ISSN: 2348-2079

MAY-2015

International Journal of Intellectual Advancements and Research in Engineering Computations

TEXT MINING-BASED SIMILARITY MEASURE FOR COLLABORATIVE POLICY ADMINISTRATION

[1]P. Vidhya Rohini, [2]A. Sudha.

ABSTRACT

Policy-based management is a very efficient method to protect sensitive information. The over maintain of privileges is common in up-and-coming applications, including mobile applications and social network services. Because the applications' users involved in policy administration have little knowledge of policy-based management. The over maintain can be leveraged by malicious applications, then lead to serious privacy leakages and financial loss.

To solve this issue, in this project we propose a novel policy administration mechanism, referred to as Collaborative Policy Administration (CPA), to simplify the policy administration. In CPA, a policy administrator can refer their own polices and also refer other policy to build a new policy for maintain the data security. This project formally defines CPA and proposes its enforcement framework. We also to obtain similar policies more effectively, which are the key step of CPA, a text mining-based similarity measure method are presented.

INTRODUCTION

The next generation of open operating systems won't be on desktops or mainframes but on the small mobile devices we carry every day. The openness of these new environments will lead to new applications and markets and will enable greater integration with existing online services. However, as the importance of the data and services our cell phones support increases, so too do the opportunities for vulnerability. It's essential that this next generation of platforms provides a comprehensive and usable security infrastructure. Developed by the Open Handset

Alliance, Android is a widely anticipated open source operating system for mobile devices that provides a base operating system, an application middleware layer, a Java software development kit (SDK), and a collection of system applications.

Although the Android SDK has been available since late 2007, the first publicly available Android ready "G1" phone debuted in late October 2008. Since then, Android's growth has been phenomenal: T-Mobile's G1 manufacturer HTC estimates shipment volumes of more than 1 million phones by the end of 2008, and industry insiders expect public adoption to increase steeply in 2009. Many other cell phone providers have either promised or plan to support it in the near future. A large community of developers has organized around Android, and many new products applications are now available for it. One of Android's chief selling points is that it lets developers seamlessly extend online services to phones. The most visible example of this feature is, unsurprisingly, the tight integration of Google's Gmail, Calendar, and Contacts Web applications with system utilities.

Android users simply supply a username and password, and their phones automatically synchronize with Google services. Other vendors are rapidly adapting their existing instant messaging, social networks and gaming services to Android, and many enterprises are looking for ways to integrate their own internal operations into it as well. Traditional desktop and server operating systems have struggled to securely integrate such personal and business applications and services on a single platform.

Author for Correspondence:

[1]Final year ME, Al-Ameen Engineering College, Erode, Tamilnadu, India.

[2]Assistant Professor/CSE, Al-Ameen Engineering College, Erode, Tamilnadu, India.

Although doing so on a mobile platform such as Android remains nontrivial, many researchers hope it provides a clean slate devoid of the complications that legacy software an cause. Android doesn't o"cially support applications developed for other platforms: applications execute on top of a Java middleware layer running on an embedded Linux kernel, so developers wishing to port their application to Android must use its custom user interface environment. Additionally, Android restricts application interaction to its special APIs by running each application as its own user identity. Although this controlled interaction has several beneficial security features, our experiences developing Android applications have revealed that designing secure applications isn't always straightforward. Android uses a simple permission label assignment model to restrict access to resources and other applications, but for reasons of necessity and convenience, its designers have added several potentially confusing revetments as the system has evolved.

RELATED WORK

Policy administration is the key to protecting or operating information systems [3]. Only after a legitimate policy set is designed, can the systems run correctly. As a result, many researchers proposed their works on this topic. This paper proposes a policy mechanism CPA under the current trust model, where a professional policy administrator or group is absent.

Recently, the security of the Android platform becomes a hotspot in the field of system security. Enck et al. [6] revealed that two-thirds of the 30 randomly chosen popular third-party applications exhibited suspicious behaviors, such as disclosing sensitive information, and that half of the applications reported users' locations to remote advertising servers. Grace et al.'s analysis tool [2] showed that among the 13 privileged permissions examined, 11 were leaked, with individual phone leaking up to eight permissions, which can be exploited to wipe out the user data, send premium rate messages, and so on. Ongtang et al. proposed a complex policy model, and Nauman et al. [8] proposed runtime security constraint components for the Android platform. Their goal is not to reduce the burden of policy design or policy verification. In fact, these two mechanisms may impose a higher requirement of professional knowledge on the Android security than previous methods. To assist an application user in determining whether he or she should accept the permissions, Nauman et al. proposed to provide additional information in the installation interface. This tool, however, only shows the static description of permission in more detail. Enck et al. proposed a tool named Kirin to analyze the manifest of Android application package files to ensure that the permissions requested by the application satisfy a certain safety policy.

