Discovering Network Control Vulnerabilities and Policies in Evolving Networks

Jill Jermyn

Submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy
in the Graduate School of Arts and Sciences

COLUMBIA UNIVERSITY

2017

ABSTRACT

Discovering Network Control Vulnerabilities and Policies in Evolving Networks

Jill Jermyn

The range and number of new applications and services are growing at an unprecedented rate. Computer networks need to be able to provide connectivity for these services and meet their constantly changing demands. This requires not only support of new network protocols and security requirements, but often architectural redesigns for long-term improvements to efficiency, speed, throughput, cost, and security. Networks are now facing a drastic increase in size and are required to carry a constantly growing amount of heterogeneous traffic. Unfortunately such dynamism greatly complicates security of not only the end nodes in the network, but also of the nodes of the network itself. To make matters worse, just as applications are being developed at faster and faster rates, attacks are becoming more pervasive and complex. Networks need to be able to understand the impact of these attacks and protect against them.

Network control devices, such as routers, firewalls, censorship devices, and base stations, are elements of the network that make decisions on how traffic is handled. Although network control devices are expected to act according to specifications, there can be various reasons why they do not in practice. Protocols could be flawed, ambiguous or incomplete, developers could introduce unintended bugs, or attackers may find vulnerabilities in the devices and exploit them. Malfunction could intentionally or unintentionally threaten the confidentiality, integrity, and availability of

end nodes and the data that passes through the network. It can also impact the availability and performance of the control devices themselves and the security policies of the network. The fast-paced evolution and scalability of current and future networks create a dynamic environment for which it is difficult to develop automated tools for testing new protocols and components. At the same time, they make the function of such tools vital for discovering implementation flaws and protocol vulnerabilities as networks become larger and more complex, and as new and potentially unrefined architectures become adopted. This thesis will present the design, implementation, and evaluation of a set of tools designed for understanding implementation of network control nodes and how they react to changes in traffic characteristics as networks evolve. We will first introduce Firecycle, a test bed for analyzing the impact of large-scale attacks and Machine-to-Machine (M2M) traffic on the Long Term Evolution (LTE) network. We will then discuss Autosonda, a tool for automatically discovering rule implementation and finding triggering traffic features in censorship devices.

Contributions

This thesis provides the following contributions:

1. The design, implementation, and evaluation of two tools to discover models of network control nodes in two scenarios of evolving networks, mobile network and censored internet
2. First existing test bed for analysis of large-scale attacks and impact of traffic scalability on LTE mobile networks

3. First existing test bed for LTE networks that can be scaled to arbitrary size and that deploys traffic models based on real traffic traces taken from a tier-1 operator
4. An analysis of traffic models of various categories of Internet of Things (IoT) devices
5. First study demonstrating the impact of M2M scalability and signaling overload on the packet core of LTE mobile networks
6. A specification for modeling of censorship device decision models
7. A means for automating the discovery of features utilized in censorship device decision models, comparison of these models, and their rule discovery

Contents

Acronyms

3GPP 3rd Generation Partnership Project.

CDRs Call Detail Records.

DL downlink.

E-UTRAN Evolved Universal Terrestrial Radio Access Network.

EPC Evolved Packet Core.

GSM Global System for Mobile Communications.

HSS Home Subscriber Server.

IMEI International Mobile Equipment Identity.

IoT Internet of Things.

LTE Long Term Evolution.

M2M Machine-to-Machine.

MAC Medium Access Control.

MME Mobility Management Entity.

NAS Non Access Stratum.

PDNs Packet Data Networks.

PGW Packet Data Network Gateway.

PHY LTE Physical Layer.

QoS Quality of Service.

RAN Radio Access Network.

RRC Radio Resource Control.

SGW Serving Gateway.

UE User Equipment.

UL uplink.

UMTS Universal Mobile Telecommunications System.

Acknowledgements

When I started at Columbia in 2012, I had gone through a career change just 1.5 years earlier and had limited Computer Science background. Although I was good at programming, I didn't have a thorough knowledge of two areas that I was very interested in learning about: networking and security. To be able to earn my PhD from that point has taken a tremendous amount of work and dedication, and it absolutely would have not been possible without the support from many people with whom I have had the honor of working. First, I would like to Salvatore Stolfo, my principal PhD advisor. I am forever indebted to him for giving me the opportunity to study at Columbia and taking me on as a research assistant in the Intrusion Detection Systems Lab. He initially had no way of knowing if a violinist with some programming skills would ever turn out to be a good security researcher, but he somehow had faith in my ability and gave me a chance. I would have never been able to receive my PhD from Columbia had he not given me that chance and I will always be grateful for that. I would also like to thank Sal for his continuous encouragement and support during my time in his lab and for teaching me to think deeply about the implications of my research.

I would like to thank Steve Bellovin for being an endlessly supportive and encour-

aging advisor. I met Steve during the second year of my studies and he immediately inspired me by his enthusiasm for networking, security, historical contexts of computing, and many, many other topics not related to computing that we have enjoyed discussing over the past few years. My experience with Steve brought back many memories from my music education that was very focused on understanding historical significance of different works. It's rather atypical to receive this perspective from engineers and professors of Computer Science, so I really appreciated it. I would also like to thank Steve for giving me tons of ideas and directions for my work and for always being available whenever I needed to get his opinion and during times of panic.

I am very appreciative of my thesis committee members and would like to thank them for all of their help and patience: Sal Stolfo, Steve Bellovin, Vishal Misra, Nick Weaver, and Vugranam Sreedhar. I am very grateful for the time they took to review my work and help me improve. I would also like to thank my colleagues in the IDS Lab over the past 5 years: Adrian Tang, Yuan Kang, Preetam Dutta, Nathaniel Boggs, Jon Voris, Ang Cui, Hang Zhao, Peter Du, David Tagatac, and Michael Costello. I am very appreciative of all that I have learned from them as well as their continuous support and humor.

My PhD studies were greatly influenced by the many internships I had the honor of participating in during the past few years, and I am extremely grateful for the help of several mentors that I have had. First, I would like to thank Roger Piqueras Jover, Ilona Murynets, Mikhail Istomin, and Gus de los Reyes for working with me at AT&T in 2013. The guidance and education they gave me led to a significant portion

of my thesis and I am very appreciative to have had that experience so early on in my PhD. I would particularly like to thank Roger for working so closely with me on the LTE security work and for teaching me everything I know about cellular networks. I can't imagine having worked with a more knowledgeable individual; I still believe he is one of the few people worldwide that has such a deep understanding of mobile networking. I also want to thank Roger for his continuous support and mentorship throughout my PhD.

At IBM I would like to thank Maja Vukovic, Jinho Hwang, Nikos Anerousis, Jin Xiao, Vugranam Sreedhar, Hani Jamjoom, Hari Ramasamy, and JR Rofrano for their help during my internship in summer 2014 and their continued support during my PhD. The enthusiasm of JR and Hari during their cloud course at Columbia really inspired me to learn more about cloud networking and come to IBM as an intern. I would like to specifically thank Sreedhar for always being supportive and brutally honest at all times since we first interacted in April 2014. Also special thanks to Maja, Jinho, and Nikos for helping me with my research and being very involved during my PhD.

At Microsoft I had the opportunity to work with many inspiring individuals in summer 2015. I would like to thank Karthick Jayaraman, Geoff Outhred, and Albert Greenberg for their mentorship and suggestions for shaping my research. I would also like to thank Nick Weaver, Michael Tschantz, and Vern Paxson for working with me at ICSI on the censorship project in summer 2016. Nick's knowledge of security and overall genius really inspired me to deepen my knowledge of security and hacking. I also greatly appreciate his mentorship and encouragement towards the end of my

PhD. I would like to thank Michael for always being available to brainstorm ideas, giving me a new perspectives on my work, and always being up for trying a new dessert. Finally, I would like to thank Vern for his input and guidance on my work. I have really admired his ability to find significance in one's research and steer things in the right direction.

I would like to thank Estie Arkin at Stony Brook University for introducing me to Computer Science back in 2010 when I asked her for recommendations on math courses. Without that recommendation, none of this work would have existed, nor would I have known that this field even existed. Thanks to Estie also for her mentorship and encouragement along the way.

I would finally like to thank my mom, June Adinolfi, for her unwavering support and encouragement not only during the past five years, but also over my entire life. She has been there for me through every up and down that I have experienced with a devotion so great, it's still even hard for me to comprehend. I am particularly grateful for her teaching me to think independently regardless of what anyone thinks and for always telling me the truth even when I didn't want to hear it. My mom has been the strongest role model that I could have imagined and without her, none of this would have been possible.

Dedication

This thesis is dedicated to my aunt Kathleen Jermyn and cousin Tommy Kelly, victims of the terrorist attacks on Pam Am Flight 103 on December 21, 1988 and the World Trade Center on September 11, 2001. They have greatly inspired me to study cybersecurity and have continuously reminded me of the importance of my work.

Left: Barbara, Kathleen, Rita, Jill Jermyn; Right: Tommy Kelly

Chapter 1

Introduction

The range and number of new applications and services are growing at an unprecedented rate. Computer networks need to be able to provide connectivity for these services and meet their constantly changing demands. This requires not only support of new network protocols and security requirements, but often architectural redesigns for long-term improvements to efficiency, speed, throughput, cost, and security. The internet now has more than 3.6 billion devices, almost twice the number of only five years ago [2]. However, this growth rate is confined not only to the internet. The rise of cloud computing has enabled cost-effective increases in the size of data center networks. Modern cellular networks are also drastically evolving to cope with an explosion of mobile devices and high bandwidth mobile services. Along with the increase in the size of communication networks is an increase in the amount and heterogeneity of the traffic that these networks carry. Unfortunately such dynamism greatly complicates security not only of the end nodes in the network, but also of the nodes of the network itself. To make matters worse, just as applications are being developed at faster and faster rates, attacks are becoming more pervasive and complex. Malware instances have increased across all platforms, and we are now faced with protecting systems against Advanced Persistent Threats (APT). Networks need

to be able to understand the impact of these attacks and protect against them.

Network control devices, such as routers, firewalls, censorship devices, and base stations, are elements of the network that make decisions on how traffic is handled. They act abstractly as finite state machines: they receive input traffic, change state based on some implemented rule, then take an action according to the new state. They typically base their decisions on characteristics of the traffic that they receive. Packet content, header data, protocol, and packet timing are a few examples of characteristics that network control devices use as their decision points. The specification of rules implemented in the network control devices can come from various sources: protocol implementation, network administrator, or even a third-party middlebox vendor. Rules make decisions for how control devices act on specific types of traffic. They may drop or inject traffic, limit throughput, or send messages to other network elements according to a specific protocol, among other actions.

Although network control devices are expected to act according to specifications, there can be various reasons why they do not in practice. Protocols could be flawed, ambiguous, or incomplete, developers could introduce unintended bugs, or attackers may find vulnerabilities in the devices and exploit them. Malfunction could intentionally or unintentionally threaten the confidentiality, integrity, and availability of end nodes and the data that passes through the network. It can also impact the availability and performance of the control devices themselves and the security policies of the network. Testing these control devices for vulnerabilities and paths for potential malfunction is not an easy task. When implementations of control devices are proprietary or access to them is not possible for other reasons, it is necessary to reverse

engineer parts of the implementation to understand the intended behavior of the device. Finding vulnerabilities for the purpose of protecting nodes is not the only use case for tools that discover network control device behavior. The same tools could be utilized by attackers who wish to find vulnerabilities for the purpose of exploitation or by non-malicious entities that wish to evade device control in order to protect their right to privacy or net neutrality. Whether the intended purpose of understanding the behavior of the control device is for protection or exploitation, behavior of the device needs to be observed and studied for a range of different inputs. This can be done by modifying input traffic in various ways and observing output of the control device.

The fast-paced evolution and scalability of current and future networks makes creating automated tools for understanding implementation of network control nodes difficult. As new network architectures become adopted, change to existing networks is typically incremental and often done by adding new components that now need to interface with old components. Such a dynamic environment makes it difficult to develop these automated tools that need to test new protocols and components, but also makes their function vital for discovering implementation flaws and protocol vulnerabilities as networks become larger and more complex, and as new and potentially unrefined architectures become adopted.

While the overall goal of such tools can be to uncover and analyze vulnerabilities of network control nodes, the purpose of the tools can be viewed as multi-use. They can be used for protection of nodes against intentional or unintentional misuse. They can also be used to maliciously find paths for exploitation of these nodes or non-

maliciously for discovering paths of circumvention to evade government-mandated censorship. In this thesis we examine two network scenarios that benefit from the use of our approach: cellular network and nation-state censored internet. For the cellular network we focus on vulnerabilities due to control plane signaling overload, and in the censored internet case we demonstrate an automated means for finding paths of censorship circumvention. Although there currently exist tools for testing specific scenarios and tests covered by our tools, the primary contributions of this work are the comprehensiveness, scalability, and extensibility of our tools, which are specifically designed for easing adoption of network evolution. This work presents the design, implementation, and evaluation of a set of tools designed for understanding vulnerabilities and implementation of network control nodes when access to them is only available through network traffic, as well as how these devices react to changes in traffic characteristics as networks evolve.

Hypothesis

Network traffic can be crafted and used to probe network control devices for which control source code, binaries, or remote login access is not possible. Automated tools that create such traffic and probe network control nodes can provide a means to detect, analyze, and compare vulnerabilities of these nodes and discover how they react to changing traffic characteristics as network evolve. These vulnerabilities could maliciously or non-maliciously affect the availability of the nodes or allow bypassing of their security policies.

Contributions

This thesis will provide the following contributions:

1. The design, implementation, and evaluation of two tools to discover models of network control nodes in two scenarios of evolving networks, mobile network and censored internet
2. First existing test bed for analysis of large-scale attacks and impact of traffic scalability on Long Term Evolution (LTE) mobile networks
3. First existing test bed for LTE networks that can be scaled to arbitrary size and that deploys traffic models based on real traffic traces taken from a tier-1 operator
4. An analysis of traffic models of various categories of Internet of Things (IoT) devices
5. First study demonstrating the impact of Machine-to-Machine (M2M) scalability and signaling overload on the packet core of LTE mobile networks
6. A specification for modeling of censorship device decision models
7. A means for automating the discovery of features utilized in censorship device decision models, comparison of these models, and their rule discovery

In the cellular network context, we present a test bed called Firecycle that is used for assessing the impact of a large-scale attack or traffic scalability on network control nodes in a LTE mobile network. Wireless cellular networks based on 3rd Generation Partnership Project (3GPP) standards have adopted LTE as the underlying technology to provide rich services to the rapidly growing number of connected devices. The

advent of the Internet of Things and the proliferation of Machine-to-Machine (M2M) systems are transforming cellular systems into diverse and heterogeneous networks, estimated to be servicing tens of billions of devices in just a few years. Yet, the exact impact, for example traffic load, CPU and RAM usage, and security on the network elements of the radio access network and packet core from such an increase in the number of these connected devices remains unknown. Furthermore, cellular networks including LTE have been long studied and optimized to support human-generated traffic, which differs greatly from machine-generated traffic. Previous cellular networks are known to be vulnerable to a variety of security threats, yet there has been little research on understanding if LTE network nodes remain vulnerable to the same and new threats. We introduce a new modeling test bed, Firecycle, for studying the impact of both M2M systems and large-scale attacks on the availability of network control nodes in the Evolved Packet Core (EPC) of LTE mobility networks as well as future network architectures. Firecycle can be distributed over multiple virtual machines in the cloud, allowing it to scale to an arbitrarily sized network for testing of large-scale attacks. It also contains flexible and realistic traffic models that are based on real traffic traces taken from an existing LTE network in the United States. Testing the impact of a large-scale attack and the scalability of M2M systems that cellular networks are required to face have not been previously possible given that there are no existing research labs or simulation platforms that are scalable, flexible, or modular enough to perform this type of research. The plans to use LTE for emergency response systems, public safety, and advanced military tactical networks, among other important and security critical applications, are examples highlighting

the importance and need for our system.

Understanding network control device implementation is not only important for building more secure and resilient networks, but it is equally important to attackers for finding vulnerabilities or non-malicious entities for circumvention of implemented rules. Analysis of nation-state censors and finding of evasion techniques has been a popular topic of recent research, and understandably so, given its pervasiveness in the world today. Censors act based on characteristics of the network traffic they examine. These characteristics can be protocol-dependent, such as packet header fields, or not, such as a keyword in a packet's content. Censors use these traffic characteristics to make decisions on how they handle traffic that they intend to censor, typically by blocking it, modifying its content, or injecting packets into the network stream. Several approaches have been taken to circumvent censors. The Tor anonymity network, although originally intended for different purpose, has now been commonly used as a tool for circumventing censorship. Rather than trying to evade a specific rule in a censor, Tor uses obfuscation techniques to make traffic appear different than it actually is. However, censorship implementations are constantly changing in order to adapt to Tor and other circumvention techniques and refine their blocking techniques. Thus, there is a great importance still to understand the implementation of these censorship devices to devise mechanisms for circumvention. Rules can be reverse engineered by specially crafting traffic and probing the censor to examine its output. Although there has been some recent existing research on understanding approaches of specific censors, such as the Great Firewall of China, the great majority of censorship techniques remain unknown. In addition, the methods to date

for discovering censorship techniques are mostly manual, which are time consuming, not scalable, and are not feasible in the long run, given the dynamic nature of censor implementations. In this work we present Autosonda, the first automated tool for discovering censorship techniques and decision models across different layers of the network stack. Our technique first finds traffic features on which a censor acts and then uses a fuzzing-based approach to automatically discover the implemented rules for those features. Autosonda runs a series of tests across different protocols to discover the model, mechanism, and technique of censors. In this work we demonstrate the utility and effectiveness of Autosonda with a study on web filters, which is presented in Chapter 6.

Chapter 2

Related Work

Attacks against LTE and tools used for assessing impact of attacks

Security research in mobility networks has been often with a purely theoretical approach and without tools used to gauge their impact in realistic environments. Moreover, the open source availability of GSM platforms, such as OpenBTS [40], results in most of the hands-on security work focusing on GSM networks. Presently, the amount of security research concentrated exclusively on LTE is very small. One exception is the work presented in [15], analyzing potential signaling attacks against LTE performed by User Equipments (UEs) within the same cell. In this work we present Firecycle, a test bed for large-scale security research in LTE mobile networks. Firecycle provides the means to analyze these known legacy vulnerabilities in a realistic environment and determine whether they also affect next-generation LTE networks, as well as identify potential new vulnerabilities.

In recent years, several researchers have detected potential vulnerabilities and proposed attacks against mobility networks. The authors of [47] proposed a potential way to saturate a cellular core network. Based on signaling generated to transition

UEs between RRC (Radio Resource Control) states, an attacker could potentially overload the network by forcing UEs to constantly switch states. The negative impact of such a signaling spike has already been experienced in the wild due to inadequate legitimate applications that induce frequent RRC connections from many UEs [22, 28]. A similar attack was discovered in [72], aiming to saturate the network resources assigned to the paging mechanism. Further signaling-based attacks against a cellular network were introduced in [39].

A widespread malware infection or a botnet of mobile devices is often considered as a potential platform to launch such attacks. Along these lines, the authors of [57] discussed the feasibility of creating and operating a botnet of infected smartphones. The impact of such a botnet launching an attack against the Universal Mobile Telecommunications System (UMTS) Home Location Register (HLR) was theoretically analyzed in [83].

There are some open access platforms available for security research. For example, the DETER project [54] provides means for large-scale security studies of IP (Internet Protocol)- based networks. Similarly, the ORBIT test bed is an impressive vehicle for testing and experimenting next-generation Wireless Local Area Networks (WLANs) with up to hundreds of nodes [68]. The authors of [34] introduced another network modeling platform to study the impact of M2M traffic on the LTE Physical Layer (PHY). Although Firecycle and the model proposed in [34] both analyze the impact of M2M on LTE mobility networks, Firecycle is concerned with the LTE network at large whereas the other focuses exclusively on the wireless interface of an LTE single cell network. [63] presents an open source LTE simulator called LTE-Sim, although

this tool is much simpler and less flexible than Firecycle, as it emphasizes only the physical layer and lacks the ability to deploy custom traffic models that are important for analysis of security attacks on the network. These two models can therefore be seen as complementary to Firecycle.

Scalability of Machine-to-Machine systems

The advent of Machine-to-Machine (M2M) systems and the pervasive presence of embedded connected devices within the IoT have resulted in a substantial increase of research work in this area. The authors of [74] introduced the first detailed study of the traffic characteristics of emergent M2M applications. Among other findings, they highlighted the potential radio resource and network resource inefficiencies of these communication systems, but they don't include simulation studies to demonstrate specific network impact. The inefficiencies they point out are well understood by the industry and standardization bodies, which acknowledge the potential threat of signaling overloads [32]. In an effort to enhance the resource utilization efficiency of M2M, cellular operators often provide guidelines and best practices [48].

In the context of the potential overload that could be generated by the forecasted large scale of the IoT, the authors of [34] presented a study on the performance of LTE under M2M traffic load. Based on a Markov model of the physical layer (PHY) parameterized by lab measurements, the results indicate that a large number of devices with small traffic rates could potentially induce the most negative effects. However, this work focuses exclusively on the Radio Access Network (RAN) and

performs experiments with arbitrary traffic, while our work expands this analysis to the impact of M2M at higher layers with a focus on the Evolved Packet Core (EPC), using M2M traffic models derived from real traffic traces. The results in [61] also provide very interesting insights on the potential impact of M2M deployments over LTE. The results are obtained from OPNET-based simulations but with arbitrary simple M2M traffic models and the OPNET LTE network models, which are very powerful in terms of network planning and traffic capacity estimations but do not model the impact of NAS procedures, signaling load, and other artifacts that are highly relevant to the scalability of M2M. By means of our custom-built LTE security research test bed we expand this study to the actual impact on the EPC.

Signaling overload and DDoS attacks in cellular networks

Security research throughout industry and academia has intensified over the last few years. The widespread availability of low-cost, open-source GSM platforms has resulted in the successful identification of potential vulnerabilities and threats to legacy mobile networks. The majority of security research has mainly focused on Second and Third Generation mobile networks (2G and 3G).

The authors of [47] introduced a theoretical signaling overload attack against cellular networks. As discussed later in this work, a low-volume attack, consisting of small data packets addressed to a large number of mobile devices, could theoretically be leveraged to force a large number of RRC state transitions. This could potentially overload the packet core of a mobile network. The causality behind this type of attack

was discussed in [80].

An accurate list of targets to send messages to within the cellular network would be very difficult to obtain due to the NAT-ing and gateways at the edge of the mobile network. Nevertheless, a similar attack could potentially be launched from the inside of the network by means of a mobile botnet. The authors of [57] discussed feasible techniques and platforms to build and operate such a botnet, including potential command and control channels. In parallel, the authors of [83] discussed theoretical attacks launched from a mobile botnet targeting the Home Location Register (HLR), with similar functionality to the HSS but in the context of 3G networks.

Further signaling-based attacks against 3G networks were introduced in [39]. Similarly, the authors of [72] presented a theoretical attack aiming to saturate the resources allocated to the paging channel, potentially preventing the delivery of phone calls, short messages, and data flows. In [66], further threats were introduced, which exploit the opportunistic scheduling of 3G networks. Finally, the authors of [82, 81] introduced new potential attacks against legacy 2G networks exploiting the fact that, in the GSM air interface, text messages share resources with essential signaling channels.

Although all these attacks target legacy 2G and 3G networks, it should be investigated whether their impact is still applicable to an LTE deployment. On top of that, some recent research work has investigated new potential attacks against LTE as well. The authors of [42] presented a simulation study of a potential attack launched by a botnet of mobile devices. The thesis discusses the potential overloading of the LTE RAN resources that a botnet could generate. Other LTE security research works

have recently surveyed other security aspects of LTE [17].

It is important to note that although there are no recorded instances of such attacks in cellular networks, similar effects due to signaling overloads at a cellular core network have been observed with a non-malicious origin. An example was caused by a poorly programmed instant messaging application which induced frequent and unnecessary RRC state transitions for a large number of UEs. This resulted in an overload situation for a major operator in the US [22]. Other foreign operators have been under similar signaling overloads [28]. These are clear motivations to investigate the effects of signaling-based threats against the cellular core network and, ultimately, defend against them.

Discovering censorship mechanisms

Discovery of censorship techniques has been the topic of much recent research, yet there are still many techniques that remain unknown. [84] offers a nice overview of known censorship mechanisms. Techniques used in Chinese censorship have received particular attention, such as how China censors Web accesses [21], Tor [86], Skype [44], the location of its censoring modules [89], and even how it discovers and blocks privacy tools [24]. Countries such as Bangladesh, Bahrain, India, Iran, Malaysia, Russia, Saudi Arabia, South Korea, Thailand, and Turkey have received some preliminary analysis in [85]. [12] extends the analysis of censorship mechanisms by focusing on finding motivation, resources, and time elements of censorship. From an ISP perspective, Clayton studied the British Telecom CleanFeed blocking system in [20].

Although there has been significant work on discovering censorship mechanisms, the bulk of the techniques have been performed manually, which is the primary motivation for our work, and have been targeted to specific types of censors. [41] proposes an approach for fingerprinting censorship devices and serves as a motivation for our work. However, the research was limited only to the Great Firewall of China and the analysis was performed manually. We used the results of this research to validate our results obtained from Autosonda on the Great Firewall.

