🏠 / Infrastructure and Automation / Infrastructure and Automation Summary

Infrastructure and Automation Summary

7.8.1

What Did I Learn in this Module?

🔖

Automating Infrastructure with Cisco

Automation is using code to configure, deploy, and manage applications together with the compute, storage, and network infrastructures and services on which they run. You have choices in how to programmatically control your network configurations and infrastructure. Walk: Read-only automation. Run: Activate policies and provide self-service across multiple domains. Fly: Deploy applications, network configurations, and more through CI/CD. Manual processes are always subject to human error, and documentation meant for humans is often incomplete and ambiguous, hard to test, and quickly outdated. Automation is the answer to these problems. Benefits of full-stack automation are self service, scale on demand, observability, and Automated problem mitigation. Software-defined infrastructure, also known as cloud computing, lets developers and operators use software to requisition, configure, deploy, and manage bare-metal and virtualized compute, storage, and network resources. Modern application architectures are increasingly distributed. They are built up out of small and relatively light components that are sometimes called microservices.

DevOps and SRE

For full-stack automation to be truly effective, it requires changes to organizational culture, including breaking down the historical divides between Development (Dev) and Operations (Ops). DevOps evolved and continues to evolve in many places in parallel. Some key events have shaped the discipline as we know it today.

Defining Moments 1: Site Reliability Engineering (SRE). The role of the SRE is intended to fuse the disciplines and skills of Dev and Ops.

Defining Moments 2: Debois and "Agile Infrastructure" Debois was a proponent of automating virtual infrastructure, using version-control to store infrastructure deployment code, and applying Agile methods to the development and maintenance of infrastructure-level solutions.

Defining Moments 3: Allspaw and Hammond

John Allspaw and Paul Hammond gave a presentation at VelocityConf in 2009. It described automation, teamwork, responsibility-sharing, transparency, trust, mutual accountability, and communications practices.

DevOps/SRE have many core principles and best practices:

- A focus on automation
- The idea that "failure is normal"
- A reframing of "availability" in terms of what a business can tolerate

Basic Automation Scripting

Automation tooling partly works by wrapping shell functionality, operating system utilities, API functions and other control plane elements. But tools still don't solve every problem of deployment and configuration. That's why every automation tool has one or more functions that execute basic commands and scripts on targets and return results. For example, in Ansible, these functions include command, shell, and raw. Automation tools like Ansible, Puppet, or Chef offer powerful capabilities compared to ad-hoc automation strategies using BASH, Python, or other programming languages. An imperative procedure is an ordered sequence of commands aimed at achieving a goal. The sequence may include flow-control, conditions, functional structure, classes, and more. To configure remote systems, you need to access and execute scripts on them. Two ways (of several) to do this are: store scripts locally, transmit them to target machines with a shell utility like `scp`, then log into the remote machine using `ssh` and execute them. Or, you can pipe scripts to a remote machine using `cat | ssh` and execute them in sequence with other commands, capturing and returning results to your terminal, all in one command. Infrastructure-as-a-Service (IaaS) cloud computing frameworks are a typical target for automation. Cloud automation enables you to provision virtualized hosts, configure virtual networks and other connectivity, requisition services, and then deploy applications on this infrastructure. IaaS and other types of infrastructure cloud also provide CLIs and SDKs that enable easy connection to their underlying interfaces, which are usually REST-based.

Automation Tools

Three of the most popular automation tools are Ansible, Puppet, and Chef. Automation tools like Ansible, Puppet, or Chef offer powerful capabilities compared to ad-hoc automation strategies using BASH, Python, or other programming languages. Idempotent software produces the same desirable result each time that it is run. In deployment software, idempotency enables convergence and composability. Procedural code can achieve idempotency, but many infrastructure management, deployment, and orchestration tools have adopted another method, which is creating a declarative. A declarative is static model that represents the desired end product.

Ansible's basic architecture is very simple and lightweight. Ansible's control node runs on virtually any Linux machine running Python 2 or 3. All system updates are performed on the control node. Plugins enable Ansible to gather facts from and perform operations on infrastructure that can't run Python locally, such as cloud provider REST interfaces. Ansible is substantially managed from the Bash command line, with automation code developed and maintained using any standard text editor.

Puppet's core architecture has the following characteristics: A designated server to host main application components called the Puppet Server, the Facter which is the fact-gathering service, the PuppetDB, which can store facts, node catalogs, and recent configuration event history, and a secure client, a Puppet Agent, installed and configured on target machines. Operators communicate with the Puppet Server largely via SSH and the command line.

Chef's main components are the Chef Workstation which is a standalone operator workstation, the Chef Infra Client (the host agent) which runs on hosts and retrieves configuration templates and implemenst required changes, and the Chef Infra Server which replies to queries from Chef Infra

Agents on validated hosts and responds with configuration updates, upon which the Agents then converge host configuration.

Infrastructure as Code

Immutability literally means "the state of being unchangeable," but in DevOps parlance, it refers to maintaining systems entirely as code, performing no manual operations on them at all. Committing to immutability enables you to treat your automation codebase the way you would any application code:

● You can trust that the codebase describes what's actually running on bare metal or cloud servers.

● You can manage the codebase Agile procedures and structured use of version control to keep things clear and simple.

Automating Testing

DevOps typically needs more fine-grained ways to define and implement infrastructures, certify that deployed infrastructures are working as required, proactively ensure its smooth operations, preemptively take action when failures are imminent, and find and fix issues when errors occur.

When you use unit-testing tools like pytest in tandem with higher-order automation and in concert with continuous delivery (CI/CD), you can build environments where code can be automatically tested when changes are made.

