Specification-based Intrusion Detection

Michael May

CIS-700

Fall 2004

Overview

- Mobile ad hoc networking (MANET) new area of protocols
- Some old networking solutions work (TCP/IP) but things change with open medium of wireless
- Goal: Define a system specification (model) and detect when behavior differs from expected

Two detection approaches

Specification

- Hand made model of states and transitions
- Detect when
  - A node moves to an illegal state
  - A node makes an illegal transition (input missing)
  - A node transitions without proper output
  - Messages sent don't follow expected model
- No false positives

Statistical

- Can find attacks where state is not violated
  - Flooding
  - Dropping
  - Partitioning
- Train on normal runs and attack runs
- Run model over test data and detect attacks
- Can detect new attacks

Two detection approaches

Specification

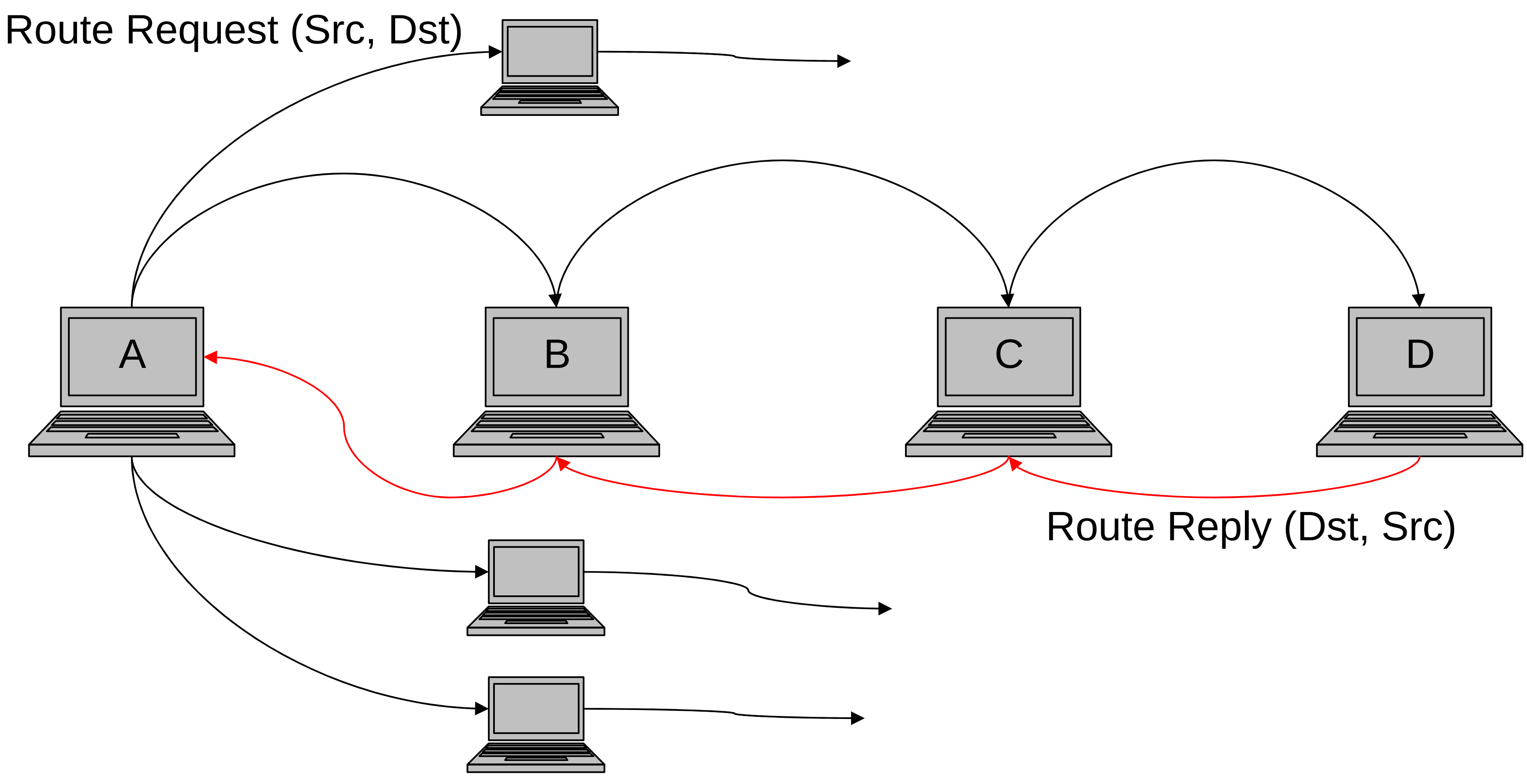
- Can't detect attacks that are not violations in the specification
- Only as good as the model used
  - Can't catch attacks at a level of the system not in the model

