Console Port Server
Command Reference Guide

Models AP9301, AP9302, AP9303
Software Version 2.6.0

American Power Conversion Corporation
www.apc.com
Release Date: December 2006
Part Number: 990-2406A
Table of Contents

Preface xv

Purpose ............................................................... xv
Audience and User Levels ....................................... xv
New Users ............................................................ xv
Power Users .......................................................... xv
How to use the CLI (Command Line Interface) .............. xvi
Modes of Operation .................................................. xvi
Keyword meanings ................................................... xvi
Interactive Mode ..................................................... xvii
CLI arguments and their meanings ............................. xix
Other important features of the CLI ......................... xix
List of CLI Keywords ............................................... xxi

How To Use This Guide ........................................... xxiii

Conventions and Symbols ......................................... xxiv
Fonts ................................................................. xxiv
Hypertext Links ....................................................... xxiv
Glossary Entries ..................................................... xxiv
Quick Steps .......................................................... xxiv
Parameter Syntax .................................................... xxv
Brackets and hyphens (dashes) ................................... xxv
Ellipses ................................................................. xxv
Pipes ..................................................................... xxv
Greater-than and less-than signs ............................... xxv
Spacing and separators ............................................. xxv

Cautionary and Instructional Information .................. xxvi

Chapter 1: Basic Network Configuration

Networking Settings ............................................... 1
Performing Basic Network Configuration Using the wiz Command 1
Log Into CPS Through the Console ............................. 1
Changing User Passwords ......................................... 1
Security Advisory .................................................... 2
Using the wiz Command to Configure Network Parameters .... 3
## Table of Contents

**Selecting A Security Profile** ................................................. 4  
- To select a security profile ............................................. 5  
- CLI mode ................................................................. 5  

**Enabling Serial Ports** ..................................................... 6  
- To Enable a Serial Port [VI method] .................................. 6  
- To Enable a Serial Port [CLI method] ................................. 7  

**Chapter 2: Device Access** 9  

**Accessing Serial Ports** ................................................... 9  
- Default Configuration Parameters .................................... 9  
- Opening or Closing a Telnet Session to a Serial Port .......... 9  
- Opening or Closing an SSH Session to a Serial Port .......... 10  
- Accessing Serial Ports using “ts_menu” .................... 10  
  - Calling ts_menu without arguments .......................... 11  
  - Calling ts_menu with arguments ............................ 11  
  - How to close the session from ts_menu  
    (from the console of your unit) ........................... 12  
  - How to close the session from ts_menu  
    (from a Telnet/SSH session to your unit) .......... 13  
  - CLI Mode—ts_menu .............................................. 13  

**Data Buffering** ............................................................. 14  
- Ramdisks ................................................................. 15  
- Linear vs. Circular Buffering ...................................... 15  
- How to Configure Data Buffering ................................. 16  
  - VI mode—parameters involved and passed values .......... 16  
  - CLI method—data buffering .................................. 18  

**Menu Shell** ................................................................. 19  
- How to Use ............................................................. 20  
- How to Configure .................................................... 20  
  - Setting up the menu shell ................................... 20  
  - Assigning ports to the menu shell ......................... 21  
  - CLI method—Terminal Profile Menu ...................... 22  

**Clustering Using Ethernet Interface** ............................... 23  
- How to Configure Clustering ....................................... 23  
  - VI mode ............................................................ 23  

**Clustering Using NAT (Enhanced)** .............................. 28  
- New Parameters and Commands .................................... 28  
  - Examples: ........................................................... 29  
- How It Works .......................................................... 30
## General Configuration
- Master box configuration example 31
- Slave-1 box configuration example 34
- Slave-2 box configuration example 34
- Slave-3 box configuration example 35
- Example of starting CPS session commands 35
- CLI method—clustering 36

### Chapter 3: Authentication 39

#### Device Authentication 39
- How to Configure Device Authentication Using VI Mode—Parameters Involved and Passed Values 39
- How to Configure Device Authentication Using CLI Method—Authentication 42
  - To configure the authentication type for the serial ports 42
  - To configure user access to the serial ports 43
  - To configure authentication type for device console access 44
  - To configure an authentication server 44
- Access Control Using the Radius Attribute NAS-Port-id 45
- NIS Client 46
  - NIS client configuration 46
  - How to test the configuration 47
  - nsswitch.conf file format 47
  - Examples 48

#### Kerberos Authentication 49
- Kerberos Server Authentication with Tickets Support 49
  - How Kerberos works 49
- Configuring CPS to Use Kerberos Tickets Authentication 50
  - CPS configuration 50
  - Test the configuration 52
- Kerberos Server Authentication 53

#### LDAP Authentication 55
- Configuring an LDAP Server on Linux 55
- Configuring the APC Console Port Server Side 57
- Active Directory 58
  - Parameters that must be set in the /etc/ldap.conf file 58
- Enabling TACACS+ Authorization for Serial Ports 58
- Configuring Authorization with a TACACS+ Server [CLI] 59
- Configuring Authorization with a TACACS+ Server [vi] 59
  - Setting user authorization permissions on the TACACS+ Server [vi] 59
Group Authorization ................................. 60
  Configuring a TACACS+ Authentication Server .......... 61
    Configuring the authorization on CPS to access the serial ports [CLI] .......................... 62
  Configuring a RADIUS Authentication Server .......... 62
    Configuring the authorization on CPS to access the serial ports [CLI] .......................... 63
  Configuring an LDAP Authentication Server .......... 63
    Configuring the authorization on CPS to access the serial ports [CLI] .......................... 64

Linux-PAM ........................................ 64
  The Linux-PAM Configuration Directory .............. 65
    Configuration file syntax .......................... 66
  Module Path ...................................... 68
  Arguments ....................................... 71

Shadow Passwords .................................. 73

Certificate for HTTP Security ......................... 73
  Procedure ....................................... 73

X.509 Certificate on SSH ............................ 75
  To Configure X.509 Certificate for SSH .............. 76
    VI mode ....................................... 76
    CLI mode ...................................... 76
    Script mode ................................... 77
  To Connect to CPS Using SSH X.509 Certificate ........ 77
  To Connect to the Serial Ports of the CPS using SSH X.509 Certificate .................. 78

Chapter 4: Network 79

Introduction ...................................... 79

Basic Network Settings ............................ 79
  Hostname ....................................... 79
    VI mode ....................................... 79
    CLI mode—hostname .............................. 79
  IP Address and Netmask ........................... 80
    VI method ..................................... 80
    CLI method—IP address .......................... 81
<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>DHCP Client</td>
<td>82</td>
</tr>
<tr>
<td>VI mode</td>
<td>82</td>
</tr>
<tr>
<td>Files related to DHCP:</td>
<td>84</td>
</tr>
<tr>
<td>CLI method—DHCP</td>
<td>84</td>
</tr>
<tr>
<td>Routes and Default Gateway.</td>
<td>85</td>
</tr>
<tr>
<td>How to Configure the Default Gateway of the CPS</td>
<td>86</td>
</tr>
<tr>
<td>VI mode</td>
<td>86</td>
</tr>
<tr>
<td>CLI method—routes</td>
<td>86</td>
</tr>
<tr>
<td>DNS Server and Domain Name.</td>
<td>87</td>
</tr>
<tr>
<td>VI mode</td>
<td>87</td>
</tr>
<tr>
<td>CLI method—DNS and Domain Name.</td>
<td>88</td>
</tr>
<tr>
<td>Bonding</td>
<td>88</td>
</tr>
<tr>
<td>VI mode</td>
<td>89</td>
</tr>
<tr>
<td>CLI method—bonding</td>
<td>89</td>
</tr>
<tr>
<td>Hosts</td>
<td>92</td>
</tr>
<tr>
<td>VI mode</td>
<td>92</td>
</tr>
<tr>
<td>CLI method—hosts</td>
<td>93</td>
</tr>
<tr>
<td>TCP Keepalive</td>
<td>93</td>
</tr>
<tr>
<td>How It Works</td>
<td>94</td>
</tr>
<tr>
<td>VI mode</td>
<td>94</td>
</tr>
<tr>
<td>CLI method—TCP Keep Alive</td>
<td>94</td>
</tr>
<tr>
<td>Filters and Network Address Translation</td>
<td>95</td>
</tr>
<tr>
<td>Description</td>
<td>95</td>
</tr>
<tr>
<td>Structure of the iptables</td>
<td>95</td>
</tr>
<tr>
<td>Table</td>
<td>95</td>
</tr>
<tr>
<td>Chain</td>
<td>96</td>
</tr>
<tr>
<td>Rule</td>
<td>96</td>
</tr>
<tr>
<td>Syntax</td>
<td>97</td>
</tr>
<tr>
<td>Command</td>
<td>97</td>
</tr>
<tr>
<td>Rule Specification</td>
<td>99</td>
</tr>
<tr>
<td>Match Extensions</td>
<td>102</td>
</tr>
<tr>
<td>TCP Extensions</td>
<td>102</td>
</tr>
<tr>
<td>UDP Extensions</td>
<td>103</td>
</tr>
<tr>
<td>ICMP Extension</td>
<td>104</td>
</tr>
<tr>
<td>Multiport Extensions</td>
<td>104</td>
</tr>
<tr>
<td>Target Extensions</td>
<td>104</td>
</tr>
<tr>
<td>LOG</td>
<td>105</td>
</tr>
<tr>
<td>REJECT (filter table only)</td>
<td>106</td>
</tr>
<tr>
<td>SNAT (NAT table only)</td>
<td>106</td>
</tr>
<tr>
<td>DNAT (NAT table only)</td>
<td>107</td>
</tr>
</tbody>
</table>
Start and Stop Services ............................................. 180
  Configuration Example ......................................... 182

Security Profiles ................................................. 183
  CLI method—selecting a predefined security profile ........... 185
  CLI method—configuring a custom profile ...................... 186

Chapter 6: Power Management 191

Power Management Configuration ............................... 192
  Prerequisites for Power Management ........................... 192
  Configuring Power Management ................................ 193
    VI method—involved parameters and passed values .......... 193
    CLI method—rPDU configuration ................................ 194
    Changing the rPDU password .................................. 196
  Accessing the Regular Menu from the Console Session ...... 198
  Using the Power Management Utility ......................... 199
  Manage Devices Plugged into a Single Outlet ............... 199
  Manage Devices Plugged into Multiple Outlets .............. 202
  To Manage Multiple rPDUs from the Command Line .......... 204
  To Manage Power Through the Console ....................... 205

Chapter 7: PCMCIA Cards Integration 211

Supported Cards .................................................. 211
  Tools for Configuring and Monitoring PCMCIA Devices ...... 211
  Ejecting Cards .................................................. 211

PCMCIA Network Device Configuration ...................... 212
  Ethernet PC Cards ............................................... 212
    VI method ...................................................... 212
    Removing the configuration from an Ethernet PCMCIA device 213
    CLI method—Ethernet PCMCIA ................................ 213
  Wireless LAN PC Cards ......................................... 214
    Removing the configuration from a wireless PCMCIA device 215
    CLI method—wireless PCMCIA ................................. 215
  Modem PC Cards ................................................ 216
    VI method ...................................................... 216
    Establishing a callback with your Modem PC card .......... 217
    CLI method—modem PCMCIA .................................. 221
  GSM Card Configuration ....................................... 222
    VI method ...................................................... 222
    CLI method .................................................... 222
CDMA Card Configuration .................................................. 224
  VI method .............................................................. 225
  CLI method ............................................................ 226
ISDN PC Cards .............................................................. 227
  VI method .............................................................. 227
  Establishing a callback with your ISDN PC card (method one) ... 228
  Establishing a callback with your ISDN PC card (method two) ... 230
  CLI method—ISDN PCMCIA ............................................. 232
Media Cards ................................................................. 233
  How it works ........................................................... 233
  VI method—configuration ............................................. 234
  CLI method—Media Cards PCMCIA .................................... 236
How to Save/Load Configuration to/from CF/IDE ...................... 237
  CLI method: backupconfig ............................................ 238
Generic Dial-Out ............................................................ 238
  Configuring the generic-dial.conf file ............................ 240
  Configuring Generic Dial-Out ...................................... 240

Chapter 8: Profile Configuration 245

The pslave.conf File ....................................................... 245
  pslave.conf Common Parameters .................................. 246
  pslave.conf CAS (Console Access Server) Parameters ......... 258
  pslave.conf TS (Terminal Server) Parameters .................. 273
  pslave.conf Dial-in Parameters .................................... 276
  pslave.conf Bidirectional Telnet Parameters .................... 277
    To configure Bidirectional Telnet ............................... 278
    CLI Method ......................................................... 278
    To configure a menu shell ...................................... 279
Using the CLI Interface to Configure
  Common Parameters .................................................. 279
    General State Parameters ...................................... 279
    Other State Parameters ........................................ 280
Examples for Configuration Testing ................................. 281
  Console Access Server .............................................. 281
  Terminal Server ..................................................... 285
  Dial-in Access ...................................................... 287
Chapter 9: Additional Features and Applications 289

Windows 2003 Server Management ............................. 289
  How It Works ..................................................... 289
  How To Configure .............................................. 297
    VI mode—parameters involved and passed values ........... 297
  Server Commands .............................................. 301

IPMI Configuration ............................................. 302
  How It Works .................................................. 302
  IPMI [ipmitool] ................................................. 303

Line Printer Daemon ........................................... 305

CAS Port Pool .................................................. 307
  How to Configure It .......................................... 308
    VI method ..................................................... 309

Billing ............................................................. 310
  General Feature Description ................................ 310
  How to Configure It .......................................... 310
    VI method—passed values and involved parameters ...... 310
  How It Works .................................................. 311
    Disk space issue ............................................. 312
  Billing Wizard ................................................ 312
    How to configure ............................................ 312
    To configure a port for billing ............................. 312

Appendix A: New User Background Information 315

User and Passwords ............................................. 315

View Logged-in Users And Their Actions ...................... 316

Linux File Structure .......................................... 316

Basic File Manipulation ....................................... 317

The VI Editor ................................................... 318

The Routing Table ............................................. 320
Secure Shell Session .................................................. 321
  The Session Channel Break Extension .......................... 322
    How it works in SSH Server (all.protocol is socket_ssh) ... 322
    How it works in SSH Client .................................. 322
    Configuring the Session Channel Break Extension in SSH Server 323

The Process Table .................................................. 324

TS Menu Script ..................................................... 325

Appendix B: Upgrades and Troubleshooting 327

Upgrades .............................................................. 327
  The Upgrade Process .............................................. 327
    CLI method—firmware upgrade ................................. 328

Troubleshooting .................................................... 330
  Flash Memory Loss ............................................... 330
  Hardware Test ..................................................... 331
    Port test ......................................................... 332
    To start the port test: ....................................... 332
    Port conversation ............................................. 332
    Test signals manually ....................................... 333
  Single-User Mode ................................................ 334
  Using a Different Speed for the Serial Console ............... 336
  Setting the Maximum Number of Bytes
    Received by the Interface .................................... 336
    To set a limit of bytes received by the interface per second: 337

LEDs ................................................................. 338
  CPU LEDs .......................................................... 338
  Rear Panel LEDs .................................................. 338
    Ethernet connector LED functionality ......................... 338
    Console connector LED functionality .......................... 339
    Serial connector LED functionality ........................... 339

Administration Parameters in the CLI Interface ............... 339
  Boot configuration parameters ................................ 339
  Administration menu ............................................. 340
Appendix C: Cabling and Hardware Information 341

General Hardware Specifications 341
   The RS-232 Standard 342
   Cable Length 343

Connectors 344
   Straight-Through vs. Crossover Cables 345
   Identifying the Cable or Adapter to Use 345
      Cables 346
      Adapters 346

Appendix D: Copyrights 349

Glossary 353
Preface

Purpose

This Reference Guide documents configuration and administration of the APC Console Port Server (CPS) using vi and CLI (Command Line Interface) methods.

VI is a text editor for UNIX type systems. Therefore, related configuration involves editing text files. All features in the CPS can be configured using the vi editor. For each configuration method, the features have an indicator that identifies whether the configuration is done using the vi editor or the CLI interface (when available). For further information about how to use the vi editor, consult Appendix A - New User Information.

Audience and User Levels

This command reference guide is intended for the users who are responsible for the deployment and day-to-day operation and maintenance of the CPS. 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 perform start-up of the CPS. 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. Before installing and configuring the CPS, you might want to review information such as common Linux/UNIX commands and how to use the vi editor. See “Appendix A: New User Background Information” on page 315. It is recommended that New Users configure the CPS using a Web browser following the User’s Guide, which documents the Web Interface. However, new users can also configure the CPS 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 configure the APC Console Port Server through the Web browser, vi, or CLI.

Note: The APC Console Port Server (CPS) is based on an embedded Linux operating system. Configurations are done using the vi text editor or the Command Line Interface (CLI). If you are new to Linux, it is advisable to refer to the APC CPS Installation, Administration, and User Guide, which documents the CPS Web Manager.
Purpose

---

Note: Appendix A—New User Background Information has a section dedicated to the vi text editor and its commands.

---

How to use the CLI (Command Line Interface)

Throughout the manual a number of features can be configured using the CLI interface instead of the vi editor (Web Manager Interface). The CLI tool is preferred by many network and system administrators since it allows automation of configuration through scripting and provides a simple way to document and record a system’s configuration. This section introduces the CLI tool and provides information on how to use the interface.

Modes of Operation

1. **Interactive mode**—commands will be read from standard input.
2. **File batch mode**—commands will be taken from a file (-f <file>).
3. **Batch mode**—commands will be taken from the command line arguments
   - Each invocation of the CLI should return a value to the shell indicating success or failure of the command.
   - Each invocation of the CLI should return a text string if any error occurred. If the command is successful then no text is returned.

   For example, from the CPS prompt, to change the hostname you can directly type:

   
   ```
   [root@CPS root]#/bin/CLI config network hostsettings hostname <host_name>
   ```

   Both modes are oriented by keywords that allow the moving from one state to another. Each state will have a specific set of keywords attached to it.

   **IMPORTANT:** Strings with spaces in CLI Batch Mode must be quoted with single and double quotes. To enter strings with spaces using the Batch Mode, the user must type "'<string1 string2>'" . For example:

   ```
   # CLI config network hostsettings banner "'Welcome to CPS'"
   ```

Keyword meanings

1. Changing from one state to another. For example: entering in configuration mode or exiting from configuration mode. Once the CLI goes to one state it will remain in this state until another keyword is entered to change the state.
2. Specifying a function or an action to be performed. For example: Apply changes (runconfig), Save changes into flash (savetoflash), back up configuration script (backupconfig), upgrade firmware (upgradefw), connect to a console (console), etc.

3. Specifying a set of parameters to be configured. For example:

```plaintext
cli> config security
security> adduser username john password john12 admin yes biouser no shell /bin/sh
```

4. Specifying a parameter to be changed. For example:

```plaintext
cli> network hostsettings
hostsettings> dhcp yes
```

**Interactive Mode**

The following features of the CLI ease its use:

1. AutoComplete of keywords using the tab key.

2. Cursor movement keys:
   - `<Ctrl> a` - Move to the start of the current line.
   - `<Ctrl> e` - Move to the end of the line.
   - `<Ctrl> b` - Move back a character (same as `<left arrow key>`).
   - `<Ctrl> f` - Move forward a character (same as `<right arrow key>`).
   - `<Esc> b` - Move back to the start of the current or previous word. Words are composed of letters and digits.
   - `<Esc> f` - Move forward to the end of the next word. Words are composed of letters and digits.
   - `<Ctrl> l` - Clear the screen and redraw the current line, leaving the current line at the top of the screen.

3. Command History keys:
   - `<Ctrl> n` - Move `forward` through the history list, fetching the next command (same as `<down arrow key>`).
   - `<Ctrl> p` - Move `back` through the history list, fetching the previous command (same as `<up arrow key>`).
The command history buffer is only available for the last 500 commands in the current session. The history is cumulative, so terminating the session will not clear the buffer. A user can login to the CLI and review the commands entered by a previous user.

4. Text-changing keys:
   - \(<Ctrl> d\) - Delete the character under the cursor (same as \(<delete key>\).
   - \(<Ctrl> h\) - Same as \(<Backspace key>\).
   - \(<Ctrl> k\) - Kill the text from the cursor to the end of the line.
   - \(<Ctrl> u\) - Kill backward from the cursor to the beginning of the current line.
   - \(<Ctrl> w\) - Kill the word behind the cursor.
   - \(<Esc> d\) - Kill from the cursor to the end of the current word, or if between words, to the end of the next word.
   - \(<Esc> <tab>\) - View the current value of the parameter keyword entered. You can then edit the value. For example, to display the current value for domain and edit it, type the following:

   **CLI> config network hostsettings**  
   **hostsettings> domain** [press \(<Esc> <Tab>\) now]

   You see:

   **hostsettings> domain cps.com**

5. Special Keywords:
   These words are global and can be used in any state. For these special keywords to work, they must be entered first (before the rest of the keywords for that state), or they must be the only word in the command line.
   - **quit**—Finish the CLI session.
   - **return**—Go back to the previous state.
   - **info**—View the help information available for the current state. For example, if you enter the network mode and type ‘info’, a brief overview about network configuration is presented. Or you can type ‘info config network’ from the **CLI>** prompt. Depending on the screen size of the user’s current shell, you may page through the info. If the informational text lines exceeds the number of lines capable of being shown in the screen, the user will get the option to type ‘m’ for more, ‘b’ for back, or ‘q’ for quit.
• *show*—Display the configuration parameter(s). This keyword is valid only in the configuration state. The following example displays some configuration parameters for port 1:

```
cli> config physicalports 1
Ports[1]> show general
```

general:
alias:
protocol: consoletelnet
speed: 9600
flow: none
parity: none
datasize: 8
stopbits: 1

**CLI arguments and their meanings**

When calling the CLI interface by typing CLI in the shell prompt, you can pass some arguments to it. Here is a brief description:

- `-q`—suppresses the output of error messages from the CLI.
- `-t <time>`—the timeout in minutes. Default: 10 minutes.
- `-T`—disable idle timeout. Same as "-t 0"
- `-s`—save changes to flash (same as savetoflash keyword) (batch mode only)
- `-r`—activate changes (same as runconfig keyword) (batch mode only)
- `-f <filename>`—executes the commands in the file `<filename>`

**Other important features of the CLI**

1. Only one user logged in as “root” or “admin” can have an active CLI or Web Manager session. A second user who connects through the CLI or the Web Manager as the “root” or “admin” has a choice to abort the session or close the other user’s session.

**Note:** If there are cron jobs running through automated scripts, a “root” or “admin” user login can cause the automated cron jobs to fail. Make sure that the users with administrative privileges are aware of this.
2. CLI has 3 possible user levels:
   
   - Root user (Linux root user) — Has access to the full functionality of the CLI. Has ‘shell’ command in the CLI that allows the user to have access to the CPS Linux shell prompt. (See note below)
   
   - Admin — Has access to the full functionality of the CLI except for the ‘shell’ command. Admin users cannot access the CPS Linux shell prompt. (See note below).
   
   - Regular user — Has access to a limited functionality of the CLI. Only has access to `cli-` applications functionality.

   **Note:** Users can change the login shell in `/etc/passwd` to execute `/bin/CLI` so that they enter CLI mode when they log on to the CPS. If the root user is configured to have `/bin/CLI` as their default shell, the root user can still access the CPS shell prompt by executing the command ‘shell’ from the CLI. Any other users who configure `/bin/CLI` as their default shell won’t have the ‘shell’ command, so they won’t be able to access the CPS shell prompt.

3. The CLI generates syslog messages when the user opens or closes a session and for each command executed. Examples:

   - Apr 19 17:51:44 src_dev_log@swes-129 CLI[413]: User root starts a interactive CLI session.
   - cli>config
   - Apr 19 16:18:02 src_dev_log@swes-129 CLI[412]: User root executed config
   - [config]config>
   - Apr 19 16:28:02 src_dev_log@swes-129 CLI[412]: Session closed due idletimeout
   - Apr 19 17:54:23 src_dev_log@swes-129 CLI[413]: User root executed [quit]
   - Apr 19 17:54:23 src_dev_log@swes-129 CLI[413]: User root finishes the CLI session

4. The CLI will write every command executed in interactive mode in the file "~/.history". This file keeps the last 1000 commands executed in any CLI session.
## List of CLI Keywords

**Table i-1:** Keywords accessible through the CLI interface

<table>
<thead>
<tr>
<th><strong>administration</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>backupconfig</td>
<td>Restore/save configurations from/to an FTP server or a storage device.</td>
</tr>
</tbody>
</table>
| sessions           | Manage sessions:  
|                    | kill—End a session to a specific serial port.  
|                    | list—Display the list of current serial port connections. |
| upgradefw          | Upgrade the firmware. Provide a domain name or the IP address of the server |

<table>
<thead>
<tr>
<th><strong>applications</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>connect</td>
<td>Access the console server connection menu.</td>
</tr>
<tr>
<td>pm</td>
<td>Access the CPS power management menu.</td>
</tr>
<tr>
<td>view</td>
<td>Display the data buffer files for a serial port.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>config</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>administration</td>
<td></td>
</tr>
<tr>
<td>bootconfig</td>
<td>Configure boot configuration parameters.</td>
</tr>
<tr>
<td>date/time</td>
<td>Set the date and time.</td>
</tr>
<tr>
<td>notifications</td>
<td>Set up alarm notifications.</td>
</tr>
<tr>
<td>ntp</td>
<td>Configure Network Time Protocol.</td>
</tr>
<tr>
<td>timezone</td>
<td>Select and set a GMT zone.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>application</th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>terminalmenu</td>
<td>Configure a terminal profile menu.</td>
</tr>
<tr>
<td>discardchanges</td>
<td>Cancel the configuration changes.</td>
</tr>
</tbody>
</table>
Table i-1: Keywords accessible through the CLI interface

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ipmi</td>
<td>Configure devices configured with IPMI.</td>
</tr>
<tr>
<td>network</td>
<td>Configure Network Parameters.</td>
</tr>
<tr>
<td>hosttable</td>
<td>Add or delete a host from the table.</td>
</tr>
<tr>
<td>pcmcia</td>
<td>Configure supported PCMCIA cards.</td>
</tr>
<tr>
<td>snmp</td>
<td>Configure SNMP server.</td>
</tr>
<tr>
<td>stroutes</td>
<td>Set up routes manually for data routing to other subnets.</td>
</tr>
<tr>
<td>syslog</td>
<td>Set up a syslog server for logging system messages.</td>
</tr>
<tr>
<td>vpn</td>
<td>Set up a VPN connection.</td>
</tr>
<tr>
<td>physicalports</td>
<td>Configure serial ports individually or collectively.</td>
</tr>
<tr>
<td>restorefromflash</td>
<td>Restore the configuration saved in flash.</td>
</tr>
<tr>
<td>savetoflash</td>
<td>Save the configuration changes to flash.</td>
</tr>
<tr>
<td>security</td>
<td>Configure security profiles and authentication servers.</td>
</tr>
<tr>
<td>virtualports</td>
<td>Cascade multiple APC CPS console servers.</td>
</tr>
<tr>
<td>portStatus</td>
<td>Display the status on all serial ports.</td>
</tr>
<tr>
<td>shell</td>
<td>Open the command shell.</td>
</tr>
<tr>
<td>version</td>
<td>Display the CLI version</td>
</tr>
<tr>
<td>runconfig</td>
<td>Activate the changes.</td>
</tr>
<tr>
<td>info</td>
<td>Display a brief description of the current CLI parameter.</td>
</tr>
<tr>
<td>quit</td>
<td>Exit the CLI mode.</td>
</tr>
<tr>
<td>return</td>
<td>Go up one level in the CLI menu structure.</td>
</tr>
<tr>
<td>show</td>
<td>Display the current configuration information.</td>
</tr>
</tbody>
</table>
How To Use This Guide

This guide is organized into the following sections:

- **Chapter 1: Basic Network Configuration**—describes the basic configuration procedures to make the APC Console Port Server operational and available on the network. It includes configuring the network parameters, logging in, and selecting a security profile.

- **Chapter 2: Device Access**—describes 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 3: Authentication**—provides configuration instructions for different types of authentication available in the CPS. This chapter includes detailed information about the Linux-PAM module and Shadow Passwords.

- **Chapter 4: Network**—explains all network configuration settings.

- **Chapter 5: Administration**—contains systems management, administration, and maintenance features.

- **Chapter 6: Power Management**—describes features for those who have an APC Rack Power Distribution Unit (PDU) being controlled by the CPS.

- **Chapter 7: PCMCIA Cards Integration**—provides information about compatible PCMCIA cards and the respective instructions to make them work with the CPS.

- **Chapter 8: Profile Configuration**—outlines the main configuration file of the unit, explains each parameter of the pslave.conf file, and includes step-by-step examples for TS, CAS and RAS profiles.

- **Chapter 9: Additional Features and Applications**—describes special features and includes step-by-step instructions for setting up the features.

- **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 CPS.

- **Appendix C: Cabling and Hardware Information**—provides detailed information and pinout diagrams for cables used with the CPS.

- **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

Fonts
This guide uses a regular text font (Times New Roman) for most of the body text, and Courier New font for data that you would input (such as a command line instruction), and data that you would receive (such as an error message). For example:

# 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 a hypertext link in the PDF version of the manual, you will be taken to that section of the manual.

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 CPS 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:

1. Modify the pslave.conf file.

   You will modify four Linux files to let the CPS know about its local environment. Open the file pslave.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)

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 (-) or indicates a list of characters, the parameter can be one of the letters listed within the brackets. For 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. For example:

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

Pipes

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

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

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; and 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. There are three levels of information:

**Warning!** A very important 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 or expedites the use of the product.
Chapter 1: Basic Network Configuration

This chapter describes the procedures for setting up the basic network configuration to make APC Console Port Server available on the network. In addition, it provides procedures to login, change the default password, and setup the security profile.

See “Chapter 4: Network” on page 79 for detailed information about configuring network settings using the vi method or the CLI method.

1.1 Networking Settings

The following section describes how to configure the network parameters using the wiz command, vi, or CLI where applicable. The instructions assume that you are installing a new APC Console Port Server in your network, or you are restarting an existing unit from factory-default parameters.

Performing Basic Network Configuration Using the wiz Command

The following procedure assumes that a hardware connection is made between the CPS’s console port and the COM port of a computer.

Log Into CPS Through the Console

From your terminal emulation application, log into the console port as root.

```
CPS login: root
Password: apc
```

Important: for secure access to the connected servers or devices, change the default password “apc” before setting up the CPS.

Changing User Passwords

Change the “root” password. By default, /etc/passwd file has the user “root” with password “apc”. To change any user password, run the following command:

```
# passwd <user>
```
Security Advisory

The following Security Advisory appears the first time CPS receives power, or when the unit is reset to factory-default parameters. After you have configured the basic network settings, you must select a Security Profile before proceeding to other configuration procedures, such as user and port settings.

See Selecting A Security Profile to configure a profile using CLI. See the APC CPS Installation, Administration, and User’s Guide for detailed information on security profiles and configuration options using the Web Manager.

Important - Security Advisory!

Console Management provides critical access to management features of attached equipment. Please take the required precautions to understand the potential impacts this device may have to your SECURITY policies.

From factory, this device is configured as follows:

- Single password for ROOT;
- All serial port DISABLED;
- DHCP, Telnet, SSHv1 & SSHv2, and HTTP & HTTPS enabled.

APC STRONGLY recommends:

1. To change the ROOT password before setting up the box for secure access to the CPS equipment.

2. That you SELECT A SECURITY PROFILE to complete the INITIAL SETUP. Security is dependent on Policy and is Configurable to fit in environments with varying levels of Security. The APC Console Port Server provides three pre-set Security Levels: SECURED, MODERATE and OPEN, and in addition, the ability to set a CUSTOM Security Profile.

3. Do not leave the equipment idle WITHOUT selecting a SECURITY PROFILE.

4. To ENABLE Serial Ports and CONFIGURE them using Web UI or CLI. Refer to the Quick Start Guide or the User's Guide for Security Profile selection details and Serial Port configuration.
Using the wiz Command to Configure Network Parameters

1. Launch the Configuration Wizard by entering the wiz command.

   [root@CPS etc]# wiz

   The system brings up a configuration wizard banner similar to the following figure and runs the wizard.

   **********************************************
   *****C O N F I G U R A T I O N   W I Z A R D**
   **********************************************
   Current configuration:

   Hostname: CPS
   DHCP: disabled
   System IP: 192.168.48.11
   Domain name: apc.com
   Primary DNS Server: 192.168.44.21
   Second DNS Server: #
   Gateway IP: 192.168.48.1
   Network Mask: 255.255.252.0

   Set to defaults? (y/n) [n]:

2. At the prompt, enter n to change the defaults.

   Set to defaults (y/n)[n]: n

3. Press Enter to accept the default hostname, or enter your own hostname.

   Hostname [CPS]: APC_CPS

4. Press Enter to keep DHCP enabled, or enter “n” to specify a static IP address for the CPS.

   By default, the CPS uses the IP address provided by the DHCP server. If your network does not use DHCP, then CPS use the default IP address 192.168.160.10.

   Do you want to use DHCP to automatically assign an IP for your system? (y/n) [n] :
5. Change the default static IP address. (See your network administrator to obtain a valid IP address.)

System IP[192.168.160.10]: CPS_IP_address

6. Enter the domain name.

Domain name[apc.com]: domain_name

7. Enter the IP address for the Primary Domain Name Server (DNS).

Primary DNS Server[192.168.44.21]: DNS_server_IP_address

8. Enter the IP address for the gateway.

Gateway IP[eth0]: gateway_IP_address

9. Enter the netmask for the subnetwork.

Network Mask[#] : netmask

The network configuration parameters appear.

10. Enter y after the prompts shown in the following screen example.

Are all these parameters correct? (y/n) [n]: y
Do you want to activate your configurations now? (y/n) [y]: y
Do you want to save your configuration to Flash? (y/n) [n]: y

11. To confirm the configuration, enter the ifconfig command.

1.2 Selecting A Security Profile
You must select a security profile before you can proceed with configuration of the CPS. For detailed information on security profiles, see APC CPS Installation, Administration, and User’s Guide.
To select a security profile

Select a pre-defined Security Profile, or define a Custom profile for specific services. The available profiles are:

- **Secured**: Disable all protocols except SSHv2, HTTPS, and SSH to Serial Ports.
- **Moderate**: Enable SSHv1, SSHv2, HTTP, HTTPS, Telnet, SSH and Raw connections to Serial Ports, ICMP, and HTTP redirection to HTTPS.
- **Open**: Enable all services, Telnet, SSHv1, SSHv2, HTTP, HTTPS, SNMP, RPC, ICMP and Telnet, SSH and Raw connections to Serial Ports.
- **Default**: Set the profile to the same configuration as Moderate.
- **Custom**: Enable or disable individual protocols and services, and configure access to ports.

CLI mode

1. Enter the CLI mode.

   [root@CPS etc]# CLI

2. At the prompt, enter the following string.

   cli > config security profile

   The following commands are available under the “profile” prompt:

   ```
   profile>
   custom  info    open    return    show
   default moderate quit secured
   ```

3. To configure a Default, Moderate, or Secured pre-defined security profile, enter the following string:

   ```
   profile> <moderate>
   or
   profile> <secured>
   or
   profile> <default>
   ```

4. To configure a custom security profile, navigate to the custom menu.

   ```
   profile > custom
   ```

5. From the custom menu enable or disable desired protocols using the following syntax.

   ```
   custom > [protocol] [yes/no]
   ```
To display the current configuration as shown in the following figure, enter:

```
custom>show
[custom]
ftp: no
telnet: no
[ssh]
..[ssh_x509]
   CA_file:
   hostkey:
      authorizedkeys:
sshv1: yes
sshv2: yes
sshd_port: 22
root_access: yes
snmp: no
.[web]
   http: yes
   https: yes
http_port: 80
https_port: 443
http2https: yes
rpc: no
ipsec: no
icmp: yes
.[ports]
   ssh2sport: yes
telnet2sport: yes
raw2sport: yes
auth2sport: no
bidirect: yes
```

### 1.3 Enabling Serial Ports

From the factory, the CPS is configured with all serial ports disabled.

**To Enable a Serial Port [VI method]**

1. From the terminal window, navigate to the portslave directory to edit the pslave.conf file.
   
   `[root@CPS ] cd /etc/portslave
   [root@CPS portslave]# vi pslave.conf`
2. Navigate to *Port-specific parameters* to uncomment the *sxx.tty* and enable the serial ports.

```plaintext
# Port-specific parameters
#
s1.tty  ttyS1
#s2.tty ttyS2
#s3.tty ttyS3
#s4.tty ttyS4
#s5.tty ttyS5
#s6.tty ttyS6
#s7.tty ttyS7
#s8.tty ttyS8
```

**To Enable a Serial Port [CLI method]**

1. Open the CLI interface by issuing the command:

   ```
   # CLI
   ```

2. To enable single or multiple serial ports, enter the following command:

   ```
   cli>config physicalports 1,2,3,4 enable yes
   ```
Chapter 2: Device Access

This chapter introduces all of the ways to access the serial ports of the CPS. From this point, this user guide assumes that the unit is properly configured using one of the possible profiles (CAS or TS). See “Chapter 8: Profile Configuration” on page 245 for more information about configuring a profile.

2.1 Accessing Serial Ports

There are four ways to access serial ports, depending on the protocol you configured for that serial port: setting all.protocol to socket_server for Telnet access, setting all.protocol to socket_ssh for SSH access, or setting all.protocol to socket_server_ssh for both.

An administrator can access the serial port by statically addressing it (using the TCP port number, alias name, or IP address) or by accessing the next free serial port available from an existent pool (using the pool's TCP port number, alias, or IP address).

Default Configuration Parameters

- DHCP enabled (if there is no DHCP Server, the IP address is 192.168.160.10, and the Netmask is 255.255.255.0)
- CAS configuration
- socket_server in all ports (access method is Telnet)
- 9600 bps, 8N1
- No Authentication

Opening or 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 <CPS hostname> <TCP port number>
```

where

- `<CPS hostname>` is the hostname configured in the workstation where the Telnet client will run (through /etc/hosts or DNS table). Alternately, use the IP address of the CPS (Ethernet 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 the factory, 7001 corresponds to serial port 1, 7002 corresponds to serial port 2, etc., and 3000 is a pool with all serial ports.

To close the Telnet session, press the Telnet hotkey configured in the Telnet client application ("Ctrl", in many applications) and "q" to quit.
Opening or Closing an SSH Session to a Serial Port

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

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

- `<Username>`—the user configured to access that serial port. It is present either in the local CPS 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.
- `<CPS hostname>`—the hostname configured in the workstation where the SSH client will run (through /etc/hosts or DNS table). Alternately, use the IP address of the CPS (Ethernet interface) configured by the user or learned from DHCP.

To close the SSH session, press the hotkey defined for the SSH client followed by a dot “.” The default is "~."

Note: Make sure you enter the escape character followed by a dot “.” at the beginning of a line to close the SSH session.

Accessing Serial Ports using “ts_menu”

ts_menu is an application that facilitates connection to the serial ports. The following are the methods of executing the ts_menu command.

- Calling ts_menu without specifying any arguments.
- Calling ts_menu with command line arguments
- Using CLI to call ts_menu.
Calling ts_menu without arguments

To access the serial port (Telnet or SSH) using `ts_menu`, log in to the CPS unit and, after receiving the shell prompt, run `ts_menu`. The servers (aliases) or serial ports are shown as options to start a Telnet or SSH connection. For example:

![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:

Calling ts_menu with arguments

Apart from calling `ts_menu` with no arguments (which directs the user to the traditional ts_menu interface), this application can be used with the following command line arguments:

```
ts_menu [-u<user>] [-l[c]] [-ro] [-s] [-auth] [<console port>]
```

The meaning of each argument is:

- `-u<user>`—Invokes ts_menu as the user named by `<user>`. The user must enter a password. The user has access only to the authorized serial ports.
- `-l[c]`—Generates a list of all ports to which the user has access, then terminates. Presents any defined port aliases. For the remote ports (clustering), if the port aliases are not defined they are shown as "ip_addr:port" (ip_addr referring to the slave CPS). By default, ports are displayed in alphabetical order, but if the "c" flag is also specified, the listing is sorted by console server (with master unit listed first).
- `-ro`—Invokes ts_menu in read-only mode. This argument works even if the user is the only user logged into a certain port. In this mode, the user can connect to any port to which he has access, but the user cannot make changes to the port. He is in sniff mode. If the user attempts to interact with the port, the message “Read-only mode” displays.

Note: A real sniff session (the user is not the first user to log in to a certain port) is only allowed if the user is authorized.

- `-s`—Invokes ts_menu in a way that all ports (including slave CPSs) are presented in a single list sorted in alphabetical order. Not using this option causes the display to be as for the old implementation.
- `auth`—For backward compatibility, this option makes the new ts_menu implementation behave as the old one so that authentication is performed again to access each port.
- `<console port>`—If issued, produces a direct connection to that port. If the user does not have access to that port or the port does not exist, the application returns the message "console not found" and terminates. `<console port>` can be the port alias or the port number. In case of clustering, the port number must include a reference to the slave CPS as "host:port" (where host is the slave hostname or IP address).

Other options:
- `-p`—Display the TCP port
- `-P`—Use the TCP port instead of the IP address
- `-i`—Display the Local IP assigned to the serial port
- `-s`—Show the ports in a sorted order
- `-u <name>`—Username to be used in SSH/Telnet command
- `-U`—Always ask for a username
- `-e `<[^]char>`—Escape character used by Telnet or SSH

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

To close the session from the ts_menu:

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 `[^].` When you enter the escape character, the following is shown:
   
   Console escape. Commands are:
   - `l`— to line mode
   - `c`—go to character mode
   - `z`—suspend Telnet
   - `b`—send break
   - `t`—toggle binary
   - `e`—exit Telnet

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/SSH session to your unit)

Make sure a different escape character is used for exiting your Telnet/SSH session; otherwise, when you exit from the session created through the ts_menu, you will close your entire Telnet session to your unit.

To assign a unique escape character, use the \(-e\) option the first time you access your unit with Telnet or SSH. For example, to set Ctrl-? as the escape character, type the following:

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

To exit from the session created through the ts_menu, enter the escape character.

To exit from the entire Telnet session to your unit, type the escape character you configured. To exit from the entire SSH session to your unit, type the escape character you configured plus the character "."(dot).

**Note:** To close an SSH session, you must enter the escape character followed by a “." at the beginning of a line.

CLI Mode—ts_menu

You can call ts_menu from the CLI interface.

1. Open the CLI interface by issuing the command:
   
   ```
   # CLI
   ```

2. Call the menu. To call the ts_menu, access the following menu:

   ```
   cli> applications connect [Enter]
   ```

   A screen similar to the following appears:

<table>
<thead>
<tr>
<th>Serial Console Server Connection Menu for your Master Terminal Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 PM      2 ttyS3      3 ttyS4      4 ttyS5</td>
</tr>
<tr>
<td>5 ttyS6    6 ttyS7      7 ttyS8      8 ttyS9</td>
</tr>
<tr>
<td>9 ttyS10   10 ttyS11    11 ttyS12    12 ttyS13</td>
</tr>
<tr>
<td>13 ttyS14  14 ttyS15    15 ttyS16</td>
</tr>
</tbody>
</table>

   Type 'q' to quit, 'b' to return to previous menu, a valid port[1-15], or anything else to refresh:
• To see the “connect” options, press [tab] twice at the following prompt:
  cli > applications > connect

  The following options display:
  consolename list readonly

• To display a list of the available ports, run the following command.
  cli > applications > connect list

• To connect to the console of a device in a read-only mode, run the following command.
  cli > applications > connect readonly consolename <consolename>

  The connection is made to the device and the message “Read only mode”
  displays.

• To make a direct connection to the console of a device run the following command.
  cli > applications > connect consolename <consolename>

3. Exit the CLI mode.

  To exit the CLI mode and return to CPS’s shell, type the following command:
  cli> quit

2.2 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 configure,
and syslog-ng.conf file should be set accordingly. (Please see “Syslog-ng” on page 137
for the syslog-ng configuration file.) For the file, all.data_buffering is the parameter to configure.

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>

For example, if data buffering is turned on for port 1, the data will be stored in the file `ttyS1.data` in local directory `/var/run/DB` or in remote path name and server indicated by `conf.nfs_data_buffering`.

**Ramdisks**

Data buffering files are created in the directory `/var/run/DB`. If the parameter `s<nn>.alias` is configured for the port `<nn>`, the alias is used. For example, if the alias is 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`. The default is cir.
How to Configure Data Buffering

VI mode—parameters involved and passed values

To configure Data Buffering:

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 Table 2-1.

Table 2-1: Data buffering parameters in `/etc/portslave/pslave.conf` file

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>all.data_buffering</td>
<td>A non-zero value activates data buffering (local or remote, according to the setting for the parameter conf.nfs_data_buffering). If local data buffering is enabled, a file is created on the CPS; if remote data buffering is enabled, a file is created through NFS in a remote server. All data received from the port is captured in this file. For local data buffering, this parameter defines the maximum file size (in bytes). For remote data buffering, this parameter is just a flag to activate (greater than zero) or deactivate data buffering. For local data buffering, 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). For remote data buffering, there is no maximum file size other than the size imposed by the remote server (linear file). You can view this file using standard UNIX tools (cat, vi, more, tail, etc.). Size is in BYTES, not kilobytes.</td>
</tr>
<tr>
<td>conf.nfs_data_buffering</td>
<td>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 <code>/var/run/DB</code>. 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.).</td>
</tr>
</tbody>
</table>
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 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"

all.syslog_buffering

When this parameter is a non-zero value, 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, see “Syslog-ng” on page 137.

---

**Table 2-1:** Data buffering parameters in /etc/portslave/pslave.conf file

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>all.DB_mode</td>
<td>When configured as <em>cir</em> for circular format, the buffer is like a revolving file that is overwritten whenever the limit of the buffer size (as configured in <em>all.data_buffering</em> or <em>s&lt;n&gt;.data_buffering</em>) is reached. When configured as <em>lin</em> 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 <em>all.flow</em> or <em>s&lt;n&gt;.flow</em> 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 resume. If <em>all.flow</em> or <em>s&lt;n&gt;.flow</em> is set to none, linear buffering isn’t possible. Default is <em>cir</em>.</td>
</tr>
<tr>
<td>all.DB_user_logs</td>
<td>When &quot;on&quot;, 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) &quot;&lt;connect&gt; [timestamp] [username] [session type] &lt;/connect&gt;&quot; 2) &quot;&lt;disconnect&gt; [timestamp] [username] &lt;/disconnect&gt;&quot; when [timestamp] = &quot;YYYY-MM-DD hh:mm:ss&quot; [session type] = &quot;Read/Write&quot; or &quot;Read Only&quot;</td>
</tr>
<tr>
<td>all.syslog_buffering</td>
<td>When this parameter is a non-zero value, 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 <em>conf.DB_facility</em>. The file /etc/syslog-ng/syslog-ng.conf should be set accordingly for the syslog-ng to take some action. For more information, see “Syslog-ng” on page 137.</td>
</tr>
</tbody>
</table>
2. Activate and save the changes made.
   
   To activate the changes, run the command:
   
   # runconf
   
   To save the changes, run the command:
   
   # saveconf

**CLI method—data buffering**

1. Open the CLI interface by issuing the command:

   # CLI

---

### Table 2-1: Data buffering parameters in /etc/portslave/pslave.conf file

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>all.syslog_sess</td>
<td>This parameter determines whether syslog is generated when a user is connected to the port. By default, syslog is always generated regardless of whether the user is connected to the port or not. CPS administrators can disable syslog message generation when 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.</td>
</tr>
<tr>
<td>all.dont_show_DBmenu</td>
<td>When 0, a menu with data buffering options displays when a user connects to a port with a data-buffering file that is not empty. 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 it is not empty. When 3, the data buffering menu is shown, but without the erase and show-and-erase options.</td>
</tr>
<tr>
<td>all.DB_timestamp</td>
<td>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.</td>
</tr>
</tbody>
</table>
2. Configure data buffering.

Data buffering parameters are under the following menu:
cli>config physicalports all databuffering

Configurable parameters are:

• buffersyslogevertime—If YES is chosen, Syslog will be buffered at all times. If NO is chosen, it will be buffered only when nobody is connected to the port.
• nfspath—Defines the NFS path.
• syslogserver—Defines the IP address of the Syslog server.
• filesize—Defines the maximum size of the data buffer file. This parameter must be greater than zero; otherwise, all parameters relating to data buffering are disregarded.
• showmenu—Controls the DB menu options. Valid values are: yes, no, noerase, file.
• syslogsize—Maximum size of syslog data buffer message.
• mode—Chooses between circular or linear data buffering.
• syslogfacility—Defines the facility number for messages generated by the CPS to be sent to the Syslog server.
• timestamp—Choose YES to enable timestamp or choose NO to disable it.

3. Activate the configuration.
   cli> config runconfig

4. Save the configuration.
   cli> config savetoflash

5. Exiting the CLI mode.
   To exit the CLI mode and return to CPS’s shell, type the following command:
   cli> quit

2.3 Menu Shell

This application allows you to customize the menu presented to users when they connect to the CPS from a dumb terminal. The menu can be set up to allow users to connect to different servers; thereby, making it easy for users to connect to those servers on the LAN.
How to Use

Once the appropriate configurations are complete, the user will connect to the CPS 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

The configuration for this feature is divided into two parts. First, use the `menush_cfg` utility to assign the users who will access the Menu Shell. Then, assign ports to the menu shell.

Setting up the menu shell

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 =>
```

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

The first question is:
Enter the name for the new menu option:
Enter a description of the host that will be accessed.

