SH-P2P: Self-Healing Peer-to-Peer Network with Optimal Multicast Routing

Maher Kaddoura
Architecture Technology Corporation
Eden Prairie, USA
mkaddoura@atcorp.com

Nathan Bahr
Architecture Technology Corporation
Eden Prairie, USA
nbahr@atcorp.com

Erin Gambucci
Architecture Technology Corporation
Eden Prairie, USA
egambucci@atcorp.com

2022 IEEE International Conference on Electro Information Technology (eIT) | 978-1-6654-8009-3/22/$31.00 ©2022 IEEE | DOI: 10.1109/eIT53891.2022.9813897

Abstract*—Applications such as blockchain systems use Peer-to-Peer (P2P) networks to multicast data among a large number of nodes. P2P multicast networks are designed for enterprise networks, which experience infrequent topology changes. As a result, current P2P multicast networks are not suited for intermittently connected networks. This paper presents an innovative approach called Self-Healing Peer-to-Peer (SH-P2P) that enables P2P multicast networks to operate efficiently in challenging network settings. SH-P2P allows edge networks to stay connected to the backbone network using the most optimal links. Also, it enables the backbone network to self-heal when partitions occur and to maintain optimal routing for distributing data among its members. The SH-P2P design is presented as are experimental results that characterize SH-P2P's performance.

***Keywords*—P2P networks, Self-healing networks, Edge networks, Multicast routing**

I. Introduction

Peer-to-Peer (P2P) networking has been deployed in distributed systems to support various applications, such as file sharing, multimedia streaming, and parallel computation [6]. P2P also serves as the underlying communication layer for blockchain systems that use it to disseminate data (messages and blocks) among all blockchain nodes. Blockchain systems need to maintain connectivity among their members even when some nodes fail or disappear, and the underlying P2P network needs to determine parameters like link bandwidth, delay, and loss in order to provide efficient and scalable data distribution.

Several application-layer multicast protocols [2,9,10] have been developed for both structured and unstructured overlays. However, these protocols rely upon inefficient techniques, such as redundant subtrees and message flooding, that incur considerable overhead and delays when the rate of topological change within the network is high. These protocols cannot handle the disappearance of many nodes; they can only handle the disconnection of a small number of nodes, such as a subtree parent. Also, P2P multicast protocols construct multicast trees to optimize traffic flow from a single source to the network members [11].

Current P2P multicast networks for blockchain systems are designed to operate in stable and well-connected networks [1,7,8,12]; thus, they fail when deployed in mobile wireless networks.

This paper presents a new approach for P2P multicast networking called Self-Healing Peer-to-Peer (SH-P2P). We are developing SH-P2P to provide the communication layer for a new blockchain that is specifically designed for the challenges of intermittently connected networks.

SH-P2P utilizes an efficient link-sensing tool to monitor overlay links between its peers. Also, it uses an innovative mechanism to reconnect the network when partitions occur, which heals network partitions dynamically. The self-healing process can reconnect the network even when a larger portion of the network disappears. SH-P2P builds the P2P network topology using messages that are similar to those used by the Open Shortest Path First (OSPF) routing protocol, which has been deployed successfully in wireless and mobile networks [4]. Finally, SH-P2P uses graph algorithms to maintain optimal routing in the network after it fixes a partition [3,13].

This paper is organized as follows. In Section II, we describe the SH-P2P network architecture. Section III presents the edge node self-healing mechanism. Section IV describes the core network self-healing mechanism. Section V presents the SH-P2P prototype implementation. Finally, the conclusion is presented in Section VI.

II. P2P Overlay Network Architecture

Figure 1 depicts the SH-P2P network architecture, which consists of two major parts: i.) a core network (i.e., backbone network) composed of high-performance nodes (e.g., data centers) connected by high-bandwidth and relatively stable links, and ii.) Edge Networks (ENs) that are connected by low-bandwidth links to the backbone network. An EN may disconnect from the backbone for undetermined periods.

