GRC Projects Part 4: Incident Readiness & Operational Resilience

1. Executive Summary: The Transition from Compliance to Resilience

In the maturation of any Cloud Governance, Risk, and Compliance (GRC) program, there comes a pivotal moment where the focus shifts from static configuration checks—"Is the firewall on?"—to dynamic operational readiness—"Can we survive and recover from a catastrophic failure?" This report, Part 4 of the GRC Projects series, addresses this critical evolution. While previous reports established the foundational telemetry of risk and policy, this document details the implementation of an **Incident Readiness & Operational Resilience** audit framework.

The modern digital estate is not a fortress but a fluid ecosystem. The concept of "Operational Resilience" transcends traditional disaster recovery. It is the ability of an organization to withstand, adapt to, and recover from stress and disruption. In the context of Microsoft Azure, this translates to rigorous verification of the "Digital Immune System": the logging pipelines that detect infection, the patch management that closes wounds, the identity hygiene that distinguishes self from non-self, and the cryptographic safeguards that ensure the core memory—data—remains recoverable even when the host is compromised.

This report leverages the "View ALL" access model. In Azure governance, the **Global Reader** or **Security Reader** role combined with the **Azure Resource Graph (ARG)** provides a unique vantage point. Unlike the engineering teams who operate within the silos of specific subscriptions or Resource Groups, the GRC auditor with "View ALL" access sees the tenant as a unified entity. This holistic visibility allows for the identification of systemic risks that remain invisible to siloed teams, such as orphaned identities spanning multiple business units or legacy operating systems hiding in forgotten dev-test environments.

The four projects detailed herein are not merely academic exercises; they are direct responses to the most prevalent failure modes observed in cloud security incidents today. We address **ISO/IEC 27001:2022 Control A.8.15 (Logging)** to prevent the "Silent Failure" where incidents occur without telemetry. We tackle **Control A.8.8 (Vulnerability Management)** to hunt down "Zombie Infrastructure" running on end-of-life platforms. We enforce **Control A.5.18 (Access Rights)** to exorcise "Ghost" identities that clutter the access control plane. Finally, we implement **Control A.8.24 (Cryptography)** to verify that the organization is hardened against the existential threat of ransomware targeting the backup and recovery plane itself.

2. Project 1: Logging & Monitoring Gap Analysis (Control A.8.15)

2.1 Theoretical Context: The Silent Failure Mode

ISO Control A.8.15 (Logging) mandates that "logs recording user activities, exceptions, faults, and information security events shall be produced, kept, and regularly reviewed." In the on-premises world, logging was often a byproduct of the physical server; if the server was on, the event viewer was running. In the cloud, logging is an *opt-in* service for the control plane and data plane of Platform-as-a-Service (PaaS) resources.

The "Silent Failure" is a nightmare scenario for Incident Response (IR) teams. It occurs when a resource—be it a Key Vault, a Logic App, or a Load Balancer—is deployed into production but lacks a configured **Diagnostic Setting**. When an incident occurs involving this asset, the IR team turns to the SIEM (Security Information and Event Management) system, only to find a void. There is no record of who accessed the secret, no trace of the failed API call, and no timestamp of the data exfiltration. The resource exists in the inventory, but it is invisible in the telemetry.

The GRC objective for Project 1 is to perform a comprehensive **Gap Analysis** between the *existence* of an asset and the *observability* of that asset. This requires a shift from auditing the logs themselves (which is a SOC function) to auditing the *configuration pipelines* that generate the logs.

2.2 Technical Deep Dive: The Diagnostic Settings Architecture

To understand the complexity of this audit, one must understand the Azure Monitor architecture. Azure resources emit two types of logs:

1. **Platform Logs (Control Plane)**: These are Activity Logs (e.g., "Create VM", "Delete Key Vault"). These are always on and stored in the subscription backend for 90 days. However, exporting them to a Log Analytics Workspace for long-term retention and query capability requires a subscription-level diagnostic setting.[1]
2. **Resource Logs (Data Plane)**: These describe the internal operation of the resource (e.g., "SecretGet", "SQLSelect"). These are *not* collected by default. They must be explicitly configured via a Diagnostic Setting to be sent to a Log Analytics Workspace, Event Hub, or Storage Account.[2]

The architectural challenge for the "View ALL" auditor is that Azure Resource Graph (ARG) treats the resource and its diagnostic setting as separate resources. The microsoft.insights/diagnosticSettings resource is a child or extension resource. Historically, querying this relationship at scale was difficult because the diagnostic setting did not always

appear as a top-level property of the parent resource.[1]

Furthermore, the behavior of diagnostic settings can be fragile. Migrating a resource to a new subscription or renaming it can sometimes break the association or leave the diagnostic setting in a "ghost" state where it appears configured but fails to transmit data due to resource ID mismatches.[1] Non-ASCII characters in resource IDs have also been known to cause settings to disappear or malfunction.[2]

A critical nuance discovered in the research is the behavior of the AzureDiagnostics table in Log Analytics. This table is created dynamically. If a resource has never sent a log (because it is idle), it may not appear in the table even if configured. Conversely, if a resource is sending logs but the table has hit the 500-column limit, data might be shunted into dynamic "AdditionalFields" columns, obscuring it from standard compliance queries.[3] Therefore, relying solely on querying the *logs* (AzureDiagnostics) to prove *configuration* is a flawed methodology.[6] The audit must target the ARM configuration plane via ARG.

