Intrusion Detection and Prevention for Mobile Ecosystems

Edited by
Georgios Kambourakis • Asaf Shabtai
Constantinos Kolias • Dimitrios Damopoulos

CRC Press
Taylor & Francis Group

CRC Series in Security, Privacy and Trust

Intrusion Detection and Prevention for Mobile Ecosystems

CRC Series in Security, Privacy and Trust

AIMS AND SCOPE

This book series presents the advancements in research and technology development in the area of security, privacy, and trust in a systematic and comprehensive manner. The series will provide a reference for defining, reasoning and addressing the security and privacy risks and vulnerabilities in all the IT systems and applications, it will mainly include (but not limited to) aspects below:

- Applied Cryptography, Key Management/Recovery, Data and Application Security and Privacy;
- Biometrics, Authentication, Authorization, and Identity Management;
- Cloud Security, Distributed Systems Security, Smart Grid Security, CPS and IoT Security;
- Data Security, Web Security, Network Security, Mobile and Wireless Security;
- Privacy Enhancing Technology, Privacy and Anonymity, Trusted and Trustworthy Computing;
- Risk Evaluation and Security Certification, Critical Infrastructure Protection;
- Security Protocols and Intelligence, Intrusion Detection and Prevention;
- Multimedia Security, Software Security, System Security, Trust Model and Management;
- Security, Privacy, and Trust in Cloud Environments, Mobile Systems, Social Networks, Peer-to-Peer Systems, Pervasive/Ubiquitous Computing, Data Outsourcing, and Crowdsourcing, etc.

PUBLISHED TITLES

Intrusion Detection and Prevention for Mobile Ecosystems
Georgios Kambourakis, Asaf Shabtai, Constantinos Kolias, and Dimitrios Damopoulos

Touchless Fingerprint Biometrics
Ruggero Donida Labati, Vincenzo Piuri, and Fabio Scotti

Empirical Research for Software Security: Foundations and Experience
Lotfi ben Othmane, Martin Gilje Jaatun, and Edgar Weippl

Real-World Electronic Voting: Design, Analysis and Deployment
Feng Hao and Peter Y. A. Ryan

Protecting Mobile Networks and Devices: Challenges and Solutions
Weizhi Meng, Xiapu Luo, Steven Furnell, and Jianying Zhou

Location Privacy in Wireless Sensor Networks
Ruben Rios, Javier Lopez, and Jorge Cuellar

Intrusion Detection and Prevention for Mobile Ecosystems

Edited by
Georgios Kambourakis, Asaf Shabtai, Constantinos Kolias, and Dimitrios Damopoulos

CRC Press
Taylor & Francis Group
Boca Raton London New York

CRC Press is an imprint of the
Taylor & Francis Group, an **informa** business

CRC Press
Taylor & Francis Group
6000 Broken Sound Parkway NW, Suite 300
Boca Raton, FL 33487-2742

Printed on acid-free paper

International Standard Book Number-13: 978-1-138-03357-3 (Hardback)

Contents

Editors

Georgios Kambourakis, PhD, earned his diploma in applied informatics from the Athens University of Economics and Business and PhD in information and communication systems engineering from the Department of Information and Communications Systems Engineering of the University of the Aegean. He also earned Master of Education (EdM) from the Hellenic Open University. Currently, Georgios is an associate professor at the Department of Information and Communication Systems Engineering, University of the Aegean, Greece, and the director of Info-Sec-Lab. His research interests are in the fields of mobile and wireless networks security and privacy, VoIP security, public key infrastructure, IoT security, DNS security, and security education and he has more than 120 refereed publications in the above areas. In the fall semester of 2017, he conducted research on IoT Security and Privacy in George Mason University, Fairfax County, Virginia. He has guest edited special issues of several journals, including *ACM/Springer Mobile Networks and Applications, Computer Standards & Interfaces, IEEE Computer Magazine, Information Sciences, and Computer Communications*. He has been involved in several national and EU funded R&D projects in the areas of Information and Communication Systems Security. He is a reviewer for a plethora of IEEE and other international journals and has served as a technical program committee member for more than 220 international conferences in security and networking. More info at: www.icsd.aegean.gr/gkamb.

Asaf Shabtai, PhD, is a senior lecturer (assistant professor) in the Department of Software and Information Systems Engineering at Ben-Gurion University (BGU) of the Negev. Asaf is also a senior researcher at the Telekom Innovation Laboratories at BGU. Asaf is an expert in information systems security and has led several large-scale projects and researches in this field. His main areas of interests are security of computer networks and smart mobile devices, malware detection, fraud detection and credit risk prediction, data leakage detection, analysis of encrypted traffic, security awareness, IoT security, development of security mechanisms for avionic systems, as well as the application of machine learning in the cyber security domain. Asaf has published over 70 refereed papers in leading journals and conferences. In addition, he has coauthored a book on information leakage detection and prevention. Asaf earned a BSc in mathematics and computer science (1998); BSc in information

systems engineering (1998); MSc in information systems engineering (2003); and PhD in information systems engineering (2011), all from Ben-Gurion University.

Constantinos Kolias, PhD, is a research assistant professor at the George Mason University, Virginia. He earned his diploma in computer science from the Technological Educational Institute of Athens, Greece. He also earned a MSc and PhD in information and communication system security, both from the Department of Information and Communication Systems Engineering, University of the Aegean, Greece. He is the creator of the AWID dataset, the first dataset benchmark for intrusion detection in wireless networks. Also, he is the lead engineer in the creation of the first Internet-of-Things laboratory at the National Institute of Standards and Technology (NIST). His primary research interests lie in the field of security for Internet of Things, next-generation wireless communication protocols, intrusion detection for wireless networks, and mobile surveillance.

Dimitrios Damopoulos, PhD, earned his BSc in industrial informatics from the Technological Educational Institute of Kavala, Greece. He also earned MSc in information & communication systems security and PhD in information and communication systems engineering, both from the Department of Information and Communication Systems Engineering, University of the Aegean, Greece. Currently, he is with Stevens Institute of Technology as an assistant professor. His research interests focus on smartphone security, mobile device intrusion detection and prevention systems, mobile malware, as well as mobile applications and services.

Contributors

Razan Abdulhammed
Department of Computer Science and Engineering
University of Bridgeport
Bridgeport, Connecticut

Abdullah J. Alzahrani
College of Computer Science and Engineering (CCSE)
University of Hail
Hail, Saudi Arabia

Ioannis Askoxylakis
Institute of Computer Science
Foundation for Research and Technology—Hellas
Heraklion, Crete, Greece

Konstantia Barmpatsalou
CISUC/DEI
University of Coimbra
Coimbra, Portugal

Thierry Benoist
European Commission
Joint Research Centre
Ispra, Italy

Laurent Beslay
European Commission
Joint Research Centre
Ispra, Italy

Gerardo Canfora
Department of Engineering
University of Sannio
Benevento, Italy

Ana R. Cavalli
Telecom SudParis and Montimage
France

Andrea Ciardulli
European Commission
Joint Research Centre
Ispra, Italy

Tiago Cruz
CISUC/DEI
University of Coimbra
Coimbra, Portugal

Eleni Darra
Department of Digital Systems
University of Piraeus
Piraeus, Greece

Khaled Elleithy
Department of Computer Science and Engineering
University of Bridgeport
Bridgeport, Connecticut

Yuval Elovici
Department of Software and Information Systems Engineering
Telekom Innovation Laboratories at Ben-Gurion University
Beer-Sheva, Israel

William Enck
Department of Computer Science
North Carolina State University
Raleigh, North Carolina

Miad Faezipour
Department of Computer Science and Engineering
University of Bridgeport
Bridgeport, Connecticut

Konstantinos Fysarakis
Institute of Computer Science
Foundation for Research and Technology—Hellas
Heraklion, Crete, Greece

Dimitris Geneiatakis
Cyber and Digital Citizens' Security Unit
European Commission
Joint Research Centre
Ispra, Italy

Luigi Gentile
Koine srl
Benevento, Italy

Ali A. Ghorbani
Canadian Institute for Cybersecurity
University of New Brunswick
Fredericton, New Brunswick, Canada

Mordechai Guri
Department of Software and Information Systems Engineering
Cyber-Security Research Center
Ben-Gurion University of the Negev
Beer-Sheva, Israel

Vinh Hoa La
Telecom SudParis and Montimage
France

Sotiris Ioannidis
Institute of Computer Science
Foundation for Research and Technology—Hellas
Heraklion, Crete, Greece

Roger Piqueras Jover
Bloomberg LP
New York, NY

Vasilios Katos
Department of Computing
Bournemouth University
Poole, Dorset, United Kingdom

Sokratis K. Katsikas
Department of Digital Systems
University of Piraeus
Piraeus, Greece

and

Center for Cyber and Information Security
Norwegian University of Science and Technology
Norway

Ioannis Kounelis
Cyber and Digital Citizens' Security Unit
European Commission
Joint Research Centre
Ispra, Italy

Lam-For Kwok
Department of Computer Science
City University of Hong Kong
Hong Kong

Apostolos Malatras
European Commission
Joint Research Centre
Ispra, Italy

Louis Marinos
European Union Agency for Network and Information Security (ENISA)
Heraklion, Crete, Greece

Fabio Martinelli
Istituto di Informatica e Telematica
Consiglio Nazionale delle Ricerche
Pisa, Italy

Ivan Martinovic
Department of Computer Science
University of Oxford
Oxford, United Kingdom

Charalabos Medentzidis
Electrical and Computer Engineering Department
Aristotle University of Thessaloniki
Thessaloniki, Greece

Weizhi Meng
Department of Applied Mathematics and Computer Science
Technical University of Denmark
Copenhagen, Denmark

Francesco Mercaldo
Institute for Informatics and Telematics
National Research Council of Italy (CNR)
Pisa, Italy

Edmundo Monteiro
CISUC/DEI
University of Coimbra
Coimbra, Portugal

Adwait Nadkarni
Department of Computer Science
North Carolina State University
Raleigh, North Carolina

Igor Nai Fovino
Cyber and Digital Citizens' Security Unit
European Commission
Joint Research Centre
Ispra, Italy

Yuri Poliak
Department of Software and Information Systems Engineering
Cyber-Security Research Center
Ben-Gurion University of the Negev
Beer-Sheva, Israel

Ignacio Sanchez
European Commission
Joint Research Centre
Ispra, Italy

Andrea Saracino
Istituto di Informatica e Telematica
Consiglio Nazionale delle Ricerche
Pisa, Italy

Asaf Shabtai
Department of Software and Information Systems Engineering
Ben-Gurion University of the Negev
Beer-Sheva, Israel

Bracha Shapira
Department of Software and Information Systems Engineering
Telekom Innovation Laboratories at Ben-Gurion University
Beer-Sheva, Israel

Nava Sherman
Department of Software and Information Systems Engineering
Ben-Gurion University of the Negev
Beer-Sheva, Israel

Paulo Simoes
CISUC/DEI
University of Coimbra
Coimbra, Portugal

Yannis Soupionis
European Commission
Joint Research Centre
Ispra, Italy

Gary Steri
Cyber and Digital Citizens' Security Unit
European Commission
Joint Research Centre (JRC)
Ispra, Italy

Vincent F. Taylor
Department of Computer Science
University of Oxford
Oxford, United Kingdom

Vasant Tendulkar
Department of Computer Science
North Carolina State University
Raleigh, North Carolina

Akash Verma
Department of Computer Science
North Carolina State University
Raleigh, North Carolina

Corrado Aaron Visaggio
Department of Engineering
University of Sannio
Benevento, Italy

Jianying Zhou
Information Systems Technology and Design
Singapore University of Technology and Design
Singapore

Introduction

During the past few years, mobile devices such as smartphones and tablets have penetrated the market at a very high rate. From an end-user perspective, the unprecedented advantages these devices offer revolve not only around their high mobility, but also extend to their ease of use and the plethora of their applications. As the permeation of mobile devices increases, the development of mobile apps follows this frantic rate, by being built in great numbers on a daily basis. On the downside, this mushrooming of mobile networks and portable devices has attracted the interest of several kinds of aggressors who possess a plethora of invasion techniques in their artillery. Such ill-motivated entities systematically aim to steal or manipulate users' or network data, and even disrupt the operations provided to legitimate users.

Their goal is assisted by the fact that while a continuously increasing number of users has embraced mobile platforms and associated services, most of them are not security-savvy and usually follow naive privacy preservation practices on their routine interaction with their devices. Until now, a great mass of research work and practical experiences have alerted the community about the nature and severity of these threats that equally affect end-users, providers, and even organizations.

