Cyclades AlterPath Console Server Reference Guide

A reference guide for users and systems administrators of Cyclades AlterPath Console Server

Product Version 2.2.0
Document Version 1.0

Cyclades AlterPath Console Server - Reference Guide
Version 2.2.0
June, 01st 2004

We believe the information in this manual is accurate and reliable. However, we assume no responsibility, financial or otherwise, for any consequences of the use of this product or manual. This manual is published by Cyclades Corporation, which reserves the right to make improvements or changes in the products described in this manual as well as to revise this publication at any time and without notice to any person of such revision or change. The operating system covered in this manual is v2.2.0. All brand and product names mentioned in this publication are trademarks or registered trademarks of their respective holders.

For latest manual revisions, please refer to Cyclades website on:
http://www.cyclades.com/support/downloads.php

Table of Contents

Chapter 3 - Network 69

Appendix D - Copyrights 285

Glossary 289

Preface

Purpose

The purpose of this Reference Guide is to provide instruction for users to independently install, configure, and maintain the Cyclades AlterPath Console Server . This guide must be used as reference, since all features are divided in a way to ease consulting. For a tour through the Web Interface, please refer to the User's Guide. Whether or not you are a UNIX user, we strongly recommend that you follow the steps given in this manual.

Audience and User Levels

This reference guide is intended for the user who is responsible for the deployment and day-to-day operation and maintenance of the Cyclades AlterPath Console Server .It assumes that the reader understands networking basics and is familiar with the terms and concepts used in Local and Wide Area Networking. UNIX and Linux users will find the configuration process very familiar. It is not necessary to be a UNIX expert, to get the Cyclades AlterPath Console Server up and running. There are two audiences or user levels for this manual:

New Users

These are users new to Linux and/or UNIX with a primarily PC/Microsoft background. You might want to brush up on such things as common Linux/UNIX commands and how to use the vi editor prior to attempting installation and configuration. This essential background information appears in "Appendix A - New User Background Information" on page 245. It is recommended that New Users configure the Cyclades AlterPath Console Server using a Web browser following the User's Guide that is totally based on the Web Interface. However, new users can also configure the Cyclades AlterPath Console Server with vi or the Command Line Interface (CLI).

Power Users

These are UNIX/Linux experts who will use this manual mostly for reference. Power Users can choose between configuring the Cyclades AlterPath Console Server via Web browser, vi, Wizard, or CLI.

How to use this Guide

This guide is organized into the following sections:

- Chapter 1 - Device Access contains the ways to access the serial ports, depending on the protocol you configured for that serial port. This chapter also has information about clustering, menu shell and data buffering.
- Chapter 2 - Authentication provides configuration instructions for different types of authentication available in the Cyclades AlterPath Console Server . This chapter includes detailed information about the Linux-PAM module and Shadow Passwords.
- Chapter 3 - Network all configuration related to network is explained in this chapter. This chapter approaches since basic configuration until the most the most advanced ones such as filters and VPN.
- Chapter 4 - Administration contains system's management, administration and maintenance related features.
- Chapter 5 - AlterPath PM integration involves features for those who have an IPDU being controled by the Cyclades AlterPath Console Server.
- Chapter 6 - PCMCIA Cards Integration this chapter has information about compatible PCMCIA cards and the respective instructions to make them work with the Cyclades AlterPath Console Server.
- Chapter 7 - Profile Configuration approaches the main configuration file of the unit. This chapter explains each parameter of the pslave.conf file. It also has step by step examples for TS, CAS and RAS profiles.
- Chapter 8 - Additional Features and Applications has information about special features and step by step instructions on how to set up them.
- Appendix A - New User Background Information contains information for those who are new to Linux/UNIX.
- Appendix B - Upgrades and Troubleshooting covers the most common problems that users faces when using the Cyclades AlterPath Console Server .
- Appendix C - Cabling and Hardware Information Information has detailed information and pinout diagrams for cables used with the Cyclades AlterPath Console Server .
- Appendix D - Copyrights lists details about applications that were incorporated into the product.
- Glossary - Contains information about specific words and terms used in this manual.

Conventions and Symbols

This section explains the significance of each of the various fonts, formatting, and icons that appear throughout this guide.

Fonts

This guide uses a regular text font for most of the body text and Courier for data that you would input, such as a command line instruction, or data that you would receive back, such as an error message. An example of this would be:

```
# telnet 200.200.200.1 7001
```

Hypertext Links

References to another section of this manual are hypertext links that are underlined (and are also blue in the PDF version of the manual). When you click on them in the PDF version of the manual, you will be taken to that section.

Glossary Entries

Terms that can be found in the glossary are underlined and slightly larger than the rest of the text. These terms have a hypertext link to the glossary.

Quick Steps

Step-by-step instructions for installing and configuring the Cyclades AlterPath Console Server are numbered with a summarized description of the step for quick reference. Underneath the quick step is a more detailed description. Steps are numbered 1, 2, 3, etc.

For example:

Step 1 - Modify the pslave.conf file.

You will modify four Linux files to let the Cyclades AlterPath Console Server know about its local environment. Open the file plsave.conf and add the following lines . . .

Parameter Syntax

This manual uses standard Linux command syntaxes and conventions for the parameters described within it.

Brackets and Hyphens (dashes)

The brackets ([])indicate that the parameter inside them is optional, meaning that the command will be accepted if the parameter is not defined. When the text inside the brackets starts with a dash (-) and/or indicates a list of characters, the parameter can be one of the letters listed within the brackets.

Example:

```
iptables [-ADC] chain rule-specification [options]
```

Ellipses

Ellipses (...) indicate that the latest parameter can be repeated as many times as needed. Usually this is used to describe a list of subjects.

Example:

```
ls [OPTION]...[FILE]...
```

Pipes

The pipe (|) indicates that one of the words separated by this character should be used in the command.

Example:

```
netstat {--statistics|-s} [--tcp|-t] [--udp|-u] [--raw|-w]
```

When a configuration parameter is defined, the Linux command syntax conventions will be also used, with a difference.

Greater-than and Less-than signs

When the text is encapsulated with the “<>” characters, the meaning of the text will be considered, not the literal text. When the text is not encapsulated, the literal text will be considered.

Spacing and Separators

The list of users in the following example must be separated by semicolons (;); the outlets should be separated by commas (,) to indicate a list or with dashes(-) to indicate range; there should not be any spaces between the values.

sXX.pmusers: The user access list. For example: jane:1,2;john:3,4. The format of this field is:

```
[<username>:<outlet list>][;<username>:<outlet list>...]
```

where <outlet list>'s format is:

```
[<outlet number>|<outlet start>-<outlet end>][,<outlet number>|<outlet
start>-<outlet end>]...
```

Cautionary and Instructional Information

Note boxes contain instructional or cautionary information that the reader especially needs to bear in mind. There are three levels of information:

WARNING: *A very important type of tip or warning. Do not ignore this information.*

IMPORTANT: *An important tip that should be read. Review all of these notes for critical information.*

TIP: *An informational tip or tool that explains and/or expedites the use of the product.*

This page has been left intentionally blank.

Chapter 1
Device Access

Introduction

This chapter will introduce all the possible ways to access the serial ports of the Cyclades AlterPath Console Server .From this point is considered that the unit is properly configured using one of the possible profiles (CAS or TS). More information about how to configure a profile can be found on Chapter 7 - Profile Configuration.

1.1 Acessing Serial Ports

There are four ways to access the serial ports, depending on the protocol you configured for that serial port (*all.protocol* being *socket_server* for telnet access, *socket_ssh* for ssh access, etc). One can access the serial port by statically addressing it (using TCP port number, alias name or IP address) or just access the next free serial port available from an existent pool (by using the pool's TCP port number, alias or IP address).

Default Configuration Parameters

These are the default configuration settings:

- DHCP enabled (if there is no DHCP Server, IP for Ethernet is 192.168.160.10 with a Netmask of 255.255.255.0)
- CAS configuration
- socket_server in all ports (access method is telnet)
- 9600 bps, 8N1
- No Authentication

Opening and closing a telnet session to a serial port

To open a telnet session to a serial port or the first free serial port belonging to a pool of serial ports, issue the command:

```
# telnet <CAS hostname> <TCP port number>
```

- <CAS hostname> is the hostname configured in the workstation where the telnet client will run (through /etc/hosts or DNS table). It can also be just the IP address of the Cyclades AlterPath Console Server (Ethernet's interface) configured by the user or learned from DHCP.

- <TCP port number> is the number associated to the serial port or pool of serial ports. From factory, 7001 corresponds to serial port 1, 7002 to serial port 2 and so forth, and 3000 is a pool with all serial ports.

To close the telnet session, just press the telnet hot key configured in telnet client application (usually it's "Ctrl]") and "q" to quit.

Opening and closing an SSH session to a serial port

To open a ssh session to a serial port or the next free serial port from a pool, issue the command:

```
# ssh -l <Username>:<Server> <CAS hostname>
```

- <Username> is the user configured to access that serial port. It is present either in the local CAS database or in a Radius/Tacacs/LDAP/Kerberos, etc database.
- <Server> can be just the TCP port number assigned for that serial port (7001, 7002, etc), pool of ports (3000, etc), the alias for the server connected to that serial port or the alias of a pool of ports.
- <CAS hostname> is the hostname configured in the workstation where the ssh client will run (through /etc/hosts or DNS table). It can also be just the IP address of the Cyclades AlterPath Console Server (Ethernet's interface) configured by the user or learned from DHCP.

To exit the SSH session, press the hot key configured for that ssh client (usually "~.").

Accessing Serial Ports using "ts_menu"

To access the serial port (telnet or ssh) using *ts_menu*, login to the CAS unit and, after receiving the shell prompt, run *ts_menu*. The servers (aliases) or serial ports will be shown as option to start a connection (telnet/ssh). After typing *ts_menu*, you will see something similar to the following:

```
Serial Console Server Connection Menu for your Master Terminal Server

1 ttyS1 2 ttyS2 3 ttyS3 4 ttyS4
5 ttyS5 6 ttyS6 7 ttyS7 8 ttyS8

Type 'q' to quit, a valid option[1-8], or anything else to refresh:
```

How to close the session from ts_menu (from the console of your unit)

To close the session from the ts_menu, follow the steps bellow:

Step 1 - Enter the escape character.

The escape character is shown when you first connect to the port. In character/text Mode, the Escape character is ^]

After entering the escape character, the following is shown:

Console escape. Commands are:

- l go to line mode
- c go to character mode
- z suspend telnet
- b send break
- t toggle binary
- e exit telnet

Step 2 - Press "e" to exit from the session and return to the original menu.

Select the exit option and you will return to the shell prompt.

How to close the session from ts_menu (from a telnet session to your unit)

You have to be sure that a different escape character is used for exiting your telnet/SSH session; otherwise, if you were to exit from the session created through the *ts_menu*, you will close your entire telnet session to your unit. To do this, when you first telnet/SSH to your unit, use the -e option. So for example, to set Ctrl-? as the escape character, type:

```
# telnet -e ^? 192.168.160.10

# ssh -e ^? user1@192.168.160.10
```

To exit from the session created through the *ts_menu*, just follow Step 1 from above. To exit from the entire telnet session to your unit, type the escape character you had set. To exit from the entire SSH session to your unit, type the escape character you had set plus character "."(dot)

1.2 Data Buffering

Data buffering can be done in local files or in remote files through NFS. When using remote files, the limitation is imposed by the remote Server (disk/partition space) and the data is kept in linear (sequential) files in the remote Server. When using local files, the limitation is imposed by the size of the available ramdisk. You may wish to have data buffering done in file, syslog or both. For syslog, *all.syslog_buffering* and *conf.DB_facility* are the parameters to be dealt with, and *syslog-ng.conf* file should be set accordingly. (Please see Syslog-ng for the *syslog-ng* configuration file.) For the file, *all.data_buffering* is the parameter to be dealt with.

Conf.nfs_data_buffering is a remote network file system where data buffering will be written, instead of using the default directory */var/run/DB*. When commented, it indicates local data buffering. The directory tree to which the file will be written must be NFS-mounted and the local path name is */mnt/DB_nfs*. The remote host must have NFS installed and the administrator must create, export, and allow reading/writing to this directory. The size of this file is not limited by the value of the parameter s1.data_buffering,though the value cannot be zero since a zero value turns off data buffering.

The *conf.nfs_data_buffering* parameter format is:

```
<server name or IP address>:<remote pathname>
```

If data buffering is turned on for port 1, for example, the data will be stored in the file ttyS1.data (or &lt;serverfarm1&gt;.data if s1.serverfarm was configured) in local directory */var/run/DB* or in remote path name and server indicated by the *conf.nfs_data_buffering*.

Ramdisks

Data buffering files are created in the directory */var/run/DB*. If the parameter *s<nn>.serverfarm* is configured for the port <nn>, this name will be used. For example, if the serverfarm is called bunny, the data buffering file will be named bunny.data.

Linear vs. Circular Buffering

For local data buffering, this parameter allows users to buffer data in either a circular or linear fashion. Circular format (cir) is a revolving buffer file that is overwritten whenever the limit of the buffer size (set by *all.data_buffering*) is reached. In linear format (lin), data transmission between the remote device and the serial port ceases once the 4k bytes Rx buffer in the kernel is reached. Then if a session is established to the serial port, the data in the buffer is shown (*all.dont_show_DBmenu* or *sxx.dont_show_DBmenu* must be 2), cleared, and data transmission is resumed. Linear buffering is impossible if flow control is set to *none*. Default is cir.

How to Configure

VI mode - Parameters Involved and Passed Values

To configure Data Buffering, follow the steps bellow:

Step 1 - Open the ***/etc/portslave/pslave.conf***** file.**

All parameters related to Data Buffering are in the pslave.conf file. Change the desired parameters according to the table below:

Parameter	Description
all.data_buffering	A non zero value activates data buffering (local or remote, according to what was configured in the parameter conf.nfs_data_buffering). If local data buffering, a file is created on the Cyclades AlterPath Console Server ; if remote, a file is created through NFS in a remote server. All data received from the port is captured in this file. If local data buffering, this parameter means the maximum file size (in bytes). If remote, this parameter is just a flag to activate (greater than zero) or deactivate data buffering. When local data buffering is used, each time the maximum file size is reached the oldest 10% of stored data is discarded, releasing space for new data (FIFO system) - circular file. When remote data buffering is used, there's no maximum file size other than the one imposed by the remote server - linear file. This file can be viewed using the normal UNIX tools (cat, vi, more, tail, etc...). Size is in BYTES not kilobytes.

Table 1.1: Data buffering paremeters in /etc/portslave/pslave.conf file

Parameter	Description
conf.nfs_data_buffering	This is the Remote Network File System where data captured from the serial port will be written instead of being written to the local directory */var/run/ DB*. The directory tree to which the file will be written must be NFS-mounted, so the remote host must have NFS installed and the administrator must have created, exported and allowed reading/writing to this directory. The size of this file is not limited by the value of the parameter *all.data_buffering*, though the value cannot be zero since a zero value turns off data buffering. The size of the file is dependent on the NFS server only (hard drive, partition size, etc.).
all.DB_mode	When configured as cir for circular format, the buffer is like a revolving file that is overwritten whenever the limit of the buffer size (as configured in *all.data_buffering* or *s<n>.data_buffering*) is reached. When configured as *lin* for linear format, once 4k bytes of the Rx buffer in the kernel is reached, a flow control stop (RTS off or XOFF-depending on how all.flow or s<n>.flow is set) is issued to prevent the serial port from receiving further data from the remote. Then when a session is established to the serial port, a flow control start (RTS on or XON) will be issued and data reception will then resume. If *all.flow* or *s<n>.flow* is set to none, linear buffering isn't possible. Default is *cir*.
all.DB_user_logs	When "on", a line containing the time stamp, the username, the event itself (connection/disconnection) and the type of session (Read/Write or Read Only) will be added to the data buffering file every time a user connects or disconnects to the corresponding port. The log message has the following formats : 1) "<connect> [timestamp] [username] [session type] </connect>" 2)"<disconnect> [timestamp] [username] </disconnect>". when [timestamp] = "YYYY-MM-DD hh:mm:ss" [session type] = "Read/Write" or "Read_Only"

Table 1.1: Data buffering paremeters in /etc/portslave/pslave.conf file

Parameter	Description
all.syslog_buffering	When non zero, the contents of the data buffer are sent to the syslog-ng every time a quantity of data equal to this parameter is collected. The syslog level for data buffering is hard coded to level 5 (notice) and facility is local plus *conf.DB_facility*. The file */etc/syslog-ng/syslog-ng.conf* should be set accordingly for the *syslog-ng* to take some action. For more information about it consult "Syslog-ng" on page 119.
all.syslog_sess	This parameter determines whether syslog is generated when a user is connected to the port or not. Originally, syslog is always generated whether the user is connected to the port or not. Now, Cyclades AlterPath Console Server administrators have the option to NOT have syslog generate messages when there is a user connected to a port. This feature does not affect the local data_buffering file. When set to 0 (default), syslog is always generated. When set to 1, syslog is only generated when there are no users connected to the port sending the data. When a user connects to the port that is sending data, syslog messages stop being generated.
all.dont_show_DBmenu	When zero, a menu with data buffering options is shown when a user connects to a port with a non empty data buffering file. When 1, the data buffering menu is not displayed. When 2, the data buffering menu is not shown but the data buffering file is displayed if not empty. When 3, the data buffering menu is shown, but without the erase and show and erase options.
all.DB_timestamp	Records the time stamp in the data buffering file (1) or not (0). If it is configured as 1, the software will accumulate input characters until it receives a CR and LF from the serial port or the accumulated data reaches 256 characters. Either way, the accumulated data will be recorded in the data buffering file along with the current time. The parameter all.data_buffering has to be with a non-zero value for this parameter to be meaningful.

Table 1.1: Data buffering paremeters in /etc/portslave/pslave.conf file

Step 2 - Activating and saving the changes made.

To activate the changes issue the command:

```
# signal_ras hup
```

To save the changes, run the command:

```
# saveconf
```

CLI Method

To configure certain parameters for a specific serial port.

Step 1 - At the command prompt, type in the appropriate command to configure desired parameters.

To activate the serial port. <string> should be ttyS<serial port number> :

```
# config configure line <serial port number> tty <string>
```

To configure nfs_data_buffering:

```
# config configure conf nfsdb <string>
```

To configure data_buffering:

```
# config configure line <serial port number> databuffering <number>
```

To configure DB_mode:

```
# config configure line <serial port number> dbmode <string>
```

To configure dont_show_DBmenu:

```
# config configure line <serial port number> dbmenu <number>
```

To configure DB_timestamp:

```
# config configure line <serial port number> dbtimestamp   <number>
```

To configure syslog_buffering:

```
# config configure line <serial port number> syslogdb <number>
```

TIP: *You can configure all the parameters for a serial port in one line: config configure line <serial port number> tty <string> conf nfsdb <string> db <number> dbmode <string> dbmenu <number> dbtimestamp <number> syslogdb <number>*

1.3 Menu Shell

The menu shell feature allows a user to be presented with a menu in order to connect to a set of hosts as defined by the ACS administrator. It can be used as an easy method for users to access servers on the LAN.

How to use

Once the appropriate configurations are done the user will connect to the ACS using a serial terminal . The user will then automatically receive a menu similar to that shown below:

```
Welcome!

1) Sun server
2) Dell server
3) Linux server
4) Quit

Option ==>
```

The user selects the option required to connect to the desired server or to exit the system.

How to configure

Basically the configuration for this feature is divided in two parts that are going to be described in this section.

Firstly it is necessary to assign which users are going to use the Menu Shell by using the proper options provided by the *menush_cfg* utility. The second part is

Setting up the Menu Shell

Step 1 - Type "*menush_cfg*" and use the options shown below to define the menu title and menu commands.

```
-------------------------------------------------
        MenuShell Configuration Utility
-------------------------------------------------

Please choose from one of the following options:

1. Define Menu Title
2. Add Menu Option
3. Delete Menu Option
4. List Current Menu Settings
5. Save Configuration to Flash
6. Quit

Option ==>
```

Step 2 - Choose the second option (*Add Menu Option*) and complete the requested fields.

The first question will be:

```
Enter the name for the new menu option:
```

Here just fill up with a description for the host that will be acessed.

The second question defines the action that must be taken:

```
Enter the command for the new menu option:
```

Action can be *telnet host_ip* or *ssh -l username host_ip* where host_ip is the ip address of the server to connect to.

Step 3 - Save the changes.

Save the changes made by choosing the fifth option:

```
5. Save Configuration to Flash
```

Assigning ports to the Menu Shell

To configure which ports will prompt the menu shell and if it will require authentication to gain access to it, follow the steps bellow:

Step 1 - If no authentication is required to gain access to the menu.

Configure the following parameters in */etc/portslave/pslave.conf* for the ports that will use this menu shell.

```
s<x>.protocol  telnet
conf.telnet  /bin/menush
s<x>.authtype  none
```

Where <x> is the port number being configured.

Step 2 - If authentication is required to gain access to the menu

The users default shell must be modified to run the */bin/menush*. So in */etc/passwd* the shell should be changed as follows. There should be something like :

```
user:FrE6QU:505:505:Embedix User,,,:/home/user:/bin/menush
```

In *pslave.conf* the port where the serial terminal is attached must be configured for login with authentication local. Configure the following lines:

```
s<x>.protocol login
s<x>.authtype local
```

Where <x> is the port number being configured.

Step 3 - Activating and saving the changes made.

To activate the changes issue the command:

```
# signal_ras hup
```

To save the changes, run the command:

```
# saveconf
```

1.4 Clustering using Ethernet Interface

Clustering is available for the Cyclades AlterPath Console Server with firmware versions 2.1.0 and up. It allows the stringing of Terminal Servers so that one Master Cyclades AlterPath Console Server can be used to access all Cyclades AlterPath Console Server 's on a LAN. The Master Cyclades AlterPath Console Server can manage up to 1024 serial ports, so that the following can be clustered:

- 1 Master ACS48 + 19 Slave ACS48s + 2ACS32, or
- 1 Master ACS16 + 63 Slave ACS16s, or
- 1 Master ACS32 + 31 Slave ACS32s

An example with one Master and two Slave is shown in the following figure.

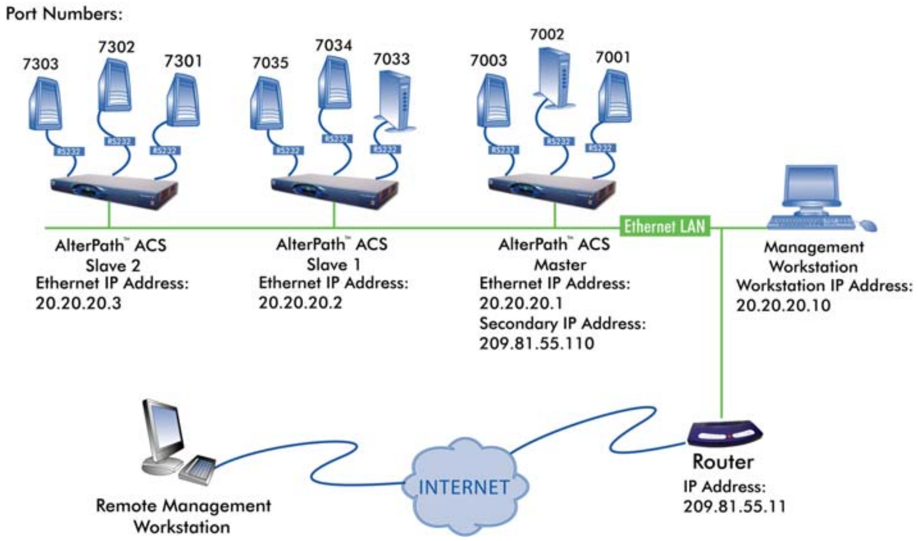


Figure 1.1 - An example using the Clustering feature

How to configure

The Master Cyclades AlterPath Console Server must contain references to the Slave ports. The configuration described for Console Access Servers should be followed with the following exceptions for the Master and Slaves.

VI mode

Step 1 - Edit the */etc/portslave/pslave.conf* **file and change the necessary parameters.**

The related file for clustering configuration is */etc/portslave/pslave.conf*, to edit this file, run the command:

```
# vi /etc/portslave/pslave.conf
```

Follow the explanation provided in the table below:

Parameter	Description	Value for this example
conf.eth_ip	Ethernet Interface IP address.	20.20.20.1
conf.eth_ip_alias	Secondary IP address for the Ethernet Interface (needed for clustering feature).	209.81.55.110
conf.eth_mask_alias	Mask for secondary IP address above.	255.255.255.0
all.socket_port	This value applies to both the local ports and ports on Slave Cyclades AlterPath Console Server .	7001+
all.protocol	Depends on the application.	socket_ssh or socket_server
all.authtype	Depends on the application.	Radius, local, none, remote, TacacsPlus, Ldap, kerberos, local/Radius, radius/local, local/TacacsPlus, TacacsPlus, local, RadiusDownLocal, LdapDownLocal, NIS
s33.tty	This parameter must be created in the Master Cyclades AlterPath Console Server file for every Slave port. Its format is: IP_of_Slave:[slave_socket_port] for non-Master ports. In this case, the slave_socket_port value is not necessary because s33.socket_port is automatically set to 7033 by all.socket_port above.	20.20.20.2:7033
s33.serverfarm	An alias for this port. (This is an optional parameter).	server_on_slave1_ serial_s1

Table 1.2: Master configuration (where it differs from the CAS standard)

Parameter	Description	Value for this example
s33.ipno	This parameter must be created in the Master Cyclades AlterPath Console Server file for every Slave port, unless configured using all.ipno.	0.0.0.0
s34.tty	See s33.tty.	20.20.20.2:7034
s34.serverfarm	An alias for this port.	server_on_slave1_ serial_s2
s34.ipno	See s33.ipno.	0.0.0.0
s35.tty	See s33.tty.	20.20.20.2:7035
s35.serverfarm	An alias for this port.	server_on_slave1_ serial_s3
s35.ipno	See s33.ipno.	0.0.0.0
etc. for s36-s64		
s65.tty	The format of this parameter is IP_of_Slave:[slave_socket_port] for non-Master ports. The value 7301 was chosen arbitrarily for this example.	20.20.20.3:7301
S65.serverfarm	An alias for this port.	server_on_slave2_ serial_s1
S65.ipno	See s33.ipno.	0.0.0.0
S66.tty	See s65.tty.	20.20.20.3:7302
S66.serverfarm	An alias for this port.	server_on_slave2_ serial_s2
S66.ipno	See s33.ipno.	0.0.0.0
S67.tty	See s65.tty.	20.20.20.3:7303
S67.serverfarm	An alias for this port.	server_on_slave2_ serial_s3
S67.ipno	See s33.ipno.	0.0.0.0
etc. for s68-s96		

Table 1.2: Master configuration (where it differs from the CAS standard)

Step 2 - Configure the Cyclades AlterPath Console Server **first slave.**

The Slave Cyclades AlterPath Console Server's do not need to know they are being accessed through the Master Cyclades AlterPath Console Server . (You are creating virtual terminals: virtual serial ports.) Their port numbers, however, must agree with those assigned by the Master. To configure the Slave units, follow the table below:

Parameter	Value for this example
all.protocol	socket_server
all.authtype	none
conf.eth_ip	20.20.20.2
all.socket_port	7033+

Table 1.3: Slave 1 configuration (where it differs from the CAS standard)

Step 3 - Configure the Cyclades AlterPath Console Server **second slave.**

To configure the second slave, follow the parameters of the table below:

Parameter	Value for this example
all.protocol	socket_server
all.authtype	none
conf.eth_ip	20.20.20.3
all.socket_port	7301+

Table 1.4: Slave 2 configuration (where it differs from the CAS standard)

Step 4 - Activating and saving the changes made.

To activate the changes issue the command:

```
# signal_ras hup
```

To save the changes, run the command:

```
# saveconf
```

Step 5 - Acessing the ports.

To access ports from the remote management workstation, use telnet with the secondary IP address.

To access the first port of the Master Cyclades AlterPath Console Server :

```
# telnet 209.81.55.110 7001
```

To access the first port of the Slave1 Cyclades AlterPath Console Server :

```
# telnet 209.81.55.110 7033
```

To access the first port of the Slave2 Cyclades AlterPath Console Server :

```
# telnet 209.81.55.110 7065
```

SSH can also be used from the remote management workstation.

To access the third port of Slave 2:

```
# ssh -l <username>:Server_on_slave2_serial_s3 209.81.55.110
```

To access the fifth port of Slave 2:

```
# ssh -l <username>:7069 209.81.55.110
```

TIP: *It is possible to get the clustering channel tunneled via IPSec. For more information about IPSec, see* VPN Configuration.

1.5 Clustering using NAT (Enhanced)

With Enhanced Clustering, the CAS ports in the slave box can be configured as ssh or telnet and can have any type of authentication available. Authentication is performed in the Slave and not in the Master anymore. Additionally, the Master no longer needs to be the default gateway for all Slave boxes.

Enhanced clustering is available on implementations running Linux 2.4.x versions or newer. This new implementation is based on “iptables/nat” which is only available in these higher versions of Linux. Enhanced Clustering has improved performance and security. Performance is greatly increased because only the NAT translation is performed on the Master box. The Master doesn't open an intermediary TCP connection with the Slave box. Also if ssh encryption and decryption is desired, it is performed on the Slave.

New Parameters and Commands

A new parameter, conf.nat_clustering_ip allows you to enable or disable the clustering via the NAT table. This parameter should be configured with the IP address used to access the serial ports. The NAT clustering will work regardless of the interface where this IP address is assigned to. Additionally, there are two chains (post_nat_cluster and pre_nat_cluster) that holds all rules to perform NAT for clustering.

Table 1.5: Abbreviation List

clustering_ip	IP address of any Cyclades AlterPath Console Server i interface (Master box). It is a public IP address and is the one that must be used to connect with the Slave's serial ports.
master_ip	Primary or secondary ethernet IP address of the Master box (usually a public IP address).
slave_ip	Primary or secondary ethernet IP address of the Slave box (usually a non public IP address).
master_port	Remote serial port parameter "socket_port" (configured in the Master box).
slave_port	Local serial port parameter “socket_port” (configured in the Slave box).

The Master Cyclades AlterPath Console Server box will issue a series of iptables commands to populate the nat table with the necessary rules to perform NAT translation for remote ports. Two chains will be created:

- post_nat_cluster (to change the source IP address)

- pre_nat_cluster (to change the destination IP address)

The Cyclades AlterPath Console Server administrator must enable clustering via NAT in pslave.conf (conf.nat_clustering_ip <clustering_ip>).

```
# iptables -D PREROUTING -t nat -p tcp -j pre_nat_cluster
# iptables -D POSTROUTING -t nat -p tcp -j post_nat_cluster
# iptables -t nat -F post_nat_cluster
# iptables -t nat -F pre_nat_cluster
# iptables -t nat -X pre_nat_cluster
# iptables -t nat -X post_nat_cluster
# iptables -t nat -N pre_nat_cluster
# iptables -t nat -N post_nat_cluster
# iptables -A PREROUTING -t nat -p tcp -j pre_nat_cluster
# iptables -A POSTROUTING -t nat -p tcp -j post_nat_cluster
# iptables -A pre_nat_cluster -t nat -p tcp -d <master_ip> --dport
<master_port> -j DNAT --to <slave_ip>:<slave_port>
.....
# iptables -A post_nat_cluster -t nat -p tcp -d <slave_ip> --dport
<slave_port> -j SNAT --to <master_ip>
.....
```

At any time the Cyclades AlterPath Console Server administrator can issue an iptables command to view, change (at his own risk), or delete the rules in the nat table. If the administrator issues a “fwset restore” command he must also execute the command “signal_ras hup” to recover the nat table.

Cyclades AlterPath Console Server clustering was primarily designed to allow a large number of serial ports (in more than one box) to be accessed using just one single public IP address. It only works for ports configured with the CAS profile. With iptables you can extend the access to the clustering.

Examples:

2. Accessing a Slave box with the WebUI from anywhere:

```
# iptables -A PREROUTING -t nat -p tcp -d 192.168.47.79 --dport 8081 -j DNAT
--to 192.168.51.2:80
```

3. Accessing a public DNS from any Slave box:

```
# iptables -A PREROUTING -t nat -p udp -d 64.186.161.2 --dport 53 -j SNAT
--to 64.186.161.79:53
```

How it works

The Master box (Cyclades AlterPath Console Server) will perform two translation for each packet. The destination IP address is translated in the PREROUTING stage. The source IP address is translated in the POSTROUTING stage.

The command to start a telnet client session has not changed. As before, it looks like this:

```
# telnet <clustering_ip> <master_port>
```

And it will have the same result as the command below issued from a local workstation:

```
# telnet <slave_ip> <slave_port>
```

The command to start an ssh client session must have the following command line option:

```
# -p <master_port>
```

The <master_port> will define at least the Slave box with which a connection is desired.

For example, you may use the following commands:

```
# ssh -l <username1>:<server1> -p 7101 <master_ip>

# ssh -l <username2>:<server2> -p 7101 <master_ip>
```

The above commands will respectively have the same result as the following commands issued from a local workstation:

```
# ssh -l <username1>:<server1> <slave1_ip>

# ssh -l <username2>:<server2> <slave1_ip>
```

If the parameter <master_port> defines the local IP address assigned to the serial port, the command can be simplified:

```
# ssh -l <username1> -p 7101 <master_ip>

# ssh -l <username2> -p 7102 <master_ip>
```

And it will have respectively the same result as the commands below issued from a local workstation:

```
# ssh -l <username1> <slave1_port1_ip>
# ssh -l <username2> <slave2_port1_ip>
```

NOTE: *In the old clustering implementation <username?> and <server?> must be valid in the Master box. In the new clustering they must be valid in the Slave. In the Master box there is no meaning anymore for remote port's serverfarm and authtype parameters. If you wish to access all clustering ports with the ssh command option -p port, you must assign an IP address to the serial port. Do not omit the parameter socket_port in the Master box.*

General Configuration

The configuration of clustering ports is pretty much the same as before. There is only one new parameter in the Master box (conf.nat_clustering_ip) that enables or disables the clustering via NAT. The parameters usernames (if authentication is local) and serverfarm for remote ports must be configured now in the related Slave box.

In the following configuration examples, looking like "s[1-32].tty ttyS[1-32]" must be seen as 32 lines. For example:

```
s1.tty ttyS1
s2.tty ttyS2
...
s32.tty ttyS32
```

Master box Configuration

All mentioned instructions must be made in the */etc/portslave/pslave.conf* file of the Master box

```
#Master box Configuration
#Enable Clustering via NAT
#

conf.nat_clustering_ip 64.186.161.108

#
#Primary ethernet IP address (must be the public IP).
#

conf.eth_ip   64.186.161.108
conf.eth_mask   255.255.255.0

conf.eth_mtu    1500
#
```

File Description 1.2: Master box: /etc/portslave/pslave.conf

```
# Secondary ethernet IP address
#

conf.eth_ip_alias       192.168.170.1
conf.eth_mask_alias 255.255.255.0

#
# Local CAS serial ports (32 socket_ssh ports)
#

all.protocol socket_ssh
all.authtype local
all.socket_port 7001+

s[1-32].tty ttyS[1-32]
#

#Remote CAS serial ports, slave-1 (32 socket_ssh ports). This kind of
#configuration can be used for ssh only; just one entry is necessary.
```

File Description 1.2: Master box: /etc/portslave/pslave.conf

```
s33.tty  192.168.170.2
s33.socket_port  7000

s65.protocol    socket_server
s66.protocol    socket_server
...
s96.protocol    socket_server

#
# Remote CAS serial ports, slave-2 (32 socket_server ports)
#

s65.tty 192.168.170.3:7101
s66.tty 192.168.170.3:7102
....

s96.tty 192.168.170.3:7132

s65.socket_port 8001
s66.socket_port 8002
...
s96.socket_port 8032

#
# Remote CAS serial ports, slave-3 (32 socket_ssh ports)
#

s97.tty 192.168.170.101
s98.tty 192.168.170.102
s99.tty 192.168.170.103
...
```

File Description 1.2: Master box: /etc/portslave/pslave.conf

Slave-1 box Configuration

All mentioned instructions must be made in the */etc/portslave/pslave.conf* file of the first Slave box:

```
#Slave-1 box Configuration

#Primary ethernet IP address
#

conf.eth_ip 192.168.170.2
conf.eth_mask 255.255.255.0
```

File Description 1.3: Slave1 box: /etc/portslave/pslave.conf

```
conf.eth_mtu 1500

#
# Local CAS serial ports (32 socket_ssh ports)
#
all.protocol socket_ssh
all.authtype local

s[1-32].tty ttyS[1-32]
s[1-32].serverfarm slave-1-port[1-32]
```

File Description 1.3: Slave1 box: /etc/portslave/pslave.conf

Slave-2 box Configuration

All mentioned instructions must be made in the */etc/portslave/pslave.conf* file of the second Slave box:

```
#Slave-2 box Configuration
#Primary ethernet IP address
#

conf.eth_ip 192.168.170.3
conf.eth_mask 255.255.255.0
conf.eth_mtu 1500
#
# Local CAS serial ports (32 socket_server ports)
#

all.protocol socket_server
all.authtype local
all.socket_port 7101+

s[1-32].tty ttyS[1-32]
```

File Description 1.4: Slave2 box: /etc/portslave/pslave.conf

Slave-3 box Configuration

All mentioned instructions must be made in the */etc/portslave/pslave.conf* file of the third Slave box:

```
#Slave-3 box Configuration
# Primary ethernet IP address
#
```

File Description 1.5: Slave2 box: /etc/portslave/pslave.conf

```
conf.eth_ip 192.168.170.4
conf.eth_mask 255.255.255.0
conf.eth_mtu 1500

#
# Local CAS serial ports (32 socket_ssh ports)
#
all.protocol socket_ssh
all.authtype local
all.ipno 192.168.170.101+

s[1-32].tty ttyS[1-32]
```

File Description 1.5: Slave2 box: /etc/portslave/pslave.conf

Example of starting CAS session commands

The serverfarm, socket_port, or tty must be provided to select which serial port is to be connected to in the Slave box 1.

```
  # ssh -l <username>:<slave-1-port[1-32]> -p 7000 64.186.161.108
```

The master_port (socket_port in the Master) will select which serial port is to be connected to in the Slave boxes 1 and 2.

```
  # telnet 64.186.161.108 80[01-32]

  # ssh -l <username> -p [7097-7128] 64.186.161.108
```

Chapter 2
Authentication

This chapter presents the procedures for assigning and configuring the authentication service(s) that the Cyclades AlterPath Console Server , system or any of its components and devices will be using. Authentication is the process by which the system, or more specifically, an authentication service such as Kerberos, Ldap or Tacacs, verifies the identity of users (to verify who thay claim to be) as well as to confirm receipt of communication to authorized recipients. This chapter includes the following topics:

- Device Authentication
- Linux-PAM
- Shadow Passwords

2.1 Device Authentication

Authentication is the process of identifying an individual, usually based on a username and password. In security systems, authentication is distinct from authorization, which is the process of giving individuals access to system objects based on their identity. Authentication merely ensures that the individual is who he or she claims to be, but says nothing about the access rights of the individual. With the Cyclades AlterPath Console Server , authentication can be performed locally, or with a remote Radius, Tacacs, ldap database, or kerberos.

How to configure

Follow the steps below to configure authentication type:

VI mode - Parameters involved and passed values

To configure the authenticaton type of the Cyclades AlterPath Console Server , follow the steps below:

Step 1 - Edit the *etc/portslave/pslave.conf* file.

All parameters related to authentication configuration are in the *pslave.conf* file. The table below contains a brief description of each one:

Parameter	Description
all.authtype	Type of authentication used. There are several authentication type options: • none (no authentication) • local (authentication is performed using the */etc/passwd* file) • remote (This is for a terminal profile only. The unit takes in a username but does not use it for authentication. Instead it passes it to the remote server where it is then used for authentication.) • radius (authentication is performed using a Radius authentication server) • TacacsPlus (authentication is performed using a TacacsPlus authentication server)

Table 2.1: Authentication parameters in /etc/portslave/pslave.conf

Parameter	Description
all.authtype (cont.)	• ldap (authentication is performed against an ldap database using an ldap server. The IP address and other details of the ldap server are defined in the file */etc/ldap.conf*) • kerberos (authentication is performed using a kerberos server. The IP address and other details of the kerberos server are defined in the file */etc/krb5.conf*) • local/radius (authentication is performed locally first, switching to Radius if unsuccessful) • radius/local (the opposite of the previous option) • local/TacacsPlus (authentication is performed locally first, switching to TacacsPlus if unsuccessful) • TacacsPlus/local (the opposite of the previous option) • RadiusDownLocal (local authentication is tried only when the Radius server is down) • TacacsPlusDownLocal (local authentication is tried only when the TacacsPlus server is down) • kerberosDownLocal (local authentication is tried only when the kerberos server is down) • LdapDownLocal (local authentication is tried only when the ldap server is down) • NIS - All authentication types but NIS follow the format all.authtype <Authentication>DownLocal or <Authentication> (e.g. all.authtype radius or radiusDownLocal or ldap or ldapDownLocal, etc). NIS requires all.authtype to be set as local, regardless if it will be "nis" or its “Downlocal" equivalent. The service related to "nis" or its “Downlocal" equivalent would be configured in the */etc/nsswitch.conf* file, not in the */etc/portslave/pslave.conf* file. Note that this parameter controls the authentication required by the Cyclades AlterPath Console Server . The authentication required by the device to which the user is connecting is controlled separately.
all.authhost1 all.authhost2	This address indicates the location of the Radius/TacacsPlus authentication server and is only necessary if this option is chosen in the previous parameter. A second Radius/TacacsPlus authentication server can be configured with the parameter all.authhost2.

Table 2.1: Authentication parameters in /etc/portslave/pslave.conf

Parameter	Description
all.accthost1 all.accthost2	This address indicates the location of the Radius/TacacsPlus accounting server, which can be used to track how long users are connected after being authorized by the authentication server. Its use is optional. If this parameter is not used, accounting will not be performed. If the same server is used for authentication and accounting, both parameters must be filled with the same address. A second Radius/TacacsPlus accounting server can be configured with the parameter all.accthost2.
all.radtimeout	This is the timeout (in seconds) for a Radius authentication query to be answered.
all.radretries	Defines the number of times each Radius/TacacsPlus server is tried before another is contacted. The first server (authhost1) is tried "radretries" times, and then the second (authhost2), if configured, is contacted "radretries" times. If the second also fails to respond, Radius/TacacsPlus authentication fails.
all.secret	This is the shared secret (password) necessary for communication between the Cyclades AlterPath Console Server and the Radius/TacacsPlus servers.

Table 2.1: Authentication parameters in /etc/portslave/pslave.conf

NOTE: *If you want to dial in to the serial port on an Cyclades AlterPath Console Server series with CHAP authentication, you need to do the following:*

1. Configure Sxx.authtype as local.
2. Add users in AlterPath Console Server.
3. Insert the users in the file /etc/ppp/chap-secrets.
4. Insert the file /etc/ppp/chap-secrets in the file /etc/config_files.
5. Execute the saveconf command.

Step 2 - Activating and saving the changes made.

To activate the changes issue the command:

```
# signal_ras hup
```

To save the changes, run the command:

```
# saveconf
```

CLI Method

To configure certain parameters for a specific serial port.

Step 1 - At the command prompt, type in the appropriate command to configure desired parameters.

To activate the serial port. <string> should be ttyS<serial port number> :

```
# config configure line <serial port number> tty <string>
```

To configure authtype:

```
# config configure line <serial port number> authtype <string>
```

To configure authhost1:

```
# config configure line <serial port number> authhost1 <string>
```

To configure accthost1:

```
# config configure line <serial port number> accthost1 <string>
```

To configure authhost2:

```
# config configure line <serial port number> authhost2 <string>
```

To configure accthost2:

```
# config configure line <serial port number> accthost2 <string>
```

To configure radtimeout:

```
# config configure line <serial port number> timeout <number>
```

To configure radretries:

```
# config configure line <serial port number> retries <number>
```

To configure secret:

```
# config configure line <serial port number> secret <string>
```

TIP: *You can configure all the parameters for a serial port in one line: config configure line <serial port number> tty <string> authtype <string> authhost1 <string> accthost1 <string> authhost2 <string> accthost2 <string> timeout <number> retries <number> secret <string>*

Step 2 - Activate and Save.

To activate your new configurations and save them to flash, type:

```
# config write
```

Access Control via Radius Attribute NAS-Port-id

This feature provides an additional way to control the access to serial ports other than the one based in usernames or groups. The authentication type must be Radius for this feature to function. The Radius server administrator must configure the user (in the radius server database) with one NAS-PORT-id attribute for each serial port that the user is allowed to access.

In the example below the user alfred can access the serial ports ttyS11, ttyS13, and ttyS17:

```
alfred Auth-Type = Local, Password = ‘alfred’

 Service-Type = Framed-User,
 Framed-Protocol = PPP,
 NAS-Port-Id = 11,
 NAS-Port-Id = 13,
 NAS-Port-Id = 17
```

The pam_radius module will check whether the NAS-Port-Id matches one of those sent by the radius server. If the radius server does not send the NAS-Port-Id attribute, no check is performed.

No configuration is needed for the AlterPath Console Server. However, the authentication type must be “radius”. Authentications like radiusDownLocal, radius/local, etc. will not validate the NAS-port-Id if the user was locally authenticated.

NIS Client

NIS (Network Information System) provides simple and generic client-server database access facilities that can be used to distribute information. This makes the network appear as a single system, with the same accounts on all hosts. The objective of this feature is to allow the administrator to manage Cyclades AlterPath Console Server accounts on a NIS server.

The NIS client feature needs these following files/commands:

File/Command	Description
/etc/yp.conf	This file contains the configuration used by ypbind.
/etc/domainname.conf	This file contains the NIS domain name (set by the command domainname).

Table 2.2: NIS client requirements

File/Command	Description
/usr/sbin/ypbind	Finds the server for NIS domains and maintains the NIS binding information.
/usr/bin/ypwhich	Returns the name of the NIS server that supplies the NIS services.
/usr/bin/ypcat	Prints the values of all keys from the NIS database specified by map name.
/usr/bin/ypmatch	Prints the values of one or more keys from the NIS database specified by map name.
/usr/sbin/domainname	Shell script to read/write the NIS domain name.

Table 2.2: NIS client requirements

NIS Client Configuration

Step 1 - Run the command domainname.

You'll want to make sure that you have the NIS domain name set.

Command :

```
# domainname [NIS domain name]
```

show or set the system's NIS/YP domain name, eg.:

```
# domainname cyclades mycompany-nis
```

Step 2 - Edit the */etc/yp.conf* file.

You will need to configure the NIS server. Example: If the NIS server has the IP address 192.168.160.110, you'll have to add the following line in the file:

```
ypserver 192.168.160.110
```

Step 3 - Edit the */etc/nsswitch.conf* file.

Change the */etc/nsswitch.conf* file ("System Databases and Name service Switch "configuration file) to include the NIS in the lookup order of the databases.

Step 4 - Configure the parameter "<all/sxx>.authype" as "local".

How to Test the Configuration

To test the configuration do the following:

Step 1 - Start up the following command:

```
# /usr/sbin/ypbind
```

Step 2 - Display the NIS server name.

Display the name of NIS server by running the following command:

```
# /usr/bin/ypwhich
```

Step 3 - Display the “all users” entry.

Displays the all users' entry in the NIS database by running the following command:

```
# /usr/bin/ypcat -t passwd.byname
```

Step 4 - Display the user's entry in the NIS passwd file.

```
# /usr/bin/ypmatch -t <userid/username> passwd.byname
```

If the preceding steps were performed successfully, you now need to change the */etc/inittab* file by uncommenting the line that performs a ypbind upon startup.

nsswitch.conf file format

The */etc/nsswitch.conf* file has the following format:

```
<database> : <service> [ <actions> <service> ]
```

where:

<database> - available: aliases, ethers, group, hosts, netgroup, network, passwd, protocols, publickey, rpc, services and shadow

<service> - available: nis (use NIS version 2) , dns (use Domain Name Service) and files (use the local files)

<actions> - Has this format: [<status> = <action>]

where:

<status> = SUCCESS, NOTFOUND, UNAVAIL or TRYAGAIN

<action> = return or continue

- SUCCESS - No error occurred and the desired entry is returned. The default action for this status is 'return'
- NOTFOUND - The lookup process works fine, but the needed value was not found. The default action for this status is “continue.”
- UNAVAIL - The service is permanently unavailable.
- TRYAGAIN - The service is temporarily unavailable.

To use NIS only to authenticate users, you need to change the lines in */etc/nsswitch.conf* that reference passwd, shadow, and group.

