BYOD-Secure Integration of Mobile Devices Into Company Network Infrastructure

A Dissertation
Submitted in partial fulfillment of the requirements
of the degree of

Master of Engineering
in
Computer Engineering

by

Suresh R.Mestry

Roll No 260

Supervisor

Dr.Y.S.Rao

Department of Computer Engineering
Sardar Patel Institute of Technology
Munshi Nagar, Andheri(W), Mumbai-400058
UNIVERSITY OF MUMBAI
2013-2014

Dissertation Approval for M. E.

This dissertation entitled ***"BYOD-Secure Integration of Mobile Devices Into Company Network Infrastructure"*** by ***Suresh R.Mestry*** is approved for the degree of ***Master of Engineering*** in ***Computer Engineering*** from ***University of Mumbai***.

Examiners

1.———————————————

2.———————————————

Supervisors

1.———————————————

2.———————————————

Head of Department

———————————————

Principal

———————————————

Date:

Place: Mumbai

Declaration

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

_____

(Signature)

_____

(Name of Student and Roll No.)

Date:

Abstract

Android OS devices such as phone, tablets are growing so faster in markets. Unfortunately, security is not concurrently guaranteed with functionality. Android has employed various mechanisms like application sandboxing, application signing, permission model, etc. to enhance the security. But all these mechanisms have certain limitations.

The Bring Your Own Device paradigm allows mobile devices to connect with virtual network infrastructure which consist of federated servers in order to access to services and functionalities. But basic security support by Android and other applications in market is insufficient for security requirements.

A security framework for android devices (specifically smart phones) where organizations or institutes decide security policies that employee must accept. Such rules and policies are aim to avoiding users to download any third party software inside the organization which invite outsider threats and even avoiding user to take sensitive information or data to outside which is nothing but insider threat.

Single -sign on server uses method of access control that enables user to log in only once and gain access to multiple resources without being prompted to log in again. Single sign on server is centralized identity management which authenticate the user by accepting users credentials. Single sign on can be use with BYOD scenario for additional and one time authentication for android devices.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The BYOD is a solution designed to specifically address the mobile enterprise needs. the system includes of key aspect called Mobile Device Management (MDM). It also supports single sign-on (SSO) and multi-tenancy.

The server enables company/institutes or organizations to manage, secure and monitor Android devices (e.g., smart phones and tablet PCs). Users need to accept the policy agreement, which states all the actions that can be carried out on the device when enrolling with federated server or MDM server.Main advantage of this system is, it only controls the corporate data that is present on the devices, while the personal data is secure by pin number provided by employee or end user.i.e.personal data is kept untouched.

The administrator can create policies in on server and define the device management rules, applications which are blacklisted and application which are allowed and need to install on android device after perticuler policy get enforced. When employees register their devices with server, the applicable policy rules (e.g., enabling the phone lock, disabling the camera, and more) will be enforced on their devices. Every moment MDM server monitor the device and accordingly can select follow-up actions like sending the user a warning message, enforce the policy again, and more) based on their security requirements.

The mobile management system consists of three key consoles: Console, Publisher and Store. Users uses the Publisher to manage enterprise apps throughout their application life cycle, which includes application states such as, published, unpublished, approved, rejected, deprecated, and retired. The Store acts as a marketplace and contains all the corporate mobile apps, which users can search, view, rate and install on-demand. The administrator uses the Console to manage users, administerand monitor policies.

1.1 Motive

The Android platform is attracting organizations/institutes that plan to adopt Android for professional use. In general, this trend consists in using employees owned private devices, e.g., smartphones and tablet devices, inside organizations. This is due to the growing capabilities(mobile device configuration that match) of smartphones,their diffusion and low costs. In this context, the idea of Bring Your Own Device(BYOD) is developing a policy and enforcement mechanisms allowing persons to use their own personal device in a professional context.

Deploying the secure BYOD paradigm on Android requires major security issues that Security Framework should natively able to manage. At the same time employees must

agree for such paradigm where they will face policies enforcement. The main challenge in this system is secure framework where android devices securly used within organization infrastructure without violeting any policy so that organization service and data remain safe within organization only.

1.2 Problem Statement

Security policies framework for android based personal devices belonging to different groups of employees and avoid employees to violate such policies by means of enforcement mechanisms that may be both embedded on the device or hosted on trusted external server(MDM server). The primary objective of security framework is MDM server to handle the mobile devices by using policy rules [1].

MDM server first create users (employee profile) and invite user by sending email to their mail id on android devices provided into profile.Once android device enroll into server, client software(.apk) from server.then user has to register.

After device get register with server,serverthen assign roles and policies like passwork, data wipe,encryption etc.

1.3 Objectives

The proposed system can be deployed as a mobile application and MDM server. Secure connection,single-sign on both can be used for deployment. Following objectives can be fulfilled with the help of the mobile application and MDM server:

- A user friendly android interface on which BYOD scenario can be implement.i.e. access services through this interface.
- Configuring and plug-ins for MDM servers which handle mobile devices,like managing services and access control.
- Preventing use of the organizations information on a device with applications not formerly approved.i.e restrict downloading of third party software.
- Encrypting the organizations data stored on a device to stop unauthorized applications and users from accessing it.
- Device locking is safeguarding of devices.By authorization of an administrator, enterprise server software can issue a command to immediately lock a managed mobile device preventing access until the necessary credentials (such as passwords, biometrics or cryp- tographic tokens) have been presented.
- Remote wipes involves configuring a device so that after a certain number of consecutive failed authentication attempts, the device will securely wipe itself.

Chapter 2

Literature Review

Research has been done to provide security framework for BYOD paradigm for employees and organisation.BYOD security framework proposed and prototyped using Mobile Device Management (MDM) but still industries not satisfied with security policies(due to inside and outside threat). Because of this reason ,BYOD security still in research.

2.1 Technical papers review

A security framework for mobile devices that ensures that only applications that comply with the organization security policy are installed on the registered devices in which the framework consists of 1) a security policy manager that mediates access to the applications stores by (i) keeping track of the security state of the registered devices and (ii) determining whether an application can be safely added to a device and 2) An installer application that tells the user which applications can be safely installed, which need further scrutiny by the security policy manager and finally which are known to violate the security policy [1, 4], paper auther present a type and effect system which allows us to infer history expressions from application implementations.

presented analysis report on impact on fast growing trend in mobile market where analysis report contains consumerisation(personal devices are used for work as well as leisure) benefits for the Enterprise,Consumerization concerns,auther explained about consumerisation trends,spread if BYOD,consumerisation benefits for the enterprise(BYOD appeals to the enterprise finance managers, where users pay for their own devices and the internal application farm is replaced by free web services),consumerization concerns(Small and light portable devices can be lost or stolen easily, and with them - confidential corporate data and access to enterprise facilities) [2].

Paper cited above cover two models which can be used to manage the mobile devices in an organization to bring in BYOD [3],i.e Mobile Device Management(MDM) and Cloud Architecture for Mobile Devices where the mobile device management tool helps the organization to fully control the devices which are generally supported by API‘s of smartphones used and the MDM tool helps on the security and management of device by monitoring, controlling and protecting the device. MDM architecture and BYOD concept control objectives a) Identification and access control b) Data protection c) Application security d) Integrity control e) compliance

Framework described application container loads the application in such a way that critical function calls outside the application, to libraries or system calls, are replaced by our

own stubs. Which allow encryption in reading and writing. This paper reviews productively used techniques to detect malicious apps through other apps and at central distribution points, i.e. app markets (II-A). At second, selected research approaches to detect malicious apps applied on-device or in app markets are presented (II-B).Then explained framework which provide security services to smartphones [5].

The paper selects major domestic and external web sites which provide cloud, VoIP, messenger, E-mail services and examines whether data are encrypted or not for constructing a safe Smart Work infrastructure,in which auther had explained about SSL/TLS (Definotion of SSL/TLS ,SSL/TLS Handshacking) which is the basis judgement of encryption of data,also auther explained about countermeasuresfor vulnerability,SSL modes [6].

Paper presented several algorithms of how to represent policy databases and how to perform policy checks without explicitly disclosing the total set of policies. This privacy-preserving set operation extends related work, which has assumed that parties trust each other [8].Author present analysis which shows that the proposed policy checks can be implemented efficiently in realistic systems.In this paper auther given a database of policies, determine if a packet matches any policy without revealing the database content or which specific policy matches the packet. This policy check should be performed in a single round without per- packet interactions between the policy provider and the policy checker.

BYOD security concerns and potential threats [7, 10] are define and targeted in different environment of access control and people in center,generally worked on data exfiltration,data tampering and data unavailibilty.

Formally define a subset of the existing se- curity framework of Android, which suffices to define proposed extensions rigorously,also described the exten- sions to the existing mechanism for incorporating usage constraints, and created a policy enforcement framework that incorporates usage policies while granting permissions to applications for accessing resources; and described and implemented an extended package installer that utilizes an easy-to-use and intuitive interface for allowing users to specify their constraints [11].

2.2 Existing Systems

Air Tight Networks organization protecting the enterprise from the onslaught of personal smart phones and tablets. Many important feature like authentication,detection malicious activities,messaging reports on emails.like wise a unique workflow (patent pending) to simplify BYOD management, automatic fingerprinting of smart phones and tablets: iPhone, iPad, Blackberry, Android, etc.,Using patented Marker Packet techniques, automatically and accurately detects all types of Rogue APs including Soft Rogues, and Mobile Wi-Fi Hotspots, Real-time quarantine of unapproved smart phones and tablets, all types of Rogues and unauthorized Mobile Wi-Fi Hotspots. Since this system is fully developed with complex architecture and it is equally costly.

The Cisco BYOD solution builds on the Cisco Borderless Network architecture and assumes best practices are followed in network infrastructure designs for campus, branch offices, Internet edge, and home office implementations. This comprehensive BYOD solution provided for wired, Wi-Fi, remote, and mobile access to the network, supporting across many device types and brands, and capable of enforcing the various policies across the spectrum of businesses and industries. In addition, as devices move from one context to another, for example from the corporate WiFi network to a public 3G/4G mobile network, the BYOD solution provide secure access while keeping the experience seamless for the user.

