Brad Semrad
SE-4930
December 11, 2011
Article Review 1

The article first discusses the environment of applications for both iOS and Android. First for iOS, it requires that every Application to be placed and bought through the App Store. In addition, each application must be signed with Apple's private key. This helps prevent malware apps from existing. In addition, these apps are sandbox. This only allows the access to the services the application would need. In comparison, the Android also has an App store and requires the application to be signed; however, the apps are not required to be placed on the app store and the applications can be self-signed. The Android apps also live in a sandbox but each application will request different services to use. This requires human intervention which can create security loop holes by allowing apps extra access to services. Each mobile operating system has the ability to remotely remove malware applications as they are discovered. The difference is that the Apple App Store has a review process to allow the application in. The Android will let any application and will remove apps only based on user reviews.

The rest of the article talks about how these mobile phone operating systems could be exploited and what additional secure measures they take. The article talks about how these mobile phones have the same attack surface as a standard desktop because they are just essentially computers. The difference is that there is less code to exploit due to programs like Java and Adobe on a desktop that contain the highest number of flaws. The main two attack vectors that are prevalent in the mobile space are SMS message processing and GSM radio due to the sandboxing that occurs. These exist outside of the sandbox security measures that the phones take. In looking at iOS, they use some additional layers to help secure the mobile phone. iOS uses such measures as data execution prevention and address space layout randomization. Data execution prevention creates a difference of data and code to prevent a hacker from just allowing giving a process data and executing that data. This data would typically be the malicious code. The way that hackers can bypass this information is by using a return-oriented programming but due to the address space layout randomization, it cannot be exploit as easily because it cannot find the location in memory for the code to exploit as it is randomized. This causes items to be harder actually exploit the vulnerabilities. A hacker can exploit iOS by using the SMS messaging because that runs out of sandbox. Android however, does not use address space layout randomization or data execution prevention. Instead, every single application including SMS lives in the sandbox. Therefore in order to break into the phone, they first have to find vulnerabilities in the application and then another vulnerability to break out of the sandbox. In addition it is harder to exploit as there is no root user or console to make it easier to exploit.

From there article, there are a few things that I can learn from it. The first is to sandbox everything if possible. This provides a safe container that can prevent any applications from destroying the core system and provides one layer of defense. In addition to this, one should have multiple layers of defense. At looking at iOS, they have an app store, code signing, sandbox, data execution prevention, and address space layout randomization. If one layer of security gets bypassed, there are other layers that can foil an attack. In looking at both of the mobile phones, a good secure solution would be to use iOS' app store and code signing, Android;s sandbox everything, iOS data execution prevention and address space layout randomization. These multiple layers provided numerous obstacles to exploit the

machine. In addition, these can create computational complex programs just to execute and find and exploit. The other lesson to take away from this is to be secure by default. In looking at Android's vs. Apple's store, this shows that Apple doesn't want malware and other items getting in first. This is an idea of secure by default. It is easier to create good layers ahead of time then to patch it later.

After reading the articles, the questions I have is how exactly does address space layout randomization and data execution prevention work? In addition, it would be interesting to hear how Blackberry and Windows Phone 7 handle security to compare to these other platforms.