International Journal of Contemporary Research in Computer Science and Technology (IJCRCST) ISSN: 2395-5325
Volume 3, Special Issue 3 (September '2017)
Proceeding of - International Conference on Recent Trends in Computational Life Science and Information Technology,
Conference held at Tiruppur Kumaran College for Women,Tirupur,Tamilnadu, India.

MULTIPLE ROUTING CONFIGURATIONS FOR FAST IP NETWORK RECOVERY

P. Banumathi,
Assistant professor,
Department of Computer Science,
Tiruppur Kumaran College for Women,
Tirupur, Tamilnadu, India.

I. Buvana,
M.Phil., Research Scholar,
Department of Computer Science,
Tiruppur Kumaran College for Women,
Tirupur, Tamilnadu, India.

Abstract: As the Internet takes more and more central role in our communications infrastructure, the slow convergence of routing protocols when a network failure becomes a growing drawback. To assure quick recovery from link and node failures in. To assure recovery of packet from the node failures, the networks have the present scheme of recover called multiple routing schemes, in the present research they use Multi slot max clique method to handle number of packets losses in the network, in present they handle unique coverage problem for communication and they handle with optimal solution strategy technique. Our proposed scheme is based on multiple routing from the source to destination but they have assume that the hop-by-hop forwarding is better than present system, because the routing information is stored in routers and it allows packets to forward from one to another hop, thus the packet forwarding link is immediately detection of failure using our approach ACK packets, and the algorithm here followed in Optimal Packet Forward (OPF) as, it has many security and inbuilt schemes to forward the packets.

Keywords: ACK, Optimal Packet Forwarding, Multiple routing schemes, Recovery.

I.INTRODUCTION

BROADCASTING data to multiple users is widely used in many wireless applications, ranging from satellite communications to Wi-Fi networks. Wireless transmissions, however, are subject to packet losses due to channel impairments, such as, wireless fading and interference. Effectively recovering these losses could provide tremendous performance improvement to many applications, especially real-time applications, such as, fast-pace local multi-player games and live video streaming. We have experienced this problem first hand. In our previous work, we built Micro Play, one of the first networking frameworks for multi-player games that exploit Wi-Fi broadcast to achieve accurate game rendering and low latency. The rendering of a player is quicker and more accurate because it is done by leveraging directly the command packets generated by the player (that are then broadcast to the other players). When deploying the system, we experienced broadcast losses, and our measurements showed that if we could recover even a small percentage of packet losses (less than 1%) in a timely manner, then our game rendering engine would benefit tremendously, i.e., animation jitters could be completely eliminated.

As another example, Crowd Wi-Fi by Stream bolico is one of the first commercial systems that exploit Wi-Fi broadcast to stream live videos to a large number of users, such as, crowds at stadiums, concerts, and conferences. Crowd Wi-Fi is able to achieve 11x bandwidth improvement over traditional uni-cast solutions. The loss recovery algorithm used by Crowd Wi-Fi is the key contributor to its superior performance: it has helped the system to keep the number of packets that miss their playback deadline low even when the loss rate is high. (The number of packets that miss deadline is kept negligible even when the wireless loss rate is up to 10%.). Motivated by the importance of loss recovery in wireless broadcast for real-time applications, in this work, we aim to find the best coding scheme to recover packet losses.

Causes of Packet Loss

Packet loss happens when packets are damaged and discarded, or when the ability of an intermediate network element is goes above, which results in packets being discarded. Packets can be damaged as they move across a wide area network, or as they pass through network method such as routers and switches. This type of damage is identified by a failure in "checksum" processing. The checksum is a mathematical sum of bits that is computed by the sender and appended to both packets. The receiver also computes the checksum and compares its calculated value to the value received with the packet. If the received and calculated checksums do not match, the receiver drops the packet. Despite of the network topology, there is forever a chance that some level of packet loss may happen due to checksum detected errors, particularly as larger numbers of routers and switches are traversed, or over telephone service links casing the "last mile" of a network connection.

When the capacity of an intermediate network component is exceeded, congestion occurs at that component and packets will be discarded. For example, if packets arrive at a router at a rate fasterthan the router can store them or transmit them, some number of packets will be discarded by that router. Service level policies can be in place, guaranteeing specific service levels to particular groups of applications. In order to meet these service levels, routers will intentionally discard packets from data streams that are outside these groups of applications.

