Efficient security management mechanism for multi-cloud systems

JungHwa Ryu

Department of Future Convergence
Technology Engineering
The Graduate School of
Sungshin Women's University

Efficient security management mechanism for multi-cloud systems

A Master's Thesis
Submitted to the
Graduate School of Sungshin Women's University

in partial fulfillment of the requirements
for the degree of
Master of Future Convergence Technology Engineering

JungHwa Ryu

November, 2024

This is to certify that we have examined the
Master's Thesis of
JungHwa Ryu
Submitted to Department of Future Convergence
Technology Engineering

Approved as to style and content:

Thesis Advisor Il-gu Lee (인)

Committee Chairman Seongmin Kim (인)

Committee Member Il-gu Lee (인)

Committee Member Yeon-sup Lim (인)

The Graduate School of Sungshin Women's University

ABSTRACT

Multi-cloud environments offer various technological and economic benefits over conventional single-cloud setups by leveraging resources from multiple Cloud Service Providers (CSPs). However, these environments often overlook the associated security issues. In particular, integrating services from various CSPs presents challenges in maintaining consistent security policies, making multi-cloud setups more challenging for detecting security threats related to identity and access management (IAM), particularly involving lateral movement and threat propagation. This study presents an analysis of potential IAM security threats in multi-cloud environments and proposes a new security response methodology, iSIEM (Interoperable Security Information and Event Management Framework), for enhancing interoperability and compatibility among security monitoring cloud services. To achieve this goal, I developed three penetration-testing scenarios based on CloudGoat to evlauate real-world IAM security threats in multi-cloud settings. The iSIEM methodology supports workload mobility across clouds and addresses discrepancies in security logs across heterogeneous cloud platforms owing to mismatch determinations arising from security logs produced by each cloud's security tools. Evaluation shows that iSIEM enhances the efficiency of security management in multi-cloud environments, thus improving accuracy by a minimum of 10.71% and maximum of 23.08% compared to legacy security tools reliant on a single CSP.

Contents

Table Contents

Figure Contents

Ⅰ. Introduction

Recently, cloud users have adopted a multi-cloud strategies, utilizing multiple public cloud services in parallel to meet various service-level objectives instead of relying on a single cloud service provider (CSP). Multi-cloud enables users to flexibly move resources across CSPs and select optimal services that meet their needs. Furthermore, multi-cloud addresses vendor lock-in by minimizing service disruptions through resource migration to other clouds during outages, thereby improving reliability and fault tolerance [1]. In particular, multi-cloud environments offer higher availability than single-cloud setups by minimizing data loss through fault domain separation and cross-region backups. Moreover, major CSPs such as Amazon Web Services (AWS), Microsoft Azure, and Google Cloud Platform (GCP), are actively adopting multi-cloud strategies [2]. Cloud computing, in particular, is evolving to provide advantages in terms of performance, security, and cost efficiency through infrastructure and services offered by various CSPs. For example, sky computing, a state-of-the-art form of multi-cloud computing, enhances conventional cloud computing by offering new benefits such as load balancing, failover, cost efficiency, and multi-tenancy [3]. This model enhances load distribution through automated resource management and dynamic scaling, thereby allocating resources across multiple data centers or regions. It also optimizes costs through real-time monitoring and resource optimization, thus ensuring

tenant isolation even in multi-tenant environments. Nevertheless, integrating monitoring and auditing services from various CSPs, particularly in multi-cloud environments, poses significant challenges in maintaining consistent security policies. This integration not only broadens the attack surface compared to single-cloud environments but also elevates the risk of security threats. Existing cloud security monitoring systems often provide simple alerts without actionable countermeasures when an attack is detected, thereby failing to respond effectively to various transition attacks in multi-cloud environments [4]. Each cloud vendor operates independent infrastructures, platforms, and security policies, resulting in varied responses to the same security threats. Moreover, the use of services from multiple cloud vendors can lead to the propagation of specific vulnerabilities from one vendor to another or create new vulnerabilities during vendor integration. Such complexity in multi-vendor environments complicates the determination of responsibility for security threats, particularly those related to Identity and Access Management (IAM). While improper IAM configurations in single-cloud environments can be attributed to the user, security threats resulting from vulnerability propagation or service heterogeneity in multi-vendor environments cannot be solely attributed to the user. The passive response of CSPs overlooking the specificity of multi-vendor environments, coupled with differing standards across vendors, forces users to manually analyze extensive and diverse logs, potentially leading to security threats across the entire multi-cloud ecosystem. However, research on consolidating security solutions from

multiple CSPs remains relatively underexplored despite the rapid advancement of multi-cloud technology. Most existing research either proposes models targeting a single CSP or analyzes methodologies at an abstract level without considering multi-cloud environments [5]. Additionally, no research has specifically focused on evaluating threat scenarios in real public cloud environments.

This study presents an analysis of IAM security threats in actual multi-cloud environments, focusing on differences in security-monitoring services among vendors. This study explores three threat scenarios in multi-cloud environments where security issues in one CSP's infrastructure may propagate to another. To audit and monitor such risks, I proposed a novel security threat-response methodology called the interoperable security information and event management framework (iSIEM), which ensures the cooperative operation of various security tools. Evaluation results show that the proposed methodology achieves an improvement in accuracy ranging from 10.71% to 23.08% compared to conventional monitoring tools provided by public cloud services.

1) This study analyzes IAM vulnerabilities in multi-cloud environments and identifies potential threat scenarios where security issues in one cloud platform may propagate to another using CloudGoat-based Penetration Testing.

2) This study proposes the iSIEM framework, a novel security threat-response methodology designed for commercial multi-cloud environments.

3) This study compares and evaluates the iSIEM framework against

conventional single-cloud-based security monitoring tools to identify discrepancies in security log statuses caused by policy differences and demonstrates an improvement in accuracy ranging from 10.71% to 23.08%.

The remainder of this paper is organized as follows. Section Ⅱ reviews the background and existing literature on security monitoring strategies and deployment models of multi-cloud environments. Section Ⅲ highlights the motivation and key research questions. Section Ⅳ derives custom IAM-related penetration testing scenarios for multi-cloud to establish the ground truth, while Section Ⅴ details the design of the iSIEM framework. Section Ⅵ evaluates the efficiency of the proposed framework in mitigating IAM-related threats. Section Ⅶ discusses the implications of this study. Finally, Section Ⅷ concludes the study.

Ⅱ. Background and Related Work

1. Multi-Cloud Deployment Model

A multi-cloud environment refers to the strategic use of various CSP services to select the optimal cloud environment that best aligns with workload characteristics [6]. According to the 2024 Flexera report, 89% of enterprises have adopted a multi-cloud strategy [7]. Three major types of multi-cloud deployment models are identified [8], as illustrated in Figure 1. First, the independent cloud deployment model is the most commonly used, where the same instance is deployed and operated independently across different clouds, accounting for 57% of deployments. Security management in this model relies on basic security settings and individual CSP monitoring. Second, the cloud-to-cloud workload mobility model, which enables the migration of application loads across various cloud platforms, accounts for 40% of deployments. This model enhances flexibility by enabling mobility across different cloud platforms; however, maintaining consistent security policies and threat detection systems remains challenging due to technical differences between CSPs. Finally, the functional segregation model, which deploys a single application across heterogeneous virtual private clouds (VPCs), accounts for 35% of deployments. This model supports high levels of performance optimization; however, additional security issues may emerge due to

incompatibility between security functions of different CSPs. In particular, VPCs are built on a cloud provider's infrastructure; therefore, users must independently review security settings and protocols to prevent malicious access.

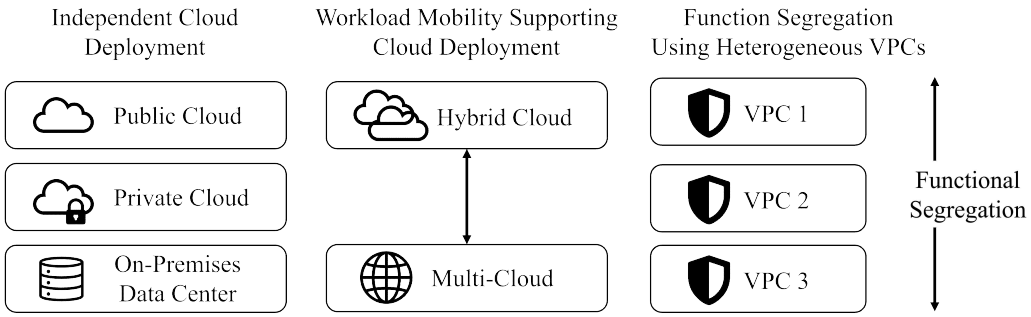

Figure 1. Multi-cloud deployment types

Establishing a concrete security auditing process for multi-cloud environments with functional segregation is particularly challenging and underexplored. When the same instance is deployed independently by each vendor, following the recommendations of security monitoring tools provided by each vendor is often sufficient. In contrast, security operators must address the risks associated with cross-cloud workload mobility and feature partitioning across VPCs in the segregated model. However, inconsistencies in security functions across CSPs necessitate complex security management, which conventional security approaches struggle to address. Given the current trend of leveraging distributed computing resources to efficiently build and run applications on cloud infrastructure models (e.g., sky computing [3]), exploring a unified security framework is both timely and essential. Notably, iSIEM addresses potential IAM-related threat propagation in cloud-to-cloud

workload mobility and functional segregation models.

