Whirlpool Routing for Mobility [1]

Laurent Lamouline    Vincent Nuttin

École Polytechnique de Louvain

June 10, 2011

[INGI2349 - Network and security seminar]
Introduction

- Whirlpool Ad-hoc Routing Protocol (WARP) is motivated by a large range of wireless sensor networks, operating in real-time in remote and sparse environments.
Introduction

- Whirlpool Ad-hoc Routing Protocol (WARP) is motivated by a large range of wireless sensor networks, operating in real-time in remote and sparse environments.

- Whirlpool Ad-hoc Routing Protocol aims to:
  - improve the efficiency of the routing of data to a sink moving within a static mesh (limited);
  - improve the ratio: $$\frac{\text{packets dropped}}{\text{packets sent}}$$
Introduction

The key insight of WARP is that
Introduction

- The key insight of WARP is that
  - it uses the existing topology to search around the old location of the moving destination;
The key insight of WARP is that
- it uses the existing topology to search around the old location of the moving destination;
- it uses a flag inside the data packets to route these packets towards the destination;
The key insight of WARP is that

- it uses the existing topology to search around the old location of the moving destination;
- it uses a flag inside the data packets to route these packets towards the destination;
- it assumes that some neighbours still have connectivity to the node moving into a mesh network.
Introduction

Figure: Mesh network composed of 9 nodes
Introduction

**Figure:** Mesh network composed of 9 nodes

**Figure:** A node is moving upward
Introduction

Figure: Some links are failing
Introduction

**Figure:** Some links are failing

**Figure:** Some neighbours still have connectivity with the moving node
Outline

1 The Whirlpool Algorithm
   - Overview
   - Four mechanisms for efficient routing
   - Summary

2 WARP Implementation

3 Evaluation and limitations of WARP

4 Conclusion
The Whirlpool Algorithm

Overview

- WARP provides a best-effort datagram delivery service
The Whirlpool Algorithm

Overview

- WARP provides a best-effort datagram delivery service
- WARP is an extended distance vector protocol
  - adds speculative routing when a destination move is detected
  - keeps a routing tree around the destination
  ⇒ could suffer from the “Count-to-infinity” problem
The Whirlpool Algorithm

Overview

- WARP provides a best-effort datagram delivery service
- WARP is an extended distance vector protocol
  - adds speculative routing when a destination move is detected
  - keeps a routing tree around the destination
  -⇒ could suffer from the “Count-to-infinity” problem
- When all the nodes are stationary, WARP operates as a standard routing protocol
The Whirlpool Algorithm

Overview

- WARP is a reactive routing protocol
  - it uses data packets to probe the network topology
  - route requests or control traffic are not used
The Whirlpool Algorithm

Overview

- WARP is a **reactive** routing protocol
  - it uses data packets to probe the network topology
  - route requests or control traffic are **not** used

- By default, **periodic beacons** are used by a destination to inform its neighbourhood of its presence
  - This mechanism could be costly
  - Upon data packet reception, the destination suppresses its next beacons
The Whirlpool Algorithm

Four mechanisms for efficient routing

- Four mechanisms are used to ensure an efficient routing
  1. Fast mobility detection
  2. Speculative routing
  3. Local repair
  4. In-band signalling
The Whirlpool Algorithm
Four mechanisms for efficient routing

1. Fast mobility detection
   - Link layer reliability is used to detect nodes mobility
   - Upon link failure detection/spiral packet reception, a node enters repairing state and it begins to send *spiral packets*
The Whirlpool Algorithm
Four mechanisms for efficient routing

1. Fast mobility detection
   - Link layer reliability is used to detect nodes mobility
   - Upon link failure detection/spiral packet reception, a node enters repairing state and it begins to send *spiral packets*.

Figure: Node mobility has been detected thanks to the link layer. Ancient neighbours (white nodes) of the mobile nodes detect the link failure and enter repairing state. When the destination overhears a data packet, it immediately initiates repair. [1]
The Whirlpool Algorithm
Four mechanisms for efficient routing

Speculative routing
- Used to proactively search the location of a destination
The Whirlpool Algorithm
Four mechanisms for efficient routing

2 Speculative routing
   - Used to proactively search the location of a destination

3 Local repair
   - When a node receives a beacon or an update packet
     - it exits whirlpooling state
     - it immediately updates its routing table
       ⇒ sometimes a bad decision
   - it informs its neighbours it has a new route towards the destination via update packets (timeout)
The Whirlpool Algorithm
Four mechanisms for efficient routing

4 In-band signalling
- WARP uses data packets as topology probes
- WARP requires 4 fields in the packet header
  1. spiral hopcount
  2. spiraling bit
  3. parent ID of the transmitter
  4. gradient cost of the transmitter
The Whirlpool Algorithm

Summary

Figure: State machine of the WARP algorithm. [1]
Outline

1. The Whirlpool Algorithm
2. WARP Implementation
3. Evaluation and limitations of WARP
4. Conclusion
WARP Implementation

- WARP is implemented as a simple modification of the “Collection Tree Protocol”
WARP Implementation

- WARP is implemented as a simple modification of the “Collection Tree Protocol”
- Uses the 6-bit reserved field of the CTP header to add the $S$ and $sthl$ fields
  - WARP drops packets with a $sthl$ value above a certain threshold
  - WARP uses the $S$ bit to notify that a packet is spiraling or not

**Figure:** WARP data packet header. The fields in blue belong to the standard Collection Tree Protocol (CTP) header. The two in white are added for WARP purposes: $S$ is the “spiraling bit” and $sthl$ stands for spiral time-has-lived, which counts how many hops a spiraling packet has taken.
WARP Implementation