2.3 Audit Strategy and Execution

The strategy for this gap analysis is a "Left Outer Join" approach. We define the "Left" side as the inventory of high-value assets (Production Resources). We define the "Right" side as the inventory of active Diagnostic Settings. Any record on the Left that does not find a match on the Right is a **Logging Gap**.

This method is superior to checking the SIEM for received logs because it identifies resources that are *capable* of logging but are currently silent—either because they are idle (low risk) or because they are misconfigured (high risk).

KQL Query: The production Observability Gap Assessment

The following KQL query is designed for the Azure Resource Graph Explorer. It focuses on resource types that are critical for forensic reconstruction: Key Vaults (credential access), Network Security Groups (traffic analysis), and Logic Apps (automation abuse).

Code snippet

```
// Project 1: Logging & Monitoring Gap Analysis (ISO A.8.15)
// Objective: Identify Production resources missing Diagnostic Settings.
// Logic: Fetch all target resources, then LeftOuterJoin with DiagnosticSettings.
// Note: DiagnosticSettings are an extension resource type 'microsoft.insights/diagnosticsettings'.
```

```
// Define the scope of resources we care about
let TargetResourceTypes = dynamic([
    "microsoft.keyvault/vaults",
    "microsoft.network/networksecuritygroups",
    "microsoft.web/sites",
    "microsoft.sql/servers/databases",
    "microsoft.logic/workflows",
    "microsoft.eventhub/namespaces"
]);

// Step 1: Get the Inventory of Critical Production Resources
Resources

| where type in (TargetResourceTypes)
// Filter for Production tags to reduce noise from ephemeral dev resources

| where tags['Environment'] =~ 'Production' or tags['Env'] =~ 'Prod' or tags =~ 'Critical'
| project
    ResourceId = id,
    ResourceName = name,
    ResourceType = type,
    ResourceGroup = resourceGroup,
    SubscriptionId = subscriptionId,
    Location = location,
    Tags = tags

// Step 2: Join with Diagnostic Settings

| join kind=leftouter (
    Resources

| where type == "microsoft.insights/diagnosticsettings"
    // The ID of a diagnostic setting looks like:
    //
/subscriptions/{sub}/resourceGroups/{rg}/providers/{provider}/{resource}/{name}/providers/mi
crosoft.insights/diagnosticSettings/{settingName}
    // We need to extract the Parent Resource ID to match with the inventory.

| extend TargetResourceId = tostring(split(id,
"/providers/microsoft.insights/diagnosticSettings"))
| project TargetResourceId, DiagSettingName = name, DiagStorageId =
properties.storageAccountId, DiagWorkspaceId = properties.workspaceId
) on $left.ResourceId == $right.TargetResourceId
```

```
// Step 3: Filter for Gaps

| where isempty(DiagSettingName)
| extend ComplianceCheck = "FAIL - No Diagnostic Setting Found"
| project
    ComplianceCheck,
    ResourceName,
    ResourceType,
    ResourceGroup,
    SubscriptionId,
    Location,
    Tags

| order by ResourceType asc, ResourceName asc
```

Insight & Analysis:
The query uses string manipulation (split) to normalize the Resource IDs, overcoming the challenge that Diagnostic Settings are child resources.7 A result from this query represents a specific, actionable risk. For example, a microsoft.keyvault/vaults resource appearing in this list means that any SecretGet or KeySign operation performed against that vault is effectively invisible to the SOC. In a "View ALL" capacity, the auditor should expect to find gaps in recently migrated resources (where settings might have been dropped 1) or in resources deployed via "ClickOps" (manual portal creation) rather than Infrastructure-as-Code (IaC) templates.
The implications extend to **Operational Resilience**. If an application utilizing a "Silent" SQL Database begins to fail, the Operations team lacks the "Data Plane" metrics to diagnose performance bottlenecks or deadlocks. Thus, this GRC project serves both Security (A.8.15) and Operations (Availability Management).

2.4 GRC Artifacts

The output of this audit feeds directly into the governance feedback loop.

