Robust Decentralized Authentication for Public Keys and Geographic Location

Vivek Pathak
Ph.D. Defense
Advisor : Liviu Iftode

December 3, 2008
Department of Computer Science
Rutgers University
What is Authentication?

- Authentication provides an assurance
  - About a person, message, place, etc.
    - Is it the correct email address?
    - Is it the correct public key?

- Application areas
  - Grant access
  - Manage reputation
  - Impart trust
Properties of Authentication

- Robustness
  - Different types of failure

- Level of decentralization
  - Centralized naming and access control
  - Centralized naming but decentralized access control
    - Email, Internet, Phone, etc.
  - Fully decentralized

- Need for human input
Problem Space

- Traditional authentication requires
  - Centralized security infrastructure
  - Decentralized human trust judgment

- Need for robust and decentralized authentication
  - Peer-to-peer systems
    - BitTorrent, Kazaa, Limewire
  - Open social networks
    - Facebook, MySpace, Friendster
  - Ad-hoc networks
Thesis & Contributions

Thesis

Robust decentralized authentication can be achieved by assuming the presence of an honest majority instead of relying on centralized security infrastructure or human trust judgment.

Contributions

- **SGKA**: Social-group key authentication for email. *Fifth Conference on Email and Anti-Spam, August 2008.*
Defense Outline

- Introduction
- Byzantine fault tolerant public key authentication for peer-to-peer systems
  - Social-group key authentication for email
  - Location authentication in ad-hoc networks
- Conclusions
Public Key Encryption

- Public-private key pair
- Bootstrap shared secret encryption
- Validation of digital signature
Authentication of Public Keys

- Mapping identities to public keys
  - Trusted third parties (TTP)
    - Certificate authority (CA)
  - Web of trust
    - PGP

```bash
$ssh eden.rutgers.edu
The authenticity of host 'eden.rutgers.edu (128.6.68.10)' can't be established.
Are you sure you want to continue connecting (yes/no)?
```
Authentication through CA

- Provide public key certificate
- Use secure channel for bootstrapping
Web of Trust

- Informal human authentication
  - PGP key rings
  - Levels of trust
Public-key Authentication Problem in Peer-to-peer Systems

- Heterogeneous peers
  - Lack of trusted third parties
  - Centralized security infrastructure is unsuitable

- Large scale peer-to-peer systems
  - Prohibitive administrative cost of centralized solutions

- Autonomous operation
  - Web of trust is unsuitable because of the lack of human feedback

- Peers can be malicious
Our Solution

- BPKA
  - Byzantine fault tolerant Public-key Authentication in peer-to-peer systems

- Properties of our solution
  - Decentralized
  - Automatic
  - Robust
System Model

- Mutually authenticating peers
  - Associate network end-point with public key
- Asynchronous network
  - No partitioning
  - Eventual delivery after retransmissions
- Disjoint message transmission paths
  - Man-in-the-middle attack on Ø fraction of peers
Attack Model

- Malicious peers
  - Honest majority
  - At most $t$ of the $n$ peers are malicious, where $t = \frac{(1-\Theta)}{3} n$
  - Threshold of malicious peers ($t$) follows from the analysis shown in the following slides

- Adversaries
  - Relax network-is-the-adversary model
  - Limited power to prevent message delivery
    - Indicated by the constant $\Theta$
  - Unlimited power to spoof

Further details in Chapter 2.3
Authentication Model

- Challenge-response protocol
  - Authenticate public key
  - Use encrypted nonce as challenge
  - No active attacks

- Man in the middle attack
  - Limited number of attacks $O(\emptyset)$

- Proof of possession of public key $K_A$
  $$P_{BA} = \{B,A,\text{Challenge},K_A(r)\}_B, \{A,B,\text{Response},r\}_A$$
Authentication Model (contd.)

- Distributed authentication
  - Challenge response from multiple peers
  - Gather proofs of possession

- Lack of consensus on authenticity
  - Malicious peers
  - Man-in-the-middle attack
Authentication Correctness

- Validity of proofs of possession
  - $P_{EA} = \{E, A, \text{Challenge}, K_A(r)\}_E, \{A, E, \text{Response}, r\}_A$
    - Nonce is recovered correctly
    - Digital signatures on messages

- Recent proofs stored by peers
  - Signed messages
  - Prove malicious behavior

<table>
<thead>
<tr>
<th>Proofs at B</th>
<th>$P_{BA}$</th>
<th>$P_{CA}$</th>
<th>$P_{DA}$</th>
<th>$P_{EA}$</th>
<th>$P_{FA}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proofs at A</td>
<td>$P_{BA}$</td>
<td>$P_{CA}$</td>
<td>$P_{DA}$</td>
<td>$P_{EA}$</td>
<td>$P_{FA}$</td>
</tr>
</tbody>
</table>
Byzantine Agreement

