Cybersecurity

RESEARCH Open Access

Valet attack on privacy: a cybersecurity threat in automotive Bluetooth infotainment systems

Vishnu Renganathan*, Ekim Yurtsever, Qadeer Ahmed and Aylin Yener

Abstract

Modern automobiles are equipped with connectivity features to enhance the user's comfort. Bluetooth is one such communication technology that is used to pair a personal device with an automotive infotainment unit. Upon pairing, the user could access the personal information on the phone through the automotive head unit with minimum distraction while driving. However, such connectivity introduces a possibility for privacy attacks. Hence, performing an in-depth analysis of the system with privacy constraints is extremely important to prevent unauthorized access to personal information. In this work, we perform a systematic analysis of the Bluetooth network of an automotive infotainment unit to exploit security and privacy-related vulnerabilities. We model the identified threat with respect to privacy constraints of the system, emphasize the severity of attacks through a standardized rating metric and then provide potential countermeasures to prevent the attack. We perform System Theoretic Process Analysis for Privacy as a part of the systematic analysis and use the Common Vulnerability Scoring System to derive attack severity. The identified vulnerabilities are due to design flaws and assumptions on Bluetooth protocol implementation on automotive infotainment systems. We then elicit the vulnerability by performing a privacy attack on the Automotive system in an actual vehicle. We use Android Open-Source Project to report our findings and propose defense strategies.

Keywords: Bluetooth, Privacy attack, Automotive infotainment unit, STPA-Priv, Common vulnerability scoring system, Android open-source project

Introduction

The comfort of a vehicle is assessed based on the level of convenience they provide to the driver and the passengers. The minimum level of convenience expected from a modern automotive is to perform certain activities like texting, calling, being notified about important events, etc., without being distracted while driving. Thus, automotive manufacturers work continuously to improve connectivity features to achieve smart mobility. However, increasing functionalities pave the way for multiple security vulnerabilities and potential attacks. For example, increased wireless connectivity increases the number of external devices with access to the vehicular network. Thus, the number of possible weak points to exploit a system increases, through which unsafe control actions leading to hazardous situations can be achieved (Dardanelli et al. 2013; Onishi et al. January 2017). Moreover, the gap between the attacks and the security mechanisms continues to widen. The standard defense techniques in software security, like firewalls and cryptography, fail in complex embedded systems at the vehicular level because of the heterogeneous nature of many vehicular variants and configurations (Cheah et al. 2018).

System Theoretic Process Analysis (STPA) is a hazard analysis technique based on systems theory to address safety and security as a control problem using functional control diagrams. STPA-security focuses on the methods of controlling system vulnerabilities by securing

*Correspondence: renganathan.5@osu.edu

Department of Electrical and Computer Engineering, The Ohio State University, Columbus, OH, USA

those control actions that leads to vulnerabilities. In this work, we perform component, sub-system, and system-level verification and validation of the network modules and their Operating System (OS) in an automotive through STPA-Privacy (STPA-Priv) (Kumaraguru and Cranor 2005; Spiekermann et al. 2015). We have chosen a model-based approach as we know the state machines (Benton et al. 2013) and the sub-component interactions with the system. By understanding the implementation of Bluetooth in the automotive unit, we derive a model representation to perform analysis rather than a black box approach (Felt et al. 2012). By performing a top-down approach—STPA-Priv for analyzing the system for privacy constraints- we efficiently identify the most security-critical part of the system. We propose security mechanisms that would potentially prevent the violation of privacy constraints. We also emphasize the importance of performing a structured analysis, which is intuitive in evaluating the privacy constraints even during the design process.

From the analysis of the Bluetooth components, it was identified that the implementation of the Bluetooth stack at the software level plays a vital role in security. This implies that updates at the firmware level could introduce new security vulnerabilities, and it is recommended to have it re-tested for Bluetooth SIG certification (BluetoothSIG 2019a). Our work in this paper focuses on system analysis, and we propose an efficient method for identifying vulnerable control actions in Bluetooth implementation apart from identifying privacy constraints (Barth et al. 2019) in Android-based automotive head units. We use Android Open-Source Project (AOSP)—Android Automotive OS 11 (Android 2021a) to delve deeper into the software implementation, experiment with changes in the Bluetooth stack, and publish vulnerabilities and potential attacks. We also evaluate the attacks on an infotainment unit from an actual vehicle running Android 6.0.1 to validate our results.

In this work, we emphasize the need for privacy-related constraints and security precautions in system design. The privacy referred to in this paper is personal information that unknowingly gets shared with the attacker, who might exploit the gathered information (Kumaraguru and Cranor 2005). This research is based on the premise that the user trusts the automotive infotainment system and gives permission to share and synchronize personal information with the system by agreeing to the disclosure agreement. But the user is unaware of the changes that might have affected the system's privacy when the user is not near the vehicle. This premise supports the work conducted by Spiekermann et al. (2015) that the users share their personal information only if they are aware of the information exchange process. Thus, the system compromised in their absence is completely neglected. We keep this in mind, along with the difficulties faced in comprehending permissions (Benton et al. 2013, Felt et al. 2012 and Deuker 2009) while proposing countermeasures. The motivation for our work is mainly from UN Regulation No. 155 (UN R155), which requires the setup and implementation of a management system that focuses on vehicular cybersecurity and mandates vehicular requirements for complying with them. The UN R155 recommends information breaches (personal data that may be breached) when car users are changed as a potential vulnerability that could be exploited. Also, the proposed mitigation to unauthorized access to the owner's private information, such as personal identity, payment-related information, or address book information, is through system design and access control to protect safety-critical personal data. Hence, we present a system theoretical framework for analyzing the system for vulnerabilities and provide an access control solution to rectify the vulnerability rather than providing a usual tactical strategy for defending them.

The contributions of this paper are:

We present a vulnerability due to the implementation of Bluetooth stack in an Automotive infotainment system and implement a privacy attack of accessing confidential data from the personal device by exploiting the vulnerability.

1. We provide a systematic methodology—STPA-Priv to analyze the system, which we use to find the most vulnerable part of the system that potentially led to hazards.
2. We derived defense strategies from the attack and system analysis that could be implemented in the Bluetooth stack to prevent the attack.

The remainder of the paper is structured as follows: We summarize the work related to automotive Bluetooth attacks in "Related work" section. In "Background" section, we briefly overview Bluetooth connectivity, Bluetooth profiles, and implementation of the Bluetooth stack in Android Automotive. In "Threat model and attack description" section, we introduce the threat and provide an attack description. We rate our proposed attack using the Common Vulnerability Scoring System (CVSS) metrics and elicit the attack by performing system analysis in "Attack rating and system analysis" section. In "Potential countermeasures" section, we propose potential countermeasures for defending against the attack and discuss the attack results and effectiveness of our proposed countermeasures and provide a conclusion of our work in "Conclusions" section.

Related work

This section discusses literature related to vulnerabilities in automotive Bluetooth security, systematic verification methodologies, and attacks. General Bluetooth security architecture and security issues are well defined (Dunning 2010; Hassan et al. 2018; Claverie and Teves 2021) and privacy issues in automotive applications are discussed in Garakani et al. (2018), Kaplun and Segal 2019, Hussain and Koushanfar 2018, Zelle et al. 2017, Zhang et al. 2018. In this paper, we document our analysis with respect to the automotive infotainment unit, as it is completely different from the conventional mobile and PC platforms (Cheah et al. 2017). For example, authentication with manual interaction like numeric comparison or static/dynamic PIN increases security (Failed 2002). A survey by Oka et al. (2014) explains the drawbacks and elucidates the attack scope of using a static PIN. Cheah et al. 2017 classifies Bluetooth attacks into "Surveillance, range extension, obfuscation, fuzzing, sniffing, denial of service, unauthorized direct data access, malware and man in the middle." They perform a systematic attack tree-based security evaluation of the automotive Bluetooth. One of the attacks considered in the tree is data extraction by Object Exchange Protocol (OBEX) (nOBEX, "nOBEX," 2016). nOBEX is a tool built on top of PyOBEX (Boddie 2017; Ballmann 2021), which allows Bluetooth profiles (Phone Book Access Profiles-PBAP and Message Access Profiles-MAP) to be cloned as virtual filesystems from a real phone. Readers can refer (Megowan et al. 2003) for more information on OBEX. Then these virtual filesystems can function as a PBAP/MAP server to the automotive head units and inject malformed user data into the In-Vehicle Infotainment (IVI) system. The attack is stealthy as it mimics a real phone by providing support for Hands-Free-Profile (HPF) and AT command (ATtention Command) responses. The attack is hazardous if the automotive head unit with malformed packets is connected to the same CAN BUS (Controller Area Network) with other safety-critical Electronic Control Units (ECU) like Engine Control Module (ECM) and Transmission Control Module (TCM). (Checkoway et al. 2011) identified vulnerabilities in the custom-built software of the telematics unit. They gained access to the telematics ECU's OS and found around 20 unsafe calls to strcpy. They found that this vulnerability could be exploited to execute arbitrary code on the telematics unit. (Failed 2022) reverse engineered the IVI unit of a specific vehicle to find vulnerabilities and took control of the vehicle by injecting CAN frames. They also developed a module "Metasploit", to control the IVI unit and inject CAN frames.

Bluesnarfer (Nasim 1206) is a privacy violation attack using AT commands. (Zhou 2014) states that vulnerabilities in smart devices are easily susceptible to leaking sensitive and personal information to attackers. This is exploited due to the requirement of trust between the paired Bluetooth devices (Kaur and Jain 2013). Also, authentication mechanisms are executed only once during the initial pairing process. For ease of use and hands-free application, redundant authentication is avoided. However, upon re-connection, the devices are completely unaware of the malicious changes that the other device has been through (Yadav et al. April 2016). (Antonioli et al. 2022) perform KNOB (CVE-2019–9506) (Antonioli et al. 2019) and BIAS (CVE-2020-10,135) (Antonioli et al. 2020) attacks to impersonate Bluetooth devices in a vehicle. They evaluated popular infotainment units from five vehicle manufacturers for information disclosure attacks and remote code execution.

