A Comparative Study on Virtual Private Networks for Future Industrial Communication Systems

Tim Lackorzynski
Chair of Privacy and Data Security
Technical University Dresden
Dresden, Germany
tim.lackorzynski@tu-dresden.de

Stefan Köpsell
Chair of Privacy and Data Security
Technical University Dresden
Dresden, Germany
stefan.koepsell@tu-dresden.de

Thorsten Strufe
Chair of Privacy and Data Security
Technical University Dresden
Dresden, Germany
thorsten.strufe@tu-dresden.de

***Abstract*—The future industrial networks will not be created from scratch. Rather, they will grow from existing installations without displacing legacy components. The secure integration of these legacy machines and networks will become an important building block in order to realize the vision of Industry 4.0. Secure and high performance virtual private networks (VPNs) will be necessary for that purpose.**

Therefore, we investigated and compared various VPN solutions. Their performance was tested on multiple hardware platforms ranging from very resource constrained to very powerful. Non-functional aspects, relating around security, manageability and ease of use, were discussed in order to assess their suitability for future use cases.

We arrive at clear recommendations on which software VPN solutions to choose for future industrial setups.

***Index Terms*—industrial networks, VPN, secure transport, network security, tunneling, industrial IoT, Industry 4.0**

I. Introduction

New trends like Industry 4.0 and big data processing for e. g. predictive maintenance or process optimization will introduce new technologies into industrial automation. They will demand higher levels of interconnection between various new and old types of devices and machinery, within an industrial facility as well as with the outside world. Networks will become very big and diverse, consisting of a mixture of new and legacy devices with connections to the Internet, as existing and amortized industrial machines will not be phased out for the sole reason of having outdated networking interfaces. Having a life cycle of multiple decades, industrial machines rather have to be integrated into future networks.

This heterogeneous environment inevitably increases the vulnerability to IT-security threats, as many examples of the recent past have shown (e. g. [9]). Furthermore, industrial machinery once running productively, is rarely changed or updated. This also includes their software. The main reasons for a lack of software updates are the age of the machinery (decades old machines just get none) and the fact that updating a machine would mean it had to be taken offline for that time and could not work productively. Furthermore, the risk of breaking the previously running system through the update is generally considered too high and updating parts of the machinery (including its software) might require a re-certification with respect to certain safety and security regulations.

Former flat Layer 2 factory networks will be transformed into more complex, hierarchical and modular Layer 3 networks [22] [1]. In order to conquer this new complexity, separation and virtualization of new as well as legacy components, networks and whole infrastructures will be employed [10]. Furthermore, those new trends also formulate the clear requirement of high performance. Network technologies added to that end, must only induce low latencies and allow for high bandwidth communication.

One approach on how to integrate legacy machinery into future network architectures is to retrofit them with special gateways, as proposed e. g. in [3]. These gateways pose as interfaces between the legacy layer 2 Ethernet of the machines and the modular layer 3 network of the envisioned smart factories of the future and effectively establish virtual bridges between legacy machines or whole legacy networks. Virtual private networks (VPNs) with the ability to securely bridge different local area networks (LANs) will be the type of software responsible for providing this service.

Therefore, this study investigates various software VPN solutions on their suitability to this task. We analyzed their performance as well as non-functional aspects, like security, configurability and ease of use. We evaluated the performance on multiple different hardware platforms, ranging from resource-restricted embedded platforms to powerful server platforms.

The remainder is organized as follows: Section II gives more background and motivates this work further, while Section III reviews the related work. Section IV introduces our experimental settings, the tools used for measurement, as well as the tested hardware platforms and software VPN solutions. Section V presents, discusses and compares results and Section VI concludes with lessons-learned and identifies possible future research questions.

II. Context and Motivation

Old factory networks used to be self-contained and physically separated from the outside world. This paradigm was called perimeter security. It meant, that no further protection of the communication as well as the devices within a network were necessary, as attackers would be stopped by physical barriers like fences and locked doors. Furthermore, all devices within the factory that were connected to the network, were

This research was supported by the BMBF (Federal Ministry of Education and Research), Germany, under the framework of the "Zwanzig20" program.
978-1-7281-1268-8/19/$31.00 ©2019 IEEE

relatively few and would be known and accounted for. For a lack of attacker surface, the network was considered secure.

However, the transformation of classic factories into Industry 4.0 smart factories will render this paradigm obsolete. Future industrial networks will consist of a very high number of heterogeneous components that will themselves have various connections to the Internet (e. g. industrial machines with service access via GSM or future 5G interfaces) in order to communicate to the cloud. This means, that the legacy hardware still installed in factories becomes susceptible to attacks from the outside. Yet, for multiple reasons, these machines receive few or no software updates, making them very vulnerable to attacks. Furthermore, even the concept of not relying on perimeter security has still not penetrated the industry, as a recent security study of a modern factory robot showed [17]. And as office and factory floors get more and more integrated, attack vectors that formerly only affected the office floor, will also become a threat for the devices within an industrial network (e. g. [8]). As a result, local networks and even communication flows that only exist between participants within a LAN cannot be considered secure per se anymore. Additional layers of protection are required.