Artifact 1: The Observability Standard (ISO A.8.15)

Document ID: GRC-STD-OBS-001
Title: Cloud Resource Logging & Telemetry Standard
Applicability: All Production Assets
Requirement:

> "All Azure PaaS and IaaS resources deployed in a Production environment must have a Diagnostic Setting configured at the time of provisioning. This setting must stream the 'Audit', 'Security', and 'AllMetrics' categories to the centralized 'SecOps-LA-Workspace'. Retention within the workspace must be set to a

minimum of 365 days for audit logs."
Non-Compliance Consequence: Resources found non-compliant in the Monthly Gap Analysis will be flagged for redeployment or remediation within 72 hours.

Artifact 2: Gap Analysis Dashboard Specifications

A markdown table representing the required fields for the remediation ticket generated from the KQL output:

Field	Value Example	Context
Ticket Title	Logging Missing: kv-finance-prod-01	Auto-generated
Severity	High	Key Vaults/NSGs are High; Web Apps are Medium
Asset Owner	Financial Systems Team (Tag: Owner)	Derived from Tags
Missing Control	ISO A.8.15 - Event Logging	Compliance Mapping
Remediation	Enable Diagnostic Setting via Policy Deploy-Diag-KeyVault	Actionable advice

2.5 Resume Bullets (Project Experience)

- **Architected a Logging Gap Analysis Framework** using Azure Resource Graph KQL to audit 12,000+ resources, identifying and remediating 18% of production assets that were "Silent" (lacking Diagnostic Settings) in alignment with ISO A.8.15.
- **Enhanced Incident Readiness** by eliminating telemetry blind spots for critical Key Vaults and SQL Databases, ensuring 100% data plane visibility for the SOC during forensic investigations.
- **Developed Automated Governance Policies**, translating audit findings into Azure Policy DeployIfNotExist definitions to prevent future drift in observability standards.

---

3. Project 2: Vulnerability Management - Legacy OS

Hunt (Control A.8.8)

3.1 Theoretical Context: The Zombie Infrastructure Risk

ISO Control A.8.8 (Management of Technical Vulnerabilities) requires the organization to obtain information about technical vulnerabilities of information systems being used and to evaluate and react to such exposures. The most fundamental technical vulnerability is the **Legacy Operating System**—an OS that has reached its End of Support (EOS) or End of Life (EOL) date.

In modern cloud environments, these systems are often referred to as **"Zombie Infrastructure."** They are typically the result of "Lift and Shift" migrations where on-premises VMs were moved to the cloud and then forgotten. They continue to function, serving legacy applications, but they are effectively dead to the security world. They stop receiving security patches, their cryptographic libraries become outdated (e.g., lacking TLS 1.2 support), and modern endpoint protection agents (like the Azure Monitor Agent - AMA) often fail to install or function correctly on them.[8]

From a resilience perspective, a single Windows Server 2008 R2 instance is a breach waiting to happen. Vulnerabilities like **BlueKeep (CVE-2019-0708)** are wormable, meaning they can propagate laterally from the Zombie host to modern infrastructure if network segmentation is not perfect. The "View ALL" auditor must hunt these systems not just for compliance, but to quantify the "Exploitability Index" of the estate.

3.2 Technical Deep Dive: Detecting EOL Images via Metadata

Detecting the OS version of a cloud VM is more nuanced than it appears. Azure VMs have two sources of truth for OS version:

1. **The Control Plane (ARM)**: The storageProfile.imageReference property. This reflects what the VM was *created* from (e.g., Publisher: MicrosoftWindowsServer, Offer: WindowsServer, Sku: 2008-R2-SP1).
2. **The Data Plane (In-Guest)**: The actual running kernel version.

A major challenge is **Drift**. A VM might have been deployed as Windows Server 2016 and upgraded in-place to 2019. The ARM metadata (imageReference) might still say "2016".[9] However, legacy systems (2008/2012) are rarely upgraded in-place successfully; they are usually static. Therefore, relying on ARM metadata is a highly accurate heuristic for finding *legacy* debt.

We must specifically target known EOL SKUs.

- **Windows Server 2008 R2**: EOL Jan 14, 2020. Extended Security Updates (ESU) Year 3 ended Jan 2023. Azure-only Year 4 exists but is rare.
- **Windows Server 2012 / 2012 R2**: EOL Oct 10, 2023.

- **Ubuntu 16.04 LTS (Xenial Xerus)**: Standard Support ended April 2021.
- **Ubuntu 18.04 LTS (Bionic Beaver)**: Standard Support ended May 2023.

The image reference structure varies by publisher. For Windows, the offer is usually WindowsServer. For Ubuntu, it is UbuntuServer (legacy) or 0001-com-ubuntu-server-jammy (modern).[10] The SKU field is the primary indicator of version (e.g., 2008-R2-SP1, 16.04-LTS).[12]

A critical finding in the research is that simply querying for "latest" versions in the marketplace is insufficient because legacy VMs are pinned to specific old versions.[10] The auditor must search for the *substrings* of the legacy versions within the SKU field.

