Toward Risk-driven Security Measurement for Android Smartphone Platforms

Reijo M. Savola, Teemu Väisänen, Antti Evesti, Pekka Savolainen
VTT Technical Research Centre of Finland
Oulu, Finland

Juha Kemppainen, Marko Kokemäki
Elektrobit Wireless Communications Ltd.
Oulu, Finland

Abstract—**Security for Android smartphone platforms is a challenge arising in part from their openness. We analyse the security objectives of two distinct envisioned public safety and security mobile network systems utilising the Android platform. The analysis is based on an industrial risk analysis activity. In addition, we propose initial heuristics for security objective decomposition aimed at security metrics definition. Systematically defined and applied security metrics can be used for informed risk-driven security decision-making, enabling higher security effectiveness.**

Keywords-Android; security metrics; risk analysis; security effectiveness

I. INTRODUCTION

Android has become the world's most widely used smartphone platform. Security management for Android-based platforms and applications poses a considerable challenge on account of, among other factors, the system's openness. Attacks of various types render it possible to compromise an Android device and potentially information systems to which there it has connections. Sufficient management of configuration and system quality is especially important for Android's security performance.

As software-intensive systems incorporate increasingly critical applications, grow more difficult to manage, and utilise ever more complex and networked software, they become exposed to security risks in increasing numbers [1]. Security engineering is challenging, requiring expertise in several domains, including business management, systems and software engineering, and risk management.

Systematically designed and managed *security metrics* increase understanding of the security effectiveness (SE) level of the target system. Security effectiveness is the assurance that the stated *security objectives* (SOs) are met in the target system and the expectations for resiliency in the use environment are satisfied, while at the same time the system does not behave other than intended [2–4]. Security objectives are high level statements of intent to counter identified threats and/or satisfy the organisational security policies and/or assumptions identified [5].

Quantification and decomposition techniques are widely used in engineering to enable informed decision-making [1]. If these metrics' design is based on prioritised results of iterative risk analysis, they support well decision-making aimed at sufficient security effectiveness of the system. Evidence of configuration correctness, system quality, and adequate implementation of security controls aid in systematic management of Android security.

The main contribution of this study is in analysing security risks of the Android platform in public safety and security (PSS) mobile network applications, and their interdependencies, and in proposing initial heuristics for the decomposition of the SOs, with the objective of development of security metrics that address these scenarios.

The paper continues with Section II discussing the study's background, including the role of risk analysis and security considerations for the Android platform. Section III describes the risk analysis process used and presents prioritised results from its application. Section IV discusses high-level security objectives based on the risk analysis results, and Section V proposes initial heuristics for deriving security metrics from the objectives. Section VI examines related work, and Section VII offers concluding remarks and discusses future research questions.

II. BACKGROUND

A. *Iterative Risk Analysis*

Sufficient knowledge of security risks is a pre-requisite for all security-critical engineering and management activities. Effective and efficient SOs are contingent on systematically acquired target-specific risk knowledge of high quality. SO definition and security metrics development activities should be based on careful, sufficiently detailed risk analysis (RA) of target cases.

The RA should be iterative – carrying it out in several, iterative phases increases the risk knowledge's quality considerably. For example, telecommunications company Ericsson has defined and carries out an iterative RA process comprising three iterative instances of RA sessions: (i) RA1, is conducted when product requirements are defined, (ii) RA2, when the product is being specified, and (iii) RA3, when the product is being designed and verified.

The main focus in RA1 is on the points where risks reside in a business value chain, while RA2 focuses mainly on analysing the risk environment from a product or solution feature perspective, and RA3 on verifying how the risks identified have been mitigated and what the residual risks are [6].

B. Security in Android Platforms in General

The Android platform encompasses several security solutions [7] that originate mainly from Linux. Management of memory, processes, users, and access control permissions is provided by the Linux kernel but modified from traditional desktop usage: applications have unique user names and run in a virtual machine, and permission-based file system sandbox, while in desktop environments many processes can share groups or a user ID, usually the user ID of the user of the device.

One challenge with Android's modified Linux kernel is the upgrade process: In several older versions, the deployed kernel's version is out of date, and people are using old or other devices whose Android version is no longer upgraded by the device manufacturer. For example, a critical vulnerability in the Android security model was discovered by Bluebox Security in February 2013 [8]. The vulnerability allows a hacker to modify code of Android application package (APK) files without affecting the application's cryptographic signature, e.g., to turn any legitimate application into a malicious one. This vulnerability has existed in Android at least since the release of version 1.6, so it affects a huge number of devices. In July 2013, ZDNet reported that Google had found a fix for the vulnerability and begun to provide patches for the relevant versions of Android [9]. Bluebox Security has released an application [10] for checking whether devices exhibit the vulnerability. Sadly, it is possible that manufacturers of old devices will never apply the associated patch to them.