Statistical

- Can't find attacks that look like normal behavior
- Subtle attacks have higher false positives

Use both to achieve greatest effectiveness

MANET routing process



Basic (Routing) Events

- Identify the smallest transactions that occur MANET routing
  - Smaller atomic actions occur, but these must be done as transactions

1. Source node sends Route Request
2. Nodes on the path receive and forward
3. Replying node receives Request and sends Route Reply
4. Nodes on the path receive and forward
5. Source node receives Reply and establishes route

- Anomalous basic event is one that doesn’t follow the system specification

Taxonomy of anomalous basic events

Compromises to Security Goals		Events by Targets		
		Routing Messages	Data Packets	Routing Table Entries
Confidentiality		Location Disclosure	Data Disclosure	N/A
Integrity	Add	**Fabrication***	Fabrication	**Add Route**
	Delete	**Interruption**	**Interruption**	**Delete Route**
	Change	**Modification***	Modification	**Change Route Cost**
		Rushing		
Availability		**Flooding**	**Flooding**	Routing Table Overflow

- Bold indicates intrusion detection should work
- Asterisk indicates cryptography can work too
  - Could encrypt routing table edits, but it’s expensive

Case Study: Ad hoc On-Demand Distance Vector (AODV) Routing

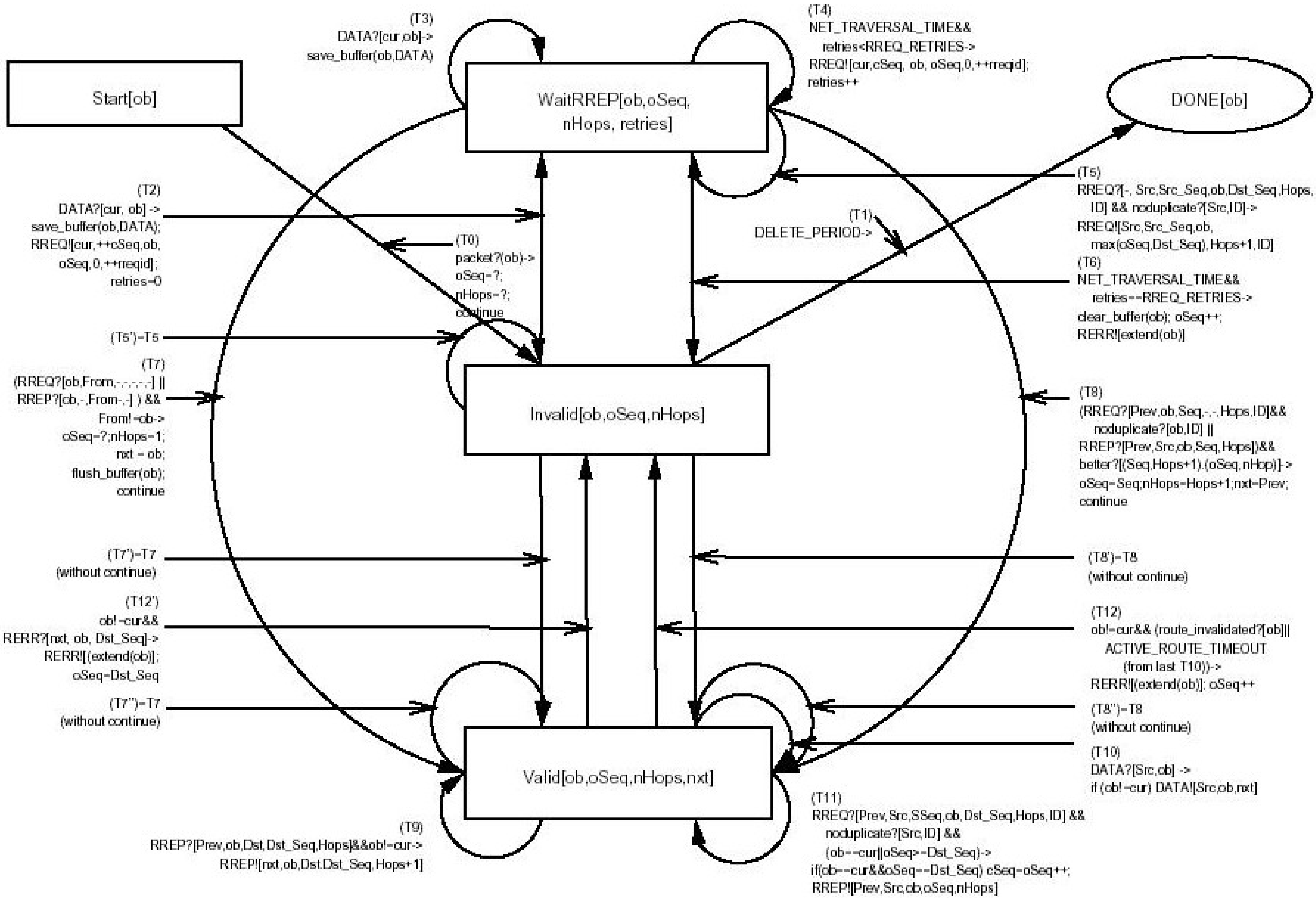
- Routing protocol for MANET using source and destination names and sequence numbers
- Nodes keep local sequence number for all messages
- Routes kept in routing table only when active
- Node discovers a route when it sends a Route Request (RREQ) and receives a Route Reply (RREP)
  - Nodes on the path watch the RREQ and RREP messages coming in and discover neighbors and paths

