HomeSpy: The Invisible Sniffer of Infrared Remote Control of Smart TVs

Kong Huang[1], YuTong Zhou[1], Ke Zhang[1], Jiacen Xu[2], Jiongyi Chen[3], Di Tang[4], and Kehuan Zhang[1]

[1]*The Chinese University of Hong Kong*,
[2]*University of California, Irvine*,
[3]*National University of Defense Technology*,
[4]*Indiana University Bloomington*

Abstract

Infrared (IR) remote control is a widely used technology at home due to its simplicity and low cost. Most considered it to be "secure" because of the line-of-sight usage within the home. In this paper, we revisit the security of IR remote control schemes and examine their security assumptions under the settings of internet-connected smart homes. We focus on two specific questions: (1) whether IR signals could be sniffed by an IoT device; and (2) what information could be leaked out through the sniffed IR control signals.

To answer these questions, we design a sniff module using a commercial-off-the-shelf IR receiver on a Raspberry Pi and show that the Infrared (IR) signal emanating from the remote control of a Smart TV can be captured by one of the nearby IoT devices, for example, a smart air-conditioner, even the signal is not aimed at the air-conditioner. The IR signal range and receiving angle are larger than most have thought. We also developed algorithms to extract semantic information from the sniffed IR control signals, and evaluated with real-world applications. The results showed that lots of sensitive information could be leaked out through the sniffed IR control signals, including account name and password, PIN code, and even payment information.

1 Introduction

Infrared (IR) has been widely used at home to control consumer electronics since a long time ago, due to the cheap and simple implementations that are broadly available. IR remote control was introduced by a Canadian cable company Viewstar in 1980 [10], and has been shipped together with every TV since then. In addition to TV, the affordable IR technology has also been added to other household electronic appliances, like air-conditioners, vacuums, lighting, etc. By 1990, 90 percent of U.S. households were using remote controls to manage their media. In 2010, a research "Global Remote Control Trends Study" showed that a majority of households in France, Australia, and the United Kingdom possess four or more devices in the living room that need remote controls.

Despite the active research in improving IR technologies, we found the security aspect has been under-studied. In fact, the IR communication was not considered as an attractive target due to a few reasons. Firstly, the infrared light will propagate in a straight line and the signal strength will be weakened after even a single reflection. As a result, it is usually used for *Line of Sight (LOS)* communication with a limited range of 10 meters. We may all have common experiences that TV does not respond to our commands if we are not pointing the IR controller directly at the TV or if we are far away from it. As such, an attacker who wants to sniff the IR signals has to be positioned physically close to the victim, in the same room. Secondly, the information carried by IR is usually not sensitive, e.g., the key presses on a TV controller like 'UP" and "DOWN". To the extent that that the IR remote control signals are not encrypted and all commands are sent in clear format.

However, we argue that the security model of IR should be revisited due to the recent development of smart home devices and apps. Firstly, nowadays many home devices have been turned to the Internet of Things (IoT). Many of the IoT devices are by nature IR receivers and they can also be remotely controlled through the Internet. IoT devices also have a bad reputation of security, and they could be easily compromised, as suggested by many cyber-attacks [12,16,36] and studies [7,29,65] on IoT. As such, to steal information from the IR channel, the attacker can remotely control an IoT device in the same room as the victim. Secondly, a lot of home devices are now keeping users' accounts for personalized services. For example, a smart TV could run many TV apps like Netflix, which all require a user to enter his/her account information. Entering with an IR remote controller is still a primary approach. Therefore, IR has a much higher chance now to carry sensitive information.

In this work, we perform the first study to understand the security implications of IR communication through the lens of IoT and smart TV. We aim to answer two important questions: (1) Is it possible for an IoT device to sniff smart TV IR remote control signals, even when it is *not on the path* between TV and controller? (2) If the IR signals can be sniffed, what can an attacker learn from the signal?

To answer the first question, we developed an attack prototype named HOMESPY. Without sacrificing the validity for evaluation, we built an IR sniffer using a Raspberry Pi 3 integrating an IR sensor VS1838b, which is comparable to the Commercial-Of-The-Shelf (COTS) IR receiver widely used by IoT devices. Surprisingly, we found such a low-cost IR receiver is able to sense IR signals that are transmitted off the path or even reflected by walls. Yet, the COTS IR receiver has more limited sensitivity and only digital signals 0/1 are outputted, which led to unsatisfactory recovery accuracy. To address this issue, we develop a novel recovery method leveraging the redundant information embedded in the IR protocols like NEC [41] and Sony SIRC [54], to correct the recovery errors. By design, remote control protocols have built-in the feature to repeat the command to improve reception quality of the targeted consumer devices at home. These repeated IR signals in our study helped improve the sensitivity by as much as 5.8%. After that, we develop another module to decode the command (e.g., "UP" and "DOWN") from the 0/1 time-series. In particular, we construct a large-scale IR code database, covering *75,901* IR codes of *1,303* devices. Given a 0/1 time-series, the corresponding device and the command can be quickly identified. We also found that there exists a one-to-many mapping for some IR signals, causing ambiguous command interpretation. We address this issue by context-aware time-series analysis and majority voting.

To answer the second question about the information leakage from IR communication, we focus on the case when a user attempts to input his/her personal information through the remote controller to smart TV. As surveyed in Section 2.1, the majority of TV apps supports information entry with remote controller, and some even allow in-app purchases for digital or physical goods. In fact, we found user's email, passwords, PIN codes, and payment information can all be embedded in IR communication. Yet, it is not trivial to recover the semantics (e.g., password) from the IR commands like "UP" and "DOWN". Our key insight is that *on-screen virtual keyboard* has to be brought up, whose layout has limited variations, so the mapping between IR commands and characters selected by a user can be determined a priori. We also develop a novel semantic extraction method to find the key sequence that is related to virtual keyboard input and recover the user's sensitive information.

To evaluate the effectiveness of HOMESPY, we deploy the IR sniffer in rooms with different layouts, and examine the inference accuracy of different types of user information (e.g., emails, passwords, and PIN). Here we highlight some results. Firstly, we examined in total 18 positions of 4 room layouts to mount the sniffer and found the most of the positions can correctly receive the IR key signals (ranging from 74.4% to 96.7% by room layouts). Secondly, on the data collected from 5 experiment volunteers, we found the candidate strings outputted by HOMESPY has on average 77% top-5 accuracy for login credential, suggesting the attack is practical.

Contributions. We summarize the contributions below:

- **New study about IR security.** To the best of our knowledge, we present the first study about sniffing IR signals in a smart home environment. We show that the rise of IoT devices with IR receiving capabilities and TV apps leads to new security threats.
- **New attack methods for IR information extraction.** We build HOMESPY, which consists of new methods for IR signal sniffing, command decoding, and semantic extraction, addressing multiple technical challenges.
- **Attack evaluation.** We evaluate HOMESPY in realistic settings with multiple room layouts, experiment participants, and different types of sensitive information. The result shows HOMESPY is effective.

In the end, our work refreshes the existing view on the security of the "simple and safe" IR remote control scheme, reminds the general public about the potential risk of data leakage through IR remote channels of Smart TV and also urges IoT system vendors to come up with mitigation measures. We have also created a video demo [1] to demonstrate HOMESPY is practical in real-world settings.

2 Background

In this section, we first introduce the basics of infrared and how it is used in consumer products. Then, we overview the functionalities provided by smart TV, the main attack target of this study, and its login methods. Finally, we summarize the security and privacy implications of infrared.

2.1 Infrared in Consumer Electronics

Consumer IR and Protocols. Infrared (IR) is a type of electromagnetic radiation whose wave length ranges from 700nm to 1mm. Consumer IR (CIR) deals with the control of various devices and typically uses wavelength from 870nm to 950nm [47]. Comparing to other transmission media like radio, IR technology has advantages including easy and cheap to implement, free of license, and hard to interfere with. Regarding its disadvantages, IR is low in bandwidth and transmission rates, easy to be shielded, and cannot penetrate walls. Hence, IR is usually used for Line of Sight (LOS), short-range communication, carrying small-sized command inputs for consumer products.

Though the consumer IR protocols have *not been standardized*, the manufacturers of consumer appliances often use the same protocol on a large number of similar devices. The

[1] https://sites.google.com/view/homespydemo

most commonly used CIR protocols are NEC, Sony SIRC and Philips RC-5 [47]. In Appendix A, we describe two examples about how IR signals are encoded.

Home devices with IR. IR is known for its wide use in controlling TV. In recent years, it has also been adopted in a broad range of devices in the smart home. We compile a sample list of these devices based on their characteristics of internet connectivity, control technology and power supply in Table 1. In fact, many of these devices are also ideal targets for the attacker to sniff IR signals. For example, entertainment systems like streaming media device, smart Hi-Fi/soundbar, smart DVD/BlueRay player are often connected to or placed next to the smart TV, hence they could receive IR transmitted to the smart TV as well. The other types of systems under lighting, air quality, and home office could also be placed in the same room with the TV. Moreover, many of these devices are connected to Internet, like smart fan, smart air-purifier, smart vacuum cleaning robot and the smart air-conditioner. Take air-conditioner as an example, the top 5 vendors (Daikin, Hitachi, LG, Carrier and Midea) [23] all support Wi-Fi and IR remote control.

2.2 Smart TV

Smart TV provides internet connectivity and computing capability on top of normal TV functionalities [5]. To support third-party video content and enable other functionalities like gaming, a lot of TV manufacturers have developed TV OS and make TV a universal entertainment platform [53]. Many TV OSes have emerged. In 2020, the front-runner Samsung's Tizen has a market share of 11% worldwide, followed by LG's WebOS, Sony PlayStation, Roku TV OS, Amazon's Fire OS, Google's Android TV, Microsoft Xbox, Google's Chromecast, Apple's tvOS, etc. [58].

The common functions of smart TV can be categorized into 4 cases:

1. **Free-to-air TV broadcasting.** A user could watch a TV program by inputting the channel number (usually 2 to 3 digits number), or pressing the standard remote control key for channel up and channel down. Smart TV could process analog or digital TV signals. For digital TV signals, electronic program guide (EPG) data are also encoded, providing information like the schedule for current and upcoming broadcast programming. The channel numbers and EPG are usually released to the public (e.g., in Hong Kong [22]).

2. **Running smart TV apps.** A smart TV usually comes with a TV app store and allows users to install their favourite apps like Netflix, YouTube and Amazon Prime Video [56]. A user could navigate the store with the remote controller, login, watch streaming video, and make purchase.

