# Observation of Enhanced Network Performance in IoT Process Control and Data Sensing with RINA

Bhushana Samyuel Neelam, *Student Member, IEEE,* and Benjamin A Shimray, *Member, IEEE*

***Abstract*—Internet of things (IoT) is one of the leading technologies which spanned from the trivial consumer applications to time-critical industrial applications. The current research in IoT focuses mostly on network performance as it is experiencing bottlenecks in data communication. IoT communication preferred UDP due to the limitations of TCP hard-state handshaking procedures on throughput. Proposed work developed a prototype with IoT devices communicating on a new internet architecture i.e. recursive inter-networking architecture (RINA) which has eliminated hard-state handshaking procedures. The impact of RINA on the network performance in process control and data acquisition is observed in terms of latency variations, network jitter and throughput. The results were compared against the network performance when the proposed prototype was communicating on TCP/IP. A Comparative analysis was provided to identify the improved network performance in RINA. This prototype was implemented in closed network configurations like LAN and WLAN in RINA as well as TCP/IP.**

***Index Terms*—IoT, inter process communication, soft-state data transfer, latency, throughput, RINA.**

## I. Introduction

Internet of things (IoT) is one of the industry 4.0 technology that transformed the automation into most powerful and intelligent systems with its computing and communication facilities. IoT populated the existing Internet like never before as predicted by [1] due to its vast scope of advantages. In the midst of large-scale deployment of IoT, communication latency plays a significant role in determining network performance. Ongoing research for latency minimization in spite of the broad spectrum of solutions [2] prove its importance in IoT future. Numerous methods or protocols were employed to improve network performance in IoT communication [2], yet there exists a scope for an optimum solution. Some of the challenges mentioned in [2] include the usage of UDP instead of TCP to achieve low latency communication. As UDP is a connection-less protocol, there is no guarantee that the data packet is received by the destination or not. At the same time it restricted the usage of TCP as its handshaking procedures impacted the network performance in terms of throughput [3]. At this juncture, we would like to investigate the impact of a network model which eliminated hard-state data transfer mechanisms.

Manuscript received February 3, 2021; revised May 15, 2021. Date of publication June 9, 2021. Date of current version June 9, 2021.

Authors are with the National Institute of Technology Manipur, India (e-mails: samsneelam@nitmanipur.ac.in, benjaminshimray@gmail.com).

Digital Object Identifier (DOI): 10.24138/jcomss-2021-0027

Recursive inter-networking architecture (RINA) is one of the future network architectures that was proposed by John Day et.al [4] and found to have achieved an integrated solution in networking. RINA has extended local inter process communication (IPC) to distributed IPC [5] and transferred data through a secured environment with its soft-state based data transfer protocols [6]. It has replaced hard-state handshaking procedures with soft-state data transfer algorithms which were based on delta-t algorithm [7], [8].

The proposed work develops a client-server IoT Application on RINA and TCP/IP networks in LAN and WLAN configurations. The client node of IoT is equipped with sensors and actuators for data sensing and process control. Network performance in terms of latency, network jitter and throughput is measured for control and data sensing operations in both network models. A comparative study, proved the advantages of soft-state data transfer protocols of RINA by improving network performance when compared to TCP/IP. The motivation for this work is to investigate the impact of soft-state design-based data transfer protocols of RINA over hard-state data transfer mechanisms of TCP/IP [3], [8]. The proposed work contributes the following state-of-art works in LAN and WLAN using RINA in IoT control and data sensing communication.

1) We explain precisely about the data transfer mechanisms of RINA and TCP/IP (section II). Surprisingly this comparison has not been done before.
2) We have developed a prototype of RINA based IoT control and data acquisition (section III) which is novel application based on RINA.
3) We have compared network performance of RINA and TCP/IP in IoT process control and data acquisition which has not been provided so far.

The novelty of this article lies in extracting the advantages of clean-state architectures like RINA in IoT control and data sensing [6] with reliable communication flows.

The text in this article is organized as follows. Section II compares architecture and data transfer mechanisms of TCP/IP and RINA. Design and deployment of proposed prototype is described in section III. Section IV provides observations and discussions of network performance in both the networks.

