TruePort Linux User Guide

This document provides the procedure for installing and using TruePort on Linux.

Table of Contents

What is TruePort?

You use TruePort when you want to connect extra terminals to a server using a Device Server rather than a multi-port serial card; it is a tty device redirector. TruePort is especially useful when you want to improve data security, as you can create an SSL/TLS connection between the TruePort host port and the Device Server, which will encrypt the data between the two points.

TruePort Full Mode vs Lite Mode

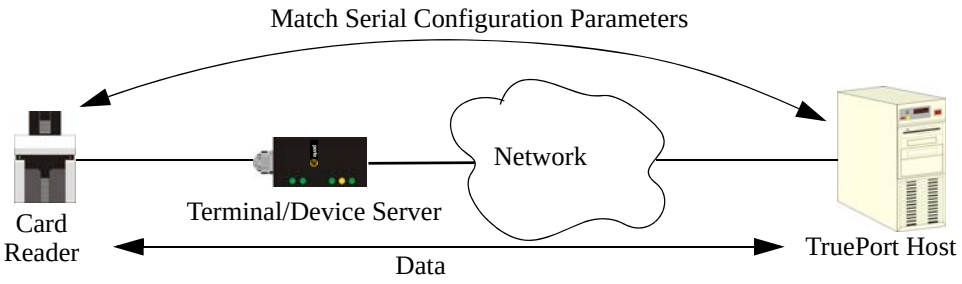
You can configure TruePort on Linux in either Full Mode or Lite Mode. When you start TruePort in Full Mode, the serial configuration parameters are set on the TruePort host. When you start TruePort in Lite Mode, the serial configuration parameters are set on the device/terminal server. On Linux, serial configuration parameters consist of bits per second (baud rate speed), data bits, parity, stop bits, flow control, and any other standard stty I/O parameters. In either mode, the data is passed in raw format, although you can enable the SSL/TLS connection option to encrypt the data going through a port.

Full Mode

This mode allows complete device control and operates exactly like a directly connected serial port. It provides a complete tty device interface between the attached serial device and the network, providing hardware and software flow control.

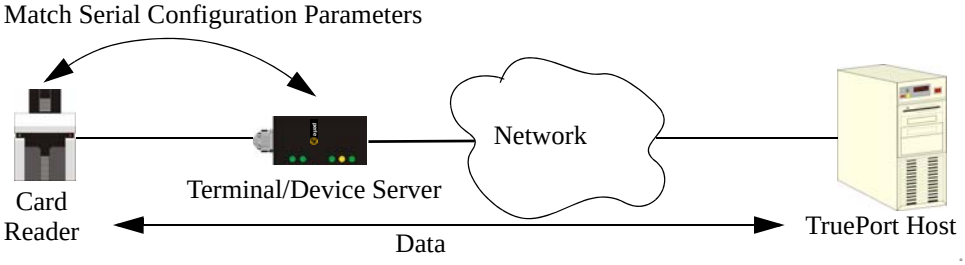
TruePort 6.1 and lower, IOLAN Device Server firmware 3.4 and lower, JetStream, and LanStream in Full Mode use the TCP protocol on the configured port and the UDP protocol on port 668 (some firewalls block UDP packets by default and might need to be reconfigured to support Full Mode communication). TruePort 6.2 and higher and IOLAN Device Server firmware 3.5 and higher do not use the UDP protocol.

The port serial configuration parameters set on the TruePort host must match the serial configuration parameters set on the device (in this example, to the Card Reader), as shown below:



Lite Mode

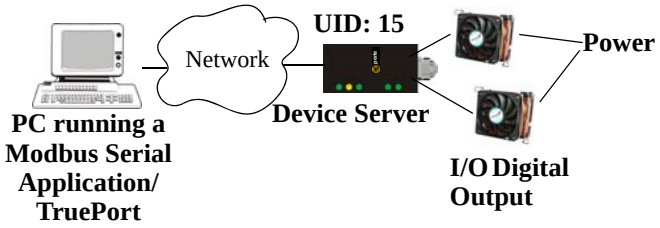
This mode provides a simple raw data interface between the device and the network. Although the port will still operate as a tty device, control signals are ignored. Lite Mode uses the TCP protocol on the configured port. In this mode, the serial communications parameters are configured on the terminal/device server and must match those configured on the device (in this example, a Card Reader), as shown below:



TruePort I/O Access Options

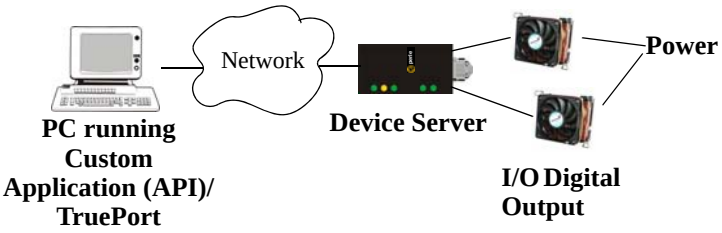
Modbus ASCII/RTU Mode

If you have a Modbus serial application running on a PC that is connected to a network, you can use TruePort as a virtual serial connection to communicate with the Device Server over the network to access I/O data. You also have the option of enabling SSL as a security option to encrypt the data that is communicated between the Device Server and the host machine (SSL/TLS must be configured on both the Device Server and in TruePort).



I/O Signal Mode

If you have a custom application that talks to a serial port, you can use TruePort as a virtual serial port to communicate with the Device Server over the network to access I/O data. You also have the option of enabling SSL as a security option to encrypt the data that is communicated between the Device Server and the host machine (SSL/TLS must be configured on both the Device Server and in TruePort).



Uninstalling TruePort Version 2.0

If TruePort version 2.0 has been installed on your system, it must be uninstalled before the new version can be installed.

1. Find the TruePort patch file to the kernel. It should be found in the **/usr/share/doc/trueport** directory. Remove the patch from the kernel source files by entering the following commands:

   ```
   cd /usr/src/linux
   patch –R –p1 /usr/share/doc/trueport/<trueport_patch_file>
   ```

2. Uninstall TruePort by entering the following command.

   ```
   rpm –e trueport
   ```

3. Recompile and install the kernel based on your specific system. For help on this procedure please refer to your Linux documentation.

The new TruePort software can now be installed.

Uninstalling TruePort

Before you can install a new version of TruePort, you need to uninstall any older version of TruePort. This section is applicable for TruePort versions 2.2 and higher.

Uninstalling an RPM TruePort Installation

To uninstall an RPM TruePort installation, type the following command:

```
rpm -e trueport
```

You are now ready to install the new version of TruePort.

Uninstalling a Tar File TruePort Installation

TruePort Version 2.2 Through 5.0.x

To uninstall a tar file TruePort installation, do the following:

1. Move to the directory where you previously installed TruePort. For example:

   ```
   cd /tmp/trueport-<version>
   ```

2. Type the following command:

   ```
   ./uninstall.sh
   ```

You are now ready to install the new version of TruePort.

TruePort Version 6.0.0 and Higher

To uninstall a tar file TruePort installation, do the following:

1. Move to the TruePort configuration directory:

   ```
   cd /etc/trueport
   ```

2. Type the following command:

   ```
   ./uninstall.sh
   ```

You are now ready to install the new version of TruePort.

