

CloudVision Mastery Workshop

CloudVision Actions

Lab Guide 5



ACTIONS

Actions can be used within a change control to run a series of commands on devices. There are a number of built-in actions that can be selected when configuring a change control operation or creating a change control template. You can also create custom actions by creating arguments and writing a Python 3 script to manipulate the arguments.

When configuring a change control, you'll select the dynamic values of an argument, like the device the action will run against and the image to install. When creating a custom action, you'll use dynamic arguments, like DeviceID, to enable users to select values when configuring a change control.

Create an Action

1. Navigate to **Provisioning > Actions**.
2. To create a new action, click on the “+ **New Action**” button at the top-right of the screen.

Provisioning

Network Provisioning

Configlets

Image Repository

Upload and Download...

Tasks

Actions

Change Control

Action Bundles

Templates

Studios

Workspaces

Snapshot Configuration

Public Cloud Accounts

Tags

Zero Touch Provisioning

Actions

Create and manage actions for use in action bundles and change control operations

Search actions

Custom Actions

Bounce Interface
Change Control
1 minute ago

Generate Campus Node IDs
Studio Autofill | Autofill Campus Studio Node IDs
1 month ago

Generate Data Centers Node IDs
Studio Autofill | Autofill L3LS Studio Node IDs
1 month ago

Port Auto Description
Change Control
1 minute ago

Restart TerminAttr
Change Control
1 minute ago

Built-In Actions

Capture CLI Snapshot
Change Control Built-in | Run a snapshot that has been defined in Snapshot Configuration.

Clean Flash
Change Control Built-in | Delete device files using a file spec and file glob. This works by collecting the existing files on the de...

Enter BGP Maintenance Mode
Change Control Built-in | Pair this action with Exit BGP Maintenance Mode to run specific tests detailed in the EOS User Manua...

Check MLAG Health
Change Control Built-in | Run this action before and after the main change control action to ensure that an MLAG device is rea...

Download File
Change Control Built-in | Download image and extension files to a device flash directory. The files specified must be available I...

Enter ZTP
Change Control Built-in | The Enter ZTP action deletes the startup-config, zerotouch-config and reloads the device to force th...

Execute Task

Exit BGP Maintenance Mode

3. A **“New Action”** pop-up will appear. In the **“Name”** field, type your initials followed by **“- Clear Counters.”**
4. Click the blue **“Save”** button to save the new action.

Provisioning

Network Provisioning

Configlets

Image Repository

Upload and Download...

Tasks

Actions

Change Control

Action Bundles

Templates

Studios

Workspaces

Snapshot Configuration

Public Cloud Accounts

Tags

Zero Touch Provisioning

Actions

Create and manage actions for use in action bundles and ch

Search actions

Custom Actions

Bounce Interface
Change Control
20 hours ago

Generate Data Centers Node IDs
Studio Autofill | Autofill L3LS Studio Node IDs
1 month ago

Port Auto Description
Change Control
20 hours ago

Restart TerminAttr
Change Control
20 hours ago

Built-In Actions

Capture CLI Snapshot
Change Control Built-in | Run a snapshot that has been defined in Snapshot Configuration.

Clean Flash
Change Control Built-in | Delete device files using a file spec and file glob. This works by collecting the existing files on the de...

Enter BGP Maintenance Mode
Change Control Built-in | Pair this action with Exit BGP Maintenance Mode to run specific tests detailed in the EOS User Manua...

Execute Task
Change Control Built-in | Run this action with a pre-defined TaskID to execute the specified network changes.

Check MLAG Health
Change Control Built-in | Run this action before and after the main change control action to ensure that an MLAG device is rea...

Download File
Change Control Built-in | Download image and extension files to a device flash directory. The files specified must be available I...

Enter ZTP
Change Control Built-in | The Enter ZTP action deletes the startup-config, zerotouch-config and reloads the device to force th...

Exit BGP Maintenance Mode
Change Control Built-in | Pair this action with Enter BGP Maintenance Mode to run specific tests detailed in the EOS User Man...