System is fully configured with network devices which nearly non affodable for small scale organization,where they always search for open soure sloution.Figure 2.1 describe high level architecture.

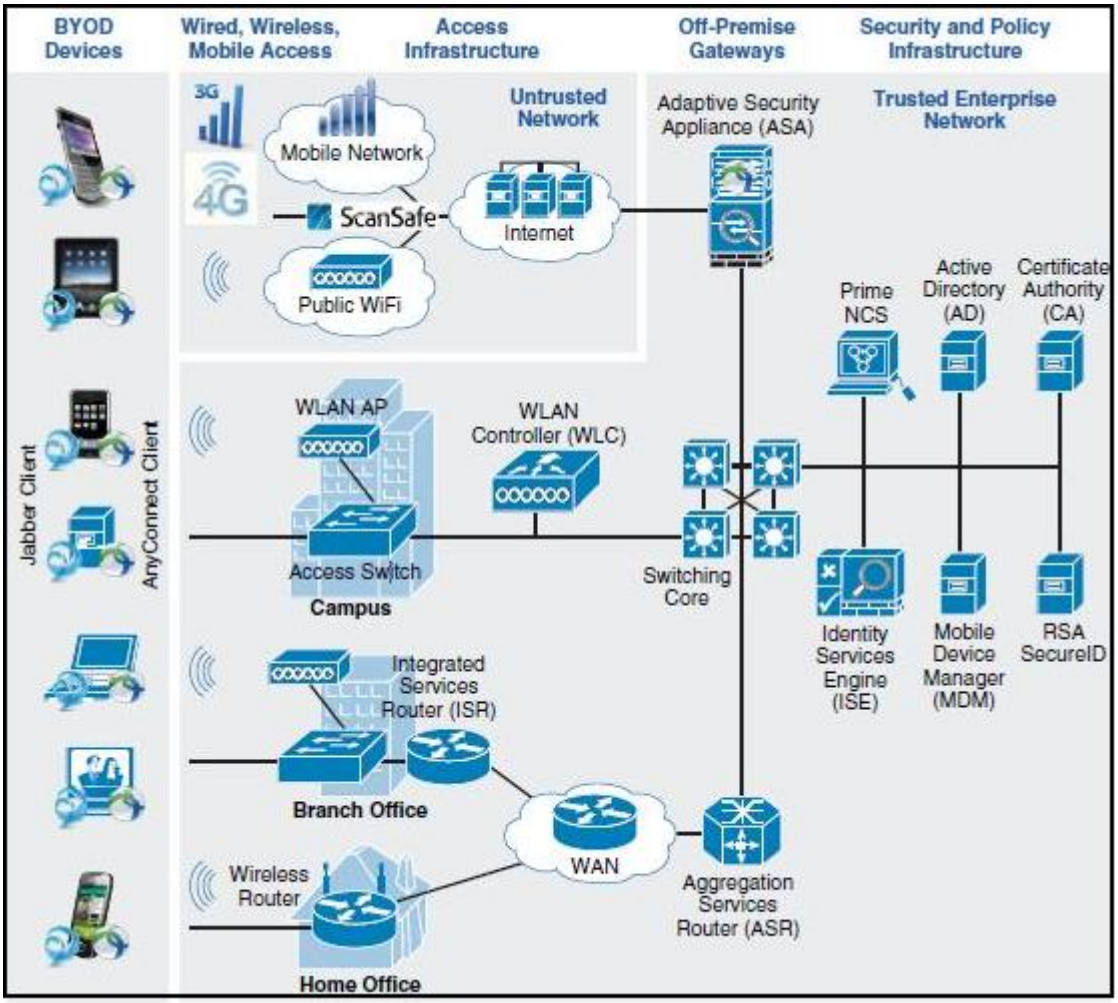

Figure 2.1: CISCO-High level BYOD Solution Architecture

Chapter 3

Policies and Services in BYOD context

System is based on Single-sign on ,Android device security policies and ssl connection.

3.1 Single-Sign On(SSO)

The Open Group defines Single Sign-On as a mechanism whereby a single action of user authentication and authorization can permit a user to access all computers and systems where that user has access permission, without the need to enter multiple passwords.

The most important security advantage that a SSO implementation offers is a common secure infrastructure, which can be carefully managed and protected [12]. Single sign-on reduces human error, a major component of systems failure and is therefore highly desirable but difficult to implement.

For example, in an enterprise using SSO software, the user logs on with their id and password. This gains them access to low risk information and multiple applications such as the enterprise portal. However, when the user tries to access higher risk applications and information, like a payroll system, the single sign on software requires them to use a stronger form of authentication. This may include digital certificates, security tokens, smart cards, biometrics or combinations thereof. Single sign on can also take place between enterprises using federated authentication. For example, a business partner's employee may successfully log on to their enterprise system. When they click on a link to your enterprise's application, the business partner's single sign on system will provide a security assertion token to your enterprise using a protocol like SAML, Liberty Alliance, WS Federation or Shibboleth. Your enterprise's SSO software receives the token, checks it, and then allows the business partner's employee to access your enterprise application without having to sign on.

Single sign on federated authentication also works with your employees. For example, an employee who is trying to access your outsourced benefits supplier to update their benefits information would click on the benefits link on your intranet. Your enterprise's single sign on software would then send a security assertion token to the benefits supplier. The benefits supplier's SSO system would then take the token, check it and grant access to your employee without making them sign on.

3.1.1 Single Sign On Benefits

Single sign on benefits are:

- Ability to enforce uniform enterprise authentication and/or authorization policies across the enterprise
- End to end user audit sessions to improve security reporting and auditing
- Removes application developers from having to understand and implement identity security in their applications
- Usually results in significant password help desk cost savings

Figure 3.1 depict single-sign on scenario.

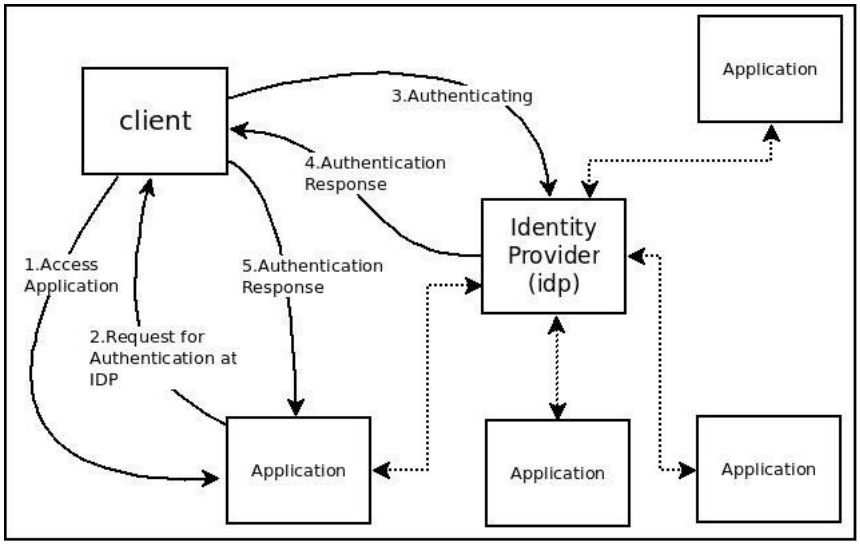

Figure 3.1: Basic Single-Sign on Paradigm

3.1.2 SAML 2.0

The Security Assertion Markup Language (SAML) standard defines a framework for exchanging security information between online business partners. It was developed by the Security Services Technical Committee (SSTC) of the standards organization OASIS (the Organization for the Advancement of Structured Information Standards).SAML is different from other security systems due to its approach of expressing assertions about a subject that other applications within a network can trust. The four main purposes behind the creation of the SAML standard are:

- Browser cookie limitations
- SSO interoperability
- Web services security

- Identity federation

There are two very important concepts that play an important role in the SAML standard:

- Identity Provider (IdP) is the system or the domain that asserts that a user has been authenticated and has associated attributes. In SAML, Identity Providers are also known as SAML authorities and Asserting Parties.
- Service Provider (SP) is the system or administrative domain that relies on information supplied by the Identity Provider.

The Service Provider decides if it trusts or not the information provided by the IdP. Service Providers are also known asRelying Parties due to the fact that they rely on information provided by an Identity Provider (Asserting Party). SAML consists of some block components that supports a number of use cases. First of all, these components permit transfer of identity, authentication, attribute and authorization information between autonomous systems that have a trust relationship. The core SAML specification defines the structure and content of both assertions and protocol messages used to transfer this information. SAML assertions carry statements about a principal that an asserting party claims to be true. The valid structure and contents of an assertion are defined by the SAML assertion XML schema. Assertions are usually created by an asserting party based on a request of some sort from a relying party, although under certain circumstances, the assertions can be delivered to a relying party in an unsolicited manner. SAML protocol messages are used to make the SAML-defined requests and return appropriate responses. The structure and contents of these messages are defined by the SAML-defined protocol XML schema.

3.1.3 SAML implementation for SSO

Prerequisites for authentication is single sign on must be enable in our computer.This system requires uses the SAML 2.0 for Single Sign-On authentication. SAML 1.1 is not supported.

The metadata document describes a service provider to an identity provider, including the following elements:

- The endpoint addresses for communication
- The X.509 certificates being used to encrypt and sign SAML assertions
- The SAML bindings supported by the service provider

To configure SAML settings for single sign-on from identity provider to system

- Gather information from your identity provider.
- Provide information to your identity provider.
- from Setup, click Security Controls, Single Sign-On Settings, and edit neccerary changes regarding host ip. Select
- You must enable SAML to view the SAML single sign-on settings.
- Specify the SAML version used by your identity provider.

