National College of Ireland

Benchmarking Disaster Recovery Strategy in Multi-Cloud Setups using CIS Benchmark

MSc Research Project
Cloud Computing

Chidiebube Okeke
Student ID: x24109487

School of Computing
National College of Ireland

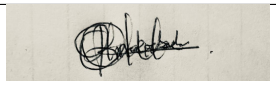
Supervisor: Yasantha Samarawickrama

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Okeke Chidiebube
Student ID:	x24109487
Programme:	MSc. Cloud Computing
Year:	2025
Module:	MSc Research Project
Supervisor:	Yasantha Samarawickrama
Submission Due Date:	11/12/2025
Project Title:	Benchmarking Disaster Recovery Strategy in Multi-Cloud Setups using CIS Benchmark
Word Count:	6622
Page Count:	23

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:	
Date:	11th December 2025

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):	

Benchmarking Disaster Recovery Strategy in Multi-Cloud Setups using CIS Benchmark

Chidiebube Okeke
x24109487

Abstract

Organizations are continuously evaluating their business processes for efficiency, and resilience against best practices. With the advantages of multicloud, such as reduced vendor lock in over single cloud, and portability for businesses, disaster recovery implementation, where each cloud provider tools, architecture, hardware virtualization in multicloud can be challenging while maximizing security. This research identifies this gap by implementing a reproducible framework that evaluates disaster recovery (DR) strategy across Azure, and AWS cloud integrated with Center for Internet Security (CIS) Benchmark for standardized security compliance, and audit. The research uses Terraform IaC tool for orchestrating automated provisioning of Variant 1(basic security),and Variant 2(CIS Level 1 hardening) on AWS, and Azure. The conducted experiment measures recovery metrics, security compliance, comparative analysis across clouds. The result demonstrates an almost linear relationship between CIS Security Compliance, and Recovery Time Objective(RTO), and zero security degradation during failover. With the properly documented findings, this research provides a realistic understanding of a valid framework for security-aware multicloud DR suitable for organizations, and cloud consumers.

1 Introduction

Cloud computing has been around for a reasonable period of time now and is no longer considered a new practice for complex information processing and storage. Despite known setbacks, the potential of cloud computing has been realized, and impactful for business scalability, high performance computing (HPC), AI training, and like every invention its current state is fine tuning of different aspects to overcome bottlenecks. At this stage, organizations have permutation-ed the strengths, and weaknesses of cloud solutions, and resources from various cloud providers, and their findings significant advantage for a heterogeneous cloud over a single cloud.

1.1 Multicloud (Merit, and Demerits)

According to Kotha (2023), multicloud is defined of the distribution of computing workloads across different cloud providers to satisfy business operation needs, while promoting flexibility using different methods such redundancy across clouds, distributed workloads, and cloud bursting, each with their pros, and cons. Caceres and Globa (2022) explored potential solutions to the gaps of emerging beneficial multicloud architectures, and how

manual adapters based on third party must be built because. Each cloud provider differs in operation, APIs, and interface. This heterogeneous overhead when handling each cloud provider's uniqueness affects multi-cloud users significantly according to studies Saraswat and Tripathi (2020).

1.2 Disaster Recovery in Multicloud

When dealing with multicloud for operational resilience, and mitigating vendor lock in from single cloud, effective disaster recovery implementation needs to be in place to ensure business continuity. Mankotia A. (2024) compartmentalizes in theory how multicloud high availability and disaster recoveries can be achieved using industry techniques. These techniques constitute redundancy, load balancing, automated failover, and geographical distribution (for data compliance, and regulation) techniques. However, this comprehensive theoretical analysis, and other studies have little or no consideration for security in their framework for multicloud.

1.3 Motivation

Multicloud is growing into mainstream computing, and recovery parameters such as Recovery Time Objective (RTO), Recovery Point Objective (RPO) alongside security concerns need to be prioritized as ever. CIS Benchmark[1] is an industry benchmarking tool that is capable of security configuration, and hardening templates for cloud providers. Organizations still lack a practical, reproducible way to benchmark DR performance and verify security compliance across public clouds, and these gaps form the foundation of this work. This research work aims to bridge that gap by integrating DR performance metrics using CIS Benchmark security assessment (that aligns with major security frameworks) under automated failover in real world conditions.

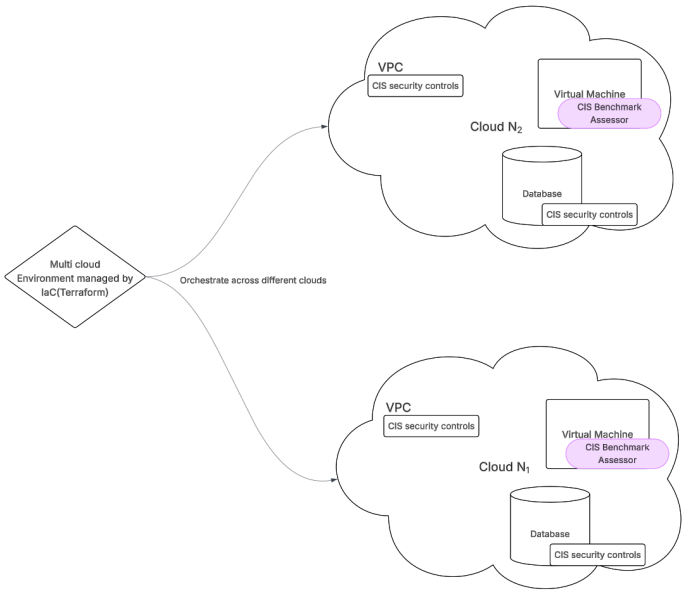


Figure 1: Proof of Concept with MultiCloud, and CIS-CAT Controls

[1]CIS Benchmark: `https://www.cisecurity.org/cis-benchmarks`

The primary research question of this study is "How can the CIS Benchmark be integrated into a framework to evaluate the performance and security of disaster recovery strategies in multi-cloud environments".

Some of the research objectives are to implement and test Active-Passive DR configurations across AWS and Azure with varying security postures, collect recovery metrics, integrate CIS assessments, analyze whether security hardening impacts recovery time multi-cloud disaster recovery. The research question, and objectives, all together contributes to the scientific literature through a real-world analysis that integrates CIS Benchmark into multicloud DR strategy for evaluating security overhead against system recovery, and cross cloud performance comparison.

An outline of the structure of this report in order are Related works, Research Methodology, Design Specifications, Implementation, Evaluation, Discussion, and finally Conclusion.

2 Related Work

Disaster recovery primarily fosters business continuation by effective planning. A qualitative study by Syed and Ali (2021) covers the growth of business disaster recovery (BDR) implementation from its preliminary stages that were practically expensive, especially for startups. Fernando (2017) discusses a five step rudimentary procedures that can be used by businesses to ensure continuity during disasters, Mazumder and AbdAllah (2023) took further step to propose an Information Centric Networking (ICN) technology for existing DR management system due to the excess economic loses in Bangladesh(a disaster prone country). These dilemmas are reasons for continuous improvement or development of DR strategies.

2.1 Disaster Recovery Concepts in Cloud Computing

DR implementation is constantly evolving to become a traditional strategy for small, and enterprise businesses, there are known variables that impact the integration and future maintenance of DR plans, According to Alhazmi and Malaiya (2013) research work, that discusses Recovery Time Objective (RTO), Recovery Point Objective (RPO), and costs as key parameters. Although their study only dictates DR scenarios where public cloud offerings prove to be more flexible than on-premises DR plan, several of the foundational ideas in their research formulate the background of this research topic. Also, Puri et al. (2019), discuss the operational nature of the public cloud models, and how cloud users should make plans including DR strategies around cloud advantages, and related risks.

The general need to ensure business continuity and requirements demands resilient architectures from cloud platforms. A real-world requirement is cloud operations with minimal vendor lock-in, which resulted in significant adoption of multicloud implementation. Avoiding vendor lock in makes cloud providers an independent variable for cloud users, the study by Pellegrini et al. (2017) covers a template architecture that covers multi-cloud design, and advantages such as portability, and cloud platform independence. The design in the paper compliments the architecture in the framework of this

research.

