Technical Article for Conduit!

Efficient Low-Cost Authentication of Distributed Data and Transactions

Roberto Tamassia

Department of Computer Science
Brown University

Introduction

Security is an essential requirement of distributed platforms for business applications. We address the problem of authenticating high volumes of data and transactions in non-trusted distributed environments. Current solutions are typically:

- ***Centralized*** – and therefore subject to network delay and denial-of-service attacks
- ***Expensive*** – to build, maintain, and operate because trusted data must be maintained in a secure environment, guarded 24x7
- ***Non-scalable*** – with limited throughput due to operating and economic constraints

We are developing a high-throughput system for authenticating data and transactions in non-trusted environments, at the network edge and outside the firewall. We use a unique, patent-pending approach for wide distribution of authentication information. As a result, we can dramatically lower the cost of authentication in such applications as wireless authorization, B2B ecommerce exchanges, distributed storage, end-to-end integrity, tamper detection, and certificate revocation checking.

This work is being conducted in collaboration with AlgoMagic Technologies, Johns Hopkins University, and the University of California, Irvine, and is supported in part by a grant from the Defense Advanced Research Projects Agency. My collaborators Steve Baron, Robert Cohen, and Rich Sneider at AlgoMagic Technologies and Mike Goodrich at the University of California, Irvine contributed to writing this article.

For more information, visit **http://www.cs.brown.edu/cgc/stms/**

Example Application Scenario

Imagine a ***trusted financial portal*** service (such as Yahoo! Finance) that provides dynamic and up-to-date financial information, including corporate and news feeds, market quotes, and online trading. Some key characteristics of this distributed application are:

- High volume of accesses, queries, and transactions
- Widely distributed clients (e.g., geographically dispersed and with varying access bandwidth)
- Information originating from various sources outside the portal (e.g., stock exchanges, government agencies, and corporate investor relations offices)

How can such a portal become fully trusted so that the integrity of all data and transactions is guaranteed authentic end to end? Today, the economics are daunting and the implementation would be largely impractical. For example, HTML pages may be dynamically constructed from several frequently changing information sources, based on user requests and user profiles. The volume and diversity of such requests makes creating individual time-stamped digital fingerprints for any given page very difficult and time-consuming. Common techniques for load balancing, replication, and multihosting subdivisions of a web site make using previously available trust approaches unrealistic, since access to sensitive data must be limited and strictly secured.

Our distributed authentication approach offers significant new capabilities and could dramatically change the economics of authentication for distributed business applications. Highlights of our approach include:

- It works with and leverages existing Internet infrastructure, protocols and computing platforms
- It is highly scalable in terms of both computational and economic cost
- It provides succinct and timely authenticity proofs for entire collections of data originating from diverse sources
- It widely distributes authentication information to non-secure replication locations, while strictly maintaining trust
- It responds to authentication requests by clients with minimal network latency and computational cost

We have developed a distributed authentication system called ***Secure Transaction Management System*** (***STMS***). A fully operational prototype of STMS has been implemented.

STMS Technology

Authenticated Dictionaries

The computing abstraction underlying STMS is a data structure called an ***authenticated dictionary***, which is a system for distributing data and supporting authenticated responses to queries about the data.

In an authenticated dictionary, the data originates at a secure central site (the ***repository***) and is distributed to servers scattered around the network (***responders***). The responders answer queries about the data made by ***clients*** on behalf of the repository.

It is desirable to delegate query answering to the responders for two primary reasons:

1. It is undesirable that the repository provide services directly on the network due to risks such as denial-of-service attacks
2. The large volume and diverse geographic origination of the queries require a distributed system of servers to provide responses efficiently (much like DNS)

Previous Approaches

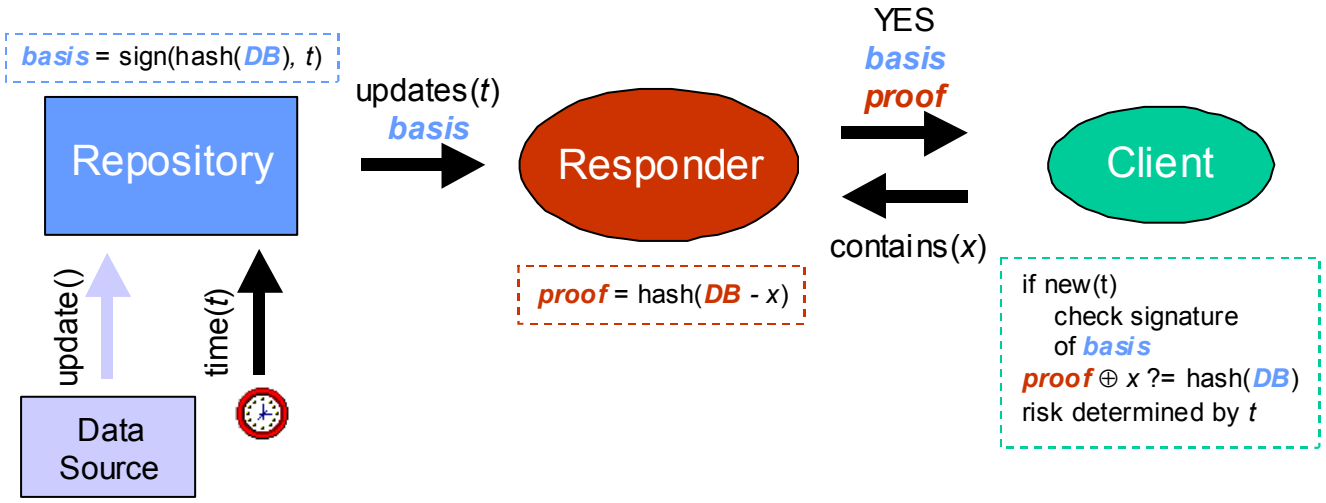
The simplest approach to implementing an authenticated dictionary is to make the responders trusted parties. This ***trusted-responder approach*** is used, for example, in the online certificate status checking protocol (OSCP), where trusted OCSP responders answer certificate revocation queries posed by clients. The main disadvantage of the trusted-responder approach is that each responder must be placed is a secure location, with ensuing large operational costs. Thus, this approach is not scalable for economic reasons.

