# Attacking the Baseband Modem of Mobile Phones to Breach the Users' Privacy and Network Security

**Christos Xenakis**
Department of Digital Systems
University of Piraeus
Piraeus, Greece
xenakis@unipi.gr

**Christoforos Ntantogian**
Department of Digital Systems
University of Piraeus
Piraeus, Greece
dadoyan@unipi.gr

**Abstract:** As people are using their smartphones more frequently, cyber criminals are focusing their efforts on infecting smartphones rather than computers. This paper presents the design and implementation of a new type of mobile malware, named (U)SimMonitor for Android and iPhone devices, which attacks the baseband modem of mobile phones. In particular, the mobile malware is capable of stealing security credentials and sensitive information of the cellular technology including permanent and temporary identities, encryption keys and location of users. The developed malware operates in the background in a stealthy manner without disrupting the normal operation of the phone. We elaborate on the software architecture of (U)SimMonitor and provide implementation details for the specific AT commands used by the malware. We analyse the security impacts of (U)SimMonitor malware and we show that it can entirely breach the privacy of mobile users and the security of cellular networks. In particular, a mobile user with an infected phone can be identified and all his/her movements can be tracked. Moreover, all his/her encrypted phone calls and data sessions can be disclosed.

***Keywords:*** *Mobile malware, Mobile networks, Android, Iphone, AT commands,*

# 1. INTRODUCTION

Cellular networks have been continuously evolving to support high data rates and provide internet access that can fulfill the demands of today's web applications [1]. Along with cellular networks, the mobile phones are also evolving to smartphones with processing capabilities and storage resources that are often equivalent to contemporary personal computers. The potential of smartphones is leveraged by mobile operating systems, such as iOS and Android OS that allow end-users to access traditional desktop applications using these portable devices. Along with the variety of new perspectives, smartphones also raise new security concerns and issues. In particular, due to their popularity, smartphones have become prime targets for malware. In 2013, 3.905.502 installation packages were used by cybercriminals to distribute mobile malware [2].

The majority of mobile malware aims at causing financial charges to infected mobile phones. For example, sending SMS messages to premium-rate numbers without the users' consent is a usual malicious activity of a mobile malware. These numbers can be either hardcoded in the malware code or downloaded at runtime to avoid detection. Other types of mobile malware collect sensitive data from the infected phone including SMS messages, phone numbers, email addresses and username/passwords from applications. Moreover, some infected phones are turned into bots for HTTP-based remote control by a botmaster. In general, we can observe that mobile malware target and exploit the characteristics of the mobile OS to perform a variety of malicious actions [3]. To the best of our knowledge, there is no mobile malware that targets the baseband modem of mobile phones to breach the privacy of mobile users and the security of cellular networks.

This paper presents the design and implementation of a new type of mobile malware, named (U)SimMonitor for Android and iPhone devices, which attacks the baseband modem of mobile phones. In particular, it is a mobile malware capable of stealing security credentials and sensitive information of the cellular technology (i.e., permanent and temporary identities, encryption keys, location of users, etc.) and profoundly compromising the privacy of users and the mobile network security. The developed malware (i.e., (U)SimMonitor) operates in the background without the victim user noticing its existence, since it does not disrupt the normal operation of the phone. We elaborate on the software architecture of (U)SimMonitor and provide implementation details for the specific AT commands used by the malware. We analyse the security impacts of (U)SimMonitor malware and we show that it can entirely breach the privacy of mobile users and the security of cellular networks. In particular, a mobile user with an infected phone can be identified and all his/her movements can be tracked. Moreover, all his/her encrypted, by the cellular technology, phone calls and data sessions can be disclosed. The criticality of this malware is evident: it eliminates the need of breaking the security of the employed cryptographic algorithms, since the encryption keys are in the possession of the attacker. Thus, this malware comprises

a threat for all mobile networks technologies, even for the security-enhanced long term evolution (LTE) networks, since it renders inadequate all possible security measures that can be employed in cellular networks. We believe that mobile antivirus products should update their signatures to detect (U)SimMonitor malware and its variants. Overall the contributions of this paper are:

- Identify and analyze the security criticality of a new type of mobile malware named (U)SimMonitor, which is capable of stealing security credentials and sensitive information of mobile users and cellular networks and profoundly compromising the privacy of users and the network security.
- Design and implementation of the (U)SimMonitor that proves the feasibility of this new attack vector.
- Release the source code of (U)SimMonitor and practical demonstration.