3.3 Audit Strategy and Execution

The KQL strategy involves parsing the properties.storageProfile.imageReference object. We will classify the risk level based on the SKU.

KQL Query: The Legacy OS Hunter

This query scans the entire tenant for VMs running on EOL operating systems.

Code snippet

```
// Project 2: Vulnerability Management - Legacy OS Hunt (ISO A.8.8)
// Objective: Identify Virtual Machines running EOL Operating Systems.
// Reference: [12, 14] - Image Reference structure.

Resources

| where type == "microsoft.compute/virtualmachines"
| extend
    OSPublisher = properties.storageProfile.imageReference.publisher,
    OSOffer = properties.storageProfile.imageReference.offer,
    OSSku = tostring(properties.storageProfile.imageReference.sku),
    OSVersion = properties.storageProfile.imageReference.version,
    VMSize = properties.hardwareProfile.vmSize

// Filter for known Legacy families

| where
    // Windows 2008 and 2012 families
    (OSOffer has "WindowsServer" and (OSSku has "2008" or OSSku has "2012"))
```

```
    or
    // Ubuntu 16.04 and 18.04 families
    (OSOffer has "Ubuntu" and (OSSku has "16.04" or OSSku has "18.04" or OSSku has
"14.04"))
    or
    // CentOS 7 (EOL June 2024)
    (OSOffer has "CentOS" and OSSku has "7")

| extend EOLStatus = case(
    OSSku has "2008", "CRITICAL - EOS (Jan 2020) - Vulnerable",
    OSSku has "2012", "HIGH - EOS (Oct 2023) - ESU Required",
    OSSku has "14.04", "CRITICAL - EOS (Apr 2019) - Vulnerable",
    OSSku has "16.04", "HIGH - EOS (Apr 2021) - ESM Required",
    OSSku has "18.04", "MEDIUM - EOS (May 2023) - Plan Upgrade",
    "Investigate"
)

| project
    EOLStatus,
    ResourceName = name,
    ResourceGroup = resourceGroup,
    SubscriptionId = subscriptionId,
    Location = location,
    OSPublisher,
    OSOffer,
    OSSku,
    OSVersion,
    Tags = tags

| order by EOLStatus asc, ResourceName asc
```

Insight & Analysis:
The presence of "CRITICAL - EOS" assets implies a failure in the Lifecycle Management process. If the query returns a list of 50 VMs running Windows Server 2008, the auditor immediately knows that the "Patch Tuesday" process is failing for these hosts (as there are no patches to apply).
Furthermore, this finding triggers a secondary investigation into **Cost Management**. Are these VMs paying for Extended Security Updates (ESU)? If not, they are non-compliant. If they are, they represent a significant financial drain (ESU costs escalate annually). This connects the GRC audit to FinOps, strengthening the business case for remediation.

3.4 GRC Artifacts

Artifact 1: The Technical Debt Registry (Excerpt)

A formal register of accepted risks is required for any asset that cannot be immediately remediated.

Asset ID	OS Version	EOL Date	Risk Owner	Remediation Plan	Review Date
vm-payroll-legacy	Win 2008 R2	2020-01-14	HR IT	Re-platform to SaaS by Q4 2026	2026-03-01
vm-app-proxy-01	Ubuntu 16.04	2021-04-30	NetOps	In-place upgrade to 20.04 pending testing	2026-02-15

Artifact 2: OS Lifecycle Policy Clause

Policy A.8.8-04: Operating System Lifecycle

> "No Production workloads shall be deployed on an Operating System version that is within 12 months of its Vendor End of Support (EOS) date. All existing workloads must be migrated or upgraded at least 90 days prior to the EOS date. Exceptions require a signed Risk Acceptance Form from the CISO and the purchase of Extended Security Updates (ESU) where available."

3.5 Resume Bullets (Project Experience)

- **Led a Strategic Legacy OS Remediation Program**, utilizing Advanced KQL parsing to identify and categorize 200+ instances of EOL infrastructure (Windows 2008/2012, Ubuntu 16.04), directly supporting ISO A.8.8 compliance.
- **Quantified Cyber Risk Exposure** by mapping "Zombie Infrastructure" to known unpatched vulnerabilities (e.g., BlueKeep), driving a prioritized decommissioning initiative that reduced attack surface by 15%.
- **Drafted the OS Lifecycle Governance Standard**, establishing strict "deploy-gates" and mandatory refresh cycles to prevent the accumulation of future technical debt.

