INSTITUTE of TECHNOLOGY CARLOW

Institiúid Teicneolaíochta Cheatharlach

April 30th, 2021

AUTOMATION OF NETWORK/SERVER SECURITY PROVISIONING USING DEVOPS TOOLS.

RESEARCH MANUAL

STUDENT: KATIE BROPHY C00224531
SUPERVISOR: JAMES EGAN

TABLE OF CONTENS

INTRODUCTION

DevOps is a set of procedures that blends software development and IT operations It aspires to reduce the systems development life cycle and provide continuous delivery with high software quality.[DevOps, 2020] Configuration management is the practice of sustaining and creating the performance of the products by maintaining its physical elements, functional elements, design, requirements, and operational information throughout its life cycle

Manual creation, configuration and management of servers and network infrastructure is time consuming, repetitive, complex and prone to errors. Using DevOps tools to automate the process reduces errors, speeds up provisioning and reduces costs. Playbooks or Recipes, seen in Ansible and Chef, can achieve this automation and are universal, applicable to a small business or a huge organization alike.

This Research Manual will provide an overview of the automation of network/server security provisioning using available DevOps tools, and look at developing playbooks that create, configure and update both servers and network infrastructure where necessary.

OVERVIEW OF AREAS, TECHNOLOGIES OR TOPICS RESEARCHED

CHEF
What is Chef?

Chef is an open source tool from Opscode for configuration management. There are also paid versions like Chef Enterprise. It is written in the language Ruby and Erlang and allows severs to be configured and maintained automatically by defining infrastructure as code. [Chef Tutorial: Components and Configuration Management, and More Explained, 2020]

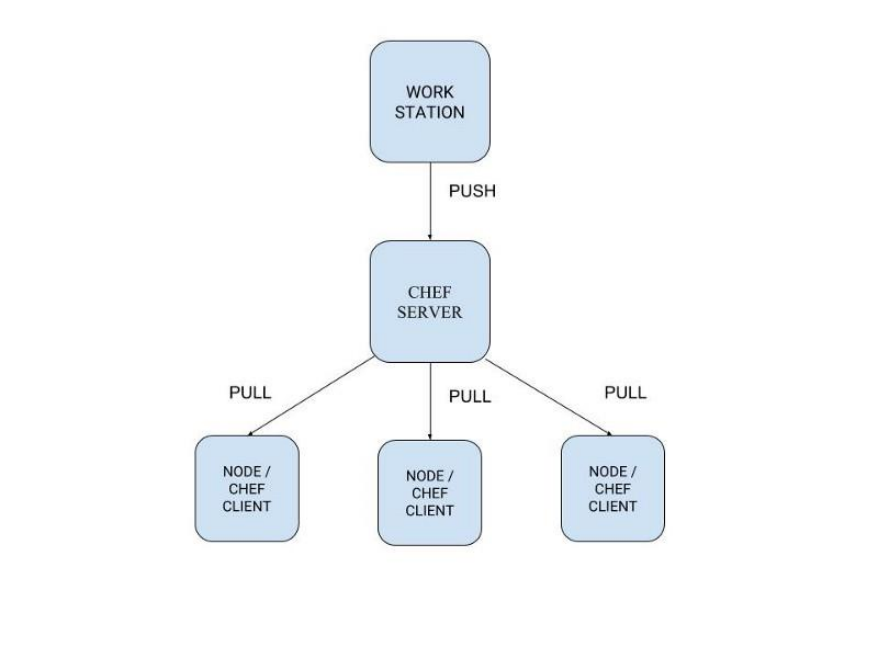
Configuration management

Configuration management is a set of engineering methods that delivers an efficient way to manage all the units needed for effective deployment. The 3 main units are code, written by system administrators to configure systems, infrastructure being the systems and servers, and the teams that maintain the infrastructure.

Whenever the infrastructure needs to be updated, with new configuration or a newer version of the software or operating system the code needs to be changed first; the configuration of the network changes with the requirements of the enterprise and with the ever changing cycle it’s important the team is coordinated.

There are two forms of configuration management: push configuration and pull configuration.

