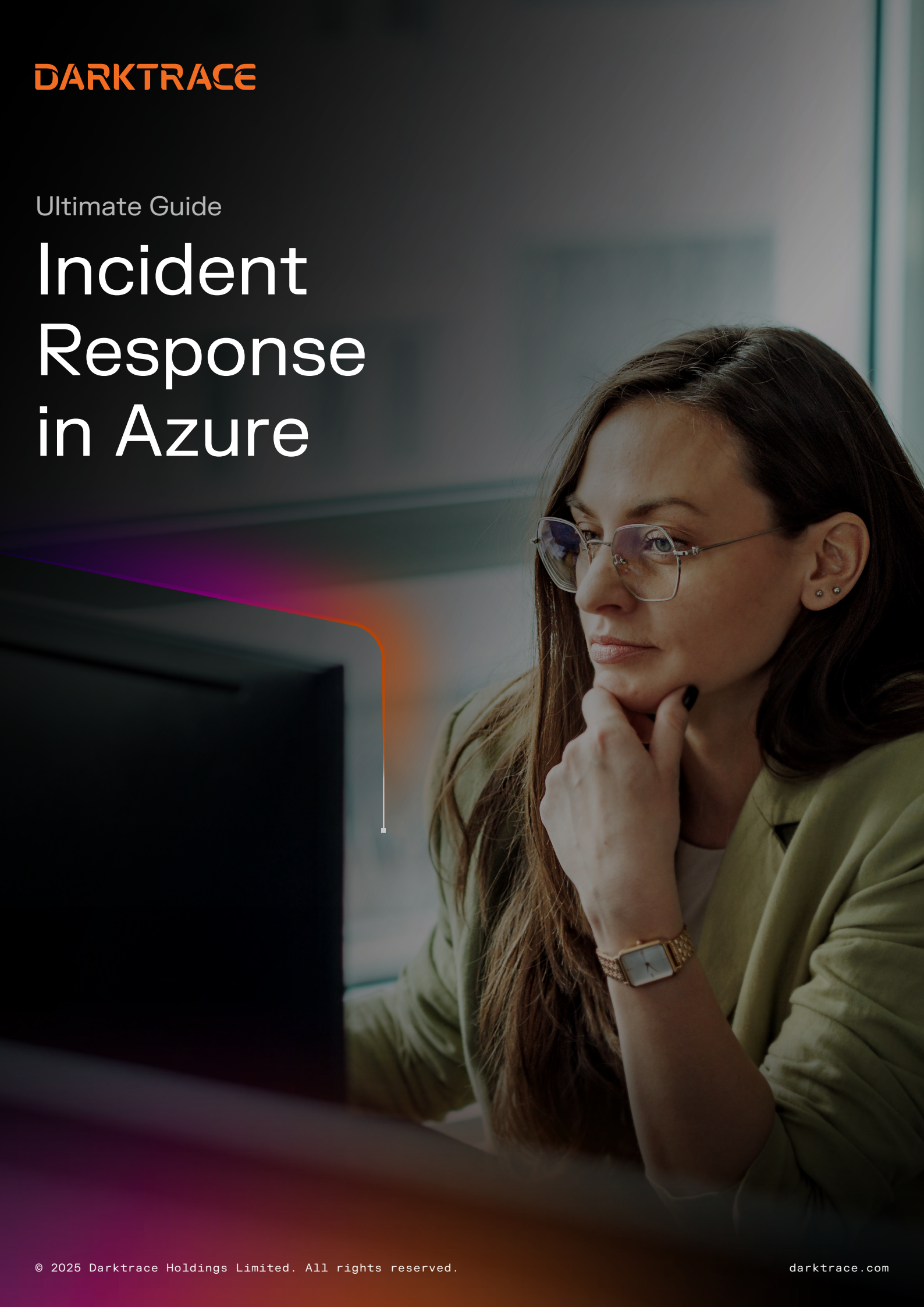
DARKTRACE

Ultimate Guide

Incident Response in Azure



Abstract

Investigating and responding to incidents in cloud environments like Azure is fundamentally different to on-premise. Further, without the right tools and processes in place, it can be more complicated. There are over 200 products and services in Azure, each with different security best practices and data sources. While the cloud can make incident response more complex, it also enables some fantastic possibilities. For example, by leveraging cloud resources to collect, process, and store evidence, you can expedite the end-to-end incident response process in ways that would be unthinkable on-premise.

Azure's incident response advice mentions two critical components to consider when measuring how well your organization is prepared to reduce risk: Mean Time To Acknowledge (MTTA) and Mean Time To Respond (MTTR). The best practices outlined in this playbook were crafted with these two key metrics in mind with the goal of yielding noticeable improvement in both.

This guide aims to provide a comprehensive overview of incident response in Azure, equipping security teams with the knowledge and tools necessary to effectively handle security incidents in the cloud.