Reece et al. [9] conducted a risk and vulnerability analysis of multi-cloud application deployments. They emphasized the need for research on security and vulnerabilities specific to multi-cloud environments, highlighting challenges in configuration, management, and integration arising from inconsistencies in security functions across CSPs. They applied the STRIDE and DREAD threat modeling methodologies to analyze six attack vectors, including cloud architecture, APIs, authentication processes, automation processes, and legal regulations, and provided criteria for systematically identifying and prioritizing threats in multi-cloud environments. This risk assessment approach has laid the foundation for quantitatively evaluating inherent risk factors in multi-cloud environments and formulating corresponding response strategies. However, their work remained at an abstract level of vulnerability assessment, lacking log management for detecting threat transitions and risk assessment methods, and did not include verification of the proposed methodology in commercial public clouds.

2. IAM Policy

IAM is a framework that defines security and management functions for verifying user identities and granting appropriate access rights to resources within the cloud environment. This framework enhances security by verifying user identities and restricting access to necessary resources. According to the Cloud Security Alliance report [10], insider threats were identified as the primary driver of cloud security incidents in 2019. However, since 2022, mismanagement, incorrect access rights settings, and access control failures have become more impactful. In particular, effective management of access rights for multiple users and resources is crucial in multi-cloud environments where resources from various CSPs are integrated. Moreover, incorrect IAM settings can allow attackers to gain system privileges and access critical data. Therefore, improper authentication and authorization due to over-privileged IAM roles have been identified as one of the TOP 10 cloud threats by the Open Worldwide Application Security Project [11].

To address this issue, research has been conducted to detect misconfigurations and errors in IAM policies for cloud environments. Van Ede et al. [12] developed a methodology for effectively detecting configuration errors in IAM policies within a cloud environment. This methodology specifically addresses security vulnerabilities in AWS. Potential policy errors can be detected by modeling authorization relationships as graphs and using node2vec embeddings to represent policy contexts in a specialized form. This method has been shown to

identify at least 3.7 times more errors than existing methodologies when evaluating IAM policy data in real corporate environments. Furthermore, Shevrin et al. [13] proposed an approach using a Boolean model to identify AWS IAM configuration errors and multi-stage attacks. This model verifies potential errors, including privilege escalation attacks, through formal verification and demonstrates the ability to detect complex multi-stage attacks within one minute in an AWS environment. However, the proposed methodology focuses only on analyzing IAM policies presented by AWS, without evaluating security issues related to IAM policies in multi-cloud environments, such as their actual operation in AWS.

3. State-of-the-Art Commoditized Cloud Security Solutions

As most enterprises adopt integrated cloud security management systems, major public cloud providers such as Amazon AWS and Microsoft Azure offer comprehensive security management solutions. Notable commoditized cloud security solutions include AWS Security Hub [14] and Azure Defender [15]. AWS Security Hub centralizes and manages security data from various AWS services. It enables users to consistently view security alerts and statuses and verify compliance with security standards across the AWS environment. Azure Defender is an integrated security management tool designed to enhance the security of the Azure cloud. Similarly, it provides cross-layer security protection for Azure resources, including virtual machines, databases, and containers.

However, in a multi-cloud setup, the security monitoring tools provided by each vendor lack plugin extensions or security-related settings for heterogeneous VPCs. Furthermore, they fail to immediately detect when connections with Windows credential providers linking AWS and Azure environments are severed. This study highlights these limitations of existing cloud security solutions through IAM security threat scenarios (see Section 3.2).

Ⅲ. Motivation

1. Key Research Questions and Basic Assumptions

This section defines the vulnerabilities arising from the intrinsic characteristics of multi-cloud environments, assuming heterogeneous VPCs. A review of existing scenarios indicated insufficient tools for conducting penetration testing targeting multi-cloud environments, nor were there prior studies on threats transferable in such environments. Although research exists on threat-modeling methodologies for multi-cloud environments [9], no multi-cloud monitoring tools provide real-time alerts based on system event logs. At the state-of-the-art level, individual vendor-specific security tools represent the best efforts in this regard; however, these vendor-dependent approaches have limitations in effectively addressing threats arising from vendor heterogeneity.

To address these issues, I formulated the following research questions:

RQ 1: What are the unique characteristics of IAM vulnerabilities in multi-cloud environments compared to single-cloud environments?

RQ 2: What are the limitations of existing methodologies or security solutions in detecting IAM vulnerabilities in multi-cloud environments?

RQ 3: Can the proposed model effectively respond to vulnerability transfers detected in a multi-cloud environment?

In multi-cloud environments, the communication channel of the virtual private network (VPN), which ensures connections, represents an additional attack surface. A VPN guarantees secure data transmission between two infrastructure environments based on network traffic encryption. Thus, instead of targeting VPN vulnerabilities, attackers often exploit weaknesses in the Windows Credential Provider or network access permissions from hosts. Hence, this study focuses on detecting and responding to IAM-based vulnerabilities in multi-cloud environments. To perform penetration testing on these IAM-based vulnerabilities, I used scenarios provided by CloudGoat [16] as a baseline to identify potential IAM vulnerabilities and derive variants specific to multi-cloud environments. CloudGoat, developed by Rhino Security Labs, provides an environment for deploying and testing scenarios that simulate vulnerable cloud environments, facilitating the identification of and response to security threats in real-world cloud environments. This enables a systematic evaluation of risks associated with IAM configuration errors [17].

The multi-cloud environment configured in this study is as follows. For simplicity, unless otherwise noted, this study assumes two VPCs from different CSPs, Microsoft Azure and Amazon AWS, as shown in Figure 2. Each environment consists of a VPC, subnets, and virtual machines (e.g., EC2 instances) deployed within the subnets. These two environments are connected through a site-to-site VPN [18]. The site-to-site VPN is configured to route AWS instances, enabling communication with external networks (e.g., Azure instances). To

enable this, a customer gateway provides Azure's VPC information to AWS, which generates a virtual private gateway to connect to the VPC. Notably, linking different cloud platforms through a site-to-site VPN is commonly used to support seamless and secure data transfers. This facilitates the smooth migration of workloads and data between CSPs, aiding companies in achieving more efficient operations and cost management.

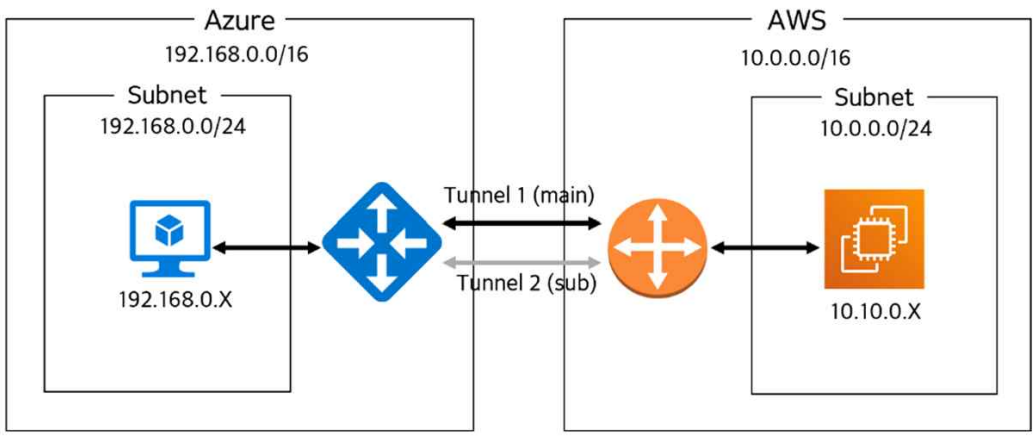

Figure 2. Strawman architecture for a multi-cloud environment

2. Threat Modeling of Multi-cloud Environment

This section presents STRIDE-LM threat modeling conducted based on a real-world multi-cloud environment. As shown in Figure 3, users can integrate additional cloud services such as Microsoft Defender for Cloud, Security Hub, IAM, and Active Directory. Table 1 presents various security threats that attackers can exploit according to the STRIDE-LM standard, considering each entity. Among these threats, misconfigurations in authentication links or privilege escalation through IAM policies were identified as potential risks that could compromise the multi-cloud system. Therefore, the research scope was defined as Denial of Service and Elevation of Privilege threats.

