Digital Forensic Practices and Methodologies for AI Speaker Ecosystems

By

Wooyeon Jo, Yeonghun Shin, Hyungchan Kim, Dongkyun Yoo, Donghyun Kim, Cheulhoon Kang, Jongmin Jin, Jungkyung Oh, Bitna Na, and Taeshik Shon

From the proceedings of

The Digital Forensic Research Conference

DFRWS 2019 USA

Portland, OR (July 15th - 19th)

Digital Investigation 29 (2019) S80–S93

Contents lists available at ScienceDirect

Digital Investigation

journal homepage: www.elsevier.com/locate/diin

DFRWS 2019 USA — Proceedings of the Nineteenth Annual DFRWS USA

Digital Forensic Practices and Methodologies for AI Speaker Ecosystems

Wooyeon Jo [a], Yeonghun Shin [a], Hyungchan Kim [b], Dongkyun Yoo [a], Donghyun Kim [b], Cheulhoon Kang [c], Jongmin Jin [c], Jungkyung Oh [c], Bitna Na [c], Taeshik Shon [a, b, *]

[a] Department of Computer Engineering, Ajou University, World cup-ro 206, Suwon, 16499, Republic of Korea
[b] Department of Cyber Security, Ajou University, World cup-ro 206, Suwon, 16499, Republic of Korea
[c] Supreme Prosecutor's Office, Seocho-dong, Seocho-gu, Seoul, Republic of Korea

ARTICLE INFO

Article history:

Keywords:
AI speaker
Ecosystem
Internet of things
Cloud
NAVER Clova
KAKAO i
SKT NUGU
KT GiGA genie

ABSTRACT

Various Internet of Things (IoT) devices, such as AI speakers, are being released with different functions to improve user convenience and better life. An AI speaker ecosystem is a cloud-based IoT system built around an AI speaker and IoT devices. In the near future, citizens in whole countries worldwide will be helped in real life when AI-equipped devices are deployed in their homes. Typically, because AI speakers are always operating, they can be used to provide vital evidence for digital forensics; however, privacy issues may arise. AI speakers have provided evidence of murders in the United States and Mexico and are being released without specific regulatory guidelines. In this study, we propose five digital forensic analysis methods for four AI speaker models from different manufacturers released in the Republic of Korea. The five proposed methods are applied to all the AI speaker models, and the results are presented in the Appendix. In particular, we developed a forensic tool for collecting user command history for NAVER Clova.

1. Introduction

An AI speaker is a user-friendly convergence ecosystem that was created by combining Internet of Things (IoT) and cloud computing. The AI speaker ecosystem operates based on the cloud and provides Q&A services through an AI-enabled speaker ecosystem. However, an AI speaker plays a pivotal role as a system that controls various IoT devices in cooperation with mobile devices, unlike general IoT devices. Both the Q&A functions and control over an IoT product family further highlight the need for digital forensic research. The AI speaker market is growing rapidly worldwide, and Gartner predicted "By 2020, 20% of citizens in developed countries will use AI assistants to help them with an array of operational tasks." (Gartner, 2018). Given that every large-scale IT company has built or linked this attractive AI speaker ecosystem, the AI speaker penetration rate will continue to increase steadily. As a result, the importance of AI speakers in digital forensics is also steadily increasing. Market expansion has led to an increase in the number of users and service providers, thus the corresponding vendors who build the ecosystem has increased. In addition to the existing global companies such as Google, Samsung, Apple, and Amazon, big Chinese companies such as Xiaomi and Alibaba have also recently launched AI speakers. Although the AI speaker ecosystem has just been released, there are already a number of cases requiring digital forensic investigation. In a 2015 murder in Arkansas in the United States, Alexa, an AI speaker, was found at the scene of the crime and provided evidence, and Amazon cooperated with the investigation of the Alexa user who was allegedly murdered. The suspect was found not guilty in 2017 (Chavez, 2017). In a 2017 double murder case where two women were murdered in New Hampshire, USA, the judge ordered Amazon to turn over the records from an AI speaker (Osborne, 2018). In Mexico in 2017, 911 was called by an AI speaker answering a question, "Did you call the police?" which was asked by a man to his girlfriend while making death threats (Hassan, 2017). In this case, he was arrested at the scene immediately, but the AI speaker still provided key evidence. In the future, as soon as an AI speaker is found at the scene of incident, it will be treated as key evidence.

However, AI speakers cannot provide their location, even though they store a lot of data based on the cloud, and packets

[*] Corresponding author. Department of Computer Engineering, Ajou University, World cup-ro 206, Suwon, 16499, Republic of Korea.
E-mail address: tsshon@ajou.ac.kr (T. Shon).

https://doi.org/10.1016/j.diin.2019.04.013

encrypted with TLS-based communication conceal data. We do not know where constantly operating speakers record voice recognition data, how they store voice data, and how personal information is stored. Even now, this emerging cloud-based platform is infinitely proliferating without separate legal guidelines. Fortunately, if digital forensic studies including this study are actively carried out, they will provide useful references and contribute to the safety of society.

In this paper, we perform digital forensic analyses of four AI speaker models from major Internet portals (NAVER and KAKAO) and mobile communication companies (SKT and KT) that are being commercialized in the Republic of Korea. The AI speaker industry in the Republic of Korea is rapidly growing, and the major speakers that are commercialized, released, and serviced can be narrowed to four as mentioned above. The AI speakers covered in this paper are *Clova* of NAVER, *Kakao I* of KAKAO, *NUGU* of SKT, and *GiGA Genie* of KT respectively. NAVER operates the top portal site (www.naver.com) in the Republic of Korea, and KAKAO operates ‘Kakao Talk’, the top messenger service. SKT and KT are the first and second-largest telecommunication companies (mobile telecom service providers) in the Republic of Korea, and both companies control approximately 70% of the market (Song, 2018). These four companies dominate most Korean AI speaker markets. Samsung has built an AI ecosystem in their smartphone, but their speaker is not shown yet on the market. In this paper, we propose five digital forensic analysis methods and introduce a user history artifact collection tool for NAVER Clova based on proposed analysis methods. All the results from these four AI speaker ecosystems are shown in Appendix A, which shows the effectiveness of our five proposed analysis methods. Especially, the detailed analysis method and practice are illustrated through the case study of Clova.

This paper is composed as follows. Section 2 describes digital forensic techniques related to existing digital forensic studies on similar platforms and explains digital forensic techniques related to the proposed analysis methods. In Section 3, we propose five digital forensic analysis methods in an AI speaker ecosystem. Section 4 provides a detailed description of the five proposed analysis methods, and then we explain the actual applied method and analysis results from the NAVER Clova case study. Section 5 describes a digital forensic analysis tool for NAVER Clova. Section 6 discusses the limitations and implications of this paper. Section 7 concludes the analysis results and provides direction for future studies. Please note that the paper focuses on the NAVER Clova, and the differences between the speakers found during the analysis are briefly described in the corresponding sections.

2. Related research

Digital forensic research on various IoT devices and services, including AI speakers, is actively being conducted. In 2013 (Oriwoh et al., 2013), asserted that an analysis of hardware, software, and networks should be considered when applying digital forensics to IoT. In 2015 (Perumal et al., 2015), proposed a more empirical digital forensic model that can be used during an actual investigation. In 2016 (Kebande and Ray, 2016), constructed a digital forensics investigation framework, not just models. As recent research on IoT (Boztas et al., 2015), showed that it is possible to intuitively acquire more useful data for digital investigation from hardware than software by performing a digital forensic study targeting smart TV from the perspective of hardware. A digital forensic framework that includes analytic techniques and tools for the Amazon Alexa ecosystem, including AI speakers, was proposed in Chung et al. (2017). The depth and breadth of digital forensic research has been increasing continuously since the commercialization of IoT.

As mentioned previously, IoT environments are primarily cloud-based, and in the case of an AI speaker ecosystem, it is more cloud-centric than other platforms. In the cloud environment, it is difficult to perform digital forensics with existing methods because it communicates using an Android mobile application or encryption protocols, such as the HTTPS protocol. In Lin et al. (2015), cloud-based network forensics was performed using the man-in-the-middle (MITM) technique to solve this problem. As a method for performing MITM, it is common to intercept an authentication certificate as shown in Soghoian and Stamm (2011), and it is possible to use a vulnerability in the case of an encryption packet using the HTTPS protocol as in Callegati et al. (2009); Burkholder (2002). In this paper, we also used MITM to analyze communication between cloud and Android mobile. However, unlike existing studies that only analyzed the content of packets, we determined the server configuration in the cloud and the information stored in the server by collecting the information such as IP, domain, user's credentials, etc. included in multiple packets.

In Android mobile applications, forensic data collection, and analysis requires identifying the Android-specific data storage structures accessible by privileges. In Lessard and Kessler (2010), data storage structures were classified based on access rights and methods. Storage format and the generation structure of stored data were analyzed in Immanuel et al. (2015). The analysis in Chung et al. (2017) focused on the web cache, which is stored data generated by Amazon Alexa. These studies show that it is important to consider the storage structure, access rights, the type of generated data inherent to the platform when analyzing Android mobile applications. In this study, not only the analysis of the web cache but also all the data generated by the application is analyzed to derive voice response information, user information, voice command time information, etc. On the one hand, forensics using application decompilation require static and dynamic analysis techniques for source code. The importance of static analysis was also mentioned in forensic analysis research for malware detection (Spreitzenbarth, 2013). On the other hand (Desnos and Gueguen, 2011), proved that the dynamic analysis technique is also essential by extracting useful detailed information that cannot be obtained by static analysis through dynamic analysis of variables, fields, and methods in Dalvik byte code. The decompilation analysis proposed in this paper also includes digital forensic analysis using both techniques. In the previous Amazon Alexa study, no mention of the information available through the decompilation. In addition, the AI speaker chip-off analysis had remained as a future project. These analyses could be a very important analysis that can cross-validate the results of other analyses.

Knowledge and many practices of the EXT4 file system are required to analyze the chip-off image from an AI speaker, and the analysis of the EXT4 file system data structures was conducted in Wong (2018). Recently, research on reconstructing deleted files based on EXT4 metadata was conducted on IoT devices that use EXT4 as default file system like Android (Jo et al., 2018). We used these studies to restore and analyze deleted files when performing chip-off image analysis (Lee et al., 2019).

