Now iOS has been compromised to Suspected Ransomware

- How we hunt and defeat profile ransomware in the wild towards iOS

Ju Zhu, Moony Li
Twitter: @ju__zhu, @Flyic
TrendMicro, Security Researcher
ju_zhu@trendmicro.com.cn, moony_li@trendmicro.com

1. Abstract

As we all know, WannayCry has caused terrible disturbance to Windows PC all in the world utilizing remote 0day SMB vulnerability. And Apple platform especially the iOS operating system is imaged to be safe enough to be immune to such kinds of security threat because of its declared system security feature by design and restrict Apple Store security policy.

However, maybe the first time, we have found a suspected ransomware remote attack towards to iOS and OSX platform distributed in the wild.

In this paper, we would like to tell you the whole story of how we hunt for the remote iOS ransomware attack based on profile installation and defeat it in cradle.

We would analysis how the iOS ransomware hijacks your phone screen or even causes system crash or hang using profile installation as we call “death profile” in technical detail and introduce both static and dynamic solution to detect and remediate such threat.

Thus, we would propose one new remote attack interface on iOS system research.

2. Outline

2.1 Story of iOS Profile Ransomware

2.1.1 Version 1.0 Appears (2016/11)

iOS profile ransomware version 1.0 (iXintpwn/YJSNPI malware) distributed in Twitter as we name the buggy profile mechanism as “death profile”.

Figure 1. Domain in Twitter

But in fact, this is a fake jailbreak website. The author uses this to trick victims into downloading and installing it.

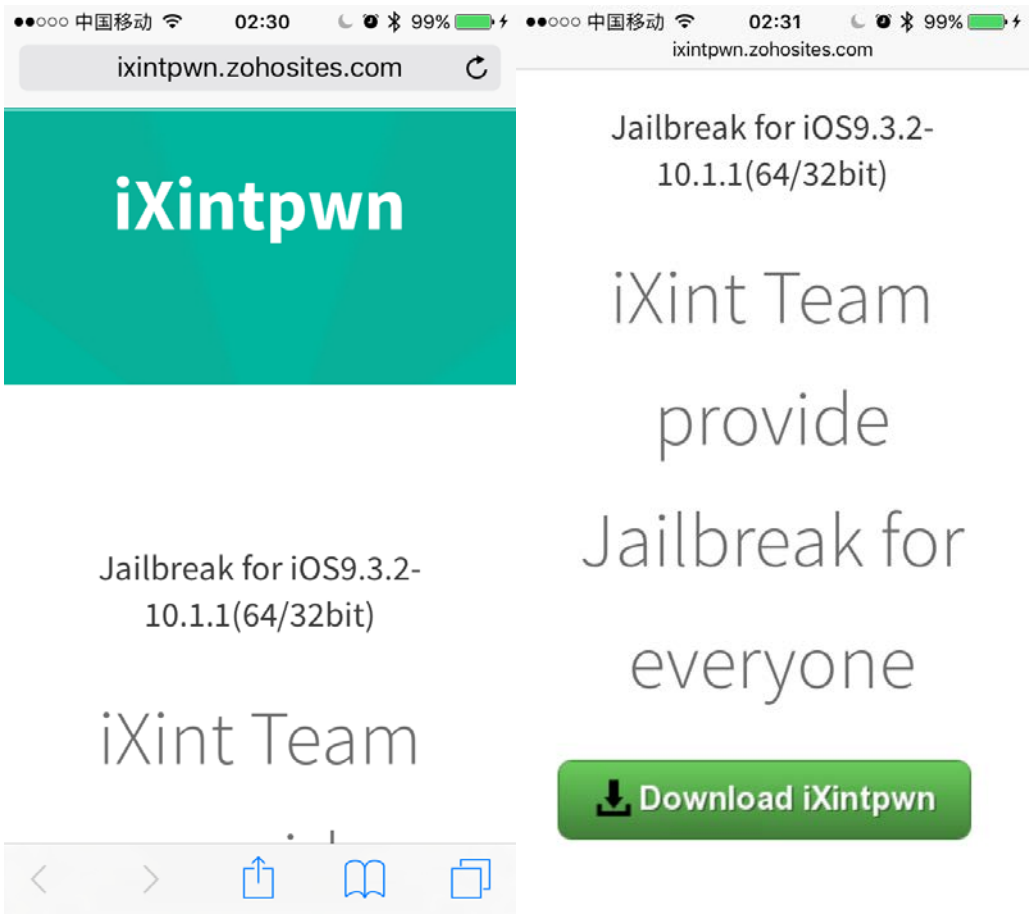

Figure 2. hxxp://ixintpwn.zohosites.com

2.1.2 The Source Code Released (Next Day)

The next day, the author just released its source code on GitHub. The URL is "hxxps://github.com/83anija/iosiconshock.js".

Figure 3. hxxps://github.com/83anija/iosiconshock.js

He also provides a web-tool for generating a malicious profile of any number of ICONS. Any bad guy can build his own malicious attack profile by this tool.

Figure 4. hxxps://83anija.github.io/iosiconshock.js/iXintpwnTester.htm

2.1.3 Version 2.0 Released (2016/12)

iOS profile ransomware version 2.0 distributed in Twitter. The author has made some improvements. Its purpose is to directly support Mobile Safari and open “multiple installations” option.

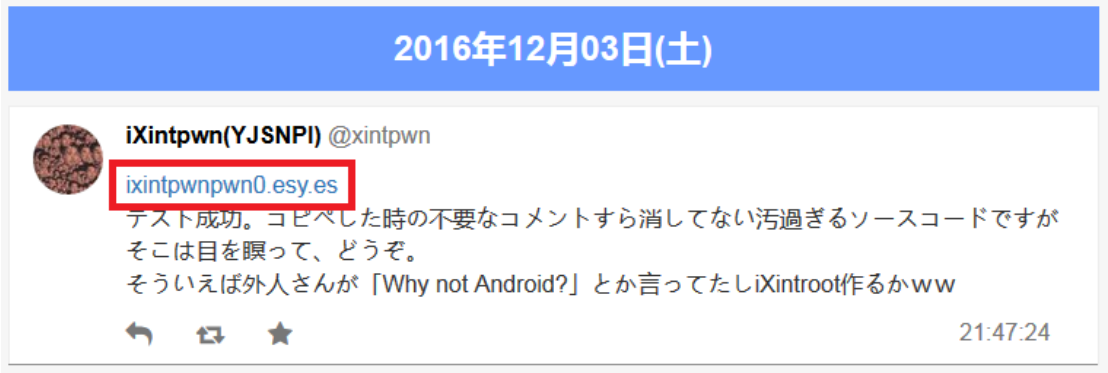

Figure 5. New Domain in Twitter

Fortunately, it is only a demo. But it has turned on “multiple installations”.

