

“Table 3: SetAttributes Parameter Specifications” describes the specifications for the SetAttributes parameter.

Table 3: SetAttributes Parameter Specifications

Parameter	Direction	Description
dwNumAttr	in	The number of server attributes in the array that are passed in, which is the number of elements in the returning arrays.
pszcAttrNames	in	The list of server attribute names (refer to “Table 4: Permissible Names for pszcAttrNames”) to be set that the client will pass to the TPN Server.
pvAttrValues	in	The list of permissible values to be set for the selected attribute name.
ppErrors	out	Array of HRESULTs indicating the success of the individual server attribute sets. The errors correspond to the names passed in pszcAttrNames. This indicates whether the HCI Server accepted the new value of the server attribute. Note that any FAILED error code indicates the HCI Server rejected the value.

Table 4: Permissible Names for pszcAttrNames

Server attribute name	Description
AccessLevel	Name for TPN access level setting.
PriorityLevel	Name for TPN priority level setting.
PVUncertainReq	Name for TPN Uncertain quality. See the following section for details.

PVUncertainReq permissible values

VT_BOOL—When a client is started, *and* the **Default Behavior for Uncertain Quality** option is selected, the PVUncertainReq becomes VARIANT_TRUE; otherwise, the value becomes VARIANT_FALSE.

A read through GetAttributes gets the value VARIANT_FALSE when the PV Uncertain option is disabled , and VARIANT_TRUE when the PV Uncertain option is enabled.

A write through SetAttributes of the value VARIANT_FALSE will disable the PV Uncertain option, and a write of VARIANT_TRUE will enable the PV Uncertain option.

- **TPN_PROPERTY method**

In the TPN_PROPERTY method, clients set the PV_UNCERTAIN option in the TPN Server. The TPN Server uses a special internally recognized point name and parameter name to change the Default Priority Level, Default Server Access Level, and Uncertain option. To read or store these values, they must be added to their own OPC group and not mixed with items to or from the TPN. The Uncertain option can be read or stored using the TPN_PROPERTY.PV_UNCERTAIN item with a value of TRUE or FALSE.

Error handling

If the client application supplies an invalid parameter for the new functionality, an error message such as 'No such interface supported' is returned to the client.

If the PVUncertainReq parameter value is set to TRUE during the SetAttribute() function call (or TPN_PROPERTY.PV_UNCERTAIN = TRUE), *and* the TPN Server has an error (such as TPN release below R680), an error is returned to the requestor.

Task: Select or clear the **Default Behavior For Uncertain Quality** option and then continue to the next section.

4.9 Determining TPN Server security

Related topics

“Security page” on page 39
“TPN Security Names” on page 39
“TPN Access Levels capability name” on page 40
“TPN Server Priority Levels capability name” on page 40

4.9.1 Security page

From the server-specific configuration page, click the **Security** tab to open the view shown in “Figure 6: Security Page”.

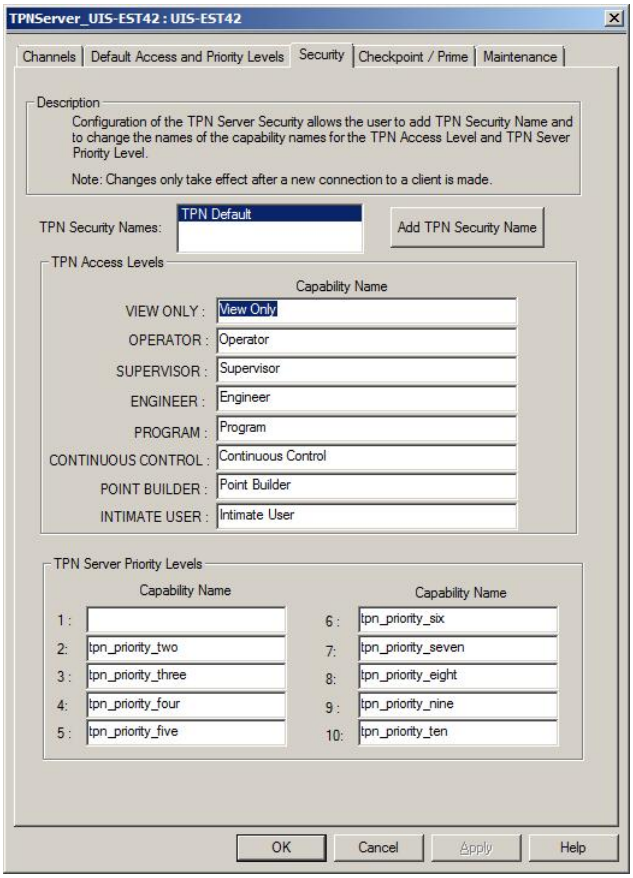


Figure 6: Security Page

4.9.2 TPN Security Names

Procedure for adding TPN Security Names

1 'TPN Default' is the default name in the **TPN Security Names** field.
The TPN Security name is normally the security name of the TPN System where the Experion nodes with TPN Servers are connected. The security policy on capability files for all TPN Servers on the same TPN System should be the same. However, you have the option to configure a TPN Server to have a different capability file with different security policy on the same TPN System by adding a new TPN Security name.

Click **Add TPN Security Name** to add a new TPN Security name.

Result: A new edit window appears to enter the new security name.

2 Type the new security name in the edit window. Then, click the name from the **TPN Security Names** window to display the capability names.

3 Continue to the next section.

4.9.3 TPN Access Levels capability name

This function is used when a user who is a member of the Product Administrators group wants to use non-standard capability files to secure TPN Server methods.

For detailed information about creating non-standard capability files, refer to the 'Customizing HCI/OPC Server Capability (Proxy) Files' section of the *Experion System Administration Guide*.

4.9.4 TPN Server Priority Levels capability name

This function is used for validating the HCI or OPC client requests to set the TPN Server priority levels, which are user-configurable. Ten capability files are available for securing the TPN Server priority level. They are located on the TPN Server's host node in the C:\ProgramData\Honeywell\ProductConfig\Security folder. Nine of the capability files and their associated ACLs are listed in “Table 5: Capability Names and Default ACLs”. **Note:** The tpn_priority_one file does not have a capability file name because it allows everyone to have access.

Table 5: Capability Names and Default ACLs

Capability name	Associated default ACL
tpn_priority_two	Allows an HCI or OPC client with View Only access or higher to set the TPN Server priority level.
tpn_priority_three	Allows an HCI or OPC client with Operator access or higher to set the TPN Server priority level.
tpn_priority_four	
tpn_priority_five	
tpn_priority_six	
tpn_priority_seven	
tpn_priority_eight	
tpn_priority_nine	
tpn_priority_ten	

Task: Complete the **Capability Name** fields for **TPN Server Priority Levels**. Then, click **OK** or **APPLY** to save the information in the registry. **Note:** A blank **Capability Name** field indicates that everyone has access.

The TPN Server administrator must add TPN client user accounts with the correct permissions to the appropriate Experion and/or Windows user groups. Use Windows Explorer (or Set Security for the selected capability file during HCI Component Configuration) to give the new TPN client user accounts access to the capability files.

Detailed descriptions of the capability files are provided in the *Experion System Administration Guide.*

4.10 Creating a new capability file

Prerequisites

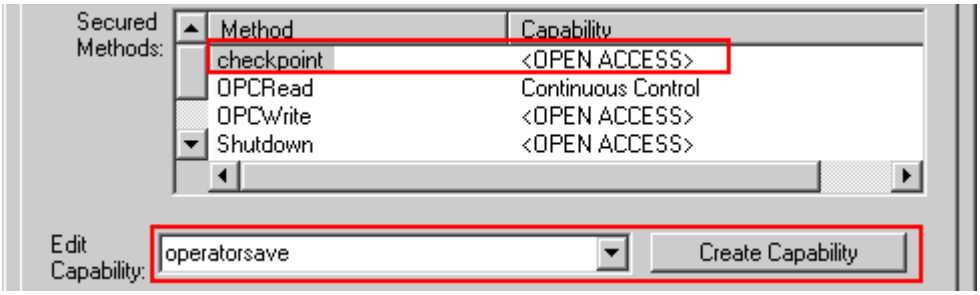
Attention

Honeywell provides standard capability files to meet most user needs. On occasion, you may want to create your own capability file, in which case, you must review the following information:

- 'Mandatory Capability File ACL Entries' section in the *Experion System Administration Guide.*
- 'HCI/OPC Server Capability Files' and 'HCI/OPC Server Capability File Customization' sections in the *Experion System Administration Guide*.

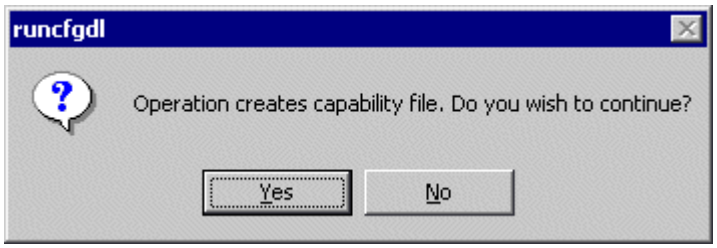
Procedure for creating a new capability file

1 From the **HCI Configure Component** page, click a **Method** to create a new **Capability** file.

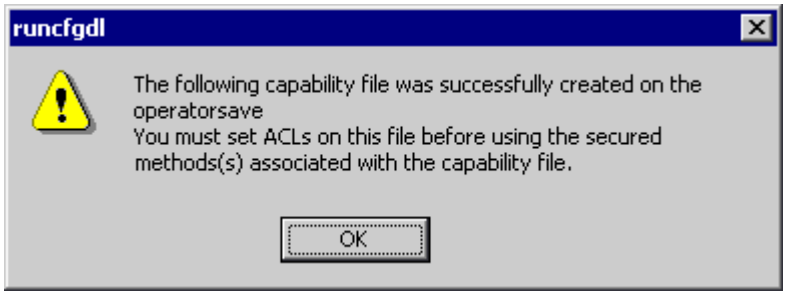


The **Create Capability** button is enabled.

2 Click **Create Capability**.
A dialog appears stating that a capability file will be created.

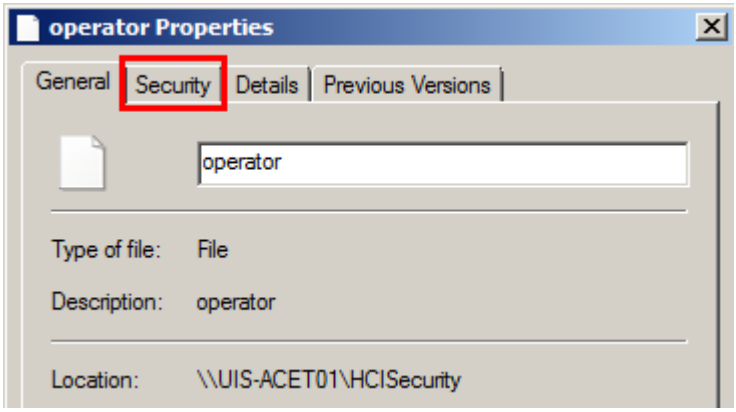


3 Click **Yes**.
A dialog appears stating that the ACLs must be set.

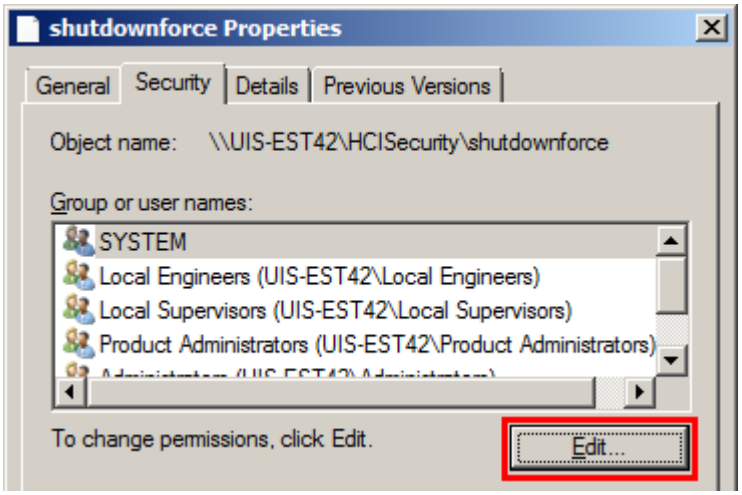


4 Click **OK**.
The **Set Security** button appears.

5 Click **Set Security**.
The **Properties** window for the capability file appears.



6 Click the **Security** tab.
The **Security** tab appears.



From this tab, click **Edit** to add/remove the appropriate users—use the guidelines provided in the *Experion System Administration Guide*.

Refer to the 'Mandatory Capability File ACL Entries' and 'HCI/OPC Server Capability File Customization' sections in the *Experion System Administration Guide*.

4.11 Establishing Auto Checkpoint

See the ' “Managing the TPN Server cache” on page 53' section to configure the TPN Server to automatically checkpoint the cache.

4.12 Testing the TPN Server AutoStart option

Procedure for verifying TPN Server AutoStart operation

1 From the **System Management Display**, select the TPN Server.

2 If the **AutoStart** option for the TPN Server was selected, then the **System Status Display** should show that the TPN Server has started.

If the **AutoStart** option was not selected, do the following steps:

- On the **System Status Display**, click the TPN Server name.
- Click **Start**.
- Click **Yes** to confirm the start.

3 The TPN Server configuration has now been completed. You can proceed with developing HCI Client applications or other forms of data access.

Attention

If the TPN Server fails to start after configuration, it usually means that the password for the DCOM and Windows accounts need to be synchronized. Refer to the ' “Synchronizing the DCOM and account passwords” on page 47' section.

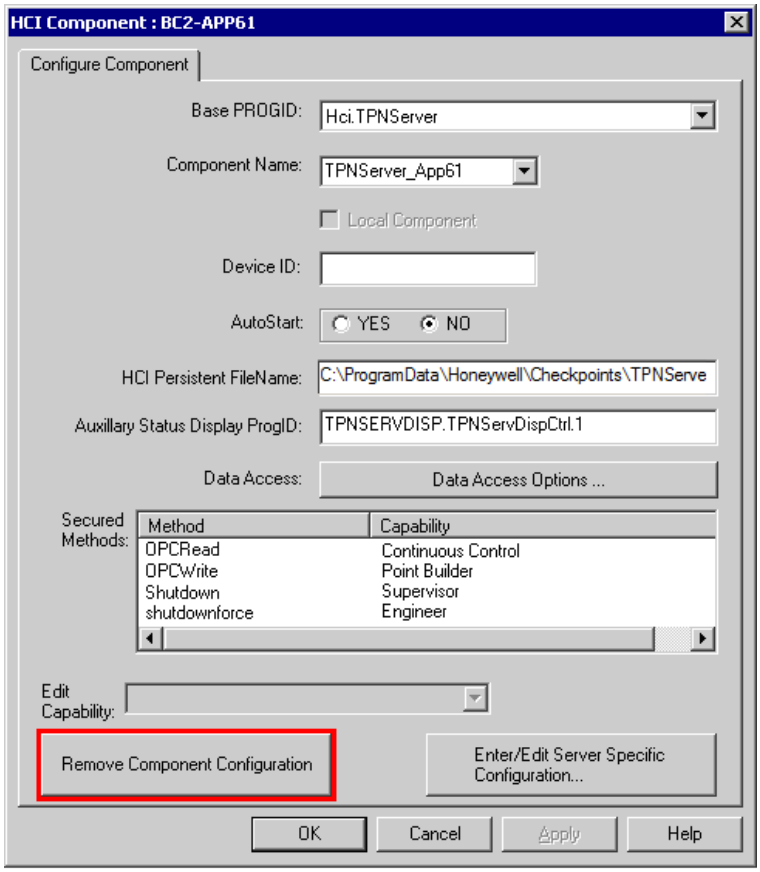
4.13 Removing a configured TPN Server

Procedure for removing a configured TPN Server

1 Stop the TPN Server (right-click the target TPN Server in the System Management Display, and then click **All Tasks** > **Stop**).

2 Choose **Start Menu** > **All Programs** > **Honeywell Experion PKS** > **System Management** > **Configuration Utility** > **Configure** > **HCI Component**.

Result: The **HCI Component** page appears.



3 Click **Remove Component Configuration**.

4 Verify that the TPN Server has been removed from the System Management display.

4.14 Modifying or removing a TPN Server

Modifying name returns a default configuration

Selecting a pre-configured TPN Server in the HCI Component Page and modifying its name removes the previous configuration and returns it to a default component configuration. You can then reconfigure the TPN Server when you select Hci.TPNServer from the HCI Component Page's Base ProgID pull down list.

Removing TPN Server component from a node

To completely remove a TPN Server component from a node, remove the TPN Server package using the Control Panel's Add/Remove Programs function.

4.15 Synchronizing the DCOM and account passwords

4.15.1 Background on TPN Server user account

The TPN Server requires a Windows account to launch the application. The local TPNServer 'runas' account is LocalCOMServer. The LocalCOMServer account is added as part of its installation with a password provided by the user during installation of the node.

Attention

In the Experion ESVT Cluster, it is critical that all nodes in the cluster have the same password assigned to the local LocalCOMServer account.

4.15.2 Password synchronization required

If the TPN Server fails a startup user identity and password check, it will not start because the password on the Windows account is not the same as the password for the TPNServer's DCOM settings. This failure is easily recovered from when you synchronize the TPNServer's DCOM password to its associated Windows account. The password utility, **Windows Services & DCOM Servers Log on tool,** is accessed from the System Management program group. The password utility itself does not change the password on the Windows account.

You can determine if the TPN Server failed to start because of password non-synchronization. To troubleshoot this problem, right-click the node of interest from the System Management display and then click **Display Events** from the context menu to invoke its Event Summary.

4.15.3 Password non-synchronization scenarios

The following scenarios illustrate how password non-synchronization could occur:

- A user may decide to change a password on a Windows account using an intervention such as Computer Management. Password non-synchronization will occur because the DCOM server for the TPN Server is still using the earlier password. After changing the Windows account password, the user also needs to enter and synchronize the new password with all DCOM servers on a node using the password utility.
- A plant is integrating an Experion system with their TPS system. The TPN Servers installed on the Experion nodes use a higher security password. The plant's current TPS system, however, already has Windows accounts for TPN Servers in place with passwords currently defined. Although the user runs a Workstation Security package, the security package will not overwrite currently used passwords on existing accounts. Password non-synchronization will occur because the TPN server on the Experion node is using the higher security password. The DCOM password in this case needs to be synchronized with the Windows account (LocalComServer) using the password utility.

  After identifying or changing the Windows account password, the account password also needs to be entered and synchronized with the DCOM servers using the password utility. If more than one node is involved, the password utility will need to be run on similarly non-synchronized nodes. If the TPNServer is configured to use the LocalComServer account is important that account password is the same on all nodes.

For procedures on how to use the password utility, refer to the 'Password Security and Synchronization' section in the *Integrated Experion-TPS User's Guide*.

4.16 Monitoring the TPN cable connection

Related topics

“How the TPN Server monitors connectivity” on page 48
“Default settings” on page 48
“Conditions for a transition to OK state” on page 48

4.16.1 How the TPN Server monitors connectivity

The Data Access side of the TPN Server continuously monitors the state of the TPN cable connection. The TPN Server sets itself to warning if the TPN side of the node reports that the node is 'off network', indicating that neither TPN cable is available. To prevent false warnings from intermittent problems, the off network condition must be detected for several consecutive scans. An event is logged in the PC's application log when the connection is considered severed. To prevent premature returns to the ok state, the return to normal TPN connection must be seen for several consecutive scans. Further, at least one other running TPN node outside the local node must be found in order to return to the ok state. By default, this check is done immediately on start up and then repeated every 5 seconds. By default, the scan must fail five consecutive scans to change to warning unless the node is initially off network at startup in which case the TPN Server will immediately start in the warning state. By default, the return to normal must be seen for 5 consecutive scans and at least one other running node must be found before the TPN Server transitions to the ok state.

4.16.2 Default settings

These defaults can be changed by settings values in the PC registry under the TPN Server's component name *in the HKEY_LOCAL_MACHINE\SOFTWARE \Honeywell\MyTpsDomain\HciComponents* folder. The 'TPNCableCheckCycle' registry value sets the time interval between scans for TPN cable faults in seconds (range 1 to 86400). The value 'TPNCableCheckRTNCycle' sets the time interval between scans for TPN cable return-to-normal in seconds (range 1 to 86400). The value 'TPNCableCheckRetries' sets how many consecutive scans are required to change the state (range 1 to 100). If the registry values do not exist or cannot be accessed, the default is used. Changing default settings to increased scan values could conceivably mask a cable fault; changing a setting should only be done if you are assured that faults are intermittent.

4.16.3 Conditions for a transition to OK state

The Alarm and Event side of the TPN Server must, at the very least, 'hear' an empty 'ping' packet or a real alarm through the alarm path to the TPN side every 4 seconds or the TPN Server will set itself to warning. This transition to warning also causes any existing alarm(s) to be set to bad:::commfail quality and connected clients are notified. This condition returns to normal as soon as at least one alarm or ping is heard again. A return to normal, in this case, causes the alarm data to be refreshed. Both the Data Access side and the Alarm and Event side must be in the ok state before the TPN Server, as a whole will transition to the ok state.

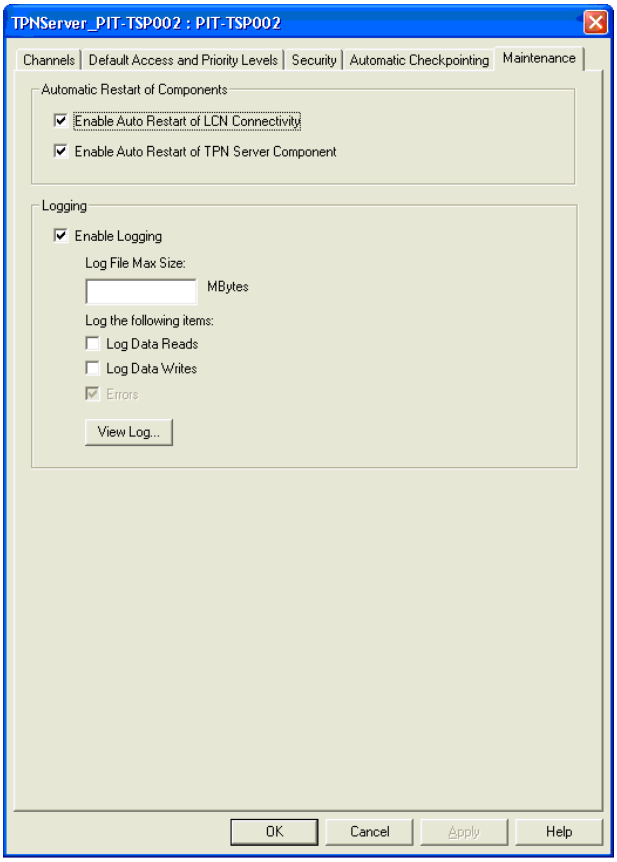
4.17 Restarting components automatically

Related topics

“Maintenance” on page 49
“Enable Auto Restart of LCN Connectivity” on page 50
“Enable Restart of TPN Server Component” on page 50
“Logging” on page 50

4.17.1 Maintenance

You can configure the TPN Server to maintain operations during critical failures such as termination of the TPN Server or communication failure with the LCN network. The options available in the **Maintenance** tab allow the TPN Server to restart the components automatically and maintain a healthy system.



The following table lists the default values for the options available in the **Maintenance** tab.

Property	Default value
Automatic Restart of Components	
Enable Auto Restart of LCN Connectivity	Disabled
Enable Auto Restart of TPN Server Component	Enabled
Logging	
Enable Logging	Enabled

Property	Default value
Log File Path	\\ProgramData\Honeywell\TPNServer
Log File Max Size	20 MB
Log Data Reads	Disabled
Log Data Writes	Disabled

4.17.2 Enable Auto Restart of LCN Connectivity

The **Enable Auto Recovery of LCN Connectivity** option allows you to restart the critical services required for the TPN Server. If this option is enabled and the connection to the LCN is lost, the TDC Emulator Service is restarted or in certain situations, the LCNP board is reset and restarted.

Attention

Do not set the **Enable Auto Recovery of LCN Connectivity** option for an APP/E-APP node as it resets the LCN. A reset could be disruptive to control loops, if LCN points are running in the AM functionality of an APP.

Attention

In the Enhanced TPS Node R431, the **Enable Auto Recovery of LCN Connectivity** option is not applicable, because the **LCNP Auto Restart** option has been disabled.

Do not set the **Enable Auto Recovery of LCN Connectivity** option for an ETN node as the LCN reset is not supported on an ETN node.

Alternatively, you can restart the emulator service. If this doesn't work, perform the following procedure:

1. Stop the TPN Server.
2. Shutdown and restart the LCN node.
3. Once the node is running, restart the TPN Server.

4.17.3 Enable Restart of TPN Server Component

The **Enable Restart of TPN Server Component** option allows for automatic shutdown and immediately restarts the TPN Server in one of the following situations.

- Non-recoverable failure in the TPN Server
- TPN Server crashed

If this option is enabled and there occurs a failure, the TPN Server is shutdown. Immediately, the TPN Server restarts and continues to collect data and events. As a result, the following event and alarm is raised.

- An event is logged in the Event Summary along with a reason for a restart, if the logging option is enabled.
- System Management Alarm is raised when the connectivity is lost, if network tree is loaded and the SES is configured.

4.17.4 Logging

The **Logging** option enables an engineer to track internal events occurred in the TPN Server, such as startup, connection events and other. If this option is enabled, additionally the following events are recorded in the event log.

To record event for…	Select the…
All errors	**Always enabled** option
Data reads	**Log Data Reads** option
Data writes	**Log Data Writes** option

The name of the log file is predefined. The path for saving the log file is provided in the **Log File Path** text box. This path is a shared folder and accessed through the network. The size of the log file cannot exceed the size mentioned in the **Log File Max Size** text box. When the log file reaches its maximum capacity, the TPNServer_log.txt.backup file is created.

In an ESVT node, the engineer can view the status of the autopriming and the event recovery in the log file.

Click the **View TPN Server Log** button to view the log file. The following depicts the format of the entries in the log file.

1/1/2010 4:45:03 PM	TPN Server Restarted Due to Lost Channels

5 Managing the TPN Server cache

Related topics

“Managing the TPN Server cache Overview” on page 54
“Configuring TPN Server automatic checkpointing” on page 55
“TPNCache Tool” on page 58

5.1 Managing the TPN Server cache Overview

Related topics

“OPC cache data versus device data” on page 54
“Caching of OPC item definitions” on page 54
“Checkpointing TPN internal IDs” on page 54

5.1.1 OPC cache data versus device data

Since the TPN Server is an OPC Data Access server supporting the reading of data from either device or cache, a client's request for a device read results in a TPN request for data from one or more TPN data owners (for example, NIM, HG, or AM). In addition to returning the result of the device read to the requesting client, the result is also stored in the TPN Server's in-memory cache. This cache contains the most current value, quality, and timestamp on an individual OPC item basis. Establishing the OPC item definition cache is completed via an automatic priming mechanism on TPN Server startup (see the 'Enabling automatic priming' section). There is a *single cache* shared between all clients of the TPN Server. Therefore, the value, time stamp, and status for an item are cached only once, even if that particular item has been added to multiple groups created by the same client or multiple clients.

