A Performance Study on REAchability Protocol in Large Scale IPv6 Networks

Habib Naderi
Department of Computer Science
University of Auckland
Auckland, New Zealand
e-mail: hnad002@aucklanduni.ac.nz

Brian Carpenter
Department of Computer Science
University of Auckland
Auckland, New Zealand
e-mail: brian@cs.auckland.ac.nz

***Abstract*-- Shim6 is a host-centric multihoming solution for IPv6 networks which has been chosen by the IETF as a multihoming solution for the future Internet. Shim6 employs REAchability Protocol (REAP) for failure detection and recovery. REAP enables multi-addressed hosts which are involved in a communication to detect and recover path failures without breaking the transport layer sessions. One concern about deploying shim6 is the performance of REAP in large networks like campus or enterprise networks. We modeled and simulated this protocol by using stochastic activity networks (SAN) and performed an analysis of its performance. This paper presents the results of the simulation. We also investigated the effect of send timer and initial probes on the performance.**

Keywords: IPv6, Multihoming, Shim6, REAP

I. INTRODUCTION

Multihoming is one of the desirable features for the current and future Internet. Fault tolerance, which is provided by multihoming, enables businesses to offer a continuous service to their customers and have the opportunity to compete with their rivals. In the current Internet, most large sites use BGP and its features to achieve this functionality. But this solution suffers from a big drawback: scaling. This solution inserts new entries to global routing tables and can lead to routing table explosion. So, in recent years researchers have been busy seeking a new solution for the future Internet using IPv6.

Different solutions have been proposed for this problem [1]. Shim6 [3] is one of those solutions. Shim6 is a host-centric multihoming solution for IPv6. In this category of solutions, hosts are responsible for providing multihoming service and the routing system is unaware of this functionality. This category of solutions is valuable because deployment would be highly incremental. Shim6 inserts a new layer inside IP. This layer performs a mapping between identifiers (ID) and locators. This mapping is transparent to the transport layer, so shim6 can switch between locators without breaking a transport session, another desirable feature of multihoming [2].

One important part of shim6 is the algorithm which is used for failure detection and recovery. Shim6 employs a separate protocol called REAchability Protocol (REAP) [4] for this purpose. REAP monitors transport layer communications and when a failure happens, it tries to recover. Recovery is performed by sending special messages called probe messages to the other end of communication to find a new working pair of available locators. One probe for each address pair is sent until a proper response is received from the peer. The new address pair is then adopted by shim6 without impact on the transport.

The Internet Engineering Task Force (IETF) chose shim6 as a solution for multihoming in IPv6 networks and the future Internet. Performance is a key feature for new Internet protocols. Shim6 should be able to present an acceptable performance, especially in large scale networks with thousands of computers, like campus and enterprise networks.

Antonio de la Oliva, *et al.* [5] performed a performance analysis on REAP to evaluate its behavior, but with just two hosts, each with two addresses, two TCP and two UDP based applications. Our results match their results although the aim of their work is different from ours.

In this paper we present the results of a performance study on REAP in large scale networks. We modeled REAP with Stochastic Activity Networks (SAN) [6] and used the Möbius modeling tool [7] to simulate the model. Our results show the behavior of the REAP in a network with thousands of connections. The remainder of the paper is organized as follows: an overview of REAP is given in section II. Section III presents the simulation environment and our assumptions. The results are presented and discussed in section IV. The effect of the send timer and initial probes are discussed in section V. We conclude our work in section VI.

II. OVERVIEW OF REACHABILITY PROTOCOL

Shim6 employs a protocol for failure detection and recovery called REAchability Protocol (REAP). REAP is responsible for two main jobs: detecting failure and finding another operational address pair, which is called address pair exploration, to recover from the failure. REAP uses FBD (Force Bidirectional Detection) for verifying reachability and detecting failure. That means if there is traffic in one direction there also should be traffic in the reverse direction. REAP sends keep-alive messages in the case there is incoming traffic but there is no data to be sent in return. It would be a sign of failure if there is outgoing traffic but no traffic in return. To manage this process, REAP employs two timers: send timer and keep-alive timer. When a payload packet is sent, REAP starts send timer and stops keep-alive timer. When a packet is received, REAP starts keep-alive timer and stops send timer. When keep-alive timer expires, it means that it's time to send a keep-alive message to the peer because it has already sent a payload packet but nothing has been sent to it in return. When send timer expires, it means that there has been no incoming traffic for a while and is a sign of failure.