The second question defines the action that must be taken:
Enter the command for the new menu option:
The action can be \texttt{telnet host\_ip} or \texttt{ssh -l username host\_ip}, where \texttt{host\_ip} is the IP address of the server to connect to.

3. Save your changes by selecting the option \texttt{Save Configuration to Flash}.

\textbf{Assigning ports to the menu shell}

To configure which ports will prompt the menu shell and whether authentication is required, follow the steps below:

1. If no authentication is required to gain access to the menu:
   Configure the following parameters in \texttt{/etc/portslave/pslave.conf} for the ports that will use this menu shell. In this example, <x> is the port number being configured.
   \begin{verbatim}
   s<x>.protocol telnet
cnf.telnet /bin/menush
   s<x>.authtype none
   \end{verbatim}

2. If authentication is required to gain access to the menu:
   The user’s default shell must be modified to run the \texttt{/bin/menush}. For example, in \texttt{/etc/passwd} the shell should be changed as follows:
   \begin{verbatim}
   user:FrE6QU:505:505:Embedix User,,,:/home/user:/bin/menush
   \end{verbatim}

   In \texttt{pslave.conf}, the port where the serial terminal is attached must be configured for login with local authentication. In this example, <x> is the port number being configured.
   \begin{verbatim}
   s<x>.protocol login
   s<x>.authtype local
   \end{verbatim}

3. Activate and save your changes.
   To activate the changes, issue the command:
   \begin{verbatim}
   # runconf
   \end{verbatim}

   To save the changes, issue the command:
   \begin{verbatim}
   # saveconf
   \end{verbatim}
CLI method—Terminal Profile Menu

To define which servers the users can access:
1. Open the CLI interface by issuing the command:
   
   # CLI

2. Configure the terminal menu (the menu the user sees when connecting from a dumb terminal).
   Enter the terminal menu configuration:
   
   cli> config applications terminalmenu

   Define the menu title. For example:
   
   terminalmenu> menutitle "Available Servers"

   Create the entries. The example below will add an entry named “Server1”, which will open a Telnet connection to 192.168.100.3:
   
   terminalmenu> add actionname Server1 command "telnet 192.168.100.3"

   You can also open an SSH connection to the server by replacing “telnet host_ip” with “ssh -l username host_ip”.

3. Activate the configuration.
   
   cli> config runconfig

4. Save the configuration.
   
   cli> config savetoflash

5. Exit the CLI mode.
   To exit the CLI mode and return to CPS’s shell, type the following command:
   
   cli> quit
2.4 Clustering Using Ethernet Interface

Clustering is available for the CPS with firmware versions 2.1.0 and up. It allows the stringing of Terminal Servers so that one Master CPS can be used to access all CPSs on a LAN. The Master CPS can manage up to 1024 serial ports. An example with one Master CPS and two Slave CPSs is shown in the following figure:

![Diagram of Clustering](image)

Figure 2.1 - An example using the Clustering feature

How to Configure Clustering

The Master CPS 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**

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
   ```
Edit parameters according to the explanations provided in the following table.:

**Table 2-2:** Master configuration (where it differs from the CPS standard)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Examples/ Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>conf.eth_ip</td>
<td>Ethernet Interface IP address.</td>
<td>20.20.20.1</td>
</tr>
<tr>
<td>conf.eth_ip_alias</td>
<td>Secondary IP address for the Ethernet Interface (needed for clustering feature).</td>
<td>209.81.55.110</td>
</tr>
<tr>
<td>conf.eth_mask_alias</td>
<td>Mask for secondary IP address above.</td>
<td>255.255.255.0</td>
</tr>
<tr>
<td>all.socket_port</td>
<td>This value applies to both the local ports and ports on Slave CPS.</td>
<td>7001+</td>
</tr>
<tr>
<td>all.protocol</td>
<td>Depends on the application.</td>
<td>socket_ssh,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>socket_server, or</td>
</tr>
<tr>
<td></td>
<td></td>
<td>socket_server_ssh</td>
</tr>
<tr>
<td>all.authtype</td>
<td>Depends on the application.</td>
<td>Radius, local, none,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>remote, TacacsPlus, Ldap,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>kerberos, local/Radius, radius/</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local, local/TacacsPlus,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TacacsPlus, local, RadiusDownLocal,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LdapDownLocal, NIS</td>
</tr>
<tr>
<td>s33.tty</td>
<td>This parameter must be created in the Master CPS 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.</td>
<td>20.20.20.2:7033</td>
</tr>
</tbody>
</table>
Table 2-2: Master configuration (where it differs from the CPS standard)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Examples/ Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>s33.alias</td>
<td>An alias for this port. (This is an optional parameter).</td>
<td>server_on_slave1_serial_s1</td>
</tr>
<tr>
<td>s33.ipno</td>
<td>This parameter must be created in the Master CPS file for every Slave port, unless configured using all.ipno.</td>
<td>0.0.0.0</td>
</tr>
<tr>
<td>s34.tty</td>
<td>See s33.tty.</td>
<td>20.20.20.2:7034</td>
</tr>
<tr>
<td>s34.alias</td>
<td>An alias for this port.</td>
<td>server_on_slave1_serial_s2</td>
</tr>
<tr>
<td>s34.ipno</td>
<td>See s33.ipno.</td>
<td>0.0.0.0</td>
</tr>
<tr>
<td>s35.tty</td>
<td>See s33.tty.</td>
<td>20.20.20.2:7035</td>
</tr>
<tr>
<td>s35.alias</td>
<td>An alias for this port.</td>
<td>server_on_slave1_serial_s3</td>
</tr>
<tr>
<td>s35.ipno</td>
<td>See s33.ipno.</td>
<td>0.0.0.0</td>
</tr>
<tr>
<td>etc. for s36-s64</td>
<td></td>
<td></td>
</tr>
<tr>
<td>s65.tty</td>
<td>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.</td>
<td>20.20.20.3:7301</td>
</tr>
<tr>
<td>S65.alias</td>
<td>An alias for this port.</td>
<td>server_on_slave2_serial_s1</td>
</tr>
<tr>
<td>S65.ipno</td>
<td>See s33.ipno.</td>
<td>0.0.0.0</td>
</tr>
<tr>
<td>S66.tty</td>
<td>See s65.tty.</td>
<td>20.20.20.3:7302</td>
</tr>
</tbody>
</table>
2. To configure the first Slave unit, follow the table below.

The Slave CPSs do not need to know they are being accessed through the Master CPS. (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:

**Table 2-3: Slave 1 configuration (where it differs from the CPS standard)**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value for this example</th>
</tr>
</thead>
<tbody>
<tr>
<td>all.protocol</td>
<td>socket_server</td>
</tr>
<tr>
<td>all.authtype</td>
<td>none</td>
</tr>
<tr>
<td>conf.eth_ip</td>
<td>20.20.20.2</td>
</tr>
<tr>
<td>all.socket_port</td>
<td>7033+</td>
</tr>
</tbody>
</table>
3. Configure the second Slave unit. To configure the second slave, follow the parameters of the table below:

**Table 2-4:** Slave 2 configuration (where it differs from the CPS standard)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value for this example</th>
</tr>
</thead>
<tbody>
<tr>
<td>all.protocol</td>
<td>socket_server</td>
</tr>
<tr>
<td>all.authtype</td>
<td>none</td>
</tr>
<tr>
<td>conf.eth_ip</td>
<td>20.20.20.3</td>
</tr>
<tr>
<td>all.socket_port</td>
<td>7301+</td>
</tr>
</tbody>
</table>

4. Activate and save the changes.
   - To activate the changes, issue the command:
     
     ```bash
     # runconf
     ```
   - To save the changes, issue the command:
     
     ```bash
     # saveconf
     ```

5. To access the ports from the remote management workstation, use Telnet with the secondary IP address.
   - To access the first port of the Master CPS:
     
     ```bash
     # telnet 209.81.55.110 7001
     ```
   - To access the first port of the Slave1 CPS:
     
     ```bash
     # telnet 209.81.55.110 7033
     ```
   - To access the first port of the Slave2 CPS:
     
     ```bash
     # telnet 209.81.55.110 7301
     ```

     SSH can also be used from the remote management workstation.
   - To access the third port of Slave 2:
     
     ```bash
     # ssh -l <username>:Server_on_slave2_serial_s3 209.81.55.110
     ```
   - To access the fifth port of Slave 2:
     
     ```bash
     # ssh -l <username>:7305 209.81.55.110
     ```

**TIP:** It is possible to get the clustering channel tunneled through IPSec. For more information about IPSec, see [VPN Configuration](#).
2.5 Clustering Using NAT (Enhanced)

With Enhanced Clustering, the CPS ports in the slave box can be configured as SSH or Telnet and can have any type of authentication available. Authentication is now performed in the Slave and not in the Master. 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 to which this IP address is assigned. Additionally, there are two chains (post_nat_cluster and pre_nat_cluster) that hold all rules to perform NAT for clustering.

<table>
<thead>
<tr>
<th>Abbreviation List</th>
</tr>
</thead>
<tbody>
<tr>
<td>clustering_ip</td>
</tr>
<tr>
<td>master_ip</td>
</tr>
<tr>
<td>slave_ip</td>
</tr>
<tr>
<td>master_port</td>
</tr>
<tr>
<td>slave_port</td>
</tr>
</tbody>
</table>
The Master CPS 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 CPS 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 -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 CPS administrator can issue an iptables command to view, change, or delete the rules in the nat table. If the administrator issues a “fwset restore” command he must also execute the command “runconf” to recover the nat table.

CPS clustering allows access to a large number of serial ports (in more than one box) 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:**

1. 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
```

2. 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 (CPS) 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.

Command to start a Telnet client session:

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

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 can use the following commands:

```
# ssh -l <username1>:@<server1> -p 7101 <master_ip>
# ssh -l <username2>:@<server2> -p 7101 <master_ip>
```

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

```
# ssh -l <username1>:@<server1> <slavel_ip>
# ssh -l <username2>:@<server2> <slavel_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>
```

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 implementation they must be valid in the Slave. In the Master box there is no meaning anymore for remote port's alias and authtype parameters. 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 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 alias for remote ports must be configured now in the related Slave box.

In the following configuration examples, “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 example
All changes must be made in the /etc/portslave/pslave.conf file of the Master box:

```plaintext
#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 2.2: Master box: /etc/portslave/pslave.conf
<table>
<thead>
<tr>
<th>Configuration Parameters</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Secondary ethernet IP address</strong></td>
<td></td>
</tr>
<tr>
<td>conf.eth_ip_alias</td>
<td>192.168.170.1</td>
</tr>
<tr>
<td>conf.eth_mask_alias</td>
<td>255.255.255.0</td>
</tr>
<tr>
<td><strong>Local CPS serial ports (32 socket_ssh ports)</strong></td>
<td></td>
</tr>
<tr>
<td>all.protocol</td>
<td>socket_ssh</td>
</tr>
<tr>
<td>all.authtype</td>
<td>local</td>
</tr>
<tr>
<td>all.socket_port</td>
<td>7001+</td>
</tr>
<tr>
<td>s[1-32].tty</td>
<td>ttyS[1-32]</td>
</tr>
<tr>
<td><strong>Remote CPS serial ports, slave-1 (32 socket_ssh ports)</strong></td>
<td></td>
</tr>
</tbody>
</table>

*File Description 2.2: Master box: /etc/portslave/pslave.conf*
Device Access

s33.tty 192.168.170.2
s33.socket_port 7000

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

#
# Remote CPS 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 CPS 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 2.2: Master box: /etc/ports/Cps/pslave.conf
Slave-1 box configuration example

All changes must be made in the `/etc/portslave/pslave.conf` file of the first Slave box:

```ini
#Slave-1 box Configuration
#Primary ethernet IP address
#
conf.eth_ip 192.168.170.2
conf.eth_mask 255.255.255.0
conf.eth_mtu 1500
#
# Local CPS serial ports (32 socket_ssh ports)
#
all.protocol socket_ssh
all.authtype local
s[1-32].tty ttyS[1-32]
s[1-32].alias slave-1-port[1-32]
```

File Description 2.3: Slave1 box: /etc/portslave/pslave.conf

Slave-2 box configuration example

All changes must be made in the `/etc/portslave/pslave.conf` file of the second Slave box:

```ini
#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 CPS 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 2.4: Slave2 box: /etc/portslave/pslave.conf
Slave-3 box configuration example

All changes must be made in the `/etc/portslave/pslave.conf` file of the third Slave box:

```
#Slave-3 box Configuration
# Primary ethernet IP address
#
conf.eth_ip 192.168.170.4
conf.eth_mask 255.255.255.0
conf.eth_mtu 1500

#
# Local CPS 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 2.5: Slave2 box: `/etc/portslave/pslave.conf`

Example of starting CPS session commands

The alias, 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
```
CLI method—clustering

Because the clustering process is configured using the ethernet interface, no hardware connection is required between the boxes. Connect the boxes in the same physical network. To configure one CPS (master) to control other CPSs (slaves) using the CLI, complete the following steps.

Information for the example:

- Master box—4-port unit; 172.22.65.2 as IP address
- Slave box—32-port unit; 172.22.65.3 as IP address
- Telnet protocol to access the serial ports.

1. Open the CLI interface by issuing the command:
   
   ```
   # CLI
   ```

2. Add a slave box.
   
   ```
   cli>config virtualports addslave 172.22.65.3
   ```

3. Configure the slave box.

   All parameters that must be configured are listed, commented, and given the correct value for this example.

   - `numports`: This parameter sets the total number of ports of the slave box. The value for this example is 32.
   - `firstlocalportnum`: This parameter acts as the numbering continuation in the slave box. Because the master unit is a 4-port box, the first port of the slave unit will be the first local port number. The value for this example is 5.
   - `localip`: This parameter sets the IP address of the slave box. The value for the example is 172.22.65.3.
   - `firstlocaltcpport`: Same function as the `firstlocalportnum` parameter, but sets the TCP port. The value for the example is 7005.
   - `remoteip`: The IP address of the master box. The value for this example is 172.22.65.2.
   - `firstremotetcpport`: Where TCP port numbering starts in the master box. The value for this example is 7001.
   - `protocol`: Protocol used to access the ports. Valid values are `telnet` or `SSH`.

4. Activate the configuration.
   
   ```
   cli>config runconfig
   ```
5. Save the configuration.
   
   `cli>config savetoflash`

6. Test and configure the virtual ports, and configure any additional options.
   
   Telnet to port 10 of the slave box. Supposing you are in the same network as the master CPS, run the command:
   
   `# telnet 172.22.65.2 7014`

   You can also edit or delete any previously configured virtual port (in this example, `n.n.n.n` is the IP address of the configured virtual port):
   
   `cli>config virtualports editslave <n.n.n.n>`
   `cli>config virtualports deleteslave <n.n.n.n>`

7. Exit the CLI mode and return to CPS’s shell by typing the `quit` command:
   
   `cli> quit`
Chapter 3: Authentication

This chapter presents the procedures for assigning and configuring the authentication service(s) that the CPS, system or any of its components and devices will use. 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 and confirms receipt of communication to authorized recipients. This chapter includes the following topics:

- Device Authentication
- Linux-PAM
- Shadow Passwords
- Certificate for HTTP Security
- X.509 Certificate on SSH

3.1 Device Authentication

Authentication is the process of identifying an individual, usually by 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 ensures that the individual is who he or she claims to be, but does not assign access rights of the individual. With the CPS, authentication can be performed locally, or authentication can be performed remotely with Radius, Tacacs, Ldap, or kerberos.

How to Configure Device Authentication Using VI Mode—Parameters Involved and Passed Values

To configure the authentication type of the CPS:

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

   Edit the all.authtype parameter in the pslave.conf file to specify the type of authentication server being configured. The following table provides a brief description of each authentication type.
Table 3-1: Authentication parameters in /etc/portslave/pslave.conf

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>all.authtype</td>
<td>Type of authentication used:</td>
</tr>
<tr>
<td></td>
<td>• None—no authentication</td>
</tr>
<tr>
<td></td>
<td>• Local—authentication is performed using the /etc/passwd file</td>
</tr>
<tr>
<td></td>
<td>• Remote—This option is for a terminal profile only. The unit takes in a username but does not use it for authentication. Instead it passes the username to the remote server, where it is then used for authentication.</td>
</tr>
<tr>
<td></td>
<td>• Radius—authentication is performed using a Radius authentication server</td>
</tr>
<tr>
<td></td>
<td>• TacacsPlus—authentication is performed using a TacacsPlus authentication server</td>
</tr>
<tr>
<td></td>
<td>• 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</td>
</tr>
<tr>
<td></td>
<td>• 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</td>
</tr>
<tr>
<td></td>
<td>• Local/Radius—authentication is performed locally first, switching to Radius if unsuccessful</td>
</tr>
<tr>
<td></td>
<td>• Radius/Local—the opposite of the preceding option</td>
</tr>
<tr>
<td></td>
<td>• Local/TacacsPlus—authentication is performed locally first, switching to TacacsPlus if unsuccessful</td>
</tr>
<tr>
<td></td>
<td>• TacacsPlus/Local—the opposite of the preceding option</td>
</tr>
<tr>
<td></td>
<td>• RadiusDownLocal—local authentication is tried only when the Radius server is down</td>
</tr>
<tr>
<td></td>
<td>• TacacsPlusDownLocal—local authentication is tried only when the TacacsPlus server is down</td>
</tr>
</tbody>
</table>

Note: This option is invalid when the serial port is configured for Power Management. The system default is “Local” if no authentication type is selected.
Note: If you want to dial in to the serial port on an CPS series with CHAP authentication, you need to do the following:
1. Configure $xx.authtype$ as local.
2. Add users in APC Console Port 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.

Note: This parameter controls the authentication required by the APC Console Port Server. The authentication required by the device to which the user is connecting is controlled separately.

Table 3-1: Authentication parameters in /etc/portslave/pslave.conf

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| all.authtype    | • Kerberos/Local—Kerberos authentication is tried first, switching to Local if unsuccessful  
                 • KerberosDownLocal—local authentication is tried only when the kerberos server is down  
                 • Ldap/Local—LDAP authentication is tried first, switching to local if unsuccessful  
                 • LdapDownLocal—local authentication is tried only when the ldap server is down  
                 • NIS—All authentication types except 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 is configured in the /etc/nsswitch.conf file, not in the /etc/portslave/pslave.conf file.  
                 Note: This parameter controls the authentication required by the APC Console Port Server. The authentication required by the device to which the user is connecting is controlled separately.  

2. Configure an authentication server.

   The parameters for each type of authentication server are stored in that server’s configuration file on CPS.

   **Table 3-2: Authentication Servers and File Path**

<table>
<thead>
<tr>
<th>Authentication Server</th>
<th>File Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>Radius</td>
<td>/etc/raddb/server</td>
</tr>
<tr>
<td>TacacsPlus</td>
<td>/etc/tacplus.conf</td>
</tr>
<tr>
<td>Kerberos</td>
<td>/etc/krb5.conf</td>
</tr>
<tr>
<td>LDAP</td>
<td>/etc/ldap.conf</td>
</tr>
<tr>
<td>NIS</td>
<td>/etc/yp.conf</td>
</tr>
</tbody>
</table>

3. Activate and save the changes.

   To activate the changes, issue the command:
   ```
   # runconf
   ```

   To save the changes, issue the command:
   ```
   # saveconf
   ```

### How to Configure Device Authentication Using CLI Method—Authentication

Follow the procedures in this section of the manual to:

- Configure the authentication type for the CPS serial ports.
- Configure user access to the serial ports.
- Configure the authentication type for accessing the console of a connected device.
- Configure an authentication server.

**To configure the authentication type for the serial ports**

1. Open the CLI interface by issuing the command:
   ```
   # CLI
   ```
2. Navigate to the following path.

    cli>config physicalports access authtype [value]

**Note**: For **physicalports**, specify a port number, select a range, or enter “all”. For example: “physicalport 4”, “physicalports 1-8”, or “physicalports all”.

3. To see the list of authentication types, from >authtypes

    <Press tab to see list of possible values>

**Note**: Authentication type “None” is not a valid option when the serial port is configured for Power Management connection protocol. The system default setting is “Local” if no authentication type is selected.

### To configure user access to the serial ports

1. Open the CLI interface by issuing the command:

    # CLI

2. Navigate to the following path.

    cli>config security

This menu lets you execute the following actions:

- **adduser**—To add a user, you must pass the following parameters: user name `<username>`; administrator privileges or not `<admin - yes|no>`; biouser or not `<biouser - yes|no>`; password `<password>`; comments `<comments>`. For example:

    security>adduser username john admin no biouser no password john1234

- **addgroup**—To add a group it is necessary to inform the group name `<groupname>` and the members `<usernames>` of this group. For example:

    security>addgroup groupname test usernames john,mary

- **delgroup**—To delete a group it is necessary to inform the name of the group `<groupname>` you want to delete. For example:

    security>delgroup groupname test

- **deluser**—To delete an existing user, it is necessary to inform the user you want to delete by specifying the `<username>` parameter. For example:

    security>deluser username APC
• **loadkey**—This option allows you to get the user’s public key via scp. The user must be enrolled in the local database of the unit. You must specify the user name `<username>` and the url `<url>`. The url must follow this syntax: `user@host:pathname`. Example:

```
security>loadkey username root url root@192.168.0.1/home/key
```

• **passwd**—Change the password of a user. You need to inform the user `<username>` and the new password `<newpassword>` of the user. Example:

```
security>passwd username root newpassword difficultpasswd1234
```

**To configure authentication type for device console access**

1. Open the CLI interface by issuing the command:

```
# CLI
```

2. Navigate to the following path.

```
cli>config security authentication authtype <string>
```

3. To see the list of authentication types, from >authtype

```
<Press tab to see list of possible values>
```

The following list of authentication types appears.

- Nis, kerberos, local/Nis, Nis/local, kerberos/local, local/TacacsPlus, NisDownLocal, kerberosDownLocal, local/radius, RadiusDownLocal, ldap, none, TacacsPlus, ldap/local, radius, TacacsPlus/local, ldapDownLocal, radius/local, TacacsPlusDownlocal, and local.

For more information, see the all.authtype parameter on Table 3-1, “Authentication parameters in /etc/portslave/pslave.conf,” on page 40

**To configure an authentication server**

1. Open the CLI interface by issuing the command:

```
# CLI
```

2. Navigate to the following path.

```
cli>config security authentication
```

3. To see the list of authentication server types, from >authentication, press the tab key to see the list of possible values. The following list of authentication types appears.
nisserver,radiussecret,tacplusauthsvr1,radiustimeout
tacplusauthsvr2, krbdomain, radiusacctsvr1, tacplusaccess,
krbserver,radiusacctsvr2, secureldap, tacplusretries,
ldapbasedomain, radiusauthsvr1, tacplussecret, ldapserver,
radiusauthsvr2,tacplusacctsvr1, tacplustimeout, nisdomain,
radiusretries,tacplusacctsvr2

4. Configure an authentication server.

#config security authentication <server option> <ip address>

- To activate the configuration:
  cli> config runconfig

- To save the configuration:
  cli> config savetoflash

- To exit the CLI mode:
  cli> quit

Access Control Using the Radius Attribute **NAS-Port-id**

This feature provides an additional way to control 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:

```plaintext
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 CPS. 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. This feature allows the administrator to manage CPS accounts on an NIS server.

The NIS client feature needs the following files/commands:

**Table 3-3:** NIS client requirements

<table>
<thead>
<tr>
<th>File/Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/etc/yp.conf</td>
<td>This file contains the configuration used by ypbind.</td>
</tr>
<tr>
<td>/etc/domainname.conf</td>
<td>This file contains the NIS domain name (set by the command domainname).</td>
</tr>
<tr>
<td>/usr/sbin/ypbind</td>
<td>Finds the server for NIS domains and maintains the NIS binding information.</td>
</tr>
<tr>
<td>/usr/bin/ypwhich</td>
<td>Returns the name of the NIS server that supplies the NIS services.</td>
</tr>
<tr>
<td>/usr/bin/ypcat</td>
<td>Prints the values of all keys from the NIS database specified by map name.</td>
</tr>
<tr>
<td>/usr/bin/ypmatch</td>
<td>Prints the values of one or more keys from the NIS database specified by map name.</td>
</tr>
<tr>
<td>/usr/sbin/domainname</td>
<td>Shell script to read/write the NIS domain name.</td>
</tr>
</tbody>
</table>

**NIS client configuration**

1. Run the command `domainname`.
   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, e.g.:
# domainname APC mycompany-nis

2. Edit the `/etc/yp.conf` file.

   You will need to configure the NIS server. For 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`

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.

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

How to test the configuration

1. Start up the following command:

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

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

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

3. Display the “all users” entry in the NIS database by running the following command:

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

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, 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 correctly, but the required 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, change the lines in /etc/nsswitch.conf that reference passwd, shadow, and group.

**Examples**

1. 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

2. 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

3. 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
3.2 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 detection of modification and the prevention of unauthorized reading.

Kerberos Server Authentication with Tickets Support

The CPS supports interaction on a kerberized network. This chapter provides a brief explanation about how kerberos works. See “Configuring CPS to Use Kerberos Tickets Authentication” on page 50 for a step-by-step example.

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 CPS 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 only describes the CPS configuration.

We will assume that the kerberos server has the following configuration:

- Principal: john
- Host (APC CPS): cps48-2.APC.com

CPS configuration

Configuring the CPS for SSH:

1. Configure and start a NTP server
   
   All involved parts must be synchronized with an NTP server. To configure an NTP server, see “NTP (Network Time Protocol)” on page 173.

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.

3. Activate and save the configuration by issuing the commands:

   # runconf
   # saveconf

4. Add a user with the same name as the “principal” configured in the Kerberos server.

   # adduser john

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 that you copy the file directly from the server, instead of editing it. To copy using scp, issue the command:

   # scp root@kerberos-server.APC.com:/etc/krb5.conf /etc/krb5.conf
6. Extract the host that is in the Kerberos server database to the CPS:

   # 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/cps48-2.APC.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.

7. Configure hostname and domain name.

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

   # hostname cps48-2
   # domainname APC.com

Rlogin and Telnet:

To access the CPS through rlogin or Telnet, complete the steps described in the preceding section, then complete the following steps.

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

2. Restart the inetd service by issuing the command:

   # daemon.sh restart NET

3. Save the configuration.

   # saveconf
Test the configuration

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

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

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

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

3. Request the ticket to the Kerberos server.

   ```bash
   # kinit -f -p john
   Password for john@APC.COM: *******
   ```
   You will be prompted to insert a password (the “principal” password that is in the Kerberos server database).

4. Check whether the ticket was successfully received:

   ```bash
   # klist
   ```

5. Using SSH, connect from the client to the CPS:

   - Opening a SSH connection to the CPS itself:
     ```bash
     # ssh john@cps48-2.APC.com
     ```
   - Opening a SSH session to one of the CPS ports:
     ```bash
     # ssh john:7001@cps48-2.APC.com
     ```

6. Using RLOGIN, connect to the CPS with forwardable tickets (to connect to the CPS ports using `ts_menu`):

   ```bash
   # rlogin -l john cps48-2.APC.com -F
   ```
   Then run `ts_menu` to access the desired serial port.

7. Using Telnet, connect to the CPS with forwardable tickets (to connect to the CPS ports using `ts_menu`):

   ```bash
   telnet -l john cps48-2.APC.com -F
   ```
   Then run `ts_menu` to access the desired serial port.
Kerberos Server Authentication

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

1. Edit the `/etc/portslave/pslave.conf` file.

   Open the `/etc/portslave/pslave.conf` file by running the following command:
   ```bash
   # vi /etc/portslave/pslave.conf
   ```
   
   Look for the `all.authtype` and `all.protocol` parameters and change their values according to the following example:
   ```
   all.authtype     kerberos
   all.protocol     socket_ssh ##or socket_server or socket_server_ssh
   ```
   
   • To use the Telnet protocol to access the serial ports of the unit, set the `all.protocol` parameter to `socket_server`.
   
   • To use both Telnet and SSH to access the unit, set the `all.protocol` parameter to `socket_server_ssh`. In this example we are using the SSH protocol.

2. Edit the `/etc/krb5.conf` file.

   Open the `/etc/krb5.conf` file by running the following command:
   ```bash
   # vi /etc/krb5.conf
   ```
   
   All of the changes required in this file are related to the network domain. Substitute all listed parameters that are configured with “APC.com” with the corresponding 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 = APC.COM
   default_tgs_enctypes = des-cbc-crc
   default_tkt_enctypes = des-cbc-crc
   ```

   *File Description 3.1: /etc/krb5.conf*
3. Activate the changes made by running the command:

```bash
# runconf
```

4. Test the configuration.

To test the configuration, access any serial port using the Telnet protocol. From a remote machine issue the following command:

```bash
# telnet 192.168.0.1 7001
```

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

In the CPS, run the command:

```bash
# w
```
The response for this command will be similar to the following example:

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

<table>
<thead>
<tr>
<th>USER</th>
<th>TTY</th>
<th>FROM</th>
<th>LOGIN@</th>
<th>IDLE</th>
<th>JCPU</th>
<th>PCPU</th>
<th>WHAT</th>
</tr>
</thead>
<tbody>
<tr>
<td>root</td>
<td>ttyS0</td>
<td>-</td>
<td>12:07pm</td>
<td>0.00s</td>
<td>1.47s</td>
<td>0.15s</td>
<td>/bin/sh /usr/bin</td>
</tr>
</tbody>
</table>

CAS users : 1

<table>
<thead>
<tr>
<th>USER</th>
<th>TTY</th>
<th>FROM</th>
<th>LOGIN@</th>
<th>PID/Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>APC</td>
<td>ttyS1</td>
<td>192.168.0.143:1503</td>
<td>01:02pm</td>
<td>512/-RW_srv ttyS</td>
</tr>
</tbody>
</table>

The last line of the command response shows the user “APC” accessing the first serial port of the CPS unit.

5. Save the configuration by running the command:
   
   # saveconf

### 3.3 LDAP Authentication

This section describes the basic steps for configuring an LDAP server on a Linux machine, and explains how to configure the CPS.

#### Configuring an LDAP Server on Linux

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

1. The packages required for the LDAP servers are:
   
   - db (Sleepycat Berkeley Database)
   - openssl (OpenSSL)
   - openldap (OpenLDAP)

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

2. Go to the directory `/etc/openldap` or `/usr/local/etc/openldap`.

   To change the directory, run the following command:

   # cd /usr/local/etc/openldap
3. Create the certificates by running the following commands in the given sequence:

```bash
# ln -s /usr/local/bin/openssl.
# ln -s /usr/local/ssl/misc/CA.pl.
# PATH=$PATH:.
# CA.pl -newca <-- answer 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
```

4. Edit `slapd.conf`. The basic configuration of the file is described in the following example:

```bash
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/demode/cacert.pem

database bdb
suffix "dc=APC,dc=com"

rootdn "cn=admin,dc=APC,dc=com"

rootpw bitadmin

directory /usr/local/var/openldap-data

index objectClass eq
```

*File Description 3.2: slapd.conf configuration*
5. Start the LDAP server by running the command:

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

   This will allow the LDAP server to accept both secured and non-secured modes.

6. Add entries. For example:

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

   To list the entries:

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

---

### Configuring the APC Console Port Server Side

To configure the unit for PAM authentication, follow the steps below.

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

2. Configure the /etc/ldap.conf file.

   Edit the following parameters:

   ```
   host 192.168.1.95 <= LDAP server IP address or name
   base dc=APC,dc=com <= distinguished name of the search base
   uri ldaps://192.168.1.95 <= to use secure LDAP
   ```

3. Activate and save the changes.

   To activate the changes, issue the command:
   
   # runconf

   To save the changes, issue the command:
   
   # saveconf
Active Directory

A Windows 2000 or Windows 2003 Server edition is necessary. In the CPS side, the /etc/ldap.conf file must be configured.

Parameters that must be set in the /etc/ldap.conf file

Follow the example below to set the parameters correctly:

```
# The Windows 2003 server IP address
host 192.168.1.95

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

# Here you can insert any user you had created, or the administrator
# user.
binddn cn=Administrator,cn=Users,dc=APC,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 3.4: /etc/ldap.conf

Enabling TACACS+ Authorization for Serial Ports

Using an authorization method in addition to authentication provides an extra level of system security. By enabling the raccess parameter, administrators require an additional level of security checking. After each user is successfully authenticated through the standard login procedure, the CPS uses TACACS+ to authorize whether or not each user is allowed to access specific serial ports.

By default, the raccess parameter is disabled, and allows all users full authorization. When the raccess parameter is enabled, users are denied access unless they have the proper authorization, which must be set on the TACACS+ server.
Configuring Authorization with a TACACS+ Server [CLI]

1. In CLI mode, enter the following string:
   
   ```
   config > security > authentication> tacplusraccess yes
   ```

2. To save the configuration, enter the command:

   ```
   config > savetoflash
   ```

Configuring Authorization with a TACACS+ Server [vi]

1. Open `/etc/tacplus.conf`

2. Edit the service parameter to be raccess:

   ```
   service=raccess
   ```

Setting user authorization permissions on the TACACS+ Server [vi]

The authorization for each user is defined on the TACACS+ server itself in the file
`/etc/tacacs/tac_plus.cfg` on the "Linux Fedora Core 3"

---

**Note:** The location of this configuration file may be different on other Linux distributions.

---

1. On the TACACS+ server, open the file `/etc/tacacs/tac_plus.cfg`.

2. Edit the following lines (the text in bold type face is included as an example):

   ```
   user = tomj{
       name = "Tom Jones"
       service = raccess {
           port1 = LAB2/ttyS2
           port2 = 192.168.0.1/ttyS1
           port3 = CAS/ttyS1
           port4 = 172.32.20.10/ttyS6
           port5 = LAB1/ttyS7
           port6 = Knuth/ttyS16
       }
   }
   ```
Table 3-4: Parameters for Specifying User Authorization on a TACACS+ Server

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Example Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>user = &lt;username&gt;</td>
<td>Define the username as specified on the CPS.</td>
<td>tomj</td>
</tr>
<tr>
<td>name = &quot;optional description&quot;</td>
<td>(Optional) Specify additional information about the user. This parameter must include quotes. The maximum number of characters allowed is 256. Adding more than 256 characters stops the server from restarting and produces a &quot;FAILED&quot; message at the time of authorization.</td>
<td>&quot;Tom Jones&quot;</td>
</tr>
<tr>
<td>service = &lt;authorization method&gt;</td>
<td>Only users who have this parameter set to raccess will have authorization to access the specified ports.</td>
<td>raccess</td>
</tr>
<tr>
<td>port&lt;#&gt; = &lt;CPS&gt;/&lt;Port&gt;</td>
<td>Specify which serial ports on the CPS the user has authorization to access.</td>
<td>port1 = LAB2/ttyS2</td>
</tr>
</tbody>
</table>

port# is a sequential label used by the CPS.

<CPS> is the name or IP address of the CPS box.

<Port> is the serial port the user can access on the specified CPS box.

3.4 Group Authorization

This feature enables the “group” information retrieval from the authentication servers TACACS+, RADIUS, and LDAP, and adds another layer of security by adding network-based authorization. It retrieves the “group” information from the authentication server and performs an authorization through CPS.
The following sections describe the procedures for configuring TACACS+, RADIUS, and LDAP authentication servers, and the corresponding configuration process on CPS.

## Configuring a TACACS+ Authentication Server

On the server, add “raccess” service to the user configuration and define the group or groups to which the user belongs.

```plaintext
user = <username> {
  service = raccess {
    group_name = <Group1>[,<Group2,...,GroupN>]
  }
}
```

On the CPS, edit the following parameters in the `/etc/tacplus.conf` file.

```plaintext
authhost1 = 192.168.160.21
accthost1 = 192.168.160.21
secret = secret
encrypt = 1
service = ppp
protocol = lcp
timeout = 10
retries = 2
```

- `authhost1`: This address indicates the location of the TacacsPlus authentication server. A second TacacsPlus authentication server can be configured with the parameter `authhost2`.

- `accthost1`: This address indicates the location of the 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 TacacsPlus accounting server can be configured with the parameter `accthost2`.

- `secret`: This is the shared secret (password) necessary for communication between the CPS and the TacacsPlus servers.

- `encrypt`: The default is 1, which means encryption is enabled. To disable encryption, change the value to 0.
**service:** The service that should be enabled. The default is ppp. If you are enabling another service (for example, “raccess” authorization on the TacacsPlus server), it should be mentioned in this field on CPS.

**protocol:** The default is line control protocol (lcp). Specify another parameter if required.

**timeout:** This is the timeout (in seconds) for a TacacsPlus authentication query to be answered.

**retries:** Defines the number of times each TacacsPlus server is tried before another is contacted. The first server authhost1 is tried for the specified number of times. If authentication at the first server fails and a second server authhost2 is configured, the second server is contacted and tried for the specified number of times. If the second server fails to respond, TacacsPlus authentication fails.

**Configuring the authorization on CPS to access the serial ports [CLI]**

In CLI mode, enter the following string:

1. cli > config security authentication tacplusraccess yes
2. cli > config physicalports <serial port number> access users/groups <list of users or group names separated by commas>
3. Save the configuration to flash
   
   cli > config > savetoflash

**Configuring a RADIUS Authentication Server**

On the server, edit /etc/raddb/users and add a new string attribute (ATTRIBUTE Framed-Filter-Id) similar to the following example:

```plaintext
    groupuser1  Auth-Type= Local, Password =”xxxx”
    Service-Type=Callback-Framed-User,
    Callback-Number=”305”,
    Framed-Protocol=PPP,
    Framed-Filter-Id=”:group_name=<Group1>[,<Group2>,...,<GroupN>]”,
    Fall-Through=No
```

If the Frame-Filter-Id already exists, just add the group_name to the string starting with a colon “:”
On the CPS, edit /etc/raddb/server
format: server[:port] secret [retries] [timeout]
For example:
auth1 172.20.0.2 APC 3 5
acct1 172.20.0.2 APC 3 5
Where:
server: The Radius server address.
port: The port field is optional. The default port name is “radius” and is looked up through /etc/services.
secret: The shared password required for communication between CPS and the Radius server.
retries: The number of times each Radius server is tried before another is contacted
timeout: The default is 3 seconds. The timeout field determines how long the system should wait before responding with a success or failure response from the authentication server.

Note: Multiple Radius servers can be configured in this file. The servers are tried in the order in which they appear. If a server fails to respond, the next configured server is tried.

Configuring the authorization on CPS to access the serial ports [CLI]
1. In CLI mode, enter the following string:
   cli > config physicalports <serial port number> access users/groups <list of users or group names separated by commas>
2. Save the configuration to flash
   cli > config > savetoflash

Configuring an LDAP Authentication Server
On the server, edit the “info” attribute for the user and add the following syntax.
   info: group_name=<Group1>[,<Group2>,...,<GroupN>];
Configuring the authorization on CPS to access the serial ports [CLI]

1. In CLI mode, enter the following string:
   
   `cli > config physicalports <serial port number> access users/groups <list of users or group names separated by commas>`

2. Save the configuration to flash
   
   `cli > config > savetoflash`

3.5 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 mechanisms it uses. The administrator can entirely upgrade the local authentication system without touching the applications themselves.

The purpose of the Linux-PAM project is 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. The PAM library has a series of configuration files located in `/etc/pam.d/` to authenticate a user request using the locally available authentication modules. Usually, the modules are located in the directory `/lib/security` and are dynamically loadable object files.

The Linux-PAM authentication mechanism gives the system administrator the freedom to stipulate which authentication scheme is to be used. The administrator can set the scheme for any/all PAM-aware applications on your Linux system. That is, the administrator 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 tasks: 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.
The following figure describes the overall organization of Linux-PAM:

![Data flow diagram of Linux-PAM](image)

**Figure 3.1 - 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 Directory**

Linux-PAM provides the system administrator significant 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 the directory `/etc/pam.d/`. In this section we discuss the correct syntax of entries and generic options respected by entries for the files in this directory.
Configuration file syntax

The reader should note that the Linux-PAM-specific tokens are case-insensitive. However, the module paths 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, two special characters are 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 a file in the /etc/pam.d/ directory has the following form:

    filename module-type control-flag module-path arguments

The meaning of each of these tokens is explained in the following table. After the meaning of the tokens is explained, the method will be described.

Table 3-5: /etc/pam.d/ tokens description

<table>
<thead>
<tr>
<th>Token</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filename</td>
<td>The service name associated with the entry. 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 lowercase or uppercase characters.</td>
</tr>
<tr>
<td>Note:</td>
<td>When there is a module specified for a named service, the ‘OTHER’ entries are ignored.</td>
</tr>
<tr>
<td>Module-type</td>
<td>One of the four types of modules. The four types are as follows:</td>
</tr>
<tr>
<td></td>
<td>• 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.</td>
</tr>
</tbody>
</table>
Authentication

Table 3-5: /etc/pam.d/ tokens description

<table>
<thead>
<tr>
<th>Token</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Account</td>
<td>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 the location of the applicant user—‘root’ login only on the console.</td>
</tr>
<tr>
<td>Session</td>
<td>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.</td>
</tr>
<tr>
<td>Password</td>
<td>This 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.</td>
</tr>
</tbody>
</table>

Control-flag

The control-flag indicates 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.d/ directory. Instead, it receives a summary of success or failure responses from the Linux-PAM library. The order of execution of these modules is that of the entries in the /etc/pam.d/ directory: 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 keywords: required, requisite, sufficient and optional.
The Linux-PAM library interprets the control-flag keywords in the following manner:

**Table 3-6: /etc/pam.d/ keywords description**

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required</td>
<td>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.</td>
</tr>
<tr>
<td>Requisite</td>
<td>Similar to required. However, if the module returns a failure, control is directly returned to the application. The return value is the value associated with the first required or requisite module to fail. This flag can be used to protect against the possibility of a user entering a password over an unsafe medium.</td>
</tr>
<tr>
<td>Sufficient</td>
<td>Indicates that the success of this module is ‘sufficient’ to satisfy the Linux-PAM library that this module-type 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.</td>
</tr>
<tr>
<td>Optional</td>
<td>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, this module will determine the nature of the response to the application in the absence of any definite successes or failures of previous or subsequent stacked modules (for example, the other modules return PAM_IGNORE).</td>
</tr>
</tbody>
</table>

**Module Path**

Module Path is the path-name of the dynamically loadable object file—the pluggable module. 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 CPS has the following modules available:

**Table 3-7: Available PAM modules in the CPS**

<table>
<thead>
<tr>
<th>Module Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pam_access</td>
<td>Provides logdaemon style login access control.</td>
</tr>
<tr>
<td>pam_deny</td>
<td>Deny access to all users.</td>
</tr>
<tr>
<td>pam_env</td>
<td>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.</td>
</tr>
<tr>
<td>pam_filter</td>
<td>Offers 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 does not work well with termcap-based editors.)</td>
</tr>
<tr>
<td>pam_group</td>
<td>Provides group settings based on the user’s name and the terminal from which they are requesting a given service, and notes the time of day.</td>
</tr>
<tr>
<td>pam_issue</td>
<td>Presents the issue file (/etc/issue by default) when prompting for a username.</td>
</tr>
<tr>
<td>pam_lastlog</td>
<td>Maintains the /var/log/lastlog file. It adds an open entry when called by 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.</td>
</tr>
<tr>
<td>pam_limits</td>
<td>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.</td>
</tr>
<tr>
<td>pam_listfile</td>
<td>Provides a way to deny or allow services based on an arbitrary file.</td>
</tr>
<tr>
<td>pam_motd</td>
<td>Outputs the motd file (/etc/motd by default) upon successful login.</td>
</tr>
<tr>
<td>pam_nologin</td>
<td>Provides standard Unix nologin authentication.</td>
</tr>
</tbody>
</table>
### Table 3-7: Available PAM modules in the CPS

<table>
<thead>
<tr>
<th>Module Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pam_permit</td>
<td>This module should be used with extreme caution; its only action is to always permit access.</td>
</tr>
<tr>
<td>pam_radius</td>
<td>Provides Radius server authentication and accounting.</td>
</tr>
<tr>
<td>pam_rootok</td>
<td>This module is for use in situations where the superuser wishes to gain access to a service without having to enter a password.</td>
</tr>
<tr>
<td>pam_securetty</td>
<td>Provides standard UNIX securetty checking.</td>
</tr>
<tr>
<td>pam_time</td>
<td>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 the terminal from which they are making their request.</td>
</tr>
<tr>
<td>pam_tacplus</td>
<td>Provides TacacsPlus Server authentication, authorization (account management), and accounting (session management).</td>
</tr>
<tr>
<td>pam_unix</td>
<td>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).</td>
</tr>
<tr>
<td>pam_warn</td>
<td>This module is principally for logging information about a proposed authentication or application to update a password.</td>
</tr>
</tbody>
</table>
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 optional arguments are likely to be understood by any module. Arguments (including these) are optional.

<table>
<thead>
<tr>
<th>Module Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pam_krb5</td>
<td>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.</td>
</tr>
<tr>
<td>pam_ldap</td>
<td>Pam_ldap looks for the ldap client configuration file “ldap.conf” in /etc/. A partial example of the ldap.conf file:</td>
</tr>
</tbody>
</table>

