Tufts University

School of Engineering

Department of Electrical and Computer Engineering

COMP 116 – Intro to Computer Security
Fall 2013

A Security Analysis of the GSM protocol

Name:	Nicholas Davis
Date Due:	12/13/2013
Submitted to:	Ming Chow

Abstract

The Global System for Mobile Communications (GSM) is an international standard for cellular communications. Originally created in 1984, GSM is a digital network that far exceeded the analog networks at the time. GSM (also known as 2G) is becoming deprecated as we move onto better network technologies such as CDMA (3G) and LTE (4G), but it is still the underlying infrastructure of cellular networks here in the United States, and in Europe. For privacy and security, GSM institutes a number of security measures, such as a sessionID analog, encryption (A5/1). In this paper, I will present an analysis and methodology of cracking GSM to intercept calls and text messages (SMS) over-the-air.

Introduction:

Although more advanced mobile networks exist today, the basic infrastructure for some of the largest cellphone carriers in the United States. Namely, AT&T still uses GSM for SMS, MMS, and voice calls. In the age of always-connected devices we tend to forget about some of our legacy technologies. This forgetfulness tied with societies demand for functionality over security creates an ideal scenario for abusing the system. GSM uses a stream cipher called A5/1. This encryption method was presented as part of the original GSM standard in 1992 by the ETSI[1]. By 1994, Ross Anderson published a theory on cracking A5/1[2]. Anderson's theory grew in popularity and people began contributing to the theory, essentially breaking GSM encryption in 2003. The computational power would not catch up with the theory for some years however, and therefore GSM was still considered relatively secure.

By 2008 it was possible to break GSM in a practical application but due to responsible disclosure, the vulnerability was not disclosed in its entirety. Steve D. Hulton published a paper at Black Hat in February of 2008 titled *Intercepting GSM traffic*[3] in which he described how to scan GSM using a system that could be built for $900 USD. In the second half of the paper Hulton talked about a practical solution to cracking A5/1 encryption. This paper shows how insecure GSM really was even in early 2008, but price was a factor. Even if people would be willing to spend $900 in order to scan for GSM packets, cracking the encryption also required immense computing power and storage. In his approach, Hulton used an FPGA, and 2 TB of storage to brute force the encryption. Between the storage and the specialized hardware GSM was still out of reach of the majority of people. In this paper I plan to demonstrate how modern technology can be used to completely undermine privacy on GSM networks for very little cost.

To the Community:

---

[1] Principles of Telecommunication Services supported by a GSM PLM, ETSI http://www.etsi.org/deliver/etsi_gts/02/0201/03.02.00_60/gsmts_0201sv030200p.pdf

[2] A5 (Was: HACKING DIGITAL PHONES) http://yarchive.net/phone/gsmcipher.html

[3] Intercepting GSM Traffic, Black Hat Feb. 2008 http://www.blackhat.com/presentations/bh-dc-08/Steve-DHulton/Whitepaper/bh-dc-08-steve-dhulton-WP.pdf

Today, as computing power continues to rise and the cost per transistor continues to decline, GSM cracking is becoming a reality for the masses. So much infrastructure has been built on top of the GSM base, that cellphone carriers still use it even though it is known to be insecure. To add to the critical mass of this problem, 6.8 billion people have mobile cellular subscriptions in the world today[4]. That is up from 6.4 billion in 2012, and 5.9 billion in 2011. As the number of people who use cell phones increases, so too does the number of people relying on GSM for their communication. This adds ever more incentive for hackers to use GSM as an attack vector against targets. Personal privacy is a hot topic in today's society and the idea that for under $100 a dedicated civilian could listen in on the phone calls and text messages of others is a frightening scenario.

Applications:

In this paper I will focus on a single application; capturing GSM packets from a specifically identified device, and decrypting all communication to and from said device. This application is particularly insidious as it can be thought of a highly surgical man-in-the-middle attack. What makes this approach unique is the use of low-cost components and already existing software. Typically to capture GSM traffic you need USRP (universal software peripheral) antennas and components. USRP products as very expensive, however, which limits the availability of this attack. Instead of using USRP, I will be using an RTL-SDR[5] combined with Osmocom, and Gnu Radio. RTL-SDR is a cheap software defined radio (SDR) that uses the RTL2832U chipset. By combining Osmocom with the RTL-SDR, we can transfer raw I/Q samples to a host. I/Q samples are officially used in DAB/DAB+/FM demodulation, and therefore we can sniff down traffic with this cheap combination using GSM.