3. **Configurations.** A smart TV can be configured under different video and audio settings, internet connection, account management.

4. **Supporting various input sources.** Smart TV can be configured to show the content delivered from other devices in the entertainment system. Usually, there is an "input" or "source" key in the remote controller to toggle between HDMI sources.

Previous works in the security and privacy of smart TV focused on the cyber and physical vulnerabilities at TVbox [3], broadband and broadcast systems [43], data handling of smart TV apps [42], and the tracking ecosystem [40, 60]. Our study presents a new side-channel attack against smart TV, which has not been studied.

Input methods of smart TV. There are different types of remote control unit for Smart TV. A full function smart TV remote control includes numeric keys, input source, D-pad [19], channel up/down, volume up/down, typically over 50 keys. Such a remote control will send a unique IR signal for each key pressed. With the numeric keys included in the remote control some Smart TV will also support the T9 text input method [11] with predictive text. There is also a simplified design of Smart TV remote control with much fewer keys and navigate using D-pad only. The function of press any key has to be mapped with the specific user interfaces. Oftentimes, a user needs to enter strings into the smart TV interface, like account name and credential. There are mainly 3 input methods: entry using virtual keyboard through remote controller, entry with a paired mobile app, and entry with a desktop PC/mobile browser by visiting the URL shown on TV screen. Table 2 shows 12 commonly used TV apps and their supported login method on smart TV. Virtual keyboard is supported by 8 out 12 apps, more than mobile app (7 out of 12 apps) and web browser (4 out of 12 apps). Although this is not an exhaustive list, it indicates that virtual keyboard still remains a popular input method on smart TV.

2.3 Security and Privacy Implications of IR

The leakage of the IR communication pattern can lead to severe privacy issues, and we highlight a few of them below.

Inferring users' interests and activities. What TV channels have been watched by a user can be considered as sensitive information. Previous research showed that such information can reveal a person's political orientation [63], and be leveraged to deliver targeted political ads that can influence his/her votes [20]. In fact, violation of viewers' privacy could face legal consequences: for instance, the US Video Privacy Protection Act (1988) was enacted to prevent the media providers from obtaining the fine-grained viewers' interests; US Federal Trade Commission (FTC) fined Vizio, a smart TV manufacturer, for collecting viewers' demographics and viewing

Systems	Exemplar Devices	Supported Control Technology	CI	CP	T
Entertainment	Audio systems(Bose Smart Soundbar 300)	IR remote/Mobile App/Voice Assistant	✓	✓	✓
Entertainment	Audio systems(Yamaha RX-V6A)	IR remote/Mobile App	✓	✓	✓
Entertainment	Disc Players (LG BP350 Bluray and DVD player)	IR remote/Mobile App	✓	✓	✓
Entertainment	Video streaming devices (Roku express, Roku Premiere)	IR remote/Mobile App	✓	✓	✓
Entertainment	Video streaming devices (Apple TV 4K)	IR/Bluetooth remote/Mobile App/Voice Assistant	✓	✓	✓
Entertainment	Game Consoles (Sony, XBox)	Bluetooth	✓	✓	
Lighting	Smart Bulb (Philips Hue)	Bluetooth remote/Mobile App	✓	✓	
Air Quality	Air-conditioner (Daikin ZENA)	IR remote/Mobile App	✓	✓	✓
Air Quality	Air-Purifier (Dyson Pure Cool)	IR remote/Mobile App	✓	✓	✓
Air Quality	Vacuum Cleaner Robot (iRobot Roomba 600)	IR remote/Mobile App	✓	✓	✓
Home Office	Video conferencing solution (Huawei TE30)	IR remote	✓	✓	✓

Table 1: List of IoT devices in living room. CI means Connected to Internet. CP means Connected to Power. T means Target as IR sniffer.

histories for advertising without consent. We envision the decoded IR keys can be exploited to breach viewers' privacy. For instance, by capturing the channel number (usually 3-4 digits) or the sequence of channel-up/channel-down in the IR keys from the TV remoter, and matching it to the regional free-to-air channel list or pay-TV electronic program guide (EPG), we can infer which channel is being watched.

Though the focus of this study is on smart TV, HOMESPY can be adjusted to other devices that are commanded through IR, enabling an attacker to infer what devices are installed in a home and how they are used by the victim. For example, (1) the attacker could know when the light or air-conditioner is turned on/off to infer the victim's daily routine; (2) the attacker could know the existence of a device like a robot vacuum cleaner with known vulnerabilities [50] and launch follow-up attacks.

Inferring users' account information. The traditional TV remote only supports basic functionalities like changing channels and TV configurations. But nowadays, smart TV and streaming devices support more complex functionalities like allowing a user to sign in to the TV/video service provider. By doing so, a user could access his/her personalized content seamlessly (e.g., sign in to YouTube account and see the recommended videos based on his/her viewing history), or enjoy a cross-screen viewing experience. For instance, as shown in Figure 10 of Appendix C, Android TV requires the user to sign in with a Google ID to download Google Play Store apps and sign in to a video app like YouTube. The input of account name and password is typically done by selecting characters on the on-screen virtual keyboard with the remoter. Sometimes, after stealing the TV or app account, the damage to the victim can be escalated to impact the security of the entire victim's home. For instance, the LG smart TV uses an LG smart world account which is also shared by other smart home devices in LG Smart ThinQ, like washer, dishwasher, and humidifier. Turning on/off these devices remotely are possible under the LG smart world account. Though a user could also sign in to his/her account with a paired mobile app or browser, we expect a significant number of users would

App Name	Virtual Keyboard	Mobile App	Web Service
Netflix	✓	✓	
YouTube	✓	✓	
Spotify	✓		✓
Apple TV	✓	✓	
Canal+	✓	✓	
Line TV	✓		
DAZN	✓		
Tencent Video/WeTV	✓		
BeIN Connect		✓	
Amazon Prime		✓	
HBO GO		✓	✓
Disney+			✓

Table 2: Input methods of different TV Apps. Virtual keyboard requires remote controller.

choose to sign in with the remoter because no mobile app needs to be installed ahead.

Besides the login information, users' payment information could also be at risk. Some smart TV apps accept the in-app purchase of new subscriptions of content, pay-per-view movie rental [64], or even commercial products [59]. Take Roku Pay as an example, a user can enter his/her credit card information with a remote controller and link it to the Roku account. Though extra information might be required to authorize each transaction, e.g., payment PIN (4-digit or 5-digit number) for Roku Pay and account password for YouTube [64], they are all inputted through the controller, hence also under the threat of sniffing.

3 Adversary Model and Attack Flow

The goal of our research is to assess the possibility of sniffing IR remote control signals and the information leakage associated with them. We make a few assumptions about the attack as described below.

1. **Availability of an IoT with IR receiver.** We first assume that the adversary is able to control a device that could

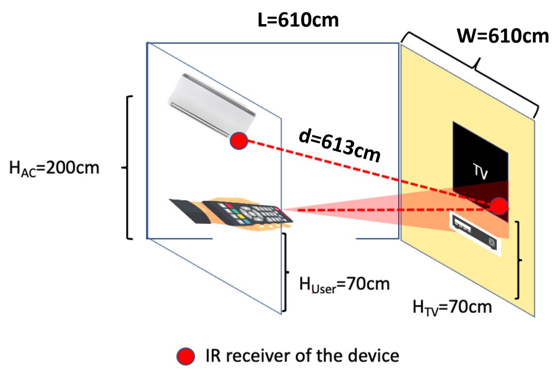


Figure 1: Living room with IR sniffer.

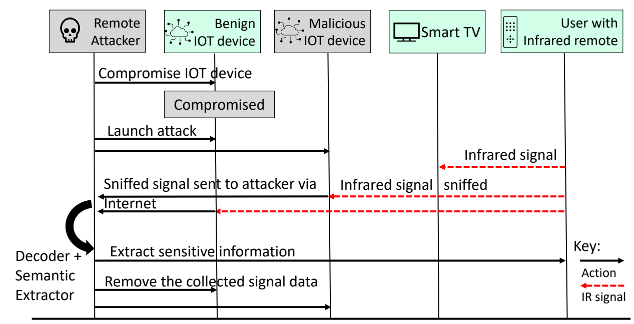


Figure 2: Attack Flow of HomeSpy.

receive IR signals and send captured data to the Internet. This condition can be satisfied by either exploiting a vulnerable IoT device or leveraging a malicious IoT device in the victim's home. We expect this condition is not difficult to satisfy in the real-world settings: (1) a large number of IoT devices come with remote controllers to receive IR signals [67] and as shown in Table 1; (2) the existence of vulnerable IoT devices is prevalent, as suggested by attacks like Mirai botnet [7] and measurement studies [29]. Different attacks could be launched with compromised IoT devices with different security impacts as we discussed in Section 7. In this paper we explore and report a new realistic attacking vector using IR signals. (3) IoT devices could be fully controlled by the attacker before being installed by the victims, through supply-chain attack (e.g., compromising IoT hardware and libraries [24]). Similar assumptions have been assumed in a prior study about Bluetooth peripherals [62].

2. **The attack device needs to be powered on.** This condition is also easy to satisfy. For example, some IoT devices are always-on after activation, like WebCam, smart speaker, etc. The attacker could also choose devices that are more likely to be powered on (e.g., air-conditioner during summer).

3. **Placement of the attack device.** Since IR is designed for short-range communication, we have assumed the attack device is placed close to the victim device that is receiving IR keys, e.g., in the same room. This condition can be satisfied reasonably: for example, as shown in Fig.1. A common living room could have a TV, a DVD/Blue-Ray player or an internet streaming device, or a smart soundbar, etc. on one side and an air-conditioner could be hanged on another wall. Once the attacker compromises one such device, HomeSpy can be launched. Though IR signal is supposed to be hard to sniff by an off-path device, our study showed that with wall reflection, sniffing is not as difficult as originally expected. There is no Line-of-Sight assumption on the placement of the attack device as we have shown in Section 5.1.

The attack flow of HomeSpy is illustrated in Fig.2, and below we describe each attack phase.

- **Gain control of an IR device.** A remote attacker gains controls of a vulnerable or malicious IoT device with the IR receiver module. Noticeably, we show that the COTS IR receiver that is common on IoT devices is sufficient for our attack.

- **Sniffing IR signals.** The IR receiver module will try to capture any IR signals in the sniffing range. Due to the small data size of IR signals, they can be sent back to the remote attacker through the internet connection and analyzed remotely.

