International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. 16, No.1 (Jul-24)

http://dx.doi.org/10.12785/ijcds/160112

A Unique Approach for Ex-filtering Sensitive Data Through an Audio Covert Channel in Android Application

Abhinav K. Shah[1], Dr. Digvijaysinh M. Rathod[2], Bharat Buddhadev[3] and Jeet Rami[4]

[1,2]*School of Cyber Security & Digital Forensics, National Forensic Sciences University, Gandhinagar, India*
[3]*HoD Computer and IT, Government Of Gujarat, Ahmedabad, India*
[4]*Threat Analyst, Microsoft, Hyderabad, India*

Received 22 Jun. 2023, Revised 7 Apr. 2024, Accepted 16 Apr. 2024, Published 1 Jul. 2024

Abstract: The rapid global evolution of technology has led to a significant shift in the field of cybersecurity. The advent of technologies such as Artificial Intelligence, Machine Learning, Blockchain, and Data Science has significantly increased the complexity and demand for research in the field of cybersecurity. Numerous authors and academics consistently engage in endeavors within this field to enhance existing implementations. Due to this process of evolution, several entities have experienced growth, leading to the transformation of numerous small-scale organizations into large-scale counterparts. The advent of the internet has resulted in significant growth for companies, albeit with divergent outcomes. Researchers are utilizing the internet to advance technological developments, while concurrently, certain individuals persistently seek means to undermine this technology. In conjunction with the proliferation of the internet, mobile phones have acquired a heightened level of sophistication, particularly with the advent of the Android operating system, leading to a shift in the manner in which individuals utilize these devices. As the Android operating system continues to advance, individuals are increasingly embracing and incorporating its features into their daily lives. Nevertheless, there are those that persistently endeavor to discover methods for illicitly extracting information from Android smartphones via a covert communication channel. This work introduces a novel methodology for transmitting confidential data, such as contact information and text messages, utilizing a concealed audio channel. The researchers conducted trials utilizing the most recent Android smartphones and arrived at the conclusion that it is feasible to transmit critical information to Android using this methodology. The experiment was conducted by the authors using various detection techniques, and it was determined that none of these tools are capable of detecting a hidden channel of this nature.

Keywords: Covert Channel, Android Application, Cyber Security, Application Security, Network Security, Privacy

1. Introduction

Technology has been growing rapidly across the globe with the evolution of the internet. Over the past few decades, several new technologies have been introduced, including Artificial Intelligence [1], Machine Learning, IoT, Blockchain, and Chatbots [2]. Cybersecurity has been a significant area of development and research in recent years. According to Nikita's [3] article, recent trends in cybersecurity include automated hacking, the use of AI in cybersecurity, cloud security, IoT security [4], insider threats, and cyber warfare. There has been an increase in cybersecurity attacks, with over 2,000 attacks happening every day, costing around $9.44 million, and estimated to increase to $8 trillion by 2023, as reported by Nivedita's [5] analysis. These attacks include DDOS [6], Phishing, Ransomware, MITM, Zero Day Exploits, Spoofing, and more [7]. After an attack, a forensic team is typically reached to investigate, including digital forensics, IoT forensics [8], SCADA forensic, and application forensic. Covert channel attacks have also been increasing, with trends including encrypted communication, system vulnerability exploitation, AI use, new technology adoption, and social engineering emphasis.

A covert channel attack is a cybersecurity threat in which two parties communicate and exchange information without detection by traditional security measures [9]. The communication is designed to be hidden, making it difficult to detect and stop. Covert channel attacks can be used for various purposes, which include stealing sensitive information, bypassing security controls, or executing malicious code. Covert channel attacks can be perpetrated by insiders, cyber criminals, or state-sponsored actors, and their targets may include a diverse range of entities, such as government agencies, businesses, and individuals. The first introduction to covert channel happened in 1973 by Lampson [10], the author used confinement problems to explain covert channels. In Android, covert channels can be created using shared resources, system calls, network interfaces, and interprocess communication methods [11]. Covert channels

E-mail address: abhinavk.shah@gmail.com, digvijay.rathod@nfsu.ac.in, bvbld@yahoo.com, jeetrami39@gmail.com

can serve multiple purposes, such as extracting sensitive information from a user's device, propagating malware, or executing illicit actions on the device. The complexity of the Android ecosystem and the wide variety of third-party applications on the market make it difficult to find and stop hidden channels in Android applications. Access control, sandboxing, static and dynamic analysis, and other methods can help to lessen the danger of covert channels in Android applications [12].

According to research conducted by Goher [13], covert channels can be classified into two distinct categories. The two categories are timing-based covert channels and storage-based covert channels. The paper concludes by examining various techniques utilized in timing-based and storage-based covert channels. Timing-based channels convey information by manipulating some aspect of the system's behavior over time, enabling the recipient program to observe the system's activity and infer sensitive information [14]. Storage-based covert channels refer to the storage of out-of-band data within messages for efficient memory usage. However, if these messages or packets are transmitted with residual data, malevolent listeners may become aware of the message's origin. This awareness could allow attackers to obtain important information such as the sender's hardware platform, operating system, or algorithms, which could then be leveraged to launch further attacks [15].

One real-world case study related to covert channel attacks in Android is the ”Triada” malware, which was discovered by the security firm Kaspersky [16] in 2016. The Triada malware used a covert channel to communicate with its command-and-control server without being detected by traditional security measures. The malware injected itself into the Android operating system's Zygote process, which runs when an application is launched, and then used the shared memory of the process to communicate with the server. By using this covert channel, the Triada malware was able to receive commands and send sensitive information without being detected by antivirus software or firewalls. The malware also could modify system libraries, allowing it to perform various malicious activities, including installing additional malware and stealing user data. The discovery of the Triada malware highlights the need for mobile security measures that can detect and prevent covert channel attacks in Android.

