

SANS



CLOUD SECURITY:

Making Cloud Environments a Safer Place


In partnership with:



Google Cloud



Microsoft



Large enterprises are increasingly operating in a multicloud environment, either by choice or by chance. As a result, organizations, security teams, and we—as security professionals—are on a continuous journey to develop multicloud security capabilities to enable businesses and effectively respond to the changing threat landscape.

In this book we have, for the first time, a coming together of security leaders from Microsoft Azure, Google Cloud, and independent technical experts from SANS Institute who are sharing their perspectives on building cloud security capabilities as well as best practices for key cloud security pillars.

This book starts with a view on cloud-specific threats that can inform cloud security strategies. What follows is foundational information for key areas such as IAM, data security, and visibility. But often, foundational information does not provide enough direction. People typically learn best when making mistakes and learning lessons the hard way. Knowing this, the contributors to this book convey these lessons in the form of various security anti-patterns that highlight important “not to do” items. Cases studies like these highlight cloud security weaknesses and what you can do to shore them up.

As they say, experience is often the best, if not bitterest, teacher. So please, take some time to learn from the experience of these expert contributors. Don’t leave your cloud security journey to chance.

Frank Kim

Fellow and Curriculum Lead
SANS Institute

Table of Contents

Chapter I **Securing the Hybrid Enterprise: Identity and Access Controls in the Cloud** **4**

Introduction	5
Security Anti-Patterns	6
Case Study: Sophisticated Attacker vs. Unsophisticated Attacker	12
Taking Control: Pragmatic Practices	12
Conclusion - Chapter I	15

Chapter II **Top 3 Cloud Security Weaknesses, Misunderstandings, and Mitigations** **16**

Introduction	17
Weakness #1: Network Visibility Challenges	17
Weakness #2: Reckless Adoption of Third-Party Code	21
Weakness #3: Cross-Provider Deployment Issues	28
Conclusion - Chapter II	28

Chapter III **Modern Cloud Security: Shared Responsibility to Shared Fate?** **31**

Introduction	32
Building a Cloud Security Strategy	33
Conclusion - Chapter III	45

**CLOUD
SECURITY:**
Making Cloud
Environments
a Safer Place

Chapter I

SECURING THE HYBRID ENTERPRISE: IDENTITY AND ACCESS CONTROLS IN THE CLOUD

Written by Moses Frost and Roberto Bamberger

SANS



Microsoft

Introduction

Organizations of all sizes and types quickly adopt cloud technologies, resulting in hybrid enterprises. In many cases, this move to cloud-centric computing and hybrid enterprises was accelerated due to the COVID-19 global pandemic and a call to “work from home.” Similarly, threat actors ranging from amateurs, financially motivated cybercriminals, and nation-state actors focused on espionage have also adapted to attack these cloud and hybrid environments. While most organizations are adopting zero trust concepts to secure their hybrid enterprises and cloud environments, threat actors have been able to exploit incomplete implementations. The partial implementation of zero trust ideas, combined with the application of security concepts from traditional on-premises environments, have resulted in a set of “anti-patterns” that threat actors of all types have used successfully. In this paper we discuss some of these common anti-patterns that have been exploited in recent attacks.

More and more companies are adopting any number of business-critical services, many of which are hosted outside the company's data center, requiring authentication to systems not under its control. The predominant authentication mechanism companies are adopting is a federated identity approach, in which the identities are stored in an identity system under company control and users have one identity for all services. In today's world, many of these federated identities rely on technologies such as **OAuth2**, **OpenID Connect**, and **SAML**. Many companies adopt cloud-based identity providers (IdPs) to remove the reliance on their own data center servers and better scale their user base. In effect, identity providers become the de facto perimeter to cloud services that include key computing infrastructure, platforms, and software. The idea of a firewalled perimeter, the standard the industry had used for decades, is no longer the actual perimeter or demarcation point. *Identity* is the new perimeter.

As more and more organizations embrace cloud-centric approaches, the cyber threats associated with these approaches are increasing, leading many organizations to experience challenges securing these environments. Although zero-day vulnerabilities and stealthy malware are still important attack vectors, the multiple cloud solutions and hybrid computing environments, which are omnipresent, provide additional opportunities for malicious actors to achieve their objectives.

Cloud-centric attacks and compromises are notably different from attacks on physical on-premises networks. Compromised identities and identity-based breaches are taking the place of malware for persistence in an environment. Traditional supply chains consisting of hardware and software are being supplanted by more complicated trust chains among platforms, identities, cloud providers, vendors, and enterprise environments. The notion of a well-established, defendable security perimeter is no longer relevant (see Figure 1).

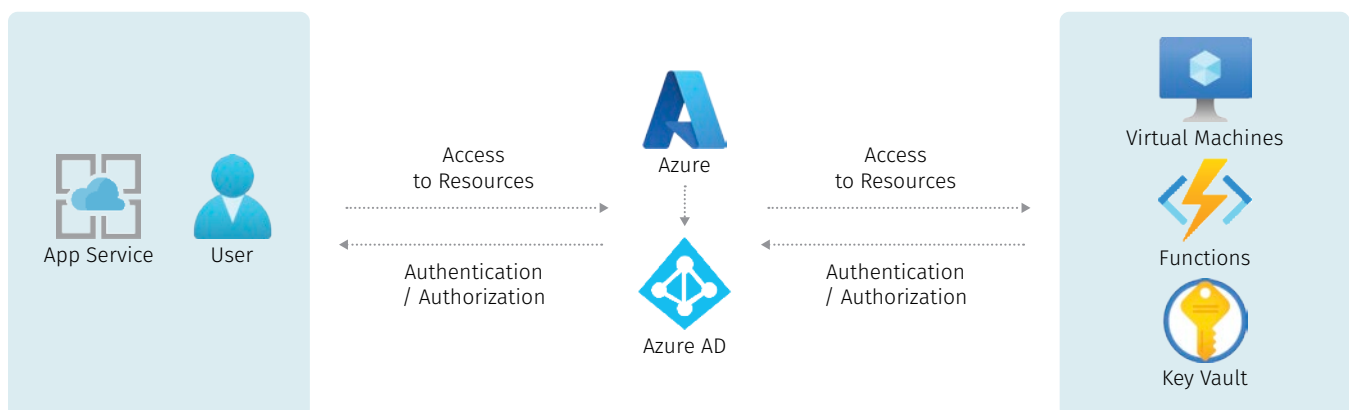


Figure 1. Identity Key to Infrastructure in the Cloud

Identity is core to not just software-as-a-service (SaaS), including Office365, Google Workspace, Salesforce, and Workday, but also to infrastructure-as-a-service (IaaS) solutions, such as Azure. Identity is a requirement for components to talk to one another through the cloud-native channels. Most of the cloud components need an identity to speak to other cloud components. This is analogous to the way a machine account is used in Azure Active Directory (AD) environments. The biggest difference is that just creating a resource in your directory is not enough for components to speak to one another. They need to be granted access and be scoped correctly.

One key component of cloud-based identity providers is that they can automatically operate at “cloud-scale” and leverage global threat intelligence and sophisticated machine-learning-based algorithms to assess the risk associated with each authentication request. Therefore, modern cloud-based identity services are also a critical component in defending an environment from malicious use and assisting security professionals in detecting threats against their environment.

When you look at today’s landscape of security products, a tight coupling of endpoint, network, and cloud in products that bolster detection and response is evident. Many of these companies use the banner **XDR**. The industry and practitioners may not realize how tightly coupled identity is to today’s landscape. In the future, we may see cloud-native malware. In that scenario, we may look at identity the way we look at endpoint technologies: Prevent where you can; detection is a must. In the future, cloud-native malware will have an identity component.

To better understand common security weaknesses, misunderstandings, and mitigations, this paper briefly reviews two recent high-impact threat actor groups (Nobelium and LAPSUS\$ group), which were responsible for significant impacts on multiple organizations. Both groups were able to compromise various cloud and traditional environments. Nobelium has been described as one of the most advanced threat actor groups, with robust operational security and highly advanced tools, techniques, and practices. In contrast, the LAPSUS\$ group is not recognized as a highly advanced group. Yet, both groups were able to be effective. Both groups exploited common weaknesses and misunderstandings to achieve their action objectives. Both groups could bypass security controls, such as requiring multifactor authentication (MFA), and move freely between cloud and on-premises environments. Both groups could access large volumes of sensitive information using compromised identities.

Security Anti-Patterns

One of the significant cultural changes that has occurred to the technology space in the past several decades is the concept of the *zero trust model*. The model was refined over time but was born out of a need to contain damage post-breach. The model itself attempts to define several characteristics of zero trust:

Explicit verification

Least privileged access patterns

Assumption of breach

Each one of these characteristics can be implemented with seemingly unintended consequences. Whether due to misconfiguration of security controls or unintended consequences of security controls, the difference between the intended and expected security posture and the security posture as implemented provides threat actors with multiple opportunities to gain unauthorized access. The following are commonly observed anti-patterns in today's modern hybrid enterprises:

- **Geofencing**—*Geofencing* is an anti-pattern observed across multiple victim organizations. An IP address is effectively used as a trust factor in its simplest form.
 - **Block/Allow lists based on IP to geolocation**—Two variations on this anti-pattern attempt to block authentication/access from specific geolocations that meet a particular risk model (e.g., a list of banned countries or assuring that access is provided only to systems from a limited set of allowed countries).
 - **Allow list for corporate IPs**—A similar anti-pattern using an IP that egresses from an organization's "trusted" IP address range is trusted without further verification.

Both patterns violate one of the fundamental concepts in zero trust architectures, which include "never trust, verify explicitly." Please note that the defaults in many IdPs, including Azure AD, *encourage* some of these patterns. They are there for *convenience* and *operations*; they are not there to enhance security (see Figure 2).

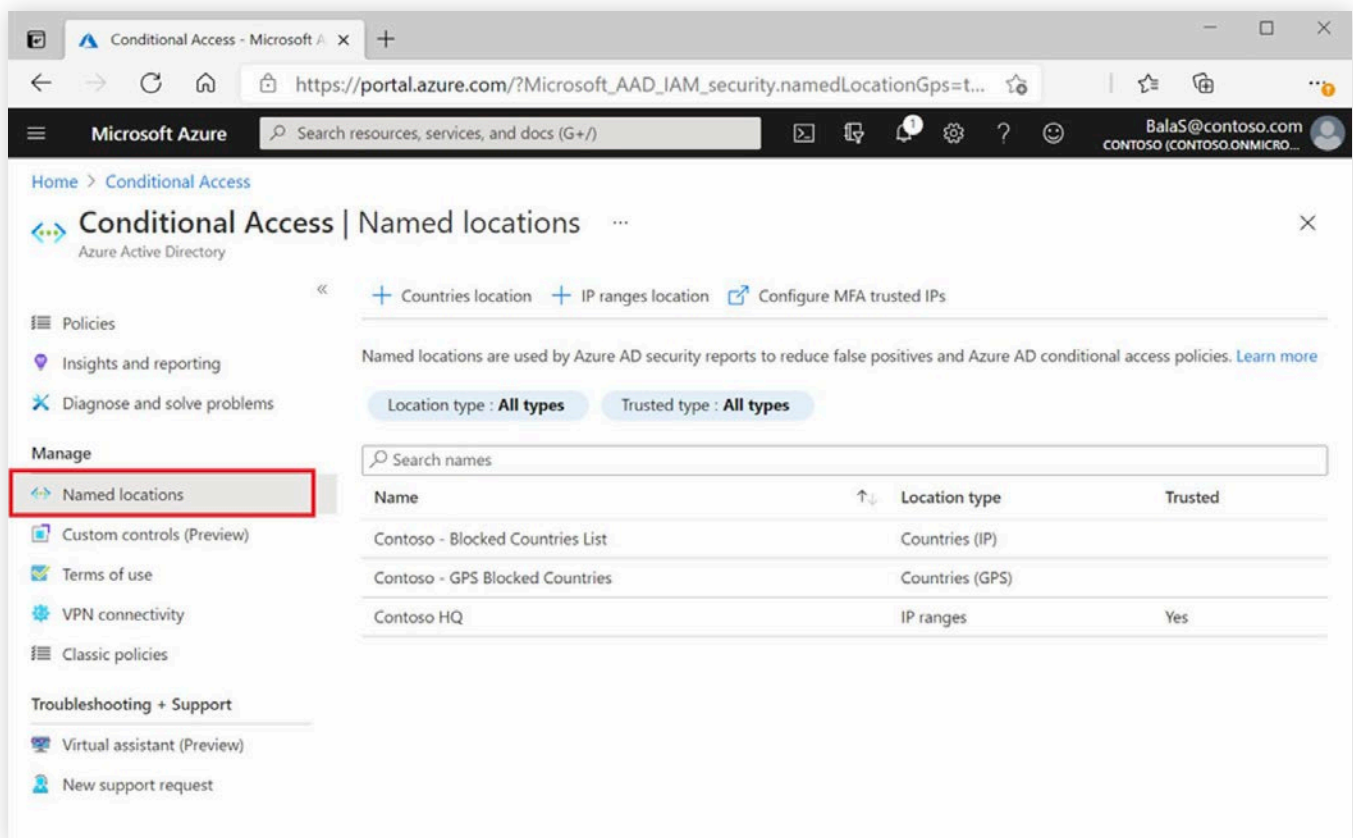


Figure 2. Named Locations in Azure AD

- **Ineffective MFA implementations**—Multiple ineffective MFA strategies have been observed being abused by threat actors. These include:
 - **Allowing use of legacy authentication protocols**—Legacy authentication protocols (e.g., **IMAP**, **POP3**, **FTP**, basic authentication, digest authentication, etc.) do not support workflows for modern authentication standards. These modern authentication standards include MFA, as well as **WebAuthN** and **FIDO**. Specific RADIUS servers can combine passwords with one-time passwords (OTP), but these are workarounds (i.e., **password123456**, where **123456** is the OTP code from your authenticator). Plan to make legacy authentication protocols obsolete.
 - **Not requiring MFA for all users/identities**—In multiple incidents, specific identities did not require MFA. In some cases, this was due to user reluctance/objections. In many cases, the higher in the organizational hierarchy a user is, the more likely they will object to adhering to security policies. In other cases, MFA is not enforced for nonhuman identities. In addition, no compensatory controls were in place for many of these identities.
 - **Not requiring MFA for trusted networks/locations (see “Geofencing”)**
 - **MFA susceptible to “prompt bombing”**—In multiple incidents, a threat actor acquired a valid username/password combination. If MFA were configured to accept a phone-app push notification or to receive a phone call and press a key as the second factor, the threat actor would make multiple requests to the end user’s legitimate device until the user accepted the authentication.
 - **MFA implementations in which a single user verifies the additional prompt for multiple parties**—MFA was configured for a set of accounts to use a voice prompt in one example. All reports were routed to the same phone number. One individual was tasked with approving all MFA requests for that phone number. While satisfying a technical policy that these accounts had to be protected by MFA, the implementation added no additional security to the authentication process.
 - **MFA implementations subject to adversary-in-the-middle (AiTM) style attacks**—Many MFA solutions can fall victim to an AiTM style attack. Only MFA solutions that support FIDO2 and certificate-based authentication are viewed as “phish-resistant.” (See [link](#) for a description of one such AiTM campaign.)
 - **Token or cookie theft**—In multiple cases, the threat actor could acquire authentication tokens (or session cookies) from a compromised end-user device and use these tokens/cookies, which contain a claim that MFA was already satisfied, to gain access to cloud resources. Threat actors have targeted personal, nonmanaged devices used by users of interest to acquire access and conduct additional reconnaissance for targeting an individual. Allowing end users to use nonmanaged devices increases the risk to an organization because the compromise of an unmanaged device may be challenging to detect and remediate.
 - **Device registration by a threat actor**—In multiple instances, threat actors could register their own devices for MFA. Threat actors have been observed social engineering a help desk to register a new device, as well as simply completing MFA device registration when a user has not yet done so when accessing an account that requires MFA.