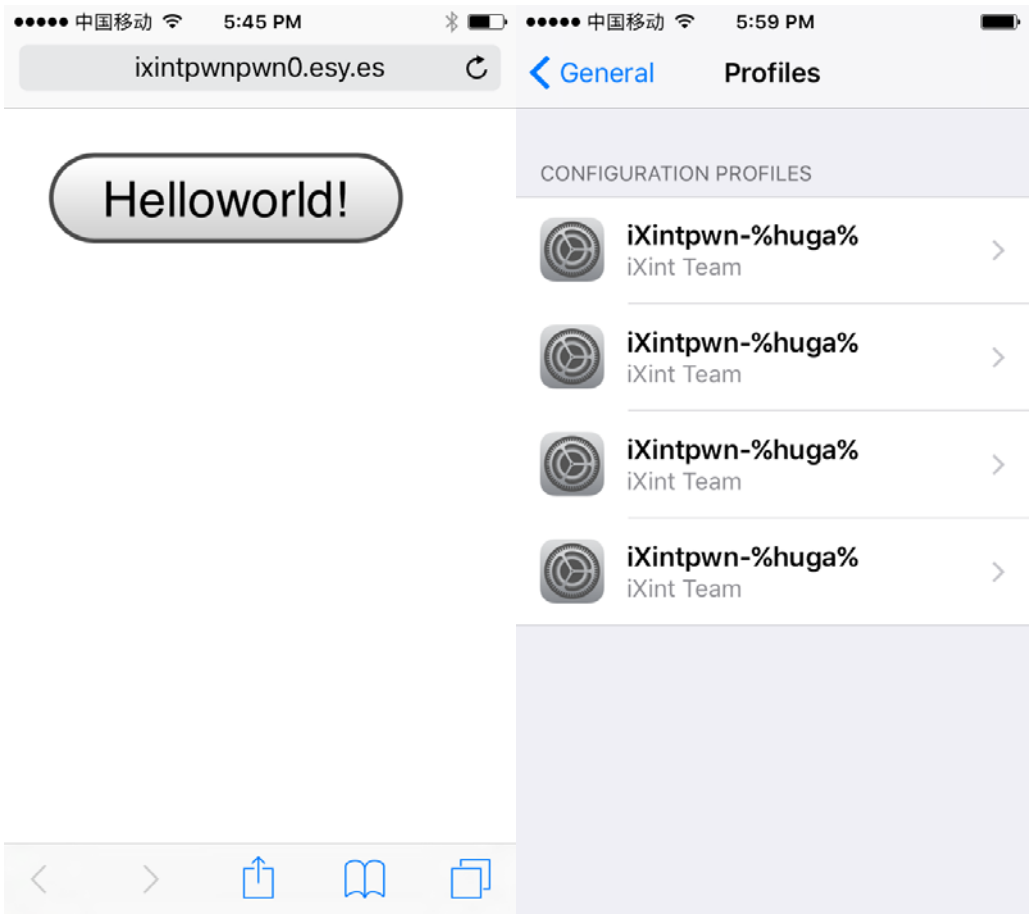

Figure 6. hxxp://ixintpwnpwn0.esy.es and Multiple Installations

2.1.4 The Author was Arrested (2017/1 ~ 2017/4)

Tweet accounts that are seen as boys are tweets when they distribute Ransomware.

Kanagawa Prefectural Police discovered Twitter accounts for male students through Cyber Patrol. Then they searched for the boy's home and seized PC and others.

Tweet account tweets that are seen as boys stop tweeting.

2.1.5 Technical Analyzing and Found Vulnerability

We found that after multiple installations, a weird icon appears. SpringBoard’s icon has overflowed.

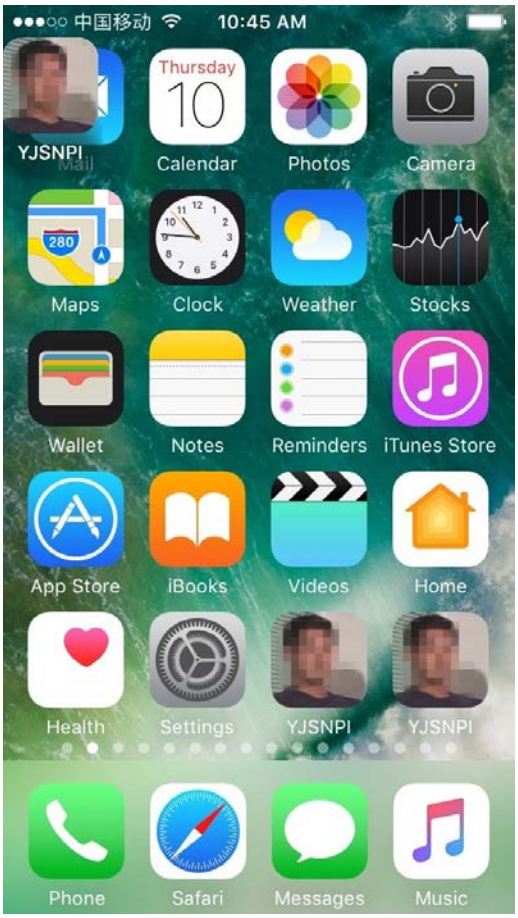

Figure 7. Weird Icon

When we click on it, SpringBoard will crash because of WatchDog's timeout monitoring.

Generally speaking, we can uninstall a malicious profile through "Settings->General" or "Apple Configurator 2->Remove->Profiles".

