Synergy: A Policy-driven, Trust-aware Information Dissemination Framework

Ragib Hasan, Marianne Winslett

rhasan@cs.uiuc.edu, winslett@cs.uiuc.edu
Department of Computer Science,
University of Illinois at Urbana-Champaign*

Abstract. Information dissemination is of vital importance in today's information-centric world. However, controlling the flow of information across multiple security domains is a problem. Most of the current solutions rely on prior knowledge of the users for authorization, which does not scale well. Also, many information sources have dynamic access control policies, which are hard to satisfy under existing schemes. In this paper, we present *Synergy*, a general purpose information sharing framework that uses *trust negotiation* to implement scalable authorization in an open environment. Synergy provides an abstraction for the information sources and consumers to accommodate new trust-aware systems as well as legacy systems. We also present a practical disaster management application that uses this framework.

1 Introduction

Information is the key to today's world. Sharing information across security domains has become necessary, as the need for inter-operability increases. This brings forward inter-domain authorization issues. Traditional authorization techniques are not applicable in the large-scale open nature of the Internet. Another issue is trust: how the resource owners in open systems can trust the clients trying to access those resources. Authorization in such situations needs to be dynamic and content-triggered. It is simply not possible to predict who may need to access the system, and therefore arrange user accounts for those clients. This calls for a policy-based authorization infrastructure that allows negotiated access to resources.

Recently, attribute-based authorization schemes like *trust negotiation* have emerged as a solution to the scalable authorization problem. Trust negotiation uses unforgeable digital credentials that can be automatically verified. Resource owners set up policies regarding resource access, and negotiate with clients to establish trust gradually in a monotonic and bilateral manner. This brings the advantage and scalability of real life into the realm of computing. However, it is difficult to change existing protocols to enable legacy applications to use trust

* This research was sponsored by NSF Award number 0331707 and 0331690 to the RESCUE project

negotiation. To solve this, we present *Synergy*, a policy-driven trust-aware framework that enables negotiated dissemination of information across multiple security domains. Synergy decouples the information producers and consumers by acting as a medium for transfer of information. The information producers and information consumers are abstracted to allow usage of legacy applications. The producers and consumers do not require any knowledge of the authorization or trust establishment details. Information is exchanged in a platform/application-independent manner, allowing the information producers and consumers to agree on any format. Synergy's approach improves on the traditional authorization approach in several aspects: first, it is highly scalable; the servers do not have to store per-client information, or any pre-existing relationship with the clients. Second, authorization and access control are dynamic, and fine-grained. Third, trust is bilateral; both the client and the server need to satisfy each other's policies. And finally, it can be integrated into existing information sharing infrastructures with minimal or no changes to legacy applications.

The rest of this paper is organized as follows: in Sect. 2, we discuss briefly the related work. Sect. 3 presents the overview of the Synergy infrastructure. We highlight VisiRescue, an application prototype information sharing, in Sect. 4. Finally, we conclude and discuss future directions in Sect. 5.

2 Related Work

In this section, we examine the related work on authorization in open systems. In attribute-based authorization, user attributes, rather than identity, are used for authorization. Trust negotiation is an iterative process that can establish trust among strangers through the gradual discovery of credentials [17]. Entities possess attributes, represented by unforgeable digital credentials, such as X.509 attribute certificates, Kerberos tickets, etc. [3]. Several systems have been built using TrustBuilder [17] as the agent for trust negotiation. For example, Traust [12] is a generic authorization service built using TrustBuilder. It uses trust negotiation for providing access to both legacy and newer services. While Synergy shares many of the mechanisms used in Traust, it builds on Traust's authorization services to provide an infrastructure for sharing information through decoupling of the information sources and consumers.

PolicyMaker [5] and Keynote [4] are two early trust management systems that are based on capabilities, but are restricted to a closed system. Trust-X [2] is a peer-to-peer framework. which uses an XML-based policy language, X-TNL. Cassandra [1] is a trust management system that uses Datalog. The RT family of role-based trust languages [13] use an extension of SPKI/SDSI. In [6], a formal framework for policy-based access regulation and information disclosure is presented. Interactive access control strategies for trust negotiation are discussed in [11]. In principle, any of these can be used as the trust agent of Synergy.

Shibboleth [15] is an attribute-based authorization system for quasi-open systems. It uses SAML assertions [10], and requires federations among organizational security domains. While it also uses attributes, it is different from trust

negotiation in several aspects. Trust negotiation does not require a federated structure, but Shibboleth is heavily dependent on the notion of pre-established organizational relationships. Shibboleth also has a very limited access control decision-making capability [7], while trust negotiation can enable fine-grained, dynamic authorization. Synergy takes advantage of trust negotiation and hence does not require the federated relationship that Shibboleth has to establish.

3 System Overview

In this section, we present an overview of the Synergy framework, the resource access protocol, and the interaction between this framework and the trust negotiation agents from TrustBuilder [17]. Fig. 1 shows the components of Synergy.

