

Secure routing in ad hoc networks

Security Protocols (bmevihim132)

Dr. Levente Buttyán
associate professor
BME Hálózati Rendszerek és Szolgáltatások Tanszék
Lab of Cryptography and System Security (CrySyS)
buttyan@hit.bme.hu, buttyan@crysys.hu

Outline

- ad hoc network routing protocols
- attacks on routing protocols
- secured routing protocols
- wormhole detection

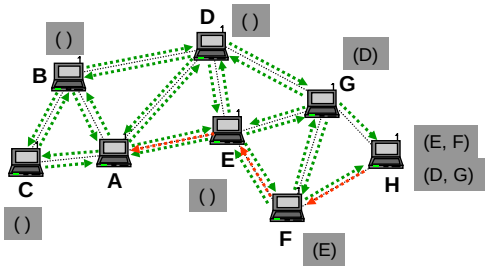
Ad hoc network routing protocols

- topology-based protocols
  - proactive
    - distance vector based (e.g., DSDV)
    - link-state (e.g., OLSR)
  - reactive (on-demand)
    - distance vector based (e.g., AODV)
    - source routing (e.g., DSR)
- position-based protocols
    - greedy forwarding (e.g., GPSR, GOAFR)
    - restricted directional flooding (e.g., DREAM, LAR)
- hybrid approaches

Dynamic Source Routing (DSR)

- on-demand source routing protocol
- two components:
  - route discovery
    - used only when source S attempts to send a packet to destination D
    - based on flooding of Route Requests (RREQ) and returning Route Replies (RREP)
  - route maintenance
    - makes S able to detect route errors (e.g., if a link along that route no longer works)

DSR Route Discovery illustrated



A → *: [RREQ, id, A, H; ()]
B → *: [RREQ, id, A, H; (B)]
C → *: [RREQ, id, A, H; (C)]
D → *: [RREQ, id, A, H; (D)]
E → *: [RREQ, id, A, H; (E)]
F → *: [RREQ, id, A, H; (E, F)]
G → *: [RREQ, id, A, H; (D,G)]

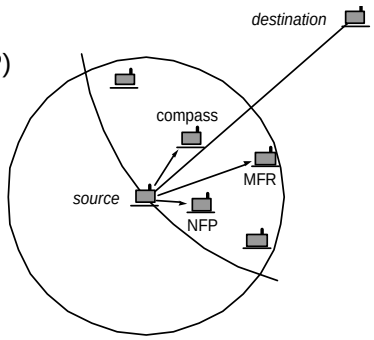
H → A: [RREP, <source route>; (E, F)]

where <source route> is obtained

- from the route cache of H
- by reversing the route received in the RREQ
  - works only if all the links along the discovered route are bidirectional
  - IEEE 802.11 assumes that links are bidirectional
- by executing a route discovery from H to A
  - discovered route from A to H is piggy backed to avoid infinite recursion

Position-based greedy forwarding

- assumptions
  - nodes are aware of their own positions and that of their neighbors
  - packet header contains the position of the destination
- packet is forwarded to a neighbor that is closer to the destination than the forwarding node
  - Most Forward within Radius (MFR)
  - Nearest with Forward Progress (NFP)
  - Compass forwarding
  - Random forwarding
- additional mechanisms are needed to cope with local minimums (dead-ends)



Attacks on routing protocols (1/2)

- general objectives of attacks
  - increase adversarial control over the communications between some nodes;
  - degrade the quality of the service provided by the network;
  - increase the resource consumption of some nodes (e.g., CPU, memory, or energy).
- adversary model
  - insider adversary
    - can corrupt legitimate nodes
  - the attacker is not all-powerful
    - it is not physically present everywhere
    - it launches attacks from regular devices