The remainder of this article is organized as follows. Section 2 presents briefly the cellular network technology and the related work. Section 3 presents the design and implementation of (U)SimMonitor. Section 4 elaborates on the security impacts of (U)SimMonitor and how it can entirely breach the privacy of users. Finally, section 5 concludes the article.

# 2. BACKGROUND

## *A. Cellular Network Technology and Android*

Cellular networks are composed of various interworking technologies including 2G and 3G networks [4]. A basic element of cellular networks is the mobile station (MS), which enables a user to connect to a serving network and enjoy services. MS includes the user's equipment (UE) and a subscriber's service identity module (SIM) or UMTS SIM (USIM) card. The latter is an integrated circuit that stores various parameters of the mobile network, including the international mobile subscriber identity (IMSI), which is the permanent identity of a subscriber in a mobile network as well as encryption and integrity keys.

The UE of MS is a smartphone that runs a mobile OS. The most prominent mobile OS is Android having an 81% market share in the third quarter of 2013 [5]. Android applications are implemented with Java programming language and executed in their own virtual machine named Dalvik. The latter relies on the Linux kernel for the underlying OS functionality, such as threading and low-level memory management. Typically, in a smartphone there are two processors: the application processor that is used to run Android OS and the baseband modem processor, where all the radio operations take place. In modern phones, these processors and all other peripheral devices are integrated into one piece of hardware (i.e., System on a Chip (SoC)).

### *B. Related Work*

The related work in this research area focuses mainly on the defensive side, proposing solutions that detect or prevent mobile malware from infecting mobile devices. In particular, several works propose security enhancements in mobile platforms that perform fine-grained access control of system resources when they are accessed by untrusted third party applications [3]. Moreover, many past works put their efforts in detecting mobile malware by applying machine learning algorithms [6].

On the other hand, there are very few papers that elaborate on AT commands and their important functionality in mobile phones. In the work closest to ours [7], the authors analyse theoretically the potential of cellular botnets that can perform a coordinated and distributed denial of service (DDOS) attack to a Home Location Register/Authentication Centre (HLR/AuC). The analysed DDOS attack is performed by a malware that can initiate appropriate AT commands that trigger network-oriented activities (e.g., location update). However, the authors have overlooked to analyse the specific AT commands that are required to perform the proposed DDOS attack. Moreover, the authors do not elaborate on the design and implementation of the mobile malware to perform AT commands. Thus, the feasibility of this malware and the related DDOS attack is not proved.

In our previous work [8], we have presented an advanced persistent threat (APT) in 3G networks that exploits a series of zero-day vulnerabilities to flood the HLR/AuC, leading to system saturation. It was proven that the discovered APT can be performed in a trivial manner using commodity hardware and software. To this end, a mobile application was implemented that performs continuous network registrations using AT commands. The application uses the dial command to initiate phone calls using a different IMSI for each call request. This was achieved using a device named simtrace [9], which acts as an active man in the middle between the modem and SIM/USIM card and can change the IMSI identity when it is requested by the modem.

In [10], the authors utilize AT commands from a different point of view: that is, they use AT commands in order to perform SMS fuzzing for iPhone, Android and Windows mobile phones. Their goal was to discover previously unknown software bugs in SMS applications that can be exploited by malicious actors to perform DOS attacks. The authors successfully discovered a set of critical bugs in both iOS and Android SMS applications. Moreover, [11] analyses the design and implementation of a passive man-in-the-middle application for iOS and Android phones that listens the communication between the radio interface layer (RIL), which is a software middleware that controls modem through AT commands, and the modem. In this way, [11] achieved to log all the invoked AT commands by the RIL and the modem during phone calls, SMS sending/receiving, etc. Apart from these works, we have discovered a free online tool named AT command tester

[12], which is implemented in Java, and allows the execution of a comprehensive set of AT commands to GSM modules via a web browser.

