2016 13th IEEE Annual Consumer Communications & Networking Conference (CCNC)

A Novel Storage Covert Channel on Wearable Devices Using Status Bar Notifications

Kyle Denney, A. Selcuk Uluagac, Kemal Akkaya, and Shekhar Bhansali
Dept. of Electrical & Computer Engineering, Florida International University, Miami, FL, 33174 USA
kdenney@fiu, auluagac@fiu.edu, kakkaya@fiu.edu, sbhansa@fiu.edu

***Abstract*—Covert channels have been used as a means to circumvent security measures and send sensitive data undetectable to an onlooker. Many covert channels in Android systems have been documented utilizing various system resources or settings available to the entire system. Nonetheless, this paper introduces a new storage covert channel on the emerging field of wearables that sends data to other applications, or even to other nearby devices, through the use of *notifications* that are normally displayed on the status bar of an Android device. In this paper, we present the design of our ongoing work for this covert channel using Android-based wearable devices. Furthermore, we evaluate the performance of this covert channel using real equipment. Our evaluation demonstrates the functionality and feasibility of the proposed covert channel.**

***Keywords*—Android, covert channel, security, mobile, wearables covert channels, Internet-of-Things covert channel**

I. Introduction

Recently, Internet-of-Things (IoT) devices, have become increasingly popular. It is predicted that by 2020, there will be 50 to 100 billion devices connected to the Internet [6], [5], forming a massive IoT. Among these devices, a considerable number of them will be the wearable devices that can be carried by individuals such as glasses, watches, sensors, etc. By 2019, it is estimated that 1 in 4 smartphone owners will also be using a wearable device [4]. With new technology comes new security risks, wearables are no exception.

Covert channels [3], [2], [10] are used to subvert security measures and steal sensitive information that is undetectable to an onlooker. In a general sense, covert channels utilize normal system resources that are available universally to processes without needing an extra resource for their malicious purposes. For instance, in a *storage covert channel* [10] one malicious process can change or alter a value on the system resource using a predetermined codebook. Another malicious process can then read the values that were changed and interpret the coded message that was sent. Two or more processes can use this method to work surreptitiously in tandem to send data to a malicious process. For example, two applications can work together to steal contact information from a smart phone. The covert sender converts the contacts into string data and transfers the information to the covert receiver. The receiver can then send the data to a malicious server for an attacker to obtain it.

In the Android operating system, many covert channels have been discovered [3]. They use resources such as changing volume settings, brightness settings, vibrations [1], or even CPU timings. *This paper aims to introduce a new storage covert channel that works by utilizing notifications in the status bar in wearable devices*. Specifically, when an application needs to alert the user to new information, a notification is created and displayed in the status bar on the smart device. If a user has a wearable synced to their smart device, notifications are shared between both devices.

The covert channel introduced in this work uses these notifications to send a hidden message to a covert receiver. The covert sender application on a smart phone or device creates a series of notifications that conform to a preset codebook. The covert receiver application then reads the notifications as they appear on the smartwatch paired with the device and interprets the coded message that was sent. Note that due to notifications being shared across multiple devices (e.g., smartwatch, smart phone, tablet), the two applications need not be on the same device for the covert channel to operate. In this paper, we introduce the design of this novel notification-based storage channel for Android wearable devices using real smartwatches. To the best of our knowledge, this is the first work in this space. Furthermore, we analyze the performance of this wearable covert channel, demonstrating its feasibility using real equipment. Our results show that the covert channel can achieve 598 bits of throughput.

The rest of the paper is organized as follows. Section II and III give details on how the covert channel on wearables using notifications works. Section IV presents the performance evaluation of the covert channel. Section V discusses the relevant work. Section VI concludes this paper.

