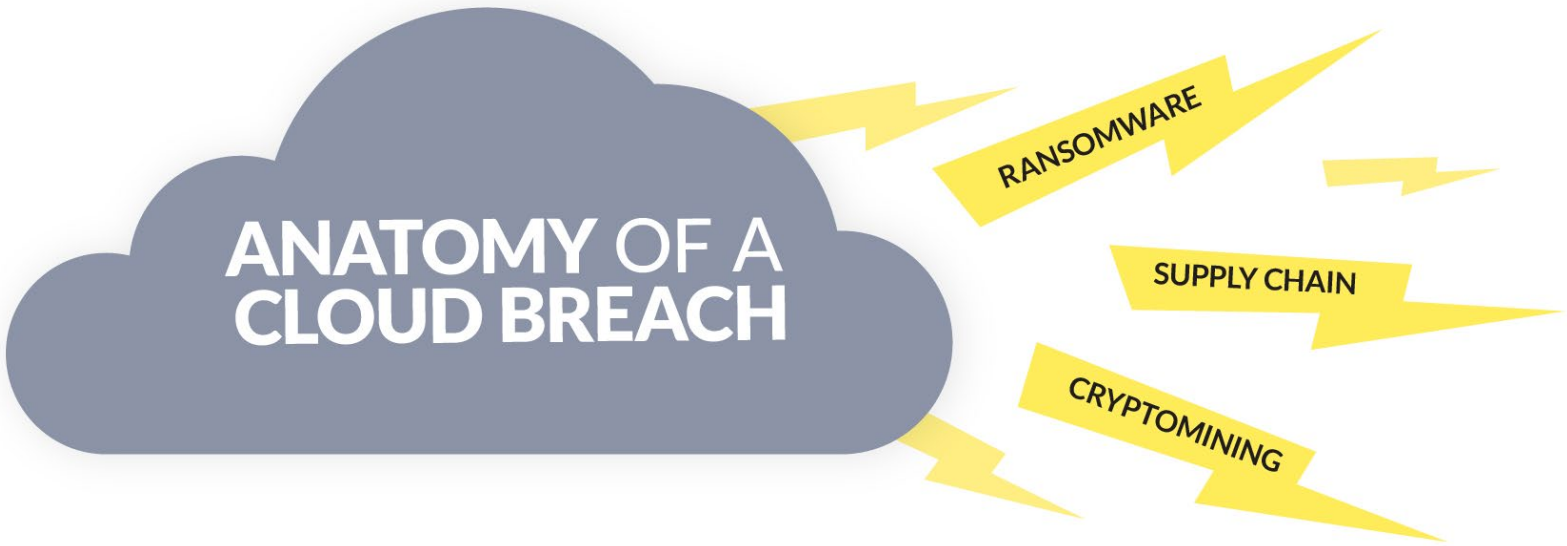
Anatomy of Cloud Attacks

Contents



As more assets and infrastructure are migrated to the cloud, attackers increase their offensive capabilities and operations in the cloud as well. Although the goals of these operations are often the same as they were for on-premises environments, the tactics, techniques, and procedures (TTPs) involved have changed. It is important to understand **how these attacks work in a cloud environment** so that you can employ proper defenses.

In order to demonstrate this change in TTPs, we will dissect offensive operations and show what they might look like in the cloud. The goal is to provide a general understanding of cloud attack patterns instead of a step-by-step guide to conducting such attacks.

This article will cover three common cloud attack patterns:

- **Ransomware:** The attacker hijacks your data, demands a reward for recovering it, and potentially further blackmails you to not leak it.
- **Supply chain compromise:** A successful attack in any link of the supply can propagate the compromised code or component downstream, completely unnoticed, and cause mayhem across different stages.
- **Malware cryptomining:** The attacker's main goal is to run malicious software in your infrastructure and use your resources for mining cryptocurrency.

There will be common themes seen throughout the article when it comes to the cloud because its inherent complexity breeds increased likelihood of misconfigurations.

CHAPTER 1

Ransomware Attack

Typical ransomware approach in the on-prem world:

Ransomware attacks use cryptography to encrypt your data and demand a certain amount of money, usually in cryptocurrency to avoid tracking, to receive the key to decrypt it.