- Publicize lack of consensus
  - Share proofs of possession

- Each peer tries to authenticate A
  - Sends its proof-of-possession vector to every peer
  - Byzantine agreement on authenticity

- Majority decision at every peer
  - Identify malicious peers
  - Complete authentication

<table>
<thead>
<tr>
<th></th>
<th>From B</th>
<th>From C</th>
<th>From D</th>
<th>From E</th>
<th>From F</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Byzantine Agreement Correctness

Consider proofs received at a peer

- Set of peers participating in the authentication protocol
- $\Phi_n$ on compromised path to A
- $\Phi_n$ on compromised path to peer
- $t$ malicious peers
Byzantine Agreement
Correctness (contd.)

- Byzantine Agreement
  - $t + 2\mathcal{O}n$ proofs may not arrive
    - Peer receives at least $n-t-2\mathcal{O}n$ proofs
  
  - $t + 2\mathcal{O}n$ proofs may be faulty
    - Peer receives at least $n-2t-4\mathcal{O}n$ correct agreeing proofs
    - Peer decides correctly by majority if $n-2t-4\mathcal{O}n > t + 2\mathcal{O}n$

  - Agreement is correct if $t < \frac{(1-6\mathcal{O})}{3} n$

- Theorems proved in Chapter 2.5
  - Honest peers are authenticated correctly if there is an honest majority
  - Honest majority is preserved
Authentication with Trusted Groups

- Reduce messaging cost

- Trusted group
  - Authenticate public keys

- Probationary group

- Un-trusted group
  - Known to be malicious

- Details in Chapter 2.5
  - Probability of incorrect authentication drops exponentially with time
Public Key Infection

- Optimistic trust
  - Lazy authentication
  - Reduced messaging cost

- Use logical and vector timestamps [Lamport78]
  - Determine messages to be exchanged during anti-entropy sessions
  - Detect message delivery

- Double exponential drop in number of uninfected peers with time

- $O(n \log n)$ cached messages
Evaluation of Peer-to-Peer Authentication

- Implemented BPKA in a C++ library
  - Simulation program makes library calls and keeps counters
    - Number of messages
  - Study effects of
    - Group size
    - Trust relationships
    - Malicious peers
Effects of Group Size

- Constant cost for trusted group peers
- Probationary peers process $O(n^2)$ messages
- Trust graph does not affect the cost
  - Randomized trusted sets from Bi-directional trust
Effects of Malicious Peers

- Rapid increase of messaging cost
  - With group size
  - With proportion of malicious peers
- Byzantine agreement has quadratic messaging cost
Summary

- Byzantine fault tolerant public key authentication protocol (BPKA)
  - Scalable to large peer-to-peer systems
  - Made feasible by using stronger network assumptions

- Robustness
  - Tolerate malicious peers

- Decentralized authentication
  - Does not need trusted third party
  - Eliminate total trust and single point of failure

- Authentication without human trust judgment
Defense Outline

- Introduction
- Byzantine fault tolerant public key authentication for peer-to-peer systems
  
  **Social-group key authentication for email**
  
  - Location authentication in ad-hoc networks
- Conclusions
Email Trustworthiness Problem

- Should I trust the email?
  - Provide login details

- How is the email trust problem solved
  - Sender authentication
  - Content integrity

- Multi-billion dollar spam industry
  - Message Labs report
  - Phishing emails 78.9%
Traditional Sender Authentication Approaches

- S/MIME Standard RFC-2633 [Ramsdell99]
  - Secure email with digital signature
    - Can not forge sender identity
  - PKI deployment
    - Difficult to use across organizational trust boundaries

- PGP – Pretty good privacy [Zimmerman95]
  - Sender authentication without trust infrastructure
  - Web of trust needs user sophistication
    - Maintain key rings
Spam Control Approaches

- DKIM - Domain Keys Identified Mail
  [Allman07, Goodrich05]
  - Infrastructure compatible
  - Authenticate sender domain
    - Need administrative control and upgrade
    - DNS attacks

- Content classification
  - Identify spam features and keywords
  - False positives of spam filters
Our Solution

- Social-Group Key Authentication for Email (SGKA)
  - Extend BPKA to support offline communication in social groups
  - Use email user base characteristics
    - Pre-existing social groups of collaborating users
    - Unsophisticated but many
  - Decentralized
    - Communication across organizational boundaries
  - Robust

- Infrastructure free end-to-end public key authentication
Threat Model

- Network attacks
  - Impersonation
  - Message delivery
    - Add or delete messages
    - Modify content en-route
    - Man in the middle attack

- Misbehaving participants
  - Stopping failures
  - Byzantine faults
    - Malicious participants may appear to work fine and mislead participants
Solution Approach

Alice has a group of trusted peers C, D, and E.