5.1.2 Caching of OPC item definitions

Along with the most current value, quality, and timestamp of a particular OPC item, the TPN Server caches the item definition information. This includes the canonical data type, the internal format of the native TPN System entity and parameter ID, access rights, and so forth. The TPN Server establishes the definition information as a result of the first IOPCItemMgt::AddItems for a given item by any client. If the OPC item has not been previously cached, then the TPN Server converts the OPC item ID to an internal TPN ID. This external to internal conversion can take approximately .5 seconds for each OPC item. Subsequent client requests for the same item do not include the 0.5 seconds to convert the OPC Item ID to the TPN internal ID.

5.1.3 Checkpointing TPN internal IDs

In addition to being cached, the TPN internal IDs along with other OPC item definition information are saved when the TPN Server shuts down. This information can also be saved to a checkpoint file through both on-demand and automatic checkpoint requests. The TPN internal IDs with corresponding definition information are restored when the TPN Server starts up.

For more information about autopriming, refer to the *Integrated Experion-TPS User's Guide*

5.2 Configuring TPN Server automatic checkpointing

Related topics

“Configuring TPN Server automatic checkpointing Overview” on page 55
“Checkpoint / Prime page” on page 55
“Establishing time periods to checkpoint” on page 56
“Establishing pre-defined times to checkpoint” on page 56
“Enabling automatic priming” on page 56

5.2.1 Configuring TPN Server automatic checkpointing Overview

Automatic checkpointing allows the cache to be saved by time interval or at pre-defined times during the day. The TPN Server must be running in order to successfully checkpoint (this applies to automatic or on-demand checkpoint). The TPN Server does not have to be running in order to configure automatic checkpointing. However, if the TPN Server is running during the configuration operation, it must be restarted in order for the configuration changes to take effect.

The automatic checkpoint dialog is invoked from the TPN Server configuration page (see the ' “Configuring TPN Server automatic checkpointing” on page 55' section). The TPN Server Auto Checkpoint functionality is initially defaults to Disabled.

5.2.2 Checkpoint / Prime page

From the server-specific configuration page, click the Checkpoint / Prime tab to open the view shown in “Figure 7: Checkpoint / Prime Page”.

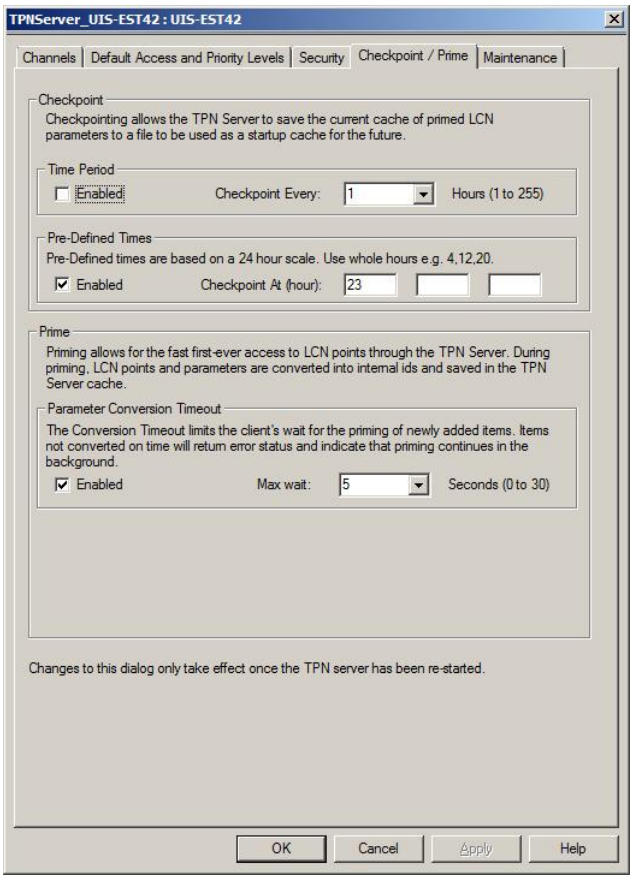


Figure 7: Checkpoint / Prime Page

5.2.3 Establishing time periods to checkpoint

Select the **Enabled** check box in the **Time Period** group to establish periodic backups of the TPN Server's cache. An integer value between 1 and 255 is a valid entry in the **Checkpoint Every** box. The time period is relative to the start-up time of the TPN Server. For example, if the TPN Server started at 10:30 A.M., it automatically checkpoints at 11:30 A.M., 12:30 P.M., 1:30 P.M., and so forth.

5.2.4 Establishing pre-defined times to checkpoint

Select the **Enabled** checkbox in the **Pre-Defined Times** group to set up pre-defined times for saving the cache to the checkpoint file. When enabled, the TPN Server copies the cache to the checkpoint file at the specified times. A 24-hour clock is used to specify the maximum of six pre-defined times, which are given to the nearest hour. A blank time box has no effect on automatic checkpointing. Duplicate values are allowed, but are ignored by the TPN Server.

Attention

The check box in the **Pre-Defined Times** group can be enabled if the check box in the **Time Period** group is also enabled. The auto checkpoint options are not mutually exclusive.

5.2.5 Enabling automatic priming

The autopriming function prepares the TPN Server cache for a fast data access to the TPN. Beginning with the Experion R400 the autopriming option is available on EST, ESVT and ACE-T nodes. Autoprime removes the

need to maintain the TPN Server cache and Experion point database using the off-line priming utilities. Experion server database is populated when the point parameters are accessed for the first time.

Experion R400 TPN Server introduces a new format of its cache, resulting in a very compact internal representation of primed items. Thereby, all points on the local LCN and all configured parameters on these points are included in the scope of autopriming.

The 'Autoprime' option is available on the ESVT nodes. The ESVT TPN Server browses and primes all points and their parameters. Other nodes in the ESVT cluster search for, and import results of priming conversions from the ESVT TPN Server when needed, over the network.

The Autopriming option is not available on the Experion APP nodes. The new compact format of the cache is supported on all Experion LCN connected nodes.

For more information, refer to the *Integrated Experion-TPS User's Guide*

5.3 TPNCache Tool

Related topics

“Installing and using the TPNCache Tool” on page 58

5.3.1 Installing and using the TPNCache Tool

The TPNCache tool is installed on all LCN connected Exeprion R400 nodes. The TPNCACHE.EXE is a command line utility, located under C:\Program Files\Honeywell\TPS\TPNServer folder. It provides functions to list and manipulate contents of the TPNServer cache.

To see command-line help, invoke: 'tpncache -?'

Attention

With Experion R410, 64-bit OS is supported. All 32-bit components are installed in 'C:\Program Files (x86)'. The TPNCACHE.EXE command line utility is located under 'C:\Program Files (x86)\Honeywell\TPS\TPNServer' folder.

6 Monitoring TPN Server status

Because the TPN Server is a managed component in the Experion system, it appears in the System Management display. The TPN Server provides status information to both the scope pane and the results pane of the System Management display.

The managed TPN Server can be configured to be auto-started upon system power-up by the System Management runtime. Alternatively, a managed component may be manually started from the System Management display or by a client with appropriate access levels.

This section describes how you manually start up and shut down a TPN Server. Additionally, node status indications are described so that you can interpret the operations status of your TPN Server.

6.1 Typical user operations

From the System Management display, you can do the following:

- Startup the TPN Server
- View the status of HCI components such as the TPN Server
- Start the TPN Server Status display
- Perform a checkpoint of the TPN Server
- Shutdown the TPN Server

6.2 Starting the System Management display

Procedure for starting the System Management display

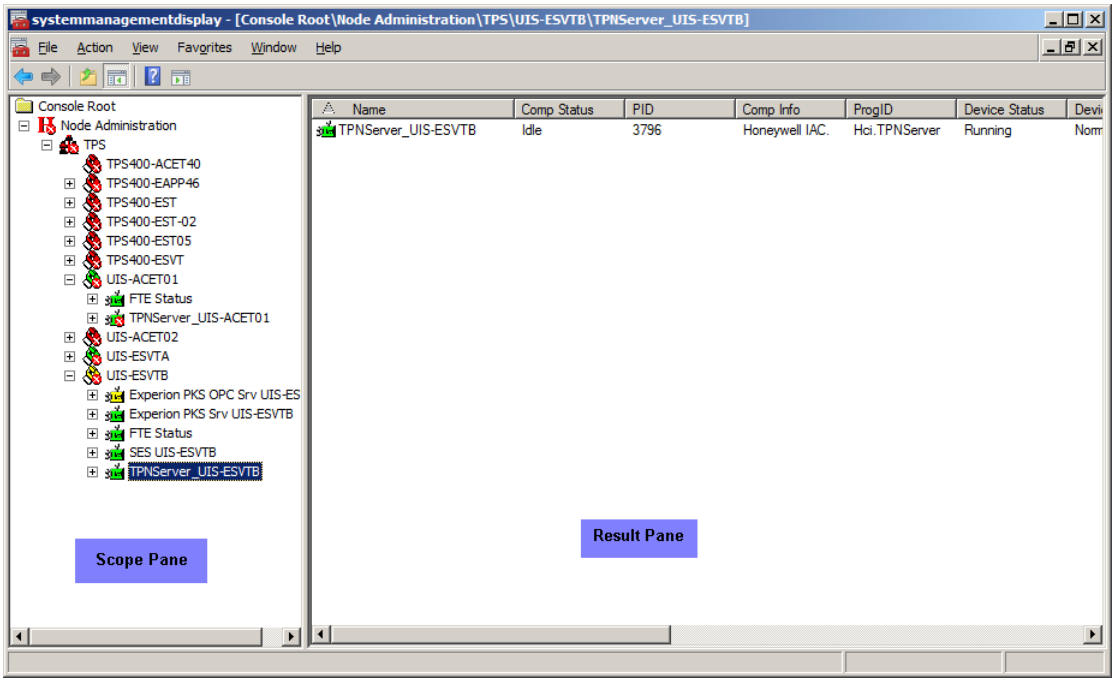
1. Perform the following procedure to start the System Management display from the Run command.Click **Start Menu** > **All Programs** > **Honeywell Experion PKS** > **System Management** > **System Management Display** to open the System Management display.
2. When the MMC appears, select a previously saved console view that represents your plant's System Management display.

6.3 Starting the TPN Server

Procedure for starting the TPN Server

1 Use one of the methods in the ' “Starting the System Management display” on page 61' section to start the **System Management** display.

2 The **System Management** display appears showing icons in the scope pane (left side) and HCI component information in the results pane (right side).



3 In the scope pane, click the node icon on which the TPN Server is configured in the TPS domain.
Scope pane—visual state of component (icon color):

Green—running or Idle state

Yellow—warning state, waiting for TPS Node Personality to load Lost connection to TPS Node Personality

Red—not in Running or Warning state

Results pane—indication in Comp Status column:

Idle—server is operational, no clients are connected

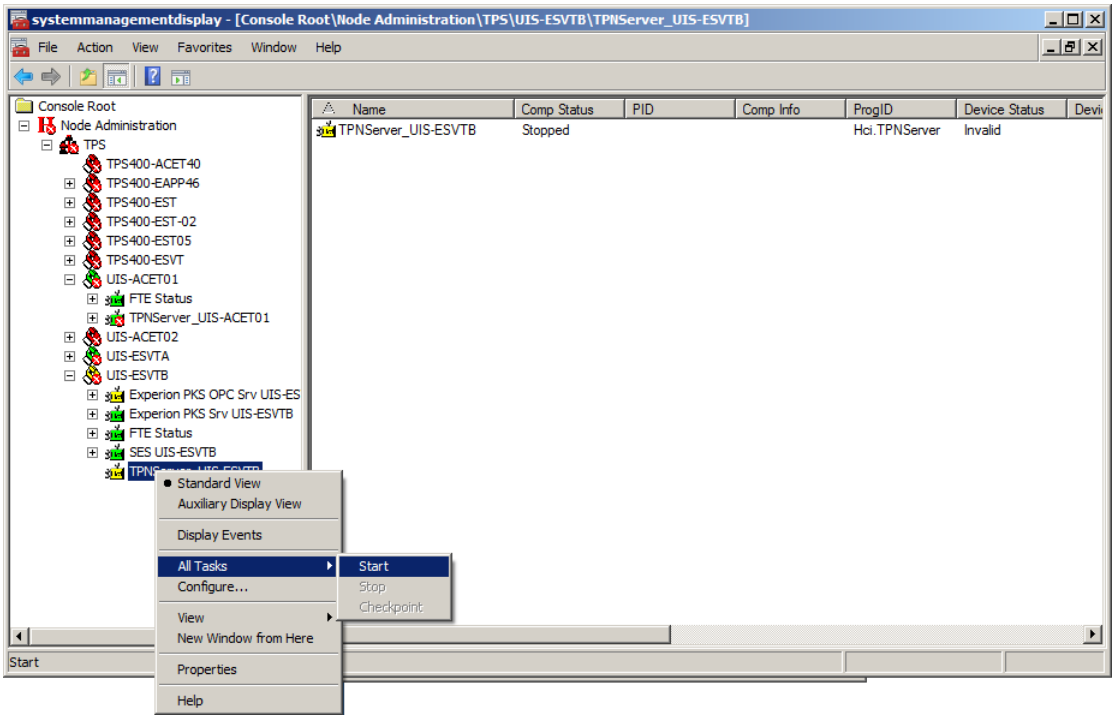
Running—server is operational. Clients are connected

Stopped—server is configured, but not running

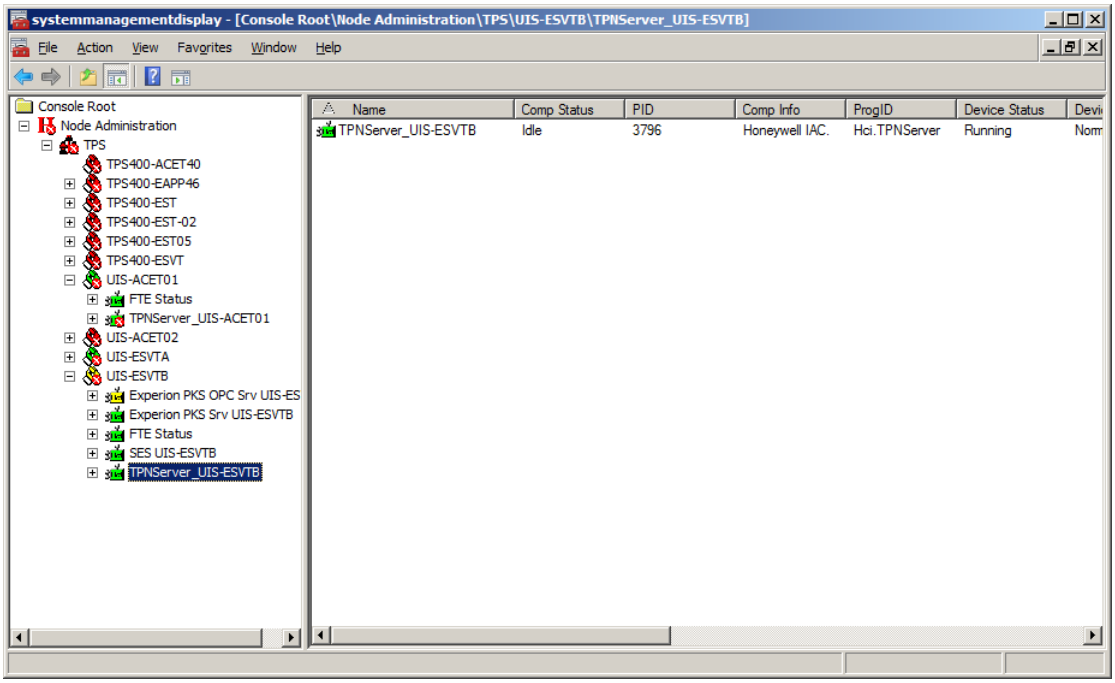
Warning— TPS Node personality is not loaded in LCNP or connection to TPN has been lost

4 If your **TPN Server** is **Stopped**, it can be started using the following steps:

- From the results pane, right-click the HCI component.
- From the shortcut menu, click **All Tasks** > **Start**.



5 In **Comp Status** column of the results pane, note that the status changed from **Stopped** to **Idle**. See step 4 for component status descriptions.



6.4 Starting the TPN Server Auxiliary Display

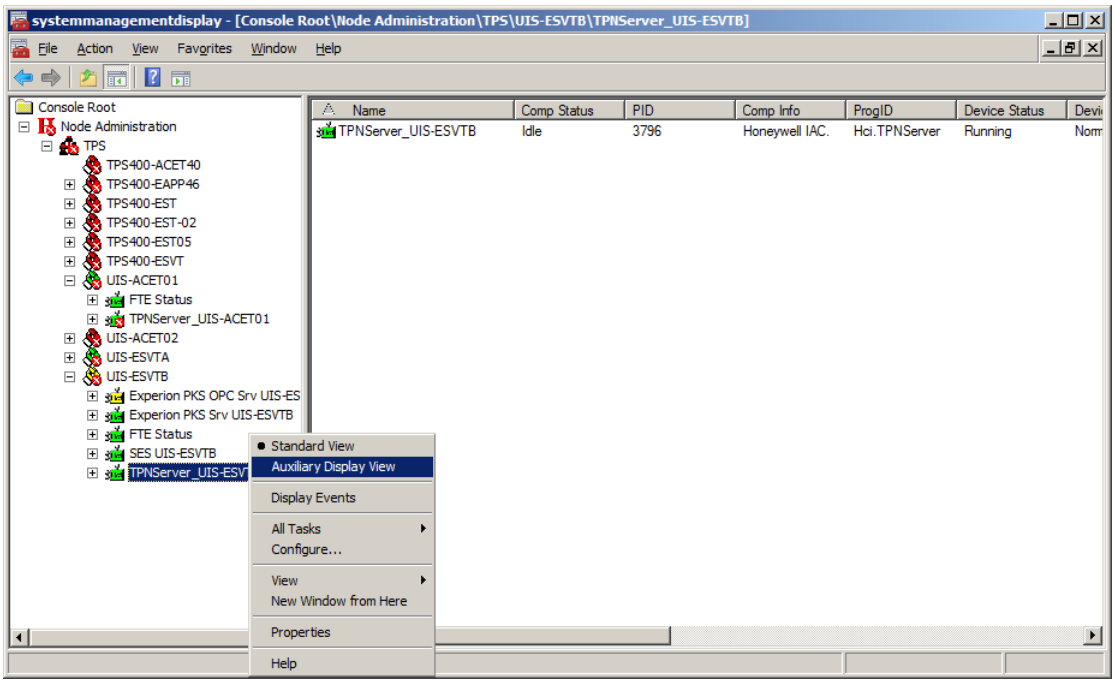
Procedure for starting the TPN Server Auxiliary Display

1 Use one of the methods in the ' “Starting the System Management display” on page 61' section to start the **System Management** display.

2 Select and then right-click the entry in the **Name** column.

Result: A shortcut menu appears.

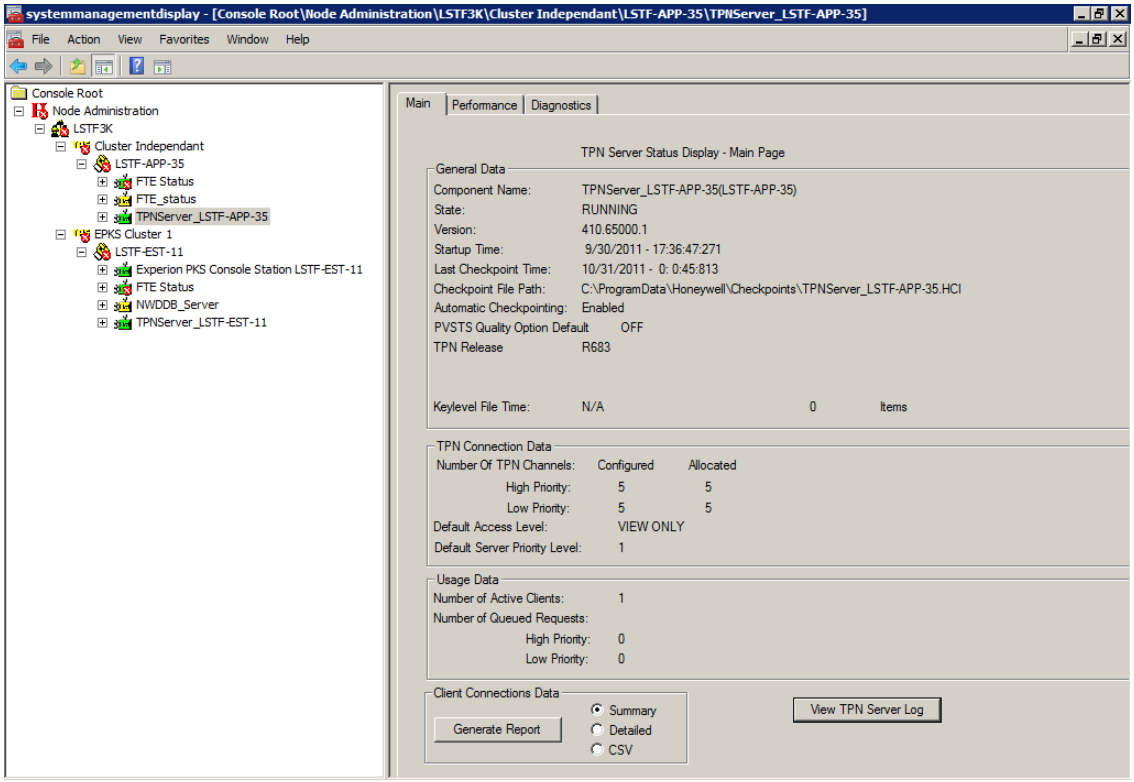
Note: The **Auxiliary Display** *cannot* be started in the **Stopped** or **Fail** states.



3 Click **Auxiliary Display View**.

4 See the ' “TPN Server Auxiliary Display” on page 73' section for details about the **Main**, **Performance**, and **Diagnostics** pages.

For LCN-connected Experion nodes, an **Events** page is also available. See the ' “Events Page” on page 84' section for details.


systemmanagementdisplay - [Console Root\Node Administration\LSTF3K\Cluster Independant\LSTF-APP-35\TPNServer_LSTF-APP-35]
File Action View Favorites Window Help
Console Root
Node Administration
LSTF3K
Cluster Independant
LSTF-APP-35
FTE Status
FTE_status
TPNServer_LSTF-APP-35
EPKS Cluster 1
LSTF-EST-11
Experion PKS Console Station LSTF-EST-11
FTE Status
NWDDB_Server
TPNServer_LSTF-EST-11
Main
Performance
Diagnostics
TPN Server Status Display - Main Page
General Data
Component Name: TPNServer_LSTF-APP-35(LSTF-APP-35)
State: RUNNING
Version: 410.65000.1
Startup Time: 9/30/2011 - 17:36:47:271
Last Checkpoint Time: 10/31/2011 - 0: 0:45:813
Checkpoint File Path: C:\ProgramData\Honeywell\Checkpoints\TPNServer_LSTF-APP-35.HCI
Automatic Checkpointing: Enabled
PVSTS Quality Option Default OFF
TPN Release R683
Keylevel File Time: N/A 0 Items
TPN Connection Data
Number Of TPN Channels: Configured Allocated
High Priority: 5 5
Low Priority: 5 5
Default Access Level: VIEW ONLY
Default Server Priority Level: 1
Usage Data
Number of Active Clients: 1
Number of Queued Requests:
High Priority: 0
Low Priority: 0
Client Connections Data
Generate Report
Summary
Detailed
CSV
View TPN Server Log

6.5 Checkpointing the TPN Server

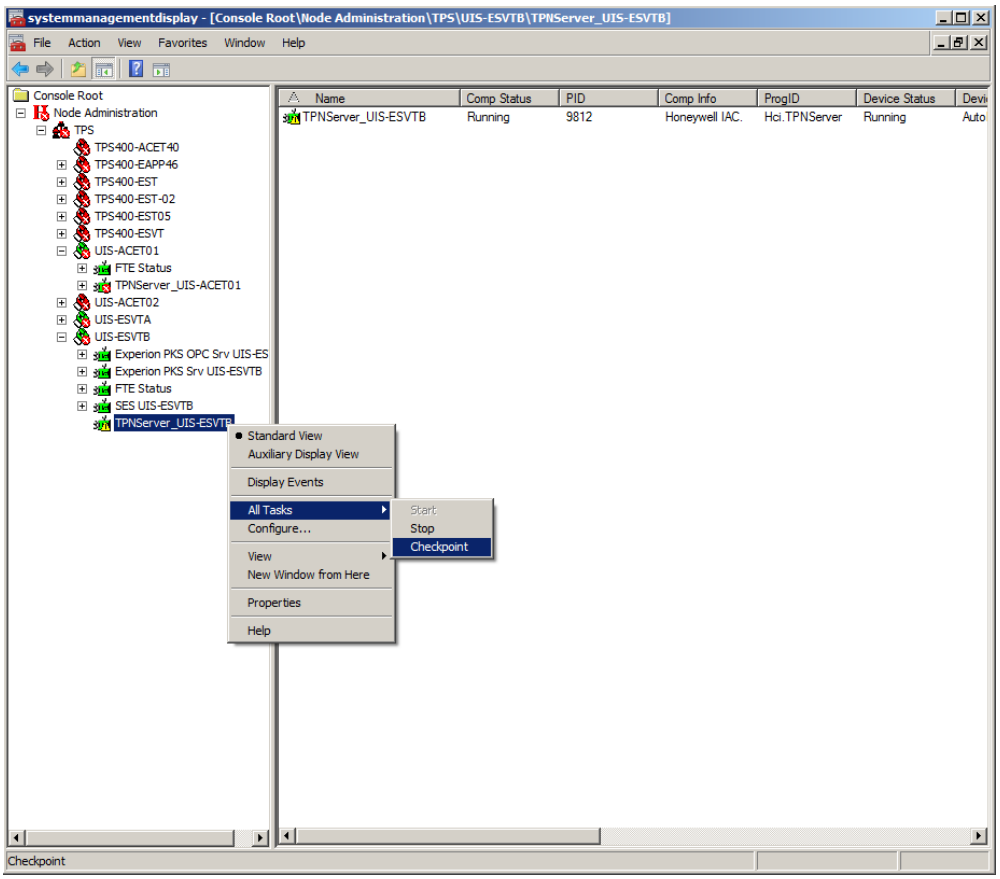
Procedure for checkpointing the TPN Server

1 Use one of the methods in the ' “Starting the System Management display” on page 61' section to start the **System Management** display.

2 Select and then right-click the **TPNServer**.

Result: A shortcut menu appears.