The typical scenario starts with a phishing email sent to a user in the form of an attachment, which exploits a vulnerability or masquerades as a benign file when it really contains malware. A user executing this malware unknowingly starts the infection and propagates it throughout the whole company.

Attackers may extort victims by leaking data if payment is not made within a certain time. It is strongly recommended not to give in and make use of backups to restore the data, but this is not always possible. One of the most famous ransomware attacks was Wannacry, which affected large companies around the world.

Ransomware attacks are usually associated with on-premise environments, but they **are certainly possible in the cloud**. We will be focusing on the "cloud-native" aspect rather than on-premise analogs, such as compute instances or other "lift and shift" type resources. S3 and other cloud-native storage services are also vulnerable to ransomware attacks. While we will use S3 as an example, other cloud storage services are vulnerable in the same basic ways.

If an S3 bucket is properly permissioned and protected, it will help mitigate impact from opportunistic attackers. However, S3 buckets are used by many services and are often inappropriately permissioned, so their discovery and compromise by an attacker could come from many different places. As we will see, these other entry points may affect the security of the data within the bucket.

Step 1: Gaining access to data

You can gain access to an S3 bucket in a number of ways. The easiest technique for an attacker is to target a public bucket with an overly permissive policy. However, even a more locked-down policy can be vulnerable to a ransomware attack. Improper permissions assigned to cloud storage services, data objects within them, or related identity and access management (IAM) policies are often the root cause of related security incidents. For this scenario, we will assume an attacker gained access to an EC2 compute instance, which has an assigned IAM policy that provides access to an S3 bucket.

Cloud Service Providers (CSP), such as AWS, offer a cloud metadata service, typically powered by APIs, to provide information about and enable operations on cloud resources. Once an attacker has access to a compute resource, they may be able to query the metadata service APIs. With a simple HTTP request to the API, the attacker can retrieve the IAM policy access key and secret key. With that information, they can then access the S3 bucket associated with the IAM account used by that compute resource.

Step 2: Script ransomware-like operations

With the stolen IAM policy above, we will assume that it has Write permissions on the S3 bucket. At this point, an attacker can script their operations to overwrite every file in the bucket with an encrypted version. Or, if they are feeling especially destructive and the Delete permission is assigned, they can just steal the data and delete the files in the S3 bucket.

A clever attacker may leverage the Key Management Service to encrypt and effectively restrict access to the data as well. They can even go so far as to restrict policy modification for any other KMS actions by using policy conditions, such as only allowing a specific Source IP address (their own attacker-controlled address) to make modifications. KMS services and encryption services can be used as defensive controls, but as ransomware has shown us, encryption can be used offensively as well as defensively.

Step 3: Advance the Ransomware attack

Cloud storage services have several options which would seemingly make these types of attacks more difficult, such as Versioning and KMS. However, these may not offer the bulletproof protection that we assume. Versioning keeps previous copies of the objects stored in the S3 bucket so that they can be restored. This is a great thing to have when you suffer any sort of data corruption or loss, such as from ransomware.

The problem is these old versions can be deleted with some simple API calls, such as “s3:DeleteObjectVersion,” which can be used to remove all previous versions of a file. This technique is similar to a typical Windows-based ransomware attack where an attacker will remove any shadow volume instances.

Use of KMS and encryption services offers another layer of protection which only allows users with a key to decrypt and view data. However, if the associated IAM policy is too permissive, an attacker may be able to call the relevant KMS API that deletes the key being used to protect an organization's data. Depending on the CSP, it may take a few days for this action to actually take place since purging of encryption keys can be destructive to a business, and CSPs may delay such actions as a safety measure. Nonetheless, the possibility of this attack technique being used stresses the importance of cloud log monitoring for fast attack detection and response. Once a key is purged, no one will be able to decrypt the bucket or data in the bucket anymore, and it's effectively lost.