DR planning has been made effective by the involvement of cloud providers which has helped practice DR adoption, Google's revenue comes from building large-scale products that are highly available through a dense domain expertise, regional documented outages, and remediation approach documented in the Google's Site Reliability Engineering books Beyer et al. (2016), and Beyer et al. (2018), introducing proactive active reviewing of failover testing, and plans against SLA's. AWS Well-Architected Framework (AWS, 2023)[2] provides a general design principles on reliability, and maintenance. With the rise of DR implementation on these platforms, these publications have definitely made a significant impact but still remain provider-eccentric.

These few literature publications provide a good foundation, and the need for DR plans, but didn't cover any standard approach in evaluating DR performance across multi-cloud.

2.2 Multi-Cloud Approaches to Disaster Recovery

There is evidence that multicloud is a competitive method used for DR planning, as it contributes to performance given its distributed nature across cloud providers. A paper that covers efficient, and resilient data distribution by Sengupta and Annervaz (2012) shows how high availability can be achieved using a Data Distribution Planner for Disaster Recovery (DDP-DR) plan on active multi-DR sites. Sengupta and Annervaz (2014) continued their research and adapted to a novel data distribution framework. Their conducted study proves useful but only models after efficient data distribution.

Abualkishik et al. (2020) documented the state-of-the-art approach plan that can direct a sustainable disaster recovery plan. Their research discusses merits and demerits in relation to cloud providers and cloud users. The study shows the significant advantage of automation tools, but a limitation of their research is that the different DR approaches examined utilize single cloud. Pandurangan (2025) underscores the latest advanced concepts in literature that could drive effective DR through automated orchestration technologies, some of which were used in this research implementation. The literature claimed results with no supporting quantitative analysis.

There has been a previous attempt at a tactical benchmarking approach in multicloud using Google's PerfKit Bench marker[3]. This kit provides a means to access cross cloud instances, networks, and addresses assumptions with cloud providers. A benchmark test on AWS, and Google by Bucur and Miclea (2021) used PerfKit, the literature highlights significant performance metric that does not include RTO, or RPO.

This section shows that there is room for much growth in multicloud DR through extensive study; that goes beyond proof of concept.

[2]AWS Well-Architected Framework: `https://docs.aws.amazon.com/wellarchitected/2023-10-03/framework/welcome.html`

[3]PerfKit: `https://github.com/GoogleCloudPlatform/PerfKitBenchmarker`

2.3 Security and Compliance Considerations in Disaster Recovery

So far, some studies have investigated security concerns, and DR separately, even though security is an essential part of most practices (DR) in the cloud. Fernandes (2014) reviewed quantitative security challenges that concern the field of cloud computing. The paper recognizes that increasing cloud trends adoption is a factor of these security challenges; some of them include identity management, policy enforcement, and data protection that become more difficult in cross clouds. This finding is remarkably similar to Badirova et al. (2023) that analyzed the inconsistent behavior in IAM across different cloud providers.

Liu et al. (2022) literature tackles multi cloud DR security using a solution based on decentralized blockchain. They explored the ethical reasons, especially the auditing problem that should be addressed for single cloud DR security risks. Their main idea leveraged blockchain's intrinsic immutable, traceable features, and encryption to ensure data consistency, and security across clouds. The encryption used remains a bottleneck for the proposed solution.

CIS Benchmarks is a key security tool in its effectiveness prioritizes security goals, and auditing. Some online documentations have discussed how it provides consistent hardening rules to secure infrastructure layers, and even with this, there is no conducted study to deep dive into its operation, especially during disaster recovery. This research recognizes this gap as an opportunity to evaluate if CIS compliance is preserved post disaster recovery in a real world multicloud architecture.

2.4 Orchestration Tools for Disaster Recovery

de Carvalho and Patricia Favacho de Araujo (2020) research contribution acknowledges the domination of multi-cloud concepts, and the recognition of a tool for powering these dynamic computing, their paper analyses several orchestration tools such as Cloud formation, Terraform, Cloudify practical for use cases especially in multicloud. A practical methodology assesses cloud orchestration on resource consumption, execution time for provisioning, with Terraform leading the edge in all conducted experiments. Its methodological process also judges CPU usage by recommending an IaC tool. Also, in the research conducted by Thiyagarajan (2020) Terraform supersedes Ansible, with both research work conducted in the year 2020, Terraform has really been a game changer in recent times.

2.5 Performance, and Benchmarking Challenges

Sunarjo et al. (2025) research theory puts into perspective the individual security factors that may inhibit the growth, and resilience of multicloud functionalities. It explores a weighted relationship between security vulnerabilities and cloud users. According to this study, the identification of security breaches by cloud consumers is dependent on their level of knowledge on potential security risk which can be improved by prioritizing secur-

ity cultures, and best practices.

A comparative study by P. Kaushik et al. (2021), uses a practical approach to assess performance for competitive cloud providers. Their study covers suite tests assessment for VM's on Google Cloud, AWS, Azure. The research conducted these performance tests using Phoronix Test Suite3, in the analysis, AWS succeeded over Azure, Google for the RAM speed tests, and Dbench test, and Azure dominated the Apache. It can be implied that DR strategies may experience varying performance across providers from their recommendations on where each cloud provider performs better. The number of conducted experiments for each testbed may be a limitation alongside the little reference of their benchmark result towards DR multicloud.

Akbari and Hauswirth (2024) highlight the potential bias that occurs in recent and previously conducted cloud benchmark types. The literature covers the obscure nature of the cloud, and designs implementation of these benchmarks. The paper follows a statistical trend of public literature with the aim of improving the general outcome of benchmarking. Despite the paper limitations, the primary aim focuses on improving transparency that can drive quality assessment of methodology, and reproducibility of cloud benchmark domain.

2.6 Research Niche

The reviewed literature standouts as a handful of them pointed out our research methods and designs in the right path. There were papers that explored DR concepts that directly impact single clouds, while some discussed about multicloud performances, this research tends to use models, simulation rather than provisioning resources for real tests resulting in an overly controlled outcome. There was experimental breakthrough in some research around security auditing in multicloud but the outcome was perceived to be computationally expensive. In summary, the research in this paper shows many generalized benchmarks, and frameworks that do not integrate existing performance metrics (RTO or RPO) with security compliance. The specificity of this current research to the multicloud DR domain is to develop a framework that evaluates both practical recovery and security compliance. The outcome is not just a narrative about DR, but a repeatable framework through proper documentation that shows how multi-cloud DR behaves in practice when CIS Benchmarks are part of the design.

3 Methodology

For the arrangement of the multicloud DR setup, essential decisions were taken to guarantee feasibility of the experiment to be carried out by identifying known, and possible limitations in the domain of the research, the first step involved the acquisition of the CIS-CAT Lite tool, this is a free minimized version of the CIS-CAT Pro tool packaged benchmarks that can be accessed via a GUI interface, or script. A sample benchmark was carried out on a Windows 10 PC to clearly understand how the CIS-CAT Lite works and validate the resulting output against the research goal and question. For this experiment, the target CIS benchmark is "CIS_Ubuntu_Linux_20.04_LTS_Benchmark_v3.0.0-xccdf.xml"

3.1 How CIS-CAT Lite Tool Works

As mentioned before, CIS-CAT Lite integration can be tested via GUI, or shell script. The necessary permission is needed to run CIS via shell script. The integration can be done against a local server, and against a remote server by providing the essential IP address and port of the target system. Identification of the use of the target system, and operating system is necessary to choose the correct benchmark for the parameters to perform a test. An example output includes a percentage value of the score earned, the maximum possible score that can be earned (234 for level 1 hardening), and a HTML file detailing every check whether pass or fail. At the time of writing this paper, except if a failed check is reviewed, and corrected, running a test gives an idempotent result.

