Apple Sandboxing

SMD, May 4, 2022

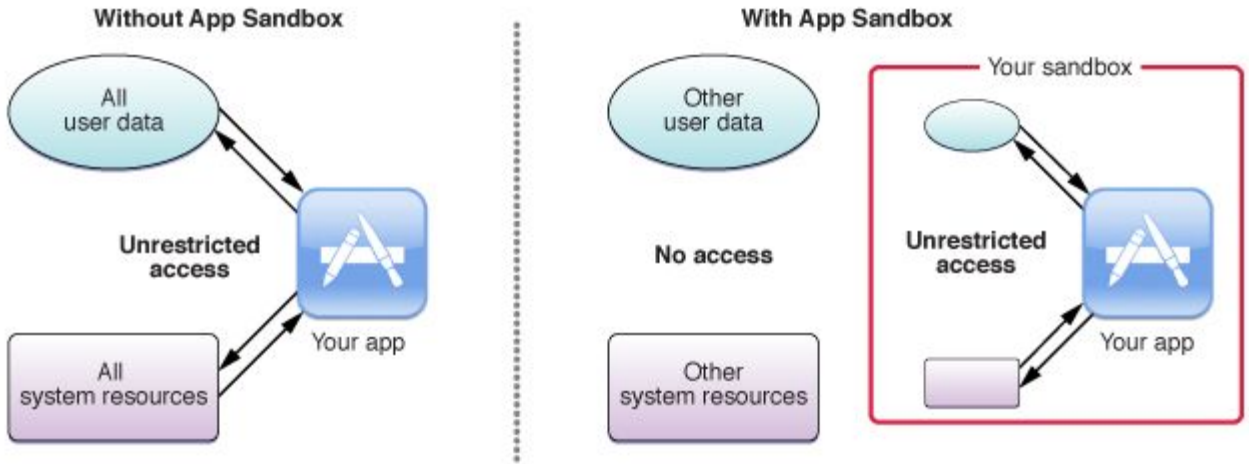
Agenda

Overview of sandboxing

Apple (iOS) sandboxing

Reversing the Apple sandbox

Assessing / Exploiting the Apple sandbox


Without App Sandbox
All
user data
Unrestricted
access
Your app
All
system resources
With App Sandbox
Other
user data
No access
Other
system resources
Your sandbox
Unrestricted
access
Your app

Sandboxing Questions

How to define sandbox rules?

How to store sandbox rules?

How to apply sandbox rules?

Linux Answers

Define sandbox rules as system calls + argument filters

Store sandbox rules as commands inside program to be sandboxed

Triggered by voluntary sandbox calls from sandboxed program, applied by kernel

Apple Sandboxing Answers

Defined as SBPL (Sandbox Profile Language) Rules (TinyScheme)

Stored in predefined sandbox profile files (.sb)

Applied voluntarily by program or enforced by kernel

SBPL Snippet

```
(allow system-info
    (require-all
        (info-type "net.link.addr")
        (require-entitlement "fairplay-client")
        (require-not (require-entitlement "com.apple.private.MobileGestalt.AllowedProtectedKeys"))))
(allow system-privilege)
(allow system-socket
    (socket-domain AF_ROUTE)
    (require-all
        (socket-domain AF_SYSTEM)
        (socket-protocol 2))
    (require-all
        (socket-domain 39)
        (require-entitlement "com.apple.private.signing-identifier"
              (entitlement-value "com.apple.Maps"))))
```

Mapping sandbox profiles to applications in iOS

As part of manifest file (xml, .plist)

When program is loaded, the manifest file is read and the corresponding profile is loaded

3rd party applications do not have a sandbox profile in the manifest file

- the container.sb sandbox profile is applied by default
- 3rd party applications are matched by their location in the filesystem

Storing & Applying iOS Sandbox Profiles

Part of the kernel cache extension (Sandbox.kext)

Binarized (compiled) and bundled together (since iOS 9)

When application is loaded (i.e. exec() call) - the corresponding binary sandbox profile is applied

Sandbox.kext stores the code to interpret and apply the sandbox rules

Customizing container.sb

container.sb applied to **all** 3rd party apps: too many applications

Making it customizable / specific:

- entitlements: hard-coded key-values in manifest files
- extensions: temporary capabilities provided during runtime (may be revoked)

Customizing container.sb (2)

```
(require-entitlement "com.apple.private.signing-identifier"
      (entitlement-value "com.apple.webbookmarksd"))


(require-all
     (subpath "/private/var/mobile/Media")
     (extension "com.apple.avasset.read-only")
     (extension-class "com.apple.mediaserverd.read")
     (extension "com.apple.tcc.kTCCServicePhotos"))
```

Our work (Malus Security)

Reversing iOS binary sandbox profiles (Sandblaster)

Assessing iOS sandbox profiles (Sandscout)

Sandblaster

https://github.com/malus-security/sandblaster

Python

Extracts and reverses (decompiles) binary sandbox profiles

Returns SBPL profiles

iExtractor, iExtractor-manager

https://github.com/malus-security/iExtractor

https://github.com/malus-security/iExtractor-manager

shell scripting, Python

downloads IPSW files and extracts all required files to be used by Sandblaster (and other tools)

Sandscout

https://github.com/malus-security/sandscout

Python, Prolog

processes reversed sandbox rules as Prolog files

applies queries (on security invariants) to detect flaws

Findings

https://www.google.com/search?channel=fs&client=ubuntu&q=cve+ios+deshotels+deaconescu

created PoC attacks based on Sandscout Prolog queries

submitted to Apple (responsible disclosure)

published as CVEs

Malus Security

https://github.com/malus-security/

https://discord.gg/m3gjuyHYw9