- **Ineffective monitoring**—If implementing an “assume breach” approach, having effective monitoring processes and solutions in place is critical, including:
 - **Not monitoring all native signals**—Many cloud-based systems have built-in monitoring and alerting. In many cases, a customer’s security operations center (SOC) was not ingesting all signals into its SOC environment; therefore, no security professionals monitored built-in alerts for specific systems. Azure AD performs a risk assessment for each authentication event it processes. In some cases, security analysts did not review these risk events.
 - **Limited data flowing to a SIEM**—SIEM’s typically are sized by the number of events per second. Cloud infrastructures may overwhelm a SIEM’s capability to ingest the total number of events per second. Those responsible for SIEM implementations may instead only ingest alerts or filtered data into the SIEM environment instead of the underlying raw data.
 - **Limited data retention**—If an organization uses a centralized SIEM or security orchestration, automation, and response (SOAR) solution, in many cases, data retention may be limited to a short period. Also, in some implementations, filtering may be applied to reduce data volume. Some data may not be available in the SIEM solution in those cases. In one case, data ingested into the customer’s SIEM was truncated because their connector used **UDP** as the transport protocol.
 - **Assuming visibility for known suspicious/malicious activities**—Many organizations believe that a threat actor activity is currently being audited and that it will become visible for alerting in their SIEM solution. Verifying that critical events such as configuration changes, are indeed flowing into the SIEM is critical. Regular penetration testing can be extremely effective in finding visibility gaps.
 - **Gaps between monitoring/security teams**—In many hybrid enterprises, security monitoring has been split between multiple organizations, which creates visibility gaps.
 - **Not enabling auditing**—Although in many SaaS products, some level of auditing is enabled by default, not all levels of auditing are enabled. In addition, in many IaaS environments, auditing needs to be enabled on a per-service basis.
 - **Use of federated identity providers (IdP)**—In many cases where a federated identity provider (e.g., Active Directory Federation Services (ADFS), Okta, Duo, Ping Federate) is used, all authentication telemetry may not be audited and/or forwarded into a SIEM environment. In one recent case, analysts discovered that successful authentications using legacy protocols to a federated identity provider were not audited/recorded by default for that IdP. In cases where ADFS was the federated IdP, ADFS Connect Health was not in use and therefore failed authentications from brute force attacks and were not detected by Azure AD Identity Protection features. Having a centralized view of authentications, including those from a federated IdP is critical for detecting malicious activities.

- **Failure to implement the least privileges approach**—A least privileges approach helps in reducing the overall attack surface and also the impact of a compromise of a single identity or cloud component. This can be implemented in several ways, including:
 - **On-premises identities mapped to privileged cloud identities**—In most cases, cloud identities are synchronized from an on-premises environment. If privileged identities are synchronized from on-premises accounts, compromise of the on-premises environment can easily lead to a loss of administrative control of the cloud environment.
 - **Privileged identities not using Privileged Identity Management (PIM) solutions**—PIM solutions can be used to temporarily grant administrative privileges to an identity. This reduces the attack surface by not having persistent administrative access associated with an identity.
 - **Privileged identities not requiring MFA**—Although many organizations have become more aggressive at requiring MFA for privileged identities, there are still cases where “nonhuman” privileged accounts may not require MFA. If such accounts must be allowed, compensating protective and/or detective controls are required.
 - **Privileged identities for “day-to-day” use**—The use of privileged identities for everyday tasks introduces risk to an organization, in part, by increasing the attack surface for those privileged identities. The use of a privileged account for nonprivileged activities can increase the attack surface, exposing the privileged account to brute force attempts (such as password sprays), credential theft (such as through credential phishing), and other attacks. Combined with privileged identity management and implementation of a least privileges approach, risk can be reduced.
 - **Over-privileged applications (service principals)**—Threat actors routinely inventory all service principals. If a service principal has privileges of interest to the threat actor, the perpetrator will attempt to gain access to that service principal as a mechanism to elevate the threat actor’s privilege. This can occur in single-tenant or multi-tenant applications.
 - **Over-privileged users in infrastructures (shadow admins)**—When federating access to IaaS, such as Azure, AWS, and Google Cloud, users can end up with equivalent administrator access as an unintended consequence of having too many permissions. While not explicitly granting “Administrator” access to the cloud environments, there are permissions that effectively enable the user to act as administrator.
 - **Allowing end users to consent to applications**—Allowing end users to consent to applications (with no restrictions) enables attack scenarios such as consent phishing.
 - **Allowing external identities to have privileges**—Allowing external identities with guest access to a tenant to have administrative privileges can lead to a multi-stage compromise. Compromise of this external identity can provide privileged access to a victim tenant.
 - **Not knowing about Delegated Admin Privileges (DAP) provided to partners**—DAP is a feature in Azure AD where resellers can be granted delegated administrative privileges to a customer’s tenant, thereby enabling them to have administrative privileges to the tenant. In many cases, customers did not realize that DAP had been granted to a third party and, therefore, were not monitoring for malicious activity.
 - **Not knowing about Admin on Behalf Of (AOBO) privileges provided to partners**—AOBO privileges can be assigned to a third-party partner organization for Azure Infrastructure as a service resource.

- **Improper handling of secrets**—In multiple recent investigations, a common theme was the improper management and handling of secrets. Secrets, as the term implies, need to be kept secret. The general concept of a secret in this context can include, but is not limited to, user and machine certificates used for authentication, username/password combinations, the ability to register a new MFA device, the ability to change a secret/access method for a user, passwords, or certificates used by **OAuth** applications. Some examples of improper handling of secrets include:
 - **Credentials in code**—Secrets can often be found in code for internally developed applications. Credentials enterprise applications/service principals can often be found for in public platforms such as **GitHub**.
 - **Credentials in documentation**—Credentials may be included in internal documentation, such as a new employee onboarding guide.
 - **Certificate handling**—User and machine certificates used for access control and authentication need to be managed as secrets. This includes smart-card certificates. In multiple cases, threat actors could acquire certificates easily accessible to end users in an enterprise environment and then use these certificates to gain further access into the environment. Threat actors have been observed collecting certificates distributed via email messages and file shares. Threat actors routinely search SharePoint, OneDrive, and Exchange for certificates and other credentials.
 - **Cleartext credentials**—There are multiple scenarios in which system access credentials are handled in cleartext. This can include the use of weak protocols, which do not encrypt credentials in transit, or using hard-coded credentials in scripts or scheduled tasks.
- **Poorly secured bastion hosts**—Bastion hosts can be a very good way to provide access to secured environments. Quite often, identities are tied to these bastion hosts, effectively ensuring that these bastion hosts have administrative rights to networks. They can be overprivileged in firewall rulesets and, due to their existence, can have a “big seat” at the table. These systems, if accessed by attackers, can quickly turn into pivot points. Misconfigurations in these systems are common and include:
 - **Insecure protocols exposed to the internet**—Remote Desktop Protocol (RDP), while a very powerful protocol, should be tightly controlled. Quite often, environments have been set up allowing RDP remote access from any IP address on the internet into a bastion. This, combined with weak authentications (only username and password), as well as insecure configurations (no account lockout), has led to compromise.
 - **Poorly configured protocols**—SSH has proven to be a powerful protocol that can accommodate many scenarios, such as port forwarding, tunneling, and remote configuration management. It was also designed to be forward facing to the internet. We still, however, see poorly configured systems, including those that allow for username and password authentication, no account lockout, and weak selections for passwords.

Case Study: Sophisticated Attacker vs. Unsophisticated Attacker

Nobelium and LAPSUS\$ are two sides of the same coin in many ways. Nobelium is a threat actor known to have breached SolarWinds. The Nobelium threat actor has a much more sophisticated toolchain.¹ The actors themselves spent a great deal of time creating a long-term, slowly persistent backdoor through the supply chain. It was not just costly to develop, but also costly to maintain. It required rigor and discipline to make the appropriate calculations and stay hidden.

Nobelium deployed Solorigate (or SUNBURST), the SolarWinds backdoor, over a matter of six months from initial entry (September 2019 to March 2020). This is a highly patient attacker group with motives to attempt to remain stealth. In March 2021, Microsoft Threat Intelligence Center wrote an article on how Nobelium used third-party service providers with delegated Azure AD privileges to access cloud accounts of third-party victims. This was another attack similar to the supply chain attack for which the group was known for using Azure AD as the trampoline.

LAPSUS\$ is a different type of group with an altogether different motivation. The group is attempting to quickly monetize and invests very little in attribution and operational security. Instead, the group has used Azure AD similarly to Nobelium, abusing the trust that people have placed in Azure AD. The group frequently uses MFA prompt bombing and Azure AD Domain Joins to attempt to circumvent controls.

These are two groups with wildly different motives, different operational practices, and completely different skill sets. However, each of these groups could leverage identity and cloud components to circumvent controls and gain access to sensitive datasets, such as source code. The same groups can be targeting your organization or using your organization to get other targets.

Taking Control: Pragmatic Practices

What actions can you take to improve your security posture, whether you are a decision maker, architect, consultant, or individual contributor? The best answer is the boring answer; but it's not the easy answer. The tedious work, which few want to address, is mundane and requires an extreme level of organizational discipline to maintain. While that level of effort is being created internally, organizations can take more pragmatic, digestible steps to start the process.

The following is a list that is pragmatic and can be done to start reducing the attack surface in identity:

- **Separate user and admin accounts.** Have a specific admin account for administrative actions and a day-to-day user account. Use the admin account only for administrator-related activities. Do not use this Azure AD admin role for the management of Azure resources.
- **Have a separate secured Privileged Access Device for administrator access.** Consider using a specific system, such as a Privileged Access Device,² that is hardened for administrator-level access. Ensure that more than one administrator account exists. Monitor these accounts closely. Consider restricting the location from which these accounts be accessed.

¹ ["FoggyWeb: Targeted NOBELIUM malware leads to persistent backdoor,"](#) Microsoft, September 27, 2021

² ["Securing devices as part of the privileged access story,"](#) Microsoft, September 02, 2022

- **Universally enable MFA regardless of location.** Remove the location-based MFA. Enforce MFA on all accounts, with the exception being accounts that are service principals (or services) that are programmatic.³
- **Train your users on MFA prompts.** Make sure your users understand when an MFA prompt is warranted and when it is suspicious. Train them on what prompt bombing is to ensure they are aware of what is happening and know when to ask for help.
- **Use MFA number matching.** Specific MFA authenticators can eliminate prompt bombing by leveraging MFA number matching, in which a user is presented a number and must enter the number into the prompt (see Figure 3). Certain nonenterprise (consumer) IdPs do this today as a default, such as iCloud. Others, like Azure AD, are currently set to Opt In.⁴

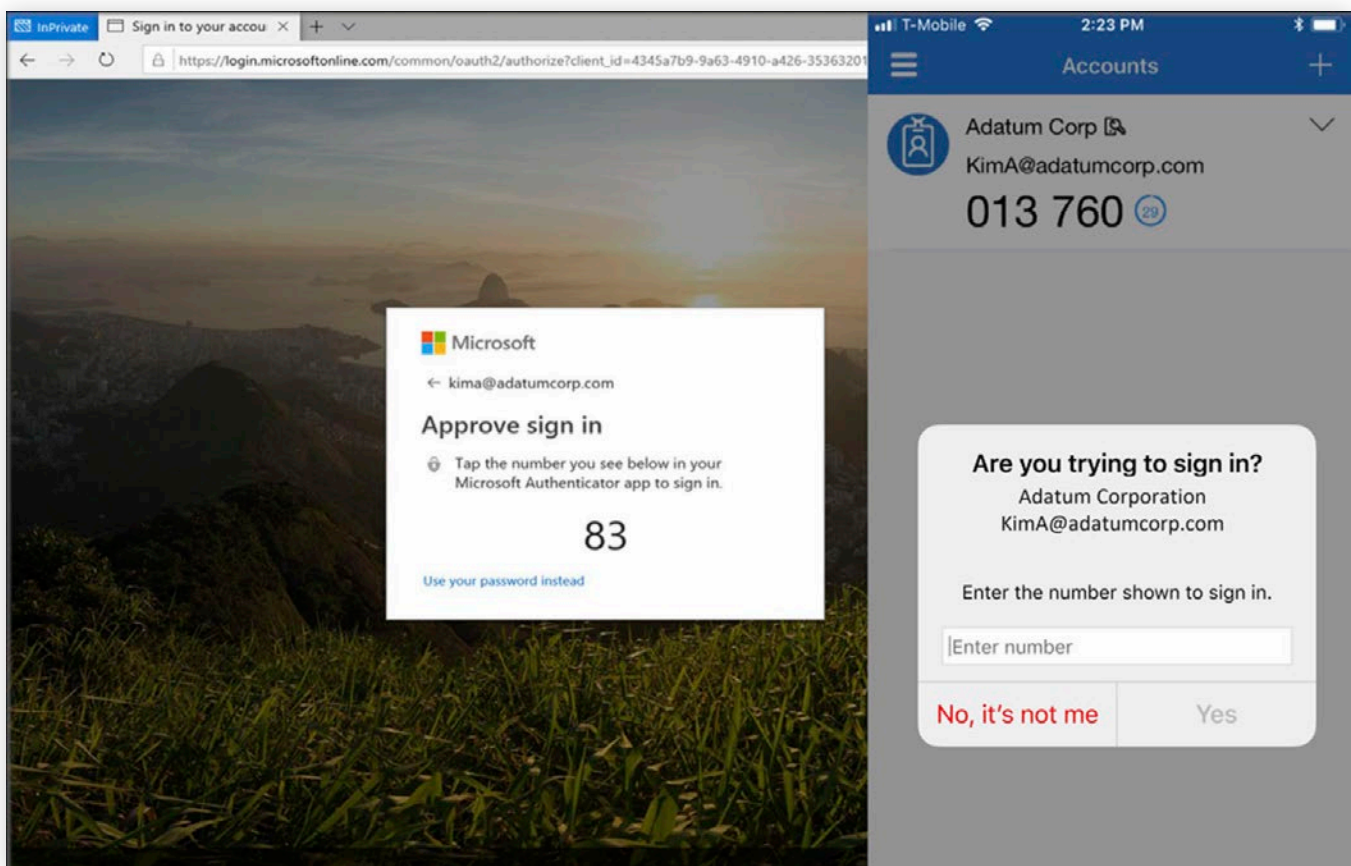


Figure 3. MFA Number Matching Example

³ ["Using the location condition in a Conditional Access policy"](#) Microsoft, August 15, 2022

