Automated Smartphone Security Configuration

William M. Fitzgerald, Ultan Neville, and Simon N. Foley

Cork Constraint Computation Centre,University College Cork, Ireland.
`wfitzgerald@4c.ucc.ie, u.neville@4c.ucc.ie, s.foley@cs.ucc.ie`

Abstract. Smartphones host operating systems that are on a par with modern desktop environments. For example, Google *Android* is a mobile operating system that is based upon a modified version of the Linux OS. Notwithstanding traditional threats to mobile phones, threats to desktop environments are also applicable to smartphones. Management of security configurations for the end-user has, to date, been complex and error-prone. As a consequence, misconfiguration of and/or a poor understanding of a security configuration may unnecessarily expose a smartphone to known threats. In this paper, a threat-based model for smartphone security configuration is presented. To evaluate the approach, a prototype smartphone security agent that automatically manages security configurations on behalf of the end-user is developed. A case study based on firewall access control demonstrates how automated security configuration recommendations can be made based on catalogues of countermeasures. These countermeasures are drawn from best-practice standards such as NIST 800-124, a guideline on cell phone and PDA security and NIST 800-41-rev1, a guideline on firewall security configuration.

1 Introduction

Modern smartphones with their processing power, operating systems and the wide variety of applications (apps for short) are on a par with modern desktop environments [25]. This has resulted in smartphones being used in a variety of domains from a personal device (such as for voice, Web browsing, Email and social media) to enterprise, medical and military domains [31]. The technological advances and the usage of smartphones in a variety of domains is not without its security implications. In addition to traditional mobile phone threats, threats to desktop environments are also applicable to smartphones [10, 17, 25]. For example, Malware threats such as DroidDream [8], a Android Market trojan app used to maliciously root Android smartphones, are on the increase [25].

Smartphones may host a variety of security mechanisms such as anti-virus, app monitoring and firewalls. In practice, security mechanisms are either disabled or configured with an open access policy [20]. Configuration of smartphone security mechanisms, for example a firewall, is typically performed by non-technical end-users. As a consequence, an effective security configuration may be hampered by a poor understanding and/or management of smartphone application requirements. Mis-configuration, may result in the failure to adequately provide smartphone app services. For example, an overly-restrictive firewall configuration may prevent normal interaction of network-based apps. An overly-permissive firewall configuration, while permitting normal operation of the app, may leave the smartphone vulnerable to attack, for example, across open ports or malicious payloads.

Smartphones operate in mobile network environments and deploying security configuration for a global set of threats is not practical. For example, a smartphone may in one scenario be connected to an enterprise WiFi network, an open access WiFi network or a 3g operator network. Thus, the deployment of smartphone security configurations must be dynamic in order mitigate the relevant threats within a given scenario. That is, what may be considered a threat in one scenario may not be a threat in another. Consider the scenario where a security configuration that permits a set of apps (such as gaming and social media apps) within a home network environment may not longer be permitted within an enterprise or teleworking environment. For example, in a teleworking scenario it is considered best practice to permit the use of "*a different brand of Web browser for telework*" and prohibit the use of the everyday Web browser [23].

The contribution of this paper is as follows. A threat-based model for smartphone security configuration is presented. Catalogues of best practice standards for smartphones are encoded within this model. A case study for smartphone firewall configuration management is considered. This research extends the work in [13] by specialising the firewall catalogues of best practice for smartphones and new catalogues of best practice for example NIST 800-114 [23] are developed. A prototype firewall app agent is developed for the Android platform [1] to automatically manage firewall configurations on behalf of the non-expert enduser.

This paper is organised as follows. Section 2 provides an introduction to Linux iptables, the stock Android platform firewall. A threat-based security model for smartphones is presented in Section 3. Section 4 outlines a set of best practice standards that are encoded within the security model. The implementation of the smartphone best practice catalogue is discussed in Section 5. A prototype automated firewall app for the Android platform is discussed in Section 6. Related research is outlined in Section 7 and Section 8 concludes the paper.

2 Background

The Android platform is a software framework for mobile devices such as smartphones and tablet PC's, and is based upon a modified version of the Linux OS. This section provides an overview of the Linux iptables firewall.

2.1 Linux iptables

Netfilter [14] is a framework that enables packet filtering, Network Address Translation (NAT) and packet mangling for Linux. A front-end called *iptables* is used to construct firewall rules that instruct Netfilter how to interpret packets.

As a firewall, iptables has stateless, stateful and application-layer packet filtering capabilities.

An iptables (firewall, NAT or mangle) rule requires the specification of a *table*, a *chain*, the accompanying *filter conditions* on packet fields that must be matched and an associated *action* outcome. With iptables, there are four tables: `filter`, `nat`, `mangle` and `raw`. A table is a set of chains and it defines the global context for common packet handling functionality. For example, the `filter` table defines the set for firewall rules, while the `nat` table defines the set of rules concerned with Network Address Translation. A chain is a set of rules that define the local context within a table. Rules within a chain are applied to the context defined both by the chain itself and the particular table. This paper focuses on the firewalling aspects of iptables, that is, the `filter` table. There are three built-in chains defined within the `filter` table that govern traffic being routed to (INPUT chain), from (OUTPUT chain) and beyond the firewall itself (FORWARD chain). Figure 1 illustrates the iptables packet traversal according to its associated chain. The reader is referred to [14,29] for further information.

