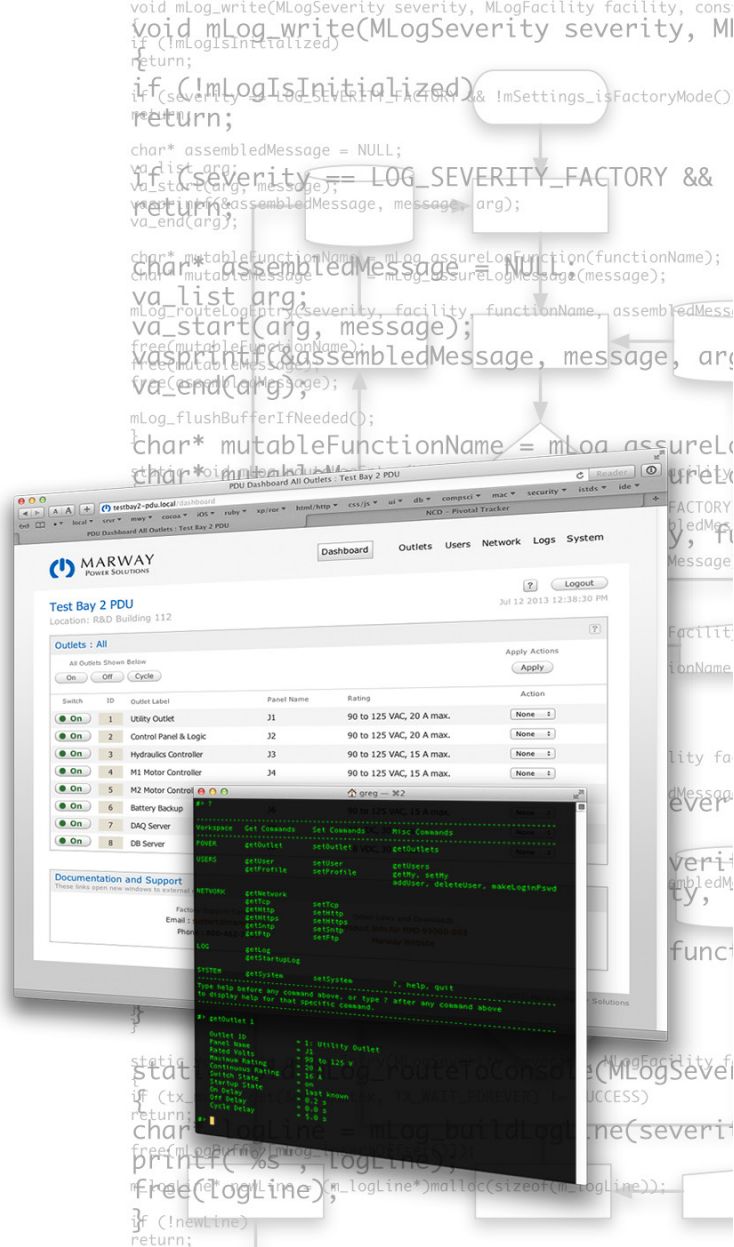




MPD Series RCM Software™ for Switched Outlets

Owner's Guide and Reference
Software Version 1.2.x

September 2015 : P/N 501014-002



© 2013-2015, Marway Power Systems, Inc. All rights reserved.

RCM Software™ is a trademark of Marway Power Systems, Inc.

MPD Series™ is a trademark of Marway Power Systems, Inc.

All other trademarks are the property of their respective owners.

Global Support Contacts

Web: www.marway.com

Email: support@marway.com

sales@marway.com

Phone: 800-462-7929 (7am–5pm PST)

There may be updates to this documentation and the software it describes at:

www.marway.com/mpd/rcm_software

MPD Series RCM Software

Owner’s Guide and Reference

Software Version 1.2.x

Contents.....	3	Personal Profile.....	17	Outlet Control	29
Getting Started.....	7	Permissions	17	Outlet Switching and Adjustable Delays.....	29
Initial TCP/IP Setup	9	Network Settings	18	Staggering Startups to Curb Inrush	30
Change root User Password	10	System Settings.....	19	Staggering Startups for Dependencies	30
If you cannot log into the PDU	10	Command Line Interface Tour	20	Using Cycle Delay	31
Set the Current Date and Time.....	11	Login.....	20	Switching Sets of Outlets	31
Feature Tour.....	13	Built-in Help	21	Outlet Control from the Web	31
Web Browser Interface.....	13	Command Syntax.....	21	Outlet Switching.....	31
Command Line Interface	13	Aliases and Variants	22	Outlet Delays.....	32
Power Terms	13	FTP and the File System	23	Outlet Control from the Command Line	32
Browser Interface Tour.....	14	Software Operation	25	Outlet Switching.....	32
Main Menu.....	14	Login.....	25	Outlet Delays.....	33
Web Dashboard	15	HTTP/HTTPS (“Web”) Login.....	26	Viewing Log Data.....	33
Outlet Control.....	15	Startup Notices.....	26	Logged Events.....	33
Power Settings	16	Documentation and Support.....	26	Logging of Outlet Switch Events.....	33
User Settings.....	17	Telnet Login.....	27	Log Format.....	33
Authentication	17	SSH Login.....	27	System Log vs. Startup Log.....	34
Access Methods	17	Viewing Outlet Status.....	28	Web Viewing of Log Data.....	35
		Outlet Status on the Web Dashboard	28	Command Line Viewing of Log Data	35
		Outlet Status by Command Line.....	28		

User Management.....	37	Command Syntax.....	56	getStartupLog.....	68
Root User.....	37	Command Aliases.....	57	System Commands.....	69
Authentication.....	37	Command Variants.....	57	getSystem.....	69
Passwords in the Settings File.....	37	Outlet Commands.....	58	setSystem.....	69
Authorization.....	37	getOutlet.....	58	RESTful API Reference.....	71
Access Methods.....	37	setOutlet.....	58	What is REST and RESTful API?.....	71
Permissions.....	38	User Commands.....	59	The Gist of REST.....	71
Auditing.....	38	getProfile (a.k.a getMy).....	59	Programming for a RESTful API.....	72
Activity Logging.....	38	makeLoginPswd.....	59	Why Use the RESTful API?.....	72
Profile.....	39	setProfile (a.k.a setMy).....	59	HTTP Verbs.....	72
Adding, Deleting, Editing Users.....	39	getUser.....	60	Responses.....	73
Software Settings.....	41	setUser.....	60	Authentication API.....	74
Adjust Settings with the Web UI.....	41	addUser.....	61	API Resources.....	74
Adjust Settings with the CLI.....	41	deleteUser.....	61	/session.....	74
Adjust Settings in the Files.....	42	Network Commands.....	62	POST /session.....	74
System Settings.....	43	getTcp.....	62	DELETE /session.....	74
Network Settings.....	44	setTcp.....	62	/outlet.....	75
User Settings.....	48	getHttp.....	63	GET /outlet:id/label.....	75
Outlet Settings.....	53	setHttp.....	63	GET /outlet:id/switch.....	75
Labels.....	53	getHttps.....	64	GET /outlet:id/rated_amps.....	75
Outlet Switch Delays.....	53	setHttps.....	64	GET /outlet:id/rated_volts.....	76
Outlet Startup State.....	53	getSntp.....	65	GET /outlet:id/startup_state.....	76
Outlet State Logging and Saving.....	53	setSntp.....	65	GET /outlet:id/on_delay.....	76
Command Line Reference.....	56	getFtp.....	66	GET /outlet:id/off_delay.....	77
Built-in Help.....	56	setFtp.....	66	GET /outlet:id/cycle_delay.....	77
		getSntp.....	67	PATCH /outlet:id/switch.....	77
		setSntp.....	67	PATCH /outlet:id/startup_state.....	78
		Log Commands.....	68		
		getLog.....	68		

PATCH /outlet:id/on_delay.....78
 PATCH /outlet:id/off_delay78
 PATCH /outlet:id/cycle_delay79
 /system79
 GET /system/1/label.....79
 GET /system/1/location.....79
 Settings Files Reference81
 Know What You’re Doing.....81
 Accessing Settings Files81
 File Format and Syntax82
 Name/Value Pairs82
 Objects and IDs82
 Object Groups of Settings.....83
 File Identity and Version.....83
 systemSettings.txt84
 networkSettings.txt84
 userSettings.txt.....84
 powerSettings.txt.....85
 factorySettings.txt86
 Firmware Updates88

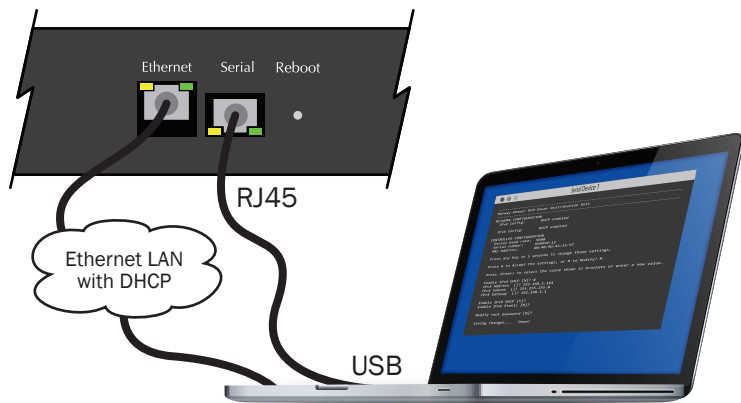
MPD Series RCM Software

Getting Started

Getting Started

To prepare for initial networking setup, you will need:

- A computer with either a USB or DB9 port and an Ethernet port.
- An Ethernet LAN with DHCP services enabled.
- Two Ethernet cables (one for the PDU, one for the computer) to connect to the LAN (usually by connecting to a switch, hub, or router).
- A “Cisco-style” serial cable to connect the RJ45 serial port to the serial input of your computer (USB or DB9). See the sidebar *Serial Cables*.
- A software application for terminal emulation.
 - For Microsoft Windows, a program named HyperTerminal is commonly used.
 - Mac OS users commonly use a program named CoolTerm.
 - If you’re a Linux user, we’ll assume you already have a favorite selected.
- The serial port configured with:
 - 9600 baud, 8 data bits, 1 stop bit, no parity



TCP/IP NETWORKING IS ON BY DEFAULT.

It is highly recommended, for security purposes, that the root user password be updated before or immediately when the system is connected to a network. When possible, use an Ethernet connection local to a single computer, or private LAN, to do the initial setup.

If the PDU is connected to a network without the password being updated, the PDU may be accessible by any person or script familiar with the default password of Marway's PDUs which may result in unintended and unauthorized access.

Serial Cables

The Marway PDU includes an RJ45 “Cisco-style” serial port. Even though it uses an RJ45, this is not an Ethernet cable. To use this port you will need a Cisco-style cable. Most come with RJ45 on one end and DB9 on the other end. If you have a DB9 connector on your computer, you’re ready. If your computer has only USB for serial communication, you can use an additional converter to go from DB9 to USB, and use the two cables together, or you can use a specialized RJ45 to USB cable. These aren’t as common, but are certainly convenient.

Next, with the PDU powered off, follow these steps.

- Connect the Ethernet cables.
- Connect the serial cable between the computer and the PDU.
- Open the serial communication application on the computer, and start a new serial session (make sure the configuration settings are correct).
- Plug the PDU into its power source.

After 10–15 seconds, the serial display should be similar to the screen illustration shown at the right (the actual numbers will be different).

- When you see the “Press any key...” prompt, press Return, the space key or, as it says, any other key. This enters what is called the serial dialog which is used to configure TCP/IP and the root user password.
- As a confirmation, a second prompt will ask that you press M to make modifications. Pressing M will then start a series of setup prompts as shown on the next page (shown after each one has been completed).

If you miss the 5-second window to change settings, restart the software by pressing the recessed reset button, or typing `setSystem restart` at the serial CLI prompt, or power cycling the PDU.

Marway RCM Power Distribution Unit

NETWORK CONFIGURATION

MAC Address:	00:40:9D:75:A8:17
IPv4 DHCP:	Enabled
IPv4 Address:	192.168.1.97
IPv4 Subnet:	255.255.255.0
IPv4 Gateway:	192.168.1.1
IPv6 DHCP:	Enabled

Press any key in 5 seconds to change these settings.

Initial TCP/IP Setup

Configuring TCP/IP the first time, or any time it is added to a new network, must be done using the serial port as the unit starts up. Follow the steps in the *Getting Started* section above to enter the serial dialog.

After confirming you want to **Modify** the PDU settings, the first TCP/IP prompt will be presented (with either a **[Y]** or a **[N]** depending on the current configuration):

Enable IPV4 DHCP [Y]?

- Type N (recommended) to manually enter the IP address, the subnet mask, and the gateway address, and press Return.
- Type Y to have DHCP (not normally recommended) assign the IP address, the subnet mask, and gateway address, and press Return.

It is highly recommended that IPv4 DHCP be disabled, and that a manual IPv4 address be used so that it will always be known when logging into the PDU by web or by command line.

If you are manually entering the IP details, a separate prompt will appear for each of the PDU's IP address, the subnet mask, and the gateway address settings. In the sample screen capture to the right, a new IPv4 address was manually entered. The pre-existing defaults for the subnet and gateway prompts were accepted as is by pressing Return with no new values typed.

```
-----
Marway RCM Power Distribution Unit
-----

NETWORK CONFIGURATION
MAC Address:      00:40:9D:75:A8:17
IPv4 DHCP:        Enabled
IPv4 Address:     192.168.1.97
IPv4 Subnet:      255.255.255.0
IPv4 Gateway:     192.168.1.1
IPv6 DHCP:        Enabled

Type any key in 5 seconds to change startup settings.
Type M to Modify the settings, or C to Cancel: M

For each setting, enter a new value, or type Return to accept the default.

MODIFY SETTINGS

RESTORE FACTORY DEFAULTS

Reset all settings to factory defaults [N]?

MODIFY ETHERNET SETTINGS

Use DHCP for IPv4 [Y]? N
IPv4 IP address  [192.168.1.97]? 192.168.1.10
IPv4 Subnet mask [255.255.255.0]?
IPv4 Gateway address [192.168.1.1]?

Use DHCP for IPv6 [Y]?
Enable static IPv6 [N]?

MODIFY ROOT USER

Update the Root Password [N]?

SAVING CHANGES...

Done.

RESTARTING PDU...
```

Change root User Password

If you can log into the PDU

If you know either the root user password, or any user/password combination which grants access to editing user profiles, you can edit the root password using the normal user profile editing methods of the web interface or the command line. There is no need to access the PDU using the serial port.

Using the web interface, log into the PDU, open the Users list from the main menu, then edit the user named root.

Using the command line interface, log into the PDU using Telnet or SSH, then execute the command `setUser root password "abc"` where abc is replaced with the new password.

If you cannot log into the PDU

If the system has been configured before, and the root password or any other password with full privileges is not known, the root password will have to be changed using the serial port connection. Follow the steps in the *Getting Started* section above to enter the serial dialog.

The TCP/IP prompts will be presented first. Press Return for each prompt to accept the current value (or change them if needed). When prompted to modify the root password, type Y and press Return. The new password will need to be entered twice.

FACTORY DEFAULT USER ACCOUNT

Account: root

Password: pP8*word

MODIFY ETHERNET SETTINGS

Use DHCP for IPv4 [Y]? N
IPv4 IP address [192.168.1.10]?
IPv4 Subnet mask [255.255.255.0]?
IPv4 Gateway address [192.168.1.1]?

Use DHCP for IPv6 [Y]?
Enable static IPv6 [N]?

MODIFY ROOT USER

Update the Root Password [N]? Y

Password must include:
at least one upper case from A-Z
at least one lower case from a-z
at least one numeral from 0-9
at least 8 characters
at least 4 letters (cannot be mostly numerals)

Enter the new Root Password: *****
Re-enter the new Root Password: *****

SAVING CHANGES...

Done.

RESTARTING PDU...

Set the Current Date and Time

After having TCP/IP configured and the root user password changed, the next task should be to verify that the PDU is aware of the current time. While not critical for operation, having the correct time will provide the logging system and the alert system with accurate time stamps.

There are two ways to have the PDU aware of the correct time. The “best” way is to have the PDU automatically connect to a network time service known as SNTP. This method is best if the PDU will be connected to the internet, or to a local network which has a time server available. This will keep the PDU’s time updated and accurate, and if you have more than one PDU, will ensure that all PDUs have the same time.

If a network time server is not available to the PDU, then another way to adjust the time settings is to manually set the PDU’s internal clock with the current date and time. All computer-based clocks like the one in the PDU will run a little fast or slow, and may need adjusting from time to time if you want it to be as accurate as possible.

Note that if the PDU is connected to an SNTP server, the local clock will not be manually editable since SNTP will be automatically keeping the date and time updated.

SNTP configuration is done in the Network tab of the web interface. The internal clock is configured in the System tab of the web interface.

Time Servers (SNTP)

The factory default SNTP servers are public volunteer servers known to work at the time of shipment, but are not run by Marway, and are not guaranteed to be available. Check with your IT administrator to see if there’s an alternative preferred set of servers to use.

If a time server is not going to be used, change the addresses of the two time servers to be 0.0.0.0 or blank.

TimeZone Settings

For details on time zone settings, find the settings in the section “**Network Settings**” on page 44.

MPD Series RCM Software

Feature Tour

Feature Tour

Marway's RCM Software for MPD Series PDUs offers multiple interfaces to monitor and/or control the unit over Ethernet:

- a web browser based graphical interface ("web UI")
- a command line interface ("CLI") over Telnet or SSH
- a RESTful API over HTTP for automated outlet control

Additionally, a serial port is used for the first-time setup of TCP/IP settings. See ["Initial TCP/IP Setup" on page 9](#) for details. The same serial port also offers the same command line capability as Telnet/SSH.

This section of the *User Guide* is an introductory tour of the major features of the web UI and CLI. It presents screenshots along with short descriptions of several of the interface and operation features of the RCM Software.

Web Browser Interface

The browser interface has been designed for reasonably broad compatibility with several of the most commonly used browsers in versions released within the past couple of years. If you notice any specific compatibility problems, please report them to Marway support.

