puppet_eos Module Documentation

Release 1.2.0

Arista Networks - EOS+ Consulting Services

August 25, 2015

Contents

Contents:

CHAPTER 1

Overview

1.1 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.

1.2 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.

1.3 Prerequisites

PuppetLabs 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 2

Quick Start

2.1 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 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
!
ip route 0.0.0.0/0 192.2.2.1
!
```

From EOS 4.15.5 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 4.15.5, 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 PuppetLabs [2] (previous releases [3]):

```
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
```

Install the rbeapi extension [4]:

```
Arista#copy http://myserver/rbeapi-0.3.0.swix extensions:
Arista#extension rbeapi-0.3.0.swix
```

Save the installed extensions:

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

2.1.1 EOS Command Aliases

If working with puppet manually from the CLI, it may be convenient to add the following aliases to your systems

```
alias pa bash sudo puppet agent --environment demo --waitforcert 30 --onetime true
alias puppet bash sudo 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
```

2.2 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

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

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: puppet-eos [5] module (Source: GitHub: puppet-eos [6]). This module is self-contained including the types and providers specific to EOS.

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

Install the rbeapi rubygem on the server:

```
$ sudo gem install rbeapi
```

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

Puppet installer:

```
$ sudo puppet module install puppet-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
```

2.3 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

```
Arista#bash sudo puppet agent [--environment <env_name>] --test --onetime --no-daemonize --waitforcer
```

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

```
$puppet cert list
$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:

[5] https://forge.puppetlabs.com/aristanetworks/puppet-eos
[6] https://github.com/arista-eosplus/puppet-eos

```
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 3

Installation

3.1 Configuring the Puppet Master

Follow the standard practices for installing either Puppet Enterprise or Puppet Open-source master servers and your environment(s). As the paths to various items and specifics may vary from system to system, you might need to make minor adjustments to the instructions, below, to conform to your particular system. The command `puppet confing print` can assist you in locating the right directories.

On the master, install the Forge: puppet-eos [1] module (Source: GitHub: puppet-eos [2]). This module is self-contained including the types and providers specific to EOS. There is also a netdev_stdlib module in which PuppetLabs maintains a common set of Types in netdev_stdlib and the EOS providers are in netdev_stdlib_eos.

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

Puppet installer:

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

Install from source:

```
$ git clone https://github.com/arista-eosplus/puppet-eos.git modulepath/eos
$ git checkout <version or branch>
```

Link using Git submodules:

```
$ git submodule add https://github.com/arista-eosplus/puppet-eos.git modulepath/eos
```

3.2 Bootstrapping EOS switches

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 to automate the steps from initial power-on to contacting the

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

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 name-server vrf default 8.8.8.8
ip domain-name example.com
ip host puppet 192.2.2.5
!
interface Management1
   ip address 192.2.2.101/24
   no shutdown
!
ip route 0.0.0.0/0 192.2.2.1
!

If EOS version is 4.14.5 or later
!
management api http-commands
   no protocol https
   protocol unix-socket
   no shutdown
!

If EOS version is below 4.14.5
username eapi privilege 15 secret icanttellyou
!
management api http-commands
   no shutdown
!
```

Install the puppet agent from PuppetLabs [3]:

```
Arista#copy http://myserver/puppet-enterprise-3.7.2-eos-4-i386.swix extensions:
Arista#extension puppet-enterprise-3.7.2-eos-4-i386.swix
Arista#copy installed-extensions boot-extensions
```

Install the rbeapi extension:

```
Arista#copy http://myserver/rbeapi-0.1.0.swix extensions:
Arista#extension rbeapi-0.1.0.swix
Arista#copy installed-extensions boot-extensions
```

3.3 Additional Puppet Master configuration

3.3.1 Configuring rbeapi

Rbeapi, in many cases, requires a configuration file describing its connection method and credentials to eAPI on the switch. Available transports include https, http, http-local, and unix socket (EOS 4.14.5). Unix socket is recommended if available in the running version of EOS due to ease of configuration and security posture. The /mnt/flash/eapi.conf file (also flash:eapi.conf) can be installed at bootstrap time or by puppet afterward. To do so with puppet, modify the sample files, below, to meet your needs.

[3] https://puppetlabs.com/download-puppet-enterprise-all#eos

Create the module skeleton on the Puppet master:

```
cd <modulepath>
puppet module generate <username-modulename>
mkdir <username-modulename>/templates/
```

Create an eapi.conf template in <modulepath>/<username-modulename>/templates/eapi.conf.erb

```
<%# rbeapi/templates/eapi.conf.erb %>
# Managed by Class['rbeapi']
[connection:localhost]
<% if @host -%>
host: <%= @host %>
<% end -%>
<% if @_transport != "http" -%>
transport: <%= @_transport %>
<% end -%>
<% if @_username != "admin" -%>
username: <%= @_username %>
<% end -%>
<% if @_password != "" -%>
password: <%= @_password %>
<% end -%>
<% if @port -%>
port: <%= @port %>
<% end -%>
```

Create a class that can be applied to nodes in <modulepath>/<username-modulename>/manifests/init.pp

```
# modules/rbeapi/manifests/init.pp
# Example to configure eAPI for use with rbeapi
#   class { rbeapi:
#    username => eapi,
#    password => icanttellyou,
#  }
class rbeapi ($host = "localhost",
              $transport = https,
              $username = admin,
              $password = "") {

  package { 'rbeapi':
    ensure => installed,
    provider => 'gem',
  }

  # Check the EOS version (split in to major.minor.patch)
  $section = split($::operatingsystemrelease, '\.')
  $major = $section[0]
  $minor = $section[1]
  if $section[2] =~ /^(\d+)/ {
    $patch = $1
  } else {
    $patch = 0
  }

  # eapi.conf can use "socket" starting with EOS 4.14.5
  if $major >= 4 and $minor >= 14 and $patch >= 5 {
    $_transport = socket
    # The following defaults cause the template to skip
    #   user/pass sections
```

```
    $_username = admin
    $_password = ""
  } else {
    # Just pass through values we received
    $_transport = $transport
    $_username = $username
    $_password = $password
  }

  # Populate the eapi.conf file
  file { 'eapi.conf':
    path => '/mnt/flash/eapi.conf',
    ensure => file,
    content => template("rbeapi/eapi.conf.erb"),
    require => Package['rbeapi'],
  }
}
```

CHAPTER 4

Types

4.1 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 2015-08-21 08:20:39 -0400)*

4.2 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.

