# Using Perl Scripts

Cisco E-DI supports perl scripting through the CLI. This feature automates many of the server and network administration tasks. This section explains how to enable and disable the perl scripting service, and how to use Cisco E-DI Perl API for daily tasks.

When a perl script is implemented by an administrator or a user, the script automatically inherits the user's security privileges and the user's operational context (either server or network).

Each invocation of a perl script will use an additional CLI session for the duration of that script implementation.

This section includes the following information:

- Setting Up a Perl Scripting Session
- Perl Script Examples

## Setting Up a Perl Scripting Session

Table 10-1 details how to setup a perl scripting session.

***Table 10-1 Commands to Setup a Perl Scripting Session***

| Action | Command |
|---|---|
| To enable perl scripting service for the CLI. By default the perl scripting service is off on Cisco E-DI. | `[SRV:/server](config)#` **`service perl-scripting`** |
| To open a vi editor.<br>Enter the perl script. Save the script | `[SVR:/server]#` **`edit filename.pl`** |
| To run the perl script filename.pl located in the ./serverdirectory. | `[SRV:/server]#` **`perl`** `/server/filename.pl` |
| To disable the perl scripting service for the CLI. | `[SRV:/server](config)#` **`no service perl-scripting`** |

# Perl Script Examples

This section includes the following examples:

- Verifying a Perl Script
- Verifying the HTTP Server is Enabled on Cisco IOS Devices
- Verifying NTP Server Configuration and Enforcing the Policy
- Verifying Password Encryption is Disabled on Cisco IOS Devices

## Verifying a Perl Script

Perl script to verify that any perl script that is run returns correct **getresponse**, **getstatuscode** and **getstatusmessage** messages:

```
use lib '/perlapi';
use JMSPERLAPI;
$| = 1;
my $api= JMSPERLAPI->getAPI ();
$api->executeCMD ("show version");
$code=$api->getStatusCode ();
$output=$api->getResponse ();
$stMesage=$api->getStatusMessage();
print "\status Code is: $code \n";
print "\Response from JMS is: \n $output \n";
print "\n Status Message is: $stMesage \n";
$api->closeAPI ();
```

When this perl script is run, the expected output is as follows:

```
admin@edi-test-dell_blr_19[SVR:/server]# perl /server/def.pl
status Code is: 0
Response from JMS is:
 Cisco Enhanced Device Interface Server, version 2.0
Copyright (C) 2005 Cisco Systems Inc. All rights reserved.
Compiled on 3-June-2005 by buildserver

Uptime 0 days 5:32:46, effective user admin
Server is running on system edi-test-dell_blr_19 with OS Linux (i386) 2.6.9-5.ELsmp
Number of processors detected: 1
Java runtime environment version 1.4.2_03

Status Message is: OK

!Status Code: 0
```

## Verifying the HTTP Server is Enabled on Cisco IOS Devices

This script checks whether the HTTP server is enabled on any of the Cisco IOS devices that are being managed by Cisco E-DI.

```
use lib '/perlapi';
use JMSPERLAPI;
$| = 1;
my $api= JMSPERLAPI->getAPI ();

($code, $output) = $api->executeCMD ("show devices");
if ($code != 0) {
    print "failed to retrieve device list.\n";
    exit;
}

@temp = split(/\n/,$output);  # Taking ouput into an array
foreach $line (@temp) {
    if(defined($line)){
        $line=~m/(\s+|\S\s)(\d+\.\d+\.\d+\.\d+)(.*)/;
        if($1 !~m/\*/) {
            # if it is a supported device

            if (defined($2)) {
                $device=$2;
                ($code, $cmdout) = $api->executeCMD ("network $device");
                if ($code != 0){
                    print "Could not change context to $device";
                    next;
                }

                ($code, $devOS) = $api->executeCMD ("show report software");
                $devOS =~m/(\s)(\d+\.\d+\.\d+\.\d+)(\s+)(\w+)(\s+)(\w+)(.*)/;
                $OS = "$6";
                if( $OS =~m/IOS/){
                    # If IOS, Check whether http is enabled
                    ($code, $runConfig) = $api->executeCMD ("show running-config | include
        \"ip http server\"");
                    if($runConfig =~/\s\sip\shttp\sserver/){
                        print"Http Service is enabled on the device: $device\n";
                   }
                }
            }
        }
    }
}

$api->closeAPI ();
```

## Verifying NTP Server Configuration and Enforcing the Policy

This script is used to verify bulk configuration changes to NEs with Cisco IOS Version. The script parses data from the **show devices** output and displays the supported devices including Cisco IOS and CatOS devices.

For Cisco IOS devices, it first checks if the NTP server is configured. If the NTP server is not configured, the script will configure the server with the specified IP address.

For devices where the NTP server is already configured, it checks the IP address with the specified one. If these do not match, the script will configure the server with the specified IP address.