A Check Point study [17] discussed the ”Judy” malware, which utilized a covert channel to communicate with its command-and-control server without detection. The malware was distributed through various Google Play Store apps that contained malicious code to generate clicks on ads, generating revenue for the creators. The malware leveraged the covert channel to receive updated instructions and evade detection by conventional security measures. The covert channel was hidden within a harmless-looking function in the malware's code, making it challenging to detect. This malware infected millions of Android devices before detection, demonstrating the dangers of covert channel attacks on mobile devices.

The incidents described above highlight the need for improved security measures within the Android ecosystem, such as better app vetting processes and more effective tools for detecting and preventing covert channel attacks. The paper under discussion proposes a new approach for stealing sensitive information by establishing an audio covert channel within Android applications. Section 2 of the paper discusses the impact of covert channel attacks and compares various covert channel detection techniques. Section 3 outlines the methodology for encoding and decoding information within the covert channel, while Section 4 presents the experimental results. Finally, in Section 5, the authors provide a conclusion for their research, and the paper concludes with a reference section.

2. Literature Review

In this section, the authors have summarized different covert channel attacks that have been conducted by various researchers in the past, along with the different detection tools that are widely used. One study conducted by Chandra [18] reveals that covert channel attacks pose a significant risk to smartphone security by exploiting hidden communication channels between different components and software layers within the device. These attacks can bypass traditional security measures, resulting in unauthorized data leakage or unauthorized access to sensitive information. This has serious implications for user privacy, confidentiality, and overall system integrity. The study identifies two new resources, namely the battery and phone calls, that can be used to create covert channels in Android. Additionally, the author examines covert channels created at three different levels: application, operating system, and hardware. The experiment concludes that a throughput of 34.17 kb was achieved with this covert channel.

Another research conducted by Al-Haiqi [19] presents a novel and concerning covert channel attack that exploits the sensors present in Android smartphones. This study highlights the potential risks associated with this covert communication channel and emphasizes the need for increased awareness and countermeasures to mitigate such attacks. The paper provides a detailed analysis of the new sensor-based covert channel, demonstrating how it leverages data from various sensors (such as accelerometers, gyroscopes, magnetometers, and ambient light sensors) embedded in Android smartphones to establish hidden communication channels between malicious applications. By utilizing sensor data, the covert channel evades traditional security mechanisms and operates without detection. The authors conclude that this approach is stealthy and can transmit sensitive information without requiring explicit permissions.

In his study, Casolare [20] extensively investigated colluding covert channel attacks, where multiple applications collaborate in secret to covertly transmit data. These attacks exploit interprocess communication mechanisms, shared resources, and permissions within the Android environment, allowing these colluding applications to establish hidden channels that evade traditional security measures, enabling the unauthorized extraction of information. The research demonstrates the feasibility of such attacks and underscores their potential impact on both user privacy and system security. The author developed a series of applications that take advantage of covert channels using the accelerometer, facilitating collusion among Android applications. The research concludes by noting that this experiment involved testing with more than 60 anti-malware tools, none of which were capable of detecting this phenomenon. Li [21], in a separate study, thoroughly examines covert channels within the Android operating system, shedding light on their characteristics, vulnerabilities, and potential consequences. The study emphasizes the necessity of gaining a deeper understanding of covert channels and the development of effective countermeasures to mitigate their associated risks.

Covert channels can be established with the assistance of magnetic signals generated by Android devices. To address this, Guri [22] introduces a novel and concerning covert channel attack in his research, which utilizes CPU-generated magnetic fields to establish communication between air-gapped systems and nearby smartphones. The study highlights the potential risks associated with this covert channel and emphasizes the need for enhanced security measures to mitigate such attacks. The paper provides a comprehensive analysis of the Magneto covert channel, which exploits the magnetic fields generated by the CPU activity of an air-gapped system. By utilizing the magnetometer sensor present in smartphones, the covert channel enables data transmission from the air-gapped system to the smartphone, bypassing conventional network-based communication channels. The author demonstrates the feasibility of this attack by successfully showcasing data transmission between air-gapped systems and smartphones using the Magneto covert channel. It underscores the potential security implications, as this covert communication can be exploited to exfiltrate sensitive information from highly secured environments without detection. The author concludes that this approach achieves a throughput of 5 bits per second in transmission.

In another research, Liang [23] presents a detailed exploration of covert channels in mobile Voice over Internet Protocol (VoIP) traffic by utilizing variations in packet lengths. The study investigates the feasibility and potential implications of establishing covert communication channels within VoIP traffic. By exploiting the inherent characteristics of packet lengths, the covert channel enables the transmission of hidden information, bypassing traditional detection mechanisms and posing potential risks to user privacy and data security. The insights provided by this study can guide the development of more robust security frameworks and aid in the prevention and detection of covert channel activities within mobile VoIP networks. After performing the experiments, the author concludes that a throughput of 41.78 - 148.13 bits/second can be achieved with this method. Liang [24] in a similar research introduces a novel covert channel that exploits payload-dependent packet rearrangement techniques within mobile Voice over Internet Protocol (VoIP) traffic. The paper presents a comprehensive analysis of the payload-dependent packet rearranging covert channel, which involves manipulating the order of packets based on the payload content within mobile VoIP traffic. By rearranging packets in a predetermined manner, hidden information can be transmitted covertly, evading conventional detection mechanisms and potentially compromising user privacy and data security.

Lalande [25], in his study, addresses the critical issue of privacy leaks in Android applications and explores the concept of low-attention raising covert channels as a means to conceal such leaks. Through this research, the authors demonstrate the feasibility of hiding privacy leaks using low-attention-raising covert channels and examine the implications of these covert communication channels on user privacy. The author conducted experiments using four different covert channels and concluded that these channels can easily bypass detection mechanisms such as TaintDroid. Zhang [26] in his research explores the concept of a covert channel established over Voice over LTE (VoLTE) networks by manipulating silence periods. By strategically modifying the duration of silence periods during voice calls, hidden information can be transmitted, bypassing traditional detection mechanisms and potentially compromising network security. Through their research, the authors demonstrate the practicality and effectiveness of this covert channel in VoLTE networks. They highlight the challenges associated with detecting such covert communication, as it operates within the normal parameters of voice call silence periods and does not raise immediate suspicion.