3. Forensic analysis of AI speaker ecosystem

An AI speaker ecosystem is classified as an IoT system and operates in conjunction with various IoT devices and services, but it has major components that are central to it. With an AI speaker or mobile application as its center, this system can be linked to most Internet-based services, including basic AI secretary services. Thus,

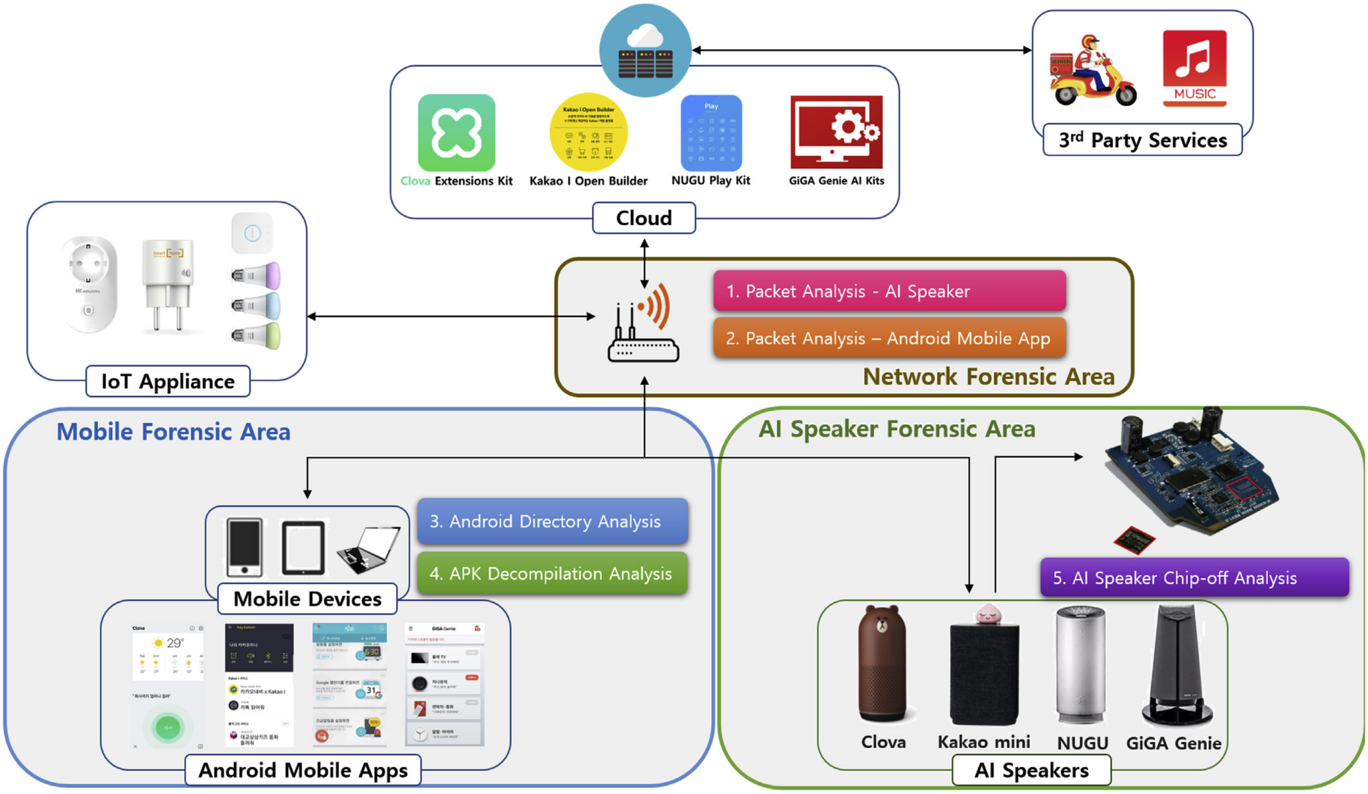


Fig. 1. Diagram of the AI speaker ecosystem analysis components.

it stores and communicates various data that can be useful for digital forensics, such as personal information, cloud storage information, and a user's whereabouts. Fig. 1 shows the major components constituting an AI speaker ecosystem, which targeted with the proposed analysis technique. This diagram can be divided into a network forensic area, mobile forensic area, and AI speaker forensic area. This diagram also shows total 5 analysis techniques: packet analysis via an AI speaker, packet analysis via Android mobile apps, Android directory analysis, Android application package (APK) decompilation analysis, and AI speaker chip-off analysis. Android mobile applications are a major part of AI speaker ecosystem, and it was determined that there exist various applications that support user recognition to provide AI secretary services directly without using an AI speaker. Understanding the need for analyzing these applications, we defined three forensic components in order to thoroughly analyze all components. The analysis techniques of network forensic area consist of packet analysis via an AI speaker and packet analysis via an Android mobile app. The analysis techniques of in the mobile forensic area consists of android directory analysis, APK decompilation analysis. Finally, AI speaker forensic area consists of chip-off image analysis of AI speaker.

3.1. *Packet analysis via AI speaker*

Packet analysis via AI speaker analyzes the communication process between an AI speaker and the cloud. This corresponds to the Network Forensic Area in Fig. 1, where packets sent from an AI speaker to the cloud via the AP(Access Point) in real time from the AP. Unlike smartphones, it is very difficult to set up a certificate on an AI speaker, so setting up a proxy is difficult. Therefore, analysis is possible only for HTTP communication.

3.2. *Packet analysis via android mobile app*

Packet analysis via an Android mobile app is used to analyze communication between an application and the cloud. This corresponds to the Network Forensic Area in Fig. 1, where packets sent from the Android mobile app to the cloud via the AP are collected in real time. Unlike an AI speaker, an Android mobile app allows the approach via MITM through the proxy setting, which enables packet analysis of HTTPS communication (Lin et al., 2015; Soghoian and Stamm, 2011; Callegati et al., 2009; Burkholder, 2002). Communication with cloud exchanges mostly JSON data (Chung et al., 2017), which includes meaningful information about the user of the device.

3.3. *Android directory analysis*

Android directory analysis corresponds to the Mobile Forensic Area in Fig. 1. Data stored in a smart phone is collected and analyzed using the Android mobile app. The Android mobile app communicates with the cloud while using applications, such as an AI speaker configuration and voice commands. During this process, some of the data exchanged with the cloud is stored in the application directory. Meaningful artifacts such as personal information, information on a connected speaker, and voice command information can be obtained by analyzing the Android mobile application directory.

3.4. *APK decompilation analysis*

The analysis technique described in Section 3.4 corresponds to Mobile Forensic Area and describes how to obtain artifacts for

Android mobile app to provide an AI speaker service. An Android mobile app communicates with the cloud to process the user's voice input, mainly using the REST API. By analyzing this data, it is possible to acquire the REST API address and the data transmitted to the server, and the other data stored in the server or stored in the device can be presumptive and classified. Furthermore, accessing the cloud directly through the acquired data can aid digital forensic investigation without the constraints of configuring the AI ecosystem.

Information sent from the AI speaker and application to the cloud with details is analyzed by decompiling the Android mobile application. The results can be used as cross-validation data for results from other analyses.

3.5. *AI speaker chip-off analysis*

AI speaker chip-off image analysis corresponds to the AI speaker forensic area in Fig. 1, which is an image-based digital forensic analysis technique. An AI speaker is an Android-based speaker device that communicates the user's voice inputs and voice responses. Therefore, this analysis technique affects the analysis of the Android mobile application and the AI speaker communication packets. In addition, most Android devices have system partitions, which can affect APK decompilation because installation applications are present on the partitions. ID information required for the cloud to recognize the user, the user's personal information, and information related to the use history are stored in an AI speaker, which can be used for digital forensics.

When the AI speaker is physically disassembled and dumped, analysis of the AI speaker image on the internal memory chip begins. Because an AI speaker communicates directly with the cloud that provides the AI speaker service through the user's voice, this analysis can be used to obtain factors that are used to identify a device and its user in the cloud. In addition, since personal information can be identified, e.g., the user's name and address, the analyst may be able to obtain meaningful information when performing actual digital forensic analysis. Detailed techniques include data accumulation, chip-off and dump, file system identification, directory structure analysis, file signature and keyword analysis, and deleted file recovery.

4. Practicing to collect and analyze artifacts in AI speaker ecosystems

4.1. *Packet analysis via AI speaker & android mobile apps*

An AI speaker and an Android mobile app use HTTP and HTTPS protocols to communicate with the cloud. The Android mobile app can communicate directly with the cloud and is used to set the communication section between the AI speaker and the cloud through connection and configuration of an AI speaker. The method for collecting and analyzing data from an AI speaker and Android mobile app requires the same analytical techniques because an AI speaker connected through an Android mobile app uses HTTP and HTTPS protocols for communication with cloud, just like the Android mobile app. However, since HTTPS requires the use of a web proxy, there is a difference in whether a certificate can be installed in the target device. This section introduces methods for collecting data from an AI speaker and Android mobile app, and the methods for analyzing HTTPS-encrypted communication are discussed. First, we configure an environment for capturing AI speaker packets and collect the usage data for AI speaker and an Android mobile app in real time. If it is difficult to set up a proxy for an AI speaker, we use Wireshark for data collection. Telerik's Fiddler web proxy, a secure HTTP protocol decoder, is used for Android mobile apps with proxy settings. Fiddler can decrypt protected traffic using the MITM technique. In order to view the contents of a packet after intercepting the certificate used by the HTTPS protocol through Fiddler, Fiddler's certificate must be installed on the smartphone.

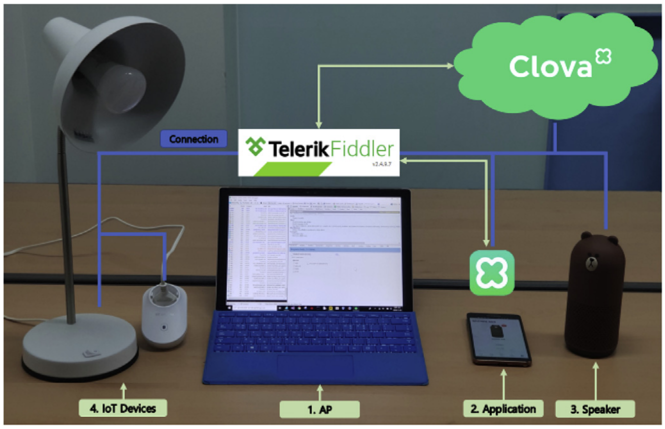


Fig. 2. Clova ecosystem testbed.

Fig. 2 shows the packet capture environment for the aforementioned data collection method. This figure is based on the Android mobile app and a includes Note PC used as an AP that mediates communication between the Android mobile app and the cloud, Android mobile apps, a NAVER Clova AI speaker, and IoT Devices (1–4 in Fig. 2, respectively). The blue solid line in the figure indicates connections between devices. All communication occurs via the AP, and the IoT device, application, speaker, and the cloud are all connected to AP. The yellow line indicates the direction of data communication, and packet data between the application and the cloud is communicated bidirectionally via the AP. At this time, the AP can collect packet data that each device communicates with the cloud in real time.

Data are collected for all supported functions by referring to the command manual provided by the AI speaker's manufacturer. The data collected through the Android mobile app can be used to analyze HTTPS communication through the proxy setting. Thus, the user profile and important information between the cloud and the Android mobile app can be collected. Therefore, one can be expected that a meaningful result can be obtained by analyzing the data collected while using the application, such as login information, in which the user's account can be acquired.