Now that the hardware environment is set up, we need to find the correct suite of software tools to effectively capture and decrypt the packets. For my research, I used Kali Linux in a virtual machine as my host. This was because Kali Linux has SDR libraries and tools built-in, which means setting up the environment was substantially less complex. The software that will actually control the RTL-SDR and pull down packets is called Airprobe. Airprobe makes it possible to capture non-hopping downlink channels with our RTL-SDR antenna (note: if the reader does not understand non-hopping downlink channels within the context of GSM, I recommend a crash course in GSM and communications before continuing). The last piece of software used during this research is already a staple for many security and IT experts alike. Wireshark was used as the final piece of software in the packet capturing stack. Airprobe will

---

[4] 2006-2013 ICT data for the world by geographic regions and by level of development http://www.itu.int/en/ITU-D/Statistics/Documents/statistics/2013/ITU_Key_2005-2013_ICT_data.xls

[5] A $40 Software Defined Radio, IEEE Spectrum 06/25/2013 http://spectrum.ieee.org/geek-life/hands-on/a-40-softwaredefined-radio

send its “packets” to Wireshark by making it believe it is receiving packets on the lo network interface. Wireshark can then create a pcap or cfile.

With the GSM capturing environment setup, it is time to talk about the theory behind capturing a specific users communications. When a phone connects to the GSM network it is assigned a TMSI of Temporary Mobile Subscriber Identifier which is used to avoid the transmission of personally identifiable information. We can use the TMSI to identify our victims phone on the network, and capture all of their packets accordingly. To uncover which TMSI belongs to our victim, we employ a simple tactic. When the victim receives anything from the network, their phone will be notified, therefore if we send something to our victim we can correlate our message with a TMSI that is being sent/received between the cell tower and the phone. By sending the victim messages with a specific pattern and frequency, it is trivial to identify which TMSI is associated with their phone.

As we noted before, GSM uses A5/1 to encrypt communications. A5/1 is a stream cipher, which means that plaintext digits are combined with a pseudorandom cipher digit stream (also called the keystream) to create the encrypted output. In order to crack the GSM encryption, we need to know the keystream. Because of the way that GSM is designed, we can actually obtain the keystream rather easily. In GSM there are some bursts of information that are constantly sent back and forth between the network and the phone. These bursts merely contain system information, and according to the GSM standard, they must be sent regardless of whether the current packets being sent are encrypted or not. This is the fatal error in GSM’s implementation, and it is why we can break its encryption. Due to this rule, we *know* what is being sent in both its plaintext and encrypted form, and therefore all that is needed to recreate the keystream is a simple XOR of the encrypted system information burst with the plain text system information burst. I have made this seem slightly easier than it is however, because we still need to be able to identify the system information bursts, and know they format. In order to figure this out, a few test calls or messages may be necessary.

At this point in the process we have the TMSI, and the keystream. Now it is time to record the communication, and begin to decrypt it. Recording the information is a simple waiting game, but first we must find a non-hopping downstream channel in the GSM frequencies. To do this we open Airprobe and begin to scan 1Mhz blocks of the spectrum until we find the GSM signal. In the Airprobe wide-channel view, the frequency will look like a box filter taking up just slightly less than 1Mhz in bandwidth. Once we have identified the correct frequency, we tune to it and packets will immediately start being captured. Cracking A5/1 is a simple brute-fore attack once you have the keystream, and therefore the only way we can improve our chances is with a very good wordlist. For A5/1 a very large wordlist exists. This word list is about 2TB large, which is about $1.75 / month on Amazon S3, and about $85 for a physical disk. The software used to crack GSM is called Kraken, and it was developed by srlabs for this exact purpose. While cracking the encryption still requires quiet a lot of parallel processing power, it is not uncommon these days for a normal user to have multiple powerful graphics cards. If these cards

are not available it is also possible to buy AWS instances that can replicate the parallel processing power of GPUs. Once decrypted the victims information is entirely at the attacker's disposal and can be used in whatever way benefits the attacker.

Conclusion:

The governing body behind GSM has not been entirely complacent in this security vulnerability. Protocol patches exist that make cracking GSM much more difficult. This is accomplished by not disclosing known plaintext messages such as the system information bursts unnecessarily. This patch is considered a short-term solution and is called 3GPP. In the long term, more modern protocols are already taking over in the form of CDMA and LTE. As GSM becomes more and more obsolete, it will eventually be retired entirely. Until that day however, I strongly recommend testing your cellphone carrier's security by trying these experiments on yourself. As always, do not use this information for any illegal activities, and always receive written permission to carry out this analysis form any third parties. Use this information at your own risk. I am not responsible for the misuse of this information or its assistance in breaking terms of service and/or applicable laws.

In conclusion, it should be no surprise that GSM, a 2^{nd} generation protocol has been owned end-to-end. Between poor security implementation and the easy access to the computing power needed fro brute-force attacks, no communication on an unpatched GSM network is safe. If any communications need to be private do not use GSM. An alternative would be communication over modern 4^{th} generation protocols such as CDMA or LTE, and I would recommend that you encrypt your communication from the network as well.