Contents

Key Azure log sources

Incident response in Microsoft Azure relies heavily on the ability to collect, analyze, and act on log data. Azure provides multiple logging services, each serving different purposes, from tracking administrative changes to monitoring network activity. Understanding these log sources and their capabilities is essential for effective detection, investigation, and response to security incidents. Below are the key Azure log sources that security teams should prioritize:

Azure activity logs

Azure activity logs provide a record of subscription-level administrative actions, such as creating Virtual Machines (VMs), modifying network configurations, or changing security settings. These logs are essential for auditing user actions and monitoring unauthorized changes in an Azure environment.

Best practices include:

- **Enabling** activity logs across all subscriptions.
- **Forwarding** logs to Azure Monitor Logs for centralized
- analysis.

  Configuring alerts for high-risk administrative actions, such as role assignments or resource deletions.

Azure Resource Logs

Azure Resource Logs (formerly known as Diagnostic Logs) capture data plane operations for specific Azure services, such as VMs, Key Vaults, and Storage Accounts. These logs provide insights into how services are used and help detect anomalous activity.

Recommendations:

- **Enable** Resource Logs for all critical Azure resources.
- **Store** logs in Azure Log Analytics for correlation and threat analysis.
- **Use** Microsoft Sentinel to detect suspicious patterns in log data.

Azure Active Directory (Azure AD) Logs

Azure AD Logs record authentication events, identity changes, and security-related activities. These logs are crucial for detecting unauthorized access attempts, password spraying, and other identitybased threats.

Key log types include:

- **Sign-in logs** - track user and service authentication attempts.
- **Audit logs** - records configuration changes and administrative actions.
- **Risky sign-ins** - identify suspicious login patterns, such as impossible travel or brute-force attempts.

Best practices include:

- **Enabling** conditional access policies to enforce security baselines.
- **Using** Azure AD identity protection to automate threat detection.
- **Forwarding** logs to Microsoft Sentinel for real-time monitoring.

Azure Monitor & Virtual Machine logs

Azure Monitor Logs provide centralized visibility across applications, infrastructure, and network activity. They aggregate logs from various Azure services, including VMs, to help security teams analyze potential threats.

For Windows and Linux VMs:

- **Use** Azure Diagnostics Extension to collect system logs.
- **Configure** Log Analytics Agent to send logs to Azure Monitor.
- **Analyze** event logs to detect signs of system compromise, unauthorized access, or malware execution.

Application insights & application logs

- **Application insights** - monitor the performance and health of web applications hosted in Azure. It captures telemetry data, including request rates, response times, and custom logs.
- **Application logs** - capture internal application errors, warnings, and debugging information. These logs help security teams detect application-layer attacks, such as SQL injection or API abuse.

Best practices include:

- **Enabling** application insights for all mission-critical applications.
- **Correlating** logs with Azure Security Center alerts for enhanced threat detection.
- **Implementing** Web Application Firewall (WAF) Logs to detect and block malicious HTTP requests.

Azure Storage Analytics Logs

Azure Storage Analytics Logs provide detailed insights into storage account activity, including access requests, authentication attempts, and data transfers.

Use cases:

- **Detecting** unauthorized access to Azure Blob Storage.
- **Monitoring** SAS Token usage to prevent excessive data exfiltration.
- **Tracking** failed access attempts to identify potential brute-force attacks.

Best practices include:

- **Enabling** logging for all storage accounts containing sensitive data.
- **Analyzing** logs for unusual download patterns or high request volumes.
- **Storing** logs in Azure Log Analytics for correlation with other security events.

Azure Network Security Group (NSG) Flow Logs

Azure NSG Flow Logs provide network-level visibility by capturing accepted and denied traffic within Virtual Networks (VNets). These logs help detect unauthorized traffic flows, lateral movement, and potential data exfiltration attempts.

Best practices include:

- **Enabling** NSG Flow Logs in all critical subnets.
- **Forwarding** logs to Azure Log Analytics for deeper analysis.
- **Using** Microsoft Sentinel to detect anomalous network behavior.

Microsoft Defender for cloud security alerts

Microsoft Defender for Cloud provides real-time security alerts for potential threats in an Azure environment. These alerts include detections from various security tools, such as:

- **Azure Defender for Servers** for VM-based threats.
- **Azure Defender for Storage** for data exfiltration monitoring.
- **Azure Defender for Key Vault** for unauthorized access detection.

Best practices include:

- **Integrating** Defender for Cloud with Microsoft Sentinel for automated incident response.
- **Enabling** Just-in-Time (JIT) VM Access to reduce exposure to brute-force attacks.
- **Automating** remediation using Azure Logic apps and playbooks.

Other Azure logging sources

Azure provides additional security-focused logging capabilities, including:

- **Azure WAF Logs** - detect and blocks web-based attacks.
- **Azure Firewall Logs** - capture network traffic logs for firewall rules.
- **Azure SQL Auditing Logs** - track database access and potential SQL injection attempts.

Before the incident: proactive security measures

The following best practices can help security teams reduce the likelihood that an incident will occur, and in the event that it does, drastically decrease recovery time.

