Zero Touch Provisioning (ZTP) User Manual

Reference Sales Model EA-XXXX, 14667-001
Regulatory Model K-XXXX

Version 1
July 2020

CHATSWORTH PRODUCTS

800-834-4969

chatsworth.com

techsupport@chatsworth.com

Contents

Introduction

This manual provides a list of commands for accessing the Chatsworth Products (CPI) eConnect® PDU or Networked RFID Electronic Lock Kit using Zero Touch Provisioning (ZTP).

The Zero Touch Provisioning (ZTP) feature allows a user to automate the configuration of a PDU with network access. This is accomplished via ethernet connection to a DHCP server and a repository server that is accessible via HTTP, FTP, or TFTP. Once the Power Distribution Unit (PDU) makes a DHCP request to the DHCP server, the DHCP server will need to respond with both IP assignment information and a URL to use for downloading a configuration file.

This configuration will be stored in industry standard JSON format, and needs to be accessible via a TFTP, FTP, or HTTP server. Once the PDU obtains the URL from the DHCP server, the PDU will download the configuration JSON file and apply it to its own configuration, thereby completing the auto-configuration process. It can take up to a minute from the time DHCP is enabled for a PDU to complete applying the configuration.

This User Manual provides instructions for configuring the DHCP server, specifying the download URL, and creating a useable JSON file for auto-configuration. Finally, it should be noted that there are certain configuration settings that will never be applied via the ZTP auto-configuration method.

Legal Information

ZTP Exceptions

The following settings will not be applied via ZTP:

- IPv4 Network settings
  - IPv4 IP address/Subnet/Gateway
  - IPv4 DHCP vs. Static
  - IPv4 DNS settings
- IPv6 Network settings
  - IPv6 IP address/Prefix/Gateway
  - IPv6 DHCP vs. Static
  - IPV6 DNS settings
- Outlet Power States

DHCP Server Configuration for ZTP

For a successful ZTP auto-configuration to take place, the DHCP server needs to be configured to provide the PDU with the configuration file's download URL. In particular, the DHCP server needs to be configured to provide clients who supply a vendor-class-id of "ChatsworthProducts" with the configuration file's URL via the bootfile-name DHCP option. For reference, bootfile-name is option 67 in the DHCP configuration. The following example is a configuration snippet from the /etc/dhcp/dhcpd.conf file of a Linux DHCP server:

```
class "ChatsworthProducts" {
match if option vendor-class-
identifier="ChatsworthProducts";
option bootfile-
name="tftp://192.168.140.200/folder/configuration.json";
}
```

In the above example, the PDU would attempt to use the TFTP protocol to download the "configuration.json" file located in the "/folder/" directory of a TFTP server accessible at IP address 192.168.140.200.

Specifying the URL

Currently, the PDU can make use of URL's specifying the TFTP, HTTP, or FTP protocol. URL Specification is relatively straight-forward unless you need to use special characters in your URL, especially since some special characters are handled differently between Windows and Linux servers. Below are instructions for specifying each protocol's URL, as well as how to handle special characters for each:

HTTP	URL for "bootfile-name" needs to be of the form: http://hostname(IP)/path/to/file For example: option bootfile-name="http://192.168.140.200/folder/configuration.json";	
Special Character Handling	`	character needs to be escaped with '\' or encoded using '%60'
	#	character needs to be encoded using '%23'
	\	character needs to be escaped with '\' (Linux only)
FTP	URL for "bootfile-name" needs to be of the form: ftp://user:password@hostname(ip)/path/to/file For example: option bootfile-name="ftp://anonymous:@192.168.140.200/folder/configuration.json";	
Special Character Handling	`	character needs to be escaped with '\' or encoded using '%60'
	#	character needs to be encoded using '%23'
	;	character needs to be encoded using '%3B'
	\	character needs to be escaped with '\' (Linux only)
TFTP	URL for "bootfile-name" needs to be of the form: tftp://hostname(ip)/path/to/file For example: option bootfile-name="tftp://192.168.140.200/folder/configuration.json";	
Special Character Handling	`	character needs to be escaped with '\
	\	character needs to be escaped with '\' (Linux only)

Creating the JSON Configuration File

Successful auto-configuration requires that the downloaded configuration file to be in the industry standard JSON format. The PDU will only apply configuration settings that meet the following criteria:

1. The configuration setting is not one of the above-mentioned configuration settings (section B Exceptions) that will not be applied (e.g.: IPv4 network)
2. The configuration setting is defined in the JSON file
3. The configuration setting has a non-default value specified in the JSON file

The recommended way to create a JSON file intended for PDU auto-configuration is:

1. Select a PDU to use as a template PDU
2. Reset the PDU to default configuration
3. Configure the PDU with non-default settings to use as a template configuration for mass auto-configuration.
4. Utilize the PDU's RESTful Application Programming Interface (API) to obtain a copy of the PDU's configuration in JSON format. Refer to the RESTful API documentation (**https://www.chatsworth.com/en-US/Documents/Software/Bulk_API_Excel_122108.zip**).
5. You will need to make a POST with a JSON file containing user credentials against the /bulk/login endpoint of the PDU. This will give you a “sessionid” to use to make a GET against the "/bulk/config" endpoint, and that will return the PDU's configuration in JSON format. See Appendix on page 7 for example.

Once you have the PDU's configuration downloaded and saved off to a file, you can upload that file to the repository server of your choice. Then, you just need to configure your DHCP appropriately to supply the PDU with the URL needed to download the file. At this point, your network should be ready to supply newly connected PDUs with auto-configuration via the ZTP process.

Notes:

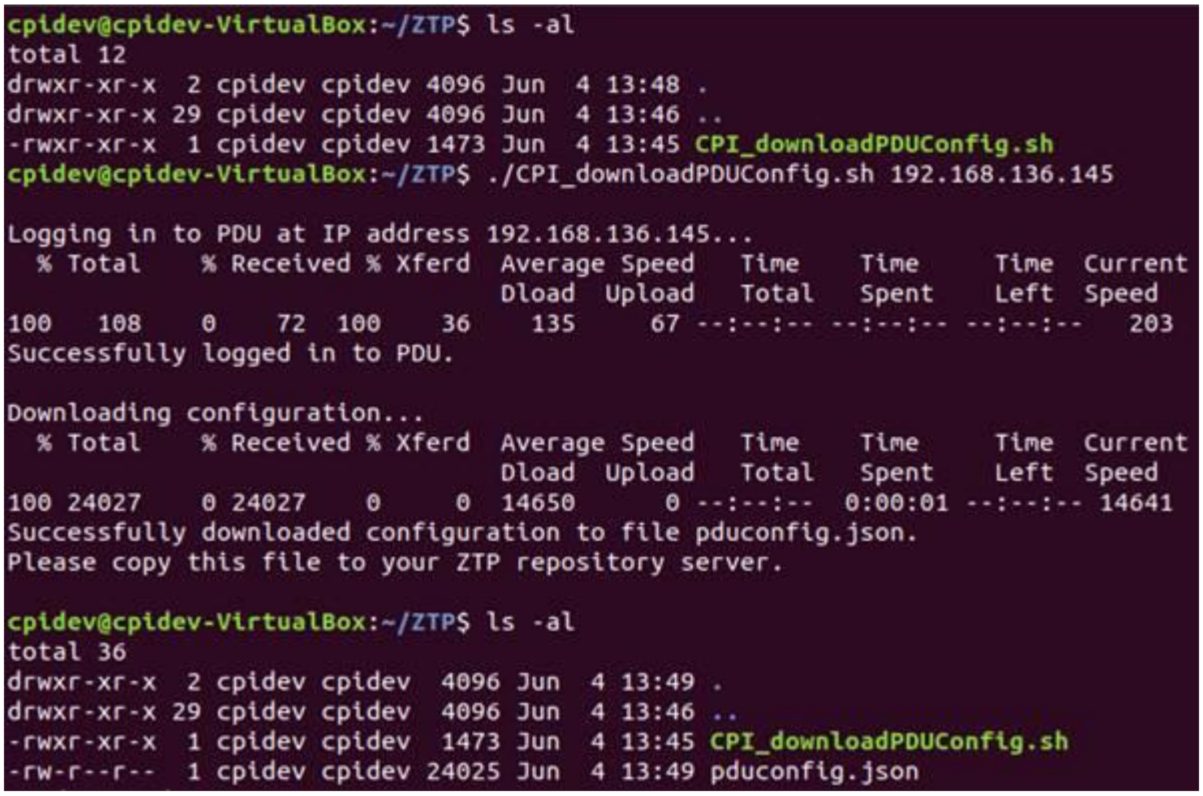
I. The template PDU needs to be the PDU with the highest functionality, number of outlets, breakers, etc., so that all settings can be configured and retrieved to be stored in the configuration json file. Lesser complex PDUs will ignore the irrelevant settings from the configuration file.