According to Google [11], on 3 June 2013 only four per cent of Android users were running the latest version (4.2.x) of Android. The figure on 8 July was 5-6%. Data are collected from each device when the user visits the Google Play Store. Google describes this as a better approach than collecting data when the device checks in with Google servers, as done before April 2013. It is possible that people with newer versions of Android visit Google Play Store more often than do people with older phones, so it would be good to have also other methods of data collection than via visit the Google Play Store. Possible effects of using old operating system (OS) version have been analysed by, for example, Juniper Networks [12].

Even if each application has its own sandbox and should have access to system resources (like the screen, wireless connections, contacts, text messages) only through application programming interface (API) methods provided by the virtual machine, possibilities have been demonstrated [13] for attacking the system by means of Advanced Reduced Instruction Set Computing Machines (ARM) native code that is called from applications and executed outside the virtual machine.

Understanding system resource permissions might be difficult for users. When installing an application, the user sees the required permissions but only two options: to let the application access all desired resources and not to install the application. In addition, there have been [14] and still are [12] several malicious applications in Google Play and other marketplaces.

Android is Open Source code: anyone is able to discover bugs and vulnerabilities by consulting the code directly. This cuts both ways: some entities report or even fix the errors found, while others exploit them for malicious purposes.

III. THE RISK ANALYSIS PROCESS AND ITS RESULTS

Below, we explain the RA process used for gaining the risk knowledge in the two Android platform cases and overview its overall results. The RA described here can be seen as an activity related to RA1 in the iterative RA process discussed above.

A. The Systems under Investigation

The purpose of the RA was to analyse the security risks of two envisioned PSS mobile network systems using the Android platform, Case 1 and Case 2. Case 1 was normal application of the Android platform for PSS mobile network use, and Case 2 security-critical application of it for the same use. Because of the different usage seen in these two cases, RA was performed separately for each. However, in general, the results for Case 1 yield basic-level considerations applicable for Case 2.

PSS mobile networks employ a mobile infrastructure similar to that of cellular networks. However, they incorporate dispatcher stations for managing the communication of the user groups. User terminal devices are essential parts of the system [15]. The main services and characteristics of PSS mobile networks are group communication, encryption services and high availability [16]. Fig. 1 visualises the use of PSS networks considered here.

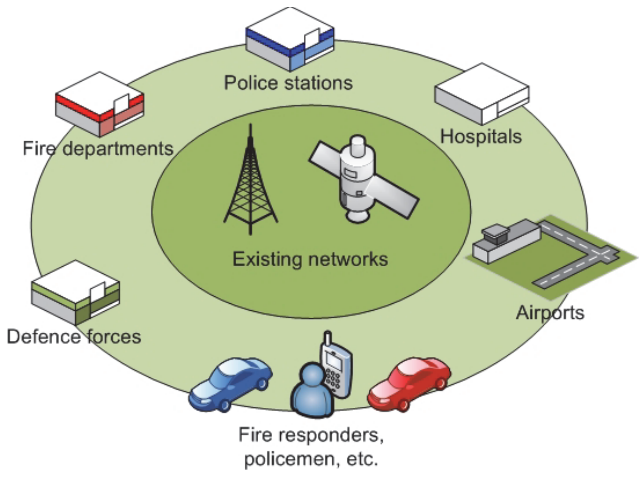

Figure 1. Use of the systems under investigation.

B. The RA Process Utilised

The RA was performed in co-operation between security researchers from VTT Technical Research Centre of Finland, and Android experts with Elektrobit Wireless Communications Ltd. The RA process consisted of two main stages: risk identification and risks' prioritisation. The latter stage included (i) severity and probability scoring and (ii) priority ordering of risks.

The process started with a brainstorming meeting of risk identification experts wherein participants were divided into two teams – both comprising persons from each organisation with enough expertise in Android technology and security. First, one team concentrated on Case 1 and the other on Case 2. The teams were asked to identify as many security risks as possible. In this phase all risks were listed without critique. After a time, the teams switched cases to investigate. Therefore, two risk sets were obtained for each case. Next, both teams categorised these risks and presented the results for the full group.

After this, the brainstorming was continued in a plenary session. The risk sets presented were combined via removal of duplicates and merging of risk categories. Naturally, the risk identification process yielded thoughts on threats and attack types also; this material was separated from the risks and utilised later on.

Finally, each risk's probability and the severity of its consequences were rated. Consequently, the first meeting produced a list of risks, probabilities, and severity estimates for consequences, alongside a set of threats and attacks.

During the brainstorming meeting, risks were written on to sticky notes, to facilitate categorisation. However, after the meeting the results were transcribed and took digital form.

In the last stage, the 'raw' prioritisation results were ordered in view of expert opinions. Expert knowledge was needed at this point, because the risks' abstract nature calls for case-by-case decision-making on whether severity or probability is more important. In general, severity was stressed a bit more because PSS mobile networks are safety- and security-critical. It should be noted that risk prioritisation is not unambiguous, and small changes in system assumptions can change it.