Interface Cable Test
Change Control Built-in | Run this action to invoke the EOS I1 cable test feature. This diagnostic is useful to determine if a Bas...

New Action

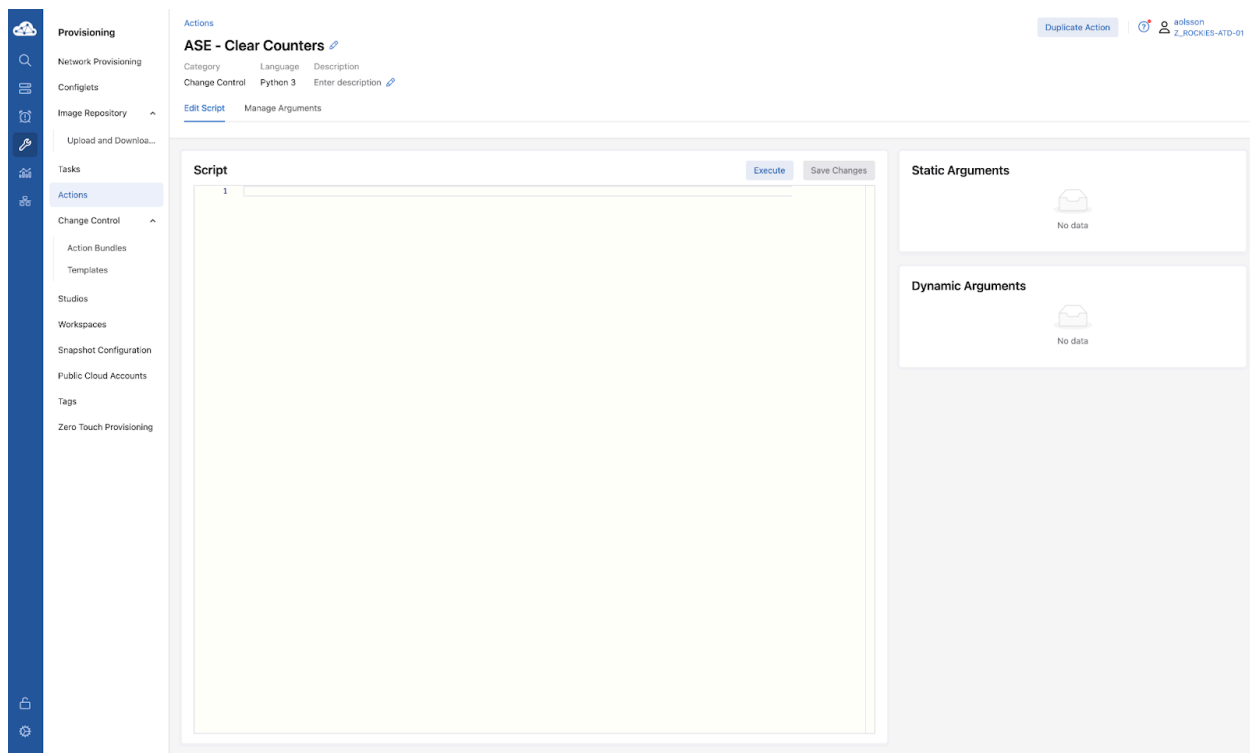
* Name
ASE - Clear Counters

Description

* Action Type
Change Control

Cancel Save

5. You'll then be redirected to the action configuration page for the new action.



6. We'll first want to create some arguments for the Device and Interface for the action to be run against. To do this, click the **"Manage Arguments"** button just below the action's name

The screenshot shows the Arista Provisioning web interface. On the left is a navigation sidebar with options like Provisioning, Network Provisioning, Configlets, Image Repository, Upload and Download, Tasks, Actions (selected), Change Control, Action Bundles, Templates, Studios, Workspaces, Snapshot Configuration, Public Cloud Accounts, Tags, and Zero Touch Provisioning. The main content area is titled 'Actions' and shows 'ASE - Clear Counters' with a 'Duplicate Action' button and a user profile 'adison_2_R00KES-ATD-01'. Below this is the 'Manage Arguments' section, which includes a 'Save Changes' button and a table with columns: Name, Type, Default, Description, Required, and Hidden. The table currently shows 'No data' and has links for '+ Static Argument' and '+ Dynamic Argument' at the bottom.