When analyzing the collected data, the main communication server, function for each server, and HTTP/HTTPS packet data are analyzed. A summary of the significant artifacts derived from the analysis of the collected data is provided in Appendix A.

Information that can be derived from analysis of the collected data includes user account information, the access token used by AI speaker to communicate with the cloud, information on connected

Table 1
NAVER Clova's account artifact.

Field	Value
birthday	19**-0*-*8
callname	shin*****hun
email	si****00*@*an**r.com
gender	M
name	shin*****hun
nickname	hoo**hun
profile_image_url	https://ssl.pstatic.net/static/pwe/address/img_profile.png
user_id	s*h**47

devices, device usage time, history of commands given to a device, information regarding data stored in the cloud, and the internal structure of the cloud.

Typically, in the case of NAVER Clova, the prod-ni-cic.clova.ai server has the highest communication share in all functions and stores user information. It is possible to obtain meaningful artifacts from the server.

Table 1 shows information on a user's account stored in the prod-ni-cic.clova.ai server. The artifacts can be obtained by analyzing the data collected during the login process. This information is in JSON format and is returned from the server, and it is possible to obtain detailed information about a user's account, such as their birthday, callname, email, gender, name, nickname, profile image, and user_id.

Table 2 shows the artifacts of the command history stored in the prod-ni-cic.clova.ai server. These artifacts can be obtained by analyzing the data collected while using the application. This information is in JSON format and is returned from the server. The cloud server stores up to 100 commands, and Table 2 indicates one of 100 stored commands. From the analysis of these artifacts, we can obtain information such as the clientName and deviceName used by a user in a voice command, actionList::value (the device operation content for a command), paragraphText (the voice response content for a command), domain (the type of a command), query (the content of a command), and time (command time). If we know the access token, these artifacts can be acquired through a replay attack or by sending a packet directly to the prod-ni-cic.clova.ai server.

Table 3 shows information on the access tokens stored in the auth.clova.ai server. These artifacts can be obtained by analyzing the data collected during the login process. This information is in JSON format and is returned from the server. This information is used for communication with the prod-ni-cic.clova.ai server. The token validity period can be expressed as follows and is valid for a period of 150 days.

4.2. Android directory analysis

Information from the configuration file on the user's personal information and cloud communication data can be stored in the directory of Android mobile app. Android directory analysis collects and analyzes stored data and derives significant information. Android directory analysis consists of three steps: preliminary data collection, primary data collection, and data analysis. The final result summarizing the significant information derived from an analysis of the collected data is provided in Appendix A. The following section shows a detailed process for collecting and analyzing data from a NAVER Clova Android mobile app.

Table 2
NAVER Clova's history artifact.

Field	Value
clientName	FRIENDS
deviceName	SALLY
dialogRequestId	09296c90-bd8e-4edf-af5a-3c35a40427c5
messageId	c2e8523c-b719-4f3e-ab5a-68736876ea9a
actionList::type	Action
actionList::value	clova://device-control?command = Increase&target = volume
paragraphText::type	String
paragraphText::value	Volume increased
domain	Control
query	Increase the volume three levels
id	586248fa-33d7-49f6-9203-c79935b7ea70
requestId	0c23cd9d-3368-48ee-bb63-5f4a91db343c
time	2018-06-21T13:58:59 + 09:00

Table 3
NAVER Clova's access token.

Field	Value
access_token	K01izy2NSkSWeH37yX9dUg
expires_in	12960000
refresh_token	AA0EudH-QbGw8mQx8hVPJg
token_type	Bearer

Android rooting and identify package name are the prerequisites required for data collection. Generally, data from an Android mobile app is stored in the '/data/data' subdirectory as the package name of the application, which we cannot access without root user privileges. Therefore, we need to perform Android rooting to access data on the Android mobile app. Android rooting uses the Odin tool, Samsung's Android smartphone repair program. After that, the identify package name is used to identify the path of the directory of the Android mobile app to be analyzed. Identify package name uses the ADB tool. Table 4 shows the result of obtaining the package name from NAVER Clova using ADB.

ADB is used to extract the NAVER Clova directory to be analyzed during the data collection step. There are three data extraction times: right after installing the Android mobile app, after using the Android mobile app, and after logging out. The data extracted right after installing the Android mobile app is compared with the data extracted after using the application. By analyzing data with different data collection times, it is possible to see how the data changes in the process of using the application. At the time of using the Android mobile app, we need to utilize all supported functions specified the manuals provided by the manufacturer of each AI speaker in order to accumulate all the data that can be generated. The data extracted after logging out from the Android mobile app can be used to determine previous usage history when the subject of an investigation logs out to conceal the usage history. Table 5 shows information on the data of the three points collected on the NAVER Clova.

The techniques used in the data analysis step include directory structure analysis, file signature analysis, and detailed analysis. These analyses are performed sequentially, and the results of prior analysis are used for further analysis. File signature analysis can be used to restore file extensions, and files containing useful information are selected and analyzed in detail. Details of each analysis technique are as follows:

Directory structure analysis is performed to determine the depth of each directory, number of files, their sizes, and the purpose of the directory. This allows us to see the information and storage paths stored on their smartphones when users use the Android mobile app.

File signature and keyword search is used to determine the file signature of the data collected using a hex editor. Typical file signatures include JPEG (FF D8 FF E0) and PNG (89 50 4E 47), and keywords include SQLite format and <xml>. This allows us to restore the original file or restore the file with no extension.

Detail analysis is used to analyze details of the data using the exclusive viewer for the file where the file signature and keyword are analyzed. For the detail analysis, if the restored file is a DB file, we can use the DB browser for SQLite, and we can use XML Viewer if

Table 4
NAVER Clova package collection details.

Application Name	Package Name	Version
NAVER Clova	com.naver.nozzle	2.11.0

Table 5
NAVER Clova package collection details.

Collection Point	Number of Files	Size
Installing Application	19/151	1,721,908 Byte
Using Application	25/1683	106,853,251 Byte
Sign out Application	25/1770	107,625,050 Byte

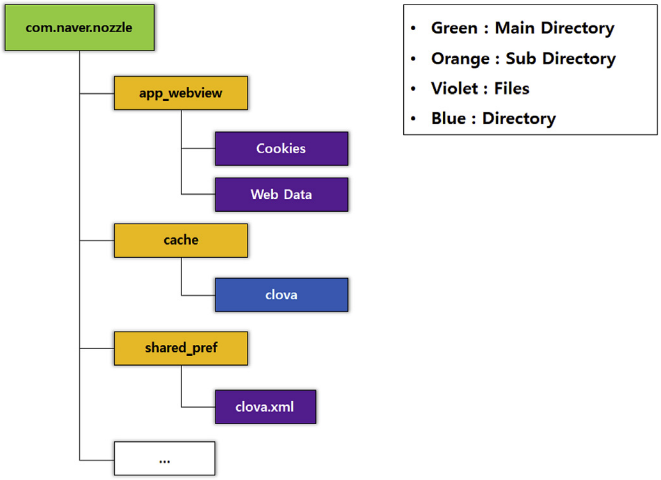


Fig. 3. Directory entry of the Android NAVER Clova application.

Table 6
Directory information.

Directory Name	Purpose
app_webview	Web contents during application operation
Cache	Cache files generated
└ clova	Speaker's voice response file
shared_prefs	Saving application settings

it is XML file (Lessard and Kessler, 2010).

The following is an example showing the results obtained from NAVER Clova with the above three analysis techniques.

NAVER Clova's directory analysis result is shown in Fig. 3. This figure shows the structure of the directory that stores significant data, where insignificant directories are excluded. Table 6 shows the results from analyzing the meaning of the information in each folder based on directory structure analysis.

File signature analysis is applied to files without an extension. Fig. 4 shows the signature analysis results of a file that lost its extension using HxD, and we confirmed that it is a .db file. Table 7 shows the major artifacts obtained as a result of analyzing the file signatures in each directory.

The results of *Detail analysis* performed for major artifacts obtained through the file signature analysis are summarized as follows:

- **Webview data:** Data related to Webview is stored in a web data file in app_webview directory, and the web data file can be determined to be SQLite DB file through file signature analysis. Fig. 5 shows the results of a detailed analysis for a web data file. Information in the web data file is stored in the autofill table, and the name field is the attribute of the storage information, and the value field is the value of the attribute in the name field. We can confirm that the web data file contains information such as the user name, phone number, and address. We also confirmed that the previous user's information was stored, even

Table 7
NAVER Clova application directory artifacts.

Client Artifact	Type	Path	File Name
Webview data	X	app_webview/	web data
Voice command data	MP3	cache/clova/	*.mp3
User setting data	XML	shared_prefs/	clova.xml

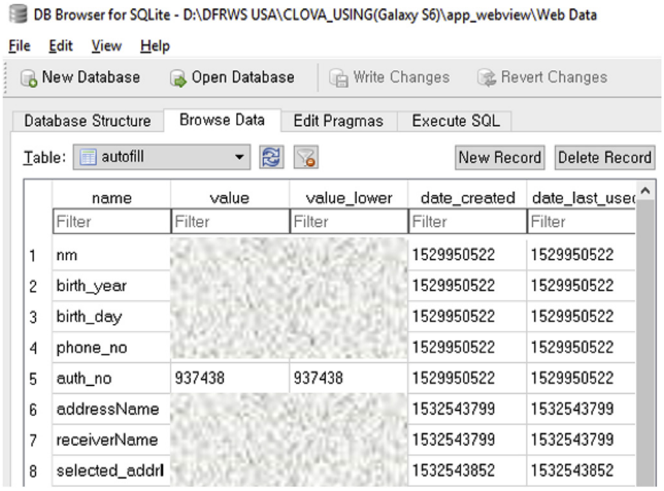
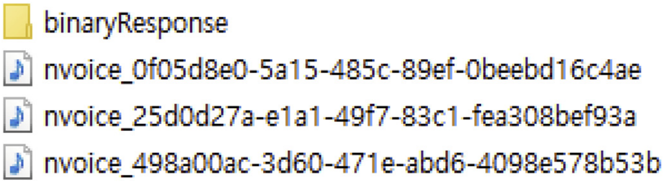