The browser interface can be used over HTTP or HTTPS. It allows for remote relay switching, user management, and editing all adjustable settings of the PDU. Note that many UI elements (fields and buttons) will display help tips if you let the mouse pointer hover over the element for a second or two. Additionally, many display panels on the web pages have their own help button on the right side of the panel title bar.

Command Line Interface

The command line interface works over Telnet and SSH. It allows for remote relay switching, user management, and editing all adjustable settings of the PDU.

Power Terms

In Marway's RCM products, the following terms are used throughout the documentation and interfaces.

- Outlet : a female power connector used to provide point-of-use power for the user's application.
- Circuit : a branch in the power distribution protected by a circuit breaker or fuse (or the whole unit if there is no breaker or fuse).
- Line : a pair of conductors (wires) across which exists a voltage potential.
- Input : a single conductor on which a current flows.

In Marway's products, Lines and Inputs are the wires directly connected to the power feed which we call the power Source. A PDU will have one or more power Sources. Each one is made of Lines and Inputs. Each Source may be branched into one or more Circuits, and each circuit would have one or more Outlets (with a variety of possible connector styles).

Browser Interface Tour

Login

The login page is fairly straight forward. One noteworthy feature aside from the obvious login functionality is the Startup Notices panel. Any errors or warnings generated during startup will be listed in this panel. If startup went as expected, the “OK” will be displayed as the screen illustration shows.

Main Menu

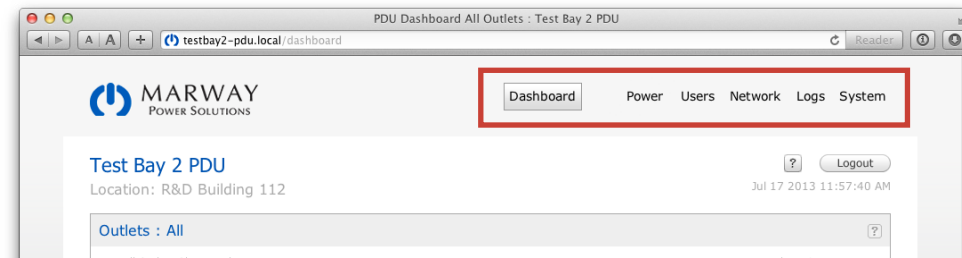
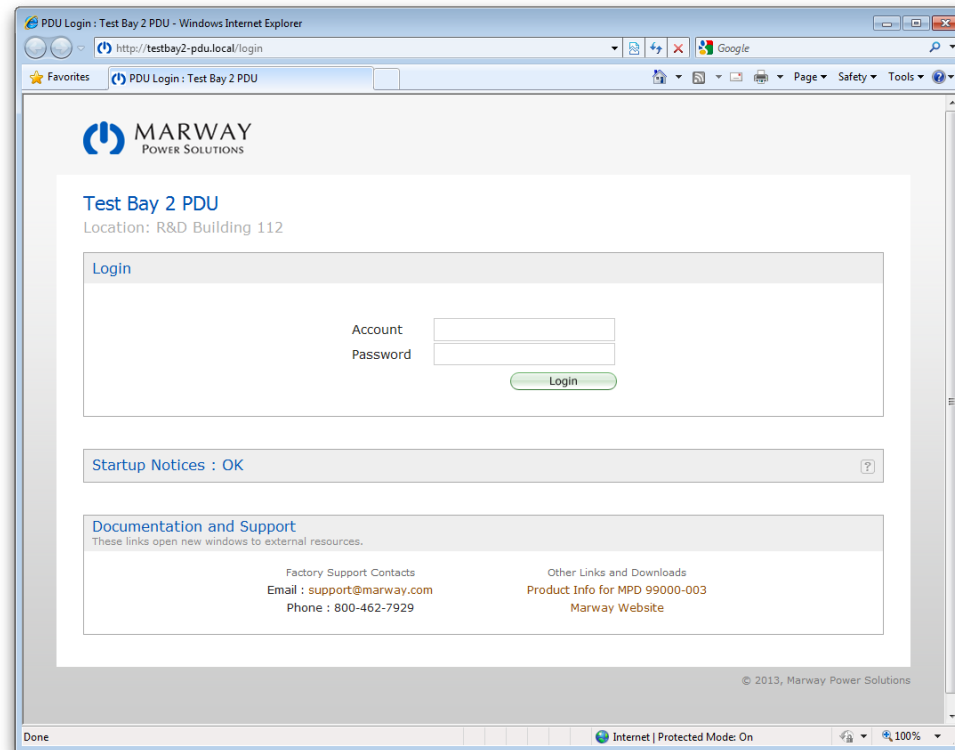
After the login page, the web browser interface is organized into the following major sections which are labeled in the browser’s main menu tabs as:

Data Viewing

- Dashboard : provides outlet control. Other than logs, Dashboard is where virtually all information viewing is performed. All other main menu tabs are primarily for adjusting settings.
- Logs : display of the system and startup logs

Settings Adjustments

- Power : settings for outlet labels, switching delays, and startup state
- Users : settings for user profiles, authentication, and authorization
- Network : settings for supported networking protocols and features
- System : settings for unit labeling and clock, command for restart, and miscellaneous system-level information



Web Dashboard

The Dashboard is where all remote outlet control is located. Other than logs, Dashboard is where virtually all information viewing is performed. All other main menu tabs are primarily for adjusting settings.

Below the main menu, the top of the dashboard includes a user-defined unit label and location on the left. These are defined in the System tab, and allow the PDU to have a unique and meaningful identification. On the right, the Help and Logout buttons sit above a time stamp of when the viewed web page was last loaded. This top section of the page is common to all web pages.

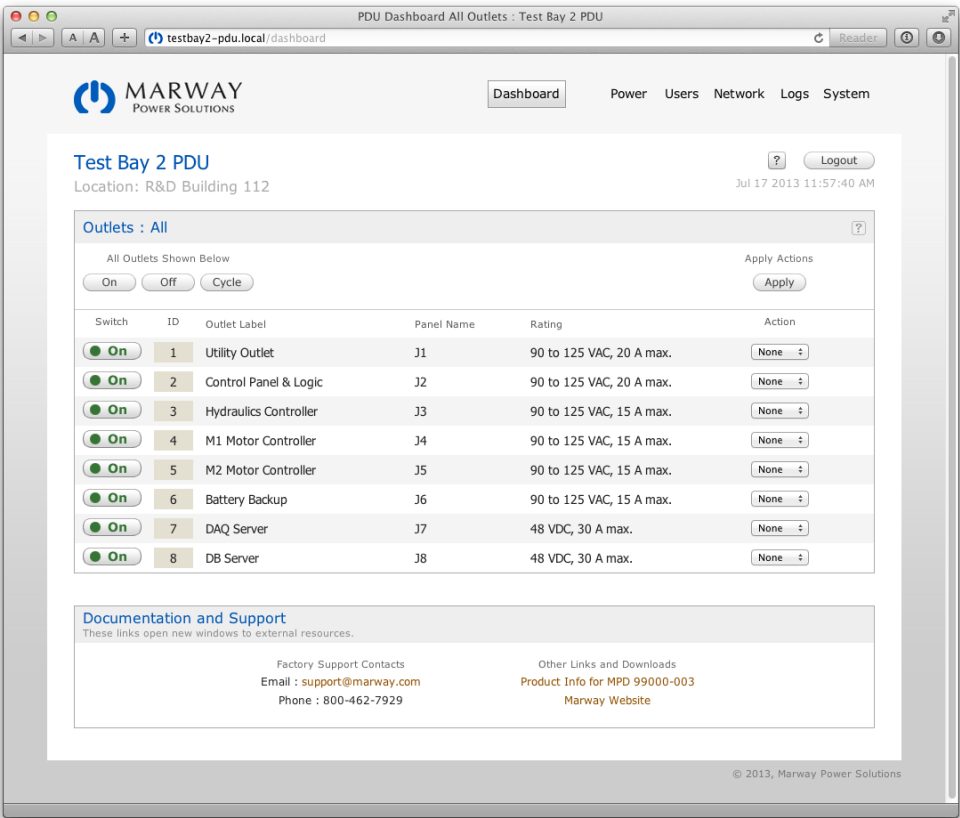
Outlet Control

The web Dashboard offers three ways to manipulate outlet state. The first is an All Outlets set of buttons to affect all outlets with a single command to turn them on, turn them off, or cycle them.

The second method for switching outlets is the Action column of the Dashboard Outlets panel. Each outlet has a popup menu to choose On, Off, or Cycle. Each outlet can be set to any of these options. When all choices have been made, pressing the Apply button at the top of the column will issue each command in sequence.

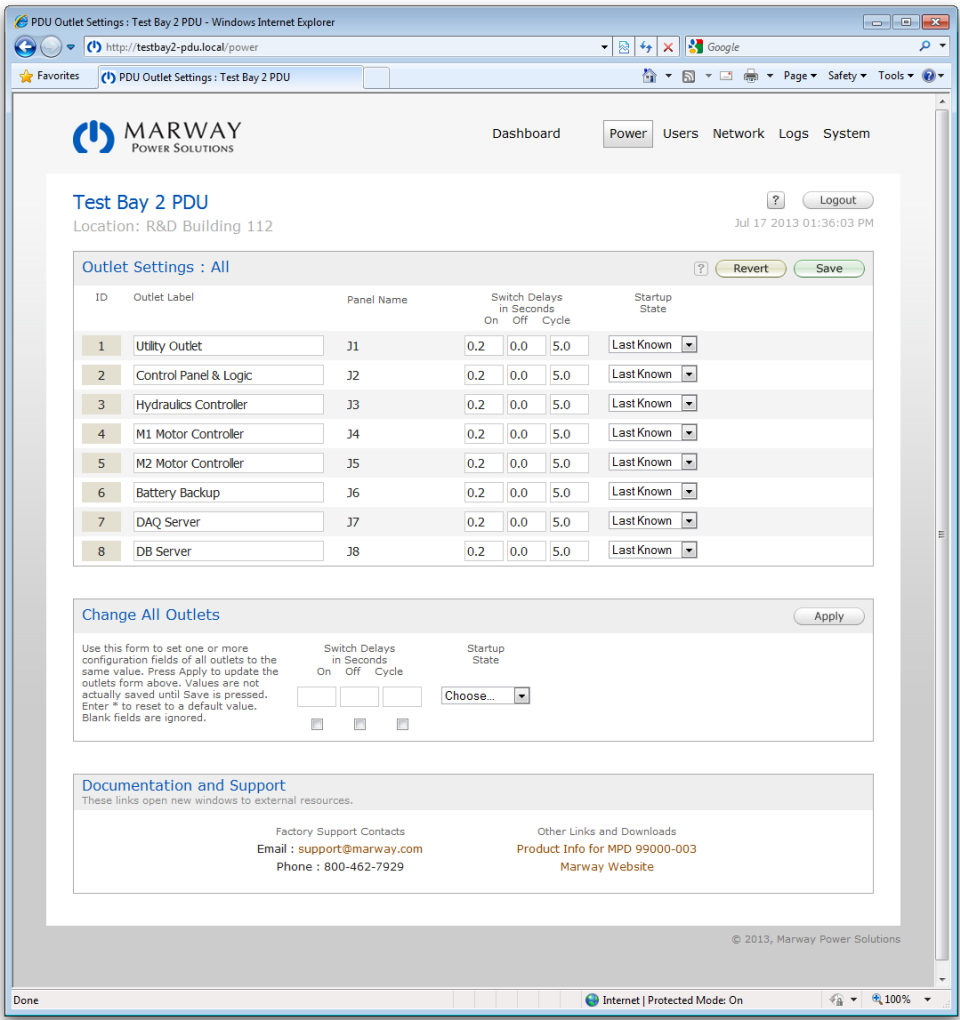
The third method for switching outlets is the Switch column buttons which provide simple and immediate manual On and Off switching control.

In addition to outlet control, the Dashboard displays the electrical load specifications of the outlet, its panel name, and an arbitrary user-entered label to identify the purpose of, or equipment connected to, the outlet.



Power Settings

The main menu Power tab displays the adjustable settings for all outlets. This includes a descriptive label, delays for on, off, and cycle actions, and a startup state configuration. The Startup State option determines what the outlet state will be set to shortly after the PDU is powered up, or after a software restart.



User Settings

Authentication

Authentication determines the credentials used to login into the PDU (via web or CLI). This is an account name and a password.

Access Methods

The Access Methods section determines whether the user can access the system via the web UI, the CLI, and the settings files. If the none of the options are selected, this effectively disables a user without having to delete him.

Personal Profile

This section allows an administrator to keep minimal contact information on board if needed. The Company and Job Role field can be used to identify third-party vendors who may be allowed access perhaps for support.

Permissions

The RCM software does not assign permissions based on user “groups.” Rather, each user can be assigned unique permissions to allow greater flexibility.

Show List Add User

Users : List						
Edit	Login Name	User Name	Email	Access	Logins	Last Login Del
	root	System Admin	sysmin@test.com	Web CLI FTP	104	2013-07-17 12:02:44
	lester	Lester Tester	les@test.com	Web	137	2013-07-12 13:32:10

PDU User Update : Test Bay 2 PDU

testbay2-pdu.local/users/update

Dashboard Power Users Network Logs System

Test Bay 2 PDU

Location: R&D Building 112

Jul 17 2013 12:04:31 PM

Logout

Show List Add User

Users : lester (Lester Tester)

Send Test Alert Cancel Save

Settings marked with * are required.

Authentication

Login Namelester

Password * (Rules)

Password Again *

Password Hint *p8

Access Methods

Allow Login To

☒ Web Pages (http, https)

☐ Command Line (telnet, ssh)

☐ Config Files (ftp)

