MARWAY
POWER SOLUTIONS

Optima RCM™ Software

User Guide and Reference

Software Version 2.0.x

Aug 2017 : P/N 501014-2.0.0-B

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/software

Optima RCM Software

User Guide and Reference

Software Version 2.0.x

Events Management

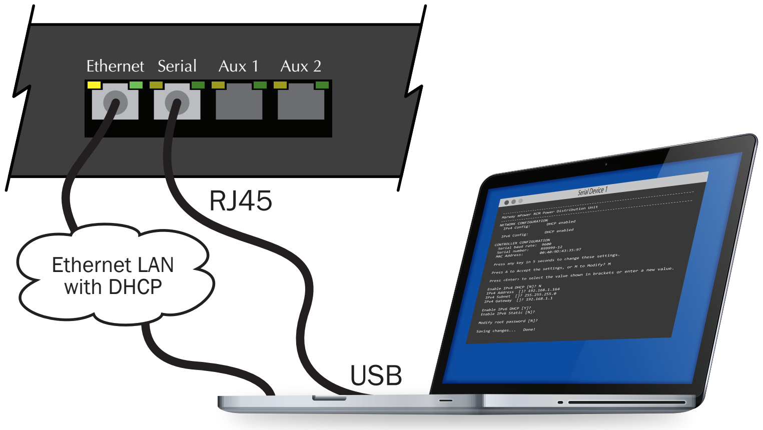
MPD Series RCM Software

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 Ethernert LAN ready for manual or DHCP IP connectivity.
- Two Ethernet cables (one for the product, 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.
  - We like CoolTerm for Windows or macOS (there are many others).
  - If you're a Linux user, we'll assume you're using the CLI or already have a favorite GUI app.
- The serial port configured for 9600 baud, 8 data bits, 1 stop bit, no parity. (Baud rate can be changed to as high as 115,200 later on.)

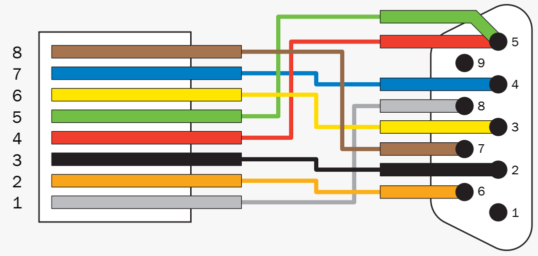


TCP/IP NETWORKING IS ON BY DEFAULT.

Be aware that TCP/IP is enabled by default. Therefore, as soon as the RCM product is connected to a network, communication with it technically possible. However, since each unit requires setup using the Serial console to start with, each unit ships with a randomized root user password which will be changed durimg the setup process. This random password, in theory, helps prevent access to devices even if they don't get properly configured after being connected.

Serial Cables

The Marway PDU includes an RJ45 "Cisco-style" serial console port. Even though it uses an RJ45, this is not an Ethernet port. To use this port, you will need a Cisco 72-3383-01 compatible cable. These cables have RJ45 on one end and DB-9 on the other end wired as shown below. If you have a DB-9 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 DB-9 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. You should find multiple sources for products using web search terms "Cisco 72-3383-01 Cable" and "DB-9 to USB serial adaptor."



RJ45	DB-9
8	7
7	4
6	3
5	5
4	5
3	2
2	6
1	8

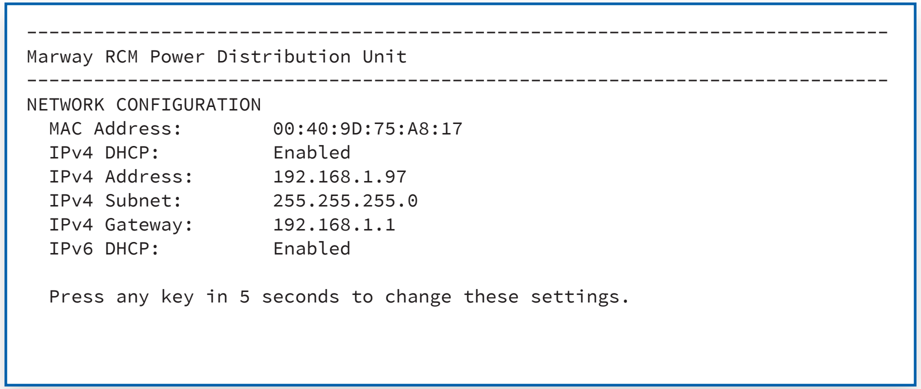
Next, with the power distribution product powered off, follow these steps.

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



```
-----------------------------------------------------------------------------
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.
```

Depending on your network setup, you will see completely different numbers. RCM software ships with DHCP enabled, but you can change that in the next few steps.

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.

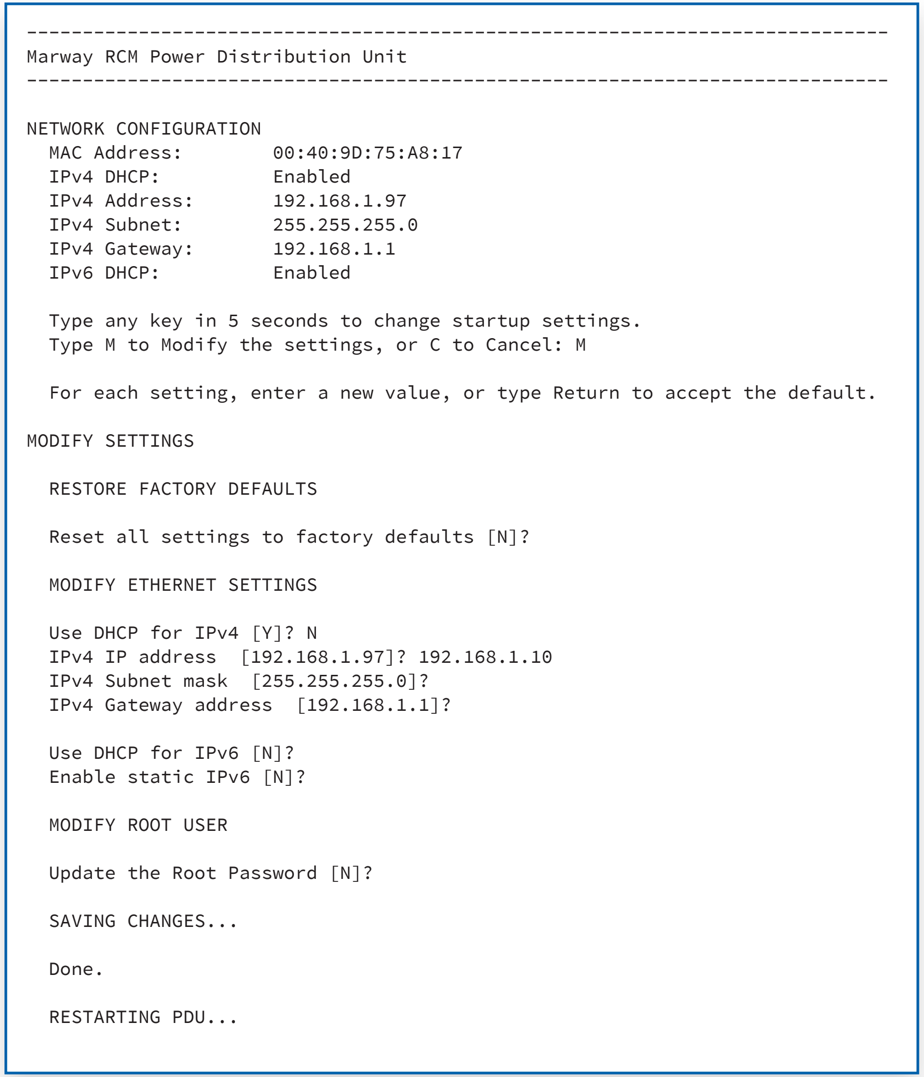
After confirming you want to `Modify` the PDU settings, the first TCP/IP prompt will be presented (with either a `[Y]` or a `[N]` depending on the current configuration):

```
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 (unless you're configuring DHCP to assign addresses based on the MAC address).

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 session 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 [N]?
  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

For new systems, or systems which have 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: There is no default password. A random password has been created at the time of shipment, and even the factory does not know what it is. You MUST use the serial dialog to change the root password to something you will continue to use.

```
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. An accurate time is not crictical for basic operations, 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 are coordinated with 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.

If the PDU is connected to an SNTP server, the local clock will not be manually editable since SNTP will be automatically keeping the time updated.

SNTP configuration is done in the Network tab of the web interface. When there are no IP address assignments to the SNTP servers, the internal clock is configured in the System tab of the web interface.

Startup TCP/IP Troubleshooting

If the PDU has begun a normal startup, but the PDU is not reachable to log into via web browser or command line (and the PDU's on board display is either blank or shows dashes even after several minutes), return to the serial interface, and restart the PDU again.

Time Servers (SNTP)

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.

The normal startup appearance through the serial console is shown to the right. After about a minute, try entering the command `getSystem`. The PDU should respond with a number of descriptive details about the PDU. If the response is an error message, wait another 15 seconds and try again (just to be sure we give the system enough time to start). PDUs with a high number of outlets (> 32) and with a large number of log entries (> 200, which is generally not the case for new setups), even two minutes should be enough time.

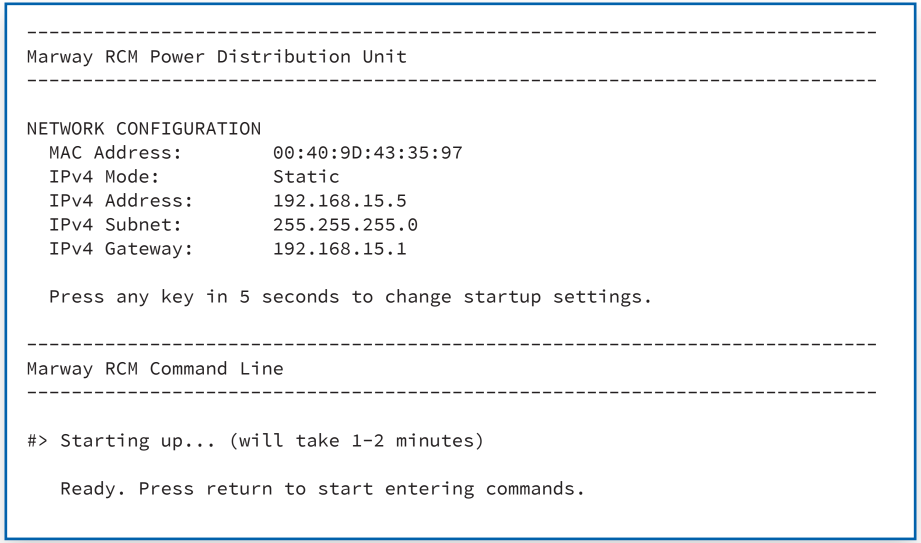
If the response to `getSystem` is still an error message, the next thing to check is that a valid TCP/IP address is assigned to the device. Enter the command `ifconfig` in the serial console. If the product has correctly connected to the network, you should see a response similar to the second window shown to the right. Note that a full address is listed for both IpAddr and GW.

If the product has not correctly connected to the network, you should see a response similar to the third window shown to the right which indicates that the connection "is down." This could mean either the TCP/IP setup done in the serial dialog has a misconfigured value, or that there is a conflict with another device which is already using the address that was assigned.

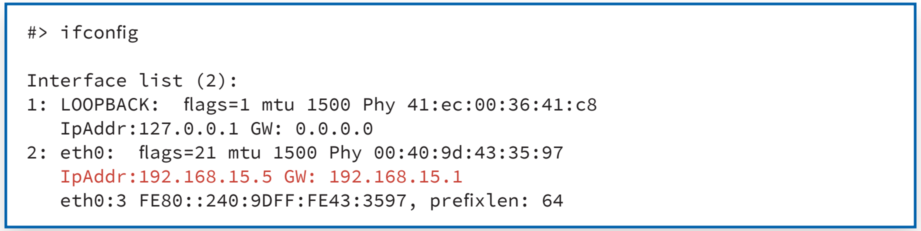
Check for address conflicts, and return to the serial dialog to re-enter the TCP/IP settings.

Up to Date Release Notes

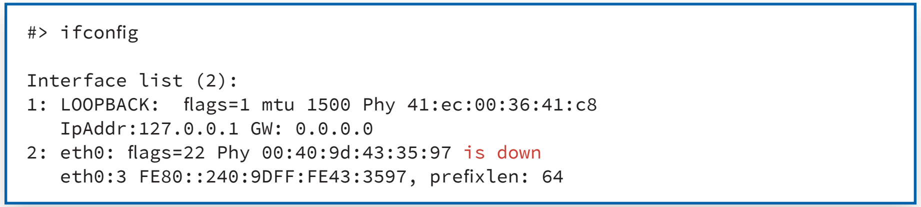
Be aware that in the FTP file system is a _Read-Me-RCM.pdf file which contains release notes as of the software installation at the factory (and may be kept up to date with updates from the Marway web site). There may be details offering new or corrected information not included in this User Guide.



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

NETWORK CONFIGURATION
  MAC Address:        00:40:9D:43:35:97
  IPv4 Mode:          Static
  IPv4 Address:       192.168.15.5
  IPv4 Subnet:        255.255.255.0
  IPv4 Gateway:       192.168.15.1

  Press any key in 5 seconds to change startup settings.

----------------------------------------------------------------------------
Marway RCM Command Line
----------------------------------------------------------------------------

#> Starting up... (will take 1-2 minutes)

   Ready. Press return to start entering commands.
```



```
#> ifconfig

Interface list (2):
1: LOOPBACK:  flags=1 mtu 1500 Phy 41:ec:00:36:41:c8
   IpAddr:127.0.0.1 GW: 0.0.0.0
2: eth0:  flags=21 mtu 1500 Phy 00:40:9d:43:35:97
   IpAddr:192.168.15.5 GW: 192.168.15.1
   eth0:3 FE80::240:9DFF:FE43:3597, prefixlen: 64
```