- **IR command decoding and information extraction** The IR sniffing attack does not assume prior knowledge of the specific smart TV model or its remote control protocol of a victim. The received IR signals will be decoded first through raw IR signal timing sequences which represent the length of time which the IR LED is on or off. Since we focus on the attack against smart TV, we will match the signals to known patterns of TV models from pre-built open IR databases [25,49]. The matched timing sequence will be used to extract the commands from users, and further infer sensitive information like credentials, TV programs viewing history and users' activities at home through reconstructing the IR commands remotely. One may argue the scarcity of users entering user name and password on smart TV. In Table 2, most of the smart TV apps are using a virtual keyboard to input user name and password. It may be rare for one person to enter a user name and password for the same app all the time. However over a period of time, the login session would be time-out by the app server and users would need to re-enter the credentials and be sniffed by the attacker. Collectively speaking, the attacker could also build a large database sniffing hundreds of thousands of victims' homes.

4 Design of HomeSpy

Based on the attack workflow described previously, we develop three components for HomeSpy, our attack prototype that can break the confidentiality of the real-world IR communications. HomeSpy consists of 3 components: (1) IR sniffer which eavesdrops IR signals in the same room of victim's TV. (2) A server-side IR key decoder that infers the corresponding commands of the signal sequence. A key sequence with timestamps will be derived (including keys such as "OK", "UP", and "DOWN"), together with the inferred device types and brands; (3) A server-side semantic extractor that recovers sensitive information like the watched TV program and victim's credentials from the key sequence. We have also uploaded the database and source codes on the demo site [2].

Challenges. Though the high-level idea of HomeSpy is simple, there are a few technical challenges we need to address to achieve satisfying inference accuracy. (1) As IR is usually leveraged for Line of Sight (LOS) communication, the captured IR signals are weak when the IR sniffer is not positioned between TV and the remoter. (2) Since the consumer IR protocols are not standardized for the most part, there is no universal guidance about the mapping between IR signals and commands, which leads to a large search space for command inference. Moreover, as we discovered during the study, there exists one-to-many mappings between IR signals and commands. (3) When inferring the account information entered through the remote controller, a virtual keyboard is often utilized. However, the layouts could be different for different TV models and even TV apps. Below we describe each component under HomeSpy, and how each challenge is addressed.

4.1 IR Sniffer

There are two types of devices that can capture IR signals: one type is photo-diode or phototransistor which converts changes of infrared signals into variations of electrical current, and the other type is called *IR receiver module* which outputs *0-1 digital* signals directly based on the received IR signals after demodulation (see Figure 11 of Appendix C). We choose the latter one, i.e., IR receiver module, for the attack, because it is simple, cheap, reliable, and has been widely used in the majority of Commercial-Of-The-Shelf (COTS) IR remote control devices, including TVs, set-top boxes, as well as IoT devices. In our prototype of HomeSpy, we used Raspberry Pi to process the data from the IR receiver module instead of hacking and modifying an existing IoT device and turn it into IR sniffer. Considering the receiving sensitivity is decided by the IR receiver module, we have not made any special assumptions in the attacking model or gained any privilege over sniffing through existing devices.

The main drawback of using COTS IR receiver modules is that their sniffing sensitivity is limited, degrading the inference accuracy on the attack device. We address this issue with a novel information recovery algorithm that exploits the features at the link layer, tailored to different protocols (like NEC and Sony SIRC). Our key insight is that those protocols have a mechanism of sending redundant information to improve communication reliability, as described in Appendix A. For example, in NEC protocol, both address and data bytes will be transmitted twice, and for Sony SIRC, the same command codes will be re-transmitted every 25ms or 45ms until the user releases the key. Hence, this redundant information can be exploited to improve the success rate of IR sniffing.

Here we describe the algorithm to recover information from the Sony SIRC protocol. The high-level idea is to first split the received data sequences into groups based on the gap timing between them. This group gap timing threshold can be learned from the interval between two keypresses on the remote controller, and which is around 200ms or more. Within each group, multiple repeated sequences can be separated by a smaller timing gap (around 25ms or 45ms, depending on specific devices) which can also be learned from the sniffed sequences. Slices within each group actually are a sequence of the same command, but may lose some pulses in the case of weak signals. One key insight is that: information (pulses) lost in one slice may be presented in another slice. So it is possible to reassemble a complete key sequence by aligning and overlapping those lossy data slices together. The pseudocode is listed in Algorithm 1 of Appendix B. The recovered timing sequences will be input to the IR command decoder.

4.2 IR Command Decoder

With the timing sequences captured by IR sniffing module described in Section 4.1, the next step is to decode the commands (buttons) from the raw timings. There are numerous IR protocols in the market. Each protocol or protocol family (i.e. NEC, Sony and etc.) has its own encoding/decoding mechanism. To decode IR commands from different protocols, we build a customized IR code database by collecting IR code in different formats from the Internet and generate raw timing sequences by ourselves. The database consists of `<Timing, Command>` key-value mappings. With such a general-purpose IR code database, we can decode the commands from raw timing sequences directly. Below we firstly describe how we build the IR code database and then introduce how to look up the key for each input raw timing and construct key sequence.

Building IR Timings Database. Although we did not find a public database providing a one-stop service or API to search any IR command by IR raw timings. we noticed that there are multiple public websites containing mappings of different IR protocols, so we crawled and merged them to cover a wide range of brands and devices. Specifically, we collect IR codes from two sources: IRDB [25] and Remote Central Forums [49].

[2] https://sites.google.com/view/homespydemo

Next we generate IR raw timings from these codebases. For IRDB [25] (noted DB_{irdb}), the IR remote control codes are represented in a very space-efficient way with <protocol, device, sub-device, function>. We employ two IR raw timings generators, MakeHex [38] and irgen [26], to generate raw timings of IRDB [25]. Different from IRDB, the Device Infrared Hex Code Database (DB_{rcf}) of Remote Central forum [49] are represented in a format named Pronto-Hex [48], which consists of 4-digit hexadecimal numbers and uses a pair of numbers to represent an on/off sequence for the IR emitter. All properties of IR raw timings are embedded in the hex sequence under its own semantics. To convert the ProntoHex codes into raw timings, we develop a ProntoHex2Timing transformer leveraging an open-source program named IrScrutinizer [27]. After the previous pre-processing steps, we combine DB_{irdb} and DB_{rcf}. We show the pseudo-code of the aforementioned process in Algorithm 2. Note that though we focus on recovering the IR command or key from timing, other information about the device like device type, device category, and brand are also helpful for the semantic extractor described in Section 4.3. Thus, each record of our IR timing database consists of <Timing, (Key, Device_Type, Device_Category, Brand)>. Finally, we build a IR timings database with 75,901 IR codes of 1,303 devices.

IR Command Decoding. With the IR timings database, we can look up the key for each raw timing and construct keys sequence. For efficiency, we firstly build a hash table of <Timing, Command> pairs based on our IR timings database. Then we query each timing in the hash table to derive the output (Key, Device_Type, Device_Category, Brand). With this, the IR command decoder can differentiate Smart TV IR signal in the presence of multiple IR signals, both from other devices using a different protocol and devices with different device types of the same protocol. In the worst case, if two devices share the same IR protocol and were used at the same time, it will generate collisions that the decoder could not handle. But we believe that the chance of this would be extremely rare. The pseudo-code of IR command decoder is listed in Algorithm 2 of Appendix B.

4.3 Semantic Extractor

After the previous steps, we are able to obtain the sequence of keys issued from the remote controller. However, these commands often do not directly reveal the secret that the attacker is interested in. For example, when the victim inputs his/her credential using the direction keys (e.g., “UP” and “DOWN”) and the confirmation key (e.g., “OK”), the selected characters are not disclosed from the key sequence. Here we set a higher bar for the attacker and assumed the victim was using the D-pad instead of pressing ”1" from the full function remote control of Smart TV. As such, we build this module to extract the semantics from the key sequence, focusing on the case when the virtual keyboard is brought up on TV.

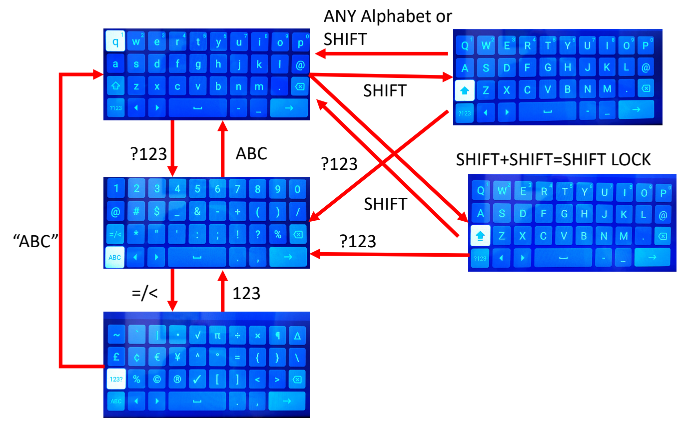


Figure 3: Layouts of android TV standard virtual keyboard.

This module takes three sub-tasks: (1) From the sequence, we identify the range of key presses that are potentially related to the virtual keyboard inputs; (2) We identify a set of candidate strings based on the known keyboard layouts and input constraints; (3) We filter out the candidates that are less likely to contain sensitive information.

Based on our observation on TV virtual keyboards (e.g., illustrated in Figure 7), when the UI is brought up, the cursor is placed always at a fixed position (e.g., upper-left) and the layout is constant. Hence, it becomes feasible to infer the input without “seeing” the keyboard. In Section 6, we discuss the defenses based on keyboard randomization.

Noticeably, there have been previous studies in the security of virtual keyboards. For instance, Diao et al. described a key injection attack against Android Input Method Editor (IME) [17]. Chen et al. [14] and Tian et al. [57] described the threat that an untrusted mobile IME could collect user’s sensitive input without consent, and proposed defenses based on sandbox and TrustZone. These works all focus on the mobile virtual keyboard, while we study the TV virtual keyboard. Below we describe each sub-task.