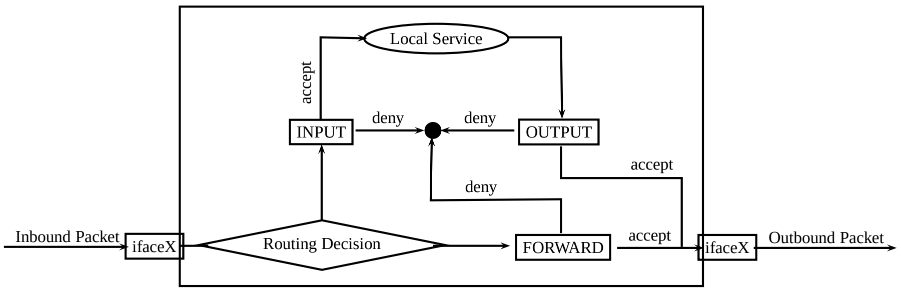


Fig. 1: Linux iptables Filter Table Chain Packet Traversal.

Example iptables rule syntax. The following (whitelist) iptables access-control rule states that outbound (`OUTPUT -o wifi`) TCP packets (`-p tcp`) that have originated from the Firefox Web browser (`-m owner --uid-owner 10101`) executing on the smartphone destined to any external Web server (`-d 0.0.0.0/0 --dport 80`) will be permitted (`-j ACCEPT`).

```
iptables -A OUTPUT -o wifi -p tcp -d 0.0.0.0/0 --dport 80 -m
owner --uid-owner 10101 -j ACCEPT
```

3 Security Threat Model

The security *State* of a smartphone represents attributes of a phone in use that may introduce vulnerabilities and/or influence how threats are mitigated. These attributes may correspond to, for example, user-preferences (indicating for instance, security risk appetite), or how the smartphone is currently used (for instance, a WiFi or 3g Internet connection). While there is potentially a large number of such attributes, for this research we focussed on five which, in-part based on best practice recommendations, have a direct impact on Network Access Controls on smartphones.

Network Interface Attribute A smartphone may be configured to communicate over WiFi and/or 3g networks. Note that a network interface configuration of WiFi and 3g, combined, corresponds to a tethering state. Let *Iface* define the set ($\mathbb{P}\,X$ denotes powerset of X) of possible network interface configurations as

$$\mathit{Iface} \mathrel{\widehat{=}} \mathbb{P}\{\mathsf{wifi}, \mathsf{3g}\}$$

Network Connection Attribute Different network connections may be trusted in different ways. For example, a WiFi connection providing WPA2-Enterprise security may be considered trusted, while an open WiFi connection in a default configuration may be considered untrusted. Let *NetConn* define the possible network connection attribute states.

$$\mathit{NetConn} \mathrel{\widehat{=}} \{\mathsf{trusted}, \mathsf{untrusted}\}$$

Risk Appetite Attribute This user-selected attribute reflects the level of risk that the user is willing to accept [5]. An appetite of hungry means that the user is willing to take risks and is satisfied with minimal countermeasures necessary to mitigate threats. An appetite of averse means that the user wishes for the most extensive countermeasures, for example, defense in depth.

$$\mathit{RiskAppetite} \mathrel{\widehat{=}} \{\mathsf{averse}, \mathsf{hungry}\}$$

Note, future research may consider additional risk appetite granularity and include minimalist, cautious and open attributes [5].

Teleworking Attribute This attribute indicates whether the smartphone is used in teleworking, or non-teleworking mode. We define:

$$\mathit{Telework} \mathrel{\widehat{=}} \{\mathsf{true}, \mathsf{false}\}$$

Battery Level Attribute The experimental results outlined in Section 6.1 found that the number of firewall rules can have an impact on battery consumption. Therefore, when battery power is low, a user with a low risk appetite may wish to reduce the number of rules in the firewall. Thus, we include the current battery level in the state of the smartphone.

$$\mathit{Battery} \mathrel{\widehat{=}} \{\mathsf{lo}, \mathsf{hi}\}$$

Security State The set of all possible states of the smartphone is defined as:

$$\mathit{State} \mathrel{\widehat{=}} \mathit{Iface} \times \mathit{NetConn} \times \mathit{RiskAppetite} \times \mathit{Telework} \times \mathit{Battery}$$

Threats Let the set *Threat* define the set of all known threats (of interest). A threat is a potential for violation of security [27] and in this paper we are interested in network-based threats that can be mitigated using a firewall. For example, $\mathsf{xmas} \in \mathit{Threat}$ represents the threat of a TCP half scan [18]. Let *threatens* define the relationship between threats and states.

$$\mathit{threatens} : \mathit{Threat} \leftrightarrow \mathit{State}$$

where, $\mathit{threatens}(t, s)$ indicates that threat t is considered to threaten a smartphone in state s. For example, we would expect a smartphone in a state with the WiFi interface enabled and an open WiFi connection to be threatened by xmas.

Countermeasures Let the set *Countermeasure* define the set of known countermeasures. For example, given firewall access-control rule:

$$f_1 = \langle \texttt{iptables -A INPUT -p tcp --tcp-flags ALL NONE -j DROP} \rangle$$

then $f_1 \in \mathit{Countermeasure}$. In this paper, we are interested in iptables-based countermeasures, and therefore, members of this set are described in terms of iptables command syntax. This could be generalized to the threat ontology described in [13] in order to extend to other kinds of countermeasures. Let relation *mitigates* define the threats mitigated by a countermeasure:

$$\mathit{mitigates} : \mathit{Countermeasure} \leftrightarrow \mathit{Threat}$$

where $\mathit{mitigates}(c, t)$ indicates that countermeasure c mitigates threat t. For example, the firewall rule f_1 above mitigates the threat of a TCP half scan, that is, $\mathit{mitigates}(f_1, \mathsf{xmas})$.