```
#> ifconfig

Interface list (2):
1: LOOPBACK:  flags=1 mtu 1500 Phy 41:ec:00:36:41:c8
   IpAddr:127.0.0.1 GW: 0.0.0.0
2: eth0: flags=22 Phy 00:40:9d:43:35:97 is down
   eth0:3 FE80::240:9DFF:FE43:3597, prefixlen: 64
```

Optima RCM Software

Feature Tour

Feature Tour

Marway's Optima RCM power distribution unit offers multiple interfaces to monitor and/or control the unit. There are three user interfaces: a web browser graphical interface ("web UI"), a command line interface ("CLI"), and an on-board LED display for power data and relay state.

Additionally, the system may be monitored and controlled through SNMP and a RESTful API.

Lastly, a serial interface is used for initial setup of the root user account and IP network settings. Upon startup, the serial interface will display startup status information, and prompt the user for changes. If there are no changes to be made, the serial interface presents the standard CLI.

This section of the *User Guide* is an introductory tour of the major features of the web UI and CLI. It presents interface images along with short descriptions of several of the interface and functional features of the PDU.

Web Browser Interface

The browser interface has been designed for reasonably broad compatibility with 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 power data monitoring, outlet 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 views have their own help button on the right side of the panel title bar.

Command Line Interface

The command line interface works over Telnet, SSH, and the serial port. It allows for power data monitoring, outlet 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 usually protected by a circuit breaker (unless an outlet is wired as a pass-through from the main breaker).
- Phase : unfortunately in the power industry, this same term describes two possible objects: a pair of conductors (wires) across which exists a voltage potential, or a single conductor on which a current flows. Voltage phases will be shown with two letters such as AB, AN, or LN. Current phases will have a single letter (A, B, C, L).
- Inlet : for alternating current power, an inlet is comprised of the phase conductors, whereas for direct current power, an inlet is comprised of the positive and negative conductors.

A PDU will have one or more power Inlets. Phases are the wires directly connected to the power source. Each phase may be branched into one or more Circuits, and each circuit would have one or more Outlets (with a variety of possible connector styles). Throughout the various user interfaces, Circuits are rarely seen as separately visible objects.

Power and Switching Data

The table to the right identifies the data values available in various PDU configurations. These values are available to view on the web Dashboard, and through the `getOutlet` and `getPhase` CLI commands.

Power Monitored systems include measuring of voltage and current of the inlet phases. From this, the other values are derived, providing a more complete power data set.

Current Monitored systems measure only the current of the inlet phases. This is useful for understanding the total loads present at the inlet breaker(s).

Switched Outlets do not include any power monitoring of the outlets, but provide control of switching each individual outlet on and off.

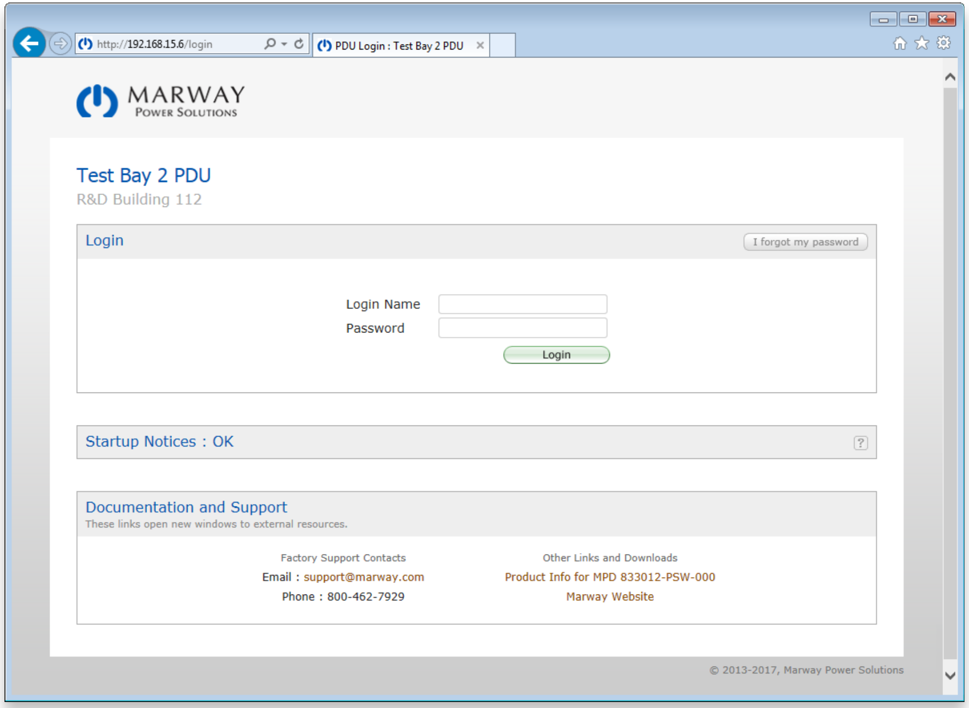
Any given PDU will have either power monitoring or current monitoring of the inlet, or neither in the case of a switched-only system. A monitored-only system may not include switched outlets, but otherwise most systems include outlet switching as that's usually a key feature desired from networked PDUs.

Value (units)	Power Monitored Inlet	Current Monitored Inlet	Switched Outlets
Rated Volts (V)	✓	✓	
Volts (V)	✓		
Rated Amps (A)	✓	✓	
Amps (A)	✓	✓	
Consumed Amps (%)	✓	✓	
Watts (W)	✓		
VoltAmps (VA)	✓		
VA Reactive (VAR)	✓		
Power Factor (PF)	✓		
Frequency (Hz)	✓		
Switch State (on/off)			✓

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 shown in this window.



http://192.168.15.6/login

PDU Login : Test Bay 2 PDU

Test Bay 2 PDU

R&D Building 112

Login

Login Name

Password

Startup Notices : OK

Documentation and Support

These links open new windows to external resources.

Factory Support Contacts

Email : support@marway.com

Phone : 800-462-7929

Other Links and Downloads

Product Info for MPD 833012-PSW-000

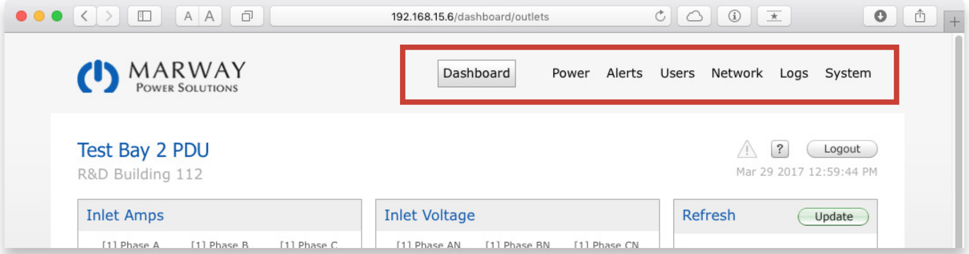
Marway Website

© 2013-2017, Marway Power Solutions

Main Menu

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

- Dashboard : outlet control, power data monitoring, and alarm monitoring. Other than logs, Dashboard is where virtually all information viewing is performed. All other main menu tabs are primarily for adjusting settings.
- Power : settings for power device labels, setpoints, and outlet delays
- Alerts : settings for notifications when specific events occur such as setpoint trips
- Users : settings for user profiles, authentication, and authorization
- Network : settings for supported networking protocols and features
- Logs : display of the system and startup logs
- System : settings for unit labeling and clock, command for restart, and miscellaneous system-level information



192.168.15.6/dashboard/outlets

Test Bay 2 PDU

R&D Building 112

Mar 29 2017 12:59:44 PM

Inlet Amps

[1] Phase A [1] Phase B [1] Phase C

Inlet Voltage

[1] Phase AN [1] Phase BN [1] Phase CN

Refresh

Web Dashboard

The Dashboard is where all power monitoring, alarm monitoring, and outlet switching capabilities are 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 on the System page, and allow the PDU to have a unique and meaningful identification. On the right, a caution icon (triangle with the !) will be colored a bold yellow if there are any major alarms such as setpoints or system errors which should be attended to. This very top section of the page is common to all pages in the web UI.

Inlet Current and Voltage

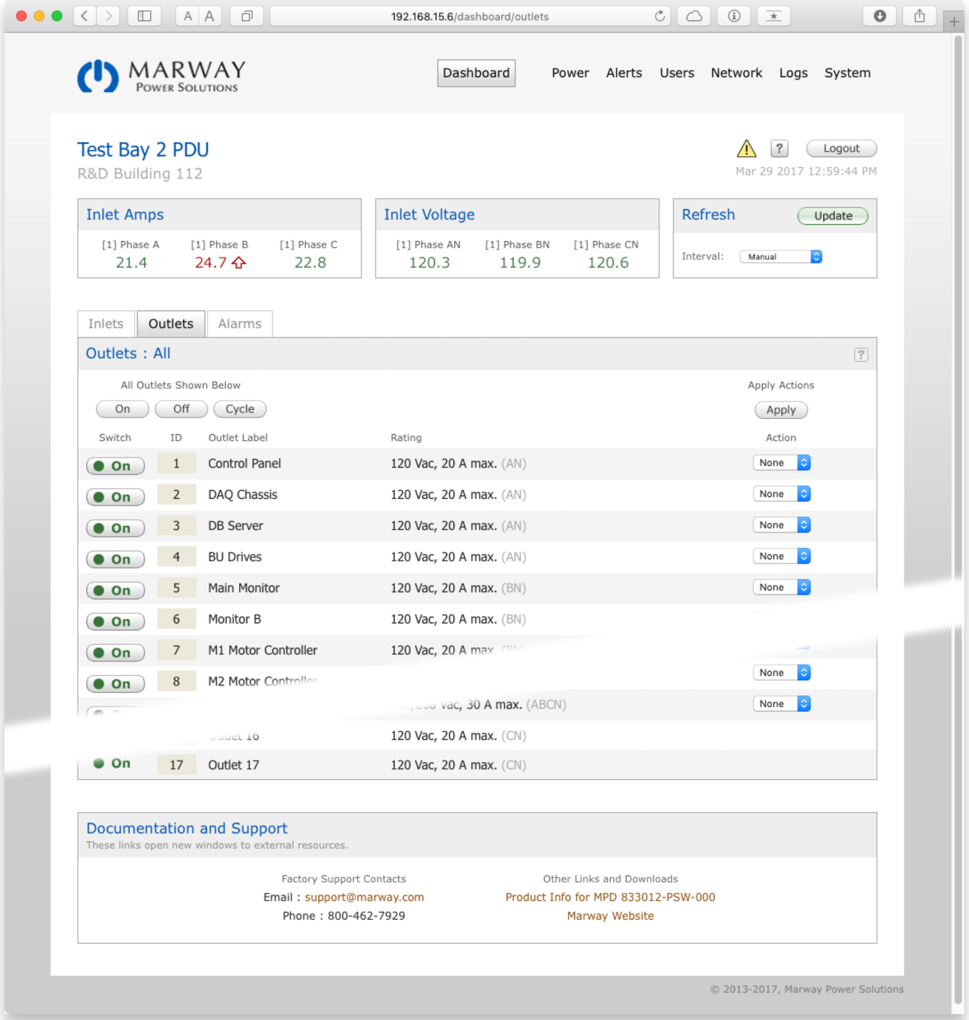
The first two data panels show the load on each phase, and the voltage of each phase. The number in the brackets [1] show which power inlet the phases are from. On most units, there will be only one inlet.

Auto Refresh

The third panel along the top provides controls to have the page auto refresh manually, or automatically on a time interval selected from the popup menu.

Outlets Tab

The large tabbed panel provides the detailed monitoring of power objects and alarms. The Outlets tab is the default view when the Dashboard is selected from the main menu at the top of the page. This dashboard shows an Optima 833 Series 3U unit with 17 outlets.



Test Bay 2 PDU

R&D Building 112

Mar 29 2017 12:59:44 PM

Inlet Amps

[1] Phase A	[1] Phase B	[1] Phase C
21.4	24.7 ⇧	22.8

Inlet Voltage

[1] Phase AN	[1] Phase BN	[1] Phase CN
120.3	119.9	120.6

Refresh

Interval: Manual

Outlets : All

All Outlets Shown Below

Apply Actions

Switch	ID	Outlet Label	Rating	Action
● On	1	Control Panel	120 Vac, 20 A max. (AN)	None
● On	2	DAQ Chassis	120 Vac, 20 A max. (AN)	None
● On	3	DB Server	120 Vac, 20 A max. (AN)	None
● On	4	BU Drives	120 Vac, 20 A max. (AN)	None
● On	5	Main Monitor	120 Vac, 20 A max. (BN)	None
● On	6	Monitor B	120 Vac, 20 A max. (BN)	
● On	7	M1 Motor Controller	120 Vac, 20 A max.	
● On	8	M2 Motor Controller		None
			Vac, 30 A max. (ABCN)	None
		Outlet 16	120 Vac, 20 A max. (CN)	
● On	17	Outlet 17	120 Vac, 20 A max. (CN)	

Documentation and Support

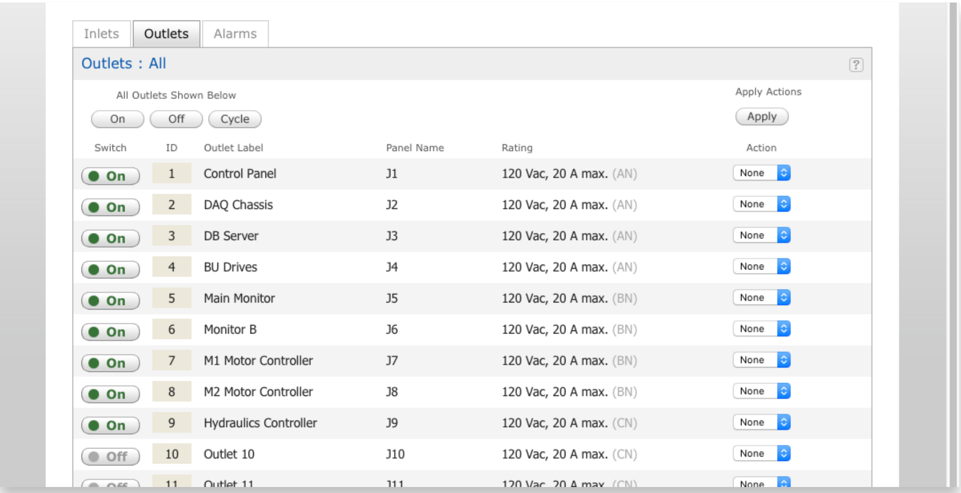
These links open new windows to external resources.

Factory Support Contacts
Email : support@marway.com
Phone : 800-462-7929

Other Links and Downloads
Product Info for MPD 833012-PSW-000
Marway Website

© 2013-2017, Marway Power Solutions

Each outlet has it's own on/off button, a user-defined label, and a couple informational columns. The Panel Name indicates what the outlet label is on the PDU chassis. The Rating provides basic capacity ratings and which phase the outlet is connected to. The last column is a popup menu allowing multiple outlets to be switched at the same time using the Apply button.



Switch	ID	Outlet Label	Panel Name	Rating	Action
● On	1	Control Panel	J1	120 Vac, 20 A max. (AN)	None
● On	2	DAQ Chassis	J2	120 Vac, 20 A max. (AN)	None
● On	3	DB Server	J3	120 Vac, 20 A max. (AN)	None
● On	4	BU Drives	J4	120 Vac, 20 A max. (AN)	None
● On	5	Main Monitor	J5	120 Vac, 20 A max. (BN)	None
● On	6	Monitor B	J6	120 Vac, 20 A max. (BN)	None
● On	7	M1 Motor Controller	J7	120 Vac, 20 A max. (BN)	None
● On	8	M2 Motor Controller	J8	120 Vac, 20 A max. (BN)	None
● On	9	Hydraulics Controller	J9	120 Vac, 20 A max. (CN)	None
● Off	10	Outlet 10	J10	120 Vac, 20 A max. (CN)	None
● Off	11	Outlet 11	J11	120 Vac, 20 A max. (CN)	None

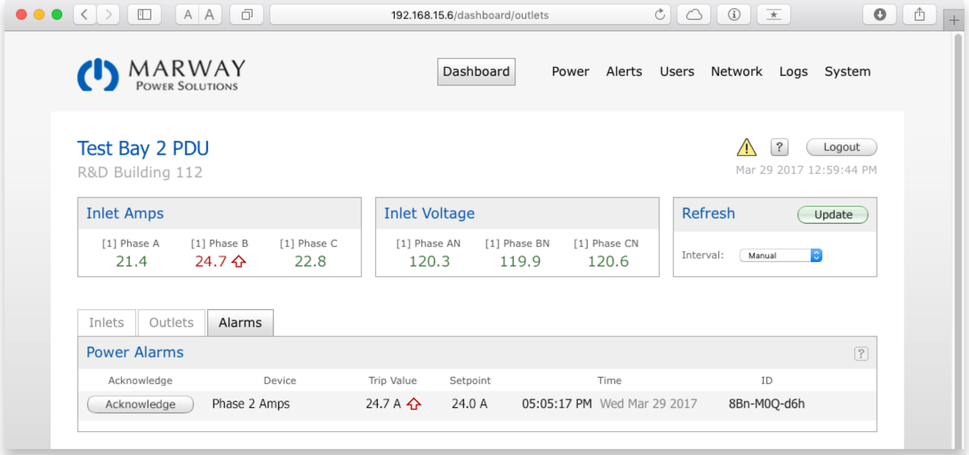
Inlets Tabs

The Inlets tab will be available for power monitored systems, and provides a detailed view of power data for each phase. This will include all data collected by the power-monitoring sensors which is not visible at the top of the Dashboard.

Alarms Tab

If phases are equipped with either current or power monitoring, each phase has alarm setpoints which can be used to trigger indicators and user notifications if current values (or voltage if measured) get outside desired limits. The Alarms tab provides a list of active alarms, and when they were first tripped. The Alerts main menu tab is used to define whether users receive notifications of these alarms by email and/or by SMS (text message), and whether they are to be broadcast as SNMP traps/notifications.

If there is at least one alarm, the caution icon in the upper right of the page will change from a dimmed gray to a bold yellow. This will be visible on every web page. Clicking this alarm indicator from any web page will immediately open the Dashboard > Alarms tab.



Test Bay 2 PDU
R&D Building 112

Mar 29 2017 12:59:44 PM

Inlet Amps

[1] Phase A	[1] Phase B	[1] Phase C
21.4	24.7 ⇧	22.8

Inlet Voltage

[1] Phase AN	[1] Phase BN	[1] Phase CN
120.3	119.9	120.6

Refresh

Interval: Manual

Power Alarms

Acknowledge	Device	Trip Value	Setpoint	Time	ID
Acknowledge	Phase 2 Amps	24.7 A ⇧	24.0 A	05:05:17 PM Wed Mar 29 2017	8Bn-M0Q-d6h

Power Settings

The main menu Power tab displays the adjustable settings for all power devices. Each device has slightly different settings.

Device Labels

Outlets and Inlets can have user-defined labels to better identify what each device is connected to, or user for. If the user does not enter labels, or sets one empty, the system will substitute a default label such as Outlet 3, Inlet 1, etc.

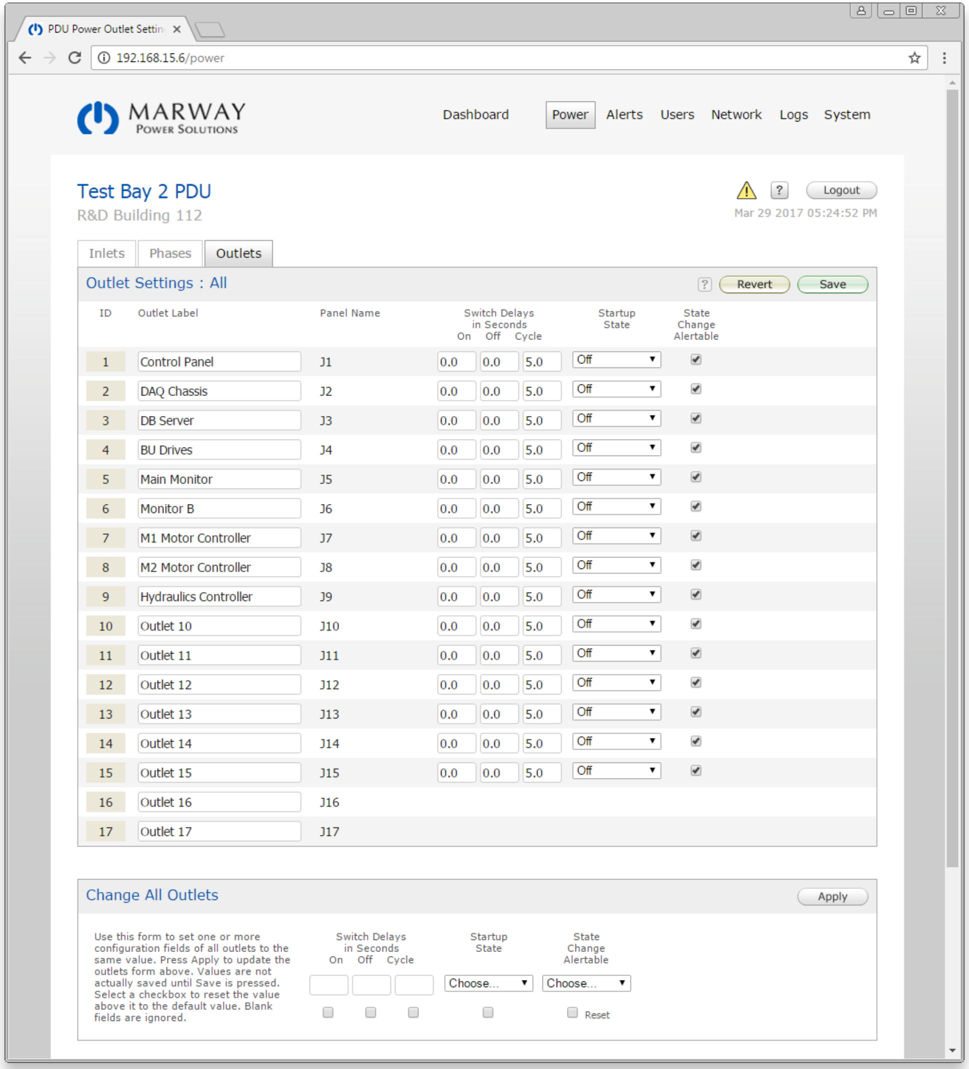
Phases do not have user labels. Their labels are fixed with LN in single-phase systems, and AN, BN, and CN in three-phase wye systems. Single-letter labels are used to identify current phases, two-letter labels are use for voltage phases.

Outlet Options

Each outlet has delay timing option. When the PDU is powered, the On Delay is applied before the outlet is enabled. If several outlets are enabled with the Dashboard outlet Action menus, the On Delay timing is also applied. Whenever a single outlet is turned on or off, the action is immediate.

When the PDU is powered up, each outlet can be set to a specific Startup State of on, off, or whatever the last known state was before the PDU was powered down. Each time an outlet is switched on or off, the state of the outlet is saved. That value is used to restore the "last known" state.

Each outlet can also be configured to trigger an alert or not. In some applications it may be extremely unusual for an outlet to be switched, and if one is switched, users may want an alert. The State Change Alertable setting in combination with configurations on the Alerts page can create flexible alert rules applied to all outlets, or only specific ones.



Test Bay 2 PDU

R&D Building 112

Mar 29 2017 05:24:52 PM

Outlet Settings : All

ID	Outlet Label	Panel Name	Switch Delays in Seconds On	Off	Cycle	Startup State	State Change Alertable
1	Control Panel	J1	0.0	0.0	5.0	Off	☑
2	DAQ Chassis	J2	0.0	0.0	5.0	Off	☑
3	DB Server	J3	0.0	0.0	5.0	Off	☑
4	BU Drives	J4	0.0	0.0	5.0	Off	☑
5	Main Monitor	J5	0.0	0.0	5.0	Off	☑
6	Monitor B	J6	0.0	0.0	5.0	Off	☑
7	M1 Motor Controller	J7	0.0	0.0	5.0	Off	☑
8	M2 Motor Controller	J8	0.0	0.0	5.0	Off	☑
9	Hydraulics Controller	J9	0.0	0.0	5.0	Off	☑
10	Outlet 10	J10	0.0	0.0	5.0	Off	☑
11	Outlet 11	J11	0.0	0.0	5.0	Off	☑
12	Outlet 12	J12	0.0	0.0	5.0	Off	☑
13	Outlet 13	J13	0.0	0.0	5.0	Off	☑
14	Outlet 14	J14	0.0	0.0	5.0	Off	☑
15	Outlet 15	J15	0.0	0.0	5.0	Off	☑
16	Outlet 16	J16					
17	Outlet 17	J17					

Change All Outlets

Use this form to set one or more configuration fields of all outlets to the same value. Press Apply to update the outlets form above. Values are not actually saved until Save is pressed. Select a checkbox to reset the value above it to the default value. Blank fields are ignored.

Switch Delays in Seconds On	Off	Cycle	Startup State	State Change Alertable
			Choose...	Choose...
☐	☐	☐	☐	☐ Reset

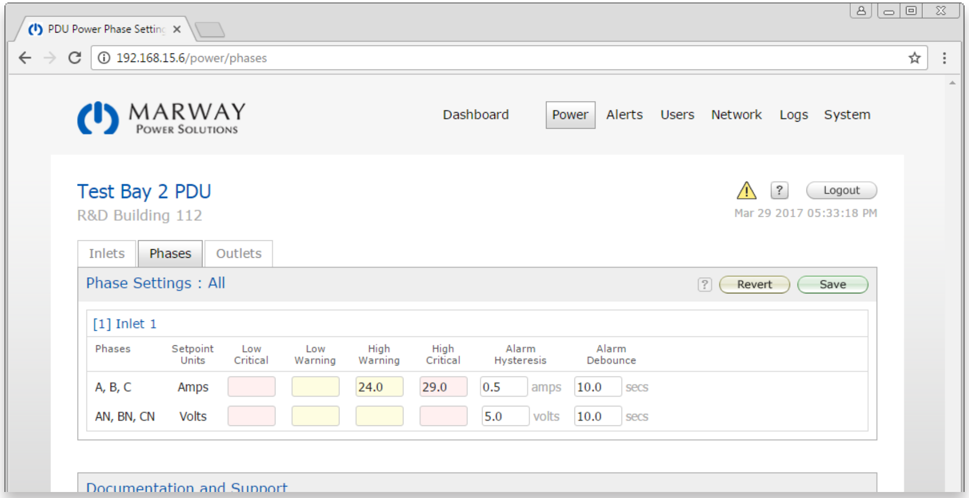
Setpoints

A setpoint is a user-adjustable value which is constantly compared to an actual measured data value. So, a setpoint value for a phase current would be compared to the measured current value each time that value is updated. If the measured value exceeds the setpoint value, we can generate a response to that, generically known as an event. The RCM software uses such events to show indicators on the `Dashboard`, send alerts to users, and generate other responses.

Phases may have setpoints for amps and/or volts (depending on the model and what sensor hardware is present).

Each setpoint has four trigger values and two additional values named hysteresis and debounce. A trigger value is the user's setting value which is compared to the measured value. The four triggers are known as Low Critical, Low Warning, High Warning, and High Critical. Each trigger can be disabled or have a value. Not all four must have values. Only the ones which you're interested in having generate an event need to have values.

Hysteresis and debounce help to define exactly how the trigger behaves when a measured value rises through or falls through the trigger value. For example, if a trigger exists for 1.5 amps, and the measured value oscillates between 1.48 and 1.52, you probably don't want the setpoint event to be repeatedly triggered—perhaps sending an email every time. Helping to prevent those cases are what hysteresis and debounce are for.



Test Bay 2 PDU
R&D Building 112

Mar 29 2017 05:33:18 PM

Phase Settings : All

[1] Inlet 1

Phases	Setpoint Units	Low Critical	Low Warning	High Warning	High Critical	Alarm Hysteresis	Alarm Debounce
A, B, C	Amps			24.0	29.0	0.5 amps	10.0 secs
AN, BN, CN	Volts					5.0 volts	10.0 secs

Alert Settings

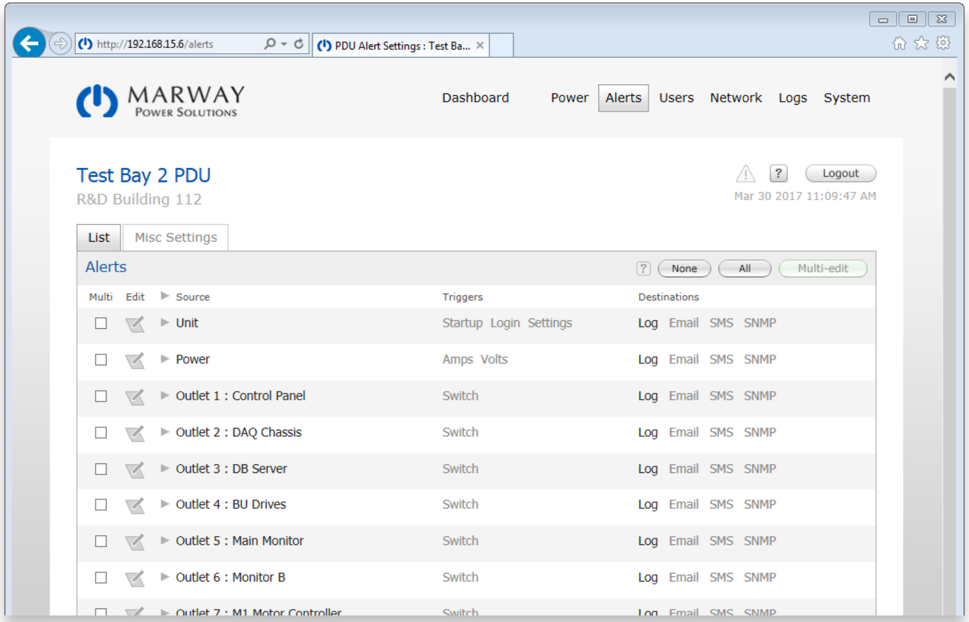
This page is used to define event alerts—the optional email, SMS, and SNMP notifications which can be sent when an event occurs.

Each alert can be defined individually. Selecting the Edit icon will open a form to edit that one alert.

Alerts can also be edited in groups. Select the Multi checkbox of several events and click the Multi-edit button. The All button can be used as a shortcut.

The alert editor lists the alerts which will be changed, and presents the settings options. All events are automatically logged*, so the user interface displays the Log as a preselected default. Alerts can be sent by email, SMS, and SNMP. To send emails and/or SMS alerts, select which users are to receive the alert messages.

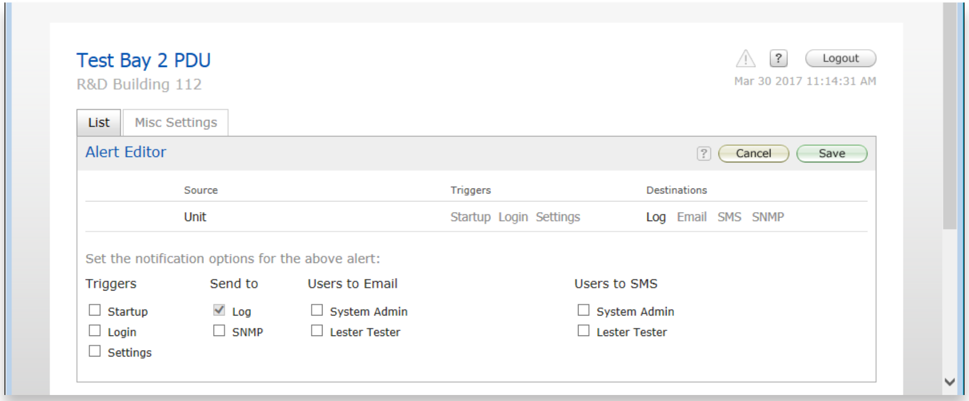
* Switch events triggered by the web UI are always logged. However, switch events triggered by CLI, SNMP, and REST can be disabled to prevent repeated, automated switching from filling the logs. These preferences are configured in the Log settings.



Test Bay 2 PDU
R&D Building 112
Mar 30 2017 11:09:47 AM

Alerts

Multi	Edit	Source	Triggers	Destinations
□		▶ Unit	Startup Login Settings	Log Email SMS SNMP
□		▶ Power	Amps Volts	Log Email SMS SNMP
□		▶ Outlet 1 : Control Panel	Switch	Log Email SMS SNMP
□		▶ Outlet 2 : DAQ Chassis	Switch	Log Email SMS SNMP
□		▶ Outlet 3 : DB Server	Switch	Log Email SMS SNMP
□		▶ Outlet 4 : BU Drives	Switch	Log Email SMS SNMP
□		▶ Outlet 5 : Main Monitor	Switch	Log Email SMS SNMP
□		▶ Outlet 6 : Monitor B	Switch	Log Email SMS SNMP
□		▶ Outlet 7 : M1 Motor Controller	Switch	Log Email SMS SNMP



Test Bay 2 PDU
R&D Building 112
Mar 30 2017 11:14:31 AM

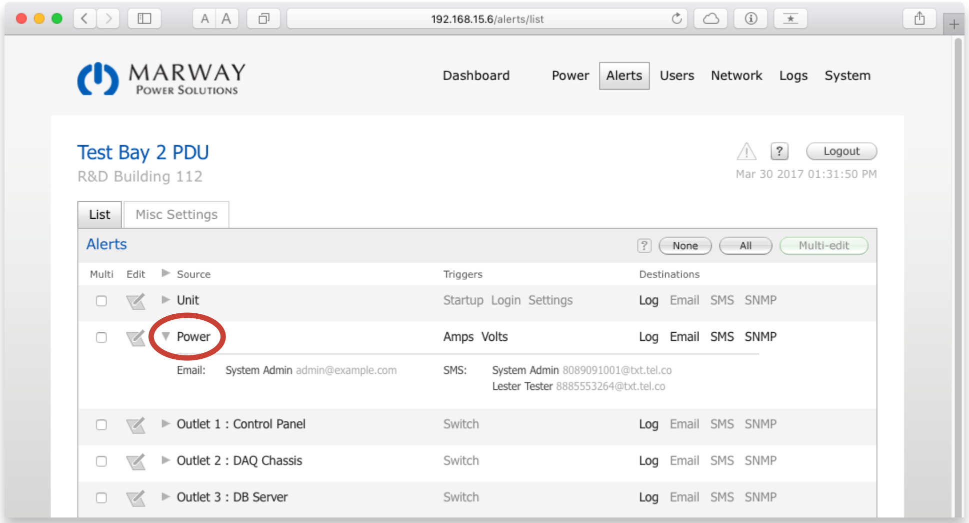
Alert Editor

Source	Triggers	Destinations
Unit	Startup Login Settings	Log Email SMS SNMP

Set the notification options for the above alert:

Triggers	Send to	Users to Email	Users to SMS
□ Startup	☑ Log	□ System Admin	□ System Admin
□ Login	□ SNMP	□ Lester Tester	□ Lester Tester
□ Settings			

Back in the alerts list, the reveal triangles can be selected to show which users are getting which alerts. Select the triangle in the list header to reveal alert details for all events. Notice also that when an event is configured to send alerts, the labels in the Destination column will change from being inactive (dimmed) to active (not dimmed).



Test Bay 2 PDU
R&D Building 112
Mar 30 2017 01:31:50 PM

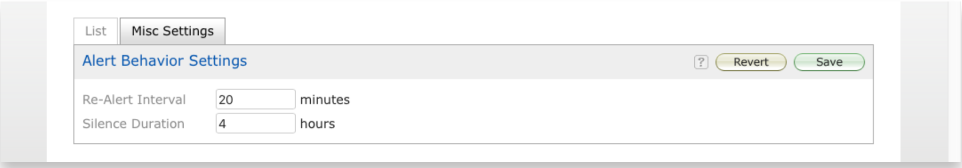
Alerts

Multi	Edit	▶ Source	Triggers	Destinations
□		▶ Unit	Startup Login Settings	Log Email SMS SNMP
□		▼ Power	Amps Volts	Log Email SMS SNMP
		Email: System Admin admin@example.com	SMS: System Admin 8089091001@txt.tel.co Lester Tester 8885553264@txt.tel.co	
□		▶ Outlet 1 : Control Panel	Switch	Log Email SMS SNMP
□		▶ Outlet 2 : DAQ Chassis	Switch	Log Email SMS SNMP
□		▶ Outlet 3 : DB Server	Switch	Log Email SMS SNMP

Notification Behavior Settings

The Misc Settings tab presents two options: Re-Alert Interval and Silence Duration.

When an alarm type event (essentially, a setpoint) is triggered, and is set to send an email, that email is sent immediately. How do we know the user got the email? Each email and SMS message includes a web link for the user to acknowledge the alert. The user needs to open that link to inform the RCM software that the message was received.



Alert Behavior Settings

Re-Alert Interval 20 minutes
Silence Duration 4 hours

Before that acknowledgment happens, there's some possible problem scenarios including the user may not get the email due to some technical problem. To help work around that and other scenarios, the alert system will repeat the email message until it is acknowledged. How often should it repeat? That's what the Re-Alert Interval is. As shown on the right, the email (and/or SMS) would be repeated every 20 minutes.

When an alert is acknowledged, that will stop the re-alerts. So, what if the user acknowledges the message and is distracted or prevented from addressing the problem? When do we "worry" that an alarm which continues to persist has been forgotten about? That's what the Silence Duration is for. An acknowledgment silences the re-alert for that period of time, after which the re-alerts will start up again.

SNMP traps/notifications are sent one time only. It is assumed the SNMP Manager will be handling the details of keeping people and/or systems informed. Part of the SNMP message will be the RCM acknowledge ID.

User Settings

There are several features packed into the user settings. Only the highlights will be covered here. Selecting a user from the list, or starting a new user, presents the user editor form. That form has four main sections (five when editing an existing user).

Authentication

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

Access Methods

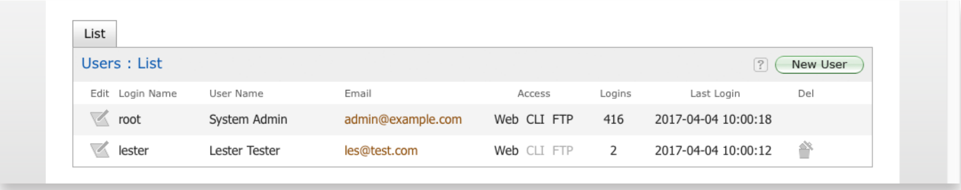
The Access Methods section determines whether the user can access the system via the web UI, the CLI, and the file system. If the none of the options are selected, this effectively disables a user record without having to delete it.

Personal Profile

This section is optional. It allows an administrator to keep minimal contact information. The Company and Job Role field can be useful to identify third-party vendors who may be allowed access perhaps for support. The email address and SMS address (which is the mobile phone number in an email address format specific to the phone's carrier) are, of course, important if the user wants to receive event alerts.

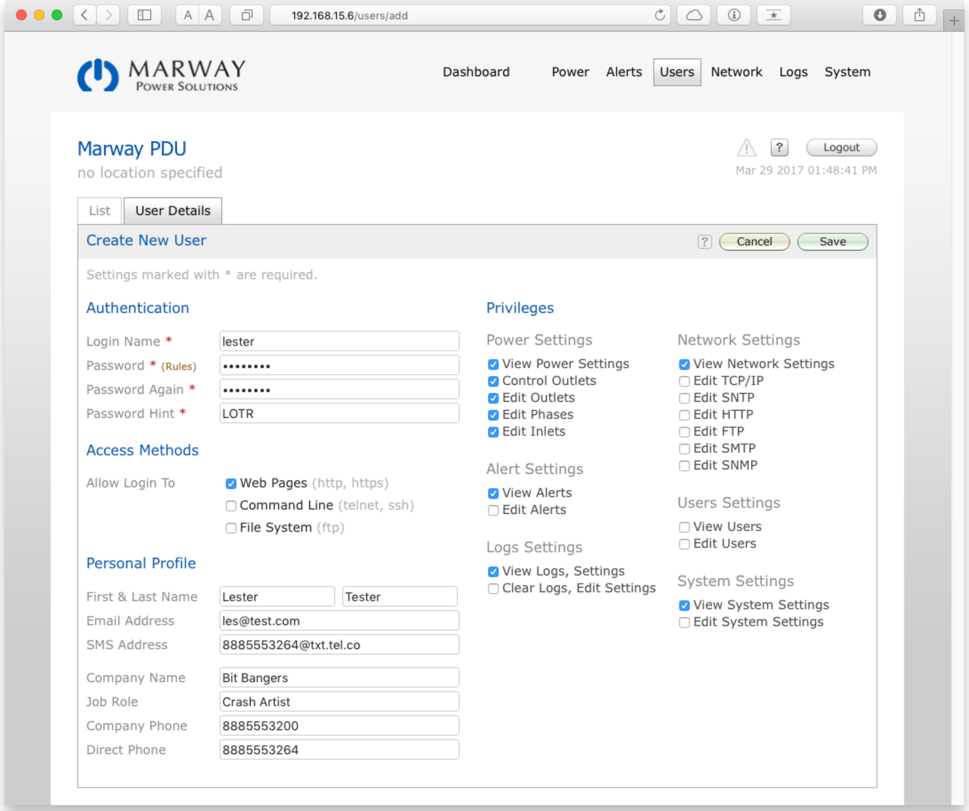
Permissions

The RCM software does not assign permissions based on user groups. Rather, each user is assigned distinct permissions to allow greater flexibility.



Users : List

Edit	Login Name	User Name	Email	Access	Logins	Last Login	Del
	root	System Admin	admin@example.com	Web CLI FTP	416	2017-04-04 10:00:18	
	lester	Lester Tester	les@test.com	Web CLI FTP	2	2017-04-04 10:00:12	



Marway PDU

no location specified

Mar 29 2017 01:48:41 PM

Create New User

Settings marked with * are required.

Authentication

Login Name * lester

Password * (Rules) ••••••••

Password Again * ••••••••

Password Hint * LOTR

Access Methods

Allow Login To
☑ Web Pages (http, https)
☐ Command Line (telnet, ssh)
☐ File System (ftp)

Personal Profile

First & Last Name Lester Tester

Email Address les@test.com

SMS Address 8885553264@txt.tel.co

Company Name Bit Bangers

Job Role Crash Artist

Company Phone 8885553200

Direct Phone 8885553264

Privileges

Power Settings
☑ View Power Settings
☑ Control Outlets
☑ Edit Outlets
☑ Edit Phases
☑ Edit Inlets

Alert Settings
☑ View Alerts
☐ Edit Alerts

Logs Settings
☑ View Logs, Settings
☐ Clear Logs, Edit Settings

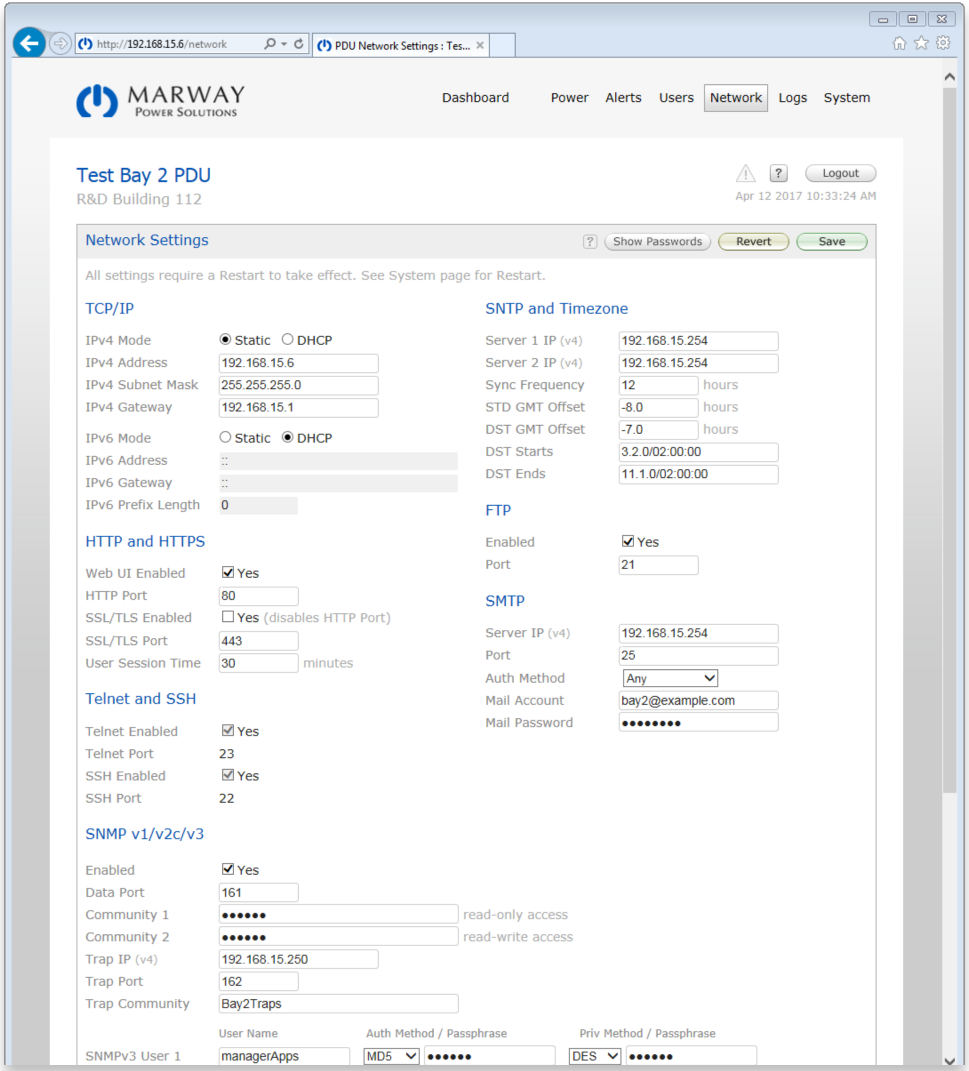
Network Settings
☑ View Network Settings
☐ Edit TCP/IP
☐ Edit SNTP
☐ Edit HTTP
☐ Edit FTP
☐ Edit SMTP
☐ Edit SNMP

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.



Test Bay 2 PDU

R&D Building 112

Apr 12 2017 10:33:24 AM

Network Settings

All settings require a Restart to take effect. See System page for Restart.

TCP/IP

IPv4 Mode	● Static ○ DHCP
IPv4 Address	192.168.15.6
IPv4 Subnet Mask	255.255.255.0
IPv4 Gateway	192.168.15.1
IPv6 Mode	○ Static ● DHCP
IPv6 Address	::
IPv6 Gateway	::
IPv6 Prefix Length	0

HTTP and HTTPS

Web UI Enabled	☑ Yes
HTTP Port	80
SSL/TLS Enabled	□ Yes (disables HTTP Port)
SSL/TLS Port	443
User Session Time	30 minutes

Telnet and SSH

Telnet Enabled	☑ Yes
Telnet Port	23
SSH Enabled	☑ Yes
SSH Port	22

SNMP v1/v2c/v3

Enabled	☑ Yes
Data Port	161
Community 1	•••••• read-only access
Community 2	•••••• read-write access
Trap IP (v4)	192.168.15.250
Trap Port	162
Trap Community	Bay2Traps

	User Name	Auth Method / Passphrase		Priv Method / Passphrase	
SNMPv3 User 1	managerApps	MD5	••••••	DES	••••••

SNTP and Timezone

Server 1 IP (v4)	192.168.15.254
Server 2 IP (v4)	192.168.15.254
Sync Frequency	12 hours
STD GMT Offset	-8.0 hours
DST GMT Offset	-7.0 hours
DST Starts	3.2.0/02:00:00
DST Ends	11.1.0/02:00:00

FTP

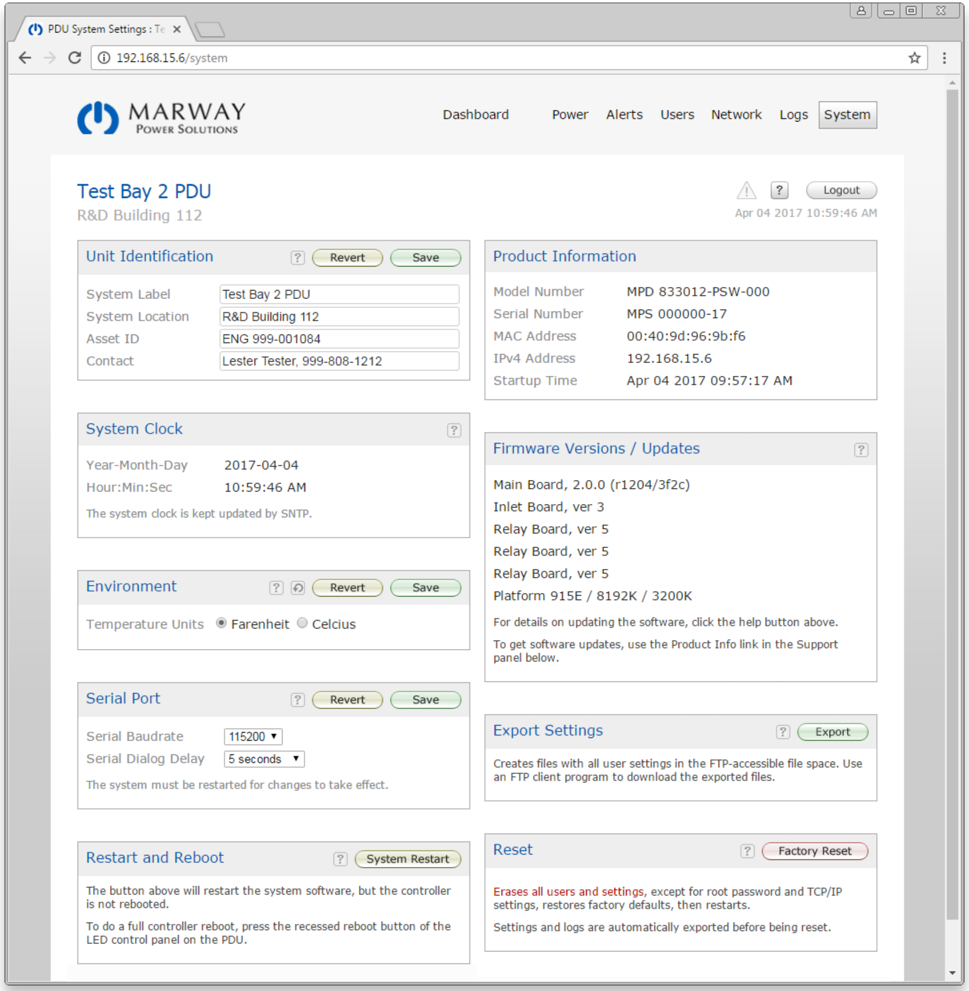
Enabled	☑ Yes
Port	21

SMTP

Server IP (v4)	192.168.15.254
Port	25
Auth Method	Any
Mail Account	bay2@example.com
Mail Password	••••••••

System Settings

The System page displays a number of unit-wide details about the product, as well as edit general purpose settings like the system label and location, the time, and a few others. All user settings can be exported from the System page to save those for documentation or even to import into other units.



Test Bay 2 PDU
R&D Building 112

Apr 04 2017 10:59:46 AM

Unit Identification

System Label	Test Bay 2 PDU
System Location	R&D Building 112
Asset ID	ENG 999-001084
Contact	Lester Tester, 999-808-1212

Product Information

Model Number	MPD 833012-PSW-000
Serial Number	MPS 000000-17
MAC Address	00:40:9d:96:9b:f6
IPv4 Address	192.168.15.6
Startup Time	Apr 04 2017 09:57:17 AM

System Clock

Year-Month-Day	2017-04-04
Hour:Min:Sec	10:59:46 AM

The system clock is kept updated by SNTP.

Firmware Versions / Updates

Main Board, 2.0.0 (r1204/3f2c)
Inlet Board, ver 3
Relay Board, ver 5
Relay Board, ver 5
Relay Board, ver 5
Platform 915E / 8192K / 3200K

For details on updating the software, click the help button above.

To get software updates, use the Product Info link in the Support panel below.

Environment

Temperature Units ● Farenheit ○ Celcius

Serial Port

Serial Baudrate	115200 ▾
Serial Dialog Delay	5 seconds ▾

The system must be restarted for changes to take effect.

Export Settings

Creates files with all user settings in the FTP-accessible file space. Use an FTP client program to download the exported files.

Restart and Reboot

The button above will restart the system software, but the controller is not rebooted.

To do a full controller reboot, press the recessed reboot button of the LED control panel on the PDU.

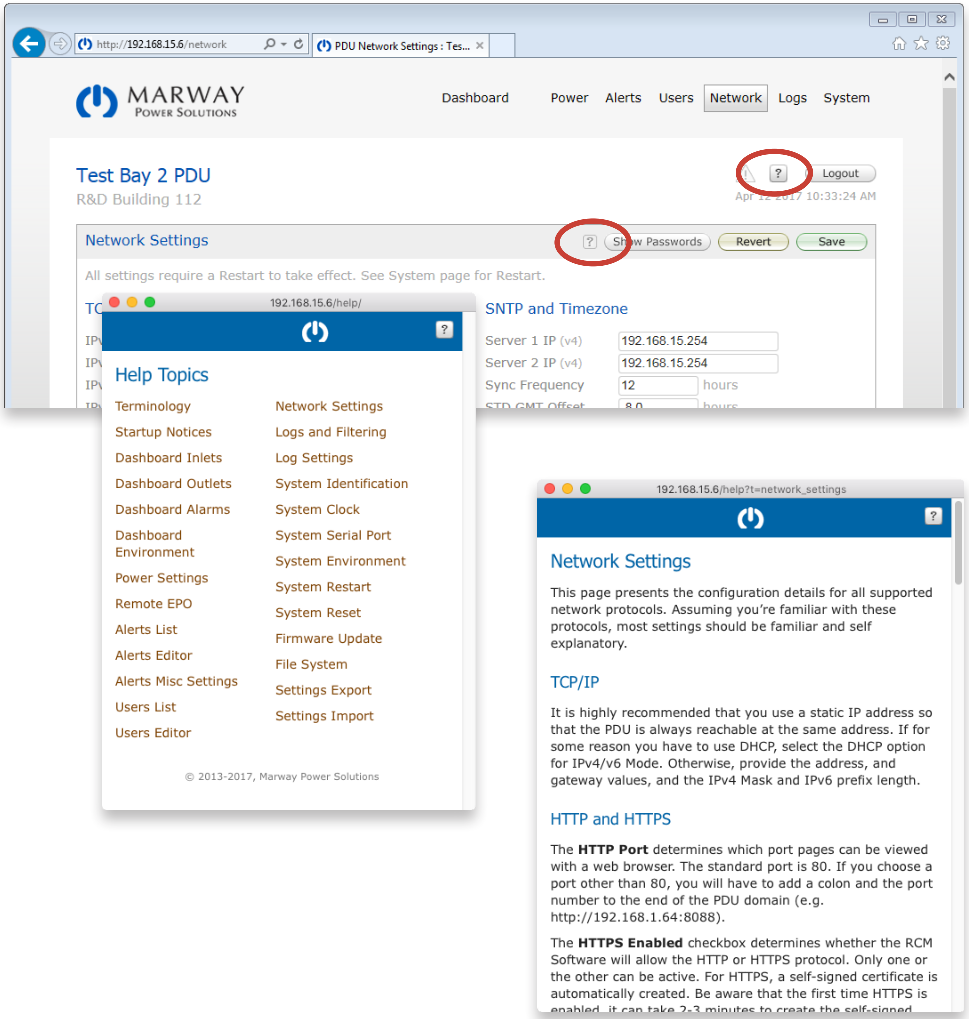
Reset

Erases all users and settings, except for root password and TCP/IP settings, restores factory defaults, then restarts.

Settings and logs are automatically exported before being reset.

Built-in Web Help

In the header of every web page is a Help button. This will open the built-in help system for the RCM software and display a topical index to explore further. Additionally, many of the individual panels on the web pages, such as the Network Settings panel shown to the right, will have a help button. This button will open directly to the help topic for that panel.



Test Bay 2 PDU

R&D Building 112

Apr 12 2017 10:33:24 AM

Network Settings

All settings require a Restart to take effect. See System page for Restart.

SNTP and Timezone

Server 1 IP (v4) 192.168.15.254

Server 2 IP (v4) 192.168.15.254

Sync Frequency 12 hours

STD GMT Offset -8.0 hours

Help Topics

- Terminology
- Startup Notices
- Dashboard Inlets
- Dashboard Outlets
- Dashboard Alarms
- Dashboard Environment
- Power Settings
- Remote EPO
- Alerts List
- Alerts Editor
- Alerts Misc Settings
- Users List
- Users Editor
- Network Settings
- Logs and Filtering
- Log Settings
- System Identification
- System Clock
- System Serial Port
- System Environment
- System Restart
- System Reset
- Firmware Update
- File System
- Settings Export
- Settings Import

© 2013-2017, Marway Power Solutions

Network Settings

This page presents the configuration details for all supported network protocols. Assuming you're familiar with these protocols, most settings should be familiar and self explanatory.

TCP/IP

It is highly recommended that you use a static IP address so that the PDU is always reachable at the same address. If for some reason you have to use DHCP, select the DHCP option for IPv4/v6 Mode. Otherwise, provide the address, and gateway values, and the IPv4 Mask and IPv6 prefix length.

HTTP and HTTPS

The **HTTP Port** determines which port pages can be viewed with a web browser. The standard port is 80. If you choose a port other than 80, you will have to add a colon and the port number to the end of the PDU domain (e.g. http://192.168.1.64:8088).

The **HTTPS Enabled** checkbox determines whether the RCM Software will allow the HTTP or HTTPS protocol. Only one or the other can be active. For HTTPS, a self-signed certificate is automatically created. Be aware that the first time HTTPS is enabled, it can take 2-3 minutes to create the self-signed

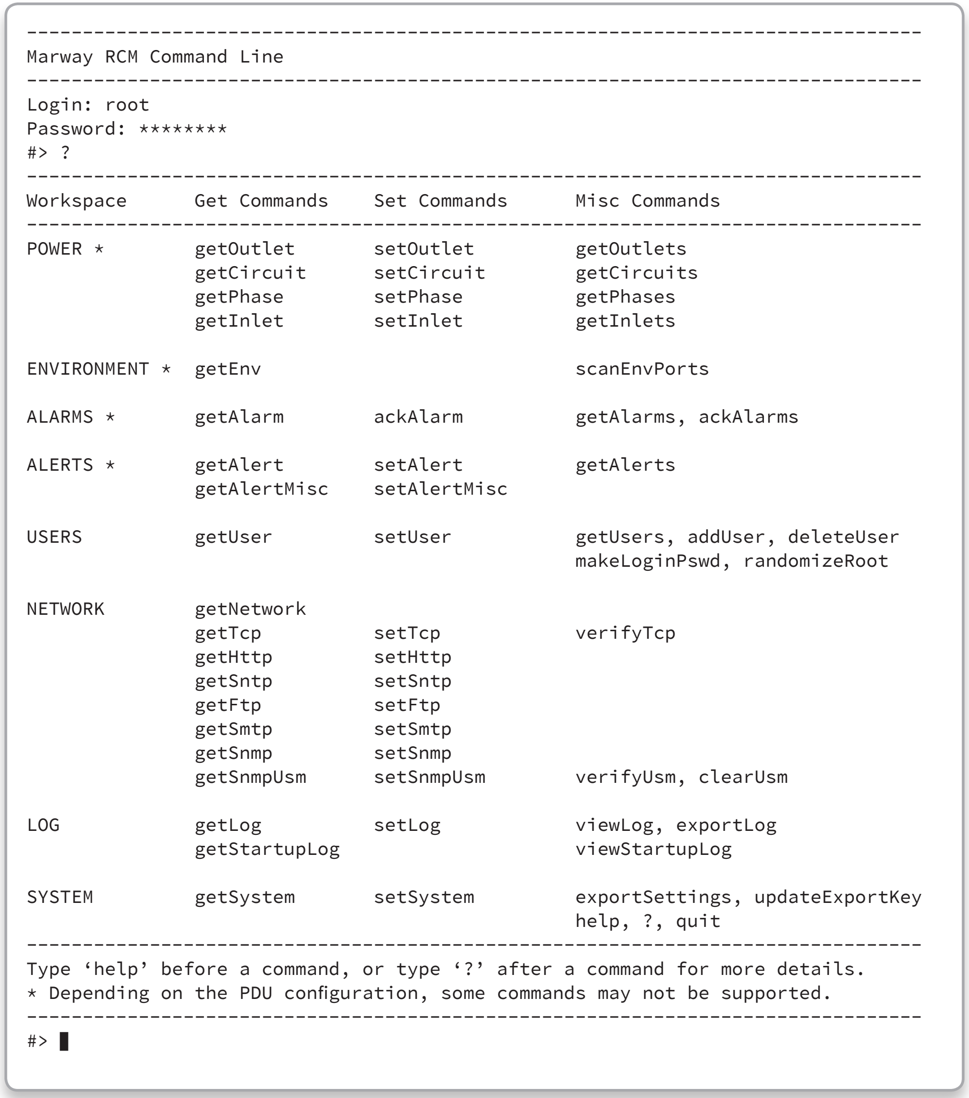
Command Line Interface Tour

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

- getOutlet / setOutlet
- getPhase / setPhase
- getAlarm / setAlarm
- getAlert / setAlert
- getUser / setUser and addUser / deleteUser
- getTcp / setTcp
- getHttp / setHttp
- getSntp / setSntp
- getSmtp / setSmtp
- getSnmp / setSnmp
- getLog / getStartupLog
- getSystem / setSystem

Login

The login task is fairly straight forward, for Telnet and SSH, enter a user login name at the prompt, and a password. Using the CLI from the serial port does not require a login.



