SANS Institute
InfoSec Reading Room

Protecting Email in a Hostile World with TLS and Postfix

This paper addresses Transport Layer Security (TLS) and how it can be a very effective enhancement to keep email safe, secure, and private.

Protecting Email in a Hostile World with TLS and Postfix

David F. Severski

Protecting Email in a Hostile World with TLS and Postfix

Introduction

Internet-enabled communications have penetrated virtually every aspect of daily life. As businesses and organizations become ever more dependent on this rapid transfer of data, they are also becoming increasingly concerned with guaranteeing the security of the elements that make up this web of information. The transmission of web traffic and the storage of data on servers – the twin cornerstones of e-commerce – have been subject to considerable public scrutiny, resulting in the implementation of a variety of effective security strategies. By comparison, e-mail and its underlying transmission mechanism, Simple Mail Transfer Protocol (SMTP), remain almost completely unsecured in most business environments. Yet virtually all businesses, including those that do not maintain web-enabled storefronts, rely on e-mail to communicate vital information, both internally and to customers and suppliers.

The Problem: Insecure Transmission of E-Mail

The moment a user presses the send key in an e-mail program (also known as a Mail User Agent or MUA) vital data such as the identity of both sender and recipient, the date and time that the message was sent, the path that the message will take, and the contents of the message itself, are completely exposed to anyone who might wish to activate a packet sniffer. Indeed, standard e-mail transactions do not provide for verification of the integrity of e-mail messages. End users have no way of determining whether or not an e-mail message has been tampered with during transfer, nor even if it was written by its purported sender.

The information being sent across this extremely vulnerable channel is often quite sensitive. Credit card numbers, internal memos, legal contracts, and passwords and activation codes are just a few examples of the types of data that are routinely transferred with little thought to the fact that they may easily be intercepted by curious or hostile third parties.

The gravity of this problem has not been lost on computer security professionals and privacy advocates. At their urgings, a variety of techniques for increasing e-mail security have been developed, many of which can be combined to provide a multi-layered, defense-in-depth solution. Pretty Good Privacy (PGP) Encryption and IP Security (IPSEC) Protocol are two of the more popular methods, but both suffer from substantial shortcomings. PGP, which can be used either to authenticate signatures or to fully encrypt complete messages, will not work unless both the sender's and the receiver's MUAs are PGP-aware. Few commercial e-mail clients are shipped in a PGP-aware configuration, which means that end users must rely on additional plug-ins or configure PGP themselves. In addition, PGP may impose a significant processing load on the recipient, making this security feature impractical, especially in the case of hand-held devices, which are often CPU bound. IPSEC allows users to establish complete end-to-end security, regardless of the protocol or the number of hops between sender and recipient. However, configuring IPSEC can be very complex and common networking techniques, such as Network Address Translation (NAT) with port overloading, can render IPSEC incapable of negotiation.

The TLS Solution

Transport Layer Security[1] (TLS) is the Internet Engineering Task Force (IETF) successor to the very successful Secure Sockets Layer (SSL) V2/V3 standards developed by Netscape Communication[5]. While most users are familiar with, and trust, SSL to secure web page transactions for crucial operations such as online ordering, SSL/TLS -- henceforth referred to as TLS -- is a generic wrapper protocol that can be adapted to a wide variety of transaction-based protocols. [In RFC 2487, P. Hoffman outlines the use of TLS by servers wishing to secure e-mail exchanges[3].]

TLS provides certificate-based point-to-point authentication and encryption. Digital certificates, like those used to certify that a website is who it claims to be, are used to authenticate the identity of each communicating party. Once this verification has occurred, a secure encryption channel for e-mail traffic can then be established. Because TLS is a server-side option, no intervention is required on the part of the client or the recipient. Users connecting via TLS-enabled servers benefit from enhanced security without having to change a single setting or alter their usage patterns.