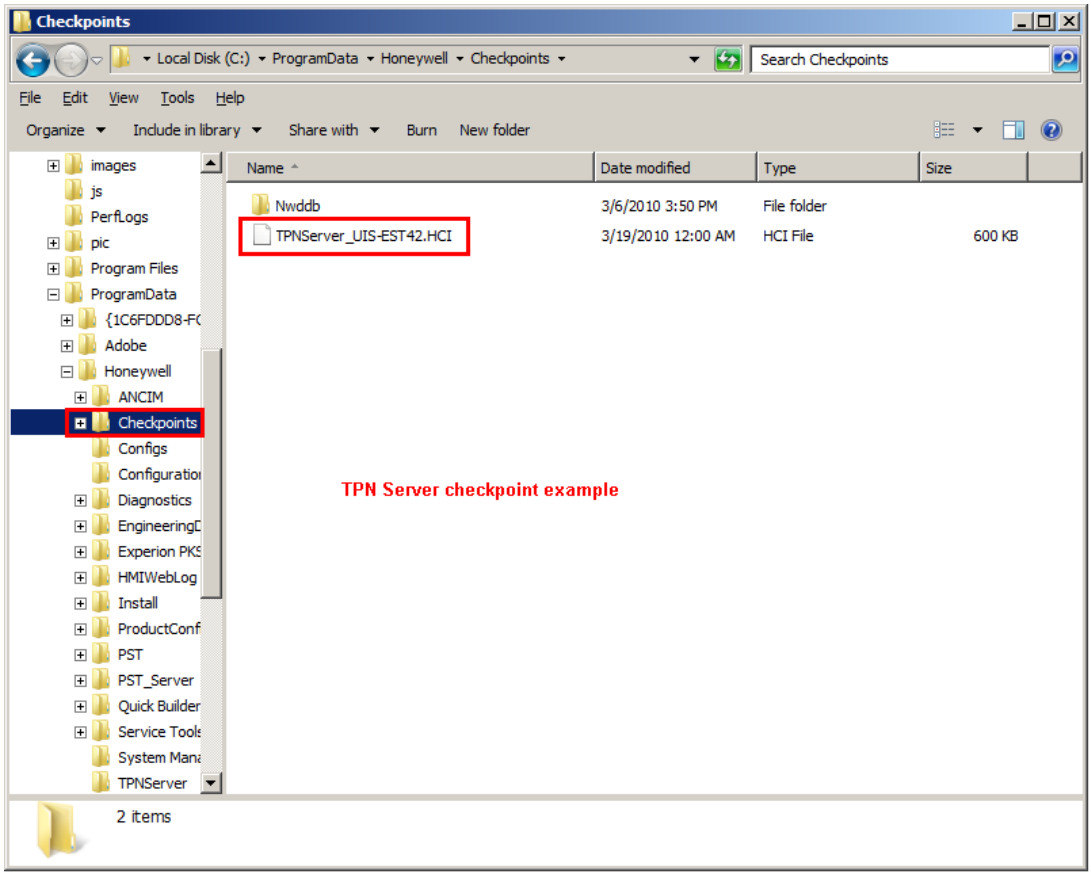
Note: The **TPN Server** should be in the **Idle** or **Running** state to checkpoint.



3 Click **All Tasks** > **Checkpoint**.

4 Performing a checkpoint saves the TPN cache and updates the timestamps into the **Checkpoints** folder under the C:\ProgramData\Honeywell\Checkpoints directory as shown.

Note: A request to checkpoint the TPN Server refreshes the KEYLEVEL.KL file from the History Module.


Checkpoints
Local Disk (C:) ▸ ProgramData ▸ Honeywell ▸ Checkpoints
Search Checkpoints
File Edit View Tools Help
Organize Include in library Share with Burn New folder
images
js
PerfLogs
pic
Program Files
ProgramData
{1C6FDDD8-FC
Adobe
Honeywell
ANCIM
Checkpoints
Configs
Configuratio
Diagnostics
EngineeringD
Experion PKS
HMIWebLog
Install
ProductConf
PST
PST_Server
Quick Builder
Service Tools
System Mana
TPNServer
Name
Date modified
Type
Size
Nwddb
3/6/2010 3:50 PM
File folder
TPNServer_UIS-EST42.HCI
3/19/2010 12:00 AM
HCI File
600 KB
TPN Server checkpoint example
2 items

6.6 Shutting down the TPN Server

Prerequisites

Attention

You must be logged on as a user that is a member of the Product Administrators group or as a user that has permission (shutdown) to do the following steps.

Procedure for shutting down the TPN Server

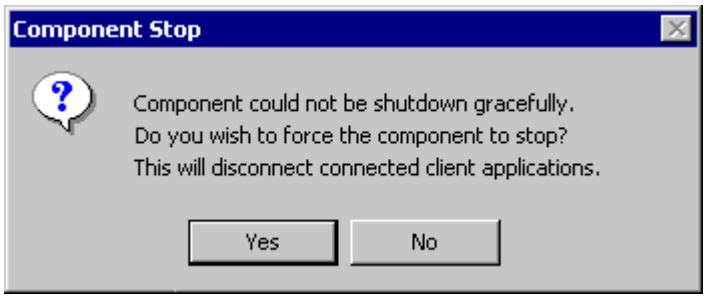
1 Use one of the methods in the ' “Starting the System Management display” on page 61' section to start the **System Management** display.

Note: If the TPN Server is in the **Running** or **Idle** state, it can be shut down from the **System Management** display.

Attention

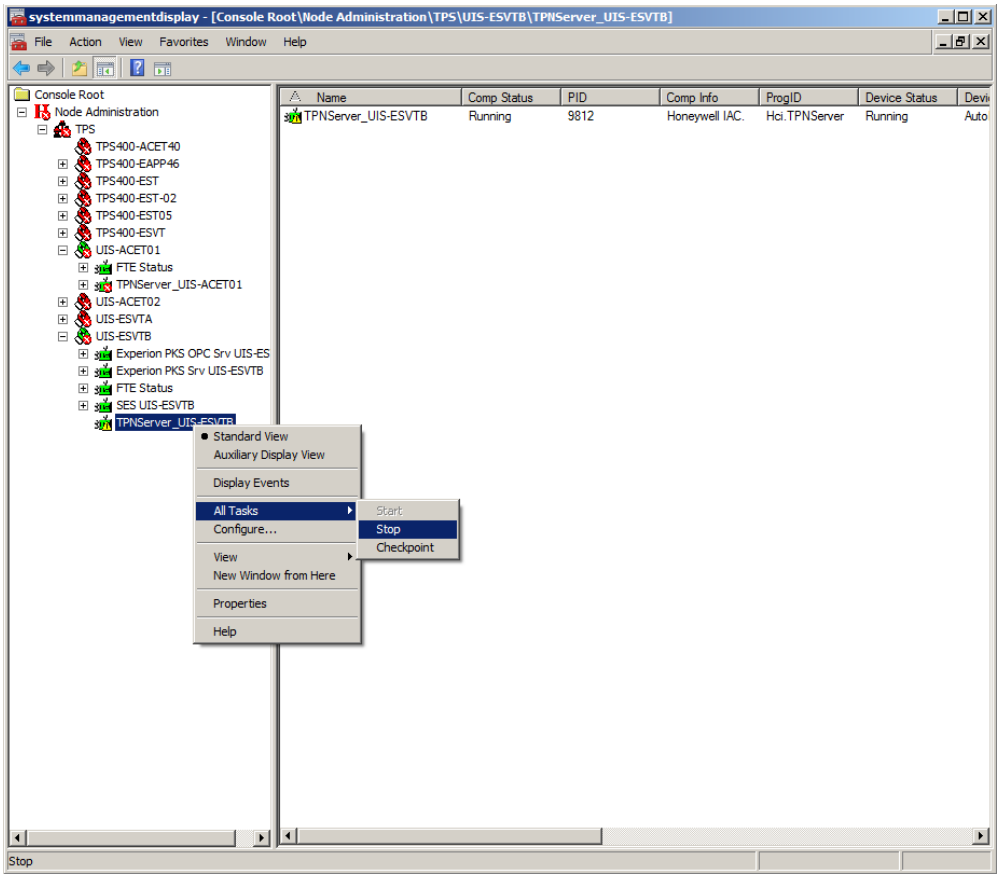
When attempting to manually shut down a component that has connected client applications, the following dialog appears. Confirming the dialog will cause the client application to be disconnected and subsequent forced shutdown.

System administrators can assign the users who are allowed to perform a forced shutdown when they define the security setting for the shutdown force capability file. To configure the security settings, see the ' “Determining TPN Server security” on page 39' section.



2 Right-click the HCI component that represents the TPN Server that you want to shut down.

Result: A shortcut menu appears**.**



3 Click **All Tasks** > **Stop**.

The **State** of the HCI component **TPNServer_APP61** transitions to **Shutting Down** and returns to **Stopped**.

The TPN Server shutdown is complete.

6.7 Interpreting status indications

Related topics

“Status in the scope and results panes” on page 70
“Description of scope and results panes” on page 70
“Scope pane” on page 70
“Results pane” on page 72
“Synchronization between nodes” on page 72

6.7.1 Status in the scope and results panes

The TPN Server is a Managed Component in the TPN System. All Managed Components will appear in the System Management display. The TPN Server is responsible for providing status information to both the scope pane and the results pane of the System Management display.

6.7.2 Description of scope and results panes

“Figure 8: Scope and Results Panes” shows the System Management display with its scope pane (left side) and results pane (right side). The scope pane shows Node Administration as the root of the tree and each of the static computers, domains, or TPS domains selected for monitoring. The results pane shows the status of each HCI managed component.

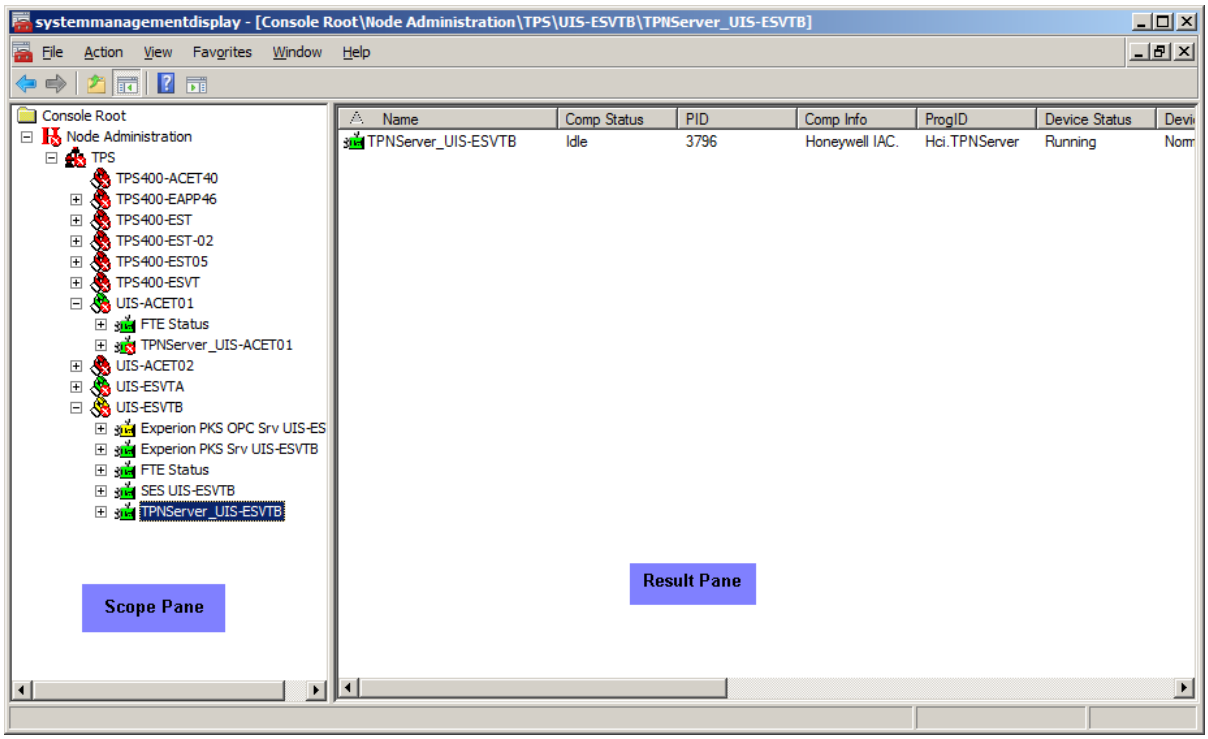


Figure 8: Scope and Results Panes

6.7.3 Scope pane

“Figure 9: Detailed Scope Pane” shows the scope pane with a tree display of several levels. Double-click a level to expand the tree display until you see the HCI managed component of interest.

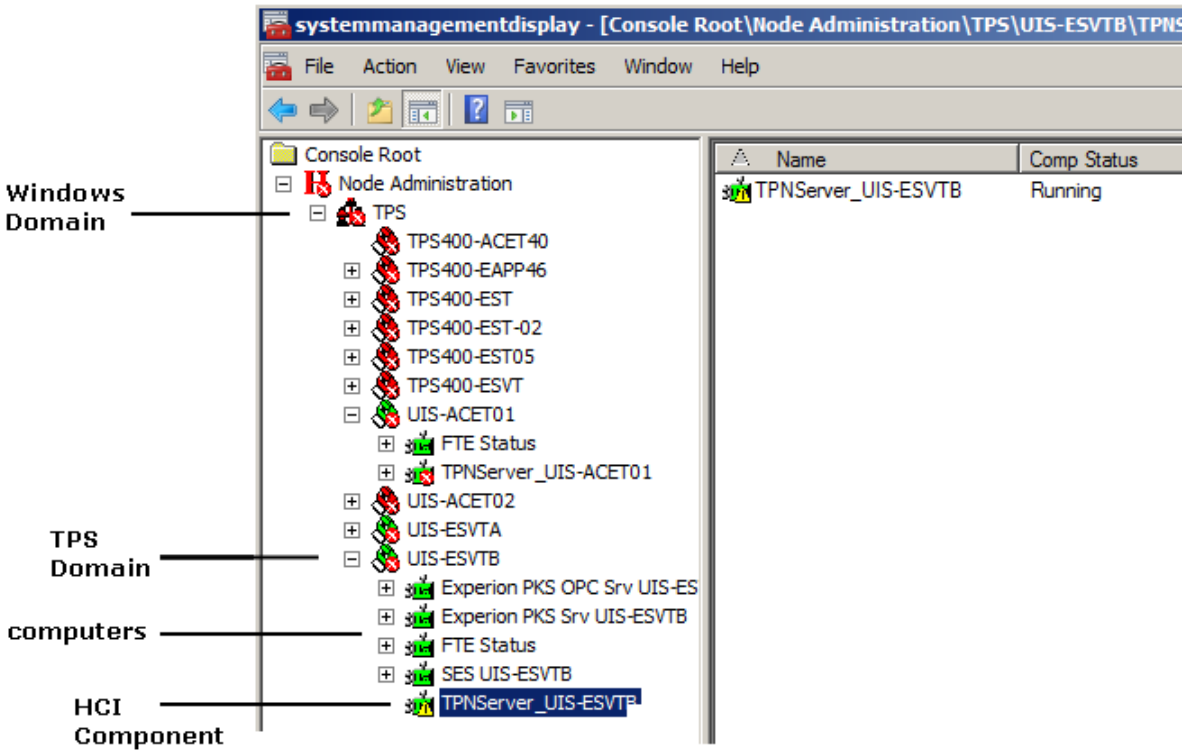


Figure 9: Detailed Scope Pane

Determining the state of a component

The scope pane provides an icon that gives a quick visual indication of the state of the component. The icon can be one of three colors as follows:

- **Green**—component is in Running or Idle state. The TPS Network Personality is running and the server is fully functional.
- **Yellow**—component is in one of the following Warning State conditions:
  - It is waiting for TPS Network Personality to be loaded and transition to the running state on the LCNP in the local node.
  - It has lost its connection to the TPS Network Personality.
- **Red**—component is not in Running or Warning State. The System Status Display will default to the Stopped state if the TPN Server is not running.

Reasons for server warning state

The server will be in the Warning state for one of the following reasons:

- The server is waiting for the TPS Network Personality to be loaded and transition to the running state on the LCNP in the local node, or
- The server has lost its connection to the TPS Network Personality.

Possible causes for losing the connection to the TPN Network Personality could be that the Personality has failed, or that the LCNP was reset.

The Server will be in the Idle or Running State when the personality is running and the server is fully functioning. The System Management display will default to the configured state if the TPN Server is not running.

6.7.4 Results pane

The results pane presents the status for the selected item from the scope pane**.** The results pane presents information in configurable columns.

- Name column contains the name of the item.
- Information column contains supplementary information about the names items.
- State column contains status information about the named component.
- Device Status column contains supplementary information about warnings.

To change column width for any of these columns, drag the column separator to the left or right. To sort a column, click on the column header.

The results pane has a field to display information about the status of the component. This field is a string of 40 characters or less. The TPN Server will only provide useful (non-NULL string) information for this field if it is in the Warning state.

Refer to the *HCI Client Developer's Guide* for a complete description of the HCI component statuses.

“Table 6: Possible Status Values in the TPN Status Display” lists the possible status values that may appear in the TPN Status Display:

Table 6: Possible Status Values in the TPN Status Display

Component status	Icon color	Server status values
Configured	Yellow	Component is not currently operational, but has been configured on the node.
Failed	Red	Component terminated abnormally.
Idle	Green	Component is operational, but no clients are connected.
Initializing	Yellow	Component is initializing.
Running	Green	Component is running.
Shutting down	Yellow	Component is shutting down.
Suspended	Yellow	Component is in a suspended state (not used).
Test	Yellow	Component is in a testing state.
Warning	Yellow	Component is running, but it has a problem that is not disabling the component.

6.7.5 Synchronization between nodes

The Experion System Status Displays on the TPS Network operate independently of each other. This means that displays on different nodes may show a different status between configurable update cycles (typically every 15 seconds).

6.8 TPN Server Auxiliary Display

Related topics

6.8.1 TPN server Auxiliary Display Overview

The TPN Server Auxiliary Display is an Active X Control that can be embedded in any Active X Container. It can also be viewed from the System Management Display as described in the ' "Starting the TPN Server Auxiliary Display" on page 64' section.

The **Auxiliary Display** is divided into three or four pages—*Main*, *Performance*, *Diagnostics and Events*. Each page is represented by a corresponding tab on the Active X Control. The Events page is not shown on the Experion APP nodes.

6.8.2 Main page

On the Auxiliary Display, click the **Main** tab to open the page shown in "Figure 10: TPN Server Status Display —Main Page". This page is revised for the Experion R410 release.

Figure 10: TPN Server Status Display—Main Page

6.8.3 General Data section-Main page

The **General Data** section provides the following information:

- **Component Name**—component name of the TPN Server and host machine name.
- **State**—state of the currently active TPN Server. The Auxiliary Status display is activated only after the component starts, and therefore only a subset of the possible HCI states is available for viewing. Due to their transitory nature, some of these states will never be shown. The State will be one of the following values:
  - **FAILED**—server has reached an unrecoverable error and has failed.
  - **IDLE**—server is running, but no clients are connected.
  - **INITIALIZING**—server is initializing.
  - **RUNNING**—server is running and clients are currently connected.
  - **SHUTDOWNCOMPLETE**—server shutdown has completed.
  - **SHUTTINGDOWN**—server is shutting down.
  - **TEST**—component is performing a self-test.
  - **WARNING**—server is running but not fully functional. New client connections are rejected.
- **Version**—version number of the currently active TPN Server.
- **Startup Time**—time and date that the TPN Server was started.
- **Last Checkpoint Time**—time and date of the Last Checkpoint.
- **Checkpoint File Path**—path of the Checkpoint file. Note that the checkpoint file contains item definitions that are valid only for the TPN system that the TPN Server is connected to.

- **Automatic Checkpointing**—associated with the **Time Period** option as configured on “Figure 7: Checkpoint / Prime Page”.
- **PVSTS Quality Option Default**—associated with the **Default Behavior for Uncertain Quality** option as configured on “Default Access and Priority Levels page” on page 34. The **PVSTS Quality Option Default** is activated with TPN software version R680 and later.
- **TPN Release**—current release number of the LCN/TPN interface.
- **Automatic Priming**—indicates if automatic priming is enabled for all the points and parameters on the LCN.
- **Keylevel File Time**—reports date/time when the TPN Server most recently went to the TPN, copied and evaluated the keylevel.kl file. Changes to this file must be made on the TPN. Any changes made on the TPN after this time are not yet being used by the TPN Server.
- **Items**—reports the number of parameters with custom keylevels. Note that this can differ from the number of non-comment lines in the file due to duplicate parameters. Only the most restrictive entry is entered into the map for duplicated parameters.

Keylevels functionality

The TPN Server reads the TPN keylevel.kl file during startup and during each checkpoint. The data in that file is assumed to be edited and error checked by the TPN using the usual TPN tools and methods. The file is used to create a special map of parameters versus custom access levels like the keylevel provides on the TPN. Whenever a single parameter store is received from Experion PKS, and the current access level is one of the 'human' access levels, the map is consulted to see if a more or less restrictive access level should be set for this write. If so, that custom access level is set for the store. This functionality is designed to mimic the way the keylevel function works on the TPN. The time that the file was last read from the TPN is reported on the TPN Server Auxiliary Display along with the number of items currently in the map.

6.8.4 TPN Connection Data section

The **TPN Connection Data** section provides the following information about the TPN Channel Connections made by the TPN Server.

- **Number of TPN Channels**—There are two columns of data provided as follows:
  - The **Configured** column contains the number of both High and Low Priority TPN channels that were configured in the TPN Server Channel Configuration page.
  - The **Allocated** column contains the number of both High and Low Priority TPN channels that the TPN Server was able to allocate for use at runtime. The number of channels configured must be available for the TPN Server to start successfully.
- **Default Access Level**—This is the default access level that was configured in the TPN Server Access and Priority Level Configuration page. If the client does not explicitly set the access level for a store request, then the TPN Server will use this default access level.
- **Default Server Priority Level**—This is the default server priority level that was configured in the TPN Server Access and Priority Level Configuration page. If the client does not explicitly set the server priority level for accessing the LCN data, then the TPN Server will use this default server priority level.

6.8.5 Usage Data section

The **Usage Data** section provides the following information about runtime usage data on the TPN Server.

- **Number of Active Clients**—number of client connections to the TPN Server.
- **Number of Queued Requests**—The TPN Server queues both high and low priority requests while all allocated channels are busy. The number of high priority and low priority requests that are waiting to be processed by an available channel is displayed. These values are updated on one-second intervals. Typically, the number of queued requests is low. A value of zero means that all requests are immediately processed. Values greater than zero mean that requests are being queued until a channel becomes available. A large number of requests being queued will signal a heavy load on the TPN Server with the potential for

performance problems. In this event, you should limit the number of client connections or increase the number of configured channels.

When the total number of queued requests exceeds 80% of the number of outstanding requests configured for the given TPN Server, the TPN Server Auxiliary Display will flash 'OVERLOADED'. This indicates the need for corrective action to reduce the load on the TPN Server. When the total number of queued requests exceeds the number of outstanding requests, the TPN Server will throttle requests until the total number of queued requests no longer exceeds the number of outstanding requests. The implemented throttling mechanism is to immediately return E_FAIL on new requests without processing them until the server recovers from the overloaded condition. The number of outstanding requests is configured on the Channels configuration page for the TPN Server.

Attention

If a value is displayed as '**?**,' then the data are either inaccessible or the TPN Server has failed. If the TPN Server has failed, all values will display '**?**' with the exception of the component name. If the TPN Server is in the WARNING state, the values for **Number of TPN Channels** and **Number of Queued Requests** will display '**?**.'

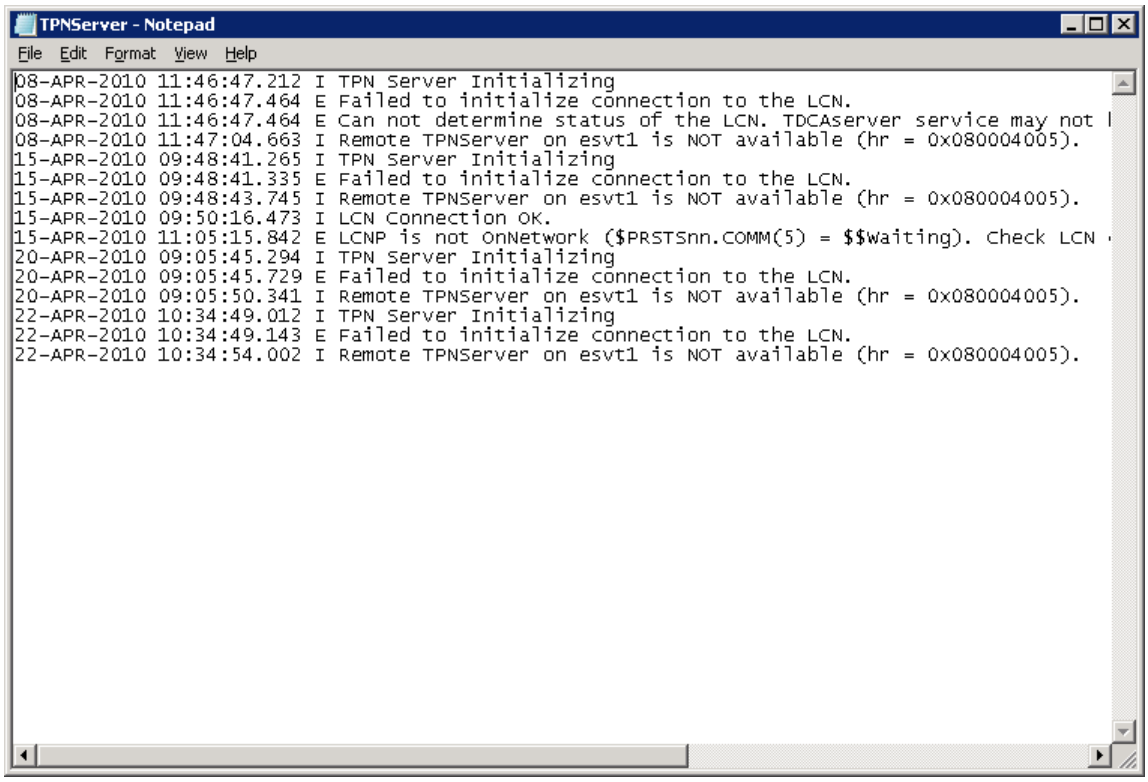
WARNING

The TPN Server has an HCI Persistent file that stores the cached TPN System entity and parameter IDs when the checkpoint command in the Experion System Status display is initiated, and when the TPN Server is shut down. If a point.parameter that has already been accessed by the TPN Server must be rebuilt due to a change in its definition on the TPN, TPN Server will update its internal definition next time it is requested.