```
--------------------------------------------------------------------------------
Marway RCM Command Line
--------------------------------------------------------------------------------
Login: root
Password: ********
#> ?
--------------------------------------------------------------------------------
Workspace      Get Commands    Set Commands      Misc Commands
--------------------------------------------------------------------------------
POWER *        getOutlet       setOutlet         getOutlets
               getCircuit      setCircuit        getCircuits
               getPhase        setPhase          getPhases
               getInlet        setInlet          getInlets

ENVIRONMENT *  getEnv                            scanEnvPorts

ALARMS *       getAlarm        ackAlarm          getAlarms, ackAlarms

ALERTS *       getAlert        setAlert          getAlerts
               getAlertMisc    setAlertMisc

USERS          getUser         setUser           getUsers, addUser, deleteUser
                                                 makeLoginPswd, randomizeRoot

NETWORK        getNetwork
               getTcp          setTcp            verifyTcp
               getHttp         setHttp
               getSntp         setSntp
               getFtp          setFtp
               getSmtp         setSmtp
               getSnmp         setSnmp
               getSnmpUsm      setSnmpUsm        verifyUsm, clearUsm

LOG            getLog          setLog            viewLog, exportLog
               getStartupLog                     viewStartupLog

SYSTEM         getSystem       setSystem         exportSettings, updateExportKey
                                                 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.
--------------------------------------------------------------------------------
#> █
```

Built-in CLI 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`
- `getPhase 1 high_critical_amps`
- `getUser lester email`
- `getSystem location`

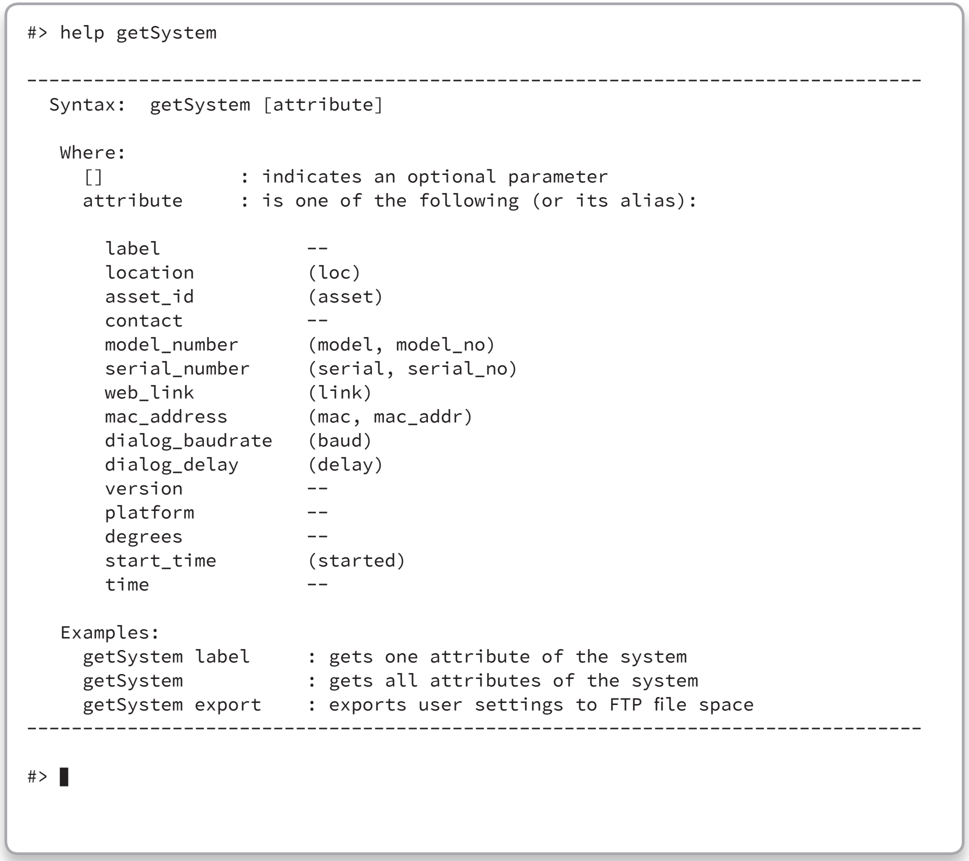
These are typical set commands:

- `setOutlet 1 switch on`
- `setPhase 1 high_critical_amps 14.5`
- `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`, `setPhase`, etc. The *instanceID* is going to be the numerical ID of a power device such as an



```
#> help getSystem

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

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

       label             --
       location          (loc)
       asset_id          (asset)
       contact           --
       model_number      (model, model_no)
       serial_number     (serial, serial_no)
       web_link          (link)
       mac_address       (mac, mac_addr)
       dialog_baudrate   (baud)
       dialog_delay      (delay)
       version           --
       platform          --
       degrees           --
       start_time        (started)
       time              --

   Examples:
     getSystem label       : gets one attribute of the system
     getSystem             : gets all attributes of the system
     getSystem export      : exports user settings to FTP file space
--------------------------------------------------------------------------------

#> █
```

outlet, phase, etc., 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.

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 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:

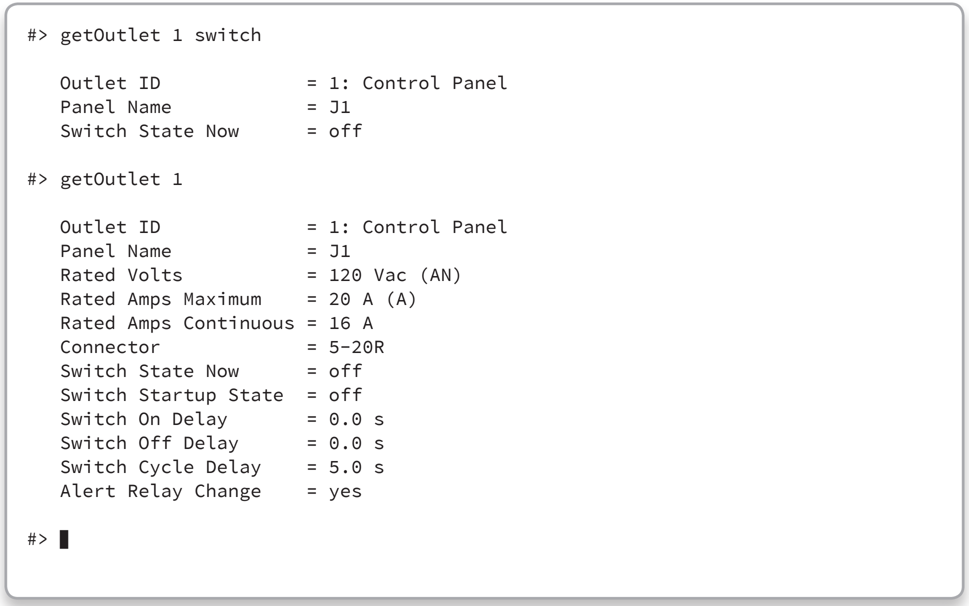
```
getOutlet 4
getUser lester
```

will return all attributes for one specific instance.

A get command used like these examples:

```
getOutlet
getUser
```

will return all attributes for all instances.

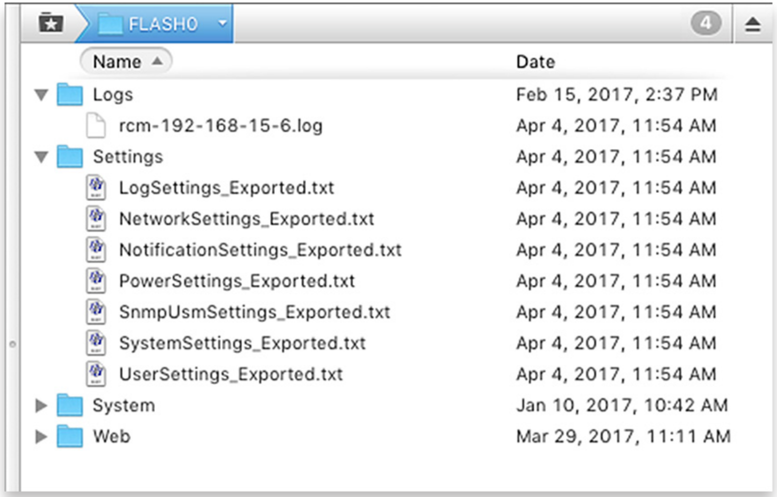


```
#> getOutlet 1 switch

   Outlet ID                = 1: Control Panel
   Panel Name               = J1
   Switch State Now         = off

#> getOutlet 1

   Outlet ID                = 1: Control Panel
   Panel Name               = J1
   Rated Volts              = 120 Vac (AN)
   Rated Amps Maximum       = 20 A (A)
   Rated Amps Continuous    = 16 A
   Connector                = 5-20R
   Switch State Now         = off
   Switch Startup State     = off
   Switch On Delay          = 0.0 s
   Switch Off Delay         = 0.0 s
   Switch Cycle Delay       = 5.0 s
   Alert Relay Change       = yes

#> ▌
```

FTP and the File System

An on-board file system is used to store exported settings and logs, SSL files, and web files. This file system is accessible through FTP.

Settings files can be exported then downloaded for archiving. They may even be edited and uploaded to another unit. This allows preconfiguration of many details (particularly for users and network settings) which can then be uploaded to multiple products.



Name	Date
Logs	Feb 15, 2017, 2:37 PM
rcm-192-168-15-6.log	Apr 4, 2017, 11:54 AM
Settings	Apr 4, 2017, 11:54 AM
LogSettings_Exported.txt	Apr 4, 2017, 11:54 AM
NetworkSettings_Exported.txt	Apr 4, 2017, 11:54 AM
NotificationSettings_Exported.txt	Apr 4, 2017, 11:54 AM
PowerSettings_Exported.txt	Apr 4, 2017, 11:54 AM
SnmpUsmSettings_Exported.txt	Apr 4, 2017, 11:54 AM
SystemSettings_Exported.txt	Apr 4, 2017, 11:54 AM
UserSettings_Exported.txt	Apr 4, 2017, 11:54 AM
System	Jan 10, 2017, 10:42 AM
Web	Mar 29, 2017, 11:11 AM

Local LED Display

The LED display and keypad (optional on some models) is used to display power values and switch states.

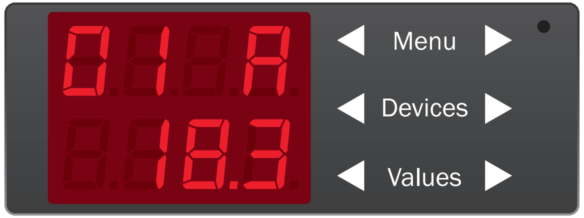
The display defaults to auto-scrolling through each phase's amps when there's no user interaction. Otherwise, the user can manually select any of the available power values by using the six pushbuttons on the keypad.

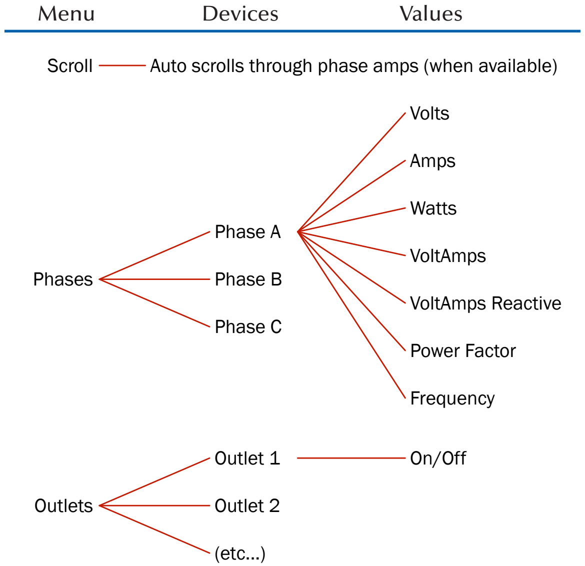
The Menu buttons cycle through the main display sections of Scroll, Phases, and Outlets. With either Phases or Outlets chosen, the display will show the first value of the first device such as Phase A Volts.

The Devices buttons cycle through the list of instances. So, if Phases was chosen, the display would start at Phase A Volts. A press of the Devices right arrow would update the display to Phase B Volts, then Phase C Volts, and back to Phase A Volts.

The Values buttons cycle through the values of the selected device. So, if Phases was chosen, the display would start at Phase A Volts. A press of the Values right arrow would update the display to Phase A Amps, then Phase A Watts, etc.

At this point, pressing the Devices right arrow would update the display from Phase A Watts to Phase B Watts, thus allowing easy scrolling through the same value for all device instances.





Scripting

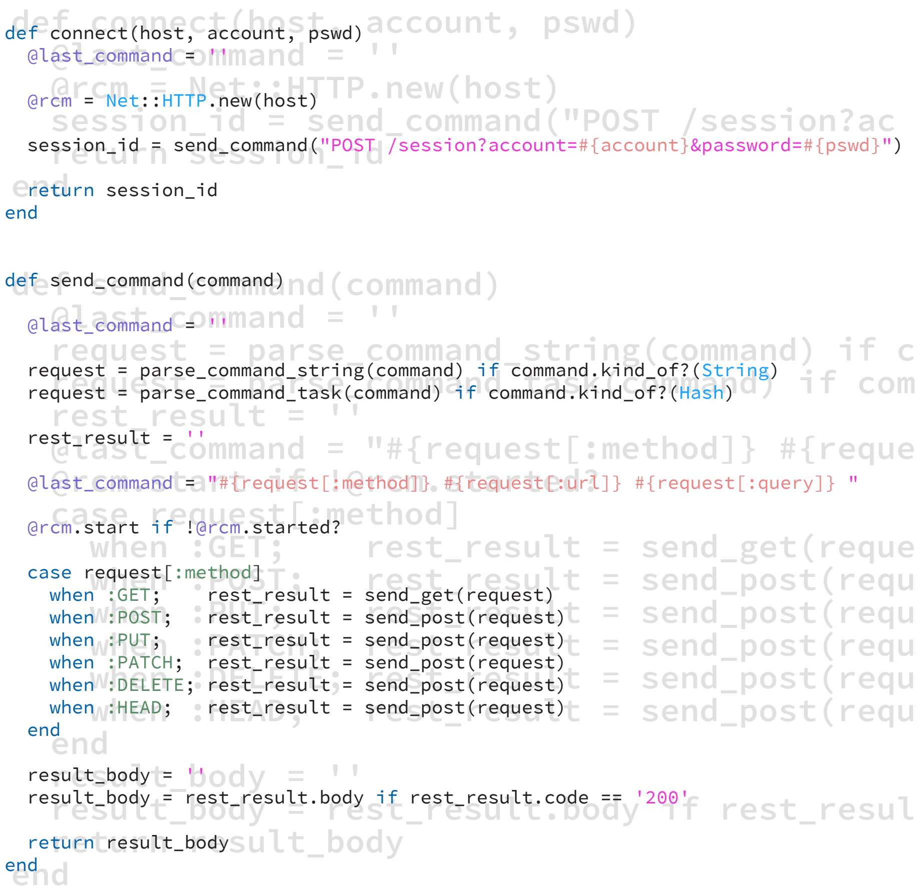
Automating data collection and outlet switching can be accomplished using three distinct techniques: use the SNMP interface, use the command line interface over Telnet or SSH, or use the RESTful API.

Of these, the RESTful API (commonly just called REST), is likely the newest technique to most people. Scripting with the REST API is similar to scripting the command line, but the commands and responses are more efficient as they're intended for machine to machine data exchange whereas the command line interface has a verbosity to it designed for human reading.

With the command line, your scripting language includes a built-in Telnet or SSH library. You create a connection, formulate a command as a string to send to the target device, and you likely get a string response which must be parsed to extract the piece of data you're wanting.

Using REST is similar except you use an HTTP library instead of the Telnet one, and you create HTTP messages. The response is also a string, but requires little or no parsing (other than type casting), because the reply is formatted as data only—there's no extra human-oriented text in the response.

The result of using REST is a more efficient dialog between the systems as each is using an interface designed for machine-to-machine data exchange.



```ruby
def connect(host, account, pswd)
  @last_command = ''

  @rcm = Net::HTTP.new(host)

  session_id = send_command("POST /session?account=#{account}&password=#{pswd}")

  return session_id
end


def send_command(command)

  @last_command = ''

  request = parse_command_string(command) if command.kind_of?(String)
  request = parse_command_task(command) if command.kind_of?(Hash)

  rest_result = ''

  @last_command = "#{request[:method]} #{request[:url]} #{request[:query]} "

  @rcm.start if !@rcm.started?

  case request[:method]
    when :GET;    rest_result = send_get(request)
    when :POST;   rest_result = send_post(request)
    when :PUT;    rest_result = send_post(request)
    when :PATCH;  rest_result = send_post(request)
    when :DELETE; rest_result = send_post(request)
    when :HEAD;   rest_result = send_post(request)
  end

  result_body = ''
  result_body = rest_result.body if rest_result.code == '200'

  return result_body
end
```

Optima 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 to these sections for an overview:

- For Setup instructions, refer to "Getting Started" on page 8.
- For an overview of the organization and features of the web browser interface refer to "CLI Setpoint Status Display" on page 51.
- For an overview of the command line interface refer to "Dashboard Data and Alarm Indication" on page 49.

Login

Full access to the software's features and settings is available using HTTP, HTTPS, Telnet, or SSH. Limited feature access is available through SNMP and the HTTP RESTful API. Additionally, access to the file system (not needed during normal operation) is available using FTP.

Each protocol has a unique login interface, but all use the same login name and password except for SNMP (SNMP uses it's own access pass names and codes). The login name is sometimes referred to as the user account name. Regardless of the label used, it refers to the PDU login credentials as entered in the User Settings form.

For discussion purposes, assume the PDU has been configured with an IP address of 192.168.15.6. We will further assume the predefined user root will be used for the examples.

DNS Names vs IP Addresses

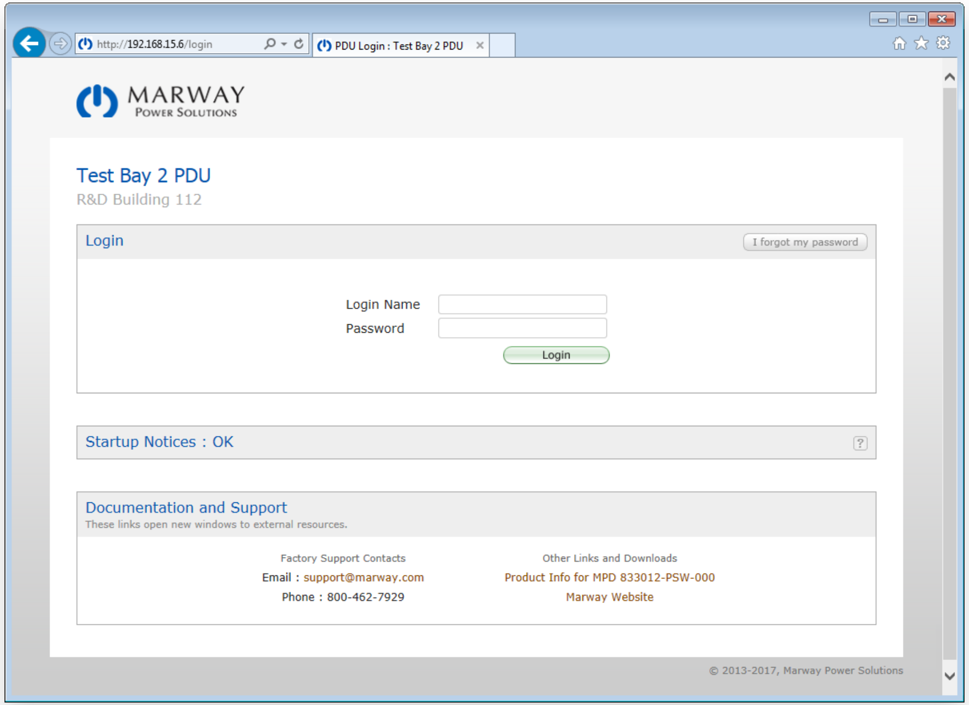
Your network administrator may have a domain name assigned to the PDU IP address, and you may have been instructed to use that name. It may be something like www.pduname.com or pdu.company.com to access a PDU across the Internet, or something like deptserverspdu.local to access the PDU on your local building network. These are DNS names, and can be used instead of the numeric IP address to connect to the PDU. However, in the PDU's own network settings, IP addresses must be used, not DNS names.

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://192.168.15.6. 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 login name and password. The login name is a unique name assigned to the user specifically for logging in. There is one default login name of 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.



Test Bay 2 PDU

R&D Building 112

Login

Login Name

Password

Startup Notices : OK

Documentation and Support

These links open new windows to external resources.

Factory Support Contacts

Email : support@marway.com

Phone : 800-462-7929

Other Links and Downloads

Product Info for MPD 833012-PSW-000

Marway Website

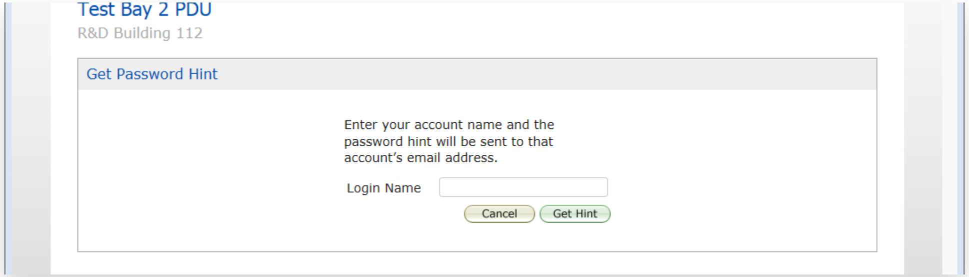
© 2013-2017, Marway Power Solutions

I forgot my password

The I forgot my password button opens a form where the user enters an email address. Each user setup includes a Password Hint field. This field accepts a short text entry intended to help the user remember what the password is, or how to find it. This hint is emailed to the user by the Get Password Hint form. The email address must match an existing user’s address.

If the user still cannot remember the password, another PDU user allowed to edit user settings will have to create a new password. This may be the root account holder, or another user. Refer to “Dashboard Data and Alarm Indication” on page 49 for details about user permissions.

If the root user password is forgotten, there are two possible ways to change it. First, if another user is allowed to edit users, that person can update the root password on the Users setting page. Otherwise, use the set up process described in “Dashboard Data and Alarm Indication” on page 49 to reset the root password.



Test Bay 2 PDU

R&D Building 112

Get Password Hint

Enter your account name and the
password hint will be sent to that
account’s email address.

Login Name

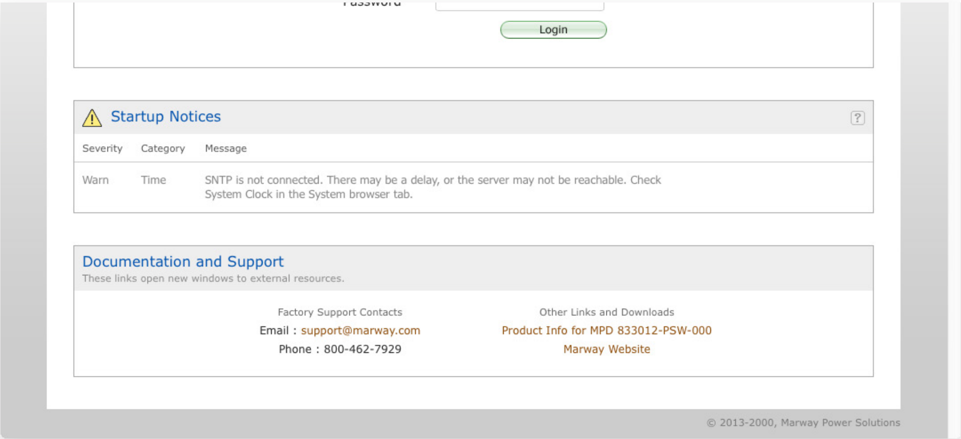
Startup Notices

This panel will show any errors encountered during the startup of the PDU. The sample login page above did not have any startup errors, and the panel title shows an "OK" status. The sample login page to the right shows one error.

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

Documentation and Support

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



⚠ Startup Notices

Severity	Category	Message
Warn	Time	SNTP is not connected. There may be a delay, or the server may not be reachable. Check System Clock in the System browser tab.

Documentation and Support

These links open new windows to external resources.

Factory Support Contacts	Other Links and Downloads
Email : support@marway.com	Product Info for MPD 833012-PSW-000
Phone : 800-462-7929	Marway Website

© 2013-2000, Marway Power Solutions

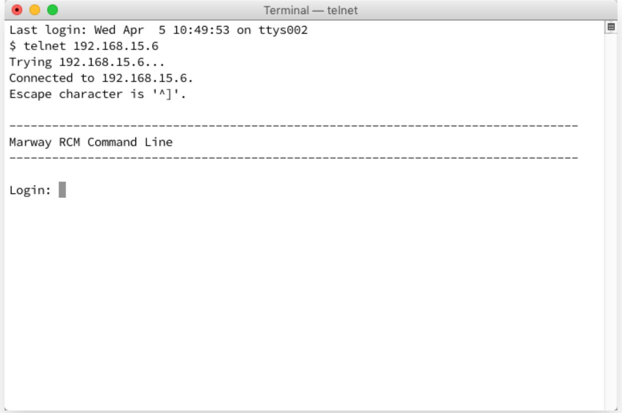
Telnet Login

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. For most *nix systems 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
```

The Windows Telnet client uses "open" instead of "telnet" to start a session. Once connected, you will be prompted for a Login (the login name) and a Password.



```
Last login: Wed Apr  5 10:49:53 on ttys002
$ telnet 192.168.15.6
Trying 192.168.15.6...
Connected to 192.168.15.6.
Escape character is '^]'.

--------------------------------------------------------------------------------
Marway RCM Command Line
--------------------------------------------------------------------------------

Login: 
```

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

```
ssh root@192.168.1.10
ssh lester@pdu3.mycompany.com
ssh jsmith@testbay2pdu.local
```

You will be prompted for the user password (the user name was already provided in the command above).

Five Strikes / Three Minutes

If a user fails to enter the correct password 5 times in a row, that account will be blocked for about 3 minutes. This means that even if on the 6th try, the password is entered correctly, the PDU will reject the login. The user must wait for 3 minutes before trying again.

This is done to prevent a certain type of attack where an automated system tries password guesses as quickly as possible. Being forced to wait 3 minutes every 5 tries means it would be an extremely long time before all possible guesses could be made.

The Difference Between Telnet and SSH

The short version is that SSH is more secure than Telnet. With Telnet, the information exchanged between your computer and the PDU is not encrypted. It is possible, under very 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 encrypted, 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 Power Data

Inlet Data

An Inlet is the combination of conductors providing power to the PDU. In RCM terminology, this comprises the current phases (individual conductors), and voltage phases (pairs of conductors).

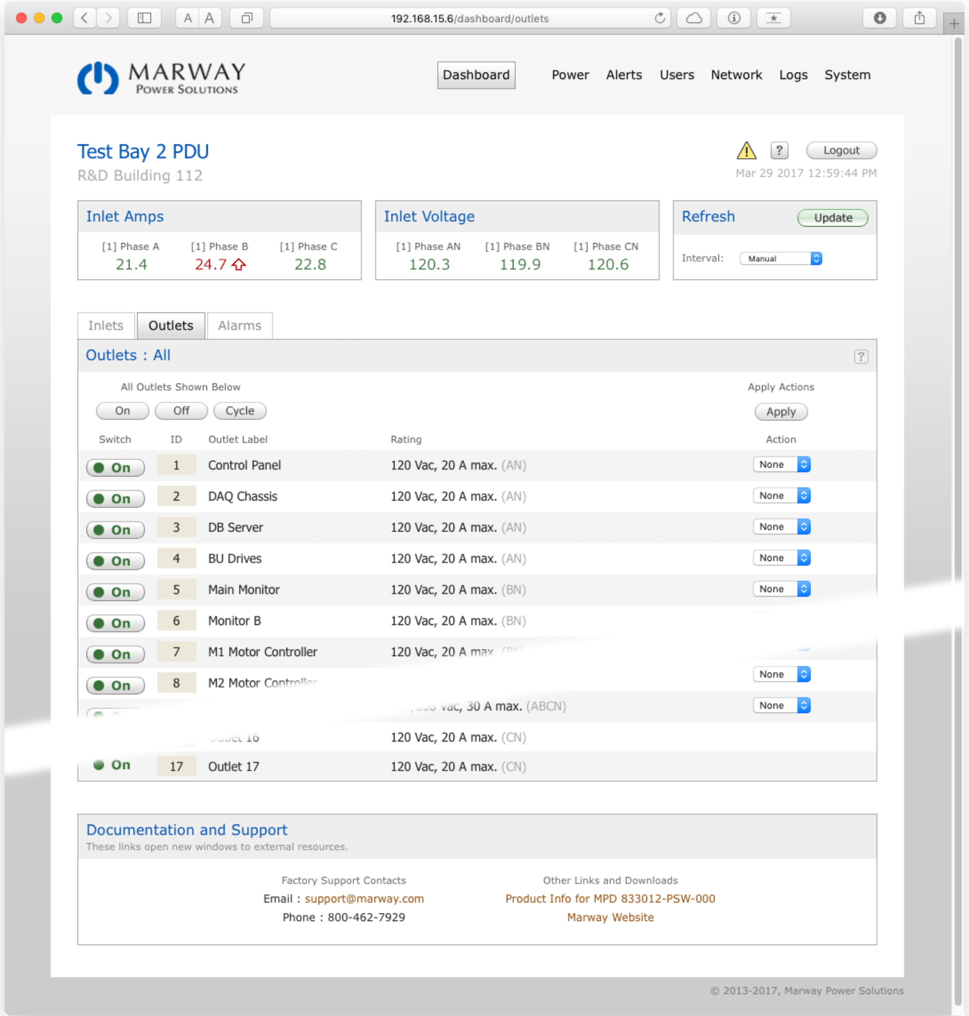
A current Phase is a single conductor (a wire) from an Inlet. Inlet current Phases carry the full current load of the PDU, which can be measured. These values are useful for understanding how close the PDU is operating to the limit of an upstream circuit breaker(s).

