

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. Story of Alice & Bob
 - Local & Network Attacks against Good EMS

- > 1.4m Applications¹ in iOS App Store
 - ~10% in Business Category
- 35% of Enterprises have an Enterprise App Store²
- 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.



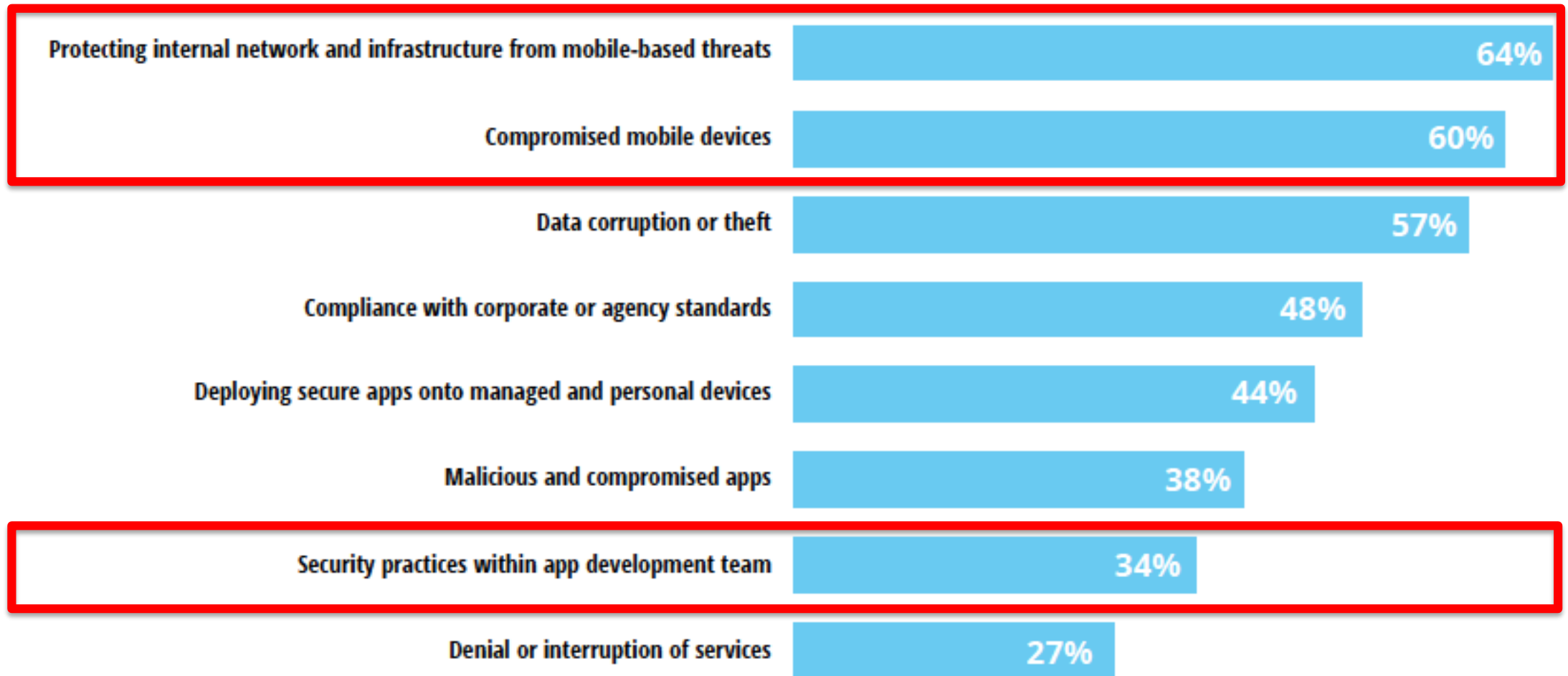
¹ <http://www.zdnet.com/article/ios-versus-android-apple-app-store-versus-google-play-here-comes-the-next-battle-in-the-app-wars/>

² https://go.appierian.com/rs/300-EOJ-215/images/Appierian%202016%20Executive%20Enterprise%20Mobility%20Report_FINAL_20160216.pdf?aliId=16373787

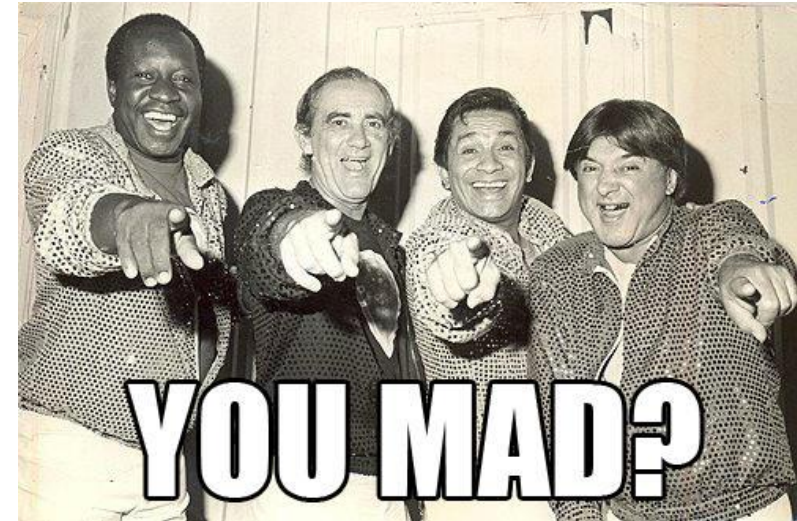
- BYOD?
 - What and Why?

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)



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

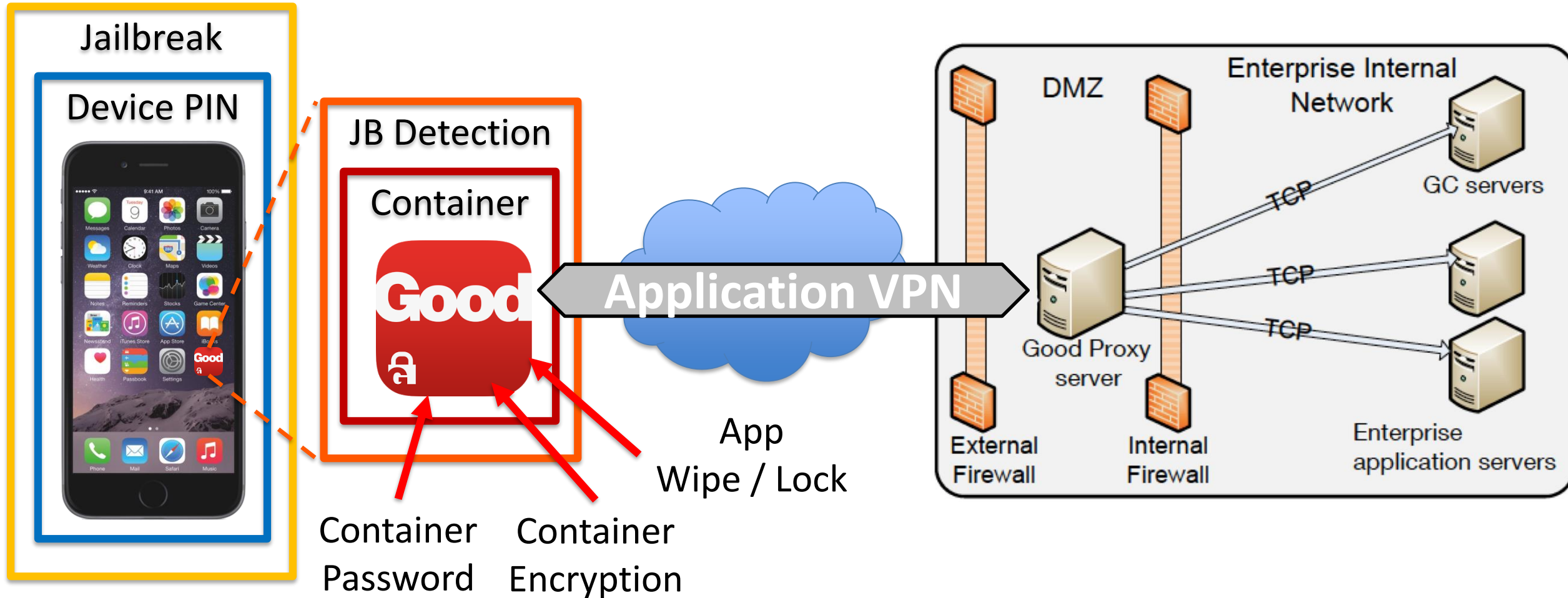


“Detect OS tampering and other policy violations”