Finally, we mention here that there are many commercial and free mobile applications for Android and iPhone devices such as [24] that can listen and record voice calls or even stream in real time the intercepted calls to the malicious actor. (U)SimMonitor is a new, alternative way to intercept phone calls by targeting the baseband modem and extracting the GSM and 3G encryption keys. Thus, the proposed malware is not only able to decrypt voice calls, but also the Internet traffic of the victim.

# 3. (U)SIMMONITOR

## *A. Overview*

In this section we present and analyze the architecture and the key functionality of (U)SimMonitor for the Android OS. Implementation details are presented in [21], while the source code of (U)SimMonitor can be found in [22]. It is important to mention that we have also developed successfully a similar malware application for the iOS operating system of iPhones. The main purpose of (U)SimMonitor is to extract security related data from SIM and USIM cards [13]. To achieve this, it communicates with the modem of the mobile phone through a set of AT commands. This procedure is executed, periodically, at specific time intervals or based on various events, as analyzed below. (U)SimMonitor stores the fetched data from the modem in a local database on the phone and periodically or on-demand it uploads the stored data to a server for further processing and analysis. The malware runs in the background, while the user can normally operate his/her phone. To this end, the (U)SimMonitor uses the least possible resources of the modem, in order to avoid blocking accidently a voice/data communication. In general, (U)SimMonitor has been designed to collect data transparently, without disrupting the proper operation of the phone. Thus, it can hide its malicious activities and avoid detection, due to its stealthy nature.

Moreover, (U)SimMonitor stops and restarts the RIL daemon when it executes an AT command to avoid possible disruptions from the Android. More specifically, the functionality of the RIL daemon is to provide the interface that handles the communication between the Android phone framework services and the radio hardware [14]. During our tests, we observed that initially (U)SimMonitor was not able to communicate directly with the modem through AT commands. After investigation, we discovered that some vendors implement RIL in a way that the modem is able to respond only to one process at a time. For this reason, the (U)SimMonitor could not execute AT commands to the modem, since the latter was always in use by Android. To overcome this limitation, the (U)SimMonitor incorporates a payload that stops the RIL daemon before initiating the execution of AT commands and restarts it immediately after the modem responds to the last AT

command. We remark here that during our experiments and usage of the (U)SimMonitor, the normal operation of the phone was not affected by stopping and starting the RIL daemon, since this procedure (i.e., restarting the RIL daemon) is executed in under one second (<1 sec) without the user receiving any notification.

Apart from its main functionality (i.e., extracting security data), the (U)SimMonitor incorporates a dropper payload for privilege escalation. More specifically, for security reasons, Android and iOS do not allow the execution of applications with root permissions. However, (U)SimMonitor is able to execute AT commands only if it has root privileges. To overcome this restriction, (U)SimMonitor includes a dropper payload, which essentially downloads binary code that exploits known vulnerabilities in Android and iOS, in order to elevate privileges. The downloaded exploitation code is obfuscated, in order to avoid detection from mobile AV [27].

### *B. AT Commands*

AT commands lie at the core of (U)SimMonitor providing various operations to control a modem, as specified in 3GPP TS 27.007 [15]. Based on the provided functionality, AT commands can be categorized as follows:

- Call control: commands for initiating and controlling calls.
- Data call control: commands for controlling the data transfer and the Quality of Service (QoS).
- Network services control: commands for supplementary services, operator selection, locking and registration.
- SMS control: commands for sending, notifying of received SMS messages, and configuring SMS services.
- Data retrieval: commands to obtain information for the subscriber and the phone, such the IMSI, the IMEI, radio signal strength, batter status. etc.

The (U)SimMonitor makes extensive use of the last category of AT commands (i.e., data retrieval) to extract security related data from the SIM/USIM. A summarizing list of the AT commands, which we used to obtain security related data as well as their proper syntax, is presented in Appendix. In all our testing mobile devices we have successfully installed and executed (U)SimMonitor. These devices are:

- Samsung S-5500
- Samsung S-6500
- Samsung Galaxy s2
- ZTE Blade
- HTC Sensation XE with Beats Audio
- Sony Ericsson Xperia LT18i

### *C. Data Collection*

(U)SimMonitor collects sensitive and security related data [16][17], which are extracted through AT commands. These data are briefly presented below:

**IMSI**: The international mobile subscriber identity (IMSI) is a unique number permanently associated to the holder of the SIM/USIM card. Its size is 8 bytes. The first three bytes of IMSI represent the mobile country code (MCC), while the next two or three bytes represent the mobile network code (MNC). The remaining bytes represent the mobile subscriber identification number (MSIN).

