Computers and Electrical Engineering 71 (2018) 752–767

Contents lists available at ScienceDirect

Computers and Electrical Engineering

journal homepage: www.elsevier.com/locate/compeleceng

Assessing the impacts of IPsec cryptographic algorithms on a virtual network embedding problem☆

Bernardo C.V. Camilo[a], Rodrigo S. Couto[b,*], Luís Henrique M.K. Costa[a]

[a] Universidade Federal do Rio de Janeiro, PEE/COPPE/GTA - DEL/POLI, P.O. Box 68504 - CEP 21941-972, Rio de Janeiro, RJ, Brazil
[b] Universidade do Estado do Rio de Janeiro, FEN/DETEL/PEL, CEP 20550-013, Rio de Janeiro, RJ, Brazil

ARTICLE INFO

Article history:
Received 1 July 2016
Revised 20 June 2017
Accepted 22 June 2017
Available online 1 July 2017

Keywords:
Network virtualization
Virtual network embedding
Security overhead
IPsec

ABSTRACT

Network virtualization has emerged as an alternative to traditional networking, allowing several different virtual networks to operate on the same physical infrastructure. Despite its wide adoption, virtualization still has some open issues. One of the challenges is related to resource allocation of virtual networks on the physical substrate. In the literature, this problem is known as virtual network embedding. Different papers propose virtual network embedding considering different aspects, but only a few address security, which is a key requirement for many applications. This work quantifies the overhead of cryptographic algorithms in order to use them in virtual network embedding solutions. Both theoretical and experimental evaluation of IPsec algorithms are conducted. The obtained results are applied to a known virtual network embedding problem, using realistic characteristics. These results demonstrate the importance of considering such overheads to perform the allocation of secure virtual networks.

1. Introduction

Research related to network virtualization have gained attention since it provides a more flexible network architecture, facilitating its innovation [1]. As we can see in Fig. 1, this technique allows several different virtual networks to operate on a single physical infrastructure. Despite its wide applicability, there are still many challenges to allow network virtualization in large scale infrastructures. A major challenge is related to the problem known as Virtual Network Embedding (VNE) [2]. One of the goals of VNE is to minimize the amount of idle resources in a physical infrastructure, trying to allocate more virtual networks on it. In the literature, we can find various approaches to this problem, which use different techniques and focus on specific aspects such as Quality of Service (QoS) [3], economical profit [4,5], survivability [6], efficiency in resource utilization [7–9], load balancing [10], and security [11].

In a virtualized network environment, both routers and physical links are shared by multiple virtual networks, as shown in Fig. 1. Considering security aspects, if there is no proper isolation, malicious users of a virtual network can capture or even modify packets of another virtual network, violating security properties such as confidentiality and integrity [13]. For this reason, the creation of secure virtual networks is essential for many applications. However, security has a cost. For

☆ Reviews processed and recommended for publication to the Editor-in-Chief by Area Editor Dr. Gregorio Martinez.

* Corresponding author.
E-mail addresses: bernardocamilo@ufrj.br (B.C.V. Camilo), rodrigo.couto@uerj.br, rodsouzacouto@ieee.org (R.S. Couto), luish@gta.ufrj.br (L.H.M.K. Costa).

http://dx.doi.org/10.1016/j.compeleceng.2017.06.025
0045-7906/

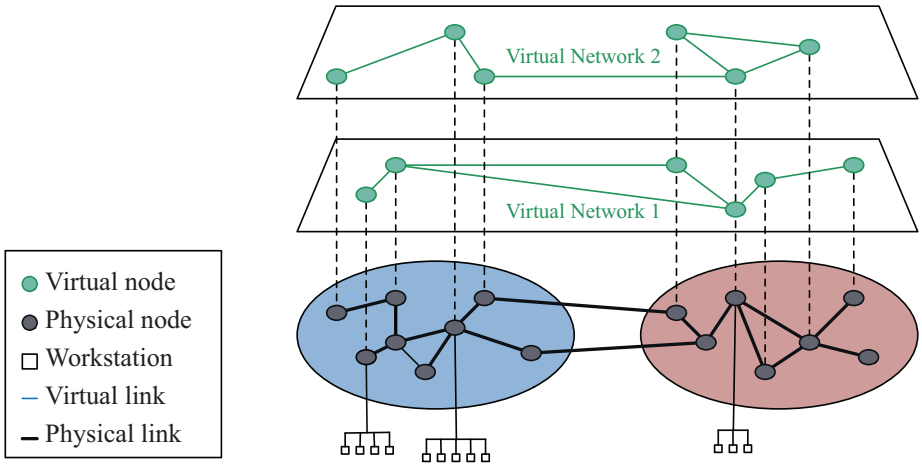


Fig. 1. Example of a network virtualization environment. Adapted from: Chowdhury and Boutaba [12].

example, providing confidentiality, imposes overheads both on packet processing, caused by cryptographic algorithms, and in bandwidth utilization, since packets receive additional headers.

In this work, we quantify the overheads imposed by security protocols and use the results to show the impact caused by its adoption in a Virtual Network Embedding problem. We consider both processing and bandwidth overheads in different experiments, using cryptographic algorithms from IPsec suite. Using the obtained results, we perform VNE simulations, evaluating the performance when security requirements are considered. These simulations have realistic characteristics, based on the analysis of different packet size distributions and on the topology of a real wide area network.

The remainder of this paper is organized as follows. Related work is discussed on Section 2. Section 3 presents an overview of the IPsec suite whereas Section 4 describes the Virtual Network Embedding problem. Section 5 elaborates the experiments conducted to evaluate the IPsec overheads, while Section 6 explains the VNE simulations and analyzes its results. Finally, Section 7 concludes this work and points out future research directions.

2. Related work

The related work spans from papers about IPSec overhead analysis in traditional networks to papers where the security of VNE algorithms is investigated. Regarding the first type of work, in [14], Xenakis et al. present an evaluation of the IPsec overheads to investigate its feasibility on portable devices. In this work, the IPsec overheads are analyzed considering both processing and bandwidth consumption, using a set of cryptographic algorithms such as DES, AES, HMAC-MD5, and HMAC-SHA-1. It presents a theoretical analysis that quantifies the number of basic operations required for execution of each algorithm and the bytes consumed by the extra fields inserted. Using simulations, Xenakis et al. concluded that the overhead imposed by IPsec using more complex security schemes, such as 3DES and AES with 192 or 256-bit keys, is substantial for the modeled mobile devices. Also, this overhead increases when adding authentication and integrity services with HMAC-MD5 or HMAC-SHA-1. This work was used as a theoretical basis to our experimental evaluation and its results are consistent with our results, as we see later in this article. However, it is worth mentioning that the processing overhead values obtained in the theoretical models are much greater than in our experimental evaluation. This happens because Xenakis et al. models assume some simplifications (e.g., only basic CPU instructions are considered and each basic instruction requires one processing cycle) and a naive implementaion of IPsec, which is no longer used in practice. In [15], Ronan et al. also analyze the impacts caused by IPsec in processing and bandwidth consumption. However, the analysis is conducted with an experimental approach and only AES and HMAC-MD5 algorithms are considered. The results reveal the same conclusion, stating that the overheads imposed by IPsec are not negligible.

