

Samuli Käppi

EVOLUTION OF MALWARE AND SECURITY IN ANDROID BASED ENVIRONMENTS

Faculty of Information Technology and Communications Sciences
Bachelor's degree
September 2024

TIIVISTELMÄ

Samuli Käppi: Evolution of Malware and Security in Android Based Environments
Kandidaatintyö
Tampereen yliopisto
Tietotekniikan kandidaatin tutkinto-ohjelma
07 2024

Android-käyttöjärjestelmän valtavan suosion nousun syynä on todennäköisesti käyttöjärjestelmän avoimen lähdekoodin luonne ja siirtyminen älypuhelimiin vuonna 2008. Tämä mahdollisti sen, että puhelinvalmistajat pystyivät saavuttamaan kustannussäästöjä käyttämällä ilmaiseksi saatavilla olevaa Android-käyttöjärjestelmää. Kun Google osti Androidin vuonna 2005, käyttöjärjestelmä sai taakseen teknologiajätin, mikä tarkoitti, että sillä olisi potentiaalia laajempaan käyttöön.

Tämä tutkielma tarkastelee Android-haittaohjelmien kehitystä niiden ensimmäisestä esittelystä FakePlayer-muodossa vuonna 2010 nykyisiin monimutkaisempiin muotoihin viime vuosina. Analyysi suoritettiin kirjallisuushaulla käyttäen tutkielman aiheeseen liittyviä julkaisuja ja muita lähteitä. Tutkielma käsittelee maksullisia tekstiviestipalveluita hyödyntävien haittaohjelmien varhaisia vaiheita, uudelleenpakattujen sovellusten nousua ja sovelluksien oikeuksien hyväksikäyttöä sekä taloudellisesti motivoituneiden haittaohjelmien, kuten kiristysohjelmien ja cryptojacking-ohjelmien lisääntyvää esiintymistä.

Lisäksi tämä tutkielma analysoi Googlen toteuttamia turvatoimia haittaohjelmaympäristön jatkuvan kehittymisen torjumiseksi. Androidin turvaominaisuudet ovat kehittyneet huomattavasti Androidin alkuvaiheista lähtien. Tavallisista pin-koodeista ja -kuvioista on siirrytty biometriaan ja salaukseen, joka estää pääsyn laitteen dataan, mikäli laite varastetaan. Sovellusten oikeusjärjestelmä on uudistettu useita kertoja eri Android-versioiden välillä, mikä on lisännyt käyttäjien valtaa järjestelmän suhteen. Nykyiset Android-laitteet ovat turvallisempia kuin koskaan, mutta haittaohjelmat kehittyvät samanaikaisesti turvajärjestelmien kanssa.

Avainsanat: Android, Haittaohjelma, Turvajärjestelmä.

Tämän julkaisun alkuperäisyys on tarkastettu Turnitin OriginalityCheck -ohjelmalla.

ABSTRACT

Samuli Käppi: Evolution of Malware and Security in Android Based Environments
Bachelor's Thesis
Tampere University
Bachelor's Degree Program in Information Technology
07 2024

The Android operating system's massive rise in popularity is most likely caused by the open-source aspect of the operating system and the transition to smart phones in 2008. This allowed the phone manufacturers to save on costs by using the Android operating system that was available for free and the fact that Google had acquired Android in 2005 meant that the operating system had a tech giant backing it, meaning it likely wouldn't be abandoned immediately.

This thesis examines the evolution of Android malware from its first introduction in the form of a FakePlayer in 2010 to the current more sophisticated forms in recent years. The analysis was conducted as a literature search utilizing publications and other sources related to the thesis topic. The thesis explores the early stages of premium service SMS theft, the rise of repackaged applications and exploitation of app permissions and the increasing prevalence of financially motivated malware such as ransomware and cryptojacking.

Furthermore, this thesis analyzes the security measures implemented by Google to counteract the ever-evolving malware landscape. The security features on Android have come a long way since the inception of Android. From basic pin codes and patterns to biometrics and encryption against physical access. The permission system for applications has been completely overhauled multiple times between different versions of Android, increasing the users control over them. Current Android devices are more secure than ever, but the malware keeps evolving alongside the security.

Keywords: Android, Malware, Security

The originality of this thesis has been checked using the Turnitin OriginalityCheck service.

SISÄLLYSLUETTELO

1.INTRODUCTION	1
2.EVOLUTION OF MALWARE	2
3.ATTACK SURFACES	9
4.EVOLUTION OF SECURITY FEATURES ON ANDROID	10
4.1 Permissions	10
4.2 Protection against physical access	12
5.CONCLUSION	14
SOURCES	15

1. INTRODUCTION

Smartphones and tablets have become increasingly popular devices which usually contain a lot of personal information. With the rise of the smartphones the Android operating system also gained its foothold in the smartphone operating system economy, its popularity can be attributed to it being free to use for the manufacturers and because Google acquired it in 2005 [1], so it had a tech giant backing it. The widespread usage has made them an increasingly attractive target for hackers. As of 2017 smartphones running the Android operating system held an 85.1% share of the global market [2]. According to Kaspersky 98.05% of malware that targets smartphones targeted Android in 2014 [3]. This statistic is quite alarming considering that on average 87.7% [4] of Android devices were exposed to a known critical vulnerability in 2015.