**$K_c$**: A 64 bit ciphering key used to encrypt voice and data communication between the MS and BTS of GSM networks [18].

**$K_{cGPRS}$**: A 64 bit ciphering key used to encrypt communication data between the MS and the SGSN of GPRS networks.

**CK**: A 128 bit ciphering key used to encrypt the communication between the MS and the RNC of UMTS.

**IK**: A 128 bit key to protect the integrity of the signaling data between the MS and the RNC of UMTS network.

**Threshold**: A 24 bit value which represents the lifetime of the CK and IK keys in UMTS networks.

**Ciphering Indicator**: This is a 1 bit flag that allows the MS to detect whether ciphering is switched on (flag set to 1) or off (flag set to 0). The ciphering indicator feature may be disabled by the mobile network operator.

**TMSI**: The temporary mobile subscriber identity (TMSI) is a temporary identity of MS, which is assigned from the mobile network and it is used instead of IMSI for enhancing anonymity. TMSI is valid for circuit switching (CS) domain and its size is 4 bytes.

**TMSI Time**: This is a 1 byte value and represents the maximum time interval which the assigned TMSI can be used.

**P-TMSI**: The packet TMSI (P-TMSI) is the complement of TMSI in the UTRAN/GERAN packet switching (PS) domain.

**P-TMSI Signature value**: This is a signature used by the 3G network for verifying the validity of P-TMSI of MS. Its size is 3 bytes.

**LAI**: The location area identity (LAI) is a 5 bytes unique identifier for each location area in the CS domain. It consists of MCC, MNC and the location area code (LAC).

**RAI**: The Routing Area Identity for PS domains is the analogous to the LAI for CS domains. RAI consists of LAI (which is 5 bytes) and a 1 byte Routing Area Code.

**Provider**: This is the name of the mobile network operator.

**Cell Id**: This is the unique identity of the cell tower, where the MS is connected at the moment of data collection.

**Network type**: This parameter indicates the mobile network technology, where the MS is connected, at the moment of data

collection. It may have several values including GPRS, EDGE, UMTS, HSDPA, LTE, UNKNOWN, etc.
**Roaming**: A 1 bit value that indicates whether MS is outside the coverage area of its home network.

Moreover, (U)SimMonitor collects some additional metadata as mentioned below:

**Event Type**: This value indicates the event that triggered the data collection. The possible event types are: i) Outgoing or incoming calls, ii) Screen on or off, iii) Power on or off, iv) Periodic (i.e., a time interval where data is collected periodically).
**Latitude, Longitude**: These values are the coordinates of the geographical position of MS at the moment of data collection. The coordinates are determined either by the GPS sensor of the phone or the Wifi signals.
**Timestamp**: The date and time of data collection

## *D. Software Architecture*

As shown in figure 1, the software architecture of (U)SimMonitor consists of five units, each one undertaking a specific task. More specifically, these units are as follows:

1. Metadata collection
2. Event listener
3. Data collection
4. Data parsing
5. Data upload

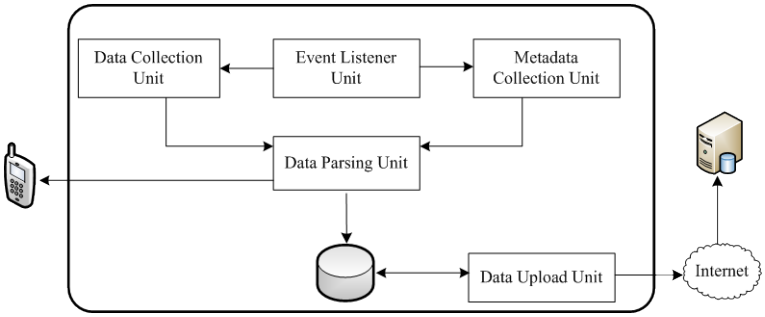


Figure 1. (U)SimMonitor application architecture

The event listener unit monitors and captures the occurrence of an event. Possible event types are: i) Outgoing or incoming calls, ii) Screen on or off, iii) Power on or off, iv) Periodic (i.e., a time interval where data is collected periodically). When one of these events occurs, the event listener unit triggers the metadata collection

