A Comparative Research on VPN Technologies on Operating System for Routers

Phu Nguyen Phan Hai, Hoa Nguyen Hong, Bao Bui Quoc, Trang Hoang
Department of Electronics, Faculty of Electrical and Electronics Engineering
Ho Chi Minh City University of Technology (HCMUT)
268 Ly Thuong Kiet Street, District 10, Ho Chi Minh City, Vietnam
{haiphu, hoa.nguyen.raven7, buiquocbao, hoangtrang}@hcmut.edu.vn

Abstract— **With the development of information technology, VPN technology has been widely applied in many fields because it can be set up at a lower cost when compared to other security technologies. For this reason, VPN technology has been integrated on most platforms, such as computers, mobile devices, or routers. In VNP technologies, SSL-VPN, IPsec, Wireguard are considered as the three most popular ones today. This paper focuses on evaluating the performances of these VPN technologies integrated into the operating system for routers. After examining the throughput of these VPN technologies, as a simple and efficient technology, Wireguard is suitable for operating systems in routers. To do experiment, our routers named as BK-Router were designed and built by ourselves in this work.**

I. INTRODUCTION

VPN (Virtual Private Network) is a technology that can help us create a secure network connection while using the public network. VPN can help users stay invisible on the internet by encrypting all user information when transmitting data. Moreover, VPN does not require users to build a new system. With the existing network infrastructure, users can create their VPN network easily [1] [2]. With these outstanding advantages, VPN has become more familiar with most network models that require high security as well as comprehensive connectivity. Thus, to meet the high demand of users, many organizations are implementing VPN technologies with many different protocols as well as many different encryptions and authentication algorithms. In particular, among those technologies, there are three popular VPN technologies today: SSL, IPsec and Wireguard, which provide users with high security, ease of use, a variety of encryption and authentication algorithms, etc [3].

Besides, Router is also an indispensable device in any network model. With any modern router, it not only provides routing features for the network but also provides users with many different convenient services, especially the ability to provide VPN. However, the more features are integrated into one device, the more expensive the router will be. Therefore, the lightweight operating system for routers was born to solve the problem of the device cost as mentioned above [4]. It provides users with many convenient services, including VPN technologies, while its minimal hardware requirements are affordable to the majority of users.

Generally, VPN technology and a lightweight router operating system have helped users save a lot of initial and operating costs. But as mentioned before, among three outstanding VPN technologies including SSL, IPsec, and WireGuard that are integrated into operating systems for routers, which is the best VPN technology for operating systems for routers, especially in the data transmission phase?

Consequently, to address this question, this paper evaluates three VPN technologies: SSL-VPN, IPsec, and WireGuard and compares them from several aspects such as VPN tunneling performance and throughput. These results in our paper could help designers choose which VPN technology is suitable for their applications.

The rest of the paper is structured as follows: Section II is a brief characterization of SSL-VPN, IPsec, and WireGuard protocols. Section III describes the test environment and the developed application to run the tests. The results and discussion are given in section IV, and the final section is the conclusion of this work.

II. BRIEF DESCRIPTION OF SSL-VPN, IPSEC, AND WIREGUARD

A. SSL-VPN

OpenVPN is an SSL-VPN application that is deployed on Layer 2 and Layer 3 of the OSI model. Several SSL implementations have been created over the years, but OpenVPN is still the most mentioned name when discussing SSL-VPN by supporting many outstanding features such as security and easy connection.

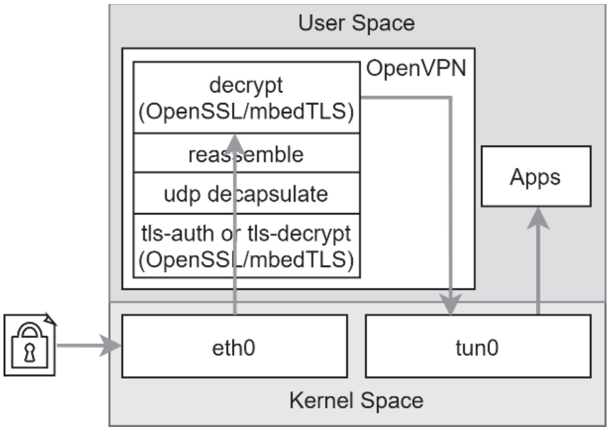
User Space
OpenVPN
decrypt (OpenSSL/mbedTLS)
reassemble
udp decapsulate
tls-auth or tls-decrypt (OpenSSL/mbedTLS)
Apps
eth0
tun0
Kernel Space