Installing TruePort on the TruePort Host

The software for TruePort for Linux can be download in either a source RPM package format (**.rpm**) or a compressed tar file (**.tgz**). When you install TruePort, you will be prompted to install OpenSSL if you don't already have OpenSSL installed or you have a version before 0.9.7.g (the new installation of OpenSSL will not interfere in any way with your existing OpenSSL installation, as it will be installed under the TruePort configuration directory). You only need to install OpenSSL if you plan on using the SSL/TLS feature.

From the RPM Source

Note: The <***packages_directory***> path name in the following instructions will be different depending on the Linux distribution you have installed. (for example, Redhat will have a **redhat** directory; Suse will have a **packages** directory).

The <***rpm_build_command***> will change depending on the version of the RPM utilities installed. For newer versions (that is, 4.2 and newer), the <***rpm_build_command***> is **rpmbuild**. For older versions, it is **rpm**.

To install the TruePort software on your system, do the following:

1. Log in to the Linux server as root user. To get the latest TruePort release, download the TruePort installation file from the www.perle.com/downloads/trueport.shtml website.
2. Copy the TruePort **trueport-<*version*>-<release>.src.rpm** file to the **/tmp** directory.
3. Install the source rpm by typing the following command:

   rpm –ivh trueport-<*version*>-<*release*>.src.rpm

4. Build the binary RPM package for your system by entering the following commands:

   cd /usr/src/<*packages_directory*>/SPECS
   <*rpm_build_command*> -bb trueport-<*version*>.spec

5. You will be asked if you want to install the SSL/TLS feature. If you enter **n** for no, the installation will continue to install TruePort. If you enter **y** for yes, the installation will determine if you have an acceptable version of OpenSSL (version 0.9.7.g or later). If no acceptable version of OpenSSL is found, you will be prompted to continue. If you enter **n** for no, the TruePort installation will be cancelled. If you enter **y** for yes, the TruePort and SSL/TLS installation will continue. The new version of OpenSSL will be installed in the **/etc/trueport** directory and will not impact any existing OpenSSL installations.
6. To install the binary RPM that was just created, enter the following commands:

   cd /usr/src/<*packages_directory*>/RPMS/<*arch*>
   rpm –ivh trueport-<*version*>-<*release*>.<*arch*>.rpm

   The <***arch***> value is the architecture of your Linux machine, for example, **i386**.

From the Tar File

To install the TruePort software on your system, do the following:

1. Log in to the Linux server as root user. To get the latest TruePort release, download the TruePort installation file from the www.perle.com/downloads/trueport.shtml website.
2. Copy the TruePort **.tgz** file to the **/tmp** directory.
3. Unpack the file using the command:

   tar –xzvf trueport-<version>-<*release*>.tgz

4. Build and install the TruePort software by entering the following commands:

   cd /tmp/trueport-<*version*>
   ./tar_install.sh

5. You will be asked if you want to install the SSL/TLS feature. If you enter **n** for no, the installation will continue to install TruePort. If you enter **y** for yes, the installation will determine if you have an acceptable version of OpenSSL (version .9.7.g or later). If no acceptable version of OpenSSL is found, you will be prompted to continue. If you enter **n** for no, the TruePort installation will be cancelled. If you enter **y** for yes, the TruePort and SSL/TLS installation will continue. The new version of OpenSSL will be installed in the **/etc/trueport** directory and will not impact any existing OpenSSL installations.

Configuring TruePort on a Terminal/Device Server

When you add a port, you need to configure the port(s) on the host running TruePort and you also need to configure the port(s) on the terminal server.

Server-Initiated Mode

When you configure TruePort for server-initiated mode, the terminal/device server will initiate communication to the TruePort host.

To configure a terminal/device server for server-initiated mode (which is the default mode), you need to set the **Line Service** to **TruePort** (firmware version 3.0 or higher) or **Silent Raw** and assign the port number to be the same port number configured on the TruePort host (by default, this number starts at 10000).

Note: All versions of the JetStream 4000 and 8500, LanStream 2000, and IOLAN DS Family software support TruePort Full Mode operation. However, the JetStream 6x series software version must be 4.03 or greater.

On the IOLAN Device Server

The following instructions provide an example of how to set up two ports the IOLAN Device Server using the CLI to TruePort. You will set the **Line Service** to **TruePort** (firmware version 3.0 or higher) or **Silent Raw** and on 1-port model you don't specify a line number.

1. Connect to the Device Server (for example, via Telnet).
2. Log in to the Device Server as the **admin** user.
3. Add the host running TruePort to the host table using the add host command as shown in the following example:

   add host linux50 192.152.247.61

   You are now ready to configure the ports that will connect to the TruePort host.

4. To configure the ports, enter each of the following commands:

```
set line 1 service trueport linux50 10000
set line 2 service trueport linux50 10001
set line 3 service trueport linux50 10002
set line 4 service trueport linux50 10003
kill line 1-4
```

5. At the command prompt, type **save** and press **Enter**.
6. At the command prompt, type **logout** and press **Enter**.

The configuration of Device Server's ports is now complete.

On the JetStream/LanStream

Configuring a JestStream\LanStream using the CLI is almost same as the Device Server CLI. You will set the **Line Service** to **Silent Raw**.

Client-Initiated Mode

Note: Client-Initiated mode is available on Terminal Server, Device Server, and Console Sever models with firmware 3.0 or higher.

When you configure TruePort for Client-Initiated mode, the TruePort host will initiate communication with the terminal/device server.

To configure a terminal/device server for Client-Initiated mode, you need to set the **Line Service** to **TruePort**, enable the **Client Initiated** option, and assign the port number to be the same port number configured on the client-initiated configured TruePort host (by default, this number starts at 10001).

On the IOLAN Device Server

The following instructions provide an example of how to set up 4 ports on a IOLAN Device Server for TruePort Client-Initiated mode.

1. Connect to the Device Server (for example, via Telnet).
2. Log in to the Device Server as the **admin** user.
3. To configure the ports, enter each of the following commands:

```
set line 1 service trueport client-initiated on 10001
set line 2 service trueport client-initiated on 10002
set line 3 service trueport client-initiated on 10003
set line 4 service trueport client-initiated on 10004
kill line 1-4
```

4. At the command prompt, type **save** and press **Enter**.
5. At the command prompt, type **logout** and press **Enter**.

The configuration of the Device Server is now complete.

Client I/O Access Mode (I/O Models Only)

Client I/O access mode allows:

- A Modbus RTU/ASCII serial application running on a TruePort host to access Device Server I/O using Modbus commands
- A serial application running on a TruePort host to access Device Server I/O using the Perle API (see the *Utilities* chapter of your *User's Guide* for TruePort API documentation).

Note: Client I/O Access is only available in conjunction with IOLAN Device Servers running version 3.1 or higher.

Modbus I/O Access

To configure a Device Server for Client I/O Access mode for a Modbus RTU/ASCII serial application, you need to enable **I/O TruePort Services**, enable **I/O Modbus Slave**, assign a Modbus slave UID to match the configured on the Modbus RTU serial application, and assign the port number to be the same port number configured UID on the client I/O Access configured TruePort host (by default, this number is 33816).

The following instructions provide an example of how to set up an IOLAN Device Server for TruePort I/O Access.

1. Connect to the Device Server (for example, via Telnet).
2. Log in to the Device Server as the **`admin`** user.
3. To enable the I/O TruePort service, enter following command:

   ```
   set io trueport mode on listen 33816
   ```

4. To enable I/O Modbus slave, enter following command:

   ```
   set io modbus mode on uid 1
   ```

5. Reboot the Device Server by entering the following command:

   ```
   reboot
   ```

   The configuration of the Device Server is now complete.

