2021 2nd International Conference for Emerging Technology (INCET)
Belgaum, India. May 21-23, 2021

Performance Assessment of RINA Adoption tool shim-tcp-udp in IoT-SCADA Application

Bhushana Samyuel Neelam
Department of Electrical Engineering
National Institute of Technology, Manipur
Imphal, India
samsneelam@gmail.com

Bejmain A Shimray
Department of Electrical Engineering
National Institute of Technology, Manipur
Imphal, India
benjaminshimray@gmail.com

2021 2nd International Conference for Emerging Technology (INCET) | 978-1-7281-7029-9/20/$31.00 ©2021 IEEE | DOI: 10.1109/INCET51464.2021.9456300

***Abstract*—The recent networking research trends express the need for more reliable protocols or networking architectures. The main requirements of modern networking have included security, heterogeneousness, and programmable QoS. But the architectural limitations of the existing network architecture caused it to depend on various add-on protocols to meet the modern networking demands. Many network architectures have evolved, but a clean state network architecture called recursive internetworking architecture (RINA) seems to be a promising alternative. Its architectural design has enabled inbuilt security, compatibility to heterogeneous networking, and QoS programmability. The proposed work investigates the network performance in terms of response times when a RINA-based IoT SCADA application ported over the Internet with the help of one of its adoption tools; *shim-tcp-udp*. The results confirmed that *shim-tcp-udp* executes SCADA control and data acquisition effectively but with added communication overheads when compared to IP networks.**

***Index Terms*—IoT, SCADA, network architectures, Recursive Internetworking Architecture (RINA), RINA over IP.**

I. Introduction

The increasing dependency of consumer electronics on the Internet of Things(IoT) unfolded several limitations of the existing Internet architecture. These limitations have included cybersecurity, dynamic configuration of Quality of service (QoS), congestion control, mobility, and multihoming. Many network paradigms evolved to counter the existing networking limitations but yet there exists a scope for the integrated solution. Some of the clean-state architectures were also designed and deployed, but most of these architectures adopted layer-wise protocols like TCP/IP [1], thus causing limitations for the integrated networking solution.

Recursive internetworking architecture (RINA) is a clean state network architecture that originated from the basics of networking. According to John Day who proposed RINA; *"Networking is inter-process communication (IPC) and IPC alone"* [2]. RINA focused on providing common API across all layers of networking by providing a generic layer instead of specific layers. RINA extended the local IPC process to distributed IPC process, which formed a layer called distributed IPC facility (DIF). This DIF will be repeated depending on the network scope. DIF consists of mechanisms required for transferring data across its participants [3]. The participant of DIF is nothing but its local instance and is called the IPC process (IPCP). IPCP acts as an interface between the applications intended to transfer data between them. RINA adopted another principle of operating systems i.e. *separation of mechanism and policy* which yielded a programmable DIF that can be configured with predefined policies. The predefined policies provided the configuration of security, QoS, and routing algorithms. The unique features of recursion, private addressing, and the late binding of the interface to the address provided the inbuilt capability for mobility and multihoming [4]. The clean state design of the RINA tried to address almost all the existing network limitations. It also provided various network adoption tools to make use of existing network infrastructure.

In the proposed work, we develop a RINA-based SCADA application and communicate over the internet to assess the performance of one of its adaptation tools. We implement the RINA-based IoT SCADA application communicating over the internet with the help of *shim-tcp-udp*. The results confirm the successful implementation of *shim-tcp-udp* in IoT-SCADA application in RINA network ported over the existing infrastructure.

To our knowledge, this is the first of its kind in executing RINA-based IoT-SCADA applications over the internet with the help of its adoption tools - *shim-tcp-udp*. It explores the feasibility of implementing RINA for IoT-enabled SCADA systems and examines the performance of its adoption tool-*shim-tcp-udp*.

The remaining text of this article is organized as follows. Section II describes the significance of RINA in the present context. Section III presents the proposed model of IoT-SCADA and its network model. Section IV discusses the results and section V concludes the article.