Identifying the range of keyboard inputs. Since the adversary is not aware when the virtual keyboard is brought up, the inferred commands could be related to browsing TV content, browsing the TV menu, or entering credentials. To differentiate these cases, we develop a method based on the frequencies of certain commands and the gaps between commands. Our key insight is that when using a virtual keyboard, a user would constantly press keys during a short time period, but the frequency is much lower when browsing the content. Moreover, the confirmation key like “OK” is pressed more often when selecting the characters on the virtual keyboard. For instance, a YouTube user usually presses the navigation keys many times before pressing the confirmation key to select the video to watch, but selecting just one character in the virtual keyboard requires pressing the confirmation key. Finally, the “BACK” key is usually used when browsing the TV menu to return to the previous level, and is rarely entered when using the virtual keyboard, so we can use its existence in the key sequence to learn if the user is browsing the TV menu.

In Figure 12 of Appendix C, we show a sub-sequence related to virtual keyboard input in a captured key sequence. Our method counts the occurrences of "OK" (termed N_{OK}) and "BACK" (termed N_{BACK}) in a time window (termed TW), and classify the sub-sequence in TW as virtual keyboard input, if N_{OK} is larger than a threshold TH_{OK} and N_{BACK} is 0. Noticeably, we do encounter false positives (i.e., keys irrelevant to virtual keyboard) in this sub-task, and we prune them in the follow-up fine-grained analysis.

Regarding the values of TH_{OK} and TW, we decided their values based on empirical analysis and previous studies. (1) According to the study about the leaked user credentials in the wild, 97% users worldwide choose no more than 15 characters for their passwords [46]. Hence, we set TH_{OK} to be larger than 15. In addition to the characters, a user sometimes presses keys like "SHIFT" and "switching keyboard page" (see Figure 3). Hence, TH_{OK} should have room for these extra keys to ensure all credential characters can be captured. (2) To determine TW, we performed a user study that involves 5 users. The detailed analysis about TH_{OK} and TW is described in Section 5.

Mapping commands to the keyboard. Next, we map the captured key sequence in TW to the character sequence on a virtual keyboard. Though there exist many TV models and apps, it turns out the number of different keyboard layouts is rather limited. In Figure 3, we show the 5 common layouts. In this work, we focus on attacking the QWERTY keyboard and number Pad used by Android TV (shown in Figure 3). In Section 6, we discuss the extension of HOMESPY to other keyboard layouts.

In particular, we build a database to map the position of the cursor on the keyboard to a character. We exploit the observation that the keyboard mapping is deterministic: for instance, the QWERTY keyboard always starts with "q" at the top left, and ends with "NEXT" at the bottom right. Also, the transition between keyboard pages (e.g., uppercase page, the lowercase page, and the symbol page) is done by pressing certain keys (the RED lines in Figure 3 show the rules of transition). With the above insights, we develop the two-step mapping method for a layout:

- We assign each keyboard character with a coordinate (x, y, z), while x and y refer to its horizontal and vertical positions, and z refers to the keyboard page index. When the virtual keyboard is brought up, the cursor is always at $(0, 0, 0)$, which maps to the letter "q". When the navigation keys are pressed, the new x and y will be computed. For example, when the cursor moves to the right and then down, the coordinate becomes $(1, 1, 0)$. If the movement and confirmation trigger a page transition, z is changed accordingly. When "OK" is pressed, the character associated with (x, y, z) will be fetched from the layout database and concatenated to the output string.
- Since we apply a coarse-grained method in the prior sub-task, many sub-sequences are discovered but not all of them are related to virtual keyboard inputs. To prune the irrelevant sub-sequences, we enumerate each "OK" within each sub-sequence and check its follow-up keys. If the cursor is moved to "Enter" and followed by an "OK", we consider the sub-sequence as a virtual keyboard candidate.

Filtering out the irrelevant candidates. For the remaining candidates, false positives still exist. We further filter them based on the syntax of different types of sensitive information that can be inputted with the remote controller. If a candidate string does not match any rule below, it will be discarded.

- A phone number string only contains digits, and its length is between 8 and 12.
- A PIN code string contains only digits and its length is between 3 and 8.
- An email string contains "." and "@", and the length of the suffix of "@" is usually larger than 3 ("com" is often appended to "@.").
- For password string, since it is entered after entering the email, we choose the next string following the identified email string.

Here, we show the pseudo-code of semantic extraction for users' account information in Algorithm 3 of Appendix B. The pseudo-code that extracts users' viewed channels is skipped due to its simplicity. We set the normal keyboard's page number as 0, the caps-lock keyboard's page number as 1, and the number keyboard's page number as 2 by default. Noticeably, multiple candidates might still be returned, but due to that their quantity is often small (e.g., at most 10 candidates during our evaluation), we did not apply additional filters. The attacker could apply app-specific rules (e.g., based on the password rule of the Netflix app) to select the best candidate.

5 Evaluation

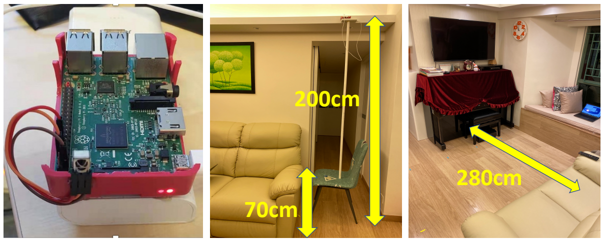
In this section, we evaluate the effectiveness of HOMESPY in a controlled environment. The evaluation results are separated by each module of HOMESPY.

Demo. The video demo of our HOMESPY attack can be found here [3].

5.1 Evaluation of IR Sniffer

IR sniffer. We evaluated the IR sniffer using a Raspberry Pi 3 with an IR sensor VS1838B as shown in Figure 4(a). The

[3] https://sites.google.com/view/homespydemo



(a) IR Sniffer on Raspberry-Pi 3 (b) Setup of IR sniffer (c) Living room – Layout C

Figure 4: IR sniffer and its setup for evaluation.

Manufafcturer	Typical IR emitter model	d_max (m)	I_e	E_e_min
Vishay	TSHA4400	12.65	40	0.25
EverLight	IR26-61C/L510	12.65	40	0.25
KingBright	WP7113SF6C-P22	20.00	100	0.25

Table 3: The maximum transmission distance of IR emitter for remote control (assuming operating at minimum of 50-100mA) and average receiver sensitivity.

IR sensor has the sensing distance of 20 meters and reception angle +/-45 degree. For attackers, it is a pretty conservative choice since VS1838B is cheaper (less than US$0.1 from some online sellers) and has shorter sensing distance when compared with other IR receiver modules (like Vishay TSOP 98240 which has a sensing distance of 24 meters). Still, we demonstrate it suffices the attack requirement.

To receive IR signal, the sniffer has to be placed within the transmission distance. Below we conduct theoretical analysis about the distance range. The transmission distance can be derived from the following equation:

$$d_{max} = \sqrt{\frac{I_e}{E_{e_{min}}}} \tag{1}$$

Where I_e is Radiant Intensity in mW/sr, $E_{e_{min}}$ is the receiver sensitivity in mW/m^2, and d_{max} is the farthest transmission distance in meters. Table 3 shows the three common IR emitter from different manufacturers for remote control applications. Assuming $E_{e_{min}}$ of the IR receiver module is 0.25mW/m^2, d_{max} would be ranged from 12.65m to 20m. And as shown in our experiment result in Table 4 (sample p17), the IR transmission distance after one reflection is at least 7.5m.

Environment setup. To understand how the efficacy of sniffing is impacted by real-world factors like distance and angle to TV, we placed the sniffer at different positions in a room with a TV. There are four living rooms of different sizes and layouts tested, as shown in Figure 5. For example, Layout A is 230cm x 200cm and the human tester is sitting on the sofa 1.8 meters away from the TV. The red marker on the figure shows the feasible location to install a wall-mounted air-conditioner or to put a tower air-conditioner. Here we assume the right-hand side of each layout floor plan is the window or the corridor and not feasible for installation. For tower air-conditioner, it would normally be put on the perimeter of the living room instead of in the middle of the room which could block the sights of users from the TV. With the above constraints, we derived the 9 feasible locations (p1 to p9) in layout A, 12 positions in layout B, 13 positions in layout C, and 18 positions in layout D. We put the sofa 3 meters from the TV in layout D because of the recommended distance viewing a 70-inches 4K TV is 3 meters. The IR sniffer has been attached to the end of a stick 2 meters high to simulate the height of a wall-mounted air-conditioner IR receiver as shown in Figure 4(b). We use Sony Smart TV KD-49X8000H and a Sony IR remote control to test HOMESPY. The TV has installed YouTube Apps to evaluate the efficacy of account inference. During each test, the tester clicks 10 consecutive keys to command the TV.

To evaluate the transmission distance in our setup, we consider a large living room in the US which is around 36.5m^2 [2] or around 6.1m x 6.1m. According to Figure 1, IR signal could be reflected by the TV wall and sniffed by the IoT device on the opposite side of the living room, the d_{max} required in a large living room would be 12.27m. From [2], the average living room size in US, UK, Australia, India and Spain are 31m^2, 17m^2, 24m^2, 30m^2 and 24m^2. The best viewing distance of any 4K resolution TV is around 1.6 times the size of the screen. For the larger model of 85-inches 4K TV, the recommended distance will be 143 inches or 3.63m. So even a user has a large living room of 36.5m^2, it is not likely they will stay 6.1m away from their 4K TV. So the d_{max} would be shorter, which makes the sniffing attack practical.

Test result. The number of the key signals are correctly extracted by our IR sniffer is measured. The result is shown in Table 4. In particular, for layout A, **98.9%** of the keys (including the repeat keys) can be correctly sniffed, even when most of the positions are not on the path of IR communication. It is also worth noticing that the larger the room, the smaller chances that an air-conditioner can act as an IR sniffer. However, even in the case of layout D (Figure 5(d)), the IR sniffer can still recover 75% of key presses, and 25% of the keys are missing. By design the IR protocol will send repeat keys and the extraction accuracy has been improved by 2.2% for layout A, 5.8% for layout B, 5.4% for layout C and 0.6% for layout D respectively.

