Control and data plane routing security

Survey presentation by:

Eric Chen
Frank Mokaya
Yu Seung Kim
Concepts

- Control plane
  - Addressing how to establish the routing path from a source to a destination
  - Accuracy, robustness
  - ex) paving roads and setting up road signs

- Data plane (forwarding plane)
  - Addressing what to do with packets arriving at a node
  - Efficiency
  - ex) interpreting road signs
Survey

- Topic: How to protect against the network layer attacks which interfere with securely establishing routing path and delivering packets

- Papers
  - Wendlandt et al., “Don’t Secure Routing Protocols, Secure Data Delivery” - Frank
Attacks on ad hoc network routing protocol

- Routing disruption attack
  - Forge routing packets to create a routing loop
  - Blackhole: attracts and drop
  - Greyhole: drops selective packets
  - Gratuitious detour: make packets appear longer
  - Rushing attack: target against on-demand protocols that use duplicated suppression
- Wormhole: record packets at one point, send at another point

Not bad if used for all packets, bad if only send control packets.
Wormhole attacks

- Dangerous in wireless applications, such as proximity based authentication systems
- Packet leash- geographical and temporal.
  - Temporal: require precise time sync, calculate received time and time stamped time.
  - Geographical: Calculate the velocity of packet, and a distance upper bound. (might not work)
Key setup

• We want to authenticate each node.
• Some routing protocol requires shared private key between all pairs of nodes
  - Can share before deployment
  - Exchange through physical link (i.e. touch)
  - If public key is established, we can exchange using PKI
Key setup - PKI

- PKI assumes we know each node’s public key, by trusting some central authority
- One node may be deployed before a future node supplies its public key
- When a new node joins the network, it has to receive information
  - prone to eavesdroppers and active attackers
Key setup - SUCV

- statistically unique cryptographically verifiable (SUCV) addresses
  - hashes its public key, and use it as its IP address (IPv6)
- Still require the network to know about the list of legitimate nodes.
Key setup - CA

- Trusted third party
- However, for a CA to be reachable, routes have to be established already...
- CA vulnerable to compromise
  - We can trust only multiple CA certified keys
Key setup - PGP

- Pretty Good Privacy (PGP)
  - nodes sign each other. $A \Rightarrow B \Rightarrow C \Rightarrow D$ implies $A$ trusts $B$ who trusts $C$ who trusts $D$

- Can require additional node-disjoint path

- Can limit certificate chain’s length
Key revocation

- If we use CA to sign the revocation, the revoked node will just drop the packet
- Good approach - flood revocation packets
Secure Efficient Ad hoc Distance vector routing protocol - SEAD

- Distance vector routing requires:
  - the node’s shortest known distance all the destinations
  - The neighbor router that is the first hop on the shortest route to the destination.
  - Broadcasts its own routing table
Secure Efficient Ad hoc Distance vector routing protocol - SEAD

- Hash chain authentication
  - Each node can only increase the routing distance of previous hash chain, but cannot decrease.
Secure on-demand routing protocol for ad hoc networks -

\[ S: \quad h_0 = MAC_{K_{SD}}(\text{REQUEST}, S, D, id, ti) \]

\[ S \rightarrow*: \quad \text{REQUEST, } S, D, id, ti, h_0, (), () \]

\[ A: \quad h_1 = H[A, h_0] \]

\[ M_A = MAC_{K_{Ati}}(\text{REQUEST, } S, D, id, ti, h_1, (A), ()) \]

\[ A \rightarrow*: \quad \text{REQUEST, } S, D, id, ti, h_1, (A), M_A \]

\[ B: \quad h_2 = H[B, h_1] \]

\[ M_B = MAC_{K_{Bti}}(\text{REQUEST, } S, D, id, ti, h_2, (A, B), (M_A)) \]

\[ B \rightarrow*: \quad \text{REQUEST, } S, D, id, ti, h_2, (A, B), (M_A, M_B) \]

\[ C: \quad h_3 = H[C, h_2] \]

\[ M_C = MAC_{K_{Cti}}(\text{REQUEST, } S, D, id, ti, h_3, (A, B, C), (M_A, M_B)) \]

\[ C \rightarrow*: \quad \text{REQUEST, } S, D, id, ti, h_3, (A, B, C), (M_A, M_B, M_C) \]

\[ D: \quad M_D = MAC_{K_{DS}}(\text{REPLY, } D, S, ti, (A, B, C), (M_A, M_B, M_C)) \]

\[ D \rightarrow C: \quad \text{REPLY, } D, S, ti, (A, B, C), (M_A, M_B, M_C), M_D, () \]