One can identify several more reasons behind the proliferation of malware and the spanning of novel invasion tactics in mobile ecosystems. First, mobile devices are used extensively for sensitive tasks, including bank transactions and e-payments, private interaction such as engagement in social media applications, or even mission critical processes in healthcare. Second, smart, ultraportable, and wearable devices such as smartwatches and smartglasses are highly personal, and thus can be correlated with a single user; they embed several sensors and functionalities capable of collecting many details about the context of users, while they are constantly connected to the Internet. Third, numerous researches and case studies have shown that despite the ongoing progress, native security mechanisms of modern mobile operating systems or platforms can be outflanked. Even worse, most of the applied wireless communication technologies are eventually proven to be prone to numerous attacks. Admittedly, under this mindset, the attack surface for evildoers grows, further augmenting the expansion of volume and sophistication of malware apps.

To cope with this situation, defenders need to deploy smarter and more advanced security measures along with legacy ones.

The book at hand comprises a number of state-of-the-art contributions from both scientists and practitioners working in intrusion detection and prevention for mobile networks, services, and devices. It aspires to provide a relevant reference for students, researchers, engineers, and professionals working in this particular area or those interested in grasping its diverse facets and exploring the latest advances in intrusion detection in mobile ecosystems. More specifically, the book consists of 16 contributions classified into 4 pivotal sections:

- *Mobile platforms security, privacy, and intrusion detection*: Introducing the topic of mobile platforms security, privacy, and intrusion detection, and offering related research efforts on attacking smartphone security and privacy, a way to create reliable smartphone end-user apps in an ad hoc manner, a privacy risk assessment for Android apps, and an inference system for mobile forensics.
- *Malware detection in mobile platforms*: Investigating advanced techniques for malware and rootkit detection in the Android platform, and exploring the different kinds of intrusive apps and data leakage due to malware in the same platform.
- *Mobile network security and intrusion detection*: Experimentally exploring mobile botnets, demonstrating ways for attacking LTE by applying low-cost software radio, and an intrusion detection framework based on SMS.
- *Intrusion detection in dynamic and self-organizing networks*: Focusing on intrusion techniques in self-organizing networks, wireless sensor networks, 6LoWPAN-based wireless sensor networks, and co-operative intelligent transportation systems.

I

MOBILE PLATFORMS SECURITY, PRIVACY, AND INTRUSION DETECTION

Taylor & Francis
Taylor & Francis Group
http://taylorandfrancis.com

Chapter 1

A Review of Intrusion Detection and Prevention on Mobile Devices: The Last Decade

Weizhi Meng, Jianying Zhou, and Lam-For Kwok

Contents

1.1 Introduction

Nowadays, mobile devices such as various phones are developing at a rapid pace and have become common in our daily lives. The worldwide smartphone market grew 0.7% year over year in 2016, with 344.7 million shipments, where Android dominated the market with an 87.6% share, according to data from the International Data Corporation (IDC) [1]. As a result, more users start utilizing mobile devices as a frequent storage medium for sensitive information (e.g., passwords, credit card numbers, private photos) [2], as well as use them for security-sensitive tasks due to their fast and convenient data connection [3]. Owing to this, smartphones have become an attractive target for hackers and malware writers [4,5].

As we know, mobile devices are easily lost, and the stored personal and sensitive information in those lost devices might be exploited for malicious use [6]. Therefore, it is unsurprising that designing secure solutions for mobile devices remains a topic of current interest and relevance. In addition to user authentication schemes [7], intrusion detection and prevention systems (IDSs/IPSs) are the most commonly used technology to protect the mobile environment.

Generally, based on specific detection approaches, intrusion detection systems (IDSs) can be generally classified into two types: *signature-based IDS* and *anomaly-based IDS*. For the former [8,9], it mainly detects a potential attack by examining packets and comparing them to known signatures. A signature (also called *rule*) is a kind of description for a known attack, which is usually generated based on expert knowledge. For the latter [10,11], it identifies an anomaly by comparing current events with preestablished normal profile. A normal profile often represents a normal behavior or network events. An alarm will be generated if any anomaly is detected. In addition, according to deployment, IDSs can be categorized into host-based systems and network-based systems.

As compared to IDSs, intrusion prevention systems (IPSs) are able to react and stop current adversary actions. Most IPSs can offer multiple prevention capabilities to adapt to various needs. IPSs usually allow security officers to choose the prevention capability configuration for each type of alert, such as enabling or disabling prevention, as well as specifying which type of prevention capability should be used. To control and reduce false actions, IPSs may have a learning or simulation mode that suppresses all prevention actions and instead indicates when a prevention action would have been performed. This allows security officers to monitor and fine-tune the configuration of the prevention capabilities before enabling prevention actions [8].

Motivation and focus. As mobile devices are often short of power and storage, traditional intrusion and prevention techniques are hard to deploy directly. However,

[!b]

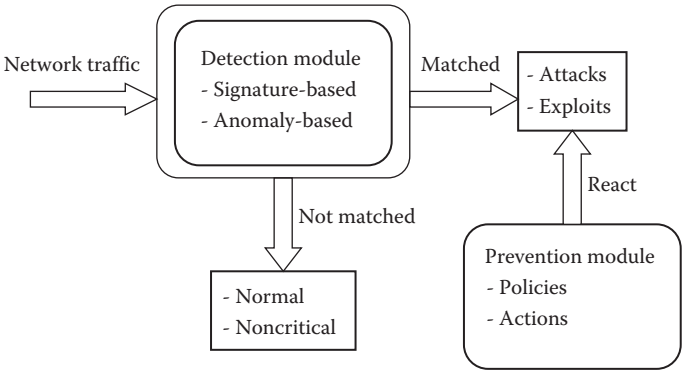

Figure 1.1 Generic workflow of IDSs and IPSs.

in the last decade, with the rapid development of mobile devices, there is an increasing need to apply IDSs/IPSs to protect these devices. It is very critical to establish an appropriate defense mechanism on such resource-limited devices. Motivated by this, in this chapter, we aim to present a review, introducing recent advancement within the last decade regarding the development of mobile intrusion detection and prevention efforts in the literature, and providing insights about current issues, challenges, and future directions in this area. Our contributions of this chapter can be summarized as below:

- First, we introduce the background of IDSs/IPSs in more detail and then investigate the development of IDS/IPS on mobile devices within the last decade, by examining notable work in the literature.
- Then, we identify the issues and challenges of designing such defense mechanisms on mobile devices, describe several potential solutions, and analyze the future directions in this field.

The remaining parts of this chapter are organized as follows. Section 1.2 introduces the background of IDSs and IPSs. Section 1.3 surveys the recent work regarding IDSs/IPSs on mobile devices within the last decade. Section 1.4 identifies issues and challenges of designing an appropriate defense on mobile devices. Section 1.5 describes several potential solutions and points out future directions in this area, and Section 1.6 concludes this chapter.

1.2 Background

In this section, we introduce the background of IDSs and IPSs, respectively. The generic workflow of IDSs/IPSs is depicted in Figure 1.1.

1.2.1 Intrusion Detection

Intrusion detection is the process of monitoring the events occurring in a local system or network and analyzing them for any sign of violations. An IDS is a software that automates the intrusion detection process. An IDS can typically provide several functions [8]:

- *Monitoring and recording information.* Targeted information can be usually recorded either locally or distributed according to concrete settings. For example, IDSs can store the information in an enterprise management server.
- *Notifying security officers.* This notification, also known as an alert, can be sent to security officers if any malicious event is identified. This alert has many forms such as emails, pages, and messages.
- *Generating reports.* Such reports often summarize the monitored events and provide details on particular interested events. Security officers can also track the historical status based on the reports.

Based on the deployment, there are two major types of IDSs: host-based IDS (HIDS) and network-based IDS (NIDS). An HIDS mainly monitors the events that occur in a local computer system, and then reports its findings. On the other hand, an NIDS aims to monitor current network traffic and detect network attacks through analyzing incoming network packets.

Moreover, according to the detection methods, an IDS can be primarily labeled as a signature-based, anomaly-based IDS. Figure 1.1 shows the generic workflow of intrusion detection.

- A signature is a pattern that corresponds to a known attack or exploit; thus, signature-based detection is the process of comparing signatures against observed events to identify possible incidents, for example, a telnet attempt with a username of "root," which is a violation of an organization's security policy [8].
- Anomaly-based detection is the process of comparing normal profiles against observed events to identify significant deviations. The profiles are developed by monitoring the characteristics of typical activity over a period of time. The system can use statistical methods to compare the characteristics of current activity to thresholds related to the profile.

Discussion. Signature-based detection is the simplest detection method, since it only compares the current unit of activity, such as a packet or a log entry, to a list of signatures using string comparison operations. Therefore, it is very effective at detecting known threats but largely ineffective at detecting previously unknown threats. By contrast, anomaly-based detection methods is good at detecting previously unknown threats, but may produce many false alarms. For example, if a particular maintenance activity that performs large file transfers occurs only once a

month, it might not be observed during the training period; when the maintenance occurs, it is likely to be considered a significant deviation from the profile and trigger an alert [8].

In addition to the above two detection methods, another detection is called stateful protocol analysis, which is the process of comparing predetermined profiles to identify deviations for each protocol status. It relies on vendor-developed universal profiles that specify how particular protocols should and should not be used. In the current literature, hybrid IDSs are developing at a rapid pace.

1.2.2 Intrusion Prevention

An IPS is a software that has all the capabilities of an IDS and can also attempt to stop possible incidents. As compared with an IDS, IPS technologies can respond to a detected threat by attempting to prevent it from succeeding. They can provide the following functions:

- *Stopping the attack.* IPSs can react to existing attacks, such as terminating the network connection or user session that is being used for the attack, and blocking access to the target from the offending user account, IP addresses, etc.
- *Changing the security environment.* An IPS can change the configuration of other security controls to disable an attack (i.e., reconfiguration of a device), or even launch patches to be applied to a host if the host has detected vulnerabilities.
- *Changing the attack's content.* Some IPSs can remove or replace malicious portions of an attack/program to make it benign. For instance, an IPS can remove an infected file attachment from an email and then permit the cleaned email to reach its recipient.

Discussion. As depicted in Figure 1.1, IPSs rely on the detection of potential threats. Thus, intrusion prevention will perform behind IDSs. A typical IPS usually contains the basic functions of an IDS, or specified as IDPS (intrusion detection and prevention system). It is worth noting that IDSs/IPSs can be classified differently according to specific targets and focuses. Table 1.1 describes several classification categories accordingly.

1.3 Review on IDSs/IPSs on Mobile Devices

As the mobile environment is different from computers, appropriate security systems should consider both detection accuracy and resource consumption. In this section, our focus is the application of IDSs/IPSs on mobile devices.

Table 1.1 Classification Categories According to Different Focuses

	Description
By Detection Topology	
Network-based methods	IDS/IPS is mainly deployed in a network.
Host-based methods	IDS/IPS is mainly deployed in a local system.
Hybrid methods	IDS/IPS searches anomalies in both network and local systems.
By Source	
Application level	IDS/IPS examines events from application level.
Kernel level	IDS/IPS examines events from kernel level.
Hardware level	IDS/IPS examines events from hardware level.
Hybrid source	IDS/IPS provides an overall protection throughout various levels.
By Detection Approach	
Anomaly-based methods	IDS/IPS identifies anomalies from normal profiles.
Signature-based methods	IDS/IPS identifies threats through signature matching.
Behavior-based methods	IDS/IPS identifies anomalies based on predefined valid behavior.
Hybrid methods	IDS/IPS combines the above detection approaches.
By Focus	
Malicious apps	IDS/IPS focuses on the detection of malware.
Information leakage system	IDS/IPS focuses on the detection of information leakage.
Hybrid	IDS/IPS considers all current threats.

1.3.1 From 2004 to 2006

Jacoby and Davis [12] designed a warning system via a host-based intrusion detection, which could alert security administrators to protect smaller mobile devices. They operated through the implementation of battery-based intrusion detection (called *B-bid*) on mobile devices by correlating attacks with their impact on device

power consumption using a rules-based host intrusion detection engine (HIDE). *B-bid* mainly measured the energy use over time to decide if an attack is present or not. As a result, probabilistic bounds for energy and time can have confidence intervals to measure abnormal behavior of power dissipation. In 2005, Nash et al. [13] also focused on battery exhaustion attacks, and designed an IDS, which takes into account the performance, energy, and memory constraints of mobile computing devices. Their system uses several parameters, such as CPU load and disk accesses, to estimate the power consumption using a linear regression model.