II. Why RINA

RINA projected a network architecture that derived from the basics. It depends on the principle of shared data transfer through Interprocess communication (IPC). RINA generalized local IPC and extended it to distributed IPC [5]. Applications have to make use of this distributed IPC to share data through a shared entity which is called distributed IPC facility (DIF). DIF forms a generic layer in RINA and provides all the

TABLE I
DIF VS LAYERS OF TCP/IP

Functionality	DIF Module/Protocol	TCP/IP Layer / protocol
Data transfer	Data transfer /DTP	Transport layer / TCP or UDP
Retransmission & flow control	Data transfer control / DTCP	Transport layer / TCP
routing	Layer management / CDAP	Network layer / ARP & RARP
congestion control	data transfer control / DTCP	Transport layer / TCP, SCTP, DCCP and RTP
security	data transfer, layer management / DTP and CDAP	Layer wise security is provided, e.g. HTTPS for application layer, SSL/TLS for transport layer, IPSec for IP Layer

mechanisms needed for transferring data across the applications/hosts. The architecture of RINA is based recursion, where DIF repeats based on the network scope [3].

The functionality of DIF adopted an integrated approach by combining data transfer, data transfer control, and layer management functionalities altogether. This kind of approach is not available on the existing internet, because it performs data transfer with the help of specific layers/protocols. But the generic layer of RINA produced generic protocols i.e. Error flow control protocol (EFCP) and common distributed application protocol (CDAP). EFCP is again divided into two minor protocols - data transfer protocol (DTP) and data transfer control protocol (DTCP). The purpose of EFCP is to take care of the data transfer and data transfer control mechanisms. The functionality of CDAP focuses on routing and layer management mechanisms. Together, EFCP and CDAP govern the total functionality of DIF [3], [6]. The following Table I reflects the comparison of functionalities of DIF and the layers of TCP/IP.

The following features distinguish RINA from the existing network architecture to provide better internet architecture.

A. Internal addressing

RINA avoided the public addressing strategy which is based on naming the interface instead of the host. It adapted an internal addressing schema whose scope is limited to the concerned DIF. Each DIF in the RINA network stack will have its own address. Due to its generic and recursive layers, the functionality of the application, node, or point of attachment (PoA) in RINA became relative. This made the address of each RINA component depend upon its relative position in the network stack and its scope. This generated unique addresses to each DIF or IPCP and made them private to its scope. At outset, RINA applications use their names as addresses in communicating with each other, but the underlying addressing scheme is private and it is the responsibility of DIF to resolve the concerned DIF or IPCP of the destination application [7].

B. Port detachment

RINA detached the port number from the address. In RINA, the port is internal and restricted to act as an interface between the application and the corresponding IPCP. The advantages of port detachment include

- resiliency against port scanning attacks as it eliminated public ports.
- security against connection opening or data transfer attacks as it avoided the overloading of ports as connection-end-point-Ids.
- inbuilt ability for mobility and multihoming.

The separation of port allocation from the connection management enabled RINA to perform its data transfer in soft-state approach [3], [7], [8].

C. Programmability

RINA implemented the principle of *separation of mechanism from the policy* in its layer i.e. DIF. Each functionality of DIF / IPCP is designed to work with this principle thus enabled DIF to be a programmable entity. Separation of mechanism from the policy enabled DIF to work with various networking configurations to satisfy the requirements of the customized networks. This eliminated the need for special cases or special protocols, unlike the existing network architecture. The programmability of DIF enabled

- customization of security of the service datagram unit (SDU) in terms of authentication, access control, and data confidentiality.
- customization of QoS parameters to achieve congestion control and improved network performance.
- customization of routing methods to increase the efficiency of the network.
- customization of data transfer control mechanisms to reduce latency and packet loss ratio.

This feature differentiated RINA from the other emerging network architectures and provided complete programmability of the network which has been the need of the hour in the presence of ubiquitous networking [3], [9]–[12].