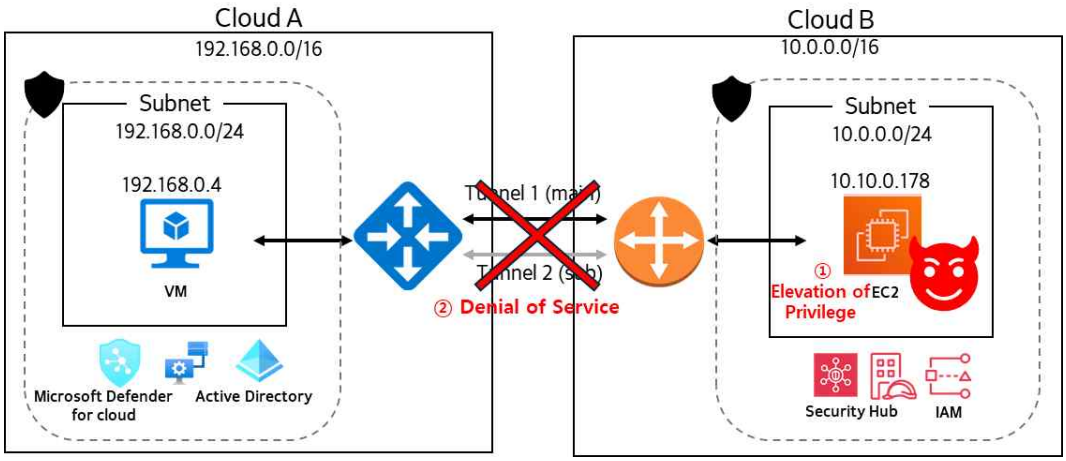

Figure 3. Threat modeling on the multi-cloud environment

Table 1. Security threats in a multi-cloud environment

STRIDE-LM	Threat Description
Spoofing	• An attacker gains unauthorized access by impersonating IAM or Active Directory (AD) credentials
Tampering	• Unauthorized modifications to log data or VM configurations • Tampering with data packets between EC2 and Azure VMs
Repudiation	• Deletion or modification of logs after privilege changes • Denial of authentication activities in Active Directory (AD)
Information Disclosure	• Exposure of sensitive data stored in EC2 instances or VMs to unauthorized parties • Misconfigured network settings that allow access to confidential data
Denial of Service	• Disabling security monitoring solutions by removing the authentication link between Security Hub and Microsoft Defender for Cloud • DDoS attacks that disable Security Hub or Microsoft Defender for Cloud operations • Service disruptions caused by VM overload
Elevation of Privilege	• Gaining administrative privileges via IAM policies • Compromising VM administrator privileges caused by misconfigured Active Directory (AD) permissions
Lateral Movement	• Expanding an attack from one CSP to another • An initial compromise in EC2 instances spreads to Azure VMs

3. IAM Security Threat Scenarios

Penetration testing in cloud environments involves simulating real-world attacks to identify and address vulnerabilities within cloud systems. However, existing tools are insufficient for conducting penetration tests specifically tailored to typical multi-cloud environments. To address this gap, existing cloud penetration testing tools were leveraged to derive potential attack scenarios. Among popular open-source tools for cloud penetration testing, CloudGoat [16] was selected to evaluate IAM security elements in multi-cloud environments. Table 2 provides a comparison of these open-source tools.

Focusing on misconfigurations in authentication links and privilege escalation through IAM policies, this study was based on three scenarios provided by CloudGoat [23, 24, 25], all targeting IAM permission-based security threats: iam_privsec_by_attachment, iam_privesc_by_rollback, and iam_privesc_by_key_rotation.

Table 2. Comparison of open source penetration testing tools

Tool Name	Description	Features	Limitations
Cloudgoat [16]	• An open-source tool by Rhino Security Labs designed to deploy vulnerable AWS	• Focused on AWS-specific vulnerabilities	• Exclusively limited to AWS • Requires an AWS account with sufficient privileges
Awsgoat· Azuregoat [19, 20]	• A deliberately vulnerable AWS and Azure infrastructure designed for security practitioners	• Includes OWASP Top 10 web application security risks • Utilizes Infrastructure as Code with Terraform • Emulates real-world AWS and Azure environments	• Exclusively designed for AWS and Azure
Stratus	• A tool	• Supports	• Potentially

[21]	designed to test security controls in cloud environments, with a focus on misconfiguratio ns and vulnerabilities	multiple cloud providers, including AWS, Azure, and GCP	limited community support compared to more established tools
Chaos Mesh [22]	• A tool developed by Netflix that randomly terminates instances in production environments to ensure system resilience	• Simulates various types of faults in Kubernetes clusters • Integrates with CI/CD pipelines for automated testing	• Primarily designed for Kubernetes

In the iam_privsec_by_attachment scenario, an attacker with limited permissions uses the instance-profile-attachment permission to create a new EC2 instance with full administrative privileges. This allows the attacker to access the newly created EC2 instance and gain permissions to perform all actions across all services and resources within the AWS account.

The iam_privesc_by_rollback scenario involves a privilege escalation attack where an IAM user with limited permissions restores a previous IAM policy version that grants full administrative rights, thereby gaining unauthorized access.

Finally, in the iam_privesc_by_key_rotation scenario, the attacker exploits an IAM user with weak security settings to expand access permissions and steal sensitive data from secret managers. This scenario involves three IAM user accounts (admin, developer, and manager), each with distinct permissions and policies.

However, these scenarios, which focus on privilege escalation and unauthorized access, are mainly designed for conventional single-cloud environments. This limitation underscores the need to proactively define new scenarios that consider the complexities of multiple and heterogeneous VPCs in multi-cloud environments.

Ⅳ. Ground Truth Derivation with Multi-cloud Penetration Testing

This section outlines the detailed procedures for each customized penetration testing scenario in multi-cloud environments. Each scenario was conducted in a multi-cloud environment involving two CSPs: CSP A and CSP B. Unless otherwise specified, the target multi-cloud architecture consists of Amazon AWS and Microsoft Azure cloud platforms, both of which provide mature security-related monitoring logs through Security Hub and Microsoft Defender, respectively.

For demonstration purposes, CSP A was configured as AWS, and CSP B as Azure to conduct scenario-based experiments in a real cloud environment. Solid lines in the following figures indicate activities performed within AWS, while dotted lines indicate activities performed within Azure. Additionally, blue and red boxes indicate conventional scenarios provided by CloudGoat and custom-defined steps derived from identifying vulnerability propagation in a multi-cloud environment, respectively. Table 3 summarizes the deployment information for each customized scenario.

Table 3. Information on the deployment of customized scenarios

#	Scenario	Deployment Information	Additional Settings
1	iam_privsec_by_attachment	1 VPC with 1 x EC2 VM, 1 IAM User	1 IAM user (attacker), Site-to-site VPN
2	iam_privesc_by_rollback	1 IAM User, 5 policy versions	
3	iam_privesc_by_key_rotation	3 IAM Users, 1 IAM Role, 1 Secret	

1. Customized Scenario 1: iam_privsec_by_attachment scenario

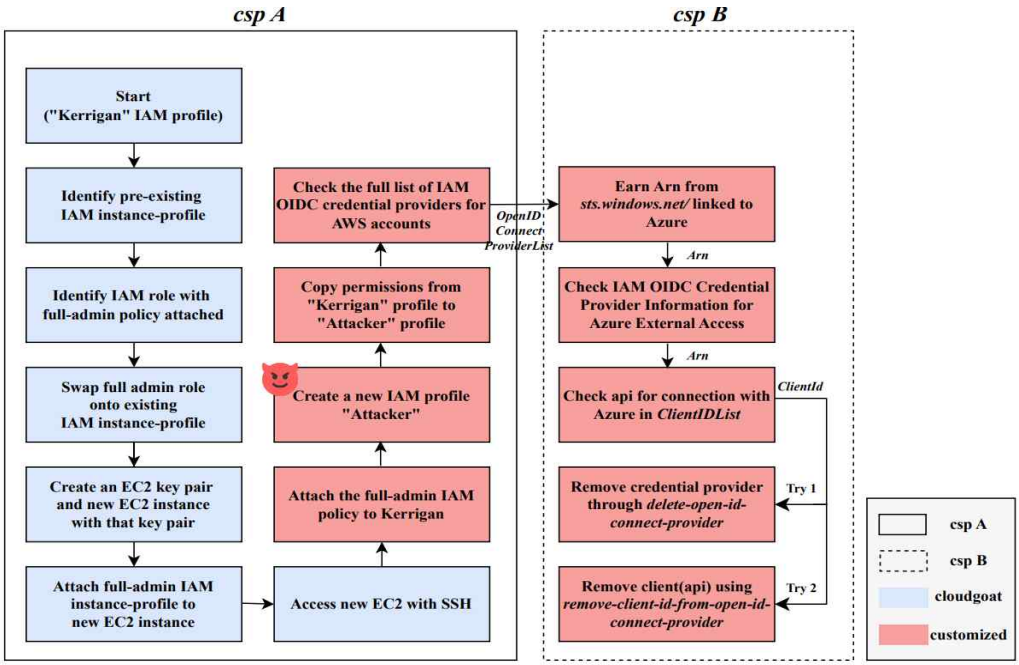

Figure 4. Customized IAM privilege escalation by attachment scenario

Figure 4 illustrates the detailed workflow for Scenario 1. The attacker, represented as a newly created IAM user named Kerrigan, starts with limited permissions and explores existing instance profiles and roles to identify exploitable resources. An instance profile is a resource that assigns roles to an EC2 instance. The attacker discovers a full administrator role (cg-ec2-mighty-policy) and modifies the instance profile to assume it.

