**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA MẠNG MÁY TÍNH VÀ TRUYỀN THÔNG**



**BÁO CÁO ĐỒ ÁN**

**Môn: Quản trị mạng và hệ thống**

**Đề tài:**

**Tìm hiểu và triển khai công cụ tự động hóa Puppet**

**GVHD: Đỗ Hoàng Hiển**

**Nhóm thực hiện: Nhóm 11**

**Sinh viên thực hiện:**

**Đỗ Hoàng Anh - 22520041**

**Nguyễn Tài Hiếu - 22520442**

**Nguyễn Việt Hoàng - 22520471**

**Ngô Hồng Phúc - 22521124**

**Nguyễn Minh Thi - 22521380**

Tp. Hồ Chí Minh, tháng 11 năm 2024

**MỤC LỤC**

# 

[**Chương 1: Lời mở đầu 3**](#_sphsmqbnb1z2)

[**Chương 2: Giới thiệu về Puppet 4**](#_tfna21ilqcv4)

[**Chương 3: Công cụ puppet và tính năng chính 6**](#_jreepaw1wg1i)

[**Chương 4: Cài đặt và cấu hình Puppet 9**](#_9ne4l8gk8412)

[**Chương 5: Triển khai và quản lý cấu hình bằng Puppet 12**](#_fj6lf0gze0pa)

[**Chương 6: Trình bày demo nhóm đã thực hiện 17**](#_o2l9q0qm8qhp)

[**Chương 7: Ứng dụng của Puppet 25**](#_daxhh2l5mtmw)

**Tham khảo…………………………………………………………27**

**Câu hỏi bổ sung giảng viên đưa ra……………………………….27**

**BÁO CÁO CHI TIẾT**

# Chương 1: Lời mở đầu

**1.1. Mục tiêu đồ án**

Tìm hiểu và vận dụng công cụ Puppet để tự động hóa việc quản lý cấu hình hệ thống. Cụ thể, đồ án hướng đến các mục tiêu sau:

* Nắm vững kiến thức cơ bản về Puppet: Hiểu rõ các khái niệm cốt lõi như Manifests, Modules, Resources, Classes, Nodes, Puppet Master, và Puppet Agent.
* Nắm được cách thức hoạt động và tương tác giữa các thành phần này. Phát triển kỹ năng viết mã Puppet
* Thành thạo trong việc sử dụng ngôn ngữ khai báo của Puppet để mô tả và quản lý trạng thái mong muốn của hệ thống. Biết cách tạo và sử dụng Modules để tổ chức và tái sử dụng mã Puppet.
* Ứng dụng Puppet vào thực tế: Xây dựng một dự án cụ thể để chứng minh khả năng ứng dụng Puppet vào việc tự động hóa quản lý cấu hình. Ví dụ, triển khai và cấu hình một máy chủ web, cơ sở dữ liệu, hoặc một ứng dụng đa tầng..

**1.2.Tầm quan trọng và ứng dụng của Puppet trong tự động hóa hệ thống**

Mục tiêu chính của việc sử dụng Puppet là **tự động hóa quản lý cấu hình cơ sở hạ tầng CNTT**. Điều này bao gồm tự động hóa các tác vụ như:

* Cung cấp và cấu hình máy chủ. Triển khai và quản lý ứng dụng.
* Quản lý người dùng và quyền truy cập.
* Đảm bảo tuân thủ chính sách và tiêu chuẩn bảo mật.

Việc tự động hóa này giúp đạt được các mục tiêu sau:

* Tăng tính nhất quán và ổn định của hệ thống.
* Giảm thiểu thời gian và công sức quản trị.
* Tăng hiệu suất và giảm chi phí vận hành.
* Cải thiện khả năng mở rộng và linh hoạt.
* Nâng cao tính bảo mật và tuân thủ.

# Chương 2: Giới thiệu về Puppet

**2.1. Khái niệm và lịch sử phát triển của Puppet**

**Khái niệm:**

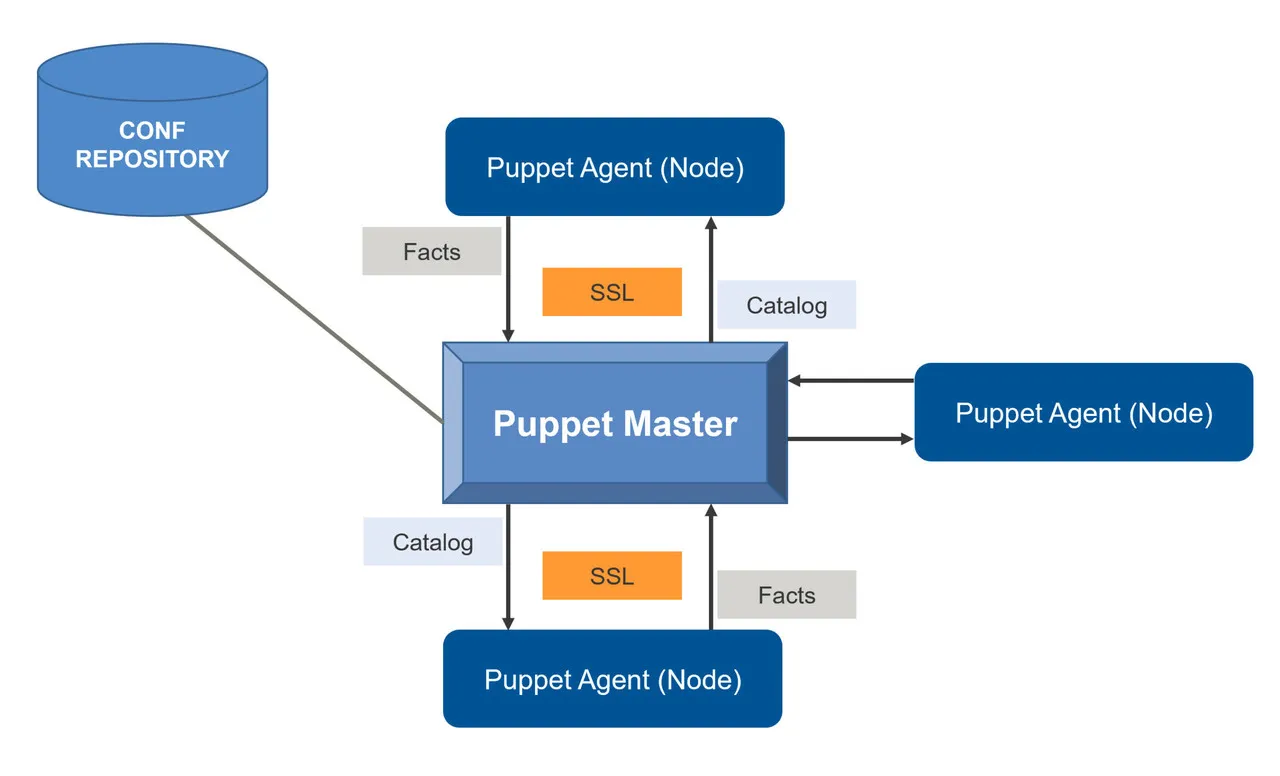
Puppet là công cụ quản lý cấu hình mã nguồn mở, tự động hóa việc cung cấp, cấu hình và quản lý cơ sở hạ tầng CNTT. Nó sử dụng mô hình client-server và ngôn ngữ khai báo để mô tả trạng thái mong muốn của hệ thống.

**Lịch sử phát triển:**

Được khởi tạo năm 2005 bởi Luke Kanies, Puppet ban đầu là dự án mã nguồn mở, sau đó phát triển thành công ty Puppet Labs (nay là Puppet). Puppet Enterprise, phiên bản thương mại, ra mắt năm 2012. Hiện nay, Puppet vẫn là công cụ quản lý cấu hình quan trọng, hỗ trợ cả môi trường cloud và container.

**2.2. Cấu trúc và thành phần chính của Puppet**

Puppet, một công cụ quản lý cấu hình mạnh mẽ, hoạt động dựa trên kiến trúc client-server, cho phép tự động hóa việc cấu hình và quản lý cơ sở hạ tầng CNTT một cách hiệu quả. Mô hình này bao gồm các thành phần cốt lõi sau, phối hợp hoạt động để đảm bảo tính nhất quán và khả năng mở rộng:



1. **Puppet Master (Máy chủ điều khiển):**

* **Vai trò:** Đóng vai trò trung tâm, chịu trách nhiệm lưu trữ và quản lý toàn bộ cấu hình.
* **Chức năng:** 
  + Lưu trữ mã nguồn Puppet (manifests và modules).
  + Biên dịch manifests thành catalog – bản hướng dẫn cấu hình cụ thể cho từng node.
  + Phân phối catalog đến các Puppet Agent.
  + Nhận báo cáo trạng thái từ Puppet Agent.

2. **Puppet Agent (Tác tử trên nút):**

* **Vai trò:** Cài đặt trên từng node (máy chủ) cần quản lý.
* **Chức năng:**
* Gửi "facts" (thông tin về node, ví dụ: hệ điều hành, phần cứng) đến Puppet Master.
* Nhận catalog từ Puppet Master.
* Áp dụng catalog, thực hiện các thay đổi cần thiết để đạt cấu hình mong muốn.
* Báo cáo kết quả về cho Puppet Master.

