Trust Infrastructure for Policy based Messaging In Open Environments

Gansen Zhao
The Computing Laboratory
University of Kent
Canterbury, CT2 7NF
UK
gz7@kent.ac.uk

David Chadwick
The Computing Laboratory
University of Kent
Canterbury, CT2 7NF
UK
d.w.chadwick@kent.ac.uk

Abstract

Policy-based messaging (PBM) aims at carrying security policies with messages, which will be enforced at recipient systems to provide security features. PBM promotes a distributed mechanism for secure messaging. The openness of computing environments challenges the PBM model due to the varying trust relations between the different systems and their different behaviours. This paper present a design of a trust infrastructure which is developed based on a Public Key Infrastructure. The trust infrastructure publishes policy enforcement information about the messaging systems, and engenders trust through consistent and mandatory policy enforcement by the systems. It incorporates policy-based management mechanisms to provide flexible and customised messaging services. Secure messaging is achieved by defining security related policies and confining messaging systems' behaviours to defined security constraints. The process of PBM is also described, including publishing certificates, sending messages, accessing messages, and enforcing policies.

1. Introduction

Messaging systems have penetrated into most of the areas of human society during the last decade or two, including the military, using systems like the Military Message System [12], commerce using products like Microsoft Exchange and Outlook, and the public, using web based email like Hotmail and Yahoo. Most nonmilitary messaging systems enable person to person communications and are based on Internet standards such as IMAP [3], MIME [6], SMTP [11], and POP3 [16]. They are among the most widely used Internet applications today.

Current trends in Internet applications are shifting from best effort, vertical network architectures towards a more flexible, open, dynamic and service aware network paradigm. Applications are being provided with distributed and customised service support. From a management point of view, messages are resources that are transferred over the network, and are critical when they contain commercially or nationally sensitive information. Information owners may wish to define rules and constraints over the use of these resources, which can serve to protect the resources, or to facilitate proper access to these resources. The main purpose of this paper is to augment messaging systems with policy based management enabled infrastructures so as to provide more flexible and secure messaging services.

This paper is organised as follows. Section 2 presents a scenario of PBM, so as to aid the readers' better understanding of the research, and is followed by the challenges that the senario presents. Section 3 formalises PBM systems. In Section 4, a trust infrastructure for policy enforcement is developed. Section 5 describes the process of PBM including the publication of policy enforcement abilities, sending messaging, accessing messages, and enforcing policies. Section 6 provides a brief resume of related work. Section 7 concludes the paper with an evaluation of the proposed mechanism and a summary of the future research that is still needed.

2. Scenario and Challenges

The main purpose of this paper is to enhance messaging with a policy based management infrastructure so as to provide more flexible and secure messaging services in open environments. The objective can be better understood by considering the following scenario.

Prior to communication, message senders compose messages and assign recipients to the messages. Then senders consider what constraints and rules may need to be applied to accessing the messages. All the constraints and rules are then defined by a policy. The policy is then attached to the message and sent with the message. The policy will be delivered, stored and protected as an integral part of the message.

To access a message, the target messaging system will retrieve the enclosed policy before allowing the user to access the contents of the message. The target system can only perform action on the message that is in accordance with the policy. Requests of action in violation of the policy will be refused. All messages are protected in the way defined by the associated policies which are enforced by the target system.

From the perspective of security and trust, the challenges of PBM systems are: how to capture the message originator's intentions and represent these accurately in the message policy, how to ensure the target messaging systems will understand the policies and implement them as expected, and how to ensure the target systems will enforce the policies as expected and protect the messages from disallowed actions being performed on them. These issues are the main ones that the work reported here aims to address. Implications of these issues are listed below.

1. Semantic Consistency. Policies must be understood by the target systems in the way that the sender envisages. Consistent semantic awareness of policies by different systems provides the basis for enforcing policies with identical expected effects.