Background

This section provides a brief background of STPA system analysis and automotive Bluetooth. The Bluetooth background section is arranged in a hierarchical structure. We first introduce Bluetooth as a wireless communication medium, then the higher-level infotainment system, followed by the specific OS in the infotainment unit. We then discuss the implementation of the Bluetooth architecture in the OS, followed by low-level Bluetooth control actions such as Bluetooth profiles and connection mechanisms.

System theoretic process analysis (STPA)

STPA performs analysis of a system in a hierarchical manner. STPA aims to identify those control actions that each sub-system is trying to modify the behavior of components at their next lower level (Young and Leveson 2013). The control actions are the constraints that enforce safety and security at the system and sub-system level. Applying STPA to our case (automotive Bluetooth security for enhancing privacy) is a four-step procedure elaborated in "System analysis for privacy issues" section. In this section, we will discuss the process of establishing the system and identifying the most vulnerable subcomponent from the privacy perspective. The higher-level privacy goal that we consider at this point is to protect the confidential user data that is shared with the infotainment unit through Bluetooth. With this privacy constraint, we analyze the system from a higher level to a lower level to identify what essential components must be secured against disruptions. Thus, the high-level system under consideration is an automotive infotainment system with Android OS. Then comes the Bluetooth implementation in the OS, followed by the safety critical control actions pertaining to privacy in the implementation.

Bluetooth overview

Bluetooth is a wireless technology operating in the unlicensed 2.4 GHz Industrial, Scientific, and Medical (ISM) band. Bluetooth uses Ultra High Frequency (UHF) with an effective range of operation being 10–100 m (without external range extenders like amplifiers and directional antenna) (Hassan et al. 2018). During connection in Bluetooth, one device is designated as the leader, and all other devices are followers. Bluetooth uses Frequency-Hopping Spread-Spectrum (FHSS) to move through 1600 frequencies per second. Thus, each channel is used only for 625 microseconds. Upon a successful connection, the follower synchronizes with the leader's clock to get the correct frequency hopping pattern. With 79 frequency channels to hop, the probability of interference between other Bluetooth devices is extremely low (Cope et al. 2017). The two main types of Bluetooth devices are (i) The classic Bluetooth device that operates at Basic Rate (BR) or Enhanced Data Rate (EDR) and (ii) Bluetooth Low Energy (BLE) (Antonioli et al. 2022). These devices with different architectures communicate with each other in dual mode. Bluetooth is secured through authentication, encryption, and authorization. All Bluetooth devices have a unique 48-bit address (BD_ADDR) assigned by the manufacturer.

The core components of the Bluetooth architecture are (i) Bluetooth Controller, (ii) Host Controller Interface (HCI), and (iii) Bluetooth Host.

Automotive infotainment unit

Modern IVI units aid the driver with audio, video entertainment, and navigation and are usually connected to the in-vehicle network to provide access to Heating, Ventilation, and Air Conditioning (HVAC) controls and some critical driver assistant features like park assist. The infotainment unit provides connectivity features to access the applications from the phone while driving. By pairing the phone with the infotainment unit, the user could access the phone's contacts and messages and perform Hands-Free calling or texting with minimum distraction while driving. The user performs these operations through the infotainment unit's Human–Machine Interface (HMI) (Bhat 2015). The intuitive and effective HMI in the front end is offered through an automotive OS. Recently, automotive manufacturers and Original Equipment Manufacturers (OEMs) have been experimenting and updating different OS. Some OEMs already have a customized version of Android in their production vehicles.

Android automotive operating system

In this paper, we focus on the Android automotive and perform system level, sub-system level, and control level analysis on the Bluetooth functionalities of the IVI system. The IVI unit that runs Android OS needs Bluetooth, Wi-Fi, and Telematics Control Unit (TCU) in interface with the vehicular network—CAN. The IVI unit performs different actions upon reception of data as input from a user. For example, with the help of HMI, the user could pair a mobile device with the head unit through Bluetooth and utilize hands-free applications like make/attend calls. Due to the addition of low-level vehicular network modules like CAN, Android's framework has a new addition of a Hardware Abstraction Layer (HAL) called Vehicle HAL (VHAL) (PK 2019). However, we focus on Automotive Services (GAS) and the system applications provided by Android (2021a) (Green boxes in Fig. 1).

GAS is a set of specific technical services defined by the Android development team (Gessler et al. 2020). Maps and Navigation, Playstore, Voice Assistant, Setup-Wizard, and Automotive Keyboard are some essential services from GAS. Apart from GAS, other important applications are (i) Media Center for the integration of the media player, (ii) Dialer for telephone application from the connected smartphone, (iii) Car Settings for the management of car system settings and (iv) Notification Center for system notifications from smartphone and the vehicle. The Android automotive architecture provides frameworks and libraries for perfect integration with wireless modules like Bluetooth, Wi-Fi, VHAL module, and other applications.

Android's bluetooth architecture

The abstracted version of Android's Bluetooth architecture is shown in Fig. 2 (Android 2021b). android.bluetooth APIs in the application layer communicate with the Bluetooth services and Bluetooth profiles located in packages/apps/Bluetooth through Binder. The Bluetooth process communicates to the Bluetooth stack through Java Native Interface (JNI). The configurations required in the HAL are implemented through the Bluetooth stack. The customizable Bluetooth stack communicates with the embedded Bluetooth chipset through HAL Interface Definition Language (HIDL). The lower-level controls on the Bluetooth chipset include radio controller, baseband controller, etc. They communicate with the host through HCI. The respective hosts implement the protocols in the middle layer (Bluetooth stack and Bluetooth Process) and the application layer. Thus, the implementation of the Bluetooth stack and the requirement of Bluetooth profiles depend on the host, and any changes to the stack or the profiles can introduce new vulnerabilities. These

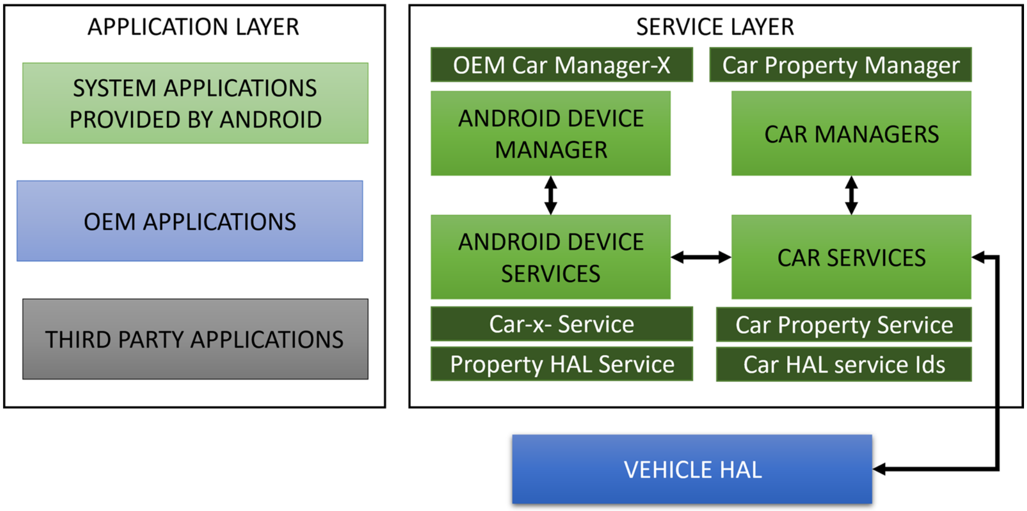


Fig. 1 Android automotive architecture depicting automotive services and system applications

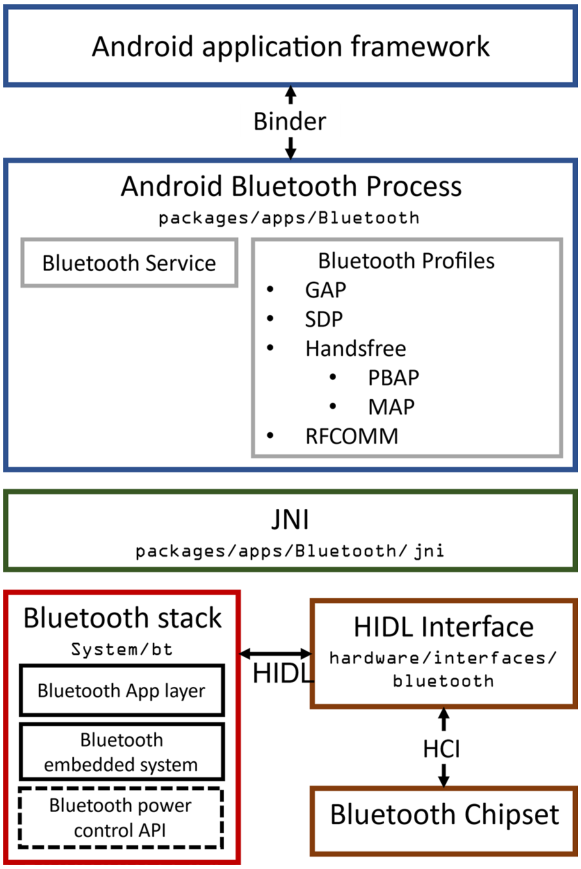


Fig. 2 Android's Bluetooth architecture

vulnerabilities are the implementation bugs introduced in the OS. Hence, getting the device re-tested and re-certified by Bluetooth SIG is highly recommended.

