Bridge Me If You Can! Evaluating the Latency of Securing Profinet

Stephan Hohmann
Fraunhofer FOKUS
Berlin, Germany
stephan.hohmann@fokus.fraunhofer.de

Tobias Mueller
Universität Hamburg
Hamburg, Germany
mueller@informatik.uni-hamburg.de

Marius Stübs
Universität Hamburg
Hamburg, Germany
stuebs@informatik.uni-hamburg.de

***Abstract*—Fieldbusses have been the backbone of inter-device communication in both industrial and home automation settings for a few decades. The underlying assumption is the availability of reliable and low-latency communication for all busses. This often implies that the busses are confined to a single physical location. With the advent of the 'Internet of Things' (IoT) and succinctly the 'Industrial Internet of Things' (IIoT) and the increased demand for control logic to be pushed into the 'Cloud', that assumption can no longer be upheld. Since no (I)IoT protocol exists to provide remote control, let alone in a secure fashion, while providing low latency at the same time, we are left with the problem of routing fieldbusses from, say, data-centres to shop-floors. This presents a challenge, because those busses have been designed for safety rather than security.**

In this paper, we elaborate on the viability of routing layer two fieldbus traffic while providing both: low latency to fulfil real-time requirements and security through cryptographic tunnels. We design and implement a network topology where Profinet traffic is routed through a VXLAN over Wireguard overlay to control a SoftPLC instance. We evaluate our implementation in a realistic test-bed and our measurements indicate that bridging Profinet over VXLAN and Wireguard induces a latency low enough for running time-critical applications.

***Index Terms*—fieldbus, IoT, routing, bridging, networking**

I. Introduction

With the rise of industrial Internet of Things (IIoT) systems, the ability to remote control stations such as turbines, centrifuges, or power plants, has become more important. The stations work with link-local fieldbusses such as Profinet or Modbus, which assume an isolated and trusted networking environment. For distributed stations, e.g. multiple offshore windmills, forming a group of interconnected nodes is challenging. This is particularly true when the stations ought to be connected through the Internet, as it can hardly be described as an isolated network. Additionally, the development continues towards the integration of IIoT into an inter-operable landscape to provide semantic services that vertically connect different domains [1].

Hence, the communication systems need to be hardened, which in turn involves the use of cryptography in order to prevent man-in-the-middle attacks. Such attacks must be prevented, because they give an attacker the opportunity to manipulate control commands in transit from the data-centres to the destination, e.g. shop floors. Cryptographic tunnels can guarantee integrity and even confidentiality of the sent packets, effectively thwarting attacks on the communication channel. Even more challenging than providing security is to provide low latency as required by some stations. Any delay that causes packets to arrive late will harm the operation, potentially resulting in malfunction or even destruction of the affected device. From the information perspective, delayed messages can lead to an incorrect representation of the current system state: Even if the SCADA devices are explicitly designed to be robust against denial-of-service attacks, delayed messages can invalidate the processed data as a whole and result in counter-productive or overshooting actions by distributed control algorithms [2]. Thus, the control units need to be operated in a secure environment. As a prominent example, we refer to the Iranian centrifuges which were destroyed due to a malicious modification of the control unit [3]. The control unit then sent malicious commands to the centrifuges, causing them to prematurely degrade by driving particularly stressful movement patterns resulting in their eventual destruction. In an effort to mask the operation, sensor readings were fabricated to suggest normal behaviour.

With this work, we aim to prevent such malicious commands in a networked environment by using protection provided by cryptographic algorithms. With our approach, an operator of IIoT systems will be able to secure their communication over the Internet while still maintaining a sufficiently low latency, despite the additional steps the cryptography requires.

One important challenge in securely transmitting fieldbus protocols is that the separation of layers according to the ISO OSI transport layers model is not made in a clean manner, i.e. the data link layer is directly accessed by upper layers [4]. This greatly complicates the migration towards the IIoT paradigm, opening up the research questions that are investigated in this paper. To this end, this paper makes the following contributions:

1) We analyse the requirements such as confidentiality, integrity protection, and delay limits.
2) We propose an architecture for securely bridging IIoT devices in several physical locations in order to enable them to be controlled remotely.
3) We measure the latency of secure bridges in a realistic test-bed. We investigate pure Ethernet and Wireguard as baselines and add VXLAN to forward Profinet frames.