For general solutions, Jansen et al. [14] focused on mobile devices such as PDAs and proposed a framework for incorporating core security mechanisms in a unified manner, through adding improved user authentication, content encryption, organizational policy controls, virus protection, firewall and intrusion detection filtering, and virtual private network communication. This is a conceptual study; thus, it needs more tests in reality. Kannadiga et al. [15] discussed the characteristics of intrusion detection for pervasive computing environments (e.g., mobile, embedded, handheld devices) and described a mobile agent-based IDS to be deployed in a pervasive computing environment.

Miettinen et al. [16] advocated that mobile systems should be equipped with proper second-line defense mechanisms that can be used to detect and analyze security incidents. They proposed a framework for intrusion detection functionalities in mobile devices, which combines both host-based and network-based data collection. The data collection module is responsible for collecting and receiving the host-based monitoring data. Then, the data collection module forwards the monitoring data to the IDS module, which is responsible for the actual intrusion detection process. Then, Buennemeyer et al. [17] proposed a Battery-Sensing Intrusion Protection System (called *B-SIPS*) for mobile devices, which could alert on power changes detected on small wireless devices.

1.3.2 From 2007 to 2010

From 2007, more research has been done in protecting mobile devices against threats. Mazhelis and Puuronen [18] focused on personal mobile devices and designed a conceptual framework for mobile-user substitution detection. The framework was mainly based on the observation that user behavior and environment could reflect the user's personality in a recognizable way. More specifically, the framework could be decomposed into a descriptive and a prescriptive part. The former is concerned with the description of an object system (i.e., the user, his/her personality, behavior, and environment). The latter considers technical components (e.g., databases, knowledge bases, and processing units) that are needed to implement the UIV system based on the above object system. The proposed approach aimed at verifying the user's identity and detecting user substitution, which can be used in intrusion detection and fraud detection.

Venkataram et al. [19] proposed a generic model called Activity Event-Symptoms (AES) model for detecting fraud and attacks during the payment process in the mobile e-commerce environment. The model was designed to identify the symptoms and intrusions by observing various events/transactions that occur during mobile commerce activity. Niu et al. [20] studied the relationship between power consumption and parameters of system state on mobile computing devices by using genetic algorithm and artificial neural network. They found that an IDS may produce many false alarms, if it only relies on the CPU load to detect and identify a battery exhaust attack. It is suggested that an IDS should at least take hard-disk read and write operations and network transmission into account.

Later, Chung et al. [21] described an approach of Just-on-Time Data Leakage Protection Technique (JTLP), which aims to protect the user's private data of mobile devices from malicious activity and leakage. In a mobile device, all users' access can be monitored by JTLP monitor process (JTLPM) by using the access control method. When an unauthorized user or malicious code attempts to transmit some host of the outer network, JTLPM monitors all outbound suspicious packets to the external host. If a packet contains user's important data, this packet should be thrown out of the mobile device by the JTLPM. Brown and Ryan [22] argued that the current security model for a third-party application running on a mobile device requires its user to trust that application's vendor, but they cannot guarantee complete protection against the threats. Thus, they introduced a security architecture that prevents a third-party application deviating from its intended behavior, defending devices against previously unseen malware.

For anomaly detection, Schmidt et al. [23] demonstrated how to monitor a smartphone running Symbian OS to extract features for anomaly detection. They found that most of the top 10 applications preferred by mobile phone users could affect the monitored features in different ways. However, their features needed to be sent to a remote server, due to capability and hardware limitations.

Started from 2009, with the popularity of smartphones, most research has moved to malware detection. Shabtai et al. [24] evaluated a knowledge-based approach for detecting instances of known classes of mobile devices malware based on their temporal behavior. The framework relied on lightweight agent to continuously monitor time-stamped security data within the mobile device and to process the data using a light version of the Knowledge-Based Temporal Abstraction (KBTA) methodology. Then, Liu et al. [25] adopted power consumption analysis and proposed *VirusMeter*, a general malware detection method, to detect anomalous behaviors on mobile devices. The rationale underlying VirusMeter is the fact that mobile devices are usually battery powered and any malicious activity would inevitably consume some battery power. By monitoring power consumption on a mobile device, *VirusMeter* could catch misbehaviors that lead to abnormal power consumption.

Later, Xie et al. [26] proposed a behavior-based malware detection system named *pBMDS*, which adopts a probabilistic approach through correlating user inputs

with system calls to detect anomalous activities in cellphones. In particular, *pBMDS* observes unique behaviors of the mobile phone applications and the operating users on input and output constrained devices, and leverages a hidden Markov model (HMM) to learn application and user behaviors from two major aspects: process state transitions and user operational patterns. Based on these, *pBDMS* identifies behavioral differences between malware and human users. Shabtai et al. [27] then proposed a behavioral-based detection framework for Android mobile devices. The proposed framework adopted an HIDS that could continuously monitor various features and events obtained from the mobile device, and apply machine learning methods to classify the collected data as normal (benign) or abnormal (malicious).

1.3.3 From 2011 to 2013

Smartphone vendors will ship more than 450 million devices in 2011. At the same period, mobile malware was propagated in a quick manner; therefore, many IDSs/IPSs were built aiming to perform the detection of malware. Chaugule et al. [28] found that mobile malware always attempts to access sensitive system services on the mobile phone in an unobtrusive and stealthy fashion. For example, the malware may send messages automatically or stealthily interface with the audio peripherals on the device without the user's awareness and authorization. They then presented *SBIDF*, a Specification Based Intrusion Detection Framework, which utilizes the keypad or touchscreen interrupts to differentiate between malware and human activity. In the system, they utilized an application-independent specification, written in Temporal Logic of Causal Knowledge (TLCK), to describe the normal behavior pattern, and enforced this specification to all third-party applications on the mobile phone during runtime by monitoring the intercomponent communication pattern among critical components.

Then, Burguera et al. [29] focused on detecting malware in the Android platform. They put the detector to embed into a framework for collecting traces from an unlimited number of real users based on crowdsourcing. The framework could analyze the data collected in the central server using two types of data sets: those from artificial malware created for test purposes and those from real malware found in the wild. They considered that monitoring system calls should be one of the most accurate techniques to determine the behavior of Android applications, since they provide detailed low-level information. Bukac et al. [30] then summarized the development of HIDS before 2012 regarding the detection of intrusions from a host network traffic analysis, process behavior monitoring, and file integrity checking. Damopoulos et al. [31] focused on anomaly detection on mobile devices and conducted an evaluation among different classifiers (i.e., Bayesian networks, radial basis function, K-nearest neighbors, and random Forest) in terms of telephone calls, SMS, and web browsing history.

La Polla et al. [32] later presented a survey on the security of mobile devices, and pointed out that mobile services had significantly increased due to the different

form of connectivity provided by mobile devices, such as GSM, GPRS, Bluetooth, and Wi-Fi. They particularly described the state of the art on threats, vulnerabilities, and security solutions over the period from 2004 to 2011, by focusing on high-level attacks, like those to user applications. Roshandel et al. [33] argued that there is a fundamental difference in the attitude of a typical user when it comes to using their mobile device as compared to their personal computers. In addition, there is little by the way of security and privacy protection available on these mobile computing platforms. As a result, they developed a Layered Intrusion Detection and Remediation framework (LIDAR), which could automatically detect, analyze, protect, and remediate security threats on Android devices. More specifically, they developed several algorithms that may help detect abnormal behavior in the operation of Android smartphone and tablets that could potentially detect the presence of malware. Li and Clark [34] discussed that mobile devices were being rapidly integrated into enterprises, government agencies, and even the military, as these devices may hold valuable and sensitive content. They then provided an overview of the current threats based on data collected from observing the interaction of 75 million users with the Internet.

Curti et al. [35] investigated the correlations between the energy consumption of Android devices and some threats such as battery-drain attacks. They then described a model for the energy consumption of single hardware components of a mobile device during normal usage and under attack. Their model can be implemented in a kernel module and used to build up an energetic signature of both legal and malicious behaviors of Wi-Fi hardware component in different Android devices. The proposed Energy-Aware Intrusion Detection solutions were able to reliably detect attacks on mobile devices based on energetic footprints. To reach this, they adopted a step-wise approach: (1) they developed some built-in solutions allowing to measure and analyze energy consumption directly on each device, neglecting the usage of external hardware; (2) they performed measurements on some devices and benign/malicious applications and resulted in a database of consumption patterns for both benign smartphone activities and known attacks.

1.3.4 From 2014 to 2016

From 2014, most research studies aimed at developing a security mechanism to protect mobile devices against malware, while less work investigated how to build an IDS/IPS. Several surveys about the malware protection can be referred to in References 7 and 36.

For building IDSs/IPSs, Yazji et al. [37] focused on the problem of efficient intrusion detection for mobile devices via correlating the user's location and time data. They developed two statistical profiling approaches for modeling the normal spatiotemporal behavior of the users: one is based on an empirical cumulative probability measure and the other is based on the Markov properties of trajectories. An anomaly could be detected when the probability of a particular evolution

(e.g., location, time) matching the normal behavior of a given user becomes lower than a certain threshold. Papamartzivanos et al. [38] identified that modern app markets had been flooded with applications that not only threaten the security of the OS, but also in their majority, trample on user's privacy through the exposure of sensitive information. They discussed and developed a cloud-based crowdsourcing mechanism that could detect and alert for changes in the app's behavior.

Later, Sun et al. [39] developed various host-based intrusion prevention systems (HIPS) on Android devices, in order to protect smartphones and prevent privacy leakage. In particular, they analyzed the implementations, strengths, and weaknesses of three popular HIPS architectures, and demonstrated a severe loophole and weakness of an existing popular HIPS product in which hackers can readily exploit. Based on this, they designed a more secure and extensible HIPS platform called *Patronus*. *Patronus* can dynamically detect existing malware based on runtime information, without the need to modify the hardware. Damopoulos et al. [40] then investigated two issues: the first one was how to define an architecture, which could be used for implementing and deploying a system in a dual-mode (host/cloud) manner and irrespectively of the underlying platform, and the second one was how to evaluate such a system. Their approach allows users to argue in favor of a hybrid host/cloud IDS arrangement and to provide quantitative evaluation facts on if and in which cases machine learning-driven detection is affordable when executed on devices such as smartphones. Damopoulos et al. [41] later described a tool that was able to dynamically analyze any iOS software in terms of method invocation (i.e., which API methods the application invokes and under what order), and produce exploitable results that can be used to manually or automatically trace software behavior to decide if it contains malicious code.

1.3.5 Discussion

Mobile devices have already become a part of people's lives. Once a mobile device is compromised, a wide range of threats may occur. For example, attackers might sell the uncovered personal data; they might leverage stored credentials to gain access to a device; or they might use the device as a gateway into enterprise data and resources by leveraging a trust relationship between the device and the IT infrastructure [34]. Even worse, the device could be put into a botnet or used to send unauthorized premium-rate SMS messages. Thus, how to manage these risks at scale and the problem is becoming more complex.

Intrusion detection and prevention techniques are one of the promising solutions to secure mobile devices. However, as compared to a desktop computer, mobile devices are often short of power and resources. Traditional IDS/IPS tools may not be applicable on mobile platforms. Therefore, there is a need for developing advanced and energy-ware intrusion detection and prevention solutions.

In addition to local mobile device protection, existing mobile networks usually consist of many mobile devices (e.g., medical smartphone network [42]), so it is also

an important topic to secure the mobile environment, such as adding monitors [43], verifying locations [44], and performing deep packet inspection [45].

To compare the performance between IDSs/IPSs, three major factors should be considered: detection accuracy, time consumption, and CPU usage.

- *Detection accuracy.* This factor is extremely important for an IDS/IPS, where an ideal detection system should provide high accuracy and low false rates.
- *Time consumption.* Intrusion detection is a time-sensitive task, where a quick identification can reduce the damage (e.g., financial loss, data leakage). Generally, an ideal detection system should be able to identify threats in a faster manner.
- *CPU usage.* As mobile devices have limited resources, CPU usage becomes a critical factor to determine whether an IDS/IPS is feasible in real-world applications. An ideal detection system is expected to consume less CPU and ensure the availability of devices.

1.4 Issues and Challenges

As mentioned, mobile devices face the same (or a higher) level of malicious attacks that have plagued the desktop computing environments [34]. However, typical mobile devices are different from common computers:

- *Mobility.* Mobile devices are much more flexible due to their size and weight so that users can bring their devices everywhere. In comparison, a typical desktop computer is often deployed in a particular site. The mobility requires to design more dynamic security mechanisms on mobile devices.
- *Limited resources.* A typical mobile device (e.g., phones, iPads) has only limited power and computation capabilities, making it not powerful enough to implement traditional intrusion detection and prevention techniques. When designing a mobile security mechanism, there is a balance that should be considered between performance and energy.