```bash
# 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.
# LDAP.
host 127.0.0.1
# The distinguished name of the search base.
base dc=padl,dc=com
```
### Table 3-8: List of valid arguments to PAM

<table>
<thead>
<tr>
<th>Arguments</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>debug</td>
<td>Use the syslog(3) call to log debugging information to the system log files.</td>
</tr>
<tr>
<td>no_warn</td>
<td>Instruct the module not to give warning messages to the application.</td>
</tr>
<tr>
<td>use_first_pass</td>
<td>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 does not work, then the user will not be authenticated. (This option is intended for auth and password modules only).</td>
</tr>
<tr>
<td>try_first_pass</td>
<td>The module should attempt authentication with the previously typed password (from the preceding auth module). If that does not work, then the user is prompted for a password. (This option is intended for auth modules only).</td>
</tr>
<tr>
<td>use_mapped_pass</td>
<td>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.</td>
</tr>
<tr>
<td>expose_account</td>
<td>In general, exposing some information about user accounts is not a secure policy for modules to adopt, because 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 information is not deemed a threat; for example, displaying a user’s full name when asking them for a password in a secured environment could be considered ‘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 a configuration file that is not formatted correctly will cause the authentication process to fail. A corresponding error is written to the system log files with a call to syslog(3).</td>
</tr>
</tbody>
</table>
3.6 Shadow Passwords

The default /etc/passwd file has the user “root” with password “apc”. Change the password for user “root” as soon as possible.

The APC Console Port Server has support for Shadow Passwords, which enhances the security of the system authentication files.

**Note:** For CPS release 2.6, Shadow Passwords are enabled by default. If you are upgrading from release 2.3.0-2 (or earlier), a previous configuration is detected and the translation from /etc/passwd to /etc/shadow happens automatically.

3.7 Certificate for HTTP Security

The following procedure enables you to obtain a Signed Digital Certificate. A certificate for the HTTP security is created by a CA (Certificate Authority). Certificates are most commonly obtained through generating public and private keys using a public key algorithm like RSA or X.509. The keys can be generated by using a key generator software.

**Procedure**

1. Enter OpenSSL command.

   On a Linux computer, key generation can be done using the OpenSSL package, through the following command:
   
   ```
   # openssl req -new -nodes -keyout private.key -out public.csr
   ```

   If this command is used, the following information is required:

   **Table 3-9:** Required information for the OpenSSL package

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Country Name (2 letter code) [AU]:</td>
<td>The country code consisting of two letters.</td>
</tr>
<tr>
<td>State or Province Name (full name)</td>
<td>Provide the full name (not the code) of the state.</td>
</tr>
<tr>
<td>[Some-State]:</td>
<td></td>
</tr>
<tr>
<td>Locality Name (e.g., city) []:</td>
<td>Enter the name of your city.</td>
</tr>
</tbody>
</table>
The other requested information can be skipped. The certificate signing request (CSR) generated by the command above contains some personal (or corporate) information and its public key.

2. Submit the CSR and some personal data to the CA. This service can be requested by accessing the CA Web site, and is not free. See the Web site pki-page.org for a list of Certificate Authorities. The request will be analyzed by the CA for policy approval, and will be signed.

   After the approval, the CA will send a certificate file to the origin, which is called Cert.cer in the following example. The certificate is also stored on a directory server.
   The certificate must be installed in the GoAhead Web server:
   a. Open a Terminal Server session and complete the login.
   b. Join the certificate with the private key into the file /web/server.pem.
      #cat Cert.cer private.key > /web/server.pem
   c. Copy the certificate to the file /web/cert.pem.
      #cp Cert.cer /web/cert.pem
   d. Include the files /web/server.pem and /web/cert.pem in /etc/config_files.
   e. Save the configuration in flash.
      #saveconf
   The certification will be effective in the next reboot.

### Table 3-9: Required information for the OpenSSL package

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Organization Name (e.g., company) [Internet Widgits Ltd]:</td>
<td>Organization that you work for or want to obtain the certificate for.</td>
</tr>
<tr>
<td>Organizational Unit Name (e.g., section) []:</td>
<td>Department or section where you work.</td>
</tr>
<tr>
<td>Common Name (e.g., your name or your server’s hostname) []:</td>
<td>Name of the machine where the certificate must be installed.</td>
</tr>
<tr>
<td>Email Address []:</td>
<td>Your email address or the administrator’s email address.</td>
</tr>
</tbody>
</table>
3.8 X.509 Certificate on SSH

The OpenSSH software included with CPS has support for X.509 certificates. To use X.509, the administrator must activate and configure SSH. To implement authentication of SSH sessions through exchange of X.509 certificates, complete the following procedures:

1. The certificates signed and issued by the CA (Certification Authority) must be loaded on both the user and the client. See section 3.7 for information about obtaining and installing signed digital certificates.

2. Extract the Client Identification by OpenSSL command from the client’s certificate, and add to the AuthorizedKeyFile in sshd_config file.
   a. Use the following command to extract the client identification:

   ```bash
   #openssl x509 -noout -subject -in cli_cert.crt
   ```
   b. Change “subject=” to “x509v3-sign-rsa distinguishedNameE:” in the result and append to AuthorizedKeysFile in “sshd_config” file in CPS.

3. Upload hostkey to CPS in /etc/ssh directory.
To Configure X.509 Certificate for SSH

**VI mode**

1. Edit `/etc/ssh/sshd_config` file
2. Uncomment or modify the following lines to read as follows.

   ```
   AllowedCertPurpose  sslclient
   CACertificateFile /etc/ssh/ca/ca-bundle.crt
   HostKey /etc/ssh/hostkey
   ChallengeResponseAuthentication no
   HostbasedAuthentication no
   StrictModes no
   PasswordAuthentication no
   PubkeyAuthentication yes
   RhostsRSAAuthentication no
   RSAAuthentication no
   UsePrivilegeSeparation yes
   ```
3. Restart SSH.

**CLI mode**

1. Enter the CLI mode:

   ```
   [root@CPS etc]# CLI
   ```
2. Enter the following string at the cli> prompt

   ```
   cli>config network profile custom ssh_x509
   At the ssh_x509> prompt, enter the following strings.
   ssh_x509>CA_file <path and filename of CA certificate>
   ssh_x509>hostkey <path and filename of hostkeys>
   ssh_x509>authorizedkeys <path and filename of authorized keys>
   ```

   For example:
   ```
   ssh_x509>CA_file /etc/ssh/ca-bundle.crt
   ssh_x509>hostkey /etc/ssh/hostkey
   ssh_x509>authorizedkeys /etc/ssh/authorized_keys
   ```

   To check the configuration, enter the following command at the prompt.
   ```
   ssh_x509>show
   ```
The following information should appear.
[ssh_x509]
CA_file: /etc/ssh/ca-bundle.crt
hostkey: /etc/ssh/hostkey
authorizedkeys: /etc/ssh/authorized_keys

Script mode
1. Run the following “ssh_act_x509” script:
   [root@CPS root]# ssh_act_x509
   The following message appears:
   For X509 authentication, first make sure that you uploaded the CA certificate, the HostKey, and the proper Authorized Key.
2. Enter the required information at each prompt.
   AuthorizedKeysFile[/etc/ssh/authorized_keys]:
   CACertificateFile[/etc/ssh/ca/ca-bundle.crt]:
   HostKey[/etc/ssh/ssh_host_key]:
   Do you want disable Password Authentication and accept only Certificates?(y/n)
3. Check the configuration in the /etc/ssh/sshd_config file.

To Connect to CPS Using SSH X.509 Certificate
1. Edit the /etc/ssh/sshd_config file. For more information, see To Configure X.509 Certificate for SSH.
2. Configure the client you need to access with X.509 certificate.
3. Copy the certificate files to the CPS. For more information, see Certificate for HTTP Security.
   To check whether the file was copied, run the following command at the prompt:
   [root@cps48 root]# ls -l /etc/ssh/ca/ca-bundle.crt
   [root@cps48 root]# ls -l /etc/ssh/hostkey
To Connect to the Serial Ports of the CPS using SSH X.509 Certificate

1. Edit the /etc/ssh/sshd_config file. For more information, see To Configure X.509 Certificate for SSH.

2. Configure the client you need to access with X.509 certificate.

3. Copy the certificate files to the CPS. For more information, see Certificate for HTTP Security.

   To check whether the file was copied, run the following command at the prompt:
   
   [root@cps48 root]# ls -l /etc/ssh/ca/ca-bundle.crt
   [root@cps48 root]# ls -l /etc/ssh/hostkey

4. Configure the serial ports for “socket_ssh” protocol and assign the IP address of the connected device.
Chapter 4: Network

4.1 Introduction
This chapter describes important network configuration settings. The contents of this chapter are presented below:

- Basic Network Settings
- DHCP Client
- Routes and Default Gateway
- DNS Server and Domain Name
- Bonding
- Hosts
- TCP Keepalive
- Filters and Network Address Translation
- VPN Configuration

4.2 Basic Network Settings
This section demonstrates how to configure basic network parameters, including configuration of IP addresses, netmasks, and hostnames.

Hostname
The most basic network configuration is setting up a hostname. In the CPS, edit the /etc/hostname file.

VI mode
To change the hostname, edit the /etc/hostname file and set the desired hostname.

[root@CPS etc]# vi /etc/hostname
CPS

File Description 4.1: /etc/hostname

CLI mode—hostname
1. Open the CLI interface by issuing the command:

   # CLI
2. Set the hostname, where <string> is the desired hostname.
   
   cli> config network hostsettings hostname <string>

3. Activate the configuration.
   
   cli> config runconfig

4. Save the configuration.
   
   cli> config savetoflash

5. Exit the CLI mode.
   
   To exit the CLI mode and return to the CPS’s shell, type the following command:
   
   cli> quit

**IP Address and Netmask**

You can use the VI method or the CLI method to configure the IP address and network mask of the CPS.

**VI method**

To set the IP address (if DHCP client is disabled) and the netmask, 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 the IP address and 255.255.255.0 as the mask:

1. Open the `/etc/pslave/pslave.conf` file.

   To change these parameters, edit the `/etc/pslave/pslave.conf` file:
   
   # vi /etc/portslave/pslave.conf

2. Change parameter values.

   With the `/etc/pslave/pslave.conf` file open, search for the parameters in the following example, and replace the existing values with your IP address and network mask.

   ```plaintext
   conf.eth_ip    192.168.160.10
   conf.eth_mask  255.255.255.0
   conf.dhcp_client      0
   ```

   *File Description 4.2: /etc/pslave/pslave.conf*
**NOTE:** To define a static IP address, you must disable the DHCP client. Set the value of the following line to “zero”:

```
conf.dhcp_client 0
```

3. Activate the changes.

```
# runconf
```

4. Test the configuration.

To ensure that the ports are set up properly, ping the CPS from a remote machine. Using the Windows OS, open a command prompt window, type in the following command, and press Enter:

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

For example:

```
# ping 192.168.160.10
```

If you receive a reply, your CPS connection is configured correctly. If there is no reply, see **Appendix C: Cabling and Hardware Information**.

5. Use Telnet to access the server connected to the first port of the CPS. (This will only work if you selected `socket_server` or `socket_server_ssh` as your `all.protocol` parameter.)

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

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

For example:

```
# telnet 192.168.160.10 7001
```

If the network settings are configured correctly, a Telnet session will open on the server connected to port 1. If a Telnet session does not open, check the configuration, follow steps 1–5 again, and see “**Appendix B: Upgrades and Troubleshooting**”.

6. Save the changes.

```
# saveconf
```

**CLI method—IP address**

It is also possible to configure the IP address and netmask using the CLI interface. The following example will set 192.168.160.10 as the IP address and 255.255.255.0 as the mask.
1. Open the CLI interface by issuing the command:

   
   # CLI

2. Configure the IP address of the unit (where 192.168.160.10 is the IP address.)

   cli> config network hostsettings primipaddress 192.168.160.10

3. Configure the network mask address of the unit (where 255.255.255.0 is the subnet mask).

   cli> config network hostsettings primsubnetmask 255.255.255.0

4. Activate the configuration.

   cli> config runconfig

5. Save the configuration.

   cli> config savetoflash

6. Exit the CLI mode.

   To exit the CLI mode and return to the CPS’s shell, type the following command:

   cli> quit

### 4.3 DHCP Client

DHCP is a protocol that allows network administrators to assign IP addresses to network devices automatically. 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. With DHCP, the IP address assigned to a device is valid for a fixed period of time. This “lease” time can vary for each device. A short lease time can be used when there are more devices than available IP addresses. For more information, see [RFC 2131](https://tools.ietf.org/html/rfc2131).

### VI mode

The DHCP client on the Ethernet Interface can be configured in two different ways, depending on the action the CPS should take if 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):
   a. In the `/etc/portslave/pslave.conf` file, set the global parameter `conf.dhcp_client` to 1.
b. Still in the `portslave.conf` file, comment all other parameters related to the Ethernet Interface (`conf.eth_ip`, etc.).

c. Add the necessary options to the file `/etc/network/dhcpcd_cmd` (some options are described later in this chapter).

2. The CPS restores the last IP address provided, then assigns this IP address to the Ethernet Interface. The first time the unit receives power, the IP address restored is 192.168.160.10 in case of failure in the DHCP. By default, DHCP is enabled (`conf.dhcp_client 2`):

   a. Set the global parameter `conf.dhcp_client` to 2.
   b. Comment all other parameters related to the Ethernet Interface (`conf.eth_ip`, etc.).
   c. Add the following lines to the file `/etc/config_files` (by default, this content is already present in `/etc/config_files`):

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

   File Description 4.3: `/etc/config_files`

   d. 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 4.4: `/etc/network/dhcpcd_cmd`

   NOTE: From the factory, `/etc/network/dhcpcd_cmd` already has this content.

   e. Add all other necessary options to the file `/etc/network/dhcpcd_cmd` (some options are described later in this chapter).

   In both cases, if the IP address or the default gateway of the CPS are changed, the CPS adjusts the routing table accordingly.
Files related to DHCP:

Table 4-1: DHCP related files and commands

<table>
<thead>
<tr>
<th>Command/File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/bin/handle_dhcp</td>
<td>The script the DHCP client runs each time an IP address negotiation takes place.</td>
</tr>
<tr>
<td>/etc/network/dhcpd_cmd</td>
<td>Contains a command that activates the DHCP client (used by the cy_ras program). Its factory contents are: /bin/dhcpd -c /bin/handle_dhcp</td>
</tr>
</tbody>
</table>

Options that can be used on this command line:

- **-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.

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

CLI method—DHCP

1. Open the CLI interface by issuing the command:

   `# CLI`

2. Activate or deactivate DHCP in the unit. Values for `<option>` are **yes** to activate DHCP or **no** to deactivate DHCP.

   `cli> config network hostsettings dhcp <option>`
3. Activate the configuration.
   `cli> config runconfig`

4. Save the configuration.
   `cli> config savetoflash`

5. Exit the CLI mode.
   To exit the CLI mode and return to the CPS’s shell, type the following command:
   `cli> quit`

### 4.4 Routes and Default Gateway

The CPS has a static routing table. To view the table, use the following commands:

```
# route
```

or

```
# netstat -rn
```

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

```
```

**Table 4-2:** Actions and options for the `route` command

<table>
<thead>
<tr>
<th>Action/Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[add</td>
<td>del]</td>
</tr>
<tr>
<td>[-net</td>
<td>-host]</td>
</tr>
<tr>
<td>target</td>
<td>Target is the IP address of the destination host or network.</td>
</tr>
<tr>
<td>netmask</td>
<td>The tags 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 (for example, 255.255.255.0).</td>
</tr>
<tr>
<td>nt_msk</td>
<td></td>
</tr>
<tr>
<td>gw gt_way</td>
<td>Specifies a gateway, when applicable. gt_way is the IP address or hostname of the gateway.</td>
</tr>
</tbody>
</table>
**How to Configure the Default Gateway of the CPS**

**VI mode**

To add routes, edit the `/etc/network/st_routes` file using the following syntax:

```bash
```

The following example will set the default gateway to the IP address 192.168.0.1:

1. Open the `/etc/network/st_routes` file using the VI editor.
   
   Run the command:
   ```bash
   # vi /etc/network/st_routes
   ```

2. Insert the default route into the file using one of the following commands:
   ```bash
   # route add -net 0.0.0.0 netmask 0.0.0.0 gw 192.168.0.1
   or
   # route add default gw 192.168.0.1
   ```

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

3. Save the changes by running the command:
   ```bash
   # saveconf
   ```

**CLI method—routes**

You can access the following menu of the CLI interface to complete all configuration regarding the addition of static routes:

```bash
cli> config network stroutes
```
The example below shows how to add a default gateway in the unit:

1. Open the CLI interface by issuing the command:
   
   # CLI

2. Insert the default route. The default gateway for the following example is 192.168.0.1
   
   cli> config network stroutes add default gateway 192.168.0.1

3. Activate the configuration.
   
   cli> config runconfig

4. Save the configuration.
   
   cli> config savetoflash

5. Exit the CLI mode.
   
   To exit the CLI mode and return to CPS’s shell, type the following command:
   
   cli> quit

### 4.5 DNS Server and Domain Name

A 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 are presented together.

**VI mode**

To set the DNS server and the domain name of your network, edit the `/etc/resolv.conf` file. The following example configures “APC.com” as the domain name and 192.168.0.2 as the DNS server.

1. Open the `/etc/resolv.conf` file by running the command:
   
   # vi /etc/resolv.conf

2. Configure the `/etc/resolv.conf` file.
   
   The syntax of this file must be as follows:

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

   **File Description 4.5: /etc/resolv.conf**

3. Save the configuration by running the command:
   
   # saveconf
**CLI method—DNS and Domain Name**

The following example configures *APC.com* as the domain name and *192.168.0.2* as the DNS server.

1. Open the CLI interface by issuing the command:
   
   # CLI

2. Configure *APC.com* as the domain name.
   
   cli> config network hostsettings domain APC.com

   **NOTE:** This parameter is disregarded when DHCP is enabled.

3. Configure the IP address of the primary DNS server.
   
   cli> config network hostsettings primdnsserver 192.168.0.2

   **NOTE:** This parameter is disregarded when DHCP is enabled.

4. Activate the configuration.
   
   cli> config runconfig

5. Save the configuration.
   
   cli> config savetoflash

6. Exit the CLI mode.
   
   To exit the CLI mode and return to CPS’s shell, type the following command:
   
   cli> quit

### 4.6 Bonding

The CPS provides failover Ethernet bonding using a PCMCIA card as a second Ethernet port. Bonding enables redundancy for the Ethernet devices, using the standard Ethernet interface as the primary mode of access and one PCMCIA card as a secondary mode of access.
When bonding is enabled, both the Ethernet port and the PCMCIA cards are configured with the same IP address and the same MAC address. The PCMCIA interface automatically takes the place of the standard Ethernet interface if any conditions prevent access to the CPS through the primary Ethernet port. When the standard interface regains functionality, it automatically assumes its role as the primary interface, and all connection sessions are kept up with no interruption.

**VI mode**

To set the failover Ethernet bonding, edit the `/etc/bonding.opts` file.

1. Open the `/etc/bonding.opts` file by running the command:
   ```bash
   # vi /etc/bonding.opts
   ```
2. Configure the `/etc/bonding.opts` file
   
   The syntax of this file must be as follows:

   ```
   enabled = < YES | NO > <--- make the bonding feature active if true and inactive otherwise
   
   miimon = <positive integer value> <--- this parameter is the interval, in milliseconds, in which the active interface will be checked to see if it is still communicating.
   
   updelay = <positive integer value> <--- this parameter will define the time, in milliseconds, that the system will wait to make the primary interface active again after it has been detected as up.
   ```

   *File Description 4.6: `/etc/bonding.opts`

3. Save the configuration by running the command:
   ```bash
   # saveconf
   ```

**CLI method—bonding**

The following example configures `APC.com` as the domain name and 192.168.0.2 as the DNS server of the CPS.

1. Open the CLI interface by issuing the command:
   ```bash
   # CLI
   ```
2. Enter the bonding menu.
   ```bash
   cli> config network hostsettings bonding
   ```
3. Enable failover bonding.
   bonding> enabled yes
   To disable fail-over bonding, type the following command:
   bonding> enabled no
   **NOTE:** This parameter is disregarded when DHCP is enabled.

4. Configure the interval, in milliseconds, in which the active interface will be checked to see if it is still communicating.
   bonding> miimon <positive integer value>

5. Configure the time, in milliseconds, that the system will wait to make the primary interface active again after it has been detected as up.
   bonding> updelay <positive integer value>

6. Optionally, confirm values.
   bonding> show

   A display similar to the following example appears:

   ```
   bonding>show

   [bonding]
   enabled: no
   miimon: 100
   updelay: 200
   ```

   *File Description 4.7: Bonding Default Configuration*

7. Activate the configuration.
   cli> config runconfig

8. Save the configuration.
   cli> config savetoflash

   The failover is enabled.
9. Exit the CLI mode.
   To exit the CLI mode and return to CPS’s shell, type the following command:
   
   cli> quit

10. Check the bonding configuration
   To check if the feature is active, execute the `ifconfig` command in the Linux shell.

```
[root@CPS /]# ifconfig
```

```
bond0 Link encap:Ethernet  HWaddr 00:60:2E:00:4F:97
      inet addr:172.20.0.131  Bcast:172.20.255.255  Mask:255.255.0.0
      UP BROADCAST RUNNING MASTER MULTICAST  MTU:1500  Metric:1
      RX packets:484130 errors:0 dropped:0 overruns:0 frame:0
      TX packets:234236 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:0
      RX bytes:40453479 (38.5 MiB)  TX bytes:25311651 (24.1 MiB)

eth0    Link encap:Ethernet  HWaddr 00:60:2E:00:4F:97
      inet addr:172.20.0.131  Bcast:172.20.255.255  Mask:255.255.0.0
      UP BROADCAST RUNNING SLAVE MULTICAST  MTU:1500  Metric:1
      RX packets:237695 errors:0 dropped:0 overruns:0 frame:0
      TX packets:121503 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:100
      RX bytes:19993021 (19.0 MiB)  TX bytes:14356423 (13.6 MiB)
      Interrupt:9 Base address:0xe00

eth1    Link encap:Ethernet  HWaddr 00:60:2E:00:4F:97
      inet addr:172.20.0.131  Bcast:172.20.255.255  Mask:255.255.0.0
      UP BROADCAST RUNNING NOARP SLAVE MULTICAST  MTU:1500  Metric:1
      RX packets:246435 errors:0 dropped:0 overruns:0 frame:0
      TX packets:112733 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:100
      RX bytes:20460458 (19.5 MiB)  TX bytes:10955228 (10.4 MiB)
      Interrupt:9 Base address:0x300

lo      Link encap:Local Loopback
      inet addr:127.0.0.1  Mask:255.0.0.0
      UP LOOPBACK RUNNING  MTU:1500  Metric:1
      RX packets:0 errors:0 dropped:0 overruns:0 frame:0
      TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:0
      RX bytes:0 (0.0 iB)  TX bytes:0 (0.0 iB)
```

[root@CPS /]#
After the failover is enabled, the bonded Ethernet interfaces are referred to as “bond0”. eth0 and eth1 in the preceding example represent the two physical interfaces. To check which physical interface is the primary interface and which is the failover interface, look for the status “NOARP”. The interface with the “NOARP” status—eth1 in the example—is the failover. eth0 is sending and receiving packets, and eth1 is in active and standby mode.

**IMPORTANT:** If you have IP filtering rules set before bonding is activated, the interface reference in the firewall IP filtering will be eth0. You need to change the interface to bond0 in order to reference the bonded interface. For example, there is a rule to drop the SSH packets to access the CPS box with no bonding:

```
[root@CPS /]# iptables -A INPUT -p tcp –dport 22 -i eth0 -j REJECT
```

If you activate bonding, change the rule to reference the bonded interface:

```
[root@CPS /]# iptables -A INPUT -p tcp –dport 22 -i bond0 -j REJECT
```

For more information on setting up filters and the structure of iptables, see “Filters and Network Address Translation” on page 95.

### 4.7 Hosts

The `/etc/hosts` 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 hostnames 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 the `/etc/hosts` file:

1. Open the `/etc/hosts` file by running 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 as follows:

<table>
<thead>
<tr>
<th>IP Address</th>
<th>Host Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>127.0.0.1</td>
<td>localhost</td>
</tr>
<tr>
<td>192.168.160.10</td>
<td>CPS</td>
</tr>
<tr>
<td>192.168.160.2</td>
<td>dns-server</td>
</tr>
</tbody>
</table>

File Description 4.8: /etc/hosts

Enter as many hosts as necessary.

2. Save the configuration by running the command:

   # saveconf

**CLI method—hosts**

The following example will add a host named *test* with the IP address *192.168.0.111* in the
known hosts file of the CPS.

1. Open the CLI interface by issuing the command:

   # CLI

2. Add a host named *test* with an IP address *192.168.0.111*.

   cli> config network hosttable add hostip 192.168.0.111 name test

   You can repeat this step as many times as necessary.

3. Activate the configuration.

   cli> config runconfig

4. Save the configuration.

   cli> config savetoflash

5. Exit the CLI mode.

   To exit the CLI mode and return to CPS ’s shell, type the following command:

   cli> quit

**4.8 TCP Keepalive**

This feature allows the CPS to recognize when the socket client (SSH or Telnet) terminates
without closing the connection properly. Currently, if this happens in a serial port, the system
administrator must close the connection manually or no user can access that port.
How It Works

The TCP engine of CPS 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 CPS (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.
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 4.9: `/bin/init_proc_fs`

CLI method—TCP Keep Alive

1. Open the CLI interface by issuing the command:
   
   # CLI

2. Configuring the pool interval (ms).

   The following command sets a 50 ms pool interval.

   cli> config physicalports all other tcpkeepalive 50

3. Activate the configuration.

   cli> config runconfig
4. Save the configuration.
   `cli> config savetoflash`

5. Exit the CLI mode.
   `cli> quit`

### 4.9 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 allowing 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 CPS, 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 CPS 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 can count the packets which match a rule, and can 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. Three independent tables are supported by the iptables, but only two will be used:

- `filter`: This is the default table.
- `nat`: This table 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.

Filter table:

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

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 the following information:

- 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 main page of the iptables Web site or see 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**—One of the following:
  - for the **filter** table: INPUT, OUTPUT, FORWARD, or a user-created chain
  - for the **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 of 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.

**Table 4-3: iptables commands options**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-A --append</td>
<td>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.</td>
</tr>
<tr>
<td>-D --delete</td>
<td>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.</td>
</tr>
<tr>
<td>-R --replace</td>
<td>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.</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>-I --insert</td>
<td>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.</td>
</tr>
<tr>
<td>-L --list</td>
<td>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.</td>
</tr>
<tr>
<td>-F --flush</td>
<td>Flush the selected chain. This is equivalent to deleting all the rules one-by-one.</td>
</tr>
<tr>
<td>-Z --zero</td>
<td>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.)</td>
</tr>
<tr>
<td>-N --new-chain</td>
<td>New chain. Create a new user-defined chain by the given name. There must be no target of that name already.</td>
</tr>
<tr>
<td>-X --delete-chain</td>
<td>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 chain that is not built into the table.</td>
</tr>
<tr>
<td>-P --policy</td>
<td>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.</td>
</tr>
<tr>
<td>-E --rename-chain</td>
<td>Rename the user-specified chain to the user-supplied name. This is cosmetic, and has no effect on the structure of the table.</td>
</tr>
<tr>
<td>-h --help</td>
<td>Help. View a brief description of the command syntax.</td>
</tr>
</tbody>
</table>
4.10 Rule Specification

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

Table 4-4: *iptables* rules specifications

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| -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. The protocol all will match with all protocols and is taken as the 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. |
| -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. |
### Table 4-4: `iptables` rules specifications

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| -j        | - - jump target  
  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, matching the rule will have no effect on the packet's fate, but the counters on the rule are incremental. The special built-in targets are:  
  - ACCEPT—let the packet through.  
  - DROP—drop the packet.  
  - QUEUE—pass the packet to userspace (if supported by the kernel).  
  - RETURN—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 through 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. The string “+” will match with any interface name. |
| -o        | - -out-interface[!]name  
  Optional name of an interface through 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 the -o option is omitted, the string "+", which will match with any interface name, is assumed. |
**Table 4-4: iptables rules specifications**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>![] -f -fragment</td>
<td>This parameter means 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. When the &quot;!&quot; argument precedes the &quot;-f&quot; flag, the rule will only match head fragments, or unfragmented packets.</td>
</tr>
<tr>
<td>-c - -set-counters PKTS BYTES</td>
<td>Enables the administrator to initialize the packet and byte counters of a rule (during INSERT, APPEND, and REPLACE operations).</td>
</tr>
<tr>
<td>-v - -verbose</td>
<td>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 (see the -x flag to display the exact value of the packet and byte counters). For appending, insertion, deletion, and replacement, this parameter prints detailed information on the rule or rules.</td>
</tr>
<tr>
<td>-n - -numeric</td>
<td>Numeric output. IP addresses and port numbers are printed in numeric format. By default, the program will try to display them as host names, network names, or services (whenever applicable).</td>
</tr>
<tr>
<td>-x - -exact</td>
<td>Expand numbers. Display the exact value of the packet and byte counters, instead of the rounded number in K (multiples of 1000), M (multiples of 1000K), or G (multiples of 1000M). This parameter is only relevant for the -L command.</td>
</tr>
<tr>
<td>- -line-numbers</td>
<td>When listing rules, add line numbers to the beginning of each rule, corresponding to that rule's position in the chain.</td>
</tr>
</tbody>
</table>
Match Extensions

Iptables can use extended packet matching modules, which 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, 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.

<table>
<thead>
<tr>
<th>Table 4-5: TCP extensions</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>TCP extension</strong></td>
</tr>
<tr>
<td>--source-port [!] [port[:port]]</td>
</tr>
<tr>
<td>--destination-port [!] [port[:port]]</td>
</tr>
<tr>
<td>--tcp-flags [!] mask comp</td>
</tr>
</tbody>
</table>
UDP Extensions

These extensions are loaded if the protocol udp is specified or “-m udp” is specified.

Table 4-6: UDP extensions

<table>
<thead>
<tr>
<th>UDP extension</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>--source-port [!] [port[:port]]</td>
<td>Source port or port range specification. See the description of the -source-port option of the TCP extension.</td>
</tr>
<tr>
<td>--destination-port [!] [port[:port]]</td>
<td>Destination port or port range specification. See the description of the -destination-port option of the TCP extension.</td>
</tr>
</tbody>
</table>
ICMP Extension

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

Table 4-7: ICMP extensions

<table>
<thead>
<tr>
<th>ICMP extension</th>
<th>Description</th>
</tr>
</thead>
</table>
| --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 |

Multiport Extensions

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

Table 4-8: Multiport extensions

<table>
<thead>
<tr>
<th>Multiport extension</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>--source-port [port[,port]]</td>
<td>Match if the source port is one of the given ports.</td>
</tr>
<tr>
<td>--destination-port [port[,port]]</td>
<td>Match if the destination port is one of the given ports.</td>
</tr>
<tr>
<td>--port [port[,port]]</td>
<td>Match if both the source and destination ports are equal to each other and to one of the given ports.</td>
</tr>
</tbody>
</table>

Target Extensions

Iptables can use extended target modules. The following modules 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 information on all matching packets (e.g., IP header fields) via the kernel log (where it can be read with syslog-ng).

**Table 4-9: LOG extensions**

<table>
<thead>
<tr>
<th>LOG extension</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>--log-level level</td>
<td>Level of logging (numeric or see syslog.conf(5)).</td>
</tr>
<tr>
<td>--log-prefix prefix</td>
<td>Prefix log messages with the specified prefix up to 29 letters long. Useful for distinguishing messages in the logs.</td>
</tr>
<tr>
<td>--log-tcp-sequence</td>
<td>Log TCP sequence numbers. This is a security risk if the log is readable by users.</td>
</tr>
<tr>
<td>--log-tcp-options</td>
<td>Log options from the TCP packet header.</td>
</tr>
<tr>
<td>--log-ip-options</td>
<td>Log options from the IP packet header.</td>
</tr>
</tbody>
</table>
**REJECT (filter table only)**

This module is used to return 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:

<table>
<thead>
<tr>
<th>LOG extension</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>--reject-with type</td>
<td>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 it 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 returned. 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).</td>
</tr>
</tbody>
</table>

**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:

<table>
<thead>
<tr>
<th>SNAT target</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>--to-source &lt;ipaddr&gt;[&lt;ipaddr&gt;][:port-port]</td>
<td>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. Where possible, no port alteration will occur.</td>
</tr>
</tbody>
</table>
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:

**Table 4-12:** DNAT target

<table>
<thead>
<tr>
<th>DNAT target</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>--to-destination &lt;ipaddr&gt;[-&lt;ipaddr&gt;][:port-port]</td>
<td>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.</td>
</tr>
</tbody>
</table>

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, 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 causes connections to be 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:

**Table 4-13:** Masquerade target

<table>
<thead>
<tr>
<th>Target</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>--to-ports &lt;port&gt;[-&lt;port&gt;]</td>
<td>Specifies a range of source ports to use, overriding the default SNAT source port-selection heuristics (see “SNAT (NAT table only)” on page 106). This is only valid if the rule also specifies -p tcp or -p udp.</td>
</tr>
</tbody>
</table>
REDIRECT (NAT table only)

This target is only valid in the NAT table, in the PREROUTING and OUTPUT chains and in 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:

<table>
<thead>
<tr>
<th>Target</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>--to-ports &lt;port&gt;[-&lt;port&gt;]</td>
<td>Specifies a range of source ports to use, overriding the default SNAT source port-selection heuristics (see “SNAT (NAT table only)” on page 106). This is only valid if the rule also specifies -p tcp or -p udp.</td>
</tr>
</tbody>
</table>

How to Configure iptables Rules

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

VI method

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.

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

   See “Syntax” on page 97 for details of the iptables syntax.


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

4. Execute updatefiles /etc/network/firewall.

   This program will save the configuration to the flash memory.
4.11 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, etc.). The implementation of IPSec used by the CPS is Openswan 2.3.0.

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, firewall machines, various application servers, and 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 protocol connects the CPS securely to a host or to a whole network configuration frequently called host-to-network and host-to-host tunnel. Considering practical aspects, this is the same thing as a VPN, but with IPSec, 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 almost 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 a remote location (e.g., a hotel).

A detailed description of these applications is below. See “Before you start” on page 110 to build the two applications.

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 secure. 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 network and decrypt packets leaving it, creating a secure tunnel through it.
Road Warriors

The typical Road Warrior is a traveler connecting to the Console Server from a laptop machine. For the 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.

Road Warrior connections require a different setup than VPN gateways with static addresses and with client systems behind them. In most situations, Openswan supports Road Warrior connections. However, difficulties may appear for some Road Warrior connections:

- If Road Warriors obtain addresses using DHCP, Openswan can build and use a tunnel to such an address, but when the DHCP lease expires Openswan 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, it breaks IPsec. IPsec authenticates packets on an end-to-end basis, to ensure they are not altered en route.

Before you start

This is a quick guide to setting 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 documenting the configuration of the remote side (the equipment with which the CPS will create a tunnel), these examples assume the remote side is also running Openswan. If it is not, adjust the instructions to accommodate your IPsec software.

Set up and test networking. Before you set up Openswan, configure and test IP networking on the Console Server and on the remote side. IPsec cannot function without a working IP network beneath it. Many reported Openswan problems are actually problems with routing or firewalling. If IPsec problems occur, it is extremely difficult to recognize and debug them unless the underlying network is configured correctly.

Enabling IPsec on your CPS. The IPsec is disabled by default in the Console Server family. To enable it, edit the file /etc/daemon.d/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 save /etc/daemon.d/ipsec.sh with the command “saveconf”.

110 APC CPS Command Reference Guide
Road Warrior Configuration

If an administrator wants to access the CPS securely from wherever he is (e.g., his office desk, his house, or a hotel room), his IP address will not always be the same. For IPsec purposes, he is a “Road Warrior”. This manual refers to remote machines as Road Warriors, but for IPsec, any machine with a dynamic IP address is a Road Warrior.

Necessary information

To set up a Road Warrior connection, you need information about the system on the other end. Connection descriptions use “left” and “right” to designate the two ends. From the Console Server, “left” describes the local system, and “right” describes the remote system.

The Console Server administrator must have the following information about each Road Warrior:

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

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

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

The output should look like this (in this example, the key is shortened for easy reading):

```
rightrsasigkey=AQNe6hpbROGVES6uXeCxpnd88fdafp00w5OT0s1LgR7/oUM...
```

The Road Warrior needs to know:

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

Generate the information by issuing the following command on the Console Server:

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

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=@cps.example.com leftrsasigkey=0sLgR7/oUM...
```

Typically, the Console Server administrator only needs to generate this information once. The same file can be given to all warriors.
Setup on the Road Warrior machine

Add a connection description `us-to-cps`, with the left and right information you gathered in the preceding section, to the `ipsec.conf` file of the warrior system. For example:

```plaintext
# pre-configured link to Console Server
conn us-to-cps

    # information obtained from Console Server admin
    left=1.2.3.4 # Console Server IP address
    leftid=@cps.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 4.1: Road Warrior ipsec.conf file*

**WARNING!** The connection description "conn us-to-cps" 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 CPS

To add Road Warrior support so people can connect remotely to your Console Server, create the file `/etc/warrior.connection` and add the following lines to the file:

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

File Description 4.2: CPS 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

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).
Insert this connection description in your `ipsec.conf` file with the variables that fit your environment:

```plaintext
# sample tunnel
# The network here looks like:
# CPS ----cpsnexthop------rightnexthop------right===rightsubnet
# If CPS and right are on the same Ethernet, omit leftnexthop and
# rightnexthop.
conn sample
    # CPS
    left=10.0.0.1
    leftid=@cps.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 4.3: 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 (or copying and pasting) the entire rsa key in the fields “leftrsasigkey” and “rightrsasigkey” inside the `/etc/ipsec.conf` file, the administrator can type in the filename where the rsa key was generated. For example:

```
leftrsasigkey=@file /etc/CPS48AL.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.
This connection descriptor can be added to both the Console Server and the other end. This is the advantage of using left and right parameters instead of local and 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 the Console Server sends packets to in order to get them delivered to right). Similarly, you may need to specify rightnexthop (the router the right sends packets to in order to get them delivered to the Console Port Server).

**Authentication keys**

To build a connection, the Console Server and the other end must be able to authenticate each other. For Openswan, the default method is public key authentication based on the RSA algorithm. IPsec does allow several other authentication methods. This chapter describes how to generate authentication keys and how to exchange keys between systems.

**Generating an RSA key pair.** The Console Server does not have an RSA key pair by default. It is generated the first time you restart the CPS after you enable the IPsec daemon in the file `/etc/daemon.d/ipsec.sh`. You also can generate your key pair by issuing the following commands as root:

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

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

Key generation may take some time. Additionally, the Console Server requires many random numbers, and it needs and uses traffic on the Ethernet to generate them. It is also possible to use keys in other formats, not generated by Openswan. This may be necessary for interoperation with other IPsec implementations.

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

If your CPS is the left side of the tunnel:

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

If your CPS is the right side of the tunnel:

```bash
# /usr/local/sbin/ipsec showhostkey --right
```
These two commands produce the key formatted for insertion in an *ipsec.conf* file. Public keys need not be protected as carefully 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, you can put a gateway key on a Web page, make it available in DNS, or transmit it using an insecure method such as e-mail.

**IPsec Management**

After you complete the configuration, you need to manage all tunnels and manage IPsec. This section lists a few useful commands for 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
```

This command also accepts usual daemon commands, such as stop and restart.

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

**NAT-Transversal**

CPS 2.6 uses Openswan 2.3.0, which has support for NAT-Transversal (NAT-T). NAT-T allows IPSec to be used behind any NAT device by encapsulating ESP (Encapsulated Security Payload) in UDP.

Add the following line to the `/etc/ipsec.conf` file to enable NAT-Transversal.

```
nat_transversal=yes
```

**Adding or removing a connection**

All of the connections can be loaded to the IPsec database at start-up if these connections have the *auto* parameter set to add. If a connection does not have this option set, use the following command to add the connection manually:

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

To take a connection out of the IPsec database, 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”.

