National College of Ireland

Integrating Zero Trust Security with Network Penetration Testing in Multi-Cloud Environments

MSc Research Project
Master of Science in Cyber Security

Ajay Mohan
Student ID: 23389796

School of Computing
National College of Ireland

Supervisor: Khadija Hafeez

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Ajay Mohan	
Student ID:	23389796	
Programme:	Master of Science in Cyber Security	**Year:** 2026
Module:	MSc (Research) Practicum Part - 2	
Supervisor:	Khadija Hafeez	
Submission Due Date:	28/01/2026	
Project Title:	Integrating Zero Trust Security with Network Penetration testing in Multi-cloud Environments	
Word Count:	7289	
Page Count:	20	

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Ajay Mohan

Date: 27/01/2026

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):	

Integrating Zero Trust Security with Network Penetration testing in Multicloud Environments

Ajay Mohan
23389796

Abstract

The rise of multicloud infrastructures has created many new security challenges including inconsistency in how security policies are enforced across the multiple clouds, as well as a reduction in the amount of visibility there are into these types of setups, which are not supported by traditional perimeter-based security approaches. The Zero Trust Security Model has been developed as an alternative approach; this security model operates under "never trust, always verify". However, because the application of Zero Trust principles is theoretical, it needs to be proven through continuous validation in order to be fully functional in practice. This research project seeks to address the need for practical validation of the Zero Trust model by providing an automated framework to implement and test Zero Trust in a multicloud environment that includes Amazon Web Services (AWS) and Microsoft Azure. This framework provides automation across the entire security lifecycle, from the provisioning of the environment to discovering assets, applying baseline policies, running automated penetration testing and providing automated enforcement of baseline policies and analyzing the results. The main experiment performed in the research was to create a controlled multicloud environment, apply the established baseline Zero Trust Policies, and perform automated penetration tests to discover any gaps in security. The findings indicate that this framework can achieve asset discovery, application of security controls, and evaluation of actual security posture at network endpoint devices. A key finding of this work is the ability of the framework to expose a misapplication of policy. The penetration testing revealed that the real-world security posture of a virtual appliance was different than its intended state of being vulnerable. Even though the logic around remediation was designed conservatively, the framework returned a final score of 98 out of 100, demonstrating the efficacy of the framework's integrated approach. Therefore, this research provides a practical and automated means for continuously testing and enforcing the principles of Zero Trust within heterogenous and complex cloud environments.

1 Introduction

1.1 Background and Context

The shift of enterprise workloads from traditional on-premises to cloud-based platforms has been a major trend over the past few years. Multicloud solutions allow companies to avoid being "locked-in" to a single vendor by leveraging tools from multiple vendors. While this strategy can provide the flexibility and redundancy that comes with utilizing multiple

vendors, it causes several unique security challenges. The management of disparate security controls, the enforcement of consistent policy and the establishment of a unified view over multiple cloud vendor services creates an extremely complex and fragmented security perimeter that cannot be protected effectively with traditional security solutions. Traditional security models assume that there is a trusted internal network and an untrusted external network. This assumption cannot be used in multicloud environments where the borders between vendors' networks do not exist.

1.2 Emergence of Zero Trust Architecture

As a result of this set of challenges, the emergence of the Zero Trust Architecture (ZTA) as a new security model has occurred. The foundational principle of Zero Trust is that entities inside a network should not be treated differently from those outside the boundary; there is no implicit trust given to any entity (Mudau et al 2025). As such, an organisation grants access to its resources on a case-by-case basis and only to the individual or device needing access at present. To facilitate this architecture, organisations must utilise strong identity management and access control solutions, as well as use micro-segmentation (George 2025) and additional monitoring to establish a more comprehensive and fine-grained security posture, supported by a robust security operation centre, to better protect distributed or remote workforces (Sarkar et al 2022).

Although the Zero Trust principles provide an excellent theoretical basis for building an organizational security framework and guiding the development of robust policies and practices, validating these policies and practices requires ongoing and thorough validation. The implementation of these policies will not guarantee that they are being applied and that they are protecting against the constantly evolving threat landscape. The gap between an organization’s intended security posture and what is operationally being utilized within its network is often quite large. As an added measure to close the gap between the intended security posture and that which is operational, the practice of penetration testing, or ethical hacking, is the best way to provide an organization with a real-world view into its security posture (Joshi et al., 2025).

1.3 Need for Continuous Validation

Conducting manual penetration tests in multi-cloud environments that are Dynamic and Scalable are ineffective as well as non-scalable approaches. In addition to that, the DevOps methodology and Continuous Deployment pipelines evolve at an extremely fast pace; hence, Security Validation must also evolve at an equally fast pace through Automation. This study will investigate the importance of both disciplines intersecting and the primary question this study will attempt to answer is: **How can the principles of Zero Trust Security be integrated with Automated Network Penetration Testing to provide Continuous Validation, Enforcement, and Improvement of the Security Posture of a Multi-Cloud Environment?**

In addressing this problem, this project outlines the design, build and evaluation of a pioneering automated framework which provides a complete security process for the AWS and Azure environments. The framework starts with the set up and asset identification, applies Zero Trust security policies as a foundation, runs automated penetration tests to discover deviations and vulnerabilities, and finally, automation of mitigation based on the discovered risks. This research presents not only a real-world end-to-end solution for applying Zero Trust controls but also demonstrates the effectiveness of these controls by leveraging offensive security techniques in a continuously automated feedback loop. The research provides a practical example of how organizations can achieve a continually monitored and adjustable security posture within the ever-changing multi-cloud ecosystem.

1.4 Report Structure

This report is organized as follows: The second section provides a review of previous research in the areas of Multicloud security, Zero Trust and Penetration testing. The third section describes the research methodology used to conduct the experiment. The fourth section outlines the design and architecture of the newly proposed Automated Framework. The fifth section describes how the automated framework was implemented and what experimental set up were used during the experiment. In the sixth section, a thorough evaluation of the automated framework's results performed during multiple experiments, as well as an associated discussion of how these results will impact and influence future research. The seventh section concludes with a summary of the key points from this report, as well as an overview of the limitations of this research and recommendations for further research.

