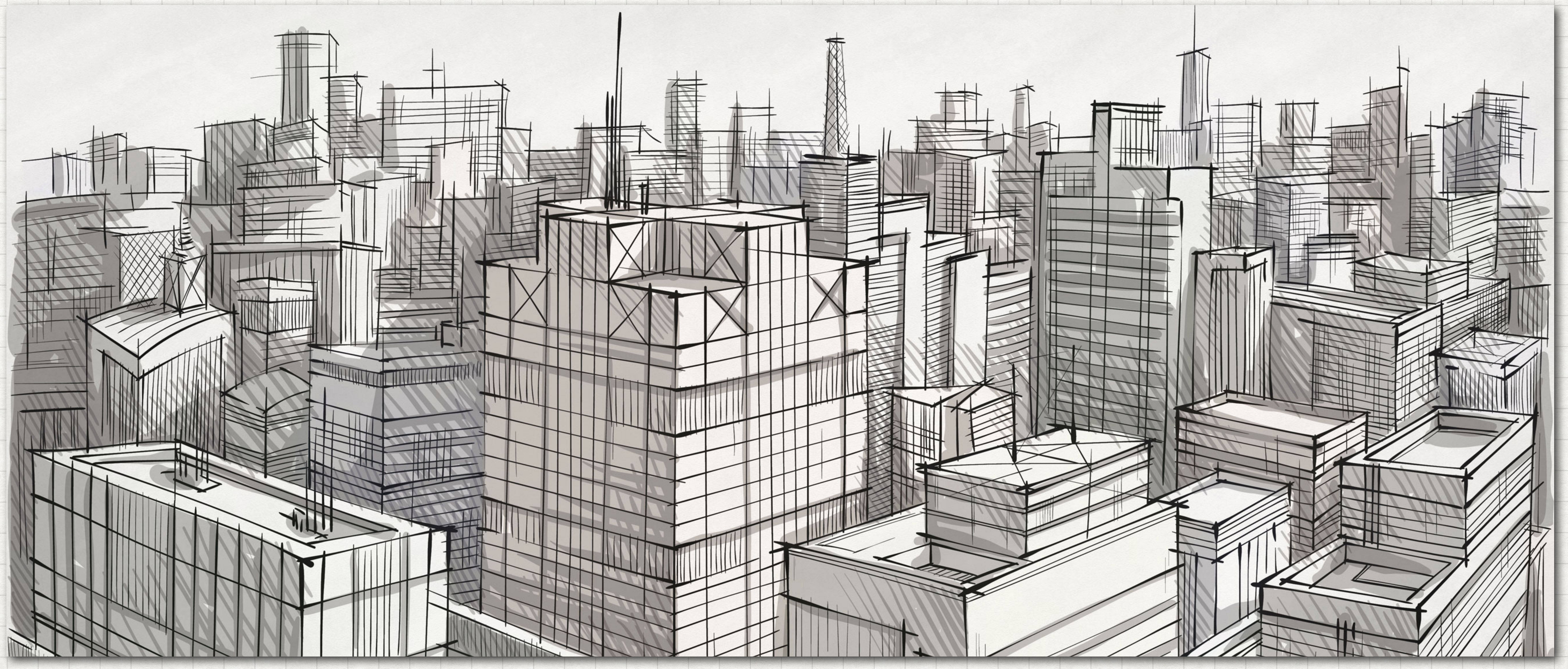
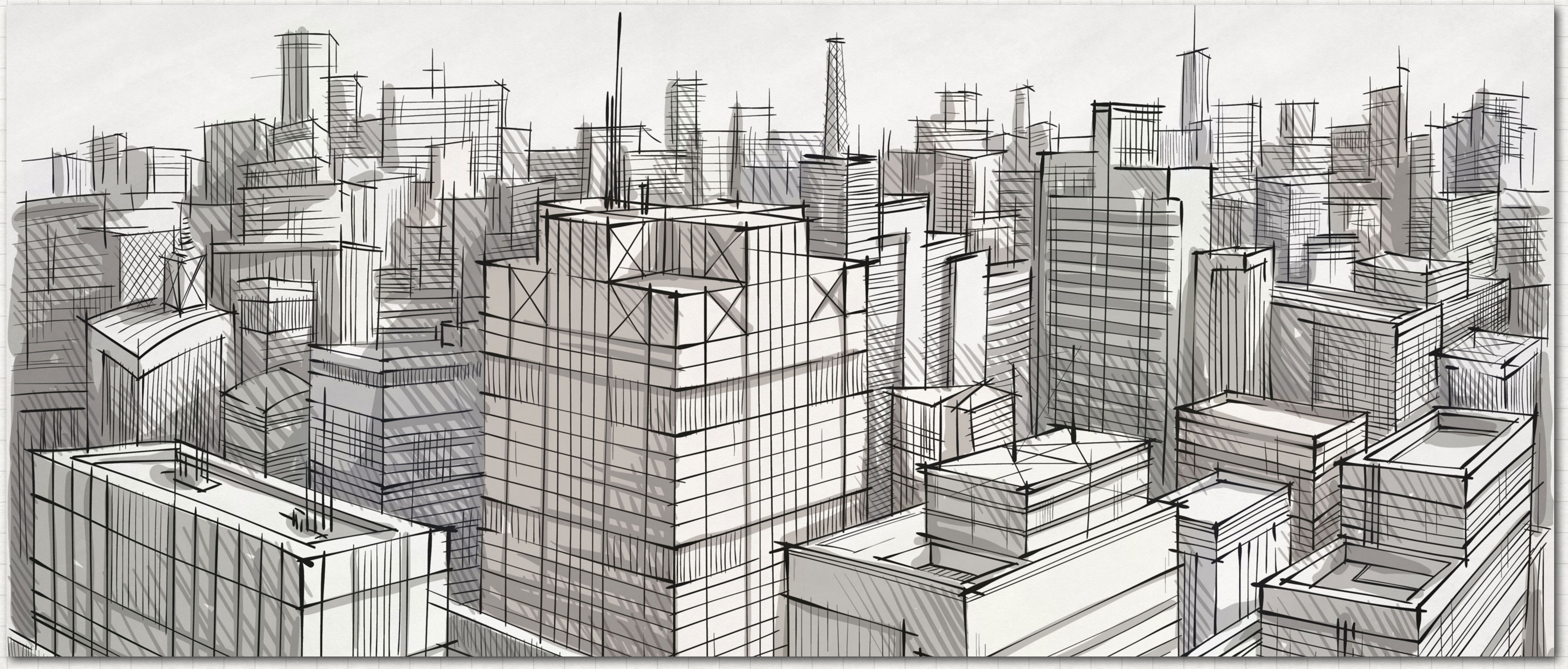
NOTARIZATION AND MACOS

WHAT IT DOES, WHY YOU NEED IT, AND HOW THE OS HANDLES IT WHEN YOU DON'T HAVE IT



THE LOYAL ORDER OF NOTARIES

HOW NOTARIZATION AFFECTS MAC ADMIN LIFE



TOM BRIDGE

(NOT A NOTARY PUBLIC)

@TBRIDGE

TECHNOLUTIONARY LLC

WHAT ARE WE HERE FOR?

SOME STRUCTURE

- What’s a Notary For?
- Gatekeeper
- SPCTL, Stapler, Other Tools
- Notarizing with Xcode
- Notarizing by hand
- Troubleshooting
- Summary

WHAT'S A NOTARY?

NOTARIES ARE GUIDEPOSTS

THERE TO SERVE AS TRUST ANCHORS FOR MEATSPACE

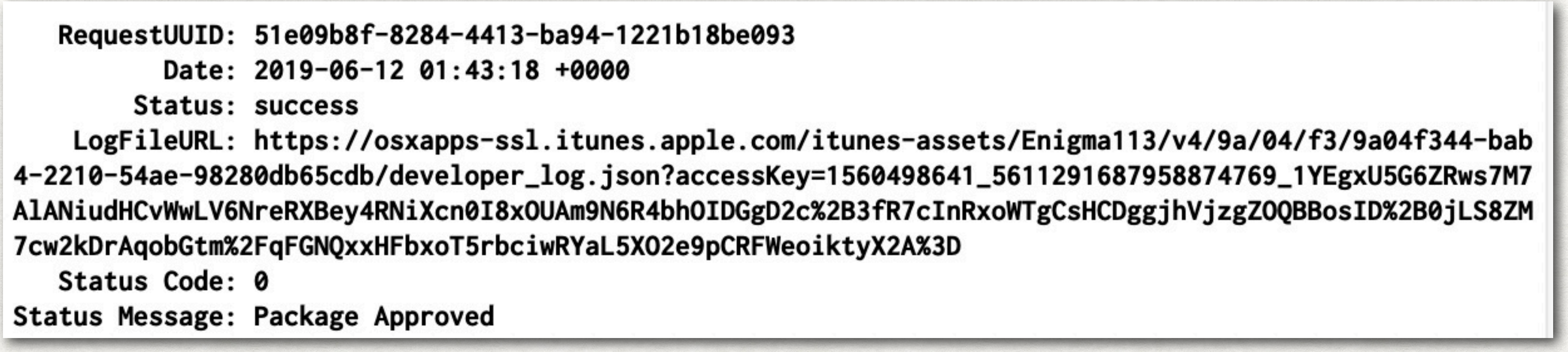
- Notaries are Human Certificate Authorities
- Notaries Verify Identity
- Affix their Seal
- Provides Third Party Trust

SO HOW DOES THAT WORK FOR SOFTWARE?

NOTARIZATION FOR SOFTWARE

A LAYER OF INTERACTION & TRUST

- Signed in Xcode or with `codesign`
- Signed objects are submitted to Apple for automated review
- Once reviewed, a ticket is created (or not!) for the signed binary
- Then the ticket can be stapled to the submitted object for offline review of the object

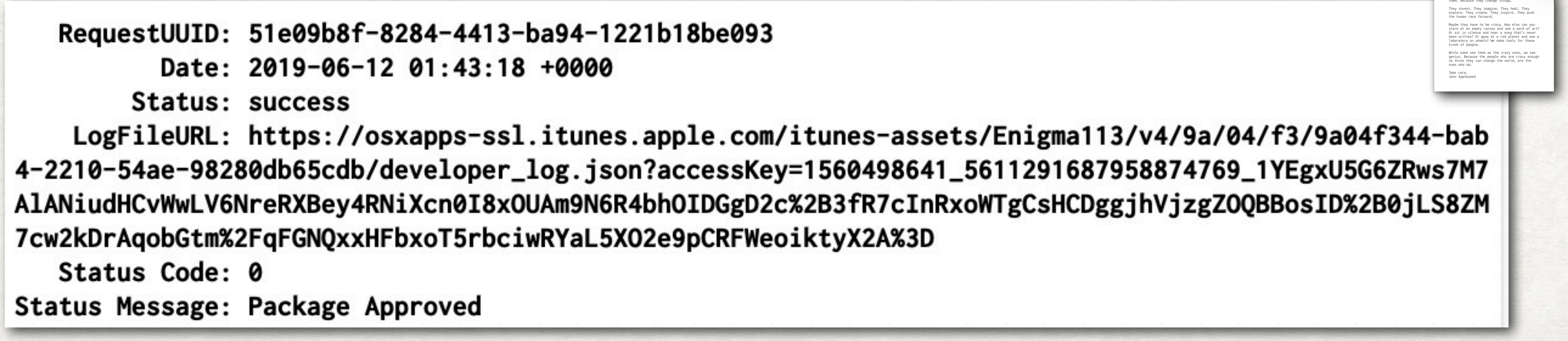


```
   RequestUUID: 51e09b8f-8284-4413-ba94-1221b18be093
          Date: 2019-06-12 01:43:18 +0000
        Status: success
    LogFileURL: https://osxapps-ssl.itunes.apple.com/itunes-assets/Enigma113/v4/9a/04/f3/9a04f344-bab
4-2210-54ae-98280db65cdb/developer_log.json?accessKey=1560498641_5611291687958874769_1YEgxU5G6ZRws7M7
AlANiudHCvWwLV6NreRXBey4RNiXcn0I8xOUAm9N6R4bhOIDGgD2c%2B3fR7cInRxoWTgCsHCDggjhVjzgZOQBBosID%2B0jLS8ZM
7cw2kDrAqobGtm%2FqFGNQxxHFbxoT5rbciwRYaL5XO2e9pCRFWeoiktyX2A%3D
   Status Code: 0
Status Message: Package Approved
```

NOTARIZATION FOR SOFTWARE

A LAYER OF INTERACTION & TRUST

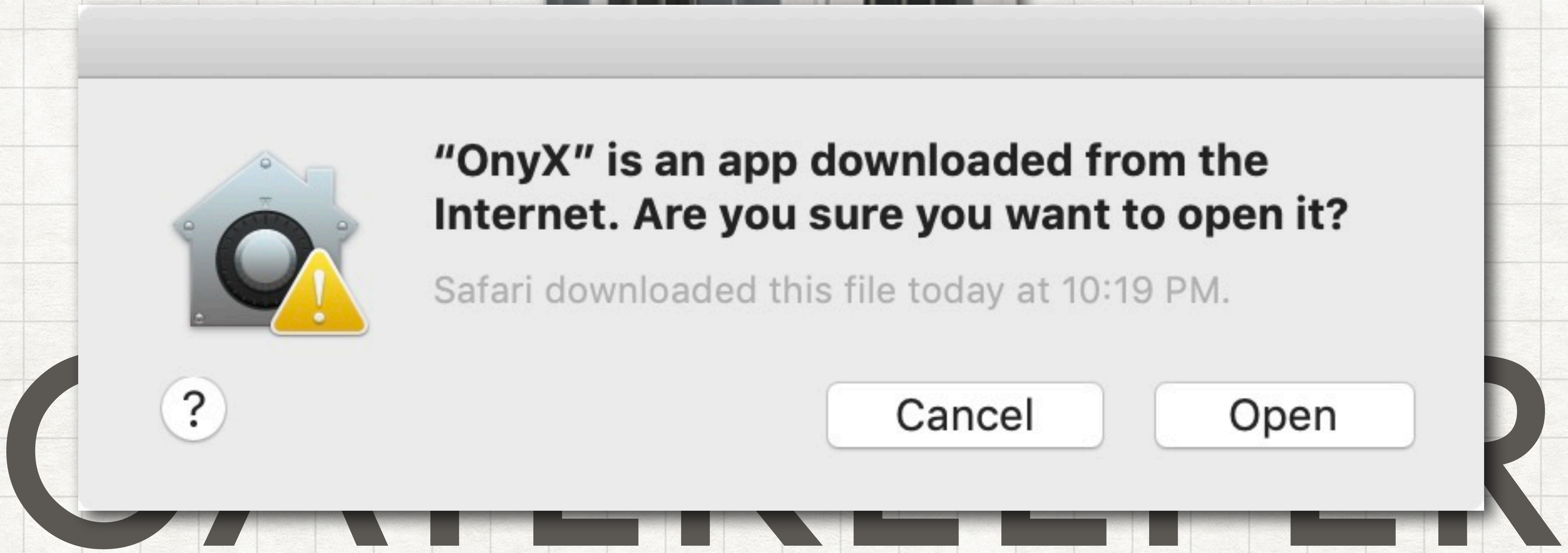
- Signed in Xcode or with `codesign`
- Signed objects are submitted to Apple for automated review
- Once reviewed, a ticket is created (or not!) for the signed binary
- Then the ticket can be stapled to the submitted object for offline review of the object



```
   RequestUUID: 51e09b8f-8284-4413-ba94-1221b18be093
          Date: 2019-06-12 01:43:18 +0000
        Status: success
    LogFileURL: https://osxapps-ssl.itunes.apple.com/itunes-assets/Enigma113/v4/9a/04/f3/9a04f344-bab
4-2210-54ae-98280db65cdb/developer_log.json?accessKey=1560498641_5611291687958874769_1YEgxU5G6ZRws7M7
AlANiudHCvWwLV6NreRXBey4RNiXcn0I8xOUAm9N6R4bhOIDGgD2c%2B3fR7cInRxoWTgCsHCDggjhVjzgZOQBBosID%2B0jLS8ZM
7cw2kDrAqobGtm%2FqFGNQxxHFbxoT5rbciwRYaL5XO2e9pCRFWeoiktyX2A%3D
   Status Code: 0
Status Message: Package Approved
```