Multicloud DR typically involves two or more cloud providers to make sense of this with the availability of several cloud providers, for this project the choice of the two cloud providers to be used were determined by how similar each both cloud providers are with the free version available. As there is no clear marker on this topic, the cloud providers Amazon Web Services (AWS)-US East, and Microsoft Azure-US Central free versions were the choice because of the following reasons;

1. The competitive edge each of these providers possesses in the cloud space.
2. The consensus of how these cloud providers are perceived in the real world.
3. Cloud provider complements or favors the level of automation required by this research project.
4. Ease of access to resources to aid the smooth learning curve of each cloud provider.

Both cloud providers (AWS, and Azure) support the above with similarities in operations, and tooling that avoid processing overhead, and ease of knowledge transfer between each cloud domain.

This research methodology uses application of software tool to perform practical disaster recovery use-cases available in real world infrastructure powering scalable applications with minimal downtime unlike ideal simulation tools that would generate results in an overly controlled scenario making this research to be appealing to more than just the educational research group.

To carry out this experimentation, the infrastructure components involved in building several cloud Saas application had to be identified, an example Saas application requires provisioning of dedicated machine or server with a specification capable of handling the running application processes, a persistent storage database for storing essential data generated by the application users, and monitoring. For continuous reliability, and minimal downtime of the application, each of these components needs to have a redundant fail-over instance to continue serving in the case of an infrastructure failure within a defined Recovery Time Objective (RTO), and Recovery Point Objective (RPO). This research methodology focuses on infrastructure recovery and security across the chosen cloud providers.

The automated approach to provisioning these identified cloud resources needed for the experiment was chosen over a manual resource provisioning using the console interface provided by these cloud providers following the latest cloud trends called Infrastructure as Code (IaC). Infrastructure as code involves scripting of cloud resource as in a versioned text format, that can roll-out an orchestrated plan to serve application hosting. Given that there are a lot of IaC options today, the choice was based around a more agnostic IaC tool, and domain programming language required, cloud provider supports. The studies by de Carvalho and Patricia Favacho de Araujo (2020), and Thiyagarajan (2020) showed Terraform as a top competitor, it was used for scripting and for these reasons.

3.2 Experiment Variants

The research method introduced the concept of "variants." Two sets of variants associated with different security levels were designed and deployed across both cloud providers. Table 1 below covers key characteristics for each variant.

Table 1: A table of detailed description, and characteristics of experiment variants

Variant	**Description**	**Characteristics**
Variant 1 (Minimal Security)	Basic Infrastructure	Default VPC/VNet, basic security groups/NSGs, unencrypted DB storage, no host hardening
Variant 2 (Standard Security)	Hardened infrastructure aligned with CIS Level 1	Custom VPC/VNet, subnet segregation, ACLs/NSGs, SSH hardening, UFW firewall, DB encryption, IAM restrictions

The infrastructures for the highlighted variants were tested on AWS and Azure clouds, resulting in four separate configurations, each executed twice for reliability and to ensure reproducibility of the experiment. The deployed system architecture for these variants is clearly shown in Figure 2.

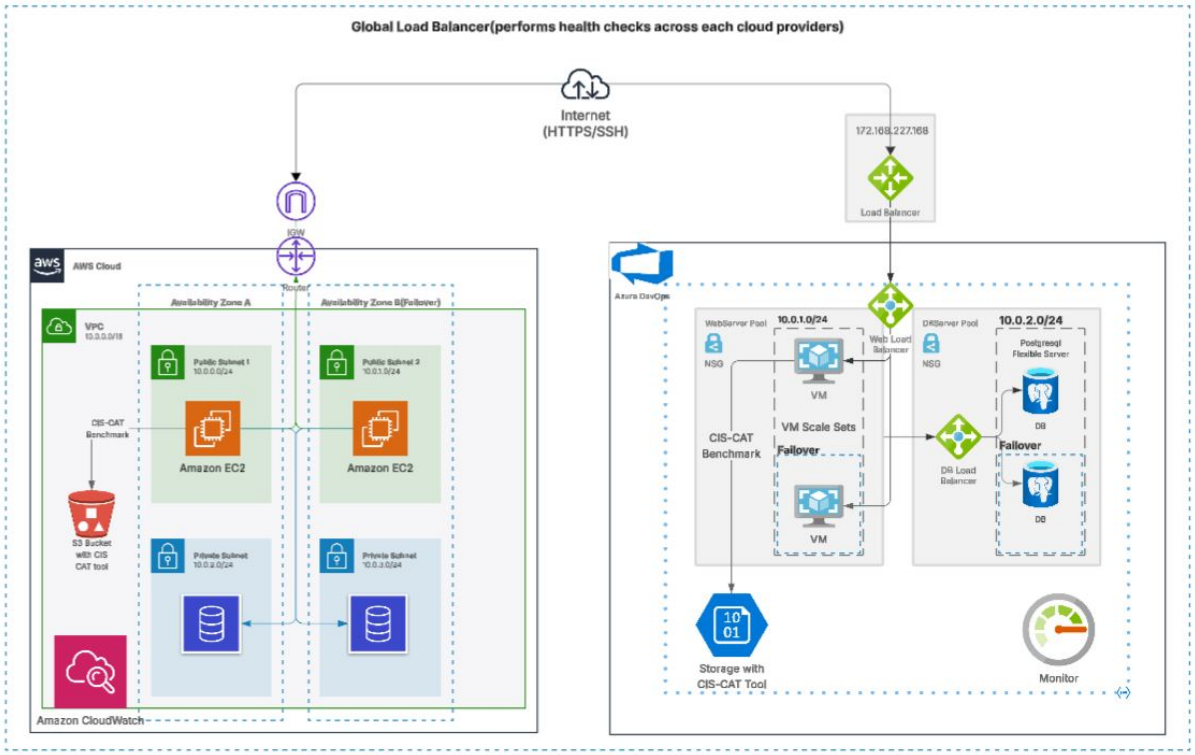


Figure 2: High level architecture of Active-Passive Disaster recovery

4 Design Specification

Disaster recovery techniques are standard, reliable approaches used by cloud technicians and site reliability engineers to ensure resilience of production systems using detailed plans. This research identified the following cloud disaster recovery approach documented by Que et al. (2024), which are as follows;

- **Active-Active**: Concurrent infrastructure setup across cloud providers.
- **Active-Passive**: Primary cloud infra setup with a standby replica.
- **Backup, and Restore**: Periodic or scheduled data backup to be restored post-disaster Harwalkar et al. (2019).

The architecture design of the disaster recovery strategy for this research experiment was configured against Active-Passive disaster recovery (DR).

4.1 Research Definition of Active-Passive DR

For this type of DR, an active primary cloud resource handles incoming requests from users typically in a production environment, accompanied by a standby resource that is setup, and initial outage in the active region. This DR setup serves production applications mostly because it is cost effective compared to maintaining an active-active DR strategy and still offers a reasonable recovery time objective. The DR architecture design and analysis are built around this strategy.

The research methodology in the previous section talked about a production Saas application, the Saas application in this context comprises of a virtual machine, a serving database hosted on the cloud, a monitoring system, and an entry point, these four significant components are the main resources that were provisioned on both cloud providers during these research, other support cloud tools were also provisioned to support these components. A unit of virtual server, database, and monitoring system were provisioned for these infrastructure setups.

The methodology introduced 2 unique variants; these variants were provisioned across both cloud providers AWS, and Azure cloud. Table 2 shows the detailed differences in infrastructure designs.

Table 2: Different Variant Specification of the resources deployed on AWS, and Azure.