2 Related Work

Through a critical analysis of existing academic and technical literature regarding multicloud security, Zero Trust Architecture and penetration testing, this section identifies the current state of the art, identifies gaps in the current knowledge, and establishes both the uniqueness and need for a fully integrated framework.

2.1 Security Challenges of Multicloud Environments

The multi-cloud and hybrid cloud approach has become an increasing trend in enterprise IT and is widely referenced in literature. Gupta (2025) provides an in-depth review of the strategies, complications, and best practices that are associated with using multiple clouds. The operational and security difficulties that plague the operating of a mixed cloud environment are emphasized. Because of the nature of using many disparate clouds, organisations now have a greatly increased chance of suffering from a data breach or a misconfigured security policy. Harper (2025) reviews potential database security threats that exist in cloud environments and points out that access controls are often misconfigured and unsecured APIs expose databases to malicious actors. Similarly, Kunduru (2023) publishes

an extensive review of the dangers of enterprise cloud computing, maintaining that the increase in attack surface area in conjunction with.

2.2 Zero Trust Architecture as A New Modern Security Model

The Zero Trust Model has recently emerged as one of the most viable approaches for addressing these issues. In their paper on "Zero Trust Network Analysis of Cloud Computing", Sarkar et al. (2022) conducted a comparative analysis of various models of Zero Trust Networks used, with respect to how each is suited for use within cloud-based applications. Although the concept of Zero Trust Network is well understood, there are still many questions that remain unanswered about how best to utilize the principles of Zero Trust Networks when designing practical solutions to protect access to cloud-based applications. Adanigbo et al. (2024) developed a Reference Architecture view of a Zero Trust Architecture on MCMs and provide an in-depth treatment of implementing Identity Governance and API Security within a Microservices Environment. Their paper provides an excellent foundation for this topic, but it does not provide an empirical evaluation of the effectiveness of their proposed architecture.

In the second article about the Zero Trust Architecture (ZTA), Mudau et al. (2025) examine the different ways to implement a ZTA and the various frameworks that exist. The authors offer evidence that, for a ZTA to be successfully adopted, it needs to take place over time using a maturity model. Micro-segmentation of networks is one of the primary components of a ZTA, which entails slicing a single environment into multiple smaller environments, thereby limiting lateral movement. To support Micro-Segmentation, George (2025) has presented an extensive review of Micro-Segmentation. Micro-segmentation allows organizations to enforce least-preference access control at the most granular level, as well as facilitate information flow in a manner that maintains confidentiality and integrity. Another emerging area of study on the ZTA is the intersection of Artificial Intelligence (AI) and ZTA. Gadkari (2024) provided an overview of the various applications of AI to ZTA, including but not limited to, allowing for on-the-fly, risk-based access determination and implementing dynamic anomaly detection rather than solely relying on static policy enforcement. Although these articles provide some foundational theoretical/architectural perspectives on ZTA, they do not specifically address the process of continuously verifying that the ZTA controls are functioning as intended.

2.3 Penetration Tests and Validating Security

Penetration testing is one of the most common ways to validate the security of IT environments. The penetration testing process has changed understanding of the unique aspects of cloud computing has improved. In their paper, Joshi et al. (2025) provide a detailed analysis of the Vulnerability Assessment and Penetration Testing (VAPT) of web-based cloud hosted applications. They have classified the various types of tools and techniques to assess the security of a cloud-hosted web application, including the increase in the automation of vulnerability assessment due to the high rate of change of cloud computing.

Okika et al. (2025) have taken this idea broader by discussing how quantum computing will change vulnerability assessments so that penetration tests can also assess the vulnerabilities of cryptographic systems. Okika et al. (2025) propose that Zero Trust Principles will provide additional protection to cyber defenses. They are conceptualizing how Zero Trust and Penetration Testing can go together but do not provide a means of doing so in an integrated automated way. The DEVLOCK principle must also be incorporated into the design phase of software development to ensure that security testing is an integral part of CI/CD. Nagasundari et al. (2025) completed an extensive review of the threat models, in which Automated Security Scanning and Testing are used in CI/CD. These authors provide evidence of the growing use of Automated Offensive Security Testing within the modern information technology (IT) environment.

Despite a large amount of research in the areas of multicloud security, Zero Trust, and Penetration Testing, there is a notable void in the intersection between the three. Significantly, as many researchers recommend implementing a Zero Trust approach (Bishukarma, 2023) or highlight the significance of VAPT (Joshi et al., 2025), few have proposed any type of practical end-to-end automated system that combines all three methodologies into an integrated, cohesive feedback loop. Most research into actual creation of ZTA has been concentrated upon defining and enforcing the policies that comprise ZTA, with validation of whether or not ZTA is appropriately being implemented frequently being considered a separate, subsequent procedure. Conversely, the primary function of Penetration Testing has been focused mainly upon the discovery of vulnerabilities, with the remediation of those vulnerabilities frequently being a manual process. This project will create a framework that integrates Automated Penetration Testing into the Zero Trust lifecycle as the actual mechanism through which Automated Penetration Tests can provide verification data necessary to trigger the adaptive, automated enforcement actions demanded by ZTA, thus advancing the concept of how to create self-validating, self-hardening secure systems in the multi-cloud space.