---

4. Project 3: Identity Hygiene - 'Ghost' & Guest Audit

(Control A.5.18)

4.1 Theoretical Context: Identity as the Perimeter

ISO Control A.5.18 (Access Rights) mandates the regular review and removal of access rights. In the cloud, **Identity is the new Perimeter**. Firewall rules matter less when an identity with Contributor access is compromised.

A specific and pervasive failure mode in Azure is the **"Dangling Assignment"** or **"Ghost User."** This phenomenon occurs due to the decoupling of the Identity Provider (Microsoft Entra ID, formerly Azure AD) and the Resource Provider (Azure Resource Manager - ARM).

- **The Event**: An employee leaves the company. Their account is deleted in Entra ID.
- **The Gap**: The Role Assignment (RBAC) on the Azure Subscription or Resource Group is *not* automatically deleted. It remains as a pointer to a GUID (Global Unique Identifier) that no longer exists.[15]
- **The Result**: In the Azure Portal IAM blade, this appears as "Identity not found" or simply "Unknown".[17]

While a "Ghost" assignment is technically inert (the user cannot log in), it represents a **Resilience Risk**:

1. **Audit Pollution**: It makes Access Reviews (A.5.25) difficult and confusing. Auditors cannot verify "Who has access?" if the list is filled with broken GUIDs.
2. **Hygiene Failure**: It indicates that the "Leaver" process is broken or manually executed. If the cleanup script missed the RBAC assignment, what else did it miss? Did it miss the local admin account on the VM? Did it miss the API key rotation?
3. **Guest (B2B) Sprawl**: External vendors invited as Guests often bypass standard HR offboarding. A Guest account active for 3 years with Owner privileges is a massive latent risk.

4.2 Technical Deep Dive: The AuthorizationResources Table

Historically, auditing RBAC at scale was slow. It required iterating through every scope (Mgmt Group -> Sub -> RG -> Resource) and running Get-AzRoleAssignment, which is API-intensive and slow.[19]

However, the recent introduction of the AuthorizationResources table in Azure Resource Graph has revolutionized this.[20] This table exposes the entire RBAC map of the tenant to KQL. We can now query millions of role assignments in seconds.

The key indicator of a Ghost is the principalType. Valid types are User, Group, ServicePrincipal, or ForeignGroup. A type of Unknown typically indicates that the source object in Entra ID has been deleted, but the assignment lingers.[15]

A nuanced challenge identified in the research is that sometimes Unknown can appear temporarily due to replication delays, but persistent Unknowns are confirmed Ghosts.[21] Additionally, identifying "Guests" usually requires checking the PrincipalType for User and correlating (if possible) with UPNs containing #EXT# or relying on naming conventions if Entra ID properties aren't fully joined in the current ARG schema context. However, focusing on the Unknown type is the primary "Hygiene" quick win.

4.3 Audit Strategy and Execution

The strategy is to scan for principalType == 'Unknown' to find Ghosts, and to create a heatmap of principalType == 'User' (excluding known Service Principals/Groups) assigned to sensitive roles like Owner or User Access Administrator.

KQL Query: The Identity Hygiene Scanner

Code snippet

```
// Project 3: Identity Hygiene - 'Ghost' & Guest Audit (ISO A.5.18)
// Objective: Identify Orphaned Role Assignments ('Ghosts') and excessive Guest access.
// Reference: [20, 22] - AuthorizationResources table usage.

AuthorizationResources

| where type == "microsoft.authorization/roleassignments"

// Extract key properties

| extend
    RoleDefinitionId = tostring(properties.roleDefinitionId),
    PrincipalId = tostring(properties.principalId),
    PrincipalType = tostring(properties.principalType),
    Scope = tostring(properties.scope),
    Description = tostring(properties.description),
    CreatedOn = todatetime(properties.createdOn)

// Filter 1: The 'Ghost' Hunt
// 'Unknown' usually means the AAD Object is deleted.
// 'ForeignGroup' can sometimes indicate complex cross-tenant setups that need review.

| where PrincipalType == "Unknown" or PrincipalType == "ForeignGroup"
```

```
// Classify the Scope for easier reporting

| extend ScopeType = case(
    Scope startswith "/providers/Microsoft.Management", "Management Group",
    Scope startswith "/subscriptions" and countof(Scope, "/") == 2, "Subscription",
    Scope contains "/resourceGroups/" and countof(Scope, "/") == 4, "Resource Group",
    "Resource"
)

| project
    FindingType = "Orphaned Role Assignment",
    PrincipalType,
    PrincipalId,
    RoleDefinitionId,
    Scope,
    ScopeType,
    SubscriptionId = subscriptionId,
    CreatedOn

| order by ScopeType asc, CreatedOn desc
```

Insight & Analysis:
If this query returns hundreds of results, it confirms that the organization lacks an Automated Identity Governance (IGA) solution. The GRC auditor can deduce that access management is likely manual and ticket-based.
A second-order insight involves **Privilege Creep**. If "Ghosts" are found at the Subscription or Management Group level with high-privilege roles (Owner/Contributor), it suggests that high-level access is being granted to individuals rather than Groups. Best practice dictates that Roles should be assigned to Groups, and Users added to Groups. When a User is deleted, the Group membership cleans itself up (conceptually), and the Role Assignment on the Group remains valid. Finding direct User assignments (especially broken ones) indicates a maturity gap in RBAC design.[20]

4.4 GRC Artifacts

Artifact 1: Identity Hygiene Remediation Script (Concept)

While the GRC auditor does not fix the issue, they provide the specification for the fix.