- In SAML Single Sign-On Settings, click New.
- Give this setting a Name for reference within your organization.
- provider inserts the corresponding API Name value, which i can customize if necessary.
- If you are enabling just-in-time provisioning for security.
- Enter the id,this is often referred to as the entity ID for the identity provider.
- Organization's domains/server ip use for deployment, specifying whether to use the base domain or the custom domain for the Entity ID. we have to share this information with identity provider.
- Generate key certificate for secure connection.

3.2 Policy Rules

For server side and client side XACML version 2.0 used to write policy rules.

3.2.1 XACML

XACML(eXtensible Access Control Markup Language) is an OASIS standard that describes both a policy language and an access control decision request/response language (both written in XML). The policy language is used to describe general access control requirements, and has standard extension points for defining new functions, data types, combining logic, etc. The request/response language lets you form a query to ask whether or not a given action should be allowed, and interpret the result. The response always includes an answer about whether the request should be allowed using one of four values: Permit, Deny, Indeterminate (an error occurred or some required value was missing, so a decision cannot be made) or Not Applicable (the request can't be answered by this service).

The typical setup is that anyone wants to take some action on a resource. That is making a request to whatever actually protects that resource (like a filesystem or a web server), which is called a Policy Enforcement Point (PEP). The PEP will form a request based on the requester's attributes, the resource in question, the action, and other information pertaining to the request. The PEP will then send this request to a Policy Decision Point (PDP), which will look at the request and some policy that applies to the request, and come up with an answer about whether access should be granted. That answer is returned to the PEP, which can then allow or deny access to the requester. Note that the PEP and PDP might both be contained within a single application, or might be distributed across several servers. In addition to providing request/response and policy languages, XACML also provides the other pieces of this relationship, namely finding a policy that applies to a given request and evaluating the request against that policy to come up with a yes or no answer.

3.2.2 Policy and Policyset

At the root of all XACML policies is a Policy or a PolicySet. A PolicySet is a container that can hold other Policies or PolicySets, as well as references to policies found in remote

locations. A Policy represents a single access control policy, expressed through a set of Rules. Each XACML policy document contains exactly one Policy or PolicySet root XML tag.

Because a Policy or PolicySet may contain multiple policies or Rules, each of which may evaluate to different access control decisions, XACML needs some way of reconciling the decisions each makes. This is done through a collection of Combining Algorithms. Each algorithm represents a different way of combining multiple descisions into a single descision. There are Policy Combining Algorithms (used by PolicySet) and Rule Combining Algorithms (used by Policy). An example of these is the Deny Overrides Algorithm, which says that no matter what, if any evaluation returns Deny, or no evaluation permits, then the final result is also Deny. These Combining Algorithms are used to build up increasingly complex policies, and while there are seven standard algorithms, you can build your own to suit your needs.

3.2.3 Parameters and Notations of Rules

In the field of access control and authorization there are several closely related terms in common use. For purposes of precision and clarity, certain of these terms are not used in this specification.For instance, the term attribute is used in place of the terms: group and role. In place of the terms: privilege, permission, authorization, entitlement and right, we use the term rule.The term object is also in common use, but we use the term resource in this specification.Requestors and initiators are covered by the term subject.

Notation

This specification contains schema conforming to W3C XML Schema and normative text to describe the syntax and semantics of XML-encoded policy statements. The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this specification are to be interpreted as described in IETF RFC 2119.

The XACML policy syntax is defined in a schema associated with the following XML namespace:

urn:oasis:names:tc:xacml:1.0:policy

The XACML context syntax is defined in a schema associated with the following XML namespace:

urn:oasis:names:tc:xacml:1.0:context

The XML Signature [DS] is imported into the XACML schema and is associated with the following XML namespace:

http://www.w3.org/2000/09/xmldsig

Requirements

The basic requirements of a policy language for expressing information system security policy are:

- To provide a method for combining individual rules and policies into a single policy set that applies to a particular decision request.

- To provide a method for flexible definition of the procedure by which rules and policies are combined.
- To provide a method for dealingwith multiple subjects acting in different capacities.
- To provide a method for basing an authorization decision on attributes of the subject and resource.
- To provide a method for dealing with multi-valued attributes.
- To provide a method for basing an authorization decision on the contents of an information resource.
- To provide a set of logical and mathematical operators on attributes of the subject, resource and environment.
- To provide a method for handling a distributed set of policy components, while abstracting the method for locating, retrieving and authenticating the policy components.
- To provide a method for rapidly identifying the policy that applies to a given action, based upon the values of attributes of the subjects, resource and action.
- To provide an abstraction-layer that insulates the policy-writer from the details of the application environment.
- To provide a method for specifying a set of actions that must be performed in conjunction with policy enforcement.

Rule and Policy Combining

The complete policy applicable to a particular decision request may be composed of a number of individual rules or policies. For instance, in a personal privacy application, the owner of the personal information may define certain aspects of disclosure policy, whereas the enterprise that is the custodian of the information may define certain other aspects. In order to render an authorization decision, it must be possible to combine the two separate policies to form the single policy applicable to the request. XACML defines three top-level policy elements: <Rule>, <Policy>and <PolicySet>. The <Rule>element contains a boolean expression that can be evaluated in isolation, but that is not intended to be accessed in isolation by a PDP. So, it is not intended to form the basis of an authorization decision by itself. It is intended to exist in isolation only within an XACML PAP, where it may form the basic unit of management, and be re-used in multiple policies. The <Policy>element contains a set of <Rule>elements and a specified procedure for combining the results of their evaluation. It is the basic unit of policy used by the PDP, and so it is intended to form the basis of an authorization decision. The <PolicySet>element contains a set of <Policy>or other <PolicySet>elements and a specified procedure for combining the results of their evaluation. It is the standard means for combining separate policies into a single combined policy.

Combining Algorithms

XACML defines a number of combining algorithms that can be identified by a RuleCombiningAlgId or PolicyCombiningAlgId attribute of the <Policy>or <PolicySet>elements, respectively. The rule-combining algorithm defines a procedure for arriving at an authorization decision given the individual results of evaluation of a set of rules. Similarly, the policy-combining algorithm defines a procedure for arriving at an authorization decision given the individual results of evaluation of a set of policies. Standard combining algorithms are defined for:

- Deny-overrides,
- Permit-overrides,
- First applicable and
- Only-one-applicable.

In the first case, if a single <Rule>or ¡Policy>element is encountered that evaluates to "Deny", then, regardless of the evaluation result of the other <Rule>or <Policy>elements in the applicable policy, the combined result is "Deny". Likewise, in the second case, if a single "Permit" result is encountered, then the combined result is "Permit". In the case of the First-applicable combining algorithm, the combined result is the same as the result of evaluating the first <Rule>, <Policy>or <PolicySet>element in the list of rules whose target is applicable to the decision request. The "Only-one-applicable" policy-combining algorithm only applies to policies. The result of this combining algorithm ensures that one and only one policy or policy set is applicable by virtue of their targets. If no policy or policy set applies, then the result is "NotApplicable", but if more than one policy or policy set is applicable, then the result is "Indeterminate". When exactly one policy or policy set is applicable, the result of the combining algorithm is the result of evaluating the single applicable policy or policy set.

3.2.4 XACML context in Policy

XACML is intended to be suitable for a variety of application environments. The core language is insulated from the application environment by the XACML context, as shown in Figure 3.2, in which the scope of the XACML specification is indicated by the shaded area. The XACML context is defined in XML schema, describing a canonical representation for the inputs and outputs of the PDP. Attributes referenced by an instance of XACML policy may be in the form of XPath expressions on the context, or attribute designators that identify the attribute by subject, resource, action or environment and its identifier. Implementations must convert between the attribute representations in the application environment (e.g., SAML, J2SE, CORBA, and so on) and the attribute representations in the XACML context. How this is achieved is outside the scope of the XACML specification. In some cases, such as SAML, this conversion may be accomplished in an automated way through the use of an XSLT transformation.

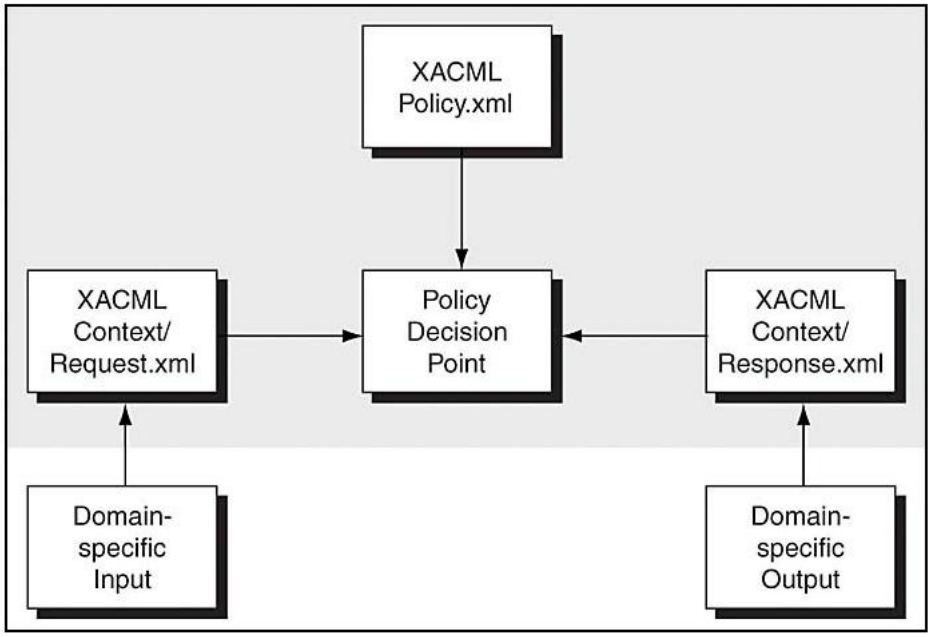

Figure 3.2: XACML Context

3.2.5 Policy Language Model

The policy language model is shown in Figure 3.3. The main components of the model are:

- Rule;
- Policy; and
- Policy set.

