Provisioning Secure Cloud Environment Using Policy-as-code and Infrastructure-as-code

MSc Industrial Internship
MSc in Cybersecurity

Ayushi Tripathi
Student ID: x21120935

School of Computing
National College of Ireland

Supervisor: Prof. Vikas Sahni

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Ayushi Tripathi		
Student ID:	x21120935		
Programme:	MSc in Cybersecurity	**Year:**	2022/23
Module:	MSc Industrial Internship		
Supervisor:	Prof. Vikas Sahni		
Submission Due Date:	06/01/2023		
Project Title:	Provisioning Secure Cloud Environment Using Policy-as-code and Infrastructure-as-code		
Word Count:	6738	**Page Count**:	22

Signature:	Ayushi Tripathi
Date:	04/01/2023

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Provisioning Secure Cloud Environment using Policy-as-code and Infrastructure-as-code

Ayushi Tripathi
Student ID: x21120935

Abstract

Cloud capabilities are being embraced and managed throughout the organization, not only by IT personnel. This decentralized system necessitates the development of automated governance methods, since it can be difficult for teams to manually apply and validate compliance, security, or operating regulations. The Center for Internet Security (CIS) Controls reduce the likelihood of data breaches, data leaks, intellectual property theft, and other cyber threats. A solution to automate the deployment of a policy compliant infrastructure by codifying numerous policies across the business, help organizations use Infrastructure as Code and Policy as Code best practices. Terraform which is used as an Infrastructure as Code (IaC) tool is utilised to enable infrastructure provisioning automation which minimizes human error, reduces future risk and saves time and resources for the team.

This work presents an approach for an automated policy compliant secure infrastructure deployed on Amazon Web Service platform using Terraform. The infrastructure is compliant with CIS Amazon Web Services Foundations v1.4.0 and AWS Foundational Security Best Practices v1.0.0. The critical severity policies from CIS Ubuntu Linux 20.04 LTS Benchmarks have been implemented for Elastic Compute (EC2) web server.

Keywords- Center for Internet Security, Terraform, AWS, Security Best Practices

1 Introduction

Modern enterprises are turning to cloud services due to the capability of on-demand supply of computing, storage, and bandwidth resources. Many major technological, commercial, and media organizations, like Netflix or Salesforce.com, exclusively rely on the cloud, which is a cutting-edge technology (Patrick Mosca1, 2014). Despite the benefits, cloud computing presents significant security and privacy issues as an emerging technology, impeding its widespread adoption. Security has been identified as the most significant obstacle to user adoption of cloud computing. In the cloud environment, customers that outsource their data and apps may only rely on the Cloud Service Provider (CSP) to safeguard their security, the distinct properties of cloud computing provide several new security issues, the immaturity of security technology and the lack of cloud security governance are barriers to meeting user's security expectations. User's confidence in embracing this technology has been eroded by frequent security breakdowns in the cloud (Liu, 2015).

Cloud suppliers give security controls on areas such as data protection, identity management, application and system/network security and availability to their clients in order to establish security and privacy. Users, on the other hand, must meet a transparency of security controls in accordance with industry requirements, while vendors must enable them to do so. Standards such as NIST, ISO, PCI DSS, and others give a measure of information security from the standpoint of security (Chemerkin, 2013). Cloud computing governance, risk, and control are crucial in order manage security risks and safeguard systems and data. The execution of policies and processes enforces governance. These rules and procedures should be based on best practices and connected with business and IT goals. Controls should be created and implemented to guarantee that essential actions are made to address risks and accomplish business and IT objectives (Mariana Carroll, 2011).

It is difficult for teams to manually apply and validate compliance, security, or operation policies in every instance being deployed, which is why it's necessary to discover automated approaches to maintain governance in this decentralized environment. For policy enforcement, compliance and governance teams traditionally authored policies in a document and referred to this whenever approving or denying requests from business. Security teams are commonly consulted after a project has been coded, creating unfair tension and unexpected delays when the code has vulnerabilities. These manual processes can be error prone and difficult to scale. Policy as Code (PaC) allows enterprises to leverage Infrastructure as Code (IaC) best practices through codifying different policies across the enterprise. Policies can be managed and documented consistently at scale, with automatic policy deployments. PaC enables enterprises to leverage IaC best practices by codifying various policies across the organization.

A technology called Terraform uses the IaC to provision infrastructure. Cloud infrastructure can be created, modified, and version controlled using Terraform. The Terraform configuration files that users create define the state of the cloud infrastructure (Wang, 2019). The infrastructure provisioning procedure can be used to safeguard the environment from security threats. When requiring the usage of IaC for any modifications in the environment, code may be evaluated to guarantee that any security flaws are identified and repaired before infrastructure is supplied. To guarantee that the environment is consistent with the rules, security or operational guardrails can be formalized and enforced using CI/CD pipelines, gates, or other automated ways.

For this research work, AWS infrastructure has been spun up using Terraform. The IaC technique has been used to accelerate the infrastructure related tasks and the PaC technique has been used to deliver the infrastructure, which is secured, standardised and consistent with AWS Foundational Security Best Practices v1.0.0 and CIS Amazon Web Services Foundations v1.4.0. The web server has been integrated with security controls to make the operating system compliant with CIS Ubuntu Linux 20.04 LTS Benchmark[1].

[1] https://downloads.cisecurity.org/#/

1.1 Research Question:

How to use Policy-as-code to reduce the occurrence of security issues in cloud infrastructure provisioned by using an infrastructure-as-code automation tool like Terraform?

1.2 Research Objective:

Automating deployment of a policy compliant AWS Infrastructure.

1.3 Research Outline:

A detailed comparison of this study with previous review articles is given in Section 2. Section 3 describes the technique and methodology used to achieve the objectives of this research. Section 4 provides the design criteria and the architecture. Sections 5 and 6 present the implementation along with tools and evaluation of the research respectively. The research article concludes with Section 7 that covers conclusions, future work, and limitations.

2 Related Work

Companies and organizations benefit from cloud computing since it removes the need for them to plan ahead for provisioning and enables them to start with small resources and progressively raise them as service demand grows. Cloud computing adoption also presents several challenges, but these difficulties also open opportunities for study into a number of cloud computing-related topics. Security of the information stored, and resources used in a cloud environment is one of the major issues (Shivam Sharma, 2018). In this paper, several studies pertaining to security concerns in cloud have been presented. The main objective of this study is to shed light on the security issues surrounding cloud computing occurring due to missing standards and policies in place, as well as the various approaches and solutions that have been suggested.

2.1 Security Challenges and Solutions in Cloud Computing

(Patrick Mosca1, 2014) presented several critical security vulnerabilities that were compelling to cloud services in 2014. The writers investigated several aspects of cloud security, such as information security, cloud risks, cloud management, and account hijacking. In 2010, attackers were successful in building an XSS hijacking attack on AWS. Amazon Relational Database Service (RDS) was also targeted, such that even if the attackers lost their unique access, they still had a backdoor into the Amazon framework. The attackers obtained the login information of anybody who clicked the login button on the Amazon landing page. The attackers used their servers to infect fresh devices with the Zeus trojan virus and effectively control machines infected with it. This contextual research revealed an important conclusion: even a single weakness in a security architecture might endanger a whole system. The authors ended by suggesting two remedies to such AWS attacks. To begin with, AWS should not enable services and clients to exchange account login information. Second, a two-factor authentication approach should be presented. The author however missed to discuss aspects like the security of cloud-stored data, data integrity with several backups for services, etc.

Security threats, weaknesses, and attacks were covered in research by (Boisrond, 2021) which arise due to misconfigurations of S3 bucket. The author has further discussed how the (access control list) ACLs of misconfigured buckets which allow public-write, public-delete, or public change of the ACL, will eventually lead to security and/or privacy concerns. For example, a bucket that is purposefully made public will raise privacy concerns. That implies sensitive and secret data might be leaked. When buckets are publicly accessible, data leaking is an issue. Second, when a writable bucket is exposed to the Internet, malicious actors have an excellent chance to drop malicious items into the bucket. To comprehend the significance of exposing S3 buckets to the Internet, the author examined the following literature: Premier Diagnostics, a medical provider in Utah, exposed 50,000 patient information in 2021 due to incorrectly setup S3 Buckets that lacked password protection or authentication. In 2021, numerous US communities suffered a significant data breach as a result of multiple misconfigured Amazon S3 buckets by PeopleGIS, a software corporation in Massachusetts, which publicly exposed critical information, including individuals' personal data. The author has concluded by stating how these misconfigurations could lead to other security issues, such as cryptocurrency mining, ransomware encryption of the objects in those buckets, and phishing due to Domain Name Exploitation. As a result, protecting S3 buckets should be one of the top responsibilities for every firm that has data in AWS. While the author’s work provides a great deal of insight on the security of S3 buckets, it fails to address others features in AWS like Security groups and IAM roles that must be secured and monitored in order to get a secured infrastructure.