The attacker then creates a new EC2 instance and assigns it an

instance profile with full administrative privileges, thereby gaining complete administrative rights. After accessing the EC2 instance, the attacker creates and uses a new IAM user named Attacker, granting it full administrative privileges. The attacker lists the IAM OpenID Connect (OIDC) credential providers associated with the AWS account and deletes the providers and client IDs linked to Azure Defender connections. This action ultimately disconnects Azure Defender, which integrates the security environments of AWS and Azure, preventing it from monitoring the security status of AWS resources in the multi-cloud environment. This attack scenario highlights a mismatch between two legacy security tools operating across different cloud platforms. Scenario 1 involves an attack using an instance profile with full administrative privileges; therefore, I reviewed the security recommendations related to EC2. During this inspection, I observed a potential mismatch in security recommendations between Microsoft Azure Defender and AWS Security Hub. Specifically, AWS and Azure assess whether the created IAM policy grants full administrative rights for all actions on a resource or employs wildcards to allow unrestricted actions on a specific service. This creates a security threat resulting from improper authorization. In this scenario, the attacker escalates privileges by replacing the EC2 instance's role with one granting full administrative rights through an instance profile. The results show that AWS Security Hub classified the policy as FAILED, whereas Microsoft Azure Defender for Cloud classified it as Healthy, highlighting a mismatch.

Table 4. Cases of mismatch in customized scenario 1

Vendor	Resource Name	Status	Description	IAM policy
AWS	cg-ec2-mighty-policy	FAILED	IAM policies should not allow full “*” administrative privileges	IAM.1
AZURE		Helathy	IAM policies that allow full “*:*” administrative privileges should not be created	
AWS	cg-ec2-mighty-policy	FAILED	IAM customer managed policies that you create should not allow wildcard actions for service	IAM.21
AZURE		Healthy	IAM customer managed policies that you crate should not allow wildcard actions for services	
AWS	Account	FAILED	Hardware MFA should be enabled for the root user	IAM.6
AZURE		Healthy	Hardware MFA should be enabled for the “root” account	

.

Table 4 presents the raw log data for the mismatch cases in Scenario 1. The cg-ec2-mighty-policy refers to an AWS policy exploited by an attacker in this scenario, granting full administrative privileges. Rule IAM.1 specifies that IAM policies should not use the wildcard character * to grant full administrative privileges for all actions. Similarly, rule IAM.21 states that customer-managed IAM policies created by users should not allow wildcard actions for services. Using * permits all actions on all resources within the environment without specifying particular resources or actions.

For these IAM rules, AWS evaluated the cg-ec2-mighty-policy as FAILED, while Azure rated it as Healthy. The “Account” column refers to a specific account, and rule IAM.6 checks whether hardware MFA is enabled for the root user of this account. AWS also evaluated this rule as FAILED, whereas Azure rated it as Healthy, highlighting differing evaluation results for the same rule.

2. Customized Scenario 2: iam_privsec_by_rollback scenario

In Scenario 2, the attakcer attempts to gain administrative privileges through a privilege escalation attack using an IAM user with limited permissions to access sensitive data. Figure 5 illustrates the flowchart of the iam_privsec_by_rollback scenario.

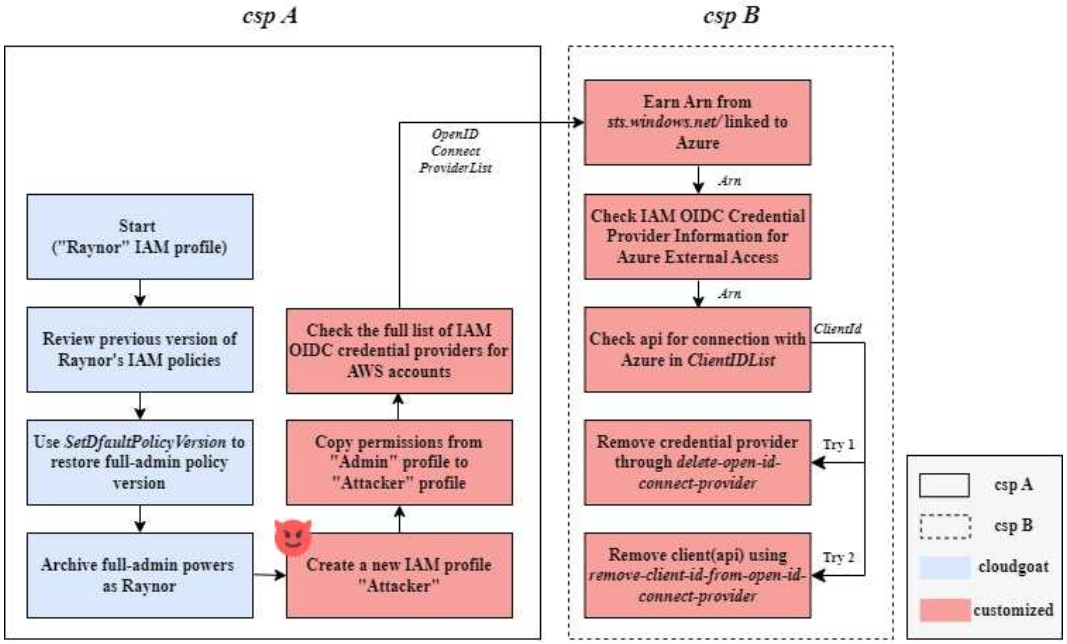

Figure 5. Customized IAM privilege escalation by rollback scenario

The scenario begins with a default IAM user account (Raynor). An analysis of the account permissions reveals five policy versions, including versions with IAM permissions and an administrator policy version. The IAM permission allows the default version of the IAM policy to be set using the AWS API, enabling the restoration of the administrator policy version. With the obtained administrative privileges,

an arbitrary account can be created to perform malicious actions.

Figure 6 demonstrates the process of this privilege escalation attack. Among the five policy versions shown, version v5 is set as the default version (IsDefaultVersion: True). By setting version v2, which grants access permissions for all actions and resources, as the default, access permissions can be easily compromised (indicated in red). After acquiring full administrative privileges, additional attacks were simulated to demonstrate attack transitions in a multi-vendor environment.

The Raynor IAM user, having obtained full administrative privileges, creates a new IAM user (Attacker) and replicates their privileges. The Attacker IAM user, with full administrative privileges, queries the entire IAM OIDC Credential Provider list of the AWS account to identify external access information and APIs linked to Azure. Using the obtained API, the attacker deletes clients and credential providers, thereby forcibly disconnecting the connection between the AWS and Azure environments.

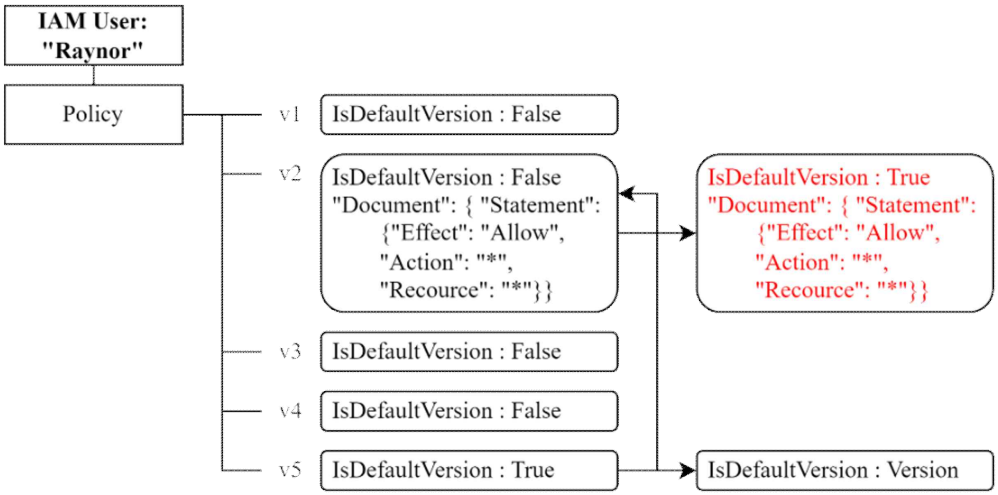

Figure 6. Privilege escalation attack phase

As in Scenario 1, a mismatch occurs between the logs generated by Azure Defender and AWS Security Hub due to differences in their evaluation criteria. First, regarding access key rotation (IAM.17), AWS recommends periodically rotating access keys, as these keys grant full access rights to resources. Failure to rotate the access key periodically prompts AWS to recommend additional security measures. Consequently, AWS Security Hub classified the status as FAILED because the necessary actions were not performed. In contrast, Azure classified the same scenario as Healthy, resulting in a mismatch. Second, AWS mandates the use of multi-factor authentication (MFA) for all IAM users utilizing console passwords (IAM.5). The absence of MFA settings led AWS Security Hub to classify the status as FAILED, whereas Azure classified it as Healthy, further highlighting the inconsistency between the two platforms.

