Scalable Access Control For Web Services

Gayatri Swamynathan
Department of Computer Science
UC Santa Barbara, CA 93106
gayatri@cs.ucsb.edu

Tyler Close, Sujata Banerjee, Rick McGeer
HP Labs
Palo Alto, CA 94043
firstname.lastname@hp.com

Ben Zhao, Kevin Almeroth
Department of Computer Science
UC Santa Barbara, CA 93106
{ravenben,almeroth}@cs.ucsb.edu

Abstract

Controlling access to a large distributed service is a potentially error prone process that may negatively impact request throughput and usability. Our Authorization-Based Access Control (or ABAC) URL rewriter solves this problem by providing locally verifiable authorizations and delegation tracking compatible with common web tools. Our access control mechanism is reusable, distributed and meets the scaling requirements of large distributed services. We demonstrate the successful operation of our proposed mechanism on HP's real-time network monitoring and measurement web service, S^3.

1 Introduction

Controlling user access is critical to the successful operation and widespread adoption of large-scale distributed web services. The security framework for such services must address the three-fold challenge of user authentication (identity verification), authorization (access provided to user) and accountability (monitoring activity and controlling abuse) in order to control user access and prevent distributed-denial-of-service (DDOS) attacks.

Access control lists (or ACLs) have remained a popular choice for securing distributed web applications due to their simplicity and ease of application integration. In ACL-based authorizations, a user is authenticated by a challenge-response mechanism (such as passwords or digital certificates) and the request is granted if the user responds properly to the challenge and possesses access rights to the requested resource. Employing ACL-based authorizations for large-scale distributed services lends itself to several problems. First, servers or nodes in the system need to either contact a central server to determine a users privilege, or maintain a local ACL. Granting or revoking user access involves synchronization between all nodes in the system. The need for periodic synchronization could bring the system down in the case of a network partition.

Second, as the overall system grows, dependence on central lists also aggravates the problem of scalability. ACLs are also known to suffer poor access times for large lists. Finally, ACLs are a coarse-grained authority mechanism and lack the flexibility to easily implement multiple security policies and fine-grained authorities.

Large-scale distributed web services, consequently, are in need of a robust and scalable access control solution. Moreover, in the hands of malicious users, such services make for an effective distributed-denial-of-service tool. In this paper, we present *ABAC (Authorization-Based Access Control) URL rewriter*, a reusable, distributed, capability-based access control solution that solves the challenge of securing large-scale distributed services in a highly scalable manner.

We implement our capability-based design on S^3, HP's real-time network monitoring and management

Fifth International Conference on
Creating, Connecting and Collaborating through Computing(C5'07)
0-7695-2806-6/07 $20.00 © 2007 IEEE

web service comprising hundreds of machines distributed across a geographically dispersed wide-area network [3]. Our solution is lightweight with minimal memory and run-time overheads. Additionally, we employ only standard web tools to develop our security solution which proves its quick and easy integration with existing distributed web applications.

The remainder of the paper is organized as follows. We discuss the design and implementation of the ABAC URL rewriter in Section 2. Next, we analyze our design and its performance in Section 3. Our conclusions and future work are presented in Section 4.

2 Approach

We tackle the problem of user authorizations by employing a capability-based security design. While ACL-based authorizations permit users to name resources and then verify whether users have access to the identified resource (for example, using password-based schemes), a capability-based approach prohibits a user from even identifying a resource she does not have access to. Only users that possess a *capability* or authority - an unforgeable pointer to a resource - have the ability to identify and access a resource.

Before discussing the design of our capability-based ABAC URL rewriter on the S^3 web service, we first introduce the operation of S^3 and present the objectives that drive our design framework for large-scale distributed services similar to S^3.

2.1 S^3: A Scalable Sensing Service

S^3 is a Scalable Sensing Service for real-time monitoring and management of large networked systems. S^3 represents a typical web service comprising hundreds of machines distributed across a geographically dispersed wide-area network. Each machine or server in the S^3 infrastructure operates a sensing pod which is a web-service enabled collection of lightweight measurement sensors that collect network information at the machine. Only authorized users of S^3 are presented the ability to conduct third-party sensing measurements between any two machines using uniform resource locators (URLs) that identify the machines and the sensing service(s) requested.

