NATILIK

Wayne Enterprises

Natilik Terraform Assessment

August 2023 version 1.0

Contents

1 Document Control

1.1 Revision History

Revision History			
Date	Version	Author	Summary of change
21/7/2023	1.0	Bryn Lewis	First release

1.2 Distribution

Client			
Name	Role	Email	Location
Bruce Wayne	Head of Networks	bruce.wayne@wayneenterprises.com	Gotham, USA

Natilik			
Name	Role	Contact details	Location
Bryn Lewis	DevOps Engineer	blewis@natilik.com	London, UK

1.3 About Natilik

Natilik is your technology solutions partner and your confident guide to digital transformation.

Your digital transformation journey is unique to you and your business and at Natilik, the team has a range of solutions that can help you to achieve the outcome you are trying to reach. With solutions including Digital Infrastructure, Workplace Evolution, Customer Experience, Application Performance and Cyber Security, Natilik can support you to achieve your goals.

Offering best in class technology across eight key areas, Natilik can help unlock the unrealised potential of your business. What's more all of this is supported by a full range of services to support you through the lifecycle of your technology.

1.4 About Wayne Enterprises

Wayne Enterprises have been using HashiCorp Terraform OSS to manage their Cisco ACI environment for the last 3 years. They would like to start adopting HashiCorp Terraform in their wider team and eventually start using some of more advanced features like Terraform Cloud or Sentinel. Before Natilik will start working on HashiCorp Terraform client success and adoption plan Wayne Enterprise has agreed to run Terraform assessment document to audit the current HashiCorp Terraform.

2 Maturity Report

2.1 Evaluation

The following report represents an assessment of the Wayne Enterprises Terraform environment and includes a findings summary, prioritization matrix, optimization plan, actionable recommendations, and options for further enablement. The report is based on the information provided and, as such, may not represent an all-encompassing solution.

Key	
Red	Recommend immediate remediation
Yellow	Working, but needs an overhaul
Green	Optimal, can be enhanced over time

2.2 Analysis

Strengths	Weaknesses
Network team is tightly coupled Agile networking team Strong Cisco ACI knowledge and skills	GitLab Pipeline Codebase readability Cisco ACI Simulator reliability
Opportunities	**Threats**
Terraform Codebase Utilize cloud engineering team Terraform knowledge Restructuring Terraform codebase Improve networking team Terraform skills	Terraform OSS Support model Terraform Security and Policy as Code Currently used Terraform version Unresolved bugs and configuration issues Static secrets in Terraform code

2.3 Summary

Topic	Sub Section/Area	Overall Assessment	Prioritization
Terraform Usage	Run Environment		High
	Variables		High
	CI/CD Integration		High
	State Storage		High
	Versioning		Low
	State Access Control		High
	Backup and Recovery		Low
	Upgrade Process		Low

3 Recommendations

3.1 Terraform Usage

Component	Response	Recommendations	Chapter
Run Environment	GitLab	Remove Cisco ACI Simulator from pipeline, reconfigure YAML file to 'plan -out' for test stage.	5.2.1
Variables	Terraform	Improve readability, maintainability. Restructuring and modifying existing Terraform codebase.	5.2.2
CI/CD Integration	GitLab	Carry on with the utilization of GITLAB/Terraform pipeline but cut down the steps by removing one of the first test stages.	5.2.3
State Versioning	Not Known	Confirm version control is enabled on S3 bucket.	5.3.2
Upgrade Process	User does not know	Consult the official TF docs, release notes for most up to date info on breaking changes and their impact. Natilik can provide monthly review and guidance on upgrades, keeping up to date with patching and new features.	5.3.4
Has Terraform been mandated to be the standard way of provisioning infrastructure? What workloads are not provisioned with TF and why?	No, it's been adopted by the Network Team as the primary tool for provisioning infrastructure	Increment the adoption of Terraform into the Second Data Centre. Identify each specific use case and build up codebase	5.3.5

3.2 Provisioning Pattern Related

Component	Client Response	Recommendations	Location
Do you have future use cases identified?	Second Data Centre	Plan for additional modules and variables, mapping together Cisco ACI infrastructure	5.4.1
What does your current Terraform provisioning process consist of?	Terraform Via GitLab	Enhance your Terraform integration with GitLab	5.4.2
What percentage of your infrastructure is	50 Percent	Assess the scale requirement and optimize new terraform code for future build phases	5.4.3