Table: autofill

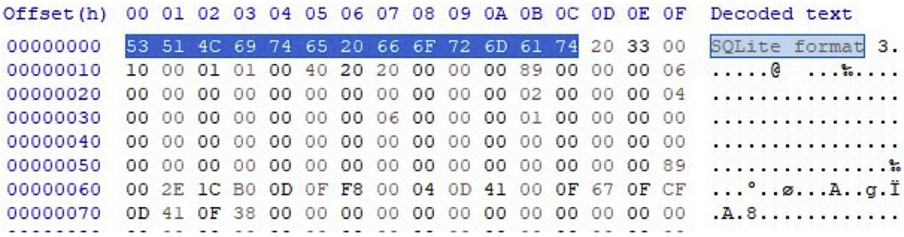
	name	value	value_lower	date_created	date_last_usec
	Filter	Filter	Filter	Filter	Filter
1	nm	[redacted]	[redacted]	1529950522	1529950522
2	birth_year	[redacted]	[redacted]	1529950522	1529950522
3	birth_day	[redacted]	[redacted]	1529950522	1529950522
4	phone_no	[redacted]	[redacted]	1529950522	1529950522
5	auth_no	937438	937438	1529950522	1529950522
6	addressName	[redacted]	[redacted]	1532543799	1532543799
7	receiverName	[redacted]	[redacted]	1532543799	1532543799
8	selected_addrl	[redacted]	[redacted]	1532543852	1532543852

Fig. 5. Web Data file with added extension (.db).



binaryResponse
nvoice_0f05d8e0-5a15-485c-89ef-0beebd16c4ae
nvoice_25d0d27a-e1a1-49f7-83c1-fea308bef93a
nvoice_498a00ac-3d60-471e-abd6-4098e578b53b

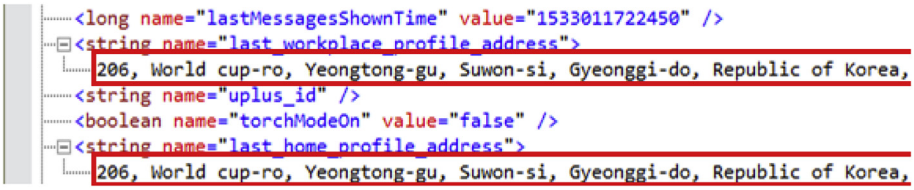
Fig. 6. Extracted file with recorded voice response.



```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text

00000000  53 51 4C 69 74 65 20 66 6F 72 6D 61 74 20 33 00  SQLite format 3.
00000010  10 00 01 01 00 40 20 20 00 00 00 89 00 00 00 06  .....@  ...‰....
00000020  00 00 00 00 00 00 00 00 00 00 00 02 00 00 00 04  ................
00000030  00 00 00 00 00 00 00 06 00 00 00 01 00 00 00 00  ................
00000040  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000050  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 89  ...............‰
00000060  00 2E 1C B0 0D 0F F8 00 04 0D 41 00 0F 67 0F CF  ...°..ø...A..g.Ï
00000070  0D 41 0F 38 00 00 00 00 00 00 00 00 00 00 00 00  .A.8............
```

Fig. 4. File signature identification using the HxD program.



```
<long name="lastMessagesShownTime" value="1533011722450" />
<string name="last_workplace_profile_address">
    206, World cup-ro, Yeongtong-gu, Suwon-si, Gyeonggi-do, Republic of Korea,
<string name="uplus_id" />
<boolean name="torchModeOn" value="false" />
<string name="last_home_profile_address">
    206, World cup-ro, Yeongtong-gu, Suwon-si, Gyeonggi-do, Republic of Korea,
```

Fig. 7. XML file containing the user's address.

after logging out of the application. This allows us to identify key information, such as the name and telephone number, even when an investigation subject has logged out in order to conceal his/her usage history.

- **Voice response data:** It can be confirmed that voice response data is stored in the cache/clova directory as an mp3 file. As shown in Fig. 6, the file is named nvoice_<hashvalue>, and it can be confirmed that the response to the user's voice command is stored.
- **User setting data:** This data has an xml extension, and user information is stored in the clova.xml file in the shared_prefs directory. Fig. 7 shows the user's login account information, his/her home and company addresses, and GPS information stored in the clova.xml file. It was confirmed that after logging out of the application, the login account information of the previous user is deleted, but some setting information such as home address and company address is not deleted.

4.3. APK decompilation analysis

The static analysis and dynamic analysis processes are used for decompilation-based forensic analysis of an Android mobile app. Static analysis uses JaDX (Dupuy and JD-Gui, 2012) for host-language level code inspection and apktool (Winsniewski, 2012) for baksmaling, a preliminary process for smali code analysis and modification. The APK may need to be modified in order to obtain meaningful artifacts. In this case, it is possible to modify the Android mobile app by modifying smali by referring to the Java bytecode (Paller, 2012) and recompiling it using apktool's smaling function. The next step is to re-sign the packed APK after recompiling and make it ready for installation. The generated APK is used for dynamic analysis. Dynamic analysis uses ADB (Android debug bridge) to analyze Android Logcat generated by the application. The operation mechanism can be identified from the log, or the transmitted/received data and URI information can be selected and analyzed.

Decompilation-based forensic analysis techniques are divided into six analysis techniques, such as call-hierarchy analysis, string-based search analysis, obfuscated class/method reconstruction, Dalvik bytecode analysis/modification, line-by-line inquiry, and Logcat-based dynamic analysis.

Call-hierarchy analysis - We can use JaDX's find call-usage function or call-hierarchy analysis function provided by the IDE by importing the decompiled source into a Java IDE.

String-based search analysis - Depending on the type of artifact, the signature of the retrieval scheme must be changed, which is not difficult for analysts; thus, this technique has the highest analytical efficiency. This can be done using the Find function in JaDX.

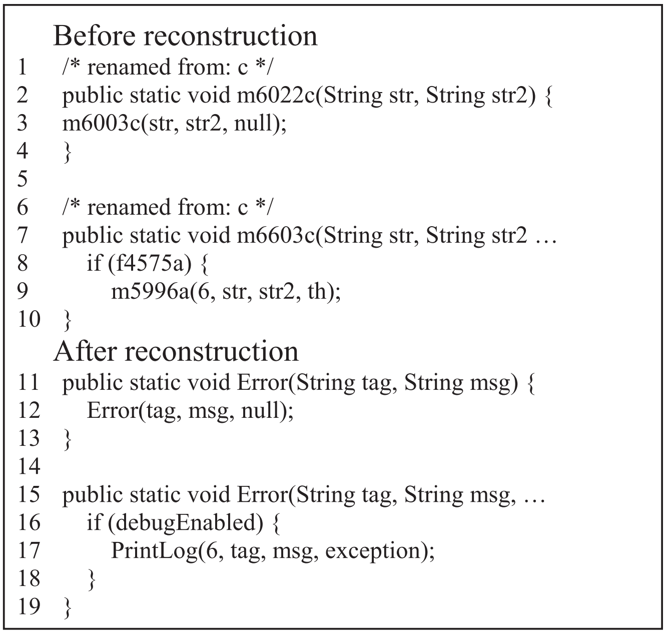
Obfuscated class/method reconstruction – While conducting an analysis, there may be a need to restore the class/method obfuscated by the obfuscator to their original forms in order to grasp their behavior. In this case, class/method reconstruction is required and contributes to the processivity of an analysis technique, indicating relatively high efficiency. We analyze the routine of an Android application by referring to the existing open-source library.

Dalvik bytecode analysis/modification – This analysis technique is used when the high-level language disassembler, e.g., JaDX, does not operate normally owing to binary obfuscation, or when the method fails to decompile owing to other factors, and it has a relatively low efficiency. Its modification with transplant-generated smali code after semi-automated build using an IDE is easier. In other cases, this is done by referring to the Dalvik op-code. Dalvik bytecode analysis requires a higher analytical level than that of Java, and so an analyst's expertise is also required.

Code line-by-line inquiring - In case an analysis of Java code fails (except Dalvik bytecode analysis), an obfuscation routine can be determined by inquiring and analyzing code line-by-line. However, when an application uses many lines of code, it requires more long time to analyze, and is used as the last static analysis method.

Logcat-based dynamic analysis - This can be performed after routine analysis of the debugging code block, or after Dalvik bytecode modification. In Android Logcat, logs are generated by threads within the application. In this case, an intentional log output result can be obtained through Dalvik bytecode modification.

Using these six techniques in the NAVER Clova first requires a static analysis and loading the Clova APK file into JaDX. It should also be determined whether to use the Android.Log package as a signature string related to debugging. A string-based search is used

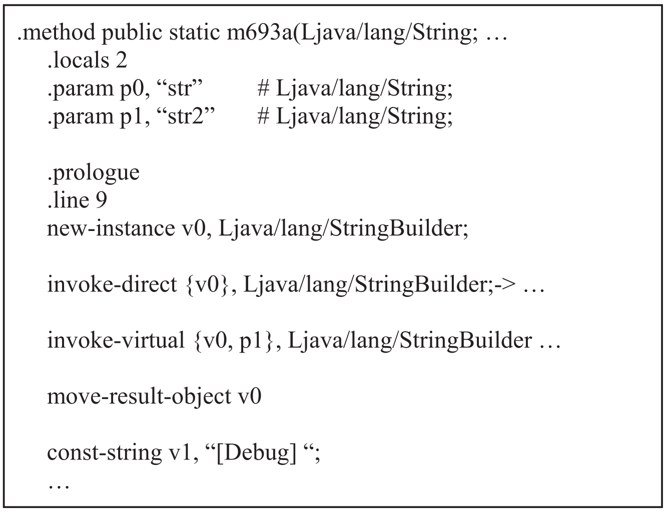


```
Before reconstruction
/* renamed from: c */
public static void m6022c(String str, String str2) {
m6003c(str, str2, null);
}

/* renamed from: c */
public static void m6603c(String str, String str2 …
    if (f4575a) {
        m5996a(6, str, str2, th);
}
After reconstruction
public static void Error(String tag, String msg) {
    Error(tag, msg, null);
}

public static void Error(String tag, String msg, …
    if (debugEnabled) {
        PrintLog(6, tag, msg, exception);
    }
}
```

Fig. 8. Reconstructing obfuscated methods from Clova Beta.



```
.method public static m693a(Ljava/lang/String; …
    .locals 2
    .param p0, “str”        # Ljava/lang/String;
    .param p1, “str2”       # Ljava/lang/String;

    .prologue
    .line 9
    new-instance v0, Ljava/lang/StringBuilder;

    invoke-direct {v0}, Ljava/lang/StringBuilder;-> …

    invoke-virtual {v0, p1}, Ljava/lang/StringBuilder …

    move-result-object v0

    const-string v1, “[Debug] “;
    …
```

Fig. 9. Target smali file of bytecode injected APK.

to locate the logging procedure.