A voltage Phase is a pair of conductors across which voltage can be measured. Knowing this value indicates how stable the incoming power is which may be critical to some types of equipment powered from the PDU.

Depending on the model, the PDU may be equipped with inlet power monitoring, current monitoring, or no monitoring.

- Power monitoring measures both current and voltage at the inlet(s), and is able to derive power data such as watts, voltamps, power factor, and more.
- Current monitoring measures only current for a single 1ϕ or 3ϕ inlet.
- If there is no inlet monitoring at all, there may still be inlet ratings available for viewing.

If the PDU is equipped with inlet monitoring, data values are available to view on the web Dashboard, through the CLI command `getPhase`, through SNMP, and through the RESTful API. This section will discuss the human interfaces for the web and CLI. See this section on page XX for details about SNMP, and for this section on page XX for details about the RESTful API. See “Dashboard Setpoint Status Display” on page 50 for available data details.



192.168.15.6/dashboard/outlets

Test Bay 2 PDU
R&D Building 112

Mar 29 2017 12:59:44 PM

Inlet Amps

[1] Phase A	[1] Phase B	[1] Phase C
21.4	24.7 ⇧	22.8

Inlet Voltage

[1] Phase AN	[1] Phase BN	[1] Phase CN
120.3	119.9	120.6

Refresh

Interval: Manual

Outlets : All

All Outlets Shown Below

Apply Actions

Switch	ID	Outlet Label	Rating	Action
● On	1	Control Panel	120 Vac, 20 A max. (AN)	None
● On	2	DAQ Chassis	120 Vac, 20 A max. (AN)	None
● On	3	DB Server	120 Vac, 20 A max. (AN)	None
● On	4	BU Drives	120 Vac, 20 A max. (AN)	None
● On	5	Main Monitor	120 Vac, 20 A max. (BN)	None
● On	6	Monitor B	120 Vac, 20 A max. (BN)	
● On	7	M1 Motor Controller	120 Vac, 20 A max.	
● On	8	M2 Motor Controller		None
			Vac, 30 A max. (ABCN)	None
		Outlet 16	120 Vac, 20 A max. (CN)	
● On	17	Outlet 17	120 Vac, 20 A max. (CN)	

Documentation and Support
These links open new windows to external resources.

Factory Support Contacts
Email : support@marway.com
Phone : 800-462-7929

Other Links and Downloads
Product Info for MPD 833012-PSW-000
Marway Website

© 2013-2017, Marway Power Solutions

Since various models and options will have different power monitoring capabilities (or even none at all in switched-only systems), the software views displayed throughout this user guide may vary from your actual unit.

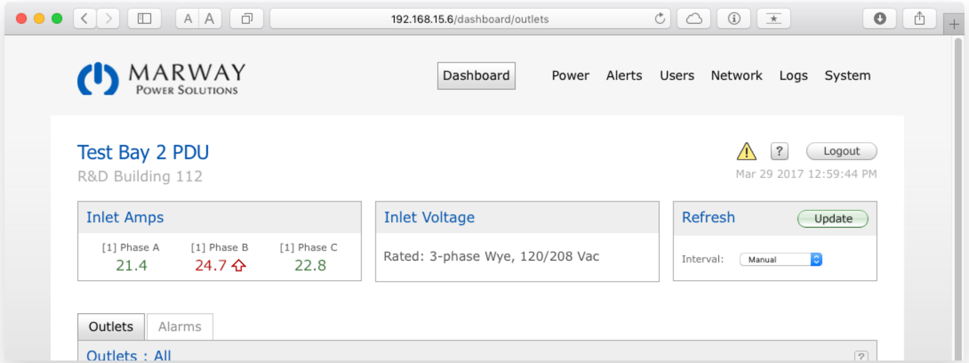
Inlet Data via Web Dashboard

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

The browser view on the previous page shows the default Dashboard layout for a three-phase inlet power monitored system with outlet switching for a number of outlets. Everything above the three panels titled Inlet Amps, Inlet Voltages, and Refresh is common to all web pages. This includes the main menu tabs at the top, the PDU label and location (which are defined in the System tab), and the cluster of items by the Logout button.

Web Dashboard Inlet Summary

If included on your PDU, the Inlet Amps and Inlet Voltages panels provide an overview of amps and volts as measured at the inlet. Current Monitored systems will show rated volts instead of measured volts. Regardless of which of the lower tabbed views is visible (Inlets, Outlets, Alarms), the top panels remains visible as a system summary. See "Dashboard Setpoint Status Display" on page 50 for examples of how alarm status is displayed with amps and volts values.



Test Bay 2 PDU
R&D Building 112

Mar 29 2017 12:59:44 PM

Inlet Amps

[1] Phase A	[1] Phase B	[1] Phase C
21.4	24.7 ⇧	22.8

Inlet Voltage

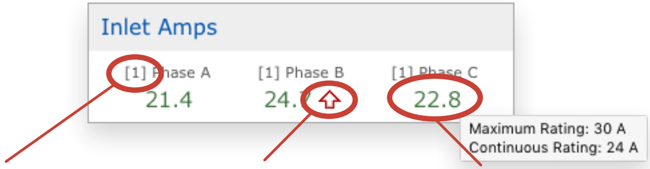
Rated: 3-phase Wye, 120/208 Vac

Refresh

Interval: Manual

Outlets : All

A power monitored inlet (previous page) will include measured amps and volts displayed on the Dashboard with more data available on the Inlets tab. A current monitored inlet (above) will include measured amps, but display the inlet's voltage rating and not measured values. There will be no Inlets tab as there is no other data to display.



Maximum Rating: 30 A
Continuous Rating: 24 A

The [1] correlates to the ID of the power Inlet in the Inlets tab.

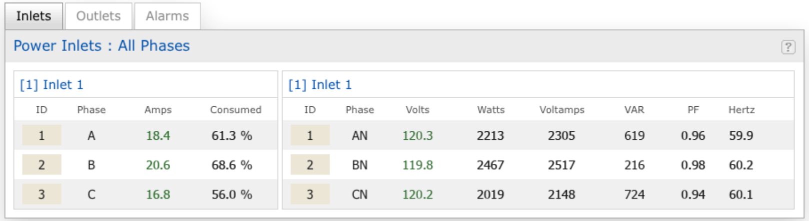
One of four icons to indicate which setpoint has been exceeded.

Hover a mouse pointer over an Amps or Volts value to see the rating of that measurement.

Web Dashboard Inlet Details

The Inlets tab displays details of all Phases. The data includes the measured current, voltage, and derived data of each Phase. Amps and Volts have setpoints, and the display of these values may include an icon depicting an alarm status. See “Dashboard Setpoint Status Display” on page 50 for examples of how alarm status is displayed with amps and volts values.

The label for each Inlet (shown as “Inlet 1” by default) can be edited, but the labels for each Phase are fixed using the phase identifiers. See “Power Settings” on page 39 for details on changing the Inlet label as well as the Phase setpoint settings.



Power Inlets : All Phases

[1] Inlet 1

ID	Phase	Amps	Consumed
1	A	18.4	61.3 %
2	B	20.6	68.6 %
3	C	16.8	56.0 %

[1] Inlet 1

ID	Phase	Volts	Watts	Voltamps	VAR	PF	Hertz
1	AN	120.3	2213	2305	619	0.96	59.9
2	BN	119.8	2467	2517	216	0.98	60.2
3	CN	120.2	2019	2148	724	0.94	60.1

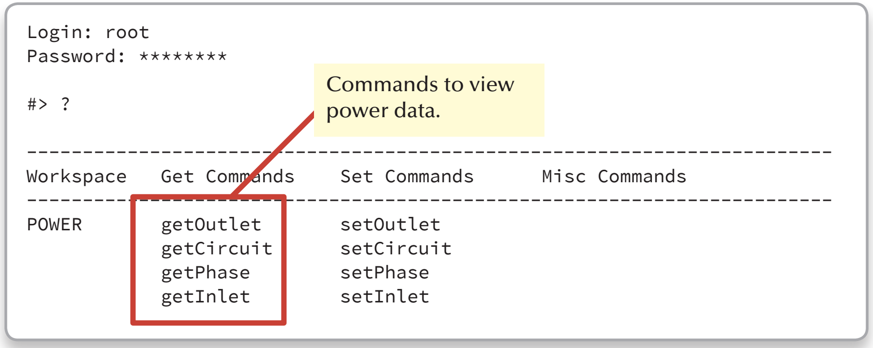
Inlet Data via Command Line

Access to power data on the command line comes in the form of three commands, each one specific to a power device: `getInlet`, `getPhase`, `getCircuit`, and `getOutlet`.

Each command may be used in various forms to extract data in a variety of ways. Review “Command Syntax” on page 91 for details of CLI get command syntax patterns.

CLI Inlet Data

The command `getInlet` provides display of the power inlet label. To see the power data of the phases in an inlet, use `getPhase`. See “setInlet” on page 93 for details on changing the label.



```
Login: root
Password: ********

#> ?

---------------------------------------------------------------------------
Workspace    Get Commands     Set Commands       Misc Commands
---------------------------------------------------------------------------
POWER         getOutlet        setOutlet
              getCircuit       setCircuit
              getPhase         setPhase
              getInlet         setInlet
```

Commands to view power data.

CLI Phase Data

Within the electrical power industry, the term Phase gets used to mean two things. It can be a single conductor (a wire) of a power source on which current is measured, and it can be a pair of conductors across which voltage is measured.

In the RCM software, single-phase conductors are labeled L(ine) and N(eutral). Therefore, the system will use the labels Phase L and Phase LN to refer to the current and voltage phases.

For three-phase systems, current phases are labeled A, B, and C. For delta configurations, voltage phases are labeled AB, BC, and CA. For wye configurations, voltage phases are labeled AN, BN, and CN.

In the RCM software, a Phase object is inherently a component of the inlet. Therefore, the current Phase measurements are the full current load of the PDU. This value is useful for understanding how close the PDU is operating to the limit of its capacity, and to the limit of the upstream circuit breaker.

The `getPhase` command displays a range of possible values depending on the supported features. Refer to "getPhase" on page 94 for additional command details, but the basic command formats to view power data are:

`getPhase [id]`	returns all power data and settings
`getPhase [id] power`	returns only power data

For current-monitored inlets, the system will provide measured amps. For power-monitored inlets, the system will provide measured amps and volts, and a number of derived calculations such as watts, voltamps, power factor and more

When phase volts and/or amps measurements are included, they will have setpoints. The display of measurement values may include an alarm indicator. See "CLI Setpoint Status Display" on page 51 for examples of the various ways which data and alarm status are displayed.

The label for each Phase is fixed to the phase name, however see "setPhase" on page 94 for details on changing the amps setpoint settings.

```
#> getPhase 1

   Phase ID                = 1: AN
   Rated Volts             = 120 Vac (AN)
   Rated Amps Maximum      = 30 A (A)
   Rated Amps Continuous = 24 A
   Amps RMS                = 18.4 A
   Consumed Max Amps       = 61.3 %
   Volts RMS               = 120.3 Vac
   Frequency               = 59.9 Hz
   Watts                   = 2213 W
   Voltamps                = 2305 VA
   Voltamps Reactive       = 619 VA
   Power Factor            = 0.96 leading
   High Critical Amps      = disabled
   High Warning Amps       = disabled
   Low Warning Amps        = disabled
   Low Critical Amps       = disabled
   Hysteresis Amps         = 0.5 A
   Debounce Amps           = 10.0 s
   High Warning Volts      = disabled
   High Warning Volts      = disabled
   Low Warning Volts       = disabled
   Low Critical Volts      = disabled
   Hysteresis Volts        = 4.0 V
   Debounce Volts          = 10.0 s
```

```
#> getPhase 1 power

   Phase ID                = 1: AN
   Amps RMS                = 18.4 A
   Consumed Max Amps       = 61.3 %
   Volts RMS               = 120.3 Vac
   Frequency               = 59.9 Hz
   Watts                   = 2213 W
   Voltamps                = 2305 VA
   Voltamps Reactive       = 619 VA
   Power Factor            = 0.96 leading
```

Outlet Data

A power Outlet is a single power connector mounted to the PDU chassis for providing power to a connected piece of electrical equipment. It may be a female receptacle, one or more pins in a multi-pin connector, or even a screw terminal—anything which conveys power from inside the PDU to a piece of equipment connected to the PDU.

Depending on the model, the PDU may or may not be equipped with outlet switching (or even a mix of some outlets being switched, and some not). Regardless of the presence of switching, outlet devices are still defined with ratings and user labels which can be viewed and altered.

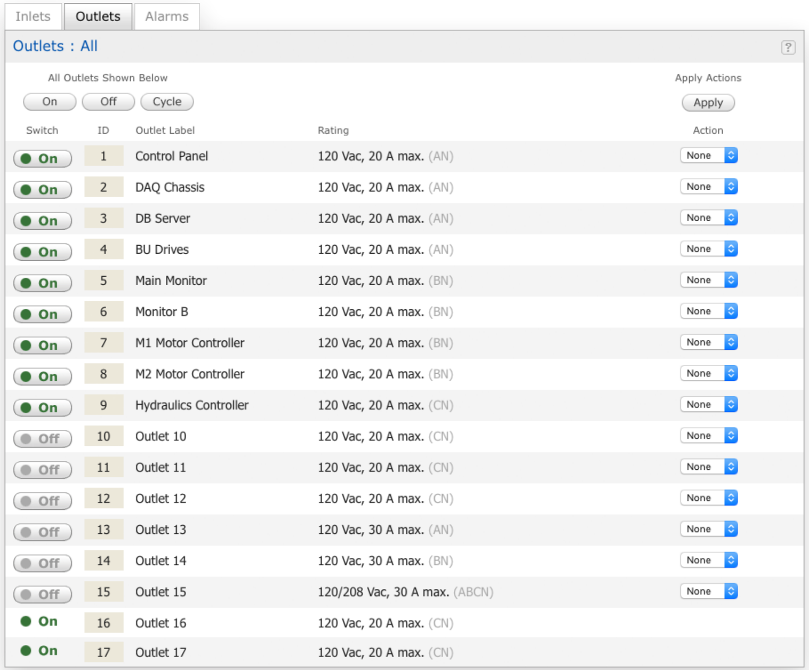
Outlet Data via Web

The Outlets tab is often "command central" of the dashboard (assuming outlet switching is the most common control task). In this section, we'll focus on the layout of the Outlets panel. For outlet switching details, refer to "Outlet Control" on page 45.

Each outlet is shown with its switch state. Outlets which are not switchable will still be displayed, but with a permanent On state and no button (such as outlets 16 and 17 in the Dashboard view to the right).

The ID column is a simple incremental ID for each outlet which corresponds to the ID value required for command line and other interfaces. In most models, hovering the browser pointer over the ID column reveals the Panel Name which is the name on the PDU chassis for that outlet. For standard products, this is typically a "J number" with the same value as the ID (J1, J2, etc.). Custom units may have different panel names.

The label for each Outlet (set to "Outlet 1" etc. by default) can be edited either on the web UI Power > Outlets tab, or using the `setOutlet` CLI command.



Switch	ID	Outlet Label	Rating	Action
● On	1	Control Panel	120 Vac, 20 A max. (AN)	None
● On	2	DAQ Chassis	120 Vac, 20 A max. (AN)	None
● On	3	DB Server	120 Vac, 20 A max. (AN)	None
● On	4	BU Drives	120 Vac, 20 A max. (AN)	None
● On	5	Main Monitor	120 Vac, 20 A max. (BN)	None
● On	6	Monitor B	120 Vac, 20 A max. (BN)	None
● On	7	M1 Motor Controller	120 Vac, 20 A max. (BN)	None
● On	8	M2 Motor Controller	120 Vac, 20 A max. (BN)	None
● On	9	Hydraulics Controller	120 Vac, 20 A max. (CN)	None
● Off	10	Outlet 10	120 Vac, 20 A max. (CN)	None
● Off	11	Outlet 11	120 Vac, 20 A max. (CN)	None
● Off	12	Outlet 12	120 Vac, 20 A max. (CN)	None
● Off	13	Outlet 13	120 Vac, 30 A max. (AN)	None
● Off	14	Outlet 14	120 Vac, 30 A max. (BN)	None
● Off	15	Outlet 15	120/208 Vac, 30 A max. (ABCN)	None
● On	16	Outlet 16	120 Vac, 20 A max. (CN)	
● On	17	Outlet 17	120 Vac, 20 A max. (CN)	

The Rating column identifies the voltage and current rating, as well as the voltage phase(s) the outlet is connected to.

The Action column provides access to a cycle command for outlets, and allows multiple outlets to be preset to turn on, off, or cycled with a single press of the Apply button at the top of the column.

Outlet Data via Command Line

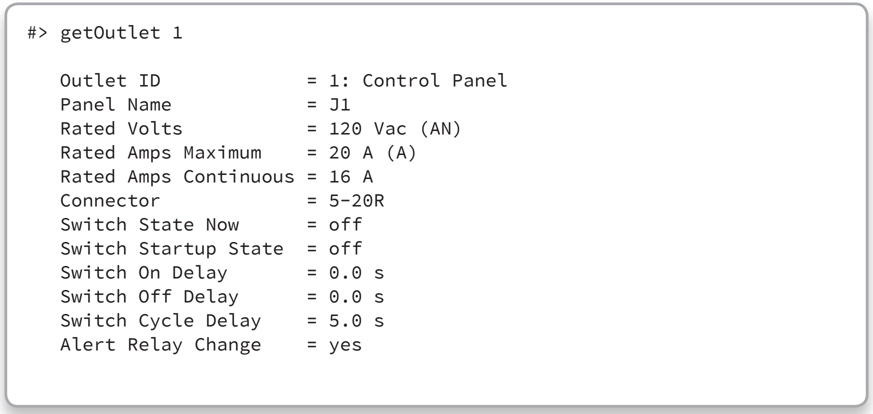
A power Outlet is a single power connector mounted to the PDU chassis for providing power to a connected piece of electrical equipment.

The `getOutlet` command displays the ratings of the outlet and it’s switch state and settings (if switchable).

Refer to “getOutlet” on page 96 for additional command details, but the basic command formats to view power data are:

`getOutlet [id]`	returns all power ratings and settings
`getOutlet [id] switch`	returns only switch state
`getOutlet [id] switch_info`	returns switch state and delay settings

The label for each Outlet may be edited. See “setOutlet” on page 96 for details on changing the label and the amps setpoint settings.



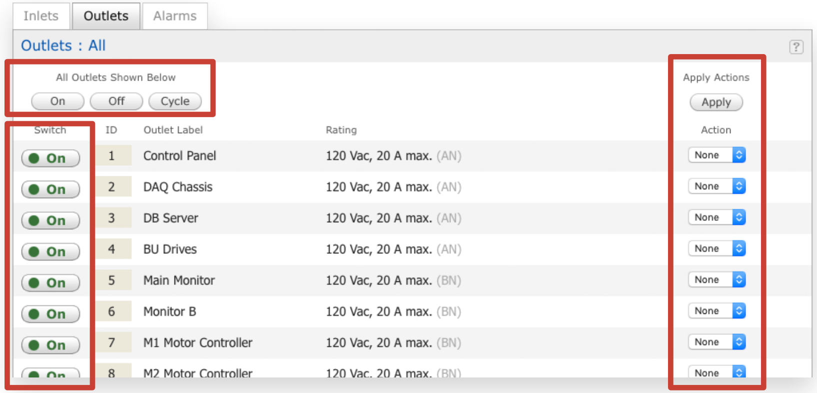
```
#> getOutlet 1

   Outlet ID               = 1: Control Panel
   Panel Name              = J1
   Rated Volts             = 120 Vac (AN)
   Rated Amps Maximum      = 20 A (A)
   Rated Amps Continuous = 16 A
   Connector               = 5-20R
   Switch State Now        = off
   Switch Startup State  = off
   Switch On Delay         = 0.0 s
   Switch Off Delay        = 0.0 s
   Switch Cycle Delay      = 5.0 s
   Alert Relay Change      = yes
```

Outlet Control

Outlets may be controlled using the web Dashboard, the CLI command `setOutlet`, SNMP, and the RESTful API. Each outlet configured for remote switching can be switched on, off, and cycled.

A cycled outlet first switches 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 actions, switching also includes the ability to add delays to these actions.



Switch	ID	Outlet Label	Rating	Action
● On	1	Control Panel	120 Vac, 20 A max. (AN)	None
● On	2	DAQ Chassis	120 Vac, 20 A max. (AN)	None
● On	3	DB Server	120 Vac, 20 A max. (AN)	None
● On	4	BU Drives	120 Vac, 20 A max. (AN)	None
● On	5	Main Monitor	120 Vac, 20 A max. (BN)	None
● On	6	Monitor B	120 Vac, 20 A max. (BN)	None
● On	7	M1 Motor Controller	120 Vac, 20 A max. (BN)	None
● On	8	M2 Motor Controller	120 Vac, 20 A max. (BN)	None

Outlet Switching Delays

Each remotely switchable outlet includes On Delay, Off Delay, and Cycle Delay settings. Even when configured with values greater than zero, these delays do not always get applied.

When any one outlet is switched on or off alone (by any of the network interfaces), the delay values are ignored, and the outlet is switched on or off immediately. However, when all outlets or multiple outlets are switched as a set, then On Delay and Off Delay values are utilized. If outlets 2, 4 and 7 were selected in the web Action column to be turned on, On Delay settings for those three outlets would be executed before each of the outlets was switched.

Cycle Delay is always applied, however, the actual delay between off and on steps can be influenced by On Delay. During a Cycle command involving multiple outlets, on step will be delayed by the larger of On Delay or Cycle Delay. This can be confusing, but the intent is to make sure the larger of either the On Delay or Cycle Delay value is used in case there was specifically adjusted startup dependencies defined by On Delay. A single outlet which is cycled on its own is switched off immediately and uses Cycle Delay before switching on.

Staggering Startups to Curb Inrush

There is generally no need to use on delays to stagger outlets to minimize inrush current. At startup, there is inherently a small time gap between turning each outlet on already.

Staggering Startups for Dependencies

A primary 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 that capacitors are fully discharged, that hard drives have stopped spinning, and other factors come to a “full stop” before power is reapplied. For example, we commonly hear that computers should be switched off for 10 seconds before turning them on again. Those details and the time needed, if any, are specific to the equipment. Check owner’s manuals for power cycling guidelines.

Switching Sets of Outlets

When multiple outlets are targeted to be switched, 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.

Outlet Control via Web

Outlet Switching

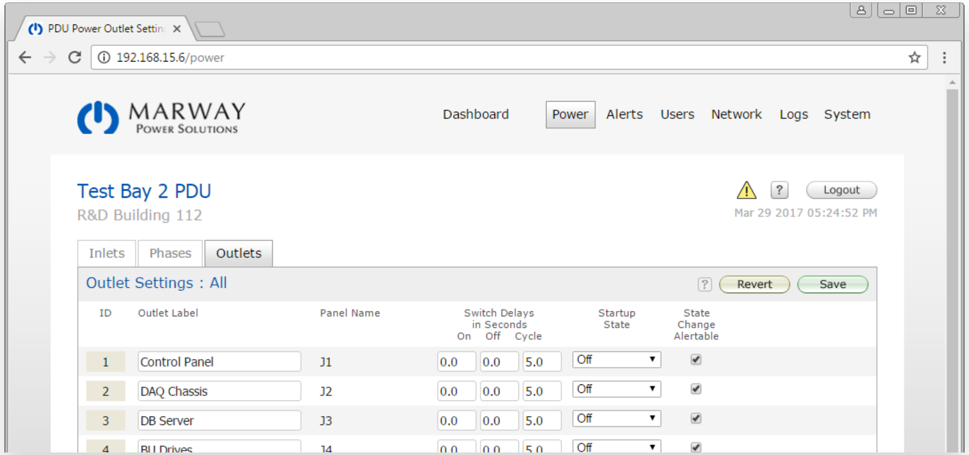
The web Dashboard offers three ways to manipulate outlet state. The first is an All Outlets set of buttons to affect all outlets with a single command to turn them on, turn them off, or cycle them. When using these controls, each outlet is handled in sequence as described in “Switching Sets of Outlets”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

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 Power > Outlets tab of the web interface. See



Test Bay 2 PDU

R&D Building 112

Mar 29 2017 05:24:52 PM

Outlet Settings : All

ID	Outlet Label	Panel Name	Switch Delays in Seconds On	Off	Cycle	Startup State	State Change Alertable
1	Control Panel	J1	0.0	0.0	5.0	Off	☑
2	DAQ Chassis	J2	0.0	0.0	5.0	Off	☑
3	DB Server	J3	0.0	0.0	5.0	Off	☑
4	BU Drives	J4	0.0	0.0	5.0	Off	☑

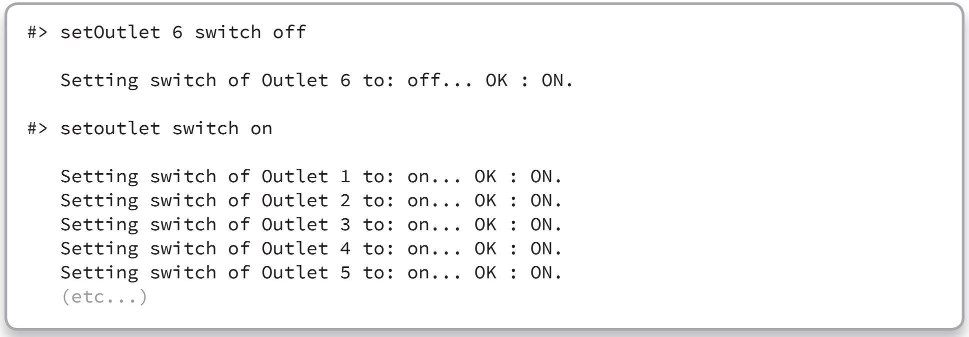
Outlet Control via Command Line

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/cycle delays are ignored. However, when switching all outlets, the on/off/cycle delays are honored. As of this version, there is not the ability to list a set of arbitrary outlets in `setOutlet`.



```
#> setOutlet 6 switch off

   Setting switch of Outlet 6 to: off... OK : ON.

#> setoutlet switch on

   Setting switch of Outlet 1 to: on... OK : ON.
   Setting switch of Outlet 2 to: on... OK : ON.
   Setting switch of Outlet 3 to: on... OK : ON.
   Setting switch of Outlet 4 to: on... OK : ON.
   Setting switch of Outlet 5 to: on... OK : ON.
   (etc...)
```

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`

See “setOutlet” on page 96 for details.

```
#> setOutlet 6 on_delay 0.5

   Setting on_delay of Outlet 6 to: 0.5... OK.

#> setOutlet cycle_delay 8.0

   Setting cycle_delay of Outlet 1 to: 8.0... OK.
   Setting cycle_delay of Outlet 2 to: 8.0... OK.
   Setting cycle_delay of Outlet 3 to: 8.0... OK.
   Setting cycle_delay of Outlet 4 to: 8.0... OK.
   Setting cycle_delay of Outlet 5 to: 8.0... OK.
   (etc...)
```

Viewing Alarm Data

Alarms are available to view on the web Dashboard, and through the CLI command `getAlarm` (or `getAlarms`). Alarm data will only be available if one or more setpoints have first been set for a supported power object, and then one of these setpoints has been tripped. Section "Dashboard Setpoint Status Display" on page 50 covers in detail the terminology and operation of alarms and setpoints.

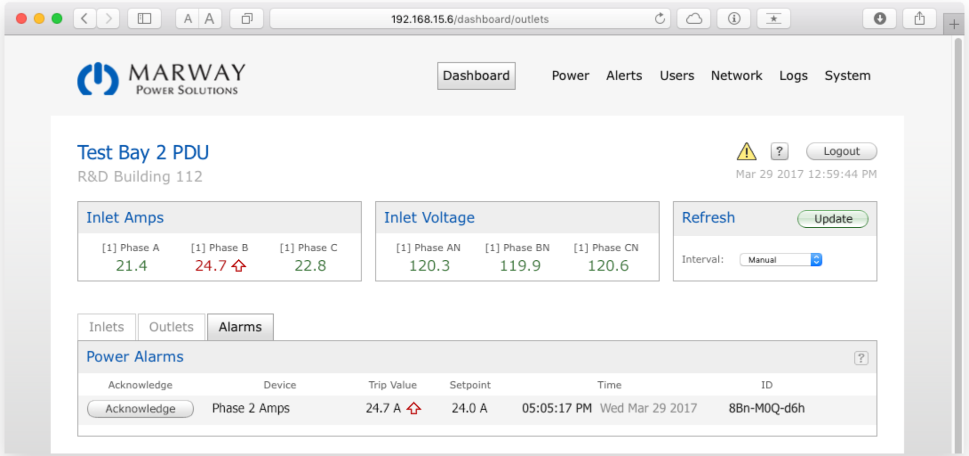
Web Dashboard Alarm Data

The top figure to the right shows the web Dashboard Inlet Amps panel with Phase B in an alarm condition. Refer to "Dashboard Setpoint Status Display" on page 50 for an explanation of the display symbols. When a value is displayed in red and has an up or down pointing arrow either solid or hollow, that value has crossed an alarm setpoint. To see more information about that alarm, open the Dashboard Alarms panel.

The Dashboard Alarms panels lists all active alarms. It is possible for an alarm to subside, so refreshing the page will always provide the most recent alarm information. The Alarm table includes the following information:

Acknowledge: If the Alerts Settings has been set up to send an alert about this alarm to one or more users, that alert will repeat until the alarm has been acknowledged. An acknowledgment is made by clicking a link in an alert message, or by pressing this button. See "Dashboard Data and Alarm Indication" on page 49 for more information.

Device: The power device, its ID, and the setpoint measurement.



192.168.15.6/dashboard/outlets

Test Bay 2 PDU
R&D Building 112

Mar 29 2017 12:59:44 PM

Inlet Amps

[1] Phase A	[1] Phase B	[1] Phase C
21.4	24.7 ⇧	22.8

Inlet Voltage

[1] Phase AN	[1] Phase BN	[1] Phase CN
120.3	119.9	120.6

Refresh Update

Interval: Manual

Inlets | Outlets | Alarms

Power Alarms

Acknowledge	Device	Trip Value	Setpoint	Time	ID
Acknowledge	Phase 2 Amps	24.7 A ⇧	24.0 A	05:05:17 PM Wed Mar 29 2017	8Bn-M0Q-d6h

Dashboard Data and Alarm Indication

8.6 A	Normal data status in green, no icon.
8.6 A ⬆	High Critical setpoint has been triggered.
8.6 A ⇧	High Warning setpoint has been triggered.
8.6 A ⇩	Low Warning setpoint has been triggered.
8.6 A ⬇	Low Critical setpoint has been triggered.
--- A	Data is not available. Check "Troubleshooting Guide"

Trip Value: This is the value that was first detected as having crossed the setpoint. The icons correspond to those shown in "Dashboard Setpoint Status Display" on page 50.

Setpoint: The value of the setpoint which has been tripped. The icon in the Value column indicates which specific setpoint was tripped.

Time and *Date*: The time and date the setpoint trip was first detected.

AlarmID: This is a random ID unique to the specific instance of the current alarm. If another alarm happens for the same device and the same trip point, it will have a different ID. The ID is used for logging, and for acknowledgment links.

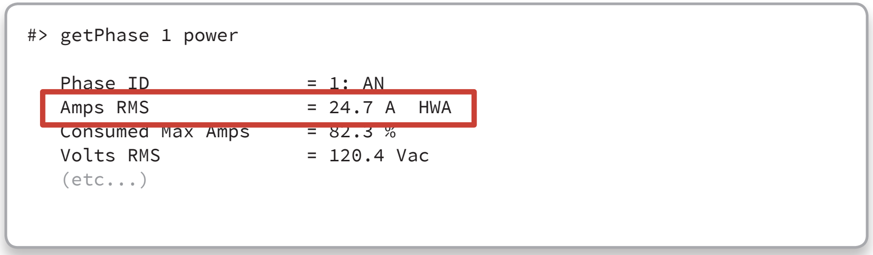
Dashboard Setpoint Status Display

All Amps and Volts values displayed on the Dashboard have setpoints which can be used to trigger an alarm. The status of a data value and setpoint alarm are indicated using color and icons as shown in the sidebar on the right. Refer to the following for details on these related topics:

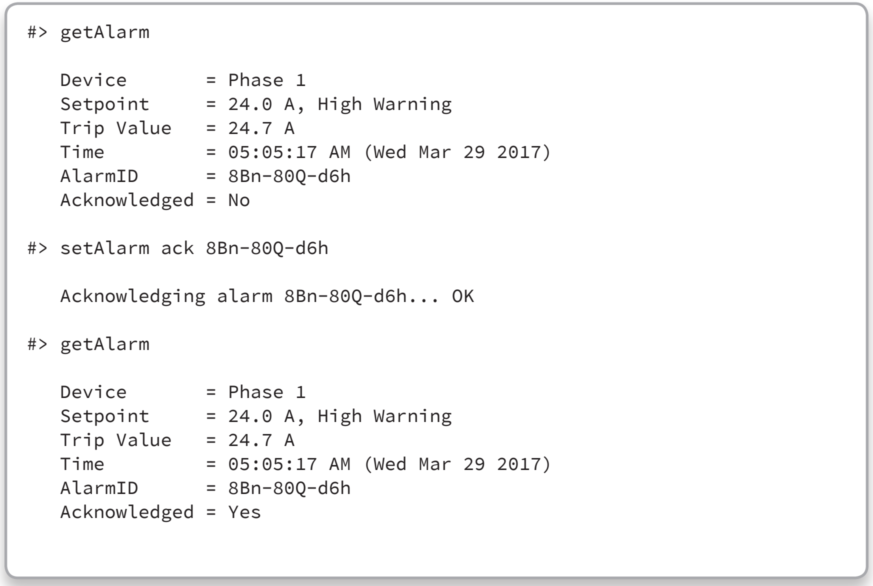
- "Viewing Alarm Data" on page 49 for details on the Dashboard Alarms tab and available CLI commands to view alarm data.
- "Setpoint Accuracy and Limits" on page 85 for details on understanding data accuracy and setpoint limits when adjusting setpoints.

Command Line Alarm Data

When viewing power data, values with alarms (e.g. amps and volts) will be followed by a setpoint acronym such as HWA, LCV or others as indicated in "CLI Setpoint Status Display" on page 51 and shown in the top figure.



```
#> getPhase 1 power

   Phase ID                = 1: AN
   Amps RMS                = 24.7 A   HWA
   Consumed Max Amps       = 82.3 %
   Volts RMS               = 120.4 Vac
   (etc...)
```



```
#> getAlarm

   Device       = Phase 1
   Setpoint     = 24.0 A, High Warning
   Trip Value   = 24.7 A
   Time         = 05:05:17 AM (Wed Mar 29 2017)
   AlarmID      = 8Bn-80Q-d6h
   Acknowledged = No

#> setAlarm ack 8Bn-80Q-d6h

   Acknowledging alarm 8Bn-80Q-d6h... OK

#> getAlarm

   Device       = Phase 1
   Setpoint     = 24.0 A, High Warning
   Trip Value   = 24.7 A
   Time         = 05:05:17 AM (Wed Mar 29 2017)
   AlarmID      = 8Bn-80Q-d6h
   Acknowledged = Yes
```

To view all alarms, use the command `getAlarm` or `getAlarms`. This will list all alarms. The command can also be used to get specific device alarms as with the command `getAlarm phase` which would display alarms for phases only.