Table 1: Comparative Analysis of Related Works

Study / Author(s)	**Primary Focus**	**Scope**	**Integrated Automation (Policy -> Test -> Enforce)**	**Identified Gap / Limitation**
Sarkar et al. (2022)	Comparative review of Zero Trust Network (ZTN) models in the cloud.	Theoretical Review	No	Lacks a practical, implemented framework for validation.
Joshi et al. (2025)	Survey of Vulnerability Assessment and Penetration Testing	Survey / Methodological	No	Focuses on testing tools and techniques; remediation is

	(VAPT) for cloud apps.			treated as a separate, manual process.
Adanigbo et al. (2024)	Proposing a Zero Trust architectural framework for multicloud microservices.	Architectural Proposal	No	Provides a strong conceptual design but lacks empirical validation against simulated attacks.
This Project	Integrating Zero Trust with automated penetration testing for continuous validation.	Empirical Implementation	Yes	Creates a complete, automated feedback loop from policy definition to real-world validation and enforcement.

3 Research Methodology

A research methodology based on scientific experimentation was selected to provide for a practical application of a functional prototype (an Automated Security Framework) that would evaluate the prototype under controlled, repeatable conditions.
This research methodology consists of four defined stages: creating the environment in which to perform the tests; creating the Automated Security Framework; executing the experiment; and analysing the results. Adopting a structured process provides for the completion of all research with a scientific basis to ensure that the results are quantifiable; based on real-world observations of what the Automated Security Framework accomplished.

3.1 Phase 1: Environment Provisioning

The first step in this multicloud initiative is to set up a simulated environment so that we can evaluate this framework. Selected two of the largest public cloud providers, AWS and Microsoft Azure, to make sure findings are applicable to what many companies are doing with multicloud solutions today. A programmatic approach to establishing the environment was used to promote consistency and reproducibility in the installation process.
For AWS, created the environment in the us-east-1 region with the following components: A custom Virtual Private Cloud (VPC), several subnets, a gateway to the Internet, and an EC2 instance with Amazon Linux 2023. For Azure, created the environment in the "eastus" area, within a dedicated resource group, with a Virtual Network (VNet), several subnets, a Network Security Group (NSG), and a virtual machine with Ubuntu Server. The purpose of the controlled environment was to provide the opportunity to test various elements of the multi-cloud security policies across various cloud environments with a focus on penetration testing techniques in addition to normal cloud operations.

3.2 Phase 2: The Automation Framework

The framework will live in a single modular Python application and will use industry-standard SDKs to interact with the cloud - boto3 (for AWS) and azure-sdk-for-python (for Azure). A modular architecture was chosen, where each of the frameworks has distinct components that represent the different phases of the security workflow. This modular design allows for more straightforward maintenance and enables a single module to be run independently from the others. Several core Python libraries were chosen to implement the framework (e.g., paramiko (for SSH) and pandas, matplotlib, seaborn (for data analysis)). These libraries will be specified in a requirements file so that developers can easily manage their dependencies. The framework will execute as a single orchestrated pipeline from the time of initial setup to when data is returned to the user.

3.3 Phase 3: Experiment Execution

Conducting Automated Experiments with The Developed Framework Is the Main Focus for Phase 3 Of the Research Process (Experimentation). The Workflow for The Primary Experiment Is Composed of Multiple Stages Simulating an End-To-End Validation and Enforcement Cycle for Security, As Follows:

1. Validating Connectivity and Authentication to Connect to The AWS And Azure APIs.
2. Conducting A Complete Discovery of all the Network and Compute Level Resources in The Provisioned Environment, Collecting A List of All Available Assets.
3. Applying A Baseline Zero Trust Security Policy by Configuring Security Groups and Network Security Groups with A Default "Deny All" Rule for Ingress and Egress.
4. Simulating A Network-Level Vulnerability by Running a Script to Open Several High-Risk Ports (Telnet, RDP, MySQL) In The Azure Firewall Configuration To Allow For Testing Whether The Framework May Recognize A Vulnerability.

Network penetration testing involved conducting an automated network penetration test against the public IP address of all identified VM's, consisting of a port scan to identify services that were publicly available and vulnerabilities that would allow attackers to access them.

The results of the network penetration test were reviewed to find any unauthorised open ports, after which the framework automatically took action to close these ports using pre-determined rules.

The framework-maintained logs and metrics during the entire process and consolidated the log data and metrics in the final stages to completely analyse the tests performed against the VM's, calculate a security score for the VM's and create visualisations and recommendations based upon that analysis.

3.4 Phase 4: Data Capture and Evaluation

So, each step in the experimental process had collected all the data available as follows: The AWS API response data; the Azure API response data; all output information from penetration tests performed with various tools within the Lab; and all internal states for every module executed by the Framework were collected as structured JavaScript Object Notation (JSON) files. This method allows us to develop a comprehensive audit trail of everything that occurred throughout this entire experiment. Once the data has been captured and stored, Phase Four will parse the stored JSON files to enable us to quantitatively assess how well the Framework is able to execute against the Zero Trust model, including key metrics related to asset discovery (number of assets identified), vulnerability assessment (number of vulnerabilities identified), remediation (number of remediation activities performed), module execution time (average time taken for completion of each module), and the final security score calculation (i.e., an overall quantitative assessment of threat detection capabilities). To evaluate the results and make conclusions regarding the integration between Zero Trust and automated penetration testing, statistical and visual analysis techniques were utilized.

4 Design Specification

This solution consists of a modular and orchestrated framework for automating the Zero Trust Security Lifecycle for multicloud environments. This design focuses on the automation of each part of the security validation/enforcement process, making it more repeatable and reliable using modules. There are 7 components in the framework that each serve a different purpose in the security workflow and are coordinated via a main orchestrator.

4.1 System Architecture

The overall architecture consists of a "Main Orchestrator" script (main.py) that runs each module in a specific order. The architecture has a flow in which the results from the module run first, the results from Module 1 (environment setup) are the inputs for Module 2, and so forth until the results of the last module (analysis of final assessment) are the inputs for the Final Analysis module. The outputs of all modules are saved as structured JSON files in an output directory that serves both as an audit trail and database for the inputs for the other modules, therefore separating the modules and enabling independent evaluation of the results from all modules.