A new investigation conducted by Shah [27] introduced a method for establishing covert communication channels and transmitting confidential data among users via an audio file. This method is crafted to evade detection by current security measures, mainly because of the intricate code scrutiny necessary for detection. The researcher suggests that this technique offers a secure means for exchanging sensitive information while underscoring the necessity for enhanced security tools capable of identifying such covert channels.

In another research, Deshotels [28] introduces two covert channels based on sound, namely ultrasonic and isolated sound. The author suggests that ultrasonic sound can be received by either the device's microphone or another device. Deshotels implemented an ultrasonic modem specifically for Android and found that it can transmit signals up to a distance of 100 feet. Additionally, the author mentions that these vibrations can be detected by the device's accelerometer, although they are not loud enough for humans to perceive. If performed while the user is not actively holding the device, these vibrations will likely go unnoticed. Ultimately, the author concludes that this technique can easily circumvent numerous flow control mechanisms. Abdullaziz [29], in his research, presents a novel approach to steganography using network packet payloads and the concept of payload parity. The study explores the feasibility and effectiveness of this steganographic technique, highlighting its potential applications and implications for network security. By exploiting the redundancy in payload data and utilizing the parity bit as a carrier, this covert communication technique allows for the covert transmission of information without raising suspicion. Through this research, the author demonstrates the practicality and effectiveness of network packet payload parity-based steganography. In another study, Yang [30] presents a novel and worrisome covert channel called NICScatter, which leverages backscatter communication to establish hidden communication channels in mobile devices. NICScatter takes advantage of ambient electromagnetic signals emitted by nearby devices or infrastructure, eliminating the requirement for conventional network connections or specialized hardware. The author's research concludes with experiments conducted on various mobile devices, demonstrating that the covert channel can achieve a transmission rate of 1.6 bps and cover distances of up to 2 meters. In a scenario involving transmission through a wall, it can reach distances of up to 70 cm.

In the next section, the author presents the impact of covert channel attacks in the real world.

A. Impact of Covert Channel Attacks

Covert channel attacks can have significant impacts in the real world, posing threats to various systems and organizations. Here are some potential impacts of covert channel attacks: Data Exfiltration, Privacy Breaches, Coordinated Attacks, Espionage and Intelligence Gathering, System Compromise and Control, Trust and Reputation Damage, and Regulatory Compliance Violations. To mitigate the impact of covert channel attacks, organizations need to implement robust security measures, including intrusion detection systems, anomaly detection techniques, encryption protocols, and regular security assessments. In Table I, the author presents real-world instances of covert channel attacks that have had notable impacts.

B. Covert Channel Detection Tools Review

In this section, the author discusses different detection tools and frameworks implemented for identifying covert channel attacks. A study conducted by Wang [34] introduces a fresh method for estimating the bandwidth of intent-based covert channels on the Android platform using time series decomposition analysis. The study examines the viability and efficacy of this technique, as well as its potential consequences for detecting and mitigating covert channels. In this approach, the author leverages the patterns of API activities and compares their relationship with the concealed covert channel signal to estimate the bandwidth of the covert channel. The author employs series decomposition and sequence analysis methods as part of this approach. In another research conducted by Qiang [35], a comprehensive framework is introduced with the purpose of auditing and detecting covert communication channels specifically on Android devices. The study addresses the crucial necessity for effective methods to identify and mitigate covert channels, which can potentially be exploited for malicious intents or result in information leakage. Through this research, the authors successfully demonstrate the feasibility and efficacy of DroidAuditor in detecting covert communication channels. The framework utilizes a range of techniques, including static analysis, dynamic analysis, and traffic analysis, to uncover hidden communication patterns, abnormal behaviors, or suspicious activities that may indicate the presence of covert channels. The author concludes that the experimental results indicate the effectiveness of the audit mechanism in reducing the capacity of target covert channels, while the associated process time and resource consumption are minimal. However, it should be noted that this framework is not compatible with Android versions greater than 4.

In another research conducted by Arzt [36], a novel and remarkably accurate static taint analysis tool called FlowDroid is presented for analyzing Android applications. Through this research, the authors successfully demonstrate the practicality and effectiveness of Flowdroid in conducting context-sensitive, flow-sensitive, field-sensitive, object-sensitive, and lifecycle-aware taint analysis for Android apps. The tool efficiently traces the flow of sensitive data within an application, taking into account various aspects of the program's execution and lifecycle. The author conducted experiments on approximately 500 applications sourced from the Google Play Store and 1000 applications from the Virus Share project. The results revealed that FlowDroid successfully detected information leaks in those applications.

Graa [12] in her research, presents a comprehensive approach that combines static and dynamic analysis techniques to detect control flow on smartphones. The study addresses the necessity for effective methods to identify and analyze control flow patterns within mobile applications. Such analysis can help uncover potential security vulnerabilities and enhance overall security measures. Through this research, the author successfully demonstrates the practicality and effectiveness of the combined static and dynamic analysis approach. By examining control flow patterns, the framework can detect anomalies, suspicious behaviors, or potential code execution vulnerabilities that may result in security breaches or malicious activities.

Shifting focus to dynamic analysis, Enck [37] introduces an innovative information-flow tracking system called Taintdroid. This study addresses the critical requirement for privacy monitoring on smartphones and presents a groundbreaking solution for tracking the flow of sensitive informa-

TABLE I. Impact of Covert Channel Attacks

Sr.No.	Year	Attack Parameter	Impact
1	2023	Radio Waves [31]	Sensitive information is leaked from air-gapped computers via radio waves generated by malware.
2	2022	Speaker and Microphone [32]	Using an ultrasound covert channel, leakage of sensitive information is possible.
3	2021	Android Permissions and Accelerometer Sensor [20]	Sensitive data such as SMS, email, contacts, IMEI, and calendar events were effortlessly transferred using sensor technology.
4	2021	Magnetometers [22]	By utilizing magnetic sensors, it becomes feasible to extract information in environments without network connectivity.
5	2018	VoIP Traffic [23] [24]	A covert communication channel was established by associating data symbols with packet lengths and transmitting information through the manipulation of packet sending order.
6	2017	USB Charging Station [33]	The adversary, who had control over the public charging station, was able to intercept and acquire sensitive data like the IMEI number, contacts, and pictures.
7	2014	Ultrasonic Sound and Vibrations [28]	A covert channel was established, enabling the transmission of messages between devices without requiring explicit permission.

tion in real-time. The author demonstrates the practicality and effectiveness of Taintdroid in monitoring privacy on smartphones in real-time. By dynamically tracking the flow of sensitive data, the system can identify possible privacy violations, unauthorized data leaks, or insecure practices in handling information by applications. Additionally, the paper explores the implications and potential applications of Taintdroid. By providing real-time privacy monitoring, Taintdroid empowers users and organizations to make informed decisions about their data privacy, detect privacy breaches, and enhance overall smartphone security.