Table 5. Cases of mismatch in customized scenario 2

Vendor	Resource Name	Status	Description	IAM policy
AWS	raynor-iam_privesec_by_rollback, attakcer	FAILED	Ensure IAM password policy expires passwords within 90 days or less	IAM.17
AZURE		Helathy	Ensure access keys are rotated every 90 days or less	
AWS	user/ multicloud	FAILED	MFA should be enabled for all IAM users that have a console password	IAM.5
AZURE		Healthy	Do not setup access keys during initial user setup for all IAM users that have a console password	

Table 5 summarizes additional mismatch cases from Scenario 2. AWS recommends setting password rotation policies to expire passwords within 90 days. For the scenario's IAM users "raynor-iam_privesec_by_rollback" and "attacker", the absence of password expiration resulted in AWS evaluating the status as FAILED, while Azure assessed it as Healthy. Furthermore, AWS flagged the "user/multicloud" resource, used to connect AWS and Azure, as FAILED due to the absence of MFA. Azure, however, evaluated this as Healthy, further illustrating the divergent approaches of the two platforms.

3. Customized Scenario 3: iam_privsec_by_key_rotation scenario

In Scenario 3, three types of IAM user profiles—manager, developer, and admin—were considered, each with distinct policies. The admin profile, which has permissions to modify access keys, replaced and created access keys in the scenario. After enabling MFA for the newly created keys, these keys were used to retrieve secret information from AWS Secrets Manager. While this baseline scenario was performed within AWS, the custom scenario introduced a new IAM profile called Attacker (Figure 7).

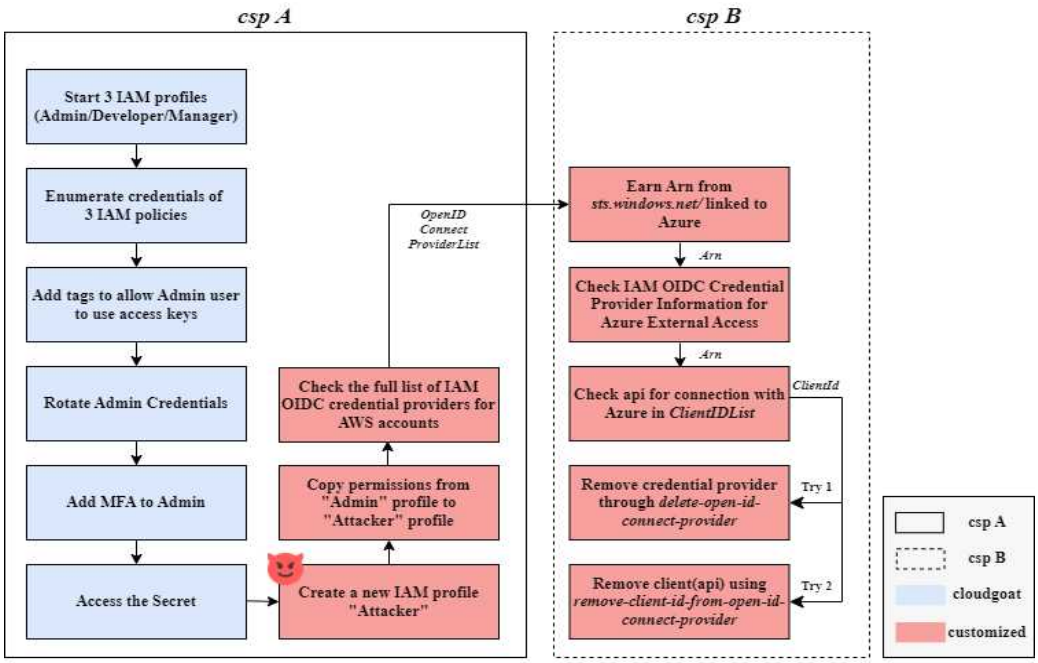

Figure 7. Customized IAM privilege escalation by key rotation scenario

Leveraging the AssumeRoles policy, the admin can delegate permissions to other users, allowing tasks to be performed with permissions from other accounts without direct resource access. The attacker exploited this by copying the admin's privileges, querying the entire IAM OIDC Credential Provider list, and disrupting multi-cloud security monitoring by disconnecting the OIDC link between vendors.

During the analysis of this scenario, another mismatch in security recommendations was identified. For the lambda-data-policies resource, AWS classified the policy as FAILED due to wildcard permissions, which violate IAM.21, a rule against granting wildcard actions in customer-managed policies. In contrast, Azure classified the same policy as Healthy, reflecting differences in their assessment criteria. Similarly, AWS marked the admin_iam_privesc_by_key_rotation policy as FAILED due to the absence of MFA (IAM.19), while Azure evaluated it as Healthy. A similar trend was observed for the developer and manager profiles, where AWS's stricter requirements for policy specificity clashed with Azure's more lenient approach.

Table 6. Cases of mismatch in customized scenario 3

Vendor	Resource Name	Status	Description	IAM policy
AWS	lambda-data-policies	FAILED	IAM customer managed policies that you create should not allow wildcard actions for services	IAM.21
AZURE		Helathy	IAM customer managed policies that you create should not allow wildcard actions for services	
AWS	admin, developer, manager_iam_privsec_by_key_rotation_cgidg7eix4mk8a	FAILED	MFA should be enabled for all IAM users	IAM.19
AZURE		Healthy	IAM policies should be attached only to groups or roles	

Table 6 highlights a specific mismatch from Scenario 3 involving the lambda-data-policies resource. This policy governs data access and per missions for AWS Lambda functions. AWS flagged the policy as FAIL ED for non-compliance with IAM.21 due to wildcard permissions, while Azure classified it as Healthy. This discrepancy underscores fundamenta l differences in policy enforcement and compliance criteria between AW S and Azure, reflecting the challenges in achieving consistent security a ssessments across multi-cloud environments.

Ⅴ. Interoperable Security Information and Event Management Framework (iSIEM)

1. Components of iSIEM

To this end, I propose iSIEM for efficient security management in multi-cloud environments that provides consistent responses to security threats across various cloud platforms. The proposed methodology addresses the limitation of manually reviewing a vast number of unnecessary logs to identify threats arising from mismatches between logs generated by monitoring tools in multi-vendor environments. This was achieved using a ground-truth database derived from penetration testing scenarios. An additional database is employed to analyze the causes of security logs collected from different CSP monitoring tools that fail to map to each other. An analysis of unmapped logs from customized scenarios in multi-cloud is presented in Section 6.3. The overall architecture of the system framework is presented in Figure 8.