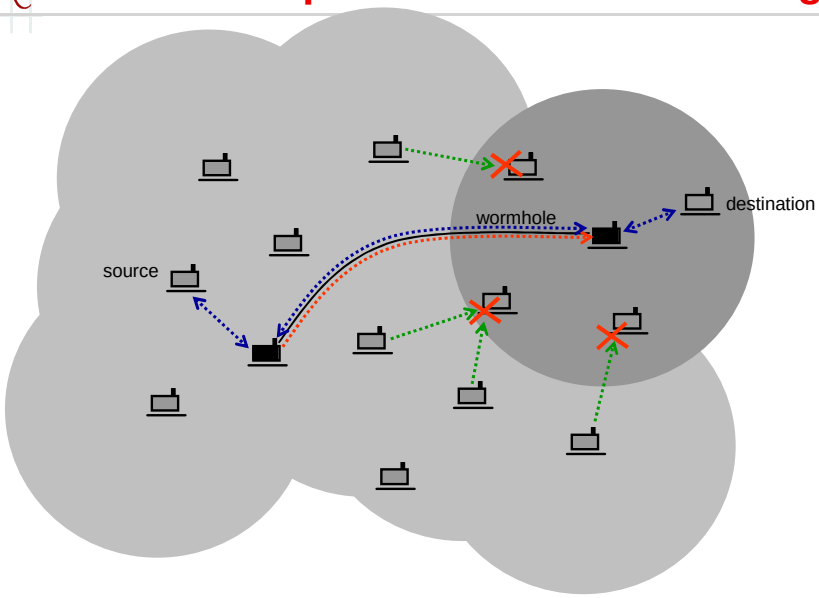
Attacks on routing protocols (2/2)

- attack mechanisms
  - eavesdropping, replaying, modifying, and deleting control packets
  - fabricating control packets containing fake routing information (forgery)
  - fabricating control packets under a fake identity (spoofing)
  - dropping data packets (attack against the forwarding function)
  - wormholes and tunneling
  - rushing
- types of attacks
  - route disruption
  - route diversion
  - creation of incorrect routing state
  - generation of extra control traffic
  - creation of a gray hole

Route disruption

- the adversary prevents a route from being discovered between two nodes that are otherwise connected
- the primary objective of this attack is to degrade the quality of service provided by the network
  - the two victims cannot communicate, and
  - other nodes can also suffer and be coerced to use suboptimal routes
- attack mechanisms that can be used to mount this attack:
  - dropping route request or route reply messages on a vertex cut
  - forging route error messages
  - combining wormhole/tunneling and control packet dropping
  - rushing

Route disruption in DSR with rushing



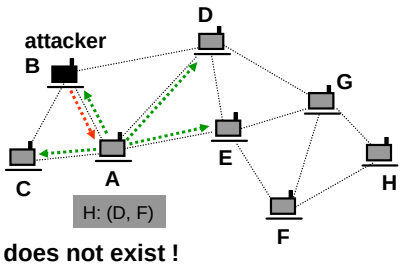
Route diversion

- due to the presence of the adversary, the protocol establishes routes that are different from those that it would establish, if the adversary did not interfere with the execution of the protocol
- the objective of route diversion can be
  - to increase adversarial control over the communications between some victim nodes
    - the adversary tries to achieve that the diverted routes contain one of the nodes that it controls or a link that it can observe
    - the adversary can eavesdrop or modify data sent between the victim nodes easier
  - to increase the resource consumption of some nodes
    - many routes are diverted towards a victim that becomes overloaded
  - degrade quality of service
    - by increasing the length of the discovered routes, and thereby, increasing the end-to-end delay between some nodes
- route diversion can be achieved by
  - forging or manipulating routing control messages
  - dropping routing control messages
  - setting up a wormhole/tunnel

Creation of incorrect routing state

- this attack aims at jeopardizing the routing state in some nodes so that the state appears to be correct but, in fact, it is not
  - data packets routed using that state will never reach their destinations
- the objective of creating incorrect routing state is
  - to increase the resource consumption of some nodes
    - the victims will use their incorrect state to forward data packets, until they learn that something goes wrong
  - to degrade the quality of service
- can be achieved by
  - spoofing, forging, modifying, or dropping control packets

Incorrect routing state in DSR



Route (A, D, F, H) does not exist !