Examples

3. You wish to authenticate the user first in the local database. If the user is not found, then use NIS:

```
passwd: files nis
shadow: files nis
group: files nis
```

4. You wish to authenticate the user first using NIS. If the user is not found, then use the local database:

```
passwd: nis files
shadow: nis files
group: nis files
```

5. You wish to authenticate the user first using NIS. If the user was not found or the NIS server is down, then use the local database:

```
passwd: nis [UNAVAIL=continue TRYAGAIN=continue] files
shadow: nis [UNAVAIL=continue TRYAGAIN=continue] files
group: nis [UNAVAIL=continue TRYAGAIN=continue] files
```

2.2 Kerberos Authentication

Kerberos is a computer network authentication protocol designed for use on insecure networks, based on the key distribution model. It allows individuals communicating over a network to prove their identity to each other while also preventing eavesdropping or replay attacks, and provides for detection of modification and the prevention of unauthorized reading

Kerberos Server Authentication with Tickets support

The Cyclades AlterPath Console Server has support to interact on a kerberized network. You can find in the next lines a brief explanation about how kerberos works. Later in this section, a practical a step by step example will be presented.

How Kerberos Works

On a kerberized network, the Kerberos database contains principals and their keys (for users, their keys are derived from their passwords). The Kerberos database also contains keys for all of the network services.

When a user on a kerberized network logs in to their workstation, their *principal* is sent to the Key Distribution Center (*KDC*) as a request for a Ticket Granting Ticket (*TGT*). This request can be sent by the login program (so that it is transparent to the user) or can be sent by the kinit program after the user logs in.

The KDC checks for the *principal* in its database. If the principal is found, the KDC creates a TGT, encrypts it using the user's key, and sends it back to the user.

The login program or *kinit* decrypts the *TGT* using the user's key (which it computes from the user's password). The *TGT*, which is set to expire after a certain period of time, is stored in your credentials cache. An expiration time is set so that a compromised *TGT* can only be used for a certain period of time, usually eight hours (unlike a compromised password, which could be used until changed). The user will not have to re-enter their password until the *TGT* expires or they logout and login again.

When the user needs access to a network service, the client uses the *TGT* to request a ticket for the service from the Ticket Granting Service (*TGS*), which runs on the *KDC*. The *TGS* issues a ticket for the desired service, which is used to authenticate the user.

Configuring ACS to use Kerberos Tickets authentication

For this example we will consider that a kerberos server with ticket support is properly configured in the network. The manual will only approache the Cyclades AlterPath Console Server configuration.

Here we will assume that the kerberos server has the following configuration:

- Principal: john
- Host (Cyclades ACS): acs48-2.cyclades.com

Cyclades AlterPath Console Server Configuration

Configuring it for SSH:

Step 1 - Configure and start a NTP server

All involved parts must be syncronized with a NTP server. To configure a NTP server see "NTP (Network Time Protocol)" on page 81.

Step 2 - Configure authentication and protocol in the *etc/portslave/pslave.conf* file. Open the file and edit these parameters:

```
all.authtype local
all.protocol socket_ssh.
```

Step 3 - Activate and save the configuration, by issuing the commands:

```
# signal_ras hup
# saveconf
```

Step 4 - Add an user with the same name as the "principal", configured in the Kerberos server.

```
# adduser john
```

Step 5 - Configure the *krb5.conf* file. The */etc/krb5.conf* file must be exactly the same as the one that is in the Kerberos server.

It is highly recommended to copy it directly from the server, instead of editing it. To copy using *scp*, issue the command:

```
# scp root@kerberos-server.cyclades.com:/etc/krb5.conf /etc/krb5.conf
```

Step 6 - Extract the host that is in the Kerberos server database to the Cyclades AlterPath Console Server:

```
# kadmin -p admin/admin
```

Where the first "admin" is the service and the second one is the user.

This will prompt a Kerberos server menu. To extract the configured hosts run the following commands in the *kadmin* menu:

```
kadmin: ktadd host/acs48-2.cyclades.com
kadmin: q
```

To list all configured hosts in the Kerberos server, run the command:

```
# klist -k
```

The above command will show all hosts added through the *ktadd* command in the Kerberos server.

Step 7 - Configure hostname and domain name.

To configure the hostname and the domain name, issue the commands:

```
# hostname acs48-2
# domainname cyclades.com
```

RLOGIN E TELNET:

To access the Cyclades AlterPath Console Server through rlogin or telnet, follow the steps described above plus the ones describe below.

Step 1 - Configure the */etc/inetd.conf* file by uncommenting the lines:

```
#KERBEROS SERVICES
klogin stream tcp nowait root /usr/sbin/tcpd /usr/local/sbin/klogind -ki
telnet stream tcp nowait root /usr/sbin/tcpd /usr/local/sbin/telnetd
```

Step 2 - Restart the *inetd* service, by issuing the command:

```
# daemon.sh restart NET
```

Step 3 - Save the configuration.

```
# saveconf
```

Test the configuration:

All the steps below will be performed in the client side:

Step 1 - The client must have a kerberized SSH and configure the */etc/ssh/ssh_config* file, according to the example below:

```
GSSAPIAuthentication yes
GSSAPICleanupCreds yes
```

Step 2 - The client must have the same krb5.conf file present in the Kerberos server.

```
# scp root@kerberos-server.cyclades.com:/etc/krb5.conf /etc/krb5.conf
```

Step 3 - Requesting the ticket to the Kerberos server.

```
# kinit -f -p john
Password for john@CYCLADES.COM: ******
```

You will be prompted to insert a password, that is the "principal" password that is in the Kerberos server database.

Step 4 - Checking if the ticket was successful received:

```
# klist
```

Step 5 - Conect, from the client, to the Cyclades AlterPath Console Server via SSH:

Opening a SSH connection to the Cyclades AlterPath Console Server itself:

```
#ssh john@acs48-2.cyclades.com
```

Opening a SSH session to one of the Cyclades AlterPath Console Server ports:

```
# ssh john:7001@acs48-2.cyclades.com
```

Step 6 - Connecting via RLOGIN to the Cyclades AlterPath Console Server itself, with forwardable tickets (to connect to the Cyclades AlterPath Console Server ports using *ts_menu*):

```
# rlogin -l john acs48-2.cyclades.com -F
```

Then run *ts_menu* to access the desired serial port.

Step 7 - Connecting via TELNET to the Cyclades AlterPath Console Server itself with forwardable tickets (to connect to the Cyclades AlterPath Console Server ports using *ts_menu*):

```
telnet -l john acs48-2.cyclades.com -F
```

Then run *ts_menu* to access the desired serial port.

Kerberos Server Authentication

To authenticate users on a Kerberos sever, it is necessary to edit two configuration files: *pslave.conf* and *krb5.conf*. Below is a step by step example:

Step 1 - Edit the /etc/portslave/pslave.conf file.

Open the */etc/portslave/pslave.conf* file running the following command:

```
# vi /etc/portslave/pslave.conf
```

Look for the *all.authtype* and *all.protocol* parameters and change their values according to the example below:

```
all.authtype     kerberos
all.protocol     socket_ssh ##or socket_server
```

If you are going to use the telnet protocol to access the serial ports of the unit, set the all.protocol parameter to socket_server. In this example we are using the SSH protocol.

Step 2 - Edit the *\/etc/krb5.conf* file.

Open the /etc/krb5.conf running the following command:

```
# vi /etc/krb5.conf
```

Basically, all the changes needed in this file are related to the network domain. Substitute all listed parameters that are configured with "cyclades.com" with the correspondent domain of your network. Below is an example of the file:

```
[logging]
default = FILE:/var/log/krb5libs.log
kdc = FILE:/var/log/krb5kdc.log
admin_server = FILE:/var/log/kadmind.log


[libdefaults]
ticket_lifetime = 24000
default_realm = CYCLADES.COM
default_tgs_enctypes = des-cbc-crc
default_tkt_enctypes = des-cbc-crc

[realms]

CYCLADES.COM = {
kdc = kerberos.cyclades.com:88
admin_server = kerberos.cyclades.com:749
default_domain = cyclades.com
}

[domain_realm]
.cyclades.com = CYCLADES.COM
cyclades.com = CYCLADES.COM
```

File Description 2.1: /etc/krb5.conf

```
[kdc]
profile = /var/kerberos/krb5kdc/kdc.conf

[pam]
     debug = false
     ticket_lifetime = 36000
     renew_lifetime = 36000
     forwardable = true
     krb4_convert = false
```

File Description 2.1: /etc/krb5.conf

Step 3 - Activating changes.

To activate the changes made, run the command:

```
# signal_ras hup
```

Step 4 - Testing configuration

To test the configuration, it is necessary to access any serial port using the telnet protocol (this is the case of the approached e.g.). From a remote machine issue the following command:

```
# telnet 192.168.0.1 7001
```

A prompt will ask for an user and password. Log with the user and password previously configured in the Kerberos server.

In the AlterPath ACS run the command:

```
# w
```

The response for this command will be something like this:

```
1:03pm  up 57 min,  1 user,  load average: 0.00, 0.00, 0.00

USER     TTY    FROM   LOGIN@   IDLE   JCPU   PCPU   WHAT
root    ttyS0    -     12:07pm  0.00s  1.47s  0.15s  /bin/sh /usr/bin



CAS users :  1

USER       TTY     FROM                 LOGIN@     PID/Command
cyclades   ttyS1    192.168.0.143:1503    01:02pm    512/-RW_srv ttyS
```

The last line of the command response shows the user “cyclades” acessing the first serial port of the AlterPath ACS unit.

Step 5 - Saving changes.

To save the configuration, run the command:

```
# saveconf
```

2.3 Linux-PAM

Linux-PAM (Pluggable Authentication Modules for Linux) is a suite of shared libraries that enable the local system administrator to choose how applications authenticate users. In other words, without (rewriting and) recompiling a PAM-aware application, it is possible to switch between the authentication mechanism(s) it uses. Indeed, one may entirely upgrade the local authentication system without touching the applications themselves.

It is the purpose of the Linux-PAM project to separate the development of privilege-granting software from the development of secure and appropriate authentication schemes. This is accomplished by providing a library of functions that an application may use to request that a user be authenticated. This PAM library is configured locally with a system file, */etc/pam.conf* (or a series of configuration files located in */etc*) to authenticate a user request via the locally available authentication modules. The modules themselves will usually be located in the directory */lib/security* and take the form of dynamically loadable object files.

The Linux-PAM authentication mechanism gives to the system administrator the freedom to stipulate which authentication scheme is to be used. S/he has the freedom to set the scheme for any/all PAM-aware applications on your Linux system. That is, s/he can authenticate from anything as generous as simple trust (pam_permit) to something as severe as a combination of a retinal scan, a voice print and a one-time password!

Linux-PAM deals with four separate types of (management) task. These are: authentication management, account management, session management, and password management. The association of the preferred management scheme with the behavior of an application is made with entries in the relevant Linux-PAM configuration file. The management functions are performed by modules specified in the configuration file.

Following is a figure that describes the overall organization of Linux-PAM:

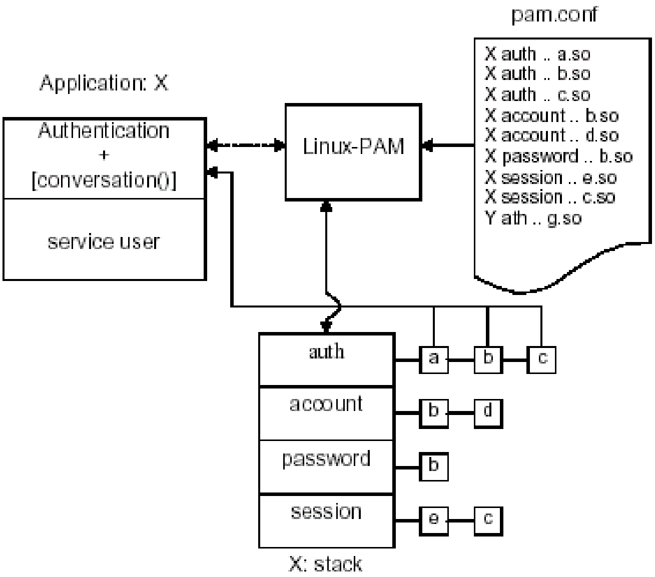


Figure 2.1 - Data flow diagram of Linux-PAM

The left of the figure represents the application: Application X. Such an application interfaces with the Linux-PAM library and knows none of the specifics of its configured authentication method. The Linux-PAM library (in the center) consults the contents of the PAM configuration file and loads the modules that are appropriate for Application X. These modules fall into one of four management groups (lower center) and are stacked in the order they appear in the configuration file. These modules, when called by Linux-PAM, perform the various authentication tasks for the application. Textual information, required from or offered to the user can be exchanged through the use of the application-supplied conversation function.

The Linux-PAM Configuration File

Linux-PAM is designed to provide the system administrator with a great deal of flexibility in configuring the privilege-granting applications of their system. The local configuration of those aspects of system security controlled by Linux-PAM is contained in a single system file */etc/pam.conf*. In this section we discuss the correct syntax of and generic options respected by entries to these files.

Configuration File Syntax

The reader should note that the Linux-PAM-specific tokens in this file are case-insensitive. The module paths, however, are case-sensitive since they indicate a file's name and reflect the case-dependence of typical Linux file systems. The case-sensitivity of the arguments to any given module is defined for each module in turn.

In addition to the lines described below, there are two special characters provided for the convenience of the system administrator:

#	Comments are preceded by this character and extend to the next end-of-line.
\	This character extends the configuration lines.

A general configuration line of the */etc/pam.conf* file has the following form:

```
Service-name module-type control-flag module-path arguments
```

The meaning of each of these tokens is explained below. After the meaning of the above tokens is explained, the method will be described.

Token	Description
Service-name	The name of the service associated with this entry. Frequently the service name is the conventional name of the given application. For example, 'ftpd', 'rlogind', 'su', etc. There is a special service-name, reserved for defining a default authentication mechanism. It has the name 'OTHER' and may be specified in either lower or upper case characters. Note, when there is a module specified for a named service, the 'OTHER' entries are ignored.
Module-type	One of (currently) the four types of module. The four types are as follows: • *Auth* - This module type provides two aspects of authenticating the user. First, it establishes that the user is who they claim to be, by instructing the application to prompt the user for a password or other means of identification. Second, the module can grant group membership, independently of the */etc/groups*, or other privileges through its credential-granting properties.

Table 2.3: /etc/pam.conf tokens description

Token	Description
Module-type (cont.)	• *Account* - This module performs non-authentication-based account management. It is typically used to restrict or permit access to a service based on the time of day, currently available system resources (maximum number of users) or perhaps the location of the applicant user—'root' login only on the console. • *Session* - Primarily, this module is associated with doing things that need to be done for the user before or after they can be given service. Such things include the logging of information concerning the opening or closing of some data exchange with a user, mounting directories, etc. • *Password* - This last module type is required for updating the authentication token associated with the user. Typically, there is one module for each 'challenge/response' based authentication (auth) module-type.
Control-flag	The control-flag is used to indicate how the PAM library will react to the success or failure of the module it is associated with. Since modules can be stacked (modules of the same type execute in series, one after another), the control-flags determine the relative importance of each module. The application is not made aware of the individual success or failure of modules listed in the '*/etc/pam.conf*' file. Instead, it receives a summary of success or fail responses from the Linux-PAM library. The order of execution of these modules is that of the entries in the */etc/pam.conf* file: earlier entries are executed before later ones. The control-flag can be defined with one of two syntaxes. The simpler (and historical) syntax for the control-flag is a single keyword defined to indicate the severity of concern associated with the success or failure of a specific module. There are four such keywords: *required*, *requisite*, *sufficient* and *optional*.

Table 2.3: /etc/pam.conf tokens description

The Linux-PAM library interprets these keywords in the following manner:

Keyword	Description
Required	This indicates that the success of the module is required for the module-type facility to succeed. Failure of this module will not be apparent to the user until all of the remaining modules (of the same module-type) have been executed.
Requisite	This is similar to required. However, in the case that such a module returns a failure, control is directly returned to the application. The return value is that associated with the first required or requisite module to fail. Note that this flag can be used to protect against the possibility of a user getting the opportunity to enter a password over an unsafe medium. It is conceivable that such behavior might inform an attacker of valid accounts on a system. This possibility should be weighed against the significant concerns of exposing a sensitive password in a hostile environment.
Sufficient	The success of this module is deemed 'sufficient' to satisfy the Linux-PAM library that this moduletype has succeeded in its purpose. In the event that no previous required module has failed, no more 'stacked' modules of this type are invoked. (Note: in this case subsequent required modules are not invoked.) A failure of this module is not deemed as fatal to satisfying the application.
Optional	As its name suggests, this control-flag marks the module as not being critical to the success or failure of the user's application for service. In general, Linux-PAM ignores such a module when determining if the module stack will succeed or fail. However, in the absence of any definite successes or failures of previous or subsequent stacked modules this module will determine the nature of the response to the application. One example of this latter case is when the other modules return something like PAM_IGNORE.

Table 2.4: /etc/pam.conf keywords description

Module Path

Module Path is the path-name of the dynamically loadable object file--the pluggable module itself. If the first character of the module path is '/', it is assumed to be a complete path. If this is not the case, the given module path is appended to the default module path: */lib/security.*

Currently, the Cyclades AlterPath Console Server has the following modules available:

Module Name	Description
pam_access	Provides logdaemon style login access control.
pam_deny	Deny access to all users.
pam_env	This module allows the (un)setting of environment variables. The use of previously set environment variables as well as PAM_ITEMs such as PAM_RHOST is supported.
pam_filter	This module was written to offer a plug-in alternative to programs like ttysnoop. Since a filter that performs this function has not been written, it is currently only a toy. The single filter provided with the module simply transposes upper and lower case letters in the input and output streams. (This can be very annoying and is not kind to termcap-based editors.)
pam_group	This module provides group settings based on the user's name and the terminal they are requesting a given service from. It takes note of the time of day.
pam_issue	This module presents the issue file (*/etc/issue* by default) when prompting for a username.
pam_lastlog	This session module maintains the */var/log/lastlog* file. It adds an open entry when called via the pam_open_session()function and completes it when pam_close_session() is called. This module can also display a line of information about the last login of the user. If an application already performs these tasks, it is not necessary to use this module.
pam_limits	This module, through the Linux-PAM open-session hook, sets limits on the system resources that can be obtained in a user session. Its actions are dictated more explicitly through the configuration file discussed in */etc/security/pam_limits.conf*.
pam_listfile	The listfile module provides a way to deny or allow services based on an arbitrary file.
pam_motd	This module outputs the motd file (*/etc/motd* by default) upon successful login.
pam_nologin	Provides standard Unix nologin authentication.

Table 2.5: Available PAM modules in the Cyclades AlterPath Console Server

Module Name	Description
pam_permit	This module should be used with extreme caution. Its action is to always permit access. It does nothing else.
pam_radius	Provides Radius server authentication and accounting.
pam_rootok	This module is for use in situations where the superuser wishes to gain access to a service without having to enter a password.
pam_securetty	Provides standard UNIX securetty checking.
pam_time	Running a well-regulated system occasionally involves restricting access to certain services in a selective manner. This module offers some time control for access to services offered by a system. Its actions are determined with a configuration file. This module can be configured to deny access to (individual) users based on their name, the time of day, the day of week, the service they are applying for and their terminal from which they are making their request.
pam_tacplus	Provides TacacsPlus Server authentication, authorization (account management), and accounting (session management).
pam_unix	This is the standard UNIX authentication module. It uses standard calls from the system's libraries to retrieve and set account information as well as authentication. Usually this is obtained from the */etc/passwd* and the */etc/shadow* file as well when shadow is enabled.
pam_warn	This module is principally for logging information about a proposed authentication or application to update a password.
pam_krb5	The Kerberos module currently used is pam_krb5. This PAM module requires the MIT 1.1+ release of Kerberos, or the Cygnus CNS distribution. It has not been tested against heimdal or any other Kerberos distributions. Important file: /etc/krb5.conf. The krb5.conf file contains Kerberos configuration information, including the locations of KDCs and admin servers for the Kerberos realms of interest, defaults for the current realm and for Kerberos applications, and mappings of hostnames onto Kerberos realms. Normally, you should install your krb5.conf file in the directory/etc. You can override the default location by setting the environment variable KRB5_CONFIG.

Table 2.5: Available PAM modules in the Cyclades AlterPath Console Server

Module Name	Description

pam_ldap	Pam_ldap looks for the ldap client configuration file “ldap.conf” in /etc/. Here's an example of the ldap.conf file (partial):
	`# file name: ldap.conf`
	`# This is the configuration file for the LDAP`
	`# nameservice`
	`# switch library and the LDAP PAM module.`
	`#`
	`# Your LDAP server. Must be resolvable without using`
	`# LDAP.`
	`host 127.0.0.1`
	`# The distinguished name of the search base.`
	`base dc=padl,dc=com`

Table 2.5: Available PAM modules in the Cyclades AlterPath Console Server

Arguments

The arguments are a list of tokens that are passed to the module when it is invoked. They are much like arguments to a typical Linux shell command. Generally, valid arguments are optional and are specific to any given module. Invalid arguments are ignored by a module, however, when encountering an invalid argument, the module is required to write an error to syslog(3).

The following are optional arguments which are likely to be understood by any module. Arguments (including these) are in general, optional.

Arguments	Description
debug	Use the syslog(3) call to log debugging information to the system log files.
no_warn	Instruct module to not give warning messages to the application.
use_first_pass	The module should not prompt the user for a password. Instead, it should obtain the previously typed password (from the preceding auth module), and use that. If that doesn’t work, then the user will not be authenticated. (This option is intended for auth and password modules only).

Table 2.6: List of valid arguments to PAM

Arguments	Description
try_first_pass	The module should attempt authentication with the previously typed password (from the preceding auth module). If that doesn't work, then the user is prompted for a password. (This option is intended for auth modules only).
use_mapped_ pass	This argument is not currently supported by any of the modules in the Linux-PAM distribution because of possible consequences associated with U.S. encryption exporting restrictions.
expose_account	In general, the leakage of some information about user accounts is not a secure policy for modules to adopt. Sometimes information such as user names or home directories, or preferred shell, can be used to attack a user's account. In some circumstances, however, this sort of information is not deemed a threat: displaying a user's full name when asking them for a password in a secured environment could- also be called being 'friendly'. The expose_account argument is a standard module argument to encourage a module to be less discrete about account information as deemed appropriate by the local administrator. Any line in (one of) the configuration file(s), that is not formatted correctly will generally tend (erring on the side of caution) to make the authentication process fail. A corresponding error is written to the system log files with a call to syslog(3).

Table 2.6: List of valid arguments to PAM

PAM support for LDAP Authentication

Here we are going to describe the basic steps to configure a LDAP server on linux. We will also give instructions about how to configure the Cyclades AlterPath Console Server box.

How to configure a LDAP server on Linux

The steps below are intended to guide the installation of a LDAP server on a generic Linux machine.

Step 1 - The packages required for the LDAP servers are:

- db (Sleepycat Berkeley Database)
- openssl (OpenSSL)
- openldap (OpenLDAP)

It's possible also to load the source codes and compile them, but it is easier to load these packages from your distribution CD-ROM or via Internet.

Step 2 - Go to the directory */etc/openldap* or */usr/local/etc/openldap*.

Change the directory running the following command:

```
# cd /usr/local/etc/openldap
```

NOTE: *The example uses /usr/local path. Change all references of /usr/local if the path is different, and check if the directory/file really exists.*

Step 3 - Create the certificates:

To create the certificates, run the following commands in the given sequence:

```
# ln -s /usr/local/bin/openssl .
# ln -s /usr/local/ssl/misc/CA.pl .
# PATH=$PATH:.
# CA.pl -newca <-- anwer questions, you MUST fill in "commonName"
# CA.pl -newreq <-- repeat
# CA.pl -signreq
# mv newreq.pem ldapkey.pem
# chmod 0600 ldapkey.pem
# mv newcert.pem ldapcert.pem
```

Step 4 - Edit *slapd.conf*. The basic configuration to make it work is:

The basic configuration of the file is like described below:

```
include /usr/local/etc/openldap/schema/core.schema
include /usr/local/etc/openldap/schema/cosine.schema

pidfile /usr/local/var/slapd.pid
argsfile /usr/local/var/slapd.args

TLSCipherSuite HIGH:MEDIUM:+SSLv2
TLSCertificateFile /usr/local/etc/openldap/ldapcert.pem
TLSCertificateKeyFile /usr/local/etc/openldap/ldapkey.pem
TLSCACertificateFile /usr/local/etc/openldap/demoCA/cacert.pem

database bdb
suffix "dc=cyclades,dc=com,dc=br"

rootdn "cn=admin,dc=cyclades,dc=com,dc=br"

rootpw bitadmin

directory /usr/local/var/openldap-data

index objectClass eq
```

File Description 2.2: slapd.conf configuration

Step 5 - Start LDAP server.

To start the server run the command:

```
# /usr/local/libexec/slapd -h "ldap:/// ldaps:///"
```

This will allow the LDAP server accept both secured mode and non-secure mode.

Step 6 - Add entries.

Example:

```
ldapadd -x -D "cn=admin,dc=cyclades,dc=com,dc=br" -w bitadmin
dn: uid=helio,dc=cyclades,dc=com,dc=br
objectClass: person
objectClass: uidobject
uid: helio
cn: Helio Fujimoto
sn: Fujimoto
userPassword: bithelio
```

To list the entries:

```
ldapsearch -x -D "cn=admin,dc=cyclades,dc=com,dc=br" -w bitadmin
'(objectClass=*)'
```

This is enough to set up a LDAP server with some users, for PAM authentication purposes.

Configuring the Cyclades AlterPath Console Server To configure the unit for PAM authentication, follow the steps below:

Step 1 - Configure *all.protocol* as ldap, in */etc/portslave/pslave.conf*

Step 2 - Configure the */etc/ldap.conf* file.

Edit the following parameters:

```
host 200.246.93.95 <== LDAP server IP address or name

base dc=cyclades,dc=com,dc=br <== distinguished name of the search base

uri ldaps://200.246.93.95 <== to use secure LDAP
```

File Description 2.3: /etc/ldap.conf configuration

Step 3 - Activating and saving the changes made.

To activate the changes issue the command:

```
# signal_ras hup
```

To save the changes, run the command:

```
# saveconf
```

For Active Directory

A Windows 2000 or Windows 2003 Server edition is necessary. In the Cyclades AlterPath Console Server side, the */etc/ldap.conf* must be configured.

What needs to be set in the */etc/ldap.conf*

Follow the example below to set correctly the necessary parameters:

```
# The Windows 2003 server IP address
host 200.246.93.118

# The Distinguished name (In our active directory, the format was set
# to Cycladescorporation.local)
base dc=CycladesCorporation,dc=local

# Here you can insert any user you had created, or the administrator
# user.
binddn cn=Administrator,cn=Users,dc=Cyclades,dc=local

# Password for that user
bindpw test123

# PAM login attribute
pam_login_attribute sAMAccountName

# Update Active Directory password, by creating Unicode password and
# updating unicodePwd attribute.

pam_password ad
```

File Description 2.4: /etc/ldap.conf

Secure Default Policy

If a system is to be considered secure, it had better have a reasonably secure 'OTHER' entry. The following is a "severe" setting (which is not a bad place to start!):

```
#
# default; deny access
#
OTHER auth required pam_deny.so
OTHER account required pam_deny.so
OTHER password required pam_deny.so
OTHER session required pam_deny.so
```

File Description 2.5: part of the /etc/pam.conf file

While fundamentally a secure default, this is not very sympathetic to a misconfigured system. For example, such a system is vulnerable to locking everyone out should the rest of the file become badly written.

The module *pam_deny* is not very sophisticated. For example, it logs no information when it is invoked, so unless the users of a system contact the administrator when failing to execute a service application, the administrator may not know for a long while that his system is misconfigured.

The addition of the following line before those in the above example would provide a suitable warning to the administrator.

```
#
# default; This application is not configured
#
OTHER auth required pam_warn.so
OTHER password required pam_warn.so
```

File Description 2.6: part of the /etc/pam.conf file

On a less sensitive computer, the following selection of lines (in */etc/pam.conf*) is likely to mimic the historically familiar Linux setup:,

```
#
# default; standard UNIX access
#
OTHER auth required pam_unix_auth.so
OTHER account required pam_unix_acct.so
OTHER password required pam_unix_passwd.so
OTHER session required pam_unix_session.so
```

File Description 2.7: part of the /etc/pam.conf file

In general this will provide a starting place for most applications.

In addition to the normal applications: *login*, *su*, *sshd*, *passwd*, and *pppd*. Cyclades also has made portslave a PAM-aware application. The portslave requires four services configured in *pam.conf*. They are local, remote, radius, and tacplus. The portslave PAM interface takes any parameter needed to perform the authentication in the serial ports from the file *pslave.conf*. The *pslave.conf* parameter *all.authtype* determines which service(s) should be used.

```
# /etc/pam.conf
#
# Last modified by Andrew G. Morgan <morgan@kernel.org>
# -------------------------------------------------------------------
# $Id: pam.conf,v 1.13 2003/11/17 16:15:04 edson Exp $
# -------------------------------------------------------------------
#
# serv.    module   ctrl      module [path]...[args..]
#
# name     type   flag
#
# -------------------------------------------------------------------

# WARNING. The services tacacs, s_tacacs, radius, s_radius, local,
# s_local, and remote are used by the applications portslave,
# socket_server, socket_ssh, and raw_data and should not be changed by
# the administrators unless he knows what he is doing.
# The PAM configuration file for the `kerberos' service
#
kerberos   auth     required    pam_krb5.so no_ccache
kerberos   auth     optional    pam_auth_srv.so
kerberos   account  required    pam_krb5.so no_ccache
kerberos   session  required    pam_krb5.so no_ccache
#
```

File Description 2.8: /etc/pam.conf complete file example

```
# The PAM configuration file for the `kerberosdownlocal' service
# If Kerberos server is down, uses the local service
#
kerberosdownlocal auth  requisite  pam_securetty.so
kerberosdownlocal auth  optionalpam_auth_srv.so
kerberosdownlocal auth\
[success=done new_authtok_reqd=done authinfo_unavail=ignore default=die]\
     pam_krb5.so no_ccache
kerberosdownlocal auth   required   pam_unix2.so
kerberosdownlocal account \
[success=done new_authtok_reqd=done authinfo_unavail=ignore default=die]\
     pam_krb5.so no_ccache
kerberosdownlocal account requiredpam_unix2.so
kerberosdownlocal session \
[success=done new_authtok_reqd=done authinfo_unavail=ignore default=die]\
     pam_krb5.so no_ccache
kerberosdownlocal session requiredpam_unix2.so
#
# The PAM configuration file for the `ldap' service
#
ldap    auth    sufficient   pam_ldap.so

ldap    account required    pam_ldap.so

ldap    session required    pam_ldap.so
#
# The PAM configuration file for the `ldapdownlocal' service
# If LDAP server is down, uses the local service
#
ldapdownlocal auth\
[success=done new_authtok_reqd=done authinfo_unavail=ignore default=die]\
        pam_ldap.so
ldapdownlocal auth  requiredpam_unix2.so
ldapdownlocal account \
[success=done new_authtok_reqd=done authinfo_unavail=ignore default=die]\
        pam_ldap.so
ldapdownlocal account requiredpam_unix2.so
ldapdownlocal session \
[success=done new_authtok_reqd=done authinfo_unavail=ignore default=die]\
        pam_ldap.so

ldapdownlocal    session  required    pam_unix2.so
```

File Description 2.8: /etc/pam.conf complete file example

```
#
# The PAM configuration file for the `tacplus' service
#
tacplus auth     requisite   pam_securetty.so
tacplus auth     required    pam_tacplus.so encrypt
tacplus auth     optional    pam_auth_srv.so
tacplus account    required     pam_tacplus.so encrypt service=ppp protocol=lcp
tacplus session    required     pam_tacplus.so encrypt service=ppp protocol=lcp

s_tacplus auth     requisite  pam_securetty.so
s_tacplus auth     required   pam_tacplus.so encrypt use_first_pass
s_tacplus account  required   pam_tacplus.so encrypt service=ppp protocol=lcp
s_tacplus session  required   pam_tacplus.so encrypt service=ppp protocol=lcp
#
# The PAM configuration file for the `radius' service
#
radius auth         requisite  pam_securetty.so
radius auth         required   pam_radius_auth.so
radius auth         optional   pam_auth_srv.so
radius account      required   pam_radius_auth.so
radius session      required   pam_radius_auth.so

s_radius auth       requisite  pam_securetty.so
s_radius auth       required   pam_radius_auth.so use_first_pass
s_radius account    required   pam_radius_auth.so
s_radius session    required   pam_radius_auth.so


#
# The PAM configuration file for the `local' service
#
local auth          requisite  pam_securetty.so
local auth          required   pam_unix2.so
local account       required   pam_unix2.so
local password      required   pam_unix2.so md5 use_authtok
local session       required   pam_unix2.so

s_local auth        requisite  pam_securetty.so
s_local auth        required   pam_unix2.so use_first_pass
s_local account     required   pam_unix2.so
s_local password    required   pam_unix2.so md5 use_authtok
s_local session     required   pam_unix2.so
```

File Description 2.8: /etc/pam.conf complete file example

```
#
# The PAM configuration file for the `remote' service
#
remote        auth       required  pam_permit.so
remote        account    required  pam_permit.so
remote        password   required  pam_permit.so
remote        session    required  pam_permit.so

#
# The PAM configuration file for the `login' service
#
login   auth       requisite  pam_securetty.so
login   auth       required   pam_unix2.so
login   auth       optional   pam_group.so
login   account    requisite  pam_time.so
login   account    required   pam_unix2.so
login   password   required   pam_unix2.so md5 use_authtok
login   session    required   pam_unix2.so
login   session    required   pam_limits.so

#
# The PAM configuration file for the `xsh' service
#
sshd    auth        required    pam_unix2.so
sshd    auth        optional    pam_group.so
sshd    account     requisite   pam_time.so
sshd    account     required    pam_unix2.so
sshd    password    required    pam_unix2.so md5 use_authtok
sshd    session     required    pam_unix2.so
sshd    session     required    pam_limits.so

#
# The PAM configuration file for the `passwd' service
#
passwdpassword   required   pam_unix2.so md5
#
# The PAM configuration file for the `samba' service
#
samba   auth     required    pam_unix2.so
samba   account  required    pam_unix2.so
```

File Description 2.8: /etc/pam.conf complete file example

```
#
# The PAM configuration file for the `su' service
#
su   auth      required       pam_wheel.so
su   auth      sufficient     pam_rootok.so
su   auth      required       pam_unix2.so
su   account   required       pam_unix2.so
su   session   required       pam_unix2.so


#
# Information for the PPPD process with the 'login' option.
#
ppp     auth     required    pam_nologin.so
ppp     auth     required    pam_unix2.so
ppp     account  required    pam_unix2.so
ppp     session  required    pam_unix2.so


#
#Information for the ipppd process with the 'login' option: local authent.
#
ippp    auth    required    pam_nologin.so
ippp    auth    required    pam_unix2.so
ippp    account required    pam_unix2.so
ippp    session required    pam_unix2.so


#Information for the ipppd process with the 'login' option: radius authent.
#
#ippp auth required   pam_radius_auth.so conf=/etc/raddb/server
#ippp auth optional   pam_auth_srv.so
#ippp account required   pam_radius_auth.so conf=/etc/raddb/server
#ippp session required   pam_radius_auth.so conf=/etc/raddb/server


# The PAM configuration file for the `other' service
#
other   auth        required    pam_warn.so
other   auth        required    pam_deny.so
other   account     required    pam_deny.so
other   password    required    pam_warn.so
other   password    required    pam_deny.so
other   session     required    pam_deny.so
```

File Description 2.8: /etc/pam.conf complete file example

Reference

The Linux-PAM System Administrators' Guide
Copyright (c) Andrew G. Morgan 1996-9. All rights reserved.
Email: morgan@linux.kernel.org

2.4 Shadow Passwords

The default */etc/passwd* file has the user “root” with password “tslinux”. You should change the password for user root as soon as possible. Before changing any password or adding new users you can also activate shadow passwords, if it is needed. The Cyclades AlterPath Console Server has support for shadow password, but it is not active by default. To activate shadow password follow the steps listed below:

Step 1 - Create an empty file called *\/etc/shadow*.

To create this file, run the command:

```
# touch /etc/shadow
```

Step 2 - Add a temporary user to the system.

This temporary user will be removed later. Run the command:

```
# adduser boo
```

Step 3 - Edit the file /etc/shadow.

For each user in passwd file, create a copy of the line that begins with “boo:” in the shadow file, then replace “boo” with the user name. The line beginning with “root” must be the first line in the file */etc/shadow*. The following lines show how the */etc/shadow* file should be.

```
root:EreFjH95c1x6Y:12408:0:99999:7:-1:-1:
rpc:EreFjH95c1x6Y:12408:0:99999:7:-1:-1:
nobody:EreFjH95c1x6Y:12408:0:99999:7:-1:-1:
sshd:EreFjH95c1x6Y:12408:0:99999:7:-1:-1:
boo:EreFjH95c1x6Y:12408:0:99999:7:-1:-1:
```

File Description 2.9: /etc/shadow

Step 4 - Edit the passwd file.

Replace the password in all password fields with an “x”. The root’s line will look like this:

```
“root:x:0:0:root:/root:/bin/sh”
      ^
      ^  password field
```

The */etc/passwd* file should look like this:

```
root:x:0:0:root:/root:/bin/sh
rpc:x:1:1:Portmapper RPC user:/:/bin/false
nobody:x:99:99:Nobody:/:
sshd:x:501:501:sshd privsep:/var/empty:/bin/false
boo:x:505:505:Embedix User,,,:/home/boo:/bin/sh
```

File Description 2.10: /etc/shadow

TIP: *Using the vi editor, put the cursor in the first byte after “root:”, then type “ct:x” plus <ESC>.*

Step 5 - Remove the temporary user boo.

For the this purpose, run the command:

```
# deluser boo
```

Step 6 - Change the password for all users and add the new ones needed.

To change user’s passwords, issue the command:

```
# passwd <username>
```

or

```
# adduser <username>
```

Step 7 - Edit the ***/etc/config_files***** file and add a line with “*****/etc/shadow*****”.**

Step 8 - Save the configuration.

To save the configuration in the flash memory, run the command:

```
# saveconf
```

This page has been left intentionally blank.

Chapter 3
Network

Introduction

This chapter will show important configuration settings regarding the network configuration or any feature related to it. The contents of this chapter is briefly presented below:

- Basic Network Settings
- DHCP Client
- Routes and Default Gateway
- DNS Server and Domain Name
- Hosts
- TCP Keepalive
- NTP (Network Time Protocol)
- Filters and Network Address Translation
- VPN Configuration

3.1 Basic Network Settings

This section will show how to configure basic network parameters. This includes configuration of ip addresses, netmasks and hostname.

Hostname

The most basic network related configuration is setting up a hostname. In the Cyclades AlterPath Console Server this can be done editing the */etc/hostname* file.

VI mode

The related file to this configuration is the */etc/hostname*. To change the hostname, edit it and set the desired hostname.

```
CAS
```

File Description 3.1: /etc/hostname

IP address and Netmask

This section will show how to configure the IP address and network mask in the unit. These settings can be made using both methods (VI and CLI).

VI mode

To set the IP address (if DHCP client is disabled) and the netmask it is necessary to edit the *conf.eth_ip* and *conf.eth_mask* parameters in the */etc/pslave/pslave.conf* file.

The example below will set 192.168.160.10 as IP address and 255.255.255.0 as mask. To do that follow the steps below:

Step 1 - Open the */etc/portslave/pslave.conf* file.

To change these parameters it is necessary to edit the */etc/portslave/pslave.conf* file:

```
# vi /etc/portslave/pslave.conf
```

Step 2 - Change parameters values.

With the */etc/portslave/pslave.conf* file opened search for the parameters described below and change their values according to your necessities:

```
conf.eth_ip    192.168.100.1
conf.eth_mask  255.255.255.0
.
.
.
conf.dhcp_client      0
```

File Description 3.2: /etc/portslave/pslave.conf

NOTE: *To define a static IP address it is necessary to disable the DHCP client. Set to "zero" the value of the following line:*

conf.dhcp_client 0

Step 3 - Activate the changes.

Execute the following command to activate the changes:

```
# signal_ras hup
```

Step 4 - Test the configuration

Now you will want to make sure that the ports have been set up properly. Ping the ACS from a remote machine. Using the Windows OS open a command prompt window, type in the following, and then press Enter:

```
# ping <IP assigned to the ACS by DHCP or you>
```

An example would be:

```
# ping 192.168.160.10
```

If you receive a reply, your ACS connection is OK. If there is no reply see Appendix C - Cabling and Hardware Information.

Step 5 - Telnet to the server connected to the first port of the AlterPath Console Server. (This will only work if you selected *socket_server* as your *all.protocol* parameter.)

While still in the DOS window, type the following and then press Enter:

```
# telnet <IP assigned to the ACS by DHCP or you> 7001
```

An example would be:

```
# telnet 192.168.160.10 7001
```

If everything is configured correctly, a telnet session should open on the server connected to port 1. If not, check the configuration, follow the above steps again, and check Appendix C - Software Upgrade and Troubleshooting.

Step 6 - Save the changes.

Execute the following command to save the configuration:

```
# saveconf
```

Step 7 - Reboot the AlterPath Console Server.

After rebooting, the initial configuration is complete.

CLI Mode

It is also possible to configure the IP address and netmask using the CLI interface. This example will set 192.168.160.10 as IP address and 255.255.255.0 as mask. To configure it, follow the steps below:

Step 1 - Open the CLI interface.

It is necessary to enter the CLI interface. To do that run the following command:

```
# config
```

Step 2 - Configuring the unit´s IP address.

Another interface menu will be presented. To configure the IP address type the following command:

```
>>configure ether ip 192.168.160.10
```

Where 192.168.160.10 is the desired IP address.

Step 3 - Configuring unit´s network mask address.

To configure the netmask, still in the CLI interface type:

```
>>configure ether mask 255.255.255.0
```

Where 255.255.255.0 is the desired netmask address.

Step 4 - Saving configuration.

To save all the changes made type the following command:

```
>>write
```

Step 5 - Exiting the CLI mode.

To exit the CLI mode and return to AlterPath's shell, type the following command:

```
>> quit
```

3.2 DHCP Client

DHCP is a protocol that allows network administrators to assign IP addresses automatically to network devices. Without DHCP (or a similar protocol like BOOTP), each device would have to be manually configured. DHCP automatically sends a new IP address to a connected device when it is moved to another location on the network. DHCP uses the concept of a fixed time period during which the assigned IP address is valid for the device it was assigned for. This “lease” time can vary for each device. A short lease time can be used when there are more devices than available IP numbers. For more information, see RFC 2131.

VI mode

The DHCP client on the Ethernet Interface can be configured in two different ways, depending on the action the Cyclades AlterPath Console Server should take in case the DHCP Server does not answer the IP address request:

1. No action is taken and no IP address is assigned to the Ethernet Interface (most common configuration):

   Step 1 - In the */etc/portslave/pslave.conf* file set the global parameter *conf.dhcp_client* to 1.

   Step 2 - Still in the *portslave.conf* file comment all other parameters related to the Ethernet Interface (*conf.eth_ip*, etc.).

   Step 3 - Add the necessary options to the file */etc/network/dhcpcd_cmd* (some options are described later in this session).

2. The Cyclades AlterPath Console Server restores the last IP address previously provided in another boot and assigns this IP address to the Ethernet Interface. For the very first time the unit is powered ON, the IP address restored is 192.168.160.10 in case of failure in the DHCP. The unit goes out from the factory with DHCP enabled (*conf.dhcp_client* 2):

   Step 1 - Set the global parameter *conf.dhcp_client* to 2.

   Step 2 - Comment all other parameters related to the Ethernet Interface (*conf.eth_ip*, etc.).

Step 3 - Add the following lines to the file ***/etc/config_files*** **(from factory file already present in** ***/etc/config_files*****):**

```
/etc/network/dhcpcd_cmd
/etc/dhcpcd-eth0.save
```

File Description 3.3: /etc/config_files

Step 4 - Add the option "-x" to the factory default content of the file ***/etc/network/dhcpcd_cmd*****:**

```
/sbin/dhcpcd -l 3600 -x -c /sbin/handle_dhcp
```

File Description 3.4: /etc/network/dhcpcd_cmd

NOTE: *From the factory, /etc/network/dhcpcd_cmd already has such content.*

Step 5 - Add all other necessary options to the file ***/etc/network/dhcpcd_cmd*** **(some options are described later in this section).**

In both cases if the IP address of the Cyclades AlterPath Console Server or the default gateway are changed, the Cyclades AlterPath Console Server will adjust the routing table accordingly.

Files related to DHCP:

Command/File	Description
/bin/handle_dhcp	The script which is run by the DHCP client each time an IP address negotiation takes place.
/etc/network/dhcpcd_cmd	Contains a command that activates the DHCP client (used by the cy_ras program). Its factory contents are: `/bin/dhcpcd -c /bin/handle_dhcp` The options available that can be used on this command line are: • *-D* - This option forces dhcpcd to set the domain name of the host to the domain name parameter sent by the DHCP Server. The default option is to NOT set the domain name of the host to the domain name parameter sent by the DHCP Server. • *-H* - This option forces dhcpcd to set the host name of the host to the hostname parameter sent by the DHCP Server. The default option is to NOT set the host name of the host to the hostname parameter sent by the DHCP Server. • *-R* - This option prevents dhcpcd from replacing the existing */etc/resolv.conf* file.

Table 3.1: DHCP related files and commands

NOTE. *Do not modify the -c /bin/handle_dhcp option.*

3.3 Routes and Default Gateway

The Cyclades AlterPath Console Server has a static routing table that can be seen using the commands:

```
# route
```

or

```
# netstat -rn
```

The file */etc/network/st_routes* is the Cyclades AlterPath Console Server method for configuring static routes. Routes should be added to the file (which is a script run when the Cyclades AlterPath Console Server is initialized) or at the prompt (for temporary routes) using the following syntax:

```
route [add|del] [-net|-host] target netmask nt_msk [gw gt_way] interf
```

Action/Option	Description
[add|del]	One of these tags must be present. Routes can be either added or deleted.
[-net|-host]	Net is for routes to a network and -host is for routes to a single host.
target	Target is the IP address of the destination host or network.
netmask nt_msk	The tag netmask and nt_mask are necessary only when subnetting is used, otherwise, a mask appropriate to the target is assumed. nt_msk must be specified in dot notation.
gw gt_way	Specifies a gateway, when applicable. gt_way is the IP address or hostname of the gateway.
interf	The interface to use for this route. Must be specified if a gateway is not. When a gateway is specified, the operating system determines which interface is to be used.

Table 3.2: Actions and options for the route command

The next lines will show how to configure the default gateway of the Cyclades AlterPath Console Server .

VI mode

To add routes it is necessary to edit the */etc/network/st_routes* file using the following syntax:

```
route [add|del] [-net|-host] target [netmask] mask [gw] gateway [metric] metric
```

The below example will set the default gateway to the IP address 192.168.0.1. To configure it follow these steps:

Step 1 - Open the *_/etc/network/st_routes_* file using the VI editor.

To do this, run the command:

```
# vi /etc/network/st_routes
```

Step 2 - Inserting the route.

Insert into this file the default route using one of the following commands:

```
# route add -net 0.0.0.0 netmask 0.0.0.0 gw 192.168.0.1
```

the same route can be added in the following way:

```
# route add default gw 192.168.0.1
```

To add a default route to the 192.168.0.1 IP address ONE of the above commands must be inserted into the file */etc/network/st_routes*.

Step 3 - Save the changes made.

To save the changes run the following command:

```
# saveconf
```

3.4 DNS Server and Domain Name

DNS server is a host that resolves host names in the network. This is related to the domain name of the unit, both configurations are made in the same file, so they will be presented together in this section.

VI mode

To set the DNS server and the domain name of your network edit the */etc/resolv.conf* file. The below example will configure "cyclades.com" as domain and 192.168.0.2 as DNS. To configure it follow the steps below:

Step 1 - Open the *l/etc/resolv.conf* file.

It is necessary to edit this file, to do this, run the command:

```
# vi /etc/resolv.conf
```

Step 2 - Configure the */etc/resolv.conf* file

The syntax of this file must be as the following example:

```
domain cyclades.com          #Domain name for the network
nameserver 192.168.0.2       #DNS server for the network
```

File Description 3.5: /etc/resolv.conf

Step 3 - Save the configuration.

To save all changes made, run the command:

```
# saveconf
```

3.5 Hosts

This file should contain the IP address for the Ethernet interface and the same hostname that you entered in the */etc/hostname* file. It may also contain IP addresses and host names for other hosts in the network. The file */etc/hosts* is consulted before the DNS server and is used to convert a name into an IP address.

VI mode

To configure this file follow the steps below:

Step 1 - Open the *\/etc/hosts* file.

To open the file, run the command:

```
# vi /etc/hosts
```

This file should also contain IP addresses and host names for other hosts in the network. The syntax of this file is the following:

```
127.0.0.1        localhost
192.168.160.10   CAS
192.168.160.2    dns-server
```

File Description 3.6: /etc/hosts

Enter as many hosts as necessary, following the above syntax.