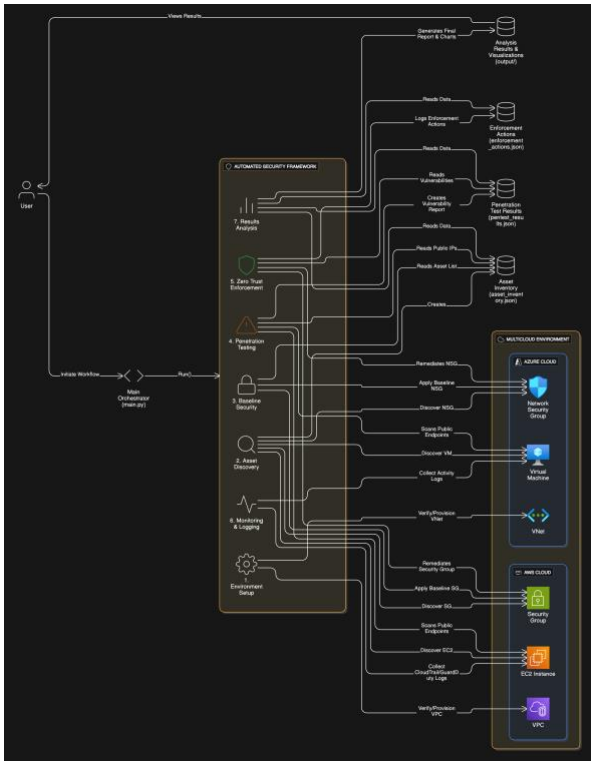

Figure 1: System Architecture

The seven basic modules of the system are: Environment Setup, which initializes and verifies the multi-cloud environment and establishes and tests connections to AWS/Azure using many types of pre-configured credentials.

1. Environment Setup programmatically constructs foundational network infrastructure (e.g., AWS Virtual Private Cloud and Azure Virtual Network; or encodes the concept of network sharing) for the purpose of developing an isolated, standard test bed for future Security Experiments.
2. Resource Enumeration (2nd Module) queries Cloud Provider APIs to Discover and Docs all the major Assets in both cloud environments. Resources include: EC2 Instances/Azure VMs; Security Groups/NSGs; VPCs/VNets. The output of Module 2 (the Asset Inventory document) contains everything required to begin the Security Experiment engagements.
3. Fundamental Security: This module implements the Zero Trust standard for "denial of all and everything by default" (fundamental safety). It collects and creates a baseline security policy for the discovered infrastructure in the cloud. For example, in AWS it will create a `zero-trust-sg` security group with restrictive ingress and egress rules, and in Azure it creates a corresponding `zero-trust-nsg` network security group. The fundamental purpose is to establish a severe fundamental security posture prior to performing penetration testing.
4. Penetration Tests: The Penetration Test Module provides the framework for validating the Zero Trust architecture and the establishment of the baseline security posture. This module collects and isolates the public facing assets in the asset inventory (i.e., those VMs with a public IP address) and runs automated penetration tests against these

assets. The primary method used for testing the assets is a network port scan that identifies the state of commonly and highly accessed ports. This technique creates a simulated external attack to validate the accessibility to the resources. The information from the scan, including the ports discovered to be open, will be recorded and treated as potential weaknesses.

5. The Zero Trust Enforcement module is defined by its ability to respond to incident events. It uses the results from the Penetration Testing Module to identify any vulnerabilities that have been found and apply a set of remediation rules. It has been created with a focus on automating the update and enforcement of firewall rules to block any inappropriate ports that may be opened because of penetration testing. In the example of SSH, the goal is to be more cautious and simply flag the vulnerability instead of blocking SSH access outright because this would create downtime for operations.
6. Monitoring and Logging is designed to gather thousands of automated security-related logs/events from multiple cloud platforms for ongoing visibility. It uses in-house cloud provider monitoring (AWS CloudTrail, AWS GuardDuty) to create a connection to collect API Logs and Threat Detection (monitoring) event traffic. By doing this, Monitoring and Logging allows for correlation of penetration results against real-world logs/events created when using cloud platforms.
7. The results analysis stage combines data from the previous stages to create a global view of the organization's security posture. In addition to calculating performance metrics, like Total Number of Vulnerabilities Detected, Total Remediation Rate and Overall Security Score, the data collected during the previous stages enables the generation of graphical representations, such as Charts of all Vulnerabilities Categorizations by Severity and Cloud Service Provider as well as a Comprehensive List of Security Recommendations that are Actionable.

The Modular Approach allows for a fully integrated system that provides a complete end-to-end automated security feedback loop by establishing the Penetration Testing Module as the verification point between the Policy Definition (Baseline Security) and Policy Enforcement (Zero Trust) Components.

5 Implementation

The framework utilizes Python for its robust set of libraries available to support cloud-based automation and data analytics (Version 3.9+). Each module was built using an independent Python script, with the central manager for coordinating all modules via main.py.

5.1 Tools/Technologies Used

All functions within the framework are based on the following major technologies:

- Cloud SDKS: The boto3 library will connect to the AWS API to manage EC2, VPCs, and Security Groups. For Azure, the azure-sdk-for-python package group (Includes

the azure-mgmt-compute library, the azure-mgmt-network library, and the azure-identity library) were used to manage VMs, VNets, and NSGs in the cloud.

- Penetration Testing Tooling: The network port scanner function of the penetration testing module was constructed using the basic socket Python library. This allows for lightweight, nonproprietary checks on TCP communications without having to use a third-party library such as Nmap to implement the port checks, allowing the framework to be as self-sufficient as possible.
- Handling and Analysing Data: The main purpose of using the pandas library is to manipulate and analyse data. Data manipulation and analysis were carried out using Pandas; the results from each module were structured using this library. The final analysis module of each module used matplotlib and seaborn in order to produce static, PNG visualisations of the security data's analysis of the module.
- Configuration Management: All sensitive information (e.g., API keys, account IDs, resource names) and environment-specific settings were stored in separate, external files, such as aws_config.json and azure_config.json. This allows for the maintenance of the configuration separately from the code, increasing both security and maintainability of the module.

