Bypass iOS Version Check and Certification Validation

written by Vikram Kulkarni | July 28, 2014

Certain iOS applications check for the iOS version number of the device. Recently, during testing of a particular application, I encountered an iOS application that was checking for iOS version 7.1. If version 7.1 was not being used, the application would not install on the device and would throw an error.

This blog is divided into three parts:

- Change version number value in SystemVersion.plist file
- Change version number value in plist file present in iOS application ipa.
- Use 'iOS-ssl-Kill switch' tool to bypass certificate validation.

Change version number value in SystemVersion.plist file

The version of the iOS device can be faked (on a jailbroken device) in two simple steps by changing the value in the SystemVersion.plist file:

1. SSH into a jailbroken device (or use ifile, available on cydia) to browse through the system folder.
2. Change the 'ProductVersion' value in the '/System/Library/CoreServices/SystemVersion.plist' file to the desired iOS version.

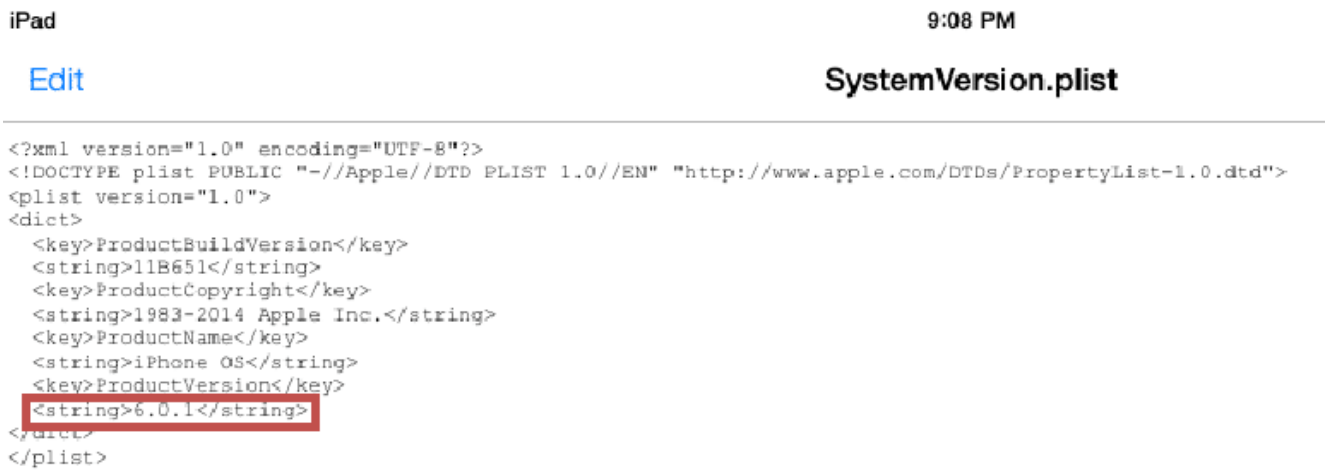

Fig 1: iOS version can be faked by changing the value of ProductVersion key.

This will change the version number displayed in version tab located in 'Settings/General/about' in the iOS device. Although this trick might work on some of the applications that check for the value saved in the '/System/Library/CoreServices/SystemVersion.plist' file, this trick won't work on every application. If it fails, we can use the second method given below.

Change version number value in plist file present in iOS application ipa.

If you are unsure about the method that the application is using to look for the version number, then we can use another simple trick to change the value in the iOS version. The version check in an IPA can be faked in three simple steps.

1. Rename the ipa to .zip file and extract the folder.
2. Find the info.plist file located usually in Payloadappname.app and change the string 'minimum ios version' to the version you need
3. Zip the file again and change it to ipa. *[Note: Some of the applications can also use other plist files instead of the info.plist file to check for minimum version]*