**Starting or stopping a connection**

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

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

To close a tunnel, use the command:

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

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

```
[root@cps_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 whack

The ipsec whack command shows the status of the connections.

[root@cps_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[@CPS ]
000 "test": ike_life: 3600s; ipsec_life: 28800s; rekey_margin: 540s; rekey_fuzz: 100%;
000 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;
000 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
000 ISAKMP

As you can see, this command shows almost the same information shown by the ipsec auto-up command. Use this command if the -up command does not display any status information on the screen (this can happen, depending on the Console Port Server Syslog configuration).

The IPsec Configuration Files in Detail

This section describes the file /etc/ipsec.conf in detail.

Description

The ipsec.conf file specifies most configuration and control information for the Openswan IPsec subsystem. (The major exception is secrets for authentication, ipsec.secrets) Its contents are not security-sensitive unless manual keying is being used for more than just testing. In that case, the encryption and authentication keys in the descriptions for the manually-keyed connections are very sensitive, and the connection descriptions should be kept in a separate file, using the include facility. See “include ipsec.*.conf” on page 119 for more information about the include facility.

The file is a text file consisting of one or more sections. Any line that begins with white space, followed by #, followed by any characters to the end of the line, is a comment and is ignored. Empty lines that are not within a section are also ignored.
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 `include` file. Such inclusions can be nested. Only a single filename can be supplied. The filename cannot contain white space, but it can include shell wildcards. For example:

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

The `include` facility permits the user to keep information about connections, or sets of connections, separate from the main configuration file. This separation permits the user to change connection descriptions, copy the connection descriptions to the other security gateways involved, etc., without having to repeatedly extract them from the configuration file and then insert them back into it.

---

**NOTE:** The `also` parameter permits splitting a single logical section (e.g., a connection description) into several actual sections. For more information about the `also` parameter, see “also” on page 120.

---

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 other sections of the same type. (Names must start with a letter and may contain only letters, digits, periods, underscores, and hyphens.) All subsequent lines that are not empty and that begin with white space are part of the section; comments within a section must begin with white space also. There may be only one section of a given type with a given name.

Lines within the section are generally of the form:

```
parameter=value
```

(Note the mandatory preceding TAB.) White space is allowed on either side of the equals sign (=). Parameter names follow the same syntax as section names, and are specific to a section type. Unless otherwise 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 quotation marks ("); a value itself cannot contain a quotation mark, and it cannot be continued across more than one line.
Each numeric value is specified to be either an integer (a sequence of digits) or a decimal number (sequence of digits optionally followed by a decimal point [.] and another sequence of digits).

Currently, one parameter 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 section, and must have the same section type. (Nesting is permitted, and there may be more than one also in a single section, although appending the same section more than once is forbidden.) 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, and a conn section specifies an IPsec connection.

**Conn sections**

A conn section contains a connection specification that defines 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. For 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 4.4: 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 participants. Which participant is considered left or right is arbitrary; IPsec identifies which end it is being run on based on internal information. This permits the use of identical connection specifications on both ends.

Many of the parameters relate to one participant or the other. Only the parameters 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 is not necessary 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, the two ends must agree exactly on the values of these parameters. The two ends can be defined as Left and Right, or Local and Remote.

- **type:** The type of the connection. Currently accepted values are *tunnel* (default setting; signifies a host-to-host, host-to-subnet, or subnet-to-subnet tunnel), *transport* (signifies host-to-host transport mode), and *passthrough* (supported only for manual keying; signifies that no IPsec processing should be done).
- **left (local) and right (remote) IP:** The IP address of the participant's 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`. (Note: 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.
- **left (local) and right (remote) subnet:** Private subnet behind the left and right participants, expressed as network/netmask.
- **left (local) and right (remote) nexthop:** NextHop gateway IP address for the left and right participant connection to the public network.
- **left (local) and right (remote) updown script:** What updown script to run to adjust routing or firewalling when the status of the connection changes. The path to the default updown script on CPS is `/usr/local/lib/ipsec/_updown`
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, the two ends must 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 route plus an ipsec auto --route), `start` (signifying that plus an ipsec auto --up), and `ignore` (the default) (signifying no automatic startup operation). This parameter is ignored unless the `plutoload` or `plutostart` configuration parameter is set suitably; see “Config setup section” on page 124.

- **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** and **rightid**: How the left and right 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** and **rightrsasigkey**: The left and right participants’ public key for RSA signature authentication, in RFC 2537 format. The magic value `%none` is equivalent to not specifying a value (useful for overriding 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**: Do not use two connection descriptions to specify different public keys for the same `leftid`. Only one connection description is permitted per public key in the same `leftid`.

- **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` (seconds), or a decimal number followed by `m` (minutes), `h` (hours), or `d` (days). Default setting: `8.0h`; maximum setting: `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, attempts to negotiate a replacement should begin. Acceptable values are an integer optionally followed by s (seconds), or a decimal number followed by m (minutes), h (hours), or d (days). Default setting: 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 the percent sign (%).

• keyingtries: (An integer) How many attempts 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 are an integer optionally followed by s (seconds), or a decimal number followed by m (minutes), h (hours), or d (days).

• 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, the two ends must agree exactly on the values of these parameters. A manually-keyed connection must specify at least one AH or ESP parameter.

• 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.

• ah: AH authentication algorithm to be used for the connection, e.g. hmac-md5-96.

• ahkey: Required if AH is present. AH authentication key

• ahreplay_window: AH replay-window setting. An integer from 0 to 64.
Config setup section

The only config section known to the IPsec software is setup, which contains information used when the software is being started. For example:

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

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 finds 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 is not already on) as IPsec is started, and turn it off again as IPsec is stopped. Acceptable values are yes and (the default) no.
- **klipsdebug**: How much KLIPS debugging output to log:
  - An empty value, or the magic value none—no debugging output (the default)
  - The magic value all—full output
- **plutodebug**: How much Pluto debugging output to log:
  - An empty value, or the magic value none—no debugging output (the default)
  - The magic value all—full output
- **dumpdir**: In what directory should daemons started by setup (notably the Pluto daemon) be allowed to dump information to. The empty value (the default) means the daemons will not dump information.
- **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 magic value 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 plutoload. 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 whether 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 (e.g., a command to decrypt an encrypted copy of the ipsec.secrets file). 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., a command to remove a decrypted copy of the ipsec.secrets file).

- **fragicmp**: Whether a tunnel needs to fragment a packet should be reported back with an ICMP (Internet Control Message Protocol) message, in an attempt to make the sender lower his PMTU (Path Maximum Transmission Unit) 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 (unless the user is an expert), 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 to which the MTU of each of the IPsec interfaces should be set, overriding IPsec's (large) default. This parameter is needed only in special situations.

---

**CLI method—VPN configuration**

It is possible to configure most VPN settings using the CLI interface.
NOTE: You cannot generate the RSA keys or copy the keys of remote hosts from the CLI interface. Complete those tasks before entering the CLI mode.

1. Open the CLI interface by issuing the command:
   
   `# CLI`

2. Access the VPN menu.

   `cli> config network vpn`

   This menu lets you add, edit or delete a VPN connection. When adding or editing a connection, you’ll be prompted to configure the following parameters:

   **Table 4-15: VPN parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>connectionname</td>
<td>&lt;name&gt; - Edit mode only</td>
</tr>
<tr>
<td>authprotocol</td>
<td>&lt;esp</td>
</tr>
<tr>
<td>authmethod</td>
<td>&lt;rsa</td>
</tr>
<tr>
<td>rightid</td>
<td>&lt;id&gt;</td>
</tr>
<tr>
<td>rightip</td>
<td>&lt;n.n.n.n&gt;</td>
</tr>
<tr>
<td>rightnexthop</td>
<td>&lt;hop&gt;</td>
</tr>
<tr>
<td>rightsubnet</td>
<td>&lt;n.n.n.n&gt;</td>
</tr>
<tr>
<td>rightrsa</td>
<td>&lt;rsa key&gt;</td>
</tr>
<tr>
<td>leftid</td>
<td>&lt;id&gt;</td>
</tr>
<tr>
<td>leftip</td>
<td>&lt;n.n.n.n&gt;</td>
</tr>
<tr>
<td>leftnexthop</td>
<td>&lt;hop&gt;</td>
</tr>
<tr>
<td>leftsubnet</td>
<td>&lt;n.n.n.n&gt;</td>
</tr>
<tr>
<td>leftrsa</td>
<td>&lt;rsa key&gt;</td>
</tr>
<tr>
<td>bootaction</td>
<td>&lt;ignore</td>
</tr>
<tr>
<td>secret</td>
<td>&lt;secret&gt;</td>
</tr>
</tbody>
</table>

See “Conn parameters: General” on page 121 for information about each parameter.
3. Activate the configuration.
   cli> config runconfig

4. Save the configuration.
   cli> config savetoflash

5. Manage the connection.
   After you configure the VPN connection, you must manage the VPN connections in the prompt shell. The CLI does not provide management utilities. For more information, see “IPsec Management” on page 116.

6. Exit the CLI mode.
   To exit the CLI mode and return to CPS’s shell, type the following command:
   cli> quit
Chapter 5: Administration

This chapter explains the tasks 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 Timezone**
- **Session Sniffing**
- **Saveconf and Restoreconf**
- **Start and Stop Services**
- **Security Profiles**

5.1 **SNMP**

Simple Network Management Protocol (SNMP) is a set of protocols for managing complex networks. 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.


**IMPORTANT!** Check the SNMP configuration before gathering information about CPS by SNMP. An unauthorized user can implement different types of attacks to retrieve sensitive information contained in the MIB. By default, the SNMP configuration in the CPS does not permit the public community to read SNMP information.
To use SNMP version 1 or 2 (which perform authentication by community), you must configure
the communities in the SNMP configuration file (/etc/snmp/snmpd.conf). For example, to
include the communities APC and public, you must add the following lines in /etc/snmp/

\[
\begin{align*}
\text{# APC is read-write community} \\
\text{rwcommunity APC} \\
\text{# public is a read-only community} \\
\text{rocommunity public}
\end{align*}
\]

\text{File Description 5.1: part of the /etc/snmp/snmpd.conf file}

To use SNMP version 3 (which performs authentication by username and password), perform
the following steps:

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

\[
\text{# createUser <username> MD5 <password> DES}
\]

For example:

\[
\text{# createUser usersnmp MD5 user_snmp_passwd DES}
\]

\text{NOTE: The SNMP v3 password must be less than 31 characters.}

2. Edit the /etc/snmp/snmpd.conf file.

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

\[
\text{# rouser <username> (eg.: rouser usersnmp)}.
\]

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

\[
\text{# rwuser <username> (eg.: rwuser usersnmp)}.
\]

3. Include the following line in /etc/config_files:

\[
\text{/etc/snmp/snmpd.local.conf}
\]

For information about configuring the /etc/snmp/snmpd.conf file, see
“Configuration” on page 132.
SNMP Reference Documentation

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
   - RFC2573—SNMP Applications.
   - RFC2575—View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP).

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 APC Vendor MIB (enterprises.2925)
   CPS remote Management Object Tree (APC.4)—This MIB allows you to obtain information about the product, read/write some configuration items, and perform some administration commands. (For more details, see the APC MIB file.)
Configuration

This section describes how to configure SNMP using the VI editor or the CLI interface.

VI method—involved parameters and passed values

You can configure SNMP v1 and v2 using the /etc/snmp/snmpd.conf file.

1. To define the public community, which allows read-only access to the MIB values, insert the following line in the /etc/snmp/snmpd.conf file.
   ```
   rocommunity public "default", hostname, or network/mask .1
   ```

2. Save the configuration changes in the snmp.conf file.
   ```
   [root@CPS root]# saveconf
   ```

3. Restart the SNMP daemon to read the new configuration.
   ```
   [root@CPS root]# daemon.sh restart SNMP
   ```

CLI method—SNMP

You can configure SNMP v1, v2 and v3 using the CLI interface.

1. Open the CLI interface by issuing the command:
   ```
   # CLI
   ```

2. To configure SNMP v1/v2, enter the following command:
   ```
   cli>config network snmp v1v2 add community test1 oid .1 permission ro source 192.168.0.200
   ```
   This command configures SNMP v1/v2 with the following characteristics:
   • community: test1
   • OID: .1
   • permission: ro (read-only)
   • source (allowed host): 192.168.0.200

3. To configure SNMP v3, enter the following command:
   ```
   cli>config network snmp v3 add username john password john1234 oid .1 permission ro
   ```
This command configures SNMP v1/v2 with the following characteristics:

- **username:** John
- **password:** John1234
- **OID:** .1
- **permission:** ro (read-only)

**NOTE:** The SNMP v3 password must be less than 31 characters.

4. Activate the configuration.

   cli> config runconfig

5. Test the configuration.

   In the example below, the targeted CPS has the IP address 192.168.0.1 and the Linux machine issuing the commands has an IP address of 192.168.0.200. Run the following command:
   - **SNMP v1/v2:**
     
     ```
     # snmpwalk -v 2c -c test1 192.168.0.1 .1
     ```
   - **For SNMP v3:**
     
     ```
     # snmpwalk -v 3 -u john -l authpriv -a MD5 -A John1234 -x DES -X John1234 192.168.0.1 .1
     ```

6. Save the configuration.

   cli> config savetoflash

7. Exit CLI mode.

   To exit the CLI mode and return to the CPS’s shell, type the following command:

   cli> quit

### 5.2 CronD

CronD is a service provided by the CPS that allows automatic, periodically-run custom-made scripts. It replaces the need to manually run commands.
How to Configure crond

Configuration of the crond daemon in the CPS is divided into three parts:

- /etc/crontab_files—The name of this file must not be changed, and it must point only to ONE file.
- source file—This file holds information about which files should be executed and the frequency of execution. It can have any name, since it is pointed out by /etc/crontab_files.
- script files—These are the script files scheduled and pointed out by the source file.

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 is 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. Set the pathname using the traditional crontab file format:

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

*File Description 5.2: /etc/crontab_files*

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

The /etc/crontab_files file can point to any file that calls the scripts to be run. The CPS has an example file (/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 5.3: /etc/tst_cron.src*
This file is called /etc/tst_cron.src, but it could have any name, since it follows the structure. The fourth line of the example file follows this structure: minutes, hours, month day, month, week day and command.
You can specify different tasks to run on different dates and times. Each command must be on a separate line. For more information, see “Crontab Syntax” on page 135.

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 it finds 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. Month also accepts names, e.g., jan, Feb (case insensitive), but the names format does not support ranges.
- Wday—day of week of execution, 0–7 (0 or 7 is sunday). Wday also accepts names, e.g. sun, Mon (case insensitive).
- Command—Anything that can be launched from the command line

Possible values for the fields:

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

VI Method—Involved Parameters and Passed Values

This example demonstrates how to configure a script named tst_cron.sh to run every minute.

1. Activate the crond daemon in the /etc/crontab_files.

As explained in the section “CronD” on page 133, 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 as shown below:

```bash
active root /etc/tst_cron.src
```
2. Edit the `/etc/tst_cron.src` file, 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 (in this example, the `tst_cron.sh` script will run every minute):

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

3. Save the changes by executing the following command:

```
# saveconf
```

4. Activate changes.

To activate the changes, reboot the CPS by issuing the command:

```
# reboot
```

### 5.3 Dual Power Management

**NOTE:** This section applies only to the dual power supply model of the CPS (APC part number AP9303).

The CPS comes with two power supplies which it can self-monitor. If either power supply fails, an audible alarm sounds and a syslog message is generated. This automanagement can be enabled (the default setting) or disabled (no actions are taken) by issuing a command:

```
# signal_ras buzzer on
or
# signal_ras buzzer off
```

To disable the buzzer at start-up, edit the shell script `/bin/ex_wdtLed.sh` and remove the keyword “buzzer.” The buzzer will not 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 for future start-ups, create a line with “/bin/ex_wdtLed.sh” in `/etc/config_files`, save and quit that file and run `saveconf`.

**How to Configure Dual-power Management**

There are no parameters to be configured.

To generate alarms in case of a power failure, you must change the `syslog-ng.conf` file. See “Generating Alarms (Syslog-ng)” on page 156.
5.4 Syslog-ng

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

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. Syslog-ng reads messages from sources (files, TCP/UDP connections, or syslogd clients), filters the messages, and takes an action. Actions include writing to files, sending SNMP traps, paging, sending e-mail, or sending Syslog messages to remote servers.

To configure syslog-ng, you must complete five steps.

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

See “Syslog-ng and its configuration” on page 138 for an explanation of each step.

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 or alarm is active. When nonzero, the contents of the data buffer are sent to 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 facility LOCAL[0+conf.DB_facility].
The Syslog Functions
This section shows the characteristics of the syslog-ng that are implemented for all members of the CPS 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
To configure Syslog, you must complete five steps.

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:

   **Table 5-1:** “Global Options” parameters (Syslog-ng configuration)

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>time_reopen(n)</td>
<td>The time to wait before a dead connection is reestablished.</td>
</tr>
<tr>
<td>time_reap(n)</td>
<td>The time to wait before an idle destination file is closed.</td>
</tr>
<tr>
<td>sync_freq(n)</td>
<td>The number of lines buffered before written to file. (The file is synced when this number of messages has been written to it.)</td>
</tr>
<tr>
<td>mark_freq(n)</td>
<td>The number of seconds between two MARKS lines.</td>
</tr>
<tr>
<td>log_fifo_size(n)</td>
<td>The number of lines fitting to the output queue.</td>
</tr>
<tr>
<td>chain_hostname</td>
<td>Enable/disable the chained hostname format.</td>
</tr>
<tr>
<td>(yes/no) or long_hostname (yes/no)</td>
<td>Enable/disable the chained hostname format.</td>
</tr>
</tbody>
</table>
2. Define sources using this statement:

```plaintext
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 parameters are required; others are optional.

The following source-drivers are available:

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>use_time_recvd (yes/no)</td>
<td>Use the time a message is received instead of the one specified in the message.</td>
</tr>
<tr>
<td>use_dns (yes/no)</td>
<td>Enable or disable DNS usage. syslog-ng blocks DNS queries, so enabling DNS may lead to a Denial of Service attack.</td>
</tr>
<tr>
<td>gc_idle_threshold(n)</td>
<td>Sets the threshold value for the garbage collector (gc; a form of automatic memory management), when syslog-ng is idle. GC phase starts when the number of allocated objects reach this number. Default: 100.</td>
</tr>
<tr>
<td>gc_busy_threshold(n)</td>
<td>Sets the threshold value for the garbage collector. When syslog-ng is busy, GC phase starts.</td>
</tr>
<tr>
<td>create_dirs(yes/no)</td>
<td>Enable the creation of new directories.</td>
</tr>
<tr>
<td>owner(name)</td>
<td>Set the owner of the created file. Default: root.</td>
</tr>
<tr>
<td>group(name)</td>
<td>Set the group of the created file. Default: root.</td>
</tr>
<tr>
<td>perm(mask)</td>
<td>Set the permission mask of the created file. Default: 0600.</td>
</tr>
</tbody>
</table>

---

**Table 5-2: “Source Drivers” parameters (Syslog-ng configuration)**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>internal()</td>
<td>Messages are generated internally in syslog-ng.</td>
</tr>
</tbody>
</table>
Examples of Defined Sources

a. Defining a source to read from a file:

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

• To read messages from “/temp/file1” file:
  
  source file1 {file('/temp/file1');};

• To receive messages from the kernel:
  
  source s_kernel { file('/proc/kmsg'); };
b. **Defining a source to receive messages from local syslogd clients:**

source sysl { unix-stream('/dev/log'); };

c. **Defining a source to receive messages from remote syslogd clients:**

source s_udp { udp(ip(<cliente ip>) port(<udp port>)); };

• To listen to messages from all machines on UDP port 514:
  
  source s_udp { udp(ip(0.0.0.0) port(514)); };

• To listen to messages from one client (with the IP address 10.0.0.1) on UDP port 999:
  
  source s_udp_10 { udp(ip(10.0.0.1) port(999)); };

3. **Define filters using this statement:**

filter <identifier> { expression; };

where:

• **identifier**—Has to uniquely identify this given filter.

• **expression**—a Boolean expression using internal functions, which has to evaluate to true for the message to pass.

The following internal functions are available:

**Table 5-3:** “Filters” parameters (Syslog-ng configuration)

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>facility</td>
<td>Selects messages based on their facility code.</td>
</tr>
<tr>
<td>(&lt;facility code&gt;)</td>
<td></td>
</tr>
<tr>
<td>level (&lt;level code&gt;) or priority (&lt;level code&gt;)</td>
<td>Selects messages based on their priority.</td>
</tr>
<tr>
<td>program (&lt;string&gt;)</td>
<td>Tries to match the &lt;string&gt; to the program name field of the log message.</td>
</tr>
<tr>
<td>host (&lt;string&gt;)</td>
<td>Tries to match the &lt;string&gt; to the hostname field of the log message.</td>
</tr>
<tr>
<td>match (&lt;string&gt;)</td>
<td>Tries to match the &lt;string&gt; to the message itself.</td>
</tr>
</tbody>
</table>
Examples of Defined Filters

a. Filtering by facility:

    filter f_facility { facility(<facility name>); };

    Examples:

    filter f_daemon { facility(daemon); };
    filter f_kern { facility(kern); };
    filter f_debug { not facility(auth, authpriv, news, mail); };

b. Filtering 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); }

c. Filtering by matching one string in the received message:

    filter f_match { match('string'); };

    To filter by matching the string “named”:

    filter f_named { match('named'); }

d. Filtering ALARM messages:

    NOTE: The following example statements should each be entered on one line with no line breaks.

    filter f_alarm { facility(local[0+<conf.DB_facility>]) and level(info) and match('ALARM') and match('<your string>'); };

    To filter ALARM messages with the string “kernel panic”:

    filter f_kpanic { facility(local[0+<conf.DB_facility>]) and level(info) and match('ALARM') and match('kernel panic'); };

    To filter ALARM messages with the string “root login”: 
filter f_root { facility(local[0+<conf.DB_facility>]) and level(info) and match('ALARM') and match('root login'); };

e. Eliminating SSHD debug messages:
filter f_sshd_debug { not program('sshd') or not level(debug); };

f. Filtering the syslog_buffering:
filter f_syslog_buf { facility(local[0+<conf.DB_facility>]) and level(notice); };

4. Define actions using this statement:

NOTE: The following example statement should be entered on one line with no line breaks.

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 parameters are required; others are optional.

The following destination drivers are available:

Table 5-4: “Destination Drivers” parameters (Syslog-ng configuration)

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>file (filename[options])</td>
<td>One of the most important destination drivers in syslog-ng. This driver allows you to output log messages to the named file. The destination filename may include macros which are expanded when the message is written. (To include macros, append a dollar sign ($) before the macro name.) Since the state of each created file must be tracked by syslog-ng, memory is consumed for each file. If no new messages are written to a file within 60 seconds (controlled by the time_reap global option), it is closed, and its state is freed.</td>
</tr>
</tbody>
</table>
Available macros in filename expansion:

- **HOST**—The name of the source host where the message was generated.
- **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 that sent the message.
- **YEAR, MONTH, DAY, HOUR, MIN, SEC**—The year, month, day, hour, minute, and second 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 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.

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.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>file (filename[options])</td>
<td>Available macros in filename expansion:</td>
</tr>
<tr>
<td>(continued)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• HOST—The name of the source host where the message was generated.</td>
</tr>
<tr>
<td></td>
<td>• FACILITY—The name of the facility the message is tagged as coming from.</td>
</tr>
<tr>
<td></td>
<td>• PRIORITY or LEVEL—The priority of the message.</td>
</tr>
<tr>
<td></td>
<td>• PROGRAM—The name of the program that sent the message.</td>
</tr>
<tr>
<td></td>
<td>• YEAR, MONTH, DAY, HOUR, MIN, SEC—The year, month, day, hour, minute, and</td>
</tr>
<tr>
<td></td>
<td>second the message was sent.</td>
</tr>
<tr>
<td></td>
<td>• TAG—Equals FACILITY/LEVEL.</td>
</tr>
<tr>
<td></td>
<td>• FULLHOST—The name of the source host and the source-driver:</td>
</tr>
<tr>
<td></td>
<td>&lt;source-driver&gt;@&lt;hostname&gt;</td>
</tr>
<tr>
<td></td>
<td>• MSG or MESSAGE—The message received.</td>
</tr>
<tr>
<td></td>
<td>• FULLDATE—The date the message was sent.</td>
</tr>
<tr>
<td>pipe (filename[options])</td>
<td></td>
</tr>
</tbody>
</table>

Table 5-4: “Destination Drivers” parameters (Syslog-ng configuration)
Table 5-4: “Destination Drivers” parameters (Syslog-ng configuration)

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>unix-stream(filename)</td>
<td>This driver sends messages to a UNIX socket in either SOCKET_STREAM or SOCK_DGRAM mode.</td>
</tr>
<tr>
<td>and unix-dgram(filename)</td>
<td></td>
</tr>
<tr>
<td>udp(&quot;&lt;ip address&gt;&quot;</td>
<td>This driver sends messages to another host (ip address/port) using either UDP or TCP protocol.</td>
</tr>
<tr>
<td>port(number);)</td>
<td></td>
</tr>
<tr>
<td>and tcp(&quot;&lt;ip address&gt;&quot;</td>
<td></td>
</tr>
<tr>
<td>port(number);)</td>
<td></td>
</tr>
<tr>
<td>program(&lt;program name</td>
<td>This driver fork()'s executes the given program with the arguments and sends messages down to the stdin of the child.</td>
</tr>
<tr>
<td>and arguments&gt;)</td>
<td></td>
</tr>
<tr>
<td>usertty(&lt;username&gt;)</td>
<td>This driver writes messages to the terminal of a logged-in username.</td>
</tr>
</tbody>
</table>

Examples of defined actions

a. 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 port: 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.

The following example sends an e-mail to z@none.com (SMTP server’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 CPS, and the message that was received from the source.

```plaintext
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 5.4: Send e-mail example**

b. To send a message to a pager server (SMS server):

```plaintext
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>
The following example sends a page to the phone number 123 (Pager server at 10.0.0.1) with a message carrying the current date, the hostname of this CPS, 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 5.5: To send a pager phone example

c.  To send an snmptrap:

```c

```

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.itable.ifentry.ifdescr.1
- s : the type of the next field (it is a string)
- "\"<message - max. size 250 characters">"

The following example sends a Link Down trap to the SNMP server at 10.0.0.1 with a message carrying the current date, the hostname of this CPS, 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 "$\" \" 2 0 "$\" "
                  .1.3.6.1.2.1.2.1.2.1 s "$FULLDATE $HOST $MSG" "));
};
```

File Description 5.6: Sending a link down trap
d. To write in a file:

```c
destination d_file { file(<filename>); }
```

To send a message to the console:

```c
destination d_console { file("/dev/ttyS0"); }
```

*File Description 5.7: Sending messages to console*

To write a message in the `/var/log/messages` file:

```c
destination d_message { file("/var/log/messages"); }
```

*File Description 5.8: Writing messages to file*

e. To write messages to the session of a logged-in user:

```c
destination d_user { usertty(<username>); }
```

The following example sends a message to all sessions with `root` as the logged-in user:

```c
destination d_userroot { usertty("root"); }
```

*File Description 5.9: Sending messages to logged user*

f. To send a message to a remote syslogd server:

```c
destination d_udp { udp(<remote IP address> port(514)); }
```

The following example sends syslog messages to a remote syslogd server located at 10.0.0.1:

```c
destination d_udpl { udp("10.0.0.1" port(514)); }
```

*File Description 5.10: Sending syslogs to a remote server*
5. Connect all sources, filters, and actions.

To connect the sources, filters, and actions, use the following statement. (An action is any message coming from one of the listed sources. A match for each of the filters is sent to the listed destinations.)

\[
\text{log} \{ \text{source}(S_1); \text{source}(S_2); \ldots \text{filter}(F_1);\text{filter}(F_2); \ldots \text{destination}(D_1); \text{destination}(D_2); \ldots \};
\]

where:

- \(S_x\) — Identifier of the sources defined in step 2.
- \(F_x\) — Identifier of the filters defined in step 3.
- \(D_x\) — Identifier of the actions/destinations defined in step 4.

Examples connecting sources, filters, and actions

a. To send all messages received from local syslog clients to the console:

\[
\text{log} \{ \text{source}(\text{sysl}); \text{destination}(\text{d_console}); \};
\]

b. To send only messages with level alert and received from local syslog clients to all logged root user:

\[
\text{log} \{ \text{source}(\text{sysl}); \text{filter}(\text{f_alert}); \text{destination}(\text{d_userroot}); \};
\]

c. To write all messages with levels info, notice, and warning and received from syslog clients (local and remote) to \text{/var/log/messages} file:

\[
\text{log} \{ \text{source}(\text{sysl}); \text{source}(\text{s_udp}); \text{filter}(\text{f_messages}); \text{destination}(\text{d_messages}); \};
\]

d. To send e-mail if a message received from a local syslog client has the string “kernel panic”:

\[
\text{log} \{ \text{source}(\text{sysl}); \text{filter}(\text{f_kpanic}); \text{destination}(\text{d_mail1}); \};
\]

e. To send e-mail and a page if a message received from a local syslog client has the string “root login”:

\[
\text{log} \{ \text{source}(\text{sysl}); \text{filter}(\text{f_root}); \text{destination}(\text{d_mail1}); \text{destination}(\text{d_pager}); \};
\]

f. To send messages with a facility kernel and received from syslog clients (local and remote) to remote syslogd:

\[
\text{log} \{ \text{source}(\text{sysl}); \text{source}(\text{s_udp}); \text{filter}(\text{f_kern}); \text{destination}(\text{d-udp1}); \};
\]
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).

VI Method

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 5.11: portslave.conf necessary configuration
   ```

2. Using vi, add the following lines to the `/etc/syslog-ng/syslog-ng.conf` 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 5.12: portslave.conf necessary configuration
   ```

Syslog-ng Configuration to Use with Multiple Remote Syslog Servers

This configuration example is used with multiple remote syslog servers.

VI Method

1. Configure `pslave.conf` parameters. In the `pslave.conf` file the facility parameter is configured as:

   ```
   conf.facility 1
   
   File Description 5.13: portslave.conf “facility” configuration
   ```
2. Add lines to the `/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 5.14: syslog-ng.conf configuration*

**CLI method—Syslog**

To configure Syslog:

1. Open the CLI interface by issuing the command:
   
   ```
   # CLI
   ```

2. Configure the syslog facility number.

   This is the facility number for the messages. The remote Syslog server filters receive messages according to this parameter.

   `cli>config network syslog facility <local0-local7>

   Possible values for the syslog facility number range from local0 to local7

3. Set up the IP address of the server where syslog messages will be sent.

   For a server with an IP address of 200.200.200.1:

   `cli>config network syslog add server 200.200.200.1`
Repeat this step as many times as necessary, depending on the number of remote servers you want to add.

4. Activate the configuration.
   
   cli> config runconfig

5. Save the configuration.
   
   cli> config savetoflash

6. Exit the CLI mode.
   
   To exit the CLI mode and return to CPS ’s shell, issue the command:
   
   cli> quit

5.5 How Syslog Messages Are Generated

The CPS can generate syslog messages, which enable system administrators to monitor changes in the box. When certain actions/conditions are met through the Web interface as well as through CLI or commands which users enter from a shell prompt, the system generates and sends messages to the syslog-ng file.

The messages use the following format:

- **Level**—the syslog level used
- **Tag**—a fixed string used by the user to create filters
- **Text**—the text that contains the condition or action

**Generated Syslog Messages**

The CPS generates syslog messages as a result of specific actions or conditions:

<table>
<thead>
<tr>
<th>Table 5-5: CPS Syslog Messages Format</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Level</strong></td>
</tr>
<tr>
<td>---------------</td>
</tr>
<tr>
<td>info</td>
</tr>
<tr>
<td>info</td>
</tr>
</tbody>
</table>
### Table 5-5: CPS Syslog Messages Format

<table>
<thead>
<tr>
<th>Level</th>
<th>Tag</th>
<th>Text</th>
</tr>
</thead>
<tbody>
<tr>
<td>info</td>
<td>AUTH</td>
<td>User [xyz] for session [abc] logged out</td>
</tr>
<tr>
<td>info</td>
<td>AUTH</td>
<td>Cancel new admin [abc] login</td>
</tr>
<tr>
<td>info</td>
<td>AUTH</td>
<td>Session [%d] timed out</td>
</tr>
<tr>
<td>info</td>
<td>CONFIG</td>
<td>Configuration saved to flash</td>
</tr>
<tr>
<td>info</td>
<td>CONFIG</td>
<td>New configuration activated</td>
</tr>
<tr>
<td>info</td>
<td>CONFIG</td>
<td>Password changed for user [xyz] by user [abc]</td>
</tr>
<tr>
<td>info</td>
<td>CONFIG</td>
<td>User [xyz] added by user [abc]</td>
</tr>
<tr>
<td>info</td>
<td>CONFIG</td>
<td>User [xyz] deleted by user [abc]</td>
</tr>
<tr>
<td>info</td>
<td>CONFIG</td>
<td>Network daemon [daemon name] stopped</td>
</tr>
<tr>
<td>info</td>
<td>APPLICATION</td>
<td>User [abc] connected to port [x] (ttySx) via socket server</td>
</tr>
<tr>
<td>info</td>
<td>APPLICATION</td>
<td>User [abc] connected to port [x] (ttySx) via socket ssh</td>
</tr>
<tr>
<td>info</td>
<td>APPLICATION</td>
<td>User [abc] connected to port [x] (ttySx) via socket ssh</td>
</tr>
<tr>
<td>alert</td>
<td>[PMD]-Serial Port $p$</td>
<td>Outlet X has been turned OFF by user &lt;username&gt;</td>
</tr>
<tr>
<td>alert</td>
<td>[PMD]-Serial Port $p$</td>
<td>Outlet X has been turned ON by user &lt;username&gt;</td>
</tr>
<tr>
<td>alert</td>
<td>[PMD]-Serial Port $p$</td>
<td>OVER CURRENT on rPDU #X (current: &lt;current detected&gt; threshold:&lt;threshold configured&gt;)</td>
</tr>
<tr>
<td>alert</td>
<td>[PMD]-Serial Port $p$</td>
<td>One or more rPDUs were removed from the chain. This chain has now X rPDUs and Y outlets</td>
</tr>
<tr>
<td>alert</td>
<td>AUTH</td>
<td>User [xyz] login failed</td>
</tr>
</tbody>
</table>
NOTE: To prevent generation of Power Management Device (PMD) syslog messages, edit the file `/etc/pmd.sh`. Change the parameter DPARM from "" to "-s", then run the commands “saveconf” and “daemon restart PMD.”

You can use the information provided in the table above to create filters and generate alarms about events that happen in the CPS. See “DCD ON/OFF Syslog Messages” on page 155 for more information.

### Table 5-5: CPS Syslog Messages Format

<table>
<thead>
<tr>
<th>Level</th>
<th>Tag</th>
<th>Text</th>
</tr>
</thead>
<tbody>
<tr>
<td>alert</td>
<td>AUTH</td>
<td>User [%s] login failed. Another admin session exists.</td>
</tr>
<tr>
<td>alert</td>
<td>AUTH</td>
<td>Previous admin session terminated by new admin [abc] login</td>
</tr>
<tr>
<td>alert</td>
<td>CONFIG</td>
<td>Network daemon [daemon name] started</td>
</tr>
<tr>
<td>alert</td>
<td>SYSTEM</td>
<td>System rebooted by admin [xyz] [hostname] [ip address]</td>
</tr>
<tr>
<td>alert</td>
<td>PORT DCD</td>
<td>Port &lt;serial port number&gt; DCD went high</td>
</tr>
<tr>
<td>alert</td>
<td>PORT DCD</td>
<td>Port &lt;serial port number&gt; DCD went low</td>
</tr>
<tr>
<td>debug</td>
<td>AUTH</td>
<td>User [%s] login failed. Group 'admin' does not exist</td>
</tr>
<tr>
<td>debug</td>
<td>AUTH</td>
<td>User [%s] login failed. Maximum number of connected users reached</td>
</tr>
<tr>
<td>notice</td>
<td>[PMD]-Serial</td>
<td>PMD has started on this port. The chain has X rPDUs and W outlets</td>
</tr>
<tr>
<td>notice</td>
<td>DAEMON</td>
<td>Web server started on port xx</td>
</tr>
<tr>
<td>notice</td>
<td>DAEMON</td>
<td>Web server stopped</td>
</tr>
<tr>
<td>notice</td>
<td>DAEMON</td>
<td>Caught SIGINT: Web server stopped</td>
</tr>
<tr>
<td>warning</td>
<td>[PMD]-Serial</td>
<td>Current is now back to normal on rPDU #X (current: &lt;current detected&gt; threshold:&lt;threshold configured&gt;)</td>
</tr>
<tr>
<td></td>
<td>Port p</td>
<td></td>
</tr>
</tbody>
</table>
DCD ON/OFF Syslog Messages

The CPS can generate an alert when a serial console cable is removed from the CPS or server/network equipment attached to the CPS is powered down. Also, if a modem is part of the configuration, detect whether the modem is still receiving power and is active.

The DCD signal will be monitored and a syslog message will be generated whenever the state of the signal changes. The syslog message can be handled by syslog-ng to generate an event notification.

To enable syslog message generation in the CPS

1. Open the /etc/portslave/pslave.conf file.
   
   # vi /etc/portslave/pslave.conf

2. Set the all.dcd or sXX.dcd parameter to 1 in the /etc/portslave/pslave.conf file. In this example, XX is the desired port number.
   
   all.dcd 1
   
   or
   
   sXX.dcd 1

3. Configure the syslog-ng.conf file to monitor DCD status.
   
   For an example, see “Generating messages and sending them to the console if the DCD signal changes its state.” on page 156.

4. Save the configuration.
   
   # saveconf

The source for all generated Syslog messages is src_dev_log. The only exceptions are Syslog messages generated when DCD goes on or off; that source is s_kernel. To create filters or to trigger alarms, see Table 5-5, “CPS Syslog Messages Format,” on page 152.
Examples

To configure the examples given below, edit the /etc/syslog-ng/syslog-ng.conf file and add the presented lines.

Generating Syslog messages to be sent to the console when the user root tries to connect together with an already logged root user.

```plaintext
filter f_info { level(info); };
filter f_named { match("AUTH"); };
destination console { usertty("root"); };
log { source(src_dev_log); filter(f_info, f_named); destination(console); };
```

Generating messages and sending them to the console when any user login attempt fails.

```plaintext
filter f_info { level(alert); };
filter f_named { match("AUTH"); };
destination console { usertty("root"); };
log { source(src_dev_log); filter(f_info, f_named); destination(console); };
```

Generating messages and sending them to the console if the DCD signal changes its state.

```plaintext
filter f_dcdchg { level(alert) and match("PORT DCD") };
destination console { usertty("root"); };
log { source(s_kernel); filter(f_dcdchg); destination(console); };
```

5.6 Generating Alarms (Syslog-ng)

Alarm generation helps the administrator manage the servers. It filters the messages received by the serial port (the server’s console port) based on the contents of the messages, then performs an action (e.g., the server sends an e-mail or pager message). To configure this feature, you must configure filters and actions in the syslog-ng.conf file. (For more information about syslog-ng, see “Syslog-ng” on page 137.)

How To Configure

Alarm generation is strictly related to the syslog-ng configuration. Before configuring this feature, read the section “Syslog-ng” on page 137.

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 must be set to a non-zero value to activate alarm generation.

Syslog-ng reads from sources (files, TCP/UDP connections, syslogd clients), filters the messages, and takes an action (writes in files or sends SNMP traps, pager messages, e-mail, or syslogs).

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 all of the filters, 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, see “Syslog-ng” on page 137.

**VI method—configuration for use with the alarm feature**

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 follows:

   ```
   all.alarm 1
   conf.DB_facility  2
   ```

2. Configure the `/etc/syslog-ng/syslog-ng.conf` file.

   By default, all commands are present (commented) in the original `syslog-ng.conf` file. Choose the example that best fits your application.

   - **Example 1**—Send all messages received from local syslog clients to console:
Insert the following lines at the END of the `syslog-ng.conf` file. Keep all other commands commented.

```plaintext
source sysl {unix-stream("/dev/log");};
destination d_console { file("/dev/ttyS0");};
log { source(sysl); destination(d_console);};
```

*File Description 5.15: part of the /etc/syslog-ng/syslog-ng.conf file*

**Example 2**—Send only messages with the level *alert* and received from local syslog clients to all logged root users.

Insert the following lines at the END of the `syslog-ng.conf` file. Keep all other commands commented.

```plaintext
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 5.16: 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 remote) to `/var/log/messages` file:

Insert the following lines at the END of the `syslog-ng.conf` file. Keep all other commands commented.

```plaintext
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_message);};
```

*File Description 5.17: part of the /etc/syslog-ng/syslog-ng.conf file*

**Example 4**—Send e-mail if a message received from a local syslog client has the string "kernel panic".
Insert the following lines at the END of the `syslog-ng.conf` file. Keep all other commands commented.

```plaintext
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 5.18: part of the `/etc/syslog-ng/syslog-ng.conf` file*

**Example 5**—Send e-mail and pager message if a message received from a local syslog client has the string "root login".

Insert the following lines at the END of the `syslog-ng.conf` file. Keep all other commands commented.

```plaintext
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 $MSG" -h 10.0.0.2")
    );
}
destination dPager {
    pipe("/dev/cyc_alarm"
    template("sendsms -d 123 -m "$FULLDATE $HOST $MSG" 10.0.0.1")
    );
}
log { source(sysl); filter(f_root); destination(d_mail1); destination(dPager); };
```

*File Description 5.19: part of the `/etc/syslog-ng/syslog-ng.conf` file*

**Example 6**—Send messages that have a facility kernel and are sent by syslog clients (local and remote) to remote syslogd.
Insert the following lines at the END of the `syslog-ng.conf` file. Keep all other commands commented.

```plaintext
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 5.20: part of the /etc/syslog-ng/syslog-ng.conf file*

3. **Activate changes.**

To activate the changes, run the following commands in the presented order:

```
# runconf
# killall syslog-ng
# syslog-ng
```

The first command activates 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.

4. **Save the changes to the flash memory.**

To save the changes, run the command:

```
# saveconf
```

**CLI method—alarm notification**

The CLI interface allows the configuration of alarm notifications when an event is generated in any CPS port. You cannot use the CLI interface to generate alarms for the CPS.

In the CLI interface, all options are under the following menu:

```
cli> config administration notifications
```

Options include:

- **addemail**—Sends a message to the configured e-mail address if the defined string appears.
- **addpager**—Sends a message to the configured pager if the defined string appears.
- **addsnmptrap**—Sends an SNMP trap to the configured server if the defined string appears.
• **alarm**—Activates or deactivates the alarm feature. If this option is disabled, syslog messages will not be generated when there is incoming data from the ports.

• **delete**—Deletes any previously configured string.

• **edit**—Edit any previously configured string.

**Example of alarm notification (CLI method):** The following example configures the CPS to send an e-mail every time the user `root` logs into the server connected to a specified port, and configures the trigger string as “root login”.

**NOTE:** The server connected to the CPS must be configured to send Syslog messages.

1. Open the CLI interface by issuing the command:
   
   # CLI

2. Enable alarm notification.
   
   This option must be enabled, or messages received in the ports will be ignored by Syslog-ng.
   
   cli> config administration notifications alarm yes

3. Add the trigger string.
   
   Define the string that will trigger the e-mail notification (“root login” in this example).
   
   cli> config administration notifications addemail "root login"

4. Configure the necessary parameters.
   
   To send an e-mail, configure the following parameters:
   
   add Email>body "Testing configuration"
   add Email>from CPS
   add Email>to someone@yourdomain.com
   add Email>smtpserver 200.200.200.2
   add Email>smtpport 25
   add Email>subject "Testing Config"

   The above commands configure the From/To fields, the SMTP server/port, and the subject/body of the e-mail message.
5. Activate the configuration.
   cli> config runconfig

6. Save the configuration.
   cli> config savetoflash

7. Exit the CLI mode.
   To exit the CLI mode and return to CPS’s shell, issue the command:
   cli> quit

5.7 Terminal Appearance

You can change the format of the login prompt and banner that are issued when a connection is made to the system. Prompt and banner appearance can be port-specific.

VI Method—involved parameters and passed values

Terminal Appearance involves the following parameters in the /etc/portslave/pslave.conf file:

<p>| Table 5-6: pslave.conf parameters for Terminal Appearance configuration |</p>
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>all.prompt</td>
<td>Defines the format of the login prompt. You can use expansion characters can be used for this parameter. Example: %h login:</td>
</tr>
</tbody>
</table>
| all.issue         | Determines the format of the login banner that is issued when a connection is made to the CPS. 

\n represents a new line and \r represents a carriage return. Expansion characters can be used.

Example:

\r\n
Welcome to terminal server %h port S%p \n\n\r\n
| all.lf_suppress   | Activates line feed suppression. When configured as 0, line feed suppression will not be performed. When 1, extra line feeds will be suppressed. |
To configure parameters for a specific serial port:

1. Open the CLI interface by issuing the command:
   
   ```
   # CLI
   ```

2. Configure the banner.

   The command below configures “testing banner” as the default banner for all ports.

   ```
   cli> config physicalports all other banner “testing banner”
   ```

   Configure different banners for different ports by changing the \textit{ALL} parameter to the port number or port range. E.g., 1 or 1–3 (from port 1 to 3).

3. Activate the configuration.

   ```
   cli> config runconfig
   ```

4. Save the configuration.

   ```
   cli> config savetoflash
   ```

5. Exit the CLI mode.

---

### Table 5-6: \textit{pslave.conf} parameters for Terminal Appearance configuration

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>all.auto_answer_input</td>
<td>This parameter is used in conjunction with the next parameter, \textit{auto_answer_output}. If this parameter is configured and if no session is established to the port, this parameter will constantly be compared to the string of bytes received from the remote server. If a match is found, the string configured in \textit{auto_answer_output} is sent back to the server. To represent the ESC character as part of this string, use the control character, \textasciicircum[.</td>
</tr>
<tr>
<td>all.auto_answer_output</td>
<td>This parameter is used in conjunction with the preceding parameter, \textit{auto_answer_input}. If this parameter is configured, and if no session is established to the port, this parameter is sent back to the server when there is a match between the incoming data and \textit{auto_answer_input}. To represent the ESC character as part of this string, use the control character, \textasciicircum[.</td>
</tr>
</tbody>
</table>
To exit the CLI mode and return to CPS’s shell, issue the command:

```shell
cli> quit
```

## 5.8 Centralized Management

The CPS allows centralized management through the use of a Master `pslave.conf` file. Administrators can use this procedure to configure multiple CPSs. Each unit has a simplified `pslave.conf` file where a Master include file is cited. This common configuration file contains information for all units, divided in separate sections, and is stored on one central server. In the following figure, this file is `/etc/portslave/TScommon.conf`. It must be downloaded to each CPS.

**NOTE:** Centralized management can describe one big configuration file (the common file) that is placed in a management host. This same file would be downloaded into all CPS boxes (each CPS would include a small config file and the large 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 a central point (master), using the common file to simplify maintenance of 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 configuration file does NOT apply to clustering. However, common config files can be used in an integrated manner.

![Figure 5.21 - Example of Centralized Management](image-url)
VI Method—in involved parameters and passed values

For the preceding example (figure 4.21), the abbreviated /etc/portslave/pslave.conf and /etc/hostname files in each unit are as follows:

Unit 1 configuration:

For the /etc/hostname file in unit 1:

```
unit1
```

*File Description 5.22: Unit 1 /etc/hostname file*

For the /etc/portslave/pslave.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 5.23: Unit 1 /etc/portslave/pslave.conf file configuration*

Unit 2 configuration:

For the /etc/hostname file in unit 2:

```
unit2
```

*File Description 5.24: Unit 2 /etc/hostname file*

For the /etc/portslave/pslave.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 5.25: Unit 2 /etc/portslave/pslave.conf file configuration*
**Unit 3 configuration:**

For the `/etc/hostname` file in unit 3:

```
unit3
```

*File Description 5.26: 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 5.27: Unit 3 /etc/portslave/portslave.conf file configuration*
For the example in figure 4.21, the common include file (located in the server) 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 5.28: Common /etc/portslave/pslave.conf file

When this file is included, unit1 reads only the information between `conf.host_config unit1` and `conf.host_config unit2`. Unit2 reads only the information between `conf.host_config unit2` and `conf.host_config unit3` and unit3 reads information after `conf.host_config unit3` and before `conf.host_config end`.

Steps for Using Centralized Configuration

1. Create and save the `/etc/portslave/pslave.conf` and `/etc/hostname` files in each CPS.
2. Create, save, and download the common configuration.
Create and save the common configuration file on the server, then download it (i.e. SCP) to each unit. Make sure to put it in the directory set in the pslave.conf file (/etc/portslave in the example).

3. Execute the command `runconf` on each unit.

4. Test each unit.
   
   If everything works, add the line `/etc/portslave/TScommon.conf` to the `/etc/config_files` file.

5. Save the file and close it.

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 CPS > + N(+)` is defined in the same way as the serial port.

---

### 5.9 Date, Time, and Timezone

To adjust the date and time, use the `date` command. To configure the timezone, use the CLI utility or the `set_timezone` script.

---

**NOTE:** Setting the system timezone creates a new file called `/etc/ localtime`, which erases the `/etc/TIMEZONE` file.

---

### Date and Time

The `date` command prints or sets the system date and time.

```
  date MMDDhhmmYYYY
```

where:

- MM = month
- DD = day
- hh = hour
- mm = minute
- YYYY = year
For example:

date 101014452002

displays:

Thu Oct 10 14:45:00 <timezone> 2002

NOTE: The time zone is configured using the CLI utility or set_timezone script.

