

# Mac, Where's My Bootstrap?

Detecting XPC logic exploits

**Csaba Fitzl**

Principal Security Researcher @Kandji

**Brandon Dalton**

Senior Security Researcher @Microsoft





**Brandon Dalton (@partyD0lphin)**

*Senior Security Researcher (Microsoft)*

OS Internals and M365 research

Developer of Red Canary Mac Monitor

ATT&CK contributor

 Family >>

Travel  / History 



# Csaba Fitzl (@theevilbit)

*Principal macOS Security Researcher @Kandji* 🐝

author of EXP-312 - macOS Exploitation training (🐙) at OffSec

macOS bug hunter (~100 CVEs from 🍎)

husband, father

hiking, trail running 🏔️ 🏔️



# Over the next 25 minutes...

- XPC vulnerabilities and abuse
  - Our idea
  - **XPC\_CONNECT** – what's missing? 😞
  - The bootstrap server
- 
-  Resolving an XPC service path
  -  Tool drop
  -  Detecting a Twitch 0day
  -  Recommendations

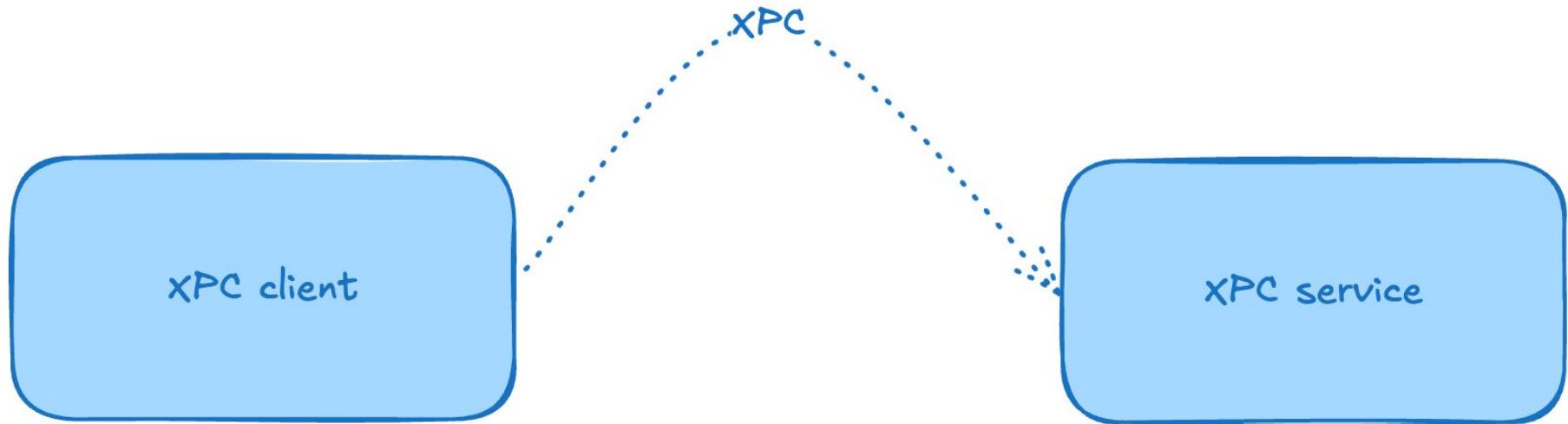


**Objective**  
by the **Sea**  
version 7.0



# How it started?

# What is XPC?



typically:

- not privileged
- sandboxed

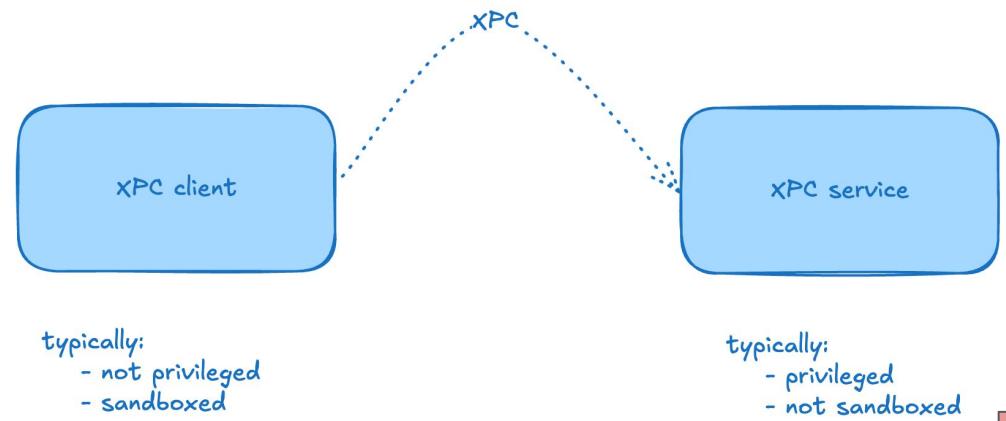
typically:

- privileged
- not sandboxed



# The XPC problem

- By default:
  - A global service reachable by every process
- If client access is not controlled ⇒ service is open for abuse
- Client validation is tricky
- Developers make mistakes...



# Safe XPC client validation

## Code signing validation

- Cert chain (Apple and developer) ⇒ to prevent fake root certificates
- Bundle ID(s) ⇒ to prevent other clients from the same vendor
- Team ID ⇒ to prevent other developers / apps
- App version ⇒ to prevent downgrade attacks

## Verify clients entitlements

- Hardened runtime ⇒ to prevent client code injection attacks
- Library validation ⇒ to prevent client code injection attacks

## Identify client process

- Use Audit token ⇒ to prevent PID reuse attacks



# XPC vulnerability scope

## Impact

1. Local privilege escalation
2. Data compromise

## Prevalence

Relatively easy for an attacker to both **identify** and **exploit**.