TRUST MATTERS

GATEKEEPER



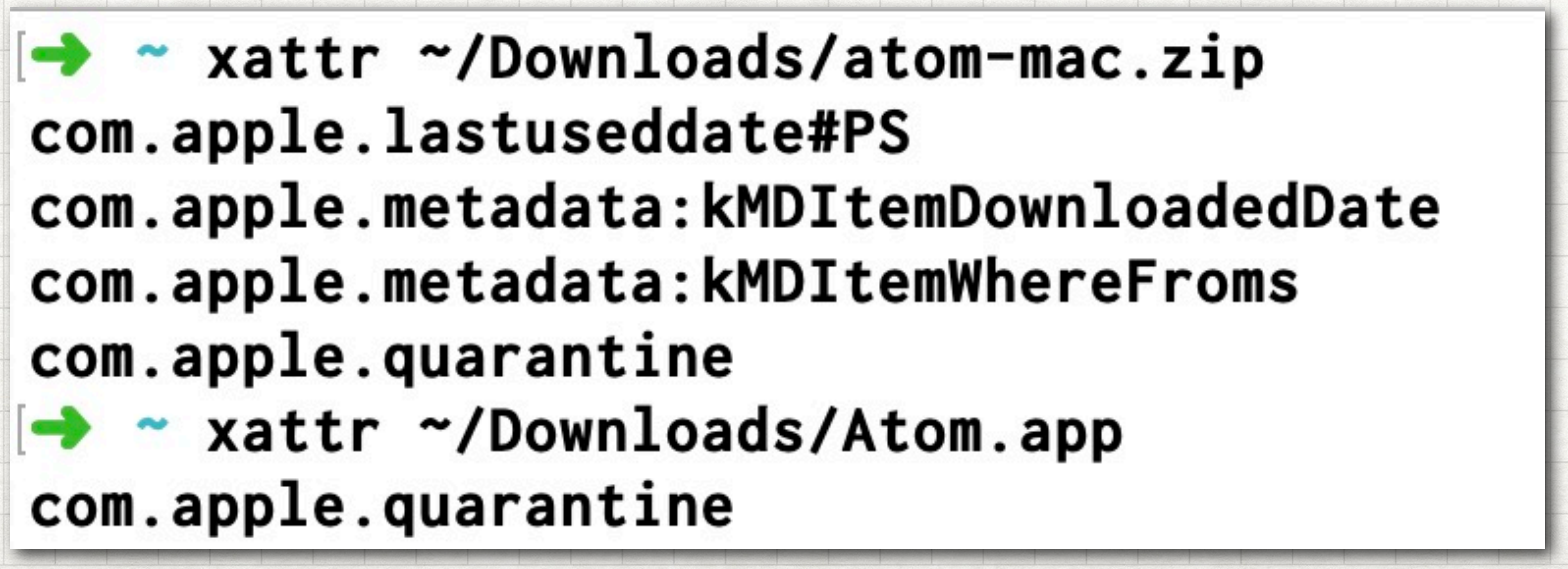
"OnyX" is an app downloaded from the Internet. Are you sure you want to open it?

Safari downloaded this file today at 10:19 PM.

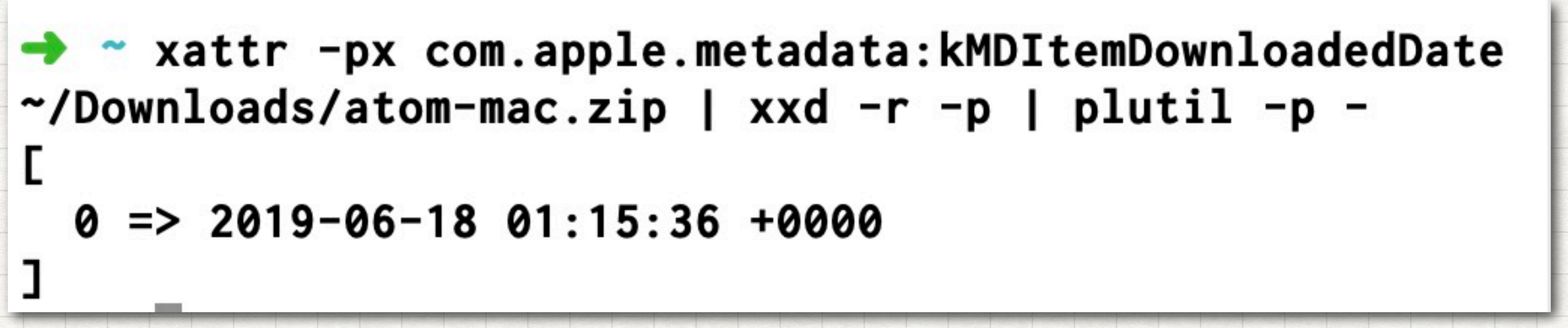
GATEKEEPER

QUARANTINE

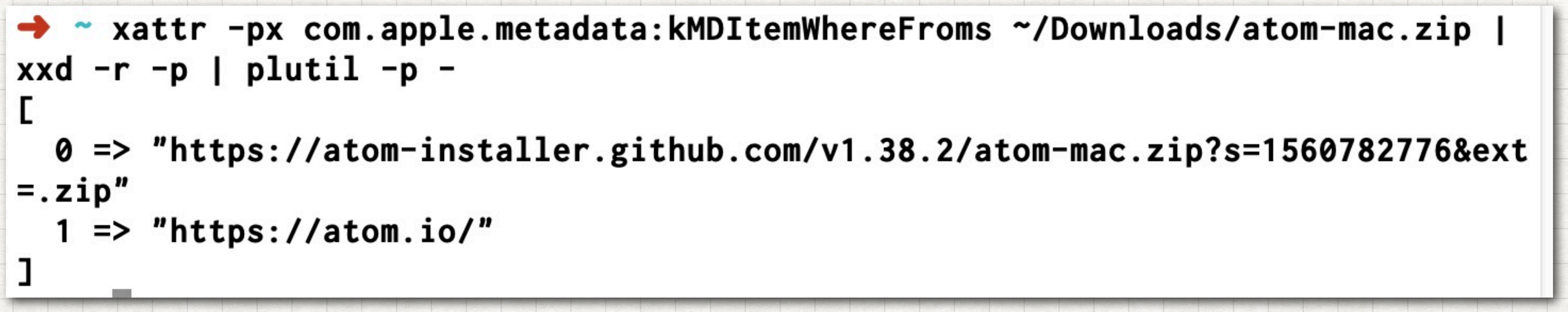
QUARANTINE



```
[➜  ~ xattr ~/Downloads/atom-mac.zip
com.apple.lastuseddate#PS
com.apple.metadata:kMDItemDownloadedDate
com.apple.metadata:kMDItemWhereFroms
com.apple.quarantine
[➜  ~ xattr ~/Downloads/Atom.app
com.apple.quarantine
```



```
➜  ~ xattr -px com.apple.metadata:kMDItemDownloadedDate
~/Downloads/atom-mac.zip | xxd -r -p | plutil -p -
[
  0 => 2019-06-18 01:15:36 +0000
]
```



```
➜  ~ xattr -px com.apple.metadata:kMDItemWhereFroms ~/Downloads/atom-mac.zip |
xxd -r -p | plutil -p -
[
  0 => "https://atom-installer.github.com/v1.38.2/atom-mac.zip?s=1560782776&ext
=.zip"
  1 => "https://atom.io/"
]
```

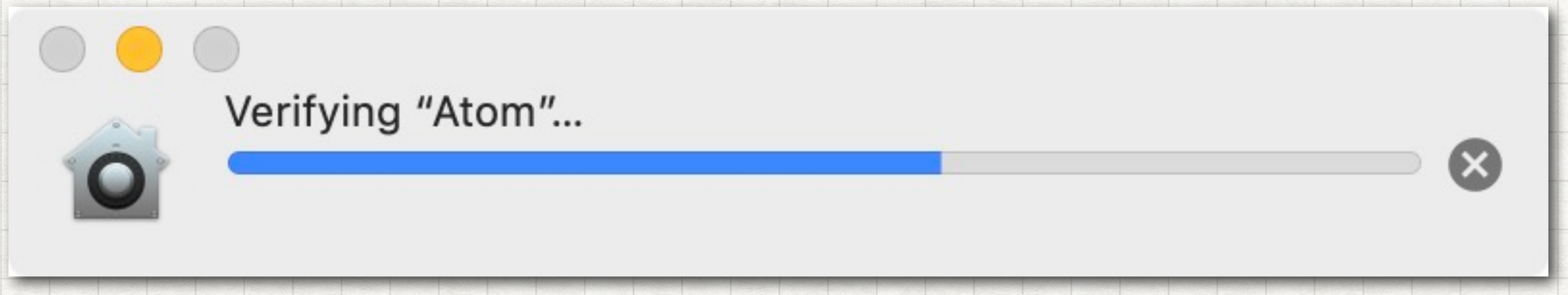
QUARANTINE

```
xattr -px com.apple.quarantine ~/
Downloads/atom-mac.zip | xxd -r -p

0083;5d083b38;Safari;20B94076-
EFD4-4A9D-809B-41B72B91DD03
```

QUARANTINE

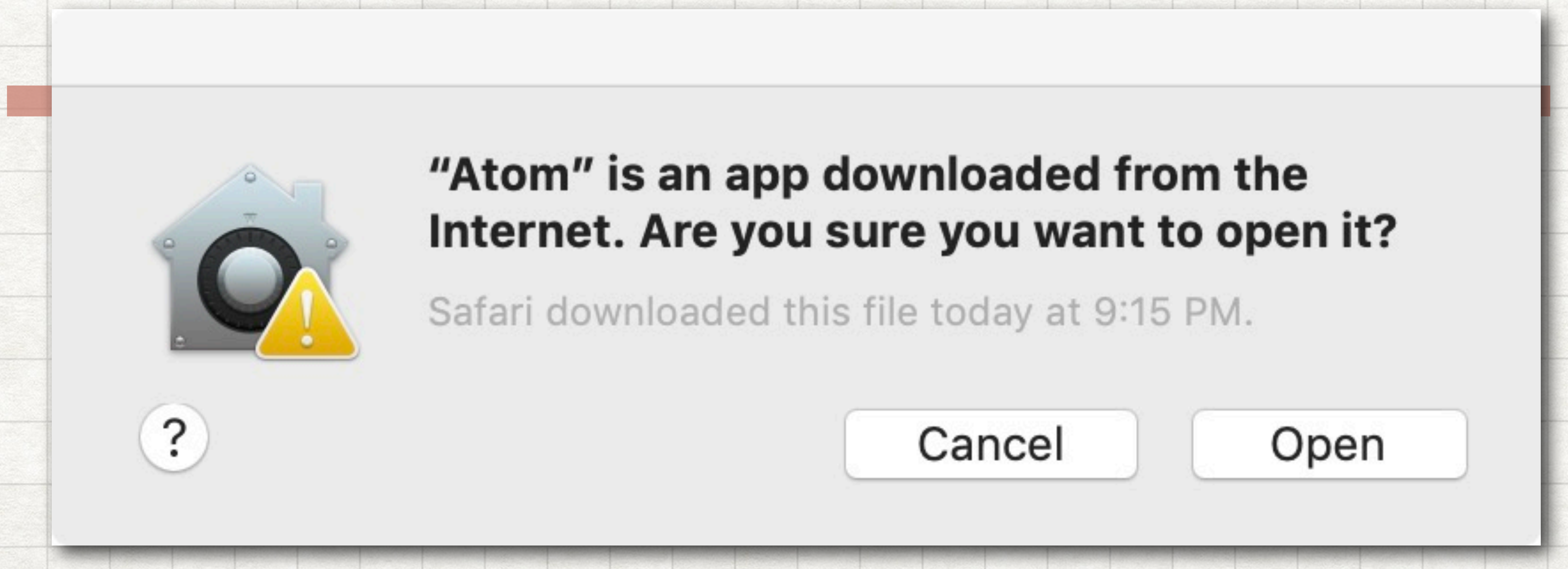
SCORE	TIME	APP	UUID
0083;	5d083b38;	Safari;	20B94076-EFD4-4A9D-809B-41B72B91DD03

QUARANTINE


Verifying “Atom”...

SCORE

0083


“Atom” is an app downloaded from the Internet. Are you sure you want to open it?

Safari downloaded this file today at 9:15 PM.

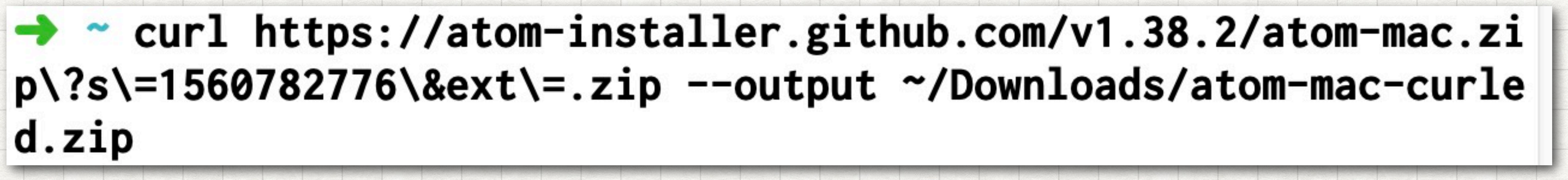
SCORE

01C3

QUARANTINE

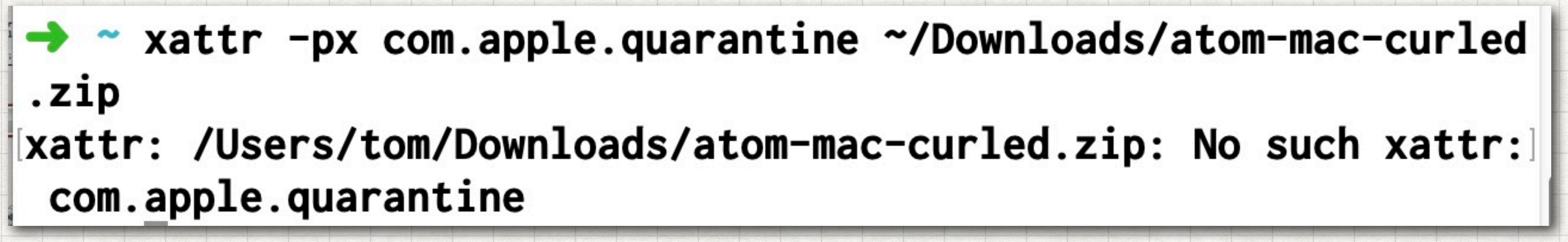
BUT WHAT IF
I CHEAT?

