CS 111 ANDROID VULNERABILITIES

Luke Jung

lukejung@ucla.edu

Eggert Winter 2019

March 15, 2019

1 INTRODUCTION

In Parnika Bhat and Kamlesh Dutta's article, *A Survey on Various Threats and Current State of Security in Android Platform,* the authors write about many different exploits and vulnerabilities in current Android software. With the popularity of smartphones to almost all people, personal data and information continues to increase on an individual's phone. Data such as payment solutions, personal information, and even confidential documents have made these devices even more valuable than before. With this in mind, hackers have been focusing effort on finding these exploits for personal gain through ransomware or denial or services attacks to extrapolate either money or information. The article organizes the bulk of these hacks into four categories, Kernel Based attacks, Hardware abstraction layer-based attacks, and application-based attacks. Each category has its own section, some bigger than others, explaining the vulnerabilities, the exploits, and ultimately the potential solution to each problem. However, it is important to note that each problem doesn't have 100% solution every time, as well as problems are constantly being created daily. Also, unfortunately most of the vulnerabilities could have been safeguarded by with better written programs, ones that were written that acknowledged the chance of these attacks. Most of the attacks are focused on obtaining root access to kernel, using this access to control various resources and leak information. While an ongoing problem, this article gives a comprehensive view of some of the problems that are currently attacking android operating systems, giving users a better view on how these attacks can occur, and how to better protect themselves from these hacks.

2 SUMMARY

Many of the problems of android software come from four main categories, Hardware based attacks, Kernel Based attacks, Hardware abstraction layer-based attacks, and application-based attacks. Even though each attack comes through at very different points, most attacks end up trying to achieve the same thing, to gain root access to the phone to access private information or control.

For Hardware Based attacks, Bhat and Dutta explain how hardware in android smartphones exist in "preinstalled vulnerable components in the device provided by the manufacturers" (5). Meaning that these vulnerabilities pop up because of unsafe software written for different hardware devices, not from

the OS itself. An example of this hardware vulnerability is that some hardware that is used when replacing touchscreens may be from 3rd party distributers which have unauthorized components which attackers can use to use "touch injection" or "buffer overflow." These attacks are common because the android operating system doesn't always check "the integrity of the communication between the replaced component and central processor of the system" (5). Other than 3rd party hardware, another common attack is called the "Rowhammer attack," which installs malicious software to gain access to the GPU and "hammers" a memory location to cause an electrical leak. This causes a change in memory state and eventual access to gain access to the kernel. This exploit has been partially solved by programs like "GuardION" which utilizes "DMA allocations [that] are shielded using rows that are empty, which results in isolated bitflips" (5). Most hardware attacks are in the similar vein, where hackers would utilize isolated pieces of hardware to gain root access. These attacks are hard to defend against as well, due to the patches are only available with the distributors and carriers.

Kernel-based attacks on the other hand have a few more avenues and different vulnerabilities that affect it. In Kernel-based attacks, attackers can target root privilege, memory, bootloaders, and device drivers. For specifically root privilege attacks, attacks are done by using vulnerabilities found in the Kernel's API to escalate privileges and execute malicious code. One specific vulnerability is in the "put_user/get_user" operations, where "attackers inappropriately perform read/write operations on the kernel memory" (7). Another well-known kernel attack is called the *DroidKungfu Attack,* where malware roots into the devices and uses system resources without asking the user for any permissions. It does this by exploiting the "setuid()" command, first calling an adb function which uses "setuid" to lower its own privilege. However, before it can do this, the malware first forks a bunch of times until it reaches the maximum number of processes, which then kills the adb process. Then, once the adb command starts again, it can't reduce its own privilege because "the exploit has already started RLIMIT_NPROC-1 processes other than adb process. So the call to setuid() fails" (7) and continues to run as root. Eventually through this, the attacker gains unprivileged access and is able to utilize the system resources. While there are many other exploits in the kernel, each ends up the same, utilizing kernel code to gain root access and cause actions that were originally supposed to be protected.

Hardware abstraction layer-based attacks differ from straight hardware attacks because these attacks target the library modules that implement interfaces, rather than the actual hardware itself. Like the hardware though, these modules are open to attacks because the API's are usually well known and replicated for many phones, meaning the code itself is more accessible and can be used on a number of devices. One specific example of this is in Broadcom Company's Wi-Fi chipset, attackers found a vulnerability that allowed them "to get control of the complete system by using Booby-Trapped Signals" (12). This problem stemmed from the WPA2 protocol that is common in almost every Wi-Fi network, which was attacked using a process called "Key Reinstallation Attack (KRACKs)." By being in the vicinity of someone's Wi-Fi range, hackers have the ability to steal confidential data and even access encrypted data. The article states hackers can also use "applications to spy on the user's data using Audio Channels, GPS, Camera, and other components" (12). And while these attacks are patched relatively quickly, the window of opportunity for hackers, as well as many outdated devices make hacks like these very realistic, especially to the everyday person.

Finally, for the last category of attacks, Bhat and Dutta give different examples of application-based attacks. These attacks are also subcategorized in library-based attacks, third-party library attacks, intra-library collusion, and privilege escalation attacks. Intra-library collusion is where an individual library obtains a combined set of privileges assigned to multiple applications, because many applications share libraries and sometimes applications get access to an entire set of privileges not usually assigned to them. Each category has their own separate details, but the main point is that many of the libraries that so many applications use all the time are usually flawed, and when these libraries are vulnerable, every device and system that uses these libraries are also vulnerable to attacks as well.

Near the end of the article, the writers also comment on the reasons why so many of these vulnerabilities exist, citing the android permission system and an overall lack of awareness for security protocols. However, with all these problems the authors also bring up points of different programs that are combating these vulnerabilities today, that even though there may be many paths to attacks, new programs and functions are constantly being created to safeguard software that might be vulnerable. This means we don't have to necessarily overhaul entire devices and code, since we can create software that will help protect against these attacks. So, while the war against cybersecurity and vulnerabilities will never let up, the article explains how by acknowledging the problems, programmers can work at writing safer and stronger programs and in the long run, use "continuous research and innovation…to deal with the latest threats" (29).

3 Comments

I first chose this article because I thought reading about hacking and vulnerabilities would be interesting, wondering if all the media and ideas I've seen in moves may be true. While hacking may still be an abstract to me, going through this article helped me to realize how vulnerable our systems really are, and that I for one need to be more careful about what I do with my devices, as well as write better code for the future if I ever get the chance. For the article itself though, I think the authors did a great job of going through each category and explaining the different attacks and vulnerabilities from an OS standpoint. Concepts in the article such as the accessing the kernel and memory management were "aha" moments to me, because I could connect the ideas with lectures specifically from CS 111. Even the concept of TLS public key infrastructure was so topical for me, since our last beagle bone project was about connecting to a server through that authentication. At the end of the day, I think this article helped me to understand better the ways that hacks on the operating system are so common, and that the lectures and concepts that I've learned in the past 10 weeks aren't useless knowledge. That these lessons shouldn't be ones that I'll forget next quarter, but really teachings that I need to understand to better write programs that will be safe from the endless amount of attacks that will only increase in the coming years.

4 References

Bhat, Parnika, and Kamlesh Dutta. "A Survey on Various Threats and Current State of Security in Android Platform." *ACM Computing Surveys (CSUR)*, ACM, Feb. 2019, dl.acm.org/citation.cfm?doid=3309872.3301285.