Blacklists and Whitelists as threats A blacklist is used to prevent the smartphone from initiating (outgoing) connections to known malicious hosts. Thus, a blacklisted host with IP address A is represented as a threat, denoted $\mathsf{blist}_o(A)$, within our model. This threat is mitigated by blocking outgoing packets to A at the smartphone firewall, that is,

$$\mathit{mitigates}(\langle \texttt{iptables -A OUTPUT, ... ,-d } A\texttt{, DROP} \rangle, \mathsf{blist}_o(A))$$

A similar interpretation is used for blacklisting inbound (INPUT and FORWARD chains) connections.

A networked Android app has associated port(s), and whitelists are used to define the apps that are permitted to engage in network connections. Whitelists are modelled in terms of threats, whereby a firewall that does not permit a whitelisted app to access the network is treated as a threat and the countermeasure is a corresponding '`ACCEPT`' iptables rule. For example, a whitelisted app that is permitted to initiate outgoing connections on port P is vulnerable to threat denoted $\mathsf{wlist}_o(P)$, and we have countermeasure:

$$\mathit{mitigates}(\langle \texttt{iptables -A OUTPUT, ... ,--sport } P\texttt{, ACCEPT} \rangle, \mathsf{wlist}_o(P))$$

A similar interpretation is used for whitelisting inbound IP addresses and ports.

Countermeasure Deployment The countermeasures deployed on a smartphone should mitigate all threats for its current state. We define a deployment operation

$$deploy : State \rightarrow \mathbb{P}\ Countermeasure$$

which selects a suitable set of countermeasures $deploy(s)$ for the state s. The next section describes our current implementation for this operation, however, in general, it should uphold the following property.

$$\begin{aligned} &\forall\, s : State;\ t : Threats \mid threatens(t, s) \\ &\qquad \Rightarrow \exists\, c : Countermeasure \mid c \in deploy(s) \wedge mitigates(c, t) \end{aligned}$$

In this paper, the implementation of $deploy(s)$ assumes the correct sequencing of the firewall rules. Future research will consider *structural analysis* techniques (for example [11] when automatically generating an anomaly-free firewall configuration. For example, the removal of redundant access-control rules.

4 Catalogues of Best Practice

A best practice standard is a high-level document that defines a set of recommended best practices (countermeasures) to protect sensitive and critical system resources. The following best practice standards NIST 800-41 [30], NIST 800-41rev1 [22], NIST 800-124 [16], NIST 800-114 [23] and NIST 800-153 [28] for firewall access control have been encoded within our model. For example, Table 1 and Table 2 illustrate excerpts of recommended best practice for general firewall configuration [30] and firewall configuration whilst teleworking [23] respectively. Note, the reader is referred to [13] for the systematic approach used to identify and categorise recommended firewall best practice in terms of detailed threats and mitigating countermeasures (iptables rules). Detailed threats identified are an interpretation of the recommendation descriptions.

The advantage of developing catalogues from best practice standards is two fold. Firstly, it provides a basis to automatically generate compliance-driven firewall configurations. Secondly, it provides a basis with which to consider real-world firewall access-control rules. For example, NIST 800-41rev1 recommendation FBPr1-2 in Table 1 recommends that (spoofed) packets arriving on an external interface claiming to have originated from either of the three RFC1918 reserved internal IP address ranges should be dropped. Such traffic indicates a denial of service attack typically involving the TCP syn flag. NIST 800-114 recommendation TBP-1 in Table 2 recommends that in a teleworking scenario a firewall should be configured with a whitelist of trusted network-based apps.

Catalogues developed as part of this work extends the catalogues in [13] specialised for mobile devices. New best practice catalogues, namely NIST 800-124 [16], NIST 800-114 [23] and NIST 800-153 [28] have also been developed. The catalogue of firewall best practice for smartphones developed as part of this research consists of one hundred and thirty five distinct threat and countermeasure pairs. Future research will extend this catalogue to include knowledge about other best practice standards. Note, the majority of the catalogue countermeasures are templates. For example, the following firewall access-control rule outlined in NIST 800-114 recommendation TBP-2 (Table 2)

```
iptables -A OUTPUT -m owner --uid-owner $appUID state --state
NEW,ESTABLISHED,RELATED -j ACCEPT
```

is a template countermeasure that has an UID variable `$appUID` which is modified each time an access-control rule is applied to a locally executing network-based smartphone app.