Figure 1. Inbound flow of OpenVPN packet

Moreover, OpenVPN provides users with a wide choice of encryption algorithms (AES-GCM, AES-CBC, CHACHA20, ...) as well as authentication algorithms (MD5, SHA, ...) with the help of OpenSSL.

In the process of receiving an OpenVPN packet, first, like any regular packet, they must first go through a physical interface. Then the packet will be processed by OpenVPN and finally decrypt a packet before transferring the packet to the TUN interface. Inbound flow of OpenVPN packet is shown as Figure 1 [5]

OpenVPN uses OpenSSL to perform packet encryption and authentication. OpenSSL is an all-around cryptography library that offers open-source applications of the TLS protocol, but it operates in user-space. Therefore, most of the time OpenVPN's packet processing is in user-space. It makes processing speed slows down since tasks have to jump between kernel-space and user-space constantly.

The next section will introduce IPsec protocol, which uses Crypto API to encrypt packets.

B. IPSec

Unlike SSL-VPN, which is implemented mainly in user-space, IPsec is considered as part of the Linux kernel, so it is implemented entirely in kernel space. Over the years, many applications have implemented IPsec, but the most popular seems to be StrongSwan. So in this paper, we take StrongSwan to represent IPSec protocol. In Figure 2, we want to show how Strongswan implements IPSec to encrypt and authenticate a packet.

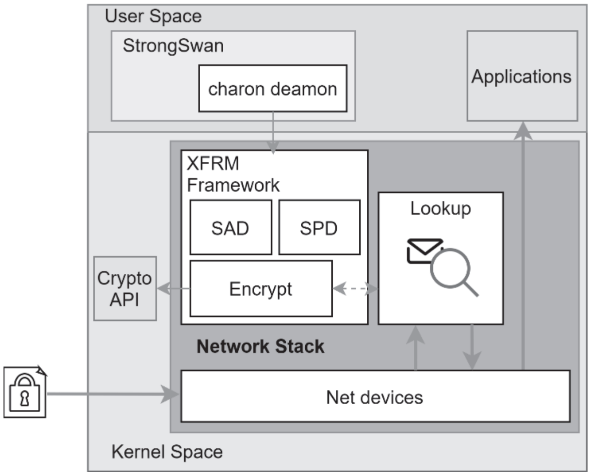

Figure 2. Inbound flow of OpenVPN packet

In Linux, the standard solution for encrypted tunnels is IPsec, using the Linux transform layer ("xfrm") [4]. The user fills in the kernel structure to define the cipher suite and key, or other transformations such as compression, to use for which packet selector passes through the subsystem. Strongswan uses the user-space Charon daemon to be responsible for updating data structures (Security Association Database and Security Policy Database) based on the results of the key exchange performed by IKEv2. Then when the tunnel has been established. Packets are encrypted and encapsulated through the XFRM Framework without the need for StrongSwan anymore. However, one thing to note is that IPSec does not do the encryption and decryption; it calls the functions in the Crypto API to perform encryption and authentication, as shown in Figure 2 [6].

So we can understand in a more straightforward way that StrongSwan is just an application to exchange keys to establish the initial VPN tunnel through IKE protocol and manage users' VPN connection information. And after the tunnel has been established, IPSec will be responsible for decoding/encrypting and encapsulating packets.

C. WireGuard

With IPsec, Users fill in a kernel structure by using a daemon in user-space to update these data structures based on the results of a key exchange, generally done with IKEv2. IPSec itself is a complicated protocol. The complexity, as well as the amount of code, of this solution, is considerable. IPSec separates the key exchange layer from the encryption part, which can be a wise separation from a semantic point of view. Furthermore, the same while separating the transformation layer from the interface layer is correct from a network viewpoint. However, this layered approach increases the complexity of the protocol. While WireGuard does away with these layering separations. Instead of the complexity of IPsec. WireGuard simply gives a virtual interface. After configuring the interface with a private key, the tunnel simply works. On the other hand, OpenVPN is a TUN/TAP user-space based on solution using TLS. By virtue of it being in user-space as we already mentioned, it has very poor performance since packets must be copied multiple times between kernel space and user-space, and a long-lived daemon is required.

