puppet_eos Module Documentation

Release 1.4.0

Arista Networks - EOS+ Consulting Services

Oct 06, 2017

Contents

1 Why Puppet and Arista **3**
- 1.1 Your business is changing. Can your network keep up? . . . 3
- 1.2 How do I take advantage of these capabilities? . . . 5

2 Overview **7**
- 2.1 Introduction . . . 7
- 2.2 Terminology . . . 7
- 2.3 Prerequisites . . . 8

3 Quick Start **9**
- 3.1 Bootstrapping a switch . . . 9
- 3.2 Configuring the Puppet Master . . . 11
- 3.3 Verifying the agent on EOS . . . 12

4 Installation **15**
- 4.1 Bootstrapping a switch . . . 15
- 4.2 Configuring the Puppet Master . . . 17
- 4.3 Verifying the agent on EOS . . . 18

5 Types **21**
- 5.1 Getting to know the Types . . . 22
- 5.2 Resource Types . . . 22

6 Cookbook **47**
- 6.1 Creating a Node Profile Manifest . . . 47
- 6.2 Recipe 1: Masterless / Headless . . . 47
- 6.3 Recipe 2: MLAG . . . 48

7 Troubleshooting **51**
- 7.1 Introduction . . . 51
- 7.2 Submitting Issues . . . 51

8 Developing **53**
- 8.1 Overview . . . 53
- 8.2 Running from source . . . 53
- 8.3 Contributing . . . 54

9 Testing Modules **55**

9.1 Introduction 55

10 FAQ **57**
10.1 Server: Error: ... cannot load such file – rbeapi/client 57
10.2 Server: Error: ... provider 'eos': undefined method *api' for nil:NilClass* 57

11 Release Notes **59**
11.1 Release 1.4.0 - January 2016 59
11.2 Release 1.3 - November 2015 60
11.3 Release 1.2 - August 2015 61
11.4 Release 1.1 - July 2015 61
11.5 Release 1.0 - May 2015 62

12 License **65**

Contents:

CHAPTER 1

Why Puppet and Arista

- *Your business is changing. Can your network keep up?*
- *How do I take advantage of these capabilities?*

Your business is changing. Can your network keep up?

Businesses need to adapt quickly. New ideas don't generate revenue or provide value until they reach customers. In addition to application development, that often requires scaling or modifying infrastructure from storage to physical servers to the network. Puppet and Arista enable your network team to be more responsive to their internal customers. This enables new servers to come online faster, IP storage to be connected to those servers, and the various components of your application to communicate and reach the customer.

- **Automatically test changes with Puppet and vEOS.** Ensure the changes to your network are tested. vEOS is Arista EOS that will run on your favorite hypervisor enabling you to build a virtual test lab for your network. This allows your Continuous Integration (CI) tools to automatically test network configrations, defined in Puppet, on EOS before deploying those into production.

  Download vEOS-lab for free by registering at Arista.com. For instructions on using vEOS in your hypervisor, search EOS Central.

- **Continually audit for compliance** Puppet can monitor your network for compliance with the central configuration. With all changes going through a revision control system, such as git, changes are clearly tracked including who requested what change. Then Puppet logs exactly what was changed on every node. If anyone makes changes directly on a device, Puppet will report that in an audit run or automatically correct the device when in enforcing mode. This quickly assures you, your team, and auditors that you have the controls in place to know the state of every device on your network, properly manage change and ensure no unauthorized configurations exist.

- **Data driven network definitions simplify configuration management** Puppet, Hiera, and configuration templates make it easy to define your network settings in a YAML data file. This means adding things like a new VLAN or changing the syslog server across your entire environent is simply a one-line change to a

data file instead of touching Puppet code or, owrse, manually configuring each device. This reduces the risk of a configuration change and saves time, getting your team on to more important tasks quickly.

Hiera data:

```
---
ntp::source_interface: 'Management1'
ntp::servers:
  - 192.0.2.251

eos_config::snmp::contact: 'NetOps'

eos_config::name_servers::name_servers:
  - 192.0.2.250
  - 192.0.2.252

vlans:
  1: { vlan_name: default }
  2: { vlan_name: TestVlan_2 }
  9: { vlan_name: Demo_vlan }
  100: { vlan_name: TestVlan_100, enable: false }
  101: { vlan_name: TEST_VLAN_101 }
```

- **Delegate routine server-port provisioning to the server team** Puppet can be the self-service catalog for other teams within your business, increasing your ability to rapidly deliver business value. Does the server team use Puppet? Do they have to open a network request every time they rack a new server? With a simple Puppet class provided by the network team, server admins can safely be allowed to self-provision standard host ports with internal checks to limit access to core network services such as uplink ports, BGP configurations, etc.

  Example:

```
esx_server_port {'row3-rack2-server38':
  description => 'ESXi host 38, vmnic0',
  switchport  => '24',
}
```

- **Work the way you work best** We proide multiple ways to manage your network configuration with Puppet: Templates or discrete resources. If you are new to Puppet or prefer to think of your network configs like files, *eos_switchconfig* may be right for you. This allows you to use a configuration template (or combine reusable configuration snippets) as the basis for your running-config. If want to take advantage of the full power of Puppet, you may prefer to represent your configuration with the *discrete resource* model. This allos you to gradually, selectively move your network configuration under Puppet control and easily group and abstract complex portions of the config in to easy to use resources.

  Switchconfig with templates:

```
eos_switchconfig {'running-config':
  content => template('network_configs/spine'),
}
```

  Discrete resources:

```
eos_ntp_server { '174.127.117.113':
  ensure => present,
}

eos_acl_entry{ 'test1:10':
  ensure       => present,
```

```
  acltype      => standard,
  action       => permit,
  srcaddr      => '192.168.1.0',
  srcprefixlen => 8,
  log          => true,
}

eos_bgp_neighbor { '192.0.2.1':
  ensure     => present,
  enable     => true,
  peer_group => 'Edge',
  remote_as  => 65004
}
```

How do I take advantage of these capabilities?

Contact Arista EOS+ Consulting Services eosplus-dev@arista.com for more information, a demo, or to let us jumpstart your Puppet environent.

CHAPTER 2

Overview

- *Introduction*
- *Terminology*
- *Prerequisites*

Introduction

Puppet is a configuration management platform which operates by way of the user defining the desired state for a resource, puppet comparing that to the current state, then resolving any differences. By having an agent running on each node, puppet can not only be operated from a master, but can also be used in a standalone (masterless, headless) configuration.

This Type / Provider module enables Types specific for managing Arista EOS device configuration from Puppet. By defining profile classes around these types, network device management can be refocused to managing network applications such as ntp, stp, ospf, vxlan, or even abstracted away from a network-centric perspective in to higher level business goals such as deploying a new application service or site.

Puppet masters can be deployed in Enterprise or Open Source varieties providing various levels of tools and support, including dashboards and reporting. Such additional toolsets provide simplified configuration and rich analysis and auditing of an environment.

Terminology

When working with Puppet there is some basic terminology which is helpful to understand. A Type is resource that Puppet knows how to manage; a hostname, VLAN, layer-2 interface, etc. A Provider is the implementation-specific code that evaluates and effects change to the respective Type. There can be multiple Providers for a Type; for example: VLAN configuration may have a different provider for each OS vendor that it supports. A Module can consist of one

or more Types and/or Providers packaged together or, it could be a grouping of related manifest classes, files, and templates.

Prerequisites

Puppet provides an EOS extension (SWIX file) for Arista switches that contains Ruby, the Puppet Enterprise agent and a number of dependencies for use with either Puppet Enterprise or Open Source Puppet masters.

On EOS, eAPI must be initially enabled and the rbeapi rubygem extension installed. These 2 components are used by the puppet modules to review the current state of resources and to bring them into compliance with the desired state.

On-switch Requirements:

- Puppet agent
  - Ruby, etc.
- rbeapi rubygem
- eAPI enabled

CHAPTER 3

Quick Start

- *Bootstrapping a switch*
  - *EOS Command Aliases*
- *Configuring the Puppet Master*
- *Verifying the agent on EOS*

Bootstrapping a switch

There are a number of ways to bootstrap the necessary components on to a switch, and automatically load the minimal, initial configuration. We strongly suggest ZTP Server[4] to automate the steps from initial power-on to contacting the Puppet master.

Sample minimal configuration on a switch includes basic IP connectivity, hostname and domain-name which are used to generate the switch's SSL certificate, a name-server or host entry for "puppet", the default master name unless otherwise specified, and enabling eAPI (management api http-commands):

```
!
hostname my-switch
ip domain-name example.com
!
ip name-server vrf default 8.8.8.8
! OR
ip host puppet 192.2.2.5
!
interface Management1
   ip address 192.2.2.101/24
   no shutdown
```

[4] https://github.com/arista-eosplus/ztpserver

```
!
ip route 0.0.0.0/0 192.2.2.1
!
```

From EOS 4.14.5 and up, it is recommended configure EOS to use unix-sockets for eAPI:

```
management api http-commands
   no protocol https
   protocol unix-socket
   no shutdown
!
```

In EOS versions below 4.14.5, it is recommended to configure EOS to use https for eAPI. This also requires the creation of a `flash:eapi.conf` in which to store user credentials to login to eAPI:

```
username eapi privilege 15 secret icanttellyou
!
management api http-commands
   no shutdown
!
```

If you configured eAPI (`management api http-commands`) for anything other than `unix-socket`, then an `flash:eapi.conf` is also required. Ensure that the connection is `localhost` and enter the transport, port, username, and password required for the puppet module to connect to eAPI. See more about configuring eapi.conf[1].