## II. TCP/IP vs RINA

TCP/IP architecture is based on layer-wise design. Each layer is meant for a specific purpose. As TCP/IP was originated from ARPANET which is designed for limited tasks,

TABLE I
RINA VS TCP/IP

| Feature | RINA | TCP/IP |
|---|---|---|
| Addresses | Private [9] | Public |
| Ports | Local, internal and detached from connection management [8]. | Public and overloaded ports as Cep-Ids |
| Protocols | Generic with programmable policies [9] | Specific |
| Address length | Variable thus eliminating length attacks [10] | Fixed and susceptible to cyber attacks |
| Modern networking demands | Architectural design of recursion and programmability of DIF accommodate dynamic networking and multi-homing on mobility [11] | Add-on protocols |
| Security | In built algorithms [12] | Add-on protocols |

it had to equip so many other protocols to counter the ever-increasing network demands. Thus, it made TCP/IP architecture more complex and patchy. Several add-on protocols were invented depending on the type of network application. But there exists a scope for the design and deployment of new add-on protocols to TCP/IP suite as its network scope is ever-increasing.

RINA is a clean state architecture that is based on the concept of inter-process communication (IPC) which is an operating system's feature. It contains a single layer called distributed IPC facility (DIF) and repeats depending on the network scope [4], [13]. DIF undertakes the unified functionality with mechanisms for data transfer, data transfer control, routing, and security which will be configured depending on the scope. The instance of DIF on the communicating device is known as the IPC process (IPCP) [5], [8], [14]. RINA network model is a recursive architecture, where its fundamental component DIF will be repeated with reference to the network scope. Each IPCP is provided with a rank based on its scope. The functionality of IPCP remains the same irrespective of its rank but can be programmed to suit the scope of DIF, which makes RINA as programmable network. The functionality of DIF can be customized with a feature called *separation mechanism from protocol to policy* [9]. This has presented a concept of generic protocols in RINA with a wide variety of policies. These policies are to be programmed to suit the network requirements thus eliminating separate protocols for each functionality. This claims RINA as a reliable network architecture with few protocols and various policies [9], [15].

IPCP with the lowest rank is called shim IPCP and it is responsible for wrapping up the legacy Ethernet to transfer data. As mentioned in [5], [15], IPCP drives data transfer and control mechanisms in RINA. The two generic protocols that controls IPCP functionality are error flow control protocol (EFCP) and common distributed application protocol (CDAP). EFCP is meant for data transfer and to control the flow characteristics. CDAP is meant for layer management functionalities like coordinating the services of IPCP across the layers of DIF [15], [16]. Table.1. summarizes the architectural differences between TCP/IP and RINA.

### A. Data Transfer Mechanism

*1) TCP/IP:* TCP/IP is a five-layer protocol suite where each layer performs a specific application. Data transfer in TCP/IP is performed by the transport layer. It is responsible for data transfer, flow control, multiplexing, and reliability of end-to-end communication. Transport layer is provided with two protocols i.e. TCP and UDP for connection-oriented or reliable communication and connection-less communication respectively. TCP uses three-way handshake in establishing a reliable connection between any two applications. This exchange of information includes sequence TCP segments like SYN, SYN/ACK, ACK. This three-way handshake is based on a hybrid state approach as understood by [17], [18]. To eliminate data duplication, TCP adopted a five-pocket protocol which provides a timer with 2xMPL to wait before closing the connection [19], [20]. So for every event of data transfer, it takes SYN, SYN/ACK, DATA+ACK, ACK and waits for a period of 2xMPL to close the connection.

*2) RINA:* In RINA, EFCP regulates data transfer and data transfer control modules of IPCP. EFCP is divided into two minor protocols i.e. data transfer protocol (DTP) and data transfer control protocol (DTCP) which are based on soft-state design [9]. Soft-state design is derived from Watson's Delta-t protocol to achieve reliable connection like TCP [7]. According to Watson [7], bounding of three timers is necessary and appropriate to establish a reliable connection. It simplified the network implementation by eliminating explicit hard-state handshaking of TCP.

Delta-t mechanism of EFCP is customized with the help of *separation mechanism from policy* and the *abstract and concrete syntaxes* [21] to make it a universal framework for data transfer across DIF. EFCP decoupled all mechanisms from one another which is not present in other protocols like TCP. For example, flow control and re-transmission control are inseparable in TCP, but they are detached in RINA and will be performed by different policies. RINA also decoupled port allocation from data synchronization and avoided overloading of the semantics of connection end-point-id. This is not provided in TCP and thus it overloads connection synchronization with security mechanisms [6]. Hence, RINA provides security in data transfer by avoiding overloading of connection end-point-ids. Table.2. provides the summary of data flow procedures of EFCP in communicating across the DIF.