II. Background

This section provides the concepts required for understanding our secure bridging architecture.

A. Cloud Computing in IIoT

The current development of moving both information technology (IT) and operational technology (OT) towards the cloud is a particular feature of the edge computing paradigm, or fog computing, as coined in 2012 by Cisco Systems [5]. This paradigm acknowledges the criticism of the high delays caused by separating data collection and data processing and addresses them through a hierarchical architecture (fig. 1). By introducing fog nodes for pre-processing of the data and for real-time processing of time critical operations, the inherent differences between IT and OT are better reflected. Additionally, the security requirements demand fog nodes to be physically separated from the higher-level processing and to be supervised on-premise [6]. In terms of security, tampering with data streams between the data-centre and the destination, e.g. shop floor (at the fog nodes), gets significantly harder the closer they are together [7], which again is an advantage of fog computing over cloud computing.

B. Programmable Logic Controller (PLC)

In industrial settings, PLCs are arguably among the most relevant devices as they exert control over those machines which are most important for the production or maintenance of goods required for conducting business. If a PLC loses control over machinery, it may damage goods, or worse, destroy equipment. The PLCs deployed in industrial settings tend to be specialised in performing controlling tasks. Performing computations that are outside or not directly related to those purposes may present a challenge. This includes running cryptographic algorithms for securing communication: Resource constraints as well as missing support facilities can hamper their adoption. Historically, PLCs assume control frames to be sent, transmitted, and retrieved in a trusted network setting. While it can be argued that this assumption has been valid for a long time, we believe, with the advent of distributed sites and the IIoT in mind, this no longer holds true.

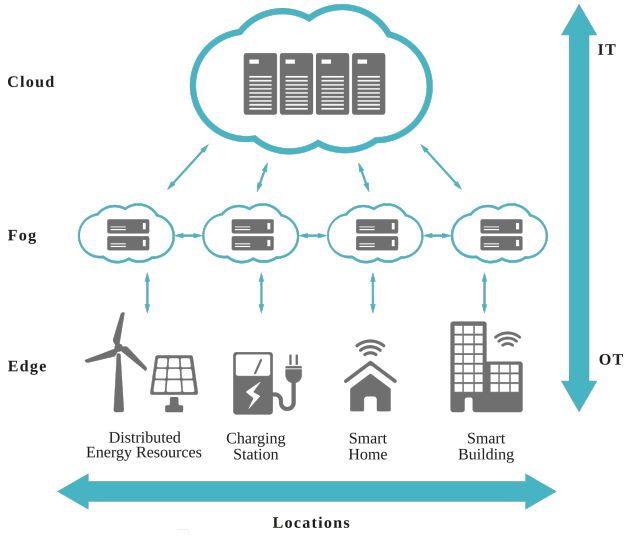

Fig. 1: The hierarchical architecture of fog computing reflects the differences between IT and OT.

C. Profinet

Profinet is an adaptation of the venerable, RS485-based Profibus fieldbus protocol, designed to sustain on raw Ethernet. In order to keep feature-parity between Profinet and Profibus and to meet demands for a genuinely networked environment, a wealth of established protocols are pressed into service: e.g. the Link-Layer Discovery Protocol (LLDP) is augmented to the Device Configuration Protocol (DCP), allowing Profinet controllers to set layer four coordinates on peers.

Championed by SIEMENS, Profinet (and in extension Profibus) is mostly found in industrial settings. Specialised equipment leveraging Profinet communication is available but only needed in situations calling for advanced real-time functionalities such as distributed clock synchronisation. Both, cyclic and acyclic real-time qualities, are available on commercial off-the-shelve hardware. For our intents and purposes, that shall suffice.

As with most industrial Ethernet solutions, Profinet provides exchange of structured messages without a means of transmitting semantics. This results in small, easy to process messages, but also requires peers to agree on the structures and the meaning of fields contained therein before runtime.

