CS 6348: Project Report
Anmol Agarwal, Gina Fossati, Rhiannon Jones, Jennifer Ward

Introduction

As home automation becomes more prevalent and commonplace, the number of everyday, household objects that can be controlled through Wi-Fi or Bluetooth with a mobile application has drastically increased. Each of the many smart devices that are available pose a risk to the owner's privacy and security and, in some cases, to the privacy of those who visit their homes. With threats like that overhead, smart devices need to have their security and privacy assessed. To that end, our project assessed the security risks and privacy concerns of two smart devices: the Wandwoo Smart Lock and the Ring Doorbell 2. This report details the findings of our investigation. It begins with existing vulnerabilities we researched prior to testing the devices, continues with how we tested the devices, and concludes with the results of our testing.

Background

Known Existing Vulnerabilities

Ring Doorbell

Ring, owned by Amazon, has suffered from a number of concerning vulnerabilities. The first of these was a vulnerability that allowed attackers to steal Ring owners' Wi-Fi credentials [9]. During the Wi-Fi provisioning process, the Wi-Fi network's credentials were sent to the device unencrypted. This allowed any attacker who was watching the network to sniff the information and gain access to the user's Wi-Fi network. This vulnerability was patched in an update after Ring was made aware of the problem.

The Ring doorbell has also had issues with data encryption [7]. Another vulnerability, since fixed according to Amazon, gave attackers unlimited access to the video and audio data transmitted from the device to the user's app. The attacker first had to access the owner's Wi-Fi network. Once connected, the attacker could see the data the Ring doorbell was sending to its owner whenever the owner was connected to the same network since this data was sent over the network unencrypted. The attacker could then spy on the owner or replay old footage.

Late last year, Ring urged users to change their passwords and enable two-factor authentication after their login information may have been exposed online [8]. Ring claims this was a case of "credential stuffing" where attackers obtained the login credentials from another hack and reused them on other platforms. Through credential stuffing, attackers can gain access to live camera feeds, recordings, phone numbers, and home addresses. Attackers can also harass owners through the two-way chat feature. Ring does not have a mechanism in place to prevent people with unusual IP addresses from logging into accounts, and there is no limit on incorrect login attempts.

Wandwoo Smart Lock

We were unable to find many reported existing vulnerabilities specific to the lock we tested, but the ones we found are troubling. A key for the lock can be shared with other users. If a guest user disconnects from the internet after receiving a key, their access cannot be revoked. Guest users will still be able to unlock the lock until they connect to a network even though the owner

has revoked their access [4]. The locks also do not restrict password-reset attempts. This leads to incorrect access control and disclosure of sensitive information about valid accounts [3].

In addition to these vulnerabilities, we were able to find a number of vulnerabilities in other, similar locks. Another Bluetooth lock, unidentified due to its continued use by a hotel in Europe, is vulnerable to a key-stealing attack [5]. The lock utilizes Bluetooth Low Energy (BLE) to connect a mobile phone app to the lock and then unlock the door. Researchers sniffed the traffic used by the system. They could then perform a key-stealing attack to circumvent the vendor's replay protection. To perform this hack, the attacker must be local and must also know the lock's MAC address in advance. This vulnerability has had difficulty being patched because most of the locks are unable to receive updates over the air.

Another Bluetooth lock, Ultraloq, also has a number of vulnerabilities [6]. An issue with the BLE system allows attackers to use a brute force credential attack to open the lock. The API also provides no authentication mechanism and instead obfuscates data by encoding it in base64 twice. With these issues, attackers can easily impersonate users. Attackers can also intercept the login process and change the user ID used in the login process. The app will then function with the user ID the attacker provided, giving the attacker access to the lock. Attackers can also retrieve the BLE encryption key and potentially all user pins.

Approach

Ring Doorbell

Threat Model

For the Ring Doorbell, we consider an attacker who does not have direct access to the doorbell but can communicate with the device remotely via the server that publishes messages to the doorbell. The attacker can also sniff traffic from the doorbell to the server, record packets, and then replay the recorded packets. The attacker can also attempt to access the user's account remotely.

