

Authorization Logic For Mobile Ecosystems

Joseph Hallett

Doctor of Philosophy
Laboratory for Foundations of Computer Science
School of Informatics
University of Edinburgh
2017

Abstract

TODO

Table of Contents

5	Applying AppPAL to BYOD policies	1
5.1	Overview of five BYOD policies	2
5.2	Review of mobile device management (MDM) software	4
5.3	Modelling BYOD policies	7
5.4	BYOD idioms in AppPAL	9
5.4.1	Delegation and Roles within Policies.	11
5.4.2	Acknowledgement.	13
5.5	Enforcing a BYOD policy with AppPAL	14
	Bibliography	17

List of Figures

5.1	Policy settings in the MaaS360 MDM tool.	6
5.2	Interactions in a company with policies.	15

List of Tables

5.1	Summary of different MDM capabilities	5
5.2	Counts of predicate-types in each policy.	8
5.3	Occurrences of predicates common to multiple policies.	10
5.4	Summary of different authorities in BYOD policies.	12

Chapter 5

Applying AppPAL to BYOD policies

Many employees bring their personal mobile devices to work. To control the access these devices have to company resources, an estimated 72% of companies publish bring your own device (BYOD) policies.¹ These BYOD policies are natural language documents that employees should read and obey. They describe steps to take to secure devices in the workplace. The policies say how employees should get access to data, and who should authorise decisions.

Companies can use their policies in varying ways. Some companies may trust employees to follow the rules on their own. Alternatively MDM software can implement part of the policies: packages such as IBM's MaaS360 and Blackberry's BES [?, ?] can configure devices to restrict functionality and manage apps.

Commercial tools have limits to what they currently enforce. Some tools can only enable simple on-off configuration settings, and ban explicitly black-listed apps. More advanced systems can use app-rewriting to recompile apps to tunnel traffic through a VPN, or geofencing to apply policies in predefined areas. These tools are not infallible. One survey found that 50% of companies with MDM software still had non-compliant devices in their networks [12]. Whilst app wrapping can protect some apps, in general it is ineffective [7].

¹According to survey of companies by LinkedIn [15]. 40% made BYOD available to all employees, 32% made it available to select employees.

5.1 Overview of five BYOD policies

Through out this chapter, we will explore how we can apply AppPAL to BYOD policies through the analysis of five *real-world* policies. We chose the policies from a variety of domains. They cover a range of different policy styles and concerns. Two are example policy templates a company might want to base its own BYOD policy on, published by security advice organizations. The remaining three are BYOD policies used inside companies.

1. Is the *Security Policy Template: Use of Handheld Devices in a Corporate Environment*, published by the SANS Institute [13]. This policy is a hypothetical policy published to help companies mitigate the threats to corporate assets caused by mobile devices. Companies change the document to suit their needs and BYOD requirements. The policy is general; not specific to any particular industry, device, or country's legislation.
2. Is from the Health information and Management Systems Society (HiMSS) [8]; a US non-profit company trying to improve healthcare through IT. The HiMSS policy is relatively short and has concerns specific to healthcare scenarios. The policy is a contract the users follow. The police has a different style to other policies: in the other policies the company states what users should do, here the user states what they will do. The policy is designed as a sample agreement for a system trying to manage personal mobile devices in a healthcare environment.
3. Is from a British hospital trust [9] and describes the BYOD scheme used in practice at the hospital. The policy is broad and covers rules for corporately owned devices. This policy also briefly describes how devices should interact with patients.
4. Is a relatively simple policy from The University of Edinburgh [16]. It is briefer, and groups rules into sections for high and low risk users, unlike the other policies.
5. Is by a company specialising in emergency sirens [?]. Again this policy is simpler, and more general than the other policies we looked at.