QUARANTINE



```
➜  ~ curl https://atom-installer.github.com/v1.38.2/atom-mac.zip\?s\=1560782776\&ext\=.zip --output ~/Downloads/atom-mac-curled.zip
```

QUARANTINE



```
➜ ~ xattr -px com.apple.quarantine ~/Downloads/atom-mac-curled
.zip
xattr: /Users/tom/Downloads/atom-mac-curled.zip: No such xattr:
 com.apple.quarantine
```

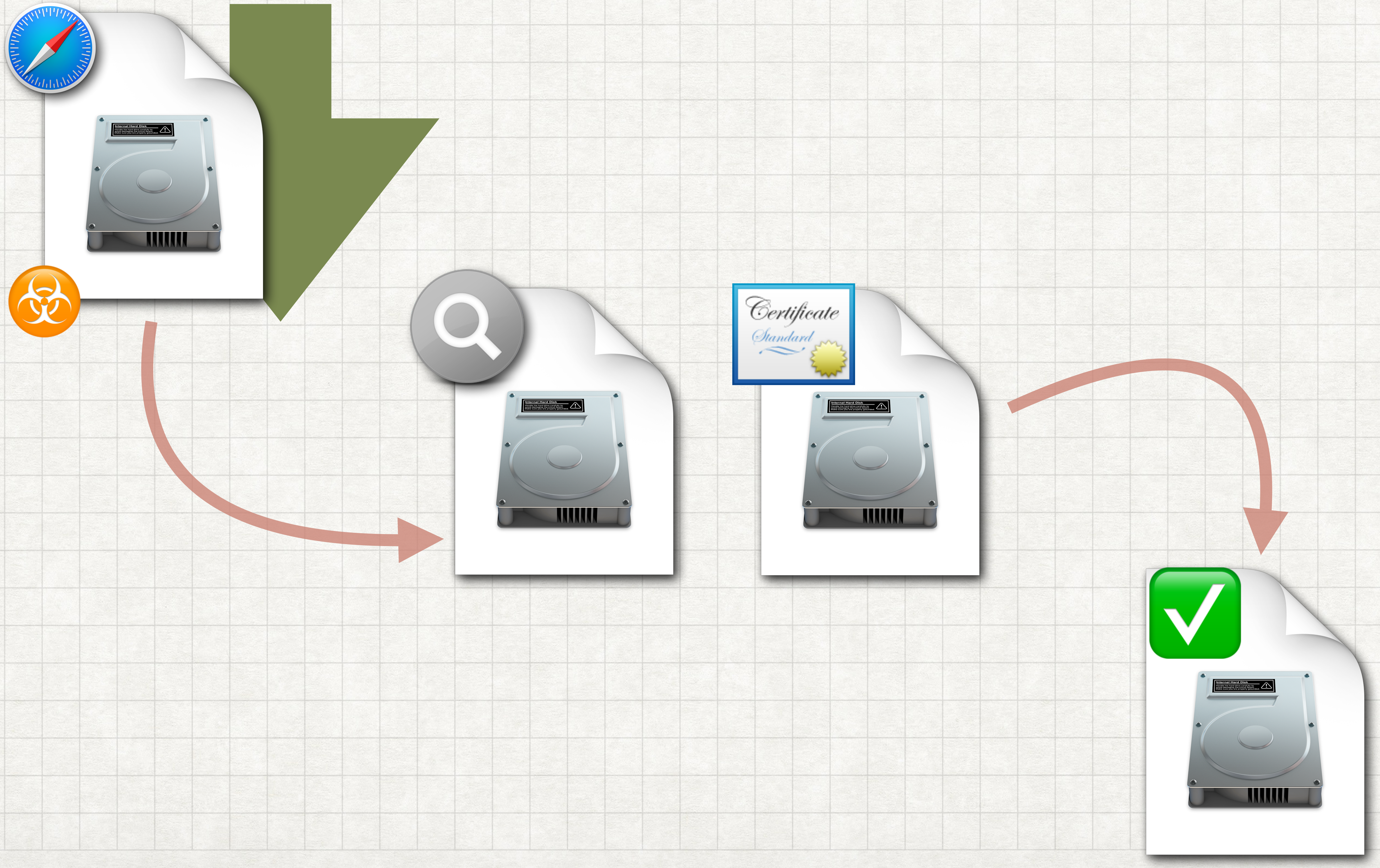
LET’S SYNTHESIZE

GATEKEEPER (10.14.4)

- Operates on downloaded objects with executable code from Safari, Chrome, Mail or other standard web applications that have opted into the quarantine schema
- Operates on items opened in Finder from External Volumes, Disk Images, or AirDropped files.
- Conducts an XProtect Scan of the executable code for malicious actors or other banned malware.
- Reviews the signature of the executable code for chain-of-trust. Signatures can resolve up to the App Store trust roots or the Developer ID trust roots, and differentiation exists.
- If all passes, quarantine flags are changed, Launch Services Database updated.

THE GOGGLES DO NOTHING

GATEKEEPER (10.14.4)



GATEKEEPER (10.14.4)

USER ACTIONS

- Opens File Stream
- Tacks On Quarantine Flag
- Opens Disk Image in Finder
- Drags App to /Applications
- Opens App.

GATEKEEPER (10.14.4)

LAUNCH SERVICES ACTIONS

- XProtect Scan
- Notes Quarantine Flag & Scans Trust
- Identifies Signed App (Developer ID)
- Dialog

GATEKEEPER (10.14.4)

ACCEPTANCE & LAUNCH

- Dialog
- Quarantine Flag Changed
- App Location Stored
- LaunchServices Database Updated

GATEKEEPER (10.14.5)

- Operates on downloaded objects with executable code from Safari, Chrome, Mail or other standard web applications that have opted into the quarantine schema

- Operates on items delivered by External Volumes, Disk Images, or AirDropped files.

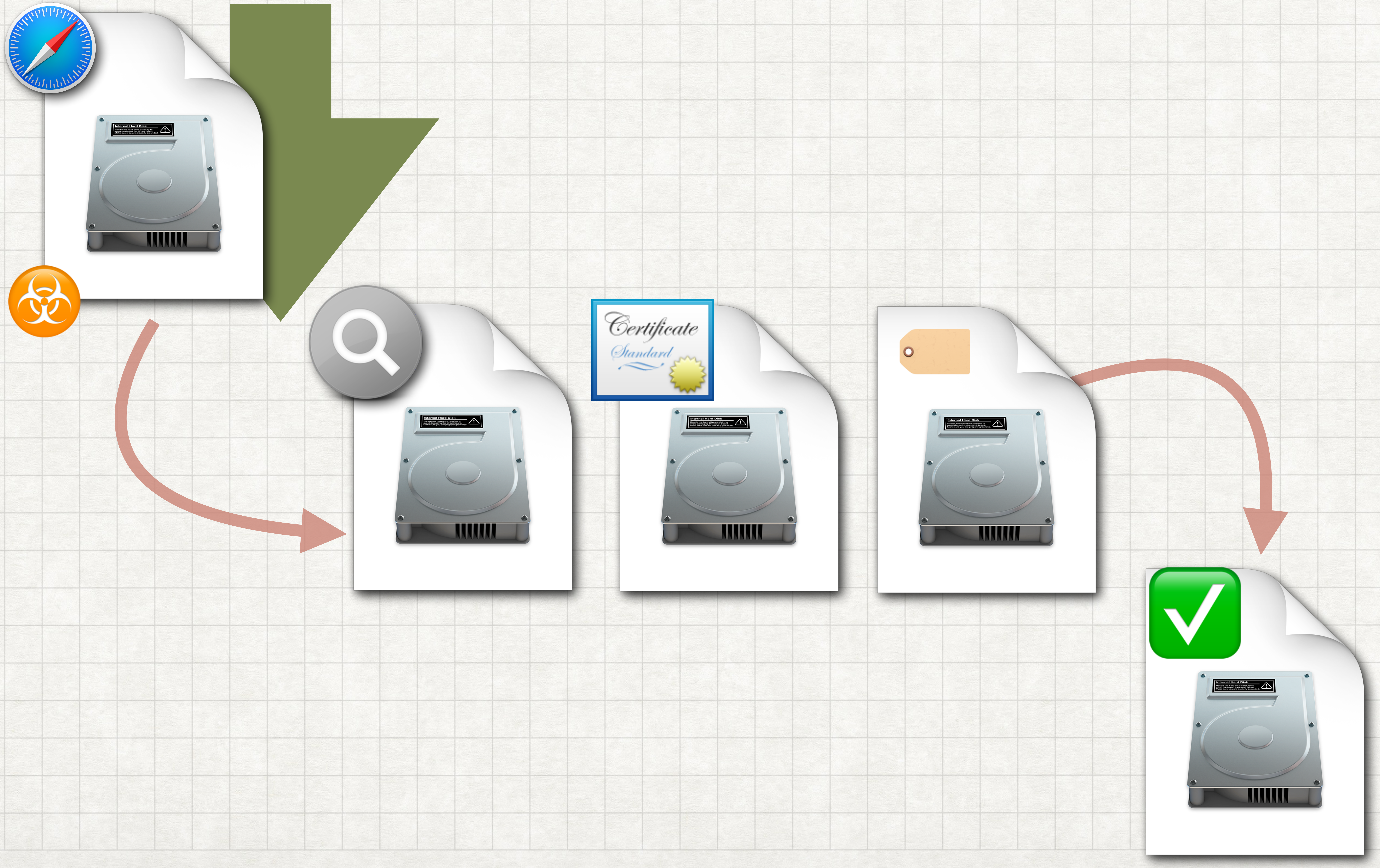
- Conducts an XProtect Scan of the executable code for malicious actors or other banned malware.

- Reviews the signature of the executable code for chain-of-trust and integrity. Signatures can resolve up to the App Store trust roots or the Developer ID trust roots, and differentiation exists.

- **Reviews executable code for notarization, compares local ticket (if present) to downloaded ticket from CloudKit, and to executable code.**

- If all passes, quarantine flags are changed, Launch Services Database updated.

GATEKEEPER (10.14.5)

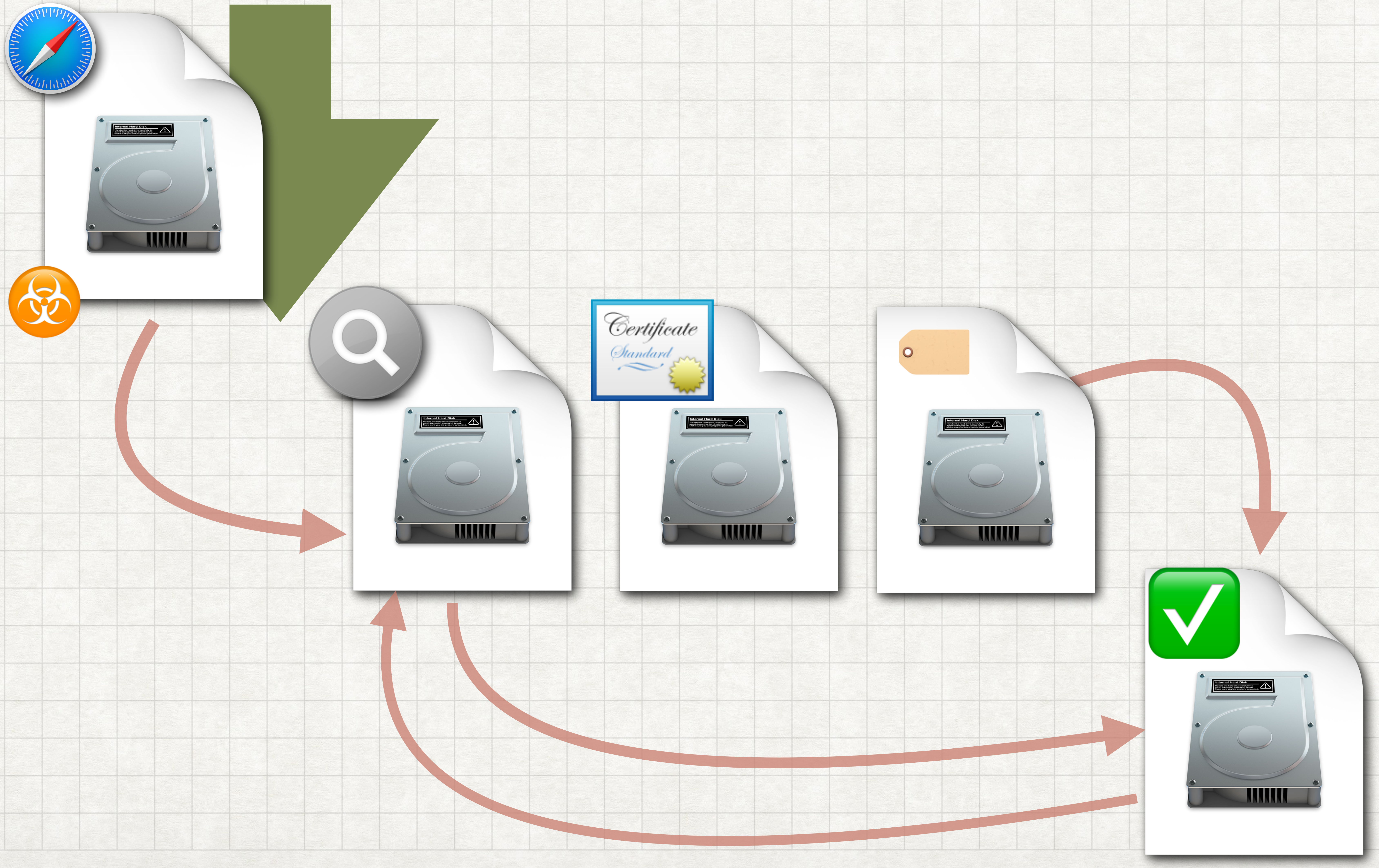


GATEKEEPER (10.15)

- Operates on downloaded objects with executable code from Safari, Chrome, Mail or other standard web applications
- Operates on items delivered by External Volumes, Disk Images, or AirDropped files.
- Conducts an XProtect Scan of the executable code for malicious actors or other banned malware.
- Reviews the signature of the executable code for chain-of-trust and integrity. Signatures can resolve up to the App Store trust roots or the Developer ID trust roots, and differentiation exists.
- Reviews executable code for notarization, compares local ticket (if present) to downloaded ticket from CloudKit, and to executable code.
- If all passes, quarantine flags are changed, Launch Services Database updated.
- **On future launches, XProtect scans are run again.**

