IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

ISO 3297:2007 Certified ⁜ Impact Factor 7.39 ⁜ Vol. 11, Issue 6, June 2022

DOI: 10.17148/IJARCCE.2022.116129

Application's Auto-Login Function Security Testing Using Virtualization at the OS Level for Android

Aditi Chatterjee

Atria Institute of Technology, Bengaluru, Karnataka, India

Abstract: This work is originally present in "**App's Auto-Login Function Security Testing via Android OS-Level Virtualization**".Most mobile apps provide the automated login option to improve user experience despite the small keyboard's limitations. Users can avoid the hassle of having to type their ID and password again whenever an app is running in the foreground. The so-called "data-clone attack" can be launched by copying the locally stored, auto-login dependent data and installing it on the attackers' smartphones, which allows the attackers to exceed the allowed number of login devices and secretly connect into the victim's account. Verifying the consistency of device-specific properties is a natural countermeasure. The programme will block the auto-login feature and hence guard against data-clone attempts as long as the new device displays distinct device fingerprints from the old one. In this article,with VP Droid, security analysts can alter several device artefacts in a virtual phone without using user-level API hooks, including CPU model, Android ID, and phone number. The isolation method of VPDroid makes sure that user-mode apps in the virtual phone cannot identify differences between devices. We simulate data-clone attacks with 234 of the most popular Android apps using VPDroid in order to evaluate how vulnerable Android apps are to these assaults. Our tests on five distinct virtual phone settings demonstrate that all evaluated apps that do device-consistency checks, such Twitter, WeChat, and PayPal, may be tricked by VPDroid's device attribute customisation. As a zero-day vulnerability, our report has been verified by 19 vendors.

INTRODUCTION

The apps that operate on the Android operating system are continuously updated to meet the rapidly expanding demand of smartphone users thanks to the successful development of the Android system and mobile networks [1], [2]. Apps are currently handling numerous crucial jobs, such as social networking [3], GPS navigation [4], IoT device remote control [5], and mobile payment [6], in addition to the conventional functionalities like communication and entertainment. The smartphone cyber arms race between overcoming user authentication inevitably results in the storage of enormous volumes of private data (such as user passwords). its "76% of network breaches leveraged weak or stolen credentials," claims Verizon's 2019 data breach investigation report [7].

A significant quantity of literature was also produced by user accounts. We focus on high-impact assaults and discover that their underlying causes are either basic design faults or system vulnerabilities. Man-in-the-middle (MitM) attacks use the password reset vulnerability to break a mobile user's account password [10], [11], and the recently developed app-virtualization technique disobeys Android's unique user ID mechanism, making guest apps vulnerable to the "shared-everything threat" [12]-[14]. These vulnerabilities are similar to the serious security flaws in Android password manager apps that allow attackers to access the stored credentials [8], [9]. In this study, we specifically address the client-side tampering vulnerability associated with auto-login features in mobile apps [15], which poses a security risk.A significant quantity of literature was also produced by user accounts. We focus on high-impact assaults and discover that their underlying causes are either basic design faults or system vulnerabilities. Man-in-the-middle (MitM) attacks use the password reset vulnerability to break a mobile user's account password [10], [11], and the recently developed app-virtualization technique disobeys Android's unique user ID mechanism, making guest apps vulnerable to the "shared-everything threat" [12]-[14]. These vulnerabilities are similar to the serious security flaws in Android password manager apps that allow attackers to access the stored credentials [8], [9]. In this study, we specifically address the client-side tampering vulnerability associated with auto-login features in mobile apps [15], which poses a security risk.

The majority of mobile apps now available allow automatic login to enhance the user experience. It saves the trouble of having to repeatedly type the user ID and password on a tiny keyboard when using the programme. Because of how convenient the autologin option is, mobile users have grown accustomed to utilising it. However, the login-device number limit can be bypassed and the user's account can be accessed without raising any red flags if an attacker steals

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified ⁜ Impact Factor 7.39 ⁜ Vol. 11, Issue 6, June 2022
DOI: 10.17148/IJARCCE.2022.116129

the auto-login dependent data from the user's device and replaces it with the user's data. This is known as a "data-clone attack."