Each of the policies is split into a series of rules and requirements employees should follow. Typically the policy is describing what employees should do

and what will happen under certain circumstances. For example the NHS policy states that:

NHS: *In the event of loss of the device, all data including apps will be wiped. The Trust is not responsible for reimbursement of any costs for personally purchased apps.*

The Siren policy matches this style, stating what conditions will lead to the company wiping an employees device:

Sirens: *The employees device may be remotely wiped if: • The device is lost or stolen. • The employee terminates his or her employment. • IT detects a data or policy breach, a virus or similar threat to the security of the companys data and technology infrastructure.*

As mentioned before the HiMSS policy has a different style. This is shown by the equivalent rule being phrased from the perspective of the policy subject (“I agree. . .”).

HiMSS: *I agree that the PDA/Smartphone can be wiped by XYZ Health System upon the decision of XYZ Health System management and understand that it will delete all data including personal files.*

The SANS policy is mostly written in the same style as the NHS policy. It is written as a hypothetical policy that a company might use as the basis for their own policy, however it sometimes says things that seem like advice to an IT department implementing the policy. It also For example:

SANS: *A corporate mobile device management solution SHALL feature remote device wiping (or possibly only blocking) mechanism for all devices accessing corporate internal networks.*

The SANS policy also distinguishes between rules that should always be followed (SHALL), and those that may depend on a specific business’s situation (SHOULD).

SANS: *Basically, sentences using the verb “SHALL” are mandatory requirements applying to practices with high probability of putting the business at risk, whereas “SHOULD” means that the policy needs to be applied according to the businesss*

specific situation.

The Edinburgh policy takes this even further. The policy groups rules by device type and gives a security level. The policy expects high and medium risk users to follow everything, and low risk users to at least consider following the rules marked with a ∇ . For example the policy groups together the rule for remote wiping devices with the rules specific to mobile devices and tablets.

Edinburgh: ∇ *Configure your device to enable you to remote-wipe it should it become lost.*

5.2 Review of MDM software

A company might use MDM software to enforce their rules. Many companies offer an MDM solutions, including IBM (MaaS360), VMware (AirWatch) and MobileIron; however they all offer a comparable set of features, namely:

1. App wrapping; where an app is modified to offer some additional features or network properties. This is often limited to routing all the app's network traffic through a VPN.
2. Basic security configuration; allows an administrator to turn on and off security settings. They might do this to enable encryption on a device.
3. Provisioning; where IT departments can install and update apps and their configuration files. Email and LDAP configuration is common.
4. A curated app store; where IT departments can white or black list apps.

The features of 9 competing MDM packages identified by a Gartner report are summarised in Table 5.1 [14]. Most of the tools link to the report on their homepages as the report lists *every* tool as either *visionary*, *leading*, or *niche player*.² In practice all the tools are very similar with the chief difference being the UI as well as some extra features only certain tools support.

Whilst MDM tools can configure devices, the policies they enforce are less sophisticated than what you can write with a policy language such as AppPAL. The policies of an MDM tool are essentially granular checkboxes (as shown in

Feature	MaaS360	BlackBerry BES	MobileIron	Citrix XenMobile	VMWare AirWatch	Microsoft	SOTI MobiControl	Sophos	Landdesk
Antivirus								✓	
App selection/store/management	✓	✓	✓	✓	✓	✓	✓	✓	✓
App wrapping/modification		✓	✓	✓	✓	✓		✓	✓
Authentication	✓	✓	✓	✓	✓	✓	✓		
Compliance reporting	✓	✓	✓	✓	✓	✓	✓	✓	
Device configuration	✓	✓	✓	✓	✓	✓	✓		✓
Email/Calendar/Contacts/Documents	✓	✓	✓	✓	✓	✓	✓	✓	✓
Feature Restrictions	✓	✓			✓	✓			
Licence distribution		✓							
Location based settings	✓				✓				
Network configuration	✓	✓	✓	✓	✓	✓	✓		
Password/Encryption settings	✓	✓	✓	✓	✓	✓	✓	✓	
Remote wipe	✓	✓	✓	✓	✓	✓	✓	✓	✓
Security auditing	✓	✓	✓	✓	✓	✓	✓	✓	
Tracking/Spyware	✓	✓				✓	✓		✓
Watermarking					✓				

Table 5.1: Summary of different MDM software's capabilities built from information from each of the tools sales pages.


