RE: Reliable Email

Michael Kaminsky (Intel Research Pittsburgh)
Scott Garrings (CMU)
Michael Freedman (NYU/Stanford)
Brad Karp (University College London)
David Mazières (Stanford)
Haifeng Yu (Intel Research Pittsburgh/CMU)
Motivation

• Spam is a huge problem today
  – More than 50% of email traffic is spam.
  – Large investment by users/IT organizations ($2.3b in 2003 on increased server capacity)

• But, more importantly…
Email is no longer reliable

• Users can't say what they want any more
  – Ex: Intel job offer goes to spam folder
  – Ex: Discussion about spam filtering

Goal:
Improve email's reliability
Outline

• Background / Related Work
• Design
  – Social networks and Attestations
  – Preserving Privacy
• Re: in Practice
• Evaluation
• Implementation
• Conclusion
Basic Terminology

• False Positives (FP)
  – *Legitimate email marked as spam*
  – Can lose important mail
  – Email less reliable

• False Negatives (FN)
  – *Spam marked as legitimate email*
  – Annoying and/or offensive
A Typical Spam Defense System

Incoming Mail → Whitelist System → Default Path → Rejection System → Default Path → Inbox

Reject → Spam
Related Work

• People use a variety of techniques
  – Content filters (SpamAssassin, Bayesian)
  – Payment/proof of work schemes
  – Sender verification
  – Blacklists
  – Human-based (collaborative) filtering
    – Whitelists

Idea:
Whitelist friends of friends

Re: is complementary to existing systems.
Traditional Whitelist Systems

Traditional WLs suffer from two problems:

1) Spammers can forge sender addresses
Traditional Whitelist Systems

Use anti-forgery mechanism to handle (1), similar to existing techniques.

Handle (2) with *social networks*

Traditional WLs suffer from two problems:

1) Spammers can forge sender addresses
2) Whitelists don’t help with strangers
Approach: Use Social Networks

- Bob whitelists people he trusts
- Bob *signs* attestation B→A
  - No one can forge attestations from Bob
  - Bob can share his attestations

**Attestation**: B→A

A is a *friend* of B

B trusts A not to send him spam
Approach: Use Social Networks

- What if sender & recipient are not friends?
  - Note that B→A and A→C
  - B trusts C because he's a **friend-of-friend** (FoF)
Find FoFs: Attestation Servers

Charlie (C) 

Charlie’s Attestation Server (AS)

A → C

Recipient (Bob) queries sender’s attestation server for mutual friends...

Bob (B)

Sharing attestations reveals your correspondents!

Note: no changes to SMTP, incremental deployment
Privacy Goals

- Email recipients never reveal their friends
- Email senders only reveal specific friends queried for by recipients
- Only users who have actually received mail from the sender can query the sender for attestations
Outline

• Background / Related Work
• Design
  – Social networks and Attestations
  – Preserving Privacy
• Re: in Practice
• Evaluation
• Implementation
• Conclusion
Cryptographic Private Matching

Sender (S)’s AS

**PM Evaluate**

encrypted mutual friends

friends

A → S
C → S
D → S
E → S

Recipient (R)

**PM Encrypt**

encrypted friends

friends

R → A
R → B
R → C

PM Decrypt

mutual friends

A → S
C → S
?

?
PM Details

• First implementation & use of PM protocol
• Based on our previous work [Freedman04]
• Attestations encoded in encrypted polynomial
• Uses Homomorphic Encryption
  – Ex: Paillier, ElGamal variant
  – $\text{enc}(m_1+m_2) = \text{enc}(m_1) \cdot \text{enc}(m_2)$
  – $\text{enc}(c \cdot m_1) = \text{enc}(m_1)^c$
Restricting FoF Queries

- Sender can use token to restrict FoF query
  - Users have a public/secret key pair
Restricting FoF Queries

- Sender can use token to restrict FoF query
  - Users have a public/secret key pair
- Recipient can use token to detect forgery
Outline

• Background / Related Work
• Design
  – Social networks and Attestations
  – Preserving Privacy
• Re: in Practice
• Evaluation
• Implementation
• Conclusion
Scenario 1: Valid Mail Rejected

Alice

Mail Client

“mortgage..."