Table 1: Extract of NIST-800-41-Rev1: Guidelines on Firewalls & Firewall Policy

ID	**Recommendation Description**	
FBPr1-1	"*deny by default policies should be used for incoming TCP and UDP traffic.*" [23].	
	Threat	**Countermeasure**
	No inbound default deny policy	`iptables -P INPUT DROP`
	No outbound default deny policy	`iptables -P OUTPUT DROP`
	No forward default deny policy	`iptables -P FORWARD DROP`
ID	**Recommendation Description**	
FBPr1-2	"*. . . an invalid source address for incoming traffic or destination address for outgoing traffic . . . should be blocked*" that is "*An IPv4 address within the ranges in RFC 1918*" and "*An address that is not in an . . . IANA . . . range*" [22]	
	Threat	**Countermeasure**
	Inbound local 192.168.0.0/16 Src IP Pkt	`iptables -A INPUT -s 192.168.0.0/16 -j DROP`
	Outbound local 192.168.0.0/16 Dst IP Pkt	`iptables -A OUTPUT -d 192.168.0.0/16 -j DROP`
	Inbound forward 192.168.0.0/16 Src IP Pkt	`iptables -A FORWARD -i $iface -s 192.168.0.0/16 -j DROP`
	Outbound forward 192.168.0.0/16 DstIP Pkt	`iptables -A FORWARD -o $iface -d 192.168.0.0/16 -j DROP`
	Inbound local 10.0.0.0/8 Src IP Pkt	`iptables -A INPUT -s 10.0.0.0/8 -j DROP`
	Outbound local 10.0.0.0/8 Dst IP Pkt	`iptables -A OUTPUT -d 10.0.0.0/8 -j DROP`
	Inbound forward 10.0.0.0/8 Src IP Pkt	`iptables -A FORWARD -i $iface -s 10.0.0.0/8 -j DROP`
	Outbound forward 10.0.0.0/8 DstIP Pkt	`iptables -A FORWARD -o $iface -d 10.0.0.0/8 -j DROP`
	Inbound local 172.16.0.0/12 Src IP Pkt	`iptables -A INPUT -s 172.16.0.0/12 -j DROP`
	Outbound local 172.16.0.0/12 Dst IP Pkt	`iptables -A OUTPUT -d 172.16.0.0/12 -j DROP`
	Inbound forward 172.16.0.0/12 Src IP Pkt	`iptables -A FORWARD -i $iface -s 172.16.0.0/12 -j DROP`
	Outbound forward 172.16.0.0/12 Dst IP Pkt	`iptables -A FORWARD -o $iface -d 172.16.0.0/12 -j DROP`
ID	**Recommendation Description**	
FBPr1-3	"*Organizations should also block . . . IP source routing information*" [22]	
	Threat	**Countermeasure**
	SSRR firewall bypass.	`iptables -A FORWARD -m ipv4options --ssrr -j DROP`
	LSRR firewall bypass.	`iptables -A FORWARD -m ipv4options --lsrr -j DROP`
ID	**Recommendation Description**	
FBPr1-4	"*Organizations should also block . . . directed broadcast addresses*" [22]	
	Threat	**Countermeasure**
	Inbound Local directed broadcast	`iptables -A INPUT -d x.x.x.255 -j DROP`
	Outbound Local directed broadcast	`iptables -A OUTPUT -d x.x.x.255 -j DROP`
	Inbound forward directed broadcast	`iptables -A FORWARD -i $iface -d x.x.x.255 -j DROP`
	Outbound forward directed broadcast	`iptables -A FORWARD -o $iface -d x.x.x.255 -j DROP`
ID	**Recommendation Description**	
FBPr1-5	To limit Denial of Service "*a firewall might redirect the connections made to a particular inside address to a slower route if the rate of connections is above a certain threshold.*" [22]	
	Threat	**Countermeasure**
	Inbound forward DoS to tethered device	`iptables -A FORWARD -i $iface -d $lanIP -m limit --limit $x/s --limit-burst $y -j ACCEPT`

Table 2: Extract of NIST-800-114: User's Guide to Securing External Devices for Telework and Remote Access

ID	**Recommendation Description**	
TBP-1	Construct an access control whitelist of locally hosted applications trusted for telework network access: "*teleworkers should install and use only trusted software*" [23].	
	Threat	**Countermeasure**
	Inbound local application whitelist traffic not permitted	`iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT`
	Outbound local application whitelist traffic not permitted	`iptables -A OUTPUT -m owner --uid-owner $appUID state --state NEW,ESTABLISHED,RELATED -j ACCEPT`
ID	**Recommendation Description**	
TBP-2	... "*silently ignore unsolicited requests sent to it, which essentially hides the device from malicious parties.*" [23].	
	Threat	**Countermeasure**
	ICMP ping network scan	`iptables -A INPUT -p icmp -j DROP`
	TCP XMAS network scan	`iptables -A INPUT -p tcp --tcp-flags ALL ALL -j DROP`
	TCP Null network scan	`iptables -A INPUT -p tcp --tcp-flags ALL NONE -j DROP`
	TCP Syn Fin network scan	`iptables -A INPUT -p tcp --tcp-flags SYN,FIN SYN,FIN -j DROP`
	TCP Rst Fin network scan	`iptables -A INPUT -p tcp --tcp-flags FIN,RST FIN,RST -j DROP`
	TCP Port 0 network scan	`iptables -A INPUT -p tcp --dport 0 -j DROP` `iptables -A INPUT -p tcp --sport 0 -j DROP`
ID	**Recommendation Description**	
TBP-3	"*Use a different brand of Web browser for telework*" [23].	
	Threat	**Countermeasure**
	Regular Web browser usage	`iptables -A OUTPUT -p tcp --dport 80 -m owner --uid-owner $untrustedHTTPUID -j DROP`
	Intended telework Web browser usage not permitted	`iptables -A OUTPUT -p tcp --dport 80 -m owner --uid-owner $trustedHTTPUID state --state NEW,ESTABLISHED -j ACCEPT`
ID	**Recommendation Description**	
TBP-4	"*Configuring primary applications to filter content and stop other activity that is likely to be malicious*" [23]	
	Threat	**Countermeasure**
	Outbound local unfiltered traffic	`iptables -A OUTPUT -m -string --algo bm --string ‘$filterString’ -j DROP`
ID	**Recommendation Description**	
TBP-5	"*Personal firewalls should be configured to log significant events, such as blocked and allowed activity*" [23]	
	Threat	**Countermeasure**
	No inbound local audit control	`iptables -A INPUT -j LOG --log-level 7`
	No inbound forward audit control	`iptables -A FORWARD -i $iface -j LOG --log-level 7`

5 Firewall Catalogue Implementation and Deployment

In the smartphone implementation of the catalogue for firewall best practice, we have:

$$\mathit{isMemberOfCategory} : \mathit{Threat} \leftrightarrow \mathit{Category}$$

where $\mathit{isMemberOfCategory}(t, c)$ indicates that threat category c includes threat t. Table 3 illustrates a fragment of the threat classification developed. The relationship between security states and threats is implemented as:

$$\mathit{threatenState} : \mathit{Category} \leftrightarrow \mathit{State}$$

where $\mathit{threatenState}(c, s)$ indicates the set of threats categorised within category c threaten the smartphone in state s. The implementation of the *threatens* relation from the model defined in Section 3 is given by the relational composition $\mathit{isMemberOfCategory} \fatsemi \mathit{threatenState}$.