2. Trust in Autonomous systems. Messaging systems are built on open environments. The openness and dynamics of the environment eliminate any possibility of a participant retrieving all knowledge about the environment in advance of any actions. System administrators only have control within their own administrative domains. Thus senders need a way to be guaranteed, or as a minimum to trust, that a remote system is able to and will enforce the attached policy.

3. Policy Enforcement

Policy enforcement models define policy systems. A policy system contains an underlying policy language which represents the constraints and rules that need to be expressed, and the effect of enforcing policies which are specified using the underlying policy language. A policy enforcement implementation is a system instance which can enforce policies as defined by a policy enforcement model. A generic model for policy enforcement [22] comprises of two main components, a Policy Decision Point (PDP) and a Policy Enforcement Point (PEP). The PDP makes decisions about issues which are of concern to policies. The PEP enforces the decisions made by the PDP to regulate the system's behaviour.

3.1. Policy Enforcement Model

A policy augmented message is denoted by $< m, p >$, where m denotes a normal message, which contains the information that needs to be delivered to the recipient, and p denotes a policy, which is attached to the message m and conveys rules and constraints.

The assumption of PBM is that the attached policy p will be enforced at the target system on accessing the message m. Thus all accesses to the message will be governed and regulated by the attached policy p.

A policy enforcement model defines the semantics of policies and the effects of policy enforcement. A policy enforcement model defines a function

$$\mathcal{E} :< P, S > \rightarrow boolean$$

where P is the policy domain that the attached policy originates from, and S is the situation domain of the system that is going to enforce the policy. S contains all the information related to the system and its environment that is necessary for the enforcement model $\mathcal{E}$ to compute its output which is the decision of the policy enforcement. $\mathcal{E}(p, s) = true$ if and only if the situation s does not violate the policy p.

Policy enforcement models are interpretation of policies, and relate policy domains to application domains to provide application specific semantics. Policy enforcement models decide whether a given situation violates a policy or not, thus applications can enforce policies based on the decisions output by policy enforcement models.

3.2. Policy Enforcement Implementations

A system K is said to have implemented the policy enforcement model $\mathcal{E}$ if, given any policy $p \in P$ and any situation $s \in S$, K accepts a situation s when enforcing the policy of p if and only if $\mathcal{E}(p, s) = true$.

If a system K has implemented a policy enforcement model $\mathcal{E}$, the system K is said to be a policy enforcement implementation of $\mathcal{E}$, denoted by $K_{\mathcal{E}}$.

The above definition of the policy enforcement implementation specifies that a policy enforcement implementation $K_{\mathcal{E}}$ accepts only those situations that are accepted by the policy enforcement model $\mathcal{E}$.

The implementation definition does not specify how to enforce the policy. In fact, there are several kinds of policy enforcement mechanism that can be used to enforce policy. The most common policy enforcement mechanism is Execution Monitoring (EM) [21]. Execution Monitoring monitors each execution of the system, and terminates those executions that are not accepted by the policy enforcement model.

4. Trust Infrastructure for PBM

In an open environment, it is hard to identify whether the recipient system can and will enforce policies in the expected way. Single implementation systems, usually proprietary, do not suffer from this problem, since the presence of the same implementation throughout ensures the decidability of the enforcement mechanism. However, in an open messaging environment, a single implementation is unrealistic. Nevertheless, it would be relatively easy to tell whether a system is believed to enforce policies in the expected way if the system were certified as such by a trusted third party. In this section, an infrastructure for PBM is developed based on trusted assertions/certificates. These trusted assertions will publish the relationships between policy enforcement models and enforcement implementations (concrete software systems).

4.1. Trusted Enforcement Implementations

Given a policy enforcement model $\mathcal{E}$ and a policy enforcement implementation K, K is said to be a trusted enforcement of $\mathcal{E}$ if and only if K is believed by a trusted authority to be able to and will enforce policies in a way that conforms to the specific policy enforcement model $\mathcal{E}$. That K is said to be a trusted enforcement of $\mathcal{E}$ is denoted by $\mathcal{E} \models K$.