4.2.1 eos_acl_entry

This type provides management of ACLs on the Arista EOS node from within Puppet.

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 :

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.

4.2.2 eos_bgp_config

Provides resource management of the global BGP routing process for Arista EOS nodes.

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`.

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 :

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

4.2.3 eos_bgp_neighbor

Provides stateful management of the neighbor statements for the BGP routing process for Arista EOS nodes.

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 :

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`.

4.2.4 eos_bgp_network

Provides stateful management of the network statements for the BGP routing process for Arista EOS nodes.

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 :

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.

4.2.5 eos_command

Execute commands on the EOS node. Commands can be either priviledged mode (enable) commands or configuration commands.

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 :

4.2.6 eos_ethernet

This type provides management of physical Ethernet interfaces on Arista EOS nodes from within Puppet. Physical Ethernet interfaces include the physical characteristics of front panel data plane ports with but does not include the out-of-band Management interface.

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 :

4.2.7 eos_interface

This type provides management of Arista EOS interfaces. The type is used as a basis type for any interface available in EOS and therefore the properties are common across all interface types

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 :

4.2.8 eos_ipinterface

This type provides management of logical IP interfaces configured in EOS. It provides configuration of IPv4 properties on physical interfaces and logical virtual interfaces.

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 :

4.2.9 eos_mlag

This type manages the global MLAG instance on EOS nodes. It provides configuration for global MLAG configuration parameters.

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 :

4.2.10 eos_mlag_interface

This type manages MLAG interfaces on the node used to establish a valid MLAG with a peer switch. The mlag_id parameter is required for this type.

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 :

4.2.11 eos_ntp_config

This type manages the nodes global NTP configuration settings. It provides a configuration resource for setting global NTP values

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 :

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 ''

4.2.12 eos_ntp_server

This type manages the list of NTP servers. It provides a configuration resource for managing the list of NTP servers used by the node.

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 :

4.2.13 eos_portchannel

This type manages Port-Channel interface instances on Arista EOS nodes. It provides configuration resources for logical Port-Channel instances and settings

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 with a valid range betwee 1 and 100.

The default value is 90

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 :

4.2.14 eos_snmp

This type manages the global SNMP configuration instance on EOS nodes. It provides configuration resources for global SNMP settings.

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 :

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 ‘’

4.2.15 eos_staticroute

Configure static route settings

Parameters

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

Valid values are `present`, `absent`.

name : The destination network prefix

route_name : The name assigned to the static route

4.2.16 eos_stp_interface

Manage Spanning Tree Protocol interface configuration.

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 :

4.2.17 eos_switchport

This type provides a resource for configuring logical layer 2 switchports in EOS. The resource provides configuration for both access and trunk operating modes.

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.

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 :

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

4.2.18 eos_system

This type manages the global EOS node settings. It provides configuration of global node attributes.

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'

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 :

4.2.19 eos_vlan

This type provides management of VLANs on the Arista EOS node from within Puppet.

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 :

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.

4.2.20 eos_vxlan

This type mananges VXLAN interface configuration on Arista EOS nodes. It provides configuration of logical Vxlan interface instances and settings

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 :

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

4.2.21 eos_vxlan_vlan

This type manages the VXLAN VLAN to VNI mappings in the nodes current running configuration. It provides a resources for ensuring specific mappings are present or absent

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 :

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.

4.2.22 eos_vxlan_vtep

This type provides management of the global Vxlan VTEP flood list.

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.

For 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 :

This page autogenerated on 2015-08-21 08:20:42 -0400

CHAPTER 5

Cookbook

5.1 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.

5.2 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.

5.3 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 useful class would accept variables or read data from hiera to use for interface IDs, VLAN IDs, peer-addresses, etc.

5.3.1 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, } }
```

5.3.2 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 6

Troubleshooting

6.1 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 PuppetLabs.

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

6.2 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 7

Developing

7.1 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.

7.2 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
```

7.3 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 8

Testing Modules

8.1 Step 1

CHAPTER 9

FAQ

9.1 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/c
```

Install the rbeapi rubygem on the server:

```
sudo gem install rbeapi
```

9.2 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 10

Release Notes

10.1 Release 1.2 - August 2015

- 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.

10.1.1 New Types

- eos_bgp_config
- eos_bgp_network
- eos_bgp_neighbor
- eos_staticroute

10.1.2 Enhancements

10.1.3 Resolved Issues

10.1.4 Known Issues

10.2 Release 1.1 - July 2015

- Adds 3 new types

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

10.2.1 New Types

- eos_acl_entry
- eos_stp_interface
- eos_command

10.2.2 Enhancements

10.2.3 Resolved Issues

10.2.4 Known Issues

10.3 Release 1.0 - May 2015

- Initial public release to Puppet Forge

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

10.3.1 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

10.3.2 Enhancements

10.3.3 Resolved Issues

10.3.4 Known Issues

CHAPTER 11

License