EvilScreen Attack: Smart TV Hijacking via Multi-channel Remote Control Mimicry

Yiwei Zhang, Siqi Ma, Tiancheng Chen, Juanru Li,
Robert H. Deng, *Fellow, IEEE,* Elisa Bertino, *Fellow, IEEE*

Abstract—Modern smart TVs often communicate with their remote controls (including the smartphone simulated ones) using multiple wireless channels (e.g., Infrared, Bluetooth, and Wi-Fi). However, this multi-channel remote control communication introduces a new attack surface. An inherent security flaw is that remote controls of most smart TVs are designed to work in a benign environment rather than an adversarial one, and thus wireless communications between a smart TV and its remote controls are not strongly protected. Attackers can leverage such a flaw to abuse the remote control communication and compromise smart TV systems.
In this paper, we propose EVILSCREEN, a novel attack that exploits ill-protected remote control communications to access protected resources of a smart TV or even control the screen. EVILSCREEN exploits a multi-channel remote control mimicry vulnerability present in today smart TVs. Unlike other attacks, which compromise the TV system by exploiting code vulnerabilities or malicious third-party apps, EVILSCREEN directly reuses commands of different remote controls, combines them together to circumvent deployed authentication and isolation policies, and finally accesses or controls TV resources remotely. We evaluated eight mainstream smart TVs and found that they are all vulnerable to EVILSCREEN attacks, including a Samsung product adopting the ISO/IEC security specification.

Index Terms—Smart TV, Remote Control, Multi-channel, Authentication and authorization, Security analysis

1 INTRODUCTION

SMART TVs present both privacy and security risks. Features such as Internet-based media playing and third-party app executing make modern TVs smarter and yet more vulnerable to security attacks and privacy intrusions. A variety of vulnerabilities have been exploited against smart TVs in recent years [1], [2], [3], [4], [5], [6], [7], [8]. In general, security threats against smart TVs can be classified into two categories: threats from Internet, and threats from programs running on smart TV OSes (e.g., Android TV OS [9]). In response, smart TV manufacturers and TV OS providers have deployed a variety of protection measures.

While security researchers and TV manufacturers are making a concerted effort to strengthen smart TVs, we observed that they often ignore a new attack surface — *multi-channel remote control communication*. Figure 1 shows a typical application scenario: a smart TV simultaneously supports three types of remote controls using different signals, i.e., Consumer Infrared (IR), Bluetooth Low Energy (BLE), and Wi-Fi. In addition to remote controls provided by specialized TV accessories, a smartphone can be used as a remote control by installing a *companion app* developed by the TV manufacturer. By sending BLE and Wi-Fi signals, users can interact with the TV. This *companion app simulated remote control* is generally more powerful than classical remote controls since it can fully make use of the resources of the host smartphone.

Although multi-channel remote control communication enhances easy-of-use and flexibility for smart TV users, it weakens security: a smart TV often treats its remote controls as benign accessories, and neither effectively authenticates their identities nor verifies data they send. Unfortunately, most remote controls lack necessary protection, and thus attackers could easily impersonate a remote control or tamper the wireless traffic. More seriously, to support enhanced features (e.g., playing video files from a companion app simulated remote control), smart TV OSes add *remote control interfaces* to handle sophisticated remote commands and execute privileged operations. If the access control mechanisms of those interfaces are not well designed, attackers could simply abuse them to hijack the TV (e.g., monitoring the screen, displaying contents, and controlling the user interface (UI) of the TV).

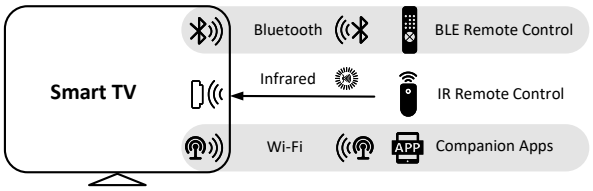

Fig. 1. A common multi-channel remote control communication scenario for popular smart TVs

EVILSCREEN Attack. In this paper, we present a new type of attack, EVILSCREEN, against multi-channel communication between a smart TV and its remote controls. Unlike existing attacks that need to install a malicious app on the TV or exploit the TV OS, EVILSCREEN only reuses communications of remote controls to hijack the victim TV, making it more difficult to prevent and detect the attack. We found that the root cause of this attack is a **multi-channel remote control mimicry** vulnerability (EVILSCREEN vulnerabilities for short). In general, an EVILSCREEN vulnerability is a smart TV access control bug which allows an attacker to combine three types of wireless communications together (i.e., IR, BLE, and Wi-Fi) to circumvent the authentication and isolation policies of each single remote control. Then

the attacker could abuse corresponding remote control interfaces to hijack the TV. In fact, exploiting single remote control does not result in severe security threats. However, by combining functionalities of multiple remote controls, one can design complex attacks.

To exploit an EVILSCREEN vulnerability, three consecutive steps are needed. First, the attack utilizes less secure wireless channels (i.e., IR and BLE) to enforce a Wi-Fi provisioning [10], a common procedure for smart TVs to receive credentials of a protected WLAN (i.e., SSID and password). When inside the same WLAN, as most smart TVs would not check the remote control paring request or apply vulnerable pairing mechanisms, the attack leverages this weakness to actively bind a fake remote control to the TV. Once the fake remote control is bound to the TV, the attacker then abuses the remote control interfaces to access TV resources and control the screen.

In comparison with attacks relying on meticulously crafted signals (e.g., leveraging inaudible voice commands to control the TV [11], [12], [13]), the EVILSCREEN attack only uses common wireless technologies and is more general. We conducted an empirical study against eight popular smart TVs from retail markets of China, Japan, Korea and the United States. Our study showed all of them were vulnerable to the EVILSCREEN attack. Unlike attacks, such as BIAS [14] against Bluetooth and KRACK [15] against Wi-Fi, the EVILSCREEN attack does not aim to break any of the three wireless protocols used by remote controls. Instead, it exploits the fact that during communications between the remote controls and the smart TV, because of usability considerations, simplified security controls, or even no security controls at all, are applied. We also present a case study for the Samsung smart TV. Although Samsung adopts the ISO/IEC 30118-1:2018 standard [16], it is still vulnerable to the EVILSCREEN attack as it does not strictly follow the security regulations involved in the standard to protect its remote control communication. In fact, a usability factor related to the Samsung SmartThings companion app significantly reduces the crypto key randomness, leading to vulnerable remote control communications. We constructed a practical brute-force attack to breach its DTLS-over-BLE and WebSocket-over-Wi-Fi communication between the TV and its companion app simulated remote control.

The main contributions of this paper are as follows:

- *New Understandings.* We systematically analyzed how the use of remote controls affects the security of popular smart TVs. We show that design flaws of remote controls break the security assumptions of protection solutions currently deployed on wireless technologies such as BLE and Wi-Fi.
- *New Attacks.* We implemented the EVILSCREEN attack [1] that affected 200 millions of popular smart TVs worldwide. Unlike attacks aiming at exploiting code vulnerabilities of TV OSes or apps, EVILSCREEN attack only utilizes legitimate protocols and services. Therefore, current protections are less effective against our attack. We also outline countermeasures for smart TV manufacturers and developers to mitigate the EVILSCREEN attack.

1. We have released more details about EVILSCREEN attack at https://sites.google.com/view/evilscreen.

2 BACKGROUND

In this section, we first discuss two new characteristics of smart TVs. Then, we describe common protection schemes of smart TVs with focusing on the ones to protect app simulated remote control communications.

2.1 Characteristics of Smart TVs

Smart TVs provide a variety of new features and functions to enrich their user experiences. Typically, a TV is considered "smart" when: (i) it supports multiple accessories that communicate through various transmission channels; (ii) it supports more advanced commands to access various local/online resources (though Internet connections). Such multi-channel communication and advanced interaction features are the key distinctive features of smart TVs compared to most traditional TV devices. In the following, we give detailed descriptions of these two features.

Multi-channel Communication. A distinct feature of smart TVs is the use of multiple wireless communication channels. As Figure 1 shows, the communication between a smart TV and its accessories (**TV-Accessory communication** for short), including remote controls and smartphones, utilizes three widely used wireless signals, i.e., Consumer Infrared (IR), Bluetooth Low Energy (BLE), and Wi-Fi.