In short, WireGuard focuses on simplicity and an auditable codebase while remaining extremely fast and suitable for a modicum of environments. By combining the key exchange and layer three transport encryption into one mechanism and using a virtual interface rather than a transform layer [7]. However, WireGuard does not do well compared to the two competitors because it supports very few encryptions and authentication algorithms. WireGuard only uses ChaCha20 [8] and Poly1305 [9] for authenticated encryption.

III. TESTBED SETUP

A. Testbed setup

Usually, there are two common VPN models: Site-to-Site model or Client-to-Site model. Companies often use Site-to-Site VPNs to connect their corporate networks and remote branch offices. This approach works when a company has in-house data centers, susceptible applications, or requires minimal bandwidth. However, most companies have moved their applications and data to the cloud, and their employees mostly work on mobile devices. It no longer makes sense for employees to go through a data center to get to the cloud. Instead, they can go to the cloud directly.

Consequently, companies need to establish a network topology with access to cloud or data center applications by applying Client-to-Site model. VPN Client-to-Site is a type of VPN that allows a user to connect to a remote private network through a VPN server. Typically, to be able to use VPN client to site, the user's computer or any mobile device will have to install a VPN client software to be able to connect to the VPN server, which can be a server or a router that supports VPN. So, in this experimental model, we use a computer acting as a client connected to a VPN server which is a router (the specifications of all the devices will be listed in the next section)

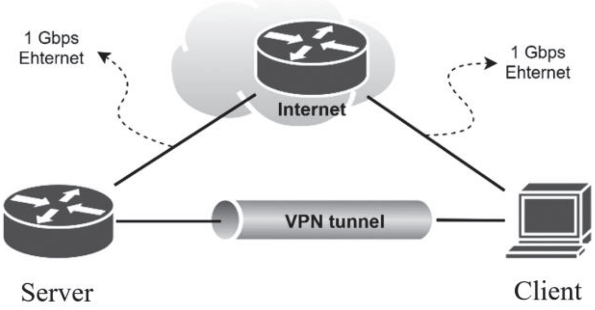

Figure 3. Experimental testbed

B. *Device specifications*

In order to make a VPN performance comparison, we designed and built a real-world network based on a Client-to-Site model shown in Figure 3. Usually, in the real-world network, there are multiple VPN clients connected to a VPN server. However, this is not necessary in our case because the VPN tunnel throughput is equal to the total throughput of the connections. Therefore, topology with one client to a server or many clients to a server, the total throughput is the same. In our experimental model, we use two devices with specifications described as follows:

- One computer with the following hardware specifications: Intel Core i5 with four cores running at 3.0 GHz (3.5GHz Turbo Frequency). 16 GB of 2400MHz DDR4 RAM and an RJ45 port 10/100/1000 Gigabit. Moreover, the computer is running Ubuntu 20.04.
- One our router uses Marvell's SOM Armada 388 running at 1.3GHz, 1GB RAM, and 5 RJ45 ports 10/100/1000 Gigabit. This device is self-developed that it can be easily controlled and operated and a deeper understanding of the working mechanism of VPN technologies. (Figure 4)

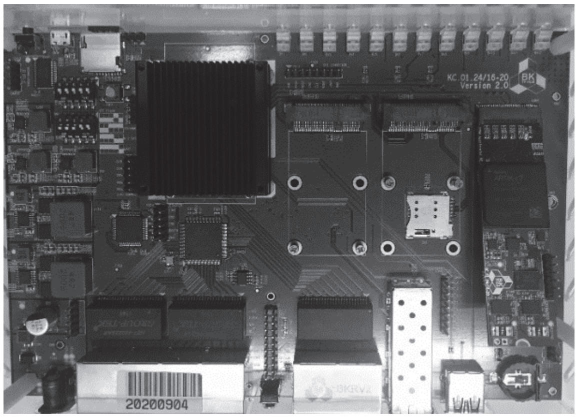

Figure 4. Our router, as a VPN server, is built to be used in this paper.

