Joint work with Mihai Chiroiu (UPB), Luke Deshotels and Will Enck (NCSU), Lucas Davi, Ahmad-Reza Sadeghi (TU Darmstadt)
Apple iOS 3rd Party Apps

AppStore

Installed on user devices

Vetted by Apple

Protected by: sandboxing, Preferences settings
Jekyll Apps

Malicious apps disguised as benign apps

Abused features that should not be exposed

https://www.usenix.org/conference/usenixsecurity13/technical-sessions/presentation/wang_tielei
The Apple Sandbox

Figure 1–1 An iOS app operating within its own sandbox

- **Sandbox**
  - **Bundle Container**
    - MyApp.app
  - **Data Container**
    - Documents
    - Library
    - Temp
  - **iCloud Container**
    - ...

MyApp
Goal and Steps

Evaluate Apple iOS sandbox

Reverse Apple sandbox

Model the sandbox

Detect policy flaws

Let Apple know

Provide lessons learned
Challenges in Evaluating iOS Sandbox

- iOS is closed source
- Limited documentation
- Kernel is encrypted
- No root access
- Sandbox policies are hidden and compiled
- Sandbox policies are large and complex
- Potential flaws need to be tested
Sandbox Architecture

1. Privileged apps grant extensions
2. Sandbox allows or denies system calls
3. Entitlements satisfy conditions
4. Extensions satisfy conditions
5. Some system apps use container profile
6. Daemons have unique profiles
7. Some system apps have unique profiles
8. AppleMatch processes regular expressions
SandBox Profile Language (SBPL)

( allow file-read*
  ( require-all
    ( subpath "/Media/Safari" )
    ( require-not
      ( literal "/Media/Safari/secret.txt" )
    )
  )
  ( require-entitlement
    "private.signing-identifier"
    ( require-any
      ( entitlement-value "mobilesafari" )
      ( entitlement-value "safarifetcherd" )
    )
  )
)
Sandbox Profiles

Specific rules attached to certain apps

Originally written in SBPL, provided as binary blobs

May be shared by multiple apps
The container Sandbox Profile

Used by all 3rd party apps

Largest sandbox profile

Main target of our analysis due to scope and complexity
Sandbox Profiles
Previous Work on Reversing the Apple Sandbox

Dionysus Blazakis: https://github.com/dionthegod/XNUSandbox

Stefan Esser: https://github.com/sektioneins/sandbox_toolkit
Reversing Methodology

1. Extract Required Profiles
2. Extract Built-in Profiles
3. Header Analysis
4. Deserialize Regular Expressions
5. Build Custom Profiles
6. Reverse Operation Nodes
7. Clean Up Profiles
Current State


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

Works on iOS 7-11, minor issue from 11.1.2

To test on iOS 12
Evaluate iOS Apple Sandbox

Is a 3rd party app allowed more than necessary? Is this dangerous?

Automated process

Focus on container

SandScout
Overview
Prolog Facts

SBPL to Prolog Compiler

- Lex
- Yacc
- Context Free Grammar
- Disjunctive Normal Form

```
decision(operation,[listOfFilters]).
```

```
allow(file-readSTAR,
    [literal("/myFile"),extension("A")]).
allow(file-readSTAR,
    [literal("/myFile"),not(extension("B"))]).
```
Prolog Queries

1. To prevent damage to the system, full write access to system file paths, is reserved for apps with system capabilities.

?- allow(file-writeSTAR, Filters),
   member(X, Filters), member(X, SysPaths),
   intersection(Filters, SysCaps, [\]).

allow(file-writeSTAR, [subpath("/Library/AddressBook/")],
   extension("AddressBook")].
Findings

Privacy leaks

- Apple Maps privacy leak
- iTunes privacy leak
- Metadata leak
- Unauthorized Collusion

Storage consumption

Deny access to system files
Hard-link Attack

- Access to AddressBook fails without privacy setting
- Request privacy setting to get access
- User should be able to revoke access by turning off privacy setting
- Create hard link to AddressBook and put the link in /KeyboardCache/
- All apps can read and write in /KeyboardCache/
- Therefore all apps get access to AddressBook regardless of privacy setting
- Hard link changes path while keeping same file inode
Disclosure to Apple

Several calls with Apple Security team

CVE-2016-4686
CVE-2016-4664
CVE-2016-4665
CVE-2015-7001
Current Work

Expanding SandScout to Apple NSXPC

Part of sandbox rules (mach-lookup)

Services provided to 3rd party apps
Summary

First full reversing of the Apple Sandbox


Flaws detected and fixed in the Apple Sandbox

https://dl.acm.org/citation.cfm?id=2978336
Takeaway

Reversing is both hard and rewarding

Complexity is the enemy of security

You need automated verification/validation

Pay attention to access control rules in system security implementation