	First Use, Quarantined	>_ First Use, Quarantined	Non-Quarantined >_
Malicious Content Scan	No Known Malicious Content	No Known Malicious Content	No Known Malicious Content
Signature Check	No Tampering	No Tampering	
Notarization Check	Notarization Required	Notarization Required	
First Launch Prompt	User Must Approve	User Must Approve Software in Bundles	

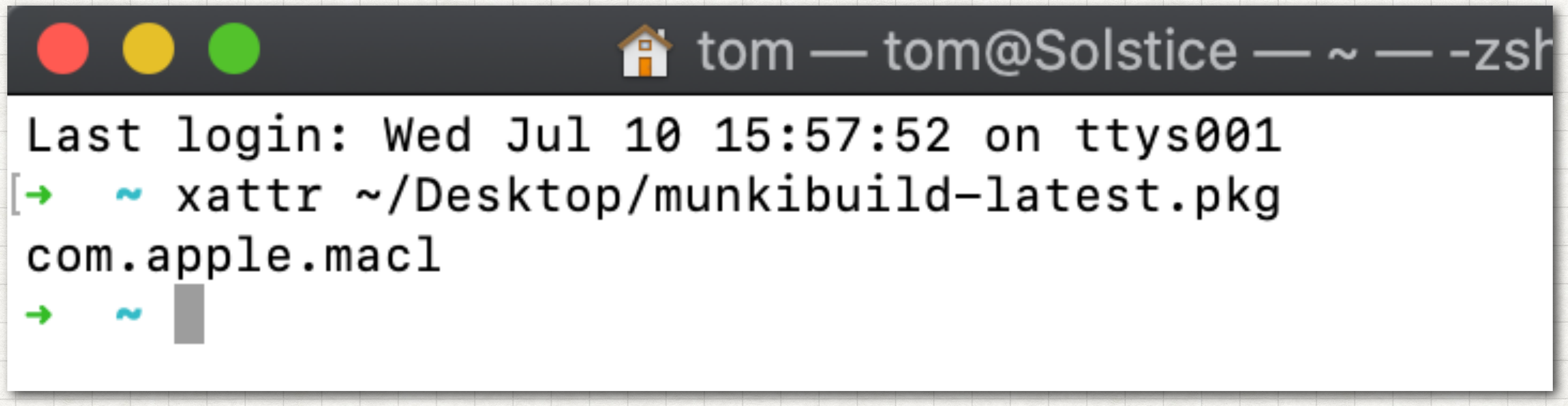
GATEKEEPER (10.15)



QUARANTINE

BUT WHAT IF
I CHEAT?

QUARANTINE

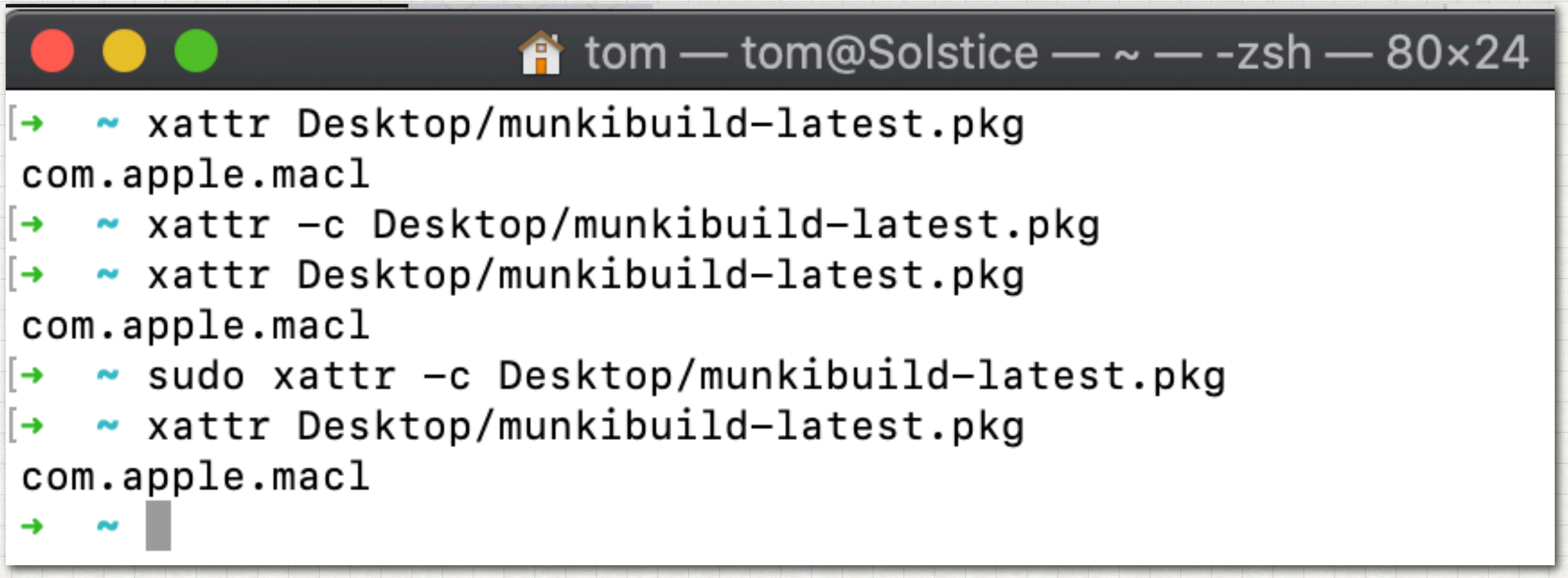


```
Last login: Wed Jul 10 15:57:52 on ttys001
[→  ~ xattr ~/Desktop/munkibuild-latest.pkg
com.apple.macl
→  ~
```

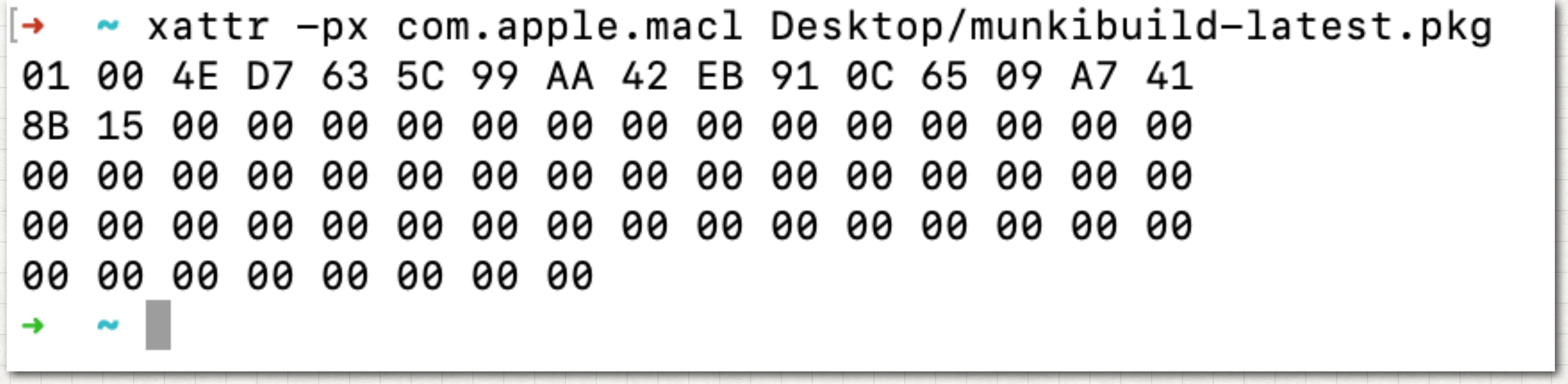
COM.APPLE.MACL

WHAT'S THAT?

I DON'T PRECISELY KNOW.



```
[→  ~ xattr Desktop/munkibuild-latest.pkg
com.apple.macl
[→  ~ xattr -c Desktop/munkibuild-latest.pkg
[→  ~ xattr Desktop/munkibuild-latest.pkg
com.apple.macl
[→  ~ sudo xattr -c Desktop/munkibuild-latest.pkg
[→  ~ xattr Desktop/munkibuild-latest.pkg
com.apple.macl
→  ~ 
```



```
[➜  ~ xattr -px com.apple.macl Desktop/munkibuild-latest.pkg
01 00 4E D7 63 5C 99 AA 42 EB 91 0C 65 09 A7 41
8B 15 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
➜  ~ 
```

THIS IS AS-OF-YET UNDOCUMENTED

XPROTECT

GATEKEEPER (10.16+)

?

GATEKEEPER (10.16+)

"You can always choose to run any software on your system."

Just Not By Default.

Advances in macOS Security, Session 701, WWDC 2019

GATEKEEPER (10.16+)

"In a future version of macOS, unsigned code **will not run** by default."

Advances in macOS Security, Session 701, WWDC 2019

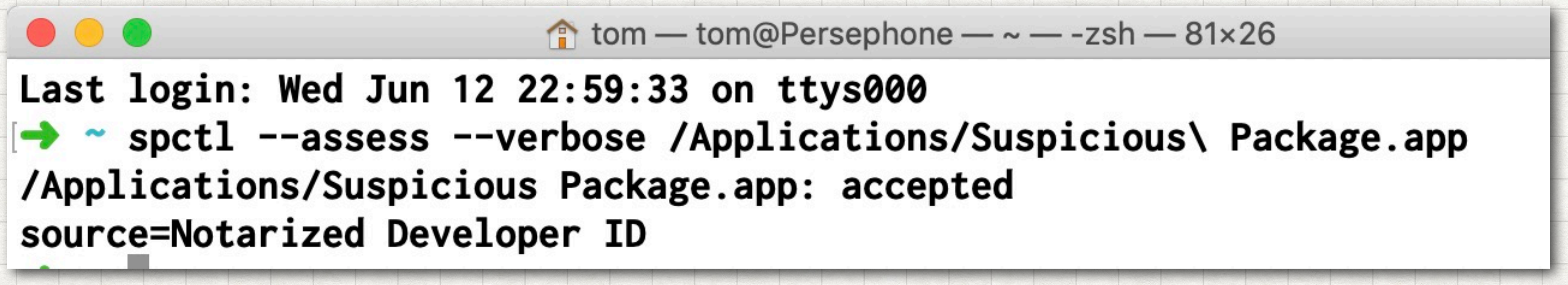
I THOUGHT THIS WAS ABOUT NOTARIZATION, TOM?

SPCTL

SPCTL

System Policy Control Binary

- —Assess for evaluating binaries
- —Verbose for detail
- —reset-default for when you mess up



```
Last login: Wed Jun 12 22:59:33 on ttys000
➜ ~ spctl --assess --verbose /Applications/Suspicious\ Package.app
/Applications/Suspicious Package.app: accepted
source=Notarized Developer ID
```

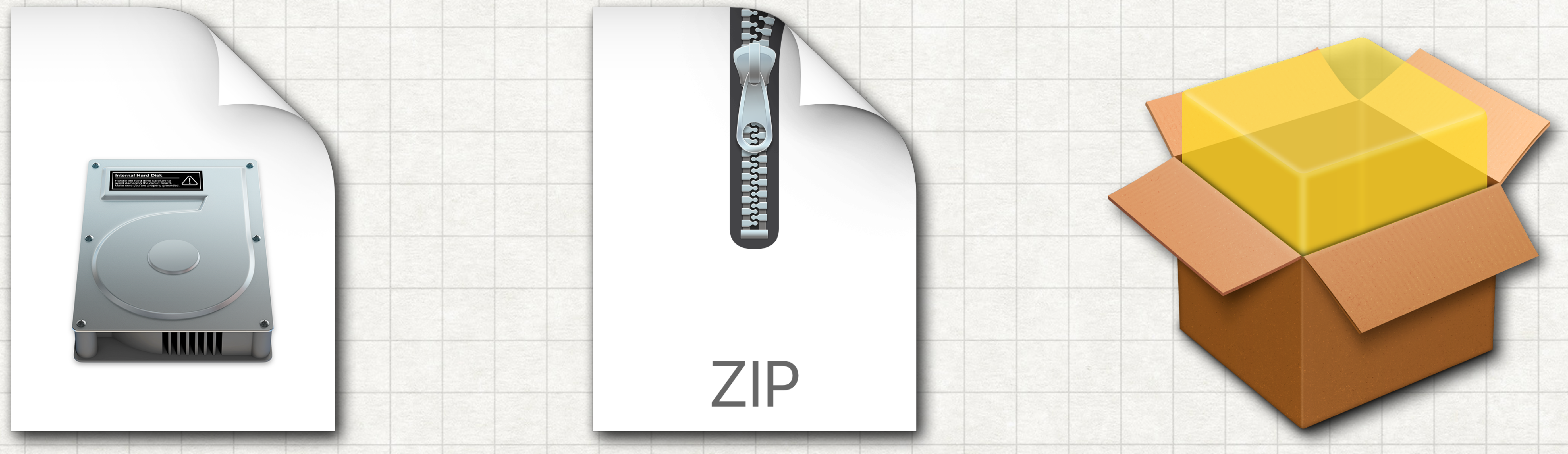
STAPLER

Office Space ©2017FOX

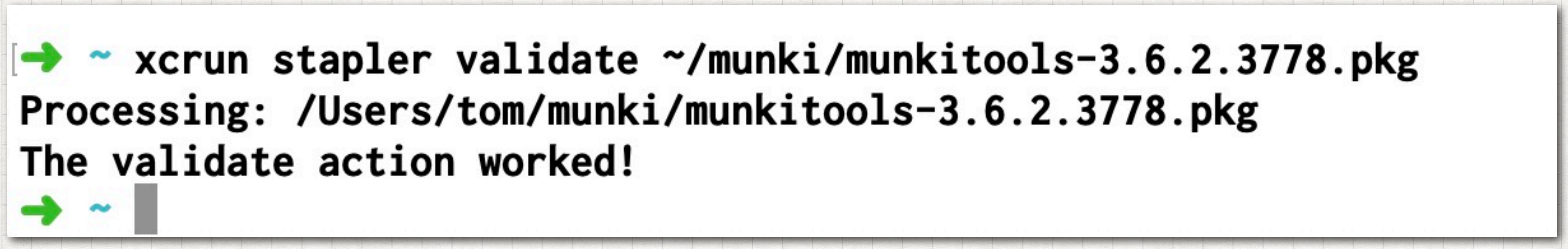
STAPLER

STAPLER

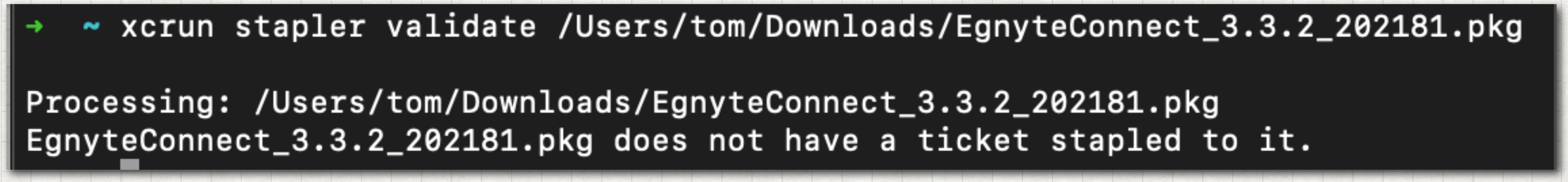
- `xcrun stapler`
- `—staple` stapling tickets to existing objects
- `—validate` for checks of existing objects



