Sliver C2

Sliver is an open source cross-platform adversary emulation/red team framework, it can be used by organizations of all sizes to perform security testing. Sliver's implants support C2 over Mutual TLS (mTLS), WireGuard, HTTP(S), and DNS and are dynamically compiled with per-binary asymmetric encryption keys.

The server and client support MacOS, Windows, and Linux. Implants are supported on MacOS, Windows, and Linux (and possibly every Golang compiler target but we've not tested them all).

WHY IS IT GETTING MORE ATTRACTION ?

Silver C2 is gaining popularity due to these reasons :

- **Open-source alternative to Cobalt Strike and Metasploit**
- **Modularity of the platform with Armory**
- **Cross-platform : OS X, Linux and Windows**

The framework provides all core capabilities for adversary simulation and most notables are:

- **Dynamic code generation**
- **Compile-time obfuscation**
- **Multiplayer-mode**
- **Staged and Stageless payloads**
- **Secure C2 over mTLS, WireGuard, HTTP(S), and DNS**

- **Windows process migration, process injection, user token manipulation, etc.**
- **Let's Encrypt integration**
- **In-memory .NET assembly execution**
- **COFF/BOF in-memory loader**
- **TCP and named pipe pivots**
- **Armory, alias and extension package manager**

THREAT ACTORS LEVERAGING SLIVER C2

Recently, some threat research teams, including the Cybereason GSOC, identified cases of BumbleBee loaders dropping Sliver C2 following the initial infection.

Infrastructure

Sliver, like many C2 frameworks, supports various network protocols such as DNS, HTTP/TLS, MTLS, and

TCP. It can also accept implant or operator connections and host files to impersonate a benign web server.

The first step in testing any C2 framework is starting listeners and scanning them to identify anomalies. Some common artifacts are unique HTTP header combinations and JARM hashes, the latter of which are active fingerprinting techniques for TLS servers. RiskIQ has shared such a methodology for Sliver and Bumblebee detection.

Payloads

Since Sliver is written in the Go programming language (GoLang), its implants are cross-platform compatible. By default, operators can generate implants in several formats, including:

- **Shellcode**

- **Executable**
- **Shared library/DLL**
- **Service**

Code execution

Just like any other C2 framework, Sliver utilizes process injection as a core part of many default commands or capabilities, such as:

- **migrate (command) — migrate into a remote process**
- **spawndll (command) — load and run a reflective DLL in a remote process**
- **sideload (command) — load and run a shared object (shared library/DLL) in a remote process**
- **msf-inject (command) — inject a Metasploit Framework payload into a process**

- **execute-assembly (command) — load and run a .NET assembly in a child process**
- **getsystem (command) — spawn a new Sliver session as the *NT AUTHORITY\SYSTEM* user**
- **extensions/aliases — Beacon Object Files (BOFs), .NET apps, and other third-party tooling**

HUNTING FOR SLIVER C2 IMPLANTS

The use of Sliver C2 generates many unique behaviors that can be used as detection triggers. In the following diagram, we list all the detection techniques identified through this research

SLIVER GETSYSTEM DETECTION

When the Sliver C2 *getsystem* command is executed from the administration panel, we identified that the process hosting the current implant will systematically inject itself into the *spoolsv.exe* process.

Hosted injected thread (CreateRemoteThread) from any process to *spoolsv.exe*.

Detection Logic

Detect call(s) to the CreateRemoteThread Windows API to run code inside another process named spoolsv.exe

SHELL FEATURE — DETECTION OF SPECIFIC POWERSHELL COMMAND LINE

As stated in the above chapters, Sliver C2 has a very unique way of spawning the *powershell.exe* process when the Sliver C2 '*Shell*' command is executed for a specific implant.

The following query finds the default, unique PowerShell command used when Sliver creates an interactive shell with the *'Shell'* command.

```
DeviceProcessEvents

| where ProcessCommandLine == 'powershell.exe -NoExit -Command
[Console]::OutputEncoding=[Text.UTF8Encoding]::UTF8'
```

Sideload/SpawnDll/Execute-Assembly/SLIVER EXECUTE-ASSEMBLY OR MIGRATE FEATURE

The *Sideload*, *SpawnDll*, and *Execute-Assembly* commands spawn and inject into *notepad.exe* by default. The following query finds process creation events where the same process creates and injects into *notepad.exe* within 10 seconds.

Sliver C2 migrate command by default injects the implant binary into newly created notepad.exe processes and creates a remote thread to run the malicious code.