Regarding VNE algorithms, Fischer et al. [2] survey the state of the art of research related to VNE. Different variants, strategies, metrics, and parameters used by VNE algorithms are considered in this work. Thus, based on an algorithm classification scheme proposed by the authors, a taxonomy of the current approaches to the VNE problem is presented along with research directions. These directions are presented in three groups: development of distributed algorithms, creation of algorithms for specific network environments and optimization of new goals. Regarding the last one, they emphasize the importance of considering security parameters in VNE algorithms, which is a topic still scarce in the literature.

Bays et al. [11] propose a VNE model that, besides optimizing the use of network resources, meets security requirements. The proposed solution is modeled using ILP (Integer Linear Programming - ILP) and considers constraints of processing, memory, location and security requirements. Security requirements are defined as a level of confidentiality for the virtual network communications. There are three confidentiality levels: end-to-end encryption, point-to-point encryption, and non-overlapping networks. At the first level, only the end points of the virtual network must be mapped to routers that are able

Table 1
Properties of a secure connection.

Property	Description
Integrity	Ensures that the message content has not been modified in transit.
Authentication	Allows source and destination to confirm the identity of their communication peer.
Confidentiality	Encrypts the message, allowing only the source and the destination to access the transmitted content.

Table 2
Security services provided by IPsec protocols. Adapted from [19].

Service	AH	ESP
Integrity	✓	
Authentication	✓	✓
Confidentiality		✓
Anti-replay protection	✓	✓

to encrypt and decrypt the traffic, running security protocol suites such as IPsec. On the second level, all virtual nodes must be mapped to routers that have this feature. Finally, in the third level, the user requests a virtual network that does not share physical elements with other virtual networks.

From the analysis of the literature on VNE, we can conclude that the study of VNE algorithms considering security parameters is quite promising. Bays' et al. [11] work goes in this direction, but security overheads are not considered. Our work has as main goal to connect research related to IPsec overheads with VNE, quantifying the impacts of cryptographic algorithms in this type of problem. Hence, we provide a basis for the development of new VNE algorithms that consider security requirements.

3. IPsec protocol suite

IPsec is a protocol suite designed to provide security for IP communications at the network layer. Its architecture ensures the establishment of secure communications, which have the properties of confidentiality, authentication and integrity, as defined in Table 1. Also, through the use of sequence numbers, IPsec provides protection against replay attacks [16].

To create a secure connection, IPsec provides two different security protocols: the Authentication Header (AH) protocol [17], and the Encapsulating Security Payload (ESP) protocol [18]. The main difference between these two protocols is that only the ESP supports confidentiality. Integrity, anti-replay protection and authentication can be provided by both protocols, as we can see in Table 2. It is important to note that AH and ESP introduce additional fields to the packets.

Both IPsec protocols, AH and ESP, support two modes of operation: transport and tunnel. The transport mode is generally employed in end-to-end connections [19], since only the payload of the IP datagram is encrypted. The tunnel mode encrypts the entire original datagram and encapsulates it as the payload of a new IP datagram. This ensures that the whole packet content is protected while its transmitted in the network. In addition to the modes of operation, IPsec has a wide range of cryptographic algorithms to encrypt, decrypt and authenticate the messages, providing different levels of security. However, they require significant processing work for the encryption, decryption and authentication of the exchanged data. Block ciphers can also pad the original packet [14], increasing its size. Given the above description, IPsec imposes two different types of overhead: processing and bandwidth overheads, which are the focus of our work. The details about these types of overhead are described later in Section 5.

4. Virtual network embedding problem

The problem known as Virtual Network Embedding (VNE) is an optimization problem, which aims to choose where the virtual networks elements will be placed on the physical substrate. VNE must be performed efficiently, which makes it the main resource allocation challenge in the network virtualization context [2]. We can find in the literature several approaches to this problem, with different assumptions, specific optimization goals, and strategies. Fischer et al. [2] present a survey that summarizes and classifies the main existing approaches. The classification scheme is based on a taxonomy which states that every VNE approach can be categorized as static or dynamic, centralized or distributed, and concise or redundant. Also, the classification considers the coordination between node and link mapping phases and the optimization strategy.

The formulation of a VNE problem can consider different types of constraints, such geographical location, resource availability and path length. However, the most common ones are related to the bandwidth and the CPU processing power. Bandwidth constraints prevent a virtual link from being mapped in a path of the physical substrate with saturated links. On the other hand, processing constraints prevent a virtual node from being allocated on a physical node where the CPU is not able to process and forward packets at the requested rate.

Fig. 2(a) shows a basic example of VNE. In this example, a physical substrate receives three virtual network requests. The physical substrate is composed of six nodes and eight links. The numbers near links represent their bandwidth capacity and

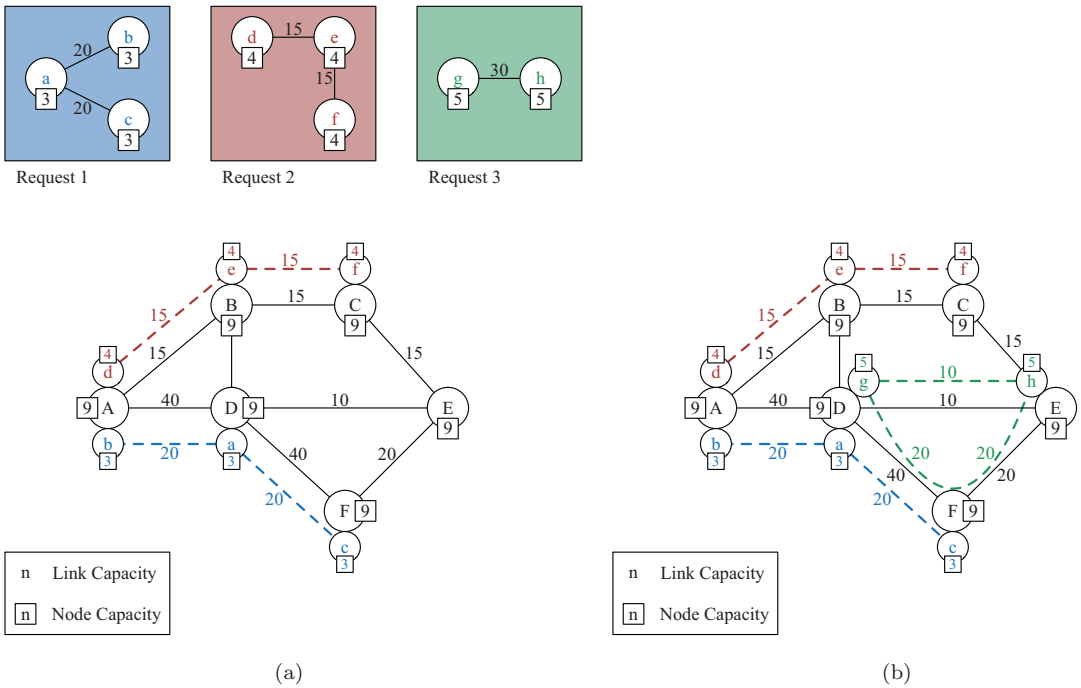


Fig. 2. Virtual Network Embedding example. Adapted from Yu et al. [20].

the numbers in rectangles are the processing capacity of each node. The virtual network requests use the same notation, showing their requested capacity. Using a given VNE algorithm, the requests 1 and 2 are mapped onto the substrate, consuming physical resources from the substrate. With that, the bandwidth requested for the virtual link of request 3 could not be satisfied, considering the remaining capacity and the constraints imposed by the algorithm. Therefore, request 3 is refused.

