

# Bad For Enterprise

Attacking BYOD Enterprise Mobile Security Solutions

Vincent Tan

Senior Security Consultant



- Sunny Singapore
- Senior Security Consultant @ Vantage Point Security
- 4+ years hacking stuff professionally, specializing in mobile & exotic stuff

1. iOS Applications in General
2. What is BYOD? Why BYOD? Who uses BYOD?
3. Security Features of BYOD Solutions
4. Good Technology
5. iOS Jailbreaks / Attack Vectors
6. Local & Network Attacks against Good EMS
  - Story of Alice & Bob

- > 1.4m Applications<sup>1</sup> in iOS App Store
  - ~10% in Business Category
- 35% of Enterprises have an Enterprise App Store<sup>2</sup>
- Simple vs Complex Functionality
  - Mobile application capabilities have not caught up with device capabilities
  - Maybe 10% of apps have advanced functionality
    - MDM, Soft Tokens, Payment Applications, HomeKit.



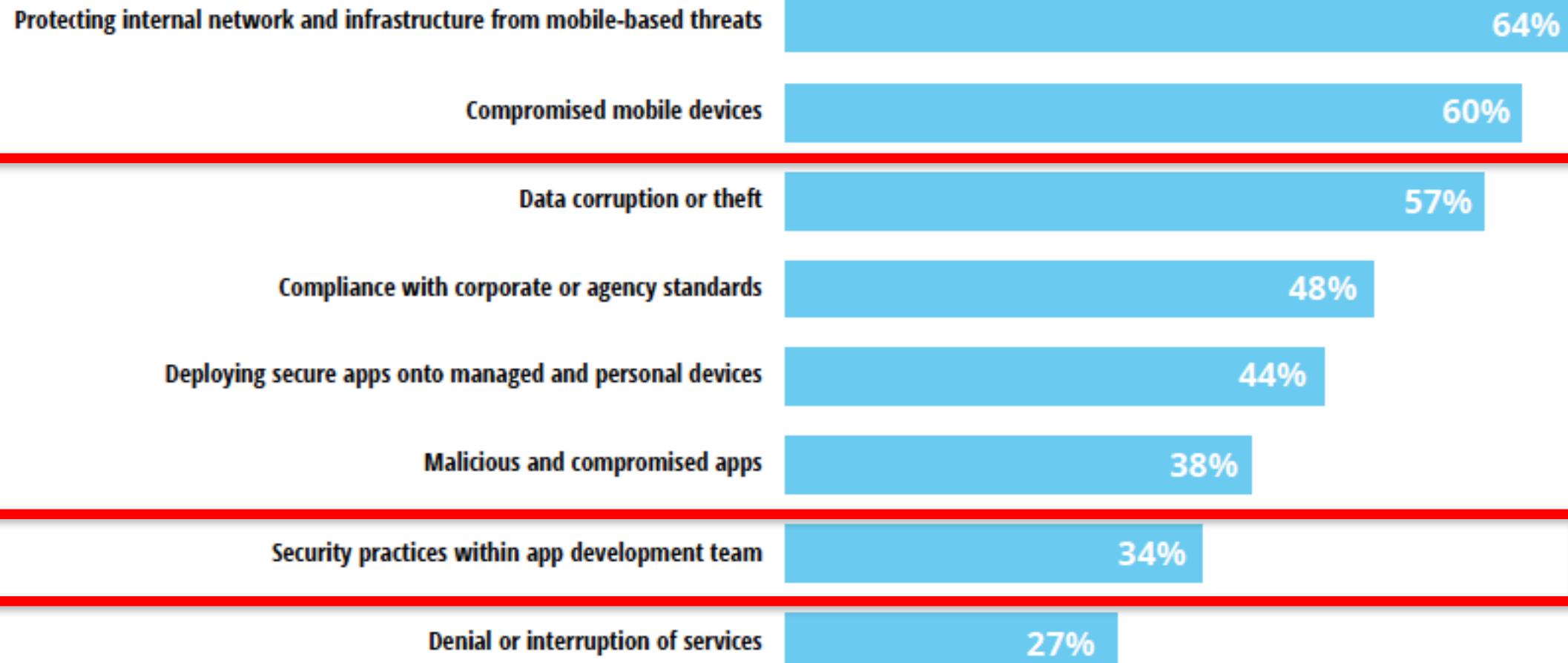
<sup>1</sup> <http://www.zdnet.com/article/ios-versus-android-app-store-versus-google-play-here-comes-the-next-battle-in-the-app-wars/>

<sup>2</sup> [https://go.apperian.com/rs/300-EOJ-215/images/Apperian%202016%20Executive%20Enterprise%20Mobility%20Report\\_FINAL\\_20160216.pdf?aliid=16373787](https://go.apperian.com/rs/300-EOJ-215/images/Apperian%202016%20Executive%20Enterprise%20Mobility%20Report_FINAL_20160216.pdf?aliid=16373787)

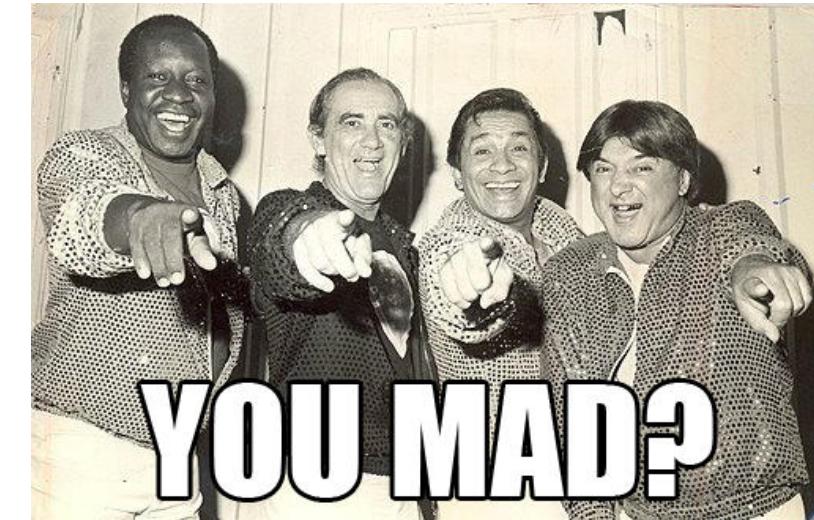
- BYOD?
  - What and Why?

## Which mobile security issues are you concerned about?

Multiple responses allowed



- BYOD?
  - What and Why?
- BYOD Adoption
  - 74% using or adopting BYOD
  - Governments
- Enterprise Mobile Security
  - MAM (Mobile Application Management)
  - MIM (Mobile Information Management)
  - MDM (Mobile Device Management)



# Protection Claims

“prevent employees from opening files in unsecured apps, backing up business data to personal cloud-based services, or copying and pasting business ...”

“Detect OS tampering and other policy violations”

“...remotely lock or wipe the device.”

“Protect mobile apps and servers from being hacked...”



# Enterprise Mobile Security Features

Jailbreak

Device PIN



JB Detection

Container

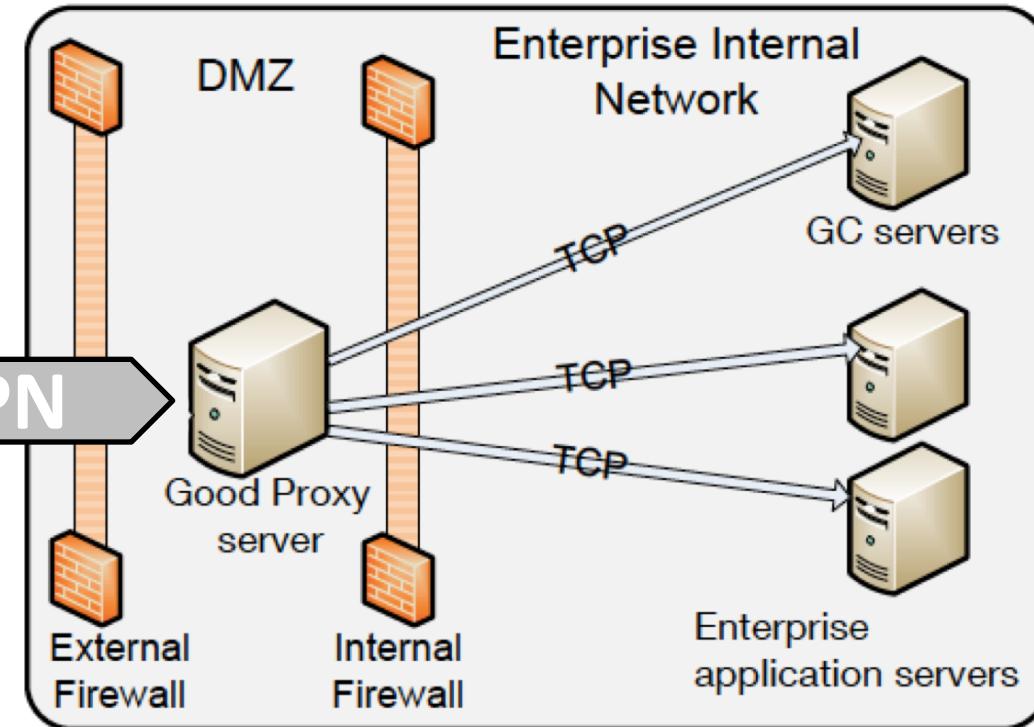


Container  
Password

Container  
Encryption

Application VPN

App  
Wipe / Lock



- Acquired by Blackberry in Nov 2015
- Top 5 EMS Solution Providers



- Acquired by Blackberry in Nov 2015
- Top 5 EMS Solution Providers
- GFE received CC EAL4+ in 2013 and GD solution in 2016
- GD platform used as a foundation to the GCS to replace GFE
- GD platform allows developers to create and distribute apps that integrates with the GD services framework

# GFE vs GCS

	<b>Good For Enterprise (GFE)</b>	<b>Good Collaboration Suite (GCS)</b>
Email	✓	Good Work
MDM	✓	✓
File Share	Local File Storage Only	Good Share – Access enterprise file share
Instant Messaging	✗	Good Connect
Intranet Access	✗	Good Access
Cloud Deployment	✗	✓
Integrated MAM	✗	✓
Common Platform	✗	✓

iOS Versions	7.0	7.1	8.0	8.0	8.1.3	9.0	9.1	10.0
	7.0.0	7.1.2	8.1	8.3	8.4	9.0.2	9.3.5	10.0