Push configuration requires the server to push the configuration to the nodes. Pull configuration entails the nodes ‘checking in’ with the server intermittently and retrieving the configurations from it. The Chef model supports pull configuration. [Chef Tutorial: Components and Configuration Management, and More Explained, 2020]



[Puri, 2020]

Infrastructure as code

Infrastructure as code is a form of IT infrastructure is when your policies and configurations are written as code allowing the automation of configuration management and other processes. The code is easily modified and deployed. [Chef Tutorial: Components and Configuration Management, and More Explained, 2020]

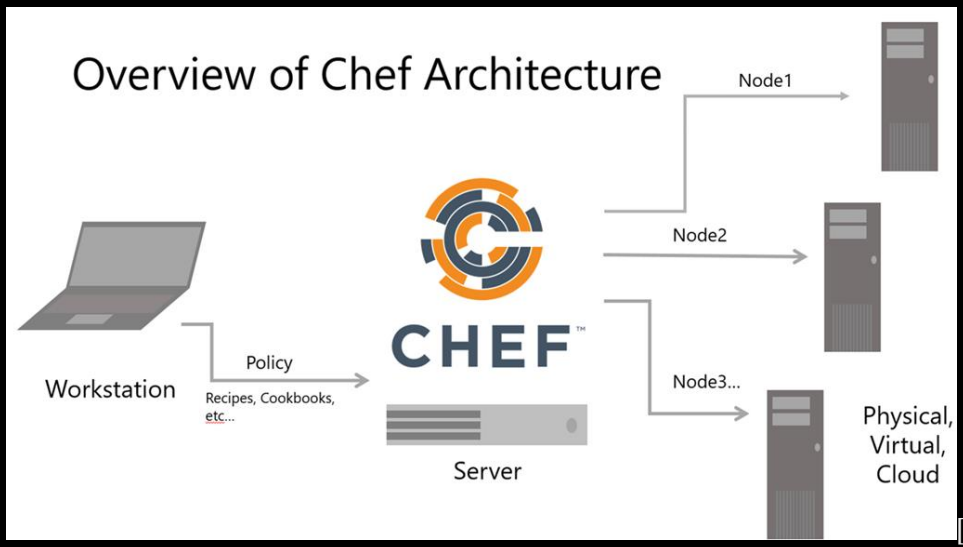
Chef Components

The first Chef component is the workstation. This is the system where the admin sits. The system produces the code, called a recipe, for the configuration and management of the infrastructure. A collection of recipes are known as a cookbook and they are uploaded to servers with the ‘knife’ command. [Puri, 2020]

The 2nd Chef module is the server where the cookbooks are stored. The servers act as a mediator between the workstation and the nodes. The server can be hosted locally or remotely and delivers the tools needed to run the node configurations.

The last Chef component are the nodes. In a Chef architecture, you can have several nodes that collect all the data about current state of the nodes and then send it to the server where its compared to the configuration files to see if any new configurations are needed. The Chef client service sits on the nodes and carries out all communication with the server, when a node needs a new recipe it communicates this to the

server. You can have many nodes in a Chef architecture, and they can each be configured differently if needed. [Chef Tutorial: Components and Configuration Management, and More Explained, 2020]



[What is Chef | Automate Infrastructure Configuration, 2020]

Flavours Of Chef:

Chef is available in many 'flavours':

- Chef Solo has no remote servers and cookbooks are stored locally.
- Hosted Chef, where a server is given as a service in the cloud and no manual set up is needed
- Chef Client/Server where a hosted remote server carries out communication between the workstation and the nodes.
- Private Chef/ Chef Enterprise where the server is hosted internally in the enterprise infrastructure.

Why Should We Use Chef?

Many companies manually configure and maintain the infrastructure, this is a complex and time-consuming process that is prone to errors and system failures. Chef allows organisations to automatically deploy configurations and features instantly and has testing ability to check the code and remediate any errors before its deployed

PUPPET

What is Puppet in DevOps

The puppet is a tool used in system management that automates and centralises the configuration management cycle. It can be used for software deployment also. It aids the configuration management and