Two AODV Specification based solutions

- Node oriented
  - Huang and Lee ’04
- Message oriented
  - Tseng, et al ‘03

An EFSA for AODV: Node Based

- Each node maintains an EFSA with the status of every other node in the system
  - Removes non-determinism by letting multiple EFSAs process each event
  - Delete old or unused EFSAs as routes to a node expire
- Small number of states (8)
- Transitions generalized and can have both input and output
  - $\delta = \{S_{old} \rightarrow S_{new}\ ,\ input_{cond} \rightarrow output_{action}\}$
  - Events that have no input (i.e. timeouts) are treated as inputs
  - State variable assignment, packet delivery, tasks are all outputs


Start[ob]
WaitRREP[ob,oSeq, nHops, retries]
DONE[ob]
Invalid[ob,oSeq,nHops]
Valid[ob,oSeq,nHops,nxt]
(T3) DATA?[cur,ob]-> save_buffer(ob,DATA)
(T4) NET_TRAVERSAL_TIME&& retries<RREQ_RETRIES-> RREQ![cur,cSeq, ob, oSeq,0,++rreqid]; retries++
(T2) DATA?[cur, ob] -> save_buffer(ob,DATA); RREQ![cur,++cSeq,ob, oSeq,0,++rreqid]; retries=0
(T0) packet?(ob)-> oSeq=?; nHops=?; continue
(T1) DELETE_PERIOD->
(T5) RREQ?[-, Src,Src_Seq,ob,Dst_Seq,Hops, ID] && noduplicate?[Src,ID]-> RREQ![Src,Src_Seq,ob, max(oSeq,Dst_Seq),Hops+1,ID]
(T6) NET_TRAVERSAL_TIME&& retries==RREQ_RETRIES-> clear_buffer(ob); oSeq++; RERR![extend(ob)]
(T5')=T5
(T7) (RREQ?[ob,From,-,-,-,-,-] || RREP?[ob,-,From,-,-]) && From!=ob-> oSeq=?;nHops=1; nxt = ob; flush_buffer(ob); continue
(T8) (RREQ?[Prev,ob,Seq,-,-,Hops,ID]&& noduplicate?[ob,ID] || RREP?[Prev,Src,ob,Seq,Hops])&& better?[(Seq,Hops+1).(oSeq,nHop)]-> oSeq=Seq;nHops=Hops+1;nxt=Prev; continue
(T7')=T7 (without continue)
(T8')=T8 (without continue)
(T12') ob!=cur&& RERR?[nxt, ob, Dst_Seq]-> RERR![(extend(ob)]; oSeq=Dst_Seq
(T12) ob!=cur&& (route_invalidated?[ob]|| ACTIVE_ROUTE_TIMEOUT (from last T10))-> RERR![(extend(ob)]; oSeq++
(T7'')=T7 (without continue)
(T8'')=T8 (without continue)
(T10) DATA?[Src,ob] -> if (ob!=cur) DATA![Src,ob,nxt]
(T9) RREP?[Prev,ob,Dst,Dst_Seq,Hops]&&ob!=cur-> RREP![nxt,ob,Dst.Dst_Seq,Hops+1]
(T11) RREQ?[Prev, Src, SSeq,ob, Dst_Seq,Hops, ID] && noduplicate?[Src,ID] && (ob==cur||oSeq>=Dst_Seq)-> if(ob==cur&&oSeq==Dst_Seq) cSeq=oSeq++; RREP![Prev,Src,ob,oSeq,nHops]