⁴ ["How to use number matching in multifactor authentication \(MFA\) notifications \(Preview\) - Authentication Methods Policy"](#) Microsoft, September 02, 2022

- **Eliminate passwords and go passwordless.** Certain systems will support the concept of *passwordless authentication*, in which a user is able to pick a **very strong** password with MFA. The system can then use various mechanisms to automatically log the user in using PIN or biometrics to unlock the system. This frictionless authentication system does not fully eliminate passwords but can allow the user to set stringent passwords that are not constantly being asked for.⁵
- **Use SSO or seamless authentications.** Use SSO to federate your login to third parties to eliminate the need for maintaining multiple accounts and passwords in as many places as possible.
- **Use a password vault.** When needed, use a password vault and ensure that all websites have different passwords. For an enterprise, a password vault can be an enterprise-wide solution that all your employees can use or the browser's built-in vault.⁶
- **Set the MFA reauthentication interval to a reasonable timeframe.** Many systems default to a 14-day reauthentication interval. This interval is designed to log the user out and ensure that the person using the system is, in fact, the owner of the identity. Some may consider the 14-day potential attacker dwell time to be too long. Consider shortening it to a reasonable level. Perhaps a weekly reauthentication is sufficient.⁷
- **Use continuous access evaluation where it is available.** Continuous access evaluation is an Azure AD-specific mechanism that uses specific signals to alert the Azure AD system to reauthenticate the user under various conditions. The most relevant one here is high user risk being triggered by Azure identity protection. Please keep in mind the caveats of what applications and systems support this Azure AD-specific feature.⁸
- **Eliminate legacy systems where possible.** Many businesses use a hybrid identity scenario in which a legacy system, such as Microsoft Active Directory Domain Services, and a third party IdP exist. Consider deprecating the use of Microsoft On-Premises Active Directory Domain Services to just the bare minimum number of systems that use it. Modernize the infrastructure, where possible, to leverage newer technologies and deprecate older technologies.
- **Secure private keys.** Secure key material, make the handling of private keys something of a sensitive nature and not something that is generally left exposed in any manner. Consider using a hardware security module (HSM), or HSM-like system that makes key management easier. Do not store keys for any length of time in a public-disclosed space, such as a desktop, email, or on server filesystem. Password protect keys with strong passwords.
- **Rotate static keys often.** Some systems support static keys and static credentials. Have a system in place to rotate keys. Catalog all the deployed keys and turn them frequently.

⁵ ["Passwordless authentication options for Azure Active Directory"](#) Microsoft, August 25, 2022

⁶ ["Microsoft Edge password manager security"](#) Microsoft, August 26, 2022

⁷ ["Optimize reauthentication prompts and understand session lifetime for Azure AD Multi-Factor Authentication"](#) Microsoft, September 01, 2022

⁸ ["Continuous access evaluation"](#) Microsoft, September 01, 2022

- **Remove nonbusiness grants.** Use a detection tool to catalog and remove app consent that is not specific to your company. These tools can audit and control third-party access.⁹
- **Remove excessive permissions from partners and third parties.** In several systems such as Azure AD, partners can gain delegated admin privileges. Similar to least privilege concepts, these should be closely audited because these partner connections can open you up to supply chain attacks.¹⁰
- **Harden applications that use authorization.** When creating applications that leverage an IdP as the authorization (or authentication) system, use secure defaults. Require **TLS**. Enable encryption and integrity checking for **SAML**. The application should check with the IdP to ensure that a grant was created and is still valid.¹¹

Conclusion - Chapter I

There are no easy answers when it comes to securing systems. If that were true, we could push the automation button and do something else. Some absolute truths are threaded across this paper if you read carefully enough. Securing an identity deployment will require discipline, oversight, persistence, and patience. This was the idea behind providing a pragmatic approach to best practices. Instead of giving a prescriptive list of “you must practice least privileges,” the pragmatic approach provides a chance to help you start strolling down the road to security and eventually progress to running. Having visibility into the environment is key. Harden where you can. Use automation and software to close the gaps as needed. Make sure you use testing, such as manual penetration testing, automation, and whatever other mechanisms you may have to validate configurations.

Disclaimer

©[2022] Microsoft Corporation. All rights reserved. This document is provided “as-is” and is for informational purposes only. Information and views expressed in this document, including URL and other internet website references, may change without notice. You bear the risk of using it and MICROSOFT MAKES NO WARRANTIES, EXPRESS OR IMPLIED. Examples herein may be for illustration only and, if so, are fictitious. No real association is intended or inferred.

This document does not provide you with any legal rights to any intellectual property in any Microsoft product.

⁹ [“OAuth app policies”](#) Microsoft, May 29, 2022

¹⁰ [“Obtain permissions to manage a customer’s service or subscription”](#) Microsoft, June 22, 2022

¹¹ [“Microsoft identity platform best practices and recommendations”](#) Microsoft, June 01, 2022

Chapter II

TOP 3 CLOUD SECURITY WEAKNESSES, MISUNDERSTANDINGS, AND MITIGATIONS

Written by [Aaron Cure](#), [Brandon Evans](#), [Pierre Lidome](#), and [AJ Yawn](#)

SANS

Introduction

Security incidents are on the rise. According to the 2022 Cloud Security Report by Checkpoint, 27% of global organizations have experienced a security incident in their public cloud infrastructure within the last 12 months, with 23% of those caused by security misconfigurations in cloud infrastructure.¹ Each major cloud service provider (CSP) lists security as a benefit of the cloud and the CSPs work hard to uphold their end of the shared responsibility model. The commitment to security has resulted in an overwhelming number of cloud services that can overwhelm security practitioners. This becomes even more challenging in a multi-cloud environment.

This paper will explore the top cloud security weaknesses that the authors have observed in 2022, the attacks that exploit them, the misunderstandings that make cloud security more difficult, and effective mitigation strategies to implement in each of the three major cloud service providers.

Weakness #1: Network Visibility Challenges

Pierre Lidome

Flow logs are important to investigations because they capture metadata for conversations between a source and a destination. Only the metadata is captured, not the actual conversation.

There are no standards to the format of cloud flow logs and each cloud provider adds to the traditional NetFlow format. Thankfully, each vendor provides good documentation showing which fields they capture as part of the flow record.

At a high level, flow logs capture information about the IP traffic going to and from network interfaces in your cloud instance. It's often written that flow logs record source IP, source port, destination IP, destination port, and protocol (also called 5-tuple). While this is correct, cloud flow logs record a lot more information. AWS records up to 29 fields if the custom format is selected. Azure records about 20 fields depending on the selection of version 1 or version 2. Google Cloud records 15 fields (with the metadata option enabled).

Once this data is recorded, where is it stored? This is an important consideration as it will impact our ability to analyze it. The choices are usually:

- In-cloud application where it can be viewed and analyzed
- Storage account for longer term retention and export out of the cloud
- Streaming to an external platform

Some of these options have different delays that could impact your investigation. We will discuss these in the section corresponding to each cloud service provider.

¹ ["2022 Cloud Security Report"](#)

Some of the main advantages of cloud flow logs are:

- They are collected out-of-band and have no impact on your cloud resources.
- They can be enabled or disabled at any time.
- They are cloud native and don't require any hardware purchase.

We will now review the features and challenges of cloud flow logs for AWS, Azure, and Google Cloud by using a well-known example. Remote Desktop Protocol (RDP) brute force attacks are a threat actor favorite. Both the FBI and CISA have issued numerous alerts about various threat actors launching brute force attacks against RDP.

Keeping this specific threat in mind, we will discuss the value of flow logs when investigating an RDP brute force attack. We will illustrate the differences in flow logs between AWS, Azure, and Google Cloud.

Flow Logs in Amazon Web Services (AWS)

AWS has excellent flow logs with the most flexibility of the three CSPs:

- Flow logs can be configured at the virtual machine level, subnet level, or virtual private cloud (VPC) level.
- The aggregation interval can be configured at either one minute or 10 minutes.
- Flow logs default to version 2 standard format.
- The optional custom format includes numerous other fields.

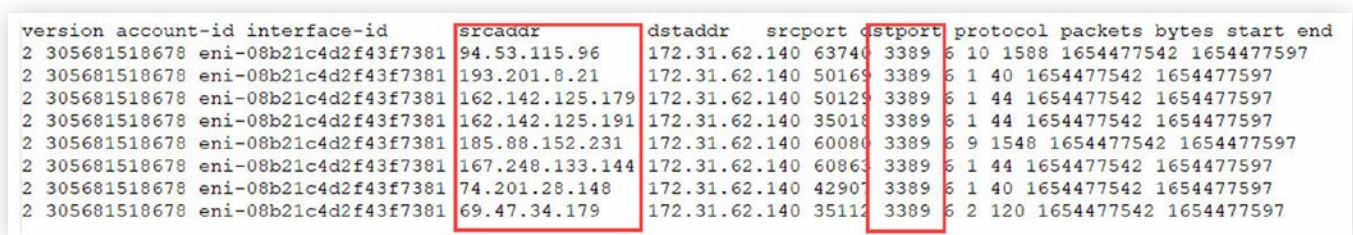
This flexibility gives great control over the granularity of the logs as well as the ability to significantly reduce their size.

Analyzing flow logs is the next challenge. AWS flow logs can be viewed in-cloud through CloudWatch Logs or stored in an S3 bucket. AWS flow logs can also be analyzed using GuardDuty or AWS Detective.

When stored in an S3 bucket, the log files are published at five-minute intervals. To save space, the files are compressed. From there, the log files can be exported to outside systems or searched via AWS Athena.

Athena is an interactive query service that allows you to query structured or unstructured data straight out of S3 buckets. This is a very efficient and cost-effective way to search a large number of logs.

In Figure 1, you can see a small sample of flow logs showing a brute force RDP attack. The key fields are the source IP address, the destination port (showing 3389 for RDP), and the status.



version	account-id	interface-id	srcaddr	dstaddr	srcport	dstport	protocol	packets	bytes	start	end
2	305681518678	eni-08b21c4d2f43f7381	94.53.115.96	172.31.62.140	63740	3389	6	10	1588	1654477542	1654477597
2	305681518678	eni-08b21c4d2f43f7381	193.201.8.21	172.31.62.140	50169	3389	6	1	40	1654477542	1654477597
2	305681518678	eni-08b21c4d2f43f7381	162.142.125.179	172.31.62.140	50129	3389	6	1	44	1654477542	1654477597
2	305681518678	eni-08b21c4d2f43f7381	162.142.125.191	172.31.62.140	35018	3389	6	1	44	1654477542	1654477597
2	305681518678	eni-08b21c4d2f43f7381	185.88.152.231	172.31.62.140	60080	3389	6	9	1548	1654477542	1654477597
2	305681518678	eni-08b21c4d2f43f7381	167.248.133.144	172.31.62.140	60863	3389	6	1	44	1654477542	1654477597
2	305681518678	eni-08b21c4d2f43f7381	74.201.28.148	172.31.62.140	42907	3389	6	1	40	1654477542	1654477597
2	305681518678	eni-08b21c4d2f43f7381	69.47.34.179	172.31.62.140	35112	3389	6	2	120	1654477542	1654477597

Figure 1. AWS Flow Logs Showing RDP Brute Force Attack

Flow Logs in Microsoft Azure

Azure flow logs are more limited compared to AWS. They have the following characteristics:

- Flow logs can only be configured at the Network Security Group (NSG) level.
- A flow log will be created for each network interface in the NSG.
- The aggregation interval is fixed at 1 minute.
- Azure supports only version 1 and version 2 formats.

Version 1 logs both ingress and egress traffic as well as allowed and denied traffic. Version 2 also provides throughput information.

Because the flow logs include flow records for each interface and each rule in the NSG, they can be quite noisy. The alternative is to create multiple NSGs, which may not be practical or even desirable from a network architecture point of view.

Also, given the mandatory one-minute aggregation interval, the result can be huge flow logs which may be hard to ingest in external analysis tools and cost significant money due to egress traffic. However, flow logs are very important in Azure because Azure implements a mandatory network address translation at the virtual network layer. This means that endpoints don't know their internet IP address. Flow logs will provide the needed translation between the internet IP address and the internal IP address of the virtual machine.

Azure flow logs can be analyzed in-cloud via the Network Watcher application (or Azure Sentinel). Alternatively, they can be written to a storage account or streamed via an event hub. When written to a storage account, the flow logs are written only once per hour, therefore introducing additional delays before you can analyze the data.

Figure 2 shows an example of an Azure flow log.

Notice that a flow record is created for each rule within the NSG. The flow record follows the 5-tuple format plus a number of cloud-specific fields that are well documented by Azure. When saved to a storage account, Azure flow logs are saved in **JSON** format. Each entry will show flow records for each rule defined in the NSG.

```
"time": "2022-06-01T00:00:11.32995642",
"systemId": "30eeb776-e720-4963-8e65-0065d9f6cbfb",
"macAddress": "6045BDB22EDD",
"category": "NetworkSecurityGroupFlowEvent",
"resourceId": "/SUBSCRIPTIONS/827D198C-72A9-48C3-BB78-9A1E98A74C64/RESOURCEGROUPS/CLOUDXCHANGE/",
"operationName": "NetworkSecurityGroupFlowEvents",
"properties": {
  "Version": 2,
  "flows": [
    {
      "rule": "DefaultRule_AllowInternetOutBound",
      "flows": [
        {
          "mac": "6045BDB22EDD",
          "flowTuples": ["1654041598,10.1.0.4,20.150.50.228,61503,443,T,O,A,C,8,3160,"]
        }
      ]
    },
    {
      "rule": "DefaultRule_DenyAllInBound",
      "flows": [
        {
          "mac": "6045BDB22EDD",
          "flowTuples": ["1654041562,79.124.62.34,10.1.0.4,46141,50601,T,I,D,B,,,,,"]
        }
      ]
    },
    {
      "rule": "UserRule_RDP",
      "flows": [
        {
          "mac": "6045BDB22EDD",
          "flowTuples": ["1654041552,185.221.134.42,10.1.0.4,63639,3389,T,I,A,B,,,,,"]
        }
      ]
    }
  ]
}
```

Figure 2. Azure Flow Log Example

Flow Logs in Google Cloud

Google Cloud has a very different philosophy when it comes to flow logs. In Google Cloud, flow logs are sampled. At best, only about one out of every 30 packets is captured. This rate can't be adjusted. Additional sampling can also be selected during the configuration, thereby further reducing the number of packets captured as shown in Figure 3.