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

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

Enterprise Mobile Security Features











- 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

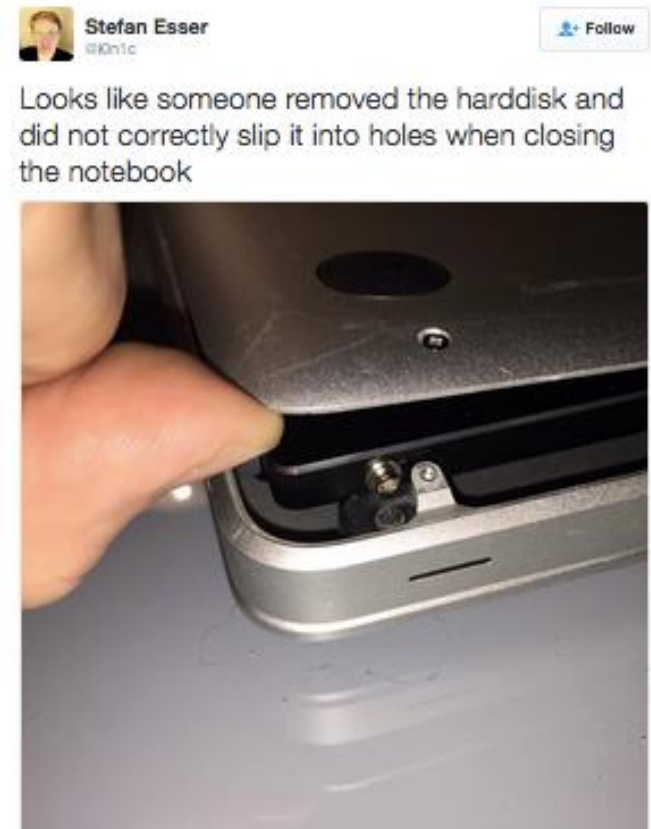
	GFE	GCS
Email	✓	Good Work
MDM	✓	✓
File Share	Local File Storage Only	Good Share – Access enterprise file share
Instant Messaging	✗	Good Connect
Intranet Access	✗	Good Access
Cloud Deployment	✗	✓
Integrated MAM	✗	✓
Common Platform	✗	✓

iOS Versions	 7.0	 7.1	 8.0	 8.0	 8.1.2	 9.0	 9.1	 10.0
Jail Broken?	evasi0n7	Pangu	Pangu8	TaiG	TaiG	Pangu9	Not Public	Not Public

- Non-Jailbroken Devices?
 - Resign via developer certificate
 - But apps will need to be reactivated

- You've got to be root?!
 - Check out Stefan Esser's talk on "iOS 678 Security - A Study In F
- Physical Access
 - DROPOUTJEEP (think NSA, GCHQ)
 - Lost Devices / Stolen Devices
- Remote Attacks

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



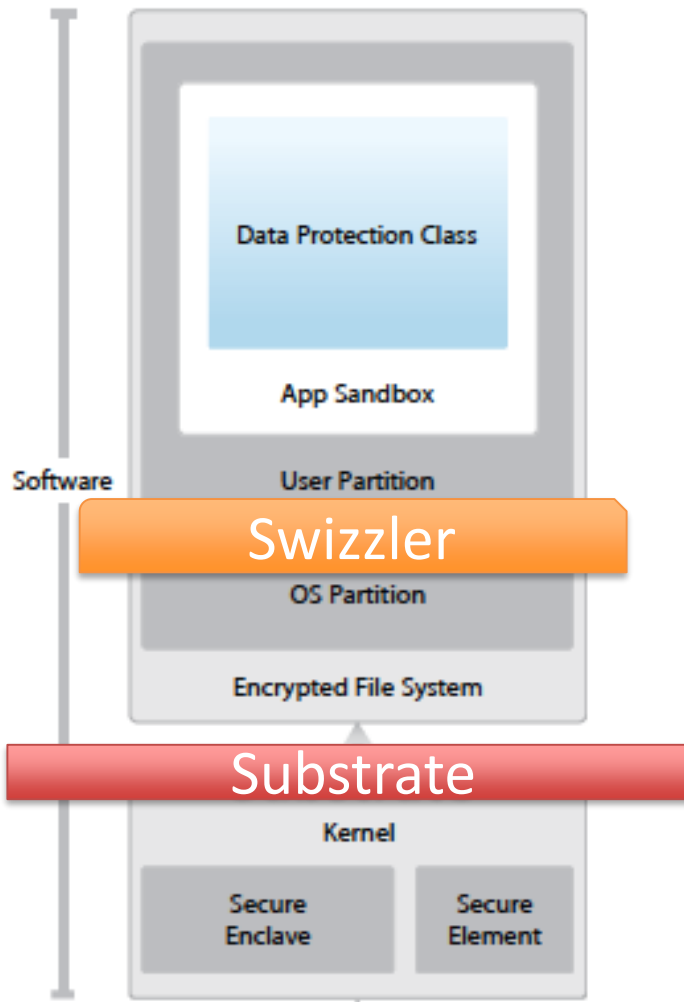
¹ <http://www.tripwire.com/state-of-security/vulnerability-management/creating-iphone-rootkits-and-like-the-nsas-dropout-jeep/>

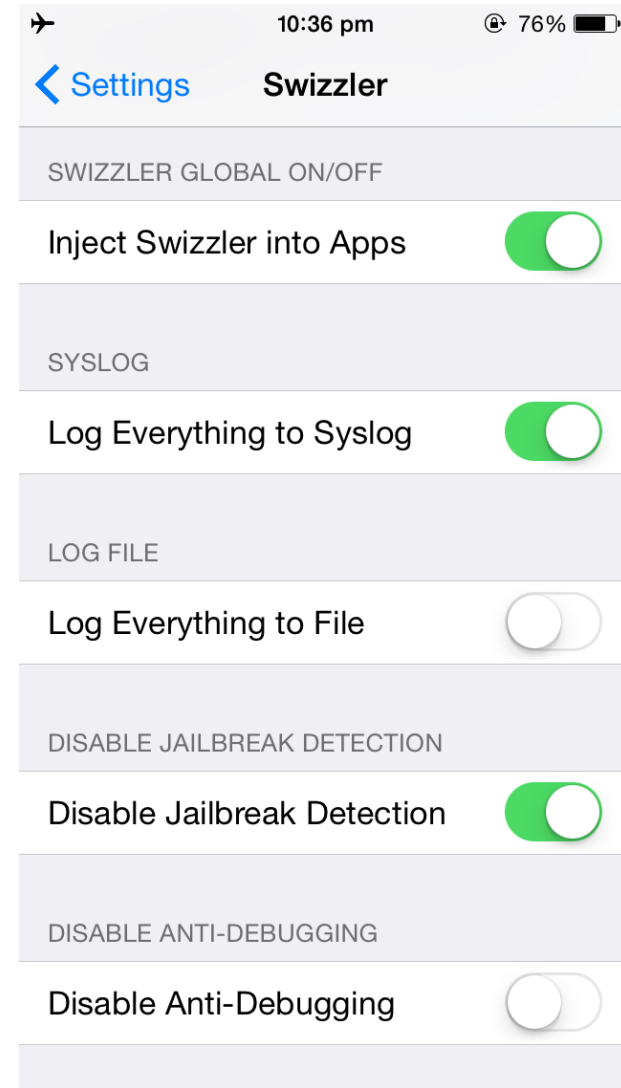
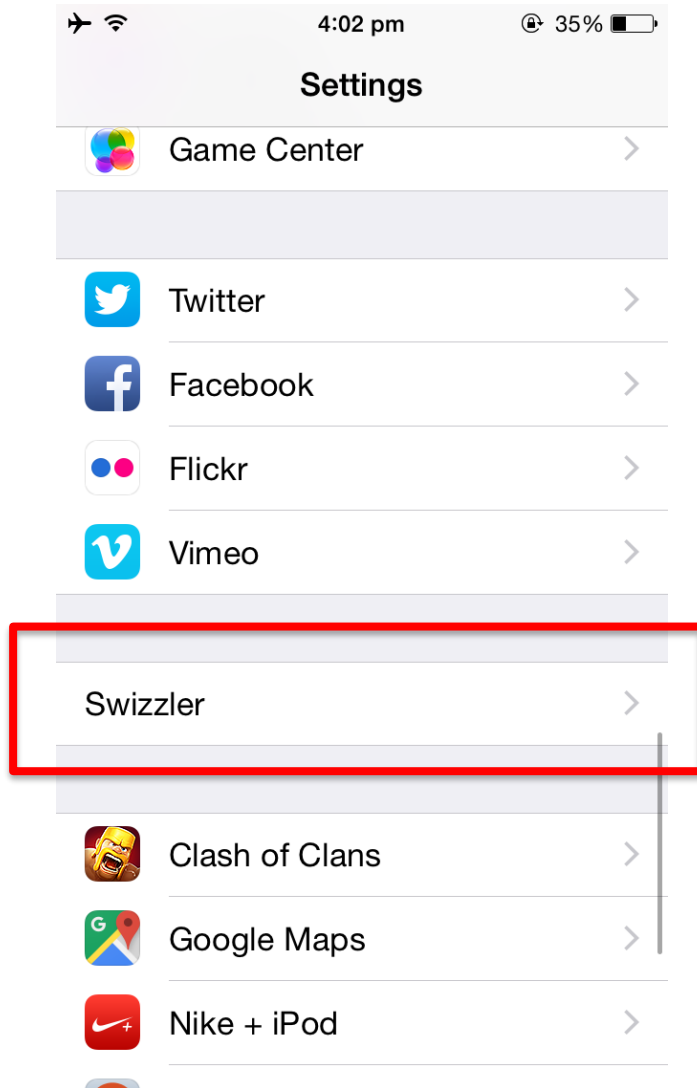
² <https://blog.fortinet.com/post/ios-malware-does-exist>