Some existing tools similar to Autosonda have been developed for different purposes. ooniprobe [91] is an app that measures internet censorship and performance. It can be used to discover which websites are blocked by a censor and if there is a system on the network that can be responsible for censorship or surveillance. The uses of ooniprobe are different than those of Autosonda, and these tools could be used together to do a full analysis of where there exists censorship, which sites are blocked, and how the filtering mechanisms work. Netalyzr [46] runs a series of tests that probe a network for both measurement and debugging in order to discover a wide range of properties of users' internet access. It runs as a Java applet that is accessed in a web browser and communicates with custom built servers.

Chapter 3

Firecycle: A Scalable Test Bed for Large-Scale LTE Security Research

3.1 Introduction

The Long Term Evolution (LTE) is the latest standard for mobile wireless communications. Such technology is rapidly being deployed by cellular network operators to provide capacity for advanced multimedia services. Mobility networks have become an essential element of our day-to-day lives, providing popular seamless services such as electronic email, location-based applications, video streaming, and the like. Hundreds of millions of people are connected to the Internet via smartphones and tablets [30]. In addition, communication networks are rapidly evolving, with connectivity reaching beyond user devices. The advent of the Internet of Things (IoT) and Machine-to-Machine (M2M) systems is pushing the number of network-enabled entities interacting with each other over mobile networks. The number of connected devices is expected to be in the range of billions within a few years [56], with the majority of these devices connecting over next-generation LTE access networks.

Although LTE presents tremendous capacity enhancements, in general, previous cellular networks are known to be vulnerable to certain security threats [29]. At the

same time, the recent security attacks against communication networks have drastically changed the security ecosystem. The massive Distributed Denial of Service (DDoS) attacks against major banking institutions [31], the attack against Spamhaus that resulted in worldwide Internet sevice deterioration [51], and the surge of mobile malware [52] are good examples of the importance of securing communication networks. Security research has been growing over the last few years, resulting in the successful identification and mitigation of many threats. However, the great majority of the work has focused exclusively on Global System for Mobile Communications (GSM) and UMTS. This is perhaps due to the open source availability of platforms such as OpenBTS [40]. To this point there has not been much security research focused on LTE networks.

We developed Firecycle, a test bed for security research in LTE and potential future mobility networks. Firecycle is designed to provide the means to implement, test, and analyze the impact of security attacks against an LTE mobility network. Based on a modular and flexible architecture, the proposed model allows for rapid prototyping, testing, and comparing of new cellular security architectures. As a result due to its versatility and the abundant statistical information obtained from simulations run on Firecycle, the test bed is also a valuable resource for designing strong security architectures for future next-generation mobility networks. Implemented on the network simulation software OPNET [59], Firecycle is designed to be scalable. The network under analysis can be arbitrarily divided into portions that are individually run on separate virtual machines (VMs) in the cloud. The VMs intercommunicate over IP, effectively simulating an arbitrarily large network over multiple machines.

This architecture, based on OPNET's System-in-the-Loop (SITL) [60], provides a platform for security research over a full-scale LTE network with millions of simulated mobile terminals.

3.2 LTE security

Wireless cellular networks provide connectivity to billions of users, electronic devices, and critical applications. Devices connect through a heterogeneous set of access networks, which range from the early 2G GSM networks to the more recent 3G UMTS networks and their Code Division Multiple Access (CDMA)-based counterparts, such as CDMA2000. These mobility networks have evolved over the recent years, enhancing capacity and throughput, defining both the Evolved High Speed Packet Access (HSPA+) and the Evolution Data Optimized (Ev-DO) and its revisions. All modern mobile networks, however, are converging towards one universal technology that will run the next-generation networks: LTE and its evolution, LTE-Advanced.

In addition to being the main wireless access network for billions of users, there are also plans to use LTE in certain critical applications with stringent security requirements. For example, LTE is the communication technology for the next-generation emergency response systems, the Nationwide Interoperable Public Safety Broadband Network [58]. LTE is also considered as the underlying technology for advanced military tactical networks [78]. Concurrent to the security requirements of LTE networks, the cyber security landscape has substantially evolved over the last few years. In the age of massive DDoS attacks, mobile malware and fraud, and the advent of the Ad-

vanced Persistent Threat, the importance of enhancing the security of LTE networks against security attacks is clear [62].

3.3 Motivation

This section introduces the main features of Firecycle, focusing on its capabilities and ability to scale up simulations and the model itself over multiple VMs in the cloud. The motivations behind this test bed are discussed as well.

The recent trend of DDoS attacks against communication networks, which impacted large banking organizations [31] and the overall Domain Name Service (DNS) [51], illustrate the importance of strengthening the reliability of mobility networks against security attacks. In order to guarantee full availability of communication systems against DDoS threats, security research is necessary to come up with both detection techniques as well as attack mitigation strategies [62].

Although mobile devices engage in IP communications over LTE, the specific characteristics of mobility networks, especially at the Medium Access Control (MAC) and Radio Resource Control (RRC) layers, make the network behavior and reaction to security threats very unique. For example, it is well known that the RRC strategies implemented in mobile networks can be theoretically exploited in a large-scale DDoS [47]. The abundant signaling traffic generated when a User Equipment (UE) transitions from a connected state to an idle state and vice versa could saturate certain nodes or links in the core network. However, there is no simple way to test such an attack and quantify its impact against a network with millions of users.

There does not currently exist a security research lab that is sufficiently large to test an attack involving an extensive number of infected smartphones. Furthermore, no simulation platform is scalable enough to run a simulation of a full-scale mobile network with tens of millions of smartphones, tablets, and M2M devices operating in parallel. Moreover, even in small-scale simulations of an LTE network, traffic is modeled following simple arbitrary probabilistic models. One such example is the random waypoint model [16]. Well-known research has proven that such models are far from accurate as opposed to traffic models derived from real network traffic traces [37, 23]. The traffic modeled in Firecycle is based on analysis of fully anonymized real traces from one of the major tier-1 operators in the US.

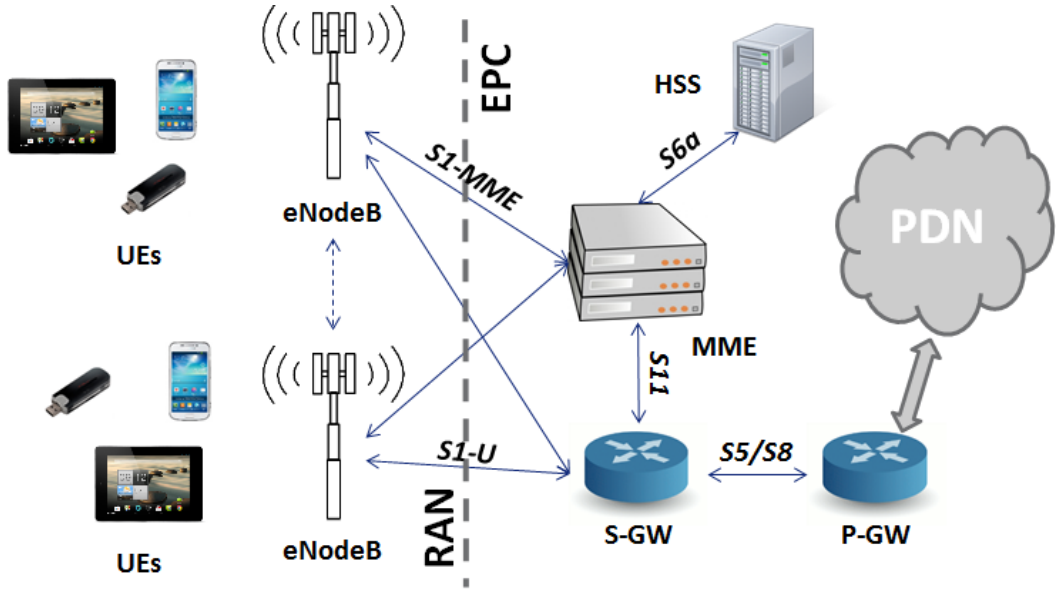


Figure 3.1: LTE network architecture, consisting of elements in the the Radio Access Network (RAN) and Evoloved Packet Core (EPC), as well as interfaces between the elements

3.4 LTE network architecture

LTE was designed with the goal of providing IP (Internet Protocol) connectivity between mobile terminals and the Internet. To that end, an LTE network is typically equipped with a number of nodes that perform specific tasks. Figure 3.1 snapshots the architecture of an LTE network. LTE networks split their architecture into two main sections: the Radio Access Network (RAN) and the core network, known as the Evolved Packet Core (EPC) [73].

The RAN of an LTE network is comprised of the mobile terminals, known as User Equipment (UE), and eNodeBs, or LTE base stations. The evolution of mobile networks towards LTE has highly isolated and specialized the operations of the RAN and the EPC. The RAN is able to, independently from the EPC, assign radio resources to UEs, manage their radio resource utilization, implement access control, and, leveraging the X2 interface between eNodeBs, manage mobility and handoffs.

The EPC is in charge of establishing and managing the point-to-point connectivity between UEs and the Internet. It contains a number of nodes; the Serving Gateway (SGW) and the Packet Data Network Gateway (PGW) are the two routing points for user traffic connectivity to the Packet Data Network (PDN). In addition, logistics of the bearer establishment and release, mobility, and other network functions, such as authentication and access control, are managed by the Mobility Management Entity (MME). In order to provide security for user traffic, the MME communicates with the Home Subscriber Server (HSS), which stores the authentication parameters, secret keys, and user account details of all the UEs.

A real LTE network deployment also implements other nodes for higher layer applications, such as the Policy Control and Charging Rules Function (PCRF), and other elements in charge of IP Multimedia Subsystem-based connectivity. These other nodes and respective higher layer applications are out of the scope of this work's analysis.

3.5 Implementation

Firecycle has been designed, implemented, and coded from scratch using OPNET Modeler as the underlying platform and simulation engine. All the nodes and elements of the model are custom-coded and assembled together to run as a network simulation on OPNET. Finally, an original set of libraries and definition files provide the means to run the realistic traffic models.

Firecycle is a standards-compliant test bed, closely following the specifications of the 3GPP LTE standards. Figure 3.2 shows Firecycle's architecture. The Evolved Universal Terrestrial Radio Access Network (E-UTRAN) is build as a set of eNodeBs (LTE base stations) that handle radio communication between UEs and the EPC (Evolved Packet Core). Our model can deploy any number of eNodeBs and UEs as well as establish the LTE X2 interface between interconnected eNodeBs.

The EPC model contains the Mobility Management Entity (MME), the Serving Gateway (SGW), the Home Subscriber Service (HSS), and the Packet Data Network Gateway (PGW). These are the main nodes required to establish connectivity and manage traffic flows to and from UEs [73]. The primary functions of the MME

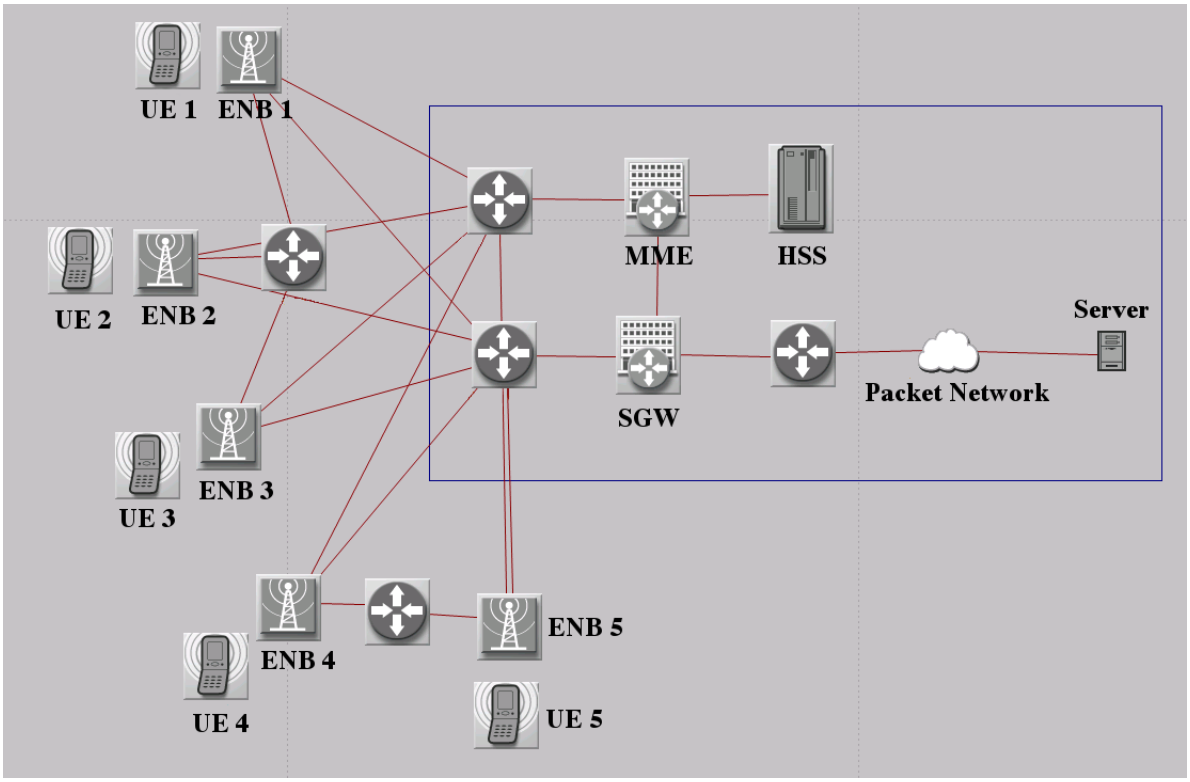


Figure 3.2: Firecycle model architecture

are control-plane signaling, SGW selection, authentication, and bearer management. The SGW routes and forwards traffic in addition to initiating the paging procedure when traffic arrives for a UE that is in idle state. The PGW handles UE IP address allocation and packet filtering. Furthermore, the HSS is a database that contains user identification, subscriber information and authentication information, as well as Quality of Service (QoS) assignments. Note that we have designed Firecycle to allow multiple SGWs, MMEs, and PGWs in order to study the impact of distributing traffic and signaling load across a larger network.

In the current implementation, UEs communicate by means of IP-based services with one or multiple external IP servers. UEs can send or receive different types of traffic to or from them. These servers connect to the LTE EPC through an IP cloud,

which mimics the timing artifacts that packets experience as they travel through a network. Packet drop rates and reordering of packets could be produced as well.

Firecycle implements several 3GPP standards-based signaling procedures that are relevant to UE connection and utilization of network resources. In order for a UE that is currently not attached (off connection state) to transmit and receive IP data traffic, it must go through the initial attach procedure to reach the RRC connected state. During this sequence of signaling messages, the UE associates with a particular eNodeB, and the EPC allocates a collection of bearers that are used for the duration of the UE's session [4, 6].

When a UE is inactive for a given period of time, the E-UTRAN and EPC release its resources, and the device is subsequently brought to the idle state. If traffic arrives at the SGW for a UE that is in idle state, the SGW sends a data notification message to the MME in order to initiate the paging procedure, leveraging the knowledge of the set of base stations with which the UE last associated [8]. Once the UE has received a page message, it will transition to connected. A UE in connected state possesses the ability to send and receive messages or data. Since in LTE both messages and data are IP traffic, we have implemented all traffic as IP packets, although the size and frequency of these packets are dependent on the traffic type.

Although Firecycle models all the relevant signaling procedures at the E-UTRAN and EPC, the actual cryptographic operations behind the HSS operation are modeled only as signaling exchanges. In the current implementation, Firecycle is not able to study vulnerabilities in the cryptographic operations behind, for example, the EEA2 confidentiality algorithm and the EIA2 integrity algorithm.

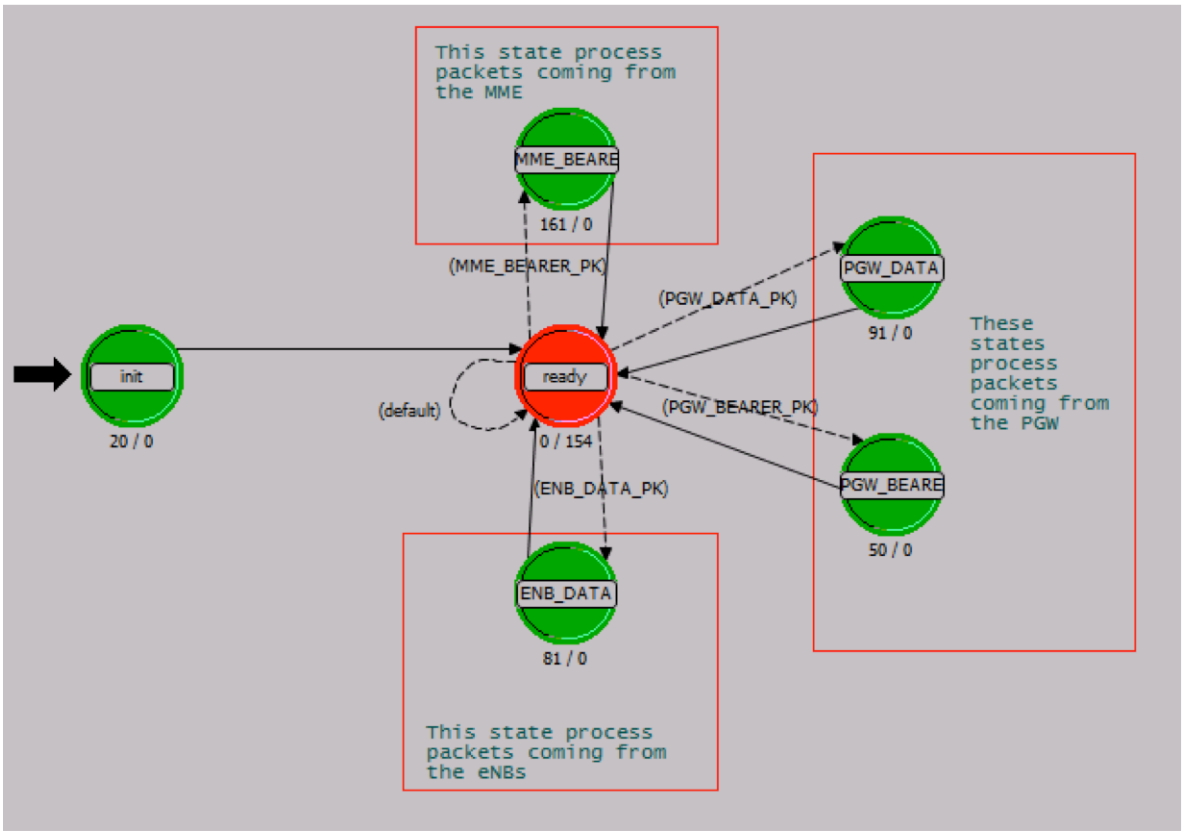


Figure 3.3: SGW state diagram: SGW changes state depending on attributes of the packet it receives as input

OPNET models communications as a chronological sequence of events. Actions in each node, for example sending and receiving messages, determine the progression of states in the system. Each node is depicted as a finite state machine, where the next state depends on the type of event currently being executed. Figure 3.3 is a screenshot of the SGW's state model. While the SGW is waiting for packets to arrive, it remains in the ready state. If it receives a data packet from an eNodeB, the SGW will move to a new state that handles processing of eNodeB data packets. The SGW could also receive signaling messages from the MME and data packets from the PGW. Depending on where the packets are received from and the type of packets,

the SGW will move to the appropriate state to process the packets. Once processing is complete, the MME will return to the ready state.

Each network node in Firecycle is implemented as a computing machine similar to those in a real production network. For example, the MME is essentially a combination of proprietary software and hardware from a given vendor. As such, each node has a limited processing speed and memory. Therefore, if a node, for example the HSS, is overloaded with authentication requests, it may get saturated, blocking or substantially delaying its processing and response packets. Such limitations are modeled in Firecycle in order to accurately quantify the impact of DDoS attacks against a mobility network. Moreover, by appropriately configuring each node, one can model the specific capacity of nodes from different vendors.

Statistical Analysis

A significant motivation for our model is to assess the impact of a large-scale security attack against LTE. To serve this purpose, Firecycle captures a variety of statistical information from each simulation. This data can be classified into various categories.

Signaling statistics collect diverse information on the load, frequency, and time occurrence of LTE signaling events. Some examples are the communication between an eNodeB, the MME, and the SGW to establish a bearer and move a UE from idle to connected mode. For example, we can compute the ratio of signaling to actual data messages per each communication flow or track when each individual UE switches between RRC states. Statistics of this category are crucial for gauging the impact of

an attack and are related to the QoS experienced by the UE. Load, frequency, and time occurrence statistics are additionally collected for user traffic.

Quantitative statistics track the total number of events associated with a particular type. The number of RRC state transitions, counting the number of data packets received at the SGW, and the prevalence of UEs entering idle state are a few examples. Such statistics can be informative when comparing the impact of multiple traffic types on particular EPC elements. Node limits statistics deal with capacities of individual nodes, including CPU and RAM usage, throughput, number of bearers, and number of UEs being handled. Link limits examine link throughput and utilization.

Scalability

No known LTE network model can be easily scaled to research the impact of a large-scale security attack involving hundreds of millions of UEs. However, with the number of connected devices expected to grow substantially within the next decade, it is crucial that a model be flexible enough to handle an arbitrarily large number of UEs.

OPNET's System-in-the-Loop (SITL) enables multiple pieces of a model to run on separate machines and communicate with each other over IP. Nodes within a single simulation piece correspond with each other via OPNET- based packets, while nodes across simulation pieces running on different machines communicate via real IP packets that contain simulation packets as payload. To distribute Firecycle over multiple VMs, subsets of the model's nodes can be arbitrarily assigned to distinct

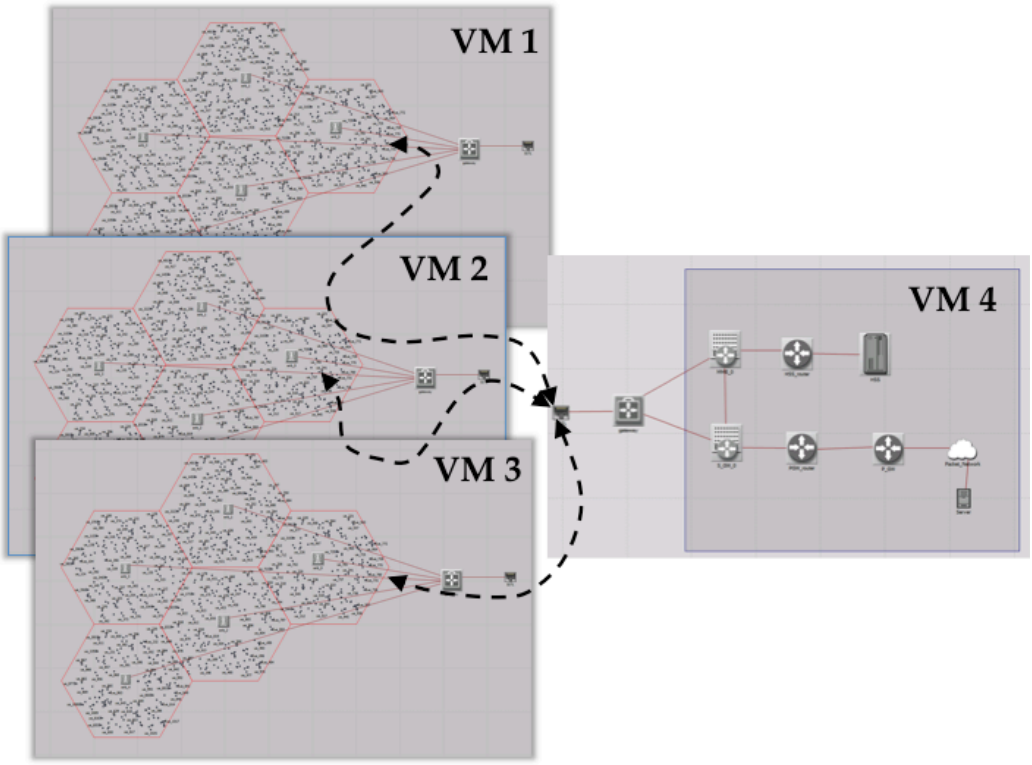


Figure 3.4: Firecycle scaled to four VMs

simulation portions. Once the pieces are each assigned a unique IP address, they can interface over an IP connection by means of SITL gateways. Figure 3.4 shows a screenshot of Firecycle with the EPC running on a VM and three instances of the E-UTRAN, each running on a separate VM.

When a node in one simulation portion needs to send a message to a node running in a different portion, the message, in the form of a simulation packet, is copied into the payload field of a real IP packet that is addressed to the corresponding remote VM. Since the SITL node operates at the data link layer, the IP packet is further encapsulated into an Ethernet packet with the destination MAC address set to the MAC address of the destination machine's network adapter.

The task of copying and encapsulating simulation packets into real network packets is performed in Firecycle's gateway node. Correspondingly, once an Ethernet packet arrives at its destination simulation portion, the gateway node extracts the encapsulated simulation packet and injects it into the OPNET-based local simulation. Thus, the receiving node in the simulation is unaware of the existence of the SITL interfaces and is able to process each packet as if the entire model were running on a single machine.

