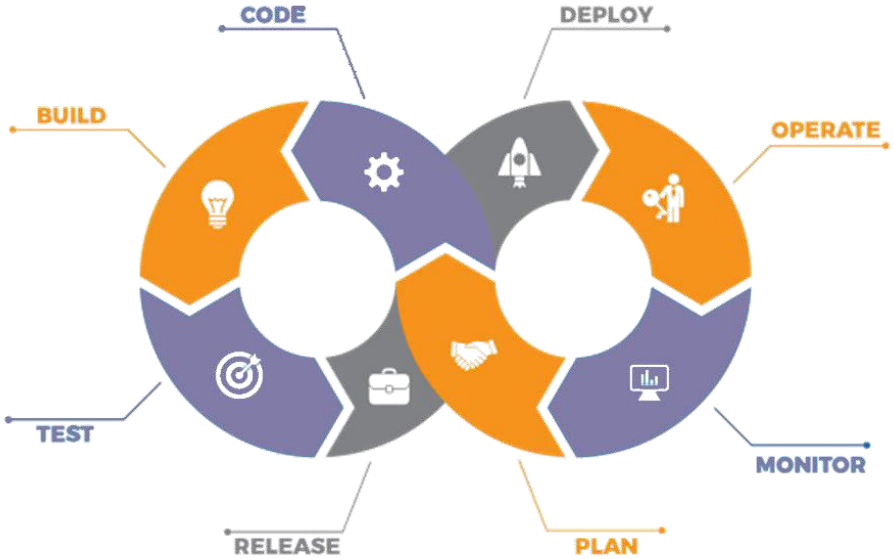
Configuration Management Using Puppet



Agenda

01 Why Configuration Management?

02 What is Configuration Management?

03 Configuration Management Tools

04 What is Puppet?

05 Puppet Architecture

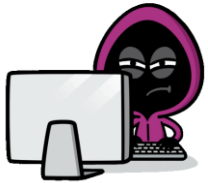
06 Puppet Master–Slave Setup

07 Puppet Code Basics

08 Applying Configuration Using Classes

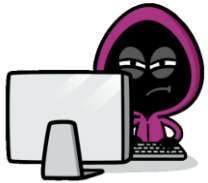
Why Configuration Management?

Why Configuration Management?



Developer

Why Configuration Management?



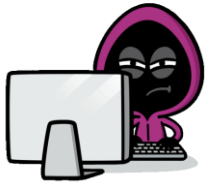
Developer

Application

PHP 5.7

MySQL 4.7

Why Configuration Management?



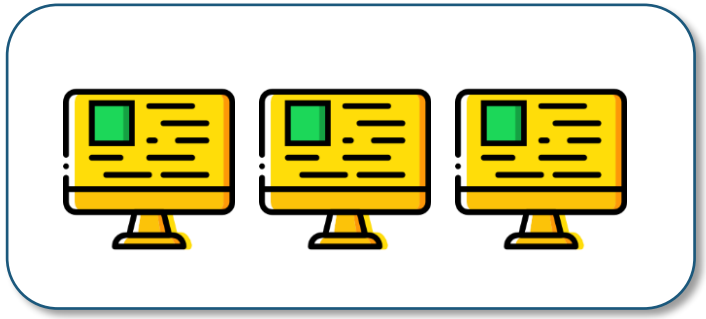
Developer

Application

PHP 5.7

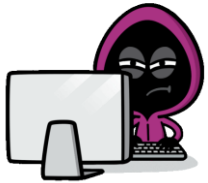
MySQL 4.7

PHP Servers



Database Servers

Why Configuration Management?