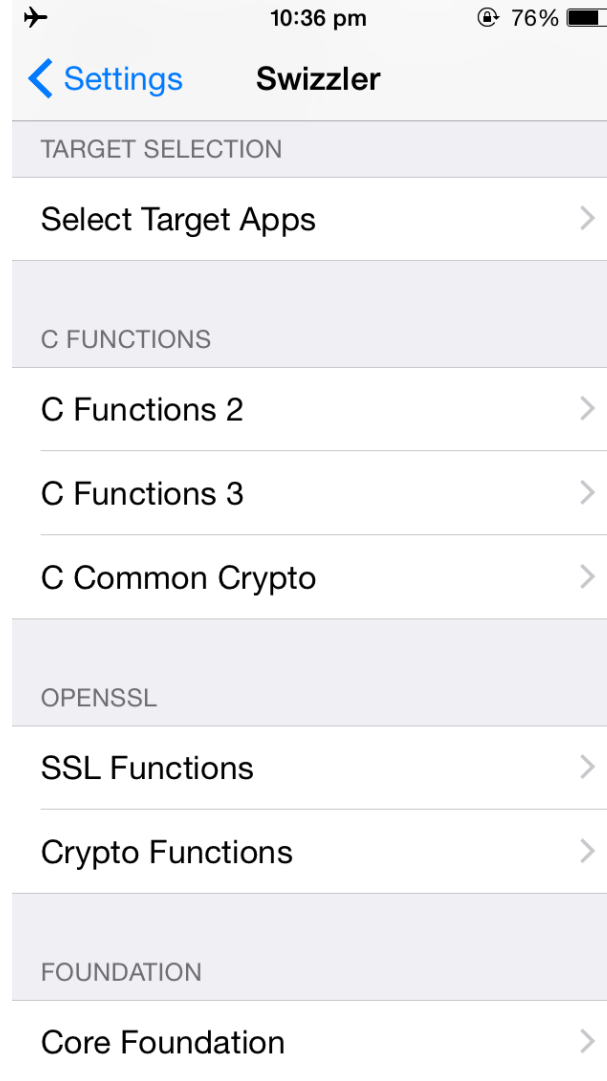
- Not normal pen testing...
 - Not just setting proxy and using Burp
- I'm not attacking the application
- Changing the environment in which the application runs.
- Not new. API Hooking and DLL Injections on Windows.
LD_PRELOAD on Linux. I'm just doing it on iOS.

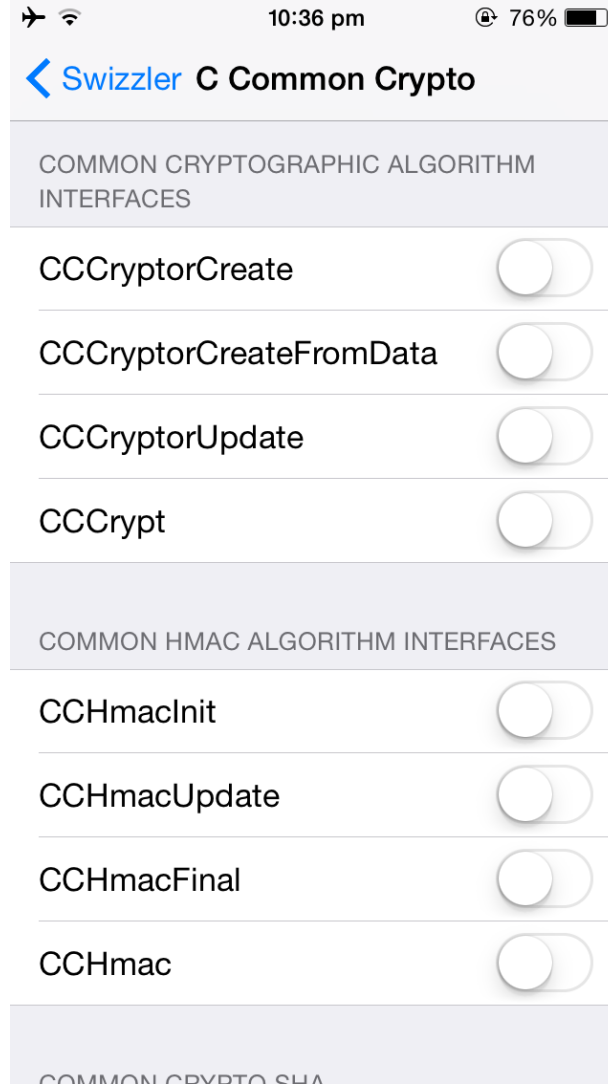
- How do I change the Environment?
- Built an App... More precisely a Dynamic Library (aka tweak)
 - DYLD_INSERT_LIBRARIES=Swizzler
- Loads itself before an application starts
 - Control all functionalities of an application

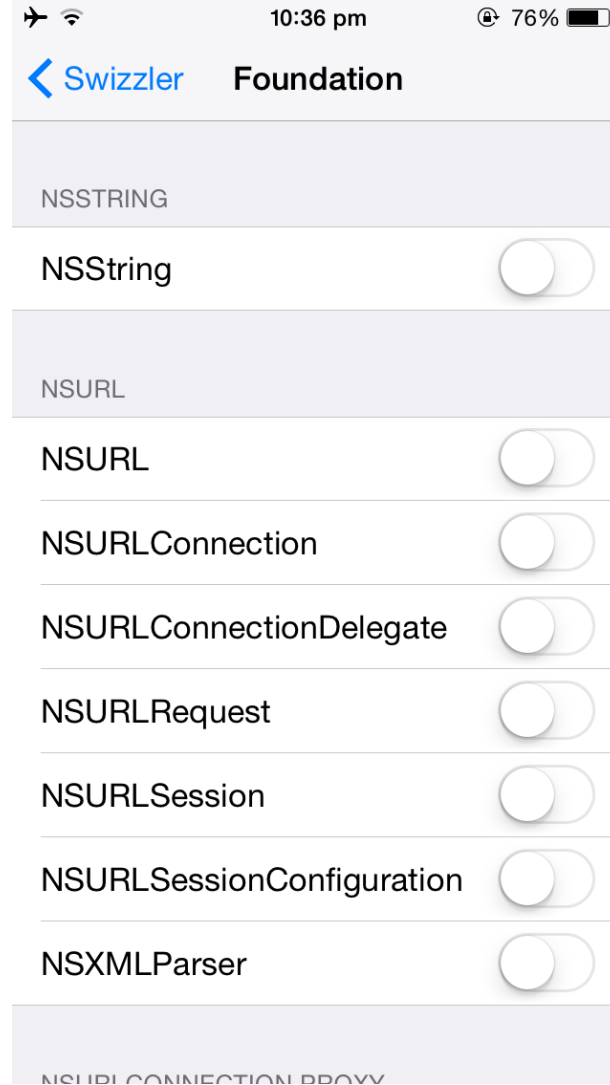
iOS Security Architecture



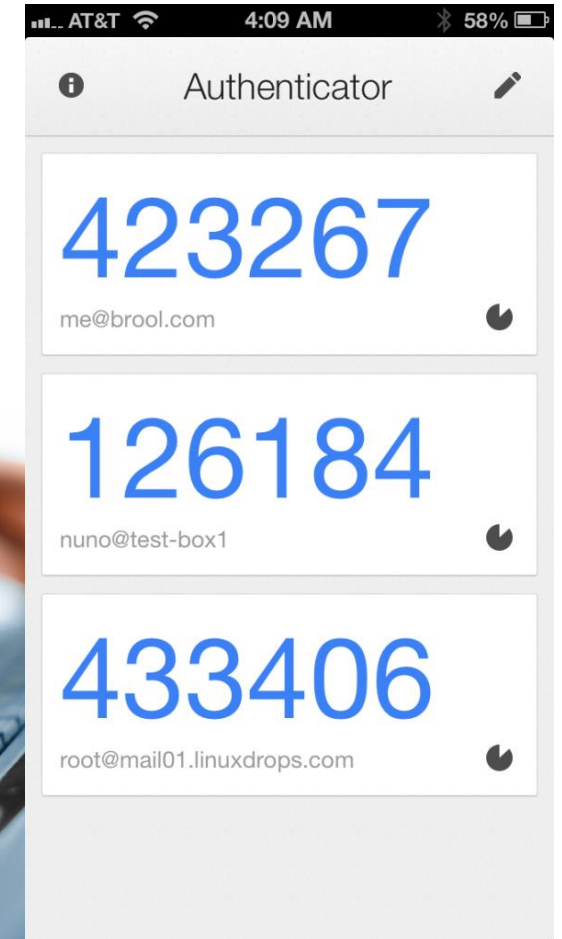
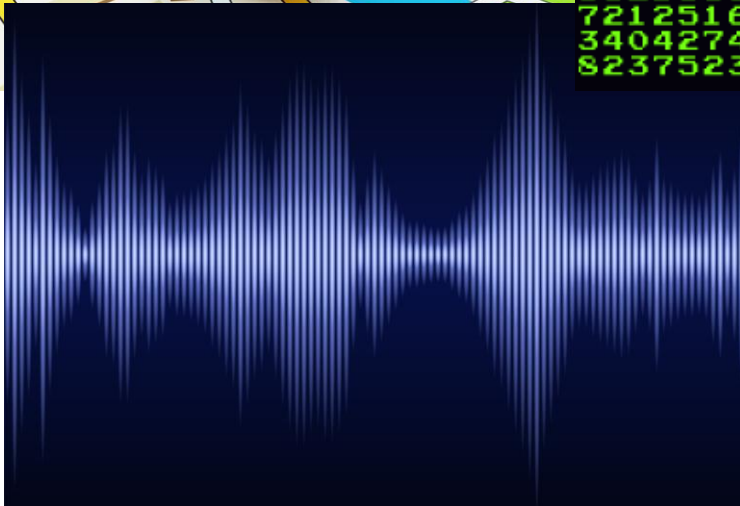
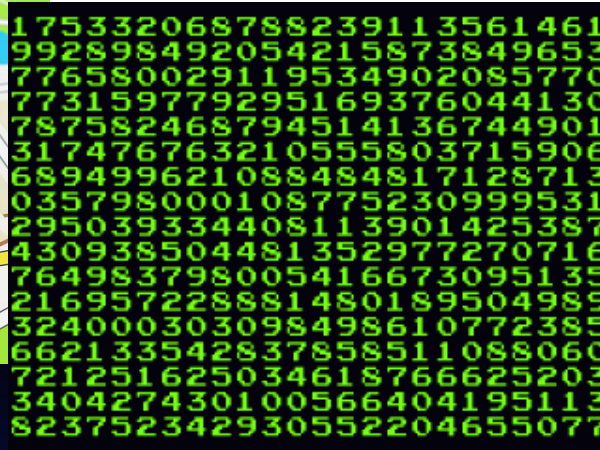
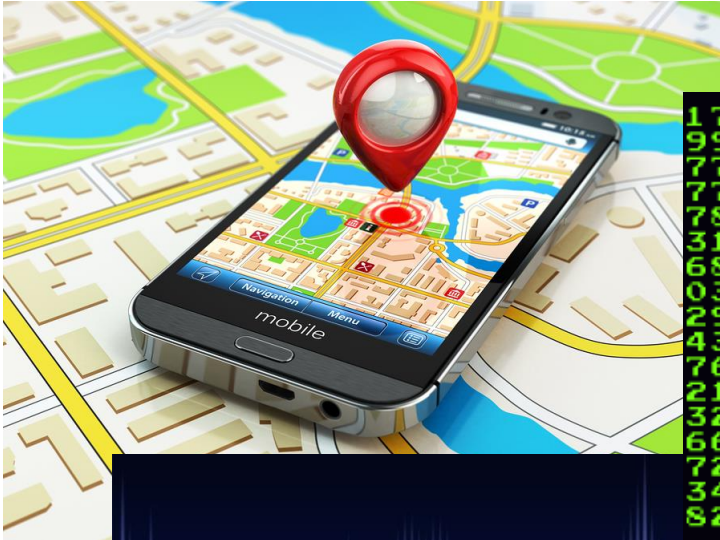