The authors (Nalini Subramanian, 2018) discussed the security challenges related to Virtual Machine layer. The security concerns were divided into categories such as VM Cloning, VM isolation, VM Escape, VM Rollback. The author has proposed the usage of Advanced Cloud Protection System (ACPS) which improves security and upholds integrity while suffering little performance hit. The author has also mentioned the requirement for deployment of a secure system the regardless of the VMs from various enterprises put on the same shared network as a future work.

The study conducted by (Abdullah Alqahtani, 2018) throw a light on the security vulnerabilities present in Amazon Web Service. The authors chose Amazon Elastic Compute Service (EC2) instances and used them to demonstrate methods for gaining access to and exploiting an instance. They also demonstrated how to circumvent the payment mechanism of paid images by changing the AMI file. The author proposed the solution that sharing of account login details between services and customers should not be permitted by AWS. Additionally, users of AWS must study the security guidelines for the service before using it.

(Johnson, 2020) have discussed how in order to build a solid Identity Access Management (IAM) system, businesses are using a variety of standards, control frameworks, and regulations. The aim of the study conducted was to improve the security and privacy feature that shield firms from data breaches, financial losses, and reputational damage. The author has cited cyber-attacks that are caused due to lack of identity and access controls and absence of well-designed IAM Framework that can effectively stop or quickly report an attack. The framework designed by the author is appropriate for crucial domains in the government and healthcare industries where identity assurance and verification of workers or users of the domain are put to the test. In accordance with the organization's needs for user authentication, the framework has to be tweaked with the necessary alterations for identity proofing.

2.2 Compliance in Cloud Computing

(Brandic, 2010) created a cloud computing (C3) architecture that addressed security, compliance, privacy, and trust concerns. According to the authors, C3 could be utilized to protect data privacy by mandating data storage in specified zones and implementing data fragmentation. They claimed that the framework could act as a middleman to connect numerous service providers. To examine rules such as HIPAA, PCI, and SOX, the authors presented a domain specific language (DSL), a metamodel, and an activity diagram. The authors however faced implementation issue and their model lacked certification process.

PCI compliance difficulties and solutions in Cloud were described by (Dereje Yimam, 2016) Costs, overlaps, legal issues, security, maintenance, complexity, code quality, and new technologies were all discussed by the writers. The solutions they provided were based on best practices. Authentication, authorisation, encryption, and monitoring are among the options. The authors did not explore how to deal with regulatory complications and overlaps.

(Wayne Jansen, 2011) listed a variety of privacy and security vulnerabilities that might affect cloud computing. The article discusses governance, compliance, trust, architecture, identity, access management, software isolation, data security, availability, and incident reporting challenges and suggestions. According to the study, one of the most difficult challenges to deal with cloud computing is compliance, understanding and implementing rules. They investigated the role of data location, loss of control, and transparency in public cloud compliance. The authors did not provide approaches for mapping complicated policies into best practices, patterns, or reference architectures. They also stated that the majority of cloud service providers utilize third-party certification to certify compliance. The study lacked to address proprietary solutions that lack vendor neutral models or architectures that all stakeholders may use as a checklist.

Authors (Cisic, 2008) contrasted GLBA, HIPAA, PCI, and SOX standards in terms of producing audit reports. According to their results, several reports and services, such as the user login report, user logoff report, user failure report, and logs access report, have common features. They came to the conclusion that SOX compliance with regard to reporting also includes the mandatory reports for GLBA, HIPAA, and PCI-DSS. Other aspects of compliance, such as privacy, security, user control, and notification, were not addressed by the writers. If the comparison had been supported by more exact artifacts, it would have been more precise.

(Dipankar Dasgupta, 2012) claimed that tools for security and compliance might aid firms in certifying compliance They looked at compliance software like WatchGuard and Trust Wave to assess and produce reports on compliance coverage. Each vendor had a different level of detail and scope for their reports. Based on "who controls what," the authors classified service models and created a compliance mapping matrix. However, the matrix lacked clarity in precisely defining the role of users in cloud. They addressed the HIPAA and

PCI standards; however additional standards might not be covered by their findings. To increase customer confidence and trust, the authors recommended conducting additional research.

One of the most critical aspects of delivering open cloud services is ensuring compliance with security and privacy requirements. Cloud providers often give their clients security and privacy control in order to preserve user privacy and security. However, in order to achieve transparency, these cloud providers must always enable their users to adhere to the established security requirements.

2.3 Cloud Resource Orchestration

Academics and practitioners have reported an increasing number of concerns and challenges as cloud computing gains traction in the IT industry. The switch from monolithic to microservices design, as well as the shift from a mostly virtual machines (VMs) to a cloud native architecture, have made it far more critical to automate infrastructure to respond quickly. The traditional deployment methods and techniques are not simple and appropriate where a huge infrastructure has to be built, managed and monitored from the vendor console. Provisioning, application deployment, load balancing and monitoring are common problems associated with deployment from vendor console (Thiyagarajan, 2021).

A number of orchestration tools, including Heat, CloudFormation, Puppet, Microsoft Azure Automation, Ansible, Terraform, and others, have been created during the past ten years, according to a systematic study by (Tomarchio, 2020) and survey conducted by (Denis Weerasiri, 2018) on orchestration methodologies. The majority of them are IaC tools with automation of daily chores as their main goal. In order to continue in the orchestration industry, IaC products constantly add new functionality.

The OASIS TOSCA (Binz, 2013) standard is noteworthy in the area of orchestration and is predicted to be more widely accepted by the cloud community in the context of next-generation IT solutions. The key benefit of TOSCA is that it offers higher-level, portable, and reusable descriptions of cloud applications in addition to the necessary management functionalities. This is essential in facilitating cloud application end-to-end orchestration activities while preserving a high level of compatibility between various clouds. However, TOSCA has only addressed the modelling of applications and their orchestration features, without putting out any implementation languages or useful tools to automate their building on top of cloud infrastructures. Instead, it leaves all implementation concerns to interested providers and users. Additionally, there is a glaring absence of software solutions that allow TOSCA-driven holistic orchestration automation. Offering completely original solutions is tiresome and useless, especially in light of the enormous range of potent DevOps techniques and technologies that are constantly evolving.

Infrastructure as a Service clouds, according to the authors (Deelman, 2011), offer the ability to provision virtual machines (VMs) on demand, but they do not provide information on how to manage those resources once they have been provisioned. As a result, in order to utilize such clouds successfully, users must have access to technologies that make it simple to deploy applications to the cloud. The authors of this study built a system to construct, configure, and manage cloud-based deployments. However, the system was identified with a drawback as it required the users to respond to failures manually which is un-realistic approach since many users often leave virtual clusters running unattended for long periods.

The author (Brikman, 2019) conducted a thorough analysis of automation solutions and provided rating information by popularity. The most popular ones are Terraform, CloudFormation, Puppet, Chef. Puppet and Chef are only used in outdated projects with lock-in technologies since they have lost their appeal. Their primary drawback is the client-server design, which makes future implementations more complicated and unstable. They established a basis for cloud computing automation, but Ansible and Terraform have taken their place. The preinstalled environment on the adjustable side does not require these IaC tools, giving engineers more options for adaptable ongoing infrastructure upkeep. The author did not cover all cloud service providers but demonstrated how to use Terraform tool in real-world setting.

2.4 Research Niche

Table 1 Research Niche

Author	Strength	Weakness
(Patrick Mosca1, 2014)	Investigated several aspects of cloud security, such as information security, cloud risks, cloud management, and account hijacking. Revealed an important conclusion: even a single weakness in a security architecture might endanger a whole system.	Missed to discuss aspects like the security of cloud-stored data, data integrity with several backups for services.
(Boisrond, 2021)	Great deal of insight on the security of S3 buckets.	Limited details on Security Groups and IAM roles.
(Nalini Subramanian, 2018)	Proposed the usage of Advanced Cloud Protection System (ACPS) for the security of Virtual Machines.	Implementation difficulties.
(Abdullah Alqahtani, 2018)	Proposed the solution that sharing of account login details between services and customers should not be permitted by AWS.	Roles and Responsibilities of users and Cloud Service Provider is not clearly defined.
(Johnson, 2020)	The framework designed is appropriate for crucial domains in the government and healthcare industries.	The framework needs tweaking and alterations to be used by ither industries for identity proofing
(Brandic, 2010)	Created a cloud computing (C3) architecture that addressed security, compliance, privacy, and trust concerns.	Implementation issue and the model lacked certification process.
(Dereje Yimam, 2016)	PCI compliance difficulties and solutions in Cloud were discussed.	Failed to explore regulatory complications and overlaps.

