Mobile Security Catching Up?
Revealing the Nuts and Bolts of the Security of Mobile Devices

Michael Becher, Felix C. Freiling
University of Mannheim, Germany

Johannes Hoffmann, Thorsten Holz, Sebastian Uellenbeck, Christopher Wolf
Horst Görtz Institute (HGI)
Ruhr-University Bochum, Germany

***Abstract*—We are currently moving from the Internet society to a mobile society where more and more access to information is done by previously *dumb* phones. For example, the number of mobile phones using a full blown OS has risen to nearly 200% from Q3/2009 to Q3/2010. As a result, mobile security is no longer immanent, but imperative. This survey paper provides a concise overview of mobile network security, attack vectors using the back end system and the web browser, but also the hardware layer and the user as attack enabler. We show differences and similarities between "normal" security and mobile security, and draw conclusions for further research opportunities in this area.**

***Keywords*-mobile security; smartphones; survey**

I. Introduction

The beginning of the smartphone era can be seen as beginning with the new millennium. Since then, numerous new "smart" devices like BlackBerries, iPhones and, recently, Android-based phones have been introduced that revolutionized the market. At the same time, many articles about smartphone security and the potential of malicious software on them were published [1]–[8]. Quite often, studies had statements similar to the following quote by Gartner which estimated "that by the end of 2007, enough factors will have come together that the risk of mobile attacks will be much greater. Those factors include less heterogeneity in operating systems, more penetration of smartphones and a greater incidence of people actually accepting downloads and sending executables to one another on mobile devices" [9]. However, up to now the expected plethora of attacks has not been observed.

Many researchers and practitioners are expecting a major security incident with mobile phones ever since these devices began to become more powerful: with increased processing power and memory, increased data transmission capabilities of the mobile phone networks, and with open and third-party extensible operating systems, phones become an interesting target for attackers. However, no major incident has happened as of the time of this writing.

The reasons for this are unclear. However, certain inherent aspects seem to have a positive effect on security, one of them being the heterogeneity of mobile operating systems. Contrary to the prediction quoted above, heterogeneity of mobile operating systems has actually *increased* instead of decreased. Besides the operating systems Windows Mobile and Symbian OS, the mobile world has seen the advent of the iPhone's iOS and the Linux-based Android operating system during the last few years. Despite of their young age, both operating systems already gained their market share and they are predicted to even increase it in the future. Table I provides an overview of global sales figures and market share for mobile operating systems and the huge growth of Android is clearly visible. Second, it might simply be the case that mobile operating systems are sufficiently secure today as voiced by Bontchev [10]. Hence, this might be another reason why no major security incident has happened until now. Third, there may be additional factors such as the different network topologies: for the Internet, it is nearly end-to-end, while strongly hierarchical for mobile networks. Last but not least, there is also the effect of the "self-defeating prophecy" of mobile security: Having the strong example of desktop insecurity, plus plausible attack scenarios, the claims of mobile insecurity might have triggered mobile security. Overall, the reasons for the non-existence of major security incidents for mobile phones are still unclear up to now.

However, we recently saw the first real attacks against smartphones: In March 2010, Iozzo and Weinmann demonstrated a drive-by download attack against an iPhone 3GS that enabled an attacker to steal the SMS database from the phone [12]. In November 2010, one of the first public exploits to perform an attack against the mobile browser

Table I
Global Sales Figures and Market Share of Mobile Operating Systems for Third Quarter of 2009 and 2010 [11]

	3Q'09		3Q'10	
OS	units/1k	share [%]	units/1k	share [%]
Symbian	18,314	44.6	29,480 ↑	36.6 ↓
Android	1,424	3.5	20,500 ↑↑	25.5 ↑↑
iOS	7,404	17.1	13,484 ↑	16.7 ↓
RIM	8,522	20.7	11,908 ↑	14.8 ↓
Windows	3,259	7.9	2,247 ↓	2.8 ↓
Linux	1,918	4.7	1,697 ↓	2.1 ↓
Others	612	1.5	1,214 ↑	1.5 =
Total	41,093	100.0	80,532	100.0

shipped with Android was released [13]. Recently, Weinmann introduced the first over-the-air exploitation of memory corruptions in GSM software stacks [14] and Oberheide and Lanier identified several attack vectors against the iTunes App Store [15]. So it is not far fetched to ask whether we are now at the beginning of an era of attacks against smartphones?

In this paper, we survey the area of *smartphone security*. This topic covers all mechanisms that are intended to increase the security of sophisticated mobile devices. The contributions of this paper are two-fold. First, we survey and structure the state of the art in the area of smartphone security. We systematize the research that has been done in this area in the last years and provide a categorization. Second, we present directions of future research on these subjects and outline challenges that we expect to emerge. In summary, this paper provides a detailed overview of different aspects of the topic smartphone security and it serves as a guide for past, present, and future work in this area.

II. From Mobile to Security

In this section, we first introduce some terms we use throughout the paper and then clarify why *mobile security* is a topic of its own. This extends some preliminary work by Oberheide and Jahanian, who recently performed a brief survey of this area [8].

A. Initial Definition

As a first approach, the investigation subject of this paper is defined as: *any mobile device that contains a smartcard that is controlled by a mobile network operator (MNO)*. Intuitively, this is the definition of a mobile phone.

This definition is too broad for us because it also covers mobile phones that are not in the focus of this paper. These are mainly the kind of phones that can only be used for the phone functionality (plus text messaging and some basic other functionality), often aligned with a limited display size. Such phones are called *feature phones*. They sometimes have proprietary operating systems and are not extensible with additional software. Even though the applications on these phones can be attacked, *e.g.*, Denial of Service (DoS) attacks with malformed short messages, they are not the typical attack target of mobile malicious software.

Other exceptions are some restricted environments that are not in the focus of this paper either: USB sticks that enable laptops to use the mobile network are also not covered. Moreover, there are some other devices with operator-controlled smartcards that are a restricted environment of their own (*e.g.*, machine-to-machine types of communication). Both are not extensible with third-party software and the operating systems are proprietary developments.

Mobile devices also have other communication interfaces like WLAN and Bluetooth, and malicious software exists that only uses these interfaces for spreading. Consequently, devices can be imagined that do not have a connection to a mobile network, *i.e.*, do not contain an operator-controlled smartcard, but are attackable by mobile malware. Fortunately, all relevant mobile device operating systems provide the interface to the mobile network together with the local communication interfaces. That is why the intuitive definition from the beginning still holds.

B. Definition & Discussion

A more rigid definition follows now as well as a distinction concerning the possible security mechanisms. We define an *MNO smartcard* as follows: *an MNO smartcard is a smartcard inside the mobile device that is controlled by a mobile network operator*. Whenever this term is used in this paper, it can be used for all smartcards in mobile devices that are controlled by an MNO regardless of the actually used technology. A second important term is *smartphone*, which we define as follows: *a smartphone contains an MNO smartcard with a connection to a mobile network. Moreover, it has an operating system that can be extended with third-party software*.

The term “smartphone” as one word is chosen intentionally. It is supposed to denote that not only “smart phones” are under attack, but that the smartphone with its two main properties defines a class of attack targets and protection needs, which takes place in a setting with mobile devices connected to the network over a wireless link and a more centralized environment of the network operators. Additional properties of these smartphones can be found in the literature [16]. We sometimes use the term *mobile device* as a synonym for smartphone within this paper. Smartphones offer various services to its users. Popular is messaging as *Short Message Service* (SMS) and *Multimedia Messaging Service* (MMS). They use certain protocols that are explained in the literature [17] and we discuss the security aspects of them later.

In contrast to mobile devices, traditional computers are called hereafter *ordinary computers*. When their fixed location is emphasized, they are called *desktop computers*.

C. Specifics of Mobile Devices

A central question for the topic *smartphone security* is: In what sense is research on the security of mobile devices different from common security research? Is it possible to transfer known security solutions from ordinary desktop computers to mobile devices? Could it possibly be the same, only with the additional word “mobile” in the title?

We argue that there are specifics of mobile device security that justify independent research on this topic. We discuss in the following unique features of mobile security compared to ordinary computer security. They are the basis to novel security mechanisms especially designed for mobile devices and their infrastructure, and these mechanisms cannot be transferred from existing computer security solutions. In

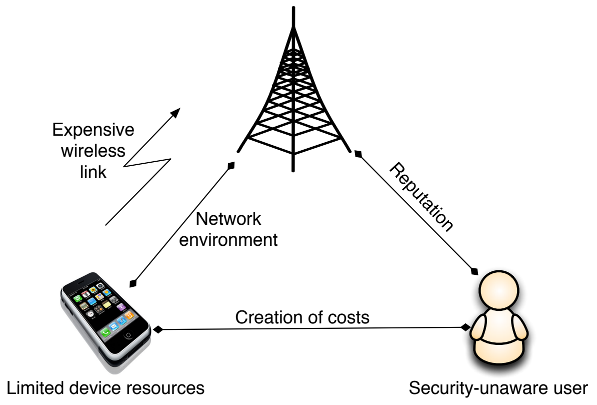


Figure 1. Specifics of Mobile Devices

addition, mobile devices have a specific bundle of attack vectors which are new to some organizations and also individuals. An overview of these differences is shown in Figure 1 and they will be introduced subsequently.

1) Creation of Costs: The specific *creation of costs* is the inherent possibility for attackers to generate costs for the user and revenue for the attacker. It has two aspects: events that are billed by mobile network operators (*e.g.*, phone calls or messages) and arising payment systems.

