Secure High-Throughput Multicast Routing in Wireless Mesh Networks

Jing Dong, Reza Curtmola, Cristina Nita-Rotaru

Presented by: Scott Hackman
22 March 2012
Objectives

- Background
- Protocol and Metric Overviews
- Types of Network Attacks
- S-ODMRP Implementation
- Experimental Results
Background

- Traditionally, routing protocols have focused on *hop count* as their selection metric.

- *Hop count* provides suboptimal performance due to unreliable links. (The system would prefer a path with fewer hops, even if those links were poor)

- In an effort to increase performance, high-throughput metrics were introduced that rely on periodic probing by nodes to their neighboring nodes to assigned a *quality* to their shared links.
Background

- These multicast solutions for high-throughput routing were derived from unicast protocols.
- In this paper, the authors used the SPP multicast metric, which was derived from the ETX unicast metric.
Expected Transmission Count Metric (ETX)

- ETX is a proposed unicast metric that calculates the expected number of transmissions needed to successfully deliver a unicast packet from the source node to the receiver node.

\[
ETX = \frac{1}{d_f \times d_r}
\]

- Where \(d_f\) and \(d_r\) are the probabilities that a packet is sent successfully from node A to node B and back.
Expected Transmission Count Metric (ETX)

\[ \text{ETX}_{S \rightarrow R} = \sum_{i=1}^{k} \text{ETX}_i \]

- The ETX value from source to receiver represents the total number of transmissions required to transmit one packet from source to receiver.
Success Probability Product (SPP)

- SPP is a metric for multicast networks adapted from the ETX model.
- Key differences:
  - For multicast, only the forward direction matters, as there is no link-layer acknowledgement.
  - Because there are no retransmissions, successful transmission is based on the product of the individual scores instead of the sum.
**Success Probability Product (SPP)**

\[
SPP_i = d_f
\]

\[
SPP_{S \rightarrow R} = \prod_{i=1}^{k} SPP_i
\]

- The new SPP value is the product of all quality measures, which represents the expected number of transmissions required to successfully deliver a message along the path. For simplicity, this number is inverted such that 0 represents a completely unreliable node and 1 represents a completely reliable node.
On-Demand Multicast Routing Protocol (ODMRP)

- ODMRP is a protocol that adds nodes through a route selection and activation scheme.
- The source periodically executes a JOIN QUERY message that uses a basic flood suppression mechanism.
- The receiver sends a JOIN REPLY message back to the source, once received.
On-Demand Multicast Routing Protocol (ODMRP)

- In the JOIN REPLY message, *next hop* refers to the upstream corresponding node. In this case, an intermediate node that is passing along a JOIN REPLY message can see that it is in the return path, which will cause it to add itself to the FORWARDING GROUP and execute its own JOIN REPLY message.

- Once the source receives its response, the multicast receivers are connected to the source through a mesh of nodes.
ODMRP-HT

- ODMRP-HT is an enhancement to ODMRP that selects routes utilizing high-throughput quality metrics, vice shortest delay.

- ODMRP-HT also uses a weighted flood suppression technique where the node processes flood duplicated for a fixed amount of time and rebroadcasts the messages that have the highest quality measures.
Attacks Against High-Throughput Multicast

Overall, the throughput of the system is directly proportional to the system’s vulnerability to malicious attacks.

Types of attacks:

- Dropping
- Injecting
- Modifying
- Replaying
Resource Consumption Attacks

ODMRP-HT has no mechanism to prevent an attacker from:

- Creating its own JOIN QUERY message
- Sending out spoofed JOIN REPLY messages
- Injecting its own illegitimate data packets into the network.
Mesh Structure Attacks

- These attacks disrupt the correct establishment of the mesh structure by maliciously manipulating JOIN QUERY or JOIN REPLY messages.
- An attacker may spoof a JOIN QUERY message and establish itself as the source, or it can drop a JOIN REPLY to avoid being established as a node within the multicast protocol.
Secure High-Throughput Multicast Routing in Wireless Mesh Networks

Metric Manipulation Attacks

- Fundamentally, an attacker manipulates its quality measures to increase the likelihood of being selected as a node in the multicast protocol.