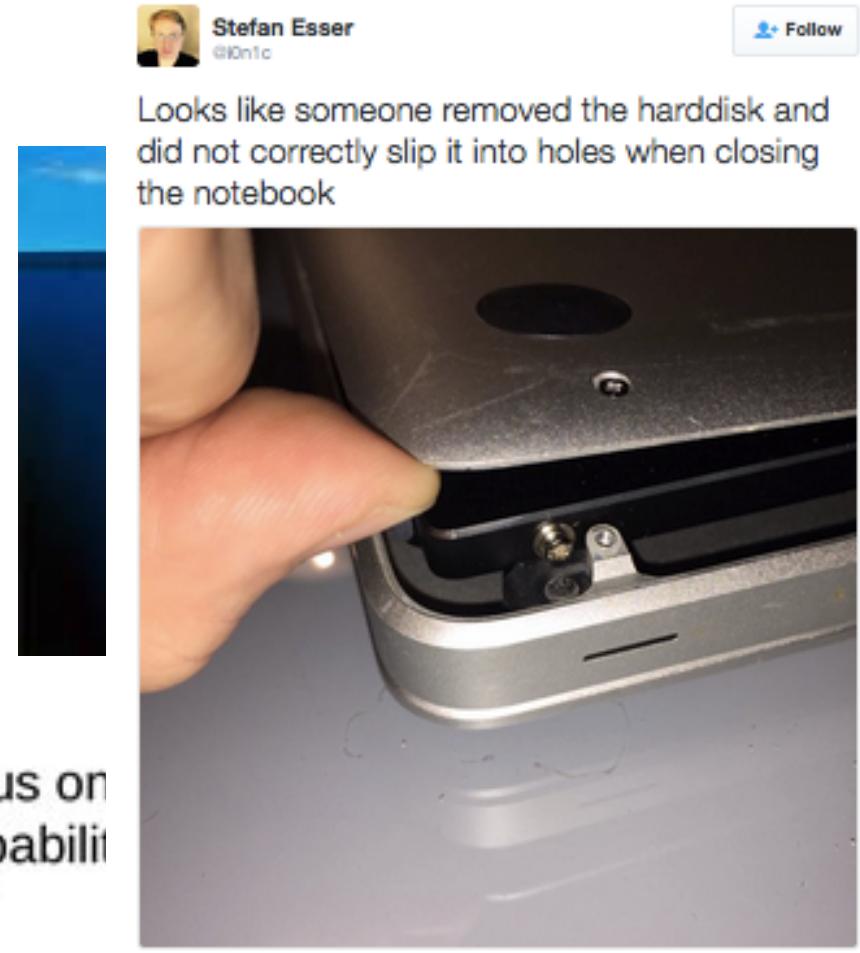
- Check out Stefan Esser's talk on “iOS 678 Security - A Study In Fail”



# What about root?

- Non-Jailbroken Devices?
  - Resign via developer certificate
    - But apps will need to be reactivated
- Physical Access
  - DROPOUTJEEP (think NSA, GCHQ)
  - Lost Devices / Stolen Devices
- Remote Attacks

(TS//SI//REL) The initial release of DROPOUTJEEP will focus on implant via close access methods. A remote installation capability for a future release.



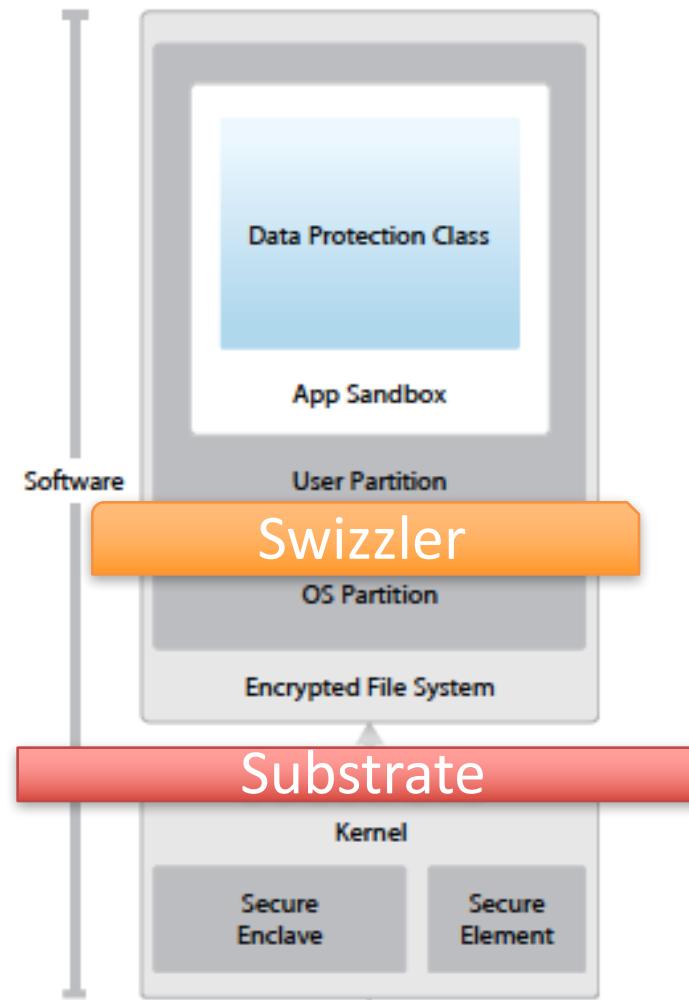
<sup>1</sup><http://www.tripwire.com/state-of-security/vulnerability-management/creating-iphone-rootkits-and-like-the-nsas-dropout-jeep/>

<sup>2</sup><https://blog.fortinet.com/post/ios-malware-does-exist>

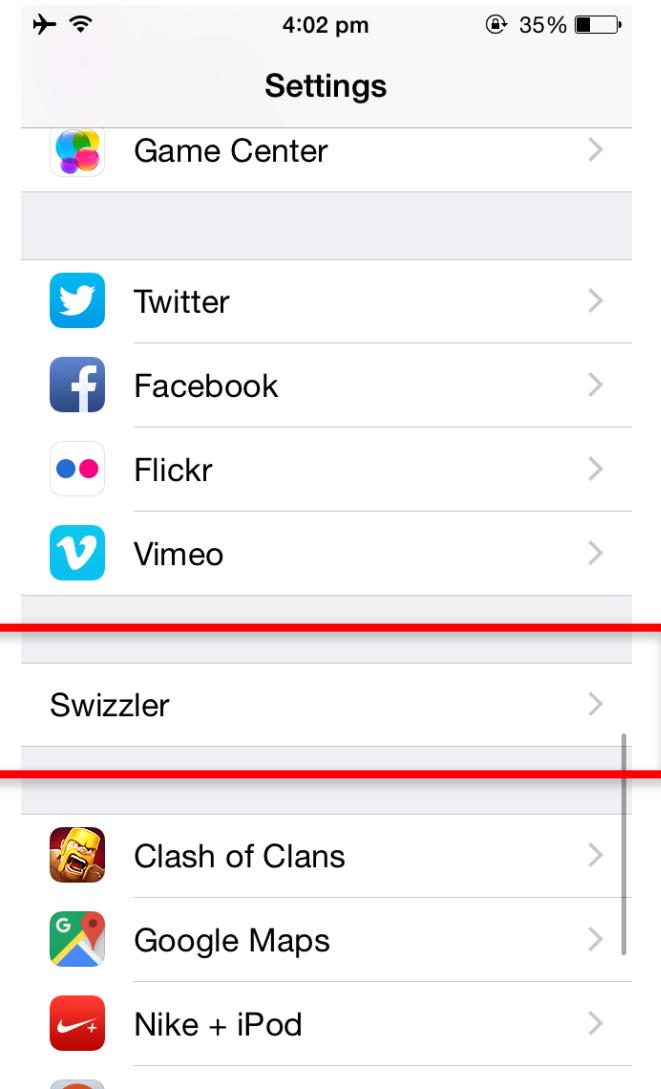
- Not normal pen testing...
  - Not just setting proxy and using Burp
- I'm not attacking the application
- Changing the environment in which the application runs.
- Not new. API Hooking and DLL Injections on Windows.  
LD\_PRELOAD on Linux. I'm just doing it on iOS.

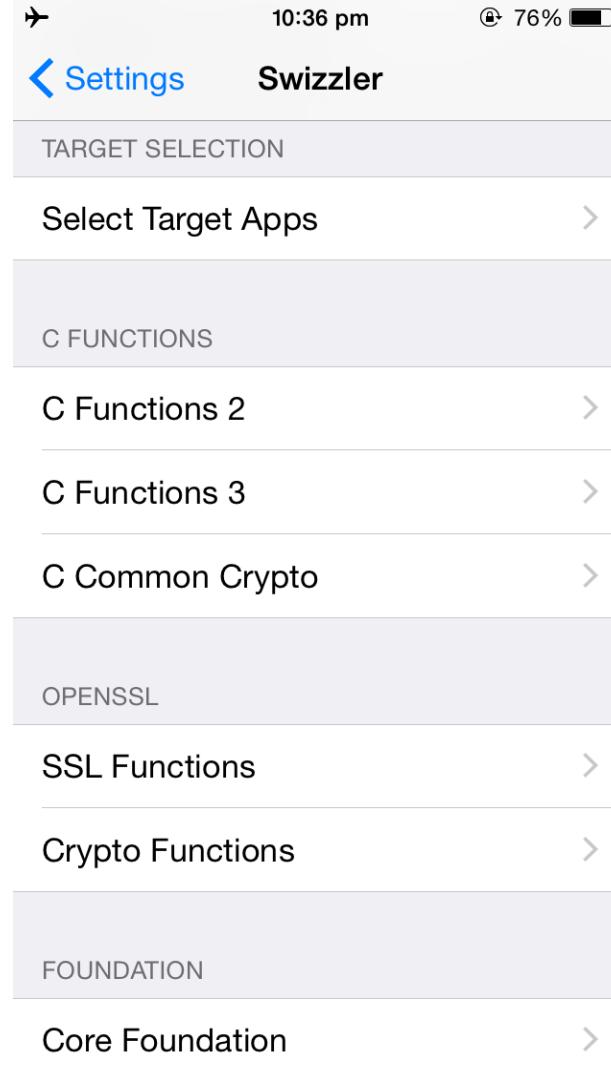
- How do I change the Environment?
- Built an App... More precisely a Dynamic Library (aka tweak)
  - DYLD\_INSERT\_LIBRARIES=Swizzler
- Loads itself before an application starts
  - Control all functionalities of an application