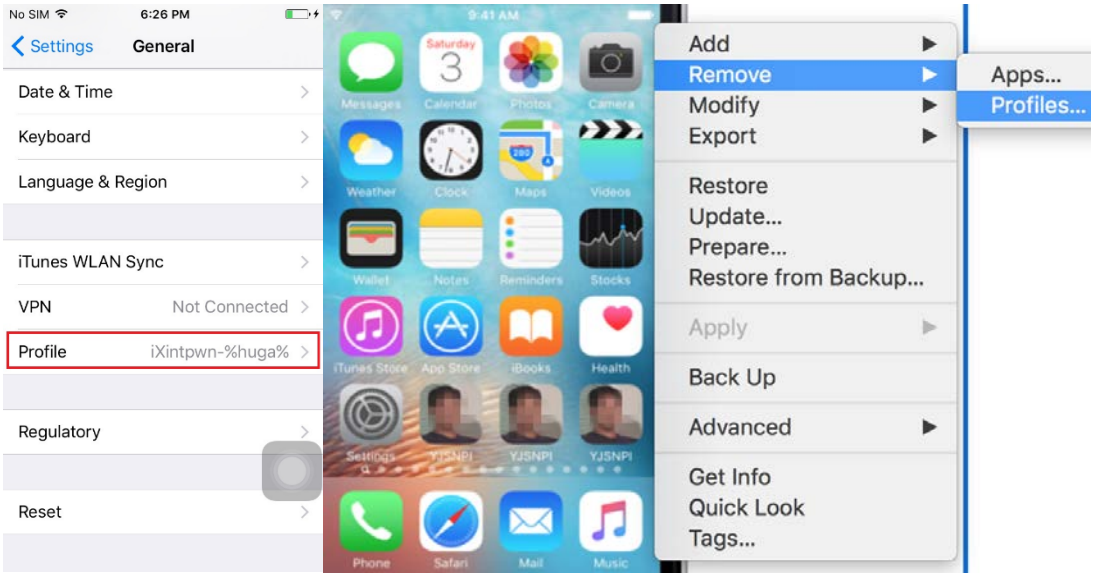

Figure 8. How to Uninstall a Malicious Profile

But after the crash, these malicious icons will reside in your system forever. That means we can‘t use any tools to remove them, unless you restore factory Settings.

In addition, we found that they were using tricks to make users fearful of being unable to delete them.

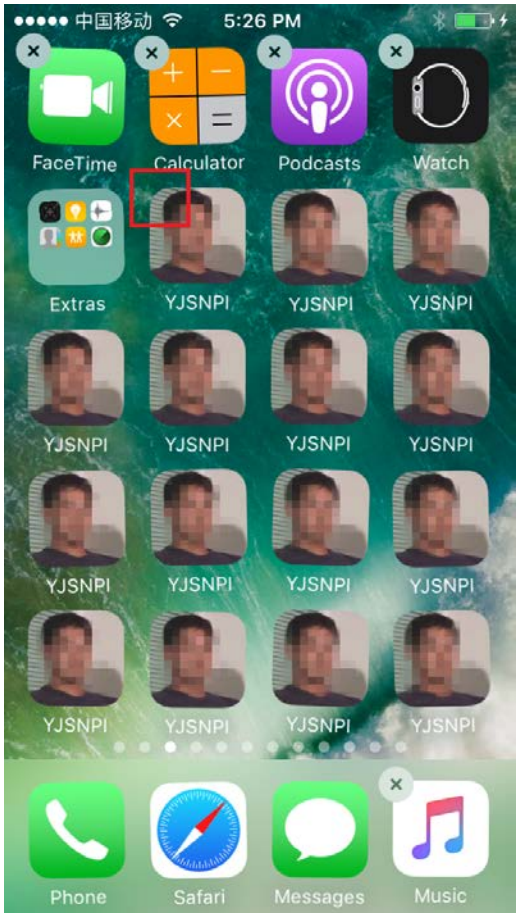

Figure 9. Can Not Remove

2.1.6 Sale (2017/9)

NHK reported: this tool was once sold in Merukari (market) and it has been successfully traded.

2.1.7 The Variant Malware Appears (2017/10)

In fact, Apple has a mitigation mechanism for installing unknown profile. Apple would use highlighted “Not Signed” to warning user.

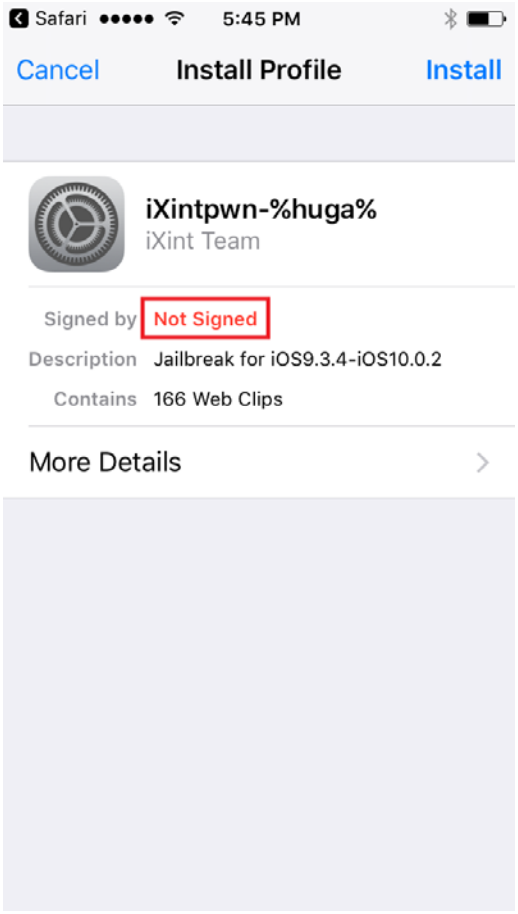

Figure 10. No Signature

But bad guy can still use a signed profile to conduct similar attack.

What is more interesting, about one month later, we have found real “death profile” attack in one third party Apple Store (hxxp://m.3454.com) in China region in our threat hunting system based on the “death profile” pattern learn from the iXintpwn malware.

Figure 11. Signed

If users access this site using iDevice, it will lure users to install a signed profile file of mobileconfig type. All the web clip Icon are un-removable and link to popular applications (e.g. 百度，爱淘宝) in this store which are repacked from official ones.