IJCRCST © 2017 | All Rights Reserved

Packet Loss Is Real

All IP service providers monitor their networks and report various statistics on average latency and packet loss rates. Latency is expressed in terms of milliseconds, and represents the time it takes a packet to travel from source to destination and back. Latency includes both the transmission time on the physical media, and the time it takes a packet to travel through routers and switches on the network. Packet loss is expressed in terms of a percentage, which represents the average packet drop rate. For example, a 1% packet loss rate (which would be extremely severe) indicates that 1 of every 100 packets is being dropped. More realistic packet loss rates are on the order of 0.1% or 0.01%, which indicate that 1 of every 1,000 or 1 of every 10,000 packets are dropped.

Unlike IP applications such as e-mail or web serving, storage replication applications are mission-critical, and involve sending large amounts of data over long periods of time. The chance of a replication running during one or more of these degraded time intervals sometime during the month is probably quite high, since replication applications tend to be characterized either by the use of long standing connections, or by connections that reoccur at predefined time intervals. In either case, storage replication applications use TCP/IP connections for long periods of time, either as one long-standing connection, or the sum of multiple shorter connections. The amount of data that gets sent over the storage replication connections actually depends on the methodology of the replication being performed.

TCP and Packet Loss

In order to satisfy required performance levels, it is crucial that the data transport used by storage replication applications over a wide area network can recover from packet loss conditions as efficiently as possible. When using TCP/IP as the data transport, there are two indications that packet loss has occurred: either a timeout occurs on a segment before receiving an acknowledgement for that segment; or duplicate acknowledgements are received. Regardless of whether the packet loss was caused by intentional discard due to congestion, or by actual data corruption, TCP/IP interprets both of these indications as congestion existing somewhere in the network between the source and the destination, and reduces its throughput

Congestion Avoidance

In congestion avoidance, the window is opened linearly by increasing the size of the congestion window by a maximum of one segment for each round trip time. This allows the sender to slowly increase its transmission rate as it approaches the point where congestion had previously occurred.

II.RELATED WORK

Packet loss often occurs in the Internet when a router becomes congested, i.e. it receives more packets to forward than it can process. Large loss bursts (outages) also occur when network pathologies exist, i.e. a router or a link fails. However, this problem is orthogonal to congestion and belongs into the (QoS) routing domain (the routing must re-converge to paths around the point of failure). Another reason for loss can b e transmission errors (bit errors) of the underlying medium. Typically the bit error ratio is extremely low however for fixed networks (but it can b e significant for wireless networks). For this thesis we only consider losses that are intrinsic to the functional blocks at the end-to-end and hop-by-hop level and thus we use the term packet loss for losses caused by congestion only. Packet loss in the Internet is a frequent and also the most serious problem that speech transmissions over the Internet have to face. In order to provide an acceptable quality, loss recovery / control must be performed.

The segment length should b e chosen to b e relatively short, such that the speech signal can be assumed to be stationary for one segment with a high probability.

If very small packets are transmitted, annoying noise impulses will occur. Additionally, the packet header overhead as well as the extreme per packet processing cost within the network is prohibitive. A large segment length in connection with packet loss may impair the speech intelligibility due to the loss of entire phonemes. Obviously for frame-based co decs this choice of the segment length is equivalent (and limited) to the choice of the number of frames per packet.

Authors: Baruch Awerbuch, Reza Curtmola, David Holmer, Cristina Nita-Rotaru y and Herbert Rubens Says that, a common scheme used by routing procedure for ad hoc wireless networks is to make the routing paths on-demand, as opposed to frequently continue a whole routing table. Since in an ad hoc system nodes not in straight range communicate via intermediate nodes, a significant concern is the capability to route in the attendance of Byzantine failures which include nodes that drop, fabricate, modify, or mis-route packets in an attempt to disrupt the routing service. We propose the first on-demand routing protocol for ad hoc wireless networks that provide resilience to Byzantine failures caused by individual or colluding nodes.

