Comparison of Security Layers in Store-and-Forward Scenarios

William R. Soley
SunSoft, Mountain View, CA 94043
soley@eng.sun.com

1 Introduction

Various security protocols operate at different layers in the OSI reference model,[3] the advantages and disadvantages of which vary according to the protocol, to the layer it operates at, and to the application. All of this may be confusing when trying to select a security protocol. The possible consequences of an inappropriate selection include weak protection, inefficient use of resources, difficulty of administration, poor scalability, and more.

1.1 Scope

This paper attempts to analyze just one factor affecting the choice of security protocols, namely, the layer at which the protocol operates. Other factors that would need consideration in a complete analysis are beyond the scope of this discussion. In particular, this paper avoids the details of specific security protocols.

Even after a complete analysis, there may not be a simple answer. The best solution for one situation may not be the best for another. In some cases, the best solution may combine more than one security protocol at more than one layer. For example, an application-layer signature on the content might be combined with transport-layer encryption.

1.2 Sample Scenario

This paper compares security layers within the context of a store-and-forward data transfer scenario. It uses a World-Wide-Web transaction as an example since it is familiar to many. However, the analysis generally holds for all store-and-forward applications. (Other examples are electronic mail and network file systems.)

Figure 1 shows the sample scenario. The *author* creates the *content* and uses the *author client* to transfer the content through the *network* to the *server* where it is stored on disk. At a later time, the *reader* uses the *reader client* to retrieve the content through the network from the server.

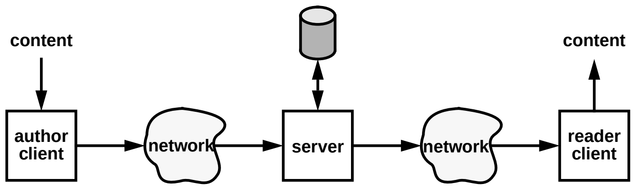

Figure 1 Sample Scenario

1.3 Layers

The three layers being compared here are *network*, *transport* and *application*. These are the layers where most popular security protocols operate, today. This is not to say that security protocols cannot or should not be implemented in other layers.

Examples of network, transport and application-layer security protocols are SKIP[1], SSL[2] and PGP[6], respectively.

2 Comparison

Table 1 shows a comparison of the features generally exhibited in a store-and-forward scenario by security protocols operating at each of the three selected layers. Desirable features are shown in white boxes. Undesirable features are shown in shaded boxes with the least desirable being darker. Some of the features are discussed below.

2.1 Authentication, Integrity, Nonrepudiation

In application-layer security protocols, the author is authenticated directly by the reader (by verifying the digital signature and certificate). The reader may be authenticated implicitly by the author who encrypts the content with the readers' public key, thereby assuring that only the intended readers may decrypt the content.

In the transport-layer case, both the author and reader are authenticated by the server which must be trusted by

feature	security layer		
	network	transport	application
author authentication	no*	by server	by reader
reader authentication	no*	by server	by sender
server authentication	by client	by client	n/a
content integrity	each hop	each hop	end to end
nonrepudiation of authorship	no	no	yes
privacy scales to many readers	yes	yes	no
secure through caching proxy	no	no	yes
traffic analysis exposure	IP addr	TCP port	URL
need to modify application	*no	some	yes
need kernel support	yes	no	no
negotiable crypto algorithms	yes	yes	no
server must be trusted	yes	yes	no
server need crypto code	yes	yes	no
server CPU requirement	high	high	low

Table 1 Comparison

its clients to correctly relay the identities of the other parties.

The end-to-end (author to reader) digital signature assures content integrity in the application-layer case. In the other cases, the protocols assure content integrity over each network hop, but the system is still vulnerable to the content being corrupted while stored on the server's disk.

In the application-layer case, the reader may save a copy of the content with its digital signature (made by the author) and use it at a later time to prove to a third party that the content came from the attributed author. This provides the nonrepudiation of authorship feature without the participation of, or need to trust the server.

2.2 Privacy

In the application-layer case, the author must enumerate all authorized readers when the content is encrypted for privacy. For *n* authorized readers, the author must perform *n* public-key operations and store *n* encrypted keys with the content. For any more than a small *n*, this is likely to be prohibitively expensive in terms of both CPU and storage requirements.

In spite of this potential scaling problem with application-layer privacy, it has the strong advantage that it is transparent to the server (including, for example, caching proxy servers). The server would at no time have access to the decrypted content and therefore need not be trusted with respect to content privacy.

2.3 Traffic Analysis

The headers of layers that are below the security layer are not encrypted and can therefore be observed by an eavesdropper. The information they contain could be used for traffic analysis. In the network-layer case, this would allow the network address (*e.g.*, IP address) of each party to be seen, which is of limited value. But, in the application-layer case, the entire application address (*e.g.*, URL) might be visible, which could be of substantial value.

2.4 Server Requirements

In all but the application-layer case, the content is unprotected while stored on the server's disk. This means the server must be trusted to protect the data from tampering or accidental damage (*e.g.*, I/O error).

In the non-application-layer cases, it is also necessary for the server to decrypt incoming content and reencrypt it each time it is retrieved by a reader. This could place a substantial demand on the server's CPU.

In the application-layer case, the server never needs to do any cryptographic operations and does not even participate in the security protocol.

3 Conclusions

In the store-and-forward scenario, the application-layer is the most favorable for authentication, integrity and nonrepudiation (often the most important features). It is prohibitively expensive as a means for privacy for large numbers of readers, but is favorable for just a few.

Transport and application-layer privacy cooperating together might solve the scaling problem while preserving some of the advantages of application-layer privacy.

Network-layer security is most favorable when complete transparency to the application is desired, especially across multiple applications.

References

[1] Ashar Aziz, "Simple Key-Management For Internet Protocols (SKIP)", draft-ietf-ipsec-skip-06.txt, Internet Draft, Sun Microsystems, Dec 1995.

[2] Alan Freier, Philip Karlton, Paul Kocher, "The SSL Protocol, Version 3.0", Internet Draft, Netscape, March 1996.

[3] ITU, *Reference Model of Open Systems Interconnection for CCITT Applications*, Rec. No. X.200, Geneva, 1985.

[4] Bruce Schneier, *Applied Cryptography, Second Edition, Protocols, Algorithms, and Source Code in C*, John Wiley & Sons, Inc., 1996.

[5] Andrew Tanenbaum, *Computer Networks, Second Edition*, Prentice-Hall, 1988.

[6] Philip Zimmermann, *The Official PGP User's Guide*, MIT Press, 1995.