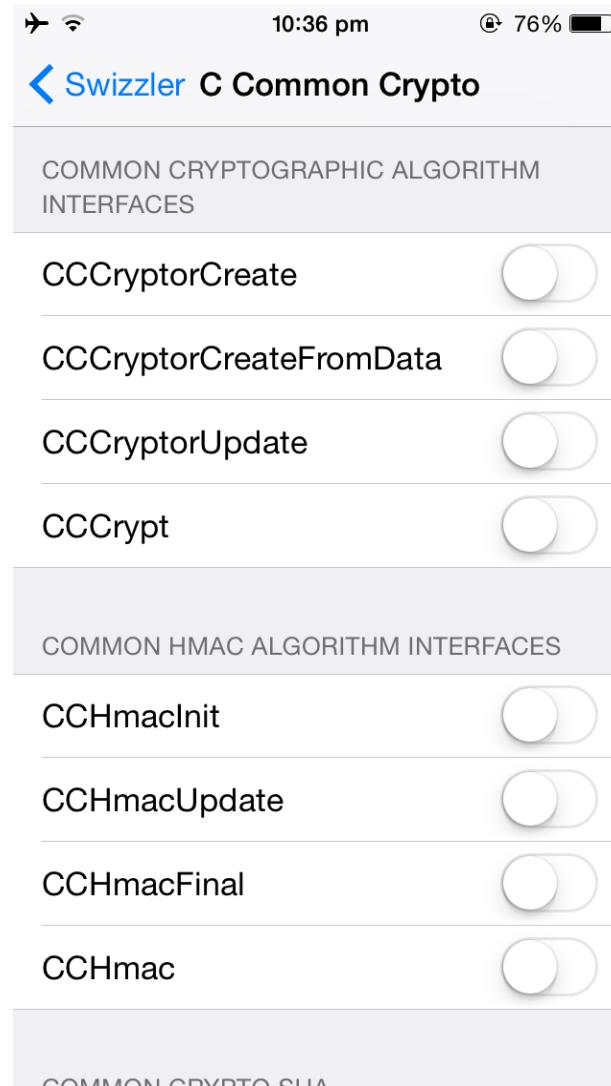
# iOS Security Architecture

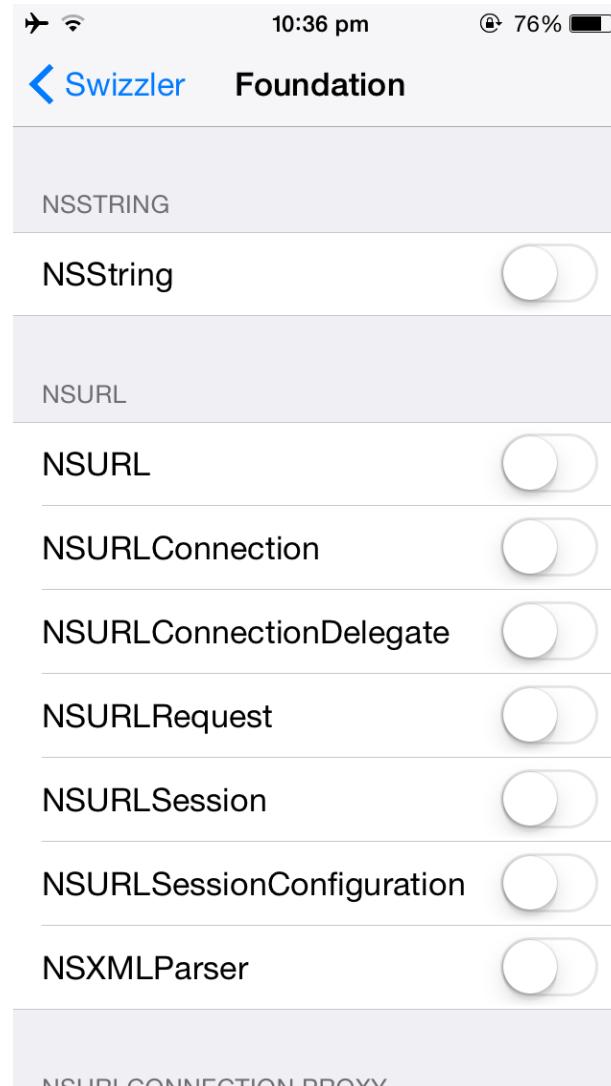


# Swizzler

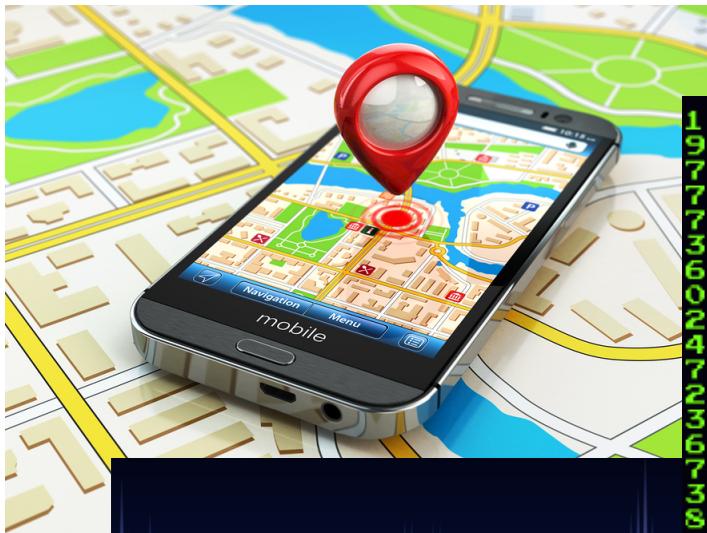




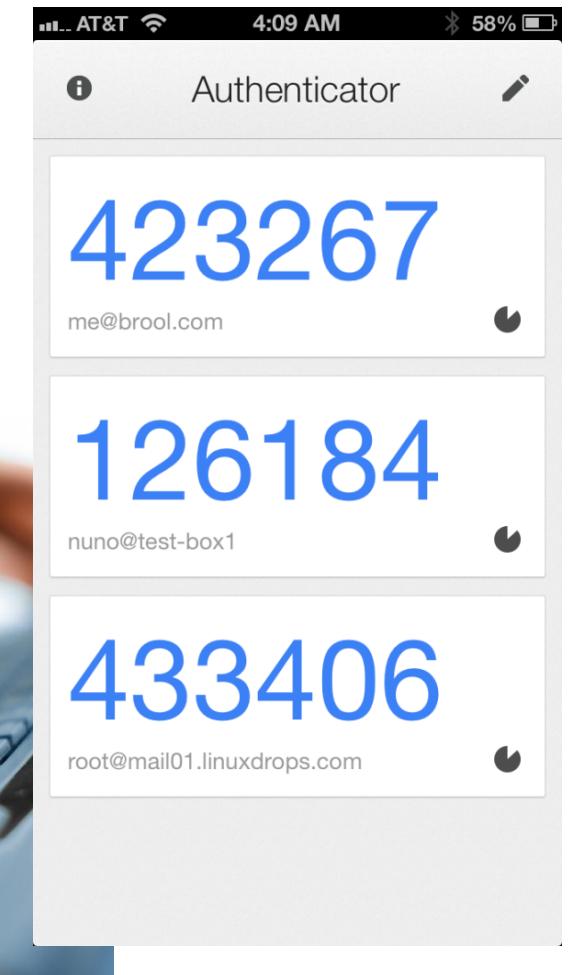




# What else can you control?



```
175332068788239113561461  
992898492054215873849653  
776580029119534902085770  
773159779295169376044130  
787582468794514136744901  
317476763210555803715906  
689499621088484817128713  
035798000108775230999531  
295039334408113901425387  
430938504481352977270716  
764983798005416673095135  
216957228881480189504989  
324000303098498610772385  
6621335428378558511088060  
721251625034618766625203  
340427430100566404195113  
823752342930552204655077
```





Container

App DLP

Jailbreak  
Detection

Jailbreak

Device PIN

Container  
Password

# Local & Network Attacks Against Good EMS







# Local Attacks



Jailbreak

Device PIN



JB Detection

Container



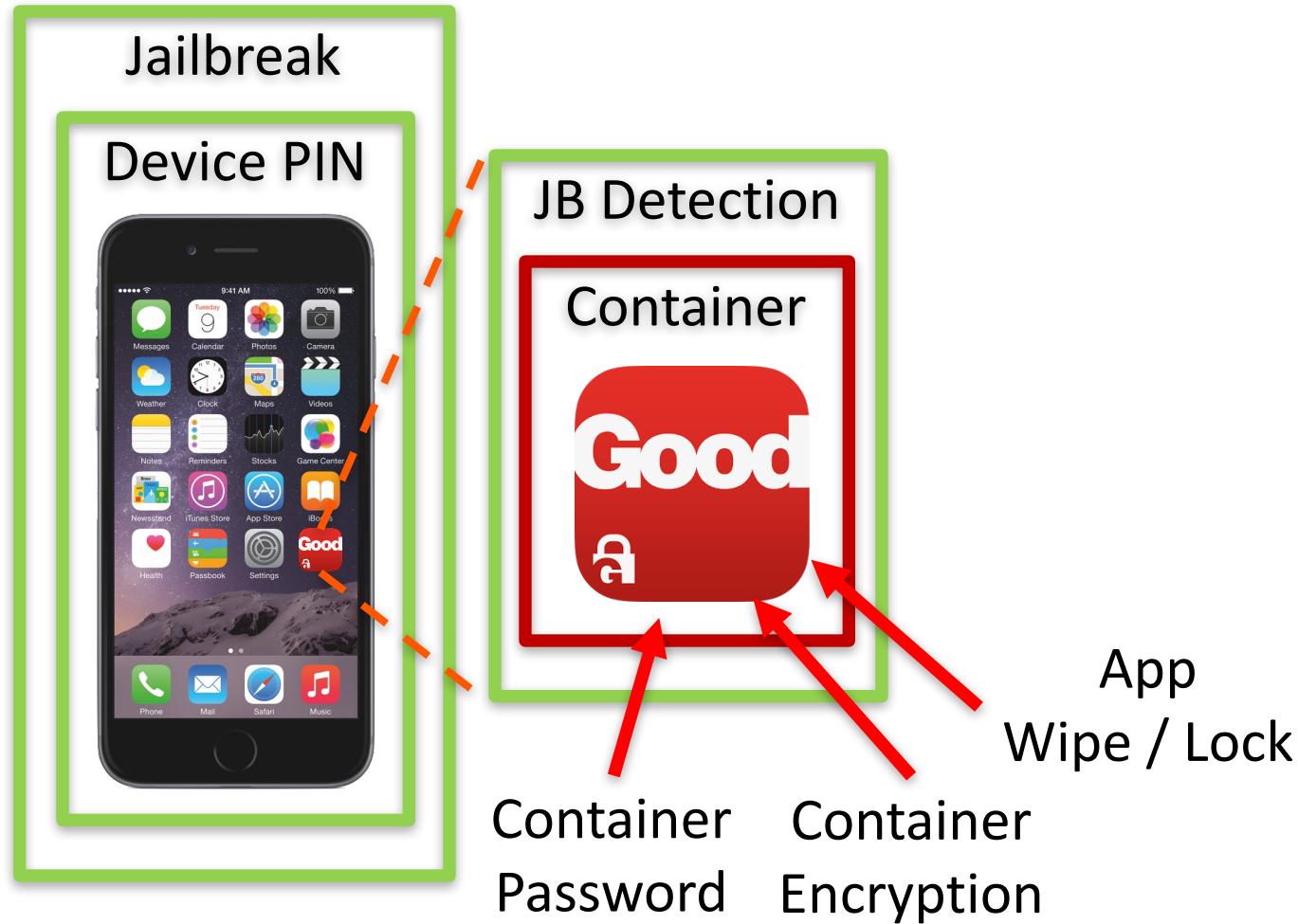
Container  
Password

Container  
Encryption

# Jailbreak Detection

App  
Wipe / Lock





- ✓ Device PIN
- ✓ Jailbreak
- ✓ Jailbreak Detection
- ✗ Container Password
- ✗ Container Encryption
- ✗ App Wipe / Lock

# Blacklist of Files

```
FILE *file = fopen("/Applications/Cydia.app",  
"r");  
if (file) {  
    fclose(file);  
    return JAILBROKEN;  
}
```

```
file = fopen("/usr/bin/ssh", "r");  
if (file) {  
    fclose(file);  
    return JAILBROKEN;  
}
```

```
FILE *replaced_fopen (const char *filename, const  
char *mode) {  
    if (blockPath(filename)) {  
        errno = ENOENT;  
        return NULL;  
    }  
}
```

```
bool blockPath(const char *fpath) {  
...  
NSArray *denyPatterns = [[NSArray alloc]  
initWithObjects: @"Cydia", @"lib/apt",  
@"/private/var/lib/apt", @"var/lib/apt",  
@"/var/tmp/cydia.log", @"etc/apt/",  
@"/var/cache/apt"  
....  
}
```

# Prohibited Functions

```
int pid = fork();
if(pid>=0)
{
    return JAILBROKEN;
}
```

```
pid_t replaced_fork(void){
    if (disableJBDetection())
        return -1;
}
pid_t ret = orig_fork();
return ret;
```

- Jailbreak Detection Methods
  - Blacklist of files
  - Directories
  - Symbolic Links
  - Prohibited Commands
  - File System
  - URL Handles
  - Kernel Parameters & many more ...