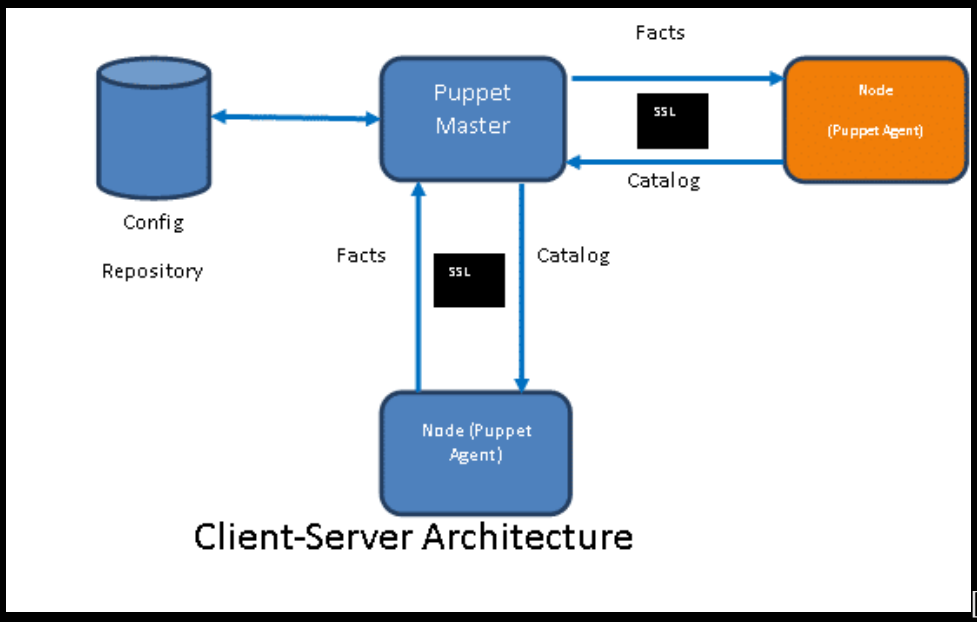
deployment of servers and the orchestration of applications on the infrastructure in an enterprise. [What is a Puppet in DevOps? | Why Puppet Software for DevOps?, 2020]

Puppet performs the following,

- Utilise distinct configurations for each node
- continuous monitoring of the server for the configurations and if changes are detected it automatically changes to a pre-defined configuration on the nodes.
- Automates deployment for all applications on the servers.
- It executes Infrastructure as a Code

Functionality of Puppet

Puppet uses the declarative Domain Specific Language [DSL] to define the configuration elements to the infrastructure. It uses a function called facter to look for information about the systems. The puppet agent collects the configurations of each node and sends it to the puppet master. Then a Configuration file is compiled detailing how each node will be configured. This is deployed so that node employs the configuration and self-updates. Puppet agent uses the pull configuration model to gather the data from the puppet master. [What is a Puppet in DevOps? | Why Puppet Software for DevOps?, 2020]



[Sheet and Sheet, 2020]

Puppet automates configuration management using the following steps:

- The puppet agents running on the nodes uses the pull configuration to collect information about itself.
- The data is sent to the puppet master
- The puppet master assembles a configuration file for the node and sends it to the puppet agent
- The puppet agent configures itself and reports back to the puppet master.

The puppet agent is typically operated as a root user or the user who has been given the required privileges to configure the nodes. The puppet agent obtains the communication privileges from the puppet master by requesting a Secure Socket Layer [SSL] certificate for the first time. [What is a Puppet in DevOps? | Why Puppet Software for DevOps?, 2020]

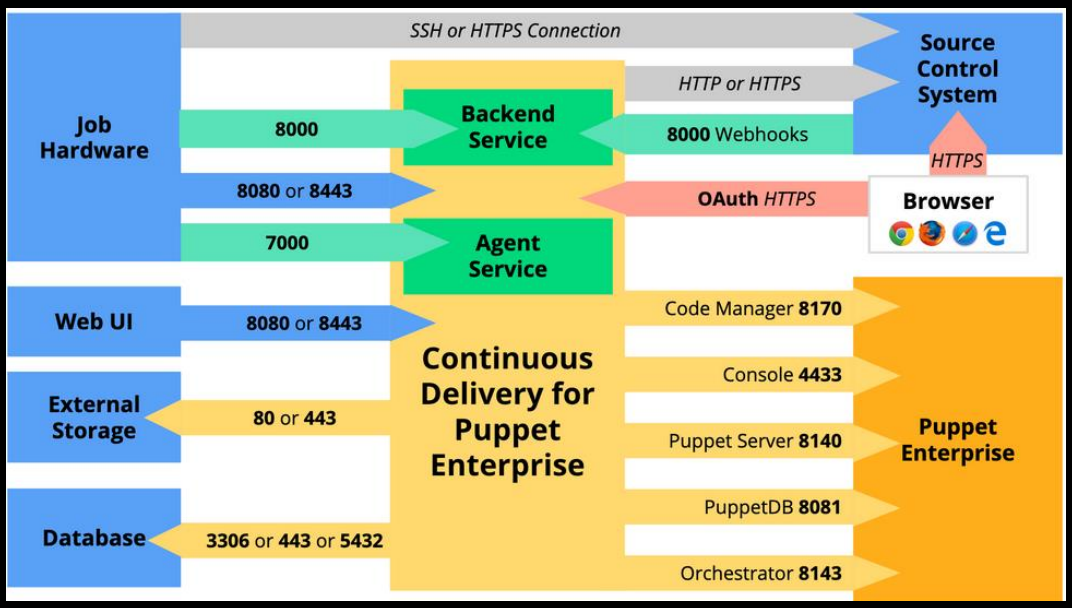
The puppet user securely gathers the information from the module that the puppet master needs. Via the puppet user, the puppet master can:

- Generate catalogues,
- Validate the SSL certificate from the puppet agent
- Store configuration files in the directory
- Send files to the puppet agents

Puppet encryption and communication is achieved by employing the SSL/TLS protocols. This encrypts all traffic between the odes and the puppet master and uses the SHA-256 hash by default. It is also used to authenticate puppet agents and puppet masters [What is a Puppet in DevOps? | Why Puppet Software for DevOps?, 2020]

Why use Puppet

Puppet provides quick and simple configuration management, cutting the latency and errors of manual process and allows for continuous delivery.

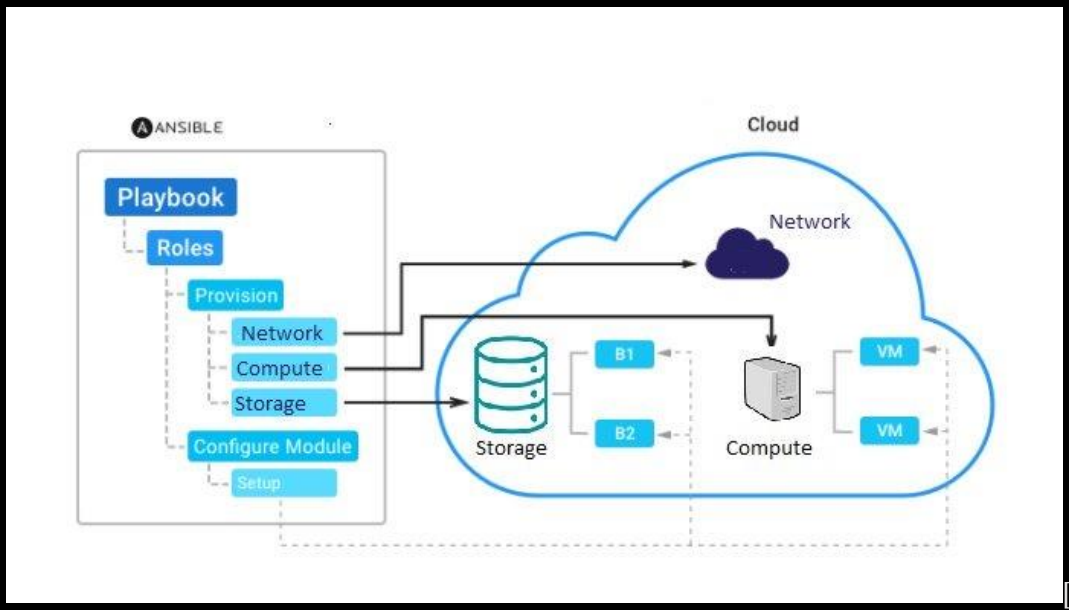


[Anon, 2020]

ANSIBLE
Automation of Provisioning

From conventional bare metal through to serverless or function-as-a-service, when trying to automate the operational life cycle of your applications the first step needs to be automating the provisioning of any

infrastructure. Ansible can provision the latest cloud platforms, virtualized hosts and hypervisors, network devices and bare-metal servers. [Ansible, 2020]