C. *Overview of Prioritized Risks*

We now present an overview of the RA results. Table I lists the prioritised risks for Case 1, and Table II presents those for Case 2. The risks in the tables are at a rather high level, but some important risks are listed at the detail level, especially for Case 2.

The rank of each risk is shown by the number in the first column. In the tables, 'S' refers to the severity of the consequences if the risk is actualised, and 'P' denotes the probability of the risk being realised. The scale for each is 0-3, with increments of 0.25. The former number represents no risk, and the latter indicates extremely high severity or probability.

In the tables, 'R:', meaning 'risk arising from', is used in connection with *attack types*, *vulnerabilities*, and *faults* that cause a risk. Most of the risks listed here are of these types.

All of the risks listed can be seen as important. However, further investigations, such as SO definition, should be carried out in priority order. It should be noted that definition of SOs and security controls is not a one-to-one mapping to the list of risks. For example, access control can be used to mitigate several risks, and some risks contribute to others. These interdependencies can be complex.

Even though there many interdependencies among risks, they should be listed in the manner shown in the tables. Otherwise, information about the prioritisation can be lost easily. This information is highly relevant for application of the proper emphasis in SO definition.

One should analyse the interdependencies of the risks to enable efficient and effective SO definition.

TABLE I. PRIORITISED RISKS FOR CASE 1

R#	Description	S	P
1	R: unauthorised input of falsified data	3.00	2.00
2	R: unavailability of the PSS Network at a critical moment (DoS, denial of service)	3.00	1.00
3	R: unauthorised root access	2.75	1.25
4	R: malicious loading of remote code	2.75	1.00
5	R: critical security functionality deployed in software (SW) but designed for hardware (HW)	2.25	3.00
6	R: network shut down due to device problems	3.00	0.50
7	Loss of life due to lack of resuscitation	3.00	0.50
8	R: activation of dormant malware at a critical moment	2.50	1.00
9	R: investigation of the target device in a laboratory environment	2.50	1.00
10	R: utilisation of open interfaces for attacks	2.25	1.00
11	Political risk	2.75	0.50
12	Terrorism risk	2.50	0.50
13	R: spoofing directed at the context system	2.00	1.50
14	Information disclosure utilising a stolen device	2.00	1.00
15	Activism risk	2.25	0.50
16	R: man-in-the-middle attacks, or use of a falsified base station	1.75	1.00
17	R: the use of third-party SW	1.75	1.00
18	R: a poor usability to security ratio	1.75	1.00
19	R: lack of position-based locking	1.50	0.75
20	Information disclosure between applications for work and personal use	1.25	0.75
21	R: poor temporary file management and buffer overflow	1.25	0.75
22	R: poor device management	1.00	0.50
23	R: an individual's unauthorised use of a device	0.75	0.50
24	R: lack/non-functioning of remote wiping when a device is lost or stolen	0.50	0.25
25	R: remote wiping by an unauthorised party	0.50	0.25
26	R: Android vulnerabilities	0.25	0.25

TABLE II. PRIORITISED RISKS FOR CASE 2

R#	Description	S	P
1	R: professional attack with unrestricted resources	2.75	2.50
2	R: paralysis of communication at a critical moment	3.00	2.25
3	R: inappropriate testing	2.00	3.00
4	R: unauthorised root access	2.50	1.50
5	R: lost devices (administrators do not know where all devices are)	2.25	2.50
6	R: jamming (network DoS)	2.75	1.00
7	R: poor key management in a device	2.75	1.00
8	R: insider attacks due to human corruption	2.50	1.50
9	R: phone eavesdropping	2.50	1.00
10	R: phone recording and records' disclosure	2.50	1.00
11	R: eavesdropping through use of device's camera or microphone	2.50	1.00
12	R: weaknesses in the key management	2.50	1.00
13	R: poor configuration management	2.25	1.5
14	R: expiry or insufficient speed of support for the OS	2.00	2.25
15	R: a device's utilisation in one's leisure time	2.00	2.25
16	R: execution of unsigned SW via swapping of central processing unit (CPU) or security chip	2.25	1.25
17	R: improper SW update management causing the wrong updates	2.00	1.25
18	R: attacks via device interfaces (serial / debug port)	2.75	0.75
19	R: communication between networks with different classifications	1.75	1.75
20	Unauthorised information disclosure (calendar, contacts)	2.25	1.50
21	R: vulnerabilities in the used SW libraries used	1.25	2.50
22	R: vulnerabilities in content pages	1.75	2.50
23	R: malware	1.25	2.25
24	R: lack of user authentication	2.25	0.75
25	R: lack of OS management	2.00	1.00
26	R: security shortcomings in the Android OS	1.00	2.50
27	R: modifications of configuration files	1.75	1.50
28	R: HW bit errors	1.75	1.50
29	R: the presence of Internet access	1.00	2.00
30	R: access to source code (Open Source vs proprietary code)	1.25	2.50
31	R: reverse engineering (access to binaries)	1.25	1.00
32	R: acceptance of insecure protocols	1.50	1.00
33	R: accessibility of secure networks	0.75	2.25
34	R: improper key/access management	1.50	0.50
35	R: execution of Android binaries on a computer	1.25	0.50
36	R: a program being modified after installation	1.25	0.50