Designing an IDS for AODV

- Intrusion detection system (IDS) will check two ways
  - Specification Violations
  - Statistical Deviations

Detecting Specification Violations

- Invalid State Violation
  - Changes in sequence numbers or hop counts in the routing tables
- Incorrect Transition Violation
  - Add Route or Routing Table Entries (without going through correct state)
  - Delete Route or Routing Table Entries
  - Fabrication of routing messages
- Unexpected Action Violation
  - Interruption of routing or data messages

Detecting Statistical Deviations

- Attacks that don’t lead to specification violations
- Flooding data packets
- Flooding routing messages
- Modification of routing messages
  - Restricted to sequence number modification
- Rushing of routing messages
  - Discovery fails due to Route Request retries running out or timeout
  - Frequency of transitioning from Route Request to Route Reply message

Testing

- IDS system on each node watches packets in and out and routing table state
- Samples every five seconds and store EFSA state and variable state
- 50 nodes wandering in 1 km^2 area for 100,000 seconds (= 27.8 hours)
- Ten attack runs and two normal runs

Results

- Specification violations
  - Data drop
  - Route drop
  - Add route
  - Delete route
  - Change sequence number, hop count
  - Active reply, False reply
  - Route invasion, Route loop
  - Partition
- No false positives,100% detection

Statistical Deviations

Anomalous basic event	Detection Rate	False Alarm Rate
Flooding of data packets	92±3%	5±1%
Flooding routing messages	91±3%	9±4%
Modification of routing messages	79±10%	32±8%
Rushing of routing messages	88±4%	14±2%

