Research Statement

Yousra Aafer

Never before has any OS been so popular as Android. Existing mobile phones are not simply devices for making phone calls and receiving SMS messages, but powerful communication and entertainment platforms for web surfing, social networking, etc. Even though the Android OS offers powerful communication and application execution capabilities, it has also brought a lot of security risks. My thread of research evolved to tackle and investigate several security risks threatening Android users. I first analyzed Android malware, a very prevalent security risk in the Android security community and designed a lightweight detection mechanism. My work is one of the earliest research in Android malware detection achieving a high accuracy[1]. I moved to explore fundamental security decisions in the Android OS and identified several security flaws in the Android Multi-User framework and application uninstallation process. My PhD work highlights several security aspects of the Android fragmentation and led to the discovery of several security threats, never investigated before. My work on Android Security has received positive recognition from leading companies such as Google, Samsung, LG and Sony.

DroidAPIMiner: Mining API-Level Features for Robust Malware Detection in Android (SecureComm' 13): I proposed DroidAPIMiner, a robust, lightweight and efficient classifier for detecting Android malware. Selecting the best features to construct the classification model is one of the challenges that I faced in this research work. The feature set should unbiasedly distinguish a malware app from a benign app and effectively describe malicious behavior that characterize Android malware. To this end, I proposed to rely on API level information extracted from the bytecode such as critical API calls, package level and parameter level information, where Package level information aims to overcome the effect of third party libraries. I followed a generic data mining approach to build a classifier for Android apps based on the features that I selected. The generated classification model using KNN algorithm yielded an accuracy of 98% and a false positive rate of 2.2%.

Hare Hunting in the Wild Android: A Study on the Threat of Hanging Attribute References (CCS' 15): My thread of research continued to investigate other security aspects introduced as a result of the decentralized and fragmented Android eco-system. In fact, Android OS has been heavily customized into thousands of system images by almost everyone in the customization chain such as hardware and device manufacturers and vendors. In this research, I investigated security problems that might be introduced as a result of the customization and discovered a vulnerability, never investigated before, that I named Hanging Attributes References (Hares). I collaborated with other researchers to further investigate this vulnerability. In our research, we show that popular Android devices are riddled with such flaws, which often have serious security implications. When an attribute (e.g., a package/authority name) is used on a device but the party defining it has been removed, a malicious app can fill the gap to acquire critical system capabilities, by simply disguising as the owner of the attribute. We showed that Hares are indeed security critical through real world attacks (Steal user voice notes, replace Google Email's Account Settings Activity, collect user's contacts without proper permissions, etc). We further designed

[1] Citation Count **219** (Based on Google Scholar, as of June 2017).

and implemented Harehunter, a new tool for automatic detection of Hares by comparing attributes defined with those used, and analyzing the references to undefined attributes to determine whether they have been protected.

DroidDiff: Harvesting Inconsistent Security Configurations in Custom Android ROMs via Differential Analysis (USENIX Security' 16): My thread of research moved from trying to reveal specific vulnerabilities that might be introduced as a result of Android customization to generalizing the problem to a broader scope. More specifically, I proposed to systematically detect security configuration changes introduced by different parties in the Android customization chain. My key intuition is that through comparing a custom device to similar devices from other vendors, carriers, regions, or OS versions, we can detect security configuration changes created unintentionally during the customization. Specifically, I first located relevant security features that might be altered during the customization and then extracted them from a large corpus of collected Android custom ROMs (around 600 ROMs). To detect inconsistent security features, I performed a differential analysis among different candidate image sets sharing similar features. The results of my differential analysis revealed that indeed the customization results in inconsistent security configurations, and more dangerously, weaker security features. For example, I found out that vendors sometimes downgrade the privilege of Linux group ids, making normal apps able to access privileged resources with a normal permission. Besides, vendors sometime downgrade the protection level of built-in permissions leading to conflicting definitions throughout different images. To prove that the security configuration inconsistencies revealed in our analysis can lead to actual vulnerabilities, I designed real-word attacks to exploit some inconsistencies. I was able to conduct security critical attacks such as factory resetting the phone without a user confirmation, reading user emails without a permission and accessing critical drivers with a normal permission.

Life after App Uninstallation: Are the Data Still Alive? Data Residue Attacks on Android (NDSS' 16, Esorics' 16). We investigated the effectiveness of Android app uninstallation process. We revealed that Android's data cleanup mechanism is vulnerable: Uninstalled apps can leave data residues that can be exploited by unprivileged applications. However, this does not necessarily lead to security breaches, as long as the data are well guarded even after the owner is removed. Although such a lifetime protection is theoretically feasible, Android seems to be confused in identifying the rightful owner of the residues because of certain implicit assumptions. Our study attempted to unveil these implicit assumptions and more importantly examine their validity. We created scenarios to make those assumptions false, and investigated how Android handles the data residues in these conditions. We conducted real-world attacks to measure possible damages of detected data residues. An adversary can exploit these cases to steal online accounts credentials, escalate its privilege, and even eavesdrop on input keystrokes.

Work(s) Under Submission and Research Agenda: During my postdoctoral research at Purdue University, I have continued investigating security aspects of Android customization. Specifically, my research aims to leverage static analysis in order to detect framework-level access control inconsistencies that are uniquely introduced by vendors or version upgrades. The results of my investigation are alarming: I was able to exploit framework inconsistencies through high impact attacks (2 were ranked as critical security by LG). These findings as well as my previous work on customization have laid the direction for my future research: I plan to propose a system for helping vendors regulate their customization process of the framework services and preloaded system apps.