Tweaks on a multi-process iOS

Ryan Petrich, for JailbreakCon 2016

What is a process?

Processes

- Instance of a program running on a computer
- Separate virtual “working space” for each so that programs can’t disrupt on another when they fail
- Kernel creates the illusion of multiple processes running simultaneously by switching between those that have work to do many times per second
  - Modern multi-core devices actually do run more than one (and we have these in our pockets now 📱)
- Security boundary that modern operating systems use to control resource access
  - No direct access to hardware on modern machines, even if processes “cheat”

How many processes are running on a fresh boot?

iPhone OS 2.0 had a dozen

`SpringBoard`

Graphics/window manager

`CommCenter`

Radio hardware manager

`MobileMail`

Mail client

`MobilePhone`

Phone client

…a handful of others

Most interesting behaviour and data was inside the `SpringBoard` process

iOS 9.0 has…

```
$ ps ax
```

```
$ ps ax
  PID   TT  STAT      TIME COMMAND
    1   ??  Ss     2:46.66 /sbin/launchd
   39   ??  Ss     2:31.94 /usr/sbin/syslogd -bsd_out 1
   44   ??  Ss     2:24.33 /usr/sbin/mediaserverd
   46   ??  Ss     0:30.99 /usr/libexec/fseventsd
   48   ??  Ss     0:16.02 /System/Library/PrivateFrameworks/AssistantServices.framework/assistantd
   52   ??  Ss     0:55.74 /usr/libexec/routined
   56   ??  Ss     0:05.79 /System/Library/PrivateFrameworks/MediaRemote.framework/Support/mediaremoted
   58   ??  Ss     0:00.30 /usr/libexec/misd
   64   ??  Ss     1:38.27 /System/Library/CoreServices/powerd.bundle/powerd
   72   ??  Ss     3:45.12 /usr/sbin/wifid
   90   ??  Ss     3:10.02 /System/Library/PrivateFrameworks/IDS.framework/identityservicesd.app/
identityservicesd
  102   ??  Ss     0:01.10 /System/Library/CoreServices/SpringBoard.app/SpringBoard
  104   ??  Ss     0:35.03 /usr/sbin/wirelessproxd
  106   ??  Ss     0:36.97 /usr/libexec/backboardd
  110   ??  Ss     0:58.07 /usr/libexec/timed
  112   ??  Ss     0:37.35 /usr/libexec/sharingd
  114   ??  Ss     0:20.56 /usr/libexec/locationd
  116   ??  Ss     1:11.13 /usr/sbin/BTServer
  118   ??  Ss     0:17.48 /System/Library/PrivateFrameworks/IMCore.framework/imagent.app/imagent
  120   ??  Ss     1:16.51 /usr/libexec/assertiond
  128   ??  Ss     6:37.88 /usr/libexec/UserEventAgent (System)
  134   ??  Ss     0:43.94 /usr/sbin/mDNSResponder
  136   ??  Ss     0:08.02 /System/Library/PrivateFrameworks/IAP.framework/Support/iaptransportd
  148   ??  Ss     8:06.13 /System/Library/Frameworks/CoreTelephony.framework/Support/CommCenter
  150   ??  Ss     0:00.23 /System/Library/PrivateFrameworks/NanoLeash.framework/companionproximityd
  152   ??  Us     0:00.56 /usr/libexec/tursd
  156   ??  Ss     0:00.61 /System/Library/PrivateFrameworks/NanoLeash.framework/companionfindlocallyd
  158   ??  Ss     0:36.93 /usr/libexec/bulletindistributord
  186   ??  Ss     2:04.56 /usr/sbin/notifyd
  197   ??  Ss     2:02.01 /usr/sbin/cfprefsd daemon
  201   ??  Us     0:10.07 /usr/sbin/distnoted daemon
  203   ??  Ss     0:11.24 /usr/libexec/lsd
  207   ??  Ss     0:12.88 /usr/libexec/diagnosticd
  210   ??  Ss     2:11.32 /usr/libexec/networkd
  220   ??  Ss     0:04.52 /usr/libexec/MobileGestaltHelper
  229   ??  Ss     0:07.27 /usr/libexec/nanoregistryd
  236   ??  Ss     2:19.62 /usr/libexec/securityd
  239   ??  Ss     1:57.45 /usr/libexec/coreduetd
  254   ??  Ss     0:44.72 /usr/libexec/hangtracerd
  260   ??  Ss     0:52.55 /System/Library/PrivateFrameworks/ApplePushService.framework/apsd