- For advanced users, programmatically, a client can update a point definition in the cache if the definition has changed. When a client performs an OPC Read operation and receives a ppError = OPC_E_UNKNOWNITEMID, or OPC_E_INVALIDHANDLE, then the client can use this as an indication that the point definition has changed. Internally, within the TPN Server, when this error is returned on a read, the point is removed automatically from the internal cache. The client can then re-add the item using the OPC AddItems() method to trigger a re-request of the item definition again from the TPN.

6.8.6 View TPN Server log

Click **View TPN Server Log** to view the current log file in the notepad, as illustrated in the following figure.



6.8.7 Client Connections Data section

The **Client Connections Data** section provides the option to generate a report for individual TPN server clients that are connected to the TPN server.

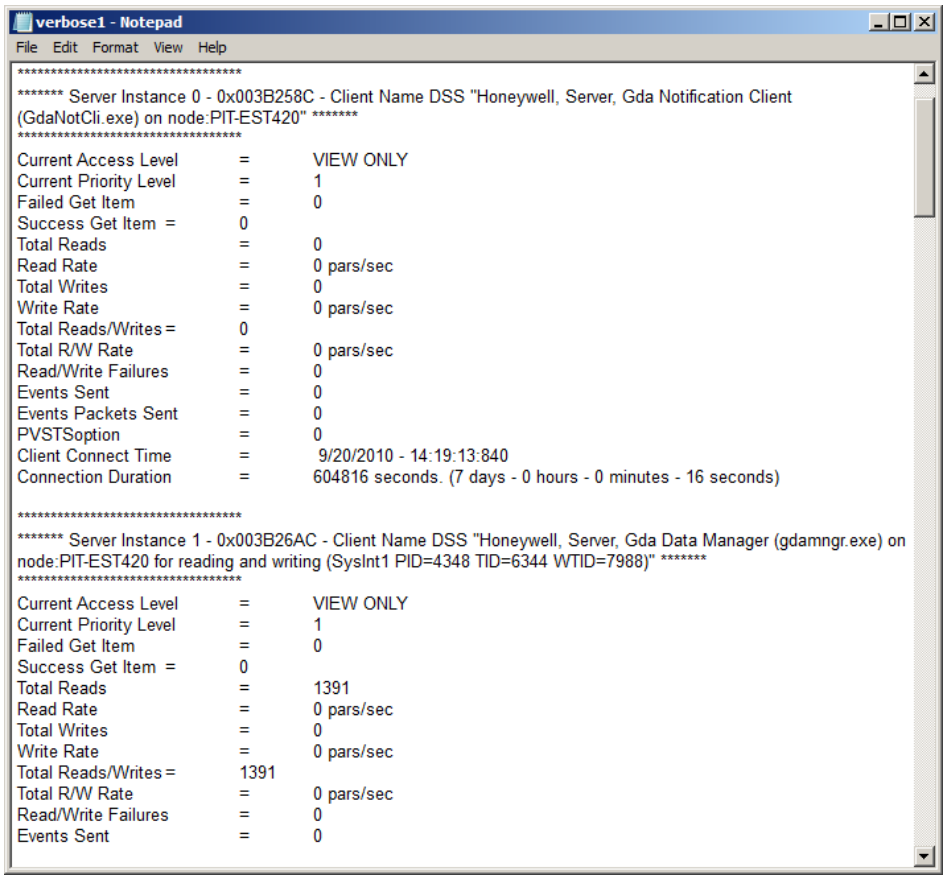
Three types of reports can be generated. They are:

- Summary report
- Detailed report
- Comma Separated Values (CSV) report

The three types of reports are explained with examples:

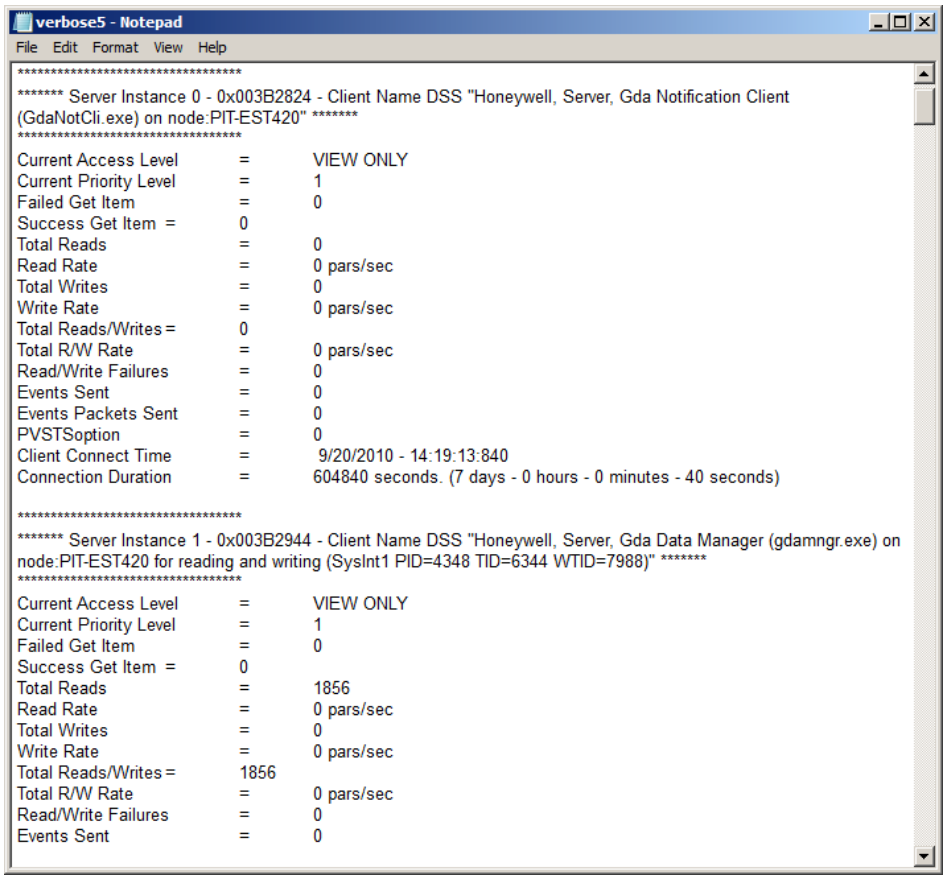
6.8.8 Summary report

This report provides a summary of all the clients that are connected to the TPN Server in text format (viewable in Notepad).



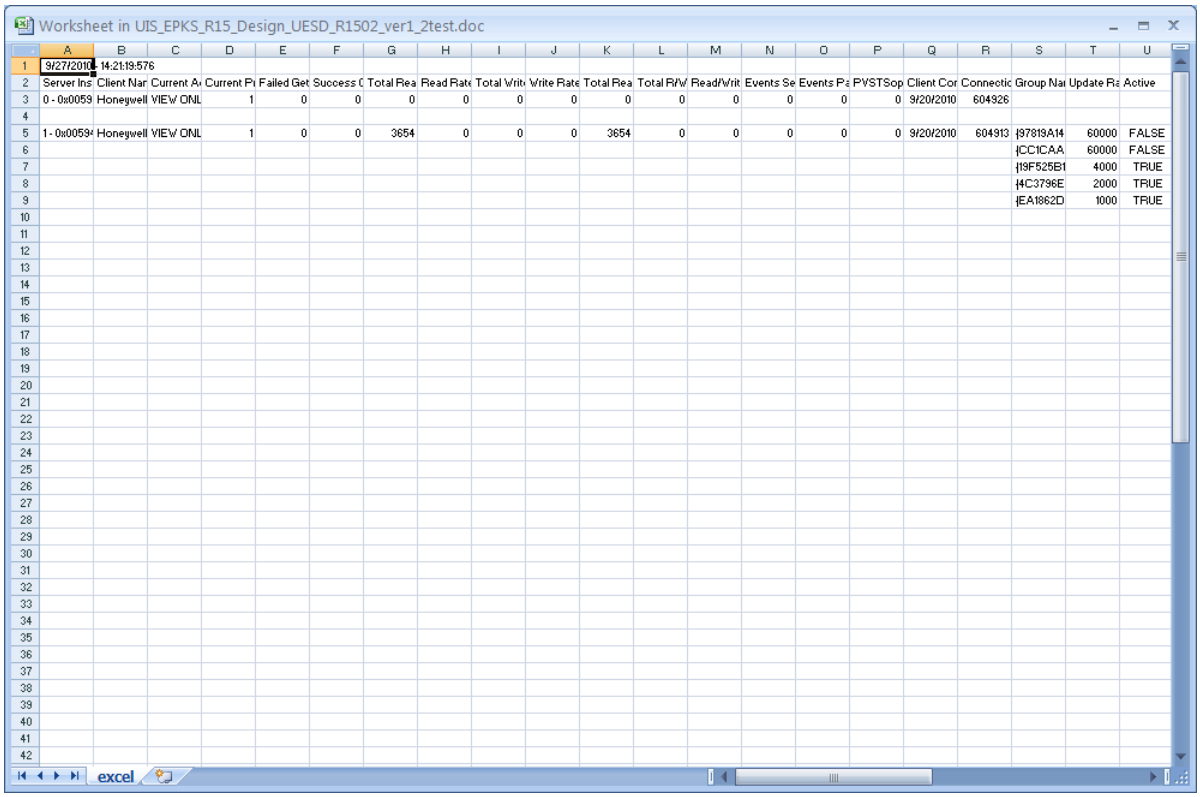
6.8.9 Detailed report

This report provides detailed information of all the clients that are connected to the TPN Server in text format (viewable in Notepad).



6.8.10 Comma Separated Values (CSV) report

This report provides detailed information of all the clients that are connected to the TPN Server in CSV format (viewable in Excel).



6.8.11 Generating reports using the Command Prompt

Reports can be generated using the command prompt.

To generate a report using the command prompt, use the **ServerEnum** command along with the following options:

Option	Description
-V0	Client summary (default),
-V1	Group summary
-V2	Group details
-V3	Item summary
-V4	Item detail
-V5	Item super detail
-Cn	Restricts report to client number n, where n = 0-99 and Default = all clients
-X	Format output as a Comma Separated Value file (csv)
-?	Displays help and exit

For example, to generate a Group Summary report, type **ServerEnum –V1** in the command prompt.

The command **ServerEnum –Vn –Cn** is used for generating a default report that contains complete details of all the connected clients.

Tip
For better viewing, set the **cmd** window width and buffer to **120**. These properties are changed from **Window properties > Layout**.

6.8.12 Performance page

On the Auxiliary Display, click the **Performance** tab to open the page shown in "Figure 11: TPN Server Status Display—Performance Page". This page is revised for the Experion R410 release.

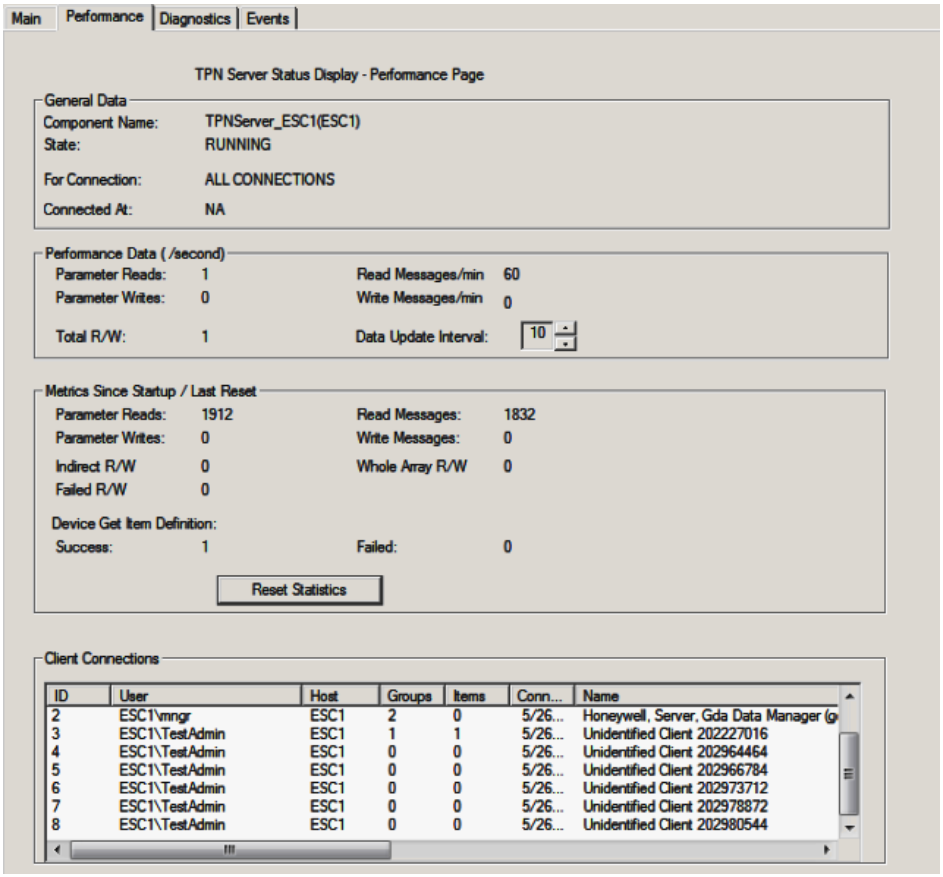


Figure 11: TPN Server Status Display—Performance Page

6.8.13 General Data section-Performance page

The **General Data** section shows the **Component Name** and **State** as on the **Main** page. **For Connection** displays the selected client. **Connected At** displays the time at which the client connected to the TPN Server.

6.8.14 Performance Data (/second) section

The **Performance Data (/second)** section provides the following information:

- **Device Data Access**— Displays the Parameter Reads (per second) and Parameter Writes (per second) from the TPN Network. This is an average value over the **Data Update Interval**. It also displays the Total Reads and Writes (Total R/W) and the TPN message reads (per minute) and TPN message writes (per minute).

- **Data Update Interval**—For usability, the **Performance Data** section has its own update interval. The status display collects the data on a one second sampling period. The **Data Update Interval** determines when the data in the Performance Data section is updated. The updated values are the average of all samples taken during the specified update interval.

6.8.15 Metrics since Startup/Last Reset section

The **Metrics since Startup/Last Reset** section provides the following information:

- **Device Data Access**—total number of parameter reads and parameter writes since startup. The **Failed R/W** displays the total number of parameters returning failed errors codes, while being accessed, is displayed. This counter reports the total number of items across both reads and writes which were not successfully read or written. A non-zero value is an indication that the TPN Server was unable to honor the client request due to overloading of the server or malfunction of the TPN or the TPN Server connection to the TPN. A non-zero value may be an indication that the TPN data owner (such as an AM) is down. It is recommended that the system (TPN, total server operation and load, client operations, and so forth.) be audited when non-zero values are seen.
- **Device Get Item Definition**—total number of successful 'get item definition requests,' and failed get item definition requests since startup. Get Item Definition calls generally occur when a point.parameter requested was not found in the checkpoint for the server.

If a 32-bit counter overflow occurs, the number will wrap around to zero and restart. If the value is displayed as '?' then the data is either inaccessible or the TPN Server has failed. In the event that the TPN Server has failed, all values will display '?', with the exception of the Component Name.

With Experion R410, the **Reset Statistics** button provides the option to reset statistics of the selected connection to zero.

6.8.16 Client Connections section

With Experion R410, individual client connections to the TPN Server are selected and monitored from a list of current connections under the **Client Connections** section. There is also an option to view and monitor the cumulative values of all the current connections to the TPN Server by selecting the **ALL CONNECTIONS** option from the list. The **Client Connections** list displays a maximum of 50 client connections at a time.

When a connection is selected from the list, data access statistics for the selected connection(s) is displayed in the **General Data**, **Performance Data** and **Metrics Since Startup/Last Reset** sections.

Tip

When the **TPN Server Single Scan** option is enabled, only the individual client data collected in the READ_FROM_DEVICE manner and the write statistics is displayed. Data subscribed by the Client is aggregated (together with other clients) under a special Single Scan client. This internal client combines all the requests from the active groups into a single collection group.

6.8.17 Diagnostics page

On the Auxiliary Display, click the **Diagnostics** tab to open the page shown in “Figure 12: TPN Server Status Display—Diagnostics Page”. This page is revised for Release 310.1.

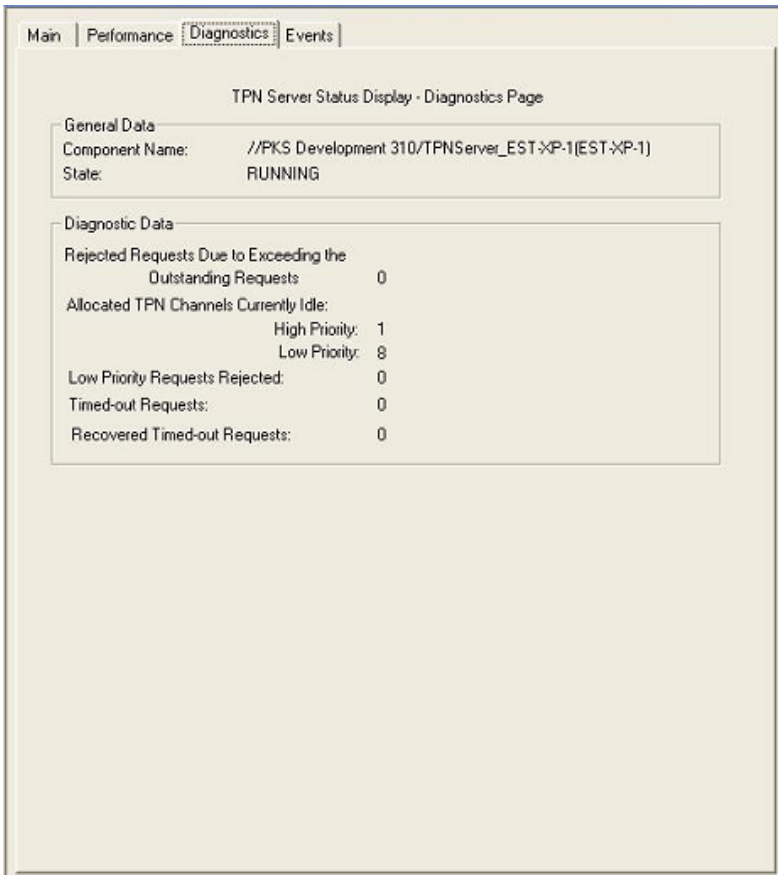


Figure 12: TPN Server Status Display—Diagnostics Page

6.8.18 General Data section-Diagnostics page

The **General Data** section shows the **Component Name** and **State** as on the **Main** page.

6.8.19 Diagnostic Data section

The **Diagnostic Data** section provides the following information:

- **Rejected Requests Due to Exceeding the Outstanding Requests Permitted**—On the **Channels** configuration page, the administrator can specify the number of Outstanding Requests permitted. When the sum of the High and Low Priority Queued Requests (displayed on the Main page) exceeds the number of Outstanding Requests, the request is rejected.
- **Allocated TPN Channels Currently Idle**—number of allocated high, and low priority TPN channels that are currently awaiting a request to process. This metric can be used to fine tune the number of high, and low priority channels configured for the TPN Server, or to change the loading of the TPN Server. If the number, for either high or low priority, is consistently zero, consideration could be given to increasing the number of channels configured, for the given priority. If this number is consistently greater than one, consideration could be given to reducing the number of channels configured, for the given priority.
- **Low Priority Requests Rejected**—number of low priority requests rejected because there were no low priority TPN channels allocated. In order for the TPN Server to process a low priority request, a low priority TPN channel must be configured and allocated. (Note: This restriction does not apply to high priority requests. If there are no high priority TPN channels configured, a high priority request can be sent on a low priority channel.)

- **Timed-out Requests**—number of TPN Server requests that have not been serviced within an eight-minute time frame. When a request is not serviced, there is a high probability that one, or more TPN channels are hung. This situation may lead to poor TPN Server performance. This metric should be used in conjunction with the Recovered Timed-out Requests metric. In the event, where timed out requests have occurred, and the TPN Server subsequently recovered from the timeout (timed-out requests = recovered timed-out requests), the TPN channels will no longer be hung.
- **Recovered Timed-out Requests**—number of TPN Server requests, that had timed-out, but were eventually processed. See the explanation on Timed-out Requests for more information.

Note: If the value is displayed as '?' then the data is either inaccessible or the TPN Server is in the Warning state or has failed. In the event that the TPN Server has failed, all values will display '?', with the exception of the Component Name.

6.8.20 Events Page

On the Auxiliary Display, click the **Events** tab to open the page shown in "Figure 13: TPN Server Event Performance Page". This page is revised for Release 310.1.

Note: The **Events** page is provided *only* when the TPN Server is installed on a ES-T or ESVT node.

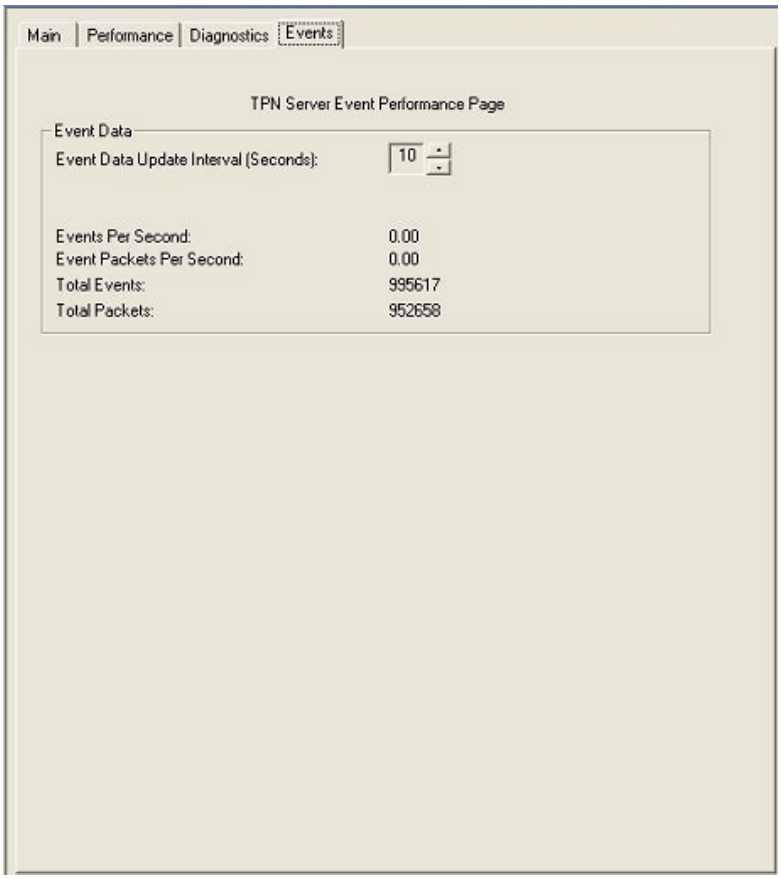


Figure 13: TPN Server Event Performance Page

6.8.21 Event Data section

The **Event Data** section shows the following information about TPN server throughput:

- **Event Data Update Interval (Seconds)**—status display collects the data on a one-second sampling period. The Update Interval determines when the data in the Events Data section is updated. The updated values are the average of all samples taken during the specified update interval.

- **Events Per Second**—indicates the rate that alarms and events are processing through the TPN Server. The total number of individual alarms and events that were processed during the screen update period are divided by the screen update rate.
- **Event Packets Per Second**—indicates the rate that alarm and event packets (lists of alarms) are processing through the TPN Server. Alarms and events are received in packets of one to many alarms per packet. The total number of alarm packets that were processed during the screen update period is divided by the screen update period. The event packets per second compared to the events per second can give an indication of packet overhead. Consider that ten events per second and ten packets per second indicate a different load than ten events per second and one packet per second. In the first case. the TPN Server received ten alarm packets with only one alarm in each packet, and it had to call the connected clients ten times in that time period. In the second case, the TPN Server received one packet with ten alarms and called the connected clients once in the time period.
- **Total Events**—an ever-increasing counter (32-bit unsigned) that indicates the total number of alarms and events processed through the TPN Server since it was started. This counter rolls over to zero when it reaches the limit of a 32-bit unsigned number (4,294,967,295).
- **Total Packets**—an ever-increasing counter (32-bit unsigned) that indicates the total number of alarm and event packets processed through the TPN Server since it was started. This counter rolls over to zero when it reaches the limit of a 32-bit unsigned number (4,294,967,295).

Attention

Calculating the values for Events per Second and Event Packets per Second over the screen update period has a smoothing effect. For example, if the screen updates every 4 seconds and 8 events come in during the first second followed by none for the remaining 3 seconds, the events per second is set to 8 divided by 4 or 2 events per second. As opposed to calculating the rate every second which would show 8 events per second for one second and then zero events per second for 3 seconds.

7 Accessing TPN data using the TPN Server

This section describes how the OPC or HCI client uses the OPC/HCI interfaces to access TPN data. It describes the TPN Server specific interface. It describes name forms and data types specific to the TPN Server.

OPC interfaces

The client will have to use the OPC/HCI interfaces to read TPN data from the TPN Server cache or from the device, and to write TPN data. This section assumes the client has already established connection to the TPN Server using the HCI Client APIs.

For more information about OPC interfaces, see the *HCI Client Developer's Guide* and the *OPC Data Access Specification Reference Manual*.

TPS network connection required to access data

A connection to the TPS Network is vital in order for the TPN Server to correctly function. Through this connection, the TPN Server gets access to TPS Network data. In order for this connection to occur, a TPS Network personality must be loaded on the LCNP.

TPN Server initialization and startup behavior

During Initialization, the TPN Server attempts to connect to the TPS Network:

- If the connection is successful, the TPN Server transitions from the Warning state to the Idle or Running State where it allows client connections and requests.
- When the connection is unsuccessful, the TPN Server transitions to the Warning state. In this state, it returns an 'E_FAIL' HRESULT to any client that attempts to connect. Upon establishing a connection to the TPS Network, the TPN Server transitions from the Warning state to the Idle or Running state where it allows client connections and requests.

TPN Server runtime behavior

During run-time, the state of the connection to the TPS Network is continuously checked. In the event that the connection is broken, the following occurs:

- The TPN Server transitions from the Idle or Running state to the Warning state.
- All current client connections with the TPN Server are maintained, but all OPC requests that are made by a client to the server fail, and the server returns an 'E_FAIL' HRESULT.
- All new client connection requests fail and return an 'E_FAIL' HRESULT.

When the connection is re-established, the following occurs:

- The TPN Server transitions from the Warning state to the Idle or Running state.
- All OPC requests made by a client are now processed.
- All new client connections are processed.

7.1 Performance considerations

Related topics

“How the Single Scan per Data Item option works” on page 88
“Disabling/enabling the Single Scan per Data Item data access option” on page 88
“Performance Monitor Counters” on page 89

7.1.1 How the Single Scan per Data Item option works

On Experion APP nodes, the TPN Server offers optional functionality to boost performance by consolidating the polling of all items into a single read each cycle. Note: on other Experion nodes, this functionality is provided by the clients of the TPN Server, eg in the Experion server. This optional function is automatically enabled in the TPN Server when configured to use 'Single Scan per Data Item'. In the default state of disabled, the TPN Server performs a device-specific read requesting all active items for each active group, in turn, at their respective update rates. The server is obligated to service each of these requests even though several groups may be requesting the same data.