In this article, a token or user identification is what is meant by "credential" or the auto-login dependent data. Initial studies on this danger have been conducted [16–18], however they are all restricted to victim identity theft on a single scale.Additionally, they overlooked a crucial fact: more and more apps verify device consistency to prevent client-side tampering; if they find any device-specific discrepancies, their auto-login functionality will be deactivated. We provide a fresh assault strategy that allows non-rooted devices to surpass the paying-subscriber cap. The income mechanisms of several subscription-based programmes [19] including Netflix, Amazon Prime Video, and Apple Music limit the amount of times a user may log in simultaneously from various devices. For instance, Netflix's Basic package only permits streaming HD content to one device at a time. A fraudster first pays the Basic plan cost for Netflix. Then he launches a data-clone assault using an OEM-made phone clone programme [20]. Without rooting devices, the OEM-made phone clone programme may replicate confidential data across identical OEM phones. The scammer makes use of the Premium plan service in this way by simultaneously streaming HD content to numerous displays.

RELATED WORKS

We begin by talking about the security risk of automated using a mobile app's login. Exploitation of Android in existing works. The auto-login features of applications are constrained. We then introduce Apps for cloned OEM phones, which we interpret as a vector to copy sensitive information. Last but not least, we go through Android's fundamentals. Virtualization at the OS level, the basis of VPDroid is active on a phone's front screen, and users regularly swap between background apps. It is mostly requiring users to repeatedly input their ID and password is inconvenient using an app. Most mobile apps aim to enhance the user experience, Apps by default support the automatic login functionality. As a users simply need to provide their ID and password at their initial login moment. After that, users may easily access the app without having to enter their ID and password again. All 234 of our tested apps' auto-login features continue to function even when we shutdown and resume their processes.

The majority of auto-login features save user credentials locally and carry out app server authentication automatically. The security tokens that are used to verify a user's identification with the app server are known as user credential data. The software maintains user credential information locally for use in future verification processes once the user initially enters ID and password to begin the authentication process. To store app-private data on Android, there are four alternatives [23]: There are four types of file storage: internal, external, shared preferences, and databases. Key value pairs like Shared Preferences or structured data in a SQLite database are frequently used to store user credentials.

Most auto-login features save user credential information locally and finish the app server authentication procedure automatically. Security tokens known as user credential data are employed by the app server to verify a user's identification. The software locally maintains user credential data for future verification when the user initially enters ID and password to complete the authentication procedure. Android offers the following four ways to preserve app-private data [23]: Internal file storage, external file storage, shared preferences, and databases round out the list. In most cases, user credentials are kept either as key-value pairs in Shared Preferences or as structured data in a SQLite database.

Paying-Subscriber Fraud & Device-Consistency Check

In this section, we first describe the unique benefit of exploiting auto-login functions to bypass user authentication. Next, we introduce a new data-clone attack model: paying subscriber fraud. Then, we perform data-clone attacks with 234 most-downloaded apps. Our results show that data-clone attacks are a real threat to both the app economy and user privacy, especially when skilled attackers are able to simulate a Hi-Fi smartphone environment.

The special advantage of the data-clone attack is that it is far more covert than the scenario where the attacker may intercept the user's password. The server-side detection of the login-device number limit would be triggered by the user entering their ID and password during the login process. Many apps restrict simultaneous logins from more than one device to a single user. This implies that by entering the user ID and password, neither the legitimate user nor the attacker can log in at the same time. For instance, a warning notification will appear on the attacker's phone when they use the victim's ID and password to sign in to the messaging service KakaoTalk from a separate phone, as seen in Figure 3. (a).

This type of identity theft is seen in our sample film at https://youtu.be/cs6LxbDGPXU. The attacker cannot get in to KakaoTalk using the same user ID and password when the legitimate user is active online. Additionally, KakaoTalk has

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

ISO 3297:2007 Certified ⁜ Impact Factor 7.39 ⁜ Vol. 11, Issue 6, June 2022

DOI: 10.17148/IJARCCE.2022.116129

the ability to recognise a new device. After we transfer the data in the "/data/data/KakaoTalk/" directory to a new device, it disables the auto-login (see Figure 3(b)). Contrarily, we tweak VPDroid's VP using the previous phone's profiles before launching a data-clone assault. We discover that both the victim and the attacker may be online simultaneously without generating any red flags.Our main finding is that calculating the number of login devices is unaffected by successive auto-login attempts from the same device, which gives us a backdoor through which to exceed the allowed number of login devices. As a result, confidential user information will be compromised covertly. If a social messaging app is penetrated in this fashion, the adversary can send messages while posing as the victim and can even access conversation history in real time.

VPDroid System Design

To aid in the account security testing of apps, we create a lightweight Android OS-level virtualization architecture called VPDroid. Security researchers may use VPDroid to customise various device attributes in accordance with the profiles of a target phone before starting up a virtual phone (VP) environment that closely resembles the target device. Additionally, our technology enables modification of device attributes without interfering with the host device's regular activities. VPDroid must satisfy two prerequisites (Reql & Req2) in order to trick the cloned applications into believing the smartphone has not been altered:

1) Requirement: The VP always has direct access to hardware; this approach offers a high-performance, nearly native virtual environment.

