Contents

Definitions

- **Component modules:** Normal modules that manage one particular technology. (Forexample, puppetlabs/apache.)
- **FQDN:** Fully qualified domainname.
- **Idempotence:** The property of certain operations in mathematics and computer science,that

canbeappliedmultipletimeswithchangingtheresultbeyondtheinitialapplication.Catalogcanbeapplied multiple times without causing issue.

- **Profiles:** Wrapperclassesthat to configure alayeredtechnology stack.
- **Roles:** configuration.

Introduction

Introduction to Puppet

Puppet Head First

- Install the Puppet Master: `./puppet-enterprise-installer`
- InstallthePuppetAgent:`curl-khttps://<puppet-master-fqdn>:8140/packages/ current/install.bash | sudo bash`
- Puppet module install: `puppetlabs-ntp --version6.0.0`
- Modules installed in `/etc/puppetlabs/code/environments/production/modules`
- `site.pp` in`/etc/puppetlabs/code/environments/production/manifests`
- `puppet agent –t`executes a Puppet run in the foreground.

Nodes

- Supported Operating Systems

InstallingPuppet

- Install the Puppet Master: `./puppet-enterprise-installer`
- InstallthePuppetAgent:`curl-khttps://<puppet-master-fqdn>:8140/packages/ current/install.bash | sudo bash`

Puppet Installation Flags

- -c-Useape.conffiletoconfigurethePuppetserver.
- -D-Displaysdebugginginformation.
- -h-Displayhelp
- -q-Runinquietmode;theinstallationprocessisnotdisplayed.Requiresanswerfile.
- -V-displayveryverbosedebugginginformation
- -y-Assumesyes/defaultandbypassany user input.

pe.conf

Thepe.conffile neededtoinstalland configure

Found in /etc/puppetlabs/enterprise/conf.d

Sample pe.conffile:

```
{
  "console_admin_password": "password",
  "puppet_enterprise::puppet_master_host": "<puppet-master-fqdn>",
  "pe_install::puppet_master_dnsaltnames": [
    "puppet"
  ]
}
```

Installation Directories

- Puppet configurationfilesareinstalledin/etc/puppetlabs/puppetfor*nixnodesand <COMMON_APPDATA>\PuppetLabs forWindows nodes.
- PuppetEnterprisesoftwarebinariesareinstalledin/opt/puppetlabs
- Executablebinariesarein/opt/puppetlabs/bin /opt/puppetlabs/sbin
- Theinstallerautomaticallycreatessymlinksin/usr/local/bin

Code and Data Directories

- **R10k:** /etc/puppetlabs/r10k
- **Environments:** /etc/puppetlabs/code/environments
- modules: Main directory for puppet modules (applies to master only)

- manifests: Contains the main starting point for catalog compilation (applies to master only)
- ssl:Containseachnodescertificateinfrastructure(allnodes)/etc/puppetlabs/puppet/ssl

Puppet Enterprise Logs

All Puppet Enterprise logs can be found in /var/log/puppetlabs.

- Puppet master logs:/var/log/puppetlabs/puppetserver
- Puppet agent logs: /var/log/messages or/var/log/system.log
- ActiveMQ logs:/var/log/puppetlabs/activemq
- MCollectiveservice /var/log/puppetlabs/
- Console /var/log/puppetlabs
- Installer /var/log/puppetlabs/installer
- Database logs: /var/log/puppetlabs/puppetdb and /var/log/puppetlabs/ postgresql

- Orchestration logs:/var/log/puppetlabs

Puppet Ports

- **3000:** Used the web-based installer of the PuppetMaster.
- **8140:**The that the Puppet Master and communicate on.
- **61613:** Used by MCollective for orchestration requests by Puppet agents.
- **443:**The port used to access the Puppet Enterprise Console.
- **5432:** PostgreSQL runs on this port. It is used by PuppetDB in a split stackconfiguration.
- **8081:**The traffic/requestport.
- **8142:** Used by Orchestration services to accept inbound traffic/responses from the Puppetagents.

Puppet Enterprise Services

On CentOS 7 the Puppet Enterprise services are installed in /usr/lib/systemd/system.

- pe-activemq: The ActiveMQ message server, which passes messages to the MCollective servers on agent nodes. Runs on servers with the Puppet master component.

- pe-console-services: Manages and serves the PEconsole.

- pe-puppetserver:ThePuppetmasterserver,whichmanagesthePuppetmastercomponent.

- pe-nginx:Nginx,servesasareverse-proxytothePEconsole.
- mcollective:The MCollective daemon, which listens for messages and invokes actions. Runs on everyagentnode.
- puppet(onELandDebian-basedplatforms):ThePuppetagentdaemon.Runsoneveryagent node.
- pe-puppetdbandpe-postgresql:Daemonsthatmanageandservethedatabasecomponents. Notethatpe-postgresqlisonlycreatedifweinstallandmanagePostgreSQLforyou.
- pe-orchestration-services: Runs the Puppet orchestration process.
- pxp-agent:RunsthePuppetagent

Puppet Enterprise Logs

AllPuppet /var/log/puppetlabs

- Puppetmaster logs /var/log/puppetlabs/puppetserver
- Puppetagentlogs:/var/log/messages /var/log/system.log
- ActiveMQ /var/log/puppetlabs/activemq
- MCollectiveservicelogs:/var/log/puppetlabs/
- Console /var/log/puppetlabs
- Installer /var/log/puppetlabs/installer
- Database /var/log/puppetlabs/puppetdb and /var/log/puppetlabs/postgresql
- Orchestration logs:/var/log/puppetlabs

puppet.conf

Thepuppet.conffileislocated /etc/puppetlabs/puppet.

- Configsections
  - main is the global section used by all commands and services. It can be overridden by the other sections.
  - master is used by the Puppet master service and the Puppet cert command.
  - agent is used by the Puppet agent service.
  - user is used by the Puppet apply command

Note: Settings are loaded at service start time, to apply changes made to puppet.conf a restart to the pe- puppet service is required.

- Interpolating variables
  - The values of settings are available as variables within puppet.conf, and you can insert them into the values of other settings. To reference a setting as a variable, prefix its name with adollar sign.
  - Example:
    - $codedir
    - $confdir
    - $vardi

r Sample

puppet.conf

```
   [main]
   certname =
   master.vagrant.vm server
   = master.vagrant.vm user
   = pe-puppet
   group = pe-puppet
   environment_timeout=0
   app_management = true
   module_groups  =
   base+pe_only
   environmentpath =
   /etc/puppetlabs/code/environments codedir   =
   /etc/puppetlabs/code
   [agent]
   graph =
   true
   [master]
   node_terminus = classifier
   storeconfigs = true
   storeconfigs_backend =
   puppetdb reports  =puppetdb
   certname =
master.vagrant.vmalways_ca
che_features  =true
```

Sample puppet.confforan node.

```
[main]
server =
master.vagrant.vm
certname
=agent1.vagrant.vm
```

- Basic settings
    - always_retry_plugins:AffectshowwecacheattemptstoloadPuppetresourcetype sand features.
    - basemodulepath: The search path for global modules. Should be specified as a listof

directories separated by the system path separator character.

- Default: $codedir/modules:/opt/puppetlabs/puppet/modules
- ca_server:Theservertouseforcertificateauthorityrequests.
- certname:The name to use when handling certificates.
- dns_alt_names:AlistofhostnamestheserverisallowedtousewhenactingasthePupp et master.Thehostnamethatanagentusesmustbeincludedthislistortheagentwillfailconnecti ng tomaster.Thehostnamecanalsoliveinthecertnamesetting.
- environment: Defaults to production, environment to request but can be overridden by masters ENC (External NodeClassifier).
- environmentpathlist of directories separated bythesystem

- manifestdirectory of manifests if one exists orifthepathends / .

