VPAF: a Flexible Framework for Establishing and Monitoring Prolonged Authorization Relationships

Eric Freudenthal and Bivas Das
Department of Computer Science
University of Texas at El Paso
El Paso, Texas 79968
Email: efreudenthal@utep.edu and bdas@miners.utep.edu

***Abstract*—We describe a generic framework for determining and monitoring access rights derived from credential documents. Distributed authorization systems intended to support collaborative coalitions (such as Trust Management systems) typically incorporate mechanisms to both validate credentials, and to determine authorization. This conjunction of distinct functions increases complexity of both components and limits overall flexibility. Furthermore, while authorization decisions frequently enable the commencement of a prolonged relationship, current authorization systems are designed to authorize instantaneous transactions and provide no mechanisms to detect and propagate revocation after an authorization decision is made.**

VPAF (a Validated and Prolonged Authorization Framework) will separate these duties in a manner that permits credential validation and authorization decisions to be managed separately. VPAF is intended to enable vigilant monitoring of prolonged authorization relationships that span mutually distrustful administrative domains such as is common when multiple organizations collaborate.

Keywords - authorization, validation, revocation, delegation, trust management.

I. Introduction

While instantaneous authorization of transactions (such as at time-of-sale) remains an important application of authorization, currently available authorization systems are ill suited for applications where continuous monitoring of prolonged authorization relationships is needed.

For example, consider a host 'H' participating in a distributed hash table 'D' containing some finger table entry 'E' referencing to some other host 'O'. In order to lookup keys in logarithmic time, H will cache O in anticipation that it will be used in future queries [1].

Observe that, H's ability to properly execute operations in D that include references to E depend upon O's providing appropriate responses when queried by H. Thus, it is of value to H that O be trustable.

Sources of H's trust in O may vary by application. For example, as in [2]–[6], trust in O may be delegated through a network of transitive delegation credentials that span multiple administrative domains, or alternatively, may be derived as an aggregation from peers' recommendations. In either case, evidence must be collected, validated, and evaluated, to determine if there is sufficient evidence that E is suitable for inclusion in H's finger table. Furthermore, H's trust in (and therefore reliance upon) O may increase if additional evidence supporting that trust is discovered. Furthermore, H's reliance upon O should decrease the evidence obtained that reduces H's trust in O.

While various systems have integrated mechanisms to monitor and propagate evidence of distrust or otherwise unsuitable characteristics within systems with dynamic behavior, the marriages have corresponded to idiosyncratic characteristics of particular implementations that do not have universal applicability.

Trust management and validation systems are typically integrated in an ad-hoc manner that impedes the inclusion of alternative components. For example, the various trust management systems presently available each provide a different model of trust management paired with a particular mechanism for credential validation and/or discovery. This integrated architecture imposes a high implementation cost for investigators and implementers considering alternative abstractions and pairings.

This project was initially motivated by our development of a scalable credential dissemination validation framework called Fern [7], [8]. Our evaluation of Fern's utility for trust management was impeded by the difficulty of composing it with an existing trust management system. Rather than investing effort in a one-time composition of Fern with a trust management system, we now are investigating the design of this more flexible framework.

By providing a flexible authorization, validation, and discovery substrate, we anticpate that VPAF will provide a unifying model for trust management in coalition contexts.

The remainder of this paper is organized as follows. The next section describes related work in authorization, certificate validation and discovery mechanisms. This is followed by a description of the planned VPAF framework that includes an examination of a model configuration suitable for providing sustained authorization that spans multiple collaborating administrative domains. We conclude with a summary of our investigation details.

II. Related Work

A. Decentralized access control

Trust-management systems such as [2], [3] can be used to provide access control in a coalition context, however they provide no mechanism to monitor if credentials are revoked *after an authorization decision has been made*. Furthermore, these

Digital Object Identifier: 10.4108/ICST.COLLABORATECOM2009.8366
http://dx.doi.org/10.4108/ICST.COLLABORATECOM2009.8366

systems are "transactional" in that all information required for an authorization decision is evaluated at each request. In contrast, in order to amortize the cost of detecting transitive authorization paths over multiple authorization operations, the first author's dRBAC [6] maintains a "credential wallet" that caches currently active credentials and already discovered authorization relationships.