Feature	**AWS Variant 1**	**AWS Variant 2**	**Azure Variant 1**	**Azure Variant 2**
Compute Type	EC2 t3.small	EC2 t3.small	Standard D2s v3	Standard D2s v3
OS	Ubuntu 20.04 LTS	Ubuntu 20.04 LTS	Ubuntu 20.04 LTS	Ubuntu 20.04 LTS
Deployment	Auto Scaling Group (2 AZs)	Auto Scaling Group (2 AZs)	VM Scale Set (2 AZs)	VM Scale Set (2 AZs)
Database Type	RDS PostgreSQL db.t3.micro	RDS PostgreSQL db.t3.micro	PostgreSQL Flexible Server (Basic)	PostgreSQL Flexible Server
Availability	Multi-AZ enabled	Multi-AZ enabled	Zone-Redundant HA	Zone-Redundant HA
DB Security	Storage: 20GB No encryption	Storage encrypted	Storage: 32GB (HA minimum)	Private endpoints All features enabled
VPC / VNet	Default	Custom	Basic (No default)	Custom
Subnet Layout	Default subnets (2 AZs)	Public / Private split (4 subnets)	Default Web and DB subnets	Web (2 AZs) Private DB subnets
Load Balancer	ALB	ALB	ALB	ALB
Net. Security	Basic SGs (HTTP, SSH 22)	Network ACLs Restrictive SGs (Least privilege)	Basic NSGs (HTTP, SSH 22)	Restrictive deny-by-default NSGs
Hardening Applied	None	CIS Level 1 Hardening (SSH, Kernel, UFW, Passwords, Auditd)	None	CIS Level 1 Hardening (SSH, Kernel, UFW, Passwords, Auditd)
Monitoring	None	VPC Flow Logs CloudWatch	None	Network Watcher Log Analytics Storage Firewall
Identity Management	–	–	Basic Service Principal	Managed Identity for storage access

The specifications of each variant differ by security hardening for both network, virtual machine, and database layers. The first variant features a basic but functional setup. It is operational to fulfil the requirements of several Saas applications. The second variant introduces more complexity by offering a more secure, and advanced infrastructure. The second variant illustrates the requirements for a sophisticated enterprise application; some of these requirements minimize security risk, obscure sensitive information, control access points, and IAM policies.

5 Implementation

The implementation phase covered the procedures which include manual, automated steps, and assumptions. Terraform v1.0+, an Infrastructure as Code (IaC), was employed to have a configuration of the infrastructure to be provisioned at command for each variant on both clouds. The terraform codes were inspired by the official terraform online web documentation. A suitable version of Azure, and AWS provider on terraform was identified for local scripting. Terraform was handy in performing CRUD operations on all resources for each variant for both cloud providers.

Key Implementation Decisions

5.1 Instance Setup Automation

From Figure 4, Terraform instrumented the key automation by encoding user-data script for VM instance configuration during development phase, this allowed automatic for upgrades, application installation of Nginx to perform periodic health checks, setup Postgres client for checks on the DB, onboarding and setup of CIS-CAT tool, and other support scripts upon VM initialization.

5.2 Resource Idempotency

Terraform configurations are designed for idempotent execution; this behavior allows easy development phase even with the continuous application operations without any side effects. This was key in performing multiple test iterations between reasonable cool off period with consistent infrastructure reproduction.

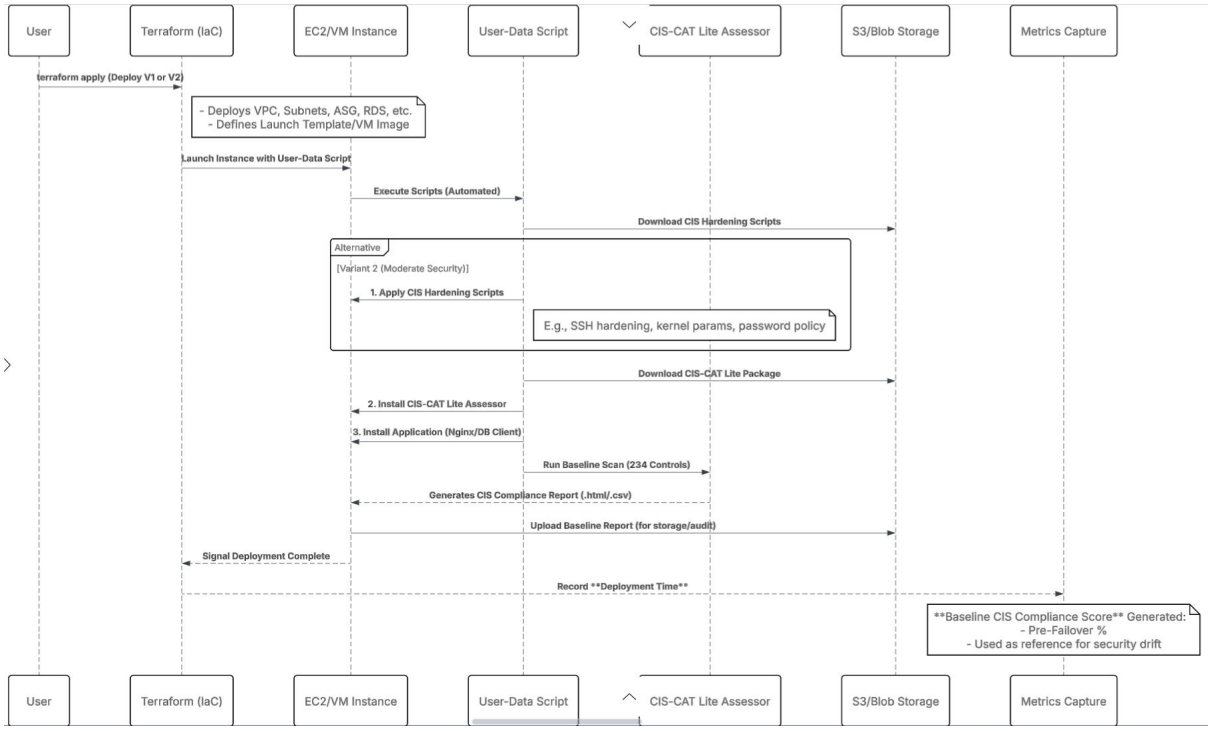


Figure 3: Approach of Automated flow of IaC deployments.

5.3 Failover Automation Scripting

A unified script was consistently tuned to automate the accompany fail-over of both AWS and Azure resources. This script was run locally on a PC authenticated to both cloud providers. Its usecase made sure to cover all variant configurations, collect verbose log information of the running process. When the script is run with the parameters "cloud_type", and "variant" e.g "aws" "1", it targets the AWS cloud, and first variant, same goes for Azure, and other variants.

The script workflow performed service discovery to the provisioned resources, through the clouds available in CLI commands, and exposed health endpoints. Through SSH, a pre fail over scan is done on the VM using a run-assessment script created during initialization, a failure is enforced through a stop command, a timestamp log is collected for this process, a 5 seconds interval check monitors the database, and VM health status, then calculates and logs RTO upon recovery detection. Post failover assessment results are also logged for analysis.

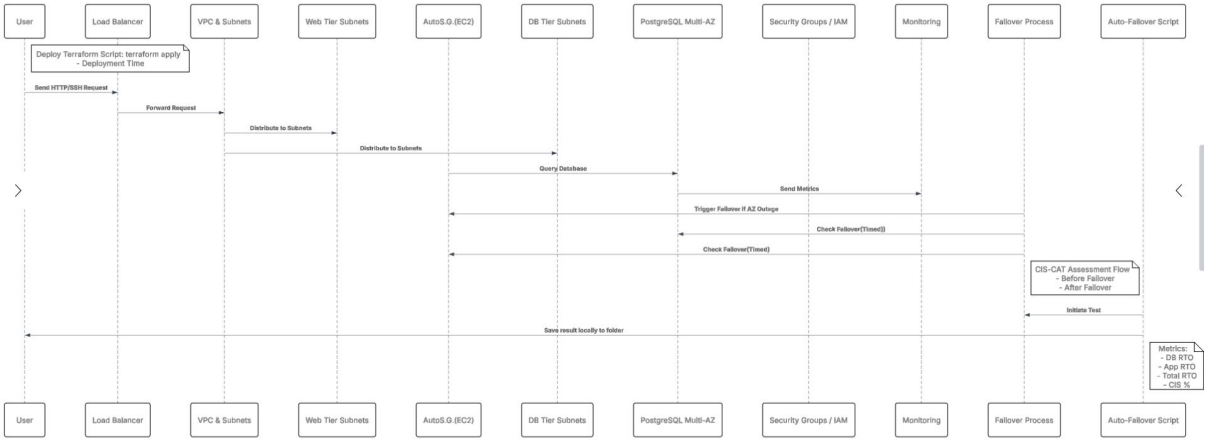


Figure 4: System Flow of Automated Failover after the Disaster

5.4 Result Aggregation

The result gathering includes manual collection, and collation of informative logs such as RTO, Compliance Percentage from CIS-CAT assessment HTML reports, into a file after a successful experiment of a variant on both clouds for analytical evaluations. A custom python script was used to generate well represented plots of the results comparing total RTO, CIS compliance preservation, etc.