Wi-Fi Provisioning

To connect the Ring Doorbell 2 to a Wi-Fi network, the user must first download the Ring app and create an account. The account password must be at least 8 characters, and users can be told that their passwords are "weak." The account also requires the user's home address in order to provide other features to the user.

When setting up the device, the user can scan the device by giving the Ring app camera access, or the user can select which device he/she has to avoid giving the app camera access. Location access is required to access features and to complete the setup process. After the doorbell is named, it can then be connected to the Wi-Fi network. The device asks to join a network that is closest to where it is installed, and the user enters the network password, which is shown (not hidden) by default. The device then performs a firmware update and connectivity test and allows the user to adjust motion and mode settings.

Packet Capturing

We captured packets using Wireshark. We set up a laptop as a Wi-Fi hotspot and then connected the mobile phone with the Ring app and the Ring doorbell to the hotspotted network.

For some of the tests, we connected the phone to an external network to limit captured traffic to just the Ring devices.

Packets were then captured during a number of different operations. This included initial setup/account registration, adding a device to the Ring app, pressing the doorbell and not answering, and pressing the doorbell and answering, both while the phone was connected to the laptop's hotspot and while it was connected to another Wi-Fi network.

Attacks

The first attack we attempted was a passive attack to recover video and audio sent from the Ring doorbell via the real-time transport protocol (RTP). We attempted to recover the video from a Wireshark capture after noticing that Ring uses H264 to send video over Wi-Fi. The H264 protocol is a codec which compresses and decompresses video and is commonly used in video streaming. An internet post [1] claimed that back in 2017 the Ring Pro was sending these H264 packets unencrypted, so we followed the steps outlined in [2] to play back an RTP stream of H264 packets. We also attempted to listen to the audio using Wireshark's RTP functionality which shows the RTP stream and assembles audio with a playback option.

Ring offers a paid subscription for users that want to view video from previous motion detection events and rings. If a user is not subscribed to this service, Ring only allows the user to answer live ring events and view live motion detection. To get around this limitation, a Python 3.6+ library called ring_doorbell [10] was developed to expose the Ring devices as Python objects. We developed a program that uses ring_doorbell to further examine the security and potential vulnerabilities that may result from this API. Our implementation initializes the Ring object, lists devices linked to the account, prints device information, lists the past 15 doorbell events, downloads the last video triggered by ding, and displays the last video capture URL.

We also attempted a replay attack. Ring uses SIP, Session Initiation Protocol, to send a video stream from the Ring to one of their servers. SIP runs on UDP, so we used a tool called udpreplay [14] to send a session invite message that we captured with Wireshark. We also noticed that the Ring device was sending an HTTP POST request to an IP address owned by Amazon (likely an AWS server). The server responds to the POST with an HTTP message of "OK". We used TCPReplay [16] to replay this packet to the server.

Our final attack was to attempt to access the user accounts we created remotely. We did this by attempting to login on other devices that were not in the same location as the doorbell or the device that initially provisioned the doorbell.

Security Settings

Ring uses two-factor authentication (2FA) for account security. During account setup, this step can be skipped, but 2FA is still enabled by default and cannot be disabled. When trying to login, users receive a verification code at the email address they used for their account. The email with the verification code contains no information about where the login attempt was made. The user can also optionally change 2FA to SMS instead of email. Devices can also be authorized to access the account without 2FA.

Ring allows for users to link other accounts (Amazon or Paypal), and the user can control which linked accounts can access the devices associated with the Ring account.

When a user changes their password, they are logged out of their account and required to complete 2FA again. The password cannot be copied or pasted while logging in.

Privacy

Ring allows users to enable and disable the audio features of the doorbell. Privacy Zones are also used to restrict video coverage to protect neighborhood privacy. Ring also claims that data sharing settings can be adjusted in their privacy policy, but we were unable to find a way to do so. Mode settings also allow users to decide for each of three modes (Disarmed, Home, Away) if motion detection and live view are enabled or disabled.