The Android operating system has gone through a dramatic evolution since its inception in 2008. Alongside its widespread adoption, there has been a continuous rise in Android malware seeking to exploit vulnerabilities and compromise user data. This thesis explored this dynamic landscape, tracing the development of Android malware from the first FakePlayer malware to the sophisticated attacks witnessed in recent years. The early years of Android saw rudimentary malware focused on premium service SMS scams. As the platform matured, so did the malware, incorporating techniques such as repackaging legitimate applications with malware. The emergence of app stores such as Android Market (latterly Google Play), which intended to be secure platforms, introduced a new common attack vector. Malicious actors abused the permission system and used social engineering to trick users into installing malware.

2. EVOLUTION OF MALWARE

The evolution of Android malware is following the trends of PC malware but at an accelerated pace, since the malware on PC systems has had time for trial and error, the mobile malware was able to learn from it instead of having to go through phases of trial and error.

The first Android malware FakePlayer was reported in August 2010 by Denis Maslennikov, who was working at Kaspersky at the time [5]. It came in a form of a video player for viewing porn. During installation the video player requested access to sending messages and if installed the application proceeded to send messages to two premium rate numbers costing approximately 5\$ per message. The whole malware was only 334 lines of code including empty lines and comments [6]. This malware was rather simple, but it showed initiative in malware. When the malware was first spotted, the Android Market wasn't available for all devices and regions, so some Android device owners had to rely on third-party websites to download their apps from. These third-party marketplaces didn't perform adequate checks for applications uploaded to them, so they provided an easy way of spreading malware. This in addition to the ease of sending money via text messages at the time made this malware simple yet effective.

This malware was improved upon by a malware family named FakeInstaller. The way the malware worked was essentially identical to FakePlayer. It tricked the user into downloading it by mimicking an already existing application. When the application is opened it displays a service agreement that notifies the user that one or more SMS's will be sent. After agreeing to the terms or in some cases before agreeing to the terms one or more SMS are sent to premium rate numbers. In some cases, the application the user was looking for was provided to them. The difference between FakePlayer and FakeInstaller is in how FakeInstaller avoids detection. Most of the FakeInstaller variants are server-side polymorphic [7], which means that the server that provides the malicious applications can provide different APK files when requesting the same URL from different IP addresses. The differences in the files are very minor and don't affect the payload, but these minor differences produce changes in the digital signature of the APK to avoid being immediately detected. The changes in the APK file also allow for the payload to be country specific based on the IP address of the person who downloaded it. The malware also included typical obfuscation where the field names, argument names, variable names, etc. Got changed. Some versions of FakeInstaller also included backdoors so

that the infected device can receive commands from a remote server. The malware was mostly spread through fake websites.

Mobile ransomware started gaining popularity in mid 2014 and skyrocketed in 2016 where the number of infected users grew nearly fourfold compared to the number of attacks in the previous year [8]. This evolution is visible in Figure 1.