Understand and protect critical data

Identify and classify sensitive data, such as Personally Identifiable Information (PII) and Payment Card Industry (PCI) data, using Azure Information Protection (AIP). Implement Role-Based Access Control (RBAC) and enforce least privilege for sensitive data access.

More information:

Azure Information Protection documentation →

Backup and recovery planning

Enable Azure Backup for critical workloads and store backups in immutable storage to prevent tampering. Regularly test restores to ensure reliability. Implement Geo-Redundant Storage (GRS) for disaster recovery.

More information:

Azure Backup documentation →

Secure administrative access

Enforce JIT access with Azure Privileged Identity Management (PIM), disable legacy authentication, and require Multi-Factor Authentication (MFA) for all accounts. Monitor risky sign-ins and audit privileged actions.

More information:

Best practices for Entra ID roles →

Network and remote access controls

Restrict exposure of Azure VMs by enabling Network Security Groups (NSGs), JIT access, and Azure Bastion for secure remote access. Avoid exposing Remote Desktop Protocol (RDP) and Secure Shell (SSH) to the public internet.

More information:

JIT VM access →

Enable logging and monitoring

Ensure forensic readiness by enabling logs for key security events:

- **Activity Logs** - track subscription-level management events.
- **Resource Logs** - capture access and operational events.
- **Azure AD Logs** - monitor authentication and identity risks.
- **NSG Flow Logs** - record inbound/outbound network traffic.
- **Microsoft Defender for Cloud Alerts** - detect and respond to threats.

Both Data Dog and Secure Works have great tutorials on how to ensure full logging is enabled.

More information:

Azure Monitor data collection →

Incident readiness and response planning

Regularly conduct tabletop exercises to simulate incidents and refine response strategies. Define escalation protocols, external engagement policies, and secure communication methods in case primary systems are compromised.

Executives should be prepared to answer to the following questions in advance of any incident:

- Under what circumstances do you notify law enforcement, regulatory authorities, auditors, and the board?
- Will we pay a ransom? If so, how?
- If required, which out-sourced incident response firm will you work with?
- What happens if you lose access to core IT systems for an extended period of time?
- Do you have business continuity and disaster recovery plans in place?
- If the primary communication methods are either unavailable or compromised, do you have backup or out-of-band communications available?
- What working hours are incident responders expected to work in a high severity incident?
- Do you have access to the data required to perform an investigation in all products and services?

More information:

Microsoft incident response plan guide →

Service-specific incident response strategies

Incident response in Azure varies depending on the affected services. Security teams need to apply tailored approaches when investigating incidents involving ADs, VMs, and Azure Kubernetes Service (AKS). This section provides detailed guidance on detecting, containing, and responding to security incidents in these critical Azure environments.

Active directory incident response (now Entra ID)

Azure Entra ID is a key identity and access management service that controls authentication and authorization for users, applications, and services. Threat actors frequently target Entra ID to gain unauthorized access or escalate privileges.

Identifying and disabling legacy authentication

Legacy authentication methods (such as Basic Authentication for Exchange Online) are a common attack vector, allowing password spray and credential stuffing attacks.

Steps to identify and block legacy authentication:

01 Navigate to the Azure Portal:

- Go to Entra ID → Sign-in Logs.
- Click on Columns → Client App, if not already displayed.
- Select Add Filters → Client App → Check all legacy authentication methods.

02 Block legacy authentication using conditional access policies:

- Navigate to Entra ID → Security → Conditional Access.
- Create a new policy:
  - Include: All users.
  - Exclude: Service accounts (if necessary).
  - Conditions: Sign-in risk level = Medium or High.
  - Grant: Block access.

03 Monitor for attempts to bypass blocks using security reports:

- Security → Reports → Sign-in Logs
- Review blocked authentication attempts and investigate suspicious IPs.

More information:

Microsoft's guide on blocking legacy authentication →

Reviewing risky sign-ins in Entra ID

Azure AD provides built-in risk-based detection to identify potentially compromised accounts.

Steps to investigate risky sign-ins:

01 Navigate to Entra ID → Security → Risky Sign-ins

- Look for anomalies such as impossible travel activity or sign-ins from anonymous IP addresses.

02 Check the “Risk Detections” tab

- Provides reports for password spray attacks, leaked credentials, and high-risk login attempts.
- Data retention: 90 days.

