Preventing Routing Loop in Routing Distance Vector Protocol using Destination Sequence Distance Vector (DS-DV) Method

Octara Pribadi\textsuperscript{1} Robet\textsuperscript{2} Feriani Astuti Tarigan\textsuperscript{3} Hendri Edi Wijaya\textsuperscript{5} Jackri Hendrik\textsuperscript{6}

\textsuperscript{1,2,4,6}Department of Information Technology, STMIK TIME, Medan, Indonesia
\textsuperscript{3,5}Department of Information System, STMIK TIME, Medan, Indonesia

*octarapribadi@stmik-time.ac.id

Abstract. Count to infinity is a problem that appear in the vector routing protocol. This is because, there is a routing loop caused by the nodes structure in the networking that create loop. The destination sequence – distance vector method can be a solution to prevent the routing loop, so that the count to infinity problem can be solved in the protocol routing distance vector.

1. Introduction

Routing is a process routing data from one network to other network. Routing process can be by static or dynamic using routing protocol. Routing protocol can learn dynamically the other node position in the network and keep or process the information automatically, if there is a change in the network topology.

There are two types of routing protocol:

- Distance Vector Routing protocol: routing protocol information (RIP), Enhanced interior gateway routing protocol (EIGRP), etc.
- Link State Routing protocol: open shortest path first (OSPF), intermediate system to intermediate system (IS-IS), etc.

In routing link state algorithm, network topology and cost connection among nodes known and can be as input value in routing link state algorithm [1]. In practice, every node will broadcast all information in the network to all another nodes. So each node will know obviously the network topology information, the shortest distance, cost among network, etc.

While, in distance vector routing algorithm, node accept, process, and transfer information among other nodes that connected directly then distribute to other node. Distance vector algorithm only guarantee node information that connected directly, whereas other node information the truth cannot be ascertained. Distance vector algorithm has a problem calls routing loop.

This is because, for example there is node A believes that the route to reach node C can pass through B, but A does not know that B must pass through A to reach C. This is because A only knows the
neighboring information that directly connected, without knowing the information what other nodes know.

There are several methods implemented to solve this problem, namely split horizon, triggered update, and hold timer. However, this method has not been able to efficiently solve all possible problems so the last method is to determine the maximum hop between nodes[1].

2. Literature Review

Loop routing is a problem that arises when a node uses incorrect information to determine the route to another node. This is because the node knows some information in the network, not the details of all network topology information [2]. So it's difficult for distance vector algorithms to be able to handle this problem. The routing loop illustration can be seen in Figure 1.

![Figure 1. The arrangement of nodes forms loops](image)

It is assumed that the metric used is hop. The metric value between A and F is 1, while other nodes except A that will go to F are sure to pass through A so that the metric value is definitely greater than 1. Now assume the connection between nodes A and F is broken (see figure 2).

![Figure 2. In Figure 2 when A and F disconnected, A will try to notify all other nodes of that information. But at the same time when the information hasn't reached node B or C, it turns out that C first sends information periodically every 30 seconds to the RIP protocol which states that C can go to F (through A) then A will believe that C can go to F, but A doesn't know that to go to F, C must pass through A itself. Then there is routing loop, which then makes the metric values of both node A and C increase each unit time to unlimited values, or known as count-to-infinity. This can be prevented if, other nodes can be ascertained to be temporarily silent, when there are nodes that are disconnected, so that the most valid information (information from nodes that were previously directly connected to the broken node) can be informed throughout the network. But this cannot be done because of the limited information known to nodes in the network.](image)
In the RIP protocol the maximum allowable metric value is 15, more than that the corresponding node will be considered unreachable. This causes the scale of the network that can be involved to be small because of the limited number of nodes.

Split horizon is one method that is used to prevent node X from passing information to node Y, retrieving information from the node Y which originally came from X. The illustration can be seen in Figure 3.

![Figure 3. Split horizon](image)

In Figure 3, information coming from node C is sent first to B, B knowing that the information coming from C will not send the information to C but to another node, node A. Split horizon method can prevent routing loops because information from one node will never return to the same node. But split horizon cannot prevent routing loops if the node structure forms a loop, for example node A is connected to node B, node B is connected to C, and node C is reconnected to A. Information from A is sent to B, B sends to C and C again sends information to A, this is because C does not know that the information coming from B turns out also come from A, so it can possibility occur routing loops.