Step 2 - Saving the configuration.

To save all the changes made, run the command:

```
# saveconf
```

3.6 TCP Keepalive

The objective of this feature is to allow the AlterPath ACS to recognize when the socket client (ssh or telnet) goes down without closing the connection properly. Currently, if this happens in a serial port the system administrator must close the connection manually or nobody else can access that port anymore.

How it works

The TCP engine of AlterPath ACS will send a tcp keepalive message (ACK) to the client. If the maximum retry number is reached without an answer from the client, the connection is closed.

VI mode

The configuration is done in the file */bin/init_proc_fs* using the linux proc filesystem.

```
# Enable TCP keepalive timer  in ACS (six retries with ten seconds
# of interval from each other).

# keepalive interval when the client is answering

echo 20 > /proc/sys/net/ipv4/tcp_keepalive_time

# keepalive interval when the client is not answering (ACS only).

echo 10 > /proc/sys/net/ipv4/tcp_keepalive_intvl

# number of retries

echo  6 > /proc/sys/net/ipv4/tcp_keepalive_probes

# Enable TCP keepalive timer  (six retries with twenty seconds
# of interval from each other).

echo 20 > /proc/sys/net/ipv4/tcp_keepalive_time
echo  6 > /proc/sys/net/ipv4/tcp_keepalive_probes
```

File Description 3.7: /bin/init_proc_fs

3.7 NTP (Network Time Protocol)

The ntpclient is a Network Time Protocol (RFC-1305) client for UNIX- and Linux-based computers. In order for the Cyclades AlterPath Console Server to work as a NTP client, the IP address of the NTP server must be set in the file */etc/ntpclient.conf*. The program */bin/daemon.sh* reads the configuration file (*/etc/ntpclient.conf*) and runs with the settings of this file.

VI mode configuration

The file */etc/ntpclient.conf* has all the configurable parameters. The parameters that are not presented in the table below MUST NOT be changed.

Step 1 - Edit the */etc/ntpclient.conf* **and change the parameters according to the table below:**

Parameter	Description
ENABLE	This parameter enables the NTP client. It defaults to NO, to enable it choose “YES”.
NTPSERVER	NTP server ip address.
NTPINTERVAL	Time in seconds to ask server.
NTPCOUNT	Specifies how many times the server will be asked. 0 means forever.
NTP_OPT	Other ntp parameters. The possible values for this parameter are listed below: • *-d* -> Print diagnostics • *-h hostname* -> NTP server host (mandatory). • *-l* -> Attempt to lock local clock to server using adjtimex(2). • *-p port* -> Local NTP client UDP port. • *-r* -> Replay analysis code based on stdin. • *-s* -> Clock set (if count is not defined this sets count to 1).

Table 3.3: /etc/ntpclient.conf parameters

Step 2 - Activate and save the changes made.

To activate the configuration, issue the following command:

```
# daemon.sh NTP restart
```

To save the changes, run the command:

```
# saveconf
```

3.8 Filters and Network Address Translation

The Filter feature is available for firmware versions 2.1.0 and above; the Network Address Translation (NAT) feature is available for firmware versions 2.1.1 and above.

Description

IP filtering consists of blocking or not the passage of IP packets, based on rules which describe the characteristics of the packet, such as the contents of the IP header, the input/output interface, or the protocol. This feature is used mainly in firewall applications, which filter the packets that could crack the network system or generate unnecessary traffic in the network.

Network Address Translation (NAT) allows the IP packets to be translated from local network to global network, and vice-versa. This feature is particularly useful when there is demand for more IP addresses in the local network than available as global IP addresses. In the Cyclades AlterPath Console Server, this feature will be used mainly for clustering (one “Master” Console server works as the interface between the global network and the “slave” Console servers).

The Cyclades AlterPath Console Server uses the Linux utility *iptables* to set up, maintain and inspect both the filter and the NAT tables of IP packet rules in the Linux kernel. Besides filtering or translating packets, the iptables utility is able to count the packets which match a rule, and to create logs for specific rules.

Structure of the iptables

The iptables are structured in three levels: table, chain, and rule. A table can contain several chains, and each chain can contain several rules.

Table

The table indicates how the iptables will work. There are currently three independent tables supported by the iptables, but only two will be used:

- *filter:* This is the default table.
- *nat:* This is consulted when a packet that creates a new connection is encountered.

Chain

Each table contains a number of built-in chains and may also contain user-defined chains. The built-in chains will be called according to the type of packet. User-defined chains will be called when a rule which is matched by the packet points to the chain. Each table has a particular set of built-in chains:

for the filter table:

- INPUT - For packets coming into the box itself.
- FORWARD - For packets being routed through the box.
- OUTPUT - For locally-generated packets.

for the nat table:

- PREROUTING - For altering packets as soon as they come in.
- OUTPUT - For altering locally-generated packets as soon as they come in.
- POSTROUTING - For altering packets as they are about to go out.

Rule

Each chain has a sequence of rules. These rules contain:

- How the packet should appear in order to match the rule -> Some information about the packet is checked according to the rule, for example, the IP header, the input and output interfaces, the TCP flags and the protocol.
- What to do when the packet matches the rule -> The packet can be accepted, blocked, logged or jumped to a user-defined chain. For the nat table, the packet can also have its source IP address and source port altered (for the POSTROUTING chain) or have the destination IP address and destination port altered (for the PREROUTING and OUTPUT chain).

When a chain is analyzed, the rules of this chain are reviewed one-by-one until the packet matches one rule. If no rule is found, the default action for that chain will be taken.

Syntax

An iptables tutorial is beyond the scope of this manual. For more information on iptables, see the iptables man page (not included with the Cyclades AlterPath Console Server) or the how-to: http://www.netfilter.org or http://www.iptables.org

The syntax of the iptables command is:

```
# iptables -command chain rule-specification [-t table] [options]

# iptables -E old-chain-name new-chain-name
```

where:

- *table* - Can be filter or nat. If the option -t is not specified, the filter table will be assumed.

- *chain* - Is one of the following:
  for *filter* table: INPUT, OUTPUT, FORWARD or a user-created chain.
  for *nat* table: PREROUTING, OUTPUT, POSTROUTING or a user-created chain.

Command

Only one command can be specified on the command line unless otherwise specified below. For all the long versions of the command and option names, you need to use only enough letters to ensure that iptables can differentiate it from all other options.

Command	Description
-A --append	Append one or more rules to the end of the selected chain. When the source and/or destination names resolve to more than one address, a rule will be added for each possible address combination.
-D --delete	Delete one or more rules from the selected chain. There are two versions of this command. The rule can be specified as a number in the chain (starting at 1 for the first rule) or as a rule to match.
-R --replace	Replace a rule in the selected chain. If the source and/or destination names resolve to multiple addresses, the command will fail. Rules are numbered starting at 1.
-I --insert	Insert one or more rules in the selected chain as the given rule number. Thus if the rule number is 1, the rule or rules are inserted at the head of the chain. This is also the default if no rule number is specified.
-L --list	List all rules in the selected chain. If no chain is selected, all chains are listed. It is legal to specify the -Z (zero) option as well, in which case the chain(s) will be atomically listed and zeroed. The exact output is affected by the other arguments given.
-F --flush	Flush the selected chain. This is equivalent to deleting all the rules one-by-one.
-Z --zero	Zero the packet and byte counters in all chains. It is legal to specify the -L, –list (list) option as well, to see the counters immediately before they are cleared. (See above.)
-N --new-chain	New chain. Create a new user-defined chain by the given name. There must be no target of that name already.

Table 3.4: iptables commands options

Command	Description
-X --delete-chain	Delete the specified user-defined chain. There must be no references to the chain. If there are, you must delete or replace the referring rules before the chain can be deleted. If no argument is given, it will attempt to delete every non-built-in chain in the table.
-P --policy	Set the policy for the chain to the given target. Only non-user-defined chains can have policies, and neither built-in nor user-defined chains can be policy targets.
-E --rename-chain	Rename the user-specified chain to the user-supplied name. This is cosmetic, and has no effect on the structure of the table.
-h --help	Help. Gives a (currently very brief) description of the command syntax.

Table 3.4: iptables commands options

Rule Specification

The following parameters make up a rule specification (as used in the add, delete, insert, replace and append commands):

Parameter	Description
-p	- -protocol[!]protocol The protocol of the rule or of the packet to check. The specified protocol can be one of tcp, udp, icmp, or all, or it can be a numeric value, representing one of these protocols or a different one. A protocol name from */etc/protocols* is also allowed. A "!" argument before the protocol inverts the test. The number zero is equivalent to all. Protocol all will match with all protocols and is taken as default when this option is omitted.
-s	- -source[!]address[/mask] Source specification. Address can be either a hostname, a network name, or a plain IP address. The mask can be either a network mask or a plain number, specifying the number of 1's at the left side of the network mask. Thus, a mask of 24 is equivalent to 255.255.255.0. A "!" argument before the address specification inverts the sense of the address. The flag - -src is a convenient alias for this option.

Table 3.5: iptables rules specifications

Parameter	Description
-d	- -destination[!]address[/mask] Destination specification. See the description of the -s (source) flag for a detailed description of the syntax. The flag - -dst is an alias for this option.
-j	- - jump target This specifies the target of the rule; i.e., what to do if the packet matches it. The target can be a user-defined chain (other than the one this rule is in), one of the special built-in targets which decide the fate of the packet immediately, or an extension (see EXTENSIONS below). If this option is omitted in a rule, then matching the rule will have no effect on the packet's fate, but the counters on the rule will be incremented. The special built-in targets are : • ACCEPT means to let the packet through. • DROP means to drop the packet on the floor. • QUEUE means to pass the packet to userspace (if supported by the kernel). • RETURN means stop traversing this chain and resume at the next rule in the previous (calling) chain. If the end of a built-in chain is reached or a rule in a built-in chain with target RETURN is matched, the target specified by the chain policy determines the fate of the packet.
-i	- -in-interface[!][name] Optional name of an interface via which a packet is received (for packets entering the INPUT and FORWARD chains). When the "!" argument is used before the interface name, the sense is inverted. If the interface name ends in a "+" then any interface which begins with this name will match. If this option is omitted, the string "+" is assumed, which will match with any interface name.
-o	- -out-interface[!][name] Optional name of an interface via which a packet is going to be sent (for packets entering the FORWARD and OUTPUT chains). When the "!" argument is used before the interface name, the sense is inverted. If the interface name ends in a "+" then any interface which begins with this name will match. If this option is omitted, the string "+" is assumed, which will match with any interface name.

Table 3.5: iptables rules specifications

Parameter	Description
[!]	-f - -fragment This means that the rule only refers to second and further fragments of fragmented packets. Since there is no way to tell the source or destination ports of such a packet (or ICMP type), such a packet will not match any rules which specify them. When the "!" argument precedes the "-f" flag, the rule will only match head fragments, or unfragmented packets.
-c	- -set-counters PKTS BYTES This enables the administrater to initialize the packet and byte counters of a rule (during INSERT, APPEND, REPLACE operations).
-v	- -verbose Verbose output. This option makes the list command show the interface address, the rule options (if any), and the TOS masks. The packet and byte counters are also listed, with the suffix 'K', 'M' or 'G' for 1000, 1,000,000 and 1,000,000,000 multipliers respectively (but see the -x flag to change this). For appending, insertion, deletion and replacement, this causes detailed information on the rule or rules to be printed.
-n	- -numeric Numeric output. IP addresses and port numbers will be printed in numeric format. By default, the program will try to display them as host names, network names, or services (whenever applicable).
-x	- -exact Expand numbers. Display the exact value of the packet and byte counters, instead of only the rounded number in K's (multiples of 1000) M's (multiples of 1000K) or G's (multiples of 1000M). This option is only relevant for the -L command.
- -line-numbers	When listing rules, add line numbers to the beginning of each rule, corresponding to that rule's position in the chain.

Table 3.5: iptables rules specifications

Match Extensions

Iptables can use extended packet matching modules. These are loaded in two ways: implicitly, when -p or - -protocol is specified, or with the -m or - -match option, followed by the matching module name; after these, various extra command line options become available, depending on the specific module.

TCP Extensions

These extensions are loaded if the protocol specified is tcp or “-m tcp” is specified. It provides the following options:

TCP extension	Description
--source-port [!] [port[:port]]	Source port or port range specification. This can either be a service name or a port number. Inclusive range can also be specified, using the format port:port. If the first port is omitted, "0" is assumed; if the last is omitted, “65535” is assumed. If the second port is greater then the first they will be swapped. The flag - -sport is an alias for this option.
--destination-port [!] [port[:port]]	Destination port or port range specification. The flag - -dport is an alias for this option.
--tcp-flags [!] mask comp	Match when the TCP flags are as specified. The first argument is the flags which we should examine, written as a comma-separated list, and the second argument is a comma-separated list of flags which must be set. Flags are: SYN ACK FIN RST URG PSH ALL NONE. Hence the command iptables -A FORWARD -p tcp - -tcp-flags SYN,ACK,FIN,RST SYN will only match packets with the SYN flag set, and the ACK, FIN and RST flags unset.
[!] --syn	Only match TCP packets with the SYN bit set and the ACK and FIN bits cleared. Such packets are used to request TCP connection initiation; for example, blocking such packets coming in an interface will prevent incoming TCP connections, but outgoing TCP connections will be unaffected. It is equivalent to - -tcp-flags SYN,RST,ACK SYN. If the "!" flag precedes the "- -syn," the sense of the option is inverted.
--tcp-option [!] number	Match if TCP option set.

Table 3.6: TCP extensions

UDP Extensions

These extensions are loaded if the protocol udp is specified or “-m udp” is specified. It provides the following options:

UDP extension	Description
--source-port [!] [port[:port]]	Source port or port range specification. See the description of the - -source-port option of the TCP extension for details.
--destination-port [!] [port[:port]]	Destination port or port range specification. See the description of the - -destination-port option of the TCP extension for details.

Table 3.7: UDP extensions

ICMP Extension

This extension is loaded if the protocol icmp is specified or “-m icmp” is specified. It provides the following option:

ICMP extension	Description
--icmp-type [!] typename	This allows specification of the ICMP type, which can be a numeric ICMP type, or one of the ICMP type names shown by the command: `iptables -p icmp -h`

Table 3.8: ICMP extensions

Multiport Extension

This module matches a set of source or destination ports. Up to 15 ports can be specified. It can only be used in conjunction with -m tcp or -m udp.

Multiport extension	Description
--source-port [port[,port]]	Match if the source port is one of the given ports.
--destination-port [port[,port]]	Match if the destination port is one of the given ports.
--port [port[,port]]	Match if the both the source and destination port are equal to each other and to one of the given ports.

Table 3.9: Multiport extensions

Target Extensions

Iptables can use extended target modules. The following are included in the standard distribution.

LOG

Turn on kernel logging of matching packets. When this option is set for a rule, the Linux kernel will print some information on all matching packets (like most IP header fields) via the kernel log (where it can be read with syslog-ng).

LOG extension	Description
--log-level level	Level of logging (numeric or see syslog.conf(5)).
--log-prefix prefix	Prefix log messages with the specified prefix; up to 29 letters long, and useful for distinguishing messages in the logs.
--log-tcp-sequence	Log TCP sequence numbers. This is a security risk if the log is readable by users.
--log-tcp-options	Log options from the TCP packet header.
--log-ip-options	Log options from the IP packet header.

Table 3.10: LOG extensions

REJECT (filter table only)

This is used to send back an error packet in response to the matched packet: otherwise it is equivalent to DROP. This target is only valid in the INPUT, FORWARD and OUTPUT chains, and user-defined chains which are only called from those chains. Several options control the nature of the error packet returned:

LOG extension	Description
--reject-with type	The type given can be icmp-net-unreachable, icmp-host-unreachable, icmp-port-unreachable, icmp-proto-unreachable, icmp-net-prohibited or icmp-host-prohibited, which return the appropriate ICMP error message (port-unreachable is the default). The option echo-reply is also allowed; it can only be used for rules which specify an ICMP ping packet, and generates a ping reply. Finally, the option tcp-reset can be used on rules which only match the TCP protocol: this causes a TCP RST packet to be sent back. This is mainly useful for blocking ident probes which frequently occur when sending mail to broken mail hosts (which won't accept your mail otherwise).

Table 3.11: LOG extension

SNAT (NAT table only)

This target is only valid in the nat table, in the POSTROUTING chain. It specifies that the source address of the packet should be modified (and all future packets in this connection will also be mangled), and rules should cease being examined. It takes one option:

SNAT target	Description
--to-source <ipaddr>[-<ipaddr>][:port-port]	This can specify a single new source IP address, an inclusive range of IP addresses, and optionally, a port range (which is only valid if the rule also specifies -p tcp or -p udp). If no port range is specified, then source ports below 1024 will be mapped to other ports below 1024: those between 1024 and 1023 inclusive will be mapped to ports below 1024, and other ports will be mapped to 1024 or above. Where possible, no port alteration will occur.

Table 3.12: SNAT target

DNAT (nat table only)

This target is only valid in the nat table, in the PREROUTING and OUTPUT chains, and user-defined chains which are only called from those chains. It specifies that the destination address of the packet should be modified (and all future packets in this connection will also be mangled), and rules should cease being examined. It takes one option:

DNAT target	Description
--to-destination <ipaddr>[-<ipaddr>][:port-port]	This can specify a single new destination IP address, an inclusive range of IP addresses, and optionally, a port range (which is only valid if the rule also specifies -p tcp or -p udp). If no port range is specified, then the destination port will never be modified.

Table 3.13: DNAT target

MASQUERADE (nat table only)

This target is only valid in the nat table, in the POSTROUTING chain. It should only be used with dynamically assigned IP (dialup) connections: if you have a static IP address, you should use the SNAT target. Masquerading is equivalent to specifying a mapping to the IP address of the interface the packet is going out on, but also has the effect that connections are forgotten when the interface goes down. This is the correct behavior when the next dialup is unlikely to have the same interface address (and hence any established connections are lost anyway). It takes one option:

Target	Description
--to-ports <port>[-<port>]	This specifies a range of source ports to use, overriding the default SNAT source port-selection heuristics (see above). This is only valid if the rule also specifies -p tcp or -p udp).

Table 3.14: Masquerade target

REDIRECT (NAT table only)

This target is only valid in the nat table, in the PREROUTING and OUTPUT chains, and user-defined chains which are only called from those chains. It alters the destination IP address to send the packet to the machine itself (locally-generated packets are mapped to the 127.0.0.1 address). It takes one option:

Target	Description
--to-ports <port>[-<port>]	This specifies a range of source ports to use, overriding the default SNAT source port-selection heuristics (see above). This is only valid if the rule also specifies -p tcp or -p udp).

Table 3.15: Redirect target

How to configure it

The file with the iptables rules is */etc/network/firewall*. The fwset script saves the iptables rules in the file */etc/network/firewall* (command iptales-save > */etc/network/firewall*) and then save the file in the flash memory. The fwset restore restores the iptables rules previously saved in */etc/network/firewall* file (command *iptables-restore </etc/network/firewall*). This command is executed at boot to invoke the last configuration saved.

VI method

Step 1 - Execute fwset restore.

This script will restore the IP Tables chains and rules configured in the */etc/network/firewall* file. This script can be called in the process, whenever the user wants to restore the original configuration.

Step 2 - Add the chains and rules using the command line.

See details of the iptables syntax earlier in this chapter.

Step 3 - Execute *iptables-save > /etc/network/firewall.*

This program will save all the rules and chains of all the tables in the */etc/network/firewall* file.

Step 4 - Execute *updatefiles /etc/network/firewall.*

This program will save the configuration to the flash memory.

3.9 VPN Configuration

The IPsec protocol provides encryption and authentication services at the IP level of the network protocol stack. Working at this level, IPsec can protect any traffic carried over IP, unlike other encryption which generally protects only a particular higher-level protocol (PGP for mail, SSH for login, SSL for Web work and so on). The implementation of IPsec used by the AlterPath Console Server is FreeS/WAN (www.freeswan.org).

IPsec can be used on any machine which does IP networking. Dedicated IPsec gateway machines can be installed wherever required to protect traffic. IPsec can also run on routers, on firewall machines, on various application servers, and on end-user desktop or laptop machines.

IPsec is used mainly to construct a secure connection (tunnel) between two networks (ends) over a not-necessarily-secure third network. In our case, the IPsec will be used to connect the Cyclades AlterPath Console Server securely to a host or to a whole network configurations frequently called host-to-network and host-to-host tunnel. Considering practical aspects, this is the same thing as a VPN, but here one or both sides have a degenerated subnet (only one machine).

Applications of IPsec

Because IPsec operates at the network layer, it is remarkably flexible and can be used to secure nearly any type of Internet traffic. Two applications, however, are extremely widespread:

- A Virtual Private Network, or VPN, allows multiple sites to communicate with the Console Server securely over an insecure Internet by encrypting all communication between the sites and the Console Server.
- Road Warriors connect to the Console Server from home, or perhaps from a hotel somewhere.

A somewhat more detailed description of each of these applications is below. Our Quick Start section will show you how to build each of them.

Using secure tunnels to create a VPN

A VPN, or Virtual Private Network lets the Console Server and a whole network communicate securely when the only connection between them is over a third network which is not trustable. The method is to put a security gateway machine in the network and create a security tunnel between the Console Server and this gateway. The gateway machine and the Console Server encrypt packets entering the untrusted net and decrypt packets leaving it, creating a secure tunnel through it.

Road Warriors

The prototypical Road Warrior is a traveler connecting to the Console Server from a laptop machine. For purposes of this document:

- Anyone with a dynamic IP address is a Road Warrior.
- Any machine doing IPsec processing is a gateway. Think of the single-user Road Warrior machine as a gateway with a degenerate subnet (one machine: itself) behind it.

These require a somewhat different setup than VPN gateways with static addresses and with client systems behind them, but are basically not problematic. There are some difficulties which appear for some Road Warrior connections:

- Road Warriors who get their addresses via DHCP may have a problem. FreeS/WAN can quite happily build and use a tunnel to such an address, but when the DHCP lease expires, FreeS/WAN does not know that. The tunnel fails, and the only recovery method is to tear it down and rebuild it.
- If Network Address Translation (NAT) is applied between the two IPsec Gateways, this breaks IPsec. IPsec authenticates packets on an end-to-end basis, to ensure they are not altered en route. NAT rewrites packets as they go by.

In most situations, however, FreeS/WAN supports Road Warrior connections just fine.

Before you start

This is a quick guide to set up two common configurations: VPN and Road Warrior. There are two examples: a Road Warrior using RSA signature and a VPN using RSA signature. When listing the configuration of the remote side (the equipment the AlterPath Console Server will create a tunnel with) these examples will assume the other end is also running the FreeS/Wan. If it is not your case, make the appropriate conversions for your IPsec software.

Before starting is worth checking some points:

Setup and test networking

Before trying to get FreeS/WAN working, you should configure and test IP networking on the Console Server and on the other end. IPsec can not function without a working IP network beneath it. Many reported FreeS/WAN problems turn out to actually be problems with routing or firewalling. If any actual IPsec problems turn up, you often cannot even recognize them (much less debug them) unless the underlying network is right.

Enabling IPsec on your AlterPath Console Server

The IPsec is disabled by default in the Console Server family. To enable it you must edit the file */etc/ipsec.sh* change "ENABLE=NO" to "ENABLE=YES" and run the "*saveconf*" command. To start IPSEC, type "*daemon.sh restart IPSEC*" <enter>. IPSEC will start automatically during subsequent reboots if you have saved */etc/ipsec.sh* with "*saveconf*".

"Road Warrior" configuration

Think about the administrator that wants to access the Cyclades AlterPath Console Server securely from wherever he is, from his office desk, from his house, or from the hotel room. His IP address will not be always the same, so, for IPsec purposes, he is a "Road Warrior." We refer to the remote machines as Road Warriors. For purposes of IPsec, anyone with a dynamic IP address is a Road Warrior.

Necessary Information

To set up a Road Warrior connection, you need some information about the system on the other end. Connection descriptions use left and right to designate the two ends. We adopt the convention that, from the Console Server's point of view, left=local and right =remote. The Console Server administrator needs to know some things about each Road Warrior:

- The system's public key (for RSA only).
- The ID that system uses in IPsec negotiation.

To get system's public key in a format suitable for insertion directly into the Console Server's *ipsec.conf* file, issue this command on the warrior machine:

```
# /usr/local/sbin/ipsec showhostkey --right
```

The output should look like this (with the key shortened for easy reading):

```
rightrsasigkey=AQNe6hpbROGVES6uXeCxpnd88fdafpO0w5OT0s1LgR7/oUM...
```

The Road Warrior needs to know:

- The Console Server's public key or the secret, and
- The ID the Console Server uses in IPsec negotiation.

which can be generated by running:

```
# /usr/local/sbin/ipsec showhostkey --left
```

on the Console Server. Each warrior must also know the IP address of the Console Server. This information should be provided in a convenient format, ready for insertion in the warrior's *ipsec.conf* file. For example:

```
# left=1.2.3.4 leftid=@acs.example.com leftrsasigkey=0s1LgR7/oUM...
```

The Console Server administrator typically needs to generate this only once. The same file can be given to all warriors.

Setup on the "Road Warrior" machine

Simply add a connection description us-to-Console Server, with the left and right information you gathered above to the *ipsec.conf* file of the warrior system. This might look like:

```
# pre-configured link to Console Server
conn us-to-acs

     # information obtained from Console Server admin
     left=1.2.3.4 # Console Server IP address
     leftid=@acs.example.com
     # real keys are much longer than shown here
     leftrsasigkey=0s1LgR7/oUM...
     # warrior stuff
     right=%defaultroute
     rightid=@xy.example.com
     rightrsasigkey=0s1LgR7/oUM
     # Start this connection when IPsec starts
     auto=start
```

File Description 3.1: Road Warrior ipsec.conf file

IMPORTANT! *The connection name line: "conn us-to-acs" must start on the FIRST column of the line. All other lines after that line must be indented by 1 TAB. This is MANDATORY.*

Setup on the AlterPath Console Server

Adding Road Warrior support so people can connect remotely to your Console Server is straightforward. Just create the file */etc/warrior.connection* and add the following lines to this file:

```
conn gate-xy
     left=1.2.3.4
     leftid=@acs.example.com
     leftrsasigkey=0s1LgR7/oUM...
     # allow connection attempt from any address
     # attempt fails if caller cannot authenticate
     right=%any
     # authentication information
     rightid=@xy.example.com
     rightrsasigkey=0s1LgR7/oUM...
     # Add this connection to the database when IPsec starts
     auto=add
```

File Description 3.2: AlterPath ipsec.conf file

IMPORTANT! *The connection name line: "conn gate-xy" must start on the FIRST column of the line. All other lines after that line must be indented by 1 TAB. This is MANDATORY.*

VPN configuration

Often it may be useful to have explicitly configured IPsec tunnels between the Console Server and a gateway of an office with a fixed IP address (in this case every machine on the office network would have a secure connection with the Console Server), or between the Console Server and the Console Server administrator machine, which must, in this case, have a fixed IP address.

To do it just insert this connection description in your ipsec.conf file with the variables that fit your environment:

```
# sample tunnel
# The network here looks like:
# ACS ----acsnexthop......rightnexthop----right====rightsubnet
# If ACS and right are on the same Ethernet, omit leftnexthop and
# rightnexthop.
conn sample
     # ACS
     left=10.0.0.1
     leftid=@acs.example.com
     # next hop to reach right
     leftnexthop=10.44.55.66
     # This line is only for RSA signature
     leftrsasigkey=0s1LgR7/oUM...
     # right s.g., subnet behind it, and next hop to reach left
     right=10.12.12.1
     rightid=@xy.example.com
     rightnexthop=10.88.77.66
     rightsubnet=192.168.0.0/24
     # Start this connection when IPsec starts
     auto=start
     # This line is for RSA signature
     rightrsasigkey=0s1LgR7/oUM...
```

File Description 3.3: Sample of the ipsec.conf file

IMPORTANT! *The connection name line: "conn sample" must start on the FIRST column of the line. All other lines after that line must be indented by 1 TAB. This is MANDATORY.*

TIP. *There is an alternative way to configure the left and right ipsec rsa keys. Instead of typing (copy/paste) the entire rsa key in the fields: leftrsasigkey and rightrsasigkey inside the /etc/ipsec.conf file, the administrator can just type in the filename where the rsa key was generated. Example:*

leftrsasigkey=@file /etc/ACS48AL.lrsa

The keyword @file and at least one space must precede the filename. Do not forget to include the path of the files containing the RSA keys in the /etc/config_files file.

The good part is that this connection descriptor can be added to both the Console Server and the other end. This is the advantage of using left and right instead of using local remote parameters.

If you give an explicit IP address for left (and left and right are not directly connected), then you must specify leftnexthop (the router which Console Server sends packets to in order to get them delivered to right). Similarly, you may need to specify rightnexthop (vice versa).

Authentication Keys

To build a connection, the Console Server and the other end must be able to authenticate each other. For FreeS/WAN, the default is public key authentication based on the RSA algorithm. IPsec does allow several other authentication methods. On this chapter you will learn how to generate authentication keys and how to exchange keys between systems.

Generating an RSA key pair

The Console Server doesn't have an RSA key pair by default. It will be generated on the first reboot after you have enabled the IPsec daemon in the file */etc/ipsec.sh*. You also can generate your key pair by issuing the following commands as root:

```
# /usr/local/sbin/ipsec newhostkey --bits <key length> --output /etc/ipsec.secrets

# chmod 600 /etc/ipsec.secrets
```

Key generation may take some time. In addition, the Console Server needs a lot of random numbers and therefore needs and uses traffic on the Ethernet to generate them. It is also possible to use keys in other formats, not generated by FreeS/WAN. This may be necessary for interoperation with other IPsec implementations.

Exchanging authentication keys

Once your Cyclades AlterPath Console Server 's key is in *ipsec.secrets*, the next step is to send your public key to everyone you need to set up connections with and collect their public keys. To extract the public part in a suitable format you can use the *ipsec_showhostkey* command. For VPN or Road Warrior applications, use one of the following:

If your AlterPath Console Server is the left side of the tunnel:

```
# /usr/local/sbin/ipsec showhostkey --left
```

If your AlterPath Console Server is the right side of the tunnel:

```
  # /usr/local/sbin/ipsec showhostkey --right
```

These two produce the key formatted for insertion in an ipsec.conf file. Public keys need not be protected as fanatically as private keys. They are intended to be made public; the system is designed to work even if an enemy knows all the public keys used. You can safely make them publicly accessible. For example, put a gateway key on a Web page or make it available in DNS, or transmit it via an insecure method such as email.

IPsec Management

After you have all the configuration done you need to manage all tunnels and manage IPsec itself. This section will show you a few commands that have proven to be useful when managing IPsec and IPsec connections.

The IPsec Daemon

The IPsec daemon (PLUTO) is the program that loads and negotiates the connections. To start the IPsec daemon use the following command:

```
  # /usr/local/sbin/ipsec setup --start
```

Similarly, this command accepts the usual daemon commands as stop and restart.

The ipsec daemon is automatically initialized when you boot your Console Server equipment.

Adding and Removing a Connection

All the connections can be loaded to the IPsec database at boot time if these connections have the auto parameter set to add. However if a certain connection doesn't have this option set and you wish to add this connection manually you can use the following command:

```
  # /usr/local/sbin/ipsec auto --add <connection name>
```

Similarly, to take a connection out of the IPsec database you can use the command:

```
  # /usr/local/sbin/ipsec auto --delete <connection name>
```

Once a connection descriptor is in the IPsec internal database, IPsec will accept the other end to start the security connection negotiation. You can also start its negotiation as explained in the next section.

Starting and Stopping a Connection

All the connections can be negotiated at boot time if these connections have the *auto* parameter set to *start*. However if a certain connection doesn't have this option set, you can set it. Once a connection descriptor is in the IPsec internal database, you can start its negotiation using the command:

```
# /usr/local/sbin/ipsec auto --up <connection name>
```

Similarly to close a tunnel you use the command:

```
# /usr/local/sbin/ipsec auto --down <connection name>
```

Below you can see the output of a successful up operation:

```
[root@acs_cas root]# ipsec auto --up test
104 "test" #5: STATE_MAIN_I1: initiate
106 "test" #5: STATE_MAIN_I2: sent MI2, expecting MR2
108 "test" #5: STATE_MAIN_I3: sent MI3, expecting MR3
004 "test" #5: STATE_MAIN_I4: ISAKMP SA established
112 "test" #6: STATE_QUICK_I1: initiate
004 "test" #6: STATE_QUICK_I2: sent QI2, IPsec SA established
```

IPsec look

It gives you a detailed information about the IPsec state:

```
[root@acs_cas root]# ipsec look
acs_cas Mon Oct 28 16:40:24 PST 2002
64.186.161.96/32 -> 64.186.161.128/32 => tun0x1006@64.186.161.128
esp0x4e1a10ce@64.186.161.128 (0)
ipsec0->eth0 mtu=16260(1443)->1500

esp0x4e1a10ce@64.186.161.128 ESP_3DES_HMAC_MD5: dir=out
src=64.186.161.96 iv_bits=64bits iv=0xd491678073a22185 ooowin=64
alen=128 aklen=128 eklen=192 life(c,s,h)=addtime(4,0,0)

esp0xa99f2a63@64.186.161.96 ESP_3DES_HMAC_MD5: dir=in
src=64.186.161.128 iv_bits=64bits iv=0x46209cee5f952117 ooowin=64
alen=128 aklen=128 eklen=192 life(c,s,h)=addtime(4,0,0)

tun0x1005@64.186.161.96 IPIP: dir=in src=64.186.161.128
policy= 64.186.161.128/32->64.186.161.96/32 flags=0x8<>
life(c,s,h)=addtime(4,0,0)

tun0x1006@64.186.161.128 IPIP: dir=out src=64.186.161.96
life(c,s,h)=addtime(4,0,0)
```

```
Destination Gateway Genmask Flags MSS Window irtt Iface
0.0.0.0 64.186.161.1 0.0.0.0 UG 40 0 0 eth0
64.186.161.0 0.0.0.0 255.255.255.0 U 40 0 0 eth0
64.186.161.0 0.0.0.0 255.255.255.0 U 40 0 0 ipsec0
64.186.161.128 64.186.161.128 255.255.255.255 UGH 40 0 0 ipsec0
```

In this output you can see that there is an activated tunnel between the networks 64.186.161.96/32 and 64.186.161.128/32. You can also see the routing table for this host after the encryption information .

IPsec whack

The ipsec whack command show the status of the connections.

```
[root@acs_cas root]# ipsec whack --status
000 interface ipsec0/eth0 64.186.161.96
000
000 "test": 64.186.161.96[@micro]...64.186.161.128[@ACS ]
000 "test": ike_life: 3600s; ipsec_life: 28800s; rekey_margin: 540s;
rekey_fuzz: 100%; keyingtries: 0
000 "test": policy: RSASIG+ENCRYPT+TUNNEL+PFS; interface: eth0; routed
000 "test": newest ISAKMP SA: #5; newest IPsec SA: #6; route owner: #6
000
000 #6: "test" STATE_QUICK_I2 (sent QI2, IPsec SA established);
EVENT_SA_REPLACE in 28245s; newest IPSEC; route owner
000 #6: "test" esp.4e1a10ce@64.186.161.128 esp.a99f2a63@64.186.161.96
tun.1006@64.186.161.128 tun.1005@64.186.161.96
000 #5: "test" STATE_MAIN_I4 (ISAKMP SA established); EVENT_SA_REPLACE in
3019s; newest ISAKMP
```

As you can see, it shows almost the same information shown by the ipsec auto -up command. You can use this command if the up command doesn't show anything on the screen (it can happen depending on the ACS syslog configuration).

The IPsec Configuration Files in Detail

This section will describe the file */etc/ipsec.conf* in detail.

Description

The *ipsec.conf* file specifies most configuration and control information for the FreeS/WAN IPsec subsystem. (The major exception is secrets for authentication; *ipsec.secrets*) Its contents are not security-sensitive unless manual keying is being done for more than just testing, in which case the encryption and authentication keys in the descriptions for the manually-keyed connections are very sensitive (and those connection descriptions are probably best kept in a separate file, via the include facility described below).

The file is a text file, consisting of one or more sections. White space followed by # followed by anything to the end of the line is a comment and is ignored, as are empty lines which are not within a section.

A line which contains include and a file name, separated by white space, is replaced by the contents of that file, preceded and followed by empty lines. If the file name is not a full pathname, it is considered to be relative to the directory containing the including file. Such inclusions can be nested. Only a single filename may be supplied, and it may not contain white space, but it may include shell wildcards for example:

```
include ipsec.*.conf
```

The intention of the include facility is mostly to permit keeping information on connections, or sets of connections, separate from the main configuration file. This permits such connection descriptions to be changed, copied to the other security gateways involved, etc., without having to constantly extract them from the configuration file and then insert them back into it. Note the also parameter (described below) which permits splitting a single logical section (e.g., a connection description) into several actual sections.

A section begins with a line of the form:

```
type name
```

where type indicates what type of section follows, and name is an arbitrary name which distinguishes the section from others of the same type. (Names must start with a letter and may contain only letters, digits, periods, underscores, and hyphens.) All subsequent non-empty lines which begin with white space are part of the section; comments within a section must begin with white space too. There may be only one section of a given type with a given name.

Lines within the section are generally of the following form:

```
        parameter=value
```

(Note the mandatory preceding TAB.) There can be white space on either side of the =. Parameter names follow the same syntax as section names, and are specific to a section type. Unless otherwise explicitly specified, no parameter name may appear more than once in a section.

An empty value stands for the system default value (if any) of the parameter, i.e., it is roughly equivalent to omitting the parameter line entirely. A value may contain white space only if the entire value is enclosed in double quotes ("); a value cannot itself contain a double quote, nor may it be continued across more than one line.

Numeric values are specified to be either an integer (a sequence of digits) or a decimal number (sequence of digits optionally followed by . and another sequence of digits).

There is currently one parameter which is available in any type of section:

also

The value is a section name; the parameters of that section are appended to this section, as if they had been written as part of it. The specified section must exist, must follow the current one, and must have the same section type. (Nesting is permitted, and there may be more than one also in a single section, although it is forbidden to append the same section more than once.) This allows, for example, keeping the encryption keys for a connection in a separate file from the rest of the description, by using both an also parameter and an include line.

A section with name *%default* specifies defaults for sections of the same type. For each parameter in it, any section of that type which does not have a parameter of the same name gets a copy of the one from the %default section. There may be multiple %default sections of a given type, but only one default may be supplied for any specific parameter name, and all %default sections of a given type must precede all non-%default sections of that type. %default sections may not contain also parameters.

Currently there are two types of sections: a *config* section specifies general configuration information for IPsec, while a *conn* section specifies an IPsec connection.

Conn Sections

A conn section contains a connection specification, defining a network connection to be made using IPsec. The name given is arbitrary, and is used to identify the connection to *ipsec_auto* and *ipsec_manual*. Here's a simple example:

```
conn snt
     left=10.11.11.1
     leftsubnet=10.0.1.0/24
     leftnexthop=172.16.55.66
     right=192.168.22.1
     rightsubnet=10.0.2.0/24
     rightnexthop=172.16.88.99
     keyingtries=0 # be very persistent
```

File Description 3.4: part of the /etc/ipsec.conf file

To avoid trivial editing of the configuration file to suit it to each system involved in a connection, connection specifications are written in terms of left and right participants, rather than in terms of local and remote. Which participant is considered left or right is arbitrary; IPsec figures out which one it is being run on based on internal information. This permits using identical connection specifications on both ends.

Many of the parameters relate to one participant or the other; only the ones for left are listed here, but every parameter whose name begins with left has a right counterpart, whose description is the same but with left and right reversed.

Parameters are optional unless marked required; a parameter required for manual keying need not be included for a connection which will use only automatic keying, and vice versa.

Conn parameters: General. The following parameters are relevant to both automatic and manual keying. Unless otherwise noted, for a connection to work, in general it is necessary for the two ends to agree exactly on the values of these parameters.

- type: The type of the connection. Currently the accepted values are: tunnel (the default) signifying a host-to-host, host-to-subnet, or subnet-to-subnet tunnel; transport, signifying host-to-host transport mode; and passthrough (supported only for manual keying), signifying that no IPsec processing should be done at all.
- left: Required. The IP address of the left participant's public-network interface. If it is the magic value %defaultroute, and interfaces=%defaultroute is used in the config setup section, left will be filled in automatically with the local address of the default-route interface (as determined at IPsec startup time). This also overrides any value supplied for leftnexthop. (Either left or right may be %defaultroute, but not both.) The magic value %any signifies an address to be filled in (by automatic keying) during negotiation; the magic value %opportunistic signifies that both left and leftnexthop are to be filled in (by automatic keying) from DNS data for left's client.
- leftsubnet: Private subnet behind the left participant, expressed as network/ netmask. If omitted, essentially assumed to be left/32, signifying that the left end of the connection goes to the left participant only.
- leftnexthop: Next-hop gateway IP address for the left participant's connection to the public network. Defaults to %direct (meaning right).
- leftupdown: What updown script to run to adjust routing and/or firewalling when the status of the connection changes.

Conn parameters: Automatic Keying. The following parameters are relevant only to automatic keying, and are ignored in manual keying. Unless otherwise noted, for a connection to work, in general it is necessary for the two ends to agree exactly on the values of these parameters.

- auto: What operation, if any, should be done automatically at IPsec startup; currently- accepted values are add (signifying an ipsec auto --add), route (signifying that plus an ipsec auto --route), start (signifying that plus an ipsec auto --up), and ignore (also the default) (signifying no automatic startup operation). This parameter is ignored unless the plutoload or plutostart configuration parameter is set suitably; see the config setup discussion below.
- auth: Whether authentication should be done as part of ESP encryption, or separately using the AH protocol, acceptable values are esp (the default) and ah.
- authby: How the two security gateways should authenticate each other. Acceptable values are secret for shared secrets (the default) and rsasig for RSA digital signatures.
- leftid: How the left participant should be identified for authentication. Defaults to left. Can be an IP address or a fully-qualified domain name preceded by @ (which is used as a literal string and not resolved).
- leftrsasigkey: The left participant's public key for RSA signature authentication, in RFC 2537 format. The magic value %none means the same as not specifying a value (useful to override a default). The value %dnsondemand means the key is to be fetched from DNS at the time it is needed. The value %dnsonload means the key is to be fetched from DNS at the time the connection description is read from ipsec.conf. Currently this is treated as %none if right=%any or right=%opportunistic. The value %dns is currently treated as %dnsonload but will change to %dnsondemand in the future. The identity used for the left participant must be a specific host, not %any or another magic value. Caution: if two connection descriptions specify different public keys for the same leftid, confusion and madness will ensue.
- pfs: Whether Perfect Forward Secrecy of keys is desired on the connection's keying channel. (With PFS, penetration of the key-exchange protocol does not compromise keys negotiated earlier.) Acceptable values are yes (the default) and no.
- keylife: How long a particular instance of a connection (a set of encryption/ authentication keys for user packets) should last, from successful negotiation to expiry. Acceptable values are an integer optionally followed by s (a time in seconds) or a decimal number followed by m, h, or d (a time in minutes, hours, or days respectively) (default 8.0h, maximum 24h).
- rekey: Whether a connection should be renegotiated when it is about to expire. Acceptable values are yes (the default) and no.
- rekeymargin: How long before connection expiry or keying-channel expiry should attempts to negotiate a replacement begin. Acceptable values as for keylife (default 9m).
- redeyfuzz: Maximum percentage by which rekeymargin should be randomly increased to randomize rekeying intervals (important for hosts with many connections). Acceptable values are an integer, which may exceed 100, followed by a %.
- keyingtries: How many attempts (an integer) should be made to negotiate a connection, or a replacement for one, before giving up (default 3). The value 0 means never give up.
- ikelifetime: How long the keying channel of a connection (buzzphrase: ISAKMP SA) should last before being renegotiated. Acceptable values as for keylife.

- compress: Whether IPComp compression of content is desired on the connection. Acceptable values are yes and no (the default).

Conn parameters: Manual Keying. The following parameters are relevant only to manual keying, and are ignored in automatic keying. Unless otherwise noted, for a connection to work, in general it is necessary for the two ends to agree exactly on the values of these parameters. A manually-keyed connection must specify at least one of AH or ESP.

- spi or spibase: Spi or spibase is required for manual keying. the SPI number to be used for the connection. Must be of the form 0xhex, where hex is one or more hexadecimal digits. (Note: it will generally be necessary to make spi at least 0x100 to be acceptable to KLIPS, and use of SPIs in the range 0x100-0xfff is recommended.)
- esp: ESP encryption/authentication algorithm to be used for the connection, e.g. 3des-md5-96.
- espenckey: ESP encryption key.
- espauthkey: ESP authentication key.
- espreplay_window: ESP replay-window setting. An integer from 0 to 64. Relevant only if ESP authentication is being used.
- leftespspi: SPI to be used for the leftward ESP SA, overriding automatic assignment using spi or spibase. Typically a hexadecimal number beginning with 0x.
- ah: AH authentication algorithm to be used for the connection, e.g. hmac-md5-96. Default is not to use AH.
- ahkey: Required if ah is present. AH authentication key
- ahreplay_window: AH replay-window setting. An integer from 0 to 64.
- leftahspi: SPI to be used for the leftward AH SA, overriding automatic assignment using spi or spibase. Typically a hexadecimal number beginning with 0x.

Config Section

At present, the only config section known to the IPsec software is the one named setup, which contains information used when the software is being started. Here's an example:

```
config setup
     interfaces="ipsec0=eth1 ipsec1=ppp0"
     klipsdebug=none
     plutodebug=all
     manualstart=
     plutoload="snta sntb sntc sntd"
     plutostart=
```

File Description 3.5: part of the /etc/ipsec.conf file

Parameters are optional unless marked required. The currently-accepted parameter names in a config setup section are:

- interfaces: Required. Virtual and physical interfaces for IPsec to use: a single virtual= physical pair, a quoted list of pairs separated by white space, or %defaultroute, which means to find the interface d that the default route points to, and then act as if the value was ipsec0=d.
- forwardcontrol: Whether setup should turn IP forwarding on (if it's not already on) as IPsec is started, and turn it off again (if it was off) as IPsec is stopped. Acceptable values are yes and (the default) no.
- klipsdebug: How much KLIPS debugging output should be logged. An empty value, or the magic value none, means no debugging output (the default). The magic value all means full output.
- plutodebug: How much Pluto debugging output should be logged. An empty value, or the magic value none, means no debugging output (the default). The magic value all means full output.
- dumpdir: In what directory should things started by setup (notably the Pluto daemon) be allowed to dump core. The empty value (the default) means they are not allowed to.
- manualstart: Which manually-keyed connections to set up at startup (can be empty, a name, or a quoted list of names separated by white space).
- plutoload: Which connections (by name) to load into Pluto's internal database at startup (can be empty, a name, or a quoted list of names separated by white space); see ipsec_auto for details. Default is none. If the special value %search is used, all connections with auto=add, auto=route, or auto=start are loaded.
- plutostart: Which connections (by name) to attempt to negotiate at startup (can be empty, a name, or a quoted list of names separated by white space). Any such names which do not appear in plutoload are implicitly added to it. Default is none. If the special value %search is used, all connections with auto=route or auto=start are routed, and all connections with auto=start are started.
- plutowait: Specify if Pluto should wait for each plutostart negotiation attempt to finish before proceeding with the next one. Values are yes (the default) or no.
- prepluto: Shell command to run before starting Pluto. For example, to decrypt an encrypted copy of the ipsec.secrets file. It's run in a very simple way. Complexities like I/O redirection are best hidden within a script. Any output is redirected for logging, so running interactive commands is difficult unless they use /dev/tty or equivalent for their interaction. Default is none.
- postpluto: Shell command to run after starting Pluto (e.g., to remove a decrypted copy of the ipsec.secrets file).
- fragicmp: Whether a tunnel need to fragment a packet should be reported back with an ICMP message, in an attempt to make the sender lower his PMTU estimate. Acceptable values are yes (the default) and no.
- packetdefault: What should be done with a packet which reaches KLIPS (via a route into a virtual interface) but does not match any route. Acceptable values are pass (insecure unless you really know what you're doing), drop (the default), and reject (currently same as drop).

- hidetos: Whether a tunnel packet's TOS field should be set to 0 rather than copied from the user packet inside. Acceptable values are yes (the default) and no.
- uniqueids: Whether a particular participant ID should be kept unique, with any new (automatically keyed) connection using an ID from a different IP address deemed to replace all old ones using that ID. Acceptable values are yes and no (the default).
- overridemtu: Value that the MTU of the ipsec interface(s) should be set to, overriding IPsec's (large) default. This parameter is needed only in special situations.