7. Create a new argument by selecting the “+ **Dynamic Argument**” at the bottom of the “**Manage Arguments**” section.

NOTE: Keep an eye on the Arguments and Variables as they are case sensitive.

8. A new line for the new argument will appear in the “**Manage Arguments**” section. Enter the following into the new line:

```
Name : DeviceID
Type : Dynamic
Default : Leave Blank
Description : The ID of the device to run this script against
Required : Yes
Hidden : No
```

The screenshot shows the Arista Provisioning web interface. On the left is a navigation sidebar with icons for Provisioning, Network Provisioning, Configlets, Image Repository, Upload and Download, Tasks, Actions (selected), Change Control, Action Bundles, Templates, Studios, Workspaces, Snapshot Configuration, Public Cloud Accounts, Tags, and Zero Touch Provisioning. The main content area is titled 'Actions' and shows 'ASE - Clear Counters' with a 'Manage Arguments' link. Below this, a table titled 'Manage Arguments' contains one row for 'DeviceID' with a 'Dynamic' type, 'Required' status, and 'Hidden' status. A 'Save Changes' button is in the top right of the table area.

Name	Type	Default	Description	Required	Hidden
DeviceID	Dynamic		The ID of the device to run this script against	Yes	No

9. Repeat steps 7 & 8 to create the interface argument using the following values:

```
Name : Interface
Type : Dynamic
Default : Leave Blank
Description : The interface number of the interface to clear counters on
Required : Yes
Hidden : No
```

The screenshot shows the Arista Provisioning web interface. On the left is a navigation sidebar with options like Provisioning, Network Provisioning, Configlets, Image Repository, Upload and Download..., Tasks, Actions (selected), Change Control, Action Bundles, Templates, Studios, Workspaces, Snapshot Configuration, Public Cloud Accounts, Tags, and Zero Touch Provisioning. The main content area is titled 'ASE - Clear Counters' and includes tabs for 'Edit Script' and 'Manage Arguments' (which is active). Below the tabs is a table for managing arguments. The table has columns for Name, Type, Default, Description, Required, and Hidden. Two arguments are listed: 'DeviceID' (Dynamic) and 'Interface' (Dynamic). A 'Save Changes' button is located at the top right of the table.

Name	Type	Default	Description	Required	Hidden
DeviceID	Dynamic		The ID of the device to run this script against	Yes	No
Interface	Dynamic		The interface number of the interface to clear counters on	Yes	No

Below the table, there are links for 'Static Argument' and 'Dynamic Argument'.

10. Click the blue **“Save Changes”** button on the right side of the screen.

11. Now that the Arguments are created, we can head to the Script section to begin writing the action. Click on the **“Edit Script”** button below the Action name.

The screenshot shows the same Arista Provisioning interface, but now the 'Edit Script' tab is active. The main content area is titled 'Script' and contains a large text editor for writing the script. To the right of the script editor are two sections: 'Static Arguments' (showing 'No data') and 'Dynamic Arguments' (showing 'DeviceID (Required)' and 'Interface (Required)'). There are 'Execute' and 'Save Changes' buttons at the top right of the script editor.

12. On line 1 of the script we'll want to identify who wrote the script. Go ahead and enter the following on line 1, replacing "your-name-here" with your name:

```
# Written by your-name-here
```

13. The next thing we need to do is create a variable in the script that references the "Interface" argument that was created. To do that, we skip a line and then add the following on lines 3 & 4.

```
# Create a variable to reflect the "Interface" dynamic argument
interface = ctx.action.args.get("Interface")
```

14. Now, we need to add two different lists of commands that will need to be run on the switch. The first list will show the interface's existing counters, and the second will clear the interface counters. Skip a line and then paste the following into lines 6 - 16:

```
# List of commands to show interface counters
showCountersCmds = [
    'enable',
    f'show interfaces ethernet{interface} counters'
]

# List of commands to clear interface counters
clearCountersCmds = [
    'enable',
    f'clear counters ethernet{interface}'
]
```

