# FreeZTP

A zero-touch provisioning system built for Cisco Catalyst switches.

---

## VERSION

The version of FreeZTP documented here is: **v0.1.0 Beta**

---

## TABLE OF CONTENTS



---

## WHAT IS FREEZTP

FreeZTP is a dynamic TFTP server built to automatically configure Cisco Catalyst switches upon first boot (Zero-Touch Provisioning). FreeZTP does this using the 'AutoInstall' feature built into Cisco IOS and automatically enabled by default. FreeZTP configures switches with individual, templatized configurations based upon the unique ID of the switch (usually the serial number).

---

## REQUIREMENTS

OS: **CentOS7**

Interpreter: **Python 2.7.5+**

---

## TERMINOLOGY

Due to the unique nature of how FreeZTP works and performs discovery of switches, there are a few terms you will need to know to understand the application.

- **Template**
  - FreeZTP relies on the Jinja2 templating standard to take a common Cisco IOS configuration and templatize it: creating variables (with the `{{ i_am_a_variable }}` syntax) in the template where unique values can be inserted for a specific switch upon configuration push.
  - FreeZTP uses two different templates: the 'initial-template', and the 'final-template'. The initial-template is used to set the switch up for discovery, the final-template is used to push the final configuration once the discovery is complete and the switch has been identified (this will make more sense in the **ZTP Process** section).
- **Keystore**
  - The counterpart to the template (specifically: the final-template) is the keystore. The keystore is the part of the ZTP configuration which holds the unique configuration values for specific switches (or for an array of switches). The keystore provides those values for the merge of the final-template once the switch has been identified by the discovery process.
  - **Keystore ID**
    - A Keystore ID is the named identifier for a specific store which holds a set of key-value pairs.
  - **Keystore Key**
    - A Keystore Key is the key side of a certain key-value pair which resides under a named Keystore ID.
  - **Keystore Value**
    - A Keystore Value is the value side of a certain key-value pair which resides under a named Keystore ID.
  - **Keystore Hierarchy**
    - The hierarchy of the Keystore works as follows: A Keystore ID can contain multiple (unique) keys, each key with a different value. The Keystore can contain multiple IDs, each with its own set of key-value pairs.
- **ID Arrays**
  - An ID Array is a method of mapping one or more real switch IDs (ie: serial numbers) to a specific keystore. Multiple real IDs can be mapped to the same Keystore ID, which comes in handy when building a configuration for a switch stack (which could take on the serial number of any of the member switches when it boots up).
  - The ID array has two pieces:
    - The **Array Name** is the name of the specific array. The Array Name must match a Keystore ID in order to pull values from that keystore.
    - The **Array ID List** is a list of real switch IDs (serial numbers) which, when searched for, will resolve to the Array Name before mapping to a Keystore ID. When configuring an IDArray in the CLI, each ID in the list is separated by a space.

---

# ZTP PROCESS

FreeZTP relies on the 'AutoInstall' function of a Cisco Catalyst switch to configure the switch upon first boot. The process followed to configure the switch is outlined below.

1. The Catalyst switch is powered on (or rebooted) with no startup-configuration. The switch should be connected (via one of its ports) to another switch on a VLAN which is ready to serve DHCP. The

DHCP scope should have DHCP OPTION 66 configured with the IP address (string) of the ZTP server.