Figure 12. Un-Removable and Link to Popular Applications

2.1.8 Other Related Findings

This kind of malicious profile is actually a configuration profile, there is also a profile called a provision profile.

We found that iTunes had a risk. It can automatically synchronizes all provision profiles in a fixed directory (/Users/{LoginUser}/Library/MobileDevice/Provisioning Profiles) to iDevice without checking the owner of this directory.

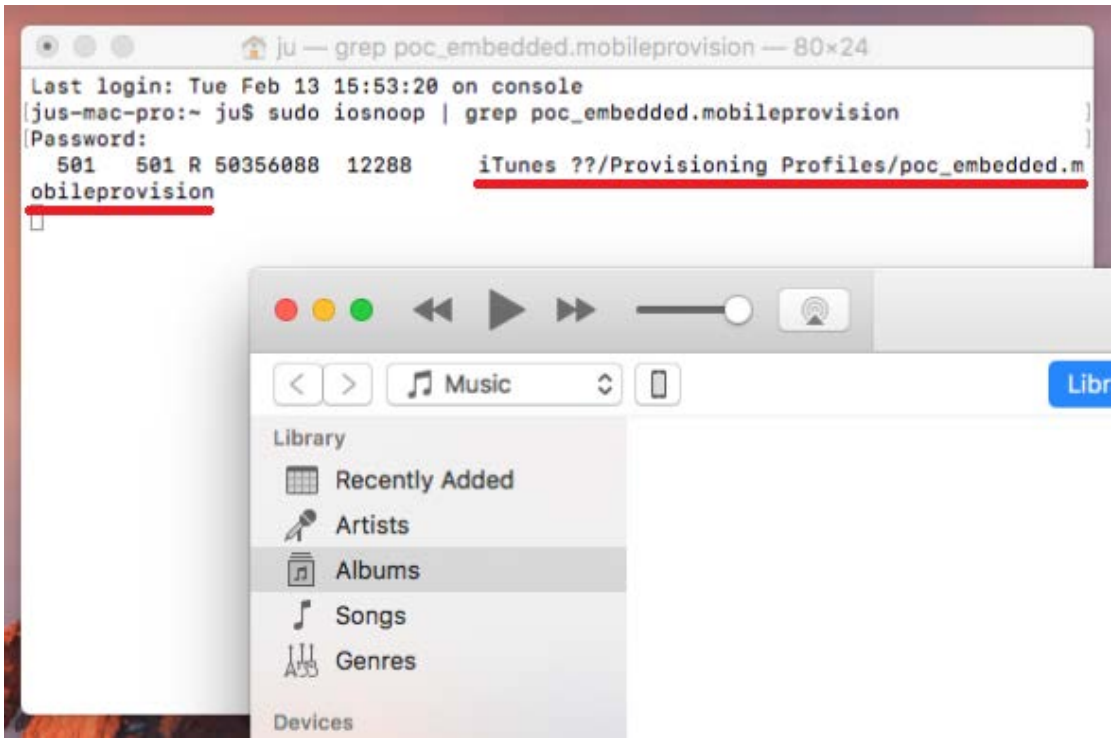

Figure 13. Auto-Sync

Maybe this will be part of an attack chain.

Moreover, we also found that a vulnerability can bypass “establish trust” mitigation mechanism through by attacking the installation of the provision profile.

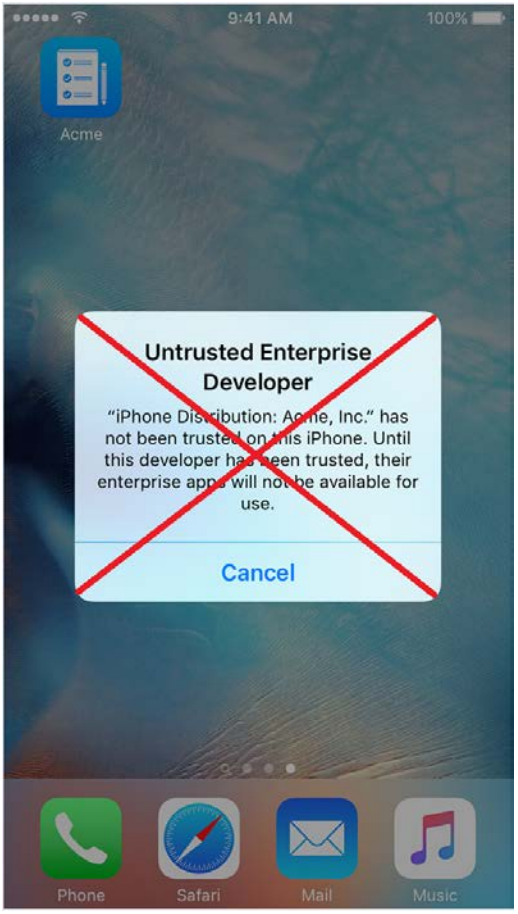

Figure 14. Bypass "Establish Trust"

2.1.9 Conclusion

As a teenager hacker, the malware author has released at least two versions of iOS ransomware by Twitter distribution before he is arrested by Japanese Kanagawa Prefectural Police.

The threat case was reported to our iOS Advanced Threat Research Team as threat response. And we realized that this would be the first ransomware towards iOS platform even though it is much like of POC in test at the first begin phase.

In brief, the attack is much like of demo test when we have found it. The user screen would be occupied by large quantities of Icons (actual WebClip) if any iOS user click the malicious web server and lured to install a malicious profile.

The WebClip ICON (person head with name "YJSNPI") could not be removed manually so as to frighten the iOS user and also cause occasional system process hang or even machine reboot.

As we tested, all iOS version including the latest iOS 11 are compromised to this kind of ransomware at that time.

This kind of ransomware which is based on configuration install mechanism is novel because:

1. The attack could be triggered remotely (by Safari browsing or mail reading).
2. It exposed another attack interface which has ignored Apple application store security policy and installation check.

2.2 iOS Profile Ransomware Technical Analysis

2.2.1 What is Profile

As apple document say, a configuration profile is an XML file that allows you to distribute configuration information. If you need to configure a large number of devices or to provide lots of custom email settings, network settings, or certificates to a large number of devices, configuration profiles are an easy way to do it.