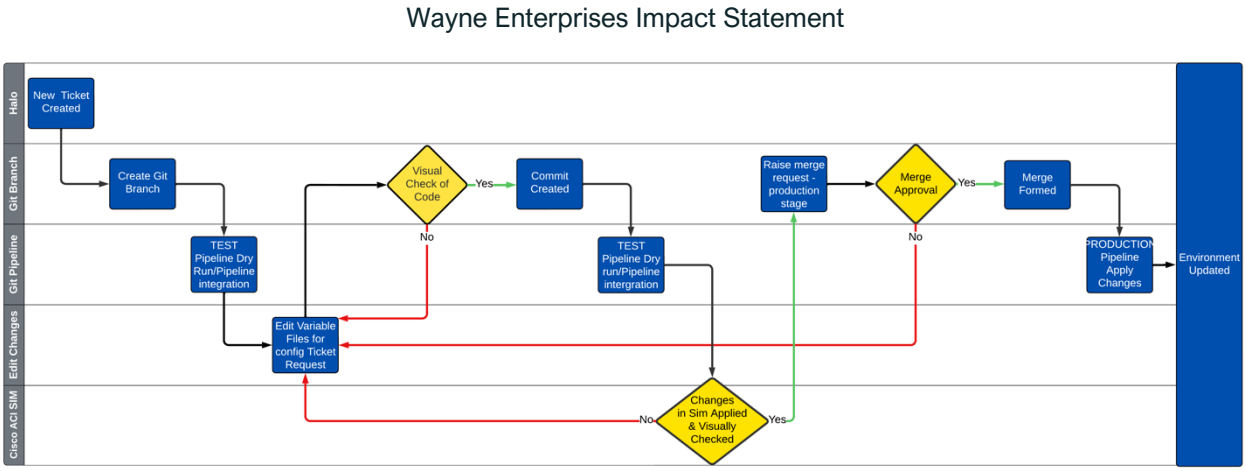
under Terraform management?			
What is the volume of requests for infrastructure provisioning?	Once a month	Prepare for future infrastructure upscaling.	5.4.4
Is there an approvals process for provisioning?	Approving each other	Implement granular GitLab approval levels based on significance and impact of the changes being made on the resource.	5.4.5
Are there any teams that impact your SLA/SLOs?	No	None	
Do you require compliance with any elevated regulatory compliance standards?	User does not know	Identify any gaps or areas to meet any possible future requirements. Resource tagging assists in categorizing related resources (“Production” “Development”)	5.4.6
Do you have infrastructure provisioned outside of Terraform e.g., in Azure ARM templates	No	Produce, resource document mapping dependencies and relationships.	5.4.7

3.3 Conclusion

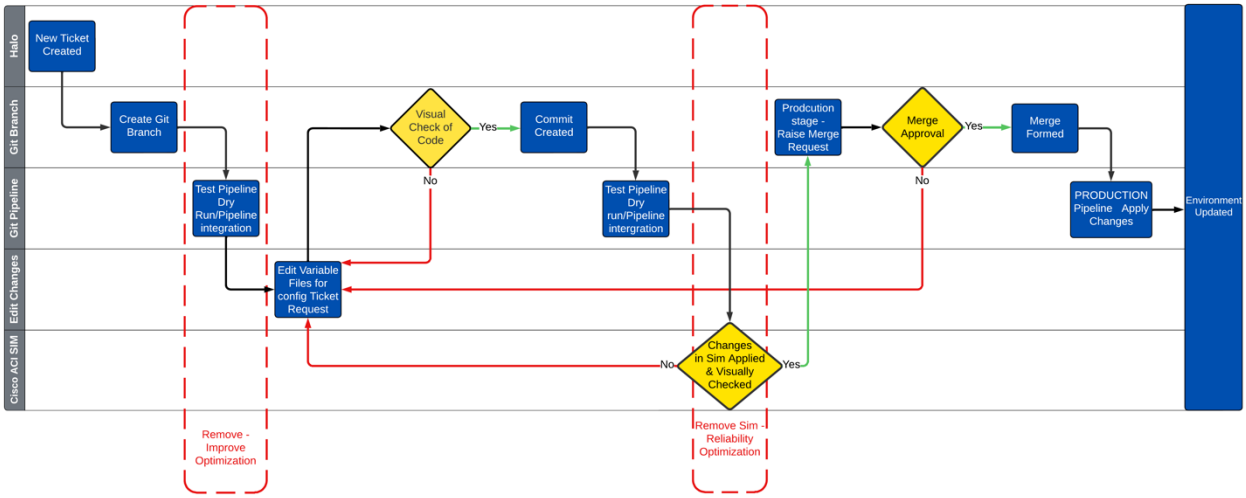
- Version control being utilised well.
- Team is tightly coupled for collaborative actions. Changes can be reviewed quickly to facilitate requests.
- Improve the CI/CD process for faster operational delivery.
- Enhance reliability by eliminating the Cisco ACI Simulator
- Terraform codebase requires improving by refactoring it, this can be achieved by restructuring the internal details of the existing variables without changing the external behaviour. The goal is to improve the readability, maintainability, and general hygiene of the code.
- Gain further understanding of the codebase by undertaking comprehensive Terraform training that focuses on modules, variables, and dynamic blocks (such as for_each, Depends_on). This training will also cover how modifications and updates can impact the Cisco ACI architecture.
- Natilik can offer continuous improvement, timely responses, and personalised assistance with Natilik Managed and Client Success services.