TLS is an optional feature whose use must not prohibit any two public mail servers from communicating[3]. Sessions may be upgraded to TLS if, and only if, a server offers TLS as an option at the start of an ESMTP transaction. When a client connects to a mail server, the client issues an EHLO command, to which the server responds with a list of capabilities. If the server advertises the ability to communicate via TLS, the client may then choose to issue a STARTTLS command in an attempt to negotiate a verified and secure channel. This conversion of an unsecured connection to a TLS-encrypted channel is known as a TLS upgrade negotiation.

How To

Demonstration Configuration

TLS is supported by a wide variety of mail servers, such as sendmail and Microsoft Exchange, as well as many different operating systems, including Solaris, Linux, and Microsoft Windows NT/2000. For the purposes of this walk-through, a reference operating system of FreeBSD 4.3-STABLE (snapshot as of July 8, 2001) will be used. The demonstration mail server is the popular sendmail alternative Postfix, authored by Wietse Venema, supplemented with Lutz Jänicke's TLS capability patches. Both FreeBSD and Postfix are Open Source software projects and may be downloaded for free from their respective websites.

In addition, a valid digital certificate and private key, along with the public key of the issuing certificate authority (CA) must be present on the server. The OpenSSL framework, which is included with FreeBSD, contains the core libraries for SSL and TLS communications and supplies utilities for generating and managing digital certificates. The procedures for requesting a digital certificate from a commercial CA such as VeriSign, or generating a self-signed certificate, lie beyond the purview of this paper. (Further information on using OpenSSL to create and manage certificates is available at the OpenSSL website.)

Step 1 - Install Postfix with TLS Support

Installation of Postfix under FreeBSD is extremely easy, thanks to the ports system provided by that operating system[2]. Simply log in as root user, change to the /usr/ports/mail/postfix

directory, and run the command "make install replace" to automatically download the Postfix source, verify its integrity, build the mail server, and install it as the active mail server. During installation of the port a menu of optional modules is provided. Selecting "TLS (Secure Sockets Layer and Transport Layer Security)" from this menu ensures that TLS support is enabled in the resulting server.

Step 2 – Install Digital Certificates

The files containing the server's digital certificate and private key, as well as the issuing CA's public key, must be made available to Postfix. Make sure that the private key is readable only by the Postfix system account. Placed in the wrong hands, the same files used to verify your server's identity may be used to impersonate it[5]! In this walk through, the digital certificate will be stored in /etc/ssl/cert.pem, the private key in /etc/ssl/private/key.pem, and the issuing CA certificate in /etc/ssl/cacert.pem.

Step 3 – Edit the Postfix Configuration Files

You may now configure Postfix by editing the main.cf file. The default installation includes several very useful sample files and they may be consulted for detailed information about particular settings[8]. Please note that separate settings are used to configure TLS in its role as a mail server (receiving mail from other clients) and as a mail client (sending mail on users' behalf to remote servers)[4].

The client side of TLS communications is configured using the following parameters:

Name	Purpose	Setting
smtp_tls_CAfile	Sets the path to the CA certificate.	/etc/ssl/cacert.pem
smtp_tls_cert_file	Sets the path to the TLS certificate.	/etc/ssl/cert.pem
smtp_tls_key_file	Sets the path to the TLS private key.	/etc/ssl/private/key.pem
smtp_tls_loglevel	Sets the level of detail to log.	1
smtp_use_tls	Globally enables/disables TLS.	Yes

The parameters for configuring the server side of TLS communications are nearly identical:

Name	Purpose	Setting
smtpd_tls_CAfile	Sets the path to the CA Certificate.	/etc/ssl/cacert.pem
smtpd_tls_cert_file	Sets the path to the TLS certificate.	/etc/ssl/cert.pem
smtpd_tls_ask_ccert	Ask for client certificate.	Yes

smtpd_tls_key_file	Sets the path to the TLS private key.	/etc/ssl/private/key.pem
smptd_tls_loglevel	Sets the level of detail to log.	1
smptd_use_tls	Globally enables/disables TLS.	Yes