Personal Profile

First & Last NameLesterTester

Alert Addressles@test.com

SMS Address

Company NameAcme Test Automation

Job RoleTest Technician

Company Phone800-999-4321

Direct Phone

Status

Last Login2013-07-12 13:32:10

Login Count137

Permissions

Power Settings

☒ View Power Settings

☒ Control Outlet On/Off

☒ Edit Outlet Labels

☒ Edit Outlet Delays

Network Settings

☒ View Network Settings

☐ Edit TCP/IP

☐ Edit SNMP

☐ Edit HTTP

☐ Edit FTP

☐ Edit SMTP

Logs Settings

☒ View Logs

Users Settings

☐ View Users

☐ Edit Users

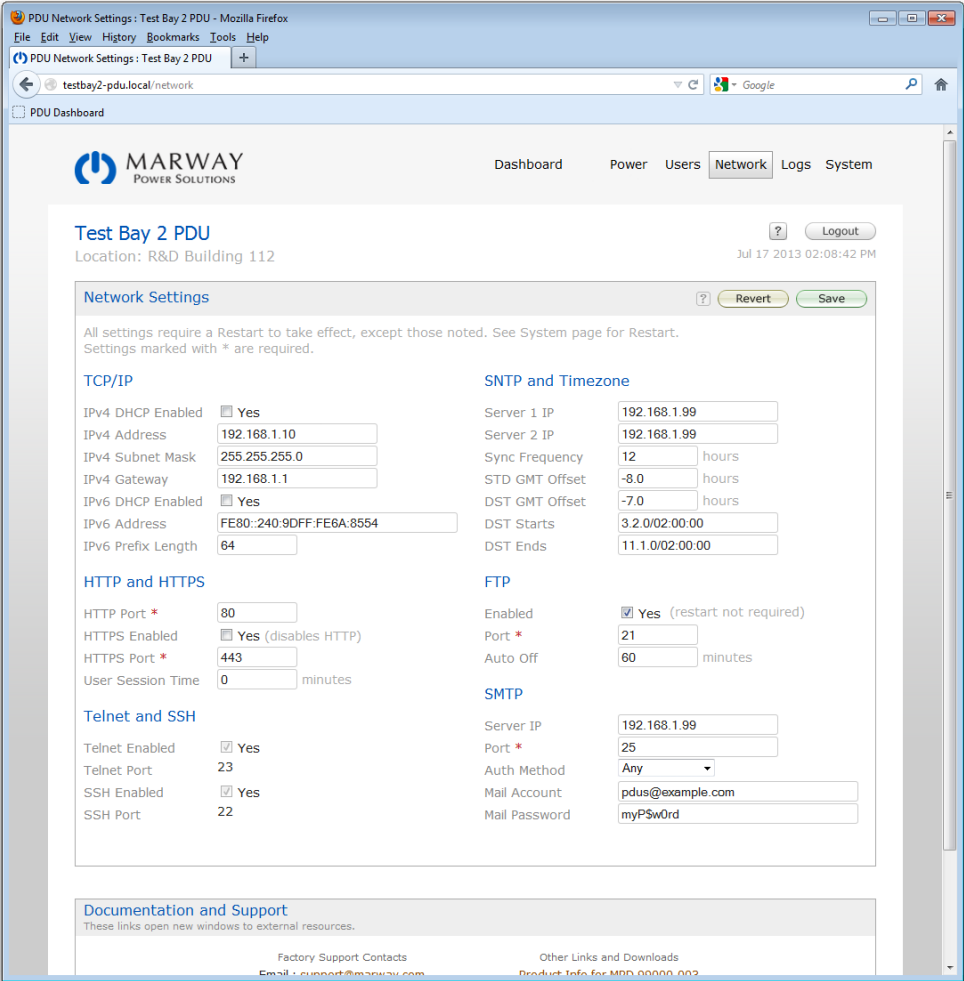
System Settings

☒ View System Settings

☐ Edit System Settings

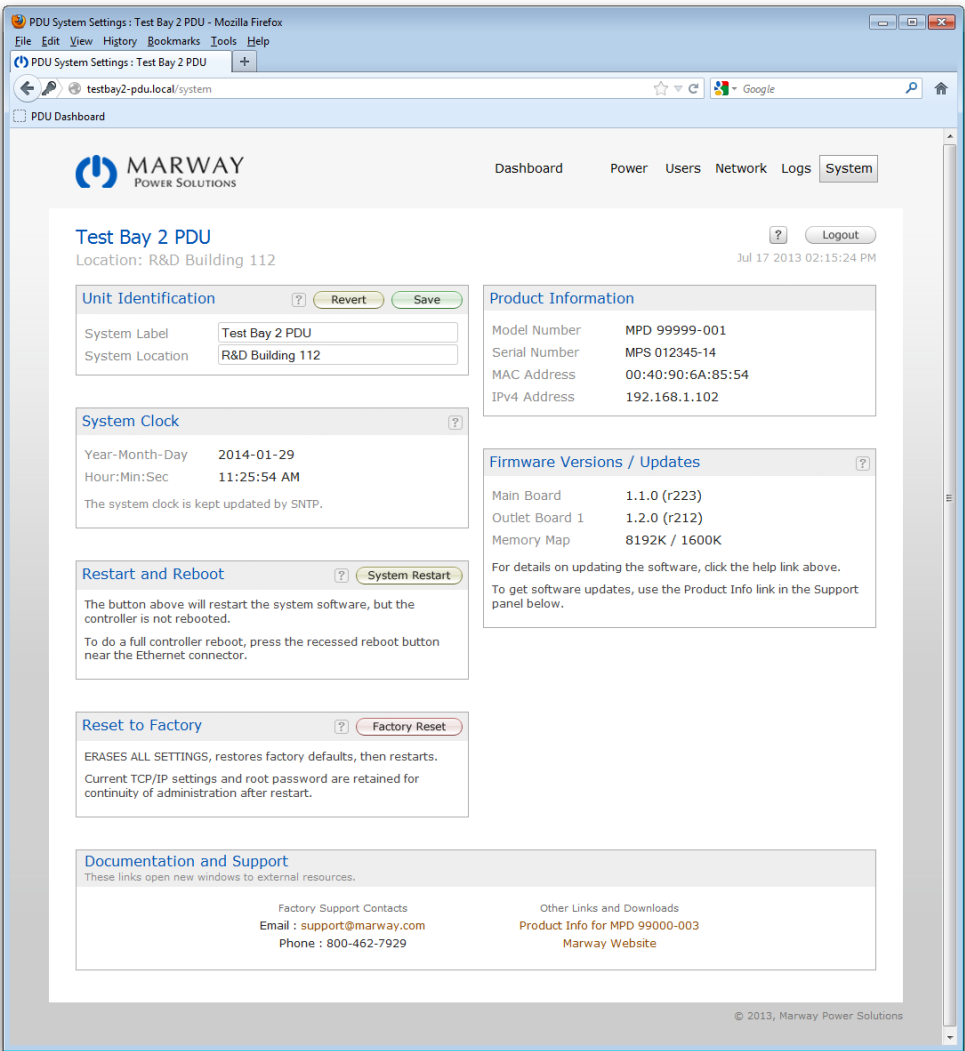
Network Settings

This page presents the settings details for all supported network protocols. Assuming you're familiar with these protocols, most settings should be self explanatory. One unique feature the RCM has is an auto-off for the FTP server. Since the need to have FTP on would be for rare access to the log file or settings files, FTP can be enabled, when needed and set to auto disable after a period time. It's a small convenience and potential security feature.



System Settings

The System page offers the place to set the system label and location. It also provides tools to restart the system. A Restart is needed after most Network setting changes. If the PDU is not connected to an SNTP server, the date and time settings will be editable.



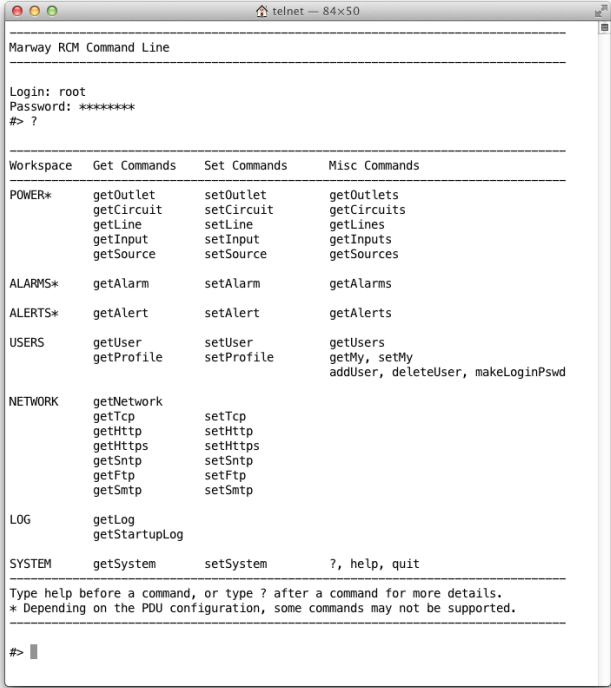
Command Line Interface Tour

After login, the CLI is organized into several get/set command sets such as these examples listed below:

- getOutlet / setOutlet
- getUser / setUser and addUser / deleteUser
- getTcp / setTcp
- getHttp / setHttp and getHttps / setHttps
- getSntp / setSntp
- getFtp / setFtp
- getLog / getStartupLog
- getSystem / setSystem

Login

The login task is fairly straight forward, for Telnet and SSH, enter a user account name at the login prompt, and a password.



Built-in Help

At any point after login, typing `help` or `?` at the prompt will display the top-level help which is a list of commands. Each command can also be followed by a `?` such as `getSystem?` (with or without a space) to display detailed usage help for that command. The command help will identify the syntax of the command, various attribute options, and some examples.

Command Syntax

The majority of commands follow a common set or get pattern. The following are typical get commands:

- `getOutlet 8 switch`
- `getUser lester email`
- `getSystem location`

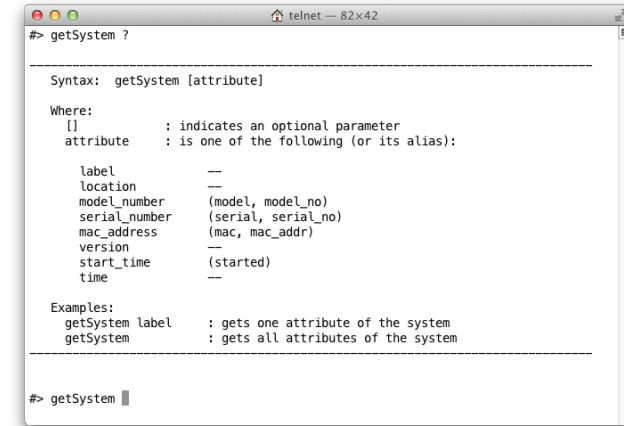
These are typical set commands:

- `setOutlet 1 switch on`
- `setUser lester email "les@example.com"`
- `setSystem location "Aisle 6 Rack 4"`

You'll notice the pattern of `get` and `set` are identical except for the last parameter of the `set` command which passes the new value. Otherwise, both use the same syntax of:

commandName instanceID attribute [setValue]

Where *commandName* is something like `getOutlet`, `getUser`, etc. The *instanceID* is the numerical ID of an outlet, or the login name of a user. A few commands do not need an instance ID such as `getSystem`/`setSystem` where there's always one implied instance. The *attribute* is the specific detail being requested or set. Finally, if the command is a `set`, the fourth parameter is the new value.



```
#> getSystem ?

Syntax: getSystem [attribute]

Where:
[]      : indicates an optional parameter
attribute : is one of the following (or its alias):

label      ---
location   ---
model_number (model, model_no)
serial_number (serial, serial_no)
mac_address (mac, mac_addr)
version     ---
start_time  (started)
time        ---

Examples:
getSystem label      : gets one attribute of the system
getSystem             : gets all attributes of the system

#> getSystem
```

Aliases and Variants

Commands are flexible in two ways. First, many command attributes have aliases. These aliases are revealed in the command help. For example, the `getSystem` command has an attribute `model_number`. That attribute name may be substituted with `model_no` or even `model`. So, `getSystem model` is the same command as `getSystem model_number`.

The second flexible feature is command variants. These are different ways of using the command to get different results—namely whether to get/set information for a single instance or multiple instances.

A get command is used like these examples:

```
getOutlet 4 switch
getUser lester email
```

will return one attribute for one specific instance.

A get command is used like these examples:

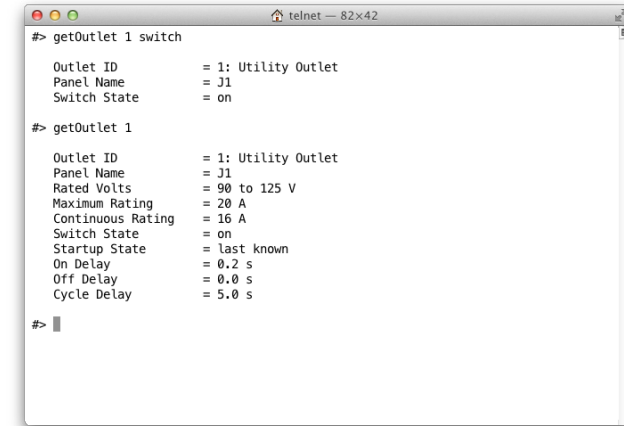
```
getOutlet 4
getUser lester
```

will return all attributes for one specific instance.

A get command is used like these examples:

```
getOutlet
getUser
```

will return all attributes for all instances.



```
telnet — 82x42
#> getOutlet 1 switch
Outlet ID      = 1: Utility Outlet
Panel Name     = J1
Switch State   = on

#> getOutlet 1
Outlet ID      = 1: Utility Outlet
Panel Name     = J1
Rated Volts    = 90 to 125 V
Maximum Rating = 20 A
Continuous Rating = 16 A
Switch State   = on
Startup State  = last known
On Delay       = 0.2 s
Off Delay      = 0.0 s
Cycle Delay    = 5.0 s

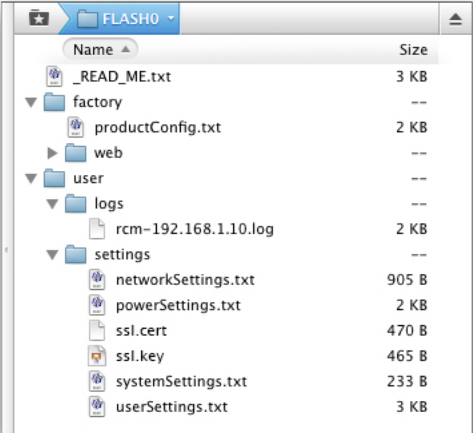
#> |
```

FTP and the File System

An on-board file system is used to store files containing all of the settings, and a small log of recent events. This file system is accessible through FTP.

Settings files can be downloaded for archiving, and even uploaded after being edited offline. This allows preconfiguration of many details which can then be uploaded to multiple PDUs.

A special file named productConfig.txt can be downloaded, but cannot be edited. This file defines exactly which features the PDU has.



MPD Series RCM Software

Software Operation

Software Operation

This chapter covers the typical operation features of the RCM Software which primarily revolves around the web interface Dashboard page, and a few CLI commands. If you have not yet read them, you may want to read these sections for an overview:

- For Setup instructions, refer to “Getting Started” on page 7.
- For an overview of the organization and features of the web browser interface refer to “Browser Interface Tour” on page 14.
- For an overview of the command line interface refer to “Command Line Interface Tour” on page 20.

Login

Access to the PDU features and settings is available using HTTP, HTTPS, Telnet, or SSH. Additionally, access to the file system is available using FTP.

Each protocol has a unique login interface, but all use the same account and password. The account is sometimes referred to as the user name or login name. Regardless of the label used, it refers to the PDU account as entered in the User Settings form.

For discussion purposes, assume the PDU has been configured with an IP address of 192.168.1.10. We will further assume the predefined account root will be used for the examples.

CHANGE THE ROOT PASSWORD BEFORE DOING ANYTHING ELSE.

It is highly recommended, for security purposes, that the root user password be updated before or immediately when the system is connected to a network. When possible, use an Ethernet connection local to a single computer, or private LAN, to do the initial setup.

If the PDU is connected to a network without the password being updated, the PDU may be accessible by any person or script familiar with the default password of Marway's MPD PDUs which may result in unintended and unauthorized access to the PDU.

DNS Names vs IP Addresses

Your network administrator may have a domain name assigned to the PDU IP address, and you may have been instructed to use that name. It may be something like www.pduname.com or pdu.company.com to access a PDU across the Internet, or something like deptserverspdu.local to access the PDU on your local building network. These are DNS names, and can be used instead of the numeric IP address.

HTTP/HTTPS (“Web”) Login

To access a PDU by the web, type the IP address in a web browser’s address field such as http://192.168.1.10. If the PDU has been configured to use https, then use https:// instead of http:// at the start of the address.

The web login form includes four panels of information. Below the Marway logo, the top of the main white panel identifies the PDU’s label and location. In the example screen capture, the label displays as Test Bay 2 PDU. This label and the location description below it are assigned by a PDU administrator.

Below the PDU label is the Login panel which includes the form to enter the user’s account and password. The account is a unique name assigned to the user specifically for logging in. There is one default account named root which automatically has access to every feature and setting. The root account must be used to do the initial setup of all settings since there would be no other users defined yet.

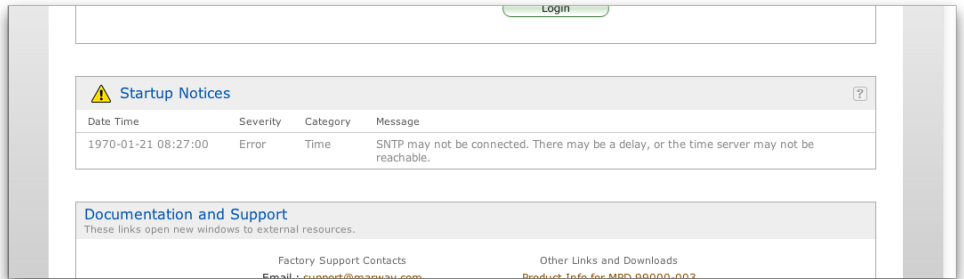
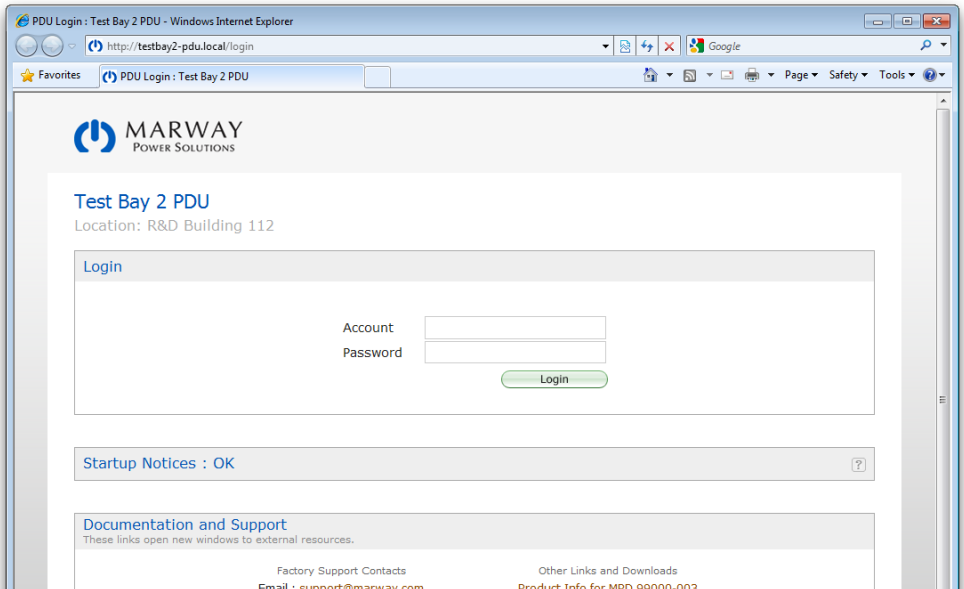
Startup Notices

This panel will show any errors encountered during the startup of the PDU. The sample login page shown did not have any startup errors, and the panel title shows an “OK” status.

The types of errors that may show up here would include problems with missing or malformed settings files in the file system, hardware errors detected by the system, and network connectivity errors.

Documentation and Support

This panel is repeated on every page of the web interface for the PDU. It simply offers useful links to support services and online reference material.



Telnet Login

Telnet is a command line program already installed on most Linux, Unix, and Mac OS X computers. It is also installed on Windows XP, but not on newer versions of Windows. If you're using Windows Vista or newer, search the Internet for instructions on how to install and enable the Telnet client.

To log into the PDU using Telnet, open your operating system's command line interface (e.g. Windows' Command Prompt program, or OS X's Terminal application). Type the word `telnet` followed by an IP address or DNS name for the PDU like these examples:

```
telnet 192.168.1.10
telnet pdu.mycompany.com
telnet deptserverspdu.local
```

SSH Login

SSH is a command line program already installed on most Linux, Unix, and Mac OS X computers. It is not installed by default on Windows computers. For Windows computers, you will need to install an SSH program. There are several popular programs for this, consult your IT or network administrator if you have one. The rest of this discussion assumes the installation has been completed.

To log into the PDU via SSH, open your operating system's command line interface. Type the word `ssh` followed by a space, the PDU account name, an `@` symbol, and an IP address or DNS name like these examples:

```
ssh root@192.168.1.10
ssh lester@pdu.mycompany.com
ssh jsmith@deptserverspdu.local
```

The Difference Between Telnet and SSH

The short version is that SSH is much more secure than Telnet. With Telnet, the information exchanged between your computer and the PDU is not encrypted. It is possible, under specific circumstances, for a skilled person to intercept that information. A worst case scenario would allow them to capture your login credentials and gain access to the PDU as though they were you.

With SSH, all traffic including the login is encrypted. This makes interception of the information pretty much useless (though nothing in the digital world is ever absolutely secure).

The Difference Between HTTP and HTTPS

If you substitute HTTP for Telnet above, and HTTPS for SSH, the story is pretty much identical. HTTPS is secure, where HTTP is not. However, the PDU determines whether you need to use HTTP or HTTPS based on the Network settings.

Which Should You Use?

Wherever possible, using SSH and HTTPS are the more secure choices. Yet, the others still exist. Many people choose to use Telnet and HTTP over the more secure options when they communicate over a closed, local network where the risk of someone capturing data to perform an attack is low (depending on the environment of course). If you're accessing the PDU across the Internet, use HTTPS and SSH. If you choose to use HTTP or Telnet, we assume you know why you're doing that.

Viewing Outlet Status

The status of all outlets are available to view on the web Dashboard, and through the CLI command `getOutlet`.

