Google Cloud

Threat Horizons

Cloud Threat Intelligence
February 2022. Issue 2 (revised)

Office of the CISO

Providing threat intelligence to those in the Cloud

Part of offering a secure cloud computing platform is providing cloud customers with cybersecurity threat intelligence so they can better configure their environments and defenses in ways most specific to their needs. Google's Office of the CISO is pleased to release Issue 2 of the Threat Horizons report. The report is based on threat intelligence observations from the Threat Analysis Group (TAG), Google Cloud Threat Intelligence for Chronicle, Trust and Safety, and other internal teams. It provides actionable intelligence that enables organizations to ensure their cloud environments are best protected against ever evolving threats.

Summary of Observations

Adversaries and researchers alike are continuing to scour the web looking for vulnerable instances of Log4j. As a result, service providers have been and continue to work with their cloud customers to ensure the infrastructure is secure as well as check the status of customer-installed tools and third-party dependencies in their environments to see if they are a ected. While adversaries continue to knock on this door, observations have shown that they are opting to use known open-source tools, native Cloud services, and previously established domains for persistence in their attacks.

Google Cloud was proactive in assuring the security of the platform and we partnered extensively with customers to provide tooling and support to help them mitigate and remediate Log4j risks. This included threat intelligence-driven updates to Cloud Armor, Cloud IDS, Security Command Center, and Event/Container Threat Detection and we were pleased to see many customers proactively respond to this event and use these capabilities to defend themselves and their customers. Google Cloud customers also used a variety of scanners and fuzzers to help identify the presence of Log4j in Cloud instances and their dependencies.

01	02	03	04	05
Vulnerable instances of Apache Log4j still sought by attackers	**Adversaries using an open-source platform to maintain network persistence**	**Cloud Shell used for reverse SSH tunneling after initial compromise vector**	**Domain previously identified by TAG used in ongoing attacks against researchers**	**Lessons Learned**
Despite patching, attackers are continuing to scan public-facing sites for those that have not been properly patched.	After compromising a network, attackers are using Sliver to maintain access.	Threat actors abuse Cloud Shell to initiate reverse SSH tunnels from compromised environments to avoid detection.	A North Korean sponsored actor appears to have re-used a domain identified by TAG in an intrusion campaign involving manipulated IDA Pro.	As Google Cloud partners with its customers in a shared fate security model, some valuable trends and lessons-learned emerge.

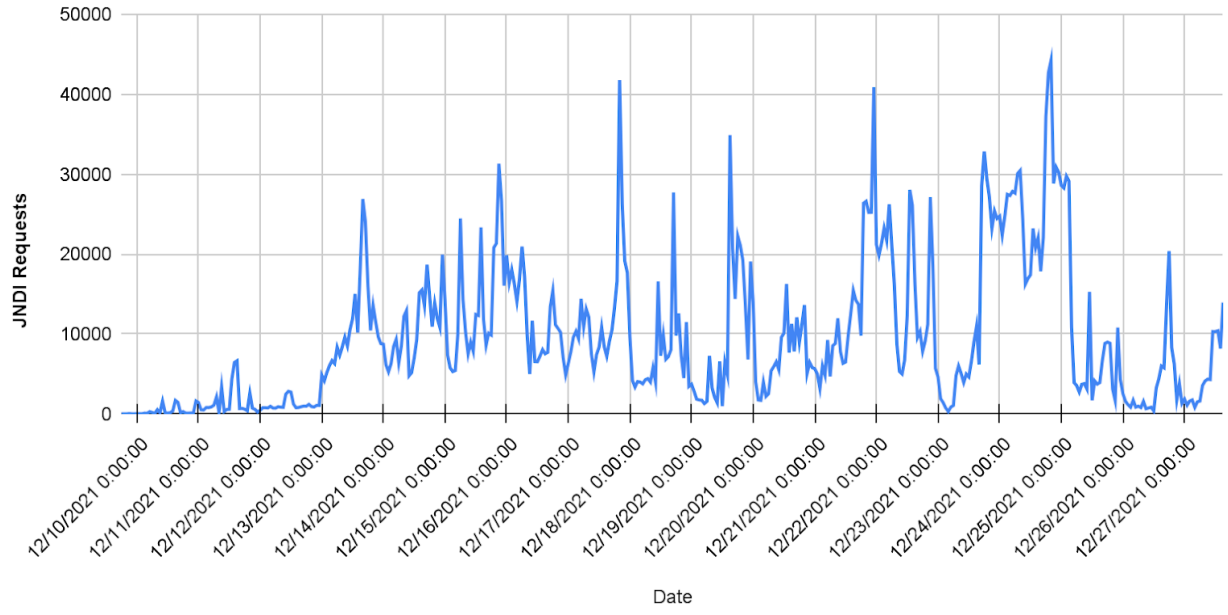
Detailed Observations

Vulnerable instances of Apache Log4j 2 still sought by attackers

Threat Description / TTPs

Shortly after the December 9, 2021 public disclosure of the Apache Log4j vulnerability (with a criticality rating of 10/10), Google Cloud Threat Intelligence and Google Trust and Safety started observing scanning for vulnerable instances of Log4j on the same day. The activity increased over the following weeks as shown in Figure 1.