Triggered update is one of the methods used to prevent count-to-infinity by sending information as fast as possible (without waiting for periodic updates) if there is a connection with other nodes interrupted. With a triggered update expected, if there is a broken node, the other nodes can immediately notify this information with other connected nodes, so that other nodes can learn this information and not send the wrong information that should be invalid into the network. It's just that triggered updates cannot be relied on for all possibilities, especially if there are many nodes in the network that broadcasting periodic information shortly before the triggered update has not sent the latest information.

Hold time is one of the exclusive methods used by Cisco routers. This method will make the node that has a neighbor node that is disconnected, will never trust any information that is obtained during a certain period (default 180 seconds on EIGRP), so that information that has expired will not poison the information table of the node concerned and not resulting routing loops. The side effects of using hold time are having a big impact on the length of time needed for convergence. The previous three methods cannot effectively prevent routing loop, so the next method is to limit the maximum distance metric between nodes, so that if the routing loop occurs the metric value will continue to increase until it reaches the maximum value of +1 and automatically the invalid information will be discarded.

There are several other methods developed to prevent routing loops occurring, such as the diffusing update algorithm (DUAL) method [4] and the destination sequence-distance vector (DS-DV) method [5]. The DS-DV method is one method that can prevent routing loop in routing protocol based on distance vector by sending sequence number information periodically, where the most reliable information is information with the largest sequence number [5].

For each periodic time (for example 5 seconds), each node sends a packet of information called "hello" which contains its own sequence number to the neighboring node. For each periodic, the sequence number sent will increase by 2, so the most reliable automatic information is information that has the largest sequence number (because it is the newest). From this sequence number the information received from other nodes will be processed and the information inserted into the routing table.

But if node X detects a node y that is broken then the sequence number of node Y will be added with 1, and node X will send this information to all other nodes. Therefore if the node Y is reconnected, the sequence number of the node Y whose value is greater (periodically increasing 2) will automatically update the sequence number information of the node y in the network.
In Figure 4 there are nodes arranged in such a way so that resulting routing table information (example node A) can be seen in table 1.

Table 1. Routing table node A

| Destination | via | metric | Sequence Num. | Install time |
|-------------|-----|--------|---------------|--------------|
| A           | A   | 0      | A-100         | 8            |
| B           | B   | 1      | B-12          | 10           |
| C           | C   | 1      | C-46          | 5            |
| D           | D   | 1      | D-22          | 7            |
| E           | B   | 2      | E-56          | 12           |
| F           | C   | 2      | F-34          | 14           |

In table 1 an example is taken, for example node A can go to node E through B where the metric distance is 2 and the sequence number is E-56 and install time 12.

A few moments later when the periodic time of node E has been reached, E will send the sequence number E-58 (E-56 + 2) to the neighboring node, node B and node F. Node B and node F will forward the information to node A. Because the number sequence E-56 < E-58, node A will trust information from node E sent from node B or F, see table 2.

Table 2. The routing table of node A after the update

| Destination | via | metric | Sequence Num. | Install time |
|-------------|-----|--------|---------------|--------------|
| A           | A   | 0      | A-100         | 8            |
| B           | B   | 1      | B-12          | 10           |
| C           | C   | 1      | C-46          | 5            |
| D           | D   | 1      | D-22          | 7            |
| E           | B   | 2      | E-58          | 12           |
| F           | C   | 2      | F-34          | 14           |

When node B and F forward sequence information node E to node A, node A will get 2 information with the same sequence number (assuming information until at the same time) then node A will select the information with the smallest metric, because if metric passes B is 2 smaller than the metric if passing C is 3, node A will select node B as the reference node towards node E.