Traffic

We analyzed anonymized IP and SMS communications in a US tier-1 cellular network to build traffic models of smartphones, tablets, and M2M devices. No personally identifiable information was gathered or used in conducting this analysis. We collected anonymized Call Detail Records (CDRs) that log communications handled by cell towers located in the greater New York City metropolitan area within one day in August 2013, examining the following fields:

1. timestamp of communication (send/receipt time of SMS or IP session connection time),
2. fully anonymized transaction identifiers (integers uniquely determining originating and terminating numbers or addresses for SMS or IP flows),
3. the number of bytes in uplink or downlink IP flows,
4. the first 8 digits of International Mobile Equipment Identity (IMEI).

The first segment of IMEI, Type Allocation Code (TAC), discloses the manufacturer and model of the wireless device and categorizes it as a smartphone, tablet, or M2M subcategory (medical, detention, smart grid, eBook, etc.). Timestamps were used to compute lengths of intervals between each pair of consecutive SMSs or IP data flows. We created custom probability distributions of the traffic parameters for each category according to traffic size and frequency. Based on the identifier fields, we calculated the total number of unique devices of a particular type communicating in a given area. For each device category, we modeled the total number of messages or traffic flows during the period of analysis as well as estimated distributions of the uplink and downlink flow sizes in bytes.

3.6 Results and Applications

Firecycle is designed to provide a platform to both analyze the impact of security attacks against LTE networks and test potential mitigation techniques. For example, from a security point of view, it is particularly important to understand the impact of signaling load on the HSS, since any impairment to the HSS could result in failure of the entire network. A similar approach could be considered for the MME.

Figure 3.5 and Figure 3.6 plot two examples of the signaling link utilization between two pairs of essential EPC nodes, MME-HSS and MME-SGW, for different numbers of UEs. During the simulation the UEs complete an attach procedure and then exchange traffic with a server. Figure 3.5 shows the results with traffic modeled from one of the M2M subcategories. In Figure 3.6 the UEs and server generate ar-

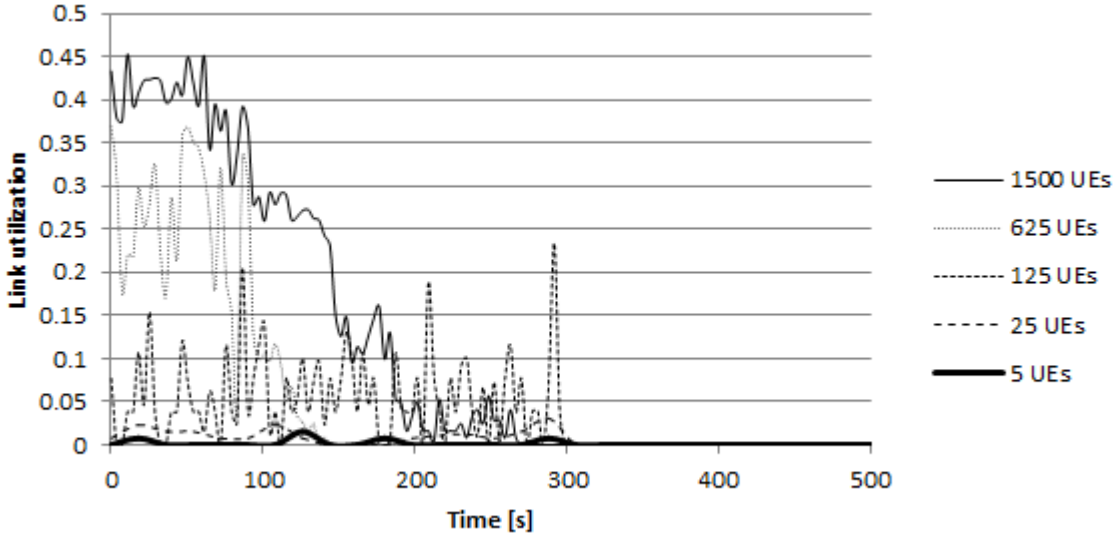


Figure 3.5: Normalized MME-HSS link utilization during initial connection and communication for a category of M2M traffic

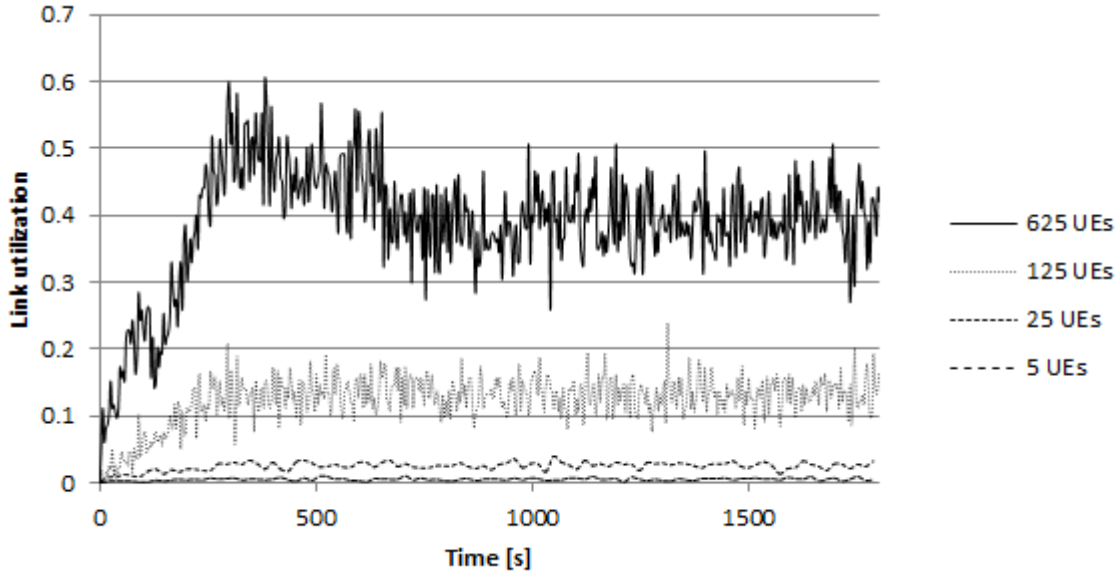


Figure 3.6: Normalized MME-SGW link utilization for traffic causing frequent RRC state transitions

bitrary intermittent traffic in order to induce frequent RRC state transitions. Note that this type of traffic behavior, forcing constant RRC state transitions, has been

proposed as a potential way to launch a signaling-based attack against the EPC [80]. This is indeed one of the security test cases being studied with Firecycle.

Note that the capacity of the links has been set arbitrarily for this example in order to highlight the results. In a real network, the bandwidth of these links would be much higher and thus the utilization lower. Note also that the number of UEs is scaled exponentially. Our work in later chapters tests Firecycle with an arbitrarily large number of UEs that increases exponentially over the entire network. The results in Figure 3.5 and Figure 3.6 are intended to exemplify the type of studies that can be performed with Firecycle.

In Figure 3.5 one can observe the spike in signaling traffic for the attach procedure that all UEs perform at the beginning of the simulation. This procedure involves authentication and attach messages that are exchanged with the HSS. However, once the UEs have successfully attached with the network, the signaling traffic at the HSS dissipates. Based on this observation, we used Firecycle to test, for example, the impact of a malware infection that forces a large percentage of smartphones to attach at the very same time. The results of this and similar studies are presented in Chapter 5.

Figure 3.6 illustrates the signaling traffic load between the MME and the SGW resulting from UE attachment and shifting of RRC states. It can be seen that the messaging between the MME and the SGW to set up and release bearers as UEs transition between idle and connected states creates a substantial and sustained amount of load in the EPC. This insight is being used to investigate the effects of a very large number of UEs constantly transitioning between RRC states.

3.7 Limitations and Future Work

Despite the great potential for security research, Firecycle cannot be applied to every type of LTE security analysis. The wireless interface at the physical layer of a LTE mobile network is based on an Orthogonal Frequency-Division Multiple Access (OFDMA) method. The characteristics of this wireless access method are not implemented, modeling instead a simplified wireless link. While capacity, throughput, and latency are equivalent to a real LTE access network, the results obtained could substantially differ from a real network in some specific cases. For example, Firecycle will not accurately model a real network in the case of an unrealistically large number of active UEs per eNodeB. In this extreme case, the specific details of a full OFDMA system would not be accurately modeled. Firecycle has not been designed to test local wireless resource saturation or radio jamming attacks. Instead, it is designed to investigate the wider scope impact of widespread malware infections, botnets of mobile terminals, and large-scale signaling attacks against the EPC. However, a standards-based implementation of the PHY layer is part of our future work.

In its current form, Firecycle does not implement certain higher layer protocols, such as IMS (IP Multimedia Subsystem) and SIP (Session Initiation Protocol). However, the modularity and scalability of the model highly simplifies the implementation of these and other applications. Firecycle is currently being used to assess the impact and scalability of the IoT and M2M on LTE networks, with focus to the signaling and traffic load in the EPC. However, we are also utilizing it to quantify the impact

of large-scale attacks against LTE. In this context, as part of the future work, Firecycle will leverage the features of SITL to be integrated with the hardware in our LTE security lab. Firecycle will also be tested with traffic generated by UE+eNodeB emulators. As part of our ongoing work, Firecycle has been recently enhanced with new M2M and smartphone statistical traffic models and the implementation of the signaling procedures related to handover and mobility.

3.8 Conclusion

Cellular networks are facing new use cases and traffic patterns now and in the near future. In order to understand the impact that these will have on network elements, tools are needed to analyze the elements, the new patterns, and the interaction of the two. New architectures are also being proposed for enhancing efficiency and security for future cellular networks, and such tools are also needed to test and analyze these new network architectures and components. This chapter introduces Firecycle, a scalable test bed for LTE security research. This platform is designed and implemented to test and realistically quantify the impact of large-scale security attacks against LTE. Firecycle provides the means for rapid prototyping and testing of new attack mitigation strategies and alternative cellular network architectures for enhanced security. Offering the ability to compare the impact of attacks on multiple architectures, Firecycle can aid in the design of robust future next-generation mobility networks.

Firecycle realistically models the main nodes of an LTE mobility network. Special emphasis is given to Layer 2 protocols and signaling traffic. User traffic is accurately

modeled based on real LTE network traffic traces from one of the main tier-1 operators in the US. We use Firecycle to particularly test for vulnerabilities in the EPC due to control plane signaling.

Firecycle is designed to be scaled over the cloud. The network under analysis can be divided into a number of portions, with each one running on a separate VM in the cloud. VMs intercommunicate over IP, enabling an arbitrarily large network that virtually behaves as though it were running over a single machine. Based on this scalability property, we used this test bed to analyze and quantify large-scale security attacks against an LTE network originated from a botnet of UEs. We have also leveraged Firecycle to study the scalability of the IoT and M2M systems over LTE mobility networks and to prototype security mitigations and attack detection schemes, which are analyzed and compared against each other to improve the security of the design. The results of these independent studies are presented in Chapter 4 and Chapter 5.

Chapter 4

Scalability of the Internet of Things on LTE Mobile Networks

4.1 Introduction

Communication networks are rapidly evolving with the surge of applications and services that require no human intervention, extending connectivity far beyond personal computers, smartphones, and tablets. The rapid increase in the number of embedded devices connected as part of Machine-to-Machine (M2M) systems is providing a new dimension to wireless networks. The pervasive presence of this great variety of network-enabled objects that are able to interact with each other is driving the advent of the Internet of Things (IoT) [35].

The surge of M2M embedded devices, with billions of them expected to join communication networks within the next few years [56], will define the foundations of a smart environment where anything is possible. From Internet-connected fridges [90] to remote healthcare [49], the IoT is fueling the advent of novel communication systems and services. Nevertheless, the rapid growth of the IoT, which resulted in the number of connected devices topping the world's population in 2013 [55], is acknowledged as one of the main foreseeable challenges for communication networks

[19]. There is particular interest in understanding the potential impact this surge of M2M devices will have on mobile cellular networks.

The LTE [10], defined since Release 8 of the 3GPP standard, is the latest cellular technology designed to provide advanced communication services to mobile devices. LTE has been designed for greatly enhanced capacity and spectrum efficiency at the RAN and with a more flexible IP-only architecture at the EPC. This emerging mobile technology is expected to play a key role in the emergence of the IoT [48]. In this context, the traffic characteristics of many M2M communication systems are known to be substantially different to those from smartphones and tablets [74]. Mobile networks, which were designed and optimized to transport human-originated traffic, suffer from network and radio resource utilization inefficiencies when handling M2M communication [61].

There is concern and increasing interest in the industry to understand and forecast the scalability dynamics of M2M growth on LTE networks, which could be overwhelmed by the surge in both traffic and signaling load [64]. Given the scale and expected number of connected devices, the standardization community has identified potential scenarios that could result in severe signaling overload at the EPC [9], which may negatively affect the Quality of Service (QoS) of mobile users. Moreover, there is great interest in the security implications of M2M on cellular networks and the potential impact of botnets of compromised devices.

We introduce, to the best of our knowledge, the first realistic and detailed study aiming to provide essential insights on how the surge of M2M systems scales in LTE networks. The analysis is performed on Firecycle, a custom-built and fully standards-

compliant LTE simulation test bed described in Chapter 3. Firecycle leverages OPNET's System-in-the-Loop in order to provide a full-scale simulation of an arbitrarily large network. It implements highly realistic statistical M2M traffic models for six popular M2M device types, such as telemedicine and smart grid. These traffic models are derived from fully anonymized LTE IP traces from one of the main tier-1 providers in the Unites States. In order to investigate the potential impact of the scale of the IoT against human mobile users, a similar statistical traffic model is derived from fully anonymized LTE traces from popular smartphones from four of the top-selling manufacturers.

Based on this simulation study, we observed that the LTE signaling traffic induced by M2M systems increases linearly with the device population as opposed to a greater increase that could be very challenging for mobile networks. Moreover, in the case of certain M2M device categories, such as asset tracking, the signaling load amplifies substantially faster than with other M2M categories as the number of devices increases. The results indicate that specific M2M categories could be substantially challenging as the IoT scales up on LTE networks. These results are very valuable for industry, academia, and the standardization community as they could be useful for the initial stages of the design and standardization of 5G wireless communication systems, which consider the scalability of the IoT as one of the main goals and challenges [79].

4.2 M2M systems over LTE cellular networks

LTE mobility networks are designed to provide a capacity and spectrum efficiency increase to support a large number of connected devices. Although currently a substantial percentage of current M2M cellular ecosystems is based on legacy second and third generation (2G and 3G) networks, LTE is acknowledged to be the main platform for the emergence of the IoT [48]. Moreover, legacy GSM networks were already shut down by AT&T at the end of 2016 [11]. In parallel, the traffic characteristics of many IoT applications, substantially different than traffic from smartphones and tablets, are a potential source for network resource utilization inefficiencies [74]. As a result, there is concern regarding the potential impact of M2M systems on the regular operation of LTE networks, which may be overwhelmed by the surge in both traffic and signaling load [64].

Cellular networks are configured to transport user communications originating at smartphones that have well-understood traffic characteristics [75]. However, the network utilization dynamics and traffic characteristics of M2M systems are very complex. M2M traffic differs substantially depending on the type of M2M system. For example, certain categories of devices have a strong imbalance between uplink (UL) and downlink (DL) traffic, with a clear dominance of UL traffic as opposed to smartphone traffic. These diversified traffic characteristics of M2M systems are one of the main challenges for the scalability of the IoT on LTE mobile networks [64].

The emergence of the IoT also brings signaling implications to the EPC. Based on stringent admission control mechanisms, the mobility network utilization needs to be

optimized. Each transaction or traffic flow from/to an M2M device results in signaling at the EPC to establish the required radio resources [4]. Unnecessary connection establishments could potentially overburden the core network. This negative impact of signaling load has already been observed in the wild in certain mobile operators [22], citeDocomo, [76], [33]. Given the expected surge in the number of M2M connected devices, the resulting growth in signaling could potentially overload the EPC, which is acknowledged by the standardization community as a potential threat against LTE networks [9].

The following section presents a brief overview of the main LTE signaling procedures involved in the operation of the network and management of connectivity from and to mobile terminals.

4.3 LTE signaling procedures

In order to operate LTE mobile networks, a series of signaling procedures, known as Non Access Stratum (NAS) functions [73], must be executed. This section briefly overviews the main signaling procedures.

Initial NAS Attach

When a mobile terminal is switched on, it must execute certain steps so it can reach the connected state. During this time, a point-to-point IP default bearer is set up to provide communication between the UE and the PGW. At this point an IP address is assigned to the UE. The NAS attach procedure contains a number of steps. Firstly, the UE performs a Cell Search so it can acquire both time and frequency

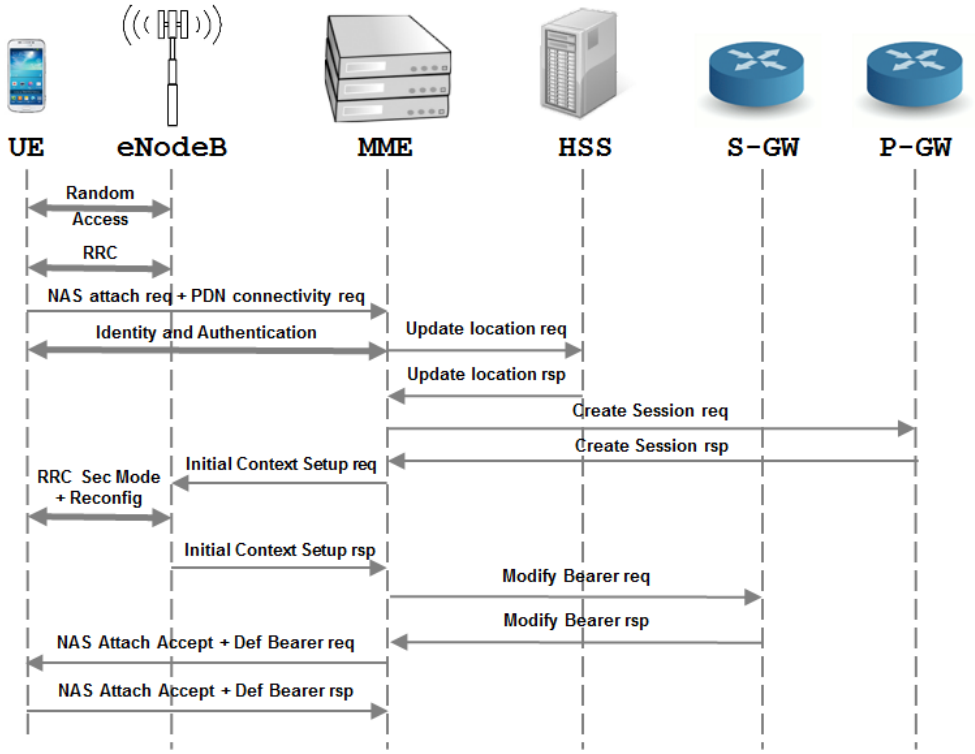


Figure 4.1: NAS attach procedure

synchronization [3]. Then, the Random Access procedure assigns radio resources to the UE so it can set up a RRC connection with the eNodeB. The next step is to execute the NAS identity/authentication procedure between the UE and the MME, which in turn leverages the HSS to configure security attributes and encryption. Then, the point-to-point bearer through the SGW and PGW is set up, and then the UE's RRC connection is reconfigured according to the type of QoS requested. The overall NAS attach process is illustrated in Figure 4.1, giving a clear visual intuition of the large number of messages exchanged among EPC elements [67]. Note that the Random Access procedure, the RRC connection, and the NAS authentication involve a substantial number of messages not shown in the figure for simplicity.

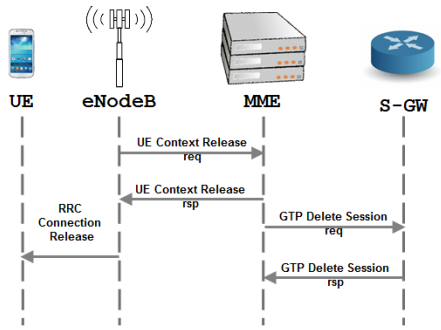


Figure 4.2: RRC state transition procedure: connected to idle

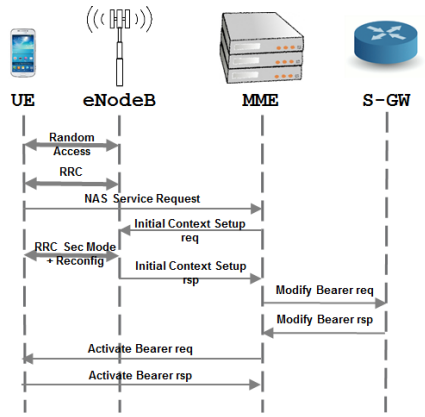


Figure 4.3: RRC state transition procedure: idle to connected

Bearer management functions

Strict radio resource management and reutilization techniques are necessary given the scarcity of spectrum and radio resources. As defined by standards [4], a UE's RRC connection is released whenever it has been observed as idle by the eNodeB for more than a few seconds (often between 8 and 10 seconds). Along with the connection release, the radio resources associated with the UE are freed to be reused for any other UE. This process involves a number of messages among EPC nodes to transition a UE from connected to idle state and vice versa. These are displayed in Figure 4.2 and Figure 4.3, respectively. Any mobile device in idle state must be transitioned to a connected state in order to be able to transmit or receive data. As seen in Figure 4.3, this involves a similar procedure to the NAS attach.

Paging

Paging is required to address mobile terminated calls to their destination (i.e. the receiver). Upon an incoming communication addressed to a UE, the location of the user, in terms of cell, must be determined. In the case this location were

known a priori, mobile terminals would be required to update their location with the HSS – a costly operation – each time they moved to a new cell, resulting in an excessive amount of location update-related signaling. Therefore, the UE's location is only known with a much larger granularity, known as the Tracking Area (TA). Upon receiving incoming traffic for a UE k in idle state, the MME triggers the broadcast of a paging message over each cell within the TA of k [5]. UE k then replies to the paging, disclosing its precise location in terms of cell. Then, the appropriate bearer management functions are executed.

Handoff and mobility functions

All wireless cellular networks provide mobility, requiring specific operations to sustain connectivity as users move between cells. This is achieved by means of a handoff procedure [7], which transfers connections from one eNodeB to another as UEs move. In the specific case of localized mobility among eNodeBs interconnected through the X2 interface, the handoff signaling exchange is local at the RAN. An S1-based handoff procedure, however, involves a series of signaling messages over the EPC.

The LTE mobile network operation involves further signaling procedures not listed here. The network functionalities described herein involve NAS signaling procedures that have been acknowledged by the standardization bodies and the industry as one of the main challenges for M2M scalability and the potential cause of signaling overload in LTE [9].

4.4 M2M scalability study

This section presents the main results of the scalability study of the IoT over LTE networks. The main goal is to provide insights on the potential impact that the surge of M2M devices could have on both the network and the QoS of smartphone users. Both the traffic models and the methodology followed to obtain and plot the results and discussed.

Traffic models

In order to obtain very accurate results of the interaction of M2M systems communicating over an LTE network, the simulated traffic of legitimate devices (both M2M embedded devices as well as smartphones) is based on highly realistic statistical traffic models. These models are derived from fully anonymized observations of real LTE traffic traces from one of the main tier-1 operators in the US. Both fully anonymized Call Detail Records (CDRs) and fully anonymized IP traffic flow metadata from exclusively LTE IP traffic are processed to derive statistical models of data traffic over LTE for typical smartphones and M2M devices. The probability distributions of the model are formulated from observed traffic characteristics such as number of uplink/downlink packets, time between packets, packet size, time between sessions, and session size.

In order to generate realistic models from popular smartphones, we process anonymized data from the latest phone models from four of the main smartphone manufacturers. We then analyze anonymized data for six typical M2M device types:

1. *Asset tracking*,
2. *Smart grid*,
3. *Personal tracking devices*,
4. *Tele-medicine*,
5. *GPS navigators*,
6. *Remote alarming systems*.

This allows for very realistic simulation of LTE device communication and interaction with a mobility network as compared to other simpler and arbitrary statistical traffic models used in the literature for similar works. All of our experiments were performed on Firecycle, described in Chapter 3. Although simulation studies have limitations, the results obtained with the test bed are as realistic as possible without performing the experiments on an actual LTE network, which is not possible given the scale of the problem under analysis.

Methodology of the analysis

It is important to highlight that the goal of the analysis is not to realistically simulate the absolute impact of M2M on a particular LTE network deployment, but rather to present valuable insights on the potential impact and scalability of the IoT over LTE. The goal is to provide an analysis of the potential scalability issues of the IoT, not to provide realistic results indicating how a large population of M2M devices would saturate the network, and what would be the specific impact. As a result the analysis herein presented is governed by a very basic but strict methodology.

All network and device population scenarios are purely generic and not specific to any given network architecture. The configuration in terms of computing power, link capacities, and actual architecture is arbitrary, aiming for generic results. Nevertheless, the security test bed is standards-compliant and accurately simulates the performance and behavior of a real LTE network. The experiments clearly demonstrate the potential scalability impact of M2M traffic and, more importantly, the potential bottlenecks or heat points in the network. However, absolutely no specific information can be inferred on the actual M2M load one would see in a particular real LTE commercial implementation.

All the results in Section 4.4 are normalized by the same arbitrary scalar in order to provide a comparative analysis rather than a quantitative analysis. The simulated network is generic, with one EPC instance (MME, SGW, PGW, and HSS) processing IP connectivity between UEs and an external server. The capacity of the server is assumed to be infinite in order to not interfere with the measurements at the EPC. The LTE RAN is modeled with the capacity and specifications of a standard 10MHz LTE deployment [73], and is composed of 10 cells.