Imposing that each virtual link must be mapped on only one substrate path, as done in the previous example, makes the embedding problem computationally intractable (NP-hard) [20]. In addition, this constraint may lead to a high refusal of requests. Thus, Yu et al. [20] propose solutions that increase the flexibility of the physical substrate, allowing the use of simpler embedding algorithms and resulting in a more efficient resource utilization. One of their proposals is to allow the substrate network to split a virtual link over multiple substrate paths. In this case, the original problem is reduced to a multicommodity flow problem, which can be solved in polynomial time [20]. In the previous example (Fig. 2), the third request is refused due to lack of resources. However, assuming flexible path splitting support, Request 3 (i.e., two virtual nodes g and h connected through a virtual link) is splitted into two paths. The first one is composed of nodes D and E and the second one is composed of nodes D, F, and E, making the request possible, as can be seen in Fig. 2(b).

Path splitting thus allow a more efficient network utilization. By harnessing small pieces of available bandwidth, the network is able to accept more virtual network requests [20]. From a practical point of view, the physical network elements must perform multipath routing for this feature to be implemented and, if needed, a mechanism should be provided to avoid packet disordering.

With that in mind, the simulations presented later in Section 6 were conducted using a VNE algorithm that consider both processing and bandwidth constraints. Besides that, in order to simplify the problem, this algorithm assumes a physical substrate that supports path splitting.

5. IPsec overhead quantification

Fig. 3 shows the experimental testbed, which is used to run our experiments and quantify the overheads imposed by IPsec.[1] This platform consists of the following components: (i) a traffic generator, called Alice; (ii) two gateways, called Moon and Sun, which can run IPsec[2] to encrypt, decrypt and authenticate the transmitted packets; and (iii) a traffic receiver, called Bob. Table 3 shows the main specifications of each machine.

Since the experiments aims to quantify IPsec overheads when providing all its security services, the gateways are configured to use the ESP protocol in tunnel mode. As seen in Section 3, this setting allows the establishment of a secure channel

[1] A virtual testing platform is available at: https://github.com/bernardocamilo/ipsec-virtual-environment.
[2] Moon and Sun run IPsec by using the strongSwan implementation, available at: https://www.strongswan.org/.

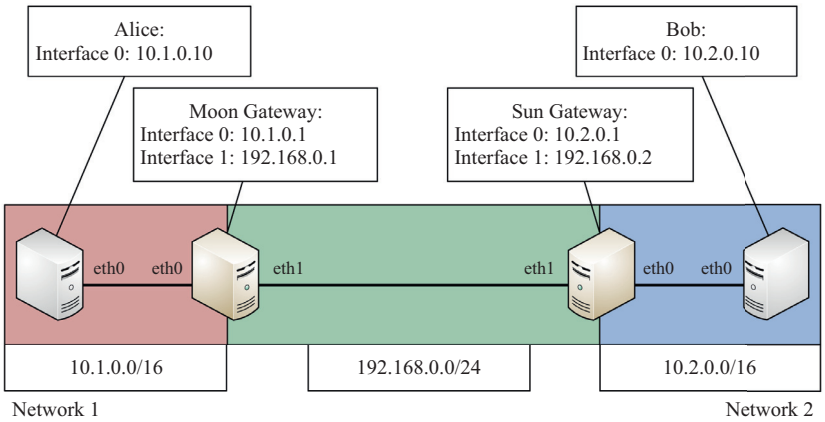


Fig. 3. Experimental testbed.

Table 3
Description of the testbed machines.

Data	Generator	Forwarders		Receiver
Nickname	Alice	Moon	Sun	Bob
Motherboard	Intel DX79SI	Intel DP55KG	Intel DX58SO2	Intel DQ45EK
Processor	Intel Core i7 CPU 3930K	Intel Core i7 CPU 860	Intel Core i7 CPU X 980	Intel Core 2 CPU 6400
RAM	8GB	8GB	12GB	2GB
Ethernet Controller	Intel Gigabit 82574L	Intel Gigabit 82576 (x4)	Intel Gigabit 82567LF-2 e 82574L	Intel Gigabit 82567LM-3
Linux kernel	3.2.0	3.13.0	3.13.0	3.2.0

that provides authentication, integrity and confidentiality. Regarding the cryptographic algorithms, we only consider those evaluated in Xenakis et al. [14], since we use their work as a theoretical basis. Among these algorithms, we choose the most secure schemes, which is AES-256 [21] to provide confidentiality and HMAC-SHA-1 [22] to provide integrity and authentication. Furthermore, these algorithms are supported by a wide range of IPsec implementations.

5.1. Processing overhead

Since AES is not a Feistel [14] cipher (i.e., a cryptographic algorithm where the decryption and encryption operations are similar), we analyze the encryption and decryption processes separately.[3] In each of these experiments, the procedure described above is executed in two different configurations. In the first one, the gateways are just forwarding packets (without IPsec), while in the second one, they are encrypting/decrypting and authenticating the traffic (with IPsec).

To collect the data needed to evaluate the processing overhead, we have developed two scripts. The first one executes in Alice and creates TCP traffic for 60 s using the Iperf[4] traffic generator. The second one runs a performance analyzing tool, called perf,[5] on Moon for 10 seconds to measure the number of instructions performed by the CPU. The difference between the execution times of these scripts guarantees that the CPU performance is measured during data transmission. We run these scripts synchronously and collect the data in both machines, varying the MSS (Maximum Segment Size) from 88 to 1398 bytes. Note that as the MSS reduces, the data size carried in each generated packet decreases, thus increasing the header overhead. For each MSS value, we repeat the experiment 30 times and all the results are obtained using a confidence interval of 95%. This process is summarized in Fig. 4.

The results of the encryption process are presented in Fig. 5, in which horizontal axes show the considered MSSs range. With the data collected in Alice machine, we plot Fig. 5(a), in which the vertical axis shows the goodput between Alice and Bob. In both curves, the goodput grows as the MSS increases. This result is expected since the header overhead is proportionally greater in smaller packages. The difference between the two curves is mainly because IPsec introduce additional headers to the packets, increasing the header overhead.

Fig. 5(b) is obtained from the data collected in Moon machine. Its vertical axis shows the number of Millions of Instructions Per Second (MIPS) performed for each value of MSS. When IPsec is not being used, the curve shows a decreasing behavior. Although the goodput is smaller for low MSS values, the number of forwarded packets per second, plotted in Fig. 6, is higher in this situation, increasing the CPU usage. Nevertheless, when using IPsec, this situation is reversed and

[3] To evaluate the decryption process, we swapped Moon and Sun positions to use the same machine as in the encryption analysis.
[4] Iperf User Docs are online at https://iperf.fr/.
[5] Perf information is available at https://perf.wiki.kernel.org.