Developer

Application

PHP 5.7

MySQL 4.7

PHP 5.7 Servers

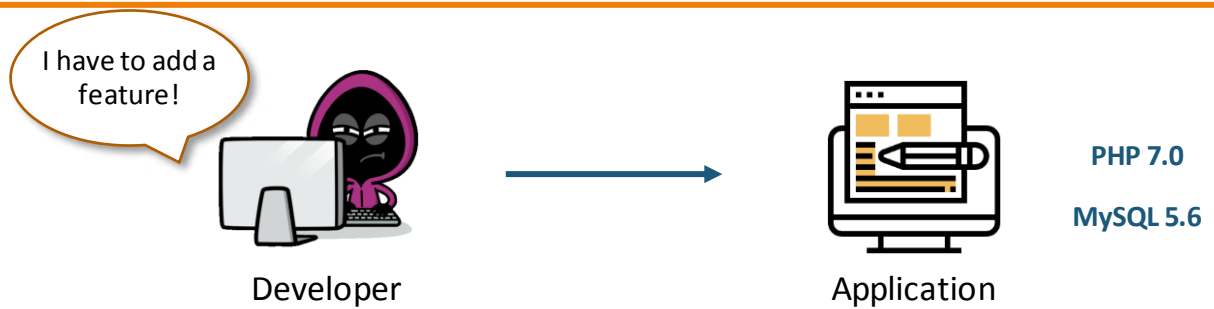
Database 4.7 Servers

What is Configuration Management?

What is Configuration Management?

Configuration management is a systems engineering process for establishing and maintaining consistency of a product's performance, functional and physical attributes with its requirements, design and operational information throughout its life.

What is Configuration Management?