With the trusted enforcement relation, $\mathcal{E} \models K$, it can be believed or assumed that K will accept a situation s if and only if $\mathcal{E}(p, s) = true$.

Thus, the trusted enforcement expresses two beliefs. Fristly, the target system, K, can understand the specified policy p in the same way as defined by the policy enforcement model $\mathcal{E}$. This will ensure that the enforcement of the policy will be consistent with the semantics defined by the policy enforcement model $\mathcal{E}$. Secondly, the target system, K, will imperatively enforce the specified policy. The sender of the message, who composes the message with a policy, will know that the attached policy will have its effects to govern access to the message, regulating the system's behaviour in regard to the use of the information.

4.2. An Architecture for PBM

Six kinds of parties are involved in the architecture for PBM systems: message senders, message recipients, client messaging systems (comprising message user agents and message transfer agents), target messaging systems (comprising message stores and message user agents), Source of Policy Authorities (SoPAs), and Policy Implementation Authorities (PIAs).

Message senders are actors who create and send policy based messages using client messaging systems. Senders compose messages and related policies which are to be delivered to recipients together. Senders trust selected trusted third parties, known as Policy Implementation Authorities, to assert that the target messaging systems will enforce their policy.

SoPAs are authorities that publish policy enforcement models. The description of a model will include its policy language, and its semantics. An example of policy enforcement models is XACML [19].

Client messaging systems support one or more policy enforcement models, and provide the user with an interface to compose messages, and an intuitive way of setting a policy according to the policy enforcement model chosen by the user.

PIAs are authorities that certify that a particular implementation conforms to a published policy enforcement model. PIAs may be delegated by one or more SoPAs to certify implementations on their behalf, or they may act independently. PIAs publish signed assertions to state their belief that certain messaging systems have implemented a particular policy enforcement model and will mandatorily enforce policies in a way that conforms to this model.

Target messaging systems are systems that receive and access messages and enforce the policies that are associated with the messages. Target systems implement one or more policy enforcement models and require PIAs to publish assertions about their policy enforcement implementations.

Message recipients are the final destinations of senders' messages, but they can only access the messages via a target messaging system which enforces the senders' policy. In order to show that a user's email address is linked to a particular target messaging system, the latter provides the user with an assertion of this fact.

Signed assertions can be encoded in many different formats and syntaxes. Two common ones are SAML assertions [18] and X.509 attribute certificates [9]. We are choosing to use the latter because of their compact size and performance [15].

Three kinds of Attribute Certificates are issued for PBM: Policy Enforcement Model Certificates (PEMCs), Policy Enforcement Implementation Certificates (PEICs) and End User Policy Certificates (EUPCs). A PEMC contains information about the issuer of the PEMC and a detailed specification of the policy enforcement model. PEMCs are universally located and identified by their URLs.

PEICs are issued to clients and target messaging systems to assert that they conform to particular policy enforcement model(s). They are issued by PIAs. A PEIC holds a universal reference to the policy enforcement model that is implemented by the target system.

EUPCs are issued by target messaging systems to the (users of the) email addresses that they service. The holder field contains the email address of the end user. EUPCs bind email addresses to the messaging systems that the users will use. Messages sent to the address specified in an EUPC is believed to be delivered to the specified messaging system.

4.3. Trust Model Summary

The trust model above can be summarised as follows. The Message Sender trusts (and uses) a policy enforcement model from a trusted SoPA. The Message Sender also trusts one or more PIAs to certify that various messaging systems conform to his trusted policy enforcement model. The trusted PIAs certify target messaging systems as conforming to one or more policy enforcement models. Each target messaging system certifies the email addresses in its domain as being ”controlled” by it and issues EUPCs to these email addresses. The Message Sender can therefore trust that any message sent to a recipient email address with a valid EUPC will have his policy enforced by the target messaging system that ”controls” that email address.

5. Policy-based Messaging

The whole process of PBM involves publishing policy enforcement abilities, sending messages, relaying messages, accessing messages, and enforcing policies.