- reports: list of report multiple report handlers, their names should be comma-separated, with (For example, reports = http, log, store.)
- http: reports via HTTP or HTTPS. This report processor submits reportsas POSTrequests to address in the reporturl setting. The body of eachPOSTrequest theYAMLdump of a Puppet::Transaction::Report object, andtheContent-T is set as application/x-yaml.
- log: all received logs to the local log destinations. Usually the log destination issyslog.
- store Store the YAML reportondisk. host sends its report as aY dump and this just stores file on disk, inthereportdir .
- Default:
- rundir locationwherePuppetPIDfilesarestored.
- server masterservertowhichthePuppetagent
- ssldir:Thelocation SSLcertsarestored.
- vardir: The locationwherePuppet growing information.

- Run behavior settings

- ignoreschedules: Schedules allow you to only execute a resource if it's during a specific time period; this setting can disable that feature that might be used when you are doing an initial setup on a node and everything needs to be executed or enforced the first timearound
- noop: Agent will not do any work only simulate changes and report to the master.
- postrun_co mand: command to run after Puppet commandexecute

- prerun_co mand: command to run before Puppet commandexecutes
- prio rity:The scheduling priority of the process.Valid values are 'high', 'normal', 'low', or 'idle', which are mapped to platform-specificvalues.
- report:Whethertosendreportsaftereverytransaction.
- runinterval:howoftenthepuppetagentdaemonruns
- tags:LimitthePuppetruntoincludeonlyresourceswithcertaintags(cool),specificdat a centers, etc
- usecacheonfa ilure:Whether to configuration when the remote configuration will notcompile.
- waitforcert: not initially available (gives time for the

Resource AbstractionLayer

- Describing/declaring the state
- Providers enforce the desired state

Resource Type:

- Every resource is managed by resource type
  - a title
  - a set attributes.

```
     <TYPE>{'<TITLE>':
<ATTRIBUTE> ><VALUE>,
     }
```

Example

```
     user { 'username':
       ensure
                  >present
       ,
       uid        >'102',
       gid        >'wheel',
       shell      >'/bin/bash',
       home
                  >'/home/usernam
       e',managehome>'',
    }
```

Commands

- puppetdescribewillprovideinformationaboutresourcetypeswithinPuppet
- puppetdescribe–llistsallresourcetypesavailable
- puppetdescribe-s<type>givesshortinformationaboutresourcetype
- puppet describe <type>gives a long listing information aboutresource
- puppetresourcewilldescribeinformationaboutresourcesalreadyinstalledonarunningnode
- puppet resource<type>
- puppet resource <type><name>
- puppet agent information about the node, this is
  - puppet agent
  - A puppetagent

Facter

- facter:Returnsalistallfacts.
- facter<fact>:Returnsaparticularfact.
- facter-pAllowsFactertoloadPuppet-specificfacts.

CertificateSigningRequest(CSR)

Puppet Server certificate authority (CA) service that accepts requests (CSRs) from nodes, serves ates and a certificate revocation list (CRL) to optionally accepts commands to sign certificates.

Command:

```
puppet cert
puppetcertlis
t
puppet cert sign
<NAME>puppet cert
revoke <NAME>
```

DNS altnames:

```
puppet cert sign (<HOSTNAME>or --all) --allow-dns-alt-names
```

Regenerating Certificates

On the Puppet Master

```
puppet cert clean<NAME>
```

Deleting SSL Certs on Agent

```
cp -r /etc/puppetlabs/puppet/ssl/
/etc/puppetlabs/puppet/ssl_bak/
```

Autosigning

- Should only be used able to connect to the Puppet master.
- The

$confdir/autosign.conf

- .domain.com

Building Modules andClasses

Class Structure ane Names

- Class names can have:
  - Lowercase letters
  - Digits
  - Underscores

```
\A[a-z][a-z0-9_]*\Z
```

- Namespaceseparatorusedouble ::

```
\A([a-z][a-z0-9_]*)?(::[a-z][a-z0-9_]*)*\Z
```

- [Reserved Variable Names] [Reserved Variable Names]: (https://docs.puppet.com/puppet/4.5/lang_ reserved.html#reserved-variable-names)

Class Syntax:

```
class <CLASS_NAME>(
  <DATA_TYPE><PARAM_NAME>
) {
  ...puppetcode  .
}
```

Example:

```
class ssh {
  file {
    "/etc/ssh/ssh_config":
    ensure >file,
    source >"puppet:///modules/ssh/ssh_config"
  }
}
```

Module Structure and Names

- Module
  - Lowercase
  - Numbers
  - Underscores
- Should begin with a lowercaseletter.
- Module cannot contain namespace separator (::
- Modules cannot be nested

```
<MODULE NAME>
  manifes
  ts files
  templat
  es lib
  facts.d
  example
  s spec
  functio
  ns
  types
```

Module Directories

- `manifests/`— Contains all of the manifests in the module.

- files/—Containsstaticfiles,whichmanagednodescandownload.
- lib/—Containsplugins,likecustomfactsandcustomresourcetypes.
- facts.d/—Containsexternalfacts,whichareanalternativetoRuby-basedcustomfacts.

- templates/—Containstemplates,whichthemodule'smanifestscanuse.
- examples/—Containsexamplesshowinghowtodeclarethemodule'sclassesanddefinedtypes.
- spec/—Containsspectestsforanypluginsinthelibdirectory.
- functions/—ContainscustomfunctionswritteninthePuppetlanguage.
- types/—Containstypealiases.

Autoloading

- Names map to thefile
  - First segment
  -
  - The last
  - Any segments manifestsdirectory.

Example

```
apache-
<MODULEDIRECTORY>/apache/manifests/init.pp
apache::mod-
<MODULEDIRECTORY>/apache/manifests/mod.pp
apache::mod::passenger-<MODULEDIRECTORY>/apache/manifests/mod/
passenger.pp
```

CustomandExternalFacts

Custom Facts

- Custom of Ruby code on the Puppetmaster.
- Usually shell commands are issued as part of the fact to return information.
- Executed on the Puppet nodes with the Plugin Module.
- Custom facts are located in<MODULE>lib/facter.

Example:

```
# hardware_platform.rb
Facter.add('hardware_platform')  do
  setcodedo
    Facter::Core::Execution.exc('/bin/uname  --hardware-
platform') end
```

```
end
```

- Facts distributed using pluginsync
  - Enabled in the [main] section of `puppet.conf` by setting`pluginsync=true`

External Facts

External facts provide a way to use arbitrary executables or scripts as facts, or set facts statically with structured data.

In a Module:

```
<MODULEPATH>/<MODULE>/facts.d/
```

On Unix/Linux/OS X:

```
/opt/puppetlabs/facter/facts.d/
/etc/puppetlabs/facter/facts.d/
/etc/facter/facts.d/
```

On Windows:

```
C:\ProgramData\PuppetLabs\facter\facts.d\
```

On Windows 2003:

```
C:\DocumentsandSettings\AllUsers\ApplicationData\PuppetLabs\fa
cter\ facts.d\
```

STDOUT in the Format:

```
key1=valu
e1
key2=valu
e2
key3=valu
e3
```

Structured Data Facts:

```
yam
```

l
jso
n
txt

DSLOverview

Resource Types

- Resource types are the basic building blocks of the Puppet DSL.
- Every resource type has:
    - a title
    - a set of attributes

```
<TYPE>{'<TITLE>':
  <ATTRIBUTE> ><VALUE>,
}
```

- Example file
    - ensure:
        - file: make sure it's a
        - directory: makes sure it is a directory (enables recursive)
        - linkensures file is a symlink (requires target attribute)
        - absent: deletes file if itexists
    - Attributes:
        -
        -
        - tar
- Review all types by visiting the Resource TypeReference

StyleGuide

- The style guide is to promote consistent formatting in the Puppet Language, especially across modules, giving users and developers of Puppet modules a common pattern, design, and style tofollow.
    - Readability matters.
    - Scoping and simplicity are key.
    - Your module is a piece ofsoftware.

- Version yourmodules.

