CSE508 Network Security

4/11/2016 Email

Michalis Polychronakis
Stony Brook University
Email Overview

**MUA:** Mail User Agent
- Thunderbird, webmail, Pine, ...

**MSA:** Mail Submission Agent
- SMTP (port 587)
- Often same as initial MTA

**MTA:** Mail Transfer Agent
- SMTP (port 25)

**MDA:** Mail Delivery Agent
- IMAP (port 143), POP3 (port 110), local, ...

Typical flow: MUA → MSA → MTA → … → MTA → MDA → MUA
SMTP Transport Example

S: 220 smtp.example.com ESMTP Postfix
C: HELO relay.example.org
S: 250 Hello relay.example.org, I am glad to meet you
C: MAIL FROM:<bob@example.org>
S: 250 Ok
C: RCPT TO:<alice@example.com>
S: 250 Ok
C: RCPT TO:<theboss@example.com>
S: 250 Ok
C: DATA
S: 354 End data with <CR><LF>.<CR><LF>
C: From: "Bob Example" <bob@example.org>
C: To: "Alice Example" <alice@example.com>
C: Cc: theboss@example.com
C: Date: Tue, 15 January 2008 16:02:43 -0500
C: Subject: Test message
C:
C: Hello Alice.
C: This is a test message with 5 header fields and 4 lines in the message body.
C: Your friend,
C: Bob
C: 
S: 250 Ok: queued as 12345
C: QUIT
S: 221 Bye
Email/Messaging Security and Privacy Goals

Protect message content

Verify communicating parties’ identities

Fight spam
  (subject of future lecture)

Fight phishing
  (subject of future lecture)

Hide communication patterns
  (subject of future lecture)
Who can read my email?

Adversaries with local or remote access to my devices

Intruders, spouse, administrator, …

Malware, stolen credentials, physical access, …
Who can read my email?

Adversaries with local or remote access to MTAs and other intermediary servers

Intruders, administrators, other insiders, LEAs, …
Who can read my email?

*Adversaries with access to any intermediate network*

Intruders, administrators, other insiders, LEAs, …

Passive eavesdropping, MitM, DNS poisoning, …
Confidentiality Threats Recap:

Stored messages

*Compromised system (either local or remote)*
Malware, intruder, insider, stolen/lost device, …

*Compromised authentication*
Password theft, brute-force phone pin, …

Messages in transit

Eavesdropping and interception

Displayed messages

Screendump, reflections, shoulder surfing, …
Securing Email Transit

These days encryption is mandatory for client-to-server email transmission and retrieval

MUA ➔ MSA: STARTTLS (port 587/25), SMTPS (port 465)
MDA ➔ MUA: POP3S (port 995), IMAPS (port 993)

mikepo@capcom:~> nc smtp.gmail.com 25
220 mx.google.com ESMTP i185sm2356739qhc.49 - gsmtp
HELO foo.example.com
250 mx.google.com at your service
MAIL FROM:<mikepo@example.com>
530 5.7.0 Must issue a STARTTLS command first.

MTA ➔ MTA relaying: Another story...
STARTTLS: Opportunistic Encryption

Many legacy MTAs still do not support TLS
  Fail-open design is necessary

MTAs do their best to deliver messages
  A recipient MTA might present a self-signed certificate (common in antispam and email AV systems)
  There is no PKI for email...

MitM is trivially easy
  STARTTLS command is sent over a plaintext channel (!)
  Analogous to SSL stripping, but in this case the client has no indication that downgrade has happened
  Just assumes that the receiving MTA does not support TLS

Message interception is still possible
  Better than nothing: bulk passive eavesdropping not possible
I want to STARTTLS

mikepo@capcom:~> nc aspmx.1.google.com 25
220 mx.google.com ESMTP h126si17458667qhh.29 - gsmtph0
EHLO foo.example.com
250-mx.google.com at your service, [128.59.23.41]
250-SIZE 157286400
250-8BITMIME
250-STARTTLS
250-ENHANCEDSTATUSCODES
250-PIPELINING
250-CHUNKING
250 SMTPUTF8
STARTTLS
220 2.0.0 Ready to start TLS
<TLS Handshake>
I want to STARTTLS

mikepo@capcom:~> nc aspmx.1.google.com 25
220 mx.google.com ESMTP h126si17458667qhh.29 - gsmtp
EHLO foo.example.com
250-mx.google.com at your service, [128.59.23.41]
250-SIZE 157286400
250-8BITMIME
250-STARTTLS
250-ENHANCEDSTATUSCODES
250-PIPELINING
250-CHUNKING
250 SMTPUTF8
STARTTLS
220 2.0.0 Ready to start TLS
<TLS Handshake>
Facebook STARTTLS Study: May 2014

~60% of all messages sent via encrypted connection, but only ~30% pass strict validation (mostly due to self-signed certs)
Facebook STARTTLS Study: August 2014

~95% of outgoing messages encrypted with PFS and strict cert validation
Mostly due to changes by big recipient networks (Microsoft, Yahoo)
How much email was encrypted in transit?