Owing to these features, mobile IDS/IPS may suffer from many issues and challenges, which are the same as in traditional computing environments. The major issues and challenges are summarized in Figure 1.2.

- *Event overload*. Mobile devices can generate a massive amount of local and network events, with the rapid development of system computation. Such events may exceed the capability of a mobile IDS/IPS. For instance, a signature-based NIDS may drop lots of network packets if the incoming packets exceed their maximum processing capability.

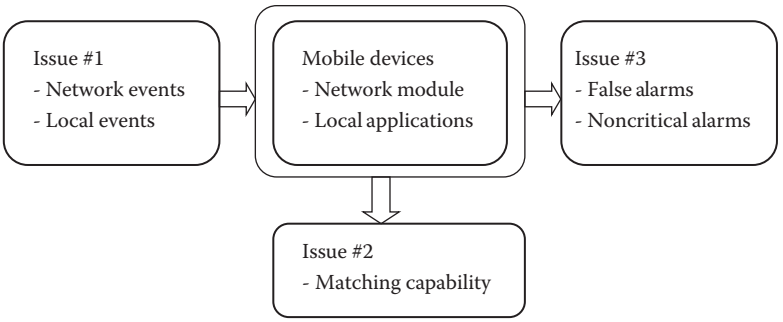

Figure 1.2 Issues and challenges of mobile IDSs/IPSs.

- *Expensive signature matching*. For signature-based IDSs/IPSs, the signature matching module is often too expensive for resources that the computing burden is at least linear to the size of an incoming string [46]. Subsequently, the performance of IDS/IPS may be greatly degraded due to the heavy operational burden.
- *Massive false alarms*. Both signature- and anomaly-based IDSs/IPSs may generate a large number of false alarms, which can significantly increase the difficulty in analyzing alarms and adversely affect the analysis results.

These issues and challenges can greatly degrade the performance of a mobile IDS/IPS, such as missing network and local events and dispersing analysis directions. In addition, these issues can increase the workload and burden of deploying IDSs/IPSs, causing mobile devices to be out of availability in a quick manner. Therefore, it is a critical topic for developing an appropriate security mechanism on the mobile environment.

1.5 Solutions and Future Trend

In this section, we describe several potential solutions for the above issues and challenges, and point out the future trend in this field.

1.5.1 Potential Solutions

In order to design a proper security mechanism on mobile devices, it is necessary to consider the above issues and make particular improvement with particular goals. According to the issues in Figure 1.2, it is promising to implement various modules or additional mechanisms (e.g., packet filter, alarm filter) to strengthen the IDS/IPS performance.

1.5.1.1 Packet Filter Development

The reduction of packet filter is a straightforward solution to improve the performance of intrusion detection and prevention on mobile devices. The idea of the packet filter is to reduce the number of target packets through filtering out certain packets early based on their IP confidence. To build an appropriate packet filtration mechanism, several factors should be considered:

- The filter should have a minimum impact on system and network performance.
- The filter should be efficient and provide a good filtration rate.
- The filter should not degrade the security level of IDSs/IPSs.

Based on these factors, there are several kinds of event/packet filters in the literature, such as blacklist-based, list-based, and trust-based filter.

- *Blacklist-based event/packet filter.* Blacklist is a common technique that is used in filtering events and packets. This type of filter can alleviate the burden of either a signature- or anomaly-based IDS/IPS in processing a massive number of target events. This filter can realize a weighted ratio-based method (statistic-based method) in the monitor engine to calculate the IP confidences and to generate the blacklist [47,48].
- *List-based event/packet filter.* In addition to blacklist, the whitelist can also be useful in real applications. Thus, it is a solution to combine the whitelist and the blacklist techniques in constructing a list-based event/packet filter [49].
- *Trust-based event/packet filter.* To leverage the blacklist generation, trust computation can be applied to such filters. For example, Bayesian inference model [50] can be used to enhance the computation of IP confidence and further improve the performance of filters in a large-scale network environment. One basic assumption is that all events/packets are independent of each other.

1.5.1.2 Alarm Filter Development

The large number of false alarms can greatly reduce the efficiency of an IDS and significantly increase the burden of analyzing these alarms. For example, thousands of alarms may be generated in one day, which are a big burden for a security officer. Even worse, false alarms may have a negative impact on the analysis of IDS outputs. Hence, false alarm reduction is an important issue for IDSs/IPSs. There are many techniques that can be considered:

- *Adaptive false alarm filter.* In real scenarios, machine learning is often applied to false alarm reduction. However, the filtration accuracy of an algorithm may be fluctuant. As a result, an adaptive false alarm filter is promising to select the best algorithm from a pool of algorithms [51]. Such filter enables the

algorithm selection to be performed in an adaptive way with the purpose of maintaining a high and stable filtration accuracy.

- *Knowledge-based alarm filter.* Expert knowledge is very crucial in deciding whether an alarm is critical or not. Therefore, knowledge-based alert verification can be combined to construct an alarm filter [52], that is, employing a rating mechanism to classify incoming alarms.
- *Contextual alarm filter*. Many false alarms are produced, since the IDSs/IPSs are not aware of the contextual information of their deployed environment. Hence, considering contextual information is a promising method to improve the quality of output alarms [53].

1.5.1.3 Matching Capability Improvement

The expensive process of signature matching is a key limiting factor for deploying IDSs/IPSs on mobile devices. Therefore, there is a need for improving the matching capability. Besides traditional string matching algorithms, exclusive signature matching scheme is a promising solution.

The major difference between regular signature matching and exclusive signature matching is that the latter aims to identify a mismatch rather than to confirm an accurate match during the signature matching [46]. This scheme can be adaptive in selecting the most appropriate single character for exclusive signature matching in terms of different network environments. In particular, our scheme respectively calculates the character frequency of both stored NIDS signatures and matched signatures with the purpose of adaptively and sequentially determining the most appropriate character in the comparison with packet payload [54].

1.5.1.4 Overall Improvement

Moreover, the above solutions can be integrated into one comprehensive mechanism [55]. Taking EFM [56] as an example, this mechanism is composed of three major components: a *context-aware blacklist-based packet filter*, an *exclusive signature matching component*, and a *KNN-based false alarm filter*. In particular, the *context-aware blacklist-based packet filter* is responsible for reducing the workload of IDSs by filtering out network packets by means of IP reputation. The *exclusive signature matching component* is implemented in the context-aware blacklist-based packet filter aiming to speed up the process of signature matching. The *KNN-based false alarm filter* is responsible for filtering out false alarms (positives) that are produced by the packet filter and the IDS.

1.5.2 Future Trend

Intrusion detection and prevention is a basic solution to protect mobile devices against malicious use. With the development of mobile platforms, malware and theft use are the major threats. Additionally, several vulnerabilities in the operation

system may threaten the phone security [57]. As a result, IDSs/IPS should be further enhanced through combining new features.

- *Behavioral-based detection.* The current smartphones often feature a touchscreen as the input method. As compared with the traditional button-based input, touchscreen enables more actions such as multitouch and touch movement. For instance, multitouch is a new feature, where users can touch the screen with multiple fingers at the same time [58,59]. The new feature may result in novel threats such as smudge attacks [60], but also enable behavioral-based detection (e.g., multitouch-included authentication [61–67]). With more biometrics implemented on mobile devices, biometric authentication should be given more attention in the future.
- *Graphical passwords.* There is an increasing number of applications installed on mobile devices such as graphical passwords; thus, it is necessary to apply IDS/IPS techniques to those passwords. Such combination can provide more comprehensive protection to the mobile environment [4,68]. Several research studies on graphical passwords can be referred to in References 69–79.
- *Cloud-based mechanism.* Computation resources are often a key limiting factor for deploying complex IDS/IPS techniques on mobile devices. With the advent of cloud, it is promising to offload expensive operations to the cloud side (e.g., offloading the signature matching process [80]).

1.6 Conclusions

Research in mobile device and smartphone security has been conducted for several years. Security solutions for mobile devices and smartphones must defend against viruses, malware, botnets, and attacks through the deployment of a wide spectrum of mobile applications. Intrusion detection and prevention techniques are one basic solution to protect mobile devices and users' privacy. However, it is not an easy task for building an appropriate defense mechanism on resource-limited mobile devices.

In this chapter, we present a review, introducing recent advancement within the last decade regarding the development of mobile intrusion detection and prevention efforts in the literature. Then, we give insights about current issues and challenges for mobile IDSs/IPSs such as overhead event/packets, massive false alarms, and matching bottleneck. By focusing on these issues, we introduce potential solutions in constructing event/packet alarm filter and improving signature matching. At last, we point out that future mobile IDSs/IPSs may cooperate with more applications such as behavioral-based detection and graphical passwords and utilize the resources from new environments such as cloud.

References

1. Smartphone Vendor Market Share, Q2. http://www.idc.com/prodserv/smartphone-market-share.jsp, 2016.

2. A. K. Karlson, A. B. Brush, and S. Schechter, Can I borrow your phone?: Understanding concerns when sharing mobile phones, in *Proceedings of the 27th International Conference on Human Factors in Computing Systems (CHI)*, ACM, New York, pp. 1647–1650, 2009.
3. P. Dunphy, A. P. Heiner, and N. Asokan, A closer look at recognition-based graphical passwords on mobile devices, in *Proceedings of the Sixth Symposium on Usable Privacy and Security (SOUPS)*, ACM, New York, pp. 1–12, 2010.
4. Y. Meng, W. Li, and L.-F. Kwok, Enhancing click-draw based graphical passwords using multi-touch on mobile phones, in *Proceedings of the 28th IFIP TC 11 International Information Security and Privacy Conference (IFIP SEC)*, Springer, Berlin, Heidelberg, Auckland, New Zealand, pp. 55–68, 2013.
5. A. Shabtai, Y. Fledel, U. Kanonov, Y. Elovici, S. Dolev, and C. Glezer, Google Android: A comprehensive security assessment, *IEEE Security Privacy*, **8**(2): 35–44, 2010.
6. Mobile and NCSA. Report on Consumer Behaviors and Perceptions of Mobile Security, January 2012. Available at: http://docs.nq.com/NQ_Mobile_Security_Survey_Jan2012.pdf.
7. W. Meng, D. S. Wong, S. Furnell, and J. Zhou, Surveying the development of biometric user authentication on mobile phones, *IEEE Communications Surveys Tutorials*, **17**: 1268–1293, 2015.
8. K. Scarfone and P. Mell, Guide to Intrusion Detection and Prevention Systems (IDPS), NIST Special Publication. http://csrc.nist.gov/publications/nistpubs/800-94/SP800-94.pdf, pp. 800–894, 2007.
9. G. Vigna and R. A. Kemmerer, NetSTAT: A network-based intrusion detection approach, in *Proceedings of the 1998 Annual Computer Security Applications Conference (ACSAC)*, IEEE Press, New York, pp. 25–34, 1998.
10. A. K. Ghosh, J. Wanken, and F. Charron, Detecting anomalous and unknown intrusions against programs, in *Proceedings of the 1998 Annual Computer Security Applications Conference (ACSAC)*, IEEE Computer Society, Scottsdale, AZ, USA, pp. 259–267, 1998.
11. A. Valdes and D. Anderson, Statistical methods for computer usage anomaly detection using NIDES, Technical Report, SRI International, January 1995.
12. G. A. Jacoby and N. J. Davis IV, Battery-based intrusion detection, in *Proceedings of IEEE Global Telecommunications Conference (GLOBECOM)*, IEEE, December Dallas, Texas, USA, pp. 2250–2255, 2004.
13. D. C. Nash, T. L. Martin, D. S. Ha, and M. S. Hsiao, Towards an intrusion detection system for battery exhaustion attacks on mobile computing devices, in *Proceedings of the 3rd IEEE International Conference on Pervasive Computing and Communications Workshops*, IEEE, Kauai Island, HI, USA, pp. 141–145, 2005.
14. W. Jansen, V. Korolev, S. Gavrila, T. Heute, and C. Séveillac, A unified framework for mobile device security, in *Proceedings of the International Conference on Security and Management (SAM)*, CSREA Press, Las Vegas, Nevada, USA, pp. 9–14, 2004.
15. P. Kannadiga, M. Zulkernine, and S. I. Ahamed, Towards an intrusion detection system for pervasive computing environments, in *Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC)*, IEEE, Bangalore, India, pp. 277–282, 2005.
16. M. Miettinen, P. Halonen, and K. Hatonen, Host-based intrusion detection for advanced mobile devices, in *Proceedings of the International Conference on Advanced Information Networking and Applications (AINA)*, IEEE Computer Society, Vienna, Austria, pp. 72–76, 2006.