A configuration profile contains a number of settings that you can specify, including: Restrictions on device features, Wi-Fi settings, VPN settings, Email server settings, Exchange settings, LDAP directory service settings, CalDAV calendar service settings, Web clips, Credentials and keys.

2.2.2 Configuration Profile Keys

At the top level, a profile property list contains the following keys:

Key	Type	Content
PayloadContent	Array	Array of payload dictionaries.
PayloadDescription	String	A description of the profile, shown on the Detail screen for the profile.
PayloadDisplayName	String	A human-readable name for the profile.
PayloadIdentifier	String	This string is used to determine whether a new profile should replace an existing one or should be added.
PayloadOrganization	String	A human-readable string containing the name of the organization that provided the profile.
PayloadUUID	String	A globally unique identifier for the profile.
PayloadRemovalDisallowed	Boolean	If present and set to true, the user cannot delete the profile.
PayloadType	String	The only supported value is Configuration.
PayloadVersion	Integer	The version number of the profile format.

Figure 15. Yellow Highlight Can Be Abused

2.2.3 Web Clip Payload

The Web Clip payload is designated by specifying com.apple.webClip.managed as the

PayloadType value. This payload defines the following keys:

Key	Type	Content
PayloadType	String	The payload type.
PayloadVersion	Integer	The version number of the individual payload.
PayloadIdentifier	String	A reverse-DNS-style identifier for the specific payload.
PayloadUUID	String	A globally unique identifier for the payload.
PayloadDisplayName	String	A human-readable name for the profile payload.
PayloadDescription	String	A human-readable description of this payload.
PayloadOrganization	String	A human-readable string containing the name of the organization that provided the profile.
URL	String	The URL that the Web Clip should open when clicked.
Label	String	The name of the Web Clip as displayed on the Home screen.
Icon	Data	A PNG icon to be shown on the Home screen.
IsRemovable	Boolean	If false, the web clip is unremovable.

Figure 16. Yellow Highlight Can Be Abused

2.2.4 Technical Analysis Part

In this part, we would analysis the core part of this iOS ransomware—configuration profile installation in detail. We would not only expose the root cause to ransom user's screen and system process crash, but also propose new attack interface for iOS phone attack.

2.2.4.1 Remote Entry for Profile Installation

There exists multiple remote entry for profile installation. Some system application of Apple would automatically download the remote profile and recognize the profile format and trigger system daemon to handle.

For example, end user could simply click the hyperlink of profile in website (as the author did) or click the profile as the mail attachment.

So how does the author do that?

1. This author will create a template on his server. This template will pre-set "Un-Removable".

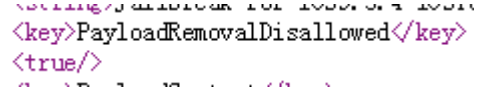

Figure 17. PayloadRemovalDisallowed = True

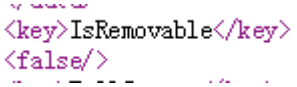

Figure 18. IsRemovable = False

2. When the browser accesses this page, this template will be downloaded first.

```
function submitReq1() {
    // XMLHttpRequestオブジェクト取得
    httpObj1 = createHttpObject();
    if (httpObj1) {
        httpObj1.open("GET", "ixint1.txt", true);
        // 状態が変わったときに実行する処理を記述
        if (httpObj1.readyState == 4) {
            if (httpObj1.status == 200) {
                // 正常に受信完了できたとき読み込
                var res = httpObj1.responseText;
```

Figure 19. Download Template

3. Then he modifies “PayloadIdentifier” and generates a new malicious profile for multiple installations.

```
var length = 8
var randomID = Math.random().toString(36).slice(-length);
value2 = insertStr(res, 377, randomID);

function insertStr(str, index, insert) {
    return str.slice(0, index) + insert + str.slice(index, str.length);
}
//1
document.write(value2)
//dl
function textSave(name, text) {

var blob = new Blob( [text], {type: 'application/x-apple-aspen-config'} )
var link = document.createElement('a')
link.download = name + '.mobileconfig'
link.click()

}
textSave('myfile', value2)
```

Figure 20. Modify “PayloadIdentifier” and Generate a New Configuration Profile

2.2.4.2 Trigger Profiled for Installation

The remote entry (e.g. Safari, email) process would then trigger profiled daemon process for specific installation.

Figure 21. Safari -> Profiled

2.2.4.3 Interact with “Settings”

When profiled installs the downloaded profile, it triggers “Settings” (com.apple.Preferences) to interact with the user.

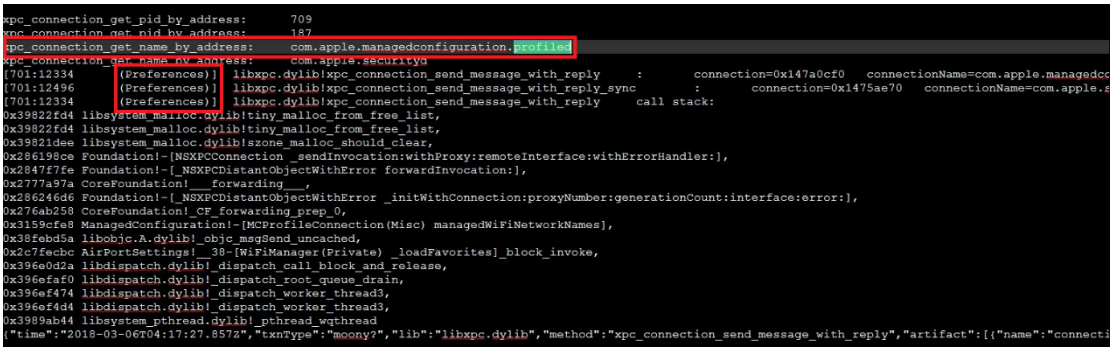

Figure 22. Profiled -> Settings

2.2.4.4 Profiled Handling

The profiled would create the profile at system folder in name of “profie-xxxxxxxxxxxxxxx.stub” in its profile install handler (installProfileHandler API).

