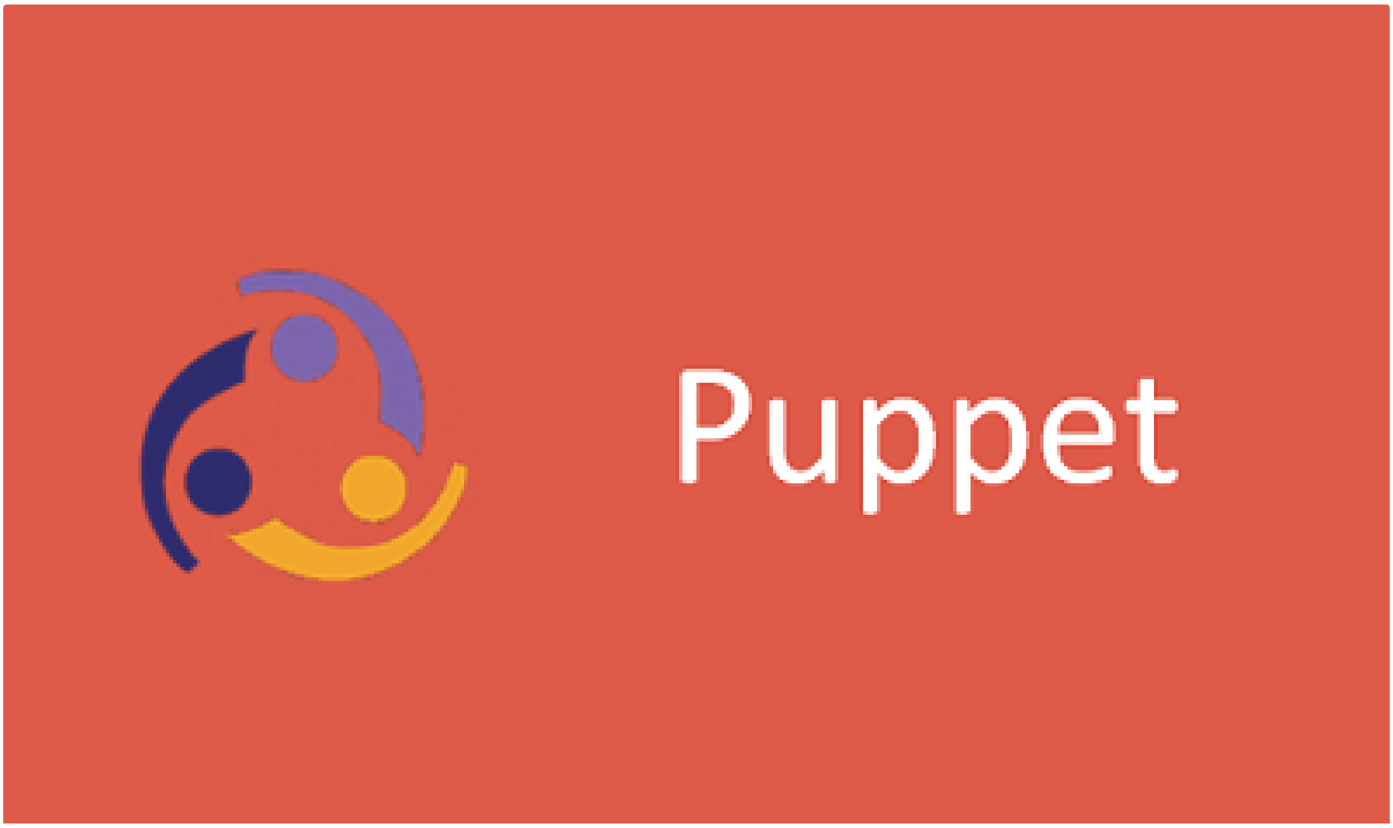
TekSlate
Watch Learn & Excel

- **Interview Questions** **Article**

Puppet Interview Questions and Answers

By Anil March 9, 2016 Total Views: 16,045



Puppet Interview Questions

1. What is Puppet ?

Puppet is a configuration Tool which is used to automate administration tasks. Puppet Agent(Client) sends a request to Puppet Master (Server) and Puppet Master Push Configuration on Agent.

2. What is Manifests ?

Manifests, in Puppet, are the files in which the client configuration is specified.