Billed Events: The problem of billed events existed previously in desktop security when dial-up connections via modem or ISDN lines were common. Malware (so called *dialers*) could dial premium-rate numbers and with it directly provide profit to the malware author. With the appearance of broadband connections (like DSL) this problem mostly vanished, because the computer is now directly connected to a computer network and does no longer have a direct interface to the premium-rate numbers of the telephone network. However, with mobile devices the cost aspect will likely be a problem for a long time. Even if flat rates for data or voice services become common, separately charged premium services will most likely be still available.

Payment Systems: A first type of payment systems uses the messaging functionality of mobile phones as a trustworthy channel for transmitting authorization information, *e.g.*, online banking with mobile transaction numbers or online payment services. In general, there are two communication channels that need to be compromised. However, the mobile device is the only channel that needs to be compromised if an attacker has access to the authentication information of the targeted account. Customized mobile malware might forward the messages to the attacker [18] or respond to them in the expected form. The necessity of these attacks being customized makes it more probable that mobile malware will use the cost-creating functionality of the mobile network.

A second type of payment systems uses mobile phones as payment devices and physical proximity as part of the authorization process, *e.g.*, payments based on Near Field Communication (NFC). In this case, the required proximity to the receiver of the payment enhances the security and makes these attacks unlikely compared with directly using the mobile network cost-creating functionality. When this feature becomes more widespread and more standardized, we expect a strong increase of incidents.

2) Network Environment: The specific *network environment* consists of the three aspects strong connection, firmware update process, and remote device management.

Strong Connection: Strong connection means the presence of the MNO and its influence on the device. Different from ordinary computers where the network provider almost always has no influence on the user's machine, the MNO owns the smartcard inside the mobile phone. Furthermore, the smartcard is a trusted device. It is possible to create trusted applications on the mobile phone with enhanced security. Although TPMs (Trusted Platform Module) appear in mobile devices, it remains an open question how to easily bootstrap trust between MNO and TPM.

Firmware Update Process: The process of updating the firmware of mobile devices changed rapidly during the last few years. A few generations of mobile phones ago, an update of a firmware could only be done in a local setting, possibly only by the device manufacturer himself. With the rise of smartphones and extensible operating systems, more sophisticated hardware architectures have been introduced. These new architectures enable firmware or third-party software updates remotely.

Even though remote updates are possible today and updates nowadays do not differ much from ordinary computers, updating mobile devices remains a challenging task. If not connected to a host computer on a regular basis, an update process has to use the expensive wireless interface.

Updating the firmware over the air is an important functionality to update vulnerable parts of the mobile device's operating system. It is also a critical feature, because most update procedures cannot be interrupted without damaging the device. Instead of a complete firmware update, the exchange of single files of the operating system's file structure is better suited. This is especially true in terms of wireless communication and device resource costs.

An additional aspect is the entity that starts the update. This has traditionally been the mobile network operator, but only recently manufacturers started to control the firmware update process themselves (examples are iOS and Android).

Remote Device Management: An important feature of mobile devices is the ability to be managed by a remote entity. This is due to the fact that usually some entity has more power over the device than in ordinary computer environments, *e.g.*, the mobile network operator, the device manufacturer, or the corporate IT department.

A user typically notices such feature changes as remote configuration updates, for example, when MMS or WAP (*Wireless Application Protocol*) settings are pushed to the device by the MNO. Other feature changes are mainly targeted at corporate environments, where the IT department has to enforce a corporate security policy on such devices. Examples of these features are disabling Bluetooth, WLAN, or memory card interfaces to prevent leaks of corporate data from protected devices. An interesting feature in this context is the *remote wiping* functionality. Lost or stolen devices can be deleted completely by a remote entity [19], [20]. This feature is mandatory in some regulated industries.

3) Limited Device Resources: A smartphones typically has *limited resources* as we discuss in the following.

Resource Limitations: The limited resources of a mobile device are the most obvious difference to ordinary computers. Even though it is always said that mobile devices today have the computing power of desktop computers of "some years ago", they are still limited compared to high-end computers. The main limiting factors are the CPU and memory such as RAM. These two factors limit the sophistication of possible security solutions, *e.g.*, sophisticated intrusion detection algorithms that hardly work for real-life applications on ordinary computers cannot be transferred to mobile devices in the foreseeable future.

Battery: A unique factor of smartphones is the battery, which severely limits the resources available for a security solution from the point of view of the general acceptance factor. Although Joe Sixpack might not notice this point, it is important that a security solution does not constantly drain large portions of available CPU time to avoid battery exhaustion.

4) Double Expensive Wireless Link: Another specific of mobile security is the *expensive wireless link*. The term *expensive* is meant twofold here. First in terms of computational costs for the algorithms and second in terms of battery power. It does *not* point to monetary costs for the user here.

Expensive Computation Costs: Compared to local computations on the device, the wireless link is always expensive for an algorithm. Thus, solutions for increasing security of mobile devices should try to avoid this communication. On the other hand, transferring computation load from the device to the mobile network is desirable as the device resources are limited. Hence, we have a trade-off here between the limited device resources (*e.g.*, processing power and memory), the design of security algorithms using the computing resources of the mobile network, and the expensive communication between these two, which needs to be balanced out and which might lead to different solutions for the same user during the lifespan of a mobile device.

High Monetary Communication Costs: A minor aspect are the communication costs, *i.e.*, the costs of using the mobile network. Communication cost means that either the user has to pay for the security solution or the network operator has to consider these communication costs in the calculation of its flat rate conditions. However, this is only a side aspect compared to the computation costs.

5) Reputation: The specific *reputation* can be seen as a weak specific of mobile devices. The mobile network operator will invoice every event that generated costs, even though it might have been generated by malicious software or an attacker. Therefore, it can be thought that the mobile network operator could be held responsible from the user's point of view. In case of a widespread mobile malware outbreak with several network operators involved, mobile malware might even have an impact on the reputation of the entire mobile phone system in general.

III. Attack Vector Classes and Attack Models

In this section, we present a classification of attack vectors for smartphones which we use as a framework for the rest of this paper. Its intention is to show the relevant attack vectors that can be used by an attacker.

Mobile device threats are classified here as belonging to one out of four classes: *hardware-centric*, *device-independent*, *software-centric*, and *user layer* attacks [21]:

- *Hardware-centric attacks* belong to mobile device security only from a broader point of view. Even though they are suited to violate security properties (*e.g.*, confidentiality of personal data violated by forensic analysis), they are not suited to be easily exploitable by an attacker, because these vulnerabilities typically cannot be exploited remotely, but only with physical access to the mobile device.
- *Device-independent attacks* directly belong to the protection targets of the mobile device user: eavesdropping on the wireless connection or leaking mirrored personal data on back end systems both violate confidentiality of the user's personal data.
- In the context of this paper, the most important class of technical vulnerabilities for mobile devices are *software-centric attacks*. Especially the rise of the—hardly security-specified—mobile web browser led to various exploited vulnerabilities in the recent past.
- *User layer attacks* contain every exploit that is not of technical nature. Many of today's mobile malware samples are not based on a technical vulnerability, but trick the user into overriding technical security mechanisms [5]. This is an important class of vulnerabilities, even if not of technical nature. Nevertheless, we do not discuss this aspect in detail in this paper since the topic is too broad to cover within our analysis.

From the point of view of defending against vulnerabilities, every class is separate from the others and needs its own security mechanisms. We will discuss the individual vectors in the next few sections.

In addition to these attack vectors, we also consider different *attack models*. Basically, attack vectors investigate *vulnerabilities on the victim's side*, while attack models limit the power of an attacker. More specifically, we distinguish between *passive attackers* who do not alter the content sent and *active attackers* who might do. Obviously, the second is more powerful than the first, while the passive attacker is more likely to go unnoticed compared to the active one. Both attackers might have the following goals:

- *Eavesdropping:* A passive attacker tries to intercept the conversation between mobile phone and base station and therefore (implicitly) between the user of the phone and her caller. In Section V-A, we will see how an active attacker can make this scenario far more likely.
- *Availability Attacks:* One possible example is an active attacker blocking the signal of the mobile phone or base station, for example via jamming and therefore rendering the mobile service unusable.
- *Privacy Attacks:* A passive attacker might use the smartphone's ID to locate its owner. Again, this attack can be made more efficient using an active attacker.
- *Impersonation Attacks:* In a nutshell, one mobile phone impersonates as another in such an attack. For example, a mobile phone uses the service of a base station without billing facility for the base station, *i.e.*, the service is used in a fraudulent way.

In the next four sections, we investigate in detail the security aspects of the four different security classes and present past work and future challenges in these areas.

IV. Hardware-Centric Security Aspects

We subdivide this attack vector into attacks on removable security modules of mobile devices, especially the MNO smartcard, and attacks against the device itself.

A. Intercepting MNO Smartcard Communication

Communication between the mobile device and the MNO smartcard is not encrypted because a man-in-the-middle (MITM) attack on this communication was considered infeasible when this interface was specified. However, nowadays a product named *TurboSIM* [22] successfully implements an MNO smartcard MITM attack. It is a small chip that intercepts the communication between the MNO smartcard and the mobile device and is attached by removing a small part of the smartcard's plastic frame. With the usage of TurboSIM it was possible to successfully remove the SIM lock of the iPhone [23]. As the hardware interface is the same for 2G SIM (*Subscriber Identity Module*) cards and 3G UICCs (*Universal Integrated Circuit Card*), it is possible to use TurboSIM for both settings. A recently started project called Osmocom SIMtrace is also able to trace the communication between the SIM card and the mobile device [24].

Without regarding the limitations of the actual implementation of TurboSIM, in general, such a MITM attack can change all communication between MNO smartcards and mobile devices and even inject new messages. This can be mitigated by encrypting the communication: As the attacking devices have no access to the internals of the MNO smartcard or the mobile device, the attack would no longer be easily realizable.