IR and BLE based communications are commonly employed in traditional TV remote controls (namely *physical remote control*). But they are still supported by most smart TVs because of user experience considerations and compatibility issues. However, both IR and BLE signals are short-range, which requires data transmission within a short distance. Worse still, they lack strong authentication mechanisms [17], [18], [19], [20], [21], [22], [23], [24]. Therefore, transmitting sensitive data via IR and BLE channels is risky.

In addition to IR and BLE, modern smart TVs also support companion apps on smartphones (called *virtual remote control*) to control the smart TV and access TV resources via Wi-Fi. After binding with a smart TV, a user can send various commands through these companion apps via the Internet, which will then be received, parsed and executed by the TV OS. Compared with IR and BLE signals, Wi-Fi data transmission adopts well-designed security specifications (e.g., WPA2 and WPA3 [25]), and therefore is suitable for data transmission with strong protection requirements.

Advanced User Interactions. With different types of remote controls, the interactions between users and TV devices differ significantly. For the physical IR/BLE remote control, a user can only execute some basic operations (like power on/off, volume up/down, channel switching) due to the limited number of key buttons and resources on the remote control device. In contrast, a companion app is able to support more advanced user interactions. In addition to basic functions, a companion app can directly send one high-level command to the smart TV instead of controlling the cursor on screen step by step to perform the operation. Typically, there are three types of advanced operations:

- **TV App Operation** function allows a user to open/install/uninstall a TV app via the companion

This article has been accepted for publication in IEEE Transactions on Dependable and Secure Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2023.3286182

app. In most cases, a user can select and install both official TV apps displayed in the companion app and customized local app files on the smart TV.
- **Screencast** function allows a user to display the contents (e.g., media files, documents) stored in the smartphone on the smart TV screen; the content can then be watched by multiple people at the same time.
- **Screenshot** function allows a user to monitor the TV screen in real time by either taking a screenshot picture of what the TV is displaying or transmitting the current screen content as streaming video data, and display it on the smartphone.

2.2 Protections against Wireless Attacks

Since complicated features have been introduced in smart TVs, a variety of vulnerabilities have been exploited against different components of smart TVs, such as the firmware and the browser [1], [2], [5], [6], [8]. In response, manufacturers and TV OS developers have built techniques for smartphones protection and have adopted several well known defenses, such as Mandatory Access Control (MAC) and Address Space Layout Randomization (ASLR). Nonetheless, a remarkable attack surface for smart TVs is their TV-Accessory wireless communication. To protect smart TVs against remote wireless signals based attacks, the following measures are often employed.

Protection I: Network Isolation. When initially launching a smart TV, the user typically configures the TV to connect to a Wi-Fi network. At this stage, the smart TV often relies on the user to send the Wi-Fi credentials (i.e., the SSID and password of the Wi-Fi) via its IR or BLE remote controls. After the network connection, the smart TV is under the protection of WLAN isolation. Thus, only authenticated devices are allowed to join the (WLAN) network and access TV resources.

Protection II: TV-Accessory Binding. The smart TV and its accessories (physical or virtual remote controls) in the same WLAN need to complete a binding process before further remote interactions. Conventionally, a binding process involves a mutual authentication between the smart TV and the accessory, which ensures the smart TV to be bound with the permitted accessories only. Otherwise, attackers might be able to exploit the smart TVs by compromising other vulnerable devices (e.g., smart routers) in the same WLAN.

Protection III: Remote Interaction Validation. The remote user is not allowed to use resources of the smart TV arbitrarily. Since a variety of remote user interactions supported by the smart TV require to access to sensitive resources (e.g., screen contents, system settings) or modify these resources, the smart TV applies access control to all remote operations to check whether a request is allowed. Specifically, the TV OS introduces new interfaces to handle different remote operations sent by the user and perform permission checks. The permissions are granted to accessories after the binding phase, and when a resource interface is invoked by an accessory, the TV OS determines whether the accessory has been granted the specific permissions to access the resource. To distinguish different accessories, the smart TV generally distributes an access token to each accessory beforehand, and asks any remote request to attach the proper access token.

3 Threat Model

Our attacker model is as follows. First, we assume the smart TV OS is securely implemented. Thus the attacker cannot hijack and compromise the smart TV by exploiting the OS. In addition, we assume that no malicious apps were previously installed on the victim TV. Second, in order to analyze the smart TV and find potential security flaws, we assume that the attacker can purchase any products beforehand. We also assume that, through his own analysis, the attacker is able to gather necessary information, such as device configurations and companion app implementations, and extract the specific side channel information (e.g., a certain range of mac addresses [26], signal characteristics [27]) to identify the model and brand of the victim TV. Third, the communication channels between the accessory and the smart TV are exposed to the attacker. The attacker can sniff network traffic and capture network packets. However, we assume that the attacker is not able to circumvent existing communication channel protections. For instance, data transmitted through the Wi-Fi channel is protected, that is, the attacker is unable to crack Wi-Fi credentials by launching brute-force attacks.

3.1 Two Attack Scenarios

In real-world scenarios, a smart TV may be bought for different usage purposes, hence be installed at various locations. Depending on the location variants, the attacker model might also be slightly different. In the following, we discuss two main real-world attack scenarios, i.e., public smart TV and personal smart TV.

Publicly Accessed Smart TVs. A large number of smart TVs are placed in public areas, such as shopping malls and gyms for commercial or entertainment purposes (e.g., advertising). Under such a scenario, the attacker can easily approach the publicly placed smart TV. Therefore, the attacker can not only monitor (and exploit) the remote communication between the TV and its accessories, but can also actively send malicious signals to the screen. However, if the attacker significantly changes the content on the screen, the attack is easily discovered. Thus attacks against publicly accessed smart TV must be *stealthy*.

Personal Smart TVs. As the personal smart TVs are generally placed in private spaces, such as homes or hotel rooms, the attacker cannot easily get close to them. The attacker would then have to place a malicious device in the private space, or compromise a vulnerable device inside the space, and use it as a *relay device* to launch the follow-up attacks. Such a relay device should satisfy the following conditions: (i) it is able to constantly sniff the victim network and actively send various types of signals (i.e., IR, BLE and Wi-Fi) to the smart TV; (ii) it can be fully controlled by an attacker, and is able to send captured data to the remote server (i.e., attacker) or receive data from the attacker. With such a relay device, the attacker is able to remotely communicate and interact with the smart TV (e.g., send commands and receive TV private data) instead of being physically present in the private places. Like publicly accessed smart TVs, attacks for personal smart TVs must also be stealthy. The attacker should launch attacks during *certain time periods* when the TV is seldom used (e.g., late nights) without the user noticing anything and then wait for the victim to use the TV. This is feasible because when

This article has been accepted for publication in IEEE Transactions on Dependable and Secure Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2023.3286182

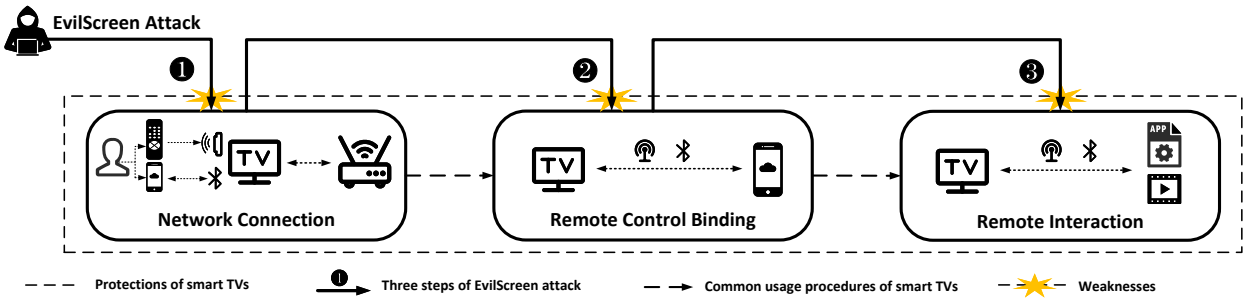

Fig. 2. A typical flow of EVILSCREEN attack: it conducts a consecutive three-step attacking process to circumvent common protection measures and finally hijacks the screen of the smart TV

the user sends a "power off" command to a smart TV, most smart TVs do not actually power off themselves but only turn off the screen and keep the OS running. Hence, an attacker would be able to send a "power on" command to wake up the TV and perform follow-up attacks.