Being located on layers two to four means that Profinet traffic can be bridged, but neither masqueraded nor forwarded. It can be sustained in 802.1q-style (VLAN) insulation, though. Being distributed in this manner provides a suitable test case for us as the treatment of layer two and four traffic is quite different in Profinet: The complexity of the protocol suite would provoke errors in the layer two isolation early and in dramatic fashion. This sets Profinet apart from other established industrial Ethernet solutions such as Ethercat, Modbus/TCP, and Ethernet/IP.

D. Codesys

Developed and sold by Codesys GmbH (formerly: 3S-Smart Software Solutions GmbH), Codesys[1] is an IEC 61131 compliant SoftPLC solution. A plethora of device drivers, network stacks, and supported platforms are provided which makes it a promising test ground.

The Codesys solution is roughly divided into two components: an Integrated Development Environment (IDE) and a runtime. Projects created in the IDE can be compiled into bytecode and then injected into the runtime for execution.

E. Layer 2 Tunnelling Protocols

Our goal is to connect a remote controller with IIoT devices in multiple remote sites (fig. 2). Our intention is to use some form of Virtual Private Networking (VPN), but since Profinet is exposed as Ethernet frames rather than IP packets, we cannot

[1] https://www.codesys.com

directly make use of readily available tunnelling software such as Wireguard, TINC, OpenVPN, or IPSec. We will use Virtual eXtensible Local Area Network (VXLAN) [8] for forwarding the Profinet Ethernet frames as it is a popular protocol within data centers and has gained wide adoption among vendors for networking equipment.

As VXLAN does not provide any protection against network-based attackers, we will use Wireguard to protect our traffic. Wireguard is a modern tool with a special focus on usability. The usability is increased by a well-designed reduction of complexity and configuration options, for example it implements exactly one cryptographic algorithm per purpose: Curve25519 for the key exchange, ChaCha20 as stream cipher, Poly1305 as message authentication code, etc [9]. This is in stark contrast to other tunnelling protocols such as IPsec [10] which are capable of using many different algorithms for one purpose. This flexibility comes at a cost, namely negotiation during the protocol run or administration on the end points. As of 2020, Wireguard is the default VPN for Ubuntu 20.04 and has been back-ported to Ubuntu 18.10 for compatibility reasons. A survey from 2019 has shown that Wireguard is remarkably scalable in terms of VPN clients. In the comparison with several technologies, namely OpenVPN, ZeroTier, Tinc, and SoftEther VPN, it is found that the response time of Wireguard is especially good when scaling large [11]. This makes it a fitting technology especially with regards to the application within IIoT-inspired SCADA environments such as Fog or Edge computing setups.

III. Problem

Fieldbusses are used in low latency environments and as such they are restrictive to the delays that might be introduced by additional functionalities. Unfortunately, routing traffic over the Internet adds the risk of datastream compromise by unauthorised parties. This makes it obligatory to add new features, such as cryptographic tunnels. These would solve several requirements at once: provide authenticity, prevent malicious commands, and guarantee confidentiality. Cryptography can introduce a significant delay to any process, therefore it is important to evaluate how to reduce the delay to the bare minimum. Besides the latency, the following problems exist when attempting to bridge Profinet traffic across multiple sites:

a) Encapsulation: While various layer two tunnel protocols exist (section II-E), e.g. IPsec, L2TP, or Wireguard, they do not address the challenges involved when tunnelling Profinet frames. Thus, Profinet can technically not be tunnelled as such, since the Profinet protocol spans several layers in the communication stack, especially Ethernet frames in contrast to application data. Therefore, additional encapsulation is necessary, further contributing to the possible causes of delay.

b) Legacy Hardware: Newer systems may make use of Open Platform Communications - Unified Architecture (OPC-UA), because it models all resources in a hierarchical tree. This is not compatible with real-life scenarios. Profinet, in comparison, has real-time guarantees: when control packets do not arrive on time, devices enter a failure state. No such guarantee is given by OPC-UA.