  263   ??  Ss     0:08.24 /System/Library/PrivateFrameworks/TCC.framework/tccd
```

```
  263   ??  Ss     0:08.24 /System/Library/PrivateFrameworks/TCC.framework/tccd
  280   ??  Ss     0:21.46 /usr/libexec/nsurlstoraged
  299   ??  Ss     0:03.87 /usr/sbin/filecoordinationd
  307   ??  Ss     0:06.97 /usr/libexec/nehelper
  327   ??  Ss     1:22.29 /usr/sbin/WirelessRadioManagerd
  336   ??  Ss     2:16.97 /usr/libexec/nsurlsessiond
  340   ??  Ss     0:08.13 /usr/libexec/biometrickitd --launchd
  344   ??  Ss     2:14.55 /usr/libexec/symptomsd
  353   ??  Ss     0:42.71 /usr/libexec/nesessionmanager
  380   ??  Ss     3:31.32 /System/Library/PrivateFrameworks/DataAccess.framework/Support/dataaccessd
  407   ??  Ss     0:02.46 /System/Library/PrivateFrameworks/HomeSharing.framework/Support/itunescloudd
  417   ??  Ss     0:56.66 /System/Library/Frameworks/PassKit.framework/passd
  429   ??  Ss     0:01.11 /usr/libexec/nfcd
  445   ??  Ss     0:26.71 /usr/libexec/duetexpertd
  460   ??  Ss     0:05.55 /System/Library/PrivateFrameworks/UserActivity.framework/Agents/useractivityd
  465   ??  Ss     0:33.15 /System/Library/PrivateFrameworks/GeoServices.framework/geod
  516   ??  Ss     0:54.82 /Applications/MobileMail.app/MobileMail
  672   ??  Ss     0:11.80 /System/Library/PrivateFrameworks/NanoPassKit.framework/NPKCompanionAgent
  776   ??  Ss     1:37.01 /Applications/SafariViewService.app/SafariViewService
  904   ??  Ss     0:00.20 /usr/libexec/swcd
 2822   ??  Ss     0:02.30 /usr/libexec/nanomediaremotelinkagent
 3447   ??  Ss     0:03.23 /usr/libexec/wcd
 3521   ??  Ss     0:40.38 /System/Library/PrivateFrameworks/Search.framework/searchd
 3708   ??  Ss     0:01.84 /System/Library/PrivateFrameworks/AuthKit.framework/akd
 4439   ??  Ss     0:17.77 /Applications/InCallService.app/InCallService
 4793   ??  Ss     0:17.24 /System/Library/PrivateFrameworks/CalendarDaemon.framework/Support/calaccessd
 4823   ??  Ss     0:06.35 /System/Library/PrivateFrameworks/TelephonyUtilities.framework/callservicesd
 4942   ??  Ss     0:08.87 /System/Library/PrivateFrameworks/CloudDocsDaemon.framework/bird
 5059   ??  Ss     0:39.30 /usr/libexec/configd
 6007   ??  Ss     0:15.55 /System/Library/PrivateFrameworks/iTunesStore.framework/Support/itunesstored
 6042   ??  Ss     0:07.60 /System/Library/Frameworks/SystemConfiguration.framework/SCHelper
 6613   ??  Ss     0:11.93 /System/Library/Frameworks/Accounts.framework/accountsd
 6913   ??  Ss     0:03.24 /usr/libexec/DuetHeuristic-BM
 7687   ??  Ss     0:00.68 /usr/sbin/BTLEServer
 8184   ??  Ss     0:03.36 /System/Library/PrivateFrameworks/IAP.framework/Support/iapd
 8187   ??  Ss     0:16.46 /Applications/Music.app/Music
 8248   ??  Us     0:00.28 /System/Library/PrivateFrameworks/MapsSupport.framework/mapspushd
 8549   ??  Ss     4:18.22 /System/Library/PrivateFrameworks/AggregateDictionary.framework/Support/aggregated
 8693   ??  Ss     0:12.78 /System/Library/PrivateFrameworks/CloudKitDaemon.framework/Support/cloudd
 8737   ??  Ss     0:01.34 /usr/libexec/replayd
 8740   ??  Ss     0:00.96 /System/Library/PrivateFrameworks/CloudPhotoLibrary.framework/Support/cloudphotod
11897   ??  Ss     0:01.09 /System/Library/PrivateFrameworks/VisualVoicemail.framework/vmd
12616   ??  Ss     0:00.37 /usr/libexec/adid
14989   ??  Ss     0:02.49 /System/Library/Frameworks/HealthKit.framework/healthd
```

```
14989   ??  Ss     0:02.49 /System/Library/Frameworks/HealthKit.framework/healthd
15160   ??  Ss     0:02.74 /usr/sbin/fairplayd.H2
15201   ??  Ss     0:00.74 /Applications/PassbookUIService.app/PassbookUIService