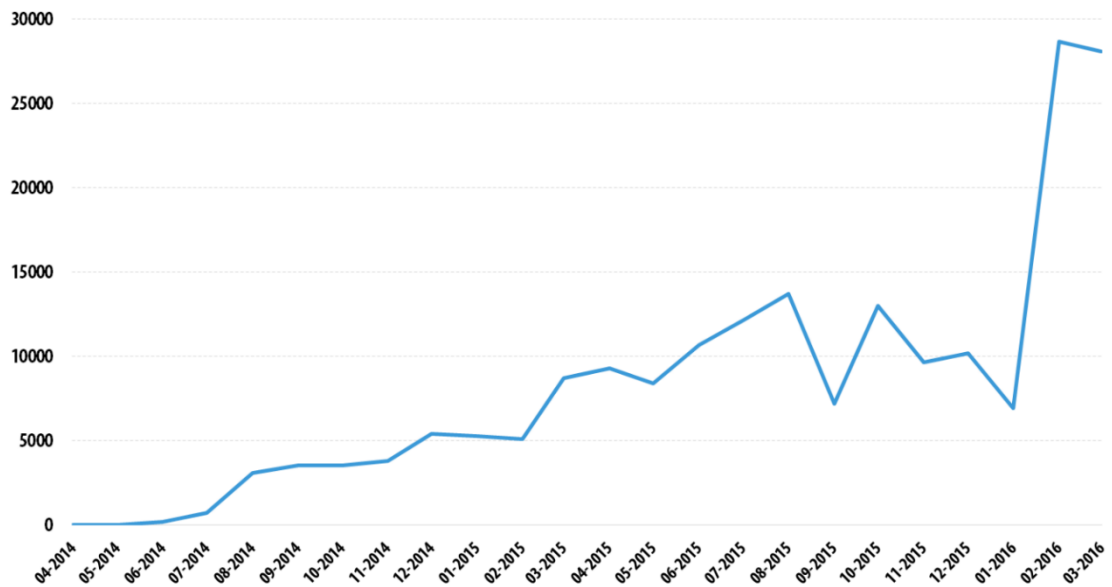


Figure 1: The number of Kaspersky's users encountering mobile ransomware at least once [8].

Even though the number of users who encountered mobile ransomware is still rather low compared to PC users who have encountered ransomware during the same period. The massive increase in numbers still painted an alarming trend with mobile malware.

In 2014 the mobile ransomware landscape consisted of multiple different families with Small leading the charge but in 2015 the two largest families had 93% of the “market” share, those being Fusob at 56.23% and Small at 37.23% [8]. The Android ransomware differs from typical PC ransomware a bit since instead of locking the user out completely it just overlays itself on top of everything else so that the device becomes unusable. Because of the security features in Android encryption ransomware isn't as popular as a blocker one since the damage encryption ransomware can do is really limited, since cloud backups are a common occurrence in Android applications. When any of these ransomwares are installed to the device, they request to become device administrators in order to prevent themselves from being uninstalled [9], as seen in Figure 2.

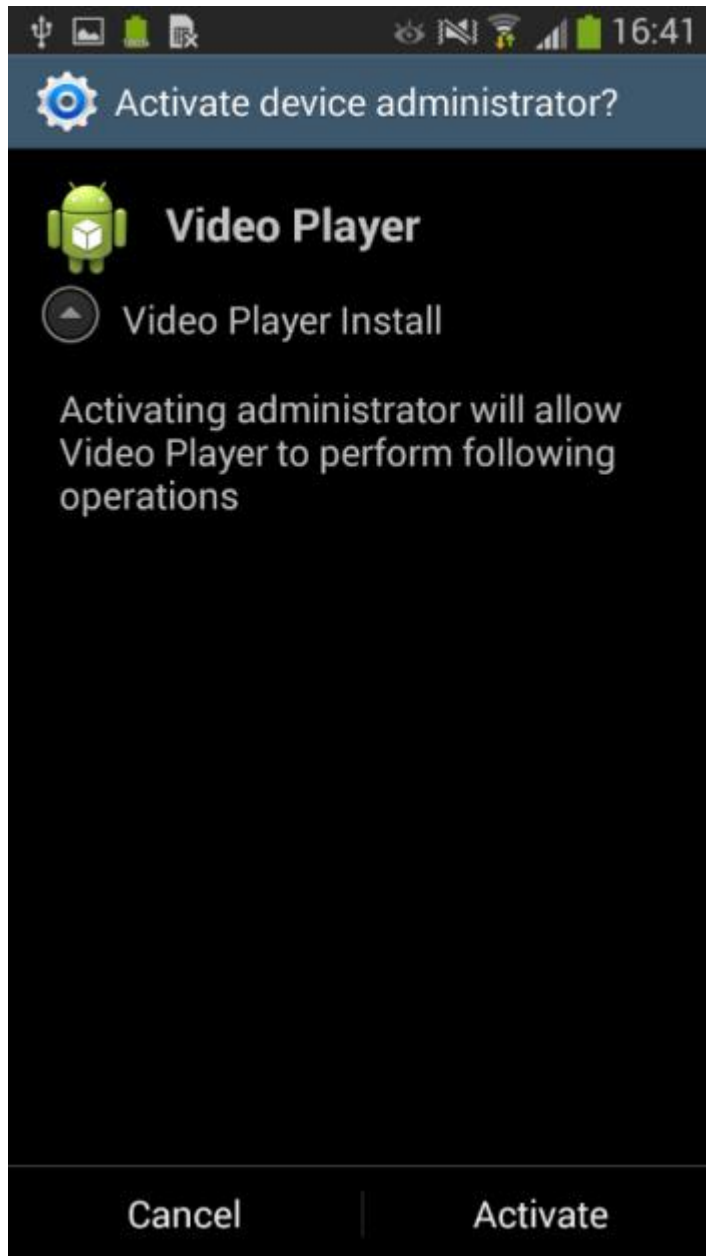


Figure 2: Example of a malware asking to become a device administrator [9].

After gaining the access, most of the Android Ransomwares follow the same pattern. After that they display a typical ransom message requesting money in the form of gift cards or cryptocurrencies. Example of the ransom message can be seen on Figure 3.