To verify the over claim of privileges among Android applications, Felt et al. [11] proposed a method where they detect API calls, then verify the policy configurations. Their method cannot be effective in defending the threats from attackers who request permission but implement a malicious relevant function. Sarma et al. [7] used the risk and benefit measures to verify the policy configurations. This paper, proposes two functions based on similar policies, collaborative policy design and collaborative policy verification. The proposed CPA and Sarma et al.'s method are different in mechanisms.

Policy recommendation was proposed to simplify the policy administration in social network services. A user, as a verifier, can view the verification result of permission requests based on the similar policy sets. This paper proposes a text mining-based method, thus can provide a more independent method to obtain similar policies, and then improve the effectiveness of the functions of collaborative policy design and collaborative policy verification. Furthermore, CPA proposed in this paper also supports both collaborative policy design and collaborative policy verification, while the policy recommendation proposed by Shehab and Marouf [12] is only used in policy verification.

POLICY BASED SECURITY SCHEMES

Policy appliances are technical control and logging mechanisms to enforce or reconcile policy rules and to ensure accountability in information systems. Policy appliances can be used to enforce policy or other systems constraints

within and among trusted systems. The emerging global information society consists of many heterogeneous but interconnected systems that are governed or managed according to different policies, rules, or principles that meet local information management needs. For example, systems may be subject to different international, national or other political subdivision information disclosure or privacy laws; or different information management or security policies among or between government agencies, government and private sector information systems, or producers and consumers of proprietary information or intellectual property, etc.

Policy appliances -- a generic term referring to any form of middleware that manages policy rules -- can mediate between data owners or producers, data aggregators, and data users, and among heterogeneous institutional systems or networks, to enforce, reconcile, and monitor agreed information management policies and laws across system with divergent information policies or needs. Policy appliances can interact with smart data intelligent agents or context-aware applications to control information flows, protect security and confidentiality, and maintain privacy.

Policy appliances support policy-based information management processes by enabling rules-based processing, selective disclosure, and accountability and oversight. Examples of policy appliance technologies for rules-based processing include analytic filters, contextual search, semantic programs, labeling and wrapper tools, and DRM, among others; policy appliance technologies for selective disclosure include anonymization, content personalization, subscription and publishing tools, among others; and, policy appliance technologies for accountability and oversight include authentication, authorization, immutable and non-repudiable logging, and audit tools, among others.

Control and accountability over policy appliances between competing systems is becoming a key determinant in policy implementation and enforcement, and will continue to be subject to ongoing international and national political, corporate and bureaucratic struggle. Transparency together with immutable and non-repudiable logs, are necessary to ensure accountability and compliance for both political, operational and civil liberties policy needs. Increasingly, international and national information policy and law will need to rely on technical means of enforcement and accountability through policy appliances.

POLICY MANAGEMENT IN DISTRIBUTED COMPUTING ENVIRONMENT

Distributed computing is a field of computer science that studies distributed systems. A distributed system is a software system in which components located on networked computers communicate and coordinate their actions by passing messages. The components interact with each other in order to achieve a common goal. Three significant characteristics of distributed systems are: concurrency of components, lack of a global clock, and independent failure of components. Examples of distributed systems vary from SOA-based systems to massively multiplayer online games to peer-to-peer applications.

A computer program that runs in a distributed system is called a distributed program, and distributed programming is the process of writing such programs. There are many alternatives for the message passing mechanism, including RPC-like connectors and message queues. An important goal and challenge of distributed systems is location transparency. Distributed computing also refers to the use of distributed systems to solve computational problems. In distributed computing, a problem is divided into many tasks, each of which is solved by one or more computers, which communicate with each other by message passing.

PDP is a component of policy-based management. When a user tries to access a file or other resource on a computer network or server that uses policy-based access management, the Policy Enforcement Point will describe the user's attributes to other entities on the system. The PEP will give the PDP the job of deciding whether or not to authorize the user based on the description of the user's attributes. Applicable policies are stored on the system and are analyzed by the PDP. The PDP makes its decision and returns the decision. The PEP will let the user know whether or not he has been authorized to access the requested resource. Policy Enforcement

Point", is the logical entity or place on a server that enforces policies for admission control and policy decisions in response to a request from a user wanting to access a resource on a computer or network server.