5.2 Modules Implementation Details

All modules were implemented according to the design specification, and each module executed its assigned function. The workflow of the application was managed by the main.py script ("Main"), which created and called each module class sequentially. The total execution time of the full workflow was approximately 66.12 seconds; this indicates that the automated workflow is efficient.

The Penetration Testing module is noteworthy in how it will have a huge impact on how businesses view their security measures. For example, this module starts by taking the asset_inventory.json file to look for all virtual machines with assigned public IP addresses, which end up being identified as scan targets. Then for each target every single entry in a predetermined list of common TCP ports (22 TCP, 23 TCP, 80 TCP, 445 TCP, 3306 TCP, 3389 TCP) is iterated over. A TCP connection is attempted; connections are made to each of the ports with a short timeout if successful indicating an "Open Port" and logged as a candidate for a Vulnerability. The end result (a list of open and closed ports for each Host) will be saved in a pentest_results.json file; within the pentest_results.json, each open port will be identified as a potential vulnerability and the severity rating will have been attached. (For instance, SSH port 22 "LOW" ⇒ Telnet port 23 "Critical").

The purpose of the Zero Trust Enforcement module was to implement a conservative logic strategy that would minimize the potential for service disruptions in production-like environments. The module reviews the pentest_results.json file for vulnerabilities and evaluates the port number associated with each vulnerability. In the case of critical ports associated with administrative services, such as SSH (port 22), the system logs a warning and suggests the need for manual review, rather than taking automated actions to block access. For all other non-critical/high-risk ports, the module contains the logic to automatically create

firewall rules to deny access to inbound connections. The conservative-by-default approach was a select design choice for developing an automated remediation system.

The Results Analysis module uses the various JSON outputs to produce a comprehensive report detailing the security assessment for the organization. This module calculates a security score starting from a maximum of 100 points and deducting points based on the severity of any un-remediated vulnerabilities that exist on the network. Low severity vulnerabilities will result in a deduction of 1 point from the total score. This quantification gives a simple visual indication of the overall security posture of the organization. The Results Analysis module will produce visual reports using the matplotlib library and will save the reports as PNG files, providing an easy way for users to consume the report.

6 Evaluation

The purpose of this section is to explain in detail the results produced by implementing the automated security framework. The results evaluation will follow the experimental process from the beginning through the final Security Assessment of the Penetration Testing (PT) Module and the Enforcement (Enforcement) against PT results.

6.1 Experimental Environment and Initial State

The experimental conditions and the beginning environment of this experiment consisted of a Multicloud Environment that included resources located in both AWS and Azure. In AWS, there was one active EC2 instance (Ajay_EC2 (i-0b9743f0724832f6a)) which had a t2.small instance type; whereas, in Azure there was one VM (zero-trust-vm) with a public IP Address of 40.71.76.1 provisioned.

The Asset Discovery module successfully identified both resources within this multicloud environment. During the Asset Discovery phase, 1 AWS EC2 Instance and 1 Azure VM were discovered as well as all their related Network Components (i.e., 6 AWS Security Groups & 3 Azure Network Security Groups). All these findings validated the ability of the Automated Security Framework to achieve cross-cloud visibility and create an initial asset inventory (14.82 sec).

The Baseline Security Module applied the restrictive Zero Trust policy after its discovery. This involved creating Security Groups (SGs) and Network Security Groups (NSGs) with a "deny-all" default rule with the intention of establishing a secure starting point for the initial operational environment. Automating the application of these baseline policies took 10.90 seconds.

6.2 Deliberate Vulnerability Introduction

A critical part of the study experiment was the planned introduction of artificial vulnerabilities to test the detection and the response capabilities of the system. The

application of a script (add_azure_vulnerabilities.py) was made to change the Azure NSG (Zero Trust NSG) that was linked to the zero-trust-vm. The script was created to allow access to multiple high-risk ports via the Internet, as follows:

- Port 23 (Telnet) -Critical
- Port 445 (SMB) -Critical
- Port 3306 (MySQL) -Critical
- Port 5432 (PostgreSQL) -Critical
- Port 3389 (RDP) -High

According to the execution log for the given script, the above-mentioned rules were successfully added to the system. Afterward, an insecure configuration was manually created to demonstrate a situation in which the baseline was not adhered to as would be expected in a real-world security misconfiguration.

6.3 Penetration Testing Execution and Key Findings

The Penetration Testing module was developed to assess an organization's actual Security Posture, regardless of what Configuration is intended. The Penetration Testing Module completed its scans in approximately 13.67 seconds.
The Penetration Testing Framework identified the Amazon EC2 instance, Ajay_EC2, with the public IP address of 54.196.176.91 as the target of the scans. The Penetration Testing Framework identified one open port on the Amazon EC2 instance during its Automatic Port Scan.

- Open Port: 22 (SSH) – Open

All other Ports scanned (including high-risk Ports such as Telnet (23) and RDP (3389)) were closed, and the Framework classified the open port on the EC2 instance as a low severity vulnerability, because it is used for administrative access and can easily be accessed from the public side of the network.

Scanning the Azure virtual machine zero-trust-vm using public IP 40.71.76.1 produced a remarkable and noteworthy finding. Although previous steps had added numerous CRITICAL and HIGH firewall rulesets, and there were many open firewall ports, 22 (SSH) was detected open through the testing, whereas all of the other intended ports (including Telnet, SMB, RDP) were all detected closed by the scan.

Thus, this discrepancy represents the most significant takeaway from this experiment - it demonstrates a conflict between two different processes that are executing simultaneously - the manual vulnerability introduction script and the automated baseline security module. The execution logs indicate that the baseline security module was executed after the creation of the initial virtual machine and very likely overwrote or replaced the NSG that contained the intentional exposure rules applied to the VM. The penetration test results indicated not the

intended state (as defined by the firewall rules), but the actual state that can be determined externally.