STAPLER

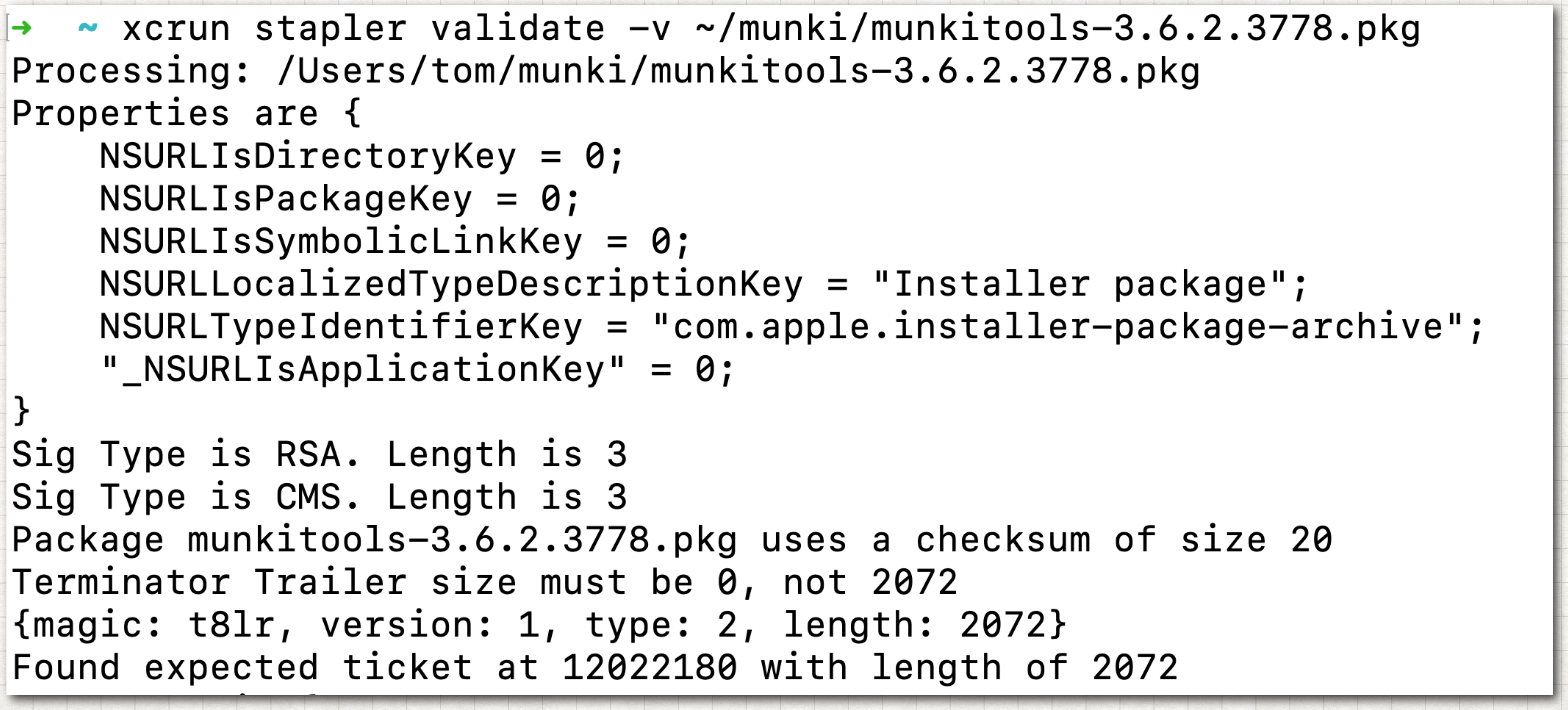


```
➜  ~ xcrun stapler validate ~/munki/munkitools-3.6.2.3778.pkg
Processing: /Users/tom/munki/munkitools-3.6.2.3778.pkg
The validate action worked!
➜  ~ 
```



```
➜  ~ xcrun stapler validate /Users/tom/Downloads/EgnyteConnect_3.3.2_202181.pkg

Processing: /Users/tom/Downloads/EgnyteConnect_3.3.2_202181.pkg
EgnyteConnect_3.3.2_202181.pkg does not have a ticket stapled to it.
```

STAPLER



```
➜  ~ xcrun stapler validate -v ~/munki/munkitools-3.6.2.3778.pkg
Processing: /Users/tom/munki/munkitools-3.6.2.3778.pkg
Properties are {
    NSURLIsDirectoryKey = 0;
    NSURLIsPackageKey = 0;
    NSURLIsSymbolicLinkKey = 0;
    NSURLLocalizedTypeDescriptionKey = "Installer package";
    NSURLTypeIdentifierKey = "com.apple.installer-package-archive";
    "_NSURLIsApplicationKey" = 0;
}
Sig Type is RSA. Length is 3
Sig Type is CMS. Length is 3
Package munkitools-3.6.2.3778.pkg uses a checksum of size 20
Terminator Trailer size must be 0, not 2072
{magic: t8lr, version: 1, type: 2, length: 2072}
Found expected ticket at 12022180 with length of 2072
```

SO HOW'S THIS WORK EXACTLY?

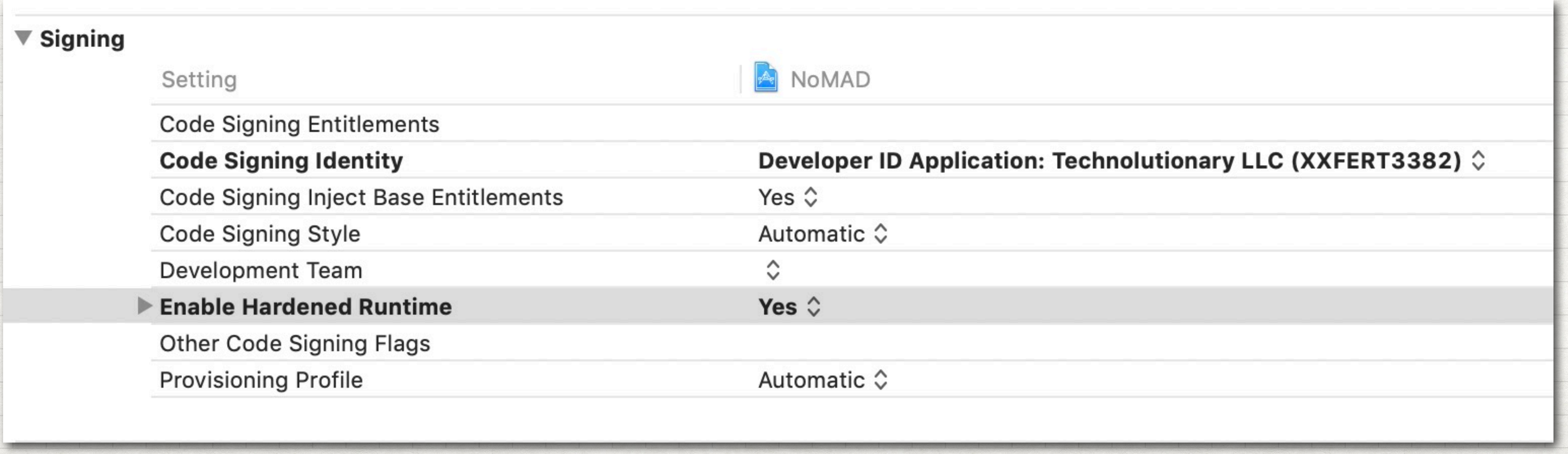
YOU & ME, WE GOT A COOL GENERATOR

STEP 1

SIGN YOUR APP

SIGN YOUR APP

- Use the Hardened Runtime
- Sign it with a Developer ID Application Certificate



▼ Signing

Setting	NoMAD
Code Signing Entitlements	
Code Signing Identity	**Developer ID Application: Technolutionary LLC (XXFERT3382)**
Code Signing Inject Base Entitlements	Yes
Code Signing Style	Automatic
Development Team	
▶ **Enable Hardened Runtime**	**Yes**
Other Code Signing Flags	
Provisioning Profile	Automatic

STEP 2

SUBMIT YOUR APP TO APPLE

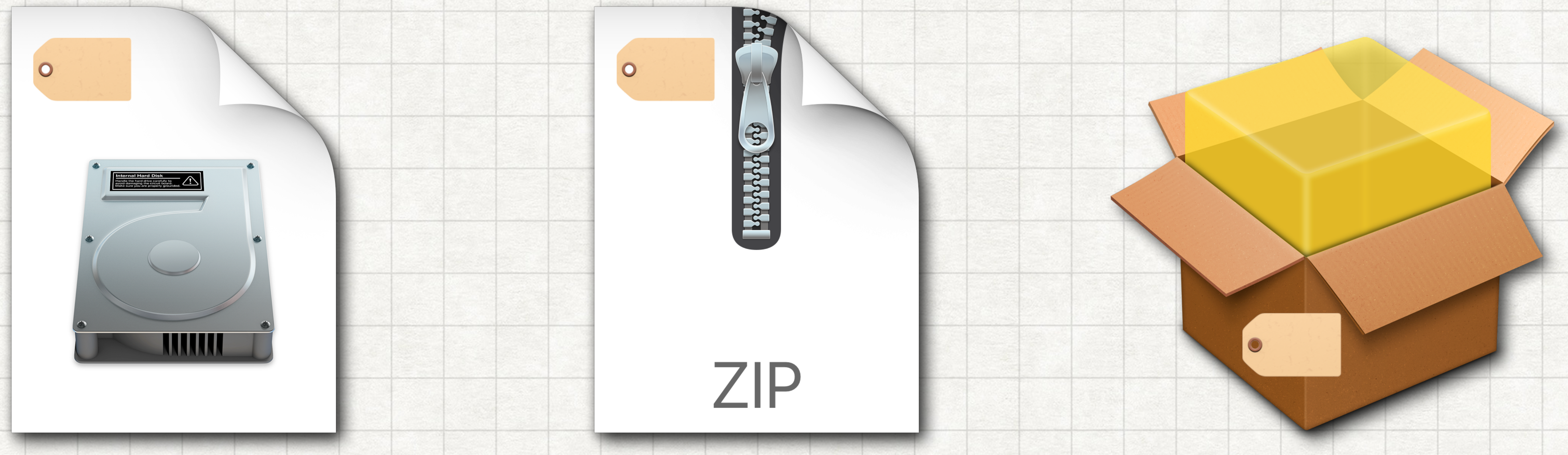
STEP 2

```
>_
xcrun altool
```

STEP 3

WAIT

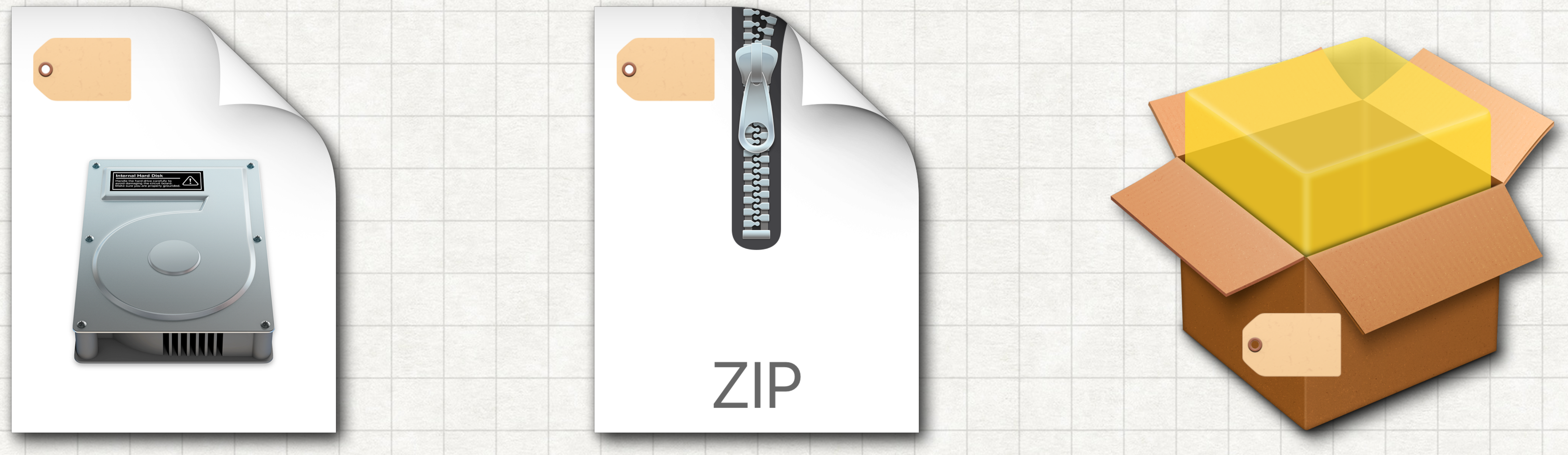
STEP 4



STAPLE YOUR TICKET

NO TICKET!