03 Remediate risky sign-ins

04 Force a password reset for affected accounts

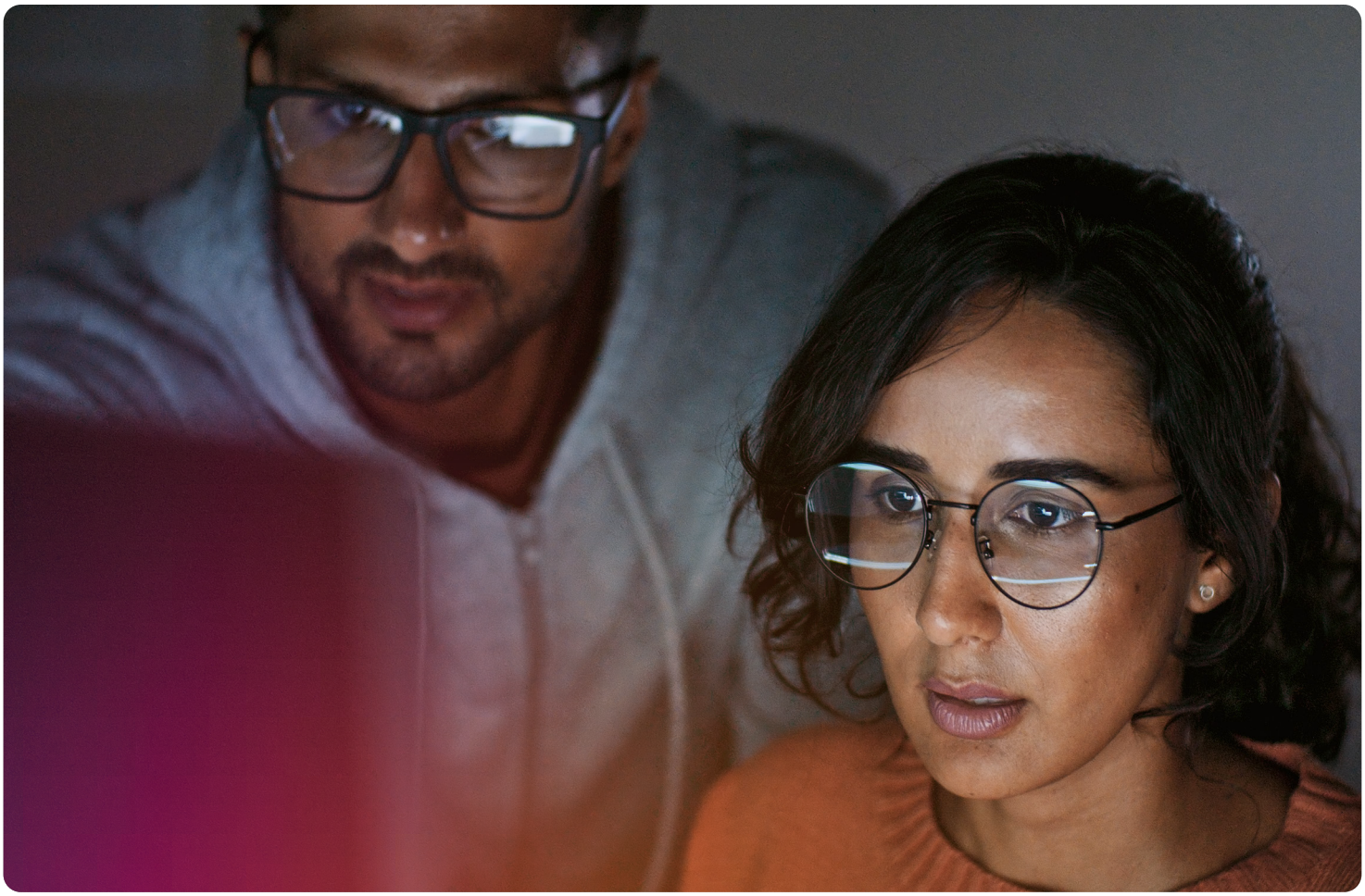
05 Enable MFA if not already enforced

06 Block high-risk users using the following command

```
az ad user update --id <user_id> --account-enabled false
```

More information:

Microsoft risk-based sign-in analysis →



VMs incident response

Azure VMs are frequently targeted by attackers who aim to escalate privileges, exfiltrate sensitive data, or deploy ransomware. Responding to VM incidents requires snapshotting compromised instances, collecting forensic evidence, and implementing containment strategies.

Creating and downloading VM snapshots

Snapshots are point-in-time copies of a VM's disk and are critical for forensic analysis.

Steps to create a snapshot:

01 Identify the resource group and disk name:

```
az disk list --resource-group <RESOURCE_GROUP>
```

02 Create a snapshot of a VM's disk:

```
az snapshot create --name <SNAPSHOT_NAME> --resource-group <RESOURCE_GROUP> --source
<DISK_NAME>
```

03 Export the snapshot for offline forensic analysis:

```
azcopy cp "<snapshot_url>" "C:\forensic_evidence\snapshot.vhd" --check-md5 nocheck
```

More information:

Azure VM snapshot guide →

Granting read-only access to VM snapshots

To ensure forensic integrity, security teams should grant read-only access to snapshots rather than modifying them.

Steps to create a snapshot:

01 Assign a temporary read-access URL:

```
az snapshot grant-access --duration-in-seconds 3600 --name <SNAPSHOT_NAME>
--resource-group <RESOURCE_GROUP>
```

02 Use the generated URL to download the snapshot securely.

Containment strategies for virtual machines

After collecting evidence, contain the compromised VM to prevent lateral movement.