D. Interdependencies

For example, critical risk R1 (unauthorised input of falsified data) in Case 1 can contribute to R7 (loss of life). Even though both of these risks are critical, the criticality of R1 can be seen as even more emphasised than before investigation of the interdependencies. In general, those risks without the symbol "R:" are higher-level than the others. Consequently, many other risks converge to there. These component risks have a more emphasised position because of this interdependence.

R24 in Case 2 (lack of user authentication) may enable R20 (unauthorised information disclosure). In this scenario, the latter risk is of higher priority. However, mitigating it requires that the first risk mentioned too be mitigated. Accordingly, it is not enough to select controls only for those risks assigned highest priority.

On the other hand, a particular risk may enable several other risks. For example, the risks 'unauthorised input of falsified data' (R1) in Case 1 and the risk 'inappropriate testing' (R3) in Case 2 are able to cause various unexpected risks. Consequently, the interdependencies of risks have to be taken into account if one is to perform SO and security control definition properly. Another example in Case 2 is 'paralysis of communication at a critical moment' (R2); this risk can be realised because of, for example, 'jamming' (R6). It should be noted, however, that the 'raw' prioritisation presented in the tables has value in the determination of the relative importance of different SOs.

Managing interdependencies in tabular form is a laborious and error-prone task. Therefore, an appropriate support tool or visualisation technique is required. For our cases, we have utilised graphical presentations to visualise risks and their mutual dependencies. The graphical presentation facilitates risks' categorisation and the recognition of interdependencies.

Finally, the interdependencies recognised are useful during the controls' selection. The security control selected for any specific risk can mitigate other risks also. As an example, consider R14 (information disclosure utilising a stolen device) and R22 (poor device management) in Case 1: defining controls for the latter risk mitigates also the former risk.

IV. SECURITY OBJECTIVE CONSIDERATIONS

We now turn to how SOs should stem from the RA results. In general, SOs are defined in view of prioritised RA results, and security controls (or 'countermeasures') are based on SOs. It should be noted that here we aim for high SE, enabling definition of metrics capable of expressing this. In practice, though, the SOs are based on risk management (RM) decisions: RM can choose for a risk to be mitigated, cancelled, or accepted.

The main SOs should mitigate the top-ranked risks. In general, many risks can be grouped together under high-level SOs corresponding to use of specific security controls.

A. Case 1

An SO related to R1 in Case 1 requires sufficient authentication and authorisation (AA), confidentiality, and integrity. Availability of the network as a whole is given focus for the SO related to R2 (DoS risks). In addition to R1 (unauthorised input of falsified data), AA objectives are needed for mitigation of R3 (unauthorised root access), R5 (related to critical functionality in SW), R16 (man-in-the-middle risks), R20 (work vs personal use), R23 (unauthorised device use), and R25 (unauthorised remote wiping). Items R4 (to do with malicious remote code), R5, R6 (network shutdown), R12 (terrorism), R16, R19 (lack of position-based locking), R21 (poor temp-file management and buffer overflow), and R24 (lack of remote wiping) imply also integrity solutions. As an example, Table III lists some authorisation SOs, based on the work in Source [17].

TABLE III. EXAMPLE AUTHORISATION SOs, BASED ON [17]

Objective	Description
Authorisation policy	The authorisation policy identifies specific users, user groups, and types of users; specifies the operations permitted and authorisation levels for authorisation objects and specifies the delegation privilege and depth.
Access control mechanism	Access control can use different mechanisms depending on contextual requirements.
User authorisation	The Authorization Manager should verify user identity (ID) and grant access to the resource allowed, with sufficient device authorisation.
Device authorisation	The Authorisation Manager should verify device ID and grant access to the resource allowed, with sufficient user authorisation.
Authorisation objects	Authorisation objects are communication channels, connections, message content, and alarms. Authorisation rights can be granted for individual authorisation objects.
Delegation of privileges	It should be possible to delegate security credentials to an entity so that it can act on the delegators behalf. The chain of delegation generally is three levels deep, except in certain situations.
Revocation of authorisation	The authorisation functionality should support revocation of authorisation from, for example, users identified as harmful.
Change of authorisation requirements	In specific situations, e.g. cases of suspected intrusion the authorisation and the security effectiveness of underlying authentication requirements should be increased.

