Revere—Disseminating Security Updates at Internet Scale

Jun Li
CIS 610 Advanced Research Topics in Network Security
January 6th, 2004

Motivation
• Threats propagate quickly through the Internet
  – Viruses, worms, Trojan horses, etc.
  – e.g., Code Red worm, Slammer worm
• Critical security info throughout the Internet is often stale
  – Victims lack up-to-date knowledge of new threats
• We must react at the same speed

Goal of Revere
• To disseminate security updates throughout the Internet quickly, securely, and with high assurance
  – Early-warning signal
  – Virus signature
  – Intrusion detection event
  – Certificate revocation list
  – Offending characteristics recorded at firewall
  – Security patches
  – . . . . .

Challenges
• Fast
  – Must not be slower than the propagation of threats
• Secure
  – Revere is a tempting target for attackers
  – A corrupted Revere can be misused or abused
• Resilient
  – Interruption threats by compromised nodes, or any kind of failure
  – Cryptography does not assure delivery
  – Authenticated acknowledgements are insufficient
• Scalable
  – Any Internet host is a potential recipient
  – Node disconnection will be common

Simple Transmission Techniques
• Unicasting
• Broadcasting
• Flooding
• IP multicasting

Virus Signature Distribution
• Let users download from a website
• Set up a central server
• A naive peer-to-peer approach
A Government Practice

- Typically, a Federal Computer Incident Response Capability team e-mails alerts to agencies
- But when facing the "I Love You" virus, many agencies shut down their email servers
- Thus, phoned and faxed alerts instead, one at a time
  - What a time-consuming tedious procedure!
- Afterwards . . .
  - A completely automated new system is designed that claims to handle 96 phone lines and deliver 880 faxes/hour
  - They also look into an AM radio system for federal employees to check every morning

Diane Frank. "One if by phone, two if by fax." Federal Computer Week, September 2000

So, How Would Revere Meet the Challenges?

- Non-centralized delivery structure
- Use redundancy to support information transmission resiliency
- Secure both the dissemination procedure and the delivery structure

The Revere Solution

- Revere builds overlay networks, called RBones, on top of the Internet
- . . . and uses RBone to deliver security updates
  - Every node can also forward updates
  - Disconnected nodes will be handled
- Runs at application level
  - Great flexibility
  - No changes to underlying network infrastructure
  - Implemented in Java
  - Deployment is easy

RBone: A Self-Organized Resilient Overlay Network

- Redundancy-based resiliency
  - Multiple delivery paths
  - Therefore multiple parents
  - Select as-disjoint-as-possible paths
- Self-organized overlay
  - Easy join
  - Easy withdrawal
  - Broken nodes
  - Broken links

RBone Join Procedure

- Search for existing nodes
  - Directory service
  - Multicast-based expanding-ring or expanding-wheel search
  - Contact already-known existing nodes
- Negotiate to select best parent
  - Again, multiple parents are allowed!
- Three-way-handshake negotiation protocol
  - Reciprocal selection

Parent Selection

- What parental qualities matter?
  - Efficiency: is the delivery via this parent fast?
  - Resiliency: is the delivery via this parent disjoint with other paths?
    - If not completely disjoint, how much is the overlap?
Parent Selection (cont’d)

• The path vector of a node
  – Describes the fastest path:
    • Latency
    • An ordered list of nodes to cross
    • Denoted as \( p_v(n) \)
• The path vector associated with a parent
  – Described the fastest path through the parent
  – Denoted as \( p_v(n, p) \)
• The resiliency level of a node’s parent
  – Calculated by comparing the path vector associated with the parent and the path vector of the node

Path-Vector-Based Parent Selection Algorithm

Suppose node \( c \) is deciding a potential parent \( x \)

if \( c \) has not reached the maximum number of parents
  select \( x \);
else if \( p_v(c, x) \) is faster than \( p_v(c) \)
  select \( x \);
else if resiliency(\( x \)) better than resiliency(a current parent)
  select \( x \);
else
  do not select \( x \);