5.2 Evaluation of IR Command Decoder

To evaluate the accuracy of the IR decoder, we focus on the 6 most frequently used keys: UP, DOWN, RIGHT, LEFT (also called *D-Pad* [19]), OK (sometimes called Select or Enter), and BACK of an IR remote controller, and we pass the sniffed timings sequence of these keys to our IR decoder for keys sequence recovery. Note that in real-world settings, more command keys like numeric digits, the POWER ON/OFF key, or VOLUME UP / DOWN keys, can also be sniffed and

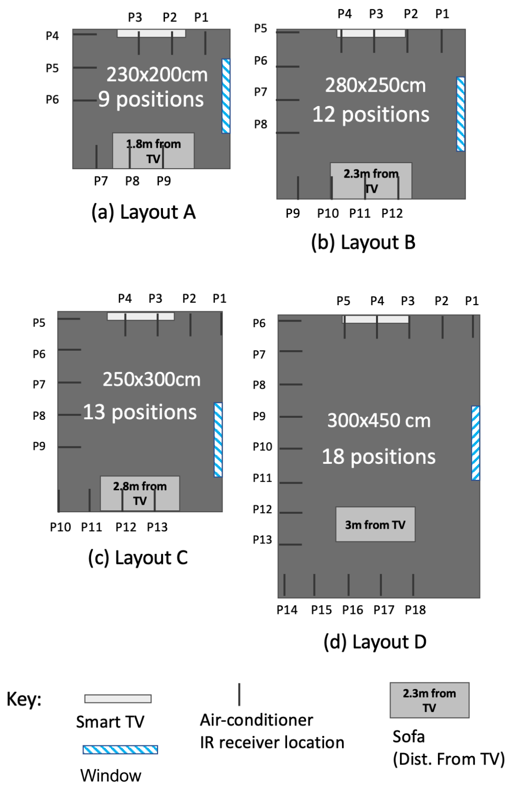


Figure 5: The layouts of the testing environment.

that would only provide more accurate information to the attacker (e.g. the number pressed or the intent of the user). Therefore we focus our evaluation on the more challenging but commonly used remote control keys of the *D-Pad*.

We have have successfully ingested 75,901 IR codes of 1,303 devices into our database. For the IR timings database constructed for evaluation purposes, since the key names may vary across different remote controllers, we unified these names into standard names (like UP, DOWN).

Among the 1303 devices, only 26 devices have overlapped D-pad[4] , OK or BACK keys, which means that for 98% of all devices have their unique mappings. Further analysis shows that 22 out of these 26 devices actually have the same set of key mappings but get labeled differently. For the remaining 4 devices, we found that their IR codes have mixed meanings, e.g., the code of LEFT in device Radix DTR-9000-Twin is the same as key DOWN in device SkyMaster 242. For such cases, the decoder will output all possible sequences.

5.3 Evaluation of Semantic Extractor

Setup of user study. To evaluate the efficacy of the semantic extractor, we performed a user study to collect the data about human input patterns with the TV. The user data is collected from 5 volunteers, who are university students and faculty

[4]The keys of D-pad include: UP, DOWN, RIGHT, LEFT, OK, and BACK.

IR Key sniffed	A	B	C	D
p1	10	6	10	10
p2	10	10	10	10
p3	10	10	10	10
p4	10	10	10	10
p5	10	10	10	10
p6	9	7	10	10
p7	10	10	8	10
p8	10	7	0	10
p9	10	10	0	10
p10	-	10	10	0
p11	-	9	10	0
p12	-	10	10	0
p13	-	-	10	0
p14	-	-	-	5
p15	-	-	-	10
p16	-	-	-	10
p17	-	-	-	10
p18	-	-	-	10
Extraction Accuracy	98.9%	90.8%	83.1%	75.0%
Missed	1.1%	9.2%	16.9%	25.0%

Table 4: The number of correctly sniffed IR keys in layout A, B, C and D, and the accuracy of IR sniffer (the ratio between sniffed keys and pressed keys).

members aged 20-50. We have got the IRB approval and the protocol and consent form is put in the appendix of this paper.

Each participant is asked to complete the following tasks:

- **Task 1**: entering 10 email-based login credentials (i.e., email addresses and passwords) and 2 phone-number-based login credentials (i.e., phone numbers and passwords), using a virtual QWERTY keyboard as shown in Figure 7-i;
- **Task 2**: entering 50 PIN code with 4-digits, using the number pad layout as shown in Figure 7-v;
- **Task 3**: navigating on the YouTube app for 10 minutes, with an IR remote controller.

The credentials such as emails and passwords are selected from an open dataset of leaked user login credentials [30, 31]. For phone number and PIN code, we randomly generated the numbers with 8 digits and 4 digits respectively. To ensure consistency across our data collection processes, all participants use the same remote controller and android streaming box (BOSSv2 TV) connected to the tested TV for experiments. The participants were also instructed to use only the D-Pad, OK and BACK keys on the remote control in all the tasks.

Overall, we collected around 10,000 keypresses on the remote control. We classify the keypresses about entering login credentials and PIN code as positive and navigation as negative. For task 1, one account (with an email or phone number and a password) is considered as 1 positive sample. For task 2, one PIN code is a positive sample. For task 3, each 10-mins sequence of keys is regarded as a negative sample. In the end, we collect a total number of 315 samples including 60 positive samples from task 1, and 250 positive samples

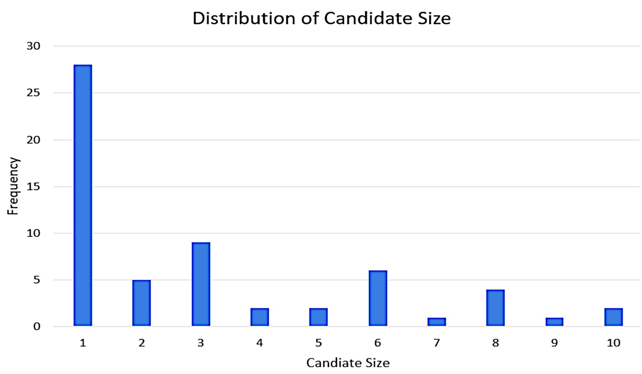


Figure 6: Distribution of Candidate String Sizes.

from task 2, and 5 negative samples from task 3. The scale of our experiment is comparative to prior works that try to infer the typed information from victim users (e.g., 800 samples of 8 users in [33]) or other side-channel attack study using voice profiles of 6 users in [39].

Overall results. For task 1 and task 2, among 310 positive samples that we collect, HOMESPY is able to identify the sequence of the keyboard input from *all of them*. No false negatives of the characters have been discovered in any sequence, but we found that 53% out of the 310 samples have at least one false positive string (considering all strings outputted by semantic extractor), and the main reason is that HOMESPY could recover more than one candidate string. Table 6 shows an example candidate list of recovered strings and Figure 6 shows the distribution of candidate string sizes. The average size of the candidate list is 2.95 characters. A login credential (username and password, or phone number and password) or a PIN is correctly inferred if it appears in the candidate list whose size is less than 5.

As shown in Table 5, the overall success rate of our semantic extraction is 47% if only one candidate is allowed (Top1). The accuracy increases to 70% for Top3 and 77% for Top5. Given that it is common for a website and app to allow multiple login attempts (usually 5 times), the inference accuracy is satisfactory.

For task 3, if the semantic extractor mistakenly produces a string about login credential or PIN code, the result is considered a false positive. Because the activation and deactivation of virtual keyboard follow a fixed sequence of commands (e.g., after OK), it is unlikely to produce false positives. In fact, none of the user's navigation activities are classified as secret (0 false positives). There is one string that looks similar as PIN code ("6.76"), and it is filtered out because the PIN code only has digits.

Threshold values. As described in Section 4.3, we use two thresholds to capture the range of virtual keyboard inputting, namely TW (time window) and TH_{OK} (the minimum number of "OK" being pressed). Below we describe how their values are decided.

User #	Top1	Top3	Top5
U1	17%	50%	58%
U2	75%	75%	75%
U3	58%	83%	100%
U4	33%	58%	67%
U5	50%	83%	83%
Mean	**47%**	**70%**	**77%**

Table 5: Accuracy of semantic extraction on the collected samples.

Table 6: Example of Recovered Strings

	Account	Password
`Target1`	TD052288@gmail.com	emerson10
`Candidate1`	qtd052288@GMAIL.COM	emerson10
`Candidate2`	td052288@GMAIL.COM	emerson10
`Target2`	XIAOLIN123@gmail.com	143ARIAN
`Candidate1`	qxiaolin123@GMAIL.COM	143ARIAN
`Candidate2`	xiaolin123@GMAIL.COM	143ARIAN

For TW, based on the data collected from the volunteers, on average, it takes 114.15 seconds to input both the email/phone number and the password. For TH_{OK}, based on the collected data, the average number of "OK" keys inputted is 34.95 to input a credential.

In the end, to accommodate the variance of users' input behaviors, we set TW and TH_{OK} to 300 (seconds) and 150, respectively. We set such high values in case of missing any events/behaviors. The experiment result shows HOMESPY could cover 100% of the activities about keyboard input.

6 Discussion and Limitations

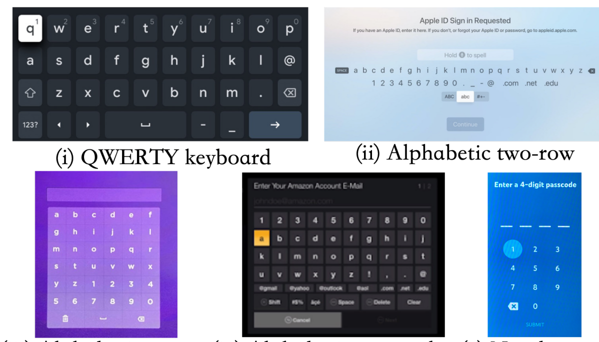
Securing IR communications. The current IR transmission mechanism does not include any security measures like encryption, as it assumes line-of-sight usage and only insensitive information is transferred. However, these assumptions are challenged based on the new use cases of IR, and our attack method HOMESPY. To enforce security measures on IR communication, Kim et al. [28] mentioned that encryption should be adopted to protect the commands from eavesdropping in such scenarios. As we know, the core of encryption is to establish a shared secret. However, in their scheme, a key (the shared secret) is generated on the remote controller and directly transmitted through the IR channel. By our assumption, this scheme is insecure as HOMESPY can capture the transmitted key as well.

Considering the trade-off between usability and security, we argue that fundamentally any security mechanisms are difficult to deploy on IR, especially in the scenario of remote control of smart TVs. For other protocols like Bluetooth [44], before secure communication is established, namely enforcing encryption schemes, the two communication parties need to exchange some messages, to negotiate a share secret (e.g., using Diffie–Hellman key exchange schemes). However, for the connection between a smart TV and a remote controller, the communication is only one way: only the remote controller can give instructions to the smart TV. If we wish to build a two-way communication, any messages from the TV to the remote controller must be delivered through a user (or using extra hardware). If the message delivered through users is short, there is no guarantee for its security with brute-forcing attacks. Otherwise, if the message delivered by users is too long, it would cause inconvenience to users.