Ring encrypts video recordings in transit using a combination of AES and TLS to secure data. There is also a companion app to the Ring app, called Neighbors, that allows users to share data with other Ring users within a 5-mile radius of their home. Only the videos a user makes publicly available can be viewed on the Neighbors app and by local public safety agencies. Personal information, such as names, is not shared within the Neighbors app.

Wandwoo Lock

Threat Model

For the Wandwoo lock, we consider an attacker who cannot directly access the lock. The attacker can sniff traffic to and from the lock, record packets sent to the lock, and replay packets to the lock. The attacker can also attempt to remotely access user accounts.

Bluetooth Provisioning

Like the Ring doorbell, to set up the Wandwoo lock, the user must download a mobile app called TTLock and create an account. The account requires a phone number or email that must be verified and a password that is between 8 and 20 characters in length and has 2 types of letters, numbers, and symbols. The lock and phone are then paired using Bluetooth 4.0BLE.

Packet Capturing

We captured Bluetooth traffic by using Android Developer options to enable the Bluetooth HCI snoop log option. Then we performed various actions on the lock, such as pairing and locking while the Bluetooth HCI snoop log option was enabled. We extracted the Bluetooth HCI snoop log from the Android device by downloading the Bug Report. We then used ADB to capture Android device contents and transferred the Android device contents to a Windows computer. We used Wireshark to examine the contents of the Bluetooth HCI snoop log.

Additionally, we captured Wi-Fi traffic from the TTLock app while creating an account. For this we used a setup similar to that of the Ring testing. A laptop was used for a hotspot, and a mobile phone with the TTLock app was connected to the hotspot. We used Wireshark to capture packets.

Attacks

The first attack we tried on the smart lock was to test the password reset verification code vulnerability as reported in [3]. We created an account through TTLock, the companion app for

the lock, and then reset the password, focusing on whether or not we could verify that the reported vulnerabilities still existed.

We also decompiled TTLock to see what vulnerabilities we could find in the app. We used two different online decompilers [12,13]. Both sites use jadx, a decompiler that produces java source code from an APK file. The decompilers produced similar results, and we were able to analyze both reproductions of the TTLock source code.

In addition to reviewing source code from decompilers, we used mobile application vulnerability scanners, Quixxi [11] and Ostorlab [15], to generate vulnerability reports on the TTLock APK.

Security Settings

The Wandwoo smart lock only allows one admin user at a time. This user can view records, adjust settings, change the lock's passcode, create new eKeys, and share eKeys. eKeys allow the smart lock owner to grant another user remote unlocking and authorized admin capabilities. If a user is granted authorized admin capabilities, this user is able to share eKeys, generate passcodes, view records, and view or modify certain settings. In order to grant another user remote unlocking capabilities, the smart lock needs to be connected to a G2 gateway (we did not purchase one of these gateways for testing). An admin user can adjust several settings for the lock, including auto lock time, passage mode, and ability to unlock remotely by an authorized user. The passage mode allows the user to specify certain time periods where the lock will remain unlocked until manually locked.

Privacy

The TTLock app, used to interface with the smart lock, runs in the background even after the user closes the app. The app also has a number of access permissions. On Android devices, it can access contacts, USB storage, precise location, battery statistics, system settings, and several others. It can also draw over other apps, allowing the app to steal passwords and read messages from other apps and also send messages as if they are from other apps. Apple permissions are more restrictive, only including location, Bluetooth, background app refresh, and notifications. The app also lets users share eKeys through WeChat, a notorious Chinese social media app that has no end-to-end encryption and is used for mass surveillance purposes. There is no mention of privacy (or a privacy policy) in the mobile app or on the TTLock developer page. There is such a lack of privacy that the support tab of the developer page displays a QR code linking to WeChat.

Challenges

The biggest challenge we faced during this project was the remote learning environment brought on by the COVID-19 pandemic. Because of this, not all team members were able to obtain and test the devices on their own. Collaboration was also difficult as we were unable to meet in person and instead had to use online resources to work together and test the devices remotely.