D. Inbuilt security

The clean state design of RINA enabled its architecture to be secure by its design. Its inbuilt security begins with its basic component i.e. DIF which forms a closed environment where it allows only its participants to transfer the data [5]. Every IPCP that wants to become part of any DIF, should enroll itself in the corresponding DIF in the first place. Then only it can send or receive data. This is implemented with the help of explicit enrollment where DIF allows the requested IPCP, only when their credentials matched [9]. The uniqueness of RINA security is its unified security policies which consist of all the security algorithms as a set. The network designer can tune the policies according to their requirement. The predefined policies of SDU protection provided

- heavy authentication algorithms like SSH2 and TLS
- encryption algorithms like AES-128, AES-256

- hashing algorithms like SHA1, SHA256, and MD5
- compression algorithms like deflate
- error check algorithms

This type of security is not available with the existing network architecture where each protocol addresses one type of security algorithm [3], [13]. The private addressing and local ports enhanced RINA security as there are no public addresses available. This internal addressing scheme eliminated most of the network flow attacks like port scanning, connection opening and data transfer attacks [8].

E. Congestion control

The recursive design of RINA divided the network into smaller scope thus broke long control loops into smaller loops and enabled granular control over communication flow. Data transfer mechanisms of RINA allow configuration of its flow characteristics using its QoS parameters. RINA provided various predefined policies to achieve fine control over network traffic congestion problems [10], [12].

F. Mobility and multihoming

The elimination of interface addressing made it easy for RINA to provide mobility and multihoming without any additional mechanisms. Elimination of public addressing resulted in only one address to the host/node irrespective of the interfaces which is contrary to the existing internet. The mapping of destination application is the responsibility of DIF, which uses its routing information base (RIB) to map the destination. Once the node is mapped, it finds the corresponding interface. This late binding enabled effortless mobility and multihoming in RINA and resolved the needs of present-day heterogeneous networking [4].

G. Adaptability

RINA developed its own network stack as a part of IRATI [14] and provided various tools to make use of the existing internet infrastructures. The adoption tools have focused on several network configurations as discussed in Table II.

TABLE II
RINA ADOPTION TOOLS

Tool	Purpose
shim-tcp-udp	RINA flows over TCP/IP network
iporina	IP flows over RINA network
rinagw	Interfacing of RINA and IP applications

The proposed model investigates the impact of one of the adoption tools i.e. *shim-tcp-udp* in RINA-based IoT SCADA communication over the internet. It also provides the response times of SCADA control and data acquisition. It also explored the possible reasons for the increased response times when compared to the native IP-based SCADA application.

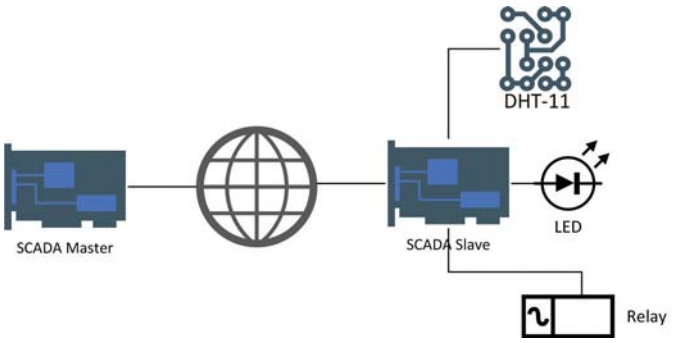

Fig. 1. Proposed IoT-SCADA model

III. NETWORK MODEL

The proposed IoT SCADA utility is implemented in master-slave mode using Raspberry Pi devices. One Raspberry Pi operates like a master and the other as a slave. A RINA-based SCADA application is developed to control the actuators and sensors. A GUI is provided in the SCADA Master application to control the actuators and to display the acquired sensor data along with response times. Fig. 1 shows the layout of IoT-SCADA utility The SCADA slave node consists of the actuators like LED, Digital relay and sensors like weather monitoring sensor DHT11. The interfacing of sensors and actuators is done with the help of *Wiring Pi* module. The physical model of the proposed SCADA slave is shown in Fig 2.