Table 3: Extract of Threat Catalogue

Detailed Threats	**Threat Category**
FBPr1-2 Threats	*Spoofing*
FBPr1-2 Threats FBPr1-4 Threats FBPr1-5 Threats	*DoS*
TBP-2 Threats	*Scanning*
FBPr1-3 Threats	*Source Routing*
TBP-4 Threats	*Malicious Content*
FBPr1-1 Threats TBP-1 Threats TBP-3 Threats	*Promiscuity Level*
TBP-5 Threats	*Non-Audit*

5.1 Threat Taxonomy

Having analysed the best practice standards outlined previously, threats where categorised in the following way: *Spoofing*, *Denial of Service*, *Scanning*, *Source Routing*, *Malicious Content*, *Promiscuity Level* and *Non-Audit*. Note, other threat categories could be chosen, for example Microsoft's STRIDE classification [15].

Threats classified as *Spoofing* are those that refer to IP address spoofing. For example, threats described by the FBPr1-2 recommendation in Table 1 are considered spoofing threats.

Denial of Service threats are those that have the capability of flooding network resources. For example, in Table 1 FBPr1-4 recommends IP address broadcast mitigation and FBPr1-5 recommends threshold-limiting to mitigate connection-based denial of service threats. Note, recommendation FBPr1-4 currently considers the more common /24 network broadcast range only and does not consider additional network broadcast ranges for example /25 or /26.

Network information disclosure threats, for example those outlined by NIST 800-114 recommendation TBP-2 in Table 2, are classified as *Scanning* threats.

Source Routing, for example NIST 800-41rev1 recommendation FBPr1-3 in Table 1, is a threat classification where an attacker may specify the route the packet takes through the network and has the potential to bypass firewalls.

From a firewall configuration perspective, *Malicious Content* threats are those that contain malformed application payloads such as URL parameters, form elements and SQL queries. Malicious Content may be mitigated in a variety of ways for example blacklisting known TCP/UDP ports or performing Deep Packet Inspection (DPI) on known malicious signatures. Recommendation TBP-4 in Table 2 illustrates a template DPI firewall rule that mitigates outbound Malicious Content threat communication.

Threats that are categorised as *Promiscuity Level* are those that refer to IP address (and/or port) reachability in terms of unintended whitelisting or backlisting. That is, an overly-promiscuous firewall configuration (unintended whitelisting), while permitting normal operation of the smartphone app, may expose other apps to unintended threats. Whilst, an overly-restrictive firewall configuration (unintended blacklisting) may prevent normal interoperation of services with the resulting failure of the smartphone app. An example of this is outlined by NIST 800-114 recommendation TBP-1 Table 2.

Non-Audit threats are those that do not log relevant traffic communications. From a compliance perspective, it is considered best practice to log traffic for auditing purposes. For example, NIST 800-114 recommendation TBP-5 in Table 2 outlines teleworking auditing threats and their corresponding firewall mitigation.

5.2 Security States

The (5-tuple) security *State* space defined in Section 3 provides a total of sixty-four states in which a smartphone may operate. However, we argue that certain attribute combinations are not valid and therefore the security state space may be reduced to twenty-eight. Table 4 illustrates the valid security state matrix.

In this paper, we assume that firewalls under the control of trusted network providers such as a 3g operator are compliant with best practice standards such as [22,30]. A user with a risk appetite of hungry, for example state-7 in Table 4, may therefore not be concerned about threats of IP spoofing, denial of service, port scanning and/or source routing where it is assumed the upstream trusted network provider firewalls are mitigating these kinds of threats.

While the trusted network providers provide firewall mitigation against threats of IP spoofing, denial of service, port scanning and source routing, it is considered best practice to also restrict access at the smartphone firewall as part of a defense in depth strategy [22]. As a consequence, security states where the user has risk appetite of averse such as state-1, state-3, and state-25 are said to be also threatened by those threats (Table 4).

The number of firewall access-control rules can have an impact on battery consumption (Section 6.1). Therefore, when battery power is lo, despite a user having specified a risk appetite of averse, the number of access-control rules will be reduced. Security state state-4 is an example, where there is a trade-off of security in depth such as IP spoofing to conserve battery power. Effectively a smartphone with a lo battery where a user has specified a risk appetite of averse will default to a state where the user is not concerned as much about his/her smartphone's security configuration (risk appetite of hungry). For example security states state-15 and state-16.

In contrast, if the smartphone is operating in a state that involves teleworking, for example security states state-1, state-5 and state-10, then a defense in depth strategy will be applied to mitigate all threat categories regardless of the network connection, the risk appetite or the battery level. This is in keeping with NIST 800-114 [23] best practice recommendations.

