International Journal of Applied Research 2018; 4(12): 339-341

ISSN Print: 2394-7500
ISSN Online: 2394-5869
Impact Factor: 5.2
IJAR 2018; 4(12): 339-341
www.allresearchjournal.com
Received: 16-10-2018
Accepted: 20-11-2018

Satish Kumar Ray
Assist. Prof, Dept. of Comp. Science., GEC Ajmer
Rajasthan, India

Correspondence
Satish Kumar Ray
Assist. Prof, Dept. of Comp. Science., GEC Ajmer
Rajasthan, India

A survey on the java virtual machine to implement secure platform in mobile devices

Satish Kumar Ray

Abstract
The growth of the applications and services marketplace for mobile devices is presently over-involved by the shortage of a flexible and reliable security infrastructure. The event and adoption of a replacement generation of mobile applications depends on the tip user's ability to finely manage system security and management application's behavior. The virtual execution surroundings for mobile software system and services should support the protection needs of users and applications. This paper proposes an extension to the protection design of the Java Virtual Machine for mobile systems, to support fine-grained policy specification and run-time control. Access management selections are supported system state, application and system history knowledge, still as request specific parameters. The implementation is running on desktops and on mobile devices, proving the high level of flexibility and security, with wonderful performance provided by the extended design.

Keywords: Mobile device, java virtual machine, secure mobile platform, trusted mobile

1. Introduction
Mobile computing and technology has evolved staggeringly over the last decade. New mobile computing devices with higher style and multiplied capabilities are discharged often on the market. Consequently, rich mobile services like e-mail, scheduler, contact synchronization and even scaled-down versions of word processors, spreadsheets and presentation package became more and additional common among mobile users, especially in the sector.

However, the protection model prevailing at mobile platforms does not supply the pliability needed to support the market enlargement. The trust model enforced at mobile platforms presently is binary: sure applications square measure given all requested rights whereas the untrusted ones square measure fastened out of the platform utterly. To be deemed as sure, an application should carry a legitimate certificate, provided by the platform supplier or network operator. This suggests that application developers should have direct agreements with the certifying parties. As a consequence, the marketplace for mobile software development is inherently closed to third-party developers. Furthermore, this security models lack the ability for fine-grained, user-defined security policy definition and social control - e.g. to regulate the amount of SMSs (Short Message Service) sent, limit the information measure utilized by an application, etc. the most reason behind the rigidity of the security model is found in demanding resource constraints of initial mobile computing devices and additionally in bound market forces.

In this paper, we have a tendency to propose associate extended security design and policy model to deal with the dearth of flexibility of the current security model utilized at mobile computing platforms. The projected model has the potential to open up the mobile device package market to third-party developers and it additionally empowers the users to tailor security policies to their needs in an exceedingly fine-grained, personalized manner. Our work focuses on the Java 2 small Edition (J2ME) - one of the foremost wide used virtual machine execution environments for mobile computing devices nowadays.

Our main contribution is that the style and implementation of associate extended version of the present J2ME, that we have a tendency to call xJ2ME, from extended J2ME. xJ2ME permits runtime enforcement of a far a lot of communicatory category of security policies compared to the present progressive, allowing for a fine-grained behavior management of

individual applications. Furthermore, initial evaluations show no important, even noticeable, performance overheads.

2. Motivation

To better illustrate the constraints of the prevailing security model for mobile applications and clarify the motivation for the work bestowed during this paper, we have a tendency to gift the subsequent example. Alice is traveling together with her new automobile equipped with a film system that is connected to the surface world via a UMTS (Universal Mobile Telecommunications System) connection. As she enters Florence, her itinerant detects a traveler guide service provided by the native traveler information workplace. If Alice permits her itinerant to attach to the service and transfer the corresponding applications programme, the navigation system are able to show sites of historical interest and restaurants in shut proximity, and transfer additional info. However, the consulting service of her sure computing platform doesn't acknowledge the signature of the applications programme and, therefore, sandboxes it from the navigation system. Annoyed by the actual fact, she forces the platform to treat it as a sure application and enjoys the traveler sites within the area. Afterwards, Alice regrets her alternative once she discovers that the applications programme didn't solely retrieve the knowledge needed, however additionally it downloaded varied photos, causing AN unwanted, expensive quantity of network information measure consumption. Moreover, in a very few areas while not direct UMTS property; the applications programme used the expensive multimedia system Messaging Service (MMS) service to transfer information.