As is discussed above, in Case 1, R7 (loss of life due to lack of resuscitation) can result from other risks. Consequently, sufficient access control, confidentiality, integrity and availability solutions are needed to mitigate this risk. Moreover, there should be relevant SOs at policy level to make resuscitation more efficient and effective. For example, suitable alarm functionality, visualisation functionality, and an easy-to-use user interface should be introduced to the system, enabling good situation-awareness for critical tasks.

The political (R11) and the terrorism risk (R12) depend largely on other than technical issues; they are affected by national and international political issues and tensions. However, the system should not make the task of attacking the system too easy by neglecting controls for any of the main risks. The activism risk (R15) depends on attracting vulnerabilities and on issues of poor management.

The two risks of information disclosure listed, R14 (by a stolen device) and R20 (between work and personal applications), require different security controls. Remote wiping is a basic security control for R14. Remote wiping related risks R24 and R25 obviously contribute to R14. Poor temp-file management and buffer overflow (R21) requires strong authorisation management and integrity solutions, along with adequate policy definition and enforcement.

There is a fundamental need for SOs ensuring adequate configuration management. For example, R6 (network shutdown), R8 (activation of dormant malware), R22 (poor device management) and R26 (Android vulnerabilities) can be mitigated by enough configuration correctness. Specific security metrics focusing on configuration correctness support this task.

A sufficient usability to security ratio (R18) is one of the core design goals. The target systems will be deployed in operations wherein usability is very important. Decisions on trade-offs between security effectiveness and usability are needed, with support from adequate metrics depicting both dimensions.

B. Case 2

In general, the RA results for Case 2 emphasise more detail-level issues than do the results for Case 1. This is understandable, since Case 2 was more security-critical. Moreover, more risks are listed for Case 2 than Case 1.

Even if the focus is closer in Table II, it should be understood that the risks listed there can contribute to high-level risks R7 (poor key management in a device), R10 (phone recording and records' disclosure), R11 (eavesdropping) and R15 (device utilisation in one's leisure time) in Case 1 too, even though not mentioned in connection with that case. In addition, all risks listed for Case 1 are relevant to Case 2 as well. In practice, the risk compilation in Table I offers a basic-level RA for Case 2, also requiring attention. However, the prioritisation order of Case 2 is different in some respects.

The results of Case 2 emphasise the risk of professional attacks that can be carried out with considerable resources. The motivation behind this kind of attack might be political issues or terrorism, for example. It is challenging to define SOs leading to security controls that are capable of effectively mitigating this risk. However, all relevant actions should be taken. For example, utilisation of HW- rather than SW-based security controls, and of effective cryptographic solutions with enough protection from side-channel attacks should be used. In practice, R1 (a professional attack carried out with unrestricted resources) contributes fundamentally to the political risk and terrorism risk, although not explicitly mentioned for Case 2.

Configuration correctness plays an important role in the SOs for Case 2: risks R12 (weaknesses in key management), R14 (OS support problems), R16 (execution of unsigned SW), R17 (SW update problems), R19 (communication between networks with different classifications), R25 (lack of OS management), and R27 (modifications of configuration files) can be mitigated thereby. As a high-level risk, R13 (poor configuration management) is listed too. The SW and system quality are emphasised in Case 2. For example, R3 (inappropriate testing) is cited among the main risks.

V. Initial Heuristics For Security Objective Decomposition

Below, we discuss the process of deriving risk-driven security metrics from the RA results and SOs, and we propose initial considerations of heuristics for SO decomposition, with the aim of security metrics development.

A. From Security Objectives to Security Metrics

Fig. 2 depicts a simplified example referring to the principle of decomposition of the SOs related to effectiveness of authentication functionality [18]. Basic Measurable Components (BMCs) are leaf components of a decomposition that clearly manifests a measurable property of the system [17]. The high-level BMCs shown in Fig. 2 are Authentication Identity Uniqueness, Authentication Identity Structure, Authentication Identity Integrity, Authentication Mechanism Reliability, and Authentication Mechanism Integrity [17]. Note that the number of nodes in the hierarchy is much larger in practice. The BMCs of Fig. 2 can be further decomposed, in account of specific system characteristics. For example, authentication typically requires about 100 or more nodes, as discussed in [6].

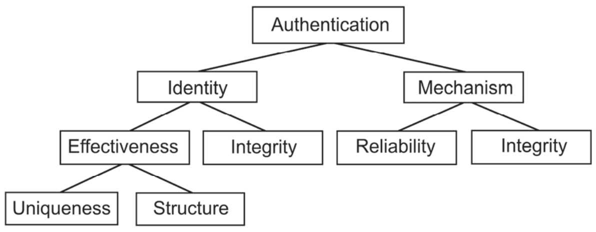

Figure 2. An example authentication decomposition based on [17]

We have proposed six strategies for security measurement objective decomposition for top-down risk-driven security metrics development and management in [1]. The *basic strategies* proposed addressed security configuration correctness, direct partial security effectiveness, and software and system quality. *Integrated strategies* were proposed to support compliance with best practice and regulations, pure security effectiveness, and the security effectiveness vs efficiency trade-off.