Unit-testing frameworks make tests a part of your codebase, following the code through developer commits, pull requests, and code-review gates to QA/test and Production. This is especially useful in test-driven development (TDD) environments, where writing tests is a continuous process that actually leads development, automatically encouraging very high levels of test coverage.

Network Simulation

Network simulation provides a means to test network configurations, debug configuration code, and to work with and learn Cisco infrastructure and APIs in a safe, convenient, and non-cost-prohibitive way.

Cisco Virtual Internet Routing Laboratory (VIRL) VIRL can run on bare metal, or on large virtual machines on several hypervisor platforms.

7.8.2

Module 7: Introduction to Automating, Developing, and Deploying Applications with Cisco Quiz

1. What is the average cost per hour of an enterprise outage for the largest enterprises?

> ⚠ Topic 7.1.0 – According to the Gartner report on 2014, if a system outage occurs, the average cost per hour for a large enterprise outage is $300,000.00 per hour. Much of the outage time will include manual configuration tasks that can take an extensive amount of time and may incur an increased risk of misconfiguration.

○ $700,000.00 per hour

○ $100,000.00 per hour

○ $500,000.00 per hour

○ $300,000.00 per hour

2. What is idempotency in IT automation?

> ⚠ Topic 7.3.0 – Idempotency is a recurring theme in automation and it refers to the quality of software.

○ the scripting of shell commands

○ the quality of software

○ the reusing of shell commands

○ the security level of software

3. A student is creating an Ansible playbook to practice management automation of a Cisco router. Which command should the student add to the playbook to find the interface names, assigned IP addresses, and status of interfaces on the router?

> ⚠ Topic 7.4.0 – The **show ip interface brief** command can be used to verify the IP address configured on the router. The **show running-config** command can be used to see how IP addresses are assigned to the interfaces; however, if an interface obtains the IP address from the DHCP server, the running configuration will not show the actual IP address assigned. The **ip address** command is used in a Linux host to display information about network interfaces.

○ **show running-config**

○ **ip address**

○ **show ip interface brief**

○ **show ip route**

4. When following the best practices and core principles of DevOps, how is failure considered?

> ⚠ Topic 7.2.0 – One of the many core principles and best practices of DevOps/SRE is that failure is normal. There is an assumption that failures will occur and that these occurrences will influence software design methodology.

○ Failure is normal.

○ Failure should only occur in Ops.

○ Failure is simply "repair work".

○ Failure should not influence software design methodology.

5. What is a canary deployment?

> ⚠️ Topic 7.5.0 – A canary deployment occurs when a project is deployed to a small fraction of users in order to reduce risk and improve the quality of feedback.

○ a production test using a chaos tool to inject random failures with a limited blast-radius

○ a server-by-server switchover to a new release deployment

○ the use of two identical servers with load balancers to redirect traffic for an entire server as needed

○ the switchover of selected users to a new release deployment

6. What are two challenges of cloud computing technologies? (Choose two.)

> ⚠️ Topic 7.1.0 – The challenges when using cloud technologies include the following:
> ○ Cloud permissions can be challenging to manage and controlling permissions and access is critical.
> ○ Private and public cloud frameworks can cause confusion due to varying UIs, APIs, and quirks.
> ○ The consumption of cloud resources can be hard to track and unexpected costs may occur and be difficult to calculate.

☐ utilizing generic host environments

☐ requesting additional resources

☐ managing costs

☐ maintaining access control

☐ maintaining configurational consistency of platforms

≡   CISCO DevNet Associate [v1.0]

> ⚠️ Topic 7.7.0 – Cisco VIRL is an ideal tool for learning and simulating networks and it can run on bare metal or on large virtual machines on several hypervisor platforms.

○ VIRL

○ Git

○ Puppet

○ Ansible

8. An analyst is using Ansible to perform remote management tasks on network devices. Which file type is in the Ansible file structure that describes the devices under management?

> ⚠ Topic 7.4.0 - Ansible uses an inventory file, also called hosts file, that contains device information used by Ansible playbooks.

○ inventory file

○ playbook file

○ variable file

○ role folders and files

9. What is an error budget in the DevOps/SRE culture?

> ⚠ Topic 7.2.0 - Error budgets define the number of allowable errors per unit time and are also used as a gate on release speed.

○ It is the planned percentage of a monetary contract that is given back to a customer when an SLO is not met.

○ It is a best-practice metric for availability, equivalent to only six minutes of unplanned offline time per year.

○ It is an internal-only metric for calculating the actual availability of a service.

○ It is a method for measuring allowable errors as a gate on release speed.

10. Which programming language is used by Chef to execute the recipes?

> ⚠ Topic 7.4.0 - Within a repo, code is organized in "cookbook" folders, comprising "recipes" (actual linear configuration code, written in the extended Ruby syntax of Chef).

○ Python

○ Javascript

○ Ruby

○ Bash

11. A developer issues a Linux command **python3 -m venv devenv**. What is the developer trying to achieve?

> ⚠ Topic 7.6.0 - The syntax for creating a Python 3 virtual environment in a Linux system is **python3 -m venv** *virtual_environment*, where **venv** is the Python 3 module to create a virtual environment and **-m** is the switch to tell Python 3 the venv module is to be used.

○ to install the devnet tool sets in the Python 3 virtual environment named venv

○ to activate the Python 3 virtual environment named devenv

○

○ to create a Python 3 virtual environment named devenv

○ to enter the Python 3 virtual environment named venv using the devnet tool sets

Check   **Show Me**

Reset