Outlet Status on the Web Dashboard

The web interface Dashboard page is the focal point for reviewing all remote features on the PDU. There is no equivalent view on the command line, so the web interface is the most convenient way to see the most data at one time.

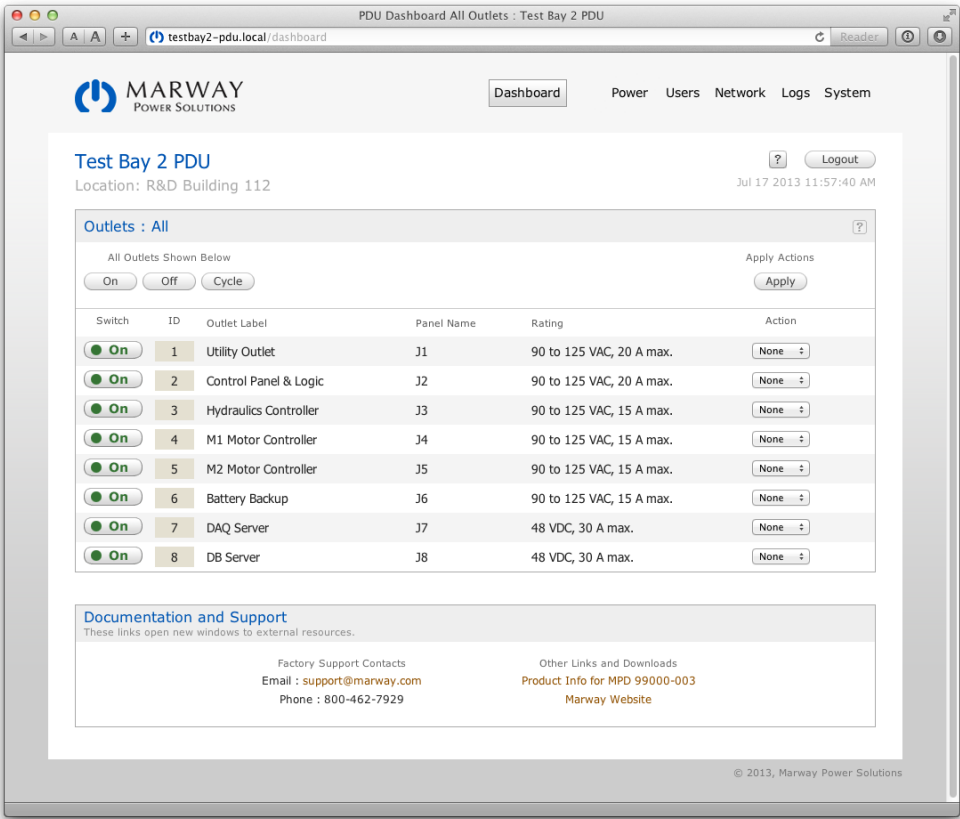
The web view at the right shows the default Dashboard layout for a PDU with 8 remote-controlled outlets. The left-most column displays an “indicating switch” for each outlet.

On and Off are normal operational values for the indicators. If the button displays ???, this indicates the system cannot detect the control circuitry for the outlet switching. Another possible display is a red blinking dot on the button which indicates that the actual state of the outlet does not match what the software expects it to be.

For outlet switching capabilities, refer to “Outlet Control” on page 29 for details. The label for each Outlet (set to “Outlet 1” etc. by default) can be edited. See “Outlet Settings” on page 53 for details regarding changing the label as well as the amps setpoint settings.

Outlet Status by Command Line

Access to outlet status on the command line is available with the `getOutlet` command. The `getOutlet` command displays several attributes. Refer to



“getOutlet” on page 58 for additional command details, but the basic command formats to view power data are:

getOutlet [id] returns all attributes
getOutlet [id] switch returns only switch status

The label for each Outlet may be edited. See “setOutlet” on page 58 for details on changing the label and the amps setpoint settings.

Outlet Control

Outlets may be controlled using the web Dashboard, and through the CLI command setOutlet. Additionally, adjustable delays in response to on and off commands can be set using the web Outlets page or the CLI setOutlet command. See “Outlet Settings” on page 53.

Outlet Switching and Adjustable Delays

Outlets may be switched on, off, and cycled. Cycle switches an outlet off, then back on with a single command. If an outlet was already off, it will be switched on. In addition to the obvious on and off action, switching also includes the ability to add delays to these actions.

An On Delay will delay the on action. If the On Delay of Outlet 1 were set to 1.5 seconds, then when an operator sends an on command to that outlet (by web or CLI), the system would count out an approximate 1.5 seconds before dispatching that command to the outlet controller.

An Off Delay will delay the off action. It works just like an on delay in adding a delay between the time a command is given and when it is performed.

Login: root
Password: *****

#> ?

Command to view outlet status.

Workspace	Get Commands	Set Commands	Misc Commands
POWER*	getOutlet getCircuit getLine getInput getSource	setOutlet setCircuit setLine setInput setSource	getOutlets getCircuits getLines getInputs getSources
ALARMS*	getAlarm	setAlarm	getAlarms
ALERTS*	getAlert	setAlert	getAlerts
USERS	getUser	setUser	getUsers

#> getOutlet 1

Outlet ID

= 1: Utility Outlet

Panel Name

= J1

Rated Volts

= 90 to 125 V

Maximum Rating

= 20 A

Continuous Rating

= 16 A

Switch State

= on

Startup State

= last known

On Delay

= 0.2 s

Off Delay

= 0.0 s

Cycle Delay

= 5.0 s

A Cycle Delay is a unique value used only between the off and on steps of a cycle command. During a Cycle command, the off step will be delayed by the Off Delay setting. The on step will be delayed by the larger of the On Delay or Cycle Delay. This can be confusing, but the intent is to prioritize the needs of attached equipment which is sensitive to startup order. This may result in longer then necessary delays for some scenarios, but it ensures the minimum delays are honored for sequence-sensitive cases.

Staggering Startups to Curb Inrush

The most common use of On Delay is to stagger the power up of attached devices to avoid an excessive inrush which might trip a circuit breaker if all devices started at the same time. Most electrical systems suitable for the PDU have very short in rush periods, and the default On Delay value should work for most applications.

Staggering Startups for Dependencies

Another use of On Delay is to intentionally connect several devices in sequential outlets, then stagger the on times to meet the needs of a startup dependency between those particular pieces of equipment.

For example, suppose we have 3 devices. Device A depends on Device B, and Device B depends on Device C, and each device should be powered for 20 seconds before the next unit is started. Set the devices as follows:

- Device C in Outlet 1 with an On Delay of 0.
- Device B in Outlet 2 with an On Delay of 20 seconds.
- Device A in Outlet 3 with an On Delay of 20 seconds.

The PDU powers each outlet in sequence. Therefore, in the above scenario, after the PDU is powered or rebooted, the sequence of events will be:

- Outlet 1 is switched on immediately
- A delay of 20 seconds elapses for the On Delay of Outlet 2.

- Outlet 2 is switched on.
- A delay of 20 seconds elapses for the On Delay of Outlet 3.
- Outlet 3 is switched on.

Using Cycle Delay

The Cycle Delay is most typically used to ensure that when a piece of equipment is switched off, enough time passes to allow capacitors to fully discharge, hard drives to stop spinning, and other factors to come to a “full stop” before power is reapplied. Those details and the time needed, if any, are specific to the equipment involved.

Switching Sets of Outlets

When multiple outlets are targeted to be switched (for example, the web UI’s Action controls), each outlet is handled in sequence. That is, outlet 1 will be addressed first, then outlet 2, 3, etc. With each outlet, any applicable delay is handled in sequence. The delays build up cumulatively from a starting point. Assume outlets 1 through 4 each have a 5 second On Delay. If these outlets start out as off, and are all commanded at one time to turn on, then after 5 seconds outlet 1 is turned on, after 10 seconds outlet 2 is turned on, after 15 seconds outlet 3 is turned on, etc. as each outlet’s own 5-second delay is accounted for.

Outlet Control from the Web

Outlet Switching

The web Dashboard offers three ways to manipulate outlet state. The first is an All Outlets set of buttons to affect all outlets with a single command to turn them on, turn them off, or cycle them. When using these controls, each outlet is handled in sequence as described in “Switching Sets of Outlets” above.

Outlets : All

All Outlets Shown Below

On

Off

Cycle

Switch	ID	Outlet Label	Panel Name	Rating	Action
<div>On</div>	1	Utility Outlet	J1	90 to 125 VAC, 20 A max.	<div>None</div>
<div>On</div>	2	Control Panel & Logic	J2	90 to 125 VAC, 20 A max.	<div>None</div>
<div>On</div>	3	Hydraulics Controller	J3	90 to 125 VAC, 15 A max.	<div>None</div>
<div>On</div>	4	M1 Motor Controller	J4	90 to 125 VAC, 15 A max.	<div>None</div>
<div>On</div>	5	M2 Motor Controller	J5	90 to 125 VAC, 15 A max.	<div>None</div>
<div>On</div>	6	Battery Backup	J6	90 to 125 VAC, 15 A max.	<div>None</div>
<div>On</div>	7	DAQ Server	J7	48 VDC, 30 A max.	<div>None</div>
<div>On</div>	8	DB Server	J8	48 VDC, 30 A max.	<div>None</div>

Apply Actions

Apply

The second method for switching outlets is the Action column of the Dashboard Outlets panel. Each outlet has a popup menu to choose On, Off, or Cycle. Each outlet can be set to any of these options. When all choices have been made, pressing the Apply button at the top of the column will issue each command in sequence.

The third method for switching outlets is the Switch column buttons which provide simple and immediate manual On and Off switching control.

Outlet Delays

Adjusting outlet delays is done in the Outlets section of the web interface.

Outlet Control from the Command Line

Outlet Switching

For all outlet actions, the command line offers the `setOutlet` command. Using the available variants and attributes, the following commands are possible:

- `setOutlet 1 switch on` — affects only the identified outlet number
- `setOutlet 1 switch off`
- `setOutlet 1 switch cycle`
- `setOutlet switch on` — affects all outlets
- `setOutlet switch off`
- `setOutlet switch cycle`

Whenever a specific outlet is identified, the command is issued immediately. That is, `setOutlet 1 switch off` works just like a manual switch, and the on/off delays are ignored. However, when switching all outlets, the on/off/cycle delays are honored.

```
#> setOutlet 1 switch off

Setting switch of outlet 1 to: off... OK.

#> setOutlet switch on

Setting switch of outlet 1 to: on... OK.
Setting switch of outlet 2 to: on... OK.
Setting switch of outlet 3 to: on... OK.
Setting switch of outlet 4 to: on... OK.
Setting switch of outlet 5 to: on... OK.
Setting switch of outlet 6 to: on... OK.
Setting switch of outlet 7 to: on... OK.
Setting switch of outlet 8 to: on... OK.

#> setOutlet 3 on_delay 2

Setting on_delay of outlet 3 to: 2... OK.

#> setOutlet off_delay 0

Setting off_delay of outlet 1 to: 0... OK.
Setting off_delay of outlet 2 to: 0... OK.
Setting off_delay of outlet 3 to: 0... OK.
Setting off_delay of outlet 4 to: 0... OK.
Setting off_delay of outlet 5 to: 0... OK.
Setting off_delay of outlet 6 to: 0... OK.
Setting off_delay of outlet 7 to: 0... OK.
Setting off_delay of outlet 8 to: 0... OK.
```


Outlet Delays

Adjusting outlet delays is also done using the `setOutlet` command.

- `setOutlet 3 on_delay 2` — sets only the identified outlet value
- `setOutlet 3 off_delay 0`
- `setOutlet 3 cycle_delay 15`
- `setOutlet switch on_delay 2` — sets all outlets to the same value
- `setOutlet switch off_delay 0`
- `setOutlet switch cycle_delay 15`

Refer to “[setOutlet](#)” on page 58 for details.

Viewing Log Data

Log entries are viewable on the web interface using the main menu item Logs where two tabs are available: System Log and Startup Notices. Additionally, the CLI commands `getLog` and `getStartupLog` may be used.

Logged Events

The RCM Software maintains an event log intended for tracking recent event history. Logged events include:

- System startup
- Successful user login
- Failed user login
- Settings updated
- Internal errors
- Outlet switched on/off or cycled (logging of switch events are normally disabled for CLI and RESTful API triggered events, then can be enabled)

Logging of Outlet Switch Events

By default, logging of switch events is enabled only for events triggered by the web interface. Events triggered over a command line interface (Telnet or SSH), or through the RESTful API are not logged. Additionally, the state of the outlet change is not recorded as the Last Known state. Logging can be enabled for these interfaces using the options on the Power > Misc web interface tab. More details can be found in “[Outlet State Logging and Saving](#)” on page 53.

Log Format

The log file is an ASCII text file with tab delimited fields and “Windows-style” line endings (\r\n). Each log entry includes the following data:

- | | |
|----------------|---------------|
| • Severity | • Message |
| • DateTime | • PDU Model |
| • PDU IP | • PDU Version |
| • PDU Location | • PDU Extra |
| • Category | |

The Severity field describes an escalating scale of importance using these keywords:

Info : normal event activity information

Warn : something which is not a problem, but may be more important to know about than Info (e.g. a failed user login)

Error : errors are not expected. An error message is an indication of something not working correctly. Occasional errors may not be a sign of a larger problem, but multiple errors should be investigated.

Critical : a serious error has occurred. The system should be restarted as soon as possible.

The DateTime field will be in the format of YYYY-MM-DD HH:MM:SS using 24-hour time. This format is naturally sortable.

The PDU IP field is the IPv4 address of the PDU. This will be the same for all entries, but allows logs to be combined, and keeps logs distinct if the IP address changes.

The PDU Location field is the user-entered location field.

The Category field is a set of keywords useful for searching or filtering the log for specific types of events. Not all categories apply to all RCM Software versions or PDUs. The keywords, in no particular order, include:

Startup : used when recording that a user issued a startup command, and also the startup event itself.

Login : used when recording successful and failed login attempts.

Alarm : used when recording that a setpoint or other alarmable condition has been triggered.

Email : used when logging alert preparations, and other activities of the SMTP system.

Switch : used when recording changes to an outlet state (on, off, or cycled).

Settings : used when recording changes to any of the setting files.

Time : used when logging activities of the internal clock system and SNTP protocol.

Files : used when logging activities of the file system, which will almost always be for an error.

System : used when logging activities of the core operating system, which will almost always be for an error.

The Message field is the text of the event message.

The PDU Model field is the model number of the PDU.

The PDU Version field is the version number of the PDU software.

The PDU Extra field provides additional internal details for errors which will be useful to factory technician when troubleshooting.

System Log vs. Startup Log

The system log has a fixed number of allowed entries. Once the available log space becomes full, the oldest message is deleted to make room for the newest message.

How long a period of time the log covers depends on how many events have been recorded. A session which includes a number of settings changes, switch state changes, etc. will write more events to the log, so the total period of time covered could be shortened to hours or even minutes. Periods where the PDU is operating normally with little or no settings changes, and no alarm events could cover weeks or months.

With such a system, it's possible that any error messages logged during the PDU startup process could be pushed out of the log at some point. To prevent this, an error message which occurs during the startup process is also logged to a separate "startup log."

The startup log is retained until the system is restarted. The startup log is visible on the PDU web login page, the Logs section of the web interface, and

by using the `getStartupLog` command. The events in this log remain viewable at any time until the next startup, during which the startup log is replaced by that startup's events (if any).

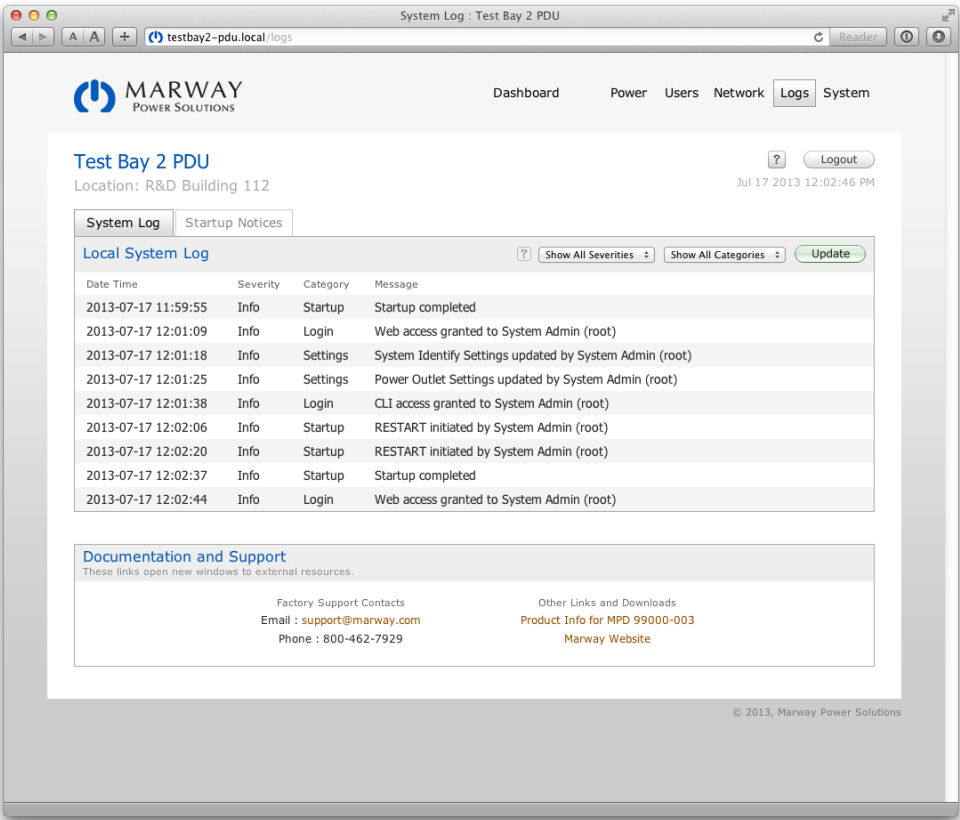
It is normal for the startup log to be empty.

Web Viewing of Log Data

The Logs section of the web interface include two tabs: System Log and Startup Notices. The System Log list is not sortable, but the log viewer allows filtering based on the Severity and Category fields making it easier to seek out specific events. Selecting an option from one or both of the filters will remove from the display (not from the log itself) everything except those items matching the selected options.