To acknowledge a specific alarm using the command line, use the command `setAlarm ack` followed by the desired alarm ID (see the sample CLI session at the right). To acknowledge all alarms, use `setAlarm ack` with no ID.

CLI Setpoint Status Display

All Amps and Volts values have setpoints which can be used to trigger an alarm. The status of a data value and setpoint alarm are indicated using acronyms after the measured value as shown in the sidebar on the right. Refer to the following for details on these related topics:

- "Dashboard Setpoint Status Display" on page 50 for details on changing the setpoints.
- "Dashboard Setpoint Status Display" on page 50 for detailed explanation of what setpoints are, how they work, and how to use them for alarms.
- "Viewing Alarm Data" on page 49 for details on the Dashboard Alarms tab and available CLI commands to view alarm data.

Command Line Data and Alarm Indication

Normal data status shows only the data value and units. If the value is followed by one of these acronyms, it indicates which alarm setpoint has been tripped.

Amps:

HCA	High Critical Amps setpoint has been triggered.
HWA	High Warning Amps setpoint has been triggered.
LWA	Low Warning Amps setpoint has been triggered.
LCA	Low Critical Amps setpoint has been triggered.

Volts:

HCV	High Critical Volts setpoint has been triggered.
HWV	High Warning Volts setpoint has been triggered.
LWV	Low Warning Volts setpoint has been triggered.
LCV	Low Critical Volts setpoint has been triggered.

```
Amps RMS      = 16.87 A

Amps RMS      = 18.32 A HCA

Volts RMS     = 109.8 V LCV
```

The first example is a normal condition with no alarm. The next two examples have alarms.

Viewing Log Data

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

Logged Events

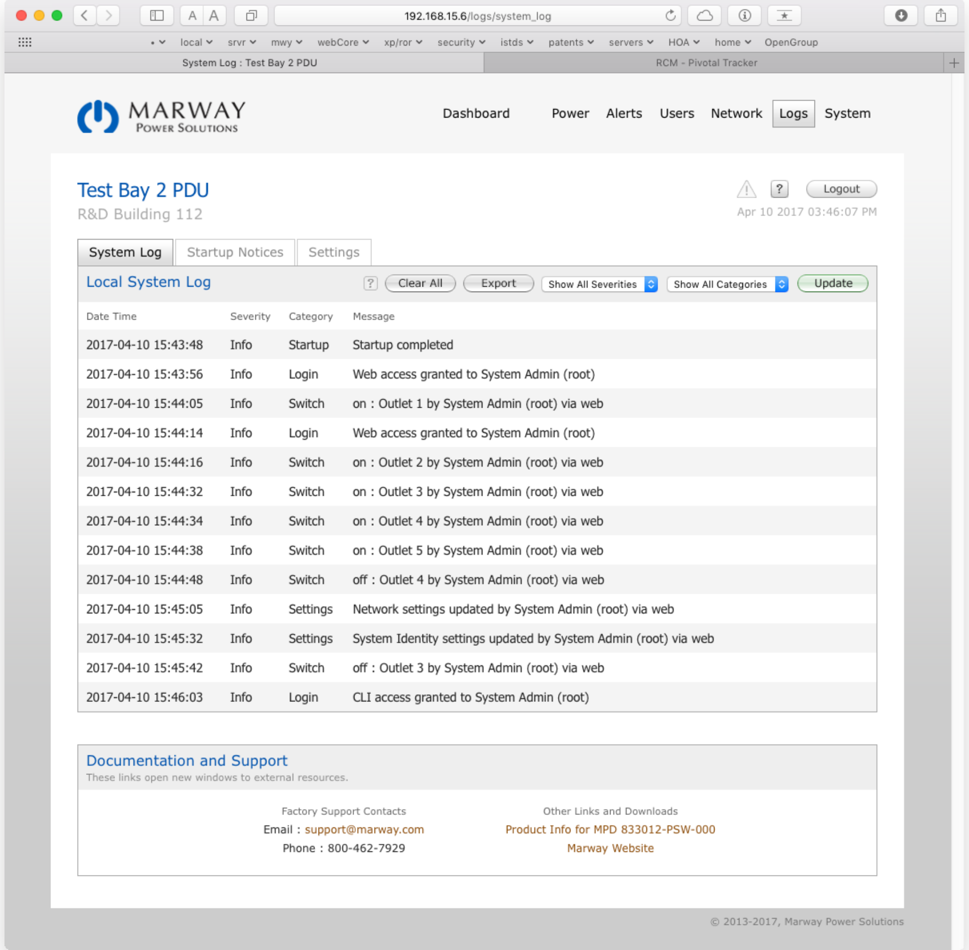
The RCM software maintains an event log intended for tracking recent event history. Logged events include:

- System startup
- Successful user login
- Failed user login
- Settings updated
- EPO pressed
- Outlet state change
- Alarm trip
- Alert preparation
- Alert sent
- Internal errors

Exported Log Format

An exported 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:

- DateTime
- Severity
- PDU IP
- PDU Location
- Category
- Message
- PDU Model
- PDU Version



Test Bay 2 PDU

R&D Building 112

Apr 10 2017 03:46:07 PM

Local System Log

Date Time	Severity	Category	Message
2017-04-10 15:43:48	Info	Startup	Startup completed
2017-04-10 15:43:56	Info	Login	Web access granted to System Admin (root)
2017-04-10 15:44:05	Info	Switch	on : Outlet 1 by System Admin (root) via web
2017-04-10 15:44:14	Info	Login	Web access granted to System Admin (root)
2017-04-10 15:44:16	Info	Switch	on : Outlet 2 by System Admin (root) via web
2017-04-10 15:44:32	Info	Switch	on : Outlet 3 by System Admin (root) via web
2017-04-10 15:44:34	Info	Switch	on : Outlet 4 by System Admin (root) via web
2017-04-10 15:44:38	Info	Switch	on : Outlet 5 by System Admin (root) via web
2017-04-10 15:44:48	Info	Switch	off : Outlet 4 by System Admin (root) via web
2017-04-10 15:45:05	Info	Settings	Network settings updated by System Admin (root) via web
2017-04-10 15:45:32	Info	Settings	System Identity settings updated by System Admin (root) via web
2017-04-10 15:45:42	Info	Switch	off : Outlet 3 by System Admin (root) via web
2017-04-10 15:46:03	Info	Login	CLI access granted to System Admin (root)

Documentation and Support

These links open new windows to external resources.

Factory Support Contacts
Email : support@marway.com
Phone : 800-462-7929

Other Links and Downloads
Product Info for MPD 833012-PSW-000
Marway Website

© 2013-2017, Marway Power Solutions

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. 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.

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. The number of messages saved is adjustable from 0 to 1000. The advantage of a small log is quicker startup time. The advantage of a larger log is a longer history of events.

How long a period of time the log covers depends on the maximum log size settings, and 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 if the maximum log size is small. 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 `viewStartupLog` 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 three tabs: System Log, Startup Notices, and Settings. 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 `viewLog` to view the system log, and `viewStartupLog` to view the startup notices. The `viewLog` command accepts filter parameters to help seek out a specific type of log message. See "viewLog" on page 113.

Log Settings

There are a few settings which control the system event log. First, the size of the log in number of records can be adjusted from 0 to 1000. A settings of 0 prevents logging. The advantage of a small log is quicker startup time. The advantage of a larger log is a longer history of events.

A log of 1000 records can add a couple of minutes to the PDU startup time. If the PDU is rarely power cycled, then perhaps the expanded history is worth the extra startup time on rare occasion the PDU is powered off.

However, a smaller log of up to perhaps 200 records may provide plenty of history for a PDU which sees a daily shutdown, and would start up more quickly.

Outlet State Change Logging

There are also settings to determine whether outlet switching events are logged. All switch events triggered through the web interface are always logged. However, the logging of switching triggered through the CLI, SNMP, and RESTful API can be disabled.

If a PDU's outlets are controlled by an automated script, and the event rate is frequent, the log may fill up with switch events the user already knows will happen. This could obscure more rare, and therefore useful, event records. By disabling logging of switch events through the automation API, the log is not over run with repetitive data.

Optima RCM Software

Events Management

Events Management

Events, Alarms, and Alerts

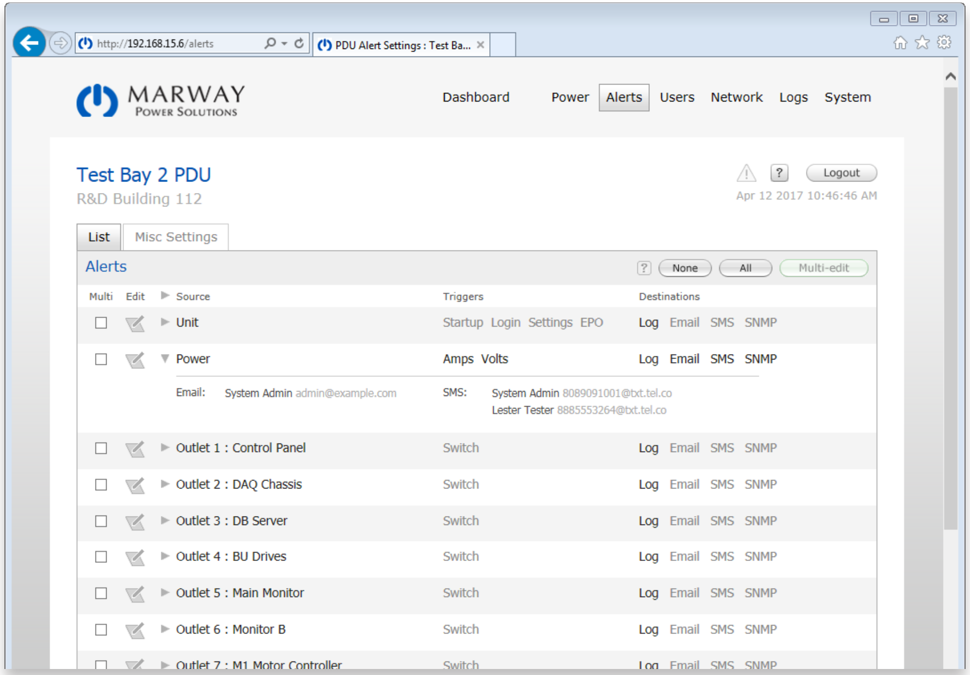
The RCM software and documentation will refer to events, alarms, and alerts. Before reading this section, it is best to get some definitions cleared up.

An *event* is any discernible action or change of state in something. Of all these terms, event is the most general.

An *alarm* is a type of event which carries the connotation of something being not normal. The power setpoints define alarms. When a setpoint is crossed, that event activates an alarm.

An *alert* is a notification. That notification could be an illuminated bulb on a control panel, or an automated email message sent from a machine. An alert may come as a result of an alarm, or a non-alarm event. For example, an email sent because a setpoint was crossed would be in response to an alarm. However, an email sent at the startup of the PDU is an alert, but the PDU startup event itself is not an alarm.

The RCM software provides user interface displays of alarms, and provides for configuration of several predefined alerts.



Test Bay 2 PDU
R&D Building 112

Apr 12 2017 10:46:46 AM

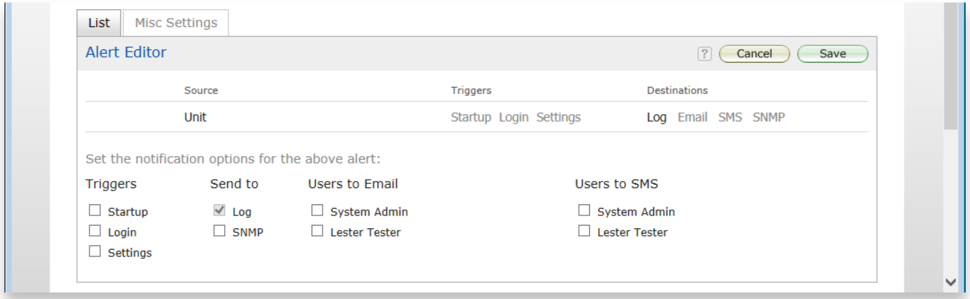
Alerts

Multi	Edit	▶ Source	Triggers	Destinations
□		▶ Unit	Startup Login Settings EPO	Log Email SMS SNMP
□		▼ Power	Amps Volts	Log Email SMS SNMP
		Email: System Admin admin@example.com	SMS: System Admin 8089091001@txt.tel.co Lester Tester 8885553264@txt.tel.co	
□		▶ Outlet 1 : Control Panel	Switch	Log Email SMS SNMP
□		▶ Outlet 2 : DAQ Chassis	Switch	Log Email SMS SNMP
□		▶ Outlet 3 : DB Server	Switch	Log Email SMS SNMP
□		▶ Outlet 4 : BU Drives	Switch	Log Email SMS SNMP
□		▶ Outlet 5 : Main Monitor	Switch	Log Email SMS SNMP
□		▶ Outlet 6 : Monitor B	Switch	Log Email SMS SNMP
□		▶ Outlet 7 : M1 Motor Controller	Switch	Log Email SMS SNMP

Adjusting Alert Settings

An alert can be configured by either the web interface or command line to send an email, SMS text message, and/or SNMP trap in response to several events.

The web interface for editing power alerts is shown to the right. Alerts can be edited individually, allowing each one to be unique, or multiple alerts can be selected an edited to have identical settings. The same is possible using the command line command `setAlert`.



Source	Triggers	Destinations
Unit	Startup Login Settings	Log Email SMS SNMP

Set the notification options for the above alert:

Triggers	Send to	Users to Email	Users to SMS
□ Startup	☑ Log	□ System Admin	□ System Admin
□ Login	□ SNMP	□ Lester Tester	□ Lester Tester
□ Settings			

Unit Alerts

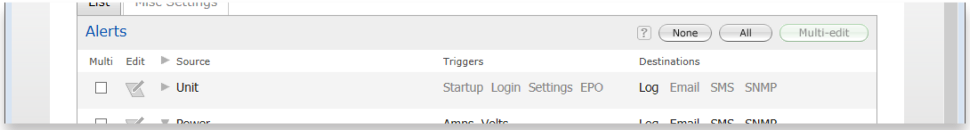
There are a number of event triggers which are grouped into a single Alert named Unit. The following events will generate an alert if configured to do:

Startup Alert :: this is sent whenever the PDU goes through a startup process whether caused by having power applied, being restarted by a software command, or being rebooted from the keypad button.

Login Alert :: this is sent whenever a user logs into the PDU via the web interface, command line, or FTP service.

Settings Alert :: this is sent whenever any setting has been edited.

EPO Alert :: this is sent whenever a local or temore EPO button has been pressed.



Multi	Edit	Source	Triggers	Destinations
□	✓	▶ Unit	Startup Login Settings EPO	Log Email SMS SNMP
□	✓	▼ Power	Amps Volts	Log Email SMS SNMP

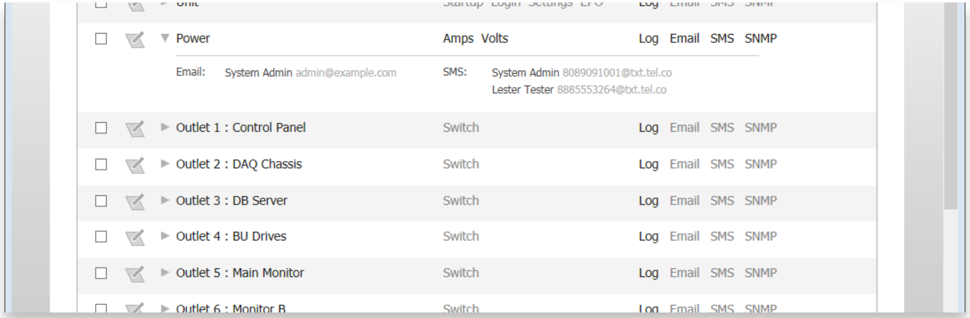
Power Alerts

If a PDU is configured with power-monitored or current-monitored inlets or circuits, it will also have configurable setpoints, and a configurable alert named Power. The Power alert is triggered by the setpoint alarms for amps or volts from any one of the phases or circuits which are monitored (depending on what the PDU hardware supports).

For example, the four possible setpoint alarms for Phase A are combined into the alert named Power. If any of the four setpoints are tripped, the same Power alert will be sent (assuming it has been configured to be sent). Depending on the features of the PDU, the following alerts may be available:

Setpoint for Phase Amps :: sent whenever one of the four setpoints of a current phase has been triggered.

Setpoint for Phase Volts :: sent whenever one of the four setpoints of a voltage phase has been triggered.



□	▶ Unit		Startup Login Settings EPO	Log Email SMS SNMP
□	▼ Power		Amps Volts	Log Email SMS SNMP
	Email:	System Admin admin@example.com	SMS:	System Admin 8089091001@txt.tel.co Lester Tester 8885553264@txt.tel.co
□	▶ Outlet 1 : Control Panel		Switch	Log Email SMS SNMP
□	▶ Outlet 2 : DAQ Chassis		Switch	Log Email SMS SNMP
□	▶ Outlet 3 : DB Server		Switch	Log Email SMS SNMP
□	▶ Outlet 4 : BU Drives		Switch	Log Email SMS SNMP
□	▶ Outlet 5 : Main Monitor		Switch	Log Email SMS SNMP
□	▶ Outlet 6 : Monitor B		Switch	Log Email SMS SNMP

Outlet Alerts

If a PDU is configured with switched outlets, each outlet can be independent configured to send an alert if the switch state changes.

Power Setpoints and Alarms

A setpoint is an adjustable value which means something specific to a process monitoring or process control system. A simple example is the thermostat in your home. When you change the temperature setting, that temperature value is a setpoint. In your home, that value is used to control the temperature by turning the heater either on or off.

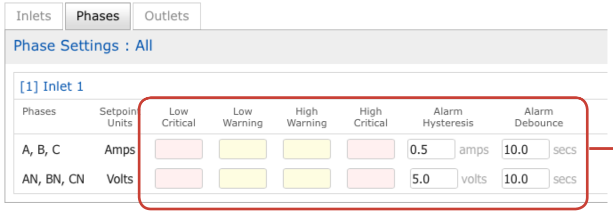
In a PDU, setpoints are often used to monitor the voltage and current of inlets, circuits, and outlets. When a setpoint value is reached, we can use that signal to send an alert to one or more people by email or SMS, or to external equipment through SNMP. This section will explain the features of RCM setpoints.

Four Setpoints Per Measurement

The thermostat example above is good for its simplicity, but setpoints can have several adjustable features. First, any one parameter we're interested in monitoring, such as the current of a power inlet, has four separate setpoints.

Let's assume we have an inlet with a circuit breaker set at 20 amps. We'd want a setpoint set to maybe 18.5 or 19.0 amps. This would let us know when we're approaching the maximum limit of the breaker. While that's a useful data point, it might also be useful to get a warning a little sooner. A second setpoint could do that. We might set that one to 16 amps. These values depend, of course, on how large the load swings are with the monitored equipment, and how quickly the load can escalate from 16 to 18.5 amps. Maybe these settings could be closer together, or lower overall. The point is, there are two setpoints to differentiate a *warning* condition from a *critical* condition, whatever those conditions are for any given system.

That example was about reaching the high limits of a circuit breaker. It's also common to want to know about the low limits. If a set of attached equipment

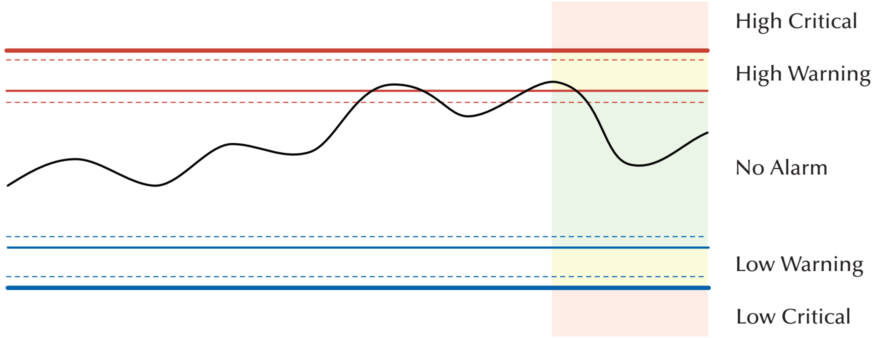


Phase Settings : All

[1] Inlet 1

Phases	Setpoin Units	Low Critical	Low Warning	High Warning	High Critical	Alarm Hysteresis	Alarm Debounce
A, B, C	Amps					0.5 amps	10.0 secs
AN, BN, CN	Volts					5.0 volts	10.0 secs

Setpoint Configuration Fields



on a specific inlet phase is expected to be drawing 13.0 amps, where even 11.5 is normal, but anything less than 10.0 indicates something is not quite right, it may be useful to have *low warning* and *low critical* setpoints to send alerts. Therefore, to be useful for a multitude of possible setups and scenarios, all setpoints in the RCM software are actually four setpoints in one.

RCM's four setpoints are labeled high critical, high warning, low warning, and low critical. In the software, these may be abbreviated using a pattern like HCA for high critical amps, or LWV for low warning volts.

Various PDU models are equipped with various sensors, so not all setpoints may be available on any one particular model, but the following are potential setpoints available on a system with this version of software:

> Phase amps :: each power conductor on the input cord (or other connection means) has its own four setpoints for amps (HCA, HWA, LWA, LCA).
>
> Phase volts :: each pair of power conductors (such as LN, or AB, BC, CB) has its own four setpoints for volts (HCV, HWV, LWV, LCV).

Hysteresis and Debounce

In addition to multiple setpoints, there are two adjustable features which can modify how the monitoring system reacts to the actual setpoint value: hysteresis and debounce. In the RCM software, each measured parameter has its own settings for hysteresis and debounce, but those settings are the same for each of the four setpoints. So, a debounce value of 2 seconds applies to high critical, high warning, low warning, and low critical. In the illustration on the next page, you can see the four setpoints, and the labels for hysteresis and debounce.

If we return to the home thermostat example, you're no doubt aware that when the setpoint is 68 degrees, the heater doesn't come on until the temperature reads 67 degrees, and probably doesn't shut off until the temperature reads 69 degrees. This seeming inaccuracy is on purpose. If the system treated 68 degrees precisely as the on/off point, you could have a case where the temperature oscillates between 67.95° and 68.05°. The heater would be switched on and off quite frequently, and house heaters don't like that. If we translate that to our PDU, it would send an email every few seconds while the current oscillated around the setpoint, and *people* don't like *that*.

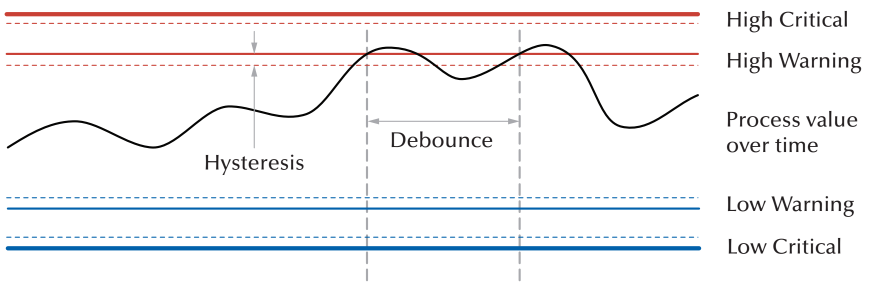
Hysteresis

To minimize the scenario of repeated activations, each setpoint has hysteresis—a buffer zone, or band, around the setpoint which fine tunes when that setpoint is actually considered to be activated and deactivated. Since hysteresis is a band, technically there can be a setting for the high side of the setpoint and the low side. Your home thermostat has just such a band (our example had 1 degree on each side). In the RCM software, hysteresis exists on only one side of the setpoint. In the figure to the right, the dashed lines represent the hysteresis band on the "normal side" of the setpoint (the side towards what would be a normal value between the warning setpoints).

The setpoint is activated as soon as the parameters exceeds the setpoint value (solid line). If Phase amps high warning amps has a setpoint of 14.5, then the setpoint will trigger at 14.51 (or whatever the smallest resolution of the measurement is). The setpoint is deactivated when the value recedes towards normal and crosses the hysteresis line (dashed line). Where is that line exactly? That's adjustable by the PDU administrator. If the attached equipment has large current swings, the hysteresis might need to be 1/2 amp, or even 1 amp. Equipment with very narrow operating ranges, or clear stepped increments, may be able to use 0.2 amps. The tighter the hysteresis, the more potential for extra alerts. Finding an ideal setting will take exeprimentation.

Debounce

Even with a hysteresis setting, it's possible with our Phase example above that a load could oscillate just above the setpoint, and just below the hysteresis point. Even when both a setpoint value and hysteresis value are selected where that's



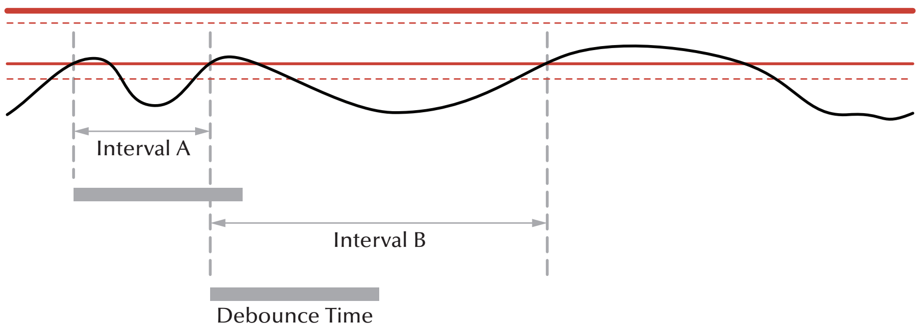
not likely, it is possible. To fend off repetitive alert activations that hysteresis alone does not prevent, we have another adjustment called debounce to use.

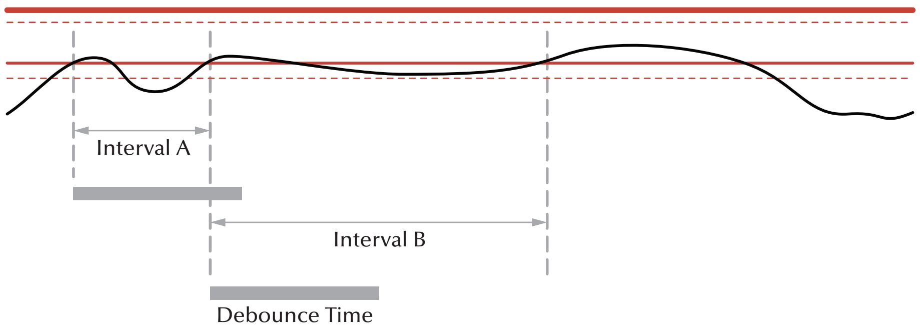
Where hysteresis concerns itself with how far away the measured value is from the setpoint value, debounce concerns itself with how long it has been since the setpoint or value was crossed. In the figure to the right, the measured value (the black wavy line) exceeds the high warning setpoint three times. Debounce concerns itself with the time interval between these three points. The first crossing would trigger an alarm. However, the value recedes and crosses the line again. These first two crossings at Interval A happen in less time than the Debounce Time (the gray bar).

The RCM software considers multiple crossings within the debounce time as a single event. Each time a setpoint is crossed, a time value is noted. If the measured value recedes and crosses again before the debounce time expires, the second crossing is not considered a new alarm. The measured value could oscillate around the setpoint, but so long as each oscillation is shorter than the debounce time, the system determines those are really the same event, and there's no point in generating a new alarm (and therefore not a new email alert) for the second crossing.

However, in the top illustration, the interval between the second and third crossings is longer than the debounce time, so those two crossing are considered to be separate events. There will be an additional alarm for the third crossing.

There's one final scenario to consider. Let's take an almost identical measured value illustration as before. However, between the second and third crossing, let's assume the value doesn't drop low enough to cross the hysteresis line. Would that third crossing generate a new alarm? It would not. The measured value must drop below the hysteresis band to clear the alarm, even though Interval B is longer than Debounce Time. Both hysteresis and debounce must be cleared in order to clear the alarm. In this illustration, the entire view shown is a single alarm started at the first crossing.





Optima RCM Software

User Management

User Management

An RCM PDU 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, login count, and more).

A PDU 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, command line, or through the serial dialog.

If the password is forgotten, and there is no other user with the permission to edit user passwords, the root password will have to be changed through the startup dialog on the serial port as described in "Change root User Password" on page 11.

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 a 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.

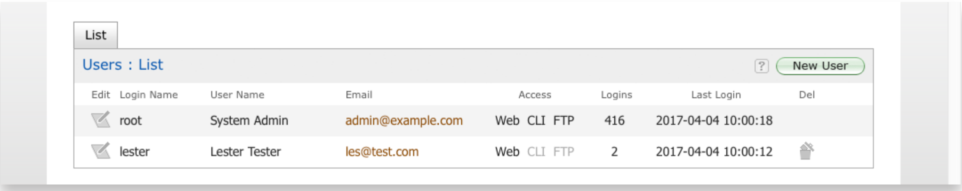
Forgotten Passwords

Given the limited resources of an embedded environment, password recovery techniques are also limited. However, to provide at least some aid to users who have forgotten/misplaced their password, the Password Hint field allows the user to enter some description to help their recall. Of course, this could be misused to store the actual password, and since the software does not know the password, there's little which can be done to effectively prevent that misuse.

Upon entering their account email address in the forgotten password form of the web interface (linked to from the login page), the hint will be emailed to that user. If that is ineffective, the user will have to contact the PDU admin to change the password.

Passwords in the Settings File

The settings file stores passwords salted and hashed. It is not possible to simply edit a user's password in the settings file. To generate a salt and hash for the settings file, use the command line command `makeLoginPswd`. See "makeLoginPswd" on page 101 for details.



Users : List

Edit	Login Name	User Name	Email	Access	Logins	Last Login	Del
	root	System Admin	admin@example.com	Web CLI FTP	416	2017-04-04 10:00:18	
	lester	Lester Tester	les@test.com	Web CLI FTP	2	2017-04-04 10:00:12	

Authorization

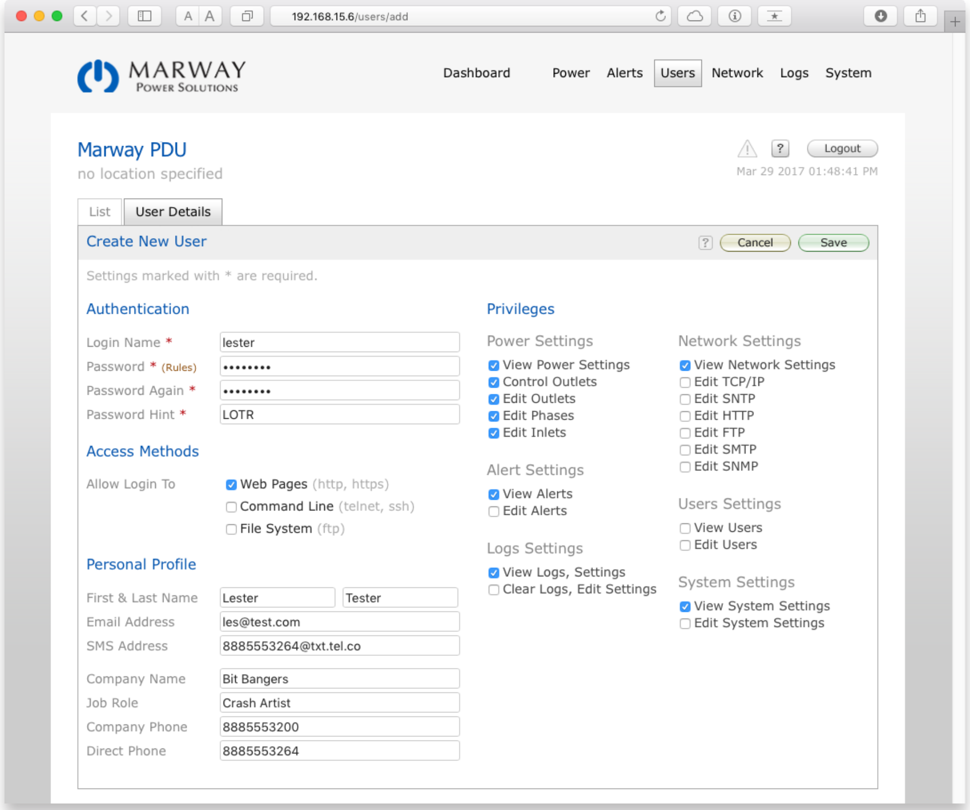
Authorization deals with allowing the user to see, utilize, or edit specific data in or features of the software. In the RCM software, authorization includes whether the user in enabled to use one or more protocols to log into the PDU, and a specific set of selected permissions once logged in.

Access Methods

The root user is inherently allowed to use all login protocols. All other users can be enabled to use the web interface (which includes RESTful API), the command line (Telnet and SSH), and the file system (FTP). Disabling all access methods essentially disables a user without having to delete the record.

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



Marway PDU

no location specified

Mar 29 2017 01:48:41 PM

Create New User

Settings marked with * are required.

Authentication

Login Name *	lester
Password * (Rules)	••••••••
Password Again *	••••••••
Password Hint *	LOTR

Access Methods

Allow Login To
☑ Web Pages (http, https)
□ Command Line (telnet, ssh)
□ File System (ftp)

Personal Profile

First & Last Name	Lester Tester
Email Address	les@test.com
SMS Address	8885553264@txt.tel.co
Company Name	Bit Bangers
Job Role	Crash Artist
Company Phone	8885553200
Direct Phone	8885553264

Privileges

Power Settings
☑ View Power Settings
☑ Control Outlets
☑ Edit Outlets
☑ Edit Phases
☑ Edit Inlets

Alert Settings
☑ View Alerts
□ Edit Alerts

Logs Settings
☑ View Logs, Settings
□ Clear Logs, Edit Settings

Network Settings
☑ View Network Settings
□ Edit TCP/IP
□ Edit SNTP
□ Edit HTTP
□ Edit FTP
□ Edit SMTP
□ Edit SNMP

Users Settings
□ View Users
□ Edit Users

System Settings
☑ View System Settings
□ Edit System Settings

access to a PDU, the assignment of explicit permissions uniquely to each user allows for more control. The exact available permissions may vary based on the available features of any given PDU model.

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

The RCM software keeps track of several user-triggered activities, and is able to log those activities. Actions triggered through the web interface are stamped with user identification. Actions triggered through the other interfaces are stamped with the originating interface (CLI, SNMP), but are not identified with the user.

Logged events include user login, settings changes, outlet switching (unless specifically disabled see "Outlet State Change Logging" on page 55), and invoking restarts via software.

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.

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 101.