Recommended containment actions:

01 Restrict network access:

```
az network nsg rule create --resource-group <RESOURCE_GROUP>
--nsg-name <NSG_NAME> --name “DenyAllTraffic” --priority 100
--direction Inbound --access Deny --protocol “*” --destination-port-range “*”
```

02 Change the VM’s administrative credentials.

03 Use JIT Access Controls to limit unauthorized access.

More information:

Microsoft’s JIT VM access →

Azure Kubernetes Service (AKS)

AKS is a managed Kubernetes service that allows enterprises to run containerized workloads. Attackers frequently target Kubernetes clusters to exploit misconfigurations, escalate privileges, or exfiltrate sensitive data.

Log collection for incident response

Kubernetes logs are essential for analyzing security events in AKS.

Key logs to collect:

01 Kubernetes API Server Logs

In AKS the control plane (including the API server) is fully managed by Microsoft. Customers do not have direct access to these pods or their logs via kubectl because they are not deployed in customer-accessible namespaces. Instead, control plane events are available through Azure Monitor (when properly configured) or via audit logs if enabled.

02 Audit Logs

Track configuration changes and user actions

```
azcopy cp “<snapshot_url>” “C:\forensic_evidence\snapshot.vhd” --check-md5 nocheck
```

03 Application Logs

Collected via Azure Monitor & Log Analytics

- Enable log collection:

```
az monitor log-analytics workspace create --resource-group <RESOURCE_GROUP>
--workspace-name <WORKSPACE_NAME>
```

- Stream logs to log analytics:

```
az aks enable-addons --addons monitoring --name <CLUSTER_NAME> --resource-group
<RESOURCE_GROUP>
```

More information:

Azure monitor for containers →

Containing a compromised AKS cluster

If an attacker compromises a Kubernetes pod, immediate containment is necessary.

Steps to isolate a compromised pod:

01 Identify the compromised pod:

```
kubectl get pods --all-namespaces --sort-by=.status.startTime
```

02 Delete the compromised pod:

```
kubectl delete pod <POD_NAME> --namespace=<NAMESPACE>
```

03 Use Network Policies to restrict external communication:

```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
name: deny-egress
namespace: default
spec:
podSelector: {}
policyTypes:
- Egress
egress: []
```

More information:

Azure Kubernetes security best practices →

Automating incident response

Incident response automation helps security teams reduce response times, minimize manual effort, and improve accuracy in detecting and mitigating security threats. Azure provides various automation capabilities that streamline incident response tasks, including isolating compromised VMs, collecting forensic artifacts, and analyzing logs for Indicators of Compromise (IoCs).

Automating VM isolation

When a VM is compromised, isolating it quickly prevents lateral movement and data exfiltration. Azure Automation allows security teams to trigger automated isolation workflows based on alerts from Microsoft Defender for Cloud or SIEM systems like Microsoft Sentinel.

Steps to automate VM isolation:

01 **Define** an Automation Runbook to update the VM's NSG and deny all inbound/outbound traffic.

02 **Trigger** the automation when an alert is generated for a high-risk event.

03 **Monitor** and log the isolation event for forensic analysis.

Azure CLI example:

Restricting VM network access

```
az network nsg rule create --resource-group <RESOURCE_GROUP>
--nsg-name <NSG_NAME> \
--name "DenyAllTraffic" --priority 100 --direction Inbound
--access Deny  \
--protocol "*" --destination-port-range "*"
```

More information:

Azure NSG documentation →

Automating memory dump collection for forensics

Memory analysis is crucial for detecting in-memory malware, credential dumping, or active attacker sessions. Azure Automation and Azure Runbooks can automate the collection of memory dumps from suspicious VMs.

Steps to automate memory dump collection:

01 **Use** Azure Automation to execute a script that captures a memory dump of a running VM.

02 **Store** the memory dump securely in Azure Blob Storage for forensic analysis.

03 **Trigger** this workflow automatically based on threat detections.

PowerShell example:

Collecting a memory dump from an Azure VM using DumpIT

```
Invoke-AzVMRunCommand -ResourceGroupName <RESOURCE_GROUP> VMName
<VM_NAME\ >
-CommandId 'RunPowerShellScript' -ScriptPath
'C:\Tools\DumpIt.exe'
```

More information:

Azure run command documentation →

Automating Log Analysis for IoCs)

Detecting IoCs in logs is a key part of incident response. Security teams can use Azure Monitor and Log Analytics to automate IoC searches across Azure environments.

Steps to automate IOC detection:

01 **Ingest** logs from critical Azure services (Activity Logs, Resource Logs, NSG Flow Logs, Defender for Cloud Alerts) into Microsoft Sentinel.

02 **Create** a Kusto Query Language (KQL) script to scan logs for known IoCs (e.g., malicious IPs, suspicious PowerShell execution).

03 **Set up** automated alerts and response actions when matches are found.

