Prototyping a Lightweight Trust Architecture to Fight Phishing

David Chau
6 October 2005
Thanks to my collaborators ...

- Ben Adida
- Susan Hohenberger
- Ron Rivest
Outline

- The phishing problem
- Existing solutions
- SIBR and LTA
- The prototype
  - DNS, key server, e-mail client
  - Message processing, cryptographic primitives
- Future work
A Phishy E-mail

From: Support PayPal  
<do-not-reply@paypal.com>
Reply-To: do-not-reply@paypal.com  
To: ddcc@mit.edu  
Subject: Please Restore Your Account Access  
Date: Sun, 3 Apr 2005 15:57:28 -0400

Dear PayPal Member,

Recently there have been a large number of identity theft attempts targeting PayPal customers. In order to safeguard your account, we require that you confirm your PayPal details (Credit Card information and login/password for PayPal login, if you have).

This process is mandatory, and if not completed within the nearest time your account may be subject to temporary suspension.

To securely confirm you PayPal details please follow the link below:

https://www.paypal.com/cgi-bin/webscr?request=Reactivate

Thank you for prompt attention to this matter and thank you for using PayPal!

PayPal - Fraud Center  
1-800-PayPal.  
fraud_prevention@PayPal.com

Do not reply to this e-mail as it is an unmonitored alias.
The problem:

E-mail messages aren’t authenticated
Existing solutions

SPF/SenderID

$ dig -t txt mit.edu

;; ANSWER SECTION:
mit.edu. 60 IN TXT "v=spf1 ip4:18.7.7.0/24
ip4:18.7.21.0/24 ip4:18.72.0.0/16 ~all"
Existing solutions

DomainKeys

DomainKey-Signature: a=rsa-shal; q=dns; c=nofws; s=s1024; d=yahoo.com; h=Message-ID:Received: Date:From:Subject:To:Cc:MIME-Version:Content-Type:Content-Transfer-Encoding;
b=kt0N/9igWyJYRe8v5XDaQZuvvdJRHh9pXHPVHbZ1XzKaA7M61D7LgrmpFAukvGgWJ3P2LRGNTpYT37mMYPdWx3fJd4qWFXpPZQtIRa+WVGD5RhjI6YdPwnPoSg6CY9GieFL8EmuyQW0ELlg2f
pX4YgcyZU+pkub+ZSUhv7BiJ40=
Existing solutions

- PGP
- X.509
- … and more …
SIBR:

Separable …

Identity-based …

Ring signatures
Identity-based:

Your name (or e-mail address) is your public key
Separable

Different users can use different parameters
Ring signatures

One of us signed it, but you can’t prove it was me

Repudiable signatures!
Lightweight Trust Architecture
- Master public key on DNS server
- Secret keys e-mailed to users
- Sign with sender’s secret key and recipient’s public key
- Just secure enough for e-mail
The components

Things we had to build
The DNS server

- Master public key in DNS
  - Generated by administrator of a domain
  - Stored as TXT record in _lta subdomain
The key server

\{ \text{user’s e-mail address} \\
\text{expiration date} \} \rightarrow \text{master secret key} \rightarrow \text{user’s secret key}

(\text{identity-based public key})
The key server

- Web-based interface
- Secret key sent to user’s e-mail account
- Selectable expiration date critical for repudiability
- Prototype implemented in Python
E-mail client integration

- Easy to sign and verify messages
- Key management
- Prototype with Rmail (Emacs mail client)
Message processing

- Message canonicalization
- Signature covers message body and key headers
- Signature inserted as header
- For the future: more sophisticated handling, MIME
Many ways to implement identity-based signature schemes

- **Bohen-Franklin keys**
  - Bilinear maps
  - Unfortunately, patented

- **Guillou-Quisquater signatures**
  - Based on RSA
Cryptography

- We implemented both
- Signing and verification reasonably fast
- Keys are short enough to fit inside 512-byte DNS reply
What we accomplished

- Demonstrated *complete* prototype of an LTA system:
  - DNS server that servers master public keys
  - Web-based key server that sends secret keys to users on demand
  - E-mail client that can sign and verify messages
    - Performs DNS key lookups on the fly
    - Imports and uses secret keys from the key server
Where to go from here …

- Deployment issues
  - LTA is easier than traditional public-key infrastructure
- More complex usage scenarios
  - For example: repudiable messages to mailing lists
- Implementation improvements
Thank you!