Example `flash:eapi.conf`:

```
[connection:localhost]
transport: https
port: 1234
username: eapi
password: password
enablepwd: itsasecret
```

Install the puppet agent from Puppet[5] (previous releases[6]):

Puppet 3.x:

```
Arista#copy http://myserver/puppet-enterprise-3.8.2-eos-4-i386.swix extensions:
Arista#extension puppet-enterprise-3.8.2-eos-4-i386.swix
```

Puppet All-In-One agent (2015.x):

```
Arista#copy http://myserver/puppet-agent-1.3.5-1.eos4.i386.swix extensions:
Arista#extension puppet-agent-1.3.5-1.eos4.i386.swix
```

Additionally, Puppet 2015.x and up should be configured to run as root in the puppet.conf file:

```
Arista#bash sudo /opt/puppetlabs/bin/puppet config set user root
```

Install the rbeapi extension[7]:

Note: The rbeapi rubygem and its requirements MAY be installed using Puppet instead of by SWIX on the CLI. Care

[1] https://github.com/arista-eosplus/rbeapi#example-eapiconf-file
[5] https://puppet.com/download-puppet-enterprise-all#eos
[6] https://puppet.com/misc/pe-files/previous-releases
[7] https://github.com/arista-eosplus/rbeapi/releases

should be taken to ensure that the rubygems are installed in a manner that will be restored upon switch reload. This is automatic with the SWIX package but, otherwise, will be re-initiated by the next Puppet agent run following a reload.

Puppet 3.x:

```
Arista#copy http://myserver/rbeapi-puppet3-0.5.1.swix extensions:
Arista#extension rbeapi-puppet3-0.5.1.swix
```

Puppet All-In-One agent (2015.x):

```
Arista#copy http://myserver/rbeapi-puppet-aio-0.5.1.swix extensions:
Arista#extension rbeapi-puppet-aio-0.5.1.swix
```

Save the installed extensions:

```
Arista#copy installed-extensions boot-extensions
```

EOS Command Aliases

If working with puppet manually from the CLI, it may be convenient to add CLI aliases to your systems. Some examples follow.

```
alias pa bash sudo puppet agent --environment demo --waitforcert 30 --onetime true
alias puppet bash sudo /opt/puppetlabs/bin/puppet
alias puppet2015 bash sudo /opt/puppetlabs/bin/puppet
alias puppet3 bash sudo puppet
alias puppet-vrf bash sudo ip netns exec <MGMT-VRF> /opt/puppetlabs/bin/puppet
```

With the above aliases, repetitive typing can be reduced to, for example:

```
Arista#pa --test
Arista#puppet resource eos_vlan
Arista#puppet describe eos_vlan
```

Configuring the Puppet Master

Follow the standard instructions for installing either a Puppet Enterprise or Puppet Open-source master server and setup your environment(s). (Standalone Puppet, also known as headless or masterless puppet, is covered in a separate section.) As the paths to various items and specifics may vary from system to system, you may need to make minor adjustments to the ommands, below, to conform to your particular system. Use `puppet config print` to locate the correct paths.

On the master, install the Forge: eos[2] module (Source: GitHub: puppet-eos[3]). This module is self-contained including the types and providers specific to EOS.

Note: There is also a netdev_stdlib module in which Puppet maintains a cross-platform set of Types in netdev_stdlib and the EOS-specific providers are in netdev_stdlib_eos.

It is NOT necessary to install the rbeapi rubygem on the server, beginning with module version 1.3.0.

[2] https://forge.puppet.com/aristanetworks/eos
[3] https://github.com/arista-eosplus/puppet-eos

Add the aristanetworks-eos module to your server's modulepath:

Puppet installer:

```
$ sudo puppet module install aristanetworks-eos [--environment production ] [--
↪modulepath $basemodulepath ]
```

Install from source:

```
$ sudo git clone https://github.com/arista-eosplus/puppet-eos.git <environment>/
↪modules/eos
$ cd <environment>/modules/eos/
$ sudo git checkout <version or branch>
```

Link using Git submodules:

```
$ cd $moduledir
$ git submodule add https://github.com/arista-eosplus/puppet-eos.git eos
$ git submodule status
$ git submodule init
$ git status
```

Verifying the agent on EOS

Run the puppet agent on EOS. This performs several key tasks:

- Generate a keypair and request a certificate from the master
- Retrieve the CA and Master certificates
- Run pluginsync (enabled by default) to download the types and providers
- Run the defined manifests, if configured

Note: Prior to the first full agent run, there may not be a link in the default PATH requiring you to fully qualify the path to puppet. Starting with Puppet 2015.x, the puppet binary is installed in /opt/puppetlabs/bin/. After the first puppet agent run, a link will be created in /usr/bin/ which is in the default PATH.

```
Arista#bash sudo /opt/puppetlabs/bin/puppet agent [--environment <env_name>] --test --
↪onetime --no-daemonize --waitforcert 30
```

On the Master, sign the node's certificate request:

```
$sudo puppet cert list
$sudo puppet cert sign <certname>
```

If you did not include `waitforcert`, above, then re-run the puppet agent command to install the signed certificate from the server:

```
Arista#bash sudo puppet agent [--environment <env_name>] --test --onetime --
↪waitforcert 30
```

Verify that the `eos_*` types are available on the switch:

```
Arista#bash sudo puppet resource --types [| grep eos]
```

View the current state of a type:

```
Arista#bash sudo puppet resource eos_vlan
eos_vlan { '1':
  ensure    => 'present',
  enable    => 'true',
  vlan_name => 'default',
}
```

View the description for a type:

```
Arista#bash sudo puppet describe eos_vlan
```

If the steps, above, were not successful, proceed to the *Troubleshooting* chapter.

CHAPTER 4

Installation

- *Bootstrapping a switch*
  - *EOS Command Aliases*
- *Configuring the Puppet Master*
- *Verifying the agent on EOS*

Bootstrapping a switch

There are a number of ways to bootstrap the necessary components on to a switch, and automatically load the minimal, initial configuration. We strongly suggest ZTP Server[4] to automate the steps from initial power-on to contacting the Puppet master.

Sample minimal configuration on a switch includes basic IP connectivity, hostname and domain-name which are used to generate the switch's SSL certificate, a name-server or host entry for "puppet", the default master name unless otherwise specified, and enabling eAPI (management api http-commands):

```
!
hostname my-switch
ip domain-name example.com
!
ip name-server vrf default 8.8.8.8
! OR
ip host puppet 192.2.2.5
!
interface Management1
   ip address 192.2.2.101/24
   no shutdown
```

[4] https://github.com/arista-eosplus/ztpserver

```
!
ip route 0.0.0.0/0 192.2.2.1
!
```

From EOS 4.14.5 and up, it is recommended configure EOS to use unix-sockets for eAPI:

```
management api http-commands
   no protocol https
   protocol unix-socket
   no shutdown
!
```

In EOS versions below 4.14.5, it is recommended to configure EOS to use https for eAPI. This also requires the creation of a `flash:eapi.conf` in which to store user credentials to login to eAPI:

```
username eapi privilege 15 secret icanttellyou
!
management api http-commands
   no shutdown
!
```

If you configured eAPI (`management api http-commands`) for anything other than `unix-socket`, then an `flash:eapi.conf` is also required. Ensure that the connection is `localhost` and enter the transport, port, username, and password required for the puppet module to connect to eAPI. See more about configuring eapi.conf[1].

Example `flash:eapi.conf`:

```
[connection:localhost]
transport: https
port: 1234
username: eapi
password: password
enablepwd: itsasecret
```

Install the puppet agent from Puppet[5] (previous releases[6]):

Puppet 3.x:

```
Arista#copy http://myserver/puppet-enterprise-3.8.2-eos-4-i386.swix extensions:
Arista#extension puppet-enterprise-3.8.2-eos-4-i386.swix
```

Puppet All-In-One agent (2015.x):

```
Arista#copy http://myserver/puppet-agent-1.3.5-1.eos4.i386.swix extensions:
Arista#extension puppet-agent-1.3.5-1.eos4.i386.swix
```

Additionally, Puppet 2015.x and up should be configured to run as root in the puppet.conf file:

```
Arista#bash sudo /opt/puppetlabs/bin/puppet config set user root
```

Alternatively, a puppet user maybe created within linux, if preferred, and the agent will automatically run as user 'puppet':

[1] https://github.com/arista-eosplus/rbeapi#example-eapiconf-file
[5] https://puppet.com/download-puppet-enterprise-all#eos
[6] https://puppet.com/misc/pe-files/previous-releases

```
#!/bin/bash
# flash:rc.eos
# This script will be executed on-startup of EOS
sudo useradd puppet
```

Install the rbeapi extension[7]:

Note: The rbeapi rubygem and its requirements MAY be installed using Puppet instead of by SWIX on the CLI. Care should be taken to ensure that the rubygems are installed in a manner that will be restored upon switch reload. This is automatic with the SWIX package but, otherwise, will be re-initiated by the next Puppet agent run following a reload.