Chapter 4
Administration

The objective of this chapter is showing any task related to the administration of the unit. This includes the following topics:

- SNMP
- CronD
- Dual Power Management
- Syslog-ng
- Generating Alarms (Syslog-ng)
- Terminal Appearance
- Centralized Management
- Date, Time and Time Zone
- Session Sniffing
- Start and Stop Services

4.1 SNMP

Short for Simple Network Management Protocol: a set of protocols for managing complex networks. The first versions of SNMP were developed in the early 80s. SNMP works by sending messages, called protocol data units (PDUs), to different parts of a network. SNMP-compliant devices, called agents, store data about themselves in Management Information Bases (MIBs) and return this data to the SNMP requesters.

Cyclades AlterPath Console Server uses the net-snmp package (http://www.net-snmp.org).

IMPORTANT! *Check the SNMP configuration before gathering information about Cyclades AlterPath Console Server by SNMP. There are different types of attacks an unauthorized user can implement to retrieve sensitive information contained in the MIB. By default, the SNMP configuration in Cyclades AlterPath Console Server cannot permit the public community to read SNMP information.*

The net-snmp supports snmp version 1, 2 and 3. To use SNMP version 1 or 2 (community), you need to configure the communities in the snmp config file (*/etc/snmp/snmpd.conf*). For example, to include the communities cyclades and public, you need add the following lines in */etc/snmp/snmpd.conf*:

```
# cyclades is read-write community
rwcommunity cyclades
# public is a read-only community
rocommunity public
```

File Description 4.1: part of the /etc/snmp/snmpd.conf file

To use SNMP version 3 (username/password), perform the following steps:

Step 1 - Create a file */etc/snmp/snmpd.local.conf* **with the following line:**

```
# createUser <username> MD5 <password> DES
```

For example :

```
# createUser usersnmp MD5 user_snmp_passwd DES
```

IMPORTANT! *The SNMP v3 password MUST be at least 8 characters long. If a password shorter than 8 characters is inserted, no error messages will be reported, but the SNMP user will not be created.*

Step 2 - Edit the */etc/snmp/snmpd.conf* **file.**

If the user has permission to read only, to add the line :

```
# rouser <username> (eg.: rouser usersnmp).
```

If the user has permission to read and write, to add the line :

```
# rwuser <username> (eg.: rwuser usersnmp).
```

Step 3 - Include the following line in */etc/config_files*:

```
/etc/snmp/snmpd.local.conf
```

You can configure the */etc/snmp/snmpd.conf* file as indicated later in this section.

1. Snmp version 1
   - RFC1155 - SMI for the official MIB tree
   - RFC1213 - MIB-II
2. Snmp version 2

- RFC2578 - Structure of Management Information Version 2 (SMIv2)
- RFC2579 - Textual Conventions for SMIv2
- RFC2580 - Conformance Statements for SMIv2

3. Snmp version 3

   - RFC2570 - Introduction to Version 3 of the Internet-standard Network Management Framework.
   - RFC2571 - An Architecture for Describing SNMP Management Frameworks.
   - RFC2572 - Message Processing and Dispatching for the Simple Network Management Protocol (SNMP).
   - RFC2573 - SNMP Applications.
   - RFC2574 - User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3).
   - RFC2575 - View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP).
   - RFC2576 - Coexistence between Version 1, Version 2, and Version 3 of the Internet-standard Network Management Framework.

4. Private UCD SNMP mib extensions (enterprises.2021)

   - Information about memory utilization (*/proc/meminfo*)
   - Information about system status (vmstat)
   - Information about net-snmp packet

5. Private Cyclades Vendor MIB (enterprises.2925)

   - Cyclades AlterPath Console Server remote Management Object Tree (cyclades.4). This MIB permits you to get informations about the product, to read/write some configuration items and to do some administration commands. (For more details see the cyclades.mib file.)

Configuration

This section will explain how to configure the SNMP using the VI editor.

VI Method - Involved parameters and passed values

The next steps are going to show a SNMP v1 configuration. The related file for SNMP configuration is: */etc/snmp/snmpd.conf* file.

Step 1 - Map the community name *public* into a *security name*.

```
#        sec.name      source      community
com2sec  notConfigUser default     public
```

File Description 4.2: Part of the file /etc/snmp/smnpd.conf

Step 2 - Map the security name into a group name.

```
#       groupName      securityModel  securityName
group   notConfigGroup v1             notConfigUser
#group  notConfigGroup v2c            notConfigUser
```

File Description 4.3: Part of the file /etc/snmp/smnpd.conf

Step 3 - Create a view to which the group has rights.

```
#    name     incl/excl   subtree     mask(optional)
view all      included    .1
```

File Description 4.4: Part of the file /etc/snmp/smnpd.conf

Step 4 - Grant the group read-only access to the all view.

```
#group               contextsec.model    sec.level    prefix    readwrite notif
access   notConfigGroup ""       any          noauth       exact  all none none
```

File Description 4.5: Part of the file /etc/snmp/smnpd.conf

Step 5 - Activate the changes, by issuing the command:

```
# signal_ras hup
```

Step 6 - Save the configuration, running the command:

```
# saveconf
```

4.2 CronD

CronD is a service provided by the Cyclades AlterPath Console Server system that allows automatic, periodically-run custom-made scripts. It replaces the need for the same commands to be run manually.

How to configure

The crond daemon in the Cyclades AlterPath Console Server has a peculiar way of configuration that is basically it is divided in three parts:

- */etc/crontab_files* - The name of this file can't be changed and it must point only to ONE file. Further information about it will be given in the next lines.
- source file - This file holds information about frequency and which files should be executed. It can have any name, since it is pointed out by the */etc/crontab_files*.
- script files - These are the script files scheduled and pointed by the source file explained above.

The following parameters are created in the */etc/crontab_files* file:

- status - Active or inactive. If this item is not active, the script will not be executed.
- user - The process will be run with the privileges of this user, who must be a valid local user.
- source - Pathname of the crontab file that specifies frequency of execution, the name of shell script, etc. It should be set using the traditional crontab file format.

```
active root /etc/tst_cron.src
```

File Description 4.6: /etc/crontab_files

NOTE: *In /etc/crontab_files, you can only have one active entry per user. For instance, from the example above, you cannot add another active entry for root because it already has an entry. If you want to add more scripts, you can just add them to the source file, eg.: (/etc/tst_cron.src).*

The */etc/crontab_files* file can point to any desired file that calls the scripts to be run. The Cyclades AlterPath Console Server has example file for it (*/etc/tst_cron.src*). The file that is pointed out in the */etc/crontab_files* file must follow this structure:

```
PATH=/usr/bin:/bin
SHELL=/bin/sh
HOME=/

0-59 * * * *   /etc/tst_cron.sh
```

File Description 4.7: /etc/tst_cron.src

This file is called */etc/tst_cron.src*, but it could have any other name, since it follows the above structure.
The fourth line of the example file follows this structure: minutes, hours, month day, month, week day and command .
It is possible to specify different tasks to run on different dates and times. Each command must be on a separated line lines. Find more information about the crontab syntax below:

Crontab Syntax. A crontab task consists of four date/time fields and a command field. Every minute cron checks all crontabs for a match between the current date/time and their tasks. If there's a match, the command is executed. The system crontab has an additional field "User" that tells cron with which user id the command should be executed.

The fields are:

- Min - minute of execution, 0-59
- Hour - hour of execution, 0-23
- Mday - day of month of execution, 1-31
- Month - month of execution, 1-12 (or names)
- Wday - day of week of execution, 0-7 (0 or 7 is sunday, or names)
- Command - Anything that can be launched from the command line

Possible values for the fields:

- * - matches all values, e.g. a * in month means: "every month"
- x-y - matches the range x to y, e.g. 2-4 in Mday means "on the 2nd, 3rd, and 4th of the month"
- x/n - in range x with frequency n, e.g. */2 in Hour means "every other hour"

Month also accepts names, e.g. jan, Feb (case insensitive). This does not support ranges, though. Weekdays can also be given as names, e.g. sun, Mon.

VI Method - Involved parameters and passed values

Example:
In this step by step example we will configure a script named *tst_cron.sh* to run every minute. This example just explains the necessary steps, because actually all files are already present in the Cyclades AlterPath Console Server by default.

Step 1 - Activate the crond daemon in the */etc/crontab_files*.

As explained before this file configures which file contains information about which scripts are going to be run. Activate the daemon, by editing the /etc/crontab_files changing the line, like below:

```
active root /etc/tst_cron.src
```

Step 2 - Edit the */etc/tst_cron.src*, to specify which scripts will be executed.

This file must point out all scripts to be executed. It also specifies the periodicity of execution of each script, according to the following syntax:

```
0-59 * * * * /etc/tst_cron.sh
```

In this case the *tst_cron.sh* script will run every minute.

Step 3 - Save the changes.

Execute the following command in to save the configuration:

```
# saveconf
```

Step 4 - Activate changes.

To activate the changes it is necessary to reboot the Cyclades AlterPath Console Server by issuing the command:

```
# reboot
```

4.3 Dual Power Management

The Cyclades AlterPath Console Server comes with two power supplies which it can self-monitor. If either of them fails, two actions are performed: sounding a buzzer and generating a syslog message. This automanagement can be disabled (no actions are taken) or enabled (default), any time by issuing the commands:

```
# signal_ras buzzer off
```

```
# signal_ras buzzer on
```

To disable the buzzer in boot time, edit the shell script /bin/ex_wdt_led.sh and remove the keyword “buzzer.” The buzzer won’t sound if there is a power failure in any power supply. This parameter does not affect the behavior of the command “signal_ras buzzer on/off.” To make this change effective even after future reboots, create a line with “/bin/ex_wdt_led.sh” in /etc/config_files, save and quit that file and run saveconf.

NOTE: *This section applies only to the dual power supply model of the Cyclades AlterPath Console Server.*

How to configure

There are no parameters to be configured. However, if you want to generate alarms in case of a power failure, the *syslog-ng.conf* file must be changed. See the section Generating Alarms.

4.4 Syslog-ng

The syslog-ng daemon provides a modern treatment to system messages. Its basic function is to read and log messages to the system console, log files, other machines (remote syslog servers) and/or users as specified by its configuration file. In addition, syslog-ng is able to filter messages based on their content and to perform an action (e.g. to send an e-mail or pager message). In order to access these functions, the syslog-ng.conf file needs some specific configuration.

The configuration file (default: */etc/syslog-ng/syslog-ng.conf*) is read at startup and is reread after reception of a hangup (HUP) signal. When reloading the configuration file, all destination files are closed and reopened as appropriate. The *syslog-ng* reads from sources (files, TCP/UDP connections, syslogd clients), filters the messages and takes an action (writes in files, sends snmptrap, pager, e-mail or syslogs to remote servers).

There are five steps required for configuring syslog-ng:

Step 1: Define Global Options.
Step 2: Define Sources.
Step 3: Define Filters.
Step 4: Define Actions (Destinations).
Step 5: Connect all of the above.

These five tasks are going to be explained in this section.

Port Slave Parameters Involved with syslog-ng

- *conf.facility* - This value (0-7) is the Local facility sent to the syslog-ng from PortSlave.
- *conf.DB_facility* - This value (0-7) is the local facility sent to the syslog-ng with data when syslog_buffering and/or alarm is active. When nonzero, the contents of the data buffer are sent to the syslog-ng every time a quantity of data equal to this parameter is collected. The syslog level for data buffering is hard coded to level five (notice) and facility local[0+ conf.DB_facility]. The file /etc/syslog-ng/syslog-ng.conf should be set accordingly for the syslog-ng to take some action. Example value: 0.
- *all.syslog_buffering* - When nonzero, the contents of the data buffer are sent to the syslog-ng every time a quantity of data equal to this parameter is collected. The syslog message is sent to syslog-ng with NOTICE level and LOCAL[0+conf.DB_facility] facility.

The Syslog Functions

This section shows the characteristics of the syslog-ng that is implemented for all members of the Cyclades AlterPath Console Server family. It is divided into three parts:

1. Syslog-ng and its Configuration
2. Syslog-ng Configuration to use with Syslog Buffering Feature
3. Syslog-ng Configuration to use with Multiple Remote Syslog Servers

Syslog-ng and its Configuration

The five steps previously mentioned are detailed below.

Step 1 - Specify Global Options.

You can specify several global options to syslog-ng in the options statement:

```
options { opt1(params); opt2(params); ... };
```

where *optN* can be any of the following:

Option	Description
time_reopen(n)	The time to wait before a dead connection is reestablished.
time_reap(n)	The time to wait before an idle destination file is closed.
sync_freq(n)	The number of lines buffered before written to file. (The file is synced when this number of messages has been written to it.)
mark_freq(n)	The number of seconds between two MARKS lines.
log_fifo_size(n)	The number of lines fitting to the output queue.
chain_hostname (yes/no) or long_hostname (yes/no)	Enable/disable the chained hostname format.
use_time_recvd (yes/no)	Use the time a message is received instead of the one specified in the message.
use_dns (yes/no)	Enable or disable DNS usage. syslog-ng blocks on DNS queries, so enabling DNS may lead to a Denial of Service attach.
gc_idle_threshold(n)	Sets the threshold value for the garbage collector, when syslog-ng is idle. GC phase starts when the number of allocated objects reach this number. Default: 100.
gc_busy_threshold(n)	Sets the threshold value for the garbage collector. When syslog-ng is busy, GC phase starts.

Table 4.1: "Global Options" parameters (Syslog-ng configuration)

Option	Description
create_dirs(yes/no)	Enable the creation of new directories.
owner(name)	Set the owner of the created file to the one specified. Default: root.
group(name)	Set the group of the created file to the one specified. Default: root.
perm(mask)	Set the permission mask of the created file to the one specified. Default: 0600.

Table 4.1: "Global Options" parameters (Syslog-ng configuration)

Step 2 - Define sources.

To define sources use this statement:

```
source <identifier> { source-driver([params]); source driver([params]); ...};
```

where:

- *identifier* - Has to uniquely identify this given source.
- *source-driver* - Is a method of getting a given message.
- *params* - Each source-driver may take parameters. Some of them are required, some of them are optional.

The following source-drivers are available:

Option	Description
internal()	Messages are generated internally in syslog-ng.
unix-stream (filename [options]) and unix-dgram (filename [options])	They open the given AF_UNIX socket, and start listening for messages. Options: owner(name), group(name), perm(mask) are equal global options keep-alive(yes/no) - Selects whether to keep connections opened when syslog-ng is restarted. Can be used only with unix_stream. Default: yes max-connections(n) - Limits the number of simultaneously opened connections. Can be used only with unix_stream. Default: 10.

Table 4.2: "Source Drivers" parameters (Syslog-ng configuration)

Option	Description

tcp([options])
and
udp([options])
	These drivers let you receive messages from the network, and as the name of the drivers show, you can use both TCP and UDP.
	None of tcp() and udp() drivers require positional parameters. By default they bind to 0.0.0.0:514, which means that syslog-ng will listen on all available interfaces.
	Options:
	ip(<ip address>) - The IP address to bind to. Default: 0.0.0.0.
	port(<number>) - UDP/TCP port used to listen messages. Default: 514.
	max-connections(n) - Limits the number of simultaneously opened connections. Default: 10.

file(filename)	Opens the specified file and reads messages.
pipe(filename)	Opens a named pipe with the specified name, and listens for messages. (You'll need to create the pipe using mkfifo command).

Table 4.2: "Source Drivers" parameters (Syslog-ng configuration)

Some Examples of Defining Sources:

1) To read from a file:

```
source <identifier> {file(filename);};
```

Example to read messages from "/temp/file1" file:

```
source file1 {file('/temp/file1');};
```

Example to receive messages from the kernel:

```
source s_kernel { file('/proc/kmsg'); };
```

2) To receive messages from local syslogd clients:

```
source sysl {unix-stream('/dev/log');};
```

3) To receive messages from remote syslogd clients:

```
source s_udp { udp(ip(<cliente ip>) port(<udp port>)); };
```

Example to listen to messages from all machines on UDP port 514:

```
source s_udp { udp(ip(0.0.0.0) port(514));};
```

Example to listen to messages from one client (IP address=10.0.0.1) on UDP port 999:

```
source s_udp_10 { udp(ip(10.0.0.1) port(999)); };
```

Step 3 - Define filters.

To define filters use this statement:

```
filter <identifier> { expression; };
```

where:

- identifier - Has to uniquely identify this given filter.
- expression - Boolean expression using internal functions, which has to evaluate to true for the message to pass.

The following internal functions are available:

Option	Description
facility (<facility code>)	Selects messages based on their facility code.
level(<level code>) or priority (<level code>)	Selects messages based on their priority.
program(<string>)	Tries to match the <string> to the program name field of the log message.
host(<string>)	Tries to match the <string> to the hostname field of the log message.
match(<string>)	Tries to match the <string> to the message itself.

Table 4.3: "Filters" parameters (Syslog-ng configuration)

Some Examples of Defining Filters:

1) To filter by facility:

```
filter f_facilty { facility(<facility name>); };
```

Examples:

```
filter f_daemon { facility(daemon); };

filter f_kern { facility(kern); };

filter f_debug { not facility(auth, authpriv, news, mail); };
```

2) To filter by level:

```
filter f_level { level(<level name>);};
```

Examples:

```
filter f_messages { level(info .. warn)};

filter f_emergency { level(emerg); };

filter f_alert { level(alert); };
```

3) To filter by matching one string in the received message:

```
filter f_match { match(‘string’); };
```

Example to filter by matching the string “named”:

```
filter f_named { match(‘named’); };
```

4) To filter ALARM messages (note that the following three examples should be one line):

```
filter f_alarm { facility(local[0+<conf.DB_facility>]) and level(info) and
match('ALARM') and match('<your string>'); } ;
```

Example to filter ALARM message with the string “kernel panic”:

```
filter f_kpanic { facility(local[0+<conf.DB_facility>]) and level(info) and
match('ALARM') and match('kernel panic'); };
```

Example to filter ALARM message with the string “root login”:

```
filter f_root { facility(local[0+<conf.DB_facility>]) and level(info) and
match('ALARM') and match('root login'); };
```

5) To eliminate sshd debug messages:

```
filter f_sshd_debug { not program('sshd') or not level(debug); };
```

6) To filter the syslog_buffering:

```
filter f_syslog_buf { facility(local[0+<conf.DB_facility>]) and
level(notice); };
```

Step 4 - Define Actions.

To define actions use this statement (note that the statement should be one line):

```
destination <identifier> {destination-driver([params]); destination-driver([param]);..};
```

where:

- identifier - Has to uniquely identify this given destination.
- destination driver - Is a method of outputting a given message.
- params - Each destination-driver may take parameters. Some of them required, some of them are optional.

The following destination drivers are available:

Option	Description
file (filename[options])	This is one of the most important destination drivers in syslog-ng. It allows you to output log messages to the named file. The destination filename may include macros (by prefixing the macro name with a '$' sign) which gets expanded when the message is written. Since the state of each created file must be tracked by syslog-ng, it consumes some memory for each file. If no new messages are written to a file within 60 seconds (controlled by the time_reap global option), it's closed, and its state is freed.

Table 4.4: "Destination Drivers" parameters (Syslog-ng configuration)

Option	Description
file (filename[options]) *continuation...*	Available macros in filename expansion: • HOST - The name of the source host where the message originated from. • FACILITY - The name of the facility the message is tagged as coming from. • PRIORITY or LEVEL - The priority of the message. • PROGRAM - The name of the program the message was sent by. • YEAR, MONTH, DAY, HOUR, MIN, SEC - The year, month, day, hour, min, sec of the message was sent. • TAG - Equals FACILITY/LEVEL. • FULLHOST - The name of the source host and the source-driver: • <source-driver>@<hostname> • MSG or MESSAGE - The message received. FULLDATE - The date of the message was sent. Available options: • log_fifo_size(number) - The number of entries in the output file. • sync_freq(number) - The file is synced when this number of messages has been written to it. • owner(name), group(name), perm(mask) - Equals global options. • template(“string”) - Syslog-ng writes the “string” in the file. You can use the MACROS in the string. • encrypt(yes/no) - Encrypts the resulting file. • compress(yes/no) - Compresses the resulting file using zlib.
pipe (filename[options])	This driver sends messages to a named pipe. Available options: owner(name), group(name), perm(mask) - Equals global options. template(“string”) - Syslog-ng writes the “string” in the file. You can use the MACROS in the string.
unix-stream(filename) and unix-dgram(filename)	This driver sends messages to a UNIX socket in either SOCKET_STREAM or SOCK_DGRAM mode.

udp("<ip address>" port(number);)
and
tcp("<ip address>" port(number);)
	This driver sends messages to another host (ip address/port) using either UDP or TCP protocol.

Table 4.4: “Destination Drivers” parameters (Syslog-ng configuration)

Option	Description
program(<program name and arguments>)	This driver fork()'s executes the given program with the arguments and sends messages down to the stdin of the child.
usertty(<username>)	This driver writes messages to the terminal of a logged-in username.

Table 4.4: “Destination Drivers” parameters (Syslog-ng configuration)

<u>Some examples of defining actions:</u>

1) To send e-mail:

```
destination <ident> { pipe(‘/dev/cyc_alarm’ template(‘sendmail <pars>’));};
```

where ident: uniquely identifies this destination. Parameters:

- *-t <name>[,<name>]* - To address
- *[-c <name>[,<name>]]* - CC address
- *[-b <name>[,<name>]]* - Bcc address
- *[-r <name>[,<name>]]* - Reply-to address
- *-f <name>* - From address
- *-s \"<text>\"* - Subject
- *-m \”<text message>\”* - Message
- *-h <IP address or name>* - SMTP server
- *[-p <port>]* - Port used. default:25

To mount the message, use this macro:

- *$FULLDATE* - The complete date when the message was sent.
- *$FACILITY* - The facility of the message.
- *$PRIORITY or $LEVEL* - The priority of the message.
- *$PROGRAM* - The message was sent by this program (BUFFERING or SOCK).
- *$HOST* - The name of the source host.
- *$FULLHOST* - The name of the source host and the source driver. Format: <source>@<hostname>
- *$MSG or $MESSAGE* - The message received.

Example to send e-mail to z@none.com (SMTP's IP address 10.0.0.2) from the e-mail address a@none.com with subject “ALARM”. The message will carry the current date, the host-name of this Cyclades AlterPath Console Server and the message that was received from the source.

```
destination d_mail1 {
     pipe('/dev/cyc_alarm'
        template('sendmail -t z@none.com -f a@none.com -s \”ALARM\” \\
           -m \'$FULLDATE $HOST $MSG\' -h 10.0.0.2'));
};
```

File Description 4.8: Send e-mail example

2) To send to pager server (sms server):

```
destination <ident> {pipe(‘/dev/cyc_alarm’ template(‘sendsms <pars>’));};
```

where ident: uniquely identify this destination

- pars: -d <mobile phone number>
- -m \’<message - max.size 160 characters>\’
- -u <username to login on sms server>
- -p <port sms - default : 6701>
- <server IP address or name>

Example to send a pager to phone number 123 (Pager server at 10.0.0.1) with message carrying the current date, the hostname of this Cyclades AlterPath Console Server and the message that was received from the source:

```
destination d_pager {
     pipe(‘/dev/cyc_alarm’
     template(‘sendsms -d 123 -m \’$FULLDATE $HOST $MSG\’ 10.0.0.1’));
};
```

File Description 4.9: To send a pager phone example

3) To send snmptrap.

```
destination <ident> {pipe(‘/dev/cyc_alarm’ template(‘snmptrap <pars>’)); };
```

where ident : uniquely identify this destination

- pars : -v 1
- <snmptrapd IP address>
- -c public : community
- \"\" : enterprise-oid
- \"\" : agent/hostname

- <trap number> : 2-Link Down, 3-Link Up, 4-Authentication Failure
- 0 : specific trap
- \"\" : host-uptime
- .1.3.6.1.2.1.2.2.1.2.1 :interfaces.iftable.ifentry.ifdescr.1
- s : the type of the next field (it is a string)
- \"<message - max. size 250 characters>\"

Example to send a Link Down trap to server at 10.0.0.1 with message carrying the current date, the hostname of this Cyclades AlterPath Console Server and the message that was received from the source:

```
destination d_trap {
pipe("/dev/cyc_alarm"
template("snmptrap -v 1 -c public 10.0.0.1 public \"\" \"\" 2 0 \"\" \\
.1.3.6.1.2.1.2.2.1.2.1 s \"$FULLDATE $HOST $MSG\" "));
};
```

File Description 4.10: Sending a link down trap

4) To write in file :

```
destination d_file { file(<filename>);};
```

Example send message to console :

```
destination d_console { file("/dev/ttyS0");};
```

File Description 4.11: Sending messages to console

Example to write a message in /var/log/messages file:

```
destination d_message { file("/var/log/messages"); };
```

File Description 4.12: Writing messages to file

5) To write messages to the session of a logged-in user:

```
destination d_user { usertty("<username>"); };
```

Example to send message to all sessions with root user logged:

```
destination d_userroot { usertty("root"); };
```

File Description 4.13: Sending messages to logged user

6) To send a message to a remote syslogd server:

```
destination d_udp { udp("<remote IP address>" port(514)); };
```

Example to send syslogs to syslogd located at 10.0.0.1 :

```
destination d_udp1 { udp("10.0.0.1" port(514)); };
```

File Description 4.14: Sending syslogs to a remote server

Connect all of the above.

Step 5 - To connect the sources, filters, and actions, use the following statement. (Actions would be any message coming from one of the listed sources. A match for each of the filters is sent to the listed destinations.)

```
log { source(S1); source(S2); ...
filter(F1);filter(F2);...
destination(D1); destination(D2);...
};
```

where :

- Sx - Identifier of the sources defined before.
- Fx - Identifier of the filters defined before.
- Dx - Identifier of the actions/destinations defined before.

Examples:

1) To send all messages received from local syslog clients to console:

```
log { source(sysl); destination(d_console);};
```

2) To send only messages with level alert and received from local syslog clients to all logged root user:

```
log { source(sysl); filter(f_alert); destination(d_userroot); };
```

3) To write all messages with levels info, notice, or warning and received from syslog clients (local and remote) to *//var/log/messages*/ file:

```
log { source(sysl); source(s_udp); filter(f_messages); destination(d_messages); };
```

4) To send e-mail if message received from local syslog client has the string “kernel panic”:

```
log { source(sysl); filter(f_kpanic); destination(d_mail1); };
```

5) To send e-mail and pager if message received from local syslog client has the string "root login":

```
log { source(sysl); filter(f_root); destination(d_mail1); destination(d_pager); };
```

6) To send messages with facility kernel and received from syslog clients (local and remote) to remote syslogd:

```
log { source(sysl); source(s_udp); filter(f_kern); destination(d-udp1); };
```

Syslog-ng Configuration to use with Syslog Buffering Feature

This configuration example uses the syslog buffering feature, and sends messages to the remote syslogd (10.0.0.1).

Step 1 - Configure ***/etc/portslave/pslave.conf*** **file parameters.**

In the *pslave.conf* file the parameters of the syslog buffering feature are configured as:

```
conf.DB_facility 1
all.syslog_buffering 100
```

File Description 4.15: portslave.conf necessary configuration

Step 2 - Add lines to ***/etc/syslog-ng/syslog-ng.conf*** **file.**

Add the following lines by vi to the file:

```
#local syslog clients
source src { unix-stream("/dev/log"); };
destination d_buffering { udp("10.0.0.1"); };

filter f_buffering { facility(local1) and level(notice); };
#send only syslog_buffering messages to remote server
log { source(src); filter(f_buffering); destination(d_buffering); };
```

File Description 4.16: portslave.conf necessary configuration

Syslog-ng Configuration to use with Multiple Remote Syslog Servers

This configuration example is used with multiple remote syslog servers.

Step 1 - Configure pslave.conf parameters.

In the *pslave.conf* file the facility parameter is configured as:

```
conf.facility 1
```

File Description 4.17: portslave.conf "facility" configuration

Step 2 - Add lines to */etc/syslog-ng/syslog-ng.conf* **file.**

```
# local syslog clients
source src { unix-stream("/dev/log"); };

# remote server 1 - IP address 10.0.0.1 port default
destination d_udp1 { udp("10.0.0.1"); };

# remote server 2 - IP address 10.0.0.2 port 1999
destination d_udp2 { udp("10.0.0.2" port(1999););};

# filter messages from facility local1 and level info to warning
filter f_local1 { facility(local1) and level(info..warn);};

# filter messages from facility local 1 and level err to alert
filter f_critic { facility(local1) and level(err .. alert);};

# send info, notice and warning messages to remote server udp1
log { source(src); filter(f_local1); destination(d_udp1); };

# send error, critical and alert messages to remote server udp2
log { source(src); filter(f_critic); destination(d_udp2); };
```

File Description 4.18: syslog-ng.conf configuration

CLI Method

To configure certain parameters for a specific serial port:

Step 1 - At the command prompt, type in the appropriate command to configure desired parameters.

To activate the serial port. <string> should be ttyS<serial port number> :

```
config configure line <serial port number> tty <string>
```

To configure conf.facility:

```
config configure conf facility <number>
```

To configure DB_facility:

```
config configure conf dbfacility <number>
```

Tip. You can configure all the conf parameters in one line:
config configure conf facility <number> dbfacility <number>

Step 2 - Activate and Save.

To activate your new configurations and save them to flash, type:

```
config write
```

(This is essentially typing signal_ras hup and saveconf from the normal terminal prompt.)

4.5 Generating Alarms (Syslog-ng)

This feature helps the administrator to manage the servers. It filters the messages received by the serial port (the server's console) based on the contents of the messages. It then performs an action, such as sending an email or pager message. To configure this feature, you need to configure filters and actions in the *syslog-ng.conf* file. (You can read more about *syslog-ng* in the Syslog-ng section.)

How to configure

Alarm generation is strictly related to the syslog-ng configuration. It is highly recommended to read the Syslog-ng section before configuring this feature. This section will show practical examples of utilization of this feature.

The */etc/portslave.conf* related parameters are:

- conf.DB_facility - This value (0-7) is the Local facility sent to the syslog-ng with data when syslog_buffering and/or alarm is active.
- all.alarm - When nonzero, all data received from the port is captured and sent to syslog-ng with INFO level and LOCAL[0+conf.DB_facility] facility. This parameter mus be set to a non zero value to activate alarm generation.

The syslog-ng reads from sources (files, TCP/UDP connections, syslogd clients), filters the messages and takes an action(writes in files, sends snmptrap, pager, email or syslogs).

Basically, alarms are triggered by a combination of sources, filters and destinations. To connect the sources, filters and actions (any message coming from one of the listed sources, matching the filters (each of them) is sent to the listed destinations). Use this statement:

```
log {   source(S1); source(S2); ...
filter(F1);filter(F2);...
destination(D1); Destination(D2);...
};
```

For more information about sources, destinations and filters, please refer to the Syslog-ng section. This

VI method - Configuration to use with Alarm Feature

This configuration example is used for the alarm feature.

Step 1 - Configure the *etc/portslave/pslave.conf* file parameter.

In the */etc/portslave/pslave.conf* file the parameters of the alarm feature are configured as:

```
all.alarm 1
conf.DB_facility  2
```

Step 2 - Configure the ***/etc/syslog-ng/syslog-ng.conf.*** **file:**

This step has the objective to configure the /etc/syslog-ng/syslog-ng.conf file. Several examples will be given here. All commands are present (commented) in the original *syslog-ng.conf* file by default. Choose the example that best fits to your application.

Example 1 - To send all messages received from local syslog clients to console. Insert the lines below at the END of the file *syslog-ng.conf* file, keeping all lines above commented.

```
source sysl {unix-stream("/dev/log");};
destination d_console { file("/dev/ttyS0");};
log { source(sysl); destination(d_console);};
```

File Description 4.19: part of the /etc/syslog-ng/syslog-ng.conf file

Example 2 - To send only messages with level alert and received from local syslog clients to all logged root user. Insert the lines below at the END of the file *syslog-ng.conf* file, keeping all lines above commented.

```
source sysl {unix-stream("/dev/log");};
filter f_alert { level(alert); };
destination d_userroot { usertty("root"); };
log { source(sysl); filter(f_alert); destination(d_userroot); };
```

File Description 4.20: part of the /etc/syslog-ng/syslog-ng.conf file

Example 3 - Write all messages with levels info, notice or warning and received from syslog clients (local and remotes) to ***/var/log/*** **messages file :** Insert the lines below at the END of the file *syslog-ng.conf* file, keeping all lines above commented.

```
source sysl {unix-stream("/dev/log");};
source s_udp { udp(ip(<ip client>) port(<udp port>)); };
filter f_messages { level(info..warn);};
destination d_message { file("/var/log/messages"); };
log { source(sysl); source(s_udp); filter(f_messages); destination(d_messages);};
```

File Description 4.21: part of the /etc/syslog-ng/syslog-ng.conf file

Example 4 - Send e-mail if message received from local syslog client has the string "kernel panic". Insert the lines below at the END of the file *syslog-ng.conf* file, keeping all lines above commented.

```
source sysl {unix-stream("/dev/log");};
filter f_kpanic{facility(local1) and level(info) and match("ALARM") and match("kernel panic");};
destination d_mail1 {
     pipe("/dev/cyc_alarm"
        template("sendmail -t z@none.com -f a@none.com -s \"ALARM\" \\
           -m \"$FULLDATE $HOST $MSG\" -h 10.0.0.2"));
     };
log { source(sysl); filter(f_kpanic); destination(d_mail1); };
```

File Description 4.22: part of the /etc/syslog-ng/syslog-ng.conf file

Example 5 - Send e-mail and pager if message received from local syslog client has the string "root login". Insert the lines below at the END of the file *syslog-ng.conf* file, keeping all lines above commented.

```
source sysl {unix-stream("/dev/log");};
filter f_root {facility(local1) and level(info) and match("ALARM") and match("root login");};
destination d_mail1 {
     pipe("/dev/cyc_alarm"
        template("sendmail -t z@none.com -f a@none.com -s \"ALARM\" \\
           -m \"$FULLDATE $HOST $MSG\" -h 10.0.0.2"));
     };
destination d_pager {
     pipe("/dev/cyc_alarm"
        template("sendsms -d 123 -m \"$FULLDATE $HOST $MSG\" 10.0.0.1"));
};
log { source(sysl); filter(f_root); destination(d_mail1); destination(d_pager); };
```

File Description 4.23: part of the /etc/syslog-ng/syslog-ng.conf file

Example 6 - Send messages with facility kernel and received from syslog clients (local and remote) to remote syslogd. Insert the lines below at the END of the file *syslog-ng.conf* file, keeping all lines above commented.

```
source sysl {unix-stream("/dev/log");};
source s_udp { udp(ip(<ip client>) port(<udp port>)); };
filter f_kern { facility(kern); };
destination d_udp1 { udp("10.0.0.1" port(514)); };
log { source(sysl); source(s_udp); filter(f_kern); destination(d-udp1); };
```

File Description 4.24: part of the /etc/syslog-ng/syslog-ng.conf file

Step 3 - Activate changes.

To activate the changes made, run the following commands in the presented order:

```
# signal_ras hup

# killall syslog-ng

# syslog-ng
```

The first command activate the changes made in the */etc/portslave/pslave.conf* file. The second and the third commands activate the changes made in the */etc/syslog-ng/syslog-ng.conf* file.

Step 4 - Save the changes to the flash memory.

To save the changes made, run the command:

```
# saveconf
```

CLI Method

The CLI method allows the configuration of some of the /etc/portslave/pslave.conf parameters. To configure certain parameters for a specific serial port:

Step 1 - At the command prompt, type in the appropriate command to configure desired parameters.

To activate the serial port. <string> should be ttyS<serial port number> :

```
# config configure line <serial port number> tty <string>
```

To configure conf.DB_facility:

```
# config configure conf dbfacility <number>
```

To configure alarm.

```
# config configure line <serial port number> alarm <number>
```

TIP. *You can configure all the parameters for a serial port in one line: config configure line <serial port number> tty <string> alarm <number>*

Step 2 - Activate and Save.

To activate your new configurations and save them to flash, type:

```
# config write
```

(This is essentially typing *signal_ras hup* and *saveconf* from the normal terminal prompt.)

4.6 Terminal Appearance

You can change the format of the login prompt and banner that is issued when a connection is made to the system. Prompt and banner appearance can be port-specific as well.

VI Method - Involved parameters and passed values

Terminal Appearance involves the following parameters in the */etc/portslave/pslave.conf* file:

Parameter	Description
all.prompt	This text defines the format of the login prompt. Expansion characters can be used here. Example value: %h login:

all.issue	This text determines the format of the login banner that is issued when a connection is made to the Cyclades AlterPath Console Server .
	\n represents a new line and \r represents a carriage return. Expansion characters can be used here.
	Value for this Example:
	`\r\n\`
	`Welcome to terminal server %h port S%p \n\`
	`\r\n`

all.lf_suppress	This activates line feed suppression. When configured as 0, line feed suppression will not be performed. When 1, extra line feed will be suppressed.
all.auto_answer_input	This parameter is used in conjunction with the next parameter, auto_answer_output. If configured and if there is no session established to the port, this parameter will constantly be compared and matched up to the string of bytes coming in remotely from the server. If a match is found, the string configured in auto_answer_output is sent back to the server. To represent the ESC character as part of this string, use the control character, ^[.
all.auto_answer_output	This parameter is used in conjunction with the previous parameter, auto_answer_input. If configured, and if there is no session established to the port, this parameter is sent back to the server when there is a match between the incoming data and auto_answer_input. To represent the ESC character as part of this string, use the control character, ^[.

Table 4.5: pslave.conf parameters for Terminal Appearance configuration

CLI Method

To configure certain parameters for a specific serial port:

Step 1 - At the command prompt, type in the appropriate command to configure desired parameters.

To activate the serial port. <string> should be ttyS<serial port number> :

```
config configure line <serial port number> tty <string>
```

To configure issue:

```
config configure line <serial port number> issue <string>
```

To configure prompt:

```
config configure line <serial port number> prompt <string>
```

To configure lf_suppress:

```
config configure line <serial port number> lf <number>
```

To configure auto_answer_input:

```
config configure line <serial port number> auto_input <string>
```

To configure auto_answer_output:

```
config configure line <serial port number> auto_output <string>
```

TIP. *You can configure all the parameters for a serial port in one line.*
config configure line <serial port number> tty <string> issue <string> prompt <string> lf <number> auto_input <string> auto_output <string>

Step 2 - Activate and Save.

To activate your new configurations and save them to flash, type:

```
config write
```

(This is essentially typing *signal_ras hup* and *saveconf* from the normal terminal prompt.)

4.7 Centralized Management

The Cyclades AlterPath Console Server allows centralized management through the use of a Master *pslave.conf* file. Administrators should consider this approach to configure multiple Cyclades AlterPath Console Server . Using this feature, each unit has a simplified *pslave.conf* file where a Master include file is cited. This common configuration file contains information for all units, properly divided in separate sections, and would be stored on one central server. This file, in our example shown in the following figure, is */etc/portslave/TScommon.conf*. It must be downloaded to each Cyclades AlterPath Console Server .

NOTE: *Centralized management can mean one big configuration file (the common file) that is placed in a management host. This same file would be downloaded into all ACS boxes (each of those boxes would include a tiny config file and that big common file). In this application, there may or may not be clustering involved. The user may want to access each box individually, without passing through la central point (master), using the common file just to make his/her life easier in regard to maintain the config file. This user could ALSO add the clustering application on a daily basis. Clustering does NOT require a common config file. A common config file does NOT apply to clustering, however, common config files can be used in an integrated manner.*

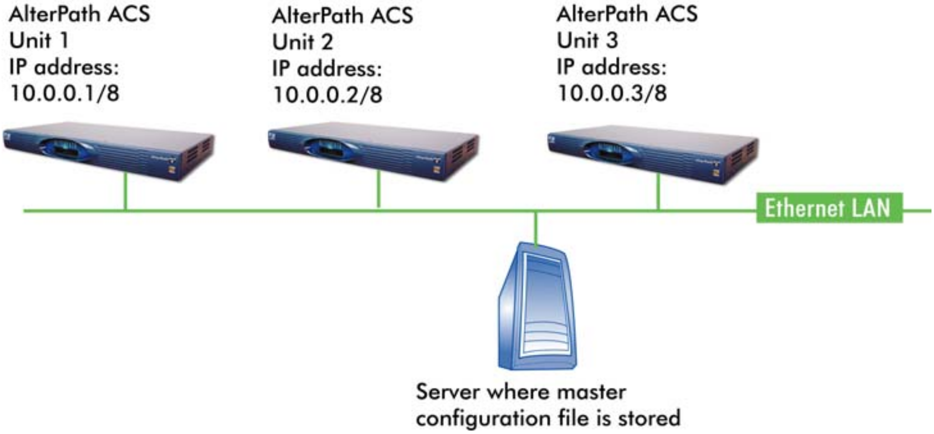


Figure 4.25 - Example of Centralized Management

VI Method - Involved parameters and passed values

The abbreviated */etc/portslave/pslave.conf* and */etc/hostname* files in each unit, for the above example are:

Unit 1 configuration:

For the */etc/hostname* file in unit 1:

```
unit1
```

File Description 4.26: Unit 1 /etc/hostname file

For the */etc/portslave/plsave.conf* file in unit 1:

```
conf.eth_ip 10.0.0.1
conf.eth_mask 255.0.0.0
conf.include /etc/portslave/TScommon.conf
```

File Description 4.27: Unit 1 /etc/portslave/portslave.conf file configuration

Unit 2 configuration:

For the */etc/hostname* file in unit 2:

```
unit2
```

File Description 4.28: Unit 2 /etc/hostname file

For the */etc/portslave/plsave.conf* file in unit 2:

```
conf.eth_ip 10.0.0.2
conf.eth_mask 255.0.0.0
conf.include /etc/portslave/TScommon.conf
```

File Description 4.29: Unit 2 /etc/portslave/portslave.conf file configuration

Unit 3 configuration:

For the */etc/hostname* file in unit 3:

```
unit3
```

File Description 4.30: Unit 3 /etc/hostname file

For the */etc/portslave/plsave.conf* file in unit 3:

```
conf.eth_ip 10.0.0.3
conf.eth_mask 255.0.0.0
conf.include /etc/portslave/TScommon.conf
```

File Description 4.31: Unit 3 /etc/portslave/portslave.conf file configuration

The common include file (located in the server) for the example is:

```
all.authtype     none
all.protocol     socket_server

conf.host_config unit1
all.socket_port  7001+
s1.tty    ttyS1
s2.tty    ttyS2
...
s16.tty    ttyS16
s17.tty    20.20.20.3:7033
s18.tty    20.20.20.3:7034
...

conf.host_config unit2
all.socket_port  7033+
s1.tty    ttyS1
s2.tty    ttyS2
...
sN.tty    ttySN

conf.host_config unit3
all.socket_port  7301+
s1.tty    ttyS1
s2.tty    ttyS2
...
sN.tty    ttySN
conf.host_config end
```

File Description 4.32: Common /etc/portslave/pslave.conf file

When this file is included, unit1 would read only the information between *conf.host_config unit1* and *conf.host_config unit2*. Unit2 would use only the information between *conf.host_config unit2* and *conf.host_config unit3* and unit3 would use information after *conf.host_config unit3* and before *conf.host_config end*.

Steps for using Centralized Configuration

Step 1 - Create and save the */etc/portslave/pslave.conf* and */etc/hostname* files in each Cyclades AlterPath Console Server .

Step 2 - Create, save, and download the common configuration.

Create and save the common configuration file on the server, then download it (probably using scp) to each unit. Make sure to put it in the directory set in the *pslave.conf* file (*/etc/portslave* in the example).

Step 3 - Execute the command signal_ras hup on each unit.

Step 4 - Test each unit.

If everything works, add the line */etc/portslave/TScommon.conf* to the */etc/config_files* file.

Step 5 - Save the file and close it.

Step 6 - Execute the *saveconf* command.

NOTE: *The included file /etc/portslave/TScommon.conf cannot contain another include file (i.e., the parameter conf.include must not be defined). Also, <max ports of Cyclades AlterPath Console Server > + N(+) is done same way as serial port.*

4.8 Date, Time and Time Zone

All configuration related to time zone adjustment is made in the */etc/TIMEZONE* file. To adjust date and time the *date* command is used.

Timezone

VI Method - Involved parameters and passed values

The content of the file */etc/TIMEZONE* can be in one of two formats. The first format is used when there is no daylight savings time in the local time zone:

```
std offset
```

The *std* string specifies the name of the time zone and must be three or more alphabetic characters. The offset string immediately follows std and specifies the time value to be added to the local time to get *Coordinated Universal Time* (UTC). The offset is positive if the local time zone is west of the Prime Meridian and negative if it is east. The hour must be between 0 and 24, and the minutes and seconds must be between 0 and 59.

The second format is used when there is daylight savings time:

```
std offset dst [offset],start[/time],end[/time]
```

There are no spaces in the specification. The initial std and offset specify the Standard Time zone, as described above. The dst string and offset specify the name and offset for the corresponding daylight savings time zone. If the offset is omitted, it defaults to one hour ahead of Standard Time.

The start field specifies when daylight savings time goes into effect and the end field specifies when the change is made back to Standard Time. These fields may have the following formats:

- *Jn* - This specifies the Julian day, with n being between 1 and 365. February 29 is never counted even in leap years.
- *n* - This specifies the Julian day, with n being between 1 and 365. February 29 is counted in leap years.
- *Mm.w.d* - This specifies day, d (0 < d < 6) of week w (1 < w < 5) of month m (1 < m < 12). Week 1 is the first week in which day d occurs and week 5 is the last week in which day d occurs. Day 0 is a Sunday.

The time fields specify when, in the local time currently in effect, the change to the other time occurs. If omitted, the default is 02:00:00.

Date and Time

The date command prints or sets the system date and time. Format of the command:

```
date MMDDhhmmCCYY
```

where:

- MM = month
- DD = day
- hh = hour
- mm = minute
- CC = century
- YY = year

For example:

```
date 101014452002
```

produces:

```
Thu Oct 10 14:45:00 DST 2002
```

The DST is because it was specified in */etc/TIMEZONE*.

Automatically adjust for Daylight Savings Time:

Here is an example of */etc/TIMEZONE* which will adjust for Central Standard Time/Central Daylight Savings Time in the USA:

```
CST+6CDST+5,M4.1.0,M10.5.0
```

Explanations:

- CST+6 : We add 6 hours to CST to get GST/GMT.
- CDST+5 : We add 5 hours to CDST to get GST/GMT.
- M4.1.0 : Month 4 (April).Week=1.Day=0 (Sunday). This is the date we switch to CDST.
- M10.5.0: Month 10 (Octorber).Week=5 (Last week).Day=0(Sunday). This is the date we switch back to CST.

Other examples:

- For EST/EDST: EST+5EDST+4,M4.1.0,M10.5.0
- For MST/MDST: MST+7MDST+6,M4.1.0,M10.5.0

- For PST/PDST: PST+8PDST+7,M4.1.0,M10.5.0

NOTE: *Remember to add an entry for /etc/TIMEZONE to /etc/config_files, if necessary, and to run the command "saveconf" to save any changes to flash.*

4.9 Session Sniffing

You can open more than one common and sniff session at the same port. For this purpose, the following configuration items are available in the file pslave.conf:

- *all.multiple_sessions* - If it is configured as no, only two users can connect to the same port simultaneously. If it is configured as yes, more than two simultaneous users can connect to the same serial port. A “Sniffer menu” will be presented to the user and they can choose either to open a sniff session; to open a read and/or write session; to cancel a connection; or to send a message to other users connected to the same serial port. If it is configured as “RW_sessions,” only read and/or write sessions will be opened, and the sniffer menu won’t be presented. If it is configured as “sniff_session” only, a sniff session will be opened, and the sniffer menu won’t be presented. Default value: no.
- *sN.multiple_sessions* - Valid only for port N. If it is not defined, it will assume the value of all.multiple_sessions.
- *all.multiuser_notif* - Multiple User notification selects if users of a certain serial port should receive a warning message every time a new user logs in or out. By default this parameter is not activated. The warning messages doesn't go to the buffering file and will be like the following example:

WARNING: *New user connected to this port.*
Current number of users: x

or

WARNING: *User disconnection from this port.*
Current number of users: x

Where x is the current number of connected users. The last user will know he/she is alone again when x = 1.

- *sN.multiuser_notif* - Valid only for port N. If it is not defined, it will assume the value of all.multiuser_notif.
- *all.escape_char* - Valid for all the serial ports; this parameter will be used to present the menus below to the user. Only characters from ‘^a’ to ‘^z’ (i.e., CTRL-A to CTRL-Z) will be accepted. The default value is ‘^z’ (CTRL-Z).
- *sN.escape_char* - Valid only for port N; this parameter will be used to present the menus below to the user. Only characters from ‘^a’ to ‘^z’ (i.e. CTRL-A to CTRL-Z) will be accepted. If it is not defined, it will assume the value of all.escape_char.