Command Line Viewing of Log Data

Use the command `getLog` to view the system log, and `getStartupLog` to view the startup notices. The `getLog` command accepts filter parameters to help seek out a specific type of log message. See “`getLog`” on page 68.



MPD Series RCM Software

User Management

User Management

An RCM Software user record includes several features:

- authentication (a login name and password),
- access methods (which network protocols the user is allowed to login to),
- a personal profile (name, company, email, and phone information),
- permissions (specific capabilities the user is allowed to use), and
- status (last login time and login count).

The RCM Software will store up to 20 locally-defined user records. One of those records is predefined as the root user (which cannot be deleted).

Root User

A user with a login name of “root” is predefined. This user cannot be deleted, and the login name cannot be changed. The permissions cannot be edited (all permissions are permanently allowed). However, the password and profile for the user may be edited. The password can be updated through the web interface or command line.

Authentication

Generally speaking, authentication deals with the identification of a user through some means of information that only the user should have. In the RCM Software, this means having both an account login name and a password.

Name and Password

Authentication requires user name and password credentials. The name is an arbitrary text value. The password requirements help to enforce moderately strong passwords, and with up to 32 characters, allows for very strong passwords.

Passwords in the Settings File

The settings file stores passwords salted and encrypted. It is not possible to simply edit a user's password in the settings file. To generate a salt and encryption string for the settings file, use the command line command `makeLoginPswd`. See “`makeLoginPswd`” on page 59 for details.

Authorization

Authorization deals with allowing the user to see, utilize, or edit specific data in or features of the software. In the RCM Software, authorization includes whether the user is enabled to use one or more protocols to log into the PDU, and a specific set of selected permissions once logged in.

Access Methods

The root user is inherently allowed to use all login protocols. All other users are allowed to web access only (they cannot use command line interfaces nor FTP).

Permissions

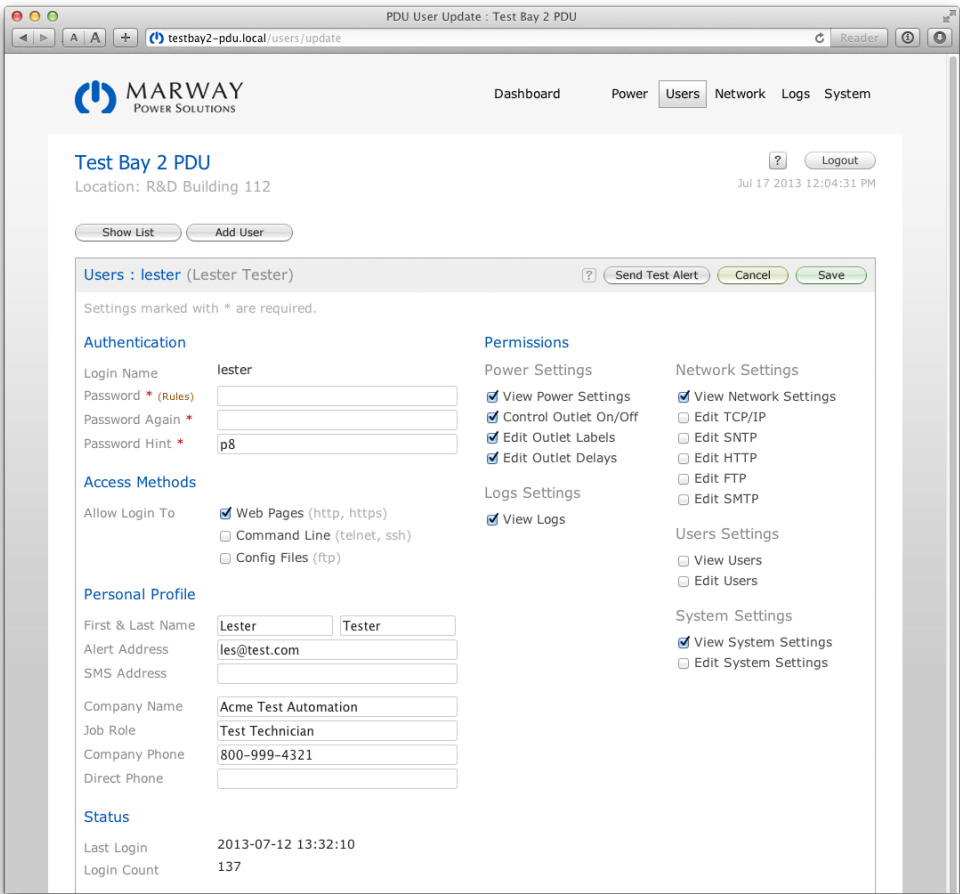
A number of specific permissions have been predefined in the software to allow an administrator to have granular control over what each user is allowed to see and do. Users do not have “roles” or belong to “groups.” These paradigms are useful for batch editing of users, but our experience in user management suggests there are always exceptions to these paradigms which results in the over-prescribing of permissions to many users. Given the relatively few users needing access to a PDU, the assignment of explicit permissions uniquely to each user allows for more control.

Auditing

Auditing deals with logging user activity so that an audit trail is available to identify exactly what a user has done in the system.

Activity Logging

Though not built with full-feature auditing capabilities, the RCM Software provides some user activity logging. The on-board log keeps track of several critical activities, and is able to stamp those activities with user identification. Software restarts, outlet switching, and settings file changes are all stamped with user login names which performed those actions. The log file size is limited, and is intended more for short-term troubleshooting than long-term audit trails.



Profile

The Personal Profile fields of the user record allow a user to be identified by name, the company represented, and with email and phone contact information.

Show ListAdd User

Users : List?

Edit	Login Name	User Name	Email	Access	Logins	Last Login	Del
	root	System Admin	sysmin@test.com	Web CLI FTP	104	2013-07-17 12:02:44	
	lester	Lester Tester	les@test.com	Web	137	2013-07-12 13:32:10	

Adding, Deleting, Editing Users

Users can be added, deleted, and edited using either the web interface or command line. Using the web interface, press the Add User button or select a user to edit or delete from the users list. The web interface for adding and editing users is shown on the previous page. Deleting a user via the web interface is a two-step process. First press the trash button for the user to be deleted. A non-editable form will display the details of that user, and provide a set of confirmation Delete and Cancel buttons.

Separate commands on the command line are used to add, delete, edit, and display user records. Refer to the addUser, deleteUser, setUser, and getUser commands in “User Commands” on page 59.

MPD Series RCM Software

Software Settings

Software Settings

Adjustable values for outlets, users, network protocols, etc. are called settings, and are accessible through the web UI, the command line, and also in files accessed through FTP.

Editing the files directly would not be an ideal way to make adjustments normally, but can be a convenient way to set up or update multiple PDU units quickly by preparing replacement files and uploading them to each PDU (followed by a restart to activate the new settings).

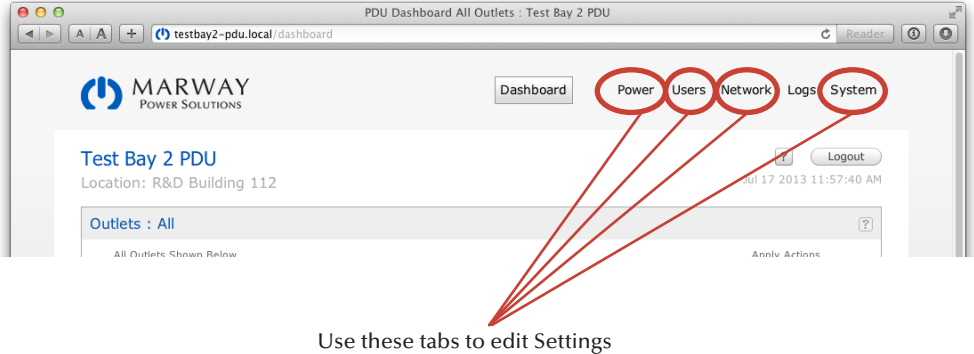
In this chapter, the purpose and acceptable values of each setting will be identified. Additionally, how to use the web UI, CLI, and files to edit them will also be covered.

Adjust Settings with the Web UI

Settings are adjusted in all sections of the web interface except the Dashboard and Logs. Changing settings is done through web forms. Values can be edited, and the form must be saved. If invalid values are entered, the web interface will highlight those entries and explain valid entry options. Additionally, most fields will display a range and/or default value tooltip by hovering a mouse pointer over the field and pausing for a short time.

Adjust Settings with the CLI

Settings are adjusted on the command line using various `set` commands specific to the setting being set (e.g. `setOutlet` or `setUser`). If an invalid value is submitted, the CLI will usually offer some text as to the valid entry



Outlet Settings : All

One or more fields below has an input error.

ID	Outlet Label	Panel Name	Switch Delays in Seconds On Off Cycle	Startup State
The field On Delay must be between 0 and 60 seconds.				
1	Utility Outlet	J1	99999 0.0 5.0	Last Known
2	Control Panel & Logic	J2	0.2 0.0 5.0	Last Known
3	Hydraulic Controller	J3	0.2 0.0 5.0	Last Known

Create New User

One or more fields below has an input error.

Settings marked with * are required.

Authentication

The field Account Name must be 3-32 characters of a-z, A-Z, 0-9, underscores, hyphens, periods, and @ symbol.

Login Name *

Password * (Rules)

Password Again *

Permissions

Outlet Settings

- ☐ View Outlet Settings
- ☐ Control Outlet On/Off
- ☐ Edit Outlet Labels
- ☐ Edit Outlet Delays

Network Settings

- ☐ View Network Settings
- ☐ Edit TCP/IP
- ☐ Edit SNMP
- ☐ Edit HTTP

requirements, and at least provide feedback that the value was not valid. Note that the CLI can be used to set multiple objects to the same value. For example:

```
setOutlet cycle_delay 5.0
```

will set the cycle delay for every outlet whereas

```
setOutlet 3 cycle_delay 5.0
```

will set the value for outlet 3 only. See “[Command Syntax](#)” on page 56 for other command line options.

Adjust Settings in the Files

Settings files may be copied from FTP, edited, then uploaded back to FTP. Many FTP client applications will open a text file directly and save it back to the same place. Settings files are simple ASCII text files saved with “Windows-style” line endings (that is, the `\r\n` sequence).

Use a text editor, not a word processor, to edit these files. On Windows, the simple Notepad utility will work (though a more sophisticated text editor will provide better editing tools). Do not use WordPad, or any of the “office” type applications like Microsoft Word, OpenOffice Writer, etc.

While this chapter discusses the individual settings within these files, refer to “[Settings Files Reference](#)” on page 81 for details about the formatting of each settings file.

The file at `/factory/factorySettings.txt` can be copied for archiving and reference, but it cannot be edited, deleted, or replaced. This file is what defines the configuration of the PDU product, and tells the software what the PDU features and specifications are.

```
#> setHttp port 999999
```

```
Error: port value must be between 1 and 65535.
```

RESTART THE PDU AFTER EDITING SETTINGS FILES

It is possible to edit settings files in place (using an FTP client application that opens text files), or by copying, editing, and uploading the settings file. It is necessary to restart the PDU after the new files have been saved/uploaded. First, this is the only way to have the new settings take effect. Second, if another user makes any changes to settings via web or CLI, or even changes the state of an outlet, the uploaded settings files will be overwritten with their previous values saved in the PDU’s memory. Always use a software restart command by web or CLI after uploading settings files.

System Settings

System settings are for items universal to the whole PDU, and which otherwise doesn't have another appropriate home. Refer to the screen capture in “[System Settings](#)” on page 19.

Setting	Description	CLI Command / Attribute	File / Setting	Type	Validation
Label	A label to uniquely identify the PDU. It is displayed on all web pages just below the logo, and in response to the <code>getSystem</code> CLI command.	<code>setSystem label</code>	<code>systemSettings.txt</code> <code>label =</code>	Text	All English letters, numerals, and punctuation except double quotes.
Location	A label to uniquely identify the location of the PDU. It is displayed on all web pages just below the system label at the top of the page, and in response to the <code>getSystem</code> CLI command.	<code>setSystem location</code>	<code>systemSettings.txt</code> <code>location =</code>	Text	All English letters, numerals, and punctuation except double quotes.
Time	Date and time values for when the PDU is not connected to a networked time server (SNTP).	<code>setSystem time</code>	(none)	Time	YYYY-MM-YY HH:MM:SS (24 hr)

Network Settings

Network settings define the network protocols such as IP addresses, ports, and other values. This guide will not attempt to explain the protocols themselves. Assistance from an experienced network administrator may be necessary to help with unfamiliar protocols or settings. Refer to the screen capture in “[Network Settings](#)” on page 18.

Setting	Description	CLI Command / Attribute	File / Setting	Type	Validation
IPv4 DHCP Enabled	Determines whether the TCP/IPv4 address will be supplied by a DHCP server (“on”) or entered manually (“off”).	setTcp ipv4_dhcp	networkSettings.txt ipv4dhcp =	Text	“on” or “off”
IPv4 Address	The TCP/IPv4 address. Leave empty for DHCP.	setTcp ipv4_address	networkSettings.txt ipv4address =	Text	0.0.0.0 format
IPv4 Subnet Mask	The TCP/IPv4 subnet mask. Usually something like 255.255.255.nnn (where nnn depends on the IP address range of the subnet). On large networks, the 3rd element may not be 255.	setTcp ipv4_subnet	networkSettings.txt ipv4subnet =	Text	0.0.0.0 format
IPv4 Gateway	The TCP/IPv4 subnet gateway address.	setTcp ipv4_gateway	networkSettings.txt ipv4gateway =	Text	0.0.0.0 format
IPv6 DHCP Enabled	Determines whether the TCP/IPv6 address will be supplied by a DHCP server (“on”) or entered manually (“off”).	setTcp ipv6_dhcp	networkSettings.txt ipv6dhcp =	Text	“on” or “off”
IPv6 Address	The TCP/IPv6 address. Leave empty for DHCP.	setTcp ipv6_address	networkSettings.txt ipv6address =	Text	IPv6 format

Setting	Description	CLI Command / Attribute	File / Setting	Type	Validation
IPv6 Prefix Length	The TCP/IPv6 prefix length. Almost always /64.	setTcp ipv6_prefix_length	networkSettings.txt ipv6PrefixLength =	Integer	0-128 (default = 64)
HTTP Port	The TCP port for the service.	setHttp port	networkSettings.txt httpPort =	Integer	1-65535 (default = 80)
HTTPS Enabled	Indicates whether HTTPS should be used instead of HTTP (only one runs at a time).	setHttps enabled	networkSettings.txt httpsEnabled =	Text	“on” or “off”
HTTPS Port	The TCP port for the service.	setHttps port	networkSettings.txt httpsPort =	Integer	1-65535 (default = 443)
HTTP Session Time	The maximum time in minutes a web session can be inactive before the login session is flagged as invalid.	setHttp session_minutes	networkSettings.txt httpSessionMins =	Integer	15-1440 (default 30) 0 = disabled (no expiration)
SNTP Server 1 IP	The TCP/IPv4 address of a network time server.	setSntp server1	networkSettings.txt sntpServerIp1 =	Text	0.0.0.0 format
SNTP Server 2 IP	The TCP/IPv4 address of a network time server.	setSntp server2	networkSettings.txt sntpServerIp2 =	Text	0.0.0.0 format
SNTP Sync Frequency	Defines how often in hours the PDU should contact the SNTP server for an updated time.	setSntp sync_interval	networkSettings.txt sntpSyncInterval =	Integer	1-168 hours (default = 12) 0 = disabled
Timezone STD GMT Offset	Defines the local standard time offset in hours from Greenwich Mean Time (GMT).	setSntp std_offset	networkSettings.txt sntpStdOffset =	Text	GMT-H.H or GMT+H.H Where H.H is the offset in hours such as GMT-8.0 or GMT+9.5
Timezone DST GMT Offset	Defines the local daylight savings time offset in hours from Greenwich Mean Time (GMT).	setSntp dst_offset	networkSettings.txt sntpDstOffset =	Text	GMT-H.H or GMT+H.H Where H.H is the offset in hours such as GMT-8.0 or GMT+9.5

Setting	Description	CLI Command / Attribute	File / Setting	Type	Validation
Timezone DST Starts	Defines when daylight savings time starts.	setSntp dst_starts	networkSettings.txt sntpDstStarts =	Text	Enter format like 11.1.0/02:00:00 meaning Month.Week.Day/Hr:Min:Sec where : Month = 1 to 12 Week of the month = 1 to 5 Day = 0 (Sun) to 6 (Sat) Hour = 0 to 24 Minute = 0 to 59 Second = 0 to 59 The example above means Sun first week of Nov at 2AM.
Timezone DST Ends	Defines when daylight savings time ends.	setSntp dst_ends	networkSettings.txt sntpDstEnds =	Text	Enter format like 11.1.0/02:00:00 (see row above for details)
FTP Enabled	Determines whether FTP is enabled or not.	setFtp enabled	networkSettings.txt ftpEnabled =	Text	“true” or “false”
FTP Port	The TCP port for the service.	setFtp port	networkSettings.txt ftpPort =	Integer	1–65535 (default = 21)
FTP Auto Off	When non-zero, this sets how many minutes after FTP is enabled that it will remain active before being automatically disabled. When zero, FTP remains active indefinitely.	setFtp auto_off	networkSettings.txt ftpAutoOff =	Integer	5–60 minutes (default 60) 0 = no expiration
SMTP Server	The TCP/IPV4 address of an SMTP mail server.	setSntp server	networkSettings.txt smtpServer =	Text	0.0.0.0 format

Setting	Description	CLI Command / Attribute	File / Setting	Type	Validation
SMTP Port	The TCP port for the service.	setSmtplib port	networkSettings.txt smtpPort =	Integer	1-65535 (default = 25)
SMTP Auth Method	The authentication login method.	setSmtplib auth_method	networkSettings.txt smtpAuthMethod =	Text	One of the following: none, any, plain, login, cram, digest
SMTP Mail Account	The user name of the SMTP account. Depending on the server this may be a simple name, or a full email address.	setSmtplib account	networkSettings.txt smtpAccount =	Text	Any characters deemed legal by the mail server for account names.
SMTP Mail Password	The password for the SMTP Mail Account.	setSmtplib password	networkSettings.txt smtpPassword =	Text	Any characters deemed legal or required by the mail server for account passwords.