Buddhadev [38] introduces a fresh approach known as FloVasion for detecting evasive information flow based on non-sensitive variables in Android applications. The study addresses the challenges associated with identifying evasive techniques employed by malicious actors to conceal their activities and evade detection. Through this research, the authors successfully demonstrate the effectiveness and practicality of FloVasion in detecting evasive information flow. The approach analyzes the behavior of non-sensitive variables, such as control flow manipulations and data dependencies, to uncover concealed communication channels or malicious activities that may elude traditional detection mechanisms. Furthermore, the paper explores the implications and potential applications of FloVasion. By enhancing the detection of evasive information flow, this approach enables improved app security, better identification of potential privacy violations or data leakage, and enhanced protection against malicious activities. The author conducted experiments using various tools such as FlowDroid, TaintDroid, DroidSafe, DroidAuditor, and other commercial tools, and concluded that FloVasion outperforms other tools in terms of detecting evasion-based information flow.

Continuing within the paper, Shah [39] provides a summary of various covert channel detection tools in their research. The authors evaluate and compare the performance of multiple detection tools, considering factors like detection accuracy, false positives, false negatives, and overall capability. This evaluation serves to comprehend the strengths and limitations of each tool, providing valuable insights into their effectiveness in detecting covert channels.

In a separate study conducted by Kar [40], a detection method is suggested which combines static detection with the analysis of activity behavior. Their approach to detection improves the security of Android devices by tackling overlay vulnerabilities and their potential risks to user privacy and data security.

3. Limitations In Existing Research

In the preceding section, the authors have delineated the diverse covert channel attacks conducted by numerous researchers, as well as the detection methodologies and tools devised in previous studies. The author has provided an analysis of the constraints inherent in the current body of research. In the realm of Android applications, the acquisition of certain rights is vital. These permissions encompass a range of functionalities, including but not limited to internet access, contact management, external storage writing capabilities, camera usage, and various others. Numerous scholars have previously conducted studies on the subject of permits. Nevertheless, due to the unrestricted nature of this particular space, it is possible to create numerous clandestine communication pathways by leveraging the rights associated with a given program. For example, let us assume a chat application where it becomes apparent that the utilization of this program necessitates the acquisition of Internet and Contact permissions. The Android API additionally provides the capability for retrieving the phone status without requiring any additional permissions. Thus, utilizing this technique enables the establishment of a hidden route.

Covert channels can be established at the hardware level in Android devices. The alarm application is designed to emit a sound at a specific moment in time. Upon the activation of the alarm, the vibrator mechanism within the gadget is triggered. Once the vibration is initiated, it becomes apparent that signals will be transmitted. A viable approach involves situating a sensor in close proximity to the device, so that when vibrations are emitted, the adjacent sensor can effectively intercept and capture these signals. By implementing this configuration, the establishment of a covert communication channel can be achieved with relative ease.

When examining the constraints of the detection tools, it is often seen that the outcomes frequently consist of false positives. Researchers have undertaken and continue to engage in efforts to enhance the precision of detection. One of the main factors contributing to this issue is the limited capability of various instruments to detect and analyze minute portions of data that are transmitted through covert channels. In the context of data transmission, it is seen that in a given packet, a message of size ranging from 2 to 5 kilobytes is conveyed. It is noteworthy that certain tools tend to overlook these smaller packets, deeming them as valid. Consequently, these diminutive data fragments amalgamate to construct a more substantial message. In order to prevent this, further investigation into the methods of detection will be necessary.

4. Methodology

In this section, the author presents various methods for encoding and decoding sensitive information covertly within an Android application. To accomplish this, the author utilizes the Base64 algorithm [41] for encoding and decoding. This algorithm converts binary data into a text format that is suitable for transmission over text-based protocols like HTTP. It employs a set of 64 characters, typically including uppercase and lowercase letters, numbers, and two additional symbols. Base64 encoding is commonly employed to encode binary data, such as images or audio files, to ensure secure transmission and storage. By representing binary data as an ASCII string, it ensures compatibility with text-based systems and safeguards against data corruption during transmission. The author adopts the Base64 algorithm multiple times within this approach to implement this methodology.

A. Pseudocode for Encoding Sensitive Information from Sender to Receiver

In this section, the author presents Algorithm 1 that illustrates the encoding process of sensitive information, enabling its covert transmission from a sender to a receiver.

Algorithm 1 Encode Data and Send to Database

```
1: Start
2: Read Device ID, Contact Number, and SMS from the Android device.
3: Convert Device ID, Contact Number, and SMS to Binary respectively.
4: while not all octets for Device ID are converted into Base64 do
5:     for each octet in Device ID do
6:         Encode the octet into Base64.
7:         Combine the encoded octet with others for Device ID.
8:     Encode the combined Device ID with Base64.
9: while not all octets for Contact Number are converted into Base64 do
10:    for each octet in Contact Number do
11:        Encode the octet into Base64.
12:        Combine the encoded octet with others for Contact Number.
13:    Encode the combined Contact Number with Base64.
14: while not all octets for SMS are converted into Base64 do
15:    for each octet in SMS do
16:        Encode the octet into Base64.
17:        Combine the encoded octet with others for SMS.
18:    Encode the combined SMS with Base64.
19: Combine all the octets of Device ID, Contact Number, and SMS.
20: Encode the above combination with an audio file using Base64.
21: Send the encoded audio file to the Database.
22: End.
```

The provided algorithm begins by retrieving the Device ID, Contact Number, and SMS from an Android device. Each of these details is then individually converted into binary format. Subsequently, a loop is executed to encode each octet of the Device ID using the Base64 algorithm. Upon completion of the loop, all octets are encoded using Base64. The process is repeated for encoding the Contact Number and SMS. Once the encoded data for the Device ID, Contact Number, and SMS is obtained, it is combined with an audio file. The combined data is then encoded again using Base64 before being stored in the database. Figure 1 describes the flow for encoding information with the Base64 algorithm.