Spacing, Indentation, and Whitespace

- Module manifests:
  - Must use two-space soft tabs,
  - Must not use literal tab characters,
  - Must not contain trailing whitespace,
  - Must include trailing commas after attributes and parameterdefinitions,
  - Must end the last line
  - Must use between the opening
- Module
  - Should not limit would impractical
  - Should leave one empty line when using dependency chains
  - May align hash rockets (=>) within blocks attributes, one space after longest resource key, arranging hashes for maximum readabilityfirst.

Example:

```
file{'/tmp/foo':     ...}
```

Arrays and Hashes

- Each element on line
- Each new indented one level
- First and last lines used for the syntax of that data type

Example

```
#arraywithmultipleelementsonmultiplelines
service { 'some_service':
  require
          >[File['some_conf
    ig_file'],
    File['some_sysconfig_fil
    e'],
  ],
        }
```

Quoting

- All strings must be enclosed in single quotes, unless the string:
  - Contains variables
  - Contains single quotes
  - Contains escaped characters not supported by single-quoted strings
  - Is an enumerable set of options, such as present/absent, in which case the single quotes are optional
- All variables must be enclosed in in a string.
- Double quotes should single quotes, unless that would require an

Example

```
file{"/tmp${file_name}":    …}
"${facts['operatingsystem']} is not supported by ${module_name}"
warning("Class[class_name'] doesn't work they way you expected
it too.")
```

Escape Characters and Comments

- Puppet uses backslash as an escapecharacter.
  - Escaping as \\ would be "\\\\"
- Comments must be hash comments (#This comment), not /* */
- Documentation comments for Puppet Strings should be included for each of classes,definedtypes, functions, resource types and providers.

Example

```
# Configures sshd
file{'/etc/ssh/ssh_config':    .}
```

Module Metadata

- Every module must have metadata defined in the metadata.json file.
- Hard dependencies must be declared in your module's metadata.json file.
- Soft dependencies should in the README.md.

Example

```
{
  "name": "tthomsen-
  my_module_name", "version":
  "0.1.0",
  "author": "TravisN.
  Thomsen","license":
  "Apache-2.0",
  "su mary": "It's a modules that does things",
  "source":"https://github.com/mygithubaccount/tthomsen-
  my_module_
name",
  "project_page":"https://github.com/mygithubaccount/tthomsen-
my_ module_name",
  "issues_url":"https://github.com/mygithubaccount/tthomsen-
my_module_ name/issues",
 "tags": ["things", "and", "stuff"],
 "operatingsystem_support": [
    {
      "operatingsystem":"Red
      Hat",
      "operatingsystemreleas
      e":[
        "5.0",
        "6.0"
      ]
    },
    {
      "operatingsystem": "Ubuntu",
      "operatingsystemrelease":[
        "12.04",
        "10.04"
     ]
    }
  ],
  "dependencies": [
    { "name": "puppetlabs/stdlib", "version_requirement": " =
    3.2.0
<5.0.0" },
  ]
}
```

Resources

- All resource names or must be quoted.
- Hash rockets (=>) in a resource's attribute/valuelistmay aligned.
- Ensure should be the first attributespecified.
- Resources should be grouped by logical relationship to each other, rather than by resourcetype.
- Semicolon-separated multiple resource bodies should be used only in conjunction with a local defaultbody.

Example

```
file{'/etc/ssh/ssh_config':
      ensure
              >file,m
      ode
              >"0600"
      ,
    }
```

Classes and Defined Types

- All classes and resource type definitions (defined types) must be separate files in themanifests directory of the module. Each separate file in the manifest directory of the module should contain nothing other than the class or resource typedefinition.

Example

```
#/etc/puppetlabs/code/environments/production/modules/apache/
manifests
# init.pp
class apache { }
#ssl.pp
class apache::ssl { }
# virtual_host.pp
define apache::virtual_host () { }
```

- When a resource include class,nodedefinition, definedtype,it is included in all catalogs. Thiscan have and is not always easy to detect.

Example

```
    #manifests/init.pp:
    class { 'some_class':
includesome_other_class
    }
```

Chaining Arrow Syntax

- When you many interdependent or order-specificitems,chaining be used.

Example

```
#Pointslefttoright
Package['package_name']  >Service['service_name']
#Onthelineoftheright-handoperand
Package['package_name']
 >   Service['service_name']
```

Nested Classes or Defined Types

- Don't define classes and defined resource types in other classes or definedtypes.
- Classes and defined types should be declared as close to node scope aspossible.

- Seriously, dude, don't nest classes or definedtypes!

Example of Bad Behavior:

```
class some_class {
  classa_nested_class{  .}
}
class some_class {
  definea_nested_define_type(){ .}
}
```

Parameter

- Declare required parameters
- Optional parameters
- Declare
- For Puppet 4.9.0 automatic parameter lookup for class parameters.
- Puppet versions less than 4.9.0, use In simple cases, you also specify the default values directly in the class ordefined

Example:

```
# parameter defaults provided via APL > puppet
4.9.0 class some_module (
  String$source,
  String$config,)
  {
    .puppetcode   .
}
```

Class Inheritance

- Class inheritance should not be used.
- Usedatabindinginsteadofparams.pppattern.
- Inheritance should only be used for params.pp, which is not recommended in Puppet 4.9.
- For maintaining older modules inheritance can be used but must not be used across module namespaces.

Example:

```
classssh{   .}
classssh::clientinheritsssh{  .}
classssh::serverinheritsssh{  .}
```

Defined Resource Types

- Defined resource types are notsingletons.
- Uniqueness
  - Can have multiple instances.
  - Resource names must be unique.

Variables

- Referencing facts
  - Whenreferencing $facts variables.
    - It's
    -
    - Distinguishes
  - Example: $facts['operatingsystem']
- Namespacing variables
  - When referencing top-scope variables other than facts, explicitly specify absolute namespaces for clarity improved readability. This includes top-scope variablessetby nodeclassifierand in the main manifest.
  - This necessary for:
    - the$factshash.
    - the$trustedhash.
    - the $server_factshash.
- Variable
  - Usenumbers
  - Use lowercase letters
  - Useunderscores
  - Don't use camel case
  - Don't use dashes

Good Examples:

- $this_is_vairable

- $so_is_this
- $also_good123

Bad Examples:

- $ThisIsNotGood
- $neither-is-this

Conditionals

- Keep resource declarations simple.
  - Don't mix conditionals
  - Separate
- Defaults
  - Case statements
  - Case and selector values

Example:

```
    $file_mode=$facts['os']['family'
      ]?{ 'Debian'>'0007',
'RedHat' >'0776',
       default >'0700',
    }
    file {
      '/tmp/readme.txt':
      ensure  >file,
      content
              >"HelloWorld\
      n", mode >$file_mode,
    }
    case $Facts[::operatingsystem]
      { 'centos':{  $version='1.2.3'
                    }
      'debian':{   $version='3.4.5' }
      default: {    fail("Module${module_name}isnotsupportedon
    ${::operatingsystem}") }
    }
```

- Review the Puppet StyleGuide.

Core Data Types

- The most common data types:

  - String
  - Integer, Float, andNumeric
  - Boolean
  - Array
  - Hash
  - Regexp
  - Undef
  - Default

Resource and Class References

- Resources
- However,they

Abstract Data Types

- Abstract data types let you do more sophisticated or permissive type checking.
  - Scalar
  - Collection
  - Variant
  - Data
  - Pattern
  - Enum
  - Tuple
  - Struct
  - Optional
  - Catalogentry
  - Type
  - Any
  - Callable

The Type Data Type

- All data types are of typeType.

Syntax:

```
Type[<ANY  DATA  TYPE>]
```

Example:

- Type: matches any data type, such as Integer String, Any, or Type.
- Type[String]: matches the of its more specific subtypes like String[3] or Enum["running",
- Type[Resource] reference.

RelationshipsandDependencies

Relationship Metaparameters

By default, Puppet applies resources in the order they're declared in their manifest. However, if a group of resources must always be managed in a specific order, you should explicitly declarerelationships with relationship metaparameters, chaining arrows, and the require function.