17. T. K. Buennemeyer, G. A. Jacoby, W. G. Chiang, R. C. Marchany, and J. G. Tront, Battery-sensing intrusion protection system, in *Proceedings of the 2006 IEEE Workshop on Information Assurance*, IEEE Computer Society, Egham, Surrey, UK, pp. 176–183, 2006.
18. O. Mazhelis and S. Puuronen, A framework for behavior-based detection of user substitution in a mobile context, *Computers & Security*, **26**(2): 154–176, 2007.
19. P. Venkataram, B. Sathish Babu, M. K. Naveen, and G. H. Samyama Gungal, A method of fraud & intrusion detection for e-payment systems in mobile e-commerce, in *Proceedings of the IEEE International Performance, Computing, and Communications Conference*, IEEE Computer Society, New Orleans, Louisiana, USA, pp. 395–401, 2007.
20. L. Niu, X. Tan, and B. Yin, Estimation of system power consumption on mobile computing devices, in *Proceedings of the International Conference on Computational Intelligence and Security (CIS)*, IEEE, Harbin, Heilongjiang, China, pp. 1058–1061, 2007.
21. B.-H. Chung, Y.-H. Kim, and K.-Y. Kim, Just-on-time data leakage protection for mobile devices, in *Proceedings of the International Conference on Advanced Communication Technology (ICACT)*, IEEE, Pyeongchang, Korea, pp. 1914–1915, 2008.
22. A. Brown and M. Ryan, Monitoring the execution of third-party software on mobile devices, in *Proceedings of the 11th International Symposium on Recent Advances in Intrusion Detection (RAID)*, Springer, Cambridge, MA, USA, pp. 410–411, 2008.
23. A.-D. Schmidt, F. Peters, F. Lamour, S. A. Camtepe, and S. Albayrak, Monitoring smartphones for anomaly detection, in *Proceedings of the 1st International Conference on MOBILe Wireless MiddleWARE, Operating Systems, and Applications*, ICST, Innsbruck, Austria, pp. 1–7, 2008.
24. A. Shabtai, U. Kanonov, and Y. Elovici, Detection, alert and response to malicious behavior in mobile devices: Knowledge-based approach, in *Proceedings of RAID 2009*, LNCS 5758, Springer, Saint-Malo, Brittany, France, pp. 357–358, 2009.
25. L. Liu, G. Yan, X. Zhang, and S. Chen, VirusMeter: Preventing your cellphone from spies, in *Proceedings of RAID 2009*, Springer, Saint-Malo, Brittany, France, pp. 244–264, 2009.
26. L. Xie, X. Zhang, J.-P. Seifert, and S. Zhu, PBMDS: A behavior-based malware detection system for cellphone devices, in *Proceedings of the 3rd ACM Conference on Wireless Network Security (WiSec)*, ACM, Hoboken, NJ, USA, pp. 37–48, 2010.
27. A. Shabtai and Y. Elovici, Applying behavioral detection on Android-based devices, in *Proceedings of Mobilware 2010*, Springer, Chicago, IL, USA, pp. 235–249, 2010.
28. A. Chaugule, Z. Xu, and S. Zhu, A specification based intrusion detection framework for mobile phones, in *Proceedings of ACNS*, Springer, Nerja, Spain, pp. 19–37, 2011.
29. I. Burguera, U. Zurutuza, and S. Nadjm-Tehrani, Crowdroid: Behavior-based malware detection system for Android, in *Proceedings of ACM Conference on Computer and Communications Security*, ACM, Chicago, IL, USA, pp. 15–25, 2011.
30. V. Bukac, P. Tucek, and M. Deutsch, Advances and challenges in standalone host-based intrusion detection systems, in *Proceedings of TrustBus 2012*, LNCS 7449, Springer, Vienna, Austria, pp. 105–117, 2012.
31. D. Damopoulos, S. A. Menesidou, G. Kambourakis, M. Papadaki, N. Clarke, and S. Gritzalis, Evaluation of anomaly-based IDS for mobile devices using machine learning classifiers, *Security and Communication Network*, **5**(1): 314, 2012.

32. M. La Polla, F. Martinelli, and D. Sgandurra, A survey on security for mobile devices, *IEEE Communications Surveys and Tutorials*, **15**(1): 446–471, 2013.
33. R. Roshandel, P. Arabshahi, and R. Poovendran, LIDAR: A layered intrusion detection and remediation framework for smartphones, in *Proceedings of the 4th ACM Sigsoft International Symposium on Architecting Critical Systems*, ACM Press, Vancouver, British Columbia, Canada, pp. 27–32, 2013.
34. Q. Li and G. Clark, Mobile security: A look ahead, *IEEE Security & Privacy*, **11**: 78–81, 2013.
35. M. Curti, A. Merlo, M. Migliardi, and S. Schiappacasse, Towards energy-aware intrusion detection systems on mobile devices, in *Proceedings of the 2013 International Conference on High Performance Computing and Simulation*, IEEE, Helsinki, Finland, pp. 289–296, 2013.
36. P. Faruki, A. Bharmal, V. Laxmi, G. Vijay, S. G. Manoj, M. Conti, and M. Rajarajan, Android security: A survey of issues, malware penetration, and defenses, *IEEE Communications Surveys and Tutorials*, **17**(2): 998–1022, 2015.
37. S. Yazji, P. Scheuermann, R. P. Dick, G. Trajcevski, and R. Jin, Efficient location aware intrusion detection to protect mobile devices, *Personal and Ubiquitous Computing*, **18**(1): 143–162, 2014.
38. D. Papamartzivanos, D. Damopoulos, and G. Kambourakis, A cloud-based architecture to crowdsource mobile app privacy leaks, in *Proceedings of the 18th Panhellenic Conference on Informatics (PCI)*, ACM, Athens, Greece, pp. 1–6, 2014.
39. M. Sun, M. Zheng, J. C. S Lui, and X. Jiang, Design and implementation of an Android host-based intrusion prevention system, in *Proceedings of the 30th Annual Computer Security Applications Conference (ACSAC)*, ACM Press, New Orleans, LA, USA, pp. 226–235, 2014.
40. D. Damopoulos, G. Kambourakis, and G. Portokalidis, The best of both worlds: A framework for the synergistic operation of host and cloud anomaly-based IDS for smartphones, in *Proceedings of the Seventh European Workshop on System Security (EuroSec)*, ACM, New York, Article 6, 2014.
41. D. Damopoulos, G. Kambourakis, S. Gritzalis, and S. O. Park, Exposing mobile malware from the inside (or what is your mobile app really doing?), *Peer-to-Peer Networking and Applications*, **7**(4): 687–697, 2014.
42. W. Meng, W. Li, X., Yang, and K. K. R Choo, A Bayesian inference-based detection mechanism to defend medical smartphone networks against insider attacks, *Journal of Network and Computer Applications*, **78**: 162–169, 2017.
43. J. S. Ransbottom and G. A. Jacoby, Monitoring mobile device vitals for Effective Reporting (ER), in *Proceedings of the IEEE Conference on Military Communications*, IEEE, Washington, DC, USA, pp. 329–335, 2006.
44. R. A. Malaney, Wireless intrusion detection using tracking verification, in *Proceedings of the IEEE International Conference on Communications*, IEEE, Glasgow, Scotland, pp. 1558–1563, 2007.
45. G. A. Jacoby and S. Mosley, Mobile security using separated deep packet inspection, in *Proceedings of the 5th IEEE Consumer Communications and Networking Conference (CCNC)*, IEEE, Las Vegas, NV, USA, pp. 482–487, 2008.
46. Y. Meng, W. Li, and L.-F. Kwok, Single character frequency-based exclusive signature matching scheme, in *The 11th IEEE/ACIS International Conference on Computer and Information Science (ICIS 2012)*, Studies in Computational Intelligence, Springer, Shanghai, China, pp. 67–80, 2012.

47. Y. Meng and L.-F. Kwok, Adaptive context-aware packet filter scheme using statistic-based blacklist generation in network intrusion detection, in *Proceedings of the 7th International Conference on Information Assurance and Security (IAS)*, IEEE, Melacca, Malaysia, pp. 74–79, 2011.
48. Y. Meng and L.-F. Kwok, Adaptive blacklist-based packet filter with a statistic-based approach in network intrusion detection, *Journal of Network and Computer Applications*, **39**: 83–92, 2014.
49. Y. Meng and L.-F. Kwok, Enhancing list-based packet filter using ip verification mechanism against ip spoofing attack in network intrusion detection, in *Proceedings of the 6th International Conference on Network and System Security (NSS)*, Lecture Notes in Computer Science 7645, Springer, Wuyishan, Fujian, China, pp. 1–14, 2012.
50. Y. Meng, L.-F. Kwok, and W. Li, Towards designing packet filter with a trust-based approach using Bayesian inference in network intrusion detection, in *Proceedings of the 8th International Conference on Security and Privacy in Communication Networks (SECURECOMM)*, Lecture Notes in ICST 106, Springer, Padua, Italy, pp. 203–221, 2012.
51. Y. Meng and L.-F. Kwok, Adaptive false alarm filter using machine learning in intrusion detection, in *Proceedings of the 6th International Conference on Intelligent Systems and Knowledge Engineering (ISKE)*, Advances in Intelligent and Soft Computing, Springer, Shanghai, China, pp. 573–584, 2011.
52. Y. Meng, W. Li, and L.-F. Kwok, Intelligent alarm filter using knowledge-based alert verification in network intrusion detection, in *The 20th International Symposium on Methodologies for Intelligent Systems (ISMIS)*, Lecture Notes in Artificial Intelligence 7661, Springer, Macau, China, pp. 115–124, 2012.
53. Y. Meng and L.-F. Kwok, Adaptive non-critical alarm reduction using hash-based contextual signatures in intrusion detection, *Computer Communications*, **38**: 50–59, Elsevier, 2014.
54. Y. Meng, W. Li, and L.-F. Kwok, Towards adaptive character frequency-based exclusive signature matching scheme and its applications in distributed intrusion detection, *Computer Networks*, **57**(17): 3630–3640, Elsevier, 2013.
55. W. Meng and L.-F. Kwok, Enhancing the performance of signature-based network intrusion detection systems: An engineering approach, *HKIE Transactions*, **21**(4): 209–222, Taylor & Francis, 2014.
56. W. Meng, W. Li, and L.-F. Kwok, EFM: Enhancing the performance of signature-based network intrusion detection systems using enhanced filter mechanism, *Computers & Security*, **43**: 189–204, Elsevier, 2014.
57. W. Meng, W. H. Lee, S. R. Murali, and S. P. T Krishnan, JuiceCaster: Towards automatic juice filming attacks on smartphones, *Journal of Network and Computer Applications*, **68**: 201–212, 2016.
58. D. Fiorella, A. Sanna, and F. Lamberti, Multi-touch user interface evaluation for 3D object manipulation on mobile devices, *Journal on Multimodal User Interfaces*, **4**(1): 3–10, 2010.
59. M. Frank, R. Biedert, E. Ma, I. Martinovic, and D. Song, Touchalytics: On the applicability of touchscreen input as a behavioral biometric for continuous authentication, *IEEE Transactions on Information Forensics and Security*, **8**(1): 136–148, 2013.
60. T. Kwon and S. Na, TinyLock: Affordable defense against smudge attacks on smartphone pattern lock systems, *Computers & Security*, **42**: 137–150, 2014.
61. D. Damopoulos, G. Kambourakis, and S. Gritzalis, From keyloggers to touchloggers: Take the rough with the smooth, *Computers & Security*, **32**: 102–114, 2013.