# The problem...

# Exploiting XPC in AntiVirus

Home > Techniques > Enterprise > Inter-Process Communication > XPC Services

Blog > Threat Intelligence

## How To Used fo Inter-Process Communication: XPC Services

Other sub-techniques of Inter-Process Communication (3)



Christian Lutz

### Exploiting GOG Galaxy XPC service

CVE-2019-0712 MS Office 2010  
for Mac Privilege Escalation via a  
Legacy Package



REGISTER SPEAKERS CFP AGENDA VENUE TRAININGS SPONSORS ARCHIVE CONTACT

TYLER BOHAN

## OSX XPC REVISITED - 3RD PARTY APPLICATION FLAWS

Author: Zhipeng Huo(@R3dF09) of Tencent Security Xuanwu Lab

Validation Bypass

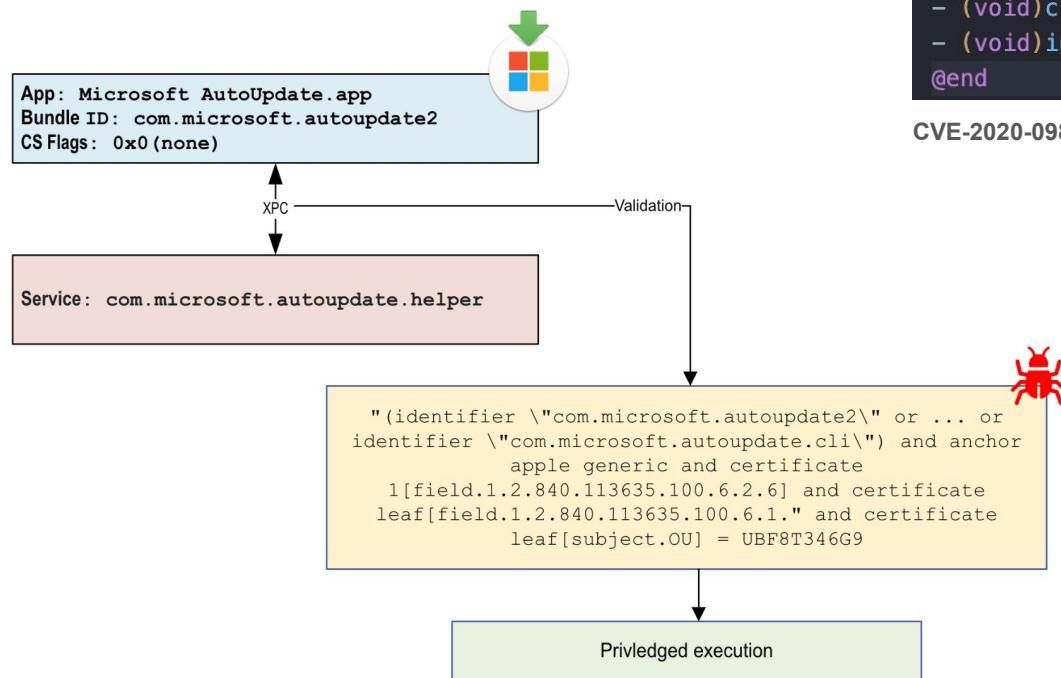
In /Library/PrivilegedHelperTools/com.microsoft.autoupdate.helper there's a  
XPC service com.microsoft.autoupdate.helper.

# For example...

## In the Wild

Restricted

Privileged



```
@protocol MAUHelperToolProtocol <NSObject>
- (void)removeInstallLogFile:(NSString *)arg1 atLevel:(int)arg2;
- (void)logString:(NSString *)arg1 atLevel:(int)arg2;
- (void)removeClone:(NSString *)arg1 withReason:(int)arg2;
- (void)restoreCloneToAppInstallLocation:(NSString *)arg1;
- (void)createCloneFromApp:(NSString *)arg1;
- (void)installUpdateWithPackage:(NSString *)arg1;
@end
```

CVE-2020-0984 – Microsoft AutoUpdate

# The Idea



# Detect XPC Attacks using Endpoint Security

- 💡 We have an XPC ES event (thanks Sonoma)!
  - 💡 Let's monitor client - service validation with an ES client, e.g.:
    - 💡 Verify TeamID/etc... for third party
- ⇒ requires code signing validation of both sides of the connection







# Endpoint Security... what do we have?

```
/**  
 * @brief Notification for an XPC connection being established to a named service.  
 *  
 * @field service_name      Service name of the named service.  
 * @field service_domain_type  The type of XPC domain in which the service resides in.  
 *  
 * @note This event type does not support caching (notify-only).  
 */  
typedef struct {  
    es_string_token_t service_name;  
    es_xpc_domain_type_t service_domain_type;  
} es_event_xpc_connect_t;
```

```
/**  
 * @brief This enum describes the types of XPC service domains.  
 */  
typedef enum {  
    ES_XPC_DOMAIN_TYPE_SYSTEM = 1,  
    ES_XPC_DOMAIN_TYPE_USER,  
    ES_XPC_DOMAIN_TYPE_USER_LOGIN,  
    ES_XPC_DOMAIN_TYPE_SESSION,  
    ES_XPC_DOMAIN_TYPE_PID,  
    ES_XPC_DOMAIN_TYPE_MANAGER,  
    ES_XPC_DOMAIN_TYPE_PORT,  
    ES_XPC_DOMAIN_TYPE_GUI,  
} es_xpc_domain_type_t;
```

⌚ XPC connection details

- Service name: com.apple.audio.SandboxHelper
- Domain type: PID

Mac Monitor

```
"event": {  
    "xpc_connect": {  
        "service_name": "com.apple.system.opendirectoryd.api",  
        "service_domain_type": 1  
    }  
},
```

eslogger



Let's take it out for a test drive...