User Settings

User settings define the authentication, profile, and permissions of each PDU user. Most of these settings are adjustable through the web or command line interfaces. Some status data such as the number of logins, most recent login date, and others are stored in the settings file. Generally, these would not be manually adjusted, and there is not a way to use the web or CLI to adjust these values. However, their presence in the settings file means technically they could be adjusted. Refer also to “[User Management](#)” on page 37.

Setting	Description	CLI Command / Attribute	File / Setting	Type	Validation
Login Name	The account name a user will provide to login into the PDU.	getUser auth addUser	userSettings.txt loginName =	Text	3-32 characters of a-z, A-Z, spaces, or underscores
Login Password	The password for the account name a user provides to login into the PDU. The userSettings.txt file requires an encrypted version of the password and a salt. Use the command line utility makeLoginPswd to generate the encrypted loginPswd value and the pswdSalt value to store in the file.	addUser setUser password	userSettings.txt loginPswd =	Text	8-32 characters, at least one from A-Z, at least one from a-z, at least one from 0-9. A minimum of four letters (i.e. it can't be mostly numerals). Should include one from !@#\$\$%^&* but no other punctuation marks.
Password Salt	This value is generated internally when passwords are added or updated via the web UI or command line and saved to the settings file. However, to manually add to or modify the userSettings.txt file use the command line utility makeLoginPswd to generate the encrypted loginPswd value and the pswdSalt values.	(no get command) (no set command) Use makeLoginPswd to create values for the settings file.	userSettings.txt pswdSalt =	Text	12 random characters from a-z, A-z, 0-9

Setting	Description	CLI Command / Attribute	File / Setting	Type	Validation
Password Is Temporary	This value is generated internally to indicate whether the password needs to be replaced by the user on the next login. [This is not yet in use.]	getUser auth (no set command)	userSettings.txt pswdIsTemp =	Text	“true” or “false”
Allowed Client IPs	A definition of which IPv4 addresses the PDU will allow client logins from. [This is not yet in use.]	(not yet available)	userSettings.txt allowedClientIPs =	Text	A list of IPv4 addresses and/or slash notation addresses.
Login Count	This value is maintained internally to indicate how many times the user has logged into the PDU.	getUser auth (no set command)	userSettings.txt loginCount =	Integer	0–65535
Most Recent Login	This value is maintained internally to indicate the last date and time the user logged into the PDU.	getUser auth (no set command)	userSettings.txt mostRecentLogin =	ISO Datetime	YYYY-MM-YY HH:MM:SS (24 hr)
Failed Attempts	This value is maintained internally to indicate the how many times the users account has attempted a login with a wrong password. After 3 failed attempts, the account will be blocked for three minutes.	getUser auth (no set command)	userSettings.txt failedAttempts =	Integer	0–65535
Lockout Time	This value is maintained internally to indicate when a user’s account was set to be blocked for excessive failed login attempts. The time is ignored and reset after it the interval exceeds three minutes.	getUser auth (no set command)	userSettings.txt lockoutTime =	ISO Datetime	YYYY-MM-YY HH:MM:SS (24 hr)
First Name	The user’s first name (given name).	getUser profile setUser first_name	userSettings.txt firstName =	Text	0–24 characters of a-z, A-Z, spaces, periods, hyphens, and apostrophes.
Last Name	The user’s last name (surname).	getUser profile setUser last_name	userSettings.txt lastName =	Text	0–24 characters of a-z, A-Z, spaces, periods, hyphens, and apostrophes.
Company Name	The company name the user represents.	getUser profile setUser company_name	userSettings.txt companyName =	Text	All English letters, numerals, and punctuation except double quotes.

Setting	Description	CLI Command / Attribute	File / Setting	Type	Validation
Job Role	A descriptive title or role the user performs for the PDU or for the company he represents.	getUser profile setUser job_role	userSettings.txt jobRole =	Text	All English letters, numerals, and punctuation except double quotes.
Company Phone	A general phone number for the company the user represents.	getUser profile setUser company_phone	userSettings.txt companyPhone =	Text	Up to 29 characters of 0-9, spaces, periods, and +()-/x (write extensions like x000).
Direct Phone	A direct phone number to reach the user.	getUser profile setUser direct_phone	userSettings.txt directPhone =	Text	Up to 29 characters of 0-9, spaces, periods, and +()-/x (write extensions like x000).
Email	An email address to reach the user. This same address is used to send the user alerts (subject to the Alerts Settings).	getUser profile setUser email	userSettings.txt email =	Text	A valid email address.
SMS	A mobile phone number in an email addressable form to send the user alert messages by SMS. This is generally in the form of mobile_number@carrier.tld such as 2125551212@txt.att.net	getUser profile setUser sms	userSettings.txt sms =	Text	A valid email address.
Login Via Web	Determines whether the user is allowed to access the PDU via the web interface.	getUser auth permissions setUser login_via_web	userSettings.txt loginViaWeb =	Text	“true” or “false”
Login Via CLI	Determines whether the user is allowed to access the PDU via Telnet and SSH. Note: as of this version, only the root user can access Telnet and SSH. Changing this setting for other users is ignored.	getUser auth permissions setUser login_via_cli	userSettings.txt loginViaCLI =	Text	“true” or “false”
Login Via FTP	Determines whether the user is allowed to access the PDU via FTP. Note: as of this version, only the root user can access FTP. Changing this setting for other users is ignored.	getUser auth permissions setUser login_via_ftp	userSettings.txt loginViaFTP =	Text	“true” or “false”

Setting	Description	CLI Command / Attribute	File / Setting	Type	Validation
View Power Settings	Determines whether the user is allowed to view power settings. If any of the power setting edit permissions is true, then this will be forced true as well.	getUser auth permissions setUser view_power_settings	userSettings.txt viewPowerSettings =	Text	“true” or “false”
Control Outlets	Determines whether the user is allowed to switch outlets on/off.	getUser auth permissions setUser control_outlets	userSettings.txt controlOutlets =	Text	“true” or “false”
Edit Outlet Labels	Determines whether the user is allowed to edit outlet labels.	getUser auth permissions setUser edit_outlet_labels	userSettings.txt editOutletLabels =	Text	“true” or “false”
Edit Outlet Delays	Determines whether the user is allowed to edit outlet delays.	getUser auth permissions setUser edit_outlet_delays	userSettings.txt editOutletDelays =	Text	“true” or “false”
View Logs	Determines whether the user is allowed to view log entries.	getUser auth permissions setUser view_logs	userSettings.txt viewLogs =	Text	“true” or “false”
View Network Settings	Determines whether the user is allowed to view network protocol settings.	getUser auth permissions setUser view_network_settings	userSettings.txt viewNetworkSettings =	Text	“true” or “false”
Edit TCP/IP	Determines whether the user is allowed to edit TCP/IP protocol settings.	getUser auth permissions setUser edit_tcpip	userSettings.txt editTCPIP =	Text	“true” or “false”
Edit SNTP	Determines whether the user is allowed to edit SNTP protocol settings.	getUser auth permissions setUser edit_sntp	userSettings.txt editSNTP =	Text	“true” or “false”
Edit HTTP	Determines whether the user is allowed to edit HTTP(S) protocol settings.	getUser auth permissions setUser edit_http	userSettings.txt editHTTP =	Text	“true” or “false”
Edit FTP	Determines whether the user is allowed to edit FTP protocol settings.	getUser auth permissions setUser edit_ftp	userSettings.txt editFTP =	Text	“true” or “false”

Setting	Description	CLI Command / Attribute	File / Setting	Type	Validation
View Users	Determines whether the user is allowed to view user settings.	getUser auth permissions setUser view_users	userSettings.txt viewUsers =	Text	“true” or “false”
Edit Users	Determines whether the user is allowed to edit user settings.	getUser auth permissions setUser edit_users	userSettings.txt editUsers =	Text	“true” or “false”
View System Settings	Determines whether the user is allowed to view system settings.	getUser auth permissions setUser view_system_settings	userSettings.txt viewSystemSettings =	Text	“true” or “false”
Edit System Settings	Determines whether the user is allowed to edit system settings.	getUser auth permissions setUser edit_system_settings	userSettings.txt viewSystemSettings =	Text	“true” or “false”

Outlet Settings

Outlet settings include labels and some items specific to switching behavior. Refer also to the screen captures in “Power Settings” on page 16.

Labels

The RCM Software defaults to labeling each power device by its device name along with a sequential numerical identification. This results in default labels such as Outlet 1, Outlet 2, etc. To allow labels with more meaningful descriptions, the labels for Outlets can be changed.

Valid characters for labels include all English letters, numerals, and most standard keyboard punctuation marks. Quotes are not allowed in a label, nor are extended range, escaped, or Unicode characters.

Outlet Switch Delays

Outlet switches feature three action delay settings. Details about these features can be found in “Outlet Delays” on page 32. In summary:

The On Delay causes a delay between the moment when an On action is initiated and when the outlet actually switches on.

The Off Delay causes a delay between the moment when an Off action is initiated and when the outlet actually switches off.

The Cycle Delay causes a delay in the cycle process after the outlet is switched off before it is switched on.

The values for these delays are in seconds with a valid range of 0 to 60. A decimal value in tenths such as 0.2 is acceptable.

Outlet Startup State

If the PDU loses facility power, or is rebooted with the reset switch in the on-board keypad, the outlets will be switched off. Software restart commands do not affect outlet state. If the outlets are switched off, the PDU needs to know what to do with the outlets after startup. This is the job of the Startup State setting. It has three values: on, off, and last known. When “on” the outlet will be switched on after startup, and “off” will leave it off. The “last known” option will restore the same state the outlet was in before power was removed (ONLY if outlet switch event logging is enabled for CLI and RESTful interfaces). See the section below for details.

Outlet State Logging and Saving

By default, the factory settings disable logging of switch events triggered over a command line interface (Telnet or SSH), or through the RESTful API. Additionally, the state of the outlet change is not recorded as the “last known” state. These events can be enabled for the CLI and REST interfaces using the options on the Power > Misc web interface tab (there are no CLI commands as of this version of the software).

Why are these events disabled? To improve performance and log value when CLI and REST are used for automated control of outlet switch state. The RESTful API by design is intended for automated control. It is very common to see the Telnet interface used for automated control as well. In such cases, there’s not much value in having the outlet switch events logged. Furthermore, when outlets are switched frequently in automated test applications, there’s little value in updating the outlet’s “last known” state up on power restoration since most cases will start with all outlets off. By disabling the tracking of these events, there’s less load on the switching system’s processor and EEPROM.

Setting	Description	CLI Command / Attribute	File / Setting	Type	Validation
Outlet Label	A label to identify the outlet (defaults to “Outlet 1”).	getOutlet setOutlet label	powerSettings.txt (outlet~ID section) label =	Text	All English letters, numerals, and punctuation except double quotes.
Outlet On Delay	Defines the length of a delay between the moment when an On action is initiated and when the outlet actually switches on.	getOutlet setOutlet on_delay	powerSettings.txt (outlet~ID section) onDelay =	Decimal	0–60.0 seconds
Outlet Off Delay	Defines the length of a delay between the moment when an Off action is initiated and when the outlet actually switches off.	getOutlet setOutlet off_delay	powerSettings.txt (outlet~ID section) offDelay =	Decimal	0–60.0 seconds
Outlet Cycle Delay	Defines the length of a delay in the cycle process after the outlet is switched off and before it is switched on.	getOutlet setOutlet cycle_delay	powerSettings.txt (outlet~ID section) cycleDelay =	Decimal	0–60.0 seconds
Outlet Startup State	After the PDU is rebooted (or power is restored after a facility power loss), this setting determines whether the outlet should be switched on, off, or set to the last known state.	getOutlet settings setOutlet startup_state	powerSettings.txt (outlet~ID section) startupState =	Text	One of the keywords: “on,” “off,” or “last known”
Outlet Switch State	Indicates the current state of the outlet (or, in the settings file the last known state).	getOutlet settings setOutlet switch	powerSettings.txt (outlet~ID section) switchState =	Text	One of the keywords: “on” or “off”
Outlet Switch Event Logging over CLI	Determines whether outlet switch events triggered over CLI will be recorded.	(none)	powerSettings.txt logCLISwitching =	Text	“true” or “false”
Outlet Switch Event Logging over REST	Determines whether outlet switch events triggered over REST will be recorded.	(none)	powerSettings.txt logRESTfulSwitching =	Text	“true” or “false”

MPD Series RCM Software

Command Line Reference

Command Line Reference

This chapter includes an overview of the command line syntax, and the entire contents of the built-in help for every command.

Built-in Help

At any point after login, typing help or ? at the prompt will display the top-level help which is a list of commands.

Each command can also be followed by a ? such as `getSystem?` (with or without a space) to display detailed usage help for that command. The command help will identify the syntax of the command, various attribute options, and some examples.

Command Syntax

The majority of commands follow a common set or get pattern. The following are typical get commands:

- `getOutlet 8 switch`
- `getUser lester email`
- `getSystem location`

```
telnet 192.168.1.10

-----
Marway PDU Command Line
-----

Login: root
Password: *****

#> ?

-----
Workspace  Get Commands  Set Commands  Misc Commands
-----
POWER*     getOutlet  setOutlet    getOutlets
           getCircuit setCircuit    getCircuits
           getLine   setLine      getLines
           getInput setInput     getInputs
           getSource setSource    getSource
ALARMS*    getAlarm   setAlarm     getAlarms
ALERTS*    getAlert   setAlert     getAlerts
USERS      getUser    setUser      getUsers
           getProfile setProfile   getMy, setMy
           addUser, deleteUser, makeLoginPswd

NETWORK    getNetwork
           getTcp    setTcp
           getHttp  setHttp
           getHttps setHttps
           getSntp  setSntp
           getFtp   setFtp
           getSmt  setSmt
LOG         getLog
           getStartupLog

SYSTEM     getSystem  setSystem    ?, help, quit
-----

Type help before a command, or type ? after a command for more details.
* Depending on the PDU configuration, some commands may not be supported.
-----
```


These are typical set commands:

- `setOutlet 1 switch on`
- `setUser lester email "les@example.com"`
- `setSystem location "Aisle 6 Rack 4"`

You'll notice the pattern of get and set are identical except for the last parameter of the set command which passes the new value. Otherwise, both use the same syntax of:

```
commandName instanceID attribute [setValue]
```

Where *commandName* is something like `getOutlet`, `getUser`, `setSystem`, etc. The *instanceID* is going to be the numerical ID of a power device such as an outlet, or the login name of a user. A few commands do not need an instance ID such as `getSystem`/`setSystem` where there's always one implied instance. The *attribute* is the specific detail being requested or set. Finally, if the command is a set, the fourth parameter is the new value.

Command Aliases

Many command attributes have aliases (synonyms). These aliases are revealed in the command help. For example, the `getSystem` command has an attribute `model_number`. That attribute name may be substituted with `model_no` or even `model`. So, `getSystem model` is the same command as `getSystem model_number`. The idea behind aliases is to enable commands to be flexible so people can use words and phrases they might naturally think of (without going overboard and having aliases for every word).

Command Variants

Most commands have variants, which are different ways of using the command to get different results. This primarily applies to get/set commands to address whether the command applies to a single instance or multiple instances of the command target.

A get command used like these examples:

```
getOutlet 4 switch
getUser lester email
```

will return one attribute for one specific instance.

A get command used like these examples:

```
getOutlet 4
getUser lester
```

will return all attributes for one specific instance.

A get command is used like these examples:

```
getOutlet
getUser
```

will return all attributes for all instances.