An important tool for securing future industrial networks, that at the same time allows to integrate legacy devices, will be Virtual Private Networks (VPNs). They can be used to separate different entities within a factory by protecting and separating the data flows that are routed over common networking infrastructure. These entities may be individual legacy machines, whole legacy networks or even only legacy software running inside a virtual machine on a factory server.

Therefore, in order to be as technology and use case agnostic as possible, this study focuses on VPNs that can transmit Layer 2 Ethernet frames over the Layer 3 IP protocol. This means, that legacy devices and machines can be integrated transparently without the need to configure them (e. g. with information about gateway IP addresses or default routes).

Many VPN software solutions are available for that purpose. We concentrated on the freely available ones for Linux, as they would be the natural choice for operators trying to increase the security of their networks. In the following, we present the choices, we chose to compare:

a) OpenVPN: OpenVPN is a de facto standard VPN solution[1]. It runs as a user space application and is used here in bridge mode (it transports Layer 2 frames). OpenVPN allows to setup the secure tunnel using either UDP or TCP as underlying transport protocol.

b) IPsec: StrongSwan/IPsec is another de facto standard VPN solution[2]. In contrast to OpenVPN, it runs in the Linux kernel. Since it offers tunneling only on the IP-Layer, the Layer 2 Tunneling Protocol (L2TP) was used to be able to actually transmit Layer 2 frames over IPsec [14].

c) Tinc: Tinc is a VPN solution freely available for Linux operating systems[3]. It runs as a user space application and allows to bridge Ethernet segments.

d) Freelan: Freelan is a VPN solution available for many operating systems[4]. It runs as a user space application and offers Ethernet bridging. It uses different cryptographic algorithms compared to the other solutions.

e) SSH VPN: Secure Shell is a cryptographic network protocol used for protecting network services, mainly known for providing remote command-line login and remote command execution[5]. Yet, it can also provide Ethernet bridging.

f) MACsec: MACsec is a pure Layer 2 encryption scheme, that was relatively recently introduced into the Linux kernel [7]. While all previously mentioned VPN solutions offered multiple ciphers for encryption and authentication, the only available cryptographic scheme within MACsec, at the time of writing this paper, was AES-128-GCM used for authenticated encryption. Since MACsec only works on Layer 2, L2TP was used as Layer 3 tunneling protocol. While the payload is encrypted and authenticated, we acknowledge, that this is not a proper VPN solution and has some security-related drawbacks. Yet, we wanted to explore the performance of a conceivable proper solution.

g) Wireguard: Wireguard is a very new VPN solution, that is about to be integrated into the Linux kernel and aims to replace IPsec. Main design goals are easier configurability and higher performance [6]. It uses only ChaCha/Poly1305 for authenticated encryption and like IPsec only works on IP-packets. Therefore, an L2TP tunnel was set up within the Wireguard tunnel, so that actual Layer 2 frames could be transmitted. While the authors of Wireguard do not regard their software as stable yet, it is already considered a very promising solution.

III. Related Work

There have already been many studies published on VPN solutions. Yet those studies generally are either old and outdated, small in scope, only tested few hardware setups, or were geared towards certain specific use cases. Even fewer specifically dealt with industrial or embedded environments.

For example, Czybik et al. did investigate security schemes for industrial communication, but were only concerned with integrity protection for very small embedded platforms for real-time applications [5].

Numerous studies, that do performance comparisons only do so for few VPN solutions, and then only for a small number of cryptographic algorithms [12] [4] [19] [20] [16]. Hardware platforms, if mentioned at all, are specified as standard PC hardware and in some cases, the involved nodes are not even identical.

Another group of studies evaluates VPN solutions not from a performance standpoint, but from a standpoint of management with a focus on operational questions [21] [15] [13] [2].

Khanvilkar et al. did an extensive study on open source VPN solutions and investigated among performance also security properties and operational concerns [11]. Yet, this research was published in 2004 and is therefore quite outdated. Much of the discussed software is already deprecated and new approaches are not mentioned.

To the best of our knowledge, there is no large scale study comparing multiple recent VPN solutions on different hardware platforms.

[1]https://openvpn.net/
[2]https://strongswan.org/
[3]https://tinc-vpn.org/
[4]https://freelan.org/

[5]https://www.openssh.com/

Frame size (bytes)	60	128	256	512	1024	1400	1514
Weight	0.35	0.3	0.1	0.1	0.05	0.05	0.05

Table I: Frame sizes and weights used to calculate the weighted latency.

IV. Methodology

The aim of this work is to study and compare the VPN software solutions introduced in Section II. We tested their performance, but also investigated on non-functional properties and aspects in order to find the most suitable solution for future factory networks that demand high performance and security at the same time.

The performance parameters measured were throughput and latency over the secure tunnels provided by the VPNs. While the throughput measurements produced a single value, which made them easily comparable, the latency measurements were done for a variety of different frame sizes up to the standard Ethernet maximum frame size. These individual values then resulted in a curve (as an example, see Fig. 4b).

Then, in order to be able to rank different latency curves, weighted arithmetic means over the measured values on the curves were calculated. Since the focus of this study is the industrial environment, smaller frame sizes were weighted higher, as this reflects actual industrial traffic patterns more closely, compared to a mere average. Therefore, we favor solutions, that perform better with small payloads. The chosen frame sizes and their weights are shown in Table I. We call this final value the *weighted latency* and we used it to compare the different experiments latency-wise.