STEP 4



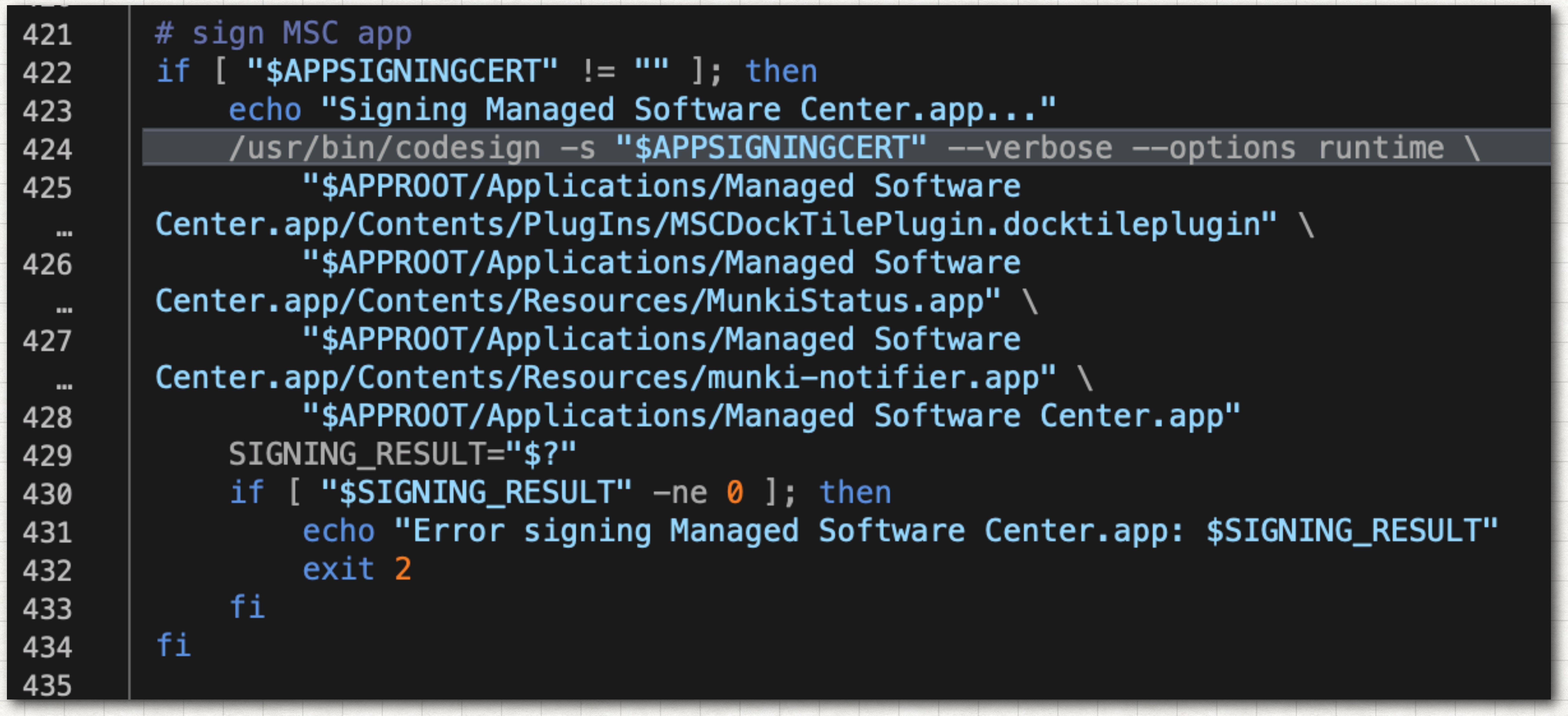
STAPLE YOUR TICKET

NOTARIZATION AT THE COMMAND LINE

WHAT YOU NEED TO NOTARIZE

- An Application-Specific Password for your Developer Apple ID
- A Zip, DMG or PKG file containing (your) signed software
- Xcode & Xcode Command Line Tools

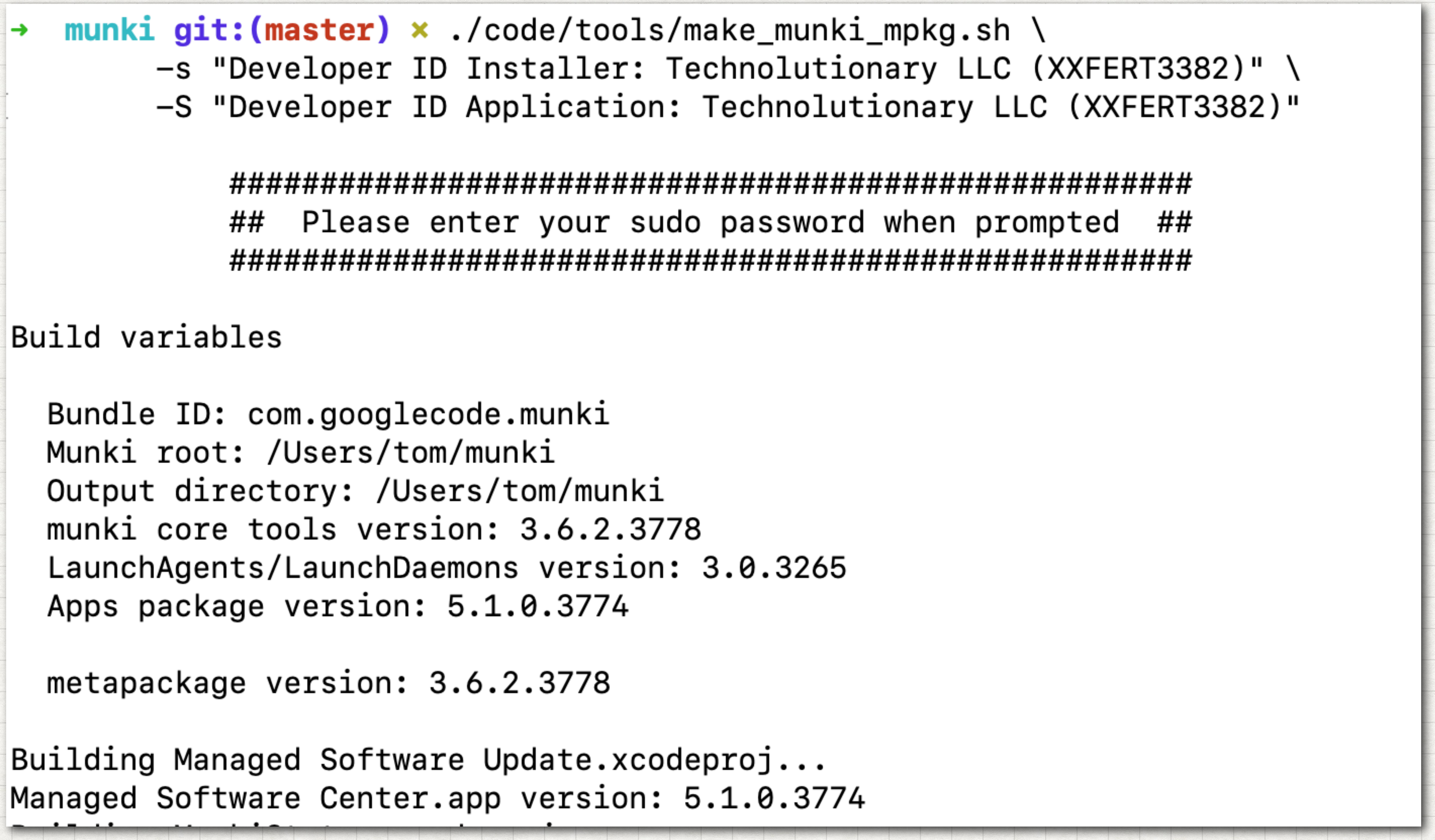
AN EXAMPLE: SIGNING & NOTARIZING MUNKI



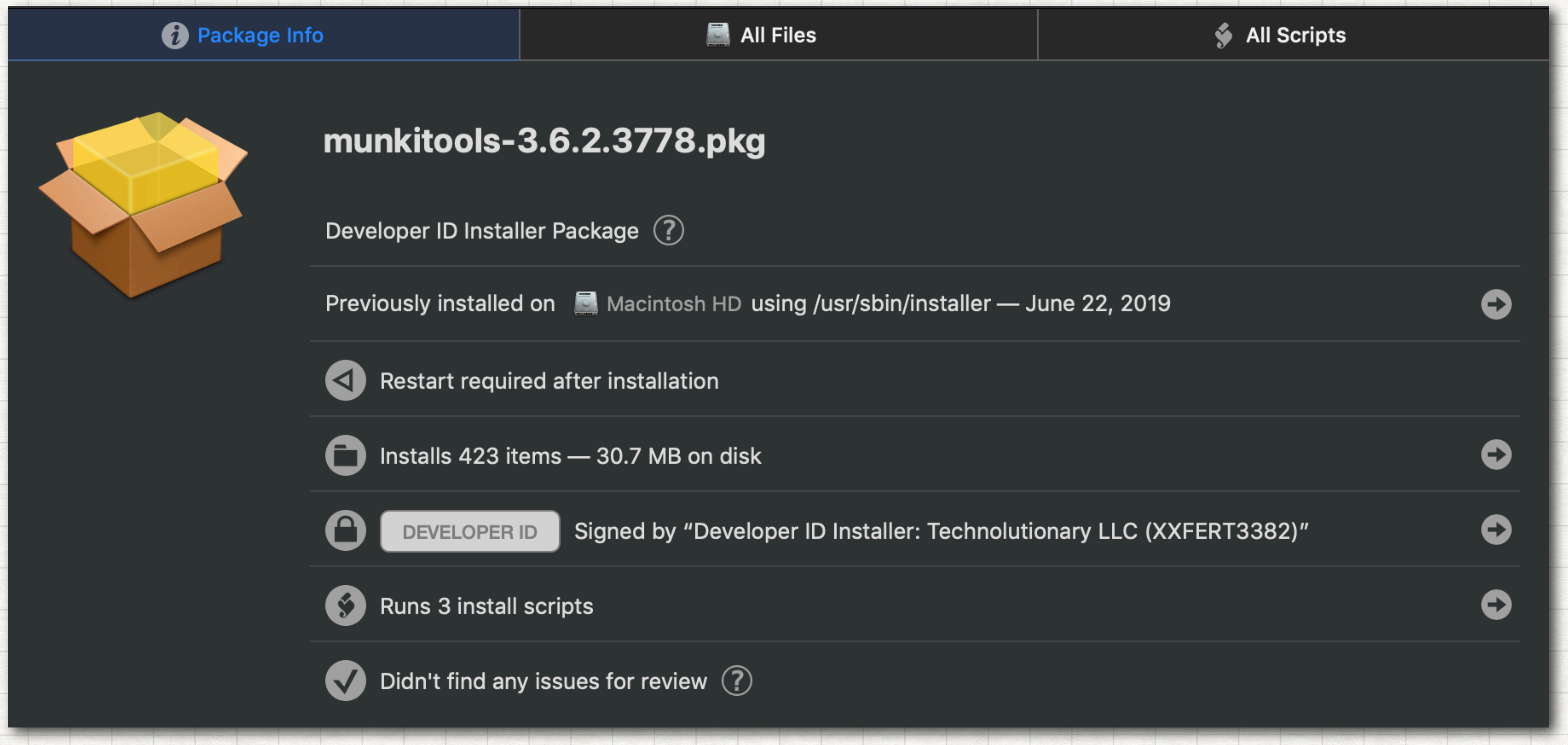
```
# sign MSC app
if [ "$APPSIGNINGCERT" != "" ]; then
    echo "Signing Managed Software Center.app..."
    /usr/bin/codesign -s "$APPSIGNINGCERT" --verbose --options runtime \
        "$APPROOT/Applications/Managed Software Center.app/Contents/PlugIns/MSCDockTilePlugin.docktileplugin" \
        "$APPROOT/Applications/Managed Software Center.app/Contents/Resources/MunkiStatus.app" \
        "$APPROOT/Applications/Managed Software Center.app/Contents/Resources/munki-notifier.app" \
        "$APPROOT/Applications/Managed Software Center.app"
    SIGNING_RESULT="$?"
    if [ "$SIGNING_RESULT" -ne 0 ]; then
        echo "Error signing Managed Software Center.app: $SIGNING_RESULT"
        exit 2
    fi
fi
```

~/munki/code/tools/make_munki_mpkg.sh

```
➜  munki git:(master) ✗ ./code/tools/
make_munki_mpkg.sh \
        -s "Developer ID Installer: 
Technolutionary LLC (XXFERT3382)" \
        -S "Developer ID Application: 
Technolutionary LLC (XXFERT3382)"
```



```
➜  munki git:(master) ✗ ./code/tools/make_munki_mpkg.sh \
        -s "Developer ID Installer: Technolutionary LLC (XXFERT3382)" \
        -S "Developer ID Application: Technolutionary LLC (XXFERT3382)"

            #####################################################
            ##  Please enter your sudo password when prompted  ##
            #####################################################

Build variables

  Bundle ID: com.googlecode.munki
  Munki root: /Users/tom/munki
  Output directory: /Users/tom/munki
  munki core tools version: 3.6.2.3778
  LaunchAgents/LaunchDaemons version: 3.0.3265
  Apps package version: 5.1.0.3774

  metapackage version: 3.6.2.3778

Building Managed Software Update.xcodeproj...
Managed Software Center.app version: 5.1.0.3774
```



munkitools-3.6.2.3778.pkg

Developer ID Installer Package

Previously installed on Macintosh HD using /usr/sbin/installer — June 22, 2019

Restart required after installation

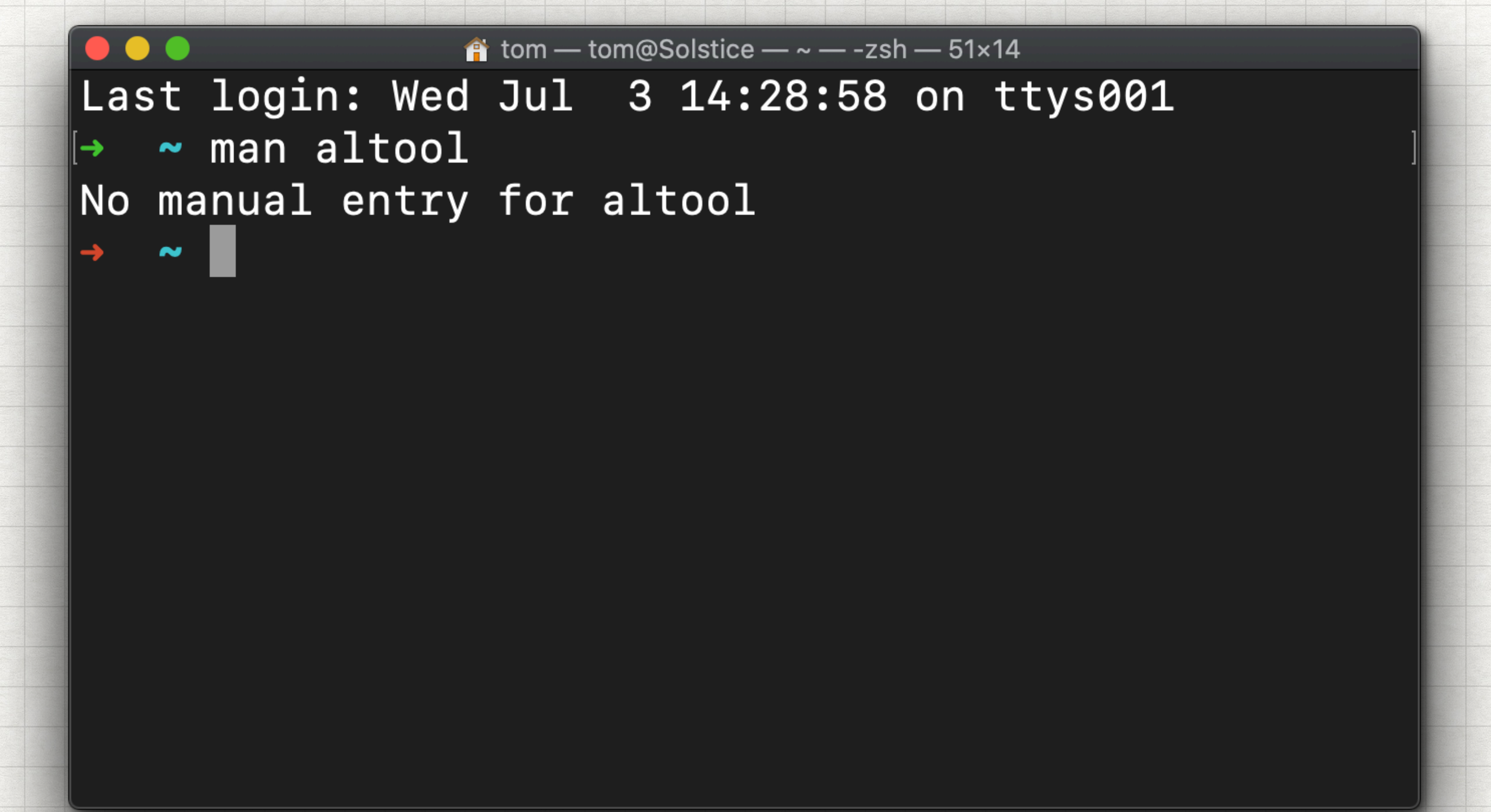
Installs 423 items — 30.7 MB on disk

DEVELOPER ID Signed by “Developer ID Installer: Technolutionary LLC (XXFERT3382)”