Command variants are explained with each command's built-in help by identifying optional parameters in square brackets like `[name]` in the example help display below:

```
#> getUser?
-----
Syntax:  getUser [name] [attribute]
```

Outlet Commands

getOutlet

Syntax: getOutlet [id] [attribute]

Where:

[]

:

indicates an optional parameter

id

:

is the number of the outlet (e.g. 1 or 3, etc.)

or leave it empty to refer to all outlets

attribute

:

is one of the following (or its alias):

label

--

rated_volts

(rv, ratedvolts)

rated_amps

(ra, ratedamps, max_amps, maxamps)

switch

(relay)

startup_state

(startup)

on_delay

--

off_delay

--

cycle_delay

--

power_data

(power, pwr)

Examples:

getOutlet 6 switch

:

gets one attribute for one outlet

getOutlet 6

:

gets all attributes for one outlet

getOutlet switch

:

gets one attribute for all outlets

getOutlet

:

gets all attributes for all outlets

setOutlet

Syntax: setOutlet [id] attribute value

Where:

[]

:

indicates an optional parameter

id

:

is the number of the outlet (e.g. 1 or 3, etc.)

or leave it empty to refer to all outlets

attribute

:

is one of the following (or its alias):

label

--

switch

(relay)

startup_state

(startup)

on_delay

--

off_delay

--

cycle_delay

--

value for label

:

is a quoted string (60 chars max)

value for switch

:

is one of: on, off, reboot|cycle|bounce

value for startup_state

:

is one of: on, off, last_known|last

value for delays

:

is a decimal of seconds, or: default

Examples:

setOutlet 6 switch reboot

:

sets one outlet

setOutlet cycle_delay 2.5

:

sets all outlets to 2.5 secs

setOutlet 3 on_delay default

:

resets to the default value

User Commands

getProfile (a.k.a getMy)

Syntax: getProfile attribute
or getMy attribute

Where attribute is one of the following:

status

- record management status details

auth

- user authentication details

profile

- user profile details

permissions

- user permissions details

Examples:

getProfile status

getProfile auth

getProfile profile

getProfile permissions

getMy permissions

makeLoginPswd

Encrypts a provided string (presumed to be a password) using RIPEMD-160. Returns a salt and encryption string suitable for adding to the userSettings.txt file stored on the file system.

```
#> makeLoginPswd "abc123"

salt = 8BnM0Qd6hCTJ
pswd = 6f23b84a083fbbbf08ae1d7d4c7c5e4d9590543d
```

setProfile (a.k.a setMy)

Syntax: setProfile attribute "value"
setMy attribute "value"

Where attribute is one of the following:

password

- following these rules:

- minimum of 8 characters,
- maximum of 32 characters,
- must include at least 4 letters (case sensitive),
- must include at least 1 digit (0-9),
- should include at least one of these !@#%^&* symbols for higher security, but not required

first_name

- the user's first name

last_name

- the user's last name

company_name

- the user's companyname

job_role

- the user's job role or title

company_phone

- 888-555-1818 x12345

mobile_phone

- 212-555-1010

email

- the user's email address

Examples:

setProfile password "Qwer!234"

setProfile webadmin enabled true

setProfile company_name "Silicon Farmers"

setProfile job_role "Server Support Contractor"

setMy job_role "Server Support Contractor"

getUser

Syntax: `getUser [name] [attribute]`

Where:
[] : indicates an optional parameter
name : is the login name of the user (e.g. john_doe)
or leave it empty to refer to all users

attribute
: is one of the following attribute groups:
status = record management status details
auth = user authentication details
profile = user profile details
permissions = user permissions details

: or one of the following profile attributes:
enabled, password_hint (hint),
first_name, last_name,
company_name, job_role,
company_phone, direct_phone, email

: or one of the following permission attributes
login_via_web (enable_web)
login_via_cli (enable_cli)
login_via_ftp (enable_ftp)
view_power
edit_outlet_labels
control_outlets, edit_outlet_delays
view_network
edit_tcpip
edit_http, edit_ftp
edit_snmp, edit_smtp
view_users
edit_users
view_logs
view_system
edit_system

Examples:
getUser john_doe edit_http : gets one attribute for one user
getUser john_doe : gets all attributes for one user
getUser edit_http : gets one attribute for all users
getUser : gets all attributes for all users

setUser

Syntax: `setUser [name] attribute "value"`

Where:
[] : indicates an optional parameter
name : is the login name of the user (e.g. john_doe)
or leave it empty to refer to all users

attribute
: is one of the following profile attributes:
enabled - enter true/false or yes/no
password - look at setMy? for details
hint - text to help remember the password
first_name - the user's first name
last_name - the user's last name
company_name - the user's companyname
job_role - the user's job role or title
company_phone - 888-555-1818 x12345
direct_phone - 212-555-1010
email - the user's email address

: or one of the following permission attributes
with a value of true/false or yes/no:

login_via_web (enable_web)
login_via_cli (enable_cli)
login_via_ftp (enable_ftp)
view_power
edit_outlet_labels
control_outlets, edit_outlet_delays
view_network
edit_tcpip
edit_http, edit_ftp
edit_snmp, edit_smtp
view_users
edit_users
view_logs
view_system
edit_system

Examples:
setUser john_doe password "Qwer!234"
setUser webadmin login_via_web true
setUser john_doe company_name "Silicon Farmers"
setUser dbAdmin job_role "Server Support Contractor"
setUser webadmin edit_system true

addUser

```
-----
Syntax:  addUser "loginName" password "loginPswd"

Where:   loginName    : the account name, following these rules:
          - minimum of 3 characters,
          - maximum of 32 characters,
          - letters (case sensitive), numerals, underscores,
            hyphens, periods, and @ symbols.
          - leading/trailing spaces will be trimmed,
          - the name cannot already be used.
          loginPswd    : the account password, following these rules:
          - minimum of 8 characters,
          - maximum of 32 characters,
          - must include at least 4 letters (case sensitive),
          - must include at least 1 digit (0-9),
          - should have one or more symbols from !@#$$^&*
            for higher security, but not required

Examples: addUser "JohnDoe" password "Qwer!234"
          addUser "mail_admin" password "wu7jsKhgr3poi"
          addUser "admin@example.com" password "wu7jsKhgr3poi"
-----
```

deleteUser

```
-----
Syntax:  deleteUser "name" password "pswd"

Where:   name         : the account name for the user
          pswd         : the account password for login

Examples: deleteUser "John Doe" password "abc#4mPwr"
-----
```

Network Commands

getTcp

Syntax: `getTcp [attribute]`

Where:
[] : indicates an optional parameter
attribute : is one of the following (or its alias):

<code>ipv4_dhcp</code>	(<code>dhcipv4</code> , <code>dhcp</code>)
<code>ipv4_address</code>	(<code>ipv4</code> , <code>ip</code>)
<code>ipv4_subnet</code>	(<code>subnet</code> , <code>mask</code>)
<code>ipv4_gateway</code>	(<code>gateway</code>)
<code>ipv6_dhcp</code>	(<code>dhcipv6</code>)
<code>ipv6_address</code>	(<code>ipv6</code>)
<code>ipv6_prefix</code>	(<code>prefixlen</code> , <code>prefix</code>)

Examples:
`getTcp` : displays all attributes
`getTcp subnet` : displays a single attribute's value

setTcp

Syntax: `setTcp attribute value`

Where:
[] : indicates an optional parameter
attribute : is one of the following (or its alias):

<code>ipv4_dhcp</code>	(<code>dhcipv4</code> , <code>dhcp</code>)
<code>ipv4_address</code>	(<code>ipv4</code> , <code>ip</code>)
<code>ipv4_subnet</code>	(<code>subnet</code> , <code>mask</code>)
<code>ipv4_gateway</code>	(<code>gateway</code>)
<code>ipv6_dhcp</code>	(<code>dhcipv6</code>)
<code>ipv6_address</code>	(<code>ipv6</code>)
<code>ipv6_prefix</code>	(<code>prefixlen</code> , <code>prefix</code>)

<code>value for ipv4_dhcp</code>	: is on off
<code>value for ipv4_address</code>	: is is formatted like 0.0.0.0
<code>value for ipv4_subnet</code>	: is is formatted like 0.0.0.0
<code>value for ipv4_gateway</code>	: is is formatted like 0.0.0.0
<code>value for ipv6_dhcp</code>	: is on off
<code>value for ipv6_address</code>	: defaults to :: for ::0.0.0.0 compatibility
<code>value for ipv6_prefix</code>	: almost always an integer of 64

Examples:
`setTcp ip 192.168.0.15` : sets the ipv4 address
`setTcp subnet 255.255.255.0` : sets the ipv4 subnet mask
`setTcp dhcpv6 on` : enables DHCP for ipv6

getHttp

Syntax: `getHttp [attribute]`

Where:

`[]` : indicates an optional parameter
`attribute` : is one of the following (or its alias):

`port` --
`session_minutes` (session, session_mins)

Examples:

`getHttp` : displays all attributes
`getHttp port` : displays a single attribute's value

setHttp

Syntax: `setHttp attribute value`

Where:

`[]` : indicates an optional parameter
`attribute` : is one of the following (or its alias):

`port` --
`session_minutes` (session, session_mins)

`value for port` : is an integer (default is 80)
`value for session_minutes` : is an integer of minutes. Applies to both http and https user sessions. (default is 30, min is 15, max is 1440)

Examples:

`setHttp port 8080` : binds HTTP to port 8080
`setHttp session 45` : session invalid after 45 mins of inactivity

getHttps

```
-----
Syntax:  getHttps [attribute]

Where:
  []      : indicates an optional parameter
  attribute : is one of the following (or its alias):

    enabled      --
    port         --
    session_minutes (session, session_mins)

Examples:
  getHttps      : displays all attributes
  getHttps enabled : displays a single attribute's value
-----
```

setHttps

```
-----
Syntax:  setHttps attribute value

Where:
  []      : indicates an optional parameter
  attribute : is one of the following (or its alias):

    enabled      --
    port         --
    session_minutes (session, session_mins)

value for enabled      : is true|false or on|off
value for port         : is an integer (default is 443)
value for session_minutes : is an integer of minutes. Applies to both
                           http and https user sessions.
                           (default is 30, min is 15, max is 1440)

Examples:
  setHttps port 8443      : binds HTTP to port 8443
  setHttps enabled on     : enables https, which disables http
  setHttps enabled off    : disables https, which enables http
  setHttps session 45     : session invalid after 45 mins of inactivity
-----
```


getSntp

Syntax: getSntp [attribute]

Where:

- [] : indicates an optional parameter
- attribute : is one of the following (or its alias):

- server1 --
- server2 --
- sync_interval (sync)
- std_gmt_offset (std_offset, std_gmt)
- dst_gmt_offset (dst_offset, dst_gmt)
- dst_start --
- dst_end --

Examples:

- getSntp : displays all attributes
- getSntp server1 : displays a single attribute's value

setSntp

Syntax: setSntp attribute value

Where:

- [] : indicates an optional parameter
- attribute : is one of the following (or its alias):

- server1 --
- server2 --
- sync_interval (sync)
- std_gmt_offset (std_offset, std_gmt)
- dst_gmt_offset (dst_offset, dst_gmt)
- dst_start --
- dst_end --

- value for servers : is an IP address like 0.0.0.0
- value for sync : is between 0 and 168 hours (default is 12)
- value for gmt offset : is from -12 to +14
- value for dst_start/end : is in format of 3.2.0/02:00:00
which means March, 2nd week, Sun, 2am

Examples:

- setSntp sync_interval 12 : updates local clock time every 12 hours
- setSntp dst_zone GMT-7 : sets DST to 7 hrs before GMT

getFtp

```
-----
Syntax:  getFtp [attribute]

Where:
  []      : indicates an optional parameter
  attribute : is one of the following (or its alias):

    enabled  --
    port     --
    auto_off --

Examples:
  getFtp      : displays all attributes
  getFtp enabled : displays a single attribute's value
-----
```

setFtp

```
-----
Syntax:  setFtp attribute value

Where:
  []      : indicates an optional parameter
  attribute : is one of the following (or its alias):

    enabled  --
    port     --
    auto_off --

  value for enabled : is true|false or on|off
  value for port    : is an integer (default is 21)
  value for auto_off : is integer of minutes, 0 = never

Examples:
  setFtp enabled true : enables FTP service
  setFtp port 2100    : binds FTP to port 2100
  setFtp auto_off 20  : FTP disables 20 mins after being enabled
-----
```

getSntp

```
Syntax: getSntp [attribute]
```

```
Where:
[]      : indicates an optional parameter
attribute : is one of the following (or its alias):

server      --
port        --
auth_method --
account     --
password    --
```

```
Examples:
getSntp      : displays all attributes
getSntp account : displays a single attribute's value
```

setSntp

```
Syntax: setSntp attribute value
```

```
Where:
[]      : indicates an optional parameter
attribute : is one of the following (or its alias):

server      --
port        --
auth_method --
account     --
password    --
test_email  (email, email_test)
test_sms    (sms, sms_test)
```

```
value for server      : must be an IP address
value for port        : is an integer (default is 25)
value for auth_method : is none|any|plain|login|cram|digest
value for account     : is an email address on the SMTP server
value for password    : is the account's password
value for test_email  : is a quoted comma list of one or more
                        user login names or email addresses
value for test_sms    : is a quoted comma list of one or more
                        user login names or sms phone@gateway addresses
```

```
Examples:
setSntp account pdu@xxx.com
setSntp server 192.168.0.124
setSntp test_email "root,me@example.com"
setSntp test_sms "root,8885551212@txt.carrier.net"
```

Log Commands

getLog

```
-----
Syntax:  getLog [severity option] [category option]

Where:
  []      : indicates an optional parameter
  attribute : is one of the following (or its alias):

          severity      (sev)
          category      (cat)

severity option is one of the following:
critical, error, warn, info, debug

category option is one of the following:
alarm, email, switch, config, startup, login,
files, time, log, system, kernel

Examples:
getLog                  : shows all log lines
getLog severity error   : shows error entries for all categories
getLog category switch  : shows only switch category log lines
getLog sev error cat files : shows only errors related to the file system
-----
```

getStartupLog

```
-----
Syntax:  getStartupLog

Examples:
getStartupLog          : shows all log lines
-----
```

System Commands

getSystem

```
-----
Syntax:  getSystem [attribute]

Where:
  []      : indicates an optional parameter
  attribute : is one of the following (or its alias):

    label      --
    location    --
    model_number (model, model_no)
    serial_number (serial, serial_no)
    mac_address  (mac, mac_addr)
    version      --
    memory_map   --
    start_time   (started)
    time         --
    file_formats formats

Examples:
  getSystem label      : gets one attribute of the system
  getSystem             : gets all attributes of the system
-----
```

setSystem

```
-----
Syntax:  setSystem attribute [value]

Where:
  []      : indicates an optional parameter
  attribute : is one of the following (or its alias):

    label      --
    location    --
    restart    --
    reset!!!    --

    value for label      : is a quoted string (30 chars max)
    value for location   : is a quoted string (30 chars max)
    value for time       : is formatted like "YYYY-MM-DD HH:MM:SS" (24 hr)
    value for restart    : none
    value for reset      : none (be careful!)

Examples:
  setSystem label "WWW Servers" : sets label of the system
  setSystem restart              : restarts the system software
  setSystem reset!!!            : erases all settings and restores
                                factory defaults. There is NO extra
                                "are you sure" step. Be careful!
-----
```

MPD Series RCM Software

RESTful API Reference

RESTful API Reference

This chapter includes an overview of what a RESTful API is, and the details of using the RCM Software implementation.

The RESTful API for RCM Software is designed to support optimized machine-to-machine control of a PDU by a remote script, primarily in the application of automated test and evaluation (“ATE”). Therefore, the focus of the API is to provide access to power management features such as switching outlets and collecting power data where available. The API does not support access to non-power-related resources such as Users, Logs, configuration settings, etc. It is possible to automate access to these resources using Telnet commands.

What is REST and RESTful API?

The Gist of REST

REST is an acronym for REpresentational State Transfer which at an academic level boils down to being an architectural style of how to represent information on a distributed hypermedia system. In other words, it's a particular style of how to design links and connections to information resources on a computer network (such as the internet).

It was originally described in 2000 (see the reference material sidebar), but has gained rapid popularity in recent years. During that time, there's been significant chatter amongst software developers about the academic and pragmatic nuances. For the purposes of the RCM Software, the objectives and implications are far simpler than what has to be considered for more complex applications.

Reference Materials for REST

The canonical academic paper which defined REST:

http://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm

Some additional internet discussions about REST:

http://en.wikipedia.org/wiki/Representational_state_transfer

<http://www.ibm.com/developerworks/webservices/library/ws-restful/>

One academic point to cover is the difference between the terms REST and RESTful. REST is an architecture not a protocol (such as HTTP, FTP, etc.). Therefore, a system does not access information via REST. Rather, the data is accessed in a REST-like or RESTful manner. The latter term has the broader usage, and thus we end up with the term RESTful API—a programmer’s interface to something using a RESTful design style.

Programming for a RESTful API

Since the REST architecture was originally conceived in the context of hypermedia (i.e. the world wide web), a fundamental tenet of REST is the use of HTTP as the communication protocol.

There are plenty of good resources online describing REST and RESTful API design. However, the bottom line for the RCM Software is that “messages” are sent to the PDU using a combination of HTTP URIs, verbs, and form data. For example to switch an outlet on, the verb and URI combination of:

```
PATCH http://192.168.1.10/outlet/3/switch
```

would be sent along with name-pair form data of an existing session and the new state value for the outlet switch like this:

```
session_id=Ygv0PkCheJr14UuyPfd10FUpD807XQ4Y  
new_value=on
```

The response to this request will be a standard HTTP response with the body having a simple string. Since all response to RCM requests are single values, the RCM Software uses plain text responses rather than incur unneeded overhead of XML, JSON, or HTML parsing (common formats for RESTful responses).

Programming for a RESTful API involves using an HTTP library for your programming language of choice in order to deliver an HTTP request to the

PDU, and acquire HTTP responses which are usually already parsed into their header and body components. Again, there are plenty of online resources to help overcome the learning curve.

Why Use the RESTful API?

The RESTful API is provided as an alternative to the common practice of scripting to a Telnet interface or even SNMP. First, Telnet is intended to be a human interface. From the machine perspective, there’s a lot of wordy noise to parse out to acquire just the facts needed. The RCM RESTful API eliminates all that noise. SNMP is a machine interface, but can be rather complex to develop interfaces with. Since a RESTful API is basic HTTP, it can be much easier to develop quick, custom automation scripts with.

HTTP Verbs

The RCM Software RESTful API makes use of GET, POST, PATCH, and DELETE behaviors. There is no need for PUT, and HEAD is not supported. However, the embedded web server supports GET, POST, and HEAD request methods. Since HEAD is not used, there’s no conflict there. However, to provide support for PATCH and DELETE, the API uses a common technique of creating a pseudo-verb through a form parameter.

Whenever a URI needs GET or POST, the standard HTTP request method can be specified. However, when PATCH or DELETE are needed, the URI must be submitted with POST, and the request must include a form parameter named `rest_method` along with a string value of “PATCH” or “DELETE” as needed. For example, while the following would be ideal:

```
PATCH https://192.168.0.10/outlet/1/switch?  
new_value=on&session_id=abcxyz
```


it is necessary to accomplish that message this way:

```
POST https://192.168.0.10/outlet/1/switch?
new_value=on&session_id=abcxyz&rest_method=PATCH
```

This form-parameter technique is quite common in RESTful APIs since many web servers (and browsers) do not support all HTTP verbs. The world wide web got started with GET, POST, and HEAD, and thus we are left to create a work around for those verbs not widely supported yet.

Responses

Following the principles of focus and simplicity for the API, it is assumed the automated test environment is one in which the power architecture (how many power sources, circuits, outlets, etc.) of the PDU is a known entity which will remain stable over the life of the script. Therefore, the API does not support discovery of the PDU power architecture. That is, it does not support requesting collections which could be parsed to navigate or “walk” available resources. The API is focused on delivering and changing state of specific attributes of resources already known to the remote script.

The result of this simplicity is that all responses to RESTful API requests are available as an html/text media type only, and have only a single value. In effect, each response is a simple string of a single value.

While these limitations might been seen as restrictive from the traditional RESTful API perspective, it helps to focus the limited resources of an embedded controller on the most value-added capabilities for the majority of applications.

If a request fails, or was malformed to start with, a return value of some type is always provided. Errors are identified as negative numbers.

RCM REST Error Codes

- 1 General error. A low-level internal error happened where no specifics were being tracked as to the cause.
- 12 the new_value value is out of range for the attribute
- 200 the resource ID in the URL is not valid (too big or too small)
- 217 the string for the switch state new_value is not valid
- 298 the resource attribute in the URL is not valid
- 317 session_id passed by the request is not recognized. Start a new session to get a new id.