## III. Experimental Setup

The proposed model consists of two IoT devices as shown in Figure.1. One IoT device works like server control and other as a client. The client IoT device is interfaced to process control actuators like LED, relay and to a temperature and humidity

TABLE II
DATA FLOW PROCEDURES IN RINA [6], [19]

| Procedure | Functionality |
|---|---|
| Creation of communication flow | Source application requests a flow to the destination application through an IPCP. An EFCP instance will be created between the IPCPs' of both applications, after its names and Quality of Service(QoS) are matched |
| Data transfer | As handshaking is not existing in RINA, a flag called data run flag (DRF) in the EFCP header is set to *true* to inform that it is the beginning of new data stream. This DRF initiates the synchronizing timers. |
| Connection state synchronization | This will be managed by four timers i.e. inactivity timers of sender and receiver, A-timer and the re-transmission timer which are necessary to manage a reliable connection [7]. |
| Duplication removal | Sequence numbers will be refreshed as soon as the inactivity timers are fired and thus avoids any kind of duplication. |
| Flow and congestion control | Flow control is meant for protecting buffers on the receiver's side and is divided into two types i.e. sliding window and rate limiting. EFCP provides explicit notice for congestion control with the help of a bit called explicit congestion notification (ECN) in its headers. |
| Dynamic configuration of connection parameters | EFCP connection parameters can be changed in runtime of the communication flow without loosing the required QoS. |

sensor. Both IoT devices are connected using a router as shown in the Figure 1. The client IoT of the proposed model is

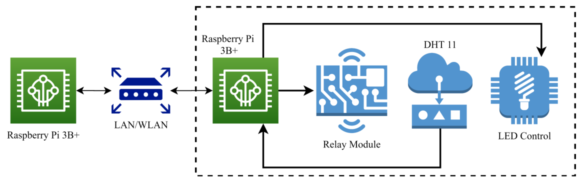


Fig. 1. Proposed IoT model

developed with the following components. Figure 2 shows the hardware implementation of the client IoT. i.

1) IoT device - Raspberry Pi 3B+
2) Actuators - LED, 4-module relay
3) Sensor - DHT11 for temperature and humidity

The proposed application is developed in a client-server mode in both the networks i.e. using IRATI [15] socket API in RINA and using Posix V socket API in TCP/IP. Server control provides a GUI to control the actuators and sensors. GUI also provides display of the data from sensors. It also provides the measurement of latency in process control and data sensing operations of remote client IoT. Wireshark is used to capture the packet flow and to verify the soft-state design-based data transfer of RINA. The communication strategy of the proposed network model is based on a connection-oriented flow. In TCP/IP, it is implemented with TCP as it is connection-oriented communication. In RINA, it is programmed in the DIF template as *reliableflow* which provides connection-oriented RINA communication as discussed in II-A2.

## IV. RESULTS

As RINA is a clean state architecture, It has many inbuilt characteristics that are aimed at providing better service than TCP/IP. Table III compares some of the key characteristics of TCP/IP and RINA. This article extracts the advantage of

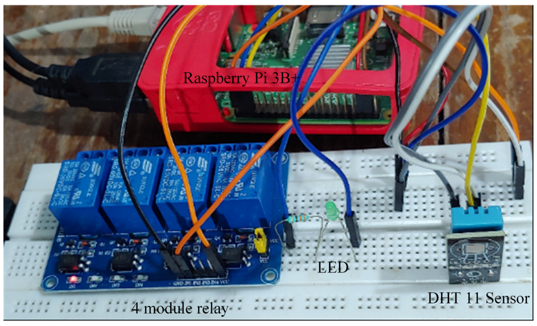


Fig. 2. Hardware implementation of client

TABLE III
COMPARISON OF KEY CHARACTERISTICS

| TCP/IP | RINA |
|---|---|
| Public addresses and known ports | Internal addresses and Private ports [8], [10], [22] |
| Hybrid state data transfer | Soft-state data transfer [6], [8], [19] |
| Add-on security | Inbuilt and integrated security [9], [10] |
| End-to-end congestion control | Congestion control at the source level [23] |
| Interface naming | Node naming [11] |

soft-state data transfer in terms of network performance and verifies its handshaking procedures.