Step 4 – Verify TLS operation

At this point, your Postfix installation should be configured to use TLS for both incoming and outgoing mail. If you consult the Postfix mail log (located at /var/log/maillog if you are using FreeBSD), you will find exchanges such as the following (IP addresses obfuscated):

```
Aug  5 00:46:24 bart postfix/smtpd[89959]: connect from
foreign.mailserver.net[x.x.x.x]

Aug  5 00:46:25 bart postfix/smtpd[89959]: setting up TLS connection
from foreign.mailserver.net[x.x.x.x]

Aug  5 00:46:25 bart postfix/smtpd[89959]: TLS connection established
from foreign.mailserver.net[x.x.x.x]: TLSv1 with cipher EDH-RSA-DES-
CBC3-SHA (168/168 bits)

Aug  5 00:46:26 bart postfix/smtpd[89959]: 1432B2E7:
client=foreign.mailserver.net[x.x.x.x]

Aug  5 00:46:29 bart postfix/cleanup[89960]: 1432B2E7: message-
id=<20010805092701.A7766@mailserver.net>

Aug  5 00:46:29 bart postfix/qmgr[64694]: 1432B2E7:
from=<homer@mailserver.net>, size=3543, nrcpt=1 (queue active)

Aug  5 00:46:29 bart postfix/smtpd[89959]: disconnect from
foreign.mailserver.net[x.x.x.x]

Aug  5 00:46:29 bart postfix/local[89963]: 1432B2E7:
to=<maggie@simpson.com>, relay=local, delay=3, status=sent
("|/usr/local/bin/procmail")
```

Here we see that the mail server known as “foreign.mailserver.net” has connected to our host (“bart.simpson.com”), and then negotiated a TLS connection using RSA authentication and 3DES encryption[6]. One email was transmitted to our mail server over this secure connection. The secure tunnel was subsequently torn down and the mail delivered to the local “maggie@simpson.com” address. Success!

But wait a moment. What about outgoing mail? An outbound message to a fellow TLS-enabled host generates the following log entries:

```
Aug  5 12:22:35 bart postfix/pickup[29045]: 103412F3: uid=1001
from=<maggie>

Aug  5 12:22:35 bart postfix/cleanup[30066]: 103412F3: message-
id=<20010805122234.B399@bart.simpson.com>

Aug  5 12:22:35 bart postfix/qmgr[18016]: 103412F3:
from=<maggie@simpson.com>, size=445, nrcpt=1 (queue active)
```

```
Aug  5 12:22:36 bart postfix/smtp[30067]: setting up TLS connection to
bart.simpson.com

Aug  5 12:22:36 bart postfix/smtp[30067]: verify error:num=19:self
signed certificate in certificate chain

Aug  5 12:22:37 bart postfix/smtp[30067]: Unverified:
subject_CN=bart.simpson.com, issuer_CN=bart.simpson.com

Aug  5 12:22:37 bart postfix/smtp[30067]: TLS connection established to
bart.simpson.com: TLSv1 with cipher EDH-RSA-DES-CBC3-SHA (168/168 bits)

Aug  5 12:22:37 bart postfix/smtp[30067]: Peer certficate could not be
verified

Aug  5 12:22:37 bart postfix/smtp[30067]: 103412F3:
to=<homer@mailserver.net>, relay=foreign.mailserver.net[192.168.0.6],
delay=2, status=sent (250 Ok: queued as 97E874C1B7)
```

Here our local mail server reports that it has accepted an email from Maggie's mail client ("bart.simpson.com"). It connected to the remote system "foreign.mailserver.net" and successfully upgraded its connection to TLS, again using RSA and 3DES encryption. Our email to "homer@mailserver.net" was then delivered and our connection torn down. We have now verified bi-directional TLS capabilities!