The significance of this finding illustrates the additive value of using penetration testing to validate security through a Zero Trust framework. A security audit using only a static review of firewall rules would erroneously classify the Azure VM as a highly vulnerable asset, whereas an automated penetration testing solution gave a true representation of the Azure VM's status by identifying that the baseline security policy had mitigated the previously identified vulnerabilities and produced a much-improved level of security. The validation process is an essential element in mitigating the gap between a formally defined security policy and Security Policy Enhancements in the real world.

6.4 Automated Enforcement and the Final Posture

Automated Enforcement (AE) utilized the results of the penetration test to determine a Final Posture of AE based on the penetration testing results. The results of the automated penetration test contained two identified vulnerabilities - one from the AWS instance with one open SSH port and one from the Azure VM with one open SSH port.

After determining the Final Posture for the project and determining that one open port (22) was critical to maintain, the AE analysed the findings of the penetration test using the AE Internal Logic. AE was developed to restrict instead of blocking access to Port 22 and therefore the AE framework did not close the port automatically.

The execution logs demonstrate the conservative approach taken by our framework, as evidenced by the "0" NSG rule addition and the "0" Security Group Update CIDR entries. As a result, the remediation rate for this experimental run is calculated at 0.0%. While this may appear to be a failure, it was the desired result resulting from a safety-first approach. The framework recognized this risk and postponed any remediation action for this service because it was too critical to automatically block. The resulting security posture will be equivalent to what was established by the penetration test findings.

6.5 Overall Performance and Security Analysis

The Results Module combined all the collected data into a single report, with one of the main outcomes being the overall security score. Starting from 100, Figure 4 represents the overall security score was decreased by 1 for each of the two "LOW" severity vulnerabilities identified as per Figure 2 during our testing phase (the open SSH ports). Therefore, as per the overall security score is 98 out of 100, which would be rated as an "EXCELLENT" overall security score. The primary reason for this high rating is that the baseline security policy was successful in preventing any of the more serious vulnerabilities created manually from being introduced. As per figure 5, two Vulnerability Remediation Effectiveness has been found and has had been secured efficiently.

The framework produced multiple visualizations, which provide an overview of the findings.