3. What is Module and How it is different from Manifest ?

Whatever the manifests we defined in modules, can call or include into other manifests. Which makes easier management of Manifests.It helps you to push specific manifests on specific Node or Agent.

4. What are the Commands to check requests of Certificates ?

puppetca –list (2.6)
puppet ca list (3.0)

5. What are the Commands to sign Requested Certificates ?

puppetca –sign hostname-of-agent (2.6)
puppet ca sign hostname-of-agent (3.0)

6. Where Puppet Master Stores Certificates

/var/lib/puppet/**ssl**/ca/signed

7. What is Facter ?

Sometimes you need to write manifests on conditional expression based on agent specific data which is available through Facter. Facter provides information like Kernel version, Dist release, IP Address, CPU info, etc. You can defined your facter also.

8. What is the use of etckeeper-commit-post and etckeeper-commit-pre on Puppet Agent ?

etckeeper-commit-post: In this configuration file you can define command and scripts which execute after pushing configuration on Agente

etckeeper-commit-pre: In this configuration file you can define command and scripts which execute before pushing configuration on Agent

9. What is Puppet Kick ?

By default *Puppet Agent* request to Puppet Master after a periodic time which was known as "runinterval". Puppet Kick is a utility which allows you to trigger Puppet Agent from Puppet runinterval. Puppet Kick is a utility which allows you to trigger Puppet Agent from Puppet Master.

10. What is MCollective ?

MCollective is a powerful orchestration framework. Run actions on thousands of servers simultaneously, using existing plugins or writing your own.

Puppet DevOps Interview Questions

11. What's special about Puppet's model-driven design?

Traditionally, managing the configurations of a large group of computers has meant a series of imperative steps; in its rawest state, SSH and a *for* a loop. This general approach grew more sophisticated over time, but it retained the more profound limitations at its root.

Puppet takes a different approach, which is to model everything — the current state of the node, the desired configuration state, the actions taken during configuration enforcement — as data: each node receives a catalog of resources and relationships, compares it to the current system state, and makes changes as needed to bring the system into compliance.

The benefits go far beyond just healing the headaches of configuration drift and unknown system state: modeling systems as data lets Puppet simulate configuration changes, track the history of a system over its lifecycle, and prove that refactored manifest code still produces the same system state. It also drastically lowers the barrier to entry for hacking and extending Puppet: instead of analyzing code and reverse-engineering the effects of each step, a user can just parse data, and sysadmins have been able to add significant value to their Puppet deployments with an afternoon's worth of perl scripting.

12. Why does Puppet have its language? Why not use XML or YAML as the configuration format? Why not use Ruby as the input language?

The language used for manifests is ultimately Puppet's human interface, and XML and YAML, being data formats developed around the processing capabilities of computers, are horrible human interfaces. While some people are comfortable reading and writing them, there's a reason why we use web browsers instead of just reading the HTML directly. Also, using XML or YAML would limit any assurance that the interface was declarative — one process might treat an XML configuration differently from another.

13. Can Puppet manage workstations?

Yes, Puppet can manage any machine and is used to manage many organizations that have a mix of laptops and desktops.

14. Does Puppet run on Windows?

14. Does Puppet run on Windows?

Yes. As of Puppet 2.7.6 basic types and providers do run on Windows, and the test suite is being run on Windows to ensure future compatibility. More information can be found on the Puppet on Windows page, and bug reports and patches are welcome.

15. What size organizations should use Puppet?

There is no minimum or maximum organization size that can *benefit from Puppet,* but there are sizes that are more likely to benefit. Organizations with only a handful of servers are unlikely to consider maintaining those servers to be a real problem, while those that have more need to consider carefully how they eliminate manual management tasks.

16. My servers are all unique; can Puppet still help?

Yes.

All servers are at least somewhat unique, but very few servers are unique; host names and IP addresses (e.g.) will always differ, but nearly every server runs a relatively standard operating system. Servers are also often very similar to other servers within a single organization — all Solaris servers might have similar security settings, or all web servers might have roughly equivalent configurations — even if they're very different from servers in other organizations. Finally, servers are often needlessly unique, in that they have been built and managed manually with no attempt at retaining appropriate consistency.