3. **Các thành phần cấu hình:**

* **Manifests:** Tệp chứa mã Puppet, mô tả cấu hình mong muốn bằng ngôn ngữ khai báo.
* **Modules:** Các gói cấu hình tái sử dụng, giúp tổ chức và quản lý mã Puppet hiệu quả. Bao gồm manifests, templates, files, và các tài nguyên khác.
* **Templates:** Tệp mẫu dùng để tạo động các tệp cấu hình dựa trên dữ liệu của từng node.

4. **Các thành phần hỗ trợ:**

* **Facter:** Thu thập thông tin (facts) về node.
* **Hiera:** Hệ thống tra cứu dữ liệu, cho phép tách biệt dữ liệu cấu hình khỏi mã Puppet.
* **PuppetDB (Tùy chọn):** Cơ sở dữ liệu lưu trữ thông tin về trạng thái, lịch sử và báo cáo, giúp tối ưu hóa hiệu suất và khả năng quản lý.
* **Chứng chỉ SSL:** Đảm bảo an toàn cho việc giao tiếp giữa Puppet Master và Puppet Agent.

**Luồng hoạt động:**

* Agent gửi facts đến Master.
* Master biên dịch catalog dựa trên facts và dữ liệu từ Hiera.
* Master gửi catalog đến Agent.
* Agent áp dụng catalog và báo cáo trạng thái.

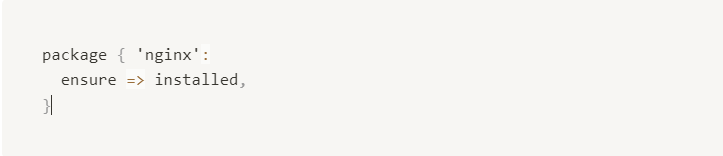
**2.3. Các khái niệm cơ bản trong Puppet**

* **Node (Nút):** Mỗi máy chủ hoặc thiết bị được quản lý bởi Puppet được gọi là một node. Mỗi node có một tên duy nhất, được Puppet Master sử dụng để xác định và áp dụng cấu hình chính xác.
* **Manifest (Bản kê khai):** Đây là tập tin chứa mã Puppet, được viết bằng ngôn ngữ khai báo riêng của Puppet. Manifest định nghĩa cấu hình mong muốn của một hoặc nhiều nodes. Chúng thường có phần mở rộng .pp.
* **Resource (Tài nguyên):** Đây là các khối xây dựng cơ bản của cấu hình Puppet. Mỗi resource đại diện cho một thành phần của hệ thống, ví dụ như một tệp tin, một gói phần mềm, một dịch vụ, hoặc một người dùng. Mỗi resource có một type (loại), title (tên), và một tập hợp các attributes (thuộc tính).
* **Class (Lớp):** Classes cho phép nhóm các resources liên quan lại với nhau, tạo thành một khối cấu hình có thể tái sử dụng. Classes cũng hỗ trợ tính kế thừa, cho phép một class kế thừa các thuộc tính và resources từ một class khác.
* **Module (Mô-đun):** Modules là cách đóng gói và phân phối mã Puppet. Một module chứa manifests, files, templates và các thành phần khác cần thiết để quản lý một chức năng cụ thể của hệ thống (ví dụ: cài đặt và cấu hình Apache).
* **Variable (Biến):** Biến được sử dụng để lưu trữ dữ liệu và có thể được sử dụng trong manifests. Chúng cho phép bạn viết mã Puppet linh hoạt hơn.
* **Fact (Thông tin thực tế):** Facts là những dữ liệu về node, được thu thập tự động bởi Facter, chẳng hạn như hệ điều hành, địa chỉ IP, tên máy chủ... Facts được sử dụng để tùy chỉnh cấu hình cho từng node cụ thể.
* **Template (Mẫu):** Templates là các tập tin văn bản chứa mã Puppet và các biến. Chúng được sử dụng để tạo ra các tệp cấu hình động, dựa trên dữ liệu cụ thể của từng node.
* **Certificate (Chứng chỉ):** Chứng chỉ được sử dụng để xác thực an toàn giữa Puppet Master và Puppet Agent, đảm bảo rằng chỉ những agent được ủy quyền mới có thể nhận cấu hình từ Master.
* **Catalog (Danh mục):** Puppet Master biên dịch manifests thành catalog, một tập hợp các resources và thuộc tính cụ thể cho từng node. Puppet Agent sau đó sử dụng catalog này để áp dụng cấu hình lên node.

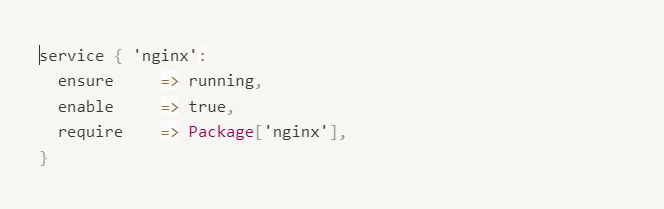
# Chương 3: Công cụ puppet và tính năng chính

**3.1. Quản lý cấu hình và triển khai tự động**

* Quản lý cấu hình:
* Là quá trình duy trì phần mềm và hệ thống máy tính (ví dụ máy chủ, lưu trữ, mạng) ở trạng thái đã biết, mong muốn và nhất quán. Nó cũng cho phép truy cập vào hồ sơ lịch sử chính xác về trạng thái hệ thống cho mục đích quản lý dự án và kiểm toán.
* Sử dụng các manifest và module để định nghĩa các trạng thái mong muốn của các tài nguyên trên hệ thống, như tệp tin, dịch vụ, gói phần mềm, và nhiều thứ khác. Trong đó:
  + Manifest: mô tả trạng thái hệ thống dùng ngôn ngữ Puppet DSL (Domain Specific-Language).
  + Module: nơi quản lí manifests, files, templates và các thành phần khác cần thiết.
* Triển khai tự động:
* Puppet hỗ trợ tự động hóa các tác vụ quản trị hệ thống bằng cách định nghĩa các tác vụ này dưới dạng mã (Infrastructure as Code - IaC).
* Cụ thể như sau:
  + Cài đặt phần mềm: Puppet quản lý thông qua resource **package** và ta có thể định nghĩa gói phần mềm cần cài đặt nhằm đảm bảo chúng luôn được cài đặt ở phiên bản mong muốn. Ví dụ sau đây, Puppet đảm bảo rằng gói phần nginx được cài đặt trên hệ thống



* + Khởi động và quản lí dịch vụ: Puppet sử dụng resource **service** để quản lý dịch vụ hệ thống. Ví dụ sau đây, Puppet đảm bảo rằng **nginx** đang chạy và kích hoạt tự động khi hệ thống khởi động, và dịch vụ này phụ thuộc vào gói phần mềm **nginx.**



* + Quản lý tệp cấu hình: Puppet sử dụng resource **file** để quản lý các tệp cấu hình. Ví dụ sau đây, Puppet đảm bảo rằng tệp cấu hình **nginx.conf** tồn tại với nội dung được định nghĩa trong template và thông báo cho dịch vụ **nginx** khởi động lại nếu có thay đổi.



* + Quản lý tệp cấu hình: Puppet có thể tự động áp dụng các bản vá bảo mật bằng cách quản lý các gói phần mềm và đảm bảo chúng luôn được cập nhật. Đồng thời, tự động hóa toàn bộ vòng đời quản trị hệ thống thông qua việc tích hợp các module và class để quản lý tất cả các khía cạnh của hệ thống.
* **Kết luận**: Puppet là công cụ mạnh mẽ và linh hoạt trong việc quản lý cấu hình và triển khai tự động, cụ thể Puppet cho phép định nghĩa cấu hình giúp đảm bảo sự nhất quán giữa các máy chủ nhằm phục vụ cho việc dễ quản lý, kiểm tra, tái sử dụng và giảm thiểu lỗi do con người gây ra; đồng thời, Puppet hỗ trợ tự động hóa toàn bộ vòng đời của cấu hình hệ thống, từ triển khai đến duy trì và cập nhật, thuận lợi cho việc giảm khối lượng công việc thủ công và tăng tính an toàn, hiệu quả cho hệ thống.

**3.2. Quản lý hạ tầng hệ thống với Puppet**

* Quản lý hạ tầng hệ thống với Puppet là việc duy trì và quản lí cấu hình mã nguồn mở giúp tự động hóa việc cài đặt, cấu hình và quản lí hệ thống. Cụ thể
* Ngôn ngữ khai báo: Puppet sử dụng Infrastructure as Code để định nghĩa trạng thái mong muốn của hệ thống.
* Cấu hình tập trung: Puppet quản lý cấu hình từ một nơi tập trung (Puppet Master).
* Puppet đảm bảo rằng việc áp dụng cùng một cấu hình nhiều lần sẽ không thay đổi hệ thống nếu nó đã ở trạng thái mong muốn. Điều này giúp ngăn ngừa việc thay đổi ngoài ý muốn.
* Modules và Manifests.
* Các khía cạnh của việc quản lý hạ tầng hệ thống với Puppet:
* Triển khai và tự động cập nhật ứng dụng: Puppet có thể được tích hợp với các công cụ CI/CD như Jenkins để tự động triển khai và cập nhật ứng dụng trên nhiều môi trường (dev, test, production) một cách nhất quán và an toàn.