B. Pseudocode for Decoding Sensitive Information from Receiver to Sender

In this section, the author presents Algorithm 2 that outlines the decoding process of sensitive information received covertly by the receiver from the sender.

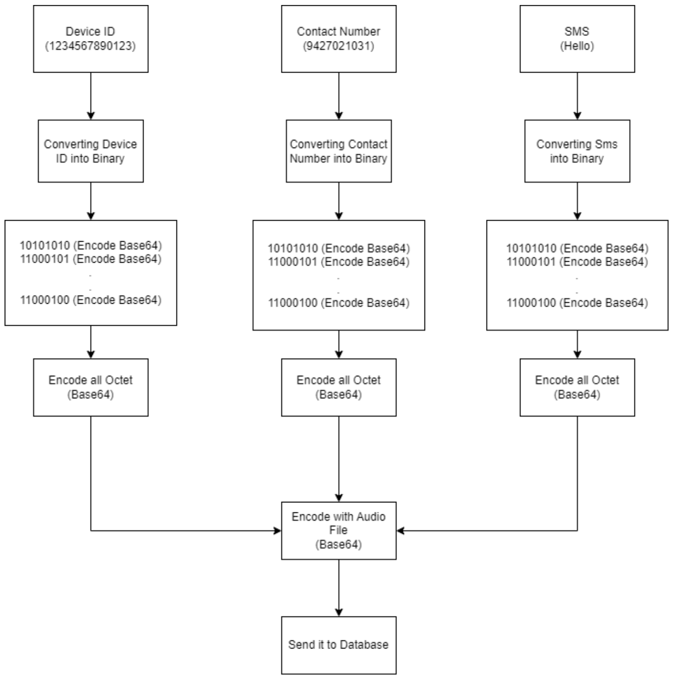
Device ID (1234567890123)
Contact Number (9427021031)
SMS (Hello)
Converting Device ID into Binary
Converting Contact Number into Binary
Converting Sms into Binary
10101010 (Encode Base64)
11000101 (Encode Base64)
11000100 (Encode Base64)
Encode all Octet (Base64)
Encode with Audio File (Base64)
Send it to Database

Figure 1. Encoding Information With Base64 Algorithm

The provided algorithm begins by fetching the audio file from the database. Once retrieved, the file is decoded using the Base64 algorithm. This decoding process yields three distinct octets. These octets are further decoded using the Base64 algorithm. A loop is executed to decode each octet of the Device ID using the Base64 algorithm. The same process is applied to decode the Contact Number and SMS. Once the data is decoded, the Device ID, Contact Number, and SMS are converted from binary to text format. The resulting text data is then displayed to the user. Figure 2 describes the flow for decoding information with the Base64 algorithm.

5. Experiments And Results

This section provides an overview of the conducted experiments and the subsequent analysis of the obtained results. The research paper introduces an Android Application developed by the author, which facilitates file sharing between users. Through this application, the author demonstrates the establishment of an audio covert channel, enabling the transmission of sensitive information such as Device ID, Contact Number, and SMS from the sender to the receiver without detection. In the subsequent subsection, the author discusses the different tools and technologies utilized during the application's development, showcasing how information flow can be achieved using various audio formats. Additionally, experiments were conducted on different versions of the Android operating system, and a comparison of the results was presented. Furthermore, the author employed several detection tools to identify the covert channel, and a detailed analysis of these findings was provided.

Algorithm 2 Decode Data and Display Information

```
1: Read information from Database and Decode information with Base64
2: Split all the octets of Device Id, Contact Number, and SMS
3: while all the octets for Device Id are not decoded with Base64 do
4:     for each octet in Device Id do
5:         Decode the octet with Base64
6: Split all the octets of Device Id which are now decoded with Base64
7: for each octet in Device Id do
8:     Decode the octet with Base64 and get Binary
9: while all the octets for Contact Number are not decoded with Base64 do
10:    for each octet in Contact Number do
11:        Decode the octet with Base64
12: Split all the octets of Contact Number which are now decoded with Base64
13: for each octet in Contact Number do
14:    Decode the octet with Base64 and get Binary
15: while all the octets for SMS are not decoded with Base64 do
16:    for each octet in SMS do
17:        Decode the octet with Base64
18: Split all the octets of SMS which are now decoded with Base64
19: for each octet in SMS do
20:    Decode the octet with Base64 and get Binary
21: Convert Device ID binary, Contact Number binary, and SMS binary to text respectively
22: Display Device ID, Contact Number, and SMS respectively
```

The experimentation process commences with the sender logging into the application. Upon successful login, the user proceeds to share an audio file with another user. Through this interaction, an audio covert channel is established, enabling the transmission of sensitive information such as Device ID, Contact Number, and SMS alongside the audio file. The sensitive information is encoded using the algorithm discussed in the preceding section. On the receiving end, the receiver logs into the application and proceeds to download the file. Once the file is downloaded, the sensitive information is decoded, and the relevant data is presented to the receiver. The decoding method for sensitive information aligns with the algorithm outlined in the previous section.