Alice knows the authentic public keys of the trusted peers.

Alice authenticates the public key of Bob with help of her trusted peers.

An unknown peer Bob sends an email to Alice.

Authenticated public keys create secure communication paths.
Public Key Authentication in the Email Environment

- Email trust environment
  - Hard to subvert message delivery
  - Easy to spoof messages
  - Authenticate self-generated public keys of peers using pre-existing social groups

- Email communication characteristics
  - Offline communication pattern
  - Diverse activity levels of email users

- More details in Chapter 3.4
Overlay Authentication Protocol on Email Messages

- SMTP extension headers X-Bft-Auth-*
  - Provide backward compatibility
  - Carry cryptographic data
    - Challenge-response nonce and authentication vote
    - Public key of the sender
    - Digital Signature

- Protocol Operations
  - Email_Peer: Request authentication
  - Email_Response: Return authentication vote
  - Infer_Trust: Decide authenticity by majority

- Two modes of operation
  - Lazy mode adds no new messages
  - Eager mode sends additional email messages to speed up authentication
Authentication Plugin for Thunderbird Email Client

- Implement end-to-end authentication in the email client
  - C++ Library
  - Javascript plugin
- Piggy-back on emails
- Able to cache protocol messages
- Downloadable software
  http://discolab.rutgers.edu/byzantine/#software
Experimental Methodology

- Protocol execution on email trace
  - Collect anonymous trace from email server log
    - cs.rutgers.edu 1.19 million emails in 92 days
    - ask.com 2.54 million emails in 56 days
  - Select peers from email social group

- Experimental parameters
  - Message payload
  - Lazy and eager modes of operation

- Measurements
  - Fraction of peers authenticated in 92 days
  - Overhead introduced
Cost of Lazy Mode Authentication

- No additional email messages
- Use local cache to store and forward protocol messages
Authentication Performance in Lazy Mode

- Full backward compatibility
  - Too slow

- Protocol messages piggy-backed on real emails
  - Avoid spam filters

- Optimistic authentication
  - Partial completion
Authentication Performance in Eager Mode

- Rapid authentication
- Need to inject emails periodically
- Authenticate 80% of peers in two periods
Email Overhead Introduced in Eager Mode

- Moderate overhead of new emails
- Overhead increases with greater eagerness
Usability Discussion

- **Lazy mode**
  - Dependent on natural email communication pattern
  - Emails have different urgency levels
    - No way to get quicker conclusion
  - Users need to send and receive emails for protocol to progress

- **Eager mode**
  - Risk of getting caught in spam filters
  - Email client needs to be online in order to participate in the authentication protocol
Summary

- Social-group key authentication
  - Decentralized
  - Use social-groups of email users
  - Compatible with existing infrastructure
  - Robust
    - Byzantine faults tolerance
  - Automatic sender authentication

- Modest overhead
  - Performance issues handled through eager mode

- Applicable to other online social networking areas
  - Facebook, Myspace, Friendster
Defense Outline

- Introduction
- Byzantine fault tolerant public key authentication for peer-to-peer systems
- Social-group key authentication for email
- Location authentication in ad-hoc networks
- Conclusions
Location Authentication

- Location aware devices
- Ad-hoc networking capability
- Authenticating geographical location
False Location Attacks

- **Motivations**
  - Strategic
    - Battlefield
  - Economic
    - Benefit of misreporting location
Location Privacy

- Surveillance
- Crime
  - Home location
Our Solution

- Geographical secure path routing (GSPR)
  - Infrastructure free geographic routing protocol
  - Anonymous nodes

- Robust
  - Malicious nodes
  - False location attack
  - Other malicious behavior like dropping packets

- Decentralized location authentication
  - Location of destination
  - Path taken by a routed message
System Model

- Ad-hoc network
- Nodes have GPS
  - Cell phones
  - Cars
  - One laptop per child
- Geographic communication
Geographic Routing Greedy Mode

- Periodic node beacon
  - Transmit node location

- Ad-hoc routing protocol [Karp00]
  - Stateless
  - Route closest to the destination
Geographic Routing Perimeter Mode

- Greedy mode failure
  - Enter perimeter mode
  - Route around the network hole
Features of Geographic Routing

- Highly effective ad-hoc routing protocol
  - Stateless
    - Handle mobility
    - Only one-hop neighborhood local state
  - Scalable
    - Large number of nodes
    - Large number of destinations

- Nodes should know their location
Traditional Usage of Geographic Routing

- Two-step usage scenario
  - Find location of the node of interest
  - Geographic routing finds route to location