Example KQL query:

Searching for Malicious IPs in NSG Flow Logs

```
AzureDiagnostics
| where Category == "NetworkSecurityGroupFlowEvent"
| where RemoteIP in ("203.0.113.5", "198.51.100.8")
| project TimeGenerated, RemoteIP, DestinationPort
```

More information:

Microsoft KQL query guide →

Using Azure automation for incident response

Azure provides multiple automation tools that security teams can use to orchestrate incident response workflows:

Azure automation tool	Use case
Azure automation runbooks	Automate tasks such as isolating VMs, collecting artifacts, and resetting passwords.
Logic apps	Orchestrate complex workflows and integrate SIEM/SOAR tools.
Microsoft Sentinel playbooks	Automate responses like blocking malicious IPs, disabling compromised accounts.
Azure functions	Run custom scripts triggered by security alerts.

More information:

Azure automation overview →

Forensic analysis in Azure

Forensic analysis is crucial for understanding security incidents, identifying root causes, and determining the extent of a breach. Azure provides several built-in tools and services to support forensic investigations, including NSG Flow Logs, Azure Network Watcher, and Microsoft Sentinel.

Analyzing network traffic

Monitoring and analyzing network traffic is key to detecting lateral movement, command-and-control communications, and data exfiltration attempts.

Using NSG Flow Logs

NSG Flow Logs record network activity for resources in a VNet and help security teams analyze suspicious traffic patterns.

Steps to analyze NSG flow logs:

01 **Enable** NSG Flow Logs in Azure Network Watcher.

02 **Store** logs in an Azure Storage account or send them to Log Analytics.

03 **Analyze** logs with KQL queries in Azure Monitor.

Example KQL query:

Identifying suspicious traffic

```
AzureDiagnostics
| where Category == "NetworkSecurityGroupFlowEvent"
| where Action == "Deny"
| project TimeGenerated, SourceIP, DestinationIP, DestinationPort
```

More information:

NSG flow logs →

Using Azure Network Watcher for packet capture

Azure Network Watcher provides deep network inspection through packet capture.

Steps to capture network traffic:

01 Start a packet capture session on a specific VM:

```
az network watcher packet-capture create \
--resource-group <RESOURCE_GROUP> -- <VM_NAME> \
--name PacketCaptureSession --storage-account <STORAGAE ACCOUNT>
```

02 Download, and analyze the packet capture using tools like Wireshark.

More information:

Azure Network Watcher packet capture →

Using Microsoft Sentinel for forensic analysis

Microsoft Sentinel is a cloud-native SIEM and SOAR platform that helps correlate and investigate security incidents across multiple data sources.

Key forensic capabilities in Sentinel:

01 **Log correlation** - collects logs from Azure AD, NSG Flow Logs, VM logs, and Microsoft Defender for Cloud.

02 **Hunting queries** - uses KQL queries to search for anomalous activity.

03 **Incident enrichment** - correlates alerts with threat intelligence feeds.

Example KQL query:

Detecting suspicious PowerShell activity

```
SecurityEvent
| where EventID == 4688
| where ProcessName contains "power shell.exe"
| where CommandLine contains "Invoke-WebRequest"
| project TimeGenerated, Account, ProcessName, CommandLine
```

More information:

Hunting threats in Microsoft Sentinel →

Post-incident review and continuous improvement

A well-structured post-incident review ensures that organizations learn from incidents and refine their response strategies. Conducting a lessons learned session and maintaining detailed documentation improves future response capabilities.

Conducting a lessons learned session

A lessons learned session helps security teams reflect on an incident and improve future readiness.

Steps to facilitate a lessons learned session:

01 **Gather** all relevant stakeholders, including security, IT, legal, and business leaders.

02 **Review** the timeline of events and response actions.

03 **Identify** gaps in detection, response, or containment.

04 **Document** key takeaways and update policies accordingly.

05 **Create** action items for improvements in tools, training, or processes.

More information:

Microsoft incident response best practices →

Documenting the incident response process

Proper documentation ensures that future investigations are more efficient and that response teams can apply insights to new threats.

Key information to document:

- **Incident summary** - attack vector, affected resources, and timeline.
- **Detection & response** - how the attack was detected and what actions were taken.
- **Root cause analysis** - underlying vulnerabilities or misconfigurations.
- **Lessons learned** - what worked well and what needs improvement.
- **Recommendations** - action items for enhancing security posture.

Maintaining an incident response knowledge base improves team efficiency and readiness for future threats.

More information:

NIST guide to incident response documentation →

Common challenges in Azure incident response

Despite strong security tools, Azure incident response faces unique challenges that require strategic planning and proactive mitigation.

Data volatility

Azure's dynamic cloud environment means resources may be created and terminated frequently, making forensic data collection challenging.

Mitigation strategies:

- **Implement** automated evidence collection workflows to capture logs and snapshots immediately after detection.
- **Use** Azure Storage for long-term log retention beyond default limits.

Multi-account environments

Many organizations use multiple Azure subscriptions and accounts, which can complicate centralized security monitoring.

Mitigation strategies:

- **Use** Azure Management Groups to apply consistent security policies across subscriptions.
- **Leverage** Azure Lighthouse for centralized security visibility and incident response.