Beyond Encryption: Other Uses for TLS

The same PKI infrastructure that many corporations are implementing in support of their e-commerce sites and transactions provides new opportunities for improving the security and efficiency of e-mail transmissions. Aside from securing the exchange of email, TLS may also be used to verify whether or not a user is authorized to pass email through a server. Mobile users who need to send mail via a corporate mail server may especially benefit from this capability. Because these users may transmit from any location, a means of identifying valid users must be in place so that legitimate mail may flow while unauthorized traffic is blocked. Password mechanisms are one way of achieving this result, but they have significant drawbacks, including, for example, the security risks posed by clear-text transmission of key codes or the difficulty of utilizing often complex proprietary authentication methods.

Mobile users need only be issued a client certificate that validates their identity and is exchanged during TLS negotiations with the mail server. The mail server then knows that the client attempting to send mail is a legitimate user, and the sending mail client knows that the mail server is actually the server it is attempting to communicate with.

The most common implementation of this scheme involves a pre-existing CA structure. (The reader is referred to the sample-tls.cf Postfix configuration file for further information on the configuration required.)

Limitations of TLS

No standard yet been adopted for informing end users that TLS-transport was used in the course of delivering an email. Most current Mail Transport Agents (MTAs) modify the "Received" headers of an email to show that TLS was used, but this information is not generally displayed to users unless they take the trouble to view the headers in full[7].

Despite its benefits, the use of TLS to secure SMTP transactions has not been widely deployed by mail servers. As a result, a TLS-enabled site may find that, even though it is properly set up to communicate via TLS-encrypted channels, few of its emails are being secured because many peer mail servers are not TLS-capable. As more attention is devoted to email security, a rise in deployment of SMTP over TLS may be expected, but a global deployment is likely some ways into the future.

In addition, because several layers of indirection are likely to exist between the sender and the recipient of any given mail message, a point-to-point security model like TLS cannot be relied upon to provide complete security. Any of the intermediary transit points an email may go through are potentially hostile. It is not uncommon for a mail message to be routed through three, four, or even more such relay mail hosts on its way to its final destination. As a mail sender has no control over the transit path of an email, and as it is also not possible to force intermediate relays to use TLS, mail flowing between two TLS-enabled servers may not be encrypted end-to-end due to unsecured relay servers.

Summary

TLS is not a magic bullet for securing email. While TLS can play an effective role in a total email security system by adding security to every email sent to or from a secured server, this can only be one part of a total security approach. TLS can protect against passive attacks, such as a packet sniffer on a transit network, and it is an extremely effective method for authenticating mobile mail users. There still remain vulnerabilities in other components of the mail architecture, such as the mail servers themselves, relay mail servers, and DNS servers. By taking a defense-in-depth design approach to email security, TLS can be a very effective enhancement to your efforts to keep email safe, secure, and private.

References

1. Dierks, T. and C. Allen. "RFC 2246 - The TLS Protocol" URL: http://sunsite.dk/RFC/rfc/rfc2246.html (20 Aug. 2001)

2. FreeBSD.org "FreeBSD Ports" URL: http://www.freebsd.org/ports (21 Aug. 2001)

3. Hoffman, P. "RFC 2487 - SMTP Service Extension for Secure SMTP over TLS" URL: http://sunsite.dk/RFC/rfc/rfc2487.html (20 Aug. 2001)

4. Jänicke, Lutz. "Postfix/TLS - Configuring main.cf and master.cf" URL: http://www.aet.tu-cottbus.de/personen/jaenicke/postfix_tls/doc/conf.html (20 Aug. 2001)

5. Netscape Communications. "How SSL Works" URL: http://developer.netscape.com/tech/security/ssl/howitworks.html (15 Aug. 2001)