B. Security Effectiveness Abstract Models

In the approach taken in [1], it is important in the first stage to think about SE by defining s Security Effectiveness Abstract Model (SEAM), a simplified model that encompasses the core knowledge of factors contributing to the SE of the target system [1].

TABLE IV. SEAMs for Authentication and Authorisation

High-level SO	**Sub-properties contributing to the SE of SO**
Authentication	1. Identity SE a. Uniqueness b. Structure 2. Policy SE 3. Mechanism SE a. Reliability b. Integrity
Authorisation	1. Authentication SE 2. Auhorisation objects SE 3. Policy SE 4. Access control mechanism SE

Table IV presents example SEAMs for authentication and authorisation, based on [1]. Note that the SE of authentication makes a vital contribution to that of authorization: effective authorisation cannot be achieved without proper authentication. The SEAMs form a nested hierarchy of SE goals. The sub-properties should be mapped to the more detailed SO descriptions. For instance, the 'Policy SE' item under of 'Authorisation' in Table IV is a sub-property corresponding to Table III's 'Authorisation policy'.

C. Decomposing Security Objectives

As can be seen from many of the risks listed, configuration correctness is one of the main security controls required in the target system. In addition to having direct importance in mitigation of specific risks, various configuration correctness controls contribute indirectly to the overall SE of the system.

Table V shows an example of how configuration correctness objectives can be decomposed, in line with the process in [1].

As the risk concerns in the RA results discussed above show, the SW and system quality should be sufficient in the target system. Strategy 3 of [1] can be used to decompose SW and system quality related objectives. This process includes incorporating applicable vulnerability database information, and, in parallel to the case of configuration correctness, an SEAM is used to guide the decomposition process.

The SE of the target system can be measured partly by means of metrics developed from the 'direct partial SE' strategy of [1]. This strategy is aimed at SE-relevant gathering of direct evidence from penetration testing and incident statistics. 'Direct partial security metrics' offer input to mitigation of R3 in Case 2.

One cannot measure SE as a whole directly; however, it is possible to measure the factors contributing to it. The 'pure' SE strategy [1] combines the results from configuration correctness, direct partial SE and SW and system quality decompositions, all of them contributing to the SE as a whole. These aggregated SE decompositions are applicable in measurement of many higher-level SOs in the target system. As information disclosure is one of the core risks, an aggregated 'pure' SE model should be defined for it. Moreover, authentication and authorisation clearly require aggregated models.

TABLE V. HEURISTICS FOR CONFIGURATION CORRECTNESS OBJECTIVES DECOMPOSITION, BASED ON [1]

Stage	Description
A1	Identify relevant security measurement objectives on the basis of relevant configuration correctness objectives. For example, in the case of SO aimed at mitigation of Case 2's R17, address issues of how SW updates affect configuration. *In fact, this can imply definition of rules for that, if not available.* The security measurement objectives should involve monitoring how the rules are obeyed. Even though the goal is correctness, set security measurement objectives emphasising SE as much as possible. Investigate available and attainable security configuration evidence and its relevance to SE. Prioritise the results with respect to SE [1].
A2–A3	Develop a relevant SEAM, which includes mapping to security measurement objectives from Stage A1. If a reference model, such as a standard of SW updates relevant from the R17 perspective, is available, analyse the correspondence of the security measurement objectives and the reference model.
A4	Identify the system components relevant to the configuration correctness objectives. The components are architectural components (like modules, devices, protocols, interfaces, platforms) [1]. In the case of SW updates (R17), it is important to set boundaries to which parts of the system can be updated via which kinds of updating procedures.
A5	Carry out iterative decomposition of the objectives. Use SEAM to guide the process in order to ensure that the resulting metrics contribute to SE.
A6	Identify the measurement points in the metrics hierarchy. These are data structures, devices or files where the configuration data and the deployment control resides [1].
A7	In the decomposition, BMCs should be aimed at feasible metrics or use of available metrics, with the goal being a conclusion of 'OK' or 'not OK' [1].

The above-mentioned initial considerations for decomposition heuristics can be applied separately to each SO resulting from the RA. The actual security measurement activity may vary. For example, in the case of configuration correctness, some work can be automated, while monitoring security policies involves assessment and interviews. No matter the specific measurement method used, a risk-driven SO decomposition method should be used to define the metrics.

Use of a PSS mobile network system typically includes compliance with a host of regulations. Although compliance was not seen as a risk in the RA, the compliance strategy of [1] should be utilised to offer evidence of compliance from the SE perspective.