Limited forensic capabilities in cloud-native services

Some Azure services provide minimal forensic data, limiting deep investigations.

Examples:

- **Azure functions & logic apps** - stateless services with limited logging.
- **AKS containers** - rapidly terminated workloads, leading to loss of forensic artifacts.
- **Cosmos DB & Azure SQL** - restricted access to internal logs.

Mitigation strategies:

- **Enable** extended logging in Log Analytics.
- **Capture** memory snapshots and process logs for containerized workloads.
- **Use** Microsoft Defender for Cloud for real-time threat detection.

Cross-region considerations

Incident response efforts can be complicated by regional data residency laws and log storage limitations.

Mitigation strategies:

- **Replicate** logs across multiple regions using Azure Monitor.
- **Use** global security solutions like Azure Sentinel for centralized investigations.
- **Implement** encryption and pseudonymization to meet compliance requirements.

Human error and misconfigurations

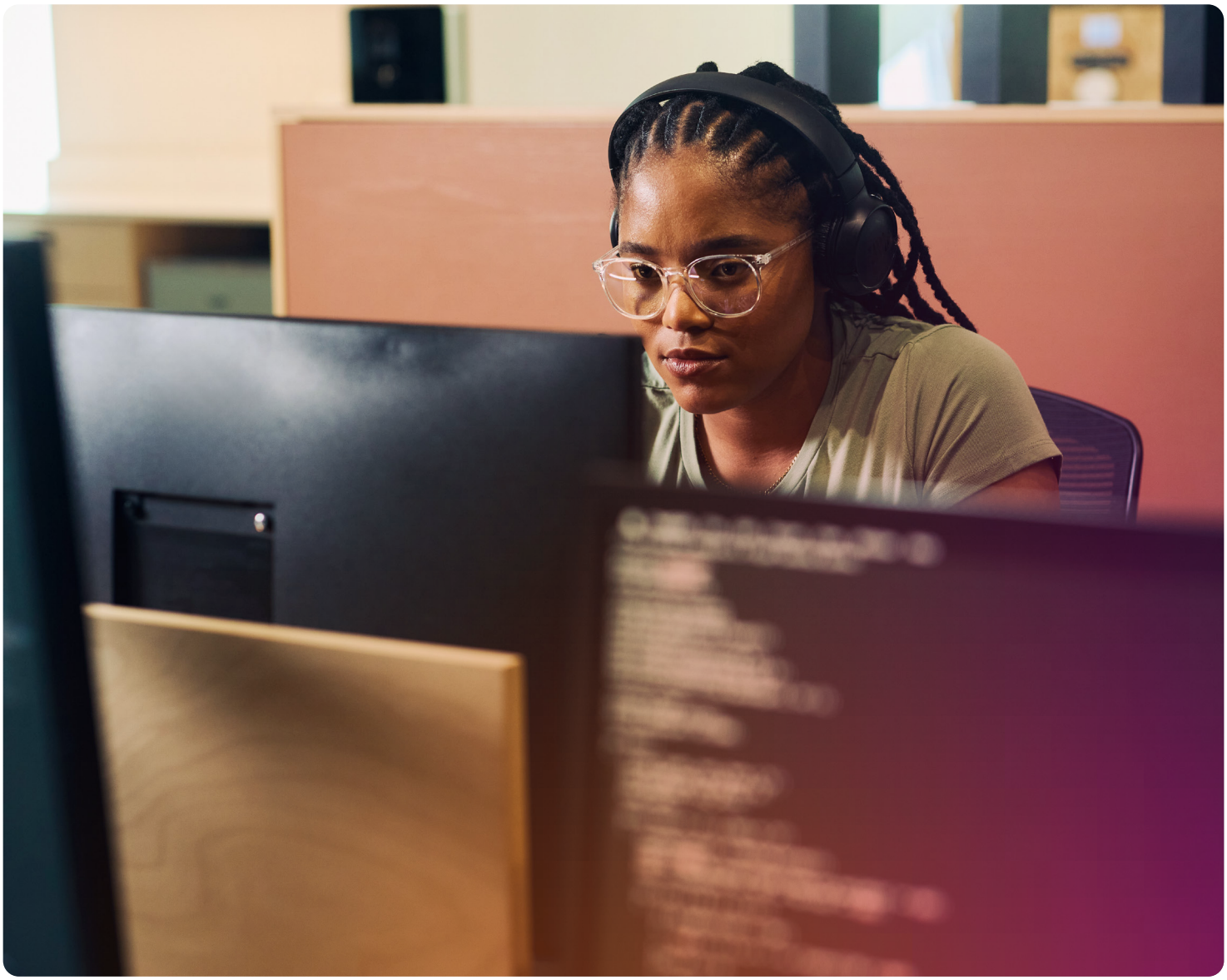
Misconfigurations and accidental exposure of sensitive resources are common security issues in Azure.