5.3 Automatic Generation of Firewall Configurations

Suitable firewall configurations are automatically generated for each smartphone security state using the information contained in Table 4 and the threat cata-

Table 4: Matrix of valid Security States

State												
	Interface	**Network Connection**	**Risk Appetite**	**Teleworking**	**Battery**	**Spoofing**	**DoS**	**Scanning**	**Source Routing**	**Malicious Content**	**Promiscuity Level**	**Non-Audit**
state-1	wifi	trusted	averse	true	hi	x	x	x		x	x	x
state-2	wifi	trusted	averse	true	lo	x	x	x		x	x	x
state-3	wifi	trusted	averse	false	hi	x	x	x		x	x	
state-4	wifi	trusted	averse	false	lo						x	
state-5	wifi	trusted	hungry	true	hi	x	x	x		x	x	x
state-6	wifi	trusted	hungry	true	lo	x	x	x		x	x	x
state-7	wifi	trusted	hungry	false	hi						x	
state-8	wifi	trusted	hungry	false	lo						x	
state-9	wifi	untrusted	averse	true	hi	x	x	x		x	x	x
state-10	wifi	untrusted	averse	true	lo	x	x	x		x	x	x
state-11	wifi	untrusted	averse	false	hi	x	x	x		x	x	x
state-12	wifi	untrusted	averse	false	lo	x	x	x		x	x	
state-13	wifi	untrusted	hungry	true	hi	x	x	x		x	x	x
state-14	wifi	untrusted	hungry	true	lo	x	x	x		x	x	x
state-15	wifi	untrusted	hungry	false	hi						x	
state-16	wifi	untrusted	hungry	false	lo						x	
state-17	3g	trusted	averse	true	hi	x	x	x		x	x	x
state-18	3g	trusted	averse	true	lo	x	x	x		x	x	x
state-19	3g	trusted	averse	false	hi	x	x	x		x	x	
state-20	3g	trusted	averse	false	lo						x	
state-21	3g	trusted	hungry	true	hi	x	x	x		x	x	x
state-22	3g	trusted	hungry	true	lo	x	x	x		x	x	x
state-23	3g	trusted	hungry	false	hi						x	
state-24	3g	trusted	hungry	false	lo						x	
state-25	3g,wifi	trusted	averse	false	hi	x	x	x	x	x	x	
state-26	3g,wifi	trusted	averse	false	lo					x	x	
state-27	3g,wifi	trusted	hungry	false	hi						x	
state-28	3g,wifi	trusted	hungry	false	lo						x	

logues (for example Table 2). Consider security states state-1 and state-3 where teleworking and non-teleworking occurs. The firewall configuration generated for security state state-1 will in addition to mitigating threat categories that similarly threaten security state state-3, include audit-based access-control rules in compliance with NIST 800-114 recommendation TBP-5 in Table 2.

While various security states may have been related to the same threat categories, the firewall configuration generated for each security state may be different. Consider security states state-3 and state-25 in Table 4. Both security states are threatened by threats within the category IP spoofing. However, the specific/individual IP spoofing threats such as those described by NIST 800-41rev1 recommendation FBPr1-2 in Table 1 will differ for both security states. Because security state state-25 is concerned with tethering, it must consider additional firewall access-control rules that mitigate IP spoofing threats along its iptables FORWARD chain to protect smartphone tethered devices [16]. Note, in a tethering scenario, the smartphone is an internet gateway for tethered devices.

There are also scenarios where permitted (trusted) network apps in one security state may no longer be permitted in another security state. For example, trusted networked apps such as telnet, FTP or games for example in security state-3 may alternate between whitelists and blacklists in a security state that involves teleworking, for example security state-1. This is ensures compliance with NIST 800-114 recommendation TPB-1 in Table 2. That is, only trusted apps defined in accordance with the enterprise-level teleworking security policy may be permitted. Note, while it may be advantageous to deny access to telnet in an enterprise network for a risk appetite of averse (for example state-3), it may also be acceptable to restrict access to telnet for trusted clients (IP address whitelist) while in a home network environment.

In a teleworking scenario, access control is not just defined at the level of IP addresses or TCP/UDP ports, for example prohibiting port 23 (Telnet) or port 80 (HTTP). Access control is also applied at the application level such as UID and Layer-7 filtering. For example, NIST 800-114 recommendation TBP-3 in Table 2 recommends that different Web browsers such Firefox and Google Chrome, should be used in teleworking and non-teleworking scenario. This is to minimise the Web browser used for for general use, which may have become compromised with malicious plugins, from communicating in a teleworking scenario. A set of suitable iptables countermeasures that filter using the owner-match extensionare defined.

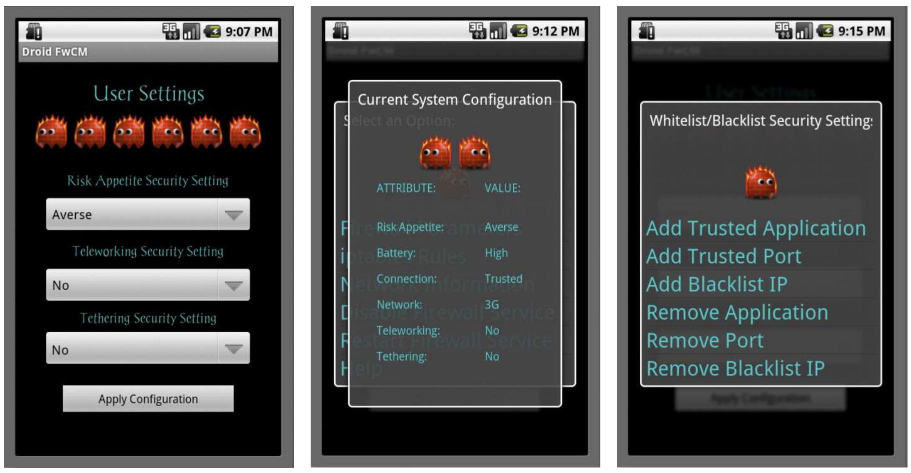
6 Prototype Android Firewall Agent App

A prototype automated agent app is developed that manages the smartphone firewall configuration on behalf of the non-expert end-user. The test-bed used for the prototype was an *Android 2.1, Revision 1* platform on a HTC hero smartphone with an ARMv6 528MHz processor and a lithium-ion battery with a capacity of 1350 mAh. Note, a rooted and customised Android ROM image that includes additional iptables extensions such as *string match* and *recent match.*