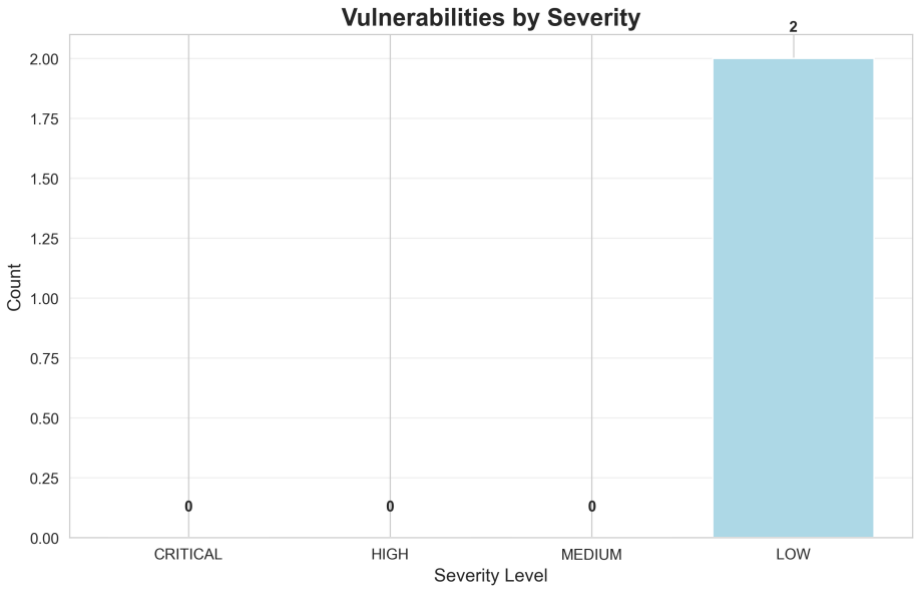

Figure 2: Vulnerabilities by Severity

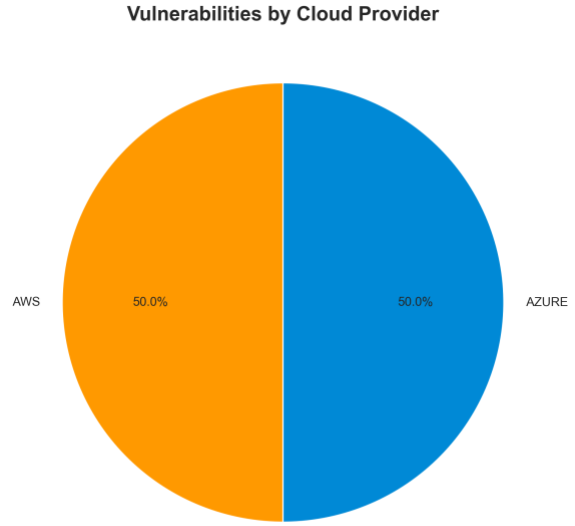

Figure 3: Vulnerabilities by Cloud Provider

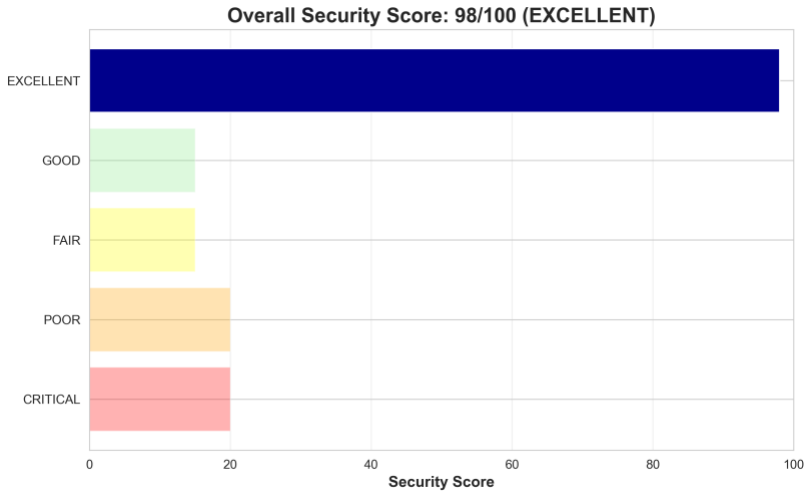

Figure 4: Overall Security Score

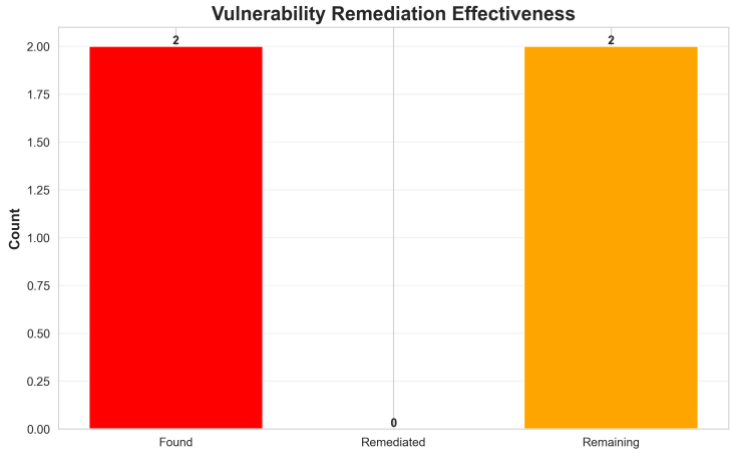

Figure 5: Vulnerability Remediation Effectiveness

Table 2: Summary of Experimental Security Metrics

Metric	AWS Target (Ajay_EC2)	Azure Target (zero-trust-vm)	Overall Framework Outcome
Public IP Address	54.196.176.91	40.71.76.1	2 Public Targets Identified
Intended Vulnerabilities	N/A (Baseline Secure)	5 (Critical/High)	Test scenario created
Discovered Open Ports (PenTest)	1 (Port 22/SSH)	1 (Port 22/SSH)	2 Total Vulnerabilities Found
Identified Severity	LOW	LOW	All Critical/High risks were not exposed
Automated Remediations	0	0	0.0% (By design for critical ports)
Final Security Score	-	-	98 / 100 (Excellent)

6.6 Discussion

The current research shows that Automated Penetration Testing, when implemented into a Zero Trust Security Workflow, may be a highly effective strategy for establishing a customer's Security Posture within the organization and the subsequent evaluation of that Security Posture by a third party. The clear success of using Automation Penetration Testing to demonstrate the 'real-world' (physical) security posture versus just a Configuration Audit is one of the more beneficial parts of the Automation Penetration Testing framework. With a clear contrast between what is wanted for vulnerable state on the Azure VM and what it is at this time demonstrates the framework's effectiveness in eradicating "false security beliefs". Furthermore, this supports the notion that the Automated Baseline Security Policy has been put in place and is functioning properly. The individual pieces of the penetration testing system were limited, in that they only served to identify the network attack surface area for the resources deployed by the organization. However, there is a definite opportunity to further expand this capability by including detection of service versions, known vulnerabilities (for example, CVE's), which will greatly assist to establish a deeper understanding of an organization's overall Security Posture (Joshi, Ajmeri, & Joshi, 2025).

As evidenced by the 0.0% remediation rate of the experiment, there is a tradeoff between the security and accessibility of any automation enforcement solution implemented in a production environment. For example, the conservative method used to address the problem of open SSH ports was a conscious design feature to avoid the occurrence of administrators being inadvertently locked out of their own systems by the enforcement solution itself. If deployed into the real world, the conservative approach could potentially be expanded to include more adaptive capabilities for example, by having the enforcement solution limit the SSH access only to the known corporate source IP address range, rather than allowing open access to all IP addresses.

The test successfully met its goal of showing that an automated solution can identify and classifying company IT assets, providing and enforcing a Zero Trust policy, and using penetration testing as a method for verifying and reporting on the evolving security status of a Multicloud service provider.

7 Conclusion and Future Work

The goal of the research conducted for this project was to evaluate how Zero Trust security principles could be combined with automated network penetration testing to continuously validate and enhance the security posture of multicloud environments. The design, implementation and evaluation of the automated framework described in this report provide clear and unequivocal proof that the integration of Zero Trust and automated network penetration testing is viable and necessary for establishing a verifiable and adaptive security posture in the increasingly complex and borderless environment of multicloud computing.

The primary conclusion of this research was that automated penetration testing provides a critical component in validating the security state of an organisation. This experiment showed

that are only using configuration management or policy-as-code as the basis for measuring organisation's security status, may very well find in a situation where believe organisation is secure while it is not. Automated penetration testing provided the ultimate measure of truth by defining the organisation's actual external attack surface and validating the effectiveness of the Zero Trust security model in the presence of misconfigurations. The framework was able to execute a complete end-to-end security lifecycle in less than one minute, resulting in a security score of 98 out of 100, demonstrating the validity of using an integrated approach to security.

This study primarily targets practitioners responsible for delivering cloud and multi-cloud security; it also presents a workable pathway away from static, declarative security policies and toward a more continuously validated security model through automation of the feedback loop between policy application, offensive testing and enforcement, thereby enabling organisations to create more robust, self-hardening systems that are better suited to adapting to rapidly changing IT environments.

While this study has achieved its stated goals, it has several limitations. Penetration-testing strategies employed in this study were rudimentary and only included basic network port scanning techniques, as well as a conservative enforcement logic that did not include complex remediation procedures. Additionally, a very basic two-Virtual Machine environment was used rather than more complex Cloud services such as serverless function hosting platforms or container orchestration solutions.