5.1. Publishing Certificates

PEMCs, PEICs and EUPCs are issued (directly or indirectly) by trusted authorities. Publication and distribution of these certificates can be achieved by different parties in different ways. They can be stored and published in LDAP directories [7], on web pages, or by forwarding to the message recipient at connection time. In the case of EUPCs message recipients can send them and their parent PEIC to the message senders prior to PBM commencing. It would also be possible for messaging systems to store their PEICs in the DNS, through the definition of an appropriate new resource record (RR). This might be a good solution in the case of direct messaging such as non-relayed messaging, since messaging clients could retrieve the remote systems PEIC at the same time as its IP address, and then automate the trust chain checking at this time.

5.2. Sending Messages

On sending messages, the message user agent will first need to have access to the recipient's EUPC and the target system's PEICs, through which the sender can find out the set of policy enforcement models that the target system supports. The retrieval process depends on the mechanism that EUPCs and PEICs are published.

With the enforcement capability set of the target system, the sender can make a choice of which policy enforcement model to use in the message. The decision will take into account the capability of each policy enforcement model, and the capability of the sending system to compose policies specified by the policy enforcement model.

After choosing a policy enforcement model, the sender needs to compose the policy as defined by the corresponding PEMC, and then attach the policy to the message before sending out the message.

5.3. Relaying Messages

Messaging system components typically relay a message from one component to the next before the message is finally received by the message recipient. For example a message sender might composes a message using a message user agent (MUA), submit this to the local message submission agent (MSA), which forwards this to a message transfer agent (MTA) at the target site, which deposits this in its local POP3 message store, from where the message recipient finally retrieves the message using his own MUA. At each of these stages, before the sending messaging system component passes the policy based message to the next messaging system component, it must check that the receiving component has a valid PEIC issued by a PIA trusted by the message sender. This ensures that the policy based message never leaves the control of trusted components throughout the entire path from the message sender to the message recipient. If a messaging system component cannot find a certified component to pass the message to, then the message must be returned to the sender along with a non-delivery report.

5.4. Accessing Messages

When a user accesses a message augmented with a policy, the target messaging system will first extract the policy from the message. This will provide the system with the policy definition, which defines the rules and constraints the system must obey for this message, and identification of the enforcement model to be used. By referring to the PEICs it holds, it can confirm that it supports the requested enforcement model. (Note that this should always be the case otherwise the client system would not have sent the message to this email address.)

The identified policy enforcement implementation will be loaded and the extracted policy will be passed to the PDP instance that will made the authorisation decision. To enforce the policy, the PEP must intercept all user accesses to the message and call the PDP and find out if this access is allowed or not. The PEP must then act on the decision result.

6. Related Work

PBM related work includes proof-carrying code, secure messaging, policy based management, and other related researches.

Proof-carrying code [17] augments codes with proofs which provide necessary information for host systems to decide whether it is safe to execute the code. Proof-carrying code provides a way to protect host systems against malicious codes. Microsoft Authenticode is one example of proof carrying code, in which the code is digitally signed by Microsoft and the signature and public key certificate of the signer accompanies the code. The host system can then validate the signature, which proves that the code has not been tampered with since its creation, and the certificate tells the host who the signer was.

Chadwick et al [1] and Mont et al [14] provide secure messaging from the perspective of roles. Roles are employed as the basic unit of authorisation. Users are authorised to perform operations on messages on behalf of roles if they hold the corresponding roles and the roles are allowed to perform the requested operations. Chadwick et al use an XML based policy determines what the roles can do. Jens [8] designs a system to provide a secure mailing list service, which provides secure messaging for group communication. The architecture design prevents information disclosure by separating the message processing into several different several independent components. Wolthusen [23] imposes mandatory security processing for all messages at operating system level.

Policy based management (PBM) is an administrative approach to establish rules and constraints for systems to deal with different situations. Polices are distributed to end systems, and instruct end systems to behave accordingly to achieve the defined management goals.