In addition, the performance comparison was made with the help of a program called iPerf3, which was selected due to its consistent performance, ease of use, and ability to produce the metrics required. Both client and server have iPerf3 installed; the client is a sender, the server is a receiver, and traffic generated by iPerf3 goes from client to server.

IV. EXPERIMENTAL RESULTS AND DISCUSSION

This section presents measured VPN throughput across different encryption algorithms when applied to packet encryption as in section A and VPN throughput across three types of VPN technologies as in section B.

A. *Comparison of encryption algorithm*

For every VPN connection, security is a prerequisite that any VPN technology must-have. There are many ways to evaluate a VPN connection as secure such as security during key exchange or security during packet transmission. But in general, packet encryption is required for both of these processes. Thus, the packet is not eavesdropped by the attacker. As a result, many encryption algorithms have been created to meet different criteria for VPN technology. However, evaluating all cryptographic algorithms is a big challenge. Therefore, this paper only selects some popular encryption algorithms such as aes-cbc, aes-gcm, and Chacha20 to evaluate the performance on each VPN technology.

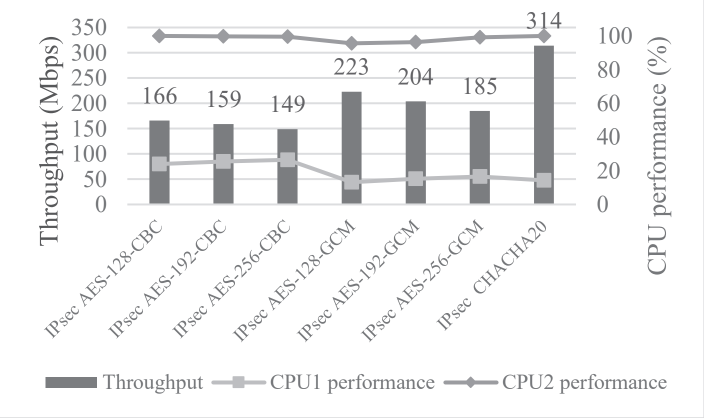

Figure 5. Average throughput of the VPN algorithms based on IPsec VPN

The average throughput after the VPN tunnel was established based on IPsec technology is shown in Figure 5. When analyzing CPU consumption, we verified that the Chacha20 algorithm achieves the highest throughput than the AES algorithm while using the same CPU consumption. The reason is ChaCha20 is based on ARX (Addition-Rotation-XOR) operations [9] [10], which are CPU-friendly instructions. In comparison, AES uses S-box and Mixcolumns computations, which are generally implemented as a look-up table.

In addition, Figure 5 also shows the effect of key size on the AES encryption algorithm. The larger the key length value, the lower the VPN throughput. In other words, the larger the key length, the higher the complexity of the encryption algorithm, so the CPU needs more time and resources to encrypt a packet [11]. However, in the security field, the higher the complexity of the algorithm, the more difficult it to crack the algorithm. Therefore, a tradeoff in terms of throughput to increase security is worthy.