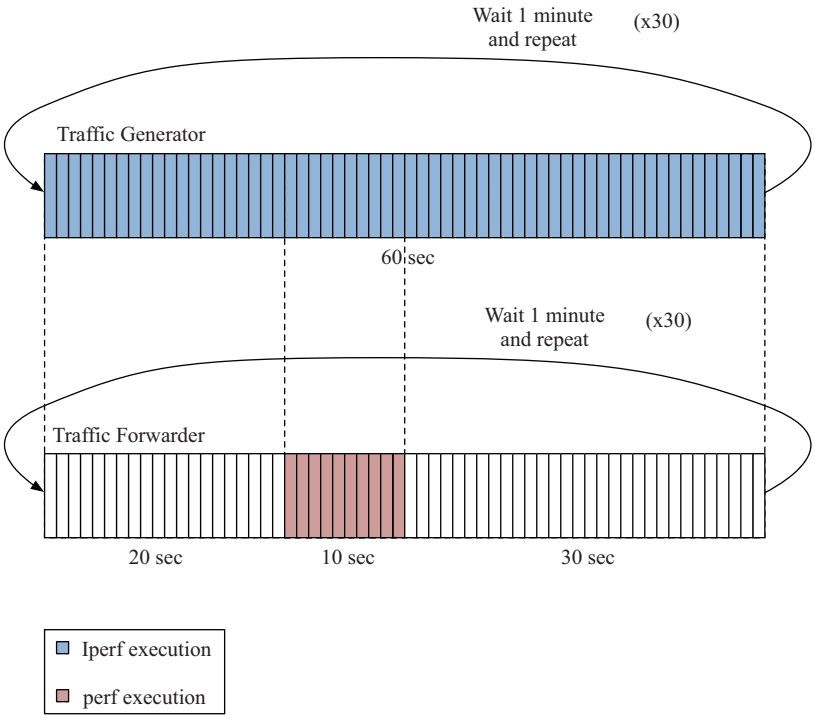
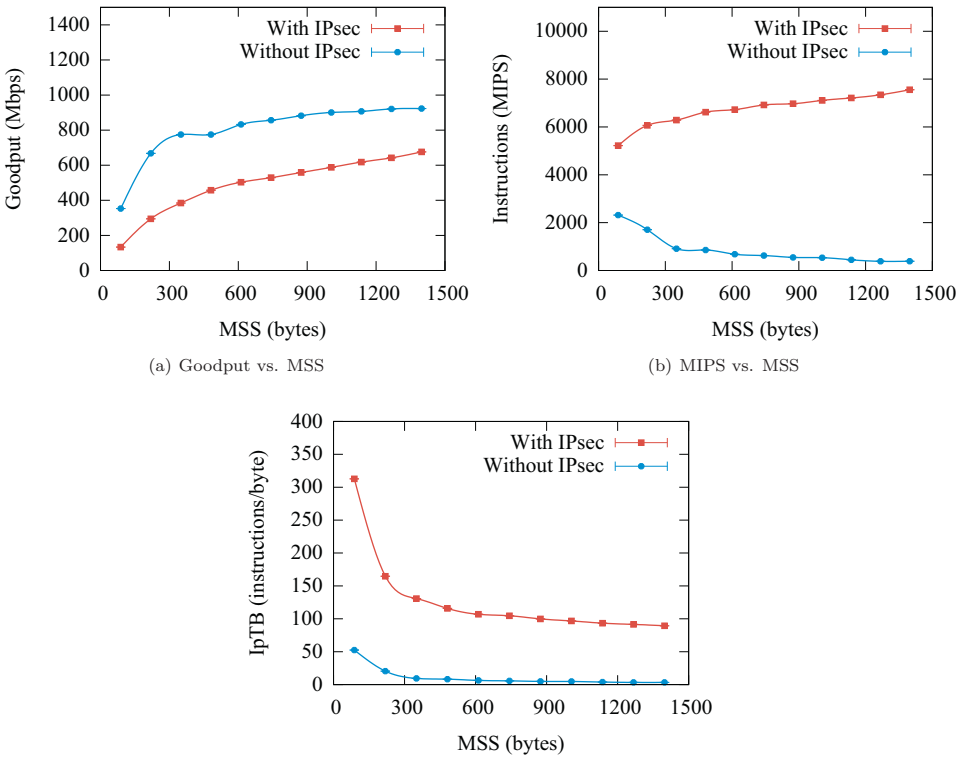


Fig. 4. Processing data gathering for each MSS value.



(a) Goodput vs. MSS

(b) MIPS vs. MSS

(c) IpTB vs. MSS

Fig. 5. Processing overhead during the encryption process.

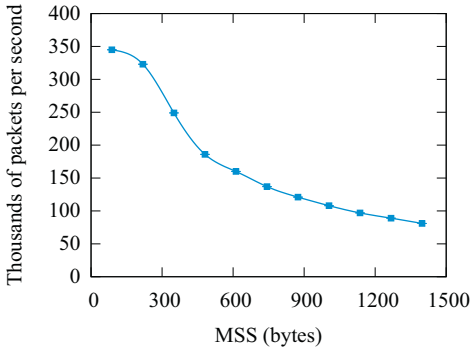
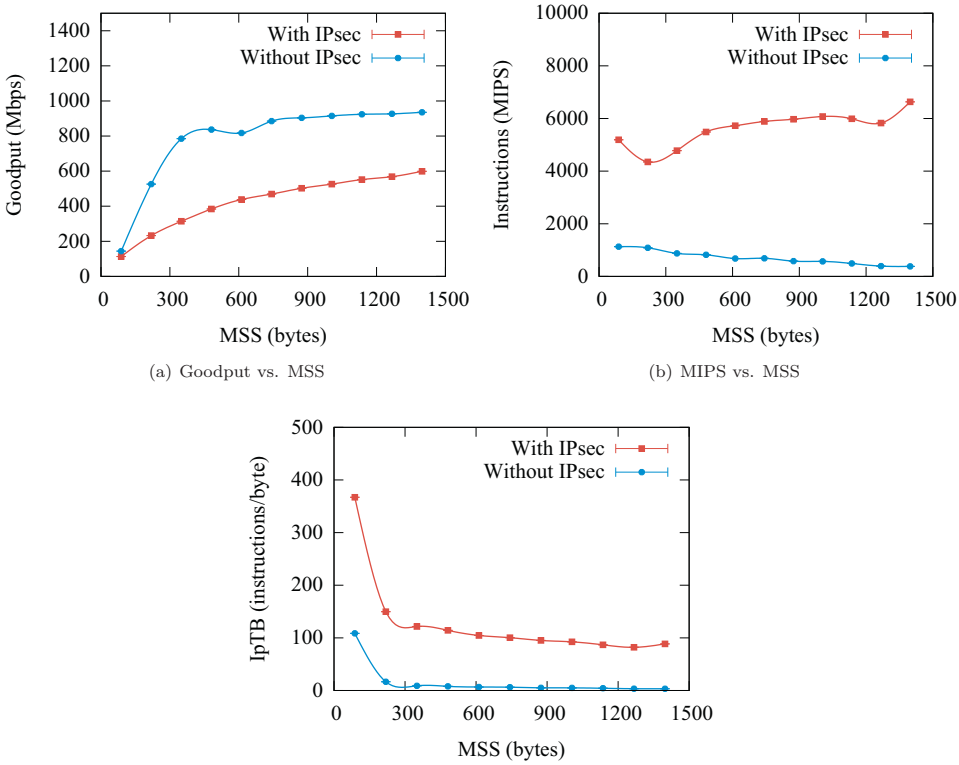


Fig. 6. Number of forwarded packets per second (without IPsec).



(a) Goodput vs. MSS

(b) MIPS vs. MSS

(c) IpTB vs. MSS

Fig. 7. Processing overhead during the decryption process.

the curve becomes increasing, as shown in Fig. 5(b). This happens because the processing effort to encrypt a packet is much higher than the effort to simply forward the packet. In addition, larger packets require more processing, since the entire payload is encrypted. Hence, at higher goodput situations, more data is encrypted, increasing the number of MIPS executed by the CPU.