Figure 1: Log4j Attack Signal Detected



Google Cloud is seeing ongoing scans for vulnerable Log4j instances by attackers and security researchers and we recommend continued vigilance to ensure available patches and mitigations are applied and sustained. During the month following the vulnerability's disclosure there was extensive scanning across the Internet. Google Cloud and other providers had a unique vantage point over this and used this to good effect to help customers identify vulnerabilities as well as watch for the evolution of attempted exploitation to rapidly assure mitigations were effective for cloud infrastructure and customers. Google Cloud is continuing to see scanning (400K times a day) and expects similar, if not more scanning levels against all providers, and so we recommend continued vigilance in ensuring patching is effective. Threat actors are predominantly seeking to target ports 80 and 443 with scanners sending payloads to many other ports with attack payloads largely using Lightweight Directory Access Protocol (LDAP) servers listening mostly on TCP ports 389 and 1389. Additionally, threat actors have been continually refining ways of obfuscating the log4j format string, starting with "jndi:ldap://" and moving more difficult to parse strings.

Strategic Significance

Attackers will continue scanning for vulnerable instances of Log4j as long as exploitation is easy to perform and vulnerable Log4j instances are found. Google Cloud has seen many customers proactively scan in-depth for affected packages that may embed Log4j. Observations are revealing that mitigation strategies for software dependencies will need to be mapped prior to patching as exploits were occurring within hours of the original vulnerability notification. Google Cloud recommends continued vigilance in looking for such dependency risks including evaluating the software supply chain, engaging in an ongoing dialog with software vendors and using tools such as deps.dev.

Google Cloud Specific Mitigations

Those operating in Google Cloud can continue to protect themselves by using various tools and techniques as described in the following:

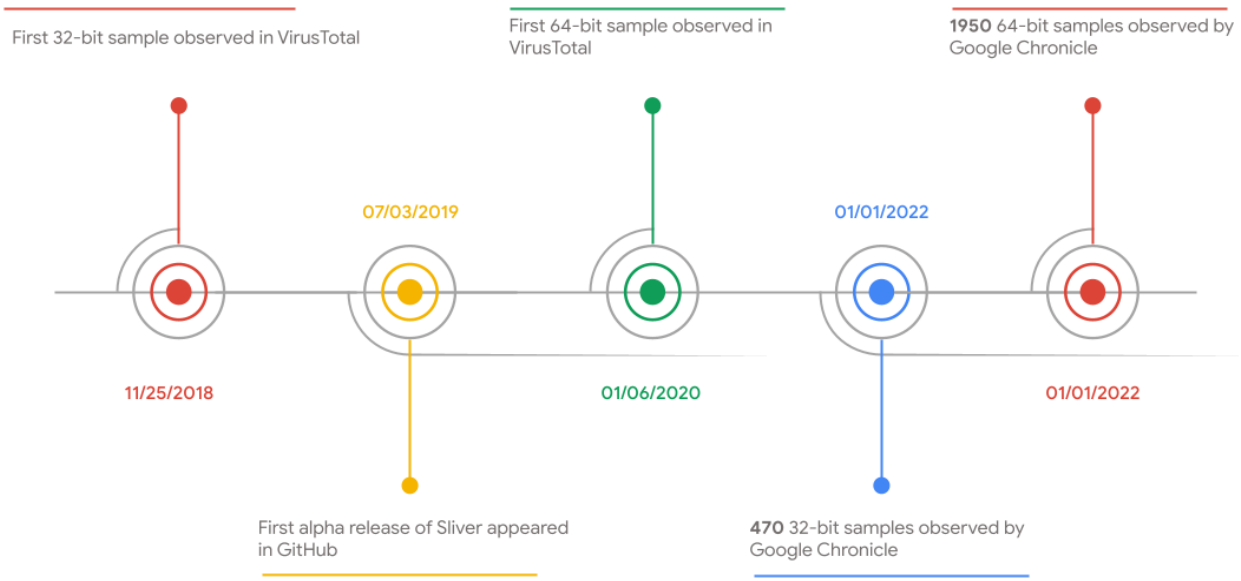
- The recommendations described in the blog post on investigating and responding to the Apache "Log4j 2" vulnerability.
- Google Cloud customers have Implemented Cloud Armor to mitigate threats for applications or services behind external HTTP(S) load balancers. Cloud Armor customers can now deploy a new pre-configured rule that will help detect and, optionally, block commonly attempted exploits of the Log4j vulnerabilities.
- The Java scanning feature of Google Cloud On-Demand Scanning can be used to help identify Linux-based container images that use an impacted version of Log4j. Identifying vulnerable images can help prevent them from being deployed in production. This functionality can be used either on a locally stored container image or one stored in Artifact Registry or Google Container Registry.
- Chronicle is Google's threat hunting tool that provides extended event collection across Google Cloud and in on-premise environments. Those using Chronicle for log ingestion/SIEM and have historical event data can detect and respond to Apache Log4j.
- Google Cloud customers deployed Cloud IDS, which was updated to help detect common types of Log4j exploit attempts. These were enabled by default for any existing or newly added deployments of Cloud IDS as described in this blog.
- Google Cloud customers used Security Command Center (SCC) Premium to actively scan DNS calls to known malicious sites associated with Log4j vulnerability abuse.
- Google Cloud customers leveraged Cloud Logging to detect Log4j 2 vulnerability exploits by querying logs that have already been ingested and are also within the user-specified retention limits. Enable logging across your environment to expand visibility and querying across the environment.
- The Google Open Source Security Team partners with the security company Code Intelligence to provide continuous fuzzing for Log4j as part of OSS-Fuzz.
- Google Cloud customers leveraged resources from the Open Source Security Foundation to secure their environments to include joining workgroups and taking training to better develop software with well crafted requirements, design, and re-use to limit exposures.