Although all the technology for supporting advanced use-scenarios is out there, the dearth of trust and security for mobile services makes complicated applications unusable. The execution setting ought to enable users to manage the behavior of the applications running on their devices on a much finer grain. Some concrete samples of such security policies, none of that area unit presently enforceable are: (1) the appliance will solely send SMS to specific phone numbers, and for a price that doesn't exceed three monetary unit per day; (2) the appliance cannot create international calls, nor phone calls on a knowledge affiliation to a premium phone number. Traditional phone calls area unit allowed solely on weekends and after-work hours; (3) the appliance might not generate more than 300Kb of traffic per session, over the UMTS connection; (4) found out most variety of MMS to be sent per unit of your time (hour/day/month/etc.). The solution that we have a tendency to propose not solely permits users to define their own policies for every of the applications to be executed on the platform, however additionally permit for fine-grained behavior management of these applications. We have a tendency to accomplish this by process an appropriate policy model and by extending the current security design of J2ME to produce for versatile run-time policy analysis and social control.

3. Related Work

Two most generally deployed mobile execution environments are. NET and Java frameworks. The previous is supported only by Windows-based platforms that restrict the movability of the applications written for the. NET framework. This is often not the case with the applications developed for the Java framework.

In the case of. NET framework, application code is translated into Common Language Runtime (CLR) and dead under the protection policies of the beneath lying package. The security policy on the device is sometimes set by the service supplier (e.g. Cingular, Sprint, T-Mobile). To provide a unique device policy, a special agreement with the service supplier is important. This effectively locks out small code developers from the market since the method is prolonged and expensive. Windows Mobile Security Model is based on a 3 permission tier, that square measure granted per application:

- Privileged: will decision any API, have full access to the registry, classification system, and may install certificates. Very few applications ought to run as privileged.
- Normal: cannot access the privileged areas •Blocked: doesn't enable application execution.

The Security Policy model of mobile devices running Windows OS offers no mechanisms to line fine-grained access control for system resources. As so much as we all know, no work has been worn out extending the. NET Security Policy model for mobile applications. As with reference to the Java framework, versatile security models have gathered respectable attention within the past. With Java a pair of normal Edition it's doable to use various Security Managers - categories implementing security-relevant operations. However, thanks to the restricted capabilities of mobile devices, the J2ME security design is, by design, not extensile, and so doesn't support this practicality. Users cannot specify various Security Managers; they cannot extend nor customize the predefined security policies.

4. Java Security Architecture

The two most well-liked platforms for mobile application development these days area Java and. NET. The former, however, still tends to be a lot of wide deployed. So as to line the foundation of our contribution and support the fabric presented within the following sections, we have a tendency to begin by in brief over viewing the Java design with attention on Java a pair of Mobile Edition. Next we have a tendency to gift the principles of the Java Security design, with attention on the mobile edition.

Since the version 2, Java technologies divided into three editions: Enterprise Edition (J2EE), customary Edition (J2SE) and small Edition (J2ME). Every caters for a distinct deployment platform. The high level design and context of the 3 editions. J2EE is intended to support multi-tier enterprise applications, J2SE provides for basic Java applications whereas the J2ME is targeted at resource strained environments like PDAs (Personal Digital Assistant) and mobile phones. At the lowest of each of the editions lies a Virtual Machine runtime environment - JVM for J2EE and J2SE, KVM and Card VM for extremely strained platforms. To support numerous target platforms and their capabilities, J2ME defines the notions of configurations and profiles. A J2ME configuration specifies the options and needs of the Java runtime setting and its genus Apis that correspond to completely different categories of devices. The present J2ME specification defines 2 main configurations: Connected Device