# What will not trigger an XPC connect event?

## bootstrap\_look\_up( ... )

Create a Mach service and register w/launchd

```
#include <mach/mach.h>
#include <servers/bootstrap.h>

int main() {
    mach_port_t port;
    mach_port_allocate(mach_task_self(), MACH_PORT_RIGHT_RECEIVE, &port);
    mach_port_insert_right(mach_task_self(), port, port, MACH_MSG_TYPE_MAKE_SEND);
    bootstrap_register(bootstrap_port, "com.microsoft.domain-example", port);
```

What launchd knows...

```
user/501 = {
    type = user
    handle = 501
    ...
    creator = bluetoothd[397]
    creator euid = 0
    session = Background
    gui asid = 100016
    security context = { - }
}

bringup time = 112 ms
death port = 0x0
subdomains = {
    gui/501
}

services = { - }

unmanaged processes = {
}

endpoints = [
    0x32703 M D com.apple.lsd.openurl
    0x133b87 U A com.microsoft.domain-example
    0xa8e23 M A 2BUA8C4S2C.com.1password.browser-helper
    0xb6c5f U A cs (Apple)_OpenStep
]
```

Looking up a Mach service name will not trigger an ES event

```
#include <mach/mach.h>
#include <servers/bootstrap.h>

int main() {
    mach_port_t port;
    bootstrap_look_up(bootstrap_port, "com.microsoft.domain-example", &port);
}
```

Endpoint Security (ES) event

```
"xpc_connect": [
    "service_domain_type": ?,
    "service_name": "com.apple.bg.system.task"
]
```

### 1. Create an launchd (XPC) service

```
#include <xpc/xpc.h>

int main() {
    xpc_connection_t service = xpc_connection_create_mach_service(
        "com.microsoft.xpc-domain-example",
        NULL,
        XPC_CONNECTION_MACH_SERVICE_LISTENER
    );
}
```

### launched service plist

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "1.0">
<plist version="1.0">
<dict>
    <key>Label</key>
    <string>com.microsoft.xpc-domain-example</string>
    <key>MachServices</key>
    <dict>
        <key>com.microsoft.xpc-domain-example</key>
        <true/>
    </dict>
</dict>
```

### What launchd knows...

```
gui/501 = {
    type = Login
    handle = 100016
    active count = 431
    service count = 430
    active service count = 257
    creator = loginwindow(407)
    creator euid = 0
    session = Aqua
    endpoint destination = com.apple.xpc.launchd.domain.user.501
    auxiliary bootstrapper = com.apple.xpc.otherbsd (complete)
    security context = {
        uid = 501
        asid = 100016
    }

    bringup time = 174 ms
    death port = 0x10d03

    environment = {
        SSH_AUTH_SOCK => /private/tmp/com.apple.launchd.7hFG3Gzg3/Listeners
    }

    services = {
        30309 - com.microsoft.xpc-domain-example
        18128 - application.com.apple.Preview.1152921500311897332.
        0 0 com.google.GoogleUpdater.wake
    }

    unmanaged processes = { - }

    endpoints = { - }

    externally-hosted endpoints = {
        0x3ce0b M A com.microsoft.xpc-domain-example
        0xb0c5f M A Multilingual (Apple)_OpenStep
        0xb0c5f U A vi (Apple)_OpenStep
        0xb0c5f U A ko (Apple)_OpenStep
    }
}
```

2. Register

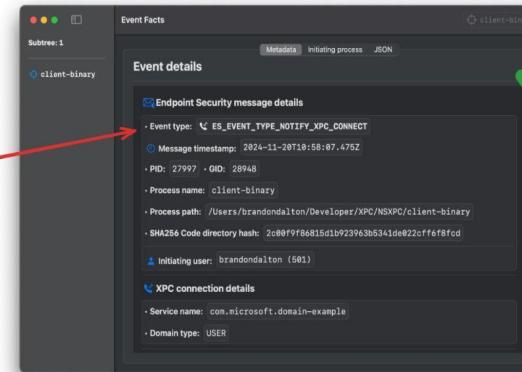
3. Client request

Looking up the service will trigger the ES event