- Vulnerabilities
  - Location errors and attacks
  - Location privacy
Geographical Secure Path Routing (GSPR) Overview

- Geographic hashes
  - Un-forgeable location dependent integers

- Periodic beacon extensions

- Secure geographic routing protocol
  - Query-response messaging model

- Detect malicious nodes
  - Operate in promiscuous networking mode
Geographic Authentication Model

- Nodes are anonymous
  - Use temporary pseudonyms
  - Generate their own public-private key pairs
  - Messages are digitally signed

- Node locations mapped to integer vector space
  - Application dependent global constant for mapping
Assumptions

- Wireless network
  - Bi-directional links
  - 802.11 MAC
- Physical layer defense against jamming
  - Spread spectrum techniques
- Global range limitation
- Overhear transmissions of neighbors
- Adversaries cannot affect honest nodes
  - Reception or transmission
Design of Geographic Hashes

- Set of integers
  - One set of geographic hashes per node

- Geographic hash value designed to depend on location

- Use one-way associative functions
  - Provide un-forgable positioning
  - Multiple calculation paths give identical value
Design of Geographic Hashes (contd.)

- One function per dimension
- Function for time
  - Every node applies it once per time period
  - Short lived geographic hashes
- Associative hash functions
  - Values must agree across space and time
  - Details in Chapter 4.5
Periodic Beacon

- Traditional role
  - Share neighboring node locations

- Our extended beacon
  - Self generated public key of the node
  - Share known geographic hashes and locations
  - Share evidence of malicious behavior
  - Beacon messages are digitally signed
Malicious Node Detection

- Each node detects malicious neighbors
  - Range constraint violation
  - Disagreeing Geographic hashes
  - Overhear malicious forwarding behavior

- Takes corrective action
  - Ignore malicious node for routing
  - Share with neighbors
  - Malicious actions are provable because messages are digitally signed
Geographical Secure Path Routing Sketch

- Query-response messaging model
  - Geographic routing on forward path
  - Source routing on reverse path

- Extend traditional geographic routing
  - Carry authenticated node location list on forward path
  - Carry locations and public keys on reverse path

- Proof in Chapter 4.6
  - Protocol completion authenticates location
  - Protocol completes if there is a sufficient density of honest nodes
Performance Evaluation

- Compare performance with traditional GPSR
  - Implement secure routing in NS2
  - Modify GPSR routing implementation to allow malicious nodes

- Experimental scenarios
  - Node density
  - Percentage of malicious nodes
  - Mobility

- Measurements
  - Data delivery rate
  - Path length traversed by the message
Effect of Node Density on Data Delivery Rate

- GPSR is susceptible to malicious nodes
- Node density does not help
- Compare with our GSPR secure routing protocol
- Take advantage of node density to resist routing errors introduced by malicious nodes
Effect of Malicious Nodes on Data Delivery Rate

- Insecure GPSR breaks down with malicious nodes
- Resilience of GSPR to a large fraction of malicious nodes
Mobility & Malicious Nodes

- Mobility does not help GPSR significantly
- Secure geographical routing improves delivery rate with mobile nodes
- Take advantage of mobility by finding new non-malicious nodes
Summary

- Geographical secure path routing (GSPR)
  - Authenticate geographic location of anonymous nodes
  - Using short lived un-forgable geographic hashes
  - Practical for resource rich ad-hoc networking environments

- Robust
  - Resist large number of malicious nodes

- Decentralized ad-hoc protocol
  - Authenticate location of anonymous nodes
  - Authenticate public key of nodes
Conclusions

- Robust decentralized authentication without security infrastructure or human trust judgment
  - Design based on co-operating honest majority

Contributions

- **BPKA**: Byzantine fault tolerant public key authentication for peer-to-peer systems
  - C++ implementation of peer-to-peer authentication protocol
- **SGKA**: Social-group key authentication for email
  - Thunderbird plugin for automatic sender authentication
- **GSPR**: Location authentication in ad-hoc networks
Future Directions

- Application of our honest majority based approach to upcoming problem areas
  - Online social networking
    - Facebook, MySpace, etc.
    - Content authentication on the Internet

- Hybrid trust models

- Use authentication results
  - Collaborative spam control
  - Economic incentive schemes
  - Social networking on ad-hoc networks
Acknowledgements

- Advisor: Liviu Iftode
- Committee:
  - Danfeng Yao
  - Vinod Ganapathy
  - Roberto Tamassia (Brown University)
- Discolab members
  - Lu Han
  - Mohan Dhawan
- External
  - Marios D Dikaiakos
  - Florin Sultan
  - Tao Yang
  - Thu Nguyen

Thanks