7.1 Limitations and Opportunities

These limitations offer many opportunities for further research.

- Development of Better Penetration Testing Capabilities: The penetration testing capability of the model needs to be much stronger than it currently is. This includes advanced testing capabilities, such as the ability to identify a service's version, the ability to perform authenticated and unauthenticated vulnerability scans against popular CVE databases and perform basic application-layer testing of well-known web-based vulnerabilities, i.e., SQL Injection and Cross-Site Scripting.

- Implementation of Artificial Intelligence-Based Enforcement Logic: The existing rule-based enforcement capability could be improved using Artificial Intelligence. By utilizing machine learning, an AI-based enforcement solution would be able to evaluate the context surrounding detected vulnerabilities, enabling it to create a more refined remediation plan. In addition, AI could potentially integrate with threat intelligence feeds to assist in determining the priority of response actions (Gadkari, 2024).

- Coverage Expansion to Other Cloud Services: The current framework could potentially be improved to allow the detection and testing of a larger array of cloud services beyond traditional virtual machine environments, including Object Storage (e.g., S3 and Blob Storage), Database (e.g., RDS or Azure SQL) and Containerization Platforms (e.g., EKS and AKS) which provide unique attack surfaces.

- Enterprise Adoption through the Integration of the Framework with Security Information & Event Management (SIEM) & Security Orchestration, Automation &

Response (SOAR). The Framework will serve as a high-fidelity alert source for Security Information & Event Management (SIEM) and as a trigger // management of enforcement action through a Security Orchestration, Automation & Response (SOAR) Workflow.

In Summary: The project completes a very important contribution by providing a practical (automated) & integrated approach to Zero Trust Security Validation in Multi-Cloud, then communicates a transition between theoretical policy & practical Security, creating a better opportunity for an on-going validated Cloud Security posture.

References

Abid, N., 2024. Advancements and best practices in data loss prevention: A comprehensive review. Global Journal of Universal Studies, 1(1), pp.190-225.

Adanigbo, O.S., Adekunle, B.I., Ogbuefi, E., Odofin, O.T., Agboola, O.A. and Kisina, D., Implementing Zero Trust Security in Multi-Cloud Microservices Platforms: A Review and Architectural Framework. ecosystems, 13, p.14.

Arif, T., Jo, B. and Park, J.H., 2025. A Comprehensive Survey of Privacy-Enhancing and Trust-Centric Cloud-Native Security Techniques Against Cyber Threats. Sensors, 25(8), p.2350.

Bishukarma, R., 2023. Scalable zero-trust architectures for enhancing security in multi-cloud SaaS Platforms. Int. J. Adv. Res. Sci. Commun. Technol, 3(3), pp.1308-1319.
Gadkari, B.R., 2024. AI Integration in Zero Trust Security Architecture: A Technical Overview.

George, A.S., 2025. The Critical Role of Micro-segmentation in Modern Cybersecurity Architectures: A Comprehensive Review.

Gonsalves, P., Salesforce Einstein Copilot for Multi-Cloud CRM Integrated With Secure LDAP Authentication.

Gupta, S., 2025. HYBRID CLOUD INTEGRATION AND MULTICLOUD DEPLOYMENTS A COMPREHENSIVE REVIEW OF STRATEGIES, CHALLENGES, AND BEST PRACTICES. International Journal of Advanced Research in Computer Science, 16(2).

Harper, C., 2025. A COMPREHENSIVE REVIEW OF DATABASE SECURITY THREATS AND MITIGATION STRATEGIES IN CLOUD ENVIRONMENTS.

Ibrahim, R., Khider, I., Edam, S. and Mukhtar, T., 2025. Comprehensive Strategies for Enhancing SD-WAN: Integrating Security, Dynamic Routing and Quality of Service Management. IET Networks, 14(1), p.e70007.

Ilochonwu, I.A., Cloud security paradigms: A systematic review of threat mitigation strategies in cloud-based applications.

Jaffal, N.O., Alkhanafseh, M. and Mohaisen, D., 2025. Large language models in cybersecurity: A survey of applications, vulnerabilities, and defense techniques. AI, 6(9), p.216.

Joshi, A., Dumka, A., Singh, D.P. and Semwal, A., 2025, November. Comprehensive Survey on Vulnerability Assessment and Penetration Testing (VAPT) for Cloud-Hosted Web Applications. In Proceedings of the International Conference on Sustainable Business Practices and Innovative Models (ICSBPIM-2025) (p. 278). Springer Nature.

Kunduru, A.R., 2023. The perils and defenses of enterprise cloud computing: a comprehensive review. Central Asian Journal of Mathematical Theory and Computer Sciences, 4(9), pp.29-41.

Mudau, K., Mudumani, K. and Zwane, S.M., 2025. Zero Trust Architecture: Frameworks and Implementation Strategies in Modern Cybersecurity. Available at SSRN 5637091.
Nagasundari, S., Manja, P., Mathur, P. and Honnavalli, P.B., 2025. Extensive Review of Threat Models for DevSecOps. IEEE Access.

Okika, N., Nwatuzie, G.A., Olarinoye, H.S., Nwaka, A.A., Igba, E. and Dunee, R., 2025. Assessing the vulnerability of traditional and post-quantum cryptographic systems through penetration testing and strengthening cyber defenses with zero trust security in the era of quantum computing. International Journal of Innovative Science and Research Technology, 10(2).

Radanliev, P., 2024. Integrated cybersecurity for metaverse systems operating with artificial intelligence, blockchains, and cloud computing. Frontiers in Blockchain, 7, p.1359130.

Sarkar, S., Choudhary, G., Shandilya, S.K., Hussain, A. and Kim, H., 2022. Security of zero trust networks in cloud computing: A comparative review. Sustainability, 14(18), p.11213.
Singhania, V., A Review of Cloud-Based Data Security Protocols.