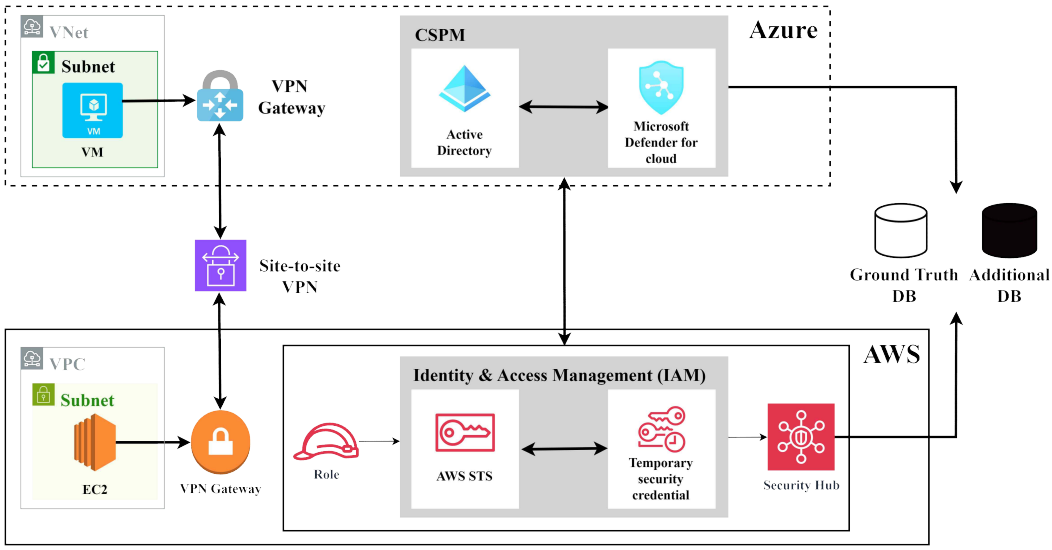

Figure 8. Framework of iSIEM

Based on the comprehensive attack surface analysis, I defined three custom attack scenarios (see section 4), which are mutations of the original CloudGoat scenarios, to project potential threats. Through preliminary customized penetration testing scenarios (see section 4), I confirmed that conventional security-monitoring tools generated different alerts or missed security logs owing to the same security issue in a multi-cloud environment. Therefore, I defined the mismatch cases of the security logs collected from the monitoring tools of each CSP, as shown in Table 7. Table 7 illustrates three cases: 'O' for compliance with security rules, '△' for cases wherein the source identifies an issue but the destination does not, and 'X' for non-compliance with security rules in a multi-cloud environment.

In the conventional approach, it is unclear whether threats are indeed propagated. This ambiguity arises because the security monitoring tools

on the propagation source platform can detect IAM security threats, while those on the propagation destination platform may not recognize these threats (e.g., "△" cases in Table 7). In contrast, iSIEM can provide a unified assessment despite discrepancies between the AWS and Azure logs as it is evaluated based on a ground-truth database derived from the penetration test.

Table 7. Results of auditing security logs from AWS and Azure

Logs from AWS	Logs from Azure	Conventional approach	iSIEM
FAILED	Unhealthy	X	X
FAILED	Healthy	△	X
PASSED	Unhealthy	△	X
PASSED	Healthy	O	O

2. Workflow of iSIEM

The overall workflow of the iSIEM framework is presented in Figure 9. The iSIEM workflow is broadly divided into regularization and determination phases.

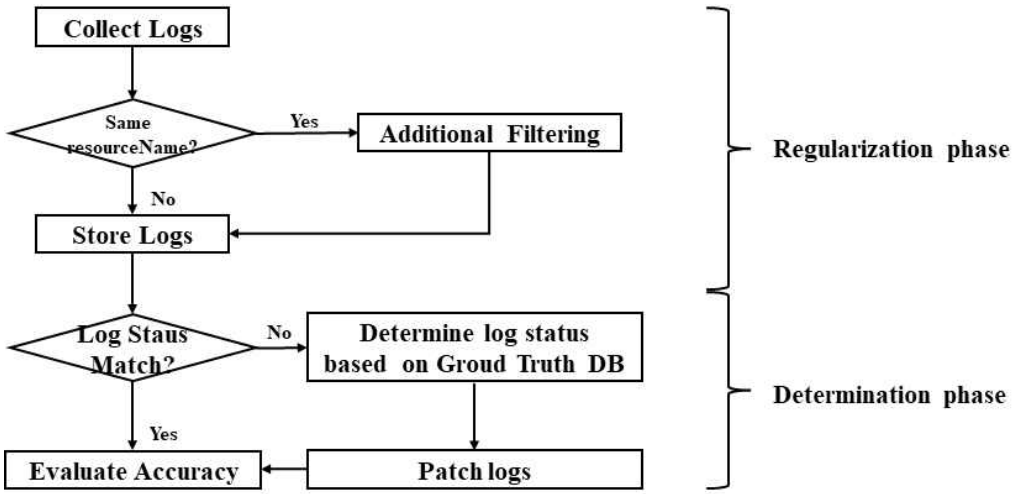

Figure 9. Workflow of iSIEM

Before performing regularization, iSIEM collects security logs from each of CSP's security monitoring tools, AWS Security Hub and Azure Microsoft Defender, for the Cloud. These tools were intrinsically configured to gather security logs only from resources within their specific clouds. To accommodate workloads and support function partitioning in a multi-cloud environment, iSIEM links accounts between AWS and Azure through an identity provider (OpenID Connect). This integration enables each security-monitoring tool to collect security logs

from resources distributed across multiple clouds via iSIEM.

iSIEM then preprocesses the logs by grouping them based on resourceName. To handle cases in which multiple logs represent the same resourceName, iSIEM performs additional filtering. During this process, the logs are mapped based on each CSP's log description, including Azure's description data and AWS's IAM role. Logs that are not mapped are stored in an additional database.

During the determination phase, iSIEM identifies mismatched cases based on the state information of the logs. The criteria for determining these mismatces are derived from the ground-truth database, which is discussed in Section 4 with three custom attack scenarios. These criteria are used to patch incorrect states. Through these phases, iSIEM demonstrates an improved accuracy, recall, and F1-score compared to conventional monitoring tools.

3. Database Schema of iSIEM

The database structure is categorized into three sections, SECURITY_LOG, MISMATCH_LOG, and GROUND_TRUTH_DB, as shown in Figure 10.

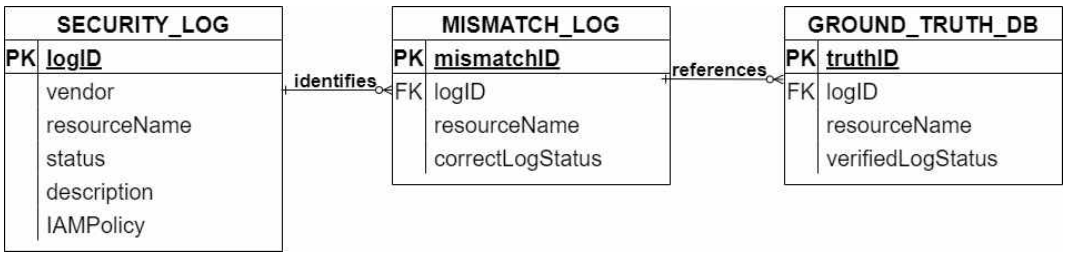

Figure 10. DB schema of iSIEM

The database schema for the ground truth was initially constructed through customized experiments (Section 4) and served as a reference for identifying mismatches in the logs generated thereafter. Each section is designed to contain specific types of log data, thus facilitating the detection and analysis of mismatches and ensuring the integrity and accuracy of security assessments. Each log entry in the SECURITY_LOG table is uniquely identified by the log_id, which serves as a reference key in the other tables. The MISMATCH_LOG and GROUND_TRUTH_DB tables contain the log_id field that acts as a foreign key, which links them back to the original entries in the SECURITY_LOG table. This relational structure ensures that all the log data are systematically organized and interconnected, thus facilitating a comprehensive analysis and an accurate risk assessment. If a mismatch occurs, iSIEM applies an OR condit

ion algorithm. It assesses the mismatched cases (FAILED-Healthy and PASSED-Unhealthy) as non-compliant with security rules in a multi-cloud environment and patches the correct state from the GROUND_TRUTH_DB.

By organizing the logs into these categories and establishing clear primary key (PK) and foreign key (FK) relationships, the iSIEM framework ensures a structured approach to log management and security assessment. This relational database schema enhances the accuracy of security monitoring and significantly reduces the false-negative rate in a multi-cloud environment. This underscores the importance of maintaining a robust GROUND_TRUTH_DB for an effective risk assessment in multi-cloud environment.

Ⅵ. Experiment Analysis

To build the architecture presented in Figure 2, I created an EC2 instance on AWS using the t2.micro instance type, with Ubuntu 20.04, 1 vCPU, and 1 GiB of RAM. Similarly, on Azure, I launched a VM using the Standard B1s instance type with Ubuntu 20.04, 1 vCPU, and 1 GiB of RAM.

1. Ground Truth Database Construction

First, I analyzed the security logs collected using the proposed iSIEM model. In Scenario 1, 47 and 100 logs were collected from AWS Security Hub and Azure Defender, respectively. In Scenario 2, 28 and 41 logs were collected from AWS Security Hub and Azure Defender, respectively. In Scenario 3, 90 and 74 logs were collected from AWS Security Hub and Azure Defender, respectively. These discrepancies in the number of logs collected for the same actions in a multi-cloud environment highlight the limitations of using a single monitoring tool for security assessments in such a setting. iSIEM addresses these limitations through regularization and mismatch determinations.