This experimental test for each variant was conducted a second time to get an average on the initially generated results. In summary, terraform scripts were able to configure and deploy the system architecture across clouds, CIS-CAT Lite assessment, failover scripts, and post fail-over instances completed successfully with logs that capture RTO.

5.5 Challenges

The AWS VM instance upon initialization could not access the CIS-CAT tool hosted on S3 bucket because of the restriction on the AWS account used. The restriction did not allow creation of new roles, IAM policies for the VM instance to access the bucket. The walk around was to add a policy directly to the bucket to allow public reads. This issue was not the same for Azure VM.

Aggregating the collected data from the experiment was done manually, which could introduce the possibility of human error. This was mitigated by being careful in a free distraction environment.

6 Evaluation

The section analyses the explored architectural design, method approach, and findings in relation to the research topic. The research method aimed at unifying a framework, that correlates Recovery Time Objective (RTO), and security compliance for a standardized infrastructure disaster recovery across multiple clouds using Amazon Web Services (Azure), and Microsoft Azure.

The following subsection evaluates the experiment for the different variants conducted for this research;

6.1 Case Study (Variant 1)

This variant covered the basic deployment for both cloud providers, and the analysis of the minimal security configuration. Table 3 shows the values gathered for each variable for Variant 1 setup.

Table 3: A table showing Total, App VM, Database Recovery Time, CIS Score, and Security Drift across AWS, and Azure Variant 1

Config	**Run**	**Deploy (min)**	**DB RTO (s)**	**App RTO (s)**	**Total RTO (s)**	**Pre-CIS (%)**	**Post-CIS (%)**	**Drift**
AWS V1	1	14:39	80	71	151	60.68	60.68	0%
AWS V1	2	15:41	79	189	268	60.68	60.68	0%
Azure V1	1	12:48	8	314	322	61.11	61.11	0%
Azure V1	2	14:25	9	295	304	61.11	61.11	0%

6.1.1 Amazon Web Services Case Study with Minimal Security

The terraform infrastructure plan for the first variant (Variant 1) was executed to provision the resources on the AWS cloud using the terraform application command. A total of 13 resources were provisioned by terraform for the first variant setup. This execution was timed, and took a total of 14 minutes, 39 seconds. The result of the fail-over automation script also confirmed 13 resources orchestrated to serve the application of AWS VM instance, and the database through health check probing. The pre, and post fail-over over scan for the VM using CIS-CAT tool generated a consistency output of 60.68% (142 out of 234 scored controls passed) for Variant 1 with basic security configuration that introduced the term zero security drift, the recovery time of the application VM or server was 71 seconds, and the database was 80 seconds, this resulted in a total recovery time of 151 seconds (about 5 minutes). A second experiment result was conducted to ensure experiment reproducibility and validity; the application VM had higher RTO, and higher result margins on the experiments compared to the database RTO.

6.1.2 Azure Case Study with Minimal Security

The infrastructure plan for Azure Variant 1 using terraform was different in the total number of resources provisioned. 22 resources were added by terraform which is significantly more than AWS variant 1, Azure cloud does not come with default VPC, which had to be created during this process. Despite provisioning 9 additional resources (69% more), the first and second deployment completed 1 minute, 51 seconds, and 14 seconds faster than AWS (with lesser resources), highlighting Azure's efficient resource provisioning capabilities This finding challenges the assumption that more resources necessarily correlate with longer deployment times in multi cloud environments. The total recovery time objective (RTO) of 314 seconds (about 10 minutes) was significantly higher than AWS for this variant, with the application VM recovery time leading to the reason for this. Both experiments for Azure showed high consistency in total RTO. Compared to AWS, the percentage of CIS-CAT assessment was 61.11%, which was a little higher than the AWS for this variant.

6.2 Case Study(Variant 2)

This variant covered the aspect of application of CIS Controls for extra security hardening of the overall design architecture, and the resources provisioned from the load balancer, network, application VM or server layer, and database layer. The CIS-CAT assessment outputs can be equally considered, as an "Audit." This audit is automatically timestamped, and contains security analysis of the target system; every failed control supported by a potential fix. Several security compliance controls that failed from analyzing the CIS-CAT audit or result were used in preparing this infrastructure design variant. This CIS compliance percentage gain shown in Table 4 was achieved by implementing approximately 50% of CIS Level 1 controls.

Table 4: A table showing Total, App VM, Database Recovery Time, CIS Score, and Security Drift across AWS, and Azure Variant 2

Config	**Run**	**Deploy (min)**	**DB RTO (s)**	**App RTO (s)**	**Total RTO (s)**	**Pre-CIS (%)**	**Post-CIS (%)**	**Drift**
AWS V2	2	17:32	78	125	203	72.65	72.65	0%
AWS V2	1	16:32	85	217	302	72.65	72.65	0%
Azure V2	1	15:12	7	361	368	72.65	72.65	0%
Azure V2	2	12:54	12	355	367	72.65	72.65	0%

6.2.1 Amazon Web Services Case Study with Hardened Security

The architectural design helped setting up this infrastructure plan on AWS cloud, Unlike the first variant 25 resources were deployed with respect to the security advancements with resulted in a higher deployment time of 16 minutes, 32 seconds, and 17 minutes, 32 seconds for both trials of these experiments. The total recovery time objective (RTO) was around 203 to 302 seconds (about 10 minutes) compared to variant 1. The application VM recovery time objective (RTO) increased by approximately 32%, and DB recovery time objective (RTO) remained relatively stable compared to the first variant in 6.1.1.
A major improvement in CIS score was recorded as pre scan, and post scan CIS compliance results increased more than 10% over Variant 1 due to the effective hardening, this result highlights that RTO metric is dependent of the level of security measures to be achieved due to more added complexity.

6.2.2 Azure Case Study with Hardened Security

The extra hardening of the security profile of Azure Variant 2 introduced a more restrictive inbound rules, and additional policies for each orchestrating layers of the infrastructure, but these enhancements resulted in a sophisticated DR plan compared to Variant 1.Terraform provisioned the hardened environment in 15 minutes 12 seconds for the first run and 12 minutes 54 seconds for the second run, interestingly faster than the hardened AWS deployments despite Azure managing more policy-heavy resources, and Azure terraform API design. Azure showed a faster database recovery time extremely low, at 7 seconds and 12 seconds. This metric was the same for Azure Variant 1, demonstrating the consistency of Azure's rapid failover at the data layer. However, like Azure Variant

1, the application VM still remained in the bottleneck, recording an RTO of 361 seconds and 355 seconds across both runs, resulting in total RTO values of 368 seconds and 367 seconds. Figure 5 shows the balance in recovery speed, and good security across clouds using a scatter plot.