**Ví dụ**: Tích hợp Jenkins vào Puppet, ta có cơ chế vận hành như sau: Cài đặt và cấu hình Jenkins → Cấu hình Git → Thiết lập Puppet Master và Puppet Agents → Viết Manifest và Modules → Thiết lập Jenkins Pipeline.

* Quản đa đám mây và hạ tầng ảo hóa: Puppet hỗ trợ quản lý hạ tầng trên nhiều nền tảng đám mây (AWS, Azure, Google Cloud) cũng như các môi trường ảo hóa (VMware, OpenStack), giúp các công ty lớn dễ dàng quản lý môi trường đa đám mây. Vì Puppet giúp định nghĩa và quản lý hạ tầng dưới dạng mã, cho phép dễ dàng sao chép và thay đổi hạ tầng khi cần thiết.

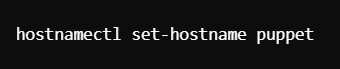
**Ví dụ** Thiết lập hệ thống Puppet cho môi trường đa đám mây với Puppet Agents tại các máy chủ trên AWS, Azure, ...

*\*\*\* KẾT LUẬN CHƯƠNG*: Puppet là một công cụ mạnh mẽ và hiệu quả trong việc quản lý cấu hình và triển khai tự động hệ thống. Với khả năng tự động hóa các tác vụ như cài đặt phần mềm, cấu hình hệ thống, và bảo trì, Puppet giúp giảm thiểu sai sót của con người, tối ưu hóa quy trình vận hành, và tăng cường tính nhất quán trên các máy chủ trong môi trường hạ tầng phức tạp.

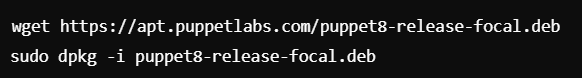
# Chương 4: Cài đặt và cấu hình Puppet

**4.1. Cài đặt Puppet Master và Puppet Agent**

**4.1.1. Cài đặt Puppet Master**

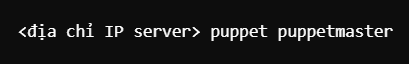
****

* Thiết lập hostname của hệ thống thành puppet
* Đặt tên cho máy chủ để định dạng trên mạng , puppet server thường được đặt tên là puppet để các agent dễ dàng nhận diện



* Tải tệp puppet8-release-focal.deb từ URL của Puppetlabs
* Cho phép hệ thống truy cập vào các gói phần mềm chính thức của Puppet Labs.

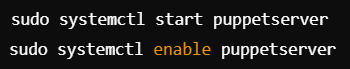




* **Puppet:** hostname chính mà agent sẽ sử dụng để kết nối đến puppetserver
* **Puppetmaster:** tên thay thế để nhận diện Puppet server

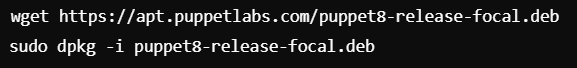


* Cài đặt gói phần mềm **puppetserver** từ kho dữ liệu vừa thêm
* Puppet Server chịu trách nhiệm biên dịch catalog (cấu hình) cho Puppet Agents và quản lý các module cấu hình.



* Đảm bảo Puppet Server hoạt động ngay sau khi cài đặt, giúp dịch vụ duy trì hoạt động mỗi lần khởi động lại máy chủ.

**4.1.2. Cài đặt Puppet Agent**

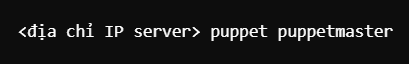
****

* Tải và cài đặt kho lưu trữ cho Puppet



* Cài đặt **puppet-agent** từ khu lưu trữ vừa tải.
* Puppet Agent chịu trách nhiệm thực thi các catalog (cấu hình) nhận từ Puppet Server.





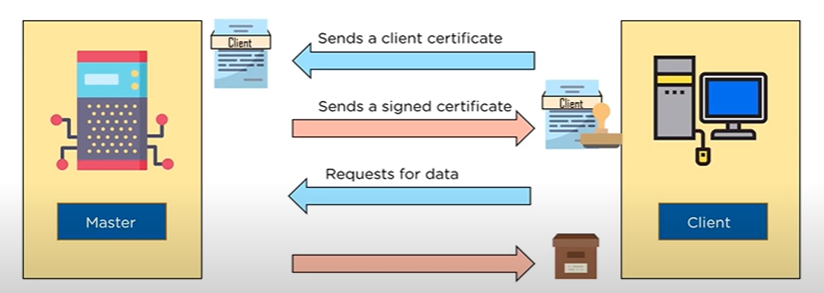
* Thêm địa chỉ IP của puppetmaster vào file hosts của agent



* Khởi động Puppet Agent.
* Thiết lập Puppet Agent để luôn sẵn sàng chạy trên hệ thống.

**Lưu ý:** Các agent khác cài đặt tương tự như các cài đặt puppet-agent trên.

**4.2. Cấu hình và thiết lập kết nối giữa Master và Agent**

****

* Sau khi đã cài đặt Puppet-master và Puppet-agent muốn mô hình hoạt động cần phải trải qua bước xác thực chứng chỉ giữa 2 phía.

Gửi chứng chỉ từ **Agent** đến **Master.**

* Puppet agent gửi một **Certificate Signing Request (CSR)** đến với Puppet Master bằng câu lệnh:

***sudo /usr/bin/puppet agent --test***

Puppet Agent sẽ khởi động ngay trong chế độ kiểm tra và đồng thời yêu cầu chứng chỉ đến Puppet Master.

* CSR chứa thông tin về Agent, bao gồm hostname, địa chỉ IP, và các chi tiết khác.
* Đây là bước khởi tạo để yêu cầu Puppet Master xác thực Agent.

**Master** ký chứng chỉ và gửi lại.

* Puppet **Master** nhận CSR từ **Agent**.

***sudo /usr/bin/puppetserver ca list***

Thực hiện câu lệnh trên để kiểm tra xem có yêu cầu chứng chỉ nào trên Puppet Master hay không.

* Sau khi quản trị viên kiểm tra và xác nhận, **Master** sẽ ký chứng chỉ bằng cách sử dụng **chứng chỉ gốc (CA)**.

***sudo /usr/bin/puppetserver ca sign --certname <hostname puppetagent>***

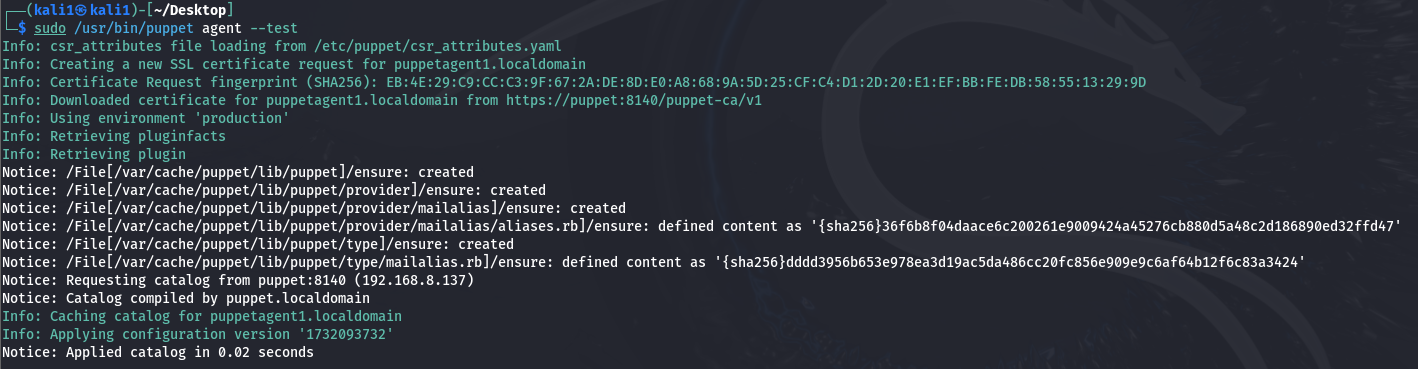
Với **–certname** là hostname của **Puppet agent**

* Chứng chỉ đã ký được gửi ngược lại cho Agent.

Kiểm tra phía client.

* Sau khi Puppet Master đã ký chứng chỉ, chúng ta có thể kiểm tra bằng câu lệnh sau trên Puppet Agent.

***sudo /usr/bin/puppet agent --test***



Sau thực hiện các bước trên, chúng ta đã hoàn thành cài đặt và kết nối giữa Puppet Master và Puppet Agent. Nếu thêm một hoặc nhiều Puppet Agent thì chúng ta có thể thực hiện các bước trên để tạo nên một mô hình hoàn chỉnh.