Results

Ring Doorbell 2

Our attempts to replay video captured by the Ring doorbell and sent in H264 packets over an RTP stream were unsuccessful. We received errors when trying to play back the captured video according to the steps outlined in [2]. Along with the video streams, all of the audio streams we captured were incorrect. They all sounded like white noise even though we used noise and music to make the audio identifiable. These results lead us to believe that the H264 packets are either encrypted or use some type of padding, thus confirming that Ring had since started encrypting the audio and video data that is sent to the user from Ring devices.

In our testing of the Ring doorbell API (ring_doorbell), we found that when a Ring account is first accessed, the username, password, and authentication code must be provided in order to access the devices and data associated with that account. This information is used to generate a token that is stored in a cache file on the user's device. When the user runs the program again, it uses the token in the cache file to reauthenticate so that the user does not have to re-enter the authentication information. We found that the token does not seem to expire so the user does not need to re-enter the username, password, and authentication code unless the cache file is deleted or the account holder changes the password, in which case the token will no longer be valid. When the cache file was transferred to another user's location in a different city, that user was able to gain access to the account by running the program with that cache file without entering any account information. Therefore, once the token is generated, the cache file can be sent to multiple people in various locations to access the account information through the use of this program. When the user initially enters the username, password, and authentication code, the account holder is notified that a user has accessed his or her account, but subsequent executions of the program with the generated token do not notify the account holder.

Both of our replay attacks were unsuccessful. The Ring did not receive any response after the SIP invite was replayed, so it is likely that an element of freshness is used. Similarly, no response was returned by the replayed POST request, so we believe there is freshness here too.

While testing the Ring accounts through the app, we found that there was seemingly no limit on the number of incorrect password attempts. After 10 attempts, a correct password was accepted, and 2FA functioned normally and did not notify the user of the failed login attempts. The 2FA emails the user received also contained no location data to let the user know where the login attempt came from. This was a previously reported vulnerability that our testing confirms has not yet been fixed. Suspicious IP addresses can still log into a Ring account provided that they can also complete the 2FA process, and the user will not know where the suspicious login was made.

Wandwoo Lock

After testing the previously reported vulnerability in the TTLock application's password reset verification process as found in [3], we discovered that the issues have not been fixed. Our testing found that previous reset codes were not invalidated when a new reset code was requested. If a user were to request a code and then request another shortly thereafter (new codes can only be requested every 60 seconds), both codes would be valid to use to reset the user's password. If the user uses one of the two codes and then attempts to use the same code again, the user is informed that the password reset verification code "is invalid or has expired." As for the unused codes, we tested multiple password reset verification codes and found that they did not expire after one hour as advertised. The codes we tested lasted for approximately 1

hour and 40 minutes. There is also no restriction on trying many different combinations to verify the code. Password verification codes are 6 digits, meaning there are 1,000,000 possible combinations for any given password reset verification code. Without a limitation to try all possible codes, it is possible for an attacker to use brute force to guess the password reset code sent to a user and then change that user's password. Despite all the continued issues, at least one vulnerability in the login process was fixed. When logging in, the error messages displayed no longer indicate a bad username. The application now displays "bad username or password" if an invalid username is entered instead of "bad user". Thus, it is no longer possible to determine if a user account with a certain email exists or not.

In our review of the decompiled source code of the smart lock app TTLock, we found that the app uses the JPush library to send push notifications [17, 18]. As the application is Chinese in origin, it uses Chinese names for some portions of the applications and in the hardcoded string values. The application uses POST with HTTP connections. To manage trust, the application uses x.509 certificates which are common in TLS/SSL and HTTP connections. The most important discovery we made regarding the application was that it appears to only use the MD5 hash function on passwords when using the Gateway for remote access. Initially designed for use as a cryptographic hash, MD5 has been found to have extensive vulnerabilities and is now used for data integrity, primarily as a checksum. We were unable to obtain the necessary equipment to setup a Gateway to test if the MD5 password was transmitted using TLS. However, when analyzing the traffic captured from TTLock, TLS is used both when creating an account and sending eKeys.

