Report on the AODV Interop

Elizabeth M. Belding-Royer
Dept. of Computer Science
University of California, Santa Barbara
ebelding@cs.ucsb.edu

The first AODV Interop event was held on March 25 and 26, 2002, at the University of California, Santa Barbara. Attending were four AODV IPv4 (AODVv4) [3] implementations, and two AODV IPv6 (AODVv6) [2] implementations. The attending groups represented both industry and academia, and were from three different countries. The attendees were:

- Henrik Lundgren and Erik Nordstrom, Uppsala University
- Manel Guerrero Zapata, Nokia Research Center, Helsinki
- Antti Tuominen, Helsinki University of Technology
- Luke Klein-Berndt, NIST
- Jari Malinen, Nokia Research Center
- Ian Chakeres and Elizabeth Belding-Royer, UC Santa Barbara

The purpose of the event was to test the interoperability of the implementations, but not to test conformance to the AODV Internet Draft. It was also to serve as a learning experience, to understand better which portions of the Internet Draft need clarifications so that it can be more easily understood and correctly implemented.

The groups were each asked to bring two laptops with their implementation. On the first day of the event, the tests were all conducted on a wired network using a switch and a router. The router was able to control the port connections so that binary connectivity was established and multi-hop routes could be created. On the second day, many of the same tests, and additional ones, were conducted on wireless channels. Prior to the event, the attendees had installed *iptables* [1] on their laptops so that binary connectivity could be established with the wireless connections. For the wireless tests, all participants used IEEE 802.11b Orinoco PCcards. Testing was conducted in the lab using *iptables* to control connectivity. In addition, one test was performed where nodes were geographically distributed and obstacles were used to control the link connections; *iptables* was not used in this test.

The following describes the base set of tests that was used to test the interoperation of the implementations:

1. **Hello and Ping**: This test requires two directly connected nodes, where each node is a different AODV implementation. Each node should be periodically transmitting AODV Hello messages. Each node should receive the other node's Hello messages, and hence should install a route to that node. To test the installation of the route, each node should ping the other node, and ensure that the reply is received. The following actions must be verified:

    - Reception of neighbor Hello messages
    - Correct processing of Hello message by installation of route to neighbor in route table
    - Deletion of route from route table when neighbor is turned off

2. **2-hop RREQ/RREP**: This test requires three nodes, configured as 1-2-3, where node 1 is the source and node 3 is the destination. The nodes must be comprised of at least two different implementations. Hence, two of the nodes were running the same implementation. Two variations of the node configurations were tested. If the test was executed with implementations X and Y, then then configuration X-X-Y, as well as X-Y-Y, was tested. In some cases, due to problems with the implementation, only one of the configurations worked. This is represented as 3/2 in the table below. Hello messages are used in this test. Node 1 issues a ping to node 3. Because a route to node 3 does not exist at node 1, a RREQ must be issued. The following actions must be verified:

   - Node 1 issues a RREQ for node 3
   - Node 2 receives the RREQ and sends a RREP for node 3
   - Node 1 receives the RREP, installs the route to node 3, and the pings are transmitted to node 3 and returned to node 1

3. **RERR**: This test requires three nodes, configured as 1-2-3, where node 1 is the source and node 3 is the destination. After running test 2 and establishing a route from node 1 to node 3, the 2-3 link should be broken (i.e., by removing the wireless card from node 3). The generation of an RERR for node 3 verifies the correct operation of precursor nodes. The following actions must be verified:

   - Node 2 issues a RERR for node 3 and removes the route to node 3
   - Node 1 receives the RERR and removes the route to node 3

4. **Re-route**: This test requires four nodes, configured as 1-2-4 and 1-3-4. Hence there are two routes from node 1 to node 4, through either node 2 or node 3. The nodes must be comprised of at least two different implementations. This test verifies the re-routing capability after a link break in an established path. A route should be established as 1-2-4. The link from node 2 to 4 should be broken, and a new route through node 3 should be discovered. Hello messages should be used for this test. Two different implementation variations were used for this test. Given implementations X and Y, nodes 1 and 2 were implementation X and nodes 3 and 4 were implementation Y in the first experiment, and then for the second experiment implementations X and Y were reversed. In the case that this was only verified to work in one of the configurations, this is shown as 4/2 in the table below. The following actions must be verified:

   - After the link break, 2 issues a RERR for node 4
   - Node 1 receives the RERR from 2, deletes the route to 4, and issues a RREQ for node 4
   - Node 3 receives the RREQ and returns a RREP
   - Node 1 receives the RREP and installs a new route to node 4 through node 3

5. **3 hops**: This test requires four nodes, configured as 1-2-3-4. There must be nodes with two different implementations. This test is functionally the same as test 3, except that the RERR message must traverse one additional hop. After the route is established, where node 1 is the source and node 4 is the destination, the link between nodes 3 and 4 is broken. This tests the generation of RERR messages, and the proper implementation of precursor nodes. Two different configurations were tested: X→X→Y→Y, and Y→Y→X→X. If only one configuration was verified to operate, this is given as 5/2 in the table below. The following actions must be verified:

   - After the link break, node 3 deletes the route to node 4 and generates a RERR for node 4
   - Node 2 receives the RERR, deletes the route to node 4, and transmits the RERR to node 1
   - Node 1 receives the RERR and deletes the route to node 4

6. **Wireless 3 hops**: This test is the same as test 5, except that it is executed over wireless links. *iptables* was used to control the link connectivities.