Summary and best practices

Ransomware is still possible in the cloud but it looks different than a typical on-premises attack. Instead of abusing endpoints and local directory services and IAM systems, attackers will take advantage of permissive policies and the standard security capabilities that CSPs provide. Policies get very complicated in most organizations' implementations. This is due to the extreme amount of flexibility that policies provide but also other factors, like architectural complexity and personas of users that must operate on or consume cloud resources. This reality is why permissioning missteps and cloud misconfigurations are often the root cause of cloud security incidents.

The answer to the ransomware scenario described here is enforcing the Principle of Least Privilege, where policies only grant the permissions to user or machine identities that are absolutely necessary. It is easy to say this is the solution, but implementing and managing appropriate permissions over time is more challenging. That is where Cloud Infrastructure Entitlements Management (CIEM) tools come into play. They can give you visibility into policies and permissions across your environment and help remediate any offending policies.

CSPs may also offer additional protections to prevent data from being changed. For example, S3 Object Lock can be used to prevent data from being modified or deleted over a set period of time. This feature is often used for purposes such as legal holds, though it may not be useful in cases where data in S3 buckets should normally be changeable. Cold storage, such as Glacier, may also be useful in preventing data loss in the event of a ransomware attack since it can enable an organization to restore from clean backup.

CHAPTER 2

Supply Chain Compromise

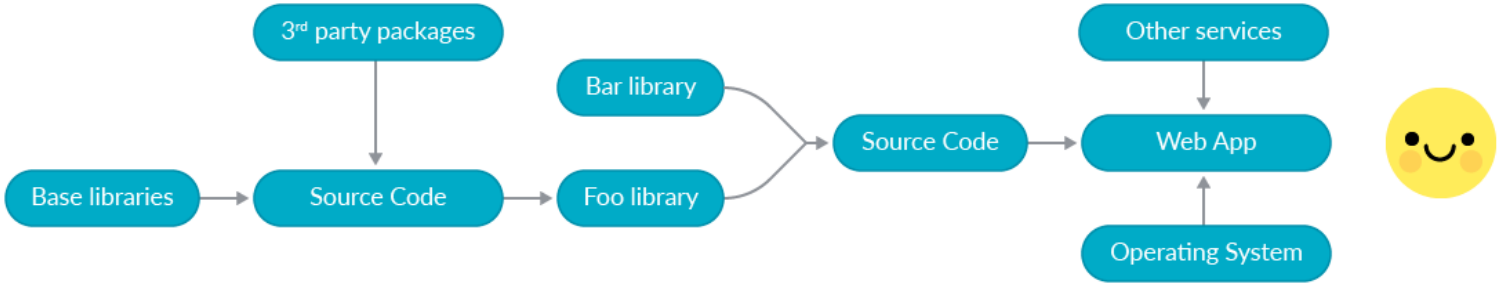
A digital supply chain attack occurs when a threat actor directly or indirectly compromises an organization's software, systems, or resources with the goal of compromising the delivered resource that is delivered to customers. Indirect digital supply chain attacks occur when attackers target partners or suppliers, or componentry that is used as part of a victim organization's technology stack. Often, malware is added to the deployed code to compromise customers and their users. Malicious code can be distributed through new installs or through patches and hotfixes. Tampered code not only hurts customers and users of the software, but also the reputation of the organization. It's also important to note that "code" can take many forms and result in different artifacts once it is built and delivered, including applications, infrastructure, or other specific objects within cloud environments.

A loss of trust with customers and partners can lead to diminished sales, decreases in usage activity, and even the creation of competitive alternatives.

One of the most well-known attacks was the one targeting SolarWinds. The attackers infiltrated SolarWinds' network and managed to inject malicious software into their build process. This malware (related to APT29, Nobelium) was bundled as part of Orion (a Network Management System) product updates. As part of the build process, the artifact was digitally signed and then downloaded by hundreds of customers. The ultimate targets of this malicious campaign were not just SolarWinds or its customers, but a large number of US government and private organizations. The total number of companies impacted by the attack was in the tens of thousands.

There are several phases of a digital supply chain that can be susceptible to attack: Design, Development, Distribution/Deployment, Maintenance, and finally Disposal. Here we will walk through a realistic example of a Distribution/Deployment phase attack by compromising the Continuous Integration/Continuous Delivery (CI/CD) pipeline. A CI/CD pipeline introduces automation and monitoring to improve the development and delivery process, particularly at the integration, test, and deployment phases.