However, it is difficult to address this attack vector with billions of vulnerable devices deployed world-wide. From a high-level point of view, it is an engineering task, but there are several challenges involved. For the solution sketched above, we are now faced with the problem of the initial key exchange using only an untrusted channel.

B. Attacking the Device

Hardware-centric attacks that target the mobile device itself can be subdivided according to the status of the mobile device: switched on (JTAG attacks) or switched off (forensic analysis).

1) JTAG Attacks: Joint Test Action Group (JTAG) is a standard for testing and debugging hardware. Even though this debugging functionality is no longer necessary in mobile devices that are sold to end users, the JTAG functionality is sometimes still accessible. This functionality allows inspecting the device on a deep level, being able to lead to exploitable vulnerabilities. This threat is addressed by industry requirements [25].

2) Forensic Analysis: The forensic analysis of mobile devices is an attack vector targeting the confidentiality of the stored data. It is an unexpected attack vector and it is only valid in the case of an attacker getting physical access to the device. There are two common possibilities for that: an attacker that takes the device for a limited period of time without the owner noticing it, and a legitimate change of ownership. Especially the second case is common today and as some studies show, it encompasses data from personal conversations to confidential corporate data [26], [27].

From a high-level point of view, this attack vector can be closed quite easily by adding sound encryption schemes to the data. Since smartphones are carried around they are prone to getting lost or stolen. In order to protect the stored data on it, non-volatile memory should be encrypted. Further, a secure store for cryptographic keys should be used to protect these against threats from the smartphones' applications itself. A TPM or special functionality of a SIM card may be utilized for this. Dealing with the solution in more detail leads to the consideration that cryptographic functions need the limited device resource processing power, leading to increased battery usage. Therefore, encryption vs. battery life need to be weighted against each other. Using specific hardware oriented ciphers, this choice becomes easier. In particular, designing a battery-friendly cipher is an open question which would have impact on this question.

V. Device-Independent Security Aspects

Device-independent vulnerabilities directly belong to the protection targets of mobile device users. Both eavesdropping the wireless connection (Section V-A) and leaking mirrored personal data on back end systems (Section V-H) violate the confidentiality of the user's personal data. Similar to the device-centric attacks of Section IV, these attacks cannot be exploited by mobile malware either. An exception could be the wireless pairing process, which could be influenced by mobile malware, *e.g.*, by forcing the device to connect to a rogue access point or base station.

A. GSM: Cryptography for Protecting the Air Link

Unlike land lines, GSM uses radio waves to connect different participants. More specifically, a mobile phone and a base station are linked via an (encrypted) channel. From a security point of view, we have several issues to consider in this setting.

Within the GSM specification, several security mechanisms are in place to prevent the attacks outlined in Section III—at least in principle. In a nutshell, each GSM phone holds a SIM card which supplies all cryptographic secrets and also cryptographic algorithms. Note the design decision here to split the mobile and user data (*e.g.*, address book) from the cryptographic secrets. In particular, we speak about the A3 algorithm for *authentication*, the A8 algorithm for *key derivation*, and the A5 algorithms (A5/1, A5/2, and A5/3) for *encryption* and the "algorithm" A5/0 for *no encryption*. For describing the protocol, we will use a more concise notion—skipping details on lower protocol levels—without abstracting away any security problem. In the following, we relate the security objectives from above to the corresponding steps in the protocol, and also discuss weaknesses and possible mitigations or even remedies.

B. Initial Connection and Encryption

To use the mobile system, a phone must prove that it has access to a genuine SIM card. To this end, symmetric cryptography is used. While asymmetric crypto might be better suited for this purpose, it was too heavy weight 25 years ago when the protocols were designed and still puts a burden on the battery of mobile devices. Hence, all solutions below use symmetric cryptography only.

In a nutshell, a secret s is used together with some fresh randomness or a nonce r to derive a new authentication string $a := \mathrm{A3}(s, r)$, and a fresh shared key $k := \mathrm{A8}(s, r)$. This key k is now used to encrypt further communication between the base station and the mobile phone. The corresponding protocol is depicted in Figure 2. The above protocol has some interesting features regarding the requirements discussed above. In particular, we can see that step 3 authenticates the mobile against the base station and therefore prevents fraud, in particular an *impersonation attack*. In addition, each mobile is given a temporary identifier t in step 4. This prevents tracking and hence *privacy attacks*. In the steps at the bottom of the figure, the protocol generates a fresh session key k that ensures that communication is protected from *eavesdropping*. Only jamming as a special *availability attack* is not prevented in this context. However,

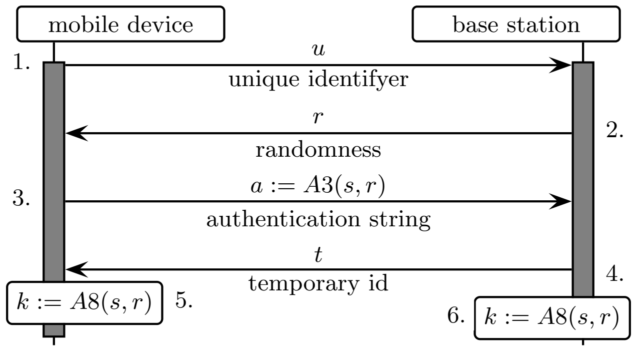


Figure 2. Initial Handshake in GSM

technically, there is nothing we can do from a cryptographic perspective to counter this attack. We therefore rely on other protocol layers to take care of this (*e.g.*, by frequency hopping).

C. Initial Problems

Without taking any further parts of the protocol into account, we start with an analysis of known weaknesses and possible remedies.

First, we note that the key derivation algorithm A8 is used for *any* encryption algorithm A5/1, /2, /3—and that A5/2 is far weaker than its counterparts. In particular, A5/2 has been specifically weakened for the use in non-Western countries and can be broken in a matter of seconds [28]. Apart from using a weak algorithm, GSM made a second, vital mistake: Rather than first encrypting the message and *then* encoding it for air transit, GSM specified it the other way around. As a result, cryptanalysis has plenty of redundancy to work with (which was subsequently exploited in the attack referenced above). Moreover, each mobile phone can be told which algorithm to use in a specific network by this very network. Hence, the following attack is feasible:

1) The mobile device is tricked by its counterpart into believing that only A5/2 is supported by the current network.
2) Key derivation takes place with some "random" value r (cf. Figure 2).
3) A phone conversation using the corresponding key k is encrypted.
4) This session key is derived by breaking A5/2 [28].
5) Now, all conversation encrypted with this session key k can be eavesdropped, no matter which encryption was used.

Interestingly, the latter also applies to phone conversations which *previously* were recorded by an eavesdropper. The reason is the following observation: the mobile has no control over the random value r, but an active attacker has full control over it. The problem is made worse by the fact that no network authentication takes place. Hence, everybody can set up a rogue base station, called an "IMSI Catcher" (*International Mobile Subscriber Identity*) [29].

Today, it is possible to set up a rogue base station using open source software and cheap hardware [30], [31].

Before criticizing GSM for this flaw, we need to recall that the technology is more than 25 years old. Back then, it was actually reasonable to assume that nobody would be able to duplicate a complete base station. As "lesson to be learned" we note that we should be careful about our security assumptions and the progress of technology. Fixed key-sizes come immediately into mind here. In addition, the wireless connection of a device and a radio access node of a mobile network can *always* be attacked by entities in the same physical area. They are called *evil twins* as they imitate the parameters of the legitimate communication partner.

In addition, we want to draw the attention to A5/3 and especially the cipher it involves (KASUMI), which is also used in UMTS (cf. Section V-F). While stronger than A5/2, severe weaknesses on its cryptographic strength are emerging [32]. While A5/3 is slightly tweaked so that the attack is not directly applicable, this raises doubts about the overall strength of KASUMI. In addition, it is not conclusively shown that A5/3 cannot be exploited in a similar way as KASUMI. This is mainly a security research question, as the algorithm needs to be tricked in processing a large quantity of GPRS (*General Packet Radio Service*) data and also into a special mode of operation. The latter is not obvious at the moment.

D. SMS Infrastructure Flaws

Beside phone and Internet services, smartphones are typically capable of sending text messages such as SMS and MMS. Because these features are an additional source of revenue for the carrier, in the early days of SMS the carrier boosted their service by permitting the sending of complimentary SMS through web providers. In 2005, Enck *et al.* evaluated the security impact of such SMS interfaces on the availability of mobile phone networks [33]. They demonstrated the ability to deny voice service to large cities by using a desktop computer connected to the Internet with a cable modem. Serror *et al.* evaluated the impact of abusing the paging channel to overload the network in 2006 [34]. In 2009, Traynor *et al.* refined the previously [33] revealed attack by using more realistic network conditions [35]. They showed that adversaries can achieve blocking rates of more then 70% with only limited resources. Therefore, one question looms: how can the SMS infrastructures robustness be improved so that abusing desired features can be mitigated? We will again pick up the topic of denying a service in Section VI-A.

E. MMS Vulnerabilities

In contrast to SMS, MMS does not suffer from the previously named GSM shortcomings because it does not use a GSM control channel to submit messages. While GSM connections are circuit-switched, GPRS—a GSM extension—makes use of packet-switching. MMS, a service capable of sending larger text messages or even multimedia messages from phone to phone, utilizes GPRS as infrastructure and WAP, SMTP and HTTP as transmission protocols. Its architecture consists mainly of one MMS Relay/Server and user agents residing on the mobile.