15689   ??  Ss     0:00.09 /System/Library/Frameworks/WebKit.framework/XPCServices/
com.apple.WebKit.Databases.xpc/com.apple.WebKit.Databases
15725   ??  Ss     0:01.19 /System/Library/TextInput/kbd
15774   ??  Ss     0:01.88 /System/Library/Frameworks/WatchKit.framework/Support/companionappd
15786   ??  Ss     0:01.42 /usr/libexec/atc
16010   ??  Ss     0:01.86 /System/Library/PrivateFrameworks/IMDPersistence.framework/XPCServices/
IMDPersistenceAgent.xpc/IMDPersistenceAgent
16016   ??  Ss     0:00.42 /usr/libexec/fmfd
16028   ??  Ss     0:00.24 /System/Library/PrivateFrameworks/CoreRecents.framework/recentsd
16031   ??  Ss     0:00.24 /System/Library/PrivateFrameworks/CoreSuggestions.framework/suggestd
16290   ??  Ss     0:00.27 /usr/libexec/gamecontrollerd
16293   ??  SNs    0:00.04 /System/Library/PrivateFrameworks/AppSupport.framework/Support/cplogd
16295   ??  Ss     0:00.21 /System/Library/Frameworks/LocalAuthentication.framework/Support/coreauthd
16307   ??  Ss     0:00.03 /System/Library/PrivateFrameworks/CoreSymbolication.framework/coresymbolicationd
16633   ??  Ss     0:00.63 /usr/libexec/addressbooksyncd
16636   ??  Ss     0:00.16 /System/Library/PrivateFrameworks/AssistantServices.framework/XPCServices/
com.apple.siri.embeddedspeech.xpc/com.apple.siri.embeddedspeech
16639   ??  Ss     0:00.42 /usr/libexec/mobileassetd
16645   ??  Ss     0:00.81 /System/Library/PrivateFrameworks/AssistantServices.framework/assistant_service
16648   ??  Ss     0:00.43 /System/Library/PrivateFrameworks/NanoAppRegistry.framework/Support/
nanoappregistryd
16651   ??  Ss     0:00.37 /System/Library/PrivateFrameworks/MusicLibrary.framework/Support/medialibraryd
16654   ??  Ss     0:00.24 /usr/libexec/eventkitsyncd
16657   ??  Ss     0:00.14 /usr/libexec/networkd_privileged
16666   ??  Ss     0:00.13 /System/Library/PrivateFrameworks/PairedUnlock.framework/pairedunlockd
16668   ??  Ss     0:00.09 /System/Library/PrivateFrameworks/NanoPreferencesSync.framework/nanoprefsyncd -
companion
16670   ??  Ss     0:00.24 /usr/libexec/pipelined
16675   ??  Ss     0:00.13 /System/Library/PrivateFrameworks/ManagedConfiguration.framework/Support/profiled
16678   ??  Ss     0:00.02 /usr/libexec/misagent
16681   ??  Ss     0:00.15 /usr/libexec/pkd -d/var/db/PlugInKit-Annotations
16684   ??  Ss     0:00.17 /usr/libexec/lockdownd
16687   ??  Ss     0:00.04 /usr/libexec/lockbot
16690   ??  Ss     0:00.05 /System/Library/PrivateFrameworks/MobileActivation.framework/Support/mobactivationd
16693   ??  Ss     0:01.90 /usr/libexec/ptpd -t usb
16696   ??  Ss     0:00.44 sshd: root@ttys000
16722   ??  Ss     0:00.53 /System/Library/PrivateFrameworks/SyncedDefaults.framework/Support/syncdefaultsd
16886   ??  Ss     0:01.31 /Applications/MobileSMS.app/MobileSMS
16888   ??  Ss     0:00.83 /Applications/MessagesNotificationViewService.app/MessagesNotificationViewService
16890   ??  Ss     0:00.17 /System/Library/PrivateFrameworks/IDSFoundation.framework/
IDSRemoteURLConnectionAgent.app/IDSRemoteURLConnectionAgent
```

```
16890   ??  Ss     0:00.17 /System/Library/PrivateFrameworks/IDSFoundation.framework/IDSRemoteURLConnectionAgent.app/
IDSRemoteURLConnectionAgent
16893   ??  Us     0:00.16 /System/Library/PrivateFrameworks/IMFoundation.framework/XPCServices/
IMRemoteURLConnectionAgent.xpc/IMRemoteURLConnectionAgent
16896   ??  Ss     0:00.13 /System/Library/PrivateFrameworks/WirelessDiagnostics.framework/Support/awdd
16698 s000  Ss     0:00.11 -sh
16901 s000  R+     0:00.01 ps ax
$ ps ax | wc -l
127
$
```