A → *: [RREQ, id, A, H; ()]
D(B) → A: [RREP, <src route>, A, H; (D, F)]

Generation of extra control traffic

- injecting spoofed control packets into the network
- aiming at increasing resource consumption due to the fact that such control packets are often flooded in the entire network

Setting up a gray hole

- an adversarial node selectively drops data packets that it should forward
- the objective is
  - to degrade the quality of service
    - packet delivery ratio between some nodes can decrease considerably
  - to increase resource consumption
    - wasting the resources of those nodes that forward the data packets that are finally dropped by the adversary
- implementation is trivial
  - adversarial node participates in the route establishment
  - when it receives data packets for forwarding, it drops them
  - even better if combined with wormhole/tunneling

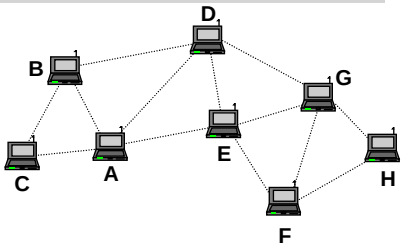
Some secured routing protocols

- SRP (on-demand source routing)
- Ariadne (on-demand source routing)
- endairA (on-demand source routing)
- S-AODV (on-demand distance vector routing)
- ARAN (on-demand, routing metric is the propagation delay)
- SEAD (proactive distance vector routing)
- SMT (multi-path routing combined error correcting)
- Watchdog and Pathrater (implementation of the “detect and react” approach to defend against gray holes)
- ODSBR (source routing with gray hole detection)

Ariadne

- Ariadne is a secured variant of DSR
- it uses control message authentication to prevent modification and forgery of routing messages
  - based on signatures, MACs, or TESLA
- it uses a per-hop hash mechanism to prevent the manipulation of the accumulated route information in the route request message

Ariadne with signatures illustrated



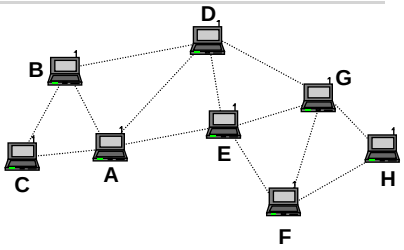
A : $h_A = mac_{AH}(RREQ \mid A \mid H \mid id)$
A → * : [RREQ, A, H, id, h_A, (), ()]

E : $h_E = H(E \mid h_A)$
E → * : [RREQ, A, H, id, h_E, (E), (sig_E)]

F : $h_F = H(F \mid h_E)$
F → * : [RREQ, A, H, id, h_F, (E, F), (sig_E, sig_F)]

H → A: [RREP, H, A, (E, F), (sig_E, sig_F), sig_H]

Ariadne with MACs illustrated



A : $h_A = mac_{AH}(RREQ \mid A \mid H \mid id)$
A → * : [RREQ, A, H, id, h_A, (), ()]

E : $h_E = H(E \mid h_A)$
E → * : [RREQ, A, H, id, h_E, (E), (mac_{EH})]

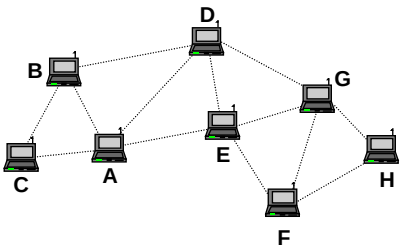
F : $h_F = H(F \mid h_E)$
F → * : [RREQ, A, H, id, h_F, (E, F), (mac_{EH}, mac_{FH})]

H → A : [RREP, H, A, (E, F), mac_{HA}]

Ariadne with TESLA

- assumptions:
  - each source-destination pair (S, D) shares a symmetric key K_{SD}
  - each node F has a TESLA key chain $K_{F,i}$
  - each node knows an authentic TESLA key of every other node
- route request (source S, destination D):
  - S authenticates the request with a MAC using K_{SD}
  - each intermediate node F appends a MAC computed with its current TESLA key
  - D verifies the MAC of S
  - D verifies that the TESLA key used by F to generate its MAC has not been disclosed yet
- route reply:
  - D generates a MAC using K_{SD}
  - each intermediate node delays the reply until it can disclose its TESLA key that was used to generate its MAC
  - F appends its TESLA key to the reply
  - S verifies the MAC of D, and all the MACs of the intermediate nodes