Results

A series of simulation experiments are run in which a number of M2M mobile terminals send and receive traffic according to the statistical traffic models, implementing M2M devices from each M2M category. The number of devices is scaled up, ranging from 125 to 4000 UEs. A variety of statistical information is recorded and plotted in

order to analyze how each category of devices scales in terms of both signaling and data load on the EPC. Further experiments are run to extract insights on how the scalability of M2M systems could potentially affect the QoS of smartphone users.

Signaling Load

As discussed in Section 4.3, traffic flows between a device and host require the establishment of resources in the case of a device initially in idle RRC state. Throughout the duration of a session or flow, the mobile device might stay in connected state or might switch states back and forth depending on the time in between packets. The standardization community acknowledges the signaling traffic increase induced by M2M systems as a potential threat to LTE networks [9]. A massive increase in the M2M population could potentially impose strain on the EPC by means of this signaling traffic spike.

The first experiment examines MME-SGW link load and MME CPU utilization to discern the signaling impact of scaling M2M devices, and the results are plotted in Figure 4.5 (a) and (b). Intuitively, the M2M categories that have the highest impact on the MME-SGW link utilization, asset tracking and personal tracking devices, also place the largest strain on the MME CPU utilization. Focusing on these two device categories, one can observe that, although they generate roughly the same MME-SGW signaling load for 125 devices, the load increases much more rapidly for asset tracking as the number of devices scales. According to Figure 4.5 (a), 2000 asset tracking devices produce 42.6% higher signaling load than do the same number of personal tracking devices, for example. A slightly similar pattern is observed for the MME CPU. Specifically, 125 personal tracking devices produce about the same

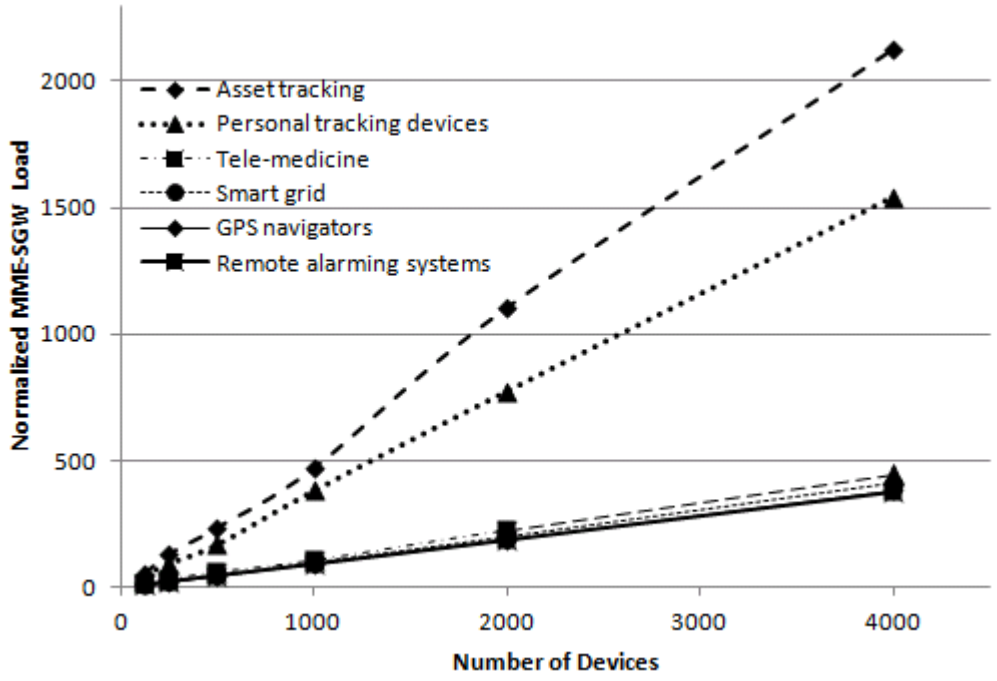
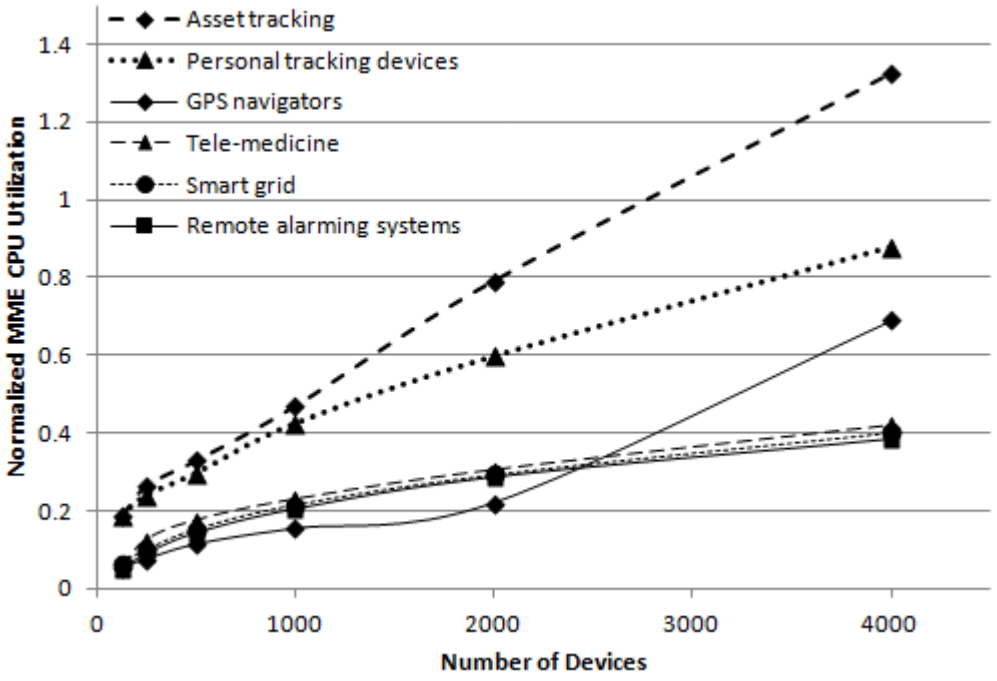


Figure 4.4: M2M scalability signaling impact: (a) normalized MME-SGW load



(b) normalized MME CPU utilization

MME CPU utilization as 125 asset tracking devices, but 2000 asset tracking devices incur 32.5% higher load than the same number of personal tracking devices. For 4000 M2M devices, asset tracking are 50.5% higher. It is also interesting to notice the sharp increase in MME CPU utilization as the number of GPS devices scales above 2000. When increasing the number of devices by a factor of 8 from 250 to 2000, the MME CPU utilization increases 1.99 times, yet when only doubling them from 2000 to 4000, the load increases by 2.14 times.

Overall, the results indicate a clear signaling load spike as the M2M device population scales up. However, the signaling increase appears to be linear and, with the exception of GPS devices, constant. This result, although based on simulation, can be considered as good news since a larger signaling load increase could indicate a great challenge for mobile networks to cope with the scalability of M2M systems.

In order to quantitatively determine the M2M device category that generates the largest signaling increase, Table 4.1 lists the MME CPU utilization and MME-SGW link load gradients. The gradient for each category has been averaged as the number of devices scales from 125 to 4000. According to the results, asset tracking and personal tracking devices induce an increase in MME CPU utilization approximately 1.5 to 2.5 times faster than devices belonging to the remaining categories. As for MME-SGW link, the load from asset tracking grows up to 17.88 times faster than the others.

These results indicate that asset tracking devices in general could potentially be one of the most challenging communication systems over LTE mobile networks, especially if the number of M2M devices corresponding to the asset tracking category

M2M category	Avg. MME CPU gradient	Avg. MME-SGW load gradient
Asset tracking	0.00035	1.66893
Smart grid	0.00015	0.24562
Personal tracking	0.00024	0.39330
Tele-medicine	0.00020	0.34757
GPS navigators	0.00014	0.12717
Remote alarming	0.00016	0.09333

Table 4.1: Average MME CPU utilization and MME-SGW load gradients

increased substantially up to millions or even billions of new connections. The signaling load stemming from the operation of M2M devices is due, in part, to frequent RRC state transitions. Many categories of M2M systems are characterized by small data bursts, both in the uplink and the downlink, at frequent intervals. The amount of time in between data bursts and communication sessions directly influences the signaling load at the EPC. For example, if data sessions recur at intervals slightly longer than the network's RRC timeout timer, the EPC will bear much more signaling load as the devices constantly transition between idle and connected states for each traffic burst. This is theoretically known to pose a potential threat against the network [47].

In order to investigate the scalability of M2M systems over LTE, it is important to determine how efficiently the network resources are used. To do so, we investigate the average number of RRC state transitions that are induced by each M2M system

M2M category	Connected-to-idle transitions per normalized traffic unit
Asset tracking	0.15593
Smart grid	45.67325
Personal tracking	5.37232
Tele-medicine	9.22127
GPS navigators	23.63992
Remote alarming	54.96471

Table 4.2: Number of connected-to-idle RRC state transitions per M2M traffic load

to communicate a fixed amount of data. Table 4.2 lists the number of idle transitions per normalized data load unit (adding uplink and downlink) in a scenario with 1000 devices. The results are computed for each M2M traffic category. Although smart grid and remote alarming systems devices produce less signaling load at the MME than asset tracking and personal tracking devices, as seen in Figure 4.5, due to their small transactions size occurring at infrequent intervals, they utilize the network most inefficiently out of all the device categories we examined.

Data Load

M2M systems are very diverse in the amount of data they transmit and receive, and often there is an imbalance in the amount of uplink and downlink traffic [74]. This is challenging for the designing of mobile network architectures that must tackle the heterogeneity of the IoT in an efficient manner. As part of this analysis, this chapter also aims to provide insights not only on the scalability of data traffic produced by each device category, but also the indirect impact the spike in data traffic could

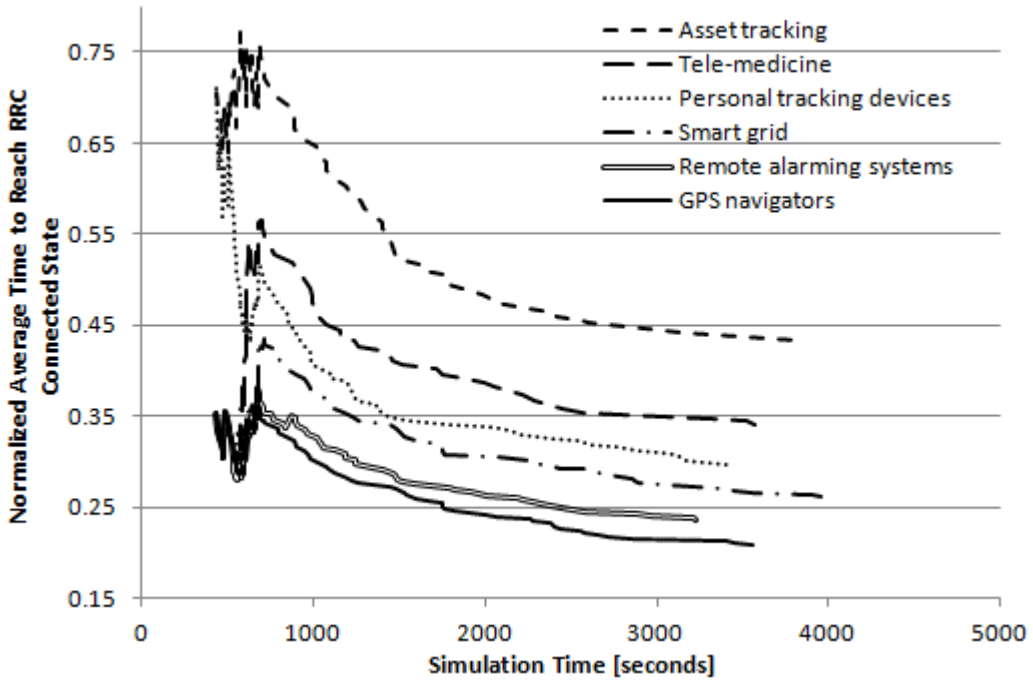


Figure 4.6: Normalized average latency to reach RRC connected state for smartphones during traffic of 1000 M2M devices

potentially pose on specific network elements.

Similarly to the analysis of the M2M signaling load scalability, we obtained results to determine the rate at which data traffic scales up as the M2M population of each category increases. Figure 4.7 examines the normalized load in the SGW to PGW link for this situation. Note that this link aggregates all the data, both uplink and downlink, for all the devices in the simulated network. As with the signaling scalability plotted in Figure 4.5 (a), asset tracking and personal tracking devices appear to generate the potentially largest increase in data traffic load. This further indicates the heterogeneity of the IoT, with certain embedded device categories scaling both signaling and data load at a much faster rate than others, in which load increases very slightly.

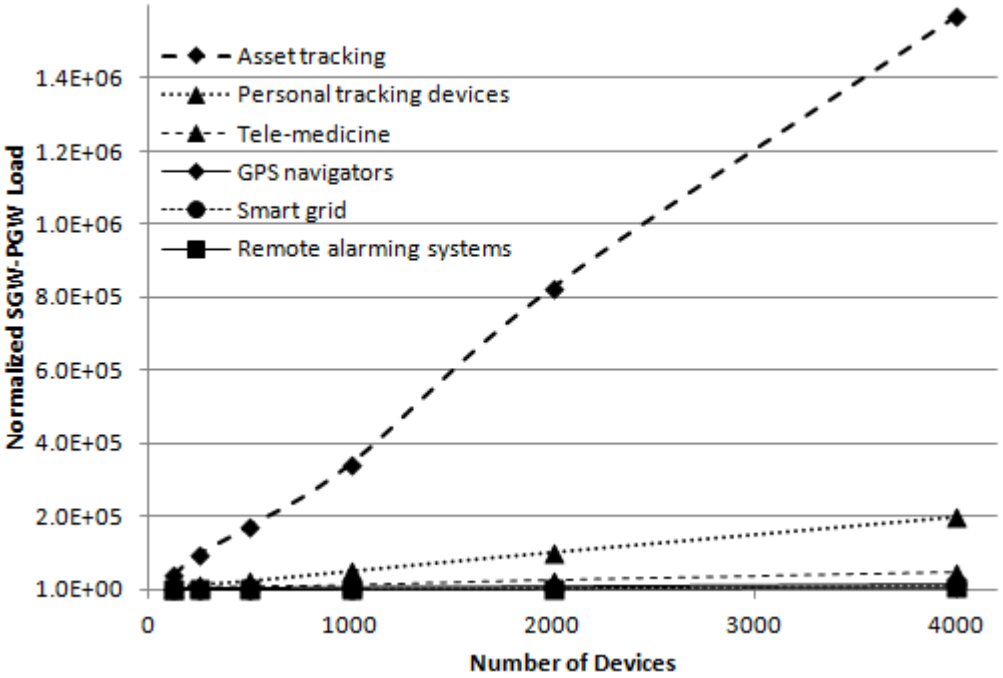


Figure 4.7: M2M data traffic load scalability

M2M category	Avg. UL traffic gradient	Avg. DL traffic gradient
Asset tracking	191.187	201.247
Smart grid	1.565	0.646
Personal tracking	43.973	6.279
Tele-medicine	11.284	0.590
GPS navigators	2.287	1.355
Remote alarming	0.689	0.908

Table 4.3: Average UL and DL data traffic load gradients

Table 4.3 summarizes the results for the increase in uplink and downlink data traffic load for each M2M category. Similarly to Table 4.1, the gradient of the data load increase is averaged as the population scales from 125 to 4000 UEs. The results

indicate that asset tracking devices increase the data traffic load one to two orders of magnitude faster than the majority of the other categories. Note that the scalability of data traffic also indicates that asset tracking could potentially be one of the most challenging M2M systems as it scales over LTE mobile networks. The asset tracking load increase is somewhat balanced in the uplink and downlink, as opposed to personal tracking devices, which have a substantial increase particularly in the uplink.

The scalability of the IoT over LTE mobile networks presents further challenges based on this largely imbalanced ratio between uplink and downlink traffic of certain embedded devices. Unlike smartphones, with a predominant downlink load, different types of M2M systems operate with highly uplink or downlink imbalanced data traffic. Figure 4.8 plots an example of this diversity in the imbalance between uplink and downlink traffic. In this example, one can observe asset tracking as a system with somewhat balanced traffic compared to personal tracking devices, which have 16 times more traffic in the uplink. This imbalance is due to the normal operation of personal tracking devices, which mostly transmit uplink periodic messages with a location update. Further details on uplink to downlink ratio of M2M traffic can be found in [74].

Impact on QoS

EPC nodes that are burdened with a large amount of signaling induced by the scaling of M2M devices could potentially yield negative consequences for other mobile users. One of the goals of this chapter is to determine how the QoS of smartphone users could be affected by an increasing number of M2M devices. Figure 4.6 illustrates the average latency experienced during RRC state transitions for a small set

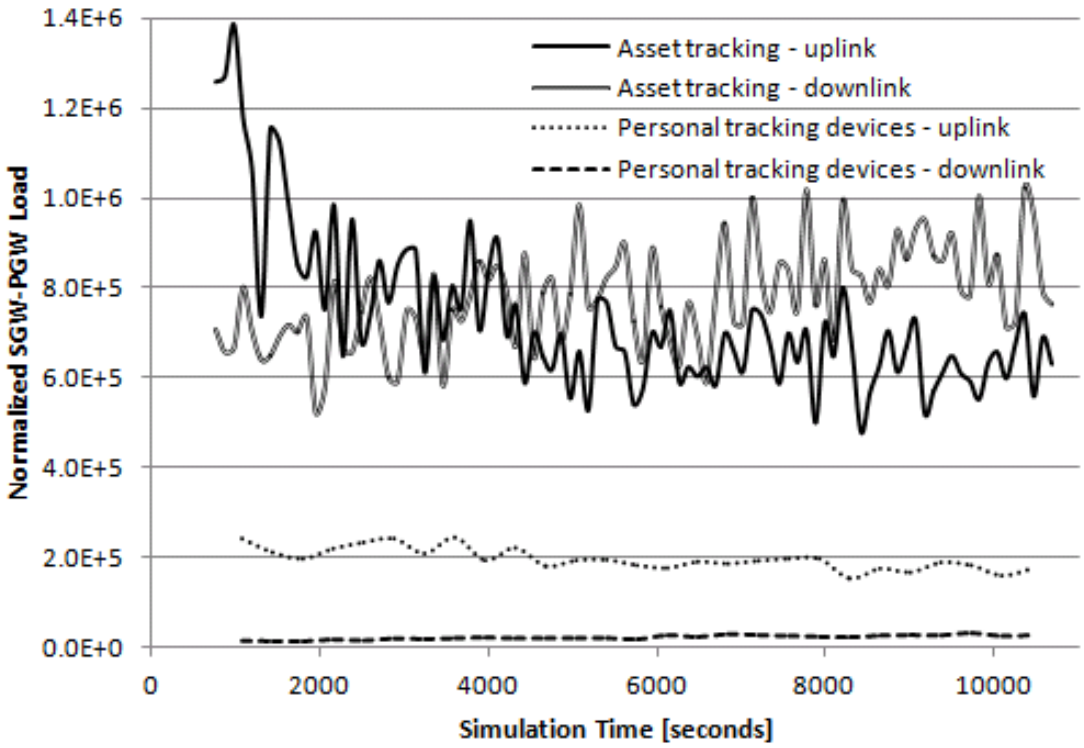


Figure 4.8: Normalized UL vs. DL SGW-PGW data load for personal and asset tracking devices

of smartphone users engaged in activity at the same time as 1000 M2M devices. The same experiment is run for each one of the six M2M categories under analysis. Note that, due to the implementation of the test bed, all UEs present in a simulation must do an initial attach before starting to communicate. Therefore, we start gathering results for smartphone QoS seven minutes into the simulation so that the QoS results are not unintentionally affected by the signaling load spike at the onset of the simulation when the 1000 M2M devices attach to the network. Note also that the QoS metric spikes initially because all the smartphones are attaching at the same time starting at minute seven.

M2M traffic category	Normalized average uplink packet latency			Average gradient
	500	1000	2000	
Asset tracking	0.00141	0.00516	0.00697	0.00020
Smart grid	0.00146	0.00147	0.00157	0.00006
Personal tracking	0.00144	0.00145	0.00158	0.00007
Tele-medicine	0.00155	0.00146	0.00194	0.00015
GPS navigators	0.00484	0.00486	0.00152	0.00002
Remote alarming	0.00150	0.00148	0.00174	0.00011

Table 4.4: Normalized average uplink packet latency experienced by smartphones with presence of scaling number of M2M devices, ranging from 500 to 2000

Each category of M2M traffic has a different influence on smartphone state transition latency, with the asset tracking category having the highest impact. Once the latencies reach a steady state, the asset tracking devices impose a latency that is more than double the one produced by GPS navigators, the category with the lowest impact. One hypothesis for the low signaling produced by GPS navigators is their use of map caching and hence infrequent and sometimes unpredictable location updates. It has been shown in [69] that users are uncertain when location queries are made, also leading to legal ramifications. Although the RRC state transition latency increases substantially, the values reached under the load influence of 1000 M2M devices are still low and would most likely not result in a noticeable degradation of QoS. However, the QoS of smartphone users may be substantially degraded with a larger population of M2M embedded devices, especially if the forecasts of billions of connections are reached.

To provide insight on how the QoS of smartphone users is impacted by scaling M2M devices, Table 4.4 displays the average normalized latency of uplink packets experienced by 50 smartphones as the number of M2M devices continuously doubles,

ranging from 500 to 2000. It also shows the average gradient of the latencies over the scaling range. Due to the high impact that asset tracking devices impose on EPC node CPU and link throughput as we have seen throughput this chapter, this category also induces the highest increase in smartphone uplink packet latency of all the categories we examined. In comparison to the latency caused by GPS navigators, the latency from asset tracking devices increases 10 times faster. As with RRC state transition latency, the uplink packet latency values produced from 2000 M2M devices is not particularly high with this number of devices, however they rise significantly as the number of M2M devices scales up. A sufficiently high amount of M2M traffic could therefore potentially cause latencies beyond the tolerance threshold of applications, particularly those with strict delay requirements.

4.5 Conclusion

The number of M2M devices using cellular communication services is expected to surge over the next few years. Such a steep expansion in the IoT presents challenges for cellular infrastructure that must now cope with an abundance of devices whose behavior differs substantially from the traditional and well-studied smartphones and tablets for which the current systems were designed. This chapter presents the first insights on the scalability of the IoT over LTE mobility networks, with a focus on the Packet Core, determining the potential impact of the surge in both signaling and data traffic load as the number of connected devices increases. The chapter also studies the potential impact that the spike of embedded devices communicating over LTE networks could have against the QoS of smartphone users.

This analysis work, which has been performed on a custom-built, standards-compliant LTE simulation test bed, realistically models six popular categories of M2M devices with statistical traffic models derived from fully anonymized LTE traffic traces from one of the main tier-1 providers in the US. Based on the results obtained, it is determined that the signaling and data traffic load appears to scale up linearly as the number of connected devices increases. This is good news, as it implies that a signaling storm should not be expected. Nevertheless, certain M2M device categories, such as asset tracking, exhibit a much faster signaling and data traffic load increase. This indicates that certain M2M communication systems will present a larger challenge to LTE mobile networks.

Chapter 5

Signaling Overload in LTE Networks

5.1 Introduction

Despite the most recent enhanced design for the cellular core network, next-generation LTE networks still heavily rely on legacy inefficient circuit-switched architectures at the EPC [43]. This is partially due to the scarcity of radio resources and spectrum, which force cellular deployments to free and efficiently reuse radio resources that appear idle for some time. The large amount of network signaling messages required to establish, release, and manage the point-to-point circuits (known as bearers in the context of LTE) between the mobile devices and the cellular gateway towards external Packet Data Networks (PDNs) is very large. This circuit-switched approach is known for increasing the threat of signaling traffic overloads, and is acknowledged by the standardization community itself as a risk to the cellular packet core [9].

The majority of recent cellular network security research has focused mainly on legacy GSM (Global System for Mobile Communications) and UMTS (Universal Mobile Telecommunications System) networks. In this section we extend the analysis of such threats to LTE and, to the best of our knowledge, we are the first to realistically implement and analyze the impact of such attacks. The results presented yield very interesting insights on the potential impact and the scalability of signaling-based at-

tacks against the EPC, which are shown to spike the signaling load at certain nodes by over 295 times. We identify as well the main congestion point and vulnerability that each type of attack under analysis is exploiting and how certain attacks can be tuned, based on the specific configuration of the target network, to maximize the impact.

The analysis of signaling overload attacks is important to understand the resiliency of the current LTE network. However, this section also suggests a paradigm shift in mobile network design that is of great value. Efforts should be made to rethink the architecture of a mobility network, shifting away from legacy and inefficient circuit-switched architectures [38, 43]. While most of the technological innovations are being done at the RAN, the inefficient networking architectures used in cellular core networks are throttling down the performance of mobile communications. After evaluating the extent to which the core network serves as the bottleneck in the current architecture, this section discusses the potential benefits of a fully packet-switched IP-based EPC and presents results that indicate the benefits and performance improvements that could be achieved.

The novelty of this work is not in proposing new attacks and solutions, but in the implementation and analysis of the potential impact of such attacks for the first time. We carry out a dynamic simulation and analysis based on the signaling interactions with the network elements. Previous analysis of these attacks has been mostly static, based on link capacity and signaling bit rates, and mostly focusing on legacy mobile networks. Moreover, core network architecture redesigns have been proposed and implemented, but not in the context of mobile network security.