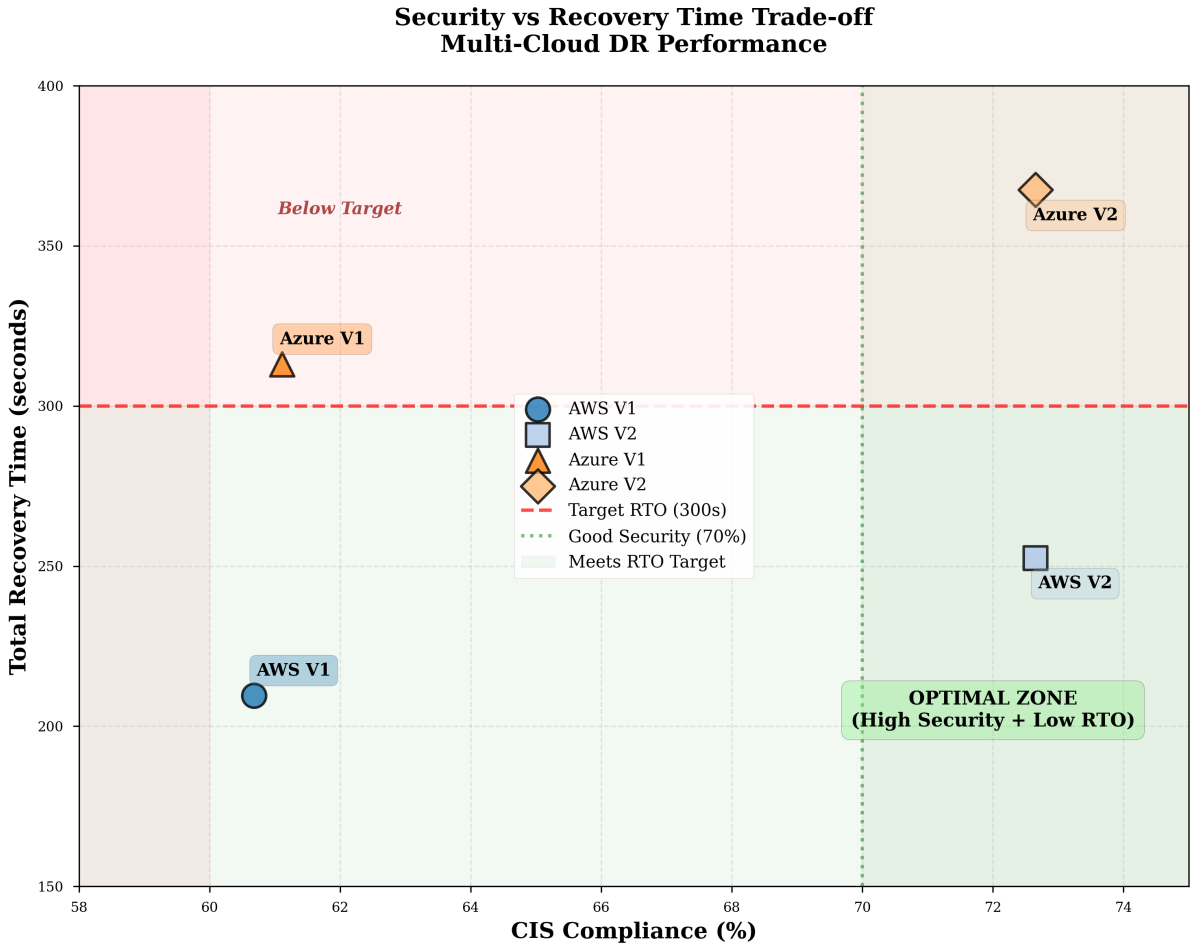


Figure 5: Multicloud Security v Recovery Time Comparison

The plot in Figure 5 highlights a use case where 75% CIS-CAT compliance score, and a Recovery Time Objective (RTO) of 300 seconds are target goals for the difference. After security hardening in AWS Variant 2, AWS DR completes both requirements successfully, and Azure Variant 2 reached the CIS security score but at a higher recovery time. Generally, CIS compliance scores were aligned with the AWS hardened variant, achieving 72.65% for both pre- and post-failover scans, again reflecting 0% drift and confirming that Azure's hardened design-maintained configuration integrity through failover. This paragraph covers information that would be considered insightful for cloud users to make certain tradeoffs to achieve business targets.

The numerical values from all the experiments conducted for the different variant is aggregated into Table 5 below;

Table 5: An aggregate of the RTO's, Security Compliance result from the experiments

Metric	**AWS V1**	**AWS V2**	**Azure V1**	**Azure V2**
Mean Total RTO	209.5 s	252.5 s	313.0 s	367.5 s
Std Dev of Total RTO	82.7 s	70.0 s	12.7 s	0.7 s
Mean Database RTO	79.5 s	81.5 s	8.5 s	9.5 s
Mean App VM RTO	130.0 s	171.0 s	304.5 s	358.0 s
Mean Deploy Time	15:10	17:02	13:36	14:03
CV% of Total RTO	39.5%	27.7%	4.1%	0.2%
Mean CIS Compliance	60.68%	72.65%	61.11%	72.65%

Table 5 above categories the each conducted tests, to identify the level of variability for the different cloud providers and variants that were used in plotting Figure 6.

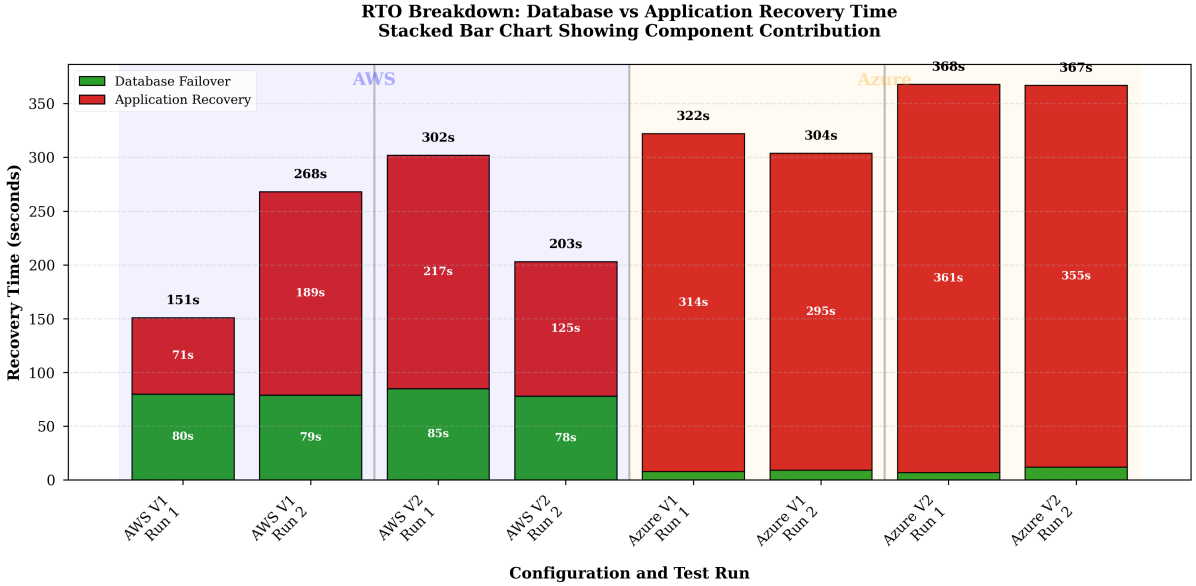


Figure 6: Stacked Bar-Chart Analysis of RTO Breakdown for the different AWS, and Azure (Database vs Application)

The application VM recovery time between Azure, and AWS for all conducted experiment showed AWS superiority, the closest between both clouds are AWS V2 Run 1, and Azure V1 Run 2, where several factors could be a result of the provider underlying hardware(setup, OS initiation, application initiation, process restoration, and IP switching time) according to Alhazmi and Malaiya (2013). Each of these hidden factors not disclosed by the cloud providers are potential credits to the varying results across AWS, and Azure.

The low database recovery time exhibited by Azure compared to AWS can be attributed to the design of Azure PostgreSQL Flexible Server's Zone-Redundant that was used in this research experiment. Both cloud providers maintained higher application recovery time around 250-300 seconds in range as shown in Figure 7. The characteristics of the recovery time results gotten from Azure include low variance, deviation, and high predictability. These are qualities that drive DR design decisions.