- before:Appliesaresourcebeforethetargetresource.
- require Applies a resource afterthetar resource.
- notify: a resource before the target resource.The target resource refreshes if the notifying resource changes.
- subscribe aresourceafterthetargetresource.Thesubscribing refreshesifthe targetresource

Chaining Arrows

You can create relationships between two resources groups of resources using the -> and ~> operators.

- -> ordering arrow: Applies the resource on the left before the resource on theright.
- ~> notifying arrow: Applies the resource on the left first. If the left-hand resource changes,the right-hand resource will refresh.

Both chaining arrows have a reversed form (<- and <~).

Chaining Arrows: Operands

- The chaining arrows accept the following kinds of operands on either side of the arrow:
  - Resource references, including multi-resource references
  - Arrays of resource references
  - Resource declarations
  - Resource collectors

Ordering

All relationships cause Puppet or more other resources.

Bydefault,unrelated manifest file.If you declare

Refreshing and Notification

- Some resource types can be refreshed dependency is changed.
- Built-in resource types that can refreshed
  - service
  - mount
  - exec
- Sometimes package
  - Rules notification and refreshingare:
  - Receiving events
  - Sending
  - No-op

Refreshing and Notification

- Certain resource types can have automatic relationships with other resources, using autorequire, autonotify, autobefore, orautosubscribe.
- A complete list can be found in the resource typereference.
- Auto relationships between types and resources are established when applying a catalog.

Missing Dependencies

- If one of the resources in a relationship is not declared the catalog will fail tocompile.
  - Could not find dependency <OTHER RESOURCE> for<RESOURCE>
  - Could not find resource '<OTHER RESOURCE>' for relationship on'<RESOURCE>'.

Failed Dependencies

- If a resource with dependencies fails to be applied, all dependent resource will be skipped.
  - notice: <RESOURCE>: Dependency RESOURCE> has failures: true
  - warning: <RESOURCE>:

Dependency Cycles

- If two or more be applied because this causes a
  - err: Could apply complete cycle: (<RESOURCE> => <OTHER RESOURCE> =><RESOURCE>)
  - Try the`--graph`option and opening the resulting`.dot`file in OmniGraffle or GraphViz

Conditional Statements

Conditionalstatementsletyour Puppetcodebehave ferently indifferentsituations. aremosthelpful when combined facts or with data retrieved external source.

- Conditionals alter logic:
  - if statement
  - unless
  - case statement
- Conditionals that return a value:
  - selector

"If" Statements

"If" statements take a boolean condition and an arbitrary block of Puppet code, and will only execute the block if the condition is true. They can optionally include elsif and else clauses.

Syntax:

```
if condition
  { block
  ofcode
}
elsif
  condition {
  block of code
}
else {
  default option
}
```

Example:

```
if$facts['os']['name']=='Windo
  ws'{ includerole::windows
}
elsif ($facts['os']['name'] == 'RedHat') and
($facts['os']['name'] == 'CentOS')       {
  includerole::redhat
}
elsif$facts['os']['name']=~/^(Debian|Ubun
  tu)$/{ includerole::debian
}
else {
 include::generic:
 :os
}
```

- Behavior
  - The if statement behaves like statements in any otherlanguage.
  - If none the conditions match and there no else block, Puppet will donothing.
- Conditions
  - Variables
  - Expressions, arbitrarily nested and and or expressions
  - Functions that return values
- Regex capture variables

- If you use a regular expression match operator as your condition, any captures from parentheses in the pattern will be available inside the associated code block as numbered variables ($1, $2, etc.), and the entire match will be available as $0:

Example:

```
if$trusted['certname'] =~
  /^www(\d+)\./ { notice("This
  is web servernumber $1.")
}
```

"Unless" statements

"Unless" is the reversed "if" statements. It takes a boolean condition and an arbitrary block of Puppet code. It will only execute the block of code if the condition is false. There cannot be a elsif clauses.

Syntax:

```
unless condition
   { block of code
}
```

Example:

```
unless  $facts['memory']['system']['totalbytes']  >  1073741824 {
   $maxclient   =  500
}
```

- Behavior
  - The condition is evaluated first and,ifit   false, the code block is executed.
  - If the condition is true, Puppet will do nothing.
  - The   statement is also an expression that produces a value, and can   used wherever a value is allowed.
- Conditions
  - Variables
  - Expressions, including arbitrarily nested and and or expressions
  - Functions   return values
- Regex capture variables
  - Although "unless" statements   variables like "if" statements, they usually aren't used.

Case Statements

Similar to the "if" statements, case statements choose one of several blocks of arbitrary Puppet code.

Syntax:

```
case condition {
    'control expression': { block of
    code } default: { block of code }
  }
```

Example:

```
  case $facts['os']['name'] {
    'Windows':
                          {includerole::window
    s} 'RedHat','CentOS':
                          {includerole::redhat
    }
    /^(Debian|Ubuntu)$/:{includerole::de
    bian} default:
                          {include::generic::o
    s}
  }
```

- Behavior
  - Compares defined.
  - The
  - The code block
  - Amaximum one code
  - If none the cases match, Puppet
- Conditions
  - Variables
  - Expressions, including arbitrarily nested and or expressions
  - Functions that return values
- Case matching
  - Most == equality operator
  - Regular =~ matching operator
  - Data types =~ matching operator
  - Arrays are compared to the
  - Hashes compare each key/valuepair.

    - Default matches anything, and unless nested inside an array or hash, is always tested last, regardless of its position in the list.

- When used as a value

    - In addition to executing the code in a block, a case statement is also an expression thatproduces a value, and can be used wherever a value is allowed.

- The value of a case expression is the value of the last expression in the executed block, or undef if no block was executed.

- Regex capture variables

  - If you use a regular expression match operator as your condition, any captures from parentheses in the pattern will be available inside the associated code block as numbered variables ($1, $2, etc.), and the entire match will be available as $0:

Example:

```
case $trusted['certname'] {
  /www(\d+)/: { notice("This is web server number
  $1."); } default: { notice("Now for something
  completely different")}
}
```

Selectors

Selectorexpressions generally onlyuse selectors in variable assignments.

Syntax:

```
case condition {
  'control expression': { block of
  code } default: { block of code }
}
```

Example:

```
$role = $facts['os']['name'] ? {
  'Windows'
                        >'role::windo
  ws',
  /^(Debian|Ubuntu)$/
                        >'role::debia
  n', default
                        >'role::redhat'
  ,
}
```

- Behavior

  - The entire selector expression is value.

  - The control expression is compared to each of the cases in the order they aredefined.
  - The default case is evaluated last.
  - The value of the matching case is returned.
  - If no conditions match the catalog will fail to compile.
- Conditions

- Variables
- Expressions, including arbitrarily nested and and or expressions
- Functions that return values

- Case matching
  - You cannot use lists ofcases.
  - Most data types == equality operator
  - Regular expressions =~ matching operator
  - Data types =~ matching
  - Arrays are compared
  -
  - default tested last,

  regardless of its

- Regex capture variables
  - If you use regularexpression match your condition, any captures from

  parentheses in the pattern will be available inside the associated code block as numbered variables ($1, $2, etc.), and the entire match will be available as $0:

Example:

```
$role = $facts['os']['name'] ? {
  /^(Debian|Ubuntu)$/ >"Youarerunning${1}",
  default              >"Youarerunninganunknownoperatingsystem!",
}
```

VariablesandScope

- Variables storevalues so be accessedlater.
- Variables are actually constants andcan't
- Facts and built-in variables.
- Variable names are prefixed with a $ (dollarsign).
- They are assigned using the = (equal sign) assignmentoperator.
- Variable names caninclude:
  - Uppercase and lowercase letters

- Numbers
- Underscores

- Append a variable by using the + symbol
  - '$variable = ['a','b']'
  - '$variable += ['c']'
  - '$variable now equals ['a', 'b', 'c']'
- Assigning multiple variables
  - You canassign multiple hash.
  - Arrays
    -
    - Arrays
  - Hashes
    - Variables are listedin an the assignmentoperator.
    - The hash is on the right of the assignmentoperator.
    - Hash keys must match their corresponding variable name.