- This can be done locally or globally.
  - Local Metric Manipulation (LMM): The node changes its own local score.
  - Global Metric Manipulation (GMM): The node changes the total accumulation and rebroadcasts the incorrect score.
Secure High-Throughput Multicast Routing in Wireless Mesh Networks

Metric Manipulation Attacks

![Diagram of network with nodes S, A1, A2, A3, B1, B2, B3, C1, C2, C3, and R, with metrics and connections labeled.]
Metric Manipulation Attacks

Fig. 3. Metric manipulation attack in a network with one source (S), two receivers (R₁, R₂), and one attacker (A). The label on each link represents the value of the link’s SPP metric. (a) Benign case. (b) Attack case.
Solution: Secure-ODMRP

- S-ODMRP is the solution the authors propose for all of the previously mentioned security issues.
- Each node in the network contains a pair of public and private keys and a client certificate.
- Authenticating nodes in the network removes a large number of attacks, specifically outsider attacks, injection of corrupt data packets, message spoofing and modification.
- To prevent authenticated nodes from performing a resource consumption attack, a limit is set on how many JOIN REPLY messages can be sent in one round.
RateGuard

- RateGuard is a component of S-ODMRP and operates under the assumption that attacks do not really affect the protocol significantly unless they affect the packet delivery ratio (PDR).
- Because of this, RateGuard takes a reactive approach and detects and isolates malicious behavior observed in the network.
RateGuard

- **Measurement-based attack detection**: Regardless of the attack used, if the data transmitted through a node does not correspond to the advertised datarate, something is wrong.

- By allowing neighboring nodes to calculate expected PDR (ePDR) and perceived PDR (pPDR) through an adjacent node, nodes can reason that their neighbors are under attack if there is a large enough discrepancy between ePDR and pPDR.
RateGuard

- **Accusation-based attack reaction**: When a node suspects one of its neighbors is malicious, it broadcasts an ACCUSATION message containing its own identity and its claim.

- This process is controlled by, and good nodes are protected by, an accusation duration timer proportional to the severity of the discrepancy.

- This allows good nodes, which may have a lapse in connectivity to rejoin the network after the timer expires.
RateGuard

- While the accusation is valid, metrics advertised by the node will not be used, and that node will not be allowed to be a part of any FORWARDING GROUP.

- To prevent accusation abuse, a node may not issue another accusation while one it issued is still valid.

- Similarly, to prevent a malicious node from locking out all its surrounding neighbors, an ACCUSATION random delay is introduced, and a node will listen for another ACCUSATION before executing its own.
Secure High-Throughput Multicast Routing in Wireless Mesh Networks

S-ODMRP Basic Procedures

\text{Sign}(m): \text{sign message } m \text{ using this node's private key}
\text{Verify}(n\_id, \text{sig}): \text{verify the signature } \text{sig} \text{ using node } n\_id\text{'s public key}
and exit the procedure if the verification fails
\text{Start\_timer}(\text{timer, } t): \text{start timer } \text{timer with timeout } t
\text{Refresh\_timer}(\text{timer, } t): \text{if timer is not active, then call}
\text{Start\_timer}(\text{timer, } t); \text{otherwise, set timeout of timer to } t
\text{Broadcast}(m): \text{broadcast message } m \text{ one hop}
\text{Flood}(m): \text{flood message } m \text{ in the entire network}
\text{Send\_message}(m, n\_id): \text{reliably send message } m \text{ to neighbor } n\_id
\text{Link\_metric}(n\_id): \text{return the measured link metric to neighbor } n\_id
\text{Get\_best\_metric}(\text{query\_set}): \text{return the best metric of all queries in the}
\text{set } \text{query\_set}, \text{regardless of accusation status}
\text{Get\_neighbor\_best\_metric}(\text{query\_set}): \text{return the neighbor that has the}
\text{best metric in the set } \text{query\_set}, \text{regardless of accusation status}


Mesh Creation Algorithm

Executed at the source node to initiate a new JOIN QUERY message:

1: create a JOIN QUERY message q
2: q.source = source_id; q.from = source_id
3: q.path_metric = 1; q.seq = join_seq
4: join_seq ++
5: Sign(q); Broadcast(q)
Mesh Creation Algorithm

Executed at a node upon receipt of a JOIN QUERY message q:

6: if (latest_received_join_seq > q.seq) then
7:   return // ignore old queries
8: Verify(q.from, q.sig)
9: get_new_query = FALSE
10: if (latest_received_join_seq < q.seq) then
11:   // get a new (non-duplicate) query
12:   latest_received_join_seq = q.seq
13:   best_metric = 0
14:   best_upstream = INVALID_NODE
15:   fastest_upstream = q.from // for fallback recovery
16:   get_new_query = TRUE
17: received_queries.insert(q) // store the query
18: if (accusation_list.contains_accused_node(q.from)) then
19:   q.path_metric = 0
20: else
21:   q.path_metric = q.path_metric × Link_metric(q.from)
22: if (get_new_query OR q.path_metric > best_metric) then
23:   best_upstream = q.from; best_metric = q.path_metric;
24: q.from = node_id
25: Sign(q); Broadcast(q)
26: if (get_new_query AND is_receiver) then
27:   Start_timer(reply_timer, REPLY_TIMEOUT)
Mesh Creation Algorithm

Executed at a node upon timeout of reply_timer:
28: Send_reply()

Executed at a node upon receipt of a JOIN_REPLY message r:
29: if (latest_received_reply_seq < r.seq) then
30: latest_received_reply_seq = r.seq
31: Refresh_timer(FG_timer, FG_TIMEOUT)
32: if (not is_receiver) then
33: Send_reply()

Send_reply()
34: create a JOIN_REPLY message r
35: r.seq = latest_received_join_seq
36: Send_message(r, best_upstream)
37: if (best_metric > 0) then
38: start monitoring the PDR of best_upstream
39: if (Get_best_metric(received_queries) > best_metric) then
40: // Activate the accused neighbor with best metric
41: Send_message(r, Get_neighbor_best_metric(received_queries))
42: received_queries.clear() // purge stored queries
Attack Detection

- pPDR is calculated by finding the ratio of packets received and packets sent in a given time window.
- ePDR-pPDR must be within a certain threshold to avoid invoking an ACCUSATION message from a neighboring node.
**Attack Reaction**

On detecting a discrepancy between ePDR and pPDR:

1:  \texttt{Start\_timer(React\_Timer, }\beta(1 \text{ - ePDR)}\texttt{)}

\textbf{Executed at node on timeout of React\_Timer:}

2:  \texttt{if (is\_receiver) then}
3:  \texttt{create salvage message ss // fallback}
4:  \texttt{Send\_message(ss, fastest\_upstream)}
5:  \texttt{if (accusation\_list\_.contains\_accuser\_node(node\_id)) then}
6:  \texttt{return // each node can only accuse once}
7:  \texttt{// create and flood accusation message}
8:  \texttt{create accusation message acc}
9:  \texttt{acc.accused = best\_upstream}
10: \texttt{acc.accuser = node\_id}
11: \texttt{acc.accusation\_time = }\alpha(ePDR - pPDR)
12: \texttt{accusation\_list\_.add(acc)}
13: \texttt{Sign(acc); Broadcast(acc)}
14: \texttt{// send recovery message to the subtree}
15: \texttt{create recovery message rr}
16: \texttt{rr.accusation = acc}
17: \texttt{Sign(rr)}
18: \texttt{for each downstream node d do}
19: \texttt{Send\_message(rr, d)}
Secure High-Throughput Multicast Routing in Wireless Mesh Networks

Attack Reaction

Executed at a node on receipt of an accusation message acc:
20:  if (accusation_list.contains_accuser_node(acc.accuser)) then
21:      return // only allow one accusation from a node at a time
22:  Verify(acc.accuser, acc.sig)
23:  accusation_list.add(acc)
24:  Broadcast(acc)

Executed at a node on receipt of a recovery message rr:
25:  if (handled_recovery_messages.contains(rr)) then
26:      return // ignore duplicate recovery
27:  if (accusation_list.contains_accuser_node(rr.acc.accuser)
       OR rr.acc.accusation_time < α(ePDR − pPDR)) then
28:      return // ignore recovery mesage if the accuser has an unexpired
           accusation or if the accused time is inconsistent
29:  Verify(rr)
30:  handled_recovery_messages.insert(rr)
31:  if (React_Timer is active) then
32:     cancel React_Timer
33:  for each downstream node d do
34:     Send_message(rr, d)
35:  if (is_receiver) then
36:    create salvage message ss // fallback
37:    Send_message(ss, fastest_upstream)