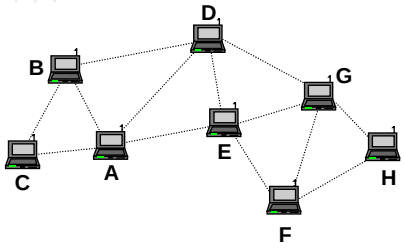
Ariadne with TESLA illustrated



A → *: [RREQ, A, H, id, h_A, (), ()]
E → *: [RREQ, A, H, id, h_E, (E), ($mac_{K_{E,i}}$)]
F → *: [RREQ, A, H, id, h_F, (E, F), ($mac_{K_{E,i}}$, $mac_{K_{F,i}}$)]

H → F: [RREP, H, A, (E, F), ($mac_{K_{E,i}}$, $mac_{K_{F,i}}$), mac_{HA}, ()]
F → E: [RREP, H, A, (E, F), ($mac_{K_{E,i}}$, $mac_{K_{F,i}}$), mac_{HA}, ($K_{F,i}$)]
E → A: [RREP, H, A, (E, F), ($mac_{K_{E,i}}$, $mac_{K_{F,i}}$), $mac_{K_{HA}}$, ($K_{F,i}$, $K_{E,i}$)]

endairA



target verifies:
- there's no repeating ID in the node list
- last node in the node list is a neighbor

each intermediate node verifies:
- its own ID is in the node list
- there's no repeating ID in the node list
- next and previous nodes in the node list are neighbors
- all signatures are valid

source verifies:
- there's no repeating ID in the node list
- first node in the node list is a neighbor
- all signatures are valid

A → * : [RREQ, A, H, id, ()]
E → * : [RREQ, A, H, id, (E)]
F → * : [RREQ, A, H, id, (E, F)]

H → F : [RREP, A, H, id, (E, F), (sig_H)]
F → E : [RREP, A, H, id, (E, F), (sig_H, sig_F)]
E → A : [RREP, A, H, id, (E, F), (sig_H, sig_F, sig_E)]

Properties of endairA

- security
  - endairA is provably secure if
    - the signature scheme is secure against chosen message attacks
    - adversarial nodes cannot covertly communicate with each other
- efficiency
  - endairA requires less computation
    - route reply is signed and verified only by the nodes on the route
    - in Ariadne, route request is signed (and potentially verified) by every node in the network

Combating gray holes

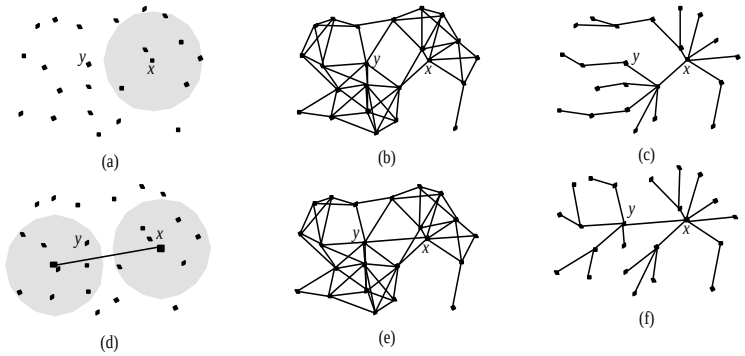
- use multiple, preferably disjoint routes
  - increased robustness
  - but also increased resource consumption
  - resource consumption can be somewhat decreased by applying the principles of error correcting coding
    - data packet is coded and the coded packet is split into smaller chunks
    - a threshold number of chunks is sufficient to reconstruct the entire packet
    - chunks are sent over different routes
- detect and react
  - monitor neighbors and/or traffic and identify misbehaving nodes/links
  - use routes that avoid those misbehaving nodes/links
  - reputation reports about nodes can be spread in the network
  - this approach has several problems
    - how to detect reliably that a node is misbehaving?
    - how to prevent false accusations and spreading of negative reputations?

What is a wormhole?