# Jailbreak / Policy Implementation

```
GT::GeneralUtilityClass::constructStringList (
    GT::GeneralUtilityClass::tamper_detection_method_t,
    std::vector<std::string, std::allocator<std::string> >
)

loc_2ddaa8:
    *(r5 + 0x150) = 0x1;
    if ((statfs("/", sp + 0x8c0) != 0x0) || ((stack[1160] & 0x1) != 0x0)) goto loc_2ddace;

loc_2ddc48:
    *(r5 + 0x150) = 0xb;
    r0 = fork();
    if (r0 != 0xffffffff) goto loc_2de498;
```

# Jailbreak / Policy Implementation

- GD::GDSecureStorage::handleWrongPwd
- GD::GDSecureStorage::wipeDevice
- GD::PolicyProcessor::processLockAction
- GD::GDLibStartupLayer::checkPartialCompliance
- GD::PolicyComplianceChecker::checkComplianceUnlocked
- GD::PolicyComplianceChecker::checkComplianceLocked

Jailbreak

Device PIN



JB Detection

Container



Container  
Password

Container  
Encryption

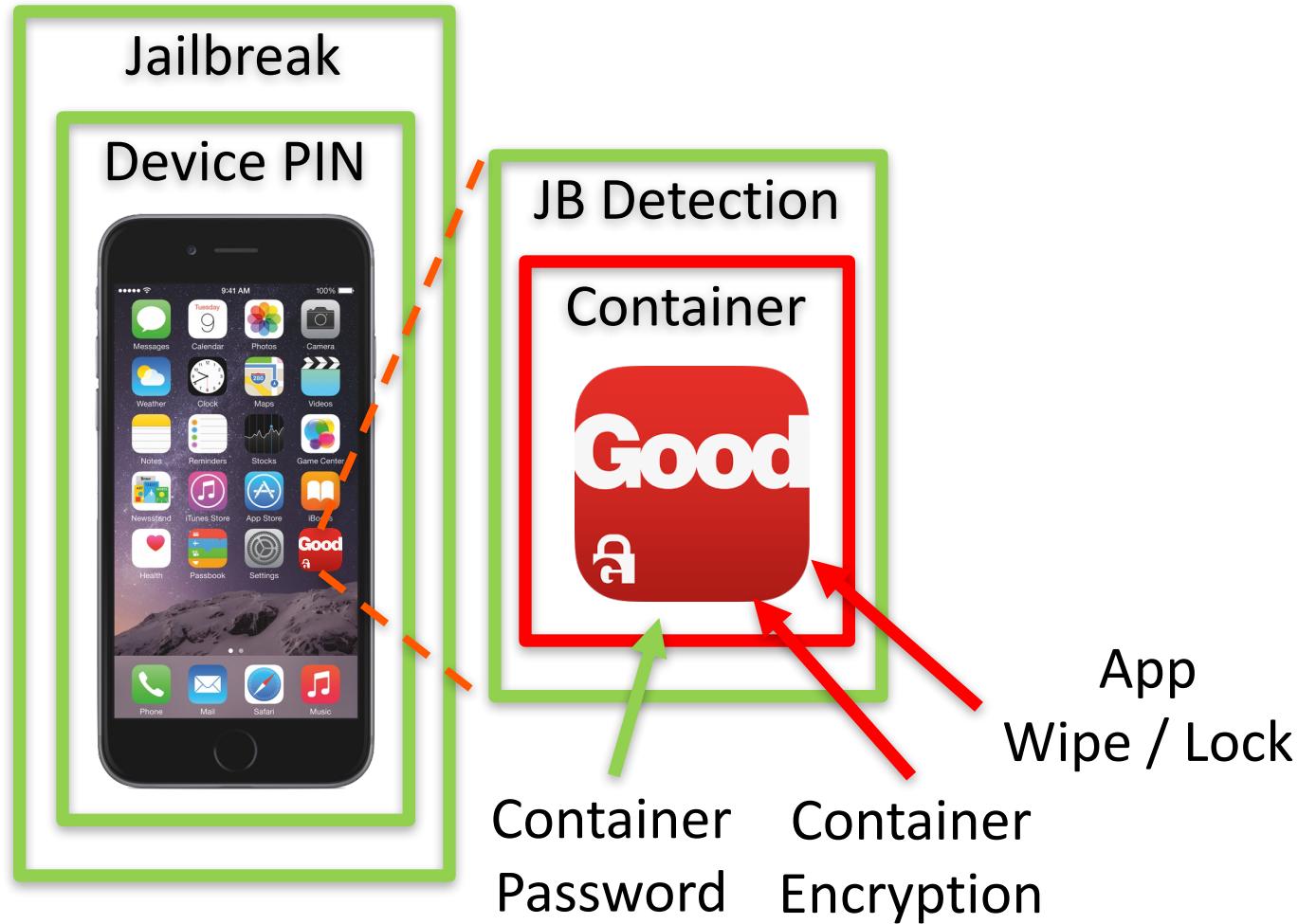
App  
Wipe / Lock

# Password Bruteforce

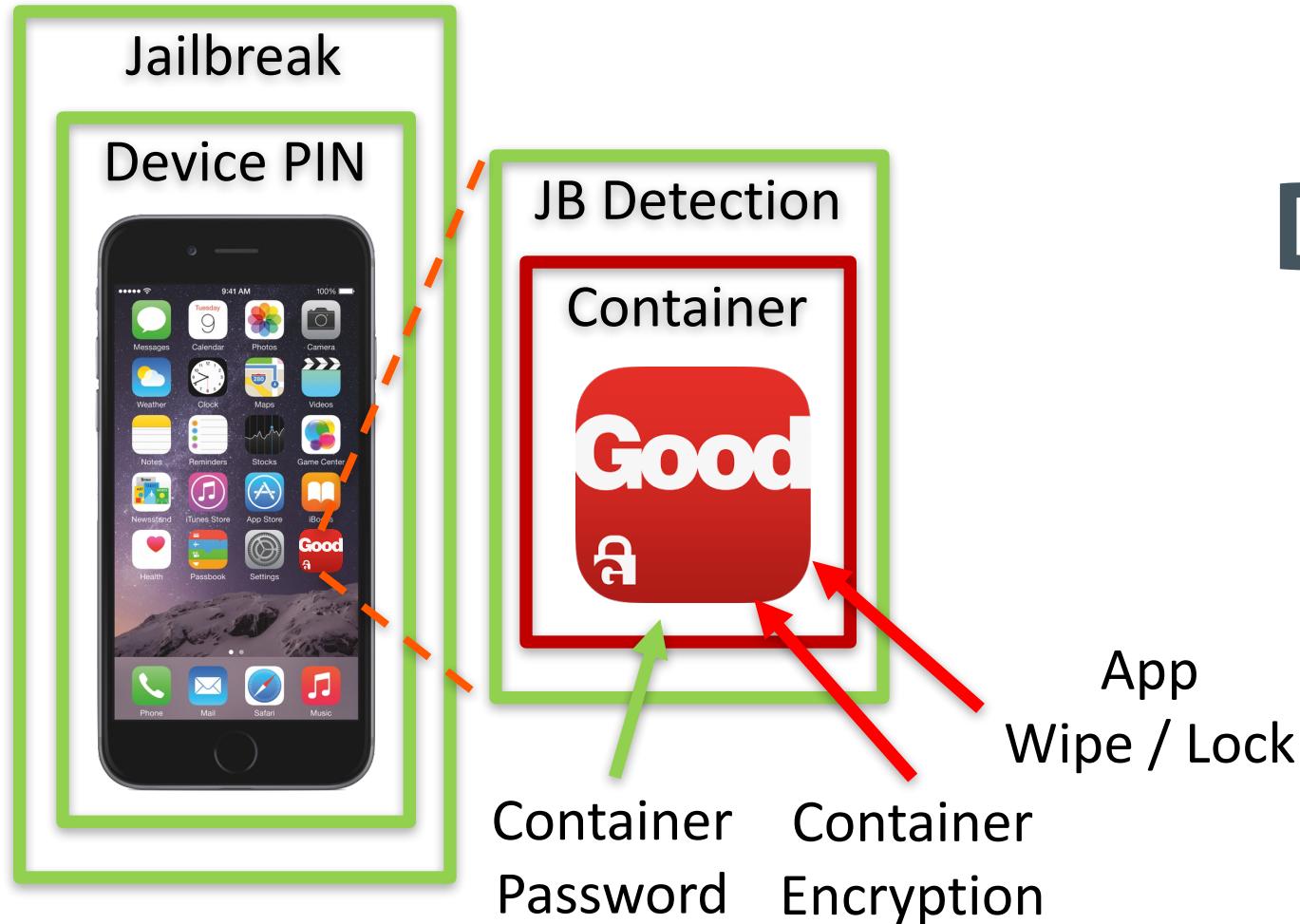


VANTAGEPOINT

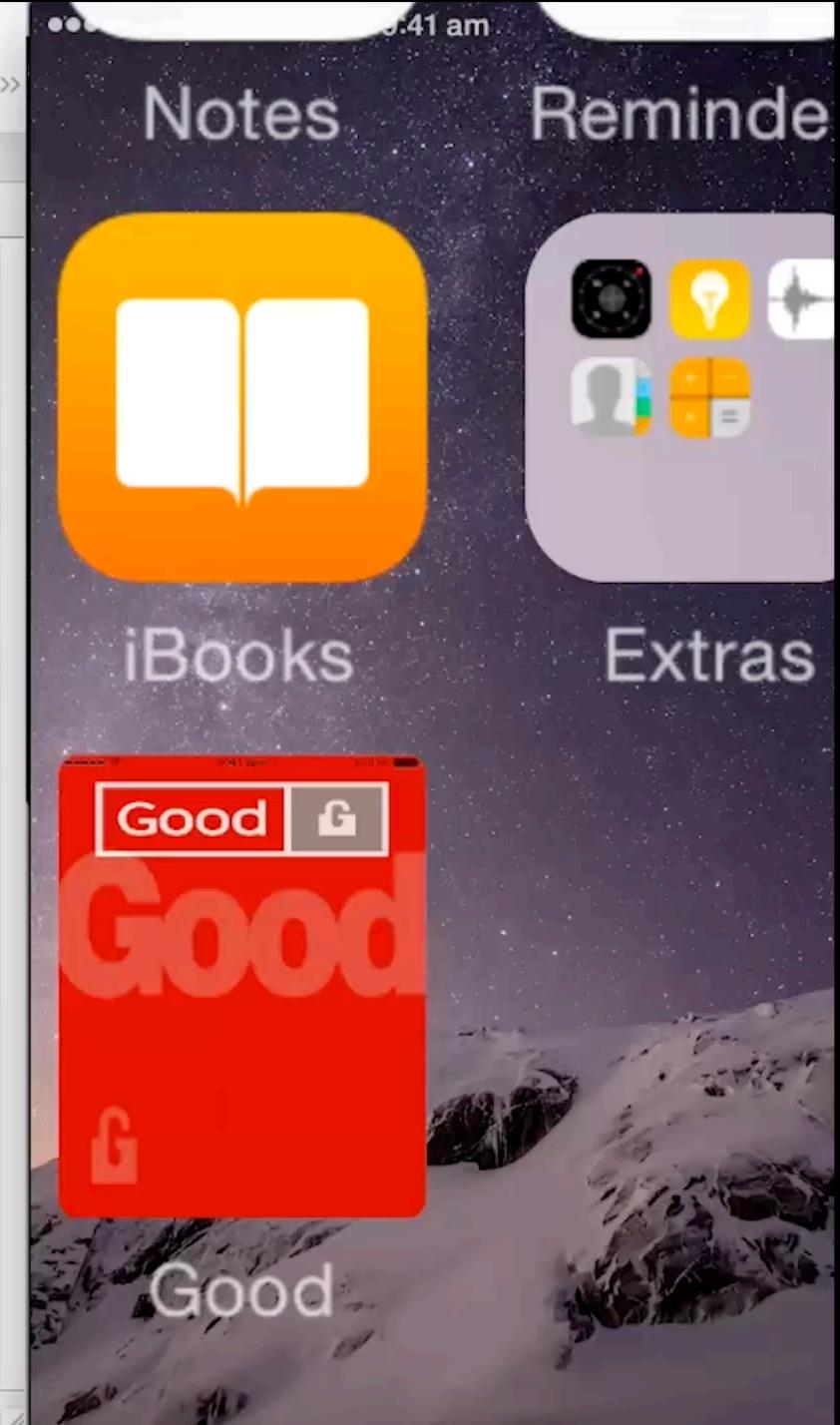
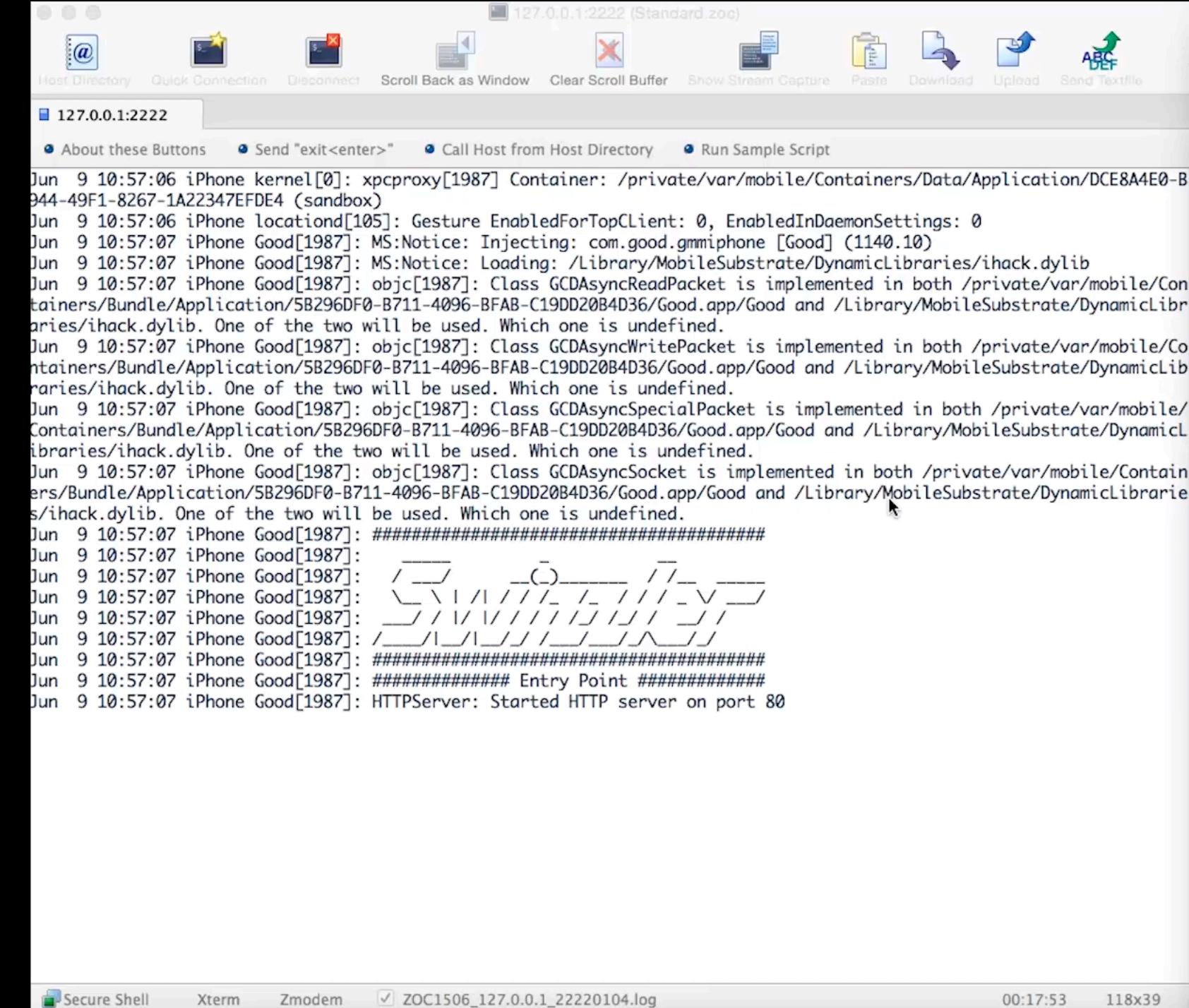


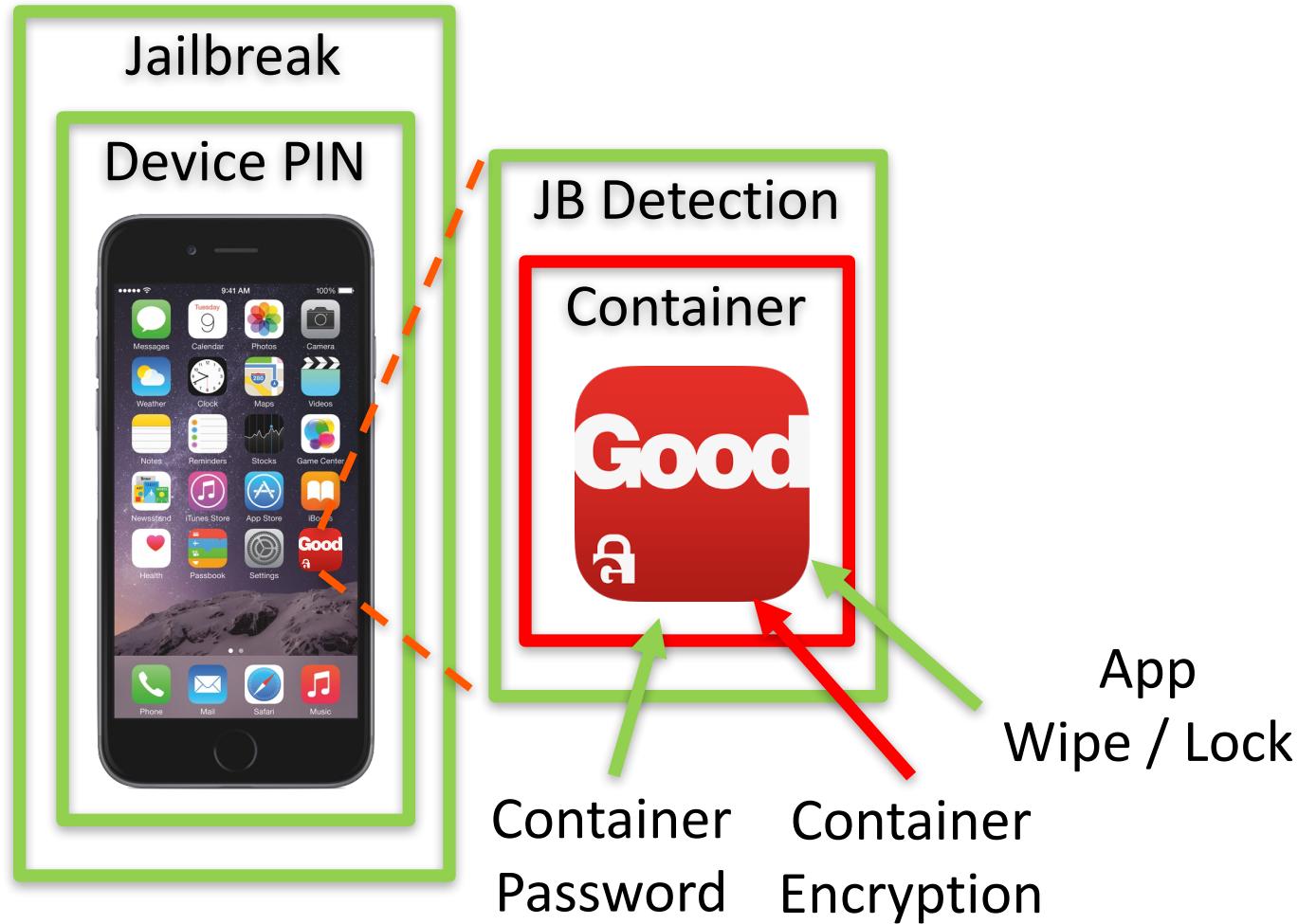


- ✓ Device PIN
- ✓ Jailbreak
- ✓ Jailbreak Detection
- ✓ Container Password
- ✗ Container Encryption
- ✗ App Wipe / Lock



# Disable App Lock & Device Wipe





- ✓ Device PIN
- ✓ Jailbreak
- ✓ Jailbreak Detection
- ✓ Container Password
- ✗ Container Encryption
- ✓ App Wipe / Lock