II. Using Wearable Notifications as a Covert Channel

The covert channel that is introduced in this paper uses Android Wearable notifications to send information between applications on the same device, or even across

978-1-4673-9292-1/16/$31.00 ©2016 IEEE

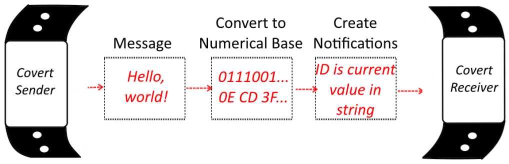


Fig. 1: Design of the covert channel

multiple devices as articulated in this section. A covert sender application creates notifications using varying notification ID numbers as the coded message. A covert receiving application reads the incoming notifications, detects the ID numbers of each notification, and interprets the covert message (Figure 2). Due to using the notification ID to transmit the covert message, the text of the notification can be disguised to increase the secrecy of the communication.

Functions that are utilized in this covert channel protocol are the *notify(int ID, notification Notification)* and *cancel(int ID)* functions. *Notify* allows the application to call a pre-declared notification and use a specific ID value for the notification. The way the function is structured allows for a notification to be fetched multiple times by using different ID numbers while still using the same notification. The *cancel* function clears a notification with a specified ID off of the status bar. If the *notify* function is called and is immediately followed by the *cancel* function, the notification is never displayed to the user. Even though the notification is not displayed, the notification is still detectable in the system.

When transmitting data, the covet sender application utilizes three separate notifications: (1) a notification to signal the start of transmission, (2) a notification that represents the data, and (3) a notification to signal the end of the transmission. When data is ready to be transmitted, the starting notification is sent out with a predetermined starting ID. Then, the data notification is called with varying ID values that correlate to the data values in the transmission. When the data has been sent, the termination notification is broadcasted and communication between the applications is halted.

In order to function properly, the covert receiving application requires the use of the BIND_NOTIFICATION_LISTENER_SERVICE permission. This permission allows the application to be running constantly and be allowed to read notifications as they are broadcasted on the system. With this, the covert receiver can detect when the sending application transmits a covert code. When the starting ID is transmitted, the receiver starts collecting data values from the sender. When the finishing ID is detected, the data is collected and converted back into the original message. From there, the covert receiver sends the data to a server to be collected by an attacker.

Fig. 2: Devices used for testbed - left: Samsung Galaxy S5; right: Sony SmartWatch 3

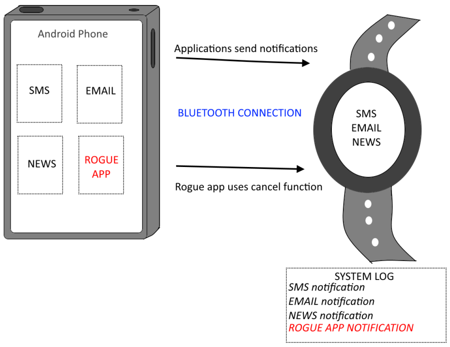


Fig. 3: Sync design between the Phone and Wearable.

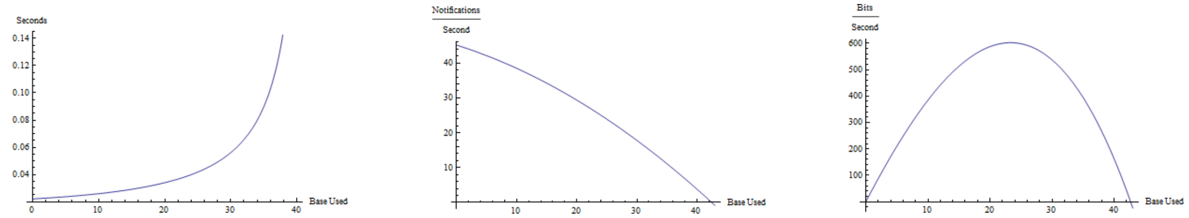
III. Wearable Covert Channel Across Android Devices

With the introduction of Android Wear, wearable devices like a smartwatch are able to sync with a smartphone and share notifications. Since all notifications are shared between both devices, the covert channel introduced in this paper can operate over multiple devices. The sending application can be on one device and the receiving application on another.