The S^3 sensing information enables network management components to detect network failure or anomalous behavior, improve path selection, and make network decisions at very fine timescales. Fast response, consequently, is a critical requirement for users of the sensing data. Centralized security solutions lead to poor response times, and also subvert the operation of the system in the event of network partitioning. Access control should also scale easily to integrate the addition of new users and measurement servers to the system. Finally, user authorizations need to be fine-grained to accommodate various policy specifications.

2.2 Objectives

The design of the ABAC URL rewriter is guided by several goals that are necessary requirements for a successful access control mechanism.

- Decentralized: A server must continue processing requests correctly, even if under network partition.
- Efficient: The mechanism must not adversely impact request throughput.
- Lightweight: Both server and client deployment costs must be minimal and must integrate well with existing web tools, such as web browsers.
- Open: The solution must facilitate and manage the inclusion of additional users, servers, and services.
- Extensible: The mechanism must support the expression of additional access checks on requests.

2.3 The ABAC URL rewriter service

To meet these objectives, our access control solution interposes a URL rewriter service between each protected web service and network clients. The central idea behind our design is that the underlying service cannot be directly accessed from the open Internet; rather, it can only be accessed by the URL rewriter. Each network client presents to the rewriter service a request to access the underlying service, and a token which explicitly grants authorization to used the service. The URL rewriter satisfies itself that the token

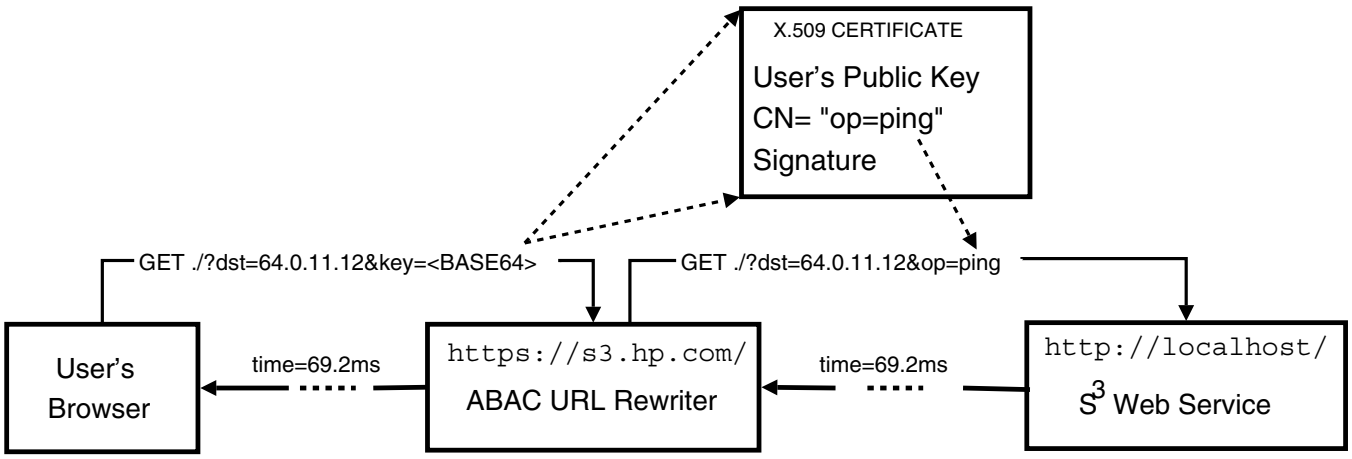

Figure 1. Request-response interaction

is genuine and still valid, then invokes the underlying service on the client's behalf, and finally returns the results to the original requestor.

Each network client and URL rewriter service is issued an *identity certificate*, signed by an identity certificate authority (CA) known to all clients and servers. Additionally, each client is issued a set of *authority certificates* by an administrative CA server. An authority certificate contains the public key of the authorized client and a URL query string specifying the authorized request arguments. For ease of implementation, this URL query string is contained in the common name (CN) field of the X.509 certificate (as seen in figure 1). For instance, S^3 users are given authorizations to access different sensing services like ping, traceroute, etc. The string 'ping' in the CN field would indicate the capability of a user to only ping any destination machine, while the string 'traceroute' in the CN field would indicate the capability to only perform the S^3 traceroute operation between any two destination hosts. A client's identity and authority certificates share the client's public key. This public key binds the various authority certificates of a user to its single identity certificate.