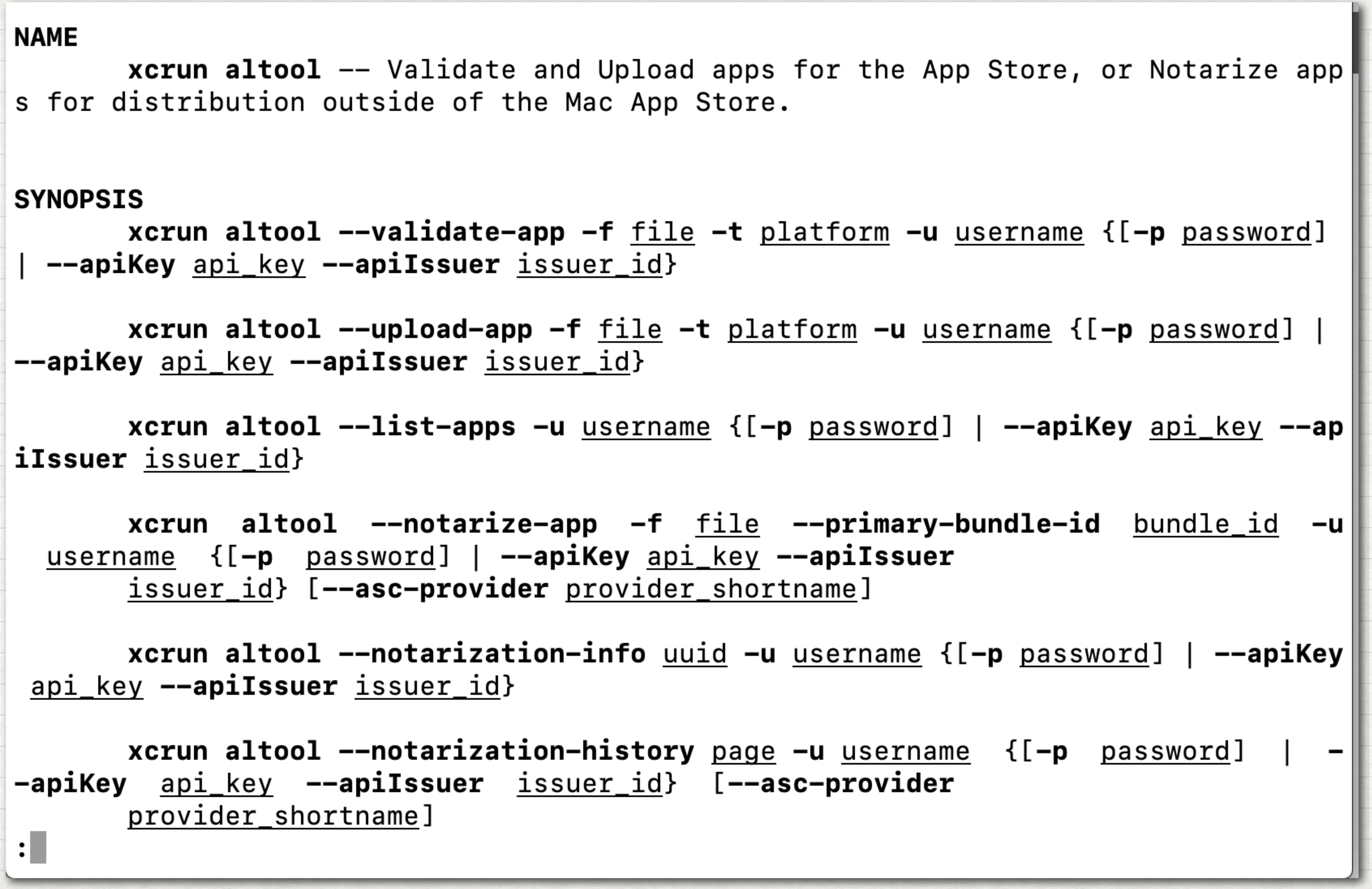
Runs 3 install scripts

Didn't find any issues for review

LET’S NOTARIZE



```
Last login: Wed Jul  3 14:28:58 on ttys001
➜  ~ man altool
No manual entry for altool
➜  ~ 
```



```
NAME
       xcrun altool -- Validate and Upload apps for the App Store, or Notarize app
s for distribution outside of the Mac App Store.


SYNOPSIS
       xcrun altool --validate-app -f file -t platform -u username {[-p password]
| --apiKey api_key --apiIssuer issuer_id}

       xcrun altool --upload-app -f file -t platform -u username {[-p password] |
--apiKey api_key --apiIssuer issuer_id}

       xcrun altool --list-apps -u username {[-p password] | --apiKey api_key --ap
iIssuer issuer_id}

       xcrun  altool  --notarize-app  -f  file  --primary-bundle-id  bundle_id  -u
  username  {[-p  password] | --apiKey api_key --apiIssuer
       issuer_id} [--asc-provider provider_shortname]

       xcrun altool --notarization-info uuid -u username {[-p password] | --apiKey
 api_key --apiIssuer issuer_id}

       xcrun altool --notarization-history page -u username  {[-p  password]  |  -
-apiKey  api_key  --apiIssuer  issuer_id}  [--asc-provider
       provider_shortname]
:
```

altool v1.1

```
--validate-app
--upload-app
--notarize-app
--notarization-info
--notarization-history
```

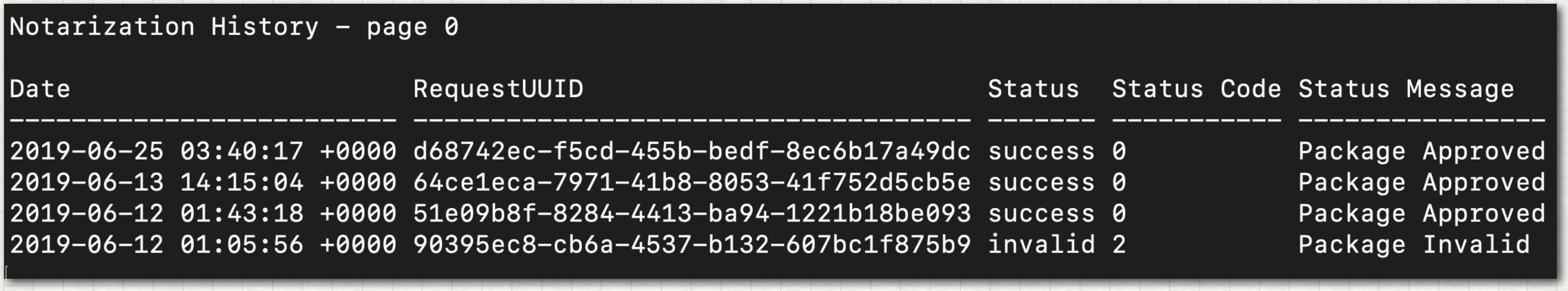
altool v4.0

```
--validate-app
--upload-app
--list-apps
--notarize-app
--notarization-info
--notarization-history
--store-password-in-keychain-item
```

```
xcrun altool --notarize-app \
  -f ~/munki/munkitools-3.6.2.3778.pkg \
  --primary-bundle-id com.googlecode.munki \
  -u tom_bridge@mac.com -p abba-cddc-effe-ghhg
```

```
   RequestUUID: d68742ec-f5cd-455b-bedf-8ec6b17a49dc
          Date: 2019-06-25 03:40:17 +0000
        Status: success
    LogFileURL: <snipped really long URL goes here>
   Status Code: 0
Status Message: Package Approved
```

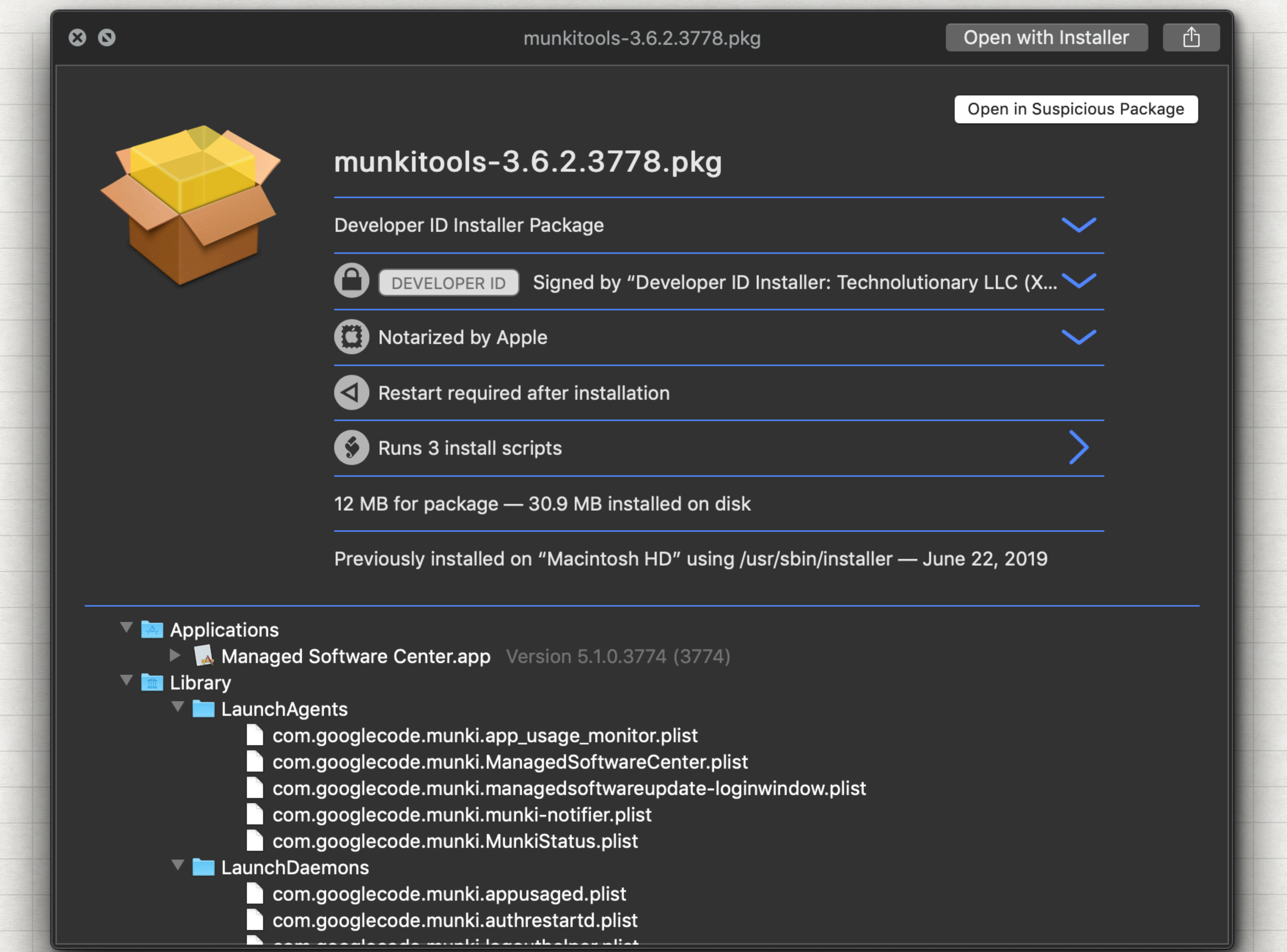
```
xcrun altool --notarization-history \
  -u tom_bridge@mac.com -p abba-cddc-effe-ghhg
```



```
Notarization History - page 0

Date                      RequestUUID                          Status  Status Code Status Message
------------------------- ------------------------------------ ------- ----------- ----------------
2019-06-25 03:40:17 +0000 d68742ec-f5cd-455b-bedf-8ec6b17a49dc success 0           Package Approved
2019-06-13 14:15:04 +0000 64ce1eca-7971-41b8-8053-41f752d5cb5e success 0           Package Approved
2019-06-12 01:43:18 +0000 51e09b8f-8284-4413-ba94-1221b18be093 success 0           Package Approved
2019-06-12 01:05:56 +0000 90395ec8-cb6a-4537-b132-607bc1f875b9 invalid 2           Package Invalid
```

```
xcrun altool --notarization-info \
  d68742ec-f5cd-455b-bedf-8ec6b17a49dc \
  -u tom_bridge@mac.com -p abba-cddc-effe-ghhg
```

```
"logFormatVersion": 1,
  "jobId": "d68742ec-f5cd-455b-bedf-8ec6b17a49dc",
  "status": "Accepted",
  "statusSummary": "Ready for distribution",
  "statusCode": 0,
  "archiveFilename": "munkitools-3.6.2.3778.pkg",
  "uploadDate": "2019-06-25T03:40:17Z",
  "sha256":
"93c7c7bca8d8d9a2a9b5355c8af500a928bbbb108ab04b5af
787ce0418c9321f"
```



munkitools-3.6.2.3778.pkg

Developer ID Installer Package

DEVELOPER ID Signed by “Developer ID Installer: Technolutionary LLC (X...

Notarized by Apple

Restart required after installation

Runs 3 install scripts

12 MB for package — 30.9 MB installed on disk

Previously installed on “Macintosh HD” using /usr/sbin/installer — June 22, 2019

```
▼ Applications
    ▶ Managed Software Center.app   Version 5.1.0.3774 (3774)
▼ Library
    ▼ LaunchAgents
        com.googlecode.munki.app_usage_monitor.plist
        com.googlecode.munki.ManagedSoftwareCenter.plist
        com.googlecode.munki.managedsoftwareupdate-loginwindow.plist
        com.googlecode.munki.munki-notifier.plist
        com.googlecode.munki.MunkiStatus.plist
    ▼ LaunchDaemons
        com.googlecode.munki.appusaged.plist
        com.googlecode.munki.authrestartd.plist
```

Open in Suspicious Package
DEVELOPER ID

munkitools-3.6.2.3778.pkg

Developer ID Installer Package

Signed by “Developer ID Installer: Technolutionary LLC (X...