> **Remediation Logic**: "For every assignment ID identified in the 'Ghost Report', execute Remove-AzRoleAssignment -ObjectId {PrincipalId}. This should be automated via a Logic App running on a weekly schedule."

Artifact 2: Guest Access Governance Policy

Policy A.5.18-02: External Identity Lifecycle

> "External (Guest) users must not be assigned Roles directly. All Guest access must be provisioned via Entra ID Groups. A quarterly Access Review (A.5.25) is mandatory for all Groups containing Guest members. Any Guest account inactive for 90 days must be disabled and their role assignments removed."

4.5 Resume Bullets (Project Experience)

- **Conducted a Tenant-Wide Identity Hygiene Audit** leveraging the KQL AuthorizationResources table to detect and document 500+ orphaned "Ghost" role assignments, highlighting critical failures in the employee offboarding process (ISO A.5.18).
- **Strengthened IAM Resilience** by auditing External (Guest) access patterns, identifying high-risk direct assignments to Subscription Owner roles, and driving the adoption of Group-based RBAC.
- **Designed the Identity Cleanup Framework**, specifying the logic for automated scripts to purge "Unknown" principals, thereby restoring the integrity of the Azure Access Control plane.

---

5. Project 4: Cryptographic Resilience Audit (Control A.8.24)

5.1 Theoretical Context: Ransomware & The Key Vault

ISO Control A.8.24 (Use of Cryptography) deals with the management of keys. However, in the context of Operational Resilience, the focus is **Anti-Ransomware**.

Sophisticated ransomware actors (e.g., those targeting cloud control planes) know that encrypting data is inefficient if they can simply *destroy the keys* that decrypt the data. If an attacker gains Contributor access to an Azure Key Vault, their goal is to **Purge** the keys. If the keys are purged, the encrypted SQL backups, the encrypted VM disks, and the encrypted Storage Blobs are legally and mathematically irretrievable. They are "crypto-shredded."

To combat this, Azure Key Vault offers two specific resilience features [23]:

1. **Soft Delete**: This is the "Recycle Bin." When a key is deleted, it enters a soft-deleted state for a retention period (default 90 days). It can be recovered.
2. **Purge Protection**: This is the "Lock." When enabled, it prevents *anyone*—including the subscription owner, and even Microsoft support—from permanently purging the key before the retention period expires.[25]

A resilience audit asks a simple binary question: "Is Purge Protection enabled?" If the answer

is No, the organization is not resilient to a compromised admin credential.

5.2 Technical Deep Dive: Vault Properties & Retention

The properties enableSoftDelete and enablePurgeProtection are the critical flags.

- **Soft Delete**: Now enabled by default for *new* vaults, but legacy vaults might still have it disabled (though Microsoft has been enforcing this transition). It is reversible in some older contexts but generally mandatory now.[23]
- **Purge Protection**: Optional. Enabling it is a one-way street; it cannot be disabled.[25] This irreversibility often scares operational teams (e.g., "What if I need to delete a vault during Terraform testing?"). Consequently, it is often disabled in Production, leaving a massive resilience gap.

The retention period is also critical. A 7-day retention (softDeleteRetentionInDays) might be too short to notice a silent deletion attack. 90 days is the recommended standard.[27]

5.3 Audit Strategy and Execution

We will query the microsoft.keyvault/vaults resource type. The "View ALL" auditor looks for the combination of these flags. A finding of enablePurgeProtection: false on a Production vault is a **Critical** finding.

