The wisdom of Homer Simpson
Don't worry. Being eaten by a crocodile is just like going to sleep... in a blender.

It takes two to lie... One to lie and one to listen.

Lisa, Vampires are make believe, like Elves, Gremlins and Eskimos.

Aw, Dad, you've done a lot of great things, but you're a very old man, and old people are useless.

Secure Position Aided Adhoc Routing (SPAAR)

Part 2

By: Stephen Carter

Introduction

What is SPAAR?
• SPAAR is a family of protocols designed to secure ad hoc routing for a high-risk environment that utilizes position information to improve performance & security.

SPAAR's Security Requirements
1. Fabricated routing messages cannot be injected into network by malicious nodes
2. Routing messages cannot be altered in transit by malicious nodes
3. Routing loops cannot be by malicious nodes
4. Routes cannot be redirected from the shortest path by malicious nodes
5. Unauthorized nodes should be excluded from route computation and discovery
6. Network topology must not be exposed to malicious nodes by routing messages
7. Nodes must not store false routing information as a result of malicious node activity

SRP

Route discovery process

The attack

SRP (cont)

This attack shows that SRP does not satisfy some of our security requirements
• SR4: Routes cannot be redirected from the shortest path by malicious nodes
• SR6: Network topology must not be exposed to malicious nodes by routing messages
• SR7: Nodes must not store false routing information as a result of malicious node activity
SRP (cont)

• How does SPAAR defend against the SRP attack?
  • Only accept routing packets from authenticated and verified one-hop neighbors listed in the Neighbor Table

SPAAR Details

• Main components of SPAAR:
  • The Neighbor Table
  • The Route discovery process
  • Route Maintenance
  • The Destination Table

Neighbor Table

• Nodes maintain a neighbor table containing:
  • Neighbor ID
  • Neighbor’s Public Key
  • Neighbor’s Group Decryption key
  • Most Recent Location
  • Transmission Range
  • Location Update Sequence Number (LUSN)

Neighbor Table Setup

• To participate in SPAAR, each node requires:
  • Public/Private key pair
  • Certificate binding identity to its public key
  • Public key of the trusted certificate server
• Each node must have access to a trusted certificate server at some time prior to deployment

Neighbor Table Creation

1. A node N periodically broadcasts a “hello” message that includes its certificate
2. Nodes within range decrypt N’s certificate to verify N’s public key. An entry for N is created in their neighbor table and N’s public key is stored
3. Nodes then respond with a “hello_reply” containing their coordinates, transmission range, encrypted with their public key range encrypted with N’s public key

Neighbor Table Creation (Cont)

\[
\begin{align*}
\text{[hello, CERT}_N] & \rightarrow X \\
\text{[hello_rep, CERT}_X, \text{[MRL, TR, LUSN]} & \text{X}_k \rightarrow \text{N}_k
\end{align*}
\]
Neighbor Table Creation (cont)

4. Upon receiving a hello reply, N attempts to verify that the node is a one-hop neighbor
   - Distance between nodes is computed
   - If this distance is less than both of the nodes transmission range, the node is assumed to be a one-hop neighbor

5. N will now create a public/private key pair, called the Neighbor Group Key Pair (NGKP)
   - The private part of the NGKP is called N's group encryption key and denoted GEK_N
   - The public part of the NGKP is called N's group decryption key and denoted GDK_N

6. N distributes GDK_N to each of its neighbors listed in the neighbor table. The key is signed with N's private key, and encrypted with each neighbor's public key.

7. Each of N's neighbors receive, decrypt, and store GDK_N in their neighbor table.

N's Neighbor Table after Step 7

<table>
<thead>
<tr>
<th>ID</th>
<th>PK</th>
<th>GDK</th>
<th>MRL</th>
<th>LUSN</th>
<th>TR</th>
</tr>
</thead>
<tbody>
<tr>
<td>X1</td>
<td>X1_k</td>
<td></td>
<td>Lat.long 159</td>
<td>850m</td>
<td></td>
</tr>
</tbody>
</table>
| Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 | Neighbor1 |Neighbor Table Maintenance

Each node periodically broadcasts a “table update” message to inform its neighbors of its new position coordinates (and new transmission range)

N -> BC: [tbl_update, MRL, TR, LUSN] GEK_N

Neighbor Table Maintenance (cont)

Each node periodically broadcasts “hello” messages (step 1 of neighbor table creation), allowing for new neighbors to be added to the neighbor table.

N -> BC: [hello, Cert_N]
Some Issues

Would a location service be useful?
How would it affect security?
GPS security

Coming Soon

SPAAR Route Discovery/Maintenance techniques
Discussion of how SPAAR satisfies the seven security requirements