When multiple sessions are allowed for one port, the behavior of the Cyclades AlterPath Console Server will be as follows:

1. The first user to connect to the port will open a common session.

2. From the second connection on, only admin users will be allowed to connect to that port. The Cyclades AlterPath Console Server will send the following menu to these administrators (defined by the parameter *all.admin_users* or *sN.admin_users* in the file *pslave.conf*):

```
* * * ttySN is being used by (<first_user_name>) !!!
*
1 - Initiate a regular session
2 - Initiate a sniff session
3 - Send messages to another user
4 - Kill session(s)
5 - Quit

Enter your option:
```

If the user selects *1 - Initiate a regular session*, s/he will share that serial port with the users that were previously connected. S/he will read everything that is received by the serial port, and will also be able to write to it.

If the user selects *2 - Initiate a sniff session*, s/he will start reading everything that is sent and/or received by the serial port, according to the parameter *all.sniff_mode* or *sN.sniff_mode* (that can be in, out or i/o).

When the user selects *3 - Send messages to another user*, the Cyclades AlterPath Console Server will send the user's messages to all the sessions, but not to the tty port. Everyone connected to that port will see all the "conversation" that's going on, as if they were physically in front of the console in the same room. These messages will be formatted as:

`[Message from user/PID] <<message text goes here>> by the AlterPathACSTo` inform the Cyclades AlterPath Console Server that the message is to be sent to the serial port or not, the user will have to use the menu.

If the administrator chooses the option *4 - Kill session(s)*, the Cyclades AlterPath Console Server will show him/her a list of the pairs PID/user_name, and s/he will be able to select one session typing its PID, or "all" to kill all the sessions. If the administrator kills all the regular sessions, his session initiates as a regular session automatically.

Option 5 - Quit will close the current session and the TCP connection.

Only for the administrator users: Typing *all.escape_char* or *sN.escape_char* from the sniff session or "send message mode" will make the Cyclades AlterPath Console Server show the previous menu. The first regular sessions will not be allowed to return to the menu. If you kill all regular sessions using the option 4, your session initiates as a regular session automatically.

VI Method - Involved parameters and passed values

Sniffing involves the following parameters in the */etc/portslave/pslave.conf*:

- *all.admin_users* - This parameter determines which users can receive the sniff menu. When users want access per port to be controlled by administrators, this parameter is obligatory and authtype must not be none. User groups (defined with the parameter conf.group) can be used in combination with user names in the parameter list. Example values: peter, john, user_group.
- *all.sniff_mode* - This parameter determines what other users connected to the very same port (see parameter *admin_users* below) can see of the session of the first connected user (main session): *in* shows data written to the port, *out* shows data received from the port, and i/o shows both streams, whereas *no* means sniffing is not permitted.The second and later sessions are called sniff sessions and this feature is activated whenever the protocol parameter is set to *socket_ssh* or *socket_server*. Example value: *out*.
- *all.escape_char* - This parameter determines which character must be typed to make the session enter menu mode. The possible values are <CTRL-a> to <CTRL-z>. Represent the CTRL with charcter: ^. This parameter is only valid when the port protocol is *socket_server* or *socket_ssh*. Default value is ^z.
- *all.multiple_sessions* - If it is configured as *no*, only two users can connect to the same port simultaneously. If it is configured as *yes*, more than two simultaneous users can connect to the same serial port. A "Sniffer menu" will be presented to the user and they can choose either to open a sniff session; to open a read and/or write session; to cancel a connection; or to send a message to other users connected to the same serial port. If it is configured as *RW_sessions*, only read and/or write sessions will be opened, and the sniffer menu will not be presented. If it is configured as *sniff_session* only, a sniff session will be opened, and the sniffer menu won't be presented. Default value: *no*.
- *all.multiuser_notif* - Multiple User notification selects if users of a certain serial port should receive a warning message every time a new user logs in or out. By default this parameter is not activated. The warning messages doesn't go to the buffering file and will be like the following example:

```
WARNING: New user connected to this port.
Current number of users: x
```

or

```
WARNING: User disconnection from this port.
Current number of users: x
```

Where x is the current number of connected users. The last user will know he/she is alone again when x = 1.

CLI Method

To configure certain parameters for a specific serial port:

Step 1 - At the command prompt, type in the appropriate command to configure desired parameters.

To activate the serial port. <string> should be ttyS<serial port number> :

```
config configure line <serial port number> tty <string>
```

To configure admin_users:

```
config configure line <serial port number> adminusers <string>
```

To configure sniff_mode:

```
config configure line <serial port number> sniffmode <string>
```

To configure escape_char:

```
config configure line <serial port number> escape <string>
```

To configure multiple_sessions:

```
config configure line <serial port number> multiplesess <string>
```

TIP. *You can configure all the parameters for a serial port in one line.*
config configure line <serial port number> tty <string> adminusers <string> sniffmode <string> escape <string> multiplesess <string>

Step 2 - Activate and Save.

To activate your new configurations and save them to flash, type:

```
config write
```

4.10 Start and Stop Services

This feature allows daemons (services) to be enabled or disabled without need of reboot the unit. A simple engine detects configuration changes (file comparison). This feature is implemented with shell scripts. There is one main shell script called *daemon.sh* and one sourced shell script (included by *daemon.sh*) for every daemon (service) that runs in the unit. The shell script *daemon.sh* must be run once by inittab and every time a configuration change is made. The *daemon.sh* reads a file */etc/daemon_list* which contains the names of all sourced shell scripts and performs the start/stop/restart operation needed if any file related to service was changed. The *daemon.sh* will keep a hidden copy, prefixed with "." and suffixed with .tmp, of all related files in the directory */var/run*.

Each sourced shell script has a set of mandatory shell variables handled directly by the shell script *daemon.sh*. The sourced shell scripts may have other shell variables not handled directly by daemon.sh. Such variables have the sole purpose of facilitating the configuration of command line parameters.

The mandatory shell variables define:

1. If the service is enabled or disabled. (ENABLE=YES/NO)
2. The pathname to the daemon. (DNAME=<daemon name, DPATH=<daemon path>)
3. How to restart the daemon: by signal (kill, hup, term, etc) or by command (start, stop. etc). (DTYPE=sig/cmd)
4. Signal to be sent to the daemon. Default is term. (DSIG=<signal>)
5. A list of configuration files. The files in this list will be checked for changes. (ConfigFiles=<config file list>)
6. A initialization shell script that will be run before start the service. (ShellInit=<shell_script_name [command line parameters]>)
7. Command line parameters to start the daemon. (DPARM=<command line parameters>)
8. Command Line parameters to stop the daemon. (DSTOP=<command line parameters>)

The *daemon.sh* may be executed in two ways:

1. Without parameters in the command line, it will check the configuration files of the service and restart or stop it if needed.
2. It will perform the requested action (stop/restart) in the list of services given in the command line regardless any configuration changes.

The command *daemon.sh* help will display a list of services available. Currently the following services are handled by *daemon.sh*. The first column is the service ID, the second is the name of the shell script file.

DB	*/etc/cy_buffering.sh*
NET	*/etc/inetd.sh*
LOG	*/etc/syslog.sh*
SSH	*/etc/sshd.sh*
NTP	*/etc/ntpclient.conf*
SNMP	*/etc/snmpd.conf*
IPSEC	*/etc/ipsec.sh*
NIS	*/etc/ypbind.conf*
PMD	*/etc/pmd.sh*

The following example will restart power management and Data Buffering services and it will stop ssh and network timer client services :

```
# daemon.sh PMD stop SSH NTP restart DB
```

How to Configure Them

Example of sourced shell script that activates the ntpclient service (type sig).

```
# This file defines the NTP client configuration

ENABLE=NO           # Must be "NO" or "YES" (uppercase)
DNAME=ntpclient     # daemon name
DPATH=/bin          # daemon path
ShellInit=          # Performs any required initialization
ConfigFiles=        # configuration files
DTYPE=sig           # must be "sig" or "cmd" (lowercase)
DSIG=kill           # signal to stop/restart the daemon (lowercase)
                    # if it's hup term will be used to stop the daemon

# daemon command line parameters
NTPSERVER="-h 129.6.15.28"   # NTP server ip address
NTPINTERVAL="-l 300"         # Time in seconds to ask server
NTPCOUNT="-c 0"              # counter : 0 means forever
DPARM="$NTPCOUNT $NTPSERVER $NTPINTERVAL"
DSTOP=
```

File Description 4.33: /etc/ntpclient.conf file

Example of sourced shell script that activates the ipsec service (type cmd).

```
# This file defines the ipsec configuration
ENABLE=NO                   # Must be "NO" or "YES" (uppercase)
DNAME=ipsec                 # daemon name
DPATH=/usr/local/sbin       # daemon path
ShellInit=/etc/ipsec.init   # Performs any required initialization
ConfigFiles=                # configuration files
DTYPE=cmd           # must be "sig" or "cmd"
DSIG=kill           # signal to stop/restart the daemon (lowercase)
                    # if it's hup term will be used to stop the daemon
# daemon command line parameters
DPARM="setup --start"
DSTOP="setup --stop"
```

File Description 4.34: /etc/ipsec.conf file

Chapter 5
AlterPath PM integration

The AlterPath™ PM is a family of Intelligent Power Distribution Units (IPDU) that enables remote power control of servers and network gear. When used in conjunction with Cyclades console servers, the AlterPath PM delivers easier management capabilities and faster problem solving by integrating console access and power control into one single interface. This chapter approaches all configuration that is integrated with the AlterPath ACS. Below are the sections that are going to be presented in this chapter:

- Power Management
- AlterPath Firmware Upgrade
- SNMP Proxy

5.1 Power Management

The AlterPath PM is a family of intelligent power strips (IPDU - Integrated Power Distribution Units), which is used for power management. Through a serial port, the administrator can use the AlterPath PM to control all the equipment connected to its outlets, using operations like On, Off, Cycle, Lock, and Unlock.

Using the AlterPath PM and the Cyclades AlterPath Console Server together, the administrator can have full control over his data center equipment. He can, for example, reboot the data center equipment when it crashes, without leaving his console session (telnet or ssh). To do that, he must simply press a configurable hotkey and select the appropriate option from the menu displayed in the session.

Configuration

This section covers only the software configuration for the Console Server when used in conjunction with the AlterPath PM. For hardware and cabling installation instructions for the AlterPath PM, Please refer to the AlterPath PM User Guide included in the product.

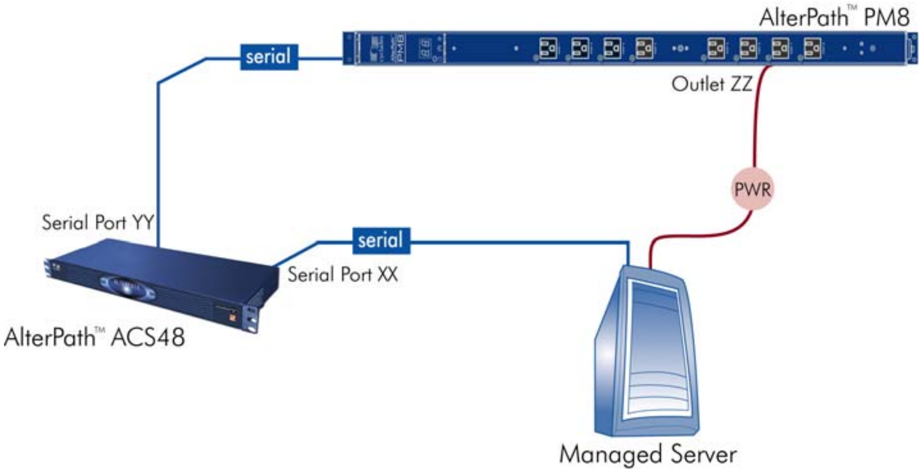


Figure 5.1 - Configuration diagram

Figure 5.1 - Configuration diagram shows a typical setup for the AlterPath PM and the AlterPath ACS. The AlterPath PM's serial console is connected to port YY of the Console Server, the server's serial console is connected to port XX of the Console Server, and the server's power plug is connected to power outlet ZZ on the AlterPath PM. These port denominations will be used in the descriptions below.

VI Method - Involved parameters and passed values

There are two different types of parameters:

1. Parameters to the port XX where the AlterPath PM is connected:

- *sXX.protocol IPDU*: New protocol Integrated Power Distribution Unit. For example: ipdu.
- *sXX.pmtype*: The IPDU manufacturer. For example: cyclades.
- *sXX.pmusers*: The user access list. For example: jane:1,2;john:3,4. The format of this field is:

```
[<username>:<outlet list>][;<username>:<outlet list>...]
```

where <outlet list>'s format is:

```
[<outlet number>|<outlet start>-<outlet end>][,<outlet number>|<outlet
start>-<outlet end>]
```

The list of users must be separated by semicolons (;); the outlets should be separated by commas (,) to indicate a list or with dashes (-) to indicate range; there should not be any spaces between the values.

- *sXX.pmNumOfOutlets*: the number of outlets of the AlterPath PM. Default: 8.

2. Parameters to the other ports where the servers are connected:

   - *all.protocol*: Protocols for the CAS profile. For example: socket_server, socket_raw, socket_ssh.
   - *all.pmkey*: The hot-key that starts a power management session. Default: ^p (Ctrl-p).
   - *sYY.pmoutlet*: The outlet list where the server YY is plugged. The outlet is passed as a pair /PM_serial_port.outlet_number/. If the server has a dual power supply, the outlets are separated by space char. For example, one power supply is plugged in the second outlet of the IPDU connected in serial port 1. The other is plugged in the third outlet of the IPDU connected in serial port 5. The value is 1.2 5.3".

SXXPMUSERS NOTES: *The ellipses in the field format for sXX.pmusers means that you can add as many users as you need. The [ ] indicates that the parameter is optional, again indicating that you can configure more than one user. The separator is the semicolon.*

How to change the IPDU Password

Step 1 - Change password using pm or pmCommand.

Step 2 - Save the configuration in the IPDU.

Step 3 - Edit the appropriate *etc/pm. config file.**

Step 4 - Restart pmd to re-read the config file.

Accessing the AlterPath PM regular menu from the Console Session

Step 1 - Open a console session.

Open a telnet or ssh session for the serial port.

Step 2 - Access the IPDU regular menu.

This should be done, for example, when the server crashes and it necessary to change the power status. Type the preconfigured hot-key.

If the user does not have permission to access any outlet, the following message will appear, and you will return to the Console Session:

```
It was impossible to start a Power Management Session
You can't access any Power Management functionality.
Please contact your Console Server Administrator.
```

If the user does not have permission to access the outlet(s) of this server, but can access another outlet, the following message will appear:

```
You cannot manage the outlet(s) of this server.
Please enter the outlet(s) (or 'h' for help):
```

The user should type the outlet(s) he wants to manage, before reaching the main menu. The main menu will appear only if the user has permission for this/these outlet(s). Typing 'h' will cause the session to show text explaining what to type, and 'l' will cause the PM session to be logged out, and the user to return to the Console Session. If the user has permission to access the outlet(s) of this server, these outlets will be managed by the PM session.

Step 3 - Regular Menu.

This is the AlterPath PM regular menu:

```
-------------------------------------------------------
Cyclades Corporation - Power Management Utility
-------------------------------------------------------

1 - Exit       2 - On      3 - Off
4 - Cycle      5 - Lock    6 - Unlock
7 - Status     8 - Help    9 - Other

Please choose an option:
```

Menu Description 5.1: AlterPath PM regular menu

Option	Description
Exit	Exits the Power Management Session.
On	Turns the outlet on.
Off	Turns the outlet off.
Cycle	Turns the outlet off and back on.
Lock	Locks the current status of the outlet.

Table 5.1: AlterPath PM regular menu options

Option	Description
Unlock	Unlocks the current status of the outlet.
Status	Shows the current status of the outlet.
Other	Allows user to control other outlets.

Table 5.1: AlterPath PM regular menu options

Step 4 - Check the status of the server's outlet or the outlet list.

Type '7' and wait for the answer. For example:

```
Please choose an option: 7

IPDU 1 Outlet 8:
Outlet Status User
8 OFF NONE
-------------------------------------------------------
Cyclades Corporation - Power Management Utility
-------------------------------------------------------
1 - Exit      2 - On      3 - Off
4 - Cycle     5 - Lock    6 - Unlock
7 - Status    8 - Help    9 - Other

Please choose an option:
```

Menu Description 5.2: Outlet List

Step 5 - Reboot the server.

If the outlet(s) is/are locked, the user must unlock the outlet(s) first (option 6 - Unlock). The Cycle command turns the power off for some seconds and the turn it on again. Type '4' and wait for the answer. For example:

```
Please choose an option: 4
IPDU 1 Outlet 8:
8: Outlet power cycled.

-------------------------------------------------------
Cyclades Corporation - Power Management Utility
-------------------------------------------------------
1 - Exit      2 - On      3 - Off
4 - Cycle     5 - Lock    6 - Unlock
7 - Status    8 - Help    9 - Other

Please choose an option:
```

Menu Description 5.3: Outlet list

Step 6 - Change the outlet list.

If the user needs to access another outlet(s) which can be managed by him, the option 9 - Other should be used. For example:

```
-------------------------------------------------------
Cyclades Corporation - Power Management Utility
-------------------------------------------------------

1 - Exit 2 - On 3 - Off
4 - Cycle 5 - Lock 6 - Unlock
7 - Status 8 - Help 9 - Other

Please choose an option: 9
Please enter the outlet(s) (or 'h' for help): 1.2

-------------------------------------------------------
Cyclades Corporation - Power Management Utility
-------------------------------------------------------

1 - Exit 2 - On 3 - Off
4 - Cycle 5 - Lock 6 - Unlock
7 - Status 8 - Help 9 - Other

Please choose an option:
```

Menu Description 5.4: Changing the outlet list

From this point, all the commands will be related to the 2nd outlet of the IPDU in the port 1.

Step 7 - Return to the Console Session.

The user can exit from the PM session and return to the Console Session in three ways:

1.Type the hot-key again, any time.

2.If the session is waiting for a menu option, type the option 1 - Exit.

3.If the session is waiting for the outlet, type 'l'.

When the user leaves the PM session, the following message will appear:

```
Exit from PM session
```

Power Management for Authorized Users

The administrator or any user that belongs to the pmusers group, can log onto the Console server itself, and have total control over all the IPDU outlets. An additional menu, with more options than the regular menu, is provided for the administrator and users contained in the pmusers group to manage any IPDU.

There are two commands which can be used to manage the IPDU. The first one (pm) deals with menu options, while the second one (pmCommand) deals with the commands as they are sent to the IPDU, and requires more knowledge about the AlterPath-PM commands.

Adding an user of the pmusers group

Only the root user and users belonging to the pmusers group can do power management by using the pm or pmCommand. To add an user as member of the pmusers group, log in as "root" and run the 'adduser' command with the following syntax:

```
# adduser -G pmusers <username>
```

Changing the group of an already existing user

It is also possible to change the group of an already existing user. In this example we will change the groups of the already existing users: “cyclades” and “test”. To do that follow the steps below:

Step 1 - Open the file ***/etc/group.***

To open this file, run the command:

```
# vi /etc/group
```

Step 2 - Addind the “cyclades” and “test” users to the pmusers group.

To change the group of these users, look for the line that begins with “pmusers”. At the end of this line, just after the ´:´ character, insert the “cyclades” and “test”users.

```
webadmin::504:root
pmusers::505:cyclades,test
cyclades:x:506:
test:x:507:
```

Step 3 - Save the configuration.

To save the changes done, run the command:

```
# saveconf
```

pm command

There are two ways to use this command: menu interface or command line. The menu is reached by typing the following command, from the prompt:

```
# pm <IPDU port>
```

```
For example:

-----------------------------------------------------------------
Cyclades Corporation - Power Management Utility
-----------------------------------------------------------------

1 - Exit             2 - Help               3 - On
4 - Off              5 - Cycle              6 - Lock
7 - Unlock           8 - Status             9 - Password
10 - Alarm           11 - Syslog            12 - Buzzer
13 - Current         14 - Save              15 - Version

Please choose an option: 2

Exit                - Exits the Power Management session
Help                - Shows this message
On                  - Turn outlet(s) ON
Off                 - Turn outlet(s) OFF
Cycle               - Turn outlet(s) OFF and back ON
Lock                - Lock the current status of outlet(s)
Unlock              - Unlock the current status of outlet(s)
Status              - Show the current status of outlet(s)
Password            - Set a password for the specific user
Alarm               - Set alarm threshold for current
Syslog              - Turn syslog on or off
Buzzer              - Turn buzzer on or off
Current             - Show current consumption for the entire unit
Save                - Save configuration and status
Version             - Displays version information
```

Menu Description 5.5: pm command options

Some of these options require the outlet number (On, Off, Cycle, Lock, Unlock, Status), and others don't. In the first case, when the option is selected, the number of the outlet will be asked. The user can enter one or more outlets (separated by commas or dashes), or "all," to apply the option to all the outlets.

Following are examples of some things which can be done through this command.

Turning the outlet off

```
Please choose an option: 3
Please enter the outlets (or 'help' for help): 4

4: Outlet turned off.

-------------------------------------------------------
Cyclades Corporation - Power Management Utility
-------------------------------------------------------
1 - Exit 2 - On 3 - Off
4 - Cycle 5 - Lock 6 - Unlock
7 - Status 8 - Help 9 - List
10 - Current

Please choose an option:
```

Menu Description 5.6: Turning the outlet off

Locking the outlets

When the outlet is locked, the previous status cannot be changed, until the outlet is unlocked. This means that if the outlet was on, it cannot be turned off and, if it was off, it cannot be turned on.

```
Please choose an option: 5
Please enter the outlets (or 'help' for help): 1-3


1: Outlet locked.
2: Outlet locked.
3: Outlet locked.

-------------------------------------------------------
Cyclades Corporation - Power Management Utility
-------------------------------------------------------
1 - Exit      2 - On         3 - Off
4 - Cycle     5 - Lock       6 - Unlock
7 - Status    8 - Help       9 - List
10 - Current

Please choose an option:
```

Menu Description 5.7: Locking the outlet

Retrieving the status of the outlets

```
Please choose an option: 7
Please enter the outlets (or 'help' for help): all


Outlet Status User
1 Locked OFF NONE
2 Locked OFF NONE
3 Locked OFF NONE
4 OFF NONE
5 OFF NONE
6 OFF NONE
7 OFF NONE
8 OFF NONE

-------------------------------------------------------
Cyclades Corporation - Power Management Utility
-------------------------------------------------------
1 - Exit      2 - On        3 - Off
4 - Cycle     5 - Lock      6 - Unlock
7 - Status    8 - Help      9 - List
10 - Current

Please choose an option:
```

Menu Description 5.8: Turning the outlet off

The second way to use the pm command is the command line. In this case, the syntax for the command is

```
# pm <IPDU port> <option> [<outlet(s)>],
```

where:

- *<option>* is the name of the option, as written in the menu.
- *<outlet(s)>* is the number of the outlet(s) the option will be applied to, and is used only if the option requires the outlet.

In the examples above, the same result can the achieved by using the command line mode:

Turning the outlet off

```
[root@TSx000 /root]# pm 1 Off 4
4: Outlet turned off.
[root@TSx000 /root]#
```

Locking the outlets

```
[root@TSx000 /root]# pm 1 Lock 1-3
1: Outlet locked.
2: Outlet locked.
3: Outlet locked.
[root@TSx000 /root]#
```

Retrieving the status of the outlets

```
[root@TSx000 /root]# pm 1 Status all
Outlet Status User
1 Locked OFF NONE
2 Locked OFF NONE
3 Locked OFF NONE
4 OFF NONE
5 OFF NONE
6 OFF NONE
7 OFF NONE
8 OFF NONE
[root@TSx000 /root]#
```

pmCommand command

Through *pmCommand* command, the administrator has access to other options beyond the menu options, because he will be accessing the IPDU itself. The administrator must have a good knowledge of the AlterPath PM command set to use it.

There are two ways to use this command. If only the IPDU port is passed as an argument, it will appear in a prompt where the administrator can write the command. Otherwise, the arguments after the IPDU port will be considered the PM command.

Syntax:

```
pmCommand <IPDU port> [<command>]
```

For example:

```
[root@CAS root]# pmCommand 1
You're entering the "Power Management Prompt".
To go back to the Console Server's command line type: exitPm

[Cyclades - Power Management Prompt]#
```

The following are examples of some things which can be done through this command.

Listing the commands available for the AlterPath PM

```
[Cyclades - Power Management Prompt]# help
```

- *on <outlet><cr>* - Turn <outlet> ON
- *off <outlet><cr>* - Turn <outlet> OFF
- *cycle <outlet><cr>* - Turn <outlet> OFF and back ON
- *lock <outlet><cr>* - Lock the current status of <outlet>
- *unlock <outlet><cr>* - Unlock the current status of <outlet>
- *status <outlet><cr>* - Show the current status of <outlet>
- *list<cr>* - List users created and eventual outlets assigned
- *exit<cr>* - Exit session
- *passwd <user><cr>* - Set a password for the specific user
- *help<cr>* - Show supported commands
- *current<cr>* - Show the instantaneous current consumption for the entire unit
- *adduser <username><cr>* - Add user to the DB (8 maximum users allowed)
- *deluser <username><cr>* - Delete user from the DB
- *assign <outlet> <username><cr>* - Assign <outlet> to a specific user
- *name <outlet> <name><cr>* - Name an outlet

Cycling all the outlets

```
[Cyclades - Power Management Prompt]# cycle all

1: Outlet power cycled.
2: Outlet power cycled.
3: Outlet power cycled.
4: Outlet power cycled.
5: Outlet power cycled.
6: Outlet power cycled.
7: Outlet power cycled.
8: Outlet power cycled.

[Cyclades - Power Management Prompt]#
```

Unlocking the outlets 1, 5 and 8

```
[Cyclades - Power Management Prompt]# unlock 1, 5, 8

1: Outlet unlocked.
5: Outlet unlocked.
8: Outlet unlocked.
```

Retrieving the status of all outlets

```
[Cyclades - Power Management Prompt]# status all

Outlet          Name               Status               Users
1                                  Unlocked ON
2                                  Unlocked ON
3                                  Unlocked ON
4                                  Unlocked ON
5                                  Unlocked ON
6                                  Unlocked ON
7                                  Unlocked ON
8                                  Unlocked ON
```

Turning the outlet off

```
[Cyclades - Power Management Prompt]# off 2

2: Outlet turned off.
```

5.2 AlterPath Firmware Upgrade

It is possible to upgrade the firmware of the IPDU unit connected to any serial port of the AlterPath ACS. It is also possible to upgrade the whole daisy-chain of AlterPath PM units, since the unit(s) before the targeted one has firmware version 1.2.2 or greater.

Upgrade Process

To upgrade the firmware of the PM units follow the steps below:

Step 1 - Download the firmware.

The first step of the upgrade process will be the download of the new firmware. Cyclades provides a directory on its ftp site where it is possible to check for new firmwares and download them to the AlterPath ACS. It is recommended to download the new firmware to the /tmp directory because files in this directory are deleted during the boot process.

Step 2 - Run the pmfwupgrade application.

After downloading it is necessary to call an application called pmfwupgrade. This application has the following syntax:

```
# pmfwupgrade [-h] [-f] [-F] [-v] <serial port number>[:<unit number>] <filename>
```

where :

- *-h* = Show the help message and exit
- *-f* = The upgrade is done without asking any questions
- *-F* = The upgrade is done without waiting logical connection with the AlterPath PM. This is should be used after possible power failure during the upgrade process.
- *-v* = show messages about the status of the upgrade.
- *<serial port number>* = the serial port where the PM unit is connected
- *[:<unit number>]* = number of the PM unit when in daisy-chain. If is not used, all units in the serial port will have the firmware upgraded, when possible
- *<filename>* = complete path of the file that has the PM firmware (default: */tmp/pmfirmware*)

IMPORTANT! *If the AlterPath PM unit is not configured with the default password, it will be necessary to inform it to the AlterPath ACS by editing the /etc/pm.cyclades file and changing the parameter admPasswd with the correct password.*

The pmfwupgrade application will try to stop all the process that are using the serial port. Just type YES to proceed into the upgrade process. Another message will prompt asking for confirmation to proceed with the upgrade process. Type 'y' to upgrade the PM unit firmware.

WARNING! *Depending on the hardware version of the AlterPath PM, it is possible that all outlets completely powers off during the upgrade process. Make sure to shutdown all devices connected to them before starting the firmware upgrade process.*

5.3 SNMP Proxy

The SNMP Proxy for Power management feature allows the Cyclades ACS console servers to proxy SNMP requests to the Cyclades Intelligent Power Distribution Units. This allows SNMP clients to query and control the remote IPDU using standard set and get commands.

How to Configure

You should ensure that the AlterPath PM is correctly installed and configured by following the procedure outlined in section 5.1 Power Management of this Reference Guide. You must also ensure that SNMP is correctly configured by following the configuration instructions in the SNMP section of Chapter 4 - Administration.

The parameters and features that can be controlled in the remote IPDU are as follows:

- The number AlterPath PM units connected to a given console server
- The number of the outlets connected to a given port
- The number the AlterPath PM units connected to this port (when a daisy chain configuration is being used).
- The instantaneous RMS current being drawn from each of the AlterPath PM unit(s) connected to this port.
- The software version of the AlterPath PM unit(s) connected to this port
- The temperature of the AlterPath PM unit(s) connected to this port
- The name of the outlet as configured in the AlterPath PM.
- The alias of the server that is configured as using this outlet
- The name of the serial console connection that corresponds to the host which this outlet controls power.
- The status of the outlet
  . power status : 0 (off), 1 (on), 3 (unknow)
  . lock state : 0 (unlock), 1 (lock) , 2 (unknow)

This feature will allow the user to control the AlterPath PM outlets using SNMP set commands. These following actions will be allowed to each outlet by this feature :

1) ON
2) OFF
3) CYCLE
4) LOCK

IMPORTANT! *The AlterPath ACS proxies all SNMP requests to the AlterPath PM unit. Therefore there is a small delay if an outlet cycling is requested by the snmpset command. To sucessfully cycle an outlet, a 4 second or higher timeout must be specified. To run this command for more than one outlet or for units configured as daisy chain, this time should be recalculated.*

Examples:

This feature allows the user do these following SNMP requests:

1) Get the number of ACS/TS serial ports that has PM connected to:

```
# snmpget -m all -v 2c -t 4 -c cyclades 10.10.0.1 .cyNumberOfPM <enter>
enterprises.cyclades.cyACSMgmt.cyPM.cyNumberOfPM.0 = 2
```

2) Get the number of outlets of the PM connected to serial port 16:

```
# snmpget -m all -v 2c -t 4 -c cyclades 10.10.0.1 .cyPMNumberOutlets.16 <enter>
enterprises.cyclades.cyACSMgmt.cyPM.cyPMtable.cyPMEntry.cyPMNumberOutlets.16 = 8
```

3) get the number of units of the PM connected to serial port 14:

```
# snmpget -m all -v 2c -t 4 -c cyclades 10.10.0.1 .cyPMNumberUnits.14 <enter>
enterprises.cyclades.cyACSMgmt.cyPM.cyPMtable.cyPMEntry.cyPMNumberUnits.14 = 2
```

For more examples and MIB definition please search the online FAQ at:
www.cyclades.com/support/faqs.php

This page has been left intentionally blank.

Chapter 6
PCMCIA Cards Integration

PCMCIA slots allow for enhanced functionality with support for many interface cards, such as Ethernet, modem (V.90, CDMA, GPRS, GSM and ISDN) and wireless LAN.

6.1 Supported Cards

The following cards are supported by the ACS:

10/100BT Ethernet Cards:

Table 6.1: Supported 10/100 PCMCIA Ethernet Cards

Brand	Model	ACS Firmware
Linksys	EtherFast 10/100 PC Card Model PCM100 ver.3	2.1.0 or higher
Linksys	EtherFast 10/100 PC Card ver.4	2.1.5 or higher
D-Link	EtherFast 10/100 PC Card Model DFE-670TXD	2.1.6 or higher

802.11b Wireless Ethernet:

Table 6.2: Supported 802.11b Wireless Ethernet Cards

Brand	Model	ACS Firmware
Linksys	WPC11 ver.3. NOTE: The ver.4 is NOT a PCMCIA card. It is a Cardbus, so it is NOT supported by ACS	2.1.5 or higher

V.90 (56k) Modem:

Table 6.3: Supported V.90 modem cards

Brand	Model	ACS Firmware
Xircom	XM5600 56K PC Card Modem Adapter	2.1.5 or higher

Table 6.3: Supported V.90 modem cards

Brand	Model	ACS Firmware
Zoom	Modem V.92 PC Card Plus Model 3075	2.1.0 or higher

ISDN:

Table 6.4: Supported ISDN cards

Brand	Model	ACS Firmware
AVM	Fritz! Card PCMCIA	2.1.1 or higher
Sedlbauer	Sedlbauer ISDN card	2.1.5 or higher

GSM:

Table 6.5: Supported GSM cards

Brand	Model	ACS Firmware
Novatel Wireless *	Merlin G201	2.1.4 or higher
Sierra Wireless *	AirCard 750	2.1.4 or higher

* WARNING: Consult with your local GSM service provider for coverage areas and support of this card prior to using it with the AlterPath™ ACS or ACS1.

Compact Flash:**

Table 6.6: Compact Flash cards

Brand	Model	ACS Firmware
Hama	64MB CF Memory	2.1.5 or higher
Kingston	128MB CF Memory	2.1.5 or higher
SanDisk	16MB CF Memory	2.1.5 or higher
Aved	16MB CF Memory	2.1.5 or higher

Table 6.6: Compact Flash cards

Brand	Model	ACS Firmware
Other	Most other adapters and compact flash should also work but have not been verified by Cyclades	

** In order to load a Compact Flash card on the AlterPath™ ACS or ACS1, use a PCMCIA Compact Flash adapter.

IDE Hard Disks:

Table 6.7: IDE Hard Disks

Brand	Model	ACS Firmware
Toshiba	MK5002MPL 5GB	2.1.5 or higher

Tools for Configuring and Monitoring PCMCIA Devices

During the ACS boot, the */etc/init.d/pcmcia* script loads the PCMCIA core drivers and the cardmgr daemon. The cardmgr daemon is responsible for monitoring PCMCIA sockets, loading client drivers when needed, and running user-level scripts in response to card insertions and removals.

- *lsmod* - This command shows the modules loaded for the PCMCIA devices.
- *cardctl* - This command can be used to check the status of a socket, or to see how it is configured. Just type cardctl to see the syntax of the command. cardctl config displays the card configuration. cardctl ident can be used to get card identification information. cardctl eject stops the application and unloads the client driver, and cardctl insert re-loads the driver and re-starts the application.

NOTE: *cardctl suspend, cardctl resume and cardctl reset are not supported.*

Ejecting Cards

You can insert the card anytime, and the drivers should be loaded automatically. But you will need to run cardctl eject before ejecting the card to stop the application using the card. Otherwise the ACS may hang during the card removal. You must specify the slot number when using the cardctl command. For example:

```
cardctl eject 0 for the lower slot
```

and

```
cardctl eject 1 for the upper slot
```

NOTE: *Due to a known problem in the current release, the I/O ports used by the card cannot be re-used after card re-insertion. In each card insertion, the card gets a different I/O port. This limits the number of times the card can be ejected and inserted. When all the I/O ports known by the card are used, the "**RequestIO: No more items**" message is displayed, and the only way to reset the I/O port usage is to reboot the system.*

6.2 PCMCIA Network Configuration

The onboard Ethernet device has the *eth0* name. The first PCMCIA Ethernet card or wireless LAN card detected will receive the *eth1* name, the second card will be *eth2*.
cardmgr will read the network settings from the */etc/network/interfaces* and assign an IP to *eth1*.

NOTE: *Before changing the /etc/network/interfaces file, unload the network client driver using cardctl eject.*

The factory default */etc/network/interfaces* file has the following lines:

```
# auto eth1
#iface eth1 inet static
#     address 192.168.0.42
#     network 192.168.0.0
#     netmask 255.255.255.0
#     broadcast 192.168.0.255
#     gateway 192.168.0.1
```

File Description 6.1: part of the /etc/network/interfaces file

Remove the # in the beginning of the line, and change the IPs to suit your network configuration. For instance, you may want the following configuration:

```
auto eth1
iface eth1 inet static
     address 192.168.162.10
     network 192.168.162.0
     netmask 255.255.255.0
     broadcast 192.168.162.255
     gateway 192.168.162.1
```

File Description 6.2: part of the /etc/network/interfaces file

Don't forget to run *saveconf* to save this configuration in the flash, so that it can be restored in the next boot. Run *cardctl* insert to load the network drivers with the new configuration.

NOTE: *Do not use ifconfig to change the network settings for the PCMCIA device. Otherwise, you may be unable to unload the network driver during cardctl eject and the Cyclades AlterPath ACS may hang. The correct way is to change the /etc/network/interfaces file.*

Wireless LAN PC Cards

First do the appropriate PCMCIA network configuration. Additionally, the configuration of the wireless driver is done in the following file:

/etc/pcmcia/wireless.opts

For instance, to configure the network name as *MyPrivateNet*, and the *WEP* encryption key as *secu1*, the following settings could be added to the default “*,*,*,*)" entry :

```
 *,*,*,*)
     INFO="This is a test"
     ESSID="MyPrivateNet"
     KEY="s:secu1"
```

File Description 6.3: part of the /etc/pcmcia/wireless.opts file

NOTE: *The "s:" prefix in the KEY line indicates that the key is an ASCII string, as opposed to hex digits. Five characters or ten digits could be entered for WEP 40-bit and 13 characters or 26 digits could be entered for WEP 128-bit.*

There is a generic sample in the end of the *wireless.opts* file that explains all possible settings. For more details in wireless configuration, search for *manpage iwconfig* on the Internet. The parameters in *wireless.opts* are used by the *iwconfig* utility. After changing any of the parameters, run *cardctl* eject followed by *cardctl insert* to load the new settings. Also, run saveconf to save the new settings to flash. *iwconfig eth1*shows the basic wireless parameters set in *eth1*. *iwlist* allows to list frequencies, bit-rates, encryption, etc. The usage is:

```
iwlist eth1 frequency
iwlist eth1 channel
iwlist eth1 ap
iwlist eth1 accesspoints
iwlist eth1 bitrate
iwlist eth1 rate
iwlist eth1 encryption
iwlist eth1 key
iwlist eth1 power
iwlist eth1 txpower
iwlist eth1 retry
```

Modem PC Cards

The modem device gets the */dev/ttySn* name, where n is the number of embedded serial devices plus 1. For instance, if the ACS has 32 onboard serial devices, the modem card becomes the */dev/ttyS33*.

When a modem card is detected, *cardmgr* starts a script which loads mgetty for the modem device automatically. mgetty provides the login screen to the remote user. mgetty may also be configured to start PPP (pppd) and let PPP login the caller. The steps to allow PPP connections are:

Step 1 - Enable login and PAP authentication in */etc/mgetty/login.config*.

Enable the desired authentication in */etc/mgetty/login.config*. For instance, you may want the following authentication in */etc/mgetty/login.config* to enable PAP and system password database authentication:

```
/AutoPPP/ - a_ppp /usr/local/sbin/pppd auth -chap +pap login nobsdcomp nodeflate
```

Step 2 - Create a user name in /etc/ppp/pap-secrets.

If +pap authentication was selected, create a user name in */etc/ppp/pap-secrets*. For instance, you may add the following line:

```
"mary" * "marypassword" *
```

Step 3 - Create the user for login in the Radius server.

If the login option was used, create the user either locally (by running *adduser*) or create the user in the Radius server for Radius authentication.

When the login option is used, */etc/pam.conf* may also need to be changed. (By default, */etc/pam.conf* has the ppp and login services configured for local authentication. You will have to change them if you want Radius authentication. More information can be found in Chapter 2 - Authentication in the Linux-PAM section.

Step 4 - Copy the */etc/ppp/options.ttyXX* as */etc/ppp/options.ttyS33* (the modem port).

Copy the */etc/ppp/options.ttyXX* to have the device name assigned to the pcmcia modem. For instance, if the modem is the *ttyS33*, */etc/ppp/options.ttyXX* should be copied as */etc/ppp/options.ttyS33*. If you are not sure which *ttySxx* is the modem device, do a "*ls -al /dev/modem*" with the modem inserted.

Step 5 - Uncomment local and remote IPs in */etc/ppp/options.ttyS33*.

Uncomment the line that assigns the local and remote IPs in */etc/ppp/options.ttyS33* (or whatever is the tty name in your system). For instance, you may want to assign 192.168.0.1 for local ip, and 192.168.0.2 for the remote ip.

Step 6 - Save ***/etc/ppp/options.ttyS33*** **in flash.**

Step 7 - Create an entry in ***/etc/config_files.***

It should have the name of the file you created, so that the new file can be saved to the flash. For instance, you will have to add a line with */etc/ppp/options.ttyS33* in */etc/config_files*.

Step 8 - Run saveconf to save the files listed in ***/etc/config_files*** **to the flash.**

Step 9 - Insert the pcmcia modem if not inserted yet.

Step 10 - Run ps to see that mgetty is running.

The ACS is ready to receive dial in calls.

Step 11 - Establish PPP connection with the ACS.

From the remote system, use pppd to dial and establish a PPP connection with the ACS. The remote system should have the login user name set in their */etc/ppp/pap-secrets* to have a successful login in the Cyclades AlterPath ACS.

Establishing a Callback with your Modem PC Card

Setting up a callback system serves two purposes:

1. Cost savings: reversing line charges - allows your company to call you back.
2. Security: makes sure users are who they pretend to be by calling a well-known or preconfigured number back.

The steps to allow callback are divided into two parts. Part One is the configuration for the Advanced Secure Console Port ServerACS (Server Side ACS Setup). Part Two is the configuration for the client side.

Server Side ACS Setup.

Step 1 - Enable authentication.

Enable the desired authentication in */etc/mgetty/login.config*. For instance, you may want the following authentication in */etc/mgetty/login.config* to enable PAP and system password database authentication:

```
/AutoPPP/ - a_ppp /usr/local/sbin/pppd auth -chap +pap login

nobsdcomp nodeflate
```

Step 2 - Configure a pseudo callback user.

Add the following line to */etc/mgetty/login.config* with the appropriate values. At the end of the file there is a line, like the presented below:

```
*       -        -        /bin/login @
```

Do this before the above presented line:

```
<pseudo callback name>-   -  /sbin/callback -S <phone number of the client>
```

eg.:

```
call   -     -        /sbin/callback -S 12345
```

Where, 'call' is the pseudo callback name and '123456' is the number to dial back.

NOTE:
*1. The order of configuration in /etc/mgetty/login.config matters. By default, it has the line *   -   -   /bin/login @ at the end of the file. This line allows any users to log in and be verified by the login program. If you were to add the callback line after this line, the callback program will not be initiated when you try logging in. Instead, the login program will be used to verify you since it was encountered first. List the callback users first if you want the option of having some users access the callback program and the others the login program.*

```
        call      -      -      /sbin/callback -S 12345
        call2     -      -      /sbin/callback -S 77777
*                 -      -      /bin/login @
```

The example above will allow you to have the option whether or not you want to use the callback functionality. If you log in with call or call2, then callback starts immediately. If you log in as anybody else other than call or call2, callback will not start and you'll be verified by the login program.

*2. Don't use * instead of some callback user name. Mgetty will fall to infinite callback.*

3. If you don't specify a telephone number, callback will ask for a number after you log in as the pseudo callback user.

Step 3 - If you plan to login through PPP with PAP authentication create pap user name in ***/etc/ppp/pap-secrets.***

Add a line similar to the following: (include the quotes and the two asterisks).

```
"myUserName"    *     "myUserNamePassword"*
```

Step 4 - If you plan to login through PPP follow steps 4 - 9 in the section above on Modem PC Cards.

Step 5 - Create users.

Step A: Create a new user with the command adduser myUserName.

This will create an entry in */etc/passwd* that resembles this:

```
myUserName:$1$/3Qc1pGe$./h3hzkaJQJ/:503:503:Embedix
User,,,:/home/myUserName:/bin/sh
```

Step B: If you want to limit myUserName to getting ONLY PPP access and NOT shell access to the server, edit the entry for myUserName in */etc/passwd*.

Do this by replacing */bin/sh* with a pathname to a script that you will be creating later. In the following example, the script is: */usr/ppp/ppplogin*

```
myUserName:$1$/3Qc1pGe$./h3hzkaJQJ/:503:503:Embedix
User,,,:/home/myUserName:/usr/ppp/ppplogin
```

Step 6 - If you executed Step 5b, create the ppp login script.

Step A: Create a script called /etc/ppp/ppplogin following this format:

```
#!/bin/sh
exec /usr/local/sbin/pppd <ppp options>
```

Step B: Make script executable.

Type chmod 755 */etc/ppp/ppplogin*.

Step C: Save this file to flash.

Save this file to flash so the next time the ACS gets rebooted, you won't lose the new file. Add */etc/ppp/ppplogin* into */etc/config_files*.
Now execute *saveconf*.

Step 7 - Change permission of pppd.

Type chmod u+s */usr/local/sbin/pppd*

TIP. *To prevent from always having to manually change permission every time your ACS reboots:*
1. Edit /etc/users_scripts by uncommenting the following line:

/bin/chmod_pppd

2. Add /etc/users_scripts into /etc/config_files.
3. Execute saveconf. The next time the ACS reboots, this change will be in effect. You should not need to manually change the pppd permission.

Step 8 - Your ACS is ready to establish a callback connection.

See Client Side Setup to start the callback connection.

Client Side Setup.

Step 1 - Activate Show Terminal Window option.

(From Win2000) Go to your Connection window (the window to dial the ACS) -> Properties -> Security -> look for Interactive Logon and Scripting -> click on Show Terminal Window.

Step 2 - Disable/enable encryption protocols.

If you are going to be using PPP connection with PAP authentication, make sure you disable all other encryption protocols.
(from Win2000) go to your Connection window (the window to dial the ACS) -> Properties -> Security -> click on Advanced (custom settings) -> click on Settings -> click on Allow these protocols -> disable all protocols except the PAP one.

Step 3 - Set up modem init string.

It is very important that before callback hangs the call, the modem in the Windows box does not tell Windows that the call has been dropped. Otherwise, Windows Dial-up Networking will abort everything (because it thinks the call was dropped with no reason).
(From Win2000) Go to Windows' control panel -> Phone and Modem -> Modems -> choose your modem -> Properties -> Advanced -> add &c0s0=1 to Extra Settings.

Step 4 - Call your ACS.

Step A - Dial to the ACS modem using either the normal username or the ppp username that you created in Step 5 when configuring the server side.

Step B - Once a connection is made, you get a login prompt.

Step C - Login with the pseudo callback name to start the callback.

Step D - Your connection gets dropped. The ACS is now calling you back.

Step E - After reconnection to you, you get a login prompt again.

Step F - Now you can:

- Log in through character mode: Log in with username and password. You will get the ACS shell prompt.
- Log in through ppp: Click on Done on the Terminal Window.

6.3 GSM Card Configuration

This works for Firmware 2.1.3 and up. To configure a GSM PCMCIA card follow the steps below:

Step 1 - In *etc/mgetty/mgetty.config*, add this entry:

```
port ttyS2
   data-only y
   init-chat "" \d\d\d+++\d\d\dATZ OK
```

Where *ttyS2* may have to be changed to the serial port that will be assigned to the GSM card, eg. replace *ttyS2* by *ttyS9* for an ACS8.

Step 2 - If the SIM card needs a PIN, edit *etc/pcmcia/serial.opts*. Uncomment the line

```
INITCHAT="- \d\d\d+++\d\d\datz OK at+cpin=1111 OK"
```

and replace '1111' by the PIN.

Step 3 - Add '*/etc/mgetty/mgetty.config*' to */etc/config_files* and call *saveconf*:

```
# echo /etc/mgetty/mgetty.config >> /etc/config_files
# saveconf
```

Insert the card. The card should flash red first. After the PIN is sent, the LED stays red, until the card found network. It then flashes green.

Sierra Aircard 750 and Merlin G201 GSM PCMCIA card

Both are GSM cards that allows wireless connections over the GSM network. It works very similar to a modem card.

Description of the Feature. This feature implements support for the Sierra Aircard 750/Merlin G201 cards. When inserted, it will be detected automatically, and the modules loaded, just like all other PCMCIA cards that are supported.

How it works. When the card is inserted, cardmgr automatically loads the *serial_cs* module. The script */etc/pcmcia/serial* is started, reading options from */etc/pcmcia/serial.opts*. It starts an mgetty process on the serial device. Because the CIS information given by the card is wrong, a CIS file is supplied with correct parameters.

How to configure it. The configuration is the same as for a PCMCIA modem card. Additionally, if the PIN is not disabled on the SIM card (some providers do not allow this), it must be configured in */etc/pcmcia/serial.opts* with an entry like:

```
# one time init chat (example: PIN number for GSM card):
# important: give '-' to expect nothing, instead of '' or ""
INITCHAT="- \d\d\d+++\d\d\datz OK at+cpin=1111 OK"
```

File Description 6.4: part of the /etc/pcmcia/serial.opts file

This will ensure that the PIN number will be given when the card is inserted.

6.4 ISDN PC Cards

You can establish synchronous PPP connections with ISDN cards. The ipppd is the daemon that handles the synchronous PPP connections.

How to configure dial in.