The application is run for certain time in each network and the network performance is observed for control and data sensing operations. Latency, network jitter and throughput are considered as indices for the network performance.

### A. Soft-sate Data Transfer

Network flows are captured using network flow scanner tool *Wireshark*. Figure 3 shows TCP connection establishment and initiation of data transfer across the server and client IoT. Hard-state data transfer mechanisms of TCP connection establishment with handshaking procedures i.e. SYN, SYN/ACK, ACK and data transfer can be seen from Figure 3. Figure 4.

shows RINA flows between client and server IoT. It can be observed that there are no explicit handshaking exchanges but simply data transfer based on timers [8]. Thus it confirms the soft-state data transfer mechanisms of RINA.

### *B. Network Performance*

Latency is one of the performance index that impacts network performance. [3] et.al claimed that hard-state handshaking procedures do impact throughput and thereby latency too. Hence we have adopted latency and throughput as network performance indices to investigate the impact of RINA which eliminated handshaking procedures in connection establishment. We have considered network jitter also as another performance index since it provides the consistency of data communication.Network jitter is the variability of latency in data communication. Lesser the network jitter, greater the consistency of the network. IETF definition [24] is adopted to calculate the jitter.

*1) Latency - LAN:* Results demonstrate that data transfer can take place in an effective manner with soft-state design based data transfer protocols. Latency is calculated in the application for both process control and data sensing in LAN and WLAN configurations. It is analyzed in terms of latency variation against number of operations and arithmetic mean of latency. Figure 5 shows comparison of latency between RINA and TCP/IP in LAN configuration. Figure 5(a) shows latency variation against number of control operations. It is evident that latency in RINA is reduced drastically in most of the occasions.

Figure 5(b) shows latency variation against number of data sensing operations for both RINA and TCP/IP. It can be observed that latency has improved slightly in RINA in comparison with TCP/IP except for few operations. It is also observed that latency is sporadic in both networks but less in RINA when compared to TCP/IP.

Arithmetic mean of latency is calculated and shown in Figure 5(c). The percentage of latency minimization in process control and data sensing using RINA stands at 46% and 1% respectively. Figure 5(d) shows comparison of average network jitter in LAN configuration. It is evident from the figures that RINA has minimized jitter substantially in process control but it is marginal in data sensing. It can be observed from the figure 5 that latency in process control is improved distinctively in all indices i.e. latency variation, arithmetic mean and network jitter. where as the improvement in data sensing is marginal in data sensing.

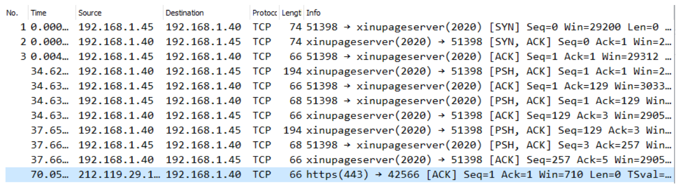

```
No. Time      Source         Destination   Protocc Lengt Info
  1 0.000…  192.168.1.45   192.168.1.40   TCP     74 51398 → xinupageserver(2020) [SYN] Seq=0 Win=29200 Len=0 …
  2 0.000…  192.168.1.40   192.168.1.45   TCP     74 xinupageserver(2020) → 51398 [SYN, ACK] Seq=0 Ack=1 Win=2…
  3 0.004…  192.168.1.45   192.168.1.40   TCP     66 51398 → xinupageserver(2020) [ACK] Seq=1 Ack=1 Win=29312 …
    34.62…  192.168.1.40   192.168.1.45   TCP    194 xinupageserver(2020) → 51398 [PSH, ACK] Seq=1 Ack=1 Win=2…
    34.63…  192.168.1.45   192.168.1.40   TCP     66 51398 → xinupageserver(2020) [ACK] Seq=1 Ack=129 Win=3033…
    34.63…  192.168.1.45   192.168.1.40   TCP     68 51398 → xinupageserver(2020) [PSH, ACK] Seq=1 Ack=129 Win…
    34.63…  192.168.1.40   192.168.1.45   TCP     66 xinupageserver(2020) → 51398 [ACK] Seq=129 Ack=3 Win=2905…
    37.65…  192.168.1.40   192.168.1.45   TCP    194 xinupageserver(2020) → 51398 [PSH, ACK] Seq=129 Ack=3 Win…
    37.66…  192.168.1.45   192.168.1.40   TCP     68 51398 → xinupageserver(2020) [PSH, ACK] Seq=3 Ack=257 Win…
    37.66…  192.168.1.40   192.168.1.45   TCP     66 xinupageserver(2020) → 51398 [ACK] Seq=257 Ack=5 Win=2905…
    70.05…  212.119.29.1…  192.168.1.40   TCP     66 https(443) → 42566 [ACK] Seq=1 Ack=1 Win=710 Len=0 TSval=…
```