What else can you control?



Jailbreak

**Jailbreak
Detection**

Container

App DLP

Device PIN

**Container
Password**

App







Local Attacks

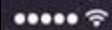


Jailbreak Detection

App
Wipe / Lock

Container
Password

Container
Encryption



9:41 am

100%



Good Work



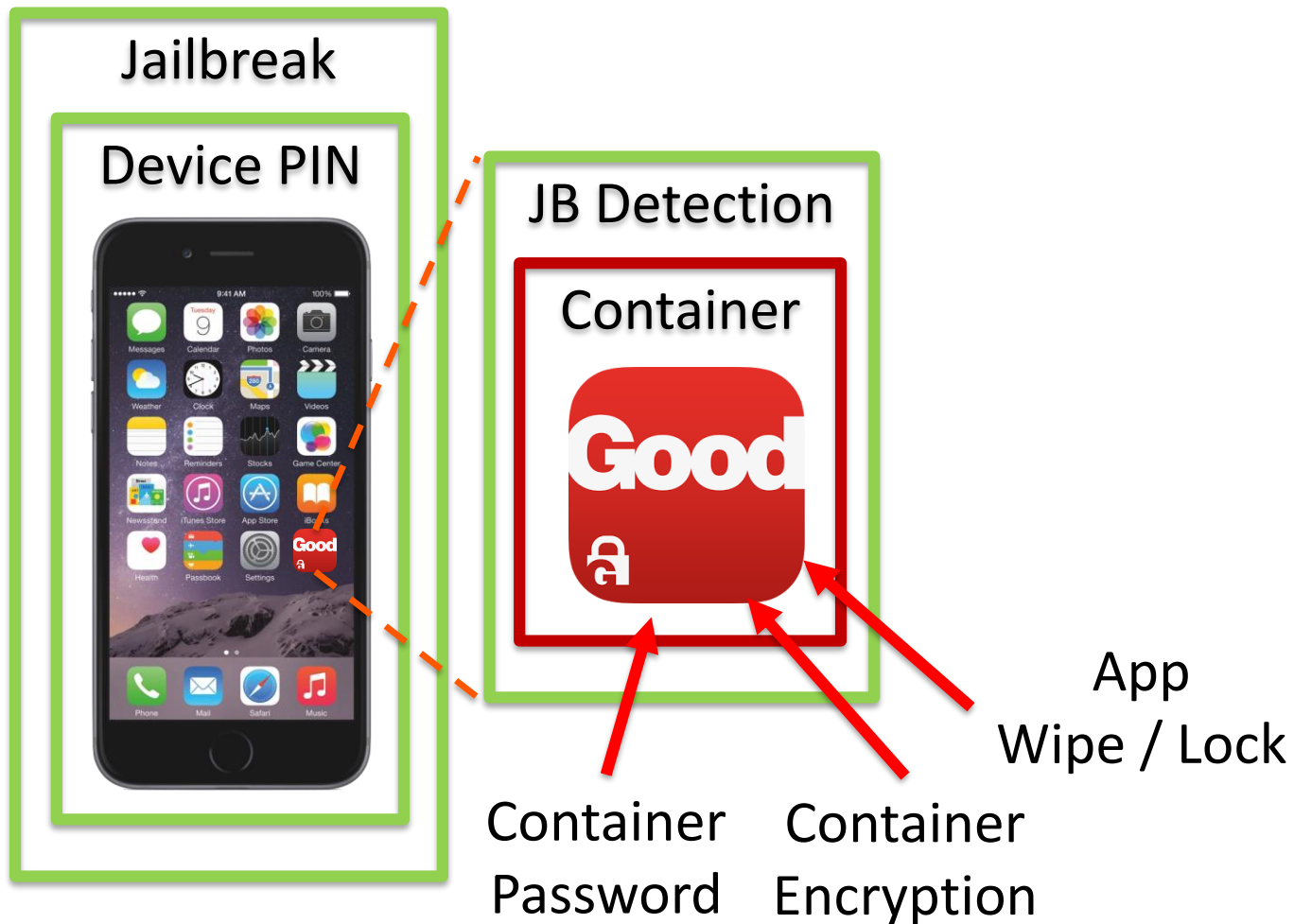
iFile



Settings



Safari



- ✓ 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;  
}
```



```
if ([[UIApplication sharedApplication] canOpenURL:[NSURL
URLWithString:@"cydia://package/com.example.package"]])
{
    return JAILBROKEN;
}

+ (id)URLWithString:(NSString *)URLString{

    NSRange range = [URLString rangeOfString:@"cydia"
options:NSRegularExpressionSearch|NSCaseInsensitiveSearch];

    if (range.location != NSNotFound) { return nil; }

    return %orig;
}
```

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

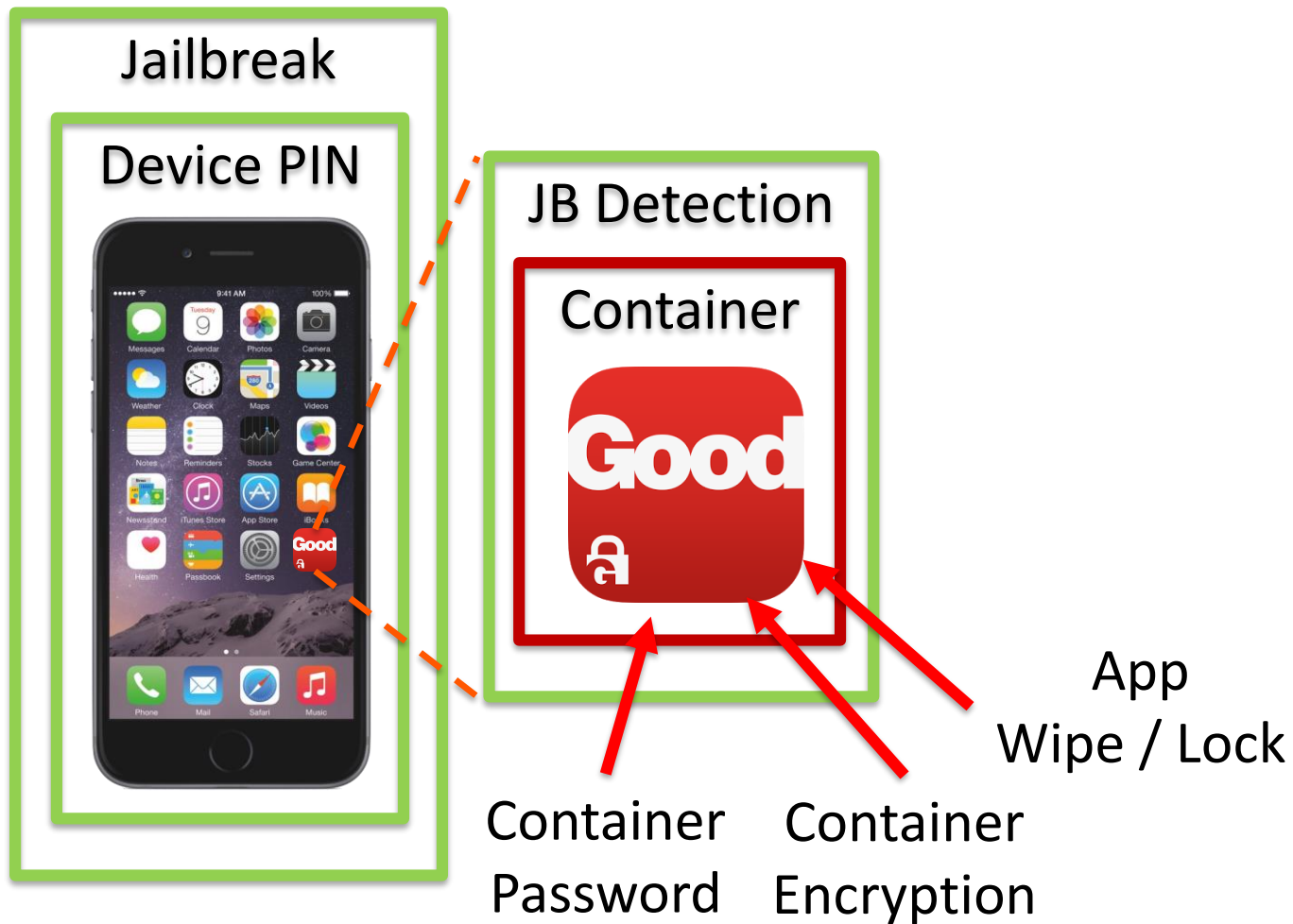
Jailbreak / Policy Implementation