Step 1 - Create a user.

Create a user in */etc/ppp/pap-secrets* or in */etc/ppp/chap-secrets*, depending if you want PAP or CHAP authentication. You will also have to create a user in */etc/ppp/pap-secrets* if you want radius or local authentication. In case you don't want to repeat all the user database from the radius server an option is to use '*' as the user in */etc/ppp/pap-secrets*:

```
*       *       ""      *
```

Step 2 - Change the options in /etc/pcmcia/isdn.opts to fit your environment.

Make sure that *$DIALIN* is set to "yes." Set the desired authentication in *DIALIN_AUTHENTICATION*. For instance, "*+pap*" for PAP, "*+chap*" for CHAP, "*login auth*" or "*login +pap*" for radius, "*login auth*" or "*login +pap*" for local. When "*login auth*" or "*login +pap*" are used, PAM libraries are used so */etc/pam.conf* should be also configured.

Step 3 - Run saveconf to save your changes to the flash.

Step 4 - If the ISDN card is not inserted, it is time to insert the card.

ipppd is started automatically. Go to step 6.

Step 5 - Restart script.

If the card was already inserted, you will need to restart the isdn script to reload any changed configuration. To restart the script, issue:

```
# /etc/pcmcia/isdn stop ippp0
# /etc/pcmcia/isdn start ippp0
```

Step 6 - You can dial from the remote system to the ACS, and get a PPP connection.

Step 7 - To hang up the connection from the ACS side, just issue:

```
# isdnctrl hangup ippp0
```

How to configure dial out.

Step 1 - Create a user.

Create a user in */etc/ppp/pap-secrets* or in */etc/ppp/chap-secrets*,depending if you want PAP or CHAP authentication.

Step 2 - Change options.

Change the options in */etc/pcmcia/isdn.opts* to fit your environment. Make sure that *$DIALIN* is set to "no". Set *$USERNAME* to the user name provided by your ISP.

Step 3 - Run saveconf to save your changes to the flash.

Step 4 - If the ISDN card is not inserted, it is time to insert the card.

ipppd is started automatically. Go to step 6.

Step 5 - Restart script.

If the card was already inserted, you will need to restart the isdn script to reload any changed configuration. To restart the script, issue:

```
# /etc/pcmcia/isdn stop ippp0
# /etc/pcmcia/isdn start ippp0
```

Step 6 - To dial out, issue the command:

```
# isdnctrl dial ippp0
```

Step 7 - To hangup the connection from the ACS side, just issue:

```
# isdnctrl hangup ippp0
```

Establishing a Callback with your ISDN PC Card

For the same cost saving reasons explained in Establishing a Callback with your Modem PC Card, the ISDN card in the ACS can be configured to callback client machines after receiving dial in calls.

The steps to allow callback are divided into two parts. Part One is the configuration for the ACS (ACS Setup) as callback server. Part Two is the configuration of a Windows 2000 Professional computer as callback client.

ACS setup (Callback Server).

Step 1 - Change the parameters in */etc/pcmcia/isdn.opts* **to fit your environment.**

Step 2 - Set the callback number in *DIALOUT_REMOTENUMBER***:**

```
DIALOUT_REMOTENUMBER="8358662" # Remote phone that you want to dial to
```

Step 3 - If your isdn line supports caller id, it is recommended that you also configure the *DIALIN_REMOTENUMBER* and enable secure calls. Otherwise skip to step 4.

```
DIALIN_REMOTENUMBER="8358662"   #Remote phone from which you will receive
                                # calls

SECURE="on"                     # "on" = incoming calls accepted only if
                                # remote phone matches DIALIN_REMOTENUMBER;
                                # "off" = accepts calls from any phone. "on"
                                # will work only if your line has the caller
                                # id info.
```

Step 4 - Make sure the CALLBACK is set to “in” in *\/etc/pcmcia/isdn.opts* file.

```
CALLBACK="in"         # "in" will enable callback for incoming calls.
```

Step 5 - Uncomment line with user "mary" in *\/etc/ppp/pap-secrets*.

Step 6 - Save changes to flash.

```
# saveconf
```

Step 7 - Activate the changes by stopping and starting the isdn script:

```
# /etc/pcmcia/isdn stop ippp0
# /etc/pcmcia/isdn start ippp0
```

The ACS side is done.

Windows 2000 Professional configuration (Callback Client).

Step 1 - Create user “mary” with password “marypasswd” in Control Panel-> "User and Passwords".

Step 2 - Create a dial-up connection that uses “Modem - AVM ISDN Internet (PPP over ISDN) (AVMISDN1)”.

(To create a dial-up connection, go to Start->Settings->Network and Dial-up Connections->Make New Connection, select “I want to set up my Internet connection manually, or I want to connect through a local area network”, select “I connect through a phone line and a modem”, select the “AVM ISDN Internet (PPP over ISDN)” modem, type the phone number you dial to connect to the ACS, and enter mary as User name and marypasswd as password.). After creating this dial-up, click on the Properties of this dial-up, select the “Options” panel, and change the Redial attempts to 0.

Step 3 - Accept incoming connections.

To accept incoming connections, go to Start->Settings->Network and Dial-up Connections->Make New Connection, select “Accept incoming connections” (the words are slightly different in XP), select AVM ISDN Internet (PPP over ISDN), select “Do not allow virtual private connections”, click the user “mary”, then click on Properties of TCP/IP to specify the IP addresses for the calling computers. Also in “mary” Properties, select the Callback tab and make sure the option “Do not allow callback” is selected. After any change in the Incoming Connection Properties, it is recommended that the Windows is rebooted to apply the changes.

The Windows side is done.

Now you can dial from Windows to the ACS. Go to Start-> Settings-> “Network and Dial-up Connections” and select the dial-up that you created. After the “Dialing” message, you will see a window with a warning message:

```
Opening port....
Error 676: The phone line is busy.
```

Just click Cancel. In a few seconds, the ACS will call you back, and you will see the connection icon in the taskbar.

Establishing a Callback with your ISDN PC Card (2nd way)

The previous section explained how to do callback at D-Channel level. The advantages of having callback at D-Channel level is that it works independent of the Operating System on the client side. But a big disadvantage is that the callback call happens before the authentication phase in PPP. The only security is by that only calls from predefined phone numbers are accepted.

To fix that drawback, this section explains another way to have callback with the ACS. The steps described here will work when the remote side is a UNIX machine, not Windows. The callback call will happen after the PPP authentication is successful.

ACS Setup (Callback Server).

Step 1 - Change the parameters in *etc/pcmcia/isdn.opts* file to fit your environment.

Step A - Set the callback number in DIALOUT_REMOTENUMBER.

```
DIALOUT_REMOTENUMBER="8358662"  # Remote phone that you want to dial to
```

Step B - Configure the DIALIN_REMOTENUMBER.

If your ISDN line supports caller id, it is recommended that you also configure the DIALIN_REMOTENUMBER and enable secure calls. Otherwise skip to Step C.

```
DIALIN_REMOTENUMBER="8358662"   # Remote phone from which you will receive
                                # calls
SECURE="on"                     # "on" = incoming calls accepted only if
                                # remote phone matches DIALIN_REMOTENUMBER;
                                # "off" = accepts calls from any phone. "on"
                                # will work only if your line has the caller
                                # id info.
```

Step C - Set the desired IPs for local and remote machines.

Step D - Set DIALIN to “yes”.

```
DIALIN="yes"          # "yes" if you want dial in, "no" if you want dial out
```

Step E - Make sure the CALLBACK parameter is disabled.

```
CALLBACK="off"        # "off" = callback disabled.
```

Step F - Add the user that will callback the client in DIALIN_AUTHENTICATION.

```
DIALIN_AUTHENTICATION="auth login user mary"
```

Step 2 - Make sure /etc/pam.conf has the configuration you want (e.g., radius).

This step is only required if you are using “auth login” in DIALIN_AUTHENTICATION. When using “auth login,” /etc/pam.conf is what defines which authentication will be used.

Step 3 - Add the user "mary" in *\/etc/ppp/pap-secrets*.

Step 4 - Uncomment lines in */etc/ppp/auth-up*.

Step 5 - Save changes to flash:

```
# saveconf
```

Step 6 - Activate the changes by stopping and starting the isdn script:

```
# /etc/pcmcia/isdn stop ippp0
# /etc/pcmcia/isdn start ippp0
```

Linux (Callback Client).

Step 1 - Configure the ipppd to have user mary and pap authentication.

Step 2 - Dial to the ACS:

```
# isdnctrl dial ippp0
```

Step 3 - As soon the ACS authenticates the user mary, the ACS will disconnect and callback.

6.5 Media Cards

Media cards (compact flashs, hard drives) are small memory cards with a capacity up to 1GB. They can be used like a normal hard disk drive using IDE. Using an adapter, CF cards can be used in PCMCIA slots. Such an adapter is very cheap, because the PCMCIA and CF card standard are the same, only the pin layout and the socket is different. On the market, there are also small PCMCIA hard drives available, eg. a drive from Toshiba with a capacity of 5GB (Toshiba MK5002MPL). CF card support can be used in the ACS for storing files. This would be especially useful for example to save the configuration. CF cards cannot be rewritten indefinitely. For this reason, CF should not be used for logging.

For data buffering, a PCMCIA hard drive is ideal:

- data will not be lost on power loss / crash / reboot of the CAS.
- no dependency on an NFS server that may fail.

Description of the Feature

When inserting an adapter with a CF card or a PCMCIA hard drive, an ide device appears. This can be mounted, eg. by:

```
# mkdir /mnt/ide
# mount /dev/hda1 /mnt/ide
```

Apart from the ext2 filesystem, the VFAT filesystem will be supported. This makes it easy to exchange data with a Windows system. To create a vfat filesystem, the it is possible to run the utility *mkdosfs* .

To initialize a card with VFAT, do:

```
# echo ",,0x0e" | sfdisk /dev/hda
# mkdosfs /dev/hda1
```

for ext2 filesystem, do:

```
# echo ",,L" | sfdisk /dev/hda
```

The "*mke2fs*" utility, is the system creator for ext2 filesystems, and can be run like the following example:

```
# mke2fs /dev/hda1
```

In addition, an utility to create or partition the CF has been added. For this, the program sfdisk will be used. sfdisk can be easily used for scripting, so it can be called from the prompt shell.

To check an ext2 or vfat filesystem, the utility fsck has been added.

```
# fsck -t <ftype> /dev/<hdxx>
```

How it works

When the card is inserted, *cardmgr* loads the ide-cs module, which depends on *ide-mod.o*. This in turn loads *ide-probe-mod.o*, which recognizes the CF as a disk, and *ide-disk.o* will be loaded. From this point on, the partitions (usually one) can be mounted using mount. If the filesystem is vfat, the modules *fat.o* and *vfat.o* will be loaded.

Configuration

Step 1 - Insert the card.

Step 2 - Automatic compact flash mounting.

The compact flash will mount automatically because by default, the parameter *DO_MOUNT* is set to YES in the */etc/pcmcia/ide.opts* file. Below is an example of the file:

```
# ATA/IDE drive adapter configuration
# The address format is "scheme,socket,serial_no[,part]".
#
# For multi-partition devices, first return list of partitions in
# $PARTS.  Then, we'll get called for each partition.

case "$ADDRESS" in
*,*,*,1)
     #INFO="Sample IDE setup"
     DO_FSTAB="y" ; DO_FSCK="n" ; DO_MOUNT="y"
     FSTYPE="vfat"
     #OPTS=""
     MOUNTPT="/mnt/ide"
     [ -d $MOUNTPT ] || mkdir $MOUNTPT
     ;;
*,*,*)
     PARTS="1"
     # Card eject policy options
     NO_CHECK=n
     NO_FUSER=n
     ;;
esac
```

File Description 6.5: /etc/pcmcia/ide.opts file

These parameters can be changed:

- *DO_FSTAB* - If set to 'y', an entry in */etc/fstab* will be created. This parameter defaults to "n" if not mentioned in the */etc/pcmcia/ide.opts* file.
- *DO_FSCK* - A boolean (y/n) setting. Specifies if the filesystem should be checked before being mounted. This parameter defaults to “n” in the */etc/pcmcia/ide.opts* file.
- *DO_MOUNT* - If set to ‘y’, the card will be mounted automatically upon insertion. This parameter defaults to ‘n’ if not mentioned in the */etc/pcmcia/ide.opts* file.
- *FS_TYPE* - Can be either 'vfat' or 'ext2'. Determines the filesystem type.
- *MOUNTPT* - The mount point where the partition will be mounted.
- *NO_CHECK/NO_FUSER* - Boolean (y/n) settings for card eject policy. If *NO_CHECK* is true, then “cardctl eject” will shut down a device even if it is busy. If *NO_FUSER* is true, then the script will not try to kill processes using an ejected device. These parameters defaults to "n" if not mentioned in the */etc/pcmcia/ide.opts* file.
- *PARTS* - A list of partitions to be mounted. The conf file will be called again for each partition. In the example above, there is an entry only for partition '1', but you can eg. set PARTS="1 3 4" and add entries for the case statement like:

```
*,*,*,3)
# settings for partition 3
;;
*,*,*,4)
# settings for partition 4
;;
```

To give different configuration for slot 0 and 1, the second parameter in the case statement can be used. For example:

```
*,0,*,1)
# settings for slot 0
;;
*,1,*,1)
# settings for slot 1
;;
```

Step 3 - Save the configuration.

To save any configuration done in the */etc/pcmcia/ide.opts* file is necessary to run the command:

```
# saveconf
```

WARNING: *Before removing the media pcmcia card from the Cyclades AlterPath Console Server you MUST run “cardctl eject”, otherwise data might not be correctly written to disk and result in corruption of the media. Correct operation of the Cyclades AlterPath Console Server is not guaranteed if eject is not executed.*

Chapter 7
Profile Configuration

This chapter begins with a table containing parameters common to all profiles, followed by tables with parameters specific to a certain profile. You can find samples of the pslave configuration files (pslave.conf, .cas, .ts, and .ras) in the */etc/portslave* directory in the Cyclades AlterPath Console Server box.

Then all possible profiles (CAS, TS and RAS) and the necessary parameters that need to be configured in the */etc/portslave/pslave.conf* file. This chapter includes the following sections:

- The pslave.conf file
- Examples for configuration testing

7.1 The pslave.conf file

This is the main configuration file (*/etc/portslave/pslave.conf*) that contains most product parameters and defines the functionality of the Cyclades AlterPath Console Server .

There are three basic types of parameters in this file:

- conf.* parameters are global or apply to the Ethernet interface.
- all.* parameters are used to set default parameters for all ports.
- s#.* parameters change the default port parameters for individual ports.

An all.* parameter can be overridden by a s#.* parameter appearing later in the *pslave.conf* file (or vice-versa).

TIP. *You can do a find for each of these parameters in vi, once you open this file by typing:*
/ <your string>
To search the file downward for the string specified after the /.

pslave.conf common parameters

The tables below will present all parameters with their respective descriptions. The first table will present parameters that are common for any profile. The second, third and fourth tables will approach specific parameters for CAS, TS and Dial in profiles respectively.

Parameter	Description	
conf.dhcp_client	It defines the dhcp client operation mode. Valid values: 0 - DHCP disabled 1 - DHCP active 2 - DHCP active and the unit saves in flash the last IP assigned by the DHCP server (default).	1 Also see Description column.
conf.eth_ip_alias	Secondary IP address for the Ethernet interface (needed for clustering feature).	209.81.55.10
conf.eth_mask_alias	Mask for the secondary IP address above.	255.255.255.0
conf.rlogin	It defines the location of rlogin utility *Note: This is a parameter specific to TS profile.*	Eg.: /bin/rlogin
conf.facility	The local facility sent to syslog-ng from PortSlave.	1-7
conf.group	Used to group users to simplify the configuration of the parameter all.users later on. This parameter can be used to define more than one group.	group_name: user1, user2
conf.eth_ip	Configured in Chapter 3. This is the IP address of the Ethernet interface. This parameter, along with the next two, is used by the *cy_ras* program to OVERWRITE the file */etc/network/ifcfg_eth0* as soon as the command "*signal_ras hup*" is executed. The file */etc/network/ifcfg_eth0* should not be edited by the user unless the *cy_ras* configuration is not going to be used.	200.200.200.1
conf.eth_mask	The mask for the Ethernet network.	255.255.255.0

Table 7.1: /etc/portslave/pslave.conf common parameters

Parameter	Description	
conf.eth_mtu	The Maximum Transmission Unit size, which determines whether or not packets should be broken up.	1500
conf.lockdir	The lock directory, which is */var/lock* for the the Cyclades AlterPath Console Server . It should not be changed unless the user decides to customize the operating system.	*/var/lock*
all.dcd	DCD signal (sets the tty parameter CLOCAL). Valid values are 0 or 1. If *all.dcd=0*, a connection request will be accepted regardless of the DCD signal and the connection will not be closed if the DCD signal is set to DOWN. If *all.dcd=1* a connection request will be accepted only if the DCD signal is UP and the connection will be closed if the DCD signal is set to DOWN.	0
all.users	Restricts access to ports by user name (only the users listed can access the port or, using the character "!", all but the users listed can access the port.) In this example, the users joe, mark and members of user_group cannot access the port. A single comma and spaces/tabs may be used between names. A comma may not appear between the "!" and the first user name. The users may be local, Radius or TacacsPlus. User groups (defined with the parameter conf.group) can be used in combination with user names in the parameter list. Notice that these are common users, not administrators.	! joe, mark, user_group

all.issue	This text determines the format of the login banner that is issued when a connection is made to the Cyclades AlterPath Console Server . \n represents a new line and \r represents a carriage return. Expansion characters can be used here. Value for this example:
	`\r\n\`
	`Welcome to terminal server %h port S%p \r\n\`
		See Description column

all.prompt	This text defines the format of the login prompt. Expansion characters can be used here.	%h login:

Table 7.1: /etc/portslave/pslave.conf common parameters

Parameter	Description	
all.media	It defines media type RS232/RS484 and operation mode half/full duplex. Valid values for all products : • rs232 - RS232 (default value). • rs232_half - RS232 with RTS legacy half duplex • rs232_half_cts - RS232 with RTS legacy half duplex and CTS control Valid values for the ACS1 only : • rs485_half - RS485 half duplex with out terminator • rs485_half_terminator - RS485 half duplex with terminator • rs485_full_terminator - RS485 full duplex with terminator • rs422 - alike rs485_full_terminator	See Description column
all.netmask	It defines the network mask for the serial port.	255.255.255.255
all.mtu	It defines the maximum transmit unit	1500
all.mru	It defines the maximum receive unit	1500
all.sysutmp	It defines whether portslave must write login records. Valid values are yes or no.	yes/no
all.syswtmp	It defines whether portslave must write login records.	yes/no
all.pmtype	Name of the IPDU manufacturer.	cyclades
all.pmusers	List of the outlets each user can access	1-3
all.pmkey	The hotkey that identifies the power management command.	^p
all.pmNumOfOutlets	The number of outlets you have on the AlterPath PM.	8

Table 7.1: /etc/portslave/pslave.conf common parameters

Parameter	Description	

all.sttyCmd	The TTY is programmed to work as configured and this user-specific configuration is applied over that serial port. Parameters must be separated by a space. The following example sets :
	`-igncr`
	This tells the terminal not to ignore the carriage-return on input,
	`-onlcr`
	Do not map newline character to a carriage return or newline character sequence on output,
	`opost`
	Post-process output,
	`-icrnl`
	Do not map carriage-return to a newline character on input.
	`all.sttyCmd -igncr -onlcr opost -icrnl`	

all.utmpfrom	It allow the administrator to customize the field "FROM" in the login records (utmp file). It is displayed in the "w" command. Eg.: "%g:%P.%3.%4" %g : process id %P : Protocol %3 : Third nibble of remote IP %J : Remote IP Note: In the pslave.conf file there is a list of all expansion variables available.	See Description Column
all.radnullpass	It defines whether the access to users with null password in the radius server must be granted or not.	yes/no
all.speed	The speed for all ports.	9600
all.datasize	The data size for all ports.	8
all.stopbits	The number of stop bits for all ports.	1
all.parity	The parity for all ports.	none

Table 7.1: /etc/portslave/pslave.conf common parameters

Parameter	Description	
all.authhost1	This address indicates the location of the Radius/TacacsPlus authentication server and is only necessary if this option is chosen in the previous parameter. A second Radius/TacacsPlus authentication server can be configured with the parameter all.authhost2.	200.200.200. 2
all.accthost1	This address indicates the location of the Radius/TacacsPlus accounting server, which can be used to track how long users are connected after being authorized by the authentication server. Its use is optional. If this parameter is not used, accounting will not be performed. If the same server is used for authentication and accounting, both parameters must be filled with the same address. A second Radius/TacacsPlus accounting server can be configured with the parameter all.accthost2.	200.200.200. 2
all.authtype	Configured in Chapter 2, Section 2.1 - Device Authentication. Type of authentication used. There are several authentication type options: • **none** (no authentication) • **local** (authentication is performed using the */etc/passwd* file) • **remote** (This is for a terminal profile only. The unit takes in a username but does not use it for authentication. Instead it passes it to the remote server where it is then used for authentication.) • **radius** (authentication is performed using a Radius authentication server) • **TacacsPlus** (authentication is performed using a TacacsPlus authentication server) • **ldap** (authentication is performed against an ldap database using an ldap server. The IP address and other details of the ldap server are defined in the file */etc/ldap.conf*)	local

Table 7.1: /etc/portslave/pslave.conf common parameters

Parameter	Description	
all.authtype *continuation...*	• **kerberos** (authentication is performed using a kerberos server. The IP address and other details of the kerberos server are defined in the file */etc/krb5.conf*) • **local/radius** (authentication is performed locally first, switching to Radius if unsuccessful) • **radius/local** (the opposite of the previous option) • **local/TacacsPlus** (authentication is performed locally first, switching to TacacsPlus if unsuccessful) • **TacacsPlus/local** (the opposite of the previous option) • **RadiusDownLocal** (local authentication is tried only when the Radius server is down) • **TacacsPlusDownLocal** (local authentication is tried only when the TacacsPlus server is down) • **kerberosDownLocal** (local authentication is tried only when the kerberos server is down) • **ldapDownLocal** (local authentication is tried only when the ldap server is down) • **NIS** - All authentication types but NIS follow the format *all.authtype* <Authentication> DownLocal or <Authentication> (e.g. *all.authtype radius* or *radiusDownLocal* or *ldap* or *ldapDownLocal*, etc). NIS requires *all.authtype* to be set as local, regardless if it will be "*nis*" or its "*Downlocal*" equivalent. The service related to "*nis*" or its "*Downlocal*" equivalent would be configured in the */etc/nsswitch.conf* file, not in the */etc/portslave/pslave.conf* file. Note that this parameter controls the authentication required by the Cyclades AlterPath Console Server . The authentication required by the device to which the user is connecting is controlled separately.	
all.break_sequence	This parameter is the string that is used to send a break to the TTY. It is only valid if TTY protocol is socket_ssh or socket_server.	~break
all.break_interval	This parameter defines the break duration in milliseconds. It is valid if TTY protocol is socket_ssh,socket_server or ssh-2 (client).	

Table 7.1: /etc/portslave/pslave.conf common parameters

Parameter	Description	
all.radtimeout	This is the timeout (in seconds) for a Radius/TacacsPlus authentication query to be answered. The first server (authhost1) is tried “radretries” times, and then the second (authhost2), if configured, is contacted “radretries” times. If the second also fails to respond, Radius/TacacsPlus authentication fails.	3
all.radretries	Defines the number of times each Radius/ TacacsPlus server is tried before another is contacted. The default, if not configured, is 5.	5
all.secret	This is the shared secret necessary for communication between the Cyclades AlterPath Console Server and the Radius/TacacsPlus servers.	secret
all.flow	This sets the flow control to hardware, software, or none.	hard

Table 7.1: /etc/portslave/pslave.conf common parameters

Parameter	Description	

all.protocol	Defines the protocol used to connect with the Cyclades AlterPath Console Server . For each profile there are some valid values:
	• CAS profile:
	- *socket_server* when telnet is used.
	- *socket_ssh* when ssh v1 or v2 is used.
	- *raw_data* to exchange data in transparent mode. It is similar to “socket_server” mode, but without telnet negotiation, breaks to serial ports, etc.
	• TS profile:
	- *login* requests username and password.
	- *rlogin* receives username from the Cyclades AlterPath Console Server and requests a password.
	- *telnet*
	- *ssh*
	- *ssh2*
	- *socket_client*
	If the protocol is configured as telnet or socket_client the socket_port parameter needs to be configured.
	• RAS profile: *slip*, *cslip*, *ppp*, *ppp_only*
	• Power Management: *ipdu*
	• Serial Printer : *lpd*
	• Billing profile: *billing*
	
	<u>ACS1 only:</u>
	• Automation profile: "*modbus*", in this case, serial mode can be *ascii* or *rtu*. To enable "*modbus*" it is necessary to uncomment the related line as shown below:
	`# Modbus/TCP Protocol`
	`modbus stream  tcp     nowait.1000`
	`root    /bin/modbusd    modbusd`
	Then, enable it by running the command:
	`daemon.sh restart NET`
	• PPP over leased lines (only authentication PAP/CHAP): “*ppp_only*”
	• PPP with terminal post dialing (Auto detect PPP): “*ppp*”
		socket_server

Table 7.1: /etc/portslave/pslave.conf common parameters

Parameter	Description	
all.web_WinEMS	Defines whether or not management of Windows Emergency Management Service is allowed from the Web.	yes or 1, or no or 0
all.xml_monitor	A non-zero value activates XML monitoring. All XML data received from the port is captured and sent to syslog-ng with facility LOCAL<DB_facility> and priority INFO. The format of the message is "XML_MONITOR (ttySx) [data]". XML tags are sent by Windows Server 2003 Emergency Management Services during boot or crash. You can read more on XML_MONITOR in: */etc/syslog-ng/syslog-ng.conf*	1
all.translation	Defines whether or not to perform translation of Fn-keys (e.g. F8 key) from one terminal type to VT-UTF8. Currently only translation from xterm to VT-UTF8 is supported.	xterm
sX.pmoutlet	sX indicates the serial port number to which the PM hardware is connected. The pmoutlet part of the parameter indicates the outlet # on the PM hardware that manages the server/network equipment in question.	8
s1.tty	The device name for the port is set to the value given in this parameter. If a device name is not provided for a port, it will not function.	ttyS1

Table 7.1: /etc/portslave/pslave.conf common parameters

pslave.conf CAS (Console Access Server) parameters

You can configure additional CAS features with the parameters given on the following tables. (The Figure X: CAS diagram with various authentication methods is used as an example in some parameters.

In addition to the above parameters which are common to all local and remote access scenarios, you can also configure the following parameters for additional options. Many of the parameters are unique to CAS, but some also apply to TS and Dial-in port profiles. These are going to be indicated in the appropriate instances.

Parameter	Description	Example
conf.nfs_data_buffering	This is the Remote Network File System where data captured from the serial port will be written instead of being written to the local directory */var/run/DB*. The directory tree to which the file will be written must be NFS-mounted, so the remote host must have NFS installed and the administrator must create, export and allow reading/writing to this directory. The size of this file is not limited by the value of the parameter all.data_buffering, though the value cannot be zero since a zero value turns off data buffering. The size of the file is dependent on the NFS server only (hard drive, partition size, etc.).	commented
conf.DB_facility	This value (0-7) is the Local facility sent to the syslog with the data when syslog_buffering is active. The file */etc/syslog-ng/syslog-ng.conf* contains a mapping between the facility number and the action (see more on Section 4.4, “Syslog-ng,” on page 119).	0
conf.nat_clustering_ip	IP address of any Cyclades AlterPath Console Server interface (master box). It is a public IP address (e.g. Ethernet's interface IP address) and it is the one that must be used to connect the slave's serial ports. You can use the same value assigned to the Ethernet's IP address as that of the master box in the chain.	64.186.161.108
all.ipno	This is the default IP address of the Cyclades AlterPath Console Server 's serial ports. The “+” indicates that the first port should be addressed as 192.168.1.101 and the following ports should have consecutive values. Any host can access a port using its IP address as long as a path to the address exists in the host's routing table.	192.168.1.101+
all.netmask	It defines the network mask for the serial port.	255.255.255.255

Table 7.2: CAS specific parameters for the pslave.conf file

Parameter	Description	Example
all.DTR_reset	This parameter specifies the behavior of the DTR signal in the serial port. If set to zero the DTR signal will be ON if there is a connection to the serial port, otherwise OFF. If set from 1 to 99 the DTR signal will be always ON. A value greater or equal 100 specifies for how long (in milliseconds) the DTR signal will be turned off before it is turned back on again when a connection to the serial port is closed.	100
all.break_sequence	This parameter is the string that is used to send a break to the TTY. It is only valid if TTY protocol is socket_ssh or socket_server.	~break
all.break_interval	This parameter defines the break duration in miliseconds. It is valid if TTY protocol is *socket_ssh*.	socket_server or ssh-2 (client)
all.modbus_smode	Communication mode through the serial ports. This parameter is meaningful only when modbus protocol is configured. The valid options are ascii (normal TX/RX mode) and rtu (some time constraints are observed between characters while transmitting a frame). If not configured, ASCII mode will be assumed.	commented
all.lf_suppress	This can be useful because telneting (from DOS) from some OS such as Windows 98 causes produces an extra line feed so two prompts appear whenever you press Enter. When set to 1, line feed suppression is active which will eliminate the extra prompt. When set to 0 (default), line feed suppression is not active.	0

Table 7.2: CAS specific parameters for the pslave.conf file

Parameter	Description	Example
all.auto_answer_input	This parameter works in conjunction with *all.auto_answer_output*. It allows you to configure a string that will be matched against all data coming in from the tty (remote server). If there is a match, the configured output string (*auto_answer_output*) will then be send back to the tty. This parameter works only when there is no session to the port. If uncommented and a string of bytes is set, matching occurs whenever there is not session established to the port. If this parameter is commented out, then no checking and matching occurs.	commented
all.auto_answer_output	This parameter works in conjunction with *all.auto_answer_input*. It allows you to configure a string that is sent back to the remote server whenever the incoming data remote server matches with *all.auto_answer_input*. This parameter works only when there is no session to the port. If this parameter is commented, then nothing will be sent back to the remote server even if *all.auto_answer_input* is uncommented. If this parameter is uncommented and if *all.auto_answer_input* is also uncommented, then the string configured will be sent back to the remote server.	commented
all.poll_interval	Valid only for protocols *socket_server* and *raw_data*. When not set to zero, this parameter sets the wait for a TCP connection keep-alive timer. If no traffic passes through the Cyclades AlterPath Console Server for this period of time, the Cyclades AlterPath Console Server will send a line status message to the remote device to see if the connection is still up. If not configured, 1000 ms is assumed (the unit for this parameter is ms). If set to zero, line status messages will not be sent to the socket client.	0

Table 7.2: CAS specific parameters for the pslave.conf file

Parameter	Description	Example
all.socket_port	In the CAS profile, this defines an alternative labeling system for the Cyclades AlterPath Console Server ports. The "+" after the numerical value causes the serial interfaces to be numbered consecutively. In this example, serial interface 1 is assigned the port value 7001,serial interface 2 is assigned the port value 7002, etc. One example on how this could be used is in the case of all.protocol or s<n>.protocol socket_ssh and the port value (7001, 7002, etc), if supplied by the ssh client like username:port value, the ssh client will be directly connected with the serial interface.	7001+
all.data_buffering	A non zero value activates data buffering (local or remote, according to what was configured in the parameter *conf.nfs_data_buffering* see Section 1.2, "Data Buffering," on page 10 in Chapter 1). If local data buffering, a file is created on the Cyclades AlterPath Console Server ; if remote, a file is created through NFS in a remote server. All data received from the port is captured in this file. If local data buffering, this parameter means the maximum file size (in bytes). If remote, this parameter is just a flag to activate (greater than zero) or deactivate data buffering. When local data buffering is used, each time the maximum is reached the oldest 10% of stored data is discarded, releasing space for new data (FIFO system) - circular file. When remote data buffering is used, there's no maximum file size other than the one imposed by the remote server - linear file. This file can be viewed using the normal Unix tools (cat, vi, more, etc.). **Size is in bytes not kilobytes**. See Data Buffering for details.	0

Table 7.2: CAS specific parameters for the pslave.conf file

Parameter	Description	Example
all.DB_mode	When configured as cir for circular format, the buffer works like a revolving file at all times. The file is overwritten whenever the limit of the buffer size (as configured in *all.data_buffering* or *s<n>.data_buffering*) is reached. As for linear format (lin), once the limit of the kernel buffer size is reached (4k), a flow control stop (RTS off or XOFF-depending on how all.flow or s<n>.flow is set) is issued automatically to the remote device so that it will stop sending data to the serial port. Then, when a session is established to the serial port, the data in the buffer is shown to the user if not empty (dont_show_DBmenu parameter assumed to be 2), cleared, and a flow control start (RTS on or XON) is issued to resume data transmission. Once exiting the session, linear data buffering resumes. If all.flow or s<n>.flow is set to none, linear buffering is not possible as there is no way to stop reception through the serial line. Default is cir.	cir
all.DB_timestamp	Records the time stamp in the data buffering file (1) or not (0). If it is configured as 1, the software will accumulate input characters until it receives a CR and LF from the serial port or the accumulated data reaches 256 characters. Either way, the accumulated data will be recorded in the data buffering file along with the current time. The parameter all.data_buffering has to be with a non-zero value for this parameter to be meaningful.	0
all.syslog_buffering	When non zero, the contents of the data buffer are sent to the syslog-ng every time a quantity of data equal to this parameter is collected. The syslog level for data buffering is hard coded to level 5 (notice) and facility local[0+conf.DB_facility]. The file */etc/syslog-ng/syslog-ng.conf* should be set accordingly for the syslog-ng to take some action. (See Section 1.2, "Data Buffering," on page 10 to use it with Syslog Buffering Feature.)	0

Table 7.2: CAS specific parameters for the pslave.conf file

Parameter	Description	Example
all.syslog_sess	Syslog_buffering must be activated for the following to work. When 0, syslog messages are always generated whether or not there is a session to the port sending data to the unit. When 1, syslog messages are NOT generated when there IS a session to the port sending data to the unit, but resumes generation of syslog messages when there IS NOT a session to the port.	0
all.dont_show_DBmenu	When zero, a menu with data buffering options is shown when a non empty data buffering file is found. • When 1, the data buffering menu is not shown. When 2, the data buffering menu is not shown but the data buffering file is shown if not empty. • When 3, the data buffering menu is shown, but without the erase and show and erase options.	1
all.alarm	When non zero, all data received from the port are captured and sent to syslog-ng with level INFO and local[0+conf.DB_facility]facility. The *syslog-ng.conf* file should be set accordingly, for the *syslog-ng* to take some action (please see Section 4.4, "Syslog-ng," on page 119 for the *syslog-ng* configuration file).	0
all.billing_records	Billing file size configuration. A non-zero value defines the maximum number of billing records within a billing file. Zero stops billing recording. The billing files are located at /var/run/DB and are named cycXXXXX-YYMMDD.hhmmss.txt (e.g., cycTS100-030122.153611.txt.	50
all.billing_timeout	Billing timeout configuration. A non-zero value defines how long (minutes) a billing file should be waiting for records before close. After a file is closed, this file is available for transfer and a new one is opened. Zero means "no timeout" and so the file is only closed after "billing_records" are received.	60

Table 7.2: CAS specific parameters for the pslave.conf file

Parameter	Description	Example
all.billing_eor	Defines the character sequence that terminates each billing record. Any character sequence is valid, including '\r' or '^M' (carriage return), '\n' or '^J' (new line), etc..."	Default value: "\n"
all.sniff_mode	This parameter determines what other users connected to the very same port (see parameter admin_users below) can see of the session of the first connected user (main session): in shows data written to the port, out shows data received from the port, and i/o shows both streams, whereas no means sniffing is not permitted. The second and later sessions are called sniff sessions and this feature is activated whenever the protocol parameter is set to socket_ssh or socket_server.	out
all.admin_users	This parameter determines which users can receive the sniff session menu. Then they have options to open a sniff session or cancel a previous session. When users want access per port to be controlled by administrators, this parameter is obligatory and authtype must not be none. User groups (defined with the parameter conf.group) can be used in combination with user names in the parameter list.	peter, john, user_group
all.multiple_sessions	Allows users to open more than one common and sniff session on the same port. The options are "yes," "no," "RW_session," or "sniff_session." Default is set to "no." Please see Section 4.9, "Session Sniffing," on page 144 for details.	no
all.escape_char	This parameter determines which character must be typed to make the session enter "menu mode". The possible values are <CTRL-a> to <CTRL-z>. Represent the CTRL with '^'. This parameter is only valid when the port protocol is socket_server or socket_ssh. Default value is '^z'.	^z

Table 7.2: CAS specific parameters for the pslave.conf file

Parameter	Description	Example
all.tx_interval	Valid for protocols socket_server and raw_data. Defines the delay (in milliseconds) before transmission to the Ethernet of data received through a serial port. If not configured, 100ms is assumed. If set to zero or a value above 1000, no buffering will take place.	100
all.idletimeout	Specifies how long (in minutes) a connection can remain inactive before it is cut off. If it set to zero, the connection will not time out.	0
s1.serverfarm	Alias name given to the server connected to the serial port. Server_connected.	serial1
s1.pool_ipno	This is the default IP of the Cyclades AlterPath Console Server's pool of serial ports. Any host can access a port from the pool using its pool's IP address as long as a path to the address exists in the host's routing table.	192.168.2.1
s1.pool_socket_port	In the CAS profile, this defines an alternative labeling system for the Cyclades AlterPath Console Server pool of ports. In this example, serial interface 1 is assigned to the pool identified by port value 3001. Using s<serial port #>.pool_socket_port one can assign each serial interface to a different pool of ports. One serial interface can belong to just one pool of ports. Each pool of ports can have any number of serial interfaces.	3000
s1.pool_serverfarm	Alias name given to the pool where this serial interface belong to.	pool_1
s2.tty	It defines the physical device name associated to the serial port (without the /dev/).	ttyS2
s8.tty	It defines the physical device name associated to the serial port (without the /dev/).	ttyS8

Table 7.2: CAS specific parameters for the pslave.conf file

pslave.conf TS (Terminal Server) parameters

The following parameters are unique to a TS setup except where indicated.

Parameter	Description	Example
conf.telnet	Location of the telnet utility	/usr/bin/telnet
conf.ssh	Location of the ssh utility.	/bin/ssh
conf.locallogins	This parameter is only necessary when authentication is being performed for a port. When set to one, it is possible to log in to the Cyclades AlterPath Console Server directly by placing a "!" before your login name, then using your normal password. This is useful if the Radius authentication server is down.	0
all.host	The IP address of the host to which the terminals will connect.	200.200.200.3
all.term	This parameter defines the terminal type assumed when performing rlogin or telnet to other hosts.	vt100
all.userauto	Username used when connected to a UNIX server from the user's serial terminal.	
all.protocol (for TS)	For the terminal server configuration, the possible protocols are login (which requests username and password), rlogin (receives username from the Cyclades AlterPath Console Server and requests a password), telnet, ssh, ssh2, or socket_client. See all.socket_port definition if all.protocol is configured as *socket_client*.	rlogin
all.socket_port	The socket_port is the TCP port number of the application that will accept connection requested by this serial port. That application usually is telnet (23).	

Table 7.3: TS specific parameters for the pslave.conf file

Parameter	Description	Example
all.telnet_client_mode	When the protocol is TELNET, this parameter configured as BINARY (1) causes an attempt to negotiate the TELNET BINARY option on both input and output with the Telnet server. So it puts the telnet client in binary mode. The acceptable values are "0" or "1", where "0" is text mode (default) and "1" is a binary mode.	
s16.tty (TS)	It defines the physical device name associated to the serial port (without the /dev/).	ttyS16

Table 7.3: TS specific parameters for the pslave.conf file

pslave.conf Dial-in parameters

The following parameters are unique to a Dial-in setup except where indicated.

Parameter	Description	Example
conf.pppd	Location of the ppp daemon with Radius.	/usr/local/sbin/pppd
all.netmask	It defines the network mask for the serial port.	255.255.255.255
all.ipno (CAS and Dial-in)	See description in CAS section.	
all.initchat	Modem initialization string.	TIMEOUT 10 "" \d\l\dATZ \ OK\r\n-ATZ-OK\r\n “” \ “” ATMO OK\R\N “” \ TIMEOUT 3600 RING “” \ STATUS Incoming %p:I.HANDSHAKE “” ATA\ TIMEOUT 60 CONNECT@ “” \ STATUS Connected %p:I.HANDSHAKE

Table 7.4: Dial-in specific parameters for the pslave.conf file

Parameter	Description	Example
all.autoppp	*all.autoppp* PPP options to auto-detect a ppp session. The cb-script parameter defines the file used for callback and enables negotiation with the callback server. Callback is available in combination with Radius Server authentication. When a registered user calls the Cyclades AlterPath Console Server , it will disconnect the user, then call the user back. The following three parameters must be configured in the Radius Server: attribute Service_type(6): Callback Framed; attribute Framed_Protocol(7): PPP; attribute Callback_Number(19): the dial number (example: 50903300).	%i:%j novj \ proxyarp modem asyncmap\ 000A0000 \ noipx noccp login auth\ require-pap refuse-chap\ mtu %t mru %t \ cb-script\ /etc/portslave/cb_script \ plugin /usr/lib/libpsr.so
all.pppopt	*all.pppopt* PPP options when user has already been authenticated.	%i:%j novj \ proxyarp modem asyncmap\ 000A0000 noipx noccp\ mtu %t mru %t netmask%m\ idle %I maxconnect %T \ plugin /usr/lib/libpsr.so
all.protocol	For the Dial-in configuration, the available protocols are *ppp*, *ppp_only*, *slip*, and *cslip*.	ppp
s32.tty	See the *s1.tty* entry in the CAS section.	ttyS32

Table 7.4: Dial-in specific parameters for the pslave.conf file

7.2 Examples for configuration testing

The following three examples are just given to test a configuration. The steps should be followed after configuring the Cyclades AlterPath Console Server .

Console Access Server

With the Cyclades AlterPath Console Server set up as a CAS you can access a server connected to the Cyclades AlterPath Console Server through the server's serial console port from a workstation on the LAN or WAN. There is no authentication by default, but the system can be configured for authentication to be performed by a Radius server, a TacacsPlus server, or even by a local database. Either telnet or ssh can be used.

See Appendix A - New User Background Information for more information about SSH. This Chapter contains all the necessary information to configure a fully-functional CAS environment. Consult the the tables above and configure the necessary parameters for the */etc/portslave/pslave.conf* file according to your environment.

An example of a CAS environment is shown in the following figure. This configuration example has local authentication, an Ethernet interface provided by a router, and serially-connected workstations

.

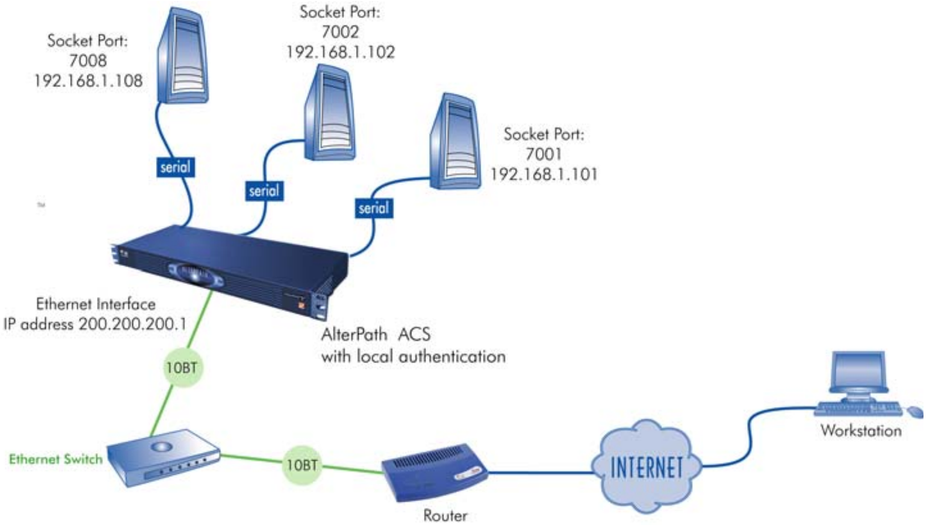


Figure 7.1 - Console Access Server diagram

The following diagram, shows additional scenarios for the Cyclades AlterPath Console Server : both remote and local authentication, data buffering, and remote access.

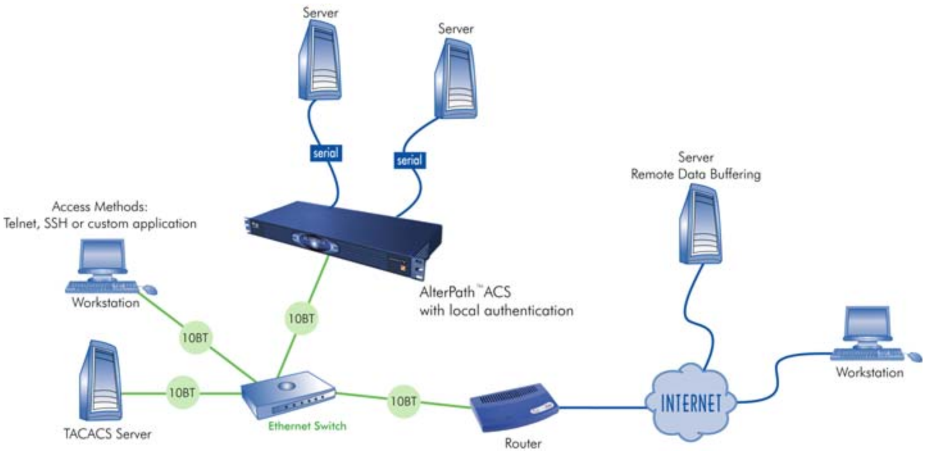


Figure 7.2 - CAS diagram with various authentication methods

As shown in the above figure, our "CAS with local authentication" scenario has either telnet or ssh (a secure shell session) being used. After configuring the serial ports as described in this chapter, the following step-by-step check list can be used to test the configuration.

Step 1 - Create a new user.

Run the *adduser <username>* to create a new user in the local database. Create a password for this user by running *passwd <username>*.

Step 2 - Confirm physical connection.

Make sure that the physical connection between the Cyclades AlterPath Console Server and the servers is correct. A cross cable (not the modem cable provided with the product) should be used. Please see Appendix B - Upgrades and Troubleshooting for pin-out diagrams.

Step 3 - Confirm that server is set to same parameters as the Cyclades AlterPath Console Server .

The Cyclades AlterPath Console Server has been set for communication at 9600 bps, 8N1. The server must also be configured to communicate on the serial console port with the same parameters.

Step 4 - Confirm routing.

Also make sure that the computer is configured to route console data to its serial console port (Console Redirection).

Telnet to the server connected to port 1.

From a server on the LAN (not from the console), try to telnet to the server connected to the first port of the Cyclades AlterPath Console Server using the following command:

```
# telnet 200.200.200.1 7001
```

For both telnet and SSH sessions, the servers can be reached by either:

1. Ethernet IP of the Cyclades AlterPath Console Server and assigned socket port.

or

2. Individual IP assigned to each port.

If everything is configured correctly, a telnet session should open on the server connected to port 1. If not, check the configuration, follow the steps above again, and check the troubleshooting appendix.

Step 5 - Activate the changes.

Now continue on “Activate the changes.” on page 70 through “Save the changes.” on page 71 listed in Chapter 3, “Network”.

NOTE: *It is possible to access the serial ports from Microsoft stations using some off-the-shelf packages. Although Cyclades is not liable for those packages, successful tests were done using at least one of them. From the application’s viewpoint running on a Microsoft station, the remote serial port works like a regular COM port. All the I/O with the serial device attached to the Cyclades AlterPath Console Server is done through socket connections opened by these packages and a COM port is emulated to the application.*

Terminal Server

The Cyclades AlterPath Console Server provides features for out-of-band management via the configuration of terminal ports. All ports can be configured as terminal ports. This allows a terminal user to access a server on the LAN. The terminal can be either a dumb terminal or a terminal emulation program on a PC.