and data collection units. The metadata collection unit obtains the coordinates of the smartphone using the GPS sensor or WiFi signals as well as the time that data extraction occurred. On the other hand, the data collection unit communicates with the modem executing AT commands. To achieve this, it creates a system process to invoke a Linux shell script. The latter communicates with the baseband modem by executing sequentially a set of AT commands. For each AT command, the baseband modem contacts to USIM/SIM to obtain the related data (see figure 2). After receiving the response of the last executed AT command, the data collection unit terminates the system process in order to save memory resources.

Both the data collection unit and the metadata collection units transfer the obtained data to the data parsing unit. The latter filters out unnecessary information and stores the final data in a local database. Optionally, the parsing unit can also display the final data in the phone's screen. Figure 3 shows an Android phone and an iPhone displaying extracted data using (U)SimMonitor. Finally, as its name implies, the upload unit transfers the database contents to a secure server via SSH and subsequently deletes the contents of the database to save memory space in the phone.

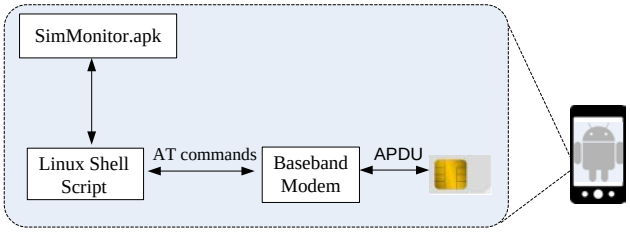


Figure 2. (U)SimMonitor execution flow


Figure 3. (U)SimMonitor displaying collected data in Android and iOS

# 4. SECURITY IMPACTS

The (U)SimMonitor introduces a new type of mobile malware for Android and iPhone devices. In particular, it is able to steal security credentials and sensitive information of the cellular technology networks (i.e., IMSI, TMSI, keys, LAI, RAI, Cell Id, etc.) and profoundly compromise the privacy of users and the mobile network security. The malware (i.e., (U)SimMonitor) is capable of operating in the background without the victim noticing its existence, since it does not disrupt the normal operation of the phone. Thus, the (U)SimMonitor can hide its malicious activities and avoid detection, due to its stealthy nature.

An attacker first should entice his/her victims to install and execute (U)SimMonitor without their permission and without raising any suspicion. There are many ways an attacker can achieve this. For example, an attacker can inject malware from a PC to a mobile Device using the USB port [27] or through advertising banners embedded in mobile applications [26]. However, the most common way is by injecting the malware functionality into a legitimate Android or iPhone application (i.e., Trojan application). In particular, the attacker first locates and downloads a popular mobile application. Next, he/she re-package the application by enclosing also the functionality of (U)SimMonitor. This procedure is also known as binding and there are freely available tools to perform it [25]. Finally, he/she submits the infected application to third party application markets. Using social engineering techniques, the attacker can lure his/her victims to download and install the infected application into his/her mobile phone.

After activation of the (U)SimMonitor in the victim's phone, the malware reads security related and sensitive data from USIM/SIM card, including the encryptions keys used in the mobile network (Kc, KcGPRS, CK) together with the TMSI/IMSI identities, the network operator of the user, the Location/Routing area and the Cell ID. Note that the malware can also extract geographical coordinates using GPS, but in order to remain stealthy it may avoid this action. The extracted data is uploaded to a server, which is deployed from the attacker. At this point the attacker can perform the following malicious actions:

- He/she can easily identify the victim user, since he/she has obtained the IMSI and TMSI identities, while using the location/routing area and Cell-ID parameters he/she can approximately track victim's movements.
- Disclose phone calls and data session of the victim user using the obtained encryption keys (i.e., Kc, KcGPRS, CK), regardless of the strength of the employed cryptographic algorithm. It is evident that first the attacker should capture the mobile communications of the victim.