```
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;
```


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



Password Bruteforce



9:41 am

100% 



Good Work



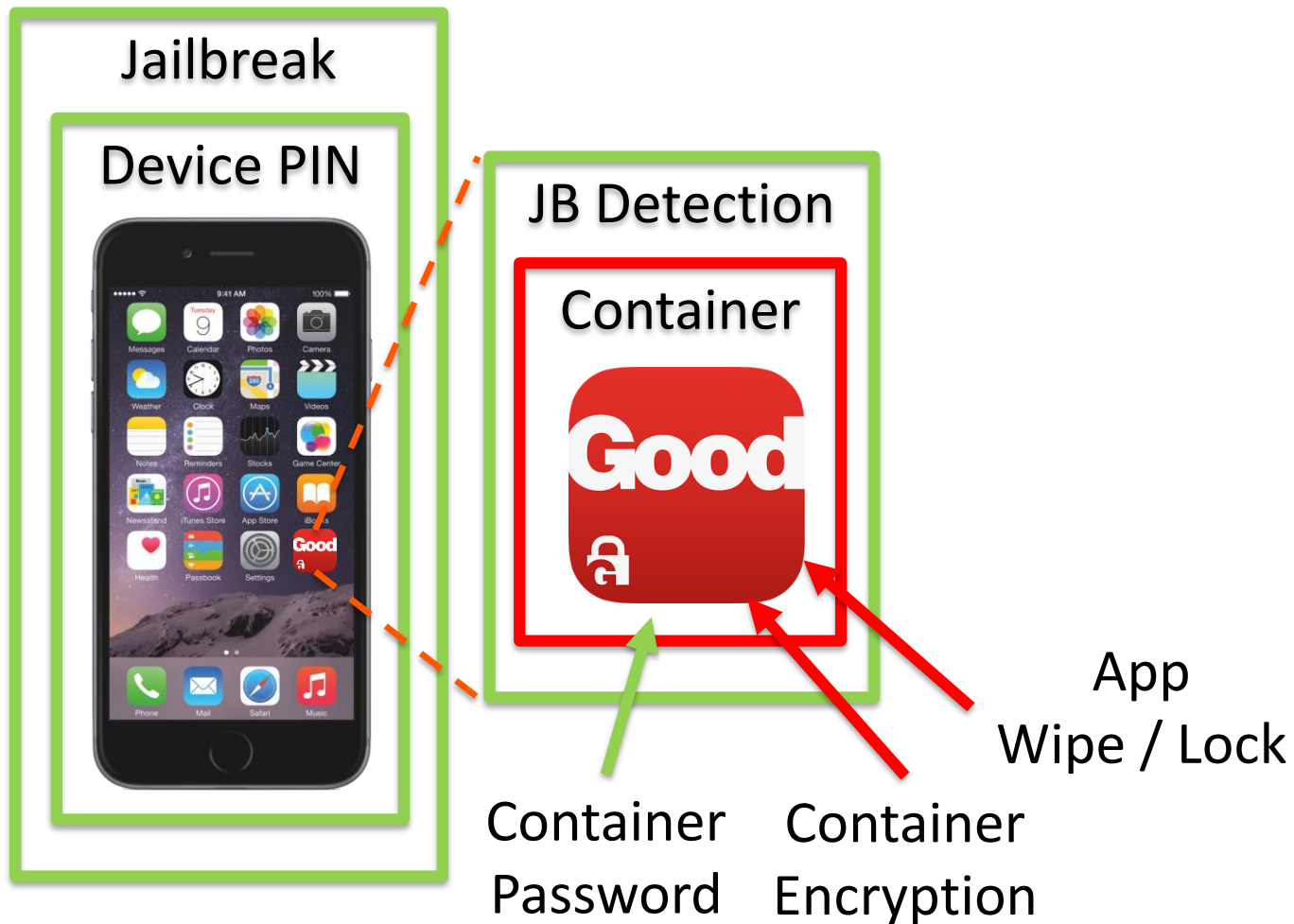
iFile



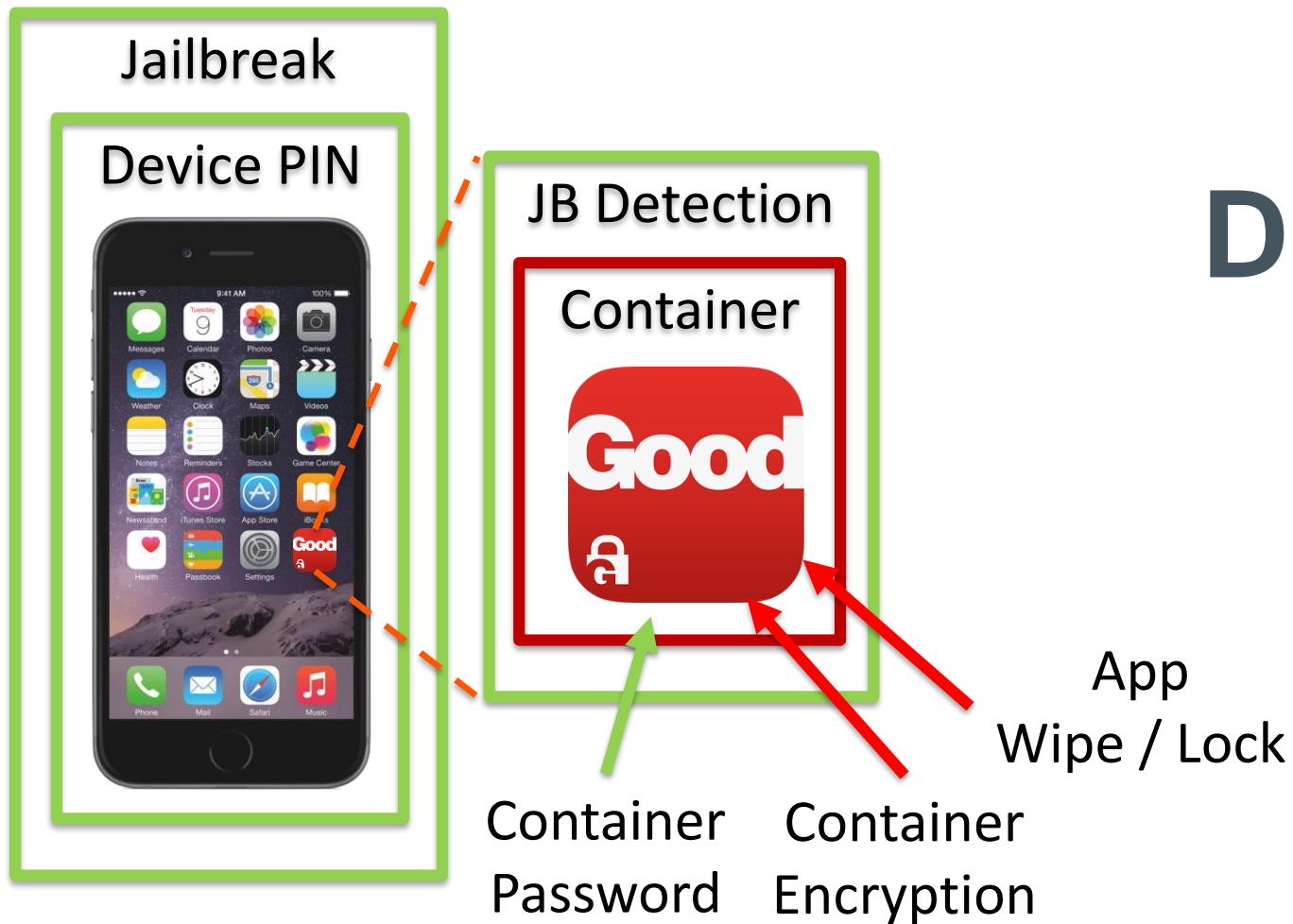
Settings



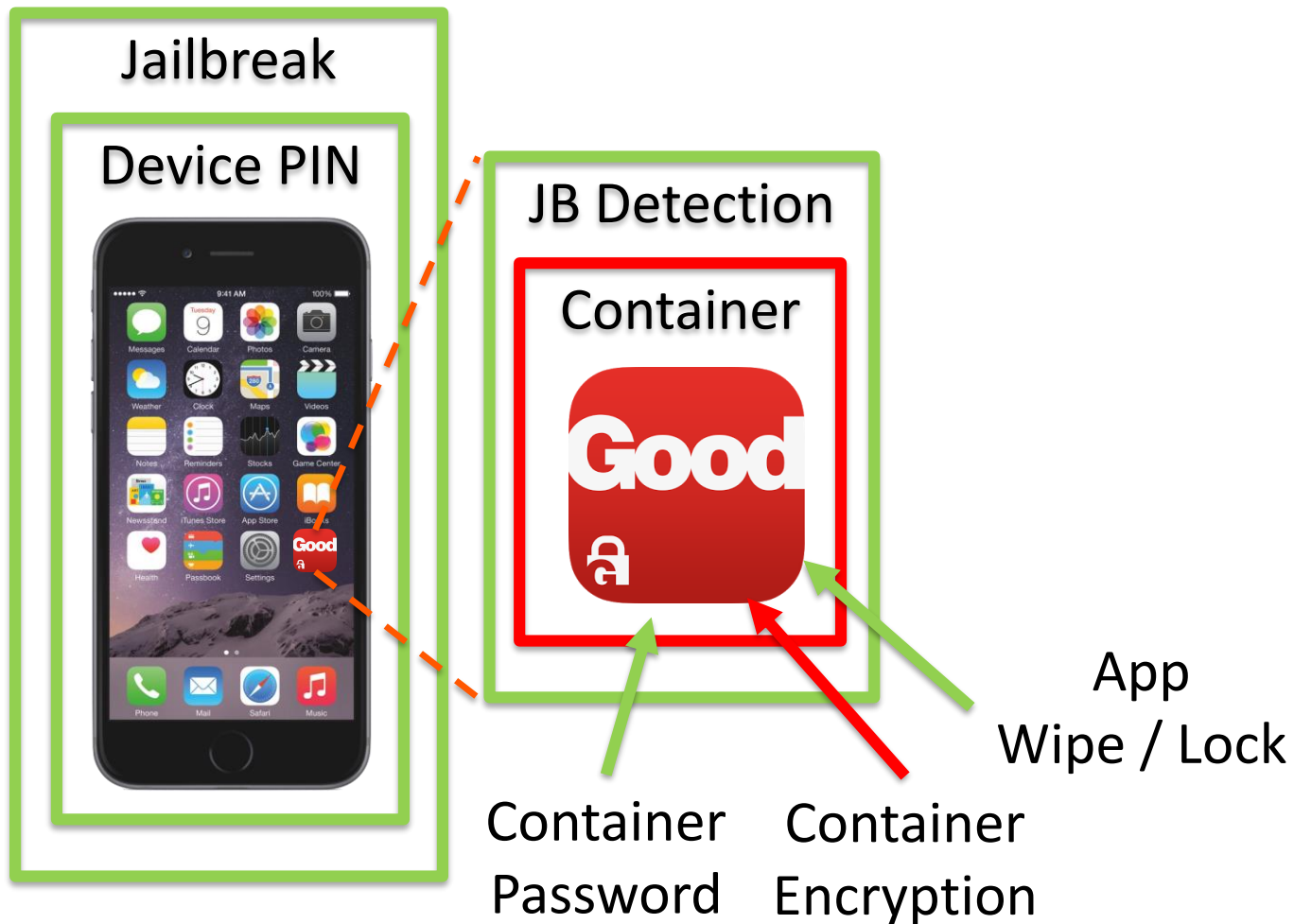
Safari



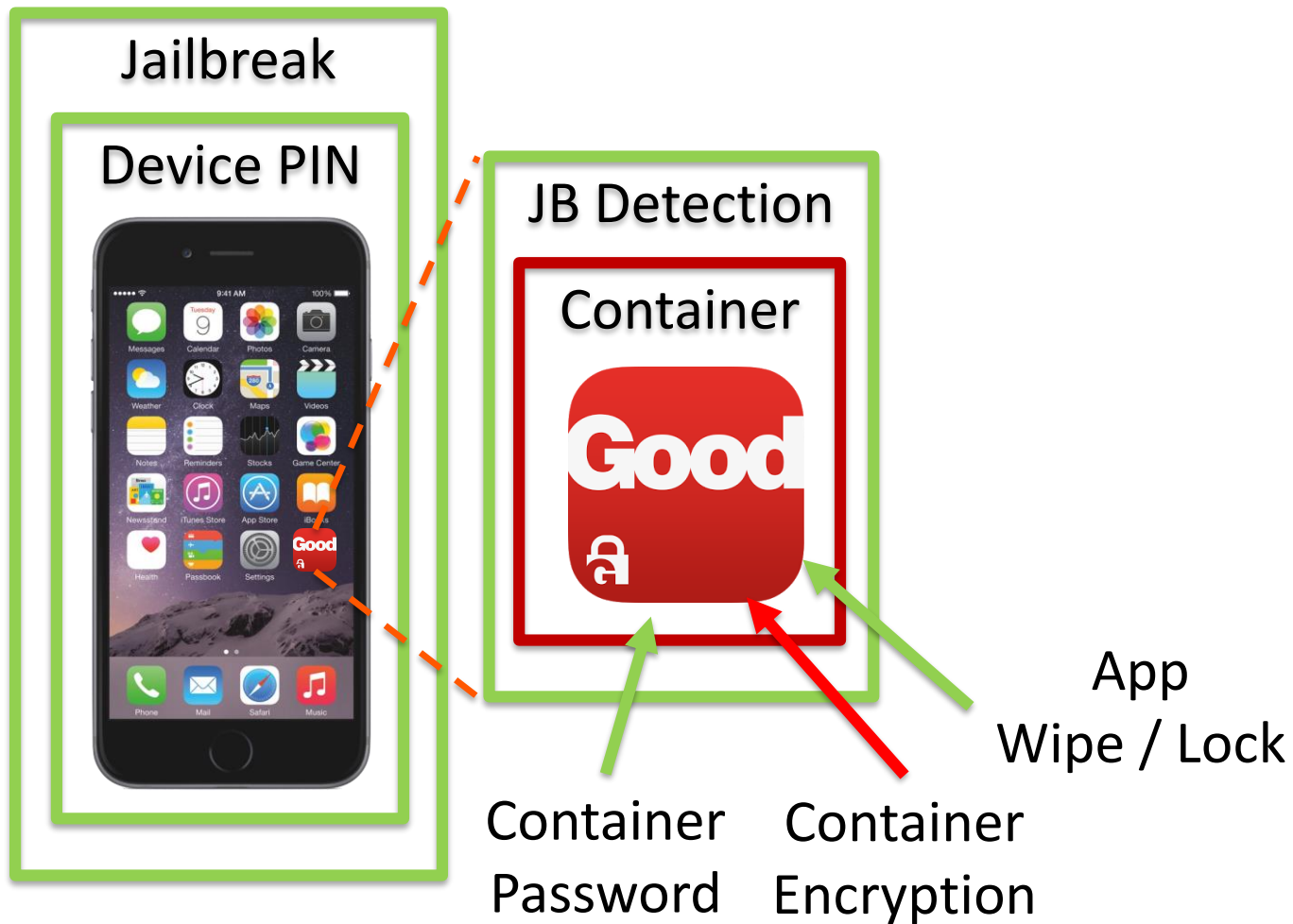
- ✓ Device PIN
- ✓ Jailbreak
- ✓ Jailbreak Detection
- ✓ Container Password
- ✗ Container Encryption
- ✗ App Wipe / Lock



Disable App Lock & Device Wipe