An alternative approach entails having the repository periodically sign a fingerprint of the current version of the entire database. The database itself, together with the signed, time-stamped fingerprint is then sent to the client as a proof of the answer. This ***database-forwarding approach*** is used, for example, to authenticate certificate revocation lists. The database-forwarding approach allows non-trusted responders and thus has lower operational costs. However, it is computationally demanding for the client, which needs to process the entire database in order to validate an answer. Also, it is not scalable for communication reasons, since sending the entire database together with the answer uses considerable network bandwidth.

The STMS Approach

The main feature of STMS is that it maintains trust even when responders are located in insecure, non-trusted locations. That is, when a client makes a query to an STMS responder, it gets back not only an answer but also a proof of the answer. The client can easily validate the answer and determine that the responder has not been tampered with, while relying solely on trusted statements signed by the repository. The design of STMS allows untrusted responders to provide verifiable authentication services on behalf of a trusted repository. This unintuitive yet mathematically provable fact is the key to achieve cost effectiveness.

The figure shows a high-level view of the STMS parties and protocol. The repository sends periodic ***updates*** to the responders together with a special signed time-stamped fingerprint of the database called the ***basis***. A responder replies to a query with an ***authenticated response***, consisting of the ***answer*** to the query, the ***proof*** of the answer and the ***basis***. Informally speaking, the proof is a "partial fingerprint" of the database that, combined with the subject of the query, should yield the fingerprint of the entire database. A proof consists of a very small amount of data (less than 300 bytes for most applications) and can be validated quickly. The client finally evaluates the risk associated with trusting the answer using the freshness of the time-stamp.



Our patent-pending algorithms, which employ only standard cryptographic functions for signing and hashing, guarantee that the following tasks can be performed in near real time and with minimal use of computing resources:

- Creation of the new signed basis by the repository for each time quantum (the basis is incrementally updated)
- Update of the database copy residing at a responder for each time quantum (the copy is incrementally updated)
- Assembly of the proof by a responder for each query (the proof is obtained by combining precomputed partial proofs)
- Query formulation by a client
- Proof verification by a client

The key benefits to the STMS approach, include:

- Answers given by the responders are as trustworthy as if they came from the repository
- Data is distributed close to the clients, minimizing network delays
- Deploying responders is inexpensive
- The repository is protected from risks such as denial-of-service attacks

An additional unique advantage of STMS is that it supports ***historical persistence***; that is, one can perform queries on past versions of the database, which makes possible a variety of nonrepudiation applications.

STMS comes with Java and C++ toolkits for building clients, including modules for validating responses. For maximum reliability and security, cryptographic functions are performed with the available standard cryptographic libraries of Java and Windows. We also provide an XML-based protocol for querying a responder over an HTTP connection. We are currently working on a Web services approach in which STMS is accessed through the SOAP protocol.

The table compares the STMS approach with the trusted-responder and database-forwarding approaches with respect to operational cost, run-time performance, scalability, and support for historical persistence.

	Cost	**Performance**	**Scalability**	**Persistence**
Trusted-responder	High	High	Low	No
Database-forwarding	Low	Low	Low	No
STMS	Low	High	High	Yes

Applications

As a foundation for delivering a wide range of trust services, STMS brings authentication to the network edge and can be viewed as a distributed low-cost "authentication cache" for high-volume transactions. Classes of trust applications in which STMS can be exploited include:

- User authentication and access control for large-scale corporate portals, private exchanges, and Virtual Private Networks – User signon can be accomplished without the need to contact a remote central server.
- Wireless two-way authentication – Mobile device identity can be confirmed by gateways, network services, and applications, and devices can also confirm the identity of selected network services and applications to establish two-way trust.
- Trusted XML Web Services – Two-way trust can be established between the parties involved in a "chains" of Web Services. Keys for signed or encrypted portions of XML can be obtained from widely distributed servers (STMS responders) rather than centralized systems.
- Certificate status checking (e.g., for SSL, secure email, and code certificates) – Identity and status can be mutually verified by both sender and receiver of email, publisher and user of code components, client and server in a secure dialog.
- Validation services for business partner networks (e.g., healthcare providers) – Access rights, roles, and derived trust assertions can be quickly checked against a localized store of consolidated credentials. Public networks can be employed instead of incurring the cost of a private value-added network.
- End-to-end integrity for content collection and distribution systems (the example scenario presented at the beginning of this article).

- Non-repudiation services for electronic commerce – third-party verification and audit capabilities can be implemented to support large-scale B2B networks (e.g., corporate portals, exchanges, trading systems), as well as high-volume B2C systems (e.g., media distribution, digital tickets, electronic payment systems).
- Secure DNS – host name to IP address mappings could be upgraded to include digital signatures. The data-distribution approach would be essentially similar to that used today. Widely distributed trusted DNS servers (as STMS responders) would not have to be located in secure settings.
- Tamper detection for Web sites – Web page digital fingerprints can be made available as a reference. The computational load and delay of signing every page on each web server at all replicated locations are avoided.
- Tamper detection for operating systems – System file fingerprints can be made available as a reference on a localized basis for large groups of client systems.
- Tamper detection and license checking for software applications – Software is increasingly distributed or accessed over the network. Metered access can be effectively implemented, identity and status of servers and clients can be mutually confirmed, and component fingerprints can be quickly verified.