 Network Settings		
Allow Wi-Fi On a Wi-Fi only device, unchecking this option will not block Wi-Fi.	Yes	Android 2.2+
Enforce Wi-Fi is always on	No	Android 2.2+
Bluetooth	User Controlled	Android 2.2+
Allow Data Network	User Controlled	Android 2.2+
Enable Background Data Synchronization Allows applications to sync, send or receive data any time.	User Controlled	Android 2.X & 3.X
Auto-Sync	User Controlled	Android 2.2+
Allow user to Mobile Data limit	Yes	SAFE 4.0+
Allow VPN Allow or disallow use of the native VPN functionality. If disabled, the user cannot establish a VPN session and the UI for using VPN through the Settings application is inaccessible.	Yes	SAFE 2.2+

Figure 5.1: Policy settings in the MaaS360 MDM tool.

Figure 5.1) an administrator can manually enable or disable to configure a device for a group of users.

As well as commercial MDM tooling there are also research tools that look at MDM software. Martinelli et al.'s work looks at creating a dynamic permissions manager, called UC-Droid. Their tool can alter what an app's Android permissions are at run time based on policies [11]. The tool allows companies to reconfigure their apps depending on whether the employee is at work, in a secret lab, or working out-of-hours. These kinds of policies are more configurable than the geofenced based policies some MDM tools offer. Other work has looked at enforcing different policies based on what roles an employee holds [6]. The work allowed a company to verify the devices within their network and what servers and services they could reach. It also describes a mechanism for providing different users with different policies.

Armando et al. developed BYODroid as a tool for enforcing BYOD policies through a secure marketplace [3]. Their tool allows companies to distribute apps through a secure app store [4]. The store ensures apps meet policies through a static analysis and app rewriting to add dynamic enforcement. Their

²The table summarises the visionary and leading MDM packages.

policies are low-level, based on ConSpec [1], allowing checks based on Dalvik VM's state. Using their tool, they implemented the parts of a NATO Communications and Information Agency policy about personal networks and data management [2]. Their work shows how to check and enforce the app-specific sections of a BYOD policy using tools. They did not look at where the checks or policies come from, however.

5.3 Modelling BYOD policies

Each of the policies are split into a series of rules. A simple example is the following example from the Sirens-company policy. The policy states that devices may get access to various company resources. For each resource we create an AppPAL assertion that states that a device may access the resource.

Sirens: *Employees may use their mobile device to access the following company-owned resources:*

• Email • Calendars • Contacts • Documents • Etc.

```
'department' says Device:D canAccess('email').
'department' says Device:D canAccess('calendars').
'department' says Device:D canAccess('contacts').
'department' says Device:D canAccess('documents').
```

The NHS policy has a more complex example. Employees are not allowed to call non-domestic, or premium rate numbers on company-owned phones³, however an employee's manager can make an exception. To write the AppPAL version of this rule we initially have the default rule that bans international calls, then we add a second rule stating that it is allowed if someone makes an exemption, finally a third rule states that an employee's manager can make the exemption.

NHS: *All mobile devices will be configured for national access only. Premium/international calls will be barred. International call barring and roaming arrangements can be lifted for specific periods, to be stipulated on request, on approval of the relevant manager/budget holder.*

³The NHS policy doesn't distinguish between company and privately owned phones and applies to both.

Check numbers

Policy	Decision				Condition			
	Can	Must	Has	Is	Can	Must	Has	Is
SANS	35% (35)	29% (29)	9 % (9)	27% (27)	2% (2)	2% (2)	8% (8)	87% (87)
HiMSS	21% (21)	41% (41)	31% (31)	7 % (7)	0	0	13% (13)	87% (87)
NHS	19% (19)	26% (26)	33% (33)	23% (23)	2% (2)	0	19% (19)	83% (83)
Sirens	27% (27)	45% (45)	11% (11)	16% (16)	2% (2)	7% (7)	2% (2)	89% (89)
Edinburgh	0	18% (18)	82% (82)	0	7% (7)	7% (7)	50% (50)	37% (37)

Table 5.2: Counts of predicate-types in each policy.

```

'nhs-trust' says Device canCall(TelephoneNumber:X)
  if Device isOwnedBy('nhs-trust'),
    X isNationalNumber, X isStandardRateNumber.

'nhs-trust' says Device canCall(TelephoneNumber:X)
  if Device isOwnedBy(Staff),
    Staff hasCallExemption.

'nhs-trust' says Manager can-say
  Staff hasCallExemption
  if Manager isManagerOf(Staff).