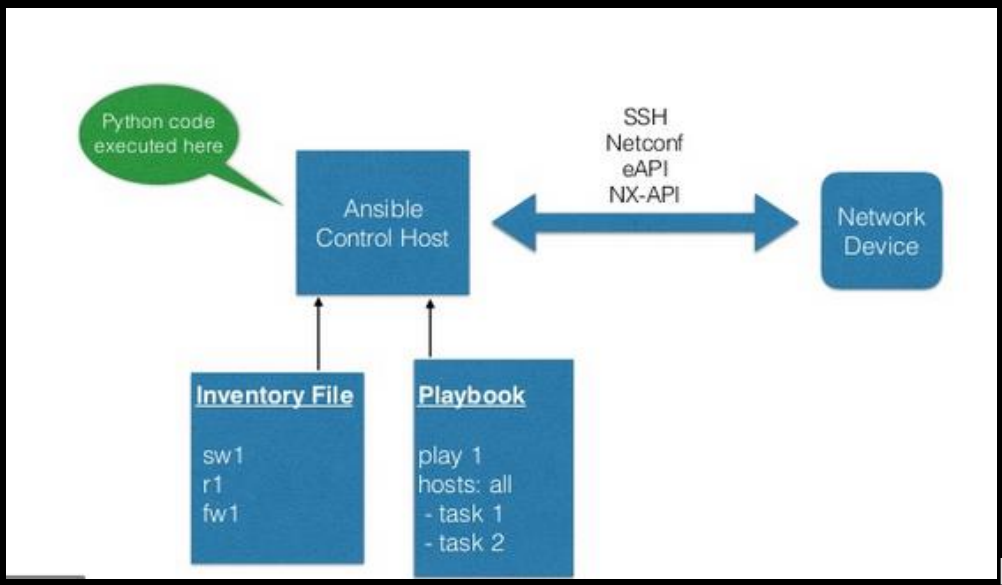
[Kolappan, 2020]

After Bootstrapping, many operational tasks can be completed separately including security patching nodes, connection to storage or added to a load balancer. Essentially The automation tool becomes the connecting point in any of the process pipelines, going from bare infrastructure to daily management automatically. Ansible uses a simple human readable language to allow seamless evolution into configuration management, orchestration and application deployment. [Ansible, 2020]

Ansible provisioning playbooks exists for bare metal infrastructure to both call and perform the provisioning steps required. They also exist for virtualized infrastructure. Virtual networks and storage are being used more and more, the cross-platform management can be simple and flexible using the automation playbooks to provision them. Multiple kinds of storage options are also supported with Ansible including cloud based, software defined and hardware appliances. [Ansible, 2020]

Automating Networks

While networks are a fundamental part of an IT organization, real automation of the network stack is not really used. Most enterprises attempt to automate their network, they use complex tools that are specific to certain commercial vendors. These processes must be managed manually, which is time consuming and results in reduced dexterity among teams; this also isolates networking teams from the evolving DevOps transformation slowing the development of IT departments and enterprises. [Ansible, 2020]



[Network Automation with Ansible, 2020]

Bringing automation into networks will provide:

- Automation of configuring the network stack from system to access to core services
- Testing and validating the current network state
- Continuous compliance to check for network configuration drift

Automating Configuration Management

Most system's configuration management is either handled manually with a combination of numerous scripts and impromptu practices carefully selected by the administrators, or automatically with complex high maintenance automation frameworks. This poses a challenge that is growing alongside the virtualization of systems and the complex cross-platform management that requires. [Ansible, 2020]

Ansible offers a simple solution to this issue; its automation playbooks are accessible, consistent and secure. The configurations are human readable and machine-parsable, requiring minimal training for administrators, IT managers and developers. Only a SSH key or password are needed to use the automation playbooks and they do not require installation of any other software meaning very little management is needed. The simplicity allows for quicker turnaround of the management cycles allowing for more availability to focus on more crucial and tactical work. [Ansible, 2020]

Ansible includes a state-driven source mode that defines the preferred state of computer systems and services, instead of the paths needed to achieve this state. Regardless of the current state of a system, ansible will understand how to configure the preferred state described. This provides a consistent, replicable solution and reduces failures seen in script-based configuration. Ansible is provides secure configurations. It uses OpenSSH for its default transport layer and does not require any 3rd party software or root login privileges meaning it has a low attack surface and is easily deployed [Ansible, 2020]

Automating Security

Firewalls regulate what traffic can pass through from one network to another, shielding applications that internet or intranet facing. Ansible can employ policies and log configuration, allowing a quicker turnaround for investigations and remediation. Intrusion detection & prevention systems [IDPS] monitor network traffic for suspect activity and broadcast alerts and prevent attacks when a recognised attack pattern is detected. Ansible simplifies rule and log administration, resulting in more proficient security operations. Security information and event management [SIEM] systems gather and examine security events aid in detection and response to incidents. Ansible provides security analysts with wide range of data sources to help better assess incidents. Privileged Access Management [PAM] tools monitor and oversee privileged accounts and access, Ansible simplifies the alternation and managing of privileged authorisations to automatically prevent and remediate high-risk activities. [Ansible, 2020]

Ansible automatically adjusts the degree of logging, creates new IDS rules and firewall policies to allow analysts detect a higher volume of threats with less turnaround. It allows for faster remediation to incidents by automating the blacklisting of known malicious Ip addresses and whitelisting approved network traffic and isolating suspect workloads.