Puppet 3.x:

```
Arista#copy http://myserver/rbeapi-puppet3-0.5.1.swix extensions:
Arista#extension rbeapi-puppet3-0.5.1.swix
```

Puppet All-In-One agent (2015.x):

```
Arista#copy http://myserver/rbeapi-puppet-aio-0.5.1.swix extensions:
Arista#extension rbeapi-puppet-aio-0.5.1.swix
```

Save the installed extensions:

```
Arista#copy installed-extensions boot-extensions
```

EOS Command Aliases

If working with puppet manually from the CLI, it may be convenient to add CLI aliases to your systems. Some examples follow.

```
alias pa bash sudo puppet agent --environment demo --waitforcert 30 --onetime true
alias puppet bash sudo /opt/puppetlabs/bin/puppet
alias puppet2015 bash sudo /opt/puppetlabs/bin/puppet
alias puppet3 bash sudo puppet
alias puppet-vrf bash sudo ip netns exec <MGMT-VRF> /opt/puppetlabs/bin/puppet
```

With the above aliases, repetitive typing can be reduced to, for example:

```
Arista#pa --test
Arista#puppet resource eos_vlan
Arista#puppet describe eos_vlan
```

Configuring the Puppet Master

Follow the standard instructions for installing either a Puppet Enterprise or Puppet Open-source master server and setup your environment(s). (Standalone Puppet, also known as headless or masterless puppet, is covered in a separate section.) As the paths to various items and specifics may vary from system to system, you may need to make minor adjustments to the ommands, below, to conform to your particular system. Use `puppet config print` to locate the correct paths.

[7] https://github.com/arista-eosplus/rbeapi/releases

On the master, install the Forge: eos[2] module (Source: GitHub: puppet-eos[3]). This module is self-contained including the types and providers specific to EOS.

Note: There is also a netdev_stdlib module in which Puppet maintains a cross-platform set of Types in netdev_stdlib and the EOS-specific providers are in netdev_stdlib_eos.

It is NOT necessary to install the rbeapi rubygem on the server, beginning with module version 1.3.0.

Add the aristanetworks-eos module to your server's modulepath:

Puppet installer:

```
$ sudo puppet module install aristanetworks-eos [--environment production ] [--
↪modulepath $basemodulepath ]
```

Install from source:

```
$ sudo git clone https://github.com/arista-eosplus/puppet-eos.git <environment>/
↪modules/eos
$ cd <environment>/modules/eos/
$ sudo git checkout <version or branch>
```

Link using Git submodules:

```
$ cd $moduledir
$ git submodule add https://github.com/arista-eosplus/puppet-eos.git eos
$ git submodule status
$ git submodule init
$ git status
```

Verifying the agent on EOS

Run the puppet agent on EOS. This performs several key tasks:

- Generate a keypair and request a certificate from the master
- Retrieve the CA and Master certificates
- Run pluginsync (enabled by default) to download the types and providers
- Run the defined manifests, if configured

Note: Prior to the first full agent run, there may not be a link in the default PATH requiring you to fully qualify the path to puppet. Starting with Puppet 2015.x, the puppet binary is installed in /opt/puppetlabs/bin/. After the first puppet agent run, a link will be created in /usr/bin/ which is in the default PATH.

```
Arista#bash sudo /opt/puppetlabs/bin/puppet agent [--environment <env_name>] --test --
↪onetime --no-daemonize --waitforcert 30
```

On the Master, sign the node's certificate request:

[2] https://forge.puppet.com/aristanetworks/eos
[3] https://github.com/arista-eosplus/puppet-eos

```
$sudo puppet cert list
$sudo puppet cert sign <certname>
```

If you did not include `waitforcert`, above, then re-run the puppet agent command to install the signed certificate from the server:

```
Arista#bash sudo puppet agent [--environment <env_name>] --test --onetime --
↪waitforcert 30
```

Verify that the `eos_*` types are available on the switch:

```
Arista#bash sudo puppet resource --types [| grep eos]
```

View the current state of a type:

```
Arista#bash sudo puppet resource eos_vlan
eos_vlan { '1':
  ensure    => 'present',
  enable    => 'true',
  vlan_name => 'default',
}
```

View the description for a type:

```
Arista#bash sudo puppet describe eos_vlan
```

If the steps, above, were not successful, proceed to the *Troubleshooting* chapter.

CHAPTER 5

Types

- *Getting to know the Types*
- *Resource Types*
  - *eos_acl_entry*
  - *eos_bgp_config*
  - *eos_bgp_neighbor*
  - *eos_bgp_network*
  - *eos_command*
  - *eos_config*
  - *eos_ethernet*
  - *eos_interface*
  - *eos_ipinterface*
  - *eos_mlag*
  - *eos_mlag_interface*
  - *eos_ntp_config*
  - *eos_ntp_server*
  - *eos_portchannel*
  - *eos_routemap*
  - *eos_snmp*
  - *eos_staticroute*
  - *eos_stp_interface*

- *eos_switchport*
- *eos_system*
- *eos_user*
- *eos_varp*
- *eos_varp_interface*
- *eos_vlan*
- *eos_vrrp*
- *eos_vxlan*
- *eos_vxlan_vlan*
- *eos_vxlan_vtep*

Getting to know the Types

There are a number of ways to browse the available EOS types:

```
$ puppet resource --types | grep eos
$ puppet describe eos_vlan
```

Display the current state of a type:

```
Arista#bash sudo puppet resource eos_vlan
eos_vlan { '1':
  ensure    => 'present',
  enable    => 'true',
  vlan_name => 'default',
}
eos_vlan { '123':
  ensure    => 'present',
  enable    => 'true',
  vlan_name => 'VLAN0123',
}
eos_vlan { '300':
  ensure    => 'present',
  enable    => 'true',
  vlan_name => 'ztp_bootstrap',
}
```

This page is autogenerated; any changes will get overwritten *(last generated on 2016-01-12 21:33:30 -0500)*

Resource Types

- The *namevar* is the parameter used to uniquely identify a type instance. This is the parameter that gets assigned when a string is provided before the colon in a type declaration. In general, only developers will need to worry about which parameter is the `namevar`.

In the following code:

```
file { "/etc/passwd":
  owner => "root",
  group => "root",
  mode  => "0644"
}
```

`/etc/passwd` is considered the title of the file object (used for things like dependency handling), and because `path` is the namevar for `file`, that string is assigned to the `path` parameter.

- *Parameters* determine the specific configuration of the instance. They either directly modify the system (internally, these are called properties) or they affect how the instance behaves (e.g., adding a search path for `exec` instances or determining recursion on `file` instances).
- *Providers* provide low-level functionality for a given resource type. This is usually in the form of calling out to external commands.

When required binaries are specified for providers, fully qualifed paths indicate that the binary must exist at that specific path and unqualified binaries indicate that Puppet will search for the binary using the shell path.

- *Features* are abilities that some providers might not support. You can use the list of supported features to determine how a given provider can be used.

Resource types define features they can use, and providers can be tested to see which features they provide.

---

eos_acl_entry

Manage access-lists on Arista EOS.

Example:

```
eos_acl_entry{ 'test1:10':
  ensure       => present,
  acltype      => standard,
  action       => permit,
  srcaddr      => '1.2.3.0',
  srcprefixlen => 8,
  log          => true,
}
```

Parameters

acltype : The ACL type which is either standard and extended. Standard ACLs filter only on the source IP address. Extended ACLs allow specification of source and destination IP addresses.

Valid values are `standard`, `extended`.

action : The action for the rule can be either permit or deny. Deny is the default value. Packets filtered by a permit rule are accepted by interfaces to which the ACL is applied. Packets filtered by a deny rule are dropped by interfaces to which the ACL is applied.

Valid values are `permit`, `deny`.

ensure : The basic property that the resource should be in.

Valid values are `present`, `absent`.

log : When set to true, triggers an informational log message to the console about hte matching packet.

Valid values are `true`, `false`.

name : The name parameter is a composite namevar that combines the access-list name and the sequence number delimited by the colon (:) character

For example, if the access-list name is foo and the sequence number for this rule is 10 the namvar would be constructed as "foo:10"

The composite namevar is required to uniquely identify the specific list and rule to configure

provider : The specific backend to use for this `eos_acl_entry` resource. You will seldom need to specify this — Puppet will usually discover the appropriate provider for your platform.Available providers are:

eos Manage IP access-lists in Arista EOS. Requires rbeapi rubygem.

srcaddr : The source IP address. The following options are supported:

network_address - subnet address where srcprefixlen defines mask any - Packets from all addresses are filtered. host ip_addr - IP address (dotted decimal notation)

srcprefixlen : The source address prefix len used when srcaddr is a network address to define the subnet. Values range from 0 to 32.

---

eos_bgp_config

Manage the global BGP routing process on Arista EOS.

Example:

```
eos_bgp_config { 65001:
  ensure             => present,
  enable             => true,
  router_id          => '192.0.2.4',
  maximum_paths      => 8,
  maximum_ecmp_paths => 8,
}
```

Parameters

bgp_as : (**Namevar:** If omitted, this parameter's value defaults to the resource's title.)

The BGP autonomous system number to be configured for the local BGP routing instance. The value must be in the valid BGP AS range of 1 to 65535. The value is a String.

enable : Configures the administrative state for the global BGP routing process. If enable is True then the BGP routing process is administartively enabled and if enable is False then the BGP routing process is administratively disabled.

Valid values are `true`, `yes`, `on`, `false`, `no`, `off`.

ensure : The basic property that the resource should be in.