Thus, when send timer expires, REAP considers it a failure and starts the address exploration process to find a working pair of addresses. First, using known host and peer addresses, a list of address pairs is created. This list is pruned and ordered using address selection rules [8]. Then REAP starts to check reachability of these addresses by sending probe messages to the peer using the list's address pairs. The list is traversed sequentially. First four probes, initial probes, are sent with a time interval equal to 0.5 second. For further attempts, an exponential backoff procedure is employed to avoid a signaling storm. This procedure increases the time between probes until it reaches 60 seconds. Later probes will be sent with this time interval. Most implementations double the time between probes although this type of increase is not mandated by RFC 5534. This process finishes when a working address pair is found. When the peer receives a REAP probe, it replies and starts a similar process for finding a working address pair. So, two ends of a communication may use different address pairs after recovery. Shim6 conceals this from the transport layer.

III. SIMULATION SETUP

We executed the model with 10,000 instances of REAP. Fig. 16 shows a graphical representation of the Möbius model. Each instance models the behavior of REAP for one shim6 context. So it can be considered, for example, as a site with 1000 hosts where each host has 10 connections that should be recovered after a failure affecting the site.

Our model enables us to specify the number of address pairs as a simulation parameter. We assumed that shim6 contexts detect the failure after between 0 and 10 seconds (send timer) with a uniform distribution. This is realistic because different contexts in a site do not detect failure at the same time. It depends on the time that a packet has been sent out through a shim6 context. The maximum time for detecting a failure is equal to the value of send timer. We assumed that all contexts detect failure via send timer expiration. After failure, different address pairs have an equal chance to be selected as the new operational addresses.

Some research has been done on Internet Round Trip Time (RTT) [9][10][11]. Various distributions (e.g., gamma, lognormal, Pareto) have been proposed for modeling that. We tried those distributions and observed that they have no significant effect on the results. In fact, the algorithms and default values for the REAP parameters appear to minimize the effect of RTT on performance. We used a gamma distribution with average 100ms to model Internet RTT.

REAP employs two timers: send timer and keep-alive timer. We modeled these two timers and used default values for them: 10 seconds for send timer and 5 seconds for keep-alive timer, although keep-alive timer doesn't play an important role in our experiments.

We did not consider the congestion which might be cause by REAP itself, because our results show that the traffic which is generated by REAP is small compared to normal traffic, even for a large site, and can be ignored. Large sites usually have high bandwidth links to the Internet and this amount of traffic does not cause a major problem for them.

Our model assumes that all contexts are recoverable. It means that at least one operational address pair exists in the list. It also assumes that all contexts will be recovered in the first iteration that REAP scans the list of address pairs.

IV. RESULTS

In this section we present and analyze the results. We have focused on time and traffic as two important performance parameters.

A. *Recovery Time*

We measured average and total recovery time for different number of address pairs. We define total recovery time as the recovery time for the whole site, i.e., the time between a failure occurring and recovering the last context. Fig. 1 shows the average and total recovery time for the experiments with 4, 6, 9, 12, 16, 20 and 25 address pairs. The average and also total recovery time are increased when the number of address pairs is increased. The correlation is not linear because REAP uses an exponential backoff algorithm for increasing the time interval between probes. The graph demonstrates degradation in performance in the case that the number of address pairs exceeds 10. It should be noted that recovery time includes failure detection and address exploration times. In our model of the Internet, 9 address pairs seems to be the worst case. It actually models two communicating sites which both have three providers and therefore each host has three addresses.

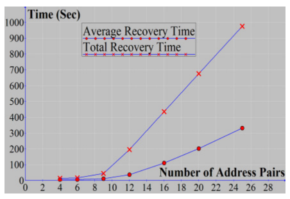

Figure 1. Average and total recovery time.

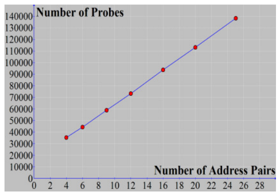

Figure 2. Total number of sent probes.

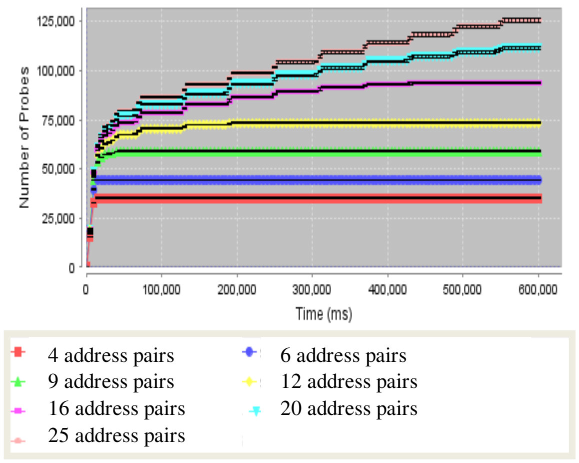

Figure 3. Number of sent probes in the first 10 minutes of the recovery phase.