Jailbreak

Device PIN



JB Detection

Container



Container  
Password

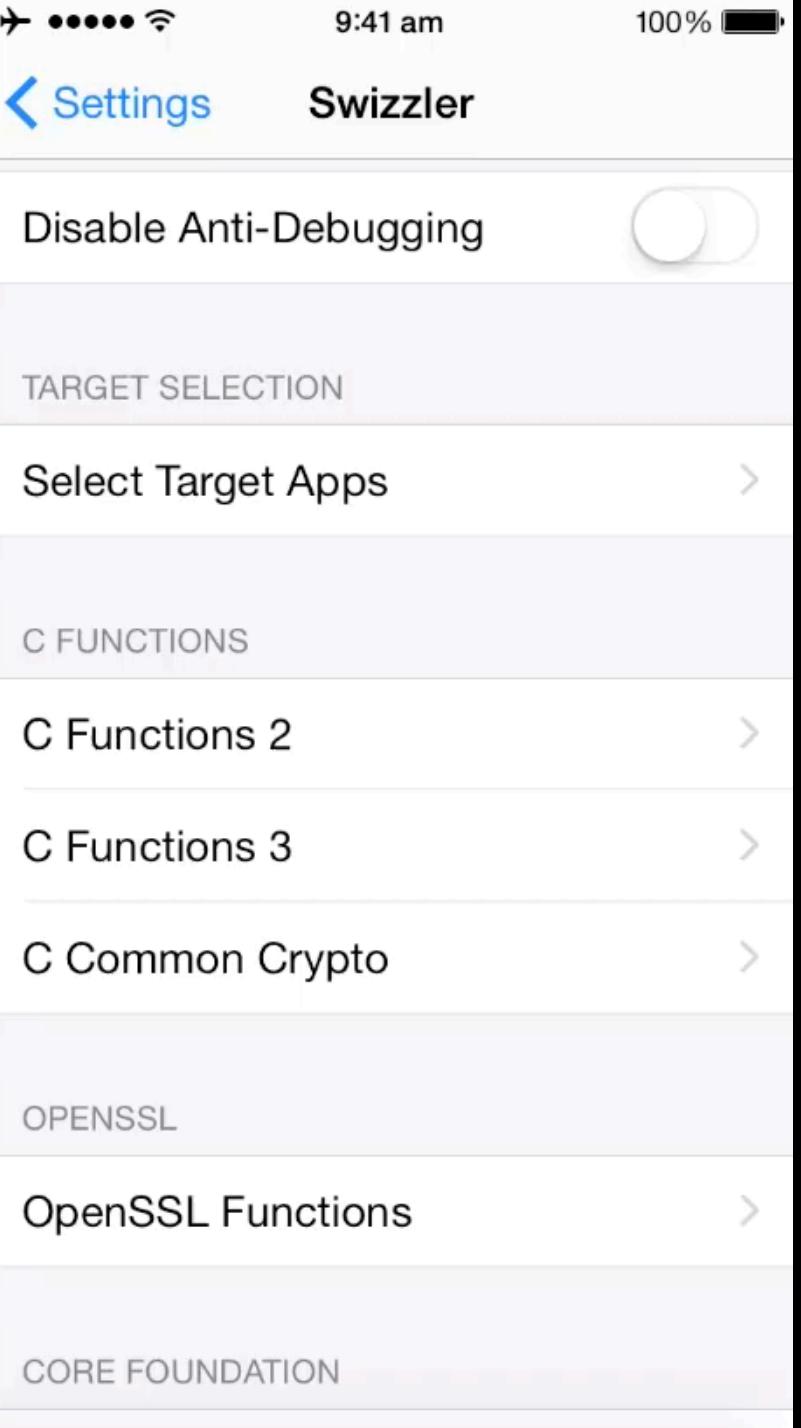
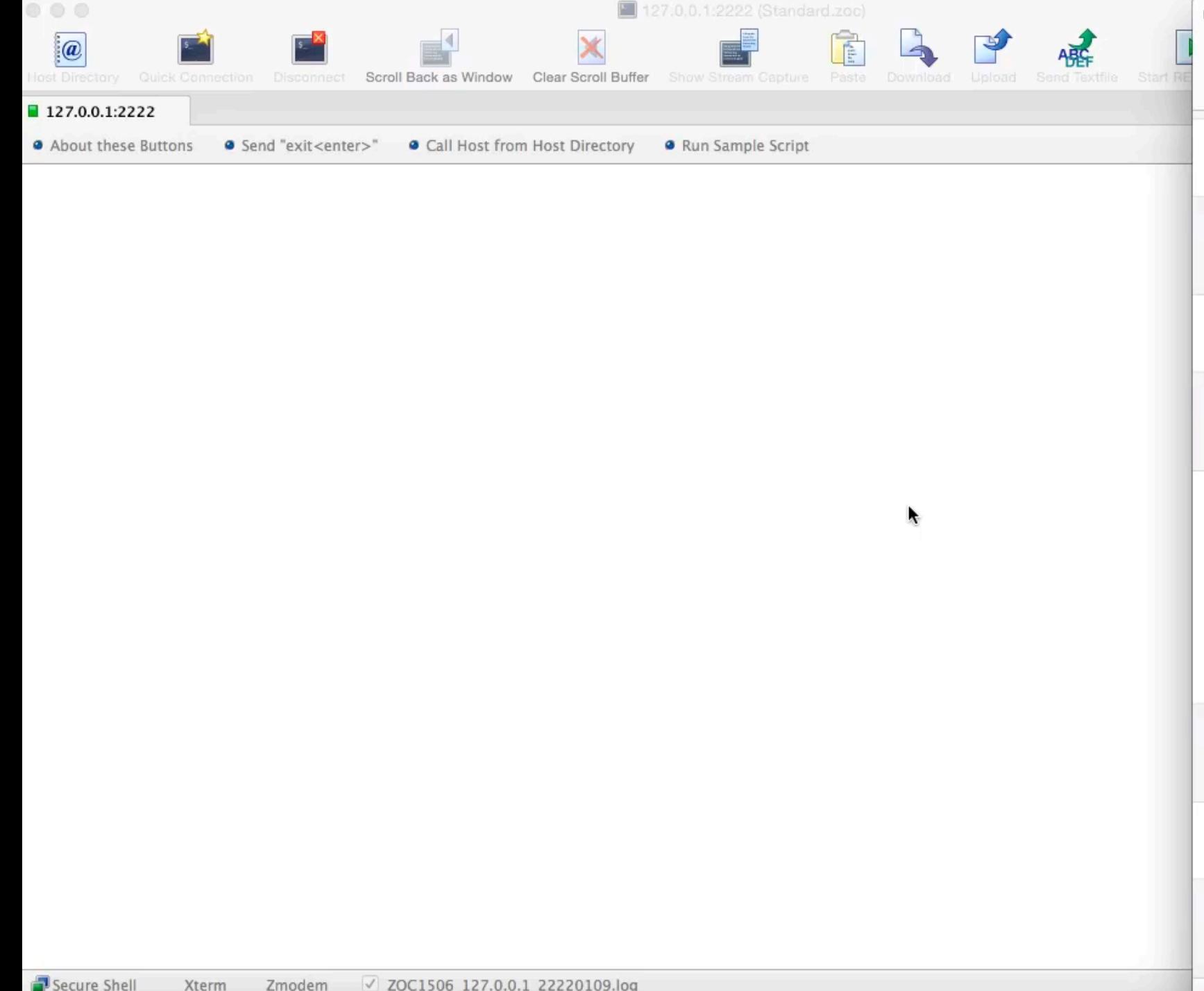
Container  
Encryption

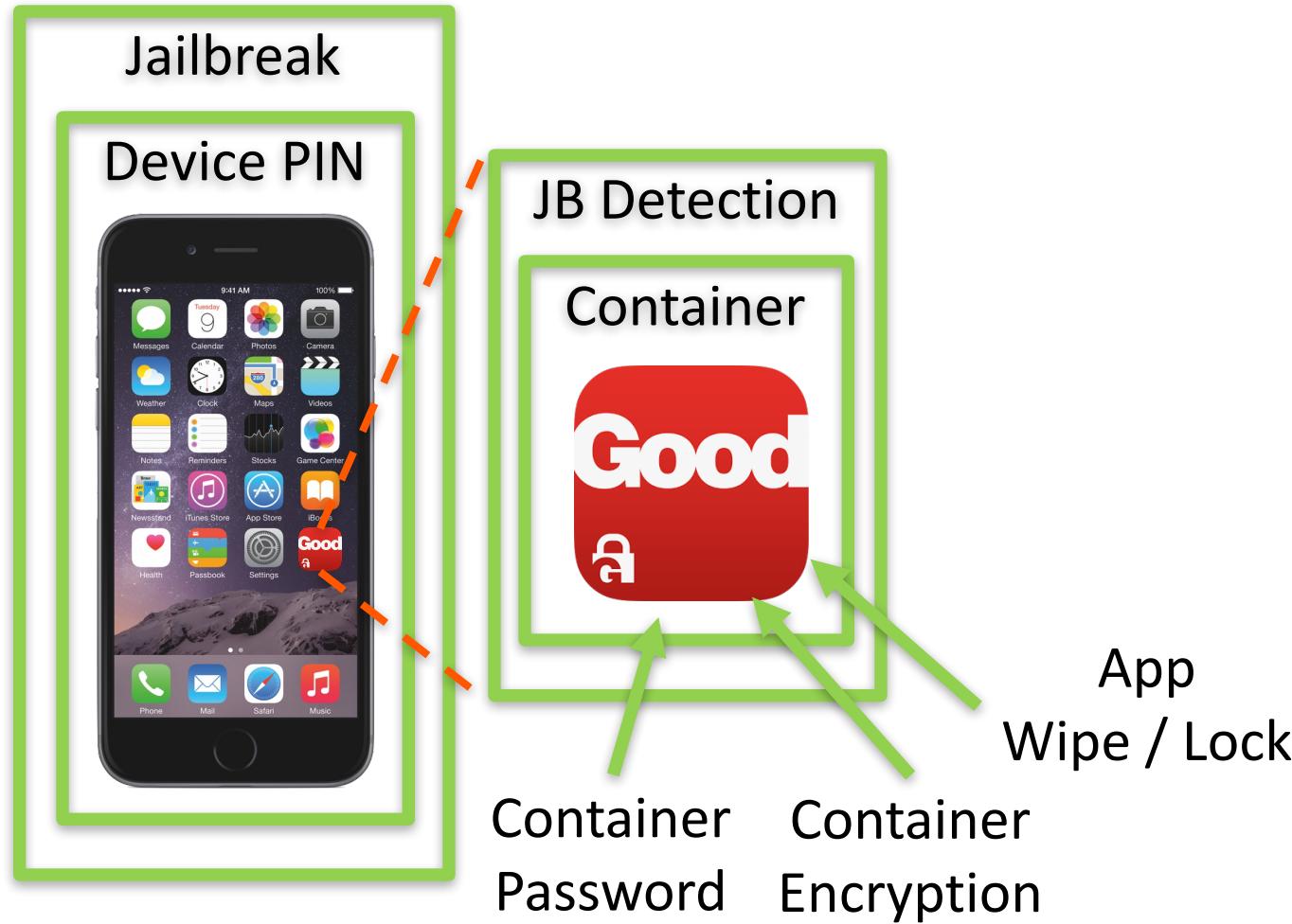
App  
Wipe / Lock

# Containerization



VANTAGEPOINT





- ✓ Device PIN
- ✓ Jailbreak
- ✓ Jailbreak Detection
- ✓ Container Password
- ✓ Container Encryption
- ✓ App Wipe / Lock

```
x Python %1 X bash %2 X Python %3 X bash %4  
vincent: /Users/vincent/Desktop/iostesting/Good  
>>> |
```