To configure the TruePort host running a Modbus ASCII serial application to access Device Server I/O, type the following command:

```
addports -client mydeviceserver:33816 -initconnect -io mb_ascii 0 0
```

The command creates a single port configured for Client I/O Access mode which will connect to host **mydeviceserver** on TCP port 33816 and will support a serial Modbus ASCII application.

Perle API I/O Access

To configure a Device Server for Client I/O Access mode for a serial application, you need to enable **I/O TruePort Services**.

The following instructions provide an example of how to set up an IOLAN Device Server for TruePort I/O Access.

1. Connect to the Device Server (for example, via Telnet).
2. Log in to the Device Server as the **admin** user.
3. To enable the I/O TruePort service, enter following command:

   ```
   set io trueport mode on listen 33816
   ```

4. Reboot the Device Server by entering the following command:

   ```
   reboot
   ```

   The configuration of the Device Server is now complete.

To configure the TruePort host running a custom serial application to access Device Server I/O, type the following command:

```
addports -client mydeviceserver:33816 -initconnect -io io_api 0 0
```

The command creates a single port configured for Client I/O Access mode which will connect to host **mydeviceserver** on TCP port 33816 and will support a custom serial application using the Perle API.

Configuring Ports on the TruePort Host

TruePort Device Names

The TruePort installation creates the following master device nodes, used by the driver:

```
/dev/px0000
/dev/px0001
/dev/px0002
```

and so forth up to /dev/px*nnnn* where *nnnn* is the highest port number. The master device nodes are used by the TruePort daemon. For each port a slave node is also created (slave device name in Full or Lite mode):

```
/dev/tx<i>nnnn</i>
```

where *nnnn* is associated with the corresponding master node number. The slave nodes are used by the host applications.

Configuration Methods

After you have configured the ports on the terminal/device server, you need to configure corresponding ports on the TruePort host. The TruePort Host can be configured in the following ways:

1. Use the **addports** script, which will automatically start each port as it is configured. See ***TruePort addports Script Options*** on page 12 for more information.
2. Use the **addports** script to create the total range of TTY ports you required and then use **tpadm** administration tool. See ***TruePort Administration Tool (tpadm) Commands*** on page 15 to remove/add ports to the **config.tp** file using **tpadm**.
3. Use the **addports** script to create the total range of TTY ports and then edit the **/etc/trueport/config.tp** file (see ***config.tp File Syntax*** on page 19), the **/etc/trueport/sslcfg.tp** file (see ***sslcfg.tp File Format*** on page 29), and the **/etc/trueport/pktfwdcfg.tp** file (see ***pktfwdcfg.tp File Format*** on page 26).

TruePort addports Script Options

The **addports** script allows you to add a range of ports to the **config.tp** file and automatically start them, without having to use the **tpadm** utility. You can run **addports** after the TruePort host software is installed. The addports options are as follows:

For Server-Initiated Mode:

```
addports [-l] [-hup] [-ssl] [-k <seconds>]
[-opmode optimize_lan|low_latency|packet_idle_timeout|custom]
[-pktidletime <timeout>] [-pf] [-server <host>] [-openwaittime <seconds>]
[-trace <level>] <firstport> <lastport>
```

For Client-Initiated Mode:

```
addports [-l] [-hup] [-ssl] [-k <seconds>]
[-opmode optimize_lan|low_latency|packet_idle_timeout|custom]
[-pktidletime <timeout>] [-pf]  -client <host>[:<TCP-port>] [-nodisc]
[-retrytime <time>] [-retrynum <number>] [-initconnect]
[-openwaittime <seconds>] [-closedelaytime <seconds>] [-norestorenet]
[-io mb_ascii|mb_rtu|io_api] [-trace <level>] <firstport> <lastport>
```

Parameter	Description
-l	(lower case L) Specifies that the TruePort port will be started in Lite mode. **addports** will configure TruePort for Full mode by default.
-hup	Causes the tty device to automatically be closed when the TCP connection is closed.
-ssl	Enables SSL/TLS on the port. You will automatically be prompted by the SSL/TLS configuration script. For more information see ***Configuring SSL/TLS*** on page 27.
-k <*seconds*>	The time, in seconds, to wait on an idle connection before sending a keep-alive message.
-opmode optimize_lan| low_latency| packet_idle_timeout| custom	Specify one of the following optimization modes: • **optimize_lan**—This option provides optimal network usage while ensuring that the application performance is not compromised. Select this option when you want to minimize overall packet count, such as when the connection is over a WAN. • **low_latency**—This option ensures that all application data is immediately forwarded to the serial device. Select this option for timing-sensitive applications. • **packet_idle_timeout**—This option detects the message, packet, or data blocking characteristics of the serial data and preserves it throughout the communication. Select this option for message-based applications or serial devices that are sensitive to inter-character delays within these messages. • **custom**—This option allows you to define the packet forwarding rules based on the packet definition or the frame definition. This is the same as the -pf option and will launch the Packet Forwarding configuration script (see ***Configuring Packet Forwarding*** on page 24).
-packetidletime <*timeout*>	The minimum time, in milliseconds, between messages that must pass before the data is forwarded to the Device Server. The range is 0-65535. The default is 10 ms.
-pf	Enables packet forwarding on the port. You will automatically be prompted by the packet forwarding configuration script. For more information see ***Configuring Packet Forwarding*** on page 24.

Parameter	Description
-server <*host*>	You can optionally supply the remote host name or IP address that a connection request will be accepted from. The default is to accept connections from any host. The host can be an IPv4 or IPv6 address or a resolvable host name. If specifying an IPv6 address, the address must be enclosed in square brackets ([ ]), for example [2001:0db8:85a3:08d3:1319:8a2e:0370:7348].
-client <*host*>[:<*TCP-port*>]	Specifies a client-initiated connection (meaning that the TruePort host will initiate the connection). You can optionally supply the starting destination TCP port for the connection (the default is 10001 see <*firstport*> option below>). The host can be an IPv4 or IPv6 address or a resolvable host name. If specifying an IPv6 address, the address must be enclosed in square brackets ([ ]), for example [2001:0db8:85a3:08d3:1319:8a2e:0370:7348].
-nodisc	Does not drop the TCP connection for a client-initiated connection when the application closes the slave TTY port.
-retrytime <*time*>	Specifies the number of seconds between TCP connection retries after a client-initiated connection failure. Valid values are 1-255. The default is 30 seconds.
-retrynum <*number*>	Specifies the number of additional retry attempts for a client-initiated connection, beyond the first attempt. Valid values are -1 to 255. If this option is -1, TruePort will attempt to reconnect forever. If this option is set to 0 (zero) and **`-norestornet`** is not specified, TruePort will try to recover a TCP connection once. The default is 5 retries.
-initconnect	Specifies that the TruePort host will try to connect to the Device Server when the TruePort daemon starts, as opposed to waiting for the application to open the serial port before initiating the connection to the Device Server.
-openwaittime <*seconds*>	Specifies the amount of time to wait, in seconds, for a TCP connection to be established when the serial port is opened. You can specify the following values: • **-2**, open the serial port and wait forever for the TruePort connection to come up. • **-1**, open the serial port without waiting, even if there is no network connection, and don't give an error. Any written data is discarded if no network exists. • **0**, open the serial port without waiting, and return an error (EIO) if no network connection exists. If a network connection exists, then no error is returned. • **1-65535**, wait the specified seconds for a network connection to be established. If a timeout occurs before a network connection is established, an error is returned (EIO). The TruePort connection is fully established when: • The TCP connection between the terminal/device server and the TruePort host is up. • The SSL/TLS negotiation succeeds (if used). • The TruePort Full mode protocol negotiation succeeds (if used). In all of the above cases, the O_NONBLOCK and CLOCAL flags are obeyed. If the O_NONBLOCK flag is specified and no network exists, the error will be returned immediately. If O_NONBLOCK is not specified in the open and CLOCAL is not set, the open will not return until the DCD signal is present. The range is -2 to 65535. The default is 5 seconds.