Racic *et al.* implemented a proof-of-concept attack that exploits MMS vulnerabilities to exhaust the mobile phone's battery [36]. In this scenario their first step is to build a target hit list by sending MMS notification messages from a false MMS Relay/Server, leading the victims to a malicious web server. When connecting to the web server, the victims disclose their IP addresses. The attacker takes advantage of the revelation by sending periodically UDP packets to the victim's mobile phone. As a result, the attacker prevents the victim's mobile phone from reaching standby mode. Since it is extremely battery consuming for mobile devices to stay in ready mode, they say that batteries are drained 22-times faster than in a mix of ready and standby mode. Taking into account that mobile phones do not indicate receipts of UDP packets, victims will not recognize the exhaustion of the battery until they observes the battery status or realize that the battery is empty.

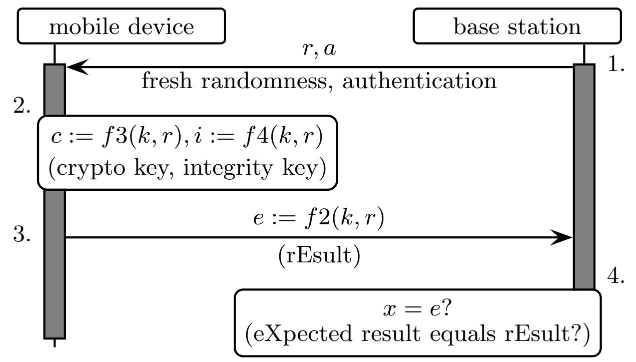
To make this scenario even worse, an attacker neither needs to build a target list nor send MMS to potential victims. It is sufficient to know the network address ranges assigned to a network operator in order to send UDP packets to all corresponding IP addresses. Thereby, all customers of an MNO would suffer from this exploit if their MNO assigns public instead of private IP addresses to a mobile phone. It remains an open question how service providers can handle stateless protocols to solve potential incidents without restricting the usability.

F. Initial Remedy: UMTS

UMTS (*Universal Mobile Telecommunications System*) is a successor of GSM and tries to avoid (most of) its flaws [37]. We will now investigate how well this went.

First, GSM failed to encrypt some vital services such as signaling or SMS. As a result, the corresponding services are vulnerable to attacks, as discussed in the two previous sections. All in all, this flaw was fixed in UMTS: encryption and encoding is in the correct order, the encryption algorithm has been updated to KASUMI, and parameter choices have been improved. In addition, all communication over the air link is encrypted within the network (in contrast to base station) and the network is authenticated against the mobile phone. This way, rogue base stations are avoided. Finally, UMTS was designed to be compatible with GSM.

To make our point, we compare the GSM handshake (steps 2, 3, 5 of Figure 2) with its counterpart in UMTS (Figure 3), the latter being called *Authentication and Key Agreement* (AKA). Note that there are two additional final steps. Sending the result e of *f2* to the base station and



mobile device
base station
r, a
fresh randomness, authentication
1.
2.
$c := f3(k,r), i := f4(k,r)$
(crypto key, integrity key)
3.
$e := f2(k,r)$
(rEsult)
4.
$x = e?$
(eXpected result equals rEsult?)

Figure 3. Authentication and Key Agreement (AKA) in UMTS

comparing it with the expected result x. Functions *f1–f5* are used to generate session keys and intermediate values from the fresh randomness r. First, we want to note that the mobile phone (to be more precise, the UICC) can verify that the randomness r is actually fresh. This can be achieved by using a block cipher in counter mode. Second, the authentication string a ensures network authentication as it depends on a shared secret k. Third, the authentication string a also contains an algorithm identifier. This is used to compute the MAC (*Message Authentication Code*) of all messages (including e). Therefore an attacker does not profit from "downgrading" a connection from a strong to a weak encryption algorithm.

However, the weakness on KASUMI is also valid here. Again, UMTS uses a slightly tweaked version of KASUMI, so it is not possible to apply the attack directly. But it is an interesting research question if this could be actually done.

Despite the cryptographically stronger AKA, UMTS suffers from an old GSM weakness: the IMSI is sent in clear and, therefore, could be eavesdropped. Furthermore, the integrity keys used between the mobile device and the Radio Network Controller (RNC) are transmitted unencrypted to the RNC [38]. Therefore, some flaws remain even in the GSM successor.

Recalling the evil twin base stations of GSM, we inspect if they also work on UMTS. The answer is affirmative. Note that in GSM networks only the mobile device has to authenticate itself (cf. Section V-A), and for increased security, UMTS was designed to provide mutual authentication of mobile devices and the network. Additionally, signaling information is integrity-protected as a mean to prevent evil twin base stations [39]. However, UMTS was also designed to be compatible to GSM, whenever no sufficient UMTS coverage can be provided. This compatibility makes a roll-back attack possible, where the compatibility mechanisms between these two mobile networking standards are exploited [40].

In addition, since no standard is perfect, several flaws have been found in the past years in UMTS. In 2007, a Denial of Service (DoS) attack was identified by Lee *et al.* that exploits the unique vulnerabilities of the signaling/control plane in an UMTS network [41]. By a well-timed, low-volume signaling attack, they can overload the control plane and detrimentally affect the key elements of the 3G infrastructure. Another DoS attack was demonstrated by Zhao *et al.* in 2009. By jamming the presence service, a core service of the IP Multimedia Subsystem (IMS), a chain reaction could be initialized, that blocks all services of IMS [42].

In theory, we could also extend this analysis to the 4G mobile networks. In practice, this is out of the scope of this paper. A detailed security analysis is therefore left as an open research question.

G. Side Channel Analysis

Taking a purely theoretical point of view, any algorithm a produces for an input i some output o, more formally: $o := a(i)$. However, this is only the theoretical picture. In reality, there is more to it. Actually, we have the following situation $o, \gamma := a(i)$ where γ is additional *side channel information* that can be observed by an attacker. This can be the rate of cache hits or misses, memory access, power consumption, or similar data sources. For cryptographic algorithms, this is fatal since i usually contains sensitive key material which should not be exposed. It has been demonstrated that this cannot be guaranteed in general [43]. In the case of SIM cards, attacks date back to 2002 [44]. Recently, Cryptographic Research has made a similar claim [45], although no attacks are known at the moment. Still, as they have pioneered research in this direction, their claims have some weight. In addition, they point to an interesting research area, *i.e.*, to exploit this attack vector in current devices.

However, the overall attack scenario of side channel analysis is not very likely in the case of SIM cards. Here, an attacker needs physical access to the SIM card to perform some measurements. While possible, this is not very plausible since users typically take their devices with them. Hence, the typical attack setting that is far more likely (and thus more interesting): are there side channels in SIM cards which can be accessed through malicious software on the phone? And in the more general case: Are there *any* side channels which can be accessed through the mobile phone? In particular, using exact timings it might be possible to establish such a side channel. Furthermore, could we use side channels such as cache hits to extract sensitive key material from some applications? For desktop computers, this has already been demonstrated [46].

H. Back End Systems

This section adds an attack vector to mobile device security that is not obvious at first glance, namely threats against the back end systems of mobile networks. However, a security incident in 2005 demonstrated how insecure back end systems can even compromise the privacy of mobile device users, as we now explain.

1) Danger Hiptop/T-Mobile Sidekick: The Hiptop device (named "Sidekick" in the T-Mobile version) of the US based company Danger, Inc. is a feature phone with a closed operating system. It differs from other mobile phones in storing its media data not only on the device itself, but mirroring the data in the MNO's network for Web accessibility. The data is protected by a password only. That means, it is possible to break a user's on-device data confidentiality by not attacking the mobile device at all.

The incident took place in the US T-Mobile network in 2005 and led to the publication of phone numbers and private data of prominent US citizens. It is reported by the Washington Post [47] to have been a combination of web application attacks and social engineering attacks. The web applications had a vulnerability that allowed to reset the access password to the mirrored data, resulting in locking the legitimate user out of its own account and giving the new password to the attacker. The only necessary piece of information for this attack was the mobile phone number. To find the mobile phone number of a prominent client of the MNO, a social engineering attack was performed on an MNO's store, tricking the employees to reveal an access password for internal systems of the MNO. From this starting point, it was possible to map names to phone numbers.

2) Attacks Against Home Location Register (HLR): Until now we evaluated a lot of security issues concerning the cryptography and the infrastructure. But what happens when a malicious software infects a number of mobile phones? Traynor *et al.* studied the impact of malicious devices, a mobile botnet strictly speaking, on a mobile network core [48]. They investigated the GSM back end and found the HLR (*Home Location Register*) to be the weakest point. The HLR is a central database that contains details of each mobile phone subscriber (customer) who are authorized to use the GSM core network. Furthermore, they showed how to reduce legitimate traffic by 93% respectively 75% when attacking the HLR running two different database systems by sending a large volume of traffic.

3) Other Back End Systems: Attacks on back end systems also comprise GPRS attacks [49] or attacks on the MMS infrastructure [36]. Moreover, the upcoming outsourcing of computation ("cloud computing") might lead to new privacy concerns and to new solutions for ensuring privacy. The solutions could possibly be transferred to solve the problems of back end systems with mobile devices.

VI. Software-Centric Security Aspects

Software-centric vulnerabilities are the most important class of vulnerabilities for mobile devices in respect to the attack model of this paper. Especially the rise of the—hardly security-specified—mobile web browser led to various exploited vulnerabilities in the recent past.

It is well known that it is very unlikely that software composed of thousands or even millions lines of code is bug-free; this is of course also true for operating systems powering modern mobile devices, especially smartphones. For years, these systems were closed source and proprietary and under almost no observance by security researchers and attackers. This has changed with Cabir [50], one of the first worms which propagated autonomously on mobile devices running Symbian OS in 2004. Since then, the security mechanisms of these systems are under inspection and malicious software targeting these devices is on a constant rise. Section VI-E1 provides an overview of malware on mobile devices. Since the first appearance of malware, all operating systems of the major competitors in this field were the target of malware writers and this trend will most likely continue with more advanced malware in the following years. The future situation of smartphone security will presumably share most aspects of the previous situation on computer security.