Valid values are `present`, `absent`.

maximum_ecmp_paths : Maximum number of installed ECMP routes. This value should be greater than or equal to maximum_paths.

maximum_paths : Maximum number of equal cost paths. This value should be less than or equal to maximum_ecmp_paths.

provider : The specific backend to use for this `eos_bgp_config` resource. You will seldom need to specify this — Puppet will usually discover the appropriate provider for your platform.Available providers are:

eos Manage global BGP routing process on Arista EOS. Requires rbeapi.

router_id : Configures the BGP routing process router-id value. The router id must be in the form of A.B.C.D

---

eos_bgp_neighbor

Manage BGP neighbor configuration on Arista EOS.

Example:

```
eos_bgp_neighbor { 'Edge':
  ensure         => present,
  enable         => true,
  description    => 'some text',
  send_community => true,
  route_map_in   => 'in_map',
  route_map_out  => 'out_map',
  next_hop_self  => false,
}

eos_bgp_neighbor { '192.0.3.1':
  ensure         => present,
  enable         => true,
  peer_group     => 'Edge',
  remote_as      => 65004,
  send_community => true,
  next_hop_self  => true,
}
```

Parameters

description : Configures the BGP neighbors description value. The value specifies an arbitrary description to add to the neighbor statement in the nodes running-configuration.

enable : Configures the administrative state for the BGP neighbor process. If enable is True then the BGP neighbor process is administartively enabled and if enable is False then the BGP neighbor process is administratively disabled.

Valid values are `true`, `yes`, `on`, `false`, `no`, `off`.

ensure : The basic property that the resource should be in.

Valid values are `present`, `absent`.

name : The name of the BGP neighbor to manage. This value can be either an IPv4 address or string (in the case of managing a peer group).

next_hop_self : Configures the BGP neighbors next-hop-self value. If enabled then the BGP next-hop-self value is enabled. If disabled, then the BGP next-hop-self community value is disabled

Valid values are `enable`, `disable`.

peer_group : The name of the peer-group value to associate with the neighbor. This argument is only valid if the neighbor is an IPv4 address.

provider : The specific backend to use for this `eos_bgp_neighbor` resource. You will seldom need to specify this — Puppet will usually discover the appropriate provider for your platform.Available providers are:

eos Manage BGP neighbor configuration on Arista EOS. Requires rbeapi.

remote_as : Configures the BGP neighbors remote-as value. Valid AS values are in the range of 1 to 65535. The value is an Integer.

route_map_in : Configures the BGP neigbhors route-map in value. The value specifies the name of the route-map.

route_map_out : Configures the BGP neigbhors route-map out value. The value specifies the name of the route-map.

send_community : Configures the BGP neighbors send-community value. If enabled then the BGP send-community value is enable. If disabled, then the BGP send-community value is disabled.

Valid values are `enable`, `disable`.

---

eos_bgp_network

Manage BGP network statements on Arista EOS.

Example:

```
eos_bgp_network { '192.0.3.0/24':
  ensure    => present,
  route_map => 'neighbor3_map',
}
```

Parameters

ensure : The basic property that the resource should be in.

Valid values are `present`, `absent`.

name : The name is a composite name that contains the IPv4_Prefix/Masklen. The IPv4 prefix to configure as part of the network statement. The value must be a valid IPv4 prefix. The IPv4 subnet mask length in bits. The value for the masklen must be in the valid range of 1 to 32.

provider : The specific backend to use for this `eos_bgp_network` resource. You will seldom need to specify this — Puppet will usually discover the appropriate provider for your platform.Available providers are:

eos Manage BGP network configuration on Arista EOS. Requires rbeapi.

route_map : Configures the BGP route-map name to apply to the network statement when configured. Note this module does not create the route-map.

---

eos_command

```
eos_command { 'Save running-config':
  mode     => 'enable',
  commands => 'copy running-config startup-config',
}
```

Parameters

commands : The specific backend to use for this `eos_command` resource. You will seldom need to specify this — Puppet will usually discover the appropriate provider for your platform.Available providers are:

eos Exec commands on Arista EOS. Requires rbeapi rubygem.

---

eos_config

Apply arbitrary configuration commands in Arista EOS. Commands will only be applied based on the absence or presence of regular expression matches. configuration for a specific command. If the command are either present or absent, the eos_config will configure the node using the command argument.

Examples:

```
eos_config { 'Set location':
  command => 'snmp-server location Here',
}

eos_config { 'Set interface description':
  section => 'interface Ethernet1',
  command => 'description My Description',
  regexp  => 'description [A-z]',
}
```

Parameters

command : Specifies the configuration command to send to the node if the regexp does not evalute to true.

name : The name parameter is the name associated with the resource.

provider : The specific backend to use for this `eos_config` resource. You will seldom need to specify this — Puppet will usually discover the appropriate provider for your platform.Available providers are:

eos Manage arbitrary config entries on EOS. Requires rbeapi rubygem.

regexp : Specifies the regular expression to use to evaluate the current nodes running configuration. This optional argument will default to use the command argument if none is provided.

section : Restricts the configuration evaluation to a single configuration section. If the configuration section argument is not provided, then the global configuration is used.

---

eos_ethernet

Manage physical Ethernet interfaces on Arista EOS. Physical Ethernet interfaces include the physical characteristics of front panel data plane ports but does not include out-of-band Management interfaces.

Example:

```
eos_ethernet { 'Ethernet3/17':
  enable              => true,
  description         => 'To switch2 Ethernet 1/3',
  flowcontrol_send    => on,
  flowcontrol_receive => on,
}
```

Parameters

description : The one line description to configure for the interface. The description can be any valid alphanumeric string including symbols and spaces.

enable : The enable value configures the administrative state of the physical Ethernet interfaces. Valid values for enable are:

- true - Administratively enables the Ethernet interface
- false - Administratively disables the Ethernet interface

Valid values are `true`, `false`.

flowcontrol_receive : This property configures the flowcontrol receive value for the specified Ethernet interface. Valid values for flowcontrol are:

- on - Configures flowcontrol receive on
- off - Configures flowcontrol receive off

Valid values are `on`, `off`.

flowcontrol_send : This property configures the flowcontrol send value for the specified Ethernet interface. Valid values for flowcontrol are:

- on - Configures flowcontrol send on
- off - Configures flowcontrol send off

Valid values are `on`, `off`.

name : The name of the physical interface to configure. The interface name must coorelate to the full physical interface identifier in EOS.

provider : The specific backend to use for this `eos_ethernet` resource. You will seldom need to specify this — Puppet will usually discover the appropriate provider for your platform.Available providers are:

eos Manage ethernet interfaces on EOS. Requires rbeapi rubygem.

---

eos_interface

Manage common attributes of all Arista EOS interfaces.

Example:

```
eos_interface { 'Management1':
  enable      => true,
  description => 'OOB management to mgmt-sw1 Ethernet42',
}
```

Parameters

description : The one line description to configure for the interface. The description can be any valid alphanumeric string including symbols and spaces.

enable : The enable value configures the administrative state of the specified interface. Valid values for enable are:

- true - Administratively enables the interface
- false - Administratively disables the interface

Valid values are `true`, `false`.

ensure : The basic property that the resource should be in.

Valid values are `present`, `absent`.

name : The name parameter specifies the full interface identifier of the Arista EOS interface to manage. This value must correspond to a valid interface identifier in EOS.

provider : The specific backend to use for this `eos_interface` resource. You will seldom need to specify this — Puppet will usually discover the appropriate provider for your platform.Available providers are:

eos Manage interfaces on EOS. Requires rbeapi rubygem.

---

eos_ipinterface

Manage logical IP (L3) interfaces in Arista EOS. Used for IPv4 physical interfaces and logical virtual interfaces.

Example:

```
eos_ipinterface { 'Ethernet3':
  address => '192.0.3.2/24',
  mtu     => 1514,

}
eos_ipinterface { 'Vlan201':
  address          => '192.0.2.1/24',
  helper_addresses => ['192.168.10.254', '192.168.11.254'],
}
```

Parameters

address : The address property configures the IPv4 address on the specified interface. The address value is configured using address/mask format.

For example

```
address => 192.168.10.16/24
```

ensure : The basic property that the resource should be in.

Valid values are `present`, `absent`.

helper_addresses : The helper_addresses property configures the list of IP helper addresses on the specified interface. IP helper addresses configure a list of forwarding address to send send broadcast traffic to as unicast, typically used to assist DHCP relay.

Helper addresses are configured using dotted decimal notation. For example

```
helper_addresses => ['192.168.10.254', '192.168.11.254']
```

mtu : The mtu property configures the IP interface MTU value which specifies the largest IP datagram that can pass over the interface without fragementation. The MTU value is specified in bytes and accepts an integer in the range of 68 to 9214.

name : The name parameter specifies the full interface identifier of the Arista EOS interface to manage. This value must correspond to a valid interface identifier in EOS.

provider : The specific backend to use for this `eos_ipinterface` resource. You will seldom need to specify this — Puppet will usually discover the appropriate provider for your platform.Available providers are:

eos Manage L3 interfaces on EOS. Requires rbeapi rubygem.

---

eos_mlag

Manage the global MLAG instance on Arista EOS. It provides configuration for global MLAG configuration parameters.

Example:

```
eos_mlag { 'settings':
  enable          => true,
  domain_id       => 'SPINE-MLAG',
  local_interface => 'Vlan4094',
  peer_address    => '192.168.0.2',
  peer_link       => 'Port-Channel900',
}
```

Parameters

domain_id : The domain_id property configures the MLAG domain-id value for the global MLAG configuration instance. The domain-id setting identifies the domain name for the MLAG domain. Valid values include alphanumeric characters

enable : The enable property configures the admininstrative state of the global MLAG configuration. Valid values for enable are:

- true - globally enables the MLAG configuration
- false - glboally disables the MLAG configuration

Valid values are `true`, `false`.

local_interface : The local_interface property configures the MLAG local-interface value for the global MLAG configuration instance. The local-interface setting specifies the VLAN SVI to send MLAG control traffic on.