The variables are as follows:

- @Device_info—Displays the parsed **$output** content.
- @Dev_List—Lists the Cisco E-DI supported devices including Cisco IOS and CatOS.
- @IOS_DEVICES—Lists the Cisco E-DI supported Cisco IOS Devices.
- @CatOS_DEVICES—Lists the Cisco E-DI supported CatOS Devices.

```
use lib '/perlapi';
use JMSPERLAPI;
$| = 1;

# IP Address of the NTP Server (Change the value below to required address)
$address = "1.1.1.1";

$dev = 0;
$ios = 0;
$catos = 0;

my $api= JMSPERLAPI->getAPI();
($code, $output) = $api->executeCMD("show devices ");
$output =~ s/^\s+//;

###Parsing The Output of 'show Devices'

@Device_info = split(/[\n]/, $output);

###Read Devices One by One and Extract EDI
###Supported Devices including IOS and CatOS

foreach $i (@Device_info) {
    if(defined($i)) {
        $i =~ m/(\s+|\S\s)(\d+\.\d+\.\d+\.\d+)(.*)/;
        if($1 !~m/\*/) {
            if(defined($2)) {

                @Dev_List[$dev] = "$2";
                ($code, $Ver_OS) = $api->executeCMD("show report software | include
                $Dev_List[$dev] ");

                ###Followind code Seperats the IOS and CatOS Devices
                $Ver_OS =~m/(\s)(\d+\.\d+\.\d+\.\d+)(\s+)(\w+)(\s+)(\w+)(.*)/;

                ### For IOS Devices Check the following Conditions
                ### 1.Whether NTP Server configured if so with specified IP
                ### if not Remove the existed IP and Configure with specified IP
                ### 2.If no NTP server Configured ,Configure with specified IP

                $tmp = "$6";
                if( $tmp =~m/IOS/) {
                    @IOS_DEVICES[$ios++] =  "$Dev_List[$dev]";
                    ($code, $cmdout) = $api->executeCMD("network $Dev_List[$dev]");
                    ($code, $check_ntp) = $api->executeCMD("show running-config | include
        ntp");
                    if($check_ntp  =~ m/(\s+)(ntp\sserver)(.*)/)  {
                        $existed_ip ="$3";
                        if($3 !~ m/($address)/)  {
                            print "NTP Server Configuration changing from $3\n";
                            print "  to $address on $Dev_List[$dev], please wait ...\n\n";
```

```
                          ($code, $cmdout) = $api->executeCMD("config setup");
                          ($code, $cmdout) = $api->executeCMD("configure");
                          if ($code != 0) {
                              print "Unable to enter configure mode of $Dev_List[$dev],
skipping the device";
                              next;
                         }
                          ($code, $cmdout) = $api->executeCMD("no ntp server
$existed_ip");
                          ($code, $cmdout) = $api->executeCMD("ntp server $address");
                          ($code, $cmdout) = $api->executeCMD("exit");
                          ($code, $status) = $api->executeCMD("commit"); #  Committing
the change to the device
                          if($status =~ m/\[OK(.*)\]/) {
                              print "\n NTP Server Successfully Configured on
$Dev_List[$dev]\n\n";
                          } else {
                              print "\n No Response From  Device....$Dev_List[$dev]\n";
                          }

                          ($code, $cmdout) = $api->executeCMD("exit");
                          ($code, $cmdout) = $api->executeCMD("server");
                      } else {
                          print "Specified NTP Server Configuration Already Present On
:$Dev_List[$dev]\n\n";
                          ($code, $cmdout) = $api->executeCMD("server");
                      }
                  } elsif ($check_ntp =~ m/(\[$Dev_List[$dev]\]\sWARNING:)(.*)/) {
                      print "Device $Dev_List[$dev] offline\(Running config not
available\).....!\n\n";
                      ($code, $cmdout) = $api->executeCMD("server");
                  } else {
                      print "Configuring NTP Server on $Dev_List[$dev]...plz Wait.! \n";
                      ($code, $cmdout) = $api->executeCMD("config setup");
                      ($code, $cmdout) = $api->executeCMD("configure");
                      ($code, $cmdout) = $api->executeCMD("ntp server $address");
                      ($code, $cmdout) = $api->executeCMD("exit");
                      ($code, $status) = $api->executeCMD("commit");

                      if($status =~ m/\[OK(.*)\]/) {
                          print "NTP Configured successfully....On : $Dev_List[$dev]
\n\n";
                      } else {
                          print "\nfailed to configure device: $Dev_List[$dev]\n\n";
                     }
                      ($code, $cmdout) = $api->executeCMD("exit");
                      ($code, $cmdout) = $api->executeCMD("server");
                  }
              } else {
              @CatOS_DEVICES[$catos++] =  "$Dev_List[$dev]"; }
          }
      }
  }
  $dev++;
}


$api->closeAPI();

1;
```

## Verifying Password Encryption is Disabled on Cisco IOS Devices

This script can be used to check whether password encryption service is disabled on Cisco IOS devices that are being managed by Cisco E-DI.

```
use lib '/perlapi';
use JMSPERLAPI;
$| = 1;

my $api= JMSPERLAPI->getAPI();

($code, $output) = $api->executeCMD ("show devices");

@temp = split(/\n/,$output);
foreach $line (@temp) {
    if(defined($line)){
        $line=~/(\s)(\d+\.\d+\.\d+\.\d+)(.*)/;
        if($2 =~m/(\d+\.\d+\.\d+\.\d+)(.*)/){
            if (defined($1)) {
                $device=$1;
                print "Verifying for Device: $device....\n";
                $api->executeCMD ("network $device");
                if ($code != 0) {
                    print "Unable to change the context to device $device\n";
                    next;
                }
                $api->executeCMD ("show report software");
                $devOS = $api->getResponse();
                $devOS =~m/(\s)(\d+\.\d+\.\d+\.\d+)(\s+)(\w+)(\s+)(\w+)(.*)/;
                $OS = "$6";
                if( $OS =~m/IOS/) {
                    ($code, $output) = $api->executeCMD ("show running-config | include
        \"no service password-encryption\"");
                    if($output=~/no\sservice\spassword-encryption/) {
                       print"Password Encryption is Disabled on the device: $device\n";
                    }
                }
            }
        }
    }
}

$api->closeAPI();
```