RBone Maintenance

• Heartbeat messages
  – To verify node liveness
  – To update path info associated with every parent
  – Carry timestamps
  – Deal with the broken parent, or any broken node on a path
• Explicit messages
  – To tear down a parent-child relationship
• Corrupted security updates also trigger adjustment

RBone with Multiple Dissemination Centers

• If a node wants to hear from multiple centers
  – It joins multiple RBones, each rooted at a different center
  – this becomes undesirable if too many centers
• Build a common RBone rooted at a rendezvous
  – Every center delivers updates to the rendezvous
• Multiple rendezvous points can be set up

Dissemination Procedure

• A dual mechanism
  – Pushing as the main method for broadcasting security updates from a dissemination center
  – Pulling as the supplementary method for catching up with missed security updates
• Security update format

<table>
<thead>
<tr>
<th>Type</th>
<th>seqno</th>
<th>timestamp</th>
<th>payload</th>
<th>signature</th>
</tr>
</thead>
</table>

Pushing Security Updates

• Adaptive transmission
  – TCP
  – UDP
  – IP multicast
  – etc.
• Duplicate checking
  – Every Revere node remembers the range of historical sequence numbers
• Security checking
  – (Will be addressed later)
Pulling Security Updates

- By the time a disconnected node reconnects, it may have missed some security updates
  - Parents do not keep old updates
  - Parents might no longer be parents
  - Retransmission by the dissemination center is not scalable
- Repository servers
  - Nodes that keep old security updates
  - Usually maintain stable connection
  - Clients directly contact those servers
- A newly pulled security update is also forwarded to child nodes, if any

Security Assumptions

- Center’s public key is wellknown
- Large percentage of nodes are cooperative
- Any node could be corrupted
  - The center cannot be corrupted, but its private key could be compromised
- No uniform security scheme to protect node-to-node control messages
  - For example, some nodes may run the Kerberos service to authenticate other nodes, some may employ public-key-based authentication

Securing the Dissemination Process

- Integrity of security updates
  - A dissemination center has a public/private key pair
  - Every security update carries a digital signature signed by the center
- Availability of security updates
  - Redundant delivery

Center Key Disclosure

- Disastrous if the private key of a center is disclosed
  - The public key must be invalidated
- Public key invalidation
  - Send a key invalidation message
    - Signed with the disclosed private key
    - Delivered in the same way as updates
    - Every recipient verifies the message with the current public key
    - Then discards this public key
  - How secure is this method?
    - Fine, if an attacker also distributes key invalidation messages
    - Resilient, since it follows the same routes as normal security updates

Securing RBone

- Every node can enforce several different security schemes
  - Node authentication
  - Message filtering
  - Etc.
- The functionality of a specific security scheme can be easily plugged in
- Node-to-node communication is initiated with security scheme negotiation

Security Scheme Negotiation

- Select scheme b for messages toward node B
  - Select scheme A for messages toward node A
  - A’s first message to B
  - Authentication: g_A/K_B = H(b, K_B) / H(g_B, K_B)
  - A’s authentication: H(a, K_B) / H(g_B, K_B)
  - B’s authentication: H(b, K_A) / H(g_A, K_A)
  - A’s signature: X_A(S) / (signed)
  - B’s signature: X_B(S) / (signed)
  - Parent (node A)
    - S = H(a, K_B) / H(g_B, K_B)
    - H(a, X_A) / (signed)
    - Broadcast to children
  - Child (node B)
    - S = H(b, K_A) / H(g_A, K_A)
    - H(b, X_B) / (signed)
    - Broadcast to parents
Revere--Delivering Security Updates at Internet Scale

Metrics

- RBone maintenance bandwidth
- Dissemination bandwidth
  - Join bandwidth
  - Join latency
  - Dissemination latency
  - Dissemination resiliency

What's the challenge?