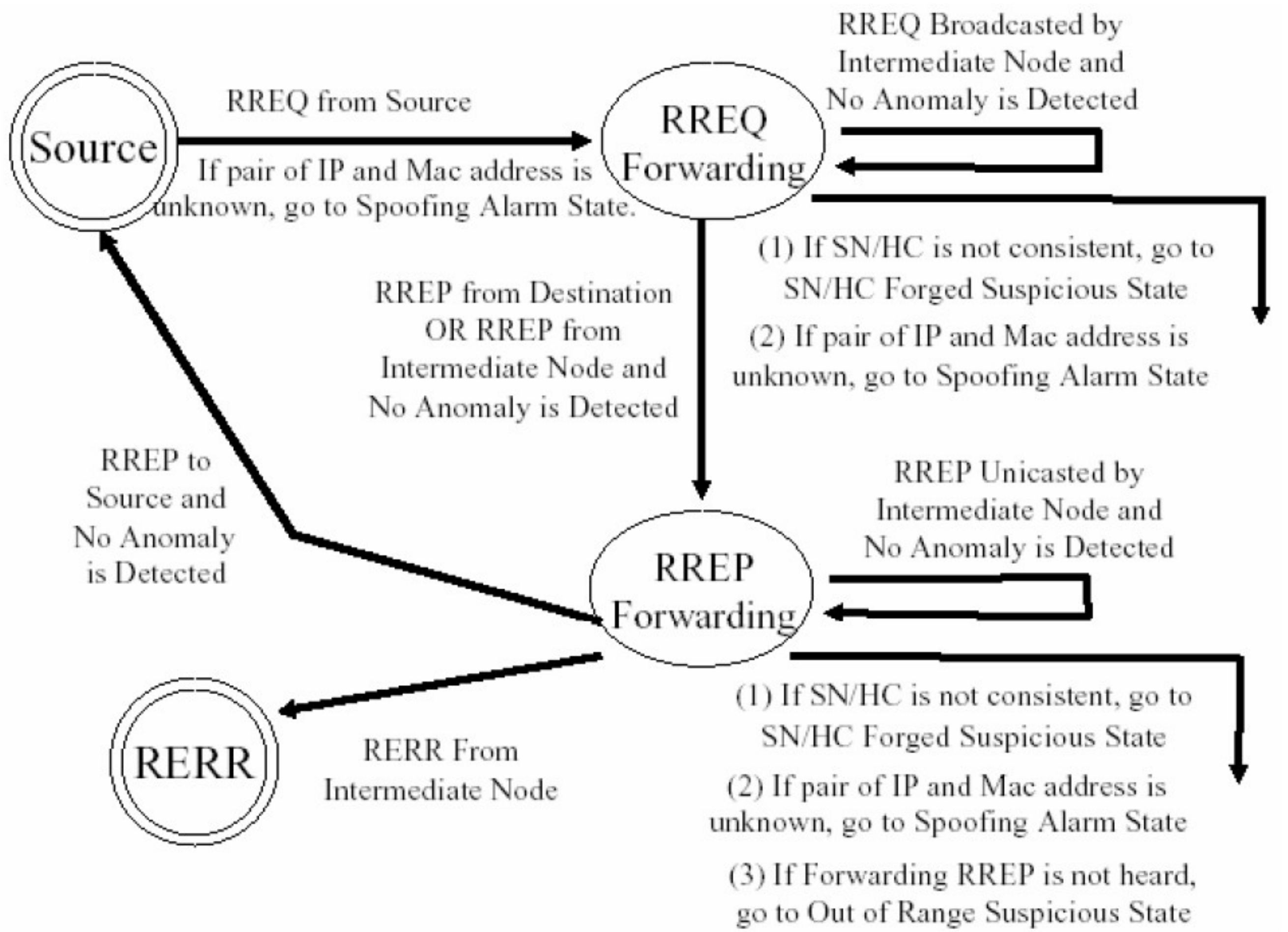
Discussion

- Detecting Flooding
  - Traffic over 20 packets per second
- Modification of Routing Messages
  - Learned by watching for sequence number jumps over a threshold
  - Doesn't work very well since randomly generated sequence number attack isn't always noticed
- Rushing of Routing Messages
  - Tries to find when node quits waiting early
  - Hard to find because it happens normally when route discovery process terminated
  - Easier to find rushing in returning route received messages because one transition (T11) happens more frequently