```
#include <xpc/xpc.h>

int main() {
    xpc_connection_t connection = xpc_connection_create_mach_service(
        "com.microsoft.xpc-domain-example",
        NULL,
        0
    );
}
```

4. ES event



Great! Now where's the path...?

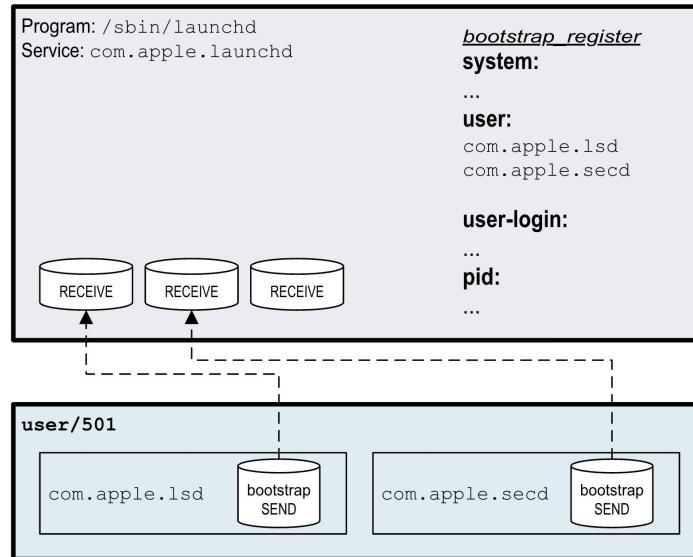


... the bootstrap server

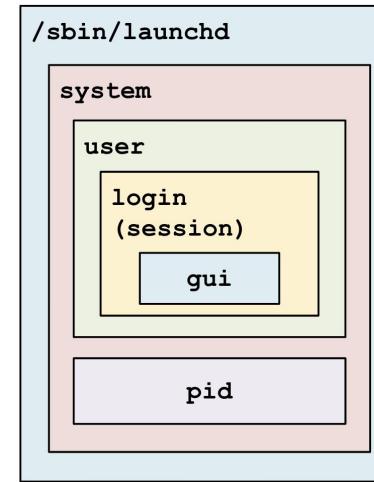
# /sbin/launchd

“Domains are, in effect, nothing more than figments of `launchd`'s twisted imagination.” - MOXII Vol 1 pg.438

- Manager for jobs and XPC / Mach service
- ... 🤔 do you know where the program is?



## Bootstrap namespaces / domains





imgflip.com

# The idea

/usr/bin/launchctl

Seems to know how to get all the information we need...

## What used to help?

Jonathan Levin's launjctl -

<https://newosxbook.com/articles/jlaunchctl.html>

Apple broke it :(

```
❯ launchctl print pid/55860/com.1password.safari.extension
pid/55860/com.1password.safari.extension = {
    active count = 3
    path = /Applications/1Password for Safari.app/Contents/PlugIns/1Password.appex
    type = Extension
    managed_by = com.apple.runningboard
    state = running
    bundle id = com.1password.safari.extension
    bundle version = 81054022
    extension point = com.apple.Safari.web-extension

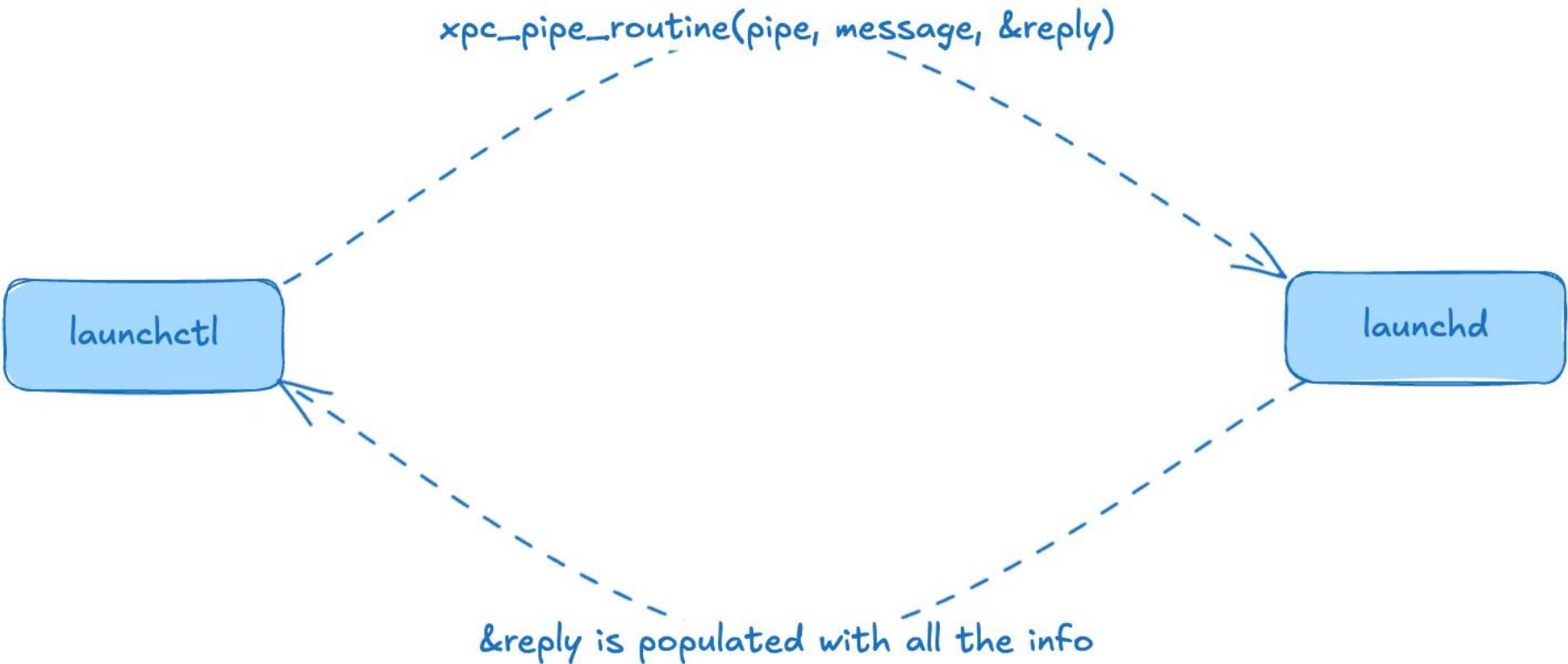
    program = /Applications/1Password for Safari.app/Contents/PlugIns/1Password.appex/Contents/MacOS/1Password
    arguments = {
        /Applications/1Password for Safari.app/Contents/PlugIns/1Password.appex/Contents/MacOS/1Password
        -AppleLanguages
        ("en-US")
    }

    sandbox profile = plugin
    inherited environment = {
        PATH => /usr/bin:/bin:/usr/sbin:/sbin
        SSH_AUTH_SOCK => /private/tmp/com.apple.launchd.Kx89exPUM0/Listeners
        HOME => /Users/brandondalon
        __CF_USER_TEXT_ENCODING => 0x1F5:0x0:0x0
        TMPDIR => /var/folders/nn/ylnbg2d51q3b3lr43mw6kxm000gn/T/
    }

