# A Trust Management Architecture for TLS

Xiaolei Zhang , Choong Seon Hong
*School of Electronics and Information, Kyung Hee University*
zxl@networking.kyunghee.ac.kr, cshong@khu.ac.kr

**ABSTRACT**

The TLS protocol suite, which provides transport-layer security for the Internet, has been standardized in the IETF. A TLS session is an association between a client and a server, created by the TLS handshake Protocol. Sessions define a set of cryptographic security parameters, which can be shared among multiple connections. The TLS protocol, however, does not address the specific method for how to manage the existing TLS sessions on the host. This paper proposes an efficient management scheme for TLS, based on the principles of trust management.

## 1 Introduction

"Web Security" has been associated with debates over cryptographic technology, protocols, and public policy, obscuring the wider challenges of building trusted Web applications. Since the Web aims to be an information space that reflects not just human knowledge but also human relationships, it will soon realize the full complexity of trust relationships among people, computers, and organizations. Within the computer security community, Trust Management has emerged as a new philosophy for codifying, analyzing, and managing trust decisions.

Transport Layer Security (TLS) can establish a private channel between two processes. A temporary session key is set up for each cryptographic handshake. For avoiding the expensive negotiation of new security parameters for each connection, a session can be shared among multiple connections between any client and server. If we manage the session with trust management, we can not only extend the lifetime of session, but also increase the security of TLS.

## 2 Related Works

TLS extensions [SMDJ01] describe extensions that may be used to add functionality to TLS. It provides both generic extension mechanisms for the TLS handshake client and server hellos, and specific extensions using these generic mechanisms.

TLS Fast-track session establishment [HD01] is a mechanism for reducing handshake network traffic and computation on both ends. In ordinary handshake session resumption relies on a server session cache. But in this proposal, session resumption relies on a client session cache, because client rarely connects to numerous TLS servers and can cache information about server for a longer time. It defines two extending fields in hello messages and two message types. When the session is established, it requires that the client obtains and stores server parameters and negotiated parameters. Both the ClientHello (or ClientHelloFT) and the ServerHello (or ServerHelloFT) messages include the fasttrack_capable extension.

Our proposal defines some extensions in TLS also. With these, TLS handshake will have more flexibility and effectivity.

## 3. Proposed Scheme

In this section we propose a mechanism for TLS based on trust management principles. It is an extension for TLS handshake.

### 3.1 Trust management

Trust management, introduced by Blaze et al. [BFIK99] [BFI+01], is a unified approach to specifying and interpreting security policies, credentials, and relationships. Management system combines the notion of specifying security policy with the mechanism for specifying credentials. Credentials describe specific delegations of trust among public keys; unlike traditional certificates, which bind keys to names, trust-

management credentials bind keys directly to authorizations to perform specific tasks. Trust-management systems support delegation, and policy specification and refinement at the different layers of a policy hierarchy, thus solving to a large degree the consistency and scalability problems inherent in traditional ACLs. Furthermore, trust-management systems are extensible and can express policies for different types of applications.

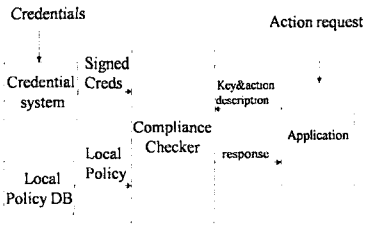


Figure1 Trust management architecture

A trust-management engine is separate system component that takes (request, Credentials, Policy) as input, outputs a decision about whether compliance with policy has been proven, and may also output some additional information about how to proceed if it hasn't (refer Figure1).

### 3.2 Transport Layer Security

SSL was originated by Netscape. Version 3 of the protocol was designed with public review and input from industry and was published as an Internet draft document. Subsequently, when a consensus was reached to submit the protocol for Internet standardization, the TLS working group was formed within IETF to develop a common standard. The current work on TLS is aimed at producing an initial version as an Internet Standard. This first version of TLS can be viewed as essentially an SSLv3.1, is very close to and has a backward compatibility with SSLv3. The primary goal of the TLS Protocol [TLS] is to provide privacy and reliability between two communicating applications. The protocol is composed of two layers. At the lowest level, layered on top of some reliable transport protocol (e.g., TCP), is the TLS Record Protocol. The TLS Record Protocol is used for encapsulation of various higher-level protocols. One such encapsulated protocol, the TLS Handshake Protocol, allows the server and client to authenticate each other and to negotiate an encryption algorithm and cryptographic keys before the application protocol transmits or receives its first byte of data. One advantage of TLS is that it is application protocol independent. A higher-level protocol can layer on top of the TLS Protocol transparently. In TLS, there are two basic concepts, session and connection. The connection is a transport that provides a suitable type of service. For TLS, it is peer-to-peer relationships, and every connection is associated with one session. A session is an association between a client and a server. It defines a set of cryptographic security parameters, which can be shared among multiple connections. That is used to avoid the expensive negotiation of new security parameters for each connection.