The protocol relies on an adaptive probing method that identifies a malicious link following log n faults have happen, where n is the length of the path. Difficult links are avoided by using a weight-based method that multiplicatively enlarges their weights and by using an on-demand route discovery protocol that discover a least weight path to the destination. Our protocol bounds the amount of damage that an attacker or a collection of colluding attackers can cause to the network. The aim of this work is to give routing survivability under an adversarial model where any middle node or group of colluding nodes can execute Byzantine attacks such as creating routing loops, misrouting packets on non-optimal paths, or selectively dropping packets; only the start and the end nodes are assumed to be trusted.

To our facts, there is no on-demand wireless routing procedure addressing Byzantine failures, mainly in a model where attackers can collude. A Byzantine fault happening along a path may be credited to a exact node using expensive and complex Byzantine agreement; though, this is provably impossible below certain circumstances, for example when a majority of the nodes are malicious. We circumvent this obstacle by avoiding the assignment of "guilt" to individual nodes. Instead, whenever the endpoints

of a link disagree, we deduce that at least one of them is faulty, therefore the link is considered faulty and should be avoided. Our method ensures that as long as a fault-free path exists between two nodes, they can communicate reliably even if an overwhelming majority of the network acts in a Byzantine manner. This work is an enhanced version of our previous work [8]. In this paper, we present a route discovery phase that is based on aggregate signatures [9], as opposed to chained digital signatures. This approach provides stronger security guarantees and improved convergence times. The fault detection phase has been redesigned so that onion encryption is no longer required, thus reducing computational cost and packet size overhead. We also describe an efficient scheme for securely boot strapping shared keys from an existing public key infrastructure.

Security Model and Considered Attacks In this work we consider only the source and destination to be trusted. Nodes that cannot be authenticated do not participate in the protocol and are not trusted. Any intermediate node on the path among the start and end point can be validated and is expected to execute the protocol correctly, but may exhibit Byzantine activities.

If the losses violate the acceptable threshold, the protocol registers a fault between the source and the destination and starts a binary search on the path, in order to identify the faulty link. The list of probes illustrates a set of non-overlapping intermission that covers the entire path, where each interval wraps the sub path among the two repeated probes that form its endpoints. When a fault is identified on an interval, the interval is divided in two by put in a new probe. This new probe is added to the list of probes added to future packets. The process of sub division continues until a fault is detected on an interval that corresponds to a single link.

$$\text{seq,data,ID}_1,\text{ID}_2,\text{ID}_3,\ldots,\text{ID}_p,\text{HMAC}_{dest},\text{HMAC}_p,\ldots,\text{HMAC}_3,\text{HMAC}_2,\text{HMAC}_1$$
where HMAC_i is computed with K_{source,ID_i} and p is the number of probes

Equation: Probe Specification

The probes are specified by listing the identifiers of the probed nodes in path order on each packet. This list is protected from tampering by appending an HMAC [14] (computed with the shared key between the source and that node) of the entire packet so far for every probed node in the reverse order they are specified on the packet. A node can detect if it is required to send an ack by checking the list for its identifier. If the node finds its identifier, it verifies the last HMAC and removes it from the packet. A node discards packets that do not have the correct number of remaining HMACs, and a probe discards packets if the last HMAC does not verify. The reverse planned HMACs avoid the adversary from incriminating other relations by successfully tampering with the node list (i.e. take away specific nodes). If an opponent attempts to change the list, it will incriminate one of its own links.

Acknowledgment Specification: For each successfully received data packet, the destination generates an ack packet containing the sequence number of the data packet and an HMAC for authentication. Each probe appends its own HMAC to the ack and forwards it along the reverse path towards the source. Timeouts are set at every probe so that if the ack is not received, the node gives up waiting and generates its own ack packet.