Variable Assignment Example:

```
$variable_name1  = "value"
```

Array Assignment Example:

```
[$a,  $b,  $c]= [1,2,3]        #$a=1,$b=2,$c=3
[$a,  [$b,  $c]]= [1,[2,3]]      #$a=1,$b=2,$c=3
[$a,  $b]  =[1,[2]]            # $a = 1, $b =   [2]
[$a,  [$b]]  =[1,[2]]          # $a = 1, $b =  2
```

Hash Assignment Example:

```
[$a,  $b]  =     >  10,  b >20}                          #  $a  =  10,  $b  =
[$a,  $c]  =     >5, b     >  10,      >  15, d   >  22}   # $a  =  5,  $c  = 15
```

Variable Interpolation

- Variable interpolation is when a variables is resolved in a double-quotedstrings.

- Inside the double-quoted strings the variable is referenced using a dollar sign with curly braces.

- `${var_name}`
- Single quotes will treat the variable as a literal.

Example:

```
$variable="${some_other_variable}isbeinginterpolationinhere."
```

Arrays and Hashes

- Arrays
  - Arrays are ordered lists of
  - There are functions functions like each.
- Hashes
  - Hashes
  - The entries maintained
  - Hashes are merged using the+

Array Example:

```
$array_variable = [ 'a', 'b', 'c']
```

Hash Example:

```
$hash_variable ={key1 >"value1",key2  > "value2"}
```

Scope

- Scope is of code that is partially isolated from other
- Topscope
  - Code that is outside any class definition, or node definition exists attopscope. Variables and defaults declared at top scope are availableeverywhere.
- Node scope
  - Code inside a node definition exists at node scope. Note that since only one node definitioncan match a given node, only one node scope can exist at a time.
- Local scopes

- Code inside a class definition, defined type, or lambda exists in a localscope.

- Variables and defaults declared in a local scope are only available in that scope and itschildren.

Metaparameters

- Metaparameters are attributes that all resource type, custom types and defined typeshave.
- AvailableMetaparameters
  - alias
  - audit
  - before
  - consume
  - export
  - loglevel
  - noop
  - notify
  - require
  - schedule
  - stage
  - subscribe
  - tag

Example:

```
file
  {'/etc/ssh/sshd_confi
  g': owner     >root,
  group >root,
  alias  >'sshdconfig',
}
service { 'sshd':
  subscribe >File['sshdconfig'],
}
```

IterationandLoops

- Iteration features are implemented as functions that accept blocks of code calledlambdas.
- List of iteration functions

- each: Repeat a block of code any number of times, using a collection of values to provide different parameters eachtime.
- slice: Repeat a block of code any number of times, using groups of values from a collection as parameters.
- filter: Use a block of code to transform some data structure by removing non-matching elements.
- map: Use a block of code to transform every value in some datastructure.
- reduce: Use a block of code tocreatea valueordatastructurebycombiningvaluesfromaprovideddatastructure.
- with: Evaluate a block scope. Doesn't iterate, but has a family resemblance

Example:

```
$values = ['a', 'b', 'c', 'd','e']
#functioncallwithlambda:
$values.each|String$value|{
  notice { "Value from a lambda code block: ${value}":  }
}
```

ClassParametersandDefaults

- Classes, defined types, and lambdascanall parameters.
- Which is for you to pass external data.

Syntax:

```
Class <CLASS NAME>(
  <DATA TYPE><PARAMETER NAME>,
  <DATA TYPE><PARAMETER NAME> = <VALUE>,
  #  .
) {
  #  .
}
```

Example:

```
class ntp (
  Boolean $service_manage = true,
  Boolean$autoupdate    =
  false,String
          $package_ensure='present
  ',
  #   -
){
```

```
  #   .
}
```

params.pp

- The main classes inherit from a <MODULE>::params class, which only sets variables.
- Using the params.pp pattern is nowdeprecated.
- Using a function or Hiera to your defaults data is now the recommended method.

Function Data Provider

- The function provider
- Thisfunction params.pp
- It takes
- Setdata_provider metadata.j son
- Puppet will try find therequested
- The <MODULE NAME>::data function can one of:
  - A Puppet language function, located at <MODULE ROOT>/functions/data.pp.
  - ARubyfunction(usingthemodernPuppet::FunctionsAPI),locatedat<MODULEROO T>/ lib/puppet/functions/<MODULENAME>/data.rb.

Example:

```
#ntp/metadata.j son
{
    -
  "data_provider": "function"
}
#
ntp/functions/data.pp
function  ntp::data() {
  $base_params = {
    'ntp::autoupdate'
                             >false,'
    ntp::service_name'
                             >'ntpd'
    ,
  }
  $os_params = case
    $facts['os']['family'] { 'AIX':{
      {'ntp::service_name'   >'xntpd'}
    }
    'Debian':{
      {'ntp::service_name'>'ntp'}
    }
    default: {
      {}
```

```
      }
    }
    # Merge the hashes and return a single hash.
    $base_params + $os_params
  }
  #ntp/manifests/ini
  t.pp class ntp (
    # default values are inntp/functions/data.pp
    $autoupdate,
    $service_name,
  ) {
     -
  }
```

PuppetFunctions

There are

Functions

- Statements
  - They do returnarguments.
- Rvalues
  - They return values.
  - They only be used in a statement a value.
  - variable assignment
  - case
- Statement
  - alert ontheserveratlevelalert.
  - create_resources: ahashintoasetof andaddsthemtothecatalog.
  - err:Logamessageontheserveratlevelerr.
  - fail: Fail with a parse error.
  - hiera_include: Uses an array merge lookup to retrieve the classes array, so every node gets every class from thehierarchy.
  - include: Declares one or more classes, causing the resources in them to be evaluated and addedtothecatalog.

- warning:Logamessageontheserveratlevelwarning.

- Rvalue Functions
  - defined: Determines whether a given class or resource type is defined and returns a Boolean value.
  - file: Loads a file from a module and returns its contents as a string.
  - generate: Calls an external command on the Puppet master and returns the results of the command.
  - hiera: Performs a standard priority lookup of the hierarchy and returns the most specific value for a given key.
  - hiera_array: Findsallmatches thehierarchyandreturnsthemasasingleflattenedarrayofunique
  - hiera_hash returnsthemina merged
  - regsubst
  - sha1:Returns
  - template LoadsanERB evaluatesit,andreturnstheresultingvalueasastring.
- Review the [Puppet Function list].

[Puppet Function (https://docs.puppet.com/puppet/latest/function.html)

Templates

- template Loads an ERB template from a module, evaluates it,andreturns resulting value as a string.
- Atemplate by template(<MODULE NAME>/<TEMPLATE FILE>)
  - template('modulename/motd.erb')
- The file is located in <MODULES DIRECTORY>/<MODULE NAME>/templates/motd.erb

Example:

```
file {
  '/etc/motd':
  ensure   >file,
  content  >template('modulename/motd.erb')
}
```

Embedded Ruby (ERB) Template Syntax

- ERB is a templating language based onRuby.
- Puppetusesthetemplateandinline_templatefunctionstoevaluateatemplatefile.

Expression-printing:

```
<%=  @value  %>
```

If statement:

```
<% ifcondition%>  .text  .<% end %>
```

Comments:

```
<%# This is a co ment.%>
```

Looping:

```
<%  @valuse.each  -%>
<% do |values| %>some value <%= value %>
<%  end -%>
```

DefinedResourceTypes

- Defined types alsocalleddefined ordefines.
- Are blocks code that can be evaluated multiple times with differentparameters.
- They act resource type.
- They are resource type.
- Definitions should be stored themanifests/directory.
- Defined type instance caninclude
- Defined type names can consist of one or more namespacesegments.
- Each namespace segment must begin with a lowercase letter and caninclude:
  - Lowercase letters
  - Digits

- Underscores

- Namespace segments should match the following regular expression:
  - \\A[a-z]\[a-z0-9_\]*\\Z
  - define_name123
- Multiple namespace segments can be joined together in a define type name with the ::(double colon) namespaceseparator.
  - \\A(\[a-z\][a-z0-9_]*)?(::[a-z]\[a-z0-9_]*)*\\Z
  - module_name::defined_type_name

Syntax:

```
define name (
 <DATA TYPE><PARAMETER> = <VALUE>,
) {
    .puppetcode   .
}
```

Declaring an Instance:

```
<DEFINED TYPE>{'<TITLE>':
   <ATTRIBUTE> ><VALUE>,
}
```

Example:

```
define apache::vhost
  ( Integer $port,
  String[1]$docroot,
  String$servername=$titl
  e,
  String[1]$vhost_name='*
  ',
) {
  #   .
}
apache::vhost
  {'mywebsite': port
          >80,
  docroot >'/var/www /mywebsite',
}
```

ResourceCollectors

- Resource collectors also called the spaceshipoperator.
- It selects a group of resources by searching the attributes of every resource in the catalog.
- This search is independent ofevaluation-order.