To narrow down our analysis in accordance with STPA, we specifically select those sub-systems that lead to vulnerabilities and violate our declared privacy constraint. Bluetooth profiles and the Bluetooth stack (system/bt) are two such sub-systems. This is because the process of storing and erasing confidential user data is executed in the Bluetooth stack, and permission for accessing the user data is obtained through the Bluetooth profiles.

Android's bluetooth stack

The Android's Bluetooth stack is called Bluedroid (shown in Fig. 2). The Bluetooth power control API belongs to the Bluetooth Kernel in the Bluetooth stack and communicates to the Bluetooth chipset through a function call. The libraries in power control API coinciding with our scope of analysis are CarPowerManager and CarPowerManagementService. Shut down and memory clear by Suspend-to-RAM (STR) are two significant outcomes of these libraries. Based on STPA, these control actions are shortlisted because they are responsible for the erasure of confidential user data. The state machine for the libraries is given in Fig. 3. The Vehicle Master Control Unit (VMCU) triggers the state transitions, and the integrator is used to ensure that the shutdown process is not infinitely delayed (Android 2021c).

The important transitions in the state machine are:

(i) *On*: The VHAL module instructs the OS to enter the ON state. The OS is fully functional at this level.

(ii) *Shutdown Prepare*: In this stage, the IVI system is in OFF state, but the OS is still running in the background for updates.

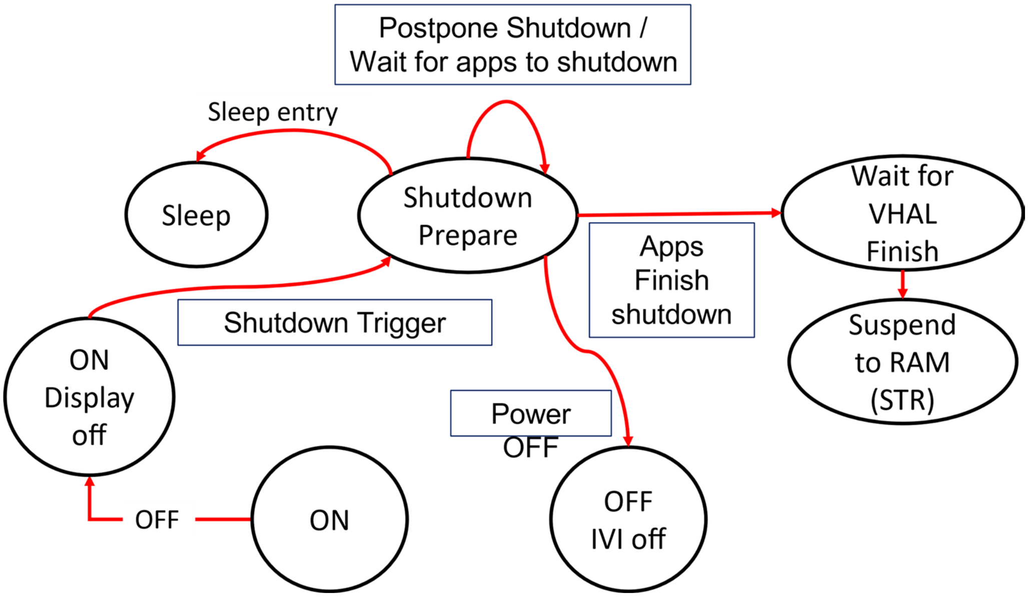


Fig. 3 Car power state machine. The states (ellipses) are triggered through transitions

(iii) *Wait for VHAL*: This is when the user is still interacting with the vehicle. VHAL still powers the System on Chip (SoC).

(iv) *Wait for VHAL Finish*: The OS is ready for shutdown. The SoC is in deep sleep, and the application processor is powered off. The OS then moves to the STR state.

(v) *STR*: The SoC and the vehicle is off, and codes are not executed.

These transitions in the state machine are important, as we will see in Sect. 4 on how we could exploit them to perform attacks on the system.

Bluetooth profiles

Bluetooth profiles define the standard protocol of applications of the Bluetooth device. It specifically defines what data is being transmitted via the Bluetooth connection. Depending on the profile, Bluetooth SIG has different physical transportation protocols. For example, we have OBEX for PBAP and Generic Attribute Profile (GATT) for BLE data transmission. With more than 30 standardized profiles, this paper focuses on the PBAP and MAP as they are the privacy-related Bluetooth profiles that store the data in plain text in the Bluetooth logs.

Phone book access profile (PBAP)

PBAP is based on client–server interaction where the client (Phone book Client Equipment—PCE) receives the phone book object from the server (Phone book Server Equipment—PSE) device. In our case, the user's phone is PSE, and the IVI unit is PCE. For its convenience in hands free application in the vehicle, PBAP is one of the most important Bluetooth profiles in the IVI unit. With this being the case, the Bluetooth SIG (BluetoothSIG 2019b) mandates specific security requirements for PBAP: (i) PCE could request PSE for phone book access only after a successful connection. (ii) The connection initialization should include service discovery, security initialization messages, link keys, and encryption. (iii) Authentication procedure as described in Generic Access Profile (GAP) should be accomplished. (iv) The user of the PSE should confirm the access for sharing their phone book.

The entire phone book is usually downloaded and stored in the PCE device. The data transmission from the PSE to the PCE uses the Generic Object Exchange Profile. The download process for PBAP (BluetoothSIG 2019b) is shown in Fig. 4.

(1). The Bluetooth stack requests for a Hands-Free Profile connection to the phone book storage manager once it receives permission from the user to share the Phone Book. (2). The phone book manager then updates the HMI to show all phone book data like recent calls, favorite contacts, etc. (3). Upon a successful connection, the phone book manager sends a successful PBAP connection acknowledgment to the stack. (4). Along with the acknowledgment, it also sends

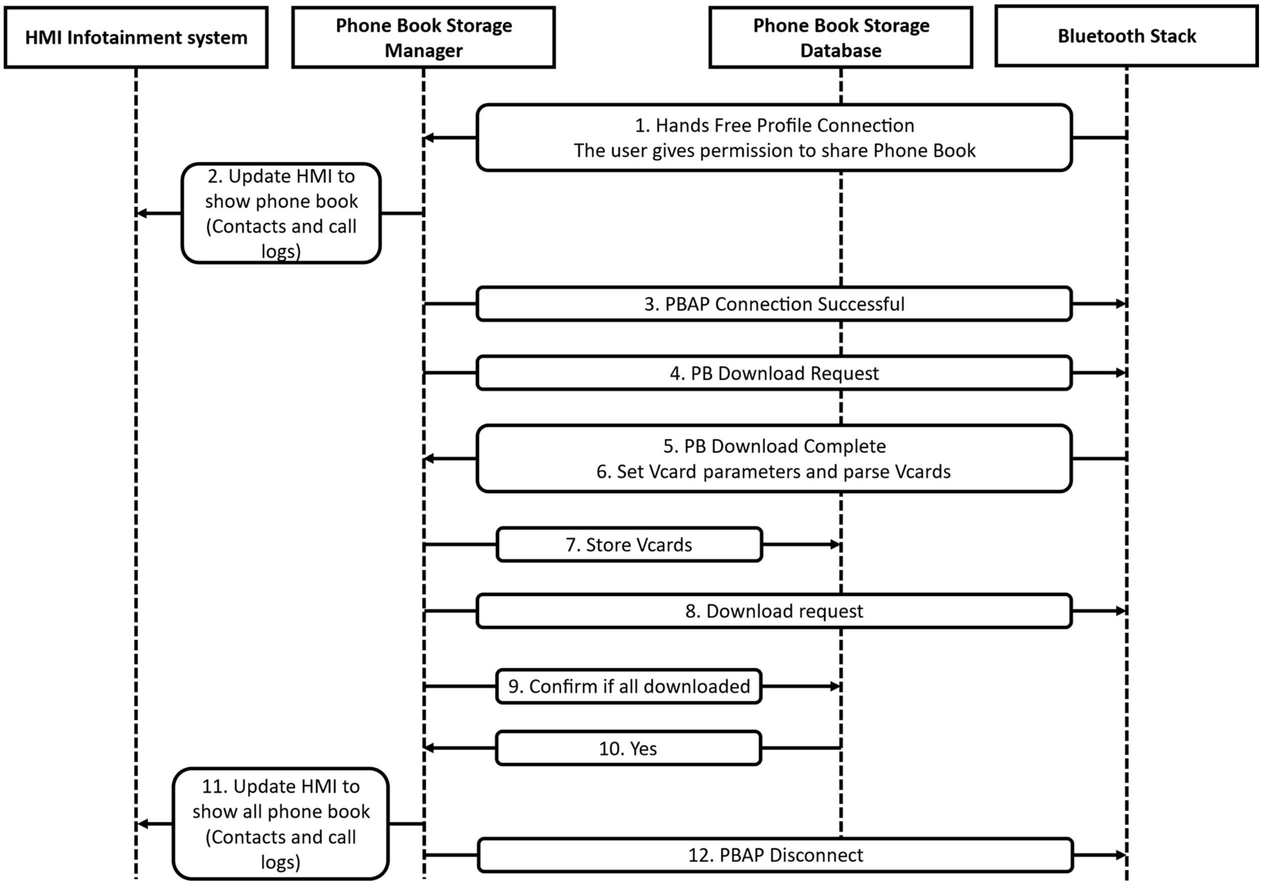
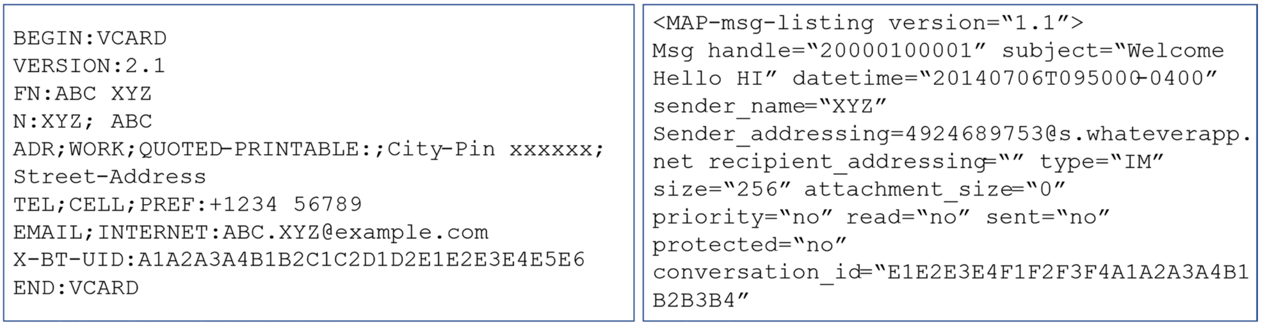