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



Containerization

Host Directory Quick Connection Disconnect Scroll Back as Window Clear Scroll Buffer Show Stream Capture Paste Download Upload Send Textfile Start RE

127.0.0.1:2222 (Standard.zoc)

127.0.0.1:2222

About these Buttons Send "exit<enter>" Call Host from Host Directory Run Sample Script

Secure Shell Xterm Zmodem [x] ZOC1506_127.0.0.1_22220109.log

Settings Swizzler

Disable Anti-Debugging

TARGET SELECTION

Select Target Apps

C FUNCTIONS

C Functions 2

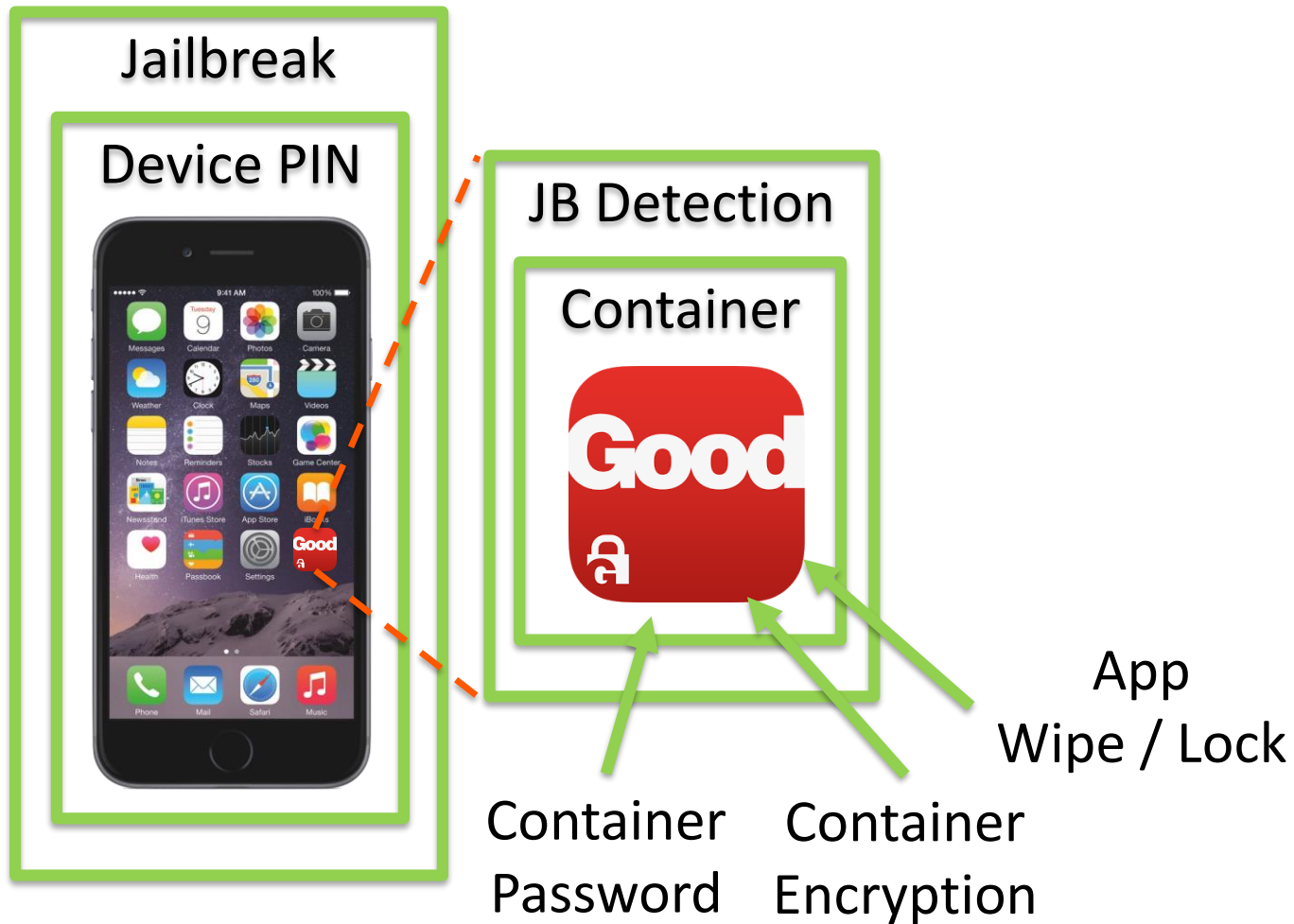