(Wayne Jansen, 2011)	The issues and recommendations in the article relate to governance, compliance, trust, identity, access management, software isolation, data security, availability, and incident reporting.	The authors didn't offer methods for converting complex policies into recommended procedures, design patterns, or reference architectures. The study did not address vendor-neutral models or designs for proprietary solutions that all stakeholders may utilize as a check list.
(Cisic, 2008)	Concluded that SOX compliance with regard to reporting also includes the mandatory reports for GLBA, HIPAA, and PCI-DSS.	The authors did not address further compliance issues such user control, privacy, security, or notification.
(Dipankar Dasgupta, 2012)	Assessed compliance tools like TrustWave and WatchGuard to produce compliance report and created a compliance mapping matrix. Addressed the HIPAA and PCI standards	The matrix lacked clarity in precisely defining the role of users in cloud. Additional standards were not covered by their findings. Recommended additional research.
(Binz, 2013)	Provides administration functionalities together with higher level, portable, and reusable descriptions of cloud applications.	Tiring to present entirely unique solutions, especially in light of the vast array of effective DevOps approaches and technologies that are continually developing.
(Deelman, 2011)	A method for creating, configuring, and managing cloud-based installations was created by the study's authors	The system was found to have a flaw in that it required users to manually respond to failures, which is an unrealistic approach given how frequently users left virtual clusters running unattended for extended periods of time.
(Brikman, 2019)	Performed a comprehensive examination of automation solutions and gave popularity rating statistics and listed out the advantages and disadvantages.	The author does not cover all cloud service providers.

3 Research Methodology

The research techniques utilized to accomplish the aforementioned research objectives are discussed in this section. The suggested solution is intended to address the practical difficulties associated with manual implementation approaches, including resource provisioning, visibility, control, governance and compliance. (Heena Kharche, 2020).

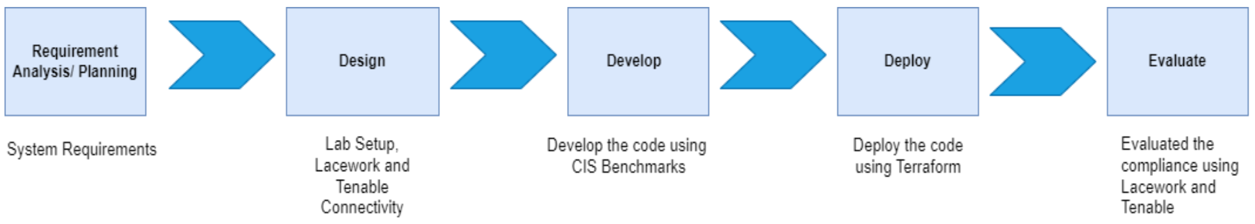

Requirement Analysis/ Planning
Design
Develop
Deploy
Evaluate
System Requirements
Lab Setup, Lacework and Tenable Connectivity
Develop the code using CIS Benchmarks
Deploy the code using Terraform
Evaluated the compliance using Lacework and Tenable

Figure 1 Proposed Research Methodology

- Requirement Analysis/Planning: The planning phases entail cooperation, discussion, review, and a strategy for security analysis. The CIS Amazon Web Services Foundations v1.4.0, AWS Foundational Security Best Practices v1.0.0, CIS Ubuntu Linux 20.04 LTS Benchmark and system requirements have been studied and reviewed in this research to analyse the feasibility of implementing the controls with respect to time and available resources. The framework and guidelines currently in use for cloud orchestration and provisioning have been examined. Information from cloud service providers and the service needed for deployment as a part of the research are analysed.
- Design: The AWS lab has been setup to carry out the deployment and infrastructure provisioning. The aws_secret_access_key and aws_access_key_ id is used to authenticate the creation of the Terraform infrastructure in AWS. The connectivity of the lab setup has been checked with evaluation tools such as Lacework and Tenable.io.
- Develop: The code required for building the CIS compliant AWS Infrastructure has been written using HashiCorp Configuration Language in Visual Studio. Using Terraform commands, the code has been reviewed for any template or syntax errors.
- Deploy: The code is then deployed using Terraform where any programmatic code is created to construct infrastructure and make API calls to CSP for resource generation.
- Evaluation: The compliance of infrastructure and Operating System is assessed using Lacework and Tenable respectively. The controls to remediate the security misconfigurations reported using these tools are integrated in the code again and the phases are repeated.
- Monitor: Log Metric Filters and Alarms have been setup in the infrastructure to report for any for unauthorized API calls, IAM policy changes, AWS Organizations changes, VPC changes and many such events.

Using this methodology and versioning with a descriptive model, Infrastructure as code (IaC) and Policy as Code (PaC) is used for defining and deploying infrastructure, such as networks, virtual machines, security groups, alarms, alerts, Identity and access management, load balancers, and connection topologies and codifying the industry standards and policies in the deployments. Terraform IaC code is built in Visual Studio and delivered to AWS Cloud Platform through Terraform CLI. The infrastructure is then scanned for CIS compliance using Lacework to provide a baseline for comparison. The modules of terraform are coded with CIS Amazon Web Services Foundations v1.4.0 and AWS Foundational Security Best Practices v1.0.0 and deployed to configure the AWS account with a baseline that is secure. The infrastructure is then once again scanned using Lacework to provide the security posture of the infrastructure. An additional module of terraform has been deployed which makes that Operating System of the Apache Webserver hosted on the EC2 instance compliant with CIS Ubuntu Linux 20.04 LTS Benchmark.

4 Design Specification

This area of research proposes the architectures that underlie the implementation of the research work. Figures 2 and 3 show the high-level design of the implemented architectures. Figure 2 shows the security features of AWS that have been utilised to create a policy compliant infrastructure which is compliant with AWS Foundational Security Best Practices v1.0.0 and CIS Amazon Web Services Foundations v1.4.0 The infrastructure built comprises of Logging and Monitoring, Storage, Identity Access Management and Networking features. In this case study an AWS VPC and one subnet has been set up. VPC's default security group has been configured to block all traffic. The security groups have been configured to provide no access to all the ports used for remote server management. The security features relate to each other, to trigger alarms, simple notifications, and log metric filters. An AWS lab setup has been used to carry out the implementation. Terraform connects with AWS Cloud Build using API. The lab connectivity is checked with Lacework to get the compliance report.

Figure 3 represents the architectural diagram where EC2 instance hosting an Apache webserver with Ubuntu 20.04 is spun up using Terraform. The Apache webserver has been coded to make it compliant using CIS Ubuntu Linux 20.04 LTS Benchmarks. In the AWS Lab, terraform was used to establish a VPC, Security Group, Subnet, and Internet Gateway, as well as host an Apache Web server. The instance has been assigned an elastic IP so that the IP address is preserved when the instance is stopped and restarted in the VPC. The server is scanned using Tenable.io. for compliance.