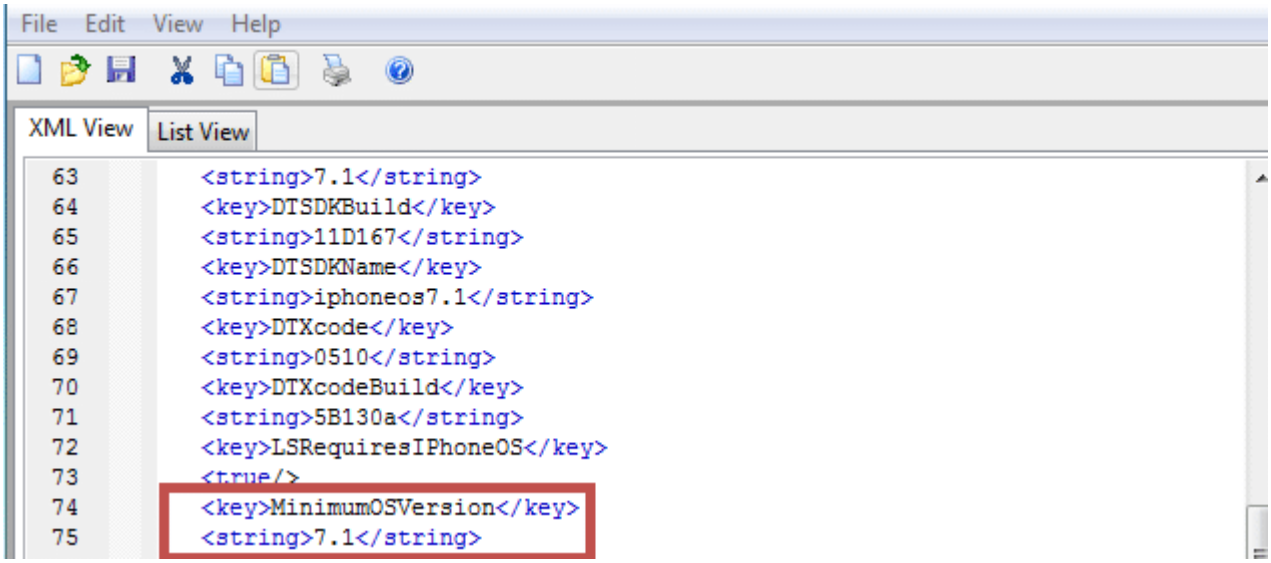

Fig 2: MinimumOSVersion requirement defined in info.plist file in the IOS application.

Manipulating any file inside the IPA will break the signature. So, to fix this problem, the IPA would need to be resigned. We can use the tool given here on [Christoph Ketzler's blog](#).

Some applications also perform the version check during the installation process. When a user tries to install the application using iTunes, or xcode using the IPA, the IPA checks for the version of iOS running on that device and if the version is lower than the minimum required version it will throw an error similar to the one given below.

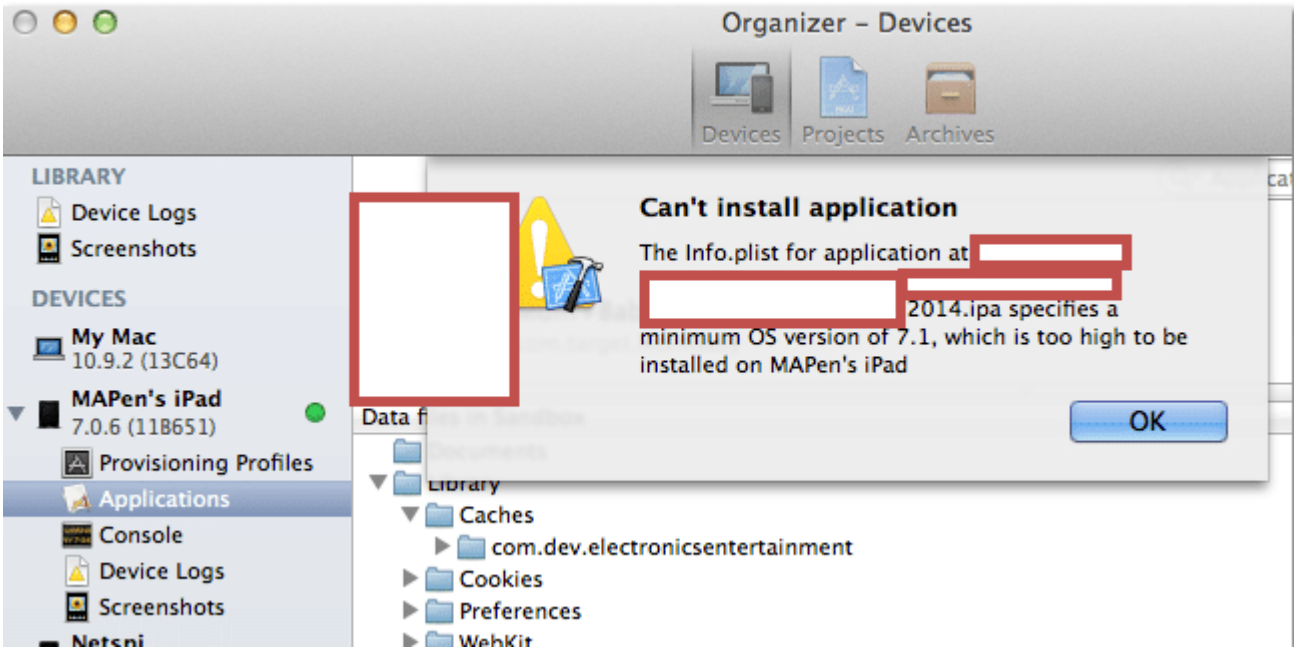

Fig 3: Error message while installing the application using

xcode.

The version check performed during the installation stage can be bypassed using this simple trick:

1. Rename the .ipa application package to .zip and then extract the .app folder.
2. Copy the .app folder to the path where iOS applications are installed (/root/application) using an SFTP client like WinSCP.
3. SSH into the device and browse to the folder where the IPA is installed, then change the permission of the .app folder to executable (chmod -R 755 or chmod -R 777). Alternately you can change the permissions by right clicking the .app in WinSCP, change properties and check all the read, write, and execute permissions.
4. Restart the iOS device and the application will be successfully installed.