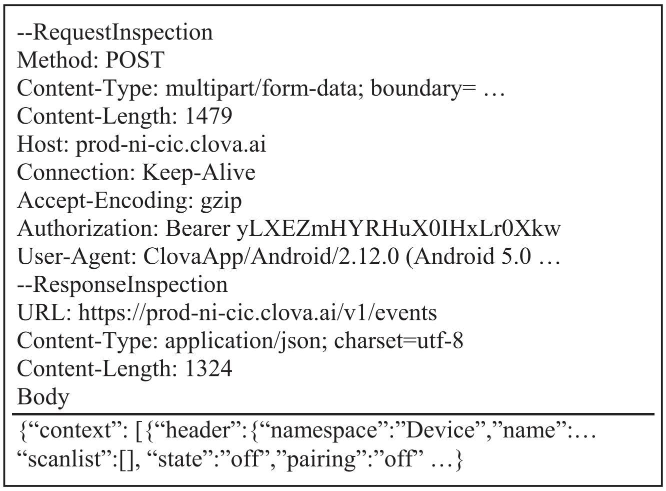
As shown in Fig. 8, the debugging method was found in Clova Beta and was not removed. The names of the top two methods were arbitrarily assigned by JaDX at decompiling time, which were originally obfuscated. The lower two methods were reconstructed by analyzing the obfuscated routines.

The debugging routine is not much different from an empty class and methods in the Beta version of NAVER Clova. This routine only adds a few additional functions. In this case, we can proceed with the Dalvik bytecode transplantation, which is possible only with some bytecode additions and modifications.

Fig. 9 shows the smali file of the baksmalied result as modified from Clova Beta. Now we apply this smali code to the latest Clova version and perform smaling to repackage the APK. However, this type of automatic smali code modification is only possible if the code is at the end so that the callee does not exist. Other smali that require injection are analyzed and built with reference to Dalvik bytecode.

Fig. 10 shows the debug log of the application extracted through ADB. The operation mechanism can be identified through this log.

As shown in Fig. 11, the final URL artifact and the data to be transmitted in the Android Logcat analysis through the additional Dalvik bytecode injection were collected, as shown in Table 8. This coincides with the results from packet analysis, which shows that it is possible to perform cross validation.



```
--RequestInspection
Method: POST
Content-Type: multipart/form-data; boundary= …
Content-Length: 1479
Host: prod-ni-cic.clova.ai
Connection: Keep-Alive
Accept-Encoding: gzip
Authorization: Bearer yLXEZmHYRHuX0IHxLr0Xkw
User-Agent: ClovaApp/Android/2.12.0 (Android 5.0 …
--ResponseInspection
URL: https://prod-ni-cic.clova.ai/v1/events
Content-Type: application/json; charset=utf-8
Content-Length: 1324
Body
{“context”: [{“header”:{“namespace”:”Device”,”name”:…
“scanlist”:[], “state”:”off”,”pairing”:”off” …}
```

Fig. 11. HTTP request/response log from the modified application.

Table 8
NAVER Clova's URL artifacts.

Behavior	URL Artifact
Speech recognition	https://prod-ni-cic.clova.ai/v1/directives
Full-text processing	https://prod-ni-cic.clova.ai/v2/events
Command history	https://prod-ni-cic.clova.ai/internal/v1/query/history
Internal ping	https://prod-ni-cic.clova.ai/ping

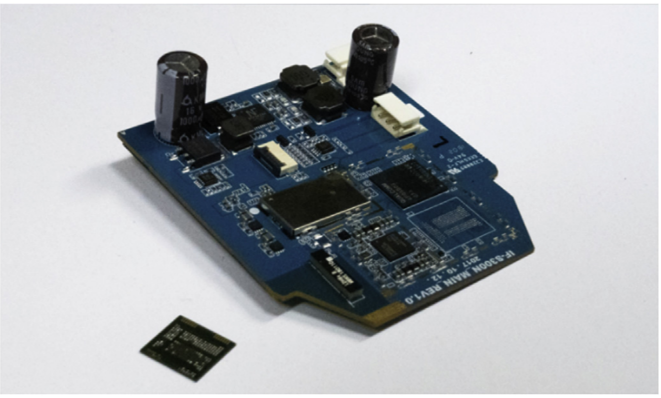
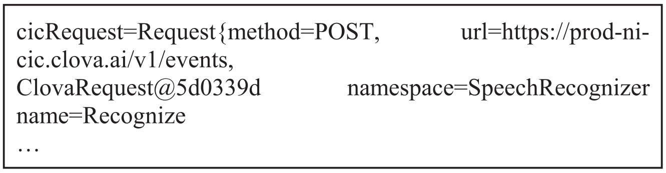


Fig. 12. Flash memory after chip-off (NAVER Clova Speaker).

Table 9
NAVER Clova's chip-off image partitions.

Partition Name	Size	Used	Filesystem
cache	1.0 GB	160 KB	EXT4
system	1.2 GB	355 MB	EXT4
userdata	4.8 GB		EXT4



```
cicRequest=Request{method=POST,           url=https://prod-ni-
cic.clova.ai/v1/events,
ClovaRequest@5d0339d            namespace=SpeechRecognizer
name=Recognize
…
```

Fig. 10. Operation log from the modified application.

4.4. AI speaker chip-off analysis

During the data accumulation step, it is not possible to know the environment in which the AI speaker is used. Therefore, all the supported functions based on the user manual are used in order to accumulate all possible data. Detailed analysis such as filesystem identification, directory structure analysis, file signature, keyword search, and deleted file recovery are performed after the flash memories containing accumulated data are collected as images by a specialist company.

Fig. 12 shows an enlarged view of the Flash memory in the NAVER Clova AI speaker, which will be discussed intensively in this section. This Flash memory is a Micron MT29TZZZ8D5JKEZB model and is an MLC NAND-based low power Flash with 8 GB capacity. Returning to the previous state and data modification are impossible after chip-off. Therefore, during data accumulation, all possible data should be accumulated by utilizing all functions supported by the AI speaker. After chip-off, information on partitions is organized as shown in Table 9, and the file system is discerned.

Table 9 shows the three main partitions, excluding the driver and simple configuration partition, as well as partitions with meaningless information, e.g., copied partitions, booting area, and backup. The primary partitions store information that can be useful to forensic analysts, such as user information, account configuration, and even voice response record. These three partitions occupy 96% of the total Clova data capacity (7,809,925,315 bytes).

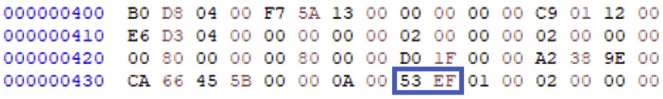
4.4.1. Detailed analysis

File system identification collects overview information from the chip-off images and then analyzes the metadata at the front of the collected image to identify the file system. In the file system identification step, the type of file system is identified, and it is confirmed whether there are multiple partitions in the image, as in the case of KT GiGA Genie. In the case of the GiGA Genie, the first boot area is uniquely created, and there are several partitions in one image. However, we were able to accurately determine size and offsets of each partition using the file system metadata.

As shown in Fig. 13, most file systems can be distinguished by their unique signatures marked at the beginning of a partition or using patterns in specific fields according to their offsets. The Android operating system is used in all the AI speakers examined in this study; Android is based on Linux and uses the EXT4 file system as the default option. The result of our identification reveals that one or more EXT4 file system partitions were detected in the chip-off images of all four AI speakers. The three main partitions of the Clova speakers selected above also use the EXT4 file system.

Directory structure analysis is used to roughly outline the internal partitions of the image after file system determination. When identifying the directory structure, the depth of each folder and the number and size of all files are identified. Table 10 shows part of the results from directory structure analysis for NAVER Clova. It represents the four largest directories by capacity, and the final result shows that major artifacts have been found in the three folders (except dalvik-cache). Only simple Android-related binaries are found in the dalvik folder. In this table, the numbers in the Folders and Files columns refer to the number of folders and files that exist directly under the path, and the numbers in the columns of Total Folders and Total Files refer to the total number of folders and files contained in the path.

File signature and keyword search is used to locate the files that correspond to the major artifacts that can be of practical help in digital investigation after the directory structure is identified. The most important thing in file signature search is the audio related files (mp3, wmv, and wav) that can contain the user's voice or response voice. Configuration files (.xml, .json) and database related files (.db, .json). (.xml, .json) are also searched. In addition to the file signature search technique, keyword search was also used, and keywords include voice, log, version, release, time, and journal. In addition, filenames found during Android directory analysis can be registered as important keywords. For example, in the Clova application on a smartphone, if a specific file name appears repeatedly while the response data is being stored as a voice file, the word is selected as a keyword and searched. In addition, the system configuration (version, release), user activity log (log, time), and file recording order (journal) in the file are searched and analyzed.



```
000000400  B0 D8 04 00 F7 5A 13 00 00 00 00 00 C9 01 12 00
000000410  E6 D3 04 00 00 00 00 00 02 00 00 00 02 00 00 00
000000420  00 80 00 00 00 80 00 00 D0 1F 00 00 A2 38 9E 00
000000430  CA 66 45 5B 00 00 0A 00 53 EF 01 00 02 00 00 00
```

Fig. 13. File signature of EXT4 file system (NAVER Clova).

Table 10
NAVER Clova's directory information.

Path	Folders	Files	Total Folders	Total Files	Size (MB)
/	36	0	592	618	175
/dalvik-cache	1	0	1	70	170
/misc	35	0	301	158	2.56
/system	13	24	85	19	0.98
/data	36	0	130	66	1.08

Deleted file recovery is used to analyze deleted artifacts due to power down during chip-off after the file search and analysis is completed. This analysis tries to recover a deleted file and repeats the detailed analysis steps for the recovered data. Fortunately, all major partitions used the EXT4 file system, so we could try file recovery on all four AI speakers (Lee et al., 2019; Wong, 2018).

4.4.2. NAVER Clova's chip-off image analysis result

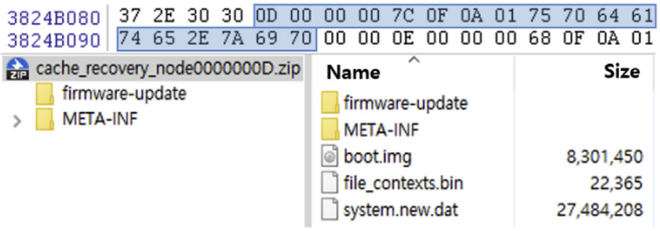
Most of the artifacts that are meaningful from a digital forensic perspective can be found in the userdata partition. The cache partition stores the data related to the update, and it is possible to restore the update file by recovering the deleted file. The system partition contains information on installed APK and configuration options.

The cache partition records information (e.g., time and version) on the most recent update centered on the 'update.zip' file shown in Fig. 14. However, update.zip (recovered) was a deleted file, leaving only traces. Fortunately, we were able to recover the file that was the core of the partition through EXT4 metadata analysis. The hex data in Fig. 14 shows the directory entry of the deleted file, which is update.zip, and the compressed file below shows the restoration result. This can be a subject of further research that can be performed by in-depth analysis of the update image.

The system partition stores .apk files, .so files, and executable binaries. These are used in APK decompilation analysis. The userdata partition contains most of user-related personal information and configuration information; this is the primary analysis target and the highest priority partition in an actual investigation.