We included larger numbers in our experiments to obtain a clearer view of REAP behavior.

B. *Traffic*

We measured the average and total number of probes which are sent during address exploration process. Fig. 2 shows the total number of sent probes during recovery. The results show that average number of received packets in different experiments is around 2 packets per host. Fig. 3 shows the number of sent probes in first 10 minutes of the recovery process. Fig. 2 shows that there is a linear correlation between number of address pairs and number of sent probes. Fig. 3 shows that a significant percentage of probes are sent at the start of exploration. For example, in the case of 4 address pairs 93% of the probes, and in the case of 25 address pairs 34% of the probes, are sent during the first 10 seconds. Also, there are some intervals when very few probes are sent. It can be seen more clearly for the case of 16 and 25 address pairs. It means that some contexts are in the recovery phase, but because of exponential backoff, the time interval between probes is large, so REAP instances have to wait for a long time before probing the next address pair. Some connections might be reset by transport or application layer before REAP can recover them. Fig. 4 shows the number of such connections. We used 300 seconds as a typical value for upper layer connection time-out in our experiments.

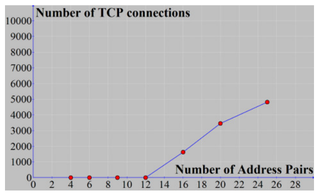

Figure 4. Number of failed connections

As the graph shows, the number of such connections grows when the number of address pairs is increased. Our experiments show that the REAP can probe 12 address pairs safely, but when the number of address pairs is more than that, the site may experience some failures. As can be seen in the graph, in case of 25 address pairs, about 50% of connections might not be able to recover before upper layer time-out.

V. Analyzing Effect of the Protocol Parameters on Performance

A. *Send timer*

We have also performed some experiments with different values for send timer to see the effect of small and large values on the performance parameters. Fig. 5, 6, 7 and 8 show the results of executing the model with 16 address pairs for different values of send timer (2, 4, 6, 8, 10, 12, 14 sec). The results show that the effect of this timer on recovery time is linear but the effect is not significant. Small values for this timer lead to a better performance but at the same time increase the chance of wrong behavior in failure detection.

B. *Initial probes*

REAP sends 4 initial probes to check for the first four address pairs in the list of address pairs and then continues with an exponential backoff algorithm. We showed in the previous section that this algorithm may cause long delays during recovery which leads to a long recovery time and therefore some connections will be reset before REAP can recover them.

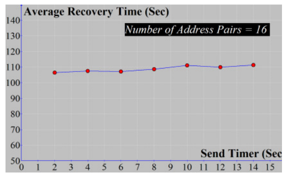

Figure 5. Average recovery time.

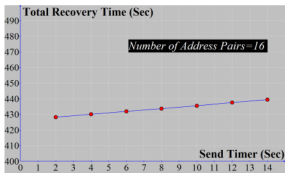

Figure 6. Total recovery time.

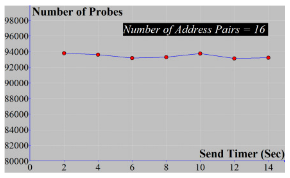

Figure 7. Total number of sent probes.

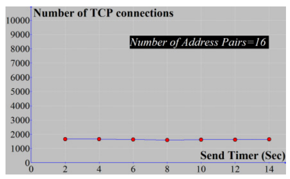

Figure 8. Number of failed connections.

We tried two different scenarios to resolve this problem: Increasing number of initial probes and sending initial probes concurrently.

Increasing number of initial probes: The default value for number of initial probes is 4. REAP sends four probes for the first four address pairs in the list and then starts exponential backoff. We tried different values for this parameter starting from 4 to number of address pairs (9). Fig. 9 shows the effect of this modification on recovery time. Increasing number of initial probes has a significant effect on recovery time. Fig. 10 shows a minor change in total number of sent probes but the traffic at the start of recovery phase is increased (Fig. 11). For example, increasing number of initial probes from 4 to 5 will cause about 7% increase in traffic in the first 10 seconds of recovery process which gradually decreases, 23% decrease in average recovery time and 34% decrease in total recovery time. It means that it can reduce the number of failed connections too. We tried the same experiment with 16 address pairs and a similar modification in number of initial probes. The results showed 37% decrease in number of failed connections.