We also studied non-functional aspects of each VPN solution, that were concerned with the security of certain parameter choices, how they have to be configured and managed, as well as the overall handling of each software solution.

We conducted experiments in two steps. First, we wanted to evaluate each VPN solution separately. Some solutions allowed for choosing different algorithms for encryption and for authentication (called message authentication codes (MACs) or digests) of the payload and we wanted to find the best performing choices. Yet, testing all possible combinations of supported encryption schemes and MACs would have been prohibitively complex, so encryption algorithms were all tested with SHA-1 as MAC, while all MACs were tested with AES-128-CBC (meaning AES-128 in the CBC mode of operation), where applicable. Furthermore, baseline measurements were taken without any cryptographic protection in place in order to find the upper boundaries for the maximum achievable performance.

We chose a simple and basic setting to conduct the measurement experiments and also to test and evaluate the non-functional aspects of each VPN solution. Two nodes were connected by wire, as depicted in Fig. 1. Layer 2 traffic was generated at one node and sent over a Layer 3 connection to the other node.

We ran tests on different hardware platforms using this same test setting in order to gain insights on how the solutions behave in different environments and how they would interact with various hardware specific properties. Platforms include very resource restricted embedded platforms up to

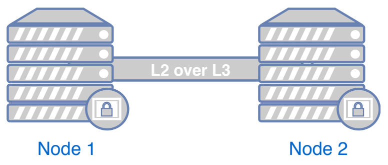


Figure 1: Basic setting consisting of two nodes.

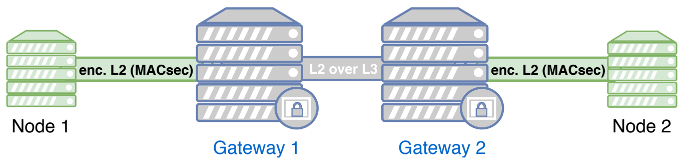


Figure 2: Extended setting consisting of two gateways plus one node each.

very powerful server machines. They are listed in Table II. No special tweaking of hardware or software was done and default settings (on operating systems and VPN software) were kept as much as possible to reach the highest degrees of comparability between the platforms. The two respective nodes consisted of identical hardware and software (versions). On the Freescale LayerScape platform not all VPN software solutions could be tested.

Only after finding the individually highest performing cipher settings for each solution on each hardware platform, could we compare them among each other. The discussion of the overall performance results independently of the underlying hardware platform as well as the non-functional aspects of the individual VPNs can be found in Section V-A. Section V-B reports on the concrete performance measurements and rankings of each VPN solution per hardware platform. Certain particularities, that are necessary to understand the results for a platform, are also examined. The overall discussion and comparative evaluation of the VPN solutions is given in Section V-C.

In a second step we wanted to leverage the potential of MACsec in order to explore future use cases with increased security demands. Since MACsec was relatively recently introduced into the Linux kernel, it will become very widespread and can be assumed to be available in many installations in the future. As the first widely available free and open source implementation of a security scheme for Layer 2 communication, it offers a previously often unfeasible approach of protecting entire Layer 2-frames, so that (within a LAN) no Layer 3 security protocol is necessary in order to protect the (local) data flows. Then, if it can be assumed, that the payload data is already protected within the LAN, the bridging of multiple LANs could be managed differently. As this might be well suited for many industrial cases, we decided to investigate.

To test different possible approaches on how to implement the protection of communication flows between nodes in separate networks beginning at Layer 2, we used the experimental setup shown in Fig. 2. It consists of two gateways which are connected by wire and have each a node connected (also by wire). A node and a gateway constitute a LAN and communicate on Layer 2, while the two gateways communicate over Layer 3 with each other. HP MicroServers (see

#	Platform	Processor	RAM	Network Interfaces	Cryptographic Hardware Acceleration	Operating System
1	HP ProDesk	Intel Core i5-4590 Quad-Core @ 3.3 GHz	16 GB	1x Gigabit Ethernet	AES	Linux 4.16.0/Debian Buster
2	Raspberry Pi 3	ARMv7 (v71) Quad-Core @ 1.2 GHz	1 GB	1x Fast Ethernet	-	Linux 4.14.32/Raspbian Stretch
3	HP ProLiant MicroServer Gen7	AMD Athlon II Neo N36L Dual-Core @ 1.3 GHz	1 GB	3x Gigabit Ethernet	-	Linux 4.16.0/Debian Buster
4	Xeon Server	Intel Xeon D-2146NT 16-Core @ 3 GHz	64 GB	4 x 10 Gigabit Ethernet	AES	Linux 4.19.4/Arch
5	Freescale LayerScape LS1020A	ARMv7 (v71) Dual-Core @ 1 GHz	1 GB	2x Gigabit Ethernet	-	Linux 4.9.98/Custom

Table II: Hardware platforms used for experiments.

Table II) were chosen as platform for the gateways and the HP ProDesks were chosen as the nodes. These platforms were chosen so that it was ensured, that the nodes could produce a MACsec-protected data stream, that would fully saturate the available bandwidth of the gateway's network interfaces. Furthermore, the HP MicroServers were close in computing capability to embedded systems and offered the necessary multiple Ethernet interfaces.