Fig. 15 shows the directory structure of the images collected from chip-off and shows a folder structure containing the main files with meaningful data, excluding a large number of meaningless folders from the image. The main files in the folder named in Fig. 15 and the information that the files contain are as follows:

- **BluetoothInfoKey:** This .xml file stores the user's personal information and Bluetooth connection information, as shown in Fig. 16. Information on the smartphone user and the smartphone itself linked to the speaker, such as the device's mac address and device type, and name of the user.
- **UseInfoKey:** This .xml file stores various key values from personal information, such as a user's location and address. The user can be specified from the input address or latitude and longitude, or the user can be clearly specified using the key values when search and confiscation is requested. Although not shown in the figure, the file has the home address information of the user written in Korean.
- **notification_log:** This is a .db file. Although this is a file in the SQL list database format containing multiple tables, its primary data is only a log table. Fig. 17 shows a comparison of the maximum and minimum values of the 'event_time_ms' field in



```
3824B080  37 2E 30 30 0D 00 00 00 7C 0F 0A 01 75 70 64 61
3824B090  74 65 2E 7A 69 70 00 00 0E 00 00 00 68 0F 0A 01
```

cache_recovery_node0000000D.zip
firmware-update
META-INF

Name	Size
firmware-update	
META-INF	
boot.img	8,301,450
file_contexts.bin	22,365
system.new.dat	27,484,208

Fig. 14. Directory entry of 'update.zip.'

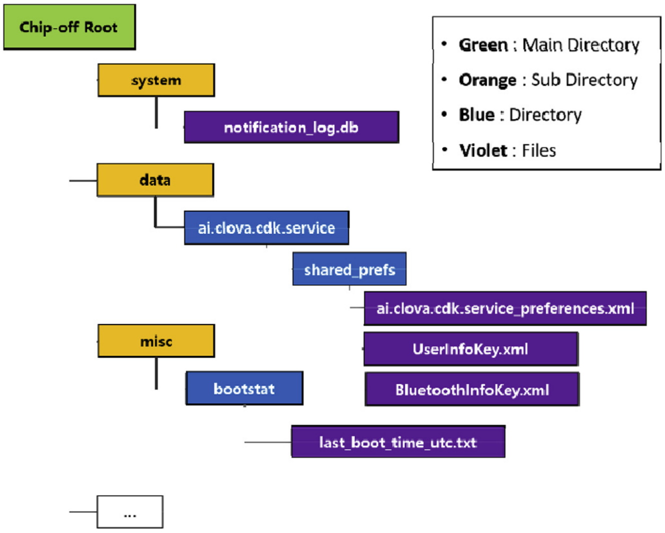


Fig. 15. NAVER Clova speaker directory entry.

the corresponding file. This file provides a rough idea of how long AI speakers have been used. Several kinds of package names and illegible broken values are shown in the 'key' field. We could find the pattern that the pkg (package) call sequences of *'friendsettings'*, *'friendsnocker'* and *'friendssound'* are always called whenever the AI speaker is turned on. And this can prove that someone was with the AI speaker at that time.

- **nvoice**: The mp3 files shown in Fig. 18 are extracted as a result of restoring deleted files. However, all files except that highlighted above are overwritten with metadata or initialized to 0, as shown in Fig. 19. Based on the fact that there was a recorded answer that greeted the user's real name when the device was started for the first time, and files with the same pattern name in the previous Android mobile application directory analysis were voice response files, we can be sure that the remaining files were originally voice responses.

5. A digital forensic tool for NAVER Clova

The artifacts collected through five proposed analysis methods can be found in Appendix A. Among the collected artifacts as shown in Appendix A, this chapter describes the tool developed to collect artifacts of command history that a user used to operate AI speaker by using user credential information collected through packet analysis. This tool works with NAVER Clova and has the function to collect command history artifacts extracted in Section 4.1 directly through communication with the cloud without using Clova Mobile Application.

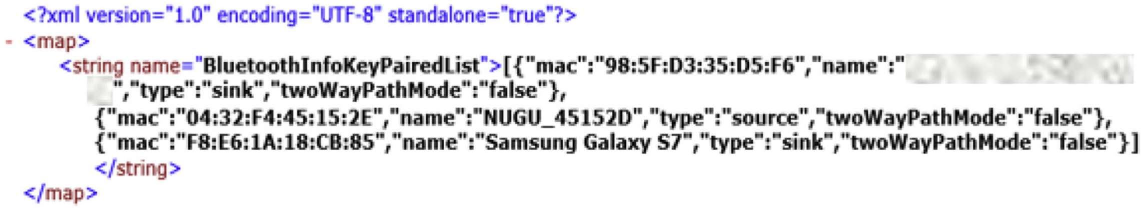
As shown in Fig. 20, the NAVER Clova Application provides a UI that shows a user the history of commands executed prior. The command record consists of a voice or text requests and responses, and up to 100 records can be called. However, the number of commands shown at a time is limited to about 20, thus we must check each by using the scroll to see the previous command. In addition, as shown in the above figure, one command occupies most of the screen. Only the date is displayed without any specific timestamp, so it is inefficient for an investigator to manually check and investigate in a restricted environment.

The NAVER Clova Application uses a unique access token assigned to each account for communication. To collect access token as shown in Fig. 21, a previously introduced analysis using a proxy could be utilized. This investigation method can sometimes be more effective than using Android mobile's storage space. Because the integrity of the digital evidence may be damaged during acquiring administrator authority on a mobile device.

According to the packet analysis results, the access token is vulnerable to a replay attack because it can be reused. Therefore, if only the access token of the user is identified, it is possible to obtain the command history stored in 'prod-ni-cic.clova.ai' server without restriction by root privileges. The Clova Digital Forensic Investigation tool has three menus: Program and Token Information, Voice Command History, and Export to Excel. Left side of Fig. 22 shows the Voice Command History is selected from those three menus. In this menu, the command history corresponding to the Access Token entered in the previous menu, called 'Program and Token Info', is obtained from the 'prod-ni-cic.clova.ai' server.

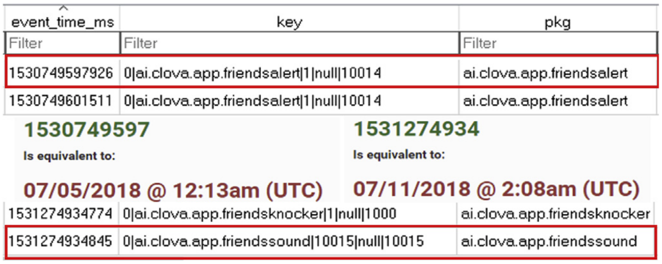
Among the command history data, the menu shows the Timestamp, Device Name, Client Name, Domain, Question, and Answer. The conversation with the AI speaker is in Korean, and the contents are also in Korean, which contains information ranging from simple music requests to a user's location and connected device information. Please note that the contents of Question and Answer in Fig. 22 are originally written in Korean and translated into English. When the conversation contained serious information (e.g., a keyword such as a murder case or suicide), the domain variable value of the command record data was confirmed as 'direct' and is indicated in red. For example, one of the questions and answers contains 'Search for the way to kill people' with 'Love all life'.

After exporting to Excel through third menu, timestamp, user ID, request id, device name, client name, and other information can be identified, as well as more detailed information than when



```
<?xml version="1.0" encoding="UTF-8" standalone="true"?>
- <map>
    <string name="BluetoothInfoKeyPairedList">[{"mac":"98:5F:D3:35:D5:F6","name":"[redacted]
      ","type":"sink","twoWayPathMode":"false"},
      {"mac":"04:32:F4:45:15:2E","name":"NUGU_45152D","type":"source","twoWayPathMode":"false"},
      {"mac":"F8:E6:1A:18:CB:85","name":"Samsung Galaxy S7","type":"sink","twoWayPathMode":"false"}]
      </string>
  </map>
```

Fig. 16. BluetoothInfoKey.xml file contents.



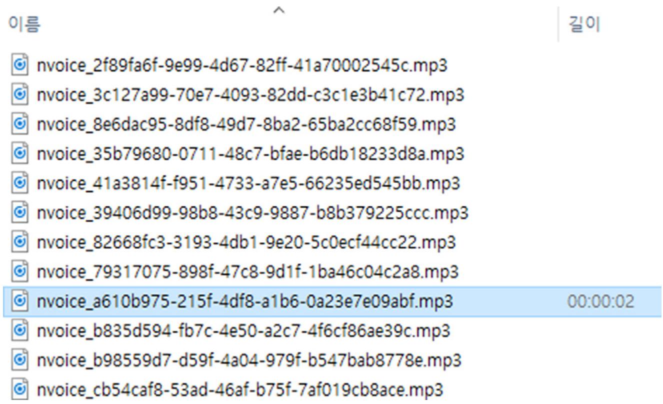
event_time_ms	key	pkg
Filter	Filter	Filter
1530749597926	0|ai.clova.app.friendsalert|1|null|10014	ai.clova.app.friendsalert
1530749601511	0|ai.clova.app.friendsalert|1|null|10014	ai.clova.app.friendsalert

1530749597
Is equivalent to:
07/05/2018 @ 12:13am (UTC)

1531274934
Is equivalent to:
07/11/2018 @ 2:08am (UTC)

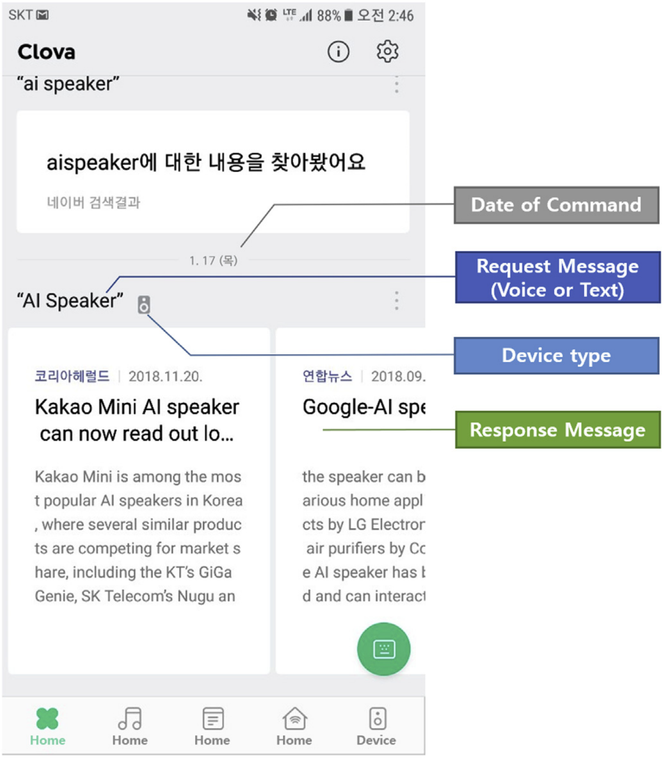
1531274934774	0|ai.clova.app.friendsknocker|1|null|1000	ai.clova.app.friendsknocker
1531274934845	0|ai.clova.app.friendssound|10015|null|10015	ai.clova.app.friendssound

Fig. 17. Notification_log.db file contents and its time in UTC.