3.2 Security and Privacy Issues

Attacks for smart TVs can result in serious security and privacy issues. We discuss two of those in what follows.

Illegal content display. Most smart TVs support the screencast function, which can display smartphone content on the TV screen for multi-people watching. However, if a smart TV screen is hijacked by an attacker, the attacker would be able to cast any illegal contents (e.g., bloody or violent videos) to the TV screens of people he/she wants to victimize, which may damage people's health, especially in the case of families with kids. Also, if the attacker is a profiteer or malicious merchant, he/she could cast nasty contents on the screen of the victims to denigrate competing products or display his/her product advertisements to potential customers for profit.

Leakage of private data. The screenshot function is supported by many smart TVs. With this function, a user can monitor what the TV is displaying on the TV screen via the companion app even if the user is not in front of the TV. Unfortunately, this could lead to a serious threat to users' privacy. If an attacker can hijack one's TV screen and monitor the contents watched by the user, the attacker would be able to learn information about the user's interests, political orientation and personality [28], [29]. What's worse, since more and more people working at home frequently cast their computer screens on the smart TV during meetings to have an enhanced viewing experience, confidential company information or secret documents may be leaked via such a TV attack. In addition, some smart TVs also allow users to sign into the TV apps or pay for subscriptions [30], [31]. If the attacker takes a screenshot when the user is inputting her account information or payment information, the user will not only suffer from personal privacy breaches but also from economic losses.

4 THE EVILSCREEN ATTACKS

EVILSCREEN attack is a new type of wireless communication attack that exploits a type of **multi-channel remote control mimicry** vulnerabilities. Abstractly, an EVILSCREEN vulnerability exploits three weaknesses in the architecture/implementation of smart TV systems, i.e., vulnerable authentication dependency, insufficient binding information, and ill-designed remote interface access control, to circumvent the three protections presented in Section 2.2. By simultaneously leveraging these three weaknesses, a successful EVILSCREEN attack would allow the attacker to remotely monitor the screen and/or perform hijacking against the victim smart TV.

4.1 Overall Attack Procedure

A generalized EVILSCREEN attacking process consists of three consecutive steps, as shown in Figure 2:

❶ **Network isolation bypassing.** The goal of the EVILSCREEN attack is to hijack the TV screen and perform sophisticated malicious activities with the use of the companion app via a Wi-Fi channel. Hence, an attacker (or a relay device) should first connect to the same Wi-Fi to which the victim smart TV is connected. The reason is that most TV manufacturers restrict the communication between smart TV and companion app to the same Wi-Fi.

❷ **Malicious remote control binding.** After the first step, the attacker can communicate with the victim smart TV. In order to further control the smart TV, the attacker must be able to construct a malicious TV-Accessory binding between the victim smart TV and the companion app on the attacker's smartphone (or a simulated one).

❸ **Remote interfaces abusing.** If the binding is successfully, the smart TV then can receive, parse and execute commands sent from the companion app. However, in order to perform sensitive operations and hijack the smart TV screen, the attacker still has to bypass permission check.

To launch the EVILSCREEN attack successfully, those steps must be executed sequentially. The reasons are that: (i) only after connecting to the same Wi-Fi, the smart TV and the companion app are be able to communicate with each other; and (ii) only after binding successfully with the smart TV, the companion app is able to control the smart TV. After that, if the attacker wants to execute some sensitive operations, then he should perform the third step. Furthermore, before performing the three attack steps, a pre-analysis is necessary to detect if a victim smart TV is vulnerable to the EVILSCREEN attack. Also, an EVILSCREEN attack should be launched via the Wi-Fi channel (as well as using companion apps) since stealthiness is required. Otherwise, an attacker would only be able to perform complex TV operations by moving the cursor on the screen step by step, which can be easily noticed.

In the following, we first describe the analysis to be carried for each step to determine if a smart TV is vulnerable to EVILSCREEN and then illustrate the attack procedure.

4.2 Network Isolation Bypassing

Weakness. Although authentication schemes for smart TVs are securely designed and implemented, they have the major issue of *authentication dependency*, that is, Wi-Fi credentials are transmitted via other channels. More specifically, we found out that Wi-Fi provisioning is commonly executed via IR/BLE communications; however both IR and BLE communications have vulnerabilities. As a result, Wi-Fi provisioning for smart TV is vulnerable to passive and active attacks (e.g., man-in-the-middle attacks and impersonation attacks).

In order to bypass network isolation, the attacker could either retrieve the Wi-Fi credentials to connect a malicious remote control to the secure Wi-Fi or force the smart TV to re-connect to a new malicious Wi-Fi. To understand how to carry out such attacks, we analyze IR/BLE communications to exploit authentication dependency.

Analysis. To detect authentication dependency, we capture the IR/BLE wireless signals to analyze the authentication schemes utilized by each type of signal. In advance, we check the user instruction and the technical manual of each smart TV to confirm which types of wireless channels are supported. Then we execute Wi-Fi provisioning. If either IR or BLE is supported, we check whether they are used for distributing Wi-Fi credentials, and the security of the distribution.

To analyze the IR signals, we leverage the existing public databases [32], [33] and mobile apps with IR remote control functions [34] to integrate the existing encoding and decoding methods. Then an IR receiver is used to capture the IR signals from remote controls during network provisioning, and we decode the IR signals by utilizing the integrated decoding methods. If Wi-Fi credentials are identified from IR signals, we consider the smart TV as suffering from authentication dependency issues. We further verify if there is any authentication mechanism embedded in the IR channel. In particular, we build an IR simulator on top of an IR emitter [35], [36] and use the simulator to send simulated IR signals in different encoding schemes. If the smart TV accepts these IR signals and executes the corresponding operations, we consider the Wi-Fi credential distribution as *authentication dependent* over a vulnerable IR channel that cannot protect against active impersonation attacks.

For analyzing the BLE signals, we leverage TI CC1352 Development Board [37] to sniff BLE packets. We first analyze the packets and examine the Secure Connection-bit in the BLE Pairing packets. If the Secure Connection-bit is set to 0, it indicates that the devices are bound via *Legacy Connection*, which is vulnerable to brute force attacks [18]. When a *Legacy Connection* vulnerable BLE scheme is identified, we further utilize Crackle [38] to decrypt the BLE packets, learn the data formats and check whether there are any credentials included. Like the IR channel analysis, if the BLE packets contain Wi-Fi credentials, we consider the credential distribution as *authentication dependent* on an insecure BLE channel, which is vulnerable to passive man-in-the-middle (MITM) and brute force attacks. Additionally, we determine which BLE pairing mode is used by checking the MITM-bit and IOCaps [39]. If none of the MITM-bits in both the exchanged device pairing features is 1, the mode is *Just Works*, in which no authentication mechanism is applied. In this case, we also regard Wi-Fi provisioning procedure as *authentication dependent* on a BLE channel vulnerable to active impersonation attacks.

Attacks. By the pre-analysis we can determine whether a smart TV is vulnerable to *authentication dependency* and learn the encoding scheme and protocol formats used in IR and BLE communication. Then we can launch either passive attacks or active attacks to compromise the smart TV. When Wi-Fi credentials are identified from the vulnerable IR or BLE communications with *Legacy Connection*, we use the extracted Wi-Fi credentials to connect a malicious device to the same Wi-Fi. Alternatively, if the Wi-Fi provisioning procedure is *authentication dependent* on a vulnerable IR or BLE channel without any authentication mechanisms against active impersonation attacks, we impersonate a legal user to compromise the smart TV. By connecting a malicious remote control with the TV, we are able to reuse the encoding methods or protocol message formats and then force the TV to reconnect a new malicious Wi-Fi.

4.3 Malicious Remote Control Binding

Weakness. Due to efficiency and usability reasons, manufacturers usually deploy light-weight, easy-to-understand, but insecure binding mechanisms. In order to bind a smart TV with its remote control, manual attestation is generally required. However, most smart TVs only display a device name or a binding token on the screen. This information is insufficient for users to distinguish whether the request is sent from a legitimate remote control or a malicious one; thus the binding mechanisms are vulnerable to impersonate attacks. Hence, we investigate what binding information is displayed on the screen and exploit the transmission channels to identify whether the binding information is verified by the smart TV.