Rule

A rule is the most elementary unit of policy. It may exist in isolation only within one of the major actors of the XACML domain. In order to exchange rules between major actors, they must be encapsulated in a policy. A rule can be evaluated on the basis of its contents. The main components of a rule are:

- a target;
- an effect; and
- a condition.

The **target** of the rule defines the set of:

- resources;
- subjects; and
- actions

to which the rule is intended to apply. The <Condition>element may further refine the applicability established by the target. If the rule is intended to apply to all entities of a particular data-type, then an empty element named <AnySubject/>, <AnyResource/>or <AnyAction/>is used. An XACML PDP verifies that the subjects, resource and action identified in the request context are all present in the target of the rules that it uses to evaluate the decision request. Target definitions are discrete, in order that applicable rules may be efficiently identified by the PDP. The <Target>element may be absent from a <Rule>. In this case, the target of the <Rule>is the same as that of the parent <Policy>element. The **effect** of the rule indicates the rule-writer's intended consequence of a "True" evaluation for the rule. Two values are allowed: "Permit" and "Deny".**Condition** represents a boolean expression that refines the applicability of the rule beyond the predicates implied by its target. Therefore, it may be absent.

Policy

From the policy model shown in figure 3.3 one can see that rules are not exchanged amongst system entities.

Therefore, a PAP combines rules in a policy. A policy comprises four main components:

- a target;
- a rule-combining algorithm-identifier;
- a set of rules; and
- obligations.

An XACML <PolicySet>, <Policy>or <Rule>element contains a <Target>element that specifies the set of subjects, resources and actions to which it applies.

A system entity that calculates a <Target>in this way is not defined by XACML, but there are two logical methods that might be used. In one method, the <Target>element of the outer <PolicySet>or <Policy>(the "outer component") is calculated as the union of all the <Target>elements of the referenced <PolicySet>, <Policy>or <Rule>elements (the "inner components"). In another method, the <Target>element of the outer component is calculated as the intersection of all the <Target>elements of the inner components. The results of evaluation in each case will be very different: in the first case, the <Target>element of the outer component makes it applicable to any decision request that matches the <Target>element of at least one inner component; in the second case, the <Target>element of the outer component makes it applicable only to decision requests that match the <Target>elements of every inner component. Note that computing the intersection of a set of <Target>elements is likely only practical if the target data-model is relatively simple.

In cases where the <Target>of a <Policy>is declared by the policy writer, any component <Rule>elements in the <Policy>that have the same <Target>element as the <Policy>element may omit the <Target>element. Such <Rule>elements inherit the <Target>of the <Policy>in which they are contained.

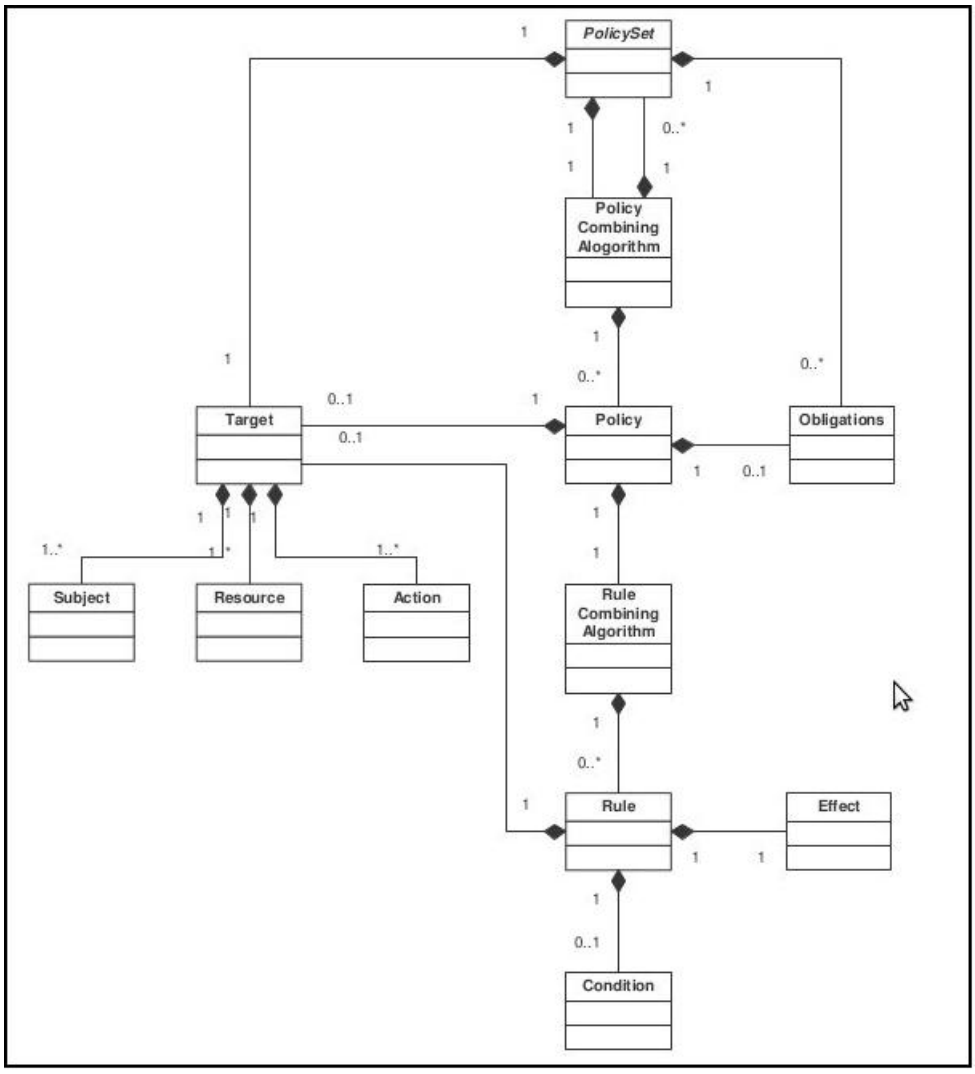

Figure 3.3: Policy Language Model

Chapter 4

Proposed System

The proposed System has the main component as securing BYOD devices i.e. MDM server which handle android based smart phones so those mobiles will be used inside the orgnization efficently and securely. Since the system is used for the purpose of giving internet privileges to employees devices, only basic operations from the MDM server side can be performed (like device lock, passcode policy, etc.). The system allows the user to enroll to the MDM server from both within the network (company WiFi) and also outside (using Cellular). When the user enrolls his/her device to the system via the private network, the device is given a session that can be used to access the internet.

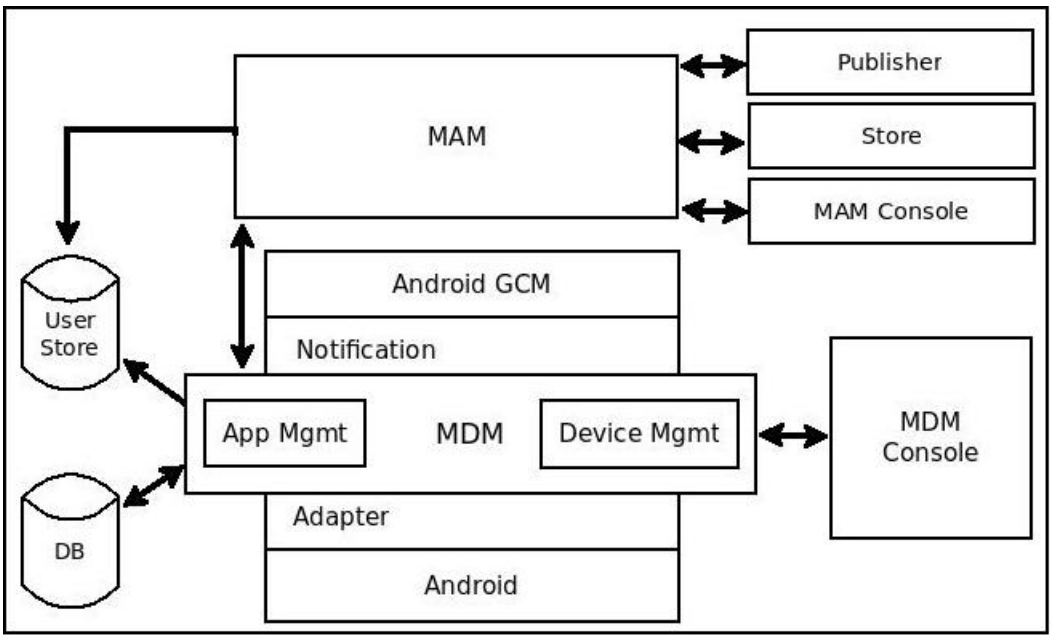

Figure 4.1: The Proposed System

- When the device connects to the private network, it checks with the router if it has a session. If so, then the device already has internet access in this network, else it will be redirected to the EMM server.
- The MDM server will check if the device is enrolled in the server.

1. If the device is already registered in the system, then it will direct the user to the WiFi Login page where the user would have to enter the LDAP credentials to get a new session.
2. If not, it will be directed to the appropriate EMM page. In the case of Android, it will be directed to the MDM Agent download page whereas for iOS, it will be directed to the EMM registration page.

- Upon successfully enrolling the device to the EMM server, the server will automatically create a new session for the device.

The important components are explained as follows:

4.1 User Android Device

Adroid devices(smartphone,tablet)are preferable with os android version 4.0 onwards. The basic requirement for device is ,that device should not be rooted.That is downloading from other servers should be enable.From android device interface user connects to MDM server through openn ssl connection by exchanging certificate which embbeded with android user file.