Finally, Fig. 5(c) shows the relation between the number of instructions performed and the MSS, disregarding goodput differences. In other words, the values of Fig. 5(b) are divided by the values of Fig. 5(a) and the confidence intervals are reevaluated, producing a new metric called Instructions per Transmitted Byte (IpTB). The results show that the lower the MSS, the more IpTB are executed. Therefore, assuming that the network transmission capacity in bits/s can be achieved regardless of the packet size, it is clear that the worst overhead occurs when small packets are forwarded in the network.

The results of the decryption process are shown in the plot of Fig. 7 and their conclusions are similar to the encryption case.

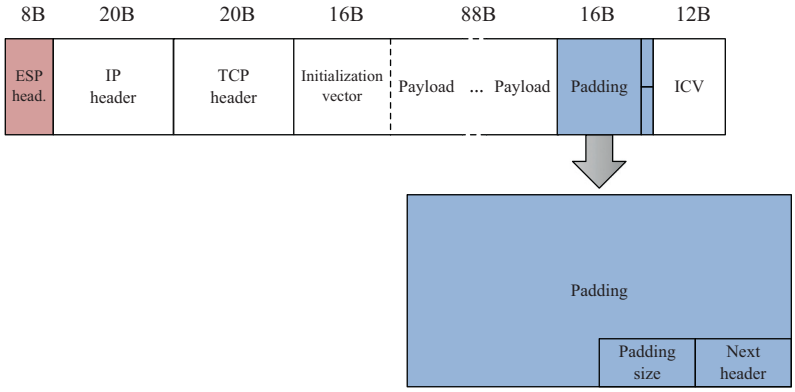


Fig. 8. Secure TCP segment.

5.2. Bandwidth overhead

Since the bandwidth overhead is only consequence of the addition of ESP field and ciphers padding, we use the theoretical concepts introduced in Section 3 to derive a general formula that gives the size of protected packets as a function of the payload size.[6] Considering the combination of security algorithms, defined previously, and a payload of 88 bytes, Fig. 8 illustrates the fields of a secure TCP segment. The fields with its respective sizes are shown below:

- ESP Header: $H_{ESP} = 8$ bytes,
- Original IP Header with no options: $H_{IP} = 20$ bytes,
- TCP Header: $H_{TCP} = 20$ bytes,
- TCP Payload: $Pl = 88$ bytes,
- Initialization Vector: $IV = 16$ bytes,
- Padding: $Pad = 14$ bytes,
- ESP Trailer: $Tr_{ESP} = 2$ bytes,
- Integrity Check Value HMAC-SHA-1: $ICV_{HMAC-SHA-1} = 12$ bytes.

The use of an Initialization Vector (IV) and the padding size are a consequence of the adopted cryptographic algorithm. In the case of AES-CBC algorithm, an IV is required and its size is 16 bytes [21]. Padding is used to set the size of the encrypted data as a multiple of the block size (Bl), which is also 16 bytes in this case. Using this reasoning, the padding size can be evaluated by Eq. (1):

$$Pad(Pl) = \left\lceil \frac{Pl + H_{TCP} + H_{IP} + Tr_{ESP}}{Bl} \right\rceil \times Bl - (Pl + H_{TCP} + H_{IP} + Tr_{ESP}). \tag{1}$$

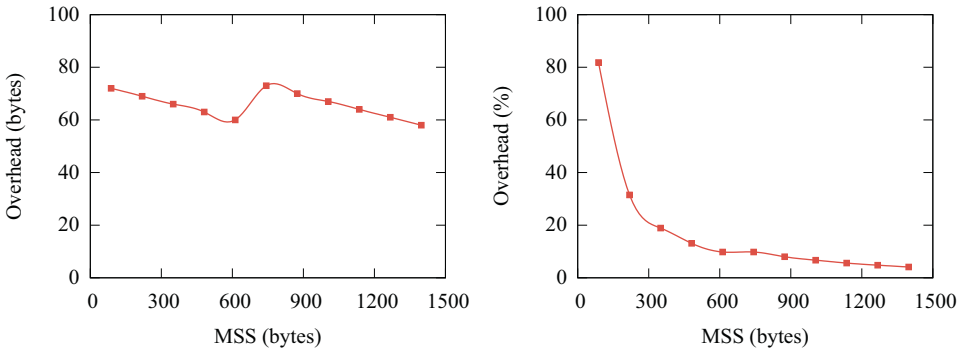
The other fields in the segment have fixed sizes.

Adding all these fields together, Eq. (2) evaluates the final size of an IP packet after being encrypted and authenticated by the ESP protocol in tunnel mode, using the AES and HMAC-SHA-1 combination:

$$\begin{aligned} PPS(Pl) &= H_{ESP} + H_{IP} + H_{TCP} + Pl + IV + Pad + Tr_{ESP} + ICV_{HMAC-SHA-1} \\ &= \left\lceil \frac{Pl + H_{TCP} + H_{IP} + Tr_{ESP}}{Bl} \right\rceil \times Bl + H_{ESP} + H_{IP} + ICV_{HMAC-SHA-1} + IV. \end{aligned} \tag{2}$$

The plots in Fig. 9 show the results of the bandwidth overhead analysis. In this figure, the horizontal axes represent the same MSS range employed in Section 5.1. The vertical axis in the plot of Fig. 9(a) shows the absolute overhead ($PPS - (H_{IP} + H_{TCP} + Pl)$) in bytes per packet. The variations present in this plot are due to the different padding sizes for each MSS value used. In the plot of Fig. 9(b), the vertical axis represents the percentage of that overhead compared to the original size of the IP packet. Thus, as in the processing overhead analysis, the worst case occurs when small packets are forwarded in the network.

[6] We validate the derived formula by analyzing the packets captured by the tcpdump tool (http://www.tcpdump.org/manpages/tcpdump.1.html) in our testbed.



(a) Absolute bandwidth overhead. (b) Relative bandwidth overhead.

Fig. 9. Bandwidth overhead.

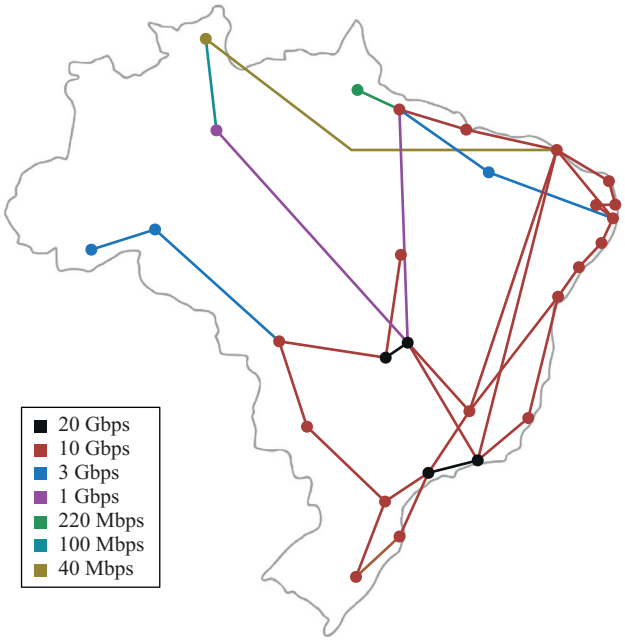


Fig. 10. Physical substrate - Ipê Network.

6. Security-aware virtual network embedding

Using the results obtained in Section 5, we conduct a series of VNE simulations to assess the impact of IPsec in a VNE problem. These simulations were performed in a realistic scenario, based on a real topology and on different packet size distributions.