A. *Tools and Technologies Used*

The author utilized Android Studio as the primary development tool and incorporated Firebase as the chosen data storage solution for the Android application. The application itself was programmed using the JAVA programming language. In terms of testing, the OnePlus 3T Android phone was selected as the main device, which operated on

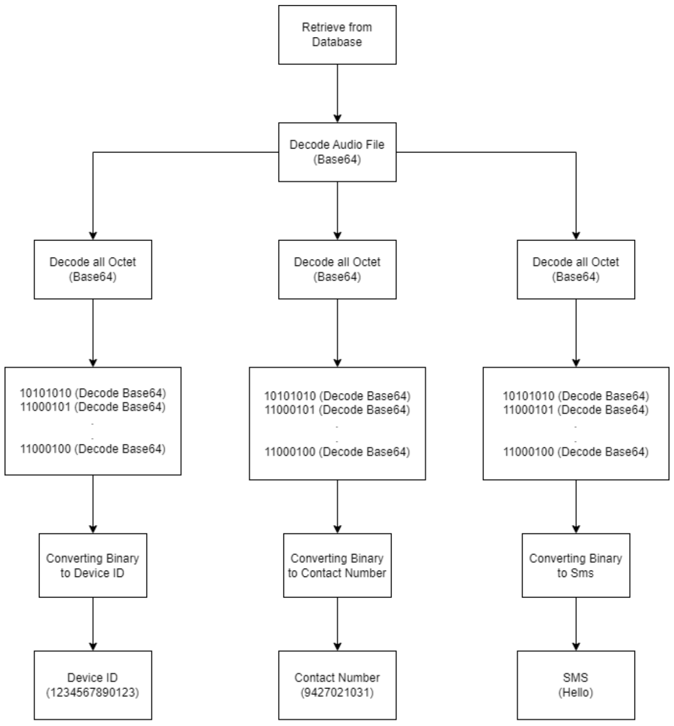
Retrieve from Database
Decode Audio File (Base64)
Decode all Octet (Base64)
Decode all Octet (Base64)
Decode all Octet (Base64)
10101010 (Decode Base64) 11000101 (Decode Base64) . . 11000100 (Decode Base64)
10101010 (Decode Base64) 11000101 (Decode Base64) . . 11000100 (Decode Base64)
10101010 (Decode Base64) 11000101 (Decode Base64) . . 11000100 (Decode Base64)
Converting Binary to Device ID
Converting Binary to Contact Number
Converting Binary to Sms
Device ID (1234567890123)
Contact Number (9427021031)
SMS (Hello)

Figure 2. Decoding Information with Base64 Algorithm

Android Version 9 and featured a Snapdragon 821 CPU with 6 GB of RAM. Additionally, a Dell Latitude 6430u Laptop with 16GB RAM and an i7 3rd Generation CPU were employed for testing purposes. Furthermore, various other Android devices were also utilized during the testing phase. A comprehensive overview of the tools, technologies, and devices employed throughout the development process can be found in Table II.

B. Information Flow with Different Audio Formats

In this section, the author provides a detailed account of the conducted experiments involving various audio file formats, along with the corresponding analysis. During the development of the application, the author ensured support for multiple file formats, which are outlined in Table III.

C. Experimenting on Different Versions of Android

In this section, the author presents an overview of the experiments conducted on various Android devices and provides an analysis of the results obtained in Table IV.

D. Execution of Different Detection Tools

In this section, the author conducts experiments to explore the effectiveness of various detection tools in identifying the covert channel. Both static and dynamic analysis techniques are employed during these experiments. The obtained results from the different detection tools are compared and presented in Table V, offering a comprehensive evaluation of their performance.

TABLE II. Tools, Technology and Testing Device Used in the Experiment

Sr.No.	Tools / Technology / Device	Description
1	Tools	Android Studio, Firebase
2	Programming Language	Java
3	Testing Device	Device:OnePlus3T Android: 9 Snapdragon 821 RAM: 6GB Device:Nexus5X Android: 11 Snapdragon 808 RAM: 4GB Device:VivoY33T Android: 12 Snapdragon 680 RAM: 12GB Laptop: Dell Latitude 6430u i7 3rd Generation RAM: 16GB

TABLE III. Supported File Formats

Sr.No.	Supported File Formats
1	.MP3, .Wav, .Ogg

6. Conclusion

A covert channel refers to a method employed for the illicit transmission of data or information between two entities, with the objective of evading discovery or circumventing security protocols. This research article introduces a novel methodology for creating an audio covert channel, which facilitates the unauthorized transmission of confidential data, such as Device ID, Contact Number, and SMS, between users without detection.

The author initiates the development of a file-sharing program that possesses the capability to facilitate the exchange of audio files among users. By utilizing the provided audio recordings, an undisclosed communication pathway is constructed with the purpose of transmitting data between multiple users. The experiment was carried out using three distinct file formats, so illustrating the viability of information transmission across all formats. Multiple iterations of Android smartphones have undergone rigorous testing, leading to the discovery that all devices examined possess the potential to inadvertently expose sensitive information. There are potential ramifications that may arise, including the infringement of privacy, unauthorized access to data, and breaches in security.

TABLE IV. Experiment on Different Android Devices

Sr.No.	Attribute	Parameter	Version 9	Version 11	Version 12
1	Permissions	Internet	Yes	Yes	Yes
		Read Phone State	No	No	No
		Read Privileged Phone State	No	No	No
		Read Contact	No	Yes	Yes
		Read Sms	No	Yes	Yes
2	API Level	N/A	28	30	31-32
3	Is File Sharing Possible?	N/A	Yes	Yes	No
4	Is File Receiving Possible?	N/A	Yes	Yes	Yes
5	Sensitive Information Received from Audio File	Device ID, Contacts, Sms	Yes	Yes	Yes
6	Testing with other Audio Format	Mp3, Wav, Ogg	Yes	Yes	Yes

TABLE V. Comparision of Results from Different Detection Tools

Sr.No.	Tool	Manifest Analysis	File / Code Analysis	Control Flow Analysis	Threat Analysis	Data Leak Identified?
1	Flowdroid	No	Yes	No	No	No
2	Apkleaks	No	No	No	No	No
3	MobSF	Yes	Yes	No	No	No
4	RiskInDroid	Yes	No	No	No	No
5	StaCoAn	No	No	Yes	No	No
6	Super	Yes	Yes	Yes	No	No
7	Pithus	Yes	Yes	Yes	Yes	No
8	Virus Total	Yes	Yes	No	Yes	No

In addition, the author conducts an evaluation of various tools designed for the identification of covert channels, ultimately determining that none of these techniques were successful in detecting the pre-existing covert channel. In order to further advance the research, the author proposes the implementation of experimental investigations to examine the potential leakage of other sensitive information, such as battery status, alerts, network connectivity, reminders, and other related data. Additionally, the author recommends the development of detection mechanisms to effectively identify and mitigate such instances of information leakage.