Logistics networks are examples of the SH-P2P network architecture. These networks include distribution centers and warehouses connected by high-speed backbone links. They also include transportation companies, customers, and vendors connecting to the backbone network over intermittent low-bandwidth connections. For example, a truck making its way through the streets of a city periodically reports its GPS position to a center where analysts can view the truck's position and track its route on a map. This architecture is especially useful for tactical military operations where edge networks commonly disconnect from the grid (backbone network) while performing a mission.

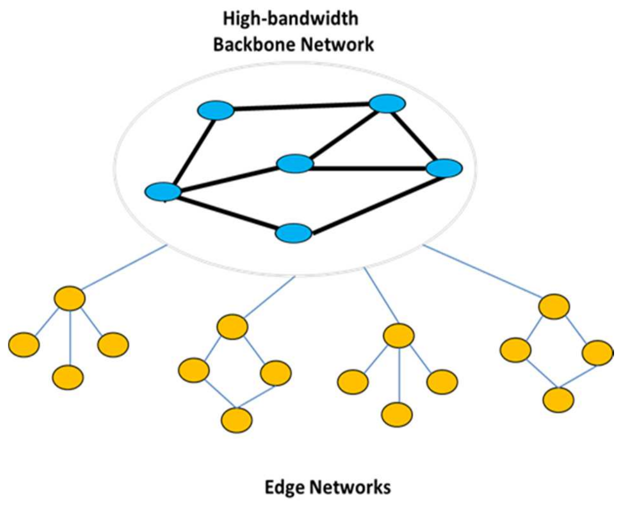


Figure 1 Network Architecture

A. End-to-End Link Sensing Service

The Link Sensing Service (LSS) lets an endpoint probe the link quality metrics between itself and one or more endpoints. Each endpoint is probed individually and is agnostic to the connection in between. For example, the endpoint could be one hop away on the same hardware switch or connected over the Internet; as long as both endpoints can communicate, LSS can probe for the link quality between the endpoints.

The LSS probes endpoints for link bandwidth, latency, and packet loss via several methods. For latency, LSS will periodically send a small probe packet to the endpoint, which is echoed back to the sender by the endpoint. Packet processing time is accounted for, and the round-trip time (RTT) is calculated. Latency probes are insignificant, even for links with very few resources, and therefore remain at a static and consistent probe interval.

For packet loss, the LSS marks certain packets such that the endpoint can determine which packets have been dropped by the network. The endpoint then periodically reports the percentage of packet loss that it has calculated to the sender. This method also has an insignificant impact on the link and is performed at a fixed and consistent interval.

For bandwidth, the LSS utilizes a packet-train method of probing. The service will queue up a significant amount of data that is sent all at once so that the endpoint can measure the bandwidth based on the time difference between receiving the first and last packet of the train. This method can incur a high link-utilization cost, which LSS avoids by intelligently minimizing the size and interval of the bandwidth probes. The LSS automatically adjusts the size of the bandwidth probe based on the current detected bandwidth to reduce the probe size while retaining the ability to measure the bandwidth accurately. LSS will also automatically adjust the bandwidth probe intervals based on the stability of the link. The LSS will detect times of fluctuation in the link and responds in kind to provide accurate and up-to-date link quality metrics to the user.

B. Topology Set Up

The ENs and the core network will set up and manage their P2P networks independently, which improves scalability. SH-P2P uses three mechanisms to set up a well-balanced Peer-to-Peer (P2P) network. Nodes can:

1. Be preconfigured with their one-hop neighbors.

2. Send a discovery packet across the network, asking nodes within a certain distance to respond directly. This method works only if the edge and backbone networks support multicast. Also, this method might fail in the backbone network if the peers are distributed across autonomous systems because the gateways that connect the autonomous systems will drop the multicast messages.

