УДК 004.056:004.75

CLOUD SECURITY MISCONFIGURATIONS: RISKS, CASE STUDIES, AND INFRASTRUCTURE AS CODE SOLUTIONS WITH TERRAFORM

Driss Alaoui Soulimani, Kadatska Olha
e-mail: olha.kadatska@nure.ua
Kharkiv National University of Radio Electronics,
V.V. Popovskyy dep. ICE,
Kharkiv, Ukraine

Cloud misconfigurations are among the leading causes of security incidents in contemporary infrastructures. This article examines how misconfigurations in access controls and encryption will affect cloud security. Case studies are discussed to bring out the challenges that come with managing complex environments. The paper also discusses how Infrastructure as Code tools, including Terraform, may be leveraged to enforce robust access control and encryption policies, thus minimizing human error and enhancing security.

The flexibility and scalability of such cloud platforms as AWS redefined the IT infrastructure landscape. But this advantage brought another major challenge in its wake, misconfiguration. From too permissive access controls to encryption not being done, configuration mistakes open the door to critical data and systems being compromised by malicious actors. In this article, the analysis of some of these risks, mostly around access control and encryption, introduces Terraform for secure and reproducible cloud configurations.

Cloud misconfigurations refer to errors in setting up cloud resources, resulting in vulnerabilities. Examples include open storage buckets, overly broad permissions, and lack of encryption [1]. These misconfigurations often arise due to:

Human Error, Manual configuration increases the likelihood of mistakes.

Complex Policies, Understanding and managing access control and encryption across diverse services is challenging.

Insufficient Monitoring, without tools to detect issues, vulnerabilities persist unnoticed.

Access Control in Cloud Environments ensures only authorized entities can access specific resources. Role-Based Access Control (RBAC): Assigns permissions based on predefined roles. For example, Kubernetes uses RBAC to manage access within namespaces or clusters by defining roles (read, write) and binding them to users or services.

Key components include Identity and Access Management (IAM): Centrally manages users, roles, and permissions. Cloud providers like AWS enable granular control, such as restricting access to specific S3 buckets or enforcing multi-factor authentication (MFA).

Failures in these areas, such as granting excessive permissions or neglecting policy updates, are common sources of breaches.

Encryption in Cloud Security protects sensitive data in transit and at rest as presented in Table 1. Inadequate implementation, such as unencrypted S3 buckets or weak key management, leaves data vulnerable

Real-World Scenario Problem: Misconfigurations and Their Consequences

Capital One Breach a misconfigured S3 bucket with public access exposed sensitive data, impacting over 100 million individuals.

Accenture Breach AWS S3 bucket misconfigurations revealed internal data, including API keys and private client information.

These breaches underline two very important concepts: access control and encryption. AWS S3 is a scalable object storage that, because of its flexibility and wide usage, often becomes a point of focus for misconfigurations.

Terraform is important in cloud security, which enforces key security features such as RBAC. This ensures the principle of least privilege since strict roles and policies are provided. Besides, it fortifies access control through proper role-based access, ensuring that only authorized users and services will have access to sensitive secrets. More importantly, Terraform facilitates encryption policies for S3 buckets, like default encryption and blocking public access [2]. It works quite well with AWS KMS for encryption at rest with 256-bit keys, also called AES-256, and in-transit encryption with TLS 1.2/1.3 protocols.

Table 1. Comparing Encryption for Data at Rest and In Transit in Cloud Security

Types	AES (Advanced Encryption Standard)	TLS (Transport Layer Security)
Purpose	Encrypt data at rest	Secure data in transit
Type	Symmetric encryption (uses the same key for encryption and decryption).	Asymmetric and symmetric encryption (e.g., public-private key pairs).
Algorithm Used	AES-256 (FIPS 140-2 validated).	TLS 1.2/1.3, often uses RSA or ECDSA certificates for key exchange.
Role in AWS KMS	Encrypts DEKs and data stored in AWS services.	Secures communication between client and AWS services.

Terrascan analyzes security vulnerabilities in the code snippet created with Terraform for insecure configurations and makes sure they get corrected before deployment. Besides this, Terraform Sentinel enforces policy-as-code to automatically reject such configurations that do not hold to the best practices of security. Big organizations fought with S3 configurations that were inconsistent, which caused buckets to get misconfigured and unencrypted, thus exposing sensitive data. By leveraging Terraform, it could define default encryption for all S3 buckets using AWS KMS, automate RBAC and IAM policies to restrict access, and integrate Terrascan for pre-deployment security checks to reduce

human errors greatly. The adoption of these measures brought about a drastic 95% reduction in data breaches and generally improved compliance with industry security standards.

Key recommendations to maximize Terraform's security benefits include codifying all infrastructure policies in order to enforce access control and encryption consistently. Additionally, pre-deployment scanning integration with tools like Terrascan will be instrumental in the early detection of vulnerabilities to ensure that insecure configurations are not allowed to reach a point where they could present a security risk.

As conclusion misconfigurations are still one of the major risks to cloud security. Enforcing access control and encryption policies with tools like Terraform enables scalability, reduces human error, and improves compliance. In such a way, organizations following best practices will reduce not only breaches but also improve the overall security posture.

References:

1. Smith, J., & Brown, P. (2021). Addressing Cloud Misconfigurations: Challenges and Mitigation Strategies. Journal of Cloud Security, 7(3), 245-260.

2. Taylor, M., & Johnson, L. (2019). Infrastructure as Code for Secure Cloud Deployments: A Case Study of Terraform. International Journal of Information Security, 18(4), 289-302.