The functionality of the proposed SCADA utility is to stimulate supervisory control and data acquisition using the above actuators and sensors over the RINA network. So, RINA-based SCADA master-slave applications are developed with the help of its native socket API [14]. Both SCADA applications are designed and deployed to communicate in client-server mode.

A. RINA network model

Fig. 3 represents the RINA network model of the proposed SCADA utility. It shows that the SCADA master and slave are connected over the internet using *shim-tcp-udp*. *shim-tcp-udp* is one of the RINA adaption tools where RINA

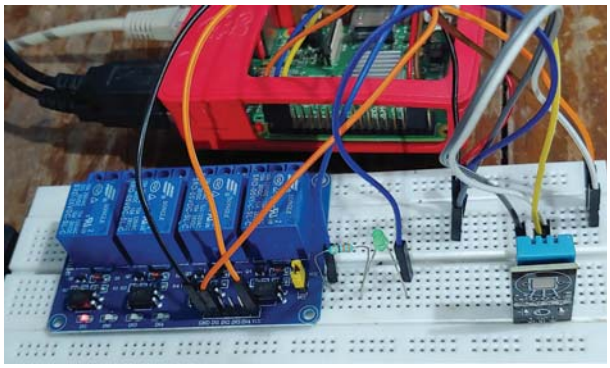

Fig. 2. SCADA Slave implementation

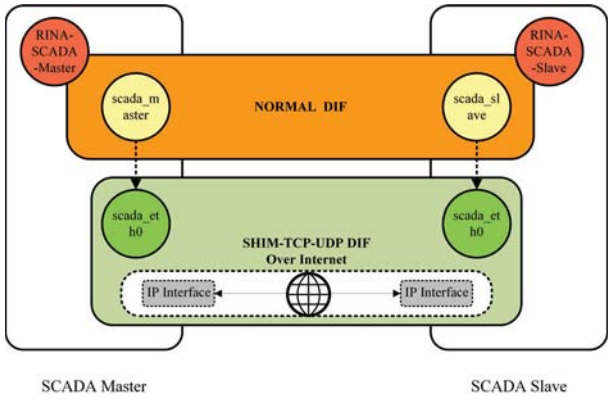

Fig. 3. RINA network model of proposed SCADA utility

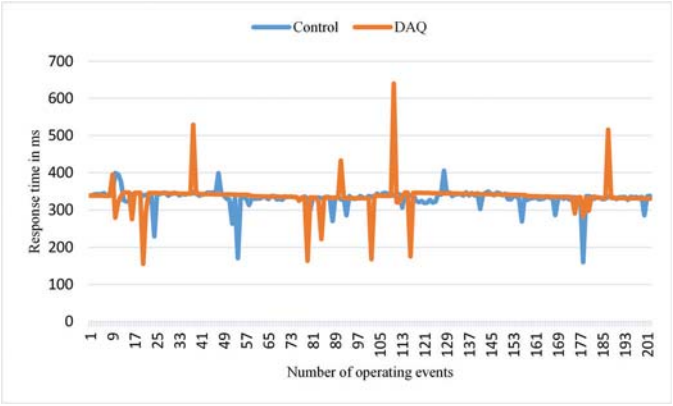

Fig. 5. Response times vs Number of operating events

applications communicate with each other over the existing IP infrastructure.

We have two types of IPCPs in the RINA network i.e. normal IPCP and shim IPCP. The normal IPCP can exist at any level of the network stack except the lowest level. The lowest level IPCP is called shim IPCP. The normal IPCP is meant for interacting with applications whereas shim IPCP is meant for interfacing the hardware devices. RINA has two types of lowest level IPCPs i.e. *shim* and *shim-tcp-udp*. *Shim* IPCP implements communication flows over the legacy Ethernet using VLAN tagging. *Shim-tcp-udp* IPCP implements communication flows over the existing IP infrastructure with the help of public IP addresses. The advantage of *shim-tcp-udp* lies in executing RINA applications where pure RINA flows are not possible due to infrastructure limitations.