3. Use rendezvous nodes to discover peers. In this method, peers utilize the rendezvous nodes to discover all the nodes in their network. They then establish direct overlay links among themselves, which results in a fully connected network. The peers build a complete weighted graph of the network. Using SH-P2P's LSS, the peers measure and exchange network metrics (e.g., bandwidth, delay, and packet loss), resulting in a graph that describes the network topology with labeled edges between any two directly connected nodes. A single node uses the graph to compute the final topology, which is subsequently distributed throughout the network.

SH-P2P computes a Minimum-weight Spanning Tree (MST) based on the application requirements to build the final topology. For example, if an application sends many messages, SH-P2P will compute MST to optimize bandwidth between the peers. On the other hand, if an application is sensitive to delays, the MST will be computed to minimize delay across the network.

III. Edge Node Self-Healing

Each EN maintains an optimal connection to the core network. Also, each EN selects a peer to be the leader of its network. The leader is responsible for establishing and maintaining communication with the core network. The EN's leader monitors its existing links to the core peers. If a link to a particular peer gets disconnected or becomes unstable, the leader selects a new peer with a superior link. For example in Figure 2, the Edge1 network is connected to Peer A, which provides Edge1 with an optimal connection to the core network. When the link bandwidth between Edge1 and Peer A degrades, the Edge1 leader will detect that its link to the core network has become unstable and search for a new link. In Figure 2, it selected Peer B to connect to the core network.

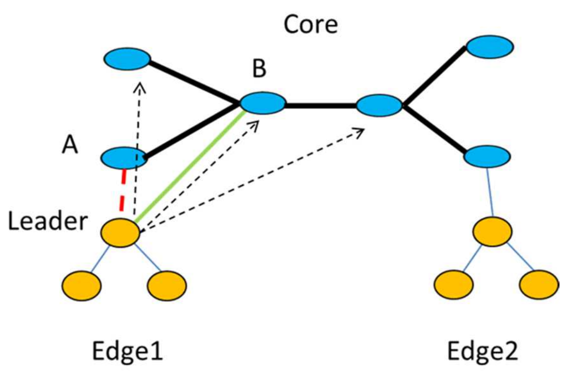
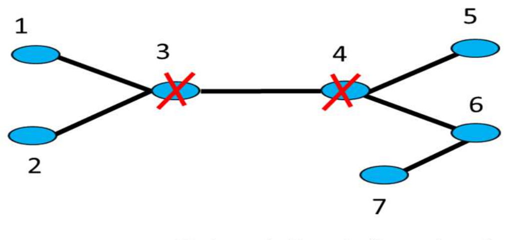


Figure 2 EN Maintains Optimal Connection to the Core Network

IV. Core Network Self-Healing

SH-P2P utilizes an innovative technique for optimally healing network partitions. Each peer builds a graph that depicts the network's topology. This is achieved by exchanging messages among peers, similar to the OSPF routing protocol. A peer detects a network partition when its 1-hop neighbors disappear. Then, the peer tries to repair the network by establishing links to nodes that are 1-hop from the disappearing nodes. If it fails to connect to a particular peer that is one 1-hop from a disappearing peer, it will try to connect to all the 2-hop peers from the disconnected peer and are connected to the 1-hop peer. The peer keeps increasing the hop distance until it reconnects the network or exhausts all the peers in the disconnected partition.

Figure 3 depicts an example of a P2P network and the network topology available to each peer in the network. Let's assume that Peer 3 and 4 vanish from the network, which will cause a partition in the network. Peer 1, 2, 5, and 6 will detect the partition and try to reconnect the network. For example, Peer 1 will first try to connect to Peer 2 and 4. When Peer 1 fails to connect with Peer 4, it will try to connect with Peer 5 and 6. Because Peer 1 can connect with Peer 5 and 6, it will stop its process of healing the network. At the end of its healing process, Peer 1 is connected to Peer 2, 5, and 6. Similarly, Peer 2 connects with Peer 1, 5, and 6; Peer 5 connects with 1, 2, and 6; Peer 6 connects with 1, 2, and 5.