127 processes!

`SpringBoard` still highly used, but most OS features broken out into separate service or “daemon” processes

Why? 🤔

Resource limits can now be placed on specific components

Allows mobile devices to be doing more work in parallel and make better use of CPU, network and disk I/O

Software bugs are now isolated to small subsystems instead of bringing down the whole system

Lets development teams better coordinate improvements to operating system components

Most of iOS is now implemented in system daemons and other helper processes

If we’re only building tweaks that are trapped in single processes, we’re limiting what we can do

Inter-Process Communication 

Inter-Process Communication 📡

- Mechanisms provided by the kernel to facilitate coordinated sharing of data and commands between processes
- Used heavily in recent versions of iOS and OS X to implement system frameworks and APIs

Standard Techniques

Save to temp files 💾

- High level APIs, easy to use 👍
- Power intensive 🔋
- Accessible file paths change between iOS versions ⚠️
- Only privileged processes can “write”
- Observing changes can be complicated

Unix Domain Sockets 🔌

- Low level API
- Standardized in POSIX, useful on other operating systems 🐧
- Stream oriented, requires basic parsing to reconstruct messages 💦
- Follows file system security model
  - …put your domain sockets in temp paths, as most everything else is locked down on iOS 🔒

Other standard techniques

- Shared Memory
- Signals
- Named pipes
- Network sockets

…mostly not that applicable for tweaks

Apple/iOS-specific Techniques

Darwin Notifications 📬

- High level API, easy to use 👍
- No data, only a simple “go” message 🚦
- Any process can post or observe
  - …but no ability to restrict this either
- Always asynchronous

🔗

Mach Ports

- Low level API
  - …seriously low level
- Message-oriented ✉️
- Zero copy 🚄
- Asynchronous, if you go through the trouble
- Service lookup is locked down on iOS 7+

CPDistributedNotificationCenter

- High level API, easy to use 👍
  - One of Apple's many wrappers for Mach messages
- Message-oriented ✉️
- Private API, does change between iOS versions ⚠️
- Asynchronous, if you go through the trouble
- Service lookup is restricted on iOS 7+ 🔒

CFMessagePort

- High level API, easy to use 👍
  - One of Apple's many wrappers for Mach messages
- Message-oriented ✉️
- Public API
- Only supports synchronous use
- Service lookup is restricted on iOS 6+ 🔒

XPC

- High level API, easy to use 👍
  - One of Apple's many wrappers for Mach messages
- Message-oriented ✉️
- Public API on OS X only ⚠️
- Asynchronous always, no synchronous versions
- Service lookup is restricted on iOS 7+ 🔒

Creative Techniques

Relax existing service permissions

- Need location data, but the process you're in doesn't have permission? "Adjust" the permission check 🔓
  - Be careful not to make world-accessible accidentally
- Permissions checks can *and do* change between iOS versions

Repurpose existing iOS services' IPC channels

Repurpose existing iOS services' IPC channels

Don't do this. Please.
Service internals are frequently rewritten in new iOS versions

Delegate to someone else

No seriously.
This is a great strategy

Delegate to someone else

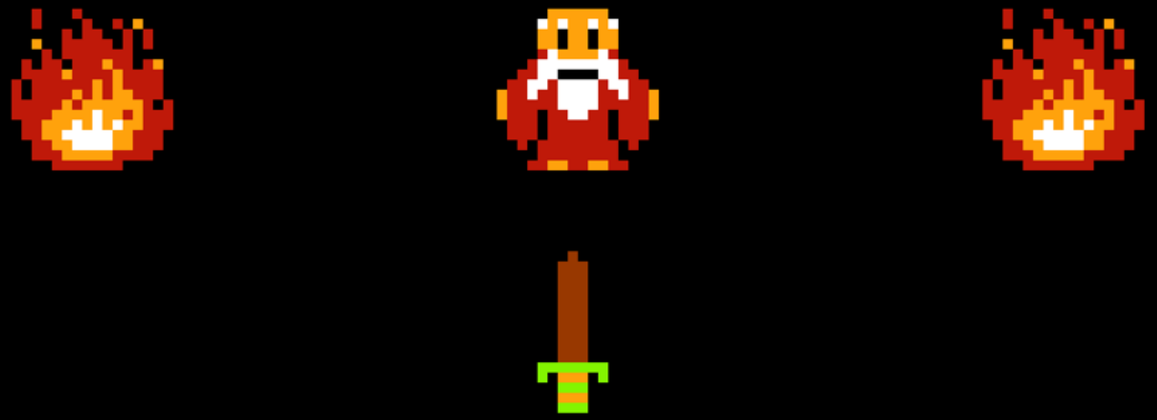
- Great for common functionality
- Shared maintenance burden
  - (in theory 🤷)
- Even if that someone else is mostly you, if it's shared between projects it's only one package to update