In this section, we first review the impact of malware and then focus on SMS and MMS specific aspects. This is followed by a discussion of the attack vector *web browser* and malicious software. We conclude with attacks against the operating system and user interface attacks.

A. Impact of Malware

We now discuss possible behavior and attack strategies of malware. Because malware can take every allowed action within its running environment, especially virtually any possible instruction when running with high privileges, we only cover the most significant malicious operations.

Information or Identity Theft, Espionage: A common malicious action is to collect any private accessible user information and to (secretly) forward it somehow to the malware author or its users. This kind of behavior may be embedded in inconspicuous looking (popular) applications such as games, which can easily be installed from (3rd-party) application-stores. One example is the recent discovery of such a game which is able to track users' locations [51]. The fact that a smartphone is a personal device and may be taken almost everywhere by a user, makes it an ideal target to snoop private or even confidential data. The application might collect the following information, which leads to a detailed profile of the affected victim: GPS coordinates, all kinds of credentials, several forms of communication (SMS, MMS, email, instant messaging, ...), contacts, accurate daily routines and personal habits, private or corporate documents, and so on. One example of such a software which only uses public available APIs (*i.e.*, it does not need a way to enhance its privileges in order to collect the desired private data) is the iPhone application SpyPhone [52]. The collected information may be forwarded through all of the smartphone's communication channels, which makes the detection of the unwanted behavior even harder. The latter is particularly true when coupled with both cryptographic and stealth techniques. We do not need to further explain the implications of the misuse of such data.

Eavesdropping: Next to the aforementioned stealing of private data, malware could also contain routines to capture voice calls and to silently record any conversations which are in range of the built-in microphone [53]. Depending on the privileges the malware has, this might happen completely in the background and will only be detectable by sophisticated monitoring of the whole operating system or the generated communication data. This type if eavesdropping is on a different layer than the aforementioned type in Section V.

Financially Motivated Attackers: As already shown in several papers [54]–[56], the business around malware got highly financially motivated in the recent years. Shady businesses were built to generate a lot of money from (initially) unaware victims. Not surprisingly, this trend has already reached smartphone malware as there is a strong connection between the smartphone and the MNO through some contract between the user and the provider in order to use the supplied services. Albeit the payment is often done in a flat rate model, some premium services are typically charged separately (*e.g.*, phone calls to special numbers or sending of short messages to some special services). The abuse of these services is ideal for attackers to generate money, *e.g.*, through offering a highly charged service number for short messages. An infected mobile device could covertly send messages to this number until the user might be aware of this situation on his monthly bill. One such malware is the malware *Trojan-SMS.AndroidOS.FakePlayer*, which pretends to be a movie player, but secretly sends messages to a service number which are highly charged [57]. Another way to generate money is to secretly redirect outgoing calls through a provider that generate an additional charge. Of course, this kind of MITM attack enables eavesdropping of the affected calls. Proof-of-concept malware with this behavior is evaluated by Liu *et al.* [58]. A third way to strip the victim of his money is to blackmail him. Although not yet seen on smartphones, one possibility is the encryption of private files and the release of the used key after some money has been paid ("ransomware"). An example of this is the *Gpcode*-Trojan for desktop computers. Mobile malware could set up similar extortion schemes, *e.g.*, by disabling certain services (such as email sending) on the mobile device and only re-enable them (for a short amount of time) after some payment has occurred (*e.g.*, by sending a premium short message to a number under the attacker's control).

An important prospective question is the way criminals earn money with mobile devices. Currently, premium-rate services or foreign country calls are such a method. In the future, smartphone-based payment systems could be exploited as well. With an abuse database and by enforcing this abuse database on a mobile device, we expect some of the currently working methods to cease.

The challenge for mobile network operators is contributing to the cessation of the current potential to earn money with exploiting mobile device security vulnerabilities, especially concerning premium services. Another challenge for future research in mobile device security is identifying the kind of successful attacks that cannot be solved with the security entities that are presented in this paper.

Mobile Botnets: Infected mobile devices are ideal remote controlled "machines", *e.g.*, within an established botnet. The different communication channels offered by smartphones enable much more (subtle) ways to control these machines next to the traditional, IP-based control structure of desktop malware. In addition, many smartphones are always turned on in contrast to ordinary computers. Singh *et al.* evaluated Bluetooth as the main command-and-control infrastructure [59] and Zeng *et al.* focused on the short message service [60]. Liu *et al.* showed that even a very small percentage of remote controlled mobile devices may successfully perform a DDoS attack on 911 call-centers [58]. In general, the topic on botnets is out of scope of this paper, but comprehensive literature exists [61], [62].

DoS Attacks Against Mobile Devices: Since mobile devices are battery powered, a huge power consumption can rapidly lead to the depletion of its power source. This can be easily done by malware by using all available CPU cycles for (junk) computations. A far more severe way to disable the service of a mobile device is the deletion or corruption of essential data stored at difficult to reach locations such as the E^{2}PROM. Fixing these issues is complex and often even impossible for average users because repairing requires fundamental knowledge of the device and can therefore often only be done by the manufacturer himself.

B. SMS Vulnerabilities

An incident of the early times of mobile phones (not even smartphones at that time) was an implementation bug in the SMS parser of the Siemens S55: receiving a short message with Chinese characters lead to a Denial of Service (DoS) [63]. This bug required a local firmware update, forcing the users to bring or send their device to customer service. This class is expected to be of less importance in the future, because modern smartphone architectures are increasingly allowing local or remote firmware updates, cf. Section II-C2.

A recent DoS attack is the "curse of silence" short message, which was published in late 2008 [64]. It is caused by an omitted sanity check of input data. Nokia published a removal tool one month after the attack was made public.

C. MMS Vulnerabilities

In 2006, a remote code execution exploit for mobile phones using MMS as the attack vector was published by Mulliner [65]. It exploited a buffer overflow in the MMS handling program of Windows Mobile CE 4.2. Being the first of its kind, it supported the public fear of that time that mobile devices would start to become commonly attacked. The exploit received some attention by a technical audience and the MNOs, who published patches for affected devices. Anti-virus companies added the exploit to their signature databases, but the exploit never appeared as part of mobile malware and thus not much harm was caused by it.

There are two possible explanations for this. The first is the probability of succeeding with the message because it has to be guessed which memory slot is in use. A second and more probable explanation is the actual code base of the devices affected. In particular, Windows CE 4.2 was already succeeded by Windows CE 5 at the time when the exploit was published. Therefore, this vulnerability only affected comparatively outdated devices.

D. Mobile Web Browser

Mobile web browsers are an emerging attack vector for mobile devices. Just as common web browsers, mobile web browsers are extended from pure web browsing software to complete application frameworks with widgets or completely browser-based mobile devices We can expect that even security-relevant functions of the operating system will be accessible in the near future.

Industry requirements even include these security-relevant features. One example are the browser requirements of OMTP (Open Mobile Terminal Platform) [66]. More specifically, BR-2540 requires: "The browser MUST support the making of voice calls and video calls from a URI/IRI". BR-2570 suggests appropriate security mechanisms in the implementation of this requirement as follows: "The browser SHOULD ask for user confirmation before initiating any call from a hyperlink". Certain versions of the iPhone web browser complies with this requirement, but they enable browser-based dialers to create costs for the user without necessarily asking for confirmation.

Therefore, the mobile web browser as an application framework of its own is able to undermine the mobile device's security model: the original (and possibly secure) model of signed applications is replaced by the security model of the web browser developer. Examples for successful attacks—besides DoS attacks on the mobile Internet Explorer [67]—are the jailbreak of the iPhone, hacking the Android browser, and using the iPhone browser as a dialer.

E. Mobile Malicious Software

Investigating the damage potential of mobile malicious software is challenging today because this new kind of malware has the potential to undermine the trust of mobile phone users in their mobile telephony system as such. Therefore, we see the main research tasks for mobile device security in the attacks that can be committed by mobile malicious software. Here the mobile device can be seen as exhibiting arbitrary and possibly malicious behavior. We will first present preliminary work and then introduce malware detection mechanisms.

1) Surveys of Mobile Malware: This section provides in chronological order an overview of important surveys of mobile malware. Peikari presents an overview of Windows Mobile and Symbian OS malware [68]. An extensive article covering nearly all malware as of the time of its writing was given by Shevchenko [69]. A book by Eren and Detken lists known malware samples until 2006, surveys the weaknesses of mobile operating systems, and describes much of the mobile and the mobile device security knowledge of that time [70]. Tyssy and Helenius list infection routes and some examples of malware of the year 2006, but their focus is on countermeasures and media perception of mobile malware [71]. Bontchev notes mobile malware classification problems and chooses Symbian OS malware as an example [72]. Although not explicitly stated, his findings can be generalized for malicious software on any operating system. A survey of mobile malware is presented by Hypponen [5]. Besides a summary of mobile device security knowledge of that time, it shows in an illustrative comic cartoon that many repetitions of an installation attempt (via Bluetooth) could even break down the resistance of a security-conscious user. McAfee published a study in 2007 as a result of surveying mobile network operators [73]. This survey shows how MNOs start preparing defenses against mobile malware. The most recent work on this topic as of the time of writing is by Oberheide and Jahanian [8].

2) Malware Detection on Smartphones: Malware detection on smartphones is a difficult task. Although in principle not different from malware detection on desktop computers, the limited processing power of such devices poses a huge challenge. We already outlined this in Section II-C3 and now discuss several different malware detection strategies.