Network Graph (topology)

Core 1: [3]
Core 2: [3]
Core 3: [1,2,4]
Core 4: [3,5,6]
Core 5: [4]
Core 6: [4,7]
Core 7: [6]

Figure 3 Network Topology

Figure 4 (a) shows the P2P network after the network healing process. The process reconnects the four partitions into a single P2P network. However, the new network has redundant links that increase the overhead of distributing messages among the peers. To optimize the network after it heals from a partition, SH-P2P computes a MST based on the application requirements (see Figure 4 (b)).

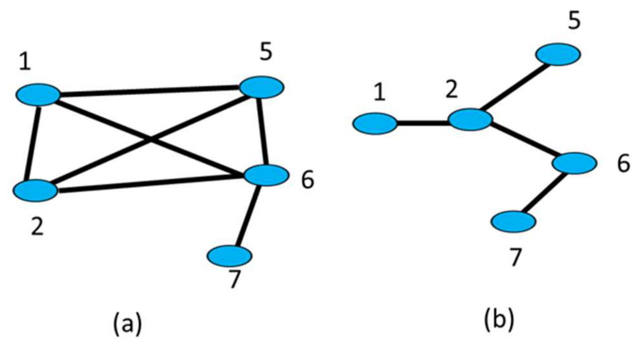


Figure 4 P2P Network after Healing

V. Prototype

In this section, we describe demonstrations of SH-P2P that were performed using a prototype implementation of the protocol in a laboratory testbed. These demonstrations highlight the major capabilities of SH-P2P: 1) detecting node disconnects; 2) self-healing when partitions occur in the core network; 3) maintaining optimal topology; and 4) enabling edge nodes to maintain optimal connections to the core network.

A. API and Display

A SH-P2P API has been developed along with a REST web service for integrating SH-P2P with third-party applications. We selected ZeroMQ [14] to transmit data between the P2P network and the REST web service. A Java display application was implemented so that the SH-P2P network topology could be graphically presented. As depicted in Figure 5, the data flow goes from the display, through the REST web service, and through the Broker to a peer that responds to the Broker's request. The request is then sent back through the same path and is ultimately presented by the Java display.

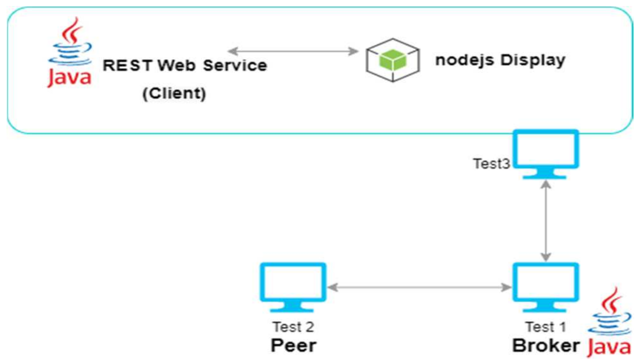


Figure 5 API Data Flow

B. Performance

The prototype was deployed and tested within ATCorp's Cyrin virtual testbed [5]. The seven-node network topology depicted in Figure 6 was used to measure SH-P2P performance.

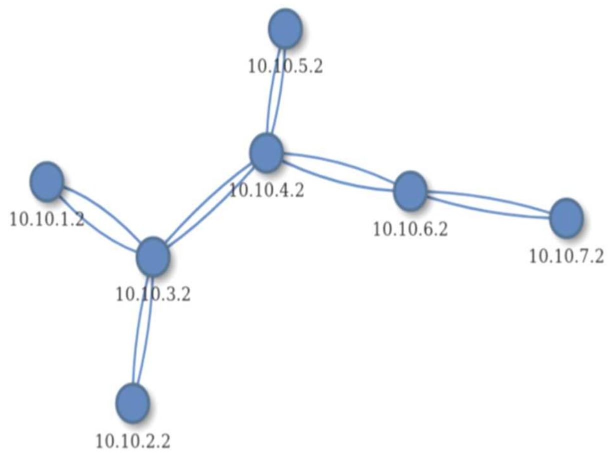


Figure 6 Testbed Network

Using the above topology, node 10.10.3.2 and node 10.10.4.2 were disabled; P2P network healing was measured at 40 seconds and resulted in the topology shown in Figure 7 (a). After 140 seconds, the P2P network optimized its routing to make the topology that is shown in Figure 7 (b).