Both the identity and administrative CA servers are hosted by the operator of the S^3 service. These servers manage the addition of users to the S^3 service and the creation, delegation and tracking of authorizations. A new user account is created with a copy of the user's X.509 identity certificate, email address and an optional description of the user. To delegate, the holder of an authorization sends a request specifying the authority certificate to delegate and the X.509 identity certificate of the user to delegate to. The administrative server, after checking the authorization certificate using the algorithm similar to that in figure 2 (described later), will log the delegator's identity certificate, the authorization certificate, and the delegates' identity certificate. The administrative server then returns a newly created authorization certificate, specifying the same URL transform, but the delegates' public key. The delegator can then transmit the new authorization certificate to the delegate by any convenient means, such as email.

All authorization certificates are only valid for a given time interval. Near expiration, the administrative CA server collects the request logs from all the measurement servers and searches them for use patterns deemed abusive according to an algorithm specified by the S^3 service operator. For example, using a given 'ping' authorization, or any of its delegations, more than 10,000 times a month, may be deemed abusive. The S^3 service operator can then choose to notify the users of the abuse, change the detection algorithm so that the pattern is not deemed abusive, or terminate the identified user accounts. Only active user accounts may refresh their authorizations for use in the next service interval.

Users of the S^3 service are notified of the system's logging features and usage expectations and so are aware they can be held accountable for any abuse, either by themselves or any of their delegates. Account-

0-7695-2806-6/07 $20.00

ability is expected to foster careful use, and delegation, of authorizations.

2.4 The ABAC algorithm

A typical runtime request through the URL rewriter is depicted in Figure 1. The algorithm implemented by the ABAC URL rewriter service is described by the pseudo-code in Figure 2.

Once an SSL session is set up between a client and a server, we first check whether the authority certificate presented by the client is valid and issued by the administrative CA. Assuming the administrative CA keeps its private key secret, a client is unable to forge an authority certificate that will pass verification.

Next, we ensure that the authority certificate is presented only by the client that it was issued to. Assuming the client keeps its private key secret, the SSL protocol ensures an attacker is unable to setup an SSL connection under the client's public key. This proof also establishes ownership of the authority certificate, since it must specify the same public key. As a result, an attacker who steals a copy of an authority certificate is unable to use it, since without the corresponding private key, the certificate cannot be presented over an authenticated SSL connection. Once a client is verified as an authorized user, S^3 performs the measurements and returns a response to the client.

By embedding the authorization certificate within a request URL argument, the protected web service remains usable by common, unmodified web browsers. All major web browsers support the SSL protocol, as well as installation of client identity certificates. Although unusually long, the request URLs can be used in normal HTML hyperlinks and forms. Modern web browsers do not have the URL length limitations of earlier generations.

Through web browser compatibility, the human user can operate the protected web service, unaware of any of the cryptographic operations used for access-control. Delivery of authorizations to a user can take the form of an HTML web page, or email, containing a series of hyperlinks, one for each authorization. Since the authorizations are only usable by the intended recipient, interception or theft of the web page, or email, is not an issue.

3 Analysis and Performance of ABAC

Some salient features of our access control design include the following.

- Attack-resistant design: We employ secure sockets in our capability-based design to deny unauthorized users and servers from maliciously participating in the system. Second, a malicious user snooping the network for authority certificates will be unable to use the acquired capabilities since it needs to be aware of the corresponding identity certificates used to initiate the SSL sessions. Third, it is impossible for authorities to be forged without the possession of the administrative CAs private key. Finally, each users activity is logged to audit usage and counter abuse.

- Decentralized design: Our capability-based solution is decentralized and operates relatively independent of any central management. Each S^3 measurement server only stores the public certificates of the identity CA and the administrative CA, and verifies the authenticity of a client identity or authority certificates using these public root certificates. No central access control lists are necessary because the possession of a authority implies that the user is authorized to request the service. Response times are quick and the security framework is robust to network partitioning.

- Lightweight and scalable: Our design is highly scalable and easily accommodates the addition of new S^3 users, servers and services. New users are assigned authorities based on their authorization level and registered with the system. New servers only need to possess the public certificates of the two CAs in order to verify clients. Lastly, addition of new S3 services does not affect the security wrapper in any way.