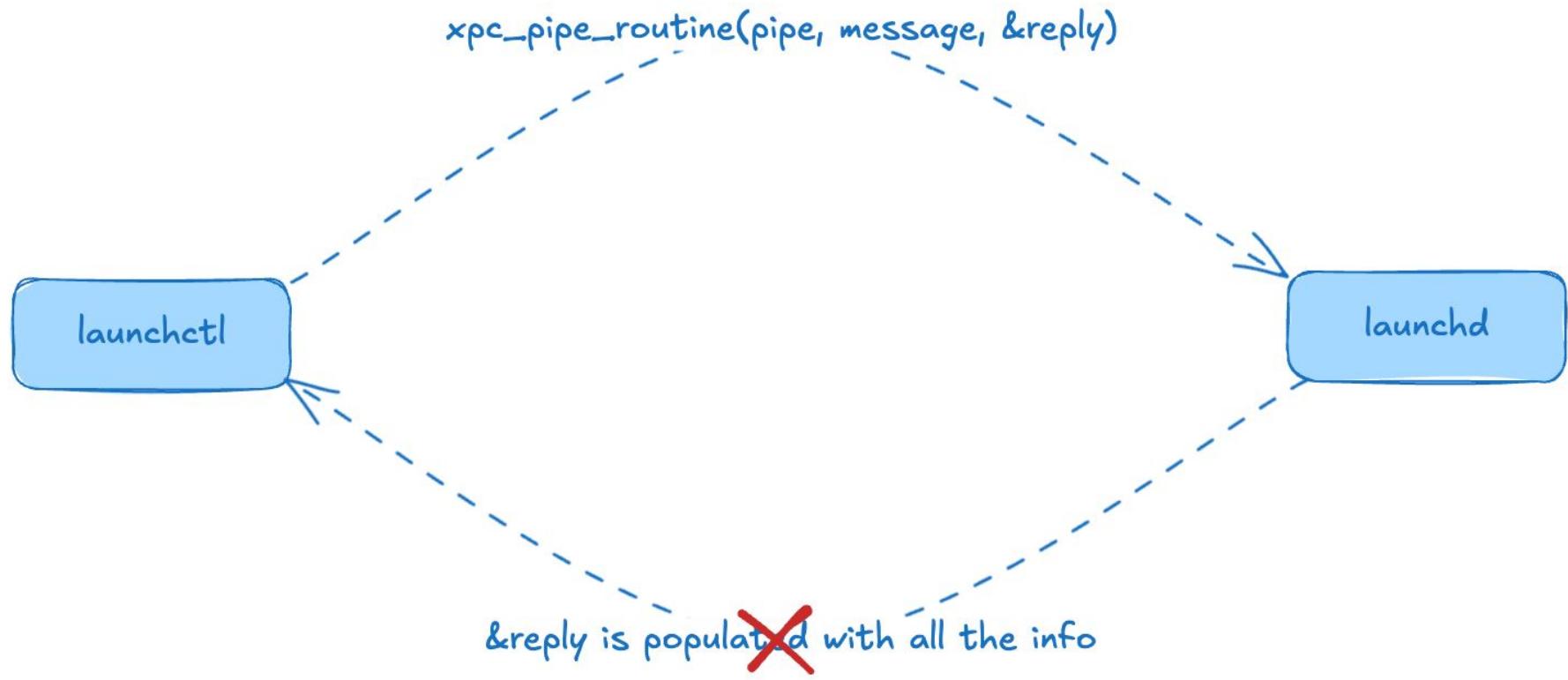
    default environment = {
        PATH => /usr/bin:/bin:/usr/sbin:/sbin
    }