Bob

Mail Server

Spam Assassin

Trash Can
Scenario 2: Direct Acceptance

Alice
- Mail Client
- Attestation Server
  - Token
  - Token OK

Bob
- Mail Server
  - Re:
  - auth. token
- Spam Assassin

Bob’s Friends
- Alice
- Hit!
- Tom

Mail Server flows to Mail Client via Attestation Server with 'mortgage...'.

Alice's 'Token OK' authenticates the transaction.
Scenario 3: FoF Acceptance

Bob

Mail Server

Mail Client

Attestation Server

Charle is a friend of
  * John
  * Alice

“mortgage...

token OK & E(?) E(Alice)

auth. token & FoF query

Mutual friend: Alice

Spam Assassin

Bob’s Friends
  * Alice
  * Tom

No Direct Hit
Outline

• Background / Related Work
• Design
  – Social networks and Attestations
  – Preserving Privacy
• Re: in Practice
• Evaluation
• Implementation
• Conclusion
Evaluation

• How often do content filters produce false positives?

• How many opportunities for FoF whitelisting beyond direct whitelisting?

• Would Re: eliminate actual false positives?
Trace Data

• For each message:
  – Sender and recipient (anonymized)
  – Spam or not as assessed by content-based spam filter

• Corporate trace
  – One month
  – 47 million messages total (58% spam)
False Positive Data

• Corporate mail server bounces spam
• Bounce allows sender to report FP
• Server admin validates reports and decides whether to whitelist sender
• We have a list of ~300 whitelisted senders
  – 2837 messages in trace from these senders that were marked as spam by content filter
  – These are almost certainly false positives
Opportunities for FoF Whitelisting

- FoF relationships help most when receiving mail from *strangers*.
- When user receives non-spam mail from a stranger, how often do they share a mutual correspondent?
  - 18% of mail from strangers
  - Only counts mutual correspondents in trace
- Opportunity: when correspondents = friends
Saved FPs: Ideal Experiment

- **Ideally**: run Re: & content filter side-by-side
  - Measure how many FPs avoided by Re:
Saved FPs: Trace-Driven Experiment

- We have an implementation, but unfortunately, no deployment yet
- No social network data for traces
  - Infer friendship from previous non-spam messages
- Recall that 2837 messages were from people who reported FPs
- How many of these would Re: whitelist?

Re: would have saved 87% of these FPs (71% direct, 16% FoF)
Implementation

• Prototype implementation in C++/libasync
  – Attestation Server
  – Private Matching (PM) implementation
  – Client & administrative utilities
  – 4500 LoC + XDR protocol description

• Integration
  – Mutt and Thunderbird mail clients
  – Mail Avenger SMTP server
  – Postfix mail client
Performance

• Direct attestations are cheap

• Friend-of-friend is somewhat slower
  – PM performance bottleneck is on sender’s AS
    • Ex: intersecting two 40-friend sets takes 2.8 sec versus 0.032 sec for the recipient
  – But…
    • Many messages accepted by direct attestation
    • Can be parallelized
    • Performance improvements possible
Nuances

• Audit Trails
  – Recipients always know why they accepted a message (e.g., the mutual friend)

• Mailing Lists
  – Attest to list
  – Rely on moderator to eliminate spam

• Profiles
  – Senders use only a subset of possible attestations when answering FoF queries
Conclusion

- Email is no longer reliable because of FPs

**Idea:**

Whitelist friends of friends

- Preserve privacy using PM protocol
- Opportunity for FoF whitelisting
- Re: could eliminate up to 87% of real FPs
- Acceptable performance cost
Backup Slides
Coverage Tradeoff

- Trusting a central authority can get you more coverage (DQE)
  - Ex: random grad student
Coverage Tradeoff

- Social relationships can help avoid the need to trust a central authority (Re:)
  - Ex: friends, colleagues
Forgery Protection

- Users have a public/secret key pair
- Sender attaches a *signed authentication token* to each outgoing email message
Forgery Protection

- Recipient asks sender's AS to verify token
  - Assume: man-in-the-middle attack is difficult
  - Advantage: Don't need key distribution/PKI
- Sender can use token to restrict FoF query
Revocation

• What if A’s key is lost or compromised?
• Two things are signed
  – Authentication tokens
  – Attestations

• Authentication tokens
  – User uploads new PK to AS
  – AS rejects tokens signed with the old key
Revocation: Attestations