Puppet can help both on the side of consistency and uniqueness. Puppet can be used to express the consistency that should exist, even if that consistency spans arbitrary sets of servers based on any data like operating system, data centre, or physical location. Puppet can also be used to handle uniqueness, either by allowing the special provision of what makes a given host unique or through specifying exceptions to otherwise standard classes.

17. Who is Puppet Labs?

Puppet Labs (formerly Reductive Labs) is a small, private company focused on re-framing the server automation problem.

18. How should I upgrade Puppet and Facter?

The best way to install and upgrade Puppet and Facter is via your operating system's package management system, using either your vendor's repository or one of Puppet Labs' public repositories.

If you have installed Puppet from source, make sure you remove old versions entirely (including all application and library files) before upgrading. Configuration data (usually located in/etc/puppet or /var/lib/puppet, although the location can vary) can be left in place between installs.

19. What characters are permitted in a class name? In a module name? In other identifiers?

Class names can contain lowercase letters, numbers, and underscores, and should begin with a lowercase letter. "::" can be used as a namespace separator.

The same rules should be used when naming defined resource types, modules, and parameters, although modules and parameters cannot use the namespace separator.

Variable names can include alphanumeric characters and underscore, and are case-sensitive.

Puppet Interview Questions And Answers For Experienced

20. How do I document my manifests?

The puppet language includes a simple documentation syntax, which is currently documented on the Puppet Manifest Documentation wiki page. The puppet doc command uses this inline documentation to automatically generate RDoc or HTML documents for your manifests and modules.

21. How do I manage passwords on Red Hat Enterprise Linux, CentOS, and Fedora Core?

As described in the Type reference, you need the Shadow Password Library, which is provided by the ruby-shadow package. The ruby-shadow library is available natively for fc6 (and higher) and should build on the corresponding RHEL and CentOS variants.

22. How do all of these variables, like $operatingsystem, get set?

The variables are all set by Facter. You can get a full listing of the available variables and their values by running facter by itself in a shell.

# facter

23. Can I access environment variables with Facter?

Not directly. However, Facter reads in custom facts from a special subset of environment variables. Any environment variable with a prefix of `FACTER_` will be converted into a fact when Facter runs. For example:

```
$ FACTER_FOO="bar"
$ export FACTER_FOO
$ facter | grep 'foo'
  foo => bar
```

The value of the `FACTER_FOO` environment variable would now be available in your Puppet manifests as $foo, and would have a value of 'bar'. Using shell scripting to export an arbitrary subset of environment variables as facts is left as an exercise for the reader.

24. Why shouldn’t I use autosign for all my clients?

It is very tempting to enable autosign for all nodes, as it cuts down on the manual steps required to bootstrap a new node (or indeed to move it to a new puppet master).

Typically this would be done with a *.example.com or even * in the autosign.conf file.

This, however, can be very dangerous as it can enable a node to masquerade as another node and get the configuration intended for that node. The reason for this is that the node chooses the certificate common name (‘CN’ – usually its fqdn, but this is fully configurable), and the puppet master then uses this CN to look up the node definition to serve. The certificate itself is stored, so two nodes could not connect with the same CN (e.g. alice.example.com), but this is not the problem.

The problem lies in the fact that the puppet master does not make a 1-1 mapping between a node and the first certificate it saw for it, and hence multiple certificates can map to the same node, for example:

- alice.example.com connects, gets node alice { } definition.
- bob.example.com connects with CN alice.bob.example.com, and also matches the node alice { } definition.

Without auto signing, it would be apparent that bob was trying to get alice’s configuration – as the puppet cert process lists the full fqdn/CN presented. With autosign turned on, bob silently retrieves alice’s configuration.

Depending on your environment, this may not present a significant risk. It essentially boils down to the question ‘Do I trust everything that can connect to my puppet master?’.

If you do still choose to have a permanent, or semi-permanent, permissive autosign.conf, please consider doing the following:

- Firewall your puppet master – restrict port tcp/8140 to only networks that you trust.
- Create puppet masters for each ‘trust zone’, and only include the trusted nodes in that Puppet masters manifest.
- Never use a full wildcard such as *.

Puppet Master Interview Questions

25. What happens if I am on Puppet 2.6x or earlier?

Nothing changes for you. Puppet 2.6.x remains licensed as GPLv2. The license change is not retroactive.