Fig. 3. Hybrid-state TCP handshake

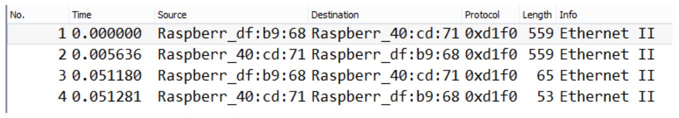

```
No.  Time      Source             Destination        Protocol Length Info
   1 0.000000  Raspberr_df:b9:68  Raspberr_40:cd:71  0xd1f0   559 Ethernet II
   2 0.005636  Raspberr_40:cd:71  Raspberr_df:b9:68  0xd1f0   559 Ethernet II
   3 0.051180  Raspberr_df:b9:68  Raspberr_40:cd:71  0xd1f0    65 Ethernet II
   4 0.051281  Raspberr_40:cd:71  Raspberr_df:b9:68  0xd1f0    53 Ethernet II
```

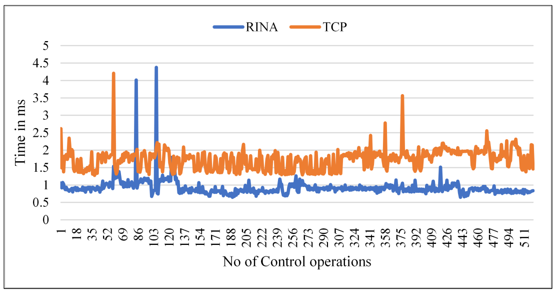
Fig. 4. Soft-state handshake in RINA

*2) Latency - WLAN:* The latency comparison between TCP/IP and RINA in WLAN configuration is shown Figure 6. Figure 6(a) shows latency variation in control against the number of operations. In this configuration, RINA and TCP/IP provides almost same latency for most of the operations. It can be observed from the figure that there is drastic increase of latency in TCP/IP for some controlling operations, but whereas it is not visible in RINA.

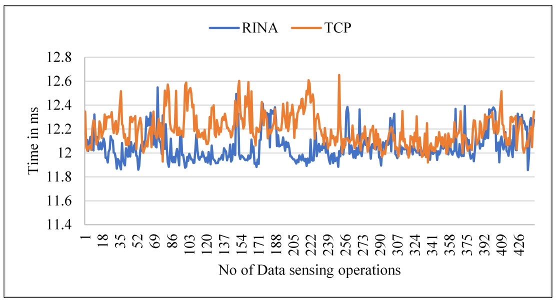
Figure 6(b) shows latency variation in data sensing against the number of operations. As per the Figure, it is evident that TCP/IP and RINA are having almost uniform latency. But there are sudden increases of latency in TCP which are not visible in RINA operations. Here, arithmetic mean of latency plays significant role in confirming improved network performance in RINA. Figure 6(c) shows arithmetic mean of latency in both control and data sensing operations. It can be observed that mean latency of control operations has reduced to 6.638 ms in RINA from 21.118 ms of TCP. Likewise mean latency of data sensing operations has reduced slightly to 17.802 ms from 18.308 ms of TCP. It is evident from the figure that RINA has edge over TCP with a reduction of 68% and 2% in control and data sensing operations respectively. Figure 6(d) shows comparison of average jitter in WLAN configuration. Here also, it can be observed that latency is more stable in process control than data sensing. It is clear that latency in process control has improved a lot in RINA when compared to data sensing in all performance indices. But latency in data control is having very nominal improvement with respect to all indices of latency.

*3) Throughput:* Window based data transfer is adopted in both the networks i.e. RINA and TCP/IP. The default window size of the IoT device that is considered is 64K bytes for both TCP/IP and RINA. Throughput is calculated with the help of [25] and average throughput in process control and data sensing operations in LAN and WLAN configurations is shown in Figure 8. It can be observed from the figure that throughput in RINA network is higher than that of TCP in both LAN and WLAN. It is more evident in process control that throughput has almost doubled in RINA when compared to TCP. Whereas it shows slight improvement of 1-3% in data sensing operations. This proves that soft-state data transfer protocols of RINA can improve throughput by eliminating handshaking procedures. It is evident from the results that TCP handshaking procedures can impact throughput as claimed by hewage et.al [3].