Libraries that use IPC under the hood 🚘

- AppList 🔗
  - “What apps are installed? What are their icons?”
- Flipswitch 🔗
  - “What is the value of this system setting? Please change it for me.”
- Activator 🔗
  - “SpringBoard, can you perform this simple action for me?”
- libcanopenurl (by rplus) 🔗
  - “Is this URL valid?”

Community Libraries to Help

IT'S DANGEROUS TO GO ALONE! TAKE THIS.



RocketBootstrap 🚀👢

Service registration and lookup system for iOS

RocketBootstrap 🚀👢

- Unlocks creation of non-Apple bootstrap services
- Uses iOS7's security model: Privileged processes can register, any process can look up
- Works with existing mach-based IPC mechanisms
- Requires package dependency, commonly installed on users' devices

RocketBootstrap 🚀👢

- Similar to Apple's bootstrap APIs:
  - `bootstrap_look_up` becomes `rocketbootstrap_look_up`
  - `bootstrap_register` becomes `rocketbootstrap_register`
- Easy to use wrappers for `CFMessagePort` and `CPDistributedMessagingCenter` (todo: XPC)
- Bring your own security model by using `audit_token_to_au32` to know who's calling ☎️

OBJCIPC

High-level API for hosting services inside apps
(by Alan Yip/a1anyip)

OBJCIPC

- High level API provides service lookup and IPC, easy to use 👍
- Background-launches and fakes app lifecycle for you 😊
- Message-oriented ✉️
- Mostly asynchronous
- “Open” security model
- Requires separate package dependency, but very small

🔗

OBJCIPC

- Simple Objective-C APIs:
  - Register using `registerIncomingMessageFromAppHandlerForMessageName:handler:`
  - Send using `sendMessageToAppWithIdentifier:messageName:dictionary:replyHandler:` 📥
  - Similar patterns for App to SpringBoard

LightMessaging 

Header-only library for simple IPC

LightMessaging 💡

- Mid-level API, not terribly difficult to use 👋
- Message-oriented ✉️
- Zero copy, for certain message types 🚄
- No additional cost over standard mach calls
- Easy integration with RocketBootstrap
- No package dependencies

🔗

LightMessaging 💡

- Start services with `LMStartService`
- Send messages with `LMConnectionSendTwoWay` (and friends) 📩
- Send replies with `LMSendReply` (and friends) 📤
- Bring your own security checks still 😐
  - Community developers, your input on API please!

Quick Tips for Developers

Be careful exposing user data or destructive functionality💥

I’m guilty of this—old versions of Activator were really trusting and allowed *any app* to send the “power down” or “reboot” actions 😂

Some tweaks allow *any app* to launch their functionality or change their settings from any app, without user prompt

Some tweaks force allow camera or location access so iOS’ privacy settings are *ineffective*

Some tweaks expose your email and text messages to *any* app that cares to ask 😱

Some tweaks even allow full access to the filesystem

Be aware of potential deadlocks 💀

iOS already has a “hierarchy” of which processes call which other processes using blocking IPC

Commonly, `SpringBoard` will block on `backboardd`—don’t call from `backboardd` to `SpringBoard`!

App and background processes can be “frozen” via `SIGSTOP` at any time and will fail to respond ❄️

Communicate with these processes using one-way IPC, asynchronous IPC, or two-way IPC with timeouts

Avoid the pitfall of accidentally sending blocking API calls to one’s own process

Distributed state management is a complicated dance 

Choosing a coordinator that “owns” all of the data for your tweak can make this much easier

Inform observers of changes or events via notifications

Devices are fast enough now that caching data may be more trouble than it’s worth

SpringBoard is usually a good choice for coordinator as it often has *much* work to do anyway

Avoid creating a chatty IPC protocol 

IPC has a fixed cost per message, and is more expensive “per byte” than procedure calls within a process

Batch all of the operations for a single user action into one IPC call, if possible

Filter to only the data required before sending

Use zero-copy for large data like images or delegated file access

@rpetrich on twitter

http://rpetri.ch/jbcon2016

For documentation and more information, click the link symbols → 🔗