In this network model, we have two sets of IPCPs i.e. *scada_master*, *scada_slave* and *scada_eth0*, and *scada_eth0*. The top-level IPCPs i.e. *scada_master*, *scada_slave* belong to *normal* type of IPCP and form a *normal* type of DIF, whereas *scada_eth0* belong to *shim-tcp-udp* type and form a *shim-tcp-udp* DIF. Fig. 4 describes the configuration of *shim-tcp-udp* that should be configured in each host. The significance of the configuration parameters is explained as follows

- *hostname* - represents the IP address of the local node.
- *dirEntry* - represents the name of the remote IPCP along with the public IP of that node.
- *expReg* - represents the IPCP that handles data transfer in the local node.

Here, *scada_master* and *scada_slave* represent the IPCPs connected to SCADA master and slave applications respectively. We have shown the configuration of the SCADA master in Fig. 4 and the SCADA slave also configured vice-versa. From Fig. 4, it can be identified that the public IP of the SCADA master is given in *hostName*. The IPCP of the SCADA slave is specified in *dirEntry* along with its IP address. Besides that, a port number should be provided which acts as an interface between IP and RINA communication flows. This provision is limited to converting RINA flows into IP flows only, it doesn't have any role in RINA applications since its port is internal to IPCP. Once the configuration is finished, the connection is established between RINA applications as explained in [15].

```
"difType" : "shim-tcp-udp",
"configParameters" : {
"hostname":"x.x.x.34",
"dirEntry":":1:scada_slave::x.x.x.214:2426",
"expReg":":1:scada_master::2426"
}
```

Fig. 4. *shim-tcp-udp* configuration

IV. Results

After establishing the successful connection between the RINA SCADA applications over the internet, response times for control and data acquisition operations are observed for a certain time. The variation of response times for several operating events of control and data acquisition is shown in Fig. 5. It can be observed that the response times are under the limits mentioned by the IEEE standards for automation and SCADA communication [16]. This will provide the applicability of *shim-tcp-udp* in IoT-SCADA systems. We have presented the minimum, maximum and the average response times of SCADA utility as shown in Table III. It provides us behavior of RINA network in SCADA utilities. The minimum and average response times of the control and data acquisition are almost similar but data acquisition operations took longer time in certain occasions.

TABLE III
RINA-SCADA RESPONSE TIMES

Response times in ms	Minimum	Maximum	Average
Control	160.05	404.79	332.62
DAQ	155.43	639.99	336.29

At the same time, we have developed the same type of SCADA system for IP networks and executed it to assess the performance of RINA-based SCADA applications which have

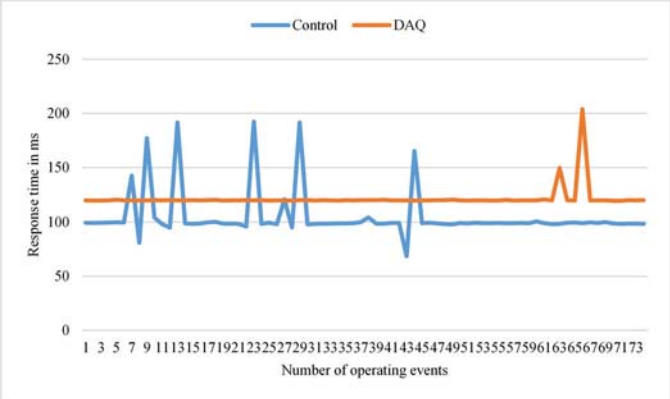

Fig. 6. Response times vs Number of operating events

utilized *shim-tcp-udp* tool. The response time in IP networks is much lesser than the RINA network as shown in Fig. 6. It can be observed from Fig. 5 and Fig. 6 that response time is increased in RINA when compared to TCP/IP network. Fig. 7 shows the comparison of the average response time of the control and data acquisition operations between both networks and emphasizes the communication overheads in RINA flows. The response times of the RINA application have increased almost 3 times to that of response times in the IP network.