Figure 3: Example of a ransom message [8]

The mobile ransomware is mostly distributed via adult entertainment websites.

In 2016 the two most common malware families were Svpeng and Ztorg. Ztorg is a typical SMS-trojan that tries to send messages to premium rate numbers to generate revenue for the attacker. Ztorg was spread through malicious applications in the Google Play store. The trojan waited 10 minutes after installation before communicating with its command-and-control server. It sends two rather innocent GET requests to the server, the first of which contains the first three digits of the devices International Mobile Subscriber Identity. The second request contains the first five digits of the IMSI. The server uses these digits to determine the country and the mobile operator of the device [10] and then responds with instructions on what to do based on the country. Ztorg also had variants that tried multiple known exploits one after another until it managed to grant itself root

privileges in order to give unlimited access to the device and making it increasingly hard to remove. Svpeng is a banking trojan that performs like any generic banking trojan does. It overlays itself on top of the mobile banking applications in order to trick the user into giving their banking details to the attacker. The reason why it was so common in 2016 is because Svpeng was spread through a unique exploit in Google's AdSense advertising network. When the user would visit a website that was using AdSense to deliver its ads, some of the ads had malicious JavaScript code that would automatically download the malware apk, if the user then installs the apk, they will get infected.

The notable trends of 2017 for Android malware were the increase in malware capable of granting itself root privileges and the increase in ransomware, but even though the amount of ransomware grew compared to 2016 the overall amount of malware fell by 1,5 times compared to 2016 [11]. The same decrease for overall attacks also applied to rooting malware, which can be attributed to the increase in security features in the newer versions of Android and the decreasing number of devices running earlier versions of Android as portrayed in Figure 4, which contained more well-known kernel exploits for obtaining root access such as Towelroot and Pingpongroot. Google also patched a well-known privilege escalation vulnerability and encouraged all users to install the latest security updates once they become available [12].

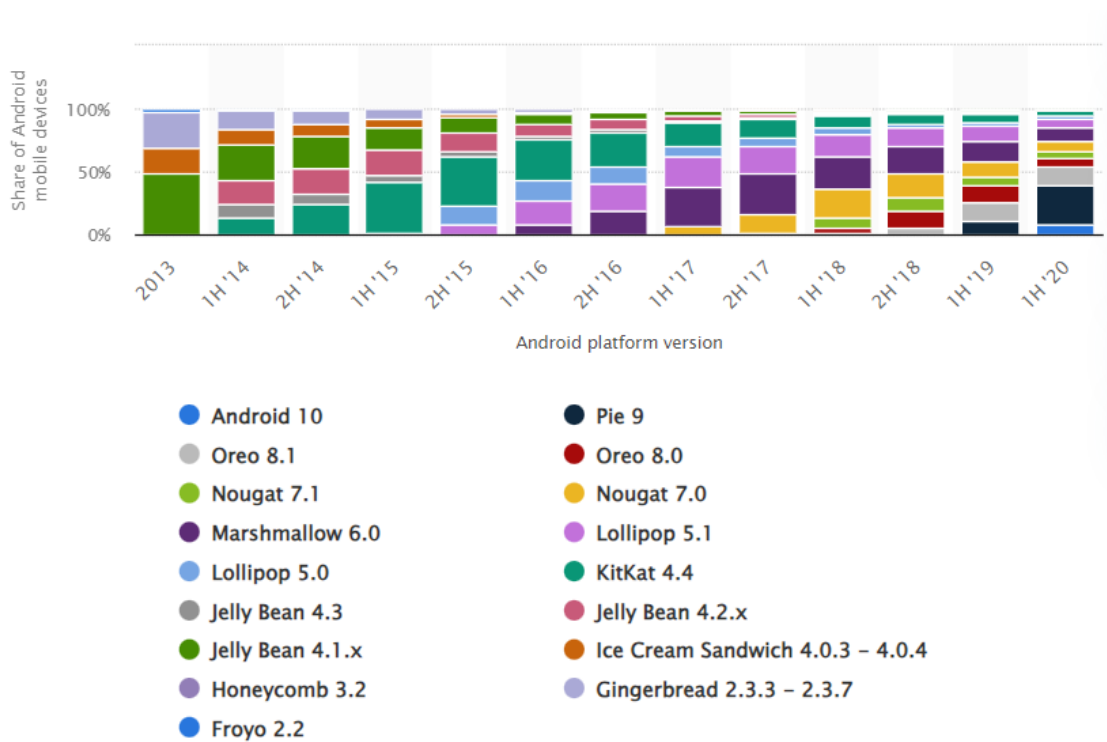


Figure 4: Android operating system version share worldwide from 2013 to 2020 [13].