CI/CD pipelines and the systems that power those pipelines are favorite targets of threat actors due to the amount of stored credentials and access granted to perform automation tasks.



Step 1: Gain foothold within a CI/CD pipeline

First, an attacker needs to gain a foothold in an environment with a valid account. This is accomplished in several ways, depending on the security posture of an environment. An attacker could find API keys and passwords leaked in source code, leverage spearphishing attacks, compromise systems with services susceptible to remote vulnerability attacks, and many more. In one instance, a configuration conflict between authorization plugins would mistakenly allow anyone to self-register and gain access to otherwise hardened CI/CD systems or services.

It is also possible to attack an organization by first compromising the repository of a software dependency. When code references dependencies from external sources, they are trusting that such imports do not also load malicious code. A threat actor could compromise the code repository of a software dependency and inject malicious code. Such malicious code could be used to leak credentials of an organization's environment or attack software users. Referenced dependencies also regularly include dependencies of their own, or transitive dependencies, which worsens the problem.

In another instance, a development service provider was compromised by an unknown threat actor. This compromise allowed the threat actor access to the code repositories of thousands of organizations. This breach highlighted the need for organizations to continually monitor their code repositories for unauthorized changes and respond to alerts from CI/CD pipelines.

Step 2: Escalate access

A threat actor may get lucky with a breached account and have all the authorization they will need. For instance, a user that is authorized to push code changes to protected repository branches is a prime target. Programs that support CI/CD services, such as GitLab Runners, can also be targets if configured with excessive permissions. Otherwise, the threat actor will need to move laterally to an authorized account by finding credentials from the inside.

Docker containers using the '--privileged' flag, which negates security isolation provided by Docker, can be abused to attack the host running those unsafe containers. By running a nested container or by using a vulnerability to escape a container, a threat actor can gain superuser privileges on the host. Once access to the host is established, the threat actor "lands and expands" where they can then search for other resources, including configuration files and scripts, which may contain stored credentials.

If the threat actor can run CI/CD jobs, they may also be able to create new jobs or modify existing ones in order to execute arbitrary or malicious custom code. Environment variables are often used to insecurely store credentials for CI/CD tasks. Such custom code can be used to dump credentials held in environment variables, or even cause a crash that can leak credentials from within an error report.

Step 3: Inject malicious code

Once a threat actor has all the access needed, they can deploy their malicious code in a number of ways. They could commit the code to the main repository and let the CI/CD services deploy their changes. However, that may be too noticeable and raise suspicion, so the threat actor could instead modify CI/CD pipeline definitions to insert their malicious code as part of the build process. The repository would go untampered and the malicious code would make its way into the final production delivery to users. All users of the compromised release would be susceptible to further attacks.

Profit!

Digital supply chains and CI/CD pipelines result in complex processes and environments with significant attack surfaces. This complexity requires methodical and comprehensive reviews to secure. Development teams may lack the expertise and depth to secure their entire digital supply chain on their own, or the organization may not even have control over all elements when considering all its partners and suppliers.

Cloud and development environments need to be secured with strong access control policies that limit access based on least privilege principles, rotate authentication keys and passwords regularly, and require other factors of authentication, such as 2FA challenges or geo-based app authenticator services. Code repositories can be secured by adding checks in the CI/CD process to validate build integrity and authenticity through verifications of digital signatures and hashes. Code and releases should be routinely tested for security flaws and unusual behavior. Systems need to be maintained with daily security updates. Cloud environment, CI/CD, and code repositories should be monitored by an operations center or response team for unusual activity.

CHAPTER 3

Malicious Cryptomining

One of the main, if not the main, malware goals nowadays is motivated by financial reasons. Cloud environments were already the perfect breeding ground for software-based cryptocurrency miners thanks to the infinite, elastic computing resources available for all users. With the boom of cryptocurrency adoption and price increases, this technique has also become more lucrative for attackers. Attackers piggyback on an organization's cloud compute to run cryptocurrency miners without incurring any of the related cloud expenses.