The security criticality of the malware is related to the fact that it eliminates the need of breaking the security of the employed cryptographic algorithms, since the encryption keys are in the possession of the attacker. Thus, this new generation of malware comprises a threat for all mobile network technologies, even for the

security enhanced LTE networks, since it renders inadequate all possible security measures that can be taken from the mobile operator.

To further elaborate on the malware characteristics of the (U)SimMonitor, we tested five popular mobile antivirus (AV) products whether they are capable of recognizing it as a virus. In particular, we tested the following mobile AVs:

- Norton Mobile Security Lite
- Kaspersky Internet Security
- Avast Mobile Security & Antivirus
- TrendMicro Mobile Security
- Zoner AntiVirus Free

We installed the above AV applications in a Samsung Galaxy S3 mobile phone running Android OS 4.2. Prior to scanning, we updated the AVs with their latest virus database as of August 2014. Unfortunately, none of the tested AVs raised an alarm. This result comes as no surprise, since the detection capabilities of mobile AVs are far lower that their desktop counterparts [19]. Therefore, mobile AVs should update their signatures to identify pattern strings that include AT commands, as they are thoroughly presented in the Appendix.

We believe that (U)SimMonitor should be also viewed as a proof of concept for the hidden dangers lurking by rooting mobile phones. As mentioned in section 3.A, (U)SimMonitor incorporates an extra payload (i.e., dropper) to download exploit code, in order to elevate root-level privileges. However, (U)SimMonitor may bypass the execution of the dropper if the malware infects an already rooted device and automatically obtain root privileges. Nowadays, the rooting procedure seems to be a common practise among many smartphone owners. In fact, the rooting procedure in many phones has been simplified into a one-click procedure [20], which allows even non-technically aware users to perform rooting. On the one hand, rooting allows users to remove vendors' software and install newer versions of android OS. On the other hand, gaining root access also entails circumventing the security restrictions put in effect by the Android and iOS operating system. This last observation seems to be often overlooked. That is, many mobile phone owners are not aware the fact that by rooting their phones, they are exposed to more threats. Thus, by analysing the security impacts of (U)SimMonitor and its potential for new attacks, we believe that this work should be also viewed as a warning of the subtle security implications of rooting mobile devices.

Having access to all the security related information and parameters of a mobile subscriber connected to a cellular network, (U)SimMonitor cannot only be employed for malicious (i.e., black hat) usage. On the contrary, it can be used to capture and analyze the security policy that a cellular operator enforces i.e., the invocation and employment of the specified security measures to protect its users, a functionality which is currently missing from Android and iPhone devices [28]. In particular, the (U)SimMonitor can inform the mobile users if ciphering is

disabled, how often the encryption keys are refreshed and how often the temporary identities are updated. In this way, mobile users can have a better view of the provided level of security, while security researches can perform a quantitative risk assessment for mobile networks.

# 5. CONCLUSIONS

In this paper we presented the design and implementation of a new type of mobile malware, named (U)SimMonitor for Android and iPhone devices. The malware targets the baseband modem of mobile phones and extracts security credentials and sensitive information from SIM/USIM cards using AT commands. The malware compromises entirely the privacy of mobile users and the security of cellular networks. That is, after infection an attacker can perform the following malicious actions:

- Identification and tracking of victim's movements using the IMSI/TMSI identities as well as the location/routing areas and the Cell ID parameters.
- Disclosure of voice calls and data connections using the extracted encryption keys of 2G and 3G mobile networks.

The criticality of the malware is evident: it eliminates the need of breaking the security of the employed cryptographic algorithms, since the encryption keys are in the possession of the attacker. Thus, this new generation of malware comprises a threat for all mobile network technologies, even for the security enhanced LTE networks, since it renders inadequate all possible security measures that can be taken from the mobile operator. Finally, we believe that mobile AVs should update their signatures and heuristic algorithms to identify pattern strings that include AT commands.

# 6. APPENDIX

In this section we present the AT commands that the (U)SimMonitor uses to extract data from the (U)SIM card. AV products can use the syntax of the following AT commands as signatures for their virus databases.