    endpoints = {
        "com.1password.safari.extension" = {
            port = 0x1097db
            active = 1
            managed = 1
            reset = 0
            hide = 0
            watching = 0
            non-launching = 1
        }
        "com.1password.safari.extension.apple-extension-service" = {
            port = 0x153a2f
            active = 1
            managed = 1
            reset = 0
            hide = 0
            watching = 0
        }
    }
}
```

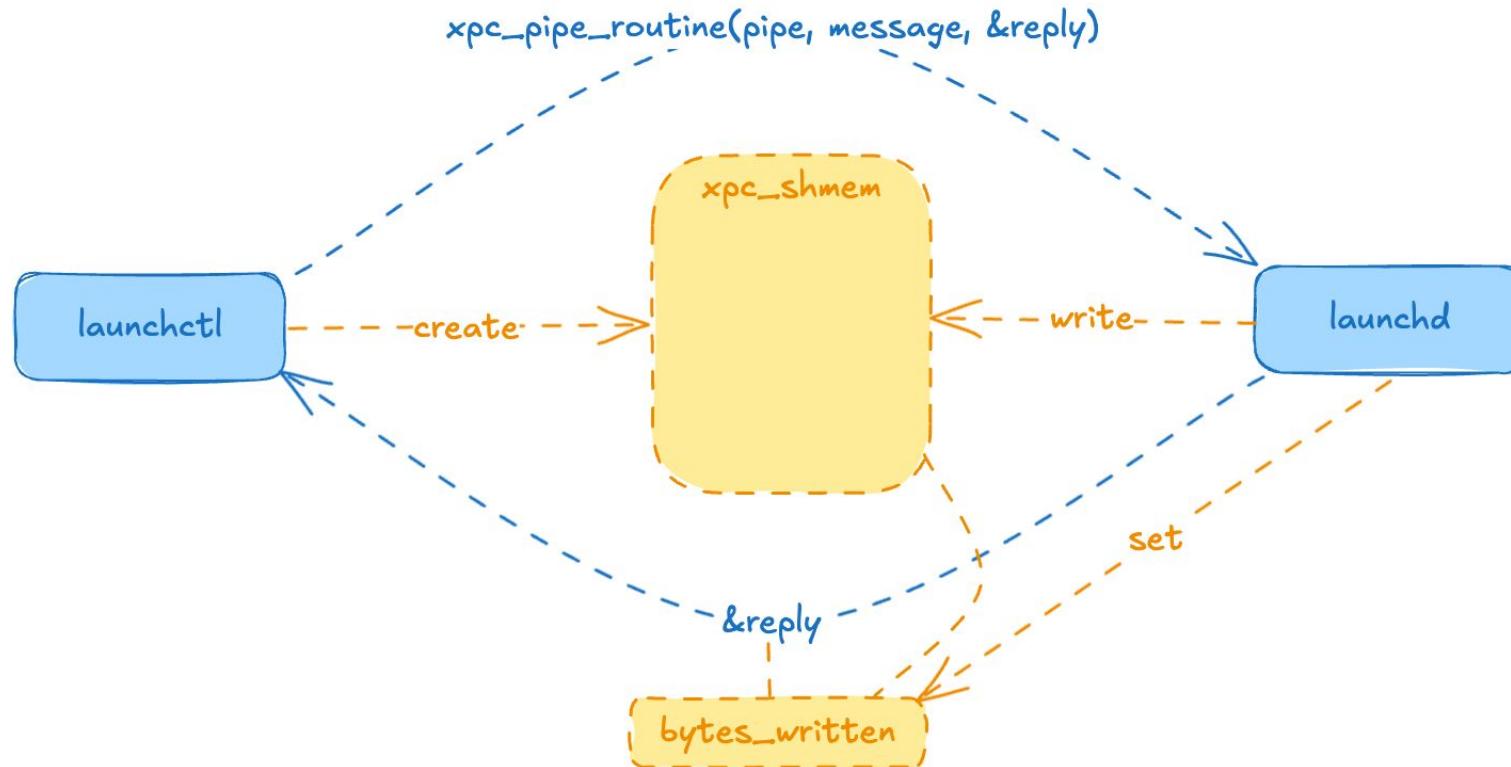
# Before



# We have an issue...



# Then how does `launchctl` do it?



Incoming request:  
1) print service domain target  
2) print domain target

Create an XPC dictionary using known keys  
and Create a shared memory region

```
let message = xpc_dictionary_create(nil, nil, 0)
xpc_dictionary_set_uint64(message, "handle", handle)
xpc_dictionary_set_uint64(message, "type", type)
xpc_dictionary_set_uint64(message, "routine", routine)
xpc_dictionary_set_uint64(message, "subsystem", subsystem)
if let name = name {
    xpc_dictionary_set_string(message, "name", name)
}

allocateSharedMemory(for: message)
```

Send the message and parse the response

```
let pipe = xpc_pipe_create_from_port(bootstrapPort, 0)
var reply: xpc_object_t?
let error = xpc_pipe_routine(pipe, message, &reply)
// ...
return parseResponse(from: reply ...)
```

Why do this to us? Apple must know  
it's a barrier to entry...

Parse... "this" ... 😊

```
gui/501 = {
    type = login
    handle = 100016
    active count = 435
    service count = 434
    active service count = 259
    creator = loginwindow[407]
    creator euid = 0
    session = Aqua
    // ...

    services = {
        65033      -      application.com.vector35.binaryninja.119897822.128734987
        672        -      com.apple.syncdefaultsd
        745        -      com.apple.assistantd
        0          0      com.apple.DataDetectorsLocalSources
        0          0      com.apple.unmountassistant.userAgent
        9903       (pe)   com.apple.mlhostd
                      0      com.apple.SafariHistoryS
    }
}
```

# What are those special keys?

MOXil Vol 1 Table 13-17

Using the **bootstrap port** we can request info from **launchd** using **XPC dictionaries**.

- **type**: The domain we're targeting.
  - `system`, `user`, `login`, `pid`, `gui`
- **subsystem**: Service or domain targets
- **handle**: The domain specifier
  - E.g. UID, ASID, or PID
- **routine**: A specific command in subsystem
- and **name**: The service name if service target



```
xpc_pipe_create_from_port(bootstrap, ;)  
xpc_pipe_routine(pipe, dictionary, &reply)
```



# Resolving a path? (PID domain example)

Initiating GID: 55860 →

com.1password.safari.extension → com.1password.safari.extension.apple-extension-service