15. Validate that the script section matches the following screenshot.

ASE-Clear Counters [🔗](#)

Category Language Description
 Change Control Python 3 Enter description [🔗](#)

[Edit Script](#) [Manage Arguments](#)

Script

Execute

Save Changes

```

1  # Written by your-name-here
2
3  # Create a variable to reflect the "Interface" dynamic argument
4  interface = ctx.action.args.get("Interface")
5
6  # List of commands to show interface counters
7  showCountersCmds = [
8      'enable',
9      f'show interfaces ethernet{interface} counters'
10 ]
11
12 # List of commands to clear interface counters
13 clearCountersCmds = [
14     'enable',
15     f'clear counters ethernet{interface}'
16 ]

```

Static Arguments



No data

Dynamic Arguments

DeviceID (Required) ⓘ

Interface (Required) ⓘ

16. Let's continue editing the script by adding some logging stating that we're getting the values of the current interface counters. Skip another line and then add the following to lines 18 & 19:

```

# Create log entry
ctx.alog(f'Getting counters for Ethernet{interface}')

```

17. Now, we can proceed with running the commands listed in the "showCountersCmds" list on the switch. To do that, add another blank line and the following to lines 21 and 22.

```

# Run the list of commands to get counters
cmdResponse = ctx.runDeviceCmds(showCountersCmds)

```

18. The commands have now been run on the switch and the response has been stored in the variable "**cmdResponse**." Let's create a log entry that outputs the response. On lines 24 - 25 add the following:

```

# Create a log entry of the counters
ctx.alog(f'The current interface counters are : {cmdResponse[1]["response"]}')

```


NOTE: The cmdResponse value returned above is similar to the following:

```
[{'error': '', 'response': {}},
{'error': '', 'response': {
  'interfaces': {
    'Ethernet1': {
      'inBroadcastPkts': 1,
      'inDiscards': 0,
      'inMulticastPkts': 38,
      'inOctets': 7529,
      'inUcastPkts': 0,
      'lastUpdateTimestamp': 1717697811.8986592,
      'outBroadcastPkts': 836, 'outDiscards': 0,
      'outMulticastPkts': 37,
      'outOctets': 45227, 'outUcastPkts': 0}
    }
  }
}]
```

The cmdResponse list contains a dictionary for each command run on the device. Since we're interested in the “show interfaces ethernet counters” command response value, we need to specify “[1]” to indicate the second item in the list (since list values start at the 0th item in Python) and “[“response”]” for the response data.

19. Now that we have logged what the counters were, let's proceed with clearing the counters. Let's create a blank line, then create another log entry stating that we will clear the counters. On lines 27 - 28 enter the following:

```
# Create log entry before clearing the counters
ctx.alog(f'Clearing counters for Ethernet{interface}')
```

20. Let's actually clear the counters now. Let's skip another line and then enter the following on lines 30 - 31:

```
# Run the list of commands to clear counters
cmdResponse = ctx.runDeviceCmds(clearCountersCmds)
```

21. Now that the counters are cleared let's create another log entry stating that the counters have been successfully cleared. To do that, let's skip another line and add the following content to lines 33 - 34:

```
# Create log entry
ctx.alog(f'Counters for Ethernet{interface} Successfully Cleared')
```

22. For the final part of the script, let's repeat steps 16 - 18 to get the interface's current counter values and log them.
23. Validate that the script matches the screenshot below, then click the blue **“Save Changes”** button to save the action.