# Chương 5: Triển khai và quản lý cấu hình bằng Puppet

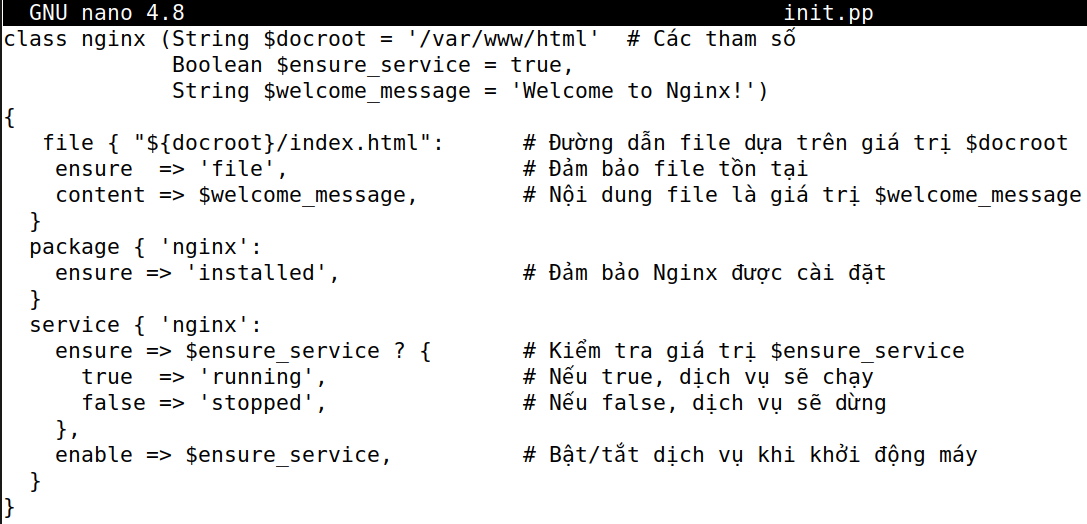
**5.1. Viết Class và Manifests trong Puppet**

**5.1.1. Class**

* **Class** là một khối mã tái sử dụng, chứa các khai báo và định nghĩa cấu hình cho tài nguyên (resources). Class được thiết kế để tổ chức mã dễ hiểu, dễ mở rộng.
* Cấu trúc cơ bản của Class:



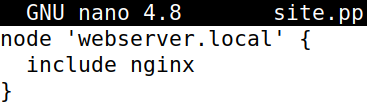
* Ví dụ: Class cài đặt Nginx:



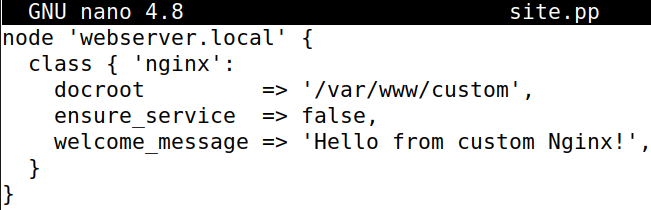
* Chỉ định nghĩa Class thì chưa được áp dụng cho hệ thống. Sau khi định nghĩa, cần khai báo Class để Puppet áp dụng các cấu hình của Class đó vào node hoặc môi trường cụ thể.

**5.1.2. Manifest**

* **Manifest** là các tệp mã Puppet được sử dụng để khai báo cấu hình cụ thể (Class), thường có phần mở rộng *.pp*.
* Có hai cách khai báo class:
* **Include-like declarations:** class được gọi mà không cần truyền tham số, nó sẽ sử dụng các giá trị mặc định đã định nghĩa trong class.



* **Resource-like declarations:** khai báo Class như một tài nguyên để có thể tùy chỉnh giá trị các tham số.



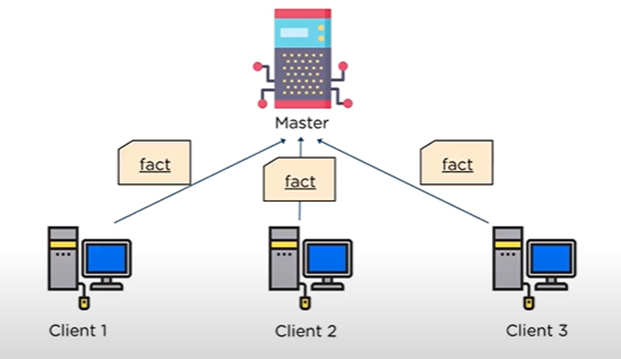
**5.2. Sử dụng Module để quản lý phần mềm và dịch vụ.**

Gửi thông tin **Facts** từ Agent:

**Facts? :** là tập hợp thông tin chi tiết về hệ thống của Puppet Agent, những thông tin này mô tả trạng thái hiện tại của hệ thống và giúp Puppet Master hiểu rõ cấu hình của Agent.

Các thông tin trong **Facts** bao gồm:

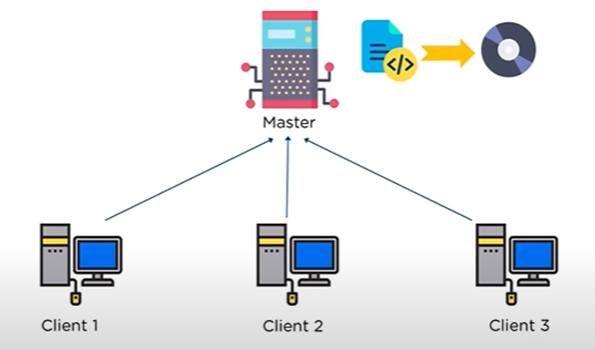
* **Hostname**: Tên của hệ thống hoặc tên miền (FQDN).
* **IP Address**: Địa chỉ IP của hệ thống.
* **Thông tin kernel**: Phiên bản kernel (Linux, Windows, hoặc các hệ điều hành khác).
* **Hệ điều hành**: Loại hệ điều hành và phiên bản (Ubuntu 20.04, Windows 10, v.v.).
* **Tình trạng phần mềm/dịch vụ**: Danh sách các gói hoặc dịch vụ đã được cài đặt, phiên bản hiện tại của các phần mềm, v.v.



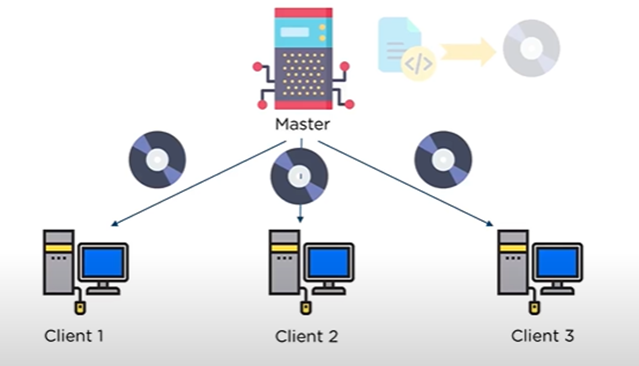
Puppet Master biên dịch **Catalog:**

**Catalog? :** một danh sách các hành động cần được thực hiện trên Puppet Agent để đảm bảo hệ thống của Agent tuân thủ đúng cấu hình định nghĩa trong tệp manifest và các module trên Puppet Master.

* Puppet Master kiểm tra **Facts** đã nhận từ Agent.
* Puppet Master so sánh Facts với cấu hình được định nghĩa trong các tệp **manifest** hoặc **module**.
* Dựa trên sự khác biệt giữa cấu hình mong muốn và thực trạng hệ thống, Puppet Master tạo một **Catalog** để yêu cầu Agent thực hiện các thay đổi.



Sau khi soạn xong **Catalogs**  Puppet Master gửi lại cho Puppet Agent…

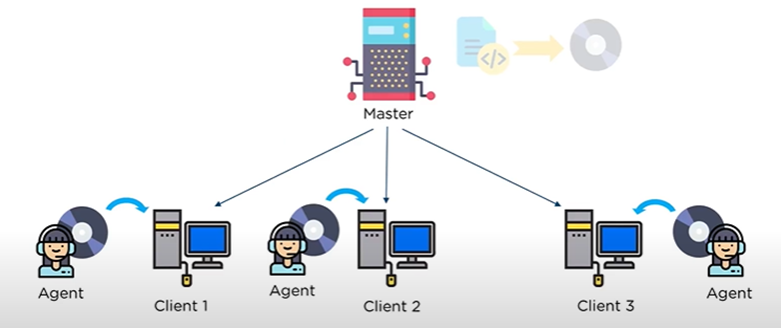


Puppet Agent áp dụng cấu hình:

Sau khi nhận được Catalog từ Puppet Master, Puppet Agent thực hiện các hành động được chỉ định trong Catalog để cập nhật cấu hình trên node. Puppet Agent kiểm tra từng mục trong Catalog và thực hiện các thay đổi cần thiết trên hệ thống.

Các loại thay đổi mà Agent có thể thực hiện:

* Cài đặt phần mềm
* Cập nhật hoặc xóa tệp
* Quản lý dịch vụ
* Thay đổi cấu hình mạng
* ….



Báo cáo trạng thái trở lại Puppet Master:

Sau khi hoàn thành tất cả các thay đổi, Puppet Agent sẽ gửi báo cáo về Puppet Master. Mục đích của báo cáo giúp quản trị viên kiểm tra trạng thái của hệ thống và ghi lại lịch sử thay đổi cấu hình.