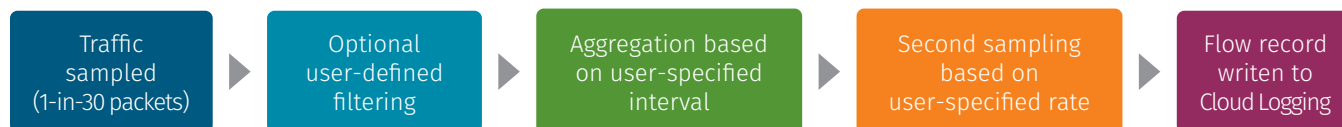


Figure 3. Flow Log Process in GCP

Google Cloud offers six possible aggregation intervals: five seconds, 30 seconds, one minute, five minutes, 10 minutes, and 15 minutes.

The format is consistent with the other cloud providers, and Google Cloud can also record additional cloud-specific fields when the “metadata” option is selected during the configuration.

Given the limitation of Google Cloud flow logs, the alternative is to log firewall rules. In that case, the logging option must be configured for each firewall rule. This log isn't sampled and will be much more complete.

Conclusion: Network Visibility Challenges

In our example, we are looking for an RDP brute force attack, and all three cloud providers' flow logs would provide the information we need for our investigation. However, if we were searching for a command-and-control beacon, the Google Cloud flow logs may miss critical packets due to the mandatory sampling.

When flow logs don't provide sufficient visibility, the following alternatives should be investigated:

- Packet mirroring
- Agent on the endpoint (such as an XDR agent)
- In-cloud firewall logs
- Virtual firewall from vendors such as Juniper, Cisco, Fortinet, Palo Alto, and many others

All in all, once you understand the features and limitations, flow logs are an excellent tool for our investigations.

Weakness #2: Reckless Adoption of Third-Party Code

Brandon Evans

As organizations continue to improve their network security, attackers have focused their attention on applications. Developers, like any other humans, will continue to make mistakes, including those that impact security. Thankfully, the industry is investing billions in Application Security² to varying levels of success. Organizations are improving their security posture through education, static analysis, dynamic analysis, web application firewalls (WAFs), runtime security solutions, and much more. Despite this, AppSec remains a challenge, as demonstrated by Google Project Zero finding a record number of zero-day exploits in 2021.³

To wreak the maximum amount of havoc at the application-level, threat actors perpetrate supply-chain attacks. These attacks, which attempt to compromise an organization by exploiting the third-party software it uses, have tripled in 2021.⁴ The year prior, an attack that Microsoft's president described as the "largest and most sophisticated attack" ever⁵ was performed by weaponizing SolarWinds Orion, a network monitoring application with tens of thousands of customers. Given that a typical organization leverages thousands of third-party applications, they should take the attacker's approach and increase their focus on the supply chain.

This problem becomes exponentially larger when considering third-party code packages. On average, only 10% of the code for an organization's website was created in-house.⁶ The rest of the code is provided through package repositories, such as Maven Central,⁷ the Python Package Index (PyPI),⁸ and **npmjs.com**⁹ (the central repository for Node.js packages). By using a single **npm** package (as shown in Figure 4 on the next page), an organization is adopting countless transitive dependencies,¹⁰ or dependencies of dependencies. Almost 70,000 **npm** packages currently depend on a single package.¹¹ The third most popular package, **request**—on which around 35,000 packages currently depend—itself depends on 46 packages.

2 ["Application Security Market Anticipated to Surpass \\$9,779.8 Million and Grow at a 16.1% CAGR During the 2020-2027 Forecast Timeframe,"](#) GlobeNewswire, May 16, 2022

3 ["Google Project Zero Detects a Record Number of Zero-Day Exploits in 2021,"](#) The Hacker News, April 20, 2022

4 ["Software Supply Chain Attacks Tripled in 2021: Study,"](#) Security Week, January 20, 2022

5 ["SolarWinds hack was 'largest and most sophisticated attack' ever: Microsoft president,"](#) Reuters, February 14, 2021

6 ["Third-Party Risks in the Digital World: Do You Know Who Else Is Coming to the Party?"](#) CPO Magazine, September 6, 2019

7 <https://search.maven.org/>

8 <https://pypi.org/>

9 <https://www.npmjs.com/>

10 ["Understanding the npm dependency model,"](#) Alexis King, August 24, 2016

11 [npm rank, GitHub Gist](#)

There are at least three distinct ways that third-party packages can be exploited:

1. **Package Vulnerability**—A legitimate package contains code written by a benevolent actor that unintentionally introduces a vulnerability. An extremely high-profile example of this was the Log4j zero-day discovered in late 2021.¹²
2. **Repository Compromise**—The repository for a legitimate package is compromised, possibly through a package author's account being compromised, and the attacker publishes a new version of the package containing malware. A popular **npm** package, **ESLint**, was compromised this way in 2018, resulting in **npmjs** revoking all previously issued **npm** tokens.¹³ More recently, three different packages were compromised in Q4 of 2021.¹⁴

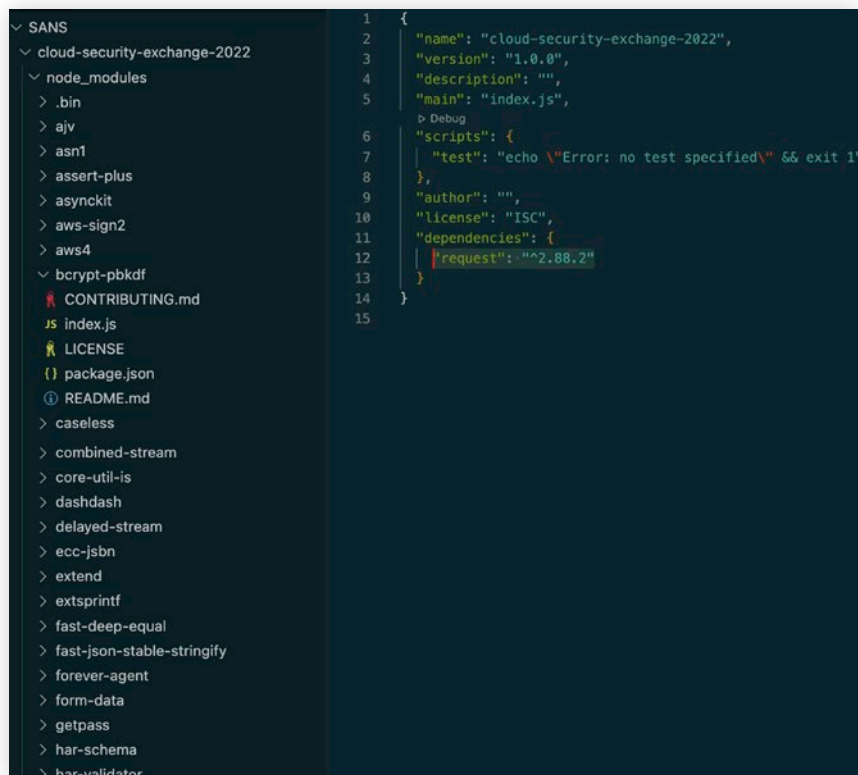


Figure 4. Sample npm Package with One Dependency Next to the Directory Containing Its Many Transitive Dependencies

3. **Protestware**—An author of a package, regardless of whether their original intentions were good, publishes malware to their repository. This method was used at least twice in 2022, once by someone protesting their work being used by big corporations¹⁵ (whose **npmjs.com** and **GitHub** accounts were controversially seized and suspended),¹⁶ and another time by someone targeting machines running their software in Russia or Belarus to protest the Russo-Ukrainian War.¹⁷ While the ethical and philosophical debate these actions have prompted, especially in the latter case, is fascinating, it is out of scope for this paper.

Third-party package exploitation can affect the confidentiality, integrity, and availability for any application, but it has specific implications for the cloud.

¹² "What Is the Log4j Exploit, and What Can You Do to Stay Safe?" PC Magazine, December 23, 2021

¹³ "Postmortem for Malicious Packages Published on July 12th, 2018," ESLint

¹⁴ "Embedded Malware in NPM: Coa, Rc, Ua-parser," FOSSA, November 8, 2021

¹⁵ "An Open-Source Developer Just Caused a Whole Lot of Chaos by Nuking Two of His Own Apps," Gizmodo, January 10, 2022

¹⁶ "Faker.js is now a community controlled project," Hacker News

¹⁷ "BIG sabotage: Famous npm package deletes files to protest Ukraine war," Bleeping Computer, March 17, 2022

Cloud Instance Metadata Services

Each cloud provider has an Instance Metadata Service (IMDS). While it is used by nearly every cloud-based application, developers may not know how it works or even that it exists. It is a service that is only accessible from a cloud compute instance and contains information about that instance. This information includes the identity and access management (IAM) credentials for AWS,¹⁸ Azure,¹⁹ and Google Cloud.²⁰ An entity with access to these credentials will be able to perform the same actions on cloud-managed resources that the compute instance is authorized to perform. Because most applications running on these instances will need access to other cloud services, these permissions can be quite broad. (See Figure 5.)