C Functions 3

C Common Crypto

OPENSSL

OpenSSL Functions

CORE FOUNDATION

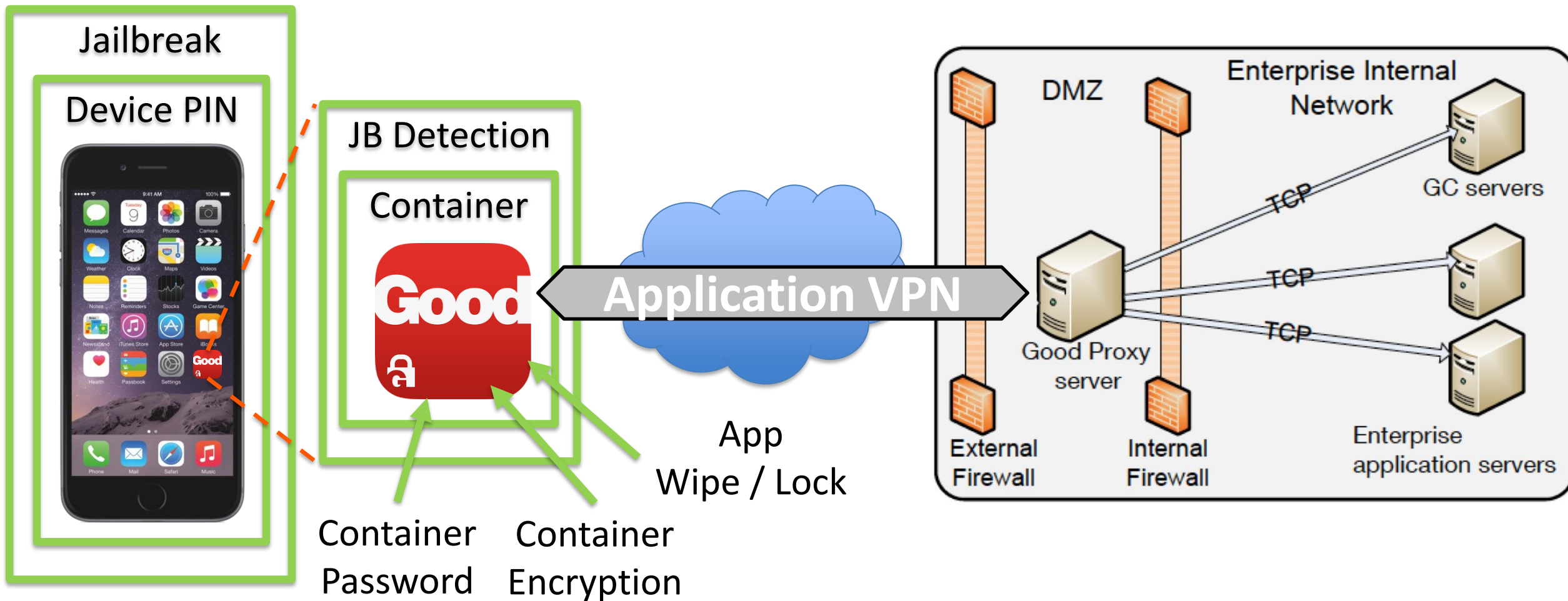


- ✓ Device PIN
- ✓ Jailbreak
- ✓ Jailbreak Detection
- ✓ Container Password
- ✓ Container Encryption
- ✓ App Wipe / Lock

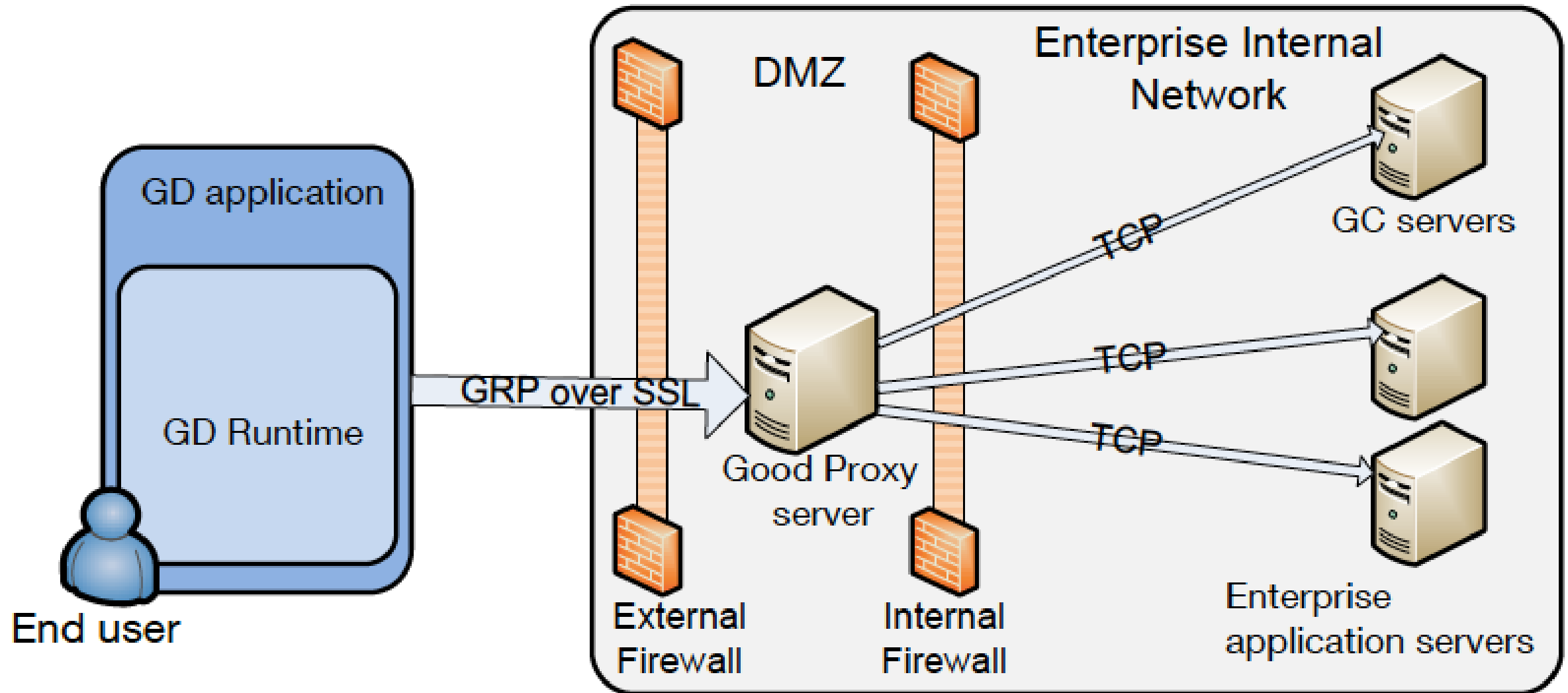
vincent: /Users/vincent/Desktop/iostesting/Good

>>>

Network Attacks

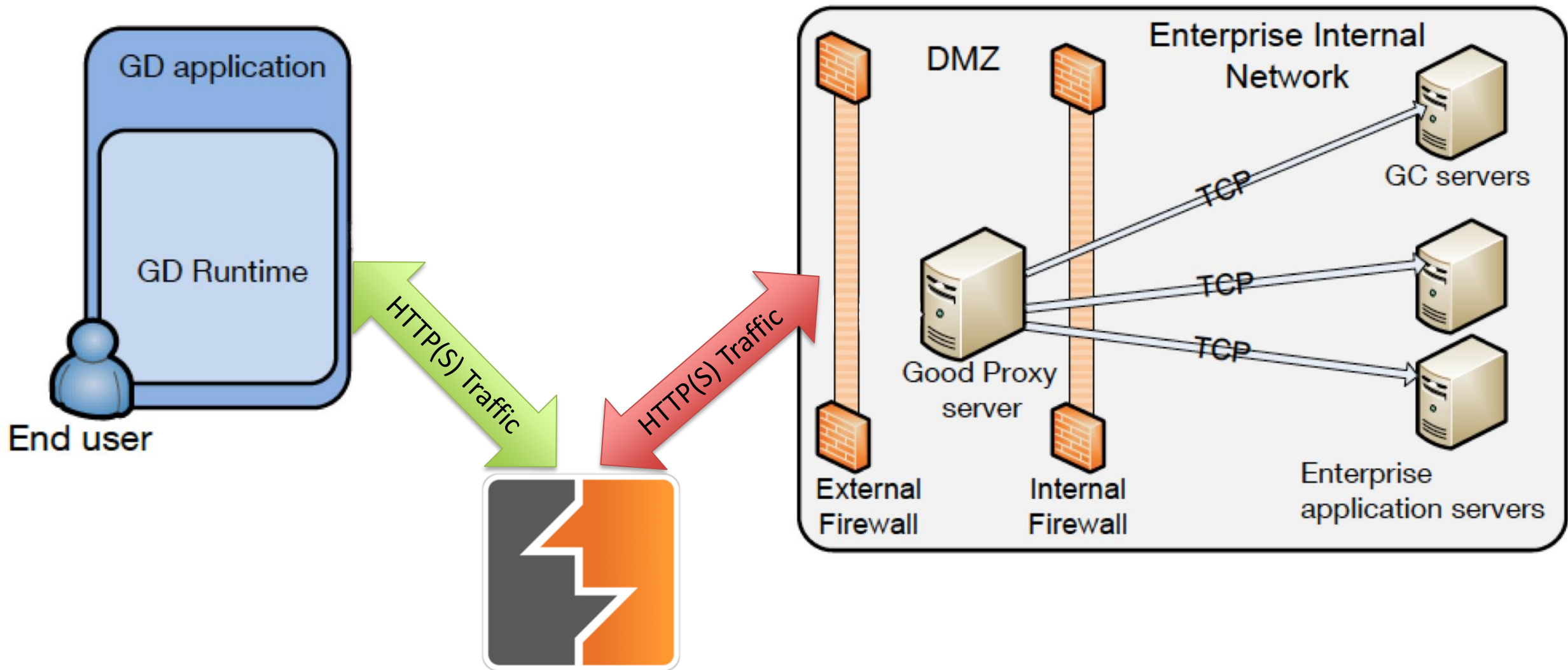


Application VPN

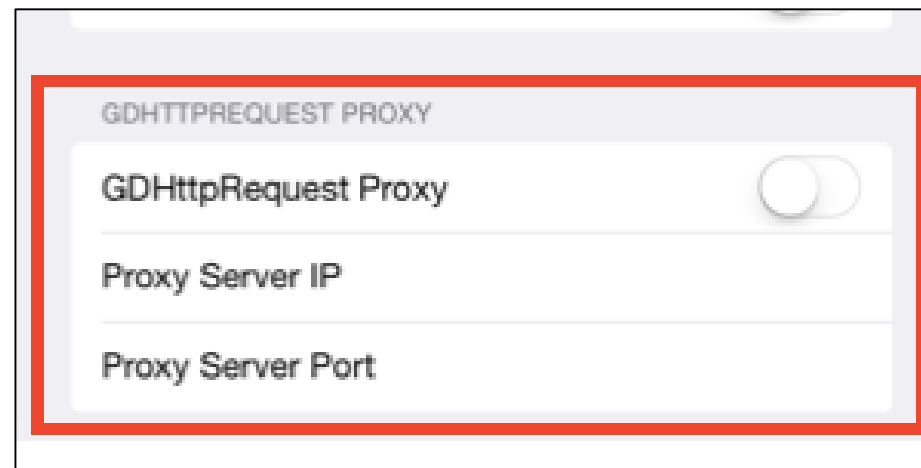


Two methods of communication with the enterprise application server,

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