Valid values must be a VLAN SVI identifier

name : The name parameter identifies the global MLAG instance for configuration and should be configured as 'settings'. All other values for name will be siliently ignored by the eos_mlag provider.

peer_address : The peer_address property configures the MLAG peer-address value for the global MLAG configuration instance. The peer-address setting specifieds the MLAG peer control endpoint IP address.

The specified value must be a valid IP address

peer_link : The peer_link property configures the MLAG peer-link value for the glboal MLAG configuration instance. The peer-link setting specifies the interface used to communicate control traffic to the MLAG peer

The provided value must be a valid Ethernet or Port-Channel interface identifer

provider : The specific backend to use for this `eos_mlag` resource. You will seldom need to specify this — Puppet will usually discover the appropriate provider for your platform.Available providers are:

eos Manage MLAG configuration on EOS. Requires rbeapi rubygem.

---

eos_mlag_interface

Manage MLAG interfaces on Arista EOS. Configure a valid MLAG with a peer switch. The mlag_id parameter is required.

Example:

```
eos_mlag_interface { 'Port-Channel10':
  mlag_id => 10,
}
```

Parameters

ensure : The basic property that the resource should be in.

Valid values are `present`, `absent`.

mlag_id : The mlag_id property assigns a MLAG ID to a Port-Channel interface used for forming a MLAG with a peer switch. Only one MLAG ID can be associated with an interface.

Valid values are in the range of 1 to 2000

Note Changing this value on an operational link will cause traffic distruption

name : The name property identifies the interface to be present or absent from the MLAG interface list. The interface must be of type portchannel.

This property expectes the full interface identifier

provider : The specific backend to use for this `eos_mlag_interface` resource. You will seldom need to specify this — Puppet will usually discover the appropriate provider for your platform.Available providers are:

eos Manage MLAG interface configuration on EOS. Requires rbeapi rubygem.

---

eos_ntp_config

Manage global NTP configuration settings on Arista EOS.

Example:

```
eos_ntp_config { 'settings':
  source_interface => 'Management1',
}
```

Parameters

name : The name parameter identifies the global NTP instance for configuration and should be configured as 'settings'. All other values for name will be siliently ignored by the provider.

provider : The specific backend to use for this `eos_ntp_config` resource. You will seldom need to specify this — Puppet will usually discover the appropriate provider for your platform.Available providers are:

eos Manage global NTP configuration on EOS. Requires rbeapi rubygem.

source_interface : The source interface property provides configuration management of the NTP source-interface value. The source interface value configures the interface address to use as the source address when sending NTP packets on the network.

The default value for source_interface is ''

---

eos_ntp_server

Manage the list of NTP servers on Arista EOS.

Example:

```
eos_ntp_server { '174.127.117.113':
  ensure => present,
}

# Remove all un-managed servers
resources { 'eos_snmp': purge => true }
```

Parameters

ensure : The basic property that the resource should be in.

Valid values are `present`, `absent`.

name : The name parameter configures the NTP server list by adding or removing NTP server entries. The value can be configured as either the host IP address or the fully qualified domain name of the desired NTP server.

provider : The specific backend to use for this `eos_ntp_server` resource. You will seldom need to specify this — Puppet will usually discover the appropriate provider for your platform.Available providers are:

eos Manage NTP server definitions on EOS. Requires rbeapi rubygem.

---

eos_portchannel

Manage logical Port-Channel interfaces on Arista EOS.

Example:

```
eos_portchannel { 'Port-Channel30':
  ensure        => present,
  description   => 'Host 39b',
  minimum_links => 2,
  lacp_mode     => active,
```

```
  lacp_fallback => individual,
  lacp_timeout  => 30,
}

eos_portchannel { 'Port-Channel31':
  ensure => absent,
}
```

Parameters

description : The one line description to configure for the interface. The description can be any valid alphanumeric string including symbols and spaces.

The default value for description is ‘’

enable : The enable value configures the administrative state of the specified interface. Valid values for enable are:

```
* true - Administratively enables the interface
* false - Administratively disables the interface
```

The default value for enable is :true

Valid values are `true`, `false`.

ensure : The basic property that the resource should be in.

Valid values are `present`, `absent`.

lacp_fallback : The lacp_fallback property configures the port-channel lacp fallback setting in EOS for the specified interface. This setting accepts the following values

```
* static  - Fallback to static LAG mode
* individual - Fallback to individual ports
* disabled - Disable LACP fallback
```

The default value for lacp_fallback is :disabled

Valid values are `static`, `individual`, `disabled`.

lacp_mode : The lacp_mode property configures the LACP operating mode of the Port-Channel interface. The LACP mode supports the following valid values

```
* active - Interface is an active LACP port that transmits and
    receives LACP negotiation packets.
* passive - Interface is a passive LACP port that only responds
    to LACP negotiation packets.
* on - Interface is a static port channel, LACP disabled.
```

The default value for lacp_mode is :on

Valid values are `active`, `passive`, `on`.

lacp_timeout : The lacp_timeout property configures the port-channel lacp timeout value in EOS for the specified interface. The fallback timeout configures the period an interface in fallback mode remains in LACP mode without receiving a PDU.

The lacp_timeout value is configured in seconds.

members : The members property manages the Array of physical interfaces that comprise the logical Port-Channel interface. Each entry in the members Array must be the full interface identifer of a physical interface name.

The default value for members is []

minimum_links : The minimum links property configures the port-channel min-links value. This setting specifies the minimum number of physical interfaces that must be operationally up for the Port-Channel interface to be considered operationally up.

Valid range of values for the minimum_links property are from 0 to 16.

The default value for minimum_links is 0

name : The name parameter specifies the name of the Port-Channel interface to configure. The value must be the full interface name identifier that corresponds to a valid interface name in EOS.

provider : The specific backend to use for this `eos_portchannel` resource. You will seldom need to specify this — Puppet will usually discover the appropriate provider for your platform.Available providers are:

eos Manage port-channel interfaces on EOS. Requires rbeapi rubygem.

---

eos_routemap

Manage route-maps on Arista EOS.

Examples:

```
eos_routemap { 'my_routemap:10':
  description => 'test 10',
  action      => permit,
  match       => 'ip address prefix-list 8to24',
}

eos_routemap { 'bgp_map:10':
  action   => permit,
  match    => 'as 10',
  set      => 'local-preference 100',
  continue => 20,
}

eos_routemap { 'bgp_map:20':
  action => permit,
  match  => [' metric-type type-1', 'as 100'],
}
```

Parameters

action : A description for the route-map.

continue : A route-map sequence number to continue on.

description : A description for the route-map.

ensure : The basic property that the resource should be in.

Valid values are `present`, `absent`.

match : Route map match rule.

name : The name of the routemap namevar composite name:seqno.

provider : The specific backend to use for this `eos_routemap` resource. You will seldom need to specify this — Puppet will usually discover the appropriate provider for your platform.Available providers are:

eos Manage routemaps on Arista EOS. Requires rbeapi rubygem

set : Set route attribute.

---

eos_snmp

Manage global SNMP configuration on Arista EOS.

Example:

```
eos_snmp { 'settings':
  contact          => 'DC02-ops@example.com',
  location         => 'DC02 POD12 Rack3'
  chassis_id       => 'JMB00000',
  source_interface => 'Loopback0',
}
```

Parameters

chassis_id : The chassis id propperty provides configuration management of the SNMP chassis-id value. This setting typically provides information to uniquely identify the SNMP agent host.

The default value for chassis_id is ‘’

contact : The contact property provides configuration management of the SNMP contact value. This setting provides informative text that typically displays the name of a person or organization associated with the SNMP agent.

The default value for contact is ‘’

location : The location property provides configuration management of the SNMP location value. This setting typcially provides information about the physical lcoation of the SNMP agent.

The default value for location is ‘’

name : The name parameter identifis the global SNMP instance for configuration and should be configured as ‘settings’. All other values for name will be silently ignored by the eos_snmp provider.

provider : The specific backend to use for this `eos_snmp` resource. You will seldom need to specify this — Puppet will usually discover the appropriate provider for your platform.Available providers are:

eos Manage global SNMP configuration on EOS. Requires rbeapi rubygem.

source_interface : The source interface property provides configuration management of the SNMP source-interface value. The source interface value configures the interface address to use as the source address when sending SNMP packets on the network.

The default value for source_interface is ‘’

---

eos_staticroute

Manage static routes in EOS.

Examples:

```
eos_staticroute { '192.168.99.0/24/10.0.0.254': }

eos_staticroute { '192.168.99.0/24/10.0.0.1':
  ensure => absent,
}

eos_staticroute { '192.168.10.0/24/Ethernet1':
  route_name => 'Edge10',
  distance   => 3,
}
```

Parameters

distance : Administrative distance of the route. Valid values are 1-255.

ensure : The basic property that the resource should be in.

Valid values are `present`, `absent`.

name : A composite string consisting of //. (namevar)

prefix - IP destination subnet prefix masklen - Number of mask bits to apply to the destination next_hop - Next_hop IP address or interface name

provider : The specific backend to use for this `eos_staticroute` resource. You will seldom need to specify this — Puppet will usually discover the appropriate provider for your platform.Available providers are:

eos Manage static routes on EOS. Requires rbeapi rubygem.

route_name : The name assigned to the static route

tag : Route tag (0-255)

---

eos_stp_interface

Manage Spanning Tree Protocol interface configuration.