The AWS IMDS was involved in the famous 2021 Capital One breach. The attacker was able to leverage a WAF, which acted like a proxy server, running on an AWS Elastic Compute Cloud (EC2) instance to access the IMDS and retrieve its IAM credentials.²¹ Using these credentials, the attacker was able to download 100 million Capital One credit card applications.²²

```

ubuntu@sec510-aws:~$ curl "http://169.254.169.254/latest/meta-data/iam/security-credentials/sec510-ec2"
{"Code": "Success",
 "LastUpdated": "2022-06-06T18:58:33Z",
 "Type": "AWS-HMAC",
 "AccessKeyId": "ASIA5C",
 "SecretAccessKey": "nJZWRCAMZN",
 "Token": "IQoJb3JpZ2l1bWVjZWVjEw//////////wEaCXVzLWVhc3Q0tMSJHMEUCIEpmtU/RREhVCCiNtGXF03cMAUmwB8fzh3+CjVRq7LVAiEart3TSB+ase+7ybIEntk72r5KTGBIE2b5LgIQxTnLkwwq9z2WQIuP//////////ARACG40Tg2NzQyNjYwODkiDGvKSMTik3uopT1XLiqvBISF5eMdbbTXLtjh/LPNOGqdxZEnYjGabyYDacK0rdclJmxcCger1AgvhaukBLIG3JMIiOKrh9oIgtGThVhqpGqoqVpXpmxeRBZmsmsGU6S+X0FuhLV4o90qeCE95L6QUiYtOWPbvaZDvvi/9vnyIbGqxbw/zv/yRrHuUqGUJAhvhdgaOW9W8AxyN3mT/6LUXLllyYx4h+xupuPNay5CD/A4V7apMedwmeHr+8uRKDKo9j1BIFM36g+2mVvzpb11JodgsFk7E1iWtejXy960L2NGcR3coyjl3NZb5GyGZEBYnBtyQPkzQFnfBs695STZNY8jj9fJN4m04rek9X7cYoqtCZtwl7pKzg3TmN6xghB8TJahD01TreG6+IomzNUI8KA5hwSSGqEoJmEiRHNNszYxhBhLaM0DJ07ZGCB0

ubuntu@sec510-azure:~$ curl -s "http://169.254.169.254/metadata/identity/oauth2/token?api-version=2018-02-01&resource=https://storage.azure.com/" -H "Metadata: true"
{"access_token": "eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6ImpTMVhvbWU5XRGPfNTJ2YndHdmd2U0U8YVnpNyYsImtpZCI6ImpTMVhvbWU5XRGPfNTJ2YndHdmd2U0U8YVnpNyYyJ3.eyJhdwQiOiJodHRwczovLjN0b3JhZ2UuYXp1cmUuY29tLmYsInZlcyI6Imh0dHBzOi8vc3RzLnRndpbmRvd3MubmV0ZS.MwZmJmZjZyYzRlWUyMjQndUk2M05MGY3LTlyZjZjcm03OTM5OCBiLCJpcyXQioje2NTQ0NDM0TksIm5iZi16MTY1NDU0bWZ0S050S5wZkxhWjJoXnJ0U8NjMwMTK5LjCjhaW8l0tJFmLnUwXobk12VWMyKzJaNEpubzNllyg3cFZZQUFBP5IsInFwcGlkIjoiaWd0Z3ZmFhZWl1bWVjZjY1Ni00Y2Q0LWl4ZTI1ZTEZSE0WFmGM3NmM4IiwiaXN0bWV0Z3R5Y29tIjoiaWwlaW8lIjoiaHR0cHM6Ly9zdHMud2luZG93cy5uZ3QvMzBmYmM3NjgtZTlYNC00NmQwLTkwZjctNjNmNmZ0YnZcSMZiI4LyIsIng1dCI6ImRmMTVlNjAxLWlnMhM2Y2NDVlMS1hOTNmLTlVZGJlYTIZ0WlZ2ZCI6Im5iZi16MTY1NDU0bWZ0S050S5wZkxhWjJoXnJ0U8NjMwMTK5LjCjhaW8l0tJFmLnUwXobk12VWMyKzJaNEpubzNllyg3cFZZQUFBP5IsInFwcGlkIjoiaWd0Z3ZmFhZWl1bWVjZjY1Ni00Y2Q0LWl4ZTI1ZTEZSE0WFmGM3NmM4IiwiaXN0bWV0Z3R5Y29tIjoiaWwlaW8lIjoiaHR0cHM6Ly9zdHMud2luZG93cy5uZ3QvMzBmYmM3NjgtZTlYNC00NmQwLTkwZjctNjNmNmZ0YnZcSMZiI4LyIsIng1dCI6ImRmMTVlNjAxLWlnMhM2Y2NDVlMS1hOTNmLTlVZGJlYTIZ0WlZ2ZCI6Im5iZi16MTY1NDU0bWZ0S050S5wZkxhWjJoXnJ0U8NjMwMTK5LjCjhaW8l0tJFmLnUwXobk12VWMyKzJaNEpubzNllyg3cFZZQUFBP5IsInFwcGlkIjoiaWd0Z3ZmFhZWl1bWVjZjY1Ni00Y2Q0LWl4ZTI1ZTEZSE0WFmGM3NmM4IiwiaXN0bWV0Z3R5Y29tIjoiaWwlaW8lIjoiaHR0cHM6Ly9zdHMud2luZG93cy5uZ3QvMzBmYmM3NjgtZTlYNC00NmQwLTkwZjctNjNmNmZ0YnZcSMZiI4LyIsIng1dCI6ImRmMTVlNjAxLWlnMhM2Y2NDVlMS1hOTNmLTlVZGJlYTIZ0WlZ2ZCI6Im5iZi16MTY1NDU0bWZ0S050S5wZkxhWjJoXnJ0U8NjMwMTK5LjCjhaW8l0tJFmLnUwXobk12VWMyKzJaNEpubzNllyg3cFZZQUFBP5IsInFwcGlkIjoiaWd0Z3ZmFhZWl1bWVjZjY1Ni00Y2Q0LWl4ZTI1ZTEZSE0WFmGM3NmM4IiwiaXN0bWV0Z3R5Y29tIjoiaWwlaW8lIjoiaHR0cHM6Ly9zdHMud2luZG93cy5uZ3QvMzBmYmM3NjgtZTlYNC00NmQwLTkwZjctNjNmNmZ0YnZcSMZiI4LyIsIng1dCI6ImRmMTVlNjAxLWlnMhM2Y2NDVlMS1hOTNmLTlVZGJlYTIZ0WlZ2ZCI6Im5iZi16MTY1NDU0bWZ0S050S5wZkxhWjJoXnJ0U8NjMwMTK5LjCjhaW8l0tJFmLnUwXobk12VWMyKzJaNEpubzNllyg3cFZZQUFBP5IsInFwcGlkIjoiaWd0Z3ZmFhZWl1bWVjZjY1Ni00Y2Q0LWl4ZTI1ZTEZSE0WFmGM3NmM4IiwiaXN0bWV0Z3R5Y29tIjoiaWwlaW8lIjoiaHR0cHM6Ly9zdHMud2luZG93cy5uZ3QvMzBmYmM3NjgtZTlYNC00NmQwLTkwZjctNjNmNmZ0YnZcSMZiI4LyIsIng1dCI6ImRmMTVlNjAxLWlnMhM2Y2NDVlMS1hOTNmLTlVZGJlYTIZ0WlZ2ZCI6Im5iZi16MTY1NDU0bWZ0S050S5wZkxhWjJoXnJ0U8NjMwMTK5LjCjhaW8l0tJFmLnUwXobk12VWMyKzJaNEpubzNllyg3cFZZQUFBP5IsInFwcGlkIjoiaWd0Z3ZmFhZWl1bWVjZjY1Ni00Y2Q0LWl4ZTI1ZTEZSE0WFmGM3NmM4IiwiaXN0bWV0Z3R5Y29tIjoiaWwlaW8lIjoiaHR0cHM6Ly9zdHMud2luZG93cy5uZ3QvMzBmYmM3NjgtZTlYNC00NmQwLTkwZjctNjNmNmZ0YnZcSMZiI4LyIsIng1dCI6ImRmMTVlNjAxLWlnMhM2Y2NDVlMS1hOTNmLTlVZGJlYTIZ0WlZ2ZCI6Im5iZi16MTY1NDU0bWZ0S050S5wZkxhWjJoXnJ0U8NjMwMTK5LjCjhaW8l0tJFmLnUwXobk12VWMyKzJaNEpubzNllyg3cFZZQUFBP5IsInFwcGlkIjoiaWd0Z3ZmFhZWl1bWVjZjY1Ni00Y2Q0LWl4ZTI1ZTEZSE0WFmGM3NmM4IiwiaXN0bWV0Z3R5Y29tIjoiaWwlaW8lIjoiaHR0cHM6Ly9zdHMud2luZG93cy5uZ3QvMzBmYmM3NjgtZTlYNC00NmQwLTkwZjctNjNmNmZ0YnZcSMZiI4LyIsIng1dCI6ImRmMTVlNjAxLWlnMhM2Y2NDVlMS1hOTNmLTlVZGJlYTIZ0WlZ2ZCI6Im5iZi16MTY1NDU0bWZ0S050S5wZkxhWjJoXnJ0U8NjMwMTK5LjCjhaW8l0tJFmLnUwXobk12VWMyKzJaNEpubzNllyg3cFZZQUFBP5IsInFwcGlkIjoiaWd0Z3ZmFhZWl1bWVjZjY1Ni00Y2Q0LWl4ZTI1ZTEZSE0WFmGM3NmM4IiwiaXN0bWV0Z3R5Y29tIjoiaWwlaW8lIjoiaHR0cHM6Ly9zdHMud2luZG93cy5uZ3QvMzBmYmM3NjgtZTlYNC00NmQwLTkwZjctNjNmNmZ0YnZcSMZiI4LyIsIng1dCI6ImRmMTVlNjAxLWlnMhM2Y2NDVlMS1hOTNmLTlVZGJlYTIZ0WlZ2ZCI6Im5iZi16MTY1NDU0bWZ0S050S5wZkxhWjJoXnJ0U8NjMwMTK5LjCjhaW8l0tJFmLnUwXobk12VWMyKzJaNEpubzNllyg3cFZZQUFBP5IsInFwcGlkIjoiaWd0Z3ZmFhZWl1bWVjZjY1Ni00Y2Q0LWl4ZTI1ZTEZSE0WFmGM3NmM4IiwiaXN0bWV0Z3R5Y29tIjoiaWwlaW8lIjoiaHR0cHM6Ly9zdHMud2luZG93cy5uZ3QvMzBmYmM3NjgtZTlYNC00NmQwLTkwZjctNjNmNmZ0YnZcSMZiI4LyIsIng1dCI6ImRmMTVlNjAxLWlnMhM2Y2NDVlMS1hOTNmLTlVZGJlYTIZ0WlZ2ZCI6Im5iZi16MTY1NDU0bWZ0S050S5wZkxhWjJoXnJ0U8NjMwMTK5LjCjhaW8l0tJFmLnUwXobk12VWMyKzJaNEpubzNllyg3cFZZQUFBP5IsInFwcGlkIjoiaWd0Z3ZmFhZWl1bWVjZjY1Ni00Y2Q0LWl4ZTI1ZTEZSE0WFmGM3NmM4IiwiaXN0bWV0Z3R5Y29tIjoiaWwlaW8lIjoiaHR0cHM6Ly9zdHMud2luZG93cy5uZ3QvMzBmYmM3NjgtZTlYNC00NmQwLTkwZjctNjNmNmZ0YnZcSMZiI4LyIsIng1dCI6ImRmMTVlNjAxLWlnMhM2Y2NDVlMS1hOTNmLTlVZGJlYTIZ0WlZ2ZCI6Im5iZi16MTY1NDU0bWZ0S050S5wZkxhWjJoXnJ0U8NjMwMTK5LjCjhaW8l0tJFmLnUwXobk12VWMyKzJaNEpubzNllyg3cFZZQUFBP5IsInFwcGlkIjoiaWd0Z3ZmFhZWl1bWVjZjY1Ni00Y2Q0LWl4ZTI1ZTEZSE0WFmGM3NmM4IiwiaXN0bWV0Z3R5Y29tIjoiaWwlaW8lIjoiaHR0cHM6Ly9zdHMud2luZG93cy5uZ3QvMzBmYmM3NjgtZTlYNC00NmQwLTkwZjctNjNmNmZ0YnZcSMZiI4LyIsIng1dCI6ImRmMTVlNjAxLWlnMhM2Y2NDVlMS1hOTNmLTlVZGJlYTIZ0WlZ2ZCI6Im5iZi16MTY1NDU0bWZ0S050S5wZkxhWjJoXnJ0U8NjMwMTK5LjCjhaW8l0tJFmLnUwXobk12VWMyK
```

Figure 5. Retrieving IAM Credentials Using the IMDS Via a Secure Shell (SSH) Session on AWS, Azure, and Google Cloud Respectively

18 [“IAM roles for Amazon EC2, AWS](#)

19 [“How to use managed identities for Azure resources on an Azure VM to acquire an access token,”](#) Microsoft, June 28, 2022.

20 “Creating and enabling service accounts for instances,” Google Cloud

21 “CLOUD SECURITY - ATTACKING THE METADATA SERVICE,” Puma Security, October 9, 2019

22 “Capital One says data breach affected 100 million credit card applications,” The Washington Post, July 29, 2019.

Seemingly in response to this breach, AWS released the IMDSv2, which seeks to protect the IMDS from open reverse proxies, Server-Side Request Forgery (SSRF) vulnerabilities, layer 3 firewalls, and NATs.²³ This introduced three security enhancements:

- Specific request headers are required. This thwarts most SSRF vulnerabilities as they usually do not allow the attacker to specify headers. This brings AWS's IMDS to parity with Azure and Google Cloud's currently supported IMDS: Azure always required a **Metadata** header, and the only Google Cloud IMDS that did not require the **Metadata-Flavor** header has been deprecated.²⁴
- To get the token used to communicate with the IMDSv2, the client must make a **PUT** request. AWS requires this because their research has found that very few open WAFs support **PUT** requests.
- The IP packets for the response containing this metadata token can have its Time To Live (TTL) set to 1. As such, if an attacker attempts to request the token from an EC2 instance that is operating as an open router, layer 3 firewall, VPN, tunnel, or NAT device, the token's TTL will be reduced to 0 as it is transferred from the IMDS to the VM, preventing it from leaving the instance.

When working with AWS, it is critical that engineers turn off access to the IMDSv1. Although the IMDSv2 is always accessible, by default, the IMDSv1 is accessible as well, eliminating the benefits shown above. It is also important that those working with Azure and Google Cloud understand that the second and third protections listed above do not exist in these providers. More details on the Capital One breach and the various cloud IMDS are provided in a course preview of SANS SEC510: Public Cloud Security: AWS, Azure, and GCP.²⁵

Unfortunately, regardless of the protections AWS's IMDSv2 introduces, by design, it still exposes IAM credentials via the link-local address. (See Figure 6.)

```
ubuntu@sec510-aws:~$ curl -s -H "X-aws-ec2-metadata-token: $(curl -s -X PUT -H 'X-aws-ec2-metadata-token-ttl-seconds: 5' http://169.254.169.254/latest/api/token)" http://169.254.169.254/latest/meta-data/iam/security-credentials/sec510-ec2
{
  "Code" : "Success",
  "LastUpdated" : "2022-06-06T18:59:25Z",
  "Type" : "AWS-HMAC",
  "AccessKeyId" : "ASIA5C",
  "SecretAccessKey" : "BbzQ3ohrBAW",
  "Token" : "IQoJb3JpZ2l1bWVjZW50LWVhc3Qtdm50dGkiDGM07akFOxiFvzfzjSqvBCEHNq2YYdQcqKxSsCfHW954LeY7p1qJdRk/YoXvLMwz82P4HiVpZQIgCSfHU7/Af1FQy/6R9f0nsKZLYdmzHghxLNNBrmDa72oq2wQIxP////////ARACGgw40Tg2NzQyNjYzODkiDGM07akFOxiFvzfzjSqvBCEHNq2YYdQcqKxSsCfHW957T4/2niitpv03PwusXf5hPiCgy6sgS/zeRYkwERvxcM98VmS+grZgnApFjoE6bwBZs0zYU+ZmqD7w4TFtunXSnDgAGyb3DVH2/9xET17k8j/adde3HvLzm5U7ILwWOXy0HaimLSeiNgLzs/VdfNKT9020KvJrLX DqpWLLUiBc66tZWYDU28agamPyJNlQm3YepSMkZAV5ULKZgb5rhJWtEm5D2imeGLjFwZ00VGIjGpWBCgF5C0DP9pdBocyG40bIeCQAvF2ls6Eez+zRZDMK839YUmmM+RWElu0oW7ahc2v7RqZS9tw/x7fF4MxoDFME4TKKH7AjNfmhLU+kEXlscmlb0Cimq8znrg8vjIV1JfTvDIMYq16I7iE2f650YRiD/nkN11Hb8jQkGkq00wEAz9Q0pka6qiWKBJPWNVCS2qsb0d000RVyAIsEXNwT9CJvI1GcpMHRyBTA939rx2i+I/05Pmge
```

Figure 6. Retrieving IAM Credentials Using the AWS IMDSv2

23 "Add defense in depth against open firewalls, reverse proxies, and SSRF vulnerabilities with enhancements to the EC2 Instance Metadata Service," AWS Security Blog, July 27, 2021

24 v0.1 and v1beta1 metadata server endpoints deprecation, Google Cloud

25 "SEC510 Multicloud Security Assessment and Defense," YouTube, June 25, 2020

Developers do not actively think about the IMDS because they interact with it via Software Development Kit (SDK) libraries for each cloud provider: the **aws-sdk**,²⁶ **@azure/identity**,²⁷ and various **@google-cloud** packages. These are official dependencies supported by the cloud providers. Similarly, a supply-chain attack can be used to access and exfiltrate these credentials. To demonstrate this, we have created a small **Node.js** code snippet with a single dependency that retrieves and prints all available cloud IAM credentials.²⁸ (See Figure 7.)

```
printAwsCredentials = async () => {
  try {
    const token = await axios({
      method: 'PUT',
      url: 'http://169.254.169.254/latest/api/token',
      headers: {
        'X-aws-ec2-metadata-token-ttl-seconds': 5
      }
    })

    const iamRole = await axios({
      url: 'http://169.254.169.254/latest/meta-data/iam/security-credentials',
      headers: {
        'X-aws-ec2-metadata-token': token.data
      }
    })

    const credentials = await axios({
      url: `http://169.254.169.254/latest/meta-data/iam/security-credentials/${iamRole.data}`,
      headers: {
        'X-aws-ec2-metadata-token': token.data
      }
    })

    console.log(credentials.data)
  } catch (err) {
    console.error('AWS credentials could not be retrieved')
  }
}
```

Figure 7. The AWS portion of the **Node.js** code snippet for printing cloud credentials is simple, even with IMDSv1 turned off.

Mitigation

Protecting the IMDS from supply-chain attacks is extremely difficult for all three providers. Regardless of how the IMDS is configured, the legitimate applications running in cloud compute infrastructure need to retrieve IAM credentials from the IMDS to interact with cloud services. Still, there are rare cases in which a cloud-based application is self-contained and does not use other cloud services. In these cases, you can turn off the **HTTP** endpoint for the AWS IMDS.²⁹ In Azure, you can get a similar affect by denying access to the IMDS using a Network Security Group (NSG).³⁰ Google Cloud's IMDS is unconditionally accessible, regardless of what firewall rules are configured.³¹ Additionally, we can prevent these credentials from being exfiltrated to the attacker by deploying aggressive egress firewall rules.

In most cases, the application will need some access to cloud services. Still, it will not need access to perform every action on every resource within every service. In the case of Capital One, it is hard to contemplate why that proxy server needed access to credit card applications. Well-crafted IAM policies can give an application the access it needs while following the principle of least privilege. Unfortunately, IAM is complicated and confusing for all three providers.

²⁶ AWS SDK for JavaScript, [npm](https://www.npmjs.com/package/aws-sdk)

²⁷ Azure Identity client library for JavaScript, [npm](https://www.npmjs.com/package/@azure/identity)

²⁸ <https://gist.github.com/BrandonE/5634e521b799b11a485413534fc4c108>

²⁹ Use IMDSv2, AWS

³⁰ Azure platform considerations, Microsoft

³¹ Always allowed traffic, Google Cloud

AWS provides built-in IAM policies called AWS Managed Policies.³² These are usually created for specific services, like the Simple Storage Service (S3). For example, *AmazonS3FullAccess* grants the principal with this policy the ability to create, read, update, and delete all data within S3 for the given AWS account, while *AmazonS3ReadOnlyAccess* only grants the ability to read this data. (See Figure 8.)