```

Table 5.2 shows a count of the different types of predicate in the policies. As before in ?? predicates The use of each is also split by whether the predicate is a *decision* made by the policy, or a *condition* for making that decision. *Can* and *must* decisions feature in all policies excepting *can* decisions in the Edinburgh policy, in part due to the structure of the policy as discussed in ?. We would expect this: these are access control decisions and reactions to events; both topics that existing MDM tools have focused on implementing. *Has* and *is* predicates are often used in the conditions, but there are also decisions using them too.

5.4 BYOD idioms in AppPAL

When examining the policies, there are some concern concerns shared between the different policies. A summary of predicates, with the same meaning, used in multiple policies by our translation is given in Table 5.3.

Acknowledgements, where the policy requests people acknowledge other policies, and predicates linking devices to owners feature in all policies. Most policies describe rules for when to enable and disable device features. Configuring device features is a common feature to many MDM packages, but tracking what a user agrees to is not seen in leading MDM packages like MaaS360 or BES [14]. Only 2 of the 5 policies had rules limiting access to networks, servers, or access points. and only the two most complex policies had rules limiting what apps could be installed. This is surprising as a common feature of MDM tools is controlling how devices and apps access networks. Users have privacy preferences about apps [10], but not all companies try to control what apps employees install. Providing curated app stores and blacklisting apps is a feature common to many MDM programs. Not all policies express rules about which apps to install, however. All policies talked about remotely wiping a device for security reasons (as shown in section 5.1); and most MDM programs implemented this by allowing an administrator remotely erase a device (Table 5.1).

Two particular idioms occur in many policies: acknowledgements and delegation. We describe both idioms in greater detail, and show how to describe them in AppPAL, below. MDM tools and research have focussed so far on implementing restrictions on apps and devices [?, 5, 11]. Implementing these controls is a vital aspect of BYOD policies and all 5 of the policies we looked at had rules that described restrictions (??). Every policy also contained rules that required employees acknowledgements, however. Only the SANS policy (which is configuration focussed) contained more rules that required restrictions than acknowledgements. All the policies contained more rules featuring delegation relationships than functionality restrictions. Restricting device functionality is tricky and important, but other aspects of BYOD policies are also worth attention.

Predicate	SANS	HiMSS	NHS	Sirens	Edinburgh
mustAcknowledge	✓	✓	✓	✓	✓
hasAcknowledged	✓	✓	✓	✓	✓
isOwnedBy	✓	✓	✓	✓	✓
isDevice	✓	✓	✓	✓	✓
mustDisable	✓		✓	✓	✓
mustWipe		✓	✓	✓	✓
isLost	✓	✓	✓	✓	
isEmployee	✓		✓	✓	✓
isApp	✓	✓	✓	✓	
isActivated	✓	✓	✓		✓
mustEnable	✓	✓		✓	
isEncrypted	✓		✓		✓
hasFeature	✓		✓		✓
hasMet	✓		✓		✓
canMonitor	✓		✓	✓	
mustInform	✓		✓		
isTelephoneNumber	✓		✓		
isString			✓	✓	
isSecurityLevel	✓	✓			
isInstallable	✓		✓		
isFeature	✓			✓	
isData	✓			✓	
isApprovedFor		✓	✓		
isApproved	✓	✓			
hasDevice		✓	✓		
hasDepartment		✓			✓
canUse	✓		✓		
canStore	✓	✓			
canInstall	✓		✓		
canConnectToServer	✓		✓		
canConnectToNetwork	✓			✓	
canConnectToAP	✓	✓			
canCall	✓		✓		
canBackupTo		✓			✓

Table 5.3: Occurrences of predicates common to multiple policies.