### 3.3 The proposed TM architecture for TLS

We propose a management architecture for TLS, based on the principles of trust management. It can increase the security of TLS, and improve the flexibility. As we mentioned above, session and connection are two basic concepts in TLS. The connection is a transport that provides a suitable type of service. For TLS, every connection is associated with one session. A session is an association between a client and a server. Session defines a set of cryptographic security parameters, which can be shared among multiple connections. It is used to avoid the expensive negotiation of new security parameters for each connection. If we use trust-management, we can easily classify the security level of connections and sessions, and limit a connection to share TLS session.

In our proposal, we have some change in the processing of TLS handshake: In the first case, it is established a new session between client and server. The client sends a client hello that includes a policy built from trust management system. When server receives the client hello message from client, it will trigger trust management system to update corresponding policy in server policy & credentials database, referring the policy from client. And then, client and server exchange the messages as ordinary handshake protocol. (refer Figuer2)

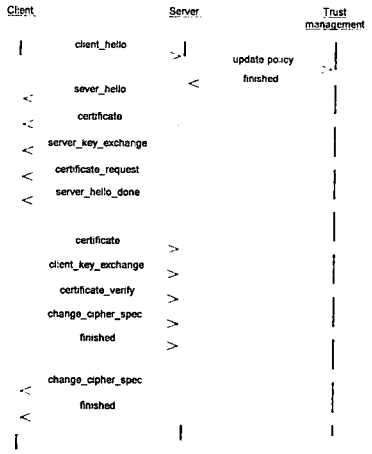


**Figure2 Handshake processing with TM**

In the second case, a client and a server will resume an existing session. The client sends client hello message, which not only includes the session ID, but also is attached credentials about the application. When the server receives this message, it triggers the trust management system to verify the credentials for determining whether establish this connection or not. If it is successful verified, the client and server will continue to exchange message as ordinary. (refer Figure3)

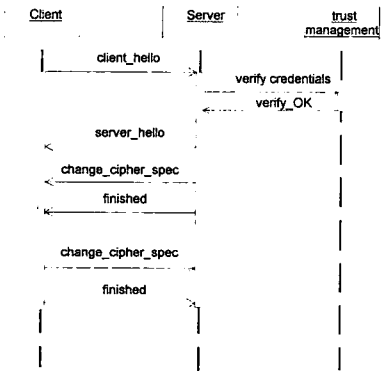


**Figure3 Handshake resume processing with TM**

In our proposal, a framework for trust management for TLS is: Both client and server have their own trust-management-specified policy governing session creation. This policy should describe the classes of session and under what circumstances the client will initiate session creation with the server, and also what types of session are willing to allow other application to share (for example, whether this session can be shared and if so what conditions are acceptable); Servers query their trust-management system to determine whether the proposed session complies with local policy or create a new session.

## 4 Conclusions and Future Work

In this paper, we propose an extension of TLS. We control the processing of the TLS handshake protocol with trust management system. As a trust management system can provide direct authorization of security critical actions and decouples the problem of specifying policy and authorization from that of distributing credentials, we can efficiently control a TLS connection to share TLS session. In this way, we can extend the lifetime of session, and increase the security of TLS. We shall continue to modify the rules for building policy in trust management. As we use new secure mechanism in TLS, we propose to simplify the message flow of TLS handshake in the future.

## References

[SMDJ01] Simon Blake-Wilson, Magnus Nystrom, David Hopwood, Jan Mikkelsen. "TLS Extension", Internet-Draft: draft-ietf-tls-extensions-02.txt, December 2001.

[HD01] Hovav Shacham, Dan Boneh, "TLS Fast-Track Session Establishment", Internet-Draft: draft-shacham-tls-fast-track-00.txt, September 2001

[BFIK99] Matt Blaze, Joan Feigenbaum, and Angelos D. Keromytis. The role of trust management in distributed systems security. In Secure Internet Programming, pages 185-210, 1999. http://www.crypto.com/papers/trustmgt.pdf

[BFI+01] Trust Management, Compliance Checking and Security Policy. Poilcy 2001, Bristol, UK, January 2001. http://www.crypto.com/talks/

[TLS] The TLS Protocol Version 1.0 RFC2246, 1999