A variety of security models have been proposed for peer-to-peer applications such as [9] Project JXTA, which rely on a PGP-like "web of trust", achieving modulation through the registering of "personal opinions". Other systems provide credit to systems witnessed to provide quality service or gather evidence when freeloaders inappropriately deny service to others (e.g. [10]). However these systems works on a ad-hoc manner meaning these distrustful behavior does not effect already trusted relationship(s) with other entities. In this context, VPAF can re-evaluate dependent trust relationships and notify access controllers.

B. Distributed Credential Discovery

Clarke et al [11] recognized the utility of reachability closures in credential discovery. dRBAC filters these closures for proofs that satisfy a required attribute value range restriction. [4], [6] contemporaneously developed a credential discovery mechanism in the context of the RT0 system that utilizes search tags embedded within authorization credentials. Furthermore, as observed by Czenko [12], if credentials can be reliably discovered, trust management can be non-monotonic. Our scalable authenticated Fern dictionary has this desirable property [7], [8].

C. Credential Validation and Revocation

Naor et al's skiplist-based [13] and our PATRICIA-tree based Fern [7], [8] authenticated dictionaries offer benefits over both online positive authorization schemes such as OCSP [14] and revocation schemes such as CRLs [15]. A client monitoring the status of a certificate using OCSP must continuously poll an authorized server (even when the credential has not changed). In contrast credential validity subscriptions only require minimal server and network resources when a credential has been updated. A feature of Fern is that it can enable scalable non-monotonic trust management [12] due to its ability to support reliable discovery of negative credentials.

III. Proposed Architecture for VPAF

As illustrated in Figure 1, VPAF's active trust management framework's principal component is an Authorization Management System (AMS) responsible for coordinating the activities of VPAF's other components, which include a Client Interface, Validation and Discovery Module (VDM), and Authorization Module (AM).

Access controllers requesting authorization decisions establish an authorization subscription with VPAF's Client Interface by providing the identity of the requesting agent, authorization credentials, and a specification of authorization criteria. These are stored within the AMS. The identity of the agent and the

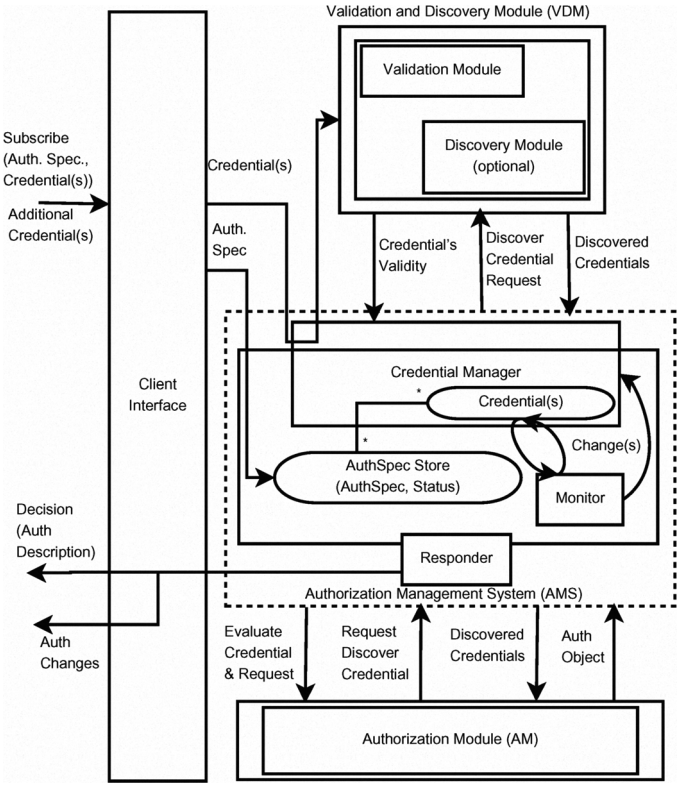

Fig. 1. Schematic Diagram of VPAF

authorization criteria are stored as authorization specification or "AuthSpec" within an "Authorization Specification" store. Credentials are stored within the AMS's "Credential Manager." The validity of credentials stored within the AMS are first checked, and then monitored by a replacable "Validation and Discovery Module" (VDM).