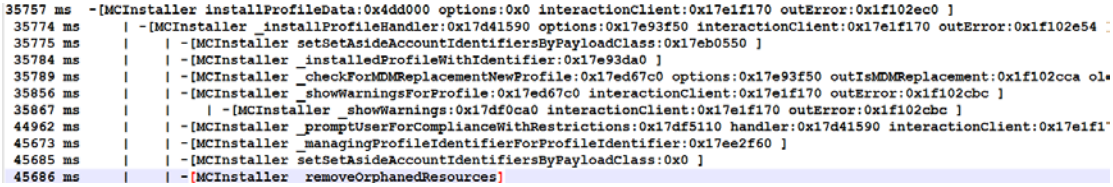

Figure 23. Profile Handle

There exists vulnerability or weakness when profiled analyze the profile. It did not check the

WebClip number limit declared in profile so as permit the malicious (countless) profile occupy the whole screen.

What is more we have found some memory corruption (e.g. UAF vulnerabilities) in the profile installation part which strong hint profiled would be a buggy module.

2.2.4.5 Possible Escalation to High Privilege Process about Profile

Actually, the profile saved would be read and analyzed by other daemon process with high privilege (e.g. launchd) or proxy process (e.g. misagent) to high privilege. For example, misagent (which would xpc with lockdownd daemon with root privilege) would be trigger to enumerate installed provision profile.

```
[1749:46650      (profiled)]:libxpc.dylib!xpc_connection_create: name=com.apple.misagent
call stack:
0x38c4429d libxpc.dylib!xpc_connection_create_mach_service,
0x3831c701 libmis.dylib!0x2701,
0x3831c8f1 libmis.dylib!MISEnumerateInstalledProvisioningProfiles,
0x5cb2f profiled!0x55b2f,
0x4631d profiled!0x3f31d,
0xbf85 profiled!0x4f85,
0x38a6cd2b libdispatch.dylib!_dispatch_call_block_and_release,
0x38a7992b libdispatch.dylib!_dispatch_queue_drain$VARIANT$mp,
0x38a78f69 libdispatch.dylib!_dispatch_queue_invoke$VARIANT$mp,
0x38a7baf1 libdispatch.dylib!_dispatch_root_queue_drain,
0x38a7b4d5 libdispatch.dylib!_dispatch_worker_thread3,
0x38c26b45 libsystem_pthread.dylib!_pthread_wqthread
```

Figure 24. Profiled -> Misagent

As the real vulnerability we have found in section 2.3, SpringBoard process would be heap overflowed when read in huge amount of WebClip icon to show on iPhone screen.

2.2.4.6 Attack Chain

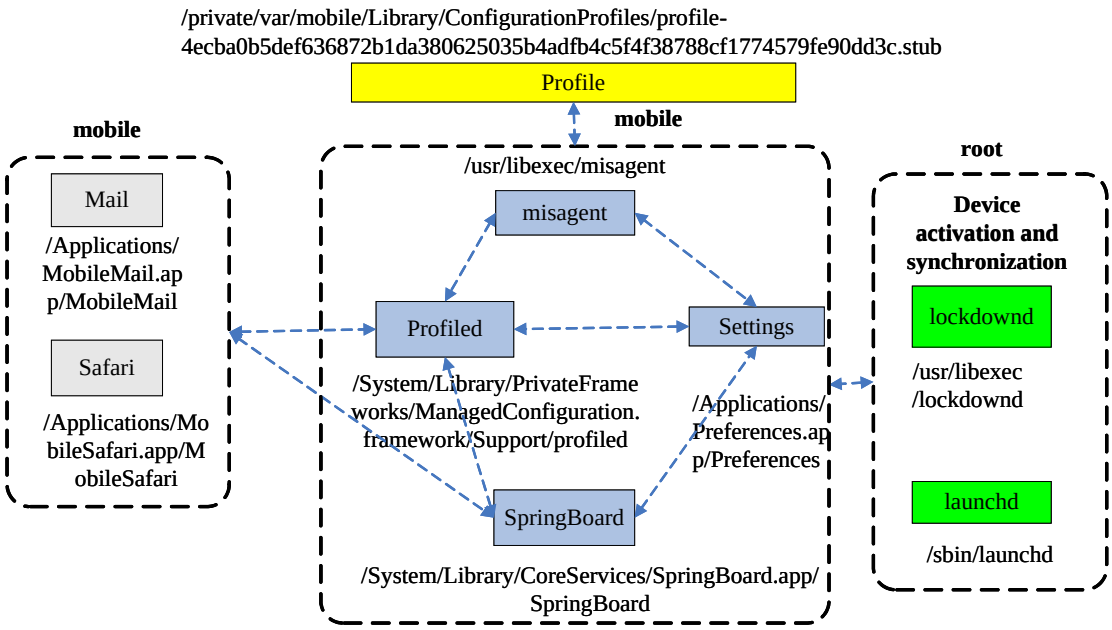

Figure 25. Attack Chain

2.3 New Remote Attack Interface

The profiled installation would be a new remote attack interface for iOS devices in the future as we expected.

2.3.1 Reason

1. Profile install is natural remote attack interface which bypass the Apple Store security audit and app install check.

2. Many daemon processes would read the profile content which hint that attacker would exploit the high privilege process related to profile step by step by construct buggy profile content.

For example, Safari->profiled->misagent->lockdownd (root).

2.3.2 Vulnerabilities Hunt

Actually, we have hunt series vulnerabilities (which we call “death profile”) about profile mechanism based on remote profile install in iOS 10.3.3 (The bugs should be existing on all versions of iOS at that time):

1. Profiled and other system process heap overflow.

The root cause lays that profile related process did not check the limitation of WebClip Payload number in one malicious profile. We have found SpringBoard process (the application that manages the home screen and controls how apps are displayed and launched) crashed when huge amount of WebClip are installed.