The protection of the data flows between nodes in different LANs can be organized differently, depending on the assessment of certain trade-offs. We chose three differing approaches. They and their reasoning are presented in the following:

a) MACsec over L2TP: If nodes already protect the communication payload using MACsec, the gateways would only need to relay the already secured Ethernet frames. L2TP was again chosen for that purpose. As stated previously, this is no proper solution and in this scenario even opens up new vulnerabilities. For example, the MAC addresses of involved nodes leak and reveal meta data about the communication streams and denial-of-service (DoS) attacks become possible against the gateways, as integrity of the data is only checked at the nodes. Additionally, MACsec would now have to be configured for nodes in different LANs, not just in the same LAN. As this is not a considered use case, tools for secure remote configuration of MACsec are not available (we did it by hand). Yet, this approach offers the least amount of additional computational overhead and we wanted to evaluate how a possible proper solution could perform.

b) MACsec over Wireguard: The security-related disadvantages of the previous prototypical approach can easily be ameliorated by setting up an additional secure Layer 3 tunnel between the gateways. This would protect the meta data of the individual data flows between LANs and would provide integrity-checking at the gateways. Wireguard was chosen for that purpose (the reasons will be explained in the next section). Data would now be encrypted twice, but this would easily and with conventional means protect the whole data flow in and between both LANs. Additionally, not only MACsec but also Wireguard would now have to be configured globally.

c) MACsec plus Wireguard: The most conventional solution would be to protect each LAN with MACsec individually and then establish a secure Layer 3 tunnel between the gateways. The gateways would therefore decrypt the MACsec frames, re-encrypt them using Wireguard and relay them to the other LAN. The configuration would also be conventional, meaning intra-LAN configuration of MACsec could be done locally and only the gateway tunnels would have to be configured globally.

The performance of these three approaches plus a baseline without encryption were tested and the results are presented and discussed in Section V-D

Finally, we used ping to measure the latency, while we used iperf3[6] for throughput measurements. Either dstat[7] or mpstat[8], depending on availability, were used for CPU utilization measurements. dstat was available on each platform except for Freescale LayerScape and was also used to measure the raw Ethernet throughput directly from the Ethernet devices. All tools are freely available and can be considered standard.

Each experimental run consisting of a certain VPN solution with a certain set of encryption and/or MAC algorithm on a certain hardware platform was conducted by sending 10000 ping packets for each Ethernet frame size. The weighted latency was calculated from the resulting round trip times. iperf3 was run in TCP mode for 10 seconds.

V. Results and Discussion

This section presents and discusses the results of our experimental runs. First, the results for each VPN solution are discussed individually. They apply independently of the hardware platforms, they were tested on. Then, the results on each hardware platform are discussed and performance rankings for each VPN solution are given. Next, the performance results of each VPN as well as various non-functional aspects are compared and evaluated. Finally, the results of the extended setting are shown and discussed as well.

A. VPN Solutions

This section discusses the individual results for each VPN solution. For each solution, first, non-functional aspects are analyzed and secondly, when available, the performance of different cipher options is summarized and evaluated independently of the underlying hardware platform.

1) OpenVPN: OpenVPN can be applied to a variety of use cases and is configurable via configuration files as well as parameters added to the start command. It is a very established and proven software and well documented, but still requires some knowledge to leverage it properly. It offers

[6]https://software.es.net/iperf/
[7]http://dag.wiee.rs/home-made/dstat/
[8]http://sebastien.godard.pagesperso-orange.fr/

Platform	OpenVPN	IPsec	Tinc	Freelan	SSH	MACsec	Wireguard
Raspberry Pi	61/0/61	23/32/55	27/0/27	6/0/6	16/13/29	1/0/1	1/0/1
HP ProLiant MicroServer Gen7	57/0/57	21/33/54	21/0/21	6/0/6	19/10/29	1/0/1	1/0/1
HP ProDesk	57/0/57	31/23/54	31/0/31	6/0/6	16/10/26	1/0/1	1/0/1
Freescale LayerScape LS1020A	59/0/59	—	—	6/0/6	29/0/29	1/0/1	1/0/1
Xeon Server	94/0/94	28/27/55	33/0/33	6/0/6	15/10/25	1/0/1	1/0/1

Table III: Experiments conducted per hardware platform and per VPN solution. When a solution offered to choose cipher options, multiple runs were conducted. The triples specify the numbers of working options, failed options and total of tested options.

many different ciphers for encryption and MAC generation, among those, many old, outdated and broken (e. g. DES, Blowfish, RC2). While these ciphers are clearly described as such, Blowfish, as of the current version at the time of writing this paper, is still set as default. This is only supposed to change in a future release and while for compatibility reasons it might make sense to still support broken ciphers, it is a very bad design choice to still use it as a default.

The comparative performance of the different ciphers was similar with respect to latency and throughput. The UDP mode achieved a little bit more throughput compared to TCP mode. Differences in latency were not discernible.

The biggest impact to performance was contributed by the mode of operation, and not so much by the actual encryption algorithm. CFB1 and CFB8 showed (on all platforms) abysmal behavior and should not be used. Other modes had no discernible impact. The best performing encryption scheme was AES. After that came Camellia and SEED.