PEP is a component of policy-based management. When a user tries to access a file or other resource on a computer network or server that uses policy-based access management, the PEP will describe the user's attributes to other entities on the system. The PEP will give the Policy Decision Point the job of deciding whether or not to authorize the user based on the description of the user's attributes. Applicable policies are stored on the system and are analyzed by the PDP. The PDP makes its decision and returns the decision. The PEP will let the user know whether or not he has been authorized to access the requested resource. Policy Administration Point (PAP) provides centralized administration, management and monitoring of entitlement policies, and delegation and integration with enterprise information repositories such as Active Directory and Lightweight Directory Access Protocol (LDAP). The features of PAP include:

- Browser-based, drag-and-drop interface for creation of granular entitlement policies based on user, resource, request context, action, and other environmental attributes
- Ability to set per-application as well as enterprise-wide policies
- Administer entitlements including ability to group users and resources and clone and inherit entitlements.
- Delegate administration for operational scalability by empowering the appropriate people to manage policies for their applications and lines of business (LOBs)
- Centralized management, review, and audit of all entitlement policies for all applications
- "What if" functionality that allows administrators to understand implications of policy changes in advance of deploying them in a production environment

COLLABORATIVE POLICY ADMINISTRATION FOR MOBILE APPLICATIONS

The method of policy-based management is widely used to manage complex and large-scale network systems [3]. The traditional framework of policy-based management consists of four core components: policy decision point (PDP), policy enforcement point (PEP), policy administration point (PAP), and policy repository (PR). A well-trained policy administrator or group will specify, verify policies in PAP and deploy the policies in PR. After a system runs, PDP will retrieve applicable policies from PR and make decisions. PEP takes charge of the decision, such as satisfying the request where a subject wants to open a file, or launching a logger to record system context.

The overclaim of privileges, where a not well-trained administrator assigns more privileges than those which are normally required by a subject, is an increasingly serious problem, especially when the method of policy-based management is applied to emerging application scenarios, such as mobile applications [9] and social network services [10]. For instance, during the process of Android application development, three roles are usually involved in the policy administration: Application Developers declare which permissions the application will request; Application Marketers verify whether the application is legitimate by an automatic tool; Application Users decide whether to approve the permission requests. These three roles are usually performed by those who are not well trained in policy-based management. That is, the developers usually declare more permissions than necessary because they are inclined to make the development of applications easier, or even misunderstand technical documents [11]; the marketers usually tend to allow more applications regardless of the malicious permission requests; and the application users may not know what the requested permissions mean, thus approving all requests because they are eager to use the application. The same issue exists in social network services, where a user is asked to grant access to private data to third-party applications [12].

This challenge to policy administration is increasingly serious due to the explosion of these

applications. Among all smartphones shipped during the second quarter of 2012, Android OS smartphones had the largest global market share [1]. Furthermore, social network services have become one of the most popular web applications in the world [10]. For example, in April 2012, Twitter, an online microblogging service created in March 2006, announced that it commanded more than 140 million active users and saw 340 million Tweets a day]. Although IETF proposed the protocol of OAuth 2.0, the policy administration is still a serious problem in the policy-based management of these emerging applications [12]. As a result, we should strengthen the policy administration mechanism in these application scenarios.

This paper proposes collaborative policy administration (CPA). The essential idea of CPA is that applications with similar functionalities shall have similar policies that will be specified and deployed. Thus, to specify or verify policies, CPA will examine policies already specified by other similar applications and perform collaborative recommendation. The degree of similarity will be calculated by predefined algorithms, which could be a category-based algorithm, a text mining-based algorithm, and so on. The main contributions of this paper are as follows:

- We propose a novel method—collaborative policy administration, to help not well-trained users, even novices to specify and verify policies. We define the formal model of CPA. In this model, two main functions in policy administration are defined based on similarity measure methods, which will select similar policies as a refinement basis to assist administrators to design or verify their target policies.
- We propose a text mining-based similarity measure method to help policy administrators to obtain similar policies. According to the evaluation, the proposed method is more effective than the category- based similarity measure method, which is more widely used in other literatures.
- We present an enforcement framework and implement a prototype of CPA. The framework supports two types of user interfaces and provides functions of collaborative policy design and collaborative policy verification.

PROBLEM STATEMENT

The traditional framework of policy-based management consists of four core components policy decision point (PDP), policy enforcement point (PEP), policy administration point (PAP), and policy repository (PR). A well-trained policy administrator or group will specify, verify policies in PAP, and deploy the policies in PR. After a system runs, PDP will retrieve applicable policies from PR and make decisions. PEP takes charge of the decision, such as satisfying the request where a subject wants to open a file, or launching a logger to record system context. The overclaim of privileges, where a not well-trained administrator assigns more privileges than those which are normally required by a subject, is an increasingly serious problem, especially when the method of policy-based management is applied to emerging application scenarios, such as mobile applications and social network services. The following problems are identified by the existing systems

- Application users may not know what the requested permissions mean, thus approving all requests because they are eager to use the application.
- User will approve all requests from third-party applications, because User wants to run the applications, thus falling into the traps of malicious applications.
- The User leakage of their privacy.