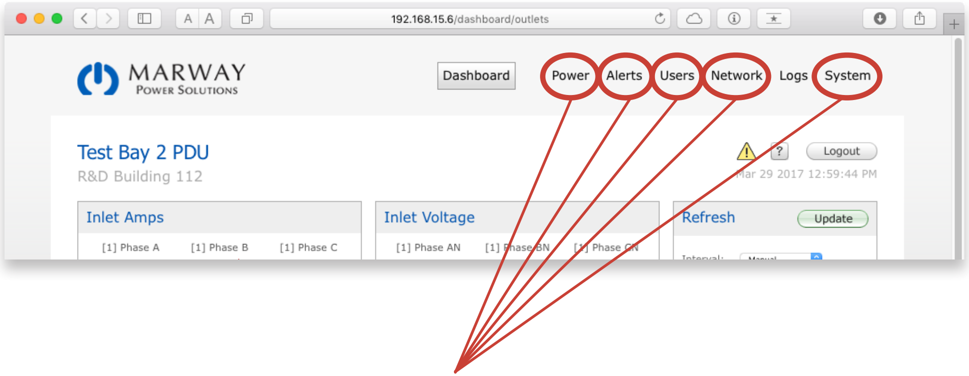
Optima RCM Software

Software Settings

Software Settings

Adjustable values for alarms, alerts, network protocols, etc. are called settings, and are accessible through the web UI and command line. The can also be exported in files accessed through FTP, to be imported to other PDUs.

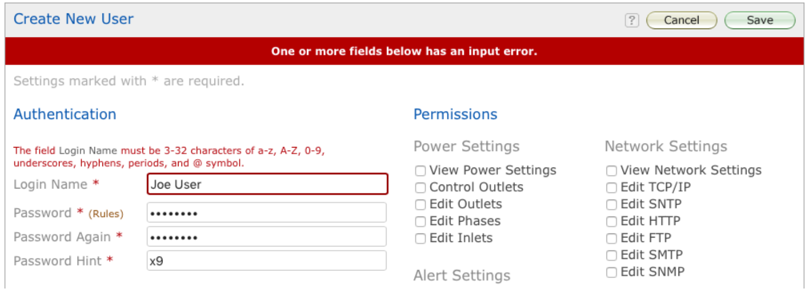
In this chapter, the purpose and acceptable values of each setting will be identified. Additionally, how to use the web UI, CLI, and files to edit them will also be covered.



Use these tabs to edit Settings

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, many fields will display a range and/or default value tooltip by hovering the browser pointer over the field and pausing for a short time.



Create New User

One or more fields below has an input error.

Settings marked with * are required.

Authentication

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

Login Name * Joe User

Password * (Rules) ••••••••

Password Again * ••••••••

Password Hint * x9

Permissions

Power Settings

□ View Power Settings
□ Control Outlets
□ Edit Outlets
□ Edit Phases
□ Edit Inlets

Alert Settings

Network Settings

□ View Network Settings
□ Edit TCP/IP
□ Edit SNTP
□ Edit HTTP
□ Edit FTP
□ Edit SMTP
□ Edit SNMP

Adjust Settings with the CLI

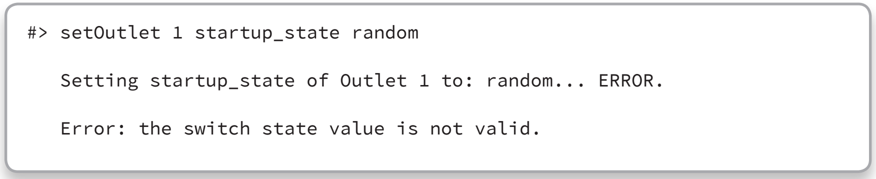
Settings are adjusted on the command line using various `set` commands specific to the setting being set (e.g. `setOutlet` or `setUser`). If an invalid value is submitted, the CLI will usually offer some text as to the valid entry requirements, and at least provide feedback that the value was not valid. Note that the CLI can be used to set multiple objects to the same value. For example:

```
setOutlet startup_state off
```

will set `startup_state` for every outlet whereas

```
setOutlet 1 startup_state last_known
```

will set the `startup_state` for outlet 1 only. See “Command Syntax” on page 91 for other command line options.

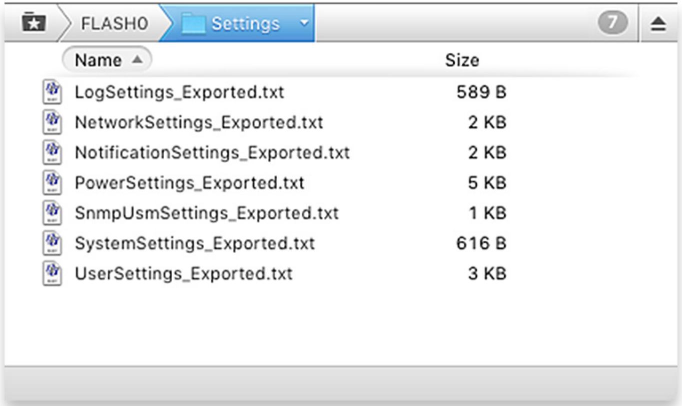


```
#> setOutlet 1 startup_state random

   Setting startup_state of Outlet 1 to: random... ERROR.

   Error: the switch state value is not valid.
```

Exporting and Importing Settings Files

Settings may be exported to text files. This is primarily intended to document and archive the setup of the PDU. However, exported files may be imported to a PDU to restore settings, or even to transfer settings to multiple PDUs.

Use this chapter as the reference for each setting name and value, but refer to “Settings Files Reference” on page 118 for details about the formatting of each settings file.



Name	Size
LogSettings_Exported.txt	589 B
NetworkSettings_Exported.txt	2 KB
NotificationSettings_Exported.txt	2 KB
PowerSettings_Exported.txt	5 KB
SnmpUsmSettings_Exported.txt	1 KB
SystemSettings_Exported.txt	616 B
UserSettings_Exported.txt	3 KB

Configuring Multiple PDUs

If there are multiple PDUs to be configured with identical or very similar settings, there are two approaches to make that process easier than configuring each one manually.

First, one PDU could be configured, the settings exported, then imported to additional units. See “Settings Files Reference” on page 118 for more details.

Second, a series of CLI commands can be automated through Telnet or SSH with a scripting language (Ruby, Python, PERL, or others). The script can send all configuration details. Use “Command Line Reference” on page 91 or the built-in CLI help as a reference.

System Settings

System settings are for items universal to the whole PDU. Refer to the web UI view in "System Settings" on page 26.

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 getSystem CLI command.	setSystem label	SystemSettings.txt label =	Text	All English letters, numerals, and punctuation except double quotes and the tilde.
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 getSystem CLI command.	setSystem location	SystemSettings.txt location =	Text	All English letters, numerals, and punctuation except double quotes and the tilde.
Asset ID	A label intended to display an ID assigned by a corporate asset accounting system.	setSystem asset_id	SystemSettings.txt assetId =	Text	All English letters, numerals, and punctuation except double quotes and the tilde.
Contact	An arbitrary label intended to include a person or department name and contact phone or email. This is also used for SNMP's sysContact field.	setSystem contact	SystemSettings.txt contact =	Text	All English letters, numerals, and punctuation except double quotes and the tilde.
Time	Date and time values for when the PDU is not connected to a networked time server (SNTP).	setSystem time	(none)	Time	YYYY-MM-YY HH:MM:SS (24 hr)
Temperature Units	The choice of °F or °C for temperature probe data.	setSystem temp_units	SystemSettings.txt temperatureUnits =	Text	Farenheit or Celcius or even just F or C (case insensitive)
Dialog Baud Rate	The data rate for the serial console port.	setSystem dialog_baudrate	SystemSettings.txt dialogBaudRate =	Integer	1200, 2400, 4800, 9600, 19200, 38400, 57600, or 115200
Dialog Delay	The delay in seconds the startup process waits for a user to press a ket to enter the setup dialog.	setSystem dialog_delay	SystemSettings.txt dialogDialogDelay =	Integer	5 or 10 (seconds)

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 web UI image in "Network Settings" on page 25.

Setting	Description	CLI Command / Attribute	File / Setting	Type	Validation
IPv4 DHCP Enabled	Deteremines whether the TCP/IPv4 address will be supplied by a DHCP server ("on") or entered manually ("off").	setTcp ipv4_dhcp	NetworkSettings.txt (tcpip) v4Dhcp =	Text	"true" or "false"
IPv4 Address	The TCP/IPv4 address. Leave empty for DHCP.	setTcp ipv4_address	NetworkSettings.txt (tcpip) v4Address =	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 (tcpip) v4Subnet =	Text	0.0.0.0 format
IPv4 Gateway	The TCP/IPv4 subnet gateway address.	setTcp ipv4_gateway	NetworkSettings.txt (tcpip) v4Gateway =	Text	0.0.0.0 format
IPv6 DHCP Enabled	Deteremines whether the TCP/IPv6 address will be supplied by a DHCP server ("on") or entered manually ("off").	setTcp ipv6_dhcp	NetworkSettings.txt (tcpip) v6Dhcp =	Text	"true" or "false"
IPv6 Address	The TCP/IPv6 address. Leave empty for DHCP.	setTcp ipv6_address	NetworkSettings.txt (tcpip) v6Address =	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 (tcpip) v6PrefixLength =	Integer	0–128 (default = 64)
HTTP Enabled	Indicates whether the internal Web UI server should be enabled.	setHttp webui_enabled	NetworkSettings.txt (http) enabled =	Text	"true" or "false"
HTTP Port	The TCP port for the service.	setHttp port	NetworkSettings.txt (http) port =	Integer	1–65535 (default = 80)
SSL Enabled	Indicates whether HTTPS should be used instead of HTTP (only one runs at a time).	setHttps enabled	NetworkSettings.txt (http) sslEnabled =	Text	"true" or "false"
SSL Port	The TCP port for the service.	setHttps port	NetworkSettings.txt (http) sslPort =	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 (http) sessionMins =	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 (sntp) serverIp1 =	Text	0.0.0.0 format
SNTP Server 2 IP	The TCP/IPv4 address of a network time server.	setSntp server2	NetworkSettings.txt (sntp) serverIp2 =	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 (sntp) syncIntervalHrs =	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 (sntp) stdOffsetHrs =	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 GMT Offset	Defines the local daylight savings time offset in hours from Greenwich Mean Time (GMT).	setSntp dst_offset	NetworkSettings.txt (sntp) dstOffsetHrs =	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 Starts	Defines when daylight savings time starts.	setSntp dst_starts	NetworkSettings.txt (sntp) dstStartsAt =	Text	Enter format like 11.1.0/02:00:00 (month. ordinal of week.weekday/ hour:min:sec.
Timezone DST Ends	Defines when daylight savings time ends.	setSntp dst_ends	NetworkSettings.txt (sntp) dstEndsAt =	Text	Enter format like 11.1.0/02:00:00 (month. ordinal of week.weekday/ hour:min:sec.
FTP Enabled	Determines whether FTP is enabled or not.	setFtp enabled	NetworkSettings.txt (ftp) enabled =	Text	“true” or “false”
FTP Port	The TCP port for the service.	setFtp port	NetworkSettings.txt (ftp) port =	Integer	1–65535 (default = 21)
SMTP Server IP	The TCP/IPv4 address of an SMTP server.	setSmtp server	NetworkSettings.txt (smtp) serverIp =	Text	0.0.0.0 format
SMTP Port	The TCP port for the service.	setSmtp port	NetworkSettings.txt (smtp) port =	Integer	1–65535 (default = 25)
SMTP Authentication Method	Determines how the account and password are communicated to the server.	setSmtp auth_method	NetworkSettings.txt (smtp) authMethod =	Text	One of these values: “Any,” “Plain,” “Login,” “CRAM-MD5,” “DIGEST-MD5”
SMTP Account	The SMTP account user name which is authorized to send emails through the server.	setSmtp account	NetworkSettings.txt (smtp) account =	Text	Usually an email address, but may another value depending on the server.

Setting	Description	CLI Command / Attribute	File / Setting	Type	Validation
SMTP Password	The password for the SMTP account user name which is authorized to send emails through the server.	setSmtp password	NetworkSettings.txt (smtp) password =	Text	All English letters, numerals, and punctuation except double quotes and the tilde.
SNMP Enabled	Determines whether SNMP is enabled or not.	setSnmp enabled	NetworkSettings.txt (snmp) enabled =	Text	“true” or “false”
SNMP Data Port	The TCP port for reading data values.	setSnmp data_port	NetworkSettings.txt (snmp) dataPort =	Integer	1–65535 (default = 161)
SNMP Community 1	The community string for read-only access to SNMP v1 and v2	setSnmp community1_string	NetworkSettings.txt (snmp) community1 =	Text	All English letters, numerals.
SNMP Community 2	The community string for read-write access to SNMP v1 and v2	setSnmp community2_string	NetworkSettings.txt (snmp) community2 =	Text	All English letters, numerals.
SNMP Trap Server IP	The IP address of the server to send Traps to.	setSnmp trap_ip	NetworkSettings.txt (snmp) trapServerIp =	Text	0.0.0.0 format
SNMP Trap Port	The TCP port for the trap server.	setSnmp trap_port	NetworkSettings.txt (snmp) trapPort =	Integer	1–65535 (default = 162)
SNMP Trap Community String	The community string for to access the trap server.	setSnmp trap_community	NetworkSettings.txt (snmp) trapCommunity =	Text	All English letters, numerals.
SNMP Notifications User Name	The SNMP v3 notifications user name.	setSnmp notifications_user	NetworkSettings.txt (snmp) notifUserName =	Text	All English letters, numerals.

Setting	Description	CLI Command / Attribute	File / Setting	Type	Validation
USM User Name	SNMP v3 USM user name.	setSnmpUsm [id] name	SnmpUsmSettings.txt (usm) name =	Text	All English letters, numerals.
USM Authentication Method	SNMP v3 USM user authentication method.	setSnmpUsm [id] auth_method	SnmpUsmSettings.txt (usm) authMethod =	Text	“none” or “md5”
USM Authentication Passphrase	SNMP v3 USM user authentication passphrase.	setSnmpUsm [id] auth_passphrase	SnmpUsmSettings.txt (usm) authPassphrase =	Text	All English letters, numerals.
USM Privacy Method	SNMP v3 USM user privacy method.	setSnmpUsm [id] priv_method	SnmpUsmSettings.txt (usm) privMethod =	Text	“none” or “md5”
USM Privacy Passphrase	SNMP v3 USM user privacy passphrase.	setSnmpUsm [id] priv_passphrase	SnmpUsmSettings.txt (usm) privPassphrase =	Text	All English letters, numerals.

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. Refer also to "User Management" on page 65.

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 create a new user in the file.	addUser setUser [name] 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
Password Hint	A text value with some meaning for the user which will help them remember their password.	getUser [name] hint setUser [name] hint	UserSettings.txt pswdHint =	Text	All English letters, numerals, and punctuation except double quotes and the tilde.

Setting	Description	CLI Command / Attribute	File / Setting	Type	Validation
First Name	The user’s first name (given name).	getUser [name] profile setUser [name] 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 [name] profile setUser [name] 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 [name] profile setUser [name] company_name	UserSettings.txt companyName =	Text	All English letters, numerals, and punctuation except double quotes and the tilde.
Job Role	A descriptive title or role the user performs for the PDU or for the company he represents.	getUser [name] profile setUser j [name] ob_role	UserSettings.txt jobRole =	Text	All English letters, numerals, and punctuation except double quotes and the tilde.
Company Phone	A general phone number for the company the user represents.	getUser [name] profile setUser [name] 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 [name] profile setUser [name] 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 [name] profile setUser [name] 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 generall in the form of mobile_number@carrier.tld such as 2125551212@txt.att.net	getUser [name] profile setUser [name] 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 [name] permissions setUser [name] login_via_web	UserSettings.txt loginViaWeb =	Text	“true” or “false”

Setting	Description	CLI Command / Attribute	File / Setting	Type	Validation
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 [name] permissions setUser [name] 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 [name] permissions setUser [name] login_via_ftp	UserSettings.txt loginViaFTP =	Text	“true” or “false”
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 [name] permissions setUser [name] view_power	UserSettings.txt viewPowerSettings =	Text	“true” or “false”
Control Outlets	Determines whether the user is allowed to switch outlets on/off.	getUser [name] permissions setUser [name] control_outlets	UserSettings.txt controlOutlets =	Text	“true” or “false”
Edit Outlet Labels	Determines whether the user is allowed to edit outlet settings.	getUser [name] permissions setUser [name] edit_outlets	UserSettings.txt editOutlets =	Text	“true” or “false”
Edit Circuit Labels	Determines whether the user is allowed to edit circuit settings.	getUser [name] permissions setUser [name] edit_circuits	UserSettings.txt editCircuits =	Text	“true” or “false”
Edit Phases	Determines whether the user is allowed to edit phase settings.	getUser [name] permissions setUser [name] edit_phases	UserSettings.txt editPhases =	Text	“true” or “false”
Edit Inlets	Determines whether the user is allowed to edit inlet settings.	getUser [name] permissions setUser [name] edit_inlets	UserSettings.txt editInlets =	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 [name] permissions setUser [name] view_users	UserSettings.txt viewUsers =	Text	“true” or “false”
Edit Users	Determines whether the user is allowed to edit user settings.	getUser [name] permissions setUser [name] edit_users	UserSettings.txt editUsers =	Text	“true” or “false”
View Network Settings	Determines whether the user is allowed to view network protocol settings.	getUser [name] permissions setUser [name] view_network	UserSettings.txt viewNetworkSettings =	Text	“true” or “false”
Edit TCP/IP	Determines whether the user is allowed to edit TCP/IP protocol settings.	getUser [name] permissions setUser [name] edit_tcpip	UserSettings.txt editTCPIP =	Text	“true” or “false”
Edit SNTP	Determines whether the user is allowed to edit SNTP protocol settings.	getUser [name] permissions setUser [name] edit_sntp	UserSettings.txt editSNTP =	Text	“true” or “false”
Edit HTTP	Determines whether the user is allowed to edit HTTP(S) protocol settings.	getUser [name] permissions setUser [name] edit_http	UserSettings.txt editHTTP =	Text	“true” or “false”
Edit FTP	Determines whether the user is allowed to edit FTP protocol settings.	getUser [name] permissions setUser [name] edit_ftp	UserSettings.txt editFTP =	Text	“true” or “false”
Edit SMTP	Determines whether the user is allowed to edit SMTP protocol settings.	getUser [name] permissions setUser [name] edit_smtp	UserSettings.txt editSMTP =	Text	“true” or “false”
Edit SNMP	Determines whether the user is allowed to edit SNMP protocol settings.	getUser [name] permissions setUser [name] edit_snmp	UserSettings.txt editSNMP =	Text	“true” or “false”
View Alerts	Determines whether the user is allowed to view alert settings.	getUser [name] permissions setUser [name] view_alerts	UserSettings.txt viewAlerts =	Text	“true” or “false”
Edit Alerts	Determines whether the user is allowed to edit alert settings.	getUser [name] permissions setUser [name] edit_alerts	UserSettings.txt editAlerts =	Text	“true” or “false”

Setting	Description	CLI Command / Attribute	File / Setting	Type	Validation
View Logs	Determines whether the user is allowed to view log entries.	getUser [name] permissions setUser [name] view_logs	UserSettings.txt viewLogs =	Text	“true” or “false”
Edit Logs	Determines whether the user is allowed to edit log settings.	getUser [name] permissions setUser [name] edit_logs	UserSettings.txt editLogs =	Text	“true” or “false”
View System Settings	Determines whether the user is allowed to view system settings.	getUser [name] permissions setUser [name] view_system	UserSettings.txt viewSystemSettings =	Text	“true” or “false”
Edit System Settings	Determines whether the user is allowed to edit system settings.	getUser [name] permissions setUser [name] edit_system	UserSettings.txt viewSystemSettings =	Text	“true” or “false”

Notification Settings

Notification settings determine who, if anyone, will receive notifications of alarms and certain events, and the form of the notification (e.g. email, SMS, or SNMP trap/notification). Refer also to “Events Management” on page 57.

Setting	Description	CLI Command / Attribute	File / Setting	Type	Validation
Realert Minutes	Determines the re-alert interval. Some alerts (e.g. setpoint excursions) require that they be acknowledged. If not acknowledged, the alert will be re-sent after Realert Minutes has transpired.	setAlertMisc realert	AlertSettings.txt realertMinutes =	Integer	5–60 mins (default 15) 0 = disabled (no re-alert)
Silence Hours	Determines the silence duration. When an alert is acknowledged, re-alerts will be disabled for that alert for the duration indicated. If that duration passes, and the alarm condition continues, re-alert will resume. Disabling Silence, in effect means realert will happen until the alarm subsides regardless of acknowledgments. Disabling Silence would be suitable only if the re-alert intervals are lengthy enough to not be a nuisance.	setAlertMisc silence	AlertSettings.txt silenceHours =	Text	1–24 hours (default 2) 0 = disabled
Unit Triggers	Identifies which general events will trigger an alert.	getAlert unit setAlert unit triggers	AlertSettings.txt (unit) triggers =	Text	Keywords: startup, login, settings, epo
Unit Email Users	A comma separated list of user login names. Each name listed will be emailed an alert if that user has an email address defined.	getAlert unit setAlert unit email_to	AlertSettings.txt (unit) emailToUsers =	Text	User login names
Unit SMS Users	A comma separated list of user login names. Each name listed will be sent an SMS alert if that user has an SMS address defined.	getAlert unit setAlert unit sms_to	AlertSettings.txt (unit) smsToUsers =	Text	User login names

Setting	Description	CLI Command / Attribute	File / Setting	Type	Validation
Unit SendTrap	Defines whether active triggers will send SNMP trap and notification messages (v1, v2, v3 are all sent).	getAlert unit setAlert unit send_snmp	AlertSettings.txt (unit) sendSnmpTrap =	Text	“true” or “false”
Power Triggers	Identifies which power events will trigger an alert. Power applies to all inlet, phase, and circuit objects.	getAlert power setAlert power triggers	AlertSettings.txt (power) triggers =	Text	Keywords: amps, volts
Power Email Users	A comma separated list of user login names. Each name listed will be emailed an alert if that user has an email address defined.	getAlert power setAlert power email_to	AlertSettings.txt (power) emailToUsers =	Text	“true” or “false”
Power SMS Users	A comma separated list of user login names. Each name listed will be sent an SMS alert if that user has an SMS address defined.	getAlert power setAlert power sms_to	AlertSettings.txt (power) smsToUsers =	Text	“true” or “false”
Power SendTrap	Defines whether active triggers will send SNMP trap and notification messages (v1, v2, v3 are all sent).	getAlert power setAlert power send_snmp	AlertSettings.txt (power) sendSnmpTrap =	Text	“true” or “false”
Outlet Triggers	Identifies which power events will trigger an alert. Power applies to all inlet, phase, and circuit objects.	getAlert outlet [id] setAlert outlet [id] triggers	AlertSettings.txt (outlet) triggers =	Text	Keywords: switch
Outlet Email Users	A comma separated list of user login names. Each name listed will be emailed an alert if that user has an email address defined.	getAlert outlet [id] setAlert outlet [id] email_to	AlertSettings.txt (outlet) emailToUsers =	Text	“true” or “false”
Outlet SMS Users	A comma separated list of user login names. Each name listed will be sent an SMS alert if that user has an SMS address defined.	getAlert outlet [id] setAlert outlet [id] sms_to	AlertSettings.txt (outlet) smsToUsers =	Text	“true” or “false”
Outlet SendTrap	Defines whether active triggers will send SNMP trap and notification messages (v1, v2, v3 are all sent).	getAlert outlet [id] setAlert outlet [id] send_snmp	AlertSettings.txt (outlet) sendSnmpTrap =	Text	“true” or “false”

Power Settings

Power settings include labels, setpoints, and in the case of outlets, some items specific to switching behavior. Not every device has every possible setting. Refer also to the web UI images in "Power Settings" on page 20.

Labels

The RCM software defaults to labeling each power device by its device name (Outlet, Circuit, Inlet) along with a sequential numerical identification. This results in default labels such as Outlet 1, Outlet 2, Circuit 1, Circuit 2, etc.

To allow labels with more meaningful descriptions, the labels for Outlets, Circuits, and Inlets can be changed. The labels for Phases are not editable. They remain fixed with their associated conductor labels (A, B, C, N).

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 Control" on page 45. 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.

Alarm Setpoints, Hysteresis, Debounce

These items determine the alarm behavior for all power devices. Understanding alarm management can be a complex topic. The section "Power Setpoints and Alarms" on page 60 provides some reference information. Assistance from someone experienced in alarm management may also be suitable if needed.

For the setting adjustments identified in the table below, setpoint values will be in the units of the measurement being monitored such as amps or volts. Hysteresis values are also in the measured units, whereas Debounce is entered in units of seconds.

One thing to be aware of is the difference between zero and disabled. A setpoint setting of zero means an alarm will trigger using that as a comparison value. Disabled means there is no alarm. On the web interface, disabled is an empty field, and on the command line an empty value or the word "disabled" is used with a `set` command.

Setpoint Accuracy and Limits

It is important to keep in mind certain practical limits of setpoints at the extreme ends of their theoretical ranges. For amps, the software will allow a value from zero to the Maximum Rating value of the device (refer to the PDU electrical specifications for detailed values). However, it is not practical to have setpoint settings at the extreme ends of this range. First, all sensors, like those being used to measure the amps and volts within the PDU, lose accuracy at their lower and upper limits. How close to those limits is practical? This can be affected by the built-in tolerances of the electronics, and by environmental factors such as temperature. There is not a single value that will always work at all times, and in all PDUs even of the same model.

For RCM PDUs with power monitoring, under ideal conditions, detecting the difference between some current and zero current is less than a tenth of an amp. This means that a Low Warning Amps setting could drop as low as 0.1A and still be considered not zero by the RCM software. Feel free to experiment, but understand that monitoring equipment at such low limits is not the primary intent of the product.

For RCM PDUs with current monitoring, the lowest detectable limit is aproximately 0.3 amps.

At the upper limit, a more important factor to be aware of than accuracy is the legal impact of local electrical safety laws. Within the USA, Canada, Europe, and many other areas of the world, there is a legal requirement that continuous duty electrical loads must not exceed 80% of the Maximum Rating of the circuit devices. For example, an outlet, fuse, or circuit breaker with a 15 amp Maximum Rating cannot have more than 12 amps (15 amps x 80%) of continuous duty load. “Continuous duty” is defined in the National Electrical Code (of the USA) as being 3 hours and longer. Therefore, for current, a practical maximum limit for the High Warning Amps setpoint is always going to be 80% of the Maximum Rating value of the device being measured (80% of the receptacle rating for outlets, 80% of the circuit breaker rating for circuits, and 80% of the plug rating for Inputs). However, a more pragmatic and informative limit is going to be closer to what you would expect the normal operating current to be.

It is legal to run closer to the example 15 amp limit for “intermittent” periods (which are not well defined). Therefore, the High Critical Amps setpoint could be set closer to the Maximum Rating value. At that point, the inherent loss of accuracy at the limits becomes a factor. Relying on a value within only a few tenths of an amp to the limit may not have the accuracy and repeatability desired for reliable alarm management. Circuit breakers too have some variability. Therefore, trying to get very close to the limit of the breakers may not yield consistent results either—some breakers may trip sooner than others.

The exact legal requirements and definitions of electrical safety code in your locale should be understood by anyone installing and operating this equipment.

As stated earlier, while the software may allow setpoint ranges of X to Y, best practices result in values much closer to the operating range of the equipment, not the range of the power circuit.

Setting	Description	CLI Command / Attribute	File / Setting	Type	Validation
Inlet Label	A label to identify the power inlet (defaults to "Power Inlet 1").	getInlet [id] setInlet [id] label	PowerSettings.txt (inlet) label =	Text	All English letters, numerals, and punctuation except double quotes and the tilde.
Phase High Critical Amps	Defines the High Critical setpoint of the measured value, which for inputs is amps.	getPhase [id] settings setPhase [id] high_critical_amps	PowerSettings.txt (inlet) highCriticalAmps =	Decimal	0 up to the Maximum Rating value of the phase. To disable an alarm, set it to empty (not zero) or the keyword "disabled"
Phase High Warning Amps	Defines the High Warning setpoint of the measured phase amps.	getPhase [id] settings setPhase [id] high_warning_amps	PowerSettings.txt (inlet) highWarningAmps =	Decimal	0 up to the Maximum Rating value of the phase. To disable an alarm, set it to empty (not zero) or the keyword "disabled"
Phase Low Warning Amps	Defines the Low Warning setpoint of the measured phase amps.	getPhase [id] settings setPhase [id] low_warning_amps	PowerSettings.txt (inlet) lowWarningAmps =	Decimal	0 up to the Maximum Rating value of the phase. To disable an alarm, set it to empty (not zero) or the keyword "disabled"
Phase Low Critical Amps	Defines the Low Critical setpoint of the measured phase amps.	getPhase [id] settings setPhase [id] low_critical_amps	PowerSettings.txt (inlet) lowCriticalAmps =	Decimal	0 up to the Maximum Rating value of the phase. To disable an alarm, set it to empty (not zero) or the keyword "disabled"
Phase Amps Hysteresis	Defines the setpoint hysteresis in amps for phase amps.	getPhase [id] settings setPhase [id] hysteresis_amps	PowerSettings.txt (inlet) hysteresisAmps =	Decimal	0.1 to 2.0 amps
Phase Amps Debounce	Defines the setpoint debounce in seconds for phase amps.	getPhase [id] settings setPhase [id] debounce_amps	PowerSettings.txt (inlet) debounceAmps =	Decimal	Up to 60 seconds. Lowest setting is hardware dependent.

Setting	Description	CLI Command / Attribute	File / Setting	Type	Validation
Phase High Critical Volts	Defines the High Critical setpoint of the measured value, which for inputs is volts.	getPhase [id] settings setPhase [id] high_critical_volts	PowerSettings.txt (inlet) highCriticalVolts =	Decimal	0 up to the Maximum Rating value of the phase. To disable an alarm, set it to empty (not zero) or the keyword "disabled"
Phase High Warning Volts	Defines the High Warning setpoint of the measured phase volts.	getPhase [id] settings setPhase [id] high_warning_volts	PowerSettings.txt (inlet) highWarningVolts =	Decimal	0 up to the Maximum Rating value of the phase. To disable an alarm, set it to empty (not zero) or the keyword "disabled"
Phase Low Warning Volts	Defines the Low Warning setpoint of the measured phase volts.	getPhase [id] settings setPhase [id] low_warning_volts	PowerSettings.txt (inlet) lowWarningVolts =	Decimal	0 up to the Maximum Rating value of the phase. To disable an alarm, set it to empty (not zero) or the keyword "disabled"
Phase Low Critical Volts	Defines the Low Critical setpoint of the measured phase amps. volts	getPhase [id] settings setPhase [id] low_critical_volts	PowerSettings.txt (inlet) lowCriticalVolts =	Decimal	0 up to the Maximum Rating value of the phase. To disable an alarm, set it to empty (not zero) or the keyword "disabled"
Phase Vots Hysteresis	Defines the setpoint hysteresis in volts for phase volts.	getPhase [id] settings setPhase [id] hysteresis_volts	PowerSettings.txt (inlet) hysteresisVolts =	Decimal	0.5 to 20.0 volts
Phase Volts Debounce	Defines the setpoint debounce in seconds for phase volts.	getPhase [id] settings setPhase [id] debounce_volts	PowerSettings.txt (inlet) debounceVolts =	Decimal	Up to 60 seconds. Lowest setting is hardware dependent.

Setting	Description	CLI Command / Attribute	File / Setting	Type	Validation
Outlet Label	A label to identify the outlet (defaults to “Outlet 1”).	getOutlet [id] settings setOutlet [id] label	PowerSettings.txt (outlet) label =	Text	All English letters, numerals, and punctuation except double quotes and the tilde.
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 (applies only when more than one outlet is being switched).	getOutlet [id] settings setOutlet [id] on_delay	PowerSettings.txt (outlet) 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 (applies only when more than one outlet is being switched).	getOutlet [id] settings setOutlet [id] off_delay	PowerSettings.txt (outlet) 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 [id] settings setOutlet [id] cycle_delay	PowerSettings.txt (outlet) 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 [id] settings setOutlet [id] startup_state	PowerSettings.txt (outlet) startupState =	Text	One of the keywords: “on,” “off,” or “last known”
Outlet IsAlertable	Determines whether the switching action of an outlet will generate an alert event. Configuration of outlet alerts is still required to define who gets the alertas and how.	getOutlet [id] settings setOutlet [id] alert_switch_change	PowerSettings.txt (outlet) switchIsAlertable =	Text	“true” or “false”

Log Settings

Log settings determine how many log records are recorded, and which APIs to log outlet switch events from.

Setting	Description	CLI Command / Attribute	File / Setting	Type	Validation
Max Records	Defines how many log records to maintain. More records increases history, but adds to startup time.	setLog max_records	LogSettings.txt masRecord =	Integer	0 to 1000
Log CLI Switching	Detemines whether switching events triggered through the CLI are logged.	setLog cli_switching	LogSettings.txt logCLISwitching =	Text	“true” or “false”
Log RESTful Switching	Detemines whether switching events triggered through the RESTful API are logged.	setLog rest_switching	LogSettings.txt logRESTfulSwitching =	Text	“true” or “false”
Log SNMP Switching	Detemines whether switching events triggered through SNMP are logged.	setLog snmp_switching	LogSettings.txt logSNMPSwitching =	Text	“true” or “false”

Optima 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.