Ponder [4] is a policy language that provides a way to specify security policies that map onto various access control implementation mechanisms. Key concepts of Ponder are roles and relationships. Roles are used to group policies relating to a position in an organisation. Relationships model interactions between roles and management structures to define a configuration of roles and relationships pertaining to an organisational unit such as a department.

The Rights Management System (RMS) [[13] enables enterprises to add security information to files produced using Microsoft Office 2003 applications. The added policy allows an author to define constraints for the circulation and operations of a document. RMS will enforce policies contained in a document, and governs the circulation and operations of the document assuming all accesses to the document are based on the proprietary systems from Microsoft.

Recent efforts also focus on incorporating messaging services with policy based mechanisms, including MailRecall^{TM} [5], the Omniva Policy Manager package [20], policy-based digital prescription delivery [2], and the policy driven approach to email services [10]. MailRecall^{TM} and the Omniva Policy Manager package provide plug-ins for several popular email clients with the ability of keeping e-mails private and governing accesses to messages according to defined security policies. Kaushik et al [10] design a three-tier approach to enhance e-mail services by using local policies from the network interaction's point of view. Each level imposes a policy to govern the behaviour of different entities. Policies are defined and enforced locally instead of being sent to remote systems.

7. Conclusions and Future Work

PBM augments messaging with policy management, promoting a distributed mechanism to secure message based communication and providing a customised messaging service according to different requirements. In order to be able to trust that a remote system will act according to a specified policy, a trust infrastructure is proposed based on trusted assertions that publishes the relationships between policy enforcement models and enforcement implementations.

The main advantages of the proposed mechanism are threefold. Firstly, it separates specifications and implementations. SoPAs specify the policy enforcement models, PEMCs are used to publish policy Enforcement Model Information, while PEICs certify the implementations of concrete systems that implement the model.

Secondly, it improves the interoperability between autonomous systems in open environments. PEICs and PEMCs provide a way to publish interaction information across systems, including policy representation and policy enforcement semantics. Interaction between systems can vary and evolve if new PEICs and PEMCs are issued, thus the proposed mechanism promotes a flexible way of interaction according to the participant systems' capabilities.

Thirdly, the proposed mechanism enables PBM services in open environments. The trust infrastructure removes the requirement of the assumption held by many systems that systems have the identical implementation of the same model. Users can specify the expected messaging processing by recipient systems with PBM. The customised messaging service caters for different security requirements in different scenarios.

The presented design of role based messaging is not without its limitations. The target messaging system can stop a deviant recipient from performing many of the actions forbidden in the policy e.g. copying or forwarding a message to another recipient, but it cannot prevent all recipient abuses, e.g. print screen, since it does not have control of the user's interaction with the operating system. These types of abuses will need to wait until trusted computing platforms become available, and applications can place controls on the operating system.

The trust infrastructure is currently only specified at a high level. Once implementation commences, more detailed specification work will be needed such as trusted target system identification, how to relaying trusted PIAs and related work. Further, the trust infrastructure relies on the deployment of a large scale PKI infrastructure, which raises the issues of establishing and maintaining policy authorities in an open environment, the issue of accessing certificates issued by multiple authorities in open environments, and the revocation of certificates, and so on. Finally, the sender's policy may request actions that may be in conflict with the recipient's system's security policy. The approach to resolving or reconciling conflicts between message policies and local security policies requires further study and investigation.

Work in the near future will mainly focus on the syntax for the augmentation of messages with policies, the implementation of the proposed mechanism, the integration of the implementation into existing messaging systems, and other related research.

Acknowledgement

The authors will like to thank Nexor who is supporting this research, and Linying Su and Wensheng Xu who have reviewed this work.

References

[1] D. Chadwick, G. Lunt, and G. Zhao. Secure Role based Messaging. In *Eighth IFIP TC-6 TC-11 Conference on Communications and Multimedia Security (CMS 2004)*, pages 303–316, Windermere, UK, September 2004.