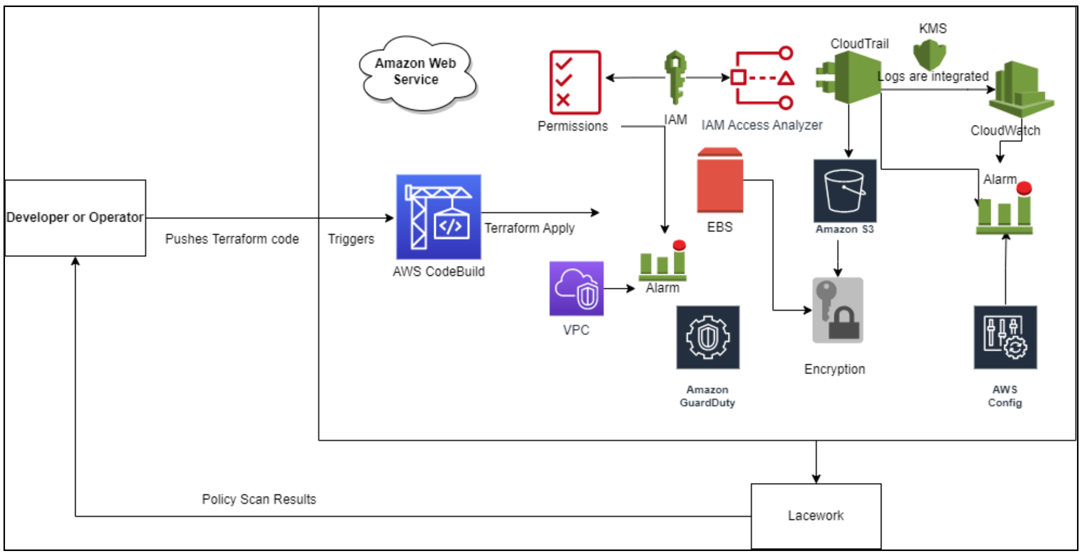


Figure 2 AWS Architectural Diagram

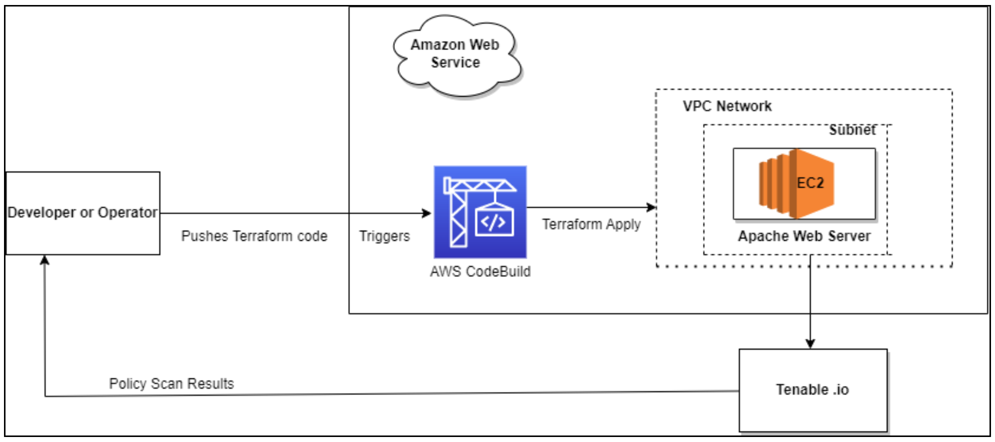


Figure 3 Architectural Diagram for Apache Webserver

5 Implementation

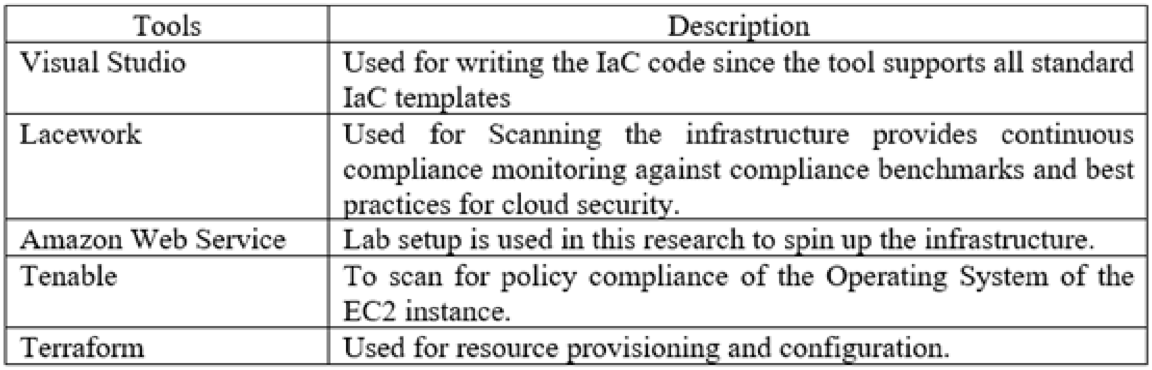
The implementation of an automated policy compliant infrastructure on a Cloud Service Provider (AWS) using Terraform has been described in this section. An AWS basic infrastructure has been spun up using Terraform and scanned using Lacework for CIS compliance in order to setup a baseline for comparison. A terraform module to configure the AWS account with a baseline that is relatively safe is then configured and deployed. The new infrastructure is again scanned against CIS Amazon Web Services Foundations v1.4.0 and AWS Foundational Security Best Practices v1.0.0 to provide the security posture of the infrastructure. Terraform-based Hashicorp Configuration Language (HCL) is used to describe Amazon Web Services (AWS) resources. Each AWS module must be understood independently to have a basic comprehension of the written code. "vpc_baselines.tf" file consists setup for VPC flow Logs and VPC baseline. The "bucket.tf" file consists of configuration of S3 to be compliant with eth CIS benchmarks. Similar to this, all required AWS resources have been organized with .tf extensions. To build all the defined resources, the terraform commands are run after scripting all the necessary files for the whole AWS architecture.

- The Terraform init command initializes the code and downloads the necessary prerequisite packages depending on CSP which is AWS in our research project.
- The Terraform plan command is comparable to the dry-run method. The code is checked for mistakes after it has been designed.
- The Terraform Apply command applies any programmatic code created to construct infrastructure and make API calls to CSP for resource generation.
- Terraform destroy function provides a convenient way to eliminate all produced in the event of any incorrect configuration, infrastructure. It greatly aids in managing the operating costs for cloud resources (Howard, 2022)

5.1 Tools

The tools used for the research work are listed in Table 2.

Table 2 Tools

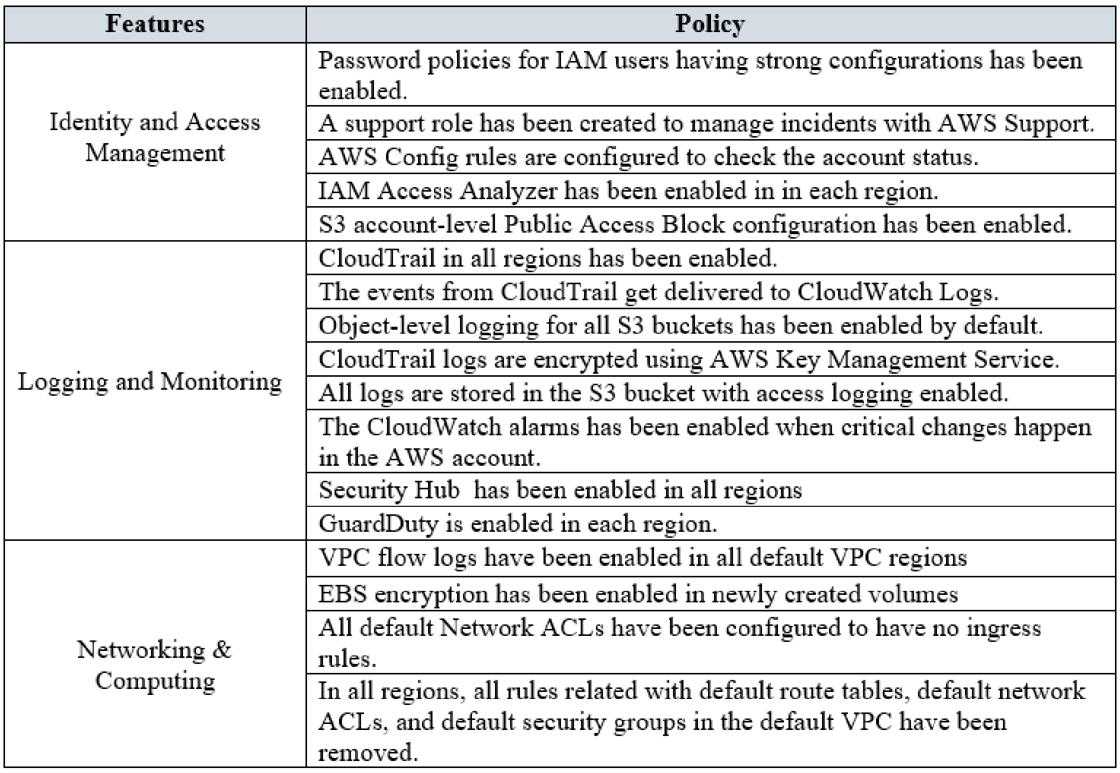


Tools	Description
Visual Studio	Used for writing the IaC code since the tool supports all standard IaC templates
Lacework	Used for Scanning the infrastructure provides continuous compliance monitoring against compliance benchmarks and best practices for cloud security.
Amazon Web Service	Lab setup is used in this research to spin up the infrastructure.
Tenable	To scan for policy compliance of the Operating System of the EC2 instance.
Terraform	Used for resource provisioning and configuration.

5.2 CIS Benchmarks

The features in the AWS Infrastructure and corresponding policies (CIS Amazon Web Services Foundations v1.4.0 and AWS Foundational Security Best Practices) which have been implemented in the work have been listed below in Table 3.

Table 3 Features and Policies



Features	Policy
Identity and Access Management	Password policies for IAM users having strong configurations has been enabled.
	A support role has been created to manage incidents with AWS Support.
	AWS Config rules are configured to check the account status.
	IAM Access Analyzer has been enabled in in each region.
	S3 account-level Public Access Block configuration has been enabled.
Logging and Monitoring	CloudTrail in all regions has been enabled.
	The events from CloudTrail get delivered to CloudWatch Logs.
	Object-level logging for all S3 buckets has been enabled by default.
	CloudTrail logs are encrypted using AWS Key Management Service.
	All logs are stored in the S3 bucket with access logging enabled.
	The CloudWatch alarms has been enabled when critical changes happen in the AWS account.
	Security Hub has been enabled in all regions
	GuardDuty is enabled in each region.
Networking & Computing	VPC flow logs have been enabled in all default VPC regions
	EBS encryption has been enabled in newly created volumes
	All default Network ACLs have been configured to have no ingress rules.
	In all regions, all rules related with default route tables, default network ACLs, and default security groups in the default VPC have been removed.