Event ID 8 related to [CreateRemoteThread] detection.

```
DeviceProcessEvents

| where ActionType == 'ProcessCreated'

| where ProcessCommandLine =~ 'notepad.exe'

| distinct InitiatingProcessId, DeviceId
```

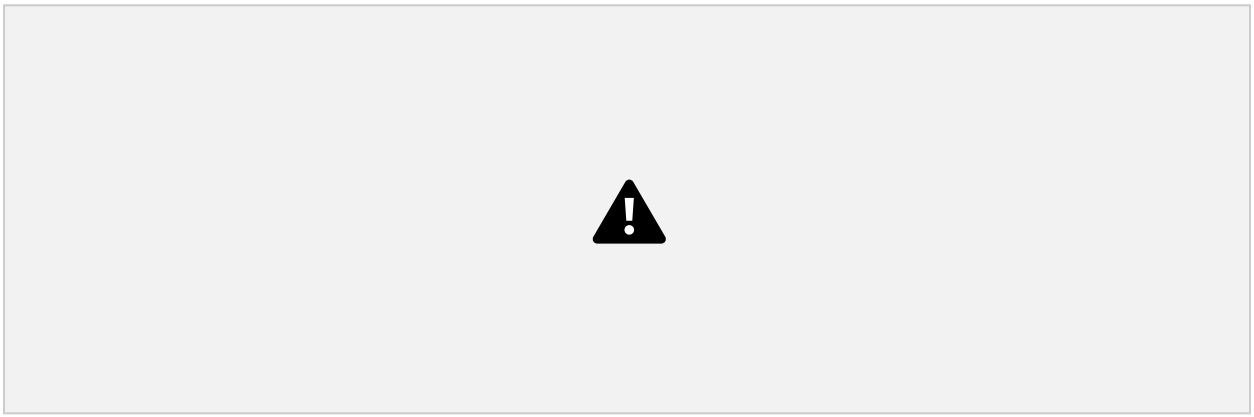
```
| join kind=inner (

   DeviceEvents

   | where ActionType == 'CreateRemoteThreadApiCall'

   | where ProcessCommandLine == 'notepad.exe'

   | where Timestamp between (ProcessCreationTime ..
(ProcessCreationTime+10s))

) on DeviceId, InitiatingProcessId
```

PSEXEC FEATURE DETECTION

Sliver C2 built-in PsExec command, used for lateral movements, creates a service on remote machine with default name “Sliver.”

The following query finds default values for the *ImagePath*, *DisplayName*, and *Description* of the

service installed on the remote system when using Sliver's *PsExec* command.



```
DeviceRegistryEvents

| where ActionType == 'RegistryValueSet'

| where (RegistryValueName == 'ImagePath' and RegistryValueData
matches regex @'^[a-zA-Z]:\\windows\\temp\\[a-zA-Z0-9]{10}\.exe') or

       (RegistryValueName == 'DisplayName' and RegistryValueData ==
'Sliver') or

       (RegistryValueName == 'Description' and RegistryValueData ==
'Sliver implant')
```

The following query is an alternative method of searching on the same service properties but within service installation events instead of registry keys.

```
DeviceEvents

| where ActionType == 'ServiceInstalled'

| extend JSON = parse_json(AdditionalFields)

| where (FolderPath endswith_cs @':\windows\temp' and FileName matches regex @'^[a-zA-Z0-9]{10}\.exe') or (JSON.ServiceName == 'Sliver')
```

SLIVER C2 PAYLOADS IN C:\WINDOWS\TEMP

Without any customization, Sliver delivers its payloads remotely in the *C:\Windows\Temp* directory.

Although it might lead to false-positives, searching for suspicious/injected processes using any image file stored in this folder can identify the use of Sliver C2.

SPECIFIC NETWORK PORT COMMUNICATION

Sliver C2 server listens on default ports if not instructed otherwise :

- **TCP Port 8888 for the mTLS service**
- **UDP Port 51820 for the Wireguard service**
- **TCP Port 443 for the HTTPS service**

References:

[1]

https://www.bleepingcomputer.com/news/security/hackers-adopt-sliver-toolkit-as-a-cobalt-strike-alternative/

[2]

https://thehackernews.com/2023/01/threat-actors-turn-to-sliver-as-open.html

[3]

https://www.cybereason.com/blog/sliver-c2-leveraged-by-many-threat-actors

[4]https://www.ncsc.gov.uk/files/Advisory%20Further%20TTPs%20associated%20with%20SVR%20cyber%20actors.pdf

[5]

https://www.team-cymru.com/post/sliver-case-study-assessing-common-offensive-security-tools

[6]

https://www.darkreading.com/vulnerabilities-threats/-sliver-cobalt-strike-alternative-malicious-c2

[7]

https://blogs.vmware.com/security/2023/01/detection-of-lateral-movement-with-the-sliver-c2-framework.html