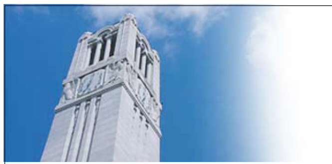


NC STATE UNIVERSITY Computer Science

CSC 774 Network Security

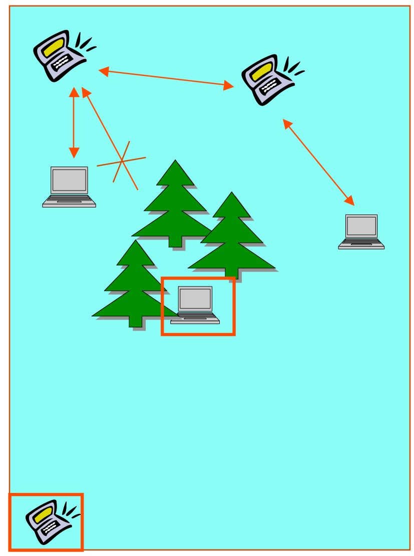
Topic 8. Security in Mobile Ad-Hoc Networks

Mobile Ad-hoc Networks (MANET)

- MANET
  - No infrastructure support
  - Wireless communication
  - Mobile
- Applications for MANET
  - Rescue missions
  - Scientific explorations
  - Military operations

Security in MANET

- Challenges
  - No physical boundary
  - No fixed topology
    - Nowhere to place firewalls
  - Unreliable communication



NC STATE UNIVERSITY Computer Science

CSC 774 Network Security

Topic 8.1 Secure MANET Routing Protocols

-- Ariadne

Outline

- Dynamic Source Routing (DSR) protocol
  - Basis of Ariadne
- Attacks against MANET routing protocols
- Ariadne

DSR

- Two phases
  - Route discovery
    - Discover routes from the initiator to the target
  - Route maintenance
    - Maintain route

DSR (Cont'd)

- Route discovery
  - Initiator broadcasts REQUEST to its neighbors
    - Each REQUEST is uniquely identified by the initiator's address and a sequence number
  - If a node receives this REQUEST
    - If it has received this REQUEST previously, discard it
    - Otherwise, append itself to a list in this REQUEST and rebroadcast it
  - When the destination receives this REQUEST
    - Sends back a REPLY, including the list of nodes in the REQUEST
  - When the initiator receives the REPLY
    - Cache the list of nodes in REPLY

DSR (Cont'd)

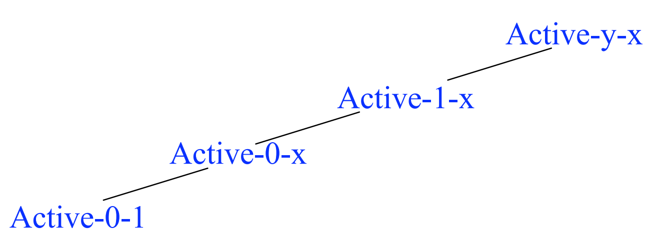
- Route maintenance
  - Source routing
    - When sending a packet, the originator includes the complete route in the packet header
    - Each node forwards the packet based on the route in the header
  - Route error
    - When a node detects a link break, it sends a ROUTE ERROR message to the originator, identifying the broken link
    - The originator removes the route from its cache, uses an alternative route or rediscover a new route

Attacks against MANET Routing Protocols

- Type of attacks
  - Passive
    - Eavesdrop
  - Active
    - Inject packets into the network
- Compromised nodes
  - Attackers may know all keying materials on compromised nodes

Attacks against MANET Routing Protocols (Cont'd)

- Active-n-m attacker
  - The attacker has compromised n nodes and owns m nodes
- Attacker hierarchy
  - Measure the security of the routing protocol
  - Ariadne: resilience against Active-1-x and Active-y-x attacks



Attacks against MANET Routing Protocols (Cont'd)

- Active attacks on ad hoc routing protocols
  - Routing disruption
    - Routing loop
    - Black hole (gray hole)
    - Wormhole
    - Rushing attack
  - Resource consumption
    - Packet injection

