Lightweight Encryption for Email

Ben Adida
ben@mit.edu
7 July 2005

joint work with
Susan Hohenberger and Ronald L. Rivest
MIT Cryptography and Information Security Group
Motivation

• To Improve/Restore the Usefulness of Email

• Lightweight Trust for Email Signatures [ACHR2005]

• Can we get reasonable encryption from similar simplified key management?
Lightweight Signatures

• Makes forging email from bob@foo.com as difficult as receiving Bob’s email.
• No explicit user key management
• Uses only existing infrastructure
ID-Based Crypto

Alice

"bob@foo.com"

keyserver

MPK

MSK

PK_{bob}

SK_{bob}

Bob

PK

SK
ID-based Domains

- $MPK_{\text{wonderland.com}}$
- $MSK_{\text{wonderland.com}}$
- $SK_{\text{alice@wonderland.com}}$
- $SK_{\text{bob@foo.com}}$
- $MPK_{\text{foo.com}}$
- $MSK_{\text{foo.com}}$

Keyserver:
- wonderland.com
- foo.com
DNS to distribute Master Public Keys

**DNS**
- wonderland.com
  - MPK\textsubscript{wonderland.com}
- foo.com
  - MPK\textsubscript{foo.com}

**Publish**
- \texttt{MSK}_{\textit{wonderland.com}}

[DomainKeys]
Email-Based Authentication

[Gar2003]

$SK_{alice@wonderland.com}$

Alice

$MSK_{wonderland.com}$

wonderland.com

keyserver

$SK_{alice@wonderland.com}$

Alice

wonderland.com incoming mail server

Review
Lightweight Sigs

1. PUBLISH

From: Alice
To: Bob
Subject: Guess?
I heard that...
I'm serious!
Signed: Alice

DNS
wonderland.com
MPK_{wonderland}
foo.com
MPK_{foo}

2. PUBLISH

wonderland.com
key server

3. SK_{A}

Alice
Wonderland.com Network

4. MPK_{bank}

“alice@wonderland.com"

5. MPK_{bank}

Bob
foo.com Network

6.
For Encryption?

1. PUBLISH
2. wonderland.com key server
3. Alice
4. MPK_{wonderland}
5. MPK_{bank}
6. Bob

From: Alice
To: Bob
Subject: Guess?
I heard that...
I'm serious!
Signed: Alice

DNS
wonderland.com
foo.com
PUBLISH
MPK_{wonderland}
1 1
MPK_{wonderland}

From: Alice
To: Bob
Subject: Guess?
I heard that...
I'm serious!
Signed: Alice

DNS
foo.com
Network

PUBLISH
DNS
wonderland.com
foo.com
PUBLISH
MPK_{foo}

PUBLISH
DNS
wonderland.com
foo.com
PUBLISH
MPK_{bank}

alice@wonderland.com

"alice@wonderland.com"
Threat Model

• Assume your incoming mail server won’t actively spoof/attack you.

• **Signatures**
  If the MSK is compromised, simply change the MSK/MPK (DNS updates).

• **Encryption**
  Different story....
Threat #1: MSK compromise

- all past encrypted emails are immediately compromised.
- if the MSK compromise is discreet, then all future encrypted emails are also compromised. (hacking into a keys server).
Splitting Keys

Alice $SK_{Alice}$ wonderland.com

$MK_{wonderland,0}$

$MK_{wonderland,1}$

$MK_{wonderland,2}$

$MPK_{wonderland}$

$SK_{Alice}$ wonderland.com,0

$SK_{Alice}$ wonderland.com,1

$SK_{Alice}$ wonderland.com,2
Threat #2: Corrupt Mail Server

- A corrupt incoming mail server can decrypt and read all secret key material.
- A passive corrupt mail server can intercept all emails.
- Even MSK splitting doesn’t help.
Recombining Keys

- Bob generates a new MPK/MSK pair
- The combined SK matches the combined MPK.
- The combined MPK provides **certification** and **protection**.
- The second MPK component needs no certification!
Single Core Solution