The exact syntax of AT Commands depends on their type. We can recognize two main types of AT commands:

- Basic commands are AT commands that do not start with "+", such as D (Dial), A (Answer), H (Hook control), and O (Return to online data state).
- Extended commands are AT commands that start with "+" and their main functionality is to retrieve data from (U)SIM cards.

(U)SimMonitor uses AT commands from the second category (i.e., extended). In particular, the most useful and frequently invoked AT command of (U)SimMonitor is +CRSM, which extracts various mobile network parameters from (U)SIM cards.

A generic format for the +CRSM command invoked by the (U)SimMonitor is the following one:

AT+CSRM=x, y, p1, p2, w

The value of parameter x indicates whether the command will write to or read data from SIM/USIM card. Since the (U)SimMonitor only extracts data, the value of x is always equal to "176", which indicates a READ operation. The value of y is an identifier for the type of data that we want to extract from the SIM and USIM card. For example, the identifier of IMSI for SIM and USIM cards is the value "6F07". The values of p1, p2 represent the high and low order offset respectively (in terms of number of bytes) from the beginning of the identifier that we want to read or write data. In (U)SimMonitor both values of p1, p2 were both equal to 0 indicating no offset. Finally, the value of w indicates the number of bytes that the specific AT command wants to read or write.

Apart from CSRM, (U)SimMonitor also uses the commands COPS to extract the name of the operator and CREG to extract the LAC and the Cell ID. In the following table, we provide the exact syntax of the AT commands as they are invoked by (U)SimMonitor and their respective functionality.

Table 1: AT commands used in (U)SimMonitor

| Functionality | Storage location in SIM and USIM cards | AT command |
|---|---|---|
| 1. Extraction of IMSI | Stored in 6F07 (decimal 28423) for SIM and USIM | (SIM/USIM) AT+CRSM=176,28423,0,0,3 |
| 2. Extraction of Ciphering Indicator | Stored in 6FAD (decimal 28589) for SIM and USIM | (SIM/USIM) AT+CRSM=176,28589,0,0,3 |
| 3. Extraction of Ciphering Key Kc | Stored in 6F20 (decimal 28448) for SIM and 4F20 (decimal 20256) for USIM | (SIM) AT+CRSM=176,28448,0,0,9<br><br>(USIM) AT+CRSM=176,20256,0,0,9 |
| 4. Extraction of Ciphering Key KcGPRS | Stored in 6F52 (decimal 28498) for SIM and 4F52 (decimal 20306) for USIM | (SIM) AT+CRSM=176,28498,0,0,9<br><br>(USIM) AT+CRSM=176,20306,0,0,9 |
| 5. Extraction of Ciphering Key CK and Integrity Key IK | Stored in 6F08 (decimal 28424), applied to USIM only | (USIM) AT+CRSM=176,28424,0,0,33 |
| 6. Extraction of TMSI, TMSI TIME and LAI | Stored in 6F7E (decimal 28542) for SIM and USIM | (SIM/USIM) AT+CRSM=176,28542,0,0,11 |
| 7. Extraction of PTMSI, PTMSI | Stored in 6F53 (decimal 28499) for SIM and 6F73 | (SIM) AT+CRSM=176,28499,0,0,14 |

| | | |
|---|---|---|
| Signature Value, RAI and RAUS | (decimal 28531) for USIM | (USIM) AT+CRSM=176,28531,0,0,14 |
| 8. Extraction of THRESHOLD | Stored in 6F5C (decimal 28508), applied to USIM only | (USIM) AT+CRSM=176,28508,0,0,3 |
| 9. Extraction of Provider | - | AT+COPS? |
| 10. Extraction of Lac and Cell ID | - | AT+CREG? |

## REFERENCES


[1] A.T. Kearney, The mobile economy, GSMA, 2013.

[2] https://www.securelist.com/en/analysis/204792326/Mobile_Malware_Evolution_2013

[3] Yajin Zhou, Xuxian Jiang, "Dissecting Android Malware: Characterization and Evolution," Proceedings of the 33rd IEEE Symposium on Security and Privacy (Oakland 2012), San Francisco, CA, May 2012.

[4] Christos Xenakis, Lazaros Merakos, "Security in third Generation Mobile Networks," Computer Communications, Elsevier Science, Vol. 27, No. 7, pp. 638-650, May 2004.