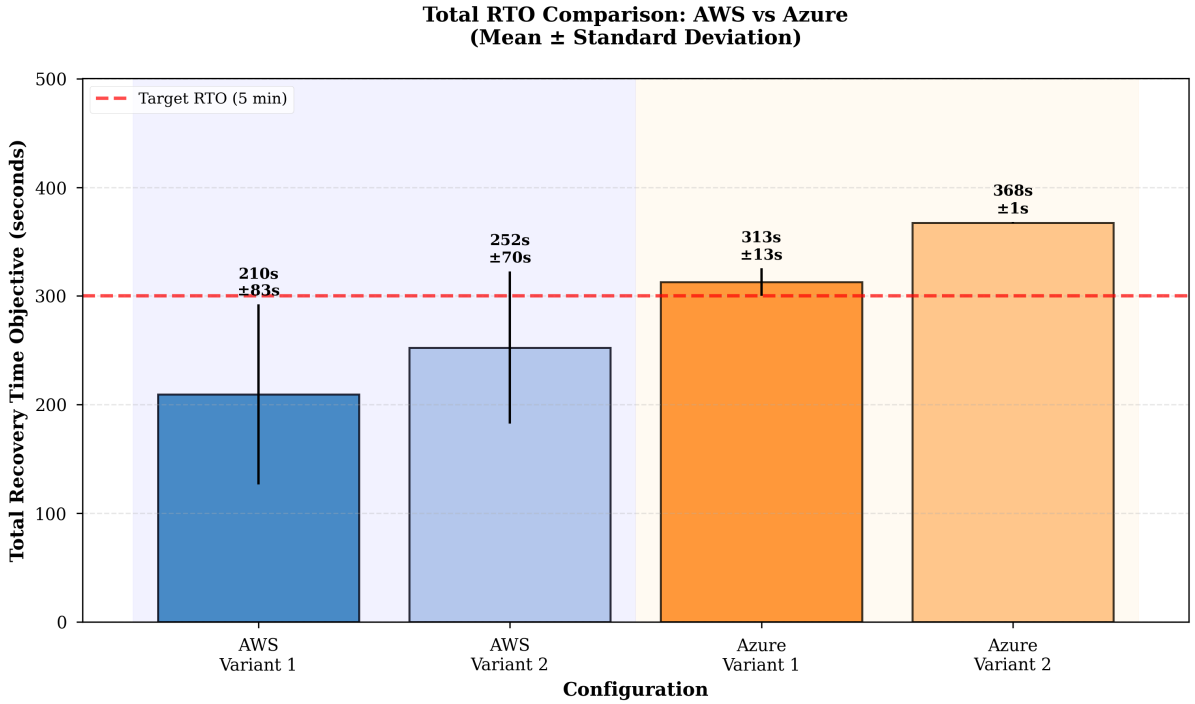


Figure 7: Mean or Average RTO for AWS, and Azure

The idea of zero security drift using CIS-CAT Lite tool became a key concept, and finding, which means for this research that the security compliance posture remained the same before the injected disaster (pre fail over).

As shown in Figure 8, after disaster recovery (post failover), and was consistent across AWS, and Azure cloud providers that indicate zero security degradation across multicloud failovers for the 8 experiments tests conducted. After implementing Level 1 hardening of CIS controls, both cloud providers passed 170/234(72.65%) controls each for both disaster recovery. This finding clears up common assumption that disaster recovery comprises security for speed, as the CIS assessments showed the use of Infrastructure as Code (IaC), and hardened VM or AMI images preserved security configuration over automated Active-Passive disaster recovery scenarios.

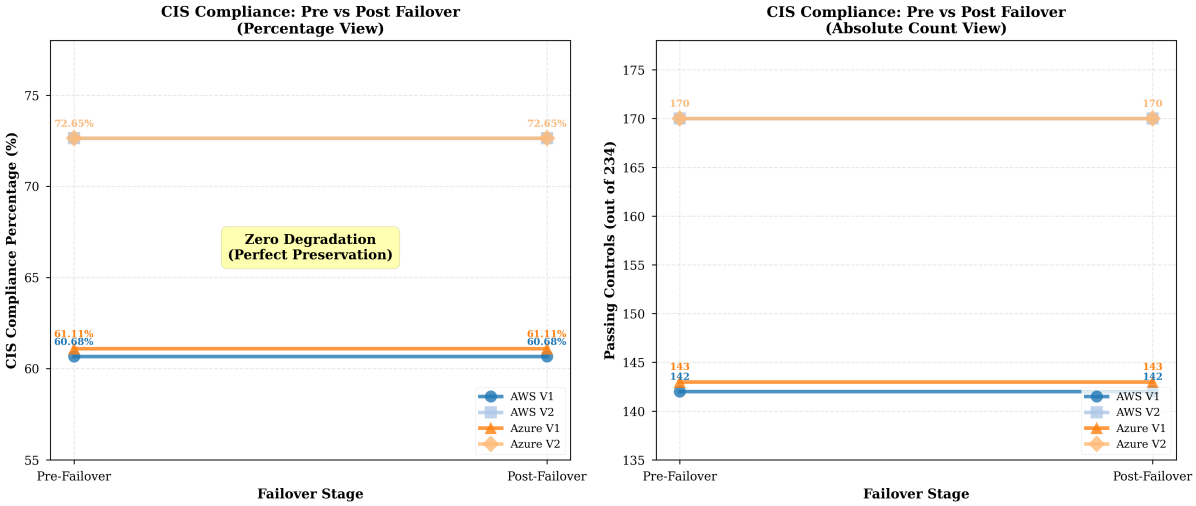


Figure 8: A line analysis proving the concept of zero-drift using CIS Benchmark.

6.3 Discussion

The CIS tool was successfully integrated into these multi cloud DR plans, by deploying the CIS-CAT Assessor tool using the infrastructure as code tool "Terraform." The CIS-CAT Assessor was automated to measure 234 security controls by performing pre and post failover security assessments for different variants (Variant 1, and Variant 2) on AWS, and Microsoft Azure. This research plan demonstrated zero security drift across multicloud disaster recoveries, analyzed security and performance (RTO) tradeoffs (19% compliance gain for 17-21% RTO increase) of almost the same proportion, this key quantitative analysis as shown in Figure 6.5 is an empirical contribution by this research and also helps answers the primary research question.

An assumption such as RTO increases with respect to complex security configuration was validated across AWS, and Microsoft Azure multicloud. For Variant 2, AWS showed +20.5% in security, +45 seconds RTO increase, and Azure showed +17.4% in security, +54 seconds RTO increase from Variant 1. AWS had 120% faster application VM disaster recovery time while Azure had 89% faster database recovery time. Generally, AWS progressed 47% overall RTO over Azure, while Azure had consistent values with lower coefficient of variation (CV) less than 5% for the experiments performed. Although this research utilizes a practical framework for its assessments, the minimal instance types (t3.micro, B1s) were leveraged to minimize cost where larger instances may show better performance metrics. CIS-CAT Lite free licensed tool has limited benchmarks to support assessment of the database layer.

6.3.1 Implications

In an Active-Active DR multi-cloud DR strategy where multiple infrastructure instances are operational simultaneously because the load balancer uses site health checks to efficiently route user request there is net zero infrastructure deployment time during a failover. During the recovery time, in the case of two instances, the previously shared workload is now temporarily handled by only one instance. For an available system, the only major drawbacks here are the increased workload on the instance server, and the

increased latency experienced by users. Among all these advantages for a business and its users, the potential cost of operation, and maintenance for N primary instance would be N times greater than the cost of an Active Passive DR plan where recovery time objective (RTO) is a variable, directly impacting business decisions on which approach to use.

The framework implemented in this research demands high technical expertise to undertake end to end of the architecture. Complex systems can sometimes be quite challenging, and the level of success is dependent on the platform, site reliability engineer, or technician Sunarjo et al. (2025). In an organization setting, an infrastructure team is mobilized to maintain, and observe the health metrics (latency, throughput, SLAs, and SLO's) of the overall system to benefit from every added advantage.