At first glance, the customer might think that the latter policy should be used in production environments, especially when it is officially supported by AWS. Unfortunately, it has two critical flaws:

1. An application that needs access to some S3 data, such as files (objects), does not necessarily need access to S3 metadata, including the owner of a file or when it was last updated.
2. This policy, like the vast majority of AWS Managed Policies, has a wide-open *Resource* clause, meaning that the actions granted can be performed on any resource within the service. It is possible that Capital One used this policy and that it played a role in the breach. Perhaps the proxy server needed access to some files in S3, but it certainly did not need access to *all* files in Capital One's AWS account.

For these reasons, we generally recommend against using AWS Managed Policies. Instead, AWS customers should use Customer Managed Policies or Inline Policies. This advice might be difficult to digest for the many seasoned engineers who instinctually want to use the official tools they are given by the cloud provider. Still, the second flaw is inevitable given how AWS policies work: AWS does not understand the customer's naming conventions and access control logic, so there is no meaningful way for them to restrict the *Resource* clause.

This flaw does not exist for the other two providers. Azure provides built-in role definitions,³³ which are a predefined set of actions that a given principal can perform. The customer can then create a role assignment,³⁴ which defines the principal who can perform these actions, the role definition, defining the actions, and,

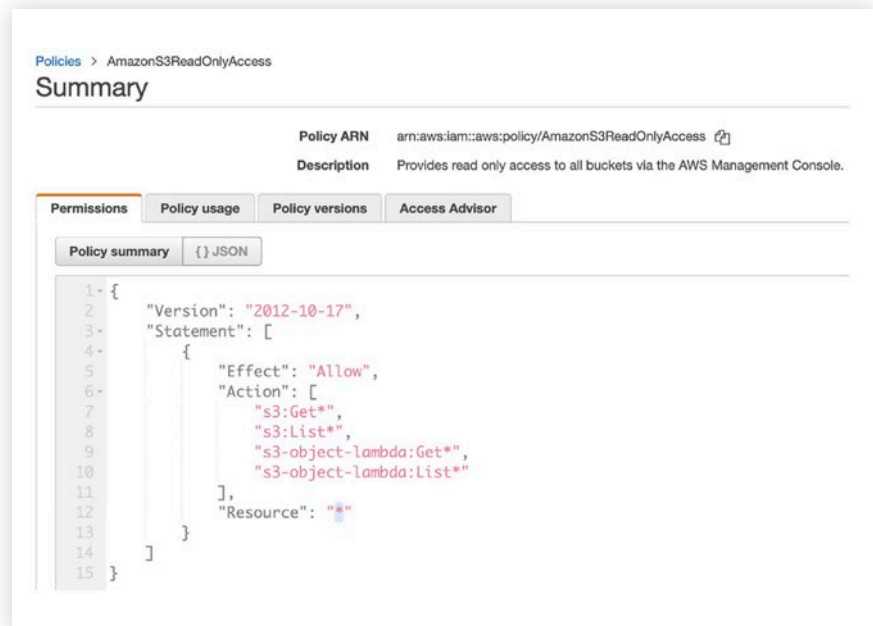


Figure 8. The *AmazonS3ReadOnlyAccess* AWS Managed Policy with the *Resource* Clause Set to "*", Meaning Everything

³² [Managed policies and inline policies, AWS](#)

³³ [Role definition, Microsoft](#)

³⁴ [Role assignments, Microsoft](#)

importantly, the scope,³⁵ which defines the resource or group of resources on which the action can be performed. Effectively, the scope serves the same purpose as AWS's *Resource* clause without being tied to the built-in role definition. Similarly, GCP's roles can be assigned to a member (principal) and bound to a specific resource. In all cases, it is still necessary to ensure that the actions allowed are appropriate for the use case.

GCP has a unique, arguably much worse flaw related to IAM. Many of GCP's default service accounts, including the one used by default for Compute Engine instances, have the Editor role.³⁶ This role gives the member permission to view, update, delete, and modify all resources with the GCP project except IAM permissions and billing details.³⁷ So, if an instance using this default service account is compromised, the attacker will be able to perform more than 5,000 actions on all resources within the GCP project. The attacker can even delete the application instance or create new ones using this role. GCP customers must eliminate the use of the Editor role, as well as the other two "primitive" roles, *Owner* and *Viewer*.

While these mitigations help, the root cause of the problem is a vulnerable or malicious piece of code. Secure dependency management is a broad topic that this paper cannot cover in-depth. Here are some high-level techniques worth considering:

- Manage and exclusively use an internal package repository that contains only vetted packages. Cloud-native solutions for this include AWS CodeArtifact,³⁸ Azure Artifacts,³⁹ and GCP's Artifact **Registry**.⁴⁰
- Use static analysis to find packages, other than the official cloud SDKs, that reference either **169.254.169.254** or **metadata.google.internal**. Note that these values can be obfuscated.
- Deploy Runtime Application Self-Protection (RASP) to prevent malicious operations and use Interactive Application Security Testing (IAST) to detect them. It is sadly quite difficult to distinguish between legitimate and illegitimate interactions with the IMDS.
- Monitor network access and block requests to the IMDS made by packages other than the official cloud SDKs. While this would theoretically prevent the attacker from accessing the IMDS, it is plausible that the attacker can access the credentials used by the official SDKs in-memory.

Conclusion: Reckless Adoption of Third-Party Code

Using third-party code is inevitable and good. Writing all code in-house would require each organization to create its own operating systems, programming languages, and more. While that approach is both extreme and absurd, the industry is arguably trending too far in the other direction, adopting third-party code without any consideration for the security implications. Responsible organizations should seek balance by properly managing dependencies and hardening their cloud environments to minimize the damage rogue code can inflict.

³⁵ [Scope, Microsoft](#)

³⁶ [Compute Engine default service account, Google Cloud](#)

³⁷ [Basic role definitions, Google Cloud](#)

³⁸ [AWS CodeArtifact](#)

³⁹ [Azure Artifacts](#)

⁴⁰ [Artifact Registry](#)

Weakness #3: Cross-Provider Deployment Issues

Aaron Cure

Cross-Provider Configuration Issues

As organizations continue to transition and expand their use of various cloud environments, the use of multiple cloud environments is becoming more common. This distribution of resources comes as the result of many functions. Probably the most significant is pricing. As companies transition from on-premise technologies to software-as-a-service (SaaS) offerings such as Microsoft Office 365, they are often offered deals of service credit or free services. These free services are very enticing to IT departments with budgets that are already stretched thin. However, taking advantage of them can often lead to vulnerabilities.

Many weaknesses in cloud configurations appear when admins and engineers don't recognize the differences in the services offered by the various cloud providers, leading to configuration issues and unintentionally exposed services. Many of these services go by the same or similar names among the providers

Consider the case of AWS Lambda functions.⁴¹ First the Lambda function must be created. At this point the function cannot be called from outside the AWS environment, so it is only accessible by internal objects such as EC2 instances and the Azure Command Line Interface (CLI). To access the function from an external source such as a website or application, the user must create an API Gateway to allow access to the function over HTTP/HTTPS. The API Gateway allows the user to create, publish, monitor, and secure access to the Lambda function.

In contrast, when the user creates the Azure function, the function is assigned a public IP by default, and it is publicly accessible.⁴² For a user who usually works in AWS, the assumption is that once a function is created it is not exposed until it is added to an API Gateway, so it's secure. To make things more confusing, Azure contains an API Gateway also, but it is a traffic load balancer for web applications.

You might be wondering how Google functions work. Google functions can be deployed with an HTTP Trigger to access it. As you might guess, Google also has an API Gateway, and it functions similarly to the one in AWS, however, it is not required to expose Google functions.

Identity and Access Management (IAM) Issues

Often referred to as Authentication (AuthN) and Authorization (AuthZ), IAM controls the overall security of the system by granting or denying access to services, objects, and data. Azure, Google, and AWS all provide variants of role-based access control (RBAC), where users assume roles defined in the system, either as pre-defined collections of privileges or custom roles defined by system administrators. Each of these environments provides similar concepts but applies them in very different ways.

⁴¹ [AWS Lambda functions](#)

⁴² [IP Addresses in Azure](#)

The AWS permissioning⁴³ ecosystem is arguably the most complex, having both account as well as IAM users. Account level users have access to create and administer resources. AWS makes no distinction between the privileges of these Account level users and IAM users, and allows granting access to objects in multiple accounts. To make matters worse, the default user used to create the account, known as the Account Root User, cannot be restricted access from any AWS services, and cannot be denied access from IAM. Additionally, the ability to configure permissioning policies at the user, service, and resource level can create confusion for administrators as to the actual permissions a particular account has to interact with an object. The additional concept of permission boundaries, where a policy can be created that effectively removes permissions from a particular user or service, can further confuse this issue. Often accounts are granted far more broad permissions than required due to the complexity of this system, as illustrated in Figure 9.

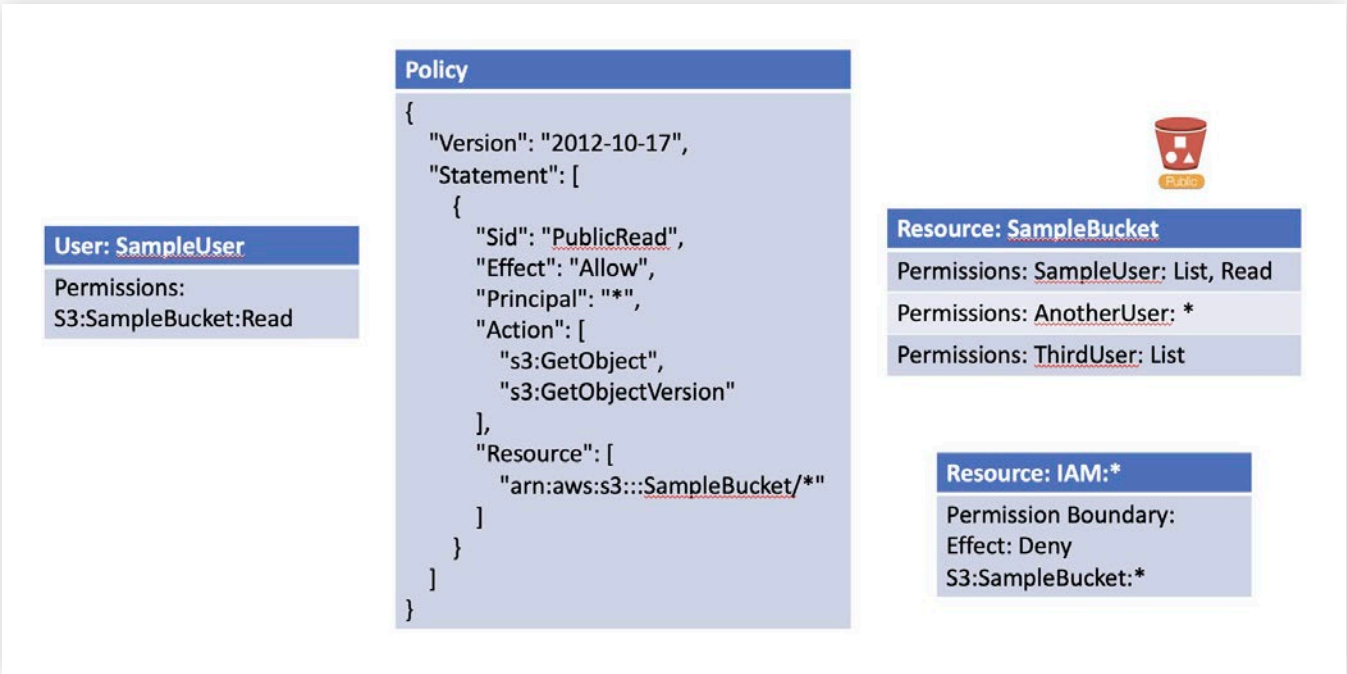


Figure 9. The AWS Permissioning Model with Permissions Set at Various Levels

Azure uses the Active Directory⁴⁴ model for defining IAM and provides predefined roles as well as the ability to create new ones. Roles are separately assigned for users and non-interactive user accounts (service principals).

Google had very basic permissions (Owner, Editor, Viewer) before the release of their IAM system, which still exist. Also, accounts can be either interactive or non-interactive (service accounts), but the same user cannot be defined for both. Individual roles and permissions must be assigned to the interactive and non-interactive user accounts. This can cause confusion around the permissions granted to various users and service accounts.

⁴³ [AWS IAM](#)

⁴⁴ [Microsoft identity platform](#)

Mitigation

While the issues presented here can have serious implications on the security of a system, a few simple mitigations can bolster the security posture of these environments.

One of the most effective controls to put in place is a strong Governance, Risk, and Compliance (GRC) program. Strong documentation and policies around cloud deployment, usage, and security can help users understand the organizational targets and strategies. With the support of management at an appropriate level, along with a strong training program at all levels, the organization can implement a strong, secure cloud-based environment.

Another strong control is the use of infrastructure-as-code (IaC). Using this deployment strategy organizations can enforce standards, ensure secure configuration of created resources, and ensure recoverability of environments in the case of disaster. IaC also allows for the versioning and tracking of these environments as they change, as well as programmatic scanning of the code in these repositories for security issues and exceptions to policies and standards. A strong issue tracking and management program can help to “close the loop” on these issues and ensure they are corrected based on priority and in a reasonable amount of time.

Finally, regular scanning and testing of the cloud environments can prove invaluable to an organization. Scheduled scans can find vulnerabilities in the environment as well as “rogue” services created outside of the normal deployment process. Continuous testing of these systems can also help identify new vulnerabilities and threats to the system. Results of these actions can be injected into the issue tracking system for further analysis and action.

Conclusion: Cross-Provider Deployment Issues

Securing the cloud is a challenging task for security practitioners regardless of the cloud service provider you are hosted on. When companies introduce multi-cloud environments, it becomes a Herculean effort for those charged with defending the cloud workloads. Even with the advancement of cloud security services, attackers are still finding innovative and advanced ways to cause breaches and harm to cloud-hosted companies. It's important for security practitioners to stay abreast of the latest threats and understand how to defend against these threats in any cloud environment.

Conclusion - Chapter II

According to the 2022 Cloud Security Report by Checkpoint and other sources, security incidents are on the rise. In multi-cloud environments, these issues are even more prevalent due to the differences and complexity involved. As the authors have shown, while each major cloud service provider (CSP) lists security as a benefit of the cloud, and the CSPs work hard to uphold their end of the shared responsibility model, the number of security issues and attacks against them continue to rise. Only through continuous education, policy, and improved understanding of the various cloud environments can we make these environments secure and defend our customers and data.

Chapter III

MODERN CLOUD SECURITY: SHARED RESPONSIBILITY TO SHARED FATE?