After enabling the option, the TPN Server reads all necessary items from the device in a single read and updates the cache with those values. It then fires all groups that are due to run. The groups then read from cache. This consolidation eliminates the considerable overhead of numerous device reads each cycle. The mechanism uses a single timer to controls both the device reads and the group updates to avoid synchronization problems that can occur with the multiple timers. Group callback functionality behaves the same as when the option is disabled.

The timer runs at the fastest UpdateRate allowed by TPN Server (currently one second). At each timer cycle, the following tasks are performed:

- The TPN Server builds a list of all items that are 'due' to be read. This list uniquely includes all items regardless of the containing groups. This list is maintained in an internal database that is modified whenever items are added or deleted from groups, when items or groups are activated or inactivated, and when group UpdateRates change.
- The TPN Server performs a device read of the items in the list and updates the server cache with the returned values.
- The TPN Server 'fires' all groups that are due to run. An internal database maintains the elapsed time of each group so that a single timer controls when each group runs.
- The groups, when fired, read from cache and handle callbacks in the same manner used in 'normal' polling.

Because a single, 1-second timer controls group execution, group UpdateRates that are not multiples of 1000 milliseconds will effectively be 'rounded up'. For example, a group with an UpdateRate of 1500 milliseconds will run every two seconds.

7.1.2 Disabling/enabling the Single Scan per Data Item data access option

For OPC OnDataChange clients, each group that has active items independently triggers a device read (from the TPN) each update rate. With the single scan option disabled, active groups function independently of each other and each adds a load on the device (that is, on each update rate expiration within a group, the server requests the data from the TPN and reports it back to the client). With the single scan option disabled; the client gets fresh data within the requested update rate as long as the server does not get overloaded with requests. As a rule of thumb, a server will become overloaded if the total number of items being collected by all clients exceeds 800 per second.

Additionally, each active OPC group in the system imposes a load on the TPN due to the expensive nature of setting up a collection group on the TPN. To overcome this TPN loading, the Single Scan per Data Item option can be enabled.

With the Single Scan per Data Item option enabled, the server optimizes data access to the TPN by consolidating all of its clients' requests into one single request. This optimization minimizes the expensive

creation of TPN collection groups. In addition, duplicate point names within one group and within all the groups get collected in the consolidated list only once.

Attention

The **Single Scan** option is not a way to generally boost performance. It must only be turned on in very limited cases, mainly on the Experion APP nodes, with multiple client connections to the TPN server that are retrieving data with the same collection rates.

7.1.3 Performance Monitor Counters

The TPN Server exports three types of performance counters—r*aw counter*, d*elta counter*, and *rate counter*. They are accessible through Windows Performance Monitor only after the server is running. Each type is described as follows and detailed in “Table 7: Types of Performance Counters”.

- **Raw Counter**—provides a 'raw' count without regard to the previous value or elapsed time between updates.
- **Delta Counter**—reports the difference from one sample to the next. Internally, the counter value is a continuously increasing number while the value displayed is the difference between the current value and the value obtained at the time of the previous sampling. The time interval between samples is completely user configurable (example: using perfmon, click **Options** > **Chart** and then set the **Periodic Update Interval**).
- **Rate Counter**—similar to the Delta Counter except that the difference from one sample to the next is also divided by the delta time before being displayed.

Table 7: Types of Performance Counters

Counter name	Counter type	Counter description
DSS: Client Connections	Raw Counter	The number of clients currently connected to this server. This value is also available through the TPN Server Auxiliary Status display.
DSS: Active List Size	Raw Counter	The number of active items currently in the TPN Server's ActiveList. This number indicates the load on the TPN due to 'OnDataChange' clients.
DSS: Definition Requests Failed	Delta Counter	Before the initial reading of an OPC item, the TPN Server must verify that this item maps to (defines) a valid TPN data point. This counter reports the number of items, which *could not be* mapped to a corresponding TPN data point. Ideally, this counter will be zero. If a non-zero value is reported, that means a client has added points, which do not exist on the TPN. The additional load imposed on the TPN can be avoided by determining which client is requesting the bad points and removing the points from the list.
DSS: Definition Requests Successful	Delta Counter	This counter reports the number of items, which *could be* mapped to a corresponding TPN data point. This counter indicates which new items are being added to the TPN Server. Only items that are not already in the cache are added to this count. Items that already exist, will be successful, but will not be reflected in this count.
DSS: Reads Per Sec	Rate Counter	This counter reports TPN Server device (that is, TPN) read requests expressed as specific point read requests per second (items of the type, 'array', are counted as single points). The value shows the total of all device-read activity from demand-read clients (both synchronous and asynchronous) as well as internally initiated from the ActiveList. This counter indicates the current load imposed by the TPN Server on the TPN. If the number of clients performing reads increases, this number increases and eventually peaks/levels-off when the maximum load the TPN can service through the allocated channels has been reached. At this point, the 'DSS:Sum of Queued Requests' counter should be monitored to determine if messages are being queued because maximum throughput has been reached. **Note:** Read requests using large OPC group sizes yield better overall throughput.

Counter name	Counter type	Counter description
DSS: Writes Per Sec	Rate Counter	This counter reports TPN Server device (that is, TPN) write requests expressed as specific point write requests per second (items of the type 'array' are counted as single points). The value shows the total of all device write activity from demand-read clients (both synchronous and asynchronous). This counter indicates the current load imposed by the TPN Server on the TPN. If the number of clients performing writes increases, this number increases and eventually peaks/levels-off when the maximum load the TPN can service through the allocated channels has been reached. At this point, the 'DSS:Sum of Queued Requests' counter should be monitored to determine if messages are being queued because maximum throughput has been reached. **Note:** Write requests using large OPC group sizes yield better overall throughput.
DSS: Reads/Writes Per Sec	Rate Counter	This counter reports the sum of counters 'DSS:Reads Per Sec' and 'DSS:Writes Per Sec'. Refer to the descriptions of these two counters for further explanation.
DSS: Read/Write Item Failures	Delta Counter	Counters 'DSS:Reads Per Sec' and 'DSS:Writes Per Sec' report the rate of point read/write requests without regard to the number of points that were not successfully read or written. This counter reports the total number of items across both reads and writes that were not successfully read or written. A non-zero value indicates that the TPN Server is unable to honor the client request due to a server overload or a TPN malfunction or a TPN Server connection to the TPN. A non-zero value may be an indication that the TPN data owner (such as, AM) is down. It is recommended that the system (TPN, total server operation and load, client operations, and so forth.) be audited when non-zero values are seen.
DSS: Sum of Queued Requests	Raw Counter	This counter reports the sum of the queued high and low priority requests waiting to be processed by the TPN Server. For optimum performance, this number should be low. A value of zero means that requests are immediately processed. Values greater than zero mean that requests are being queued until a channel becomes available. Corrective action must be taken when the sum of queued requests exceeds the number of Outstanding Requests configured for the given TPN Server. In this event, the TPN Server throttles requests until the total number of queued requests no longer exceeds the number of Outstanding Requests. The number of Outstanding Requests is configured on the Channels Configuration page for the TPN Server.
DSS: High Priority Channels Idle	Raw Counter	High Priority TPN Channels Idle is the number of high priority channels currently awaiting a request to process. This metric can be used to fine-tune the number of high priority channels configured for the TPN Server. If this number is consistently zero, consideration should be give to increasing the number of configured high priority channels. If this number is consistently greater than one, consideration should be given to reducing the number of configured high priority channels.
DSS: Low Priority Channels Idle	Raw Counter	Low Priority TPN Channels Idle is the number of low priority channels currently awaiting a request to process. This metric can be used to fine-tune the number of low priority channels configured for the TPN Server. If this number is consistently zero, consideration should be given to increasing the number of configured low priority channels. If this number is consistently greater than one, consideration should be given to reducing the number of configured low priority channels.
DSS:Events Per Sec[1]	Rate Counter	This counter reports TPN Server alarms and events received from the TPN side per second.
DSS:Event Msgs Per Sec[1]	Rate Counter	This counter reports TPN Server alarm and event messages received from the TPN side per second.
DSS:Events Raw[1]	Raw Counter	This counter reports the raw number of events received from the TPN side since startup or counter rollover.
DSS:Event Msgs Raw[1]	Raw Counter	This counter reports the raw number of alarm and event messages received from the TPN side since startup or counter rollover.
DSS:Items in Cache	Raw Counter	This counter reports the raw number of items in the TPN Server (HCI) compact cache.

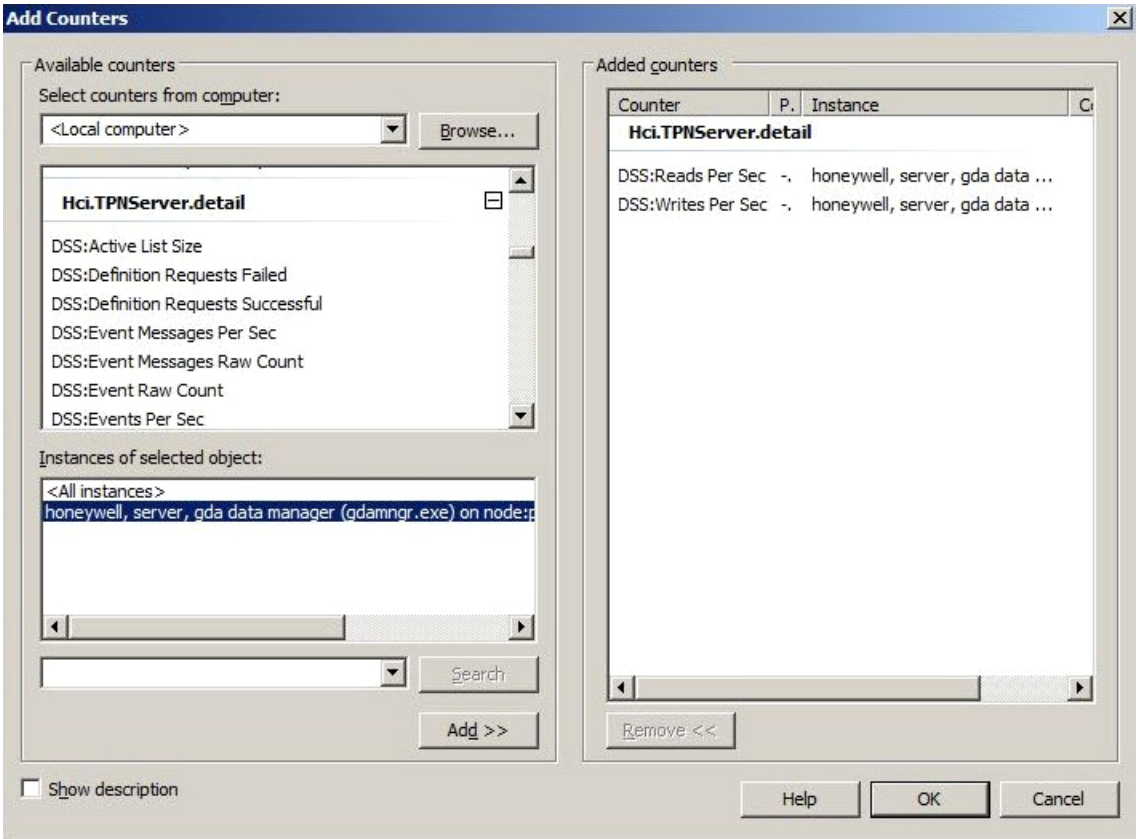
Counter name	Counter type	Counter description
DSS:Indirect per Sec	Rate Counter	This counter reports the number of items that use Indirection that are being read per second. Indirection requires one pass to read the value of the secondary point and another pass to read the final value.
DSS:Whole array/Blind Rec Per Sec	Rate Counter	This counter reports the number of items that are being read as a whole-array-at-once or are the complex (blob) type of parameter known as a *blind record* on the LCN. Due to their potential size, these parameters are read one at a time even though they may be presented to TPN Server in a list.
[1] These particular counters apply to a TPN Server installed on LCN-connected Experion nodes.		

To view TPN Server counters

1. Start the TPN Server and then start the Windows Performance Monitor ('perfmon.exe').
2. With Performance Monitor running, click **Edit** > **Add to Chart**. If the TPN Server is running on a remote node, do the following steps.
   a. Enter the remote computer name.
   b. Select the counter or counters to be monitored and then click **Add**.

   Once the Performance Monitor is running, the TPN Server may be stopped and restarted without needing to stop/restart Performance Monitor.

   If both the TPN Server and the Performance Monitor are stopped, the TPN Server must be re-started first before the Performance Monitor is restarted.



Tip

- With Experion R410, individual client performance counters are available in the **Windows Performance Monitor** application.
- A Global Instance is available if the **Single Scan** option is enabled.

7.2 TPN Server Client OPC interfaces

Related topics

7.2.1 OPC topics described herein are TPN Server specific

The client will have to use the OPC interfaces as discussed in the 'HCI Client Developer's Guide' to read TPN data from the TPN Server cache or from the device, and to write TPN data. This section only covers topic specific to TPN Server such as the format of the OPC item definitions, and the setting of TPN access level and server priority level. It assumes that the client has already established connection to the TPN Server using the DCOM and/or HCI Client APIs as discussed in the 'HCI Client Developer's Guide.'

7.2.2 Name Forms for server specific portion

The server specific portion of the TPS Name for the TPN Server must be of the form [lcnID.]point.p or [lcnID.]point.p.p, where p may be of the form param or param (I). The index, I, must be an integer. The lcnID is optional, and is required only if accessing data from a second TPN that is connected through a Network Gateway.

The HCI TPN Server supports the standard TPS name form of:

//TPS_Domain_Name/Server_Name/Server_Specific

The Server_Specific portion of the name conforms to TPN name syntax as follows:

[lcn\]point.parameter [(i)][.parameter[(j)]][.property]

where items in brackets [ ] may be omitted. The indexes 'i' and 'j' are integer literals. Index values are constant for the life of an OPC item definition. A single level of indirection is supported.

7.2.3 TPN data access using names forms

When TPN data is accessed in the form [lcn\]point.parameter [(i)] the access is a direct access and the data returned will be in the form of one of the TPN Data Types.

When TPN data is accessed in the form [lcn\]point.parameter [(i)].parameter[(j)], the access is a single level indirection access. When this type of access occurs, the TPN data type of the first accessed point.parameter[(i)] must be of type entity ID.

When TPN data is accessed in the form [lcn\]point.parameter and the parameter is an array, the whole array is indicated. In other words, to refer to an entire array as a single item, omit the index and parentheses.

7.2.4 Storing enumeration name form with property option

The [.property] option is used to store Enumeration name form. The syntax for the Enumeration name form can be any of the following:

For internal property:

- point.parameter.internal
- point.parameter1.parameter2.internal

For external property:

- point.parameter
- point.parameter.external
- point.parmeter1.parameter2
- point.parameter1.parameter2.external

In rare cases where the actual indirected parameter name is **internal** or **external**, the syntax can be either of the following:

- point.parameter./internal (meaning the parameter ‘internal' not the property)
- point.parameter./external (meaning the parameter ‘external' not the property)

7.2.5 Array access characteristics

Whole array access supports the following characteristics:

- **Data types**—TPN Server supports the following TPN network array types:
  - Arrays of Integers, 16 bits signed
  - Arrays of Real, 32 bits IEEE
  - Arrays of Standard and Self-Defining Enumeration, 8 bytes (external form)
  - Arrays of Standard and Self-Defining Enumeration, 2 bytes (internal form)
  - Arrays of Boolean, 8 bits
  - Arrays of Times, 48 bits
  - Arrays of Entity Ids, 64 bits (see data types for special indexing considerations.)
  - Arrays of Strings are not supported
  - Arrays of Blind Record are not supported
- **Whole array**—TPN Server provides access to OPC items that are entire TPN arrays.
- **Multiple arrays**—TPN Server supports multiple array read/write requests within a single OPC group.
- **Items mix**—TPN Server supports any combination of whole arrays, individual array items, or any non-array items, all within a single OPC group.
- **Array indexing**—array indexing is identical to the TPN Network array indexing. TPN Network array indexing is nonzero based. Note that using TPN Network enumeration type as array indexes is not supported.
- M**aximum size**—maximum size of an array that can be read in this manner is limited to 1023 items. This is a TPN Network limit.
- **Array segments**—not supported.

Some data owners on TPN do not support stores of whole arrays for certain parameters. If this is the case, whole array stores from the TPN Server also will not work. This is a TPN limitation. Refer to the TPN parameter dictionaries for more information.

7.2.6 TPN data types mapping to HCI/OPC data types

"Table 8: Supported TPN Data Types" shows the TPN data types that are supported and how they are mapped into HCI/OPC data types. Coercion from these canonical data types to a client's requested data type is handled by the HCI GOPC. HCI uses the Microsoft *VariantChangeType()* API to transform these canonical data types to the requested data type.

Table 8: Supported TPN Data Types

TPN type description	HCI/OPC type
Real - 32 bit IEEE format	VT_R4
Standard and Self Defining Enumeration - 14 bytes	External form [1]: VT_BSTR Internal form: VT_I2
Boolean - 8 bits (0 or 1)	VT_BOOL
Time - 48 bits	External form [2] : VT_DATE Internal form: VT_R8
Entity ID - 64-bit unsigned	VT_ARRAY (VT_VARIANT(VT_I4)), 2 elements
String - 42 bytes	VT_BSTR
Integer - 16-bit signed	VT_I2
Array of Integers, 16-bit signed - 1023 items max	VT_ARRAY of VT_I2
Array of Real, 32-bit IEEE - 1023 items max	VT_ARRAY of VT_R4
Array of Standard and Self Defining Enumeration, 14 bytes - 1023 items max	External form [3]: VT_ARRAY of VT_BSTR Internal form: VT_ARRAY of VT_I2
Array of Boolean, 8 bit - 1023 items max	VT_ARRAY of VT_BOOL
Array of Times, 48 bits - 1023 items max	External form[4]: VT_ARRAY of VT_DATE Internal form: VT_ARRAY of VT_R8
Array of Entity IDs, 64 bit - 1023 items max	VT_ARRAY of VT_I4, 2 elements per Entity ID [5]
Blind Record - 1 K words max.	VT_ARRAY (VT_VARIANT (VT_UI1))

[1]The external form is the default. The internal form may be explicitly requested by appending an 'internal' property indicator to the parameter name.

[2]The external form is the default. The internal form may be explicitly requested by appending an 'internal' property indicator to the parameter name. The internal form is used to indicate a time interval as opposed to a date/time. The units of the interval are 'days'. For example: an internal value of 3.14159625 would indicate an interval of 3 days, 3 hours, 23 minutes, 53.9376 seconds. Note that TPN time values have a maximum accuracy of 1/10,000 of a second.

[3]The external form is the default. The internal form may be explicitly requested by appending an 'internal' property indicator to the parameter name.

[4]The external form is the default. The internal form may be explicitly requested by appending an 'internal' property indicator to the parameter name. The internal form is used to indicate a time interval as opposed to a date/time. The units of the interval are 'days'. For example: an internal value of 3.14159625 would indicate an interval of 3 days, 3 hours, 23 minutes, 53.9376 seconds. Note that TPN time values have a maximum accuracy of 1/10,000 of a second.

[5]Each entity ID takes 2 VT_I4 to specify. Therefore, the array appears to have twice as many members as it actually has on the LCN. To examine each entity ID one must examine 2 adjacent elements as a unit (For example: lVal[0] and lVal[1] represent one entity ID in the array).

The TPN Server supports access of those parameter types except for Blind Record, which requires the LCN side to be at least Release 600. Access of an entire array as single item is supported. The TPN Server can access individual elements of an array parameter type as individual items.

7.2.7 Special data conversions

For several special data types, the TPN Server ‘overrides' the VariantChangeType() and does a custom TPN-specific conversion. These conversions can be done with or without ‘.internal' or ‘.external' notation. Following the VariantChangeType() model, only single values can be converted. A whole array cannot be converted to an array of the requested type. These data types are listed below.

- **Enumerated values** can be referred to by either the ordinal value (VT_I2) or the string value (VT_BSTR). The TPN Server uses the EUType information (see the following ' “Engineering units” on page 96' section) to access the appropriate value in either string or numeric form from the actual enumeration members.
- **Boolean values** can be referred to by either a Boolean value (where VT_BOOL -1 = true and 0 = false) or by the string value (VT_BSTR) **OFF** or **ON**.
- **Entity ID values**(for example, PRIMMOD) are stored on the TPN as the entity handle rather than the string name of the entity. Therefore, the entity parameter's natural type is an array of two VT_I4 values, which is the size of a TPN entity handle. The TPN Server converts this handle to string (VT_BSTR) form for both reads and writes of entity type variables. Since the TPN Server must make an additional trip to the TPN in order to do these conversions, there is some performance degradation for the non-converted point ID.
- **NaN values**—The Microsoft API, VariantChangeType(), does not correctly convert NaN values between 64-bit (VT_R8) and 32-bit (VT-R4) precisions. The TPN primarily uses 32-bit floating-point values. However, many OPC clients prefer the 64-bit form for increased precision. In order to enable the client using 64-bit precision to store a NaN to the TPN (for example, to turn off a value), the TPN Server makes this conversion internally. The NaN conversion maps Signaling NaN and Quiet NaN, to Quiet NaN. The conversion also converts +/- infinity. The BSTR values ‘QNaN' and ‘—' (two dashes) is also converted to VT_R4 NaN. Other real number values continue to use the VariantChangeType() method for conversions.

7.2.8 Engineering units

The OPC specification specifies that the device specific server may optionally return engineering unit information (that is, value range) for analog (real, float) values and enumerated values. The TPN Server support of this feature is limited. Most values do not offer range information. For values that do not offer range information, the dwEUType field of the OPCITEMATTRIBUTES is set to OPC_NOENUM (=0) indicating that fact.

7.2.9 Enumerated values

For enumerated values that are TPN standard, custom, or self-defining enumerations, the dwEUType field of the OPCITEMATTRIBUTES is set to OPC_ ENUMERATED (=2) to indicate that fact. Further, a VARIANT array indexed from zero to the number of enumeration members minus one of VT_BSTR listing the possible enumeration members in ordinal order is set in the vEUInfo field of the OPCITEMATTRIBUTES.

7.2.10 Analog values

For analog values, only a limited number of parameters offer range information. When range information is offered, the dwEUType field of the OPCITEMATTRIBUTES is set to OPC_ANALOG (=1). Further, the vEUInfo field of OPCITEMATTRIBUTES is set to a VARIANT array indexed zero to one of VT_R8. The item indexed zero indicates the low range of the associated real value. The item indexed one indicates the high range of the associated real value. Note that the precision of the range on the TPN is only R4. The R8 level of precision is returned per the OPC specification.

Range information is only returned when the parameter is real and associated with the following TPN point types:

- HG analog in
- HG analog out

- HG analog composite
- CTL counter
- HG Regulatory
- HG counter
- AM Regulatory
- AM Counter
- NIM analog in
- NIM analog out
- NIM Regulatory PV
- NIM Regulatory CTL
- NIM numeric
- NIM IPC analog in
- NIM IPC analog out
- NIM IPC regulatory PV

When the value is real and listed in the previous ranged point types, “Table 9: Parameters that can have range information returned” shows the parameters that can have range information returned and the source of that range information.

Table 9: Parameters that can have range information returned

Parameter	Low-range source	High-range source
PV	PVEULO	PVEUHI
CV	CVEULO	CVEUHI
SP	SPEULO	SPEUHI
Z	ZEULO	ZEUHI
AV	AVEULO	AVEUHI

Note that some points have, so called, ‘extended range' so the parameter may exceed the range under certain circumstances. For example, the PV of HG Regulatory points can have a set of extended range parameters (PVEXEULO, PVEXEUHI) that typically represent -6.9% to 106.9% of the base range. This allows a feature that is sometimes called ‘tight shutoff'. The range referred to here is the base range, not the extended range.

7.2.11 Group size limits

The TPN Server is constrained by the underlying TPN hardware and software. Typically, items are grouped on the TPN network in a TPN specific request packet called an IDB. An OPC group is only limited by the amount of resources available on the node hosting the TPN Server. A group that is larger than the limits of the TPN is broken up internally into smaller IDB requests when collected from the TPN Network. It is then reconstructed back to the original group size when passed back to the requesting client. Through this, the group size limit becomes transparent to the client.

7.3 Setting and reading TPN Server access level and priority level

Related topics

“Reason for setting” on page 98
“TPN Server priority level” on page 98

7.3.1 Reason for setting

An HCI client can explicitly set the access level and server priority level using the SetAttributes method of the IHciAttributes interface. The IHciAttributes interface and its methods are described in reference #6.

The client can also read the current settings of the access level and server priority level using the GetAttributes method of the same interface. By specifying the desired list of attribute names, the client can set or read either one or both levels with a single method call. The attribute names used by the TPN Server are PriorityLevel for setting the server priority level and AccessLevel for setting the access level. They are described in the next section.

When the operation of the SetAttributes call is successful the new access and/or priority levels will only be valid on the current client connection and subsequent requests. The client must set the access and priority levels after each connection or the configured default levels will be used. If a client has multiple OPC Groups in a single connection then all the OPC Groups will have the same access and priority levels. To have different access and priority levels on each OPC Group then the client has to make multiple connections and set the desired access and priority levels for each connection. Details of setting the access and priority levels are discussed below.

7.3.2 TPN Server priority level

The TPN Server Priority level specifies what TPN system data access priority level (function level) operations are to be performed at and the request priority level (TPN channel priority). There are ten server priority levels that you can set to determine whether to use a high- or low-priority TPN channel for communication with the TPN System.

In addition to the TPN channel priority, the Server Priority level will be used to indicate which TPN System Function Level to perform TPN System operations at. The TPN Server Priority Level is set using the SetAttributes method call of the IHciAttributes interface with the attribute name 'PriorityLevel' and an attribute value of an integer variant type (VT_I2) from 1 to 10. The integer set is range checked for a value from 1 to 10 with '1' being the lowest priority and '10' being the highest. The table below indicates the TPN channel priority and TPN System Function level for each of the ten priorities.

Table 10: Server Channel Priority and System Function Level

TPN Server Priority	TPN Channel Priority	TPN System Function Level
1	LOW_RQST_PRIORITY (Low)	$D_normal_display_update_fl
2	LOW_RQST_PRIORITY (Low)	$D_normal_display_update_fl
3	NON_CRITICAL_RQST_PRIORITY (Low)	$D_new_display _fl
4	NON_CRITICAL_RQST_PRIORITY (Low)	$D_new_display _fl
5	MEDIUM_RQST_PRIORITY (Low)	$D_history_collection _fl
6	MEDIUM_RQST_PRIORITY (Low)	$D_history_collection _fl
7	CRITICAL_RQST_PRIORITY (High)	$D_hilevel_control_fetch_store _fl
8	CRITICAL_RQST_PRIORITY (High)	$D_hilevel_control_fetch_store _fl

TPN Server Priority	TPN Channel Priority	TPN System Function Level
9	HIGH_RQST_PRIORITY (High)	$D_control_fetchstore_fl
10	HIGH_RQST_PRIORITY (High)	$D_control_fetchstore_fl

The TPN Server will validate the value being stored and verify if the client has permission to set the requested server priority level. You can verify the capability filename assigned to the requested server priority level using the TPN Server Security configuration page as discussed in the previous section. Then verify the ACL associated with the capability file. An E_ACCESSDENIED error will be returned if security validation fails.