Analysis. To exploit the protection schemes of remote control binding, we execute UI differential analysis to investigate the binding information utilized by the smart TV and how the binding request is formed. First, we use different smart remote controls with unique identifiers (e.g., series numbers and MAC addresses) to send binding requests to each smart TV manually. Then we record the binding information displayed on the screen while using different remote controls. By comparing the information generated for different remote controls, we examine whether the displayed information is invariant. The binding authentication is regarded as vulnerable if the display information is constant or only limited device information is provided.

Attacks. After having obtained the binding information, we modify the binding request to connect a malicious remote control with the smart TV. First, we check whether the binding request is properly validated by the smart TV. By monitoring network traffic, we intercept communication packets to study the packet format. If the binding request is transmitted over an insecure communication channel, we can directly explore the packet format and further change the authentication-related fields containing binding information (e.g., “username”, “password”, “device name”) with the legitimate remote control information and then send the modified packets to the smart TV to request for binding. For the binding requests protected by the SSL/TLS protocol, we

use Burp Suite [40] to retrieve the binding request format and then replace the authentication-related fields with the legitimate information. The forged request is then sent from a fake client. If the smart TV accepts the request and displays an indistinguishable binding information on the screen, we consider such a binding mechanism as vulnerable and a malicious remote control binding can be established.

Some smart TVs rely on binding tokens to protect against impersonation attacks. Nonetheless, such a binding scheme is still vulnerable because the involved token is not well-protected. In particular, a binding token may be broadcast to the remote control, or embedded in the companion app by default. If the token is broadcast, we intercept the communication packets to obtain the corresponding token. Otherwise, we reverse engineer the companion app to retrieve the token. If the binding token is implemented without enough entropy, brute force methods can be used to obtain such a token.

4.4 Remote Interface Abusing

Weakness. As a smart TV stores a variety of sensitive resources (e.g., system setting, media files, user configuration), the smart TV checks access permissions to avoid arbitrary resource access. However, unlike personally owned devices (e.g., smartphones, laptops), smart TVs commonly apply a coarse-grained access control to protect against unauthorized access because they are commonly shared by a group of people such as a family (private-use) or consumers (public-use). Such an authorization scheme has *permission check* weaknesses that can be exploited by an attacker to access resources without having the corresponding permissions. Considering the usage purposes and scenarios of smart TVs, we believe that smart TVs should be developed with restrictive fine-grained access control [41]. That is, users connecting to the smart TV should not be able to access all resources but be allowed to use basic functionalities. Only after the TV owner grants the proper permissions, users should perform sensitive operations.

In order to check whether the permissions are properly granted, we investigate the protocols for remote interactions and forge commands to access resources to which we are not unauthorized for access. Note that we focus on analyzing Android companion apps because most users operate the TV smart features (e.g., screencast and screenshot) by using their smartphone.

Analysis. To study the protocols for remote interactions, we first capture network traffics transmitted between the smart TV and its companion app with tcpdump. By analyzing network packets, we identify the communication protocol (e.g., MQTT, HTTP and private application layer protocols over TCP or UDP) used for transmitting control commands by Wireshark. According to the communication protocol, we determine which standard APIs [42], [43] should be applied for sending and receiving data.

After retrieving network configurations, we recover remote interaction protocols and identify the authentication fields. Specifically, we utilize JEB and IDA PRO to reverse engineer the companion app. Starting from each argument of identified network APIs, we carry out a backward program slicing to identify the variables that are directly/indirectly data-dependent on the argument to determine how the argument is constructed. Given the Data Dependence Graph, we further identify the authentication variables that are related to access control. If the variable is assigned a constant or a value generated by a pseudo-random number generator or a timestamp, we consider the variable irrelevant to access control since these values do not contain any identity information. Otherwise, when a variable is assigned by a value related to the smart TV (e.g., binding credentials) or a return value of a memory-read function (e.g., `openFileOutput`) we conclude that the variable is relevant to access control.

Attacks. Given the variables that are related to access control, we launch remote interface abusing attacks to execute sensitive operations such as screencast and screenshot. First, we extract how the operation commands are formed and then modify the values of these variables by using the dynamic instrumentation framework Frida [44]. We further send the modified commands via a malicious device. If the smart TV is operated successfully without any warnings and we can access the unauthorized resources arbitrarily, we regard the remote interaction access control of the smart TV as vulnerable.

5 Experimental Results

We launched EvilScreen attacks to test real-world devices and reported observed security flaws.

5.1 Experiment Setup

To investigate whether protections are securely implemented in real-world smart TVs, we tested eight smart TVs manufactured by different well-known manufacturers [45], i.e., Samsung, TCL, Hisense, Xiaomi, Sony, Skyworth, LeTV, and Konka from China, Japan, Korea and United States. Shipments of smart TVs [46] from those manufacturers range from 7 million (Konka) to 48 million (Samsung).

All these eight smart TVs are equipped with smart TV OSes and remote controls. Except for the Samsung TV developed with Tizen OS [47], all manufacturers customize their smart TV OSes on top of Android TV [9]. Since Android TV and Samsung TV are two widely popular smart TVs throughout the world and dominate the global market [48], we focus on them in our experiments. We list technical details of each smart TV and its remote control in Table 1. By default, all eight smart TVs support IR communications and four of them (i.e., Samsung, Xiaomi, LeTV, Sony) also support BLE communications. In our analysis of the remote controls, we found that not all of them are based on IR. Only TCL, Hisense, Skyworth and Konka smart TVs provide IR-based remote controls, whereas the others only provide BLE-based remote controls. An interesting observation is that although the official manuals of Samsung and Xiaomi TVs did not mention about receiving IR signals, we could operate their smart TVs by sending IR commands. This indicates that IR receivers are still integrated in these two smart TVs.

We further analyzed each smart TV by launching the EvilScreen attack. We also reported all the discovered flaws and the consequences to the corresponding manufacturers, and Xiaomi responded with a confirmation before the submission of this paper. Besides, to comply with research ethics guidelines, all the experiments were performed on

our own devices and conducted in our lab testing environment.

5.2 Network Isolation Bypassing

We exploited Wi-Fi provisioning of smart TVs and successfully compromised all the smart TVs, that is, all the smart TVs are authentication dependent on vulnerable channels when provisioning Wi-Fi. Hence, the attacker can utilize other vulnerable wireless channels to crack network isolation. The detailed experimental results are described in what follows.

All eight smart TVs support the IR based Wi-Fi provisioning. By sending the tampered IR signals to the smart TV, we successfully forced all these smart TVs to reconnect to a new (malicious) WLAN. These results show that WLANs can be bypassed because all these IR channels are vulnerable to impersonation attacks. With respect to the BLE channels, the smart TVs of Samsung, Xiaomi, LeTV, and Sony can provision Wi-Fi through BLE remote controls. However, all the involved Wi-Fi credentials were distributed insecurely. We discovered that these four smart TVs adopt the *Legacy Connection* scheme to bind with the remote controls; hence their BLE channels are vulnerable to brute force attacks. We then ran Crackle to decrypt all the communication packets and successfully recovered all the transmitted data including Wi-Fi authentication credentials. We further found that all the four smart TVs are implemented with *Just Works* pairing mode for BLE remote controls binding; as a result, they are also vulnerable to active impersonation attacks. Thus, we successfully connected to these four smart TVs and forced them to reconnect a new malicious network from a fake BLE client.

We found an exceptional case in using BLE for Wi-Fi provisioning: Samsung introduces a companion app, SmartThings, by which users can provision Wi-Fi. The app still sent encoded commands via the BLE channel, and Samsung specifically designed a solution with a customized DTLS protocol to protect it. Nonetheless, we still found a security flaw of this BLE protection, which we discuss in Section 5.6.

5.3 Malicious Remote Control Binding

We conducted a UI differential analysis to analyze the binding information displayed on the smart screen. The results demonstrated that none of these smart TVs implemented a correct binding mechanism because only limited device information was provided and some of the binding information was not even protected. Referring to the binding information, we successfully cheated all users and bound malicious remote controls with the smart TV.

5.3.1 Remote Control Binding

All the eight smart TVs supported either IR-based or BLE-based remote controls. Unfortunately, neither of them was secure (refer to Section 5.2). They all silently connected with the TV without prompting any connection information or warning.

5.3.2 Companion App Binding

We analyzed the binding schemes between companion apps and smart TVs to explore the displayed binding information and whether the binding scheme is protected by secure identity validation.