CLI method—date and time

Configuring date and time using CLI automatically disables any previously configured NTP server.

To configure date/time using the CLI:

1. Open the CLI interface by issuing the command:
   
   # CLI

2. Configuring the date.
   
   The date format must follow this syntax: \texttt{mm/dd/yyyy}, where:
   
   • \texttt{mm}—month
   • \texttt{dd}—day
   • \texttt{yyyy}—year
   
   The following example configures the date, December 31, 2006.
   
   cli> config administration date/time date 12/31/2006

3. Setting the time.
   
   The time format must follow this syntax: \texttt{hh:mm:ss}, where:
   
   • \texttt{hh}—hour
   • \texttt{mm}—minutes
   • \texttt{ss}—seconds
   
   The following example configures the time, nine o’clock AM:
   
   cli> config administration date/time time 09:00:00
4. Activate the configuration.
   `cli> config runconfig`

5. Save the configuration.
   `cli> config savetoflash`

6. Exit the CLI mode.
   To exit the CLI mode and return to CPS’s shell, issue the command:
   `cli> quit`

**Setting Local Timezone**

You can set the time to your local timezone using the `set_timezone` command or the CLI utility. The system uses GMT (Greenwich Mean Time) as a reference point.

**Configuring the timezone using the set_timezone command**

1. From a shell within the CPS box, run `set_timezone` as the user root by entering the following command:
   ```
   #set_timezone
   ```
The following options appear:

<table>
<thead>
<tr>
<th></th>
<th>Time Zone</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>GMT</td>
</tr>
<tr>
<td>1</td>
<td>1h West GMT</td>
</tr>
<tr>
<td>2</td>
<td>10h West GMT</td>
</tr>
<tr>
<td>3</td>
<td>11h West GMT</td>
</tr>
<tr>
<td>4</td>
<td>12h West GMT</td>
</tr>
<tr>
<td>5</td>
<td>2h West GMT</td>
</tr>
<tr>
<td>6</td>
<td>3h West GMT</td>
</tr>
<tr>
<td>7</td>
<td>4h West GMT</td>
</tr>
<tr>
<td>8</td>
<td>5h West GMT</td>
</tr>
<tr>
<td>9</td>
<td>6h West GMT</td>
</tr>
<tr>
<td>10</td>
<td>7h West GMT</td>
</tr>
<tr>
<td>11</td>
<td>8h West GMT</td>
</tr>
<tr>
<td>12</td>
<td>9h West GMT</td>
</tr>
<tr>
<td>13</td>
<td>1h East GMT</td>
</tr>
<tr>
<td>14</td>
<td>10h East GMT</td>
</tr>
<tr>
<td>15</td>
<td>11h East GMT</td>
</tr>
<tr>
<td>16</td>
<td>12h East GMT</td>
</tr>
<tr>
<td>17</td>
<td>13h East GMT</td>
</tr>
<tr>
<td>18</td>
<td>14h East GMT</td>
</tr>
<tr>
<td>19</td>
<td>2h East GMT</td>
</tr>
<tr>
<td>20</td>
<td>3h East GMT</td>
</tr>
<tr>
<td>21</td>
<td>4h East GMT</td>
</tr>
<tr>
<td>22</td>
<td>5h East GMT</td>
</tr>
<tr>
<td>23</td>
<td>6h East GMT</td>
</tr>
<tr>
<td>24</td>
<td>7h East GMT</td>
</tr>
<tr>
<td>25</td>
<td>8h East GMT</td>
</tr>
<tr>
<td>26</td>
<td>9h East GMT</td>
</tr>
</tbody>
</table>

Type your option:
2. Type the number corresponding to your Local GMT and press <Enter>.
   A message verifies your selection. For example, if you choose 8, the system displays the following message:
   
   Your choice was: GMT+4

3. Run `saveconf` to save your changes.

---

**NOTE:** Setting your system timezone creates a new file called `/etc/localtime`, which erases the old `/etc/TIMEZONE`.

---

### Configuring the local timezone using CLI

You can configure your local timezone using the CLI utility.

1. Enter the following command to enter the CLI mode.
   
   `#CLI`

2. At the cli> prompt, enter the following command.
   
   `#cli>config>administration>timezone <value>`

   **NOTE:** You can enter the value if known, otherwise, press tab to see the list of possible values.

   `#cli>config>administration>timezone <Press tab to see list of possible values>`

   The following possible values display:

<table>
<thead>
<tr>
<th>10h_East_GMT</th>
<th>13h_East_GMT</th>
<th>3h_East_GMT</th>
<th>6h_East_GMT</th>
<th>9h_East_GMT</th>
</tr>
</thead>
<tbody>
<tr>
<td>10h_West_GMT</td>
<td>14h_East_GMT</td>
<td>3h_West_GMT</td>
<td>6h_West_GMT</td>
<td>9h_West_GMT</td>
</tr>
<tr>
<td>11h_East_GMT</td>
<td>1h_East_GMT</td>
<td>4h_East_GMT</td>
<td>7h_East_GMT</td>
<td>GMT</td>
</tr>
<tr>
<td>11h_West_GMT</td>
<td>1h_West_GMT</td>
<td>4h_West_GMT</td>
<td>7h_West_GMT</td>
<td></td>
</tr>
<tr>
<td>12h_East_GMT</td>
<td>2h_East_GMT</td>
<td>5h_East_GMT</td>
<td>8h_East_GMT</td>
<td></td>
</tr>
<tr>
<td>12h_West_GMT</td>
<td>2h_West_GMT</td>
<td>5h_West_GMT</td>
<td>8h_West_GMT</td>
<td></td>
</tr>
</tbody>
</table>

3. Select your GMT zone and enter it at the prompt. For example:
   
   `#cli>config>administration>timezone 2h_West_GMT`
4. Activate the configuration.
   `cli> config runconfig`

5. Save the configuration.
   `cli> config savetoflash`

6. Exit the CLI mode.
   `cli> quit`

5.10 NTP (Network Time Protocol)

The ntpclient is a Network Time Protocol client for UNIX- and Linux-based computers. (See RFC-1305 for more information about the Network Time Protocol.) In order for the CPS to work as an NTP client, the IP address of the NTP server must be set in the file `/etc/daemon.d/ntpclient.conf`. The program `/bin/daemon.sh` reads the configuration file `/etc/daemon.d/ntpclient.conf` and runs with the settings of this file.

VI mode configuration

The file `/etc/daemon.d/ntpclient.conf` has all of the configurable parameters. The parameters that are not presented in the table below should not be changed.

1. Edit the `/etc/daemon.d/ntpclient.conf` file and change the parameters according to the table below:

   **Table 5-7: `/etc/daemon.d/ntpclient.conf` parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENABLE</td>
<td>This parameter enables the NTP client. The default setting is NO. To enable this parameter, choose “YES”.</td>
</tr>
<tr>
<td>NTPSERVER</td>
<td>NTP server IP address.</td>
</tr>
<tr>
<td>NTPINTERVAL</td>
<td>Time in seconds to ask server.</td>
</tr>
<tr>
<td>NTPCOUNT</td>
<td>Specifies how many times the server will be asked. 0 means forever.</td>
</tr>
</tbody>
</table>
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
```

**CLI method—NTP server**

To configure an NTP server using the CLI, follow the steps below:

1. Open the CLI interface by issuing the command:
   ```
   # CLI
   ```

2. Set the IP address of the NTP server (in the following example, `xxx.xxx.xxx.xxx` is the IP address of the NTP server).
   ```
   cli> config administration ntp xxx.xxx.xxx.xxx
   ```

   **NOTE:** To deactivate the NTP service, configure the date by issuing the command:
   ```
   cli> config administration date/time date <mm/dd/yyyy>
   ```

3. Activate the configuration.
   ```
   cli> config runconfig
   ```

### Table 5-7: `/etc/daemon.d/ntpclient.conf` parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NTP_OPT</td>
<td>Other NTP parameters. The possible values for this parameter are listed below:</td>
</tr>
<tr>
<td></td>
<td>• <code>-d</code> -&gt; Print diagnostics</td>
</tr>
<tr>
<td></td>
<td>• <code>-h hostname</code> -&gt; NTP server host (mandatory).</td>
</tr>
<tr>
<td></td>
<td>• <code>-l</code> -&gt; Attempt to lock local clock to server using adjtimex(2).</td>
</tr>
<tr>
<td></td>
<td>• <code>-p port</code> -&gt; Local NTP client UDP port.</td>
</tr>
<tr>
<td></td>
<td>• <code>-r</code> -&gt; Replay analysis code based on stdin.</td>
</tr>
<tr>
<td></td>
<td>• <code>-s</code> -&gt; Clock set (if count is not defined this sets count to 1).</td>
</tr>
</tbody>
</table>
4. Save the configuration.

   cli> config savetoflash

5. Exit the CLI mode.

   To exit the CLI mode and return to CPS’s shell, type the following command:

   cli> quit

5.11 Session Sniffing

When multiple sessions are allowed for one port, the behavior of the CPS will be as follows:

1. The first user to connect to the port will open a common session.

2. While the first user is connected, only Admin users are allowed to connect to that port. The CPS 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 you select 1 - *Initiate a regular session*, you will share that serial port with the users that were previously connected. You will read everything that is received by the serial port, and will also be able to write to it.

If you select 2 - *Initiate a sniff session*, you will start reading everything that is sent or received by the serial port, according to the parameter all.sniff_mode or sN.sniff_mode (the parameter setting can be in, out, or i/o).

If you select 3 - *Send messages to another user*, the CPS will send your messages to all the sessions, but not to the tty port. Everyone connected to that port will see all the “conversations”, 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 CPS.
To inform the CPS that the message is to be sent to the serial port or not, you will have to use the menu.

If you select 4 - Kill session(s), the CPS will show you a list of the pairs PID/username. Select one session by typing its PID, or type “all” to kill all the sessions. If you kill all of the regular sessions, your session initiates as a regular session automatically.

Option 5 - Quit will close the current session and the TCP connection.

Administrator users only: Typing all.escape_char or sN.escape_char from the sniff session or “send message mode” will make the CPS show the previous menu. The other regular sessions will not be allowed to return to the menu. If you kill all regular sessions using Option 4, your session initiates as a regular session automatically.

VI method—involved parameters and passed values

Session 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 required 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 same port (see the preceding parameter, admin_users) can see of the session of the first connected user (the main session): in shows data written to the port, out shows data received from the port, i/o shows both streams, and 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, socket_server, or socket_server_ssh. 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 key with the circumflex character: ^. This parameter is only valid when the port protocol is socket_server or socket_ssh. The default value is ^z.

- **all.multiple_sessions**—If this parameter is configured as no, only two users can connect to the same port simultaneously. If this parameter is configured as yes, more than two users can connect to the same serial port simultaneously. A “sniffer menu” will be presented to the user and they can choose either to open a sniff session; to open a read or write session; to cancel a connection; or to send a message to other users connected to
the same serial port. If this parameter is configured as \textit{RW Sessions}, only read or write sessions will be opened, and the sniffer menu will not be presented. If this parameter is configured as \textit{sniff session} only, a sniff session will be opened, and the sniffer menu will not be presented. Default value: \textit{no}.

\• \textit{all.multiusernotif}—Multiple User notification determines whether 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 do not go to the buffering file, and will be similar to the following examples:

\begin{verbatim}
WARNING: New user connected to this port.
Current number of users: x

or

WARNING: User disconnection from this port.
Current number of users: x
\end{verbatim}

In the examples, \textit{x} is the current number of connected users. The last user will know they are alone again when \textit{x} = 1.

\section*{CLI method—session sniffing}

To configure session-sniffing using the CLI interface:

1. Open the CLI interface by issuing the command:

\begin{verbatim}
# CLI
\end{verbatim}

2. Configure sniffing parameters.

The multiuser state configuration is under the menu:

\begin{verbatim}
cli>config physicalports <'all' or range/list[1-4]> multiuser
\end{verbatim}

Under this menu, you can configure the following parameters:

\begin{itemize}
  \item \textit{hotkey}—This parameter configures the escape character. The chosen character must be preceded by the circumflex character (^), which represents the CTRL key. E.g.: \texttt{^k}
  \item \textit{notifyusers}—To configure multiuser notification. Valid values: YES or NO.
  \item \textit{multisessions}—To configure multiple sessions. Valid values: yes, no, ro, and rw.
  \item \textit{privilegeusers}—This parameter determines which users can receive the sniff menu.
  \item \textit{sniffmode}—Determines what other users who are connected to the same port can see of the session of the first connected user (main session). Valid values: \textit{in}—shows data
written to the port; *out*—shows data received from the port; *in/out*—shows both streams; *off*—disables sniffing.

3. Activate the configuration.
   
   `cli> config runconfig`

4. Save the configuration.
   
   `cli> config savetoflash`

5. Exiting the CLI mode.
   
   To exit the CLI mode and return to CPS’s shell, type the following command:
   
   `cli> quit`

### 5.12 Saveconf and Restoreconf

The CPS has two utilities that are responsible for saving and restoring the configuration of the unit, *saveconf* and *restoreconf*.

**Saveconf utility**

The syntax is:

```
# saveconf [media <media parameters>]
```

*media* can be any of these options:

- `<none>`—Save the configuration to internal flash.
- `local <remote Path and filename>`—Save the configuration to the local file `<remote Path and filename>`.
- `ftp <remote Path and filename> <IP address of the FTP server> <username> <password>`—Save the configuration to the remote FTP server.
- `sd [default] [replace]`—Save the configuration to the PCMCIA storage device (Compact Flash or IDE).

The new media is *storagedevice* which has the two parameters, *default* and *replace*.

The *saveconf* utility creates one file in the storage device to save the *default* and *replace* flags. The file is named `/mnt/ide/proc/flash/storageOptions`, and it can contain the words *DEFAULT* and/or *REPLACE*. 
**Restoreconf utility**

The syntax is:

```bash
# restoreconf [media <media parameters>]
```

Where *media* can be any of these options:

- `<none>`—Read the configuration file from the PCMCIA storage device, and if the DEFAULT flag is set, use this file as the configuration default. If the REPLACE flag is set, copy this file to the internal flash of the CPS. If the DEFAULT flag is not set or there is no configuration file in the PCMCIA storage device, read the configuration from the internal flash.
- `local <remote Path and filename>`—Read the configuration from the local file `<remote Path and filename>`.
- `ftp <remote Path and filename> <IP address of the FTP server> <username> <password>`—Read the configuration from the remote FTP server.
- `sd`—Read the configuration from the PCMCIA storage device (Compact Flash or IDE) and if the REPLACE flag is set, copy the file to the internal flash of the CPS.

**CLI method—save or restore configuration**

Configuration can be saved/restored through the following menus:

- Saving the configuration to the internal flash memory:
  ```bash
  cli>config savetoflash
  ```

- Saving the configuration to a PCMCIA storage device:
  ```bash
  cli>administration backupconfig saveto sd [-default] [-replace]
  ```

- Saving the configuration to a remote server using FTP:
  ```bash
  cli>administration backupconfig saveto ftpserverip <n.n.n.n> pathname <string> username <string> password <string>
  ```

- Loading the configuration from a PCMCIA storage device:
  ```bash
  cli>administration backupconfig loadfrom sd
  ```

- Loading the configuration from a remote server using FTP:
  ```bash
  cli>administration backupconfig loadfrom ftpserverip <n.n.n.n> pathname <string> username <string> password <string>
  ```
5.13 Start and Stop Services

This feature allows daemons (services) to be enabled or disabled without requiring a reboot of the unit. A simple engine detects configuration changes (by performing a 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 must be run 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 a dot (.) 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. Whether 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. An initialization shell script that will be run before the service is started. (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 of any configuration changes.
The command `daemon.sh help` will display a list of services available. 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.

<table>
<thead>
<tr>
<th>Service ID</th>
<th>Shell Script File</th>
</tr>
</thead>
<tbody>
<tr>
<td>NIS</td>
<td>/etc/daemon.d/ypbind.conf</td>
</tr>
<tr>
<td>RPC</td>
<td>/etc/daemon.d/portmap.conf</td>
</tr>
<tr>
<td>DB</td>
<td>/etc/daemon.d/cy_buffering.sh</td>
</tr>
<tr>
<td>NET</td>
<td>/etc/daemon.d/inetd.sh</td>
</tr>
<tr>
<td>LOG</td>
<td>/etc/daemon.d/syslog.sh</td>
</tr>
<tr>
<td>SSH</td>
<td>/etc/daemon.d/sshd.sh</td>
</tr>
<tr>
<td>NTP</td>
<td>/etc/daemon.d/ntpc-client.conf</td>
</tr>
<tr>
<td>SNMP</td>
<td>/etc/daemon.d/snmpd.sh</td>
</tr>
<tr>
<td>IPSEC</td>
<td>/etc/daemon.d/ipsec.sh</td>
</tr>
<tr>
<td>PMD</td>
<td>/etc/daemon.d/pmd.sh</td>
</tr>
<tr>
<td>LP</td>
<td>/etc/daemon.d/lpd.sh</td>
</tr>
<tr>
<td>WEB</td>
<td>/etc/daemon.d/webui.conf</td>
</tr>
<tr>
<td>GDF</td>
<td>/etc/daemon.d/gendial.sh</td>
</tr>
</tbody>
</table>

The following example restarts power management and Data Buffering services, and stops SSH and network timer client services:

```
# daemon.sh PMD stop SSH NTP restart DB
```
Configuration Example

Example of sourced shell script that activates the ntpclient service.

```
# 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)

# 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 5.29: /etc/daemon.d/ntpclient.conf file
```

Example of sourced shell script that activates the ipsec service.

```
# 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

# daemon command line parameters
DPARM="setup --start"
DSTOP="setup --stop"

File Description 5.30: /etc/daemon.d/ipsec.sh file
```
5.14 Security Profiles

A Security Profile consists of a set of parameters that can be configured to control access to the CPS. The CPS offers three pre-defined security profiles, **Secured**, **Moderate**, **Open**, and an option to configure a **Custom** profile. A fifth option, **Default**, sets the parameters to the same as **Moderate**.

The following tables illustrates the properties for each of the Security Profiles. The enabled services in each profile are designated with a check mark.

**NOTE:** The **Default** option sets the parameters to the same as **Moderate**, and the **Custom Profile** allows for individual configurations.

<table>
<thead>
<tr>
<th>Access to CPS</th>
<th>Secured</th>
<th>Moderate</th>
<th>Open</th>
<th>Default</th>
<th>Custom</th>
</tr>
</thead>
<tbody>
<tr>
<td>Telnet</td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>SSHv1</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>SSHv2</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Allow SSH root access</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>HTTP</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>HTTPS</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>HTTP redirection to HTTPS</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>
Table 5-9: Enabled services to access the serial ports for each profile.

<table>
<thead>
<tr>
<th>Access to Serial Ports</th>
<th>Secured</th>
<th>Moderate</th>
<th>Open</th>
<th>Default</th>
<th>Custom</th>
</tr>
</thead>
<tbody>
<tr>
<td>Console (Telnet)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Console (SSH)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Console (Raw)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Serial Port Authentication</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bidirect (Dynamic Mode Support)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

Table 5-10: Enabled protocols for each profile shown with a check mark.

<table>
<thead>
<tr>
<th>Other Services</th>
<th>Secured</th>
<th>Moderate</th>
<th>Open</th>
<th>Default</th>
<th>Custom</th>
</tr>
</thead>
<tbody>
<tr>
<td>SNMP</td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RPC</td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ICMP</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>FTP</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>IPSec</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
</tbody>
</table>
CLI method—selecting a predefined security profile

1. Open the CLI interface by issuing the command:
   
   #CLI

2. Enter the Security Profile menu:
   
   cli> config security profile

3. Type one of the pre-defined Security Profiles and press Enter:
   
   profile> secured moderate open default

4. To view the details of the selected profile, type the command:
   
   profile> show

   A window appears, showing the details of the profile:

```
profile> show
[profile]
  [open]: custom
  [moderate]: custom
  [secured]: custom
  .[custom]
    ftp: no
telnet: yes
..[ssh]
  sshv1: yes
  sshv2: yes
  sshd_port: 22
  root_access: no
  snmp: yes
..[web]
  http: yes
  https: yes
  http_port: 80
  https_port: 443
  http2https: no
  rpc: yes
  ipsec: no
  icmp: yes
..[ports]
  ssh2sport: yes
telnet2sport: yes
  raw2sport: yes
auth2sport: no
```
CLI method—configuring a custom profile

1. Open the CLI interface by issuing the command:
   
   #CLI

2. Enter the Custom Security Profile menu:
   
   cli> config security profile custom
   
   custom>

3. Configure network protocols.
   The following parameters are available under the custom menu:
   
   • FTP
   • ICMP
   • IPSec
   • RPC
   • SNMP
   • Telnet
   
   To enable or disable a parameter issue the following command:
   
   custom> [parameter] <option>
   
   Where possible, values for <option> are yes to enable a parameter and no to disable it.
   
   To see the Custom profile configuration, type the command “show”.
   
   custom> show
A window appears, showing the details of the profile:

```
custom> show
[custom]
  ftp: no
telnet: yes
[ssh]
  sshv1: yes
  sshv2: yes
  sshd_port: 22
  root_access: no
  snmp: yes
[web]
  http: yes
  https: yes
  http_port: 80
  https_port: 443
  http2https: no
  rpc: yes
  ipsec: no
  icmp: yes
[ports]
  ssh2sport: yes
telnet2sport: yes
raw2sport: yes
auth2sport: no
```

4. Configure Secure Shell (SSH) options.

Change the directory from `custom>` to `ssh>`. The following parameters are available under the `ssh>` menu:

- SSHv1 - Secure Shell version 1
- SSHv2 - Secure Shell version 2
- sshd_port - SSH port ID
- root_access - Allow root access

To enable or disable a parameter, issue the following command:

```
ssh> [parameter] <option>
```

Where possible, values for `<option>` are `yes` to enable the parameter and `no` to disable it.
To assign an SSH port, type:

```
ssh> sshd_port <portnumber>
```

To see the SSH configuration, type the command “show”

```
ssh> show
```

A window appears, showing the details of the profile:

```
ssh>show
[ssh]
  sshv1: yes
  sshv2: yes
  ssd_port: 22
  root_access: no
```


Change the directory from `custom>` to `web>`. The following parameters are available under the `web>` menu:

- `http`
- `http_port`
- `http2https`
- `https`
- `https_port`

To enable or disable a parameter, type the command:

```
web> [parameter] <option>
```

Where possible, values for `<option>` are `yes` to enable a parameter and `no` to disable it.

To assign an http or https port, type:

```
web> http <portnumber> https <portnumber>
```

To see the Web configuration, type the command “show”:

```
ssh> show
```

Change the directory from custom> to ports>. The following parameters are available under the ports> menu:

- auth2sport—Authentication to Access Serial Ports
- ssh2sport—SSH to Serial Ports
- raw2sport—Raw Connection to Serial Ports
- telnet2sport—Telnet to Serial Ports

To enable or disable a parameter, issue the command:

```
ports> [parameter] <option>
```

Where possible, values for <option> are yes to enable a parameter and no to disable it.

To see the ports configuration, type the command:

```
ports> show
```

```
ports>show
[ports]
  ssh2sport: yes
  telnet2sport: yes
  raw2sport: yes
  auth2sport: no
```

7. To activate the configuration, type the following command:

```
cli> config runconfig
```
8. To save the configuration, type the following command:
   \texttt{cli> config savetoflash}

9. To exit CLI and return to the shell, type the following command:
   \texttt{cli> quit}

\textbf{NOTE:} The protocols and access methods for the Serial Ports must match the selected Security Profile. To configure parameters for the Serial Ports, see "Chapter 8: Profile Configuration" on page 245.
Chapter 6: Power Management

Power Management (PM) comprises a family of Rack Power Distribution Units (rPDUs) that enable remote power control of servers and network gear. Through a serial port, the administrator can use power management to control all the equipment connected to its outlets. When used in conjunction with APC console servers, the APC Console Port Server delivers easier management capabilities and faster problem solving by integrating console access and power control into a single interface. The administrator can, for example, reboot the data center equipment when it crashes, without leaving a console session (Telnet or SSH). To do that, the administrator must simply press a configurable hotkey and select the appropriate option from the menu displayed in the session.

This chapter explains all power management configuration that is integrated with the APC Console Port Server.
6.1 Power Management Configuration

The CPS can have multiple rPDUs connected to serial ports that are configured for power management. Devices can be plugged into outlets on the rPDUs and also connected to other serial ports on the CPS. In addition, one or more outlets can be configured for each port and controlled individually or simultaneously with other outlets in a configured group. The CPS administrator can control all outlets or can assign outlets to individual users or groups of users.

![Configuration Diagram]

Figure 6.1 - Configuration diagram

Figure 6.1 shows a typical setup for the rPDU and the APC Console Port Server. The rPDU'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 rPDU. These port denominations will be used in the descriptions below.

Prerequisites for Power Management

In order to control individual outlets or groups of outlets from the Multi Outlet Control page, the following prerequisites must be met:

- An rPDU must be plugged into one of the serial ports, and that serial port must be configured for power management.
- The APC PDU must be running firmware version 2.7.3 or later.
• A device must be plugged into at least one outlet on the rPDU, and this device must be connected to a serial port.
• The rPDU and the outlet numbers to which the device is plugged must be configured on the serial port to which the device is connected.

Configuring Power Management

There are two types of parameters.

VI method—involved parameters and passed values

1. Parameters to the port YY where the rPDU is connected:
   • sYY.protocol: Independent Power Distribution Unit (iPDU).
   • sYY.pmtype: The PDU manufacturer (e.g., APCPDU).
   • sYY.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 the format of <outlet list> 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.
   • sYY.pmNumOfOutlets: The number of outlets of the rPDU. Default: 8.
   • sYY.pmSessions: Only users logged in with the connection method defined by this parameter will be allowed to access the rPDU. You must also define the authentication method in the sXX.authtype parameter and configure the sXX.users parameter in order to allow users to access the rPDU port. Valid values are: none, ssh, telnet or ssh_telnet.

   IMPORTANT: By defining the sYY.pmSessions parameter and making all other necessary configurations, a user can access the rPDU directly by opening an SSH or Telnet connection to the desired port. After entering the username and password, the user will have direct access to the pmCommand menu.

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 hotkey that starts a power management session. Default: ^p (Ctrl+p).

• *sYY.pmoutlet*: The outlet list where the server XX 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 a space. For example, one power supply is plugged in the second outlet of the rPDU connected in serial port 1. The other is plugged in the third outlet of the rPDU connected in serial port 5. The value is 1.2 5.3.

---

**SXX.pmusers NOTES:** The ellipses in the field format for sXX.pmusers indicate that you can add as many users as you need. The brackets (i.e., [ ]) indicate that the parameter is optional, again indicating that you can configure more than one user. The separator is the semicolon.

---

**CLI method—rPDU configuration**

The CLI allows you to set some ports to the *pm* protocol. (Configuring a port with the *pm* protocol means that you have an rPDU connected to it.) Then configure power management parameters in other ports that are not configured as protocol *pm*.

**Example of power management in the CLI:**

This example explains power management for an 8-port CPS unit and an rPDU with eight outlets connected to serial port 1 of the CPS.

To manage the rPDU, configure port 1 as *pm* protocol and then associate which ports (any port except port 1) of the CPS unit will be allowed to issue power management commands to port 1.

1. Open the CLI interface by issuing the command.
   
   ```
   # CLI
   ```

2. Enable the serial port to which the rPDU is connected. For example, serial port 1 is being configured for rPDU.
   ```
   cli>config physicalports 1 enable yes
   ```

3. Configure the serial port to which the rPDU is connected. For example, serial port 1 is being configured for rPDU.
   ```
   cli>config physicalports 1 general protocol pm
   ```
You can specify the type of session (SSH, Telnet, or both) from which users will be allowed to connect to the rPDU.

`cli>config physicalports 1 general pmsessions ssh`

The command above restricts access to the rPDU port to users who connect to the CPS using SSH. Valid values for the `pmsessions` parameter are `ssh`, `telnet`, `ssh_telnet`, and `none`.

4. Configure the CPS ports from which commands to the configured rPDU port will be allowed to be issued.

The following command will allow users connected to serial port 2 to run power management commands to the rPDU, which is connected to serial port 1 of the CPS.

`cli>config physicalports 2 powermanagement powermanagement>enable addoutlet pm 1 outlets 1,2`

The above configuration makes an association between the outlet numbers (outlet 1 and outlet 2) and the power strip connected at port 1. The logic means that this server's power cords are plugged into outlets 1 and 2 of the power strip connected to port 1.

5. Configure allowed users.

You can restrict access to rPDU units to specified users by issuing the command:

`enable> pmusers test1,test2`

The command above allows the users test1 and test2 to run power management commands into the rPDU connected to serial port 1.

6. Configure the hotkey.

You must define a hotkey, or users will not be able to open the rPDU menu:

`enable> pmkey ^p`

7. Checking the configuration

`enable> show`

A prompt similar to the following prompt should appear:

```
[enable]
pmkey: ^p
pmusers: test1,test2
[PM Alias and Outlet Number]:
Port1 - [no alias] Outlets:1,2
```
8. Activate the configuration.
   Return to the main menu by running the command:
   
   enable> return
   powermanagement> return
   cli> config
   config>

9. Run the command.
   
   config > runconfig

10. Save the configuration.
    
   config > savetoflash

11. Manage the rPDU unit.
    
   To manage the outlets of the rPDU, issue the following command, where “1” is the port number where the rPDU is connected in the CPS:
   
   cli> applications pm 1
   
   You will be prompted with the pm command menu. For more information about the pm command menu, see “Accessing the Regular Menu from the Console Session” on page 198.

12. Exit the CLI mode.
    
   To exit the CLI mode and return to CPS’s shell, type the following command:
   
   cli> quit

Changing the rPDU password

1. Change the connection protocol on the serial port to which the rPDU is connected.
   
   a. Change the connection protocol on the serial port by editing the /etc/ports/pslave.conf file. For example, change the serial port 1 protocol from iPDU to socket_ssh or socket_server.
      
      s1.protocol socket_ssh, or
      s1.protocol socket_server
   
   b. Save the pslave.conf file and activate the new configuration by entering the following command:
      
      [root@CPS root]# runconf
2. Access the rPDU console using the protocol you configured, indicating the port to which the rPDU is connected.

For example, if a socket sever protocol is configured in step 1 type the following command, where the TCP port number is the serial port to which the rPDU is connected:

[root@CPS root]# telnet <CPS ip address> [tcp port number]

A prompt similar to the following appears:

Entering charater/text Mode
Escape character is ^
Username:

3. Log in to the rPDU console (the default username and password are both apc).

4. Change the rPDU password.

Select System, then User Manager.

Select the user (Administrator or Device User), then follow the prompts to enter the new password. When you finish entering the new password, you will return to the User Manager menu.

To save the new password, select Accept Changes.

5. To close the connection to the rPDU console, select the main menu option Logout.

6. Edit the /etc/pm.APCPDU config file and change the passwd parameter as follows:

\[
\text{admPasswd} = "<the password saved in the rPDU interface>"
\]

Save the new pm.APCPDU file and activate the new configuration by entering the following command.

[root@CPS root]# saveconf

7. Change the connection protocol for the serial port back to the original iPDU.

a. Edit the /etc/portslave/pslave.conf file as follows:

\[
\text{s1.protocol iPDU}
\]

b. Save the pslave.conf file and enter the following command to activate the new configuration:

[root@CPS root]# runconf
8. Restart the pmd process for the new configuration file to take effect.

*pmd* is a Linux daemon process to control communication between the CPS and the PM.

a. Execute the *ps* command to note the current *pmd* process

```
[root@CPS root]# ps -ef | grep pmd
878 root  644 S   /bin/pmd
1108 root  552 S   grep pmd
```

b. Restart the *pmd* process by issuing the following command:

```
[root@CPS root]# daemon.sh restart PMD
```

The system prompts the following:

```
[root@CAS root]# Sep 22 14:32:04 src_dev_log@CAS showlogmsg:/bin/daemon.sh: CONFIG: Network
daemon [pmd] started
```

c. Check to see if the process restarted. Note the process ID, which should be different from the earlier executed *ps* command.

```
[root@CPS root]# ps -ef | grep pmd
1126 root  680 S   /bin/pmd
1130 root  552 S   grep pmd
```

**Accessing the Regular Menu from the Console Session**

The Power Management Utility can be used to manage power on devices plugged into one or more outlets on an rPDU. The PM Utility can be started by the following two methods.

1. Issuing the *pm* command, which invokes the following menu and prompt:

```
---------------------------------------------------------------------
APC Corporation  
Power Management Menu v1.0  
---------------------------------------------------------------------
---------------------------------------------------------------------
APC Power Management Menu  
---------------------------------------------------------------------
1. Exit 2. individual rPDUs 3. multi-outlet device 4. Info 
Please choose an option:
```
The following table explains each menu item, its use, and its behavior.

**Table 6-11: Menu Options for PM Utility**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exit</td>
<td>Exits the PM Utility and returns to the CPS shell.</td>
</tr>
<tr>
<td>Individual rPDUs</td>
<td>Invokes a menu for controlling and monitoring rPDUs and for controlling power on their individual outlets. See “Manage Devices Plugged into a Single Outlet” on page 199 for more details.</td>
</tr>
<tr>
<td>Multi-outlet device</td>
<td>Invokes a menu for controlling power on groups of outlets connected to devices. These outlets can be on the same rPDUs or on different rPDUs. See “Manage Devices Plugged into Multiple Outlets” on page 202 for more details.</td>
</tr>
<tr>
<td>Info</td>
<td>Shows help text explaining each option.</td>
</tr>
</tbody>
</table>

2. Issuing the *pmCommand*

Use: `pmCommand <serial port number> <command> <arguments>`

where:

- `<serial port number>`—the serial port number configured as rPDU.
- `<command> <arguments>`—the PM command and its arguments.

For more information, see the list of commands in Table 6.2.

**Using the Power Management Utility**

You can use the Power Management Utility to control rPDUs and individual outlets.

**Manage Devices Plugged into a Single Outlet**

Selecting option “2” for “Individual rPDUs” from the PM menu invokes the following prompt:

```
Give serial port number:
```
Entering the serial port number that is configured for power management brings up the following menu options:

```
American Power Conversion
Power Management Menu v1.0

PowerPort: SerialPort9
Number of units: 0
Serial Port: 9

APC Power Management Menu

1. Return 8. Current 15. PDU Cold Start Delay

Please choose an option: _
```

The following table explains each menu item, its use and behavior.

**Table 6-12: Power Management Individual rPDUs Menu**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Return</td>
<td>Exits and returns to the main power management menu.</td>
</tr>
<tr>
<td>Help</td>
<td>Provides a brief description of the menu items.</td>
</tr>
<tr>
<td>Who Am I</td>
<td>Displays the current username.</td>
</tr>
<tr>
<td>On</td>
<td>Turns an outlet On. Prompts you to enter the outlet number.</td>
</tr>
<tr>
<td>Off</td>
<td>Turns an outlet Off. Prompts you to enter the outlet number.</td>
</tr>
<tr>
<td>Reboot</td>
<td>Restarts the PDU.</td>
</tr>
<tr>
<td>Status</td>
<td>Provides an overall status of the selected outlet.</td>
</tr>
<tr>
<td>Current</td>
<td>Displays the amount of current that is running through the rPDU.</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>--------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Power</td>
<td>Reading of instantaneous power consumption by the rPDU.</td>
</tr>
<tr>
<td>Reboot Duration</td>
<td>The delay between removing power from an outlet because of a reboot and reapplying power to an outlet.</td>
</tr>
<tr>
<td>Power On Delay</td>
<td>Determines the time interval between the triggering event and power being applied to the outlet.</td>
</tr>
<tr>
<td>Power Off Delay</td>
<td>The time from the triggering event (such as a server confirming a shutdown) until power is removed from the outlet.</td>
</tr>
<tr>
<td>Name</td>
<td>Add a name or an alias for an outlet.</td>
</tr>
<tr>
<td>Version</td>
<td>View the software and hardware version of the rPDU.</td>
</tr>
<tr>
<td>PDU Cold Start Delay</td>
<td>Set or read the PDU cold start delay, in seconds.</td>
</tr>
</tbody>
</table>
| Overload Restriction | Prevent users from applying power to outlets during an overload condition. You can set the following restrictions for each outlet:  

**None:** You can apply power to outlets regardless of an Overload Alarm or Near Overload Warning.  

**On Warning:** You cannot apply power to an outlet on the selected phase or bank if the current for that phase or bank has exceeded the Near Overload Warning threshold.  

**On Overload:** You cannot apply power to an outlet on the selected phase or bank if the current for that phase or bank has exceeded the Overload Alarm threshold.  

| Overload Alarm        | View or configure the number of amps that will cause an overload of this phase or bank.               |
| Near Overload Warning | View or configure the number of amps at which to generate a warning that the rPDU is nearing overload of a phase or bank. |
Once you select a command, the following options may occur:

- If command you select applies to the rPDU unit, then the command will be issued
- If the command you select applies to the outlets, the following prompt appears

```plaintext
Outlet name or outlet number(? for help, m for main menu):
```

Enter one or more outlet numbers separated by commas or dashes, as shown in the following screen example, or enter “all.”

```plaintext
Outlet name or outlet number(? for help, m for main menu): 1,3,5
```

## Manage Devices Plugged into Multiple Outlets

You can use the Power Management Utility to simultaneously control all outlets that are configured to the same serial port, regardless of whether the outlets are on the same rPDU. This option is applicable to devices with multiple power supplies.

Selecting option “3” for “Multi-outlet Devices” from the PM menu invokes the following menu and prompt:

```
-------------------------------------------
APC Power Management Menu
-------------------------------------------

Please choose an option:
```
The following table explains each menu item, its use, and its behavior.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Return</td>
<td>Return the user to the first menu.</td>
</tr>
<tr>
<td>On</td>
<td>Apply power to all of the outlets belonging to this multi-outlet device.</td>
</tr>
<tr>
<td>Off</td>
<td>Remove power from all of the outlets belonging to this multi-outlet device.</td>
</tr>
<tr>
<td>Cycle</td>
<td>Turn the outlets off and back on.</td>
</tr>
<tr>
<td>Status</td>
<td>Issue individual status commands on each of the outlets belonging to this multi-outlet device, as shown in the following example:</td>
</tr>
<tr>
<td></td>
<td>OK 1: ON : Outlet 1</td>
</tr>
<tr>
<td>Show</td>
<td>Shows which outlets in which rPDU chain belong to the multi-outlet device, as shown in the following example:</td>
</tr>
<tr>
<td></td>
<td>alias: (null) port: ttyS4 outlets: 3.1 3.5 3.4</td>
</tr>
<tr>
<td>Info</td>
<td>View help text explaining each option.</td>
</tr>
</tbody>
</table>

**NOTE:** The multi-outlet device menu is inaccessible if there are no devices configured with the `pmoutlet` parameter in `/etc/portslave/pslave.conf`. You must configure the “sxx.pmoutlet” line in the pslave.conf file.

In the following example, the Sun Server is a multi-outlet device connected to outlet 7 of rPDU 1 and outlet 2 of rPDU 2. The start-up interval sequence is 1.7, then 2.2.

```
  s3.pmoutlet 1.7, 2.2
  s3. alias    Sun Server
```
To Manage Multiple rPDUs from the Command Line

1. Connect to the CONSOLE port of the CPS or use Telnet or SSH to access the CPS, and log in.

2. Enter the `pm` command.

   [root@CPS root]# pm

   The power management menu displays as shown in the following screen example.

   To control power on multi-outlet devices, enter the number 3.

   Please choose an option: 3

   The power management utility displays as shown in the following screen example.

3. To choose a menu option, enter its corresponding number.
The following prompt appears

Please supply the serial port number or the alias for the multi-outlet devices
If in doubt, type ? followed by enter and a list of available devices will be shown

4. Enter the number or alias of the serial port to which the multi-outlet device is connected. The command is executed.

To Manage Power Through the Console

1. Open a console session for the serial port, using Telnet or SSH.

2. Access the rPDU regular menu.

   For example, access the rPDU regular menu to change the power status when the server crashes. Type the pre-configured hot-key, ^P.

   If you do 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 you do 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):

   Type the outlets you want to manage, before reaching the main menu. The main menu appears only if you have permission for the outlets. Type 'h' to view text that explains what to type. Type 'l' to cause the PM session to be logged out, and to cause the user to return to the Console Session. If you have permission to access the outlets of this server, these outlets will be managed by the PM session.
3. The PM Regular User Menu:

```
American Power Conversion
Power Management Menu v1.0
---------------------------------------------
PowerPort: SRPDU
Number of units: 1
Serial Port: 7
---------------------------------------------
APC Power Management Menu
PowerPort: SRPDU
5. Off                    10. Reboot Duration

Please choose an option:
```

*Menu Description 6.1: APC Power Management regular menu*

**Table 6-13: PM regular user menu options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Return</td>
<td>Exits and returns to the main power management menu.</td>
</tr>
<tr>
<td>Help</td>
<td>Provides a brief description of the menu items.</td>
</tr>
<tr>
<td>Who Am I</td>
<td>Displays the current username.</td>
</tr>
<tr>
<td>On</td>
<td>Turns an outlet on. The system prompts you to enter an outlet number.</td>
</tr>
<tr>
<td>Off</td>
<td>Turns an outlet off. The system prompts you to enter an outlet number.</td>
</tr>
<tr>
<td>Reboot</td>
<td>Turns an outlet off and on again, and recycles the power. The system prompts you to enter an outlet number.</td>
</tr>
</tbody>
</table>
Table 6-13: PM regular user menu options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status</td>
<td>Provides an overall status of the selected outlet.</td>
</tr>
<tr>
<td>Power</td>
<td>Set the time interval (in seconds) that the system waits between turning on the currently-selected outlet and the next outlet.</td>
</tr>
<tr>
<td>Name</td>
<td>Add a name or an alias for an outlet.</td>
</tr>
<tr>
<td>Current</td>
<td>Displays the amount of current that is running through the rPDU.</td>
</tr>
<tr>
<td>Version</td>
<td>Displays the software version and hardware version of the rPDU.</td>
</tr>
</tbody>
</table>

4. To check the status of the server's outlet or the outlet list, type '7' and wait for the answer. For example:

```
OK
1: ON   : Outlet 1
-------------------------------------------------------------------
APC Power Management Menu
-------------------------------------------------------------------
Please choose an option:
```

Menu Description 6.2: Outlet Status
5. Reboot the server.

The Reboot command turns the power off for several seconds, then turns it on again. Type '6' and wait for the answer. For example:

```
American Power Conversion
Power Management Menu v1.0
-------------------------------
PowerPort: SRPDU
Number of units: 1
Serial Port: 7
-------------------------------
APC Power Management Menu
PowerPort: SRPDU
-------------------------------
5. Off          10. Reboot Duration
-------------------------------
Please choose an option:
```

*Menu Description 6.3: Outlet Cycle*
6. Manage other outlets.

Perform the following procedure if you need to access other outlets.

a. Enter option “7. Status” to view the outlets you are authorized to manage, and at the outlet name or outlet number prompt, type “all”.

b. Select a command from the menu, and then select the outlet that you are authorized to manage.

From this point, all commands will be related to the second outlet of the rPDU in the port 1.

7. Return to the Console Session.

You can exit from the PM session and return to the Console Session in three ways:

a. Type the hot-key at any time.

b. If the session is waiting for a menu option, type the option 1 - Exit.

c. 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
Chapter 7: PCMCIA Cards
Integration

PCMCIA slots are available only on the AP9303 CPS. PCMCIA slots allow enhanced functionality with support for many interface cards, such as Ethernet, modem (V.90, GSM, CDMA, and ISDN) and wireless LAN.

7.1 Supported Cards
To view a list of supported PCMCIA cards, see the APC Web site: http://www.apc.com

Tools for Configuring and Monitoring PCMCIA Devices

During the CPS 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`—Use this command to check the status of a socket, or to see how the socket is configured. Type `cardctl` to see the syntax of the command. Type `cardctl config` to display the card configuration. Use `cardctl ident` to obtain card identification information. Use `cardctl eject` to stop the application and unload the client driver, and use `cardctl insert` to reload the driver and restart the application.

**NOTE:** `cardctl suspend`, `cardctl resume`, and `cardctl reset` are not supported.

Ejecting Cards

You can insert the card at any time, and the drivers should be loaded automatically. However, you must run `cardctl eject` before ejecting the card, to stop the application using the card. Otherwise the CPS may hang during the card removal. You must specify the slot number when you use the `cardctl` command. For example:

- `cardctl eject 0` for the lower slot
- `cardctl eject 1` for the upper slot
7.2 PCMCIA Network Device Configuration

Ethernet PC Cards

The onboard Ethernet device has the name *eth0*. The first PCMCIA Ethernet card or wireless LAN card detected will receive the name *eth1*, and 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`.

VI method

The factory default for the `/etc/network/interfaces` file has the following lines:

```plaintext
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 7.1: part of the `/etc/network/interfaces` file*

Remove the `#` in the beginning of the line, and change the IP addresses to suit your network configuration. For example, you may want the following configuration:

```plaintext
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 7.2: part of the `/etc/network/interfaces` file*

Run `saveconf` to save this configuration in the flash, so that it can be restored in the next start-up. Run `cardctl insert` to load the network drivers with the new configuration.
IMPORTANT: 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 CPS may hang. To change the network settings, edit the /etc/network/interfaces file.

Removing the configuration from a Ethernet PCMCIA device

Before removing the configuration from an Ethernet PCMCIA card configured in CPS, run “cardctl eject <slot number>”, then delete the lines of the interface to remove from the /etc/network/interfaces file.

CLI method—Ethernet PCMCIA

To configure an Ethernet PCMCIA card using the CLI:

1. Open the CLI interface by issuing the command:
   
   # CLI

2. Configure the IP address and network mask.

   In the following example, the PCMCIA card is placed on slot 1 of the unit. Run the command:

   cli>config network pcmcia 2 ethernet ip 192.168.0.100 mask 255.255.255.0

   This command will configure 192.168.0.100 as the IP address and 255.255.255.0 as the netmask.

3. Save the configuration.

   cli>config savetoflash

4. Activate the configuration.

   Due to CLI restrictions, the configuration must be activated in the shell prompt, by running the following two commands in the presented sequence:

   # cardctl eject
   # cardctl insert

5. Exit the CLI mode.

   To exit the CLI mode and return to the CPS’s shell, type the following command:

   cli> quit
Wireless LAN PC Cards

First, complete the PCMCIA network configuration. Additionally, configure the wireless driver in the file /etc/pcmcia/wireless.opts.

For example, to configure the network name as *MyPrivateNet*, and the WEP encryption key as *secu1*, add the following settings to the default "*,*,*,*)" entry:

```
*,*,*,*)
  INFO="This is a test"
  ESSID="MyPrivateNet"
  KEY="s:secu1"
```

*File Description 7.3: part of the /etc/pcmcia/wireless.opts file*

**NOTE:** The "s:" prefix in the KEY line indicates that the key is an ASCII string, not hexadecimal digits. You can enter five ASCII characters or ten hexadecimal digits for WEP 64-bit (also known as 40-bit). You can enter 13 ASCII characters or 26 hexadecimal digits for WEP 128-bit (also known as 104-bit). Any ASCII characters will be accepted if the line starts with "s:". Otherwise, only characters between [0-9,a-f] will be accepted. Check your PCMCIA card specifications.

There is a generic sample in the end of the wireless.opts file that explains all possible settings. For more details about 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, then run 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 you 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
Removing the configuration from a wireless PCMCIA device

Before removing the configuration from a Wireless PCMCIA card configured in CPS, you should first run “cardctl eject <slot number>” and then delete the lines of the desired interface from the /etc/network/interfaces file.

CLI method—wireless PCMCIA

You must configure four parameters to set up the wireless network:

• **ESSID**—The identifying name of an 802.11b wireless network. Specify the ESSID in your client setup to ensure that you connect to your wireless network instead of your neighbor’s network by mistake.

• **IP address**—The IP address of the wireless interface.

• **Network Mask**—Network mask of the wireless interface.

• **Encryption**—Enables WEP data encryption. Encryption is not required to set up a wireless network, but it is strongly recommended due to security issues.

To configure a wireless PCMCIA card using the CLI:

1. Plug the PCMCIA wireless device into one of the available slots (slot 2, for this example) and open the CLI interface by issuing the command:

   # CLI

2. Configure basic parameters.

   The command below configures 192.168.100.1 as the IP address and 255.255.255.0 as the network mask:

   cli>config network pcmcia 2 wireless ip 192.168.100.1 mask 255.255.255.0

   Next, configure the ESSID (Extended Service Set ID). The ESSID will be the string “test” for this example:

   cli>config network pcmcia 2 wireless essid test

3. Set security parameters.

   It is strongly recommended that you enable encryption on wireless connections. The following command enables connection encryption and sets the string “test1” as the key.

   cli>config network pcmcia 2 wireless encrypt yes key s:test1

**IMPORTANT:** See the note about WEP keys on Page 214.
4. Activate the configuration.
   
   `cli>config runconfig`

5. Save the configuration.
   
   `cli>config savetoflash`

6. Exiting the CLI mode.
   
   To exit the CLI mode and return to the CPS’s shell, type the following command:
   
   `cli> quit`

**Modem PC Cards**

The modem device is named `/dev/ttySn`, where \( n \) is the number of embedded serial devices plus 1. For instance, if the CPS has 32 onboard serial devices, the modem card is named `/dev/ttyS33`.

**VI method**

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 allow PPP to login the caller.

To allow PPP connections:

1. Enable login and PAP authentication in `/etc/mgetty/login.config`.
   
   Enable the desired authentication in `/etc/mgetty/login.config`. For example, 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 nosdcomp nodeflate`

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 example, you may add the following line:
   
   `mary * marypasswd *`

3. Create the user for login in the RADIUS server.
   
   If the login option was used, either create the user locally (by running `adduser`) or create the user in the RADIUS server for RADIUS authentication.

4. Copy `/etc/ppp/options.ttyXX` as `/etc/ppp/options.ttyS33` (the modem port).
Copy `/etc/ppp/options.ttyXX` to have the device name assigned to the PCMCIA modem. For instance, if the modem is `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.

5. Uncomment local and remote IP addresses in `/etc/ppp/options.ttyS33`.

   Uncomment the line that assigns the local and remote IP addresses in `/etc/ppp/options.ttyS33` (or the `tty` name in your system). For instance, you may want to assign a local IP address of 192.168.0.1 and a remote IP address of 192.168.0.2.

6. Save `/etc/ppp/options.ttyS33` in flash.

7. Create an entry in `/etc/config_files`.

   The entry should have the name of the file you created, so that the new file can be saved to the flash. For example, you must add a line with `/etc/ppp/options.ttyS33` in `/etc/config_files`.

8. Run `saveconf` to save the files listed in `/etc/config_files` to the flash.

9. Insert the PCMCIA modem, if it is not already inserted.

10. Run `ps` to see that `mgetty` is running. If so, the CPS is ready to receive dial-in calls.

11. Establish a PPP connection with the CPS.

   From the remote system, use `pppd` to dial and establish a PPP connection with the CPS. The remote system should have the login user name set in its `/etc/ppp/pap-secrets` to successfully log in to the CPS.

**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—Callback ensures that users are who they claim to be by calling the user back at a well-known or pre-configured number.

The steps to allow callback are divided into two parts. Part One is the configuration for the Advanced Secure Console Port Server (Server Side CPS Setup). Part Two is the configuration for the client side (Client Side Setup).
Server Side CPS Setup.

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
```

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 similar to the following:

```
* - - /bin/login @
```

Add this line above the preceding line:

```
<pseudo callback name>- - /sbin/callback -S <phone number of the client>
```

For example, with the pseudo callback name 'call' and the dial-back number '123456':

```
call - - /sbin/callback -S 12345
```

NOTES:

1. The order of configuration in `/etc/mgetty/login.config` is important. By default, the line `* - - /bin/login @` is at the end of the file. This line allows any user to log in and be verified by the login program. If you add the callback line after this line, the callback program will not be initiated when you try to log in. Instead, the login program will be used to verify you since the verification command will be encountered first. List the callback users first if you want the option of having some users access the callback program and of having the other users access the login program.

```
call - - /sbin/callback -S 12345

call2 - - /sbin/callback -S 77777
* - - /bin/login @
```

The example above will allow you to choose whether to use the callback functionality. If you log in as call or call2, then callback starts immediately. If you log in as any user other than call or call2, callback will not start and you will be verified by the login program.