- Collectors realize virtual resources.
- Can be used in chaining statements
- Can override resource attributes.
- Can function as both a statement and a value.
- The resource type, capitalized.

Operators

- ==
- !=
- and
- or

Syntax:

```
<RESOURCE TYPE><| <SEARCH EXPRESSION>  |>
```

Example:

```
User <| groups == 'admin'  |>
```

ExportedResources

- Exported resources require catalog storage and searching to be enabled on Puppetmaster.
- Formerly "storeconfigs".
- Both the and the searching (among other features) are PuppetDB.
- Exported resource declaration specifies a desired state fora
- It does not manage the resource on
- Publishes the resource for use by other nodes.
- Any node can then collect the exported resource and manage its own copy of it.

Purpose

- Exported resources allow the Puppet compiler to share information among nodes by combining information from multiple nodes' catalogs.

- This helps you manage things that rely on nodes knowing the states or activity of other nodes.

Syntax:

```
class <CLASS NAME>{
   #Declare:
   @@<RESOURCE  BEING  EXPORTED>{ <TITLE>:
      <ATTRIBUTE> ><VALUE>,
   }
   #Collect:
   <REFERENCE RESOURCE BEING EXPORTED><<| |>>
}
```

Example:

```
class ssh {
   #Declare:
   @@sshkey{$::hostname
      :  type        >dsa,
      key     >$::sshdsakey,
   }
   # Collect:
   Sshkey <<|
   |>>
}
```

Declaring an Exported Resource

- To declare exported resource, prepend @@ (a double "at" sign) to the resource type of a standard resource declaration:

Syntax:

```
      @@<RESOUCE TYPE>{ <TITLE>:
<ATTRIBUTE> ><VALUE>,
      }
```

NTPModule

ntp.conf.erb

```
# File  Managed  by Puppet
#Formoreinformationaboutthisfile,seethemanpages
#ntp.conf(5),ntp_acc(5),ntp_auth(5),ntp_clock(5),ntp_misc(5),ntp_
```

```
mon(5).
driftfile /var/lib/ntp/drift
#Permittimesynchronizationwithourtimesource,butdonot
#permitthesourcetoqueryormodifytheserviceonthissystem.
restrictdefaultnomodifynotrapnopeernoquery
```

```
#Permitallaccessovertheloopbackinterface.  Thiscould
#betightenedaswell,buttodosowouldeffectsomeof
# the administrative
functions. restrict  127.0.0.1
restrict ::1
# Hosts on local network are less   restricted.
#restrict192.168.1.0mask255.255.255.0nomodifynotrap
#Usepublicserversfromthepool.ntp.orgproject.
#Pleaseconsiderjoiningthepool(http://www.pool.ntp.org/join.htm
l).
<% @servers.each do |server| -%>
<%= server %>
<% end -%>
#broadcast192.168.1.255autokey # broadcast server
#broadcastclient          # broadcastclient
#broadcast224.0.1.1autokey       #multicastserver
#multicastclient224.0.1.1  #multicastclient
#manycastserver 239.255.254.254 # manycastserver
#manycastclient 239.255.254.254 autokey # manycast client
# Enable public key cryptography.
#crypto
includefile/etc/ntp/crypto/pw
#Keyfilecontainingthekeysandkeyidentifiersusedwhenoperating
#withsymetrickeycryptography.
keys /etc/ntp/keys
#Specifythekeyidentifierswhicharetrusted.
#trustedkey 4 8 42
#Specifythekeyidentifiertousewiththentpdcutility.
#requestkey 8
#Specifythekeyidentifiertousewiththentpqutility.
#controlkey 8
#Enablewritingofstatisticsrecords.
#statistics clockstats cryptostats loopstatspeerstats
#Disablethemonitoringfacilitytopreventamplificationattacksusi
ng ntpdc
#monlistcomandwhendefaultrestrictdoesnotincludethenoquery
flag. See
#CVE-2013-5211formoredetails.
#Note:Monitoringwillnotbedisabledwiththelimitedrestrict
ion flag.
disablemonitor
```

ExportedResources

[puppet-sshkeys]

Roles andProfiles

Overview

Therolesandprofilesareusedtobuildreliable, reusable, configurable,

andrefactorablesystemconfigurations. They are two extra layers of indirection between your node

classifier and your component modules.

- **(Component modules:** Normal modules that manage one particular technology. (Forexample, puppetlabs/apache.)
- **Profiles:**Wrapperclassesthatusemultiplecomponentmodulestoconfigurealayeredtechnology stack.
- **Roles:**Wrapperclassesthatusemultipleprofilestobuildacompletesystemconfiguration.

Profiles

- A profile is justa
- Make declarations on them.
- Profiles caninclude
- Profilesownall class parameters
- Components class shouldn't use a value data.
- There are ways a profile can get the data it needs to configure componentclasses:
  - Hardcode it in theprofile.
  - Look from Hiera.

Example:

```
classprofiles::apache(
  String$apache_vhost_na
  me,
  String$apache_vhost_docroot,
  Bolean $apache_default_vhost =
  false, String$apache_vhost_port=80,
) {
  class { 'apache':
    default_vhost  >$apache_default_vhost,
  }
  apache::vhost{$apache_vhost_nam
    e:  port  >$apache_vhost_port,
    docroot
             >$apache_vhost_docro
```

```
          ot,
        }
      }
```

Roles

- The only thing roles should do is declare profileclasses.

- Use `include<PROFILENAME>`.
- Don't declare any component classes or normal resources in a role.
- Roles can use conditional logic to decide which profiles touse.
- Roles should not have any class parameters of their own.
- Roles should not set class parameters for anyprofiles.
- The name of a role should be based on your business's conversational name for the type of node it manages.
- Assigning a role to a node
    - The PE console node
    - The main
    - Hiera

Roles Names Example:

```
role::web
role::jenkins::master
role::jenkins::slave
```

Example:

```
class role::web {
  includeprofile::bas
  e
  includeprofile::apa
  che
  includeprofile::ph
  p
}
```

HieraOverview

- Hiera is a key/value datastore for looking up data.
- Let you set node-specific datawithout

Why use Hiera?

- Single source of truth for your data.

    - Configure default data with hierarchaloverrides.
- Use Puppet modules from theforge.
    - No need to edit the module, just put the data in Hiera.

- Publish your own modules for collaboration.
    - Keeps your data out of your module before sharing it.
    - No more clashing variable names.

Setting Up Hiera

- The `hiera.yaml` file is located in `/etc/puppetlabs/puppet/`.
- `:backends:` tells Hiera what kind of data sources it should process. In this case, we'll be using YAMLfiles.
- `:yaml:` configures theYAML
- `:datadir:` tells
- `:hierarchy:`
    - Separate
    - Morespesific
    - Least spesific at thebottom.
- You can use facts in your Hieralookups.

hiera.yaml

```
---
:backends:
  -yaml
:yaml:
  :datadir:"/etc/puppetlabs/code/environments/%{environment}
  /hieradata"
:hierarchy:
  - "nodes/%{::trusted.certname}"
  - comon
```

Automatic Parameter Lookup

- Process of automatic parameter
- Look for parameters passed using the class {} declaration
    - If no pass parameter it will look in hiera data source for the parameter <CLASS NAMESPACE>::parameter
    - If not found in hiera data source it will use the default set "default"

Hiera Lookup Functions

hiera:

Performsastandardprioritylookupofthehierarchyandreturnsthemostspecificvalueforagivenkey.The returned value can be any type of data.