The references for this research paper can be found in the subsequent section.

References

[1] H. Y. Raval, S. M. Parikh, and H. R. Patel, "Self-maintained health surveillance artificial intelligence assistant," in *Handbook of Research on Lifestyle Sustainability and Management Solutions Using AI, Big Data Analytics, and Visualization*. IGI Global, 2022, pp. 168–184.

[2] H. Raval, "Limitations of existing chatbot with analytical survey to enhance the functionality using emerging technology," *International Journal of Research and Analytical Reviews (IJRAR)*, vol. 7, no. 2, 2020.

[3] N. Duggal. (2023) Top 20 cybersecurity trends to watch out for in 2023. [Online]. Available: https://www.simplilearn.com/top-cybersecurity-trends-article

[4] J. Kotak, A. Shah, and P. Rajdev, "A comparative analysis on security of mqtt brokers," 2019.

[5] N. James. (2023) 160 cybersecurity statistics 2023 [updated]. [Online]. Available: https://www.getastra.com/blog/security-audit/cyber-security-statistic

[6] A. Shah, D. Rathod, and D. Dave, "Ddos attack detection using artificial neural network," in *International Conference on Computing Science, Communication and Security*. Springer, 2021, pp. 46–66.

[7] S. M. (2023) Types of cyber attacks you should be aware of in 2023. [Online]. Available: https://www.simplilearn.com/tutorials/cyber-security-tutorial/types-of-cyber-attacks

[8] A. Shah, P. Rajdev, and J. Kotak, "Memory forensic analysis of mqtt devices," *arXiv preprint arXiv:1908.07835*, 2019.

[9] C. Cilli. (2017) Understanding covert channels of communication. [Online]. Available: https://www.isaca.org/resources/news-and-trends/isaca-now-blog/2017/understanding-covert-channels-of-communication

[10] B. W. Lampson, "A note on the confinement problem," *Communications of the ACM*, vol. 16, no. 10, pp. 613–615, 1973.

[11] N. T. Cam, P. N. Duy, V. N. Tan, N. V. Thinh, N. C. Duc, and N. Q. Bao, "Inter-application communications using covert channels in android applications," in *International Conference on Intelligent Computing & Optimization*. Springer, 2022, pp. 200–209.

[12] M. Graa, N. Cuppens-Boulahia, F. Cuppens, and A. Cavalli, "De-

tecting control flow in smarphones: Combining static and dynamic analyses," in *Cyberspace Safety and Security: 4th International Symposium, CSS 2012, Melbourne, Australia, December 12-13, 2012. Proceedings 4*. Springer, 2012, pp. 33–47.

[13] S. Z. Goher, B. Javed, and N. A. Saqib, "Covert channel detection: A survey based analysis," in *High Capacity Optical Networks and Emerging/Enabling Technologies*. IEEE, 2012, pp. 057–065.

[14] CWE. (2023) Covert storage channel. [Online]. Available: https://owasp.org/www-community/vulnerabilities/Covert_storage_channe

[15] OWASP. (2023) Cwe-385: Covert timing channel. [Online]. Available: https://cwe.mitre.org/data/definitions/385.html

[16] Kaspersky. (2016) Triada: organized crime on android. [Online]. Available: https://www.kaspersky.com/blog/triada-trojan/11481/

[17] CHECKPOINT. (2017) The judy malware: Possibly the largest malware campaign found on google play. [Online]. Available: https://blog.checkpoint.com/securing-user-and-access/judy-malware-possibly-largest-malware-campaign-found-google-play/

[18] S. Chandra, Z. Lin, A. Kundu, and L. Khan, "Towards a systematic study of the covert channel attacks in smartphones," in *International Conference on Security and Privacy in Communication Networks: 10th International ICST Conference, SecureComm 2014, Beijing, China, September 24-26, 2014, Revised Selected Papers, Part I 10*. Springer, 2015, pp. 427–435.

[19] A. Al-Haiqi, M. Ismail, R. Nordin *et al.*, "A new sensors-based covert channel on android," *The Scientific World Journal*, vol. 2014, 2014.

[20] R. Casolare, F. Martinelli, F. Mercaldo, A. Santone *et al.*, "Colluding covert channel for malicious information exfiltration in android environment." in *ICISSP*, 2021, pp. 811–818.

[21] S. Li, L. Zhang, and C. Zhao, "An overview of android covert channel," in *2018 2nd IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC)*. IEEE, 2018, pp. 482–485.

[22] M. Guri, "Magneto: Covert channel between air-gapped systems and nearby smartphones via cpu-generated magnetic fields," *Future Generation Computer Systems*, vol. 115, pp. 115–125, 2021.

[23] C. Liang, Y.-a. Tan, X. Zhang, X. Wang, J. Zheng, and Q. Zhang, "Building packet length covert channel over mobile voip traffics," *Journal of Network and Computer Applications*, vol. 118, pp. 144–153, 2018.

[24] C. Liang, X. Wang, X. Zhang, Y. Zhang, K. Sharif, and Y.-a. Tan, "A payload-dependent packet rearranging covert channel for mobile voip traffic," *Information Sciences*, vol. 465, pp. 162–173, 2018.

[25] J.-F. Lalande and S. Wendzel, "Hiding privacy leaks in android applications using low-attention raising covert channels," in *2013 international conference on availability, reliability and security*. IEEE, 2013, pp. 701–710.