[2] D. W. Chadwick and D. Mundy. Policy Based Electronic Transmission of Prescriptions. In *IEEE 4th International Workshop on Policies for Distributed Systems and Networks*, Lake Como, Italy, June 2003.

[3] M. Crispin. RFC 3501 - Internet Message Access Protocol - Version 4rev1. Request For Comment, Network Working Group, March 2003.

[4] N. Damianou, N. Dulay, E. Lupu, and M. Sloman. The Ponder Policy Specification Language. In *Proceedings of the International Workshop on Policies for Distributed Systems and Networks*, pages 18–38. Springer-Verlag, 2001.

[5] V. DeMarines. MailRecall: Secure E-mail for the Enterprise, May 2004. Authentica, Inc.

[6] N. Freed and N. Borenstein. RFC 2045 - Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies. Request For Comment, Network Working Group, November 1996.

[7] R. Harrison and K. Zeilenga. RFC 3771 - The Lightweight Directory Access Protocol (LDAP) - Intermediate Response Message. Request For Comment, Network Working Group, April 2004.

[8] J. Hasselbach. A Practice-Oriented Approach to Security Enhanced Mailing Lists. In *The Sixth International Conference on Electronic Commerce Research*, Dallas, US, October 2003.

[9] ITU-T. Recommendation X.509, ISO/IEC 9594-8. Information Technology – Open Systems Interconnection – The Directory: Public-key and Attribute Certificate Frameworks, 4th ed., 2000. ITU.

[10] S. Kaushik, P. Ammann, D. Wijesekera, and W. Winsborough. A Policy Driven Approach to Email Services. In *Fifth IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY'04)*, Yorktown Heights, New York, June 2004.

[11] J. Klensin. RFC 2821 - Simple Mail Transfer Protocol. Request For Comment, Network Working Group, April 2001.

[12] C. Landwehr, C. Heitmeyer, and J. McLean. A Security Model for Military Message Systems: Retrospective . New Orleans, Lousiana, December 2001.

[13] Microsoft Corporation. Technical Overview of Windows Rights Management Services for Windows Server 2003, November 2003. Microsoft Corporation.

[14] M. Mont, P. Bramhall, and K. Harrison. A Flexible Role-based Secure Messaging Service: Exploiting IBE Technology for Privacy in Health Care. In *Proceeding of the 14th International Workshop on Database and Expert System Applications*. IEEE, 2003.

[15] D. Mundy and D. Chadwick. An XML alternative for performance and security: ASN.1. *IEEE IT Professional*, 6(1), Jan 2004.

[16] J. Myers. RFC 1939 - Post Office Protocol - Version 3. Request For Comment, Network Working Group, May 1996.

[17] G. C. Necula. Proof-carrying code. In *The 24th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 106–119, Paris, France, January 1997.

[18] OASIS Security Service TC. Security Assertion MarkupLanguage (SAML) 2.0 Specification, November 2004. available on http://www.oasis-open.org/committees/security/ Last visit Dec 6, 2006.

[19] OASIS XAMCL TC. eXtensible Access Control Markup Language (XACML) v1.0, Dec 6, 2004. available on http://www.oasis-open.org/specs/index.php#xacmlv1.0.

[20] Omniva Policy Systems. Omniva Policy Manager Technical White Paper, January 2004. Omniva Policy Systems.

[21] F. B. Schneider. Enforceable security policies. *ACM Transactions on Information System Security*, 3(1):30–50, 2000.

[22] J. Vollbrecht, P. Calhoun, S. Farrell, L. Gommans, G. Gross, B. de Bruijn, C. de Laat, M. Holdrege, and D. Spence. RFC 2904 - AAA Authorization Framework. Request For Comment, Network Working Group, August 2000.

[23] S. D. Wolthusen. A Distributed Multipurpose Mail Guard. In *The 2003 IEEE Workshop on Information Assurance*, West Point, NY, US, June 2003.