Users of Amazon Web Services (AWS) can manage users and user permissions in AWS using the web service known as AWS Identity and Access Management (IAM).
One can manage users, security credentials like access keys, and permissions that govern which AWS services users can access from a single location with IAM (Pandit, 2021).
To setup IAM Password Policy which should have minimum password length, should require numbers, symbols, uppercase and lowercase alphabets. The policy also states the number of

past passwords that the user is not permitted to use and the duration (days) of a user password's validity. A support role that has been created to manage incidents with AWS Support.

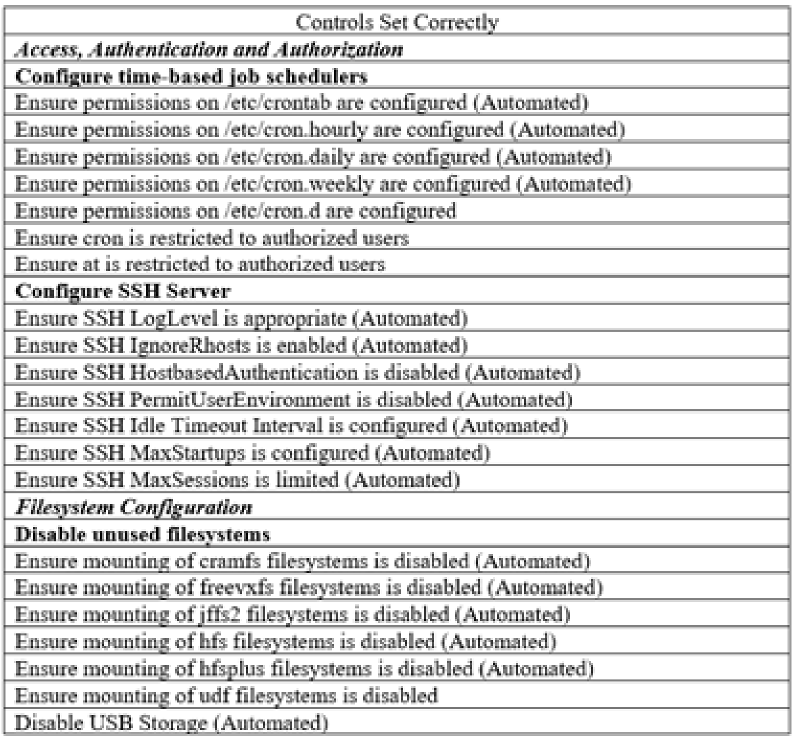
By helping the user proactively assess and modify their infrastructure, a well-architected monitoring and alerting system enhances dependability and performance of the AWS architecture (Nizam, 2021). For logging and monitoring purpose various resources are used to provide visibility into the resources and events in the AWS account. The log metric and alarms have been set for the following events:

- API calls which are unauthorized
- IAM changes
- CloudTrail configuration changes
- Console sign-in failures
- S3 bucket policy changes
- Security group changes
- Route table changes
- VPC changes
- NACL changes

The "aws_sns_topic" is the SNS topic the CloudWatch alarms transmit events to.

A second module in Terraform has been separately created to make the Operating System of the EC2 instance compliant with CIS Ubuntu Linux 20.04 LTS Benchmark. The AMI image chosen for this activity is Ubuntu Linux 20.04 LTS. A VPC, Security Group, Subnet and Internet Gateway has been created in the AWS Lab using Terraform and an Apache Web server has been hosted. The EC2 instance has been given elastic IP. The recommendations and technical controls which have been implemented are mentioned below.

Table 4 Controls Set Correctly



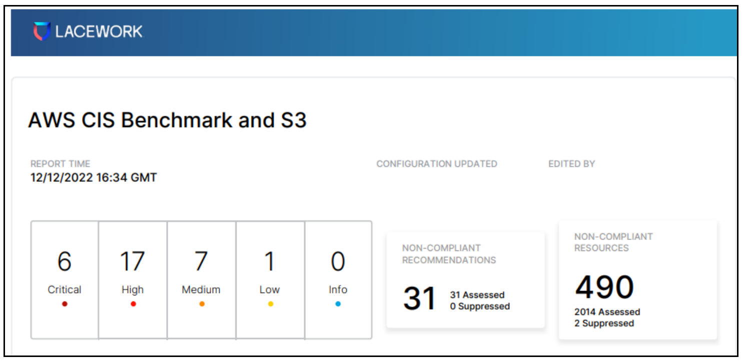
Controls Set Correctly
Access, Authentication and Authorization
Configure time-based job schedulers
Ensure permissions on /etc/crontab are configured (Automated)
Ensure permissions on /etc/cron.hourly are configured (Automated)
Ensure permissions on /etc/cron.daily are configured (Automated)
Ensure permissions on /etc/cron.weekly are configured (Automated)
Ensure permissions on /etc/cron.d are configured
Ensure cron is restricted to authorized users
Ensure at is restricted to authorized users
Configure SSH Server
Ensure SSH LogLevel is appropriate (Automated)
Ensure SSH IgnoreRhosts is enabled (Automated)
Ensure SSH HostbasedAuthentication is disabled (Automated)
Ensure SSH PermitUserEnvironment is disabled (Automated)
Ensure SSH Idle Timeout Interval is configured (Automated)
Ensure SSH MaxStartups is configured (Automated)
Ensure SSH MaxSessions is limited (Automated)
Filesystem Configuration
Disable unused filesystems
Ensure mounting of cramfs filesystems is disabled (Automated)
Ensure mounting of freevxfs filesystems is disabled (Automated)
Ensure mounting of jffs2 filesystems is disabled (Automated)
Ensure mounting of hfs filesystems is disabled (Automated)
Ensure mounting of hfsplus filesystems is disabled (Automated)
Ensure mounting of udf filesystems is disabled
Disable USB Storage (Automated)

6 Evaluation

This section of the work presents the important findings of the research as well as the experimental investigation and an illustration of a secured policy complaint IT infrastructure orchestrated using a single tool.

6.1 Experiment 1/ Testing with Terraform based non-compliant Infrastructure:

As indicated in the implementation phase, the infrastructure with missing policies was first deployed on AWS using Terraform. Lacework tool was used to generate the compliance report for this infrastructure. The tool also evaluates the resources that were pre-existing in the AWS lab environment with each scan.
As per the report, 2014 resources were assessed out of which 490 resources were reported to be non-compliant. There were 31 policies and recommendations that were identified as missing out of which 6 were Critical, 17 were High, 7 and 1 were Medium and Low severity policies respectively.


LACEWORK

AWS CIS Benchmark and S3

REPORT TIME
12/12/2022 16:34 GMT

CONFIGURATION UPDATED

EDITED BY

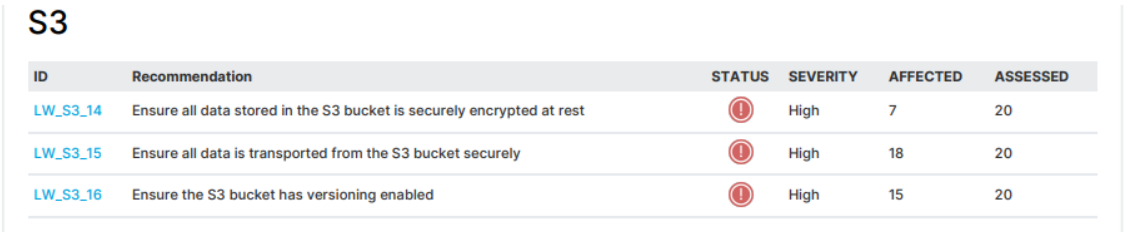
6	17	7	1	0
Critical ●	High ●	Medium ●	Low ●	Info ●

NON-COMPLIANT RECOMMENDATIONS
31 31 Assessed 0 Suppressed

NON-COMPLIANT RESOURCES
490 2014 Assessed 2 Suppressed

Figure 4 Summary

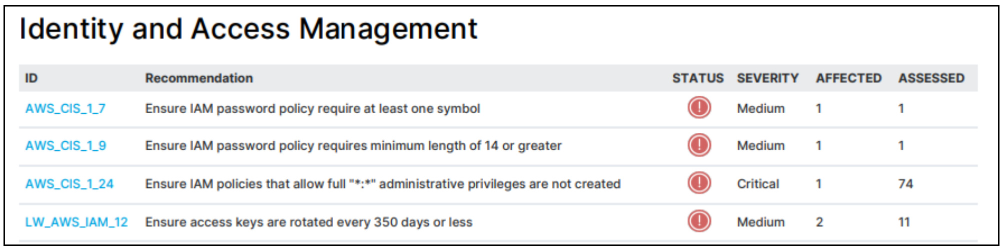
- The S3 bucket misconfiguration was reported due to missing encryption of data stored in the bucket, secure transportation of data and bucket versioning. The number of assessed and affected resources is shown below.