6. OpenSSL.org. "OpenSSL: Documents, ciphers(1)" URL: http://www.openssl.org/docs/apps/ciphers.html (21 Aug. 2001)

7. Recorla, Eric. SSL and TLS: Designing and Building Secure Systems. Addison-Wesley, 2001.

8. Venema, Wietse. "Postfix Configuration - Basics" URL: http://www.postfix.org/basic.html (19 Aug. 2001)

Last Updated: October 20th, 2018

Upcoming SANS Training

Click here to view a list of all SANS Courses

SANS Houston 2018	Houston, TXUS	Oct 29, 2018 - Nov 03, 2018	Live Event
SANS Gulf Region 2018	Dubai, AE	Nov 03, 2018 - Nov 15, 2018	Live Event
SANS Sydney 2018	Sydney, AU	Nov 05, 2018 - Nov 17, 2018	Live Event
SANS DFIRCON Miami 2018	Miami, FLUS	Nov 05, 2018 - Nov 10, 2018	Live Event
SANS London November 2018	London, GB	Nov 05, 2018 - Nov 10, 2018	Live Event
SANS Dallas Fall 2018	Dallas, TXUS	Nov 05, 2018 - Nov 10, 2018	Live Event
Pen Test HackFest Summit & Training 2018	Bethesda, MDUS	Nov 12, 2018 - Nov 19, 2018	Live Event
SANS Mumbai 2018	Mumbai, IN	Nov 12, 2018 - Nov 17, 2018	Live Event
SANS Rome 2018	Rome, IT	Nov 12, 2018 - Nov 17, 2018	Live Event
SANS Osaka 2018	Osaka, JP	Nov 12, 2018 - Nov 17, 2018	Live Event
SANS San Diego Fall 2018	San Diego, CAUS	Nov 12, 2018 - Nov 17, 2018	Live Event
SANS November Singapore 2018	Singapore, SG	Nov 19, 2018 - Nov 24, 2018	Live Event
SANS ICS410 Perth 2018	Perth, AU	Nov 19, 2018 - Nov 23, 2018	Live Event
SANS Paris November 2018	Paris, FR	Nov 19, 2018 - Nov 24, 2018	Live Event
SANS Stockholm 2018	Stockholm, SE	Nov 26, 2018 - Dec 01, 2018	Live Event
SANS Austin 2018	Austin, TXUS	Nov 26, 2018 - Dec 01, 2018	Live Event
SANS San Francisco Fall 2018	San Francisco, CAUS	Nov 26, 2018 - Dec 01, 2018	Live Event
European Security Awareness Summit 2018	London, GB	Nov 26, 2018 - Nov 29, 2018	Live Event
SANS Khobar 2018	Khobar, SA	Dec 01, 2018 - Dec 06, 2018	Live Event
SANS Dublin 2018	Dublin, IE	Dec 03, 2018 - Dec 08, 2018	Live Event
SANS Santa Monica 2018	Santa Monica, CAUS	Dec 03, 2018 - Dec 08, 2018	Live Event
SANS Nashville 2018	Nashville, TNUS	Dec 03, 2018 - Dec 08, 2018	Live Event
Tactical Detection & Data Analytics Summit & Training 2018	Scottsdale, AZUS	Dec 04, 2018 - Dec 11, 2018	Live Event
SANS Frankfurt 2018	Frankfurt, DE	Dec 10, 2018 - Dec 15, 2018	Live Event
SANS Cyber Defense Initiative 2018	Washington, DCUS	Dec 11, 2018 - Dec 18, 2018	Live Event
SANS Bangalore January 2019	Bangalore, IN	Jan 07, 2019 - Jan 19, 2019	Live Event
SANS Sonoma 2019	Santa Rosa, CAUS	Jan 14, 2019 - Jan 19, 2019	Live Event
SANS Amsterdam January 2019	Amsterdam, NL	Jan 14, 2019 - Jan 19, 2019	Live Event
SANS Threat Hunting London 2019	London, GB	Jan 14, 2019 - Jan 19, 2019	Live Event
Secure DevOps Summit & Training 2018	OnlineCOUS	Oct 22, 2018 - Oct 29, 2018	Live Event
SANS OnDemand	Books & MP3s OnlyUS	Anytime	Self Paced