Information Display. Only Sony and Samsung TVs displayed binding information on their screens. Specifically, Sony TV showed the device name, a pseudo-random pincode, and the remaining time. Instead Samsung TV displayed a pseudo-random pincode (in DTLS over BLE communication) or directly prompted an alert with the device name for users to decide whether to confirm the device binding (in WebSocket over Wi-Fi communication). However, such displayed information was insufficient for users to pinpoint each unique device. Even worse, the other smart TVs (i.e., TCL, Hisense, Xiaomi, Skyworth, LeTV and Konka) accepted the binding requests without showing any alert on their screens, and thus legitimate users were unaware of malicious connections.

Connection Authentication. We successfully sent a forged binding request to connect a malicious remote control to the smart TVs of TCL, Xiaomi, Skyworth, LeTV, and Konka. By inspecting the communication protocols, we found that these smart TVs customized their own protocols without verifying identities of remote controls. For Hisense TV, its binding credential (i.e., username and password) was embedded in the companion app; thus the attacker could obtain the credential by reverse engineering the app and connect with the TV.

Sony and Samsung supported two types of connection authentication. In particular, Sony supported BLE-based binding credential distribution. To bind the remote control with the smart TV, Sony TV generated a token (i.e., a pseudo-random number) for each binding request and then broadcast it via Bluetooth. When its companion app received the token, it automatically connected with the smart TV without notifying the user. Unfortunately, we inspected these tokens and found that were transmitted in plaintext; thus, any apps with the BLE permission could also receive the password and complete the binding. If Bluetooth was turned off, Sony displayed a pseudo-random pincode in four-digit and waited for the user to type in the pincode from the app. However, the token was transmitted through HTTP in plaintext, which is vulnerable to MITM attacks. Even though a secure network connection was established, attackers could still crack the four-digit token by launching brute-force attacks. Similarly, Samsung TV also displayed a pseudo-random pincode in eight-digit when Bluetooth of the companion app was turned on; however the pincode needed to be filled in manually. On the other hand, the companion app could send its binding request through Websocket over Wi-Fi communications. Nevertheless, Saumsung TV created the TLS connection without checking its certificate, which is vulnerable to impersonation attacks.

5.4 Remote Interface Abusing

By analyzing remote user interfaces supported by each smart TV, we compromised all the smart TVs successfully and accessed unauthorized resources arbitrarily. Our results thus indicate that all smart TVs did not grant permissions properly.

5.4.1 Screen Operation Through Companion app

The companion apps of all tested smart TVs could be used as a remote control for controlling the cursor. Except for Samsung TV, the other smart TVs could also perform app

This article has been accepted for publication in IEEE Transactions on Dependable and Secure Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2023.3286182

TABLE 1
Implementation features of remote controls related to our analyzed smart TVs

TV	Wi-Fi Provisioning	Remote Control Binding			User Interactions			
		Channel	Protocol	Attestation	CA as RC	TV App Operation	Screencast	Screenshot
Samsung	IR + BLE + CA	BLE	UDP(DTLS)	Pincode	✓	✗	✗	✗
		TCP	WebSockets	Prompt+Confirmation				
TCL	IR	TCP	Proprietary	User operating	✓	Open/Install/Uninstall	✓	✓
Hisense	IR	TCP	MQTT	Password	✓	Open/Install/Uninstall	✓	✓
Xiaomi	IR + BLE	TCP	Proprietary	User operating	✓	Open/Install/Uninstall	✓	✓
Sony	IR + BLE	TCP	Proprietary	Pincode/password	✓	Open	✗	✗
Skyworth	IR	TCP	WebSocket	User operating	✓	Open/Install/Uninstall	✓	✓
LeTV	IR + BLE	UDP	Proprietary	User operating	✓	Open	✗	✓
Konka	IR	TCP	Proprietary	User operating	✓	Open/Install/Uninstall	✓	✓

operations (i.e., execution, installation, and uninstallation). By analyzing the communication traffic, we noticed that commands sent by the companion apps of Hisense, Xiaomi, Sony, LeTV and Konka were transmitted in plaintext. Although the TCL companion app used the AES algorithm to encrypt the transmitted commands, we found that its encryption key was encoded in the companion app; thus attackers could retrieve the key by reverse engineering.

Moreover, we discovered that the smart TVs of TCL, Hisense, Xiaomi, Skyworh, LeTV and Konka accepted all commands without checking their validity. Therefore, we tampered the commands to execute malicious operations such as malware installation. Although Sony TV verified the commands by checking a cookie shared beforehand, attackers could extract the cookie by intercepting communication packets. By inspecting the packets manually, we found that the commands of app downloading sent by the companion apps of the Hisense and Konka TVs contained URLs. The smart TVs further downloaded app installation packets from the URLs instead of official app stores. Thus, we replaced the URL with the one of a malicious app installation packets and sent it to the smart TV. The smart TV successfully downloaded and installed the malicious app.

5.4.2 Screencast

Five smart TVs, i.e., TCL, Hisense, Xiaomi, Skyworth and Konka, provided media files (i.e., video, photos, and music) and documents (e.g., ppt, pdf, txt) casting service (from the smartphone to the TV). Instead of transmitting files directly, URLs for downloading the files were delivered to the smart TVs. All these smart TVs displayed the received files directly without checking whether the sender was authorized for screencast. Apart from the TCL TV, the screencast procedure of the other smart TVs is not protected, that is, the URLs were not encrypted during transmission and the integrity of these URLs was not verified. For the TCL TV, the communications were encrypted by the AES algorithm whose encryption key is embedded in the companion app.

Not only did the smart TVs suffer from remote interaction abuse, so did the corresponding companion apps. Therefore, we could easily access the sensitive files on the smartphone by capturing the packets for the screencast service to obtain the URLs.

5.4.3 Screenshot

Six smart TVs (i.e., TCL, Hisense, Xiaomi Skyworth, LeTV, and Konka) support interfaces for screenshot. Similar to screencast, the TCL, Xiaomi, Skyworth, and Konka TVs sent a URL from which to download the screenshot instead of sending the screenshot images. The corresponding companion apps further obtained the screenshot images through the received URLs. On the contrary, Hisense and LeTV TVs directly transmitted the screenshot images back to their companion apps. In addition, Xiaomi TV provided an interface which synchronizes the screen content to the companion app.

Nonetheless, we successfully launched screen hijacking attacks by continuously sending screenshot requests and then monitoring the screen contents watched by the user. Our results show that none of them provides authorization mechanism to protect screenshot images.

5.5 Attack Efficiency

EVILSCREEN involves three steps and only if all the three steps are launched successfully can an attacker hijack the TV screen. Since EVILSCREEN exploits multi-channel communication and requires stealthiness, there two key requirements for EVILSCREEN to be successful are: (i) the attack distance from the victim must be short to guarantee signal transmission and (ii) the time required must be small to avoid being noticed.

Distance. The EVILSCREEN attack can be performed remotely, as discussed in Section 3, but there are still requirements for the location of the attacker (i.e., attack launched without relay devices) or the relay device since we have to exploit vulnerable IR and BLE channels in the first and second step. Since both the IR and BLE communication techniques are designed for short-range wireless communications, the attacker or relay device should be close to the smart TV. Based on our experiments, we claim that the attack distance should be within 10 meters. Considering the usage scenarios of both public and personal smart TVs [49], such a distance requirement is easily satisfied in real-world scenarios. In this way, the attacker can receive enough data to extract credentials or send data to communicate with the TV in a short time, thus bypassing network isolation, creating the binding and performing further attacks.

Time. We discuss the time to perform an EVILSCREEN attack in two parts, i.e., pre-analysis and attack process.

Pre-analysis of smart TV weaknesses should be done before launching an attack, so that one can determine if the smart TV is vulnerable to the EVILSCREEN attack. Therefore, the pre-analysis time does not affect the attack effectiveness. During the pre-analysis process, manual efforts are

usually involved to forge messages or perform app reverse engineering. But carrying out the pre-analysis takes a short time because smart TVs do not adopt complicated communication protocols due to the restricted resources on remote control devices, and to the fact that companion apps are not usually protected by packing and obfuscation techniques either. In our experiments, the time for the pre-analysis took no longer than 3 hours.