S3

ID	Recommendation	STATUS	SEVERITY	AFFECTED	ASSESSED
LW_S3_14	Ensure all data stored in the S3 bucket is securely encrypted at rest	!	High	7	20
LW_S3_15	Ensure all data is transported from the S3 bucket securely	!	High	18	20
LW_S3_16	Ensure the S3 bucket has versioning enabled	!	High	15	20

Figure 5 S3 Bucket

- The Identity and Access Management feature was misconfigured majorly due to missing password policies where the requirement of symbol in the password and password length was not set correctly and access key rotation.

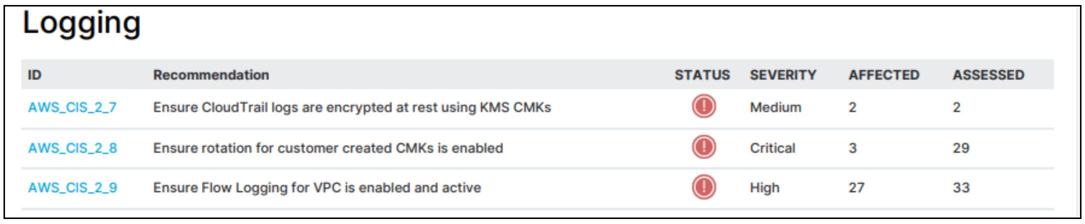


Identity and Access Management

ID	Recommendation	STATUS	SEVERITY	AFFECTED	ASSESSED
AWS_CIS_1_7	Ensure IAM password policy require at least one symbol	❗	Medium	1	1
AWS_CIS_1_9	Ensure IAM password policy requires minimum length of 14 or greater	❗	Medium	1	1
AWS_CIS_1_24	Ensure IAM policies that allow full "*:*" administrative privileges are not created	❗	Critical	1	74
LW_AWS_IAM_12	Ensure access keys are rotated every 350 days or less	❗	Medium	2	11

Figure 6 Identity and Access Management

- The logging feature was misconfigured due to missing encryption of CloudTrail logs and flow logging for VPC.

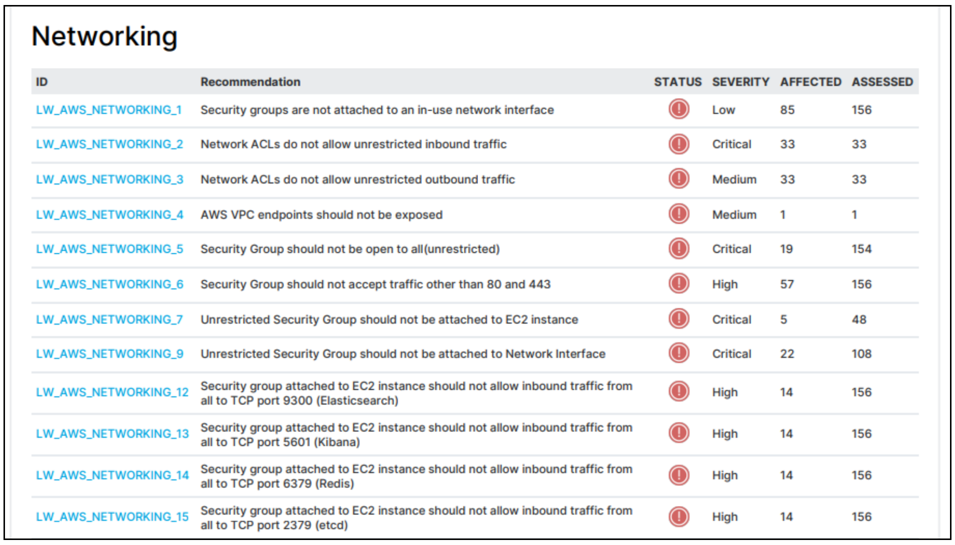


Logging

ID	Recommendation	STATUS	SEVERITY	AFFECTED	ASSESSED
AWS_CIS_2_7	Ensure CloudTrail logs are encrypted at rest using KMS CMKs	❗	Medium	2	2
AWS_CIS_2_8	Ensure rotation for customer created CMKs is enabled	❗	Critical	3	29
AWS_CIS_2_9	Ensure Flow Logging for VPC is enabled and active	❗	High	27	33

Figure 7 Logging and Monitoring

- The Networking feature was vulnerable due to various missing policies associated with Security Groups.



Networking

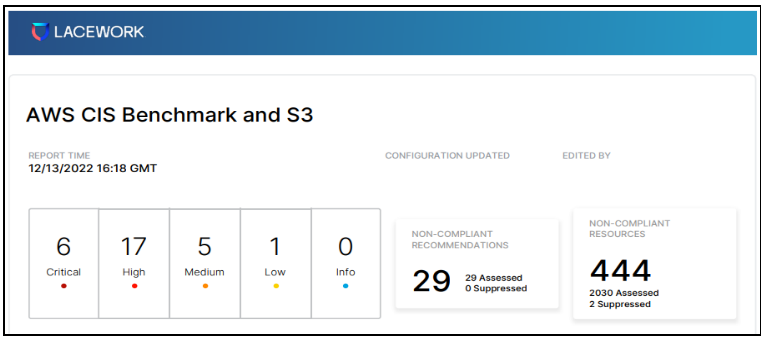
ID	Recommendation	STATUS	SEVERITY	AFFECTED	ASSESSED
LW_AWS_NETWORKING_1	Security groups are not attached to an in-use network interface	❗	Low	85	156
LW_AWS_NETWORKING_2	Network ACLs do not allow unrestricted inbound traffic	❗	Critical	33	33
LW_AWS_NETWORKING_3	Network ACLs do not allow unrestricted outbound traffic	❗	Medium	33	33
LW_AWS_NETWORKING_4	AWS VPC endpoints should not be exposed	❗	Medium	1	1
LW_AWS_NETWORKING_5	Security Group should not be open to all(unrestricted)	❗	Critical	19	154
LW_AWS_NETWORKING_6	Security Group should not accept traffic other than 80 and 443	❗	High	57	156
LW_AWS_NETWORKING_7	Unrestricted Security Group should not be attached to EC2 instance	❗	Critical	5	48
LW_AWS_NETWORKING_9	Unrestricted Security Group should not be attached to Network Interface	❗	Critical	22	108
LW_AWS_NETWORKING_12	Security group attached to EC2 instance should not allow inbound traffic from all to TCP port 9300 (Elasticsearch)	❗	High	14	156
LW_AWS_NETWORKING_13	Security group attached to EC2 instance should not allow inbound traffic from all to TCP port 5601 (Kibana)	❗	High	14	156
LW_AWS_NETWORKING_14	Security group attached to EC2 instance should not allow inbound traffic from all to TCP port 6379 (Redis)	❗	High	14	156
LW_AWS_NETWORKING_15	Security group attached to EC2 instance should not allow inbound traffic from all to TCP port 2379 (etcd)	❗	High	14	156

Figure 8 Networking

6.2 Experiment 2/ Testing with Terraform based compliant Infrastructure:

A new infrastructure was deployed in AWS that consisted of terraform modules to make the AWS account compliant with CIS Amazon Web Services Foundations v1.4.0 and AWS Foundational Security Best Practices v1.0.0. The infrastructure was yet again evaluated using Lacework and the findings were as follows:

2030 resources were assessed out of which, 444 resources reported to be non-compliant. There were 29 policies and recommendations that were identified as missing out of which 6 were Critical, 17 were High, 5 and 1 were Medium and Low severity policies respectively.