Generally speaking, use of encryption in transit increases over time, as more providers enable and maintain their support. Factors such as varying volumes of email may explain other fluctuations.

**Outbound**

- **79%**
  - Messages from Gmail to other providers.

**Inbound**

- **56%**
  - Messages from other providers to Gmail.

Download data
How much email was encrypted in transit?

Generally speaking, use of encryption in transit increases over time, as more providers enable and maintain their support. Factors such as varying volumes of email may explain other fluctuations.

**Outbound**

![84%](image)

Messages from Gmail to other providers.

![Graph](image)

**Inbound**

![73%](image)

Messages from other providers to Gmail.

![Graph](image)

Download data
A tiny GUI change prompted many networks to deploy STARTTLS
Google, Yahoo SMTP email severs hit in Thailand

Staff writer  |  September 12, 2014  |  telecomasia.net

Internet users in Thailand have been hit by a massive man-in-the-middle attack aimed grabbing email login credentials from fake SMTP servers.

The attack has been verified on Google’s and Yahoo’s email servers and on two of the country’s largest fixed-line ISPs, though preliminary analysis suggest that all SMTP servers are targeted.

The STRIPTLS attack as it has become known works by inserting a man-in-the-middle at the ISPs. This is done via a transparent proxy.
On SMTP, STARTTLS and the Cisco ASA

During the course of trying to increase the security of my e-mail while in transit, I was working on enabling TLS in Postfix to opportunistically encrypt connections to SMTP servers. While verifying my configuration, I ran into an interesting issue.

In order to test my configuration out I was sending e-mails to a Gmail address via Postfix, unfortunately I wasn't seeing any logging in Postfix indicating that TLS was being used. So I attempted to investigate whether STARTTLS was actually being advertised by manually connecting to Google's SMTP servers using telnet:

telnet aspmx.1.google.com 25
Trying 2607:f8b0:4001:c02::1a...
Connected to aspmx.1.google.com.
Escape character is '^]'.
220 ******************************************
EHLO example.com
250-mx.google.com at your service,
250-SIZE 35882577
250-8BITMIME
250-XXXXXXX
250 ENHANCEDSTATUSCODES

Every server I connected to in Google's MX record was not advertising STARTTLS. On a whim, I attempted to connect to Google's SMTP servers from an entirely different network:

telnet 173.194.68.26 25
Trying 173.194.68.26...
Connected to qa-in-f26.1e100.net (173.194.68.26).
Escape character is '^]'.
220 mx.google.com ESMTP l3si4081429qct.164
EHLO stomp.colorado.edu
250-mx.google.com at your service, 1
250-SIZE 35882577
250-8BITMIME
250-STARTTLS
250 ENHANCEDSTATUSCODES
DNS Hijacking

STARTTLS stripping is not the only message interception way

DNS MX record poisoning: spoofed MX response

Compromised name server, MotS, …

Messages are diverted through the attacker’s mail server

**DANE** (DNS-based Authentication of Named Entities)

Allow X.509 certs to be bound to DNS names through DNSSEC

Trust anchor assertions: domain operator can securely convey information about which certificate authority should be trusted

Certificate pinning for SMTP: ongoing effort
End-to-End Email Encryption

Two major standards: **PGP** and **S/MIME**

- Similar, but incompatible
- Both rely on public-key cryptography
- Both support signing and/or encryption

**Main difference: how certificates are signed**

Typical workflow

- Encrypt message with a random symmetric key
- Encrypt symmetric key with the public key(s) of recipient(s)
- Digitally sign a hash of the message

**Metadata still in the clear!**

- Email headers
- Appended “Received:” records
- Subject line
Pretty Good Privacy

De fact standard for secure email

PGP (Phil Zimmermann) -> OpenPGP (RFC 4880)
  Gnu Privacy Guard (GPG): GPL implementation

Authentication
  Senders attach their digital signature to the message
  Receivers verify the signature using public-key cryptography

Confidentiality
  Symmetric key encryption
  Random session key generated for each message
  Session key is encrypted with recipient’s public key

Both are typically used on the same message
PGP Encryption

Use a different keypair for signing and encryption

http://www.slideshare.net/rvenkatesh25/network-security-primer
PGP Signed Message Example

From: alice@wonderland.com  
Date: Mon, 16 Nov 1998 19:03:30 -0600  
Subject: Message signed with PGP  
MIME-Version: 1.0  
Content-Type: text/plain; charset=US-ASCII  
Content-Transfer-Encoding: 7bit  
Content-Description: "cc:Mail Note Part"

-----BEGIN PGP SIGNED MESSAGE-----

Bob,

This is a message signed with PGP, so you can see how much overhead PGP signatures introduce. Compare this with a similar message signed with S/MIME.