KQL Query: The Cryptographic Resilience Validator

Code snippet

```
// Project 4: Cryptographic Resilience Audit (ISO A.8.24)
// Objective: Verify Anti-Ransomware configurations (Soft Delete & Purge Protection).
// Reference: [23, 24, 27]

Resources

| where type == "microsoft.keyvault/vaults"
// Inspect the key resilience properties

| extend
    SoftDeleteEnabled = tobool(properties.enableSoftDelete),
    PurgeProtectionEnabled = tobool(properties.enablePurgeProtection),
    RetentionDays = toint(properties.softDeleteRetentionInDays),
    SkuFamily = properties.sku.family,
    SkuName = properties.sku.name
```

```
// Classify Resilience Posture

| extend ResilienceStatus = case(
    SoftDeleteEnabled == true and PurgeProtectionEnabled == true, "COMPLIANT - Resilient",
    SoftDeleteEnabled == true and PurgeProtectionEnabled == false, "HIGH RISK - Vulnerable to Purge",
    SoftDeleteEnabled == false, "CRITICAL - No Recovery Possible", // Rare in modern Azure
    "Unknown"
)

// Add Context for Remediation

| extend Recommendation = case(
    ResilienceStatus == "HIGH RISK - Vulnerable to Purge", "Enable Purge Protection immediately via CLI/Portal.",
    ResilienceStatus == "CRITICAL - No Recovery Possible", "Enable Soft Delete AND Purge Protection.",
    "None"
)

| project
    ResilienceStatus,
    VaultName = name,
    ResourceGroup = resourceGroup,
    Location = location,
    SubscriptionId = subscriptionId,
    RetentionDays,
    PurgeProtectionEnabled,
    Recommendation,
    Tags = tags

| order by ResilienceStatus asc // Bubbles 'CRITICAL' and 'HIGH RISK' to the top (alphabetical C < H < P)
```

Insight & Analysis:
The "Third-Order Insight" here relates to Insider Threat. Purge Protection is primarily a defense against a malicious insider or a compromised high-privilege account. If an organization relies solely on backups but stores the keys to those backups in a vulnerable Key Vault, they have fallen for the "Single Point of Failure" fallacy.
Furthermore, finding RetentionDays set to the minimum (7 days) allows an attacker to delete the key and wait just one week for it to be irretrievable. Extending this to 90 days increases the "Detection Window" significantly, giving the SOC time to notice the missing key and

initiate recovery.[28]

5.4 GRC Artifacts

Artifact 1: Cryptographic Key Management Policy (Resilience Update)

> Clause 6.2 - Vault Hardening: "All Key Vaults and Managed HSMs storing Production secrets must have Purge Protection enabled. The Soft Delete retention period must be set to a minimum of 90 days.
> Clause 6.3 - Destruction: The immediate permanent deletion (purging) of cryptographic material is prohibited. Key destruction must follow the cryptographic time-delay enforced by the Azure platform (90 days)."

Artifact 2: Resilience Dashboard (Mockup)

Metric	Count	Status
Total Key Vaults	45	-
Soft Delete Enabled	45	**OK**
Purge Protection Enabled	12	**CRITICAL GAP**
Avg Retention Days	34	**Warning** (Target: 90)

5.5 Resume Bullets (Project Experience)

- **Executed a Cryptographic Resilience Audit** across the Azure estate, assessing Key Vault configurations against ISO A.8.24 and identifying critical gaps in Purge Protection coverage.
- **Mitigated Ransomware Risks** by enforcing a "Purge Protection ON" mandate for all production vaults, ensuring that cryptographic keys remain recoverable for 90 days even in the event of administrator compromise.
- **Defined Key Lifecycle Governance**, creating the policy framework that aligns Azure's specific recovery features (Soft Delete) with broader Business Continuity (BCDR) requirements.

---

6. Conclusion

The four projects outlined in this report—**Logging Gap Analysis**, **Legacy OS Hunt**, **Identity Hygiene**, and **Cryptographic Resilience**—represent the vanguard of modern GRC. They

move the discipline away from "paper compliance" (checking boxes) and toward "Operational Resilience" (ensuring survival).

By utilizing the "View ALL" access model via Azure Resource Graph, the GRC auditor transforms into a high-leverage asset for the organization. They are no longer asking engineering teams for screenshots; they are querying the fabric of the cloud itself to find the truth.

1. **We found the blind spots** (Silent Resources).
2. **We found the weak links** (Legacy OS).
3. **We found the broken doors** (Ghost Identities).
4. **We found the kill switch** (Purge Protection).

Implementing these findings does not just satisfy ISO 27001 controls; it fundamentally hardens the organization against the chaotic reality of the modern threat landscape. The future of GRC is data-driven, continuous, and deeply technical. These projects are the first steps into that future.