Authentication API

MPDe resources are not public. A conversation over the RESTful API must be initialized with an authentication which results in the creation of a session. Each subsequent request must be authenticated with a session ID.

A session ID is obtained with a POST to /session with parameters `account=&password=`. A 32-character string is returned. That string has to be submitted as part of all other requests in a parameter named `session_id=`.

Reminder: for maximum security on any open network, the PDU should have HTTPS enabled to encrypt the authentication dialog which occurs between the automation script and the PDU web server.

API Resources

The following sections identify the available RESTful resources and the API requirements for each. Remember that all requests require authentication. See the section above for details.

/session

A session must be created before any other PDU resource can be requested. Then, the session id must be passed to all subsequent requests, and when the session's use is no longer needed, it should be destroyed.

The RESTful API session space is shared with the regular web user space. If a script makes use of a specific user login, then a human (or another script) subsequently uses the same login credentials, the originally running script will be blocked because its session will have been destroyed (overwritten by the

second session). Each script and each human should have unique user login credentials.

POST /session

URI Parameters:

- none

Form Parameters

- account (required) — the user account name
- password (required) — the password for the account (be sure this is URL encoded)

Response

- (session_id) a 32-character string which must be saved to send along with future requests

DELETE /session

URI Parameters:

- none

Form Parameters

- session_id (required) — the previously acquired ID
- rest_method (required) — must be “DELETE” in upper case

Response

- a string of 32 zeros

/outlet

There will always be one or more outlets. The resource GET/outlets is not supported, the script must know which outlet of the PDU to request. Each attribute of the outlet is requested directly as identified in the sections below.

GET /outlet/:id/label

URI Parameters:

- :id — the PDU's numeric instance of the Outlet (values start at 1, a label cannot be used as an :id)

Form Parameters:

- session_id — the string returned from a POST /session

Response

- a string of arbitrary words labeling the outlet instance

GET /outlet/:id/switch

URI Parameters:

- :id — the PDU's numeric instance of the Outlet (values start at 1, a label cannot be used as an :id).

Form Parameters:

- session_id — the string returned from a POST /session

Response

- a value of “on” or “off” would be normal
- a value of “unknown” means the outlet status cannot be determined (this may require a power cycle, and/or an All On / All Off cycle of the outlets to clear).
- a value of “stuck on” or “stuck off” means the actual outlet status is out of sync with what the RCM controller expects it to be. This may indicate a hardware failure, or may require an All On / All Off cycle of the outlets to clear.

GET /outlet/:id/rated_amps

URI Parameters:

- :id — the PDU's numeric instance of the Outlet (values start at 1, a label cannot be used as an :id)

Form Parameters:

- session_id — the string returned from a POST /session

Response

- a numeric string (i.e. “20”) indicating the maximum amps rating of the outlet

GET /outlet:id/rated_volts

URI Parameters:

- :id — the PDU's numeric instance of the Outlet (values start at 1, a label cannot be used as an :id)

Form Parameters:

- session_id — the string returned from a POST /session

Response

- a numeric string (i.e. "125") indicating the maximum volts rating of the outlet

GET /outlet:id/startup_state

URI Parameters:

- :id — the PDU's numeric instance of the Outlet (values start at 1, a label cannot be used as an :id)

Form Parameters:

- session_id — the string returned from a POST /session

Response

- "on" or "off" or "last known" corresponding to the Startup State options

GET /outlet:id/on_delay

URI Parameters:

- :id — the PDU's numeric instance of the Outlet (values start at 1, a label cannot be used as an :id)

Form Parameters:

- session_id — the string returned from a POST /session

Response

- a numeric string (i.e. "1.50" or "2") representing seconds

GET /outlet/:id/off_delay

URI Parameters:

- :id — the PDU's numeric instance of the Outlet (values start at 1, a label cannot be used as an :id)

Form Parameters:

- session_id — the string returned from a POST /session

Response

- a numeric string (i.e. "1.50" or "2") representing seconds

GET /outlet/:id/cycle_delay

URI Parameters:

- :id — the PDU's numeric instance of the Outlet (values start at 1, a label cannot be used as an :id)

Form Parameters:

- session_id — the string returned from a POST /session

Response

- a numeric string (i.e. "1.50" or "2") representing seconds

PATCH /outlet/:id/switch

Actually sent as a POST, but identified as a PATCH through a form parameter.

URI Parameters:

- :id — the PDU's numeric instance of the Outlet (values start at 1, a label cannot be used as an :id)

Form Parameters:

- session_id — the string returned from a POST /session
- rest_method — must be "PATCH" in upper case
- new_value — a value of "on" turns the outlet on (apply power to it), a value of "off" turns the outlet off (remove power from it), and a value of "cycle" turns the outlet off, pauses for cycle_delay time, then turns the outlet back on. Regardless of whether the first state is on or off, the cycle command is always off-on (it never cycles on-off).

Response

- if new_value is "on" the result should also be "on"
- if new_value is "off" the result should also be "off"
- if new_value is "cycle" the result should be "on"

PATCH /outlet/:id/startup_state

Actually sent as a POST, but identified as a PATCH through a form parameter.

URI Parameters:

- :id — the PDU's numeric instance of the Outlet (values start at 1, a label cannot be used as an :id)

Form Parameters:

- session_id — the string returned from a POST /session
- rest_method — must be "PATCH" in upper case
- new_value — a value of "on" or "off" or "last known" corresponding to the Startup State options

Response

- the response should be the same string as what was submitted

PATCH /outlet/:id/on_delay

Actually sent as a POST, but identified as a PATCH through a form parameter. URI

Parameters:

- :id — the PDU's numeric instance of the Outlet (values start at 1, a label cannot be used as an :id)

Form Parameters:

- session_id — the string returned from a POST /session
- rest_method — must be "PATCH" in upper case
- new_value — a numeric string such as 0.5, 2.0, or 1

Response

- the response should be the same string as what was submitted

PATCH /outlet/:id/off_delay

Actually sent as a POST, but identified as a PATCH through a form parameter. URI

Parameters:

- :id — the PDU's numeric instance of the Outlet (values start at 1, a label cannot be used as an :id)

Form Parameters:

- session_id — the string returned from a POST /session
- rest_method — must be "PATCH" in upper case
- new_value — a numeric string such as 0.5, 2.0, or 1

Response

- the response should be the same string as what was submitted

PATCH /outlet/:id/cycle_delay

Actually sent as a POST, but identified as a PATCH through a form parameter. URI

Parameters:

- :id — the PDU's numeric instance of the Outlet (values start at 1, a label cannot be used as an :id)

Form Parameters:

- session_id — the string returned from a POST /session
- rest_method — must be "PATCH" in upper case
- new_value — a numeric string such as 0.5, 2.0, or 1

Response

- the response should be the same string as what was submitted

/system

System identification can be useful for automated logging. The script will already know which IP address is being connected to, but the System label and location can provide addition arbitrary user description of the PDU.

GET /system/1/label

URI Parameters:

- note that in place of :id, this resource is always called with a fixed 1 for the ID

Form Parameters:

- session_id — the string returned from a POST /session

Response

- a string of arbitrary words labeling the PDU system

GET /system/1/location

URI Parameters:

- note that in place of :id, this resource is always called with a fixed 1 for the ID

Form Parameters:

- session_id — the string returned from a POST /session

Response

- a string of arbitrary words intended to represent the PDU's location

MPD Series RCM Software

Settings Files Reference

Settings Files Reference

This chapter includes an overview of the settings files format, syntax, and default content, and how the file might be used to save time and effort managing multiple PDUs.

Know What You're Doing

Read the warnings at the right!

The safest way to change settings on an MPD RCM PDU is to use the built-in web interface or command line. However, it is understood that making changes to multiple PDUs this way can be time consuming.

Making settings available in user-editable files is an advanced capability. It is intended for people with experience in managing server-like configuration files, and who are practiced at learning and recognizing the format, syntax, and requirements of such files.

Not every suitable scenario for or consequence of editing files directly are explained in this guide.

Accessing Settings Files

Settings files are accessed through FTP. Using the IP address of the PDU as the FTP server identification, login with the root user account. This will expose the file system of the PDU. It is highly recommended that files be copied to a computer or tablet for editing. Only after all changes are completed, should the files be uploaded, and then the PDU restarted.

EDITING SETTINGS FILES DIRECTLY CAN RESULT IN A MISCONFIGURED PDU WHICH DOES NOT OPERATE AS EXPECTED.

It is highly recommended that if you plan to edit files directly, that you have a non-mission-critical PDU available to experiment, practice, and learn with. Only after you're comfortable (and perhaps your boss is comfortable) that you know what you're doing should you work on the files of PDUs in service.

Marway Power Solutions has attempted to make the MPD RCM software self-healing from the affects of malformed or missing settings files. However, we cannot guarantee that every conceivable malformed variation within a file or combination of files will not result in a PDU which must be returned to the factory to restore it to complete operation.

Be careful. Test your changes on a non-mission-critical PDU before deploying.

AFTER UPLOADING SETTINGS FILES THE PDU MUST BE RESTARTED FOR THE CHANGES TO TAKE EFFECT.

If the PDU is not restarted, any other user making changes via the web or CLI could overwrite the changes just uploaded.

File Format and Syntax

Settings files are ASCII text with “Windows-style” line endings (\r\n). Most files are not whitespace sensitive, but powerSettings.txt is. The format of each file is detailed in the sections below.

Name/Value Pairs

All files include name/value pairs where a setting name is followed by an = and the setting value. The whitespace surrounding the = is not significant. Most files have whitespace before = in order to neatly align the setting values, but this spacing is not meaningful. If the file is manually edited, and the whitespace altered, it will revert back to the aligned spacing whenever the RCM software updates the file. Examples of these pairs (not from the same file):

```
loginName = root
ipv4Address = 192.168.1.10
```

Objects and IDs

Some files include named objects with IDs in the format of `object~N` where `object` is a label such as `outlet`, `circuit`, `user` (and others), and where `N` is a simple incremental integer (1, 2, 3, etc.). The whitespace around the `~` is not significant, but will be removed whenever the RCM Software updates the file. Examples of objects and IDs:

```
outlet~3
user~root
```

A block like this will be used to show a whole file (if it can fit on the page).

A block like this will be used to show a fragment of a file (that is, the whole file is not being shown, only a portion of it).

Object Groups of Settings

Files using objects will have settings for that object grouped together under the object. The grouped settings will be indented. In most files (except `factorySettings.txt` and `powerSettings.txt`), the indentation will be two spaces. These spaces aren't meaningful, but will be restored automatically when the file is updated by the RCM Software. An example of an object and its settings:

```
outlet~8
  label          = Frequency Generator A
  switchState    = on
  startupState   = last known
  onDelay        = 0.10
  offDelay       = 0.00
  cycleDelay     = 2.00
```

File Identity and Version

The first two lines of a settings file are critical to identifying the contents of the file. The first line identifies the file type, and the second line identifies the format version of the file (not the version of the RCM Software).

Being a part of the text file, these values are technically editable (but they should never be edited), therefore it is critical to ensure they remain correct.

systemSettings.txt

Refer to “System Settings” on page 43 for details about the settings.

This is a simple file consisting of only the file identity a couple of settings. The entire file is shown at the right. The formatting of the file consists of basic name/value pairs. There are no objects.

networkSettings.txt

Refer to “Network Settings” on page 44 for details about the settings.

This file consists of only the file identity with basic name/value pairs. There are no objects. The order of the protocols and the spacing between them are not significant, but the RCM Software will restore its default formatting whenever the file is updated by the web or CLI.

userSettings.txt

Refer to “User Settings” on page 48 for details about the settings.

This file consists of a file identity and multiple user objects. Each user has a sequential object ID starting with 1, and is followed by several settings (not all are shown in the sample at the right).

There is one required user whose loginName must be root. The root user must be the first user in the file and have an ID of 1.

```
identity = RCM_SYS
version  = 1.0.0

// NOTE: user added comments are not preserved in this file.
// See the User Guide for details about format and options.

label      = Test Bay 2 PDU
location   = R&D Building 112
```

(a fragment of networkSettings.txt)

```
identity = RCM_NTWK
version  = 1.0.0

// NOTE: user added comments are not preserved in this file.
// See the User Guide for details about format and options.

ipv4Dhcp      = on
ipv4Address    = 192.168.1.10
ipv4Subnet     = 255.255.0.0
ipv4Gateway    = 192.168.1.1

ipv6Dhcp      = on
ipv6Address    =
ipv6PrefixLength = 0

httpPort      = 80
httpsEnabled   = false
httpsPort     = 443
httpSessionMins = 30
```

After the root user, the order of the users is not significant, though their IDs must increment by one (1, 2, 3, etc.).

The order of the settings within the user object and the spacing between user objects is not significant, but the RCM Software will restore its default formatting whenever the file is updated by the web or CLI.

If you need but do not know the root user password, the correct way to reset and update the root user password is explained in “[Change root User Password](#)” on page 10.

powerSettings.txt

Refer to “[Outlet Settings](#)” on page 53 for details about the settings.

This file consists of a file identity and multiple nested objects. Each level of nested objects has its own series of sequential object IDs starting with 1. For example, a unit may have three Circuits numbered 1, 2, and 3. There may also be Outlets with IDs of 1, 2, and 3. The incremental numbering starts over with each change of object type. Each object is followed by several settings (not all are shown in the sample at the right).

The MPD series RCM Software is adapted from a more complex line of products which support many features per power object (Sources, Inputs, Lines, etc.). These objects will be listed in the powerSettings file, but will rarely have any attributes for an MPD series product where the remote control emphasis is on outlet switching. To generate a factory default file, delete the powerSettings.txt file, and restart the RCM Software. A new file will be created.

(a fragment of userSettings.txt)

```
identity = RCM_USER
version = 1.0.0

// NOTE: user added comments are not preserved in this file.
// See the User Guide for details about format and options.

user~1
  loginName      = root
  loginPswd      = c33e6f892497cc56e23d8159834a0db299e27c8b
  pswdSalt       = 15bGJp1QpJmm
  pswdHint       = p8
  // (big piece cut out of this example)

user~2
  loginName      = lester
  loginPswd      = c122bc8666ec22658f35cb0f0b438d85d56d8ade
```

In the example listing at the right, the > characters are not a part of the text file, but rather an illustrative way to show how many tab characters have been inserted before the start of the text on that line.

As discussed previously, the spacing around the = symbols is not significant. In this file, only the number of tabs before the start of the text is significant.

The order of the objects is significant (source, input, line, circuit, outlet). The order of the individual settings underneath each object is not significant. However, this does not mean that something like `switchState` can be anywhere. It must be under, and correctly indented for, the outlet object to which the setting belongs.

While it is generally safe to edit the values of the settings, it is not advisable to alter the structure or order of the lines within file.

factorySettings.txt

This file is not editable, not deletable, and not replaceable. It may be copied for reference and archiving. It is formatted similarly to `powerSettings.txt`.

WHITE SPACE MATTERS in `powerSettings.txt`

The `powerSettings.txt` file IS whitespace sensitive—it uses meaningful tab indentation. Make sure your text editor does NOT replace tabs with spaces.

(a fragment of `powerSettings.txt`)

```
identity = RCM_POWR
version  = 1.0.0

// NOTE: user added comments are not preserved in this file.
// See the User Guide for details about format and options.
// This file's format IS sensitive to the leading whitespace tabs.
// Be sure your editor inserts tabs, and does not substitute spaces.

source~1
> label                = Source 1
> input~1
> input~2
> line~1
>>   circuit~1
>>>   label            = Circuit A
>>>   outlet~1
>>>>   label           = Outlet 1
>>>>   switchState      = on
>>>>   startupState     = last known
>>>>   onDelay          = 0.20
>>>>   offDelay         = 0.00
>>>>   cycleDelay       = 5.00
>>>   outlet~2
>>>>   label           = Outlet 2
>>>>   switchState      = on
>>>>   startupState     = last known
>>>>   onDelay          = 0.20
>>>>   offDelay         = 0.00
>>>>   cycleDelay       = 5.00
```

MPD Series RCM Software

Firmware Updates

Firmware Updates

Should there be a need to update the firmware in the MPD RCM PDU, there's a very simple way to do that. A single file named `image.bin` is the main application for the PDU. The file will be enclosed in a folder identifying its version and accompanied by release notes, but the upload file will always be named simply `image.bin`.

Use an FTP client to upload the file to the on-board FLASH0 volume of the PDU. (What you see with your FTP program may look different to what's shown on the right). Wait approximately 2 minutes after the upload to ensure the file has been successfully copied and installed, then restart the PDU using the web interface or command line.

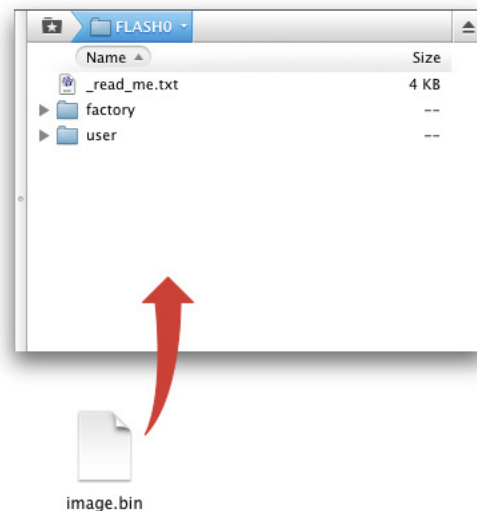
After the file has uploaded, it is moved to another location, so it won't remain visible in the FTP file listing.

FIRMWARE UPDATE SUPPORT

All firmware updates which have been made for security or bug patches are available at no cost from Marway's web site. Look for firmware updates at:

http://www.marway.com/pdu/mpd_rcm_software

There may or may not be any which apply to your system. Other conditions may apply. Check your warranty, and possibly your purchase agreement if your PDU was custom designed.



RCM Software™ for MPD Series™ Switched Outlet PDUs

Software Version: 1.2.x

Owner's Guide P/N: 501014-002

© 2013-2015, Marway Power Systems, Inc. All rights reserved.



Marway Power Solutions
1721 S. Grand Ave., Santa Ana, CA 92705
800-462-7929 • marway@marway.com