Written by Dave Shackleford and Anton Chuvakin

SANS



Google Cloud

Introduction

Many organizations are increasing the scope of public cloud deployments steadily, and have been for some time. At the same time, large cloud service providers have expanded the catalog of tools and advanced cloud infrastructure and services, making it easier for a wide variety of IT and business teams to take advantage of cloud scale and capabilities.

Today, it's common for more mature enterprises to use both platform-as-a-service (PaaS) and infrastructure-as-a-service (IaaS) capabilities in single or multiple cloud environments.

With IaaS, a service provider hosts a complete virtual workload network for customers. With IaaS offerings, providers can support multiple distinct virtual machines or interconnected virtual machines with supporting virtual infrastructure. In many environments, providers can also offer truly “private” networks and VMs for customers, where the consumer pays for a physically separate hypervisor server to avoid the security and performance risks inherent in multitenancy. Either way, in the IaaS model consumers and providers share in the security responsibility. Consumers provision VMs and networks and must update and configure these components continuously. Providers support the hypervisor, underlying networks, and management components. A key takeaway is this: Cloud users may be responsible for all patching and configuration in IaaS environments, just as they are in their own data centers. In fact, many IaaS deployments resemble traditional hosting, but with the addition of dynamic provisioning and virtualization for scalability and rapid resource availability.

On the other hand, PaaS services are customized system and application platform stacks offered to customers on demand. These usually take the form of standardized OS platforms or OS platforms with very specific application stacks installed and are often used for application development or customized hosting requirements. In almost every PaaS implementation, the provider offers a standardized set of APIs that are extended to customers. These are then built into applications developed and deployed within the PaaS infrastructure.

The third leg of the cloud stool, software-as-a-service (SaaS) is not covered in this paper.

Overall, the biggest driver for organizations today is the need to innovate more rapidly than in the past. To that end, cloud services have quickly emerged as critical elements of an infrastructure and application development strategy for many, and cloud security is now a critical function for both providers and consumers. As the breadth of deployed services and assets has grown, cloud providers have steadily enhanced computing capabilities and services to make the cloud ecosystem even more tightly coupled, with a staggering array of innovative services and controls that can radically improve how IT infrastructure operates in a combined PaaS and IaaS cloud. With this expanded and evolved environment, however, there are new security requirements and responsibilities that need to be addressed.

Building a Cloud Security Strategy

Let's review several key elements for building the cloud security strategy for your organization.

Threat Modeling for Cloud

The cloud opens up entirely new possibilities for threats—while some threat scenarios go away or lose importance. Our data could potentially be exposed in transit, in storage within the cloud, while applications and systems are operating in IaaS and PaaS environments, and so on.

Recent industry reports cover data breach scenarios and incidents involving cloud assets, and many cases were due to compromised web and application services running in cloud environments and various cloud misconfigurations. A significant number of accidental exposure scenarios were also noted due to misconfiguration within cloud accounts, leading to compromise or data leakage. Attackers are focusing on cloud more and more as many organizations move assets and data there, and the attackers are becoming savvier about how to gain initial entry, compromise accounts, escalate privileges, take advantage of misconfiguration, and much more.

By better understanding actual attacker behaviors in cloud attack scenarios, defenders have a much better chance of detecting breaches before any data or assets are exposed, stopping the attacks earlier, and preventing lasting damage to cloud assets and data.

To better prepare for threats to cloud deployments, organizations are performing more cloud-centric threat modeling than ever. What is threat modeling? In short, threat modeling is the use of abstraction techniques to think about risk. As part of threat modeling exercises, we need to consider:

- Adversaries
- Attack techniques
- Outcomes and risks
- Countermeasures

Threat modeling helps us think more “tactically and practically” about threats and security overall. There are a number of good reasons to spend time threat modeling with a focus specifically on cloud scenarios:

- Threat modeling helps you identify threats your cloud infrastructure and deployed workloads face.
- Threat modeling can help disparate teams to challenge assumptions about architecture, controls, deployment models and processes, and more.
- Threat modeling may help to prioritize security controls and guardrails based on exposure or identified risk.
- Threat modeling can contribute to ongoing risk assessment practices and governance for cloud deployments.

To sum up the major reasons why threat modeling is a good idea, it's really a somewhat defined method to think through what systems look like, and work toward understanding the potential threats and issues the system could face. Moreover, it helps get numerous parties on the same page.

First, we have to define what we are modeling threats for, such as an entire system or a component. For cloud scenarios, we can focus on a full system (a cloud native application deployment or a “lift and shift” workload environment) or we could look at just a component (a serverless workload or management console).

Second, we look at threats—what can go wrong? An account hijack? A vulnerable package exploited in a container image? Third, we look at mitigations and controls that can reduce or eliminate risk. Finally, we perform a final validation that we’ve been thorough and reasonable in our analysis. Given the scale and complexity of cloud infrastructure, focusing on elements of the system and specific trust boundaries probably makes sense versus the “system” as a whole. Focusing on specific attacks and attack modalities may make sense for specific threat models, as well.

Picking a requirements model when starting your threat modeling exercise is important. Prevent/Detect/Respond is common, as is People/Process/Technology. There is no “right” answer here, incidentally. Keep key business drivers, compliance requirements, and use cases or scenarios in mind too. Who will be using the cloud application? How will it be used? What kinds of data are in the application? Although it might seem obvious, make sure you focus on non-requirements too. In other words, don’t make your model too bloated with things that don’t really matter.

Naturally, apart from threat modeling (what can go wrong?), we need to observe various threat activities, and so focus on threat observations (what does go wrong?). Recent threat reports¹ that cover cloud environments help with this aspect of the project.

Many organizations today are leveraging the MITRE ATT&CK™ model to help frame potential cloud attack and threat models. With the growth in public cloud, the ATT&CK program has expanded to incorporate cloud-specific attack patterns. The MITRE Enterprise ATT&CK Cloud Matrix now includes a number of cloud-centric attack patterns and methods that cloud security operations teams can leverage in hunting for threats, monitoring for intrusions, and responding to incidents. These include a number of methods and techniques:

Initial Access

The initial access phase of an attack is when an attacker finds an initial means of ingress into cloud accounts or resources. Common methods to accomplish this include:

- Exploiting public-facing applications
- Discovering and exploiting trusted relationships
- Discovering valid accounts in cloud environments

¹ [“Illicit coin mining, ransomware, APTs target cloud users in first Google Cybersecurity Action Team Threat Horizons report,”](#) November 23, 2021.

Persistence

The persistence phase of an attack is where an attacker seeks to stage a foothold in the victim's environment to ensure they can return at will. Within a newly compromised cloud environment or asset, attackers may use the following tactics:

- Account manipulation to grant later or ongoing access
- Creating new accounts
- Implanting container Images for PaaS Environments
- Creating redundant access with network and identity controls
- Continuing to leverage valid accounts

Privilege Escalation

Escalating privileges is a common goal for many attackers once they've initially breached the environment. In a cloud setting, the most common method for this is to attempt access with or to *valid accounts* within the environment, or to manipulate identity role assignment to then use these valid accounts. A recently launched cloud vulnerability database² is a good resource for this.

Defense Evasion

Once an attacker has gained access to a cloud account or environment, it's paramount that they take active steps to avoid defenses that may be in place, or simply operate more stealthily to avoid detection. Some of the common ways attackers may seek to avoid defenses in a cloud environment include:

- Avoiding detection via redundant access
- Reverting cloud Instances to a previous state
- Establishing a presence in unused/unsupported cloud regions
- Continuing to leverage valid accounts

Credential Access

One of the most common ways an attacker can advance in a focused campaign is by accessing and using varied types of cloud account and asset credentials. There are several well-known types of credential access attackers attempt in the cloud, such as:

- Account manipulation to access credentials
- Querying an identity role with a cloud instance's metadata API
- Discovering credentials in files

² [The Open Cloud Vulnerability & Security Issue Database, Wiz](#)

Discovery

Any attacker seeking to advance further into a compromised cloud environment will eventually start to seek out other resources that may be vulnerable or available to target. As the cloud is a highly interconnected software fabric, there are numerous opportunities/ locations for asset and service discovery such as:

- Cloud service dashboards
- Cloud service discovery (through network visibility, interaction with other services, etc.)
- Network service scanning
- Network share discovery
- Remote system discovery
- System information discovery
- System network connection discovery

Collection

The goal of most attackers is to access and collect data and other assets of value. The top focal areas for attackers are likely to include:

- Data from cloud storage objects (items in a cloud storage node, for example)
- Data from other cloud information repositories (databases or big data warehouses)
- Data from local systems
- Data staged in application scenarios

Exfiltration

Once an attacker has gained access to data, most attack campaigns lead to eventual exfiltration of data to a location under the attacker's control. In the MITRE framework, this is accomplished via the act of *transferring data to a cloud account*. Most savvy attackers will do this gradually and somewhat slowly so as to avoid detection of large, sudden data transfers. In many cases, the data is encrypted or otherwise obfuscated first, as well, to avoid any data loss prevention (DLP) or other content-focused monitoring.

Impact

The final potential "stage" of the MITRE framework is eventual cloud service impact, which is categorized in the current model as *resource hijacking*.

As a part of threat modeling that includes actors, infrastructure, and mitigation techniques and controls, another critical factor must be evaluated, as well: the shared responsibility model. While not a new concept (we've shared security responsibilities with most outsourcing arrangements for many years), the nature of shared security responsibilities has changed with the advent of cloud. Security for cloud workloads and service configuration is the responsibility of the customer, and this includes data protection, identity and access management, OS configuration, network security (access controls), and encryption. Cloud providers are responsible for the underlying pieces of the infrastructure, including the compute elements, storage infrastructure, databases, and networking.

All cloud providers are wholly responsible for physical security of their data center environments. Cloud providers are also now responsible for data center disaster recovery planning, business continuity, and legal and personnel requirements that pertain to security of their operating environments. Cloud customers will still need to plan for their own disaster recovery and continuity processes, however. They also need to closely collaborate with their cloud provider(s) on detection and response ([example model](#)).³

One major trend that has occurred in the past several years is the gradual move to the “shared fate model”⁴ of shared responsibility in the cloud. Leading providers are now hosting IoT platforms, payment processing for global financial organizations, and healthcare patient data processing and application integration. For example, the automotive industry can now take advantage of Google Cloud’s Connected Car Telemetry Platform⁵ to collect and coordinate data from self-driving vehicles and those with telemetry reporting for speed, location, camera footage, and more. Organizations are increasingly dependent on cloud infrastructure, and cloud providers potentially have more exposure due to the proliferation of critical assets running in their environments. To that end, there’s a definitive need to ensure all parties are clearly informed as to where responsibility lies, and cloud providers are more transparent than ever about what they’re doing to defend their environments and protect workloads and data.

To address the types of threats we face in the cloud, cloud service providers have increasingly offered a growing array of capable security controls that tenants can employ for prevention, detection, and response. For many organizations, the challenge may be where to start, and the following three core pillars of cloud security should help security and cloud engineering teams make architecture and design decisions that greatly improve the security posture of any deployment.



Security Pillar #1 Identity and Access Management

In the Cloud Security Alliance’s [Top Threats to Cloud research released at the RSA conference in 2022](#), they revealed that the top concern for cloud consumers is insufficient identity, credential, access, and key management. Identity and access management (IAM) is really the practice of defining who needs access to what and then controlling the entire life cycle of user and access management across resources. There are a number of specific areas within IAM, and any IAM program may include the following areas:

- **User management**—Defines and dictates the governance of identities
- **Authentication**—Determines who an entity is based on one or multiple factors
- **Authorization**—Determines what rights and privileges an entity has based on policies and roles

3 [“Who Does What In Cloud Threat Detection?”](#) Medium, February 8, 2022,

4 [“Demystifying ‘shared Fate’ - A New Approach To Understand Cybersecurity,”](#) Forbes, April 19, 2022,

5 [“Harness vehicle data to drive insights for production innovation,”](#) 2020,

- **Access management**—The definition of access control policies that enforce entity access to resources
- **Provisioning**—A set of processes and tools that provision identities and identity attributes within authorization guidelines, as well as deprovisioning those same identities
- **Monitoring and auditing**—Monitoring, auditing, and reporting services related to entity access and activities; primarily logging and event creation in many environments.

One of the most significant cloud-driven shifts that has occurred in identity management is the advent of “machine identities” versus traditional “human identities.” Machine identities, sometimes referred to as “non-people identities,” are digital identities associated with computing resources that have access rights and control over other identities or compute services and resources in a public cloud environment. These can often be broken down into four distinct categories:

- **Compute resources**—In the cloud, any compute resources, such as compute instances, cloud functions, and containers, can represent machine identities.
- **DevOps and engineering**—These include shared testing accounts, service accounts, and other technical accounts used for programmatic actions and deployments, which are often associated with elements of the DevOps pipeline like build tools, QA and testing platforms, and others.
- **Automation**—Deployment roles and account definitions, particularly for infrastructure as code (IaC) template deployments, are common in more automated cloud scenarios.
- **Cloud services identities**—Many distinct public cloud applications require identities that allow them to interact with other services and resources in the cloud environment.

Human identities, on the other hand, represent traditional interactive users and groups with defined sets of privileges for performing a variety of actions. These may be members of your own organization or external users with whom you collaborate, who interact with Google Cloud Platform (GCP) resources via a range of different cloud interfaces. For security teams, it’s important to know the use cases and context for any types of identities defined in the cloud. Human identities are usually reserved for administrators and hands-on engineers (and potentially some end users that need to interact with specific services), while machine identities are in place to facilitate cloud service and resource interactions and deployments.

Wherever possible, it’s critical to build strong authentication and authorization access strategies for Google Cloud access using multifactor authentication (MFA), and there are many mature options for organizations today, including:

- Push notifications
- Google Authenticator
- Phishing-resistant Titan Security Keys,⁶ or using your Android or iOS device as a security key.⁷

⁶ [Titan Security Keys](#)

⁷ [Use your phone’s built-in security key](#)

Mature organizations will centralize identity and access wherever possible. This is a sound practice to pursue for a number of reasons. First, when looking to manage provisioning and deprovisioning of identity accounts, it's ideal to perform account creation and revocation within a single, centralized identity store like Active Directory. This can help to prevent local accounts from being created uniquely in cloud services, application stacks, or workloads, many of which may be forgotten or overlooked over time. Second, once a central identity repository is defined, this is then ideally synchronized with a unified authentication and authorization portal (often referred to as Single Sign-On, or SSO) that can validate a user identity via credentials of various types, and then facilitate access to additional resources from there. This approach is usually accomplished through federation standards for authorization, which most leading cloud services support readily. Another benefit of a centralized identity approach is reduced operational overhead and security governance.

The first element of any identity strategy in Google Cloud is the GCP Identity and Access Management (IAM) service. IAM users are associated with credentials for making API calls to interact with cloud services and only exist within the cloud environment itself. If you link your directory services like Active Directory to the cloud, you can leverage in-house existing users and map them to IAM groups and roles, but a standalone user created within the cloud is only useful there. New IAM users have no permissions (an implicit “deny all” policy). This is a good thing, as permissions must be explicitly granted. This can also help with the common problem of over-allocating privileges to users and groups in the environment.