이름	길이
nvoice_2f89fa6f-9e99-4d67-82ff-41a70002545c.mp3	
nvoice_3c127a99-70e7-4093-82dd-c3c1e3b41c72.mp3	
nvoice_8e6dac95-8df8-49d7-8ba2-65ba2cc68f59.mp3	
nvoice_35b79680-0711-48c7-bfae-b6db18233d8a.mp3	
nvoice_41a3814f-f951-4733-a7e5-66235ed545bb.mp3	
nvoice_39406d99-98b8-43c9-9887-b8b379225ccc.mp3	
nvoice_82668fc3-3193-4db1-9e20-5c0ecf44cc22.mp3	
nvoice_79317075-898f-47c8-9d1f-1ba46c04c2a8.mp3	
nvoice_a610b975-215f-4df8-a1b6-0a23e7e09abf.mp3	00:00:02
nvoice_b835d594-fb7c-4e50-a2c7-4f6cf86ae39c.mp3	
nvoice_b98559d7-d59f-4a04-979f-b547bab8778e.mp3	
nvoice_cb54caf8-53ad-46af-b75f-7af019cb8ace.mp3	

Fig. 18. Recovered voice response files.



Clova

"ai speaker"

aispeaker에 대한 내용을 찾아봤어요

네이버 검색결과

1. 17 (목)

"AI Speaker"

코리아헤럴드 | 2018.11.20.

Kakao Mini AI speaker can now read out lo...

Kakao Mini is among the mos t popular AI speakers in Korea , where several similar produc ts are competing for market s hare, including the KT's GiGa Genie, SK Telecom's Nugu an

연합뉴스 | 2018.09.

Google-AI spe

the speaker can b arious home appl cts by LG Electror air purifiers by Cc e AI speaker has t d and can interact

Fig. 20. Voice command history of Android Clova application.

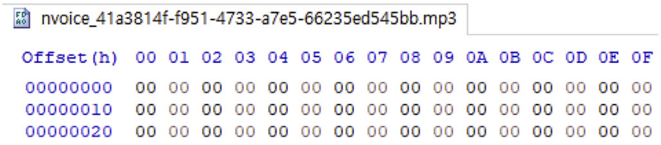
identifying the user command history using a normal mobile application.

6. Discussion

An AI speaker with the cloud-based ecosystem has a commonality in that it is served through AI speakers, mobile applications, and various IoT devices. In addition, AI speaker device platforms all use Android-based operating systems, and Android mobile applications exist, although their supported functions are different. Therefore, in the case that AI is embedded in a speaker or a mobile device without a cloud, there may be a limit to collecting meaningful information even with forensic analysis for packet communication. Furthermore, even in the case that an AI speaker uses the operating system of the embedded device, it may be difficult to analyze the data stored in the device because it contains very limited information related to the file system and analysis. If there are no mobile applications linked to AI speakers, we cannot use some of the analysis methods proposed in this paper. As mentioned in the previous analysis, AI speakers have a limitation in analyzing encrypted packets because it is impossible to set up a proxy on the speaker device itself. It may be difficult to overcome this limitation if the speaker's data cannot be modified in real time.

Therefore, in the case of an Android mobile device with proxy settings, forensic analysis based on the web proxy proposed in this paper can help analyze encrypted packets, and various user artifacts can be collected from this analysis. After that, if we analyze the stored data in the mobile application directory, compare the result of this analysis with the packet analysis results, and perform cross validation, the utility of the collected forensic artifact can increase. The same process proceeds for AI speakers, but even if the analysis does not exactly identify the data in the encrypted packets, comparing its result with chip-off analysis results, unencrypted packets, and the Android mobile application may yield sufficiently meaningful results.