HMI Infotainment system
Phone Book Storage Manager
Phone Book Storage Database
Bluetooth Stack
1. Hands Free Profile Connection
The user gives permission to share Phone Book
2. Update HMI to show phone book (Contacts and call logs)
3. PBAP Connection Successful
4. PB Download Request
5. PB Download Complete
6. Set Vcard parameters and parse Vcards
7. Store Vcards
8. Download request
9. Confirm if all downloaded
10. Yes
11. Update HMI to show all phone book (Contacts and call logs)
12. PBAP Disconnect

Fig. 4 Pone book access profile download process between the infotainment system and the Bluetooth device

phone book download request (5). Upon a successful download, the Bluetooth stack sends a download complete command to the phone book manager. (6). The phone book manager sets the vCard (Virtual Contact File -VCF) parameters and parses the vCards in accordance with storage requirements. An example vCard is shown in Fig. 5 (left) (7). The vCards are then stored in the phone book storage database. (8, 9, 10). Once the storage is successful in the database, the phone book manager requests the next set of phone book storage. Steps 4–7 are reiterated until all the contacts are stored in the database. (11). Upon the confirmation, the HMI is updated with all the contacts and phone book, and then the profile is disconnected in step 12.



```
BEGIN:VCARD
VERSION:2.1
FN:ABC XYZ
N:XYZ; ABC
ADR;WORK;QUOTED-PRINTABLE:;City-Pin xxxxxx;
Street-Address
TEL;CELL;PREF:+1234 56789
EMAIL;INTERNET:ABC.XYZ@example.com
X-BT-UID:A1A2A3A4B1B2C1C2D1D2E1E2E3E4E5E6
END:VCARD
```

```
<MAP-msg-listing version="1.1">
Msg handle="20000100001" subject="Welcome
Hello HI" datetime="20140706T095000-0400"
sender_name="XYZ"
Sender_addressing=4924689753@s.whateverapp.
net recipient_addressing="" type="IM"
size="256" attachment_size="0"
priority="no" read="no" sent="no"
protected="no"
conversation_id="E1E2E3E4F1F2F3F4A1A2A3A4B1
B2B3B4"
```

Fig. 5 Example vCard format (left) and message format (right)

Message access profile (MAP)

MAP is similar to PBAP and uses similar client–server interaction to exchange message objects (Message Client Equipment—MCE and Message Server Equipment—MSE). MAP in the hands-free profile of the IVI unit provides the convenience of using the HMI or even voice commands through the audio system to easily read, send, notify, or browse messages. The supported MAP versions are (BluetoothSIG 2019c) SMS, MMS, email, and Instant Messages (IM). The IM format that we exploit in our case is shown in Fig. 5 (right). The security requirements for MAP are very similar to the PBAP—the pairing and encryption requirements, authentication with GAP, and user conformance. MAP is based on the OBEX profile, and the following OBEX services are used: the message Access service (MAS) and the Message Notification Service (MNS). In all services except MNS, MCE acts as OBEX client and MSE as OBEX server. In MNS, MSE acts as an OBEX Client and connects to the MCE that acts as an OBEX Server. In addition to the OBEX profile for data transmission from the client to the server, MAP also uses PBAP for referencing contacts.

Threat model and attack description

Bluetooth security enforces authentication, authorization, and encryption based on the premise that the user trusts the device with which they are pairing their personal device (Tschirschnitz et al. 2021). However, this assumption cannot always be true with IVI units in vehicles, especially the ones that are handled by multiple users. An attacker could manipulate the IVI unit and stealthily compromise the user's privacy. We have formulated three attack scenarios in this paper, as shown in Fig. 6, which are pretty common in day-to-day vehicle usage.

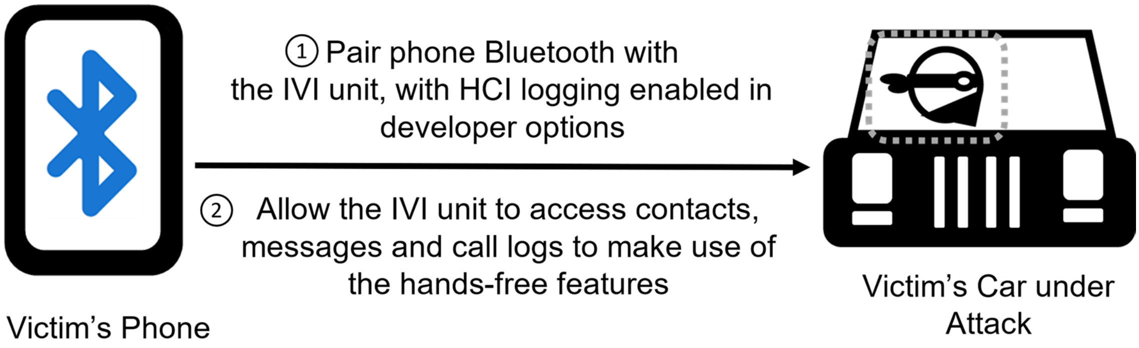
Android being a free and open-source OS, is very developer friendly. One of the best ways to test a developed software or application is by physically testing it on a device. Hence for testing purposes, Android has developer options, which lets the developer access some device features that are usually locked. Developer options in Android devices are "hidden" in an easily accessible location. According to the Android community, it is safe and secure to have developer options enabled, and enabling developer options would not void the device's warranty. Also, the usual working functions of the device is not altered in any way by enabling developer options. Hence it is exceedingly difficult for a typical user to know if the developer options are enabled or not unless they investigate it in the device settings. An essential feature of developer options is Bluetooth HCI snoop log. This snoop log stores PBAP and MAP in plain text. The log file is created upon a Bluetooth connection and captures, monitors, and analyzes Bluetooth packets. This data is stored in the device and can be retrieved through USB or wireless Android debugging.

In this paper, we exploit this feature to successfully conduct privacy attacks on the IVI unit with Android OS. Successful attack execution depends on developer options being enabled, which we consider an implementation bug and an exploitable weakness in Android's Bluetooth stack. In the three attack scenarios we have considered, the attacker had secretly enabled the developer options in the IVI unit, and the infotainment system users are entirely unaware of this. When the user pairs their phone with the IVI system using Bluetooth for hands-free applications like calling, texting, and entertainment, their personal information gets synchronized with the IVI unit according to the Bluetooth profiles. The attacker can retrieve this stored information either through a wired connection with the vehicle (USB—In our case/CAN) or a wireless connection (common Wi-Fi network—in our case). The data (personal contacts, messages, and call logs) retrieved by the attacker through the developer options are not encrypted. After analyzing the IVI system—car power state machine in Fig. 3, we also found that the memory clear (suspend to RAM) function is triggered by the vehicle's shutdown and not on the Bluetooth connection status. Thus, the data stored in the IVI system stays in the system until the vehicle ignition is turned off, increasing the time window for attack. We tested our proposed attacks on an actual production vehicle with Android OS-Version 6.0.1 in their infotainment system. For publication purposes, we report our attacks on the Android Open Source (AOS) platform (Android 2021d). Also, the attack is consistent with IVI units with Android OS and an option to enable developer options (Android versions 4.2 and higher) (Failed 2022). The experimental setup is shown in Fig. 7. The Android Automotive OS 11 was set up on Raspberry Pi-4B (Sutton 2021).

The procedure for the attack is as follows:

1. Enable developer options in the Android IVI unit.

    a. In some systems, this could be a similar operation to the Android mobile phone. In devices with Android 9 or higher, by tapping Build Number 7 times in—Settings > About Phone > Build Number (Android 2021e)


① Pair phone Bluetooth with the IVI unit, with HCI logging enabled in developer options
② Allow the IVI unit to access contacts, messages and call logs to make use of the hands-free features
Victim's Phone
Victim's Car under Attack

Victim's personal information from phone gets synced to the infotainment unit and stored in plain text format. Making it easily accessible to the attacker wired/ wirelessly (wired–physical access with the car, wirelessly–car connected to a common network). The data can also be transferred to storage devices without any traces of attack.

Attack Scenario 1 (Valet Attack)

The victim hands over the car to valet and walks away. The Bluetooth connection with the phone gets disconnected beyond a certain range. However the data in the infotainment unit is still available to the attacker (in this case, the valet driver).

Attack Scenario 2 (Rental Car)

The victim pairs the phone with a rental car to use hands-free features and returns the car without deleting the personal data. The victims personal data could be accessible to the attacker or the next potential users of the vehicle.