Signature Based Detection: This is the classic approach when a malware is identified and its characteristics are known. A signature may be generated and can thereafter be used to detect this special type. Classical AV software is signature based and works exactly this way on almost all computers. However, we cannot simply port this approach to smartphones. The main reason is that the matching algorithm must be regularly active to scan all processes for suspicious code. Obviously, this puts a heavy burden on the CPU and might even be noticeable by the user (*e.g.*, unresponsive graphical interface or a faster battery exhaustion). To avoid this, Oberheide *et al.* presented a virus scanner for mobile devices which offloads the actual scanning to the cloud [74]. Furthermore, the experience on desktop computers shows that signature based approaches are doomed to fail given the large number of newly emerging threats.

Next to the "classical signatures" for AV scanners, *static function call analysis* may provide clues about the intents of the corresponding program. This is typically done once at installation time for new programs. The used function calls may be classified and, if necessary, appropriate actions can be taken. This has been tested for the Android [75] and the Symbian platform [76].

A proactive way to detect malware before it even gets the chance to perform its malice intent is the way how Apple's *App Store* application vetting works. Each application that is uploaded by its developers is checked before it can be downloaded. Albeit the performed checks are unknown, its aim is to detect malicious code and to withhold the application if something suspicious is found. Since it is hard to detect malicious code hidden somewhere deep in the code path, some unwanted software slips through this mechanism from time to time [15], [77].

Anomaly Detection: In contrast to signature based detection approaches, anomaly detection techniques attempt to detect malware with unknown behavior. One example is *SmartSiren*, a tool developed by Cheng *et al.* [78]. SmartSiren is able to detect unknown malware based on its communication behavior through Bluetooth and SMS. Privacy conform data about these communication media is sent to a central proxy which attempts to detect abnormal behavior. This approach is able to detect fast spreading worms and "slow working" malware, which collects private data and forwards it from time to time in aggregated form.

Another example is the infrastructure presented by Portokalidis *et al.* [79]. Here a "tracer" runs on the mobile device and records all necessary information which is needed by a "replayer" to playback the instruction trace on an external virtual machine which represents a replica of the mobile device. This approach also offloads expensive computation to much more powerful computers in the cloud. This way, off-site virus detection routines may be applied in addition to complex, dynamic taint analysis, which may detect events such as buffer overflows or foreign code execution.

A completely different approach is evaluated by Liu *et al.* [80] and Kim *et al.* [81]. Since mobile devices have comparatively small batteries, malware should be detectable by the amount of battery power consumed by their conducted instructions. If the running applications, the user behavior, and the state of the battery is well known and precisely defined, additional hidden (malicious) activity can be detected. However, it is unclear and an open research question how well malware can be detected on smartphones that is in daily use, especially with continuously changing user behavior.

Rootkit Detection: Malware with high privileges may attempt to hide itself at kernel level. The rootkit techniques do not differ from ordinary computers and, hence, their detection is to a certain extent identical—and therefore very hard. A first rootkit for Android has already been presented [82] and Bickford *et al.* evaluated rootkit detection on mobile devices [83]. It is an open question how rootkits on smartphones can be detected effectively and efficiently.

Software-based Attestation: Jakobsson and Johansson describe an approach to retroactively detect any active software based on *memory printing* [84]. The idea is to use light-weight cryptographic constructions with the property that it takes notably longer to compute a given function when the performing algorithm is given less usable RAM than for which it was configured. This approach is suited to detect software that wants to hide its presence on mobile devices. Active malware, whose memory is swapped out to much slower (Flash) memory during execution, will experience a huge latency penalty which is measurable. This makes active malware detectable through either observed memory changes or especially due to timing discrepancies when the malware attempts to evade detection during attestation by computing the expected result for the attestation.

F. Operating System Protection

Because smartphones deal with broader and broader application domains, their operating system and their programs become comparable to desktop computers. Both systems share the same architecture and in many cases even the exact same technologies, *e.g.*, the operating system or web browser back end. Thus, their security can be tightened with the same technologies. We now focus on some ways to enhance the security of smartphone operating systems.

Limited privileges and process isolation: Exploited applications may run foreign code within the boundaries of their given privileges. Higher privileges endanger the whole system and are often not necessary for the vast majority of applications. Smartphone applications should be run with the same principles in mind as their counterparts in ordinary computers. A good example is the Android platform, where applications run with different UIDs and are further separated through their own JVMs. Virtualization in general may enhance the overall security of smartphones (*e.g.*, separated VMs for private and work-related tasks), but currently there is no (hardware) support for this yet.

Hardened Kernels: Ordinary computer operating system kernels and utilities constantly raise the stakes to execute foreign code through critical security bugs. These techniques should also be used at the heart of smartphone operating systems and include *Address Space Layout Randomization*, *stack protection* and *non-executable writable memory*. *Mandatory Access Control* lists may further enhance overall smartphone security. It is ongoing work to enable all these techniques on the different platforms.

Sane Default Settings: Smartphone services and applications should have sound default configurations and should only run when their services are required. One such example is Bluetooth connectivity. Another is shown by Habib *et al.* [85], where some Symbian based smartphones are prone to DoS attacks in their default configuration via a network service that is reachable by default. To our knowledge, such an evaluation has not been done except for Windows Mobile and Symbian platforms.

Updates: The more people work in the area of smartphone security, the more likely it is that (security) bugs will be found. In order to close the corresponding emerging security holes, applications and operating systems need to be updated. This has to been done in an easy and straightforward fashion, as common users are not interested nor motivated in long update procedures because of their missing security understanding, cf. Section VII. Better update procedures are an open research question. A challenge for future offensive research can be the abuse of firmware flashing functionality, which leads to more subtle attacks. However, note that this kind of attacks might be detected successfully.

Software Attestation: A feature of smartphones is the ability to install all kinds of (closed source) 3rd-party software. Those applications may contain unwanted routines which are very hard to detect. One example is the previously mentioned private data stealing routine in a game. The Android OS allows to see which capabilities (Internet-access, send/receive SMS, ...) an application requests during install-time and to eventually deny them. Sadly, this is an all-or-nothing decision. Either the application is installed and may anytime make use of the granted capabilities, or it is not installed and therefore not usable. But even an unsuspicious looking weather application which requests forecasts over the Internet connection and which might automatically retrieve updates based on the current location (which is known through the embedded GPS sensor) is perfect to spy on the user. In order to detect such unwanted behavior, some research has been done on this topic. *Kirin* [86] is a framework for Android which decides on the basis of the stakeholders' *policy invariants* which application requests are granted or denied. Ongtang *et al.* developed *SAINT*, a modified infrastructure for Android which assigns permissions during install-time for run-time access to or communication between applications [87]. Another proposed Android security tool is *SCanDroid* [88], which may automatically detect unwanted information flows in applications based on the requested capabilities. The downside of this approach is that the source code is required and that it is solely for analysis purposes as no intervention is possible.

Complementary approaches are *TaintDroid* [89] and *PiOS* [90]. Both systems perform dynamic taint analysis and are able to reveal hidden and possible unwanted information flows of private data. These tools do not need the source code for their operation, but work on the binary level. A related issue is the ongoing challenge of application revocation of mobile device applications [15].

G. Limited Graphical User Interface

We will now look into two challenges concerning the user interface of mobile devices. First, it is possible that the user interface does not display the message that the program or the operating system intended to. Examples are APIs for dialog boxes that accept strings of arbitrary length for the message to be displayed. Niu *et al.* presented some phishing techniques based on these inadequatenesses [91]. Second, and even more challenging, is malware that is able to simulate user actions, *e.g.*, by automatically reacting to security confirmations. Some desktop computer operating systems provide APIs for this task and it can be imagined that mobile operating systems will provide this functionality in the future. It becomes even more intriguing in combination with malware that rewrites some parts of the OS.

A first solution to these problems is the introduction of Turing tests (CAPTCHAs) for every security-relevant event on the mobile device in order to prove that an event was confirmed by a human user. The task for future research could be to explore the portion of security problems that remain when this solution is applied. An additional question lies in the area of human computer interaction (HCI) design: enough to be of use, but not so much so that it becomes a burden.

VII. The User as Attack Vector

Many studies have been performed to evaluate the security knowledge of the average user. Most of them show what already the well-known study of Whitten and Tygar [92] found out: normal users are not able to use security mechanisms correctly. Several attempts have been made to simplify the security interface for users, but even the very simple Windows security slider with only *four* possible positions was not completely understood and therefore wrongly set by users who rated themselves as *IT proficient* [93].

Therefore, we need to ask ourselves the purpose of these numerous security mechanisms if the user does not understand them. And even if he understands to work with one specific mechanism, another mechanism might be difficult to grasp for him. People working in security do want to achieve the desirable goal of more security-aware users, but for the vast majority of users, their will, their interest, and in particular the time they can devote to learn more than one security mechanism, is likely to be limited.

Security and usability started out of general research on HCI. We can trace it back to the famous study by Whitten and Tygar in 1999 [92] and observe that it gained attention in subsequent years [94], [95]. User studies regularly show that security mechanisms are neither understood nor correctly used by most of their users [92], [93], [96], [97]. In addition, some authors propose to embed security in products [98] and in the development process [99] rather than having it stand-alone. Usability heuristics have been developed by Nielsen [100] and Shneiderman and Plaisant [101]. They are a good starting point for usability of security solutions.