```
Last login: Thu Dec 5 21:27:13 on ttys001
> launchctl print pid/55860
pid/55860 = {
    type = pid
    handle = 55860
    active count = 23
    user-domain count = 3
    service count = 22
    active service count = 18
    originator = /System/Volumes/Preboot/Cryptexes/App/System/Applications/Safari
    creator = Safari[55860]
    creator euid = 501
    uniqueid = 55860
    security context = {
        uid = 501
        asid = 100018
    }
}

bringup time = 0 ms
death port = 0x114b3b

environment = {
    PATH => /usr/bin:/bin:/usr/sbin:/sbin
    SSH_AUTH_SOCK => /private/tmp/com.apple.launchd.Kx89ePUM0/Listeners
    HOME => /Users/brandondalon
    _CF_USER_TEXT_ENCODING => 0x1F5:0x0:0x0
    TMPDIR => /var/folders/nm/ylnbgb2d51q3b3lr43mw5kxm000gn/T/
}

services = {
    56090 -> com.apple.WebKit.WebContent.A0942BFD-0478-49E1
    55878 -> com.apple.WebKit.WebContent.424FF76-C22F-4761
    55876 -> com.apple.Safari.SandboxBroker
    55875 -> com.apple.WebKit.WebContent.283C3A3-870F-4DF6
    55874 -> com.apple.Safari.SearchHelper
    55882 -> com.apple.webkit.xpc.openAndSavePanelService
    55862 -> com.apple.WebKit.WebContent.D4270-E1-E764-4723
    0 -> com.apple.WebKit.WebContent
    56184 -> com.apple.MTLCCompilerService.FA670E45-1AB8-45E6
    55877 -> com.apple.WebKit.WebContent.182_FFC0-E981-419E
    0 -> com.apple.WebKit.GPU
    55868 -> com.apple.Safari.ContentBlocker.Loader
    55881 -> com.apple.password.safari.extension
    55865 -> com.apple.WebKit.WebContent.D5AC0AD0-D641-4CC6
    55864 -> com.apple.Safari.SandboxHelper
    0 -> com.apple.WebKit.Networking
    55873 -> com.apple.GPU.FC49D0C3-13A8-4949-8332-6
    56091 -> com.apple.SirikitSService.TrialProxy
    55867 -> net.shinyfrog.bear.Bear-Safari-Extension
    56026 -> com.apple.MTLCCompilerService.DA943D04-7A89-427
    55866 -> com.apple.WebKit.Networking.2D1204F1-E3D7-434C
}
```

```
> launchctl print pid/55860/com.1password.safari.extension
pid/55860/com.1password.safari.extension = {
    active count = 3
    path = /Applications/1Password for Safari.app/Contents/PlugIns/1Password.appex
    managed by = Extension
    state = running
    bundle id = com.1password.safari.extension
    bundle version = 81054022
    extension point = com.apple.Safari.web-extension

    program = /Applications/1Password for Safari.app/Contents/PlugIns/1Password.appex/Contents/
    arguments = {
        /Applications/1Password for Safari.app/Contents/PlugIns/1Password.appex/Contents/
        -AppleLanguages
        {"en-US"}
    }

    sandbox profile = plugin
    inherited environment = {
        PATH => /usr/bin:/bin:/usr/sbin:/sbin
        SSH_AUTH_SOCK => /private/tmp/com.apple.launchd.Kx89ePUM0/Listeners
        HOME => /Users/brandondalon
        _CF_USER_TEXT_ENCODING => 0x1F5:0x0:0x0
        TMPDIR => /var/folders/nm/ylnbgb2d51q3b3lr43mw5kxm000gn/T/
    }

    default environment = {
        PATH => /usr/bin:/bin:/usr/sbin:/sbin
    }
}

endpoints = {
    "com.1password.safari.extension" = {
        port = 0x1097db
        active = 1
        managed = 1
        reset = 0
        hide = 0
        watching = 0
        non-launching = 1
    }
    "com.1password.safari.extension.apple-extension-service" = {
        port = 0x153a2f
        active = 1
        managed = 1
        reset = 0
        hide = 0
        watching = 0
    }
}
```

XPC service name: com.apple.audio.SandboxHelper  
Domain: pid  
Initiating GID: 34496

Print per-pid domain target:  
pid/<initiating proc GID>

For each launchd service:  
Print the service target:  
pid/<initiating proc GID>/<service>

Look for the XPC service  
name in the  
"endpoints"

Return the "program" path

Program path not found

[github.com/Brandon7CC/mac-whereis-my-bootstrap/releases/](https://github.com/Brandon7CC/mac-whereis-my-bootstrap/releases/)

# *Detection* not AUTH