Arguments:

- A string key that Hiera searches for in the hierarchy.Required.
- An optional default value to return if Hiera find anything matching thekey.
- The optional name of an arbitrary top of thehierarchy.

hiera_array:

Findsall ofunique values. Ifanyof This iscalled an array merge lookup.

Arguments:

- Astring key Hiera searches for in the Required.
- An optional default value to return if Hiera doesn't find anything matching thekey.
- The optional name of an arbitrary hierarchy level to insert the top of thehierarchy.

hiera_hash:

Finds all matches a key throughout the hierarchy and returns them in a merged hash. If any of the matchedhashes keys, the final hashusesthe from thehighestpriority This iscalleda hash mergelookup.

Arguments:

- A string key that Hiera searches for in the hierarchy.Required.
- An optional default value to if Hieradoesn'tfind matching thekey.
- The optional name of an arbitrary hierarchy level to insert at the top of thehierarchy.

Overview

- Code Manager and r10k are used to manage and deploying your Puppet code.

- Install Puppet modules.
- Create and maintain environments.
- Deploy new code to your masters.
- Keep your module code in Git.

- Code Manager automates the management and deployment of your new Puppet code.
    - Push your code updates to your Gitrepository.
    - Puppet creates environmentsbasedoff branch.
    - Installs modules.
    - Deploys and
    - All
- You canr10k
    - Youshould
    - Code Manager works withr10k.
- Both tool are built into Puppet Enterprise.
- Create a repository for maintaining your environments and code.
- Set up Puppetfiles, if you want to install modules in yourenvironments.
- Configure Code Manager(recommended)
- Existing environments will not preserved.
- /etc/puppetlabs/code/environments/production will be overwritten.

Set Up and Configuring CodeManager

- Create your own control repo.

```
wgethttps://github.com/puppetlabs/control-
repo/archive/production.zip yum installunzip-y
unzip
production.zip
cdproduction
```

- Create a control repo in GitHub.
    - Log in to your Github account.

- Click Repositories.

- Click the New button.
- Enter puppet-control for the Repository name.
- Click CreateRepository.
- Initialize your the control repo.
  - Check in code.
  - Add remote repo.
  - Push code.

```
git init
git remote add origin
<URL_TO_REPOSITORY>git  co  mit  -am
"first  co mit"
git push origin master
```

- Create an

```
mkdir-p/etc/puppetlabs/puppetserver/ssh
ssh-keygen -trsa -b 4096 -C "your_email@example.com"
```

- Enter the path to where the rsa key will go.
  - /etc/puppetlabs/puppetserver/ssh/id_rsa
- Press enter   an empty passphrase
- Make sure   is owned by pe-puppet

```
chmod  -R  pe-puppet:pe-puppet  /etc/puppetlabs/puppetserver/ssh
```

- Update PE   node group.
- Add the   parameters to thepuppet_enterprise::profile::master
  - code_manager_auto_configure totrue
  - update r10k_remotewith
  - updater10k_private_keywiththepathtoyourrsakey

```
/etc/puppetlabs/puppetserver/ssh/id_rsa
```

- Executeapuppetagent–tonthePuppetmasterserver.

- View code managerconfiguration.

```
r10k deploy display  --fetch
```

- Create a deployuser.
- Reset the password for your deployuser.
- Add the deploy user to the Code Deployers User role.
- Create a token for your deployuser.

```
puppet-accesslogin--service-
urlhttps://<HOSTNAMEOFPUPPETENTERPRISECONSOLE>:4433/rbac-
api--lifetime180d.
```

- Deploying your code to themaster.

```
puppet-code deploy --all--wait
```

Git URL Example:

```
git@<YOUR.GIT.SERVER.COM>:puppet/control.git
```

RSA Key Example:

```
"/etc/puppetlabs/puppetserver/ssh/id-control_repo.rsa"
```

NginxModule

nginx.conf.erb

```
      # File Managed by
Puppet user  <%=
@process_user  %>;
worker_processes
<%=@processorcount%>;error_log
               <%= @log_dir
%>/error.log;pid     <%=@pid_file%>;
events  {
   worker_connections
   1024;
}
http {
   server_tokens off;
   include
                    <%=@config_dir%>/mime.types
   ; default_type
```

```
                application/octet-stream;
      access_log
                <%=@log_dir%>/access.log;se
      ndfile    on;
      #tcp_nopushon;
      tcp_nodelay
                 on;
      include<%=@confd%>/*.conf;
    <% if@vdir_enable %>
      include<%=@vdir_enable%>/*;
    <%  end %>
    }
```

vhost.conf.erb

```
    # File Managed by Puppet
    server {
      listen<%=@port%>;
      root   <%= @vhost_docroot%>;
      server_name <%= @name %><%= @serveraliases
      %>; access_log
               <%=@log_dir%>/<%=@name%>.access.log;
      error_log <%=@log_dir%>/<%=@name%>.error.log;
    }
```

NodeClassification

Node Definition Lookup

Node Definition

Attempt to match

webserver01.mylabserver Attempt to

webserver01.mylabserver Attempt to

webserver01

Match Default

No Match (if no default)

Note: if a node multiple node definitions regular expressions, puppet use ONE of them with no guarantee which one it will use.

External Node Classifiers

- ENCs can standard node definitions in site.pp, and declared in each source are effectivelymer
- node_terminus:TellsPuppet using.
  - Default node_terminus=classifier
- external_nodes:ThisisthepathtotheexecutableoftheENC
- Replace node_terminus=console withnode_terminus=exec.

Example:

```
[master]
  node_terminus  = exec


  external_nodes=/usr/local/bin/puppet_node_classifier
```

Using Hiera as an ENCs

- hiera_include: Assigns classes to a node using an array merge lookup that retrieves the value for a user-specified key from Hiera'sdata.
- You can use Hiera as an ENCby:
  - Use your default node insites.pp
  - Add hiera_include('classes'))
  - Define classes inyour

Example:

```
# Assuming apache.yaml:
classes:
  - role::apache
# Assuming co mon.yaml:
classes:
   - role::base
```

External Node Classifiers (ENCs) & Site.pp Merging

- A Puppet catalog is made upof:
  - ENCs with thesite.ppby mer node objects
  - All specified in the node object defined insite.ppORnode_terminus executable

- Any resources which are in the site manifest but outside definitions

Puppet OrchestratorOverview

Overview

- The Puppet orchestrator is a set of interactive command line tools that give you the ability to control the rollout of configuration changes when and how you wantthem.
- Tools:
  - puppetjob
    - Allows you to manage and enforce the order if Puppet agent runs across an environment.

    - Enforces the order of agent runs by instantiating an application model and assigning nodes to application components.
  - puppet app
    - Lets you view the application models and application instances written and stored on the Puppetmaster.
    - Lets you see what is available to include in an orchestration run.
- You control when Puppet runs and where node catalogs areapplied.
- You no longer need to waiton arbitrary update your nodes.

Orchestrator Workflow

- WritePuppet
- puppet parservalidate
- puppet app show application instances looks correct.
- puppet job plan commandto applicationinstancesandthenoderunorderthatwouldbeincludedinajob.
- puppet job run command to enforce change on your infrastructure and configureyour application.
  - The with the--noop
- puppet job show commandtoreview abouttherun.

MCollectiveOverview

Overview

- Puppet Enterprise includes MCollective.
- Which is used to invoke actions in multiple nodes.
- You can write custom plugins to add newactions.
- MCollective is built around the idea of predefinedactions.
- It is essentially a highly parallel remote procedure call (RPC) system.
- Actions are distributed in plugins

MCollective Plugins:

- package:Installanduninstallsoftwarepackages.
- puppet:RunPuppetagent,getitsstatus,andenable/disableit.
- puppetral:ViewresourceswithPuppet'sresourceabstractionlayer.
- rpcutil: General helpful actions that expose stats and internals to SimpleRPC clients.
- service: Start and stop systemservices.