Adversaries using an open-source platform to maintain network persistence

Threat Description / TTPs

Google Cloud Threat Intelligence has observed from across the Internet that Sliver is being used by adversaries post initial compromise in attempts to ensure they maintain access to networks. Recent observations as listed in Figure 2, show attackers using Sliver as the second stage malware to compromise organizations, moving the possibility of attackers using Sliver for malicious actions from a hypothetical to a reality. Sliver is an open-source, cross-platform adversary emulation framework that allows adversaries to deploy and control implants on victims' Windows, Mac, or Linux computers from a central coordinating server. The framework is an open-source alternative to Cobalt Strike, is freely available on Github, and is included in the package repository for Kali Linux, a popular open source Debian-based Linux distribution.

Figure 2: Sliver Timeline



Strategic Significance

Adversaries without significant sophistication now have easy access to an adversary emulation framework which gives them tools similar to Cobalt Strike. Additionally, Sliver has very little presence on a victim's computer outside of its implants' sizes, which makes detecting the behavioral aspects of Sliver difficult. Network communication is heavily encrypted and the transports are designed to blend into normal traffic. Cloud customers would likely need to perform specific threat hunting to identify this malicious activity as it may not be obvious when configured.

Network detection is possible whenever an implant uses the HTTP transport to observe the initialization session, such as with the "out-of-the-box" implementation of Sliver; however, when customized, it will be more difficult to detect. Similarly, it is possible to detect the server's periodic polling and response communication from a Sliver implant.

Google Cloud Specific Mitigations

Chronicle has developed YARA rules to detect “out-of-the-box” Sliver implants for any operating system on i386 or x64 CPUs, which can be used by Google Cloud customers to protect themselves. These YARA rules can be downloaded and compatible with open source tools and platforms. These are:

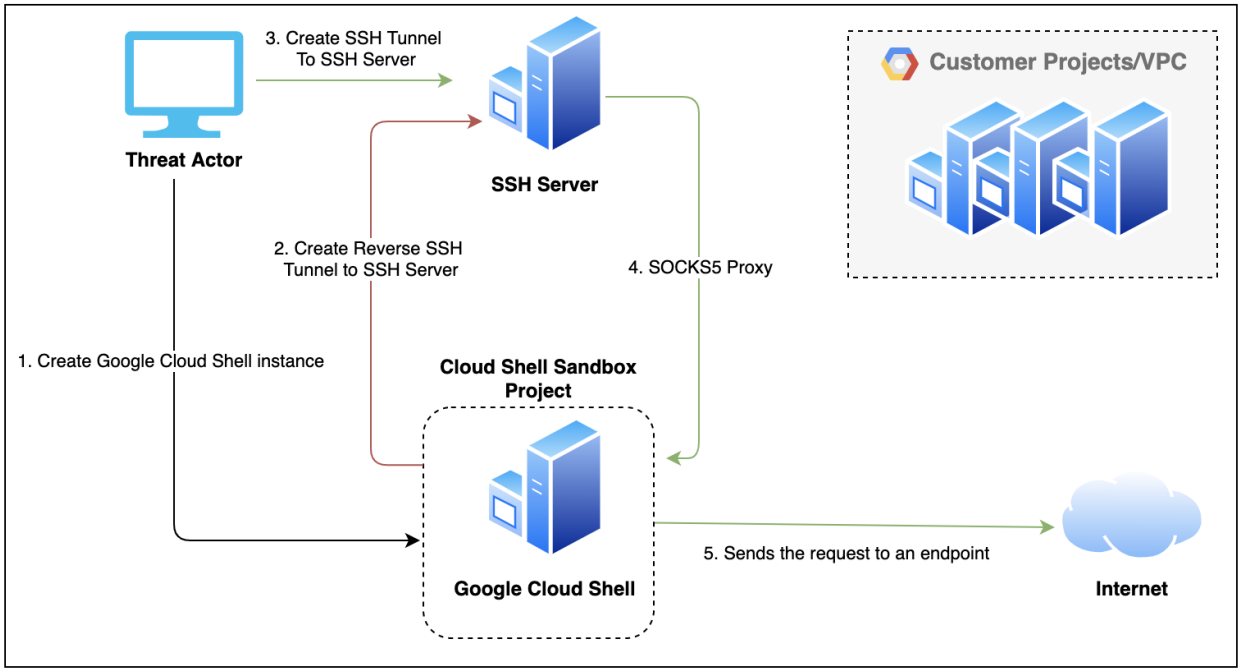
1. https://github.com/chronicle/detection-rules/tree/main/yara/malware/Sliver__Implant_32bit.yara
2. https://github.com/chronicle/detection-rules/tree/main/yara/malware/Sliver__Implant_64bit.yara

Cloud Shell used for reverse SSH tunneling after initial compromise vector (revised on 3/15)

Threat Description / TTPs

Already compromised systems use outbound Cloud Shell to establish reverse SSH tunneling. Additionally, Google Trust and Safety has observed activity pertaining to SSH tunnels available for rent and trade, which supports the argument that this abuse is being monetized. Google Cloud allows users to create a free sandboxed environment of the Linux system (“Cloud Shell”) outside customer projects or networks, with the ability to execute commands using a provided shell. While Cloud Shell inbound connections are restricted, the outbound connections are not. The availability of outbound connections to conduct a reverse SSH tunneling from Cloud Shell to any public endpoint is serving as a means for threat actors to distribute malicious campaigns or perform harmful activity. This is described in Figure 3. Customers who may be concerned should enable a VPC service perimeter to limit access to services and users inside their perimeter.

Figure 3: Cloud Shell enabling SSH Tunneling



In the process described, after a threat actor has access to a Google Cloud Shell instance, for example, through the exploitation of leaked credentials, the actor then creates a reverse shell SSH tunnel to an SSH server. The threat actor then connects to the SSH server through an SSH tunnel. A SOCKS5 proxy is then used to connect to the Google Cloud Shell before connecting to the endpoint. This results in an obfuscation of the threat actor's origin.

Because Cloud Shell doesn't allow inbound connections, it cannot be used as a proxy server by default. By creating a reverse SSH tunnel to other endpoints, the "Cloud Shell" can perform port forwarding, allowing the endpoint to connect to any port of the "Cloud Shell," meaning that the endpoint can use the Cloud Shell as a proxy server.

Strategic Significance

As your strategic cloud provider, we want to make sure that customers are using our products safely and ensure no abuse is happening inside their environments. Threat actors can abuse the proxy capabilities to perform an action on behalf of a compromised Cloud Shell instance without revealing their identity. As part of their proactive monitoring and configuration reviews, it is recommended that Google Cloud customers review account and security settings to limit the potential for initial credential compromise enabling this abuse vector.

Google Cloud Specific Mitigations

In order to protect the initial compromise of credentials, which can then lead to the scenario of Cloud Shell abuse, Google in general strongly recommends that customers implement 2-step verification and where possible implement it in the form of hardware-based tokens. Additionally, customers can use VPC Service Controls to prevent data exfiltration from their production environments.