6.1. Scenario description

In our scenario, several research and education institutions are interconnected through a WAN that supports virtualization and multipath routing, increasing the network efficiency, as discussed in Section 4. Each institution makes virtual network requests to create a virtual connection with any other institution. These requests are mapped using a VNE algorithm. Services which connect universities and research institutes with private connections are very common in National Research and Education Networks (NRENs), such as the Brazilian RNP (*Rede Nacional de Ensino e Pesquisa*).[7]

Hence, our physical substrate example, illustrated in Fig. 10, is based on the Ipê Network, operated by RNP. The Ipê network consists of a WAN, with 28 Points of Presence (PoPs) spread throughout Brazil, interconnecting universities and research institutes.

[7] RNP website is available at http://www.rnp.br/en.

Table 4
Definition of PI variable.

Model (m)	Formula	Approximate result (IpTB)
Uniform distribution (α)	$PI_\alpha = \frac{\sum_{s=0}^{10} PCI_s}{11}$	11.35
Smallest packet only (β)	$PI_\beta = PCI_0$	52.48
Biggest packet only (γ)	$PI_\gamma = PCI_{10}$	3.35
Bimodal distribution (δ)	$PI_\delta = \frac{4PCI_0 + \frac{4}{9}\sum_{s=1}^{9} PCI_s + 2PCI_{10}}{11}$	24.68

As for the location of each PoP, we consider the central geographic coordinates of the city where the PoP is installed, as reported by GoogleMaps.[8] The capacity of each substrate link is the real capacity reported at the RNP website, specified in Fig. 10. Also, we assume that each PoP has a machine with the same specifications of Moon (Table 3) and, therefore, the processing capacity of each PoP was based on benchmarks[9] of Moon's CPU.

To enable a broader analysis of IPsec impact, the simulations are performed using four different packet size distributions. We use in the models the same MSS range from the experiments in Section 5. Hence, the packet payload ranges from 88 bytes to 1398 bytes. The intermediate packet sizes vary at intervals of 131 bytes between these two values.

The first packet size distribution model (α) assumes that transmitted packets follow an uniform distribution. Therefore, all the packets with payload size from 88 to 1398 bytes have the same probability of being generated. The second (β) and the third (γ) models consider, respectively, worst and best cases. In the worst case, only the smallest packet is transmitted on the network. By contrast, in the best case, only the largest packet is transmitted. Finally, the fourth model (δ) assumes that the traffic is bimodal, similar to the behavior observed on the Internet, in which 40% of the packets have 40 bytes and 20% have 1500 bytes [23]. As the MSS values considered in the experiments vary only from 88 to 1398 bytes, we assume that 40% of the traffic consists of packets with a 88-byte payload, 20% of packets with a 1398-byte payload, and the remaining 40% by intermediate packet sizes uniformly distributed.

We have developed a script that generates random network topologies to create virtual networks requests. Each request has a central node (i.e., the requesting institution), which has to be connected to a given set of nodes (i.e., the peer institutions). Therefore, the topology of requested virtual networks is a star. Also, we assume that each requesting institution can connect with up to four other peers and requested virtual link capacities spans from 100 Mbps to 1 Gbps, in steps of 100 Mbps. Both the number of peers and the virtual link capacities are randomly chosen with the same probability.

6.2. Generation of virtual environments

Initially, four base environments are generated, which do not consider security overheads. Each environment is specific to one of the packet size distribution models described in Section 6.1. Based on each model, a set of security parameters are evaluated. Such parameters are applied to the virtual requests of the base environments to generate the secure environments, which consider security overheads.

The first step to create the base environments is to run the random network generator script mentioned in Section 6.1, which evaluates the bandwidth demands. In addition, the script evaluates the processing demand for each virtual node using Eq. (3):

$$PD = \sum_{l=1}^{degree} C_l \times PI_m, \tag{3}$$

where C_l is the capacity requested for the virtual link l and PI_m is the number of Instructions per Transmitted Byte (IpTB) with IPsec turned off, obtained in Fig. 5.[10] The PI_m parameter takes different values for each packet size distribution model ($m = \{\alpha, \beta, \gamma, \delta\}$), as shown in Table 4. This parameter is evaluated as the average overhead considering all packets sizes in a given distribution, as shown by the associated formula in Table 4. In these formulae, PCI denotes the number of IpTB obtained without IPsec and s is the index of the MSS employed, which is in the range of 0 (MSS = 88 bytes) to 10 (MSS = 1398 bytes).

Using the methodology described above, we generate 100 virtual networks requests and, based on these requests and on the physical substrate, we create the four base environments to be used in the simulation. The generation of the secure environments is performed using these base environments and a set of security parameters specific to each packet size distribution model. Such parameters can be interpreted as penalties that are applied to the virtual network requests, increasing their demands for resources. We define two types of parameters, processing penalties (PP_α, PP_β, PP_γ, PP_δ) and bandwidth penalties (BP_α, BP_β, BP_γ, BP_δ), shown in Table 5. In the formulae, PCI variable and the index s have the same meanings

[8] GoogleMaps is available at https://www.google.com/maps.
[9] CPU World Benchmarks are available at http://www.cpu-world.com/.
[10] Since both Figs. 5 and 7 revealed similar results, only the first one is used in these simulations.

Table 5
Security penalties.

Model	Formula	Approximate result
Uniform distribution (α)	$PP_\alpha = \frac{\sum_{s=0}^{10} PCS_s}{\sum_{s=0}^{10} PCI_s}$	11.35
	$BP_\alpha = \frac{\sum_{s=0}^{10} BCS_s}{\sum_{s=0}^{10} BCI_s}$	1.24
Smallest packet only (β)	$PP_\beta = \frac{PCS_0}{PCI_0}$	5.96
	$BP_\beta = \frac{BCS_0}{BCI_0}$	2.09
Biggest packet only (γ)	$PP_\gamma = \frac{PCS_{10}}{PCI_{10}}$	26.66
	$BP_\gamma = \frac{BCS_{10}}{BCI_{10}}$	1.06
Bimodal distribution (δ)	$PP_\delta = \frac{4PCS_0 + \frac{4}{9}\sum_{s=1}^{9} PCS_s + 2PCS_{10}}{4PCI_0 + \frac{4}{9}\sum_{s=1}^{9} PCI_s + 2PCI_{10}}$	7.60
	$BP_\delta = \frac{4BCS_0 + \frac{4}{9}\sum_{s=1}^{9} BCS_s + 2BCS_{10}}{4BCI_0 + \frac{4}{9}\sum_{s=1}^{9} BCI_s + 2BCI_{10}}$	1.51