[5] StrategyAnalytics,http://blogs.strategyanalytics.com/WSS/post/2013/10/31/Android-Captures-Record-81-Percent-Share-of-Global-Smartphone-Shipments-in-Q3-2013.aspx

[6] Brandon Amos, Hamilton A. Turner, Jules White, "Applying machine learning classifiers to dynamic Android malware detection at scale", 9th IEEE International Wireless Communications and Mobile Computing Conference (IWCMC 2013), Italy, July 2013

[7] Patrick Traynor, Michael Lin, Machigar Ongtang, Vikhyath Rao, Trent Jaeger, Patrick Drew McDaniel, Thomas F. La Porta, "On cellular botnets: measuring the impact of malicious devices on a cellular network core", ACM Conference on Computer and Communications Security (CCS 13), 223-234, Berlin, Germany, November 2009.

[8] Christos Xenakis, Christoforos Ntantogian, "An advanced persistent threat in 3G networks: Attacking the home network from roaming networks," Computers & Security, Elsevier Science, Vol. 40, Issue 1, pp:84-94, February 2014.

[9] Simtrace, http://bb.osmocom.org/trac/wiki/SIMtrace

[10] Collin Mulliner, Charlie Miller, "Fuzzing the Phone in your Phone", Black Hat USA 2009.

[11] Fabien Sanglard, "Tracing the broadband: Part 1 and Part 2", http://fabiensanglard.net/cellphoneModem/

[12] AT module tester, http://m2msupport.net/m2msupport/module-tester/

[13] ETSI TS 102 221 V9.0.0 (2010-02), Smart Cards, UICC-Terminal interface, Physical and logical characteristics (Release 9).

[14] Android Platform Development Kit, Radio Layer Interface, Netmite, http://www.netmite.com/android/mydroid/development/pdk/docs/telephony.html

[15] 3GPP TS 27.007 V11.5.0 (2012-12), 3rd Generation Partnership Project, Technical Specification Group Core Network and Terminals, AT command set for User Equipment (UE) (Release 11).

[16] 3GPP TS 35.201, 3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; 3G Security; Specification of the 3GPP Confidentiality and Integrity Algorithms; Document 1: f8 and f9 Specification (Release 9), 2009.

[17] 3GPP TS 33.102, 3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; 3G Security; Security architecture (Release 9), 2009.

[18] Karsten Nohl, "Attacking phone privacy", BlackHat USA, Las Vegas, August 2010.

[19] Vaibhav Rastogi, Yan Chen, Xuxian Jiang, “DroidChameleon: Evaluating Android Anti-malware against Transformation Attacks,” Proceedings of the 8th ACM Symposium on InformAtion, Computer and Communications Security (ASIACCS 2013), Hangzhou, China, May 2013.

[20] OneClickRoot, http://www.oneclickroot.com/

[21] Dimitrios Raptodimos, “Design and implementation of an Android application for extraction of security related data from SIM/USIM”, MSc thesis, University of Piraeus, http://digilib.lib.unipi.gr/dspace/bitstream/unipi/5851/1/Raptodimos.pdf

[22] https://github.com/SSL-Unipi/U-SIMonitor

[23] Lucas Davi, Alexandra Dmitrienko, Ahmad-Reza Sadeghi, Marcel Winandy, “Privilege Escalation Attacks on Android.,” Proceedings of the 13th international conference on Information security, (ISC 10), Florida, USA, 2010.

[24] http://www.flexispy.com/

[25] https://github.com/funsecurity/apk_binder_script

[26] Malwarebytes unpacked, https://blog.malwarebytes.org/mobile-2/2014/10/mobile-advertisers-use-malware-tricks-to-get-installs/

[27] Rafael Fedler, Julian Schütte, Marcel Kulicke, “On the Effectiveness of Malware Protection on Android”, April 2013, Technical Report, Fraunhofer AISEC.

[28] Iosif Androulidakis, Dionisios Pylarinos, and Gorazd Kandus, “Ciphering Indicator approaches and user awareness”, Maejo International Journal of Science and Technology, Vol. 6, No 03, pp: 514-527, 2012.