III.BACKGROUND STUDY

PROBLEMS IN PRESENT SYSTEM

Link error and malicious packet dropping are two bases for packet losses in multi-hop wireless ad hoc network. Even as examining a sequence of packet losses in the network, whether the losses are caused by link errors only, or by the mutual effect of link errors and malicious drop is to be recognized. In the insider-attack case, whereby cruel nodes that are part of the route utilize their facts of the communication condition to selectively drop a small quantity of packets serious to the network performance. Since the packet dropping rate in this case is comparable to the channel error rate, conventional algorithms that are based on identifying the packet loss rate cannot achieve satisfactory recognition exactness. To develop the detection correctness, the correlations among lost packets is recognized. Homomorphic linear authenticator (HLA) based public auditing construction is developed that permits the detector to prove the truthfulness of the packet loss data reported by nodes. This construction is privacy preserving, collusion proof, and incurs low communication and storage overheads. To decrease the calculation overhead of the baseline design, a packet-block based system is also proposed, which allows one to trade finding accuracy for lower calculation complexity.

Identifying selective packet-dropping attacks is really challenging in a extremely dynamic wireless environment. The difficulty comes from the obligation that we need to not only identify the place (or hop) where the packet is dropped, but also recognize whether the drop is intentional or unintentional. Particularly, due to the open nature of wireless intermediate, a packet drop in the system could be caused by harsh channel situations e.g., fading, noise, and interference, link errors, or by the insider attacker. In open wireless surroundings, association errors are fairly important, and may not be considerably minor than the packet dropping rate of the insider attacker. So, the insider attacker can camouflage under the background of harsh channel environment. In this case, just by observing the packet loss rate is not sufficient to exactly identify the correct reason of a packet loss.

IV.PROPOSED METHODOLOGY

Fast packet switching (or its variants like ATM) are now broadly accepted as the universal technique for constructing high-speed multi-media communication networks. Due to the high throughput demands and the mixed traffic environment, these networks usually employ simplified and universal congestion control mechanisms which are based on call bandwidth reservation, input rate enforcement and packet discard at the intermediate nodes. Understanding the packet loss behavior is crucial for the proper design of the real time (e.g. voice and video) coding and playback mechanisms and the methods of congestion control and error recovery for data. It is alsoessential for sizing the various buyers at the transmission links and switching elements. Individual packet loss probabilities (where in our

terms a packet is the integral transmitted quantity) are usually not sufficient for proper understanding of the system behavior. In general, packets are more likely to be rejected because of buffer overflow as their rate of arrival to the buyer increases.
Also, if a packet has been rejected, then it is more likely that consecutive packets will also be rejected1. Consequently, it is clear that there is a strong correlation between consecutive packet losses, and therefore losses are bursty. It is well known that the bursty nature of the packet loss process can effectively reduce the service quality, especially for sources which are sensitive to long bursts of losses (such as voice, video and some error recovery techniques employed for data). It is less known that some sources may actually benefit from the bursty nature of the loss process.

A transmitter can send video packets to the receiver using the default Internet path or by means of a relay node which then forwards the video packets to the receiver. By choose a correct relay node, the packets pass through an underlying physical path that is dissimilar from the one used by default Internet path. Nodes can be deployed at various locations on the Internet, although redundant paths via nodes in the same AS may have larger number of shared links. Hence, nodes are preferably located in different AS to reduce correlated congestion and outages so as to improve the performance of the OPF system.

A participating node which is neither a receiver nor a sender, may still receive and forward packets on behalf of other senders and receivers. For two way applications such as video conferencing, they act simultaneously as both senders and receivers. In addition, the senders and receivers do not essentially have to be the contributed nodes. We predict the contributing nodes as an overlay network to be deployed by companies or organizations that are involved in providing low delay communication services such as video streaming or conferencing over the Internet between their geographically diverse sites. Companies and organizations typically have multiple gateways from their ASes to other ASes that potentially enable large number of independent paths
To set up a communication channel between the sender and receiver, the sender first executes trace route from itself to all the participating nodes and the receiver. The information returned from the trace route includes the link latencies, and the names of the routers along the default path from the sender to the receiver and all paths between the sender and the participating nodes. The sender also sends a setup packet to all contributing nodes instructing them to complete sketch route from themselves to the receiver. After that, all the contributing nodes send the path information among themselves and the receiver get from trace route to the sender. The sender now has the names of the routers and their related link latencies for the default path, the paths among itself to all participating nodes, and the paths among contributing nodes to the receiver.