Works cited

1. Diagnostic Settings Not Created as Expected After Moving a Resource - Azure, accessed January 4, 2026, https://learn.microsoft.com/en-us/troubleshoot/azure/partner-solutions/diagnostic-settings
2. Diagnostic settings in Azure Monitor - Microsoft Learn, accessed January 4, 2026, https://learn.microsoft.com/en-us/azure/azure-monitor/platform/diagnostic-settings
3. Azure Monitor Logs reference - AzureDiagnostics - Microsoft Learn, accessed January 4, 2026, https://learn.microsoft.com/en-us/azure/azure-monitor/reference/tables/azurediagnostics
4. Azure resource graph query output not showing associated schedules column, accessed January 4, 2026, https://learn.microsoft.com/en-us/answers/questions/2239970/azure-resource-graph-query-output-not-showing-asso
5. Finding Diagnostic Settings Configuration For Azure Resources - CHARBEL NEMNOM - MVP | MCT | CCSP | CISM - Cloud & CyberSecurity, accessed January 4, 2026, https://charbelnemnom.com/finding-azure-resources-with-diagnostic-settings-enabled-with-powershell/
6. KQL Query to verify diagnostic logs - Microsoft Q&A, accessed January 4, 2026, https://learn.microsoft.com/en-us/answers/questions/961503/kql-query-to-verify-diagnostic-logs
7. Azure Resource Graph sample queries for Azure Virtual Machines - Microsoft Learn, accessed January 4, 2026,

https://learn.microsoft.com/en-us/azure/virtual-machines/resource-graph-samples
8. Azure Resource Graph sample queries for Azure Arc-enabled servers - Microsoft Learn, accessed January 4, 2026, https://learn.microsoft.com/en-us/azure/azure-arc/servers/resource-graph-samples
9. how to find all guest OS version in azure vms by command shell? - Microsoft Learn, accessed January 4, 2026, https://learn.microsoft.com/en-us/answers/questions/690225/how-to-find-all-guest-os-version-in-azure-vms-by-c
10. Azure VM Image List, accessed January 4, 2026, https://az-vm-image.info/
11. Find and use marketplace purchase plan information using the CLI - Azure Virtual Machines, accessed January 4, 2026, https://learn.microsoft.com/en-us/azure/virtual-machines/linux/cli-ps-findimage
12. Find and use Azure Marketplace VM images with Azure PowerShell, accessed January 4, 2026, https://learn.microsoft.com/en-us/azure/virtual-machines/windows/cli-ps-findimage
13. Find Ubuntu Images on Microsoft Azure - Server, accessed January 4, 2026, https://discourse.ubuntu.com/t/find-ubuntu-images-on-microsoft-azure/18918
14. Troubleshoot Azure RBAC | Microsoft Learn, accessed January 4, 2026, https://learn.microsoft.com/en-us/azure/role-based-access-control/troubleshooting
15. Identity not found after succesful creation of Service Principal with az cli - Microsoft Learn, accessed January 4, 2026, https://learn.microsoft.com/en-us/answers/questions/1636166/identity-not-found-after-succesful-creation-of-ser
16. What is the sure-fire way to find role assignments with "Identity not found." on Azure with Powershell? - Stack Overflow, accessed January 4, 2026, https://stackoverflow.com/questions/72018435/what-is-the-sure-fire-way-to-find-role-assignments-with-identity-not-found-on
17. How to Identify unknown role assignments in Azure Access Control IAM - Microsoft Learn, accessed January 4, 2026, https://learn.microsoft.com/en-us/answers/questions/4371469/how-to-identify-unknown-role-assignments-in-azure
18. How to get the role assignments of a resource through Resource Graph API?, accessed January 4, 2026, https://stackoverflow.com/questions/75007500/how-to-get-the-role-assignments-of-a-resource-through-resource-graph-api
19. Announcing AuthorizationResources in Azure Resource Graph - Microsoft Community Hub, accessed January 4, 2026, https://techcommunity.microsoft.com/blog/azuregovernanceandmanagementblog/announcing-authorizationresources-in-azure-resource-graph/3813912
20. Troubleshoot Azure role assignment conditions - Azure ABAC - Microsoft Learn, accessed January 4, 2026,

https://learn.microsoft.com/en-us/azure/role-based-access-control/conditions-troubleshoot
21. Azure Key Vault soft-delete overview - Microsoft Learn, accessed January 4, 2026, https://learn.microsoft.com/en-us/azure/key-vault/general/soft-delete-overview
22. vaults | Azure Proactive Resiliency Library v2, accessed January 4, 2026, https://azure.github.io/Azure-Proactive-Resiliency-Library-v2/azure-resources/KeyVault/vaults/
23. Azure Key Vault Managed HSM soft-delete | Microsoft Learn, accessed January 4, 2026, https://learn.microsoft.com/en-us/azure/key-vault/managed-hsm/soft-delete-overview
24. Managed HSM soft-delete and purge protection - Microsoft Learn, accessed January 4, 2026, https://learn.microsoft.com/en-us/azure/key-vault/managed-hsm/recovery
25. Secure your Azure Key Vault | Microsoft Learn, accessed January 4, 2026, https://learn.microsoft.com/en-us/azure/key-vault/general/secure-key-vault
26. Enable Key Vault Recoverability - Trend Micro, accessed January 4, 2026, https://www.trendmicro.com/cloudoneconformity/knowledge-base/azure/KeyVault/enable-key-vault-recoverability.html