- a wormhole is an out-of-band connection, controlled by the adversary, between two physical locations in the network
  - the adversary installs radio transceivers at both ends of the wormhole
  - it transfers packets (possibly selectively) received from the network at one end of the wormhole to the other end via the out-of-band connection, and re-injects the packets there into the network
- notes:
  - the adversary's transceivers are not regular nodes (no node is compromised by the adversary)
  - adversary doesn't need to understand what it tunnels (e.g., encrypted packets can also be tunneled through the wormhole)
  - it is easy to mount a wormhole, but it may have devastating effects on routing

Effects of a wormhole

- at the data link layer: distorted network topology



- at the network layer:
  - routing protocols may choose routes that contain wormhole links
    - typically those routes appear to be shorter
    - flooding based routing protocols (e.g., DSR, Ariadne) may not be able to discover other routes but only through the wormhole
  - adversary can then monitor traffic or drop packets (DoS)

Classification of detection methods

- centralized mechanisms
  - data collected from the local neighborhood of every node are sent to a central entity
  - based on the received data, a model of the entire network is constructed
  - the central entity tries to detect inconsistencies (potential indicators of wormholes) in this model
  - can be used in sensor networks, where the base station can play the role of the central entity
- decentralized mechanisms
  - each node constructs a model of its own neighborhood using locally collected data
  - each node tries to detect inconsistencies on its own
  - advantage: no need for a central entity (fits well some applications)
  - disadvantage: nodes need to be more complex

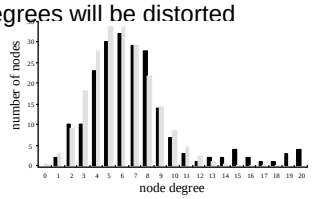
Statistical wormhole detection

- each node reports its list of believed neighbors to the base station
- the base station reconstructs the connectivity graph (model)
- *a wormhole always increases the number of edges* in the connectivity graph
- this increase may change the properties of the connectivity graph in a detectable way (anomaly)
- detection can be based on statistical hypothesis testing methods (e.g. the χ^2-test)

Examples

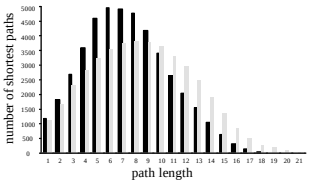
- a wormhole that creates many new edges may increase the *number of neighbors* of the affected nodes

→ distribution of node degrees will be distorted



- a wormhole is usually a shortcut that decreases the length of the shortest paths in the network

→ distribution of the length of the shortest paths will be distorted

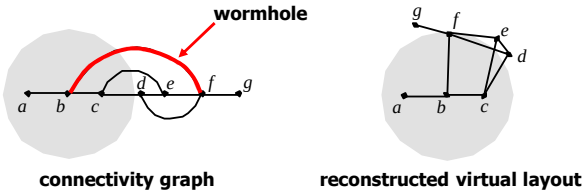


Multi-dimensional scaling

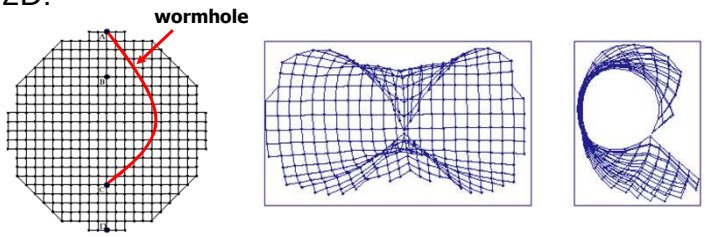
- the nodes not only report their lists of neighbors, but they also estimate (inaccurately) their distances to their neighbors
- connectivity information and estimated distances are input to a multi-dimensional scaling (MDS) algorithm
- the MDS algorithm tries to determine the possible position of each node in such a way that the constraints induced by the connectivity and the distance estimation data are respected
  - the algorithm has a certain level of freedom in “stretching” the nodes within the error bounds of the distance estimation
- let us suppose that an adversary installed a wormhole in the network
  - if the estimated distances between the affected nodes are much larger than the nodes’ communication range, then the wormhole is detected
  - hence, the adversary must also falsify the distance estimation → distances between far-away nodes become smaller
  - this will result in a distortion in the virtual layout constructed by the MDS algorithm

Examples

- in 1D:



- in 2D:



Packet leashes

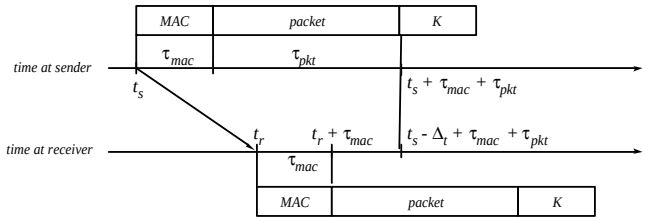
- packet leashes ensure that packets are not accepted “too far” from their source
- geographical leashes
  - each node is equipped with a GPS receiver
  - when sending a packet, the node puts its GPS position into the header
  - the receiving node verifies if the sender is really within communication range
- temporal leashes
  - nodes’ clocks are very tightly synchronized
  - when sending a packet, the node puts a timestamp in the header
  - the receiving node estimates the distance of the sender based on the elapsed time and the speed of light

$$d_{est} < v_{light}(t_{rcv} - t_{snd} + \Delta_t)$$

  - note: $v_{light}\,\Delta_t$ must be much smaller than the communication range

TESLA with Instant Key-disclosure

idea: authentication delay of TESLA can be removed in an environment where the nodes' clocks are tightly synchronized



- by the time the sender reveals the key, the receiver has already received the MAC
- security condition: $t_r < t_s - \Delta_t + \tau_{pkt}$
- note: Δ_t must be very small or otherwise packets must be very long

Mutual Auth with Distance-bounding

u		v
— initialization phase —		
generate random numbers $r \in \{0,1\}^\ell$, $r' \in \{0,1\}^{\ell'}$		generate random numbers $s \in \{0,1\}^\ell$, $s' \in \{0,1\}^{\ell'}$
compute commitment $c_u = H(r\|r')$		compute commitment $c_v = H(s\|s')$
	$\xrightarrow{c_u}$	
	$\xleftarrow{c_v}$	
— distance-bounding phase —		
the bits of r are $r_1, r_2, \ldots, r_\ell$		the bits of s are $s_1, s_2, \ldots, s_\ell$
$\alpha_1 = r_1$	$\xrightarrow{\alpha_1}$	
	$\xleftarrow{\beta_1}$	$\beta_1 = s_1 \oplus \alpha_1$
	...	
$\alpha_i = r_i \oplus \beta_{i-1}$	$\xrightarrow{\alpha_i}$	measure delay between β_{i-1} and α_i
measure delay between α_i and β_i	$\xleftarrow{\beta_i}$	$\beta_i = s_i \oplus \alpha_i$
	...	
$\alpha_\ell = r_\ell \oplus \beta_{\ell-1}$	$\xrightarrow{\alpha_\ell}$	measure delay between $\beta_{\ell-1}$ and α_ℓ
measure delay between α_ℓ and β_ℓ	$\xleftarrow{\beta_\ell}$	$\beta_\ell = s_\ell \oplus \alpha_\ell$
— authentication phase —		
$s_i = \alpha_i \oplus \beta_i$ $(i = 1, \ldots, \ell)$		$r_1 = \alpha_1$ and $r_i = \alpha_i \oplus \beta_{i-1}$ $(i = 2, \ldots, \ell)$
$\mu_u = mac_{k_{uv}}(u\|v\|r_1\|s_1\|\ldots\|r_\ell\|s_\ell)$		$\mu_v = mac_{k_{uv}}(v\|u\|s_1\|r_1\|\ldots\|s_\ell\|r_\ell)$
	$\xrightarrow{r'\|\mu_u}$	
	$\xleftarrow{s'\|\mu_v}$	
verify c_v and μ_v		verify c_u and μ_u

- MAD allows precise distance estimation without synchronized clocks

Summary

- routing is a fundamental function in networking, hence, an ideal target for attacks
- attacks against routing aim at
  - increasing adversarial control over the communications between some nodes;
  - degrading the quality of the service provided by the network;
  - increasing the resource consumption of some nodes (e.g., CPU, memory, or energy)
- many attacks (but not all!) can be prevented by authenticating routing control messages
  - however, it is difficult to protect the mutable parts of control messages
- several secured ad hoc network routing protocols have been proposed
  - we discussed Ariadne and endairA
- wormhole detection is still an active research area
  - centralized and decentralized approaches have been proposed