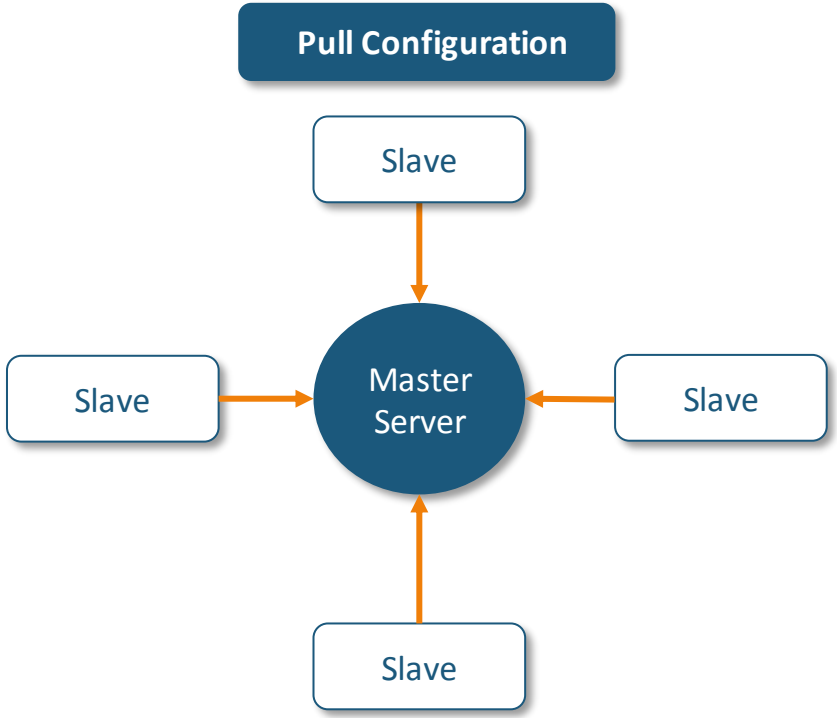


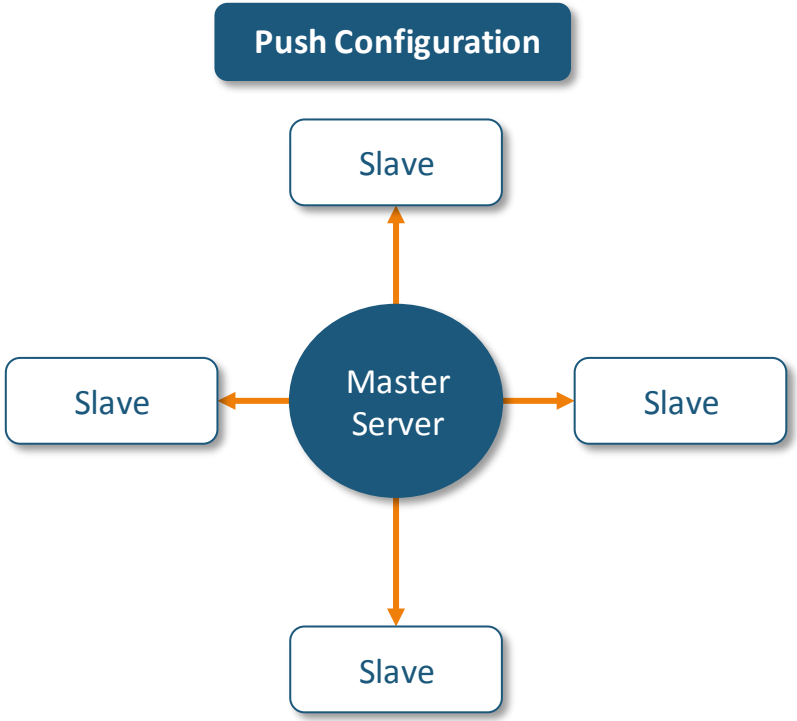
Configuration Management Features

- Automation
- Consistency
- Software Updates
- Software Rollback

Configuration Management Tools

Types of Configuration Management Tools





Types of Configuration Management Tools

Pull Configuration

Push Configuration

What is Puppet?

What is Puppet?

Puppet is an open-source software configuration management tool. It runs on many Unix-like systems, as well as on Microsoft Windows, and includes its own declarative language to describe the system configuration.

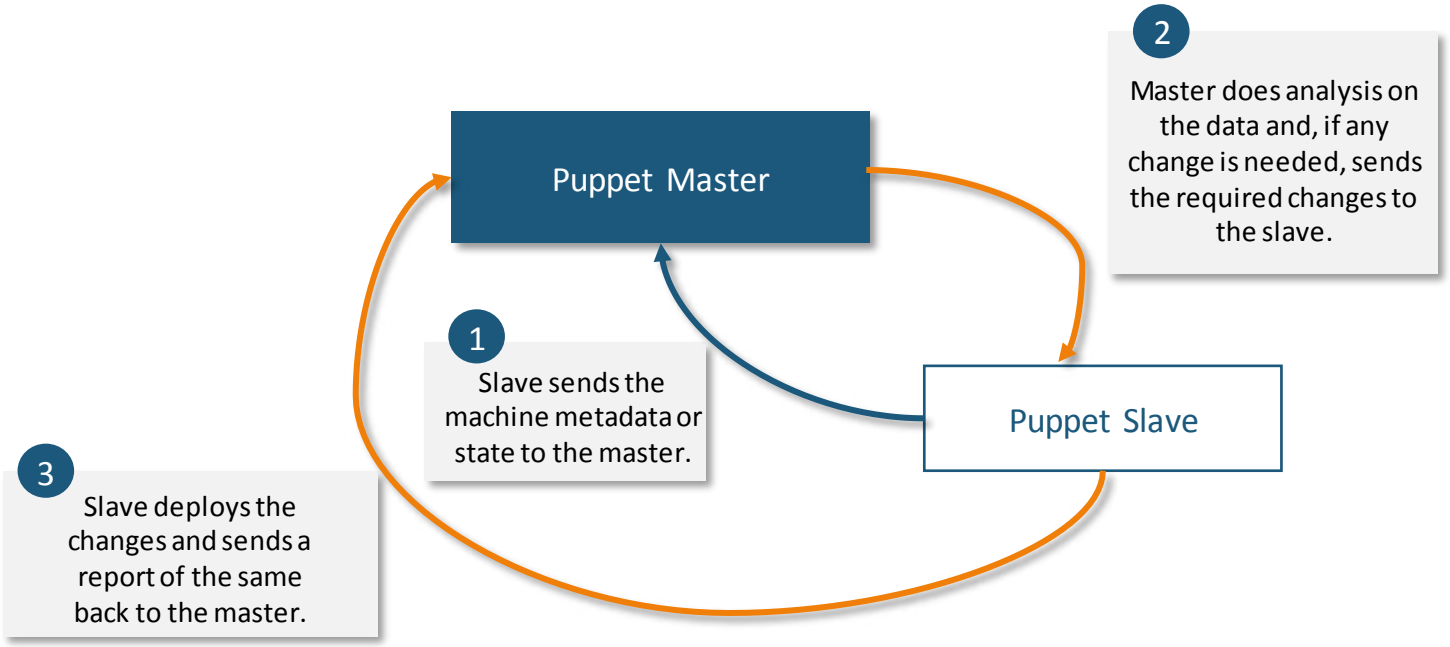
Key Features of Puppet

- Large User Base
- Big Open-source Community
- Documentation
- Platform Support

Puppet Architecture

Puppet Architecture

Puppet follows a Master–Slave architecture, the working of which has been explained in the below diagram.