Executed at a node on receipt of a salvage message ss:
38:  Refresh_timer(FG_timer, FG_TIMEOUT)
39:  Send_message(ss, fastest_upstream)
Fallback Recovery

- Each node catalogs its fastest_upstream node, which serves as a last-resort recovery node to be used if the node’s upstream neighbor becomes accused. While fastest_upstream may not be the highest quality node, it is the fastest and serves as a lifeline until a new round is initiated and computes new routes.
Impacts of False Positives

- False positives are alleviated twofold:
  - First, this scheme only provides temporary lockouts from accusations, so a wrongfully accused good node can recover.
  - Second, the magnitude of the accusation lockout is proportional to the magnitude of the discrepancy, so an “honest mistake” by an honest node will not be punished too severely.
S-ODMRP Security Analysis

- Two theorems are proposed that show the upper bound on attacks, given parameters for round time (\(\lambda\)), accusation timer length (\(\alpha\)), and discrepancy threshold (\(\gamma\)).

**Theorem 1.** In a network with \(k\) metric manipulation attackers, for any \(\alpha \geq \frac{k\lambda}{\delta^2}\), the attack impact on any nonattacker receiver node in S-ODMRP is upper bounded by \(\delta\) during any time interval of duration \(T \gg \alpha\).
Theorem 1

Implications of Theorem 1. According to Theorem 1 for large enough $\alpha$, the impact of metric manipulation attacks is bounded by the attack detection threshold $\delta$. For example, with $\delta = 20\%$, round duration of $\lambda = 3$ seconds, and a total of 10 attackers, according to Theorem 1, we can set $\alpha \geq 750$ seconds to ensure the attack impact on any nonattacker receiver node is bounded by $\delta$. Theorem 1 assumes the attacker nodes can coordinate perfectly and completely disrupt the fallback procedure, thus it gives an upper bound on the impact of the attack.
Theorem 2. In a network with $k$ metric manipulation attackers, if S-ODMRP uses a fallback procedure that restores the PDR to the same level as in a benign network, then for any $\alpha \geq \frac{k \lambda}{\delta}$, the attack impact on any nonattacker receiver node is upper bounded by $\delta$ during any time interval of duration $T \gg \alpha$.

Implications of Theorem 2. In Theorem 2, we see that if we assume the ideal case where the fallback procedure is always able to restore the data rate to the normal level, then we can bound the attack impact under $\delta$ with a much smaller value for the accusation duration $\alpha$. For example, with the same settings of $\delta$ and $\lambda$ as above, we only need to set $\alpha \geq 150$ seconds.
RateGuard Attack Resiliency

- **Accusation Message Dropping**: Unless the attacker is at a chokepoint, since the ACCUSATION message is a broadcast, good nodes will still see it.

- **Accusation Message Injection**: Attacker node may only have one pending accusation, which will limit them. Also, the penalty time will be short, if a good node has very small discrepancy.
RateGuard Attack Resiliency

- **Recovery Message Injection**: Nodes will ignore recovery messages unless there is a corresponding accusation message.

- **Recovery Message Dropping**: The downstream node does not cancel its reaction timer, so this will ultimately end up with the attacker node being accused.

- **Attacks on the Fallback Procedure**: This is possible, however the impact will still be low since the routes are recalculated after the round.
Experimental Data

- 100 random nodes in a 1500m x 1500m area
- Considered the following scenarios:
  - No attack
  - Drop-only
  - LMM-drop
  - GMM-drop
  - False Accusation
Fig. 7. The effectiveness of metric attacks on ODMRP-HT. For comparison, we include attacks against ODMRP without high-throughput metrics. (a) Attacks on ODMRP. (b) Attacks on ODMRP-HT. (c) Attack strength against ODMRP-HT.
Experimental Data

Fig. 8. The effectiveness of S-ODMRP for different attacks. (a) Drop-Only attack. (b) LMM-Drop attack. (c) GMM-Drop attack.
Experimental Data

Fig. 9. Impact of the *False-Accusation* attack on S-ODMRP.
Conclusions

- Previous high-throughput multicast routing protocols have not sufficiently implemented enough security techniques to preserve the protocol in the presence of malicious attackers.

- S-ODMRP is a novel ODMRP implementation that utilizes RateGuard to combat a wide array of attacks, while providing relatively small amounts of overhead.
Questions?