Interference to IR communications. Indoor IR communications may be subject to interference of other light sources, including fluorescent lamps, TV displays and sunlight. The impact of which has been mitigated by IR receiver control circuits that provide feedback of the received signal to the automatic gain control (AGC) circuitry and the solution is available in the market from different vendors. The AGC helps ensure high S/N ratio and make our attack practical under interference. We could study the impact from other ambient light sources in the future. For direct interference closer to carrier frequency of IR (e.g. 38kHz), there was reported interference from fluorescent lamps operating on high frequency electronic ballasts. The problem has been resolved by choosing the operating frequency of electronic ballasts over 40kHz [1].

Other means of in-home control. IR has been the oldest and most deployed wireless control technology in the home. Although other technologies such as Bluetooth and RF are used for in-home appliance control, IR remains the major and cost-effective wireless control technology in the home. RF remote control can be used without a line of sight and supports bi-directional communications. There are also Bluetooth remote controls using Bluetooth Low Energy (BLE) in the market, especially for higher-end TV models. Though RF and BLE provide better security by design (e.g., with pairing and traffic encryption) and enable communication, not in the line of sight, they are vulnerable to sniffing and MITM attacks due to unavoidable bugs in the implementation [9, 34]. RF and BLE can transfer more information with a smart TV for additional functions like voice search or channel logo display. In some sense, RF and BLE are overkill for controlling home appliances remotely.

Many smart home devices also support remote controller apps on smartphones, we see that IR would continue to play a major role in remote control for years to come given its low cost and simplicity. Smart home devices and smartphones can be connected through an IP network or Bluetooth directly. Using a smartphone to control smart home devices incurs no additional hardware cost but it is not a common behaviour among general users based on the following observations. First of all, consumer demands a standalone controller unit that can be used any time with or without a smartphone. Hence remote controller unit is usually provided with the smart home device. Secondly, if one wants to turn on a smart TV after setup, he or she would need to unlock the smartphone, find the remote control app, launch the remote control app, and then click an on-screen button. Compare this to one click on the IR remote control unit, the worse usability undermines the adoption of using a smartphone to replace the remote control. Perhaps a better alternative is to use smart speakers and voice commands to turn on a smart TV. However, voice commands could only support a finite set of simple commands. In cases where users need to enter user credentials on a smart TV, a remote control unit would still be preferred (not to mention that one would not likely shout out his PIN or password). As a result, HOMESPY continues to pose a valid threat to smart homes.



(i) QWERTY keyboard (ii) Alphabetic two-row (iii) Alphabetic square (iv) Alphabetic rectangle (v) Number pad

Figure 7: 5 common on-screen keyboard layouts.

Different layouts of virtual keyboards. In this work, we focus on the QWERTY virtual keyboard used by Android TV and the Number pad. Based on our survey, there are 5 common types shown in Figure 7. Lui et al. identified three common smart TV onscreen keyboards layout in TV apps [35], namely QWERTY (used by Samsung TV, LG TV, Google's Android TV, Xbox), Alphabetic square (used by Roku), and Alphabetic two-row (used by Apple TV). We found that the Amazon Fire TV uses a slightly different layout from the Alphabetic square and we call it an Alphabetic rectangle. Finally, there is a simplified Number Pad layout for number-only entry. We plan to extend HOMESPY to other layouts in the next step. In particular, based on the observation that the frequencies of hitting UP/DOWN/LEFT/RIGHT on the QWERTY keyboard, Alphabetic two-row keyboard, and Number pad are different to select a character, they can be distinguished readily. However, since the alphabetic square and rectangle keyboard have similar layouts, finer-grained analysis is needed. We can also distinguish the use of virtual keyboard from T9 input method. When multiple numeric keys are pressed, text input could be inferred using the sequence of numeric keys sniffed and the T9 predictive text algorithm. The privacy attack on T9 and the predictive text learning system has been studied by Wilkinson et al. [61].

Activation and filter of virtual keyboard input. We assume the activation of virtual keyboard follows a command sequence that is known to the adversary, e.g., after the "OK" key. We found this assumption hold for the common cases like entering a PIN code or password to confirm the purchase, and inputting login credential. However, the sequence can be changed when the user needs to select some menu options between the "OK" key and keyboard activation, which could confuse the semantics extractor. To address this issue, the semantics extractor could consider *every* key after "OK" to be the candidate start of the virtual keyboard (i.e., "q"). And we use ENTER button to separate the username and password. We could also evaluate other algorithms to filter keyboard input instead of using a fixed threshold as future work.

Randomization of keyboard layout. Layout randomization has been proposed to make keyboard logging more difficult, but it also brings in extra overhead for normal user to input information. Pak et al. [45] proposed a randomized QWERTY keyboard, which however increased around 30% of input time. Another approach is to randomly insert "white space" in each row [51]. Yet, keyboard randomization has not been widely deployed in any of the major smart TV platforms, and we expect the semantic extractor of HOMESPY will not be impacted in the near future.

7 Related Work

Our work demonstrated the commands embedded within IR signals can be inferred and leak sensitive information in the smart home. In this section, we review previous studies related to smart home and IR.

Security and privacy issues in smart home. Home-based IoT devices have become the major target for cyber-attackers in the recent decade, due to numerous vulnerabilities they embed and their close interaction with users' daily lives. Alrwai et al. surveyed the existing literature in IoT security and privacy and categorized the attack surface by device, mobile apps, cloud endpoint, and communications [6]. The issues we identified relate to communications.

Researchers have explored various approaches to infer private information from the communications among different entities in a smart home [4, 8, 13, 15, 18, 21, 52, 55]. For instance, Dong et al. [18] presented a neural network-based approach to identify the types of active devices in smart home by analyzing the network traffic. Acar et al. [4] identified the types of IoT devices, their states, and ongoing user activities, by passively sniffing wireless network traffic like Wi-Fi, ZigBee, and BLE, in a smart home. Srinivasan et al. [55] showed that in-home private activities such as cooking, showering, toileting, and sleeping can be observed by eavesdropping on the wireless transmissions of sensors in a home, even when the transmission is encrypted. Schwittmann et al. [52] leveraged ambient light sensors on a mobile phone to infer the video watched on the TV screen. Those works rely on side-channels from light, Ethernet, or wireless communication, to infer sensitive information, while we look into IR, whose security implications in smart home have not been well understood.

Regarding the issues unique to smart TV, Section 2.2 has described the related works.

IR-related security issues. A few previous works have shown IR can be used as a covert channel to send sensitive information out from air-gapped machines. Maiti et al. [37] exploited the new multimedia visualization methods of smart light to infer the content watched by a victim, and used IR as a covert channel to send the inferred content out. Zhou et al. [67] leveraged a malicious IR hardware module (MIRM) embedded in a USB keyboard to collude with malware in the victim's PC. Receiving the sensitive information stolen by the malware, MIRM can transmit it to a nearby IoT device supporting IR. The IoT device will relay the information to the remote attacker. Besides covert channel, IR has also been leveraged to break the integrity of the computing systems.

For instance, Zhou et al. [66] demonstrated that the out of face recognition systems can be misled when part of a human's face is illuminated by IR, which cannot be perceived by a human observer. Our work explores different directions from the above works, as we focus on the privacy of IR itself, rather than leveraging IR as a medium to attack the confidentiality or integrity of other systems.

Ling et al. [32] built a new transceiver to bridge the IR communication channel with the Internet, and studied the possible vulnerabilities of that transceiver at the protocol level. Our work is different and focused on a new attack vector at the IR communication channel itself. A related study by Kim et al. [28] runs some sensitivity tests of IR signals from different angles, distances, and even through a curtain. However, our work has significant improvements because our attacks are more practical by being able to handle IR signal reflections and steal real world private information.

8 Conclusion

Our research re-examines the general belief that clicking an IR remote control is a homely and safe thing to do. We have developed a HOMESPY attack and evaluate its performance using test data of received signal and our offline IR code database. Our studies show that IR signals at home can be easily sniffed by an IoT device sitting in the same room, and attackers can uniquely reproduce the key pressed by the victim and derive sensitive information via semantic extraction techniques. In the era of IoT, many smart devices are connected to the Internet and support IR for compatibility with universal IR controllers. This means the invisible IR vulnerability presented in this paper will continue to cause significant threats to smart home security.

Acknowledgement We want to thank our anonymous reviewers for their valuable comments. This work was supported in part by National Key Research & Development Project of China (Grant No. 2019YFB1804400), and Hong Kong S.A.R. Research Grants Council (RGC) General Research Fund No. 14209720. Jiongyi Chen was supported by the Natural Science Foundation of Hunan Province, China (Grant No. 2022JJ40553).

References

[1] (2003) Interference problems of fluorescent lamps operating on high frequency electronic ballasts with infrared remote control equipment and infrared simultaneous interpretation system. [Online]. Available: https://www.emsd.gov.hk/filemanager/en/content_764/infrared_interference_emsdweb.pdf

[2] (2021) Average size of living room. [Online]. Available: https://www.homenish.com/average-size-living-room/

[3] Y. Aafer, W. You, Y. Sun, Y. Shi, X. Zhang, and H. Yin, "Android smarttvs vulnerability discovery via log-guided fuzzing," in *30th {USENIX} Security Symposium ({USENIX} Security 21)*, 2021.

[4] A. Acar, H. Fereidooni, T. Abera, A. K. Sikder, M. Miettinen, H. Aksu, M. Conti, A.-R. Sadeghi, and S. Uluagac, "Peek-a-boo: I see your smart home activities, even encrypted!" in *Proceedings of the 13th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, 2020, pp. 207–218.

[5] I. Alam, S. Khusro, and M. Naeem, "A review of smart TV: Past, present, and future," in *2017 International Conference on Open Source Systems & Technologies (ICOSST)*. IEEE, Dec. 2017. [Online]. Available: https://doi.org/10.1109/icosst.2017.8279002

[6] O. Alrawi, C. Lever, M. Antonakakis, and F. Monrose, "Sok: Security evaluation of home-based iot deployments," in *2019 IEEE symposium on security and privacy (sp)*. IEEE, 2019, pp. 1362–1380.

[7] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, M. Kallitsis, D. Kumar, C. Lever, Z. Ma, J. Mason, D. Menscher, C. Seaman, N. Sullivan, K. Thomas, and Y. Zhou, "Understanding the mirai botnet," in *Proceedings of the 26th USENIX Conference on Security Syposium*, ser. SEC'17. USA: USENIX Association, 2017, p. 1093–1110.