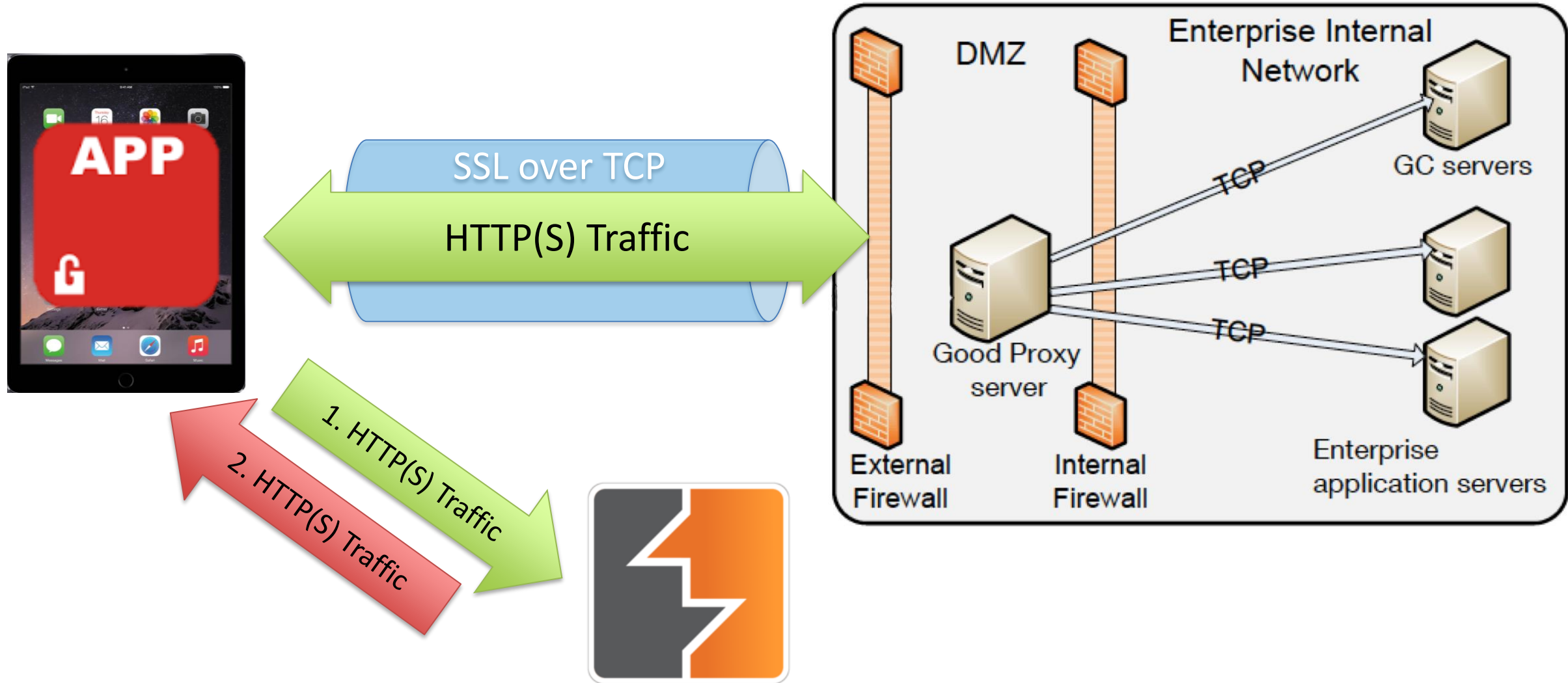


- Part of the GD SDK
 - `#import <GDNETiOS.h>`
- Easy to enable proxy
 - `[GDHttpRequest enableHttpProxy:ip withPort:port];`
 - `[GDHttpRequest disablePeerVerification];`



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

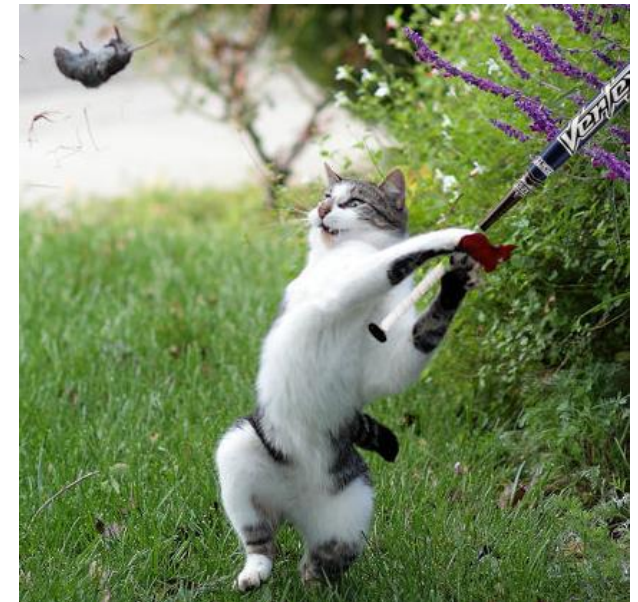
Hooking the Network



Does everything suck?

- Local device access is important, but remote attacks possible.
 - <https://www.youtube.com/watch?v=STIHO2XOOiM>
- 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