62. A. De Luca, A. Hang, F. Brudy, C. Lindner, and H. Hussmann, Touch Me Once and I Know It's You!: Implicit authentication based on touch screen patterns, in *Proceedings of the 2012 ACM Annual Conference on Human Factors in Computing Systems (CHI)*, ACM, New York, pp. 987–996, 2012.
63. Y. Meng, D. S. Wong, and L.-F. Kwok, Design of touch dynamics based user authentication with an adaptive mechanism on mobile phones, in *Proceedings of the 29th Annual ACM Symposium on Applied Computing (SAC)*, ACM Press, Gyeongju, Korea, pp. 1680–1687, 2014.
64. Y. Meng, D. S. Wong, R. Schlegel, and L.-F. Kwok, Touch gestures based biometric authentication scheme for touchscreen mobile phones, in *Proceedings of the 8th China International Conference on Information Security and Cryptology (INSCRYPT)*, Springer, Beijing, China, pp. 331–350, 2012.
65. I. Oakley and A. Bianchi, Multi-touch passwords for mobile device access, in *Proceedings of the 2012 ACM Conference on Ubiquitous Computing (UbiComp)*, ACM Press, Pittsburgh, PA, USA, pp. 611–612, 2012.
66. N. Sae-Bae, K. Ahmed, K. Isbister, and N. Memon, Biometric-rich gestures: A novel approach to authentication on multi-touch devices, in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI)*, ACM Press, Austin, Texas, USA, pp. 977–986, 2012.
67. N. Sae-Bae, N. Memon, K. Isbister, and K. Ahmed, Multitouch gesture-based authentication, *IEEE Transactions on Information Forensics and Security*, **9**(4): 568–582, 2014.
68. J. Thorpe and P. C. Van Oorschot, Human-seeded attacks and exploiting hotspots in graphical passwords, in *Proceedings of 16th USENIX Security Symposium on USENIX Security Symposium*, USENIX Association, Boston, MA, USA, pp. 1–16, 2007.
69. W. Meng, W. Li, L. F. Kwok, and K. K. R Choo, Towards enhancing click-draw based graphical passwords using multi-touch behaviours on smartphones, *Computers & Security*, **65**: 213–229, 2017.
70. T. Takada and Y. Kokubun, MTAPIN: Multi-touch key input enhances security of PIN authentication while keeping usability, *International Journal of Pervasive Computing and Communications*, **10**(3): 276–290, 2014.
71. F. Tari, A. A. Ozok, and S. H. Holden, A comparison of perceived and real shoulder-surfing risks between alphanumeric and graphical passwords, in *Proceedings of the 2nd Symposium on Usable Privacy and Security (SOUPS)*, ACM, New York, NY, USA, pp. 56–66, 2006.
72. D. Van Thanh, Security issues in mobile eCommerce, in *Proceedings of the 11th International Workshop on Database and Expert Systems Applications (DEXA)*, IEEE, USA, pp. 412–425, 2000.
73. P. C. Van Oorschot, A. Salehi-Abari, and J. Thorpe, Purely automated attacks on PassPoints-style graphical passwords, *IEEE Transactions on Information Forensics and Security*, **5**(3): 393–405, 2010.
74. P. C. Van Oorschot and J. Thorpe, Exploiting predictability in click-based graphical passwords, *Journal of Computer Security*, **19**(4): 699–702, 2011.
75. S. Wiedenbeck, J. Waters, J.-C. Birget, A. Brodskiy, and N. Memon, Passpoints: Design and longitudinal evaluation of a graphical password system, *International Journal of Human-Computer Studies*, **63**(1–2): 102–127, 2005.
76. S. Wiedenbeck, J. Waters, J.-C. Birget, A. Brodskiy, and N. Memon, Authentication using graphical passwords: Effects of tolerance and image choice, in *Proceedings*

of the 2005 Symposium on Usable Privacy and Security (SOUPS), ACM, Pittsburgh, Pennsylvania, USA, pp. 1–12, 2005.

77. T.-S. Wu, M.-L. Lee, H.-Y. Lin, and C.-Y. Wang, Shoulder-surfing-proof graphical password authentication scheme, *International Journal of Information Security*, **13**(3): 245–254, 2014.
78. B. B. Zhu, J. Yan, G. Bao, M. Yang, and N. Xu, Captcha as graphical passwords—A new security primitive based on hard AI problems, *IEEE Transactions on Information Forensics and Security*, **9**(6): 891–904, 2014.
79. Y. Meng, Designing click-draw based graphical password scheme for better authentication, in *Proceedings of IEEE International Conference on Networking, Architecture, and Storage (NAS)*, IEEE Press, Xiamen, China, pp. 39–48, 2012.
80. Y. Meng, W. Li, and L.-F. Kwok, Design of cloud-based parallel exclusive signature matching model in intrusion detection, in *The 15th IEEE International Conference on High Performance Computing and Communications (HPCC)*, IEEE, pp. 2013.

Chapter 2

Attacking Smartphone Security and Privacy

Vincent F. Taylor and Ivan Martinovic

Contents

2.1 Introduction

The smartphone landscape continues to grow at an explosive pace as devices become more powerful, feature-rich, and more affordable to the average consumer. Gartner reports smartphone sales as exceeding 1.4 billion units in 2015, up 14.4% over the previous year [1]. Smartphones offer greatly increased functionality over traditional feature phones due to the availability of full-blown operating systems providing advanced APIs to third-party app developers. Smartphones are predominantly powered by Android or iOS, with Android maintaining a commanding lead of the market with 84.7% market share as of 2015 Q3 with iOS in a far second at 13.1% [2]. Other operating systems represented in the landscape include Windows Phone and BlackBerry among others. On top of the operating system, smartphones offer a variety of network interfaces for connectivity, multitasking facilities, and open application programming interfaces (APIs) for supporting third-party app development. Android and iOS have rich app marketplaces, each offering access to approximately 1.5 million apps [3,4] that add additional functionality to the smartphone. The always-connected, extremely extensible nature of smartphones exposes a large footprint on the device where weaknesses in the underlying hardware or software may be exploited by an attacker.

The smartphone landscape is very large, and has a number of layers of software, protocols, and services that work together to deliver an experience to the consumer. The interaction between consumer, apps, smartphone, service provider, and the wider Internet is supported by various wireless protocols that provide connectivity. Thus, a smartphone may be vulnerable to attacks coming from installed apps, wireless interfaces, running services, and the underlying configuration of the device. We are motivated to systematize this knowledge of attacks and attack vectors, as this will provide a compendium to security researchers intending to develop intrusion detection and prevention systems* (IDS) for the smartphone ecosystem. We do this by comprehensively enumerating the ways in which the security and privacy of a smartphone can be attacked. By understanding the ways in which smartphones can be attacked, we obtain a mechanism to compare them to traditional workstations, giving useful insight into the additional or varied risks that need to be addressed when building technology to secure smartphones.

* For brevity, we refer to intrusion detection and prevention systems as simply IDS for the remainder of this chapter.

2.2 Background

To understand the ways smartphones are attacked, we first need to understand the operating systems that run on these devices and the security models and features they employ. There are four major smartphone operating systems: Android, iOS, Windows Phone, and BlackBerry [5]. In addition to typical low-level tasks such as memory management and process scheduling, smartphone operating systems provide features critical in today's smartphone landscape, such as allowing access to a touchscreen, camera, Bluetooth, Wi-Fi, NFC, GPS, microphone, and other such hardware. Aside from providing access to the typical smartphone hardware, the operating system also mediates access to the underlying cellular radio, enabling communication with a mobile network carrier.

2.2.1 iOS

iOS is a mobile operating system developed by Apple Inc. It has a healthy app ecosystem that surrounds it with over 1.4 million iOS applications available for download. The operating system itself is proprietary, closed source, and written in C, C++, Objective-C, and Swift. It is a Unix-like operating system and features a hybrid kernel that runs on 64/32-bit ARM processors. Before iOS apps are made available to the public in the Apple App Store, they must undergo a thorough vetting process by Apple. Apps must pass reliability testing and other analysis to ensure that they are not malicious or otherwise unsavory. Apple's vetting process includes manual testing and static analysis to determine whether an app tries to perform actions outside of what it claims to do [6]. This vetting process is not always perfect and indeed security researchers have uncovered ways of circumventing the protections put in place by Apple [7]. In the case of Jekyll [8], the malicious app passed the vetting process by rearranging its code to add new, malicious functionality, after passing the approval process. The iOS kernel uses code signing to ensure that all apps running on a device come from an approved source and have not been tampered with [9]. Additionally, all third-party apps are sandboxed by iOS to prevent them from accessing data stored by other apps and modifying the system. However, Han et al. described how to "break out" of the iOS sandbox by leveraging dynamically loaded, private APIs in malicious apps [10]. Finally, iOS enforces a secure boot chain and file encryption using a per-file key.

2.2.2 Android

Android is a mobile operating system developed by Google and the Open Handset Alliance. Android devices are powered by a healthy app ecosystem providing access to over 1.6 million apps. The core of the operating system is written in C, with additional components written in C++, and the user interface portions written in Java. Like iOS, it is also a Unix-like operating system; however, it features

a monolithic kernel, designed to run on a number of processor platforms such as ARM, x86, and MIPS. In stark contrast to the Apple App Store, the Google Play Store does not require an exhaustive app vetting process before an app is admitted to the store. In general, apps are dynamically tested with a Google security service known as Bouncer [12]. Google automatically scans apps using dynamic analysis and combines the results of this analysis with signals from its reputation engine after it has analyzed the account of the app developer themselves. Security researchers John Oberheide and Charlie Miller demonstrated techniques that could be used to fingerprint Bouncer [13]. They identified unique characteristics of the Bouncer emulation framework such as the hostname, phone number, and Android ID. By checking for these fingerprints, malware can pretend to be benign when being tested by Bouncer and then become malicious when installed on victim devices. In an early study [14], Enck et al. analyzed the source code of 1100 Android apps and found no evidence of malware or exploitable vulnerabilities. Unfortunately, the landscape has deteriorated since then. Indeed, Zhou and Jiang [15] provide a characterization of the evolution of Android malware. On the Android platform, every app runs in its own sandbox by default. As a result of this, each app is isolated from other apps and the system itself, except by using well-defined APIs and system services such as interprocess communication (IPC). However, researchers have found ways for apps to break out of their sandbox and read arbitrary files using symbolic links [16]. Android uses a Linux-like user approach, where each app is executed as a different user and thus inherits the security provided by the operating system in protecting its resources and files. In addition to sandboxing and permissions, Android is also designed to prevent platform modification by malware and also has the capability of remotely removing malware from a device if required [17].

Comparison of Smartphone Operating Systems: Table 2.1 shows a comparison of the similarities and differences between the four most popular smartphone operating systems, and summarizes our effort in distilling this information from the literature [18–23]. For brevity, we do not compare an exhaustive set of features for these operating systems. Instead, we target the main characteristics of the operating systems that contribute the most to vulnerabilities, and thus are most interesting to IDS developers. We look at the OS family, CPU architectures supported, source code model, programming languages used, and reverse engineering tools that are available. Android is based on the Linux family of operating system, while iOS' Darwin and BlackBerry's QNX are Unix-like operating systems. The outlier here is the Windows Phone operating system, which is built around the Windows family of operating systems. All four of these smartphone operating systems are built to run on ARM processors, with Android offering the capability to run on x86 and MIPS processors as well. Android dominates the market, being delivered on 82.8% of smartphones, iOS on 13.9%, and Windows Phone and BlackBerry trailing distantly with 2.6% and 0.3% deployment, respectively, as of 2015 Q3 [2]. Android is the only open-source operating system on the list and all are written in C/C++

Table 2.1 Summary of the Main Characteristics of Android, iOS, Windows Phone, and BlackBerry That Contribute to Their Attack Surface

Operating System	*Android*	*iOS*	*Windows Phone*	*BlackBerry*
OS family	Linux	Darwin	Windows CE-7, Windows NT-8	QNX
Vendor	Google Inc., Open Handset Alliance (and OEMs)	Apple Inc.	Microsoft (and OEMs)	BlackBerry Ltd.
CPU architecture	ARM, ARM64, x86, MIPS	ARM, ARM64	ARM (ARM64 upcoming [11])	ARM
Market share (2015 Q3) [2]	84.7%	13.1%	1.7%	0.3%
Source code	Open	Closed	Closed	Closed
Programming language	C, C++, Java	C, C++, Objective-C, Swift	C, C++	C++
Application store	Google Play Store	App Store	Windows Phone Store	BlackBerry World
Reverse engineering tools	apktool, dex2jar, JD-Compiler, XDA auto tool	iRET toolkit, Windows Explorer, oTool, iExplorer, Class-dump-z	Decompresser, Visual Studio, .NET Decompiler	JD-GUI, VSMTool, COD extractor

or other variants of C. Each of the operating systems is supported by a single official application marketplace, which provides third-party apps to users. Importantly, Android and Windows Phones have OEMs that (sometimes) modify the standard operating systems and unwittingly introduce vulnerabilities [24]. Finally, all four operating systems have a suite of reverse engineering tools available to assist with vulnerability analysis.

Definitions: We now define key terms that are used throughout the chapter.

- *Attack vector*: The means by which an attack is carried out against a system.
- *Exploit*: The method used to take advantage of a vulnerability.
- *Vulnerability*: Any weakness in a system that exposes it to risk.