Attack Scenario 3 (Shared Car Attack)

The victim pairs the phone with a shared car (Taxi) to enjoy their music playlist. If the user accepts to share data along with the music, their private information gets compromised to the attacker (driver in this case).

Fig. 6 High-level attack overview—attack scenarios in which the stealthy attacker compromises the victim's personal information

b. In some vehicles, this feature could be hidden by the manufacturer, which can be unlocked by selecting a combination of buttons in the IVI unit. However, deciphering the combination of the buttons is not complicated; for the popular units, they are available on online forums.

2. Enable Bluetooth HCI snoop log under the developer options Settings > Developer options > Enable Bluetooth HCI snoop log.
3. Retrieve the data from the IVI unit after the user has connected their personal device

   a. The log file is usually stored in memory storage. This can be transferred to a memory device through the USB port. We can also use the Android debugging bridge and pull the log file through ADB pull command with a Linux terminal (when the Linux computer and the IVI system are connected over the same network).

4. Analyze the log captured in Wireshark. The Bluetooth OBEX packets captured in the log reveal the phone contacts, call logs, and messages in plain text, as shown in Fig. 8.

The analyzed Bluetooth packets revealed all the contacts and text messages till the last moment from accessing the stored information in the paired IVI device. This implies that the attacker would have access to all

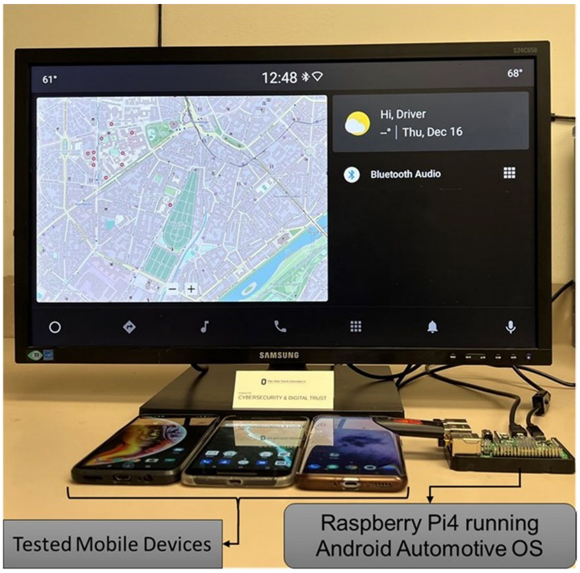


Fig. 7 Experimental Setup mimicking the Android IVI unit on a raspberry Pi4 and three tested mobile devices for privacy attack

the contacts and the previous and current text messages, which might include security-critical messages from banks, password reset messages, or even One-Time Passwords (OTPs), which could lead to adverse privacy violations. For example, Fig. 8 reveals the messages received by the victim from their banks and a sign-in attempt from an e-commerce website to the attacker in plain text. With the steps in performing the attack as mentioned, not being technically complicated and not requiring expensive computational tools like most cybersecurity breaches, the attacker could easily use the proposed attack to exploit the privacy of a targeted victim.

$$\begin{aligned} Exploitability &= 8.22 \times AttackVector \times AttackComplexity \times PrevilegesRequired \times UserInteraction \\ &= 8.22 \times 0.55 \times 0.77 \times 0.27 \times 0.62 = 0.5827 \end{aligned}$$

$$\begin{aligned} Impact\ SubScore(ISS) &= 1 - \left[\left(1 - confidentiality\right) \times \left(1 - integrity\right) \times \left(1 - availability\right)\right] \\ &= 1 - [(1 - 0.56) \times (1 - 0) \times (1 - 0)] = 0.56 \end{aligned}$$

Attack rating and system analysis

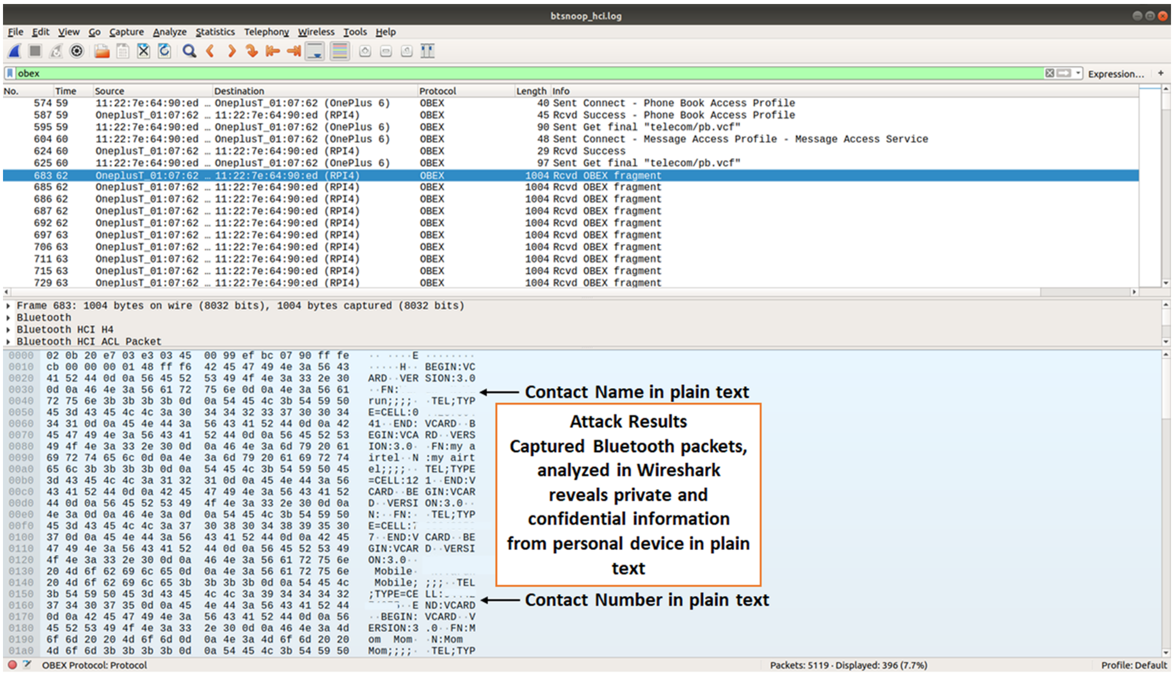
In addition to identifying a privacy-related issue in the system, we provide a methodology to model the privacy threat. By modeling the threat with respect to the system, we emphasize the severity of the attack, elicit privacy requirements, and suggest potential countermeasures for the attack. We have modeled the system in accordance with (Shapiro 2016) STPA-Priv. Other data flow techniques for privacy analysis like "conditional flow identification and joint flow tracking" (Lu et al. 2015), data exchanging, and data observing techniques (Egele et al. 2011; Enck et al. 2014) mainly focus on the analysis of the data origin device and not on the data that has been shared with consent to other known devices. Another comprehensive data flow analysis is LINDDUN (linkability, identifiability, non-repudiation, detectability, information disclosure, unawareness, and noncompliance), proposed by Wuyts et al. (2014). LINDDUN uses a bottom-up approach in analyzing the privacy constraints in the data flow, unlike the STPA-Priv, which uses a top-down approach. Intuitively, tracking data flow in a bottom-up approach becomes much more complex when human interaction through a user interface is involved (Mindermann et al. 2017).

As the first step, we analyze the severity of the attack using the CVSS (Mell et al. 2006)—a score ranging from 0.0–10.0 (Qualitative Severity Rating Scale). Based on the CVSS score, a rating of None: 0.0, Low: 0.1–3.9, Medium: 4.0–6.9, High: 7.0–8.9 and Critical: 9.0–10.0 is assigned. The CVSS consists of three metric groups: Base, Temporal, and Environmental. The base score implies the severity of the vulnerability, the temporal score implies the factors that change over time, and the environmental score implies the changes due to the computing environment. We focus on the base score, as the other two can be highly volatile. The numerical values are derived in accordance with the CVSS metric, as shown in Table 1. The exploitability metrics define the characteristics of the vulnerability, and the impact metrics define the effect of the exploited vulnerability on the component that suffers the worst outcome due to the attack (Mell et al. 2006).

The impact and the base score for unchanged scope is $6.42 \times ISS = 3.5952$

$$\begin{aligned} &BaseScore \\ &\quad = Roundup\left(Minimum\left[\left(Impact + Exploitability\right), 10\right]\right) \\ &\quad = 4.2 \end{aligned}$$