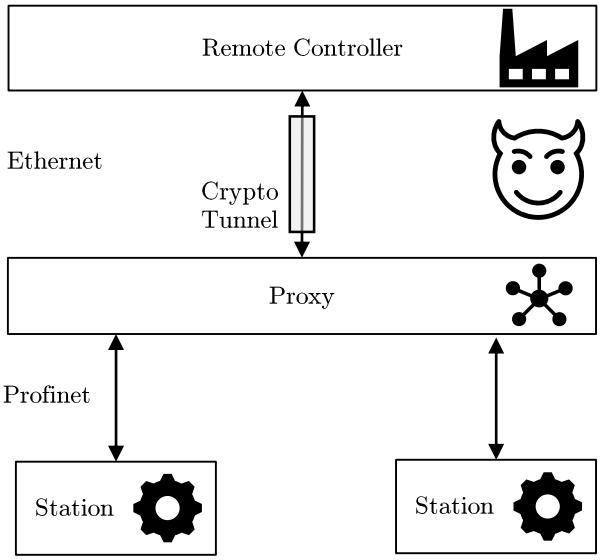

Fig. 2: Architecture of our prospective setup to bridge legacy stations in order to allow them to be remote controlled through a secure connection.

c) High Density of Devices: We regard the deployment of wireless communication technologies as problematic, because it introduces several challenges outside the scope of this paper such as the interference of communication signals. The traditional approach of using standard copper wire has the advantage of being robust, especially in shop-floors with several IIoT devices connected in series.

IV. Measuring the Security-Induced Latency

This section presents our experimental setup and the design of our experiment.

The proposed solution is to firstly encapsulate the Profinet traffic in the data-centre into UDP packets through VXLAN. The UDP packets are then transmitted towards the shop floor using an encrypted Wireguard tunnel. Once the packets arrive, the Profinet communication is restored from the again-decrypted UDP packets.

A. Architecture

In our architecture we have three elements: Remote Controller, Proxy, and Controlled Station.

Our experimental setup consists of two MinnowBoard Turbot Dual Ethernet Quad Core Boards. Both come equipped with an Intel Atom E3845[2] clocked at 1.91GHz and two Intel® i210-AT[3] 1Gb Ethernet cards with four queues and two physical ports, each. This allows us to free one port each

[2]https://ark.intel.com/content/www/us/en/ark/products/78475/intel-atom-processor-e3845-2m-cache-1-91-ghz.html

[3]https://ark.intel.com/content/www/us/en/ark/products/64400/intel-ethernet-controller-i210-at.html

Fig. 3: The hardware setup of our test-bed: Two MinnowBoards acting as Profinet peers are connected directly via Ethernet. A Raspberry Pi 3B+ is acting as a control node. Management traffic is routed through dedicated ports; RS232 terminals are in service as backup. The left MinnowBoard is our chosen Profinet controller while the right board is our Profinet device.

for management duty and arrange a direct connection as test track, thus achieving sufficient isolation (fig 3).

Both nodes were provisioned with Debian 11 (Bullseye), running a v5.6.0 Linux kernel with real-time patches applied. Codesys v3.5.16.10 runtimes were provided through privileged Docker containers in an effort to mitigate dependency issues between the runtimes and the packages provided by Debian. Experience gained from prior experiments suggests no or negligible impact on performance when operating the runtime in this fashion.

In addition, two CPU cores were exempted from OS scheduling through the `isolcpus=` kernel parameter to allow for unimpeded operation of the runtime. Cores were prevented from entering idle states via `intel_idle.max_cstate=0`. Likewise, the relevant Ethernet interfaces had Energy Efficient Ethernet (EEE) disabled.

To simulate a roughly realistic data exchange, both nodes were injected with a custom Codesys project, turning one node into a Profinet master (the Remote Controller in our setup) and the other into a Profinet device (vulgo: 'slave,' the Controlled Station). In the course of this, a process image was submitted with a bit toggling every 125 ms. This provoked cyclic exchange of Ethernet frames 64 bytes in size. It is worth mentioning that this is a size commonly found in real applications. Both parts of the application have been organised into the following three threads:

- A thread housing the higher logic, clocked at a 50 ms clock cycle,
- a thread for Profinet communications allowed to free-cycle (i.e. spin as fast as possible),
- and a Profinet I/O task, set for a 1 ms cycle.