```
!service.is_apple && service.team_id != requestor.team_id
```



## Distribution entitled

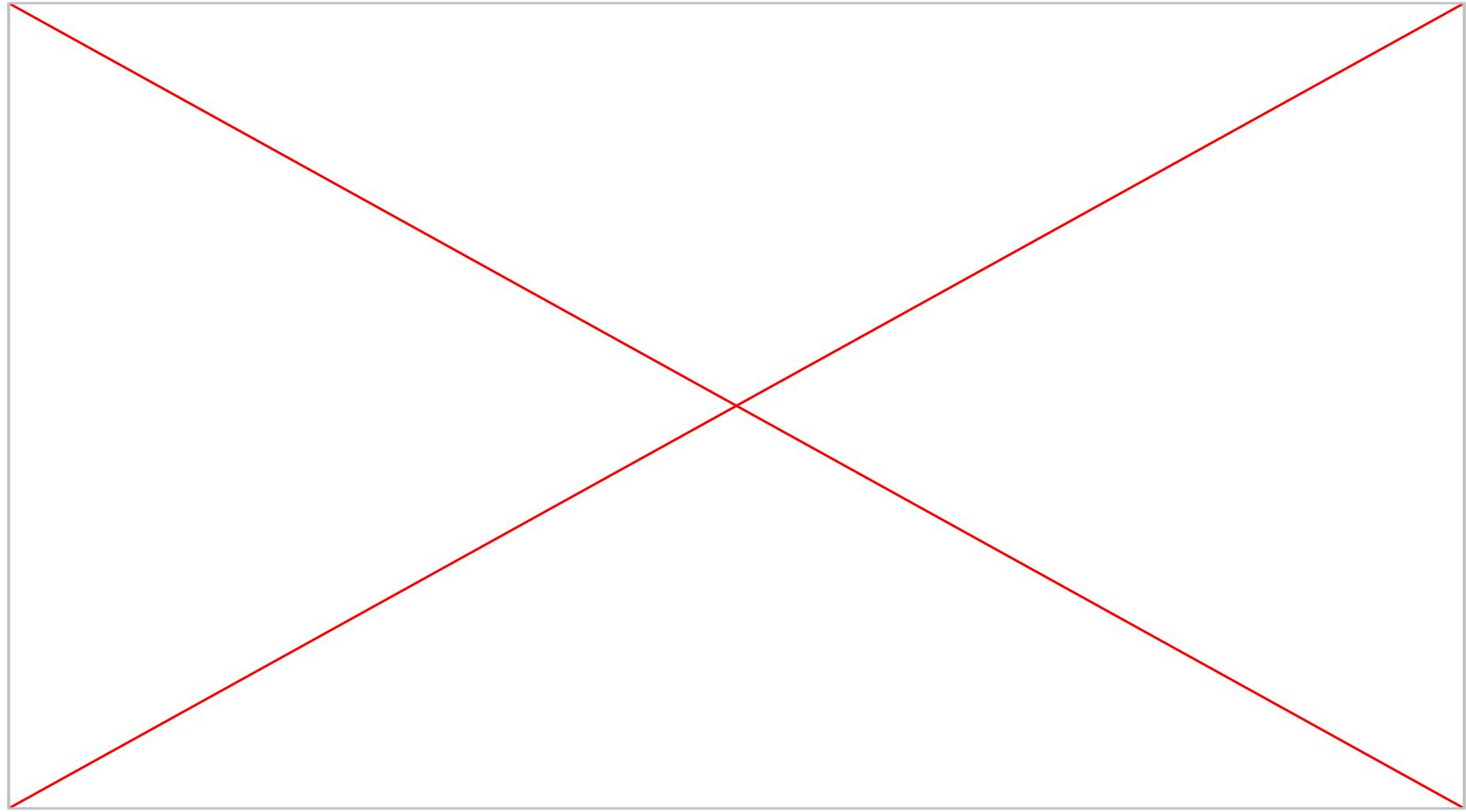


DEMO TIME

# Detecting a Twitch 0day\* 🔥

\* <https://www.kandji.io/blog/twitch-privileged-helper> by Chris Lopez 🐝





# This is not an ending note...

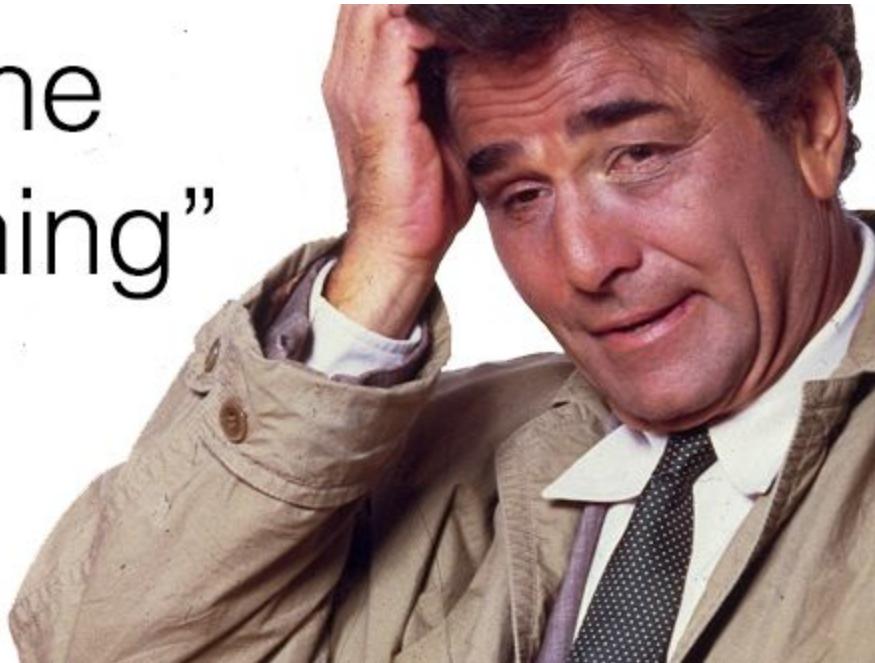
 Apple, help us detect XPC exploits and keep users safe!

`es_event_xpc_connect_t`

- Provide the hosting program path of the XPC service attempting to be connected to.
- Include code signing information for the process hosting the XPC connection (Team ID, etc).



“Just one  
more thing”



# macOS Vulnerability Research Training

by @theevilbit

Late Coming in  
Late 2025 / Early 2026





# Objective by the Sea

version 7.0

❤️ Thank you!

[github.com/Brandon7CC/mac-wheres-my-bootstrap/releases/](https://github.com/Brandon7CC/mac-wheres-my-bootstrap/releases/)