As we know, cryptocurrency miners use the victim's computing power to mine cryptocurrencies on non-custodial wallets. This trend is increasing yearly and largely through targeting container-based cloud resources. Criminals are increasingly exploiting cloud and containerized resources, obscuring their own containers which operate cryptocurrency mining, in a multitude of instances.

Step 1: Exploiting public-facing workloads

Getting access to a public-facing workload might sound difficult to achieve due to the various security mechanisms that might be in place. However, we have seen many cases in the past where public-facing applications are exploited due to unpatched services, misconfigurations, or inappropriate permissions. Several malware families have implemented methods to automate the deployment of mining malware once they gain access, for example:

- Sysrv-Hello Botnet
- Muhstik Botnet
- RinBot

Exploiting web application vulnerabilities like remote code executions and SQL Injection or OS-level infrastructure vulnerabilities make it possible for an attacker to gain access, escalate privileges, and take full control of workloads.

As container and container platform adoption have increased in recent years, we've seen more containers unsecurely deployed and then used by attackers for their purposes.

Exploiting public-facing workloads, applications, and services is one of the most frequently used ways to obtain access and gain the initial foothold in a cloud environment and start using the resources for the attacker's benefit.

Step 2: Pivoting within the cloud tenant

We shouldn't think that once the attackers have control over the compromised host or are confined in a container they have reached their final goal. Of course they can use the instance as they prefer, like using the resources for cryptocurrency mining and escalating permissions inside the host. However, smart attackers know that this might be the starting point for something bigger as part of a more complex attack chain.

Since most modern infrastructure is cloud hosted, there might be an opportunity for the attacker to use other cloud resources for their goals and aim higher than compromising just one container instance or a single K8s pod. In cloud environments, there are internal services available for instances that might be used to extract further information.

An example often used is the instance metadata service which is accessible within the cloud tenant, allowing users to retrieve information regarding their resources in the tenant. However, this information can be also used by attackers for their own gain. In this case, along with the instance details, it's also possible to retrieve the roles attached to the instances and, more importantly, extract temporary credentials related to these roles.

Once an attacker owns these temporary credentials, they can directly access the cloud environment and use the cloud metadata API to retrieve further information and find new attack vectors to proceed with the attacks.

Step 3: Final jackpot!

The attacker is now logged into the cloud environment and they can start evaluating the permissions obtained from the instances and check if there are any particular privileges on specific services which might permit modifications or the spawning of new resources into the cloud account.

As we know, it's not that difficult to find privilege misconfiguration in cloud environments. Due to the fine granularity of permissions available in cloud environments, applying the **least privileges concept** is hard to achieve and hard to maintain operationally. Just a single misconfigured privilege could lead to an attacker escalating the privileges inside the environment, allowing the attacker to use the resources in that tenant to achieve the goals.

In this way, if an attacker is able to find a misconfiguration, such as a permissive AssumeRole setting, they may be able to leverage compute or managed container services to start mining cryptocurrency. The compromised organization would be footing the bill for their malicious and profitable activity.

Advice and prevention

In the attack path that we have seen, there are various mitigations that can be put in place to prevent this kind of misuse.

The first obvious point is patching well known vulnerabilities, especially on public-facing applications. As we know, those are the most targeted to attackers and the first part or our cloud infrastructure that we need to keep safe. Patching and maintaining services and applications, especially those that are Internet-facing, is vital to avoiding these types of attacks.

Another recommended mitigation is to restrict access to instance metadata services, which, as we have seen, is a rich source of useful information that might be misused by attackers.

In Summary: Is Cloud More Secure Than On-Premises?

Attackers' motives and goals have not changed much over the years despite major changes in the technology we use. However, they have had to adjust their tactics to use the cloud, and as has been shown, they can still accomplish their goals. In some ways the cloud has made certain tasks more difficult due to its architecture and built-in protections, like Ransomware. In others, like Supply Chain attacks, cloud has made it easier for an attacker to succeed due to taking the process out of the perimeter and making it more open. Defenders must understand the benefits and risks cloud technologies introduce and adapt accordingly.

www.sysdig.com