5.4.1 Delegation and Roles within Policies.

Delegation is an important part of the policies. Each of the policies describes through rules how separate entities are responsible for making some decisions. These rules could be a delegation to an employee's manager to authorize a decision (as in the NHS policy). It could be to technical staff to decide what apps are part of a standard install (as in the sirens and SANS policies).

When translating the policies, the author of the policy is the primary speaker of the policy's rule. This is typically the company; except in the HiMSS case where it is the user (??). All the policies describe multiple entities that might make statements and delegate.

Some policies have more authorities, than others (Table 5.4). The NHS policy has various managers that approve decisions for their staff. There are different groups that make decisions for the clinical and business halves of the business. If a clinical user wishes to use an app with a patient they must seek approval from two policy groups, as well as their line manager. Others make less use of different authorities. In the Edinburgh policy, the records-management office states how to configure a low or high risk device. There is no delegation to others to further specify aspects of the policy. Delegation of responsibilities is an important part of BYOD policies. MDM software seems largely to ignore it, however. These tools instead allow IT staff to set fixed policies and push them to devices. No further requesting of information is typically needed or required.

When a policy decision requires input from a third-party delegation is used. For example, an employee's manager has to authorise an app install. The SecPAL *can-say* statement is the basis for a delegation. We can ask the HR department to state who is someone's manager.

```
'company' says 'hr-department' can-say
  Employee:E hasManager(Employee:M).
```

If we wish to delegate to someone, we can add conditionals to the can-say statement that enforces any relationship between the delegating and delegated parties.

```
'company' says Manager can-say
  Employee canInstall(App:A)
  if Employee hasManager(Manager).
```


SANS	Authorities	10
	Primary Authority	company
	Technical Authority	it-department
	User Authority	user
HIMSS	Authorities	3
	Primary Authority	user
	Technical Authority	xyz-health-system
	User Authority	department
NHS	Authorities	11
	Primary Authority	nhs-trust
	Technical Authority	it-department
	User Authority	staff
Sirens	Authorities	4
	Primary Authority	department
	Technical Authority	it-department
	User Authority	employee
Edinburgh	Authorities	2
	Primary Authority	records-management
	Technical Authority	
	User Authority	employee

Table 5.4: Summary of different authorities in BYOD policies.

5.4.2 Acknowledgement.

All the policies we looked at require their subjects to be aware and acknowledge certain rules or policies, and that the company may perform certain actions. For example, the NHS and HiMSS policies state that the organisation will wipe devices remotely to protect confidential information a user loses their device. Both policies also say that employees would lose personal information if they had it on the device and the company needed to erase it. The employee is required to be aware of this, and in the case of the HiMSS policy, agree to hold the company harmless for the loss.

HiMSS: *I agree to hold XYZ Health System harmless for any loss relating to the administration of PDA/Smartphone connectivity to XYZ Health System systems including, but not limited to, loss of personal information stored on a PDA/Smartphone due to data deletion done to protect sensitive information related to XYZ Health System, its patients, members or partners.*

```
'xyz-health-system' says 'user' mustAcknowledge('data-loss-policy').
```

NHS: *Individuals who have personal data of any kind stored on a corporately issued mobile device must be aware that in the event of loss of the device the above data wipe will include removal of all personal data.*

```
'nhs-trust' says Staff:S can-say
  S hasAcknowledged('data-loss-policy').
'nhs-trust' says Staff:S mustAcknowledge('data-loss-policy').
```

Both the SANS and the siren-company policies use acknowledgements to link to other sets of rules that employees should follow. These policies are not further specified, and in the case of an acceptable use policy may be hard to enforce automatically. The SANS policy requires that all employees follow an email security, acceptable use, and an eCommerce-security policy. The Sirens

policy expects an employee to use their devices ethically and abide by an acceptable use policy.