2. Once the operating system is loaded on the switch and it completes the boot-up process, it will start the AutoInstall process
    - 2.1 The switch will enable all of its ports as access ports for interface Vlan1.
    - 2.2 The switch will enable interface (SVI) Vlan1 and begin sending out DHCP requests from interface Vlan1.
    - 2.3 Once the switch receives a DHCP lease with a TFTP server option (option 66), it will send a TFTP request for a file named "network-confg".
3. When the request for the "network-confg" file is received by the ZTP server, it performs an automatic merge with the initial-template:
    - 3.1 The `{{ autohostname }}` variable is filled by an internally generated hexadecimal temporary name (example: ZTP-22F1388804). This temporary name is saved in memory by the ZTP server for future reference because the switch will use this name to request a new TFTP file in a later step.
    - 3.2 The `{{ community }}` variable is filled with the value set in the `community` configuration field
    - 3.3 This merged configuration is passed to the Cisco switch as the "network-confg" file. The switch will load it into its active running-config and proceed to step 5.
4. After the file is passed to the switch, the ZTP server initiates a SNMP discovery of the switch
    - 4.1 The SNMP request targets the source IP of the initial TFTP request for the "network-confg" file
    - 4.2 The SNMP request uses the value of the `community` configuration field as the authentication community (which the switch should honor once it loads the configuration from the "network-confg" file)
    - 4.3 The SNMP request uses the OID from the `snmpoid` configuration field which, by default, is the OID to obtain the serial number of the switch.
    - 4.4 Once the SNMP query succeeds, the ZTP server maps the serial number of the discovered switch to its temporary hostname generated in step 3.
5. After the switch loads the "network-confg" file into its running-config, it sends out a new TFTP request to the ZTP server
    - 5.1 The file name for the new TFTP request is based upon the hostname passed to the switch in the initial ("network-confg") file (example filename: "ZTP-22F1388804-confg")
6. The ZTP server receives the new TFTP request and uses the requested filename to identify the requesting switch
    - 6.1 The ZTP server receives a TFTP request for a file based on a temporary hostname (example filename: "ZTP-22F1388804-confg").
    - 6.2 It uses this hostname to look up the discovery information (serial number) of the requesting switch (it was saved in step 4.4). This discovery information/ID can be referred to as the "real ID" of the switch. Once the real ID of the requesting switch is known, the ZTP server begins the process of merging the final configuration for the switch using the final-template and the Keystore/IDArrays.
7. The ZTP server uses the switch's real ID to assemble the final configuration and pass it to the switch to be loaded into the running-config
    - 7.1 The real ID is used to search through all of the Keystore IDs to see if one of them matches the real ID. If a Keystore ID matches, then the server proceeds to 7.3 with that Keystore ID.
    - 7.2 If there is no match between the real-ID and a Keystore ID, then the server looks to the IDArrays for a match. It searches through the ID list in each IDArray, once a match is found, the

server resolves the real-ID to the IDArray Name and re-searches the Keystore IDs for a match using the resolved IDArray name. Once a match is found, the server continues to step 7.3 with the matched Keystore ID.
    - 7.3 Once a candidate Keystore ID is found the server performs a Jinja2 merge between the final-template and the key-value pairs in the matched Keystore ID.
    - This merged configuration is then passed to the switch via TFTP with the filename requested by the switch ("ZTP-22F1388804-confg" in this case).
8. The switch loads this final configuration into its active running-config and begins normal operations
    - 8.1 If you configured static IP addresses in the final-template, then the switch starts using those static IPs and can be remotely accessible via them (assuming you also included config for AAA and SSH)
    - 8.2 The switch does not save the new configurations into its startup-config. That has to be done manually.

---

# INSTALLATION

The installation of FreeZTP is quick and easy using the built-in installer. Make sure you are logged in as root or are able to `sudo su` to install and operate FreeZTP.

1. Install OS with appropriate IP and OS settings and update to latest patches (recommended)
    - Check out the CentOS Minimal Server - Post-Install Setup page for help with some of the post-OS-install configuration steps.
2. Download the FreeZTP repository (may require access to the GitHub ConvergeOne organization)
3. Change to the directory where the FreeZTP main code file (ztp.py) is stored: `cd freeztp`
4. Run the FreeZTP program in install mode to perform the installation: `python ztp.py install`. Make sure the machine has internet access as this process will download and install several packages for this to happen.
    - FreeZTP will perform the install of the packages and services for full operation.
    - The installation will also install a CLI helper script. You will need to logout and log back into your SSH session to activate this helper script.

---

# COMMAND INTERFACE

The FreeZTP service runs in the background as a system service. All commands to the FreeZTP service start with `ztp` at the command line. For example: you can check the status of the background service using the command `ztp show status`, you can check the current configuration of the ZTP server with the command `ztp show config`.

The command interface is fully featured with helpers which can be seen either by hitting ENTER with an incomplete command in the prompt, or by using the TAB key to display available options/autocomplete the command (similar to a Cisco IOS command line, but without the use of the question mark).