# Network Attacks



Jailbreak

Device PIN



JB Detection

Container

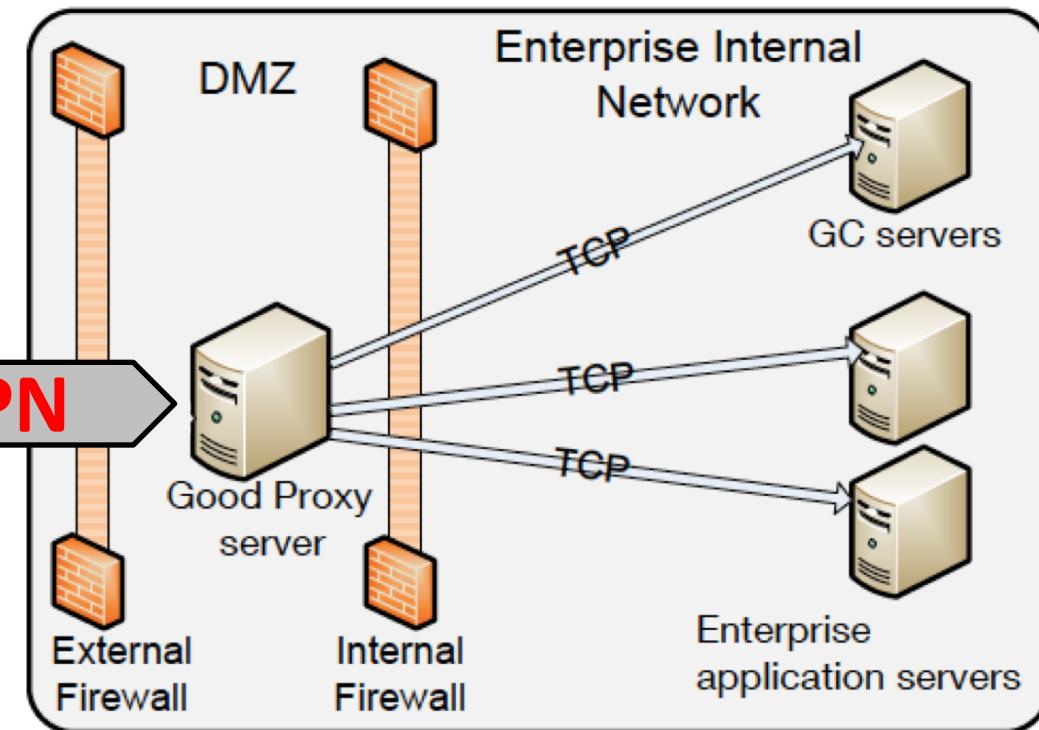


Application VPN

App  
Wipe / Lock

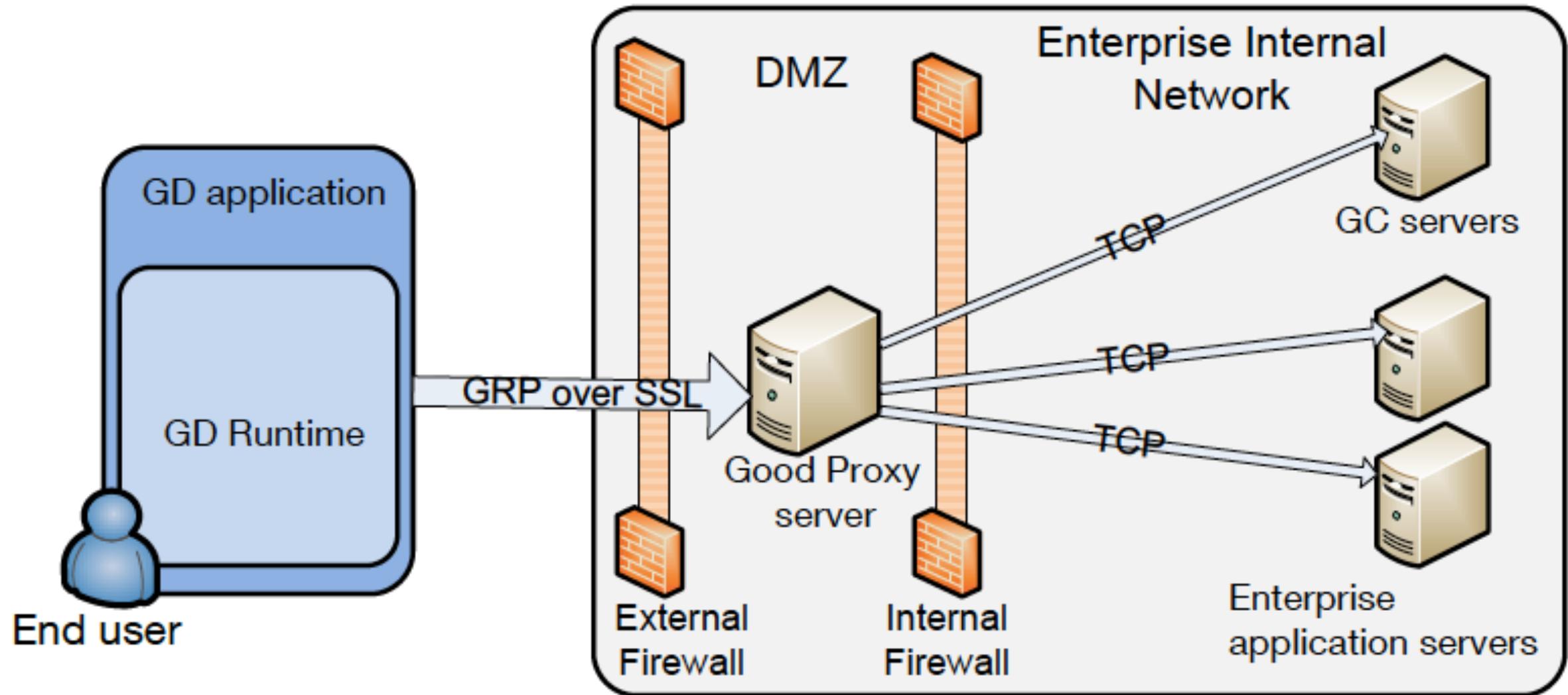
Container  
Password

Container  
Encryption



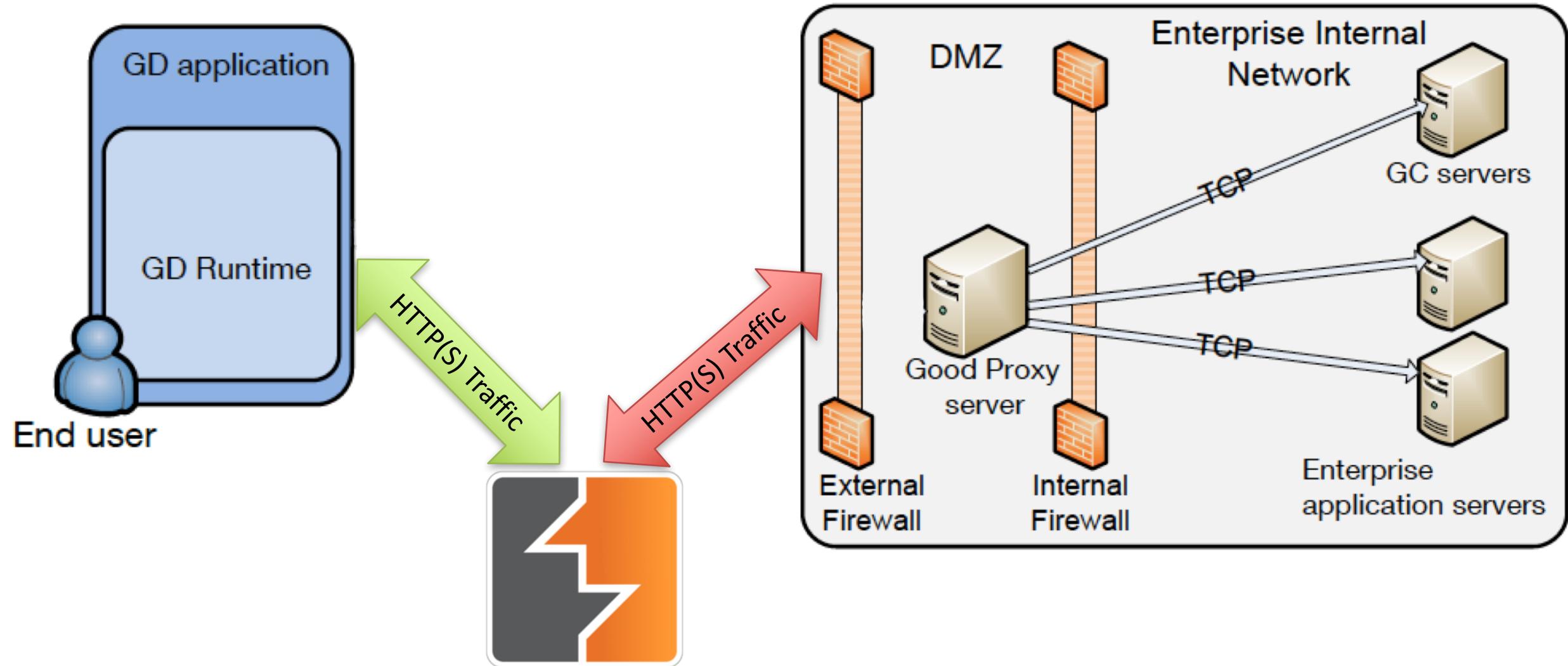
# Application VPN



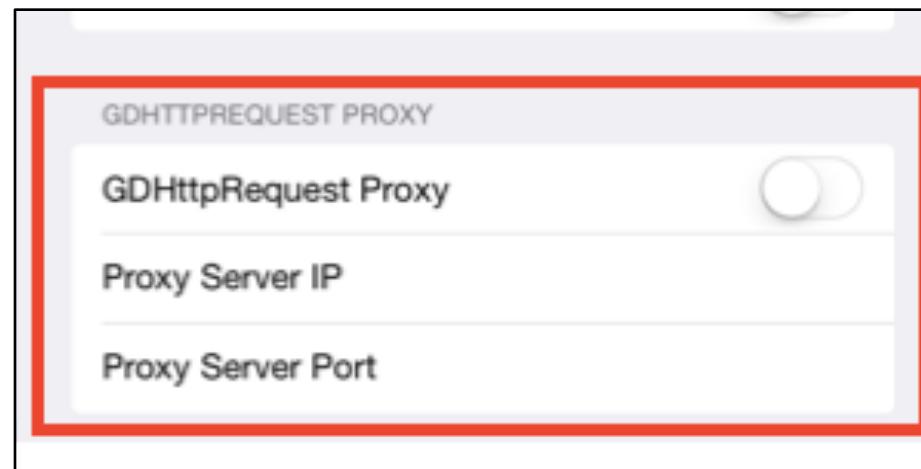


Two methods of communication with the enterprise application server,

1. GDHttpRequest
2. Native URL Loading (NSURL, NSMutableURLRequest, etc.)

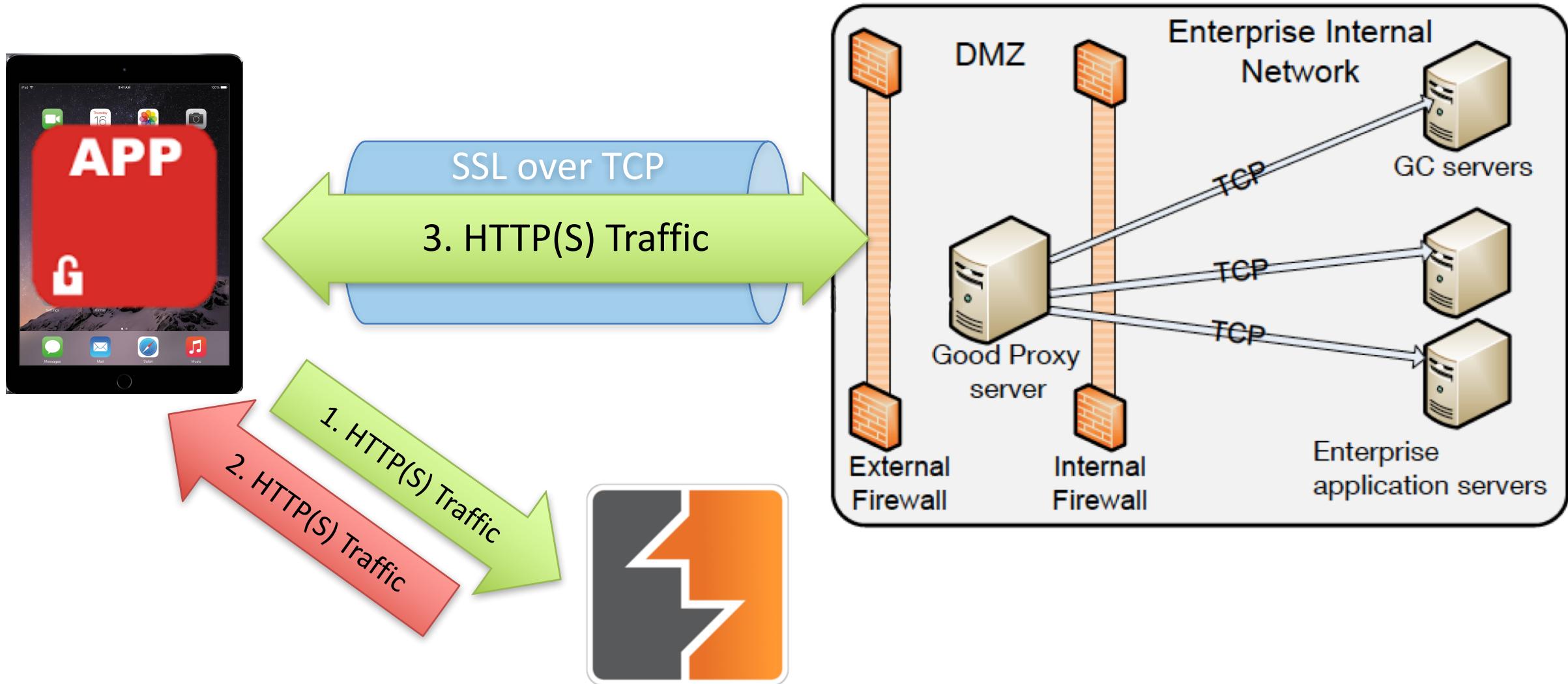


- Part of the GD SDK
  - `#import <GDNETiOS.h>`
- Easy to enable proxy
  - `[GDHttpRequest enableHttpProxy:ip withPort:port];`
  - `[GDHttpRequest disablePeerVerification];`



- Enabled via GDURLLoadingSystem Class
  - [GDURLLoadingSystem enableSecureCommunication]
- Enabled by default
- Proxying traffic is harder
- Doesn't obey iOS network proxy settings
- Swizzle [NSURLConnection initWithRequest]

# Hooking the Network



Burp Intruder Repeater Window Help

Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Options Alerts

Intercept HTTP history WebSockets history Options

Filter: Hiding CSS, image and general binary content

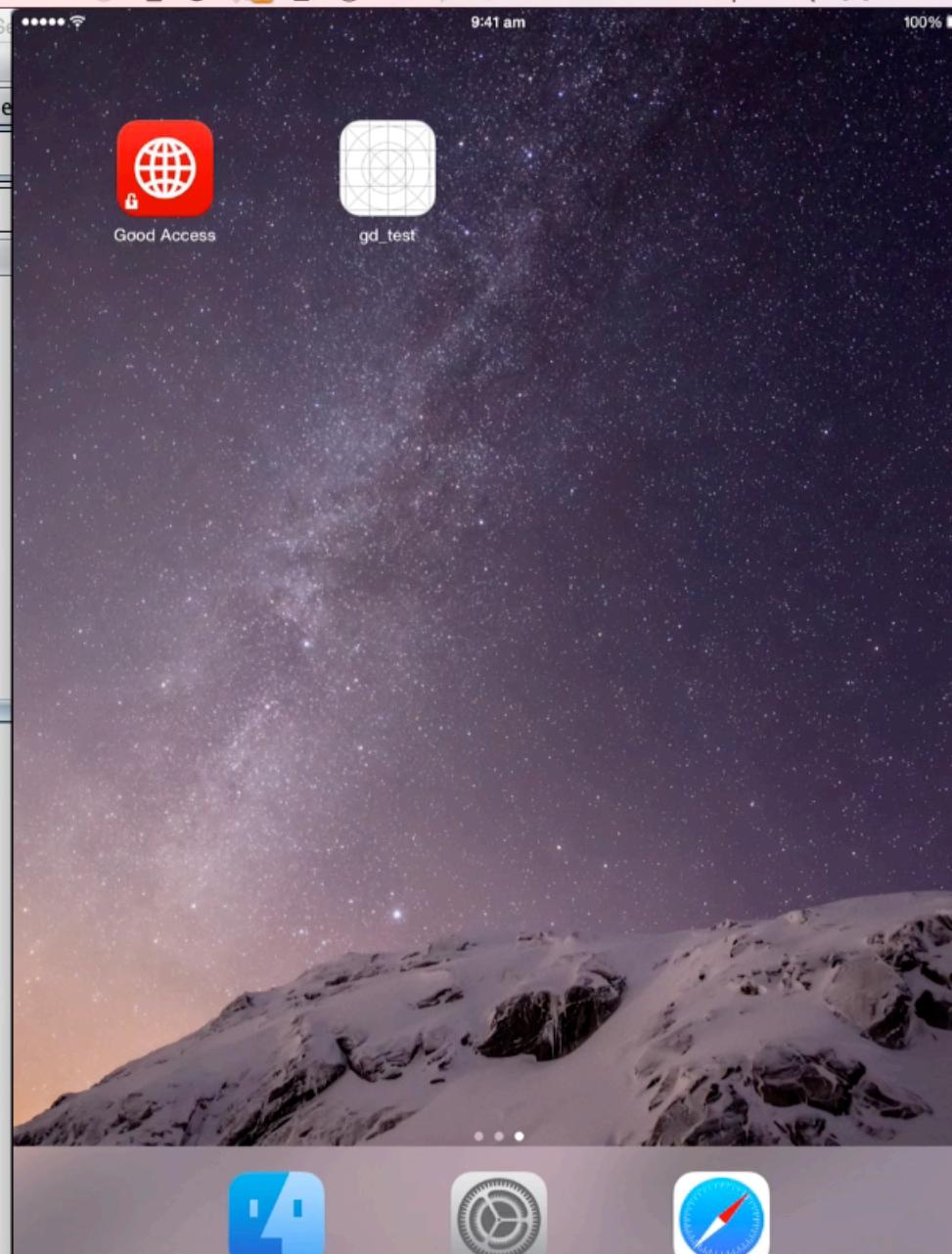
#	Host	Method	URL	Params	Edited	Status
---	------	--------	-----	--------	--------	--------



Good Access



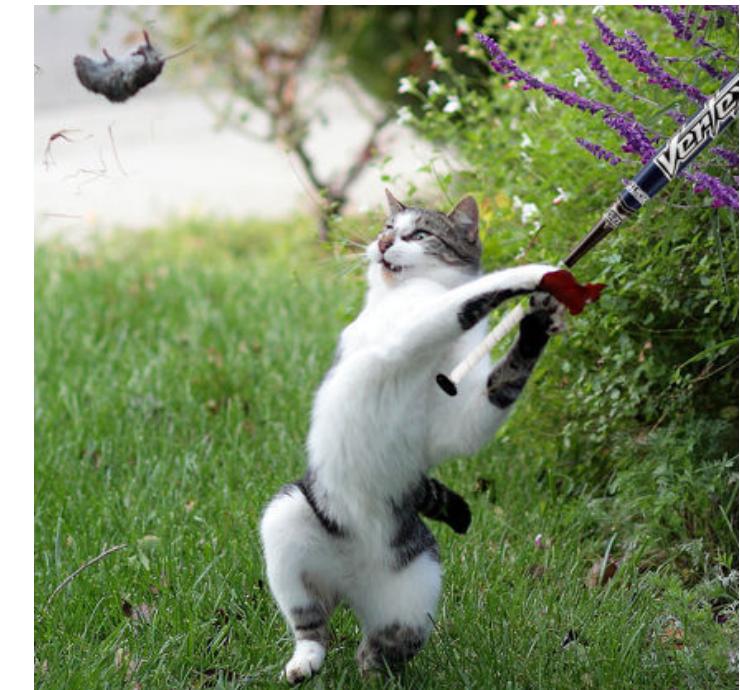
gd\_test



# Does everything suck?

- Local device access is important, but remote attacks possible.
  - <https://www.youtube.com/watch?v=STIHO2XOOiM>
- Employee Education.
  - Be careful of USB chargers. BadUSB.
- Intranet == Internet
- Additional security checks on apps

- Think outside the box to break the box!
- BYOD Policy helps to a certain extent, but such attacks will always be possible.
- Do not blindly trust what the vendors sell you.
- <https://github.com/vtky/Swizzler>





vincent.vtky@outlook.com



<https://www.linkedin.com/in/vincenttky>



@vincent\_tky