After the unnecessary path via a selected relay node is chosen, the sender sends a setup packet to the selected relay node, instructing it to forward packets to the receiver on behalf of the sender from then onward. The setup packet contains a flow ID, IP address, and the port number of the receiver. Upon receiving the setup packet, the relay node stores an entry containing a flow ID, an IP address, and a port number of the receiver in a table. This table is used to forward packets on behalf of different senders to their receivers. Each packet sent from a different sender to the relay node contains a different flow ID in its header. The relay node forwards a packet to the right IP address and port number of the receiver based on its flow ID. Note that all the setup messages and executions of trace route are done only at the start of the session.

Let us formally denote a network topology as directed graph G = (V,E) consisting of the vertices vi ∈ V and edges e = (vi, vj) ∈ E. Vertices vi's can be thought of as routers or domains, and the path p(v1, vn)=[v1, v2, ..., vn] as the physical path from v1 to vn. A redundant path p (v1, vk, vn) from v1 to vn, via node vk is then p(v1, vk) ∪ p(vk, vn). Related with each pair of vertices (vi, vj) is a weight w(vi, vj). This weight can be thought of as delay, bandwidth, or loss rate associated with the physical link between vi and vj . Hence, the weight w(p(vk, vl)) associated with a path from vk to vl is the sum of the weights of the individual physical links joining vk and vl. In this paper, weights indicate the latencies among participating nodes since they are readily available from trace route. Let O = [vm, ..., vn] be the set of contributing nodes; then the relay node k for making the Redundant path between vertices u and v is computed in a two-step procedure.

1) First, we compute a set of relay nodes O that result in the minimum number of joint links between the default Internet path and all the redundant paths via a node in O, namely, O = arg mink p (u, k, v) ∩ p∗ (u, v) where k ∈ O, p (u, k, v) denotes the redundant path via node k, and p∗ (u, v) signify the default Internet path. Note that the set O can have more than one element since there could potentially be two or more nodes in O that result in the same least number of joint links between the default and redundant paths. Since the average number of links between two nodes on the Internet is 16 [9], the operation p (u, k, v) ∩ p∗ (u, v) can be done fast. To find arg mink p (u, k, v) ∩ p∗ (u, v), exhaustive search for the set of nodes O that results in minimum number of joint links between the redundant and default paths can be done in O(N) with N being the number of participating nodes.
2) Next, we choose the node k that results in minimum weight associated with the corresponding redundant path, namely, k = arg minl w(p (u, l, v))

V. CONCLUSION

A lot of proposed error control mechanisms recover lost frames by sending additional information alone with the original data. Thus, these error control mechanism s put more traffic into wireless networks and make the small bandwidth wireless network have more chance to be congested. Thus, some alternative methods can be devised to achieve better network resource utilization. Our work, which ado pts the hybrid approach, differs from all of the above works. In small bandwidth wireless networks, we ad

opt a RTT-based flow control mechanism that can dynamic ally use different combinations of audio codes according to network stat us in order to avoid the network from being congested . Though for real-time media streaming applications that are delay sensitive and where multicasting is used, such procedures cannot be used. Packet losses can happen without a complete failure of a system. These losses happen for many causes and differ based on type of the network association. For a well control, guaranteed bandwidth type connection, the packet loss rate raise can happen due to routing modifies as a result of an individual link failure or a link (possibly terrestrial or satellite) maybe degraded. For a "best effort" type service, packet loss may be a result of obstruction in the network.

VI.REFERENCES

[1] G.Ghinta, S.Sultana, "A Lightweight Secure Scheme for Detecting Provenance Forgery and Packet DropAttacks in Wireless Sensor Networks".

[2] A. Ghani and P. Nikander, "Secure In-Packet Bloom Filter Forwarding on the Netfpga," Proc. European NetFPGA Developers Workshop, 2010.

[3] C.Rothenberg,C.Macapuna,M.Magalhaes, F.Verdi, A.Wiesmaier,In-Packet Bloom Filters: Design Networking Application, Computer Networks,vol.55,no,pp. 1364-1378,2011.

[4] S. Sultana, E. Bertino, and M. Shehab, "A Provenance Based Mechanism to Identify Malicious Packet Dropping Adversaries in Sensor Networks," Proc. Int'l Conf. Distributed Computing Systems (ICDCS) Workshops, pp. 332-338, 2011.