TEXT MINING BASED SIMILARITY MEASURE FOR COLLABORATIVE POLICY ADMINISTRATION

This paper proposes collaborative policy administration (CPA). The essential idea of CPA is that applications with similar functionalities shall have similar policies that will be specified and deployed. Thus, to specify or verify policies, CPA will examine policies already specified by other similar applications and perform collaborative recommendation. The degree of similarity will be calculated by predefined algorithms, which could be a category-based algorithm, a text mining-based algorithm, novel method, enforcement framework

and implement a prototype of CPA. The framework supports two types of user interfaces and provides functions of collaborative policy design and collaborative policy verification. This project contains the following modules Collaborative policy design, Collaborative policy verification and Enforcement framework

COLLABORATIVE POLICY DESIGN

Admin refers to all involved policy administrators, including, e.g., developers, marketers, and end users in the Android framework. Policy administrator Admin can obtain a refined policy set PS ref according to a refinement function. We design the policy using the system such as a new user can register and logins and upload any file. The user can design the policy in it. That is the policy may be like download option available or not, client details view options such that options.

COLLABORATIVE POLICY VERIFICATION

A policy administrator Admins can obtain a verification result. VeriResult for a target policy set PS target, which contains all polices assigned to a target subject SUBJS, according to a verification function.

ENFORCEMENT FRAMEWORK

A policy administrator can leverage the framework to administrate policies via a phone, web browser, or development tool. The direction of arrows is the direction of key data flows. The history policy base and similarity measure methods are two key components in the enforcement framework. To enforce CPA, the administrator should prepare a sufficient number of policies at first. Furthermore, collaborative policy design and collaborative policy verification are the two key functions provided by the framework. These two functions depend on the history policy base and similarity measure methods. After obtaining the similar policies, the two functions call a refinement algorithm and a verification algorithm respectively. Finally, collaborative policy design and collaborative policy verification will display the results to the administrator on various user interfaces, e.g., a phone, web browser, or development tool.

CONCLUSION

The mobile application security mechanism is designed with the text mining techniques. Policy analysis operations are carried out using the text mining concepts. Collaborative policy verification helps the end users to identify malicious permission requests. The system supports securer and more acceptable applications for end users. The system improves the mobile application security and access permission management process.

REFERENCES

[1]. IDC, "Android and iOS Surge to New Smartphone OS Record in Second Quarter, According to Idc," 2012.

[2]. M. Grace, Y. Zhou, Z. Wang, and X. Jiang, "Systematic Detection of Capability Leaks in Stock and Roid Smartphones," Proc. 19th Ann. Symp. Network and Distributed System Security, 2012.

[3]. W. Han and C. Lei, "A Survey on Policy Languages in Network and Security Management," Computer Networks, vol. 56, no. 1, pp. 477-489, 2012.

[4]. Twitter, "Twitter Turns Six," twitter-turns six.html, 2012.

[5]. OAuth, "The OAuth 2.0 Protocol," 2011.

[6]. W. Enck and A. Sheth, "Taintdroid: An Information-Flow Tracking System for Realtime Privacy Monitoring on Smartphones," Proc. Ninth USENIX Conf. Operating Systems Design and Implementation, pp. 1-6, 2010.

[7]. B. Sarma, N. Li, C. Gates, R. Potharaju, C. Nita-Rotaru, and I. Molloy, "Android Permissions: A Perspective Combining Risks and Benefits," Proc. 17th ACM Symp. Access Control Models and Technologies, 2012.

[8]. M. Nauman, and Zhang, "Apex: Extending Android Permission Model and Enforcement with User-Defined Runtime Constraints," Proc. Fifth ACM Symp. Information, Computer and Comm. Security, pp. 328-332, 2010.

[9]. D. Barrera and Somayaji, "A Methodology for Empirical Analysis of Permission-Based Security Models and Its Application to Android," Proc. 17th ACM Conf. Computer and Comm. Security, pp. 73-84, 2010.

[10]. G. Cheek and M. Shehab, "Policy-by-Example for Online Social Networks," Proc. 17th ACM Symp. Access Control Models and Technologies, pp. 23-32, 2012.

[11]. A. Felt, E. Chin, S. Hanna, D. Song, and D. Wagner, "Android Permissions Demystified," Proc. 18th ACM Conf. Computer and Comm. Security, 2011.

[12]. M. Shehab and S. Marouf, "Recommendation Models for Open Authorization," IEEE Trans. Dependable and Secure Computing, July/Aug. 2012.

[13]. Weili Han, Gang Pan and Zhaohui Wu, "Collaborative Policy Administration", IEEE Transactions On Parallel And Distributed Systems, February 2014.