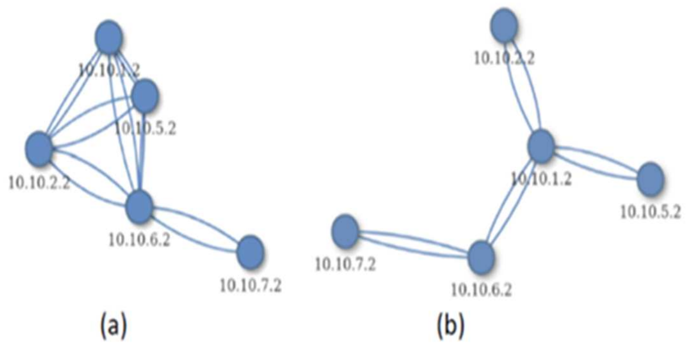


Figure 7 Network after Recovering from Node 10.10.3.1 and 10.10.4.2 Failure

In the test described above, node 10.10.1.2 generated traffic at constant intervals. Figure 8 depicts the traffic at node 10.10.5.2. When the partition occurred, node 10.10.5.2 got disconnected for 40 seconds. When the network self-healed, node 10.10.5.2 had three peers, while initially it had one. As a result, node 10.10.5.2 experienced a spike in the amount of traffic passing through it. However, after 140 seconds, SH-P2P optimized its routing, which reduced the traffic passing through node 10.10.5.2.

After a partition has occurred, SH-P2P waits for 60 seconds before topology optimization; this tunable wait interval prevents route flapping. During this time interval, the topology's stability is tested. If the topology is deemed unstable, SH-P2P waits for another 60 seconds and retests. Once the topology is stable, it computes an optimal new topology.

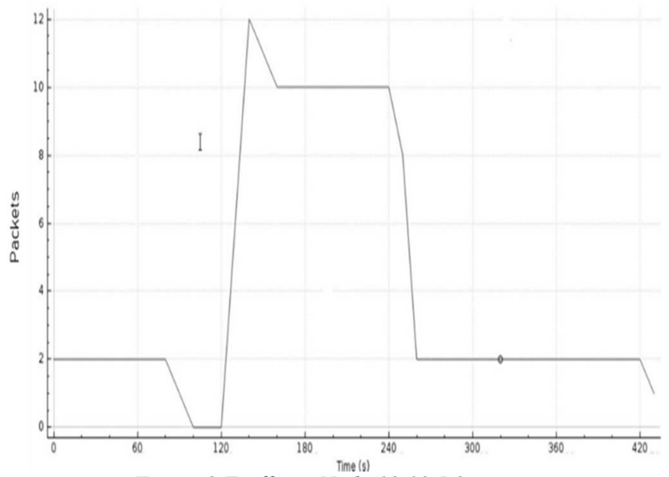


Figure 8 Traffic at Node 10.10.5.2

To test that a node can dynamically join the network, we enabled node 10.10.3.2 and node 10.10.4.2. First, the two nodes joined the network by reconnecting to their original peers, as depicted in Figure 9 (a). Then SH-P2P optimized its topology (see Figure 9 (b)).