The current setting of the TPN Server Priority Level can be read using the GetAttributes method call of the IHciInterface interface and using the same attribute name as discussed earlier. The return attribute value will be an integer variant type of VT_I2.

7.4 TPN System function levels

Related topics

“About TPN System function levels section” on page 100
“TPN System function levels overview” on page 100
“TPN System function levels importance” on page 100
“TPN System function levels implications” on page 100
“TPN System function levels implementation” on page 100

7.4.1 About TPN System function levels section

Attention

The following information is intended for the advanced user only. The intent is to provide the background information on the function levels and priority scheme of the TPN system and link this information to the operational knowledge of the legacy equipment.

7.4.2 TPN System function levels overview

The TPS Network has a priority scheme in place intended to prevent higher importance parameter accesses from being blocked by lower importance parameter accesses. Furthermore, the scheme allows higher priority requests to be serviced ahead of lower priority requests, in most cases. Function levels are part of the TPN's priority scheme. Function levels are intended to identify the *purpose* of a particular parameter access request so that it may receive the appropriate priority treatment.

7.4.3 TPN System function levels importance

Function levels are important so that your parameter access requests receive the *appropriate* prioritization. The function level sets the priority of your request in job queues on the TPN, causes TPN job queues to be sorted by priority, sets the task priority of the task on the TPN that ultimately services the request, and sets the time out value for the request. An appropriately chosen function level will cause your request to get serviced at the appropriate level of importance on the TPN, while minimizing your request's impact on other requests on the TPN.

7.4.4 TPN System function levels implications

Parameter access requests of a given function level compete for resources on an even keel with other requests at the same level. They 'bump down' requests that have a lower priority function level. They may, in turn, be bumped by requests of a higher priority function level. Because of this bumping behavior, abuse of the higher priority function levels can have detrimental effects. Such bad effects include: missed screen updates, gaps in history collections, interruption of level 2 control, delayed event detection, or even loss of view. Incorrectly choosing a low priority could impose long time delays on the request if the request encounters the time out scenario. Incorrectly choosing a high priority for a large request could cause the request to often encounter a time out since high priorities tend to have shorter time out values.

7.4.5 TPN System function levels implementation

The main design of the TPN is to grant highest priority to a 'path to the valve.' This highest priority is given to functions the process operator must perform to control the process, especially in upset conditions. The next priority is 'level 2 control' from the Application Module, CG, and so forth. Following that, a priority is provided for history data collection, then status displays. The lowest priority is for 'off line activities' like point building.

In “Table 11: TPN Function Levels” that follows, the **Q** column shows the priority assigned to the message when it is in a queue. Higher numbers mean more power. Queues are sorted so that the higher priority items are serviced first. The **TO** column reveals the associated timeout value in minutes:seconds:fractions format. A TPN parameter access request first checks to see that the destination appears to be running before sending the request, so that the request does not time out needlessly. Requests that do encounter a time out are retried one time so that the total time that a request might be outstanding before returning time out error can be up to twice the time out value.

Note that the highest priority items include the path to valve actions like ramping a set point with the raise lower key. There are also a couple of high priority items that Hiway Gateways and NIM nodes use to do a self-test to assure that the parameter access requests continue to flow. The tasks that detect and report events in the HG and NIM run at a priority between the tasks that service queue level 10 and level 9 parameter requests—below that is control fetch store. This is the function level that the Application Module uses to do level 2 control actions. The CG uses high-level control fetch store for its high priority requests, while using history collection for its low priority requests. The History Module collects historical values using the history collection function level. Following are the function levels for the first invoke of a picture and its subsequent periodic updates. Finally, there are the 'off line' function levels used by the DEB, the system documentation tool, History Module disk synchronization, and so forth. Function levels in bold font correspond to function levels that are available to the TPN Server**.**

Table 11: TPN Function Levels

Q	TO	Function Level (fl)
10	0:5.0	$D_schematic_store_fl
10	0:25.0	$D_HG_periodic_communication_check_fl
10	0:25.0	$D_HG_demand_communication_check_fl
10	0:2.0	$D_operator_demand_action_fl
10	0:1.0	$D_raise_lower_fl
10	0:1.0	$D_manual_valve_operation_fl
9	0:4.0	**$D_control_fetchstore_fl**
9	0:1.0	$D_control_local_fetchstore_fl
8	0:5.0	$D_display_update_following_change_fl
8	0:3.0	**$D_hilevel_control_fetch_store_fl**
7	0:2.0	**$D_history_collection_fl**
7	0:5.0	**$D_new_display_fl**
6	0:15.0	$D_batch_history_fl
6	0:8.5	**$D_normal_display_update_fl**
6	0:15.0	$D_trend_pen_fl
6	2:10.0	$D_load_store_fl
6	45:0.0	$D_slow_load_store_fl
5	1:0.0	$D_checkpoint_fl
5	1:0.0	$D_point_build_fl
5	0:20.0	$D_query_fl
5	1:0.0	$D_smcc_fl
5	0:15.0	$D_slow_display_update_fl
5	0:15.0	$D_query_collect_fl

7.5 TPN System access level

Related topics

“Background on setting TPN System access” on page 102
“Access levels description” on page 102
“TPN Server error returned” on page 103

7.5.1 Background on setting TPN System access

The TPN System Access level specifies at what Access privilege the data access operations are to be performed. You can set eight access levels. The TPN System Access Level is set by using SetAttributes method call of the IHciAttributes interface with the attribute name 'AccessLevel,' and an attribute value of BSTR variant type. The BSTR variant type must have one of the following strings:

- View Only
- Operator
- Supervisor
- Engineer
- Program
- Continuous Control
- Point Builder
- Intimate User

The TPN Server will validate the string without case sensitivity and verify if the client has permission to set the requested access level. You can verify the capability filename assigned to the requested access level using the TPN Server Security configuration page as discussed in the previous section. Then verify the ACL associated with the capability file to verify that the account running the client has 'RX' access permission. An E_ACCESSDENIED error will be returned if security validation fails.

The current setting of the TPN System Access Level can be read using the GetAttributes method call of the IHciAttributes interface and using the same attribute name as discussed earlier. The return attribute value will be a BSTR variant type with one of the access level strings in the previous list.

7.5.2 Access levels description

The TPS Network uses an access level security scheme to prevent unauthorized changes to parameters on the TPN. Access levels only apply when a TPN parameter value is being stored—they have no effect on reads. Each parameter on the TPN has an associated lock applied to it when the parameter is defined at system build time. In order to store a new value to a TPN parameter, the store request must possess a key to open that lock. The keys have a hierarchy; that is, a key can open all the locks at or below its power level, with certain exceptions for Program and Continuous Control. For example, an engineer key can open the operator lock because it is a more powerful key than the operator key. Conversely, the operator key cannot open the engineer lock because it is not powerful enough. “Table 12: Security Choices” describes the security choices.

Table 12: Security Choices

Key	Description
$D_view_only	The weakest key / the strongest lock. This key does not open any lock. Parameters with this lock are read only or are calculated.
$D_operator	The key/lock combination most often used for everyday operations. This is the key that is used on the TPN when the keyboard lock is in the operator position.

Key	Description
$D_supervisor	The key/lock combination used for unusual daily operations. Parameters with this lock have a bit more risk associated with changes than other daily operations. This is the key that is used on the TPN when the keyboard lock is in the supervisor position. This key also opens the operator lock.
$D_engineer	The key/lock combination used by the site engineers to set up the TPN. This is the key that is used by the TPN when the engineer is in the Data Entity Builder, CL compiler, Picture Builder, and so forth. This is the key that is set when the keyboard lock is in the engineer position. This key also opens the operator and supervisor locks.
$D_program	The key/lock combination used by CG applications, CL programs, and so forth. This key also opens the engineer, supervisor, and operator locks. This key cannot be set at the keyboard.
$D_cont_control	The key/lock combination used by continuous control; that is, the point processors and CL in the Application Module. For instance, when a cascaded point in an AM is 'on control' and cascading to a secondary point this key would be used. This key also opens the operator, supervisor, engineer, and program locks. This key cannot be set at the keyboard.
$D_point_builder	The key/lock combination used by the Data Entity Builder when it is setting up the basic structure/storage for points it is building in the TPN database. Because of this, some of these parameters should not be stored at runtime. This key also opens the operator, supervisor, engineer, program, and continuous control locks. This key cannot be set at the keyboard.
$D_intimate_user	The super 'pass' key / the weakest lock. This key opens all locks and can store all parameters that can be stored. Of course, it cannot store calculated values like CPUFREE. Some parameters that can be stored using this key should not be stored at runtime. This key cannot be set at the keyboard.

7.5.3 TPN Server error returned

The TPN Server has a list of specific OPC ppError codes that can be returned for each individual item. These ppError codes are TPN System error status and are shown in “Table 16: Mapping of TPN Value Status to OPC Quality”. For OPC Sync Read method, the TPN Server returns the TPN System error status and an OPC Quality (wQuality), which is mapped from TPN System error status as shown in Appendix B. For other OPC methods that do not return an OPC Quality the TPN Server will only return the ppError codes

7.6 OPC security

The OPC Security Custom Interface Specification defines two optional custom interfaces that servers may implement to give client applications more flexibility for access security. Since the TPN Server uses Windows user ids and passwords as a basis for security, it implements IOPCSecurityNT.

The core of the OPC NT-based security is the IOPCSecurityNT::ChangeUser() call. When a client makes that call to the TPN Server, the COM identity used for the call is captured, and used as the basis for all succeeding calls, until another ChangeUser call is made, or the connection is broken. So the primary job of the client is to correctly establish the COM identity for the ChangeUser call, based on whatever mechanism the client chooses.

COM implements the CoSetProxyBlanket call which supports two mechanisms for setting the call identity for a COM call. One way is to supply textual credentials (domain, user, and password) to the CoSetProxyBlanket call. The other requires the client to impersonate the new user, and then make the CoSetProxyBlanket call with the **cloaking** flag set correctly. There are several issues the client developer must address when deciding which mechanism to use. The TPN Server always returns a value of RPC_C_IMP_LEVEL_IDENTIFY for QueryMinImpersonationLevel.

For considerations related to use of textual credentials versus impersonation, refer to the *HCI Client Developer Guide.*

7.7 IOPCBrowseServerAddressSpace

TPN Servers support the OPC data access interface, IOPCBrowseServerAddressSpace. The organization of the TPN is reported to the client as hierarchical. TPN unit names are listed as branches of the root, point names within each unit are listed as branches of the unit they reside in, and the parameters of each point are listed as leaves of the point. Since all the points are unique across the TPN and can be accessed directly without knowing the unit, the organization of the TPN could be considered flat. However, since points can be assigned to a unit, the organization is reported as hierarchical.

7.8 IOPCItemProperties interfaces

The *OPC Data Access Specification* includes three methods regarding *item properties*. An *item* in this context is a TPN point (for example, A100). The *properties* of the TPN point include its current value, the data type of the current value, alarm limits, descriptions, and point type information (as in analog or discrete). In other words, just what kind of a thing *is* the item in question? The item can be specified in the point-dot-parameter format or with the point name alone. In either case, only the point name is used to find the item's properties.

These functions use a base file and a user-editable text file to define the properties of interest. The base file will be preloaded with a default list of properties based on the list of recommended properties defined in the *OPC Data Access Specification* (in the section that describes IOPCItemProperties). A user can use any text editor to add, delete, or modify properties by editing the user file. The TPN Server will use these files to create a read request for the TPN. The list contains a mix of TPN parameters that can apply to only analog data or only to discrete data. The TPN Server requests all the parameters in the list for the point name specified. Any parameters that errors out are assumed to be non-applicable for the point in question. Successful parameter fetches will be used to form the answer for the GetItemProperties method.

“Table 13: Cross Reference of OPC Item Properties to TPN Parameters”that follows provides a cross-reference of OPC item properties to TPN parameters.

Table 13: Cross Reference of OPC Item Properties to TPN Parameters

ID	Data type of returned variant	Standard description	TPN parameter	Notes
1	VT_I2	'Item Canonical DataType' (VARTYPE stored in an I2)	N/A	Deduced by Generic HCI OPC Server (GOPC) from ID 2
2	<varies>	<Item Value> (VARIANT) Note the type of value returned is as indicated by the <Item Canonical DataType> above and depends on the item. This will behave like a read from DEVICE.	PV OP	Process Variable actually read from the TPN. For some points on TPN, the *item value* concept does not make sense or cannot be determined. Examples include Processor Status Data Point (PSDP), AM custom point, or Array point, where no item value is reported. Other points do not have a PV but do have an OP (such as Analog Output points). For these points, the OP value is used as the item value.
3	VT_I2	'Item Quality' (OPCQUALITY stored in an I2) Behaves like a read from DEVICE.	N/A	Deduced by Generic HCI OPC Server (GOPC) from ID 2
4	VT_DATE	'Item Timestamp' (will be converted from FILETIME) Behaves like a read from DEVICE.	N/A	Supplied by GOPC
5	VT_I4	'Item Access Rights' (OPCACCESSRIGHTS stored in an I4)	N/A	Supplied by GOPC

ID	Data type of returned variant	Standard description	TPN parameter	Notes
6	VT_R4	'Server Scan Rate' (in msec) Represents fastest (best case) rate at which the server can obtain data from the underlying data source. Although undefined, the nature of this source is typically a Device Network, DCS system, SCADA system, PLC via a COMM port or network, and so on. This value generally represents the best case RequestedUpdateRate which could be used if this item were added to an OPCGroup. Accuracy of this value (server's ability to attain ‘best case' performance) can be greatly affected by system load and other factors.	N/A	Supplied by GOPC
7-99		Reserved for future OPC use	N/A	

ID Set 2 - Recommended Properties—This is additional information that is commonly associated with ITEMs. This includes additional ranges of values that are reserved for use by other future OPC Specifications. For information about the newest field ID assignments, consult the other OPC Foundation Specifications.

The position of the OPC Foundation is this: If you have properties associated with an item which seem to fit the descriptions in “Table 14: Server's Subset of Values” that follows, then it is recommended that you use these specific descriptions and ID codes to expose those properties through this interface.

Note: The OPC Foundation reserves the right to expand this list from time to time.

Table 14: Server's Subset of Values

ID	Data type of returned variant	Standard description	TPN parameter or alternate	Notes
		Properties related to the Item Value.		
100	VT_BSTR	'EU Units' Example, 'DEGC' or 'GALLONS'	EUDESC	8 characters
101	VT_BSTR	'Item Description' Example: 'Evaporator 6 Coolant Temp'	PTDESC	24 characters
102	VT_R8	'High EU' Present only for analog data. This represents the highest value likely to be obtained in normal operation. Used for automatically scaling a bar graph display and so on. Example: 1400.0	PVEUHI	Real, already supplied when an ‘analog' PV is fetched through TPNSERVER

ID	Data type of returned variant	Standard description	TPN parameter or alternate	Notes
103	VT_R8	'Low EU' Present only for ‘analog' data. This represents the lowest value likely to be obtained in normal operation. Used for automatically scaling a bar graph display and so on. Example: -200.0	PVEULO	Real, already supplied when an ‘analog' PV is fetched through TPNSERVER
104	VT_R8	'High Instrument Range' Present only for ‘analog' data. This represents the highest value that can be returned by the instrument. Example: 9999.9	PVEXEUHI	Real, PV ‘extended' high range.
105	VT_R8	'Low Instrument Range' Present only for ‘analog' data. This represents the lowest value that can be returned by the instrument. Example: -9999.9	PVEXEULO	Real, PV ‘extended' low range.
106	VT_BSTR	'Contact Close Label' Present only for discrete data. This represents a string to be associated with this contact when it is in the closed (non-zero) state. Example: 'RUN,' 'CLOSE,' 'ENABLE,' 'SAFE,' and so on.	STATE2	Enumerated value, for STATE2, 8 characters for STATETXT(1). TPN also allows two other states for ‘moving' (or ‘in between') and ‘bad'. TPN also may have a reversing configuration that swaps the 0 and 1 state.
107	VT_BSTR	'Contact Open Label' Present only for discrete data. This represents a string to be associated with this contact when it is in the open (zero) state Examples: 'STOP,' 'OPEN,' 'DISABLE,' 'UNSAFE' and so on.	STATE1	Enumerated value, for STATE1, 8 characters for STATETXT(0). TPN also allows two other states for ‘moving' (or ‘in between') and ‘bad'. TPN also may have a ‘reversing' configuration that swaps the 0 and 1 state.
108	VT_I4	'Item Timezone' The difference in minutes between the items UTC Timestamp and the local time in which the item value was obtained. See the OPCGroup TimeBias property. Also see the WIN32 TIME_ZONE_INFORMATION structure.	N/A	The TPN does not support the concept of a time zone. GOPC deduces time zone from the time zone in which TPNSERVER is running
109-199		Reserved for future OPC use. Additional IDs may be added without revising the interface ID.	N/A	
		Properties related operator displays		
200	VT_BSTR	'Default Display' The name of an operator display associated with this ItemID	N/A	The TPN allows the definition of a native window associated schematic (not a GUS picture).
201	VT_I4	'Current Foreground Color' The COLORREF in which the item should be displayed	N/A	

ID	Data type of returned variant	Standard description	TPN parameter or alternate	Notes
202	VT_I4	'Current Background Color' The COLORREF in which the item should be displayed	N/A	
203	VT_BOOL	'Current Blink' Should a display of this item blink?	N/A	
204	VT_BSTR	'BMP File' Example: *c:\media\fic101.bmp*	N/A	
205	VT_BSTR	'Sound File' Example: *c:\media\fic101.wav*, or *.mid*	N/A	
206	VT_BSTR	'HTML File' Example: *http:\\mypage.com/FIC101.HML*	N/A	
207	VT_BSTR	'AVI File' Example: *c:\media\fic101.avi*	N/A	
207-299		Reserved for future OPC use. Additional IDs may be added without revising the interface ID.	N/A	
		Properties Related to Alarm and Condition Values (preliminary)… IDs 300 to 399 are reserved for use by OPC Alarms and Events. See the OPC Alarm and Events specification for more information.		
300	VT_BSTR	'Condition Status' The current alarm or condition status associated with the Item Example: 'NORMAL', 'ACTIVE', 'HI ALARM', and so forth	HIGHAL	Enumerated value. Using the external form of the enumeration would return an 8-character string. The HIGHAL parameter on the TPN reports the highest alarm state the point is currently at (including no-alarm/normal) according to documented alarm hierarchy.
301	VT_BSTR	'Alarm Quick Help' A short text string providing a brief set of instructions for the operator to follow when this alarm occurs.	N/A	Not available on the TPN.
302	VT_BSTR |VT_ARRAY	'Alarm Area List' An array of strings indicating the plant or alarm areas, which include this ItemID.	N/A	The TPN does not allow an easy way to find which process areas contain the unit in question.
303	VT_BSTR	'Primary Alarm Area' A string indicating the primary plant or alarm area including this ItemID	UNITNAME	String 8 characters long, name of TPN process ‘unit' in which the point is a member.

ID	Data type of returned variant	Standard description	TPN parameter or alternate	Notes
304	VT_BSTR	'Condition Logic' An arbitrary string describing the test being performed. Example: 'High Limit Exceeded' or 'TAG.PV >= TAG.HILIM'	N/A	This is not available from the TPN.
305	VT_BSTR	'Limit Exceeded' For multistate alarms, the condition exceeded Example: HIHI, HI, LO, LOLO	HIGHAL	Enumerated value. The HIGHAL parameter on TPN reports highest alarm state the point is currently at (including no-alarm/ normal) according to a documented alarm hierarchy.
306	VT_R8	'Deadband'	PVALDBEU	The TPN has an enumerated value (PVALDB) that specifies several fixed dead bands (0.5, 1, 2, 3, 4, or 5% of range) or it indicates a custom dead band in EUs is contained in the PVALDBEU parameter. The enumerated value is not returned —only the EU parameter is returned if it exists.
307	VT_R8	'HiHi Limit'	PVHHTP	Real, PV high high trip point.
308	VT_R8	'Hi Limit'	PVHITP	Real, PV high trip point.
309	VT_R8	'Lo Limit'	PVLOTP	Real, PV low trip point.
310	VT_R8	'LoLo Limit'	PVLLTP	Real, PV low low trip point.
311	VT_R8	'Rate of Change Limit'	PVROCPTP	Real, PV rate-of-change trip point. TPN supplies two rate-of-change limits—one for PV going up and one for PV going down. The positive ROC limit is chosen here.
312	VT_R8	'Deviation Limit'	DEVHITP	Real, PV deviation trip point. TPN supplies two deviation limits—one for how much the PV is above the SP and the other for how far the PV is below the setpoint. The high deviation limit is chosen here.
313-399		Reserved for future OPC Alarms and Events use. More IDs may be added without revising interface ID.	N/A	
400-4999		Reserved for future OPC use. More IDs may be added without revising interface ID.	N/A	

7.9 Property Definition Files

The property definition files (PDF) are plain text Windows files that can be edited by any primitive editor such as Notepad. The PDF has a defined syntax and order so that user can easily maintain it. The base file is preloaded with parameters based on the OPC suggested list. The file has a comment notation so that the file can contain self-documentation comments. Comment lines begin with a comment character (the slash /) and are ignored by the underlying software. A version line will begin with a version character (the pound #) and is used by the underlying software to distinguish between any differing (PDF) formats that may require translation. Two files are necessary so that changes to the base file made by Honeywell do not destroy customizations made by the customers on installation of newer versions. The files reside in the *<drive>: ProgramData\Honeywell \TPNServer* directory structure as BASEPD.TXT and CUSTOMPD.TXT.

There are two special cases in the area of alarm limits. The OPC specification only allows one value for the deviation limit and one value for the rate-of-change limit. The TPN defines two limits for each of these conditions. The TPN has both a high and low deviation limit that alarm when the PV deviates above or below the set point. Furthermore, the TPN defines two rate-of-change limits. One ROC alarms when the PV is changing in a positive direction. The other ROC alarms when the PV is changing in the negative direction. To address this, the base PDF arbitrarily contains only the high deviation limit and the positive ROC limit. It is expected that in the usual case these pairs of limits will be equal to their respective low limits. In the cases where they are not, the customers who are interested can add the missing alarm limit(s) to their custom PDF file.

The base file is read and acted on first. The effect of this is that if an item in the custom file has the same item ID as the base file, the custom file item will override the base file item. For example, if a customer wanted to choose the low deviation limit instead of the high deviation limit as the default deviation limit (item 312), the custom file could contain an item 312 specifying DEVLOTP. The effect of this would be that the software would read the base file, get the DEVHITP, place the value in memory, read the custom file, get the DEVLOTP, and overwrite the value of DEVHITP in memory. If one were interested in both the DEVHITP and the DEVLOTP, one could put an entry into the custom file with an item number greater than 5000 (that is, NOT 312). The software would then return both values.

The property definition files are read during TPN Server startup. Therefore, in order for PDF changes to take effect, the TPN Server must be stopped and restarted.

7.9.1 Example of the base property definition file

```
//
// This file contains the base property list for TPN points.
// The format of the file is as follows:
//
// Field one is the property ID number(0-5000+)
// Field two is the TPN parameter associated with that property
// Field three is a text description of the property
//
// The fields are tab separated.
//
// For points that have the parameter PV, it will be used as the ‘value'.
// For points that do not have PV but do have an OP (Example: AO points) OP will
// be the ‘value'. Otherwise, OP will not be used.
// For points where ‘value' does not make sense (Example: PSDP) no value
// will be reported as the ‘value'.
//
#version 1.0
```

2	PV	‘Item Value'
2	OP	‘Item Value'
100	EUDESC	‘EU Units'
101	PTDESC	‘Item Description'
102	PVEUHI	‘High EU'

103	PVEULO	‘Low EU'
104	PVEXEUHI	‘High Instrument Range'
105	PVEXEULO	‘Low Instrument Range'
106	STATE2	‘Contact Close Label'
107	STATE1	‘Contact Open Label'
300	HIGHAL	‘Condition Status'
303	UNITNAME	‘Primary Alarm Area'
305	HIGHAL	‘Limit Exceeded'
306	PVALDBEU	‘Deadband'
307	PVHHTP	‘High High Limit'
308	PVHITP	‘High Limit'
309	PVLOTP	‘Low Limit'
310	PVLLTP	‘Low Low Limit'
311	PVROCPTP	‘Rate of Change Limit'
312	DEVHITP	‘Deviation Limit'

7.10 Public Methods summary

There are three public methods on the IOPCItemProperties interface—GetItemProperties, QueryAvailableProperties, and LookUpItemIDs.

7.10.1 IOPCItemProperties::GetItemProperties

Scenario: You invoke Get Item Properties method, passing the point in question, the dwCount, and the pdwPropertyIDs.

For each successful property read, the related ppvData and ppErrors element is set. For successful values, the ppvData(n) element is set to the value and data type of the value returned from the TPN, and the ppErrors(n) is set to S_OK. For unsuccessful values, the ppvData(n) is set to VT_EMPTY, and ppErrors(n) is set to OPC_E_INVALID_PID or to a specific device specific error, as appropriate.

7.10.2 IOPCItemProperties::QueryAvailableProperties

Scenario: You invoke Query Available Properties method, passing the point in question. The TPN Server uses the Property Definition File (PDF) for a list of properties to query.

When the results are returned, the results are used to form the method's return values. The pdwCount is set to the number of successful parameter accesses that were returned out of PDF list. The ppPropertyIDs are set by cross- referencing the successful parameters to their respective Ids in the PDF. Cross-referencing the successful parameters to their respective descriptions in the PDF sets the ppDescriptions. The ppvtDataTypes are set from the data types returned as a result of the parameter access.

Attention

When invoking the QueryAvailableProperties method, the property IDs that are returned represent every property ID (LCN Parameter) available for a point's data structure—not just those of the currently user-configured data structure. As a result, non-user-configured property IDs will display invalid values.

7.10.3 IOPCItemProperties::LookUpItemIDs

Scenario: You invoke Look Up Item IDS method, passing the point in question and dwCount.

A list of point.parameter values are formed and returned in ppszNewItemIDs(n), and the related ppErrors(n) is set to S_OK. If the element cannot be found in the PDF, the ppszNewItemIds(n) for that element is set to the null string and the ppErrors(n) for that element is set to an appropriate non-OK HRESULT error.