Thus, from the qualitative CVSS rating scale, the attack proposed has a medium severity. However, the attack has an even better scope as more vehicle manufacturers are moving towards Android Automotive OS in their IVI units. By 2023, it is expected to have more than 40 cars


btsnoop_hci.log
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
obex
Expression…

```
No.     Time  Source                Destination                       Protocol   Length Info
   574 59     11:22:7e:64:90:ed … OneplusT_01:07:62 (OnePlus 6)     OBEX          40 Sent Connect - Phone Book Access Profile
   587 59     OneplusT_01:07:62 … 11:22:7e:64:90:ed (RPI4)          OBEX          45 Rcvd Success - Phone Book Access Profile
   595 59     11:22:7e:64:90:ed … OneplusT_01:07:62 (OnePlus 6)     OBEX          90 Sent Get final "telecom/pb.vcf"
   604 60     11:22:7e:64:90:ed … OneplusT_01:07:62 (OnePlus 6)     OBEX          48 Sent Connect - Message Access Profile - Message Access Service
   624 60     OneplusT_01:07:62 … 11:22:7e:64:90:ed (RPI4)          OBEX          29 Rcvd Success
   625 60     11:22:7e:64:90:ed … OneplusT_01:07:62 (OnePlus 6)     OBEX          97 Sent Get final "telecom/pb.vcf"
   683 62     OneplusT_01:07:62 … 11:22:7e:64:90:ed (RPI4)          OBEX        1004 Rcvd OBEX fragment
   685 62     OneplusT_01:07:62 … 11:22:7e:64:90:ed (RPI4)          OBEX        1004 Rcvd OBEX fragment
   686 62     OneplusT_01:07:62 … 11:22:7e:64:90:ed (RPI4)          OBEX        1004 Rcvd OBEX fragment
   687 62     OneplusT_01:07:62 … 11:22:7e:64:90:ed (RPI4)          OBEX        1004 Rcvd OBEX fragment
   692 62     OneplusT_01:07:62 … 11:22:7e:64:90:ed (RPI4)          OBEX        1004 Rcvd OBEX fragment
   697 63     OneplusT_01:07:62 … 11:22:7e:64:90:ed (RPI4)          OBEX        1004 Rcvd OBEX fragment
   706 63     OneplusT_01:07:62 … 11:22:7e:64:90:ed (RPI4)          OBEX        1004 Rcvd OBEX fragment
   711 63     OneplusT_01:07:62 … 11:22:7e:64:90:ed (RPI4)          OBEX        1004 Rcvd OBEX fragment
   715 63     OneplusT_01:07:62 … 11:22:7e:64:90:ed (RPI4)          OBEX        1004 Rcvd OBEX fragment
   729 63     OneplusT_01:07:62 … 11:22:7e:64:90:ed (RPI4)          OBEX        1004 Rcvd OBEX fragment

▸ Frame 683: 1004 bytes on wire (8032 bits), 1004 bytes captured (8032 bits)
▸ Bluetooth
▸ Bluetooth HCI H4
▸ Bluetooth HCI ACL Packet

0000  02 0b 20 e7 03 e3 03 45  00 99 ef bc 07 90 ff fe   ·· ····E ········
0010  cb 00 00 00 01 48 ff f6  42 45 47 49 4e 3a 56 43   ·····H·· BEGIN:VC
0020  41 52 44 0d 0a 56 45 52  53 49 4f 4e 3a 33 2e 30   ARD··VER SION:3.0
0030  0d 0a 46 4e 3a 56 61 72  75 6e 0d 0a 4e 3a 56 61   ··FN:[redacted]
0040  72 75 6e 3b 3b 3b 3b 0d  0a 54 45 4c 3b 54 59 50   run;;;;· ·TEL;TYP
0050  45 3d 43 45 4c 4c 3a 30  34 34 32 33 37 30 30 34   E=CELL:0 [redacted]
0060  34 31 0d 0a 45 4e 44 3a  56 43 41 52 44 0d 0a 42   41··END: VCARD··B
0070  45 47 49 4e 3a 56 43 41  52 44 0d 0a 56 45 52 53   EGIN:VCA RD··VERS
0080  49 4f 4e 3a 33 2e 30 0d  0a 46 4e 3a 6d 79 20 61   ION:3.0· ·FN:my a
0090  69 72 74 65 6c 0d 0a 4e  3a 6d 79 20 61 69 72 74   irtel··N :my airt
00a0  65 6c 3b 3b 3b 3b 0d 0a  54 45 4c 3b 54 59 50 45   el;;;;·· TEL;TYPE
00b0  3d 43 45 4c 4c 3a 31 32  31 0d 0a 45 4e 44 3a 56   =CELL:12 1··END:V
00c0  43 41 52 44 0d 0a 42 45  47 49 4e 3a 56 43 41 52   CARD··BE GIN:VCAR
00d0  44 0d 0a 56 45 52 53 49  4f 4e 3a 33 2e 30 0d 0a   D··VERSI ON:3.0··
00e0  4e 3a 0d 0a 46 4e 3a 0d  0a 54 45 4c 3b 54 59 50   N:··FN:· ·TEL;TYP
00f0  45 3d 43 45 4c 4c 3a 37  30 38 30 34 38 39 35 30   E=CELL:[redacted]
0100  37 0d 0a 45 4e 44 3a 56  43 41 52 44 0d 0a 42 45   7··END:V CARD··BE
0110  47 49 4e 3a 56 43 41 52  44 0d 0a 56 45 52 53 49   GIN:VCAR D··VERSI
0120  4f 4e 3a 33 2e 30 0d 0a  46 4e 3a 56 61 72 75 6e   ON:3.0·· [redacted]
0130  20 4d 6f 62 69 6c 65 0d  0a 4e 3a 56 61 72 75 6e    Mobile·
0140  20 4d 6f 62 69 6c 65 3b  3b 3b 3b 0d 0a 54 45 4c    Mobile; ;;;··TEL
0150  3b 54 59 50 45 3d 43 45  4c 4c 3a 39 34 34 34 32   ;TYPE=CE LL:[redacted]
0160  37 34 30 37 35 0d 0a 45  4e 44 3a 56 43 41 52 44   ·E ND:VCARD
0170  0d 0a 42 45 47 49 4e 3a  56 43 41 52 44 0d 0a 56   ··BEGIN: VCARD··V
0180  45 52 53 49 4f 4e 3a 33  2e 30 0d 0a 46 4e 3a 4d   ERSION:3 .0··FN:M
0190  6f 6d 20 20 4d 6f 6d 0d  0a 4e 3a 4d 6f 6d 20 20   om  Mom· ·N:Mom
01a0  4d 6f 6d 3b 3b 3b 3b 0d  0a 54 45 4c 3b 54 59 50   Mom;;;;· ·TEL;TYP

OBEX Protocol: Protocol                    Packets: 5119 · Displayed: 396 (7.7%)                    Profile: Default
```

Contact Name in plain text

Attack Results
Captured Bluetooth packets, analyzed in Wireshark reveals private and confidential information from personal device in plain text

Contact Number in plain text

```
BEGIN:VCARD
VERSION:2.1
FN:DEF TUV
N:3216549870;
ADR;WORK;QUOTED-PRINTABLE:;City-Pin xxxxxx;
Street-Address
EMAIL;INTERNET:DEF.TUV@example.com
X-IRMC-CALL-DATETIME;TYPE=RECEIVED:20210502T174042END:VCARD
```

```
<msg handle="040000000000223A"
subject="https://amazon.com/a/c/r
?k=AiFTNaAniAoi72H6rPRs3husqAmazo
n: Sign-in attempt from IL, US.
Tap link to respond."
datetime="20210420T133259"
sender_name=""
sender_addressing="25392"
type="SMS_GSM" size="111"
read="yes" />
```

```
<msg handle="0400000000002239"
subject="Chase QuickPay(R) with
Zelle(R): XXYYZZ sent you $90.97
&amp; it's ready now. Reply STOP
to cancel these texts."
datetime="20210417T195633"
sender_name=""
sender_addressing="24273"
type="SMS_GSM" size="132"
read="no" />
```

```
<msg handle="040000000000222D"
subject="Chase Fraud: Thank you for
confirming your transaction activity. If
the purchase was declined, it can be
attempted again."
datetime="20210331T230108"
sender_name="" sender_addressing="72166"
type="SMS_GSM" size="121" read="yes" />
```

```
<msg handle="040000000000222B"
subject="Chase Fraud: We declined $55.86
with card ending XXXX at ABC
Merchandising. Was this you? Reply YES
or NO. If yes, you will not be charged
unless you try again. If no, we will
close your current card and send you a
new one. Msg &amp; data rates may
apply." datetime="20210331T221507"
sender_name="" sender_addressing="72166"
type="SMS_GSM" size="255" read="yes" />
```

Fig. 8 Attack results in Wireshark revealing phone contacts and confidential user information to the attacker in plain text

Table 1 Vulnerability rating using CVSS metrics

Metrics defined by CVSS		Value	Numerical value
Exploitability matrix	Attack Vector (AV)	L	0.55
	Attack Complexity (AC)	L	0.77
	Privileges Required (PR)	H	0.27
	User Interaction (UI)	R	0.62
	Scope (S)	U	
Impact Metrics	Confidentiality (C)	H	0.56
	Integrity (I)	N	0
	Availability	N	0

from around 15 manufacturers have a version of Android Automotive in their infotainment systems, which could potentially be affected by the proposed attack (Popa 2021) (Honda, "Hondanews," 2021).

System analysis for privacy issues

The system analysis with STPA-Priv is a four-step process (Shapiro 2016), and the control actions are analyzed in accordance with the state machine diagram in Fig. 3.

1. Identify Adverse privacy consequences—These adverse privacy consequences can result from one or more vulnerable system states. In our case, the adverse privacy consequence is the loss of confidential user information to the attacker. This is because PBAP and MAP contain confidential user information and are stored in plain text in the Bluetooth log, which is accessible to the attacker through developer options.
2. Identify vulnerabilities that lead to adverse privacy consequences—We now identify the sub-system states or the environment states that lead to the adverse privacy consequences. Here, the identified vulnerability is that the memory clear is synchronized with the shutdown prepare state in the power control API. The libraries responsible for the power control are CarPowerManager and CarPowerManagementService, as mentioned in "Automotive infotainment unit" section. Also, this vulnerability increases the scope of the attack by providing a long duration of memory retention until the car ignition is off.
3. Identify system privacy constraints and functional control structure—In this step, we can assign some constraints that the system must enforce to mitigate identified vulnerabilities.

    a. One such constraint is that the user can be notified that the developer options is switched on with Bluetooth snoop logging enabled while pairing their device.
    b. Another constraint is that the memory clear must not be triggered upon the car ignition shutdown state.
    As discussed by Mindermann et al. (2017), Shapiro (2016), the control structure is not in our scope of analysis.
4. Identify privacy-compromising control actions—The control action analysis is again a two-step process

    a. Identifying erroneous control action for each privacy constraint as shown in Table 2
    b. Identify causal factors to the control actions, i.e., suggest mitigation strategies that would reduce the impact of the vulnerability. The proposed mitigation strategies are discussed in detail in "Potential countermeasures" section.