26. Does this change affect all the components of Puppet?

As part of this change, we’re also changing the license of the Facter system inventory tool to

Apache. This change will take effect with Facter version 1.6.0, and earlier versions of Facter will remain licensed under the GPLv2 license. This change will bring the licensing of Puppet's two key components into alignment.

Our other major product, MCollective, is already licensed under the Apache 2.0 license.

27. What does this mean if I or my company have or want to contribute code to Puppet?

As part of this license change, Puppet Labs has approached every existing contributor to the project and asked them to sign a Contributor License Agreement or CLA.

Signing this CLA for yourself or your company provides both you and Puppet Labs with additional legal protections and confirms:

1. That you own and are entitled to the code you are contributing to Puppet
2. That you are willing to have it used in distributions

This gives assurance that the origins and ownership of the code cannot be disputed in the event of any legal challenge.

28. What if I haven't signed a CLA?

If you haven't signed a CLA, then we can't yet accept your code contribution into Puppet or Facter. Signing a CLA is very easy: simply log into your GitHub account and go to our CLA page to sign the agreement.

We've worked hard to try find to everyone who has contributed code to Puppet, but if you have questions or concerns about a previous contribution you've made to Puppet and you don't believe you've signed a CLA, please sign a CLA or contact us for further information.

29. Does signing a CLA change who owns Puppet?

The change in license and the requirement for a CLA doesn't change who owns the code. This is a pure license agreement and NOT a Copyright assignment. If you sign a CLA, you maintain full copyright to your code and are merely providing a license to Puppet Labs to use your code.

All other code remains the copyright of Puppet Labs.

30. Which versions of Ruby does Puppet support?

Puppet requires an MRI Ruby interpreter. Certain versions of Ruby are tested more thoroughly with Puppet than others, and some versions are not tested at all. Run ruby –version to check the version of Ruby on your system.

Starting with Puppet 4, puppet-agent packages do not rely on the OS's Ruby version, as it bundles its own Ruby environment. You can install puppet-agent alongside any version of Ruby or on systems without Ruby installed. Likewise, Puppet Enterprise does not rely on the OS's Ruby version, as it bundles its own Ruby environment. You can install PE alongside any version of Ruby or on systems without Ruby installed. The Windows installers provided by Puppet of Ruby or on systems without Ruby installed. The Windows installers provided by Puppet Labs don't rely on the OS's Ruby version and can be installed alongside any version of Ruby or on systems without Ruby installed.

Summary

Reviewer	David
Review Date	2016-03-09
Reviewed Item	Useful Interview Questions To Revise
Author Rating	★★★★★

Meet the author

ANIL

"At TekSlate, we are trying to create high quality tutorials and articles, if you think any information is incorrect or want to add anything to the article, please feel free to get in touch with us at info@tekslate.com, we will update the article in 24 hours."

2 Responses on Puppet Interview Questions and Answers"

devops training says: **REPLY**

DECEMBER 1, 2016 AT 6:14 AM

Nice Article. How it help to developer in terms of balance the day to day life.

DevOps Consult says: **REPLY**

MARCH 15, 2017 AT 11:32 AM

Nice Article .In short description good explanation. Thanks For sharing the informative news.

Leave a Message

YOUR EMAIL ADDRESS WILL NOT BE PUBLISHED. Required fields are marked *

Comment

Name*

Email*

Website

POST COMMENT

Latest Tutorials >

Browse all 157 Tutorials >

Ansible Tutorials

Welcome to the Ansible Tutorials. The objective of these tutorials is to get in depth understanding

SilverLight Tutorials

SilverLight Tutorials Overview Welcome to SilverLight Tutorials. The objective of these tutorials is to gain in

SAP BPC Tutorial

SAP Business Objects Planning and Consolidation – The SAP BusinessObjects Planning and Consolidation ., version

MORE

Home

About

Become Instructor

Disclaimer

COURSES

Blue Prism Training

Hybris Training

Blockchain Training

VIEW ALL COURSES

TUTORIALS

Informatica Tutorials & Interview Questions

Tableau Tutorials & Interview Questions

Salesforce CRM Tutorials

VIEW ALL TUTORIALS

INTERVIEW QUESTIONS

Tableau Interview Questions And Answers

JMeter Interview Question And Answers

Maven Interview Questions

VIEW ALL QUESTIONS