In addition to the 2-step verification and VPC Service Controls, Google Cloud customers have additional resources including:

- A variety of access control options within Compute Engine including using service accounts to authenticate apps instead of using user credentials along with options for 2-step verification.
- Policy Intelligence tools to help understand and manage policies to proactively improve security configurations.
- Best practices for handling compromised credentials

As general best practices, Google Cloud customers can also use Container Analysis to perform vulnerability scanning of metadata storage for containers and the Web Security Scanner in the Security Command Center to identify security vulnerabilities in their App Engine, Google Kubernetes Engine, and Compute Engine web applications that could lead to an initial compromise. The Web Security scanner will crawl applications, following all links within the scope of the starting URL and attempt to exercise as many user input and event handlers as possible.

Domain previously identified by TAG used in ongoing attacks against researchers

Threat Description / TTPs

A domain previously identified by Google's Threat Analysis Group (TAG) was associated with intrusions launched by a North Korean government-backed entity against security and vulnerability researchers. These intrusions originated from a trojanized version of IDA Pro as discovered by ESET Research. The actor presumably shared the modified installer for IDA Pro, a disassembler popular among security researchers, in order to replace a specific DLL within the installer package. When executed, it would attempt to download and install a backdoor from a North Korean controlled domain.

Over the past 12+ months, the actor has launched multiple campaigns against the security and vulnerability research community including the following techniques:

- Developing fake social media profiles and submitting real bugs to bug bounty programs in order to build credibility.
- Creating fake exploit broker companies with websites, accounts, *etc.*
- Trojanizing exploit proof of concepts and other software packages commonly shared among researchers.
- Suspected of using 0-days, which were stolen from some of their victims.

Strategic Significance

Based on TAG's analysis, these targets are not restricted to one region. They include numerous researchers and companies including some in China. Google Cloud customers, who engage in security and vulnerability research have the potential for being included in those targeted.

Google Cloud Specific Mitigations

Code downloaded by clients should undergo hashing authentication. It is a common practice for clients to download updates and code from various Internet sources, raising concern that unauthorized

code may be downloaded in the process. Meddler in the Middle (MITM) attacks may cause unauthorized source code to be pulled into production. By hashing and verifying all downloads, the integrity of the software supply chain can be preserved and an effective chain of custody can be established.

Recommendations

Google Cloud continues to operate within a "shared fate" model that exemplifies a true partnership with its customers. This partnership includes providing trends and lessons learned from recent incidents or close-calls in the wild with which Google assisted. The following is a summary of Google Cloud's recommendations based upon customer incidents across various platforms that we helped address:

When securing an environment and ensuring exfiltration has not occurred after an incident, **customers should review logs associated with file sharing and domain-wide Takeout in addition to looking for implanted malware.** Threat actors have been known to use tools native to the Cloud environment rather than downloading custom malware or scripts to avoid detection. This "living off the land" technique has been used extensively in on-premise compromises and is being adopted in Cloud environments.

In the situations where it is necessary to respond to a security incident, where there was an exploitation of third-party software in the Cloud instance, **customers should review logs to determine the point of initial compromise as well as determining if additional unauthorized software was installed after the initial compromise or if unauthorized configurations were made.** Adversaries have been known to close doors behind themselves, i.e., patch the vulnerability which grants initial access, and then pivot to new software, malware, or reverse shells to maintain presence.

Code downloaded by clients should undergo hashing authentication. It is a common practice for clients to download updates and code from various Internet sources, raising concern that unauthorized code may be downloaded in the process. Meddler in the Middle (MITM) attacks may cause unauthorized source code to be pulled into production. By hashing and verifying all downloads, the integrity of the software supply chain can be preserved and an effective chain of custody can be established.

Table 1: Observed risks and countermeasures

Risk	Countermeasures
Instance vulnerabilities	Follow password best practices and best practices for configuring Cloud environments. Update third-party software prior to a Cloud instance being exposed to the web. Avoid publishing credentials in GitHub projects.

	Use Container Analysis to perform vulnerability scanning and metadata storage. Leverage Web Security Scanner in the Security Command Center to identify security vulnerabilities in App Engine, Google Kubernetes Engine, and Compute Engine. Use service accounts with Compute Engine to authenticate apps instead of using user credentials. Implement Policy Intelligence tools to help understand and manage policies. Use predefined configurations through Assured Workloads to reduce misconfigurations. Set up conditional alerts in the Cloud Console to send alerts upon high resource consumption. Enforce and monitor password requirements for users through the Google Admin console.
Downloading software updates	Establish a strong chain of custody by hashing and verifying downloads.
Using public code repositories	Audit projects published on GitHub and other sites to ensure credentials and certificates were not included.