\[ C \rightarrow B: \quad \text{REPLY, } D, S, ti, (A, B, C), (M_A, M_B, M_C), M_D, (K_{Cti}, K_{Bti}, K_{Ati}) \]

\[ D \rightarrow A: \quad \text{REPLY, } D, S, ti, (A, B, C), (M_A, M_B, M_C), M_D, (K_{Cti}, K_{Bti}, K_{Ati}) \]

\[ A \rightarrow S: \quad \text{REPLY, } D, S, ti, (A, B, C), (M_A, M_B, M_C), M_D, (K_{Cti}, K_{Bti}, K_{Ati}) \]
Ad hoc On-demand Distance Vector routing protocol (AODV) - ARAN

\[ S \rightarrow *: \quad (\text{ROUTE REQUEST, } D, \text{cert}_S, N, t)_K^- \]

\[ A \rightarrow *: \quad ((\text{ROUTE REQUEST, } D, \text{cert}_S, N, t)_K^-)_K^- A, \text{cert}_A \]

\[ B \rightarrow *: \quad ((\text{ROUTE REQUEST, } D, \text{cert}_S, N, t)_K^-)_K^- B, \text{cert}_B \]

\[ C \rightarrow *: \quad ((\text{ROUTE REQUEST, } D, \text{cert}_S, N, t)_K^-)_K^- C, \text{cert}_C \]

\[ D \rightarrow C: \quad ((\text{ROUTE REPLY, } S, \text{cert}_D, N, t)_K^-) \]

\[ C \rightarrow B: \quad ((\text{ROUTE REPLY, } S, \text{cert}_D, N, t)_K^-)_K^- C, \text{cert}_C \]

\[ B \rightarrow A: \quad ((\text{ROUTE REPLY, } S, \text{cert}_D, N, t)_K^-)_K^- B, \text{cert}_B \]

\[ A \rightarrow S: \quad ((\text{ROUTE REPLY, } S, \text{cert}_D, N, t)_K^-)_K^- A, \text{cert}_A \]
Ad hoc On-demand Distance Vector routing protocol (AODV) - ARAN

- Routing error:

\[ B \rightarrow A : \langle (\text{ROUTE ERROR}, S, D, \text{cert}_B, N, t)_{\kappa_B} \rangle \]
\[ A \rightarrow S : \langle (\text{ROUTE ERROR}, S, D, \text{cert}_B, N, t)_{\kappa_B} \rangle \]
Ad hoc On-demand Distance Vector routing protocol (AODV)- SAODV

\[
\begin{align*}
S \rightarrow *: & \langle\text{RREQ, id, S, seq}_S, D, \text{oldseq}_D, h_0, N\rangle_{K_S^-}, 0, h_N \\
A \rightarrow *: & \langle\text{RREQ, id, S, seq}_S, D, \text{oldseq}_D, h_0, N\rangle_{K_S^-}, 1, h_{N-1} \\
B \rightarrow *: & \langle\text{RREQ, id, S, seq}_S, D, \text{oldseq}_D, h_0, N\rangle_{K_S^-}, 2, h_{N-2} \\
C \rightarrow *: & \langle\text{RREQ, id, S, seq}_S, D, \text{oldseq}_D, h_0, N\rangle_{K_S^-}, 3, h_{N-3} \\
D \rightarrow C: & \langle\text{RREP, D, seq}_D, S, \text{lifetime}, h'_0, N\rangle_{K_D^-}, 0, h'_N \\
C \rightarrow B: & \langle\text{RREP, D, seq}_D, S, \text{lifetime}, h'_0, N\rangle_{K_D^-}, 1, h'_{N-1} \\
B \rightarrow A: & \langle\text{RREP, D, seq}_D, S, \text{lifetime}, h'_0, N\rangle_{K_D^-}, 2, h'_{N-2} \\
A \rightarrow S: & \langle\text{RREP, D, seq}_D, S, \text{lifetime}, h'_0, N\rangle_{K_D^-}, 3, h'_{N-3} \\
\end{align*}
\]
Ad hoc On-demand Distance Vector routing protocol (AODV)- ARAN

- Routing error:

\[ B \rightarrow A: \quad (\text{RERR}, D, \text{seq}_D)_{K_B} \]
\[ A \rightarrow S: \quad (\text{RERR}, D, \text{seq}_D)_{K_A} \]
Reputation-based system

- 4 components - monitor, trusted monitor, reputation system, and path manager
- Each node’s monitor ensure the next-hop node is forwarding correctly.
- Every node’s location reputation system gets updated to the trusted monitor routinely
Securing Routing Protocols vs. Securing Data Delivery

Main Ideas:
- Secure communication possible without securing routing protocols
- Since end-hosts/edge routers already provide end-to-end security, Routing infrastructure only has to guarantee availability
- Availability is best achieved by exposing multiple paths than heavy weight mechanisms
Problem with current Routing security focus

- Secure Routing too little
  - No safety against eavesdroppers, packet modification
  - No protection against data plane bugs such as misconfigured ACLs e.t.c.