[8] N. Apthorpe, D. Reisman, S. Sundaresan, A. Narayanan, and N. Feamster, "Spying on the smart home: Privacy attacks and defenses on encrypted iot traffic," *arXiv preprint arXiv:1708.05044*, 2017.

[9] A. Barua, M. A. A. Alamin, M. S. Hossain, and E. Hossain, "Security and privacy threats for bluetooth low energy in IoT and wearable devices: A comprehensive survey," *IEEE Open Journal of the Communications Society*, vol. 3, pp. 251–281, 2022. [Online]. Available: https://doi.org/10.1109/ojcoms.2022.3149732

[10] C. A. Benson-Allott, *Remote control*, ser. Object lessons. Bloomsbury Academic, an imprint of Bloomsbury Publishing Inc., in association with Loyola University New Orleans, The Atlantic, Georgia Tech Center for Media Studies, 2015.

[11] M. Bohan, A. Chaparro, and C. G. Halcomb, "A psychophysical comparison of two stylus-driven soft keyboards," in *Proceedings of Graphics Interface '99*, 1999, pp. 92–97.

[12] C. Brook, "Travel Routers, NAS Devices Among Easily Hacked IoT Devices," https://threatpost.com/travel-routers-nas-devices-among-easily-hacked-iot-devices/124877/, Accessed: November 2017.

[13] J. Bugeja, A. Jacobsson, and P. Davidsson, "On privacy and security challenges in smart connected homes," in *2016 European Intelligence and Security Informatics Conference (EISIC)*. IEEE, 2016, pp. 172–175.

[14] J. Chen, H. Chen, E. Bauman, Z. Lin, B. Zang, and H. Guan, "You shouldn't collect my secrets: Thwarting sensitive keystroke leakage in mobile {IME} apps," in *24th {USENIX} Security Symposium ({USENIX} Security 15)*, 2015, pp. 657–690.

[15] J. Chen, W. Diao, Q. Zhao, C. Zuo, Z. Lin, X. Wang, W. C. Lau, M. Sun, R. Yang, and K. Zhang, "Iotfuzzer: Discovering memory corruptions in iot through app-based fuzzing." in *NDSS*, 2018.

[16] L. Constantin, "Hackers found 47 new vulnerabilities in 23 IoT devices at DEF CON," http://www.csoonline.com/article/3119765/security/hackers-found-47-new-vulnerabilities-in-23-iot-devices-at-def-con.html, Accessed: November 2017.

[17] W. Diao, X. Liu, Z. Zhou, K. Zhang, and Z. Li, "Mind-reading: Privacy attacks exploiting cross-app keyevent injections," in *European Symposium on Research in Computer Security*. Springer, 2015, pp. 20–39.

[18] S. Dong, Z. Li, D. Tang, J. Chen, M. Sun, and K. Zhang, "Your smart home can't keep a secret: Towards automated fingerprinting of iot traffic," in *Proceedings of the 15th ACM Asia Conference on Computer and Communications Security*, 2020, pp. 47–59.

[19] (2021) Android developer's guide: Tv navigation. [Online]. Available: https://developer.android.com/training/tv/start/navigation

[20] (2018) Facebook and data privacy in the age of cambridge analytica. [Online]. Available: https://jsis.washington.edu/news/facebook-data-privacy-age-cambridge-analytica/

[21] K. Fawaz, K.-H. Kim, and K. G. Shin, "Protecting privacy of {BLE} device users," in *25th {USENIX} Security Symposium ({USENIX} Security 16)*, 2016, pp. 1205–1221.

[22] (2021) Digital tv broadcast in hong kong. [Online]. Available: https://www.digitaltv.gov.hk/eng/

[23] (2020) Global hvac (heating, ventilation, air conditioning) industry factsheet 2020: Top 10 largest hvac companies in the world. [Online]. Available: https://blog.bizvibe.com/blog/uncategorized/top-10-largest-hvac-companies

[24] (2020) Risks in iot supply chain. [Online]. Available: https://unit42.paloaltonetworks.com/iot-supply-chain/

[25] (2021) irdb. [Online]. Available: https://github.com/probonopd/irdb

[26] (2017) irgen. [Online]. Available: https://github.com/elupus/irgen

[27] (2021) Irscruitinizer. [Online]. Available: https://github.com/bengtmartensson/IrScrutinizer

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

[29] D. Kumar, K. Shen, B. Case, D. Garg, G. Alperovich, D. Kuznetsov, R. Gupta, and Z. Durumeric, "All things considered: an analysis of iot devices on home networks," in *28th {USENIX} Security Symposium ({USENIX} Security 19)*, 2019, pp. 1169–1185.

[30] (2015) password-lists. [Online]. Available: https://github.com/lavalamp-/password-lists

[31] (2015) username-lists. [Online]. Available: https://github.com/nyxgeek/username-lists

[32] Z. Ling, C. Gao, C. Sano, C. Toe, Z. Li, and X. Fu, "STIR: A smart and trustworthy IoT system interconnecting legacy IR devices," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 3958–3967, May 2020. [Online]. Available: https://doi.org/10.1109/jiot.2019.2963767

[33] X. Liu, Z. Zhou, W. Diao, Z. Li, and K. Zhang, "When good becomes evil: Keystroke inference with smartwatch," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, 2015, pp. 1273–1285.

[34] A. Lonzetta, P. Cope, J. Campbell, B. Mohd, and T. Hayajneh, "Security vulnerabilities in bluetooth technology as used in IoT," *Journal of Sensor and Actuator Networks*, vol. 7, no. 3, p. 28, Jul. 2018. [Online]. Available: https://doi.org/10.3390/jsan7030028

[35] S. Luo and Y. Hu, "Dual-cursor: Improving user performance and perceived usability for cursor-based text entry on tv using remote control," *Interact. Comput.*, vol. 31, pp. 263–281, 2019.

[36] J. Lyne, "Uncovering IoT Vulnerabilities in a CCTV Camera," https://www.rsaconference.com/videos/demo-uncovering-iot-vulnerabilities-in-a-cctv-camera, Accessed: November 2017.

[37] A. Maiti and M. Jadliwala, "Light ears: Information leakage via smart lights," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 3, no. 3, pp. 1–27, Sep. 2019. [Online]. Available: https://doi.org/10.1145/3351256

[38] (2017) Makehex. [Online]. Available: https://github.com/probonopd/MakeHex

[39] Y. Meng, H. Zhu, J. Li, J. Li, and Y. Liu, "Liveness detection for voice user interface via wireless signals in IoT environment," *IEEE Transactions on Dependable and Secure Computing*, pp. 1–1, 2020. [Online]. Available: https://doi.org/10.1109/tdsc.2020.2973620

[40] 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.

[41] (2020) Nec protocol. [Online]. Available: https://www.sbprojects.net/knowledge/ir/nec.php

[42] 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.

[43] Y. Oren and A. D. Keromytis, "From the aether to the ethernet—attacking the internet using broadcast digital television," in *23rd {USENIX} Security Symposium ({USENIX} Security 14)*, 2014, pp. 353–368.

[44] J. Padgette, K. Scarfone, and L. Chen, "Guide to bluetooth security," *NIST Special Publication*, vol. 800, p. 121, 2017.

[45] W. Pak, Y. Cha, and S. Yeo, "High accessible virtual keyboards for preventing key-logging," in *2016 Eighth International Conference on Ubiquitous and Future Networks (ICUFN)*. IEEE, Jul. 2016. [Online]. Available: https://doi.org/10.1109/icufn.2016.7537017

[46] (2017) Average number of characters of leaked user passwords worldwide as of 2017. [Online]. Available: https://www.statista.com/statistics/744216/worldwide-distribution-of-password-length/

[47] O. R. Popoola, W. O. Popoola, R. Ramirez-Iniguez, and S. Sinanovic, "Design of improved IR protocol for LED indoor positioning system," in *2017 13th International Wireless Communications and Mobile Computing Conference (IWCMC)*. IEEE, Jun. 2017. [Online]. Available: https://doi.org/10.1109/iwcmc.2017.7986402

[48] (2015) Prontohex. [Online]. Available: https://github.com/probonopd/ProntoHexx

[49] (2021) Remotecentral. [Online]. Available: https://github.com/probonopd/irdb

[50] S. Sami, Y. Dai, S. R. X. Tan, N. Roy, and J. Han, "Spying with your robot vacuum cleaner: eavesdropping via lidar sensors," in *Proceedings of the 18th Conference on Embedded Networked Sensor Systems*, 2020, pp. 354–367.

[51] (2015) Samsung pay on tv 2. [Online]. Available: https://techweez.com/2015/08/05/samsung-makes-it-possible-to-pay-for-tv-content-using-samsung-pay/samsung-pay-on-tv-2/

[52] L. Schwittmann, V. Matkovic, M. Wander, and T. Weis, "Video recognition using ambient light sensors," in *2016 IEEE International Conference on Pervasive Computing and Communications (PerCom)*. IEEE, Mar. 2016. [Online]. Available: https://doi.org/10.1109/percom.2016.7456511

[53] P. Seungho, L. Yoojin, and K. Tomimatsu, "Design proposal for smart tv interface and remote controller," *International Journal of Asia Digital Art and Design Association*, vol. 22, no. 1, pp. 1–13, 2018.

[54] (2020) Sony sirc protocol. [Online]. Available: https://www.sbprojects.net/knowledge/ir/sirc.php

[55] V. Srinivasan, J. Stankovic, and K. Whitehouse, "Protecting your daily in-home activity information from a wireless snooping attack," in *Proceedings of the 10th international conference on Ubiquitous computing*, 2008, pp. 202–211.

[56] (2017) Top 25 tv apps. [Online]. Available: https://www.tcl.com/us/en/top-tv-apps

[57] C. Tian, Y. Wang, P. Liu, Q. Zhou, and C. Zhang, "Using im-visor to stop untrusted ime apps from stealing sensitive keystrokes," *Cybersecurity*, vol. 1, no. 1, pp. 1–17, 2018.