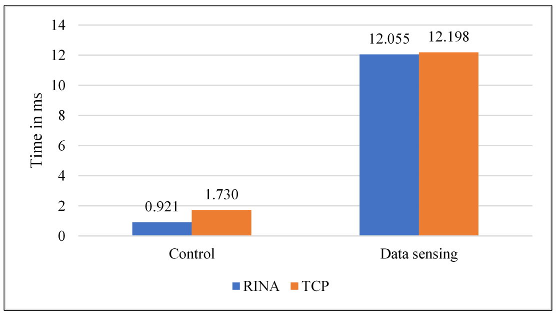
It can be concluded from the above results that RINA provides improved latency, network jitter and throughput with its soft-state data transfer protocols in LAN and WLAN. The resiliency by design of RINA provides inbuilt security as verified by [8], [14], [22] to secure data communication. It is convinced from the above results that RINA has edge over TCP/IP in improving overall network performance with inbuilt security with its clean state design.



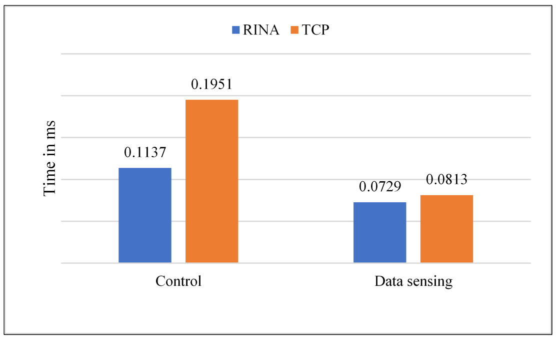
(a) Latency vs No of control operations



(b) Latency vs No of data sensing operations

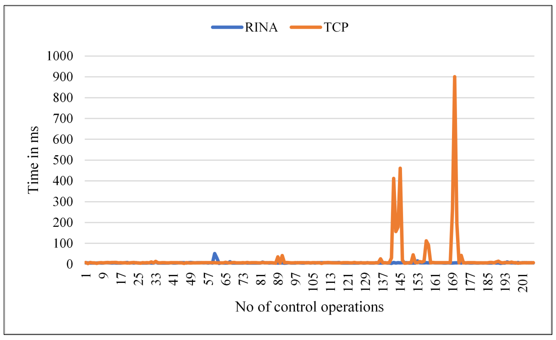


(c) arithmetic mean

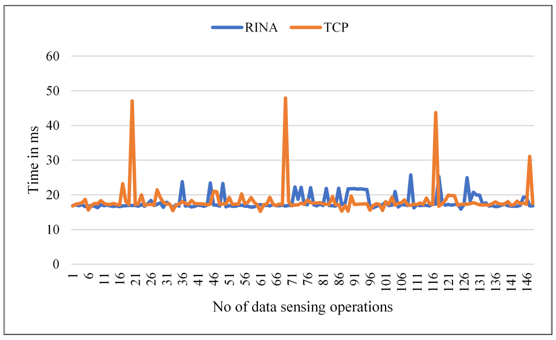


(d) Network jitter

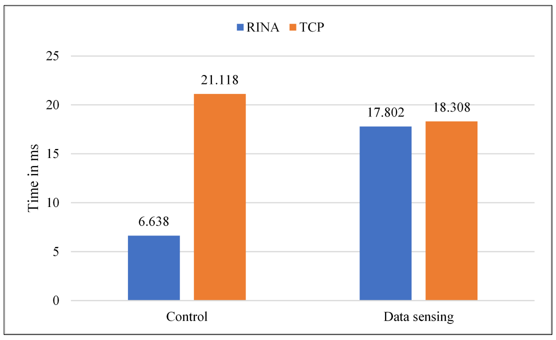
Fig. 5. Latency in LAN



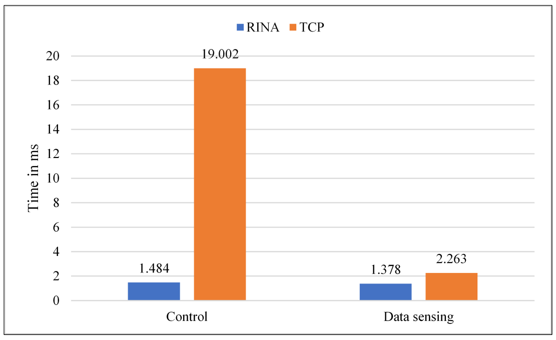
(a) Latency vs No of control operations



(b) Latency vs No of data sensing operations



(c) arithmetic mean



(d) Network jitter

Fig. 6. Latency in WLAN

## V. Conclusion

Proposed work investigated the impact of RINA network on the network performance of IoT process control and data acquisition. We have explored the differences between data transfer mechanisms of RINA and TCP/IP. We have verified the handshaking procedures of establishing communication flow in TCP/IP and RINA network models using Wireshark. We have presented RINA prototype of IoT control and data

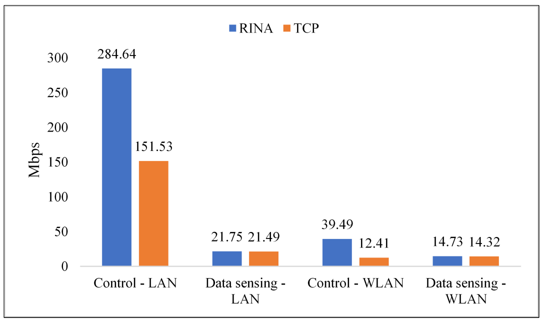


Fig. 7. Comparison of average throughput

acquisition. The developed prototype is subjected to communicate in RINA and TCP/IP networks to assess the network performance in terms of latency, jitter and throughput. The comparative analysis of the experimental results have confirmed that RINA has improved network performance when compared to TCP/IP. RINA has improved network performance substantially with minimized latency, reduced network jitter and increased throughput in LAN and WLAN. It is significantly improved in process control where as marginal in data acquisition. The contribution of this article is to verify the impact of RINA network on network performance in the closed network configurations which can motivate to experiment it in the open systems. Our current research is focused on latency variations in the presence of security algorithms in different network configurations. The limitations of RINA is its network stack which is limited to Linux-based systems.