- Secure Routing too much
  - Mechanisms in place too heavy weight
    - Require router hardware upgrades for crypto processing

- Solution: Routing -> availability only
  End Systems -> confidentiality, integrity
How do you guarantee availability?

- Create a system where:
  1. Clients learn multiple potential paths to a destination, instead of a single “best path”
  2. Clients use end-to-end security mechanisms and monitor path performance to detect good paths.
  3. Clients can use adequate paths and change routes if necessary
Availability Centric Routing

- Consists of 4 components
  - Multipath via Availability providers
    - Uses deflection points in AP’s network to service customer requests for new routes
  - End to End Integrity Check
    - Allows end systems to authenticate destinations via generic IPSec, SSL mechanisms
  - Availability monitoring
    - Allow for path switching (e.g. in case of poor performance)
  - Multiple Path distribution algorithm
    - Select working paths with high probability
ACR Overview: How it works

1) Source attempts to set-up a secure channel using default path
2) If set-up fails, it can request alternate paths from its AP, “probing” until it finds a working path
3) Sources monitor path performance, requesting alternate paths if the current path is inadequate
Threat to ACR: sub-prefix hijacks...

- If a provider ISP is duped, it is possible that a stub AS will not be reachable by any path seen by the AP.

Problem:
D is only reachable by legacy provider P. M announces a smaller prefix, routing all traffic to D through it. D becomes unreachable by the availability provider A.

Solution:
ACR causes D to emulate ‘flat address routing’. It announces its prefixes as /24’s, the longest prefix allowed by most ISPs. P no longer diverts its packets.
Is ACR Feasible?

- Low barriers to adoption
  - No need for crypto hardware
  - Supports current authentication methods
  - Deflection services already available

- Backwards compatibility
  - Independently runs alongside BGP without need for separate infrastructure

- Well incentivized deployment
  - Allow customers to take advantage of deflection for better performance
ACR SUMMARY

- Secure inter-domain routing proposals are heavy-weight, but still insufficient
- If end-points set up secure channels, the routing infrastructure must only provide multiple paths to guarantee availability
- This approach has highly attractive properties for incentivized deployment
Packet-dropping Adversary

- Attacker drops benign packets in the midst of routing path to reduce the network throughput
- It is hard to distinguish between network failure and attack
  - It is harder to detect the attack if it drops packets selectively (e.g., random packets, certain type of packets, packets from certain sources)
- It is more related to the data plane since the control plane security cannot defend against it at all.
Overview of AAI protocol

- The ack-based adversary identification (AAI) protocol can localize the packet dropping attackers
  - The source nodes identify the adversary based on the ack from the destination node and the intermediate nodes

- Achieving goals
  - High detection rate
  - Low communication overhead
  - Low storage overhead
AAI Classification

- Depending on which data packets to acknowledge and which intermediate nodes should send ack packets
  - Full-ack (strawman approach)
    - Every intermediate node sends an ack for every lost data packet
  - Probabilistic AAI-1 (PAAI-1 approach)
    - Only a subset of data packets are acknowledged
  - Probabilistic AAI-2 (PAAI-2 approach)
    - Only a subset of intermediate nodes respond
Strawman Approach

m: message, Ki: symmetric key shared by S and Fi
ai: ack of Fi <H(m)|Ami>, aD: <[H(m)]Kd>
Ami: report computed by node Fi for m <[i|Ri|Ami+1]Ki>
AmD: <[i|Ri]Kd>
q: onion report request, Ri: local report <i|H(m)|aD>

increase score for i

drop m!
**PAAI-1 Approach**

- **p**: probe frequency fixed at setup time.
- **c**: probe \( <H(m)> \)
- **ai**: ack of node \( Fi <H(m)|Ai> \), \( aD: <[H(m)]Kd> \)
- **Ai**: if no downstream ack, \( <[i|H(m)]Ki> \), else \( <[i|Ri|Ai+1]Ki> \)
- **AD**: \( <[i|Ri]Kd> \)
- **Ri**: local report \( <i|H(m)> \)

---

**Example Diagram:**

- **S** (source) sends probes to **F1**.
- **F1** to **Fi**:
  - **Fi**: if no ack, next node sends \( <[i|H(m)]Ki> \).
  - **Fi+1** receives probe and sends ack.
- **D** (destination) receives probe and sends ack.