4 Impact Assessment

4.1 Current

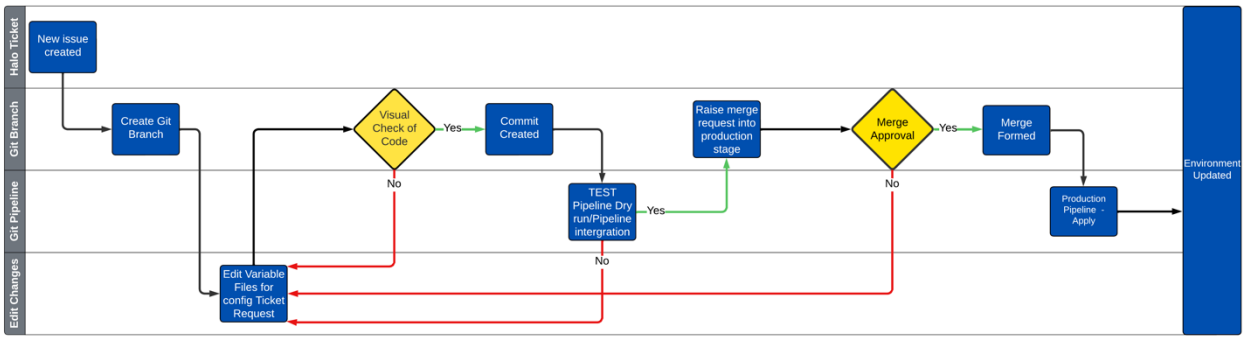


4.2 Proposal



4.3 Future

Wayne Enterprises Impact Statement



5 Adoption Plan

5.1 Introduction

Wayne Enterprises Network team provision Cisco ACI resources utilizing Terraform, managed GitLab CI/CD. The Network team engaged with Natilik in reviewing their architecture and be provided recommendations based on ongoing and future practices to ensure a robust, scalable, and resilient platform.

5.2 Optimisation

5.2.1 Natilik could remove the Cisco ACI Simulator, reconfiguring YAML file. To bypass Cisco ACI Simulator a test branch config will be created, that allows user to perform a plan on the pull request.

5.2.2 Natilik could refactor Terraform Variables and Modules, restructuring the internal details to improve the readability, maintainability, and general hygiene of the code. Initial update to the ACI Access Port, Block Module, App EPG, Bridge Domain, Attachable access entity profile. After these changes the next phase of objects can be refactored.

5.2.3 Natilik could reconfigure CI/CD pipeline. By removing the first test step, by reconfiguring the YAML file. Allowing user to go straight into editing the variable.

Review codebase, confirm understanding of changes to variables. Ensuring the correctness and reliability of the infrastructure provisioning process.

5.3 Efficiency

5.3.1 To confirm if the S3 bucket is encrypted, check its configuration and verify if the encryption is enabled.

5.3.2 Manage scheduled Monthly checks looking for new features, upgrades, and patching requirements. By keeping up to date with Terraform website and GitHub repository. By following these actions, you can ensure the infrastructure remains secure and optimized with the latest features and improvement mitigating potential risk and disruptions.

5.3.3 To incrementally adopt Terraform into the Cisco ACI environment, identify the use cases where Terraform can be beneficial, such as provisioning tenants, EPGs and contracts along with configuring network policies and security groups.

5.4 Innovation

5.4.1 Plan a design for Terraform infrastructure with module and variable development in mind, focus on creating additional reusable modules that are tailored to resource provisioning and management. This approach helps with consistency, simplifies maintenance, and enables easier scaling of your infrastructure.

5.4.2 Terraform/Gitlab environment team collaboration is good. Just ensure careful handling of Variables when updating.

5.4.3 To prepare for any high-volume infrastructure provisioning requests, review the existing code config and identify common patterns or resources that can be created into new modules, optimizing any scale requirements.