- Negligible run-time overheads: As seen in figure 3, our performance results for S^3 indicate that authority certificate verification takes about 0.0025% of the overall request run-time (less than 0.0005 seconds) for a 4096 bit certificate. This overhead is expected to also be negligible in typical web services.

LABELS
$id - cert_c$: Client's X509 identity certificate obtained from SSL session setup
$auth - cert_c$: Client's X509 authority certificate obtained from request URL

BEGIN
1: Create SSL Session between client c and server m
2: **if** $auth - cert_c$ not valid, **then**
3: Quit /*Forged authority detected*/
4: **end**
5: if public_key ($id - cert_c$) != public_key ($auth - cert_c$), **then**
6: Quit /*Stolen authority detected*/
7: **end**
8: Log request URL /*Monitor usage to detect abuse*/
9: Replace $auth - cert_c$ URL query string argument with CN field from $auth - cert_c$
10: Forward request to protected web service and return response to client
END

Figure 2. ABAC URL Rewriter Algorithm

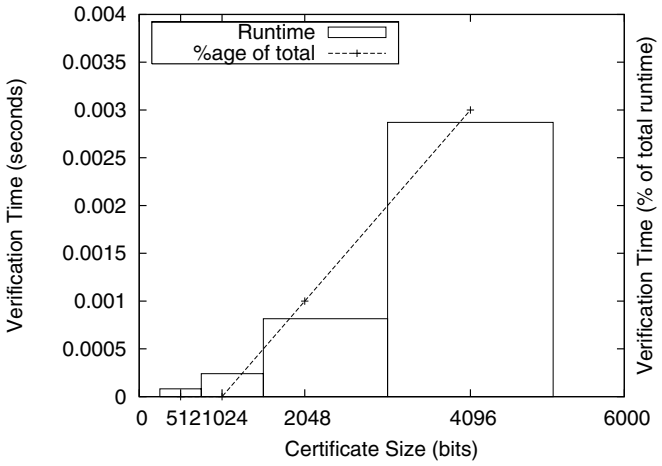

Figure 3. Certificate Runtime and Overhead

- Ease of application integration: We employ standard web technologies (like openssl) to develop our security solution and this enables quick and easy integration with existing web applications.

3.1 Competitive Approaches

Access to a web service is typically controlled through the use of username/password ACLs, implemented by HTTP Auth [2] or HTTP cookies. For the S^3 service, this approach was rejected due to the complexity, and fragility, of maintaining consistent ACLs across all measurement servers. Either a measurement server must check each request against a canonical ACL housed on a central server, or changes to the canonical ACL must be securely broadcast to all measurement servers. Both approaches fare poorly under network partition and involve significant runtime overhead. A locally verifiable authorization certificate eliminates this coordination burden. Avoiding the use of passwords also helps with usability. A URL with an embedded authorization certificate requires no user effort or action beyond that of the unprotected web-service.

4 Conclusions and Future Work

After successful preliminary experiments, we are now in the process of building a prototype. The ABAC URL rewriter is implemented to the standard CGI interface and is currently deployed on top of the Apache web server. All cryptographic operations are implemented using OpenSSL [1]. Our testbed operates on a set of 15 Planet-Lab nodes that run the S^3 measurement servers wrapped by the ABAC URL rewriter. Our next step is a full-scale deployment on the full S^3 infrastructure. Our vision is to make S^3 an open,

subscription-based web service for network measurements.

Furthermore, while we have layered the authorization protocol on top of http for convenience, we believe that this approach can be used by other application level protocols and, indeed, at the transport or networking layer of the stack. The critical feature that we require is the ability to pass an encrypted authorization to a service authenticator, and this can be supported by virtually any well-designed protocol.

References

[1] Openssl: The open source toolkit for SSL/TLS, http://www.openssl.org/.

[2] RFC 2617.

[3] P. Yalagandula, P. Sharma, S. Banerjee, S. Basu, and S.-J. Lee. S3: a scalable sensing service for monitoring large networked systems. In *INM '06: Proceedings of the 2006 SIGCOMM workshop on Internet network management*, September 2006.

0-7695-2806-6/07 $20.00 © 2007 IEEE