Figure 2 illustrates examples of the Graphical User Interface developed as part of the prototype. The user settings interface is illustrated in Figure 2a where a user may specify his/her risk appetite and whether or not the smartphone will operate as a tether or in telework environment. Figure 2b, illustrates an example of the current security state. The interface with which a user may define his/her whitelist and blacklist for (un-) trusted apps is illustrated in Figure 2c.

Fig. 2: Example Screenshots of Firewall Agent App Graphical User Interface

(a) User Settings (b) Current System Settings (c) Promiscuity Settings



Risk Appetite Security Setting: Averse

Teleworking Security Setting: No

Tethering Security Setting: No

Current System Configuration

ATTRIBUTE:	VALUE:
Risk Appetite:	Averse
Battery:	High
Connection:	Trusted
Network:	3G
Teleworking:	No
Tethering:	No

6.1 Firewall Configuration and Battery Consumption Correlation

A number of preliminary experiments where carried out to evaluate the impact of firewall configuration size with respect to battery consumption. The experimental set-up was as follows. Firewall configurations of 0, 500 and 1000 access-control rules where deployed on the smartphone for each of the three experiments. The battery capacity for each experiment was 100% (fully charged). A 2GB TCP data-stream was transmitted to the smartphone (from an external machine) where packets are not matched until the last access-control rule in the firewall configuration. Each experiment was repeated 5 times to get the average battery depletion rate. Table 5 illustrates the preliminary findings. The first column reflects the firewall configuration size. The second column reflects the remaining (average) battery level after each experiment. The results indicate that during periods of network communication, the firewall configuration size does have an impact on battery consumption. For example, to filter a 2GB data-stream, a firewall with a 1000 access-control rules consumed 36% more battery charge than a firewall with 0 access-control rules.

While in practice, smartphones do not tend to process large data-streams and/or be configured with a large number of access-control rules, these experiments were intended to stress test the smartphone. Note, in future work, an

Table 5: Correlation between Firewall Configuration and Battery Level

# of Access-Control Rules	Battery Level
0	88%
500	70%
1000	52%

additional set of experiments that may reflect a more real world scenario will be considered. For example, a 20MB-90MB data-stream range (from Web browsing to video streaming [2]) tested against firewall configurations consisting of 0, 100 and 250 access-control rules. In addition, experiments where the initial battery capacity is set to 50% and 25% rather than 100% should be considered. As the battery nears a capacity of minimal charge, we conjecture that even a modest sized rule-set consisting a few hundred access-control rules will have a significant impact on battery consumption [7, 9, 21]. Therefore, the battery level is considered within the security model presented in Section 3.

7 Related Research

There are a number of existing techniques for smartphone Malware and intrusion detection [25]. For example, the authors in [24] adopt a static analysis approach to detect Malware. A machine learning approach is taken in [26] to detect application anomalies. There are a number of Android apps for firewall configuration management, for example DroidWall [3] and WhipserMonitor [4]. However, the level of access control granularity provided is limited. For example, only egress access control (iptables OUTPUT chain) to whitelist or blacklist apps is considered. The model presented in this paper considers fine-grained ingress (iptables INPUT and FORWARD chains) and egress (iptables OUTPUT and FORWARD chains) access control. In existing works, Android firewall configuration is performed on an ad-hoc basis. For example, there are no recommended guidelines for whitelisting or blacklisting apps in a given security context. In contrast, the automatic generation of smartphone firewall configurations in this research is guided by best practice recommendations.

There are a number of existing techniques that can be used by enterprise security administrators to generate [12, 13], query [13, 19] and perform structural analysis [6, 11] on network access control configurations. In this research, we do not consider these approaches and assume that smartphone firewall configurations are conflict free. Future research will explore the effectiveness of the approach for firewall configuration management outlined in [13] with respect to the Android platform.

8 Conclusion

This paper presented a formal model for smartphone security configuration. Catalogues developed as part of this work extends the catalogues in [13] specialised for mobile devices and provided a basis with which to evaluate the security model. The prototype firewall app agent may be used by non-expert end-users to automatically generate suitable firewall configurations that are compliant with best practice.

Future research will extend the current modelled smartphone firewall catalogues and consider for example catalogues related to smartphone Malware and intrusion detection mitigation. In addition, a future iteration of our (preliminary) security model may consider additional attributes. For example, the physical location of a smartphone where it may be advantageous to prevent a smartphone operating in a teleworking scenario for example when it is located in a certain (untrusted) country or region of the world.

Acknowledgement

The authors would like to thank the anonymous reviewers for their valuable feedback. This research has been supported in part by Science Foundation Ireland grant 08/SRC/11403.