Alice

-----BEGIN PGP SIGNATURE-----

Version: PGP for Personal Privacy 5.0  
Charset: noconv

iQCVAwUBM+oTwFcsAarXHFeRAQEsJgP/X3noON57U/6XYygOFjSY5lTpvAduPZ8M/aIFalUkCNuLLGxmtsbwRiDWLtCeWG3k+7zXDfx4YxuUcofGJn0QaTk8b3nxADL0O/EIvC/k8zJ6aGaPLB7rTIizamG0t5n6/08rPwwVkB03tmT8UNMAUCgoM02d6HXrKvnc2aBPFI=
=muAH

-----END PGP SIGNATURE-----

http://www.slideshare.net/rvenkatesh25/network-security-primer
PGP Additional Features

Compression

Sign -> Compress -> Encrypt
Compression after encryption is pointless (no redundancy)
Signature does not depend on the compression algorithm

Email Compatibility

Ciphertext contains arbitrary 8-bit octets
Some email systems may interpret some of them as control commands
Solution: base64 encoding (33% overhead)

Segmentation

Transparent message segmentation and reassembly for very large messages
Segments mailed separately
Encrypted Email: Two Main Challenges

Public key authenticity

Assurance that a public key is correct and belongs to the person or entity claimed

Has not been tampered with or replaced by an attacker

Public key discovery

How can we find the public key of a person/entity?

Especially the very first time we contact them
PGP: Web of Trust

Decentralized trust model

- In contrast to the centralized hierarchical model of PKI
- Users create their own certificates

Users validate other users’ certificates, forming a “web of trust”

- No trusted authorities: trust is established through friends
- Adjustable “skepticism” parameters: # fully and # partially trusted endorsers required to trust a new certificate (1 and 3 for GnuPG)
- Key signing parties

Main problems

- Privacy issues: social graph metadata
- Bootstrapping: new users are not readily trusted by others
- When opinions vary, “stronger set” wins: impersonation through collusion/compromised keys
- Scalability: WoT for the whole world?
Hey, I just got home from the party.

The one with the IRC folks?

Yeah.

How was it?

GOT TOO DRUNK.
I SCREWED UP, BAD.

WHAT HAPPENED?

There was a girl.
No idea who she was.
Don't even know her name.
I was too drunk to care.

And what, you slept with her?

No.
I signed her public key.

Shit, man.
S/MIME

Based on standard X.509 certificates
  Analogous operation to SSL: trusted CA sign certificates
  Traditional PKI

Uses multipart MIME to include cryptographic information in the message

Widely supported by most email readers (e.g., iOS)

Works well within corporations
  Certificate distribution through Active Directory infrastructure
S/MIME Signed Message Example

From: alice@wonderland.com
Date: Mon, 16 Nov 1998 19:03:08 -0600
Subject: Message signed with S/MIME
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="simple boundary"

--simple boundary
Content-Type: text/plain; charset=US-ASCII
Content-Transfer-Encoding: 7bit
Content-Description: "cc:Mail Note Part"

Bob,

This is a message signed with S/MIME, so you can see how much overhead S/MIME signatures introduce. Compare this with a similar message signed with PGP.

Alice

--simple boundary
Content-Type: application/octet-stream; name="smime.p7s"
Content-Transfer-Encoding: base64
Content-Disposion: attachment; filename="smime.p7s"

MIIQQwYJKoZIhvcNAQcCoIiQNDCCEDACAQExCzAJBgUrDgMCGgUAMAsGCSqGSIb3DQEHAaCCDnww
ggnGMIJ1L6ADAgECAhBQQR9a+DX0FXfQO4VHqhPMAQGCSqGSIb3DQEBBAUAMFIxETAPBgNVBAcT
CEludGVybmV0MRcwFQYDVQQKEw5WZXJpU2lnbiwgSW5jLjE0MDIGA1UECwMrVmvyaVnpZ24gQ2xh
c3MGMSBDBQSAIIEluZGl2aWR1YWwgU3Vic2NyaWJlcjAeFw05NzAzMjc0MDAwMDBaFw05ODAxMjcy
MzU5NTIaMIIBFezERMA8GA1UEBxMIcW50ZXJuZXQxFzAVBgNVBAsTD1Zlc1TaWduLCBJbmMuMTQw
MgYDVQQLEytWZXJpU2lnbiBDdBGFZcyAxIENBIC0gSW5kaXZpZHVyCCTdWJzY3JpYmVYmUYwRAYD

http://www.slideshare.net/rvenkatesh25/network-security-primer
Finding Public Keys

Public PGP key servers
  pgp.mit.edu
  keyserver.pgp.com

Cache certificates from received emails

Integration with user management (LDAP)

Ad-hoc approaches
  List public key on home page
  Print on business card
  Exchange through another medium on a case by case basis

Association with social profiles/identities
  keybase.io
MIT PGP Public Key Server

Help: Extracting keys / Submitting keys / Email interface / About this server / FAQ
Related Info: Information about PGP

Extract a key

Search String:   Do the search!