Automating Security Operations allow for:

- Configuration a sequence of jobs that share inventory, playbooks or permissions to fully automate investigations
- Centralising job access and execution and role provisioning/building by pairing user directory with infrastructure
- classify patterns, analyse infrastructure events, monitor glitches, and compare unrelated events, by integrating with 3rd party external log cluster services [Ansible, 2020]

CONCLUSIVE COMPARISON

Category	Chef	Puppet	Ansible
Availability	If a failure occurs on the main server, the chef server, there is a secondary server to replace the primary server	Puppet has 'multi-master architecture', when the current running master fails, the other master replaces it.	It runs with a single active node, called the Primary instance. If the primary node fails, a Secondary instance available to replace it.
Ease of setup	Chef has a master-agent architecture. *Chef server* runs on the master machine and Chef client runs as an agent on each client machine. There is another element called the workstation, this is where configurations are stored and tested before being pushed to the chef server. This means that the setup can be relatively complex.	Also uses a master-agent architecture. there is also a certificate signing between the agent and the master meaning the setup is also quite complex	Does have the master operating on the server machine but does not have any agents operating on the client machines, instead it uses SSH to login to the client's system so you can configure them. This is quicker and easier to setup
Management	Uses Ruby DSL for the configuration so configuration management requires a programmer. The client machines pull	Hardest to manage configurations on because it uses Puppet DSL, an original language. The management is very system system-administrator based and allows for non-	Easy to learn to manage the configurations as it uses YAML a human-readable language similar to the English language, so it is quick and easy to

	configurations from the chef server	immediate remote execution. Also uses a pull configuration model	learn. Uses the push configuration model. Allows for immediate remote execution and real-time use.
Scalability	The use of Ruby means it requires programmers to use and needs extensive learning meaning it is not easily used and applied. Low scalability.	Like Chef it uses its own original DSL, so vast learning for configurations but it is aimed to be used by system administrators. Still not very scalable	Ansible uses Python, it is easy to learn and it is aimed to be used by system administrators. Python is integral in most Linux and Unix implementations, allowing a quick and easy set up. It is high scalability.
Configuration Language	Ruby DSL	Puppet DSL	YAML [Python]
Interoperability	Chef Server must be on Linux or Unix, but workstation and Chef Client can be used on windows.	Puppet Master must be on Linux or Unix, but Puppet Agent can be used on windows	Ansible Server must be on Linux or Unix, but Clients and nodes can be used on windows

[Johari, 2020]

From the comparison above, it became clear that basing my design off the model seen in Ansible would be the best approach. It will allow for automation that is easy setup and manage, highly scalable and accessible.

WEB FRAMEWORKS FOR GUI

Django is a web framework that is compatible with any website, format of content and client-side framework. It has automatic security functionality that allow for identity access management, clickjacking, cross-site request forgery, Cross-site scripting and SQL injection. The design principles are to recycle and maintain the code making it easily manageable; each component in Django is stand-alone, allowing the framework to have a high scalability. It is written in Python. ["Security in Django | Django Documentation | Django"]

Flask is a micro web framework that is flexible, simple and allows full developer control of the application. It is inbuilt with a logical API and development server and supports unit testing. ["Flask vs Django in 2020: Which Framework to Choose?"]

Django	Flask
Full stack framework	Light-weight minimalistic framework
Includes common web application formatting and template, allowing for easy site building and best practice	Developers have more control over design and base of the application and can utilise any plugins and libraries
Inbuilt framework for Administrator access	No Admin feature included
Does not work well with applications that has evolving requirements. However more suitable when starting complex projects that have high functionality	Easily adaptable and scalable for adding functionality
Requires significantly more code to implement functions	Needs much less code compered to Django for the same functionality
Significant learning curve	Only needs basic programming skills

After comparing the two, I believe Django will be a more suitable front end for my application.

CONCLUSION

After researching multiple DevOps automation tools, I believe that I can create an application to automate the creation, configuration and maintenance of both servers and network infrastructure allowing me to automate their security. I will be basing my work off Ansible's automation model.