Threat model: In evaluating the landscape as it concerns intrusion detection and prevention, we systematize adversaries based on their capabilities, goals, and relationship to the smartphone under attack:

- *Local adversary (active/passive)*: This attacker is present on or controls the local network. A local attacker may also be logically adjacent to the device (e.g., spoofing a cell tower) or have close physical access to a device (e.g., in close physical contact with the victim).
- *Remote adversary (active/passive)*: This attacker is present outside of the local network and may control segments of the network between the victim and the destination of their traffic.

Passive adversaries may eavesdrop on and observe traffic from the communication channels in the network. They may also observe data from *side channels*, such as device sensors [25], power consumption [26,27], and wireless transmissions [28,29]. Conversely, active adversaries are able to read, modify, or inject data into a communication channel. Note that malicious app developers (or adversaries who modify/repackage apps) fall into the category of active remote adversary. Our types of adversary are not necessarily mutually exclusive. Indeed, adversaries may change position in the network and more than one adversaries may collude to achieve a more complex objective. The specific target of the adversary may be one or more of

- *The victim themselves*: The adversary is intending to cause harm to the victim and does this by attacking their smartphone to cause loss of data or perform denial-of-service (DoS).
- *The device itself*: The adversary may be intent on exfiltrating personal data from the device such as contacts, credit card information, social security numbers, pictures, or videos. In the case of corporate espionage, the adversary may be targeting the employee of a company to obtain intellectual property, unpublished reports, or other sensitive business data.
- *Device resources*: Data on a device may be immaterial to an adversary who is targeting smartphones to exploit their resources such as storage, processing power, and bandwidth. This is especially common for adversaries interested in "recruiting" devices for a botnet.

For the remainder of this chapter, we frame the attacks and attack vectors in relation to the position and intent of the adversary. This is summarized by the flowchart

of the decision-making process of an attacker shown in Figure 2.1. In general, an adversary has one of three objectives when attacking a smartphone:

- *Perform DoS*: The adversary is concerned with preventing the device from performing its prescribed functionality. This attack is fairly noticeable, since the user will perceive degradation in performance or a missing device (in the case of theft).
- *Utilize device resources*: The adversary is concerned with leveraging the resources (CPU, memory, network access) of a device to further their own goals, for example, recruiting devices for a botnet or as a proxy for launching further attacks.
- *Steal data*: The adversary is concerned with obtaining sensitive data from a device such as user account information, credit cards, multimedia, and sensor data. Note that the sensitive data the adversary is interested in may not yet exist, so the adversary may plant a backdoor, for example, when spying on a spouse.

2.3 Smartphone Attack Vectors

Developing IDSs for smartphones is complicated by the fact that smartphones are devices that communicate over a variety of wireless interfaces/networks and provide a highly customizable and extensible platform. Thus, smartphones will necessarily have a number of areas that must be exposed in order for them to provide their stated functionality. Moreover, what really distinguishes smartphones from other computing platforms is the multitude of sensors they contain and their ultra-high mobility, which makes them susceptible to loss/theft/physical access. The following list distils [30–32] 13 vulnerable areas (or "weak points") on typical smartphones that will continue to be targets for delivering exploits:

- *Browser*: May contain vulnerabilities in parsing web pages, processing Javascript, or providing WebView functionality to apps.
- *Baseband processor*: Smartphones can be tricked into connecting to rogue base stations, which can then attack the mobile radio interface.
- *Messaging services*: Short message service (SMS)/multimedia messaging service (MMS) messages may be used to deliver malicious payloads.
- *Wireless interfaces*: Attackers can attempt to attack a smartphone from any one of the myriad of (noncellular) wireless interfaces.
- *SIM card*: Attackers may be able to manipulate SIM cards to attack a device or steal data.
- *Memory card*: Many smartphones provide slots for external memory cards. These are frequently unencrypted and data can be retrieved if the smartphone or memory card itself is misplaced.

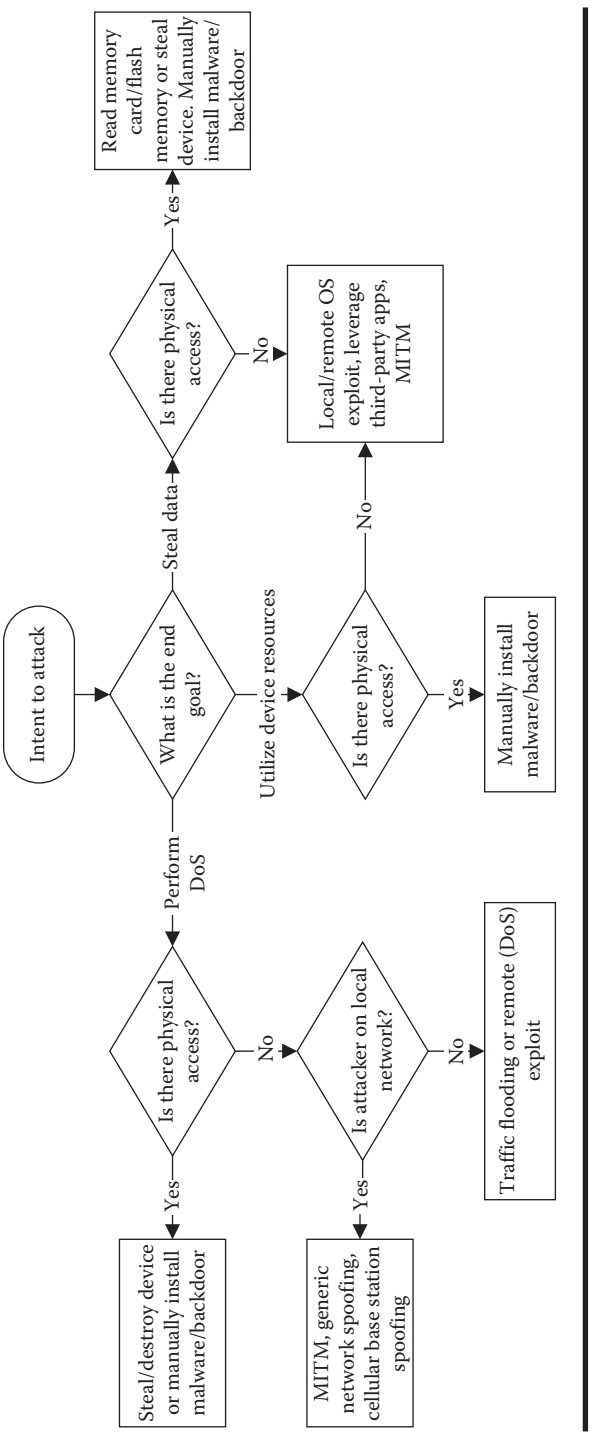

Figure 2.1 Flowchart of a typical smartphone attack from the attacker's point of view.

- *Hardware interfaces/ports*: Smartphones may be vulnerable to attacks coming over their exposed ports, such as USB ports. By opening a device, an attacker can also try to pilfer data through low-level circuitry such as JTAG ports.
- *Operating system*: Adversaries can attack typical weaknesses found in operating systems. In some cases, smartphone operating systems may not be as mature (or robust) as desktop operating systems.
- *Third-party apps*: Third-party apps can access any resource that they have been given permission to access. Additionally, apps can attempt to break out of the OS-provided sandbox. Attackers can also use vulnerable third-party apps as a proxy for conducting further attacks on a smartphone through privilege escalation.
- *Users*: Users can advance attacks if they make bad device configuration choices or are victims of social engineering.
- *Memory*: Physical memory on the device can be modified to remove protective mechanisms from the system.
- *Firmware*: An attacker may target submodule (such as Wi-Fi interface cards) firmware to obtain long-term elevated privileges on the device.
- *Device itself*: An attacker may target any one of the side channels coming from the device itself, for the purposes of device fingerprinting or recovering sensitive data such as encryption keys or screen-lock codes.

We systematize attack vectors as belonging to either of four categories: drive-by attacks, app ecosystems, physical attacks, and social engineering [31,33,34]. In detailing attack vectors, we use *italicized text* to denote the vulnerable areas affected. Also, we only briefly identify some attacks to put the attack vectors into perspective. We leave a detailed treatment of all the major attacks against smartphones for Section 2.4.

2.3.1 Drive-by Attacks

Vulnerable areas affected: Browser, baseband processor, messaging services, wireless interfaces, operating system, third-party apps, users.

In the case of drive-by attacks (or watering hole attacks), an attacker attempts to exploit existing bugs in the software running on the smartphone that processes external data. A common method of delivering drive-by attacks involves exploiting vulnerabilities in the *browser* on the smartphone to make it execute a malicious payload. These attacks can be carried out *en masse* since popular web pages can be compromised and laced with malicious payloads. Alternately, links to malicious pages can be sent to *users* through traditional channels such as email, *messaging services*, and social media. An attacker can also manipulate unencrypted HTTP pages to insert malicious payloads or take advantage of improperly handled SSL/TLS (in *third-party apps*) to perform a man-in-the-middle (MITM) attack [35] and insert the malicious payload that way. Note, however, that attacks targeting smartphone

browsers have more limited potential than desktop browsers due to application isolation via sandboxing.

Drive-by attacks can also leverage any one of a device's *wireless interfaces*. Various spoofing attacks can be performed against Wi-Fi and Bluetooth, and indeed, base station spoofing can used against the *baseband processor*. Other targets of drive-by attacks may include WebViews (*operating system*), a user interface component in smartphone development frameworks that allows apps to easily render web pages from within the app. A WebView can be used to provide an interface to a Javascript component and thus external Javascript can be executed on a device. Additionally, WebViews can also allow a website to access data stored on devices. If an attacker is able to intercept or modify the content of a URL that is loaded by a WebView (using an MITM attack or cross-site scripting), they can use functions from within the WebView framework to access data from the device. Worryingly, privilege escalation exploits have been published that allow arbitrary code execution on vulnerable devices through WebViews [36]. Another class of attacks, known as component hijacking attacks [37], leverage the drive-by attack principle to access private data and spoof intents.

Drive-by attacks may exploit network services, pieces of software running on a device that open ports to listen for incoming connections. Traditionally, network services only run on devices acting as servers, such as web (HTTP) servers listening on port 80. In the smartphone landscape, however, to satisfy the great need for interconnectivity, mobile devices can be found running network services such as Android Debug Bridge (ADB) [38], Virtual Private Network (VPN), Virtual Network Computing (VNC), Remote Desktop (RDP), and Secure Shell (SSH) services. In the case where smartphones are configured to share their Internet connection through a mobile hotspot, they can be expected to also run Dynamic Host Configuration Protocol (DHCP) services and act as a default gateway. These additional services all increase the number of avenues for exploit. Network services offer an attractive interface for attackers to attempt to exploit, since they provide a (usually) always open entrance that is accessible via the network. Exploiting network services is also particularly attractive to an adversary since no user intervention is typically required to allow the exploit to take place and after successful exploitation there may be no immediate indication to the user that an attack has indeed happened. By default, smartphones may not have any network services installed, but there are a wide variety of third-party apps that users install, which offer additional functionality that requires the use of network services. Indeed, Nielson reports that the average user uses 26.8 different apps per month [39], and any of these could potentially leverage network services.

Drive-by attacks, while successful on traditional workstations, may have more limited impact when translated to the smartphone arena. On Android, most software is implemented in Java and executed by the Dalvik Virtual Machine. This mitigates some of the typical attack strategies (such as buffer overflows) since low-level data structures are protected by boundary checks. However, many Android apps also

leverage libraries implemented in native code; thus, some parts of many apps continue to be susceptible to traditional attacks against memory corruption bugs. These attacks are quite dangerous since they can lead to code execution on the device [40], with the user not necessarily knowing that they have been compromised.

2.3.2 App Ecosystems

Vulnerable areas affected: Third-party apps.

By and large, user installation of grayware/malware is limited due to the use of "trusted" software repositories such as the official app stores. App stores and smartphone operating systems utilize strong technical mechanisms to ensure a restriction on the *third-party apps* that can be installed on a device. Attacks coming from app ecosystems leverage the fact that if grayware/malware can be placed in a marketplace, it can quite quickly be available for infecting the entire ecosystem. Additionally, grayware/malware authors are incentivized by the fact that users are typically more trusting of apps if they find them in the official app marketplaces.

As mentioned in Section 2.2, app stores employ various degrees of vetting before allowing an app to become available for the general public to download. Additionally, smartphone operating systems restrict, by default, the "sideloading" of apps, that is, installing apps to a device through unofficial channels. For this reason, most grayware/malware affecting smartphones are delivered as Trojan-horse apps via an app store. Thus, malicious authors must develop apps with some functionality, but containing malicious payloads hidden from app store vetting using timebombs, dynamic code loading, reflection, code obfuscation, and/or IP address checking (to determine whether the app is being run through an app store's vetting engine). Recently, the BrainTest trojan [41] utilized all the aforementioned strategies to evade app store detection.