- $ params$
  - $MSK_1$
  - $MPK_1$
  - $MSK_2$
  - $MPK_2$

- $SK_1$
- $SK_2$

- CombineSecretKey
- CombineMasterKey

- $SK_{combined}$
- $MPK_{combined}$

- $bob@foo.com$

- VerifySecretShare

- $SK_1$
- $MPK_1$
Building These Features on Boneh-Franklin and Waters Identity-Based Encryption
Bilinear Maps

$G_1, G_2$, both of prime order $q$

$e : G_1 \times G_1 \rightarrow G_2$

$g, h$ generate $G_1$

$Z = e(g, h)$ generates $G_2$

$e(g^a, h^b) = e(g, h)^{ab}$

$e(ug, h) = e(u, h)e(g, h)$
Boneh-Franklin Keys

Public Parameters: $G_1, G_2, q, g, H$

- $MSK = s \in \mathbb{Z}_q$
- $MPK = g^s \in G_1$
- $PK_{ID} = H(ID)$
- $SK_{ID} = H(ID)^s$
Splitting & Recombining Boneh-Franklin Keys

\[ MSK_1 = s_1 \quad MSK_2 = s_2 \]
\[ MPK_1 = g^{s_1} \quad MPK_2 = g^{s_2} \]
\[ SK_1 = H(ID)^{s_1} \quad SK_2 = H(ID)^{s_2} \]
\[ \text{CombineMasterKey} \quad MPK = MPK_1 \cdot MPK_2 = g^{s_1+s_2} \]
\[ \text{CombineSecretKey} \quad SK = SK_1 \cdot SK_2 = H(ID)^{s_1+s_2} \]
\[ \text{Effective } MSK = s_1 + s_2 \]
Waters Keys

Public Parameters: $G_1, G_2, q, g, h, F$

$$MSK = h^s$$
$$MPK = g^s$$
$$PK_{ID} = F(ID)$$
$$SK_{ID} = (h^s F(ID)^r, g^r)$$
Splitting & Recombining Waters Keys

\[
\begin{align*}
M SK_1 &= h^{s_1} \\
M PK_1 &= g^{s_1} \\
S K_1 &= (h^{s_1} F(ID)^{r_1}, g^{r_1})
\end{align*}
\]

\[
\begin{align*}
M SK_2 &= h^{s_2} \\
M PK_2 &= g^{s_2} \\
S K_2 &= (h^{s_2} F(ID)^{r_2}, g^{r_2})
\end{align*}
\]

CombineMasterKey

\[
M PK = M PK_1 \cdot M PK_2 = g^{s_1+s_2}
\]

CombineSecretKey

\[
S K = (h^{s_1} F(ID)^{r_1} \cdot h^{s_2} F(ID)^{r_2}, g^{r_1} \cdot g^{r_2})
\]

\[
= (h^{s_1+s_2} F(ID)^{r_1+r_2}, g^{r_1+r_2})
\]

Effective \( M SK = g^{s_1+s_2} \)
Additional Details

- **Malicious Share Generation:** NIZK Proof of Knowledge of MSK share

- **Malicious SK Distribution:** k-out-n shares using Lagrange coefficients [GJKR99]
Putting it All Together

1. $MPK_1^{foo.com}$
2. $MPK_2^{foo.com}$
3. $SK_{Bob,1}^{foo.com}$
4. $SK_{Bob,2}^{foo.com}$
5. $SK_{Bob}^{foo.com}$
6. $MPK_{Bob}$
7. $SK_{Bob}$

From: Alice
To: Bob
Subject: Secret
Alice’s Point of View

- **Finding Bob’s Public Key:**
  automatic: a lookup, a computation against MPK. No trust decision necessary.

- **Decryption Key Management:**
  automatic, just upgrade the mail client

- **Key Revocation, etc...:**
  automatic, with upgraded mail client

Automation!
Summary

• Lightweight key infrastructure is not enough for encryption

• To protect against MSK compromise: **key splitting**

• To protect against mail server compromise: **key recombination**

• Both can be accomplished with the same trick on Boneh-Franklin and Waters keys
Questions?
Backup Slides
Another Solution

Yahoo.com incoming mail server

SK_{Alice, yahoo.com}

Gmail.com incoming mail server

SK_{Alice, gmail.com}

Alice