This chapter presents the following contributions:

1. We implement and test signaling-based security vulnerabilities and attacks against next-generation LTE networks on a realistic test bed
2. We present a simulation-based analysis of the scalability and potential impact of signaling-based threats against the LTE EPC, determining the specific nodes and links in the network that are most severely affected. We also evaluate the potential impact of each attack on customer Quality of Service (QoS). Realistic smartphone traffic is modeled from anonymized LTE traffic traces from one of the main tier-1 operators in the United States in order to determine the impact of these attacks against legitimate users
3. We evaluate the potential security benefits of a mobile network architecture redesign, transitioning to a fully packet-switched architecture of the EPC which maintains QoS for users during an attack while providing strong potential security and performance enhancements

5.2 Mobile malware and signaling overloads

Arguably, the most efficient way to create a botnet of mobile devices is by means of a malware or trojan infection. Different malware-related problems with mobile phone applications, especially on Android platforms, have made it to the headlines over the last couple of years [14]. Although mobile malware is often motivated by either dissemination of spam campaigns [71] or fraud [70], such malware infections could potentially have adverse effects for the cellular network. If aimed to disrupt

communications, the reported numbers of, for example, Android devices that were infected in 2013 (with over 30,000 new malware samples just within this year) could comprise a very large botnet capable of attacking the network [52]. There are several well-studied avenues to infect mobile devices and spread malware in order to multiply the botnet, such as "contact" propagation via Bluetooth and transmission over text messages containing a link to a malicious application [26].

In recent months we have also seen several attacks stemming from the Mirai bot, which infects IoT devices and uses them for large-scale DDoS attacks. Because it can infect such a large number of devices, the Mirai botnet has been used in some of the most disruptive DDoS attacks, which have scaled up to 1 Tbps. Some of the recent attacks were on computer security journalist Brian Krebs's site *Krebs on Security* in September 2016, an attack on OVH, and the attack in October 2016 on Dyn [45]. Mirai performs a wide-range scan of IP addresses to locate IoT devices operating on easily guessable login credentials. A brute force approach is used to guess passwords, typically factory default usernames and passwords.

Once a mobile botnet is assembled, it can be dormant until the attacker triggers specific actions over a command and control channel. In the context of mobile devices, there are multiple options to carry such command messages, such as text messages, phone calls from a specific originating number, or data communications. Moreover, the malware infection could have a specific date and time to initiate an attack without requiring direct communication with the attacker.

It is important to note that 3GPP also considers other potential origins for a signaling overload, such as a node failure [9]. In the event of an EPC node failure, a

large number of UEs would be disconnected from the network and potentially generate a surge of signaling messages upon re-attaching to the same or an alternative EPC node. Considering that a successful attack could result in an EPC node failure, the impact of such threat could potentially be enhanced when the failed node recovered and a surge of UEs attempted to reattach or if the victim UEs attempted to attach and flooded an alternate node.

5.3 Experiments and results

Methodology

The goal of the analysis herein presented is not to realistically simulate the actual impact and effect of signaling overload attacks against a real LTE network deployment, but rather to give insights on the potential impact, scalability, and optimization of the threat. The goal of our security research is to enhance the security of LTE networks, not to provide realistic results indicating how an attack against the EPC could be launched, what would be the most effective attack, and what would be its specific impact. Therefore, we have defined a very basic but strict methodology for our security analysis.

Although the security test bed is standards-compliant and accurately simulates the performance and behavior of a real LTE network, the scenarios simulated in this chapter are generic and not specific to any network architecture. The configuration of the simulated network in terms of processing power, link capacities, and architecture

is arbitrary in order to provide generic results. Our simulation-based experiments demonstrate the threat of signaling-based attacks and, more importantly, the potential optimization and tuning of such attacks to maximize their impact. However, no specific information can be inferred on how to launch the attack and, more importantly, what is necessary to carry out a successful attack (i.e. how many bots or infected devices and how they should be geographically distributed).

As a result, the simulated LTE network under analysis is generic and consists of one instantiation of the EPC, including the MME, SGW, PGW, and HSS, which connects over an IP network to an external server. In order to not interfere with the measurements at the EPC, we assume that the server the mobile terminals connect to has infinite capacity and is therefore able to serve all the UEs in our network. Note that this is a realistic assumption considering that millions of mobile terminals connect constantly to all types of applications being served from the Internet or some cloud infrastructure. The LTE RAN is modeled with the same capacity as a standard 10MHz LTE deployment [73] and is composed of 10 cells that communicate with the EPC.

Moreover, all the results presented herein (link utilization, node utilization, message loads, CPU loads, etc.) are normalized. The aim is to provide a comparative analysis rather than a quantitative analysis. Each operator's network is configured differently and might be equipped with different vendors and architectures. Therefore, our concern is not to quantify the exact impact of an attack against a specific network, but to determine how the impact scales up if, for example, the number of attackers doubles. Also, we aim to determine what links or nodes are most affected

given each attack category. This information, while highly valuable for the security research community, cannot be directly utilized by an attacker. For example, perhaps one of the attacks analyzed affects the S6a interface, and by doubling the number of attacking devices, the impact scales by a factor of, say, 10. However, whether 10, 10,000 or 10,000,000 infected devices are necessary is not disclosed.

In summary, the results presented in this chapter give insights on *what* could happen and why we should reconsider the current architecture. However, an attacker cannot learn *how* to make it happen.

Signaling amplification attack due to frequent RRC state transitions

In our experimental study we implement a botnet of rogue mobile devices in an LTE network that are leveraged to launch a signaling-based attack against the LTE EPC similar to the one proposed in [47]. The attack is originated by a variable number of mobile devices that induce constant RRC state transitions between idle and connected states. Our simulated LTE network is configured with an RRC idle timer (T-idle) of 12 seconds. Note that although it is always in the range of 10 seconds, the value of this timer may be different from operator to operator.

The attack, which can be server-initiated or UE-initiated, consists of repeatedly sending one data packet either to (server-initiated) or from (UE-initiated) each member of the botnet. If a packet is sent by the server and arrives at the core network for a device that is in idle state, the SGW will initiate a paging procedure to bring

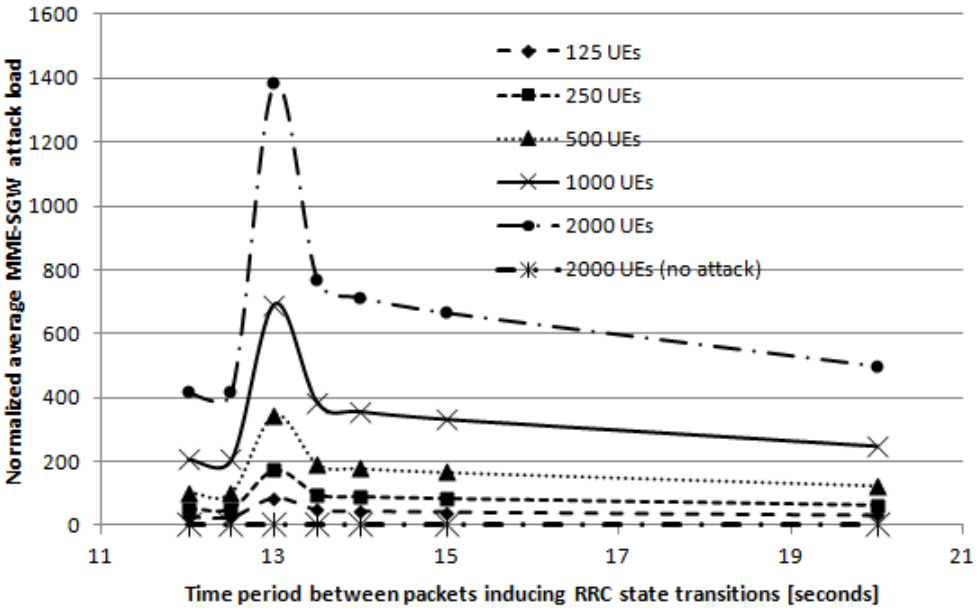


Figure 5.1: Normalized MME-SGW load averaged over the attack duration for a server-initiated RRC state transition attack

the UE back to connected state. Similarly, if a UE that is in idle has a data packet to send, this device must execute a series of signaling messages in order to reconnect and send the data. In the case of the packets being sent by a UE, its recipient is the same external server. The data packets are repeatedly sent every T seconds, with T = T-idle + Δ. We run the experiment with a variable value for Δ, ranging from 0 seconds to 8 seconds, in order to determine the value of Δ that maximizes the impact of the attack.

Figure 5.1 plots the normalized load experienced at the S11 interface between the MME and the SGW averaged over the duration of this attack when it is server-initiated. The attack is simulated for a variable number of UEs contributing to the botnet and for a variable time in between periods of data packets. As a reference point, we simulated a scenario with 2000 UEs running only the smartphone traffic

model derived from anonymized traces from a real LTE network (indicated by *no attack* in Figure 5.1). Based on these results, one can observe the clear increase in load at the EPC when the network is under attack. As expected, the larger the size of the botnet (with respect to the size and population of the network under analysis), the larger the impact of the attack. In the case of 2000 UEs, the attack increases the signaling load at the EPC by 295 times that produced by the same number of devices running legitimate smartphone traffic. It is interesting to observe that for the value of Δ that maximizes the MME-SGW load, a small botnet of 125 UEs generates signaling traffic about 19 times higher than a population of legitimate devices 16 times larger than the botnet (baseline with 2000 legitimate UEs). This gives a good idea of the potential scaling of such an attack and the impact it could have with a very large botnet.

Moreover, it is important to note that one can observe a clear load spike for a given value of Δ and, as a result, the attack period T. The attack period that induces the greatest number of pages and RRC transitions results in the largest spike in load at the MME-SGW link, maximizing the attack impact. However, this specific value is only valid in our arbitrary configuration. We prove that this attack could be optimized to maximize its impact but we do not provide any information on how to do so in a real LTE network.

The next step of our analysis is to compare the impact of this signaling-based attack at the EPC when it originates at an external server or from within the botnet itself. Figure 5.2 displays the normalized load experienced at the MME-SGW link averaged over the duration of the attack. The experiment is run for a variable number

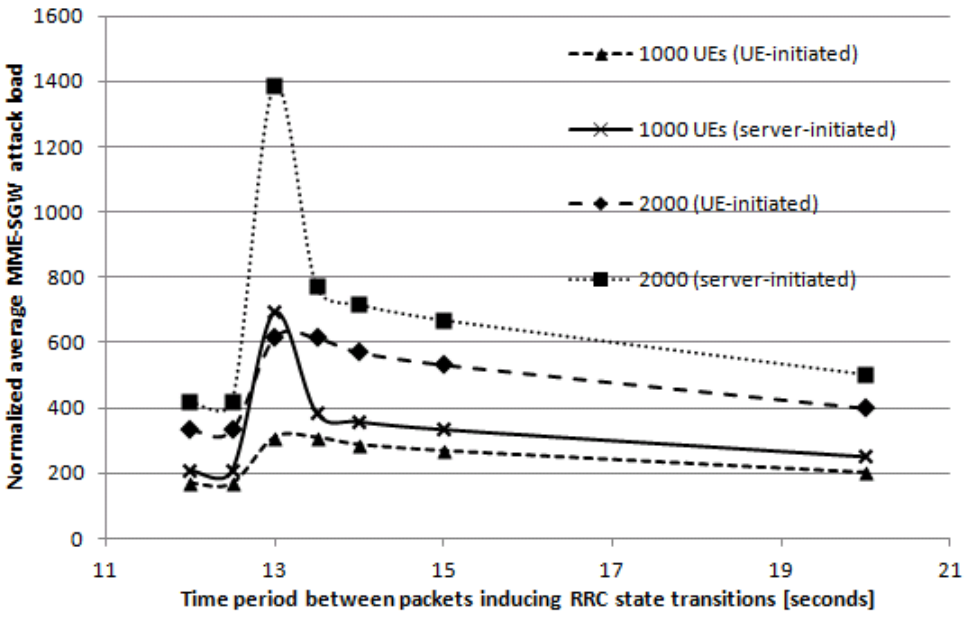


Figure 5.2: Normalized MME-SGW load averaged over the attack duration: server-initiated vs UE-initiated

of UEs within the botnet and compares the server-initiated and the UE-initiated attack strategies. The results indicate a higher attack impact when the spike in RRC transitions is originated due to the botnets receiving traffic from an external server. Further investigation highlights that this is due to the fact that, for mobile terminated traffic, the RRC state transition includes extra signaling messages for the paging procedure. Therefore, when the attack is originated by a server sending messages to a botnet in a synchronized fashion, the EPC signaling overload is enhanced by a flood of paging procedures. It is interesting to observe that when the attack is maximized and tuned to the period that induces the largest number of RRC state transitions, the load generated by a server-initiated attack is higher than the load generated by a UE-initiated attack with a botnet twice the size.

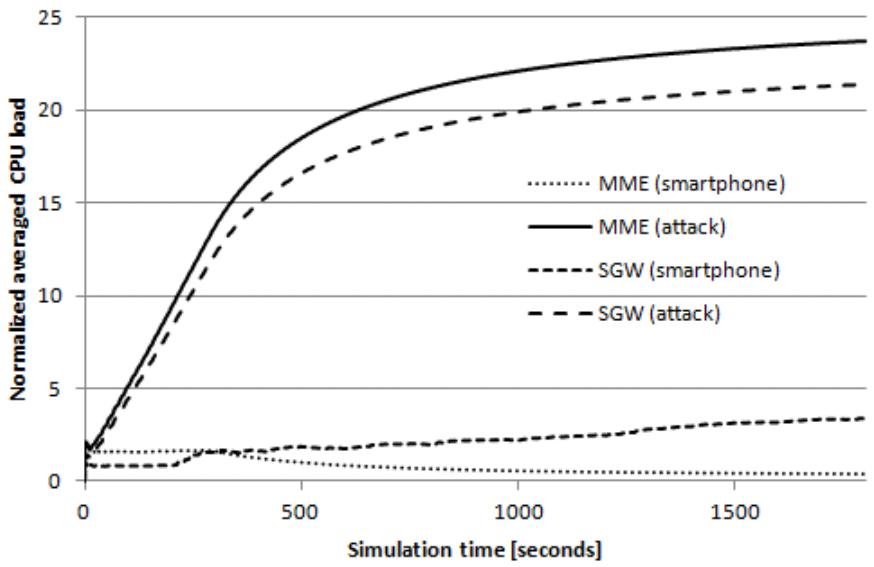


Figure 5.3: Normalized average MME and SGW CPU utilization for a server-initiated RRC state transition attack and processing legitimate smartphone traffic

As part of our analysis, we aim to determine where the most vulnerable points lie in the network under these security attacks. Some insights to this question are presented in Figure 5.3, which plots the normalized average CPU consumption of the MME and the SGW when both are processing traffic for 1000 legitimate smartphone devices compared to when these devices belong to a botnet attacking the network (UE-initiated). One can clearly observe that once the attack reaches a steady stable state, the CPU load at the MME is 59 times higher than normal and the CPU load at the SGW is about 7 times higher than usual. Moreover, our results seem to indicate that the MME is the node that is most severely affected by the attack.

In order to study the impact such an attack would have on the QoS experienced by mobile users, we simulate a botnet of 2000 UEs forcing RRC state transitions. We overlap this attack with legitimate communications of smartphone users and observe

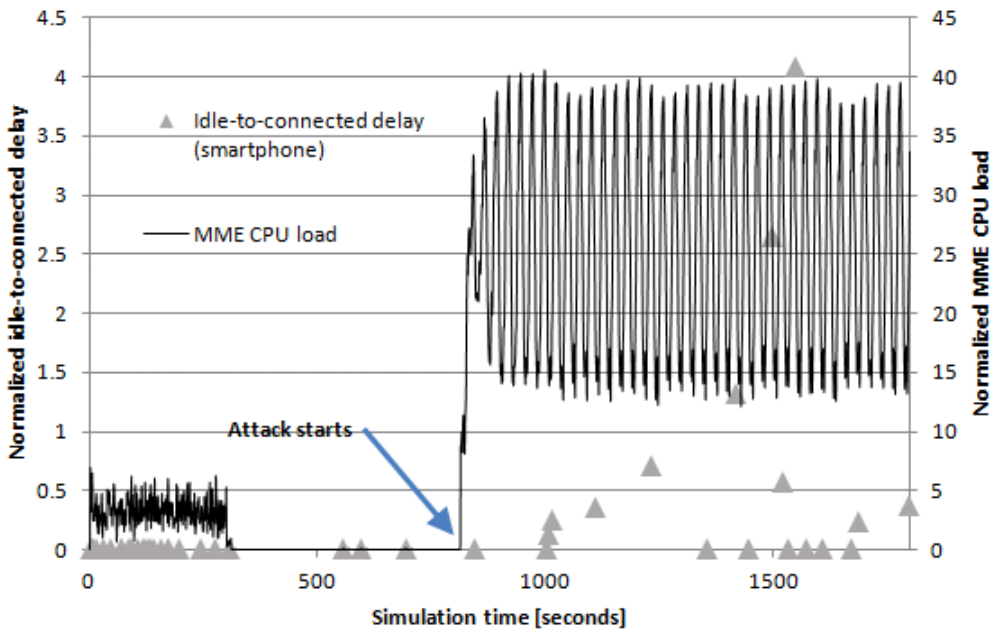


Figure 5.4: Impact of a signaling amplification attack on QoS (RRC state transition delay)

how the attack influences the delay in RRC state transitions of legitimate users. The attack starts at time 800 seconds.

Figure 5.4 plots both the CPU load at the MME as well as the delay legitimate users experience to transition from idle to connected states. One can observe the CPU load at the MME increases substantially for the duration of the attack, staying at a value 15 to 20 times higher than the average value observed at the beginning of the simulation. Note that at the initial stage of the simulation, all the UEs present in this test are connecting and attaching to the network, hence the increase of CPU load at the beginning of the simulation. Once the attack starts, one can observe how the delay to transition between idle and connected states starts to spike. Certain UEs experience a delay up to 1500 times higher than usual. Note that this will significantly impact the QoS witnessed by customers because their mobile devices will experience

a long delay to transition to a connected state. In this situation, the network could potentially appear unresponsive for the affected legitimate devices.

HSS overloading

The second test scenario considered in our experimental study is the case of a botnet attempting to saturate the HSS by means of signaling traffic, namely authentication and attach attempts. Similar threats have been theoretically analyzed in recent years in, for example, [83] and [9].

The attack is implemented by forcing a large group of UEs, members of a botnet, to reset at the same time, thus initiating a NAS Attach procedure at the same time. In a real implementation, the botnet could be instructed to reboot by an external command and control server or it could have a hard-coded instruction to reboot at a given time on a given day. Nonetheless, we assume that it would be very difficult to achieve a totally synchronized reboot of all the bots, both due to variable network delays and latencies and unsynchronized clocks. Therefore, in the current attack implementation, all the bots reboot and initiate a NAS attach procedure within a window of 30 seconds.

Figure 5.5 plots the normalized load at the S6a interface between the MME and the HSS during such an attack for a variable number of malicious devices, or bots, within a botnet. The plot also includes, for reference, the average generated load at this link by 2000 legitimate smartphones operating normally. Intuitively, the larger the size of the botnet, the higher the overload at the link. One can also observe that,

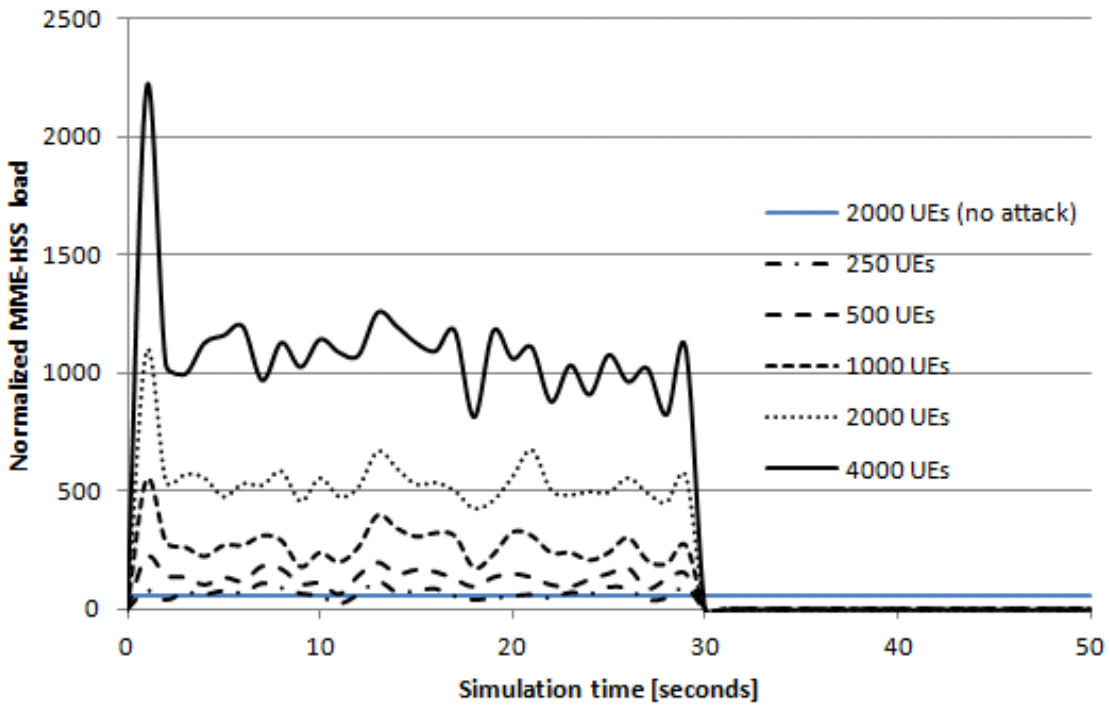


Figure 5.5: Normalized MME-HSS load during an HSS overloading attack

on average, the load generated by the botnet with 2000 UEs is 11 times higher than the case of the same number of devices operating in a legitimate way. The baseline load for 2000 UEs demonstrates that a botnet of only 250 UEs produces 32% more load between the MME and HSS on average than does 8 times that number of UEs exhibiting normal behavior.

Figure 5.5 also confirms that this specific signaling-based attack against the HSS has a limited duration depending on the time that it takes for all the UEs to attach to the network. With a sufficiently large botnet, the HSS overload would result in many attach errors and, therefore, many UEs attempting to attach repeatedly, which would scale up the attack impact and extend its duration. In order to sustain this attack, the botnet could keep rebooting the mobile devices or engage in other signaling-based

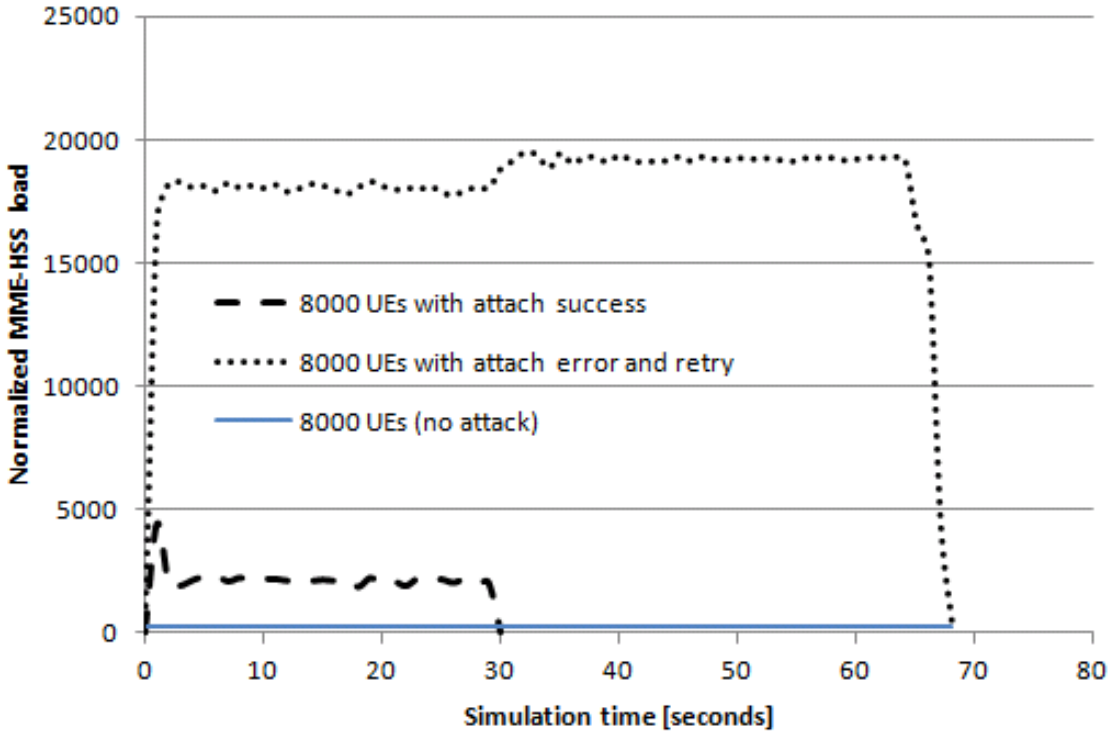


Figure 5.6: Normalized MME-HSS load during an HSS overloading attack with UE attach attempt timeout and retry

floods against the HSS, such as the ones proposed in [83]. Investigating the impact of such a scenario, we configured our test bed in such a way that attach procedures from the UEs can timeout or fail based on the overloading of the HSS. In the event of a failed attach attempt, the UE attempts to attach again. The UE will keep attempting to attach until it is successful. Figure 5.6 plots the normalized load on the S6 interface for this scenario in the case of 8000 UEs initiating the attach procedure within a 30 second time window, both in the case of a single attach attempt and with repeated attach attempts. As observed in the results, the HSS load is 9.47 times higher in the case of UEs re-attempting the attach in the event of a failure or timeout. Moreover, in this case the duration of the congestion sustains for 2.3 times longer.