Table 2 STPA-Priv analysis of the automotive infotainment unit

Privacy constraint	Incorrect control action	No control action	Control action provided too soon or too late	Control action applied too long or not long enough
The user must be notified that the developer options with Bluetooth snoop logging enabled while pairing their device	Unauthorized capture of data from the infotainment unit	Notification on the developer options with Bluetooth snoop logging enabled	Privacy disclosure information not provided prior to connection establishment	
System memory clear	Memory clear (STR) is triggered upon car ignition shutdown state	Memory clear (STR) triggered upon Bluetooth disconnection status		Memory is not cleared until car ignition is in the off state

Potential countermeasures

This section proposes potential countermeasures based on the system analysis results. Even though encryption of the phone book storage manager is a viable option, it is not sufficient in our case. Some HMI of the infotainment system requires decrypted data for certain services. Hence, we formulate countermeasures for the specific attack to make sure the system is trustworthy with personal information (Tanaka et al. 2017).

i. Check if the developer options is enabled in the IVI unit. If the developer options are enabled, notify the user and request additional consent from the user. This would make the user aware of the potential attack situation or privacy risks. They might be prepared to turn off the ignition before handing over the vehicle to an attacker or be cautious when connecting their device to unknown cars.

ii. Check the Bluetooth status frequently, and if the check status is preparing for a disconnection, call the *Wait for VHAL Finish* state and proceed to the *Suspend to RAM (STR)* state. The STR state clears the memory and all the saved personal data on the IVI unit upon disconnection. Thus, the STR state automatically clears the memory when the user walks away after leaving the car to the attacker and goes beyond the Bluetooth operating range. Hence, to a great extent narrows down the time to attack. The updated state machine for the proposed countermeasure is shown in Fig. 9.

The pseudo-code for the proposed mitigations in AOSP—bluetoothdeviceconnectionpolicy.java—a device connection policy management to decide the Bluetooth connection and disconnection is given by Algorithm 1.

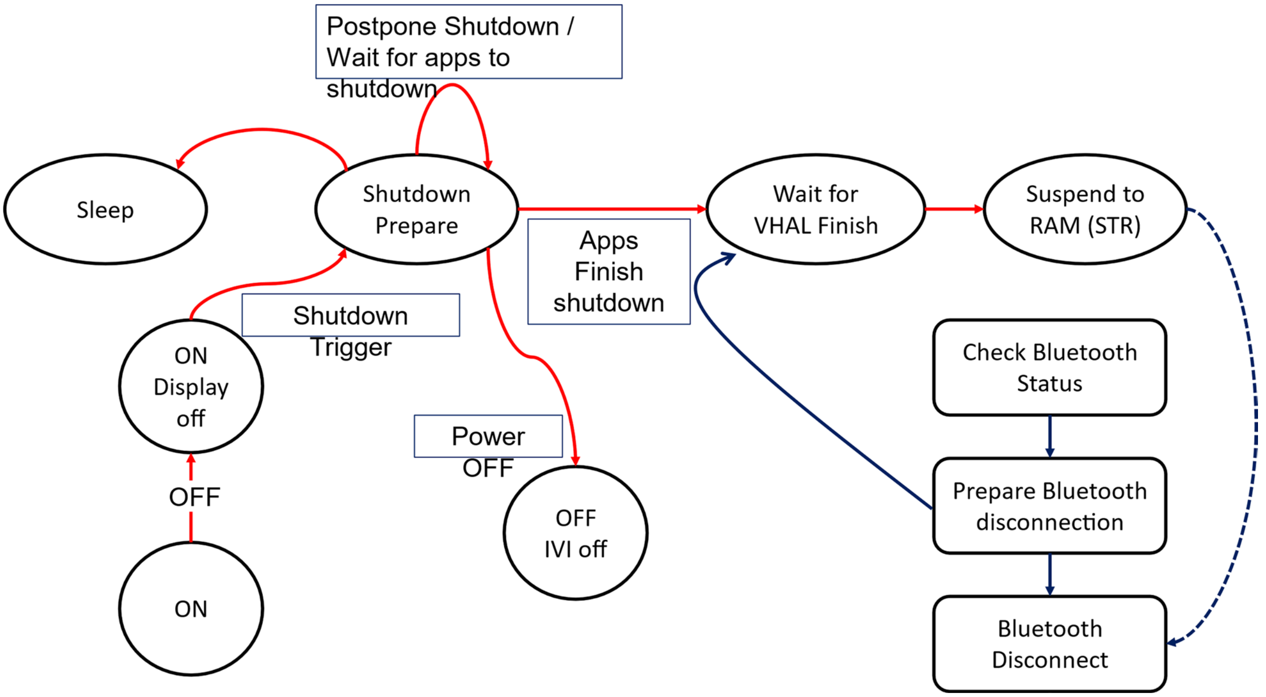


Fig. 9 Proposed state machine to mitigate the attack—clear the stored memory upon Bluetooth disconnection. The Suspend to RAM (STR) and Bluetooth Disconnection, shown by the dotted line, is a synchronized process, not an edge transition

Algorithm 1 to clear memory upon disconnection

```
carState ← CarPowerManager.CarPowerStateListener
BTAdapterState ← mBluetoothAdapter.getState
DeveloperMode ← Settings.Global DEVELOPMENT_SETTINGS_ENABLED
if carState is ON then
  enableBluetooth
  if BTAdapterState is ON and DeveloperMode is ENABLED then
    SendNotification
    connectDevices
  end if
endif
if carState is ShutdownPrepare then
  disableBluetooth
end if
StateChange ← BluetoothAdapter.ACTION_STATE_CHANGED
if StateChange is ON then
  ConnectDevices
else if StateChange is DISCONNECT_REQUESTED then
  disableBluetooth
  enableBluetooth
  connectDevices
end if
```

The importance of developer options in Android OS is highlighted in "Threat model and attack description" section. Hence, we made sure that the proposed countermeasure does not affect the working of developer options. By notifying the user about the potential hazard, the user could turn off the developer mode manually and get cautioned about a potential attacker in their vicinity. Bluetooth logs are not stored in the device and cannot be accessible to the attacker when the device is not in developer mode. The second countermeasure proposed clears the Bluetooth logs upon disconnection. Thus, when the user is not near the vehicle (within the Bluetooth range), the Bluetooth logs get cleared and become inaccessible to the attacker. Hence, in both cases, the user is protected from being a victim as the attacker does not have access to the Bluetooth logs. However, an actual developer could still have access to these logs during the testing phase of application development.

Limitations and scope of our work

The valet attack identifies two potential vulnerabilities and proposes a method to exploit the vulnerability. However, it is evident that the attack has a narrow scope in practical execution. The attack is based on the premise that the attacker has access to the vehicle and can alter the original status of the infotainment system. That is, the attacker can enable developer options and enable Bluetooth HCI snoop log. The attacker can access confidential and private information from the victim's device only when the infotainment unit is in the same network as the attacker or the attacker again has physical access to the vehicle. The attack is not possible if the vehicle ignition is turned off before the attacker can access the vehicle.

In this work, by coming up with an attack and countermeasures, we intend to show OEMs and vehicle users that their private and confidential information is susceptible. We strictly followed Engineering Ethics and did not conduct the attack on devices other than the designated testing ones. We do not disclose the vehicle's manufacturer and do not encourage trying out the attacks and accessing confidential information from unknown devices.

Responsible disclosure

We had responsibly reported the bug to the Android automotive team. We hope they will acknowledge the problem and work towards a feasible solution. We assure to work further and collaborate with them to develop defense mechanisms and countermeasures if needed.

Conclusions

Bluetooth is an essential wireless communication technology in automotive infotainment units. The HMI of the infotainment unit aids the driver with Hands-Free calling and texting features without being distracted while driving. With crucial applications in the automotive domain, the security of Bluetooth plays a vital role in protecting the user's personal information and privacy. In this paper, we briefly introduced the application of Bluetooth technology in the automotive infotainment unit and proposed an attack by exploiting a privacy vulnerability. We also described the systematic approach (STPA-Priv) we followed in eliciting and exploiting the vulnerability. Besides, we also proposed a potential defense solution to prevent the attack. We note that the cause for these attacks is based on assumptions about trusted devices and could be rectified through software updates on the Android Bluetooth stack.

Acknowledgements
This work was performed at the Center for Automotive Research at The Ohio State University.

Author contributions
All authors read and approved the final manuscript.

Funding
The work presented in this paper was self-funded.

Availability of data and materials
There are no public data or materials for this paper.

Declarations

Competing interests
The authors declare that they have no competing interests.

Received: 6 May 2022 Accepted: 6 September 2022
Published online: 04 October 2022

References

Android (2021a) Automotive: android open source project [Online]. Available: https://source.android.com/devices/automotive

Android (2021b) Bluetooth [Online]. Available: https://source.android.com/devices/bluetooth

Android (2021c) Automotive: android open source project—power management [Online]. Available: https://source.android.com/devices/automotive/power/power

Android (2021d) Automotive: android open source project—what is android automotive? [Online]. Available: https://source.android.com/devices/automotive/start/what_automotive.

Android (2021e) Configure on-device developer options [Online]. Available: https://developer.android.com/studio/debug/dev-options.

Antonioli D, Payer M (2022) On the insecurity of vehicles against protocol-level bluetooth threats. In 2022 IEEE Security and Privacy Workshops (SPW) (pp. 353-362). IEEE.