Puppet Architecture: SSL Connection

Because Puppet nodes have to interact with the master, all the information which is communicated between the master node and slave nodes are encrypted using SSL certificates.
The certificate signing process is as follows:

Setting up Puppet Master–Slave on AWS

Code Basics for Puppet

Code Basics for Puppet

The most basic component of Puppet Code is a **resource**. A resource describes something about the state of the system, such as if a certain user or file should exist, or a package should be installed, etc.

Syntax

```
resource_type { 'resource_name':

        attribute => value,
        ...
}
```

Code Basics for Puppet: Resource Example

Example

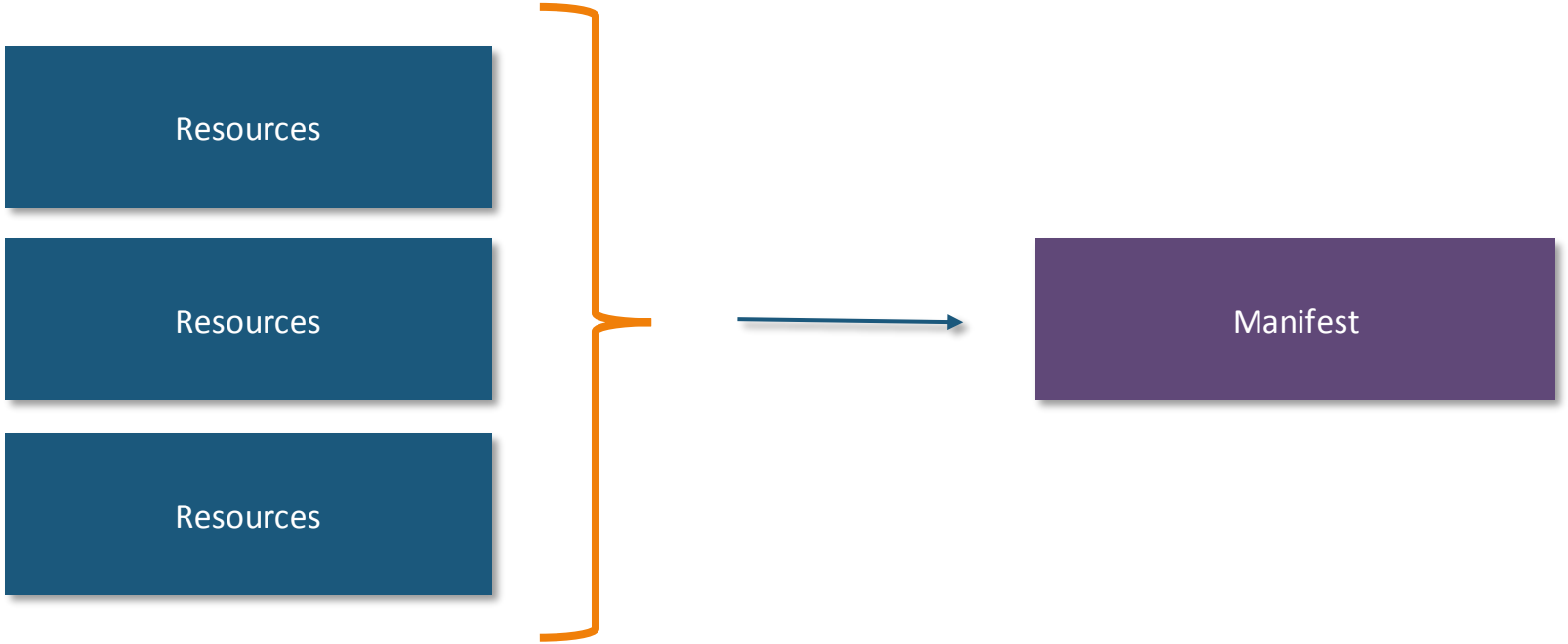
```
package { ‘nginx’:

ensure => ‘installed’,

}
```

Sample nginx package

Code Basics for Puppet: Manifest



Code Basics for Puppet: Manifest

Manifests are basically a collection of resource declarations, using the extension **.pp**.