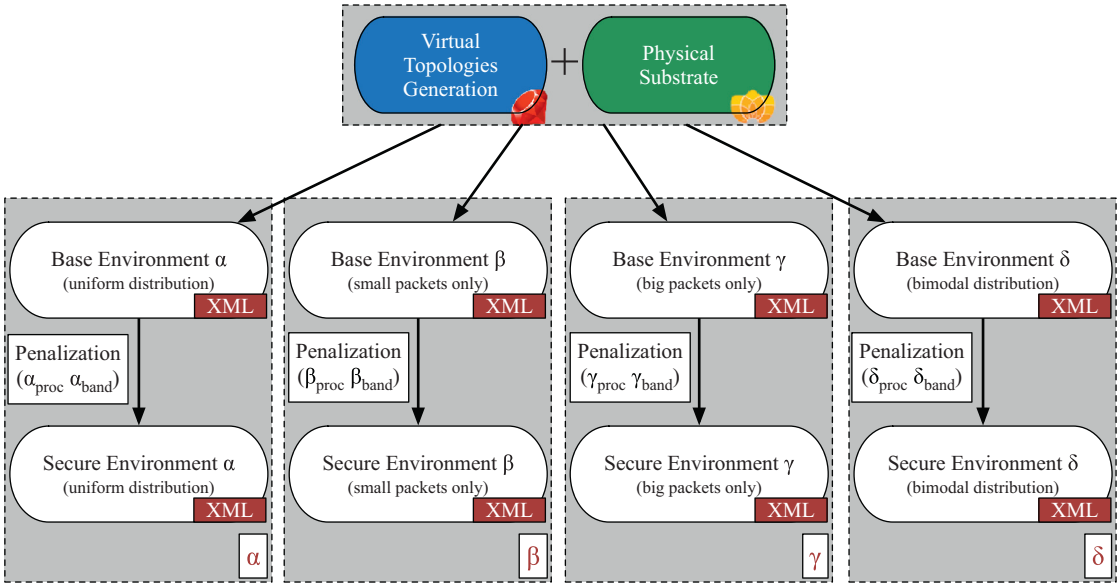


Fig. 11. Virtual environments generation.

as in Table 4. The *PCS* variable denotes the number of IpTB demanded with the adoption of IPsec. Finally, the *BCI* and *BCS* variables represent, respectively, the packet sizes transmitted in the cases without and with IPsec. In a nutshell, each formula represents the ratio between the average overhead with IPsec and without IPsec. The diagram in Fig. 11 summarizes the process used to create base and secure environments. Each environment is described as an XML file to be used in the simulator, described next.

To map the generated requests into the physical substrate, we employ a VNE simulator called ALEVIN.[11] According to its developers, ALEVIN is a framework created to support the design of new VNE algorithms, to ease their comparison and analysis, and to provide arbitrary evaluation metrics [24]. The VNE environments can be created either using a graphical interface or writing an XML file. Moreover, many VNE algorithms proposed in the literature and its parameters are already implemented in the simulator, together with performance metrics that can be used to compare them.

6.3. Simulation results

As mentioned in the earlier sections, we consider that our physical substrate implements multipath routing. Hence, we employ an algorithm from ALEVIN that can take advantage of this feature. This algorithm is based on Yu's proposal [20] and uses linear programming to solve the multicommodity flow problem.

[11] ALEVIN is available at http://sourceforge.net/projects/alevin/.

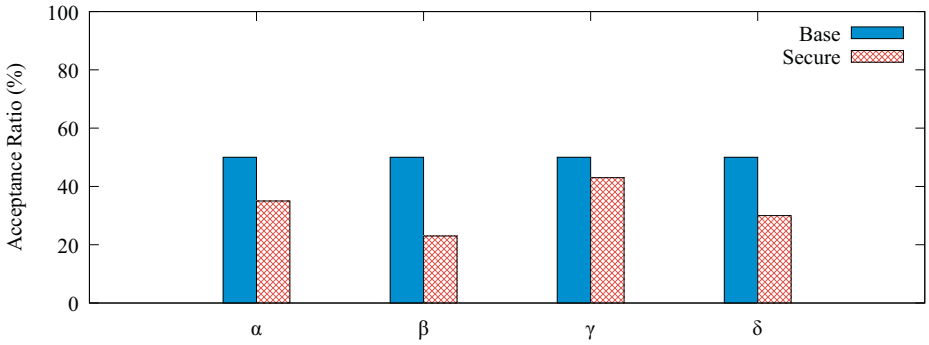
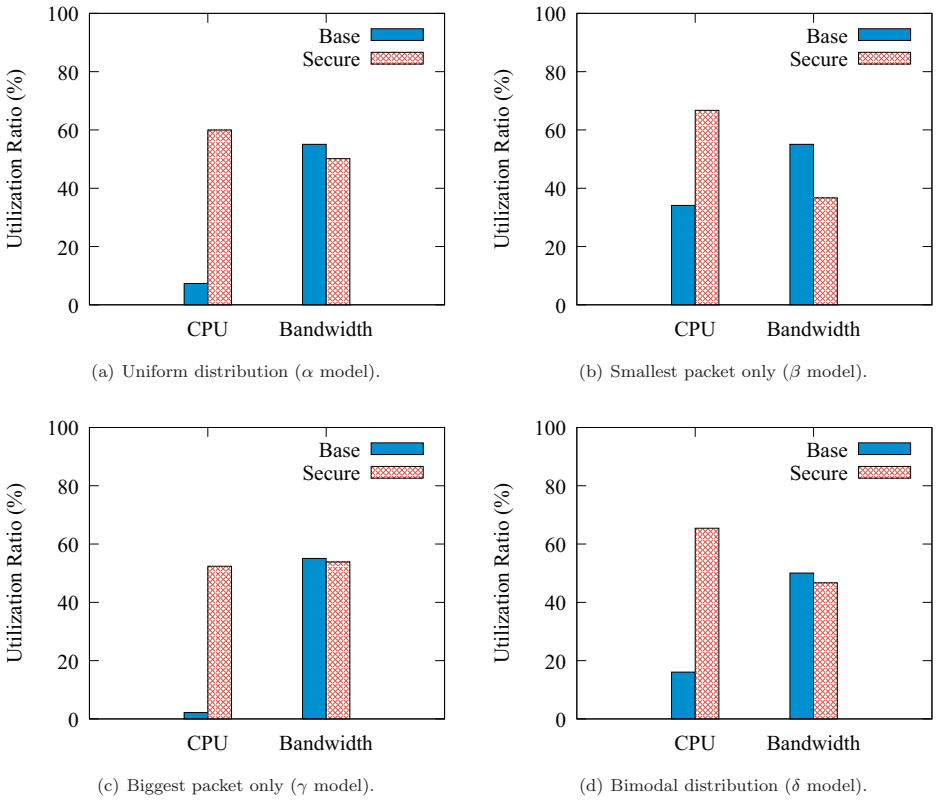


Fig. 12. Acceptance Ratio.



(a) Uniform distribution (α model).

(b) Smallest packet only (β model).

(c) Biggest packet only (γ model).

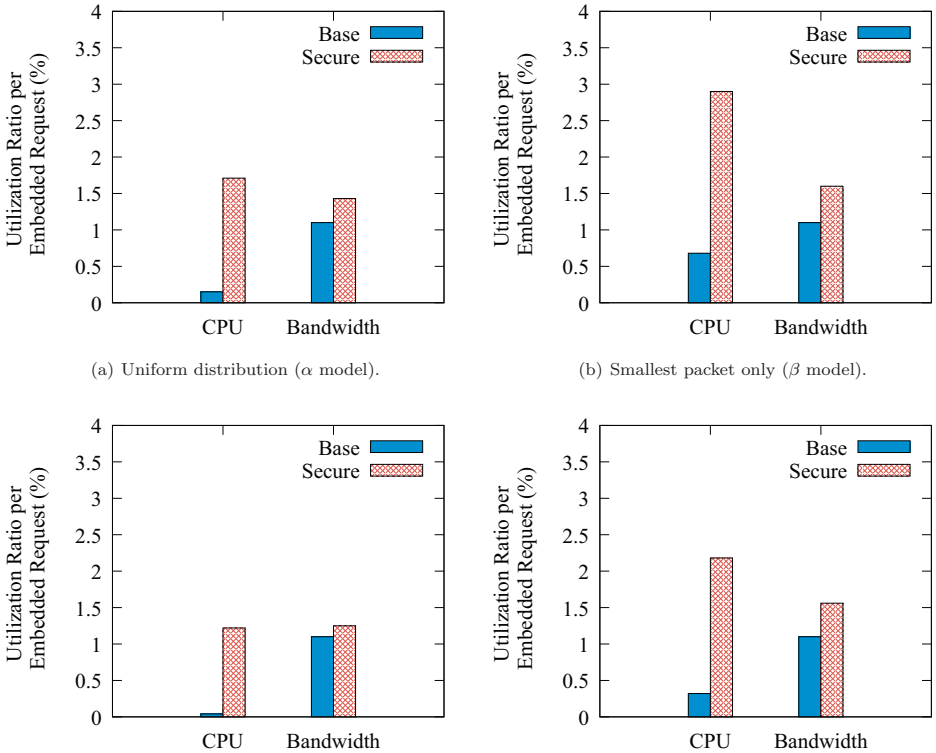
(d) Bimodal distribution (δ model).