# Chương 6: Trình bày demo nhóm đã thực hiện

**6.1. Triển khai cài đặt Nginx cho Puppet Agent**

**Kịch bản:**

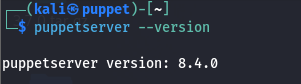
* Puppet Agent và Puppet Master được cài đặt trên 2 máy chủ khác nhau. Puppet Agent gửi yêu cầu chứng chỉ đến Puppet Master.
* Quản trị viên chấp nhận chứng chỉ từ Puppet Master để thiết lập kết nối bảo mật giữa hai bên.
* Quản trị viên viết một module Puppet để tự động cài đặt Nginx, kích hoạt dịch vụ, và tạo file HTML mặc định cho trang web.
* Puppet Master gán module Nginx vào node của Puppet Agent
* Puppet Agent chạy lệnh ***puppet agent --test*** để nhận cấu hình từ Puppet Master và áp dụng nó.
* Quản trị viên kiểm tra trên Puppet Agent

**Thực hiện:**

* Kiểm tra phiên bản puppet trên Puppet Master và Puppet Agent.

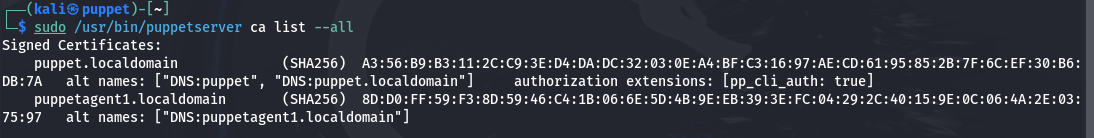


***Puppet Agent***

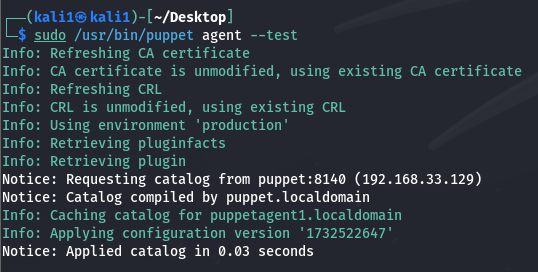


***Puppet Master***

* Kiểm tra chứng chỉ Puppet Master kí với Puppet Agent đã thực hiện ở mục trên.



***Danh sách chữ ký trên Puppet Master***

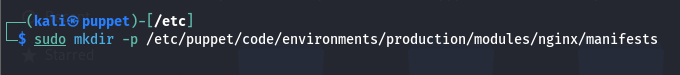
******

***Kiểm tra chứng chỉ trên Puppet Agent***

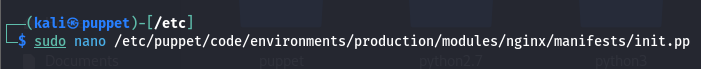
* Khởi chạy Puppet Master.



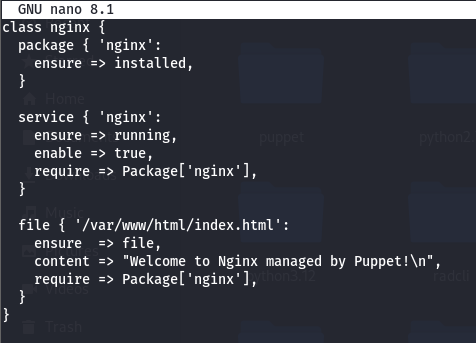
* Tạo module theo đường dẫn sau. Module trong Puppet là tập hợp các tệp và cấu hình được tổ chức để quản lý một ứng dụng hoặc dịch vụ cụ thể.



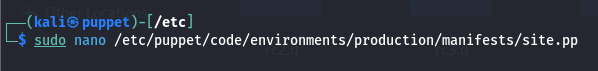
* Tạo file cấu hình trong thư mục vừa tạo.



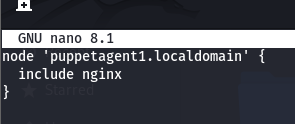
* Thêm nội dung



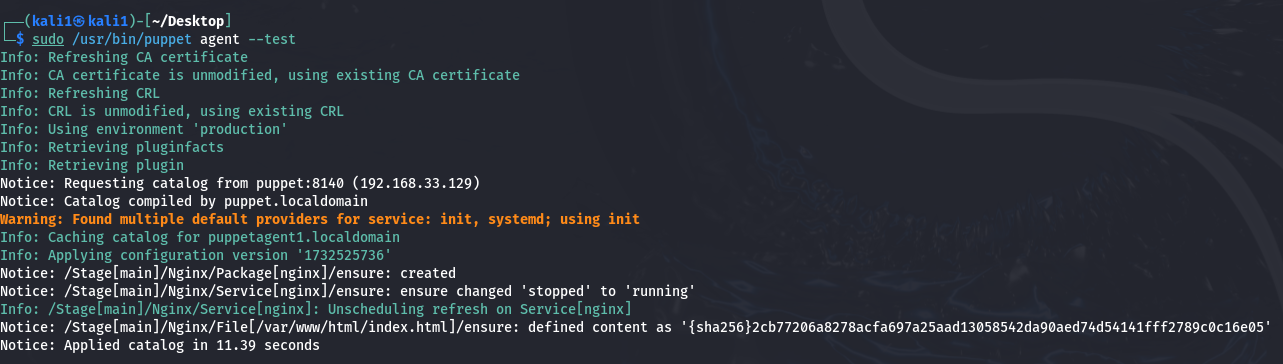
* Tạo file site manifests



* Thêm nội dung

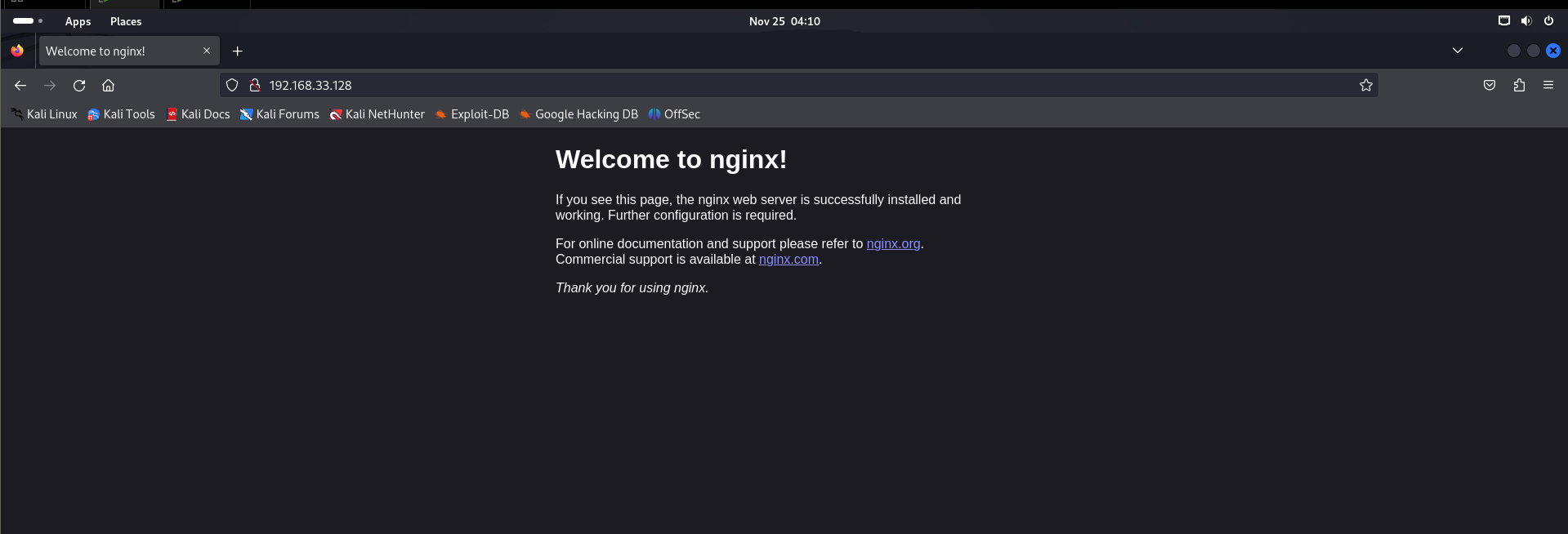


* Node là domain name của Puppet Agent



* Chạy lệnh kiểm tra bên Puppet Agent

**Kết quả:**

****

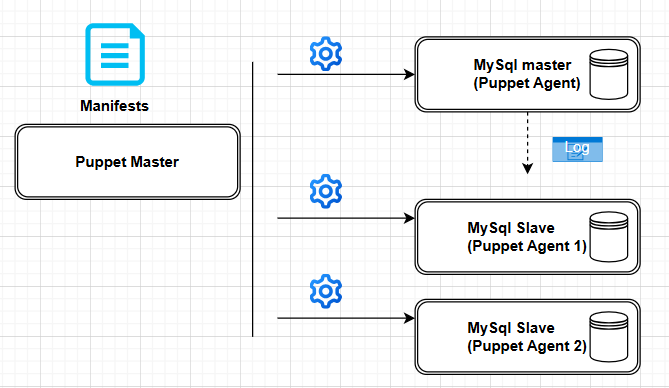
**6.2. Triển khai HA MySQL với Puppet**

**Kịch bản:** Triển khai High Availability (HA) MySQL với Puppet bao gồm việc thiết lập một hệ thống MySQL Master-Slave Replication để đảm bảo tính sẵn sàng cao và đồng bộ dữ liệu giữa các node.