5.4.4 To prepare for potential future upscaling of infrastructure, consider creating a documented plan of when and how you will roll out the architecture, identifying resources, setting priorities and timeline.

5.4.5 Assign Gitlab approval rules prioritising critical resources with stricter requirements, while streamlining less impactful changes. Approvers are assigned based on expertise, ensuring thorough evaluations before merging. This workflow reinforces compliance, ensuring reliable and stable infrastructure. By implementing granular GitLab approval levels based on the significance and impact of changes, you can enforce control over Terraform Config

5.4.6 Define and implement resource tagging. Use tags to help organise and categorise resources. By using them, Terraform can improve the efficiency of management, environment, and automation of the infrastructure, enabling better visibility, control, and collaboration. For example, EPGs, Interface Profiles, Access Ports can be associated with tags, allowing for categorising and group resources based on specific attributes.

5.4.7 Create a resource document in Terraform-Docs to map the dependencies and relationships. The document should be organised hierarchically, showing the relationships between modules, resource blocks, and variables.

6 Operations Plan

Presently, Wayne Enterprises operates in separate environments for the cloud and network teams. However, we've identified an opportunity to merge these teams into one organisational unit, using either Terraform Cloud or, for more advanced use cases, Terraform Enterprise. By incorporating the network team as a workspace within Terraform Cloud and managed by the Wayne Enterprises cloud team, you can achieve improved security, cost-effectiveness, service delivery, and agility. We propose initiating a discussion with the Wayne Enterprises teams

here at Natilik, providing you with in-depth insights of HashiCorp Cloud and Enterprise and how you can merge into one managed Cloud Organisation. It can provide Wayne Enterprises with a single cloud operating model, providing a unified approach across the different teams on prem and cloud, enabling overall company efficiency.

7 Reliability Plan

7.1 Overview and Considerations

Writing infrastructure as code for multiple resources across a variety of services can be challenging. Terraform is a resource deployment tool that solves many of these challenges by giving expanded value to your as the complexity and its size grows. Wayne Enterprises environment must be architected for scalability, high availability, and automated change management (map your resources).

Prepare for contingencies, by having a strategy for recovering from failed or incomplete deployments. Consider backups of critical Terraform data and establishing mechanisms to revert changes if needed.

7.2 Reliability

To ensure better reliability for the future, code changes made will need to be understood in relation to the variables and modules/resource blocks they are linked with. A cleaner codebase will assist in ensuring future code breakages do not happen.

7.3 Terraform state file - Backup and Recovery

By running the "*terraform plan*" or the "*terraform apply*" commands, a file called *terraform.tfstate* is created which contains a list of infrastructure resources in JSON format. Before each operation, Terraform checks the file to know the current state of the infrastructure and will proceed accordingly with the creation, modification, or elimination of one or more resources. This file is very important because if it were to be corrupted or to be eliminated, Terraform, not having a vision of what the state of the infrastructure is, would not be able to deploy the resources or bring the infrastructure that is being created to an inconsistent state. Your *Terraform Backend is connected* to an S3 bucket and to ensure recovery enable *versioning,* so that every revision of the state file is stored and is possible to roll back to an older version if something goes wrong.

7.4 Terraform Support Levels

HashiCorp Terraform product support comes in many different favours depending on the type of the product. Natilik recommends staying for now with HashiCorp Terraform OSS and eventually Terraform Cloud Free however with increasing criticality of IaC Natilik would suggest starting to explore and moving to enterprise graded Terraform products like Terraform Cloud Standard and Plus or Terraform Enterprise.

7.4.1 Terraform Open Source

HashiCorp Terraform OSS offer product docs and guides, community feedback, and learning resources for the terraform open-source application. Terraform OSS has limited level of the support and it depends on response time of the community which is maintaining Terraform OSS codebase. Natilik can however offer to enhance support by developing an incident response plan specific to Terraform-related issues either as part of managed or professional services. Including escalations procedures and post-incident analysis for continuous improvement.

7.4.2 Terraform Cloud

A managed service that enables teams to collaborate, allowing easy access to shared state tied to workspaces, which helps keep state associated with the config that created it and offers policy controls for managing configurations. Terraform Cloud comes in 3 tiers: Free, Standard and Plus. Terraform Cloud Free tier in terms of support is dependent on the community therefore there are no guaranteed SLAs or any premium support or services. Terraform Cloud offering also includes Standard and Plus tiers which provides additional support with guaranteed SLAs. Standard and Plus guaranteed SLAs are essential when Terraform will become critical infrastructure to Wayne Enterprises.