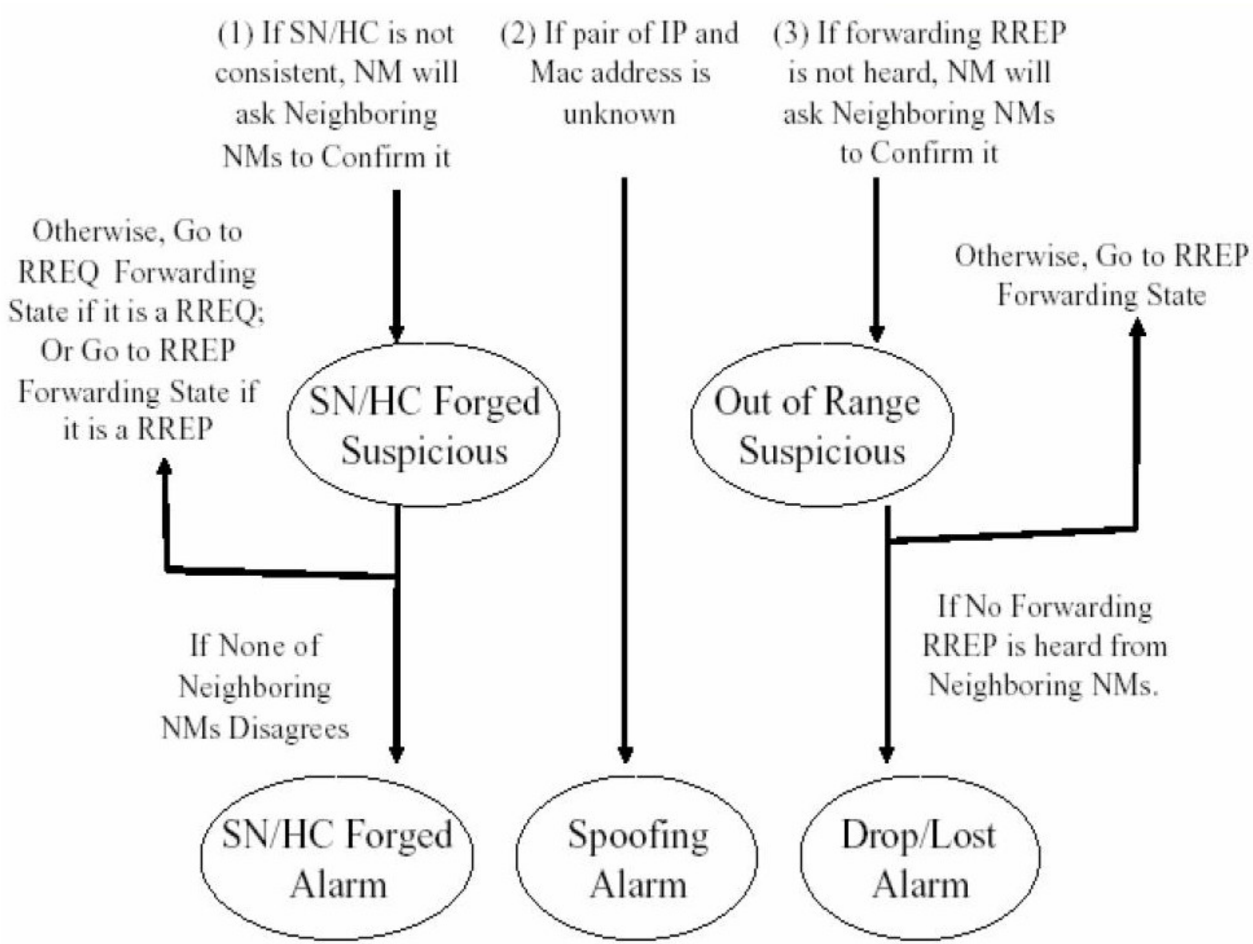
Another way to do it: Message Oriented

- Use a network monitor (NM) to watch all messages in a network area
- NMs keep a tree of all Route Request and Route Reply messages
  - Correlate messages by source, destination, and request ID number
  - NMs share information with each other and nodes
- If sequence numbers or hop counts change between messages, register attack

EFSA for normal behavior



EFSA for anomalous behavior



Attacks detected

- Forging sequence numbers, hop count
- Man in the middle attack
  - NMs will notice declared source doesn’t match true source
- Tunneling attack
  - Route declared is not the one really taken, NMs will notice forwarding is forged

Comparison and Discussion

- Node oriented specification catches routing table attacks
- Node oriented requires close analysis of protocol to build complex state diagram
  - Once built it can be used for statistical deviation attacks too
- Message oriented gives a global view of messages sent
  - Can catch network topology attacks better
- Message oriented could be used for flooding attacks, message modification attacks, and rushing as well or better than node oriented

Conclusion

- Intrusion detection by comparing actual behavior with specification
- Choice of specification (i.e. node/message orientation) determines what can be detected
- Not all attacks are specification attacks, so statistical deviation analysis is needed too

References

- AODV: RFC3561
  - http://www.ietf.org/rfc/rfc3561.txt
- Huang, Yi-an and Wenke Lee. Attack Analysis and Detection for Ad Hoc Routing Protocols., In *Proceedings of the 7th International Symposium on Recent Advances in Intrusion Detection (RAID'04)*, French Riviera, France. September 2004.
- Tseng, Chin-Yang, et al. A Specification-based Intrusion Detection System for AODV., In *Proceedings of the 1st ACM Workshop on Security of Ad hoc and Sensor Networks (SASN'03)*. Fairfax, VA. 2003.
- Ning, Peng and Kun Sun. How to Misuse AODV: A Case Study of Insider Attacks Against Mobile Ad hoc Routing Protocols. In *Proceedings of 2003 IEEE Workshop on Information Assurance.* West Point, NY. 2003.