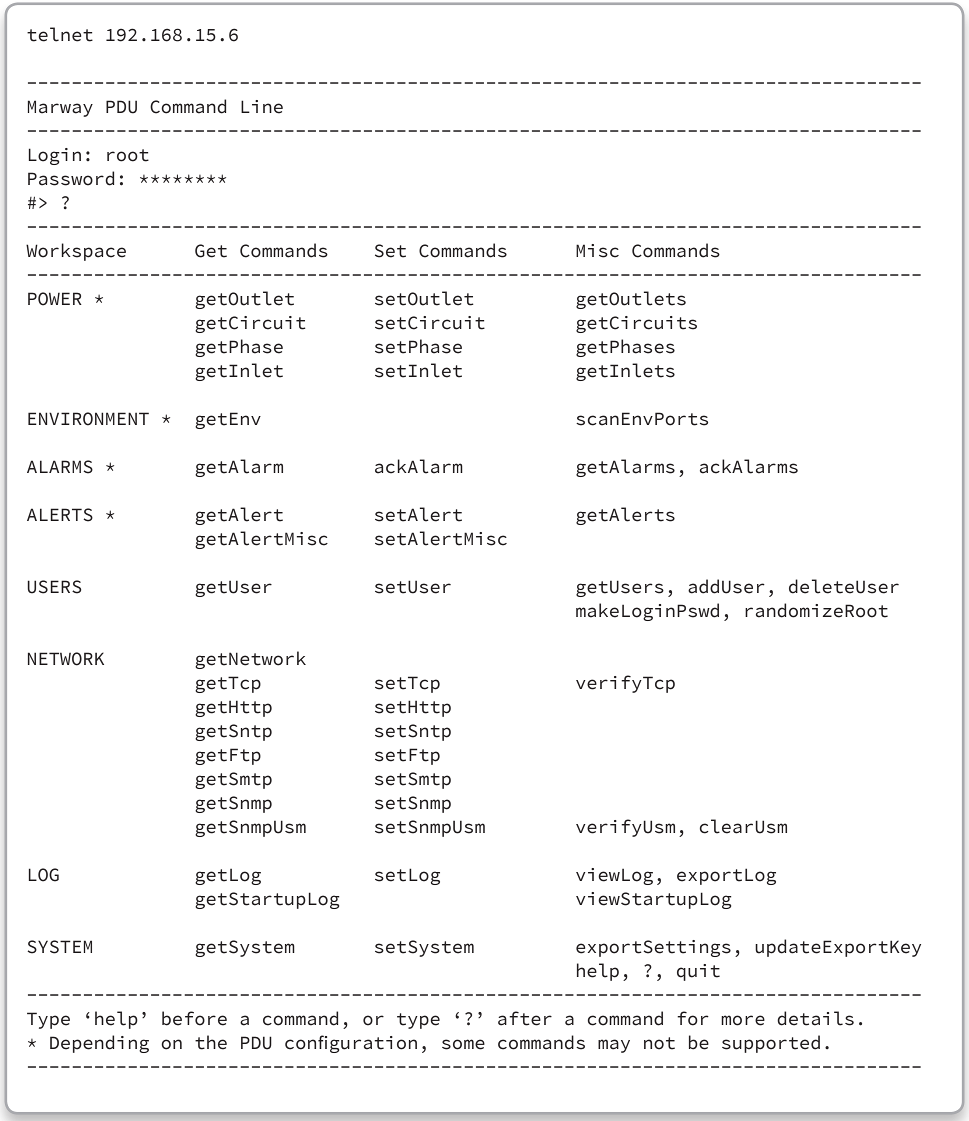
Each command can also be followed by a `?` such as `getSystem?` (with or without a space) to display detailed usage help for that command. The command help will identify the syntax of the command, various attribute options, and some examples.

Typing ? after a command name will redisplay the command after the help info, ready to accept the rest of the command. Typing help before the command will display the help only.

Command Syntax

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

- `getOutlet 8 switch`
- `getPhase 1 high_critical_amps`
- `getUser lester email`
- `getSystem location`



```
telnet 192.168.15.6

--------------------------------------------------------------------------------
Marway PDU Command Line
--------------------------------------------------------------------------------
Login: root
Password: ********
#> ?
--------------------------------------------------------------------------------
Workspace      Get Commands     Set Commands      Misc Commands
--------------------------------------------------------------------------------
POWER *        getOutlet        setOutlet         getOutlets
               getCircuit       setCircuit        getCircuits
               getPhase         setPhase          getPhases
               getInlet         setInlet          getInlets

ENVIRONMENT *  getEnv                             scanEnvPorts

ALARMS *       getAlarm         ackAlarm          getAlarms, ackAlarms

ALERTS *       getAlert         setAlert          getAlerts
               getAlertMisc     setAlertMisc

USERS          getUser          setUser           getUsers, addUser, deleteUser
                                                  makeLoginPswd, randomizeRoot

NETWORK        getNetwork
               getTcp           setTcp            verifyTcp
               getHttp          setHttp
               getSntp          setSntp
               getFtp           setFtp
               getSmtp          setSmtp
               getSnmp          setSnmp
               getSnmpUsm       setSnmpUsm        verifyUsm, clearUsm

LOG            getLog           setLog            viewLog, exportLog
               getStartupLog                      viewStartupLog

SYSTEM         getSystem        setSystem         exportSettings, updateExportKey
                                                  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`
- `setCircuit 1 high_critical_amps 14.6`
- `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`, `setPhase`, etc. Command names are not case sensitive. The `instanceID` is going to be the numerical ID of a power device such as an outlet, or phase, 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. Attributes names are not case sensitive. Finally, if the command is a `set`, the fourth parameter is the new value. Values will be stored with the same case they are entered with.

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 `[object]` in the example help display below:

```
#> getOutlet ?

------------------------------------------------------------------
   Syntax:  getOutlet [id] [attribute]
```

Power Commands

getInlet

```
-------------------------------------------------------------------------------
   Syntax:  getInlet [id] [attribute|group]

   Where:
     []             : indicates an optional parameter
     id             : is the number of the Inlet (e.g. 1 or 3, etc.)
                      or leave it empty to refer to all Inlets
     attribute      : is one of the attributes below (or its alias):
     group          : is one of the attribute groups below (or its alias):

   Attributes:
     label                  --
     phase                  --

   Attribute Groups:
     rating                 --

   Examples:
     getInlet 3 phase       : gets one attribute for one Inlet
     getInlet 2             : gets all attributes for one Inlet
     getInlet phase         : gets one attribute for all Inlets
     getInlet               : gets all attributes for all Inlets
-------------------------------------------------------------------------------
```

setInlet

```
-------------------------------------------------------------------------------
   Syntax:  setInlet [id] attribute value

   Where:
     []             : indicates an optional parameter
     id             : is the number of the Inlet (e.g. 1 or 3, etc.)
                      or leave it empty to refer to all Inlets
     attribute      : is one of the following (or its alias):

       label                  --

     value for label          : is a quoted string (30 chars max)

   Examples:
     setInlet 1 label "Power Bank A"      : sets one label
     setInlet label default               : resets all to the default value
-------------------------------------------------------------------------------
```

getPhase

```
-------------------------------------------------------------------------------
  Syntax:  getPhase [id] [attribute|group]

   Where:
     []             : indicates an optional parameter
     id             : is the number of the Phase (e.g. 1 or 3, etc.)
                      or leave it empty to refer to all Phases
     attribute      : is one of the attributes below (or its alias):
     group          : is one of the attribute groups below (or its alias):

   Attributes:
     phase                      --
     rated_volts                (rv, ratedvolts)
     rated_amps                 (ra, ratedamps, max_amps, maxamps)
     amps                       (a, amps_rms, ampsrms, arms)
     consumed_amps              (ca, amps_used, amps)
     volts                      (v, volts_rms, voltsrms, vrms)
     frequency                  (f, freq)
     voltamps                   (va, kva)
     voltamps_reactive          (var, kvar)
     watts                      (w, kw)
     power_factor               (pf, powerfactor)
     high_critical_amps         (hca)
     high_warning_amps          (hwa)
     low_warning_amps           (lwa)
     low_critical_amps          (lca)
     hysteresis_amps            (hysa)
     debounce_amps              (dba)
     high_critical_volts        (hcv)
     high_warning_volts         (hwv)
     low_warning_volts          (lwv)
     low_critical_volts         (lcv)
     hysteresis_volts           (hysv)
     debounce_volts             (dbv)

   Attribute Groups:
     rating                     --
     power_info                 (power, pwr)
     setpoints_info             (setpoint_info)
     settings                   --

   Examples: {not shown}
-------------------------------------------------------------------------------
```

setPhase

```
-------------------------------------------------------------------------------
   Syntax:  setPhase [id] attribute value

   Where:
     []             : indicates an optional parameter
     id             : is the number of the Phase (e.g. 1 or 3, etc.)
                      or leave it empty to refer to all Phases
     attribute      : is one of the following (or its alias):

       high_critical_amps      (hca)
       high_warning_amps       (hwa)
       low_warning_amps        (lwa)
       low_critical_amps       (lca)
       hysteresis_amps         (hysa)
       debounce_amps           (dba)
       high_critical_volts     (hcv)
       high_warning_volts      (hwv)
       low_warning_volts       (lwv)
       low_critical_volts      (lcv)
       hysteresis_volts        (hysv)
       debounce_volts          (dbv)

     value for setpoints      : is a decimal, an empty quoted
                                string "" to disable it, or: disabled
     value for hysteresis     : is a decimal, or: default
     value for debounce       : is a decimal of seconds, or: default

   Note: setting any one phase of an inlet, updates all phases of that inlet.

   Examples:
     setPhase 1 high_warning_volts 129.0 : sets all hwv of the parent inlet
     setPhase high_warning_volts 129.0   : sets all hwv values of all inlets
     setPhase low_critical_volts ""      : disables lcv of the parent inlet
     setPhase hysteresis 0.5             : sets all hysteresis to 0.5 volts
     setPhase 1 debounce default         : resets to the default value
-------------------------------------------------------------------------------
```

getCircuit

```
-------------------------------------------------------------------------------
   Syntax:  getCircuit [id] [attribute]

   Where:
     []            : indicates an optional parameter
     id            : is the number of the Phase (e.g. 1 or 3, etc.)
                     or leave it empty to refer to all Phases
     attribute     : is one of the attributes below (or its alias):
     group         : is one of the attribute groups below (or its alias):

   Attributes:
     label                  --
     phase                  --
     rated_volts            (rv, ratedvolts)
     rated_amps             (ra, ratedamps, max_amps, maxamps)
     amps                   (a, amps_rms, ampsrms, arms)
     consumed_amps          (ca, amps_used, amps)
     volts                  (v, volts_rms, voltsrms, vrms)
     frequency              (f, freq)
     voltamps               (va, kva)
     voltamps_reactive      (var, kvar)
     watts                  (w, kw)
     power_factor           (pf, powerfactor)
     high_critical_amps     (hca)
     high_warning_amps      (hwa)
     low_warning_amps       (lwa)
     low_critical_amps      (lca)
     hysteresis_amps        (hysa)
     debounce_amps          (dba)
     high_critical_volts    (hcv)
     high_warning_volts     (hwv)
     low_warning_volts      (lwv)
     low_critical_volts     (lcv)
     hysteresis_volts       (hysv)
     debounce_volts         (dbv)

   Attribute Groups:
     rating                 --
     power_info             (power, pwr)
     setpoints_info         (setpoint_info)
     settings               --

   Examples: {not shown}
-------------------------------------------------------------------------------
```

setCircuit

```
-------------------------------------------------------------------------------
   Syntax:  setCircuit [id] attribute value

   Where:
     []            : indicates an optional parameter
     id            : is the number of the Phase (e.g. 1 or 3, etc.)
                     or leave it empty to refer to all Phases
     attribute     : is one of the following (or its alias):

       high_critical_amps     (hca)
       high_warning_amps      (hwa)
       low_warning_amps       (lwa)
       low_critical_amps      (lca)
       hysteresis_amps        (hysa)
       debounce_amps          (dba)
       high_critical_volts    (hcv)
       high_warning_volts     (hwv)
       low_warning_volts      (lwv)
       low_critical_volts     (lcv)
       hysteresis_volts       (hysv)
       debounce_volts         (dbv)

     value for setpoints      : is a decimal, an empty quoted
                                string "" to disable it, or: disabled
     value for hysteresis     : is a decimal, or: default
     value for debounce       : is a decimal of seconds, or: default

   Examples:
     setCircuit 2 low_warning_amps 1.2   : sets one circuit
     setCircuit low_warning_amps 1.2     : sets all circuits
     setCircuit 2 debounce default       : resets the default value
     setCircuit high_critical_amps ""    : disables alert for all
-------------------------------------------------------------------------------
```

getOutlet

```
-------------------------------------------------------------------------------
   Syntax:  getOutlet [id] [attribute|group]

   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 attributes below (or its alias):
     group         : is one of the attribute groups below (or its alias):

   Attributes:
     label                  --
     phase                  --
     switch                 (relay)
     startup_state          (startup)
     on_delay               --
     off_delay              --
     cycle_delay            --
     alert_switch_change    (alertable)

   Attribute Groups:
     rating                 --
     switch_info            (relay_info)

   Examples:
     getOutlet 3 phase       : gets one attribute for one Outlet
     getOutlet 2             : gets all attributes for one Outlet
     getOutlet phase         : 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            --
       alert_relay_change     (alertable)

     value for label          : is a quoted string (30 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
     value for alertable      : is one of: true, false

   Examples:
     setOutlet 2 label "Web Server ws01"  : sets one label
     setOutlet label default              : resets all to the default value
     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
-------------------------------------------------------------------------------
```

Environment Commands

getEnv

```
-------------------------------------------------------------------------------
   Syntax:  getEnv [id] [attribute]

   Where:
     []             : indicates an optional parameter
     id             : is the number of the Env Port (1 or 2)
                      or leave it empty to refer to all ports
     attribute      : is one of the following (or its alias):

       label           --
       temperature     (temp)
       humidity        (hum)

   Examples:
     getEnv 1 temp     : gets one attribute of the environment port
     getenv            : gets all attributes of all ports
-------------------------------------------------------------------------------
```

scanEnvPorts

```
-------------------------------------------------------------------------------
   Syntax:  scanEnvPorts

   Description:
      Use this command after inserting or removing a device from
      one of the environment sensor ports. This will scan the ports
      to determine what is installed.
-------------------------------------------------------------------------------
```

Alarm Commands

getAlarm

```
-------------------------------------------------------------------------------
   Syntax:  getAlarm [object]

   Where:
     []            : indicates an optional parameter
     object        : is the word system, input, line, circuit, or outlet

   Examples:
      getAlarm outlet   : shows details for outlet alarms
      getAlarm          : shows details for all alarms
-------------------------------------------------------------------------------
```

ackAlarm

```
-------------------------------------------------------------------------------
   Syntax:  ackAlarm [ID]

   Where:
     []            : indicates an optional parameter
     ID            : is the case-sensitive ID of the alarm
                     (use getAlarm to see the ackID)

   Examples:
      ackAlarm abc-123-xyz  : Acknowledge the alarm abc-123-xyz
      ackAlarm              : Acknowledge all alarms
-------------------------------------------------------------------------------
```

Alert Commands

getAlert

```
-------------------------------------------------------------------------------
   Syntax:  getAlert [object] [id] [command “value”]

   Where:
     []             : indicates an optional parameter
     object         : is one of the following:
                      unit, power, outlet
     id             : is the ID number of the object (e.g. 1 or 3, etc.)
                      or leave it empty to refer to all objects
     command        : is one of the following (or its alias):

       triggers        --
       send_to         (send)
       email_to        (email)
       sms_to          (sms)
       send_snmp       --

     value for triggers is as follows (depends on model features*):
       : for unit - startup, login, settings, epo
       : for power - amps, volts
       : for outlet - switch

     value for email/sms is any user login name

     value for send_snmp is true|false or yes|no

   Examples:
     getAlert outlet 2           : shows the alert details for only Outlet 2
     getAlert outlet             : shows the alert details for all outlets
     getAlert send_to "lester"   : shows alerts which lester will recieve
     getAlert triggers "switch"  : shows alerts with switch enabled
-------------------------------------------------------------------------------
```

setAlert

```
--------------------------------------------------------------------------------
  Syntax:  setAlert [object] [id] [command "value"]

   Where:
     []             : indicates an optional parameter
     object         : is one of the following:
                      unit, power, outlet
     id             : is the ID number of the object (e.g. 1 or 3, etc.)
                      or leave it empty to refer to all objects
     command        : is one of the following (or its alias):

       triggers        --
       send_to         (send)
       email_to        (email)
       sms_to          (sms)
       send_snmp       --
       add             --
       email_add       (add_email)
       sms_add         (add_sms)
       delete          --
       email_delete    (delete_email)
       sms_delete      (delete_sms)

     value for triggers is as follows (depends on model features*):
       : for unit - startup, login, settings, epo
       : for power - amps, volts
       : for outlet - switch

     value for email/sms is any user login name

     value for send_snmp is true|false or yes|no

   Examples:

    setAlert outlet send_to "root,lester"
        : sets all outlet object alerts to go to only these two users,
          to both email and sms addresses (if they're defined)

    setAlert unit triggers "startup,login"
        : enables alerts for the startup and login events of a Unit

    (many more examples shown in software)
--------------------------------------------------------------------------------
```

getAlertMisc

```
-------------------------------------------------------------------------------
   Syntax:  getAlertMisc [attribute]

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

       realert         --
       silence         --

   Examples:
      getAlertMisc realert     : shows only the realert attribute value
      getAlertMisc silence     : shows only the silence attribute value
      getAlertMisc             : shows all atributes
-------------------------------------------------------------------------------
```

setAlertMisc

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

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

       realert         --
       silence         --

     value for realert    : is minutes (default is 15)
     value for silence    : is hours (default is 2)

   Examples:
     setAlertMisc realert 20   : realerts every 20 minutes
     setAlertMisc silence 4    : acknowledged alerts are quiet for 4 hours
-------------------------------------------------------------------------------
```

User Commands

makeLoginPswd

```
--------------------------------------------------------------------------------
   Syntax:  makeLoginPswd "string"

   Where:
     string    : is a plain-text string, presumed to be a password, which will be
                 salted and encrypted (both values are returned).

   Examples:
     makeLoginPswd "mP9ur2T$gk"
--------------------------------------------------------------------------------
```

```
#> makeLoginPswd "abc123"

   salt = 8BnM0Qd6hCTJ
   pswd = 6f23b84a083fbbbf08ae1d7d4c7c5e4d9590543d
```

randomizeRoot

```
--------------------------------------------------------------------------------
   Syntax:  randomizeRoot

   Creates a new, random password for the root user.
   To change the root password, connect to the system using the serial port,
   then restart the system, pressing any key when prompted to invoke the
   Serial Dialog settings. See the Getting Started section of the User Guide.
--------------------------------------------------------------------------------
```

getUsers

```
--------------------------------------------------------------------------------
   Syntax:    getUsers


   Examples:
     getUsers     : displays list of all users and their login information
--------------------------------------------------------------------------------
```

getUser

```
-------------------------------------------------------------------------------
   Syntax:  getUser [name] [attribute|group]

   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 attributes below (or its alias):
     group         : is one of the attribute groups below (or its alias):

   Attributes:
     enabled, password_hint (hint),
     first_name, last_name,
     company_name, job_role,
     company_phone, direct_phone,
     email, sms
     login_via_web (enable_web)
     login_via_cli (enable_cli)
     login_via_ftp (enable_ftp)
     view_power
       control_outlets
       edit_outlets
       edit_phases
       edit_inlets
     view_alert, edit_alert
     view_network
       edit_tcpip, edit_sntp
       edit_http, edit_ftp
       edit_smtp, edit_snmp
     view users
       edit_users
     view_logs
       edit_logs
     view_system
       edit_system

   Attribute Groups:
     status      = record management status details
     auth        = user authentication details
     profile    = user profile details
     permissions = user permissions details

   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:
            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
            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
            sms           - address for SMS (e.g. 2125551010@txt.carrier.net)
     lockout        - use the value 'clear' to clear a login lockout

       : 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
              control_outlets
              edit_outlets
              edit_phases
              edit_inlets
            view_alert, edit_alert
            view_network
              edit_tcpip, edit_sntp
              edit_http, edit_ftp
              edit_smtp, edit_snmp
            view users
              edit_users
            view_logs
              edit_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 "name" with_password "loginPswd"

   Where:
     name          : 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" with_password "Qwer!234"
     addUser "mail_admin" with_password "wu7jsKhgr3poi"
     addUser "admin@example.com" with_password "wu7jsKhgr3poi"
-------------------------------------------------------------------------------
```

deleteUser

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

   Where:
     name          : is the login name of the user (e.g. john_doe)
     pswd          : is the login password in quotes

   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):

       ipv4_dhcp          (dhcpv4, dhcp)
       ipv4_address       (ipv4, ip)
       ipv4_subnet        (subnet, mask)
       ipv4_gateway       (gateway)
       ipv6_dhcp          (dhcpv6)
       ipv6_address       (ipv6)
       ipv6_gateway       (gatewayv6)
       ipv6_prefix       (prefixlen, prefix)

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

verifyTcp

```
-------------------------------------------------------------------------------
   Syntax:  verifyTcp [attribute]

   Where:
     []             : indicates an optional parameter
     attribute      : is one of the following:

       v4           : to verify IPv4 static settings
       v6           : to verify IPv6 static settings

   Examples:
     verifyTcp v4   : verifies only IPv4
-------------------------------------------------------------------------------
```

setTcp

```
-------------------------------------------------------------------------------
   Syntax:  setTcp attribute value


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

       ipv4_dhcp          (dhcpv4, dhcp)
       ipv4_address       (ipv4, ip)
       ipv4_subnet        (subnet, mask)
       ipv4_gateway       (gateway)
       ipv6_dhcp          (dhcpv6)
       ipv6_address       (ipv6)
       ipv6_gateway       (gatewayv6)
       ipv6_prefix       (prefixlen, prefix)

     value for ipv4_dhcp      : is on|off
     value for ipv4_address   : is formatted like N.N.N.N (8-bit integers)
                                where each N is an 8-bit integer
     value for ipv4_subnet    : is formatted like N.N.N.N
     value for ipv4_gateway   : is formatted like N.N.N.N
     value for ipv6_dhcp      : is on|off
     value for ipv6_address   : is formatted like H:H:H:H:H:H:H:H
                                where each H is a 16-bit hex value
     value for ipv6_prefix   : is an integer, most frequently 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 off            : disables DHCP for ipv6 (enables static)
-------------------------------------------------------------------------------
```

getHttp

```
-------------------------------------------------------------------------------
   Syntax:  getHttp [attribute]

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

       webui_enabled     (enabled)
       port              --
       ssl_enabled       --
       ssl_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):

       webui_enabled     (enabled)
       port              --
       ssl_enabled       --
       ssl_port          --
       session_minutes   (session, session_mins)

     value for webui_enabled      : is true|false or on|off
     value for port               : is an integer (default is 80)
     value for ssl_enabled        : is true|false or on|off
     value for ssl_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:
     setHttp port 8080       : binds HTTP to port 8080
     setHttp 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 IPv4 address
     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        --

   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        --

     value for enabled     : is true|false or on|off
     value for port        : is an integer (default is 21)

   Examples:
     setFtp enabled true   : enables FTP service
     setFtp port 2100      : binds FTP to port 2100
-------------------------------------------------------------------------------
```

getSmtp

```
-------------------------------------------------------------------------------
   Syntax:  getSmtp [attribute]

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

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

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

setSmtp

```
-------------------------------------------------------------------------------
   Syntax:  setSmtp 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         : is an IPv4 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:
     setSmtp account pdu@xxx.com
     setSmtp server 192.168.0.124
     setSmtp test_email "root,me@example.com"
     setSmtp test_sms "root,8885551212@txt.carrier.net"
-------------------------------------------------------------------------------
```

getSnmp

```
--------------------------------------------------------------------------------
   Syntax:  getSnmp [attribute]

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

       enabled               --
       data_port             --
       community1_string     (comm1)
       community2_string     (comm2)
       trap_ip               --
       trap_port             --
       trap_community        (trap_comm)
       notifications_user   (notif_user)

   Examples:
     getSnmp               : displays all attributes
     getSnmp trap_port     : displays a single attribute's value
--------------------------------------------------------------------------------
```

setSnmp

```
--------------------------------------------------------------------------------
   Syntax:  setSnmp attribute value

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

       enabled               --
       data_port             --
       community1_string     (comm1)
       community2_string     (comm2)
       trap_ip               --
       trap_port             --
       trap_community        (trap_comm)
       notifications_user   (notif_user)

     value for enabled            : is true|false or on|off
     value for data_port          : is an integer (default is 161)
     value for community strings  : is a passphrase of 6-32 characters
     value for trap_ip            : is an IPv4 address
     value for trap_port          : is an integer (default is 162)
     value for trap_community     : is a passphrase of 6-32 characters
     value for notifications_user : is a passphrase of 6-32 characters

   Examples:
     setSnmp data_port 161
     setSnmp trap_ip 192.168.0.124
     setSnmp comm1 "read-only-w5dK-u#d9"
--------------------------------------------------------------------------------
```

getSnmpUsm

```
-------------------------------------------------------------------------------
   Syntax:  getSnmpUsm [id] [attribute]
            alternate form 'getUsm' also allowed

   Where:
     []             : indicates an optional parameter
     id             : is the the ID of the USM record (1-4)
                      or leave it empty to refer to all records
     attribute      : is one of the following (or its alias):

       name                --
       auth_method         authm
       auth_passphrase     authp
       priv_method         privm
       priv_passphrase     privp

   Examples:
     getSnmpUsm 1 name    : gets one attribute for one user
     getSnmpUsm 1         : gets all attributes for one user
     getSnmpUsm name      : gets one attribute for all users
     getSnmpUsm           : gets all attributes for all users
-------------------------------------------------------------------------------
```

setSnmpUsm

```
-------------------------------------------------------------------------------
   Syntax:  setSnmpUsm [id] attribute "value"
            alternate form 'setUsm' also allowed

   Where:
     []             : indicates an optional parameter
     id             : is the the ID of the USM record (1-4)
                      or leave it empty to refer to all records
     attribute      : is one of the following (or its alias):
       name                --
       auth_method         authm
       auth_passphrase     authp
       priv_method         privm
       priv_passphrase     privp

     value for name              : is a quoted string identifying the user
                                   (4-32 printable ASCII chars, but not ~ or =)
     value for auth_method       : is either 'none' or 'md5'
     value for auth_passphrase   : is a quoted password/passphrase
                                   (6-32 printable ASCII chars, but not ~ or =)
     value for priv_method       : is either 'none' or 'des'
     value for priv_passphrase   : is a quoted password/passphrase
                                   (6-32 printable ASCII chars, but not ~ or =)

   Examples:
     setSnmpUsm 1 name "managementApp"    : sets one record's name
     setSnmpUsm 1 auth_method MD5         : sets one record
     setSnmpUsm 1 authp "mngrpP8*word"    : sets one record
-------------------------------------------------------------------------------
```

verifyUsm

```
-------------------------------------------------------------------------------
   Syntax:  verifyUsm

   Verifies that all USM records have valid configurations, which includes:
   - name with both auth and priv methods as none, and empty passphrases.
   - name with non-none auth method and auth passprhrase, but no priv settings.
   - name with non-none auth and priv methods, and both non-empty passphrases.
   - all settings with empty values (to leave the record unconfigured).
-------------------------------------------------------------------------------
```

clearUsm

```
-------------------------------------------------------------------------------
   Syntax:  clearSnmpUsm id
            alternate form 'clearUsm' also allowed

   Where:
     id     : is the the ID of the USM record (1-4)

   Examples:
     clearSnmpUsm 4   : clears only USM record 4 to empty values
-------------------------------------------------------------------------------
```

Log Commands

viewLog

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

viewStartupLog

```
-------------------------------------------------------------------------------
   Syntax:  viewStartupLog

   Examples:
      viewStartupLog       : shows all log lines
                             (doesn't use severity & category filters)
-------------------------------------------------------------------------------
```

exportLog

```
-------------------------------------------------------------------------------
   Syntax:  exportLog

   Description:
      Exports the internal event log to a tab-delimited text file in the FTP
      file system at FLASH0/Logs/. (Startup log records are included.)
-------------------------------------------------------------------------------
```

getLog

```
-------------------------------------------------------------------------------
   Syntax:  getLog attribute

   Where:
     []             : indicates an optional parameter
     attribute      : is one of the following:

       max_records
       cli_switching
       rest_switching
       snmp_switching

   Examples:
     getLog                       : shows all log setting attributes
     getLog cli_switching         : shows the cli_switching attribute value
-------------------------------------------------------------------------------
```

setLog

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

   Where:
     []             : indicates an optional parameter
     attribute      : is one of the following:

       max_records
       cli_switching
       rest_switching
       snmp_switching
       reset!!!

     value for max_records  : 0 to disable all logging, or
                              an integer between 20 and 1000
     value for switching    : either yes/no or on/off
     value for reset!!!     : no value

   Examples:
     setLog max_records 250    : Erases all log entries, and sets a
                                 new limit of 250 records. There is NO
                                 "are you sure" step.
     setLog cli_switching on   : sets label of the system
     setLog reset!!!           : Erases all log entries. There is NO
                                 "are you sure" step.
-------------------------------------------------------------------------------
```

System Commands

getSystem

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

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

       label              --
       location           (loc)
       asset_id           (asset)
       contact            --
       model_number       (model, model_no)
       serial_number      (serial, serial_no)
       web_link           (link)
       mac_address        (mac, mac_addr)
       dialog_baudrate    (baud)
       dialog_delay       (delay)
       version            --
       platform           --
       degrees            --
       start_time         (started)
       time               --

   Examples:
     getSystem label      : gets one attribute of the system
     getSystem            : gets all attributes of the system
     getSystem export     : exports user settings to FTP file space
-------------------------------------------------------------------------------
```

setSystem

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

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

       label              --
       location           (loc)
       asset_id           (asset)
       contact            --
       dialog_baudrate    (baud)
       dialog_delay       (delay)
       degrees            --
       time               --
       restart            --
       reset!!!           --

     value for label       : is a quoted string (30 chars max)
     value for location    : is a quoted string (22 chars max)
     value for asset_id    : is a quoted string (30 chars max)
     value for contact     : is a quoted string (80 chars max)
     value for baudrate    : is an integer from the options of:
                             1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200
     value for delay       : is an integer from the options of: 10 or 5
     value for degrees     : is either F or C
     value for time        : is formatted like "YYYY-MM-DD HH:MM:SS" (24 hr)
                             (applicable only when not connected to SNTP)
     value for restart     : none
     value for reset       : none (be careful)

   Examples:
     setSystem label "WWW Servers"     : sets label of the system
     setSystem asset "19-12-27-032"    : 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!
-------------------------------------------------------------------------------
```

exportSettings

```
-------------------------------------------------------------------------------
   Syntax:  exportSettings

   Description:
      Exports user-defined settings to tab-delimited text files in the FTP
      file system at FLASH0/Settings/.
-------------------------------------------------------------------------------
```

updateExportKey

```
-------------------------------------------------------------------------------
   Syntax:  updateExportKey

   Description:
     This command generates a new export key.
     Any exports created prior to this new key will not be importable.
-------------------------------------------------------------------------------
```

restart

```
--------------------------------------------------------------------------------
   Syntax:  restart

   Description:
   Restarts the RCM main board software. Does not reset peripheral boards
   running power sensors or outlet relays. Power to outlets will continue.
--------------------------------------------------------------------------------
```

* Synonymous with the command `setSystem restart`

Optima RCM Software

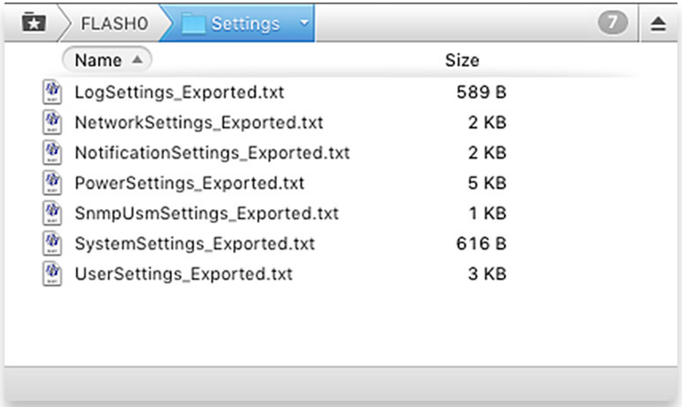
Settings Files Reference

Settings Files Reference

Settings may be exported to text files. This is primarily intended to document and archive the setup of the PDU. However, exported files may be imported to a PDU to restore settings, or even to transfer settings to multiple PDUs.

Since settings files are just text files, they can be edited. This can be taken advantage of by exporting one PDU's settings, and creating multiple slightly different variations of settings for different PDUs.

If a handful of PDUs are to be configured once and rarely changed, then working with settings files is not likely a productive option. However, multiple new PDUs are frequently provisioned for identical or similar applications, then learning how to export/edit/import settings files may be a faster way to get them configured compared to editing via the web UI or CLI. (Scripting the Telnet interface is also a productive alternative.)



Name	Size
LogSettings_Exported.txt	589 B
NetworkSettings_Exported.txt	2 KB
NotificationSettings_Exported.txt	2 KB
PowerSettings_Exported.txt	5 KB
SnmpUsmSettings_Exported.txt	1 KB
SystemSettings_Exported.txt	616 B
UserSettings_Exported.txt	3 KB

Exporting Settings Files

Files can be exported using the web interface or command line.

To export settings using the web interface, open the System page, and find the User Settings panel. Press Export.

To export settings using the command line, log into Telnet or SSH (or use the Serial CLI) and execute the command `exportSettings`.

All settings will be exported to the FTP file system /FLASH0/Settings/ folder. Each file name will have an _Exported suffix and .txt file extension.

Additionally, a file named ExportKey.txt will be exported to /FLASH0/System/ which should also be copied and stored (through separately from the settings files). See the next two sections to understand the use of the ExportKey file.

Export Key File

Some files, such as NetworkSettings, may have sensitive data fields. These fields are encrypted in the exported text files. One component of that encryption is called the Export Key. When files are exported, that key is also exported in the /FLASH0/System/ExportKey.txt file.

This key will be needed to import settings into other PDUs, and will be useful for importing back to the same PDU if the key had changed, or the PDU has been reset to default settings.

The ExportKey.txt file should be removed from the FTP file system after a settings export and kept separately from the settings files. Anyone with access to the exported settings file and the key could potentially use another RCM PDU to important those settings—thus decrypting them. As with all encryption keys, keep the Export Key value safe and separate from the data it is used to encrypt.

The key is created automatically. It can be forced to change using the CLI command `updateExportKey`.

Importing Settings Files

To import settings, put the settings files into the /FLASH0/Settings/ folder of any PDU, and remove the _Exported suffix from each file. The file names must be as follows:

- LogSettings.txt
- NetworkSettings.txt
- NotificationSettings.txt
- PowerSettings.txt
- SnmpUsmSettings.txt
- SystemSettings.txt
- UserSettings.txt

THE EXPORT KEY SHOULD BE STORED SEPARATELY FROM THE EXPORT SETTINGS FILES.

Anyone with access to the exported settings file and the key could potentially use another RCM PDU to important those settings—thus decrypting them.

As with all encryption keys, keep the Export Key value safe and separate from the data it is used to encrypt.

Next, also upload the ExportKey.txt file to the /FLASH0/Settings/ folder. If you know the key has not changed, the key file is not needed to import settings into the same PDU from which they were exported. If the correct key is not available, affected fields will be decrypted incorrectly, and will need updated manually via CLI or web UI.

Once all files are uploaded, restart the PDU. During startup, the RCM software will import each file. This will result in the PDU startup taking longer than normal—depending on the number of settings details being imported, the extra delay can be several minutes. Be patient, do not interrupt the process, and wait for the PDU to indicate that it is ready. If the PDU has an on-board display, the display will provide indication of the progress through the imports and normal startup steps.

After the files have been imported, the file names will be automatically changed to have _Imported suffixes so they don't get imported again at the next startup. The ExportKey.txt file will have been automatically deleted (as there's no need to keep it there).

After the PDU has completed startup, check the log for any errors which may have been noted.

Know What You’re Doing

Read the warning at the right!

The safest way to change settings on an 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.

File Format and Syntax

Settings files are ASCII text files saved with “Windows-style” line endings (that is, the \r\n sequence). Files may be copied, opened, and edited before moving them to another PDU to be imported.

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, Apple Pages, etc.

The settings files are generally not white-space sensitive (other than needing a line ending at the end of the file). White spaces between delimited items on a line can be tabs or spaces, or mixes of both. The format of each file is detailed in the sections below, however, refer to “Software Settings” on page 69 for details about the internal settings of each file.