When a notification is made on a smartphone, it is sent to the Android Wear application which then sends the notification to the synced wearable. The overall architecture is shown in Figure 3. Notifications from all applications on the smart device are sent to the wearable via the Android Wear application using a Bluetooth connection. These notifications are immediately displayed on the wearable's screen.

However, for a rogue application which is trying to send data through a covert channel this is not the case. The notifications can be sent in the same way via the Android Wear app as shown in the figure but after each transmission a *cancel()* function can be sent to prevent the notifications to appear on the wearable's screen. While the notifications will appear in the system log, they will not appear on the actual screen to the user to keep the channel further surreptitious.

Since all of the rogue notifications transmitted are cancelled immediately after being called, the average user will never see the transmission occurring. The only way for a 'warden' to detect the covert channel is by going into the system log and seeing the notifications from



(a) Individual notification time (t). (b) Notifications/second (y/s). (c) Bits/second (b/s).

Fig. 4: Performance of the wearables covert channel with Sony Smartwatch 3.

there. Even so, the only information that is displayed is the text that is provided in the notification, which can easily be disguised as something harmless. Without the BIND_NOTIFICATION_LISTENER_SERVICE permission, the warden would not be able to detect the message that is being sent between the devices. Overall, this covert channel is discrete since there is no specific text or content exchanged between the covert sender and the receiver.

IV. Performance Evaluation

To test the functionality and the feasibility of the proposed wearable covert channel, a Samsung Galaxy S5 was used as the smart device and a Sony SmartWatch 3 was used as the wearable. The Galaxy phone holds a 2.5GHz quad-core processor with 2GB of RAM. The SmartWatch has a 1.2Ghz quad-core processor and 512MB of RAM.

The proposed covert channel is unique given the fact that it has effectively an infinite ways the coded message can be sent. The most obvious way is by changing the base of the message as mentioned earlier. Indeed, a covert sender can cycle from different bases to even make the channel more clandestine to evade and decrease the difficulty of any detection. In this section, the efficiency of the covert channel is discussed by analyzing how discretely the message is sent. We also analyze the transmission time for the messages sent.

To increase the efficiency of the message, more complex bases can be used to send the message. The downside to increasing the base is that it takes a longer runtime to send the notifications due to a larger number of possible ID values the message can utilize. To evaluate this, varying bases were tested by sending the message, ”Hello, world!” over the wearable covert channel. By calculating the runtime of the entire transmission and dividing the time by the number of notifications sent, the average time for an individual notification can be obtained:

$$t = \frac{\omega}{z} \tag{1}$$

where t is the individual notification time in seconds, ω is the total runtime, and z is the total amount of notifications that were sent. From there, the amount of notifications sent in one second can be evaluated:

$$y/s = t^{-1} \tag{2}$$

where y/s is the amount of notifications per second. With the amount of notifications/second, the throughput of the transmission is calculated with:

$$b/s = (y/s) * x \tag{3}$$

where b/s is the bitrate and x is the base that is used.

Figure 5 shows how each equation is altered as the base increases. The individual notification rate slowly increases due to the higher amount of possible values each notification can take (Figure 5a). By the time the base is 42, the time for a single notification to be sent reaches 1 second. Due to the increasing time for a notification to be made, the amount of notifications per second decreases with higher bases (Figure 5b). The bandwidth is afftected by both the notification rate and the base (Figure 5c). As the base increases, more bits can be sent in a single notification. The maximum bandwidth is reached at a base of 23 with a throughput of 598 b/s. Any base higher than 23 becomes too complex and the bandwidth starts to decrease.

V. Related Work

After Lampson’s initial seminal paper on covert channels [9], many covert channels [10], [2], [3] have been documented over the years. Covert channels have been detected on a single computer or smart device, or even over the network using networking protocols [10]. In this section, work relevant covert channels, specifically on mobile devices, are discussed.

Wade and Yang were among the first to document that new approaches needed to be developed in order for covert channels to work on mobile devices [8]. Due to the sandboxing methods that Android encorporates with applications, they proposed that a covert channel must find a way around the sandboxing in order to communicate effectively.