We launch the EVILSCREEN attack after the pre-analysis. Two major operations are involved, i.e., sending and receiving messages, and brute forcing to crack communications and recover credentials. Obviously, brute force cracking takes much longer time than data transmission. Hence whether the brute force cracking to extract credentials can be executed in a short time is the factor determining whether the attack can be successfully launched. In the EVILSCREEN attack, brute force cracking techniques are used in the first and second steps. When bypassing network isolation in the first step, we need to decode or decrypt IR and BLE packets to check whether there is an authentication dependency. Specifically, we iterate over the existing encoding schemes to decode IR signals and utilize Crackle to decrypt BLE packets. Both of those two processes take less than 1 second. In the second step, we perform a brute force cracking on the binding tokens if their entropy is not large enough. If the binding token is 8-digits, it will take no longer than 40 seconds to recover the token. We give a detailed case study for Samsung smart TV in Section 5.6.

5.6 Case Study: Samsung Smart TV

Considering Samsung smart TV as an example, we now describe in details how an EVILSCREEN attack was launched to exploit security flaws in the smart TV remote control binding. In particular, the binding between the companion app, SmartThings[2] [50], and the TV is regulated by the Open Connectivity Foundation (OCF) security specification of ISO/IEC 30118-1:2018 [16]; however the security controls implemented based on this specification are not adequate due to the presence of "smart" user interfaces. The EVILSCREEN attack was able to crack both device binding mechanisms, that is, BLE based binding and Wi-Fi based binding.

5.6.1 BLE based Binding Attack

The communication protocol between SmartThings and the smart TV is a customized DTLS protocol, built on top of Bluetooth 4.2. The detailed binding authentication is illustrated in Figure 3. First, SmartThings sends its device UUID as a binding request to the smart TV. Such a request is sent over an unprotected BLE channel. When the request is received, the smart TV generates a pseudo-random number (PRN) of eight digits and displays the number on its screen. The smart TV then regards the eight digits PRN as a pincode and calculates a pre-shared key (PSK) according to the following expression:

$$PSK = PBKDF2 \text{ (HMAC\_SHA256, } PIN\text{, UUID, c, dKLen)}$$

2. SmartThings is an official companion app developed by Samsung for home automation. It can connect with Samsung smart devices and control them. These devices include bulbs, speakers and smart TVs, etc.

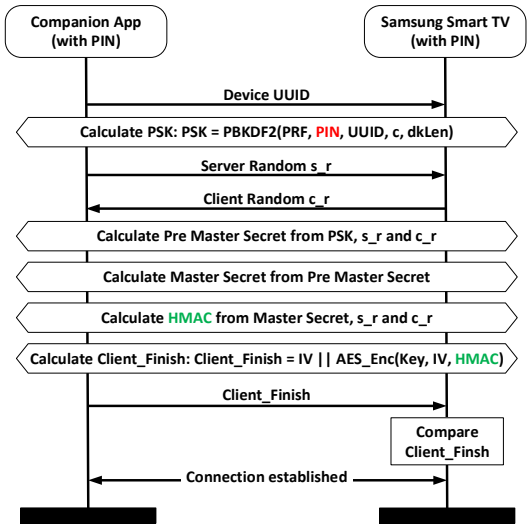

Fig. 3. Connection establishment between Samsung Smart TV and its companion app

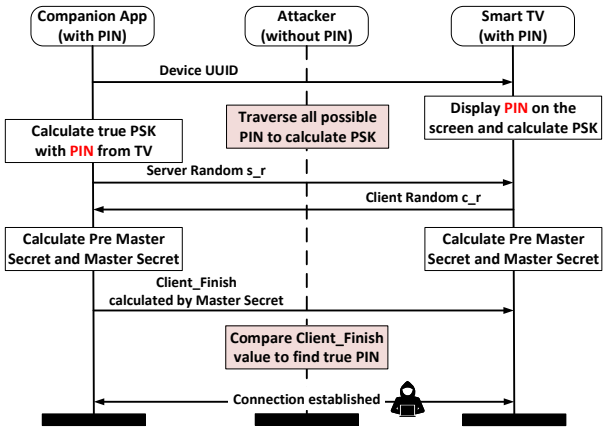

Fig. 4. MITM Attacks on Samsung Smart TV and its companion app

The calculation relies on the PBKDF2 algorithm. In this algorithm, the only secret field is the pincode (i.e., the PRN). Simultaneously, the user types in the same pincode on SmartThings to generate the same PSK. When both PSKs are confirmed to be the same, the smart TV and the user generate two pseudo-random numbers, *s_random* and *c_random*, respectively, and further calculate a pre-master secret (PMS) and a master secret (MS) according to the following expression:

$$PMS = TLS\_ECDHE\_PSK\ (PSK\text{, s\_random, c\_random})$$
$$MS = PRF\ \text{(HMAC\_SHA256, } PMS\text{, Padding)}$$

for generating a standard TLS session key. Finally, SmartThings sends a message containing the HMAC-SHA256 digest to establish the TLS connection for binding credential distribution.

The only issue in this authentication protocol is the usage of the eight digits pincode. The security specification in OCF is to choose a PRN with enough entropy as the input of the PBKDF2 algorithm. However, Samsung smart TV reduces the search space against PSK to 10^8 which is breakable within a short period of time. To exploit such a flaw, we launched a MITM attack against the binding

authentication demonstrated in Figure 4. We first monitored the BLE communication between SmartThings and the smart TV. When a device UUID was detected, we pre-calculated all possible PSKs within the search range of 10^8 and the corresponding MSs. By using the Client_Finish packet, we tried all the possible PSKs as the secret key to decrypt the packet until an effective key was identified.

We conducted the key search by utilizing a 2080 Ti GPU to speed up the key search with hashcat [51]. The entire guessing took only 40 seconds on average, which shows that the attacker definitely has enough time to retrieve the PSK and reconnect the smart TV to a malicious device. Although the PSK cannot be cracked within a restricted period, we could still obtain the PSK by launching offline attacks and then decrypting the transmitted messages.

5.6.2 Wi-Fi based Binding Attack

Prior to binding, SmartThings and the smart TV were connected to the same WLAN. SmartThings first sent a token request to the smart TV through a Websocket communication using the TLS protocol. A request with a specific binding URL was constructed according to the format `wss:{IP}:{PORT}?name={DEVICE}&token={TOKEN}`. When the URL was first accessed, a window with the device information popped up on the TV screen and asked whether the mobile device was allowed to connect to the TV. If it was allowed, a token was then sent back to SmartThings.

Nevertheless, the device name displayed on the TV screen can be obtained by the attacker easily. Then we utilized a malicious device to request for the TLS connection. To construct a WebSocket request, we modified the *name* field by using the displayed device name. As there is no warning message indicating to the user that a remote control has connected, the user will be easily misled (by thinking the previous connection to be unsuccessful) to accept the request and connect the smart TV with the malicious device.

5.7 Discussion

Although several security flaws of the smart TVs were exploited, some constraints may limit the capabilities of the EVILSCREEN attack.

Prerequisite information. Since the EVILSCREEN attack needs to be launched stealthily, the attacker needs to learn detailed information of a smart TV, e.g., device name, BLE pairing mode, displayed binding information by either purchasing the same brand of smart TVs or (physically/virtually) entering the private property to check the corresponding smart TV.

Activity constraint. Attacks' capability will be affected if the smart TV is switched off, in which case the smart TV is disconnected from any wireless channels. In this case, the attack is not possible as the attack requires at least one of the wireless channels on the smart TV to be active. However, when the smart TV is turned on, by using EVILSCREEN the attacker can not only eavesdrop user's private data but also actively attack the victim TV.

6 COUNTERMEASURES

Based on our findings, we suggest that TV manufacturers improve their protection schemes as follows.

6.1 Securing network connection

Transmit Wi-Fi credentials with SmartCfg. The root cause of EVILSCREEN attack is the involvement of multiple wireless channels. Among the supported wireless channels, IR and BLE are the most vulnerable; thus it is essential to avoid transmitting sensitive information (e.g., Wi-Fi credentials) through these channels. SmartCfg is a popular mechanism to facilitate Wi-Fi connection for smart devices. This mechanism simply transmits credential information via specific broadcasting packets. Hence, by deploying a secure SmartCfg solution [52] for network connections, one can ensure that Wi-Fi credentials are distributed within the WLAN with confidentiality guarantees.