SANS: *Users MUST agree to the email security/acceptable use policy and eventually to the eCommerce security policy.*

```
'company' says Employee:U mustAcknowledge('email-security').
'company' says Employee:U mustAcknowledge('acceptable-use').
'company' says Employee:U mustAcknowledge('ecommerce-security
').
```

Sirens: *The employee is expected to use his or her devices in an ethical manner at all times and adhere to the company's acceptable use policy.*

```
'department' says Employee:E mustAcknowledge('acceptable-use')
.
```

When there is a (usually separate) set of rules and concerns employees should be aware of acknowledgements are used. The company may not have wish to enforce these separate rules automatically, however. For instance, a company may have an ethical policy that says employees should not use devices for criminal purposes. The company is not interested in, or capable of, defining what is criminal. They trust their employees to make the right decision and to be aware of the rules.

To implement these in AppPAL, a policy author creates two rules: the first stating their employees must have acknowledged the policy, the second delegating the acceptance of the policy to the employee themselves.

```
'company' says Employee:E mustAcknowledge('policy').
'company' says Employee:E can-say
  E hasAcknowledged('policy').
```

5.5 Enforcing a BYOD policy with AppPAL

This work has so far focused on the contents of the BYOD policies. The concerns and delegation relationships have been highlighted as they are not handled

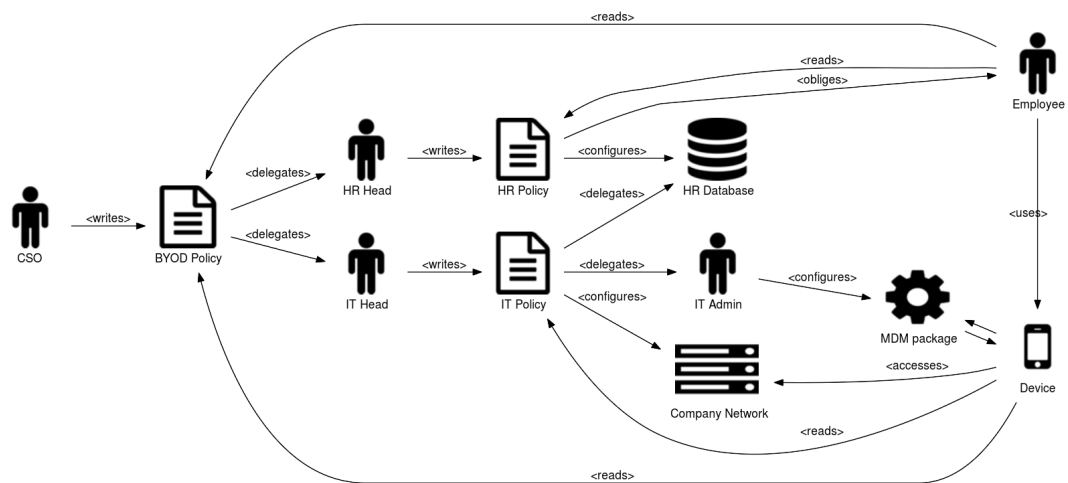


Figure 5.2: Interactions in a company with policies.

well by existing MDM tools. The policies also contain rules that the MDM tools do manage well: configuring, provisioning and wiping devices. The MDM tools fail, however, to give a means to control *how* they themselves are configured; with the typical method being through the manual intervention of an IT administrator.

AppPAL exists as a tool for checking whether a policy is satisfied. It is reasonable to ask what would a company have to do to *enforce* their mobile device policies. A company could have a structure such as in Figure 5.2: the Chief Security Officer sets the BYOD policy, which delegates to IT and HR to further specify parts of the policy. Employees are obliged to read all the policies put out and follow them (by HR). The IT department who maintain the company network, configure the MDM software on the employee's device, to ensure it meets their policies.

What AppPAL gives you is a means for checking the policy; to implement the policy the company would need to implement actions atop of these checks. If AppPAL checks a policy and finds a user must install an app or disable a feature on their phone, then the company might want to configure their MDM software to perform the necessary steps. If AppPAL were to discover that an employee hadn't satisfied the obligation to sign a contract, then the company might want to email a reminder to the employee.