2018 saw a new threat in the form of Click Fraud malware [14]. Click Fraud malware doesn't cause harm to the user directly since it only clicks ads in the background without user knowing about it, but it does generate financial gains to the distributor of the malware and hurts the advertisers since they are the ones left holding the bill. There was also a sharp increase in banking trojans in 2018, the amount of banking trojans grew by 60% compared to the previous year [15]. One of them being Asacub that spread through social engineering via SMS and once it managed to get a user to download it, it would attempt to propagate itself by sending a copy of itself to all the contacts that were saved on the user's device.

2019 saw an overall decrease in the number of total threats compared to 2018 [16]. Banking malware evolved from mimicking a banking application to steal credentials to instead using Android's accessibility features to perform actions on behalf of the user once the banking application has been logged into. This allows the malware to autonomously empty the victims' bank account without any outside input. Notable banking malware called Cerberus started spreading in the summer of 2019 [17]. It was capable of redirecting calls, reading messages to obtain OTP codes and 2FA codes and mimic banking applications with overlays [18]. Ransomware also grew compared to 2018 but had a sharp decrease throughout the year. The fourth quarter had 7 times less new installation packages compared to the first quarter [29].

Covid-19 played a big role in the 2020 malware landscape, with a lot of malwares hiding behind popular keywords such as covid and corona. Adware saw a large increase in its market share of the malware landscape, its share almost tripled compared to 2019 [19], but the number of unique users infected with adware decreased slightly. A probable cause for this is the increased number of apps containing aggressive advertising libraries in the Google Play store. These adware applications would usually wait a set period of time before displaying full screen ad banners even when not using the application [20]. Banker trojans also saw an increase in their market share in 2020. One of the reasons for this is the publishing of the source code for the malware Cerberus [17]. A strain of Cerberus was spread through Coronavirus-related applications [21].

2021 saw an overall decrease in malicious installation packages with the number of packages decreasing by 39% [22]. The malicious applications have to keep evolving to keep up with the ever-evolving security features in Android, this decreases the amount of people who are able to develop malware for Android. One major malware that started to spread in 2021 was FluBot. It was a Banking Trojan that was able to overlay itself on top of banking apps and cryptocurrency apps to steal the user credentials. It spread via

text messages and was able to access the contacts of the infected device to help with the spreading.

The overall decreasing trend of malware continued in 2022. The total number of package installers had halved in 2022 compared to 2021 [23]. Banking trojans market share quintupled compared to 2021, even though Europol managed to shut down the C&C server for the widely spread FluBot.

2023 saw a slight decrease in new unique malicious installation packages and the threat landscape stayed similar to 2022 with one exception of adware doubling its market share [24].

3. ATTACK SURFACES

Android devices come in all shapes and sizes, which leads them to have large and complex set of attack surfaces. The largest and most often exploited attack surface is classified as remote, which means that the attacker doesn't need to have physical access to the system. Instead, the attacks are executed over a network such as the internet or Bluetooth. Attacks via this surface are great for attackers since they don't require physical access and increase the number of potential targets. One remote attack surface that is specific to most Android devices is cellular communication via Short Message Service (SMS) and Multimedia Messaging Service (MMS).

Devices running Android as their operating system are usually handheld devices such as smartphones. Smartphones offer a wider array of attack surfaces than a typical computer does since smartphones have more external connection possibilities such as Bluetooth and SMS that a typical computer might not have.

In order for an Android app to access the system it must be granted permission by the user during its installation or after it, depending on the Android version. This also applies to anti-virus apps, which prevents them from introspecting other apps. Because of this they are mostly relying on signature-based solutions and are unable to do anything about newer malware that's signature hasn't been added to the libraries yet or ones that are performing signature spoofing. According to a study that tested 86 anti-virus products for Android with the most downloads and review on Google Play 30 of the 86 installed apps were unable to identify any of the installed malware and a significant amount of the applications were generating alarming amounts of network traffic of which some contained personal information [25].

4. EVOLUTION OF SECURITY FEATURES ON ANDROID

The newer versions of Android are robust and secure. Most of the exploits aren't targeted at Android, but to the device's firmware and drivers instead. Earlier versions of Android on the other hand contained a lot of vulnerabilities that were on average a lot more serious than vulnerabilities discovered on current versions on Android.

Android version 1.0 security consisted of five subsystems: lock screen PIN code, which protects against physical access; sandboxing applications, which prevented the application from accessing files outside of its own sandbox; specific permissions for each application; applications had to be signed with developer's key, which prevented installing newer versions of pre-existing application if it wasn't signed by the original developer; applications are written in Java, which protected against typical types of attacks that work against insecure languages such as buffer overflows or reusing freed memory. These security features didn't prevent vulnerabilities in other components of the OS, that were written in insecure languages such as C and C++. An example of a hack targeting one of these components was a hack to HTC Dream that targeted preinstalled service called Telnet that was running with root privileges [26]. Standard applications ran inside a sandbox and didn't have root privileges but had their own set of privileges that were listed in its manifest file. It didn't need to explicitly ask the user for the permissions, but instead the application got all the permissions listed in the manifest file when installed and user could only choose between installing or not installing the application. The privileges could include access to read text messages, display graphics over other applications, run in the background indefinitely and information about nearly all system events. The privileges weren't vulnerabilities per se, since Android wasn't wildly popular at the time, so they weren't used maliciously, but the relaxed application privileges allowed for easy development.