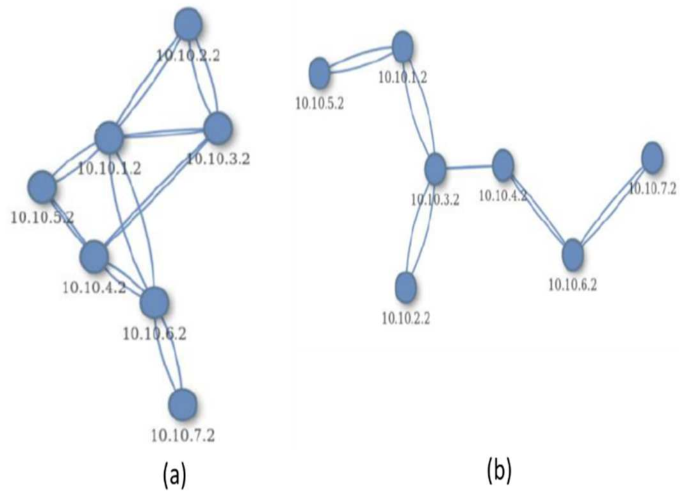


Figure 9 Nodes Joining the Network

In Figure 10 (a), an edge network node 10.10.7.2 is connected to the core network through node 10.10.6.2. By disabling node 10.10.6.2 and node 10.10.4.2, we were able to measure the time it takes for the edge network node to rejoin the core network by connecting to a different node. Figure 10 (b) shows the network's topology after node 10.10.6.2 and node 10.10.4.2 were disabled, and the edge network node reconnected through node10.10.3.2. In the test, node 10.10.1.2 generated traffic at constant intervals. As shown in Figure 11, node 10.10.7.2 was disconnected from the core network for 40 seconds.

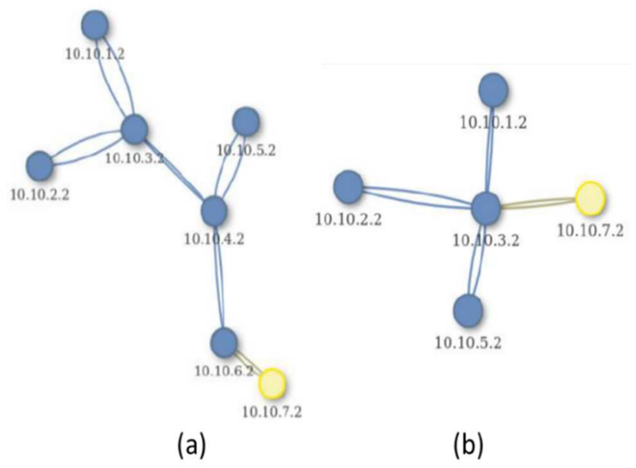


Figure 10 Edge Network Reconnect to the Core Network

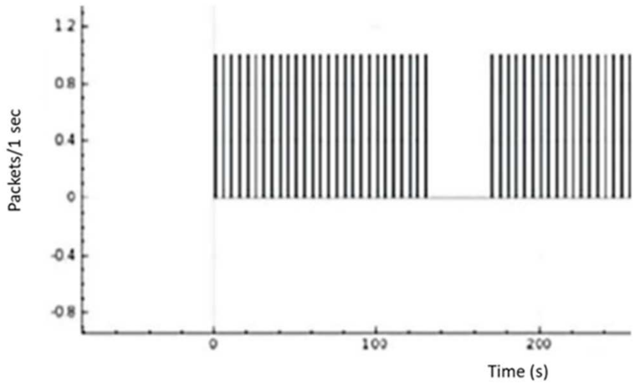


Figure 11 Traffic at Node 10.10.7.2

Figure 12 depicts bandwidth measurement overhead between two peers. On average, the LSS bandwidth computing process will consume around 1 percent of the link bandwidth per probe. On the other hand, LSS latency probes will consume an insignificant amount of bandwidth because they use few packets.