Making our scope a bit broader, we also have to see the administrator or security professional as a possible attack vector: new and changing environments, poorly documented hard- and software, and interplay of components which were not developed to work together, offers very interesting—but also challenging—questions. Wrong decisions on the security professionals side will have very drastic consequences. It would therefore be very helpful for them to have access to some guidelines specific to mobile security, even going down to the level of comparing currently available solutions (both open and closed source). To the best of our knowledge, there is no such tool yet, leading to another open research question.

VIII. Conclusion

We recall the mobile device security specifics introduced earlier, which form the framework for the investigations in this paper (cf. Section III). Important for future research is knowing whether these specifics remain valid. We see the following directions, which directly influence our results:

- *Creation of costs* is a topic that will be even more important in the future. As more and more services are introduced to mobile devices, it is likely that some of them will be payment services which might be abused.
- The *network environment* is likely to remain unchanged. Using observations of recent years, there is a tendency towards an increased remote device management and remote firmware update. This mitigates the users' behavior (cf. Section VII).
- The importance of the *expensive wireless link* will decrease. From a monetary perspective, communication costs will decrease because more bandwidth becomes available in mobile networks. From a computational side, costs for algorithms on the device will also decrease because the fourth generation of mobile networks (4G) is designed for high bandwidth and low latency for data traffic. Hence, having more bandwidth, lower latency, and faster data transfer can be used for new security solutions.
- The *limited device resources* are likely to decrease, leading to more processing power and more memory. Still, the battery seems to stay the most limiting factor in mobile devices in the near future. Altogether, *limited device resources* remain a unique specific of mobile device security.
- The *security-unaware user* might become a little more security-aware when mobile device security moves into broad attention. This is likely to be connected to *reputation*, as more understanding for mobile device security might relieve the MNOs from unsubstantiated claims from their customer base.
- *Heterogeneity* in devices, operating systems, and applications was a trait of mobile security in the past. It is difficult to predict if monopolies like in the desktop world will emerge also in the mobile world. However, as we start off with a high diversity this might mitigate the security risks outlined above.

In summary, smartphones are rapidly closing the gap to ordinary computers in terms of processing power, display size, and versatility of operating systems. However, they have inherent constraints that will remain valid in the future. There is much evidence, that indeed we are at the beginning of an era of attacks against smartphones. In any case, research in mobile security will be an interesting area in the years to come.

Acknowledgments: This work has been supported by the Ministry of Economic Affairs and Energy of the State of North Rhine-Westphalia (grant 315-43-02/2-005-WFBO-009), the Federal Ministry of Education and Research (grant 01BY1020 – MobWorm), and the DFG (Emmy Noether grant *Long Term Security*). We also thank the anonymous reviewers for their valuable insights and comments.

References

[1] N. Leavitt, "Malicious Code Moves to Mobile Devices," *IEEE Computer*, vol. 33, no. 12, 2000.

[2] S. N. Foley and R. Dumigan, "Are Handheld Viruses a Significant Threat?" *Commun. ACM*, vol. 44, no. 1, 2001.

[3] D. Dagon *et al.*, "Mobile Phones as Computing Devices: The Viruses are Coming!" *IEEE Pervasive Computing*, vol. 3, no. 4, 2004.

[4] N. Leavitt, "Mobile Phones: The Next Frontier for Hackers?" *IEEE Computer*, vol. 38, no. 4, 2005.

[5] M. Hypponen, "State of Cell Phone Malware in 2007," 2007, http://www.usenix.org/events/sec07/tech/hypponen.pdf.

[6] J. Kleinberg, "The Wireless Epidemic," *Nature*, vol. 449, no. 20, Sep. 2007.

[7] G. Lawton, "Is It Finally Time to Worry about Mobile Malware?" *IEEE Computer*, vol. 41, no. 5, 2008.

[8] J. Oberheide and F. Jahanian, "When Mobile is Harder Than Fixed (and Vice Versa): Demystifying Security Challenges in Mobile Environments," in *Workshop on Mobile Computing Systems and Applications (HotMobile)*, February 2010.

[9] M. Kotadia, "Major Smartphone Worm 'by 2007'," Jun. 2005.

[10] V. Bontchev, "Virusability of Modern Mobile Environments," in *Virus Bulletin Conference*, Sep. 2007.

[11] Gartner Research, "Gartner Says Worldwide Mobile Phone Sales Grew 35 Percent in Third Quarter 2010; Smartphone Sales Increased 96 Percent," 2010, http://www.gartner.com/it/page.jsp?id=1466313.

[12] A. Portnoy, "Pwn2Own 2010," 2010, http://dvlabs.tippingpoint.com/blog/2010/02/15/pwn2own-2010.

[13] M. Keith, "Android 2.0-2.1 Reverse Shell Exploit," 2010, http://www.exploit-db.com/exploits/15423/.

[14] R.-P. Weinmann, "All Your Baseband Are Belong To Us," hack.lu, 2010, http://2010.hack.lu/archive/2010/Weinmann-All-Your-Baseband-Are-Belong-To-Us-slides.pdf.

[15] A. Greenberg, "Google pulls app that revealed Android flaw, issues fix," 2010, http://news.cnet.com/8301-27080_3-20022545-245.html.

[16] P. Zheng and L. M. Ni, "The Rise of the Smart Phone," *IEEE Distributed Systems Online*, vol. 7, no. 3, 2006.

[17] S. C. Guthery and M. J. Cronin, *Developing MMS Applications - Multimedia Messaging Services for Wireless Networks*. McGraw-Hill Professional, Jun. 2003.

[18] D. Barroso, “ZeuS Mitmo: Man-in-the-mobile,” September 2010, http://securityblog.s21sec.com/2010/09/zeus-mitmo-man-in-mobile-i.html.

[19] OMTP Limited, “Advanced Device Management,” Jan. 2008.

[20] ——, “Mobile Application Security - Requirements for Mobile Applications Signing Schemes - Version 1.23,” Dec. 2006.

[21] M. Becher, “Security of smartphones at the dawn of their ubiquitousness,” Ph.D. dissertation, University of Mannheim, Oct. 2009.

[22] Bladox, s.r.o., “Turbo SIM,” http://www.bladox.com/.

[23] J. Berka, “Turbo SIM add-on allows full iPhone unlocking,” Aug. 2007, http://arstechnica.com/apple/news/2007/08/turbo-sim-add-on-allows-full-iphone-unlocking.ars.

[24] OsmocomBB project, “SIMtrace,” http://bb.osmocom.org/trac/wiki/SIMtrace.

[25] OMTP Limited, “Advanced Trusted Environment - OMTP TR1,” May 2008.

[26] University of Glamorgan, “Disk Study 2008-2009,” May 2009, http://isrg.weblog.glam.ac.uk/2009/5/7/disk-study-2008-2009.

[27] F. C. Freiling *et al.*, “Reconstructing people’s lives: A case study in teaching forensic computing,” in *Conference on IT-Incidents Management & IT-Forensics - IMF*, 2008.

[28] E. Barkan *et al.*, “Instant Ciphertext-Only Cryptanalysis of GSM Encrypted Communication,” *J. Cryptology*, vol. 21, no. 3, 2008.

[29] J. Frick and R. Bott, “Method for identifying a mobile phone user or for eavesdropping on outgoing calls,” http://v3.espacenet.com/publicationDetails/biblio?CC=EP&NR=1051053&KC=&FT=E.

[30] OsmocomBB project, “OpenBSC,” http://openbsc.osmocom.org/trac/.

[31] OpenBTS, “GSM. Simplified.” http://openbts.sf.net/.

[32] E. Biham *et al.*, “A Related-Key Rectangle Attack on the Full KASUMI,” in *ASIACRYPT*, 2005.

[33] W. Enck *et al.*, “Exploiting Open Functionality in SMS-capable Cellular Networks,” in *ACM Conference on Computer and Communications Security (CCS)*, 2005.

[34] J. Serror *et al.*, “Impact of Paging Channel Overloads or Attacks on a Cellular Network,” in *ACM Workshop on Wireless Security (WiSe)*, 2006.

[35] P. Traynor *et al.*, “Mitigating Attacks on Open Functionality in SMS-capable Cellular Networks,” *IEEE/ACM Trans. Netw.*, vol. 17, no. 1, 2009.

[36] R. Racic *et al.*, “Exploiting MMS Vulnerabilities to Stealthily Exhaust Mobile Phone’s Battery,” in *Security and Privacy in Comm. Networks (SecureComm)*, 2006.

[37] 3GPP, “3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; 3G security; Security principles and objectives (Release 4),” 3rd Generation Partnership Project (3GPP), Tech. Rep., Mar. 2001.

[38] M. Hassan *et al.*, “Comprehensive Analysis of UMTS Authentication and Key Agreement,” *International Journal of Computer and Network Security*, vol. 2, no. 2, 2010.

[39] S. Pütz, R. Schmitz, and T. Martin, “Security Mechanisms in UMTS,” *Datenschutz und Datensicherheit*, vol. 25, no. 6, 2001.

[40] U. Meyer and S. Wetzel, “A Man-in-the-Middle Attack on UMTS,” in *ACM Workshop on Wireless Security (WiSe)*, 2004.

[41] P. P. C. Lee *et al.*, “On the Detection of Signaling DoS Attacks on 3G/WiMax Wireless Networks,” *Comput. Netw.*, vol. 53, no. 15, 2009.

[42] B. Zhao *et al.*, “A Chain Reaction DoS Attack on 3G Networks: Analysis and Defenses,” in *IEEE Conference on Computer Communications (INFOCOM)*, 2009.

[43] P. C. Kocher *et al.*, “Differential Power Analysis,” in *CRYPTO*, ser. Lecture Notes in Computer Science, M. J. Wiener, Ed., vol. 1666. Springer, 1999.