- Revere is a large-scale distributed system
- Empirical experiments incur prohibitive cost
  - Required to obtain, access, configure, maintain, and collect
    results from more than a few hundred machines
- Simulation is more scalable, but
  - Expensive to develop
  - Slow to run
  - Possibly inaccurate (hidden costs and subtle timing effects) & buggy
    Must be validated against real system

The "Overloading" Approach

- A physical machine is overloaded with multiple (virtual) Revere nodes
- Each Revere node runs the real software
- Achieves larger scalability using multiple machines

Three ways to handle resource contention

- Locking mechanism
  - Only one virtual node at a time initiates operation
  - No contention because of serialization
- Divide and conquer
  - Divide a task into non-overlapping subtasks
  - Measure each subtask in non-overloaded environments
  - Measure occurrences in full system, and then sum
  - Resource contention now omitted from total
- Slowdown analysis
  - Processing time $T_0$: $n$ logical nodes on $n$ machines
  - Processing time $T$: $n$ logical nodes on 1 machine
  - Slowdown factor $T/T_0$

Measurement Environment

- A testbed of 10 machines
  - Overloaded with up to 3,000 Revere nodes
- Topology
  - GT-ITM topology generator
  - A topology server for node assignment
- Configuration
  - Every node must have 2 parents, but $\leq 10$ children

Join Performance

- Join phase
  - Token-controlled resource-locking mechanism
  - One-at-a-time join
  - No contention because of serialization
Revere--Delivering Security Updates at Internet Scale

### Join Performance

- **Outbound join bandwidth per node (KB)**
  - Measured in dissemination phase

- **Join latency per node (sec)**
  - Dissemination latency

![Join Performance Diagram](image)

### Dissemination Speed

- **Maximum:** $y = 1.818 \ln x - 3.902$
  - Number of total Revere nodes

- **Average:** $y = 0.771 \ln x - 1.810$
  - Number of total Revere nodes

![Dissemination Speed Diagram](image)

### Dissemination Resiliency

- **Resiliency test phase**
  - Every node assigned a uniform probability of failure
  - Measured a 3000-node RBone with maintenance disabled

- **Results:**
  - All reachable with less than 2% failure probability
  - Still very resilient with higher failure probability

![Dissemination Resiliency Diagram](image)

---

**Join Performance**

- **Outbound join bandwidth per node (KB)**
  - Join latency per node (sec)

**Dissemination Speed**

- **Maximum:** $y = 1.818 \ln x - 3.902$
  - Number of total Revere nodes

- **Average:** $y = 0.771 \ln x - 1.810$
  - Number of total Revere nodes

**Dissemination Resiliency**

- **Resiliency test phase**
  - Every node assigned a uniform probability of failure
  - Measured a 3000-node RBone with maintenance disabled

- **Results:**
  - All reachable with less than 2% failure probability
  - Still very resilient with higher failure probability
Some Related Work

- RON – resilient overlay network
  - Inserts an overlay network layer between routing and application
  - Allows faster routing failure recovery & application-specific routing

- Other overlay networks
  - Tree-structured dissemination is not resilient
  - Nodes are not always connected at delivery time
  - Security handling is not sufficient

- Multi-path routing
  - A router-level implementation
  - Primarily for load balancing or congestion control
  - Must handle security issues at router level
  - Replay prevention, key distribution
  - Deployment is challenging

Work Summary of Revere Project

- Designed
  - The structure, the dissemination, the security, the . . .

- Implemented
  - 45,010 lines of Java code in the prototype system

- Measured
  - The number of nodes varies from 250 to 3,000

- Demonstrated
  - DARPA Site Visit
  - UCLA Annual Research Review

- Published and presented
  - NSPW’99, NISSC’99, Testcom’02
  - Also submitted to OSDI’02
  - Dissertation draft is at your hand

Conclusions

- **Necessary work**: Since attackers already distribute malicious functions rapidly, an even faster notification system is required.

- **Encouraging results**: It is feasible to disseminate security updates to much of today’s connected Internet quickly, securely, and with high assurance.

- **Broad applicability**: Revere is not limited to only security updates.