Configuration (CDC) [1] and Connected Limited Device Configuration (CLDC) [1]. The previous targets high-end mobile devices with richer options. CLDC, on the opposite hand, is aimed toward extremely strained client devices. It supports solely a set of a JVM (including genus Apis, libraries etc.), called KVM. The layer higher than CLDC is that the Mobile data Device Profile (MIDP) [2]. J2ME profiles have the role of process API libraries that alter specific type of applications to be developed for the target platform- in accordance with the underlying configuration. In conjunction, MIDP and CLDC represent application execution environment and supply for the connected practicality. The standardized J2ME setting for extremely strained consumer devices consists of MIDP, CLDC and supporting libraries.

Applications running on high of MIDP area unit spoken as MIDlets. The varied files representing MIDlet code (a JAR file), application supporting knowledge and different resources are bundled along in MIDlet suites. Namely, a MIDlet suite is comprised of: (1) one JAR file containing the Java category (MIDlet), the manifest file, and application resources (images, etc.), and (2) Java Application Descriptor file (JAD) that specifies data associated with the applying. The contribution of this paper builds on the principles of the Java security design. To stipulate those principles, we discuss with the generalized security setting as provided in J2SE (Java a pair of customary Edition). This offers North American nation grounds to introduce the strained model of J2ME (Java a pair of small Edition) and clearly position our contribution relative to that.

The fundamental construct within the generalized Java security architecture is that the sandbox. Sandbox represents AN execution environment with strict, policy-based resource access control and robust isolation properties. Code capital punishment within a sandbox (i.e. the sandbox itself) is related to a protection domain that, in turn, determines the permission set granted to the application1. Generally, JVMs enable the definition of protection domains through Java security policy files. The static sets of permissions so such area unit dynamically mapped at runtime by the JVM. The policy file entries specifying permissions are mentioned as grant entries. A policy file for a J2SE run-time surroundings is absolutely outwardly configurable by the platform users and directors. This includes the liberty to outline and specify each permissions and domains in J2SE. Java a pair of security design. The trust model of Java a pair of distinguishes between 2 main categories of applications: trustworthy and untrusted. The former is usually allowed to run unrestricted, the latter (applets and remote code) area unit continually subjected to the safety policy.

Due to the restricted capabilities of the devices running Java a pair of small Edition (J2ME), the corresponding security design has been significantly simplified. Whereas finding the resource consumption issue, such simplification represents a clear trade-off against variety of different aspects of the various security models. Our work targets the adverse effects that this simplification has on flexibility and roughness of policy specification and social control. While code in J2SE runs among the JVM, applications executed in J2ME (on prime of CLDC and MIDP, on an affected mobile device run among KVM - a scaled-down version of JVM. The sandbox model defined by KVM is significantly completely different thereto of JVM: it restricts the exposed API thereto predefined by CLDC; application management happens at native code level; user is prohibited from touching the classloader or downloading any native libraries.

5. Conclusion

This paper has given a sensible extension to the Java virtual machine for mobile devices that supports fine grained security policy and enforces them through run-time monitoring. It addresses the users' want for application control and opens the chance of a replacement generation of mobile services and applications. Although the given model for Run-time Monitor has been enforced for the MIDP profile, the introduced design concepts is applied to different J2ME profiles as well. Since the extended security design is enforced solely at the amount of Java libraries and modules, the modifications done don't have an effect on the KVM nor in operation system. Therefore, the Run-time monitor ought to be easily transportable to various implementations.

6. References

1. Java ME. Java Platform Micro Edition, Sun Microsystems. Available from http://java.sun.com/javame/index.jsp/.
2. JSR 118. Mobile Information Device Profile 2.0, Sun Microsystems. Technical report. Available from http://jcp.org/en/jsr/detail?id=118.
3. Opie. Open Palmtop Integrated Environment Applications and libraries for mobile devices. http://opie.handhelds.org/cgi-bin/moin.cgi/.
4. Phone ME. https://phoneme.dev.java.net/.
5. Qt/Embedded. http://www.trolltech.com/download/qt/embedded.html.