[5] W. Zhou, M. Sherr, T. Tao, X. Li, B. Loo, and Y. Mao, "Efficient Querying and Maintenance of Network Provenance at Internet-Scale," Proc. ACM SIGMOD Int'l Conf. Management of Data, pp. 615-626, 2010.

[6] B. Awerbuch, R. Curtmola, D. Holmer, C. Nita-Rotaru, and H.Rubens, "ODSBR: An on-demand secure byzantine resilient routing protocol for wireless ad hoc networks," ACM Trans. Inf. Syst.Secur., vol. 10, no. 4, pp. 11–35, 2008.

[7] K. Balakrishnan, J. Deng, and P. K. Varshney, "TWOACK: Preventing selfishness in mobile ad hoc networks," in Proc. IEEEWireless Commun. Netw. Conf., 2005, pp. 2137–2142.

[8] S. Buchegger and J. Y. L. Boudec, "Performance analysis of the confidant protocol (cooperation of nodes: Fairness in dynamic adhocnetworks)," in Proc. 3rd ACM Int. Symp. Mobile Ad Hoc Netw.Comput. Conf., 2002, pp. 226–236.

[9] Tao Shu and Marwan Krunz "Privacy-Preserving and Truthful Detection of Packet Dropping Attacks in Wireless Ad Hoc Networks" IEEE Transactions on Mobile Computing DOI:10.1109/TMC.2014

[10] K. Balakrishnan, J. Deng, and P. K. Varshney "TWOACK: preventing selfishness in mobile ad hoc networks" In Proceedings of the IEEE WCNC Conference, 2005.

[11] G. Ateniese, S. Kamara, and J. Katz " Proofs of storage from homomorphic identification protocols" In Proceedings of the International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT), 2009.

[12] Q. He, D. Wu, and P. Khosla "Sori: a secure and objective reputation based incentive scheme for ad hoc networks" In Proceedings of the IEEE WCNC Conference, 2004.

[13] S. Zhong, J. Chen, and Y. R. Yang. "Sprite: a simple cheat-proof, credit based system for mobile ad-hoc networks" In Proceedings of the IEEE INFOCOM Conference, pages 1987–1997, 2003.

[14] W. Kozma Jr. and L. Lazos "REAct: resource-efficient accountability for node misbehaviour in ad hoc networks based on random audits" In Proceedings of the ACM Conference on Wireless Network Security (WiSec), 2009.

[15] W. Galuba, P. Papadimitratos, M. Poturalski, K. Aberer, Z. Despotovic, and W. Kellerer. "Castor: Scalable secure routing for ad hoc networks." In INFOCOM, 2010 Proceedings IEEE, pages 1 –9, march 2010.

[16] T. Hayajneh, P. Krishnamurthy, D. Tipper, and T. Kim. "Detecting malicious packet dropping in the presence of collisions and channel errors in wireless ad hoc networks." In Proceedings of the IEEE ICC Conference, 2009.

[17] W. Kozma Jr. and L. Lazos. "Dealing with liars: misbehavior identification via Renyi-Ulam games." In Proceedings of the International ICST Conference on Security and Privacy in Communication Networks (SecureComm) , 2009.

[18] A. Proano and L. Lazos. "Packet-hiding methods for preventing selective jamming attacks." IEEE Transactions on Dependable and Secure Computing, 9(1):101–114, 2012.

[19] Q. He, D. Wu, and P. Khosla. Sori: a secure and objective reputation-based incentive program for circumstantial networks. In Proceedings of the IEEE WCNC Conference, 2004.

[20] D. B. Johnson, D. A. Maltz, and J. Broch. DSR: the dynamic supply routing protocol for multihop wireless circumstantial networks. Chapter 5, circumstantial Networking, Addison-Wesley, pages 139–172, 2001.

[21] W. Kozma Jr. and L. Lazos. coping with liars: misconduct identi-fication via Renyi-Ulam games. In Proceedings of the International ICST Conference on Security and Privacy in Communication Networks (SecureComm), 2009.

[22] W. Kozma Jr. and L. Lazos. REAct: resource-efficient answerability for node misconduct in circumstantial networks supported random audits. In Proceedings of the ACM Conference on Wireless Network Security (WiSec), 2009.