LACEWORK

AWS CIS Benchmark and S3

REPORT TIME
12/13/2022 16:18 GMT

CONFIGURATION UPDATED

EDITED BY

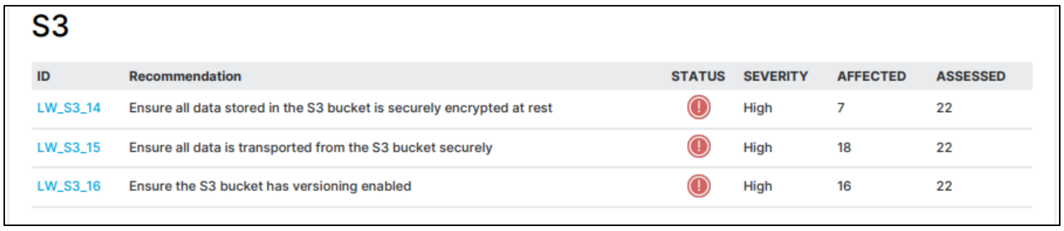
6	17	5	1	0
Critical	High	Medium	Low	Info

NON-COMPLIANT RECOMMENDATIONS
29 — 29 Assessed, 0 Suppressed

NON-COMPLIANT RESOURCES
444 — 2030 Assessed, 2 Suppressed

Figure 9 Summary

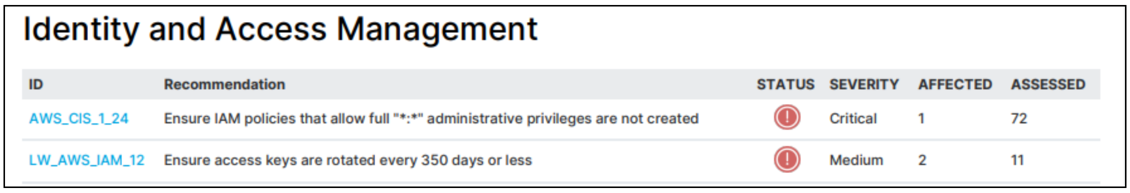
- In the below figure, it can be observed that for S3 bucket, the number of assessed resources increased to 22 while the number of non-compliant resources remained the same for encryption and secure data transport policies However, the versioning of S3 bucket remained non-compliant in the new infrastructure as well.


S3

ID	Recommendation	STATUS	SEVERITY	AFFECTED	ASSESSED
LW_S3_14	Ensure all data stored in the S3 bucket is securely encrypted at rest	❗	High	7	22
LW_S3_15	Ensure all data is transported from the S3 bucket securely	❗	High	18	22
LW_S3_16	Ensure the S3 bucket has versioning enabled	❗	High	16	22

Figure 10 S3 Bucket

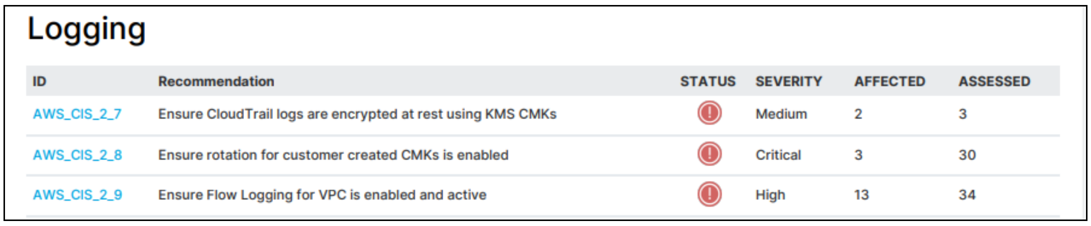
- For Identity and Access Managmenet feature, only two policies are flagged but the number of affected reources remain the same.


Identity and Access Management

ID	Recommendation	STATUS	SEVERITY	AFFECTED	ASSESSED
AWS_CIS_1_24	Ensure IAM policies that allow full "*:*" administrative privileges are not created	❗	Critical	1	72
LW_AWS_IAM_12	Ensure access keys are rotated every 350 days or less	❗	Medium	2	11

Figure 11 Identity and Access Management

- The number of affected resources for CloudTrail logs, rotation of CMKs and VPC logging are observed to be 2, 3 and 13 respectively.

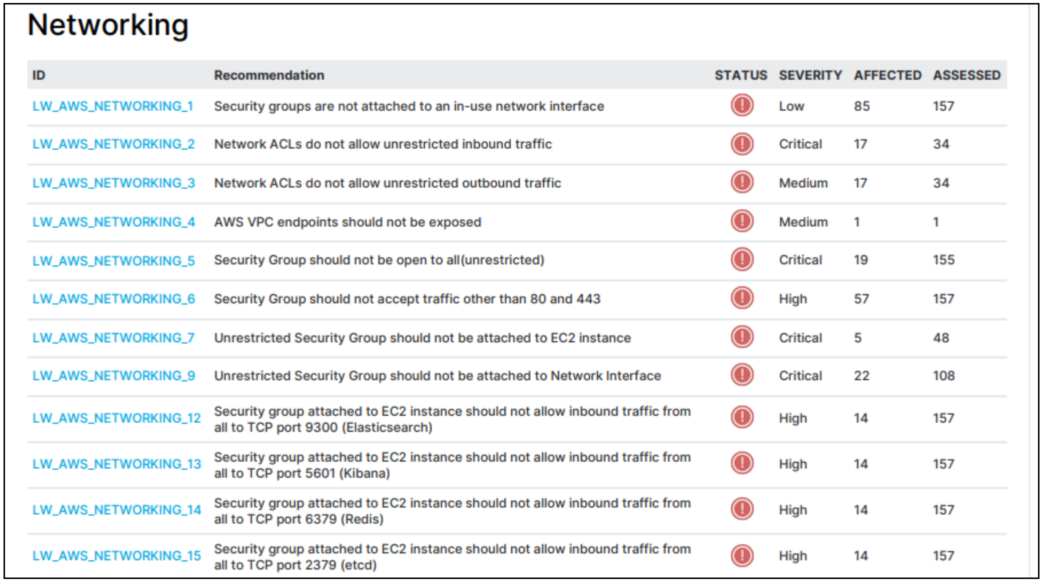


Logging

ID	Recommendation	STATUS	SEVERITY	AFFECTED	ASSESSED
AWS_CIS_2_7	Ensure CloudTrail logs are encrypted at rest using KMS CMKs	❗	Medium	2	3
AWS_CIS_2_8	Ensure rotation for customer created CMKs is enabled	❗	Critical	3	30
AWS_CIS_2_9	Ensure Flow Logging for VPC is enabled and active	❗	High	13	34

Figure 12 Logging and Monitoring

- For Networking feature, the number of affected resources remain the same with the increased number of assessed resources or decline as in case of Network ACLs.



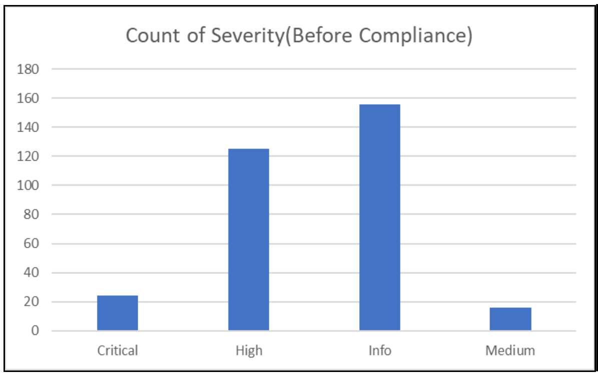
Networking

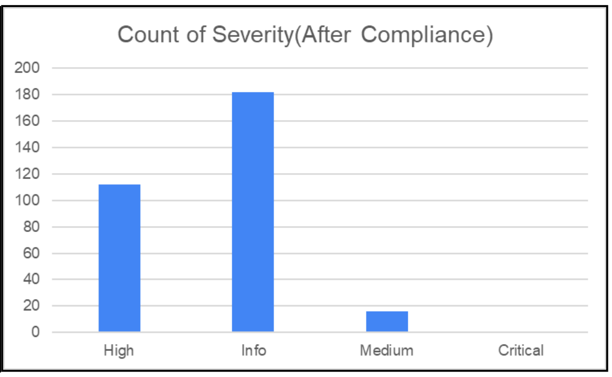
ID	Recommendation	STATUS	SEVERITY	AFFECTED	ASSESSED
LW_AWS_NETWORKING_1	Security groups are not attached to an in-use network interface	❗	Low	85	157
LW_AWS_NETWORKING_2	Network ACLs do not allow unrestricted inbound traffic	❗	Critical	17	34
LW_AWS_NETWORKING_3	Network ACLs do not allow unrestricted outbound traffic	❗	Medium	17	34
LW_AWS_NETWORKING_4	AWS VPC endpoints should not be exposed	❗	Medium	1	1
LW_AWS_NETWORKING_5	Security Group should not be open to all(unrestricted)	❗	Critical	19	155
LW_AWS_NETWORKING_6	Security Group should not accept traffic other than 80 and 443	❗	High	57	157
LW_AWS_NETWORKING_7	Unrestricted Security Group should not be attached to EC2 instance	❗	Critical	5	48
LW_AWS_NETWORKING_9	Unrestricted Security Group should not be attached to Network Interface	❗	Critical	22	108
LW_AWS_NETWORKING_12	Security group attached to EC2 instance should not allow inbound traffic from all to TCP port 9300 (Elasticsearch)	❗	High	14	157
LW_AWS_NETWORKING_13	Security group attached to EC2 instance should not allow inbound traffic from all to TCP port 5601 (Kibana)	❗	High	14	157
LW_AWS_NETWORKING_14	Security group attached to EC2 instance should not allow inbound traffic from all to TCP port 6379 (Redis)	❗	High	14	157
LW_AWS_NETWORKING_15	Security group attached to EC2 instance should not allow inbound traffic from all to TCP port 2379 (etcd)	❗	High	14	157

Figure 13 Networking

6.3 Experiment 3/ Testing with Operating System

An Apache webserver was spun up using Terraform on the AWS platform. The server was scanned using Tenable.io to identify the missing policies and configuration as per the CIS Ubuntu Linux 20.04 LTS Benchmarks. The bar graph below shows the count of policies per severity that were detected. 25 Critical severities, 125 High, 16 Medium and 156 Informational policies were missing.