Example

```
package {  'nginx':
    ensure => 'installed',
}

file {'/tmp/hello.txt':
    ensure => present,
  content => 'hello world',
    mode => '0644',

}
```

Sample Manifest File

Code Basics for Puppet: Manifest

- **Variables**
- Loops
- Conditions

Variables can be defined at any point in a manifest. The most common types of variables are strings and arrays of strings, but other types are also supported, such as Booleans and hashes.

Example

```
$text = “hello world”

file {‘/tmp/hello.txt’:
  ensure => present,
content => ‘hello world’,
  mode => ‘0644’,
}
```

Code Basics for Puppet: Manifest

Variables

Loops

Conditions

Loops are typically used to repeat a task using different input values. For instance, instead of creating 10 tasks for installing 10 different packages, you can create a single task and use a loop to repeat the task with all different packages you want to install.

Example

```
$packages = ['nginx','mysql-server']

        package { $packages:
          ensure => installed,
      }
```

Code Basics for Puppet: Manifest

- Variables
- Loops
- **Conditions**

Conditions can be used to dynamically decide whether or not a block of code should be executed, based on a variable or an output from a command, for instance.

Example

```
exec { "Test":
command =>'/bin/echo apache2 is installed > /tmp/status.txt',
onlyif => '/bin/which apache2',
}
```

Code Basics for Puppet: Manifest

- Variables
- Loops
- **Conditions**

Conditions can be used to dynamically decide whether or not a block of code should be executed, based on a variable or an output from a command, for instance.

Example

```
exec { "Test":
command =>'/bin/echo apache2 is not installed > /tmp/status.txt',
unless => '/bin/which apache2',
}
```

Applying Configuration Using Modules

What are Modules?

A collection of manifests and other related files organized in a predefined way to facilitate sharing and reusing parts of a provisioning

1. sudo puppet module generate <name>
2. Edit the init.pp with a class, and build the module
3. Finally, install the module

What are Classes?

Just like with regular programming languages, classes are used in Puppet to better organize the provisioning and make it easier to reuse portions of the code.

Example

```
Class hello{

                                    exec { “Test”:
        command =>‘/bin/echo apache2 is installed > /tmp/status.txt’,
                                    unless => ‘/bin/which apache2’,
                                }

}
```

Hands-on: Applying Configuration Using Modules

Hands-on: Invoking Module's Classes Based on Node Names

Quiz

Quiz

1. Which of these can be re-used in a Puppet program?

A. Resource

B. Manifest

C. Class

D. None of these

Quiz

1. Which of these can be re-used in a Puppet program?

A. Resource

B. Manifest

C. Class

D. None of these

Quiz

2. What is the mode of communication between the Puppet Master and Slaves?

A. SSH

B. SSL Certificates

C. RDP

D. None of these

Quiz

IntelliPaat

2. What is the mode of communication between the Puppet Master and Slaves?

A. SSH

B. SSL Certificates

C. RDP

D. None of these

Quiz

3. Can we create Modules manually rather than using the utility?

A. Yes

B. No

Quiz

3. Can we create Modules manually rather than using the utility?

A. Yes

B. No

Quiz

4. Which of these is the main manifest file?

A. init.pp

B. site.pp

C. main.pp

D. None of these

Quiz

4. Which of these is the main manifest file?

A. init.pp

B. site.pp

C. main.pp

D. None of these

Quiz

5. Which Loop statement allows the command to execute if the condition is true?

A. unless

B. onlyif

Quiz

5. Which Loop statement allows the command to execute if the condition is true?

A. unless

B. onlyif

India: +91-7847955955

US: 1-800-216-8930 (TOLL FREE)

sales@intellipaat.com

24/7 Chat with Our Course Advisor