BIBLIOGRAPHY

[DevOps,2020]	En.wikipedia.org. 2020. *Devops*. [online] Available at: <https://en.wikipedia.org/wiki/DevOps#cite_note-loukides-2012-2> [Accessed 13 November 2020].
[Chef Tutorial: Components and Configuration Management, and More Explained, 2020]	Simplilearn.com. 2020. *Chef Tutorial: Components And Configuration Management, And More Explained*. [online] Available at: <https://www.simplilearn.com/chef-tutorial-article> [Accessed 12 November 2020].
[Puri, 2020]	Puri, M., 2020. *A Complete Beginner'S Guide To Chef And Infrastructure As Code*. [online] freeCodeCamp.org. Available at: <https://www.freecodecamp.org/news/an-introduction-to-chef-and-infrastructure-as-code-7d8ad2689b8/> [Accessed 12 November 2020].
[What is Chef | Automate Infrastructure Configuration, 2020]	Apachebooster Blog: *What Is Chef | Automate Infrastructure Configuration*. [online] Available at: <https://apachebooster.com/blog/automate-infrastructure-configuration/> [Accessed 12 November 2020]. Showcasing the tech blogs written by our writers. 2020.
[What is a Puppet in DevOps? | Why Puppet Software for DevOps?, 2020]	Staragile.com. 2020. *What Is A Puppet In Devops? | Why Puppet Software For Devops?*. [online] Available at: <https://staragile.com/blog/puppet-in-devops> [Accessed 12 November 2020].
[Sheet and Sheet, 2020]	Sheet, P. and Sheet, P., 2020. *Puppet Cheat Sheet - Download In PDF & JPG Format - Intellipaat*. [online] Intellipaat Blog. Available at: <https://intellipaat.com/blog/tutorial/devops-tutorial/puppet-cheat-sheet/> [Accessed 12 November 2020].
[Anon, 2020]	Puppet.com. 2020. [online] Available at: <https://puppet.com/docs/continuous-delivery/3.x/cd_architecture.html> [Accessed 12 November 2020].
[Ansible, 2020]	Ansible, R., 2020. *Ansible For Provisioning*. [online] Ansible.com. Available at: <https://www.ansible.com/use-cases/provisioning> [Accessed 12 November 2020].
[Kolappan, 2020]	Kolappan, K., 2020. *Automating Infrastructure Deployments [IAAS] In AWS Cloud With Ansible*. [online] Oneglobesystems.com. Available at: <https://www.oneglobesystems.com/blog/automating-infrastructure-deployments-iaas-in-aws-cloud-with-ansible> [Accessed 12 November 2020].
Ansible, 2020	Ansible, R., 2020. *Red Hat Ansible Network Automation - Red Hat Ansible*. [online] Ansible.com. Available at: <https://www.ansible.com/use-cases/network-automation> [Accessed 12 November 2020].
[Network Automation with Ansible, 2020]	Slideshare.net. 2020. *Network Automation With Ansible*. [online] Available at: <https://www.slideshare.net/atarsha/network-automation-with-ansible-banog-meetup> [Accessed 12 November 2020].
[Ansible, 2020]	Ansible, R., 2020. *Ansible For Configuration Management*. [online] Ansible.com. Available at: <https://www.ansible.com/use-cases/configuration-management> [Accessed 12 November 2020].

Ansible, 2020]	Ansible, R., 2020. *Red Hat Ansible	Security Automation*. [online] Ansible.com. Available at: <https://www.ansible.com/use-cases/security-automation> [Accessed 12 November 2020].
[Johari, 2020]	Johari, A., 2020. *Chef Vs Puppet Vs Ansible Vs Saltstack: Which One To Choose	Edureka*. [online] Edureka. Available at: <https://www.edureka.co/blog/chef-vs-puppet-vs-ansible-vs-saltstack/> [Accessed 12 November 2020].
[“Security in Django | Django Documentation | Django”]	Security in Django | Django Documentation | Django.” Docs.Djangoproject.com, docs.djangoproject.com/en/3.1/topics/security/. Accessed 12 Nov. 2020.
[“Flask vs Django in 2020: Which Framework to Choose?”]	“Flask vs Django in 2020: Which Framework to Choose?” Hackr.Io, hackr.io/blog/flask-vs-django. Accessed 13 Nov. 2020.