4.2 Mobile Device Management(MDM)

MDM enables organizations to secure, manage and monitor Android and iOS powered devices (i.e., smart phones, tablet PCs), on both corporate and personal (employee-owned) devices, irrespective of the mobile operator, service provider or the organization. The Device Management sub-module, which is within the MDM, enables the MDM to communicate with the MDM Console. The Notification module handles the Google Cloud Messaging (GCM) service for Android.These notification services send data from the MDM server to the respective users' devices. MDM uses the Adapter as a mediator to communicate with iOS and Android powered devices. The Application Management sub-module, which is within the MDM, enables communication between the MDM server and the MAM server. Administrators can connect their own user store to EMM. However, we do not allow a secondary user store to be connected.Android powered devices are enrolled to the MDM server using Device Provisioning process. This is an automated process that takes place in the backend. Administrators can assign policies with various criteria (e.g., lock password, length of the password, and more) to users, roles and platforms.

4.3 Massaging(Local and Android GCM)

the system uses local massaging and Google Cloud Messaging(GCM).Carbon server API from WSO2 uses massaging service to send massgaes to user email id.Detail procedure about GCM is discuss in results.

4.4 MDM Console

MDM console is administrator interface. It create users,roles,and policies and pass this information to mobile device management.

4.5 Mobile App Management(MAM)

MAM enables organizations to control the corporate applications on mobile devices, which are enrolled with the MDM. MAM has three key consoles to manage the corporate applications: Publisher, Store and MAM Console. Users who belong to the Publisher role can use the Publisher to publish, unpublish, deprecate or retire their mobile apps. While, users who belong to the Reviewer role can approve or reject mobile apps, which are submitted for approval. Users can use the Store to discover mobile apps easily and install apps to their own mobile devices. Administrators can use the MAM Console to manage users and policies; while, also being able to install and uninstall mobile apps in bulk.

4.6 Databases

Mysql is used to creat database named EMM_DB. Then imported all related tables from api service. Database includes,

1. device_awake
2. device_pending
3. device_policy
4. devices
5. featuregroup
6. features
7. featuretype
8. group_policy_mapping
9. notifications
10. platform_policy_mapping
11. platformfeatures
12. platforms
13. policies
14. policy_device_profiles
15. policy_priority
16. tenantplatformfeatures
17. user_policy_mapping

Chapter 5

Deployment and Digital Certificate Generation

Since secure BYOD is distributed heterogeneous system.Every code is createed using java platform ,but modules of this systems are runs on different platforms. Following is the deployment procedure of Android client, MDM server, and General server.

5.1 General Server Configuraton

1. Configure the DeviceMonitorFrequency parameter in the emm-config.xml file, which is in the PRODUCT_HOME/repository/conf/ directory. Specify this value in milliseconds. The server uses this parameter to determine how often the devices enrolled with server need to be monitored. By default, this value has been configured to 60000ms (1min).

   <DeviceMonitorFrequency>60000</DeviceMonitorFrequency>

2. Navigate to the sso-idp-config.xml file, which is in the PRODUCT_HOME/ repository/conf/ directory and modify the 'localhost'to server IP address.

5.2 Android Client Configuraton

1. Using ADT bundle(eclipse 4.0 onwards) needed to develope android client. SERVER_IP, SERVER_PORT and SERVER_PROTOCOL are specified into commonUtilities.java for coonecting android device to server.

2. Add location of android client (.apk) file into MDM server inside PRODUCT_HOME/repository/deployment/server/jaggeryapps/emm/config/ directory.

5.3 Digital Certificate Generation Procedure

for generating digital certificate for android client and MDM server openssl is used. If the SSL certificate is self signed, then created a BKS file that is compatible with Android. Name the BKS file to emm_truststore.bks and add it inside Android projects res/raw/ directory.

The BKS file password needs to be added as the value for the TRUSTSTORE_PASSWORD parameter.

Following is the procedure for generating three certificates:CA certificate,RA certificate and IA certificate.

1. Generate a self signed Certificate Authority (CA) certificate and convert the certificate to .pem format using the following commands:

   - openssl genrsa -out ca_private.key 4096
   - openssl req -new -key ca_private.key -out ca.csr
   - openssl x509 -req -days 365 -in ca.csr -signkey ca_private.key -out ca.crt -extensions v3_ca
   - openssl rsa -in ca_private.key -text ¿ ca_private.pem
   - openssl x509 -in ca.crt -out ca_cert.pem

2. Generate a Registration Authority (RA) certificate signed it with the CA and convert the certificate to .pem format using the following commands:

   - openssl genrsa -out ra_private.key 4096
   - openssl req -new -key ra_private.key -out ra.csr
   - openssl x509 -req -days 365 -in ra.csr -CA ca.crt -CAkey ca_private.key -set_serial 02 -out ra.crt -extensions v3_req
   - openssl rsa -in ra_private.key -text ¿ ra_private.pem
   - openssl x509 -in ra.crt -out ra_cert.pem

3. Generate the SSL certificate based on server IP address:

   - openssl genrsa -out ia.key 4096
   - openssl req -new -key ia.key -out ia.csr
   - openssl x509 -req -days 730 -in ia.csr -CA ca_cert.pem -CAkey ca_private.pem -set_serial 044324343 -out ia.crt

4. Export the SSL, CA and RA files as PKCS12 files with an alias.

   - openssl pkcs12 -export -out KEYSTORE.p12 -inkey ia.key -in ia.crt -CAfile ca_cert.pem -name "wso2carbon"
   - openssl pkcs12 -export -out ca.p12 -inkey ca_private.pem -in ca_cert.pem -name "cacert"
   - openssl pkcs12 -export -out ra.p12 -inkey ra_private.pem -in ra_cert.pem -chain -CAfile ca_cert.pem -name "racert"

5. Copy the three P12 files to PRODUCT_HOME/repository/resources/security directory.

6. Import the generated P12 files as follows:

- keytool -importkeystore -srckeystore KEYSTORE.p12 -srcstoretype PKCS12 -destkeystore wso2carbon.jks
- keytool -importkeystore -srckeystore KEYSTORE.p12 -srcstoretype PKCS12 -destkeystore client-truststore.jks
- keytool -importkeystore -srckeystore ca.p12 -srcstoretype PKCS12 -destkeystore wso2emm.jks -importkeystore -srckeystore ra.p12 -srcstoretype PKCS12 -destkeystore wso2emm.jks

7. update following parameters like alias,and password for wso2emm.jks file into emm-config.xml.

    - EMM Keystore file location
    - EMM Keystore type
    - EMM Keystore password
    - Certificate authority certificate alias
    - Certificate authority private key password
    - Registration authority certificate alias
    - Registration authority private key password

Chapter 6

Implementation Results

A system for above proposed system is developed. Following are the main components of the implementation.

6.1 Android Client Module

This module gives the proposed system an interface to interact with server.A WiFi enabled android client receive invitation from Admin MDM server.After creating employee user profile on server, admin invite user by sending invitation on user's emial id which mentioned in profile. This module consists of features like:

- Invitation from Administrator(Registration): Employee use their mobile to open the registration email from administrator(server) and click on the URL provided. The Androd Agent application will get downloaded. Figure 6.1 depicts the same.Once device successfully emrolled with server,user screen get redirect to server ,from there user can download agent, See Figure 6.2.

- Once agent get downloaded and installed (during installation policy agrrement set by admin, has to accept by user) successfully on android device user then register by providing domain(server IP), Username(any name), password(any password) as shown in Figure 6.3.

- Agent gives provision to set password(pin),so that MDM server can not touch personal data.

6.2 MDM server

MDM server can be start from terminal command prompt by command "./wso2server.sh". On javascript enabled web browser server can open by entering url "http://172.16.90.168:9763/emm". Admin can enter default username and password to start MDM console. On console admin will able to creat users,provide role,enforceing policy.

- User creation: On the Configurations tab in the Console, click **Users**.Admin can either view existed users or can add new user.As Shown in Figure 6.5

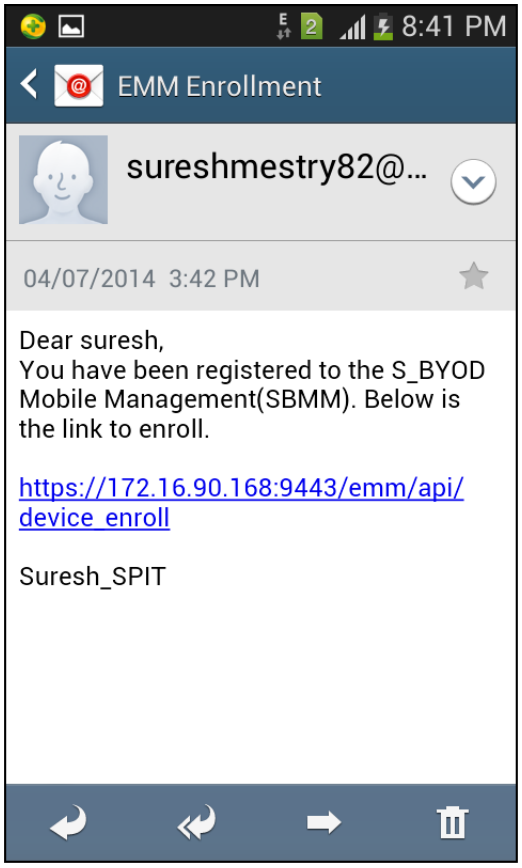

Figure 6.1: Enrollment invitation on user email-id

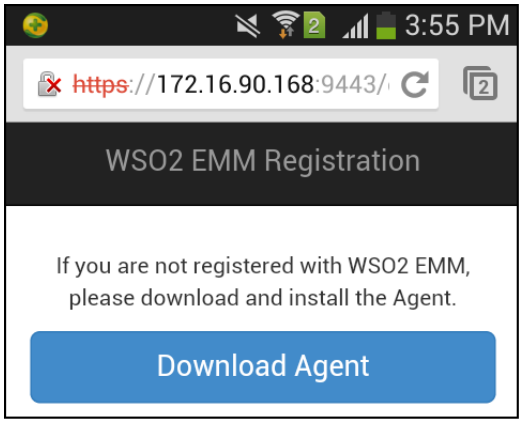

Figure 6.2: Android agent download

- Creating and Assigning Role:Once user is created, Admin either create a new role for that user or admin can assign already existed role as per employees job type.Figure 6.6 depict role creation and Figure 6.7 depict assign role to that user.