AppPAL doesn't remove the need for MDM software. Much of an MDM package's functionality could be re-implemented as part of AppPAL, but this is an unnecessary duplication of work. Rather AppPAL should be used to

configure other, existing tools. An MDM package could enforce a password policy, and enable remote wipe (for example). Google's *For Work* tools can enforce access control policies for a company's documents, as can Microsoft's Office Suite. The settings on an Android app can control what permissions an app can have.

AppPAL works as a glue between existing tools. It tells you what configuration changes need to be made to satisfy a policy, if these can be automated then so much the better. In this respect AppPAL is similar to a configuration language. Giving employees a means to make AppPAL statements (for example a dialog box that says "*type your password to acknowledge the following policy.*") lets us track what people have done and make decisions on the basis of their actions.

Bibliography

- [1] Irem Aktug and Katsiaryna Naliuka. ConSpec A Formal Language for Policy Specification. *Electronic Notes in Theoretical Computer Science*, February 2008.
- [2] A. Armando, G. Costa, A. Merlo, L. Verderame, and K. Wrona. Developing a NATO BYOD security policy. In *International Conference on Military Communications and Information Systems*, May 2016.
- [3] Alessandro Armando, Gabriele Costa, and Alessio Merlo. Bring Your Own Device, Securely. In *Proceedings of the 28th Annual ACM Symposium on Applied Computing, SAC '13*, pages 1852–1858, New York, NY, USA, 2013. ACM.
- [4] Alessandro Armando, Gabriele Costa, Alessio Merlo, and Luca Verderame. Enabling BYOD through secure meta-market. In *ACM Conference on Security and Privacy in Wireless and Mobile Networks*, August 2014.
- [5] Alessandro Armando, Gabriele Costa, Alessio Merlo, and Luca Verderame. Formal modeling and automatic enforcement of Bring Your Own Device policies. *International Journal of Information Security*, August 2014.
- [6] G. Costantino, F. Martinelli, A. Saracino, and D. Sgandurra. Towards enforcing on-the-fly policies in BYOD environments. In *International Conference on Information Assurance and Security*, December 2013.
- [7] Hao Hao, Vicky Singh, and Wenliang Du. On the effectiveness of API-level access control using bytecode rewriting in Android. *Proceedings of the 8th ACM SIGSAC symposium on Information, computer and communications security*, 2013.
- [8] Healthcare Information and Management Systems Society. Mobile Security Toolkit: Sample Mobile Device User Agreement. *Healthcare Information and Management Systems Society*, 2012.
- [9] Gary Kennington, Kevin Pointer, Christine Morey, Linsey Budge, Vanessa Dunn, and Sue Ball. Mobiles Devices Policy. Technical report, Torbay and Southern Devon Health and Care NHS Trust, March 2014.
- [10] Jialiu Lin, Bin Liu, Norman Sadeh, and Jason I. Hong. Modeling Users Mobile App Privacy Preferences: Restoring Usability in a Sea of Permission Settings. *Symposium On Usable Privacy and Security*, 2014.

- [11] Fabio Martinelli, Paolo Mori, and Andrea Saracino. Enhancing Android Permission Through Usage Control: A BYOD Use-case. In *Symposium on Applied Computing*, 2016.
- [12] MobileIron Security Labs. Q4 Mobile Security and Risk Review. Technical report, MobileIron Security Labs, December 2015.
- [13] Nicholas R. C. Guerin. Security Policy for the use of handheld devices in corporate environments. Technical report, SANS, May 2008.
- [14] Rob Smith, Bryan Taylor, Chris Silva, Manjanath Bhat, Terrence Cosgrove, and John Girard. Magic Quadrant for Enterprise Mobility Management Suites. Technical Report G00279887, Gartner, June 2016.
- [15] Holger Schulze. BYOD & Mobile Security 2016 Spotlight Report. Technical report, LinkedIn Information Security, 2016.
- [16] David Williamson, Anne Grzybowski, and Susan Graham. Bring Your Own Device Policy. Policy 15, University of Edinburgh, February 2015.