Each implementation group was to execute each of the above tests with every other implementation group in attendance that had the same AODV IP version implementation. In other words, the four AODVv4 groups were supposed to implement these tests together, and the two AODVv6 groups were supposed to accomplish the tests together.

AODVv4 results: There were four AODVv4 groups in attendance, groups A, B, C, and D. Two of the groups, A and B, were able to successfully complete each of the tests. Groups A and C successfully completed tests 1, 2, 3, 4, and 5. They did not complete test 6 due to time constraints (it was not attempted). Groups B and C successfully completed tests 1, 2, 3, and 4. They did not complete tests 5 and 6 due to time constraints (they were not attempted). Implementation D had some problems with the way that the route table entry lifetimes were implemented. Lifetimes for Hello messages were not properly maintained, and so the implementation was not able to detect losses of connectivity. Because part of the time spent debugging this implementation, groups A and D were only able to complete tests 1, 2, and half of 3, groups B and D completed tests 1, 2, and half of 5, and groups C and D completed tests 1, 2, and half of 4. Note, though, that test 3 an inherent component of tests 4 and 5. So while test 3 was not explicitly completed between groups B and D, and C and D, the functionality was implicitly shown to operate. The explicit testing of test 3 was sometimes skipped due to lack of time. The following table summarizes the successfully completed tests:

	A	B	C	D
A	-	1,2,3,4,5,6	1,2,3,4,5	1,2,3/2
B	-	-	1,2,3,4	1,2,5/2
C	-	-	-	1,2,4/2
D	-	-	-	-

For the two IPv6 implementations, tests 1 and 2 were successfully completed. A more limited set of tests (less than 3 hops) were performed primarily due to lack of adequate filtering support for IPv6 on the tested platforms and a smaller number of participating implementations. These tests provided the first evidence that the same interoperation as with IPv4 is feasible using IPv6 version of the protocol.

In addition to these standard tests, a number of other tests were performed between different IPv4 implementations:

1. **TCP over 3 hops**: This test requires four nodes, configured as 1-2-3-4. A TCP session is established from node 1 to node 4. This test verifies route establishment for TCP sessions. The following actions must be verified:
   - A route is established from node 1 to node 4 using a TCP application (i.e., an HTTP request)
2. **6 wireless nodes, multiple functionalities**: This test requires five nodes, configured as 1-2-4-5 and 1-3-4-5. Hence node 1 can route through either node 2 or 3 to reach node 4 and 5. To complete this test, there were three different implementations. *iptables* was again used for controlling link connections. In this test, a ping was sent from node 1 to node 5 through nodes 2 and 4. Node 2 was then removed from the network, a re-route through node 3 was verified. Once the route 1-3-4-5 was constructed and the ping was successful, a HTTP request was issued from node 1 to node 5, through nodes 3 and 4. This verified the operation of a TCP connection. A sixth node, node 6, was then connected to node 1, such that the network was connected as 6-1-3-4-5. An HTTP request was issued from node 6 to node 5, and the web page was successfully received. This required node 6 to issue a RREQ for node 5, and node 1 to return a RREP for that route.

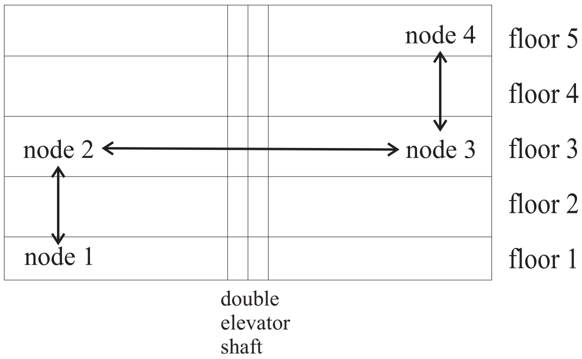

Figure 1: Multi-hop Test.

3. **Pure wireless**: This test required 4 nodes, and ran over true wireless links (i.e., *iptables* was not used to control link connections). The nodes were distributed throughout a building in the configuration shown in figure 1. The combination of floors, walls, and the elevator shaft resulted in the following network configuration: 1-2-3-4. An HTTP request was issued from node 1 to node 4. Route discovery was executed, and the web page was successfully retrieved from node 4. This test incorporated all four participating AODVv4 implementations.

In summary, the tests demonstrated interoperability between the independent AODV implementations. The majority of the tests were successfully completed during the course of the interop. Minor programming bugs in some of the implementations were discovered and fixed during the event, and all implementations were shown to interoperate with each other. A number of potential Internet draft modifications were discussed as a result of the event that will ease in the understanding of the draft meaning and correct implementation of AODV functionality. This is the first recorded interoperability testing of an ad hoc routing protocol, and the first successful ad hoc network created from four independent implementations of the same protocol.

References

[1] Jozsef Kadlecsik, Harald Welte, James Morris, Marc Boucher, and Rusty Russell. Iptables. ¡http://www.iptables.org/¿.

[2] C. E. Perkins, E. M. Belding-Royer, and S. R. Das. Ad hoc On-Demand Distance Vector (AODV) Routing for IP version 6. *IETF Internet Draft, draft-ietf-perkins-aodv6-01.txt*, November 2001. (Work in Progress).

[3] C. E. Perkins, E. M. Belding-Royer, and S. R. Das. Ad hoc On-Demand Distance Vector (AODV) Routing. *IETF Internet Draft, draft-ietf-manet-aodv-10.txt*, March 2002. (Work in Progress).