7.11 TPN Server Specific Interfaces

The ITPNECTrigger interface is used to cause a command execution ('EC') file to be scheduled to run on the TPN.

Public Methods summary-TPN Server Specific Interfaces

There is one Public Method on the ITPNECTrigger interface called 'RunECFile.'

7.11.1 ITPNECTrigger Interface

```
STDMETHODIMP RunECFile(
LPCWSTR TPNECpath,
DATE TPNECtime);
```

7.11.2 Operational description

The file must already have been prepared, checked, and saved to the TPN History Module; this interface only schedules the file to be executed. How the EC file is prepared is outside the scope of this function. Further, any output resulting from executing the EC file is not returned from this interface. The EC file is not checked for content or if it even exists. A successful return from this routine only indicates that the EC has been queued to run on the TPN, it does not indicate that the EC has run, will run, or that the EC will be successful when and if it does run.

To determine if the EC actually runs, it is expected that the EC itself will create a results file on the TPN History Module (for example, include a data out (also known as ‘DO path') as one of its statements). The fact that this results file exists and can be opened can indicate that the EC has run. The contents of the result file should indicate the status of the tasks performed by the EC file. However this left up to you. How the results file, if it even exists, is examined is also outside the scope of this function.

The time indicated will be converted to a TPN time. Since the TPN has no time zone specification, during conversion it will be assumed that the TPN is in the same time zone as the TPN Server. Further it is assumed that the time zone can be retrieved. Times that convert to the past will be interpreted as meaning ‘now' or ‘as soon as possible'. This function can only fully function on a GUS; Other TPN nodes will return an error. The function depends on having a US-type personality running on the TPN side of the node.

7.11.3 Parameter specifications

Parameters	Direction	Description
TPNECpath	Input	A string describing the path to the TPN EC that is to be scheduled to run. Only ‘net' path form is allowed. The path length is limited to 28 characters do to TPN file system limits.
TPNECtime	Input	The UTC time when the above EC is to be scheduled to run. This time will ultimately be converted to a TPN time assuming that the server is in the same time zone as the TPN. A time/date in the past will be interpreted as ‘now' or ‘as soon as possible'.

7.11.4 HRESULT return codes

Return code	Description
S_OK	The function was successful.
E_INVALIDARG	An invalid argument is being used.

Return code	Description
E_FAIL	The function was unsuccessful.
E_NOTIMPL	The server does not support EC Trigger

7.11.5 Sample code-TPN Server-specific interface

The following sample code would be in a TPN Server client for usage of the RunECFile method on the ITPNECTrigger interface:

```
HRESULT RunEcFile(LPCWSTR ECPath, DATE TimeToRun, IUnknown *pUnk)
{
int ref = 0;
HRESULT
hr = E_FAIL;
ITPNECTrigger *pW1 = NULL;
hr = pUnk->QueryInterface(IID_ ITPNECTrigger, (void**)&pW1);
if (SUCCEEDED(hr))
{
hr = pW1-> RunECFile (ECPath, TimeToRun);
ref = pW1->Release();
}
else
{
MessageBox(0,_T('TPN Server ITPNECTrigger Interface Not Implemented\n') +
HResultToString(hr),0,MB_OK | MB_SETFOREGROUND);
}
return hr;
}
```

7.11.6 TPN Server IDL

The TPN Server will have an IDL file to specify the ITPNECTrigger interface that is provided by the TPN Server. The contents of the TPN Server IDL file as follows:

```
// TPNDSSIDL.IDL
import 'oaidl.idl';
// Note: A device Specific Server Author should create an idl
// project and idl file similar to this to define the DSS
// custom interfaces. The definition of such interfaces is out
// side the scope of Hci.
//
// The DSS author is required to update the DSS self registration
// or reg file to reflect interface uuid defined in this file.
//------------------------------------------------------------------------
[
object,
uuid(170388FA-B1B6-11d0-8A02-00C04FC97D9D),
pointer_default(unique)
]
interface ITPNECTrigger: IUnknown
{
HRESULT RunECFile ([in] LPCWSTR ECPath, [in] DATE TimeToRun);
}
```

7.12 Mapping floating-point exceptions

"Table 15: Floating-Point Values Mapped to Windows Indications" maps the special case, floating-point values to indications on the Microsoft Windows side. There is a direct correlation between the values on the Windows-side and the TPN System-side.

Table 15: Floating-Point Values Mapped to Windows Indications

Floating-point exception	Windows indication
QNAN	1.#QNAN
0	0
Negative 0	0
Positive Infinity	1.#INF
Negative Infinity	-1.#INF

The TPN Server will return an OPC BAD Quality status (OPC_BAD_NON_SPECIFIC =0x00), with a TPN Server specific ppError of S_OK if the value of the real (VT_R4 or VT_R8) point.parameter read is a NAN (either signaling or quiet varieties), positive infinity, or negative infinity. If any or all elements of an array of real values are NAN (signaling or quiet), or +/- infinity for a whole array fetch, these statuses will also be returned. Users must scan the array and examine each element for NAN if they are interested in which element or elements of an array fetched as a whole are NAN.

7.13 TPN value status to OPC error and quality mapping

The TPN Server attempts to validate the OPC items (such as the item syntax or types) before making requests to the TPN network. Appropriate OPC defined errors are returned if the validation fails. If the validation succeeds, then the TPN Server makes TPN network requests that in-turn may return errors status. The TPN Server then maps these error codes to OPC item errors (ppError) and OPC quality flags.

In “Table 16: Mapping of TPN Value Status to OPC Quality” that follows, the first column provides the textual description of the TPN network error. The second column contains the TPN network error status code. The third column lists two ppError codes, the first is the OPC compliant ppError as returned to the client due to an OPC AddItems operation, and the second is the OPC compliant ppError as returned to the client due to OPC Read or Write operations. Lastly, the fourth column contains the OPC quality associated with the item.

The OPC defined Quality values that are returned by the TPN Server are:

- WORD OPC_GOOD_QUALITY = 0xc0;
- WORD OPC_BAD_NON_SPECIFIC =0x00;
- WORD OPC_BAD_CONFIG_ERROR = 0x04;
- WORD OPC_BAD_NOT_CONNECTED =0x08;
- WORD OPC_BAD_DEVICE_FAILURE = 0x0c;
- WORD OPC_BAD_SENSOR_FAILURE = 0x10;
- WORD OPC_BAD_LAST_KNOWN = 0x14;
- WORD OPC_BAD_COMM_FAILURE = 0x18;
- WORD OPC_BAD_OUT_OF_SERVICE = 0x1c;
- WORD OPC_UNCERTAIN_NON_SPECIFIC = 0x40;
- WORD OPC_UNCERTAIN_NOT_ACCURATE = 0x50;
- WORD OPC_UNCERTAIN_EU_EXCEEDED = 0x54;
- WORD OPC_UNCERTAIN_SUB_NORMAL = 0x58;

In “Table 16: Mapping of TPN Value Status to OPC Quality”, the * in the TPN Server specific error codes (OPC_E_* or OPC_S_*) map to the string shown in the TPN System Error Status Description field. Note that the TPN specific constants that define these errors and OPC quality are defined in tpnsrvmc.h file that is located under *InstallDrive:\Program Files\Honeywell\Tps\TPNServer*.

Attention

With Experion R410, 64-bit OS is supported. All 32-bit components are installed in 'C:\Program Files (x86)'. The tpnsrvmc.h file is located in '*InstallDrive*:\Program Files(x86)\Honeywell\Tps\TPNServer'

Table 16: Mapping of TPN Value Status to OPC Quality

TPN System error status description	TPN error	Hresults AddItems ppError Read/Write ppError	OPC quality
LCN_WS_Everything_OK	0x0000	S_OK S_OK	OPC_GOOD_QUALITY
LCN_WS_Invalid_Point_Name	0x0001	OPC_E_UNKNOWNITEMID OPC_E_UNKNOWNITEMID	OPC_BAD_CONFIG_ERROR
LCN_WS_Param_Name_Cnvt_Error	0x0002	OPC_E_BADTYPE OPC_E_INVALIDITEMID	OPC_BAD_CONFIG_ERROR

TPN System error status description	TPN error	Hresults AddItems ppError Read/Write ppError	OPC quality
LCN_WS_Invalid_Parameter_Name	0x0003	OPC_E_BADTYPE OPC_E_INVALIDITEMID	OPC_BAD_CONFIG_ERROR
LCN_WS_Param_ID_Cnvt_Error	0x0004	OPC_E_BADTYPE OPC_E_INVALIDITEMID	OPC_BAD_CONFIG_ERROR
LCN_WS_Blind_Record_Size_Too_ Large_for_fetch	0x0005	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_WS_Blind_Record_Size_Too_ Large_for_store	0x0006	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_WS_Unsupported_Element_ Type_on_Fetch	0x0007	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_WS_Unsupported_Array_Type_ on_Fetch	0x0008	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_WS_Undefined_Param_Qualifier_ on_Fetch	0x0009	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_WS_Std_Enum_Cnvt_Elem_ ID_Error	0x000a	OPC_E_BADTYPE	OPC_BAD_CONFIG_ERROR
LCN_WS_Std_Enum_Cnvt_Array_ ID_Error	0x000b	OPC_E_BADTYPE	OPC_BAD_CONFIG_ERROR
LCN_WS_Real_Param_ID_Error_ For_Element	0x000c	OPC_E_BADTYPE OPC_E_RANGE	OPC_BAD_CONFIG_ERROR
LCN_WS_Real_Param_ID_Error_ For_Array	0x000d	OPC_E_BADTYPE OPC_E_RANGE	OPC_BAD_CONFIG_ERROR
LCN_WS_Parameter_Collect_Cmd_ Not_Supported	0x000e	OPC_E_BADTYPE	OPC_BAD_CONFIG_ERROR
LCN_WS_Invalid_Message_Type	0x000f	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_WS_Entity_ID_Cnvt_Error	0x0010	OPC_E_UNKNOWNITEMID OPC_E_UNKNOWNITEMID	OPC_BAD_CONFIG_ERROR
LCN_WS_Enum_Member_Cnvt_ For_Element_Error	0x0011	OPC_E_UNKNOWNITEMID OPC_E_INVALIDITEMID	OPC_BAD_CONFIG_ERROR
LCN_WS_Enum_Member_Cnvt_ For_Array_Error	0x0012	OPC_E_UNKNOWNITEMID OPC_E_INVALIDITEMID	OPC_BAD_CONFIG_ERROR
LCN_WS_Unsupported_Data_Type_ For_Store	0x0013	E_FAIL (N/A) OPC_E_BADTYPE	OPC_BAD_CONFIG_ERROR
LCN_WS_String_Size_Out_of_Range_ for_Fetch	0x0014	E_FAIL (N/A)	OPC_BAD_CONFIG_ERROR

TPN System error status description	TPN error	Hresults AddItems ppError Read/Write ppError	OPC quality
LCN_WS_Blind_Record_Size_Out_of_ Range_for_Fetch	0x0015	E_FAIL (N/A)	OPC_BAD_CONFIG_ERROR
LCN_WS_Integer_Array_Size_Out_of_ Range_for_Fetch	0x0016	E_FAIL (N/A)	OPC_BAD_CONFIG_ERROR
LCN_WS_Real_Array_Size_Out_of_ Range_for_Fetch	0x0017	E_FAIL (N/A)	OPC_BAD_CONFIG_ERROR
LCN_WS_Enum_Array_Size_Out_of_ Range_for_Fetch	0x0018	E_FAIL (N/A)	OPC_BAD_CONFIG_ERROR
LCN_WS_Boolean_Array_Size_Out_of_ Range_for_Fetch	0x0019	E_FAIL (N/A)	OPC_BAD_CONFIG_ERROR
LCN_WS_Time_Array_Size_Out_of_ Range_for_Fetch	0x001a	E_FAIL (N/A)	OPC_BAD_CONFIG_ERROR
LCN_WS_Entity_Array_Size_Out_of_ Range_for_Fetch	0x001b	E_FAIL (N/A)	OPC_BAD_CONFIG_ERROR
LCN_WS_Parameter_Array_Size_Out_ of_Range_for_Fetch	0x001c	E_FAIL (N/A)	OPC_BAD_CONFIG_ERROR
LCN_WS_Standard_Enum_Array_Size_ for_Store_Error	0x001d	E_FAIL (N/A)	OPC_BAD_CONFIG_ERROR
LCN_WS_SD_Enum_Array_Size_for_ Store_Error	0x001e	E_FAIL (N/A)	OPC_BAD_CONFIG_ERROR
LCN_WS_Integer_Array_Size_for_ Store_Error	0x001f	E_FAIL (N/A)	OPC_BAD_CONFIG_ERROR
LCN_WS_Real_Array_Size_for_Store_ Error	0x0020	E_FAIL (N/A)	OPC_BAD_CONFIG_ERROR
LCN_WS_Boolean_Array_Size_for_ Store_Error	0x0021	E_FAIL (N/A)	OPC_BAD_CONFIG_ERROR
LCN_WS_Time_Array_Size_for_Store_ Error	0x0022	E_FAIL (N/A)	OPC_BAD_CONFIG_ERROR
LCN_WS_Entity_Array_Size_for_Store_ Error	0x0023	E_FAIL (N/A)	OPC_BAD_CONFIG_ERROR
LCN_WS_Parameter_Array_Size_for_ Store_Error	0x0024	E_FAIL (N/A)	OPC_BAD_CONFIG_ERROR
LCN_WS_Real_Param_ID_Array_Size_ for_Store_Error	0x0025	E_FAIL (N/A)	OPC_BAD_CONFIG_ERROR
LCN_WS_String_Size_Too_Large_for_ store	0x0026	E_FAIL (N/A)	OPC_BAD_CONFIG_ERROR
LCN_WS_add_IDB_items_unmatch_ pattern	0x0027	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_WS_collect_IDB_unmatch_pattern	0x0028	E_FAIL	OPC_BAD_CONFIG_ERROR

TPN System error status description	TPN error	Hresults AddItems ppError Read/Write ppError	OPC quality
LCN_WS_store_IDB_unmatch_pattern	0x0029	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_WS_get_IDB_item_unmatch_ pattern	0x002a	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_WS_delete_IDB_unmatch_pattern	0x002b	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_WS_Exceeds_Maximum_IDB_ Items_on_Store	0x002c	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_WS_Exceeds_Maximum_IDB_ Items_on_Collect	0x002d	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_WS_Number_of_IDB_Items_Out_ of_Range	0x002e	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_WS_Boolean_Value_Out_of_ Range_for_Store_Error	0x002f	E_FAIL (N/A)	OPC_BAD_CONFIG_ERROR
LCN_WS_Invalid_Function_Level_Error	0x0030	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_WS_Invalid_IDB_Item_Number	0x0031	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_WS_Memory_Resource_Error	0x0032	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_WS_Unsupported_Array_Type_ on_Parameter_Fetch	0x0033	E_FAIL (/N/A)	OPC_BAD_CONFIG_ERROR
LCN_WS_Arrays_Are_Not_Supported_ on_Store_for_IDB	0x0034	E_FAIL (/N/A)	OPC_BAD_CONFIG_ERROR
LCN_WS_Array_Are_Not_Supported_ on_Fetch_for_IDB	0x0035	E_FAIL (/N/A)	OPC_BAD_CONFIG_ERROR
LCN_WS_Boolean_Array_Value_Out_ of_Range_for_Store_Error	0x0036	E_FAIL (/N/A)	OPC_BAD_CONFIG_ERROR
LCN_WS_Index_Out_of_Range_Error	0x0037	OPC_E_BADTYPE OPC_E_INVALIDITEMID	OPC_BAD_CONFIG_ERROR
LCN_WS_Std_Enum_Set_Number_ Out_of_Range_for_Store_Error	0x0038	E_FAIL (/N/A)	OPC_BAD_CONFIG_ERROR
LCN_WS_Store_Access_Level_out_of_ Range_for_Single_Access	0x0039	E_FAIL (/N/A)	OPC_BAD_CONFIG_ERROR
LCN_WS_Store_Access_Level_out_of_ Range_for_IDB_Access	0x003a	E_FAIL (/N/A)	OPC_BAD_CONFIG_ERROR
LCN_WS_Array_Indexed_by_Enum_ Not_Supported_Error	0x003b	E_BADTYPE OPC_E_INVALIDITEMID	OPC_BAD_CONFIG_ERROR
LCN_WS_Missing_Request_Priority_ Level_Error	0x0200	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_WS_No_HIGH_DA_Servers_ Configured_Error	0x0201	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_WS_No_LOW_DA_Servers_ Configured_Error	0x0202	E_FAIL	OPC_BAD_CONFIG_ERROR

TPN System error status description	TPN error	Hresults AddItems ppError Read/Write ppError	OPC quality
LCN_WS_Unsupported_Request_ riority_Level_Error	0x0203	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_WS_Missing_Subsequent_ Request_Priority_Level_Error	0x0204	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_WS_Unsupported_Subsequent_ Request_Priority_Level_Error	0x0205	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_WS_Available_Channel_Error	0x0206	E_FAIL	OPC_BAD_COMM_FAILURE
LCN_D_nothing_really_wrong	0x1000	S_OK S_OK	OPC_GOOD_QUALITY
LCN_D_cs_index_out_of_range_error	0x1001	OPC_E_INVALIDITEMID OPC_E_INVALIDITEMID	OPC_BAD_CONFIG_ERROR
LCN_D_data_realm_error	0x1002	OPC_E_INVALIDITEMID OPC_E_INVALIDITEMID	OPC_BAD_CONFIG_ERROR
LCN_D_duplicate_entity_error	0x1003	OPC_E_UNKNOWNITEMI D OPC_E_UNKNOWNITEMI D	OPC_BAD_CONFIG_ERROR
LCN_D_entity_name_error	0x1004	OPC_E_ UNKNOWNITEMID OPC_E_UNKNOWNITEMI D	OPC_BAD_CONFIG_ERROR
LCN_D_entity_id_error	0x1005	OPC_E_INVALIDITEMID OPC_E_INVALIDITEMID	OPC_BAD_CONFIG_ERROR
LCN_D_entitytype_id_error	0x1006	OPC_E_UNKNOWNITEMI D OPC_E_UNKNOWNITEMI D	OPC_BAD_CONFIG_ERROR
LCN_D_func_set_error	0x1007	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_func_set_qual_error	0x1008	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_function_level_error	0x1009	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_IDB_busy_error	0x100a	E_FAIL	OPC_BAD_COMM_FAILURE
LCN_D_IDB_id_error	0x100b	E_FAIL	OPC_BAD_COMM_FAILURE
LCN_D_IDB_not_built_error	0x100c	E_FAIL	OPC_BAD_COMM_FAILURE
LCN_D_IDB_value_buffer_allocation_ consistency_error	0x100d	E_FAIL	OPC_BAD_COMM_FAILURE
LCN_D_insufficient_memory_error	0x100e	E_FAIL	OPC_BAD_COMM_FAILURE
LCN_D_item_no_error	0x100f	OPC_E_UNKNOWNITEMI D OPC_E_UNKNOWNITEMI D	OPC_GOOD_QUALITY
LCN_D_item_not_processed_error	0x1010	E_FAIL	OPC_BAD_CONFIG_ERROR

TPN System error status description	TPN error	Hresults AddItems ppError Read/Write ppError	OPC quality
LCN_D_job_id_error	0x1011	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_local_name_code_error	0x1012	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_local_routing_code_error	0x1013	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_logicalnode_id_error	0x1014	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_max_idb_exceeded_error	0x1015	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_must_access_as_cs_error	0x1016	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_NIL_ptr_error	0x1017	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_node_unavailable_error	0x1018	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_non_base_entity_id_error	0x1019	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_non_composite_entity_id_error	0x101a	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_non_custom_descriptor_error	0x101b	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_non_local_entity_type_error	0x101c	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_not_in_cs_error	0x101d	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_not_verifiable	0x101e	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_num_elements_error	0x101f	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_num_items_error	0x1020	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_num_stores_error	0x1021	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_OK	0x1022	S_OK S_OK	OPC_GOOD_QUALITY
LCN_D_OK_with_item_errors	0x1023	OPC_E_INVALIDITEMID OPC_E_INVALIDITEMID	OPC_BAD_CONFIG_ERROR
LCN_D_parameter_id_error	0x1024	OPC_E_INVALIDITEMID OPC_E_INVALIDITEMID	OPC_BAD_CONFIG_ERROR
LCN_D_parameter_qualifier_error	0x1025	OPC_E_INVALIDITEMID OPC_E_INVALIDITEMID	OPC_BAD_CONFIG_ERROR
LCN_D_point_already_built_error	0x1026	OPC_E_INVALIDITEMID OPC_E_INVALIDITEMID	OPC_BAD_CONFIG_ERROR
LCN_D_point_in_service_error	0x1027	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_request_id_error	0x1028	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_search_incomplete	0x1029	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_segment_id_error	0x102a	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_subscript_out_of_range_error	0x102b	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_unit_error	0x102c	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_user_IDB_id_error	0x102d	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_batch_name_error	0x102e	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_not_all_nodes_responded_error	0x102f	E_FAIL	OPC_BAD_COMM_FAILURE
LCN_D_job_queue_disabled_error	0x1030	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_cannot_delete_active_point_ error	0x1031	E_FAIL	OPC_BAD_CONFIG_ERROR

TPN System error status description	TPN error	Hresults AddItems ppError Read/Write ppError	OPC quality
LCN_D_cannot_rebuild_active_point_error	0x1032	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_point_not_data_owner_established_error	0x1033	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_BCHHisAm_points_cannot_be_built_externally_error	0x1034	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_base_segment_does_not_exist_error	0x1035	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_inhibit_status_error	0x1036	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_file_manager_error	0x1037	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_num_parameters_error	0x1038	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_node_type_error	0x1039	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_rev_no_mismatch_error	0x103a	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_illegal_parameter_type_for_CDS_error	0x103b	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_array_size_error	0x103c	OPC_E_INVALIDITEMID OPC_E_INVALIDITEMID	OPC_BAD_CONFIG_ERROR
LCN_D_enm_set_invalid_error	0x103d	OPC_E_INVALIDITEMID OPC_E_INVALIDITEMID	OPC_BAD_CONFIG_ERROR
LCN_D_no_response_error	0x103e	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_Data_Point_is_too_large_error	0x103f	OPC_E_INVALIDITEMID OPC_E_INVALIDITEMID	OPC_BAD_CONFIG_ERROR
LCN_D_Custom_descriptors_inconsistent_error	0x1040	OPC_E_INVALIDITEMID OPC_E_INVALIDITEMID	OPC_BAD_CONFIG_ERROR
LCN_D_Data_Base_Full_error	0x1041	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_not_enough_contiguous_memory_error	0x1042	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_missing_off_node_reference_error	0x1043	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_Point_State_Change_In_Progress_Retry_Deletion	0x1044	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_CDS_file_already_opened	0x1045	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_parameter_not_found_error	0x1046	OPC_E_BADTYPE OPC_E_INVALIDITEMID	OPC_BAD_CONFIG_ERROR
LCN_D_incompatible_IDB_form	0x1047	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_Point_Build_In_Progress_Error	0x1048	E_FAIL	OPC_BAD_OUT_OF_SERVICE
LCN_D_NG_security_error	0x1049	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_NG_version_revision_mismatch_error	0x104a	E_FAIL	OPC_BAD_CONFIG_ERROR

TPN System error status description	TPN error	Hresults AddItems ppError Read/Write ppError	OPC quality
LCN_D_Entity_Name_length_error	0x104b	OPC_E_INVALIDITEMID OPC_E_INVALIDITEMID	OPC_BAD_CONFIG_ERROR
LCN_D_spare13_minor_status_enm	0x104c	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare14_minor_status_enm	0x104d	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare15_minor_status_enm	0x104e	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare16_minor_status_enm	0x104f	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare17_minor_status_enm	0x1050	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare18_minor_status_enm	0x1051	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare19_minor_status_enm	0x1052	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare20_minor_status_enm	0x1053	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare21_minor_status_enm	0x1054	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare22_minor_status_enm	0x1055	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare23_minor_status_enm	0x1056	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare24_minor_status_enm	0x1057	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare25_minor_status_enm	0x1058	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_END_minor_status_enm	0x1059	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_NULL_pa_status_enm	0x2000	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_do_value_clamped_error	0x2001	E_FAIL (N/A) OPC_S_CLAMP	OPC_UNCERTAIN_EU_ EXCEEDED
LCN_D_pa_string_truncated_on_ store_error	0x2002	E_FAIL (N/A) HR_CREATE_RESULT (HR_SV_SUCCESS, ID_SUB_HCI_TPN_SERVER, ushTpnSysErr)	OPC_UNCERTAIN_EU_ EXCEEDED
LCN_D_do_initialization_warning	0x2003	E_FAIL (N/A) HR_CREATE_RESULT (HR_SV_SUCCESS, ID_SUB_HCI_TPN_SERVER, ushTpnSysErr)	OPC_UNCERTAIN_NON_SPECIFIC
LCN_D_do_end_of_history_data_file	0x2004	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_end_of_user_specified_ samples_or_time	0x2005	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_user_allocated_buffer_full	0x2006	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_no_ACP_connected	0x2007	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_checkpoint_partial_restore	0x2008	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_do_UCN_Num_Fast_Points_ Clamped	0x2009	E_FAIL (N/A) OPC_S_CLAMP	OPC_UNCERTAIN_EU_ EXCEEDED
LCN_D_array_string_truncated_on_ fetch	0x200a	E_FAIL (N/A) HR_CREATE_RESULT (HR_SV_SUCCESS, ID_SUB_HCI_TPN_SERVER, ushTpnSysErr)	OPC_UNCERTAIN_EU_ EXCEEDED

TPN System error status description	TPN error	Hresults AddItems ppError Read/Write ppError	OPC quality
LCN_D_do_data_overlaps_with_other_ array_point_data_warning	0x200b	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_pa_OK	0x200c	S_OK S_OK	OPC_GOOD_QUALITY
LCN_D_do_access_level_invalid	0x200d	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_algorithm_error	0x200e	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_algorithm_must_be_DDC	0x200f	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_algorithm_must_be_SPC	0x2010	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_bad_type_error	0x2011	OPC_E_BADTYPE OPC_E_BADTYPE	OPC_BAD_CONFIG_ERROR
LCN_D_do_bias_is_undergoing_ initialization	0x2012	E_FAIL (N/A) HR_CREATE_RESULT (HR_SV_SUCCESS, ID_SUB_HCI_TPN_SERVER, ushTpnSysErr)	OPC_UNCERTAIN_NOT_ ACCURATE
LCN_D_do_both_init_and_tracking_ cannot_be_configured	0x2013	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_box_failure_error	0x2014	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_box_invalid	0x2015	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_mode_must_be_manual	0x2016	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_cannot_convert_numeric_ variable_to_ieee	0x2017	OPC_E_BADTYPE OPC_E_BADTYPE	OPC_BAD_NON_SPECIFIC
LCN_D_do_cannot_convert_numeric_ variable_to_jfp	0x2018	OPC_E_BADTYPE OPC_E_BADTYPE	OPC_BAD_NON_SPECIFIC
LCN_D_do_cannot_reset_timer_unless_ stopped	0x2019	E_FAIL (N/A)	OPC_BAD_CONFIG_ERROR
LCN_D_do_cascade_request_flag_ must_be_set	0x201a	E_FAIL (N/A)	OPC_BAD_CONFIG_ERROR
LCN_D_do_change_not_permitted_ by_operator	0x201b	E_FAIL (N/A)	OPC_BAD_CONFIG_ERROR
LCN_D_do_cascade_enable_flag_ must_be_set	0x201c	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_configuration_mismatch_ error	0x201d	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_control_output_connection_ not_configured	0x201e	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_current_mode_disallows_ mode_change	0x201f	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_current_mode_disallows_ store	0x2020	E_FAIL	OPC_BAD_CONFIG_ERROR