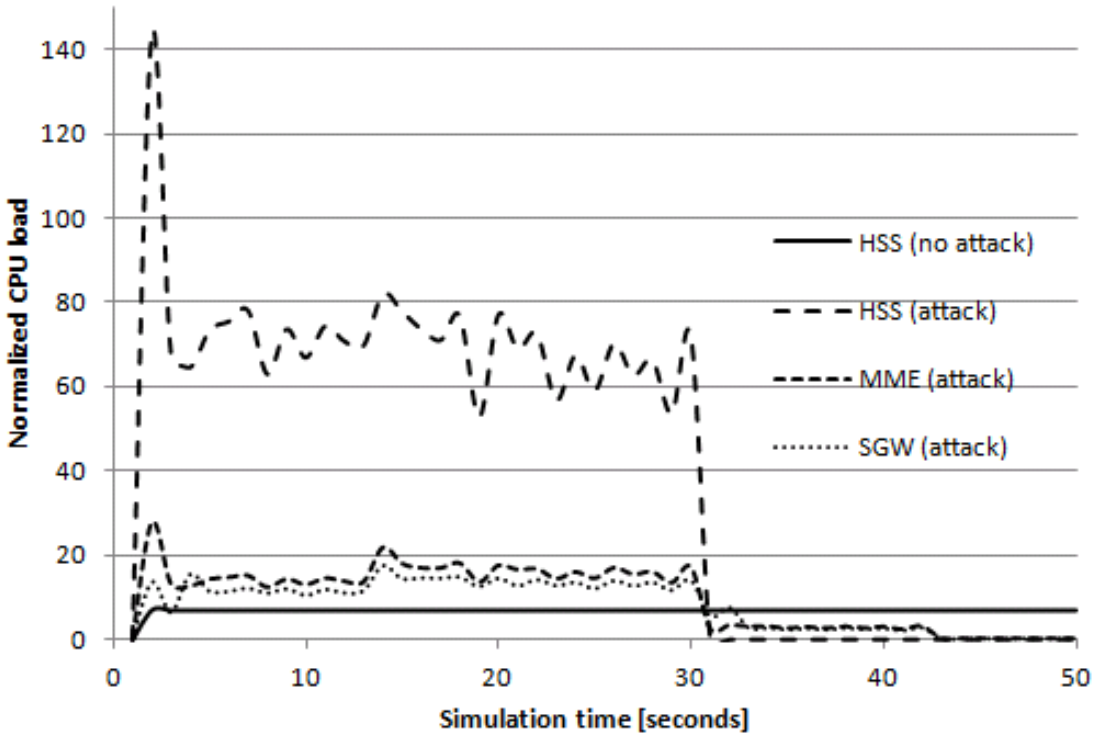


Figure 5.7: Normalized HSS, MME, and SGW CPU utilization for an HSS overloading attack compared to the average HSS CPU load when UEs attach legitimately

As part of our analysis, we also aim to determine the impact of the attack on the EPC and which nodes are affected the most. Figure 5.7 compares the CPU load at each one of the EPC nodes during an HSS overloading attack launched by a botnet of 1000 UEs. These results are compared to the average load of the HSS when the same number of smartphones are attaching to the network in a legitimate fashion (indicated by *HSS (no attack)* in Figure 5.7). As expected, the major impact of this threat occurs at the HSS, which experiences a CPU stress five to six times larger than that of the MME and the SGW due to the high cost of generating authentication vectors and security keys when UEs request to authenticate. When comparing this attack to a legitimate operation, the results indicate that the average CPU load at the HSS is about 10 times larger during an attack.

Note that once the network has been able to process all the attach requests, the attack impact dissipates and the HSS CPU load goes down to zero. Once the UEs have been attached, the HSS is not contacted frequently. The CPU load at the MME and SGW is drastically reduced once the attack is over, but stays at a specific level as the EPC is managing the UE RRC state transitions to idle.

It is important to note that, comparing Figure 5.7 and Figure 5.3, the impact on the MME is much higher for a signaling amplification attack as compared to an HSS overload attack. Specifically, the MME's CPU load increases 6 times more, as compared to processing legitimate traffic, when the attack occurs as a result of frequent RRC state transitions induced by a botnet. In addition, the MME is more severely impacted during a *UE-initiated* signaling amplification attack than is the HSS during an HSS overload attack.

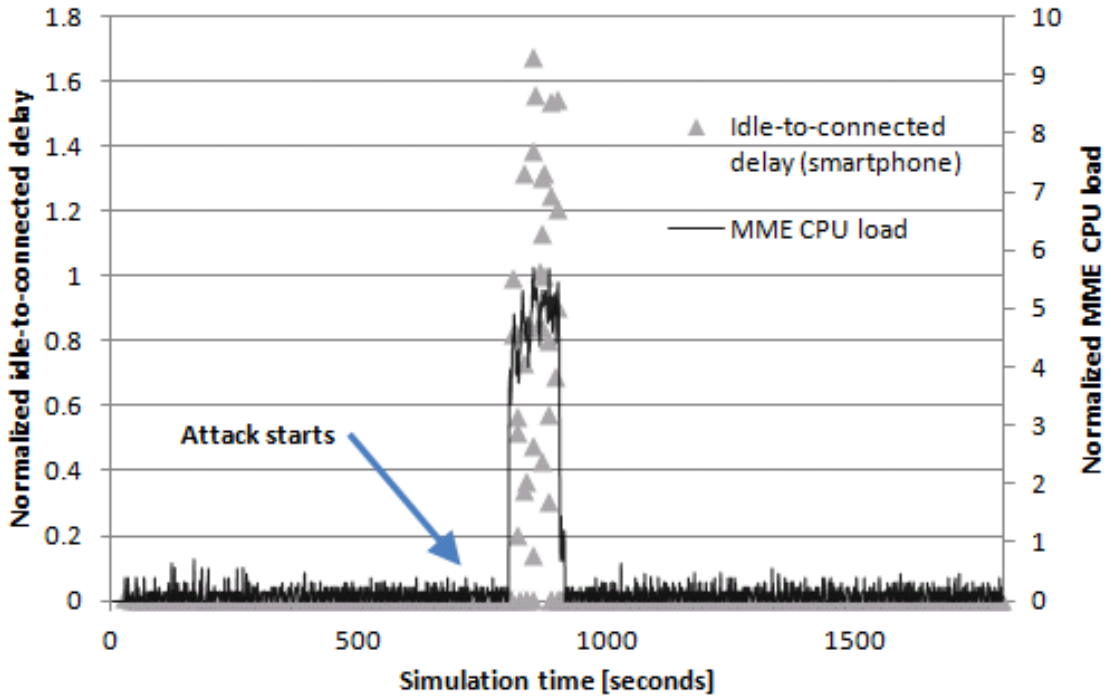


Figure 5.8: Impact of an HSS overloading attack on the QoS (RRC state transition delay)

In order to determine the potential impact of a malicious flood of network attach procedures, we simulate this attack being launched by 1000 UEs. The attack traffic is overlapped with the regular operation of legitimate smartphones. Figure 5.8 plots both the CPU load at the MME and the delay legitimate users experience on their RRC state transitions. The attack begins at time 800 seconds, when 1000 UEs reboot within a time window of 30 seconds. One can observe a clear spike in the time it takes legitimate users to transition from idle to connected state, with this delay increasing up to 711 times.

5.4 Towards a flat and resilient next-generation mobility architecture

In the recent years, the major innovation and technology breakthroughs have occurred at the RAN. The transition to an OFDMA-based modulation results in a substantial increase of the capacity of the wireless channel and, more importantly, it provides a resilient communication medium against frequency distortion and fading. This, combined with advanced MIMO (Multiple-Input Multiple-Output) techniques, is pushing the capacity of the wireless medium to the great throughputs achieved by LTE-Advanced.

This fast-paced innovation at the air interface and RAN side contrasts with the inefficient networking architectures used in cellular core networks. Over three decades' worth of research has resulted in very mature IP-based communication systems with

great efficiency, performance, and reliability. However, the cellular world still carries over a legacy circuit-switched architecture.

In a similar fashion to that of the old Public Switched Telephone Network (PSTN), bearers are constantly being established and released as mobile devices communicate and transition between RRC states [4]. This operation was perhaps efficient in the PSTN when the communicating parties make only a few phone calls per day with average duration of a couple of minutes. However, in the case of cellular networks, constantly switching, establishing, and releasing bearers for devices that connect and disconnect constantly, often for small bursts of traffic, is known to be highly inefficient [65].

LTE standardization aimed for a flat, all-IP architecture, but still relies on a circuit-switched packet core. Meanwhile, technology and innovation at the RAN is moving fast. As an example, earlier this year, the first multi-gigabit wireless chipset was demonstrated [53] and, still in a research stage, a recent world record was set by pushing 100 gigabits per second wirelessly [88]. When one is able to push multiple gigabits per second over the air, a point may be reached where the RAN ceases to be the bottleneck, and this burden shifts to the EPC. In fact, the latency induced by the circuit-switched nature of the EPC is already considered to be a capacity bottleneck [43].

It is also important to note that the traffic traversing the future mobile core network will not only originate at the cellular RAN. Future heterogeneous mobile networks will require seamless connectivity and mobility among different access network technologies. Therefore, the core will be expected to process high throughput

either coming from or destined to other wireless networks, potentially increasing the severity of the EPC bottleneck.

In order to guarantee the performance and security reliability of mobility networks as we transition into future wireless communication systems, the way the packet core is architected should be rethought. Such EPC redesign is further motivated as we move into the connected world, with a drastic increase of connected devices due to the advent of the Internet of Things (IoT) paradigm and Machine-to-Machine systems [56].

In the next section we introduce a potential cellular core architecture redesign that could potentially mitigate and perhaps fully address the threat of signaling overload in next-generation mobile networks. Note that the novelty in this is not to propose a new architecture redesign direction, but to analyze a direction for future mobile network design proposed in the literature (for example in [43]) in the context of security.

Next-generation mobile architecture

Next-generation LTE mobile networks have been designed to satisfy the demands of modern broadband services for a growing number of mobile connected devices. Nevertheless, they lag and carry over the limitations of circuit-switched legacy networks. Such outdated approaches for the cellular core are motivated from early designs of cellular networks, based on circuit-switched protocols for cellular voice. Cellular data communications were originally designed and built as an overlay to the circuit-switched cellular network.

Figure 5.9 describes this legacy circuit-switched architecture as compared to a possible next-generation architecture for mobility networks. In this hypothetical future scenario, a RAN providing high speed and wide-band transmission would be connected to a fully IP-based packet data network. Data communications to and from user devices would then be appropriately and effectively routed through the best and least-cost path to the nearest gateway, for example, to the external destination of the flow. Note that packet-switched technology for IP-based networks is very mature already and proven effective to efficiently route data flows and throughputs much larger than those in mobile networks, even for high QoS-demanding real-time services.

In this ideal architecture, the RAN would independently manage and execute the wireless-related PHY layer and MAC operations, much like in today's LTE networks. However, instead of depending on complex and inefficient signaling exchanges to manage bearers, a given eNodeB will maintain a clear mapping of what network gateways should be reached given an application, source, and destination. With this approach, very similar to regular mature and well-understood IP communication networks, the eNodeB will just have to forward the traffic to the appropriate cellular network edge routing point. Note that this could potentially provide great benefits, including the avoidance of signaling overloads or congestion at specific network points. All in all, cellular networks should not be much different than well-understood WiFi networks, perhaps with a much more complex access point, the eNodeB.

The authors of [43] introduced very interesting directions to achieve such a flexible architecture. They address and discuss how to fit some complex functionalities, such

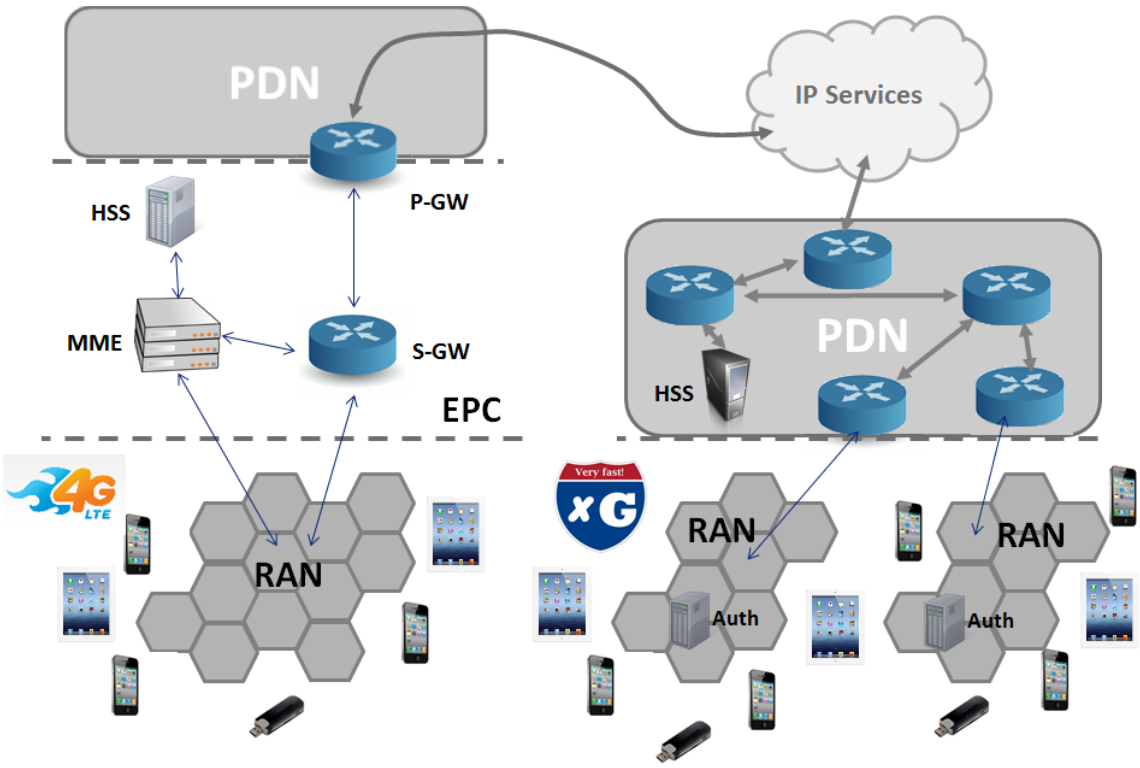


Figure 5.9: LTE circuit-switched legacy architecture vs next-generation full IP architecture

	Total signaling messages	**Total signaling overhead**	**Signaling overhead associated with RRC and NAS procedures**
EPC	29,720	41.89%	23.91%
Global (EPC+RAN)	66,001	61.55%	49.28%

Table 5.1: Total signaling overhead for smartphone traffic compared with overhead associated with only RRC procedures, authentication, and location updates

as roaming and mobility, into this new cellular network paradigm. On top of that, the authentication and other HSS-assisted operations could also potentially be distributed and deployed within the RAN, as depicted in Figure 5.9. In order to provide full mobility and roaming, though, certain HSS-functions should be kept centralized

Attack Type	Normalized average time to reach RRC connected	Normalized connection establishment latency due to bearer setup, modification, and tear down	% of latency due to bearer signaling	% of latency due to bearer signaling with multi-Gbps RAN capacity
HSS Overloading	0.0075	0.0045	59.86%	94.89%
Signaling Amplification	0.0100	0.0047	46.66%	95.36%

Table 5.2: Impact of bearer signaling on smartphone's attach and RRC state transition latency

over the PDN. As a result, it would be essential to protect such resources from potential attacks the same way, for example, important Internet-facing assets from big corporations and banking institutions are protected.

In order to measure the extent of the EPC's circuit-switched inefficiencies, we ran 30 minute simulations of 1000 legitimate smartphone devices with traffic modeled from real smartphones and examined the signaling message overhead as UEs engaged in various signaling procedures. We calculated total signaling overhead (ratio of signaling messages to signaling plus data packets) and signaling overhead associated only with bearer setup, modification, and tear down for the EPC as well as globally across the EPC and the RAN. Table 5.1 is a culmination of the simulation results, which indicate that the elimination of messages due to manipulation of bearers can reduce the total EPC signaling overhead by nearly half and globally by 20%. Therefore, adoption of a purely packet-switched next-generation architecture, such as where the RAN is connected to a fully IP-based packet data network as described above, would

burden the EPC with significantly less signaling overhead.

Table 5.2 further evaluates the consequences of such signaling overhead by examining the amount of latency a UE experiences while transitioning to a connected state. These are the results of two simulations where a botnet of 1000 UEs performs an attack while legitimate UEs send and receive normal traffic. In one simulation, the botnet attempts to attach over a 30 second time window (HSS overloading), and in the second simulation, the botnet produces a signaling amplification attack as it receives periodic traffic from a server as described in Section 5.3. In our test bed's current setup, approximately half of the average latency experienced by the legitimate UEs while transitioning to connected state is a result of bearer setup, modification, and tear down.

To understand how the bottleneck and inefficiency of manipulating bearers is amplified as technology at the RAN advances, we repeated the simulations after increasing the RAN capacity and throughput 1000 times to artificially model a multi-gigabit RAN. For both attack simulations, bearers account for approximately 95% of the average latency experienced by the legitimate UEs to reach connected. These results imply that circuit-switching degrades the performance and efficiency in the EPC, which could be highly improved with the adoption of an architecture redesign. It also provides a clear indication that, if technology follows the current trends, the EPC could eventually become the capacity bottleneck of cellular networks if the inefficient circuit-switched architectures are not replaced.

Finally, we implemented an alternative mobile network architecture, similar to the one depicted in Figure 5.9, on our LTE security test bed. The eNodeB has been

modeled to independently route traffic to the external IP network and appropriately forward data traffic from/to the external server. All the nodes of the EPC have been eliminated as they are not necessary in this test alternative architecture. Note that the design of either an HSS-like entity securely distributed over multiple RAN units or a centralized design of the HSS over the cloud is out of the scope of this work.

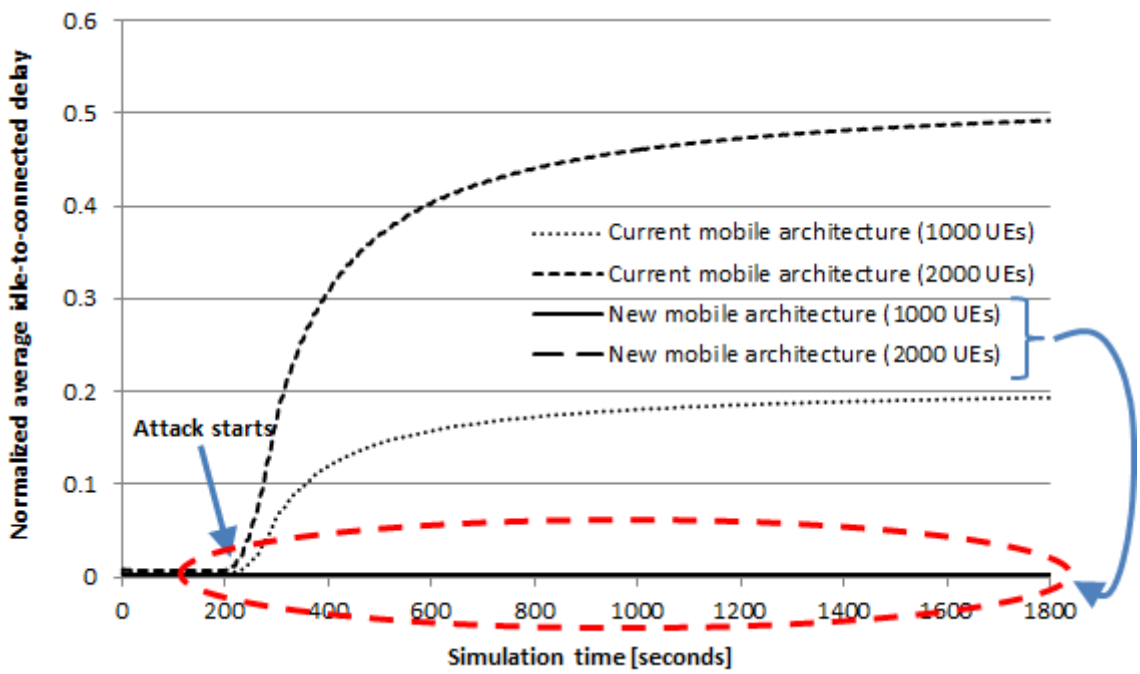


Figure 5.10: Attack resilience comparison: Current circuit-switched vs New packet-switched architecture

This alternative mobile network architecture is run with the periodic traffic used in Section 5.3 to induce a large number of RRC state transitions and simulate a signaling amplification attack. The performance of the network under such traffic load is plot on Figure 5.10, where it is compared to the performance of the current circuit-switched cellular core architecture. The results are generated for a population of 1000 and 2000 UEs and, at time 200 seconds, we initiate a signaling amplification

attack. From the results, one can observe that the performance, in terms of RRC state transition delay, improves significantly, with this delay being 76 times lower in the case of the packet-switched mobile architecture with 1000 UEs and 193 times lower in the case of 2000 UEs.

The transition towards a new secure and robust mobile architecture would not be simple, as it would require deep changes in well-established network architecture, billing systems, and data collection systems. However, such an architecture, drifting away from legacy and inefficient circuit-switched concepts, would result in exceptional benefits in terms of network performance, network cost and maintenance, and, as this chapter highlights, security.

5.5 Conclusion

This chapter presents the first security simulation and analysis of the impact and scalability of signaling overload attacks against an LTE mobile network, aiming to saturate its EPC resources. Due to its reliance on a legacy circuit-switched architecture, the EPC experiences a large signaling overhead that could be maliciously exploited and put the network at risk. We implement and analyze two types of attacks against the EPC via a custom-built, standards-compliant LTE security research test bed. Based on the results herein presented, we identify the most severely burdened points in the network and the potential scalability of the attack's impact when launched from a botnet of mobile devices. Moreover, based on realistic smartphone IP data traffic models derived from real anonymized LTE traffic traces from one of

the main tier-1 operators in the US, we evaluate the attack impact on customer QoS.

Our results indicate that signaling attacks against the EPC can increase the EPC signaling load by up to 295 times and can be optimized, based on the network's specific configuration, to maximize the impact. Based on the security analysis and attack simulations, this chapter discusses the inherent vulnerabilities exploited by these attacks in modern cellular networks. We argue and advocate for the importance of a cellular architecture redesign in order to guarantee full mobility availability against security attacks. The ultimate goal is to diverge from the current inefficient circuit-switched architecture and move toward a fully packet-switched IP-based packet core. Based on simulations, we demonstrate the potential security and performance enhancement of such a redesigned cellular packet core architecture. We present results that provide compelling evidence that strong resilience against signaling overload threats can be achieved through a fully packet-switched core network.

Chapter 6

Autosonda: Discovering Rules and Triggers of Censorship Devices

6.1 Introduction

Censorship devices control the type of content that can be accessed, viewed, or published over a network. Censorship most typically refers to nation-state internet censorship, where countries or regions of countries set rules of the censorship devices to prohibit types of content for citizens within the boundaries of that country's networks. It can also be employed through means such as web proxies, typically by using commercial filtering software, or even enterprise data leak prevention. Although the extent to which a network is censored differs on a country or enterprise basis, most censors act based on characteristics of the network traffic they examine. These characteristics can be protocol-dependent, such as packet header fields, or protocol-independent, such as simply looking for a keyword in a packet's content. Censors use these traffic characteristics to make decisions on how they handle the traffic that they intend to censor, typically by blocking it, modifying its content, or injecting packets into the network stream.

Internet censorship is extremely pervasive in the world today. In 2015 more than 70% of countries employed some type of censorship [27]. Although censorship is so common, many people oppose it. According to an Internet Society survey, 83% of people worldwide believe that access to the internet is considered a basic human right [36]. As a result, censorship evasion techniques have become very popular among citizens in censored countries. The Tor anonymity network [77] and encrypted VPN connections are two examples of tools that have been commonly used for censorship evasion. In 2014 an estimated 400 million people were using VPNs to circumvent censorship or increase privacy [50]. Because such tools are widespread, censorship device rule implementations are frequently updated to adapt to and prevent these anticensorship mechanisms. Once censorship device rules change, anticensorship tools must be updated to evade the censors using new techniques. This creates very frequent reactive behavior between censorship rule creators and engineers of the anticensorship tools.

In order for developers to improve upon their anticensorship tools, they must understand something about how the censorship device makes decisions and how its rules are implemented. However, developers cannot directly access the device rule implementation; from their point of view, the device is just a black box. However, they can reverse engineer rules by specially crafting traffic and probing the censor to examine its output. Such approaches have been applied to specific censors, such as the Great Firewall of China [87], however the large majority of censorship techniques and how often they change remain unknown. Existing methods to date for discovering censorship techniques are mostly manual, which are time consuming, not scalable, and

are not feasible in the long run, given the dynamic nature of censor implementations. Censor devices could be activated on a range of behavior, but finding the exact trigger is often too arduous a task to be done manually at scale. Yet, such knowledge is necessary for properly analyzing censorship devices, comparing their models, and designing evasion techniques. Although there currently exist tools for discovering *when* something is censored, we do not have tools to tell us *how* the censorship is done and what are the details and implementation of the rule enforcement.