The Profinet controller has been primed for a 1 ms send clock with a reduction ratio of four, resulting in a process image being transferred every fourth millisecond. Both peers contain logic to count the rising edges of the toggling bit. This is to account for frames lost in a manner that may not trigger a bus error. If that were to happen, the counters would notably diverge.

B. Remote Controller to Proxy

The communication between the controller and the station to be controlled is not performed through a direct Ethernet connection. Instead, it talks through the proxy which, in turn, establishes a connection to the controlled device. Since we assume that the controlled device is not capable of running cryptographic algorithms, we have placed the proxy between the two parties. This proxy mediates the traffic by encrypting and decrypting the packets before forwarding them.

Our Proxy is custom software which is capable of translating Profinet frame to Ethernet frames and, additionally, forwarding those frames securely.

C. Proxy to Station

Since our remote controller does not talk directly to the controlled device, the Profinet frames are translated by the proxy in the middle. Some literature refers to this architecture as "bump in the middle".

V. Evaluation

This section presents the results obtained through the experiment mentioned in the previous section.

We ran several bridging and non-bridging setups and measured their latency through the `iperf3` tool. Table I lists the recorded timings. The first row shows the round-trip time for simple UDP packets over Ethernet from the (thought) datacentre to the shop floor and back, thus establishing the baseline latency. The second row displays the round-trip latency that we measured when encapsulating them into VXLAN packets, reflecting our need for proper insulation of Profinet frames. The third row contains measurements from traffic routed through Wireguard encryption, as done before in [11] without

TABLE I: The latency (in microseconds) for different bridging protocols. The measurements were taken from 1000 iterations of running `iperf3 --affinity 2,3 --udp --b100M --client <peer>`. Ethernet is the baseline and VXLAN is the basic bridging protocol for transporting Profinet frames. Wireguard and Wireguard+VXLAN are cryptographically protected methods for bridging packets.

Method	min	p25	med	p75	max
Ethernet	23	41	47	56	73
VXLAN	35	52	57	61	84
Wireguard (WG)	46	65	74	86	180
WG+VXLAN	47	70	81	87	118

VXLAN. Finally, the fourth row shows the results of the final setup where packets are bridged using VXLAN on top of an encrypted Wireguard tunnel.

Our results indicate that mere bridging of packets over Ethernet through VXLAN increases the latency in the worst (best) case by 52.2% (15.1%). However, we could observe a median increase of latency by 21.3%.

Adding cryptography to the baseline channel increases the latency from 47 µs to 74 µs, or 57.4%. In the worst (best) case, the latency increased by 147% (53.6%). Transmitting packets over the encrypted channel through VXLAN additionally increased the median latency by 9.5%. We note that we have observed a decrease in latency from 180 µs to 118 µs or 35%. However, we consider this data point an outlier as our test setup ensured 1000 repetitions of the experiment and the latency has increased in all but the most extreme cases.

Compared to the unencrypted bridging, we observed an increased in latency from 57 µs to 81 µs or 42.1% in the median case. The increases range from 34.3% to 42.6%.

In addition to running the rather synthetic, throughput-oriented `iperf3` benchmark, we decided to take advantage of the statistics provided by the Codesys IDE as presented in Table II. In doing so, we established Profinet connectivity and recorded the average send and receive times after transmitting 10 000 frames.

We could not observe breakage of the cyclic data exchange as indicated by the absence of bus errors. The counts for the rising edges of the control bit also did not diverge. Given that the set parameters would result in a smallest possible timeout[4] of 12 ms, the recorded timings are perfectly acceptable and seem to hint at the feasibility of even tighter cycle times. The nodes exhibited no signs of CPU saturation, suggesting I/O-bound workload.

VI. Related Work

To the best of our knowledge, existing work addresses the security of distributed IIoT systems communication, but does not take the low-latency requirements into account. In fact, the cost of securing the communication has already been explored [12], albeit with a strong focus on energy consumption rather than on whether the latency still meets the real-time requirements. It has been found that AES-CBC is most efficient [12], but still significantly more expensive than having no cryptography at all. Hence we focus on securing the channel to the proxy rather than to the actual controlled station. One method of achieving such a secure communication is iTLS [13] which in turn makes use of TLS [14] connections for securing communication. However, the overhead involved in cryptography is non-negligible, i.e. roughly 10 ms.