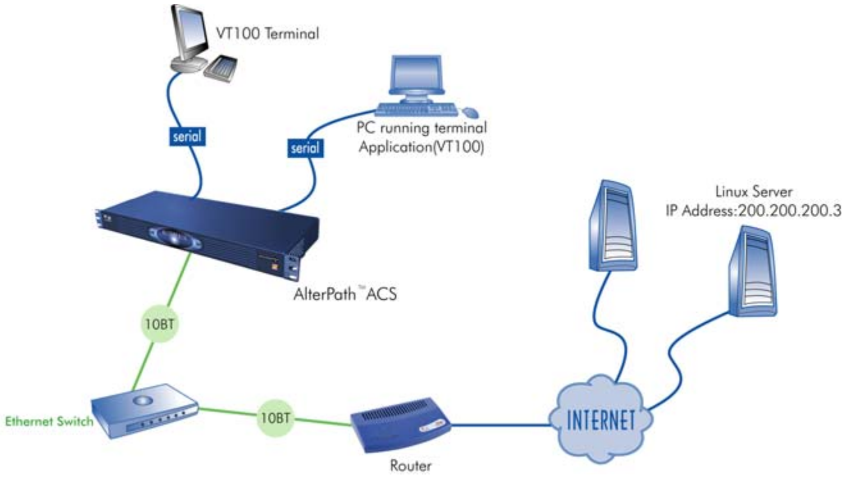


Figure 7.3 - Terminal Server diagram

No authentication is used in the example shown above and rlogin is chosen as the protocol. After configuring the serial ports as described in this chapter, the following step-by-step check list can be used to test the configuration.

Step 1 - Create a new user.

Since authentication was set to none, the Cyclades AlterPath Console Server will not authenticate the user. However, the Linux Server receiving the connection will. Create a new user on the server called test and provide him with the password test.

Step 2 - Confirm that the server is reachable.

From the console, ping 200.200.200.3 to make sure the server is reachable.

Step 3 - Check physical connections.

Make sure that the physical connection between the Cyclades AlterPath Console Server and the terminals is correct. A cross cable (not the modem cable provided with the product) should be used. Please see the Appendix B - Upgrades and Troubleshooting for pin-out diagrams.

Step 4 - Confirm that terminals are set to same parameters as the Cyclades AlterPath Console Server .

The Cyclades AlterPath Console Server has been set for communication at 9600 bps, 8N1. The terminals must also be configured with the same parameters.

Step 5 - Log onto server with new username and password.

From a terminal connected to the Cyclades AlterPath Console Server , try to login to the server using the username and password configured in step one.

Step 6 - Activate changes.

Now continue on "Activate the changes." on page 70 through "Save the changes." on page 71 listed in Chapter 3, "Network".

Dial-in Access

The Cyclades AlterPath Console Server can be configured to accommodate out-of-band management. Ports can be configured on the Cyclades AlterPath Console Server to allow a modem user to access the LAN. Radius authentication is used in this example and ppp is chosen as the protocol on the serial (dial-up) lines. Cyclades recommends that a maximum of two ports be configured for this option.

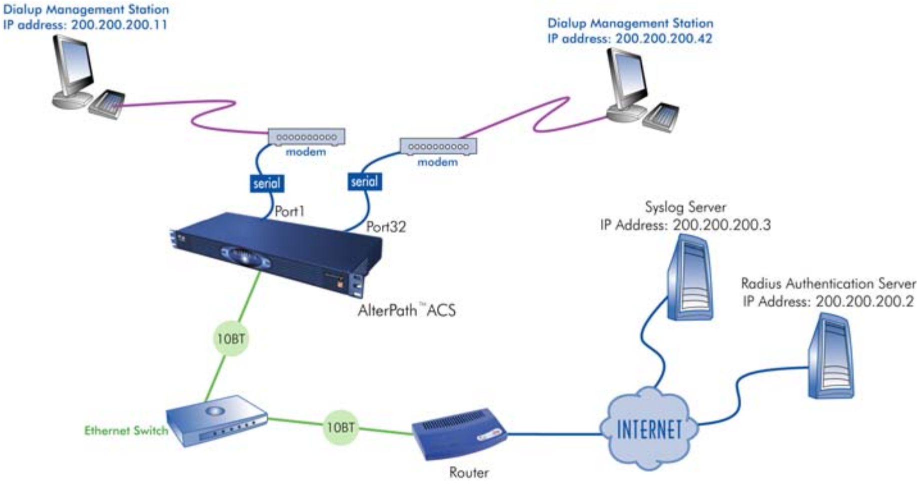

Dialup Management Station
IP address: 200.200.200.11
Dialup Management Station
IP address: 200.200.200.42
modem
modem
serial
serial
Port1
Port32
Syslog Server
IP Address: 200.200.200.3
Radius Authentication Server
IP Address: 200.200.200.2
AlterPath™ACS
10BT
Ethernet Switch
10BT
Router
INTERNET

Figure 7.4 - Ports configured for dial-in access

After configuring the serial ports as described in this Chapter, the following step-by-step check list can be used to test the configuration.

Step 1 - Create a new user.

Since Radius authentication was chosen, create a new user on the Radius authentication server called test and provide them with the password test.

Step 2 - Confirm that the Radius server is reachable.

From the console, ping 200.200.200.2 to make sure the Radius authentication server is reachable.

Step 3 - Confirm physical connections.

Make sure that the physical connection between the Cyclades AlterPath Console Server and the modems is correct. The modem cable provided with the product should be used. Please see Appendix C - Cabling and Hardware Information for pinout diagrams.

Step 4 - Confirm modem settings.

The Cyclades AlterPath Console Server has been set for communication at 57600 bps, 8N1. The modems should be programmed to operate at the same speed on the DTE interface.

Step 5 - Confirm routing.

Also make sure that the computer is configured to route console data to the serial console port.

Step 6 - Perform a test dial-in.

Try to dial in to the Cyclades AlterPath Console Server from a remote computer using the username and password configured in step one. The computer dialing in must be configured to receive its IP address from the remote access server (the Cyclades AlterPath Console Server in this case) and to use PAP authentication.

Step 7 - Activate changes.

Now continue on "Activate the changes." on page 70 through "Save the changes." on page 71 listed in Chapter 3, "Network".

This page has been left intentionally blank.

Chapter 8
Additional Features and Applications

This chapter covers special features or applications that does not fit into any of the previous chapters. The following features will be shown in this chapter:

- Windows 2003 Server Management
- IPMI
- Line Printer Daemon
- CAS Port Pool
- Billing

8.1 Windows 2003 Server Management

Emergency Management Services (EMS) is a new feature in the Windows 2003 Server that allows out-of-band remote management and system recovery tasks. All Emergency Management Services output is accessible using a terminal emulator connected to the server serial port. Besides the normal character mode output sent to the serial console, Windows also sends xml tags. Those tags can be captured and processed by the Cyclades AlterPath Console Server so that the administrator can automate the actions to be taken.

You can manage the server through the Special Administration Console (SAC), which is the console when connected directly to the Windows Server through telnet or SSH session.

How it works

To manage a Windows 2003 server it is necessary to enable the EMS (Emergency Management Services) service using the following syntax:

```
bootcfg /ems [EDIT|OFF|ON] [/s [computer] [/u [[domain\]user] /p password
[/baud baud_rate] [/port communications_port] /id line_number
```

Where:

Parameters:

- *EDIT* - Allows changes to port and baud rate settings by changing the redirect=COMx setting in the [bootloader] section. The value of COMx is set to the value of the /port.
- *OFF* - Disables output to a remote computer. Removes the /redirect switch from the specified line_number and the redirect=comX setting from the [boot loader] section.

- *ON* - Enables remote output for the specified line_number. Adds a /redirect switch to the specified line_number and a redirect=comX setting to the [boot loader] section. The value of comX is set by the /port.

Switches:

- */ems* - Enables the user to add or change the settings for redirection of the EMS console to a remote computer. By enabling EMS, you add a "redirect=Port#" line to the [boot loader] section of the BOOT.INI file and a /redirect switch to the specified operating system entry line. The EMS feature is enabled only on servers.
- */baud baud_rate* - Specifies the baud rate to be used for redirection. Do not use if remotely administered output is being disabled. Valid values are: 9600, 19200, 38400, 57600, 115200
- */id line_number* - Specifies the operating system entry line number in the [operating systems] section of the Boot.ini file to which the operating system load options are added. The first line after the [operating systems] section header is 1.
- */p password* - Specifies the password of the user account that is specified in /u.
- */port communications_port* - Specifies the COM port to be used for redirection. Do not use if remotely administered output is being disabled.
  BIOSSET get BIOS settings to determine port
  COM1
  COM2
  COM3
  COM4
- */s computer* - Specifies the name or IP address of a remote computer (do not use backslashes). The default is the local computer.
- */u [[domain\]user]* - Runs the command with the account permissions of the user specified by User or Domain\User. The default is the permissions of the current logged on user on the computer issuing the command.

With the EMS service enabled in the Windows machine, just configure the Cyclades AlterPath Console Server as CAS profile to manage the Windows 2003 server.

- Windows sends xml tags in the following situations:
- During Windows installation, it sends <channel-switch> with the setup logs.
- During boot, it sends the <machine-info> information.
- When switching channels, it sends the <channel-switch> information.
- During system crash, it sends the <BP> to indicate BreakPoint.

The <machine-info> tag is emitted once by Windows Server during its system boot sequence. This tag is also emitted as part of the <BP> tag. The following elements are included in <machine-info> tag:

Element	Description
<guid>	It is the GUID that uniquely identifies the server platform. Normally, this is an SMBIOS provided identification. If no such value is available, all 0's GUID string is used (see sample encoding below).
<name>	Is the system name.
<os-build-number>	Is a numeric string that identifies a successive Windows Build.
<os-product>	Is the name of the Windows Server 2003 product currently running on this server. It is one of the following: • Windows Server 2003 Datacenter Edition • Windows Server 2003 Embedded • Windows Server 2003 Enterprise Edition • Windows Server 2003
<os-service-pack>	Is an alphanumeric string that identifies the most up-to-date service pack installed. If none installed, the string is None.
<os-version>	Is the numeric identification of the Windows version currently running.
<processor-architecture>	Is either x86 or IA64, designating the two processor architectures currently supported by Windows Server 2003.

Table 8.1: machine info tag

A sample encoding of this tag follows:

```
<?xml>
<machine-info>
<name>NTHEAD-800I-1</name>
<guid>00000000-0000-0000-0000-000000000000</guid>
<processor-architecture>x86</processor-architecture>
<os-version>5.2</os-version>
<os-build-number>3735</os-build-number>
<os-product>Windows Server 2003 Enterprise Edition</os-product>
<os-service-pack>None</os-service-pack>
</machine-info>
```

File Description 8.1: Machine info sample tag

The console environment provided by the serial port is called Special Administration Console (SAC). In the SAC command line, each time we enter the "cmd" command we create a channel. A channel is the "Command Prompt" environment, where you can enter the Command Prompt commands (dir, cd, edit, del, copy, etc). We can switch back and forth between channel(s) and SAC by pressing Esc Tab keys. We can create up to 9 channels, i.e., up to 9 Command Prompt sessions. Whenever we switch channels, the <channel-switch> tag is sent. The following elements are included in the <channel-switch> tag:

Element	Description
<application-type>	Is a hexadecimal GUID signifying the application or tool that is running on the Windows Server platform and communicating via this active channel. It is to be used to discern the different interaction modes. During the Windows GUI-mode Setup phase, the following GUIDs identify the specific types of data being emitted: 1) Debug Log (5ED3BAC7-A2F9-4E45-9875-B259EA3F291F) 2) Error Log (773D2759-19B8-4D6E-8045-26BF38402252) 3) Action Log (D37C67BA-89E7-44BA-AE5A-112C6806B0DD) During nominal Windows Server operations, the following GUIDs can be expected: 1) SAC (63D02270-8AA4-11D5-BCCF-806D6172696F) 2) CMD (63D02271-8AA4-11D5-BCCF-00B0D014A2D0) The above are constant GUIDs and should not be confused with those provided via the <guid> tag below.
<description>	Is the user-friendly name of the active channel. For the GUI-Mode Setup tool they are: Debug Log (Setup tracing log) Error Log (Setup errors log) Action Log (Setup actions log) For the Windows Server, they are: SAC (Special Administration Console) CMD (Command Prompt)

Table 8.2: Elements in the <channel-switch> tag

Element	Description
<guid>	Is a hexadecimal GUID that identifies a specific instance of a channel. During a life-span of a Windows Server (between any two system boots), there is a total of 10 channels being allocated. Of those, one can be expected a GUID for each of the following channel types: 1) GUI-Mode Setup Debug Log 2) GUI-Mode Setup Error Log 3) GUI-Mode Setup Action Log 4) SAC The remaining GUIDs are of the CMD channel type. For example, during Windows setup, there are 3 GUIDs assigned to Setup, 1 to SAC and the remaining 6 to CMD. However, during normal Windows operations, there is 1 GUID assigned to SAC and the remaining 9 to CMD. These GUIDs are created a new for each instance of channels, and should not be confused with the constant GUIDs provided via the <application-type> tag above.
<name>	Is the system name of the active channel. For the GUI-mode Setup tool, they are the file names where the data is written: 1) Debug Log (setuplog.txt) 2) Error Log (setuperr.log) 3) Action Log (setupact.log) For Windows Server, they are: 1) SAC (SAC) 2) CMD (Cmdnnnn), where nnnn indicates the corresponding channel number
<type>	Is the type of data being emitted on the active channel. Currently, there are two types of data supported: 1) Raw for the 3 GUI-Mode Setup channels 2) VT-UTF8 for the SAC and CMD channels

Table 8.2: Elements in the <channel-switch> tag

A sample encoding of the SAC channel tag follows:

```
<channel-switch>
<name>SAC</name>
<description>Special Administration Console</description>
<type>VT-UTF8</type>
<guid>1aee4cc0-cff3-11d6-9a3d-806e6f6e6963</guid>
<application-type>63d02270-8aa4-11d5-bccf-806d6172696f</application-type>
</channel-switch>
```

File Description 8.2: SAC channel tag example

A sample encoding of the CMD channel tag follows:

```
<channel-switch>
<name>Cmd0001</name>
<description>Command Prompt</description>
<type>VT-UTF8</type>
<guid>970438d1-12bb-11d7-8a92-505054503030</guid>
<application-type>63d02271-8aa4-11d5-bccf-00b0d014a2d0</application-type>
</channel-switch>
```

File Description 8.3: CMD channel tag example

A sample encoding of the GUI-Mode Setup Debug Log channel tag follows:

```
<channel-switch>
<name>setuplog.txt</name>
<description>Setup tracing log</description>
<type>Raw</type>
<guid>6f28e904-1298-11d7-b54e-806e6f6e6963</guid>
<application-type>5ed3bac7-a2f9-4e45-9875-b259ea3f291f</application-type>
</channel-switch>
```

File Description 8.4: GUI-Mode setup debug log channel tag example

The <BP> tag is emitted when the Windows Server system halts such that only elements of the kernel are the most recently operating logic.

Element	Description
<INSTANCE CLASSNAME=>	Is the type of break point. Currently, there is only one type emitted, i.e. “Blue Screen” which indicates the system was halted prematurely. It is represented by the CLASSNAME=”BLUESCREEN” value.
<machine-info>	Is described above.
<PROPERTY NAME=>	Provides additional details, such as error code of the abnormal condition that caused the break point.

Table 8.3: <BP> tags description

A sample encoding of the Break Point tag follows:

```
<?xml>
<BP>
<INSTANCE CLASSNAME="BLUESCREEN">
<PROPERTY NAME="STOPCODE" TYPE="string"><VALUE>"0xE2"</VALUE>
</PROPERTY>
<machine-info>
<name>NTHEAD-800I-1</name>
<guid>00000000-0000-0000-0000-000000000000</guid>
<processor-architecture>x86</processor-architecture>
<os-version>5.2</os-version>
<os-build-number>3735</os-build-number>
<os-product>Windows Server 2003 Enterprise Edition</os-product>
<os-service-pack>None</os-service-pack>
</machine-info>
</INSTANCE>
</BP>
```

File Description 8.5: Break Point tag example

How to Configure

Some parameters need to be configured in the */etc/portslave/pslave.conf* to configure this feature. To enable it, follow the instructions below.

VI mode - Parameters Involved and Passed Values

There is a new parameter in */etc/portslave/pslave.conf* to monitor for xml data. For instance, for *ttyS1* we could configure:

```
s1.xml_monitor       1
```

When the *xml_monitor* is set, *cy_buffering* will search for xml packets coming from the serial port. When a complete xml packet is received, *cy_buffering* will send it to *syslog-ng*. In *syslog-ng.conf*, the following filters are available to filter the xml messages:

```
filter f_windows_bluescreen { facility(local<conf.DB_facility>) and
       level(info)nd match("XML_MONITOR") and match("BLUESCREEN"); } ;
```

and

```
filter f_windows_boot { facility(local<conf.DB_facility>) and
       level(info) and match("XML_MONITOR") and
       not match("BLUESCREEN") and match("machine-info"); } ;
```

Once the desired message is filtered, we have to define which actions we would like to take. *Syslog-ng* will create macros that can give easy access for the administrators to access the xml information. If the administrator uses these macros, syslog-ng replaces the macros by the data received in the xml packet. For instance, the following table shows the macros that are available when filter *f_windows_bluescreen* is successful and the examples of values that can replace the macros:

Macro	Description	Value to replace macro
$<INSTANCE CLASSNAME=>	Reason for the break point. Currently there is only one type, BLUESCREEN.	BLUESCREEN
$<PROPERTY NAME=>	Additional details about break point.	STOPCODE
$<VALUE>	Additional details about break point.	0xE2
$<name>	Machine name	MY_WIN_SERVER
$<guid>	GUID that uniquely identifies this server. If no such value is available, all 0's GUID string is used.	4c4c4544-8e00-4410-8045-80c04f4c4c20

Table 8.4: f_windows_boot macros

Macro	Description	Value to replace macro
$<processor-architecture>	Processor architecture. It can be either x86 or IA64.	x86
$<os-version>	Windows version.	5.2
$<os-product>	Which Windows Server product. It can be: Windows Server 2003 Datacenter Edition, Windows Server 2003 Embedded, Windows Server 2003 Enterprise Edition or Windows Server 2003.	Windows Server 2003
$<os-service-pack>	Alphanumeric string that identifies the most up-to-date service pack installed. If none installed, the string is None.	None
$<tty>	Cyclades AlterPath Console Server serial port tty or serverfarm name.	S1.ttyS1

Table 8.4: f_windows_boot macros

For the *f_windows_boot*, the following macros are available:

Macro	Description	Value to replace macro
$<name>	Machine name	MY_WIN_SERVER
$<guid>	GUID that uniquely identifies this server. If no such value is available, all 0's GUID string is used.	4c4c4544-8e00-4410-8045-80c04f4c4c20
$<processor-architecture>	Processor architecture. It can be either x86 or IA64.	x86
$<os-version>	Windows version.	5.2
$<os-build-number>	Numeric string that identifies a successive Windows Build.	3763

Table 8.5: f_windows_boot available macros

Macro	Description	Value to replace macro
$<os-product>	Which Windows Server product. It can be: Windows Server 2003 Datacenter Edition, Windows Server 2003 Embedded, Windows Server 2003 Enterprise Edition or Windows Server 2003.	Windows Server 2003
$<os-service-pack>	Alphanumeric string that identifies the most up-to-date service pack installed. If none installed, the string is None.	None
$<tty>	Cyclades AlterPath Console Server serial port tty or serverfarm name.	S2.server_connected_to_serial2

Table 8.5: f_windows_boot available macros

As an example on how we can use above macros, let's say we want the Cyclades AlterPath Console Server to send an e-mail to the administrator whenever a crash happens. The e-mail should have the information about the reason of the crash, machine name and windows version information. So we just have to create the following entry in *syslog-ng.conf*:

```
destination win2003mail { pipe("/dev/cyc_alarm"
     template("sendmail -t administrator@cyclades.com -f acs -s \"\
     Server $<name> crashed\" -m \'\
     Break Point: $<INSTANCE CLASSNAME=>  $<PROPERTY NAME=> $<VALUE>\
     Server: $<name>\
     OS: $<os-product>\
     Build: $<os-build-number>  Version: $<os-version>\
     Service Pack: $<os-service-pack>\
     Processor: $<processor-architecture>\
     Server GUID: $<guid>\
     ACS port: $<tty>\
     \' -h mail.cyclades.com "));};
```

File Description 8.6: Send e-mail when crashing example

And the following entry will activate the *win2003mail* action when the *f_windows_bluescreen* filter is successful:

```
source src { unix-stream("/dev/log"); };

log { source(src); filter(f_windows_bluescreen); destination(win2003mail); };
```

Server Commands

The following are the different commands and their descriptions that can be sent to the server.

Command Set	Description
ch	Channel management commands.
ch -ci <#>	Close a channel by its number.
cmd	Create a Command Prompt channel.
ch -si <#>	Switch to another channel (from Channel 0).
d	Dump the current kernel log.
f	Toggles the information output by the t-list command, which shows processes only, or shows processes and threads.
i	List all IP network numbers and their IP addresses.
i <#> <ip> <subnet> <gateway>	Set network interface number, IP address, subnet and gateway.
id	Display the computer identification information.
k <pid>	Kill the given process.
l <pid>	Lower the priority of a process to the lowest possible.
lock	Lock access to Command Prompt channels. You must provide valid logon credentials to unlock a channel.
m <pid> <MB-allow>	Limit the memory usage of a process to <MB-allow>.
p	Causes t-list command output to pause after displaying one full screen of information.
r <pid>	Raise the priority of a process by one.
s	Display the current time and date (24 hour clock used).
mm/dd/yyyy hh:mm	Set the current time and date (24 hour clock used).
t	Tlist.
crashdump	Crash the system. Crash dump must be enabled.
restart	Restart the system immediately.

Table 8.6: Server Commands

Command Set	Description
shutdown	Shut down the system immediately.

Table 8.6: Server Commands

8.2 IPMI

Intelligent Platform Management Interface (IPMI) is a service-level protocol and implementation that provides intelligent management to servers (and other system types in the future). IPMI allows server control and monitoring by means of a small "always-on" computer located on the server's motherboard called the Baseboard Management Controller (BMC) that can respond to IPMI commands out-of-band.

The Cyclades AlterPath Console Server has an implementation of IPMI over LAN which allows the unit to control power (i.e. power cycle) on these servers and also to obtain sensor readings such as CPU temperature(s), fan speed(s) etc.

The IPMI support in the Cyclades AlterPath Console Server extends it's functionality so that the unit can be used for serial console access to the servers and also provide power control through the IPMI protocol.

How it works

The core of the IPMI functionality on the Cyclades AlterPath Console Server will be implemented by means of a command-line utility named *ipmiutil*. This utility will enable the user to connect to a server via IPMI to control the power state of the server and to retrieve sensor readings from the server.

CLI Method

The basic format of the ipmiutil command is as follows.

```
ipmiutil <IP address> <command> <state> <auth method> <privilege>
<username> <password>
```

Where:

- *IP address* - The IP address of the IPMI server to monitor/control
- *command* - The command to issue. Valid values are power and sensors to control the power or provide sensor readings respectively
- *state* - This argument needs to specify on or off or status when the power command is given to power on, power off, or retreive status of the IPMI server (respectively). Nothing should be specified when the sensors command is given.
- *auth method* - The authmethod can be specified as none, straight, md5, or md2 depending on BMC configuration.
- *privilege* - The privilege can be specified as callback, user, operator, or admin depending on the user configuration on the BMC.
- *username* - This argument specifies the username to log in. This argument should not be specified when the auth method is set to none.

- *password* - This argument specifies the password associated with the username to log in. This argument should not be specified when the auth method is set to none.

8.3 Line Printer Daemon

This feature implements the Unix Line Printer Daemon (LPD) in the Cyclades AlterPath Console Server and can be used with local serial printers. It enables the Cyclades AlterPath Console Server to receive network print requests and service them using locally attached Serial printers.

To configure the lpd you need to follow these steps :

Step 1 - Setup the serial port where the serial printer is connected.

Edit the /etc/portslave/pslave.conf file (PortSlave configuration) and set the protocol of the serial port as "lpd".

Example :

```
s2.protocol    lpd
```

Step 2 - Create the printer definition.

Edit the */etc/printcap* file and configure the printer. The spool directory is created automatically by *cy_ras* process. Example :

```
#comment
# primary printer name and alias
#    lp |lp2| serial printer on port ttyS2
#suppress header and/or banner page
#    :sh:
#spool directory - the name is fixed as lp_ttySnn when nn is the
#serial port number
#    :sd=/var/spool/lpd/lp_ttyS2:
#printer device
#    :lp=/dev/ttyS2:
#log filename
#    :lf=/var/log/lpd.log:
#set serial port speed as 115.200 bps
#    :br115200:
lp|lp2| serial printer on port ttyS2:\
:sh: \
:sd=/var/spool/lpd/lp_ttyS2: \
:lp=/dev/ttyS2: \
:lf=/var/log/lpd.log:
```

File Description 8.7: /etc/printcap file

Step 3 - Enable the printer daemon.

Edit the file */etc/lpd.sh* and change the option ENABLE to YES

Step 4 - Allow clients to use the service.

Edit the file */etc/hosts.lpd* and include the hosts name that you allow to user the Cyclades AlterPath Console Server printers.

NOTE: *(The lpd needs to translate the IP address of the request message to the host name, check your resolv.conf file).*

Step 5 - Restart the processes, use the command "signal_ras hup" and "daemon.sh".

Step 6 - Save the configuration in flash, use the command "saveconf".

In your Linux client machine type the following command to check the Cyclades AlterPath Console Server configuration is OK:

```
# lpr -P lp@<ACS IP address> <file that you want printer> <enter>
```

8.4 CAS Port Pool

This feature is available for the ACS 2.1.4 onward. CAS Port Pooling allows you to access a free serial port from a pool in addition to the original feature where you could access a specific serial port. When you access a serial port through the pool the features sniff session and multiple sessions are not available. This feature is available for serial ports configured as CAS profile only.

You can define more than one pool of serial ports. Each serial port can only belong to ONE pool.The pool is uniquely identified by a four parameter scheme:

- protocol,
- pool_ipno,
- pool_serverfarm, and
- pool_socket_port

The three new parameters: *pool_ipno*, *pool_serverfarm*, and *pool_socket_port* have the same meaning as *ipno*, *serverfarm*, and *socket_port* respectively. Ports belonging to the same pool MUST be configured with the same value in these fields.

It is strongly recommended that you configure the same values in all parameters related to authentication for all serial ports belonging to a pool. Some of the authentication parameters are *users*, *admin_users*, and *authtype*.

You can access the serial ports from a pool with the same commands you use today to access a specific serial port. You just need to use pool_ipno, pool_serverfarm, or pool_socket_port instead ipno, serverfarm, or socket_port respectively in the SSH/telnet command.

When a connection request arrives using one of pool_ipno, pool_serverfarm, or pool_socket_port the Cyclades AlterPath Console Server will look for the first free serial port from the pool and that port will be assigned to connection. If there is no serial port free in the pool the connection is just dropped.

How to Configure it

The configuration for this feature is made directly in the */etc/portslave/pslave.conf* file. Don't forget to activate and save the configuration by issuing the commands *signal_ras hup* and *saveconf* respectively.

VI method

Following is an example of serial port pool configuration:

```
# Serial port pool: pool-1
#
s1.tty ttyS1
s1.protocol socket_server
s1.socket_port 7001 // TCP port # for specific allocation
s1.pool_socket_port 3000 // TCP port # for the pool
s1.ipno 10.0.0.1 // IP address for specific allocation
s1.pool_ipno 10.1.0.1 // IP address for the pool
s1.serverfarm serial-1 // alias for specific allocation
s1.pool_serverfarm pool-1 // alias for the pool
s2.tty ttyS2
s2.protocol socket_server
s2.socket_port 7002 // TCP port # for specific allocation
s2.pool_socket_port 3000 // TCP port # for the pool
s2.ipno 10.0.0.2 // IP address for specific allocation
s2.pool_ipno 10.1.0.1 // IP address for the pool
s2.serverfarm serial-2 // alias for specific allocation
s2.pool_serverfarm pool-1 // alias for the pool
#
# Serial port pool: pool-2
#
s3.tty ttyS3
s3.protocol socket_ssh
s3.socket_port 7003 // TCP port # for specific allocation
s3.pool_socket_port 4000 // TCP port # for the pool
s3.ipno 10.0.0.3 // IP address for specific allocation
s3.pool_ipno 10.2.0.1 // IP address for the pool
s3.serverfarm serial-3 // alias for specific allocation
s3.pool_serverfarm pool-2 // alias for the pool
s4.tty ttyS4
s4.protocol socket_ssh
s4.socket_port 7004 // TCP port # for specific allocation
s4.pool_socket_port 4000 // TCP port # for the pool
s4.ipno 10.0.0.4 // IP address for specific allocation
s4.pool_ipno 10.2.0.1 // IP address for the pool
s4.serverfarm serial-4 // alias for specific allocation
s4.pool_serverfarm pool-2 // alias for the pool
```

File Description 8.8: Part of the /etc/portslave/pslave.conf file

In the example above, there are two pools:

- pool-1 (identified by Protocol socket_server, TCP port #3000, IP 10.1.0.1, and alias pool-1)
- pool-2 (identified by Protocol socket_ssh, TCP port #4000, IP 10.2.0.1, and alias pool-2)

The serial ports ttyS1 and ttyS2 belong to the pool-1. The serial ports ttyS3 and ttyS4 belong to the pool-2.

You can access specifically serial port ttyS1 by using TCP port 7001, IP address 10.0.0.1 or alias serial-1. If the ttyS1 is being used by somebody else the connection will be dropped if the user is not a admin_user. Alternately, you can access ttyS1 through pool (if it's free) using TCP port 3000, IP 10.1.0.1 or alias pool-1. If it is not free ttyS2 will be automatically allocated. Additionally, if ttyS2 is not free, the connection will be dropped.

8.5 Billing

All the AlterPath ACS family can also be simply used as an intermediate buffer to collect serial data (like billing tickets from a PABX), making them available for a posterior file transfer.Different ports can have simultaneous "billing sessions".

General Feature Description

The AlterPath ACS reads the serial port and saves information to Ramdisk files, limited to a maximum number of records per file or a maximum lifetime. After they are closed, these files are available for file transfer at */var/run/DB*.

How to configure it

The configuration for this feature is made in the */etc/portslave/plsave.conf* file. Below are presented the parameters that need to be configured.

VI method - Passed Values and Involved Parameters

Open the */etc/portslave/pslave.conf* file and configure the following parameters according to your application:

- all.protocol - billing

Data Buffering Section:

- all.billing_records - 50
- all.billing_timeout - 60 min
- all.billing_eor - "\n"

For detailed description about the parameters shown above, please see Chapter 7, “Profile Configuration”.

NOTE: *All presented values above are going to implement the billing feature for ALL ports of the product. If the configuration for a specific port is required, all related parameters beginning with all must be changed to S.x, where x is the number of the port to be configured.*

How it works

Once the *cy_ras* program detects the protocol as “billing,” it starts the billing application. The billing application then opens the port (as configured in pslave.conf) and starts reading it. Records terminated by "billing_eor string" are expected to be received. The AlterPath ACS doesn’t change the termination method, transferring the same sequence to the file. The name of the temporary file used to write these records is:

```
cycXXXXX-YYMMDD.hhmmss.tmp
```

where:

- XXXXX is the “hostname” or “serverfarm”
- YYMMDD is the year/month/day
- hhmmss is the hour:min:sec

This name helps the user archive and browse their directory as the file can be chronologically listed, not based on its creation or modification times, but based on when its contents were recorded. Also, whenever “hostname” is not significant, the user can use the “serverfarm” name (s1.serverfarm in pslave.conf) to match their actual plant (like PABX-trunk9). The temporary file described above is closed and renamed to cycXXXXX-YYMMDD.hhmmss.txt and a new temporary file is opened when:

1. The maximum number of records specified by “billing_records” is reached;

2. The lifetime specified by “billing_timeout” finishes.

If no record is received within a file lifetime period, no file will be actually saved.

NOTE: *A zero-value for “billing_records” stops the application and a zero-value for “billing_timeout” means no timeout is desired and so the file will only be closed after “billing_records” are received.*

Disk Space Issue

Finally, it is important to note that there is a protection against disk space problems. If you configure flow control to “hardware” for the serial port (all.flow = hard in the pslave.conf file), the application monitors the available disk space and if it is less than 100 Kb, the serial interface deactivates “RTS” signal on the RS-232. “RTS” is reactivated once the disk free space is greater than 120 Kb.

This page has been left intentionally blank.

Appendix A
New User Background Information

This appendix has the objective to introduce new users with commands, file structure, processes, programs and other features used by the Cyclades AlterPath Console Server operating system. This appendix includes the following sections:

A.1 User and Passwords

A username and password are necessary to log in to the Cyclades AlterPath Console Server. The user root is predefined, with a password tslinux. A password should be configured as soon as possible to avoid unauthorized access. Type the command:

```
# passwd
```

To create a password for the root user. To create a regular user (without root privileges), use the commands:

```
# adduser user_name

# passwd user_password
```

To log out, type “logout” at the command prompt.

A regular user who wants to run the command su - to become a superuser needs to:

Step 1 - Make sure the group wheel is already created.

An administrator with root access would run the following command:

```
# addgroup wheel
```

In file */etc/group* there should be a line with at least the following:

```
wheel::zzz:
```

Step 2 - Belong to the group wheel.

An administrator with root access would edit */etc/group* file and insert the username at the end of the wheel line. For example, for user steve, the administrator would edit the line in file*/etc/group*:

```
wheel::zzz:
```

to add "steve" at the end like this:

```
wheel::zzz:steve
```

A.2 Who is logged in and what they are doing

The command "*w*" displays information about the users currently on the machine, and their processes. It calls two commands: *w_ori* and *w_cas*. The *w_ori* is the new name of the original command "*w*" and the *w_cas* shows the CAS sessions information.

The header of *w_ori* shows, in this order: the current time, how long the system has been running, how many users are currently logged on (excluded the CAS users), and the system load averages for the past 1, 5, and 15 minutes.

The following entries are displayed for each user (excluded the CAS users): login name, the tty name, the remote host, login time, idle time, JCPU time (it is the time used by all processes attached to the tty), PCPU time (it is the time used by the current process, named in the "what" field), and the command line of their current process.

The header of *w_cas* shows how many CAS users are currently logged on. The following entries are displayed for each CAS user: login name, the tty name, the remote host and remote port, login time, the process ID and the command line of the current process.

A.3 Linux File Structure

The Linux file system is organized hierarchically, with the base (or root) directory represented by the symbol "/". All folders and files are nested within each other below this base directory. The directories located just below the base directory are:

- */home* - Contains the work directories of system users.
- */bin* - Contains applications and utilities used during system initialization.
- */dev* - Contains files for devices and ports.
- */etc* - Contains configuration files specific to the operating system.
- */lib* - Contains shared libraries.
- */proc* - Contains process information.

- */mnt* - Contains information about mounted disks.
- */opt* - Location where packages not supplied with the operating system are stored.
- */tmp* - Location where temporary files are stored.
- */usr* - Contains most of the operating system files.

A.4 Basic File Manipulation

The basic file manipulation commands allow the user to copy, delete, and move files and create and delete directories.

cp *file_name destination* a) cp text.txt /tmp b) cp /chap/robo.php ./excess.php	Copies the file indicated by file_name to the path indicated by destination. a) Copies the file text.txt in the current directory to the tmp directory. b) Copies the file robo.php in the chap directory to the current directory and renames the copy excess.php.
rm *file_name*	Removes the file indicated by file_name.
mv *file_name destination*	Moves the file indicated by file_name to the path indicated by destination.
mkdir *directory_name* a) mkdir spot b) mkdir /tmp/snuggles	Creates a directory named directory_name. a) creates the directory spot in the current directory. b) creates the directory snuggles in the directory tmp.
rmdir *directory_name*	Removes the directory indicated by directory_name.

Other commands allow the user to change directories and see the contents of a directory.

pwd	Supplies the name of the current directory. While logged in, the user is always “in” a directory. The default initial directory is the user's home directory: */home/<username>*
ls [options] *directory_name*	Lists the files and directories within directory_name. Some useful options are -l for more detailed output and -a which shows hidden system files.
cd *directory_name*	Changes the directory to the one specified.
cat *file_name*	Prints the contents of file_name to the screen.

Shortcuts:

. (one dot)	Represents the current directory.
.. (two dots)	Represents one directory above the current directory (i.e. one directory closer to the base directory).

A.5 The vi Editor

To edit a file using the vi editor, type:

```
vi file_name
```

Vi is a three-state line editor: it has a command mode, a line mode and an editing mode. If in doubt as to which mode you are in, press the <ESC> key which will bring you to the command mode.

Mode	What is done there	How to get there
Command mode	Navigation within the open file.	Press the <ESC> key.
Editing mode	Text editing.	See list of editing commands below.
Line mode	File saving, opening, etc. Exiting from vi.	From the command mode, type ":" (colon).

Table A.1: vi modes

When you enter the vi program, you are automatically in command mode. To navigate to the part of the file you wish to edit, use the following keys:

h	Moves the cursor to the left (left arrow).
j	Moves the cursor to the next line (down arrow).
k	Moves the cursor to the previous line (up arrow).
l	Moves the cursor to the right (right arrow).

Table A.2: vi navigation commands

Having arrived at the location where text should be changed, use these commands to modify the text (note commands "i" and "o" will move you into edit mode and everything typed will be taken literally until you press the <ESC> key to return to the command mode).

i	Inserts text before the cursor position (everything to the right of the cursor is shifted right).
o	Creates a new line below the current line and insert text (all lines are shifted down).
dd	Removes the entire current line.
x	Deletes the letter at the cursor position.

Table A.3: vi file modification commands

After you have finished modifying a file, enter line mode (by typing ":" from command mode) and use one of the following commands:

w	Saves the file (w is for write).
wq	Saves and closes the file (q is for quit).
q!	Closes the file without saving.
w file	Saves the file with the name <file>.
e file	Opens the file named <file>.

Table A.4: vi line mode commands

A.6 The Routing Table

The Cyclades AlterPath Console Server has a static routing table that can be seen using the commands:

```
# route
```

or

```
# netstat -rn
```

The file */etc/network/st_routes* is the Cyclades AlterPath Console Server's method for configuring static routes. Routes should be added to the file (which is a script run when the Cyclades AlterPath Console Server is initialized) or at the prompt (for temporary routes) using the following syntax:

```
route [add|del] [-net|-host] target netmask nt_msk [gw gt_way] interf
```

- *[add|del]* - One of these tags must be present. Routes can be either added or deleted.
- *[-net|-host]* - Net is for routes to a network and -host is for routes to a single host.
- *target* - Target is the IP address of the destination host or network.
- *netmask* and *nt_msk* - The tag netmask and nt_mask are necessary only when subnetting is used, otherwise, a mask appropriate to the target is assumed. nt_msk must be specified in dot notation.
- *gw* and *gt_way* - Specifies a gateway, when applicable. gt_way is the IP address or hostname of the gateway.
- *interf* - The interface to use for this route. Must be specified if a gateway is not. When a gateway is specified, the operating system determines which interface is to be used.

A.7 Secure Shell Session

SSH is a command interface and protocol often used by network administrators to connect securely to a remote computer. SSH replaces its non-secure counterpart rsh and rlogin. There are two versions of the protocol, SSH and SSH2. The Cyclades AlterPath Console Server offers both. The command to start an ssh client session from a UNIX workstation is:

```
ssh -t <user>@<hostname>
```

where

- <user> = <username>:ttySnn or
  <username>:socket_port or
  <username>:ip_addr or
  <username>:serverfarm

NOTE: *"serverfarm" is a physical port alias. It can be configured in the file pslave.conf.*

An example:

```
username:               cycladesmycompany
ACS16 IP address:       192.168.160.1
host name:              acs16
servername for port 1:  file_server
```

ttyS1 is addressed by IP 10.0.0.1 or socket port 7001. The various ways to access the server connected to the port are:

```
ssh -t cyclades:ttyS1@acs16
ssh -t cyclades:7001@acs16
ssh -t cyclades:10.0.0.1@acs16
ssh -t cyclades:file_server@acs16
ssh -t -l cyclades:10.0.0.1 acs16
ssh -t -l cyclades:7001 acs16
```

For openssh clients, version 3.1p1 or later SSH2 is the default. In that case, the -1 flag is used for SSH1.

```
# ssh -t cyclades:7001@acs16

# ssh -t -2 cyclades:7001@acs16

# ssh -t cyclades:7001@acs16
```

(openssh 3.1p1 or later - AlterPath Console Server version 2.1.0 or later -> ssh2 will be used)

```
# ssh -t -1 cyclades:7001@acs16
```

(openssh 3.1p1 or later - AlterPath Console Server version 2.1.0 or later -> ssh1 will be used)

To log in to a port that does not require authentication, the username is not necessary:

```
ssh -t -2 :ttyS1@acs16
```

Note: In this case, the file sshd_config must be changed in the following way:

```
PermitRootLogin Yes
PermitEmptyPassword Yes
```

The Session Channel Break Extension

This is a new feature for the AlterPath Console Server version 2.1.3. The ACS provides new way to send a break signal during a SSH version 2 terminal session. This method is defined by “Session Channel Break Extension : draft-ietf-secsh-break-00.txt.” In previous ACS versions there is one break lenght in milliseconds (break duration). Now the ACS have a new parameter <all/Sx>.break_interval, which is used with all.break_sequence (<all/Sxx>.break_sequence). (This improves the SSH-break Cyclades implementation).

The ssh2-client receives a command ("<ssh escape char>B") from the user and sends one "break request" to ssh-server. The ssh-server receives the "break request" and sends a break command to the serial port. The ssh client can send the break duration (break interval), so the user can configure this value by command line (" -B <break interval in miliseconds> ") or by ssh_config file ("breakinterval <break interval in miliseconds>").

How it works in SSH Server (all.protocol is socket_ssh)

The serial driver accepts the parameter break interval in the break command. If the version is 2 (ssh-2), the server accepts and treats the "break request" sent by the client. The "break request" defines the break-length in miliseconds. The server sends a break command with the break-length to the serial driver to perform the break in the serial port. If the parameter all.break_sequence is configured and the server finds the sequence in the data received from client, the server sends a break command with all.break_interval to serial driver.

How it works in SSH Client

The SSH client has a new option "-B <break_interval in miliseconds>" and accepts break_interval in ssh_config. When the user types "<ssh-escape>B" (where ssh-escape is "~") the client sends a "break request" to ssh-server. When the ACS/TS calls the ssh-client automatically, it uses the parameter all.break_interval to calls the ssh-2 client.

Configuring sshd's client authentication using SSH Protocol version 1

Step 1 - Only *RhostsAuthentication yes* **in** *sshd_config***.**

In the linux host enable in the file */etc/ssh/ssh_config* the parameters:

```
Host *
   RhostsAuthentication yes
   UsePrivilegedPort yes
```

- One of these: hostname or ipaddress in:
  /etc/hosts.equiv
  or
  /etc/ssh/shosts.equiv

  hostname or ipaddress and username in *~/.rhosts* or *~/.shosts* and *IgnoreRhosts no* in *sshd_config*

- Client start-up command: *ssh -t* <Cyclades AlterPath Console Server_ip or Serial_port_ip> (if the ssh client is running under a session belonging to a username present both in the workstation's database and the Cyclades AlterPath Console Server's database).

- Client start-up command: *ssh -t -l* <username> <Cyclades AlterPath Console Server_ip or Serial_port_ip> (if the ssh client is running under a session belonging to a username present only in the workstation's database. In this case, the <username> indicated would have to be a username present in the Cyclades AlterPath Console Server's database).

NOTE: *For security reasons, some ssh clients do not allow just this type of authentication. To access the serial port, the Cyclades AlterPath Console Server must be configured for local authentication. No root user should be used as username.*

Step 2 - Only *RhostsRSAAuthentication yes* in *sshd_config*.

- One of the RhostsAuthentication settings, described in Step 1.
- Client machine's host key (*$ETC/ssh_host_key.pub*) copied into the *T/tmp/known_hosts* file. The client hostname plus the information inside this file must be appended in one single line inside the file */etc/ssh/ ssh_known_hosts* or *~/.ssh/known_hosts* and *IgnoreUserKnownHosts no* inside *sshd_config*. The following commands can be used for example:

```
# echo 'n 'client_hostname' >> /etc/ssh/ssh_known_hosts or ~/.ssh/known_hosts
# cat /tmp/known_hosts >> /etc/ssh/ssh_known_hosts or ~/.ssh/known_hosts
```

- client start-up command: *ssh -t* <Cyclades AlterPath Console Server_ip or Serial_port_ip>

NOTE: *"client_hostname" should be the DNS name. To access the serial port, the Cyclades AlterPath Console Server must be configured for local authentication. No root user should be used as username.*

Step 3 - Only *RSAAuthentication yes* in sshd_config.

- Removal of the Cyclades AlterPath Console Server's **.equiv*, *~/.?hosts*, and **known_hosts files*.
- Client identity created by ssh-keygen and its public part (*~/.ssh/identity.pub*) copied into Cyclades AlterPath Console Server's *~/.ssh/authorized_keys*.
- Client start-up command: *ssh -t* <Cyclades AlterPath Console Server_ip or Serial_port_ip>.

Step 4 - Only PasswdAuthentication yes in sshd_config.

Removal of the Cyclades AlterPath Console Server's **.equiv*, *~/.?hosts*, **known_hosts*, and **authorized_keys* files.

Client startup command: *ssh -t -l* <username> <Cyclades AlterPath Console Server_ip or Serial_port_ip> or *ssh –t –l* <username:alias><Cyclades AlterPath Console Server_ip>.

Configuring sshd's client authentication using SSH Protocol version 2

Only *PasswdAuthentication yes* in *sshd_config* DSA Authentication is the default. (Make sure the parameter *PubkeyAuthentication* is enabled.)

- Client DSA identity created by *ssh-keygen -d* and its public part (*~/.ssh/id_dsa.pub*) copied into the Cyclades AlterPath Console Server's *~/.ssh/authorized_keys2* file.
- Password Authentication is performed if DSA key is not known to the Cyclades AlterPath Console Server. Client start-up command: *ssh -2 -t* <TS_ip or Serial_port_ip>.

NOTE: *All files "~/*" or "~/.ssh/*" must be owned by the user and readable only by others. All files created or updated must have their full path and file name inside the file config_files and the command saveconf must be executed before rebooting the Cyclades AlterPath Console Server.*

Configuring the Session Channel Break Extension in SSH Server

Step 1 - Configure the parameter *break_interval* in */etc/portslave/pslave.conf*.
This can be done by the admin using the vi editor or CLI.

Step 2 - Configure the parameter ssh_interval in ssh_config.
This can be done using the vi editor.

A.8 The Process Table

The process table shows which processes are running. Type ps -a to see a table similar to that below.

PID	UID	State	Command
1	root	S	/sbin/inetd
31	root	S	/sbin/inetd
32	root	S	/sbin/cy_ras
36	root	S	/sbin/cy_wdt_led wdt led
154	root	R	/ps -a

Table A.5: Process Table

To restart the cy_ras process use its process ID or execute the command:

```
# signal_ras hup
```

This executes the *ps* command, searches for the *cy_ras* process id, then sends the signal hup to the process, all in one step. Never kill *cy_ras* with the signals -9 or SIGKILL.

A.9 TS Menu Script

The *ts_menu* script can be used to avoid typing long telnet or SSH commands. It presents a short menu with the names of the servers connected to the serial ports of the Cyclades AlterPath Console Server. The server is selected by its corresponding number. *ts_menu* must be executed from a local session: via console, telnet, ssh, dumb terminal connected to a serial port, etc. Only ports configured for console access (protocols *socket_server* or *socket_ssh*) will be presented. To start having familiarity with this application, run *ts_menu - h*:

```
# ts_menu -h

USAGE: ts_menu options

-p : Display Ethernet Ip and Tcp port
-i : Display local Ip assigned to the serial port
-u <name> : Username to be used in ssh/telnet command
-U : Allows choosing of different usernames for different ports
-h : print this help message
```

Below is presented an example of how TS Menu can be used:

```
# ts_menu

Master and Slaves Console Server Connection Menu

1 TSJen800
2 test.Cyclades.com
3 az84.Cyclades.com
4 64.186.190.85
5 az85.Cyclades.com

Type 'q' to quit, a valid option [1-5], or anything else to refresh:
```

By selecting 1 in this example, the user will access the local serial ports on that Cyclades AlterPath Console Server. If the user selects 2 through 5, remote serial ports will be accessed. This is used when there is clustering (one Cyclades AlterPath Console Server master box and one or more Cyclades AlterPath Console Server slave boxes).

If the user selects 1, the following screen is displayed:

```
Serial Console Server Connection Menu for your Master Terminal Server

1 ttyS1      2 ttyS2     3 s3serverfarm

Type 'q' to quit, 'b' to return to previous menu, a valid option[1-3], or
anything else to refresh:
```

Options 1 to 3 in this case are serial ports configured to work as a CAS profile. Serial port 3 is presented as an alias name (*s3serverfarm*). When no name is configured in *pslave.conf*, ttyS<N> is used instead. Once the serial port is selected, the username and password for that port (in case there is a per-user access to the port and -U is passed as parameter) will be presented, and access is granted.

To access remote serial ports, the presentation will follow a similar approach to the one used for local serial ports.