  Optionally, Admin can select the users to be assigned to this role. The selected users appear in the right-hand list. Created roles can be filter by using filter text box.

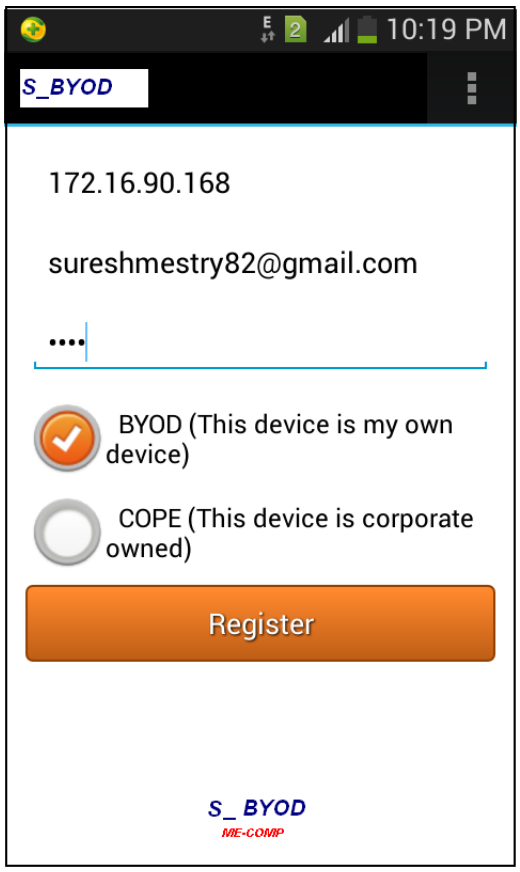

Figure 6.3: Registration

1. If admin wish to specifically select a user, click on the user's name in the left-hand list.
2. If admin wish to add all the displayed users to the selected list, click two right arrows.
3. If admin wish to remove a user from the selected list, click on the user's name in the right-hand list.
4. If admin wish to remove all the selected users, click two left arrows.

- Inviting users in a role:On the Configurations tab in the MDM Console, click Roles. Search for the role and click Invite in the Action column. Click Ok. The users who have been added to this role will receive an email inviting them to register with MDM.

- Roles and Permissions: The following are the roles that are available by default in MDM. These role are default, so that when admin want to assign roles to internal or equivalent to admin user ,these role are assigned to them.
  - administrator - Role assigned to the super tenant administrator by default.
  - mdmadmin - Role assigned to the MDM tenant administrator. This administrator will have full control over the MDM interface with the exception of being able to create, edit or delete MDM administrators.

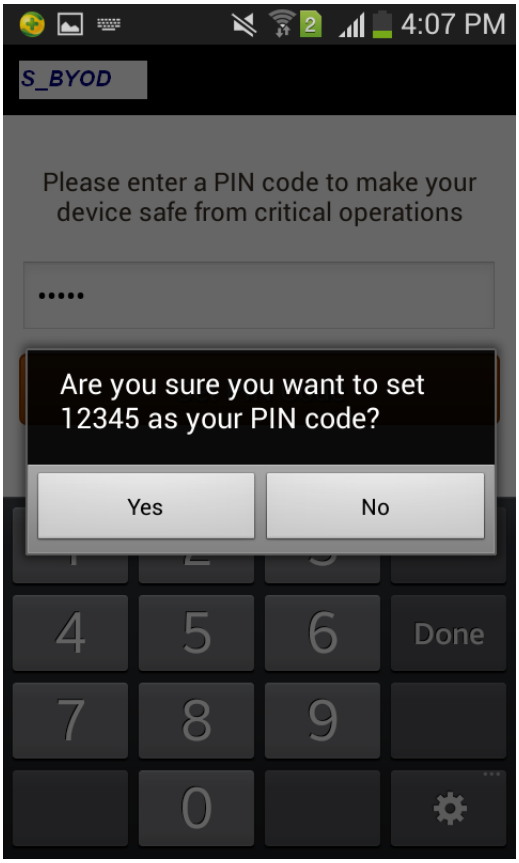

Figure 6.4: Setting PIN to avoid personal data to be access by admin

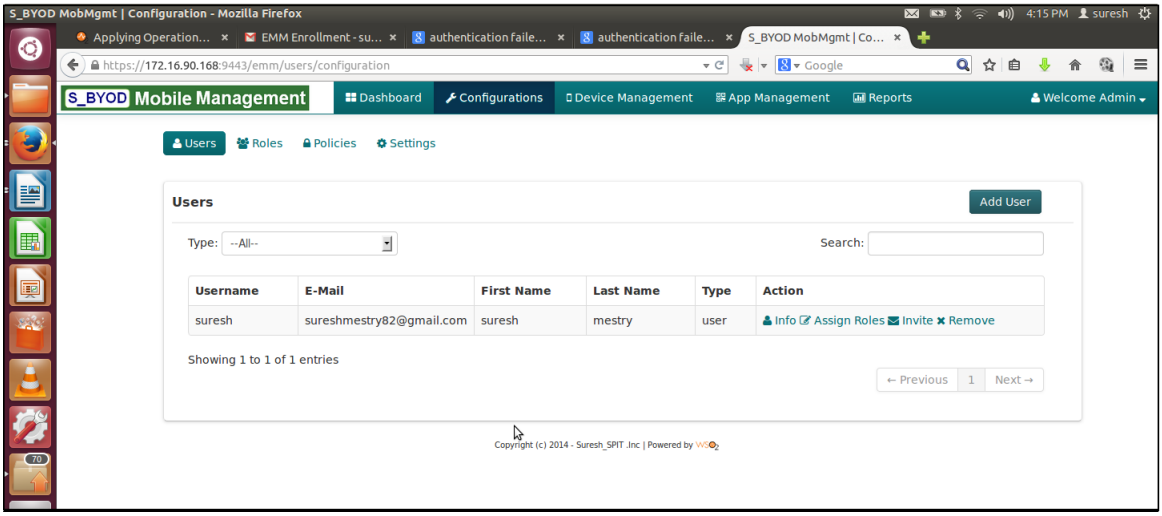

Figure 6.5: Creation of User

– Internal/publisher - Users in this role will be allowed to create new mobile apps (i.e., the author of a mobile app belong to this role).

– Internal/reviewer - Users in this role are considered as the store reviewers. Every mobile app needs to be reviewed by a user in this role, before the mobile app is