The "Authorization Module" (AM) is responsible for determiing the authorization status of an AuthSpec and is notified by the AMS when an associated credential's authorization status changes. In is notified when the VDM the status of a credential is updated.

AuthSpecs and credentials associated with them are presented to are stored in a many to many relationship. A monitor keeps track of the stored credentials and notifies the Credential manager in case of a change. Figure 1 depicts the major components of VPAF. An access control module requesting confirmation that some subject satisfies a required authorization criterion requests that VPAF create a corresponding AMS. This request may contain evidence in the form of certificates containing credentials that must first be validated by the VDM. The VDM and AM subsystems are invoked by AMS to validate the credential(s) and authorize the request. AMS is also responsible for communicating back to client interface regarding the decision taken.

A. Client Interface

Access controllers will connect to VPAF via a "Client Interface." In order to enable prolonged authorization, the pro-

Digital Object Identifier: 10.4108/ICST.COLLABORATECOM2009.8366
http://dx.doi.org/10.4108/ICST.COLLABORATECOM2009.8366

tocol will permit access controllers to establish subscriptions to authorization decisions. In addition, access controllers can present additional authorization credentials to VPAF.

The client interface also provides a mechanism to communicate authorization decisions (and any changes) to the access controller.

B. Authorization Management System (AMS)

VPAF's AMS tracks the status of requests and certificates managed by VPAF. As they are validated, AM evaluates whether unauthorized requests are satisfied. It manages the credential - validity pairs with authorization requests in a many-to-many relationship, meaning a credential can be used for many requests as well as a single request may require many credentials. The monitor keeps track of the credentials and asks the authentication module to re-evaluate decision(s) if credentials changes.

C. Validation and Discovery Module (VDM)

As illustrated in Figure 1, the certificate validation and Discovery mechanisms are tightly coupled within the VDM module. This tentative design decision is driven by our observation that protocols for credential validation and discovery are frequently linked. For example, Fern provide both services and can simultaneously monitor validity of credentials published by multiple certificate authorities and thus can obtain and validate certificates from multiple mutually distrustful collaborators. We may revisit this design choice as the implementation proceeds.

A single validation unit may implement multiple validation policies. For example, a credential may only be valid over a limited range of dates and also specify an online protocol for detecting revocation. We are examining the potential of implementing a generic validation interface that will permit multiple independent validation mechanisms to be installed simultaneously in a manner that they are selected automatically based upon a credential's characteristics. A similarly modular model is also being considered for the Discovery Unit, When a credential is encoded within a certificate's validation structure, the corresponding validation mechanism will extract and store it within its AMS.

D. Authorization Module (AM)

VPAF's AM is responsible for determining when the set of validated credentials indicates that a subject's authorization satisfies the constraints specified by its AuthSpec. Upon change in related credentials' status, the AM will be tasked to evaluate/re-evaluate whether an authorization decision must be updated.

Rather than implement a single policy within the Authorization Module, we instead provide a generic interface. Like the Validation Module, we are examining the potential of permitting multiple policy engines.

