App Sandbox and the Mac App Store

Session 204

Ivan Krstić
Core OS Security Samurai

App Sandbox

- New damage containment mechanism in Lion
- Last line of defense against exploitation and coding errors
- Not an anti-virus system; does not target intentionally malicious software
- Drives policy by user intent

Mac App Store Submissions

- Must adopt App Sandbox
- Specify entitlements for every executable
- Describe how each entitlement is used

Choosing Entitlements

- Absolute minimum set you need to support your features
- Entitlements must match app features, especially for personal information
- Overly broad entitlements are cause for rejection from the Store
- Exercise your application, check logs for sandbox violations, examine stack traces

All the Entitlements

Meet Your Entitlements

- All App Sandbox entitlements in the com.apple.security namespace
- Big on/off switch: com.apple.security.app-sandbox
  - App runs in a container
  - No network access, no access to user documents, can not use Open or Save panels
  - No printing, no camera, no microphone
  - No access to address book, calendars, user location

User-selected Files

- com.apple.security.files.user-selected
  - .read-only
  - .read-write
- Controls access to Powerbox
- Document-based apps, import/export

Network

- com.apple.security.network
  - .client
  - .server
- Making and receiving network connections
- Presently does not offer specific domain or port restrictions
- Games with leaderboards, Twitter clients

Personal Information

- com.apple.security.personal-information
  - .addressbook
  - .calendars
  - .location
- Mail and calendar clients

Media Assets

- com.apple.security.assets
  - .music
  - .pictures
  - .movies
- All available as read-only or read-write

Devices

- com.apple.security.devices
  - .camera
  - .microphone
  - .usb
- Photo and podcasting apps

Other

- com.apple.security.print
- com.apple.security.inherit
  - For bundled helper executables via fork/exec
  - Inherits parent sandbox

Temporary Exceptions

- Transitional; to ease initial adoption
- com.apple.security.temporary-exception
  - .apple-events
  - .mach-lookup
  - .home-relative-path, .absolute-path

Developing for App Sandbox

Handle Errors Gracefully

- APIs and facilities that “always worked” before may fail
- Proper error handling is key to providing a good user experience
- Degrade as gracefully as possible
- When developing, check Console frequently for Sandbox violation reports

Filesystem

- Each app gets its own “home directory” called a container
- Stores app data that’s not user-created or visible
  - Preferences
  - Caches
  - Databases
  - Documents
- `$HOME` now points to the container, no access to real home directory

Filesystem

- Use `NSOpenPanel` and `NSSavePanel` for user documents
- Use Apple API like `NSTemporaryDirectory()` and `NSFileManager`
- NSDocument-based apps get safe save for free
- If you need to implement your own, use `NSFileManager`’s `URLForDirectory:inDomain:appropriateForURL:create:error:`

Launching Processes

- `fork()/exec()` and `posix_spawn()` work, but all sandbox restrictions are inherited
- Anything you launch needs to be able to deal with the sandbox
- Use LaunchServices API such as `LSOpen()` to open documents in the correct application regardless of App Sandbox

Apple Events

- App Sandbox allows receiving, but not sending Apple Events
- API replacements exist for many common patterns
- Opening a Finder window: `NSWorkspace openFile:`
- Controlling iTunes
  - Must request a temporary exception entitlement

Spotlight

- From your app, Spotlight searches only the container
- Primarily useful to apps that keep their own searchable data stores, e.g., mail clients
- Documents from your app still indexed and available to the user from anywhere