7 Conclusion and Future Work

The competition demonstrated by cloud provider in recent times by producing several cloud services either for storage or processing capabilities targeted to drive adoption and significantly attract more cloud services users or consumers. This race, if not carefully reasoned would be one of the rising assumptions of cloud vendor lock in. This section sums how every careful decision involved in this research to determine a framework that validates security configurations through series of eight multicloud disaster recovery failover tests. This research successfully addresses the research question "How can the CIS Benchmark be integrated into a framework to evaluate the performance and security of DR strategy (Active-Passive) in multi-cloud environments?". The iterative steps that involved the use Terraform IaC tool to implement a multicloud disaster recovery setup across Amazon Web Services (AWS), and Microsoft Azure for different security hardening, that leveraged automated failovers integrated with CIS-CAT benchmarks. With the analyzed results from this research such as zero drift security, the deterministic outcome of what security configuration on one cloud provider translates to another, emphasizes that the possibility of multicloud interoperability is a result of this competitiveness. This goal shared by cloud provider giants is rather positive that will drive multicloud as a utility.

The findings of this research are relevant to real cloud service users and applications that require high availability and strong security controls, such as enterprise systems and regulated workloads. Organizations can also have great confidence in integrating CIS Benchmarks into automated disaster recovery testing for any configuration drift, and to help verify that failover processes preserve security configurations across cloud providers.

A future plan regarding this research that includes Google cloud is inevitable, to see how its compliments with already acquired results from this research, CIS-CAT Pro tool, a more sophisticated version, that was applied in this research experiment would be handy to cut across different instances, database engine types, and regions available for this different cloud providers. Also, one cannot mention cloud providers, without costs, therefore there is a better framework involving these extra variables for future work.

References

Abualkishik, A. Z., Alwan, A. A. and Gulzar, Y. (2020). Disaster recovery in cloud computing systems: An overview, *International Journal of Advanced Computer Science and Applications* **11**(9).
URL: *http://dx.doi.org/10.14569/IJACSA.2020.0110984*

Akbari, S. and Hauswirth, M. (2024). Sampling in cloud benchmarking: A critical review and methodological guidelines, *2024 IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*, pp. 160–167.

Alhazmi, O. H. and Malaiya, Y. K. (2013). Evaluating disaster recovery plans using the cloud, *2013 Proceedings Annual Reliability and Maintainability Symposium (RAMS)*, pp. 1–6.

Badirova, A., Dabbaghi, S., Moghaddam, F. F., Wieder, P. and Yahyapour, R. (2023). A survey on identity and access management for cross-domain dynamic users: Issues, solutions, and challenges, *IEEE Access* **11**: 61660–61679.

Beyer, B., Jones, C., Petoff, J. and Murphy, N. R. (2016). *Site Reliability Engineering: How Google Runs Production Systems*.
URL: *http://landing.google.com/sre/book.html*

Beyer, B., Murphy, N., Rensin, D., Thorne, S. and Kawahara, K. (2018). *The Site Reliability Workbook*.
URL: *https://landing.google.com/sre/book.html*

Bucur, V. and Miclea, L. (2021). Optimizing towards a multi-cloud environment through benchmarking data transfer speeds in amazon web services and google cloud, *2021 IEEE 17th International Conference on Intelligent Computer Communication and Processing (ICCP)*, pp. 69–76.

Caceres, A. and Globa, L. (2022). State-of-the-art architectures for interoperability of heterogeneous clouds, *2022 IEEE 16th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering (TCSET)*, pp. 704–709.

de Carvalho, L. R. and Patricia Favacho de Araujo, A. (2020). Performance comparison of terraform and cloudify as multicloud orchestrators, *2020 20th IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing (CCGRID)*, pp. 380–389.

Fernandes, Diogo A. B., S. L. F. B. G. J. V. F. M. M. I. P. R. M. (2014). Security issues in cloud environments: A survey, *International Journal of Information Security* .

Fernando, M. S. (2017). It disaster recovery system to ensure the business continuity of an organization, *2017 National Information Technology Conference (NITC)*, pp. 46–48.

Harwalkar, S., Sitaram, D. and Jadon, S. (2019). Multi-cloud draas using openstack keystone federation, *2019 International Conference on Advances in Computing and Communication Engineering (ICACCE)*, pp. 1–6.

Kaushik, P., Rao, A. M., Singh, D. P., Vashisht, S. and Gupta, S. (2021). Cloud computing and comparison based on service and performance between amazon aws, microsoft azure, and google cloud, *2021 International Conference on Technological Advancements and Innovations (ICTAI)*, pp. 268–273.

Kotha, R. (2023). Multi-cloud strategies for enhanced resilience and flexibility, *Journal of Artificial Intelligence Cloud Computing* **2**: 1–5.

Liu, B., Xin, Y. and Dai, S. (2022). Blockchain-based disaster recovery data storage and security auditing solution in multi-cloud environment, *2022 International Applied Computational Electromagnetics Society Symposium (ACES-China)*, pp. 1–4.

Mazumder, K. A. and AbdAllah, E. G. (2023). Information-centric networking (icn) based disaster recovery and business continuity (drbc) of bangladesh, *2023 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech)*, pp. 0014–0019.

Pandurangan, S. K. (2025). Advanced disaster recovery system in cloud infrastructure, *Journal of Computer Science and Technology Studies* **7**(8): 433–443.
URL: *https://al-kindipublishers.org/index.php/jcsts/article/view/10547*

Pellegrini, R., Rottmann, P. and Strieder, G. (2017). Preventing vendor lock-ins via an interoperable multi-cloud deployment approach, *2017 12th International Conference for Internet Technology and Secured Transactions (ICITST)*, pp. 382–387.

Puri, G. S., Tiwary, R. and Shukla, S. (2019). A review on cloud computing, *2019 9th International Conference on Cloud Computing, Data Science Engineering (Confluence)*, pp. 63–68.

Que, H.-K., Huang, S., Jiang, X., Zeng, L.-K., Yang, J.-X. and Wu, M.-Q. (2024). Strategy of smart grid disaster recovery and scheduling based on cloud computing, *2024 The 9th International Conference on Power and Renewable Energy (ICPRE)*, pp. 672–677.

Saraswat, M. and Tripathi, R. (2020). Cloud computing: Comparison and analysis of cloud service providers-aws, microsoft and google, *2020 9th International Conference System Modeling and Advancement in Research Trends (SMART)*, pp. 281–285.

Sengupta, S. and Annervaz, K. (2014). Multi-site data distribution for disaster recovery—a planning framework, *Future Generation Computer Systems* **41**: 53–64.
URL: *https://www.sciencedirect.com/science/article/pii/S0167739X1400140X*

Sengupta, S. and Annervaz, K. M. (2012). Planning for optimal multi-site data distribution for disaster recovery, *in* K. Vanmechelen, J. Altmann and O. F. Rana (eds), *Economics of Grids, Clouds, Systems, and Services*, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 161–172.

Sunarjo, R. A., Emmanuel Widjaja, A., Magdalena, L., Azzudin, M., Atmadja Effendi, W. W., Zul Friandi, S. and Ikhsan, R. Z. (2025). Addressing cybersecurity risks in multi cloud environments for digital transformation, *2025 4th International Conference on Creative Communication and Innovative Technology (ICCIT)*, pp. 1–6.

Syed, A. A. M. and Ali, S. (2021). Evolution of backup and disaster recovery solutions in cloud computing: Trends, challenges, and future directions, *JOURNAL OF RECENT TRENDS IN COMPUTER SCIENCE AND ENGINEERING (JRTCSE)* **9**(2): 56–71. **URL:** *https://jrtcse.com/index.php/home/article/view/JRTCSE.2021.2.6*

Thiyagarajan, R. B. (2020). Single and multi-cloud disaster recovery management using terraform and ansible, *Norma NCI* .