IMPORTING EDITED SETTINGS FILES 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 import files into PDUs in service.

Marway has attempted to make the RCM software self-healing from the affects of malformed 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.

```
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).
```

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. Likewise, settings which are subordinate to a named group may be indented. The indentation is not significant. Examples of name-value pairs:

```
loginName = root
onDelaySeconds = 5.0
```

Any valid setting name which does not have a valid value, will be reverted to the factory default for that setting.

Objects and IDs

Some files include named objects with IDs in the format of `object~N` where object is a label such as outlet, http, user (and others), and where N is a simple incremental integer (1, 2, 3, etc.), The whitespace around the ~ is not significant. Examples of objects and IDs:

```
outlet~3
user~1
http~1
```

Object Groups of Settings

Files using named objects will have groups of settings for that object listed under the object name. The start of the group is a line with an `object~id`. The end of the group is defined when a line with another named object is reached, or the end of the file is reached. The grouped settings may be indented, but these spaces are not meaningful.

Some files have a mixture of "root level" settings (name-value pairs which are not part of named object group) as well as group level settings. In these files, all root level settings must be be above the first named object group. Here's an example of ungrouped settings followed by an object group and its settings from NotificationSettings.txt:

```
// root-level settings
realertMinutes = 20
silenceHours   = 4

// named object with group-level settings
unit~1
  triggers      = startup, epo
  emailToUsers  = root,lester
  smsToUsers    = root,lester
  sendSnmpTrap  = true

power~1
  triggers      = amps, volts
  emailToUsers  = root
  smsToUsers    = root,lester
  sendSnmpTrap  = true
```

The order of root-level name-value pairs is not significant. In the above example, `silenceHours` could be listed first.

The order of named groups within a file *may be significant*. This will be detailed in the sections below.

However, the order of the name-value pairs within any one group is not significant. In the above example, the `triggers` setting within each group could be first, second, last—it doesn't matter. However, a name-value pair which belongs to a specific group must be within its group (after the group name, and before another group name).

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).

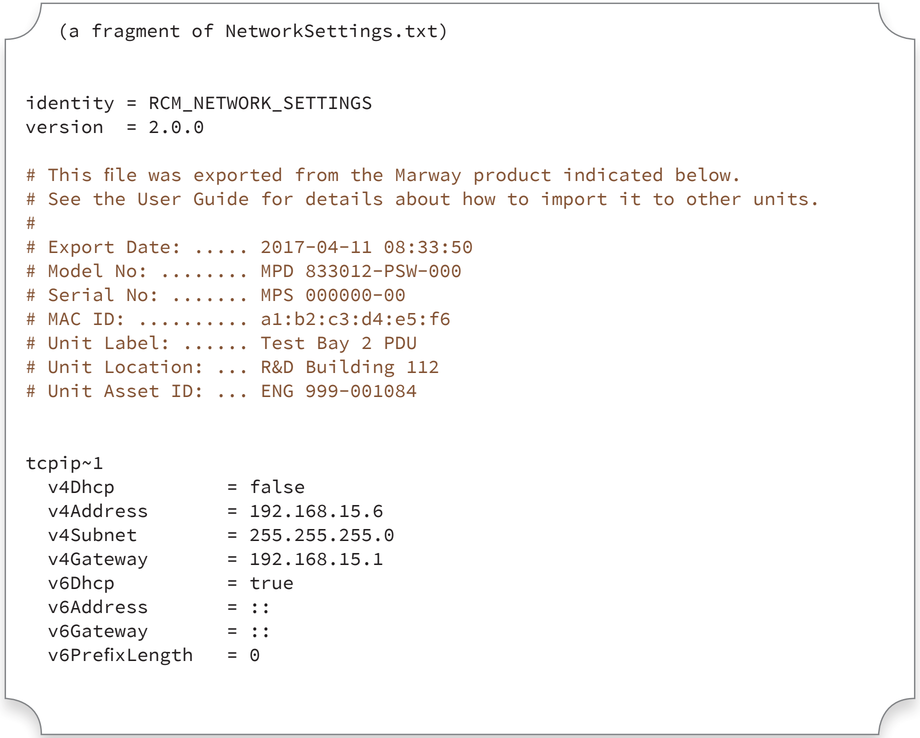
Being a part of the text file, theses values are technically editable, therefore it is critical to ensure they remain correct. Be sure they are correct for the target PDU before importing. The best way to assure the correct version and formatting is being used is to export settings from a PDU and verify the identity and version data of those files.

Comments

Comment lines are allowed. Comments must begin at the start of the line—that is, be on a line all to themselves, and not be on the same line as a settings definition. Comments can start with either # or //.

Exported File Comments

Files which have been exported will have a block of commented lines near the top of the file. These lines identify details of the source of the exported file.



(a fragment of NetworkSettings.txt)

```
identity = RCM_NETWORK_SETTINGS
version  = 2.0.0

# This file was exported from the Marway product indicated below.
# See the User Guide for details about how to import it to other units.
#
# Export Date: ..... 2017-04-11 08:33:50
# Model No: ........ MPD 833012-PSW-000
# Serial No: ....... MPS 000000-00
# MAC ID: .......... a1:b2:c3:d4:e5:f6
# Unit Label: ...... Test Bay 2 PDU
# Unit Location: ... R&D Building 112
# Unit Asset ID: ... ENG 999-001084


tcpip~1
  v4Dhcp          = false
  v4Address       = 192.168.15.6
  v4Subnet        = 255.255.255.0
  v4Gateway       = 192.168.15.1
  v6Dhcp          = true
  v6Address       = ::
  v6Gateway       = ::
  v6PrefixLength  = 0
```

SystemSettings.txt

Refer to “System Settings” on page 71 for details about the settings.

This is a simple file consisting of a relative few settings. The entire file is shown at the right. The formatting of the file consists entirely of root-level name-value pairs. There are no named object groups.

SystemSettings can be shared across all PDU models running the same version of RCM software.

```
identity = RCM_SYSTEM_SETTINGS
version  = 2.0.0

label               = Test Bay 2 PDU
location            = R&D Building 112
assetId             = ENG 999-001084
contact             = Robin Banks, 999-888-5555
temperatureUnits    = Farenheit
dialogBaudRate      = 115200
dialogDelay         = 5
```

LogSettings.txt

Refer to Log Settings on page XX for details about the settings.

This is a simple file consisting of a relative few settings. The entire file is shown at the right. The formatting of the file consists entirely of root-level name-value pairs. There are no named object groups.

LogSettings can be shared across all PDU models running the same version of RCM software.

```
identity = RCM_LOG_SETTINGS
version  = 2.0.0

maxRecords              = 100
logCLISwitching         = false
logRESTfulSwitching     = false
logSNMPSwitching        = false
```

NetworkSettings.txt

Refer to “Network Settings” on page 72 for details about the settings.

This file consists of name-value pairs organized under named object groups corresponding to the protocols. Note the object ID of each protocol is 1. The order of the named protocol groups is not significant.

NetworkSettings can be shared across all PDU models running the same version of RCM software—however, the `tcpip` named group should not be included in files to be imported. These values are exported for documentation purposes, but since each PDU should have its own IP settings, this group will be ignored during import.

```
 (a fragment of NetworkSettings.txt)


identity = RCM_NETWORK_SETTINGS
version  = 2.0.0

http~1
  enabled         = true
  httpPort        = 80
  httpsEnabled    = false
  httpsPort       = 443
  sessionMins     = 30

sntp~1
  serverIp1       = 192.168.15.254
  serverIp2       = 192.168.15.254
  syncIntervalHrs = 12
  stdOffsetHrs    = -8.0
  dstOffsetHrs    = -7.0
  dstStartsAt     = 3.2.0/02:00:00
  dstEndsAt       = 11.1.0/02:00:00
```

SnmpUsmSettings.txt

Refer to "Network Settings" on page 72 for details about the settings.

The order of the named groups *is* significant.

This file consists of name-value pairs organized under named object groups corresponding to a fixed number of user objects matching the SNMPv3 USM user settings. The object ID of each protocol is sequential starting with 1.

SnmpUsmSettings can be shared across all PDU models running the same version of RCM software.

```
 (a fragment of SnmpUsmSettings.txt)


identity = RCM_USM_SETTINGS
version  = 2.0.0

usm~1
  name              = managerApps
  authMethod        = MD5
  authPassphrase    =
  privMethod        = DES
  privPassphrase    =
```

UserSettings.txt

Refer to "User Settings" on page 77 for details about the settings.

This file consists of name-value pairs organized under named object groups corresponding to each user. Each user has a sequential object ID starting with 1, and is followed by several settings (not all are shown).

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.

After the root user, the order of the users must each have an ID incremented by one relative to the user before it (1, 2, 3, etc.).

If you need, but do not know, the root user password, the correct way to update the root user password is to use the serial interface. See the section “Change root User Password” on page 11. Re-export the users file to get the password hash and salt.

To create a new password hash and salt for a new user (other than root), use the CLI command `makeLoginPswd`. (otherwise the exported values of existing users will propagate that user’s password to imported PDUs).

```
 (a fragment of UserSettings.txt)


identity = RCM_USER_SETTINGS
version  = 2.0.0

user~1
  loginName      = root
  loginPswd      = c33e6f892497cc56e25e8159834a0db299e27c8b
  pswdSalt       = l5bGJrjQpJmm
  pswdHint       = uncle billy
  // (big piece cut out of this example)

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

NotificationSettings.txt

Refer to “Notification Settings” on page 82 for details about the settings.

NotificationSettings can be transferred only to other PDUs of the same model. The order of the named groups *is* significant.

This file consists of name-value pairs organized under named object groups corresponding to a set of internally created notification objects specific to a PDU model’s features.

The order of the named groups *is* significant. Each notification object has a sequential object ID for its object type starting with 1, and is followed by several settings. Therefore, there may be a `unit~1`, `power~1`, `outlet~1`, `outlet~2`, etc. depending on the PDUs hardware features.

```
 (a fragment of NotificationSettings.txt)


identity = RCM_NOTIFICATION_SETTINGS
version  = 2.0.0

realertMinutes = 20
silenceHours   = 4

unit~1
  triggers     = sartup,epo
  emailToUsers = root
  smsToUsers   = root
  sendSnmpTrap = false
```

PowerSettings.txt

Refer to “Power Settings” on page 84 for details about the settings.

PowerSettings can be transferred only to other PDUs of the same model. The order of the named groups *is* significant.

This file consists of multiple structurally nested objects. Each layer of nested objects has its own series of sequential object IDs starting with 1. For example, a three-phase powered unit will have three `currentPhases` numbered 1, 2, and 3 as well as `voltagePhases` numbered 1, 2, and 3. There may be Circuits, and there will 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).

```
(a fragment of powerSettings.txt)

identity = RCM_POWER_SETTINGS
version  = 2.0.0

inlet~1
  label             = Power Bank A

currentPhase~1
  highCriticalAmps  = disabled
  highWarningAmps   = disabled
  lowWarningAmps    = disabled
  lowCriticalAmps   = disabled
  hysteresisAmps    = 0.50
  debounceSeconds   = 10.00

voltagePhase~1
  highCriticalVolts = disabled
  highWarningVolts  = disabled
  lowWarningVolts   = disabled
  lowCriticalVolts  = disabled
  hysteresisVolts   = 5.00
  debounceSeconds   = 10.00

outlet~1
  label             = Control Panel
  onDelaySeconds    = 0.00
  offDelaySeconds   = 0.00
  cycleDelaySeconds = 3.00
 startupState       = on
  switchIsAlertable = true

outlet~2
  label             = DAQ Chassis
  onDelaySeconds    = 0.00
  offDelaySeconds   = 0.00
  cycleDelaySeconds = 8.00
  startupState      = on
  switchIsAlertable = true
```

Optima 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 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 pedantic 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 URL

```
PATCH http://192.168.1.10/outlets/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=Ygv0PkCheJrl4UuyPfd1OFUpD807XQ4Y
new_value=on
```

The response to this request will be a standard HTTP response with the body having a simple string. Many responses to RCM requests are single values, so the HTTP content type will be text/plain rather than incur unneeded overhead of XML or JSON 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. 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. While SNMP is designed as a machine-to-machine interface, it can be complex to develop scripts with, and difficult for people to read. Since a RESTful API is basic HTTP, and is more human readable than SNMP, it can be both quicker to develop custom automation scripts with, and more efficient to for the machine-to-machine dialog.

Request Syntax

Each request will be made of up elements in the URL as well as additional HTTP form parameters. We refer to the key URL elements as the verb, noun, instance ID, and attribute. The form of a request URL like this:

```
POST https://192.168.0.10/outlets/1/switch?session_id=x
```

can be represented generically as this:

```
:verb https://192.168.0.10/:noun/:id/:attribute?:form_parameter(s)
```

The noun is what REST refers to as the resource.

API 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 utilized in any meaningful way. However, the embedded web server, like most web servers, supports only 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 URL needs GET or POST, the standard HTTP request method can be specified. However, when PATCH or DELETE are needed, the URL 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/outlets/1/switch?
new_value=on&session_id=abcxyz
```

it is necessary to accomplish that message this way:

```
POST https://192.168.0.10/outlets/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.

API Nouns / Resources

The target or object of a RESTful API request is a resource. This is the noun of the request. In the RCM software, supported resources revolve primarily around the power device objects. Valid nouns include: session, system, inlets, phases, circuits, and outlets. Not all power objects are supported on every PDU.

API Attributes

Attributes are the details about an object (noun). For an RCM `system` object, that includes the label, location, asset ID, contact, and more. For an `outlet` object, attributes include label, switch, rated_volts, and others.

For the most part, any object supported by the RESTful API will support all of the attributes which are supported on the command line for that same object. Sometimes, the CLI includes attribute groups such as `power`, `rating`, or `switch_info`, but these are not individual data point attributes, and would not be supported in the RESTful API.

API Form Parameters

These are request parameters passed as HTTP form parameters. In a POST, these will usually be part of the header, and in a GET they would normally be part of the URL, but either style works.

Common form parameters will be `session_id` and `rest_method`. Though specific request may have others.

Responses

As of this release, the API does not support discovery of the PDU power architecture (how many power inlets, phases, outlets, etc.). That is, it does not support requesting collections which could be parsed to navigate or "walk" available resources. It is assumed the power architecture of the PDU is a known entity which will remain stable over the life of a given script of automated tasks.

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.

Given the API's focus on working with specific individual attributes, responses are single-value text strings with the `Content-Type` of `text/plain` (the API does not adapt to requests for specific formats), and there should be no need to parse the HTTP response body.

If a request fails, or was malformed to start with, a value in the form of a negative number is returned. Additionally, the HTTP content type will be set to `text/html` to indicate the body is an error (see side bar "RCM REST Error Codes"). Therefore, while the body should not need parsed, the HTTP `Content-Type` header needs to be tested to determine if the response is an error or valid data.

Authentication API

RCM 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=.

The user can be any RCM user allowed access via the web. It is beneficial to define a user specific to each script using the RESTful API. Two sources logging into the PDU using the same name and password will interrupt each others sessions.

Reminder: for enhanced 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.

RCM REST Error Codes

All responses are returned with an HTTP status code of 200 (limitations in the underlying embedded HTTP server make it impractical to return meaningful status codes for the REST implementation). Therefore, to avoid having to parse the response body to determine if the content is an error, we're using the work around of returning error codes with a Content-Type of text/html (whereas data is returned as text/plain).

-902	Request command was too long (field data too long).
-903	Invalid rest_method value.
-904	Submitted rest_method not supported by requested resource.
-911	Invalid session_id. Start a new session.
-912	Invalid login: name or password was not recognized.
-913	Session login failed (internal error, try again).
-922	Invalid instance ID.
-923	Invalid attribute name.
-924	Invalid new_value.
-932	Requested attribute not supported by the requested resource. Note that some instances of a resource may not support all valid attributes.

If the response starts with `<!DOCTYPE` or `<html` then a request was made to a non-RESTful API resource.

API Resources

Available API resources are focused on providing remote switching, and access to reading power data and system identification.

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 (which is essentially the same as a logout).

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.

More than one script should not be running at the same time.

POST /session

URI Parameters

- None

Form Parameters

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

Response

- The `session_id` as a 32-character string which must be saved to send along with future requests.

DELETE /session

URI Parameters

- None

Form Parameters

- `session_id` — The previously acquired ID from `POST /session`.
- `rest_method` — Must be "DELETE" in upper case.

Response

- A string of 32 zeros.

/inlets

Even though the PDU itself will always have a power inlet, there may or may not be inlet REST resources based on the PDU model. If the PDU is either power monitored or current monitored, there will be at least one inlet. As of this version, GET/inlets does not return a collection. The script must know which phase of the PDU to request.

GET /inlets/:id/:attribute

`URI :id`

- The PDU's numeric instance of the Inlet (values start at 1).

`URI :attribute`

The following attributes respond with a text/plain version of their respective values.

- `phase` [string]
- `label` [string]

Form Parameters

- `session_id` — The previously acquired ID from `POST /session`.

Response

- The response should be the text version of the value of the attribute.
- Note that the value of the phase response provides an indication of the power form of the inlet. LN is a single-phase power source. ABC is a three-phase delta, and ABCN is a three-phase wye.

/phases

There may or may not be phase resources based on the PDU model. If the PDU is either power monitored or current monitored, there will be at least one phase (depending on the power form and number of monitored inlets). Even if a phase is available, some attributes listed below may not be supported. As of this version, GET/phases does not return a collection. The script must know which phase of the PDU to request.

GET /phases/:id/:attribute

`URI :id`

- The PDU's numeric instance of the Phase (values start at 1).

`URI :attribute`

The following attributes (and their aliases) respond with a text/plain version of their respective values.

- `phase` [string]
- `rated_volts` (rv, ratedvolts) [decimal]
- `rated_amps` (ra, ratedamps, max_amps, maxamps) [decimal]
- `amps` (a, amps_rms, ampsrms, arms) [decimal]
- `consumed_amps` (ca, amps_used, amps) [decimal]
- `volts` (v, volts_rms, voltsrms, vrms) [decimal]
- `frequency` (f, freq) [decimal]
- `voltamps` (va, kva) [integer]
- `voltamps_reactive` (var, kvar) [integer]
- `watts` (w, kw) [integer]
- `power_factor` (pf, powerfactor) [decimal]
- `high_critical_amps` (hca) [decimal]
- `high_warning_amps` (hwa) [decimal]
- `low_warning_amps` (lwa) [decimal]
- `low_critical_amps` (lca) [decimal]
- `hysteresis_amps` (hysa) [decimal]
- `debounce_amps` (dba) [decimal]
- `high_critical_volts` (hcv) [decimal]
- `high_warning_volts` (hwv) [decimal]
- `low_warning_volts` (lwv) [decimal]
- `low_critical_volts` (lcv) [decimal]
- `hysteresis_volts` (hysv) [decimal]
- `debounce_volts` (dbv) [decimal]

Form Parameters

- `session_id` (required) — The previously acquired ID from `POST /session`.

Response

- The response should be the text version of the value of the attribute. Units are not included with numeric values. Decimal precision is meaningful.

/outlets

There will always be one or more outlets, however not all outlets are necessarily switched nor have power data. Therefore, some outlets may not support all attributes listed below. As of this version, the resource `GET/outlets` does not return a collection. The script must know which outlet of the PDU to request.

GET /outlets/:id/:attribute

URI `:id`

- The PDU's numeric instance of the Outlet (values start at 1).

URI `:attribute`

The following attributes (and their aliases) respond with a text/plain version of their respective values.

- `label` [string]
- `phase` [string]
- `switch` (relay) [string]
- `startup_state` (startup) [string]
- `on_delay` [decimal]
- `off_delay` [decimal]
- `cycle_delay` [decimal]
- `alert_switch_change` (alertable) [true|false]

Form Parameters

- `session_id` (required) — The previously acquired ID from `POST /session`.

Response

- The response should be the text version of the value of the attribute. Units are not included with numeric values. Decimal precision is meaningful.

Response to `switch`

- a value of "on" or "off" would be normal
- a value of "unknown" means the outlet status cannot be determined (this may require an All On / All Off cycle of the outlets to clear, or restart of the controller).
- a value of "fail on" or "fail 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 an active EPO state preventing power flow.

PATCH /outlets/:id/switch

Sent as a POST, but identified as a PATCH through a form parameter.

URI `:id`

- The PDU's numeric instance of the Outlet (values start at 1).

Form Parameters

- `session_id` (required) — The previously acquired ID from `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

- The response is the ending state of the outlet switch after the state change, which should be "on" or "off," but could be "fail on" or "fail off" is there was an error.

PATCH /outlets/:id/startup_state

Sent as a POST, but identified as a PATCH through a form parameter.

`URI :id`

- The PDU's numeric instance of the Outlet (values start at 1).

Form Parameters

- `session_id` — The previously acquired ID from `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 /outlets/:id/on_delay

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

`URI :id`

- The PDU's numeric instance of the Outlet (values start at 1).

Form Parameters

- `session_id` — The previously acquired ID from `POST /session`.
- `rest_method` — Must be "PATCH" in upper case.
- `new_value` — A numeric string such as 0.5, 2.0, or 1 representing a decimal number of seconds.

Response

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

PATCH /outlets/:id/off_delay

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

`URI :id`

- The PDU's numeric instance of the Outlet (values start at 1).

Form Parameters

- `session_id` — The previously acquired ID from `POST /session`.
- `rest_method` — Must be "PATCH" in upper case.
- `new_value` — A numeric string such as 0.5, 2.0, or 1 representing a decimal number of seconds.

Response

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

PATCH /outlets/:id/cycle_delay

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

URI `:id`

- The PDU's numeric instance of the Outlet (values start at 1).

Form Parameters

- `session_id` (required) — The previously acquired ID from `POST /session`.
- `rest_method` — Must be "PATCH" in upper case.
- `new_value` — A numeric string such as 0.5, 2.0, or 1 representing a decimal number of seconds.

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/:attribute

URI `:id`

- Always set to 1.

URI `:attribute`

The following attributes respond with a text/plain version of their respective values.

- `label` [string]
- `location` (loc) [string]
- `asset_id` (asset) [string]
- `contact` [string]
- `degrees` [Farenheit|Celcius]
- `version` [string]
- `model_number` (model, model_no) [string]
- `serial_number` (serial, serial_no) [string]
- `web_link` (link) [string]
- `mac_address` (mac, mac_addr) [string]
- `serial_baudrate` (baud) [integer]
- `dialog_delay` (delay) [integer]
- `start_time` (started) [M D Y H:M:S xM]
- `time` [M D Y H:M:S xM]

Form Parameters

- `session_id` (required) — The previously acquired ID from `POST /session`.

Response

- The response should be the text version of the value of the attribute.

SNMP Support

SNMP Support

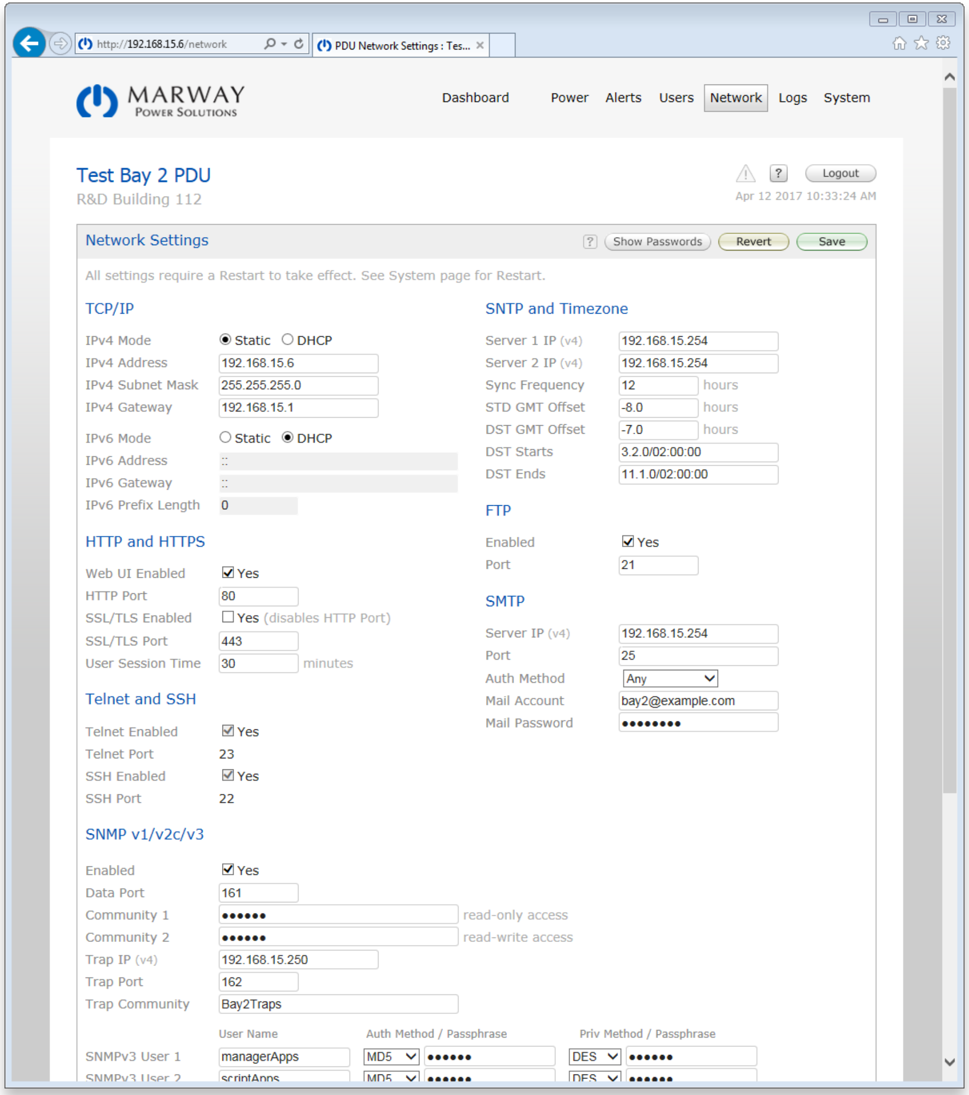
The RCM software includes support for SNMP v1/v2c/v3. Configuration of SNMP is done on the Network web page or using the `setSnmp` CLI command.

RFC style documentation for the SNMP interface, as well as MIBs for managers and browsers are available at http://www.marway.com/software/, and consist of the following documents:

- Marway-MIB-Memo-SMI.pdf
- Marway-MIB-Memo-AgentCaps.pdf
- Marway-MIB-Memo-Chassis.pdf
- Marway-MIB-Memo-Sensor.pdf
- Marway-MIB-Memo-Power.pdf

Generally, the capabilities supported through SNMP include power data reading, outlet switching, power alarm status, and traps/notifications covering all key event alerts available to the web UI.

Given the nature of the small embedded environment of the RCM software, there are some limits to SNMP support. There is no support for the sysOR table, there's a limit of four distinct USM users, and encryption for v3 includes the original v3 requirements of MD5 for authentication and DES for privacy. Traps/notifications are not encrypted.



Test Bay 2 PDU
R&D Building 112

Apr 12 2017 10:33:24 AM

Network Settings

All settings require a Restart to take effect. See System page for Restart.

TCP/IP

IPv4 Mode	● Static ○ DHCP
IPv4 Address	192.168.15.6
IPv4 Subnet Mask	255.255.255.0
IPv4 Gateway	192.168.15.1
IPv6 Mode	○ Static ● DHCP
IPv6 Address	::
IPv6 Gateway	::
IPv6 Prefix Length	0

HTTP and HTTPS

Web UI Enabled	☑ Yes
HTTP Port	80
SSL/TLS Enabled	□ Yes (disables HTTP Port)
SSL/TLS Port	443
User Session Time	30 minutes

Telnet and SSH

Telnet Enabled	☑ Yes
Telnet Port	23
SSH Enabled	☑ Yes
SSH Port	22

SNMP v1/v2c/v3

Enabled	☑ Yes
Data Port	161
Community 1	•••••• read-only access
Community 2	•••••• read-write access
Trap IP (v4)	192.168.15.250
Trap Port	162
Trap Community	Bay2Traps

	User Name	Auth Method / Passphrase		Priv Method / Passphrase	
SNMPv3 User 1	managerApps	MD5	••••••	DES	••••••
SNMPv3 User 2	scriptApps	MD5	••••••	DES	••••••

SNTP and Timezone

Server 1 IP (v4)	192.168.15.254
Server 2 IP (v4)	192.168.15.254
Sync Frequency	12 hours
STD GMT Offset	-8.0 hours
DST GMT Offset	-7.0 hours
DST Starts	3.2.0/02:00:00
DST Ends	11.1.0/02:00:00

FTP

Enabled	☑ Yes
Port	21

SMTP

Server IP (v4)	192.168.15.254
Port	25
Auth Method	Any
Mail Account	bay2@example.com
Mail Password	••••••••

Optima RCM Software

Firmware Updates

Firmware Updates

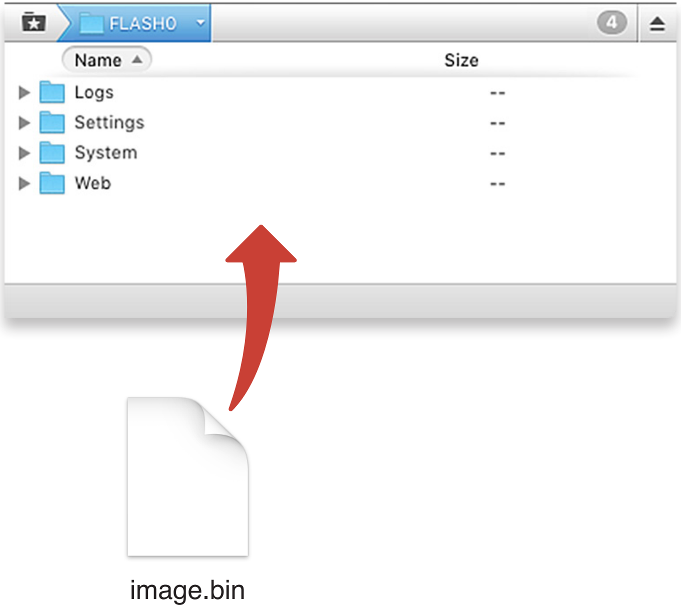
Should there be a need to update the firmware in the RCM PDU, there's a very simple way to do that. There's two possible firmware files which may need to be updated:

> 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.
>
> rom.bin is the bootloader for the PDU. This may or may not be included in an update. If present, 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 rom.bin.

With either of these files, use an FTP client to upload them 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 each upload to ensure each file has been successfully copied and installed, then restart the PDU using the web interface or command line, or reboot the PDU using the recessed button on the keypad.

NOTE: If you connect to the Serial port,

After the files are uploaded, they are moved to another location, so they won't remain visible in the FTP file listing.



Optima RCM Software

Display Operation

Display and Keypad Operation

Optionally mounted to the PDU is an LED display and keypad used to display power data.

If the PDU has current or power monitored inlets, the display defaults to auto-scrolling through each phases's amps when there's no user interaction. However, the user can manually select any of the available power values by using the six pushbuttons on the keypad.

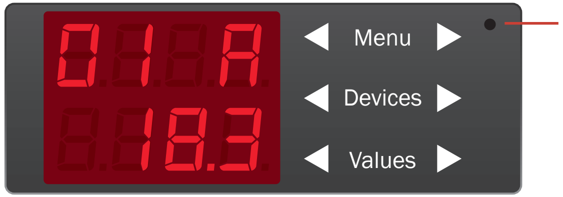
The diagram on the right shows the typical display options for a power-monitored, 3-phase unit. The exact options will vary based on a specific PDU's features.

Viewing Power Data

The Menu buttons cycle through the list of the options shown under the Menu column: Auto Scroll (AUTO), Phases (PHAS), Circuits (CIRC), and Outlets (OUTL). Once the desired menu item is displayed, press the Devices ► button to show the first power data item.

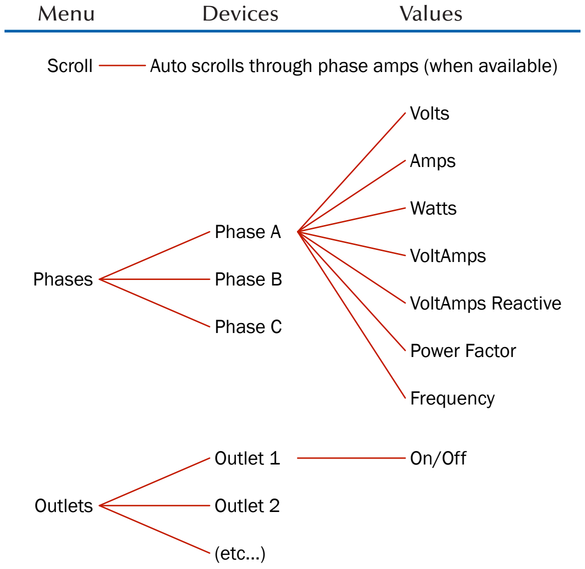
Each power data item identifies the device ID and measurement on the top line, and the value on the bottom line. In the example display shown at the right, assume we selected PHAS from the menu, then pressed the right Devices button. The top line includes 01 A which is Phase 1 amps. The bottom line displays the value: 12.74.

The Devices buttons cycle through each of the items selected by the menu. Using the above example again, if PHAS was chosen from the Menu options, the display would start at 01 V, phase 1 volts. A press of the Devices right button would update the display to 02 V, phase 2 volts, then 03 V, phase 3 volts, and back to 01 V, phase 1 volts.



Reboot — press recessed button with a pencil tip, paper clip, or other small-tipped object.

POWER TO OUTLETS WILL NOT BE LOST DURING A REBOOT

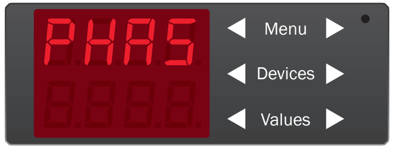


The Values buttons cycle through the data values of the selected device. So, if PHAS was chosen, the display would start at phase1 volts. A press of the Values right arrow would update the display to phase 1 amps, then phase 1 watts, etc.

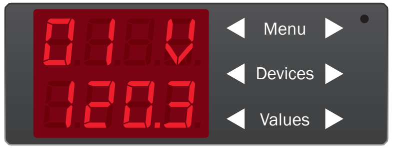
At this point, pressing the Devices right arrow would update the display from phase 1 watts to phase 2 watts, thus allowing easy scrolling through the same value for all device instances.

Navigating Power Data

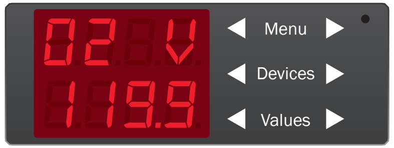
Press Menu ▶ until the desired option is visible.



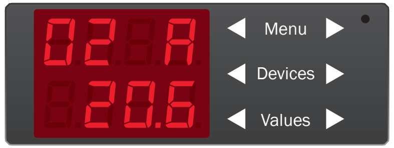
Press Devices ▶ to see the first instance of that device.



Press Devices ▶ again to see the next instance of that device.



Press Values ▶ to see the next power data value of that device.



RCM Software™
for Optima RCM™ Networked PDUs

Software Version:	2.0.x
Owner's Guide P/N:	501014-2.0.0-B

MARWAY
Power Solutions

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