- Local attestations
  - Delete local attestations (A→*)

- Remote attestations: expiration
  - If A gave A→B to B, Re: does not currently provide a way for A to tell B to delete the attestation
    - When A→B expires, B will stop using it for FoF
  - If C→A, C should stop trusting attestations signed by A’s old key
    - When C→A expires, C will re-fetch A’s public key
False Negatives

• Assumption: people will not attest to spammers
  – Therefore Re: does not have false negatives

• What if this assumption does not hold?
  – Remove offending attestations using audit trail
  – Attest without transitivity
    • A trusts B, but not B’s friends
  – Don’t share attestation with attestee
    • Ex: a mailing list
PM Protocol Details

Recipient (R)

$P(y) = (x_1 - y)(x_2 - y)\ldots(x_{k_R} - y) = \sum_{u=0}^{k_R} a_u y^u$

Each $x_i$ is one of R’s friends

R has $k_R$ friends

Canonica version of $P(y)$

R constructs the $P(y)$ so that each friend is a root of the polynomial
PM Protocol Details

Sender’s Attestation Server (AS)

Recipient (R)

\[ P(y) = (x_1 - y)(x_2 - y) \ldots (x_{k_R} - y) \]

\[ = \sum_{u=0}^{k_R} a_u y^u \]
**PM Protocol Details**

**Sender’s Attestation Server (AS)**

**Recipient (R)**

\[ P(y) = (x_1 - y)(x_2 - y)\ldots(x_{k_R} - y) \]

\[ = \sum_{u=0}^{k_R} a_u y^u \]

**Use homomorphic encryption**

[Paillier, ElGamal variant]

\[ enc(m_1+m_2) = enc(m_1) \cdot enc(m_2) \]

\[ enc(c \cdot m_1) = enc(m_1)^c \]

Note: R never sends its attestations
PM Protocol Details

For each $y_1 \ldots y_{k_S}$ compute (people who have attested to $S$):

$$
enc(P(y_i)) = enc\left(\sum_{u=0}^{k_R} a_u y_i^u\right) = enc(a_0) + enc(a_1)y_i + \ldots + enc(a_{k_R})y_i^{k_R}
$$
PM Protocol Details

Sender’s Attestation Server (AS)

Recipient (R)

\[ P(y) = (x_1 - y)(x_2 - y) \cdots (x_{k_R} - y) \]

\[ = \sum_{u=0}^{k_R} a_u y^u \]

\[ enc(a_0), enc(a_1), \ldots, enc(a_{k_R}) \]

For each \( y_1 \ldots y_{k_s} \) compute (people who have attested to \( S \)):

\[ enc(P(y_i)) = enc\left( \sum_{u=0}^{k_R} a_u y_i^u \right) = enc(a_0) + enc(a_1) y_i + \ldots + enc(a_{k_R}) y_i^{k_R} \]

Then

\[ enc\left( r \cdot P(y_i) + \{ y_i \rightarrow S \} \right) \]

random value

attestation

Recover \( y_i \rightarrow S \)

or a random value

Computation complexity is \( O(k_S^2) \)
PM Performance

![Graph showing computation time against sender input size for different recipient input sizes. The graph illustrates a linear relationship between sender input size and computation time, with lines for recipient input sizes of 640, 320, 160, 80, 40, 20, and 10. Each line represents a different recipient input size, with the computation time increasing as the sender input size increases.]
WL Effectiveness: Conservative

![Graph showing the effectiveness of WL with and without FoF for Corporations and Universities. The graph illustrates the percentage of mail whitelisted over the number of messages processed (in millions). The data points indicate a 12% gain and a 17% gain for Corporations and Universities respectively.]
WL Effectiveness: Strangers Only, Conservative

- 425% gain
- 320% gain
WL Effectiveness: Best Case

16% gain

13% gain

Percentage of Mail Whitelisted vs. # Messages Processed (millions)

With FoF (Corp)
Without FoF (Corp)
With FoF (Univ)
Without FoF (Univ)
WL Effectiveness: Strangers Only, Best Case

Percentage of Mail Whitelisted

# Messages Processed (millions)

With FoF (Corp)  
Without FoF (Corp)  
With FoF (Univ)  
Without FoF (Univ)

380% gain

550% gain