In addition, the limitation that real time access to data from an

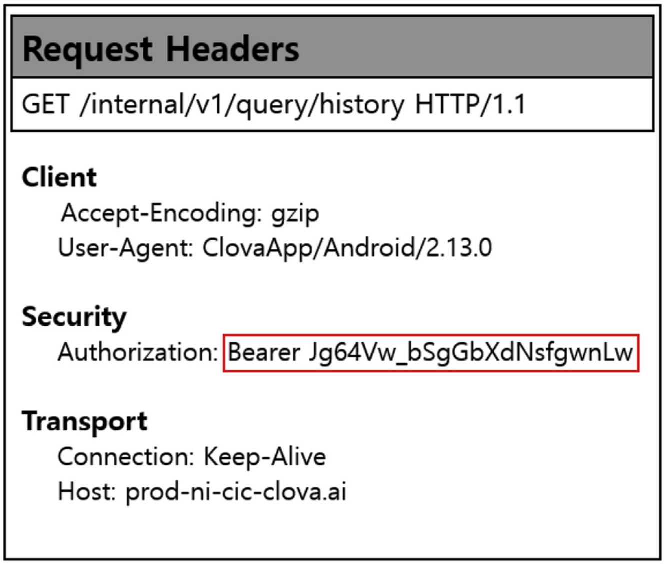


```
nvoice_41a3814f-f951-4733-a7e5-66235ed545bb.mp3

Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

00000000  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000010  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Fig. 19. Voice response file in hex view.



Request Headers

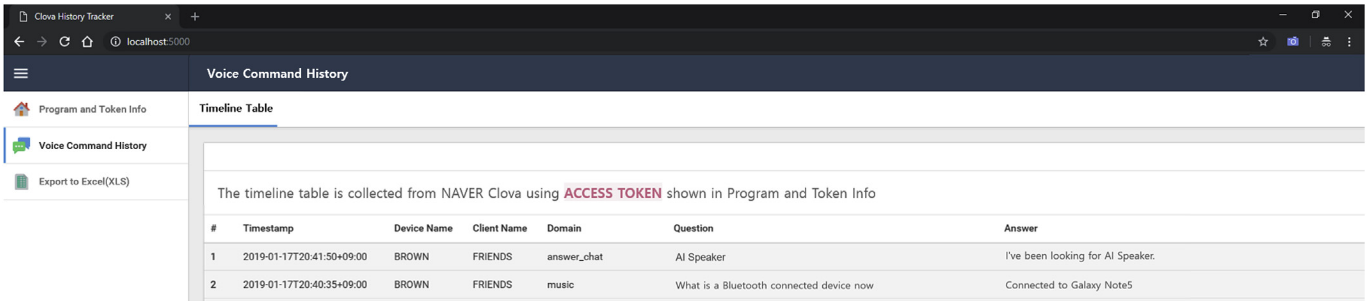
GET /internal/v1/query/history HTTP/1.1

Client
Accept-Encoding: gzip
User-Agent: ClovaApp/Android/2.13.0

Security
Authorization: Bearer Jg64Vw_bSgGbXdNsfgwnLw

Transport
Connection: Keep-Alive
Host: prod-ni-cic-clova.ai

Fig. 21. Access token information collected by Fiddler.



The timeline table is collected from NAVER Clova using ACCESS TOKEN shown in Program and Token Info

#	Timestamp	Device Name	Client Name	Domain	Question	Answer
1	2019-01-17T20:41:50+09:00	BROWN	FRIENDS	answer_chat	AI Speaker	I've been looking for AI Speaker.
2	2019-01-17T20:40:35+09:00	BROWN	FRIENDS	music	What is a Bluetooth connected device now	Connected to Galaxy Note5

Fig. 22. A digital forensics tool for NAVER Clova.

AI speaker is impossible prevents in analysis of stored data as well as proxy. Because some AI speakers reinstall all applications and check for updates every time they are run, their existing data is overwritten. In this case, if the metadata of the file system can be clearly identified, one can attempt to restore the deleted file. In addition, knowledge of the file system metadata is also used to determine the beginning and end of multiple partitions within the AI speaker. Therefore, digital forensic research on IoT devices including AI speakers must be preceded by analysis using knowledge of the device's file system. Finally, decompilation analysis (e.g., the APK) should be performed at the beginning of its release, if possible. In the case of NAVER Clova, the 2.xx version is being used in 2019. Debug mode remains in the 0.xx, 1.xx, and 2.xx versions, but the difficulty and complexity of decompilation analysis has increased sharply with newer versions. Therefore, it seems that digital forensic research can be used to obtain more information efficiently when few obstacles (e.g., code obfuscation) are present during the initial product launch.

Finally, one of the things to consider when conducting digital forensic research is the relevant legal issues. Digital forensics, among other things, does not allow the integrity to be compromised during the analysis process because only data acquired through a legitimate analysis process can be deemed legally valid. Due to this legal issue, the directory analysis method for Android mobile applications will be available only for reference purposes, such as receiving confessions, because its integrity is compromised in the process of acquiring administrator privileges. However, the analysis tool that utilizes Access Token extracted during packet analysis are different. In the case of NAVER Clova, the tools used to collect data directly from the service provider's cloud using a legitimate communication protocol have sufficient legal force. In addition, the identification information obtained from most analyses, including the record of the command collected through the tool, helps the investigating agency easily acquire a concrete search and seizure warrant. In addition, because the AI speaker chip-off analysis can ensure integrity during imaging, immediate investigation by an analyst is possible in case that a device is found in the field.

7. Conclusion

Five digital forensic analysis methods and practices for four types of AI speaker ecosystems from Korea's major Internet portals and mobile carriers are proposed in this paper. In search for network packet to identify the data stored in cloud, we performed forensic research on an AI speaker and Android mobile application, and we found that every request from a user stored in the cloud has personal information, such as identification information and the model name of used device. This is in contrast to the belief that AI speaker manufacturers will do anonymization, also known as de-identification, when storing personal information. In order to analyze mobile device storage data from an AI speaker ecosystem, we analyzed the package folder of an Android mobile application and the flash memory of an AI speaker, and we found that both mobile and speaker devices store personal information, such as a user's name, address, and account information. Furthermore, we confirmed that some personal information remains in mobile application, even if we log out. In case of an AI speaker, we were able to detect some speakers where voice response data were deleted due to a power down by restoring deleted files. In addition, in the case of the Kakao I speaker, all the voice response data was stored without any deletion. We were able to analyze most encrypted packets through proxy, but this analysis will be difficult if the tunneled traffic increases through additional internal authentication.

In future research, we will examine the tunneled session by obtaining the key in the security analysis of key distribution step and understanding the encryption algorithm through android package decompilation. On the other hand, we will extend the previously developed NAVER Clova prototype tool into an integrated forensic framework for AI speaker ecosystem.

Acknowledgements

This work was supported by the Institute for Information & Communications Technology Promotion (IITP) grant funded by the Korean government (MSIT) (No.2018-0-01000, Development of Digital Forensic Integration Platform).

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT and Future Planning (NRF-2018R1D1A1B07043349).

Appendix A. Summary of Key Artifacts

Category	Vendor	AI Cloud (Packet Analysis)	Android mobile (Android Chip-off Analysis)	AI speaker (AI Speaker Chip-off Analysis)
User Information: Information that can be used or helpful in identifying a user (e.g. user's name, Interlocking Account Data, MAC, address, ID, email, Key value, Wifi MAC, etc.)	NAVER	auth.clova.ai (/user_profile/personal_info/*) prod-ni-cic.clova.ai (/result/{DEVICE_#}/*)	shared_prefs/ (NaverOAuthLoginPreferenceData.xml clova.xml)	root\data\ai.clova.cdk.service\shared_prefs (BluetoothInfoKey.xml, UserInfoKey.xml)
	KAKAO	auth.kakao.com (/account/profile/*) app.i.kakao.com (/contents/{DEVICE_#})	shared_prefs/ (CrashReporter.Crashlytics.xml)	P6\misc\bluedroid\(bt_config.xml) P6\data\com.kakao.i.speaker\shared_prefs\(kakaoi.pref.xml)
	SKT	api.sktnugu.com (/*,/simpleSetting/*, accountSetting/*)	shared_prefs/(optiondata.xml)	userdata\data\com.skt.aicloud.speaker.service\shared_prefs\(AICloud.xml)
	KT	gbas.megatvdnp.co.kr (/user*,/devList/*) gsvr.ktipmedia.co.kr (/devUserList/{USER_#}/*)	shared_prefs/(*.xml)	data\com.kt.gigagenie.launcher\databases\ (launcherCommon.db)
Time: Hard to deduce a specific command, but relevant information to build an event timeline. (e.g., use time, boot time, end time, package usage history, and alarm setting history)	NAVER	prod-ni-cic.clova.ai (/meta/*)	app_webview/(*)	root\system\notification_log.db root\misc\bootstat\last_boot_time_utc.txt
	KAKAO	app.i.kakao.com (/result/result/*) app.i.kakao.com /alarms/{alarm_#}	databases/ (com.kakao.kinsight.sdk.android.~.sqlite)	data\com.android.providers.media\databases\(external.db)
	SKT	pif.t-aicloud.com (/clientStatus/*)	databases/(aladdin.db)	system\usagestats\(usage-history)
	KT	gdialog.ktipmedia.com	files/(dxshield.sys)	data\com.kt.gigagenie.tts\shared_prefs\ (com.kt.gigagenie.tts.xml) system\(appops.xml)
History: Information about the command history that can be used to infer the user's command (e.g., cookie data, webview data, cache image, event recording)	NAVER	prod-ni-cic.clova.ai (/result/historyQuery/*)	cache/clova/(*.mp3) cache/org.chromium.android_webview/(*_0, *_1) cache/image_manager_disk_cache/(*.0)	[deleted] nvoice_{hash}.mp3
	KAKAO	kinsight-event.kakao.com (/sessions/events/*,/sessions/headers/*)	**-**	**data\com.android.providers.media\databases\(external.db)** **media\0\KakaoICache\audio\(cached.{hash}.mp3)**
	SKT	—	databases/(aladdin.db) cache/image_manager_disk_cache/(*.0) app_webview/(*)	data\com.skt.aicloud.speaker.service\databases\ (AladdinGeneral.db)
	KT	gdialog.ktipmedia.com	**cache/picasso-cache/(*.0, *.1)**	system\recent_tasks\(#_task.xml) system\recent_images (#_task_thumbnail.png)

References

Boztas, Abdul, Riethoven, A.R.J., Roeloffs, Mark, 2015. Digit. Invest. 12, S72–S80.

Burkholder, Peter, 2002. SSL Man-In-The-Middle Attacks. The SANS Institute.

Callegati, Franco, Walter, Cerroni, Ramilli, Marco, 2009. IEEE Secur. Priv. 7 (1), 78–81.

Chavez, Nicole, 2017. Arkansas Judge Drops Murder Charge in Amazon Echo Case. CNN.

Chung, Hyunji, Park, Jungheum, Lee, Sangjin, 2017. Digit. Invest. 22, S15–S25.

Desnos, Anthony, Gueguen, Geoffroy, 2011. Android: from reversing to decompilation. In: Proc. of Black Hat Abu Dhabi.

Dupuy, E., JD-Gui, 2012. Yet Another Fast Java Decompiler.

Gartner, 2018. What's Ahead for AI, Smart Speakers and Smartphones?.

Hassan, Carma, 2017. Voice-activated Device Called 911 during Attack, New Mexico Authorities Say. CNN.

Immanuel, F., Martini, B., Choo, K.K.R., 2015. IEEE Trustcom/BigDataSE/ISPA 1, 1094–1101.

Jo, WooYeon, Chang, Hyunsoo, Shon, Taeshik, 2018. J. Supercomput. 74 (8), 3704–3725.

Kebande, Victor R., Ray, Indrakshi, 2016. A Generic Digital Forensic Investigation Framework for Iot. FiCloud.

Lee, Seokjun, et al., 2019. ExtSFR : Scalable File Recovery Framework Based on Ext File System.

Lessard, Jeff, Kessler, Gary, 2010. Android Forensics. Simplifying Cell Phone Examinations.

Lin, Feng-Yu, Huang, Chien-Cheng, Chang, Pei-Ying, 2015. Forensic Sci. Int. 255, 64–71.

Oriwoh, Edewede, et al., 2013. Collaborative Computing: Networking, Applications and Worksharing, pp. 608–615.

Osborne, Mark, 2018. Judge Orders Amazon to Hand over Echo Recordings in Double Murder Case. ABC News.

Paller, Gabor, 2012. Dalvik Opcodes. http://pallergabor.uw.hu.

Perumal, Sundresan, Norwawi, Norita Md, Raman, Valliappan, 2015. ICDIPC 19–23.

Soghoian, C., Stamm, S., 2011. In: International Conference on Financial Cryptography and Data Security, pp. 250–259.

Song, Su-hyun, 2018. South Korea's Top Telco SKT Losing Market Share. The Korea Herald.

Spreitzenbarth, Michael, 2013. Dissecting the Droid: Forensic Analysis of Android and its Malicious Applications.

Winsniewski, R., 2012. Android-apktool: A Tool for Reverse Engineering Android Apk Files.

Wong, D.J., 2018. Ext4 Disk Layout. Ext4 Wiki, 2016.

Wooyeon Jo Wooyeon Jo received the B.S. degree in computer engineering from Ajou University, Suwon, Republic of Korea, in 2015. Since 2015, he is in the M.S/Ph.D. integrated program, Ajou university. His research interest includes the development of digital forensic tools and techniques that utilize metadata from various file systems, network forensics, extension to digital forensic science, digital forensics applied on control systems.

Yeonghun Shin Yeonghun Shin received the B.S. degree in cyber security from Ajou University, Suwon, Republic of Korea, in 2019. Since 2019, he is in the M.S/Ph.D. integrated program, Ajou university. His research interest includes digital forensics for Filesystems, Network, IoT devices, Mobile devices and AI Speakers.

HyungChan Kim HyungChan Kim is an undergraduate at the Division of Cyber Security, Ajou University. His research interest includes the development of digital forensic tools and techniques that utilize file systems, network forensics, mobile forensics, AI Speaker forensics.

Dongkyun Yoo Dongkyun Yoo is an undergraduate student at Division of Software Engineering, Ajou University. He graduated Korea Digital Media High School, Department of Hacking Defense. While he working for CERT(Computer Emergency Response Team) in Ministry of National Defense, He researched spam detection, prevention and its reverse-engineering. He also researched SEO(Search engine optimization) for NAVER while he working at company. He is interested in network forensics, web security, application analysis, embedded devices, machine learning and .NET C# programming. His research interests include Digital Forensics and Application Reverse-engineering.

Donghyun Kim Donghyun Kim was an undergraduate at the Division of Cyber Security, Ajou University. His research interest includes digital forensics and intrusion prevention. He had won the Volatility Analysis Contest 2018. He joined the M-Secure in 2018 and is currently working as a R&D engineer and contributing to open sources related to digital forensics, developing the Autopsy module.

CheulHoon Kang CheulHoon Kang works at the Supreme Prosecutor's Office, Seoul in Korea. he is Team Manager and Senior Investigator in Digital Forensic Division. he received his master's degree in Computer Engineering from Yonsei University. He is currently a team manager of Digital Forensic center, senior investigator, Supreme Prosecutors' Office.

Jongmin Jin Investigator Jongmin Jin received his master's degree in Digital Forensic from Seoul University, Seoul, Korea in 2017. He is currently working as a database forensic investigator at Supreme Prosecutors' Office NDFC(National Digital Forensic Center) for four years.

Jungkyung Oh Jungkyung Oh received a law degree from Hongik University in 2013. He joined the Supreme Public Prosecutor's Office in 2013. he has been working as a digital forensic investigator. He was in charge of mobile forensics from 2015 to 2016, and from 2017 to present, he was in charge of database forensics.

Bitna Na Bitna Na is a prosecutor working at the Supreme Prosecutor's Office in South Korea since 2015. She Joined the Digital Forensic team in 2018 and currently carries out research tasks on database forensics.

Taeshik Shon Dr. Taeshik Shon received his Ph.D. degree in Information Security from Korea University, Seoul, Korea in 2005 and his M.S. and B.S. degree in Computer Engineering from Ajou University, Suwon, Korea in 2000 and 2002, respectively. While he was working toward his Ph.D. degree, he was awarded a KOSEF scholarship to be a research scholar in the Digital Technology Center, University of Minnesota, Minneapolis, USA, from February 2004 to February 2005. From Aug. 2005 to Feb. 2011, Dr. Shon had been a senior engineer in the Convergence S/W Lab, DMC R&D Center of Samsung Electronics Co., Ltd. He is a visiting professor of Electrical Computer Engineering Department at Illinois Institute of Technology, Chicago, USA, in 2017. He is currently a professor at the Division of Cyber Security, College of Information Technology, Ajou University, Suwon, Korea. He was awarded the Gold Prize for the Sixth Information Security Best Paper Award from the Korea Information Security Agency in 2003, the Honorable Prize for the 24th Student Best Paper Award from Microsoft-KISS, 2005, the Bronze Prize for the Samsung Best Paper Award, 2006, the Second Level of TRIZ Specialist certification in compliance with the International TRIZ Association requirements, 2008, and the Silver, Bronze, Excellent Publication Prize for Ajou University Award, 2013, 2014, 2016. He is a senior member of IEEE and also serving as a guest editor, an editorial staff and review committee of Computers and Electrical Engineering - Elsevier, Mobile Network & Applications - Springer, Security and Communication Networks - Wiley InterScience, Wireless Personal Communications - Springer, Journal of The Korea Institute of Information Security and Cryptology, IAENG International Journal of Computer Science, and other journals. His research interests include Industrial Control System, Anomaly Detection Algorithms, and Digital Forensics.