ASE-Clear Counters

Category

Language

Description

Change Control

Python 3

Enter description [Edit Script](#)[Manage Arguments](#)

Script

[Execute](#)[Save Changes](#)

```

1  # Written by your-name-here
2
3  # Create a variable to reflect the "Interface" dynamic argument
4  interface = ctx.action.args.get("Interface")
5
6  # List of commands to show interface counters
7  showCountersCmds = [
8      'enable',
9      f'show interfaces ethernet{interface} counters'
10 ]
11
12 # List of commands to clear interface counters
13 clearCountersCmds = [
14     'enable',
15     f'clear counters ethernet{interface}'
16 ]
17
18 # Create log entry
19 ctx.alog(f'Getting counters for Ethernet{interface}')
20
21 # Run the list of commands to get counters
22 cmdResponse = ctx.runDeviceCmds(showCountersCmds)
23
24 # Create a log entry of the counters
25 ctx.alog(f'The current interface counters are : {cmdResponse[1]["response"]}')
26
27 # Create log entry before clearing the counters
28 ctx.alog(f'Clearing counters for Ethernet{interface}')
29
30 # Run the list of commands to clear counters
31 cmdResponse = ctx.runDeviceCmds(clearCountersCmds)
32
33 # Create log entry
34 ctx.alog(f'Counters for Ethernet{interface} Successfully Cleared')
35
36 # Create log entry
37 ctx.alog(f'Getting counters for Ethernet{interface}')
38
39 # Run the list of commands to get counters
40 cmdResponse = ctx.runDeviceCmds(showCountersCmds)
41
42 # Create a log entry of the counters
43 ctx.alog(f'The current interface counters are : {cmdResponse[1]["response"]}')

```

24. Now, we can proceed with testing our Action. To do so, click on the light blue **“Execute”** button.

25. An **“Execute Action”** dialog box will appear on the right side of the screen. Click **“Dynamic Argument Values”** to expand the arguments section.

The screenshot displays the Arista Provisioning interface. On the left is a navigation sidebar with categories like Provisioning, Tasks, and Actions. The main panel shows the 'ASE - Clear Counters' action, which is a Python 3 script. The script is designed to clear interface counters for a specified interface. It includes comments and code for logging, command execution, and response handling. To the right of the script editor is the 'Execute Action' dialog, which contains a 'Dynamic Argument Values' section. This section has two input fields: 'Interface' (with a dropdown arrow) and 'DeviceID' (with a dropdown arrow). An 'Execute' button is located at the bottom right of the dialog.

Script Content:

```

1 # Written by your-name-here
2
3 # Create a variable to reflect the "interface" dynamic argument
4 interface = ctx.action.args.get("interface")
5
6 # List of commands to show interface counters
7 showCountersCmds = [
8     'enable',
9     f'show interfaces ethernet(interface) counters'
10 ]
11
12 # List of commands to clear interface counters
13 clearCountersCmds = [
14     'enable',
15     f'clear counters ethernet(interface)'
16 ]
17
18 # Create log entry
19 ctx.log(f'Getting counters for Ethernet(interface)')
20
21 # Run the list of commands to get counters
22 cmdResponse = ctx.runDeviceCmds(showCountersCmds)
23
24 # Create a log entry of the counters
25 ctx.log(f'The current interface counters are : {cmdResponse[0]["response"]}')
26
27 # Create log entry prior to clearing the counters
28 ctx.log(f'Clearing counters for Ethernet(interface)')
29
30 # Run the list of commands to clear counters
31 cmdResponse = ctx.runDeviceCmds(clearCountersCmds)
32
33 # Create log entry
34 ctx.log(f'Counters for Ethernet(interface) Successfully Cleared')
35
36 # Create log entry
37 ctx.log(f'Getting counters for Ethernet(interface)')
38
39 # Run the list of commands to get counters
40 cmdResponse = ctx.runDeviceCmds(showCountersCmds)
41
42 # Create a log entry of the counters
43 ctx.log(f'The current interface counters are : {cmdResponse[0]["response"]}')

```

Execute Action Dialog:

- Dynamic Argument Values
- Interface
- DeviceID
- Execute

26. Let's use Ethernet1 for this test, so specify "1" for the "Interface."

27. In the DeviceID dropdown, select "Leaf-1A"

28. Next, click the blue "Execute" button.

The screenshot displays the Arista Provisioning web interface. On the left is a navigation sidebar with options like Provisioning, Network Provisioning, Configlets, Image Repository, Upload and Download..., Tasks, Actions, Change Control, Action Bundles, Templates, Studios, Workspaces, Snapshot Configuration, Public Cloud Accounts, Tags, and Zero Touch Provisioning. The main panel is titled 'Actions' and shows a specific action named 'ASE - Clear Counters'. Below the title, there are tabs for 'Edit Script' and 'Manage Arguments'. The 'Edit Script' tab is active, showing a Python script for clearing interface counters. The script includes comments and code for listing and clearing counters on a specified interface. To the right of the script editor is an 'Execute' button. Further right, an 'Execute Action' dialog box is open, showing 'Dynamic Argument Values'. It has two input fields: 'Interface' with the value '1' and 'DeviceID' with a dropdown menu showing 'Leaf-1A'. An 'Execute' button is at the bottom of this dialog.

ASE - Clear Counters

Category: Change Control | Language: Python 3 | Description: Enter description

[Edit Script](#) | [Manage Arguments](#)

Script

```

1 # Written by your-name-here
2
3 # Create a variable to reflect the "Interface" dynamic argument
4 interface = ctx.action.args.get("Interface")
5
6 # List of commands to show interface counters
7 showCountersCmds = [
8     'enable',
9     f'show interfaces ethernet {interface} counters'
10 ]
11
12 # List of commands to clear interface counters
13 clearCountersCmds = [
14     'enable',
15     f'clear counters ethernet {interface}'
16 ]
17
18 # Create log entry
19 ctx.log(f'Getting counters for Ethernet({interface})')
20
21 # Run the list of commands to get counters
22 cmdResponse = ctx.runDeviceCmds(showCountersCmds)
23
24 # Create a log entry of the counters
25 ctx.log(f'The current interface counters are : {cmdResponse[0]["response"]}')
26
27 # Create log entry prior to clearing the counters
28 ctx.log(f'Clearing counters for Ethernet({interface})')
29
30 # Run the list of commands to clear counters
31 cmdResponse = ctx.runDeviceCmds(clearCountersCmds)
32
33 # Create log entry
34 ctx.log(f'Counters for Ethernet({interface}) Successfully Cleared')
35
36 # Create log entry
37 ctx.log(f'Getting counters for Ethernet({interface})')
38
39 # Run the list of commands to get counters
40 cmdResponse = ctx.runDeviceCmds(showCountersCmds)
41
42 # Create a log entry of the counters
43 ctx.log(f'The current interface counters are : {cmdResponse[0]["response"]}')

```

Execute Action

Dynamic Argument Values

Interface ①
1

DeviceID ①
Leaf-1A

Execute

29. As the action executes, it will provide the logging output in the pane below the “**Execute**” button. The oldest log entry will be at the bottom, so you should read the logs from the bottom up. You’ll be able to see that the current interface counters were displayed, they were then cleared, and the new counters were displayed.

The screenshot displays the Arista CloudVision interface. On the left is a navigation sidebar with options like Provisioning, Network Provisioning, Configlets, Image Repository, Upload and Download..., Tasks, Actions, Change Control, Action Bundles, Templates, Studios, Workspaces, Snapshot Configuration, Public Cloud Accounts, Tags, and Zero Touch Provisioning. The main panel is titled 'ASE - Clear Counters' and shows a Python script for clearing interface counters. The script includes comments and code for listing, clearing, and logging interface counters for a specific interface.

On the right, the 'Execute Action' panel is open, showing 'Dynamic Argument Values' with 'Interface' set to '1' and 'DeviceID' set to 'Leaf-1A'. Below this is a search bar for logs and a list of execution logs. The logs show the action being executed successfully on Jun 6, 2024, at 13:13:13.125 GMT-6, with a response indicating that the counters for Ethernet1 were successfully cleared.

30. Now, your custom action is available to be used within a change control.

NOTE: Check out this document for more information on custom actions.

<https://www.arista.com/en/support/toi/cvp-2021-3-0/14901-ui-for-custom-action-scripts>

Arista also has a repo with some custom actions you might find useful. That repo can be accessed here: <https://github.com/aristanetworks/cloudvision-python-actions>

<https://www.arista.com/en/support/toi/cvp-2021-3-0/14901-ui-for-custom-action-scripts>

<https://github.com/aristanetworks/cloudvision-python-actions>