4.1 Permissions

Before Android version 4.3 the end-user had no control over application permissions other than to choose to not install the application. Android version 4.3 was the first version to introduce App Ops framework which allowed the user to manually disable any permissions they didn't want the application to use. App Ops was later removed by Google on Android version 4.4.2. Google stated that App Ops was there only for development purposes [27].

Android version 6.0 introduced the currently used runtime permissions that allowed the applications to check which permissions have already been granted to them and to request new permissions at runtime. The main problem with this approach was that it wasn't enforced but developers could specifically target the older directive in the build rules and the app would gain all permissions during installation. The system had to be implemented this way to maintain compatibility with older applications.

Android version 10 restricted the applications access to permissions further by adding soft and hard restrictions for permissions, which allowed to revoke permissions from applications created create for Android version 5.1 and earlier versions.

Android 11 updated the permissions by granting the user a choice to grant the application with a onetime permission that would get revoked once the application is minimized. Allowing users to deny access to the permissions was just part of the solution to prevent malicious usage of the permission system. Google also decided to restrict what permissions and APIs a third-party application is allowed to use. Starting from Android version 8 Google started to heavily restrict the number of unique identifiers that an application can obtain. Android version 8 prevented applications from accessing the devices serial number and the Android ID would be different for each application. Wi-fi MAC address is also randomized whenever the device scans for available networks to prevent tracking. Android version 8 also introduced Project Treble [28], that limited the permission vendor-provided hardware abstraction layers (HALs) were able to obtain. Project Treble was designed to decrease the severity of the bugs in HALs by reducing the number of permissions that each of them has. The goal of this was to reduce exploitation of HALs since they contained a lot of bugs compared to core kernel as seen in Figure 5.

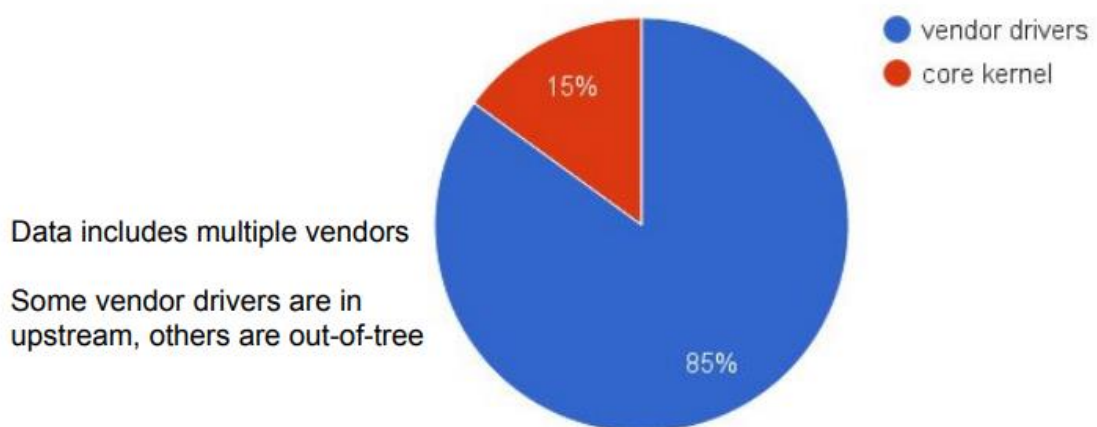


Figure 5: Android kernel bugs origins from 2014 to 2016 [29].

Before Project Treble every HAL was granted the same permissions that the other HALs had. This led to them having more permissions than any of them actually needed for them to work properly and if an attacker could exploit one of them that meant the attacker had access to a process that that had access and permissions to hardware that the process normally would not need to have in order to work properly. Project Treble isolated the HALs into their own sandboxes and they are only granted the permissions that they require to work properly.

Android version 9 went one step further and prevented the access to camera and microphone and every other sensor from being accessed from an application that is running in the background, which made it a lot harder for malicious applications to collect compromising information of the user. Android 9 also introduced a complete prohibition of the HTTP without TLS for all apps. Android 10 prevented applications that were running in the background from launching activities without having the explicit permission granted by the user. Android 11 prevents applications from creating their own application specific directories but provides each application with their own directory and applications can only access their own directory.