**Kiến trúc:**

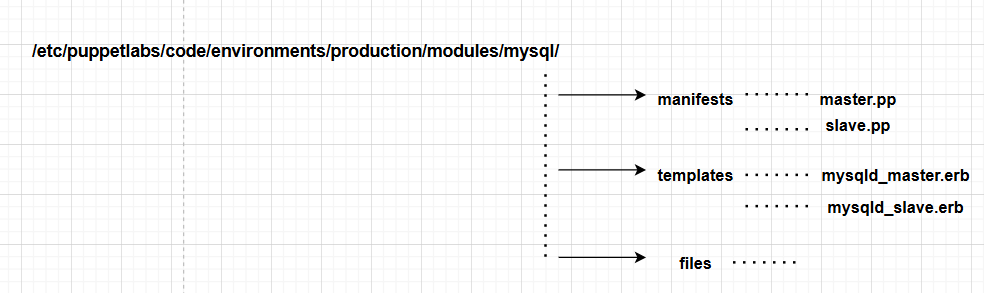
* **MySQL Master**: Máy chủ chính xử lý các thao tác ghi (INSERT, UPDATE, DELETE).
* **MySQL Slave(s)**: Máy chủ phụ đồng bộ dữ liệu từ máy chủ Master. Chủ yếu dùng cho đọc hoặc làm dự phòng khi Master gặp sự cố.
* **Puppet Master**: Quản lý cấu hình MySQL trên các Puppet Agent (bao gồm Master và Slave MySQL).

**Mô hình:**

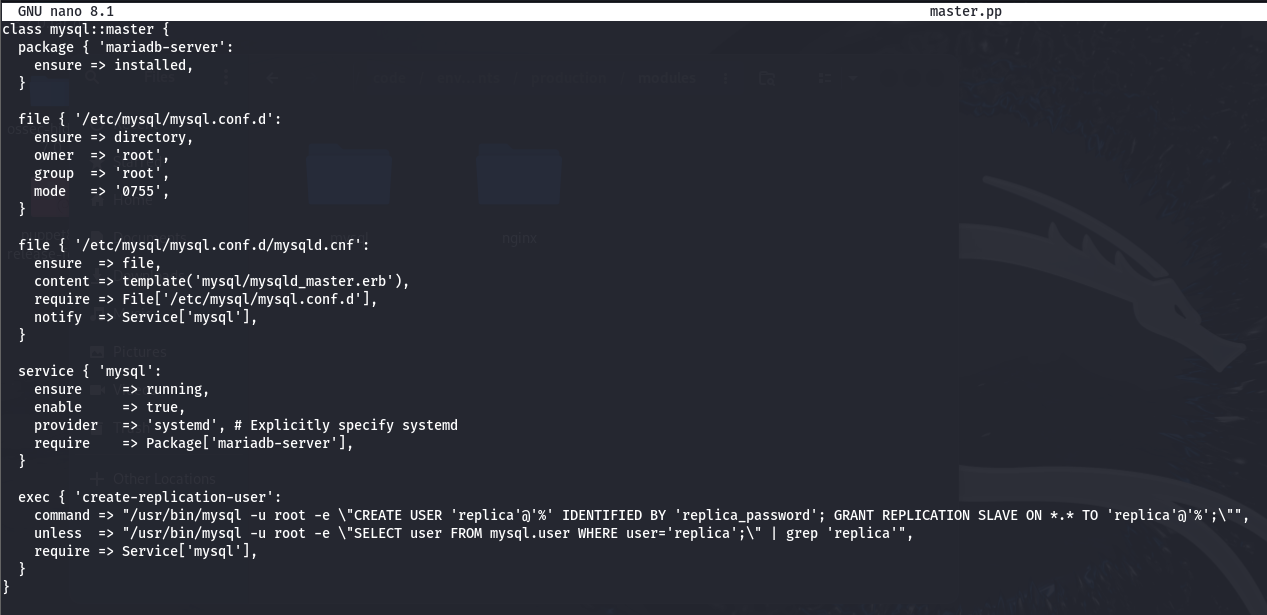
****

**Triển khai:**

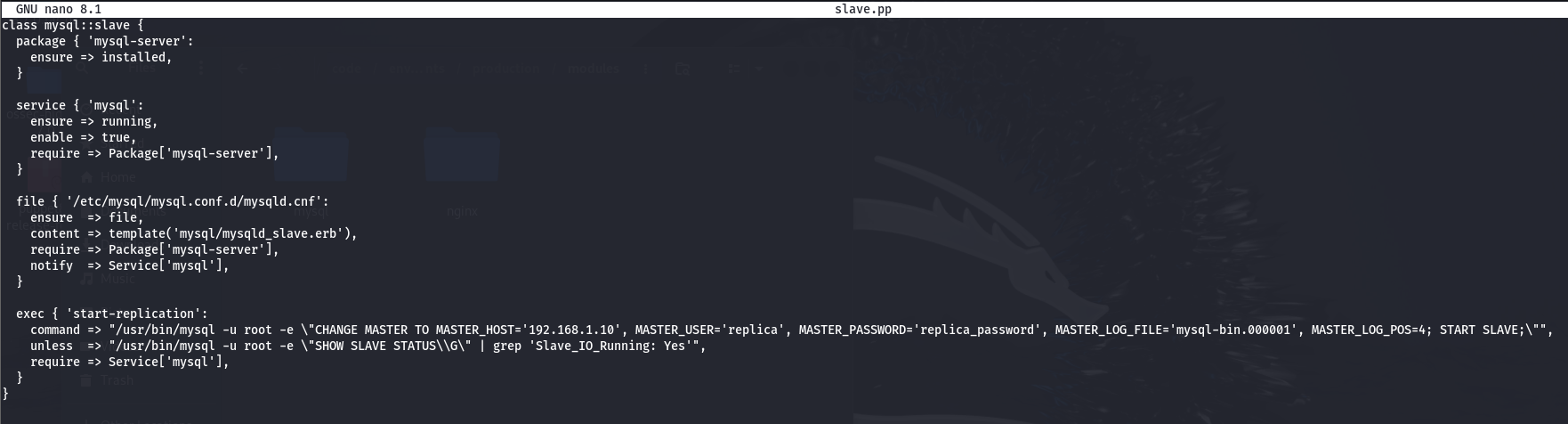
* Cài đặt Puppet Master và Puppet Agent như các mục trên, đồng thời xác nhận chứng thực và thiết lập kết nối giữa Puppet Master và Puppet Agent.
* Tạo cấu trúc thư mục modules như sau:



* Ở file master.pp và file slave.pp thêm cấu hình như sau:

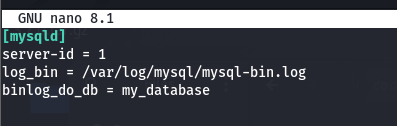


***master.pp***

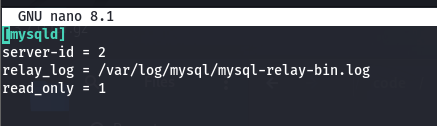
******

***slave.pp***

* Ở 2 file mysqld\_master.erb và mysql\_slave.erb cấu hình như sau:

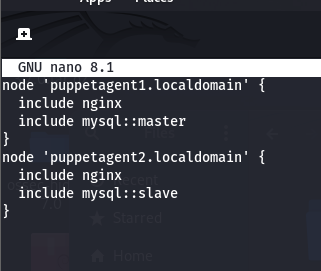


***Mysqld\_master.erb***

******

***Mysqld\_slave.erb***

* Cấu hình file site.pp

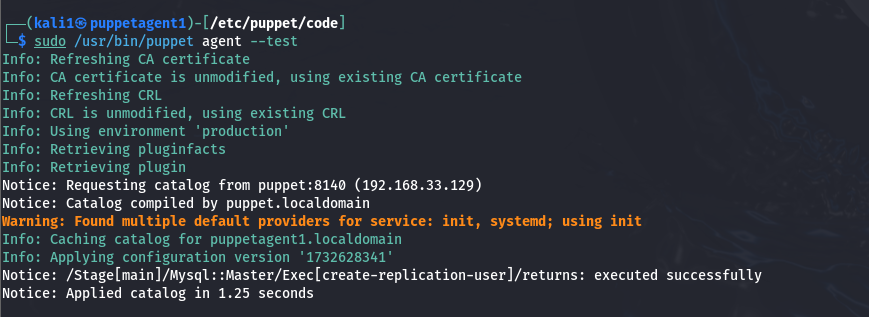


***site.pp***

* Chạy lệnh để app dụng cấu hình



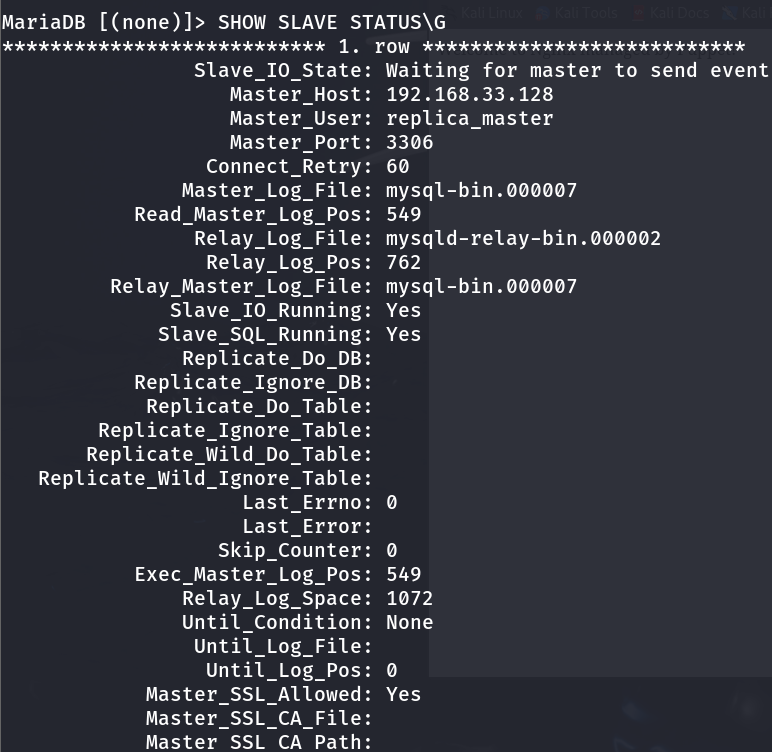
* Kiểm tra trên Puppet Agent đóng vai trò master