3.1 Components

Synergy uses a client-server framework consisting of the following components: information producers, servers, clients, consumers, and trust agents. The *information producer* component is an abstraction of the resources, e.g. a wind sensor. The framework does not specify the content or the nature of the resource. The *information servers* act as front-ends for the resources. Each security domain needs to have one or more servers handling its resources. Rather than keeping per-client states, the servers maintain time-stamped access tokens, issued to a client when it satisfies the access policy. The *information clients* interact with the servers. After getting the information, the clients transfer it to the *information consumer*, which is an abstraction of the end-application. We ensure modularity by separating the trust and policy-awareness from the clients and servers. The *trust agents* negotiate trust between a server and a client. The server instructs the client to invoke its trust agent to obtain an authorization token.

3.2 Mode of Operation

The client operates in three main phases: *initiation*, *resource discovery*, and *resource access*. During initiation, it establishes contact with the server, and sends initiation messages. During resource discovery, the client discovers the list of resources served by the server. Finally, in resource access phase, it retrieves the resources from the server. Any resource requiring an establishment of trust triggers a trust negotiation session, using the trust agents. The server's operation consists of two phases: *initiation*, and *resource access control*. During initiation, the server listens for client requests. During resource access control, the server responds to the client requests or commands. If the client's request does not have the authorization token, the token has expired, or the token is incorrect for the type of access requested, the server instructs the client to obtain a token using the trust agents. The *Resource Access Protocol* [8] is used for client-server communication. Each resource is represented by a resource type, a resource name, and an XML definition of the resource contents.

3.3 Implementation

Synergy is implemented using about 1100 lines of commented Java code. We used TrustBuilder as the trust agent. For specification of access policies, we use the IBM Trust Policy Language (TPL) [9] which is currently supported by TrustBuilder [17]. This has the advantage of being monotonic, sufficiently expressive, and simple enough for automated processing [16]. Synergy currently supports digital credentials in the form of X.509 Certificates. More details of the implementation can be found in [8].

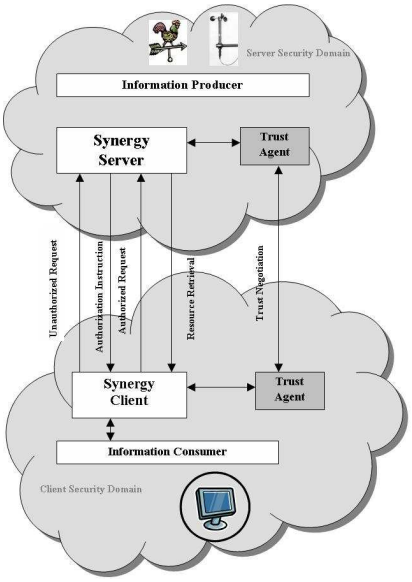

Fig. 1. Components of Synergy

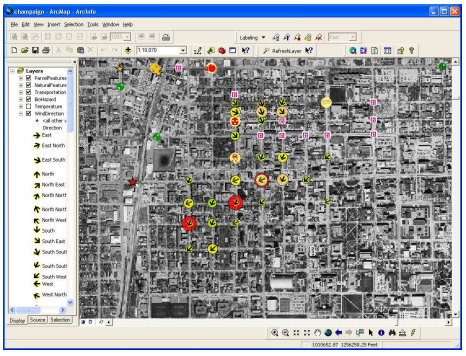

Fig. 2. Screenshot of VisiRescue front-end

4 VisiRescue Application Prototype

In this section, we present an application prototype built on top of Synergy. This prototype, named *VisiRescue*, is a situational awareness system. We built it for possible use in an Emergency Operations Center (EOC), as part of the Responding to Crises and Unexpected Events (RESCUE) initiative[14].

4.1 Overview

Situational awareness is important in disaster management. To get information for, say, a wind storm, sensors and cameras can be deployed around the city. However, outside access to these resources may be restricted due to privacy

concerns. For example, the mall owners are not likely to allow police to monitor the mall's security cameras all the time, except for emergencies. Traditional authorization schemes would require scaling to a large number of heterogeneous organizations and setting up *a priori* relationships with them. Also, schemes like Shibboleth [15], would require coalitions. We argue that the Synergy framework is appropriate here, because each of the resource owners can set access policies to allow attribute-based authorization. Also, Synergy is lightweight, and can be retro-fitted to almost any legacy system.

The information produced by the sensors of VisiRescue map to information producer module of Synergy. A server collecting information from sensors map to the Synergy servers. The aggregator client that runs at the EOC and updates the geo-databases maps to the Synergy client. Finally, ArcGIS display or the Google map interface correspond to the information consumer component. A daemon periodically invokes the Synergy client, which negotiates access to the sensors. The daemon updates the retrieved information in the geo-database used by ArcGIS. We use a loosely defined XML based scheme for resource information. A screenshot of the front-end is shown in Fig. 2. Next, we discuss a usage scenario.