The *ts_menu* script has the following line options:

-p : Displays Ethernet IP Address and TCP port instead of server names.

```
Cyclades AlterPath Console Server: Serial Console Server Connection menu

1 209.81.55.79 7001   2 209.81.55.79 7002   3 209.81.55.79 7003
4 209.81.55.79 7004   5 209.81.55.79 7005   6 209.81.55.79 7006

Type 'q' to quit, a valid option [1-6], or anything else to refresh :
```

-i : Displays Local IP assigned to the serial port instead of server names.

```
Cyclades AlterPath Console Server: Serial Console Server Connection menu

1 192.168.1.101    2 192.168.1.102    3 192.168.1.103 4 192.168.1.104
5 192.168.1.105    6 192.168.1.106

Type 'q' to quit, a valid option [1-6], or anything else to refresh :
```

-u <name> : Username to be used in the ssh/telnet command. The default username is that used to log onto the Cyclades AlterPath Console Server.

-h : Lists script options.

Appendix B
Upgrades and Troubleshooting

This appendix has the objective to cover the most common problems that users faces when using the Cyclades AlterPath Console Server . This appendix will also show the necessary steps to upgrade the firmware of the Cyclades AlterPath Console Server unit and how to correctly interpret the CPU LED status.

B.1 Upgrades

Users should upgrade the Cyclades AlterPath Console Server whenever there is a bug fix or new features that they would like to have. Below are the six files added by Cyclades to the standard Linux files in the */proc/flash* directory when an upgrade is needed. They are:

- boot_ori - original boot code
- boot_alt - alternate boot code
- syslog - event logs (not used by Linux)
- config - configuration parameters, only the boot parameters are used by the boot code
- zImage - Linux kernel image
- script - file where all Cyclades AlterPath Console Server configuration information is stored

The Upgrade Process

To upgrade the Cyclades AlterPath Console Server , follow these steps:

Step 1 - Log in to the Cyclades AlterPath Console Server as root.

Provide the root password if requested.

Step 2 - Go to the */proc/flash* directory using the following command:

```
cd /proc/flash
```

Step 3 - FTP to the host where the new firmware is located.

Log in using your username and password. Go to the directory where the firmware is located. Select binary transfer and “get” the firmware file.

```
# ftp
```

```
ftp> open server
ftp> user admin
ftp> Password: adminpw
ftp> cd /tftpboot
ftp> bin
ftp> get zImage.134 zImage
ftp> quit
```

NOTE: *The destination file name in the /proc/flash directory must be zImage. Example (hostname = server; directory = /tftpboot; username= admin; password = adminpw; firmware filename on that server = zImage.134).*

NOTE: *Due to space limitations, the new zImage file may not be downloaded with a different name, then renamed. The Cyclades AlterPath Console Server searches for a file named zImage when booting and there is no room in flash for two zImage files.*

Step 4 - Run zImage.

To make sure the downloaded file is not corrupted or that the zImage saved in flash is OK the user should run:

```
md5sum -b /proc/flash/zImage
```

Step 5 - Check text file information.

Now the user should check with the information present in the text file saved in the Cyclades site (e.g. zImage.134.md5sum). If the numbers match, the downloaded file is not corrupted.

Step 6 - Issue the command reboot.

```
# reboot
```

Step 7 - Confirm that the new Linux kernel has taken over.

After rebooting, the new Linux kernel will take over. This can be confirmed by typing the following to see the Linux kernel version:

```
# cat /proc/version
```

B.2 Troubleshooting

Flash Memory Loss

If the contents of flash memory are lost after an upgrade, please follow the instructions below to restore your system:

Step 1 - Turn the Cyclades AlterPath Console Server OFF, then back ON.

Step 2 - Using the console, wait for the self test messages.

If you haven't got any, make sure you have the right settings. If you really get no boot message, press <s> right after powering ON and skip ALTERNATE boot code. That will make the boot run its ORIGINAL boot code.

Step 3 - During the self test, press <Esc> after the Ethernet test.

Step 4 - When the Watch Dog Timer prompt appears, press <Enter>.

Step 5 - Choose the option Network Boot when asked.

Step 6 - Enter the IP address of the Ethernet interface.

Step 7 - Enter the IP address of the host where the new zImage file is located.

Step 8 - Enter the file name of the zImage file on the host.

Step 9 - Select the TFTP option instead of BOOTP.

The host must be running TFTPD and the new zImage file must be located in the proper directory. e.g. */tftpboot* for Linux.

Step 10 - Accept the default MAC address by pressing <Enter>.

The Cyclades AlterPath Console Server should begin to boot off the network and the new image will be downloaded and begin running in RAM. At this point, follow the upgrade steps above (login, cd */proc/flash*, *ftp*, and so forth) to save the new zImage file into flash again.

NOTE: *Possible causes for the loss of flash memory may include: downloaded wrong zImage file, downloaded as ASCII instead of binary; problems with flash memory.*

If the Cyclades AlterPath Console Server booted properly, the interfaces can be verified using ifconfig and ping. If ping does not work, check the routing table using the command route. Of course, all this should be tried after checking that the cables are connected correctly.

The file */etc/config_files* contains a list of files acted upon by saveconf and restoreconf. If a file is missing, it will not be loaded onto the ramdisk on boot. The following table lists files that should be included in the */etc/config_files* file and which programs use each.

File	Program
/etc/securetty	telnet, login, su
/etc/issue	getty
/etc/getty_ttyS0	login (via console)
/etc/hostname	tcp
/etc/hosts	tcp
/etc/host.conf	tcp
/etc/nsswitch.conf	dns
/etc/resolv.conf	dns
/etc/config_files	saveconf
/etc/passwd	login, passwd, adduser...
/etc/group	login, passwd, adduser...
/etc/ssh/ssh_host_key.pub	sshd
/etc/ssh/sshd_config	sshd
/etc/ssh/ssh_config	ssh client
/etc/ssh/ssh_host_key	sshd (ssh1)
/etc/ssh/ssh_host_key.pub	sshd (ssh1)
/etc/ssh/ssh_host_dsa_key	sshd (ssh2)
/etc/ssh/ssh_host_dsa_key.pub	sshd (ssh2)
/etc/snmp/snmpd.conf	snmp

Table B.1: Files to be included in /etc/config_file and the program to use

File	Program
/etc/portslave/pslave.conf	cy_ras, portslave,Cyclades AlterPath Console Server configuration information
/etc/network/ifcfg_eth0	ifconfig eth0, cy_ras, rc.sysinit
/etc/network/ifcfg*	ifconfig, cy_ras, rc.sysinit
/etc/network/ifcfg_lo ifconfig	lo, cy_ras, rc.sysinit
/var/run/radsession.id	radinit, radius authentication process
/home	adduser, passwd
/etc/network/st_routes	ifconfig, cy_ras, rc.sysinit
/etc/syslog-ng/syslog-ng.conf	syslog-ng

Table B.1: Files to be included in /etc/config_file and the program to use

IMPORTANT! *If any of the files listed in /etc/config_files is modified, the Cyclades AlterPath Console Server administrator must execute the command saveconf before rebooting the Cyclades AlterPath Console Server or the changes will be lost. If a file is created (or a filename altered), its name must be added to this file before executing saveconf and rebooting.*

IMPORTANT! *Cyclades Technical Support is always ready to help with any configuration problems. Before calling, execute the command*

```
# cat /proc/version
```

and note the Linux version and Cyclades AlterPath Console Server version written to the screen. This will speed the resolution of most problems.

Hardware Test

A hardware test called tstest is included with the Cyclades AlterPath Console Server firmware. It is a menu-driven program, run by typing tstest at the command prompt. The various options are described below. Note that the Cyclades AlterPath Console Server should not be tested while in use as the test will inactivate all ports. You should inactivate all processes that may use the serial ports: *inetd*, *sshd*, *cy_ras*, and *cy_buffering*. Following are the hardware test steps:

Step 1 - signal_ras stop.

Step 2 - Perform all hardware tests needed.

Step 3 - signal_ras start.

Port Test

Either a cross cable or a loop-back connector is necessary for this test. Their pinout diagrams are supplied in Appendix C - Cabling and Hardware Information. Connect the loop-back connector to the modem cable and then connect the modem cable to the port to be tested (or connect a cross cable between two ports to be tested). When *tstest* senses the presence of the cable or connector, the test will be run automatically and the result shown on the screen.

Each line of data corresponds to a port in test. The last four columns (DATA, CTS, DCD, and DSR) indicate errors. The values in these columns should be zero. Below is an example of the output screen.

```
 <- Packets ->                          <- Errors ->

From     To     Sent   Received     Passes    Data   CTS    DCD    DSR
2 <->    2      35     35           35        0      0      0      0
4 <->    5      35     35           35        0      0      0      0
5 <->    4      35     35           35        0      0      0      0
```

When this test is run with a cable or connector without the DSR signal (see the pinout diagram for the cable or connector being used), errors will appear in the DSR column. This does not indicate a problem with the port. In the example above, tstest perceived that a loop-back connector was attached to port 2 and that a cross cable was used to connect ports 4 and 5.

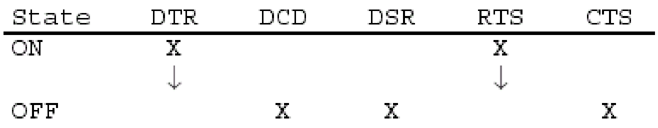
Port Conversation

This test sends and receives data on the selected port. One way to run this test is to place a loop-back connector on the port to be tested and begin. Enter the number of the port and a baud rate (9600 is a typical value). Type some letters, and if the letters appear on the screen, the port is working. If the letters do not appear on the screen (which also occurs if the loop-back connector is removed), the port is not functioning correctly.

A second method that can be used to test the port is to connect it to a modem with a straight cable. Begin the test and type “at”. The modem should respond with “OK”, which will appear on the screen. Other commands can be sent to the modem or to any other serial device. Press Ctrl-Q to exit the terminal emulation test.

Test Signals Manually

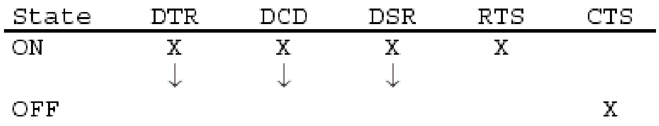
This test confirms that signals are being sent and received on the selected port. Neither the loop-back connector nor the cross cable are necessary. Enter the number of the port to be tested and begin the test.



```
State     DTR      DCD      DSR      RTS      CTS
ON        X                          X
          ↓                          ↓
OFF                X        X                 X
```

Figure B.1 - Initial Test

First, type Ctrl-D to see the X in the DTR column move position, then type Ctrl-R to see the X in the RTS column change position. If each of the Xs moves in response to its command, the signals are being sent. Another method to test the signals is to use a loop-back connector. Enter the number of the port with the loopback connector and start the test. In this case, when Ctrl-D is typed, the Xs in the first three columns will move as shown below.



```
State     DTR      DCD      DSR      RTS      CTS
ON        X        X        X        X
          ↓        ↓        ↓
OFF                                           X
```

Figure B.2 - Second screen, showing changed positions

This is because the test is receiving the DTR signal sent through the DCD and DSR pins. When Ctrl-R is typed, the Xs in the RTS and CTS columns should move together. If the Xs change position as described, the signals are being sent and received correctly.

Single User Mode

The Cyclades AlterPath Console Server has a single user mode used when:

- The name or password of the user with root privileges is lost or forgotten,
- After an upgrade or downgrade which leaves the Cyclades AlterPath Console Server unstable,
- After a configuration change which leaves the Cyclades AlterPath Console Server inoperative or unstable.

Type the word “ single” (with a blank space before the word) during boot using a console connection. This cannot be done using a telnet or other remote connection. The initial output of the boot process is shown below.

```
Entry Point = 0x00002120
loaded at: 00002120 0000D370
relocated to: 00300020 0030B270
board data at: 003052C8 0030537C
relocated to: 002FF120 002FF1D4
zimage at: 00008100 0006827E
relocated to: 00DB7000 00E1717E
initrd at: 0006827E 0024F814
relocated to: 00E18000 00FFF596
avail ram: 0030B270 00E18000
Linux/PPC load: root=/dev/ram
```

After printing “Linux/PPC load: root=/dev/ram,” the Cyclades AlterPath Console Server waits approximately 10 seconds for user input. This is where the user should type “<sp>single” (spacebar, then the word “single”). When the boot process is complete, the Linux prompt will appear on the console:

```
[root@(none) /]#
```

If the password or username was forgotten, execute the following commands:

```
# passwd
# saveconf
# reboot
```

For configuration problems, you have two options:

Step 1 - Edit the file(s) causing the problem with vi, then execute the commands:

```
# saveconf
# reboot
```

Step 2 - Reset the configuration by executing the commands:

```
# echo > /proc/flash/script
# reboot
```

If the problem is due to an upgrade/downgrade, a second downgrade/upgrade will be necessary to reverse the process. First, the network must be initialized in order to reach a ftp server. Execute the following script, replacing the parameters with values appropriate for your system. If your ftp server is on the same network as the , the *gw* and *mask* parameters are optional. Edit the file(s) causing the problem with vi, then execute the commands:

```
# saveconf

# reboot
```

Step 1: Reset the configuration by executing the commands:

```
# echo > /proc/flash/script
# reboot
```

If the problem is due to an upgrade/downgrade, a second downgrade/upgrade will be necessary to reverse the process. First, the network must be initialized in order to reach a ftp server. Execute the following script, replacing the parameters with values appropriate for your system. If your ftp server is on the same network as the , the *gw* and *mask* parameters are optional.

```
# config_eth0 ip 200.200.200.1 mask 255.255.255.0 gw 200.200.200.5
```

At this point, the DNS configuration (in the file */etc/resolv.conf*) should be checked. Then, download the kernel image using the *ftp* command.

Using a different speed for the Serial Console

The serial console is originally configured to work at 9600 bps. If you want to change that, it is necessary to change the configuration following the steps:

Step 1 - Run bootconf. The user will be presented with the screen:

```
Current configuration
MAC address assigned to Ethernet [00:60:2e:00:16:b9]
IP address assigned to Ethernet interface [192.168.160.10]
Watchdog timer ((A)ctive or (I)nactive) [A]
Firmware boot from ((F)lash or (N)etwork) [F]
Boot type ((B)ootp,(T)ftp or Bot(H)) [T]
Boot File Name [zvmppcts.bin]
Server's IP address [192.168.160.1]
Console speed [9600]
(P)erform or (S)kip Flash test [P]
(S)kip, (Q)uick or (F)ull RAM test [F]
Fast Ethernet ((A)uto Neg, (1)00 BtH, 100 Bt(F), 10 B(t)F, 10 Bt(H)) [A]
Fast Ethernet Maximum Interrupt Events [0]
```

Type <Enter> for all fields but the Console Speed. When presented the following line:

```
Do you confirm these changes in flash ( (Y)es, (N)o (Q)uit ) [N] :
```

Step 2 - Enter Y and the changes will be saved in flash.

Step 3 - Logout and login again to use the console at the new speed.

B.3 LEDs

CPU Leds

Normally the CPU status LED should blink consistently one second on, one second off. If this is not the case, an error has been detected during the boot. The blink pattern can be interpreted via the following table:

Event	CPU LED Morse code
Normal Operation	S (short, short, short . . .)
Flash Memory Error - Code	L (long, long, long . . .)
Flash Memory Error - Configuration	S, L
Ethernet Error	S, S, L
No Interface Card Detected	S, S, S, L
Network Boot Error	S, S, S, S, L
Real-Time Clock Error	S, S, S, S, S, L

Table B.2: CPU LED Code Interpretation

NOTE: *The Ethernet error mentioned in the above table will occur automatically if the Fast Ethernet link is not connected to an external hub during the boot. If the Fast Ethernet is not being used or is connected later, this error can be ignored.*

Rear Panel LEDs

The ACS' rear panel has connectors (serial, console and Ethernet) with some LEDs that have the following functionalities:

Ethernet Connector

- *Col (collision)* - Shows collision on the LAN every time the unit tries to transmit an Ethernet packet.
- *DT/LK (data transaction/link state)* - DT flashes when there's data transmitted to or received from the LAN. It's hardware-controlled. LK keeps steady if the LAN is active. The green LED is Data Transaction activity and the yellow one is LinK state.
- *100* - If 100BT is detected the LED lights on. If 10BT is detected it turns off.

Console Connector

- *CP* - CPU activity. It flashes at roughly 1 second intervals.
- *P1* - Power supply #1 ON.
- *P2* - Power supply #2 ON.

Serial Connector

- *LK* - DTR. It's software-controlled.
- *DT* - Data transmitted to or received from the serial line. It's hardware-controlled.

Appendix C
Cabling and Hardware Information

This appendix will show all hardware specifications of the Cyclades AlterPath Console Server . It will also show all cables and conectors characteriscs.

C.1 General Hardware Specifications

The power consumption and heat dissipation, environmental conditions and physical specifications of the Cyclades AlterPath Console Server are listed below.

Cyclades AlterPath ACS Products Power Consumption and Heat Dissipation				
	Input = 120Vac		Input = 230 Vac	
Model	**Power (watts)**	**Heat Exchange (BTU/hr.)**	**Power (Watts)**	**Heat Exchange (BTU/hr.)**
ACS1	12	41.0	17	58.1
ACS4	16	54.6	25	85.4
ACS8	18	61.5	28	95.6
ACS16	22	75.1	30	102.5
ACS32	24	81.0	32	109.3
ACS48	26	88.8	35	119.5

Table C.1: ACS Products Power Consumption and Heat Dissipation

Environmental Information						
	ACS1	ACS4	ACS8	ACS16	ACS32	ACS48
Operating Temperature	50F to 122F (10ºC to 50ºC)	50F to 112F (10ºC to 44ºC)	50F to 112F (10ºC to 44ºC)	50F to 112F (10ºC to 44ºC)	50F to 112F (10ºC to 44ºC)	50F to 112F (10ºC to 44ºC)
Relative Humidity	10 - 90%, non-condensing	10 - 90%, non-condensing	10 - 90%, non-condensing	10 - 90%, non-condensing	10 - 90%, non-condensing	10 - 90%, non-condensing

Table C.2: AlterPath Console Server environmental conditions

Physical Information						
	ACS1	ACS4	ACS8	ACS16	ACS32	ACS48
External Dimensions	2.76in.x 3.35in.x 1.18 in.	8,5in.x 4,75in. x1 in.	8.5 in.x 4.75 in.x 1 in.	17 in. x 8.5 in.x 1.75 in.	17 in.x 8.5 in.x 1.75 in.	17 in. x 8.5 in.x 1.75 in.
Weight	0.3 lb.	1.5 lb.	1.6 lb.	6 lb.	6.2 lb.	8 lb.

Table C.3: AlterPath Console Server physical information

Safety Information						
	ACS1	ACS4	ACS8	ACS16	ACS32	ACS48
Approvals	FCC and CE, Class A					

Table C.4: AlterPath Console Server Safety Information

The following section has all the information you need to quickly and successfully purchase or build cables to the Cyclades AlterPath Console Server . It focuses on information related to the RS-232 interface, which applies not only to the Cyclades AlterPath Console Server but also to any RS-232 cabling.

The RS-232 Standard

RS-232C, EIA RS-232, or simply RS-232 refer to a standard defined by the Electronic Industries Association in 1969 for serial communication. More than 30 years later, more applications have been found for this standard than its creators could have imagined. Almost all electronic devices nowadays have serial communication ports.

RS-232 was defined to connect Data Terminal Equipment, (DTE, usually a computer or terminal) to Data Communication Equipment (DCE, usually a modem):

```
DTE > RS-232 > DCE > communication line > DCE > RS-232 > DTE
```

RS-232 is now mostly being used to connect DTE devices directly (without modems or communication lines in between). While that was not the original intention, it is possible with some wiring tricks. The relevant signals (or wires) in a RS-232 cable, from the standpoint of the computer (DTE), are:

- *Receive Data (RxD) and Transmit Data (TxD)* - The actual data signals
- *Signal Ground (Gnd)* - Electrical reference for both ends
- *Data Terminal Ready (DTR)* - Indicates that the computer (DTE) is active
- *Data Set Ready (DSR)* - Indicates that the modem (DCE) is active.
- *Data Carrier Ready (DCD)* - Indicates that the connection over the communication line is active
- *CTS (Clear to Send, an input)* - Flow control for data flowing from DTE to DCE
- *RTS (Request to Send, an output)* - Flow control for data flowing from DCE to DTE

Not all signals are necessary for every application, so the RS-232 cable may not need all 7 wires. The RS-232 interface defines communication parameters such as parity, number of bits per character, number of stop-bits and the baud rate. Both sides must be configured with the same parameters. That is the first thing to verify if you think you have the correct cable and things still do not work. The most common configuration is 8N1 (8 bits of data per character, no parity bit included with the data, 1 stop-bit to indicate the end of a character). The baud rate in a RS-232 line translates directly into the data speed in bits per second (bps). Usual transmission speeds range between 9,600 bps and 19,200bps (used in most automation and console applications) to 115,200 bps (used by the fastest modems).

Cable Length

The original RS-232 specifications were defined to work at a maximum speed of 19,200 bps over distances up to 15 meters (or about 50 feet). That was 30 years ago. Today, RS-232 interfaces can drive signals faster and through longer cables. As a general rule, consider:

- If the speed is lower than 38.4 kbps, you are safe with any cable up to 30 meters (100 feet)
- If the speed is 38.4 kbps or higher, cables should be shorter than 10 meters (30 feet)
- If your application is outside the above limits (high speed, long distances), you will need better quality (low impedance, low-capacitance) cables.

Successful RS-232 data transmission depends on many variables that are specific to each environment. The general rules above are empirical and have a lot of safety margins built-in.

C.2 Connectors

The connector traditionally used with RS-232 is the 25-pin D-shaped connector (DB-25). Most analog modems and most older computers and serial equipment use this connector. The RS-232 interface on DB-25 connector always uses the same standard pin assignment.

The 9-pin D-shaped connector (DB-9) saves some space and is also used for RS-232. Most new PC COM ports and serial equipment (specially when compact size is important) uses this connector. RS-232 interfaces on DB-9 connectors always use the same standard pin assignment.

The telephone-type modular RJ-45 plug and jack are very compact, inexpensive and compatible with the phone and Ethernet wiring systems present in most buildings and data centers. Most networking equipment and new servers use RJ-45 connectors for serial communication. Unfortunately there is no standard RS-232 pin assignment for RJ-45 connectors. Every equipment vendor has its own pin assignment.

Most connectors have two versions. The ones with pins are said to be "male" and the ones with holes are said to be "female."

RS-232 Signal	Name/Function (Input/Output)	DB-25 pins (Standard)	DB-9 pins (Standard)	RJ-45 pins (Cyclades)
Chassis	Safety Ground	1	Shell	Shell
TxD	Transmit Data (O)	2	3	3
RxD	Receive Data (I)	3	2	6
DTR	Data Terminal Ready (O)	20	4	2
DSR	Data Set Ready (I)	6	6	8
DCD	Data Carrier Detect (I)	8	1	7
RTS	Request To Send (O)	4	7	1
CTS	Clear To Send (I)	5	8	5
GnD	Signal Ground	7	5	4

Table C.5: Cables and their pin specifications

Straight-Through vs. Crossover Cables

The RS-232 interface was originally intended to connect a DTE (computer, printer and other serial devices) to a DCE (modem) using a straight-through cable (all signals on one side connecting to the corresponding signals on the other side one-to-one). By using some "cabling tricks," we can use RS-232 to connect two DTEs as is the case in most modern applications.

A crossover (a.k.a. null-modem) cable is used to connect two DTEs directly, without modems or communication lines in between. The data signals between the two sides are transmitted and received and there are many variations on how the other control signals are wired. A "complete" crossover cable would connect TxD with RxD, DTR with DCD/DSR, and RTS with CTS on both sides. A "simplified" crossover cable would cross TxD and RxD and locally short-circuit DTR with DCD/DSR and RTS with CTS.

Which cable should be used?

First, look up the proper cable for your application in the table below. Next, purchase standard off-the-shelf cables from a computer store or cable vendor. For custom cables, refer to the cable diagrams to build your own cables or order them from Cyclades or a cable vendor.

To Connect To	Use Cable
DCE DB-25 Female (standard) • Analog Modems • ISDN Terminal Adapters	Cable 1: RJ-45 to DB-25 M straight-through (Custom). This custom cable can be ordered from Cyclades or other cable vendors. A sample is included with the product ("straight-through").
DTE RJ-45 Cyclades (custom) • All Cyclades Console Ports	Cable 2: RJ-45 to RJ-45 crossover (custom). A sample is included with the product ("straight-through") This custom cable can be ordered from Cyclades or other cable vendors using the provided wiring diagram.

Table C.6: Which cable to use

Cable Diagrams

Before using the following cable diagrams refer to the tables above to select the correct cable for your application. Sometimes, crossover cables are wired slightly differently depending on the application. A “complete” crossover cable would connect the TxD with RxD, DTR with DCD/DSR, and RTS with CTS across both sides. A “simplified” crossover cable would cross TxD and RxD and locally short-circuit DTR with DCD/DSR and RTS with CTS.

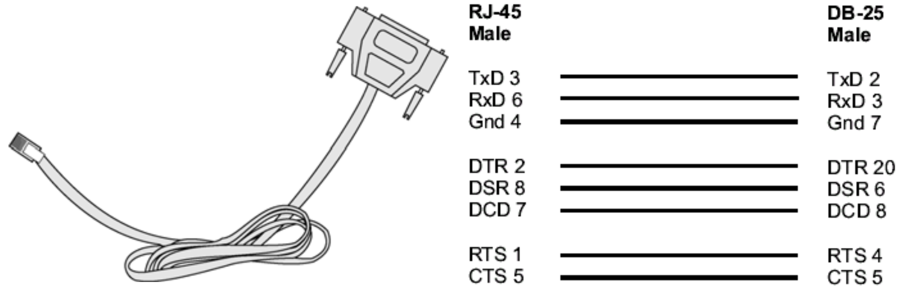
Most of the diagrams in this document show the “complete” version of the crossover cables, with support for modem control signals and hardware flow control. Applications that do not require such features have just to configure NO hardware flow control and NO DCD detection on their side. Both ends should have the same configuration for better use of the complete version of the cables.

NOTE: *These cables appear in Cable Package #1 and/or Cable Package #2. You may or may not find them in your box depending on which package you received.*

C.3 Cable Packages

Cable #1: Cyclades RJ-45 to DB-25 Male, straight-through

Application: This cable connects Cyclades products (serial ports) to modems and other DCE RS-232 devices. It is included in both Cable Package #1 and #2.



RJ-45 Male	DB-25 Male
TxD 3	TxD 2
RxD 6	RxD 3
Gnd 4	Gnd 7
DTR 2	DTR 20
DSR 8	DSR 6
DCD 7	DCD 8
RTS 1	RTS 4
CTS 5	CTS 5

Figure C.7 - Cable 1 - Cyclades RJ-45 to DB-25 Male, straight-through

Cable #2: Cyclades RJ-45 to DB-25 Female/Male, crossover

This cable connects Cyclades products (serial ports) to console ports, terminals, printers and other DTE RS-232 devices. If you are using Cable Package #1, after connecting the appropriate adapter to the RJ-45 straight-through cable, you will essentially have the cable shown in this picture. If you are using Cable Package #2, no assembly is required. You will have the cable shown below.

Cable #3: Cyclades RJ-45 to DB-9 Female, crossover

This cable connects Cyclades products (serial ports) to console ports, terminals, printers and other DTE RS-232 devices. If you are using Cable Package #1, after connecting the appropriate adapter to the RJ-45 straight-through cable, you will essentially have the cable shown in this picture. If you are using Cable Package #2, no assembly is required. You will have the cable shown below.

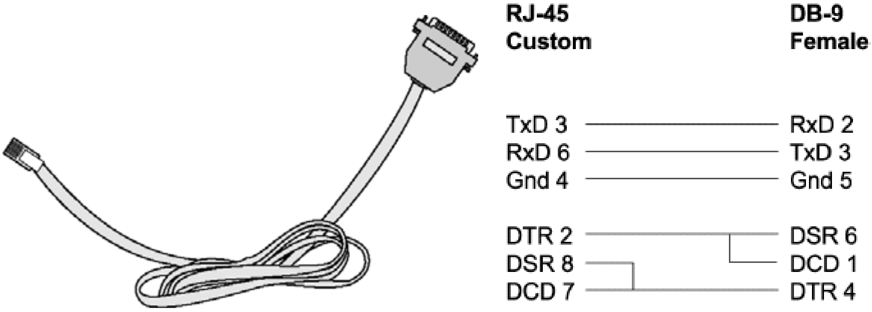


Figure C.8 - Cable 3 - Cyclades RJ-45 to DB-9 Female, crossover

Cable #4: Cyclades RJ-45 to Cyclades RJ-45, straight-through

This cable is the main cable that you will use. Along with one of the adapters provided (RJ-45 to DB-9 or RJ-45 to DB-25) you can create a crossover cable like the ones explained in Cable #2 or #3 for configuration or to connect to a server. This cable is only included in Cable Package. #1.

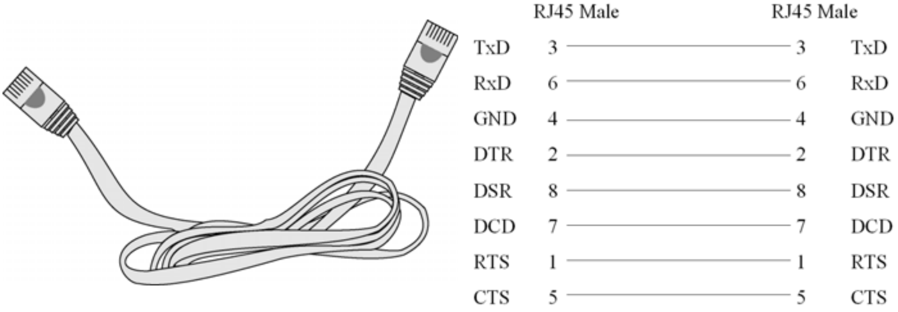


Figure C.9 - Cable 4 - Cyclades RJ-45 to Cyclades RJ-45, straight-through

Cable #5: Cyclades/Sun Netra Cable

This Adapter attaches to a Cat 3 or Cat 5 network cable. It is usually used in console management applications to connect Cyclades products to a Sun Netra server or to a Cisco product. This cable is included in Cable Package #2.

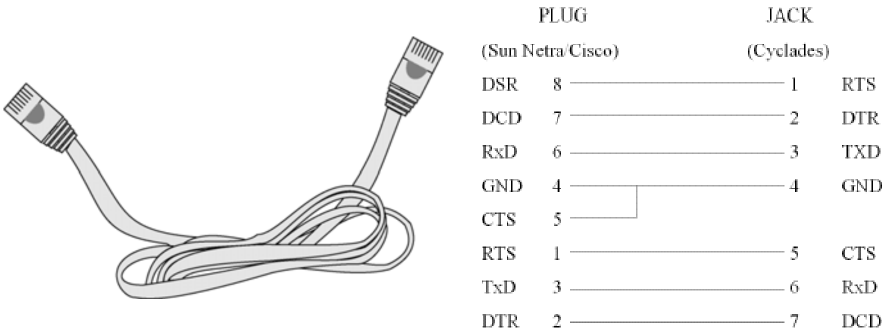


Figure C.10 - Cable 4 - Cyclades RJ-45 to Cyclades RJ-45, straight-through

Adapters

The following four adapters are included in the product box. A general diagram is provided below and then a detailed description is included for each adapter.

Loop-Back Connector for Hardware Test

The use of the following DB-25 connector is explained in the Troubleshooting chapter. It is included in both Cable Package #1 and #2.

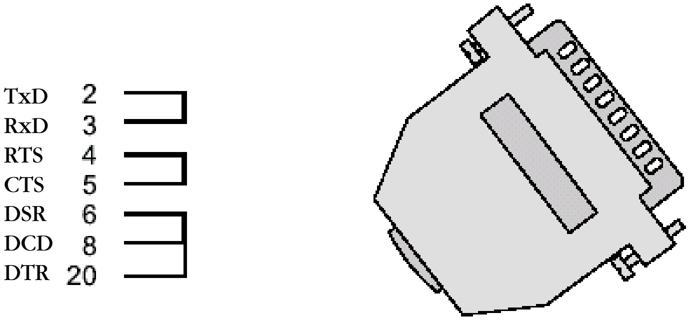


Figure C.11 - Loop-Back Connector

Cyclades\Sun Netra Adapter

This Adapter attaches to a Cat 3 or Cat 5 network cable. It is usually used in console management applications to connect Cyclades products to a Sun Netra server or to a Cisco product. At one end of the adapter is the black CAT.5e Inline Coupler box with a female RJ-45 terminus, from which a 3-inch-long black Sun Netra-labeled cord extends, terminating in an RJ-45 male connector. This adapter is included in Cable Package #2.

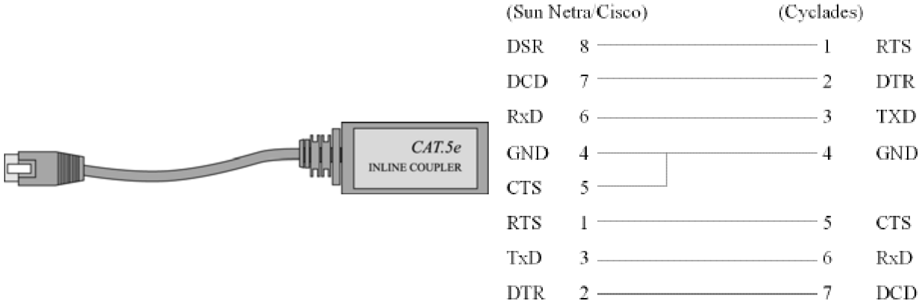


Figure C.12 - Cyclades\Sun Netra Adapter

RJ-45 Female to DB-25 Male Adapter

The following adapter may be necessary. It is included in Cable Package #1.

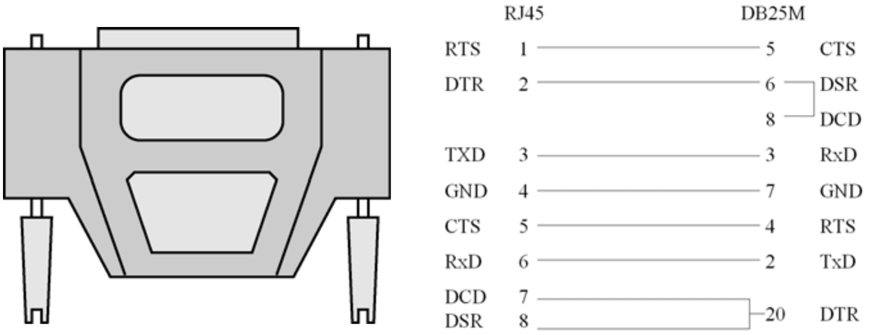


Figure C.13 - RJ-45 Female to DB-25 Male Adapter

RJ-45 Female to DB-25 Female Adapter

The following adapter may be necessary. It is included in Cable Package #1.

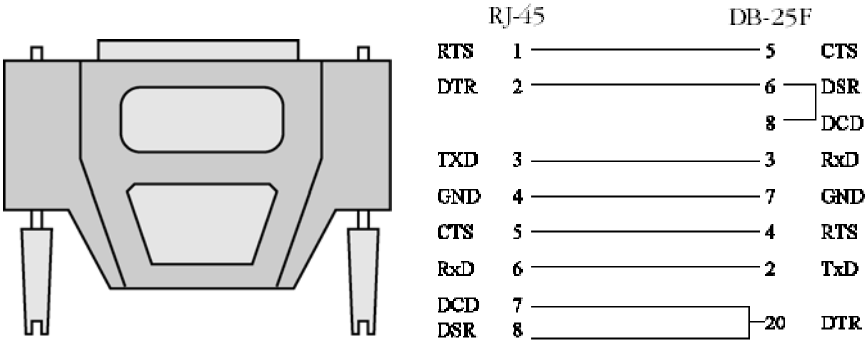


Figure C.14 - RJ-45 Female to DB-25 Female Adapter

RJ-45 Female to DB-9 Female Adapter

The following adapter may be necessary. This is included in Cable Package #1.

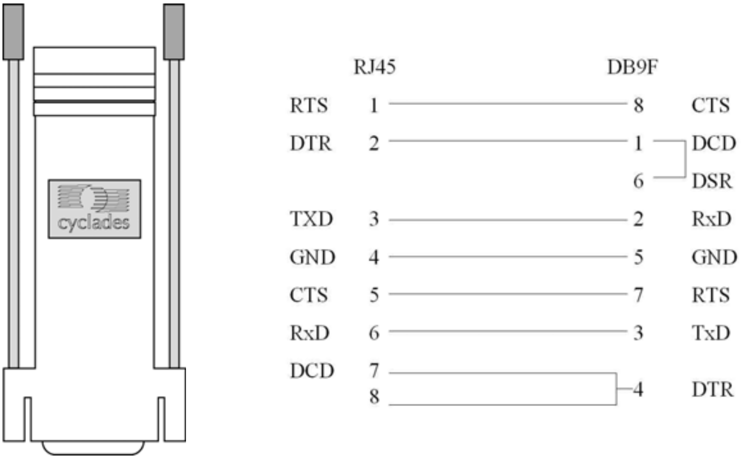


Figure C.15 - RJ-45 Female to DB-9 Female Adapter

C.4 ACS1-only Cabling Information

ACS1 Connectors

RS-485 Signal	Name/Function	Terminal Block pins
Chassis	Not in use	1
TXA-	Transmit Data - (A)	2
TXB+	Transmit Data + (B)	3
RXA-	Receive Data - (A)	4
RXB	Receive Data + (B)	5
Chassis	Not in use	6

Table C.16: RS-485 Pinout for the ACS1 - Connector pin assignment

ACS1-only Cabling Information

The RS-485 Standard

The RS-485 is another standard for serial communication and is available only in the ACS1. Different from the RS-232, the RS-485 uses fewer wires - either two wires (one twisted pair) for half duplex communication or four wires (two twisted pairs) for full duplex communication. Another RS-485 characteristic is the "termination." In a network that uses the RS-485 standard, the equipment is connected one to the other in a cascade arrangement. A "termination" is required from the last equipment to set the end of this network.

ACS1 Connectors

Although the RS-485 can be provided in different kinds of connectors, the ACS1 uses a 9-pin D-shaped connector (DB-9) and a Terminal Block with the pin assignment described below.

Cable #1: Terminal Block to Terminal Block, crossover half duplex

Application: It connects the ACS1 (serial port) to DTE RS-485 devices with half duplex communication.

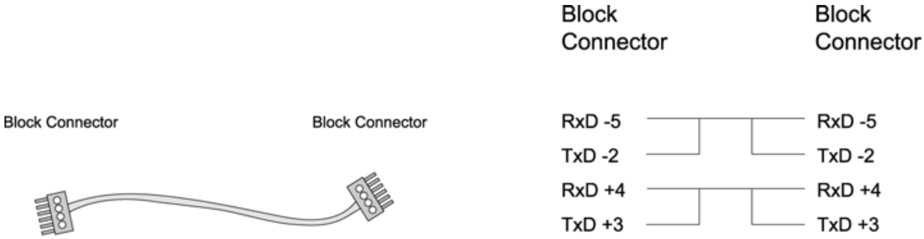


Figure C.17 - Cable 1 for the ACS1 - Terminal Block to Terminal Block, crossover half duplex

Cable #2: Terminal Block to Terminal Block, crossover full duplex

Application: It connects the ACS1(serial port) to DTE RS-485 devices with full duplex communication.

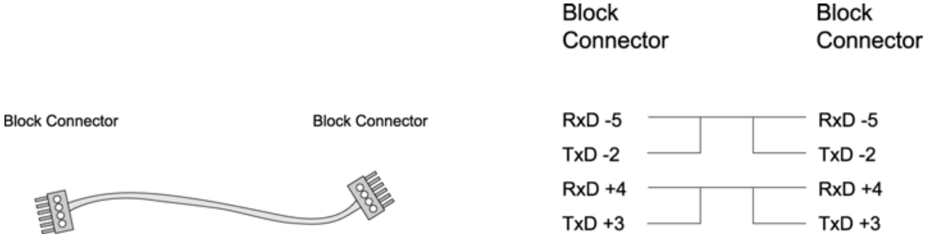
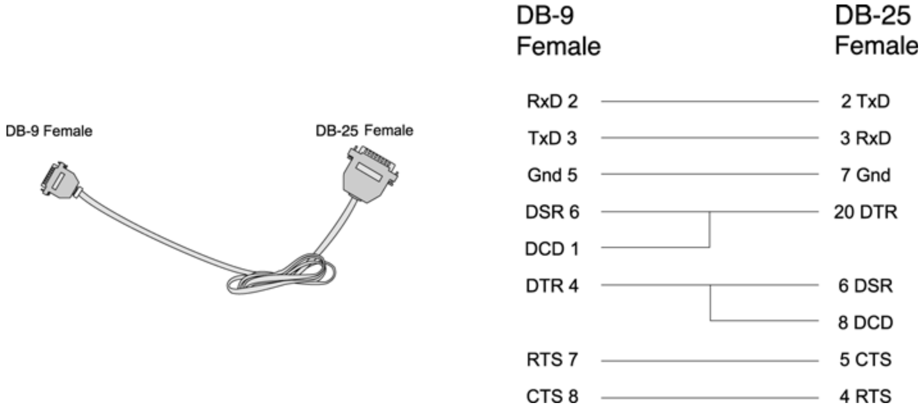


Figure C.18 - Cable 2ACS1 - Terminal Block to Terminal Block, crossover full duplex

Cable #3: DB-9 Female to DB-25 Female, crossover This cable connects the ACS1to console ports, terminals, printers and other DTE RS-232 devices. You will essentially have the cable shown in this picture:

Figure C.19 - Cable 3 for the ACS1 - DB-9 Female to DB-25 Female, crossover



This page has been left intentionally blank.

Appendix D
Copyrights

The Cyclades AlterPath Console Server is based in the HardHat Linux distribution, developed by Montavista Software for embedded systems. Additionally, several other applications were incorporated into the product, in accordance with the free software philosophy.

The list below contains the packets and applications used in the Cyclades AlterPath Console Server and a reference to their maintainers. The copyrights notices required in some packets are placed in the /COPYRIGHTS directory of the Cyclades AlterPath Console Server .

Bash

Bourne Again Shell version 2.0.5a. Extracted from the HardHat Linux distribution.
http://www.gnu.org/software/bash

Bootparamd

NetKit Bootparamd version 0.17
ftp://ftp.uk.linux.org/pub/linux/Networking/netkit

Busybox

BusyBox version 0.60.2
ftp://ftp.lineo.com/pub/busybox/

Cron

Paul Vixie's cron version 3.0.1.
paul@vix.com

DHCPCD

PhysTech DHCP Client Daemon version 1.3.20.p10.
http://www.phystech.com/download/dhcpcd.html

Flex

Flex version 2.5.4
vern@ee.lbl.gov
COPYRIGHT: This product includes software developed by the University of California, Berkeley and its contributors

GNU

The GNU project
http://www.gnu.org

HardHat Linux

MontaVista Software - HardHat version 2.1
http://www.montavista.com

IPSec

The Linux FreeS/WAN IPsec version 1.9.8
http://www.freeswan.org
COPYRIGHT: This product includes software developed by Eric Young (eay@cryptsoft.com)

IPtables

Netfilter IPtables version 1.2.2. Extracted from the HardHat Linux distribution.
http://www.netfilter.org

Linux Kernel

Linux Kernel version 2.2.17 2.4.18. Extracted from the HardHat Linux distribution
http://www.kernel.org

Net-SNMP

SourceForge Net-SNMP project version 5.0.3
http://sourceforge.net/projects/net-snmp/

NTP

NTP client
http://doolittle.faludi.com/ntpclient/

OpenSSH

OpenSSH version 3.5p1
http://www.openssh.org
COPYRIGHT: This product includes software developed by the University of California, Berkeley and its contributors.

OpenSSL

OpenSSL Project version 0.9.6g
http://www.openssl.org
COPYRIGHT: This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit. (http://www.openssl.org/)
COPYRIGHT: This product includes cryptographic software written by Eric Young (eay@cryptsoft.com)

PAM

Linux PAM version 0.75
http://www.kernel.org/pub/linux/libs/pam/

Portslave

SourceForge Portslave project version 2000.12.25. (modified). Includes pppd version 2.4.1 and rlogin version 8.10
http://sourceforge.net/projects/portslave/

RSYNC

rsync version 2.5.5
http://rsync.samba.org/rsync/

Syslog-ng

Syslog new generation version 1.5.17
http://www.balabit.hu/products/syslog-ng/

Tinylogin

TinyLogin version 0.80
ftp://ftp.lineo.com/pub/tinylogin/

UCD-SNMP

SourceForge Net-SNMP project version 4.2.4.pre1
http://sourceforge.net/projects/net-snmp/

WEBS

GoAhead WEBS version 2.1 (modified)
http://goahead.com/webserver/webserver.htm

ZLIB

zlib version 1.1.4
http://www.gzip.org/zlib/

Glossary

Authentication

Authentication is the process of identifying an individual, usually based on a username and password. In security systems, authentication is distinct from authorization, which is the process of giving individuals access to system objects based on their identity. Authentication merely ensures that the individual is who he or she claims to be, but says nothing about the access rights of the individual. (Source: www.webopedia.com)

Break Signal

A break signal is generated in an RS-232 serial line by keeping the line in zero for longer than a character time. Breaks at a serial console port are interpreted by Sun servers as a signal to suspend operation and switch to monitor mode.

Console Access Server (CAS)

A CAS has an Ethernet LAN connection and many RS-232 serial ports. It connects to the console ports of servers and networking equipment and allows convenient and secure access from a single location.

Console Port

Most of the equipment in a data center (servers, routers, switches, UPS, PBX, etc.) has a serial console port for out-of-band management purposes.

Cluster

A cluster is a group of one or more computers working as a group to execute a certain task. From the user standpoint, a cluster acts as a large computer system.

Flash

Flash refers to a type of memory that can be erased and reprogrammed in units of memory known as blocks rather than one byte at a time; thus, making updating to memory easier.

In-band network management

In a computer network, when the management data is accessed using the same network that carries the data, this is called “in-band management.”

IP packet filtering

This is a set of facilities in network equipment that allows the filtering of data packets based on source/destination addresses, protocol, TCP port number and other parameters. Packet filtering is one of the main functions of a firewall.

KVM Switch (KVM)

Keyboard-Video-Mouse Switches connect to the KVM ports of many computers and allow the network manager to access them from a single KVM station.

Mainframe

Large, monolithic computer system.

MIBs

Management Information Bases. SNMP-compliant devices, called agents, store data about themselves in MIBs and return this data to the SNMP requesters.

Out-of-band network management

In a computer network, when the management data is accessed through a network that is independent of the network used to carry data, this is called "out-of-band network management."

Off-line data buffering

This is a CAS feature that allows capture of console data even when there is no one connected to the port.

Profile

Usage setup of the AlterPath Console Server : either as a Console Access Server (CAS), a Terminal Server, or a Remote Access Server.

RADIUS

Protocol between an authentication server and an access server to authenticate users trying to connect to the network.

RISC

Reduced Instruction Set Computer. This describes a computer processor architecture that uses a reduced set of instructions (and achieves performance by executing those instructions very fast.) Most UNIX servers (Sun Sparc, HP, IBM RS6000, Compaq Alpha) were designed with a processor using a RISC architecture. The Intel ® x86 architecture.

RS-232

A set of standards for serial communication between electronic equipment defined by the Electronic Industries Association in 1969. Today, RS-232 is still widely used for low-speed data communication.

Secure Shell (SSH)

SSH has the same functionality as Telnet (see definition below), but adds security by encrypting data before sending it through the network.

Server Farm

A collection of servers running in the same location (see Cluster).

SNMP

Short for Simple Network Management Protocol, a set of protocols for managing complex networks. The first versions of SNMP were developed in the early 80s. SNMP works by sending messages, called protocol data units (PDUs), to different parts of a network. SNMP-compliant devices, called agents, store data about themselves in Management Information Bases (MIBs) and return this data to the SNMP requesters. (Source: Webopedia)

Telnet

Telnet is the standard set of protocols for terminal emulation between computers over a TCP/IP connection. It is a terminal emulation program for TCP/IP networks such as the Internet. The Telnet program runs on your computer and connects your PC to a server on the network. You can then enter commands through the Telnet program and they will be executed as if you were entering them directly on the server console. This enables you to control the server and communicate with other servers on the network. To start a Telnet session, you must log in to a server by entering a valid username and password. Telnet is a common way to remotely control Web servers. (from webopedia.com)

Terminal Server

A terminal server has one Ethernet LAN port and many RS-232 serial ports. It is used to connect many terminals to the network. Because they have the same physical interfaces, terminal servers are sometimes used as console access servers.

TTY

The UNIX name for the COM (Microsoft) port.

U Rack height unit

A standard computer rack has an internal width of 17 inches. Rack space on a standard rack is measured in units of height (U). One U is 1.75 inches. A device that has a height of 3.5 inches takes 2U of rack space.

List of Tables

This page has been left intentionally blank.

List of Figures

This page has been left intentionally blank.