Using IPsec over a 6LoWPAN wireless channel [15] works in scenarios that do not depend on a high density of devices. The mechanism cannot simply be transferred to cable bound networking, because it does not have the security mechanisms provided by IEEE 802.15.4.

It is worthwhile mentioning, that depending on the use case, the encryption details should be adjusted to the constraints. A comparison between DTLS and IPsec has shown that DTLS is more suitable for a memory limited environment, while IPsec performs better in an environment with less computational resources [16]. The assumption of exclusively using CoAP or MQTT cannot be upheld in the industrial context, especially for machines that are designed to run for decades. Similarly, 6LowPAN is not suitable for certain environments due to electromagnetic shielding or inference. Another important aspect is that retrofitting appliances is more economically feasible with cable-bound networking, also because the appliances can be daisy-chained. If wireless communication was an option, we refer to a study on wireless technologies, including 6LoWPAN and Bluetooth LE [17].

In our scenario, we assume that the station to be controlled cannot afford cryptographic operations. We note that academia has investigated securing the stations themselves [18].

Due to space constraints we refer to existing work for a more comprehensive study of the challenges and potential solutions for IIoT systems [19].

[4]The lower barrier is determined by send clock (1 ms) times reduction ratio (4) times 3

TABLE II: Average timings for sending (TX) and receiving (RX) frames on remote controller (Controller) and controlled device (Device) side in microseconds as reported by the Codesys IDE after 10 000 transmissions.

	Ethernet	**VXLAN**	**WG+VXLAN**
Controller RX	3.90	3.70	4.22
Controller TX	6.86	12.10	18.90
Device RX	16.33	19.90	14.35
Device TX	6.63	12.20	16.90

VII. Discussion

In our scenario we assume hard real-time requirements and cable-bound networking. We base the first assumption on our experience and note that industrial control machines, such as centrifuges or turbines, will enter an error state if they do not receive instructions on time. The second assumption is based on inherent problems of wireless communication technologies, such as signal blocking with many devices. The main finding is that we were able to bridge Profinet over Wireguard+VXLAN: round-trip times in cyclic data exchange resulted in no timeouts. The narrowest bottlenecks are assumed to be the NICs and intermediaries. In this area, we see potential for future research on how to achieve better timings. This would in particular concern techniques such as offloading frame processing from the CPU to capable hardware, switching the medium from copper wire to optical fibre, or augmenting the crypted traffic through Time-Sensitive Networks (TSN), thus mitigating scalability issues and keeping jitter in check. The most important fact is that we show that it is feasible to bridge fieldbus protocols over crypto tunnels in general. That is an important step for industrial edge and fog computing.

VIII. Conclusion

In this paper, we have shown how to efficiently bridge a remote controller with a station through an untrusted network. In our experimental setup we make use of hardware and software which are in use for IIoT projects and reflect real-world scenarios. We argue that industrial control systems have hard real-time requirements. Our evaluation presents the technical details that are important for achieving remote control when bridging Profinet using VXLAN and Wireguard and we discuss these with regards to their latency. We consider it worth mentioning that we believe the proposed solutions can be applied to other Ethernet-based fieldbusses that do not require specialised network adaptors.

IX. Acknowledgements

This research was partly (or fully) supported by the H2020 IoTwins project (Distributed Digital Twins for industrial SMEs: a big-data platform) funded by the EU under the call ICT-11-2018-2019, Grant Agreement № 857191

References

[1] A. Willner, C. Diedrich, R. B. Younes, S. Hohmann, and A. Kraft, "Semantic communication between components for smart factories based on onem2m," in *2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, IEEE, 2017, pp. 1–8.

[2] M. Stübs, P. Dambrauskas, M. H. Syed, K. Köster, H. Federrath, G. M. Burt, and T. Strasser, "Scalable power system communications emulation with opc ua," in *Proceedings of CIRED conference, Madrid*, Madrid, Spain: AIM, 2019, pp. 1–5. DOI: 10.34890/894.