Parameter	Description
-closedelaytime <***seconds***>	Specifies the amount of time, in seconds, to wait after the application closes the serial port, before the TCP connection is closed to avoid bringing the TCP connection down and up if the application is closing and opening the COM port often. The range is 0-65535. The default is 3 seconds.
-norestorenet	By default, when the network connection fails for client-initiated mode, TruePort will attempt to restore it. When this option is specified, if the network connection fails, there is no attempt to restore it.
-io mb_ascii **-io mb_rtu** **-io io_api**	Enables client I/O access for this client-initiated session for one of the following: • A serial Modbus application configured for either the ASCII or RTU protocol will be using this port. • A custom serial application using the Perle I/O Access API will be using this port. If you did not specify the ***:TCP-port*** option with **-client**, the **-io** option will make the destination TCP port default to 33816 (the default value of the TruePort client in the Device Server).
-trace <***level***>	The trace level for debugging purposes. The default is 1. The trace file for each port can be found under **/etc/trueport/trace.px***nnnn*, where *nnnn* is the TTY port number.
<***firstport***>	The first TTY to add starting at 0 (added as port 10001 for a client-initiated (TruePort) connection or port 10000 for a server (terminal/device server) initiated connection).
<***lastport***>	The last TTY to add.

The **addports** script also creates pseudo master TTY device nodes **px*nnnn*** and pseudo slave TTY device nodes **tx*nnnn*** in the **/dev** directory.

The **/dev/px*nnnn*** devices are used by the TruePort daemon and are shown in the **config.tp** file.

The **/dev/tx*nnnn*** devices are used by the Linux applications and once the TruePort daemon is started on each port you can use them as standard Linux serial TTY's

Examples

Adding Server-Initiated Ports

The following **addports** command will create 4 ports configured for Server-Initiated mode which will listen for connections from host **myjetstream** on TCP ports 10000 to 10003, while running in TruePort Full mode.

```
addports -server myjetstream 0 3
```

Adding Client-Initiated Ports

The following **addports** command will create 4 ports configured for Client-Initiated mode which will connect to host **myiolansds4** on TCP ports 10001 to 10004, while running in TruePort Lite mode.

```
addports -l -client myiolansds4 0 3
```

Adding Client I/O Access Ports

The following **addports** command will create a single port configured for Client I/O Access mode which will connect to host **myiolaniods1** on TCP port 33816 and will support a serial Modbus RTU application.

```
addports -client myiolaniods1 -io mb_rtu 0 0
```

TruePort Administration Tool (tpadm) Commands

This section describes the commands and syntax for the TruePort Administration tool.

Syntax

Description You can use the **tpadm** utility to add, list, start, and delete ports.

Syntax

```
tpadm -a <new_portnumber> [-m|-n]
 [--opmode optimize_lan|low_latency|packet_idle_timeout|custom]
 [--pktidletime <milliseconds>]
 [-e|-F|-e -F [-c [<existing_host>:]<existing_portnumber>]]
 [-C <host> [-r <seconds>] [-R <retries>] [-o] [--initconnect]
   [--openwaittime <seconds>] [--closedelaytime <seconds>]
   [--norestorenet] [-I mb_ascii|mb_rtu|io_api]]
 [-S <host>] [-t <level>] [-h] -p <ttyname> [-k <seconds>]

tpadm -l <portnumber>|<host>:|<host>:<TCP_number>|ALL

tpadm -s <portnumber>|<host>:|<host>:<TCP_number>|ALL

tpadm -d <portnumber>|<host>:|<host>:<TCP_number>
```

Options

-a *<new_portnumber>*

The TCP/IP port number configured for the port on the remote device or terminal server. We recommend that you use the range 10000+.

-m

Adds a terminal in TruePort Full Mode (not Lite Mode) for full device control. This is the default.

-n

Adds a terminal in TruePort Lite Mode (not Full Mode) for terminal/device server device control.

--opmode optimize_lan|low_latency|packet_idle_timeout|custom

Specify one of the following optimization modes:

- **optimize_lan**—This option provides optimal network usage while ensuring that the application performance is not compromised. Select this option when you want to minimize overall packet count, such as when the connection is over a WAN.
- **low_latency**—This option ensures that all application data is immediately forwarded to the serial device. Select this option for timing-sensitive applications. This is the default.
- **packet_idle_timeout**—This option detects the message, packet, or data blocking characteristics of the serial data and preserves it throughout the communication. Select this option for message-based applications or serial devices that are sensitive to inter-character delays within these messages.
- **custom**—This option allows you to define the packet forwarding rules based on the packet definition or the frame definition. This is the same as the -F option and will launch the Packet Forwarding configuration script (see ***Configuring Packet Forwarding*** on page 24).

--pktidletime

The minimum time, in milliseconds, between messages that must pass before the data is sent. The range is 0-65535. The default is 10 ms.

-e

Enables SSL/TLS for the port. You will automatically be prompted for the SSL/TLS configuration information when you use this command line option. See ***Configuring SSL/TLS*** on page 27 for more information.

-F

Enables packet forwarding for this port. You will automatically be prompted for the Packet Forwarding configuration information when you use this command line option. See ***Configuring Packet Forwarding*** on page 24 for more information.

-c [<*existing_host*>**:]**<***existing_portnumber***>

Copies the specified SSL/TLS and/or packet forwarding configuration data from the specified entry to the new port entry being created

-C <*host*>

Enables a client-initiated connection (by the TruePort host) for this session and will connect to the specified host and port number. The host can be an IPv4 or IPv6 address or a resolvable host name. If specifying an IPv6 address, the address must be enclosed in square brackets ([ ]), for example [2001:0db8:85a3:08d3:1319:8a2e:0370:7348].

-r <*seconds*>

Sets the number of seconds between TCP connection retries. The default is 1 second.

-R <*retries*>

Specifies the number of additional retry attempts for a client-initiated connection, beyond the first attempt. Valid values are -1 to 255. If this option is -1, TruePort will attempt to reconnect forever. If this option is set to 0 (zero) and **`-norestornet`** is not specified, TruePort will try to recover a TCP connection once. The default is 5 retries.

-o

Keeps the client-initiated TCP connection open even when the application closes the slave TTY port.

-I mb_ascii|mb_rtu|io_api

Enables client I/O access for this client-initiated session.

--initconnect

Specifies that the TruePort host will try to connect to the Device Server when the TruePort daemon starts, as opposed to waiting for the application to open the serial port before initiating the connection to the Device Server.