II. Avoid configuring settings that will differ across PDUs configured via ZTP. For example: PDU Name, PDU Description, Cabinet ID, etc.

III. Post-ZTP manual configurations on the PDU will be preserved across reboots *as long as the configuration file does not specify a non-default value for the respective configuration item.*

Appendix

The following screenshot shows the use of the CPI provided Linux shell script, “CPI_downloadPDUConfig.sh”. You can run this script from any Linux terminal with network access to the template PDU. When running the script, provide the PDU’s IP address an argument. The script will then login to the PDU using the default admin credentials, and download the PDU’s configuration into a file called “pduconfig.json”. This configuration file should be created at the same location where the script is run from:



```
cpidev@cpidev-VirtualBox:~/ZTP$ ls -al
total 12
drwxr-xr-x  2 cpidev cpidev 4096 Jun  4 13:48 .
drwxr-xr-x 29 cpidev cpidev 4096 Jun  4 13:46 ..
-rwxr-xr-x  1 cpidev cpidev 1473 Jun  4 13:45 CPI_downloadPDUConfig.sh
cpidev@cpidev-VirtualBox:~/ZTP$ ./CPI_downloadPDUConfig.sh 192.168.136.145

Logging in to PDU at IP address 192.168.136.145...
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   108    0    72  100    36    135     67 --:--:-- --:--:-- --:--:--   203
Successfully logged in to PDU.

Downloading configuration...
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 24027    0 24027    0     0  14650      0 --:--:--  0:00:01 --:--:-- 14641
Successfully downloaded configuration to file pduconfig.json.
Please copy this file to your ZTP repository server.

cpidev@cpidev-VirtualBox:~/ZTP$ ls -al
total 36
drwxr-xr-x  2 cpidev cpidev  4096 Jun  4 13:49 .
drwxr-xr-x 29 cpidev cpidev  4096 Jun  4 13:46 ..
-rwxr-xr-x  1 cpidev cpidev  1473 Jun  4 13:45 CPI_downloadPDUConfig.sh
-rw-r--r--  1 cpidev cpidev 24025 Jun  4 13:49 pduconfig.json
```

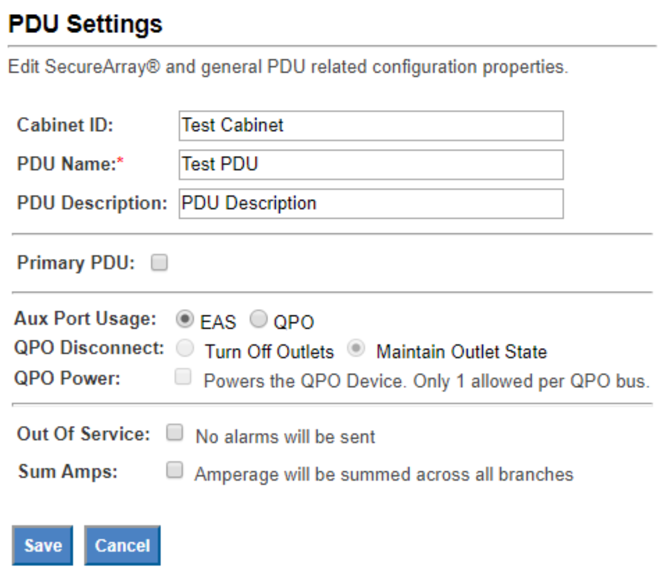
Content of “CPI_downloadPDUConfig.sh”:

```
#!/bin/bash
if [ -z "$1" ] ; then
    echo "Login Error: PDU IP address unspecified."
    exit
fi
ip=$1
#-----------------------------------------------
# Setup - Login to the system and retrieve our sessionid
echo ""
echo "Logging in to PDU at IP address $ip..."
reply=$(curl --header "Content-Type:application/json" -X POST --data
"{\"user\":\"admin\", \"password\":\"admin\"}" http://$ip/bulk/login)
while [ $? -ne 0 ]
do
    echo "Failed to POST to http://$ip/bulk/login."
    sleep 1
    echo "Re-trying..."
    reply=$(curl --header "Content-Type:application/json" -X POST --data
"{\"user\":\"admin\", \"password\":\"admin\"}" http://$ip/bulk/login)
done
result=$(echo $reply | jq .resultCode)
if [ $result -ne 0 ]; then
    echo ""
    echo -n "Error: "
    echo $reply | jq .message
    exit 1
fi
sessionid=$(echo $reply | jq -r .sessionid)
echo "Successfully logged in to PDU."
echo ""
##########################################################################
#######
echo "Downloading configuration..."
PDUConfig=$(curl -H "SessionID: $sessionid" --header "Content-Type:application/json"
-X GET http://$ip/bulk/config)
while [ -z "$PDUConfig" ]
do
    sleep 1
    PDUConfig=$(curl -H "SessionID: $sessionid" --header "Content-
Type:application/json" -X GET http://$ip/bulk/config)
done
echo "Successfully downloaded configuration to file pduconfig.json."
echo "Please copy this file to your ZTP repository server."
echo ""
if [ -f "./pduconfig.json" ]; then
    rm ./pduconfig.json
fi
echo $PDUConfig >> pduconfig.json
```