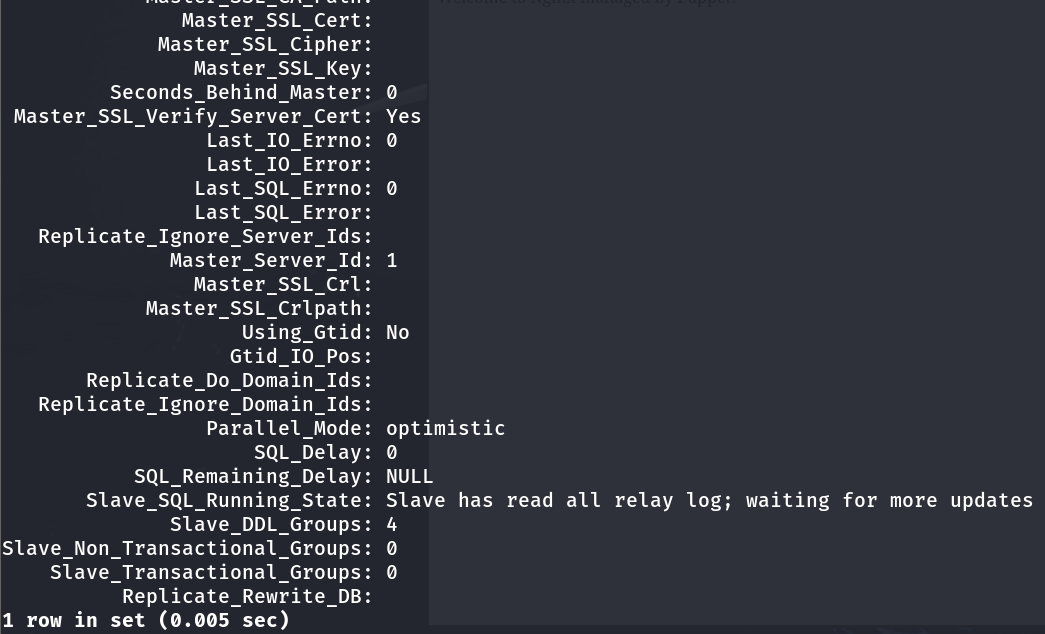


* Kiểm tra trên Puppet Agent đóng vai trò là slave

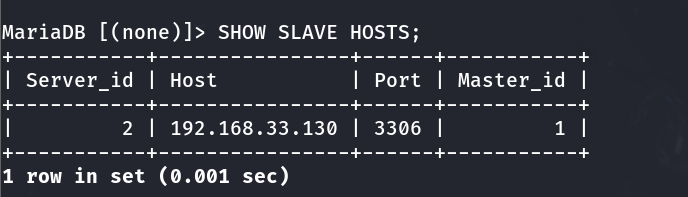


* Ta sẽ thực hiện kiểm tra xem Puppet đã triển khai thành công mô hình Master-Slave.





* Trên Puppet Agent đóng vai trò là slave đã kết nối thành công.



* Trên Puppet Agent đóng vai trò là master đã nhận được yêu cầu từ slave. Slave ở đây có IP là 192.168.33.130.

# Chương 7: Ứng dụng của Puppet

**7.1. Quản lý và cấu hình hệ thống máy chủ**

Puppet là một công cụ mạnh mẽ dùng để tự động hóa và quản lý cấu hình của hệ thống máy chủ. Việc này giúp các quản trị viên hệ thống có thể dễ dàng triển khai và duy trì các cấu hình nhất quán trên hàng nghìn máy chủ mà không cần phải thực hiện thủ công.

**Các tính năng chính**:

***Khai báo cấu hình hệ thống (Manifests):*** Puppet sử dụng các tập tin cấu hình gọi là manifests để khai báo trạng thái mong muốn của hệ thống. Mỗi manifest chứa một hoặc nhiều khai báo về cấu hình, như cài đặt phần mềm, cấu hình dịch vụ, hoặc các cài đặt hệ thống khác. Puppet so sánh trạng thái thực tế của máy chủ với trạng thái mong muốn và thực hiện các thay đổi cần thiết.

***Quản lý cấu hình tập trung***: Puppet cho phép quản lý các cấu hình từ một máy chủ trung tâm (Puppet Master) và triển khai các cấu hình này trên tất cả các máy chủ (Puppet Agent) trong hệ thống. Điều này giúp đảm bảo tính nhất quán và giảm thiểu lỗi do sự khác biệt cấu hình giữa các máy chủ.

***Quản lý nhiều hệ điều hành***: Puppet hỗ trợ nhiều hệ điều hành khác nhau như Linux, Windows, và macOS. Điều này giúp việc quản lý hệ thống trong môi trường đa nền tảng trở nên dễ dàng và đồng nhất.

***Quản lý máy chủ đồng loạt:*** Puppet có thể áp dụng cùng một cấu hình cho hàng loạt máy chủ hoặc phân chia các máy chủ thành các nhóm (node groups) để áp dụng các cấu hình khác nhau cho từng nhóm. Điều này cực kỳ hữu ích khi có nhiều loại máy chủ với các yêu cầu cấu hình khác nhau.

***Tái sử dụng cấu hình***: Puppet hỗ trợ tái sử dụng cấu hình thông qua các classes và modules, giúp giảm thiểu việc phải viết lại mã cấu hình. Các module có thể được chia sẻ giữa các tổ chức hoặc sử dụng lại trong nhiều dự án khác nhau.

***Chạy theo lịch trình và tự động hóa***: Puppet có thể được cấu hình để chạy tự động theo lịch trình, tự động áp dụng các cấu hình mới khi có thay đổi hoặc khi có các bản vá bảo mật mới, giúp duy trì hệ thống luôn được cập nhật và bảo mật.

**7.2. Tự động hóa các tác vụ hệ thống**

Một trong những ứng dụng quan trọng của Puppet là tự động hóa các tác vụ quản trị hệ thống, giúp giảm thiểu công việc thủ công và các lỗi do con người.

**Các tính năng chính**:

***Cập nhật phần mềm và bản vá tự động:*** Puppet giúp tự động cập nhật phần mềm và cài đặt các bản vá bảo mật trên các máy chủ mà không cần sự can thiệp thủ công. Điều này giúp đảm bảo rằng các hệ thống luôn được bảo mật và hoạt động ổn định.

***Quản lý dịch vụ và tiến trình***: Puppet có thể tự động khởi động, dừng hoặc khởi động lại các dịch vụ trên máy chủ khi cần thiết. Điều này giúp đảm bảo rằng các dịch vụ quan trọng (như web server, database server, v.v.) luôn chạy đúng và không bị gián đoạn.

***Giám sát và báo cáo***: Puppet cung cấp các công cụ giám sát giúp theo dõi trạng thái của các hệ thống và dịch vụ. Các báo cáo chi tiết được tạo ra để người quản trị hệ thống có thể dễ dàng phát hiện lỗi hoặc các cấu hình không chính xác và khắc phục kịp thời.

***Tự động hóa bảo trì hệ thống***: Các công việc bảo trì định kỳ như sao lưu dữ liệu, xóa các tệp log cũ, kiểm tra tình trạng của hệ thống hoặc tối ưu hóa cơ sở dữ liệu có thể được tự động hóa bằng Puppet, giúp giảm thiểu công sức và đảm bảo rằng các công việc này luôn được thực hiện đúng hạn.

***Quản lý tài nguyên hệ thống***: Puppet giúp theo dõi và quản lý các tài nguyên hệ thống như bộ nhớ, ổ đĩa, và CPU để tối ưu hóa hiệu suất máy chủ. Puppet có thể tự động điều chỉnh các tham số cấu hình của hệ thống để duy trì hiệu suất ổn định và tránh các sự cố liên quan đến tài nguyên.

***Chạy các tác vụ định kỳ (cron jobs)***: Puppet có thể tạo và quản lý các cron jobs hoặc các tác vụ định kỳ, giúp tự động thực hiện các công việc bảo trì và xử lý tác vụ vào các thời điểm đã định.