2) Requirement 2: User-mode apps operating in the VP must be undetectable to changes in device, which necessitates that our virtualization and device-attribute customisation features be hidden from user-mode apps.

New User-level Device Virtualization

Rewriting every kernel driver for Cells's out-of-date device virtualization systems is difficult and prone to errors. Particularly, some hardware makers provide closed-source, proprietary software stacks. It would be challenging, if not impossible, to virtualize them in the kernel without the help of the hardware vendor. Without leaving any in-guest virtualization components, user-level virtualization on VPDroid provides a versatile and portable option. There are two ways to virtualize various devices in our mechanism.

Make the VP's Device Attributes your own

We take it a step further and alter the VP's device properties based on the new virtualization framework at the Android OS level. The workflow is shown in Figure 7. Users of VPDroid supply a configuration file called "build.VPDroid.prop" in advance, which holds attributes unique to their particular device in the fopairs. The qualities of the Android system, the properties of user-level virtualized devices, and the properties of kernel-level virtualized devices are the three categories into which we divide these key-value pairs. There are several ways to customise each category. In addition, we use many namespaces to separate our modification.

VPDroid Assessment

The VP pictures are produced on a PC and sent over USB to the host device. We give VPDroid users a control centre app so they can quickly switch between the VP and the host system. A user follows these three steps to launch a fresh VP to represent a different device: Exiting the initial VP; changing the existing "build.VPDroid.prop" configuration file with a new one; and establishing a new VP using the control centre app. Performance tests are initially shown in this part to demonstrate how VPDroid shows native performance.In our second experiment, we assess VPDroid's capacity to customise device attributes by using the data-clone attack as a case study. Our findings demonstrate that VPDroid significantly raises the likelihood that a data-clone attack would succeed.

DISCUSSION

The most important safeguard against data-clone attacks is the rule that no user credentials should ever be stored in local files by a mobile app. This tactic, which sacrifices usability, only works for important programmes that don't need regular user interactions. Utilizing a Trusted Execution Environment (such as ARM TrustZone) to encrypt/decrypt user credential data prior to usage is another option. Data-clone attacks cannot replicate the decryption key to another device together with encrypted user credential data since it is kept in the TrustZone environment. As a result, the server will be

International Journal of Advanced Research in Computer and Communication Engineering

ISO 3297:2007 Certified ⁜ Impact Factor 7.39 ⁜ Vol. 11, Issue 6, June 2022

DOI: 10.17148/IJARCCE.2022.116129

unable to validate the login credentials. The most recent articles, TruApp [38] and IMVisor [39], investigate whether TrustZone may be used to preserve app integrity and sensitive data, while Rubinov et alwork .'s divides a crucial system component.

Although it can significantly raise the cost, we do not presume that identifying the existence of VPDroid is necessarily impossible.If a VP programme has root access, it can determine which footprint of our device virtualization at the user level. For instance, the Java telephony libraries of VP do not communicate with VP's RilD. The auto-login feature might verify the accuracy of some unknown gadget characteristics that we don't cover, and it remains a challenge to locate all of them.

CONCLUSION

In this study, we describe, investigate, and assess the client-side device-consistency check defence against the data clone attack. Our technological contribution entails the creation of a transparent framework for customising device attributes via virtualization at the Android OS level. Our analysis of the most popular applications shows that the data-clone attack is a real danger that may do significant harm to user privacy and the app economy. We hope that our research and the open-source VPDroid tool will assist researchers in redesigning the auto-login features of applications and assessing their device artefact detecting abilities.

REFERENCES

[1] Anthony Canino, Yu David Liu, and Hidehiko Masuhara. Stochastic Energy Optimization for Mobile GPS Applications. In Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE'18), 2018.
[2] Sascha Segan. Fastest Mobile Networks 2019. https://www.pcmag.com/ Fastest-Mobile-Networks, June 2019.
[3] Monica S. Lam. Omlet: A Revolution against Big-Brother Social Networks (Invited Talk). In Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering (FSE'14), 2014.
[4] Mark Sherman. An Introduction to Mobile Payments: Market Drivers, Applications, and Inhibitors. In Proceedings of the 1st International Conference on Mobile Software Engineering and Systems, 2014.
[5] Stephan Huber, Siegfried Rasthofer, and Steven Arzt. Extracting All Your Secrets: Vulnerabilities in Android Password Managers. HackInTheBox 2017, 2017.