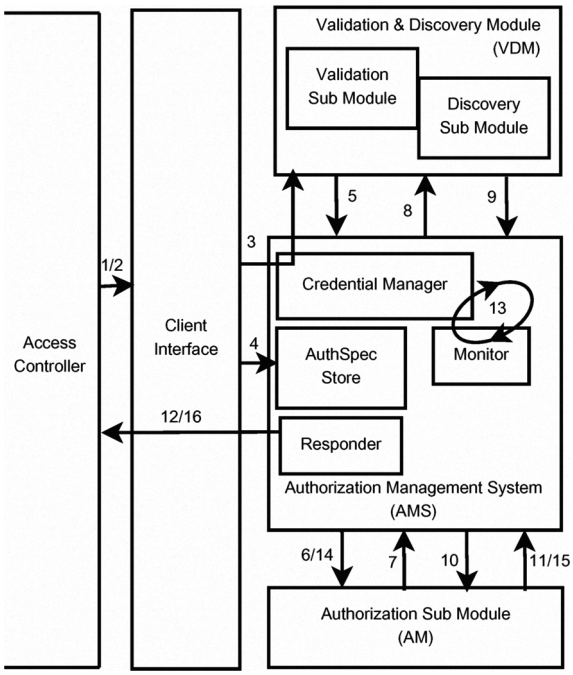

Fig. 2. Communication Flow Diagram of VPAF

TABLE I
TYPICAL COMMUNICATION FLOW WITHIN VPAF

Step	Typical communication flow within VPAF.
1	Authorization Request (AuthSpec + Credential set are received from the Access Controller
2*	Additional credentials received (if provided)
3	Credentials are sent to Credential Management System who forwards them to validation module
4	AuthSpec is forwarded to Auth Spec. Store
5	Validation module asynchronously informs the AMS when changes to validation status are determined for particular credentials
6	As validity of credentials changes AMS requests AM to determine if authorization status is changed
7*	Authorization module queries for additional credentials
8*	Discover requests are forwarded to VDM for discovery
9*	VDM reports validity of newly discovered credentials to AMS
10*	AM is informed about new credentials
11	AM reports status of the authorization request which gets stored in Auth. Spec. Store
12	AMS reports the authorization status to Access Controller
13	Monitor continuously monitors the credential manager changes in credentials
14	Upon detection of change(s), AM is asked for re-evaluation
15	Re-evaluated decision is notified and stored in AMS
16	Access controller is notified of the re-evaluation decision

Steps marked with * are optional

E. Data flow within VPAF

Figure 2 indicates the major data flows within VPAF. Each edge represents a communication, and their purposes are tabulated in Table 1 and described below. In Step 1, an

Digital Object Identifier: 10.4108/ICST.COLLABORATECOM2009.8366
http://dx.doi.org/10.4108/ICST.COLLABORATECOM2009.8366

authorization-sensitive access controller issues a request to a trusted VPAF that identifies (a) an Auth. Spec. representing the request, identification of requester and required criteria to grant the request (b) evidence of the subject's eligibility for the request..

Step 2 illustrates the providing of additional credentials of relevance to the authorization process which may be provided at any time after the initial request.

AMS caches the AuthSpec and validated credential(s) as evidences and asks AM to evaluate the request in step 3, 4, 5 and 6.

Step 7, 8, 9 and 10 shows the discovery process, if required.

After the decision is taken the Authorization object is generated with authorization status. After receiving the same, AMS asks Client Interface to notify access controller about the decision in step 11 and 12.

The monitor keeps track of authentication changes and asks for re-evaluation in case of any change. In which case, from step 13 to 16, it notifies the access controller about changes in authorization.

IV. Implementation Objective

The primary objective of our initial implementation will be a evaluation of the proposed architecture's utility and flexibility. The initial implementation will be constructed in the python language using the event-driven Twisted libraries.

Our initial authorization module will implement a transitive trust management system and our initial credential validation module will utilize a Fern client and chronological expiration. Once successful, we will also utilize VPAF to simulate an existing transitive authorization system for coalition contexts.

We are considering several initial target applications. As suggested in the Introduction of this paper, a strictly authorized DHT-based content distribution network (e.g. Coral) suitable for coalitions that span multiple distrustful organizations is a potential target. We also are evaluating the complexity of re-implementing the trust mechanisms of common protocols such as DNS using this framework. Finally, we are considering implementing mechanisms for login sessions that are effectively terminated if authorization is lost, and mutual authorization of mobile code and execution containers.

V. Synopsys

We are constructing a flexible authorization mechanism that separates its major functions into replaceable modules. Our evaluation will examine the effectiveness of this approach.