Like:

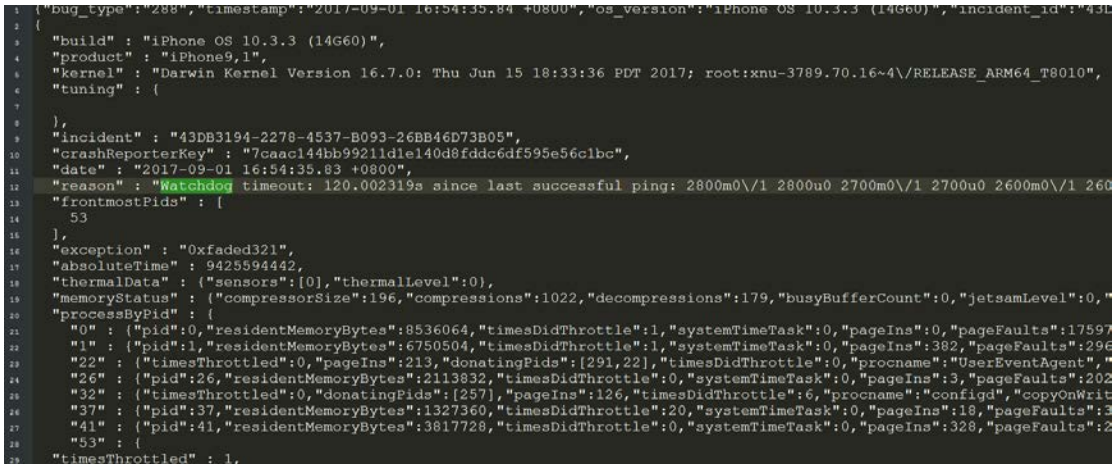

Figure 26. SpringBoard Crash Because of WatchDog's Timeout

2. Enterprise Certificate App Trust Vulnerability.

By fuzzing the profile reinstall operations, we found a new related vulnerability again, and exploit them to bypass Enterprise Certificate App Trust under certain conditions. The root cause would lay the trust state disorder during profile reinstall.

So far, Apple hasn't fixed it.

2.3.3 Issue Analysis

When SpringBoard's vulnerability was triggered, we found that its process had a particularly high CPU usage.

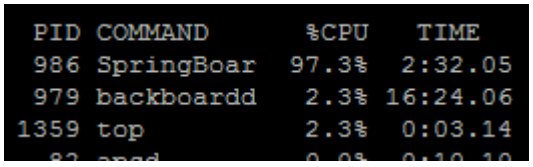

Figure 27. SpringBoard's High CPU Usage

Then we analyzed SpringBoard's dynamic call stack, we found a null pointer reference and a dead loop.

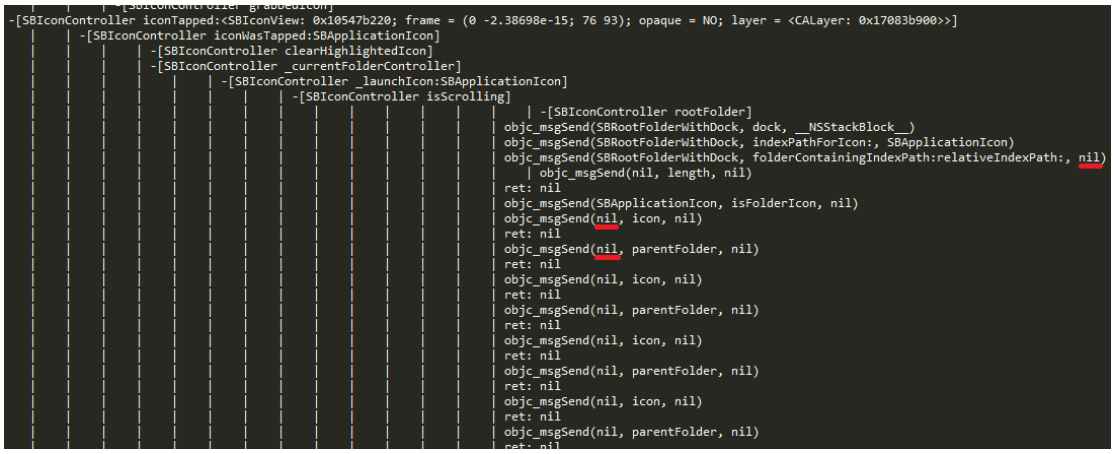

Figure 28. Null Pointer Reference

Back to SpringBoard's decompiling code, we found the final issue function by analyzing the function call flow.