Install time function is to find out the time of information inserted into the table. If the install time has exceeded a certain time (for example 15, 3 times the periodic time "hello") then it is assumed that the corresponding node is broken and the node will add the sequence node number value that broken with 1 and send this information to the neighboring node [5].
3. Methodology

In this experiments, the authors made a simulation program to try out the effectiveness of DS-DV in preventing routing loops in distance-vector routing algorithms. The metric unit used is the number of hops where the distance between nodes that are directly related is 1. Data that will be inserted into the simulation can be seen in Figure 5.

![Figure 5. Sample data to be simulated](image)

In Figure 5 (a) the sample data that will be simulated does not form a loop while in (b) there is a loop between the A-B-C nodes. After the data is inserted into the simulation, the next step of the connection between nodes will be disconnected, and it will be seen whether DS-DV can prevent routing loops on each node in the network.

In Figure 6, a simulation flow diagram of the program is created.

![Figure 6. Program simulation diagram](image)
In the simulation there are 3 timers that are used so that the entire node process can be carried out simultaneously, namely:

a. Timer hello
b. Timer neighbor
c. Timer routing

The timer hello will send sequence number information that increases periodically every 5 seconds (5 second selection is based on the EIGRP protocol) to all neighboring nodes. See figure 7.

Figure 7. "hello" package

Each node that has received the hello packet will check the neighbour table, if the neighboring table has not saved the node that sends the hello packet, then the node will be inserted into the neighbor table.

Timer neighbour, serves to synchronize between hello package and neighbor table, this neighbor table is the reference for the node to send "hello" information and routing information. The neighbor table only lists neighboring nodes that are directly connected.

Timer routing, serves to process all information received from neighboring nodes, from this information the node will determine the closest distance to reach the other node and determine when the information in the routing table expires. The routing table contains information, namely: destination node, via, metric, sequence number and install time.

Each hello packet sent from another node will be processed by the node in concerned with some conditions. Assumed:

a. $v_{ij}k$: node i goes to node j via k
b. $s_{ij}$: sequence number of node j stored by node i
c. $d_{ij}$: node i to j metric value
d. $t$: periodic time

In Figure 7, it is assumed that when $t = 10$ all nodes are in a stable state (convergence), the routing node A information is taken to node F as follows:

| Table 3. Routing Table |
|------------------------|
| Destination | via | metric | Sequence Num. | Install time |
| F           | B   | 2      | F-22           | 4            |

on table 3 it can be concluded $v_{AF} B$, $s_{AF} = 22$ and $d_{AF} = 2$, while node B has values $v_{BF} B$, $s_{BF} = 22$ and $d_{BF} = 1$. When $t = 15$ the periodic time of node F will send the hello packet $s_{FF} + 2 = 24$ to node B. Because $s_{FF} > s_{BF}$ then node B will update the routing table information with the sequence number according to $s_{FF}$ and send this information to node A which is automatically due to $s_{BF} > s_{AF}$ node A.
will update its routing information while changing the install time value back to 1 and sending this information to all neighboring nodes, except to node B where this information come from (split horizon).

It should be noted that, at the same time, node C can send routing information F to B before B receives routing information directly from F, but because \( s_{CF} = s_{BF} \) and \( d_{BF} < d_{CF} \) then node B will ignore the information from node C.

![Figure 8. Disconnect node B and F](image)

In Figure 8 it is assumed that the relationship between nodes B and F is broken. Then node B will know this if the install time information of node F on B has reached a value of 15 (5 times the periodic time hello), which means node F fails 3 times in a row to send sequence number information to its neighbors.

Then node B will change the sequence number of node F to \( s_{BF} + 1 \) and \( d_{AF} = \infty \) then send this information to the neighboring node, which will then be forwarded to another node. If you notice node F only has a direct connection with B, so that in the network information \( s_{BF} + 1 \) is greater than the sequence number of other nodes (except node F). So if at the time before node B sends this latest information to another node, it turns out it first gets information, for example \( v_{CF}B \) but because \( s_{CF} < s_{BF} \) node B will ignore information from C and thus the routing loop can be prevented.

4. Results

In Figure 9, a sample is inserted into the simulation program.
In Figure 9 (a) when $t = 5$ all nodes are stable. Then in (b) when $t = 7$, the periodic time of each node will reach node A and node A will update the sequence number of each node and reset the install time back from 1.