References

[1] M. Freedman, E. Freudenthal, D. Mazires, and D. M. Eres, "Democratizing content publication with coral," in *In NSDI(USENIX)*, 2004, pp. 239–252.

[2] M. Blaze, J. Feigenbaum, and J. Lacy, "Decentralized trust management," in *Proc. CCS*. ACM, 1996.

[3] M. Blaze, J. Feigenbaum, and A. D. Keromytis, "KeyNote: Trust management for public-key infrastructures," in *Proc. of Security Protocols International Workshop*, 1998.

[4] N. Li, W. Winsborough, and J. Mitchell, "Distributed credential chain discovery in trust management," in *Proc. CCS*. ACM, 2001.

[5] C. M. Ellison, B. Frantz, B. Lampson, R. Rivest, B. M. Thomas, and T. Ylonen, "SPKI Certificate Theory," IETF RFC 2693, 1998. [Online]. Available: http://www.ietf.org/rfc/rfc2693.txt

[6] E. Freudenthal, T. Pesin, L. Port, E. Keenan, and V. Karamcheti, "dRBAC: Distributed Role-Based Access Control for Dynamic Coalition Environments," in *Proc. ICDCS*, 2002.

[7] E. Freudenthal, D. Herrera, S. Gutstein, R. Spring, and L. Longpré, "Fern: An updatable authenticated dictionary suitable for distributed caching," in *Computer Network Security: Fourth International Conference on Mathematical Methods, Models and Architectures for Computer Network Security, MMM-ACNS 2007, St. Petersburg, Russia, September 13-15, 2007, Proceedings*. Springer, 2007, p. 141.

[8] E. Freudenthal, D. Herrera, S. Gutstein, R. Spring, and L. Longpré, "Fern: An updatable authenticated dictionary suitable for distributed caching," Computer Science Department, University of Texas at El Paso, Tech. Rep. 06-45, 2006.

[9] R. Chen and W. Yeager, "Poblano: A Distributed Trust Model for Peer-to-Peer Networks," Available at http://www.jxta.org/project/www/docs/trust.pdf, 2001.

[10] A. Haeberlen, P. Kouznetsov, and P. Druschel, "Peerreview: practical accountability for distributed systems," in *SOSP '07: Proceedings of twenty-first ACM SIGOPS symposium on Operating systems principles*. ACM, 2007, pp. 175–188. [Online]. Available: http://dx.doi.org/10.1145/1294261.1294279

[11] D. Clarke, J.-E. Elien, C. Ellison, M. Fredette, A. Morcos, and R. L. Rivest, "Certificate Chain Discovery in SPKI/SDSI," Available at citeseer.nj.nec.com/article/clarke99certificate.html, 1999.

[12] M. Czenko1, H. Tran, J. Doumen, S. Etalle1, P. Hartel, and J. den Hartog, "Nonmonotonic trust management for p2p applications," in *Proceedings of the First International Workshop on Security and Trust Management (STM 2005)*, vol. 157, May 2006, pp. 113–130.

[13] M. T. Goodrich, M. Shin, R. Tamassia, and W. H. Winsborough, "Authenticated dictionaries for fresh attribute credentials." in *iTrust*, ser. Lecture Notes in Computer Science, P. Nixon and S. Terzis, Eds., vol. 2692. Springer, 2003, pp. 332–347.

[14] M. Myers, R. Ankney, A. Malpani, S. Galperin, and C. Adams, "X.509 Internet Public Key Infrastructure Online Certicate Status Protocol," IETF RFC 2560, 1996. [Online]. Available: http://www.ietf.org/rfc/rfc2560.txt

[15] R. Housley, W. Ford, W. Polk, and D. Solo, "Internet X.509 Public Key Infrastructure Certificate and CRL Profile," IETF RFC 2459, 1999.

Digital Object Identifier: 10.4108/ICST.COLLABORATECOM2009.8366
http://dx.doi.org/10.4108/ICST.COLLABORATECOM2009.8366