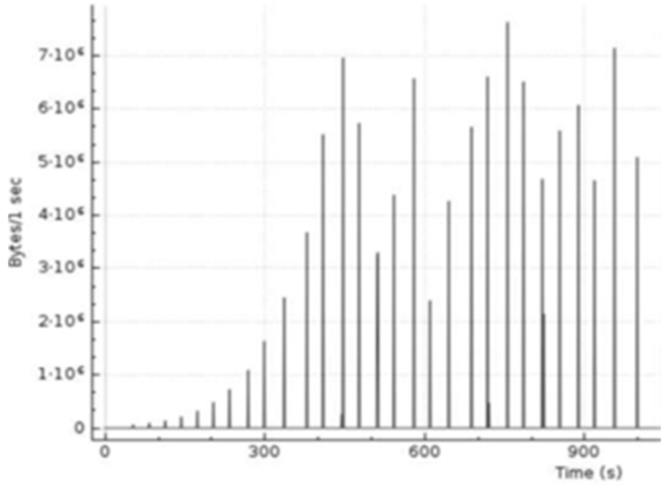


Figure 12 LSS Bandwidth Overhand Per Probe

VI. Conclusion

This paper described a new approach that enables P2P networks to connect peers operating in intermittently connected networks using a multicast network. A prototype was developed to demonstrate the new approach's capabilities and measure its performance. We have shown that edge nodes can maintain connections to the backbone peers when link failures occur. In addition, we have shown that the backbone peers can reconnect and form an optimal multicast network when partitions occur.

ACKNOWLEDGMENT

This material is based upon work supported by the US ARMY under contract number W56KGU-20-C-0071. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the US ARMY or the US Government.

References

[1] E. Androulaki et al., "Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains," Proceedings of the Thirteenth EuroSys Conference, April 2018.

[2] S. Banerjee, B. Bhattacharjee. "A Comparative Study of Application Layer Multicast Protocols," wmedia.grnet.gr/P2PBackground/acomparative-study-ofALM.pdf.

[3] M. Borokhovich, L. Schiff, and S. Schmid, "Provable Data Plane Connectivity with Local Fast Failover: Introducing Openflow Graph Algorithms," Proceedings of the Third Workshop on Hot Topics in Software Defined Networking, August 2014.

[4] B. Burnett, J. Al-Gharaibeh, B. Trent, R. Ramanujan, and H. Bloss, "HART: Hybrid Autonomous Router for Tactical Networks," Proceedings of the 2015 IEEE Military Communications Conference (MILCOM 2015), October 2015.

[5] Cyrin, https://cyrin.atcorp.com.

[6] G. Fox, "Peer-to-Peer Networks," Computing in Science and Engineering, June 2001.

[7] G. Greenspan, "MultiChain Private Blockchain," White Paper, https://www.multichain.com/download/MultiChain-White-Paper.pdf.

[8] M. Hearn, "Corda: A distributed ledger," White Paper, https://www.corda.net/content/corda-technical-whitepaper.pdf, November 2016.

[9] M. Hosseini, D.T. Ahmed, S. Shirmohammadi, N.D. Georganas. "A Survey of Application-Layer Multicast Protocols," IEEE Communications Surveys & Tutorials, Vol. 9(3), 2007, pp. 58-74.

[10] K. Katrinis, M. May. "Application-Layer Multicast, in Peer-to-Peer Systems and Applications," Lecture Notes in Computer Science 3485, Springer Verlag, 2005.

[11] B. Polaczyk, P. ChoBda, and A. Jajszczyk. "Peer-to-Peer Multicasting Inspired by Huffman Coding," Journal of Computer Networks and Communications, May 2013.

[12] "Quorum Whitepaper," https://github.com/jpmorganchase/quorum/blob/master/docs/Quorum%20Whitepaper%20v0.2.pdf.

[13] S. Sultana and A. Asaduzzaman, "A Minimum Spanning Tree Based Routing Protocol For Multi-hop And Multi-channel Cognitive Radio," 2018 Joint 7th International Conference on Informatics, Electronics & Vision (ICIEV) and 2018 2nd International Conference on Imaging, Vision & Pattern Recognition (icIVPR), June 2018.

[14] ZeroMQ, http://zeromq.org.