[26] X. Zhang, Y.-A. Tan, C. Liang, Y. Li, and J. Li, "A covert channel over volte via adjusting silence periods," *IEEE Access*, vol. 6, pp. 9292–9302, 2018.

[27] A. Shah, D. Rathod, and J. Rami, "Uncovering the threat: Exploring covert channel attacks via audio files in android applications," in *Lecture Notes in Networks and Systems: International Conference on Data Science and Applications*. Springer, 2024, pp. 13–22.

[28] L. Deshotels, "Inaudible sound as a covert channel in mobile devices," in *8th USENIX Workshop on Offensive Technologies (WOOT 14)*, 2014.

[29] O. I. Abdullaziz, V. T. Goh, H.-C. Ling, and K. Wong, "Network packet payload parity based steganography," in *2013 IEEE Conference on Sustainable Utilization and Development in Engineering and Technology (CSUDET)*. IEEE, 2013, pp. 56–59.

[30] Z. Yang, Q. Huang, and Q. Zhang, "Nicscatter: Backscatter as a covert channel in mobile devices," in *Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking*, 2017, pp. 356–367.

[31] M. Guri, "Air-gap electromagnetic covert channel," *IEEE Transactions on Dependable and Secure Computing*, 2023.

[32] K. Pandya, B. Borisaniya, and B. Buddhadev, "Shoutimei: Ultrasound covert channel-based attack in android," in *Security, Privacy and Data Analytics: Select Proceedings of ISPDA 2021*. Springer, 2022, pp. 293–301.

[33] R. Spolaor, L. Abudahi, V. Moonsamy, M. Conti, and R. Poovendran, "No free charge theorem: A covert channel via usb charging cable on mobile devices," in *Applied Cryptography and Network Security: 15th International Conference, ACNS 2017, Kanazawa, Japan, July 10-12, 2017, Proceedings 15*. Springer, 2017, pp. 83–102.

[34] J.-C. Wang, H.-M. Lee, C.-W. Chen, and A. B. Jeng, "Estimating intent-based covert channel bandwidth by time series decomposition analysis in android platform," in *2017 IEEE Conference on Application, Information and Network Security (AINS)*. IEEE, 2017, pp. 31–36.

[35] W. Qiang, S. Xin, H. Jin, and G. Sun, "Droidauditor: A framework for auditing covert communication on android," *Concurrency and Computation: Practice and Experience*, vol. 29, no. 19, p. e4205, 2017.

[36] S. Arzt, S. Rasthofer, C. Fritz, E. Bodden, A. Bartel, J. Klein, Y. Le Traon, D. Octeau, and P. McDaniel, "Flowdroid: Precise context, flow, field, object-sensitive and lifecycle-aware taint analysis for android apps," *Acm Sigplan Notices*, vol. 49, no. 6, pp. 259–269, 2014.

[37] W. Enck, P. Gilbert, S. Han, V. Tendulkar, B.-G. Chun, L. P. Cox, J. Jung, P. McDaniel, and A. N. Sheth, "Taintdroid: an information-flow tracking system for realtime privacy monitoring on smartphones," *ACM Transactions on Computer Systems (TOCS)*, vol. 32, no. 2, pp. 1–29, 2014.

[38] B. Buddhadev, P. Faruki, M. S. Gaur, S. Kharche, and A. Zemmari, "Flovasion: towards detection of non-sensitive variable based evasive information-flow in android apps," *IETE Journal of Research*, vol. 68, no. 4, pp. 2580–2594, 2022.

[39] A. K. Shah, D. M. Rathod, and J. Rami, "Evaluating the efficacy and capability of different covert channel detection tools in an android application," 2023.

[40] A. Kar, S. Natalia, and E. Branca, "Detecting overlay attacks in android," *Procedia Computer Science*, 2024.

[41] bunny.net. (2023) What is base64? [Online]. Available: https://bunny.net/academy/http/what-is-base64-encoding-and-decoding/

Abhinav K. Shah is pursuing his Ph.D. from the National Forensic Sciences University. Alongside, he works as a Senior Engineer at ArmorCode Inc. With approximately 7 years of experience in security and development across private and public sectors, his expertise includes Application Security Assessment, Vulnerability Assessment, Penetration Testing, Configuration Review, Vulnerability Management, Risk Reporting, Software Development, Web Application Development, Mobile Application Development, and more.

Dr. Digvijaysinh M. Rathod , Dean of School of Open Learning and Associate Professor in School of Cyber Security and Digital Forensics, and Head of Center of Excellence in Cyber Security at the National Forensic Sciences University, an Institution of National Importance under the Ministry of Home Affairs, Government of India. With 18 years of teaching, consultancy, and research experience in Cyber Security and Digital Forensic domains, Dr. Rathod has served as a cyber security consultant to the governments of Saudi Arabia and Bangladesh. He has also trained over 800 law enforcement officers from India and abroad and has been involved in high-profile digital forensic analysis cases in India. His research interests include Industrial Control System Security, Web and Mobile Security, and Dark and Deep web investigation.

Bharat Buddhadev is a Professor and Head of the Computer Engineering Department at Government Engineering College, Sector 28, Gandhinagar, Gujarat (India). Prior to this, he served as a Professor in the Computer Engineering Department at L D College of Engineering for over 24 years. With extensive experience in teaching and research spanning more than thirty years, he has supervised 5 PhD candidates covering various aspects of Computer Engineering and Information Technology. He earned his Master's degree in Computer Science and Engineering from the Indian Institute of Technology, Delhi. Currently, he is a Ph.D. research scholar at Malaviya National Institute of Technology Jaipur, actively engaged in research in the domain of Malware.

Jeet Rami works as a Threat Hunter Researcher at Microsoft. He recently graduated with a master's degree in Cyber Security from the National Forensic Sciences University. His interests lie in low-end development and information security, particularly in Windows Exploit Development. He has experience in stack-based exploits, heap exploits, and is currently exploring kernel exploit development. In malware development, he has worked on various techniques. Previously, he was involved in developing his own Linux Distro.