During the regularization stage, the collected security logs from each scenario are consolidated, and log mapping is performed using resourceName and Description as key values. In Scenario 1, 56 of the 147 collected logs (approximately 38%) were mapped. In Scenario 2, 26 (approximately 36%) of the 73 collected logs were mapped. In Scenario

3, 94 (approximately 57%) of the 164 collected logs were mapped. The numbers of mapped logs are listed in Table 8. The colored section indicates the sum of the patched logs and the existing True Positive logs. All three scenarios exhibit a log-mapping rate of approximately 50%, indicating a lack of compatibility among the security tools of different vendors. Next, during the mismatch determination stage, logs with mismatched states were assessed using the ground-truth DB based on the scenario. Conventional monitoring tools designed for each CSP often lack a basis for determination in multi-cloud threat situations and fail to warn users even when an attack occurs.

Table 8. Counts of mapped and patched logs collected by each monitoring tool

AWS	Azure	Scenario 1		Scenario 2		Scenario 3	
		Conv	iSIEM	Conv	iSIEM	Conv	iSIEM
FAILED	Healthy (FN)	6	0	6	0	6	0
	Unhealthy (TP)	18	24	8	14	2	8
PASSED	Healthy (TN)	32	32	12	12	39	39
	Unhealthy (FP)	0	0	0	0	0	0

2. Mismatch Determination

Based on the OR condition algorithm, iSIEM classifies FAILED-Unhealthy, FAILED-Healthy, and PASSED-Healthy as vulnerable, propagated, and benign cases, respectively. Given that AWS is defined as the source platform and Azure as the destination platform in customized penetration testing, this study constructed a ground-truth database based on the log status from AWS. For each status, FAILED and Unhealthy indicates non-compliant cases. Thus, FAILED-Unhealthy is defined as a True Positive (TP). Other logs were categorized using the indicators: True Positives (TPs), True Negatives (TNs), False Positives (FPs), and False Negatives (FNs). Since the setting defines AWS and Azure as the source and destination platforms, respectively, there were no logs indicating PASSED-Unhealthy. Therefore, only FAILED-Healthy was considered a propagated case. In propagated cases, conventional methods fail to detect transferred IAM vulnerabilities and classify them as Healthy, resulting in zero logs for security responses. In contrast, the proposed iSIEM performs patches based on a ground-truth database for mismatch cases, thereby enabling security responses for logs with propagated vulnerabilities.

Figure 11 presents the comparison between the conventional and proposed methods. Conventional (Conv) refers to the use of individual monitoring tools within each VPC, while Proposed denotes the suggested iSIEM model. In each custom scenario, the conventional method achieves limited accuracy, recall, and F1-score due to

mismatched logs. Conversely, the proposed model achieves 100% in these metrics through an OR-condition-based patching process. Consequently, each scenario demonstrated improvements in accuracy by 10.71%, 23.08%, and 12.77%, respectively. The recall improved by 25%, 42.86%, and 75%, while the F1-score increased by 14.29%, 27.27%, and 60%, respectively. These improvements were achieved by revising False Negative results through the patching process.

Among the collected logs, the accuracy, recall, and F1-score were calculated for logs that were successfully mapped between each control tool. The lessons learned from this analysis are further discussed in Section 7.

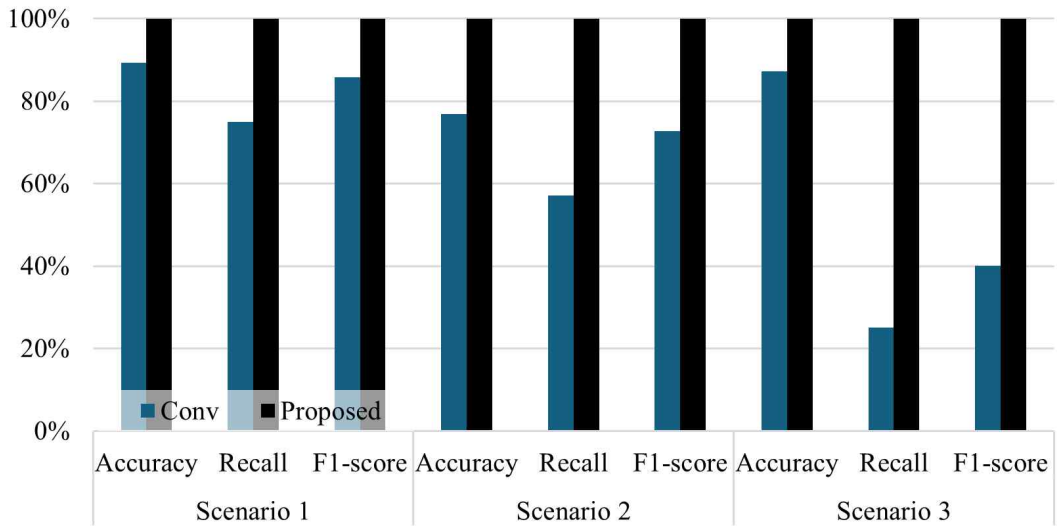

Figure 11. Comparison of evaluation metrics' performance between the proposed and conventional methods

3. Analysis of Unmapped Logs

Owing to the differing security policies among vendors, the proportion of mapped security logs in a multi-cloud environment for the same event was approximately 50%. iSIEM identifies the causes of unmapped security logs by storing them in a separate database for additional filtering. Although all three scenarios tested security threats in a multi-cloud environment through IAM privilege escalation attacks, they exhibited varying tendencies in terms of service usage and approaches to security policies. This section identifies the unmapped logs for each scenario and analyzes their causes based on the specific scenario.

In Scenario 1, Azure examines whether decryption operations for all Key Management Service (KMS) keys are possible through IAM inline policies for unique IAM identifiers (AGPA, AIDA, AROA), whereas AWS evaluates these security items based on KMS rules. This disparity complicates the analysis using IAM rules alone and necessitates additional reference to KMS rules. AWS Security Hub evaluates IAM user passwords based on robust security policies, whereas Azure Defender only verifies whether the IAM password policy includes at least one uppercase letter. Furthermore, AWS configures password policies through IAM, while Azure utilizes Azure Active Directory (AD) to manage password policies. This difference necessitates the application of additional filtering criteria beyond IAM rules.

In Scenario 2, logs related to IAM Policy were not mapped. AWS

defines several policies for IAM, such as wildcard permissions, weak password settings, and root user configurations. In contrast, Azure Defender does not define detailed items for IAM Policy, leading to a significant number of unmapped logs. Both AWS and Azure assign the unique ID prefix 'AROA' to security logs concerning IAM Role types. However, during log collection, AWS evaluates security using KMS rules, while Azure Defender assesses security using IAM rules, further contributing to the unmapped log discrepancies.

In Scenario 3, user/multi-cloud logs, IAM Role logs, and attacker logs were not mapped. Logs related to the multi-cloud environment connecting AWS and Azure were present in AWS but absent in Azure. For example, 'AROA' logs related to IAM roles involving key-related operations were evaluated using KMS rules in AWS, making it challenging to analyze these logs solely based on IAM rules. Additionally, attacker-related logs were only available in AWS Security Hub, as Azure did not recognize the process of creating and defining the attacker. These unmapped logs highlight the challenges of assessing and responding to attacks in a multi-vendor environment without a ground-truth-based evaluation metric such as iSIEM.

Ⅶ. Discussion

Conventional security-monitoring services in a multi-cloud environment still face several challenges. First, configuration security measures are often inadequate during the multi-cloud setup process. The integration of various cloud platforms can lead to inconsistent security settings across platforms. For example, a specific setting might be correctly configured in AWS but not in Azure, resulting in security vulnerabilities. Additionally, differences in log aggregation methods between AWS and Azure further complicate security monitoring in a multi-cloud environment. For instance, AWS collects logs based on system events, while Azure aggregates them around access-control events. These differences make integrated log management and analysis in a multi-cloud setting particularly challenging.

In this study's scenario experiments, accuracy improved by at least 10.8% and up to 23.1%, recall improved by at least 25% and up to 75%, and the F1-score improved by at least 14.3% and up to 60%. These improvements can be attributed to the need for standardization in a heterogeneous environment, where an imbalance in the log quality among foundational security tools used to derive the ground truth is evident. This analysis highlights the necessity of minimizing discrepancies between security solutions to adequately address a broader range of threat scenarios. Moreover, the research reveals that approximately 50% of security logs remain unmapped and were not

simply filtered out. These unmapped logs represent a significant opportunity to enhance the ground-truth database, thereby facilitating the detection of potential mismatches and security threats that might otherwise go unnoticed. For example, specific unmapped logs related to IAM vulnerabilities could be categorized and relabeled to increase their utility in security analyses. Although further investigation is required to determine the exact instances from which these unmapped logs originate, the mismatch determination capabilities of iSIEM could significantly improve accuracy.