One strategy used by grayware/malware authors, and most commonly observed in third-party app marketplaces, involves the repackaging of legitimate apps to inject malware [15], which can then attempt to exploit the *operating system/firmware* or steal data from the *memory card*. Fraudsters have also been known to modify the advertising portions of legitimate apps to insert their own code. This allows them to fraudulently obtain revenue from a legitimate app [42]. Other less malicious apps (and their included libraries) have been known to leverage additional and unnecessary dangerous permissions, ostensibly to have greater access to sensitive data and resources, which can then be used for profiling a user [43–45] for reasons such as better advertisement targeting, or more maliciously, selling user data directly to other third parties.

Smartphone worms are much more limited than Trojan-horse apps but may begin to see wider adoption with the availability of operating system exploits propagated by modern smartphone connectivity features such as portable hotspots/NFC and even older channels such as Bluetooth/SMS/MMS/WAP. Operating system protection mechanisms, such as SELinux, offer mitigation for system exploits by

enhancing the boundaries of app sandboxes [46] and thus worms may have more limited success.

2.3.3 Physical Attacks

Vulnerable areas affected: SIM card, memory card, hardware interfaces/ports, memory, firmware, device itself.

One class of physical attacks come about from dismantling the *device itself* and/or being able to connect to and interface directly with the *hardware interfaces/ports* on the device; we call these *physical (tampering)* attacks. Physical attacks may also make use of side channels that enable the inference of private data located on a device; we call these *physical (general)* attacks. We expand on specific physical attacks in Section 2.4.1, but right now we enumerate general physical attack approaches:

1. Accessing a device that does not use a screen-lock and transferring the data from the device using copy/paste/attach features within the operating system.
2. Accessing *memory cards* within the device itself and removing them to obtain data that was stored on the device.
3. Inferring PIN/screen-lock codes from smudges on a smartphone touchscreen [47].
4. Leveraging ports, such as USB ports, on the device to perform further attacks [48–50].
5. Modifying physical *memory* chips on the circuit board to introduce new software and/or affect the *firmware* of low-level hardware (such as Wi-Fi adaptors).
6. The *SIM card(s)* in a device can be removed to retrieve sensitive data such as messages and phone numbers. Malicious payloads can also be written to a SIM card.

An attacker can leverage their access to the *hardware interfaces/ports* of a device to place malware or other data on the device or execute commands. Lau et al. demonstrated how it was possible to install arbitrary apps on an iOS device through the USB port [49]. The ADB can also be used to launch attacks. The ADB is a command line tool that can be used to connect to and run commands on Android devices using a desktop.

Another class of physical attacks comes from leveraging the physical state of a device or physical access to the device to attack it. Leveraging the physical sensors on a smartphone is an example of utilizing the physical state of a device to enable attacks. The literature exemplifies using the accelerometer/gyroscope [25,51], and light sensor [52] to steal device credentials/passwords.

Attackers can also leverage physical access to a device to attempt to pass the "lock screen," provided that screen-locking is enabled in the first place [53]. A locked device is usually guarded by PINs, patterns, and passwords, and more

recently, using biometrics such as fingerprints. An adversary being able to successfully unlock a device depends on the complexity of the credential used to lock the device. Approaches as rudimentary as looking at screen smudges have demonstrated potential in assisting attackers to bypass locked screens [47]. Biometric approaches, which show much promise, have been shown to be dangerous if implemented incorrectly [54], with the end result being a potential compromise of a user's biometric data such as a fingerprint. Needless to say, the compromise of a user's biometric is a serious problem, as by nature it cannot be replaced.

2.3.4 Social Engineering

Vulnerable areas affected: Browser, operating system, users.

With social engineering, the user of a smartphone is tricked into revealing credentials or performing actions that assist the attacker in furthering their attack. These attacks are dangerous in that they employ nontechnical strategies to elicit private information from users and, as such, generic IDS solutions to address social engineering are not available. The problem of social engineering is exacerbated by the fact that users may not know that they have been successfully attacked until long after the fact, if at all. Three common social engineering attacks specific to smartphones are

1. *Making malicious apps look like legitimate apps*: Malware/grayware authors typically build clones of popular applications to trick a user into installing their version because it has a name and description very similar to the app they actually want [42,55].
2. *Enticing users using device-specific details*: Smartphone users may be tricked by ads and web pages that give them advice specific to their device make and model. Attackers commonly use the User-Agent sent by a browser/app to identify the device before sending customized messages to the user about faults with their specific device such as poor battery or malware infections. The users are then led to download malware, which supposedly solves their "problems" [56].
3. *Malware pretending to be a second factor of authentication*: Desktops infected by the Zeus malware may instruct users to download an authentication component to their smartphone as a second factor of authentication when the user attempts to log in to their online bank [57]. The malware then captures a user's bank login credentials.

Aside from social engineering that leads to malware installation, other typical social engineering attacks that result in the user giving away their credentials are just as detrimental as on traditional desktops. Especially considering that a user may be logged into several services from their smartphone at the same time, social engineering presents a high-reward attack vector to adversaries.

Table 2.2 Attack Vectors and What Vulnerable Area on the Smartphone They Target

Attack Vector	*Vulnerable Area Affected*
Drive-by attacks	Browser, messaging services, wireless interfaces, SIM card, memory card, operating system, third-party apps, users, memory, firmware
App ecosystems	Third-party apps
Physical attacks	Baseband processor, SIM card, memory card, hardware interfaces, USB, memory, firmware
Social engineering	Browser, operating system, users

Table 2.2 summarizes the relationship between the attack vectors and the vulnerable areas that they target. From the table, it can be seen that drive-by attacks have the potential to affect the most areas on a smartphone. This is perhaps unsurprising, as drive-by attacks are made possible by bugs/vulnerabilities in the software on the smartphone itself, and thus there is a rich attack surface that can be targeted. Physical attacks have the second largest number of vulnerable areas and target weaknesses in the physical hardware/characteristics of the smartphone. App ecosystems and social engineering target fewer vulnerable areas directly, but can be used as a proxy for delivering more dangerous drive-by exploits if users are tricked into installing apps or performing particular actions on their device.

Table 2.3 shows common attack vectors, the level of sophistication required to achieve success, and the potential effect of device compromise. The level of sophistication refers to the technical expertise required from the attacker and ranges from low (minimal technical ability required), medium (moderate technical ability required, with published exploits easily available/adaptable), to high (advanced technical ability required, usually requiring the development of zero-day exploits or advanced reverse-engineering skills). The effect of compromise ranges from low (information disclosure or minor annoyance), medium (low + greater annoyance

Table 2.3 Attack Vectors and Their Main Characteristics

Attack Vector	*Level of Sophistication*	*Effect of Compromise*
Drive-by attacks	Medium/high	Low/medium/high
App ecosystems	Low	Low/medium
Physical (tampering)	High	High
Physical (general)	Low/medium	Low/medium
Social engineering	Low	Low/medium

and potentially costing the user money, e.g., premium rate SMS/calls), to high (full compromise of the device with unfettered access by the adversary). The main insight from Table 2.3 is that social engineering requires minimal skill and has the potential to affect many victims, but the effect of the attack is typically low. Drive-by attacks, on the other hand, can prove to be very effective to attackers since published exploits are available (especially for older devices that are still widely used [58]) and can yield good returns in terms of the effect of compromise while targeting a moderate number of victims. Physical tampering of devices requires high sophistication by adversaries but may yield significant rewards and are usually employed at the nation-state/law-enforcement level. Worryingly, unsophisticated attackers can combine social engineering with published drive-by exploits to obtain a significant return on investment, especially if attacks target users with older devices.

2.4 Smartphone Attack Hierarchy

We classify smartphone attacks based on the position of the attacker in the "space" relative to the smartphone under attack as follows:

- *Physical versus nonphysical*: As shown in Figure 2.2, the first level of differentiation is whether the attack is performed by physically accessing the device. This is a logical separation of attacks as it broadly divides attacks into those that require tangible access to a device as opposed to those that access the device in an intangible way. IDS developers will typically focus on nonphysical attacks. Nonphysical (or intangible) attacks can be further separated into two categories: local and remote.
- *Local versus Remote*: Local and remote refer to the logical proximity of the attacker to the victim device in terms of location on the network. Broadly speaking, local attacks are carried out by attackers that are on the current local

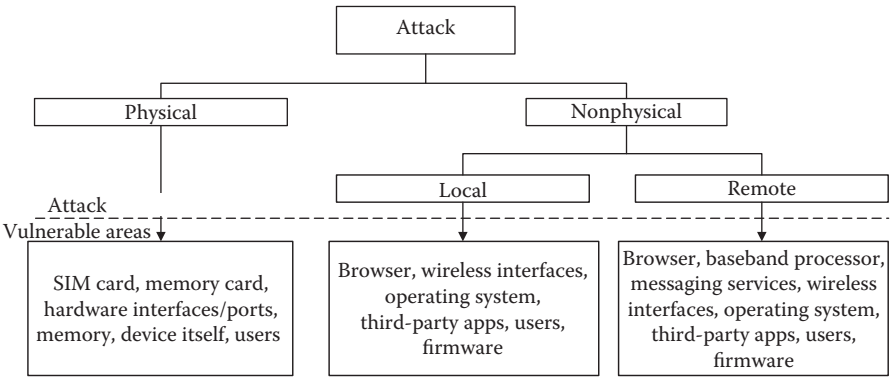

Figure 2.2 Taxonomy of smartphone attacks.

area network (LAN) segment (or otherwise logically adjacent to the victim) and includes well-known attacks such as Address Resolution Protocol (ARP) spoofing, MITM, and traffic analysis attacks. Remote attacks are carried out by adversaries who are able to launch attacks from beyond the local network segment, that is, more than one network hop away.

- *Interactive versus noninteractive*: Attacks can also be categorized into whether they are interactive or noninteractive. By interactive and noninteractive, we mean whether the smartphone user is required to perform a particular action on their smartphone for the attack to be successful. In general, noninteractive attacks are more attractive (and more difficult to exploit) since no user intervention is required and, as a result, may be more stealthy.

In Sections 2.4.1 and 2.4.2, we compare typical attacks that fall into the categories of physical and nonphysical, to assess their characteristics relative to each other and gain an understanding of the motivations of attackers for using one type of attack over another. Table 2.4 provides a compendium of examples of attacks for all the exploits mentioned in this section.

2.4.1 Physical Attacks

Physical attacks are carried out by attackers that target the hardware of the device itself. In other words, these attacks require attackers to physically touch the device in order to carry out their malfeasance. The main classes of physical attacks are: hardware tampering, attacking the device over its built-in ports, and leveraging physical sensors on the device to garner data.

2.4.1.1 Hardware Tampering

Hardware tampering attacks are directed at the physical circuitry of the device itself. Security of hardware is often an afterthought since many manufacturers consider the device hardware secure through obscurity. Tampering with hardware requires esoteric knowledge and a particular skillset, but common low-level interfaces and circuitry on most electronic devices allow attacks to be performed against a wide range of devices. For example, many electronic devices, when disassembled to the circuit board level, will have exposed serial and JTAG ports. These ports can be used to intercept debug messages, send commands, or flash the firmware of the device. Serial and JTAG interfaces are widely used for communication between submodules in embedded systems and an attacker with reasonable skill and patience can usually find ways of accessing these buses. By being in physical possession of or in close proximity to a device, an attacker may also leverage data gleaned from any of the physical side channels on the device such as power consumption or electromagnetic emanations. By leveraging side-channel information, attackers can cheaply [59] determine secret keys from a device's embedded circuitry [27].

Table 2.4 Examples of Common Attacks against Smartphones and Their Characteristics

Name	*Vector*	*Vulnerable Area*	*Local/Remote*	*Interactivity*	*Examples*
Man-in-the-middle attacks	Drive-by	Browser, third-party apps, baseband processor, wireless interfaces	Both	Noninteractive	SSL MITM on apps [35,66]
Network spoofing	Drive-by	Wireless interfaces, browser, operating system	Local	Noninteractive	DNS spoofing [67], DHCP spoofing [68], ARP spoofing [69,70], Wi-Fi attacks [67,71]
Base station spoofing	Drive-by	Baseband processor, messaging services	Local	Noninteractive	Cell tower spoofing [72]
Network service attacks	Drive-by	Operating system, network services	Both	Noninteractive	ADB exploits [38], SSH exploits [73]
SMS/MMS/WAP attacks	Drive-by	Messaging services	Remote	Both	Stagefright [74–76], iOS SMS bugs [77], sending USSD codes via WAP [78]

(*Continued*)