VI. RELATED WORK

Android OS security risks have been studied by Fedler et al. [19], who conclude that most successful attacks affecting Android can be attributed to negligent user behaviour. However, they admit that attacks on Android devices are becoming more sophisticated. Fedler et al. list fraud involving conductible money, industrial espionage, corporate and military networks' infiltration, and DoS attacks as threat scenarios. All of these, except financial fraud, were addressed in our RA, since it is not among the core concerns for PSS networks. Even if Android were to use disk encryption, it has been demonstrated that it is possible to perform cold boot attacks to retrieve disk encryption keys from random access memory (RAM) [20]. Lack of visual indicators for Secure Sockets Layer – Transport Layer Security (SSL/TLS) usage and the inadequate use of SSL/TLS can be exploited for launch of man-in-the-middle attacks as presented in [21]. Juniper Networks [12] describes the following drivers of Android malware: an emerging Android monoculture, anonymity of application developers, loosely managed open application marketplaces, malicious mobile marketplaces, and loose management of Android devices. On 6 June 2013, Kaspersky Lab presented analysis [22] of a multi-functional Android Trojan that is capable of, for example, sending SMS text messages to premium-rate numbers; downloading other malware programs, installing them, and/or sending them further via Bluetooth; and remotely running commands on the console. The Trojan used both known and previously unknown errors in the Android OS.

The SO decomposition approach discussed in this paper is similar in general to the Goal Question Metrics (GQM) of Basili et al. [23], a three-level decomposition approach for refining specification of software measurements. The highest level is the conceptual level (goals), the next one operational (question), and the third quantitative (metric). The GQM definitions lack strategies or heuristics to define their security relevant content aimed at *security* metrics. Requirement decomposition in general has been was discussed by Kirkman [24] and Koopman [25]. They consider the challenges in decomposition: 'gaming' promoted by too great a focus on goals, excessive subsystem decomposition, insufficient decomposition, unattributed requirements, excessive hierarchy and issues of change management. There are already numerous security metrics proposed in the literature, as summarised e.g. in [26–29]. However, the state-of-the-art lacks widely accepted and well-validated approaches to security metrics, because security is often considered to be an 'add-on' property, the security research field itself is in its infancy, and there is lack of suitable data for use in metrics development [30].

VII. CONCLUSIONS AND FUTURE WORK

According to the risk analysis discussed in this study, the main security risks of the Android platform as utilised in public safety and security mobile networks include risks arising from unauthorised input of falsified data, denial-of-Service scenarios and unauthorised root access. In a more security-critical case for the same target system, the risks arising from professional attacks, paralysis of communication at a critical moment, and inappropriate testing were seen as being of the highest priority.

We analysed security objectives on the basis of risk analysis results. The basic building blocks for security objectives are authentication and authorisation, integrity, and confidentiality controls. Access control plays an important role in the target system. Moreover, considerations of software and system quality are important. Many of the risks identified can be mitigated through assurance of configuration correctness.

Some initial heuristics for security objective decomposition were proposed. All heuristics emphasise security effectiveness. The decomposition of configuration correctness is based on investigation of specific settings in the configuration which contribute to security effectiveness. Decomposition work for software and system quality objectives incorporates investigation of vulnerability databases. Security effectiveness metrics designed for penetration testing and for gathering of incident information can be used to offer early visibility of the security level. Aggregation of security objective decompositions presenting factors that contribute to security effectiveness is a tool for well-grounded thinking about security as a whole.

We plan to focus in our future work on defining more detailed security objectives for the target systems and defining a hierarchy of security metrics by applying the decomposition heuristics described in this paper.

ACKNOWLEDGEMENT

The work presented here has been carried out in two research projects: the IoT Program (2012-2014), launched by the Finnish Strategic Centre for Science, Technology and Innovation TIVIT Plc., and SASER-Siegfried Celtic-Plus project (2012-2015).

REFERENCES

[1] R. Savola, “Strategies for Security Measurement Objective Decomposition,” ISSA 2013, 15–17 August 2013, Johannesburg, South Africa, 8 p.

[2] R. Savola, “Security Metrics Taxonomization Model for Software-Intensive Systems,” Journal of Information Processing Systems, Vol. 5, No. 4, Dec. 2009, pp. 197–206.

[3] W. Jansen, “Directions in Security Metrics Research,” U.S. National Institute of Standards and Technology, NISTIR 7564, Apr. 2009, 21 p.

[4] ITSEC. Information Technology Security Evaluation Criteria (ITSEC), Version 1.2, Commission for the European Communities, 1991.

[5] ISO/IEC 15408-1:2005. Common Criteria for Information Technology Security Evaluation – Part 1: Introduction and General Model, International Organization for Standardization and the International Electrotechnical Commission, 2005.