Strengthen the security of IR and BLE channels. Another approach to secure network connections and avoid credential leakage is to enhance the security mechanisms used in the IR and BLE channels. For instance, smart TV vendors should consider encryption for IR communication [53] to protect data from eavesdropping. As for BLE communication, secure pairing and connection methods [22], [54] should be employed.

6.2 Securing remote control binding

Force physical access when first remote control binding. A user should be required to physically access the TV when executing the first pairing. In this way the smart TV can verify the ownership to the maximum extent. The subsequent guest bindings should be confirmed by the first bound device (i.e., the authenticated user) to prevent attacks of which the user is unaware.

Limit the number of connected remote controls. It is essential for manufacturers to limit the number of remote controls that are allowed to connect with the smart TV. For instance, when two remote controls are originally provided by a manufacturer, the smart TV can then be limited to bind with remote controls up to two. The best would be to bind with one remote control only at each time.

Apply multi-factor authentication. Instead of relying on the binding credential only, the smart TV could authenticate the remote control through multiple factors such as both binding credential and user confirmation. For instance, the smart TV could distribute a binding credential to the remote control and provide a reminder for the user with the binding details simultaneously.

Avoid transmitting credentials via unprotected channels. Transmitting credentials in plaintext or embedding them in companion apps must be forbidden to avoid MITM attacks. It is essential to strengthen the credential protection scheme such as utilizing mutual authentication and end-to-end encryption.

6.3 Securing Remote Interaction

Adopt fine-grained authorization. A fine-grained authorization scheme is necessary to protect sensitive remote interfaces on smart TVs and an authenticated user (i.e., the owner) should be involved in the authorization process. To be specific, the smart TV OSes should be designed so as to constrain the operations that can be executed by the remote controls. When a remote control is bound to the smart TV,

least privileges should be authorized and advanced privileges for sensitive operations should be further requested and confirmed by the owner of the smart TV.

7 Related Work

Smart TV Security. Past research has uncovered many security issues in smart TV communications. A recent work [55] has demonstrated a less powerful attack model for smart TV by inferring information from IR communication patterns, which would lead to privacy issues such as inferring viewers' interests and activities. Moghaddam *et al.* [6] focused on privacy practices of Over-the-Top streaming devices (e.g., smart TVs) and discovered that these devices collect users' private information and behavior habits for advertising and tracking. Michéle *et al.* [5] explored a weakness in the multimedia layer of TV which allow attackers to gain full control without physical access to the TV. Bachy *et al.* [1], [2] contributed to the security threats of ADSL and DVB network. Apart from communications, Aafer *et al.* [56] developed a fuzzing method to dynamically evaluate Android smart TVs, in which they focused on custom public APIs in TV OS and utilized log analysis to infer API input specification.

Unlike attacks that utilize low-level code implementation bugs to compromise smart TV systems (e.g., Android TV OS), our EvilScreen attack is more general because it exploits weaknesses of the existing smart TV muti-channel designs. And instead of focusing on single network channel, we advance previous works by combining less powerful and yet not-well-protected IR and BLE signals and the powerful Wi-Fi based remote control (including the companion app) operations to fulfill a more severe attack against many modern smart TVs, which can not only result in privacy information leakage but also actively hijacking smart TV screen.

Short-range Wireless Communication Security. Short-range wireless techniques (e.g., IR and BLE) are widely used but very few of them are implemented securely. Zhou *et al.* [17] investigated the potential security issues in IoT devices supporting infrared remote control and observed sensitive data leakage. Ryan *et al.* [18] proved that some security flaws in BLE could make it easy for attackers to implement eavesdropping attacks. In addition, Garbelini *et al.* [57] developed a systematic automated fuzzing framework for BLE protocol to discover insecure implementation behaviour. Some works have also focused on the security of Wi-Fi schemes, especially Wi-Fi provisioning. Li *et al.* [52] conducted a security analysis against eight different Wi-Fi provisioning solutions and found out that unsafe transmission in Wi-Fi smart configuration could lead to password disclosure and other issues. Liu *et al.* [58] proposed a new Wi-Fi connection method based on audio waves. Wang [59] proposed a solution for IoT provisioning scheme with universal cryptographic tokens.

Though such previous works have raised security issues for short-range communication also used for smart TVs, exploiting these wireless channel vulnerabilities alone may not cause result in severe security threats. Instead, an EvilScreen attack, by combining vulnerable wireless communications with various remote interfaces provided by smart TVs, could hijack a victim smart TV and monitor user behaviors and habits.

8 Conclusion

In this paper, we systematically analyzed the security of wireless communications between smart TVs and their remote controls. Based on this analysis we proposed a new attack, EvilScreen attack, which exploits insecure multi-channel remote control communication. By executing different remote control commands with multiple wireless channels in a sophisticated way, our attack allows the attacker to access and modify resources on the victim TVs. We have reported security issues of eight popular smart TVs, which are vulnerable to EvilScreen attack, to the corresponding manufacturers, and suggested countermeasures to help address these issues.

References

[1] Y. Bachy, F. Basse, V. Nicomette, E. Alata, M. Kaâniche, J.-C. Courrege, and P. Lukjanenko, "Smart-tv security analysis: practical experiments," in *2015 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*. IEEE, 2015, pp. 497–504.

[2] Y. Bachy, V. Nicomette, M. Kaâniche, and E. Alata, "Smart-tv security: risk analysis and experiments on smart-tv communication channels," *Journal of Computer Virology and Hacking Techniques*, vol. 15, no. 1, pp. 61–76, 2019.

[3] M. Ghiglieri, M. Volkamer, and K. Renaud, "Exploring consumers' attitudes of smart tv related privacy risks," in *International Conference on Human Aspects of Information Security, Privacy, and Trust*. Springer, 2017, pp. 656–674.

[4] S. Kang and S. Kim, "How to obtain common criteria certification of smart tv for home iot security and reliability," *Symmetry*, vol. 9, no. 10, p. 233, 2017.

[5] B. Michéle and A. Karpow, "Watch and be watched: Compromising all smart tv generations," in *2014 IEEE 11th Consumer Communications and Networking Conference (CCNC)*. IEEE, 2014, pp. 351–356.

[6] H. Mohajeri Moghaddam, G. Acar, B. Burgess, A. Mathur, D. Y. Huang, N. Feamster, E. W. Felten, P. Mittal, and A. Narayanan, "Watching you watch: The tracking ecosystem of over-the-top tv streaming devices," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, 2019, pp. 131–147.

[7] M. Niemietz, J. Somorovsky, C. Mainka, and J. Schwenk, "Not so smart: On smart tv apps," in *2015 International Workshop on Secure Internet of Things (SIoT)*. IEEE, 2015, pp. 72–81.

[8] N. Sidiropoulos and P. Stefopoulos, "Smart tv hacking," *Research project*, vol. 1, pp. 2012–2013, 2013.

[9] "Android TV," https://www.android.com/tv/.

[10] "Wi-Fi Provisioning," https://www.wi-fi.org/download.php?file=/sites/default/files/private/Device_Provisioning_Protocol_Specification_v1.1_1.pdf.

[11] G. Zhang, C. Yan, X. Ji, T. Zhang, T. Zhang, and W. Xu, "Dolphinattack: Inaudible voice commands," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 103–117.

[12] N. Roy, S. Shen, H. Hassanieh, and R. R. Choudhury, "Inaudible voice commands: The long-range attack and defense," in *15th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 18)*, 2018, pp. 547–560.

[13] T. Sugawara, B. Cyr, S. Rampazzi, D. Genkin, and K. Fu, "Light commands: laser-based audio injection attacks on voice-controllable systems," in *29th {USENIX} Security Symposium ({USENIX} Security 20)*, 2020, pp. 2631–2648.

[14] D. Antonioli, N. O. Tippenhauer, and K. Rasmussen, "Bias: Bluetooth impersonation attacks," in *Proceedings of the IEEE Symposium on Security and Privacy (S&P)*, 2020.

[15] M. Vanhoef and F. Piessens, "Key reinstallation attacks: Forcing nonce reuse in wpa2," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 1313–1328.

[16] "Open Connectivity Foundation (OCF) Specification," https://www.iso.org/standard/53238.html.

[17] Z. Zhou, W. Zhang, S. Li, and N. Yu, "Potential risk of iot device supporting ir remote control," *Computer Networks*, vol. 148, pp. 307–317, 2019.