***Quản lý tài nguyên hạ tầng phức tạp***: Trong các môi trường phức tạp, nơi có nhiều máy chủ và dịch vụ tương tác, Puppet giúp tự động hóa việc cấu hình và triển khai các tài nguyên, dịch vụ, hoặc các liên kết giữa các hệ thống. Điều này đảm bảo tính nhất quán và ổn định cho toàn bộ hạ tầng.

**Lợi ích của việc sử dụng Puppet:**

***Tiết kiệm thời gian và công sức***: Việc tự động hóa các tác vụ giúp giảm thiểu công việc thủ công và các thao tác lặp đi lặp lại.

***Giảm thiểu lỗi***: Với Puppet, các thao tác cấu hình được thực hiện tự động và đồng nhất, giúp giảm thiểu khả năng xảy ra lỗi do con người.

***Đảm bảo tính nhất quán***: Puppet đảm bảo rằng cấu hình của tất cả các máy chủ trong hệ thống luôn đồng nhất, bất kể số lượng máy chủ có lớn hay nhỏ.

***Dễ dàng mở rộng***: Puppet giúp quản lý và triển khai cấu hình trên số lượng lớn máy chủ mà không gặp phải các vấn đề về hiệu suất hay sự thay đổi cấu hình thủ công.

***Cải thiện hiệu suất và bảo mật***: Việc cập nhật tự động các bản vá bảo mật và cài đặt phần mềm mới giúp hệ thống luôn được bảo mật và hoạt động hiệu quả.

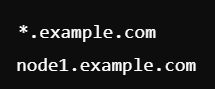
**7.3. Giám sát và báo cáo**

# Puppet không chỉ là một công cụ tự động hóa cấu hình mạnh mẽ, mà còn cung cấp các tính năng giám sát và báo cáo chi tiết về trạng thái của các máy chủ được quản lý. Tính năng này giúp quản trị viên nhanh chóng nhận biết các vấn đề phát sinh, theo dõi lịch sử thay đổi và đảm bảo rằng hệ thống luôn tuân thủ các chính sách đã đặt ra.

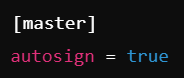
**Câu hỏi bổ sung:**

1. **Trong quá trình ký xác thực SSL certificate, một tổ chức gồm 100 máy trở lên, việc ký xác thực có thể tự động được hay không?**

* Puppet hỗ trợ tự động ký certificate thông qua tệp *autosign.conf*. Tệp này quy định các node (máy khách) nào sẽ được ký chứng chỉ mà không cần can thiệp thủ công từ Puppet Master.
* Thêm vào danh sách các tên hoặc mẫu hostname (dùng wildcard) của các node được phép tự động ký vào tệp */etc/puppetlabs/puppet/autosign.conf* trên Puppet Master.



* Kích hoạt autosign trong */etc/puppetlabs/puppet/puppet.conf*



* Puppet có API REST cho phép kiểm tra và ký yêu cầu chứng chỉ.
* Xem danh sách các yêu cầu chứng chỉ đang chờ ký.



* Ký tất cả các chứng chỉ đang chờ.



1. **Trong ngữ cảnh không tồn tại client nào thì dịch vụ Puppet có thể hoạt động hay không?**

Dịch vụ Puppet cần sự tồn tại của Puppet Agent để thực hiện việc quản lý cấu hình trên các máy khách (client). Nếu không có Puppet Agent, Puppet Server sẽ không thể trực tiếp thực thi các resource hoặc áp dụng catalog lên client. Tuy nhiên, có thể sử dụng các phương pháp gián tiếp để đạt được một số chức năng, ví dụ như kích hoạt Ansible từ Puppet Server để quản lý cấu hình.

**Chi tiết**

**Tầm Quan Trọng của Puppet Agent:**

* Puppet Agent đóng vai trò trung gian giữa Puppet Server (hoặc Puppet Master) và các máy client.
* Agent nhận catalog từ Puppet Server, sau đó thực thi các resource được khai báo trong catalog trên client.
* Nếu không có Puppet Agent, Puppet Server sẽ không có phương tiện để áp dụng catalog lên máy client, dẫn đến việc dịch vụ Puppet không thể thực hiện nhiệm vụ quản lý cấu hình một cách trực tiếp.

**Giải Pháp Thay Thế:**

Kích Hoạt Ansible Qua Puppet Server Trong trường hợp không thể sử dụng Puppet Agent, một phương pháp gián tiếp là sử dụng Puppet Server để kích hoạt Ansible – một công cụ quản lý cấu hình khác – nhằm thực thi lệnh trên các máy client. Quy trình bao gồm:

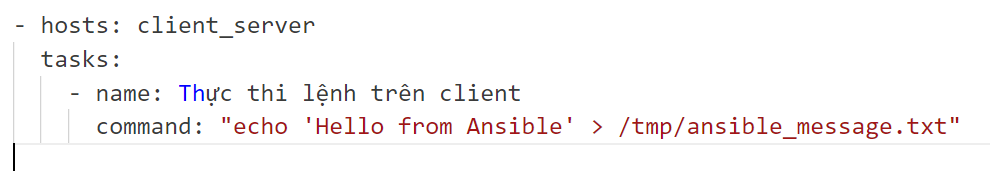
* Cấu hình Puppet Server để chạy một script shell: Tạo một manifest trong Puppet Server, khai báo resource exec để thực hiện script shell. Lệnh trong script sẽ gọi Ansible Playbook để áp dụng cấu hình lên client. Ví dụ manifest:



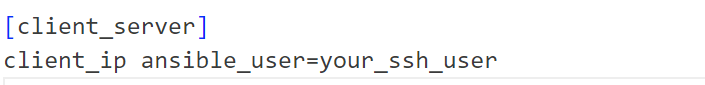
Sử dụng Ansible để thực thi lệnh:

Tạo một Ansible Playbook và file inventory để kết nối với máy client qua SSH, ví dụ:

* File Playbook:



* File Inventory:



* **Kết quả:** Puppet Server sẽ thực hiện script shell, script này kích hoạt Ansible để thực thi lệnh trên client.

**Những Lưu Ý:**

* Phương pháp này chỉ gián tiếp giải quyết vấn đề, vì Ansible là công cụ chính thực hiện việc quản lý cấu hình trên client, không phải Puppet.
* Cần đảm bảo Ansible được cài đặt và cấu hình đúng trên Puppet Server.
* Cần có kết nối SSH giữa Puppet Server và client.

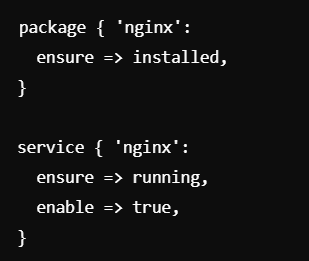
**Kết Luận:**

Nếu không có Puppet Agent, dịch vụ Puppet không thể hoạt động như một hệ thống quản lý cấu hình toàn diện. Tuy nhiên, có thể áp dụng phương pháp gián tiếp, như sử dụng Puppet Server để kích hoạt Ansible, nhằm thực hiện một số tác vụ quản lý trên client. Để đảm bảo hiệu quả, cần cân nhắc kỹ lưỡng về kiến trúc hệ thống và mục tiêu quản lý cấu hình trước khi triển khai.

1. **Với cùng một file cấu hình được viết trên máy chỉ Puppet - master thì có thể dùng để cấu hình trên các hệ điều hành khác hay không (ví dụ trên Centos, Window, Ubuntu…)?**

* Puppet được thiết kế để hỗ trợ đa nền tảng:
* Về file cấu hình: Thường có thể dùng chung mà không cần thay đổi, miễn các tham số như server, cername, environment đúng cho hệ thống.
* Puppet sử dụng cú pháp Ruby-based nên các manifests và modules thường hoạt động xuyên suốt trên các hệ điều hành.
* Tuy nhiên dù dùng chung cấu hình nhưng cần tùy chỉnh thêm đối với mỗi hệ điều hành:
* Linux: Các file cấu hình về Puppet sẽ lưu ở đường dẫn ***/etc/puppetlabs/puppet/puppet.conf*** , SSL: ***/etc/puppetlabs/puppet/ssl.***
* Windows: Các file cấu hình về Puppet sẽ lưu ở đường dẫn ***C:\ProgramData\PuppetLabs\puppet\etc\puppet.conf,*** SSL: ***C:\ProgramData\PuppetLabs\puppet\etc\ssl***
* Mỗi hệ điều hành sử dụng công cụ quản lý gói khác nhau:
* **Ubuntu/Debian**: apt.
* **CentOS/RHEL**: yum hoặc dnf.
* **Windows**: Chạy MSI installer hoặc chocolatey.

Việc này puppet sẽ xử lý bằng cách khai báo sử dụng các resource tiêu chuẩn như **package** hoặc **service**.



***Tài liệu tham khảo***

1. [*Installing Puppet*](https://www.puppet.com/docs/puppet/7/install_puppet#enable_the_puppet_platform_repository)
2. [*What is Puppet? | How Puppet Works? | Puppet Tutorial For Beginners | DevOps Tools | Simplilearn*](https://www.youtube.com/watch?v=llcjg1R0DdM&list=PLEiEAq2VkUULKeFdfrw25YUayg0omb9wK)
3. [*Puppet Tutorial for Beginners: What is Puppet & How to Use?*](https://www.guru99.com/puppet-tutorial.html)