--openwaittime <*seconds*>

Specifies the amount of time to wait, in seconds, for a TCP connection to be established when the serial port is opened. You can specify the following values:

- **-2**, open the serial port and wait forever for the TruePort connection to come up.
- **-1**, open the serial port without waiting, even if there is no network connection, and don't give an error. Any written data is discarded if no network exists.
- **0**, open the serial port without waiting, and return an error (EIO) if no network connection exists. If a network connection exists, then no error is returned.
- **1-65535**, wait the specified seconds for a network connection to be established, before returning. If a timeout occurs before a network connection is established, an error is returned (EIO).

The TruePort connection is fully established when:

- The TCP connection between the terminal/device server and the TruePort host is up.
- The SSL/TLS negotiation succeeds (if used).
- The TruePort Full mode protocol negotiation succeeds (if used).

In all of the above cases, the O_NONBLOCK and CLOCAL flags are obeyed. If the O_NONBLOCK flag is specified and no network exists, the error will be returned immediately. If O_NONBLOCK is not specified in the open and CLOCAL is not set, the open will not return until the DCD signal is present.

The range is -2 to 65535. The default is 5 seconds.

--closedelaytime <*seconds*>

Specifies the amount of time, in seconds, to wait after the application closes the serial port, before the TCP connection is closed. The range is 0-65535. The default is 3 seconds.

--norestorenet

By default, when the network connection fails for client-initiated mode, TruePort will attempt to restore it. When this option is specified, if the network connection fails, there is no attempt to restore it.

-S <*host*>

Specifies the remote host name or IP address that a connection request will be accepted from in Server-Initiated mode. The default is to accept connections from any host. The host can be an IPv4 or IPv6 address or a resolvable host name. If specifying an IPv6 address, the address must be enclosed in square brackets ([ ]), for example [2001:0db8:85a3:08d3:1319:8a2e:0370:7348].

-t <*level*>

Sets the trace level for debugging. The default is 1.

-h

Causes the tty device to automatically be closed when the TCP connection is closed.

-p <*ttyname*>

The tty name for the port. We recommend that you use the name range px0001-px0255.

-k <*seconds*>

The time, in seconds, to wait on an idle connection before sending a keep-alive message.

-l

Displays the port entries in the **`config.tp`** file.

-d <*portnumber*><*host*>**:**|<*host*>**:**<*TCP_number*>

Deletes the specified port entry from the **`config.tp`** file. The host can be an IPv4 or IPv6 address or a resolvable host name. If specifying an IPv6 address, the address must be enclosed in square brackets ([ ]), for example [2001:0db8:85a3:08d3:1319:8a2e:0370:7348].

-s <*portnumber*>|<*host*>**:**|<*host*>**:**<*TCP_number*>|**ALL**

Starts a specific TruePort port or all the TruePort ports. The host can be an IPv4 or IPv6 address or a resolvable host name. If specifying an IPv6 address, the address must be enclosed in square brackets ([ ]), for example [2001:0db8:85a3:08d3:1319:8a2e:0370:7348].

Examples

Adding a Port

To add port 10000 in Full mode with SSL/TLS enabled, use the following command:

```
tpadm -a 10000 -e -p px0000
```

To add a Client-Initiated port to connect to host **`myiolansds4`** on remote port 10001 with packet forwarding enabled, use the following command:

```
tpadm -a 10001 -F -p px0002 -C myiolansds4
```

To add a I/O Access port to connect to host **`myiolaniods1`** on port 33816 to us a serial Modbus RTU application with a keep alive time of 3 minutes, use the following command:

```
tpadm -a 33816 -p px0003 -C myiolaniods1 -I mb_rtu - k 180
```

Deleting a Port

To delete port 10000, use the following command:

```
tpadm -d 10000
```

To delete port 10001 on host **myiolands** use the following command:

```
tpadm -d  myiolands:10000
```

Note: The <*host*>:<*port*> combination you use must exist in the **config.tp** configuration file.

When you remove a terminal using this command, it does not stop the software running, it just deletes the entry for this terminal in the **config.tp** configuration file. You must then kill the TruePort daemon process.

Displaying Port Entries

To displays the ports configured in the config.tp file, use the following command:

```
tpadm -l all
```

To display all the ports for a specific host in the config.tp file, use the following command:

```
tpadm -l myiolands:
```

To display a specific port for a specific host in the configuration file, use the following command:

```
tpadm -l myiolands:10002
```

Starting the TruePort Daemon

To start port number 10000, use the following command:

```
tpadm -s 10000
```

To start all configured ports, use the following command:

```
tpadm -s ALL
```

To start port number 10001 on host 172.16.45.8, use the following command:

```
tpadm -s  172.16.45.8:10001
```

To start all configured port on host **myjetstream**, use the following command:

```
tpadm -s myjetstream:
```

config.tp File Syntax

Note: If you use **addports** to enable TruePort you do not need to use the **tpadm** utility.

An entry in the **config.tp** configuration file used to control a terminal in server Full Mode with all the options enabled looks like this:

```
/usr/bin/trueportd -trueport -ssl -hup -tty /dev/px0001 -port 10000
                   -server myjetstream -ka 30 -trace 4 -opmode low_latency
                   -openwaittime 12 -closedelaytime 15
```

An entry in the **config.tp** configuration file used to control a terminal/device server in Full Mode via Client-Initiated mode and some options enabled looks like this:

```
/usr/bin/trueportd -trueport -ssl -pf -hup -tty /dev/px0000 -port 10001
                   -client myiolansds4 -retrytime 3 -retrynum 10 -nodisc
                     -ka 30 -trace 4
```

The **config.tp** port parameter options are:

-trueport	Enables TruePort Full Mode (not TruePort Lite) for full device control.
-ssl	Enables SSL/TLS on the port.
-hup	Causes the tty device to automatically be closed when the TCP connection is closed.
-tty <*ttyname*>	<***ttyname***> is the tty name for the port. This must be the complete path name. We recommend that you use the name range px0001-px0255.
-port <*port_number*>	For a server-initiated connection (terminal/device server), the TCP port number the TruePort daemon will listen on for connection requests. For a client-initiated connection (TruePort host), the Device Server TCP port number (**DS Port**) the TruePort daemon will attempt to connect to.
-ka <*seconds*>	<***seconds***> is the number of seconds to wait on an idle connection before sending a keep-alive message.
-trace <1-4>	<1-4> is the trace level for debugging purposes, the default is **1**.
-client <*host*>	Indicates a client-initiated connection. The <***host***> can be IPv4, IPv6, or a resolvable host name. If specifying an IPv6 address, the address must be enclosed in square brackets ([ ]), for example [2001:0db8:85a3:08d3:1319:8a2e:0370:7348].
-retrytime <*seconds*>	For client-initiated connections, the number of seconds between TCP connection retries. The default is 1 second.
-retrynum <*number*>	Specifies the number of additional retry attempts for a client-initiated connection, beyond the first attempt. Valid values are -1 to 255. If this option is -1, TruePort will attempt to reconnect forever. If this option is set to 0 (zero) and **-norestornet** is not specified, TruePort will try to recover a TCP connection once. The default is 5 retries.
-nodisc	For client-initiated connections, does not close the TCP connection when the application closes the slave TTY port.
-io mb_ascii| mb_rtu| io_api	For client-initiated connections, enables I/O access for the session. • A serial Modbus application configured for either the ASCII or RTU protocol will be using this port. • A custom serial application using the Perle I/O Access API will be using this port. If you did not specify the ***:TCP-port*** option with **-client**, the **-io** option will make the destination TCP port default to 33316 (the default listen TCP port number of the I/O TruePort services in the IOLAN Device Server).