[18] M. Ryan, "Bluetooth: With low energy comes low security," in *7th USENIX Workshop on Offensive Technologies (WOOT 13)*, 2013.

[19] W. K. Zegeye, "Exploiting bluetooth low energy pairing vulnerability in telemedicine." International Foundation for Telemetering, 2015.

[20] T. Melamed, "An active man-in-the-middle attack on bluetooth smart devices," *Safety and Security Studies*, vol. 15, p. 2018, 2018.

[21] C. Zuo, H. Wen, Z. Lin, and Y. Zhang, "Automatic fingerprinting of vulnerable ble iot devices with static uuids from mobile apps," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, 2019, pp. 1469–1483.

[22] Y. Zhang, J. Weng, R. Dey, Y. Jin, Z. Lin, and X. Fu, "Breaking secure pairing of bluetooth low energy using downgrade attacks," in *29th USENIX Security Symposium (USENIX Security 20)*, 2020, pp. 37–54.

[23] J. Wu, Y. Nan, V. Kumar, D. J. Tian, A. Bianchi, M. Payer, and D. Xu, "BLESA: Spoofing attacks against reconnections in bluetooth low energy," in *14th USENIX Workshop on Offensive Technologies (WOOT 20)*, 2020.

[24] M. von Tschirschnitz, L. Peuckert, F. Franzen, and J. Grossklags, "Method confusion attack on bluetooth pairing," in *2021 IEEE Symposium on Security and Privacy (SP)*, 2021.

[25] "Wi-Fi Protected Access," https://en.wikipedia.org/wiki/Wi-Fi_Protected_Access.

[26] J. Chen, C. Zuo, W. Diao, S. Dong, Q. Zhao, M. Sun, Z. Lin, Y. Zhang, and K. Zhang, "Your iots are (not) mine: On the remote binding between iot devices and users," in *2019 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. IEEE, 2019, pp. 222–233.

[27] Q. Xu, R. Zheng, W. Saad, and Z. Han, "Device fingerprinting in wireless networks: Challenges and opportunities," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 94–104, 2015.

[28] X. Xu and J. B. Peterson, "Differences in media preference mediate the link between personality and political orientation," *Political Psychology*, vol. 38, no. 1, pp. 55–72, 2017.

[29] J. B. Weaver III, "Exploring the links between personality and media preferences," *Personality and individual differences*, vol. 12, no. 12, pp. 1293–1299, 1991.

[30] "How to purchase movies and TV shows on YouTube on smart TVs," https://support.google.com/youtube/answer/9676953?hl=en.

[31] "How to watch Disney+ on your TV," https://help.disneyplus.com/csp?id=csp_article_content&sys_kb_id=94f0bf3a872f19904e3f7557cebb3593.

[32] "LIRC," https://www.lirc.org/.

[33] "IRDB," https://github.com/probonopd/irdb.

[34] "Universal remote control," https://play.google.com/store/apps/details?id=codematics.universal.tv.remote.control&hl=en_US&gl=US.

[35] "IR Blaster," https://www.androidauthority.com/phones-with-ir-blaster-858845/.

[36] "Orvibo WiFi IR Remote Control," https://www.orvibo.com/en/product/xiaofang.html.

[37] "TI CC1352," https://www.ti.com/product/CC1352R.

[38] "Crackle," http://lacklustre.net/projects/crackle/.

[39] "BLE Core Specifications," https://www.bluetooth.com/specifications/bluetooth-core-specification/.

[40] "Burp suite," https://portswigger.net/burp.

[41] "Owasp authorization cheat sheet," https://cheatsheetseries.owasp.org/cheatsheets/Authorization_Cheat_Sheet.html.

[42] "Java api specifications," https://docs.oracle.com/en/java/javase/16/docs/api/index.html.

[43] "Linux man page," https://linux.die.net/man/.

[44] "Frida," https://frida.re/.

[45] "TV unit shipments worldwide from 4Q'15 to 1Q'20, by vendor," https://www.statista.com/statistics/667034/lcd-tv-shipments-worldwide-by-vendor/.

[46] "Smart & connected tvs - statistics & facts," https://www.statista.com/topics/4761/smart-and-connected-tvs/.

[47] "Tizen," https://www.tizen.org/.

[48] "Market share of leading tv manufacturers worldwide from 2019 to 2022," https://www.statista.com/statistics/1266988/global-leading-manufacturers-tv-market-share-sales-volume/.

[49] "Average Size of a Living Room," https://www.homenish.com/average-size-living-room/.

[50] "SmartThings," https://www.smartthings.com/.

[51] "Hashcat," https://github.com/hashcat/hashcat.

[52] C. Li, Q. Cai, J. Li, H. Liu, Y. Zhang, D. Gu, and Y. Yu, "Passwords in the air: Harvesting wi-fi credentials from smartcfg provisioning," in *Proceedings of the 11th ACM Conference on Security & Privacy in Wireless and Mobile Networks*, 2018, pp. 1–11.

[53] M. Kim and T. Suh, "Eavesdropping vulnerability and countermeasure in infrared communication for iot devices," *Sensors*, vol. 21, no. 24, p. 8207, 2021.

[54] M. Cäsar, T. Pawelke, J. Steffan, and G. Terhorst, "A survey on bluetooth low energy security and privacy," *Computer Networks*, vol. 205, p. 108712, 2022.

[55] K. Huang, Y. Zhou, K. Zhang, J. Xu, J. Chen, D. Tang, and K. Zhang, "Homespy: The invisible sniffer of infrared remote control of smart tvs."

[56] Y. Aafer, W. You, Y. Sun, Y. Shi, X. Zhang, and H. Yin, "Android smarttvs vulnerability discovery via log-guided fuzzing." in *USENIX Security Symposium*, 2021, pp. 2759–2776.

[57] M. E. Garbelini, C. Wang, S. Chattopadhyay, S. Sumei, and E. Kurniawan, "Sweyntooth: Unleashing mayhem over bluetooth low energy," in *2020 USENIX Annual Technical Conference (USENIX ATC 20)*, 2020, pp. 911–925.

[58] L. Liu, Z. Han, L. Fang, and Z. Ma, "Tell the device password: Smart device wi-fi connection based on audio waves," *Sensors*, vol. 19, no. 3, p. 618, 2019.

[59] W. Kang, "U2fi: A provisioning scheme of iot devices with universal cryptographic tokens," *arXiv preprint arXiv:1906.06009*, 2019.

Yiwei Zhang is currently a first-year Ph.D. student in Computer Science at Purdue University. She received a Master degree in Computer Science from Shanghai Jiao Tong University in 2022. Her research interests include IoT (Smart Homes) security and software security.

Siqi Ma received the B.S. degree in computer science from Xidian University, Xi'an, China in 2013 and Ph.D. degree in information system from Singapore Management University in 2018, respectively. She is currently a senior lecturer of the University of New South Wales Canberra, Australia and was a research fellow of distinguished system security group from CSIRO. Her research interests include data security, IoT security and software security.

Tiancheng Chen received the B.S. degree from Hangzhou Dianzi University, Hangzhou, China in 2017 and Master degree in Computer Science at Shanghai Jiao Tong University in 2020. His research interests include Android security and program analysis.

Juanru Li is the director of G.O.S.S.I.P, Shanghai Jiao Tong University. He received his Ph.D. degree from Shanghai Jiao Tong University in 2019. His research interests include developing secure crypto software and systems, and hardening existing crypto software against evolving security threats.

Robert H. Deng (F'16) is AXA Chair Professor of Cybersecurity and Professor of Information Systems in the School of Information Systems, Singapore Management University since 2004. His research interests include data security and privacy, multimedia security, network and system security. He served/is serving on the editorial boards of many international journals, including TIFS, TDSC. He is a fellow of the IEEE.

Elisa Bertino (F'02) is a Professor of computer science at Purdue University. She is currently the Director of the Purdue Cyberspace Security Laboratory (Cyber2Slab), where she leads multi-disciplinary research in data security and privacy. Her recent research focuses on security of IoT systems and cellular network protocols, and machine learning techniques for cybersecurity. She is a Fellow member of ACM, IEEE and AAAS. She received the 2002 IEEE Computer Society Technical Achievement Award for outstanding contributions to database systems and database security and advanced data management systems and the 2005 IEEE Computer Society Tsutomu Kanai Award for pioneering and innovative research contributions to secure distributed systems.