Assume at $t = 5$, the connection between node A and D is broken, then (c) when $t = 21$ or 16 seconds after that node A just realizes that node D is disconnected. Then A will change the sequence number $D-18$ plus 1 to $D-19$ and the metric value becomes -1. This information will be forwarded to other nodes during certain $t$ intervals (assumed 30 seconds). If at 30 seconds interval or $t = 55$ node D still has not sent the sequence number then in (d) when $t = 56$ node A will immediately delete node D information from its routing table, this also applies to other nodes. In the RIP algorithm, the time interval before the node is removed from the routing table is called flush time and it is 240 seconds [4].

If split horizon is activated, then without the DS-DV method can prevent routing loops occurring. This is because nodes B and C learn that to reach node D it can pass through node A. So that it is impossible for nodes B and C to send information that B and C can reach node D to node A. Therefore, node A will not get information invalid from other nodes. Of course it will be different if for example node B is connected to node C, see figure 10.

![Figure 10. Node A and D disconnect](image_url)

In Figure 10 node B will send information $v_{BD}$ A and $d_{BD} = 2$ to C and C will forward this information back to A, because node A has $d_{AD} = -1$ then node A will believe it can go to D through B, so the node A information becomes $v_{AD} B$.

Figure 11 showing results that entered into the simulation program:
In Figure 11 (a) when $t=1$ all nodes including A are stable, then in (b) when $t=6$, all nodes send information to node A, then node A will adjust the information based on the sequence number. At the same time the connection between nodes A and F is disconnected.

In (c) when $t=22$ or 16 seconds after node F is disconnected, node A will assume the connection to node F is broken, so that A will change $s_{AF}+1$ and $d_{AF}=-1$ and send this information to all nodes, and because other nodes have sequence numbers smaller than $s_{AF}$, the information from A will be trusted by all nodes.

For example node C sends information that node C can reach node F to A, but because $s_{CF}<s_{AF}+1$ then node A will ignore this information, and it is impossible for other nodes to have sequence numbers greater than $s_{AF}$ because node F is connected only to node A.

In (d) when $t=53$ or 31 seconds after node F is declared broken, it turns out it has not been connected, then automatically the node F information will be deleted from the routing node A table, it also applies to all other nodes. If after deleting it then node F is reconnected but with node C as shown in Figure 12.
Then node F sends the hello packet to C, C then passes this information to another including node A, so the routing table is node A in figure 11 (e) where \( v_{AF} = 2 \). Node A will forward this information to all nodes and information on the network to be stable. From these results it can be concluded that, with the DS-DV method, distance vector routing algorithms can avoid routing loops, because nodes use the latest information from sequence numbers sent periodically.

5. Summary and Conclusions

From the results of this experiments, can conclude as follows:
1. The DS-DV method can prevent routing loops from distance vector routing algorithm.
2. The structure of nodes that do not form loops can be completely prevented by the split horizon method.
3. The DS-DV method can accelerate convergence between nodes in the network
4. The advice for the next experiments are
5. Needed for the right time \( t \) so that it can reduce the load on nodes that have to broadcast in a short period but without reducing the effectiveness of the DS-DV.
6. Using a lot of samples to test DS-DV speed in stabilizing the network.

References

[1]. A.S. Tanenbaum and D. J. Wetherall, Computer Networks 5th edition, New York: Prentice Hall, 2011.
[2]. J. F. Kurose and K. W. Ross, Computer Networking : A top down approach 6th edition, New Jersey: Pearson, 2013.
[3]. C. E. Perkins and P. Bhagwat, "Highly dynamic Destination-sequenced Distance-Vector routing (DSDV) for mobile computers," SIGCOMM '94, pp. 234-244, 1994.
[4]. C. M. Kozierok, TCP/IP Guide A Comprehensive, Illustrated Internet Protocol Reference, San Fransisco: No starch press, 2005.
[5]. C. Hedrick, "Routing Information Protocol," Internet Request For Comments RFC 1058, June 1988.
[6]. J. J. Garcia-Lunes-Aceves, "Loop-Free Routing Using Diffusing Computations," IEEE/ACM Transactions on Networking, vol. 1, no. 1, pp. 130-141, 1993.