TPN System error status description	TPN error	Hresults AddItems ppError Read/Write ppError	OPC quality
LCN_D_do_device_failure_error	0x2021	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_device_in_reset_error	0x2022	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_engr_unit_span_too_small	0x2023	E_FAIL (N/A) HR_CREATE_RESULT(HR_SV_SUCCESS, ID_SUB_HCI_TPN_SERVER, ushTpnSysErr)	OPC_UNCERTAIN_NOT_ ACCURATE
LCN_D_do_entity_subscript_not_ implemented_error	0x2024	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_external_mode_switching_ enabled	0x2025	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_external_switching_option_ not_selected	0x2026	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_fatal_error_no_match_ for_case_selector	0x2027	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_hiway_failure_error	0x2028	E_FAIL	OPC_BAD_COMM_FAILURE
LCN_D_do_hiway_access_failure_ error	0x2029	E_FAIL	OPC_BAD_COMM_FAILURE
LCN_D_do_illegal_value	0x202a	E_FAIL (N/A)	OPC_BAD_NON_SPECIFIC
LCN_D_do_init_cannot_be_configured	0x202b	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_init_must_be_configured	0x202c	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_initialization_error	0x202d	E_FAIL	OPC_BAD_OUT_OF_SERVICE
LCN_D_do_internal_variable_number_ invalid	0x202e	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_invalid_algo_id_in_base_ seg_error	0x202f	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_invalid_algorithm_equation	0x2030	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_invalid_control_algorithm_ id_error	0x2031	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_invalid_destination_parameter	0x2032	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_invalid_destination_point_id	0x2033	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_invalid_mode	0x2034	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_invalid_mode_attribute	0x2035	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_invalid_pboption	0x2036	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_invalid_point_for_point_build	0x2037	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_invalid_spoption	0x2038	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_invalid_target_value_ processor_state	0x2039	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_max_structural_parameter_ exceeded_error	0x203a	E_FAIL	OPC_BAD_CONFIG_ERROR

TPN System error status description	TPN error	Hresults AddItems ppError Read/Write ppError	OPC quality
LCN_D_do_mode_attribute_error	0x203b	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_invalid_pv_control_ algorithm_combination	0x203c	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_duplicate_batch_id	0x203d	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_mode_error	0x203e	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_mode_illegal_for_this_ point_type	0x203f	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_mode_keylock_error	0x2040	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_mode_not_allowed_ with_configured_algo	0x2041	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_mode_not_currently_legal	0x2042	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_mode_not_man_or_point_ active	0x2043	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_must_have_digital_output	0x2044	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_no_value_error	0x2045	E_FAIL	OPC_BAD_OUT_OF_SERVICE
LCN_D_do_nocopts_must_be_zero to_change_copctype	0x2046	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_normal_attribute_not_ configured	0x2047	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_normal_mode_is_not_ configured	0x2048	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_not_selected_error	0x2049	E_FAIL	OPC_BAD_NOT_CONNECTE D
LCN_D_do_num_outputs_must_ exceed_zero	0x204a	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_off_process_error	0x204b	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_output_is_undergoing_ initialization	0x204c	E_FAIL	OPC_BAD_OUT_OF_SERVICE
LCN_D_do_override_required_error	0x204d	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_parameter_can_not_be_ changed	0x204e	E_FAIL (N/A) OPC_E_BADRIGHTS	OPC_BAD_CONFIG_ERROR
LCN_D_do_parameter_index_invalid	0x204f	OPC_E_INVALIDITEMID OPC_E_INVALIDITEMID	OPC_BAD_CONFIG_ERROR
LCN_D_do_parameter_invalid	0x2050	OPC_E_INVALIDITEMID OPC_E_INVALIDITEMID	OPC_BAD_CONFIG_ERROR
LCN_D_do_parameter_is_undergoing_ initialization	0x2051	E_FAIL (N/A)	OPC_BAD_OUT_OF_SERVICE
LCN_D_do_parameter_not_configured	0x2052	OPC_E_INVALIDITEMID OPC_E_INVALIDITEMID	OPC_BAD_CONFIG_ERROR

TPN System error status description	TPN error	Hresults AddItems ppError Read/Write ppError	OPC quality
LCN_D_do_parameter_not_valid_for_ configured_control_algo	0x2053	OPC_E_INVALIDITEMID OPC_E_INVALIDITEMID	OPC_BAD_CONFIG_ERROR
LCN_D_do_parameter_not_valid_for_ configured_pv_algo	0x2054	OPC_E_INVALIDITEMID OPC_E_INVALIDITEMID	OPC_BAD_CONFIG_ERROR
LCN_D_do_partial_failure_error	0x2055	E_FAIL	OPC_BAD_COMM_FAILURE
LCN_D_do_point_must_be_inactive	0x2056	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_point_is_red_tagged	0x2057	E_FAIL	OPC_BAD_OUT_OF_SERVICE
LCN_D_do_point_is_undergoing_ initialization	0x2058	E_FAIL	OPC_BAD_OUT_OF_SERVICE
LCN_D_do_point_not_available_to_ point_build	0x2059	OPC_E_INVALIDITEMID OPC_E_INVALIDITEMID	OPC_BAD_OUT_OF_SERVICE
LCN_D_do_point_not_established_error	0x205a	OPC_E_UNKNOWNITEMID OPC_E_UNKNOWNITEMID	OPC_BAD_CONFIG_ERROR
LCN_D_do_point_not_secondary	0x205b	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_point_status_error	0x205c	E_FAIL	OPC_BAD_SENSOR_FAILURE
LCN_D_do_point_type_invalid	0x205d	OPC_E_UNKNOWNITEMID OPC_E_UNKNOWNITEMID	OPC_BAD_CONFIG_ERROR
LCN_D_do_pps_only_active_points_error	0x205e	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_process_box_restore_in_ operation	0x205f	E_FAIL	OPC_BAD_OUT_OF_SERVICE
LCN_D_do_process_module_off	0x2060	E_FAIL	OPC_BAD_OUT_OF_SERVICE
LCN_D_do_PV_source_disallows_PV_ change	0x2061	E_FAIL (N/A)	OPC_BAD_CONFIG_ERROR
LCN_D_do_ratio_is_undergoing_ initialization	0x2062	E_FAIL (N/A)	OPC_BAD_OUT_OF_SERVICE
LCN_D_do_read_only_parameter	0x2063	E_FAIL (N/A) OPC_E_BADRIGHTS	OPC_BAD_CONFIG_ERROR
LCN_D_do_red_tag_error	0x2064	E_FAIL	OPC_BAD_OUT_OF_SERVICE
LCN_D_do_secondary_has_too_ many_primaries	0x2065	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_secondary_must_be_ off_node	0x2066	E_FAIL	OPC_BAD_OUT_OF_SERVICE
LCN_D_do_secondary_point_has_ no_primary	0x2067	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_slot_invalid	0x2068	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_source_of_request_invalid	0x2069	E_FAIL	OPC_BAD_CONFIG_ERROR

TPN System error status description	TPN error	Hresults AddItems ppError Read/Write ppError	OPC quality
LCN_D_do_source_of_Y_is_not_ configured	0x206a	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_sp_is_undergoing _initialization	0x206b	E_FAIL	OPC_BAD_OUT_OF_SERVICE
LCN_D_do_sp_is_undergoing_ pvtracking	0x206c	E_FAIL	OPC_BAD_OUT_OF_SERVICE
LCN_D_do_store_not_allowed_with_ spc_connection	0x206d	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_store_not_permitted	0x206e	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_store_not_permitted_from_ off_point	0x206f	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_subscript_error	0x2070	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_system_error	0x2071	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_tracking_cannot_be_ configured	0x2072	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_tracking_error	0x2073	E_FAIL	OPC_BAD_OUT_OF_SERVICE
LCN_D_do_tracking_must_be_ configured	0x2074	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_duplicate	0x2075	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_Y_cannot_be_configured_ to_own_LSP	0x2076	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_pointtype_id_error	0x2077	OPC_E_UNKNOWNITEMI D OPC_E_UNKNOWNITEMI D	OPC_BAD_CONFIG_ERROR
LCN_D_do_segment_class_error	0x2078	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_invalid_control_algo_error	0x2079	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_invalid_pv_algo_error	0x207a	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_segment_size_exceeds_ max_seg_size_error	0x207b	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_memory_unavailable_error	0x207c	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_get_memory_error	0x207d	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_structural_parameter_ missing_error	0x207e	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_structural_parameter_ max_violated_error	0x207f	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_invalid_cc_rank	0x2080	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_store_not_permitted_ while_casreq	0x2081	E_FAIL	OPC_BAD_CONFIG_ERROR

TPN System error status description	TPN error	Hresults AddItems ppError Read/Write ppError	OPC quality
LCN_D_do_analog_output_used_ for_modulating_control	0x2082	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_y_must_be_configured_ to_own_lsp	0x2083	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_store_not_permitted_ unless_free_variable	0x2084	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_control_lock_error	0x2085	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_point_not_primary	0x2086	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_off_node_parameter_ access_error	0x2087	E_FAIL	OPC_BAD_SENSOR_FAILURE
LCN_D_do_memory_allocation_error	0x2088	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_unit_mismatch	0x2089	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_point_must_be_scheduled	0x208a	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_before_after_period_ incompatible	0x208b	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_integer_value_required	0x208c	E_FAIL (N/A)	OPC_BAD_CONFIG_ERROR
LCN_D_do_invalid_pv_equation	0x208d	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_invalid_ctl_equation	0x208e	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_invalid_number_of_ pv_inputs	0x208f	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_invalid_number_of_ ctl_inputs	0x2090	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_invalid_number_of_ ctl_outputs	0x2091	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_invalid_pv_algoid	0x2092	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_invalid_ctl_algoid	0x2093	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_invalid_pv_algorithm	0x2094	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_invalid_ctl_algorithm	0x2095	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_invalid_rboption	0x2096	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_pv_algoid_cannot_ be_null	0x2097	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_ctl_algoid_cannot_ be_null	0x2098	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_pvalgid_structural_ parameter_fetch_error	0x2099	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_ctlalgid_structural_ parameter_fetch_error	0x209a	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_rboption_structural_ parameter_fetch_error	0x209b	E_FAIL	OPC_BAD_DEVICE_FAILURE

TPN System error status description	TPN error	Hresults AddItems ppError Read/Write ppError	OPC quality
LCN_D_do_spoption_structural_ parameter_fetch_error	0x209c	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_pveqn_structural_ parameter_fetch_error	0x209d	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_ctleqn_structural_ parameter_fetch_error	0x209e	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_number_of_pv_inputs_ structural_parameter_fetch_error	0x209f	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_number_of_ctl_inputs_ structural_parameter_fetch_error	0x20a0	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_number_of_ctl_outputs_ structural_parameter_fetch_error	0x20a1	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_copctype_structural_ parameter_fetch_error	0x20a2	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_structural_parameter_ fetch_error	0x20a3	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_invalid_structural_ parameter	0x20a4	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_configuration_error	0x20a5	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_number_of_ctl_outputs _is_zero	0x20a6	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_fetch_not_permitted	0x20a7	E_FAIL	OPC_BAD_OUT_OF_SERVICE
LCN_D_do_change_restricted_to_ engineering_personality	0x20a8	E_FAIL	OPC_BAD_OUT_OF_SERVICE
LCN_D_do_change_restricted_to_on_ process_personality	0x20a9	E_FAIL	OPC_BAD_OUT_OF_SERVICE
LCN_D_do_access_level_error	0x20aa	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_attribute_id_error	0x20ab	E_INVALIDITEMID E_INVALIDITEMID	OPC_BAD_DEVICE_FAILURE
LCN_D_do_function_level_error	0x20ac	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_parameter_id_error	0x20ad	E_INVALIDITEMID E_INVALIDITEMID	OPC_BAD_DEVICE_FAILURE
LCN_D_do_parameter_qualifier_ type_error	0x20ae	E_INVALIDITEMID E_INVALIDITEMID	OPC_BAD_DEVICE_FAILURE
LCN_D_do_parameter_type_invalid	0x20af	E_BADTYPE	OPC_BAD_CONFIG_ERROR
LCN_D_do_prefetch_item_error	0x20b0	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_data_type_error	0x20b1	OPC_E_BADTYPE	OPC_BAD_CONFIG_ERROR
LCN_D_do_rev_20_disallows_ both_SOPL_and_trend_memory	0x20b2	E_FAIL	OPC_BAD_CONFIG_ERROR

TPN System error status description	TPN error	Hresults AddItems ppError Read/Write ppError	OPC quality
LCN_D_do_limit_or_range_exceeded	0x20b3	E_FAIL (N/A) OPC_E_RANGE	OPC_UNCERTAIN_EU_EXCEEDED
LCN_D_do_limit_or_range_crossover	0x20b4	E_FAIL (N/A) OPC_E_RANGE	OPC_UNCERTAIN_EU_EXCEEDED
LCN_D_do_inconsistent_lrc_for_ point_build	0x20b5	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_point_build_HG_only	0x20b6	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_must_configure_enough_ cl_slots_for_standard_actions	0x20b7	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_point_must_be_inactive_ or_ready	0x20b8	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_point_must_be_active	0x20b9	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_no_batch_slot_or_no_ slot_of_sufficient_size_available	0x20ba	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_number_of_concurrent_ batches_not_established	0x20bb	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_only_active_or_inactive_ allowed	0x20bc	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_prototype_must_have_ cls_to_be_cloned	0x20bd	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_only_one_dual_output_ store_permitted_per_request	0x20be	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_illegal_timer_state	0x20bf	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_illegal_counter_state	0x20c0	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_invalid_number_of_ainpts	0x20c1	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_invalid_number_of_ordstns	0x20c2	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_invalid_number_of_orvals	0x20c3	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_not_enough_contiguous_ memory	0x20c4	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_data_point_is_too_large_ error	0x20c5	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_point_is_local_manual	0x20c6	E_FAIL	OPC_BAD_OUT_OF_SERVICE
LCN_D_do_group_not_config_error	0x20c7	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_upc_iol_communications_ error	0x20c8	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_upc_iol_resource_error	0x20c9	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_upc_tvr_too_small	0x20ca	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_upc_calib_in_progress	0x20cb	E_FAIL	OPC_BAD_OUT_OF_SERVICE

TPN System error status description	TPN error	Hresults AddItems ppError Read/Write ppError	OPC quality
LCN_D_do_upc_command_not_ supported	0x20cc	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_upc_function_not_ implemented	0x20cd	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_upc_iol_pf_table_full	0x20ce	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_upc_iol_ps_table_full	0x20cf	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_upc_max_point_processing_ rate_exceeded	0x20d0	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_upc_no_pf_connections_ for_selected_slot	0x20d1	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_upc_only_one_connection_ allowed_to_iol	0x20d2	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_upc_pf_buffer_overflow	0x20d3	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_upc_ps_buffer_overflow	0x20d4	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_upc_ptexecst_or_modlstat_ inconsistent	0x20d5	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_upc_point_in_alarm	0x20d6	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_upc_ppx_task_not_started	0x20d7	E_FAIL	OPC_BAD_OUT_OF_SERVICE
LCN_D_do_alarm_configuration_ mismatch	0x20d8	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_control_state_read_error	0x20d9	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_control_state_basic_error	0x20da	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_control_state_test_error	0x20db	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_input_and_output_required	0x20dc	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_input_and_output_box_ mismatch	0x20dd	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_current_state_disallows_ operational_state_change	0x20de	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_configuration_forcing_in_ effect	0x20df	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_current_procmod_disallows_ store	0x20e0	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_device_must_be_in_idle	0x20e1	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_device_must_be_in_reset	0x20e2	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_illegal_box_protocol	0x20e3	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_invalid_box_status	0x20e4	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_invalid_characterization_type	0x20e5	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_invalid_database_index	0x20e6	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_parameter_not_loaded	0x20e7	E_FAIL	OPC_BAD_OUT_OF_SERVICE

TPN System error status description	TPN error	Hresults AddItems ppError Read/Write ppError	OPC quality
LCN_D_do_pvsource_invalid	0x20e8	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_event_overload	0x20e9	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_sp_lock_error	0x20ea	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_pa_access_level_error	0x20eb	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_pa_attribute_id_error	0x20ec	OPC_E_UNKNOWNITEMID OPC_E_UNKNOWNITEMID	OPC_BAD_CONFIG_ERROR
LCN_D_pa_entity_id_error	0x20ed	OPC_E_UNKNOWNITEMID OPC_E_UNKNOWNITEMID	OPC_BAD_CONFIG_ERROR
LCN_D_pa_function_level_error	0x20ee	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_pa_initialized	0x20ef	E_FAIL	OPC_BAD_LAST_KNOWN
LCN_D_pa_job_queue_disabled_error	0x20f0	E_FAIL	OPC_BAD_OUT_OF_SERVICE
LCN_D_pa_parameter_id_error	0x20f1	OPC_E_INVALIDITEMID OPC_E_INVALIDITEMID	OPC_BAD_CONFIG_ERROR
LCN_D_pa_parameter_qualifier_type_error	0x20f2	OPC_E_INVALIDITEMID OPC_E_INVALIDITEMID	OPC_BAD_CONFIG_ERROR
LCN_D_pa_parameter_type_invalid	0x20f3	OPC_E_BADTYPE	OPC_BAD_CONFIG_ERROR
LCN_D_pa_prefetch_item_error	0x20f4	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_pa_serial_no_error	0x20f5	OPC_E_UNKNOWNITEMID OPC_E_UNKNOWNITEMID	OPC_BAD_CONFIG_ERROR
LCN_D_pa_store_data_type_error	0x20f6	E_FAIL (N/A)	OPC_BAD_CONFIG_ERROR
LCN_D_pa_store_value_error	0x20f7	E_FAIL (N/A)	OPC_BAD_CONFIG_ERROR
LCN_D_pa_subscript_error	0x20f8	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_pa_unable_to_perform_limit_check_error	0x20f9	E_FAIL (N/A) HR_CREATE_RESULT(HR_SV_SUCCESS, ID_SUB_HCI_TPN_SERVER, ushTpnSysErr)	OPC_UNCERTAIN_SUB_NORMAL
LCN_D_pa_value_out_of_range_error	0x20fa	E_FAIL (N/A) OPC_E_RANGE	OPC_UNCERTAIN_EU_EXCEEDED
LCN_D_parameter_unavailable_error	0x20fb	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_pa_bad_data_returned	0x20fc	E_FAIL (N/A)	OPC_BAD_NON_SPECIFIC
LCN_D_pa_total_value_size_exceeded_error	0x20fd	E_FAIL	OPC_BAD_COMM_FAILURE
LCN_D_pa_type_value_envelope_incompatible_error	0x20fe	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_upc_ppx_task_not_terminated	0x20ff	E_FAIL	OPC_BAD_DEVICE_FAILURE

TPN System error status description	TPN error	Hresults AddItems ppError Read/Write ppError	OPC quality
LCN_D_do_upc_umscanrate_must_be_null	0x2100	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_upc_umscanrate_must_not_be_null	0x2101	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_upc_wm_invalid	0x2102	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_upc_wm_insuf_data	0x2103	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_upc_wm_invalid_st	0x2104	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_upc_scan_table_overflow	0x2105	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_upc_calculator_syntax_error	0x2106	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_upc_calculator_expression_too_big	0x2107	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_upc_permissive_error	0x2108	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_upc_interlock_error	0x2109	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_upc_hpn_comm_error	0x210a	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_upc_entity_id_error	0x210b	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_upc_slot_checkpoint_exceeds_avail_extension_memory	0x210c	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_upc_invalid_state_pa_status	0x210d	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_upc_invalid_for_secondary_pa_status	0x210e	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_upc_checksum_bad	0x210f	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_upc_invalid_sequence_execution_state	0x2110	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_upc_invalid_process_module_state	0x2111	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_upc_abnormal_handler_not_available	0x2112	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_upc_sequence_program_must_be_loaded	0x2113	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_upc_sequence_program_exceeds_configured_slot_size	0x2114	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_upc_no_confirmable_message_pending	0x2115	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_upc_fatal_sequence_code_errors	0x2116	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_upc_pf_communications_error	0x2117	E_FAIL	OPC_BAD_COMM_FAILURE
LCN_D_do_upc_hpn_encode_error	0x2118	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_upc_hpn_reception_error	0x2119	E_FAIL	OPC_BAD_DEVICE_FAILURE

TPN System error status description	TPN error	Hresults AddItems ppError Read/Write ppError	OPC quality
LCN_D_do_upc_hpn_transmit_error	0x211a	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_upc_com_error	0x211b	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_hpn_data_unavail_tgt_ node_sts_error	0x211c	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_point_is_shutdown	0x211d	E_FAIL	OPC_BAD_OUT_OF_SERVICE
LCN_D_do_cannot_leave_current_block	0x211e	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_cannot_leave_current_step	0x211f	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_cannot_leave_current_phase	0x2120	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_UPC_target_node_local_error	0x2121	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_illegal_LM_redundancy_state	0x2122	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_NG_string_scan_error	0x2123	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_NG_enum_scan_error	0x2124	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_NG_cannot_convert_blind_rec_ scan_error	0x2125	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_NG_enum_arr_scan_error	0x2126	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_illegal_off_network_ reference_error	0x2127	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_NG_sde_scan_error	0x2128	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_NG_sde_arr_scan_error	0x2129	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_NG_scan_members_req_error	0x212a	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_NG_security_no_access_error	0x212b	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_NG_security_store_access_error	0x212c	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_dump_in_progress	0x212d	E_FAIL	OPC_BAD_OUT_OF_SERVICE
LCN_D_NG_version_revision_mismatch	0x212e	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_error_cmd_not_valid_for_ on_process_personality	0x212f	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_error_cmd_not_valid_for_ simulation_personality	0x2130	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_iol_scan_max_limit_ exceeded	0x2131	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_iol_scan_cycle_limit_ exceeded	0x2132	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_state_must_be_pause	0x2133	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_state_must_be_run_ or_pause	0x2134	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_state_must_be_idle_ or_pause	0x2135	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_array_mapping_violation	0x2136	E_FAIL	OPC_BAD_CONFIG_ERROR

TPN System error status description	TPN error	Hresults AddItems ppError Read/Write ppError	OPC quality
LCN_D_do_delete_not_allowed_while_func_blocks_assigned	0x2137	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_control_output_write_not_permitted	0x2138	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_lc_address_alias_mismatch	0x2139	E_FAIL	OPC_BAD_CONFIG_ERROR
LCN_D_do_parameter_resource_file_contention	0x213a	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_parameter_resource_busy	0x213b	E_FAIL	OPC_BAD_DEVICE_FAILURE
LCN_D_do_writes_not_permitted	0x213c	E_FAIL (N/A)	OPC_BAD_CONFIG_ERROR
LCN_D_spare_error317	0x213d	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare_error318	0x213e	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare_error319	0x213f	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare_error320	0x2140	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare_error321	0x2141	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare_error322	0x2142	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare_error323	0x2143	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare_error324	0x2144	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare_error325	0x2145	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare_error326	0x2146	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare_error327	0x2147	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare_error328	0x2148	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare_error329	0x2149	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare_error330	0x214a	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare_error331	0x214b	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare_error332	0x214c	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare_error333	0x214d	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare_error334	0x214e	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare_error335	0x214f	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare_error336	0x2150	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare_error337	0x2151	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare_error338	0x2152	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare_error339	0x2153	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare_error340	0x2154	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare_error341	0x2155	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare_error342	0x2156	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare_error343	0x2157	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare_error344	0x2158	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare_error345	0x2159	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare_error346	0x215a	E_FAIL	OPC_BAD_NON_SPECIFIC

TPN System error status description	TPN error	Hresults AddItems ppError Read/Write ppError	OPC quality
LCN_D_spare_error347	0x215b	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare_error348	0x215c	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_spare_error349	0x215d	E_FAIL	OPC_BAD_NON_SPECIFIC
LCN_D_END_pa_status_enm	0x215e	E_FAIL	OPC_BAD_NON_SPECIFIC

8 Glossary

“Table 17: Abbreviations and Acronyms” and “Table 18: Miscellaneous Terminology” list the various Honeywell terms and corresponding definitions used in this guide. You should become familiar with them before using this guide.

9 Abbreviations and Acronyms

"Table 17: Abbreviations and Acronyms" lists the abbreviations and acronyms used in this guide.

Table 17: Abbreviations and Acronyms

Term	Definition
Experion APP	Experion Application Processing Platform
ACE-T	Application Control Environment TPS
ES-T	Experion Station TPS
ESVT	Experion Server TPS
LCN	Local Control Network
OLE	Object Linking and Embedding
OPC	OLE for Process Control
PV	Process Value
PVSTS	Process Value Status
PVSTSREC	An internal blind record that contains both the real PV and the PVSTS value in one parameter fetch
TPN	TotalPlant\ Network
TPS	**TotalPlant**\ Solution
UCN	Universal Control Network

10 Miscellaneous Terminology

"Table 18: Miscellaneous Terminology" lists the miscellaneous terminology used in this guide.

Table 18: Miscellaneous Terminology

Term	Definition
Application	A set of programmed or configured information designed to be executed by a specific set of rules and/or environments. Used in context, it could be taken to have any of a number of specific meanings, such as:
TPD Application	An application that uses TPD support and execution software
Windows Application	An application built for execution as a Windows executable (.exe)
TDC Application	An application built for execution on a Honeywell TDC System.
CL-INIT Application	An application built to run on an AMW and is initiated by a Honeywell CL (Control Language) Server. It is a window-less console application.
Generic	A term applied to applications and application packages that indicates that the application being discussed applies to any of a particular kind of equipment, equipment grouping, or product.
Specific	A term applied to applications ('specific application') that indicates that the application being discussed applies to a specific set of equipment or product.
Project	A development consisting of one or more applications.
Component	A piece or part of something, such as an application or hardware. In context, it can also mean: • **HCI Server Component**—An HCI Server which minimally supports OPC interfaces. An HCI Server component can be in-process or out-of-process. • **HCI Managed Component**—A TPS component, which runs in its own process and which supports the COM interfaces necessary to be managed by TPS System Management. Current HCI Managed Components include all out-of-process HCI Server Components and the CL/NT Application Server. HCI Managed Components are always out-of-process.