2. Do not use an asterisk (*) instead of a callback user name. Mgetty will fall to infinite callback.

3. If you do not specify a telephone number, callback will ask for a number after you log in as the pseudo callback user.
3. If you plan to login through PPP with PAP authentication, create a PAP user name in /etc/ppp/pap-secrets.

   Add a line similar to the following: (include the quotes and the two asterisks).

   "myUserName" * "myUserNamePassword"

4. If you plan to log in through PPP, follow steps 4–9 in the preceding section, “Modem PC Cards”. See “Modem PC Cards” on page 216.

5. Create users.

   a. Create a new user with the command adduser myUserName.

      This will create an entry in /etc/passwd that resembles the following example:

      myUserName:$1$/3Qc1pGe$.h3hzkaJQJ/:503:503:Embedix
      User,,,:/home/myUserName:/bin/sh

   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 in step 6. In the following example, the script is: /usr/ppp/ppplogin

      myUserName:$1$/3Qc1pGe$.h3hzkaJQJ/:503:503:Embedix
      User,,,:/home/myUserName:/usr/ppp/ppplogin

6. If you executed Step 5b, create the ppp login script.

   a. Create a script called /etc/ppp/ppplogin following this format:

      #!/bin/sh
      exec /usr/local/sbin/pppd <ppp options>

   b. Make the script executable.

      Type chmod 755 /etc/ppp/ppplogin.

   c. Save this file to flash.

      Save this file to flash so the next time the CPS is restarted, you will not lose the new file. Add /etc/ppp/ppplogin into /etc/config_files, then execute saveconf.

7. Change the permission of pppd.

   Type chmod u+s /usr/local/sbin/pppd
**TIP.** To prevent manually changing permission every time your CPS 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 CPS reboots, this change will be in effect. You should not need to manually change the pppd permission.

8. Your CPS is ready to establish a callback connection. See “Client Side Setup” on page 220 to start the callback connection.

**Client Side Setup.**

1. Activate the Show Terminal Window option.
   (From Windows 2000) Navigate to your Connection window (the window to dial the CPS) and select, in order, Properties>Security>Interactive Logon and Scripting>Show Terminal Window.

2. Disable or enable encryption protocols.
   If you are going to use PPP connection with PAP authentication, make sure you disable all other encryption protocols.
   (From Windows 2000) Navigate to your Connection window (the window to dial the CPS) and select, in order, Properties>Security>Advanced (custom settings)>Settings>Allow these protocols. Disable all protocols except the PAP protocol.

3. Set up the 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 all tasks because it thinks the call was dropped with no reason.
   (From Win2000) Navigate to the Windows control panel and select, in order, Phone and Modem>Modems>select your modem>Properties>Advanced. Add &c0s0=1 to Extra Settings.

4. Call your CPS.
   a. Dial to the CPS modem using either the normal username or the ppp username that you created in Step 5 when configuring the server side.
   b. Once a connection is made, you receive a login prompt.
   c. Log in with the pseudo callback name to start the callback.
d. Your connection will be dropped, and then the CPS will call you back.
e. After the CPS is reconnected to you, you receive a login prompt again.
f. Now you can:
   • Log in through character mode: Log in with username and password. You will get the
     CPS shell prompt.
   • Log in through ppp: Click Done on the Terminal Window.

**CLI method—modem PCMCIA**

To configure a modem PCMCIA card using the CLI, follow the steps:

1. Open the CLI interface by issuing the command:

   ```
   # CLI
   ```

2. Enable the PCMCIA modem and configure it.

   The line below configures a PCMCIA modem placed on slot 2 with the local IP address
   10.0.0.1 and the remote IP address 10.0.0.2:

   ```
   cli>config network pcmcia 2 modem ppp yes localip 10.0.0.1 remoteip 10.0.0.2
   ```

3. Enable callback (optional).

   To enable the callback feature for the PCMCIA modem placed in slot2, run the following
   commands in the given sequence:

   ```
   cli>config network pcmcia 2 modem
   modem>ppp yes
   modem>enablecallback yes
   modem>callbacknum 55552515 localip 10.0.0.1 remoteip 10.0.0.2
   ```

4. Activate the configuration.

   ```
   cli>config runconfig
   ```

5. Save the configuration.

   ```
   cli>config savetoflash
   ```

6. Exit the CLI mode.

   To exit the CLI mode and return to the CPS’s shell, type the following command:

   ```
   cli> quit
   ```

7. Create users.
The CLI does not automatically provide the PPP nor the system users addition, so you must add them through the shell. See “Create a user name in /etc/ppp/pap-secrets.” on page 216 and “Create the user for login in the RADIUS server.” on page 216. To set up the client side, see “Client Side Setup” on page 220.

GSM Card Configuration

This works for firmware version 2.1.3 and up. The GSM card can be used either as a Dial-in or Dial-out profile, but both will be connected through GSM modulation. You also have the option to close a ppp connection using GSM (CLI mode). All of these options are shown in the following steps:

**VI method**

1. In `/etc/mgetty/mgetty.config`, add this entry:
   ```
   port ttyS2
   data-only y
   init-chat "" \d\d\d++\d\d\d\dATZ OK
   ```
   In the preceding example, `ttyS2` may have to be changed to the serial port that will be assigned to the GSM card. For example, replace `ttyS2` with `ttyS9` for a CPS AP9303.

2. If the SIM card needs a PIN, edit `/etc/pcmcia/serial.opts`. Uncomment the line
   ```
   INITCHAT="- \d\d\d++\d\d\d\datz OK at+cpin=1111 OK"
   ```
   and replace '1111' with the PIN.

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 locates the network. It then flashes green.

**CLI method**

To configure a GSM PCMCIA card using the CLI, follow the steps:

1. Open the CLI interface by issuing the command:
   ```
   # CLI
   ```

2. Configure the GSM parameters.
Depending on the way you wish to use the GSM card, some parameters do not need to be configured.

**Configurable parameters:**

**PIN NUMBER:** The command below will configure 1010 as the PIN number:

```
cli>config network pcmcia 2 gsm pin 1010
```

**LOCALIP/REMOTEIP:** Only configure this parameter if you want to establish a PPP connection. The first command below defines the unit’s local IP address and the second command defines the other side IP address.

```
cli>config network pcmcia 2 gsm localip
cli>config network pcmcia 2 gsm remoteip
```

**ENABLECALLBACK:** Configure this parameter if you want to call back another GSM modem.

```
cli>config network pcmcia 2 gsm enablecallback yes callbacknum 5552255
```

3. Activate the configuration.

```
cli>config runconfig
```

4. Save the configuration.

```
cli>config savetoflash
```

5. Exit the CLI mode.

To exit the CLI mode and return to the CPS’s shell, type the following command:

```
cli> quit
```
CDMA Card Configuration

CDMA cards are modem cards that enable the CPS to receive a dial-in connection and callback using the “ppp” protocol.

CDMA card configuration is made by setting the following parameters, which are similar to the parameters set in a modem card.

**Table 7-1: CDMA configuration parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local and Remote IP addresses (optional)</td>
<td>IP addresses used by “ppp” connection and set in /etc/ppp/options.ttyXX file, where XX is the serial port being configured. The syntax is <em>local_IP:<em>remote_</em>IP</em>.</td>
</tr>
<tr>
<td>Phone number (optional)</td>
<td>This number will be used by the callback feature when activated and set in /etc/mgetty/login.config file.</td>
</tr>
<tr>
<td>Speed</td>
<td>This parameter defines the speed the CPS uses to access the card. The parameter is set in the /etc/mgetty/mgetty.config file.</td>
</tr>
<tr>
<td>Additional Initialization (optional)</td>
<td>Set an additional initialization parameter to be sent to the card. There is a default command sequence to initialize the card, but if an additional initialization command is required by the card, it can be added using the feature. The command sequence is set in the /etc/mgetty/mgetty.config file.</td>
</tr>
</tbody>
</table>
VI method

1. In `/etc/mgetty/mgetty.config`, add this entry (xx is the serial port number that will be assigned to the CDMA card):

```
port ttyxx
    speed 57600
    data-only y
    init-chat " \\
    \d\d\d+++\\d\\d\dATZ OK AT$QCVAD=4 OK
```

2. In `/etc/pcmcia/serial.opts`, add this entry:

```
*,0,*)
INFO="Modem Slot 1 Setup"
LINK="/dev/modem"
INITCHAT="- \d\d\d+++\d\d\d\datz OK"
INITTAB="/sbin/mgetty"
start_fn () { return; }
stop_fn () { return; }
NO_CHECK=n
NO_FUSER=n
```

3. If configuring a local and remote IP, modify `local_IP:remote_IP` entry in `/etc/ppp/options.ttyXX` file.

4. To enable the callback feature, add the following entry to `/etc/mgetty/login.config`:

```
PSEUDO_CB_NAME - - /sbin/callback -S PHONE (PSEUDO_CB_NAME=cbuser)
```

5. At the end of the `login.config` file there is a line similar to the following:

```
* - - /bin/login @
```

Enter the following command before the preceding line:

```
<pseudo callback name>- - /sbin/callback -S <phone number of the client>
```

In the following example, 'call' is the pseudo callback name and '5551212' is the number to dial back:

```
call - - /sbin/callback -S 5551212
```
**CLI method**

To configure a CDMA PCMCIA card using the CLI method:

1. Open the CLI interface by issuing the command:
   
   ```
   # CLI
   ```

2. Configure the CDMA parameters.
   
   - To configure speed:
     
     ```
     cli>config network pcmcia <slot> cdma speed <speed>
     ```
   - To configure local IP and remote IP to establish a PPP connection:
     
     ```
     cli>config network pcmcia <slot> cdma localip <ip_address>
     cli>config network pcmcia <slot> cdma remoteip <ip_address>
     ```
   - To enable the callback option:
     
     ```
     cli>config network pcmcia <slot> cdma enablecallback <yes> callbacknum <number>
     ```
   - To include additional initialization commands:
     
     ```
     cli>config network pcmcia <slot> cdma addinit <command>
     ```

3. Activate the configuration.

   ```
   cli>config runconfig
   ```

4. Save the configuration.

   ```
   cli>config savetoflash
   ```

5. Exit the CLI mode.

   To exit the CLI mode and return to the CPS’s shell, type the following command:

   ```
   cli> quit
   ```
ISDN PC Cards

You can establish synchronous PPP connections with ISDN cards. The ipppd is the daemon that handles the synchronous PPP connections.

VI method

How to configure dial in.

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. If your system requires RADIUS or local authentication, you must also create a user in /etc/ppp/pap-secrets. If you do not want to repeat the entire user database from the RADIUS server, use an asterisk "*" as the user in /etc/ppp/pap-secrets:
   *
   *
   *
   *
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 example, "+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.d/login should be configured also.
3. Run saveconf to save your changes to the flash.
4. Insert the ISDN card or restart the ISDN script:
   • If the ISDN card is already inserted, restart the ISDN script.
      If the card was already inserted, you must restart the ISDN script to reload any changed configuration. To restart the script, issue the following commands, then proceed to step 5:
      
      # /etc/pcmcia/isdn stop ippp0
      # /etc/pcmcia/isdn start ippp0
   • If the ISDN card is not inserted, insert it now.
      ipppd is started automatically. Proceed to step 5.
5. You can dial from the remote system to the CPS, and get a PPP connection.
6. To hang up the connection from the CPS side, issue the following command:
   
   # isdnctrl hangup ippp0
How to configure dial out.

1. Create a user.
   Create a user in /etc/ppp/pap-secrets if you want PAP authentication, or in /etc/ppp/chap-secrets if you want CHAP authentication.

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.

3. Run saveconf to save your changes to the flash.

4. Insert the ISDN card or restart the ISDN script:
   • If the ISDN card is not inserted, insert the card.
     ipppd is started automatically. Proceed to step 5.
   • If the ISDN card is already inserted, restart the script.
     If the card was already inserted, you must restart the ISDN script to reload any changed configuration. To restart the script, issue the following commands, then proceed to step 5:

     ```
     # /etc/pcmcia/isdn stop ippp0
     # /etc/pcmcia/isdn start ippp0
     ```

5. To dial out, issue the command:
   # isdnctrl dial ippp0

6. To hang up the connection from the CPS side, issue the following command:
   # isdnctrl hangup ippp0

Establishing a callback with your ISDN PC card (method one)

The ISDN card in the CPS 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 CPS as callback server (CPS Setup). Part Two is the configuration of a Windows 2000 Professional computer as callback client (Callback Client).

**CPS setup (Callback Server).**

1. Change the parameters in /etc/pcmcia/isdn.opts to fit your environment.

2. Set the callback number in DIALOUT_REMOTE NUMBER:
DIALOUT_REMOTENUMBER="8358662" # Remote phone that you want to dial to

3. If your ISDN line does not support caller ID, skip to step 4. If your ISDN line supports caller identification (caller ID), it is recommended that you also configure the DIALIN_REMOTENUMBER and enable secure calls.

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.

4. Make sure the CALLBACK is set to “in” in the /etc/pcmcia/isdn.opts file.

CALLBACK="in" # "in" will enable callback for incoming calls.

5. Uncomment the line with user "mary" in /etc/ppp/pap-secrets.

6. Save the changes to flash.

# saveconf

7. Activate the changes by stopping and starting the ISDN script:

# /etc/pcmcia/isdn stop ippp0
# /etc/pcmcia/isdn start ippp0

Windows 2000 Professional configuration (Callback Client).

1. From the Control Panel, select “User and Passwords”, then create a user “mary” with the password “marypasswd”.

2. Create a dial-up connection that uses “Modem—AVM ISDN Internet (PPP over ISDN) (AVMISDN1)”. To create a dial-up connection:

   a. From the Start menu, select Settings>Network and Dial-up Connections>Make New Connection.

   b. Select “I want to set up my Internet connection manually, or I want to connect through a local area network”, then select “I connect through a phone line and a modem”.

   c. Select the “AVM ISDN Internet (PPP over ISDN)” modem.
d. Type the phone number you dial to connect to the CPS, enter mary as User name, and enter marypasswd as password.

e. After creating this dial-up connection, click on the Properties of this dial-up, select the “Options” panel, and change the redial attempts to 0.

3. Accept incoming connections.
   a. From the Start menu, select Settings>Network and Dial-up Connections>Make New Connection.
   b. Select “Accept incoming connections”.
   c. Select AVM ISDN Internet (PPP over ISDN).
   d. Select “Do not allow virtual private connections”.
   e. Select the user “mary”, then select “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.
   f. After any change in the Incoming Connection Properties, restart Windows to apply the changes.

When the Callback Server and Callback Client configuration is complete, you can dial from Windows to the CPS. From the Start menu, select Settings, then select “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.

Click Cancel. In a few seconds, the CPS will call you back, and you will see the connection icon in the task bar.

**Establishing a callback with your ISDN PC card (method two)**

The preceding section explained how to configure callback at the D-Channel level. The advantage of having callback at D-Channel level is that it works independent of the Operating System on the client side. The disadvantage is that the callback call happens before the authentication phase in PPP. The only security is that only calls from predefined phone numbers are accepted.

To enhance security, this section explains another way to configure callback with the CPS. The following steps work when the remote side is a UNIX machine, not a Windows machine. The callback call will happen after the PPP authentication is successful.
CPS Setup (Callback Server).

1. Change the parameters in `/etc/pcmcia/isdn.opts` file to fit your environment.
   
a. Set the callback number in `DIALOUT_REMOTENUMBER`.

   ```
   DIALOUT_REMOTENUMBER="8358662" # Remote phone you want to dial to
   ```

   b. If your ISDN line does not support caller ID, skip to step c. If your ISDN line supports caller ID, it is recommended that you also configure the `DIALIN_REMOTENUMBER` and enable secure calls.

   ```
   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.
   ```

   c. Set the desired IP addresses for local and remote machines.

   d. Set `DIALIN` to “yes”.

   ```
   DIALIN="yes" # "yes" if you want dial in, "no" if you want dial out
   ```

   e. Make sure the `CALLBACK` parameter is disabled.

   ```
   CALLBACK="off" # "off" = callback disabled.
   ```

   f. Add the user that will callback the client in `DIALIN_AUTHENTICATION`.

   ```
   DIALIN_AUTHENTICATION="auth login user mary"
   ```

2. Make sure `/etc/pam.d/` has the configuration files you want (e.g., RADIUS). This step is only required if you are using “auth login” in `DIALIN_AUTHENTICATION`. When “auth login” is used, `/etc/pam.d/` defines which authentication will be used.

3. Add the user "mary" in `/etc/ppp/pap-secrets`.

4. Uncomment lines in `/etc/ppp/auth-up`.

5. Save changes to flash:

   ```
   # saveconf
   ```

6. Activate the changes by stopping and starting the ISDN script:

   ```
   # /etc/pcmcia/isdn stop ippp0
   # /etc/pcmcia/isdn start ippp0
   ```
Linux (Callback Client).

1. Configure the ipppd to have user “mary” and PAP authentication.
2. Dial to the CPS:
   
   # isdnctrl dial ippp0
3. As soon the CPS authenticates the user mary, the CPS will disconnect and callback.

CLI method—ISDN PCMCIA

To configure an ISDN PCMCIA card using the CLI:

1. Open the CLI interface by issuing the command:

   # CLI

2. Configure ISDN parameters.

   Depending on how you will use the ISDN card, some parameters do not need to be configured.

   **Configurable parameters:**

   - **LOCALIP/REMOTEIP:** Configure this parameter if you want to establish a PPP connection. The first command below defines the unit’s local IP address, and the second command defines the other side IP address.

     cli>config network pcmcia 2 isdn localip
     cli>config network pcmcia 2 isdn remoteip

   - **ENABLECALLBACK:** Configure this parameter if you want to call back another ISDN modem.

     cli>config network pcmcia 2 isdn enablecallback yes callbacknum 55552244

3. Activate the configuration.

   cli>config runconfig

4. Save the configuration.

   cli>config savetoflash

5. Exit the CLI mode.

   To exit the CLI mode and return to CPS’s shell, type the following command:

   cli>quit
7.3 Media Cards

Media cards (compact flash, hard drives) are small memory cards with a capacity up to 5 Gigabytes. They can be used like a normal hard disk drive using IDE. With an inexpensive adapter, you can use CF cards in PCMCIA slots. (The PCMCIA and CF card standard are the same, though the pin layout and the socket are different.) Small PCMCIA hard drives are also available, e.g., a drive from Toshiba with a capacity of 5GB (Toshiba MK5002MPL). CF card support can be used in the CPS for storing files. This would be especially useful to save the configuration, for example. 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 CPS.
- No dependency on an NFS server that may fail.

How it works

When you insert an adapter with a CF card or a PCMCIA hard drive, an IDE device appears. This can be mounted, e.g. by:

```
# mkdir /mnt/ide
# mount /dev/hda1 /mnt/ide
```

Except for the ext2 filesystem, the VFAT file system 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:

```
# echo ",,0x0e" | sfdisk /dev/hda
# mkdosfs /dev/hda1
```

For ext2 filesystem:

```
# echo ",,L" | sfdisk /dev/hda
```

The "mke2fs" utility is the system creator for ext2 filesystems, and can be run as follows:

```
# mke2fs /dev/hda1
```

In addition, a utility to create or partition the CF has been added. For this, use the program sfdisk. 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/<hdx>
```
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.

**VI method—configuration**

1. Insert the card.
2. Compact flash mounts automatically.

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:

```bash
# 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 7.4: /etc/pcmcia/ide.opts file
```

The following parameters can be changed:

- `DO_FSTAB`—If set to 'y', an entry in `/etc/fstab` will be created. By default, this parameter is 'n' if it is 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. By default, this parameter is 'n' in the /etc/pcmcia/ide.opts file.

• **DO_MOUNT**—If set to 'y', the card will be mounted automatically upon insertion. By default, this parameter is 'n' if it is 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. By default, these parameters are '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, for example, set PARTS="1 3 4" and add entries for the case statement such as:

```bash
*,*,*,3)
# settings for partition 3
;;
*,*,*,4)
# settings for partition 4
;;
```

To give different configurations for slot 0 and 1, the second parameter in the case statement can be used. For example:

```bash
*,0,*,1)
# settings for slot 0
;;
*,1,*,1)
# settings for slot 1
;;
```

3. **Save the configuration.**

To save any configuration done in the /etc/pcmcia/ide.opts file, run the command:

```
# saveconf
```

**WARNING:** Before removing the media PCMCIA card from the CPS you MUST run “cardctl eject”. Otherwise, data might not be correctly written to disk and may result in corruption of the media. Correct operation of the CPS is not guaranteed if `eject` is not executed.
CLI method—Media Cards PCMCIA

Mounting PCMCIA storage devices using the CLI is extremely simple. Just follow the steps below:

1. Open the CLI interface by issuing the command:
   
   # CLI

2. Enable the Compact Flash or mini hard drive.
   
   In the following example, the PCMCIA card is placed on slot 1 of the unit. Run the command:
   
   cli>config network pcmcia 1 cflash enable yes

   To enable data buffering on this device run the command:
   
   cli>config network pcmcia 2 cflash databuf yes

3. Activate the configuration.
   
   cli>config runconfig

4. Save the configuration.
   
   cli>config savetoflash

5. Check the configuration
   
   The device should be mounted under the /mnt/ide directory.

6. Exit the CLI mode.
   
   To exit the CLI mode and return to CPS’s shell, type the following command:
   
   cli> quit

**WARNING:** Before removing the media PCMCIA card from the CPS you MUST run “cardctl eject”, from the shell prompt (not possible using the CLI), otherwise data might not be correctly written to disk and may result in corruption of the media. Correct operation of the CPS is not guaranteed if eject is not executed.
How to Save/Load Configuration to/from CF/IDE

You can save and restore the configuration file to/from any PCMCIA-connected file system. By configuring the `saveconf` utility, you can enable the CPS to save the configuration to a PCMCIA-mounted file system and define the type of the configuration saved in the device.

The administrator can define this feature in two ways:

- **default**—The system will apply the configuration in the storage device to the current running system after reboot
- **replace**—The system will use the configuration in the storage device to replace the existing one in the internal flash of the CPS.

The PCMCIA cards are detected when they are inserted in the slot. If the card is a storage device (Compact Flash or IDE), the system mounts the file system ext2 in the `/mnt/ide` directory.

During the boot time, before the call of the `restoreconf` from the internal flash, the system checks for the existence of a config file in the `/mnt/ide/proc/flash/script` directory. If the DEFAULT flag is set, the system will use the file in the storage device as the config file. If the DEFAULT flag is not set, the system will use the file in internal flash as the config file.

**Saveconf Utility.** The syntax is:

```bash
# saveconf sd [default | replace]
```

The `saveconf` utility allows you to save configuration to PCMCIA mounted file system and will define the type of the configuration saved in the device. The administrator can define the following types:

- **default**: the configuration in the storage device should be applied to the current running system after reboot
- **replace**: the configuration in the storage device should be used to replace the existing in the internal flash of the CPS.

The `saveconf` utility creates one file in the storage device to save the default and replace flags. The filename is: `/mnt/ide/proc/flash/storageOptions` and it can contain the words *DEFAULT* and/or *REPLACE*.

**Restoreconf Utility.** The syntax is:

```bash
# restoreconf sd [default | replace]
```

The `restoreconf` utility can read the configuration from a storage device mounted file system and perform the following actions:
**CLI method: backupconfig**

To save or restore the configuration to or from a PCMCIA media card:

1. Open the CLI interface by issuing the command:
   
   ```
   # CLI
   ```

2. Save the configuration to a Storage Device:
   
   ```
   cli> administration backupconfig saveto sd [default] [replace]
   ```

3. Restore the configuration from a Storage Device:
   
   ```
   cli> administration backupconfig loadfrom sd
   ```

4. Exit the CLI mode.
   
   To exit the CLI mode and return to the CPS’s shell, type the following command:

   ```
   cli> quit
   ```

**Generic Dial-Out**

This feature allows an application to connect from a central office to a remote location to inquire system status. The remote system can then send asynchronous alarm notification to the application at the central office.

The connection between the central office and the remote location can be performed using TCP/IP over an Ethernet network (In-Band), or through GSM/GPRS and CDMA/1xRTT profiles (Out-of-Band).

Currently “dial-out” application is supported. Use the `/etc/generic-dial.conf` file to configure dial-out PPP connections through GPRS and 1xRTT profiles. The following example illustrates a dial-out configuration with a wireless PPP connection. The text in bold type face indicates edited text.
The tail of the file /etc/generic-dial.conf

#begin dial-out testApp  
#  
#inPort.name            InPort   
#inPort.device         /dev/ttyS1  
#  
#outPort.name            OutPort   
#outPort.pppcall        wireless  
#outPort.remote_ip      200.246.93.87  
#outPort.remote_port    7001  
#  
#appl.retry            7  
#  
#end dial-out

The content of the file /etc/ppp/peers/wireless

nodetach
#debug
/dev/ttyM1
57600
crtscts
lock
noauth
#nomagic
user claro
show-password
noipdefault
defaultroute
ipcp-accept-local
ipcp-accept-remote
noproxyarp
novj
novjccomp
lcp-echo-interval 0
connect '/usr/local/sbin/chat -v -t3 -f /etc/chatscripts/wireless'
Configuring the generic-dial.conf file

The file "/etc/generic-dial.conf" contains sections that corresponds to instances of generic-dial applications. For example:

# begin <application-type> [instanceID]
#....
#....
# end <application-type>

In the preceding example, [instanceID] is an optional string to identify a particular instance, and <application type> corresponds to specific applications built over the infrastructure. Within each application, the parameters needed to create the objects for that specific instance are inserted.

Configuring Generic Dial-Out

1. To enable the generic dial-out application, configure the desired ports with the protocol parameter in /etc/portslave/pslave.conf.

   For example:
   
   s<N>.protocol generic_dial
   where <N> is the port number.

2. To enable dial-out for the ports chosen in pslave.conf, configure the file /etc/generic-dial.conf as described in the following table.

   Table 7-2: Dial-out parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>begin &lt;dial-out&gt; [instance-id]</td>
<td>Begins the dial-out application. Optionally, specify a name for the particular instance.</td>
</tr>
<tr>
<td>inPort.name &lt;name&gt;</td>
<td>A label for the incoming port to be used in log messages.</td>
</tr>
<tr>
<td>inPort.device &lt;/dev/ttyXX&gt;</td>
<td>The modem device used for this interface.</td>
</tr>
<tr>
<td>inPort.speed &lt;9600&gt;</td>
<td>Connection speed.</td>
</tr>
<tr>
<td>inPort.datasize &lt;8&gt;</td>
<td>The number of data bits.</td>
</tr>
<tr>
<td><strong>Parameter</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------------------------------------------------</td>
</tr>
<tr>
<td><code>inPort.parity</code> [ none</td>
<td>even</td>
</tr>
<tr>
<td><code>inPort.stopbits &lt;1&gt;</code></td>
<td>The number of stop bits.</td>
</tr>
<tr>
<td><code>inPort.flowctrl</code> [ none</td>
<td>hw</td>
</tr>
<tr>
<td><code>outPort.name &lt;name&gt;</code></td>
<td>A label for the outgoing port to be used in log messages.</td>
</tr>
<tr>
<td><code>outPort.pppcall &lt;filename&gt;</code></td>
<td>Name of file from which the pppd reads options. The file is located at <code>/etc/ppp/peers/filename</code>.</td>
</tr>
<tr>
<td><code>outPort.remote_ip &lt;IP address&gt;</code></td>
<td>IP address of remote work station to be connected to this interface.</td>
</tr>
<tr>
<td><code>outPort.remote_port &lt;port&gt;</code></td>
<td>Remote TCP port for connections from this interface.</td>
</tr>
<tr>
<td><code>outPort.connection</code> [ permanent</td>
<td>on_demand ]</td>
</tr>
<tr>
<td></td>
<td>• permanent—always connected.</td>
</tr>
<tr>
<td></td>
<td>• on_demand—connects only when data enters through the serial port.</td>
</tr>
<tr>
<td><code>outPort.timeout &lt;timeout&gt; (seconds)</code></td>
<td>Specify the inactivity time, in seconds, after which the connection is dropped. Any value other than zero enables the timeout.</td>
</tr>
<tr>
<td><code>appl.retry &lt;interval&gt; (minutes)</code></td>
<td>Specify the time to wait before reconnecting after a connection failure.</td>
</tr>
<tr>
<td><code>end &lt;dial-out&gt;</code></td>
<td>Ends the dial-out application.</td>
</tr>
</tbody>
</table>
3. Configure the PPP options (pppd) in /etc/ppp/peers/<name>, where <name> is the same as the <filename> variable specified in the outPort.pppcall <filename> parameter in /etc/generic-dial.conf.

The following example shows the /etc/ppp/peers/wireless file.

In this example, the “connect” script initiates the connection. The file “wireless” executes using the “chat” automated modem communication scrip with the parameters -v (verbose mode), -t (timeout), and -f (read the chat script from the /etc/chatscripts/wireless file).

```bash
[root@CPS root]# more /etc/ppp/peers/wireless
nodetach
#debug
/dev/ttyM1
57600
crtscts
lock
noauth
#nomagic
user claro
show-password
nolpdefault
defaultroute
ipcp-accept-local
ipcp-accept-remote
noproxyarp
novj
novjccomp
lcp-echo-interval 0
connect '/usr/local/sbin/chat -v -t3 -f /etc/chatscripts/wireless'
INITCHAT="- \d\d\d++\d\d\d\datz OK at+cpin=1111 OK"
```

4. Edit the /etc/pcmcia/serial.opts file:

a. If the SIM card (GSM) needs a PIN, uncomment the following line and replace 1111 with the PIN.

```bash
INITCHAT="- \d\d\d++\d\d\d\datz OK at+cpin=1111 OK"
```

b. To inactivate mgetty on the specified port so that the port will be directly controlled by the pppd application, comment out the following line.

```bash
#INITTAB="/sbin/mgetty"
```
5. Activate the function to automatically restart the dial-out application after reboot.

   Edit the following parameter in the `/etc/daemon.d/gendial.sh` file to enable the "automatically established" feature. The script restarts the dial-out function after a reboot.
   
   a. Set the parameter as described below.
      
      ```
      ENABLE = YES
      ```
   
   b. Save the `gendial.sh` configuration file by issuing the following command in CPS.
      
      ```
      [root@CPS root]# saveconf
      ```

6. To activate the dial-out function, issue the following command.

   **NOTE:** It is not necessary to reboot the CPS to activate the dial-out function. You can do this by restarting the GDF daemon.

   ```
   [root@CPS root]# daemon.sh restart GDF
   ```

   A message similar to the following displays, confirming the GDF daemon restart.

   ```
   [root@CPS root]# Sep 23 18:06:10 src_dev_log@CPS showlogmsg: /bin/daemon.sh: CONFIG: Network daemon [generic-dial] started
   ```

   The default route is not replaced in the static router table. The following message displays.

   ```
   [root@CPS root]# Sep 23 18:06:17 src_dev_log@CPS pppd[1028]: not replacing existing default route to eth0 [172.20.0.1]
   ```

7. To change a static route or create a new one:
   
   a. Edit the parameters in the `/etc/network/st_routes` file.
   
   b. Activate the new routes by issuing the following command:
      
      ```
      #> runconf
      ```
   
   c. Save the new configuration to flash.
      
      ```
      #> saveconf
      ```
   
   d. Check the routes by issuing the following command.
      
      ```
      #> route -n
      ```
Chapter 8: Profile Configuration

This chapter presents 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, pslave.cas, pslave.ts, and pslave.ras) in the /etc/portslave directory.

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

8.1 The pslave.conf File

The pslave.conf file is the main configuration file (/etc/portslave/pslave.conf) that contains most product parameters and defines the functionality of the CPS.

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.** Once you open this file, you can do a find for each of these parameters in vi by typing:

```
/ <your string>
```

To search the file downward for the string specified after the /.
pslave.conf Common Parameters

The tables below present all parameters and their descriptions. The first table presents parameters that are common for any profile. The second, third, and fourth tables define specific parameters for CAS, TS, and Dial-in profiles, respectively.

### Table 8-1: /etc/portslave/pslave.conf common parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Factory Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>conf.dhcp_client</td>
<td>Defines the DHCP client operation mode.</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0—DHCP disabled</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1—DHCP active</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2—DHCP active, and the unit saves the last IP assigned by the DHCP</td>
<td></td>
</tr>
<tr>
<td></td>
<td>server in flash</td>
<td></td>
</tr>
<tr>
<td>conf.eth_ip_alias</td>
<td>Secondary IP address for the Ethernet interface (needed for clustering feature).</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td><strong>NOTE:</strong> This parameter is inactive by default. To activate, uncomment the parameter and set the value.</td>
<td></td>
</tr>
<tr>
<td>conf.eth_mask_alias</td>
<td>Mask for the secondary IP address.</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td><strong>NOTE:</strong> This parameter is inactive by default. To activate, uncomment the parameter and set the value.</td>
<td></td>
</tr>
<tr>
<td>conf.rlogin</td>
<td>Defines the location of the rlogin utility</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>NOTE:</strong> This is a parameter specific to the TS profile.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>/usr/local/bin/rlogin-radius</td>
<td></td>
</tr>
<tr>
<td>conf.facility</td>
<td>The local facility sent to syslog-ng from PortSlave.</td>
<td>7</td>
</tr>
</tbody>
</table>
### Table 8-1: /etc/portslave/pslave.conf common parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Factory Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>conf.group</td>
<td>Group users, to simplify the configuration of the parameter all.users. This parameter can be used to define more than one group. <strong>NOTE:</strong> This parameter is inactive by default. To activate, uncomment the parameter and set the value.</td>
<td>0</td>
</tr>
<tr>
<td>conf.eth_ip</td>
<td>The IP address of the Ethernet interface. This parameter is configured in Chapter 4: Network. This parameter, along with the next two, is used by the <code>cy_ras</code> program to OVERWRITE the file <code>/etc/network/ifcfg_eth0</code> as soon as the command “runconf” is executed. The file <code>/etc/network/ifcfg_eth0</code> should not be edited by the user unless the <code>cy_ras</code> configuration is not going to be used. <strong>NOTE:</strong> This parameter is inactive by default. To activate, uncomment the parameter and set the desired value.</td>
<td>0 (IP address received from DHCP Server)</td>
</tr>
<tr>
<td>conf.eth_mask</td>
<td>The mask for the Ethernet network. <strong>NOTE:</strong> This parameter is inactive by default. To activate, uncomment the parameter and set the value.</td>
<td>0 (IP mask received from DHCP Server)</td>
</tr>
<tr>
<td>conf.eth_mtu</td>
<td>The Maximum Transmission Unit size, which determines whether or not packets should be broken up.</td>
<td>1500</td>
</tr>
</tbody>
</table>
conf.lockdir The lock directory, which is /var/lock for the CPS. This parameter should not be changed unless the user decides to customize the operating system.

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.

Table 8-1: /etc/portslave/pslave.conf common parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Factory Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>conf.lockdir</td>
<td>The lock directory, which is /var/lock for the CPS. This parameter should not be changed unless the user decides to customize the operating system.</td>
<td>/var/lock</td>
</tr>
<tr>
<td>all.dcd</td>
<td>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.</td>
<td>0</td>
</tr>
</tbody>
</table>
### Table 8-1: /etc/portslave/pslave.conf common parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Factory Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>all.users</td>
<td>Restricts access to ports by user name (only the listed users can access the port or, if you use the exclamation point “!”), all users except the listed users 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. Do not place a comma 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. <strong>NOTE:</strong> These are common users, not administrators. <strong>NOTE:</strong> This parameter is inactive by default. To activate, uncomment the parameter and set the value.</td>
<td>null</td>
</tr>
</tbody>
</table>
| all.issue | This text determines the format of the login banner that is issued when a connection is made to the CPS. \n represents a new line and \r represents a carriage return. Expansion characters can be used here. The default parameter is: \r\n\r
Welcome to Console Port Server %h port %p \n\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 8-1: /etc/portslave/pslave.conf common parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Factory Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>all.media</td>
<td>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</td>
<td>rs232</td>
</tr>
<tr>
<td>all.netmask</td>
<td>Defines the network mask for the serial port.</td>
<td>255.255.255.255</td>
</tr>
<tr>
<td>all.mtu</td>
<td>Defines the maximum transmit unit</td>
<td>1500</td>
</tr>
<tr>
<td>all.mru</td>
<td>Defines the maximum receive unit</td>
<td>1500</td>
</tr>
<tr>
<td>all.sysutmp</td>
<td>Defines whether portslave must write login records. Valid values are yes (1) or no (0).</td>
<td>1</td>
</tr>
<tr>
<td>all.pmtype</td>
<td>The name of the rPDU manufacturer.</td>
<td>APC PDU</td>
</tr>
<tr>
<td></td>
<td><strong>NOTE:</strong> This parameter is inactive by default. To activate, uncomment the parameter and set the value.</td>
<td></td>
</tr>
<tr>
<td>all.pmusers</td>
<td>List of the outlets each user can access.</td>
<td>null</td>
</tr>
<tr>
<td></td>
<td><strong>NOTE:</strong> This parameter is inactive by default. To activate, uncomment the parameter and set the value.</td>
<td></td>
</tr>
<tr>
<td>all.pmkey</td>
<td>The hotkey that identifies the power management command.</td>
<td>off</td>
</tr>
<tr>
<td></td>
<td><strong>NOTE:</strong> This parameter is inactive by default. To activate, uncomment the parameter and set the value.</td>
<td></td>
</tr>
</tbody>
</table>
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. For example:

```
all.sttyCmd -igncr -onlcr opost -icrnl
```

where:
- **-igncr**
  This command 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.

**NOTE:** This parameter is inactive by default. To activate, uncomment the parameter and set the value.

---

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Factory Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>all.sttyCmd</td>
<td>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. For example: all.sttyCmd -igncr -onlcr opost -icrnl</td>
<td>null</td>
</tr>
<tr>
<td><strong>-igncr</strong></td>
<td>This command tells the terminal not to ignore the carriage-return on input.</td>
<td></td>
</tr>
<tr>
<td><strong>-onlcr</strong></td>
<td>Do not map newline character to a carriage return or newline character sequence on output.</td>
<td></td>
</tr>
<tr>
<td><strong>opost</strong></td>
<td>Post-process output.</td>
<td></td>
</tr>
<tr>
<td><strong>-icrnl</strong></td>
<td>Do not map carriage-return to a newline character on input.</td>
<td></td>
</tr>
</tbody>
</table>

---
### all.utmpfrom

Allows the administrator to customize the field "FROM" in the login records (utmp file). It is displayed in the "w" command.

The default value is "%g:%P.%3.%4".

- %g — process id
- %P — Protocol
- %3 — Third nibble of remote IP
- %J — Remote IP

**NOTE:** A list of all expansion variables is available in the pslave.conf file.

### all.radnullpass

Defines whether access must be granted to users with null password in the Radius server.

Value: 0

### all.speed

The speed for all ports.

Value: 9600

### all.datasize

The data size for all ports.

Value: 8

### all.stopbits

The number of stop bits for all ports.

Value: 1

### all.parity

The parity for all ports.

Value: none

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Factory Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>all.utmpfrom</td>
<td>Allows the administrator to customize the field &quot;FROM&quot; in the login records (utmp file). It is displayed in the &quot;w&quot; command.</td>
<td>&quot;%g:%P.%3.%4&quot;</td>
</tr>
<tr>
<td>all.radnullpass</td>
<td>Defines whether access must be granted to users with null password in the Radius server.</td>
<td>0</td>
</tr>
<tr>
<td>all.speed</td>
<td>The speed for all ports.</td>
<td>9600</td>
</tr>
<tr>
<td>all.datasize</td>
<td>The data size for all ports.</td>
<td>8</td>
</tr>
<tr>
<td>all.stopbits</td>
<td>The number of stop bits for all ports.</td>
<td>1</td>
</tr>
<tr>
<td>all.parity</td>
<td>The parity for all ports.</td>
<td>none</td>
</tr>
</tbody>
</table>
Table 8-1: /etc/portslave/pslave.conf common parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Factory Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>all.authtype</td>
<td>Type of authentication used. (Configured in “Device Authentication” on page 39.) Authentication type options:</td>
<td>none</td>
</tr>
<tr>
<td></td>
<td>• none—No authentication.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• local—Authentication is performed using the /etc/passwd file.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• remote—Terminal profile only. The unit takes in a username but does not use it for authentication. Instead it passes the username to the remote server where it is used for authentication.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• radius—Authentication is performed using a Radius authentication server.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• TacacsPlus—Authentication is performed using a TacacsPlus authentication server.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• 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.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• 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.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• local/radius—Authentication is performed locally first, switching to Radius if unsuccessful.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• radius/local—Authentication is performed with Radius first, switching to local if unsuccessful.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• local/TacacsPlus—Authentication is performed locally first, switching to TacacsPlus if unsuccessful.</td>
<td></td>
</tr>
</tbody>
</table>
all.authtype  
(continued)

- **TacacsPlus/local**—Authentication is performed with TacacsPlus first, switching to local if unsuccessful.
- **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:** This parameter only controls the authentication required by the CPS. 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`, `socket_server`, or `socket_server_ssh`.  

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Factory Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>all.authtype</td>
<td>• TacacsPlus/local—Authentication is performed with TacacsPlus first, switching to local if unsuccessful.</td>
<td></td>
</tr>
<tr>
<td>all.authtype</td>
<td>• RadiusDownLocal—Local authentication is tried only when the Radius server is down.</td>
<td></td>
</tr>
<tr>
<td>all.authtype</td>
<td>• TacacsPlusDownLocal—Local authentication is tried only when the TacacsPlus server is down.</td>
<td></td>
</tr>
<tr>
<td>all.authtype</td>
<td>• kerberosDownLocal—Local authentication is tried only when the kerberos server is down.</td>
<td></td>
</tr>
<tr>
<td>all.authtype</td>
<td>• ldapDownLocal—Local authentication is tried only when the ldap server is down.</td>
<td></td>
</tr>
<tr>
<td>all.authtype</td>
<td>• NIS—All authentication types but NIS follow the format <code>all.authtype &lt;Authentication&gt; DownLocal</code> or <code>&lt;Authentication&gt;</code> (e.g. <code>all.authtype radius</code> or <code>radiusDownLocal</code> or <code>ldap</code> or <code>ldapDownLocal</code>, etc). NIS requires <code>all.authtype</code> to be set as local, regardless if it will be &quot;nis&quot; or its “Downlocal” equivalent. The service related to &quot;nis&quot; or its “Downlocal” equivalent would be configured in the <code>/etc/nsswitch.conf</code> file, not in the <code>/etc/portslave/pslave.conf</code> file.</td>
<td></td>
</tr>
<tr>
<td>all.authtype</td>
<td><strong>NOTE:</strong> This parameter only controls the authentication required by the CPS. The authentication required by the device to which the user is connecting is controlled separately.</td>
<td></td>
</tr>
<tr>
<td>all.break_sequence</td>
<td>This parameter is the string that is used to send a break to the TTY. It is only valid if TTY protocol is <code>socket_ssh</code>, <code>socket_server</code>, or <code>socket_server_ssh</code>.</td>
<td>~break</td>
</tr>
</tbody>
</table>
### Table 8-1: /etc/portsave/pslave.conf common parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Factory Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>all.break_interval</td>
<td>This parameter defines the break duration in milliseconds. It is valid if TTY protocol is <code>socket_ssh</code>, <code>socket_server</code>, <code>socket_server_ssh</code>, or <code>ssh-2 (client)</code>.</td>
<td>500</td>
</tr>
<tr>
<td>all.flow</td>
<td>This sets the flow control to <code>hardware</code>, <code>software</code>, or <code>none</code>.</td>
<td>none</td>
</tr>
</tbody>
</table>
Table 8-1: /etc/portslave/pslave.conf common parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Factory Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>all.protocol</td>
<td>Defines the protocol used to connect with the CPS. Valid values for each profile are:</td>
<td>socket_server</td>
</tr>
<tr>
<td></td>
<td>• CAS profile:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- <em>socket_server</em> when Telnet is used.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- <em>socket_ssh</em> when SSHv1 or SSHv2 is used.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- <em>socket_server_ssh</em> when Telnet or SSHv1 or SSHv2 is used.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- <em>raw_data</em> to exchange data in transparent mode. It is similar to “socket_server” mode, but without Telnet negotiation, breaks to serial ports, etc.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• TS profile:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- <em>login</em> requests username and password.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- <em>rlogin</em> receives username from the CPS and requests a password.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- <em>Telnet</em></td>
<td></td>
</tr>
<tr>
<td></td>
<td>- <em>SSHv1</em></td>
<td></td>
</tr>
<tr>
<td></td>
<td>- <em>SSHv2</em></td>
<td></td>
</tr>
<tr>
<td></td>
<td>- <em>socket_client</em></td>
<td></td>
</tr>
<tr>
<td></td>
<td>If the protocol is configured as Telnet or socket_client the socket_port parameter must be configured.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Bidirectional Telnet profile: <em>socket_server</em> (<em>CAS</em>), and <em>login</em> (<em>TS</em>)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• RAS profile: <em>slip</em>, <em>cslip</em>, <em>ppp</em>, <em>ppp_only</em></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Power Management: <em>ipdu</em></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Serial Printer: <em>lpd</em></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Billing profile: <em>billing</em></td>
<td></td>
</tr>
</tbody>
</table>
### Table 8-1: /etc/portslave/pslave.conf common parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Factory Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>all.web_WinEMS</td>
<td>Defines whether management of Windows Emergency Management Service is allowed from the Web.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>NOTE:</strong> This parameter is inactive by default. To activate, uncomment the parameter and set the value.</td>
<td></td>
</tr>
<tr>
<td>all.xml_monitor</td>
<td>A non-zero value activates XML monitoring. All XML data received from the port is captured and sent to syslog-ng with facility LOCAL (&lt;\text{DB_facility}&gt;) and priority INFO. The format of the message is &quot;XML_MONITOR (ttySx) [data]&quot;. XML tags are sent by Windows Server 2003 Emergency Management Services during boot or crash. More information about XML_MONITOR is available in: /etc/syslog-ng/syslog-ng.conf</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>NOTE:</strong> This parameter is inactive by default. To activate, uncomment the parameter and set the value.</td>
<td></td>
</tr>
<tr>
<td>all.translation</td>
<td>Defines whether or not to perform translation of function keys (e.g., F8 key) from one terminal type to VT-UTF8. Currently, only translation from xterm to VT-UTF8 is supported.</td>
<td></td>
</tr>
</tbody>
</table>
|                     | **NOTE:** This parameter is inactive by default. To activate, uncomment the parameter and set the value. | null
Table 8-1: /etc/portslave/pslave.conf common parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Factory Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>sX.pmoutlet</td>
<td>sX indicates the serial port number to which the rPDU hardware is connected. The pmoutlet part of the parameter indicates the outlet # on the rPDU hardware that manages the server/network equipment in question.</td>
<td>8</td>
</tr>
<tr>
<td>s1.tty</td>
<td>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. <strong>NOTE:</strong> This parameter is disabled by default. To activate, uncomment the parameter.</td>
<td>disabled</td>
</tr>
</tbody>
</table>

pslave.conf CAS (Console Access Server) Parameters

In addition to the preceding `pslave.conf` common parameters, which are common to all local and remote-access scenarios, you can configure additional CAS features with the parameters given in the following tables. Many of the parameters are unique to CAS, but some, as noted in the following tables, also apply to TS and Dial-in port profiles.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Factory Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>conf.nfs_data_buffering</td>
<td>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.). <strong>NOTE:</strong> This parameter is inactive by default. To activate, uncomment the parameter and set the value.</td>
<td>null</td>
</tr>
<tr>
<td>conf.DB_facility</td>
<td>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 (For more information, see “Syslog-ng” on page 137).</td>
<td>7</td>
</tr>
</tbody>
</table>
### Table 8-2: CAS specific parameters for the pslave.conf

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Factory Configuration</th>
</tr>
</thead>
</table>
| conf.nat_clustering_ip | IP address of any CPS 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 IP address as that of the master box in the chain.  

**NOTE:** This parameter is inactive by default. To activate, uncomment the parameter and set the value.                                                                 | 0                     |
| all.ipno               | The default IP address of the CPS's serial ports. The plus sign “+” 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.  

**NOTE:** This parameter is inactive by default. To activate, uncomment the parameter and set the value.                                                                 | 0                     |
| all.netmask            | Defines the network mask for the serial port.                                                                                                                                                                | 255.255.255.255        |
**all.DTR_reset** Specifies the behavior of the DTR signal in the serial port. If this parameter is 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 than or equal to 100 specifies 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.

**all.break_sequence** The string that is used to send a break to the TTY. It is only valid if the TTY protocol is `socket_ssh`, `socket_server`, or `socket_server_ssh`.

**all.break_interval** Defines the break duration in milliseconds. This parameter is valid if the TTY protocol is `socket_ssh`.

**all.modbus_smode** Communication mode through the serial ports. This parameter is meaningful only when the 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 this parameter is not configured, ASCII mode will be assumed.

**NOTE:** This parameter is inactive by default. To activate, uncomment the parameter and set the value.
Table 8-2: CAS specific parameters for the pslave.conf

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Factory Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>all lf suppress</td>
<td>Telneting (from DOS) from an OS such as Windows 98 produces an extra line feed, so two prompts appear whenever you press Enter. When this parameter is set to 1, line feed suppression is active, so the extra prompt is eliminated. When set to 0 (default), line feed suppression is not active.</td>
<td>0</td>
</tr>
</tbody>
</table>
| 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 sent back to the tty. This parameter works only when there is no session connected to the port. If this parameter is uncommented and a string of bytes is set, matching occurs whenever there is not a session established to the port. If this parameter is commented, then no checking and matching occurs.  

**Note:** This parameter is inactive by default. To activate, uncomment the parameter and set the value.                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|                            | null                                                                                                                                                                                                       | null                  |
Table 8-2: CAS specific parameters for the pslave.conf

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Factory Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>all.auto_answer_output</td>
<td>This parameter works in conjunction with <code>all.auto_answer_input</code>. It allows you to configure a string that is sent back to the remote server whenever the incoming data remote server matches with <code>all.auto_answer_input</code>. This parameter works only when there is no session connected to the port. If this parameter is commented, then nothing will be sent back to the remote server even if <code>all.auto_answer_input</code> is uncommented. If this parameter is uncommented and if <code>all.auto_answer_input</code> is also uncommented, then the configured string will be sent back to the remote server. <strong>NOTE:</strong> This parameter is inactive by default. To activate, uncomment the parameter and set the value.</td>
<td>null</td>
</tr>
<tr>
<td>all.poll_interval</td>
<td>Valid only for protocols <code>socket_server</code> and <code>raw_data</code>. When not set to zero, this parameter sets the wait time (in milliseconds) for a TCP connection keep-alive timer. If no traffic passes through the CPS for this period of time, the CPS will send a line status message to the remote device to see if the connection is still up. If this parameter is not configured, 1000 ms is assumed. If set to zero, line status messages will not be sent to the socket client. <strong>NOTE:</strong> This parameter is inactive by default. To activate, uncomment the parameter and set the value.</td>
<td>1000</td>
</tr>
</tbody>
</table>
Table 8-2: CAS specific parameters for the pslave.conf

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Factory Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>all.socket_port</td>
<td>In the CAS profile, this parameter defines an alternative labeling system for the CPS ports. The plus sign “+” 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 of how this could be used is in the case of all.protocol or s&lt;n&gt;.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.</td>
<td>7001+</td>
</tr>
</tbody>
</table>
A non-zero value activates data buffering (local or remote, according to what was configured in the parameter `conf.nfs_data_buffering`). See “Data Buffering” on page 14. If local data buffering is configured, a file is created on the CPS; 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 is used, this parameter defines the maximum file size (in bytes). If remote, this parameter is used as 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 is 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.

### Table 8-2: CAS specific parameters for the pslave.conf

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Factory Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>all.data_buffering</td>
<td>A non-zero value activates data buffering (local or remote, according to what was configured in the parameter <code>conf.nfs_data_buffering</code>). See “Data Buffering” on page 14. If local data buffering is configured, a file is created on the CPS; 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 is used, this parameter defines the maximum file size (in bytes). If remote, this parameter is used as 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 is 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.). <strong>Size is in bytes, not kilobytes.</strong> See Data Buffering for details.</td>
<td>0</td>
</tr>
</tbody>
</table>
all.DB_mode When configured as cir for circular format, the buffer works like a circular 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. When configured as 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 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 the buffer is 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. When the user exits the session, linear data buffering resumes. If all.flow or s<n>.flow is set to none, linear buffering is not possible because there is no way to stop reception through the serial line. Default is cir.

Table 8-2: CAS specific parameters for the pslave.conf

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Factory Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>all.DB_mode</td>
<td>When configured as cir for circular format, the buffer works like a circular file at all times. The file is overwritten whenever the limit of the buffer size (as configured in all.data_buffering or s&lt;n&gt;.data_buffering) is reached. When configured as 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&lt;n&gt;.flow is set) is issued automatically to the remote device so 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 the buffer is 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. When the user exits the session, linear data buffering resumes. If all.flow or s&lt;n&gt;.flow is set to none, linear buffering is not possible because there is no way to stop reception through the serial line. Default is cir.</td>
<td>cir</td>
</tr>
</tbody>
</table>
Table 8-2: CAS specific parameters for the pslave.conf

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Factory Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>all.DB_timestamp</td>
<td>Records the time stamp in the data buffering file (1) or does not record it (0). If this parameter 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. Then the accumulated data will be recorded in the data buffering file along with the current time. The parameter all.data_buffering has to be configured with a non-zero value for this parameter to be meaningful.</td>
<td>0</td>
</tr>
<tr>
<td><strong>NOTE:</strong> This parameter is inactive by default. To activate, uncomment the parameter and set the value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>all.syslog_buffering</td>
<td>When this parameter is a non-zero value, 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 “Data Buffering” on page 14 to use it with the Syslog Buffering Feature.)</td>
<td>0</td>
</tr>
</tbody>
</table>
### Table 8-2: CAS specific parameters for the pslave.conf

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Factory Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>all.syslog_sess</td>
<td>Syslog_buffering must be activated for the following settings to work:</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>• When 0, syslog messages are always generated, whether or not there is a session to the port sending data to the unit.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When 1, syslog messages are NOT generated when there IS a session to the port sending data to the unit, but generation of syslog messages is resumed when there IS NOT a session to the port.</td>
<td></td>
</tr>
<tr>
<td>all.dont_show_DBmenu</td>
<td>• When zero, a menu with data buffering options is shown when a data buffering file that is not empty is found.</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>• When 1, the data buffering menu is not shown.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When 2, the data buffering menu is not shown but the data buffering file is shown if it is not empty.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When 3, the data buffering menu is shown, but without the erase and show and erase options.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>NOTE:</strong> This parameter is inactive by default. To activate, uncomment the parameter and set the value.</td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Factory Configuration</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>all.alarm</td>
<td>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 (see “Syslog-ng” on page 137 for the syslog-ng configuration file).</td>
<td>0</td>
</tr>
<tr>
<td>all.billing_records</td>
<td>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 using the format: cycXXXXXX-YYMMDD.hhmmss.txt (e.g., cycTS100-060122.153611.txt.</td>
<td>50</td>
</tr>
<tr>
<td>all.billing_timeout</td>
<td>Billing timeout configuration. A non-zero value defines how long (minutes) a billing file should wait for records before closing. After a file is closed, this file is available for transfer and a new one is opened. Zero means “no timeout”, so the file is only closed after “billing_records” are received.</td>
<td>60</td>
</tr>
<tr>
<td>all.billing_eor</td>
<td>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.</td>
<td>&quot;\n&quot;</td>
</tr>
</tbody>
</table>
**Table 8-2: CAS specific parameters for the pslave.conf**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Factory Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>all.sniff_mode</td>
<td>Determines what other users connected to the same port (see the parameter <em>admin_users</em> below) can see of the session of the first connected user (main session): &lt;br&gt;  - <em>in</em> shows data written to the port  &lt;br&gt;  - <em>out</em> shows data received from the port  &lt;br&gt;  - <em>i/o</em> shows both streams  &lt;br&gt;  - <em>no</em> means sniffing is not permitted. &lt;br&gt; The second and later sessions are called sniff sessions and this feature is activated whenever the protocol parameter is set to <em>socket_ssh</em>, <em>socket_server</em>, or <em>socket_server_ssh</em>.</td>
<td>no</td>
</tr>
<tr>
<td>all.admin_users</td>
<td>Determines which users can receive the sniff session menu. Users who can receive the menu have two options: 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 <em>authtype</em> must not be none. User groups (defined with the parameter <em>conf.group</em>) can be used in combination with user names in the parameter list. &lt;br&gt; <strong>NOTE:</strong> This parameter is inactive by default. To activate, uncomment the parameter and set the value.</td>
<td>null</td>
</tr>
</tbody>
</table>
### Table 8-2: CAS specific parameters for the pslave.conf

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Factory Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>all.multiple_sessions</td>
<td>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. The default is no. See “Session Sniffing” on page 175 for details.</td>
<td>no</td>
</tr>
<tr>
<td>all.escape_char</td>
<td>Defines which character must be typed to make the session enter “menu mode”. The possible values are &lt;CTRL-a&gt; to &lt;CTRL-z&gt;. Represent the CTRL key with a circumflex ‘^’. This parameter is only valid when the port protocol is socket_server, socket_ssh, or socket_server_ssh. Default value is ‘^z’.</td>
<td>^z</td>
</tr>
<tr>
<td>all.tx_interval</td>
<td>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, 100 ms is assumed. If set to zero or a value above 1000, no buffering will take place.</td>
<td>100</td>
</tr>
</tbody>
</table>
Table 8-2: CAS specific parameters for the pslave.conf

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Factory Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>all.idleinterval</td>
<td>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.</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td><strong>NOTE:</strong> This parameter is inactive by default. To activate, uncomment the parameter and set the value.</td>
<td></td>
</tr>
<tr>
<td>s1.alias</td>
<td>Alias name given to the server connected to the serial port. Server_connected.</td>
<td>null</td>
</tr>
<tr>
<td></td>
<td><strong>NOTE:</strong> This parameter is inactive by default. To activate, uncomment the parameter and set the value.</td>
<td></td>
</tr>
<tr>
<td>s1.pool_ipno</td>
<td>This is the default IP address of the CPS'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.</td>
<td>192.168.2.1</td>
</tr>
<tr>
<td>s1.pool_socket_port</td>
<td>In the CAS profile, this defines an alternative labeling system for the CPS pool of ports. In this example, serial interface 1 is assigned to the pool identified by port value 3001. Using $s&lt;serial port #&gt;.pool_socket_port$ one can assign each serial interface to a different pool of ports. One serial interface can belong to only one pool of ports. Each pool of ports can have any number of serial interfaces.</td>
<td>3000</td>
</tr>
</tbody>
</table>
**Table 8-2: CAS specific parameters for the pslave.conf**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Factory Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>conf.telnet</td>
<td>Location of the Telnet utility</td>
<td>/usr/bin/telnet</td>
</tr>
<tr>
<td>conf.ssh</td>
<td>Location of the SSH utility.</td>
<td>/bin/ssh</td>
</tr>
</tbody>
</table>

**pslave.conf TS (Terminal Server) Parameters**

The following parameters are unique to a TS setup except where indicated.

**Table 8-3: TS specific parameters for the pslave.conf file**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Factory Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>s1.pool_alias</td>
<td>Alias name given to the pool to which this serial interface belongs.</td>
<td>pool_1</td>
</tr>
<tr>
<td>IMPORTANT: pool_alias cannot have the characters '~', '*', '?', ' ' (space), and '/'.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>s2.tty</td>
<td>Defines the physical device name associated to the serial port (without the /dev/).</td>
<td>disabled</td>
</tr>
<tr>
<td>Note: This parameter is inactive by default. To activate, uncomment the parameter.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>s8.tty</td>
<td>Defines the physical device name associated to the serial port (without the /dev/).</td>
<td>disabled</td>
</tr>
<tr>
<td>Note: This parameter is inactive by default. To activate, uncomment the parameter.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 8-3: TS specific parameters for the pslave.conf file

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Factory Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>conf.locallogins</td>
<td>This parameter is only necessary when authentication is being performed for a port. When set to one, you can log in to the CPS directly by inserting an exclamation point “!” before your login name, then using your normal password. This is useful if the Radius authentication server is down.</td>
<td>1</td>
</tr>
<tr>
<td>all.host</td>
<td>The IP address of the host to which the terminals will connect.</td>
<td>192.168.160.8</td>
</tr>
<tr>
<td>all.term</td>
<td>Defines the terminal type assumed when performing rlogin or Telnet to other hosts.</td>
<td>vt100</td>
</tr>
<tr>
<td>all.userauto</td>
<td>Username used when connected to a UNIX server from the user’s serial terminal.</td>
<td>null</td>
</tr>
</tbody>
</table>

**NOTE:** This parameter is inactive by default. To activate, uncomment the parameter and set the value.

| all.protocol (for TS) | For the terminal server configuration, the possible protocols are:  
- *login* (requests a username and password)  
- *rlogin* (receives a username from the CPS and requests a password)  
- *Telnet*  
- *SSHv1*  
- *SSHv2*  
- *socket_client*  
   If the protocol is configured as *Telnet* or *socket_client*, the *all.socket_port* parameter must be configured. | socket_server         |
### Table 8-3: TS specific parameters for the pslave.conf file

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Factory Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>all.socket_port</td>
<td>The <em>socket_port</em> is the TCP port number of the application that will accept connection requested by this serial port. Usually, that application is Telnet (23).</td>
<td>7001+</td>
</tr>
<tr>
<td>all.telnet_client_mode</td>
<td>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 &quot;0&quot; or &quot;1&quot;, where &quot;0&quot; is text mode (default) and &quot;1&quot; is a binary mode.</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td><strong>NOTE:</strong> This parameter is inactive by default. To activate, uncomment the parameter and set the value.</td>
<td></td>
</tr>
<tr>
<td>s16.tty (TS)</td>
<td>Defines the physical device name associated to the serial port (without the /dev/).</td>
<td>disabled</td>
</tr>
<tr>
<td></td>
<td><strong>NOTE:</strong> This parameter is inactive by default. To activate, uncomment the parameter.</td>
<td></td>
</tr>
</tbody>
</table>
pslave.conf Dial-in Parameters

The following parameters are unique to a Dial-in setup except where indicated.

Table 8-4: Dial-in specific parameters for the pslave.conf

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Factory Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>conf.pppd</td>
<td>Location of the ppp daemon with Radius.</td>
<td>/usr/local/sbin/pppd</td>
</tr>
<tr>
<td>all.netmask</td>
<td>Defines the network mask for the serial port.</td>
<td>255.255.255.255</td>
</tr>
<tr>
<td>all.ipno</td>
<td>See description in CAS section.</td>
<td>192.168.1.101+</td>
</tr>
<tr>
<td>all.initchat</td>
<td>Modem initialization string.</td>
<td>null</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> This parameter is inactive by default. To activate, uncomment the parameter and set the value.</td>
<td></td>
</tr>
<tr>
<td>all.autoppp</td>
<td><em>all.autoppp</em> PPP options to auto-detect a ppp session.</td>
<td>null</td>
</tr>
<tr>
<td></td>
<td><em>cb-script</em> 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 CPS, 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).</td>
<td></td>
</tr>
</tbody>
</table>

**NOTE:** This parameter is inactive by default. To activate, uncomment the parameter and set the value.
Bidirectional Telnet Parameters

Bidirectional Telnet protocol, or “Dynamic Mode”, supports both “socket_server” connection (CAS) and “login” (TS) profiles. Both connection protocols are supported on one port, but connections cannot be opened simultaneously.

The CPS accepts the incoming TCP connection directed to a serial port as a socket_server connection. When the serial port is configured for “login”, the TCP connection is refused.

The “login” mode allows the administrator to build custom menus using the menush_cfg MenuShell Configuration Utility. When the attached terminal is receiving power and the keyboard’s [Enter] key is pressed, a login banner and a login prompt display.

If the user does not log in within the time frame you configure, the serial port returns to an idle state.

Table 8-4: Dial-in specific parameters for the pslave.conf

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Factory Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>all.pppopt</td>
<td>all.pppopt PPP options when user has already been authenticated.</td>
<td>null</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> This parameter is inactive by default. To activate, uncomment the parameter and set the value.</td>
<td></td>
</tr>
<tr>
<td>all.protocol</td>
<td>For the Dial-in configuration, the available protocols are ppp, ppp_only, slip, and cslip.</td>
<td>ppp</td>
</tr>
<tr>
<td>s32.tty</td>
<td>See the s1.tty entry in the CAS section.</td>
<td>disabled</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> This parameter is inactive by default. To activate, uncomment the parameter.</td>
<td></td>
</tr>
</tbody>
</table>

pslave.conf Bidirectional Telnet Parameters

Bidirectional Telnet protocol, or “Dynamic Mode”, supports both “socket_server” connection (CAS) and “login” (TS) profiles. Both connection protocols are supported on one port, but connections cannot be opened simultaneously.

The CPS accepts the incoming TCP connection directed to a serial port as a socket_server connection. When the serial port is configured for “login”, the TCP connection is refused.

The “login” mode allows the administrator to build custom menus using the menush_cfg MenuShell Configuration Utility. When the attached terminal is receiving power and the keyboard’s [Enter] key is pressed, a login banner and a login prompt display.

If the user does not log in within the time frame you configure, the serial port returns to an idle state.
The following parameters are specific to Bidirectional Telnet in the /etc/portslave/pslave.conf file.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>sxx.protocol</td>
<td>The connection protocol for the specified serial port.</td>
<td>s1.protocol bidirect</td>
</tr>
<tr>
<td>sxx.authtype</td>
<td>The authentication method configured for the serial port.</td>
<td>s1.authtype local</td>
</tr>
<tr>
<td>sxx.logintimeout</td>
<td>The time (in seconds) for the serial port to return to idle state, if the user does not log in.</td>
<td>s1.logintimeout 60</td>
</tr>
<tr>
<td>sxx.sh</td>
<td>The shell command used to present a menu to the user.</td>
<td>s1.sh /bin/menush</td>
</tr>
</tbody>
</table>

“xx” represents the serial port number being configured.

Table 8-5: Bidirectional Telnet specific parameters for pslave.conf
See “CAS specific parameters for the pslave.conf” on page 259, and see “TS specific parameters for the pslave.conf file” on page 273 for the respective parameters.

To configure Bidirectional Telnet

CLI Method

To configure Bidirectional Telnet protocol:

1. Open the CLI interface by issuing the command:
   
   # CLI

2. Activate bidirectional Telnet
   
   cli> config physicalports <'all' or range/list[1-4]> general protocol <protocolname>

3. Specify a login timeout.
   
   cli> config physicalports <'all' or range/list[1-4]> access logintimeout <login timeout in seconds>

4. Save the configuration.
   
   cli> config savetoflash
5. Exit the CLI mode.

To exit the CLI mode and return to the CPS’s shell, type the following command:

```cli> quit```

**To configure a menu shell**

Enter the following command at the prompt:

```[root@CPS /]# menush_cfg```

The following configuration utility displays, allowing you to configure a menu shell for the user.

```
--------------------------------------
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

**Using the CLI Interface to Configure Common Parameters**

You can use the CLI interface to configure some of the physical port parameters that were presented in the preceding pages.

**General State Parameters**

General configurations are under the menu:

```cli> config physicalports <'all' or range/list[1-4]> general```

Under this menu, you can configure the following parameters:

- `alias`—To configure an alias for a server connected to the serial port.
- `datasize`—Configure the number of bits per character. Valid values are 5–8.
• **flow**—Set the flow control. Valid values are: *none*, *hard* for hardware, and *soft* for software.

• **parity**—Configure the parity. Valid values are *none*, *even*, and *odd*.

• **protocol**—Configure the protocol that will be used to connect to the ports. Valid values are *bidirectional telnet*, *consoledraw*, *cslip*, *ppp*, *slip*, *telnet*, *consolessh*, *local*, *pppnoauth*, *sshv1*, *consoletelnet*, *pm*, *rawsocket*, and *sshv2*.

• **speed**—Configure the serial port speed. Valid values are:

```
300 1200 2400 4800 9600 14400 19200 28800
38400 57600 76800 115200 230400 460800 921600
```

• **stopbits**—This parameter configures the number of stop bits. Valid values are *1* and *2*.

**Other State Parameters**

Other state configurations are under the menu:

```
cli> config physicalports <'all' or range/list[1-4]> other
```

Under this menu you can configure the following parameters:

• **authbio**—Configure whether a CPS Bio authentication scanner is used.

• **banner**—Set the banner that will be issued when the user connects to the port. Strings must be entered between “” (double quotes).

• **breakinterval**—Set break interval in milliseconds (ms).

• **breaksequence**—Set the break sequence. Usually a character sequence, ~break (Ctrl+b)

• **host**—IP address of the device to which you are connecting.

• **idletimeout**—Configure idle timeout in minutes.

• **portip**—Configure an IP alias to the serial port.

• **sttyoptions**—Configure stty parameters.

• **tcpkeepalive**—Configure pool interval in milliseconds (ms).

• **tcpport**—Configure the socket port number. Four-digit values are valid for this parameter (e.g., 7001).

• **terminaltype**—The terminal type when using a TS profile to connect to a host system.

• **winems**—Enables or disables Windows EMS.
8.2 Examples for Configuration Testing

The purpose of the following three examples is to test a configuration. Follow the steps after configuring the CPS.

Console Access Server

With the CPS set up as a CAS, you can access a server that is connected to the CPS 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 to perform authentication using a Radius server, a TacacsPlus server, or even a local database. You can use Telnet or SSH.

See “Appendix A: New User Background Information” on page 315 for more information about SSH.

This chapter contains all of the necessary information to configure a fully-functional CAS environment. Consult the tables from the preceding section of this chapter 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 8.1 - Console Access Server diagram*
The following diagram shows additional scenarios for the CPS—data buffering, remote access, and both remote and local authentication.

![CAS diagram with various authentication methods](image)

As shown, you can use either Telnet or SSH for the “CAS with local authentication” scenario. After configuring the serial ports as described in this chapter, use the following step-by-step checklist to test the configuration.

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>`.

2. Confirm physical connection.

   Ensure that the physical connection between the CPS and the servers is correct. Use a cross cable, not a modem cable. See “Appendix C: Cabling and Hardware Information” on page 341.

3. Confirm that the server is set to the same parameters as the CPS.
The CPS is set for communication at 9600 bps, 8N1. The server must be configured to communicate on the serial console port with the same parameters.

4. Confirm routing.

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 CPS using the following command:

```
# telnet 200.200.200.1 7001
```

For both Telnet and SSH sessions, the servers can be reached by either the Ethernet IP of the CPS and assigned socket port, or the 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.

5. Activate the changes.

See “Activate the changes,” on page 81 and “Save the changes,” on page 81 of Chapter 4: Network.

**NOTE:** For an application running on a workstation with Windows, the remote serial port works like a regular COM port. All the I/O with the serial device attached to the CPS is done through socket connections, and a COM port is emulated to the application.
Terminal Server

The CPS provides features for out-of-band management through 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.

![Terminal Server diagram](image)

*Figure 8.3 - Terminal Server diagram*

In the preceding figure, no authentication is used and rlogin is chosen as the protocol. After configuring the serial ports as described in this chapter, use the following step-by-step checklist to test the configuration.

1. Create a new user.
   
   Since authentication is set to *none*, the CPS will not authenticate the user. However, the Linux Server receiving the connection will. Create a new user called *test* on the server, with the password *test*.

2. Confirm that the server is reachable.
   
   From the console, ping *200.200.200.3* to make sure the server is reachable.

3. Check physical connections.
Ensure that the physical connection between the CPS and the terminals is correct. Use a cross cable (not a modem cable). See “Appendix C: Cabling and Hardware Information” on page 341 for cabling information.

4. Confirm that the terminals are set to the same parameters as the CPS.

   The CPS is set for communication at 9600 bps, 8N1. The terminals must be configured with the same parameters.

5. From a terminal connected to the CPS, try to log in to the server using the username and password configured in step 1.

6. Activate changes.

   See “Activate the changes,” on page 81 and “Save the changes,” on page 81 of Chapter 4: Network.
Dial-in Access

The CPS can be configured to accommodate out-of-band management. Ports can be configured on the CPS 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. APC recommends that a maximum of two ports be configured for this option.

After configuring the serial ports as described in this chapter, use the following step-by-step checklist to test the configuration:

1. Create a new user.
   Since Radius authentication was chosen, create a new user called test on the Radius authentication server, with the password test.

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.

3. Confirm physical connections.

Ensure that the physical connection between the CPS and the modems is correct. Use the modem cable provided with the CPS. See “Appendix C: Cabling and Hardware Information” on page 341 for cabling information.

4. Confirm modem settings.

The CPS is been set for communication at 57600 bps, 8N1. The modems should be programmed to operate at the same speed on the DTE interface.

5. Confirm routing.

Make sure that the computer is configured to route console data to the serial console port.

6. Perform a test dial-in.

Try to dial in to the CPS from a remote computer using the username and password configured in step 1. The computer dialing in must be configured to receive its IP address from the remote access server (the CPS in this case) and to use PAP authentication.

7. Activate changes.

See “Activate the changes.” on page 81 and “Save the changes.” on page 81 of Chapter 4: Network.
Chapter 9: Additional Features and Applications

This chapter covers special features or applications that do not fit into any of the preceding chapters. The following features are described in this chapter:

- Windows 2003 Server Management
- IPMI Configuration
- Line Printer Daemon
- CAS Port Pool
- Billing

9.1 Windows 2003 Server Management

Emergency Management Services (EMS) is a Windows 2003 Server feature 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 CPS 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 CPS 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:

**Table 9-1: machine info tag**

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;guid&gt;</code></td>
<td>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).</td>
</tr>
<tr>
<td><code>&lt;name&gt;</code></td>
<td>Is the system name.</td>
</tr>
<tr>
<td><code>&lt;os-build-number&gt;</code></td>
<td>Is a numeric string that identifies a successive Windows Build.</td>
</tr>
<tr>
<td><code>&lt;os-product&gt;</code></td>
<td>Is the name of the Windows Server 2003 product currently running on this server. It is one of the following:</td>
</tr>
<tr>
<td></td>
<td>• Windows Server 2003 Datacenter Edition</td>
</tr>
<tr>
<td></td>
<td>• Windows Server 2003 Embedded</td>
</tr>
<tr>
<td></td>
<td>• Windows Server 2003 Enterprise Edition</td>
</tr>
<tr>
<td></td>
<td>• Windows Server 2003</td>
</tr>
<tr>
<td><code>&lt;os-service-pack&gt;</code></td>
<td>Is an alphanumeric string that identifies the most up-to-date service pack installed. If none installed, the string is None.</td>
</tr>
<tr>
<td><code>&lt;os-version&gt;</code></td>
<td>Is the numeric identification of the Windows version currently running.</td>
</tr>
<tr>
<td><code>&lt;processor-architecture&gt;</code></td>
<td>Is either x86 or IA64, designating the two processor architectures currently supported by Windows Server 2003.</td>
</tr>
</tbody>
</table>
A sample encoding of the `<machine-info>` tag:

```xml
<?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-service-pack>None</os-service-pack>
</machine-info>
```

*File Description 9.1: 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:

**Table 9-2: Elements in the <channel-switch> tag**

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
</table>
| <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.
### Table 9-2: Elements in the `<channel-switch>` tag

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;description&gt;</code></td>
<td>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)</td>
</tr>
<tr>
<td><code>&lt;guid&gt;</code></td>
<td>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 <code>&lt;application-type&gt;</code> tag above.</td>
</tr>
</tbody>
</table>
Table 9-2: Elements in the <channel-switch> tag

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
</table>
| <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 |

Sample encoding of the SAC channel tag:

```xml
<channel-switch>  
    <name>SAC</name>  
    <description>Special Administration Console</description>  
    <type>VT-UTF8</type>  
    <guid>1ae4cc0-cff3-11d6-9a3d-806e6f6e6963</guid>  
    <application-type>63d02270-8aa4-11d5-bccf-806d6172696f</application-type>  
</channel-switch>
```

*File Description 9.2: SAC channel tag example*
Sample encoding of the CMD channel tag:

```
<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 9.3: CMD channel tag example*

Sample encoding of the GUI-Mode Setup Debug Log channel tag:

```
<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 9.4: 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.

**Table 9-3: `<BP>` tags description**

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;INSTANCE CLASSNAME=&gt;</code></td>
<td>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.</td>
</tr>
<tr>
<td><code>&lt;machine-info&gt;</code></td>
<td>Is described above.</td>
</tr>
<tr>
<td><code>&lt;PROPERTY NAME=&gt;</code></td>
<td>Provides additional details, such as error code of the abnormal condition that caused the break point.</td>
</tr>
</tbody>
</table>
Sample encoding of the Break Point tag:

```xml
<?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-service-pack>None</os-service-pack>
</machine-info>
</INSTANCE>
</BP>
```

File Description 9.5: 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:

```bash
filter f_windows_bluescreen { facility(local<conf.DB_facility>) and
   level(info)nd match("XML_MONITOR") and match("BLUESCREEN"); } ;
```

and

```bash
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:

**Table 9-4: f_windows_boot macros**

<table>
<thead>
<tr>
<th>Macro</th>
<th>Description</th>
<th>Value to replace macro</th>
</tr>
</thead>
<tbody>
<tr>
<td>$&lt;INSTANCE</td>
<td>Reason for the break point.</td>
<td>BLUESCREEN</td>
</tr>
<tr>
<td>CLASSNAME=&gt;</td>
<td>Currently there is only one type, BLUESCREEN.</td>
<td></td>
</tr>
<tr>
<td>$&lt;PROPERTY</td>
<td>Additional details about break point.</td>
<td>STOPCODE</td>
</tr>
<tr>
<td>NAME=&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$&lt;VALUE&gt;</td>
<td>Additional details about break point.</td>
<td>0xE2</td>
</tr>
<tr>
<td>$&lt;name&gt;</td>
<td>Machine name</td>
<td>MY_WIN_SERVER</td>
</tr>
<tr>
<td>$&lt;guid&gt;</td>
<td>GUID that uniquely identifies this server. If no such value is available, all 0’s GUID string is used.</td>
<td>4c4c4544-8e00-4410-8045-80c04f4c4e20</td>
</tr>
<tr>
<td>$&lt;processor-</td>
<td>Processor architecture. It can be either x86 or IA64.</td>
<td>x86</td>
</tr>
<tr>
<td>architecture&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$&lt;os-version&gt;</td>
<td>Windows version.</td>
<td>5.2</td>
</tr>
</tbody>
</table>
Basic Network Configuration

Table 9-4: f_windows_boot macros

<table>
<thead>
<tr>
<th>Macro</th>
<th>Description</th>
<th>Value to replace macro</th>
</tr>
</thead>
<tbody>
<tr>
<td>$&lt;os-service-pack&gt;</td>
<td>Alphanumeric string that identifies the most up-to-date service pack installed. If none installed, the string is None.</td>
<td>None</td>
</tr>
<tr>
<td>$&lt;tty&gt;</td>
<td>CPS serial port tty or alias name.</td>
<td>$1.tty$1</td>
</tr>
</tbody>
</table>

For the f_windows_boot, the following macros are available:

Table 9-5: f_windows_boot available macros

<table>
<thead>
<tr>
<th>Macro</th>
<th>Description</th>
<th>Value to replace macro</th>
</tr>
</thead>
<tbody>
<tr>
<td>$&lt;name&gt;</td>
<td>Machine name</td>
<td>MY_WIN_SERVER</td>
</tr>
<tr>
<td>$&lt;guid&gt;</td>
<td>GUID that uniquely identifies this server. If no such value is available, all 0’s GUID string is used.</td>
<td>4c4c4544-8e00-4410-8045-80c04f4c4e20</td>
</tr>
<tr>
<td>$&lt;processor-architecture&gt;</td>
<td>Processor architecture. It can be either x86 or IA64.</td>
<td>x86</td>
</tr>
<tr>
<td>$&lt;os-version&gt;</td>
<td>Windows version.</td>
<td>5.2</td>
</tr>
<tr>
<td>$&lt;os-build-number&gt;</td>
<td>Numeric string that identifies a successive Windows Build.</td>
<td>3763</td>
</tr>
</tbody>
</table>
As an example on how we can use above macros, let’s say we want the CPS 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`:

```plaintext
source src { unix-stream("/dev/log"); }
log { source(src); filter(f_windows_boot); destination(win2003mail); };
```

### Table 9-5: `f_windows_boot` available macros

<table>
<thead>
<tr>
<th>Macro</th>
<th>Description</th>
<th>Value to replace macro</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>$&lt;os-service-pack&gt;</code></td>
<td>Alphanumeric string that identifies the most up-to-date service pack installed. If none installed, the string is None.</td>
<td>None</td>
</tr>
<tr>
<td><code>$&lt;tty&gt;</code></td>
<td>CPS serial port tty or alias name.</td>
<td>S2.server_connected_to_serial2</td>
</tr>
</tbody>
</table>

As an example on how we can use above macros, let’s say we want the CPS 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`:

```plaintext
destination win2003mail { pipe("/dev/cyc_alarm"
  template("sendmail -t administrator@APC.com -f cps -s "\n    Server $<name> crashed\" -m \"Break Point: $<INSTANCE CLASSNAME=> $<PROPERTY NAME=> $<VALUE>\n    Server: $<name>\n    OS: $<os-product>\n    Build: $<os-build-number> Version: $<os-version>\n    Service Pack: $<os-service-pack>\n    Processor: $<processor-architecture>\n    Server GUID: $<guid>\n    CPS port: $<tty>\n    \' -h mail.APC.com ");};
```

*File Description 9.6: Send e-mail when crashing example*

And the following entry will activate the `win2003mail` action when the `f_windows_bluescreen` filter is successful:

```plaintext
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.

**Table 9-6: Server Commands**

<table>
<thead>
<tr>
<th>Command Set</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ch</td>
<td>Channel management commands.</td>
</tr>
<tr>
<td>ch -ci &lt;#&gt;</td>
<td>Close a channel by its number.</td>
</tr>
<tr>
<td>cmd</td>
<td>Create a Command Prompt channel.</td>
</tr>
<tr>
<td>ch -si &lt;#&gt;</td>
<td>Switch to another channel (from Channel 0).</td>
</tr>
<tr>
<td>d</td>
<td>Dump the current kernel log.</td>
</tr>
<tr>
<td>f</td>
<td>Toggles the information output by the t-list command, which shows processes only, or shows processes and threads.</td>
</tr>
<tr>
<td>i</td>
<td>List all IP network numbers and their IP addresses.</td>
</tr>
<tr>
<td>i &lt;#&gt; &lt;ip&gt;</td>
<td>Set network interface number, IP address, subnet and gateway.</td>
</tr>
<tr>
<td>id</td>
<td>Display the computer identification information.</td>
</tr>
<tr>
<td>k &lt;pid&gt;</td>
<td>Kill the given process.</td>
</tr>
<tr>
<td>l &lt;pid&gt;</td>
<td>Lower the priority of a process to the lowest possible.</td>
</tr>
<tr>
<td>lock</td>
<td>Lock access to Command Prompt channels. You must provide valid logon credentials to unlock a channel.</td>
</tr>
<tr>
<td>m &lt;pid&gt; &lt;MB-allow&gt;</td>
<td>Limit the memory usage of a process to &lt;MB-allow&gt;.</td>
</tr>
<tr>
<td>p</td>
<td>Causes t-list command output to pause after displaying one full screen of information.</td>
</tr>
<tr>
<td>r &lt;pid&gt;</td>
<td>Raise the priority of a process by one.</td>
</tr>
</tbody>
</table>
9.2 IPMI Configuration

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 CPS 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 CPS 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

This program lets you manage Intelligent Platform Management Interface (IPMI) functions of either the local system or of a remote system, using IPMI V1.5. These functions include printing FRU information, LAN configuration, sensor readings, and remote chassis power control.

You can configure IPMI using the following methods:

- ipmitool – IPMI Configuration
- CLI – IPMI [CLI]

### Table 9-6: Server Commands

<table>
<thead>
<tr>
<th>Command Set</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>Display the current time and date (24 hour clock used).</td>
</tr>
<tr>
<td>mm/dd/yyyy</td>
<td>Set the current time and date (24 hour clock used).</td>
</tr>
<tr>
<td>hh:mm</td>
<td>Tlist.</td>
</tr>
<tr>
<td>crashdump</td>
<td>Crash the system. Crash dump must be enabled.</td>
</tr>
<tr>
<td>restart</td>
<td>Restart the system immediately.</td>
</tr>
<tr>
<td>shutdown</td>
<td>Shut down the system immediately.</td>
</tr>
</tbody>
</table>
**IPMI [ipmitool]**

Utility for controlling IPMI-enabled devices.

**Name**

ipmitool

**Usage**


**Options**

Use the following options to configure IPMI.

**Table 9-7:** Options for ipmitool

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>-h</td>
<td>Get basic usage help from the command line.</td>
<td>N/A</td>
</tr>
<tr>
<td>-v</td>
<td>Increase verbose output level. This option may be specified multiple times to increase the level of debug output.</td>
<td>N/A</td>
</tr>
<tr>
<td>-V</td>
<td>Display version information.</td>
<td>N/A</td>
</tr>
<tr>
<td>-I &lt;interface&gt;</td>
<td>Selects IPMI interface to use.</td>
<td>lan</td>
</tr>
<tr>
<td>-H &lt;address&gt;</td>
<td>Remote server address, can be IP address or hostname. This option is required for the LAN interface connection.</td>
<td>N/A</td>
</tr>
<tr>
<td>-U &lt;username&gt;</td>
<td>Remote username.</td>
<td>Default is NULL.</td>
</tr>
<tr>
<td>-L &lt;privlvl&gt;</td>
<td>Force session privilege level.</td>
<td>USER</td>
</tr>
<tr>
<td></td>
<td>Default is USER</td>
<td></td>
</tr>
<tr>
<td>-A &lt;authtype&gt;</td>
<td>Force session authentication type.</td>
<td>PASSWORD</td>
</tr>
</tbody>
</table>
Expressions

1.0 chassis
   1.1 status
       Returns information about the high-level status of the system chassis
       and main power subsystem
   1.2 poh
       Returns the Power-On Hours counter
   1.3 identify <interval>
       Controls the front panel identify light. Default is 15. Use 0 to turn off.
   1.4 restart_cause
       Queries the chassis for the cause of the last system restart
   1.5 policy
       Sets the chassis power policy in the event power failure
       1.5.1 list – Return supported policies
       1.5.2 always-on – Turn on when power is restored
       1.5.3 previous – Return to previous state when power is restored
       1.5.4 always-off – Stay off after power is restored
   1.6 power
       Performs a chassis control command to view and change the power
       state
       1.6.1 status – Show current chassis power status
       1.6.2 on – Power up chassis
       1.6.3 off – Power down chassis into soft off
       1.6.4 cycle – Provide power off interval of at least 1 second
       1.6.5 reset – Perform a hard reset
   1.7 sensor
       1.7.1 list – Lists sensors and thresholds in a wide table format

---

Table 9-7: Options for ipmitool (Continued)

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>-P &lt;password&gt;</td>
<td>Remote server password.</td>
<td>Valid password for specified username account.</td>
</tr>
</tbody>
</table>
IPMI [CLI]
You can configure IPMI using the `ipmi` keyword and the following attributes in CLI mode:

1.0 config – enter into configuration state
1.1 ipmi – configure IPMI devices
   1.1.1 add <alias> – add a IPMI device
       serverIP <n.n.n.n> – IP address of the device
       authType <authentication options: md2, md5, none, password> – authentication type
       privilege <user or operator or admin> – user access level
       username <string> – user name used to access the device
       password <string> – password used to access the device
   1.1.2 edit <alias> – edit the IPMI device
       serverIP <n.n.n.n> – IP address of the device
       authType <authentication options: md2, md5, none, password> – authentication type
       privilege <user or operator or admin> – user access level
       username <string> – user name used to access the device
       password <string> – password used to access the device
   1.1.3 delete <alias> – delete the IPMI device
1.2 physicalports <port number(s)> – configure physical serial ports
   1.2.1 powermanagement state disableIPMI – disable the IPMI menu
       1.2.1.1 enableIPMI – to enable the IPMI menu
       server <alias / list IPMI devices> – alias of the IPMI device
       key <^(character)> – the hotkey used to access the IPMI menu.

**NOTE:** The default IPMI hotkey is “^I”, where ^ stands for the Ctrl key on the keyboard. The hexadecimal code for the <Ctrl-I> default IPMI hotkey is the same as the keyboard’s <Tab> key. You can choose to change the default through the CLI command,
cli>config>physicalports [port]>powermanagement>enableIPMI>key [value]

9.3 Line Printer Daemon
This feature implements the Unix Line Printer Daemon (LPD) in the CPS and can be used with local serial printers. It enables the CPS to receive network print requests and service them using locally attached Serial printers.

To configure the lpd you need to follow these steps:
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
   ```

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: \\n   :sd=/var/spool/lpd/lp_ttyS2: \\
   :lp=/dev/ttyS2: \\
   :lf=/var/log/lpd.log:
   ```
   
   File Description 9.7: /etc/printcap file

3. Enable the printer daemon.
   Edit the file /etc/lpd.sh and change the option ENABLE to YES

4. Allow clients to use the service.
   Edit the file /etc/hosts.lpd and include the hosts name that you allow to user the CPS printers.
NOTE: (The lpd needs to translate the IP address of the request message to the host name, check your resolv.conf file).

5. Restart the processes, use the command "runconf" and "daemon.sh".

6. Save the configuration in flash, use the command "saveconf".

   In your Linux client machine type the following command to check the CPS configuration is OK:
   
   # lpr -P lp@<CPS IP address> <file that you want printer> <enter>

9.4 CAS Port Pool

This feature is available for the CPS 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_alias, and
- pool_socket_port

The three new parameters: pool_ipno, pool_alias, and pool_socket_port have the same meaning as ipno, alias, 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_alias, or pool_socket_port instead of ipno, alias, or socket_port respectively in the SSH/Telnet command.
When a connection request arrives using one of `pool_ipno`, `pool_alias`, or `pool_socket_port` the CPS 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 `runconf` 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.alias serial-1 // alias for specific allocation
s1.pool_alias 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.alias serial-2 // alias for specific allocation
s2.pool_alias 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.alias serial-3 // alias for specific allocation
s3.pool_alias 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.alias serial-4 // alias for specific allocation
s4.pool_alias pool-2 // alias for the pool
```

File Description 9.8: Part of the /etc/portslave/pslave.conf file

In the preceding example, 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.