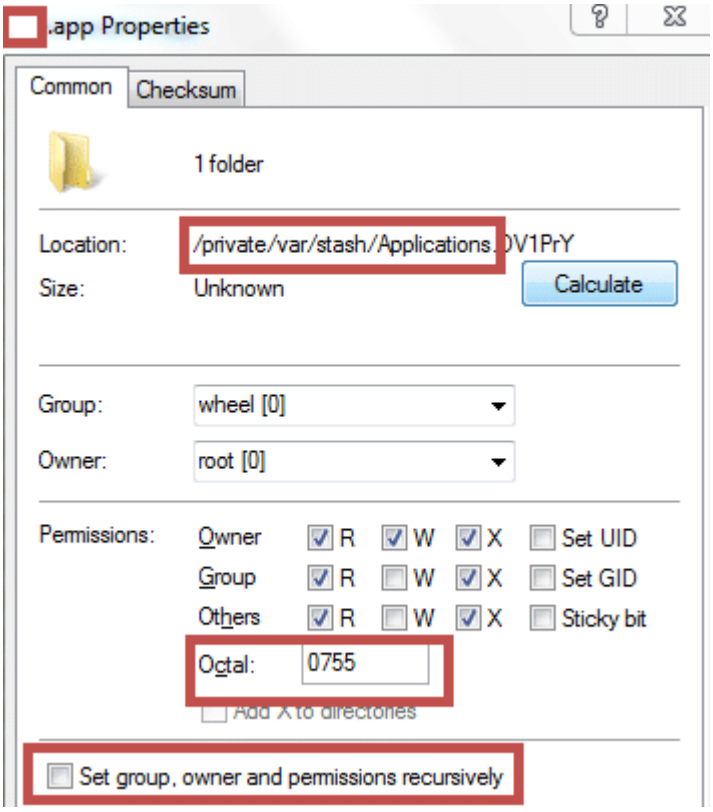

Fig 4: Changing permissions of the IPA to executable.

iOS Certification validation bypass

Some applications perform certification validation. Certification validation is performed to prevent application traffic from being proxied using a MitM proxy like Burp. Typically the application has a client certificate hard coded into the binary (i.e. the application itself). The server checks for this client certificate and if it does not match then it throws a certificate validation error. Refer to my co-worker Steve Kern's blog on Certificate Pinning in a Mobile Application for further details.

Sometimes it is difficult to extract the certificate from the application and install it into the proxy. An alternative approach is to use a tool developed by iSEC Partners called ios-ssl-kill-switch. This tool hooks into the Secure Transport API, which is the lowest level of API, and disables the check for certificate validation. Most certificate validation techniques use NSURLConnection, which is a higher level API call to validate client certificates. More technical details can be found here.

Bypassing Certificate validation can be performed in the following steps:

1. Install the tool kill-ssl-switch
2. Make sure the dependencies given on the installation page are installed prior to the installation of the software.
3. Restart the device or restart SpringBoard using following command 'killall -HUP SpringBoard'
4. Enable the Disable Certificate Validation Option in 'Settings/SSL Kill Switch'
5. Restart the application and confirm that a MitM proxy can intercept the traffic successfully.

Certificate pinning can be bypassed by hooking into the API which makes the check for certificate validation and return a

true value for certificate validated all the time. Mobilesubstrate is a useful framework for writing tweaks for disabling certificate pinning checks. There are few other handy tools as well, like ‘Trustme’ by Intrepidusgroup and ‘Snoop-it’ by Nesolabs to disable Certificate pinning.

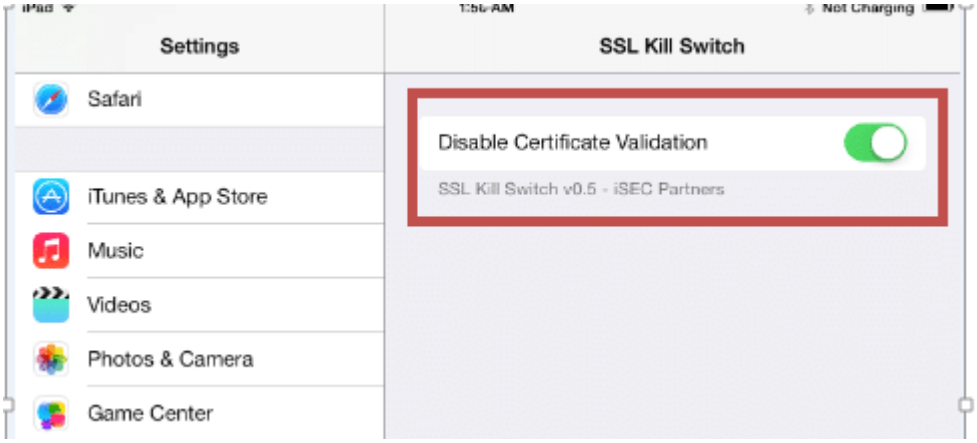

Fig 5: Turn off certificate validation using SSL Kill Switch.