Example:

```
eos_stp_interface { 'Ethernet16':
  portfast      => true,
  portfast_type => network,
  bpduguard     => false,
}
```

Parameters

bpduguard : Enable or disable the BPDU guard on a port. A BPDU guard-enabled port is disabled when it receives a BPDU packet. Disabled ports differ from blocked ports in that they are re-enabled only through manual intervention. Valid BPDU guard values:

- true - Enable the BPDU guard for the interface
- false - Disable the BPDU guard for the interface (default value)

Valid values are `true`, `false`.

name : The name parameter specifies the full interface identifier of the Arista EOS interface to manage. This value must correspond to a valid interface identifier in EOS and must be either an Ethernet or Port Channel interface.

portfast : The portfast property programs an STP port to immediately enter forwarding state when they establish a link. PortFast ports are included in spanning tree topology calculations and can enter blocking state. Valid portfast values:

- true - Enable portfast for the interface
- false - Disable portfast for the interface (default value)

Valid values are `true`, `false`.

portfast_type : Specifies the STP portfast mode type for the interface. A port with edge type connect to hosts and transition to the forwarding state when the link is established. An edge port that receives a BPDU becomes a normal port. A port with network type connect only to switches or bridges and support bridge assurance. Network ports that connect to hosts or other edge devices transition ot the blocking state. Valid portfast mode types:

- edge - Set STP port mode type to edge.
- network - Set STP port mode type to network.
- normal - Set STP port mode type to normal (default value)

Valid values are `edge`, `network`, `normal`.

provider : The specific backend to use for this `eos_stp_interface` resource. You will seldom need to specify this — Puppet will usually discover the appropriate provider for your platform.Available providers are:

eos Manage interface spanning-tree settings on Arista EOS. Requires rbeapi

---

eos_switchport

Manage logical layer 2 switchports in EOS.

When creating a logical switchport interface, if the specified physical interface was previously configured with an IP interface, the logical IP interface will be removed.

Examples:

```
eos_switchport { 'Ethernet14':
  mode        => access,
  access_vlan => 200,
}

eos_switchport { 'Ethernet15':
  mode                => trunk,
  trunk_allowed_vlans => [1, 100, 101, 102, 103, 104],
  trunk_native_vlan   => 10,
}
```

Parameters

access_vlan : The access_vlan property specifies the VLAN ID to be used for untagged traffic that enters the switchport when configured in access mode. If the switchport is configured for trunk mode, this value is configured but has no effect. The value must be an integer in the valid VLAN ID range of 1 to 4094.

The default value for the access_vlan is 1

ensure : The basic property that the resource should be in.

Valid values are `present`, `absent`.

mode : The mode property configures the operating mode of the logical switchport. Suppport modes of operation include access port or trunk port. The default value for a new switchport is access

- access - Configures the switchport mode to access
- trunk - Configures the switchport mode to trunk

Valid values are `access`, `trunk`.

name : The name parameter specifies the full interface identifier of the Arista EOS interface to manage. This value must correspond to a valid interface identifier in EOS.

Only Ethernet and Port-Channel interfaces can be configured as switchports.

provider : The specific backend to use for this `eos_switchport` resource. You will seldom need to specify this — Puppet will usually discover the appropriate provider for your platform.Available providers are:

eos Manage L2 interface settings on Arista EOS. Requires rbeapi rubygem.

trunk_allowed_vlans : The trunk_allowed_vlans property configures the list of VLAN IDs that are allowed to pass on the switchport operting in trunk mode. If the switchport is configured for access mode, this property is configured but has no effect.

The list of allowed VLANs must be configured as an Array with each entry in the valid VLAN range of 1 to 4094.

The default value for a new switchport is to allow all valid VLAN IDs (1-4094).

trunk_native_vlan : The trunk_native_vlan property specifies the VLAN ID to be used for untagged traffic that enters the switchport in trunk mode. If the switchport is configured for access mode, this value is configured but has no effect. The value must be an integer in the valid VLAN ID range of 1 to 4094.

The default value for the trunk_natve_vlan is 1

---

eos_system

Manage global EOS switch settings.

Example:

```
eos_system { 'settings':
  hostname   => 'dc02-pod2-rack3-leaf1',
  ip_routing => true,
}
```

Parameters

hostname : The global system hostname is a locally significant value that identifies the host portion of the nodes fully qualified domain name (FQDN).

The default hostname for a new system is localhost'

ip_routing : Configures the ip routing state

Valid values are `true`, `yes`, `on`, `false`, `no`, `off`.

name : The name parameter identifies the global node instance for configuration and should be configured as 'settings'. All other values for name will be siliently ignored by the eos_system provider.

provider : The specific backend to use for this `eos_system` resource. You will seldom need to specify this — Puppet will usually discover the appropriate provider for your platform.Available providers are:

eos Manage global system settings on Arista EOS. Requires rbeapi rubygem.

---

eos_user

Manage Arista EOS user accounts.

Example:

```
$pub_key = 'ssh-rsa AAAAB3u...QHLzF9 admin@example.com'

eos_user { 'admin':
  privilege  => 15,
  role       => 'network-admin',
  encryption => sha512,
  secret     => '$1$rnfsWaC6$ZFPdsxxLS4wvSCA9p1wGg/',
  sshkey     => $pub_key,
}
```

Parameters

encryption : Defines the encryption format of the password provided in the corresponding secret key. Note that cleartext passwords are allowed via manual CLI user creation but are not supported in this module due to security concerns and idempotency.

Valid values are `md5`, `md5`, `sha512`, `sha512`.

ensure : The basic property that the resource should be in.

Valid values are `present`, `absent`.

name : The switch CLI username.

nopassword : Create a user with no password assigned.

Valid values are `true`, `yes`, `on`, `false`, `no`, `off`.

privilege : Configures the privilege level for the user. Permitted values are integers between 0 and 15. The EOS default privilege is 1.

provider : The specific backend to use for this `eos_user` resource. You will seldom need to specify this — Puppet will usually discover the appropriate provider for your platform.Available providers are:

eos Manage user accounts on Arista EOS. Requires rbeapi rubygem.

role : Configures the role assigned to the user. The EOS default for this attribute is managed with aaa authorization policy local default-role; this is typically the network-operator role.

secret : This key is used in conjunction with encryption. The value should be a hashed password that was previously generated.

sshkey : Configures an sshkey for the CLI user. This sshkey will end up in /home/USER/.ssh/authorized_keys. Typically this is the public key from the client SSH node.

---

eos_varp

Manage global VARP settings on Arista EOS. Configure the Virtual-ARP mac address.

Example:

```
eos_varp { 'settings':
    mac_address => '001c.7300.0099',
}
```

Parameters

ensure : The basic property that the resource should be in.

Valid values are `present`, `absent`.

mac_address : Assigns a virtual MAC address to the switch.

name : Resource name defaults to 'settings' and is not used to configure EOS. Returns an error if a name other than 'settings' is specified.

provider : The specific backend to use for this `eos_varp` resource. You will seldom need to specify this — Puppet will usually discover the appropriate provider for your platform.Available providers are:

eos Manage global VARP virtual MAC address on Arista EOS. Requires rbeapi.

---

eos_varp_interface

Manage VARP interface settings on Arista EOS. Will create interface with the designated name if none exists when assigning Virtual-ARP shared_ip addresses.

Example:

```
eos_varp_interface { 'Vlan2':
  shared_ip => '192.0.2.1',
}
```

Parameters

ensure : The basic property that the resource should be in.

Valid values are `present`, `absent`.

name : Resource name for the VARP interface instance.

provider : The specific backend to use for this `eos_varp_interface` resource. You will seldom need to specify this — Puppet will usually discover the appropriate provider for your platform.Available providers are:

eos Manage interface VARP config on Arista EOS. Requires rbeapi rubygem.

shared_ip : Array of virtual IP addresses for the interface.

---

eos_vlan

Manage VLANs on Arista EOS.

Examples:

```
eos_vlan { '1':
  vlan_name => 'default',
}

eos_vlan { '4094':
  enable       => true,
  vlan_name    => 'MLAG_control',
  trunk_groups => ['trunkpeer'],
}

# Remove all un-managed VLANs
resources { 'eos_vlan': purge => true }
```

Parameters

enable : The enable property configures the administrative state of the VLAN ID. When enable is configured as true, the ports forward traffic configured with the specified VLAN and when enable is false, the specified VLAN ID is blocked. Valid VLAN ID values:

- true - Administratively enable (active) the VLAN
- false - Administratively disable (suspend) the VLAN

Valid values are `true`, `false`.

ensure : The basic property that the resource should be in.

Valid values are `present`, `absent`.

provider : The specific backend to use for this `eos_vlan` resource. You will seldom need to specify this — Puppet will usually discover the appropriate provider for your platform.Available providers are:

eos Manage VLANs on Arista EOS. Requires rbeapi rubygem.

trunk_groups : The trunk_groups property assigns an array of trunk group names to the specified VLANs. A trunk group is the set of physical interfaces that comprise the trunk and the collection of VLANs whose traffic is carried only on ports that are members of the trunk gorups to which the VLAN belongs

Example configuration

```
trunk_groups => ['group1', 'group2']
```

The default configure is an empty list

vlan_name : The vlan_name property configures the alphanumber VLAN name setting in EOS. TThe name consists of up to 32 characters. The system will automatically truncate any value larger than 32 characters.

vlanid : (**Namevar:** If omitted, this parameter's value defaults to the resource's title.)