[58] (2020) Connected tv streaming devices in use market share worldwide as of 1st quarter 2020, by platform. [Online]. Available: https://www.statista.com/statistics/1171132/global-connected-tv-devices-streaming-market-share-by-platform/

[59] (2020) Qvc, hsn lead launch of shop time livestream marketplace on lg smart tv. [Online]. Available: https://www.qurateretailgroup.com/newsroom/happenings/qvc-hsn-lead-launch-of-shop-time-livestream-marketplace-on-lg-smart-tv/

[60] J. Varmarken, H. Le, A. Shuba, A. Markopoulou, and Z. Shafiq, "The tv is smart and full of trackers: Measuring smart tv advertising and tracking," *Proceedings on Privacy Enhancing Technologies*, vol. 2020, no. 2, 2020.

[61] G. Wilkinson and P. Legg, ""what did you say?": Extracting unintentional secrets from predictive text learning systems," in *2020 International Conference on Cyber Security and Protection of Digital Services (Cyber Security)*. IEEE, Jun. 2020. [Online]. Available: https://doi.org/10.1109/cybersecurity49315.2020.9138882

[62] F. Xu, W. Diao, Z. Li, J. Chen, and K. Zhang, "Badbluetooth: Breaking android security mechanisms via malicious bluetooth peripherals." in *NDSS*, 2019.

[63] 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, Oct. 2015. [Online]. Available: https://doi.org/10.1111/pops.12307

[64] (2021) How to purchase movies & tv shows on smart tvs. [Online]. Available: https://support.google.com/youtube/answer/9676953?hl=en

[65] N. Zhang, S. Demetriou, X. Mi, W. Diao, K. Yuan, P. Zong, F. Qian, X. Wang, K. Chen, Y. Tian, C. A. Gunter, K. Zhang, P. Tague, and Y. Lin, "Understanding IoT Security Through the Data Crystal Ball: Where We Are Now and Where We Are Going to Be," *CoRR*, vol. abs/1703.09809, 2017.

[66] Z. Zhou, D. Tang, X. Wang, W. Han, X. Liu, and K. Zhang, "Invisible mask: Practical attacks on face recognition with infrared," *arXiv preprint arXiv:1803.04683*, 2018.

[67] 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, Jan. 2019. [Online]. Available: https://doi.org/10.1016/j.comnet.2018.11.014

Appendices

A IR Signal and Encoding

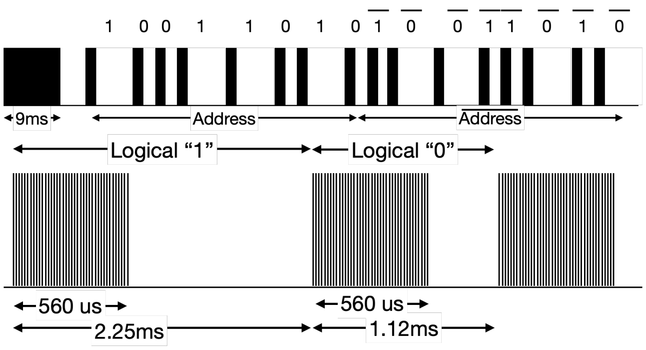


Figure 8: The upper figure shows the NEC protocol at carrier frequency of 38kHz while the lower one shows the modulation.

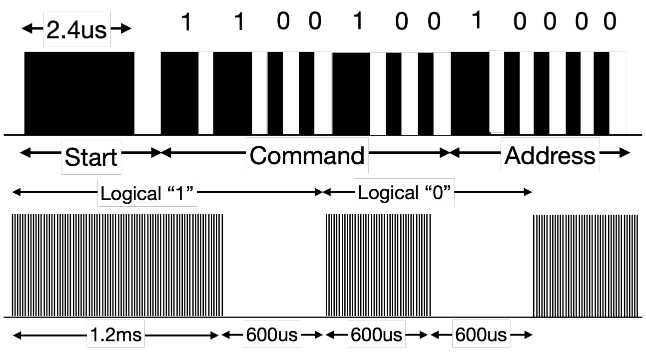


Figure 9: The upper figure shows the Sony SIRC protocol at carrier frequency of 40kHz while the lower one shows the corresponding modulation.

To control a consumer device from a remote controller, the IR light is typically generated by an IR LED. The carrier frequency is different for different CIR protocol: e.g., NEC uses 38 kHz, Sony uses 40 kHz and RC-5 uses 36 kHz. As shown in Fig.8 and Fig.9, we illustrate the NEC and Sony SIRC modulation and protocol respectively, showing their designs are quite different.

In particular, the NEC code uses pulse distance modulation. Logical 0 is a 562.5µs pulse burst followed by a 562.5µs space, with a total transmit time of 1.125ms. Logical 1 is a 562.5µs pulse burst followed by a 1.6875ms space, with a total transmit time of 2.25ms. The code always starts with a burst of 9ms, followed by a pause of 4.5ms, before any data word. 8 address bits are used to identify the device to be controlled, and then 8 bits are used as the command data. Both the address bytes and the data bytes are transmitted twice, first as a normal byte and then followed by an inverted byte.

For the case of Sony SIRC, there are 3 versions and the one shown in Fig.9 is a 12-bit SIRC protocol. The SIRC protocol uses pulse width encoding of the bits. A logical "1" is a 1.2ms long burst of the 40kHz carrier, while the burst width for a logical "0" is 0.6ms long. The start burst is always 2.4ms wide, followed by a standard space of 0.6ms. Apart from signaling the start of an SIRC message this start burst is also used to adjust the gain of the IR receiver. Then the 7-bit Command is transmitted, followed by the 5-bit Device address. Commands are repeated every 45ms (measured from start to start) for as long as the key on the remote control is held down.

B Pseudo-codes of HomeSpy

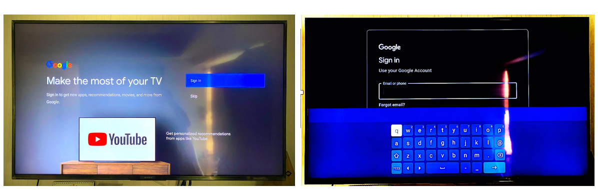
Algorithm 1: Information Recovery Algorithm for Sony SIRC Protocol.

```
input  : RawSequence, GroupGapThreshold, RepeatGapThreshold
output : RecoveredSequence
groupList= []; newGroup = []; newSlice = [];
for i=0 ... RawSequence.length do
    input = RawSequence[i]
    if input is SPACE AND input.Value > GroupGapThreshold
     then
        groupList.append(newGroup)
        newGroup = []
        newGlice = []
    else if input is SPACE AND input.Value > RepeatGapThreshold
     then
        newGroup.append(newSlice)
        newSlice = []
    else
        newSlice.append(input)
end
RecoveredSequence = []
for group in groupList do
    temp_slice = []
    for slice in group do
        t = findStartSymbolPosition(slice)
        if temp_slice == [] then
            temp_slice = align(slice, t)

        else
            temp_slice = combine(temp_slice, align(slice, t))
    end
    RecoveredSequence.append(temp_slice)
end
return RecoveredSequence
```

Algorithm 3: Keyboard Information Extraction

```
input  :Sequence, Keyboard([page_number, y_position, x_postion])
output :Extracted private information candidates
candidates = []
// Establish the Keyboard position mapping
for p = 1...Keyboard.page_number do
    for y = 1...Keyboard[p].y_number do
        for x = 1...Keyboard[p,y].x_number do
            mapping[p-1,y-1,x-1] = Keyboard[p,y,x]
        end
    end
end
// Check each potential sequence and generate
   candidates
for i = 0 ... Sequence.length do
    position ← [0,0,0]
    data = Sequence[i:]
    message = ''
    for c in data do
        if c == 'BACK' then
            break
        else if c in ['UP','DOWN','LEFT','RIGHT'] then
            position = Change_position(position, c)
        else if c == 'OK' then
            if mapping[position] == 'ENTER' then
                candidates.append(message)
                message = ''
                position ← [0,0,0]
            else if mapping[position] == 'SHIFT' then
                if position.page == 1 then
                    position.page = 0
                else if position.page = 0 then
                    position.page = 1
            else if mapping[position] == 'NUMBER' then
                if position.page == 2 then
                    position.page = 0
                else if position.page = 0 then
                    position.page = 2
            else
                message += mapping[position]
    end
    candidates = Filter(candidates)
    return candidates
end
```



(a) YouTube App on Smart TV (b) Login screen of Google Account

Figure 10: YouTube app and login.

Algorithm 2: IR Command Decoder

```
Input  :DB_irdb, DB_rcf, timingsSequence
Output :DB_timings, keysSequence
//Building IR Timings Database
Function DB_build(DB_irdb, Db_rcf):
    DB_timings = {};
    for code_irdb in DB_irdb do
        //code_irdb = <protocol, device_type,
         device_category, key>
        if code.protocol ∈ MakeHex.protocols then
            timing = MakeHex(code_irdb);
        else if code.protocol ∈ irgen.protocols then
            timing = irgen(code_irdb);
        else
            continue;
        //command = (Key, Device_Type,
         Device_Category, Brand)
        timing = hash(timing);
        command = (code_irdb.function, code_irdb.device,
         code_irdb.sub_device, code_irdb.brand);
        record = <timing, command>;
        DB_timings[record.timing] = record;
    end
return DB_timings
Function DB_query(DB_timings, timing):
    timing = hash(timing);
    command = DB_timings[timing];
return commmand
//Decode input timings sequence
Function decoder(DB_timings, timingsSequence):
    timings = split(timingsSequence);
    keysSequence = [];
    for timing in timings do
        command = DB_query(DB_timings, timing);
        keysSequence.append(command);
    end
return keysSequence
```

C Supporting Figures

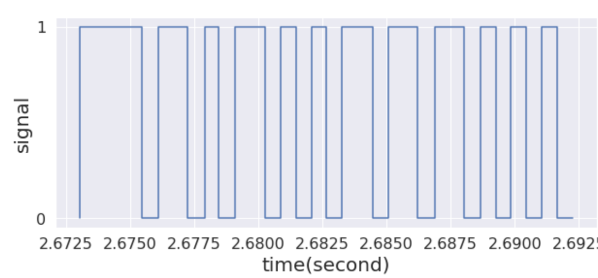


Figure 11: The signals captured by the COTS IR receiver module when "OK" key of the Sony TV remote is pressed.

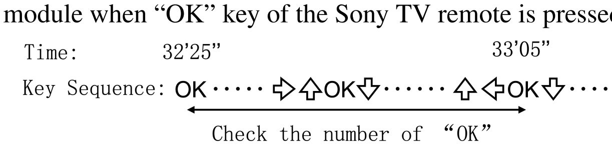


Figure 12: Key sequence about virtual keyboard inputs.