4.2 Protection against physical access

Earlier versions of Android didn't have much protection against malicious attacks if the attacker had physical access to the device. This changed when Android version 5 introduced full-disk encryption, which used a single key that was protected with the user's password [30]. The encryption was using AES-128 algorithm, and the key was protected with key-encryption-key that was generated from the users PIN or pattern or password that they were using for the device [31]. This feature had some major downsides such as increased battery consumption and core features such as alarms and phone calls not working until the user had given their password after a reboot.

Android version 7 introduced file-based encryption which became mandatory on Android 10 and later [32]. File-based encryption had major advantages over full-disk encryption, it allowed the introduction of Direct Boot which allowed the device to boot directly to the lock screen allowing alarms and other accessibility features to keep working even without user entering their password. File-based encryption split the storage in two major partitions, Credential Encrypted storage which stayed encrypted until the user had entered their password and Device Encrypted storage which was available even when the user had not entered their password. Android 5 also introduced the possibility of storing the

encryption keys in a Trusted Execution Environment. Different manufacturers have implemented their own versions of Trusted Execution Environments, such as Google's Titan M chip.

Android 4.4 introduced Verified Boot that establishes a chain of trust that starts from hardware and goes through every step of the boot and verifies the integrity and authenticity of each stage before giving them the execution rights [33]. This aims to prevent the attacker from tampering with the system. Android 8 improved upon this by adding roll-back protection which prevents the attacker from downgrading the system into an older version to introduce old exploits that could be used to gain access to the system.

5. CONCLUSION

The goal of this research was to get a general understanding of Android based malware and how it has changed over the years. The aim of the malware has clearly changed from basic methods such as premium-rate SMS theft with direct financial gain for the attacker in the early 2010s. The early 2010s the user had very limited control over application permissions.

In mid and late 2010s the data stored on the smartphones was a lot more personal and sensitive than it was before, so stealing it became a lot more lucrative than before and that could be seen in the shift towards stealing or disrupting it, in the forms of a spyware and ransomware. With the introduction of Android 4.3 in 2013 users finally got the access to more granular permission controls allowing them to choose what data the apps could access. Google also introduced regular security updates to patch known vulnerabilities and full disk encryption to prevent the access to users' data when the device is lost or stolen. Biometric authentication also became standard features for unlocking the device and Google released Play Protect to actively scan installed applications for harmful behavior.

The current modern malware is targeting specific users or organizations such as banking malware targeting specific banks. The improved computing power of mobile devices has also made them potential targets for cryptojacking. Current Android devices are more secure than ever, but the malware keeps evolving alongside the security.

SOURCES

- [1] "A Brief History of Android: Founding, Evolution & Industry Impact," A Brief History of Android: Founding, Evolution & Industry Impact. Accessed: Jul. 22, 2024. [Online]. Available: <https://codesubmit.io/blog/history-of-android-operating-system/>
- [2] "Smartphone OS market share forecast 2014-2023," Statista. Accessed: Jul. 22, 2024. [Online]. Available: <https://www.statista.com/statistics/272307/market-share-forecast-for-smartphone-operating-systems/>
- [3] Kaspersky Lab and INTERPOL Joint Report, "Mobile Cyber Threats," Oct. 2014. [Online]. Available: <https://media.kaspersky.com/pdf/Kaspersky-Lab-KSN-Report-mobile-cyberthreats-web.pdf>
- [4] D. R. Thomas, A. R. Beresford, and A. Rice, "Security Metrics for the Android Ecosystem," in *Proceedings of the 5th Annual ACM CCS Workshop on Security and Privacy in Smartphones and Mobile Devices*, Denver Colorado USA: ACM, Oct. 2015, pp. 87–98. doi: 10.1145/2808117.2808118.
- [5] "First SMS Trojan for Android." Accessed: Jul. 22, 2024. [Online]. Available: <https://securelist.com/first-sms-trojan-for-android/29731/>
- [6] K. Dunham, S. Hartman, M. Quintans, J. A. Morales, and T. Strazzere, *Android Malware and Analysis*. CRC Press, 2014.
- [7] F. Ruiz, "'FakeInstaller' Leads the Attack on Android Phones," McAfee Blog. Accessed: Jul. 22, 2024. [Online]. Available: <https://www.mcafee.com/blogs/mobile-security/fakeinstaller-leads-the-attack-on-android-phones/>
- [8] Kaspersky Lab, "Ransomware in 2014-2016," Jun. 2016. [Online]. Available: https://media.kasperskycontenthub.com/wp-content/uploads/sites/43/2018/03/07190822/KSN_Report_Ransomware_2014-2016_final_ENG.pdf
- [9] R. Lipovský, L. Štefanko, and G. Braniša, "The Rise of Android Ransomware." [Online]. Available: https://web-assets.esetstatic.com/wls/2016/02/Rise_of_Android_Ransomware.pdf
- [10] "Ztorg: from rooting to SMS." Accessed: Jul. 22, 2024. [Online]. Available: <https://securelist.com/ztorg-from-rooting-to-sms/78775/>
- [11] "Mobile malware evolution 2017." Accessed: Jul. 22, 2024. [Online]. Available: <https://securelist.com/mobile-malware-review-2017/84139/>
- [12] "Android Security Advisory — 2016-03-18," Android Open Source Project. Accessed: Jul. 22, 2024. [Online]. Available: <https://source.android.com/docs/security/bulletin/advisory/2016-03-18>
- [13] "Android version market share 2020," Statista. Accessed: Jul. 22, 2024. [Online]. Available: <https://www.statista.com/statistics/271774/share-of-android-platforms-on-mobile-devices-with-android-os/>
- [14] Google, "Android Security & Privacy 2018 Year In Review," Mar. 2019. [Online]. Available: https://source.android.com/docs/security/overview/reports/Google_Android_Security_2018_Report_Final.pdf
- [15] "The number of mobile malware attacks doubles in 2018, as cybercriminals sharpen their distribution strategies," www.kaspersky.com. Accessed: Jul. 22, 2024. [Online]. Available: https://www.kaspersky.com/about/press-releases/2019_the-number-of-mobile-malware-attacks-doubles-in-2018-as-cybercriminals-sharpen-their-distribution-strategies
- [16] "Mobile malware evolution 2019." Accessed: Jul. 22, 2024. [Online]. Available: <https://securelist.com/mobile-malware-evolution-2019/96280/>
- [17] "The rise of Cerberus: Android banking malware is available for free in underground forums," www.kaspersky.com. Accessed: Jul. 22, 2024. [Online]. Available: https://www.kaspersky.com/about/press-releases/2020_the-rise-of-cerberus
- [18] "Cerberus Banking Trojan (Android)." Accessed: Jul. 22, 2024. [Online]. Available: <https://www.pcrisk.com/removal-guides/17387-cerberus-banking-trojan-android>
- [19] "Mobile malware evolution 2020." Accessed: Jul. 22, 2024. [Online]. Available: <https://securelist.com/mobile-malware-evolution-2020/101029/>
- [20] "Aggressive in-app advertising in Android." Accessed: Jul. 22, 2024. [Online]. Available: <https://securelist.com/in-app-advertising-in-android/97065/>