Sending initial probes concurrently: Initial probes are sent sequentially with a time interval of 0.5 second. It seems that the recovery time will be decreased if all initial probes are sent at the same time [5]. We added this feature to the model to enable us to analyze the effect of this modification on performance.

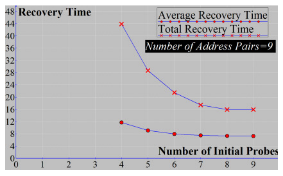

Figure 9. Average and total recovery time

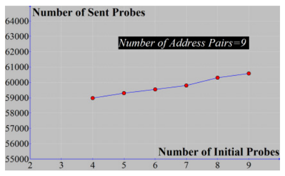

Figure 10. Number of sent probes.

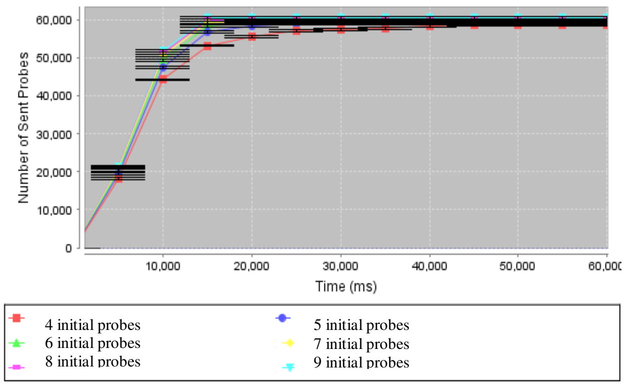

Figure 11. Number of sent probes in first 60 seconds of recovery process. (Number of address pairs=9)

Fig. 12 shows average and total recovery time for different number of address pairs. Fig. 13 shows the total number of sent probes and Fig. 15 shows the number of failed connections. The graphs show that in case of 9 address pairs, this modification will cause 11% decrease in average recovery time, 4.5% decrease in total recovery time, and 8.2% increase in traffic compared to default behavior of the protocol (Fig. 1-4).

VI. Conclusion

This paper presents the results of a performance evaluation on REAP. We built a model of REAP using SAN and simulated that model to study behavior and performance of this protocol. We focused on two important performance parameters: recovery time and traffic. The simulation results show a reduction in performance when number of address pairs is increased, especially when it exceeds 10. It also shows that the traffic which is generated in the start of recovery process can be large but it is unlikely to cause a major problem for a large scale network with a high bandwidth link. We also investigated the effect of send timer and initial probes on the performance. The results show that send timer does not have a significant effect on the performance but deferring the exponential backoff algorithm (e.g. increasing number of initial probes) and sending initial probes in a burst can improve recovery time. These modifications increase traffic but it seems that their positive effect on the recovery time is more than their negative impact.

References

[1] C. De Launois, , M. Bagnulo, The paths toward IPv6 multihoming, IEEE communications surveys, Vol 8, No 2, 2006.

[2] J. Abley, B. Black and V. Gill, Goals for IPv6 Site-Multihoming Architectures, IETF RFC 3582, 2003.

[3] E. Nordmark, M. Bagnulo, Shim6: level 3 Multihoming Shim Protocol for IPv6, IETF RFC 5533, June 2009.

[4] J. Arkko, I. van Beijnum, Failure Detection and Locator Pair Exploration Protocol for IPv6, IETF RFC 5534, June 2009.

[5] A. de la Oliva, M. Bagnulo, A. Garcia-Martinez and I. Soto, Performance Analysis of the REAchability Protocol for IPv6 Multihoming, Proceedings of the 7th international conference on Next Generation Teletraffic and Wired/Wireless Advanced Networking, 2007.

[6] J. F. Meyer, A. Movaghar, W. H. Sanders, Stochastic activity networks: structure, behavior and application, Proc. Int. Workshop on Timed Petri Nets(Torino)(Los Alamitos, CA: IEEE Computer Society Press) , 1985.

[7] D. D. Deavours, G. Clark, T. Courtney, D. Daly, S. Derisavi,J. M. Doyle, W. H. Sanders, and P. G. Webster, The Mobius framework and its implementation, IEEE Transactions on Software Engineering, vol. 28, no. 10, pp. 956–969, Oct. 2002 .

[8] R. Draves, Default Address Selection for IPv6, IETF RFC 3484, February 2003.

[9] Z. Hongli, Z. Yu and L. Ying, Modeling Internet Link Delay based on Measurement, International Conference on Electronic Computer Technology, 2009.

[10] W. Zhang, J. He,Statistical Modeling and Correlation Analysis of End-to-End Delay in Wide Area Networks, Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, 2007.