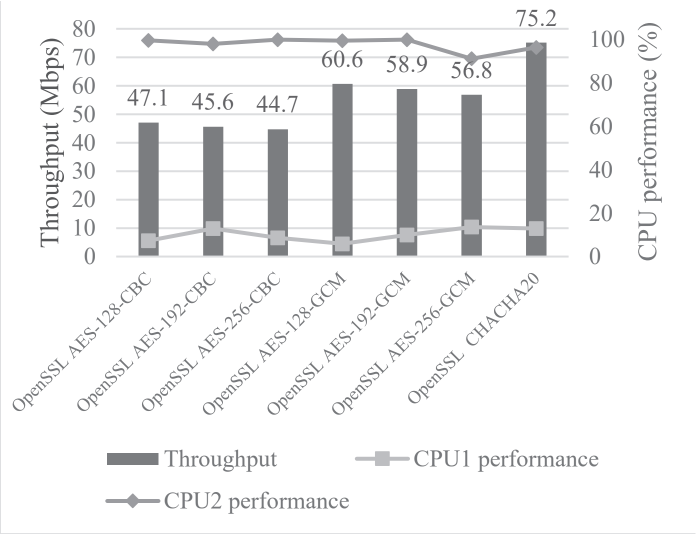

Figure 6. Average throughput of the VPN algorithms based on SSL-VPN

The throughput of the VPN algorithms based on SSL-VPN is shown in Figure 6. It can be seen that the trend of increasing throughput of cryptography algorithms is the same when compared with IPsec technology. However, the difference in average throughput between the algorithms is not significant. The reason is that the duration of copying packets between kernel-space and user-space is significant when compared to the duration that SSL technology spends on encryption. So the influence of the speed of the encryption algorithms no longer has many effects on throughput. Besides, this leads to the time it takes to encrypt a packet increase, which results in lower throughput based on SSL technology compared to IPsec technology.

B. *Comparison of VPN technologies*

Section A showed that with both SSL and IPSec technologies, the throughput result of ChaCha20 algorithm is the highest. So in this section, we choose ChaCha20 algorithm as a representative to compare all three VPN technologies: SSL, IPsec, and WireGuard. Partly because WireGuard only supports ChaCha20 algorithm.

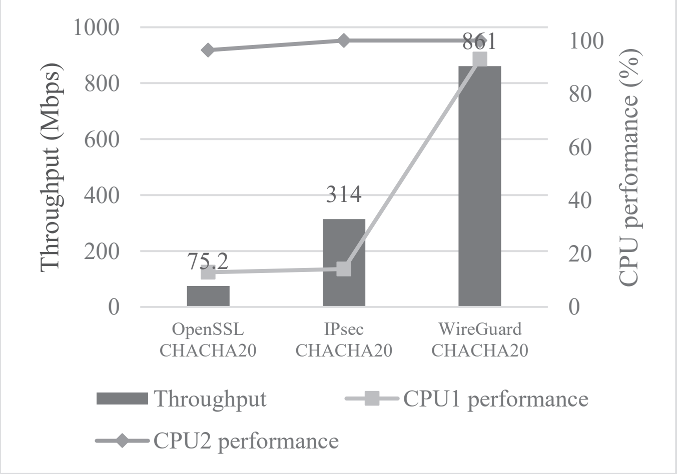

Figure 7. Average throughput of three VPN technologies

Figure 7 shows that the VPN throughput of the WireGuard technology is as large as the approximation of the maximum throughput that the hardware allows. In contrast, OpenSSL is performing quite poorly when its throughput is only 10% of WireGuard. OpenSSL throughput becomes worse than the other two technologies because it operates on user-space, which means there is added latency and overhead of the scheduler and copying packets between user space and kernel space several times.

While IPsec also operates in kernel space, but it loses significantly in performance when compared to WireGuard. The reason is due to the mechanism of IPsec technology. IPsec is a complex technology to access and understand. Layering can be semantically as well as networking, but this makes IPsec complicated. WireGuard, in contrast, starts from the basis of flawed layering violations that made it faster than other VPN technologies.

In brief, WireGuard, a technology with less than 4000 lines of code [7], demonstrates that it is effective. Its simplicity and non-layering mechanism allow it to achieve throughput higher than the other two VPN technologies efficiently. In addition, completely embedding the program in the kernel makes it faster. The time wasted copying a packet between the kernel-space, and the user-space is eliminated (Figure 8).