9.5 Billing

All CPS family of products can be used as an intermediate buffer to collect serial data (like billing tickets from a PBX), making them available for a posterior file transfer. Different ports can have simultaneous "billing sessions".

General Feature Description

CPS reads the serial port and saves the information to Ramdisk files, which is limited to the maximum number of records per file. After the files are closed, they are available for transfer at /var/run/DB, or an alternate path defined by the user in the pslave.conf file. See Table 8-2 on page 259 for additional details on billing file configuration.

How to Configure It

The configuration for this feature is made in the /etc/portslave/plsave.conf file. Billing parameters can be configured using the vi method and by using the wizard.

VI method—passed values and involved parameters

Open the /etc/portslave/plsave.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, see “Chapter 8: Profile Configuration” on page 245.

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 CPS 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 “alias”
• 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 “alias” name (s1.alias 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.

Billing Wizard

This feature improves the billing application by using a script and automating the upload of the billing records files from the CPS to a remote server using FTP or SSH.

How to configure

The config_billing.sh script is used to configure a serial port for billing protocol, and configure upload scripts using FTP or SSH. The config_billing.sh script configures the files /etc/billing_up.conf /etc/billing_crontab, and /etc/crontab_files.

To configure a port for billing

1. Execute the config_billing.sh and enter the parameters to be configured.

   Usage: config_billing.sh [X] [options]

   where:
   X is the port number to be configured
   [options] are:

   -s speed
   -d data size
   -b stopbit
   -p parity
   -r billing records
   -e billing EOR (this parameter must be on " ", like "\n")
   -D billing dir
-S serverFarm
-t date
-T timeout
-i ip
-n netmask
-R route
-u upload

Any parameter that is not specified will remain unchanged. The following parameters are configured by default for billing:

sxx.authtype none
sxx.protocol billing
sxx.flow none
sxx.dcd 0
sxx.sniff_mode no

Select the -u option to execute the `billing_upload_files.sh` script. The script presents the following sequential menu where the upload options can be configured.

```
[root@CPS etc]# billing_upload_files.sh
Transfer Mode (ftp or scp) [ftp]:
Local Directory [/var/run/DB]:
Remote server IP [192.168.1.101]:
Remote directory [/var/billing]:
User [billing]:
Password [billing]:
Upload Interval in minutes []:
```

**NOTE:** Instead of running the -u option, the `/etc/billing_up.conf` can be configured manually to change the parameters. If the parameters remains unchanged the default parameters are uploaded.

**NOTE:** If the “scp” transfer mode is selected and there is no defined authentication, the script generates a key and uploads to the server. The key must be stored on the server with the appropriate configuration.

2. Execute saveconf.
3. Restart CPS to activate the options related to billing upload.
Appendix A: New User
Background Information

The objective of this appendix is to introduce new users to commands, file structure, processes, programs, and other features used by the APC Console Port Server operating system. This appendix includes the following sections:

- User and Passwords
- View Logged-in Users And Their Actions?
- Linux File Structure
- Basic File Manipulation
- The VI Editor
- The Routing Table
- Secure Shell Session
- The Process Table
- TS Menu Script

A.1 User and Passwords

A username and password are necessary to log in to the CPS. The user “root” is predefined with a password “apc”. Change the password as soon as possible to avoid unauthorized access.

To change the password for the “root”, enter the following command.

```
# passwd
```

To create a regular user (without root privileges), enter the following command.

```
# adduser user_name
```

To change the password for a regular user, enter the following command.

```
# passwd user_name
```

To log out, type `logout` at the command prompt.

A regular user can run the command `su -` to become a superuser by completing the following steps:

1. Make sure the group wheel is already created.
An administrator with root access would run the following command:

```
# addgroup wheel
```

In the file `/etc/group` there should be a line with at least the following:

```
wheel::zzz:
```

2. Belong to the group wheel.

An administrator with root access would edit the `/etc/group` file and insert the username at the end of the wheel line. For example, for the user “steve”, the administrator would edit the line `wheel::zzz:` in the file `/etc/group` to add "steve" at the end:

```
wheel::zzz:steve
```

A.2 View Logged-in Users And Their Actions

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 command `w_ori` is the new name of the original command “w” and the command `w_cas` shows the CAS session’s 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 (excluding the CAS users), and the system load averages for the past 1, 5, and 15 minutes.

The following entries are displayed for each user (excluding the CAS users): login name, the tty name, the remote host, login time, idle time, JCPU time (the time used by all processes attached to the tty), PCPU time (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*.

**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*.

**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`, which displays 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 you are not sure which mode you are in, press the `<ESC>` key, which will bring you to the command mode.

**Table A-1: vi modes**

<table>
<thead>
<tr>
<th>Mode</th>
<th>What is done there</th>
<th>How to get there</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command mode</td>
<td>Navigation within the open file.</td>
<td>Press the <code>&lt;ESC&gt;</code> key.</td>
</tr>
<tr>
<td>Editing mode</td>
<td>Text editing.</td>
<td>See list of editing commands below.</td>
</tr>
</tbody>
</table>
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.

**Table A-2: vi navigation commands**

<table>
<thead>
<tr>
<th>Mode</th>
<th>What is done there</th>
<th>How to get there</th>
</tr>
</thead>
<tbody>
<tr>
<td>h</td>
<td>Moves the cursor to the left (left arrow).</td>
<td>From the command mode, type “:” (colon).</td>
</tr>
<tr>
<td>j</td>
<td>Moves the cursor to the next line (down arrow).</td>
<td></td>
</tr>
<tr>
<td>k</td>
<td>Moves the cursor to the previous line (up arrow).</td>
<td></td>
</tr>
<tr>
<td>l</td>
<td>Moves the cursor to the right (right arrow).</td>
<td></td>
</tr>
</tbody>
</table>

When you navigate to the text to be changed, use the following commands to modify the text.

**NOTE:** The commands “i” and “o” will move you into edit mode and everything you type will be taken literally until you press the <ESC> key to return to the command mode.

**Table A-3: vi file modification commands**

<table>
<thead>
<tr>
<th>Mode</th>
<th>What is done there</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td>Inserts text before the cursor position (everything to the right of the cursor is shifted right).</td>
</tr>
<tr>
<td>o</td>
<td>Creates a new line below the current line and inserts text (all lines are shifted down).</td>
</tr>
<tr>
<td>dd</td>
<td>Removes the entire current line.</td>
</tr>
<tr>
<td>x</td>
<td>Deletes the letter at the cursor position.</td>
</tr>
</tbody>
</table>

After you finish modifying a file, enter line mode (by typing a colon “:” from command mode) and use one of the following commands.

**Table A-4: vi line mode commands**

<table>
<thead>
<tr>
<th>Mode</th>
<th>What is done there</th>
</tr>
</thead>
<tbody>
<tr>
<td>w</td>
<td>Saves the file (w is for write).</td>
</tr>
</tbody>
</table>
A.6 The Routing Table

The CPS has a static routing table that you can see by using the commands:

```
# route
```
or

```
# netstat -rn
```

The file `/etc/network/st_routes` is the CPS’s method for configuring static routes. Routes should be added to the file (which is a script run when the CPS is initialized) or at the prompt (for temporary routes) using the following syntax:

```
```

- `[add|del]` — One of these tags must be present. Routes can be either added or deleted.
- `[-net|-host]` — The `-net` tag is for routes to a network and the `-host` tag is for routes to a single host.
- `target` — Target is the IP address of the destination host or network.
- `netmask` and `nt_msk` — The tags `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 counterparts rsh and rlogin. There are two versions of the protocol, SSHv1 and SSHv2. The CPS offers both. The command to start an SSH client session from a UNIX workstation is:

\[ \text{ssh} \ -t \ <\text{user}>@<\text{hostname}> \]

where

• \(<\text{user}> = <\text{username}>:\text{ttySnn} \text{ or}\\
  <\text{username}>:\text{socket_port} \text{ or}\\
  <\text{username}>:\text{ip_addr} \text{ or}\\
  <\text{username}>:\text{alias}\\
\]

\text{NOTE: “alias” is a physical port alias. It can be configured in the file pslave.conf.}

For example:

\begin{tabular}{ll}
username: & APCmycompany \\
AP9302 IP address: & 192.168.160.1 \\
host name: & ap9302 \\
servername for port 1: & file_server \\
\end{tabular}

ttyS1 is addressed by IP 10.0.0.1 or socket port 7001. The ways to access the server connected to the port include:

\begin{verbatim}
ssh -t APC:ttyS1@ap9302
ssh -t APC:7001@ap9302
ssh -t APC:10.0.0.1@ap9302
ssh -t APC:file_server@ap9302
ssh -t -l APC:10.0.0.1 ap9302
ssh -t -l APC:7001 ap9302
\end{verbatim}

For OpenSSH clients, SSHv2 version 4.1p1 or later is the default. In that case, the -l flag is used for SSHv1.

\begin{verbatim}
# ssh -t APC:7001@ap9302
# ssh -t -2 APC:7001@ap9302
# ssh -t APC:7001@ap9302
(OpenSSH 4.1p1 or later - CPS version 2.1.0 or later -> SSHv2 will be used)
# ssh -t -l APC:7001@ap9302
\end{verbatim}
To log in to a port that does not require authentication, the username is not necessary:

```bash
ssh -t -2 :ttyS1@ap9302
```

**NOTE:** In this case, the file `sshd_config` must be changed in the following way:

```bash
PermitRootLogin Yes
PermitEmptyPassword Yes
```

### The Session Channel Break Extension

This feature is a method of sending a break signal during an SSHv2 terminal session. The implementation is defined by “Session Channel Break Extension: draft-ietf-secsh-break-00.txt” (an IETF Internet-Draft document).

In the previous versions of CPS there was one break length, measured in milliseconds. The current version of CPS supports the parameter `<all/Sx>.break_interval`, which is used with `all.break_sequence` (<all/Sxx>.break_sequence).

The SSHv2-client receives a command (<ssh escape char>B or <break_sequence> [For example: "~break"] 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 then send the break duration (break interval) so the user can configure this value using the command line (" -B <break interval in milliseconds>" For example: "ssh -l <user>:<port> <CPS IP Server > -B <breakinterval in milliseconds>").

#### 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 SSHv2, then the server accepts and treats the "break request" sent by the client. The "break request" defines the break-length in milliseconds. 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 the serial driver.

#### How it works in SSH Client

The SSH client has an option " -B <break interval in milliseconds> ". For example: "ssh -l <user>:<port> <CPS IP Server > -B <break interval in milliseconds>"
When the user types "<ssh-escape>B" (where ssh-escape is "~") or "<break_sequence>" the client sends a "break request" to ssh-server. When CPS calls the ssh-client automatically, it uses the parameter all.break_interval to call the ssh-2 client.

**Configuring the Session Channel Break Extension in SSH Server**

1. Configure the parameters `break_interval` and `break_sequence` in `/etc/portslave/pslave.conf`.

   This can be done by the admin using the vi editor.

   For example:
   
   ```
   all.break_interval         500
   all.break_sequence      ~break
   ```

2. Enter the following commands.

   ```
   #runconf
   #saveconf
   ```

   You can then get the kernel's attention by sending a BREAK by typing `<ENTER>` + (the break sequence configured) + (the key corresponding to a given command [from SysRq]). If you send an invalid command key, the SysRq help will display.

   For example,

   a. Establish an SSH connection to the CPS console port.

      ```
      ssh -l <user>:<socket_port> <CPS_TS_IP>
      ```

   b. You can get the kernel's attention by sending a BREAK signal.

      ```
      <ENTER> + ~break
      ```

   The Result will be:

   ```
   SysRq : HELP : loglevel0-8 reBoot Crash tErm kIll saK showMem Nice powerOff showPc unRaw Sync showTasks Unmount
   ```

   or if you type, for example,

   ```
   <ENTER> + ~breakp
   ```
The Result will be:

SysRq : Show Regs

PId: 0, comm:            swapper
EIP: 0060:[<c010103b>]} CPU: 0
EIP is at default_idle+0x23/0x29
EFLAGS: 00000246    Not tainted (2.6.10-1.771_FC2)
EAX: 00000000 EBX: 00010809 ECX: de0f3000 EDX: 0baf3110
ESI: 00099100 EDI: c03dc120 EBP: 00461007 DS: 007b ES: 007b
CR0: 8005003b CR2: b7ff2000 CR3: 19b6a000 CR4: 000006d0
[<c010108f>] cpu_idle+0x1f/0x34

A.8 The Process Table

The process table shows which processes are running. Type \texttt{ps -a} to see a table similar to the following.

\textbf{Table A-5: Process Table}

<table>
<thead>
<tr>
<th>PID</th>
<th>UID</th>
<th>VmSize</th>
<th>State</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>root</td>
<td>592</td>
<td>S</td>
<td>/sbin/inetd</td>
</tr>
<tr>
<td>31</td>
<td>root</td>
<td>928</td>
<td>S</td>
<td>/sbin/inetd</td>
</tr>
<tr>
<td>32</td>
<td>root</td>
<td>584</td>
<td>S</td>
<td>/sbin/cy_ras</td>
</tr>
<tr>
<td>36</td>
<td>root</td>
<td>1148</td>
<td>S</td>
<td>/sbin/cy_wdt_led wdt led</td>
</tr>
<tr>
<td>154</td>
<td>root</td>
<td>808</td>
<td>R</td>
<td>/ps -a</td>
</tr>
</tbody>
</table>

To restart the \texttt{cy_ras} process, use its process ID or execute the command:

\texttt{# runconf}

This executes the \texttt{ps} command, searches for the \texttt{cy_ras} process id, then sends the signal \texttt{hup} to the process, all in one step. Never kill \texttt{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 CPS. The server is selected by its corresponding number. ts_menu must be executed from a local session (console, Telnet, SSH, a dumb terminal connected to a serial port, etc). Only ports configured for console access (protocols socket_server, socket_ssh, or socket_server_ssh) will be presented. To gain familiarity with this application, run ts_menu - h:

```
# ts_menu -h
```

**USAGE:** ts_menu options {<console port>}

- **-p** : Display TCP port
- **-P** : Use the TCP port instead of just IP
- **-i** : Display Local Ip assigned to the serial port
- **-u <name>** : Username to be used in ssh/telnet command
- **-U** : Always ask for an username
- **-e <[^>]char>** : Escape character used by telnet or ssh
- **-l[c]** : Sorted list ports (c option sort by console server) and exit
- **-auth** : Interactive authentication
- **-ro** : Read Only mode
- **-s** : Show sorted ports

Example: How TS Menu can be used.

```
# ts_menu
```

Master and Slaves Console Server Connection Menu

1 TSJen800
2 test.APC.com
3 az84.APC.com
4 64.186.190.85
5 az85.APC.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 CPS. If the user selects 2 through 5, remote serial ports will be accessed. This is used when there is clustering (one CPS master box and one or more CPS 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 s3alias
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 (s3alias). 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 (if 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 Addresses and TCP ports instead of server names.

CPS: 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 the Local IP Addresses assigned to the serial ports instead of server names.

CPS: 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 the same as the username used to log onto the CPS.

-h : Lists script options.
Appendix B: Upgrades and Troubleshooting

The objective of this appendix is to explain the most common problems that users encounter when using the CPS. This appendix will also show the necessary steps to upgrade the firmware of the CPS unit and how to correctly interpret the CPU LED status.

B.1 Upgrades

Upgrade the CPS whenever there is a bug fix or new features that you would like to have. APC adds six files to the standard Linux files in the /mnt/flash directory when an upgrade is required:

- boot_alt — alternate boot code
- boot_conf — active boot code
- boot_ori — original boot code
- config.tgz — CPS configuration information
- zImage — Linux kernel image

The Upgrade Process

To upgrade the CPS, follow these steps:

1. Log in to the CPS as root.
   
   Provide the root password if requested.

2. Go to the /mnt/flash directory using the following command:
   
   cd /mnt/flash

3. FTP to the host where the new firmware is located.
   
   Log in using your username and password. Navigate 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 /mnt/flash directory must be *zImage*. For 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 CPS searches for a file named *zImage* when booting and there is no room in flash for two *zImage* files.

4. Verify *zImage*.

To make sure the downloaded file is not corrupted and to verify that the file *zImage* was saved in flash, run the following command:

```
md5sum /mnt/flash/zImage
```

The system responds with a message similar to the following:

```
5bcc7d9b3c61502b5c9269cbecd20317 /mnt/flash/zImage
```

5. Check the system’s response against the “.md5” *zImage* text file on the tftp server.

For example, the *zImage* `zvmppccs.1005_qa.cps-k26.md5` text file contains the following information:

```
5bcc7d9b3c61502b5c9269cbecd20317 /tftpboot/zvmppccs.1005_qa.cps-k26
```

If the alphanumeric string matches, the downloaded file is not corrupted.

6. Issue the command `reboot`.

```
# reboot
```

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
```

**CLI method—firmware upgrade**

To upgrade the CPS firmware, follow the steps below:

1. Open the CLI interface by issuing the command:

```
# CLI
```

2. Upgrade the firmware.
To upgrade the CPS’s firmware, you need to know the remote IP address of the FTP server and the path of the image file in the remote server.

For the example, the following settings are used:

- FTP Server: 192.168.100.111
- Path: /images/zImage
- User: john
- Password: john1234

**Example:**

```
cli> administration upgradefw ftpsite 192.168.100.111 username john password john1234 filepathname /images/zImage checksum no
```

3. Return to the main menu by issuing the command.

```
cli> return
```

4. Activate the configuration.

```
cli> config runconfig
```

5. Save the configuration.

```
cli> config savetoflash
```

6. Exit the CLI mode.

To exit the CLI mode and return to CPS’s shell, type the following command:

```
cli> quit
```

7. To make the changes effective, reboot the unit by issuing the command (in the shell prompt):

```
# reboot
```

8. Test the configuration (in the shell prompt).

To check the version of the installed image, run the command:

```
# cat /proc/version
```
B.2 Troubleshooting

Flash Memory Loss

If the contents of flash memory are lost after an upgrade, follow the instructions below to restore your system:

1. Turn the CPS OFF, then back ON.

2. Using the console, wait for the self test messages.
   
   If you get no boot messages, verify that you have the correct setting, otherwise press “s” immediately after applying power to skip an alternate boot code. The CPS will boot using its original boot code.

3. During the self-test, press <Esc> after the Ethernet test.
   
   Testing Ethernet .........................

4. When the Watch Dog Timer prompt appears, press <Enter>.
   
   Watchdog timer ((A)ctive or (I)nactive) [I] :

5. Choose the option Network Boot at the prompt.
   
   Firmware boot from ((F)lash or (N)etwork) [N] :

6. 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).
   
   Boot type ((B)ootp, (T)f tp or Bot(H)) [H] :

7. Enter the filename of the zImage file on the host.
   
   Boot File Name [zvmppccs.1004_ga.cps-k26] :

8. Enter the IP address of the Ethernet interface.
   
   IP address assigned to Ethernet interface [192.168.48.11] :

9. Enter the IP address of the host where the new zImage file is located.
   
   Server's IP address [192.168.49.127] :

10. Accept the default MAC address by pressing <Enter>.
    
    MAC address assigned to Ethernet [00:60:2E:01:6B:61] :

11. When the “Fast Ethernet” prompt appears, press <Enter>.
    
    Fast Ethernet ((A)uto Neg, 100 (B)tH, 100 Bt (F), 10 B(t)F, 10 Bt (H)) [A] :
The CPS 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 process in section B.1 to save the new zImage file into flash again.

**NOTE:** Possible causes for the loss of flash memory:
- Downloaded wrong zImage file
- Downloaded as ASCII instead of binary
- Problems with flash memory

If the CPS booted properly, the interfaces can be verified using `ifconfig` and `ping`. If `ping` does not work, check that the cables are connected correctly, then check the routing table (using the command `route`).

The file `/etc/config_files` contains a list of files that are affected by the `saveconf` and `restoreconf` commands. At the command prompt, issue the command `cat /etc/config_files` to see the list of files that are available in the flash and are loaded into the RAMDisk at start-up.

**IMPORTANT!** If a file listed in `/etc/config_files` is modified, the CPS administrator must execute the command `saveconf` before rebooting the CPS or the changes will be lost. If a file is created (or a filename is altered), add the filename to this file before executing `saveconf` and rebooting.

**IMPORTANT!** For help with configuration problems, contact APC Customer Support at a number on the back cover of this manual. Before you call, execute the command:

```
# cat /proc/version
```

Note the Linux version and CPS version written to the screen. This will speed the resolution of most problems.

## Hardware Test

A hardware test called `tstest` is included with the CPS firmware. It is a menu-driven program, run by typing `tstest` at the command prompt. The various options are described in this section of the manual.

**NOTE:** The CPS should not be tested while in use, because the test will inactivate all ports. Inactivate all processes that may use the serial ports: `inetd`, `sshd`, `cy_ras`, and `cy_buffering`. 
Following are the hardware test steps:

1. signal_ras stop.
2. Perform all hardware tests needed.
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 crossover cable between two ports to be tested).

When ttest senses the presence of the cable or connector, the following information is displayed on your screen.

HW Test/Linux

This tool is for internal use ONLY!

It should not be used if the serial port is configured and/or running.

Press any key to continue or <ESC> to cancel ==> 

To start the port test:

1. Select option 2, Test Asynchronous Ports.
2. Enter the media type [(0) RS232 (1) RS485-FULL] :
   (For example, select 0 for RS232)

When the test runs 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.

Port conversation

The port conversation 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 loopback connector nor the cross cable are necessary. Enter the number of the port to be tested and begin the test.

<table>
<thead>
<tr>
<th>State</th>
<th>DTR</th>
<th>DCD</th>
<th>DSR</th>
<th>RTS</th>
<th>CTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>ON</td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>OFF</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>

*Figure B.1 - Initial Test*

First, type Ctrl+D to see the X in the DTR column change 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 you type Ctrl+D, the Xs in the first three columns will move as shown below because the test is receiving the DTR signal sent through the DCD and DSR pins.

<table>
<thead>
<tr>
<th>State</th>
<th>DTR</th>
<th>DCD</th>
<th>DSR</th>
<th>RTS</th>
<th>CTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>ON</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>OFF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>

*Figure B.2 - Second screen, showing changed positions*

When you type Ctrl+R, 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 CPS 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 CPS unstable,
- After a configuration change which leaves the CPS inoperative or unstable.

Type the word “single” (with a blank space before the word) during boot using a console connection.

**IMPORTANT:** This cannot be done using a Telnet or other remote connection.

The initial output of the boot process:

```
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: 00E17000 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 CPS waits approximately 10 seconds for user input. This is where the user should type “<sp>single” (press the spacebar, then type 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:

1. Edit the file(s) causing the problem with vi, then execute the commands:

   ```bash
   [root@CPS root]# saveconf
   [root@CPS root]# reboot
   ```
2. Reset the configuration by executing the following commands:

   [root@CPS root]# defconf

   The following warning message displays:

   WARNING: This will erase all of your current configuration and restore the system's factory default configuration. This action is irreversible and the CPS must be rebooted to apply that.

   Are you sure you wish to continue? (y/N)

   Enter y or N. If you entered ‘y’, type reboot at the following prompt.

   [root@CPS root]# reboot

   The system reboots and displays the following message.

   Sep 27 19:39:09 src_dev_log@CPS reboot: The system is going down. Rebooted by root.

   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. The \texttt{gw} and \texttt{mask} parameters are optional.

   [root@CPS root]# config_eth0 ip 200.200.200.1 mask 255.255.255.0 gw 200.200.200.5

   Edit the file (or files) causing the problem with \texttt{vi}, then execute the commands:

   [root@CPS root]# saveconf
   [root@CPS root]# reboot

   Check the DNS configuration in the file \texttt{/etc/resolv.conf}, then download the kernel image using the \texttt{ftp} command.
Using a Different Speed for the Serial Console

The serial console is originally configured to work at 9600 bps. To change that configuration:

1. Run bootconf. The following screen appears:

   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]
   Maximum rate of incoming bytes per second [0]:

   Type <Enter> for all fields except the Console Speed.

2. When presented the following line, enter Y to save the changes in flash.

   Do you confirm these changes in flash ( (Y)es, (N)o (Q)uit ) [N] :

3. Log out and log in again to use the console at the new speed.

Setting the Maximum Number of Bytes Received by the Interface

To avoid CPU overload due to too many bytes received by the interface at one time, set a limit for the rate of bytes received. The bootconf utility offers a way of setting this limit. The default limit is 0, which disables the function. For optimum performance, set the value to 50000.

Configure the maximum allowed rate of bytes to be received, or enter 0 to disable the feature completely and allow all bytes to be processed at any rate.

**NOTE:** Values greater than 50000 will not cause harm, but may make the system more susceptible to storms. Smaller values can cause the feature be triggered by normal equipment traffic.
To set a limit of bytes received by the interface per second:

1. Run bootconf.

The following screen appears:

<table>
<thead>
<tr>
<th>Current configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAC address assigned to Ethernet [00:60:2e:00:16:b9]</td>
</tr>
<tr>
<td>IP address assigned to Ethernet interface [192.168.160.10]</td>
</tr>
<tr>
<td>Watchdog timer ((A)ctive or (I)nactive) [A]</td>
</tr>
<tr>
<td>Firmware boot from ((F)lash or (N)etwork) [F]</td>
</tr>
<tr>
<td>Boot type ((B)ootp, (T)ftp or Bot(H)) [T]</td>
</tr>
<tr>
<td>Boot File Name [zvmppcts.bin]</td>
</tr>
<tr>
<td>Server's IP address [192.168.160.1]</td>
</tr>
<tr>
<td>Console speed [9600]</td>
</tr>
<tr>
<td>(P)erform or (S)kip Flash test [P]</td>
</tr>
<tr>
<td>(S)kip, (Q)uick or (F)ull RAM test [F]</td>
</tr>
<tr>
<td>Fast Ethernet ((A)uto Neg, (1)00 BtH, 100 Bt(F), 10 B(t)F, 10 Bt(H)) [A]</td>
</tr>
<tr>
<td>Fast Ethernet Maximum Interrupt Events [0]</td>
</tr>
<tr>
<td>Maximum rate of incoming bytes per second [0]</td>
</tr>
</tbody>
</table>

2. Press <Enter> for all fields except the field **Maximum rate of incoming bytes per second**.

3. Between the brackets following the **Maximum rate of incoming bytes per second** field, type the maximum amount of bytes that can be received by the interface per second.

   A value of 0 disables the feature. Enter a value of 50000 for optimum performance.

---

**NOTE:** Values greater than 50000 will not cause harm, but may make the system more susceptible to storms. Smaller values can cause the feature be triggered by normal equipment traffic.

4. When presented the following line, enter Y to save the changes in flash.

   Do you confirm these changes in flash ( (Y)es, (N)o (Q)uit ) [N] :
B.3 LEDs

CPU LEDs

If the CPU is operating normally, the status LED blinks consistently (one second on, one second off). If this is not the case, an error has been detected during the boot. Interpret the blink pattern using the following table:

Table B-1: CPU LED Code Interpretation

<table>
<thead>
<tr>
<th>Event</th>
<th>CPU LED Morse code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal Operation</td>
<td>S (short, short, short . . . )</td>
</tr>
<tr>
<td>Flash Memory Error—Code</td>
<td>L (long, long, long . . . )</td>
</tr>
<tr>
<td>Flash Memory Error—Configuration</td>
<td>S, L</td>
</tr>
<tr>
<td>Ethernet Error*</td>
<td>S, S, L</td>
</tr>
<tr>
<td>No Interface Card Detected</td>
<td>S, S, S, L</td>
</tr>
<tr>
<td>Network Boot Error</td>
<td>S, S, S, S, L</td>
</tr>
<tr>
<td>Real-Time Clock Error</td>
<td>S, S, S, S, S, L</td>
</tr>
</tbody>
</table>

*The Ethernet Error 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 rear panel of the CPS has serial, console, and Ethernet connectors with LEDs.

Ethernet connector LED functionality

- *Col (collision)*—Shows collision on the LAN every time the unit tries to transmit an Ethernet packet.
- *DT/LK (data transaction/link state)*—The green LED is Data Transaction (DT) activity and the yellow LED is LinK (LK) state. DT is hardware-controlled and flashes when data is transmitted to or received from the LAN. LK is software-controlled and remains steady if the LAN is active.
- *100*—If 100BT is detected, the LED is activated. If 10BT is detected, the LED turns off.
Console connector LED functionality

- **CP**—CPU activity. This LED flashes at approximately 1-second intervals.
- **P1**—Power supply #1 ON.
- **P2**—Power supply #2 ON.

Serial connector LED functionality

- **LK**—Software-controlled. Remains steady if the LAN is active.
- **DT**—Hardware-controlled. Flashes when data is transmitted to or received from the LAN.

Administration Parameters in the CLI Interface

Some of the procedures described above can be configured using the CLI interface. See “Boot configuration parameters” on page 339 for information about boot configuration, or see “Administration menu” on page 340 for information about administration configuration.

Boot configuration parameters

To configure boot parameters, access the menu:

```
cli>config administration bootconfig
```

When you enter this menu, you can configure the following parameters:

- **boottype**—Chooses the network boot type. Valid values: tftp, bootp, or both.
- **bootunit**—Sets where the unit will boot from. Valid values: flash and network.
- **consolespeed**—To configure the console speed. Valid values: 115200, 57600, 38400, 19200, 9600, and 4800.
- **ethernetip**—Temporary IP address assigned to the Ethernet interface.
- **ethernetmode**—Fast Ethernet mode. Valid values: auto, 10F, 10H, 100H, 100F.
- **filename**—File name of the image placed in the tftp server.
- **flashtest**—Enables or disables the flash test. Valid values: full and skip.
- **maxevents**—Maximum number of Ethernet events handled at once.
- **ramtest**—Chooses the type of ram test. Valid values: full, quick, and skip.
- **tftpserver**—Sets up the IP address of the tftp server.
- **wdt**—Enables or disables the watch dog timer (wdt).
**Administration menu**

The administration section of the CLI interface is divided into 3 parts: Session Management, Backup Configuration, and Firmware Upgrade (see “CLI method—firmware upgrade” on page 328 for firmware upgrade information).

**Session Management:** To manage sessions, access:

```
cli>administration sessions
```

This menu lets you do following:

- **kill**—Cancels a connection to the serial port <n>
- **list**—Lists the current sessions

**Backup Configuration:** It is possible to save/restore configurations to/from a FTP server. To configure backup configuration, access the menu:

```
cli>administration backupconfig
```

The following options can be set up:

- **loadfrom**—When loading configuration from a server it is necessary to specify the following parameters: server IP address <serverip>; username <username>; password <password>; path <pathname>.
- **saveto**—The same parameters of the **loadfrom** command must be specified for the **saveto** command.

For example, the command below will load configuration from a server with the IP address 192.168.0.1, username “john”, password “john1234” and the configuration file located at /home/configuration.

```
backupconfig>loadfrom serverip 192.168.0.1 pathname /home/configuration
username john password john1234
```
Appendix C: Cabling and Hardware Information

This appendix lists the hardware specifications of the CPS and the characteristics of cables and connectors.

C.1 General Hardware Specifications

The power consumption and heat dissipation, environmental conditions, and physical specifications of the CPS are listed below.

Table C-1: CPS Products Power Consumption and Heat Dissipation

<table>
<thead>
<tr>
<th>Model</th>
<th>Power (Watts)</th>
<th>Heat Exchange (BTU/hr.)</th>
<th>Power (Watts)</th>
<th>Heat Exchange (BTU/hr.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AP9301</td>
<td>18</td>
<td>61.5</td>
<td>28</td>
<td>95.6</td>
</tr>
<tr>
<td>AP9302</td>
<td>22</td>
<td>75.1</td>
<td>30</td>
<td>102.5</td>
</tr>
<tr>
<td>AP9303</td>
<td>24</td>
<td>81.0</td>
<td>32</td>
<td>109.3</td>
</tr>
</tbody>
</table>

Table C-2: CPS environmental conditions

<table>
<thead>
<tr>
<th>Environmental Information</th>
<th>AP9301</th>
<th>AP9302</th>
<th>AP9303</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating Temperature</td>
<td>50°F to 112°F (10°C to 44°C)</td>
<td>50°F to 112°F (10°C to 44°C)</td>
<td>50°F to 112°F (10°C to 44°C)</td>
</tr>
<tr>
<td>Relative Humidity</td>
<td>10–90%, non-condensing</td>
<td>10–90%, non-condensing</td>
<td>10–90%, non-condensing</td>
</tr>
</tbody>
</table>
The following section has all the information you need to purchase cables for the CPS. It focuses on information related to the RS-232 interface, which applies not only to the CPS but also to any RS-232 cabling.

### The RS-232 Standard

RS-232, also known as RS-232C or EIA RS-232, refers to a standard defined by the Electronic Industries Association in 1969 for serial communication. RS-232 was defined to connect Data Terminal Equipment (DTE, usually a computer or terminal) to Data Communication Equipment (DCE, usually a modem):

\[
\text{DTE} \rightarrow \text{RS-232} \rightarrow \text{DCE} \rightarrow \text{communication line} \rightarrow \text{DCE} \rightarrow \text{RS-232} \rightarrow \text{DTE}
\]

RS-232 is now most commonly used to connect DTE devices directly (without modems or communication lines in between). While that was not the original intention, it is possible with a crossover cable.

<table>
<thead>
<tr>
<th>Physical Information</th>
<th>AP9301</th>
<th>AP9302</th>
<th>AP9303</th>
</tr>
</thead>
<tbody>
<tr>
<td>External Dimensions</td>
<td>8.5 in</td>
<td>17 in</td>
<td>17 in</td>
</tr>
<tr>
<td></td>
<td>4.75 in</td>
<td>8.5 in</td>
<td>8.5 in</td>
</tr>
<tr>
<td></td>
<td>1 in</td>
<td>1.75 in</td>
<td>1.75 in</td>
</tr>
<tr>
<td>Weight</td>
<td>1.6 lb</td>
<td>6 lb</td>
<td>6.2 lb</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Safety Information</th>
<th>AP9301</th>
<th>AP9302</th>
<th>AP9303</th>
</tr>
</thead>
<tbody>
<tr>
<td>Approvals</td>
<td>FCC &amp; CE, Class A</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
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 baud rate. Both sides must be configured with the same parameters.

**NOTE:** If you have the correct cable and your serial connection still does not work, verify that the communication parameters are the same for both sides.

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 19,200 bps (used in most automation and console applications) to 115,200 bps (used by the fastest modems).

**Cable Length**

Successful RS-232 data transmission depends on many variables that are specific to each environment. The following general rules are empirical and have safety margins.

- If the speed is lower than 38.4 kbps, you can use any cable up to 30 meters long (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.
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 the 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 use this connector (especially when compact size is important). RS-232 interfaces on DB-9 connectors always use the same standard pin assignment.

The telephone-type modular RJ-45 plug and jack are 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. Because there is no standard RS-232 pin assignment for RJ-45 connectors, each equipment vendor has its own pin assignments.

Most connectors have two versions. Connectors with pins are said to be “male” and connectors with holes are said to be “female”.

Table C-5: Cables and their pin specifications

<table>
<thead>
<tr>
<th>RS-232 Signal</th>
<th>Name/Function (Input/Output)</th>
<th>DB-25 pins (Standard)</th>
<th>DB-9 pins (Standard)</th>
<th>RJ-45 pins (APC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chassis</td>
<td>Safety Ground</td>
<td>1</td>
<td>Shell</td>
<td>Shell</td>
</tr>
<tr>
<td>TxD</td>
<td>Transmit Data (O)</td>
<td>2</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>RxD</td>
<td>Receive Data (I)</td>
<td>3</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>DTR</td>
<td>Data Terminal Ready (O)</td>
<td>20</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>DSR</td>
<td>Data Set Ready (I)</td>
<td>6</td>
<td>6</td>
<td>8</td>
</tr>
<tr>
<td>DCD</td>
<td>Data Carrier Detect (I)</td>
<td>8</td>
<td>1</td>
<td>7</td>
</tr>
<tr>
<td>RTS</td>
<td>Request To Send (O)</td>
<td>4</td>
<td>7</td>
<td>1</td>
</tr>
<tr>
<td>CTS</td>
<td>Clear To Send (I)</td>
<td>5</td>
<td>8</td>
<td>5</td>
</tr>
<tr>
<td>GnD</td>
<td>Signal Ground</td>
<td>7</td>
<td>5</td>
<td>4</td>
</tr>
</tbody>
</table>
Straight-Through vs. Crossover Cables

The RS-232 interface was originally intended to connect a DTE (e.g., a computer, a printer, or another serial device) to a DCE (e.g., a 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 a crossover cable, you can use RS-232 to connect two DTEs (the most common configuration for modern applications).

A crossover cable (also known as a null-modem cable) is used to connect two DTEs directly, without modems or communication lines in between them. 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.

Identifying the Cable or Adapter to Use

Determine the proper cable for your DCE or DTE applications using the following table, then purchase cables and adapters. To use CAT5 cables with RJ-45 connectors, pair the CAT5 cabling with the appropriate cross-adapter.
### Cables

**Table C-6: Cables to use**

<table>
<thead>
<tr>
<th>Application</th>
<th>APC Part Number</th>
<th>Cable Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DCE DB-25 Female • Analog Modems • ISDN Terminal Adapters</td>
<td>AP9308</td>
<td>APC Console Server DB25M to RJ45 Cable</td>
</tr>
<tr>
<td>DTE RJ-45 All APC Console Ports</td>
<td>AP9307</td>
<td>APC Console Server DB9F to RJ45 Cross Cable</td>
</tr>
<tr>
<td></td>
<td>AP9309</td>
<td>APC Console Server DB25F to RJ45 Cross Cable</td>
</tr>
<tr>
<td></td>
<td>AP9317</td>
<td>APC KVM/Console Server RJ45 to APC Switched Rack PDU RJ12 Cable</td>
</tr>
</tbody>
</table>

**NOTE:** A derivative of this cable is included with the CPS for initial installation and configuration.

**NOTE:** Use this cable for Power Management of APC Switched Rack PDUs

### Adapters

<table>
<thead>
<tr>
<th>APC Part Number</th>
<th>Adapter Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AP9310</td>
<td>APC Console Server DB25F to RJ45 Cross Adapter</td>
</tr>
<tr>
<td>AP9311</td>
<td>APC Console Server DB25M to RJ45 Cross Adapter</td>
</tr>
<tr>
<td>AP9313</td>
<td>APC Console Server DB9F to RJ45 Cross Adapter</td>
</tr>
<tr>
<td>AP9314</td>
<td>APC Console Server DB9M to RJ45 Cross Adapter</td>
</tr>
</tbody>
</table>
### APC Part Number

<table>
<thead>
<tr>
<th>APC Part Number</th>
<th>Adapter Description</th>
</tr>
</thead>
</table>
| AP9315          | APC Console Server RJ45 to RJ45 Cisco/SUN Adapter  
**NOTE:** This adapter attaches to a CAT3 or CAT5 network cable. It is typically used in console management applications to connect APC products to a SUN Netra server or to a Cisco product. At one end of the adapter is the CAT.5e Inline Coupler Box with a female RJ-45 connector. A 3-inch-long black SUN Netra-labeled cord extends from the Coupler Box to a RJ-45 male connector. |
| AP9316          | APC Console Server to Serial APC UPS DB9 Adapter |
Appendix D: Copyrights

The APC Console Port 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 APC Console Port Server and a reference to their maintainers. The copyrights notices required in some packets are placed in the /COPYRIGHTS directory of the APC Console Port 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 1.0

**Cron**
Paul Vixie's cron version 3.0.1.
paul@vix.com

**DHCPDCD**
http://www.phystech.com/download/dhcpd.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 OpenSwan IPsec version 2.3.0
http://www.openswan.org

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 5.2.1.2
http://sourceforge.net/projects/net-snmp/

NTP
NTP client
http://doolittle.faludi.com/ntpclient/

OpenSSH
OpenSSH version 4.1p1
http://www.openssh.org
COPYRIGHT: This product includes software developed by the University of California, Berkeley and its contributors.
OpenSSL
OpenSSL Project version 0.9.8
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

UCD-SNMP
SourceForge Net-SNMP project version 5.2.1.2
http://sourceforge.net/projects/net-snmp/
WEBS
GoAhead WEBS version 2.1 (modified)
http://goahead.com/webserver/webserver.htm
Copyright (c) 20xx GoAhead Software, Inc. All Rights Reserved

ZLIB
zlib version 1.2.3
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 Port Server (CPS)

A CPS 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, which makes 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**

IP packet filtering is 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**

A CPS feature that allows capture of console data even when there is no one connected to the port.

**Profile**

Usage setup of the CPS: 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 quickly. 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. RS-232 is still widely used for low-speed data communication.

Secure SHell (SSH)

SSH has the same functionality as Telnet (see “Telnet” on page 356), 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” on page 353).

Shadow password

Normally, each user's password is stored, encrypted, in the file /etc/passwd. This file must be readable by all users so that certain system functions will operate correctly. This means that copies of user's encrypted passwords are easily obtained, making it possible to run an automated password-guessing program against them. Shadow passwords, on the other hand, store the encrypted passwords in a separate highly-protected file, making it much more difficult to crack passwords.

SNMP

Short for Simple Network Management Protocol, a set of protocols for managing complex networks. 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: www.webopedia.com)
**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. (Source: www.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 2 U of rack space.

**X.509**

A widely used standard for defining digital certificates. X.509 is an ITU recommendation, which means that it has not yet been officially defined or approved for standardized usage. As a result, companies have implemented the standard in different ways. For example, both Netscape and Microsoft use X.509 certificates to implement SSL in their Web servers and browsers, but an X.509 Certificate generated by Netscape may not be readable by Microsoft products, and vice versa.