This analysis can be used both to find evasion techniques and also for strengthening systems against evasion. Opinions and ethical considerations on censorship are out of the scope of this thesis, however there remain valid use cases both for strengthening and breaking these network control devices. In order to circumvent a censor, its behavior needs to be deeply understood, and the same could be said to strengthen a device's rule implementation. The analysis in this work assists with both.

In this chapter we present Autosonda, a tool used for automated rule reverse engineering and fingerprinting of censorship middleboxes. Our solution uncovers the model and mechanism used by a censor for making decisions on how to handle traffic, as well as identifies the censor's approach used to block content that is deemed prohibited. To identify feature triggers, our tool runs a series of protocol-aware probe tests at each layer and then narrows down its search space to uncover how a particular censor rule is implemented. Our tests are based on those used in previous manual analysis of censors to understand their decision making process. The goal of Autosonda is to capture that knowledge into a tool that performs the analysis automatically and at scale. It can be used not only to create implementation finger-

prints of censors, but to also track how the fingerprints change over time and compare them with fingerprints of other devices. The following sections of this chapter will discuss the architecture and implementation of our system and provide results from our experiments on web filters.

6.2 Use Cases

In addition to discovering censorship decision models, Autosonda can be used for a variety of other purposes. Network management policies are difficult to write because of so many protocol ambiguities. Software developers, no matter how skilled, will often create holes in implemented rules and regular expressions, unintentionally weakening the intended policies. Yet, even if someone writes error-free code, there are still multiple ways to express a rule and multiple implementations. Because there are so many ambiguities and cases to cover, manual inspection and testing of the code is not feasible. Autosonda can be used to test network device policies for this purpose, to find bugs and paths of rule circumvention. Similarly, because these mistakes are so common, software updates and patches are very frequently applied. Autosonda can also be used to discover when and how often these updates are applied.

Enterprises often outsource building of tools for their network management. As customers, they have control over which rules may be enforced, but not necessarily the implementation of those rules. In this case, a grey-box scenario, enterprises could use Autosonda to ensure that implementations are robust. This type of assurance is crucial, considering that enterprises are often liable for any data entering or exiting

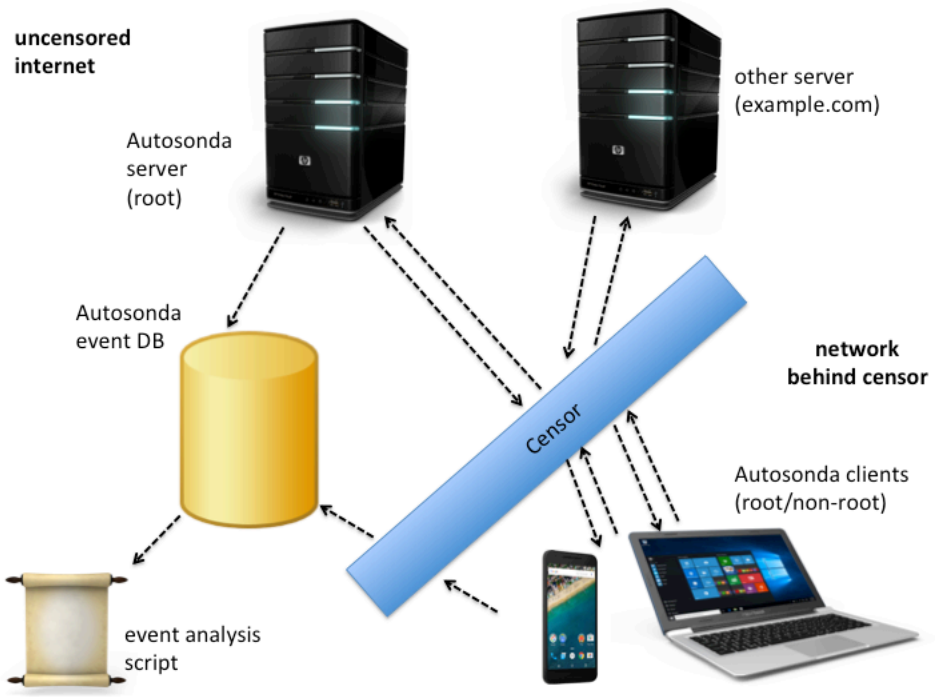


Figure 6.1: Autosonda architecture

on their networks, so there is little tolerance for rule evasion. Subtle variations in an implemented rule from sloppy programming can completely change a security policy and lead to exploitable vulnerabilities.

6.3 Tool design and implementation

Figure 6.1 shows the architecture of Autosonda. It consists of a client device located within a censored country or behind a filtering agent and a set of custom servers running on Amazon EC2 [1] outside of the censored region. The client and server execute a series of tests that craft traffic for the purpose of bidirectional probing the censorship device to discover various attributes about its decisions for traffic handling. The client and the server both log events that are stored either locally or

on a database outside of the censored region. Once the tests are completed, we do postanalysis of the event logs to determine the results by comparing the actual events with the expected events.

The goal of Autosonda is to create an implementation fingerprint of a censorship device. This involves discovery in three different categories: *model, mechanism*, and *technique*. Model refers to discovery of network traffic features that the censorship device keys in on. Some examples are IP destination address and the Host header field of an HTTP GET request. Autosonda examines features at each layer of the network stack to determine which trigger a censor. Because there are an exponential number of values to test, it is not feasible to discover the entire censor model. However, it is also not necessary to discover the entire model in order to create an implementation fingerprint of a censor. For example, a particular rule set might include a rule such as "block traffic if byte 8 is 0." It could potentially take an exponential number of tries to find such a rule, yet knowledge of it doesn't necessarily impact a significant amount of client traffic. Instead, we aim to discover a subset of the censor's model according to which traffic features have been identified in existing literature as the most impactful. For each of these features, we take protocol layers and semantics into account. For example, if we speculate that TCP port is a feature of interest, we could test the censor by sending packets with different port numbers to see if and how its behavior changes. Another important aspect of a censor's model is its maintenance of state. Autosonda runs a series of tests to determine if the censor maintains state at all and, if so, at which network layer. For each state that is maintained in a censor, there is a point at which that state expires. Autosonda runs additional tests

to determine the timeout period of a state.

For each feature that triggers a censor, there is a certain mechanism that is used to look for that feature in the network traffic. There could be a particular regular expression implemented in one of the censor's rules that tries to match on a field in a packet header. Autosonda tries to uncover the mechanisms used for various features in a censor's model by utilizing a fuzzing-like approach. Fuzzing is a software testing technique that provides different types of input to a program to test how it responds. It typically uses unexpected or random data as input, while Autosonda uses a more protocol-aware approach for crafting its input data. An example of Autosonda's fuzzing technique is shown in Figure 6.2. This example assumes that the censor is keying in on the 'GET' keyword; perhaps it wants to censor every HTTP GET request that it receives. Once Autosonda knows that the 'GET' keyword is a feature of interest, it runs a series of tests to determine the censor's rule for that keyword. In the example, we start with a typical 'GET' and then try to change the capitalization to 'GeT' and then the spacing between 'GET' and the forward slash. Discovering the censor's mechanism is particularly important because of protocol ambiguity. There are often nuances in syntax that are not perfectly specified in protocol specification, leading to ambiguity. Such ambiguities are difficult for middlebox developers to handle when accepting and parsing input and often lead to poorly implemented rules.

Finally, technique refers to the action taken by the censor to prohibit censored content. In the case of the Great Firewall, this could be sending a TCP RST packet to the client that requests forbidden content. It could also be modification of a packet's content or dropping of certain packets once they reach the censor.

```
GET / HTTP/1.1\r\nHost: www.a.com\r\n\r\n
GeT / HTTP/1.1\r\nHost: www.a.com\r\n\r\n
GET/ HTTP/1.1\r\nHost: www.a.com\r\n\r\n
```

Figure 6.2: Example of a fuzzing approach to determine the rule implemented to match GET

Assumptions

When creating our set of tests, we made several initial assumptions. The first is that there exists some form of censorship on the network and that we have a given string that triggers a censorship event, most typically a censored URL. We get this information by testing URLs that have a high probability of being censored and only include networks that show signs of censorship. We further assume root access on the control servers for our experiments. However, Autosonda can run two sets of tests on the client: one with root access and one without. Therefore, even if it is not possible to have root on the client device, there is still a substantial amount of censorship device discovery that can be done. These tests could be useful, for example, in a scenario where the client is an unrooted smartphone.

Test Sets

Because we focus on web filtering and internet censorship, Autosonda primarily tests TCP, UDP, HTTP and DNS protocols. However, functionality can be extended by simply adding additional tests for other protocols. As mentioned above, Autosonda starts with a censored URL and discovers the censor's model by executing a series of tests to find features that are of interest to the censor. When filtering occurs for a

particular domain, the censor typically identifies that domain by URL or IP address. Autosonda first tries to determine which, or both, of these features the censor looks at. Other types of filtering, such as with data leak protection, as well as types of censorship could also use keywords in data content as triggers. Although we didn't have the opportunity to test Autosonda on these types of filters, its approach could easily be extended to do so.

6.4 Experiments and Results

Experimental Setup and Censor Modeling

To demonstrate Autosonda's utility and use it to discover models of web filtering devices, we performed experiments on 76 censored wifi networks over several months in 2017 in the New York City metropolitan area. The wifi networks were all open (not password protected) and located in establishments such as banks, community institutions, clothing stores, grocery stores, home furnishing stores, restaurants/fast food chains, and medical clinics, to name a few. For the purpose of our experiments, we labeled the networks as censored if we were not able to retrieve content from the number one most popular Adult category site in Alexa's top 500 sites by category [13]. The experiments involved connecting a client mobile device to a wifi network that enforces censorship via web filtering. The client then ran Autosonda to probe the web filter with traffic and gather data about its filtering mechanism. There were three test servers, located on EC2 in the United States, that our clients corresponded

with over the series of tests. The client and server tests were implemented in Java and Python, and we used Scapy to specially craft network traffic. Autosonda's tests are designed to discover the model, mechanism, and technique used by a censor only by examining network traffic from both the client and server. There was no physical access or remote direct control of any of these devices or communication with network system administrators during our experiments. We specifically defined the model, mechanism, and technique of the web filters as follows.

The model of a web filter is characterized by the feature that triggers the censorship of a URL. For these experiments we focused on two types of triggers: URL and IP address. We discovered that all of the web filters we tested censored by maintaining a blacklist of one of these two characteristics. As discussed in Section 6.3, mechanism refers to implementation details for how the web filter performs its censorship. For our experiments, we considered the following characteristics for identifying mechanism:

1. How protocol-specific is the implementation of the censor
    a) Does it censor only for port 80
    b) Does it censor only TCP
    c) Does it censor multiple Host headers
    d) Is the censor triggered on keywords in the URI
    e) Does it censor resent requests and responses
2. How does the censor handle protocol ambiguities
    a) How does it respond to HTTP GET and DNS fuzzing

Device category	Number of devices	Web filter vendor
DNS filtering	21	OpenDNS, Skydns, Savvis, Amazon, Fortinet, Conversant, Norton Connectsafe
Host header filtering	44	SonicWall, AT&T, Juniper, Fortinet, Cisco Meraki, ZScaler, Global Technology Associates, BHI, 12 unknown
Host header lookup	11	OpenDNS, Cisco Scansafe, Squid Proxy, Wayport, 1 unknown

Table 6.1: Web filter vendors encountered for each category of device

3. Does the censor maintain state and what is the state expiry time

    a) TCP, HTTP

4. Where is the censor's blacklist logically located

    a) Outbound HTTP request, inbound DNS response

5. Does the censor reassemble IP fragments and TCP segments

The techniques of web filters are the means by which the filters perform their censoring. The types of techniques that we observed in the experiments were modification of HTTP responses and modification of DNS responses. In addition to identifying the model, mechanism, and technique for our 76 censored networks, we also observed the vendor of each web filter when possible (through identifying features of network traffic). We also took note of the percentage of censors whose filtering rules we were able to bypass, which for these experiments was 100%. Overall we found that the approaches taken by web filters are not robust and are easily breakable with a slight change in protocol attributes or using a different protocol to transmit data.

Table 6.1 shows the web filter vendors that we encountered for each category of device (see below). We were most often able to infer the vendors by traffic that we

received, such as in response messages, but there were several filters that did not give us clues about vendor information. Those are labeled as unknown. About half of the web filter vendors we observed in the DNS filtering category were OpenDNS. Norton ConnectSafe and Amazon were about 15% each. 20% of the devices in the Host header filtering category were Fortinet and 30% were Cisco Meraki.

Results

Model and Technique: We divided our 76 web filters by the primary traffic charateristic with which they were triggered and their approach to triggering. Autosonda was able to break down the filters into two main categories and one subcategory: DNS filtering, Host header filtering, and Host header lookup. 21 of the filters (27.63%) maintained a DNS blacklist and performed all of their censorship via DNS (DNS filtering category). Each of these filters monitored DNS responses when clients made an HTTP request and compared the result against a blacklist of IPs. If the returned IP matched one in the blacklist, the filter overwrote the DNS response to redirect the HTTP request to a static block page. The block page is typically maintained by the vendor of the web filter. This category of filters only censored requests that contained the URL of a censored webpage. They did not censor requests going to our servers on EC2, which contained Host header values of censored URLs. 44 of the 76 web filters (57.89%) censored based on the HTTP Host header of a GET request (Host header filtering category). These filters checked the GET Host header against a blacklist of URLs to determine if the response should be blocked. The remaining 11 filters

(14.47%) could be classified as a subcategory of the DNS blacklist category. They censored by ignoring the destination URL of an HTTP request and instead doing a DNS lookup of the Host header value in the HTTP request (Host header lookup category). Once the filter receives the DNS response, it searches for the returned IP in a blacklist of IP addresses. If found, it creates a new response from the IP of the Host header URL, writes content describing that the page is blocked, and sends this message to the client. Source IP addresses of responses coming from censored URLs or our EC2 servers were left unmodified by filters in the Host header filtering category. However, since requests were redirected to static pages with the DNS filtering category, responses received by our clients were from the source IP address of the static pages for this category.

Mechanism: To discover details about the implementation of our web filters, we ran the tests described above (Section 6.4). Our tests for understanding how protocol-specific the filter's implementation were involved crafting HTTP requests with slight variations in HTTP attributes. Web filters likely assume that all HTTP requests will be using TCP on port 80. When we sent exactly the same HTTP request using UDP, all of our filters allowed the request and response without any censorship. Thus, the filters were only examining TCP communications. Although HTTP requests and responses are only expected to be on port 80, it is still useful to know that the filters are only looking at TCP traffic. However, when we sent requests using TCP on port 9900, a port not typically used with HTTP, our results differed quite a bit. Most filters did not care that the requests were not on port 80; they censored them anyway. Others, 17 out of the 49 filters that censored requests to our servers, only inspected

requests going to port 80 and allowed those going to 9900 without censoring them. None of our filters expected keywords in the URI field of the HTTP GET request and they all allowed responses from censored URLs if the request included a keyword.

RFC 7230 [25] states that HTTP requests with multiple Host headers should be rejected with a 400 response. To test if the filters were properly implementing HTTP, we ran various tests in which the client device sent an HTTP request with multiple Host headers and changed the ordering of the hosts. These tests yielded very interesting results. For our 55 filters that examined Host header, 26 only looked at the first Host header, 27 only looked at the last Host header, and two looked at both. Such poor implementations can lead to severe security problems, clearly demonstrated in our experiments with bypassing security policies, and also with HTTP cache poisoning [18].

We saw a lot of variety among web filter vendors for how they handle TCP segmentation and IP fragmentation reassembly. Of our 44 filters that were not DNS censored, eleven did not reassemble TCP segments and seven different filters did not reassemble IP fragments. Five filters had short timeout periods for fragments and segments, just below two seconds, even though the timeout period of HTTP request state was more than 8.5 seconds for all of the filters. For practicality purposes, we did not test state expiry for more than 8.5 seconds.

Since filters in the DNS filtering category censor right at DNS lookups, the mechanism tests for HTTP are not relevant to devices in this category. However, we did perform some additional tests for these devices to understand more about how the filters are implemented. The first test was a DNS fuzzing approach where we made

Tables 6.2-13: Fuzzed HTTP GET request tests and number of web filters that were bypassed out of the 44 that examined HTTP request strings (the first %s is replaced by a censored URL; the second, when present, is replaced with a test id)

HTTP GET request	**Number of web filters bypassed**
GET / HTTP/1.1\r\n Host: xxxxxxxxxxxx%sxxxxxxxxxxxxxxxxxx\r\nX-id:%s\r\n\r\n	34
GET / HTTP/1.1\r\nHost:xxxxxxxxxx%sxxxxxxxxxxxxxxxxxxx\r\nX-id:%s\r\n\r\n	33
GET / HTTP/1.1\r\nHost:xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx\r\nX-id:%s\r\n\r\n	35

Table 6.2: long hostname tests

HTTP GET request	**Number of web filters bypassed**
GeT / HTTP/1.1\r\nHost: %s\r\nX-id:%s\r\n\r\n	16
/ HTTP/1.1\r\nHost: %s\r\nX-id:%s\r\n\r\n	28
/ HTTP/1.1\r\nHost: %s\r\nX-id:%s\r\n\r\n	29
get / HTTP/1.1\r\nHost: %s\r\nX-id:%s\r\n\r\n	20
XXX / HTTP/1.1\r\nHost: %s\r\nX-id:%s\r\n\r\n	24
GE / HTTP/1.1\r\nHost: %s\r\nX-id:%s\r\n\r\n	24

Table 6.3: GET word tests

DNS lookups for a censored URL and changed capitalization and domain extensions of the URL. Although capitalization didn't influence any of the filters, changing of domain extension, for example from .com to .org, did manage to return the correct IP address of a URL that should have been filtered for three of the filters. As with any of these filtering approaches, the success of the approach relies on the robustness of the blacklist, which is difficult and time-consuming to maintain. Another important attribute that we wanted to discover with the DNS filtering category is how it handles custom DNS responses messages. For these tests we created our own DNS server and forwarded requests for censored URLs to the server from our clients. Fourteen of the web filters modified our DNS responses, while six did not.

The last group of Autosonda's tests take a fuzzing approach to HTTP GET requests in order to test to strength of filters' regular expression matching. Using Scapy, we created request messages with slight modifications to see how the web fil-

HTTP GET request	Number of web filters bypassed
GET / HTTP/\r\nHost: %s\r\nX-id:%s\r\n\r\n	21
GET / http/1.1\r\nHost: %s\r\nX-id:%s\r\n\r\n	18
GET / HTTP/1.\r\nHost: %s\r\nX-id:%s\r\n\r\n	20
GET / HTT/1.1\r\nHost: %s\r\nX-id:%s\r\n\r\n	22
GET / /1.1\r\nHost: %s\r\nX-id:%s\r\n\r\n	21
GET / /1\r\nHost: %s\r\nX-id:%s\r\n\r\n	22
GET / HTTP/ \r\nHost: %s\r\nX-id:%s\r\n\r\n	19
GET / /\r\nHost: %s\r\nX-id:%s\r\n\r\n	20
GET / HtTP/1.1\r\nHost: %s\r\nX-id:%s\r\n\r\n	6
GET / /11.1\r\nHost: %s\r\nX-id:%s\r\n\r\n	21
GET / XXXX/1.1\r\nHost: %s\r\nX-id:%s\r\n\r\n	22
GET / HTTP9\r\nHost: %s\r\nX-id:%s\r\n\r\n	21
GET / HTTP\r\nHost: %s\r\nX-id:%s\r\n\r\n	20
GET / \r\nHost: %s\r\nX-id:%s\r\n\r\n	20
GET / HTTP/9\r\nHost: %s\r\nX-id:%s\r\n\r\n	20

Table 6.4: HTTP word tests

HTTP GET request	Number of web filters bypassed
GET / HTTP/1.1\r\n%s\r\nX-id:%s\r\n\r\n	38
GET / HTTP/1.1\r\nHost%s\r\nX-id:%s\r\n\r\n	36
GET / HTTP/1.1\r\n Host: %s\r\nX-id:%s\r\n\r\n	23
GET / HTTP/1.1\r\nXXX: %s\r\nX-id:%s\r\n\r\n	29
GET / HTTP/1.1\r\nH: %s\r\nX-id:%s\r\n\r\n	38
GET / HTTP/1.1\r\nHostwww.%s\r\nX-id:%s\r\n\r\n	37
GET / HTTP/1.1\r\nHoSt: %s\r\nX-id:%s\r\n\r\n	20
GET / HTTP/1.1\r\n %s\r\nX-id:%s\r\n\r\n	38
GET / HTTP/1.1\r\nXXXX: %s\r\nX-id:%s\r\n\r\n	36
GET / HTTP/1.1\r\n: %s\r\nX-id:%s\r\n\r\n	37
GET / HTTP/1.1\r\nHost www.%s\r\nX-id:%s\r\n\r\n	37
GET / HTTP/1.1\r\nHost %s\r\nX-id:%s\r\n\r\n	37

Table 6.5: Host word tests

HTTP GET request	Number of web filters bypassed
GET / HTTPx/1.1\r\nHost: %s\r\nX-id:%s\r\n\r\n	21
GET / HTTP /1.1\r\nHost: %s\r\nX-id:%s\r\n\r\n	21
GET / HTTP/ 1.1\r\nHost: %s\r\nX-id:%s\r\n\r\n	28
GET / HTTP/1.1x\r\nHost: %s\r\nX-id:%s\r\n\r\n	18
GET / HTTP/x1.1\r\nHost: %s\r\nX-id:%s\r\n\r\n	19
GET / HTTP/1.1 \r\nHost: %s\r\nX-id:%s\r\n\r\n	16

Table 6.6: spacing after HTTP tests

ters responded. Table 6.2 - 6.12 show a list of the 76 tests that we performed in this group, along with the number of filters that were bypassed with each request. With just simple modifications, many of the web filters allowed censored content to bypass their policies. Among all the filters we see a great deal of variation in behavior. These

HTTP GET request	Number of web filters bypassed
GET / HTTP/1.1\r\nHost: %s\r\nX-id:%s\r\nAccept: text/html, */*;q=0.8\r\n\r\n	0
GET / HTTP/1.1\r\nHost: %s\r\nX-id:%s\r\nAccept:text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8\r\n\r\n	0
GET / HTTP/1.1\r\nHost: %s\r\nX-id:%s\r\nAccept: text/html\r\n\r\n	0
GET / HTTP/1.1\r\nHost: %s\r\nX-id:%s\r\nAccept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8\r\n\r\n	0
GET / HTTP/1.1\r\nHost: %s\r\nX-id:%s\r\nAccept:text/html\r\n\r\n	0
GET / HTTP/1.1\r\nHost: %s\r\nX-id:%s\r\nAccept:text/html,*/*;q=0.8\r\n\r\n	0

Table 6.7: Accept tests

HTTP GET request	Number of web filters bypassed
GET / HTTP/1.1\rHost: %s\r\nX-id:%s\r\n\r\n	21
GET / HTTP/1.1\nHost: %s\r\nX-id:%s\r\n\r\n	2
GET / HTTP/1.1 Host: %s\r\nX-id:%s\r\n\r\n	36
GET / HTTP/1.1Host: %s\r\nX-id:%s\r\n\r\n	34

Table 6.8: \r\n tests

HTTP GET request	Number of web filters bypassed
GET / HTTP/1.1\r\nHost: [host without extension]\r\n\r\n	23
GET / HTTP/1.1\r\nHost: %s#\r\nX-id:%s\r\n\r\n	20
GET / HTTP/1.1\r\nHost:%s\r\nX-id:%s\r\n\r\n	4
GET / HTTP/1.1\r\nHost:www.%s\r\nX-id:%s\r\n\r\n	6
GET / HTTP/1.1\r\nHost:x%s\r\nX-id:%s\r\n\r\n	20
GET / HTTP/1.1\r\nHost: www.%s\r\nX-id:%s\r\n\r\n	1
GET / HTTP/1.1\r\nHost: %s\r\nX-id:%s\r\n\r\n	1
GET / HTTP/1.1\r\nHost: www.%s\r\nX-id:%s\r\n\r\n	0
GET / HTTP/1.1\r\nHost: %s\r\nX-id:%s\r\n\r\n	0

Table 6.9: hostname tests

HTTP GET request	Number of web filters bypassed
GET / HTTP/1.1\r\nHost:.com\r\nX-id:%s\r\n\r\n	36
GET / HTTP/1.1\r\nHost: \r\nX-id:%s\r\n\r\n	29
GET / HTTP/1.1\r\nHost:a.com\r\nX-id:%s\r\n\r\n	35
GET / HTTP/1.1\r\nHost:\r\nX-id:%s\r\n\r\n	30

Table 6.10: after Host tests

results really demonstrate the utility of Autosonda, that it is able to discern slight variations and find trends in censor behavior by trying many tests that modify subtle details of traffic. These results give us many clues about how filters implement their regular expressions. Note that actual retrieval of prohibited content depends on the implementation of the server. A server can intentionally be liberal in the formatting

HTTP GET request	Number of web filters bypassed
GET/ HTTP/1.1\r\nHost: %s\r\nX-id:%s\r\n\r\n	26
GET z HTTP/1.1\r\nHost: %s\r\nX-id:%s\r\n\r\n	24
GET ? HTTP/1.1\r\nHost: %s\r\nX-id:%s\r\n\r\n	21
GET HTTP/1.1\r\nHost: %s\r\nX-id:%s\r\n\r\n	26
GET /HTTP/1.1\r\nHost: %s\r\nX-id:%s\r\n\r\n	12
GETHTTP/1.1\r\nHost: %s\r\nX-id:%s\r\n\r\n	28
GET/HTTP/1.1\r\nHost: %s\r\nX-id:%s\r\n\r\n	27
GET**HTTP/1.1\r\nHost: %s\r\nX-id:%s\r\n\r\n	25
GET /xHTTP/1.1\r\nHost: %s\r\nX-id:%s\r\n\r\n	13
GET HTTP/1.1\r\nHost: %s\r\nX-id:%s\r\n\r\n	22
GETx/ HTTP/1.1\r\nHost: %s\r\nX-id:%s\r\n\r\n	29

Table 6.11: Request-URI tests

HTTP GET request	Number of web filters bypassed
GET / HTTP/1.1\r\nHost:[host+host]\r\n\r\n	20
GET / HTTP/1.1\r\nHost: [host+host]\r\n\r\n	22

Table 6.12: Host substring tests

of HTTP requests that it accepts or it can also unintentionally contain bugs in its rules for parsing requests. Regardless, our goals were to test the implementation of the filter rather than the server, so our control servers returned an HTTP 200 response and content for any request it received.