After using Quixxi [11] to scan the app used by the Wandwoo smart lock, TTLock, we found a number of vulnerabilities. Of these, 6 were high severity threats, 11 were medium severity, and 4 were low severity threats. The vulnerabilities found included weak random number generation, insecure cryptographic tools, weak hashing algorithms, improper export of Android Activities, Services, Broadcast, and Content Providers, and command injection vulnerabilities. This combined with the inherent insecurities of the app (passcode sharing through insecure apps such as WeChat, etc.) shows that TTLock not only fails to protect its users' privacy but also does not provide its users with security either. The results of the Quixxi vulnerability report also suggest that the app itself may pose a threat to users.

The results of our scan of the TTLock application with Ostorlab [15] were not much better than those of the Quixxi scan. The scan reported 1 medium risk, 4 potential risks, 1 important risk, and 24 information risks. Of these, most were API calls, and several were the same as the vulnerabilities reported by the Quixxi scan.

Division of Labor Among Team Members

Anmol researched existing vulnerabilities in the Wandwoo smart lock and the Ring doorbell. She also tested the Wandwoo smart lock and its companion app. Using the Bluetooth HCI snoop log option, she captured Bluetooth traffic between the smart lock and a smartphone. She was also responsible for the decompilation of the TTLock application and helped examine the decompiled source code for vulnerabilities. Anmol also performed the vulnerability scans on the TTLock application using Quixxi and Ostorlab, and she tested the TTLock application for existing vulnerabilities, such as the password reset errors.

Gina researched and analyzed privacy and security settings for both devices, captured Wi-Fi traffic for both devices using Wireshark, analyzed the captured traffic, attempted attacks on the Ring device, and helped examine the decompiled TTLock application.

Jen researched and analyzed existing vulnerabilities as well as the security and privacy settings of both devices. She wrote and tested the program that uses ring_doorbell. She also assisted in the analysis of the captured Ring traffic and examining the decompiled TTLock application source code for potential vulnerabilities. She also helped with remotely testing the devices.

Rhiannon researched vulnerabilities in both the Wandwoo smart lock and the Ring doorbell. Based on her research, she offered ideas and suggestions for testing as she was unable to test the devices herself. She helped organize and compile findings. She also helped perform remote tests, including logging into both apps from a different location to see how the apps responded, and she helped examine the decompiled source code of the TTLock application.

Conclusion

Recommendations

Based on our findings, we recommend against using the Wandwoo smart lock. The companion app, TTLock, is riddled with vulnerabilities and privacy concerns. User accounts can be hacked through the password reset process, leaving homes and businesses protected by the lock insecure, and the app itself has far more permissions than are necessary for it to perform its desired functions. As such, simply downloading the app poses a security threat to users. Between this and TTLock's insecurity, the Wandwoo smart lock is an example of an IoT device that lacks security and privacy and causes more harm than good.

Our testing concluded that Ring has improved their security. After a slew of reported vulnerabilities last year, Ring has fixed many of them, making its devices far more secure than they were to begin with. That said, we would still recommend against using the Ring Doorbell 2. We were unable to obtain video and audio sent from the doorbell to the user, but we were unable to conclude whether this was due to an encryption scheme, padding, or an encoding scheme. As such, we believe there is still a possibility that the data sent from the doorbell to the user is vulnerable. Due to the privacy and security concerns such a threat poses, we cannot recommend using the device.

Future Works

Though we found many vulnerabilities in TTLock, the Wandwoo smart lock companion app, we were unable to test most of them. In the future, the companion app should be tested to see what attacks are possible with the vulnerabilities we found. In particular, the high severity threats should be tested. These include improper export of Android Activities, Services, Broadcast, and Content Providers, unsafe TrustManager implementation, unsafe file deletion, critical permission usage, and command injection vulnerabilities. The use of MD5, reported as a weak hashing algorithm in the Quixxi report and verified by our review of the decompiled source code, should also be tested to determine if passwords can be recovered from the application.