MCollective Components:

- pe-activemq: Service (whichrunson master server) routes all MCollective-related messages.
- pe-mcollective: for authorized commands and invokes actions in
- mco ssue authorized commands

UsingMCollective

- To run MCollective commands youmust:
  - Be logged in to the Puppet masterserver.
  - Use the peadmin user account.
  - By the peadmin account cannot with a password.

Using sudo

```
sudo –i -u peadmin
```

Adding SSH keys

- You can have other users to runcommands.
- Add the user's public SSH keys to peadmin's authorized keysfile.
- /var/lib/peadmin/.ssh/authorized_keys

The mco command

- All MCollective actions are invoked with the mcocommand.
- The mco command relies on a configfile.

- /var/lib/peadmin/.mcollective
- It is only readable by the peadmin use.

Using mco help

```
mcohelp
mco help <SUBCOMMAND>
mco <SUBCOMMAND>--
help
```

Synstax:

```
mco <SUBCOMMAND><ACTION>
mco rpc <AGENT PLUGIN><ACTION><INPUT =<VALUE>
```

Examples:

```
mcoping
mcorpcrpcutilping
mco rpc service restart service=puppet
```

Host Filters

- -W, --withFILTER Combined classes and facts filter
- -S,--select FILTER Compound filter combining facts andclasses
- -F, --wf--with-factfact=valMatch hosts with a certain fact
- -C, --wc--with-classCLASS Match with a certain config management class
- -A, --wa--with-agentAGENT Match hosts with a certain agent
- -I, --wi--with-identityIDENTMatch hosts with a certain configured identity

Troubleshooting

Common Installer Problems

- Check yourDNS
- Puppet communicates on ports 8140, 61613, and 443.
- If you are installing the console and the Puppet master on separate servers and tried to install the console first, the installer mayfail.

- Recovering from a failed install.

- If you encounter errors during installation, you can fix them and run the installeragain.

Troubleshooting Connections

- Troubleshooting connections betweencomponents
  - Is the agent able to reach the Puppet master?
  - Try 'telnet <puppet master's hostname>8140'
  - Make sure the agent can reach the DNS name that is configured inpuppet.conf.
  - Check that the pe-puppetserver service
- Make sure the agent has
- Check the logs
  -
- Revoke thecertificate
  - On the master:
    - `puppet cert clean <NODENAME>`
  - On the agent:
    - `rm-r$(puppetagent--configprintssldir)    puppetagent-t (or--test)`

Troubleshooting   filebucket:

If you get the following error during a Puppet run:

```
err:
/Stage[main]/Pe_mcollective/File[/etc/puppetlabs/mcollective
/ server.cfg]
/content:change from {md5}778087871f76ce08be02a672b1c48bdc
to{md5} e33a27e4b9a
87bb17a2bdff115c4b080 failed: Could not back up/etc/puppetlabs/
mcollective/se
rver.cfg: getaddrinfo: Name or service not known
```

Example:

```
#Definefilebucket'mai
n': filebucket{'main':
  server   > '<PUPPET MASTER'S DNS NAME>',
  path    >false,
}
```

General Troubleshooting

- Use--profiloraddprofiletotrueintheagent'spuppet.conffile.
- Use--logdestand--debugtologadditionaldetailstosyslog.

Database Troubleshooting

- Troubleshootclassification
    - You can cURL the console to troubleshoot the nodeclassifier.

Determine What Node Groups the NC Has and What Data They Contain:

```
curl https://$(hostname -f):4433/classifier-api/v1/groups
>classifier_ groups.json
      --cacert/etc/puppetlabs/puppet/ssl/certs/ca.pem
      --
      cert/etc/puppetlabs/puppet/ssl/certs/<WHITELISTEDCERTNAME
      >.pem
      --key /etc/puppetlabs/puppet/ssl/private_keys/<WHITELISTED
CERTNAME>.pem
```

Determine What Data the NC Will Generate for a Given Node Name:

```
curl https://$(hostname -f):4433/classifier-
api/v1/classified/
nodes/<SOMENODENAME>>node_classification.json
      --cacert/etc/puppetlabs/puppet/ssl/certs/ca.pem
      --
      cert/etc/puppetlabs/puppet/ssl/certs/<WHITELISTEDCERTNAME
      >.pem
      --key /etc/puppetlabs/puppet/ssl/private_keys/<WHITELISTED
CERTNAME>.pem
```

- PostgreSQL taking up too much space
    - PostgreSQL should have autovacuum=on set bydefault.
- PostgreSQL memory causes PE install to fail

Check /var/log/pe-postgresql/pgstartup.log

```
FATAL: could not create shared memory segment: No space left
ondeviceDETAIL:Failedsystemcallwasshmget(key=5432001,
size=34427584512,03600).
```

- Tweaking the machine's shmmax and shmall kernel settings before installingPE.
    - shmmax should equal 50% of the total RAM.
    - shmall should be calculated by dividing the new shmmax setting by thePAGE_SIZE.
    - Get the PAGE_SIZE by running getconfPAGE_SIZE.

To Set the New Kernel Settings by Run:

```
sysctl -w kernel.sh max=<your sh max
calculation>sysctl-
wkernel.shmall=<yourshmallcalculation>
```

Optimizing the Databases

- Changing PuppetDB's parameters.
  - PuppetDBparametersaresetinthejetty.ini.
  - jetty.iniismanagedbyPE.
  - You need toupdate the overwritten.
- Changing the
  - Onthedatabase psql execute
    - ALTER USER console PASSWORD '<newpassword>';
  - Edit /etc/puppetlabs/puppetdb/conf.d/database.ini and update password.
  - Start the pe-puppetdb service.

Vacuuming PostgreSQL

```
su - pe-postgres -s /bin/bash -c "vacuumdb -z --verbose <DATABASE
NAME>"
```

Backing Up PostgreSQL

```
sudo -u pe-postgres
/opt/puppetlabs/server/apps/postgresql/bin/pg_ dumpall-
c–f<BACKUP_FILE>.sql
```

Reporting

- Information found on reports:
  - Total: Total number of resources beingmanaged.
  - Skipped: How many resources were skipped (either due to tags or schedule metaparameter).
  - Scheduled: How many resources met the scheduling restriction, if one is present.

- Out of Sync: How many resources were out of sync (not in the desired configurationstate).
- Applied: How many resources were aelempted to be put into the desired configurationstate.

- Failed: How many resources were not successfully fixed (put into the desiredconfiguration state).
- Restarted: How many resources were restarted.
- Failed restarts: how many resources could not be restarted.
- Total time for configuration run (puppet agentexecution).
- How long it took to retrieve the configuration (compiled catalog) from the puppetmaster.

- Built in report processors
  - http: send reports to https/http.
  - log: Send logs to local
  - store: setting
- Report
  - tagmail: send

Puppet Enterprise Roles Based Access Control

RBAC Permissions

RemovingNodes

- You will to do the following step to a node from Puppet Enterprise:
  - Deactivates the node in PuppetDB.
  - Deletes Puppet master's information cache for the node.
  - Frees that the node was using.
  - Allows you to re-use hostname for a new node.

On the Agent Node:

```
service  puppet stop
```

On the Puppet Master:

```
puppet node purge
<CERTNAME>puppet agent-t
service pe-puppetserver restart
```

- If the deactivated node still shows up, stop MCollective.

On the Agent Node:

```
servicemcollectivestop
/etc/puppetlabs/mcollective/ssl/clients.
```

Checking Values ofSettings

- puppetmaster--configprint<CONFIGNAME>
- puppetconfigprint<CONFIGNAME>
- puppetconfigprint<CONFIGNAME>--section<SECTIONNAME>

Puppet ResourceCommand

- puppet resource <RESOURCENAME>
- puppet resource <RESOURCENAME>