Antonioli D, Tippenhauer NO, Rasmussen KB (2019) The KNOB is broken: exploiting low entropy in the encryption key negotiation of bluetooth BR/EDR. In: 28th USENIX security symposium (USENIX security 19)

Antonioli D, Tippenhauer NO, Rasmussen K (2020) BIAS: bluetooth impersonation attacks. In: 2020 IEEE symposium on security and privacy (SP)

Antonioli D, Tippenhauer NO, Rasmussen K, Payer M (2022) BLURtooth: exploiting cross-transport key derivation in bluetooth classic and bluetooth low energy. In: Proceedings of the 2022 ACM on Asia conference on computer and communications security

Ballmann B (2021) Feeling bluetooth on the tooth. In: Understanding network hacks: attack and defense with Python 3, Springer Berlin, Heidelberg, p 139–162

Barth S, de Jong MDT, Junger M, Hartel PH, Roppelt JC (2019) Putting the privacy paradox to the test: Online privacy and security behaviors among users with technical knowledge, privacy awareness, and financial resources. Telematics Inform 41:55–69

Benton K, Camp LJ, Garg V (2013) Studying the effectiveness of android application permissions requests. In: 2013 IEEE international conference on pervasive computing and communications workshops (PERCOM Workshops)

Bhat A (2015) HMI Architecture and bluetooth phonebook design in car infotainment, vol. 2, p 257903

BluetoothSIG (2019a) Bluetooth qualification process overview [Online]. Available: https://www.bluetooth.com/develop-with-bluetooth/qualification-listing/

BluetoothSIG (2019b) Phone book access profile [Online]. Available: https://www.bluetooth.org/docman/handlers/downloaddoc.ashx?doc_id=457095

BluetoothSIG (2019c) Message access profile [Online]. Available: https://www.bluetooth.org/docman/handlers/downloaddoc.ashx?doc_id=457095

Boddie D (2017) PyOBEX. [Online]. Available: https://gitlab.com/dboddie/pyobex.

Cheah M, Shaikh SA, Bryans J, Nguyen HN (2016) Combining third party components securely in automotive systems. In: Information security theory and practice, Cham

Cheah M, Shaikh SA, Haas O, Ruddle A (2017) Towards a systematic security evaluation of the automotive Bluetooth interface. Veh Commun 9:8–18

Cheah M, Shaikh SA, Bryans J, Wooderson P (2018) Building an automotive security assurance case using systematic security evaluations. Comput Secur 77:360–379

Checkoway S, McCoy D, Kantor B, Anderson D, Shacham H, Savage S, Koscher K, Czeskis A, Roesner F, Kohno T (2011) Comprehensive experimental analyses of automotive attack surfaces. In: Proceedings of the 20th USENIX conference on security, USA

Claverie T, Teves JL (2021) BlueMirror: reflections on bluetooth pairing and provisioning protocols. In: 2021 IEEE security and privacy workshops (SPW)

Cope P, Campbell J, Hayajneh T (2017) An investigation of Bluetooth security vulnerabilities. In: 2017 IEEE 7th annual computing and communication workshop and conference (CCWC)

Costantino G, Matteucci I (2022) Reversing Kia motors head unit to discover and exploit software vulnerabilities, J Comput Virol Hacking Techniques

Costantino G, Vincenzi MD, Matteucci I (2022) A comparative analysis of UNECE WP. 29 R155 and ISO/SAE 21434. In: IEEE European symposium on security and privacy workshops (EuroS&PW), Genoa

Dardanelli A, Maggi F, Tanelli M, Zanero S, Savaresi SM, Kochanek R, Holz T (2013) A security layer for smartphone-to-vehicle communication over Bluetooth. IEEE Embed Syst Lett 5:34–37

Deuker A (2009) Addressing the privacy paradox by expanded privacy awareness–the example of context-aware services. In: IFIP PrimeLife international summer school on privacy and identity management for life

Dunning J (2010) Taming the blue beast: a survey of bluetooth based threats. IEEE Secur Priv 8:20–27

Egele M, Kruegel C, Kirda E, Vigna G (2011) PiOS: detecting privacy leaks in iOS applications. In: NDSS

Enck W, Gilbert P, Han S, Tendulkar V, Chun B-G, Cox LP, Jung J, McDaniel P, Sheth AN (2014) TaintDroid: an information-flow tracking system for realtime privacy monitoring on smartphones. ACM Trans Comput Syst, 32

Felt AP, Ha E, Egelman S, Haney A, Chin E, Wagner D (2012) Android permissions: user attention, comprehension, and behavior. In: Proceedings of the eighth symposium on usable privacy and security

Friedberg I, McLaughlin K, Smith P, Laverty D, Sezer S (2017) STPA-SafeSec: safety and security analysis for cyber-physical systems. J Inf Secur Appli 34(2):183–196

Garakani HG, Moshiri B, Safavi-Naeini S (2018) Cyber security challenges in autonomous vehicle: their impact on RF sensor and wireless technologies. In: 2018 18th international symposium on antenna technology and applied electromagnetics (ANTEM)

Gessler P. Muller T, Mailat M (2020) Android automotive OS whitepaper: android automotive OS book

Hassan SS, Bibon SD, Hossain MS, Atiquzzaman M (2018) Security threats in Bluetooth technology. Comput Secur 74:308–322

Honda (2021) Hondanews. [Online]. Available: https://hondanews.com/en-US/releases/honda-and-google-collaborate-on-in-vehicle-connected-services.

Hussain SU, Koushanfar F (2018) P3: Privacy preserving positioning for smart automotive systems. ACM Trans Des Autom Electron Syst (TODAES) 23:1–19

Kaplun V, Segal M (2019) Breaching the privacy of connected vehicles network. Telecommun Syst 70:541–555

Kaur G, Jain B (2013) Data communication via bluetooth-a trusted device, Atharva, p 4

Kumaraguru P, Cranor LF (2005) Privacy indexes: a survey of Westin's studies. Carnegie Mellon University, School of Computer Science, US

Lu K, Li Z, Kemerlis VP, Wu Z, Lu L, Zheng C, Qian Z, Lee W, Jiang G (2015) Checking more and alerting less: Detecting privacy leakages via enhanced data-flow analysis and peer voting. In: 22nd annual network and distributed system security symposium, NDSS 2015, San Diego, California, USA, February 8–11, 2014

Megowan P, Suvak D, Kogan D (2003) Object exchange protocol. [Online]. Available: https://www.irda.org/standards/pubs/OBEX13.pdf

Mell P, Scarfone K, Romanosky S (2006) Common vulnerability scoring system. IEEE Secur Priv 4:85–89

Mindermann K, Riedel F, Abdulkhaleq A. Stach C, Wagner S (2017) Exploratory study of the privacy extension for system theoretic process analysis (STPA-Priv) to elicit privacy risks in eHealth. In: 2017 IEEE 25th International requirements engineering conference workshops (REW)

Nasim R (2012) Security threats analysis in Bluetooth-enabled mobile devices. Preprint http://arxiv.org/abs/1206.1482

nOBEX (2016) nOBEX. [Online]. Available: https://github.com/nccgroup/nOBEX.

Oka DK, Furue T, Langenhop L, Nishimura T (2014) Survey of vehicle IoT Bluetooth devices. In: 2014 IEEE 7th international conference on service-oriented computing and applications

Onishi H, Wu K, Yoshida K, Kato T (2017) Approaches for vehicle cyber-security in the US: vulnerabilities of carry-in devices, GNSS, & vehicle-to-vehicle communication. Int J Automot Eng 8:1–6

PK (2019) Challenges in android based in-vehicle- infotainment (IVI). [Online]. Available: https://medium.com/@pkurumbudel/challenges-in-android-based-in-vehicle-infotainment-ivi-93819acc650a

Popa B (2021) Autoevolution. [Online]. Available: https://www.autoevolution.com/news/heres-the-full-list-of-cars-powered-by-android-automotive-169169.html

Shapiro SS (2016) Privacy risk analysis based on system control structures: adapting system-theoretic process analysis for privacy Engineering. In: 2016 IEEE security and privacy workshops (SPW)

Spiekermann S, Acquisti A, Böhme R, Hui K-L (2015) The challenges of personal data markets and privacy. Electron Mark 25:161–167

Sutton A (2021) Android automotive OS 11 on a raspberry Pi. [Online]. Available: https://medium.com/snapp-automotive/android-automotive-os-11-on-a-raspberry-pi-2abaa133f468.

Tanaka T, Skoglund M, Sandberg H, Johansson KH (2017) Directed information and privacy loss in cloud-based control. In: 2017 American control conference (ACC)

von Tschirschnitz M, Peuckert L, Franzen F, Grossklags J (2021) Method confusion attack on bluetooth pairing. In: 2021 IEEE symposium on security and privacy (SP)

Wuyts K, Scandariato R, Joosen W (2014) LIND (D) UN privacy threat tree catalog, vol. 675, Department of Computer Science, KU Leuven

Yadav A, Bose G, Bhange R, Kapoor K, Iyenger NCSN, Caytiles R (2016) Security, vulnerability and protection of vehicular on-board diagnostics. Int J Secur Appl 10:405–422

Yee K-P (2002) User interaction design for secure systems. In: Information and communications security, Berlin

Young W, Leveson N (2013) Systems thinking for safety and security. In: Association for computing machinery, New York

Zelle D, Krauss D, von Pape T (2017) A privacy-aware data access system for automotive applications. In: 15th ESCAR embedded security in cars conference

Zhang Y, Li J, Zheng D, Li P, Tian Y (2018) Privacy-preserving communication and power injection over vehicle networks and 5G smart grid slice. J Netw Comput Appl 122:50–60

Zhou X (2014) The security and privacy of mobile platforms in a rapidly evolving world

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.