Wade and Yang described in another paper how to make network-based covert channels using delays in network transmissions to make a timing-based covert channel [8], [7]. The channel in question sends a video stream to a remote server. By incorporating delays in the packets sent to the server, a covert channel was made.

Al-Haiqi et al. proposed a covert channel using the vibration setting on phones [1]. Since all applications had permission to alter the vibrations, the applications could communicate by using a series of pulses. These pulses, when set to a timer, can become a timing-based covert channel.

Moreover, Chandra et al. give a comprehensive list of viable covert channels on the Android operating system [3]. Included are audio settings, screen brightness, etc. Adding to this list, they developed covert channels that utilizes the calling component in a smart phone. Using this new covert channel in tandem with other known channels, they produced a covert channel protocol that allowed a throughput of 30kb/s.

Our work is different from earlier work as we use notifications to create a storage covert channel protocol on Android-based wearable devices. To the best of our knowledge, this is the first work in this space to discuss a covert channel for wearables.

VI. Conclusion and Future Work

As more wearable consumers want notifications to be shared with across all of their devices, abuse of notifications for malicious purposes to establish covert communication channels will always be prevalent. On the Android marketplace currently, there are applications that allow for notifications to be sent across all the users' devices. These applications are given permission to view all notifications on a smart phone and interpret the information from them. If the application is untrustworthy, it can steal data from the user and send it to an offsite malicious receiver.

This paper aimed to demonstrate how a notification-based storage covert channel can be realized on wearables. With notifications, we designed and evaluated the performance of this wearable covert channel that can send hidden messages at a rate of up to 598 b/s using real Android-based wearable equipment (i.e., Sony Smartwatch). Furthermore, we explained how the covert channel can be used across multiple devices on the Android Wear platform. In the future, we will work on enhancing the capabilities of the covert channel.

VII. Acknowledgements

This work was partly supported by the US NSF Research Experiences for Undergraduates Sites Program Grant No: REU-CNS-1461119.

References

[1] Ahmed Al-Haiqi, Mahamod Ismail, and Rosdiadee Nordin. A new sensors-based covert channel on android. *The Scientific World*, 2014, 2014.

[2] Moreno Ambrosin, Mauro Conti, Paolo Gasti, and Gene Tsudik. Covert ephemeral communication in named data networking. In *Proceedings of the 9th ACM Symposium on Information, Computer and Communications Security*, ASIA CCS '14, 2014.

[3] Swarup Chandra, Zhiqiang Lin, Ashish Kundu, and Latifur Khan. Towards a systematic study of the covert channel attacks in smartphones. *SECURECOMM 2014*, 2014.

[4] Jonah Comstock. http://mobihealthnews.com/37543/pwc-1-in-5-americans-owns-a-wearable-1-in-10-wears-them-daily/. *Mobi Health News*, 2014.

[5] Peter F. Drucker. Internet of Things position paper on standardization for IoT technologies, Jan. 2015.

[6] Dave Evans. The Internet of Things: How the next evolution of the Internet is changing everything, Apr. 2011.

[7] W. Gasior and Li Yang. Exploring covert channel in android platform. In *Cyber Security (CyberSecurity), 2012 International Conference on*, Dec 2012.

[8] Wade C. Gasior and Li Yang. Network covert channels on the android platform. *Cyber Security and Information Intelligence Research 2011*, 2011.

[9] Butler W. Lampson. A note on the confinement problem. *Communications of the ACM*, 16, 1973.

[10] S.V. Radhakrishnan, A.S. Uluagac, and R. Beyah. Realizing an 802.11-based covert timing channel using off-the-shelf wireless cards. In *Global Communications Conference (GLOBECOM), 2013 IEEE*, pages 722–728, Dec 2013.

[11] Gustavus J. Simmons. The prisoners problem and the subliminal channels. In David Chaum, editor, *Advances in Cryptology*, pages 51–67. Springer US, 1984.