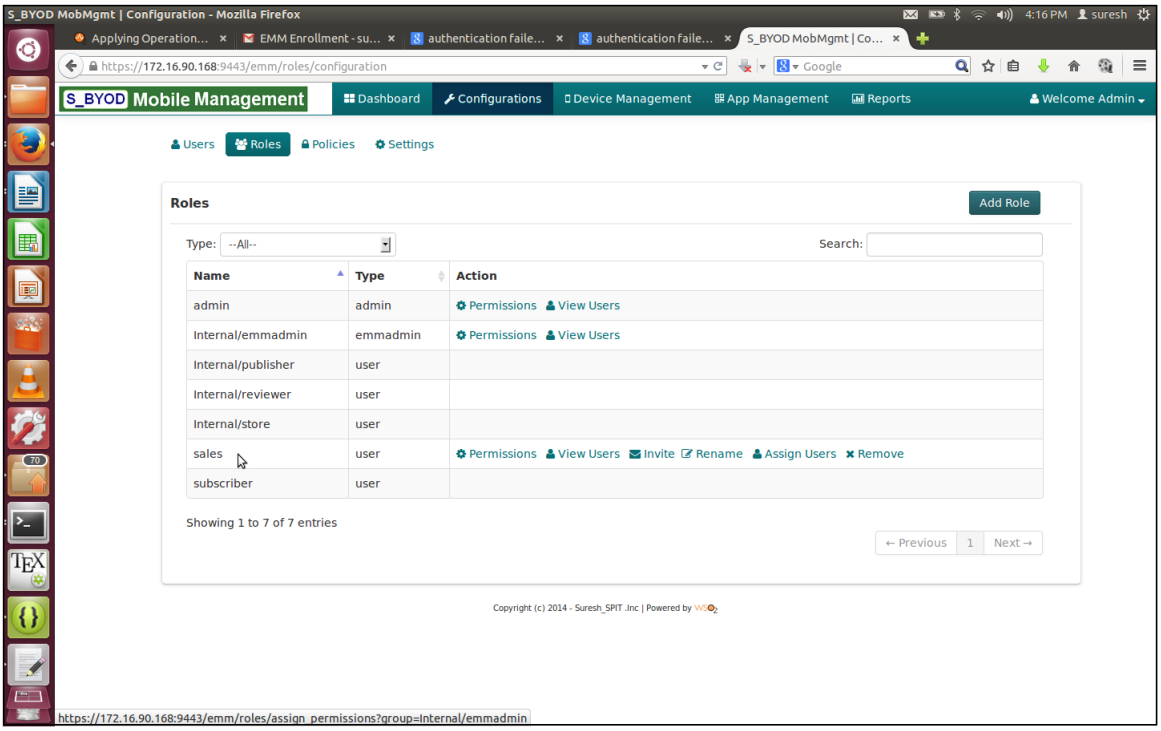

Figure 6.6: Creating a New Role

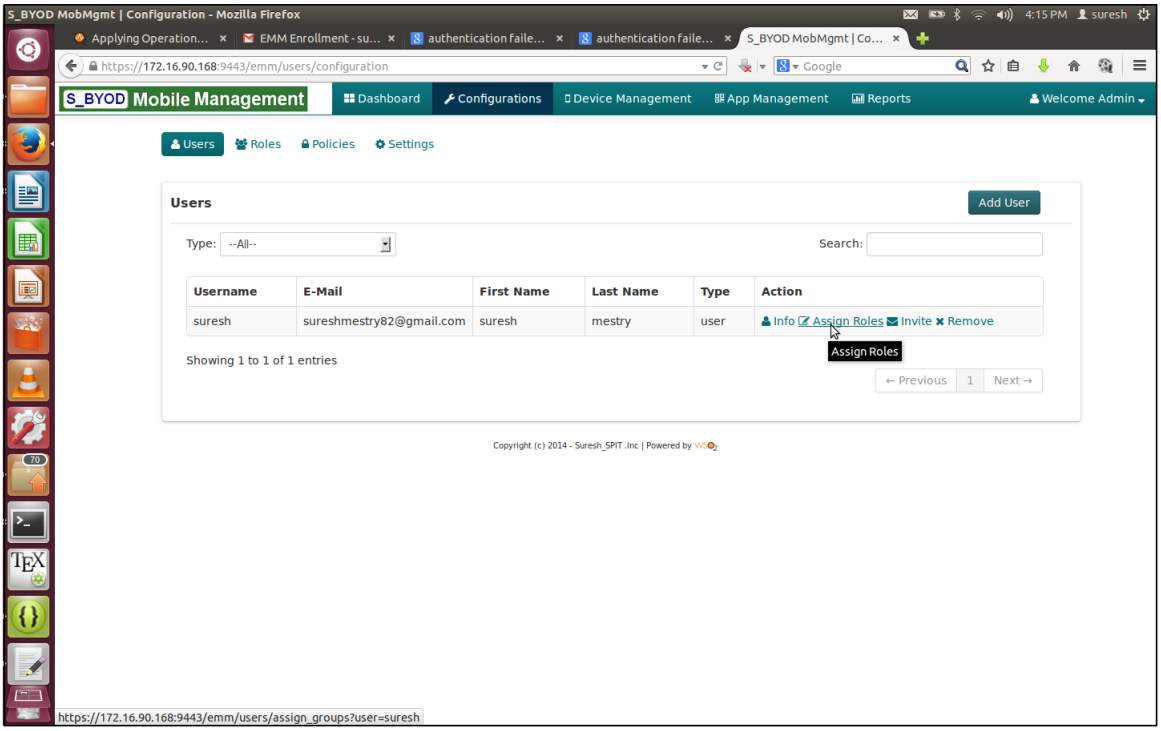

Figure 6.7: Assigning Roles

published by the admin into the Store.

– Internal/store - Users in this role will be allowed to access the store and interact

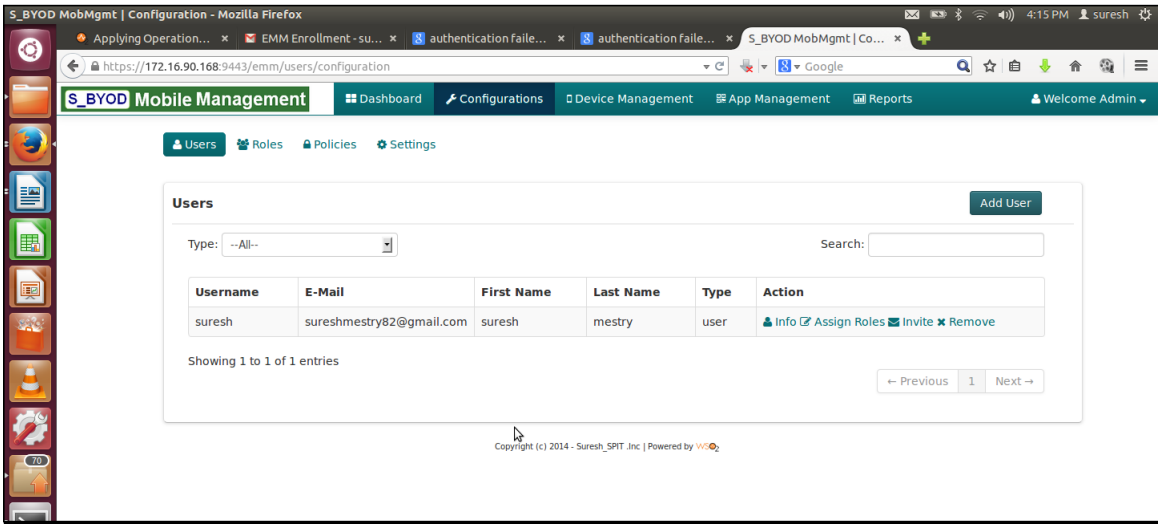

Figure 6.8: Inviting users by sending Email

with the store mobile apps.

- subscriber - This is a role that is used for backend processing in MDM to communicate with WSO2 API Manager. As a result, users can not be assigned to this role.
- private-(username) - Users private role. Every user in the MDM are automatically associated with a role that is created by prefixing their username with private.
- This role is used to control per user permissions.
- Internal/everyone - This is a system reserved role to create system operations.

- Permissions associated with user roles:
  - Administrator:The super tenant administrator belongs to this role. By default, a super tenant administrator will have full control on all the MDM Consoles.
  - mdmadmin:By default, no permissions will be assigned to this feature. The super tenant administrator has to assign users to this role.
  - Internal/publisher:Log in to the Publisher and Store, Create mobile apps, Submit mobile apps to be reviewed, Publish approved mobile apps, Unpublish mobile apps, Deprecate published mobile apps. Retire deprecated mobile apps, Deprecate unpublished assets.
  - Internal/reviewer:Log in to the Publisher, Approve or reject mobile apps that have been submitted to be reviewed.
  - Internal/store:Log in to the Store, Bookmark apps, Install apps.
  - private-(username):By default, only the login permission to the Store and Publisher are assigned to this role. However, if there are permissions that need to be allowed to specific users, they can be assigned using this role. Administrators need to replace (username) with the respective user's username.

6.3 Registering with Google Cloud Messaging

Their are two options are available for messaging ,local and GCM.Local messaging allow device to contact the MDM server periodically.GCM will send a notification to the device when there are pending operations available for a specific device. If GCM has been selected as the notifier type, register with Google Cloud Messaging (GCM). Figure 6.9 and Figure 6.10 shows the sender id and server api key.

If GCM has been selected as the notifier type, the following configurations need to be added:

- API keys - The API key that you received after registering with Google Cloud Messaging (GCM).
- Sender IDs - This refers to the project IDs. For more information, see Retrieve the Sender ID.

Following are the instructions to register with Google Cloud Messaging (GCM):

1. Create an APIs project
2. Create a server key
3. Retrieve the Sender ID

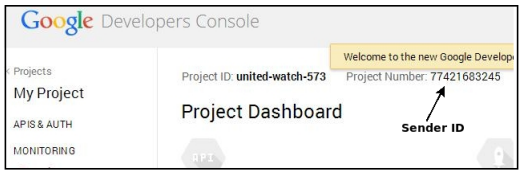

Figure 6.9: Project id is sender-id

Figure 6.10: Server API key

6.4 Operations on user mobile

Once user are enrolled into organization infrastructure,MDM assign roles and assign policies to user device. When user enrolled,its device information taken by MDM. On Device Management tab,admin can view enrolled all devices, and perform some operation on that selected device like mute device, passcode policy, device lok etc. See Figure

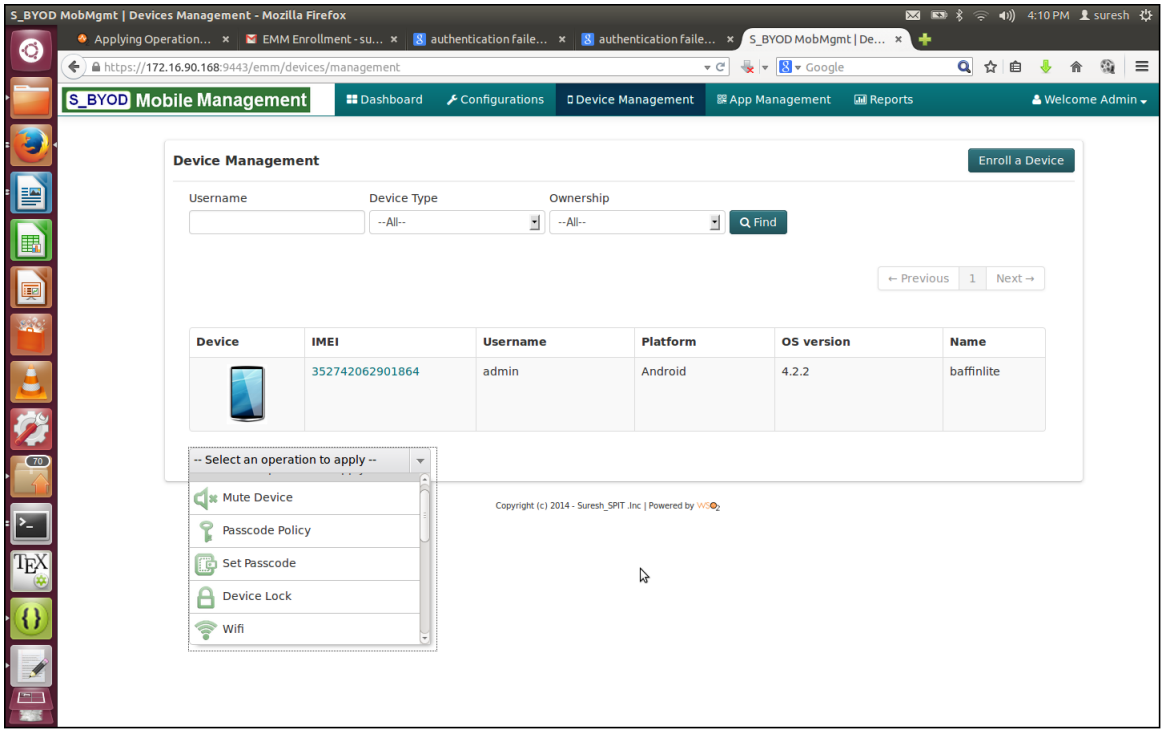

Figure 6.11: Operation perform by admin on user device

6.5 MDM Statistics

MDM provide statistic based on number of users enrolled See Table 6.1.