This study also discusses the effectiveness of the OR condition algorithm for managing security across multiple CSPs. While this algorithm functions adequately under current conditions, its performance in more complex multi-cloud environments, such as sky computing, warrants further exploration. Consequently, future research should focus on developing more suitable algorithms to better address the unique challenges posed by multi-cloud environments.

Finally, protective measures for OIDC disconnection in a multi-cloud environment remain insufficient. The use of a centralized authentication management system through OIDC in a multi-cloud environment facilitates the unified management of user credentials, making OIDC a critical component in such settings. OIDC enables secure and efficient user credential exchanges via token-based authentication. However, inadequate protection of OIDC connections can lead to severe security threats. In the aforementioned scenarios, if the validity of an OIDC token is not properly verified, unauthorized users may gain access,

leading to malicious attacks. This situation not only results in vulnerabilities in authentication and access control within a multi-cloud environment, but also hinders compatibility among cloud security monitoring services. Consequently, it is imperative to enhance security measures for OIDC in multi-cloud environments to both diminish the attack surface and improve compatibility across diverse security monitoring services.

Ⅷ. Conclusion

The demand for multi-cloud architectures is rising, as they offer numerous technical and economic benefits over conventional single-cloud setups. Many organizations are adopting multi-cloud strategies, and vendors are developing products tailored specifically for these environments. However, multi-cloud systems integrate diverse service resources, significantly expanding the potential attack surface compared to single-cloud setups. Identifying this attack surface, which serves as the primary gateway for attackers, is crucial for ensuring robust security measures.

This study utilized CloudGoat as a benchmark tool for penetration testing in a multi-cloud scenario that incorporates a site-to-site VPN. The findings highlight the substantial impact of privilege escalation attacks within real-world multi-cloud frameworks. To address the limitations of conventional single-cloud monitoring, this study introduces the iSIEM system, which consolidates and evaluates security logs from multiple CSPs (Cloud Service Providers). This novel approach enhances security log monitoring by leveraging a verified ground-truth database, reducing the false-negative rate and improving accuracy by 10.71% to 23.08% compared to conventional tools.

Future research will expand security log integration to include other prominent CSPs, such as Google Cloud Platform (GCP), to enhance the generalizability and applicability of this method. Additionally, this study will explore security threats and countermeasures across various

multi-cloud service instances. Efforts will also focus on standardizing security logs, which serve as critical input data, to improve the accuracy of detecting and mitigating propagated attacks.

References

[1] Hamza Ali Imran, Usama Latif, Ataul Aziz Ikram, Maryam Ehsan, Ahmed Jamal Ikram, Waleed Ahmad Khan, and Saad Wazir. Multi-cloud: a comprehensive review. In 2020 ieee 23rd international multitopic conference (inmic), pages 1 - 5. IEEE, 2020.

[2] Aditi Rajan Khot. A comparative analysis of public cloud platforms and introduction of multi-cloud. International Journal of Innovative Science and Research Technology, 5(9):448 - 454, 2020

[3] Ion Stoica and Scott Shenker. From cloud computing to sky computing. In Proceedings of the Workshop on Hot Topics in Operating Systems, pages 26 - 32, 2021.

[4] Rajendra Patil, Harsha Dudeja, and Chirag Modi. Designing an efficient security framework for detecting intrusions in virtual network of cloud computing. Computers & Security, 85:402 - 422, 2019.

[5] Shaharyar Khan, Ilya Kabanov, Yunke Hua, and Stuart Madnick. A systematic analysis of the capital one data breach: Critical lessons learned. ACM Transactions on Privacy and Security, 26(1):1 - 29, 2022.

[6] Nicolae Paladi, Antonis Michalas, and Hai-Van Dang. Towards secure cloud orchestration for multi-cloud deployments. In Proceedings of the 5th Workshop on CrossCloud Infrastructures & Platforms, pages 1 - 6, 2018.

[7] Juncal Alonso, Leire Orue-Echevarria, Valentina Casola, Ana Isabel Torre, Maider Huarte, Eneko Osaba, and Jesus L Lobo.

Understanding the challenges and novel architectural models of multi-cloud native applications – a systematic literature review. Journal of Cloud Computing, 12(1):6, 2023.

[8] Flexera. 2024 state of the cloud report, 2024. Accessed: 2024-06-30.

[9] Morgan Reece, Theodore Edward Lander Jr, Matthew Stoffolano, Andy Sampson, Josiah Dykstra, Sudip Mittal, and Nidhi Rastogi. Systemic risk and vulnerability analysis of multi-cloud environments. arXiv preprint arXiv:2306.01862, 2023.

[10] Cloud Security Alliance. Cloud security alliance's top threats to cloud computing: Pandemic 11 report finds traditional cloud security issues becoming less concerning, 2022. Accessed: 2024-06-30.

[11] OWASP Foundation. Owasp cloud-native application security top 10, 2022. Accessed: 2024-06-30

[12] Thijs van Ede, Niek Khasuntsev, Bas Steen, and Andrea Continella. Detecting anomalous misconfigurations in aws identity and access management policies. In Proceedings of the 2022 on Cloud Computing Security Workshop, pages 63 – 74, 2022.

[13] Ilia Shevrin and Oded Margalit. Detecting {Multi-Step}{IAM} attacks in {AWS} environments via model checking. In 32nd USENIX Security Symposium (USENIX Security 23), pages 6025 – 6042, 2023.

[14] AWS. Aws security hub, 2024. Accessed: 2024-06-30

[15] Microsoft. Microsoft defender for cloud, 2024. Accessed: 2024-06-30.

[16] Rhino Security Labs. Cloudgoat, 2024. Accessed: 2024-06-30.

[17] Viktor Engstr¨om, Pontus Johnson, Robert Lagerstr¨om, Erik Ringdahl, and Max W¨allstedt. Automated security assessments of

amazon web services environments. ACM Transactions on Privacy and Security, 26(2):1 - 31, 2023.

[18] Microsoft. Quickstart: Onboard aws accounts to defender for cloud, 2024. Accessed: 2024-06-30.

[19] ine-labs. Cloudgoat, 2024. Accessed: 2024-06-30.

[20] ine-labs. Cloudgoat, 2024. Accessed: 2024-06-30.

[21] DataDog. stratus-red-team, 2024. Accessed: 2024-06-30.

[22] Chaos-mesh, 2024. Accessed: 2024-06-30.

[23] Rhino Security Labs. IAM privilege escalation by key attachment scenario, 2024. Accessed: 2024-06-30.

[24] Rhino Security Labs. IAM privilege escalation by key rollback scenario, 2024. Accessed: 2024-06-30.

[25] Rhino Security Labs. IAM privilege escalation by key rotation scenario, 2024. Accessed: 2024-06-30.

논문 개요

멀티 클라우드 시스템을 위한 효율적인 보안 관리 메커니즘

류정화
미래융합기술공학과
성신여자대학교 대학원

멀티 클라우드 환경은 여러 클라우드 서비스 제공업체(CSP)의 자원을 활용하여 종래 단일 클라우드 환경 대비 다양한 기술적·경제적 이점을 제공한다. 그러나 종래 보안 관제도구들은 멀티 클라우드 환경에 특화된 보안 요소들을 고려하지 못하고 있다. 특히, 여러 CSP의 서비스를 통합함에 따라 일관된 보안 정책을 유지하는 데 어려움이 발생하며, 아이덴티티 및 접근 관리(IAM) 기반의 측면 이동 및 새로운 보안 위협은 멀티 클라우드 환경 전반의 보안 위협으로 작용한다. 본 연구는 멀티 클라우드 환경에서의 잠재적인 IAM 보안 위협에 대한 분석을 제공하고, 보안 모니터링 클라우드 서비스 간의 상호 운용성과 호환성을 향상시키기 위한 새로운 보안 대응 방법론인 iSIEM(상호 운용 보안 정보 및 이벤트 관리 프레임워크)을 제안한다. 이를 위해, CloudGoat를 기반으로 한 세 가지 침투 테스트 시나리오를 도출하여 다중 클라우드 환경에서의 실제 IAM 보안 위협을 평가했다. iSIEM 방법론은 클라우드 간 워크로드 이동성을 지원하고,

각 클라우드 보안 도구에서 생성된 보안 로그의 불일치 판단으로 인해 발생하는 보안 로그 간 불일치를 해결한다. 평가 결과, iSIEM은 다중 클라우드 환경에서 보안 관리의 효율성을 향상시키며, 단일 CSP에 의존하는 기존 보안 도구 대비 최소 10.71%에서 최대 23.08%까지 정확도가 향상되었다.

ACKNOWLEDGEMENTS

본 논문을 지도해주신 이일구 교수님과 김성민 교수님, 공저자로 함께 기여해 준 김서이, 김리영, 김예은 학생께 감사드립니다. 본 논문은 Securecomm 2024에서 발표한 논문을 확장·보완하여 작성했습니다.