The name parameter specifies the VLAN ID to manage on the node. The VLAN ID parameter must be in the valid VLAN ID range of 1 to 4094 expressed as a String.

---

eos_vrrp

Manage VRRP settings on Arista EOS. Configures the Virtual Router Redundancy Protocol settings.

Example:

```
eos_vrrp { 'Vlan50:10':
  description      => 'Virtual IP'
  priority         => 100,
  preempt          => true,
  primary_ip       => '192.0.2.1',
  secondary_ip     => ['10.2.4.5'],
  timers_advertise => 10,
}
```

Parameters

delay_reload : Delay between system reboot and VRRP initialization. Value must be a postive integer. Default value is 0.

description : Associates a text string to a virtual router.

enable : Enable the virtual router. Default value is :true

Valid values are `true`, `yes`, `on`, `false`, `no`, `off`.

ensure : The basic property that the resource should be in.

Valid values are `present`, `absent`.

ip_version : The VRRP version for the VRRP virtual router. Valid values are 2 and 3. Default value is 2.

mac_addr_adv_interval : Interval in seconds between advertisement packets sent to VRRP group members. Value must be a postive integer. Default value is 30.

name : The name parameter is a composite namevar that combines the name of the layer 3 interface with the virtual router ID. The virtual router ID must be between 1 - 255. Both values are seperated by the colon (:) character

For example, if the interface name is Vlan50 and the virtual router ID is 10 then the namvar would be constructed as "Vlan50:10"

The composite namevar is required to uniquely identify the specific layer 3 interface and virtual router ID to configure.

preempt : A virtual router preempt mode setting. When preempt mode is enabled, if the switch has a higher priority it will preempt the current master virtual router. When preempt mode is disabled, the switch can become the master

virtual router only when a master virtual router is not present on the subnet, regardless of priority settings. Default value is :true

Valid values are `true`, `yes`, `on`, `false`, `no`, `off`.

preempt_delay_min : The minimum time in seconds for the virtual router to wait before taking over the active role. Value must be a postive integer. Default value is 0.

preempt_delay_reload : The preemption delay after a reload only. This delay period applies only to the first interface-up event after the router has reloaded. Value must be a postive integer. Default value is 0.

primary_ip : The primary IPv4 address for the specified VRRP virtual router. The address must be in the form of A.B.C.D. Default value is 0.0.0.0

priority : The priority setting for the virtual router. The value must be between 1 and 254. Default value is 100.

provider : The specific backend to use for this `eos_vrrp` resource. You will seldom need to specify this — Puppet will usually discover the appropriate provider for your platform.Available providers are:

eos Manage Virtual Router (VRRP) settings on Arista EOS. Requires rbeapi

secondary_ip : The secondary IPv4 address array for the specified virtual router. The IP address list must be identical for all VRRP routers in a virtual router group. The array cannot be empty. The address must be in the form of A.B.C.D

timers_advertise : The interval between successive advertisement messages that the switch sends to routers in the specified virtual router ID. The value must be between 1 and 255. Default value is 1.

track : Array of track settings. Each option in the array is a hash containing track settings. An example of the track property follows: track: [{ name: 'Ethernet2', action: 'decrement', amount: 33 }, { name: 'Ethernet2', action: 'decrement', amount: 22 }, { name: 'Ethernet2', action: 'shutdown' }]

The hash key definitions for a track entry follow: name - Name of an interface to track. action - Action to take on state-change of the tracked interface. amount - Amount to decrement the priority. Only specified if the action is set to 'decrement'.

---

eos_vxlan

Manange VXLAN interface configuration on Arista EOS. Configure logical Vxlan interface instances and settings

Example:

```
eos_vxlan { 'Vxlan1':
  source_interface => 'Loopback1',
  udp_port         => 5500,
}
```

Parameters

description : The one line description to configure for the interface. The description can be any valid alphanumeric string including symbols and spaces.

The default value for description is ''

enable : The enable value configures the administrative state of the specified interface. Valid values for enable are:

```
* true - Administratively enables the interface
* false - Administratively disables the interface
```

The default value for enable is :true

Valid values are `true`, `false`.

ensure : The basic property that the resource should be in.

Valid values are `present`, `absent`.

multicast_group : The multicast group property specifies the multicast group address to use for VTEP communication. This value configures the vxlan multicast-group value in EOS. The configured value must be a valid multicast address in the range of 224/8.

The default value for multicast_group is ‘’

name : The name parameter specifies the name of the Vxlan interface to configure. The value must be the full interface name identifier that corresponds to a valid interface name in EOS.

provider : The specific backend to use for this `eos_vxlan` resource. You will seldom need to specify this — Puppet will usually discover the appropriate provider for your platform.Available providers are:

eos Manage global VxLAN settings on Arista EOS. Requires rbeapi

source_interface : The source interface property specifies the interface address to use to source Vxlan packets from. This value configures the vxlan source-interface value in EOS

The default value for source_interface is ‘’

udp_port : The udp_port property specifies the VXLAN UDP port associated with sending and receiveing VXLAN traffic. This value configures the vxlan udp-port value in EOS. The configured value must be an integer in the range of 1024 to 65535.

The default value for the udp_port setting is 4789

---

eos_vxlan_vlan

Manage VXLAN VLAN to VNI mappings in Arista EOS.

Examples:

```
eos_vxlan_vlan { '100':
  vni => '100',
}

eos_vxlan_vlan { '200':
  vni => '10.10.200',
}
```

Parameters

ensure : The basic property that the resource should be in.

Valid values are `present`, `absent`.

name : The VLAN ID that is associated with this mapping in the valid VLAN ID range of 1 to 4094. The VLAN ID is configured on the VXLAN VTI with a one-to-one mapping to VNI.

provider : The specific backend to use for this `eos_vxlan_vlan` resource. You will seldom need to specify this — Puppet will usually discover the appropriate provider for your platform.Available providers are:

eos Manage VxLAN VLANs on Arista EOS. Requires rbeapi

vni : The VNI associate with the VLAN ID mapping on the VXLAN VTI interface. The VNI value is an integer value in the range of 1 to 16777215.

---

eos_vxlan_vtep

Manage the global Vxlan VTEP flood list on Arista EOS.

Example:

```
eos_vxlan_vtep { '10.1.1.1': }
```

Parameters

ensure : The basic property that the resource should be in.

Valid values are `present`, `absent`.

name : The name property associates the IPv4 flood address on the specified VXLAN VNI interface. The address value is configured using address format.

Example

```
name => 192.168.10.16
```

provider : The specific backend to use for this `eos_vxlan_vtep` resource. You will seldom need to specify this — Puppet will usually discover the appropriate provider for your platform.Available providers are:

eos Manage global VxLAN VTEP flood list on Arista EOS. Requires rbeapi.

This page autogenerated on 2016-01-12 21:33:31 -0500

CHAPTER 6

Cookbook

- *Creating a Node Profile Manifest*
- *Recipe 1: Masterless / Headless*
- *Recipe 2: MLAG*
  - *Spine1 Sample*
  - *ToR Sample*

Creating a Node Profile Manifest

A common pattern is to use node profile manifests to define reusable blocks that get applied to individual nodes, as needed. Node profile manifests define contain classes which define the desired state for one or more settings. These profile classes are, then, assigned to nodes based on the node classification. Profile classes may use parameters (specified in a resource definition or Hiera) to allow customization per node.

Recipe 1: Masterless / Headless

Puppet may be run in a masterless / headless manner. This method is useful for testing as well as full deployments. When running headless, modules, manifests, etc are made available to each node (NFS, wget, git, subversion) then are applied at the node with the `puppet apply <manifest>` command. For example: `puppet apply site.pp`

Recipe 2: MLAG

Below are two sample manifests (classes) that can be applied to nodes to configure MLAG between a spine and ToR switch. This is a very basic example to illustrate the use of the eos types. A more advanced class would accept variables or read data from hiera to use for interface IDs, VLAN IDs, peer-addresses, etc.