Index:  Verbose Index:  

☐ Show PGP fingerprints for keys
☐ Only return exact matches

Submit a key

Enter ASCII-armored PGP key here:
Michalis Polychronakis

keybase.io/mikepo

8EBD 8F30 8899 8AFF
polychronakis • tweet
polychronakis • gist

mikepo has an invitation available
If you know mikepo, you can ask them for an invitation to Keybase.

Encrypt

keybase join  # if you're new, or
keybase login  # if you're not.

# then
keybase push  # if you already have a public key, or
keybase gen  # if this is all new to you

mikepo from the command line

mikepo

Tracking (6)
hargikas
mstamat

gianluca_string

Trackers (6)
hargikas
kontaxis
mstamat
Biggest Issue: Usability

Non-trivial setup
   S/MIME: complex certificate enrollment process
   PGP: user is responsible for everything

Key management

Key revocation

Public key fingerprints

Poor mail client integration
   Can lead to catastrophic failures: e.g., Enigmail+Thunderbird silent encryption failure

(Let alone key discovery and trustworthiness issues)
Enigmail 1.7 is completely broken for my purposes.

Steps to reproduce the problem:

1) Write an email in TB.
2) Ensure "Force encryption" in Enigmail.
3) Ensure "Force signing" in Enigmail.
4) Recheck encryption and signing settings... OK.
5) Send the email.
6) Look at the received email. OOPS. It is NOT signed and NOT encrypted.

Sorry to say this so directly, but an encryption system, which CONFIRMS to the user in it's graphical user interface on two different places that it will encrypt AND THEN SENDS THE EMAIL WITHOUT ANY ENCRYPTION IN Plain Text ... is just the BIGGEST IMAGINABLE CATASTROPHE.

Sorry for my profane language but there is simply no excuse for such
Runa A. Sandvik
@runasand

Swedish media org @Aftonbladet publishes its GPG private key for a second time (first time was in 2012):

Anders Nilsson @nilssonanders
Sweden's biggest newspaper #Aftonbladet includes their private key in guide to PGP mail them (via @_zulln ) bit.ly/1FfHAOI

2:39 PM - 5 Mar 2015
End-to-End vs. Cloud-to-Cloud

IMAP: one of the oldest “cloud” services!
   Keep messages on the server
   Conveniently access them from multiple devices

Useful cloud-based email features
   Powerful search, collaborative SPAM filtering, …
   Need access to the plaintext! Gmail cannot index encrypted messages

Tradeoff: privacy vs. convenience
   Active research on searchable encryption
Encrypted Webmail?

Several recent efforts to transparently combine the convenience of webmail with PGP encryption

Is this really possible in a secure way?

JavaScript crypto is not a good idea

Secure JS code delivery?
Secure key storage?
Secure runtime (it’s a web browser?)?

Google end-to-end: implement crypto functionality within a browser extension

More control
Still not trivial
My Fellow Users,

I have been forced to make a difficult decision: to become complicit in crimes against the American people or walk away from nearly ten years of hard work by shutting down Lavabit. After significant soul searching, I have decided to suspend operations. I wish that I could legally share with you the events that led to my decision. I cannot. I feel you deserve to know what’s going on—the first amendment is supposed to guarantee me the freedom to speak out in situations like this. Unfortunately, Congress has passed laws that say otherwise. As things currently stand, I cannot share my experiences over the last six weeks, even though I have twice made the appropriate requests.

What’s going to happen now? We’ve already started preparing the paperwork needed to continue to fight for the Constitution in the Fourth Circuit Court of Appeals. A favorable decision would allow me resurrect Lavabit as an American company.

This experience has taught me one very important lesson: without congressional action or a strong judicial precedent, I would _strongly_ recommend against anyone trusting their private data to a company with physical ties to the United States.

Sincerely,
Ladar Levison
Owner and Operator, Lavabit LLC

Defending the constitution is expensive! Help us by donating to the Lavabit Legal Defense Fund [here](http://lavabit.com/).
Lavabit: “so secure that even our administrators can’t read your e-mail”

But they could, if they wanted to…

“Basically we generate public and private keys for the user and then encrypt the private key using a derivative of the plain text password. We then encrypt user messages using their public key before writing them to disk.”

“Because we need the plain text password to decrypt a user’s private key, we don’t support secure password authentication. We decided to support SSL instead (which encrypts everything; not just the password).”

http://highscalability.com/blog/2013/8/13/in-memoriam-lavabit-architecture-creating-a-scalable-email-s.html