References

1. http://www.android.com/
2. http://www.vodafone.ie/internet-broadband/internet-on-your-mobile/usage/
3. http://code.google.com/p/droidwall/
4. http://www.whispersys.com/
5. Thinking about risk - managing your risk appetite: A practitioner's guide. HM Treasury on behalf of the Controller of Her Majesty's Stationery Office (HMSO) (November 2006)
6. Al-Shaer, E.S., Hamed, H.H., Boutaba, R., Hasan, M.: Conflict Classification and Analysis of Distributed Firewall Policies. IEEE Journal on Selected Areas in Communications, Issue: 10, Volume: 23, Pages: 2069 - 2084 (October 2005)
7. Balanza, M., Abendan, O., Alintanahin, K., Dizon, J., Caraig, B.: Battery Discharge Characteristics of Wireless Sensor Nodes: An Experimental Analysis. 2nd Conference on In Sensor and Ad Hoc Communications and Networks, IEEE (September 2005)
8. Balanza, M., Abendan, O., Alintanahin, K., Dizon, J., Caraig, B.: DroidDreamLight Lurks Behind Legitimate Android Apps. 6th International Conference on Malicious and Unwanted Software (MALWARE) (April 2011)
9. Buennemeyer, T.K., Gora, M., Marchany, R.C., Tront, J.G.: Battery Exhaustion Attack Detection with Small Handheld Mobile Computers. IEEE International Conference on In Portable Information Devices, (PORTABLE) (May 2007)
10. Chin, E., Felt, A.P., kate Greenwood, Wagner, D.: Analyzing inter-application communication in android. 9th international conference on Mobile systems, applications, and services, (MobiSys), ACM, USA (2011)
11. Cuppens, F., Cuppens-Boulahia, N., García-Alfaro, J.: Detection and Removal of Firewall Misconfiguration. IASTED International Conference on Communication, Network and Information Security (CNIS) (November 2005)
12. Cuppens, F., Cuppens-Boulahia, N., Sans, T., Miège, A.: A Formal Approach to Specify and Deploy a Network Security Policy. 2nd Workshop on Formal Aspects in Security and Trust (FAST) (August 2004)

13. Foley, S.N., Fitzgerald, W.M.: Management of Security Policy Configuration using a Semantic Threat Graph Approach. Journal of Computer Security (JCS), IOS Press, Volume 19, Number 3 (2011)
14. Gheorghe, L.: Designing and Implementing Linux Firewalls with QoS using netfilter, iproute2, NAT and l7-filter. PACKT Publishing (October 2006)
15. Hernan, S., Lambert, S., Ostwald, T., Shostack, A.: Uncover Security Design Flaws Using The STRIDE Approach. http://microsoft.com/
16. Jansen, W., Scarfone, K.: Guidelines on Cell Phone and PDA Security: Recommendations of the National Institute of Standards and Technology. NIST-800-124 (2008)
17. Khadem, S.: Security issues in smartphones and their effects on the telecom networks. MSc Dissertation, Chalmers University of Technology, University of Gothenburg, Sweden (August 2010)
18. Lyon, G.: NMAP Network Scanning: Official Nmap Project Guide to Network Discovery and Security Scanning. Insecure LLC, CA, United States (2008)
19. Marmorstein, R., Kearns, P.: A Tool for Automated iptables Firewall Analysis. Usenix Annual Technical Conference, Freenix Track, Pages: 71-81 (April 2005)
20. Ruggiero, P., Foote, J.: Cyber threats to mobile phones. TIP-10-105-01, United States Computer Emergency Readiness Team (US-CERT) (April 2010)
21. Saha, B., Goebel, K.: Modeling Li-ion Battery Capacity Depletion in a Particle Filtering Framework. Annual Conference of the Prognostics and Health Management Society, San Diego, CA, USA (September 2009)
22. Scarfone, K., Hoffman, P.: Guidelines on Firewalls and Firewall Policy: Recommendations of the National Institute of Standards and Technology. NIST Special Publication 800-41, Revision 1 (September 2009)
23. Scarfone, K., Souppaya, M.: User's Guide to Securing External Devices for Telework and Remote Access: Recommendations of the National Institute of Standards and Technology. NIST-800-114 (2007)
24. Schmidt, A.D., Bye, R., Schmidt, H.G., Clausen, J., Kiraz, O., Yüksel, K.A., Camtepe, S.A., Albayrak, S.: Static analysis of executables for collaborative malware detection on android. In: Proceedings of the 2009 IEEE international conference on Communications. ICC'09, IEEE Press, Piscataway, NJ, USA (2009)
25. Shabtai, A., Fledel, Y., Kanonov, U., Elovici, Y., Dolev, S., Glezer, C.: Google android: A comprehensive security assessment. Security and Privacy, IEEE Computer Society, Volume 8, Issue 2 (March 2010)
26. Shabtai, A., Kanonov, U., Elovici, Y., Glezer, C., Weiss, Y.: "andromaly": a behavioral malware detection framework for android devices. J. Intell. Inf. Syst. 38(1) (Feb 2012)
27. Shirey, R.: RFC 2828: Internet Security Glossary. http://ietf.org (May 2000)
28. Souppaya, M., Scarfone, K.: Guidelines for Securing Wireless Local Area Networks (WLANs): Recommendations of the National Institute of Standards and Technology. NIST-800-153 (2012)
29. Suehring, S., Ziegler, R.L.: Linux Firewalls: Third Edition. Novell Publishing (2006)
30. Wack, J., Cutler, K., Pole, J.: Guidelines on Firewalls and Firewall Policy: Recommendations of the National Institute of Standards and Technology. NIST-800-41 (2002)
31. Wei, X., Gomez, L., Neamtiu, I., Faloutsos, M.: Malicious Android Applications in the Enterprise: What Do They Do and How Do We Fix It? Workshop on Secure Data Management on Smartphones and Mobiles, Washington D.C (April 2012)