7.5 Terraform Enterprise

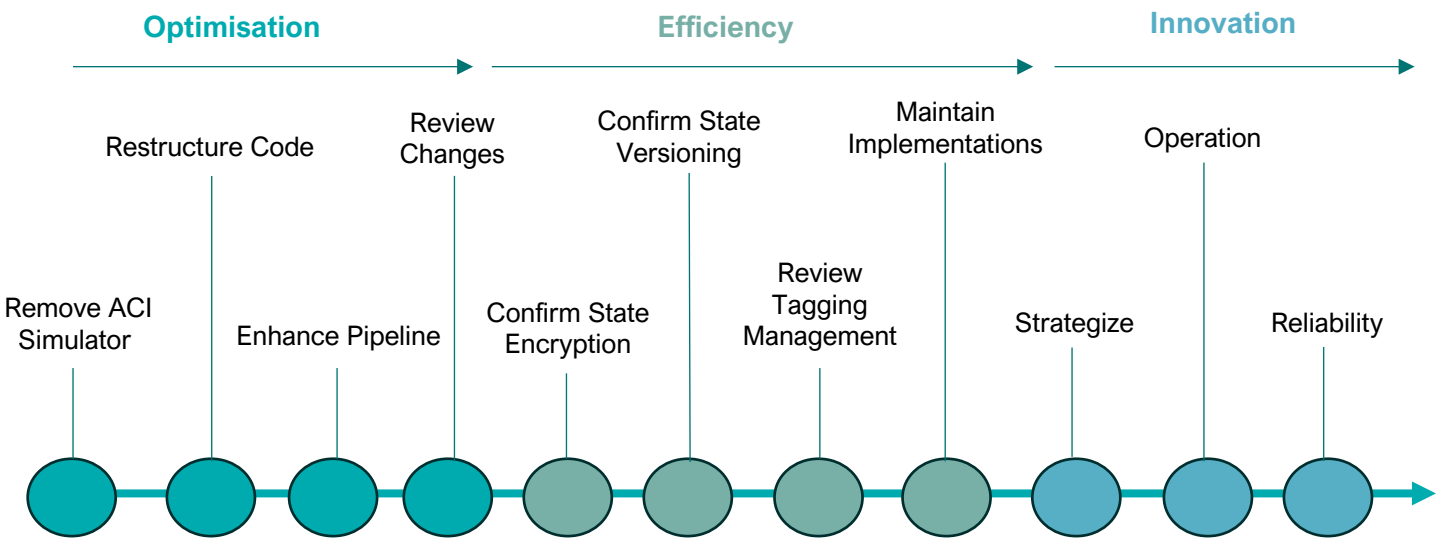
If the Terraform Cloud subscription proves insufficient as your needs grow, the next option is to deploy on Terraform Enterprise, the on-premises version of Terraform Cloud. Enterprise offers the same features as the cloud version but allows you to host the Enterprise version within your own datacentre. It offers no resource limits and additional enterprise-grade architectural features like audit logging and SAML single sign-on with guaranteed premium support and services and guaranteed SLAs.

7.5.1 Natilik IaC Managed Services

Natilik can provide wide range of managed services. Wayne Enterprises' desired managed service outcomes and key service requirements will be defined in the next phase during a workshop. Today, Natilik IaC Managed Services cover following areas and we look forward to further discussing these with the Wayne Enterprises team:

- **24/7/365 follow-the-sun technical Support** – Providing peace of mind that your estate is supported 24/7 by our experts .
- **Monitoring and Alert Management –** Ensuring 24/7 cloud-based monitoring of Vault and Terraform Cloud, across vital metrics in Vault such as Health, Leader, and Replication to ensure Wayne Enterprises' deployments are running as intended.
- **Service performance reporting** - Delivering key consumption/utilisation metrics with a focus on continual service improvement (CSI) including system and security recommendations.
- **Product Upgrades** – Major and minor upgrades to ensure Wayne Enterprises can leverage new features and functionality as it becomes available.
- **Product Updates and Training** – Product roadmaps, HashiCorp Academy Training, Natilik Training and Adoption Services
- **Client Success and Lifecycle Management Services** – Tailored success plans to help Wayne Enterprises accelerate time to value across your chosen solution and subscriptions.

7.6 Path to the production



NATILIK

Property of Natilik Ltd

NATILIK

London Office

9A Devonshire Square
London
EC2M 4YN
United Kingdom

+44 203 597 8000
info@natilik.com

New York Office

2 Park Avenue
20th Floor
New York, NY 10016
United States of America

+1 646 766 8600
info@natilik.com

Sydney Office

Level 57
25 Martin Place
Sydney 2000, NSW
Australia

+61 282 945 500
info@natilik.com

natilik.com