-server <*host*> Specifies the remote host name or IP address that a connection request will be accepted from in Server mode. The default is to accept connections from any host. The <***host***> can be IPv4, IPv6, or a resolvable host name. If specifying an IPv6 address, the address must be enclosed in square brackets ([ ]), for example [2001:0db8:85a3:08d3:1319:8a2e:0370:7348].

-nagleoff For client-initiated connections, turn off the TCP Nagle Algorithm, which inserts a short delay so that each character is not sent individually, but sent in small packets instead. The default is On.

-opmode optimize_lan| low_latency| packet_idle_timeout| custom Specify one of the following optimization modes:

- **optimize_lan**—This option provides optimal network usage while ensuring that the application performance is not compromised. Select this option when you want to minimize overall packet count, such as when the connection is over a WAN.
- **low_latency**—This option ensures that all application data is immediately forwarded to the serial device. Select this option for timing-sensitive applications.
- **packet_idle_timeout**—This option detects the message, packet, or data blocking characteristics of the serial data and preserves it throughout the communication. Select this option for message-based applications or serial devices that are sensitive to inter-character delays within these messages.
- **custom**—This option allows you to define the packet forwarding rules based on the packet definition or the frame definition. This is the same as the -pf option and will launch the Packet Forwarding configuration script (see ***Configuring Packet Forwarding*** on page 24).

-pktidletime <*seconds*> The minimum time, in milliseconds, between messages that must pass before the data is sent. The range is 0-65535. The default is 10 ms.

-openwaittime <*seconds*> Specifies the amount of time to wait, in seconds, for a TCP connection to be established when the serial port is opened. You can specify the following values:

- **-2**, open the serial port and wait forever for the TruePort connection to come up.
- **-1**, open the serial port without waiting, even if there is no network connection, and don't give an error. Any written data is discarded if no network exists.
- **0**, open the serial port without waiting, and return an error (EIO) if no network connection exists. If a network connection exists, then no error is returned.
- **1-65535**, wait the specified seconds for a network connection to be established, before returning. If a timeout occurs before a network connection is established, an error is returned (EIO).

The TruePort connection is fully established when:

- The TCP connection between the terminal/device server and the TruePort host is up.
- The SSL/TLS negotiation succeeds (if used).
- The TruePort Full mode protocol negotiation succeeds (if used).

In all of the above cases, the O_NONBLOCK and CLOCAL flags are obeyed. If the O_NONBLOCK flag is specified and no network exists, the error will be returned immediately. If O_NONBLOCK is not specified in the open and CLOCAL is not set, the open will not return until the DCD signal is present.

The range is -2 to 65535. The default is 5 seconds.

-closedelaytime <*seconds*>	Specifies the amount of time, in seconds, to wait after the application closes the serial port, before the TCP connection is closed. The range is 0-65535. The default is 3 seconds.
-initconnect	Specifies that the TruePort host will try to connect to the Device Server when the TruePort daemon starts, as opposed to waiting for the application to open the serial port before initiating the connection to the Device Server.
-norestorenet	By default, when the network connection fails for client-initiated mode, TruePort will attempt to restore it. When this option is specified, if the network connection fails, there is no attempt to restore it.

Managing Ports on the TruePort Host

Starting TruePort

A TruePort daemon needs to be run for each port configured. There are three ways to start TruePort daemons:

- Use the **`addports`** script, which will automatically starts each port as it is configured.
- When the TruePort host reboots, a TruePort daemon for each port configured in the **`config.tp`** file will automatically be started by the **`trueport`** script, which can be found in the **`/etc/init.d/trueport`** script. The **`trueport`** script is enabled when the TruePort software is installed.
- Enter the **`tpadm -s`** command to start specific individual ports or all the ports at one time; see ***TruePort Administration Tool (tpadm) Commands*** on page 15 for the command syntax.

Deleting a Single Port

To delete serial ports, do the following:

1. In the **`/etc/trueport`** directory, use an editor to delete the port entry in the **`config.tp`** file or type the following command:

   `tpadm -d <`*`portnumber`*`>|<`*`host`*`>:|<`*`host`*`>:<`*`portnumber`*`>`

2. You must then kill the TruePort daemon process.
3. If you had configured a login for this port, you should remove it using the **`tplogin -r`** command (see ***tplogin*** on page 22 for more information).

Deleting All Ports

There is a script you can run called **`cleanports`** that will kill all the TruePort daemon processes and delete all entries in the **`config.tp`**, **`sslcfg.tp`**, and **`pktfwdcfg.tp`** files, with the exception of any lines that have been commented out.

If you configured any logins, you should remove them by using the **`rmlogins`** command (see ***rmlogins*** on page 24 for more information).

Restarting all Ports

The **`/etc/init.d/trueport`** script can be used to **stop**, **start,** or **restart** the trueportd daemon for all ports configured in the **`config.tp`** file.

To restart all the configured ports type the following:

```
/etc/init.d/trueport restart
```

Managing Logins

Several configuration scripts are included in your TruePort installation, which can be used to manage logins for the configured TruePort devices.

Note: The following scripts assume the Linux system uses the **inittab** file used by the sysv-compatible init process. If this is not the case, do not use these script files to manage your Logins.

tplogin

The **tplogin** script adds, enables, disables, removes, or lists configured logins for a TruePort device.

Note: To add or remove logins for more than one port, you may wish to use the **addlogins** and **rmlogins** scripts.

Syntax

Description Uses the system's **/sbin/agetty** program to add, enable, disable, remove, or lists configured logins for a TruePort device.

Syntax

```
tplogin -a <port_number> [<baud_rate>]

tplogin -e <port_number>

tplogin -d <port_number>

tplogin -r <port_number>

tplogin -l
```

Options

-a

Adds an agetty entry for the port in the **/etc/inittab**.

-e

Enables agetty for the port.

-d

Disables agetty for the port.

-r

Removes the agetty entry for the port in the **/etc/inittab** file.

-l

Lists the login entries.

<port_number>

The port number, range is 0-255.

<baud_rate>

The baud rate the getty will open the port at. If not provided or null, the default will be 9600.

Examples

tplogin -a 10 19200

This example adds a login for device **/dev/tx0010** at **19200** baud.

tplogin -a 21

This example adds a login for device **/dev/tx0021**. The default baud rate of 9600 will be used.

tplogin -r 10

This example removes the login for **/dev/tx0010** created in the first example.

tplogin -d 21

This example disables the login for **/dev/tx0021**, but does not remove it.

addlogins

The **addlogins** script adds logins for a range of ports, using the **tplogin** script.

Syntax

Description Adds logins for a range of ports by calling the **tplogin** script.

Syntax **addlogins [-t <*baud_rate*>] <*first*> <*last*>**

Options

-t <*baud_rate*>

Indicates the baud rate to use for the port(s). If not given, the **tplogin** script's default will be used (9600).

<*first*>

The number that specifies the start of the range of ports to add logins for. A login for a single port can be added by setting both ***first*** and ***last*** to that port's number.

<*last*>

The number that specifies the end of the range of ports to add logins for. A login for a single port can be added by setting both ***first*** and ***last*** to that port's number.

Examples

addlogins -t 4800 0 95

This example adds logins for devices **/dev/tx0000** to **/dev/tx0095**. The ports will be set to **4800** baud.

addlogins 5 12

This example adds logins for devices **/dev/tx0005** to **/dev/tx0012**.

rmlogins

The **rmlogins** removes logins for a range of ports, using the tplogin script. Its usage is similar to the **addlogins** script used to create logins.