[3] N. Falliere, L. O Murchu, and E. Chien. (Feb. 2011). W32.Stuxnet Dossier, [Online]. Available: https://cyber-peace.org/wp-content/uploads/2013/06/w32_stuxnet_dossier.pdf (visited on 03/29/2019).

[4] P. Bellagente, P. Ferrari, A. Flammini, S. Rinaldi, and E. Sisinni, "Enabling profinet devices to work in iot: Characterization and requirements," in *2016 IEEE International Instrumentation and Measurement Technology Conference Proceedings*, IEEE, 2016, pp. 1–6.

[5] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*, 2012, pp. 13–16.

[6] C. Alcaraz, "Secure interconnection of it-ot networks in industry 4.0," in *Critical Infrastructure Security and Resilience*, Springer, 2019, pp. 201–217.

[7] M. Al Yami and D. Schaefer, "Fog computing as a complementary approach to cloud computing," in *2019 International Conference on Computer and Information Sciences (ICCIS)*, IEEE, 2019, pp. 1–5.

[8] M. Mahalingam, D. G. Dutt, K. Duda, P. Agarwal, L. Kreeger, T. Sridhar, M. Bursell, and C. Wright, "Virtual eXtensible Local Area Network (VXLAN): A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks," RFC Editor, RFC7348, Aug. 2014, RFC7348. DOI: 10.17487/rfc7348. [Online]. Available: https://www.rfc-editor.org/info/rfc7348 (visited on 09/20/2020).

[9] J. A. Donenfeld, "Wireguard: Next generation kernel network tunnel.," in *NDSS*, 2017.

[10] S. Kent and K. Seo, "Security Architecture for the Internet Protocol," RFC Editor, RFC4301, Dec. 2005, RFC4301. DOI: 10.17487/rfc4301. [Online]. Available: https://www.rfc-editor.org/info/rfc4301 (visited on 09/20/2020).

[11] T. Goethals, D. Kerkhove, B. Volckaert, and F. De Turck, "Scalability evaluation of vpn technologies for secure container networking," in *2019 15th International Conference on Network and Service Management (CNSM)*, IEEE, 2019, pp. 1–7.

[12] S. Alharby, A. Weddell, J. Reeve, and N. Harris, "The cost of link layer security in iot embedded devices," *IFAC-PapersOnLine*, vol. 51, no. 6, pp. 72–77, 2018.

[13] P. Li, J. Su, and X. Wang, "Itls: Lightweight transport layer security protocol for iot with minimal latency and perfect forward secrecy," *IEEE Internet of Things Journal*, 2020.

[14] E. Rescorla, *The Transport Layer Security (TLS) Protocol Version 1.3*, RFC 8446, Aug. 2018. DOI: 10.17487/RFC8446. [Online]. Available: https://rfc-editor.org/rfc/rfc8446.txt.

[15] S. Raza, S. Duquennoy, J. Höglund, U. Roedig, and T. Voigt, "Secure communication for the internet of things—a comparison of link-layer security and ipsec for 6lowpan," *Security and Communication Networks*, vol. 7, no. 12, pp. 2654–2668, 2014.

[16] V. Kuna, *Performance analysis of end-to-end dtls and ipsec based communication in iot systems*, Karlskrona, Sweden, 2017.

[17] T. Gebremichael, L. P. Ledwaba, M. H. Eldefrawy, G. P. Hancke, N. Pereira, M. Gidlund, and J. Akerberg, "Security and privacy in the industrial internet of things: Current standards and future challenges," *IEEE Access*, vol. 8, pp. 152 351–152 366, 2020.

[18] A.-V. Duka, B. Genge, P. Haller, and B. Crainicu, "Enforcing end-to-end security in scada systems via application-level cryptography," in *International Conference on Critical Infrastructure Protection*, Springer, 2017, pp. 139–155.

[19] H. HaddadPajouh, A. Dehghantanha, R. M. Parizi, M. Aledhari, and H. Karimipour, "A survey on internet of things security: Requirements, challenges, and solutions," *Internet of Things*, p. 100 129, 2019.