[6] R. Savola, C. Frühwirth, and A. Pietikäinen, “Risk-driven Security Metrics in Agile Software Development – an Industrial Pilot Study”, Journal of Universal Computer Science, Vol. 18, No. 2, Sept. 2012, pp. 1679–1702.

[7] Android Security Overview. http://source.android.com/devices/tech/security/index.html [accessed on 10 July 2013].

[8] J. Forristal, Uncovering Android Key That Makes 99% of Devices Vulnerable, 3 July 2013. http://bluebox.com/corporate-blog/bluebox-uncovers-android-master-key/ [accessed on 10 July 2013].

[9] S. J. Vaughan-Nichols, Google Releases Fix to OEMs for Blue Security Android Security Hole, 8 July 2013. http://www.zdnet.com/google-releases-fix-to-oems-for-blue-security-android-security-hole-7000017 782/ [accessed on 10 July 2013].

[10] J. Forristal, Scan Your Device for the Android “Master Key” Vulnerability, 7 July 2013. http://bluebox.com/corporate-blog/free-scanner-to-manage-risk-of-major-android-vulnerability/ [accessed on 10 July 2013].

[11] Google, Android Developer Page. https://developer.android.com /about/dashboards/index.html [accessed on 10 July 2013].

[12] Juniper Networks. Juniper Networks Third Annual Mobile Threat Report, March 2012 through March 2013.

[13] J. Oberheide, Remote Kill and Install on Google Android, 24 June 2010. http://jon.oberheide.org/blog/2010/06/25/remote-kill-and-install-on-google-android/ [accessed on 10 July 2013].

[14] J. Boutet, Malicious Android Applications: Risks and Exploitation, SANS Institute, 22 March 2010.

[15] D. Gray, TETRA: Advocate’s Handbook, from Paper Promise to Reality, 2003.

[16] M. Peltola, “Evolution of Public Safety and Security Mobile Networks,” licentiate thesis, Aalto University School of Electrical Engineering, Espoo, Finland, 2011.

[17] R. Savola and H. Abie, “Development of Measurable Security for a Distributed Messaging System,” International Journal on Advances in Security, Vol. 2, No. 4, 2009, pp. 358–380 (published in March 2010).

[18] C. Wang and W.A. Wulf, “Towards a Framework for Security Measurement”, Proceedings of 20th National Information Systems Security Conference, 1997, pp. 522–533.

[19] R. Fedler, C. Banse, C. Krauss and V. Fusenig, “Android OS Security: Risks and Limitations – A Practical Evaluation,” Fraunhofer AISEC Technical Report, May 2012.

[20] T. Müller, M. Spreitzenbarth and F.C. Freiling, “FROST - Forensic Recovery of Scrambled Telephones,” Technical Report, Department of Computer Science, Friedrich-Alexander University of Erlangen-Nuremberg, Germany, 2012.

[21] S. Fahl, M. Harbach, T. Muders, L. Baumgärtner, B. Freisleben, and M., “Why Eve and Mallory Love Android: An Analysis of Android SSL (In)security,” Proceedings of the 2012 ACM Conference on Computer and Communications Security, New York, NY, pp. 50–61.

[22] R. Unuchek, “The Most Sophisticated Android Trojan”, Karpersky Lab, 6th June 2013, https://www.securelist.com/en/blog/8106/The_most_sophisticated_Android_Trojan [accessed on 10 July 2013].

[23] V. Basili, G. Caldiera, and H.D. Rombach, ”The Goal Question Metric Approach,” J. Marciniak (Ed.), Enclyclopedia of Software Engineering, Wiley, 1994.

[24] D. Kirkman, “Requirement Decomposition and Traceability,” Requirements Engineering, Vol. 3, No. 2, 1998, pp. 107–114.

[25] P. Koopman, “A Taxonomy of Decomposition Strategies Based on Structures, Behaviors, and Goals,” Design Theory & Methodology ’95, 1995.

[26] D. S. Herrmann, Complete Guide to Security and Privacy Metrics – Measuring Regulatory Compliance, Operational Resilience and ROI, Auerbach Publications, 2007, 824 p.

[27] A. Jaquith, Security Metrics: Replacing Fear, Uncertainty and Doubt, Addison-Wesley, 2007.

[28] N. Bartol, B. Bates, K.M. Goertzel, and T. Winograd, Measuring Cyber Security and Information Assurance: A State-of-the-art Report, Information Assurance Technology Analysis Center, May 2009.

[29] V. Verendel, “Quantified Security Is a Weak Hypothesis: A Critical Survey of Results and Assumptions,” New Security Paradigms Workshop, Oxford, U.K., 2009, pp. 37–50.

[30] R. Savola, “On the Feasibility of Utilizing Security Metrics in Software-Intensive Systems,” International Journal of Computer Science and Network Security, Vol. 10, No. 1, 2010, pp. 230–239.