The pduconfig.json file will contain the PDU’s entire configuration represented in JSON format. There should be no need to make any manual configuration edits. If you wish to make changes to the file’s configuration, it is recommended to make the configuration change on the template PDU and re-download the configuration into a new .json file. Once you have downloaded the pduconfig.json file, you can copy it straight to your repository server and use it for ZTP auto-configuration provided the necessary configuration on the DHCP server has been done.

Examples of ZTP Configuration

The following example will show the conditional application of the ZTP auto-configuration. As stated above, only settings with non-default values will be applied during ZTP. The below screenshot shows a PDU prior to ZTP auto-configuration:



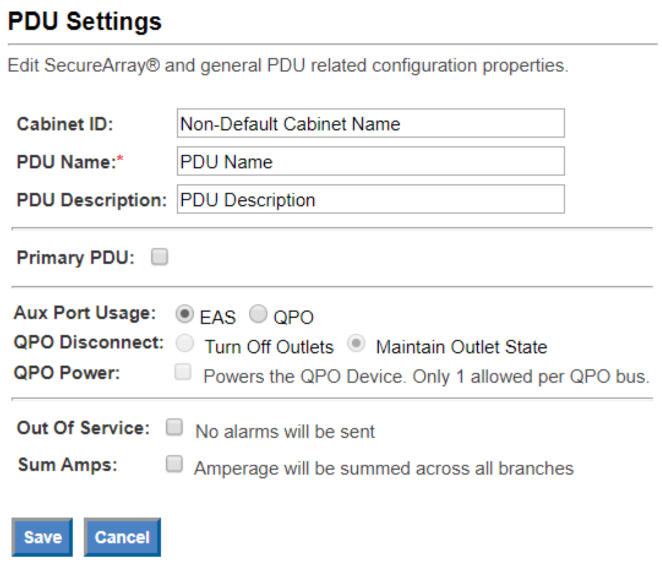
PDU Settings

Edit SecureArray® and general PDU related configuration properties.

Cabinet ID:	Test Cabinet
PDU Name:*	Test PDU
PDU Description:	PDU Description
Primary PDU:	☐
Aux Port Usage:	[✓ EAS] QPO
QPO Disconnect:	Turn Off Outlets [✓ Maintain Outlet State]
QPO Power:	☐ Powers the QPO Device. Only 1 allowed per QPO bus.
Out Of Service:	☐ No alarms will be sent
Sum Amps:	☐ Amperage will be summed across all branches

As you can see, it has had its Cabinet ID and PDU Name manually edited at some point.