[11] C.J. Bovy, H. T. Mertodimedjo, G. Hooghiemstra, H. Uijterwaal and P. Van Mieghem, Analysis of End-to-End Delay Measurements in Internet, Proceedings of the Passive and Active Measurements Workshop (PAM2002), March 2002.

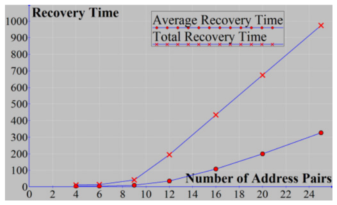

Figure 12. Average and total recovery time

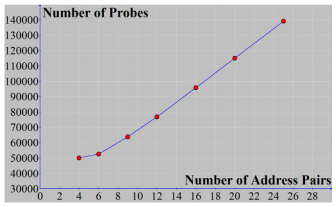

Figure 13. Total numner of sent probes.

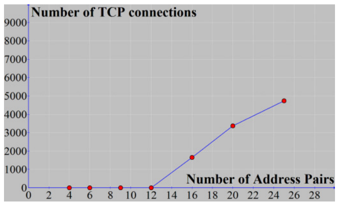

Figure 15. Number of failed connections.

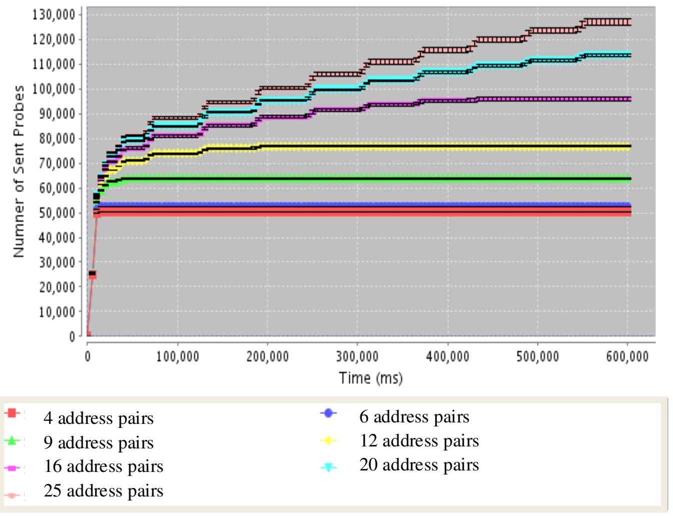

Figure 14: Number of sent probes in the first 10 minutes of the recovery process.

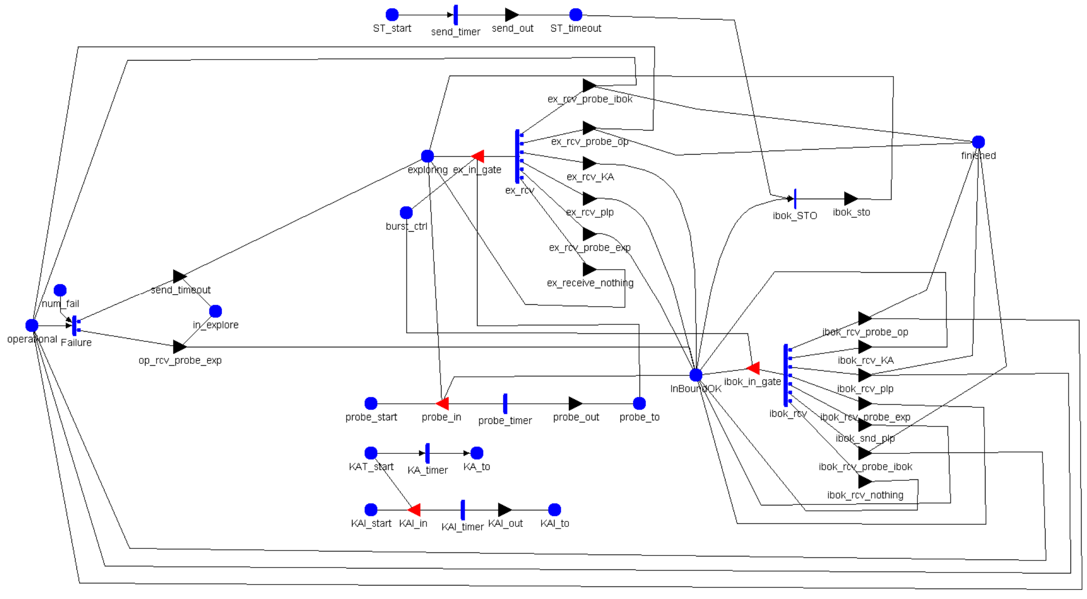

Figure 16. SAN model of the REAP