The reason for these increased response times lies in converting every RINA packet into a TCP packet by the *shim-tcp-udp* tool. Since we choose reliable connection in RINA, *shim-tcp-udp* employed TCP connection to communicate between the two nodes. Hence, every RINA data packet will be converted to a TCP packet by *shim-tcp-udp*. As RINA implemented *reliable* flow, the other node tends to acknowledge whatever it receives. This can be observed in Fig. 8 where we have used Wireshark to capture the network flow. It shows us the network flow for one control/data acquisition event that was originated from the SCADA master. It can be observed that the data packet from the RINA application is converted to a TCP packet at the sending end and vice versa at the receiving end. After that, the receiving node acknowledges the

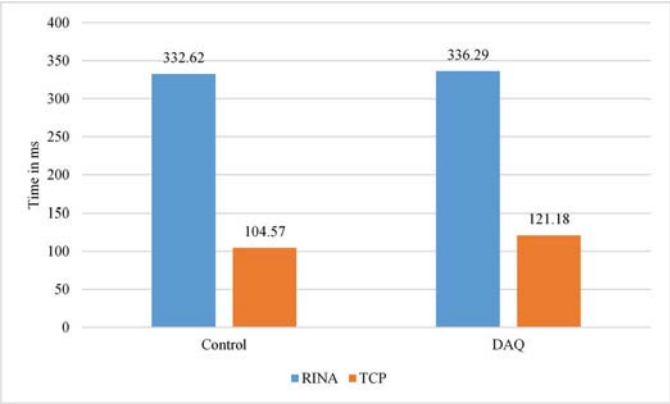

Fig. 7. Comparison of average response time

Fig. 8. Network flow of RINA over IP network

reception of data. Now as the RINA application received the data packet, it again responds with a RINA acknowledgment message which will be converted as a TCP packet. This kind of network abstraction increased the response times when RINA applications use the *shim-tcp-udp* tool.

V. Conclusion

The proposed work investigated the impact of the RINA adaption tool *shim-tcp-udp* on the public internet. We have discussed the necessity of recursive internetworking architecture in the context of modern networking demands. We have discussed the advantage of RINA through its design and functionality. We have developed a RINA SCADA application with the help of IoT devices and communicated over the internet with the help of one of its adaption tools *shim-tcp-udp*. The response times in control and data acquisition operations are presented along with their minimum, maximum and average values. A comparative analysis is presented to find out the communication overheads caused by *shim-tcp-udp*. The probable causes of these overheads are discussed with the help of network flow captured using the Wireshark tool. We conclude that RINA applications can be successfully implemented over the internet using its adaption tool *shim-tcp-udp* but at the cost of increased response times. This work may be extended to enhance the *shim-tcp-udp* tool so that response times may be reduced. The limitation of this tool is the requirement of public IPs or the requirement of expertise in configuring the incoming and outgoing routers to forward the data packets to the concerned host.

References

[1] R. Khondoker, B. Nugraha, R. Marx, and K. Bayarou, "Security of selected future internet architectures: A survey," in *2014 Eighth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*. IEEE, 2014, pp. 433–440.

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

[3] 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.

[4] 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.

[5] 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.

[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] E. Grasa, E. Trouva, P. Phelan, M. P. de Leon, J. Day, I. Matta, L. T. Chitkushev, and S. Bunch, "Design principles of the recursive internetwork architecture (rina)," 2011.
[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] 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.
[10] 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.
[11] Y. Wang, I. Matta, and N. Akhtar, "Experimenting with routing policies using protorina over geni," in *2014 Third GENI Research and Educational Experiment Workshop*. IEEE, 2014, pp. 61–64.
[12] K. A. Hiorth and M. Welzl, "Design considerations for rina congestion control over wifi links," in *2019 22nd Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN)*. IEEE, 2019, pp. 54–59.
[13] "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.
[14] 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.
[15] N. B. Samyuel and B. A. Shimray, "Securing iot device communication against network flow attacks with recursive internetworking architecture (rina)," *ICT Express*, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405959520300461
[16] I. PES, "Ieee standard for scada and automation systems," *vol. IEEE Std C*, vol. 37, 2008.