Mitigation strategies:

- **Use** Azure Policy to enforce security best practices automatically.
- **Monitor** security posture continuously with Microsoft Defender for Cloud.
- **Regularly** conduct security assessments and automated compliance checks.

More Information:

Azure policy documentation →



Tools and resources

Official Azure tools

These Microsoft-provided tools help detect, investigate, and respond to security incidents in Azure environments:

- **AWS Security Hub** - provides real-time security posture management and advanced threat protection.
- **Azure Sentinel** - a cloud-native SIEM for threat detection, log correlation, and automated response.
- **Azure Defender** - protects workloads such as VMs, databases, and containers with built-in threat detection.
- **Azure AD Incident Response PowerShell Module** - facilitates incident analysis by querying and investigating security events in Azure Active Directory.

Community tools

In addition to Microsoft tools, several community-driven forensic utilities can assist in Azure security investigations:

- **Sparrow** - developed by CISA, Sparrow helps analyze Azure AD and detect compromise indicators.
- **Mandiant Azure AD Investigator** - identifies suspicious activity in Azure AD.
- **AzureHound** - part of the BloodHound suite, AzureHound maps privileges and permissions across Azure environments.
- **Hawk** - a PowerShell-based tool that collects and analyzes Azure AD and O365 logs.
- **Cloud Forensic Utils** - a collection of open-source forensic tools for analyzing Azure environments.

Training and resources

To enhance incident response capabilities, security teams should continuously train and leverage industry-recognized resources:

- **Microsoft Documentation** - covers Azure security best practices, forensic methodologies, and threat detection guidance.
- **SANS Institute** - provides professional training on cloud incident response, threat hunting, and forensic investigations.
- **Microsoft Learn & Certifications** - offers role-based security certifications, including Azure Security Engineer Associate (AZ-500).

More information:

Microsoft security training →

Further reading

Microsoft provides playbooks for specific incident response scenarios in Azure, which can help security teams follow structured response workflows:

- Phishing investigation
- Password spray investigation
- Ransomware attack
- App consent grant
- Compromised or malicious application
- Forensic / legal investigation

Microsoft security checklists and best practices

Microsoft provides a number of security best practice guides for securing Azure environments:

- Azure security best practices and patterns
- Azure Operational Security best practices
- Security best practices for Azure solutions

Conclusion and recommendations

As cloud environments become more dynamic and complex, organizations must adopt a proactive approach to incident response in Azure.

Key takeaways

1

Preparation is critical

Organizations should establish robust incident response plans and enable forensic logging before an incident occurs.

2

Forensic readiness is essential

Ensuring access to NSG Flow Logs, Azure AD sign-in logs, and VM snapshots improves investigation accuracy.

3

Automation enhances response

Leveraging Azure Sentinel Playbooks, Logic Apps, and SOAR tools helps accelerate incident mitigation.

4

Post-incident learning strengthens security

Continuous review and refinement of security processes help adapt to evolving threats.

Recommendations for improving incident response in Azure

Adopt a zero trust security model

Implement least privilege access, MFA enforcement, and network segmentation.

Enhance logging & monitoring

Enable and centralize logs across Azure AD, NSG Flow Logs, and Defender for Cloud.

Utilize incident response playbooks

Create automated workflows using Azure Sentinel Playbooks and Runbooks to improve detection and response times.

Invest in continuous security training

Ensure security teams remain up to date with the latest Azure threat detection and forensic analysis techniques.

Regularly test incident response plans

Conduct tabletop exercises and live response simulations to validate security readiness.

/ CLOUD

How Darktrace can help

Darktrace delivers a proactive approach to cyber resilience in a single cybersecurity platform, including cloud coverage. Darktrace / CLOUD is a real time Cloud Detection and Response (CDR) solution built with advanced AI to make cloud security accessible to all security teams and Security Operations Centers (SOCs). By using multiple machine learning techniques, Darktrace brings unprecedented visibility, threat detection, investigation, and incident response to hybrid and multi-cloud environments.

Darktrace's cloud offerings have been bolstered with the acquisition of Cado Security Ltd., which enables security teams to gain immediate access to forensic-level data in multi-cloud, container, serverless, SaaS, and on-premises environments.

Darktrace is a valuable tool for automating incident response processes.

Learn more about Darktrace / CLOUD for Azure:

Read the data sheet →

Dive into how Darktrace / CLOUD works:

Read the solution brief →

See Darktrace in action with a personalized meeting:

Request a demo →

About Darktrace

Darktrace is a global leader in AI for cybersecurity that keeps organizations ahead of the changing threat landscape every day. Founded in 2013 in Cambridge, UK, Darktrace provides the essential cybersecurity platform to protect organizations from unknown threats using AI that learns from each business in real-time. Darktrace's platform and services are supported by 2,400+ employees who protect nearly 10,000 customers globally. To learn more, visit http://www.darktrace.com.

North America: +1 (415) 229 9100 Europe: +44 (0) 1223 394 100 Asia-Pacific: +65 6804 5010 Latin America: +55 11 4949 7696

darktrace.com | info@darktrace.com