Ariadne

- Overview
  - Based on DSR
  - Use
    - Pairwise shared secret keys
    - TESLA
    - Digital signature
      - Undesirable due to the overhead and possible DOS attacks
  - Main target
    - Active-y-x attackers

Ariadne (Cont'd)

- Notations
  - *A*, *B* : principals
  - K_{AB}, K_{BA} : secret MAC keys between *A* and *B*
  - MAC (*M*) : MAC of message *M* using MAC key K_{AB}

Route Discovery

- Goals
  - Target can authenticate initiator
    - Initiator includes a MAC computed with K_{SD}
  - Initiator and target can authenticate the node list in REPLY and REQUEST
  - No intermediate node can remove a previous node's entry in the node list

Route Discovery (Cont'd)

- Authenticate node list in REQUEST
  - With TESLA
    - Each node has a TESLA instance
    - An intermediate node appends a MAC computed with its TESLA key
  - With digital signature
    - Each node digitally signs REQUEST
  - With pairwise key
    - Each node appends a MAC computed with a pairwise key shared with the target
- Authentication of node list in REPLY is done similarly

Route Discovery (Cont'd)

- Per-hop hashing
  - Prevent intermediate node from removing entries from the node list in REQUEST or REPLY
  - The source initializes the *hash chain* field to a MAC generated with a pairwise key shared between the source and the target
  - Each node A updates the *hash chain* field with *H*[A, *hash chain*]

Route Discovery with TESLA

- Route Request
  - <RREQ, *initiator*, *target*, *id*, *time interval*, *hash chain*, *node_list*, *MAC_list*>
  - Initiator initializes *hash chain* to MAC_{KSD}(*initiator*, *target*, *id*, *time interval*)
  - Intermediate node *A* which receives the request checks <*initiator*, *id*> and checks *time interval*
    - *Time interval* : TESLA time interval at the pessimistic expected arrival time of the RREQ at target (say $T + 2d$)
    - If any condition fails, discard the request

Route Discovery with TESLA (Cont'd)

    - If all conditions hold, *A* appends its address to *node list*, replaces *hash chain* with *H*[*A*, *hash chain*], appends MAC of entire Request with TESLA key K_{A_i} to *MAC list*
  - Target checks validity of Request
    - the TESLA keys are not disclosed yet
    - the *hash chain* is equal to $H[n_n, H[n_{n-1}, H[\ldots, H[n_1, MAC_{K_{SD}}(initiator, target, id, interval)]\ldots]$
    - If Request is valid, target returns a Route Reply

Route Discovery with TESLA (contd.)

- Route Reply
  - <RREP, *target*, *initiator*, *time interval*, *node list*, *MAC list*, *target MAC*, *key list*>
  - Packet is sent to initiator along the route in *node list*
  - Forwarding node waits until it can disclose its key and then append its key
  - Initiator verifies that
    - Each key is valid (TESLA security condition)
    - *target MAC* is valid (based on K_{DC} shared with target)
    - Each MAC in *MAC list* is valid (based on TESLA keys)