The terraform module was then modified with certain configurations and controls mentioned in the implementation section in order to remediate the critical severity policies and deployed.

The webserver was again scanned with Tenable.io to generate the new report and the findings illustrated above show that 25 critical severity policies were corrected. The count of High, Medium, and Informational severity policies remained the same.

6.4 Discussion

A comparison of the two infrastructures demonstrated in Experiment 1 and 2 can be drawn. The infrastructure initially spun up shows 2014 resources were assessed out of which 490 resources were reported to be non-compliant. The infrastructure spun up in Experiment 2 illustrates that 2030 resources were assessed out of which, 444 resources reported to be non-compliant. The number of assessed resources in the report generated in Experiment 2 had increased and due to implementation of certain policies and configurations in all regions, the number of non-compliant resources were observed to be lessened which clearly indicates that the new infrastructure is relatively more policy compliant and secure. The S3 bucket has been configured correctly for data encryption at rest and secure data transport, however the versioning of the bucket is still missing and hence there is increase in the affected number of resources. For Identity and Access Management feature, the missing policies have been configured correctly. The access key rotation has been set correctly which can be inferred from the number of assessed assets being increased but the number of affected assets remain the same as in Experiment 1. The lack of encryption for CloudTrail logs and flow logging for VPC was improperly enabled earlier which was remediated in the new infrastructure. It is worth noting that enablement of VPC flow logging in all regions has brought down the number of affected resources from 27 to 13 despite the increase in the number of assessed resources. As demonstrated above, the number of affected resources due to missing Security Groups policy have also sharply declined in Experiment 2.

The number of critical severity policies in the Apache webserver were initially 25, demonstrated in Experiment 3 which were remediated post modifying the Terraform module with configuration changes. The critical severity missing policies can prove to be a great threat in increasing data breaches and loss and thus have been remediated on priority. This work may be developed further to address any severity policies and any configuration that is lacking from the operating system to bring it into compliance with CIS standards.

7 Conclusion and Future Work

The challenges to manually apply and verify compliance, security, or operational regulations in each instance that is deployed can lead to security issues like greater chance of credentials from a hacked or abandoned account being used, increased vulnerability of accounts to brute force login attempts and much more. This research work presents automated methods for upholding governance. The CLI-based IAC and PAC technique solves the problem of handling centralized deployment and administration. The implemented plan shows a centralized architecture that has Logging & Monitoring, Identity and Access Management and Networking and Computing features secured which makes the infrastructure compliant with CIS Amazon Web Services Foundations v1.4.0, AWS Foundational Security Best Practices. The Operating System of the web server is made compliant with CIS Ubuntu Linux

20.04 LTS Benchmarks. Twenty-five critical severity policies have been remediated in this work.

While terraform proved to be a great automation tool, it does come with the limitation of not allowing rolling back. All the resources constructed via Terraform need to be deleted and re-run-in order to fix any blip. The Lacework tool utilised for scanning the infrastructure only presents the number of non-compliant resources and not the names, which was challenging while evaluating the work.

As a part of future work, hardware Multi-Factor-Authentication can be enabled for root account and all IAM users which will provide an extra layer of security. Versioning of S3 bucket can be enabled by writing additional code. The possibilities of making a policy compliant infrastructure using Terraform can be investigated on other public Cloud Service Providers like Google Cloud Platform and Azure and in a multi-cloud environment. This work can be further explored to resolve all severity policies and missing configuration in the Operating System to make it compliant with CIS benchmarks.

8 References

Abdullah Alqahtani, H. G., 2018. Cloud Computing and Security Issues—A Review of Amazon Web Services. *International Journal of Applied Engineering Research ISSN 0973-4562l,* 13(22), pp. 16077-16084.

Binz, T. B. U. K. O. L. F., 2013. TOSCA: Portable Automated Deployment and Management of Cloud Applications. In: A. S. Q. D. F. Bouguettaya, ed. *Advanced Web Series.* New York: Springer, p. 527–549.

Boisrond, P. D., 2021. *A Position Paper on Amazon Web Services (AWS) Simple Storage Service (S3) Buckets.* s.l.:A.T. Still University of Health Sciences.

Brandic, S. D. T. A. D. S. F. L. a. R. K., 2010. *Compliant Cloud Computing (C3): Architecture and Language Support for User-Driven Compliance Management in Clouds.* Miami, 2010 IEEE 3rd International Conference on Cloud Computing, pp. 244-251.

Brikman, Y., 2019. *Terraform Up and Running.* Second ed. United States of America: O'Reilly Media, Inc.

Chemerkin, Y., 2013. *Limitations of Security Standards against Public Clouds.* Toronto, EEE Toronto Section i-Society 2013 Proceedings Contents.

Cisic, D. &. H. Z. &. B. M. &. M. M. &. V. D., 2008. *MIPRO 2008 Proceedings.* 5 ed. s.l.:Croatian Society for Information and Communication.

Deelman, G. J. a. E., 2011. *Automating Application Deployment in Infrastcruture Cloud.* California, Third IEEE International Conference on Coud Computing Technology and Science, pp. 658-665.

Denis Weerasiri, M. C. B. B. B. Q. Z. S. a. R. R., 2018. A Taxonomy and Survey of Cloud Resource Orchestration Techniques. *ACM Computing Surveys,* 50(2), pp. 1-41.

Dereje Yimam, E. B. F., 2016. A survey of compliance issues in cloud. *Journal of Internet Services and Applications,* 7(5), pp. 1-12.

Dipankar Dasgupta, D. N., 2012. *Security and Compliance Testing Strategies for Cloud Computing Dr,* memphis: s.n.

Heena Kharche, T. S. T. G., 2020. Infrastructure as a code - on Demand Infrastructure. *Special Issue of First International Conference on Advancements in Research and Development,* 02(08), pp. 194-197.

Howard, M., 2022. *Terraform — Automating Infrastructure As A,* Portland: s.n.
Johnson, F. M. P. D., 2020. *ROBUST IDENTITY AND ACCESS MANAGEMENT FOR CLOUD SYSTEMS,* Alberta: Concordia University of Edmonton.

Liu, Y. &. S. Y. &. R. J. &. R. S. &. V. A., 2015. A Survey of Security and Privacy Challenges in Cloud Computing: Solutions and Future Directions. *Journal of Computing Science and Engineering,* 9(3), pp. 119-133.

Mariana Carroll, A. V. d. M. A. V. d. M., 2011. *Secure cloud computing: Benefits, risks and controls.* s.l., Information Security South Africa (ISSA), 2011, pp. 1 - 9.

Nalini Subramanian, A. J., 2018. Recent security challenges in cloud computing. *Computers & Electrical Engineering,* Volume 71, pp. 28-42.

Nizam, K., 2021. *Designing and implementing logging and monitoring with Amazon Cloud Watch.* [Online]
Available at: https://docs.aws.amazon.com/pdfs/prescriptive-guidance/latest/implementing-logging-monitoring-cloudwatch/implementing-logging-monitoring-cloudwatch.pdf
[Accessed 7 December 2022].
Pandit, P., 2021. *Case Study on AWS Identity and User Management,* Mumbai: Research Gate.

Patrick Mosca1, Y. Z. Z. X. Y. W., 2014. Cloud Security: Services, Risks, and a Case. *International Journal of Communications, Network and System Sciences,* 7(12), pp. 529-535.

Shivam Sharma, D. N., 2018. CLOUD COMPUTING SECURITY CHALLENGES AND SOLUTIONS. *International Research Journal of Computer Science (IRJCS),* 5(02), pp. 65-69.

Thiyagarajan, S., 2021. *Automate Provisioning and Orchestration of Cloud Infrastcture using AWZ,* Dublin: National College of Ireland.

Tomarchio, O. &. C. D. &. D. M. G., 2020. Cloud resource orchestration in the multi-cloud landscape: a systematic review of existing frameworks. *Journal of Cloud Computing,* 9(1), p. 49.

Wang, T., 2019. *A Service for Provisioning Compute,* Uppsala: UPPSALA University.
Wayne Jansen, T. G., 2011. *Guidelines on Security and and Privacy in Public Cloud Computing,* Gaithersburg: NIST Special Publication 800-144.