The most efficient MACs were MD5 and SHA-1. MD5 is also old and considered broken, and should therefore only be used against unintentional corruption. Since SHA-1 also shows signs of age (as collisions have been found, for details refer to [18]), it would be more wise to use SHA-2. It only incurs a minor additional performance penalty. Certain MACs (Whirlpool, BLAKE2, MDC-2) performed very bad on some or all of the platforms and performance of the MACs generally varied greatly. Latency was less impacted by the choice of MAC compared to throughput.

2) IPsec: IPsec is also a proven standard VPN solution and as such well documented. Yet, is it more complicated to configure (via configuration files) compared to OpenVPN and needs more effort and expert knowledge. The strongSwan IPsec suite offers old and known to be broken ciphers for compatibility reasons, but describes them as such. The default encryption scheme is AES-128 and the default MAC is SHA-2. Both choices are considered secure.

IPsec runs in the Linux kernel and encrypts IP packets on the fly, if their destination address was previously configured. It does not create a special virtual device (compared to all the other solutions), which can be used for routing. This makes it hard to detect, whether outgoing packets are actually protected. If wrongly configured, IPsec can fail silently, meaning the connectivity remains, but packets are sent in plaintext. The kernel integration makes IPsec very fast, but also more complex to use, as IPsec needs to be monitored constantly. Furthermore, in case of failure, in most cases the applications and services using this channel would probably not be informed about the lack of protection.

Furthermore, not all available cipher options actually worked (for comparison among solutions see Table III). Encryption algorithms could be chosen with or without a mode of operation. Choosing an algorithm without mode always worked. AES with a mode selected, worked sometimes on some platforms. Camellia with a chosen mode never worked (without one, it did).

Two MACs always worked (AES-XCBC, SHA). MD5 and AES-CMAC worked sometimes on some platforms and all different variants of AES-GMAC never worked.

Camellia generally showed best performance for throughput, yet worst for latency (while worst means 15% higher compared to respective best). AES performed slightly worse in throughput but showed less increase in latency. Best MACs were MD5 and SHA-1. SHA-2 was only marginally worse and, as previously explained, should therefore be preferred.

3) Tinc: Tinc is a less common VPN solution with a smaller user base. It is configurable via configuration files and is slightly more complex to use compared to OpenVPN (as it offers more functionality). The documentation states, that all ciphers from LibreSSL or OpenSSL in CBC mode are supported. And while this is true, when inquiring the options (via `openssl list -cipher-algorithms`), OpenSSL only lists all possibilities and does not offer an assessment about the security of these algorithms. While old and outdated ciphers are available, Tinc defaults to up-to-date AES-256 and SHA-256.

ChaCha20/Poly1305 showed best performance. On the platforms where it was not available, AES was best. After that came Camellia and SEED. If hardware acceleration was available, AES always performed best.

SHA-2 was on the same level as SHA-1 and MD5 and should therefore be preferred as MAC. Whirlpool showed the worst performance by a big margin and BLAKE2 was either among the best or among the worst, depending on whether there was a CPU bottleneck.

Tinc also showed some strange behavior. On the Xeon platform, the Shake-128 MAC showed the best throughput performance with a big margin and at the same time a very bad latency. ChaCha20 (without Poly1305) showed abysmal performance.

4) Freelan: Freelan is a less common VPN solution that offers much more functionality and serves different use cases compared to the standards OpenVPN and IPsec. It is therefore also more complex to configure, which is done also via configuration files.

It does not offer outdated or broken ciphers and defaults to the strongest available one. It uses elliptic-curve cryptography and users may choose different curves and whether AES should be run with a key size of 128 or 256 bits.

All options were tested and differences in performance were minimal. So the strongest option should be considered.

Yet, compared to the other discussed solutions, Freelan performed very poorly and should hence only be used in use cases, the other solutions cannot accommodate.

5) SSH VPN: Secure Shell is a standard tool, yet probably not for the use case of network bridging. Therefore, configuration tends to be less intuitive and is done via parameters at startup.

It offers outdated and broken ciphers without warning or discussion. It defaults to ChaCha20/Poly1305 and other strong and up-to-date ciphers.

Not all offered cipher options actually worked (see Table III). Independently of the platform, AES-CTR, AES-GCM and ChaCha20/Poly1305 always worked and encryption in CBC mode always failed. Two MACs (UMAC, HMAC-SHA-2) always worked, while HMAC-SHA-1 worked on most platforms. Other MACs only worked on a single platform or never.

Performance results were very volatile, probably owing to the fact, that Secure Shell operates in user space and VPN is not its intended use case. The most efficient encryption schemes were AES and ChaCha20/Poly1305. The results for the MACs showed no clear picture. They were pretty random and characteristics like tag size, ETM (encrypt-then-mac) or not, UMAC or HMAC did not help to differentiate or group the results. Yet, MACs had major influence on the performance (on some platforms). Sometimes choosing ETM increased latency considerably.

6) MACsec: MACsec running inside the Linux kernel, can be statically configured via the iproute tool set. Since being new, more comfortable options are not yet available. In stark contrast to the previous solutions, the only cipher it offers, is AES-128-GCM. MACsec does authenticated encryption and hence no extra MAC or digest needs to be specified. This cipher is considered up-to-date and no misconfiguration in this respect can happen.