Syntax

Description Removes logins for a range of ports by calling the **tplogin** script.

Syntax **rmlogins <*first*> <*last*>**

Options <***first***>

The number that specifies the start of the range of ports to remove logins for. A login for a single port can be removed by setting both ***first*** and ***last*** to that port's number.

<***last***>

The number that specifies the end of the range of ports to remove logins for. A login for a single port can be removed by setting both ***first*** and ***last*** to that port's number.

Examples

rmlogins 0 95

Removes logins for devices **/dev/tx0000** to **/dev/tx0095**.

rmlogins 5 12

Removes logins for devices **/dev/tx005** to **/dev/tx0012**.

Configuring Packet Forwarding

The Packet Forwarding feature allows you to control how the data written by a Linux application to the slave TTY port is packetized before forwarding the packet onto the LAN network.

Configuration Script

Packet forwarding is configured using the **addports** or **tpadm** utilities. If packet forwarding is enabled, a configuration script is automatically launched as follows:

```
Enable Packet Definition (y/n): y
Packet Size [0] ( 1 - 1024):
Idle Time ([0] - 65535):
Force Transmit Time ([0] - 65535):
Enable End Trigger1 (y/n): y
End Trigger1 Character ([0] - ff):
Enable End Trigger2 (y/n):
End Trigger2 Character ([0] - ff):
Enter the Forwarding Rule ([trigger], trigger+1, trigger+2, strip-trigger):

Enable Packet Definition (y/n): n
Enable Frame Definition (y/n): y
SOF1 Character ([0] - ff):
Enable SOF2 (y/n):
SOF2 Character ([0] - ff):
Transmit SOF Character(s) ([on]/off):
EOF1 Character ([0] - ff):
Enable EOF2 (y/n):
EOF2 Character ([0] - ff):
Enter the Forwarding Rule ([trigger], trigger+1, trigger+2, strip-trigger):
```

The following table describes the options:

Packet Definition	This section allows you to set a variety of packet definition options. The first criteria that is met causes the packet to be transmitted. For example, if you set a **Force Transmit Timer** of **1000** ms and a **Packet Size** of **100** bytes, whichever criteria is met first is what will cause the packet to be transmitted.
Packet Size	The number of byte that must be written by the application before the packet is transmitted to the network. A value of zero (0) ignores this parameter. Valid values are 0-1024 bytes. The default is 0.
Idle Time	The amount of time, in milliseconds, that must elapse between characters before the packet is transmitted to the network. A value of zero (0) ignores this parameter. Valid values are 0-65535 ms. The default is 0.
Force Transmit Timer	When the specified amount of time, in milliseconds, elapses after the first character is written by the application, the packet is transmitted. A value of zero (0) ignores this parameter. Valid values are 0-65535 ms. The default is 0.
End Trigger1 Character	When enabled, specifies the character that when written by the application will define when the packet is ready for transmission. The content of the packet is based on the Trigger Forwarding Rule. Valid values are in hex 0-FF. The default is 0.
End Trigger2 Character	When enabled, creates a sequence of characters that must be written by the application to specify when the packet is ready for transmission (if the End Trigger1 character is not immediately followed by the End Trigger2 character, TruePort waits for another End Trigger1 character to start the End Trigger1/End Trigger2 character sequence). The content of the packet is based on the Trigger Forwarding Rule. Valid values are in hex 0-FF. The default is 0.
Frame Definition	This section allows you to control the frame that is transmitted by defining the start and end of frame character(s). If the internal buffer (1024 bytes) is full before the EOF character(s) are received, the packet will be transmitted and the EOF character(s) search will continue. The default frame definition is SOF=00 and EOF=00.
SOF1 Character	When enabled, the Start of Frame character defines the first character of the frame, any character(s) received before the Start of Frame character is ignored. Valid values are in hex 0-FF. The default is 0.
SOF2 Character	When enabled, creates a sequence of characters that must be received to create the start of the frame (if the SOF1 character is not immediately followed by the SOF2 character, TruePort waits for another SOF1 character to start the SOF1/SOF2 character sequence). Valid values are in hex 0-FF. The default is 0.
Transmit SOF Character(s)	When enabled, the SOF1 or SOF1/SOF2 characters will be transmitted with the frame. If not enabled, the SOF1 or SOF1/SOF2 characters will be stripped from the transmission.
EOF1 Character	Specifies the End of Frame character, which defines when the frame is ready to be transmitted. The content of the frame is based on the Trigger Forwarding Rule. Valid values are in hex 0-FF. The default is 0.

EOF2 Character — When enabled, creates a sequence of characters that must be received to define the end of the frame (if the EOF1 character is not immediately followed by the EOF2 character, TruePort waits for another EOF1 character to start the EOF1/EOF2 character sequence), which defines when the frame is ready to be transmitted. The content of the frame is based on the Trigger Forwarding Rule. Valid values are in hex 0-FF. The default is 0.

Trigger Forwarding Rule — Determines what is included in the Frame (based on the EOF1 or EOF1/EOF2) or Packet (based on Trigger1 or Trigger1/Trigger2). Choose one of the following options:

- **Strip-Trigger**—Strips out the EOF1, EOF1/EOF2, Trigger1, or Trigger1/Trigger2, depending on your settings.
- **Trigger**—Includes the EOF1, EOF1/EOF2, Trigger1, or Trigger1/Trigger2, depending on your settings.
- **Trigger+1**—Includes the EOF1, EOF1/EOF2, Trigger1, or Trigger1/Trigger2, depending on your settings, plus the first byte that follows the trigger.
- **Trigger+2**—Includes the EOF1, EOF1/EOF2, Trigger1, or Trigger1/Trigger2, depending on your settings, plus the next two bytes received after the trigger.

pktfwdcfg.tp File Format

The packet forwarding configuration file is called **`pktfwdcfg.tp`** and is broken up into ports and their defined values as shown in the example below:

```
[10001]
packet_size = 1
idle_time = 2
force_transmit_time = 3
[mysds:10002]
SOF1_char = aa
SOF2_char = bb
transmit_SOF_chars = off
EOF1_char = cc
EOF2_char = dd
trigger_forwarding_rule = trigger
[yoursds:10003]
packet_size = 1000
idle_time = 99
force_transmit_time = 10000
end_trigger1_char = aa
end_trigger2_char = bb
trigger_forwarding_rule = trigger
[172.16.44.21:10004]
packet_size = 1000
idle_time = 99
force_transmit_time = 10000
end_trigger1_char = aa
end_trigger2_char = bb
trigger_forwarding_rule = trigger
```

Configuring SSL/TLS

The SSL/TLS feature is designed to work with the SDS, SCS, and STS Device Server models. When TruePort is used with the Device Server, the cipher specified by the Device Server will be used for the TruePort connection. Also, if the Device Server is set for **SSL/TLS Type Server**, then you need to set the **TruePort SSL type** to **client**, and vice versa.

SSL/TLS Configuration Information

SSL/TLS is configured using the **`addports`** or **`tpadm`** utilities. If SSL/TLS is enabled, the following prompts will ask for the SSL/TLS configuration information:

```
Certificate file name (full path and file name): /etc/trueport/sslcert.pem
SSL type (client or server): client
SSL/TLS version (any, TLSv1, or SSLv3]: any
Perform peer verification (y/n): y
```

The next section is asked only if peer verification is performed. If you press **Enter** instead of entering a value, the parameter will not appear in the **`sslcfg.tp`** file for peer validation.