---

With **p**:

- **Increase score for i**
- **Drop m!**

---

**Network Layout:**

- **S** (source) sends probes to **F1**.
- **F1** to **Fi**:
  - **Fi**: if no ack, next node sends \( <[i|H(m)]Ki> \).
  - **Fi+1** receives probe and sends ack.
- **D** (destination) receives probe and sends ack.

---

**Network Elements:**

- **Fi**: Intermediate node receiving a probe.
- **Fi+1**: Intermediate node sending an acknowledgment.
- **D**: Destination node receiving the acknowledgment.

---

**Network Protocols:**

- **ai**: Acknowledgment of probe from node **Fi**.
- **AD**: Acknowledgment of data from node **Fi**.
- **Ri**: Local report sent by node **Fi**.

---

**Network Flow:**

1. **S** sends probe to **F1**.
2. **F1** to **Fi**:
   - **Fi** sends probe to **Fi+1**.
   - If no acknowledgment, **Fi** sends \( <[i|H(m)]Ki> \).
3. **Fi+1** receives probe and sends acknowledgment.
4. **D** receives acknowledgment and sends acknowledgment.
PAAI-2 Approach

c : probe $<H(m) | Z>$, Z : random challenge
Ti : keyed-pseudorandom-function-based predicate $1/(d-i+1)$
ai : $<H(m) | Ai>$, AD : $<[H(m)]Kd>$
Ai : if no downstream ack, $<Eki([i | c | aD]K_i)>$, otherwise
    if Fi is sampled, overwrite with encrypted report Ai
else, $<Eki(Ai+1)>$
## Performance Comparison

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Detection Rate</th>
<th>Communication</th>
<th>Storage</th>
<th>Storage</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$\frac{\ln(\frac{2}{\epsilon})}{8\epsilon^2 \cdot (1 - \rho)^2 + d}$</td>
<td>$O(1 + \psi d)$</td>
<td>$O(2r_0 \nu)$</td>
<td>$O(r_0 \nu)$</td>
</tr>
<tr>
<td>Full-ack</td>
<td>$\frac{\ln(\frac{2}{\epsilon})}{8\epsilon^2 \cdot (1 - \rho)^2 + d}$</td>
<td>$O(p \cdot d)$</td>
<td>$O(r_0 (0.5 + p) \nu)$</td>
<td>$O(r_0 (0.5 + p) \nu)$</td>
</tr>
<tr>
<td>PAAI-1</td>
<td>$\frac{\ln(\frac{2}{\epsilon})}{8\epsilon^2 \cdot (1 - \rho)^2 + d}$</td>
<td>$O(1)$</td>
<td>$O(2r_0 \nu)$</td>
<td>$O(r_0 \nu)$</td>
</tr>
<tr>
<td>PAAI-2</td>
<td>$2^d \frac{\ln(\frac{2}{\epsilon})}{18\epsilon^2} \cdot d \cdot \log(d)$</td>
<td>$O(1)$</td>
<td>$O(2r_0 \nu)$</td>
<td>$O(r_0 \nu)$</td>
</tr>
</tbody>
</table>

### Detection Rate (minutes)

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Detection Rate bound</th>
<th>Detection Rate average</th>
<th>Storage bound</th>
<th>Storage average</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full-ack</td>
<td>0.25</td>
<td>0.17</td>
<td>12</td>
<td>3.2</td>
</tr>
<tr>
<td>PAAI-1</td>
<td>9</td>
<td>4.2</td>
<td>3.2</td>
<td>3.0</td>
</tr>
<tr>
<td>PAAI-2</td>
<td>100</td>
<td>50</td>
<td>12</td>
<td>6.4</td>
</tr>
</tbody>
</table>
Combinations

- Combination 1 (full ack + PAAI-1)
  - Every node much acknowledge a selected fraction of lost data packets
  - Same detection rate, less communication overhead, more storage overhead

- Combination 2 (PAAI-1 + PAAI-2)
  - One selected node acknowledges a selected fraction of data packets
  - Less communication overhead, low detection rate
Fair Assumptions?

- Secure distribution of symmetric keys
- Symmetric routing path?
- Network throughput
  - For 100 packets per second, PAAI-1 converges in 8 minutes
Questions
Packet “Deflections”

- ISPs offer users alternate paths (deflections) in addition to the normal path advertised via BGP.

A, B, C, D, F is normal BGP path for A -> F. To avoid D, A could request that C deflects packets to E, yielding path A, B, C, E, F.