7) Wireguard: Usability being a design goal of Wireguard, configuration was the easiest and least complex. It does much of the configuration automatically and does not need to be constantly monitored (in contrast to IPsec). Like MACsec, it does not offer different cipher choices and uses up-to-date ChaCha20/Poly1305.

B. Hardware Platforms

This section presents the performance results for the VPN solutions on each hardware platform. The individually highest performing cipher options were chosen to represent each solution in a performance ranking for each platform. Absolute rankings as well as relative distances of the VPN solutions compared to the baseline can be derived from Fig. 3. Certain particularities of the hardware platforms are discussed as they are necessary to understand the results.

1) HP ProDesk: On this platform, the CPU was powerful enough, so that nearly all solutions (except Freelan) achieved line speed. The small differences in achieved throughput stem most probably from the individually different protocol overheads. Also the selection of ciphers (where applicable) did have less to no impact on the results. AES was generally the most efficient choice for cipher, due to the CPU providing AES hardware acceleration. On OpenVPN, some of the cipher options behaved very badly (apart from the previously discussed CFB1 and CFB8 modes of operation). MACsec again showed best latency performance. On this platform, no solution outperformed the others. All, except for Freelan, can be recommended. If low latency is necessary, then MACsec or IPsec should be preferred.

2) Raspberry Pi: The results are generally dominated by the less powerful CPU of the Raspbery Pi and the limited Ethernet device, which is connected to the system over the USB interface. While the system achieves line speed when only sending and receiving data, the speed is considerably reduced, when data is also protected. Network flows had high volatility, suffered frequent random outliers and showed erratic behavior. Furthermore, only Freelan used more than one CPU core. MACsec (uncharacteristically) performed worst for throughput, yet this may be explained by the way, the network interface is attached. Overall Wireguard showed the best performance, achieving the highest throughput and good latency.

3) HP ProLiant MicroServer Gen7: In the absence of hardware-based acceleration for AES within the CPU, ChaCha20/Poly1305 showed the best performance for multiple solutions. Wireguard, using the same cipher, showed a vastly better throughput performance, compared to all other solutions, while also utilizing the CPU best. MACsec showed the lowest latency. Overall Wireguard should be preferred, as it is the only solution which achieves almost line speed while showing acceptable levels of latency.

4) Xeon Server: The baseline performance measurement shows that this system can achieve line speed of 10 Gbit/s. But, it uses frame sizes of up to 64k between the nodes. This is no standard behavior and only works when traffic directly flows between the physical Ethernet devices. It does not work over tunnels and probably not over ordinary networking hardware (switches, routers), that would be expected in a real industrial setting. Therefore, already only using an L2TP tunnel (without additional encryption scheme) reduces throughput to 4.2 Gbit/s. Generally, there were big differences between different ciphers within the same solutions, yet the bottleneck on this platform clearly was not the CPU. Hence, the available AES hardware acceleration did not have an impact on the results. It must rather be attributed to something else, and we suspect the memory interface. Accordingly, the best performing solutions for both throughput and latency were MACsec and IPsec, probably because they run in kernel space. Wireguard, which runs also in the kernel (yet is still a prototype) was close in performance, while the solutions running in user space performed very bad.

5) Freescale LayerScape LS1020A: CPU performance showed to be the main limiting factor on this platform and while line speed was achievable for unprotected traffic, encryption reduced the performance considerably. This platform is an embedded platform and was available as a demo board and did not run an of-the-shelf Linux operating system. Therefore, not all solutions could be tested. Again Wireguard showed the best throughput performance, while MACsec showed the lowest latency. Overall Wireguard should be preferred, as it achieves considerably more throughput than the other solutions, while recording second best latency in this case.

C. Comparison of VPN Solutions

This section compares and evaluates the performance results as well as important non-functional aspects that distinguish the studied VPN solutions.

1) Performance: Fig. 3 summarizes the individual performance rankings for each platform. Since the platforms contained Ethernet devices of different speeds, the measurements were normalized to the same order of magnitude, in order to make their relative distance from the baseline (the theoretically possible) comparable. The evaluation showed a clear trend towards the latest approaches MACsec and Wireguard. While MACsec (together with IPsec) was consistently best or second best performing solution for latency, Wireguard showed the highest throughput achievable (or line speed) on 4 of 5 platforms.

For 10 Gbit/s links, the equations seem to change considerably. In order to saturate these links, hardware support for encryption and big CPUs are not sufficient anymore and the bottleneck moved somewhere else. Where to, we can only speculate.

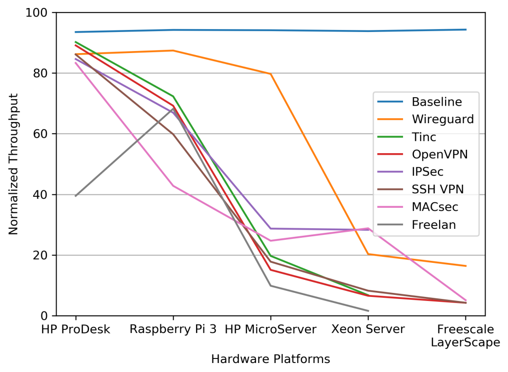
2) Non-functional Aspects: The customary and established solutions (OpenVPN, IPsec, Tinc, Secure Shell) offer a multitude of ciphers to choose from. And, while variety is ostensibly a good feature, it has detrimental effects as well.