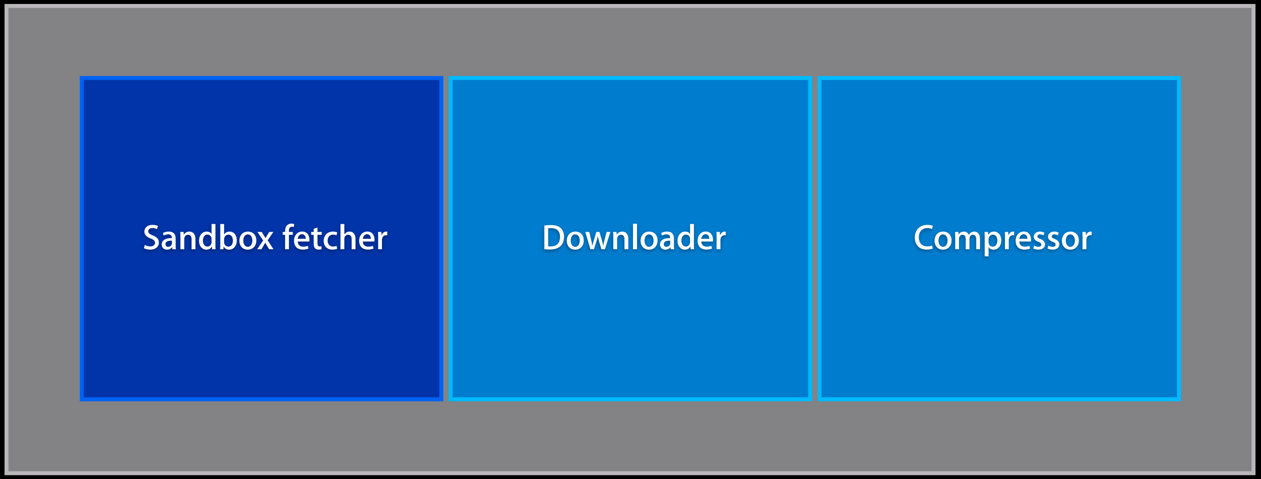
Privilege Separation

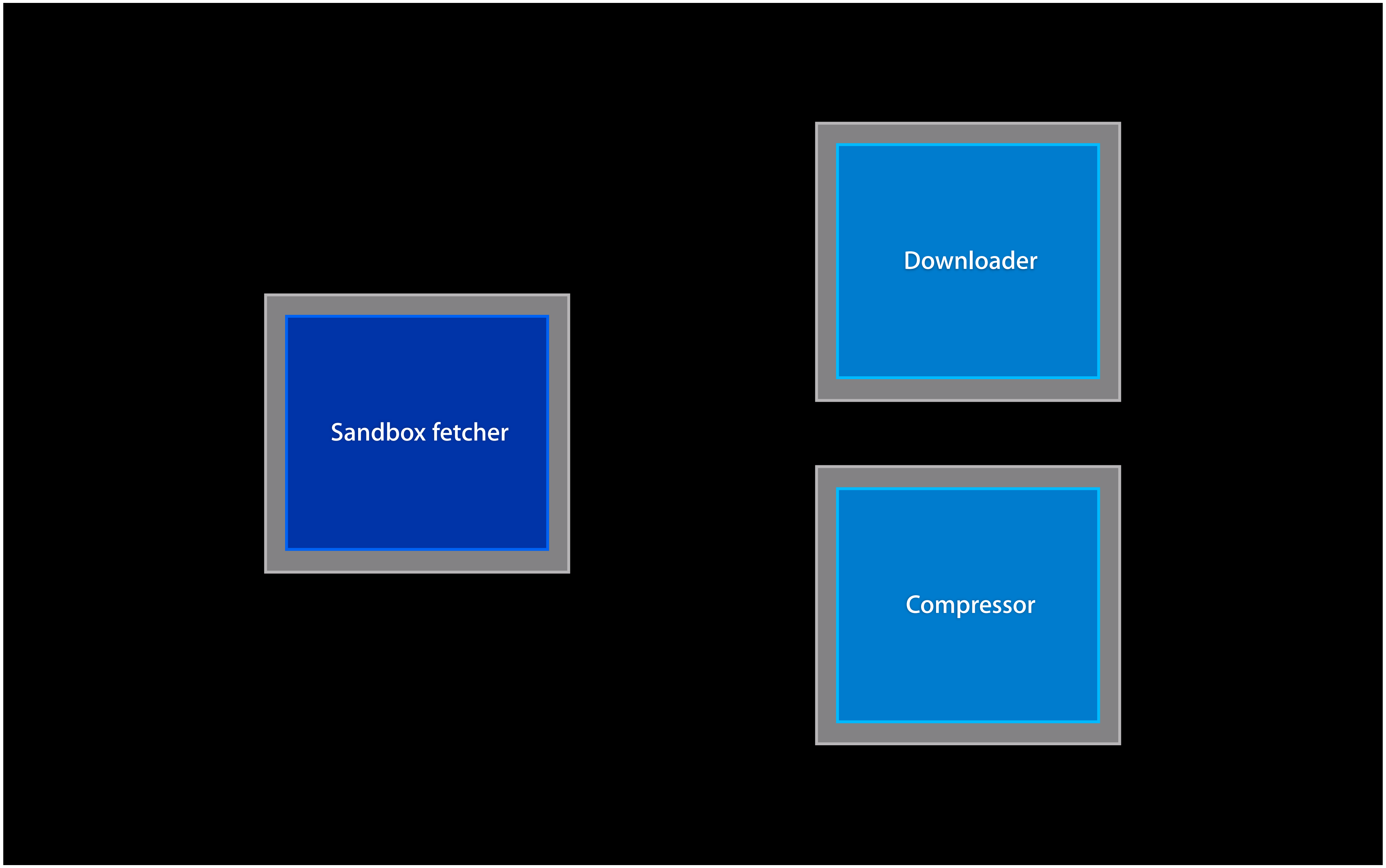
Tymshare

- Big time share systems company in 60s and 70s
- Submit jobs to a compiler, provide filename for debug output
- Compiler can write out statistics to (SYSX)STAT and billing to (SYSX)BILL
- User provides (SYSX)BILL as debug filename, compiler overwrites billing file

Privilege Separation

- Aim to separate components facing the network from those that are not
- Isolate components with access to personal information
- Separation is only effective when components do not blindly obey commands


Sandbox fetcher
Downloader
Compressor


Sandbox fetcher
Downloader
Compressor

Sample Code Walkthrough

- Web page downloader
  - UI component, network service, compression service
  - Blueprint for privilege separation, XPC Services, and App Sandbox use

XPC Services

- Services ship in your app bundle
- Each service gets its own entitlements and container
- Information shared between application and its services must be explicit over IPC

Odds and Ends

Container Inspection and Migration

```
$ asctl container path Preview ⏎
~/Library/Containers/com.apple.Preview

$ man asctl
```

- Container migration manifest
  - Runs on first launch, can copy or move data from outside the container
  - Shipping apps can easily move their data into the container

Learn More

App Sandbox Lab	Core OS Lab Wednesday 9:00AM
Introducing XPC	Russian Hill Wednesday 11:30AM

- Code Signing and Application Sandboxing Guide
- Sample code: Sandbox Fetcher

Summary

- App Sandbox—Last line of defense against exploitation and coding errors
- Protection only as good as the entitlements, choose wisely
- Use Apple API when interacting with the system
- App Sandbox required for Mac App Store submissions

WWDC2011