Notarized by Apple

Restart required after installation

Runs 3 install scripts

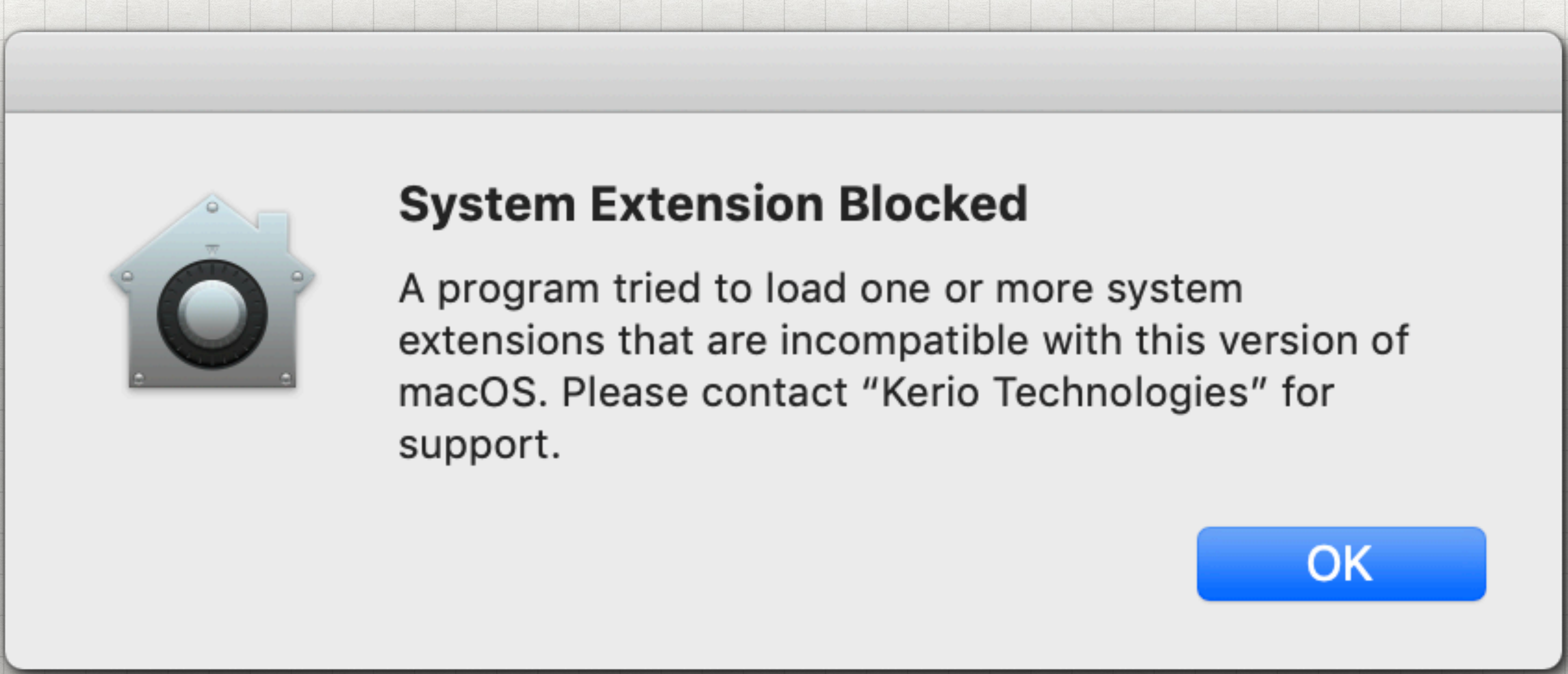
12 MB for package — 30.9 MB installed on disk

Previously installed on “Macintosh HD” using /usr/sbin/installer — June 22, 2019

```
▼ Applications
    ▶ Managed Software Center.app   Version 5.1.0.3774 (3774)
▼ Library
    ▼ LaunchAgents
```

BUT WHAT IF IT’S NOT NOTARIZED OR STAPLED?

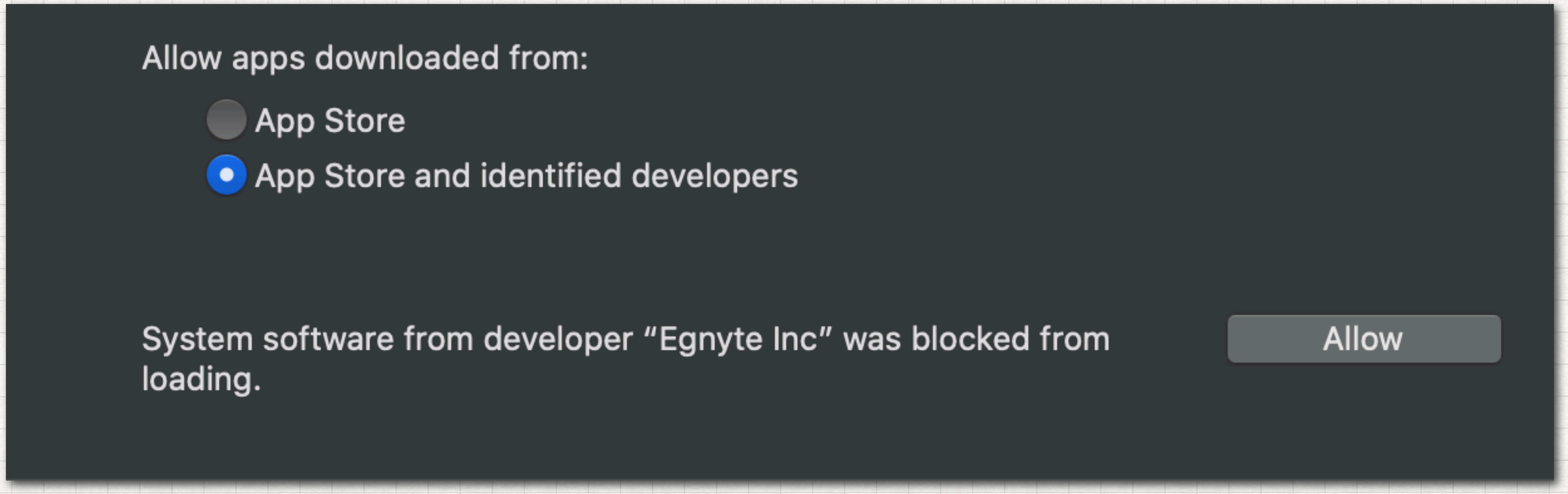
THAT’S THE BIG QUESTION, RIGHT?



System Extension Blocked

A program tried to load one or more system extensions that are incompatible with this version of macOS. Please contact “Kerio Technologies” for support.

USER ACCEPTED KERNEL EXTENSION LOADING

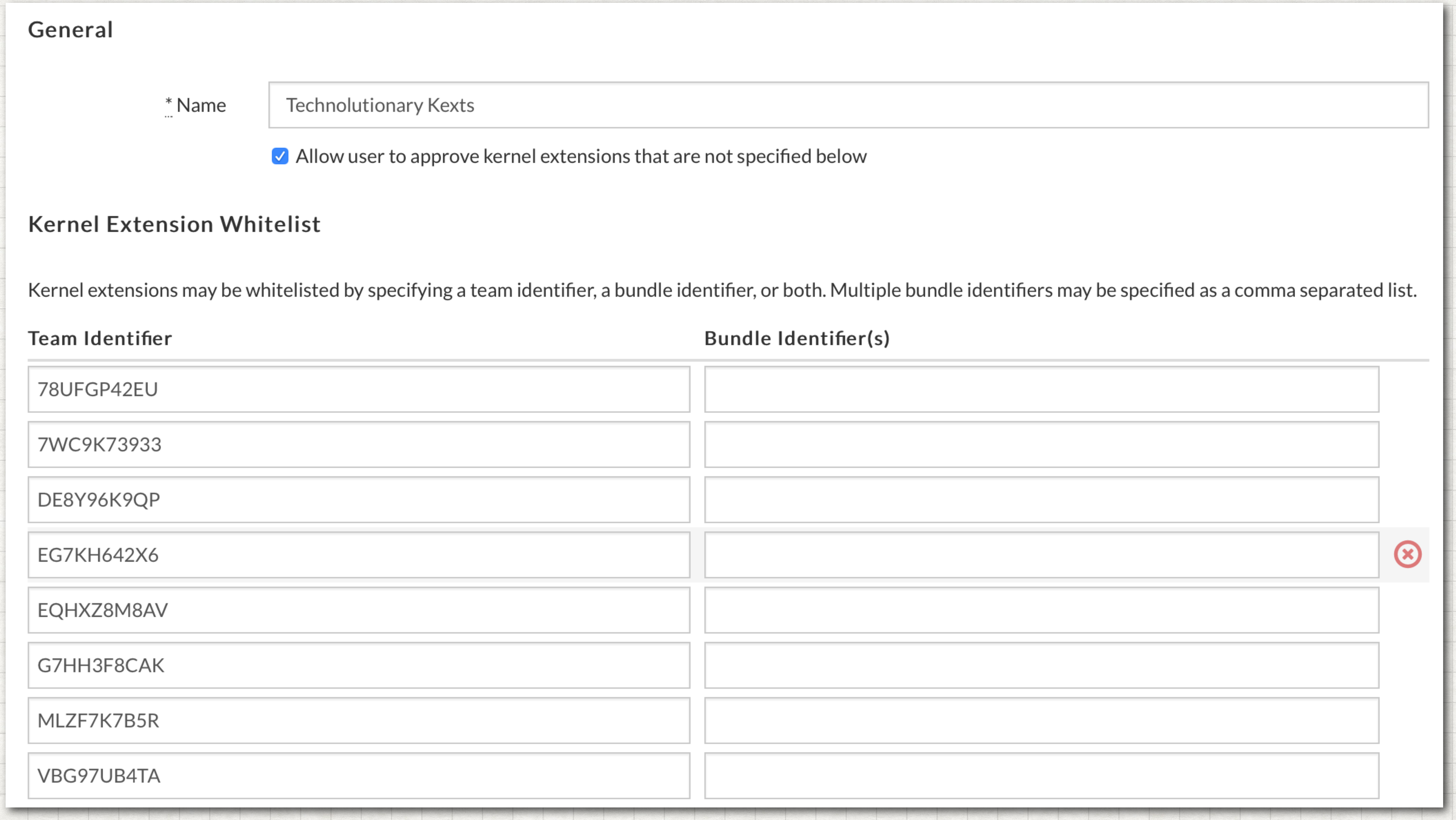


Allow apps downloaded from:

○ App Store

● App Store and identified developers

System software from developer “Egnyte Inc” was blocked from loading.



General

* Name: Technolutionary Kexts

☑ Allow user to approve kernel extensions that are not specified below

Kernel Extension Whitelist

Kernel extensions may be whitelisted by specifying a team identifier, a bundle identifier, or both. Multiple bundle identifiers may be specified as a comma separated list.

Team Identifier	Bundle Identifier(s)
78UFGP42EU	
7WC9K73933	
DE8Y96K9QP	
EG7KH642X6	
EQHXZ8M8AV	
G7HH3F8CAK	
MLZF7K7B5R	
VBG97UB4TA	

WHITELISTING A TEAM ID AFFECTS THE NOTARIZATION RESTRICTION

REMEMBER: ANYONE CAN STAPLE

AS LONG AS THE DEVELOPER HAS SUBMITTED FOR NOTARIZATION

TROUBLESHOOTING

WHAT DOES APPLE SAY?

https://help.apple.com/xcode/mac/current/#/dev033e997ca

Apple Help

WHAT DOES APPLE SAY?

https://developer.apple.com/documentation/security/notarizing_your_app_before_distribution/resolving_common_notarization_issues

Apple Docs

SEVEN COMMON NOTARIZATION ISSUES

- **Invalid Code Signature** - Use `codesign` to verify/validate
- **Use a Valid Certificate** - Check your certs
- **Include a Secure Timestamp** - Use `codesign` to verify/validate
- **Avoid Get-Task-Allow** - Turn off before Archiving
- **Use the 10.9 SDK or Later** - Drop 10.8 builds, people
- **Enable the Hardened Runtime** - Method Swizzling Is Bad
- **Stapler Problems** - Develop on 10.14, Use Xcode 10.2+

IN SUMMARY

WHY NOTARIZATION MATTERS

- Record of all signed binaries
- Record of components inside signed software.
- XProtect Checks on all launches can stop individual builds

WHAT YOU NEED TO NOTARIZE

- Signed Binary with Hardened Runtime
- Xcode Installed, with CLI Tools
- Apple Developer ID with Application-Specific Password

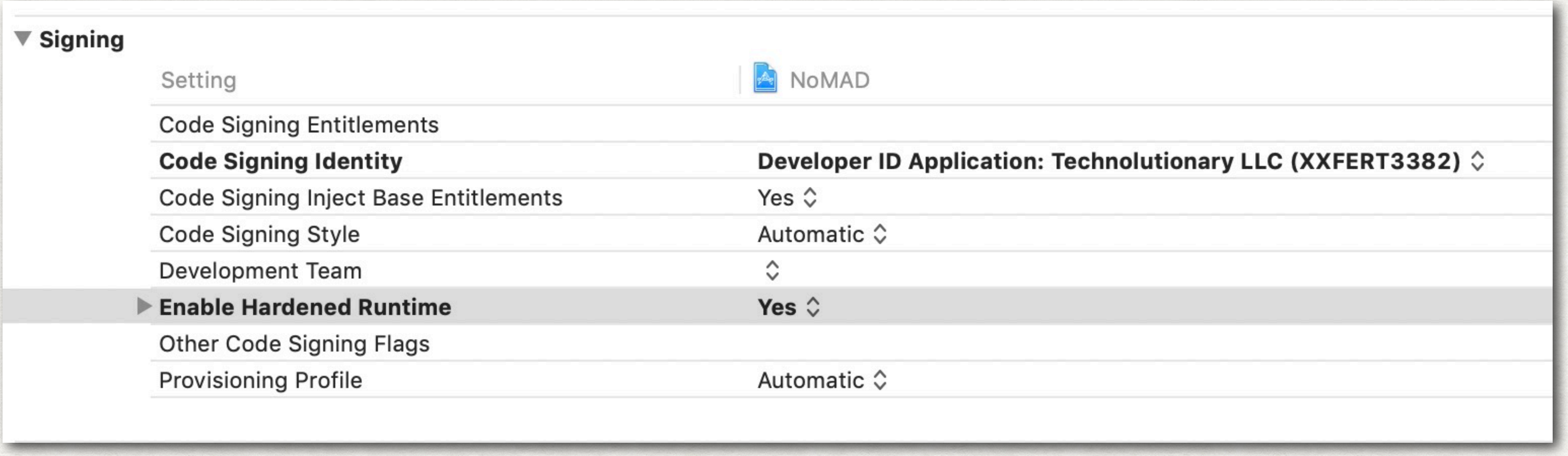
WHO NEEDS TO NOTARIZE

- Any software installed via methods covered by Gatekeeper
- Any kernel extension, period.*
- In 10.15, every application and tool invoked by LaunchServices.
- In 10.14.5, every app signed by a developer signed up after 7 April

* have you asked your software partner if they’re building a system extension yet?

HOW DO I NOTARIZE?

- Use Xcode Directly



▼ **Signing**

Setting	NoMAD
Code Signing Entitlements	
Code Signing Identity	**Developer ID Application: Technolutionary LLC (XXFERT3382)** ◊
Code Signing Inject Base Entitlements	Yes ◊
Code Signing Style	Automatic ◊
Development Team	◊
▶ **Enable Hardened Runtime**	**Yes** ◊
Other Code Signing Flags	
Provisioning Profile	Automatic ◊

- Or Do It By CLI
  - `xcrun altool --notarize-app`
  - `xcrun stapler staple`

QUESTIONS?

QUESTIONS?

Statements Aren’t Questions.

QUESTIONS?

Statements Aren’t Questions.

Speculation Is For The Bar.

DOWNLOADS & NOTES & STUFF

NOTES & STUFF

Notes & PDF

https://tombridge.com/?p=686