Some solutions offer ciphers in their documentation, but once configured just do not work (see Table III) and furthermore, the sets of working algorithms change between platforms. We could not find conclusive evidence as to why this is the case. It is at least puzzling, as all platforms ran an up-to-date Linux kernel, with, in most cases, a current software distribution on top (see Table II). Some ciphers even worked on none of the tested platforms. Within the ciphers that did work, some individual ciphers (e. g. Whirlpool, MCDC-2) always showed abysmal performance. The modes of operation CFB1 and CFB8 also performed very badly, independently of the configured cipher. Other ciphers showed very good and very poor performance depending on the platform (e. g. BLAKE2). Furthermore, some ciphers are so old, that they have been broken by now, and must be considered insecure. Blowfish was proposed in 1993 and is even still the default setting for OpenVPN. Legacy support cannot be used as an argument here. Performance of the ciphers between platforms also differs widely. If performance actually is an issue, tweaking of the individual system becomes necessary and as we have showed, this is a non-trivial task.

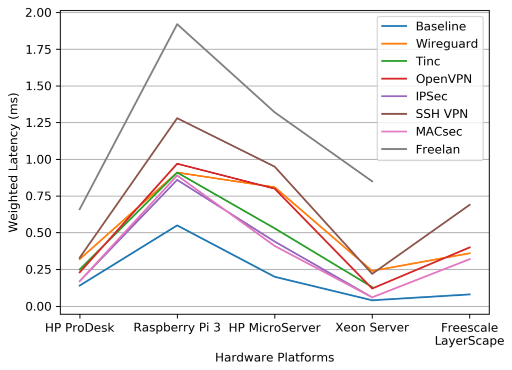
This wealth of options, that probably accumulated over many years of development and maintenance of each VPN software, seems to make it hard to manage it. In our minds, users would be better served, if the configurable cipher sets would be drastically reduced.

On contrast, the new approaches MACsec and especially Wireguard go in the opposite direction and do not offer the user multiple ciphers, thereby eliminating the chance for misconfiguration. Additionally, this gives the software developers the chance to address performance and compatibility issues, that may arise on different hardware and operating system architectures. Therefore, we clearly recommend the use of those two solutions, wherever possible.

[9]Normalization factors for the platforms were 0.1, 1, 0.1, 0.01, 0.1



(a) Throughput ranking.[9]



(b) Latency ranking.

Figure 3: General performance rankings of solutions over all platforms.

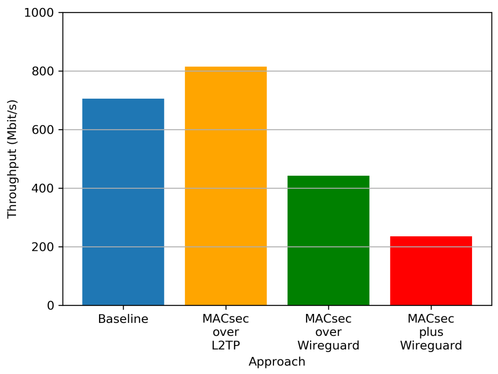
D. Extended Setting

Fig. 4 shows the achieved throughput and latency performances of each approach.

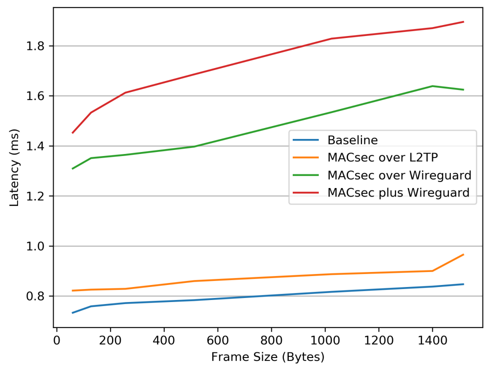
The throughput of the baseline measurement is lower than the measurement for MACsec/L2TP. This is probably due to the measurement tools rate adjustment algorithm getting confused by the setup, meaning the data flows being interrupted by multiple send and receive queues of the different involved devices. The measured CPU usage does not indicate a bottleneck.

The performance of the 'MACsec over L2TP'-approach shows almost line speed. This is no surprise, as the gateways only relay already encrypted frames. Yet, with the other two approaches, the performance drops considerably. The additional encryption steps performed on the gateways, have big impact. The additional Wireguard tunnel within the second approach halves achieved throughput and almost doubles latency. The further step of the third approach of de- and encrypting the MACsec frames on the gateways halves the achievable throughput yet again.

For resource restricted environments, where performance is non the less an issue, it seems unfeasible to protect communication data within and in between LANs with conventional means (second and third approach). Therefore, the aforementioned trade-off between configuration complexity and performance should be answered individually depending on



(a) Throughput comparison.



(b) Latency comparison.

Figure 4: Performance comparison of different approaches of the extended gateway setting.

the use case. The first prototypical approach of just relaying MACsec frames should be investigated further.

VI. Conclusion

This study investigated different software solutions on how to securely interconnect local area networks. Non-functional aspects as well as their performance were analyzed, discussed and compared.