## References

[1] J. Bradley, J. Barbier, and D. Handler, "Embracing the internet of everything to capture your share of $14.4 trillion: More relevant valuable connections will improve innovation productivity efficiency & customer experience," *White Paper, Cisco Systems Inc*, 2013.

[2] N. Srinidhi, S. D. Kumar, and K. Venugopal, "Network optimizations in the internet of things: A review," *Engineering Science and Technology, an International Journal*, vol. 22, no. 1, pp. 1–21, 2019.

[3] K. Hewage, S. Duquennoy, V. Iyer, and T. Voigt, "Enabling tcp in mobile cyber-physical systems," in *2015 IEEE 12th International Conference on Mobile Ad Hoc and Sensor Systems*. IEEE, 2015, pp. 289–297.

[4] D. John, *Patterns in network architecture*. Pearson Education India, 2007.

[5] E. Grasa, L. Bergesio, M. Tarzan, E. Trouva, B. Gaston, F. Salvestrini, V. Maffione, G. Carrozzo, D. Staessens, S. Vrijders *et al.*, "Recursive internetwork architecture, investigating rina as an alternative to tcp/ip (irati)," 2017.

[6] M. Tarzan, L. Bergesio, and E. Grasa, "Error and flow control protocol (efcp) design and implementation: A data transfer protocol for the recursive internetwork architecture," in *2019 22nd Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN)*. IEEE, 2019, pp. 66–71.

[7] J. G. Fletcher and R. W. Watson, "Mechanisms for a reliable timer-based protocol," *Computer Networks (1976)*, vol. 2, no. 4-5, pp. 271–290, 1978.

[8] G. Boddapati, J. Day, I. Matta, and L. Chitkushev, "Assessing the security of a clean-slate internet architecture," in *2012 20th IEEE International Conference on Network Protocols (ICNP)*. IEEE, 2012, pp. 1–6.

[9] E. Grasa, O. Rysavy, O. Lichtner, H. Asgari, J. Day, and L. Chitkushev, "From protecting protocols to layers: designing, implementing and experimenting with security policies in rina," in *2016 IEEE International Conference on Communications (ICC)*. IEEE, 2016, pp. 1–7.

