Usable Email Cryptography

(End-to-End)

tcpqp mxamt qkxme mmdoi tbqsa xlgzv

Marcus Brinkmann
<mb@g10code.com>
LSM 2011-07-12

Introduction

Mail Encryption Fail

The Vision

Four Examples

The Zen Way of Implementation

Mail Encryption Fail

Why Johnny Can't Encrypt (Whitten, Tygar 1999):

User Tests	12 Participants	
Kept Message Secret	+++++++++	---
Encrypt	+++++++++++	-
… with correct key	+++++	-------
Key upload	++++++++++	--
Key download	++++++++	----
Raise Trust Issue	+++	---------
… and address it		------------
Backup Revocation Certificate		ZOMGLOL!!!1

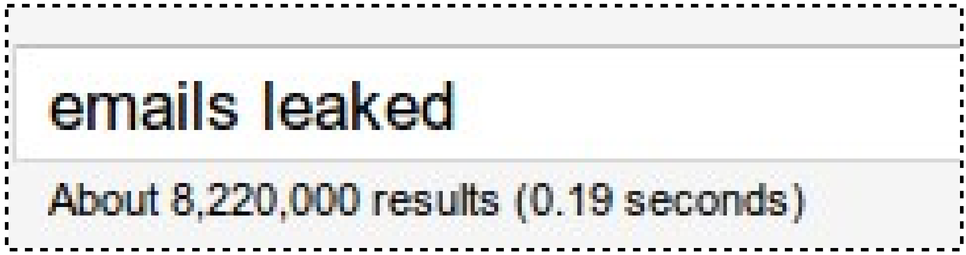
Mail Encryption Fail

People do not use mail encryption.

- ✗ too small to measure

Plaintext in the cloud.

- ✗ attack escalation


emails leaked

About 8,220,000 results (0.19 seconds)

Mail displaced by „PHP doodads“ (E. Moglen)

- ✗ 59% decline 12-17 yo (comcast)

Mail Encryption Fail

Organisations:

- ✗ X.509 (PKI with CA)

Server-side (not end-to-end):

- ✗ data retention
- ✗ provability of send and receive
- ✗ business models

The Vision

Suck less.

Four Examples

Opportunistic Encryption

Automatic Key Generation

Key Distribution through DNS

Trust On First Contact/Persistence of Pseudonym

Opportunistic Encryption

„Do you [the sender] want to encrypt this mail for this recipient and if yes, with which key?“

Sucks:

- Wrong person to ask.
- Wrong place to store preferences.
- Mistake leads to plaintext leak.

Sucks less:

- Get key and preferences from recipient.
- Always encrypt if possible.

Automatic Key Generation

„What key type, size, expiration time do you want, what is your name and mail address?“

Sucks:

- People choose inappropriate key parameter.
- Software-Amnesia.

Sucks less:

- Use best practices by default.
- Mail client knows name and mail address.
- No more stupid questions.

Automatic Key Distribution

„What keyserver do you want to use? How do you want to export your key? Which file do you want to import?“

Sucks:

- Keyservers disjoint and quality varies.
- No undo.
- What is exported?

Sucks less:

- Distribute keys through DNS (PKA).
- No search.
- Trust inheritance (DNSSEC).

TOFU/POP

„Does this key belong to that person?“

Sucks:

- Yes/no question with amnesia at critical time.
- What is the consequence of being wrong?
- What does it even mean?

Sucks less:

- Trust on first contact (like SSH)
- DNSSEC as CA
- Remember earlier contacts (persistence of pseudonym)
- „Trustiness“ mental model: „same key as last N times“

TOFU/POP

„Perspectives“ (Wendlandt et al., 2008):

- Network of monitors („notaries“) recording fingerprint histories in the network over time.
- Clients consult notary servers on trust decisions (first contact, fingerprint mismatch).

Notaries provide non-local majority vote over time, disabling many MITM attacks.

The Zen Way of Implementation

Reuse existing infrastructure:

- ✗ Full compatibility to OpenPGP and S/MIME.
- ✗ Full compatibility to other PKIs.
- ✗ PKA/CERT DNS available for many years now.
- ✗ TOFU/POP well-known from SSH.

The Zen Way of Implementation

Acceptance by modularity:

- ✗ Experts generate or publish keys manually, or use different trust policies.

Deeper integration can provide better user experience:

- ✗ Mail app has user name and account data.
- ✗ Mail app has semantic information on previous contacts.

The Zen Way of Implementation

The big challenge:

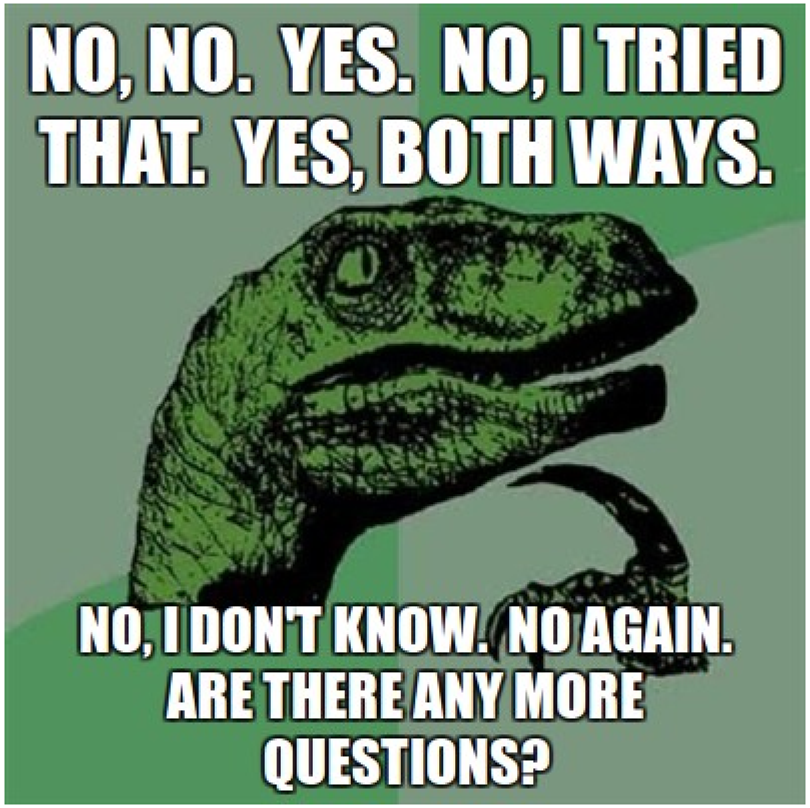
- ✗ Changed trust model requires new generation of user interfaces.
- ✗ Opportunistic encryption requires widespread adoption of PKA/CERT DNS.

The Zen Way of Implementation

Can we reach critical mass?

- Develop tool support and guidelines for user interaction.
- Engage privacy protection organisations.
- Shame providers into adapting their applications.

Thank you!


NO, NO. YES. NO, I TRIED THAT. YES, BOTH WAYS.
NO, I DON'T KNOW. NO AGAIN. ARE THERE ANY MORE QUESTIONS?