All commands which change the ZTP configuration use the `set` or `clear` arguments. Commands issued

with the `set` argument will overwrite an existing configuration item if that item already exists. The `clear` argument allows you to remove configuration items.

The initial and final template configurations are entered as multi-line text blocks. To facilitate this, you must specify a delineation character in the `set` command. As an example, you will issue the command `ztp set final-template ^` where the carat (`^`) character is set as the delineation character. After that command is issued, you can paste in the multi-line Cisco IOS template text. Once finished, enter that delineation character (`^` in this case) on a line by itself to exit the text block entry mode.

Below is the CLI guide for FreeZTP. You can see this at the command line by entering `ztp` and hitting ENTER (after installation).

```
-------------------------------------------------------------------------------------------
                    ARGUMENTS                    |                                DESCRIPTI
-------------------------------------------------------------------------------------------
 - run                                           |  Run the ZTP main program in shell mode be
-------------------------------------------------------------------------------------------
 - install                                       |  Run the ZTP installer
-------------------------------------------------------------------------------------------
 - show (config|run)                             |  Show the current ZTP configuration
 - show status                                   |  Show the status of the ZTP background ser
 - show version                                  |  Show the current version of ZTP
-------------------------------------------------------------------------------------------
 - set suffix <value>                            |  Set the file name suffix used by target w
 - set initialfilename <value>                   |  Set the file name used by the target duri
 - set community <value>                         |  Set the SNMP community you want to use fo
 - set snmpoid <value>                           |  Set the SNMP OID to use to pull the targe
-------------------------------------------------
 - set initial-template <end_char>               |  Set the initial configuration j2 template
 - set final-template <end_char>                 |  Set the final configuration j2 template p
-------------------------------------------------
 - set keystore <id/arrayname> <keyword> <value>  |  Create a keystore entry to be used when m
 - set idarray <arrayname> <id_#1> <id_#2> ...    |  Create an ID array to allow multiple real
-------------------------------------------------------------------------------------------
 - clear keystore <id> (all|<keyword>)           |  Delete an individual key or a whole keyst
 - clear idarray <arrayname>                     |  Delete an ID array from the configuration
-------------------------------------------------------------------------------------------
 - request merge-test <id>                       |  Perform a test jinja2 merge of the final
 - request initial-merge                         |  See the result of an auto-merge of the in
-------------------------------------------------------------------------------------------
 - service (start|stop|restart|status)           |  Start, Stop, or Restart the installed ZTP
-------------------------------------------------------------------------------------------
 - version                                       |  Show the current version of ZTP
-------------------------------------------------------------------------------------------
```

The following is the default configuration seen on the ZTP server after installation when doing a `ztp show config`.

```
[root@ZTP ~]# ztp show config
!
ztp set suffix -confg
```

```
ztp set initialfilename network-confg
ztp set community secretcommunity
ztp set snmpoid 1.3.6.1.2.1.47.1.1.1.1.11.1000
!
!
!
ztp set idarray STACK1 SERIAL1 SERIAL2 SERIAL3
!
!
!
!
ztp set keystore SERIAL100 vl1_ip_address 10.0.0.201
ztp set keystore SERIAL100 hostname ACCESSSWITCH
!
ztp set keystore STACK1 vl1_ip_address 10.0.0.200
ztp set keystore STACK1 hostname CORESWITCH
!
!
#########################################################
ztp set initial-template ^
hostname {{ autohostname }}
!
snmp-server community {{ community }} RO
!
end
^
!
!
!
#########################################################
ztp set final-template ^
hostname {{ hostname }}
!
interface Vlan1
 ip address {{ vl1_ip_address }} 255.255.255.0
 no shut
!
ip domain-name test.com
!
username admin privilege 15 secret password123
!
aaa new-model
!
!
aaa authentication login CONSOLE local
aaa authorization console
aaa authorization exec default local if-authenticated
!
crypto key generate rsa modulus 2048
!
ip ssh version 2
!
line vty 0 15
login authentication default
transport input ssh
line console 0
login authentication CONSOLE
end
^
[root@ZTP ~]#
```

## CONTRIBUTING

If you would like to help out by contributing code or reporting issues, please do!

Visit the GitHub page (https://github.com/convergeone/freeztp) and either report an issue or fork the project, commit some changes, and submit a pull request.