Spine1 Sample

```
# Configure peer link and MLAG peer.
eos_vlan { "4094":
  trunk_groups => ["mlagpeer"],
}
eos_interface { "Port-Channel10":
  description => "MLAG Peer link",
  ensure => present,
}
eos_portchannel { "Port-Channel10":
  lacp_mode => active,
  members => ["Ethernet1", "Ethernet2"],
}
eos_switchport { "Port-Channel10":
  ensure => present,
  mode => trunk,
  # trunk_group => "mlagpeer",
}
eos_stp_config { "4094":
  mode => "none",
}
eos_ipinterface { "Vlan4094":
  address => "10.0.0.1/30",
}
eos_mlag { "Rack2":
  local_interface => "Vlan4094",
  peer_address => "10.0.0.2",
  peer_link => "Port-Channel10",
  domain_id => "mlag1",
  enable => true,
}

# Configure downstream links
eos_portchannel { "Port-Channel3":
  lacp_mode => active,
  members => ["Ethernet2/4"],
}
eos_mlag_interface { "Port-Channel3":
  mlag_id => 3,
  ensure => present,
}
eos_switchport { "Port-Channel3":
  ensure => present,
  mode => trunk,
  trunk_native_vlan => 300,
  trunk_allowed_vlans => [301, 302, 303, 305, 306, 307],
}

# Create vlans
```

```
eos_vlan { "300":
  vlan_name => "ztp_bootstrap",
  ensure => present,
}

$vlans = ["301", "302", "303", "305", "306", "307"]
each($vlans) |$value| { eos_vlan { $value: ensure => present, } }
```

ToR Sample

```
eos_interface { "Port-Channel3":
ensure => present,
description => "MLAG uplink to spine"
}
eos_switchport {'Ethernet1':
  ensure => present,
}
eos_switchport {'Ethernet2':
  ensure => present,
}
eos_portchannel { "Port-Channel3":
  lacp_mode => active,
  members => ["Ethernet1", "Ethernet2"],
}
eos_switchport { "Port-Channel3":
  ensure => present,
  mode => trunk,
  trunk_native_vlan => 300,
  trunk_allowed_vlans => [301, 302, 303, 305, 306, 307],
}

eos_switchport {'Ethernet3':
  access_vlan => 302,
  mode => access,
  ensure => present,
}
eos_switchport {'Ethernet4':
  access_vlan => 301,
  mode => access,
  ensure => present,
}

$vlans = ["301", "302", "303", "305", "306", "307"]

# In Puppet 3.7 with "parser = future"
#each($vlans) |$value| { eos_vlan { $value: ensure => present } }

# Existing syntax
define newvlan {
  eos_vlan { $name:
    ensure => present
  }
}
newvlan { $vlans :
}
```

CHAPTER 7

Troubleshooting

- *Introduction*
- *Submitting Issues*

Introduction

The Puppet-EOS module is developed by Arista EOS+ CS and supported by the Arista EOS+ community. Support for the module as well as using Puppet with Arista EOS nodes is provided on a best effort basis by the Arista EOS+ CS team and the community. Support for the puppet-enterprise agent extension is provided by Puppet.

For customers looking for a premium level of support, please contact your local Arista account team or email eosplus@arista.com for assistance.

Submitting Issues

The Arista EOS+ CS development team uses Github Issues to track discovered bugs and enhancement request to the Puppet-EOS module.

For defect issues, please provide as much relevant data as possible as to what is causing the issue, if and how it is reproducible, the version of EOS and Puppet being run.

For enhancement requests, please provide a brief description of the enhancement request, a use case, and the version of EOS to be supported.

The issue tracker is monitored by Arista EOS+ CS and issues submitted are categorized and scheduled for inclusion in upcoming Puppet-EOS versions.

CHAPTER 8

Developing

- *Overview*
- *Running from source*
- *Contributing*

Overview

This module can be configured to run directly from source and configured to do local development, sending the commands to the node over HTTPS/HTTP. The following instructions explain how to configure your local development environment.

Running from source

This module requires one dependency in addition to Puppet that must be checked out as a Git working copy in the context of ongoing development in addition to running Puppet from source.

- Ruby client for eAPI: rbeapi

The dependency is managed via the bundler Gemfile and the environment needs to be configured to use local Git copies:

```
cd /workspace
git clone https://github.com/arista-eosplus/rbeapi.git
export GEM_RBEAPI_VERSION=file:///workspace/rbeapi
```

Once the dependencies are installed and the environment configured, then install all of the dependencies:

```
git clone https://github.com/arista-eosplus/puppet-eos.git
cd puppet-eos
bundle install --path .bundle/gems
```

Once everything is installed, run the spec tests to make sure everything is working properly:

```
bundle exec rspec spec
```

Finally, configure the eapi.conf file for rbeapi See rbeapi for details and set the connection environment variable to run sanity tests using *puppet resource*:

```
export RBEAPI_CONNECTION=veos01
```

Contributing

Contributions to this project are gladly welcomed in the form of issues (bugs, questions, enhancement proposals) and pull requests. All pull requests must be accompanied by spec unit tests and up-to-date inline docstrings otherwise the pull request will be rejected.

CHAPTER 9

Testing Modules

- *Introduction*

Introduction

Testing infrastructure manifests and modules is, generally, the same as for any other Puppet manifest or module. The use of tooling such as puppet-lint, rspec-puppet, puppet apply with noop, and deploying canary nodes with Arista vEOS are strongly encouraged. Be aware that some tools are not immediately available on Arista EOS such as integration with beaker or server-spec.

We recommend using pre-commit hooks and Continuous Integration (CI) systems to encourage good development and testing practices on your Puppet modules.

CHAPTER 10

FAQ

- *Server: Error: ... cannot load such file – rbeapi/client*
- *Server: Error: ... provider 'eos': undefined method* api' for nil:NilClass

Server: Error: ... cannot load such file – rbeapi/client

If you see the following error on the master:

```
Server: Error: Could not autoload puppet/provider/eos_vlan/default: cannot load such␣
↪file -- rbeapi/client
```

Install the rbeapi rubygem on the server:

```
sudo gem install rbeapi
```

Server: Error: ... provider 'eos': undefined method *api' for nil:NilClass*

If you try to apply a class or nmanifest and receive the following error:

```
Server: Error: Could not prefetch eos_vlan provider 'eos': undefined method `api' for␣
↪nil:NilClass`
```

The eos provider requires a connection to an EOS device and cannot be applied on an OS that does not support Arista eAPI except in development mode.

Either ensure this manifest/class only gets applied to EOS devices or redirect eAPI communications on this system to a real or virtual EOS device:

```
export RBEAPI_CONF=/path/to/my/.eapi.conf
export RBEAPI_CONNECTION=<connection-name>
```

CHAPTER 11

Release Notes

Release 1.4.0 - January 2016

- *Known Issues*
- *Enhancements*
- *Fixed*

Known Issues

- **The minimum recommended 2015.x Puppet agent is 2015.3.2** Early versions of the 2015.x puppet agent for EOS do not store configuration in persistent storage on the switch which can cause the node to create a new SSL private-key and certificate request after each reload. Use at least release 2015.3.2 (v4.3.2), puppet-agent-1.3.5-1.eos4.i386.swix

Enhancements

- **Add requirements section to metadata (67) [jerearista]**
- **Add additional examples in docstrings (64) [jerearista]**

Fixed

- **Ensure order of array does not affect idempotency (70)** This resolves several potential, but rare, issues. In the event that a port-channel is in a state where some members were up and others not, Puppet could receive the list of members out of order, and believe that one or more members were not properly configured, reapplying their config. (4 6)

- **eos_vlan provider does not properly set trunk_groups (38)** The eos_vlan provider now properly sets the trunk_groups:

```
eos_vlan { '4094':
  trunk_groups => ['mlag_peer'],
}
```

- **mock not intercepting acl.getall call (14)**

Release 1.3 - November 2015

- *New Resource Types*
- *Enhancements*
- *Fixed*
- *Known Caveats*

New Resource Types

- **eos_vrrp (53) [devrobo]**
- **eos_routemap (52) [websitescenes]**
- **eos_config (50) [devrobo]**
- **eos_varp and eos_varp_interface (47) [websitescenes]**
- **eos_user (42) [websitescenes]**

Enhancements

- **Confine providers to only run on AristaEOS and when rbeapi >= 0.3.0 is present (48) [jerearista]** Implements puppet feature :rbeapi. Example use: `confine :feature => :rbeapi`
- **eos_system (58) [websitescenes]** Add support for managing the global 'ip_routing' setting
- **Feature bgp update (41) [websitescenes]**

Fixed

- None

Known Caveats

- **eos_portchannel members not idempotent when interface order is not the same (46)**
- **eos_vlan provider does not properly set trunk_groups (38)**
- **All providers should have a description (55)**
- **eos_stp_interface provider unit test is incomplete. (51)**

- **Cleanup documentation (19)**

Release 1.2 - August 2015

- *New Types*
- *Enhancements*
- *Resolved Issues*
- *Known Issues*

- Adds 3 new types

See GitHub issues for the current state of any known issues.

Note: puppet-eos 1.2.0 requires a minumum rbeapi version of 0.3.0. Prior versions of puppet-eos will only work with rbeapi 0.2.0 or lower.

New Types

- eos_bgp_config
- eos_bgp_network
- eos_bgp_neighbor
- eos_staticroute

Enhancements

Resolved Issues

Known Issues

Release 1.1 - July 2015

- *New Types*
- *Enhancements*
- *Resolved Issues*
- *Known Issues*

- Adds 3 new types

See GitHub issues for the current state of any known issues.

New Types

- eos_acl_entry
- eos_stp_interface
- eos_command

Enhancements

Resolved Issues

Known Issues

Release 1.0 - May 2015

- *New Types*
- *Enhancements*
- *Resolved Issues*
- *Known Issues*

- Initial public release to Puppet Forge

See GitHub issues for the current state of any known issues.

New Types

- eos_ethernet
- eos_interface
- eos_ipinterface
- eos_mlag
- eos_mlag_interface
- eos_ntp_config
- eos_ntp_server
- eos_portchannel
- eos_snmp
- eos_switchport
- eos_system
- eos_vlan
- eos_vxlan
- eos_vxlan_vlan
- eos_vxlan_vtep

Enhancements

Resolved Issues

Known Issues

CHAPTER 12

License

```
Copyright (c) 2014-2015, Arista Networks EOS+
All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions are met:

* Redistributions of source code must retain the above copyright notice, this
  list of conditions and the following disclaimer.

* Redistributions in binary form must reproduce the above copyright notice,
  this list of conditions and the following disclaimer in the documentation
  and/or other materials provided with the distribution.

* Neither the name of Arista Networks nor the names of its
  contributors may be used to endorse or promote products derived from
  this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE
FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER
CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY,
OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
```