6.6 Comparison

Following is comparison of top 10 available BYOD security solution.All these solution are paid service,none of them is opensource.The Proposed system is open source ,thats why it is less costly/free as compare to below solution.It supports android and IOS.

Table 6.1: Statistics

Statistic	Description
Devices by ownership	This depicts the percentage of users, based on the device ownership (BYOD or COPE) that they belong to.
Devices by OS	This depicts the percentage of users, based on the device mobile OS (Android or iOS) that they belong to.
Devices by policy compliance	This depicts the percentage of users, that are compliant and non-compliant to the enforced policy.

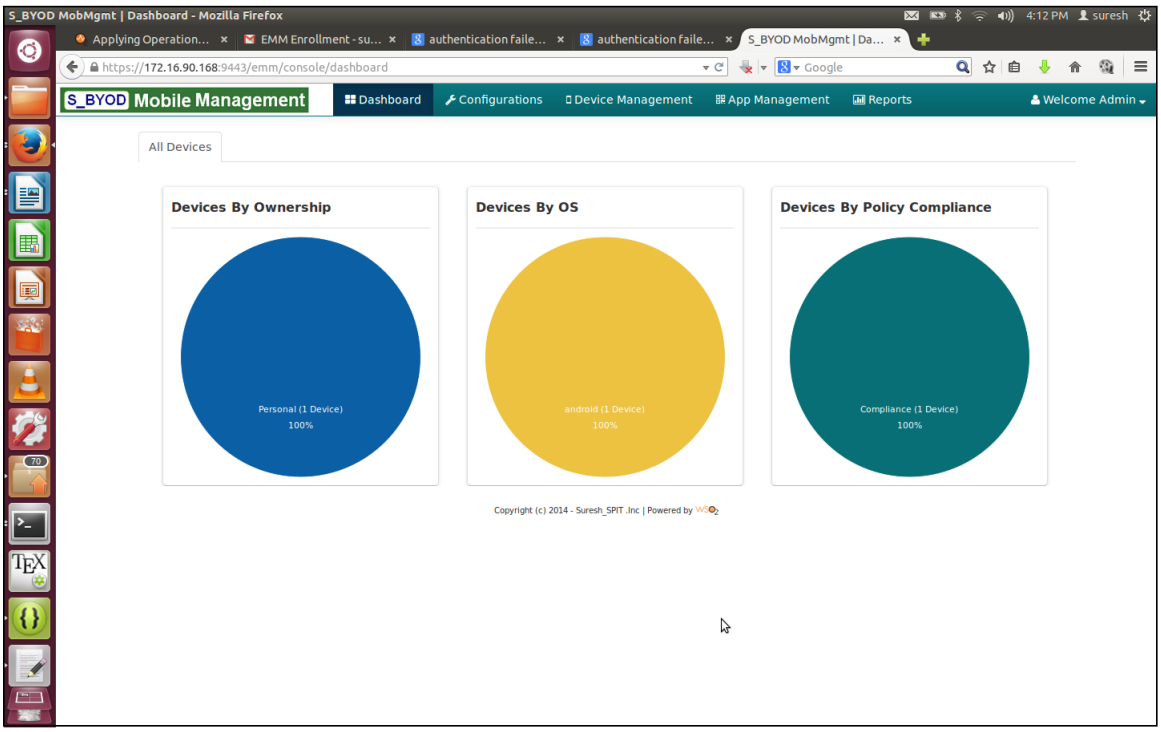

Figure 6.12: dashboard statistics

Name of security solution	**Provider support**	**Operating system support**	**Policies**	**Costly/Lower costly/Open source**
Fortinet BYOD	No cloud support	Android, BlackBerry,	Firewall, Intrusion Prevention, Antimalware/Antivirus, Application Control, Data Loss Prevention, Web Content Filtering, VPN (SSL and IPsec, Two-factor Authentication.	Lower Cost
ForeScout	Cloud based	iOS, Android, Symbian, BlackBerry, Windows, and webOS	ForeScout CounterACT, ForeScout Mobile Security Module and ForeScout MDM.	Costly
HuaWei BYOD	No cloud support	iOS, Android	mobile terminal security, network transmission security, application security, sensitive data security, and security managemen	Lower Cost
FixMo BYOD	Cloud based	Android	Device-level security,container level security,such as OS rooting or policy tampering.	Lower Cost
ZenPrise BYOD	Cloud based,Hybid device	iOS, Android	Firewall, Intrusion Prevention, Antimalware/Antivirus, Application Control	Costly
Good Technology BYOD	No cloud support	iOS, Android, Symbian, BlackBerry,	Data and application security	Lower cost
MobileIron BYOD	No cloud support	iOS, Android, Symbian, BlackBerry,	Device Choice, User Experience and Privacy Trust Model, App Design and Governance,Liability, Economics, Sustainability	Lower cost

Figure 6.13: Comparison between BYOD Security Slolution

Chapter 7

Conclusions and Future Scope

The proposed system was developed taking in mind the benefits of the employees and organization. In this work i presented a framework for modeling the behaviour of Android applications and verifying their compliance w.r.t. a security policies. Furthermore, i applied this method to a specific context, i.e., securing BYOD-based virtual organisations, and we detailed the features of a prototype under implemen- tation.The developed system is based on open source APIs,due to this cost matrix is reduce.It is more benificial to organization such as small scale companies to adapt this system.Using SAML xACML technology for security policy,i achieve basic but a complete security framework.In the result we can clearly observe that the system not only adapt organization needs but also user friendly system which keep personal data untouched by organization.

7.1 Future Scope

There are quite a few things that can be polished or add in the future work.

- Future work will consist of creating API for other operating platform of the user device.So Any other device which is IOS,Blackberry,Sambian powered will support the system.
- This system is limited either wi-fi or wired connection.But user can not jump from wifi-wired or from 2G to 3G/4G.

Bibliography

[1] Alessandro Armando, Gabriele Costa, Alessio Merlo, Bring Your Own Device, Securely, *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, ACM New York, NY, USA , pp. 1852-1858 March 2013.

[2] Rebecca Copeland, Noel Crespi, Analyzing Consumerization - Should Enterprise Business Context Determine Session Policy?, 2012 IEEE*Proceedings of 16th International Conference on Intelligence in Next Generation Networks (ICIN)*, pp.187-193 ,2012.

[3] Prashant Kumar Gajar, Arnab Ghosh, Shashikant Rai,BRING YOUR OWN DEVICE (BYOD): SECURITY RISKS AND MITIGATING STRATEGIES*Proceedings of Journal of Global Research in Computer Science* , Volume 4, No.4, April 2013.

[4] Sean Chung, Sam Chung, Teresa Escrig, Yan Bai,Barbara Endicott-Popovsky,2TAC: Distributed Access Control Architecture for Bring Your Own Device Security ,' *proceeding of the International Conference on BioMedical Computing*, 2012.

[5] Dennis Titze, Philipp Stephanow, Julian Schutte,'A Configurable and Extensible Security Service Architecture for Smartphones,' *IEEE Transactions on 27th International Conference on Advanced Information Networking and Applications Workshops*, 2013.

[6] Dae Hyeob Kim, Ji Hoon Gong, Won Hyung Park, Neo Park, 'Vulnerability of Information Disclosure in Data Transfer Section for Safe Smartwork Infrasturucture',*IEEE transaction on International Conference on Information Science and Applications (ICISA), 2013* , pp. 1 - 3, 24-26 June 2013.

[7] Dennis Gessner, Joao Girao, Ghassan Karame, Wenting Li 'Towards a User-Friendly Security-Enhancing BYOD Solution'*Proceedings of the NEC technical journal*, Vol.7 No.3/2013.

[8] Hanay, Y.S. , Wolf, T.,'Techniques for Policy Enforcement on Encrypted Network Traffic', *Proceedings of the Military Communications Conference, 2009. MILCOM 2009. IEEE* , PP. 1-7,2009.

[9] Ioannis Koskosas,'A Short Literature Review in Information Systems Security Approaches,' *Proceeding of the IJAKECS*, Vol.1 No.1,pp.1-7, January- June 2013.

[10] Antonio Scarf, 'New security perspectives around BYOD' *IEEE Transactions Seventh International Conference on Broadband, Wireless Computing, Communication and Applications*, Page(s): 446-451,2012

[11] Mohammad Nauman, Sohail Khan, Xinwen Zhang,'Apex: Extending Android Permission Model and Enforcement with User-dened Runtime Constraints', *ASIACCS '10 Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security*, Pages 328-332,2010

[12] Ionut Andronache, Claudiu Nisipasiu,'Web Single Sign-On Implementation Simple-SAMLphp Application', *Journal of Mobile, Embedded and Distributed Systems*, vol. III, no. 1, 2011

Acknowledgement

I have great pleasure in presenting the report on **BYOD-Secure Integration of Mobile Devices Into Company Network Infrastructure.** I take this opportunity to express my sincere thanks towards my guide **Dr.Y.S.Rao,** Head of Department, Department of Electrinics and Telecommunication Engineering, S.P.I.T., Mumbai for providing the technical guidelines and suggestions regarding line of work. I would like to express my gratitude towards his constant encouragement, support and guidance through the development of project.

I thank **Dr. D. R. Kalbande,** Head of Department, Computer Engineering, S.P.I.T., Mumbai for his encouragement during progress meeting and providing guidelines to write this report.

I thank **Prof. Anand. A. Godbole,** M.E. co-ordinator, Department of Computer Engineering, S.P.I.T., Mumbai for being encouraging throughout the course and for guidance.

I also thank the entire staff of S.P.I.T., Mumbai for their invaluable help rendered during the course of this work. I wish to express my deep gratitude towards all my colleagues of S.P.I.T., Mumbai for their encouragement.

Mr. Suresh R. Mestry
Roll No. 2012430011
Registration No.260