- [21] et al, "COVID-19 goes mobile: Coronavirus malicious applications discovered," Check Point Research. Accessed: Jul. 22, 2024. [Online]. Available: <https://research.checkpoint.com/2020/covid-19-goes-mobile-coronavirus-malicious-applications-discovered/>
- [22] "Mobile malware evolution 2021." Accessed: Jul. 22, 2024. [Online]. Available: <https://securelist.com/mobile-malware-evolution-2021/105876/>
- [23] "Mobile cyberthreat report for 2022." Accessed: Jul. 22, 2024. [Online]. Available: <https://securelist.com/mobile-threat-report-2022/108844/>
- [24] "Kaspersky's report on mobile threats in 2023." Accessed: Jul. 23, 2024. [Online]. Available: <https://securelist.com/mobile-malware-report-2023/111964/>
- [25] W. Yao *et al.*, "Security Apps under the Looking Glass: An Empirical Analysis of Android Security Apps," in *2020 IEEE 45th Conference on Local Computer Networks (LCN)*, Nov. 2020, pp. 381–384. doi: 10.1109/LCN48667.2020.9314784.
- [26] "Android security: evolution from version 1 to version 11 – HackMag." Accessed: Jul. 22, 2024. [Online]. Available: <https://hackmag.com/mobile/android-privacy-history/>
- [27] "App Ops - what you need to know," Android Authority. Accessed: Jul. 22, 2024. [Online]. Available: <https://www.androidauthority.com/app-ops-need-know-324850/>
- [28] "Architecture overview," Android Open Source Project. Accessed: Jul. 22, 2024. [Online]. Available: <https://source.android.com/docs/core/architecture>
- [29] J. V. Stoep, "Android: protecting the kernel," Aug. 2016. [Online]. Available: <https://events.static.linuxfound.org/sites/events/files/slides/Android-%20protecting%20the%20kernel.pdf>
- [30] "Encryption," Android Open Source Project. Accessed: Jul. 22, 2024. [Online]. Available: <https://source.android.com/docs/security/features/encryption>
- [31] "Full-Disk Encryption," Android Open Source Project. Accessed: Jul. 22, 2024. [Online]. Available: <https://source.android.com/docs/security/features/encryption/full-disk>
- [32] "File-Based Encryption," Android Open Source Project. Accessed: Jul. 22, 2024. [Online]. Available: <https://source.android.com/docs/security/features/encryption/file-based>
- [33] "Verified Boot," Android Open Source Project. Accessed: Jul. 22, 2024. [Online]. Available: <https://source.android.com/docs/security/features/verifiedboot>