[10] T. Ramezanifarkhani and P. Teymoori, "Securing the internet of things with recursive internetwork architecture (rina)," in *2018 International Conference on Computing, Networking and Communications (ICNC)*. IEEE, 2018, pp. 188–194.

[11] V. Ishakian, J. Akinwumi, F. Esposito, and I. Matta, "On supporting mobility and multihoming in recursive internet architectures," *Computer Communications*, vol. 35, no. 13, pp. 1561–1573, 2012.

[12] V. Maffione, F. Salvestrini, E. Grasa, L. Bergesio, and M. Tarzan, "A software development kit to exploit rina programmability," in *2016 IEEE International Conference on Communications (ICC)*. IEEE, 2016, pp. 1–7.

[13] J. Day, I. Matta, and K. Mattar, "Networking is ipc: a guiding principle to a better internet," in *Proceedings of the 2008 ACM CoNEXT Conference*, 2008, pp. 1–6.

[14] T. Ramezanifarkhani and P. Teymoori, "Securing the Internet of Things with Recursive InterNetwork Architecture (RINA)," *2018 International Conference on Computing, Networking and Communications, ICNC 2018*, no. November 2017, pp. 188–194, 2018.

[15] S. Vrijders, D. Staessens, D. Colle, F. Salvestrini, E. Grasa, M. Tarzan, and L. Bergesio, "Prototyping the recursive internet architecture: the irati project approach," *IEEE Network*, vol. 28, no. 2, pp. 20–25, 2014.

[16] S. Vrijders, V. Maffione, D. Staessens, F. Salvestrini, M. Biancani, E. Grasa, D. Colle, M. Pickavet, J. Barron, J. Day, and L. Chitkushev, "Reducing the complexity of virtual machine networking," *IEEE Communications Magazine*, vol. 54, no. 4, pp. 152–158, 2016.

[17] R. Braden, "Rfc1644: T/tcp–tcp extensions for transactions functional specification," 1994.

[18] G. Gursun, I. Matta, and K. Mattar, "On the performance and robustness of managing reliable transport connections," *CS Department, Boston University, Tech. Rep. BUCS-TR-2009-014*, 2009.

[19] ——, "Revisiting a soft-state approach to managing reliable transport connections," in *Proceedings of the Eighth International Workshop on Protocols for Future, Large-Scale and Diverse Network Transports (PFLDNeT)*, 2010.

[20] J. Postel, "Rfc 793: Transmission control protocol, september 1981," 2003.

[21] X. ITU-T, "680: Itu-t recommendation x. 680 (1997)— iso/iec 8824-1: 1998," in *Information Technology-Abstract Syntax Notation One (ASN. 1): Specification of Basic Notation*.

[22] N. B. Samyuel and B. A. Shimray, "Securing iot device communication against network flow attacks with recursive internetworking architecture (rina)," *ICT Express*, 2020.

[23] P. Teymoori, M. Welzl, S. Gjessing, E. Grasa, R. Riggio, K. Rausch, and D. Siracusa, "Congestion control in the recursive internetworking architecture (rina)," in *2016 IEEE International Conference on Communications (ICC)*. IEEE, 2016, pp. 1–7.

[24] C. Demichelis and P. Chimento, "Ip packet delay variation metric for ip performance metrics (ippm)," RFC 3393, November, Tech. Rep., 2002.

[25] B. Constantine, G. Forget, R. Geib, and R. Schrage, "Framework for tcp throughput testing," *Internet Engineering Task Force*, vol. 2011, pp. 1–27, 2011.

**Bhushana Samyuel Neelam** received M.Tech in Electrical Power Engineering from JNTU College of Engineering, Hyderabad, India. Then worked in design and development of custom network stack for mission critical applications before shifting to teaching profession. Currently He is pursuing Ph.D in the department of Electrical Engineering in National Institute of Technology, Manipur, India. He is also student member of IEEE. His research areas include RINA in real time applications, industrial utilities and Cyber resiliency.

**Benjamin A Shimray** received his Ph.D. and M.Tech from the National Institute of Technology Manipur (2018), Imphal and The National Institute of Engineering, Mysore (2010) respectively. Since 2011 he has been working as faculty member of department of Electrical Engineering, National Institute of Technology Manipur. He is also a member of various professional organization such as IEEE, IEI and IAENG. His area of interest is application of Soft Computing techniques to renewable energy planning and management, control system applications, and in recent years in cyber vulnerabilities and system resilience.