

Ghaymah Systems

Automated Security Script Documentation & Baseline Mapping

Overview

This document provides a technical breakdown of the "ghaymah-audit.sh" bash script. It explains how each block of code functions and maps directly to the rules defined in the 15-point Ghaymah Cloud Infrastructure Security Baseline.

1. Permissions Check (صلاحيات)

Technical Code Explanation:

The script utilizes the built-in Bash variable "\$EUID" (Effective User ID) to evaluate the execution privileges. In Linux architectures, the root (administrator) user is always assigned an ID of 0. By evaluating the condition ["\$EUID" -eq 0], the script can definitively determine if the environment is running with maximum privileges.

Connection to Ghaymah Baseline:

- **Rule 2.2 (Isolate Workloads - Rootless & Read-Only):** This check directly enforces our container security policy. Containers and workloads deployed on Ghaymah must run as non-root users. Blocking root execution prevents "container escape" vulnerabilities, protecting the underlying host nodes.
- **Rule 1.2 (Just-in-Time Privileges):** It also aligns with the principle of least privilege, ensuring scripts and automation tools do not operate with standing root access.

2. Open Ports Check (منافذ)

Technical Code Explanation:

The script uses network diagnostic commands ("ss -tln" or "netstat -tln") to list all active, listening network ports on the machine without resolving DNS names (for speed). It pipes (|) this output into the "grep -E ':(22)\s'" command. This isolates the output to check specifically for Port 22, which is the default listening port for SSH (Secure Shell).

Connection to Ghaymah Baseline:

- **Rule 3.1 (Zero Trust Micro-segmentation):** By verifying that SSH is not exposed, we enforce our network isolation policies. Management ports should never be publicly exposed; access must be gated through Zero Trust Network Access (ZTNA).
- **Rule 5.3 (Prevent Security Misconfiguration):** Leaving default management ports open is a critical OWASP misconfiguration. This check serves as an automated guardrail against deployment errors.

3. SSL/TLS Certificate Check (SSL)

Technical Code Explanation:

The script chains several tools to validate cryptographic health. It uses "openssl s_client -connect" to ping the domain on port 443 (HTTPS) and download the live SSL certificate. It passes this to "openssl x509 -enddate" to extract the expiration date. Finally, it uses the "date +%s" command to convert both the expiration date and the current date into "Epoch time" (seconds elapsed since January 1, 1970). Comparing these two integers allows the script to accurately determine if the certificate has expired.

Connection to Ghaymah Baseline:

- **Rule 4.1 (Universal KMS Encryption & TLS 1.3):** This maps directly to our data security mandates. Data in transit must be encrypted. An expired certificate breaks the trust chain and compromises the encrypted tunnel, violating our security SLA.
- **Rule 3.3 (Enforce mTLS):** Secure inter-service communication relies on valid certificates. This check ensures that the foundational layer for mutual TLS remains active and trusted.