Note: The values that you enter here are case sensitive, so the peer certificate must match exactly or the connection will fail.

```
CA file name (full path and file name): /etc/trueport/ca.pem
Country (2 letter code): CA
State or Province: Ontario
Locality (e.g. city): Markham
Organisation (e.g. company): Acme Software
Organisation Unit (e.g. section): Engineering
Common Name (e.g. your name or your server's hostname): linux50
Email Address: engineering@acme.com
```

The following section provides more information about the SSL/TLS configuration parameters:

Certificate file name — The full path and file name of the certificate file. If you press **Enter**, the default path, **`/etc/trueport/sslcert.pem`**, will be used.

SSL type — Specify whether the TruePort daemon will act as an SSL/TLS client or server.

SSL/TLS version — Specify whether you want to use:

- **Any**—The TruePort daemon will try a TLSv1 connection first. If that fails, it will try an SSLv3 connection. If that fails, it will try an SSLv2 connection.
- **TLSv1**—The connection will use only TLSv1.
- **SSLv3**—The connection will use only SSLv3.

Perform peer validation — The certificate received from the peer will be verified against the CA list, along with any values entered in the validation criteria, for an SSL connection; any fields left blank will not be validated against the peer certificate.

CA file name — The full path and file name of the CA (certificate authority) file. If you press **Enter**, the default path, **`/etc/trueport/ca.pem`**, will be used.

Country — A two character country code; for example, US.

State or Province	Up to a 128 character entry for the state/province; for example, IL.
Locality	Up to a 128 character entry for the location; for example, a city.
Organisation	Up to a 64 character entry for the organisation; for example, Acme Software.
Organisation Unit	Up to a 64 character entry for the unit in the organisation; for example, Payroll.
Common Name	Up to a 64 character entry for common name; for example, the host name or fully qualified domain name.
Email Address	Up to a 64 character entry for an email address; for example, acct@anycompany.com.

SSL/TLS Support Files

When you enable the SSL/TLS option for a port, you need to make sure the TruePort host and Device Server have the appropriate support files: certificates/private keys and/or the CA list file. The IOLAN CD-ROM contains a self-signed RSA certificate named **`samplecert.pem`**. The **`samplecert.pem`** file can be used for both the certificate file on the SSL/TLS server and the CA list file on the SSL/TLS client.

TruePort Port Configured as SSL/TLS Server

When the TruePort port is configured as an SSL/TLS server, the SSL/TLS private key and certificate is required for all key exchange methods except ADH (Anonymous Diffie-Hellman). The private key cannot be encrypted since TruePort on Linux does not support the configuration of an SSL/TLS passphrase. The private key needs to be appended to the certificate file, to create one certificate/private key file. This can be done using the Linux command **`cat myprivatekey.pem >> mycert.pem`**. This certificate/private key file then becomes the TruePort certificate. Copy the TruePort certificate file to the directory you specified in the SSL/TLS configuration.

If the TruePort SSL/TLS server is configured to verify an SSL client, a CA list file is also required. The CA list file is a certificate, or list of certificates, of the Certificate Authorities (CA) who created and signed the peer certificates (the peer certificate(s) must be downloaded to the Device Server).

TruePort Port Configured as SSL/TLS Client

When the TruePort port is configured as an SSL/TLS client and peer verification is configured, a CA list file is required. The CA list file is a certificate, or list of certificates, of the Certificate Authorities (CA) who created and signed the peer certificates (the peer certificate(s) must be downloaded to the Device Server). This CA list file should be copied to the TruePort host directory specified in the SSL/TLS configuration.

sslcfg.tp File Format

The **sslcfg.tp** file is created in the following format:

```
[10001]
certificate-file = /etc/trueport/sslcert.pem
ssl-type = server
ssl-version = any
verify-peer = yes
CA-file = /etc/trueport/ca.pem
country = CA
state-province = Ontario
locality = Markham
organisation = Acme Software
organisation-unit = Engineering
common-name = linux50
email = engineering@acme.com
[yoursds:10002]
certificate-file = /etc/trueport/sslcert.pem
ssl-type = client
ssl-version = TLSv1
verify-peer = yes
CA-file = /etc/trueport/ca.pem
country = UK
locality = London
common-name = linuxuk
```

The [10001] specifies the port for which the SSL/TLS configuration parameters are configured.

SSL/TLS Trouble Shooting

If you are experiencing problems obtaining a successful SSL/TLS connection, you can add the **-trace 4** option at the end of the appropriate port entry in the **config.tp** file. After editing the **config.tp** file, you will have to kill the TruePort daemon process for the port and restart it again. Adding the **-trace** option will create a trace file called **/etc/trueport/trace.*xxxxx***, where ***xxxxx*** is the TCP/IP port number; for example, **/etc/trueport/trace.10000**.

Could not obtain peer's certificate

Reason 1	User has selected a cipher key exchange of ADH (anonymous Diffie-Hellman) and enabled Peer verification. ADH does not use certificates so they will not be sent in an SSL/TLS handshake.
Solution 1	Disable Peer Verification or change to a cipher suite that uses certificates.
Reason 2	User has selected Peer Verification on the configured SSL/TLS server and has not configured a certificate for the client.
Solution 2	Either disable peer verification on the SSL/TLS server or configure a certificate for the SSL/TLS client.

SSL_accept failed on the SSL/TLS server device.

Reason	The device has failed to accept an SSL/TLS connection on top of a TCP connection that has just been established. This could indicate that the peer from which TruePort is trying to accept a connection from is not configured for SSL/TLS.
Solution	Verify that the peer has been configured for an SSL/TLS client connection.

Certificate did not match configuration

Reason	The message is displayed when **Verify Peer Certificate** has been enabled, but the configured **Validation Criteria** does not match the corresponding data in the certificate received from the peer.
Solution	The data configured must match exactly to the data in the certificate. The data is also case sensitive.

Encrypted private keys are not supported in TruePort

Reason	This message is displayed by the TruePort daemon when the user has created a certificate with an encrypted private key for TruePort. This applies to either **Client-Initiated** mode or **Server-Initiated** mode with configured peer validation criteria.
Solution	Create a certificate with a private key that is not encrypted.

unknown protocol message when trying to make an SSL/TLS connection

Reason 1	This will be displayed when both sides of the TCP connection are configured as SSL/TLS clients.
Solution 1	Change one of the end points to act as an SSL/TLS server.
Reason 2	One of the endpoints is not configured for SSL/TLS.
Solution 2	Make sure both endpoints are configured for SSL/TLS, verify that one is a client and the other is a server.

tlsv1 alert handshake failure or **sslv3 alert handshake failure**

Reason	The remote site has an SSL/TLS error and is sending this message with an alert message.
Solution	Look at the error messages on the remote end and fix the problem indicated.

Certificate verify failed.

Reason 1	TruePort has been configured to verify the peer certificate and there is a mismatch between the peer's certificate and the TruePort CA list.
Solution 1	Make sure the CA lists contains the certificate of the CA which signed the peer's certificate.
Reason 2	The peer's certificate or the CA certificate might have expired. Each certificate is created with a valid date interval.
Solution 2	Make sure the certificate of the peer and CA are up to date. Also verify that the host has the correct date/time. If the date configured on the host is not correct, it can make it look like the certificate is invalid.