The categories that allowed the most bypasses were long hostname tests, Host word tests, \r\n tests, and after Host tests. The results for long hostname tests give us intuition that most filters do not search entire hostnames for censored URL substrings, however the Host substring tests yield mostly censored results. A likely explanation is that filters are looking at the first or last part of the hostname to search for a censored URL. When the URL occurs in the middle of a long string, such as in the long hostname tests, the filters usually do not find it.

Most filters hardcode the Host word in their regular expressions, according to the results in Table 6.5. It also seems that their regular expressions are rather particular,

since we see a majority of filters bypassed with strings that change the spacing and characters in and around the Host word. However, changes in capitalization seem to be caught by most of the filters, which we can see when we tried "HoSt". Filters' regular expressions just before Host are also particular. We can see in Table 6.8 that nearly all of the filters were bypassed by removing the \r\n and spacing before Host. Filters are likely using these tokens as delimiters to split sections of the GET request. Unsurprisingly, filters were easily bypassed in our after Host tests, Table 6.10. In these request strings we placed the censored URL in the X-id field and then modified tokens just after Host. Clearly most of the filters look only at the Host field for a censored URL and ignore everything afterward. Thus, with special implementation of a server, a client could craft a request by putting a censored URL after the Host field and bypass the policies of the filter.

Web filter vendors: A very interesting result that we noticed is the diversity of behavior among web filters of the same vendor. For example, we observed 13 Cisco Meraki web filters and found that only two of them considered both Host fields when multiple Hosts were used in a request, whereas the other 11 filters only looked at the last Host. Similarly, two different Meraki filters used a timeout period of less than two seconds for IP fragments, where the other 11 filters had a timeout of over 8.5 seconds. A logical explanation for this behavior diversity is that some of the filters were running more updated software. For differences that are not due to poorly implemented software, customer preferences could also be a factor in behavior diversity. Not only did we see subtle variations in implementation among the same vendors, but we also noticed completely different approaches taken to filtering by the

same vendors. Fortinet filters, for example, accounted for eight of our Host header filtering category devices. These devices solely considered an HTTP request Host header when making filtering decisions. Yet, we found two filters also by Fortinet that performed only DNS filtering and completely ignored the HTTP Host field.

Bypassing filtering mechanisms: Through our experiments we were able to bypass 100% of the web filters we tested. Although all of the approaches that we took for bypassing were protocol-related, it is also worth mentioning that since all of the web filters we studied were blacklist-based and did not do content filtering, data could easily be transferred through any URL/IP not on the blacklist. Although Autosonda's tests did not account for content filtering, its fuzzing approach could easily be extended to handle devices that implement this type of censorship.

Since all of the filters in our DNS filtering category worked by modifying DNS responses, they were easily bypassed by sending HTTP GET requests directly to an IP address rather than performing a DNS lookup of a URL. Similarly, HTTP requests that we made to our servers on EC2 with a censored Host header also successfully returned responses without modification for the DNS filtering category. As mentioned above, sending HTTP requests over UDP bypassed filters 100% of the time, as did adding keywords to URI field of requests and using multiple host headers. Our HTTP request fuzzing approach also bypassed filters very well for the Host header filtering category. For 76 different tests, we saw that individual filters failed to censor up to 65 of those tests for SonicWall, 60 tests for Cisco Meraki, 52 for ZScaler, 52 for Juniper, 38 for AT&T, and up to 51 for unknown vendors.

The web filters on which we ran our experiments were likely programmed to block

access to specific categories of websites, which is why they keep a blacklist of URLs or IP addresses to block. A blacklist approach is difficult to maintain, since websites are constantly changing and content can be moved around to different sites, easily allowing one to bypass the filter. To even attempt to keep blacklists up-to-date, they need to be frequently pushed updates. During our experiments we ran some additional tests to see how robust the blacklists were for our test filters. To do this, we downloaded the Alexa top 100 Adult category sites on the web and tried to connect to them through each filter that blocked the number one most popular site. Not one of the filters we tested blocked access to all 100 of the sites. We even saw some filters with as low as 31 blocked sites. Also interesting to see was that different filters of the same vendor blocked different subsets of sites.

Establishment type categorization: We didn't find any correlation between type of establishment and its censorship mechanisms. However, we were surprised to see that there was also very little correlation between type of establishment and whether its networks were filtered at all. Many community centers, such as libraries, where we expected to find censorship, did not employ web filters at all. We also noticed that although retail chains typically use the same type of filtering mechanisms among all their branches, it is not always consistent. We observed several institutions that filtered in one branch but not another, as well as completely different filtering vendors between branches. Finally, although we ran our experiments over a short period of time, we noticed a change at one national chain from an uncensored to a censored network in three different locations.

6.5 Conclusion

Discovering decision models of censorship devices and network management devices is useful to find evasion techniques and strengthen implementations. However, it is difficult to perform this type of analysis in depth and at scale. We introduce Autosonda, a tool used for automated rule reverse engineering and fingerprinting of censorship middleboxes. Autosonda is used to uncover the model, mechanism, and technique used by a censor when access to the device is only available through network traffic probing. Through a series of tests across multiple protocols, Autosonda characterizes devices according to their decision models, techniques for enforcing censorship, and discovers clues about the regular expressions used by these devices for rule enforcement. The value and effectiveness of our tool is demonstrated by using it in a study of 76 web filters, where we discover a variety of implementation and decision-making techniques. Autosonda enabled us to find methods for bypassing 100% of the filters we studied as well as categorize common implementation flaws and rule sets for popular device vendors.

Chapter 7

Conclusion

Networks are constantly facing changing requirements due to continuous surges in the number of connected devices. As technology continues to flourish, new types of devices will continue to enter the market, increasing the demand for high speed and secure network communication. Along with a rise in the number of connected devices is also an increase in the instances and sophistication of attacks. The scale and heterogeneity of devices and networks makes it very difficult to provide proper security to defend against these threats, both for the end nodes as well as the elements of the network itself. Network control devices enforce policies and security mechanisms on networks, and they now need to protect against increasingly sophisticated attacks and understand the impact of threats. However, automated tools are necessary to detect, analyze, and compare vulnerabilities of network control devices, especially as networks become more dynamic and evolve to adapt to future use cases. This thesis presents the design, implementation, evaluation, and discussion of two tools that can be used for automatically discovering subsets of policies and vulnerabilities of network control nodes. These tools treat the nodes as black boxes by crafting network traffic input and observing the nodes' output. We use them to discover how the nodes react to changes in traffic characteristics as networks evolve.

For LTE cellular networks, we built a scalable test bed that is the first of its kind for analyzing the impact of traffic scalability and large-scale attacks on the Packet Core. Cellular networks are facing drastic change with the introduction of the Internet of Things, expecting to bring billions of connected devices in the near future. However, LTE is known to be vulnerable to attacks, and it has been theorized that the traffic nature of IoT devices will be troublesome for cellular networks, particularly at the volume expected of these devices. This thesis provides the first study demonstrating the impact of large-scale attacks on availability and scalability of Machine-to-Machine traffic. Studies have not previously been possible due to the scale and flexibility required for such an analysis. Our test bed, Firecycle, contains traffic models taken from real traces of smartphone and IoT devices and can be scaled up to an arbitrarily-sized network. The results of our analysis highlight the points in the network most severely impacted by IoT scalability and signaling overload due to a botnet of devices. They also provide insight on the QoS impact for smartphone users that could occur with a spike in the number of embedded devices communicating over LTE networks. The analysis discussed in this thesis is very beneficial for understanding particular vulnerabilities of control plane signaling and can aid in the design of future cellular network architectures.

Understanding vulnerabilities of network control devices is useful not only to enhance their security, but it also can assist in finding paths for circumvention of these devices and understanding how evasion is possible. However, tools are needed to deeply understand decision making behavior and implementation details when direct access is not possible and the device acts as a black box. The final chapter

of this thesis presents Autosonda, a tool to discover and study decision models of censorship devices. Through network traffic alone, Autosonda fingerprints censorship devices by discovering their models and mechanisms for how they enforce rule sets. The strength of Autosonda is demonstrated in a study that we present of 76 web filters currently in use in the New York City metropolitan area. In our study we encounter a great variety of behavior and implementation techniques for blocking prohibited web content. Not only does Autosonda help us to find implementation flaws and rule sets, it also allows us to find circumvention paths for 100% of our test subjects. Being able to perform this type of detailed analysis automatically and at scale is a large contribution for understanding censorship and how network control device behavior can be classified.

Bibliography

[1] https://aws.amazon.com/ec2/.

[2] http://www.internetlivestats.com/internet-users/. 2016.

[3] 3rd Generation Partnership Project; Technical Specification Group Radio Access Network. "Evolved Universal Terrestrial Radio Access (E-UTRA) - Physical layer procedures. 3GPP TS 36.213." In: v9.3.0 (2010).

[4] 3rd Generation Partnership Project; Technical Specification Group Radio Access Network. "Evolved Universal Terrestrial Radio Access (E-UTRA) - Radio Resource Control (RRC) - Protocol Specification. 3GPP TS 36.331." In: v8.20.0 (2012).

[5] 3rd Generation Partnership Project; Technical Specification Group Radio Access Network. "Evolved Universal Terrestrial Radio Access (E-UTRA) - User Equipment (UE) procedures in idle mode. 3GPP TS 36.304." In: v9.11.0 (2012).

[6] 3rd Generation Partnership Project; Technical Specification Group Radio Access Network. "Evolved Universal Terrestrial Radio Access Network (E-UTRAN); S1 Application Protocol (S1AP). 3GPP TS 36.413." In: v11.4.0 (2013).

[7] 3rd Generation Partnership Project; Technical Specification Group Radio Access Network. "Evolved Universal Terrestrial Radio Access Network (E-UTRAN); S1 General Aspects and Principles. 3GPP TS 36.410." In: v10.3.10 (2012).

[8] 3rd Generation Partnership Project; Technical Specification Group Radio Access Network. "Physical layer aspects for Evolved Universal Terrestrial Radio Access (UTRA). 3GPP TR 25.814." In: v7.1.0 (2006).

[9] 3rd Generation Partnership Project; Technical Specification Group Services and Systems Aspects. "Study on Core Network Overload and Solutions. 3GPP TR 23.843." In: v0.7.0 (2012).

[10] 3rd Generation Partnership Project. *Long Term Evolution (LTE)*. http://www.3gpp.org/technologies/keywords-acronyms/98-lte.

[11] *AT&T has officially shut down its 2G network*. GSM Arena. http://www.gsmarena.com/at_t_has_officially_shut_down_its_2g_network-blog-22811.php. 2017.

[12] Nicholas Aase et al. "Whiskey, Weed, and Wukan on the World Wide Web: On Measuring Censors' Resources and Motivations." In: *FOCI*. 2012.

[13] Alexa. *The top 500 sites on the web by category*. http://www.alexa.com/topsites/category/Top/Adult. 2017.

[14] Charles Arthur. *More than 50 Android apps found infected with rootkit malware*. http://goo.gl/l7DJbT. 2011.

[15] Ramzi Bassil et al. "Signaling oriented denial of service on LTE networks." In: *Proceedings of the 10th ACM international symposium on Mobility management and wireless access*. ACM. 2012, pp. 153–158.

[16] Christian Bettstetter, Giovanni Resta, and Paolo Santi. "The node distribution of the random waypoint mobility model for wireless ad hoc networks." In: *Mobile Computing, IEEE Transactions on* 2.3 (2003), pp. 257–269.

[17] Jin Cao et al. "A Survey on Security Aspects for LTE and LTE-A Networks." In: *IEEE Communications Surveys and Tutorials* ().

[18] Jianjun Chen et al. "Host of Troubles: Multiple Host Ambiguities in HTTP Implementations." In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM. 2016, pp. 1516–1527.

[19] *Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update 2013–2018*. Cisco. http://goo.gl/XnJYCK. 2014.

[20] Richard Clayton. "Failures in a hybrid content blocking system." In: *International Workshop on Privacy Enhancing Technologies*. Springer. 2005, pp. 78–92.

[21] Richard Clayton, Steven J Murdoch, and Robert NM Watson. "Ignoring the great firewall of china." In: *International Workshop on Privacy Enhancing Technologies*. Springer. 2006, pp. 20–35.

[22] M. Dano. *The Android IM app that brought T-Mobile's network to its knees*. Fierce Wireless. http://goo.gl/O3qsG. 2010.

[23] Peter Danzig et al. "An empirical workload model for driving wide-area TCP/IP network simulations." In: *Internetworking: Research and Experience* 3.1 (1992), pp. 1–26.

[24] Roya Ensafi et al. "Examining how the Great Firewall discovers hidden circumvention servers." In: *Proceedings of the 2015 ACM Conference on Internet Measurement Conference*. ACM. 2015, pp. 445–458.

[25] R. Fielding and J. Reschke. *Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing. RFC 7230 (Proposed Standard)*. 2014.

[26] Chris Fleizach et al. "Can you infect me now?: malware propagation in mobile phone networks." In: *Proceedings of the 2007 ACM workshop on Recurring malcode*. ACM. 2007, pp. 61–68.

[27] Freedom House. *Freedom on the Net 2015*. 2015.

[28] C. Gabriel. *DoCoMo demands Google's help with signalling storm*. Rethink Wireless. http://goo.gl/dpLwyW. 2012.

[29] Emmanuel Gadaix. "GSM and 3G Security." In: *In BlackHat Asia*. http://tinyurl.com/85plhlv. 2001.

[30] *Global mobile statistics*. mobiThinking. http://goo.gl/pSWJJ2. 2013.

[31] D. Goldman. *Major banks hit with biggest cyberattacks in history*. CNN Money. http://goo.gl/4qCXG. 2012.

[32] Chan-Kyu Han et al. "Evaluation of authentication signaling loads in 3GPP LTE/SAE networks." In: *Local Computer Networks, 2009. LCN 2009. IEEE 34th Conference on*. IEEE. 2009, pp. 37–44.

[33] *How The New iPad Creates 'Signaling Storm' For Carriers*. Forbes. http://goo.gl/fzOxLR. 2012.

[34] Christoph Ide et al. "Influence of M2M communication on the physical resource utilization of LTE." In: *Wireless Telecommunications Symposium (WTS), 2012*. IEEE. 2012, pp. 1–6.

[35] Antonio Iera et al. "Special Issue on the Internet of Things." In: *IEEE Wireless Communications*. Vol. 17. 2010, pp. 8–9.

[36] Internet Society. *Global Internet User Survey*. http://www.internetsociety.org/surveyexplorer/. 2012.

[37] Sibren Isaacman et al. "Human mobility modeling at metropolitan scales." In: *Proceedings of the 10th international conference on Mobile systems, applications, and services*. ACM. 2012, pp. 239–252.

[38] Xin Jin et al. "SoftCell: Taking Control of Cellular Core Networks." In: *arXiv preprint arXiv:1305.3568* (2013).

[39] G. Kambourakis et al. "DoS attacks exploiting signaling in UMTS and IMS." In: *Computer Communications* 34.3 (2011), pp. 226–235.

[40] Kestrel Signal Processing, Inc. *The OpenBTS Project*. http://openbts.sourceforge.net/.

[41] Sheharbano Khattak et al. "Towards illuminating a censorship monitor's model to facilitate evasion." In: *Presented as part of the 3rd USENIX Workshop on Free and Open Communications on the Internet*. 2013.

[42] Masood Khosroshahy et al. "Botnets in 4G cellular networks: Platforms to launch DDoS attacks against the air interface." In: *Mobile and Wireless Networking (MoWNeT), 2013 International Conference on Selected Topics in*. IEEE. 2013, pp. 30–35.

[43] Byoung-jo Kim and Paul Henry. "Directions for future cellular mobile network architecture." In: *First Monday* 17.12-3 (2012).

[44] Jeffrey Knockel, Jedidiah R Crandall, and Jared Saia. "Three Researchers, Five Conjectures: An Empirical Analysis of TOM-Skype Censorship and Surveillance." In: *FOCI*. 2011.

[45] B. Krebs. *Hacked Cameras, DVRs Powered Today?s Massive Internet Outage*. Krebs on Security. https://goo.gl/LVkgk9. 2016.

[46] Christian Kreibich et al. "Netalyzr: illuminating the edge network." In: *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*. ACM. 2010, pp. 246–259.

[47] P.P.C. Lee, Tian Bu, and T. Woo. "On the Detection of Signaling DoS Attacks on 3G Wireless Networks." In: *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*. May 2007.

[48] Debi Lewis. *Closing in on the Future With 4G LTE and M2M*. Verizon Wireless News Center. http://goo.gl/ZVf7Pd. 2012.

[49] *M2M Report: Wireless Health Market 2018*. M2M Magazine. http://goo.gl/BGCcP3. 2014.

[50] Marcello Mari. *How Facebook's Tor service could encourage a more open web.* The Guardian. 2014.

[51] J. Markoff. *Firm Is Accused of Sending Spam, and Fight Jams Internet.* The New York Times. http://goo.gl/76ipU. 2013.

[52] *McAfee Threats Report: Second Quarter 2013.* McAfee Labs. http://goo.gl/qJPh3e. 2013.

[53] Lucas Mearian. *Wilocity, Qualcomm demo first multi-gigabit wireless chipsets.* ComputerWorld. http://goo.gl/0bbOj1. 2013.

[54] Jelena Mirkovic et al. "The DETER project: Advancing the science of cyber security experimentation and test." In: *Technologies for Homeland Security (HST), 2010 IEEE International Conference on.* IEEE. 2010, pp. 1–7.

[55] *Mobile internet devices will outnumber humans this year.* The Guardian. http://goo.gl/dQuwxI. 2013.

[56] *More than 50 billion connected devices.* Ericsson White Paper. http://goo.gl/BVtvbq. Ericsson, 2011.

[57] Collin Mulliner and Jean-Pierre Seifert. "Rise of the iBots: 0wning a telco network." In: *Proceedings of the 5th IEEE International Conference on Malicious and Unwanted Software (Malware).* 2010.

[58] *Nationwide Public Safety Broadband Network.* US Department of Homeland Security: Office of Emergency Communications. http://goo.gl/AoF41. 2012.

[59] *OPNET Modeler.* http://goo.gl/oCMo0b.

[60] *OPNET System in the Loop.* http://goo.gl/aP16T3.

[61] T Petsch et al. "Influence of Future M2M Communication on the LTE system." In: *Wireless and Mobile Networking Conference (WMNC), 2013 6th Joint IFIP.* IEEE. 2013, pp. 1–4.

[62] Roger Piqueras Jover. "Security Attacks Against the Availability of LTE Mobility Networks: Overview and Research Directions." In: *IEEE Global Wireless Summit - Wireless Personal Multimedia Communications Symposium (GWS - WPMC).* Atlantic City, NJ, 2013.

[63] G. Piro et al. "Simulating LTE Cellular Systems: An Open-Source Framework." In: *Vehicular Technology, IEEE Transactions on* 60.2 (2011), pp. 498–513. ISSN: 0018-9545. DOI: 10.1109/TVT.2010.2091660.

[64] A Prasad. "3GPP SAE-LTE Security." In: *NIKSUN WWSMC*. 2011.

[65] Feng Qian et al. "Characterizing radio resource allocation for 3G networks." In: *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*. ACM. 2010, pp. 137–150.

[66] Radmilo Racic et al. "Exploiting Opportunistic Scheduling in Cellular Data Networks." In: *Proceedings of the 15th Annual Network and Distributed System Security Symposium (NDSS 2008)*. Citeseer. 2008.

[67] Srinivasa Rao and Gajula Rambabu. *Protocol Signaling Procedures in LTE*. White Paper. http://goo.gl/eOObGs. Radisys, 2011.

[68] Dipankar Raychaudhuri et al. "Overview of the ORBIT radio grid testbed for evaluation of next-generation wireless network protocols." In: *Wireless Communications and Networking Conference, 2005 IEEE*. Vol. 3. IEEE. 2005, pp. 1664–1669.

[69] Chris Riederer et al. "I Don't Have a Photograph but You Can Have my Footprints—Revealing the Demographics of Location Data." In: *Proceedings of COSN '15*. 2015.

[70] *Security Alert - Android Trojan GGTracker Charges Premium Rate SMS Messages*. The Lookout Blog. http://goo.gl/oo5oQZ. 2011.

[71] *Security Alert - SpamSoldier*. The Lookout Blog. http://goo.gl/7lkRM. 2012.

[72] Jeremy Serror. "Impact of paging channel overloads or attacks on a cellular network." In: *Proceedings of the ACM Workshop on Wireless Security (WiSe)*. 2006.

[73] S. Sesia, M. Baker, and I. Toufik. *LTE, The UMTS Long Term Evolution: From Theory to Practice*. Wiley, 2009. ISBN: 9780470697160.

[74] M.Z. Shafiq et al. "Large-Scale Measurement and Characterization of Cellular Machine-to-Machine Traffic." In: *Networking, IEEE/ACM Transactions on* 21.6 (2013), pp. 1960–1973.

[75] Alex Shye et al. "Characterizing and modeling user activity on smartphones: summary." In: *ACM SIGMETRICS Performance Evaluation Review*. Vol. 38. 1. ACM. 2010, pp. 375–376.

[76] *Signal storm caused Telenor outages*. Norway News in English. http://goo.gl/pQup8e. 2011.

[77] *The Tor Project*. https://www.torproject.org/.

[78] Ardy Thompson. *Army examines feasibility of integrating 4G LTE with tactical network*. The Official Homepage of the United States Army. http://goo.gl/F6OYNA. 2012.

[79] John Thompson et al. "Guest Editorial on 5G Wireless Communication Systems: Prospects and Challenges." In: *IEEE Communications Magazine*. Vol. 52. 2014, pp. 62–64.

[80] Patrick Traynor, Patrick McDaniel, and Thomas La Porta. "On attack causality in internet-connected cellular networks." In: *Proceedings of 16th USENIX Security Symposium on USENIX Security Symposium*. Boston, MA: USENIX Association, 2007, 21:1–21:16. ISBN: 111-333-5555-77-9.

[81] Patrick Traynor et al. "Exploiting open functionality in SMS-capable cellular networks." In: *J. Comput. Secur.* Vol. 16. Amsterdam, The Netherlands, The Netherlands: IOS Press, 2008, pp. 713–742.

[82] Patrick Traynor et al. "Mitigating attacks on open functionality in SMS-capable cellular networks." In: *IEEE/ACM Trans. Netw.* Vol. 17. Piscataway, NJ, USA: IEEE Press, 2009, pp. 40–53.

[83] Patrick Traynor et al. "On cellular botnets: measuring the impact of malicious devices on a cellular network core." In: *Proceedings of the 16th ACM conference on Computer and communications security*. CCS '09. Chicago, Illinois, USA: ACM, 2009, pp. 223–234. ISBN: 978-1-60558-894-0.

[84] Michael Carl Tschantz, Sadia Afroz, Vern Paxson, et al. "SoK: Towards Grounding Censorship Circumvention in Empiricism." In: *Security and Privacy (SP), 2016 IEEE Symposium on*. IEEE. 2016, pp. 914–933.

[85] John-Paul Verkamp and Minaxi Gupta. "Inferring Mechanics of Web Censorship Around the World." In: *FOCI*. 2012.

[86] Philipp Winter and Stefan Lindskog. "How china is blocking Tor." In: *arXiv preprint arXiv:1204.0447* (2012).

[87] Philipp Winter and Stefan Lindskog. "How the Great Firewall of China is Blocking Tor." In: *FOCI*. 2012.

[88] *World Record: Wireless Data Transmission at 100 Gbit/s*. Karlsruhe Institute of Technology. http://goo.gl/kRyCP6. 2013.

[89] Xueyang Xu, Z Morley Mao, and J Alex Halderman. "Internet censorship in China: Where does the filtering occur?" In: *International Conference on Passive and Active Network Measurement*. Springer. 2011, pp. 133–142.

[90] Jamie Yap. *5 cool things you did not know M2M could do*. ZDNet. http://goo.gl/1SQAoC. 2013.

[91] *ooniprobe - Measure Internet Censorship & Performance*. https://ooni.torproject.org/post/ooni-mobile-app/.