4.2 Usage Scenario

Here, a shopping mall provides access to surveillance cameras via the Synergy servers. For brevity, we present the policies informally here. Sample policies written in TPL can be found in [8]. The mall has the following policy: *Access to video feeds from the surveillance cameras is given only when the requester has the following characteristics: a) The requester is affiliated with either the police department or the fire department, b) The requester is a certified first responder, and c) The fire department provides proof of a fire alarm at the mall.* During an emergency, first responders at the EOC can request a feed from the cameras by clicking the camera icon in the VisiRescue GIS display. The Synergy client will try to access the feed, and may need to perform a trust negotiation session to satisfy the mall's policies. To do so, it will have to provide proof of having the three attributes specified in the policy. On success, the server provides a temporary, limited usage URL for the video feed, which the end-application can use to access the video.

5 Conclusions and Future Work

In this paper, we have presented an information sharing architecture for large scale open systems. The decoupling of the information sources and consumers from the dissemination framework and the authorization mechanism enables Synergy to adapt to different scenarios with the minimal management overhead. The application prototype, VisiRescue, shows the advantage of this approach in practical situations. Our ultimate goal is to have a distributed set of servers and trust agents with increased fault tolerance and robustness. We plan to build a full-fledged prototype application, such as VisiRescue, for possible deployment in

real-life. The robustness of Synergy against different types of attacks also needs to be explored. A detailed security analysis using formal security models may also be done in order to analyze and protect against vulnerabilities.

References

[1] M. Becker and P. Sewell. Cassandra: Distributed access control policies with tunable expressiveness. In *5th IEEE Intl. Workshop on Policies for Distributed Systems and Networks*, 2004.

[2] E. Bertino, E. Ferrari, and A. C. Squicciarini. Trust-X: A peer-to-peer framework for trust establishment. *IEEE Trans. on Knowledge and Data Engineering*, 16(7), 2004.

[3] E. Bina, R. McCool, V. Jones, and M. Winslett. Secure access to data over the internet. In *Proc. of the 3rd Intl. Conf. on Parallel and Distributed Information Systems*, 1994.

[4] M. Blaze, J. Feigenbaum, and A. D. Keromytis. KeyNote: Trust management for public-key infrastructures. *Lecture Notes in Computer Science*, 1550, 1999.

[5] M. Blaze, J. Feigenbaum, and J. Lacey. Decentralized trust management. In *Proc. IEEE Symp. on Security and Privacy*, 1996.

[6] P. Bonatti and P. Samarati. Regulating service access and information release on the web. In *Proc. of the 7th ACM Conf. on Computer and Communications Security*, 2000.

[7] D. Chadwick, S. Otenko, W. Xu, and Z. Wu. Adding distributed trust management to Shibboleth. In *Proc. of the 4th Annual PKI Workshop*. NIST, 2005.

[8] R. Hasan. *Synergy: A Policy-driven, Trust-aware Information Dissemination Framework*. Masters Thesis, Dept. of Computer Science, University of Illinois at Urbana-Champaign, 2005.

[9] A. Herzberg, Y. Mass, J. Michaeli, Y. Ravid, and D. Naor. Access control meets public key infrastructure, or: Assigning roles to strangers. In *Proc. of the IEEE Symp. on Security and Privacy*, 2000.

[10] J. Hughes, E. Maler, H. Lockhart, T. Wisniewki, P. Mishra, and N. Ragouzis. Technical overview of the OASIS security assertion markup language (SAML) v1.1. *OASIS Open*, 2004.

[11] H. Koshutanski and F. Massacci. An interactive trust management and negotiation scheme. In *Proc. of the 1st Intl. Workshop on Formal Aspects in Security and Trust*, 2004.

[12] A. Lee. *Traust: A Trust Negotiation Based Authorization Service for Open Systems*. Masters Thesis, Dept. of Computer Science, University of Illinois at Urbana-Champaign, 2005.

[13] N. Li, J. Mitchell, and W. Winsborough. Design of a role-based trust management framework. In *Proc. of the IEEE Symp. on Security and Privacy*, 2002.

[14] RESCUE Project. The RESCUE Project website. http://www.itr-rescue.org.

[15] T. Scavo, S. Cantor, and N. Dors. Shibboleth architecture technical overview. http://shibboleth.internet2.edu/docs/draft-mace-shibboleth-tech-overview-latest.pdf, 2005.

[16] K. Seamons, M. Winslett, T. Yu, B. Smith, E. Child, J. Jacobson, H. Mills, and L. Yu. Requirements for policy languages for trust negotiation. In *3rd IEEE Intl. Workshop on Policies for Distributed Systems and Networks*, 2002.

[17] M. Winslett, T. Yu, K. E. Seamons, A. Hess, J. Jacobson, R. Jarvis, B. Smith, and L. Yu. Negotiating trust on the web. *IEEE Internet Computing*, 6(6), 2002.