[44] J. R. Rao *et al.*, “Partitioning Attacks: Or How to Rapidly Clone Some GSM Cards,” in *IEEE Symposium on Security and Privacy*, 2002.

[45] E. Mills, “Leaking crypto keys from mobile devices,” http://news.cnet.com/8301-27080_3-10379115-245.html, 2009.

[46] G. Bertoni *et al.*, “AES Power Attack Based on Induced Cache Miss and Countermeasure,” in *Conference on Information Technology: Coding and Computing (ITCC)*, 2005.

[47] B. Krebs, “Paris Hilton Hack Started With Old-Fashioned Con,” May 2005, http://www.washingtonpost.com/wp-dyn/content/article/2005/05/19/AR2005051900711.html.

[48] P. Traynor *et al.*, “On Cellular Botnets: Measuring the Impact of Malicious Devices on a Cellular Network Core,” in *ACM Conference on Computer and Communications Security (CCS)*, 2009.

[49] C. Xenakis, “Malicious Actions against the GPRS Technology.” *Journal in Computer Virology*, vol. 2, no. 2, 2006.

[50] F-Secure, “Bluetooth-Worm:SymbOS/Cabir,” ”http://www.f-secure.com/v-descs/cabir.shtml”.

[51] Symantec Security Response, “AndroidOS.Tapsnake: Watching Your Every Move,” 2010, http://www.symantec.com/connect/blogs/androidostapsnake-watching-your-every-move.

[52] N. Seriot, “SpyPhone,” 2010, ”https://github.com/nst/spyphone/”.

[53] R. Schlegel *et al.*, “Soundminer: A Stealthy and Context-Aware Sound Trojan for Smartphones,” in *Network and Distributed System Security Symposium (NDSS)*, Feb. 2011.

[54] J. Franklin *et al.*, “An Inquiry into the Nature and Causes of the Wealth of Internet Miscreants,” in *ACM Conference on Computer and Communications Security (CCS)*, 2007.

[55] R. Thomas and J. Martin, “The Underground Economy: Priceless,” *USENIX ;login:*, vol. 31, no. 6, Dec 2006.

[56] T. Holz *et al.*, “Learning More About the Underground Economy: a Case-Study of Keyloggers and Dropzones,” in *European Conference on Research in Computer Security (ESORICS)*, 2009.

[57] Kaspersky Lab, “First SMS Trojan Detected for Smartphones running Android,” 2010, ”http://www.kaspersky.com/news?id=207576156”.

[58] L. Liu *et al.*, “Exploitation and Threat Analysis of Open Mobile Devices,” in *ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS)*, 2009.

[59] K. Singh *et al.*, “Evaluating Bluetooth as a Medium for Botnet Command and Control,” in *Conference on Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA)*, 2010.

[60] Y. Zeng *et al.*, “Design of SMS Commanded-and-Controlled and P2P-Structured Mobile Botnets,” University of Michigan, Tech. Rep. CSE-TR-562-10, 2010.

[61] M. Abu Rajab *et al.*, “A Multifaceted Approach to Understanding the Botnet Phenomenon,” in *ACM SIGCOMM Conference on Internet Measurement (IMC)*, 2006.

[62] F. C. Freiling *et al.*, “Botnet Tracking: Exploring a Root-Cause Methodology to Prevent Distributed Denial-of-Service Attacks,” in *European Conference on Research in Computer Security (ESORICS)*, 2005.

[63] Bugtraq, “Siemens M Series SMS DoS Vulnerability,” Mar. 2003, http://www.securityfocus.com/bid/7004/.

[64] T. Engel, “Remote SMS/MMS Denial of Service - ”Curse Of Silence” for Nokia S60 phones,” Nov. 2008, http://berlin.ccc.de/~tobias/cos/s60-curse-of-silence-advisory.txt.

[65] C. Mulliner and G. Vigna, “Vulnerability Analysis of MMS User Agents,” in *Computer Security Applications Conference (ACSAC)*, 2006.

[66] OMTP Limited, “Browser,” Oct. 2007.

[67] CVE-2007-0685, “Denial of Service for Internet Explorer on Windows Mobile 5.0 and Windows Mobile 2003 and 2003SE for Smartphones and PocketPC,” Feb. 2007, http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2007-0685.

[68] C. Peikari, “PDA Attacks, Part 2: Airborne Viruses – Evolution of the Latest Threats,” *(IN)SECURE Magazine*, vol. 4, Oct. 2005.

[69] A. Shevchenko, “An Overview of Mobile Device Security,” Sep. 2005, http://www.viruslist.com/en/analysis?pubid=170773606.

[70] E. Eren and K.-O. Detken, *Mobile Security*. Hanser, 2006.

[71] S. Töyssy and M. Helenius, “About Malicious Software in Smartphones.” *Journal in Computer Virology*, vol. 2, no. 2, 2006.

[72] V. Bontchev, “SymbOS Malware Classification Problems,” in *Virus Bulletin Conference*, Aug. 2006.

[73] McAfee, “Mobile Security Report,” 2008, http://www.mcafee.com/us/local_content/reports/mcafee_mobile_security_report_2008.pdf.

[74] J. Oberheide *et al.*, “Virtualized In-Cloud Security Services for Mobile Devices,” in *Workshop on Virtualization in Mobile Computing (MobiVirt)*, 2008.

[75] A.-D. Schmidt *et al.*, “Static Analysis of Executables for Collaborative Malware Detection on Android,” in *IEEE International Conference on Communications (ICC)*, 2009.

[76] ——, “Detecting Symbian OS Malware through Static Function Call Analysis,” in *International Conference on Malicious and Unwanted Software (Malware)*, 2009.

[77] J. Diaz, “How a 15-yo Kid Tricked Apple With a Disguised iPhone Tethering App,” 2010, http://gizmodo.com/5592521/how-a-guy-tricked-apple-with-a-disguised-iphone-tethering-ap.

[78] J. Cheng *et al.*, “SmartSiren: Virus Detection and Alert for Smartphones,” in *International Conference on Mobile Systems, Applications and Services (MobiSys)*, 2007.

[79] G. Portokalidis *et al.*, “Paranoid Android: Versatile Protection For Smartphones,” in *Annual Computer Security Applications Conference (ACSAC)*, 2010.

[80] L. Liu *et al.*, “VirusMeter: Preventing Your Cellphone from Spies,” in *International Symposium on Recent Advances in Intrusion Detection (RAID)*, 2009.

[81] H. Kim *et al.*, “Detecting Energy-Greedy Anomalies and Mobile Malware Variants,” in *International Conference on Mobile Systems, Applications, and Services (MobiSys)*, 2008.

[82] N. J. Percoco and C. Papathanasiou, “This is not the Droid you’re looking for...” in *Defcon 18*, Aug. 2010.

[83] J. Bickford *et al.*, “Rootkits on Smart Phones: Attacks, Implications and Opportunities,” in *Workshop on Mobile Computing Sys. and Appl. (HotMobile’10)*. ACM, 2010.

[84] M. Jakobsson and K.-A. Johansson, “Retroactive Detection of Malware With Applications to Mobile Platforms,” in *USENIX Workshop on Hot Topics in Security (HotSec)*, 2010.

[85] S. M. Habib *et al.*, “An Analysis of the Robustness and Stability of the Network Stack in Symbian-based Smartphones,” *Journal of Networks*, vol. 4, no. 10, 2009.

[86] W. Enck *et al.*, “On Lightweight Mobile Phone Application Certification,” in *ACM Conference on Computer and Communications Security (CCS)*, 2009.

[87] M. Ongtang *et al.*, “Semantically Rich Application-Centric Security in Android,” in *Annual Computer Security Applications Conference (ACSAC)*, 2009.

[88] A. P. Fuchs *et al.*, “SCanDroid: Automated Security Certification of Android Applications,” 2010.

[89] W. Enck *et al.*, “TaintDroid: An Information-flow Tracking System for Realtime Privacy Monitoring on Smartphones,” in *USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, 2010.

[90] M. Egele *et al.*, “PiOS: Detecting Privacy Leaks in iOS Applications,” in *Network and Distributed System Security Symposium (NDSS)*, Feb. 2011.

[91] Y. Niu *et al.*, “iPhish: Phishing Vulnerabilities on Consumer Electronics,” in *USENIX Workshop on Usability, Psychology, and Security*, 2008.

[92] A. Whitten and J. D. Tygar, “Why Johnny Can’t Encrypt: A Usability Evaluation of PGP 5.0,” in *USENIX Security Symposium*, 1999.

[93] S. Furnell *et al.*, “The Challenges of Understanding and Using Security: A Survey of End-Users,” *Computers & Security*, vol. 25, no. 1, 2006.

[94] L. Cranor and S. Garfinkel, *Security and Usability: Designing Secure Systems That People Can Use*. O’Reilly Media, 2005.

[95] S. L. Garfinkel, “Design Principles and Patterns for Computer Systems that are Simultaneously Secure and Usable,” Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, MA, USA, 2005.

[96] S. Furnell, “Why Users Cannot Use Security,” *Computers & Security*, vol. 24, no. 4, 2005.

[97] ——, “Making Security Usable: Are Things Improving?” *Computers & Security*, vol. 26, no. 6, 2007.

[98] P. Kuper, “The State of Security,” *IEEE Security & Privacy*, vol. 3, no. 5, 2005.

[99] S. Görling, “The Myth of User Education,” in *Virus Bulletin Conference*, Oct. 2006.

[100] J. Nielsen, “Ten Usability Heuristics,” 2005, http://www.useit.com/papers/heuristic/heuristic_list.html.

[101] B. Shneiderman and C. Plaisant, *Designing the User Interface: Strategies for Effective Human-Computer Interaction*, 4th ed. Pearson Addison Wesley, 2004.