IAM users can represent any asset/resource—an IAM user is a simple identity with associated permissions. This means that IAM users can be enabled for application access to Google resources too, not just as actual interactive user accounts. Once service-oriented users are created, they should be placed in defined groups, if warranted. Permissions and privileges can be directly assigned to users (not advised) or groups (better to manage and maintain). For service interactions within the environment, however, cloud security teams should focus on defining specific roles (groups of permissions), and then granting roles to authenticated principals. A principal can be a Google Account (for end users), a service account (for applications and compute workloads), a Google group, or a Google Workspace account or Cloud Identity domain that can access a resource.

Highly granular permissions models can be easily assigned through role definition, and the Google Cloud Policy Analyzer can easily assess any defined roles, principals, and groups to assist with privilege minimization, or alert an organization when permissions are excessive based on usage analysis.

For larger enterprises, centralizing identity services across multi-account environments will be important. Within the Google cloud, the centralized identity service for managing users, groups, policies, and role assignments across numerous accounts is known as the Organization Policy Service. With this service, you can create policies that restrict and control how IAM is applied across a set of accounts and service implementations. The Organization Policy Service can actually control the entire account, group, and role life cycle with regard to policy application, and can do so for accounts that need to interact or have some relationship. A basic example of how this could be practical would be governing business unit (BU) account use (as they may have totally different requirements, but still need some central control or billing) as well as governing and controlling DevOps and other team accounts (for the same reasons).

The Organization Policy Service is the lynchpin of a multi-account blast radius limitation strategy in GCP. Creating a centralized policy model with clear constraints can allow security administrators to create different and “least privilege” policies for the appropriate accounts and assign them and/or revoke them easily.



Security Pillar #2 Data Security

The CSA's Top Threats to Cloud research also found consumers concerned with accidental cloud data disclosure/disclosure and cloud storage data exfiltration. Clearly, a sound data security strategy for the cloud is important, and there are many things a mature organization needs to consider to adequately protect data in the cloud today. This ranges from implementation of various controls to governance and process adaptation within cloud engineering and operations teams. Let's explore some of the types of controls and focal areas most organizations rely on today for data security in Google Cloud.

Encryption

Undoubtedly, one of the most important security controls for data protection in the cloud is encryption. Cloud providers have the capability to implement encryption at scale fairly easily, and accordingly, all data at rest within Google Cloud is encrypted by default. For some organizations, this automatic encryption will prove sufficient for protecting data at rest for both workloads shifted into the cloud and new cloud native application stacks. For others, customer-generated encryption keys will be preferred or required for compliance, and this is also easily managed through Cloud Key Management System (Cloud KMS),⁸ Google's key management service that offers software-based encryption or hardware-backed hardware security modules (HSMs), easily imported keys from on-premises cryptographic systems, simple rotation and policy control over keys, and much more.

Google has added additional encryption solutions for data processing in Compute Engine and BigQuery, as well. The Cloud External Key Manager⁹ service provides segregation for encryption keys with external key storage in a third-party environment for GCP data, and encryption policies for access and use still apply here.

Secrets Management

For most organizations, managing sensitive secrets (including encryption keys, API keys, passwords and other credentials, connection strings, and more) has proven immensely challenging within a diverse technology ecosystem. The following are examples of traditional best practices for secrets management that should be taken into account when designing a secrets management strategy:

- No secret should be written to disk in cleartext or transmitted over a network in cleartext, and any tools you utilize should ensure this is not done.
- All secret life-cycle and access events should be recorded in an incorruptible audit log, which should ideally be sent to a remote syslog-compatible event store.
- Secret distribution should be coordinated by an authoritative delegator, such as a container/service scheduler, or by working in a close trust relationship with the scheduler.

⁸ [Cloud Key Management, Google Cloud](#)

⁹ [Cloud External Key Manager, Google Cloud](#)

- Operator access to secret cleartext should be limited with least privilege access models to secret data and values.
- Secret versioning should be easier to accomplish than revealing cleartext whenever possible.
- All infrastructure components related to secret management and distribution should be mutually authenticated using keys and/or certificates.
- Secure system configuration should be implemented for any platform managing secrets or storing secret data.
- The attachment of a secret to a service or container should be protected by strong access control mechanisms. Role-based access control is preferred.

Within Google Cloud, all of these considerations are embedded into cloud services used to build applications (workloads, storage, orchestration, etc.) and centralized within Secret Manager, a service that manages secret versioning, access controls, life cycle and rotation, and audit trails simply and centrally. In addition, secrets can be automatically detected within Google Cloud DLP (covered next) with a catalogue of detectors that identify and tag them for protection within Secret Manager.¹⁰

Data Loss Prevention (DLP)

To track and control sensitive data, many organizations turn to data loss prevention (DLP) tools and services, which can be notoriously difficult to implement and maintain. Within a cloud environment, discovering, classifying, and tracking data requires deep integration with a cloud provider's storage infrastructure. Fortunately, Google Cloud DLP¹¹ provides the following benefits:

- All data discovery is automated for BigQuery databases.¹² This is important, as large scale data storage environments like BigQuery are constantly changing and may be difficult to monitor without automation.
- DLP can be implemented across a variety of storage types, including BigQuery, Cloud Storage, and Datastore. Providing DLP across a range of different data storage types in a cloud environment is critical for security professionals needing comprehensive data protection coverage.
- Google Cloud DLP can help organizations flexibly protect data with policies to classify, mask, tokenize, and transform data as desired.

Cloud DLP, for many organizations, may prove simpler to implement and maintain than traditional on-premises DLP options, and at lower cost. Even organizations that traditionally didn't implement DLP due to cost and complexity can now affordably discover, track, and protect data with this advanced capability.

¹⁰ [Secret Manager, Google Cloud](#)

¹¹ [Cloud Data Loss Prevention](#)

¹² ["Automatic data risk management for BigQuery using DLP," Google Cloud, April 14, 2022](#)

Data Catalog

One of the most challenging data protection controls we've sought for many years is easy, flexible data classification. In a traditional data center, discovering and classifying data has been daunting at best, and almost impossible at worst. In the cloud, all data nodes are linked to the cloud fabric, and data is easily discoverable and identifiable by various stakeholders. [Google Data Catalog](#) is a service that allows different teams to easily identify data nodes and types, and then apply metadata labels that can be used to track and control data access and use. Data Catalog also integrates with Cloud DLP to identify and control data with DLP policies, as well.

Confidential Computing

One of the more innovative data security technologies available in Google Cloud is Confidential Computing, which provides real-time encryption for virtual machines that can be used and accessed while still encrypted. Confidential VMs¹³ provide automatic encryption for data that is still accessed and processed, improving efficiency and simplicity for security teams trying to control data access permissions models. Confidential Computing pairs perfectly¹⁴ with Google Cloud External Key Manager to enable the cloud experience where even the cloud provider cannot see the data being processed on its systems and cannot get to such data at all. This is useful for data sovereignty use cases.



Security Pillar #3 Visibility

The third critical pillar of cloud security is visibility, with emphasis on logging, event management, and automation through guardrails. Visibility goes beyond traditional system and network visibility, but must cover applications, systems, networking, and their configurations in the cloud.

This concept is also applicable to the following as part of a building a comprehensive strategy:

- **Control plane visibility**—A critical category of visibility is the cloud environment itself: the control plane. In addition to extensive logging of all activity within the cloud, a number of new services are available to continuously monitor cloud accounts and infrastructure for best practices configuration and security controls status.
- **Network visibility**—The types of controls often used to achieve network visibility include network firewalls, network intrusion detection and prevention, load balancers, proxy tools, and the collecting of network flow data (behavioral). Cloud-native access controls (e.g., security groups and firewall services) and monitoring capabilities can also be used to monitor and track network events and behaviors.

¹³ [Confidential Computing concepts](#)

¹⁴ ["Trust Google Cloud more with ubiquitous data encryption."](#) Google Cloud, October 19, 2021,

- **Application visibility**—Application visibility relies on tracking events and behaviors at scale as workloads communicate within the cloud environment as a whole, in addition to the local application logs on individual systems and containers. Developing true application visibility often relies on feeding events into event management and SIEM platforms, which have also been well adapted into cloud environments (often via API integration).
- **Serverless/container visibility**—Logs and events generated by PaaS solutions and applications should be automatically collected and sent to a central monitoring platform.
- **Database/storage visibility**—Many cloud deployments employ a wide variety of storage types, including block storage, blob-type storage, databases, and more. Major cloud storage often includes various forms of logging, as well as a range of additional configuration controls that can be monitored and observed.

Fortunately, Google Cloud provides a way for organizations to centrally enable cloud security guardrails that improve visibility with Security Command Center.¹⁵ This highly automated service provides an enormous range of security functionality, including asset discovery and inventory, threat detection and prevention through configuration assessment and integrated threat intelligence, and continuous monitoring and data discovery and protection.

To build a monitoring model for the cloud, security teams should implement baseline monitoring and logging for virtual machine instances, containers, and other workloads. Baseline monitoring can be accomplished by:

- Gathering and processing logs made available via the cloud service provider APIs
- Gathering specific workload logs
- Performing image and instance integrity validation to help ensure the environment is sound
- Gathering any available networking logs (e.g., VPC flow logs)

Control plane logging in Google Cloud is handled with Cloud Logging,¹⁶ a logging service that records any API calls made within the GCP fabric. The service captures an extensive amount of data that security professionals need to achieve observability, including the following:

- Identity of API caller
- Time of API call
- Source IP address of API caller
- API request parameters
- Any response elements returned by GCP services

The Google Logs Router checks each log event against defined inclusion and exclusion filters to determine which log entries to discard, which to ingest, and which to include in exports. Organizations can also integrate Google Cloud Audit Logs events for Compute Engine, Google Cloud networking, Cloud Storage, Cloud IAM, and other services into Security Command Center.

¹⁵ [Security Command Center](#)

¹⁶ [Cloud Logging](#)

Google's security team continuously monitors threats and attacks against cloud resources, and the Event Threat Detection service directly benefits from this monitoring and analysis. Threat Detection is a native service for Security Command Center that continuously monitors your organization and identifies threats within your systems in near-real time. Event Threat Detection is regularly updated with new detectors to identify emerging threats at scale.

VPC Flow Logs are incredibly simple to enable for any VPCs and include important details about observed network traffic patterns and behavior within each VPC environment. Flow log records include values for the different components of IP flow, including the source, destination, and protocol in observed traffic. VPC Flow Logs can help security teams in a number of ways, such as troubleshooting and analyzing security group rules, monitoring workload network traffic patterns, and determining the direction and patterns of traffic to and from cloud network interfaces.

Another capability many network security teams have sought in the cloud is full network packet capture controls. Packet Mirroring permits network traffic to be copied from any compatible workload in a VPC to a suitable destination. Many network brokering and intrusion detection solutions can leverage this mirroring capability to pull traffic from instances in VPCs, enabling security operations teams to perform deep packet inspection, network forensics, and even selective packet filtering.

Today, more cloud engineering and security teams are actively looking for opportunities to automate processes that often take up too much time by highly skilled analysts, as well as those that require lots of repetition (and may provide little value in investigations). Common activities that many teams consider for automation include the following:

- **Identifying and correlating alerts**—Many analysts spend inordinate amounts of time wading through repetitive alerts and alarms from many log and event sources, as well as piecing together correlation strategies for similar events.
- **Identifying and suppressing false positives**—Identifying false positives can often be streamlined or automated using cloud-integrated event management and response automation tools.
- **Initial investigation and threat hunting**—Analysts need to quickly find evidence of compromise or unusual activity, and often need to do so at scale.
- **Opening and updating incident tickets/cases**—Due to improved integration with ticketing systems, event management and monitoring tools used by response teams can often automatically generate tickets to the right team members and update these as evidence comes in.
- **Producing reports and metrics**—Once evidence has been collected and cases are underway or resolved, generating reports and metrics automatically can save quite a bit of time.

There are many additional areas where automation can help incident responders, especially in forensic evidence gathering, threat hunting, and even automated quarantine or remediation activities on workloads and other assets. Deciding what triggers to implement and what actions to take are really the most time-consuming aspects of building a semi-automated or automated detection and response framework in the cloud. Do you focus on user actions, specific events generated by instances or storage objects, failure events, and/or others? Spending time learning about cloud environment behaviors and working to better understand “normal” patterns of use may be invaluable here. Organizations looking to build automation playbooks that hinge on observability and then integrate to security automation response actions should focus on the following practices:

- **Enable and collect all vital observability event data.** This includes cloud control plane logs, workload logs (if separate), network flow data, and outputs from cloud-enabled and cloud-native assessment/guardrail services.
- **Ensure all event data is collected and aggregated.** Use cloud-accessible storage or services that easily facilitate triggered automation sequences.
- **Determine unusual behaviors and observable events of interest.** These will be somewhat subjective, but should usually start with obvious patterns or indicators of account and/or privilege abuse, access failures, suspicious volumes of questionable/unidentified network traffic, cloud service cost increases that are unusual, and so on. Common starting points include any login activity to cloud management consoles, any changes or attempted changes to important cloud objects and data, and any creation, deletion, or modification of credentials or cryptographic keys.
- **Determine appropriate response actions to pursue.** These range from relatively passive actions, such as generating a ticket or assigning a metadata tag to assets, to more intrusive or disruptive response actions like revoking access keys, isolating/quarantining Compute Engine workloads, and shutting down services.
- **Build triggers to perform your desired response actions.** Based on pattern matching, using security alerts or a SIEM, you then trigger some sort of follow-up action.

Conclusion - Chapter III

As the types of cloud services available grow, and organizations begin to deploy large PaaS and IaaS environments that employ numerous interconnected services, the range of cloud security controls needed and the number of surfaces to protect also gets larger. To keep up with the array of different cloud services in use, security teams will need to learn and use more advanced controls and develop more dynamic and continuous processes for evaluating security conditions in their environments.

First, cloud security teams should conduct regular threat modeling exercises to evaluate controls, architecture, and other risks in their deployments. Cloud security should focus on three primary categories—identity and access management, data security, and visibility related to events and activity within the cloud. Retooling and updating security processes will be critical for organizations that want to ensure their cloud security programs are progressive and dynamic, keeping pace with the rapid rate of development and change occurring in cloud today.

Finally, cloud security is getting better¹⁷ all the time. The key advantage of public cloud is that cloud providers leave in a virtuous circle of security improvements. Even if cloud users do nothing, their security will improve over time. Cloud—when properly architected—truly does have security magic.

¹⁷ ["Megatrends drive cloud adoption—and improve security for all,"](#) Google Cloud, January 11, 2022,

SANS



Access more free educational
content from SANS at:

[SANS Reading Room](#)

Checkout upcoming
and on demand webcasts:

[SANS Webcasts](#)

**CLOUD
SECURITY:**
Making Cloud
Environments
a Safer Place



Google Cloud



Microsoft