- WARP uses 3 different packet types
  1. Tree packet: packet sent to the next hop with minimum cost when the topology is stabilized ($S = 0, STHL = 0$)
  2. Spiral packet: packet sent in the whirlpooling/repairing state ($S = 1, STHL \neq 0$)
  3. Update packet: specify that the sender has left the spiraling state because it has found a new valid route ($S = 0, STHL = 1$)
WARP Implementation

- WARP uses 3 different packet types
  1. Tree packet: packet sent to the next hop with minimum cost when the topology is stabilized ($S = 0, STHL = 0$)
  2. Spiral packet: packet sent in the whirlpooling/repairing state ($S = 1, STHL \neq 0$)
  3. Update packet: specify that the sender has left the spiraling state because it has found a new valid route ($S = 0, STHL = 1$)

- The WARP implementation memory needs
  - 2.8kB of code over CTP’s 5.5kB (total of 8.3kB)
  - 33 bytes of RAM over CTP’s 1kB
Outline

1. The Whirlpool Algorithm
2. WARP Implementation
3. Evaluation and limitations of WARP
4. Conclusion
Evaluation

- We know what WARP is and how it is implemented. What about performances?
Evaluation

- We know what WARP is and how it is implemented. What about performances?

- What do we need? Mainly,
  1. parameters;
  2. metrics;
  3. mobility models.
Evaluation

Parameters

The evaluation examines three parameters.

- The network *topology*;
- The *speed* of the destination;
- The *data rate* to the destination.
Evaluation

Metrics

The evaluation is based on three metrics.

- The reliability;

\[
Reliability \triangleq \frac{\text{number of packets delivered to the destination}}{\text{number of packets sent to the destination}}
\]

- The cost;

\[
Cost \triangleq \frac{\text{number of packets that the network transmits}}{\text{number of unique packets that a destination receives}}
\]

- The path-length.

\[
Path \text{ – length} \triangleq \text{number of hops a packet has taken}
\]
Evaluation

Mobility models

The evaluation considers two mobility models.

- The synthetic mobility.
- The real mobility.
Mobility model 1
The synthetic mobility

Figure: The destination “moves” along the red trajectory. Actually, turn by turn, each node of the trajectory plays the role of destination.

Advantage: Experiments are easily replayed.
Mobility model 2

The real mobility

- Real mobility is just ... real;

- There is signal attenuation due to the presence of a human.
Evaluation

Changing parameter: speed of the destination

- Under synthetic and real mobility models, results were similar.

- When the mobility increases,
  - reliability \(\downarrow\)
  - cost and path-length \(\uparrow\)

<table>
<thead>
<tr>
<th>Speed</th>
<th>Synthetic</th>
<th></th>
<th>Real</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Reliability</td>
<td>Cost</td>
<td>Path-length</td>
</tr>
<tr>
<td>0 m/s</td>
<td>99%</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>1 m/s</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1.5 m/s</td>
<td>91%</td>
<td>9</td>
<td>3.7</td>
</tr>
<tr>
<td>2.9 m/s</td>
<td>88%</td>
<td>12</td>
<td>4</td>
</tr>
</tbody>
</table>
Limitations of WARP

At very high speed

- Assuming an high speed of 2 hops/s ⇒ 0.5 s/hop.

<table>
<thead>
<tr>
<th>Data rate</th>
<th>Reliability</th>
<th>Cost</th>
<th>Path-length</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.3 pps</td>
<td>84.2%</td>
<td>43.7</td>
<td>12</td>
</tr>
<tr>
<td>7.1 pps</td>
<td>88.7%</td>
<td>72.1</td>
<td>13.1</td>
</tr>
<tr>
<td>21.3 pps</td>
<td>48.8%</td>
<td>96</td>
<td>10.3</td>
</tr>
</tbody>
</table>

Table: Cost and path-length increase. (‘pps’ stands for ‘packets per second’)

- Both cost and path-length increase dramatically
  ⇒ spiral whirlpool packets fail to find the new location.
  ⇒ network topology does not reconfigure.
  ⇒ WARP drops packets due to overflowing queues.
Limitations of WARP

At very high data rate

- Assuming an high data rate of 64 pps.

<table>
<thead>
<tr>
<th>Speed</th>
<th>Reliability</th>
<th>Cost</th>
<th>Path-length</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 hops/s</td>
<td>-</td>
<td>36.1%</td>
<td>29.7</td>
</tr>
<tr>
<td>0.13 hops/s</td>
<td>7.7 s/hop</td>
<td>44.8%</td>
<td>27.6</td>
</tr>
<tr>
<td>0.17 hops/s</td>
<td>5.9 s/hop</td>
<td>46.4%</td>
<td>26.2</td>
</tr>
<tr>
<td>0.25 hops/s</td>
<td>4 s/hop</td>
<td>45.9%</td>
<td>27.3</td>
</tr>
<tr>
<td>0.5 hops/s</td>
<td>2 s/hop</td>
<td>42.1%</td>
<td>30</td>
</tr>
</tbody>
</table>

Table: Cost and path-length remain stable.

- The reliability, the cost and the path-length remain stable.
Outline

1. The Whirlpool Algorithm
2. WARP Implementation
3. Evaluation and limitations of WARP
4. Conclusion
Conclusion

- WARP maintains high reliability for different data rates and mobility speeds.

- Key principles:
  - Rapid mobility detection (monitoring of wireless links).
  - Rapid adaptation of the underlying topology.
  - Speculatively sending packets around the last known location of the destination.

- WARP weakens the separation between the control and data planes.
Questions ?
References

Jung Woo Lee, Branislav Kusy, Tahir Azim, Basem Shihada, and Philip Levis.
Whirlpool routing for mobility.
In Vaidya et al. [2], pages 131–140.

Nitin H. Vaidya, Christoph Lindemann, and Jitendra Padhye, editors.