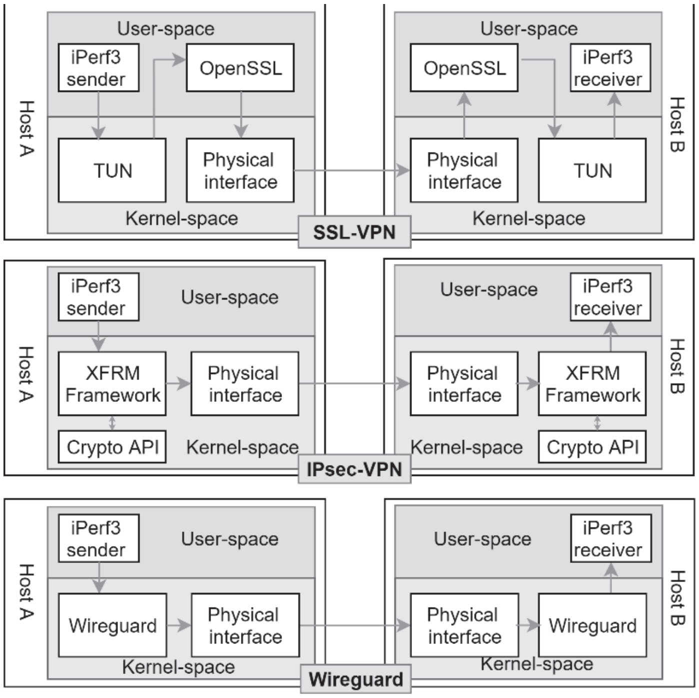

Figure 8. The flow of a packet on three VPN technologies

Also, when we measure the CPU performance of the VPN throughput measurement, the CPU consumption used by WireGuard technology is the maximum performance on both cores of the Router's SOM. While, with both IPsec and OpenVPN technologies, it seems that they can only run on one core of the SOM. This shows that WireGuard has effectively used the device's resources to optimize its performance using a multi-core mechanism for VPN implementation.

V. CONCLUSION

This paper evaluates three VPN technologies: SSL-VPN, IPsec, and WireGuard and compares them in terms of VPN tunneling performance and throughput. From these results, WireGuard has probably outperformed than other two VPN technologies. However, it is not true to say that IPsec and SSL work inefficiently. From other perspectives such as algorithm support, they actually do much better than WireGuard. Furthermore, they are also the basis for creating WireGuard technology development. To sum up, after methodical evaluation of the three VPN technologies, WireGuard demonstrated that the simplicity and the breaking of rules in the network layer significantly affect the VPN's performance. The results in our paper could help designers choose which VPN technology is suitable for their applications.

ACKNOWLEDGEMENTS

We acknowledge the support of time and facilities from Ho Chi Minh City University of Technology (HCMUT), VNU-HCM for this study.

REFERENCES

[1] S. Sridevi and M. D. H, "Technical Overview of Virtual Private Networks(VPNs)", *International Journal of Scientific Research,* vol. II, no. 7, pp. 93-96, 2012.

[2] Q. Jing, A. Vasilakos, J. Wan, J. Lu and D. Qiu, "Security of the Internet of Things: Perspectives and challenges", *Wireless Networks,* vol. 20, p. 2481–2501, 2018.

[3] M. X. Zheng Wu, "Performance Evaluation of VPN with Different Network Topologies", *International Conference on Electronics Technology,* pp. 51-55, 10 May 2019.

[4] P. Weidenbach and J. v. Dorp, "Home Router Security Report 2020", Fraunhofer, München, June 2020.

[5] Pippin, "How Packets Flow", OpenVPN, 17 December 2019. [Online]. Available: https://community.openvpn.net/openvpn/wiki/HowPacketsFlow.

[6] H. Dhall, D. Dhall, S. Batra and P. Rani, "Implementation of IPSec Protocol", *Advanced Computing & Communication Technologies*, Rohtak, 2012.

[7] J. A. Donenfeld, "WireGuard: Next Generation Kernel Network Tunnel", *Network and Distributed System Security Symposium*, California, 2017.

[8] D. J. Bernstein, "ChaCha, a variant of Salsa20", 28 January 2008. [Online]. Available: https://cr.yp.to/chacha/chacha-20080128.pdf.

[9] D. J. Bernstein, "The Poly1305-AES Message-Authentication Code", 29 March 2005. [Online]. Available: https://cr.yp.to/mac/poly1305-20050329.pdf.

[10] R. Andriani, S. E. Wijayanti and F. W. Wibowo, "Comparision Of AES 128, 192 And 256 Bit", *Information Technology, Information Systems and Electrical Engineering,* pp. 120-124, 13 November 2018.

[11] D. A. F. Saraiva, V. R. Q. Leithardt, D. d. Paula, A. S. Mendes, G. V. González and P. Crocker, "PRISEC: Comparison of Symmetric Key Algorithms for IoT Devices ," *Sensors,* vol. 19, no. 19, pp. 4312 - 4312, 2019.