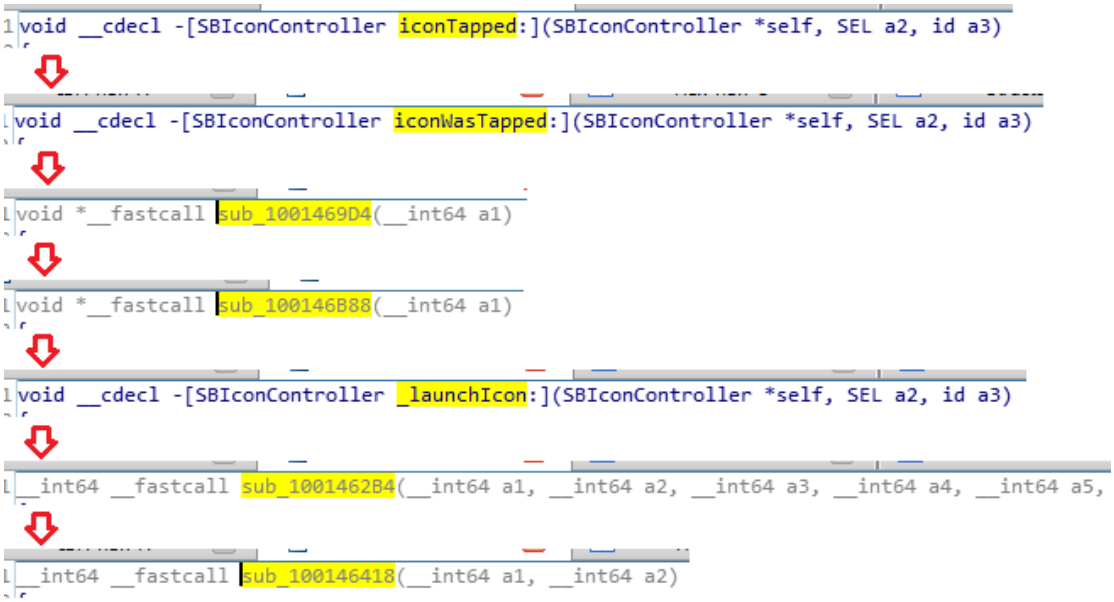

Figure 29. Function Call Flow

And we found that because "v6" is equal to 0, that leads to "v18" which is equal to 0. SpringBoard goes into the dead loop.

```
objc_msgSend(v4, "indexPathForIcon:", *(_QWORD *)(v3 + 40));
v6 = objc_retainAutoreleasedReturnValue();
v23 = 0LL;
objc_msgSend(v4, "folderContainingIndexPath:relativeIndexPath:", v6, &v23);
v7 = objc_retainAutoreleasedReturnValue();
v9 = objc_retain(v23, v8);
objc_release(v6);
*(_QWORD *)v2 = objc_msgSend(*(void **)(v3 + 40), "isFolderIcon", v9);
*(_BYTE *)(v2 + 9) = v7 != (_QWORD)v4;
v11 = (void *)objc_retain(*(_QWORD *)(v3 + 40), v10);
v13 = objc_retain(v7, v12);
v14 = v13;
if ( (void *)v13 == v4 )
{
  v19 = v13;
}
else
{
  v15 = (void *)v13;
  v16 = v11;
  do
  {
    objc_msgSend(v15, "icon");
    v11 = (void *)objc_retainAutoreleasedReturnValue();
    objc_release(v16);
    objc_msgSend(v15, "parentFolder");
    v17 = objc_retainAutoreleasedReturnValue();
    objc_release(v15);
    v18 = v17 == (_QWORD)v4;
    v15 = (void *)v17;
    v16 = v11;
    v19 = (__int64)v4;
  }
  while ( !v18 );
}
```

Figure 30. Dead Loop

2.4 Solution to Defeat iOS Profile Ransomware

As we know, mobile security defense is hard to implement because security vendor usually does not have enough system privilege to monitor system behavior. However, there are always catches to implement security defense.

2.4.1 Detection Solution

2.4.1.1 Static Detection

Static detection is suitable for be deployed at iOS front end to detect death profile attack at near-real time. As we all known, iOS applications are usually limited in apple sandbox so as not to get enough privilege to implement system scan. So we deploy profile detection in TrendMicro TMMS production based on M (Mobile) D (Device) M (Management).

2.4.1.1.1 Profile Abnormality Detection

The profiled daemon could not process the badly constructed profile properly so as to permit huge quality of WebClip icon and occasional system crash.

As we summarize some static pattern for bad profile is:

1. Too much WebClips.
2. PayloadRemovalDisallowed is true for every WebClip.

2.4.1.2 Dynamic Detection

Dynamic detection is the iOS back-end solution based on jail-broken iOS devices.

2.4.1.2.1 Frida Hook

For jail-broken iOS (10.3.2 and earlier currently), we could utilize Frida to hook such point for profile install entrance process (e.g. profile as URL in Safari, profile as attachment in Mail).

2.4.1.2.1.1 Profile Configuration Management

ManagedConfiguration module is the wrapper of XPC client to request profile install.

For example, one typical API:
“ManagedConfiguration!+[MCManifest installedProfileWithIdentifier:]”

[2916:289299 (profiled)]: libsystem_kernel.dylib!__read_nocancel call stack:
0x26032dcb Foundation!_NSReadFromFileDescriptorWithProgress,
0x26032c11 Foundation!_NSReadBytesFromFileWithExtendedAttributes,
0x260328ad Foundation!-[NSData(NSData) initWithContentsOfFile:],
0x26037f33 Foundation!+[NSData(NSData) dataWithContentsOfFile:],
0x2f1063d3 ManagedConfiguration!+[MCManifest installedProfileDataWithIdentifier:],
0x2f10629d ManagedConfiguration!+[MCManifest installedProfileWithIdentifier:],
0x14208f profiled!0x4f08f,
0xf7eeb profiled!0x4eeb,
0x37284d2b libdispatch.dylib!_dispatch_call_block_and_release,
0x3729192b libdispatch.dylib!_dispatch_queue_drain$VARIANT$mp,
0x37290f69 libdispatch.dylib!_dispatch_queue_invoke$VARIANT$mp,
0x37293af1 libdispatch.dylib!_dispatch_root_queue_drain,
0x372934d5 libdispatch.dylib!_dispatch_worker_thread3,
0x3743eb45 libsystem_pthread.dylib!_pthread_wqthread

2.4.1.2.1.2 Web Traffic Request

CFNetwork framework is the typical http protocol foundation framework.

For example, the typical send request API:
“+[NSURLConnection sendSynchronousRequest:returningResponse:error:]”

Or We could wait for the profiled process to be launched to handle the profile install.

At that moment, we could get the profile content for checking.

2.4.1.2.2 Remote Packet Capture for iOS Devices

Since from iOS 5, apple has introduced “Remote virtual interfaces” mechanism to represents the remote network stack of the iOS device, what is more, you need not to jail break the iOS.

We could capture the iOS network traffic in real time and analysis the profile content for check on MacOS.

2.5 Real Attack Demo and Detection Demo

In this part, we would show the iOS profile ransomware remote attack step by step in live and in video format.

And also we would show our solution to detect such kind of threat in live and in video format also.

3. Reference

3.1 国内初のランサムウェア作成事案と他に関連が疑われる作成物についてまとめてみた

http://d.hatena.ne.jp/Kango/20170611/1497198757#20170611fn1

3.2 未成年者がランサムウェアを作る時代、日本初の逮捕事例を読み解く

http://blog.trendmicro.co.jp/archives/15133

3.3 Configuration Profile

https://developer.apple.com/library/content/featuredarticles/iPhoneConfigurationProfileRef/Introduction/Introduction.html

3.4 iXintpwn/YJSNPI Abuses iOS's Config Profile, can Crash Devices

http://blog.trendmicro.com/trendlabs-security-intelligence/ixintpwn-yjsnpi-abuses-ioss-config-profile-can-crash-devices

3.5 App Stores that Formerly Coddled ZNIU Found Distributing a New iXintpwn/YJSNPI Variant

http://blog.trendmicro.com/trendlabs-security-intelligence/app-stores-formerly-coddled-zniu-found-distributing-new-ixintpwnyjsnpi-variant