Example

$$
\begin{array}{ll}
S: & h_0 = \text{MAC}_{K_{SD}}(\text{REQUEST}, S, D, id, ti) \\
S \to *: & \langle \text{REQUEST}, S, D, id, ti, h_0, (), () \rangle \\
 & \\
A: & h_1 = H[A, h_0] \\
 & M_A = \text{MAC}_{K_{A_{ti}}}(\text{REQUEST}, S, D, id, ti, h_1, (A), ()) \\
A \to *: & \langle \text{REQUEST}, S, D, id, ti, \underline{\boldsymbol{h_1}}, (\underline{\boldsymbol{A}}), (\underline{\boldsymbol{M_A}}) \rangle \\
 & \\
B: & h_2 = H[B, h_1] \\
 & M_B = \text{MAC}_{K_{B_{ti}}}(\text{REQUEST}, S, D, id, ti, h_2, (A, B), (M_A)) \\
B \to *: & \langle \text{REQUEST}, S, D, id, ti, \underline{\boldsymbol{h_2}}, (A, \underline{\boldsymbol{B}}), (M_A, \underline{\boldsymbol{M_B}}) \rangle \\
 & \\
C: & h_3 = H[C, h_2] \\
 & M_C = \text{MAC}_{K_{C_{ti}}}(\text{REQUEST}, S, D, id, ti, h_3, (A, B, C), (M_A, M_B)) \\
C \to *: & \langle \text{REQUEST}, S, D, id, ti, \underline{\boldsymbol{h_3}}, (A, B, \underline{\boldsymbol{C}}), (M_A, M_B, \underline{\boldsymbol{M_C}}) \rangle \\
 & \\
D: & M_D = \text{MAC}_{K_{DS}}(\text{REPLY}, D, S, ti, (A, B, C), (M_A, M_B, M_C)) \\
D \to C: & \langle \text{REPLY}, D, S, ti, (A, B, C), (M_A, M_B, M_C), \underline{\boldsymbol{M_D}}, () \rangle \\
 & \\
C \to B: & \langle \text{REPLY}, D, S, ti, (A, B, C), (M_A, M_B, M_C), M_D, (\underline{\boldsymbol{K_{C_{ti}}}}) \rangle \\
 & \\
B \to A: & \langle \text{REPLY}, D, S, ti, (A, B, C), (M_A, M_B, M_C), M_D, (K_{C_{ti}}, \underline{\boldsymbol{K_{B_{ti}}}}) \rangle \\
 & \\
A \to S: & \langle \text{REPLY}, D, S, ti, (A, B, C), (M_A, M_B, M_C), M_D, \\
 & \quad (K_{C_{ti}}, K_{B_{ti}}, \underline{\boldsymbol{K_{A_{ti}}}}) \rangle
\end{array}
$$

Route Maintenance

- Security issue
  - Prevent unauthorized nodes from sending (bogus) ROUTE_ERRORs

Route Maintenance (Cont'd)

- Route Error
  - <ROUTE_ERROR, *sending address*, *receiving address*, *time interval*, *error MAC*, *recent TESLA key*> source routed back to initiator
  - Intermediate node
    - Forwards the packet and searches its route cache for all routes that use <*sending address*, *receiving address*>
    - If exists, checks validity of *time interval*
    - If valid, checks authentication of the Error
    - Until authentication, saves Error info in memory until a key is disclosed and uses routes in route cache
    - If authenticated, removes all such routes

Thwarting Routing Misbehaviors

- What if intermediate nodes in the source route don't forward packets?
  - ➢ A feedback based reputation scheme to detect misbehaving nodes - relies on feedback about which packets were successfully delivered
  - ➢ A node with multiple routes sends a fraction along each route and sends packets along the successful route
  - ➢ Malicious node avoidance in Route Discovery
    - Route Request includes a list of malicious nodes to avoid and the MAC h_0 computed over that list – no details
    - Malicious nodes can be detected by target

Thwarting Malicious REQUEST floods

- Since REQUEST are authenticated at the target, and not at every hop, attacker can flood malicious REQUESTs
- *Route Discovery chains*
  - ➢ To weakly authenticate REQUESTs instantly
  - ➢ One-way chains generated as $K_i = H^{N-i} [K_N]$

  1. Release one key for each Discovery (rate-limiting requests)
     - ➢ A node not partitioned from the network can prevent an attacker from reusing keys
  2. Dictate schedule for disclosure of key + loose clock synchronization
     - ➢ Any node can prevent an attacker from reusing keys
     - ➢ Computationally slightly more expensive

An Optimization

- Observation
  - Only the initiator can use the discovered route, since the intermediate nodes cannot authenticate the target
  - Would be more efficient if the intermediate nodes can also use the discovered route
- Optimization
  - Target uses TESLA key to generate the MAC
  - The key is released by target, to initiator, after appropriate delay, which the intermediate nodes also use.