As for the Ring doorbell, the video and audio sent from the device to the user should be tested more. We were unable to watch videos sent to the user or listen to recorded audio, but, given more time and resources, we think it could be possible to still obtain the data simply by sniffing the packets as they are sent.

References

[1] "Well, I Captured Some Traffic from My Ring Pro." Accessed on: Apr 23, 2020. [Online]. Available: https://www.reddit.com/r/ringdoorbell/comments/7as4a9/well_i_captured_some_traffic_from_my_ring_pro/

[2] "How to playback video from Network Traces," Apr. 22, 2020. Accessed on: Apr. 28, 2020. [Online]. Available: https://knowledgebase-iframe.polycom.com/kb/viewContent.do;jsessionid=86374ECDD4225AB15ECA0F1EC663C6BA?externalId=34917

[3] A. Viderberg, "Vulnerability Report TTLock Password Reset Mechanism Attack," June 2019. Accessed on: Apr. 27, 2020. [Online]. Available: https://www.kth.se/polopoly_fs/1.923564.1568098316!/Vulnerability_Report_TTLock_Password_Reset.pdf

[4] A. Viderberg, "Vulnerability Report TTLock State Consistency Attack ," June 2019. Accessed on: Apr. 27, 2020. [Online]. Available: https://www.kth.se/polopoly_fs/1.923565.1568098364!/Vulnerability_Report_TTLock_State_Consistency.pdf

[5] L. O'Donnell, "Hack of High-End Hotel Smart Locks Shows IoT Security Fail," Aug. 19, 2019. Accessed on: Apr. 13, 2020. [Online]. Available: https://threatpost.com/hack-of-high-end-hotel-smart-locks-shows-iot-security-fail/147178/

[6] T. Spring, "Smart Lock Turns Out to be Not So Smart, or Secure," June 27, 2019. Accessed on: Apr. 13, 2020. [Online]. Available: https://threatpost.com/smart-lock-turns-out-to-be-not-so-smart-or-secure/146091/

[7] L. O'Donnell, "Ring Plagued by Security Issues, Flood of Hacks," Dec. 18, 2019. Accessed on: Apr. 13, 2020. [Online]. Available: https://threatpost.com/ring-plagued-security-issues-hacks/151263/

[8] D. Wroclawski, "3,000 Ring Doorbell and Camera Accounts May Be Vulnerable to Hackers," Dec. 19, 2019. Accessed on: Apr. 6, 2020. [Online]. Available: https://www.consumerreports.org/hacking/ring-doorbell-accounts-may-be-vulnerable-to-hackers/

[9] A. Ng, "Ring doorbells had vulnerability leaking Wi-Fi login info, researchers find," Nov. 7, 2019. Accessed on: Apr. 6, 2020. [Online]. Available: https://www.cnet.com/news/ring-doorbells-had-vulnerability-leaking-wi-fi-login-info-researchers-found/

[10] Accessed on: Apr. 6, 2020. [Online]. Available: https://python-ring-doorbell.readthedocs.io/

[11] Accessed on: Apr. 6, 2020. [Online]. Available: https://quixxisecurity.com/

[12] Accessed on: Apr. 12, 2020. [Online]. Available: http://www.javadecompilers.com/

[13] Accessed on: Apr. 12, 2020. [Online]. Available: https://www.apkdecompilers.com/

[14] Accessed on: Apr. 12, 2020. [Online]. Available: https://github.com/rigtorp/udpreplay

[15] Accessed on: Apr. 12, 2020. [Online]. Available: https://www.ostorlab.co/

[16] Accessed on: Apr. 12, 2020. [Online]. Available: https://linux.die.net/man/1/tcpreplay

[17] Accessed on: Apr. 12, 2020. [Online]. Available: https://docs.jiguang.cn/en/jpush/guideline/intro/

[18] Accessed on: Apr. 12, 2020. [Online]. Available: https://github.com/jpush