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

"bob@foo.com"

keyserver

MPK

MSK

$PK_{bob}$

$SK_{bob}$

Alice

Bob
ID-based Domains

MPK_{wonderland.com}

MSK_{wonderland.com}

SK_{alice@wonderland.com}

Alice

MPK_{foo.com}

MSK_{foo.com}

SK_{bob@foo.com}

Bob
DNS to distribute Master Public Keys

Publish $MPK_{\text{wonderland.com}}$

DNS

$\text{wonderland.com}$
$MPK_{\text{wonderland.com}}$

$\text{foo.com}$
$MPK_{\text{foo.com}}$

[DomainKeys]
Email-Based Authentication

[Gar2003]
Lightweight Sigs

1. **PUBLISH**

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

2. **Alice**
   - Wonderland.com Network
   - SK_A
   - From: Alice
   - To: Bob
   - Subject: Guess?
   - Signed: Alice
   - I heard that...
   - I'm serious!

3. **Bob**
   - foo.com Network
   - MPK_{bank}
   - From: Alice
     - To: Bob
     - Subject: Guess?
     - Signed: Alice
     - I heard that...
     - I'm serious!

4. **“alice@wonderland.com”**

5. **PUBLISH**

6. **foo.com key server**
For Encryption?

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

2. Alice
   - Wonderland.com Network
   - SK_A

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

4. “alice@wonderland.com”

5. MPK_{bank}

6. Bob
   - foo.com Network
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 keyserver).
Splitting Keys

$MPK_{\text{wonderland}}$

$MPK_{\text{wonderland},0}$

$MSK_{\text{wonderland},0}$

$SK_{\text{Alice}}_{\text{wonderland.com},0}$

$MPK_{\text{wonderland},1}$

$MSK_{\text{wonderland},1}$

$SK_{\text{Alice}}_{\text{wonderland.com},1}$

$MPK_{\text{wonderland},2}$

$MSK_{\text{wonderland},2}$

$SK_{\text{Alice}}_{\text{wonderland.com},2}$

Alice $SK_{\text{Alice}}_{\text{wonderland.com}}$
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

M PK _1
M SK _1

M PK _2
M SK _2

SK _1
SK _2

CombineMasterKey

CombineSecretKey

VerifySecretShare

bob@foo.com

MPK _combined

SK _combined
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

\[ MSK_1 = s_1 \quad MSK_2 = s_2 \]
\[ MPK_1 = g^{s_1} \quad MPK_2 = g^{s_2} \]
\[ SK_1 = H(ID)^{s_1} \quad SK_2 = H(ID)^{s_2} \]

[BF2000]

\[ MPK = MPK_1 \cdot MPK_2 = g^{s_1+s_2} \]
\[ SK = SK_1 \cdot SK_2 = H(ID)^{s_1+s_2} \]

Effective \( MSK = s_1 + s_2 \)
Waters Keys

Public Parameters: \( G_1, G_2, q, g, h, F \)

\[
\begin{align*}
M SK &= h^s \\
M PK &= g^s \\
PK_{ID} &= F(ID) \\
SK_{ID} &= (h^s F(ID)^r, g^r)
\end{align*}
\]
Splitting & Recombining Waters Keys

\[
\begin{align*}
M SK_1 &= h^{s_1} \\
M PK_1 &= g^{s_1} \\
S K_1 &= (h^{s_1} F(ID)^{r_1}, g^{r_1}) \\
M SK_2 &= h^{s_2} \\
M PK_2 &= g^{s_2} \\
S K_2 &= (h^{s_2} F(ID)^{r_2}, g^{r_2}) \\
M PK &= M PK_1 \cdot M PK_2 = g^{s_1+s_2} \\
S K &= (h^{s_1} F(ID)^{r_1} \cdot h^{s_2} F(ID)^{r_2}, g^{r_1} \cdot g^{r_2}) \\
&= (h^{s_1+s_2} F(ID)^{r_1+r_2}, g^{r_1+r_2}) \\
\text{Effective } M SK &= g^{s_1+s_2}
\end{align*}
\]
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. CombineMasterKey
2. DNS
3. Lightweight Cert. Server
4. GenerateShare
5. CombineMasterKey
6. Encrypt
7. CombineSecretKey

- **MPK**
  - **MPK**
    - **MPK**
      - **MPK**
        - **MPK**
          - **MPK**

- **SK**
  - **SK**
    - **SK**
      - **SK**
        - **SK**

- **Bob**
- **Alice**

```
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

![Diagram showing two incoming mail servers, one for yahoo.com and one for gmail.com, each connected to Alice with a shared secret key SK_Alice@yahoo.com and SK_Alice@gmail.com.]