The classic and well established solutions, like OpenVPN and IPsec, were found to exhibit significant drawbacks in the face of new and upcoming solutions. We believe, that these, namely MACsec and Wireguard, should be preferred in the future, where and whenever possible.

This study also revealed starting points for future research. ChaCha/Poly1305 performed best in resource restricted environments, where AES hardware acceleration within the CPU did not exist. It might therefore be promising to include this cipher into other VPN solutions and protocols in order to increase their performance in certain use cases. Furthermore, extended security schemes, that already protect communication data within a LAN should be further researched in order to be used efficiently.

References

[1] K. Ahmed, N. S. Nafi, J. O. Blech, M. A. Gregory, and H. Schmidt. Software defined industry automation networks. In *2017 27th International Telecommunication Networks and Applications Conference (ITNAC)*, pages 1–3, Nov 2017.

[2] A. Alshalan, S. Pisharody, and D. Huang. A survey of mobile vpn technologies. *IEEE Communications Surveys Tutorials*, 18(2):1177–1196, Secondquarter 2016.

[3] A. Bluschke, W. Bueschel, M. Hohmuth, F. Jehring, R. Kaminski, K. Klamka, S. Koepsell, A. Lackorzynski, T. Lackorzynski, M. Matthews, P. Rietzsch, A. Senier, P. Sieber, V. Ulrich, R. Wiggers, and J. Wolter. fastvpn - secure and flexible networking for industry 4.0. In *Broadband Coverage in Germany; 12th ITG-Symposium*, pages 1–8, April 2018.

[4] I. Coonjah, P. C. Catherine, and K. M. S. Soyjaudah. Performance evaluation and analysis of layer 3 tunneling between openssh and openvpn in a wide area network environment. In *2015 International Conference on Computing, Communication and Security (ICCCS)*, pages 1–4, Dec 2015.

[5] B. Czybik, S. Hausmann, S. Heiss, and J. Jasperneite. Performance evaluation of mac algorithms for real-time ethernet communication systems. pages 676–681, 07 2013.

[6] J. A. Donenfeld. WireGuard: Next Generation Kernel Network Tunnel. Technical report, www.wireguard.com, 2018.

[7] S. Dubroca. MACsec: Encryption for the wired LAN. In *netdev 1.1*. Red Hat, February 2016.

[8] S. Escher and S. Köpsell. Durchführung eines integrierten Anti-Phishing-Trainings. In *Christian Paulsen (Hg.): Sicherheit in vernetzten Systemen. 23. DFN-Konferenz*, 2018.

[9] A. Greenberg. The Untold Story of NotPetya, the Most Devastating Cyberattack in History. Wired.com, September 2018.

[10] Industrial Internet Consortium. *Industrial Networking Enabling IIoT Communication*, August 2018.

[11] S. Khanvilkar and A. Khokhar. Virtual private networks: an overview with performance evaluation. *IEEE Communications Magazine*, 42(10):146–154, Oct 2004.

[12] I. Kotuliak, P. Rybar, and P. Trúchly. Performance comparison of ipsec and tls based vpn technologies. 10 2011.

[13] A. Lakbabi, G. Orhanou, and S. E. Hajji. Vpn ipsec amp; ssl technology security and management point of view. In *2012 Next Generation Networks and Services (NGNS)*, pages 202–208, Dec 2012.

[14] J. Lau and M. Townsley. Layer Two Tunneling Protocol - Version 3 (L2TPv3). RFC 3931, March 2005.

[15] H. Mao, L. Zhu, and H. Qin. A comparative research on ssl vpn and ipsec vpn. In *2012 8th International Conference on Wireless Communications, Networking and Mobile Computing*, pages 1–4, Sep. 2012.

[16] S. Narayan, K. Brooking, and S. de Vere. Network performance analysis of vpn protocols: An empirical comparison on different operating systems. In *2009 International Conference on Networks Security, Wireless Communications and Trusted Computing*, volume 1, pages 645–648, April 2009.

[17] D. Quarta, M. Pogliani, M. Polino, F. Maggi, A. M. Zanchettin, and S. Zanero. An experimental security analysis of an industrial robot controller. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 268–286, May 2017.

[18] M. Stevens, E. Bursztein, P. Karpman, A. Albertini, and Y. Markov. The first collision for full sha-1. pages 570–596, 07 2017.

[19] A. V. Uskov. Information security of ipsec-based mobile vpn: Authentication and encryption algorithms performance. In *2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications*, pages 1042–1048, June 2012.

[20] M. H. M. Zaharuddin, R. A. Rahman, and M. Kassim. Technical comparison analysis of encryption algorithm on site-to-site ipsec vpn. In *2010 International Conference on Computer Applications and Industrial Electronics*, pages 641–645, Dec 2010.

[21] Z. Zhipeng, S. Chandel, S. Jingyao, Y. Shilin, Y. Yunnan, and Z. Jingji. Vpn: a boon or trap? : A comparative study of mpls, ipsec, and ssl virtual private networks. In *2018 Second International Conference on Computing Methodologies and Communication (ICCMC)*, pages 510–515, Feb 2018.

[22] ZVEI. *The Reference Architectural Model Industrie 4.0 (RAMI 4.0)*, April 2015.