The screenshot below shows the configuration of a PDU to be used as a template for auto-configuration:

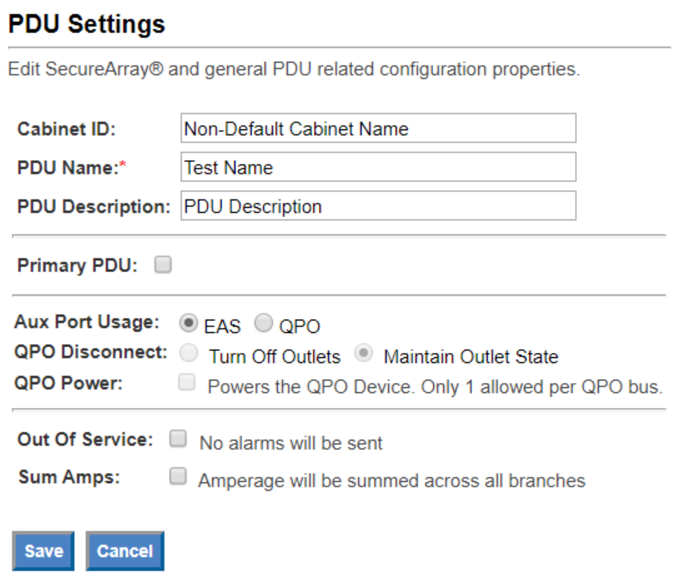


PDU Settings

Edit SecureArray® and general PDU related configuration properties.

Cabinet ID:	Non-Default Cabinet Name
PDU Name:*	PDU Name
PDU Description:	PDU Description
Primary PDU:	☐
Aux Port Usage:	[✓ EAS] QPO
QPO Disconnect:	Turn Off Outlets [✓ Maintain Outlet State]
QPO Power:	☐ Powers the QPO Device. Only 1 allowed per QPO bus.
Out Of Service:	☐ No alarms will be sent
Sum Amps:	☐ Amperage will be summed across all branches

Notice it has a non-default value for “Cabinet ID”, but still has the default values for “PDU Name” and “PDU Description”. If this PDU’s configuration was used as a ZTP configuration template, then it would produce the following results when applied to the first PDU:


PDU Settings

Edit SecureArray® and general PDU related configuration properties.

Cabinet ID:	Non-Default Cabinet Name
PDU Name:*	Test Name
PDU Description:	PDU Description
Primary PDU:	☐
Aux Port Usage:	● EAS ○ QPO
QPO Disconnect:	○ Turn Off Outlets ● Maintain Outlet State
QPO Power:	☐ Powers the QPO Device. Only 1 allowed per QPO bus.
Out Of Service:	☐ No alarms will be sent
Sum Amps:	☐ Amperage will be summed across all branches

The PDU has applied the non-default “Cabinet ID” value, but the PDU has retained its customized “PDU Name” value of “Test Name”. This is because the template PDU had a non-default value for its “Cabinet ID” value, therefore, this non-default value was applied to the PDU. However, the template PDU still had the default value for the “PDU Name” configuration item, and therefore, the “PDU Name” value was not overwritten during ZTP auto-configuration.

Frequently Asked Questions

Q: What if I don't want to worry about accidental ZTP auto-configuration overwriting customized settings on my PDUs in scenarios such as reboots and DHCP re-binds?

A: You can disable DHCP on the PDU once it has received the necessary ZTP auto-configuration, and this will prevent any attempts at ZTP auto-configuration. If DHCP is needed for the PDU's continuous functionality, then preventing any unwanted ZTP auto-configuration will be entirely environmentally dependent. ZTP auto-configuration is ideally intended for first boot-up and initial configuration. Because of this, one recommendation is to preserve your environment's ZTP configuration until all newly installed PDU's have been configured. After that, you can change your DHCP server's configuration to no longer point to the ZTP configuration JSON file. This will prevent your PDU's from attempting ZTP auto-configuration while still preserving remaining DHCP functionality.

Q: Why is the ZTP configuration not working?

A; There are a lot of reasons the ZTP process can break down. Ensure that the following are true:

- PDU is enabled for DHCP over IPv4
- PDU has network connectivity and a successful DHCP transaction has taken place (i.e. PDU has a DHCP assigned IPv4 address)
- DHCP server configuration is correct.
  - The class is "ChatsworthProducts"
  - The match line specifies a "vendor-class-identifier" of "ChatsworthProducts"
  - The "bootfile-name" syntax is appropriate for the protocol specified
  - The "bootfile-name" specifies the correct URL for the configuration file
- The repository server specified (TFTP, FTP, HTTP) is running
- The PDU has network access to the repository server after receiving DHCP network configuration
- The specified .json configuration file is present in the repository server
- The specified .json configuration file contains no syntax errors