Fig. 13. Utilization Ratio.

In order to ease the measurement and comparison between the mapping results, we define the following metrics:

- Acceptance Ratio: The ratio between the number of accepted and requested virtual networks,
- Utilization Ratio: The ratio between the amount of used resources and the total amount of available resources, considering a particular resource type (CPU or bandwidth),
- Utilization Ratio per Embedded Request: The ratio between the Utilization Ratio of a given resource and the number of embedded virtual networks.

From these metrics, our simulation results highlight the importance of considering security algorithms overheads in the embedding of secure virtual networks. Fig. 12 shows the Acceptance Ratio for each environment generated by the process of Fig. 11. These results show that, in all the considered models, the Acceptance Ratio of virtual networks has a considerable drop for secure network requests, reaching a decrease of 54% in the worst case, model β, where all the packets have the smallest payload. To complement these results, Fig. 13 shows the resource Utilization Ratio of the physical substrate. The increased use of CPU resources is evident, especially using the γ model, in which processing overhead reaches the highest



(a) Uniform distribution (α model).

(b) Smallest packet only (β model).

(c) Biggest packet only (γ model).

(d) Bimodal distribution (δ model).

Fig. 14. Utilization Ratio per Embedded Request.

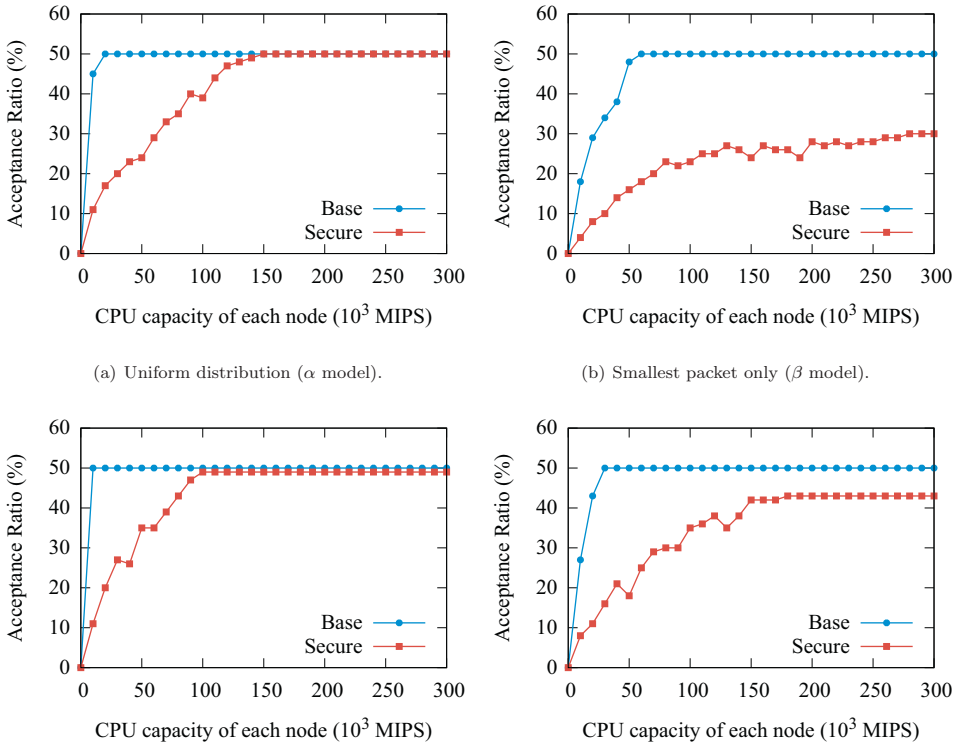
value due to the large penalty applied. On the other hand, bandwidth consumption suffers a slight drop because fewer networks were accepted when we introduce security parameters. Furthermore, the bandwidth overhead is not as high as the processing overhead which, even with less accepted networks, has a higher utilization. Fig. 14 shows the Utilization Ratio per Embedded Request. Hence, this result analyzes the overhead regardless of the number of accepted networks, showing the high impact of IPsec.

To make the analysis agnostic of the specific CPU capacity used in our previous simulations, we repeat the process using different CPU capacities in ALEVIN. Fig. 15 shows the Acceptance Ratio of virtual network requests as a function of the processing power assigned to each node in the physical substrate. As expected, the acceptance of virtual networks tends to increase as the processing capacity increases. However, after a certain CPU capacity, the Acceptance Ratio saturates. For example, assuming the β model, we can reach an Acceptance Ratio of 50% on the base environment, using a CPU that can perform around 60,000 MIPS. But when we consider the secure environment, it is only possible to reach an Acceptance Ratio of 30%, even with a very powerful machine that can perform around 300,000 MIPS. This demonstrates that, even assuming very high values to the processing capacity of the nodes, IPsec overhead when embedding virtual networks is still significant. Hence real network virtualization platforms, such as FITS (Future Internet Testbed with Security) [25], should consider this overhead in their allocation schemes.

7. Conclusion

The idea of our work is to draw attention for the security overhead problem. Hence, our work is an effort to help the community developing VNE algorithms that consider this impact. To achieve this goal, we presented an experimental evaluation of the IPsec communication overhead and evaluated its impacts in the VNE problem. We have first set up a testing platform to run experiments and quantify both processing and bandwidth overheads of IPsec. Then, we used the experimental results to perform a series of VNE simulations. These simulations considered a realistic scenario, based on a real topology and different packet size distributions. From that, we have concluded that IPsec overheads can have major influence on the acceptance rate of virtual networks in a VNE problem.

Considering all the evaluated packet size distributions, the adoption of IPsec increases significantly both CPU and bandwidth consumption per embedded virtual network request. The CPU usage increases up to 300% with large packets only (γ model) while the bandwidth consumption increases up to 45% with small packets only (β model). Although the CPU overhead is much greater than the bandwidth overhead, we cannot neglect its impacts. Even assuming unlimited CPU capacity,



(a) Uniform distribution (α model).

(b) Smallest packet only (β model).

(c) Biggest packet only (γ model).

(d) Bimodal distribution (δ model).

Fig. 15. Acceptance ratio vs. CPU capacity.

the Acceptance Ratio can decrease by 40% (β model). It is worth mentioning that, regardless the employed topology or VNE algorithm, the IPsec overhead remains the same and thus significantly affect the Acceptance and Utilization ratio.

As a future work, using the same methodology, new experiments can be conducted using other combinations of cryptographic algorithms, to obtain results for different security levels. Also, we consider to perform this analysis in a real platform with virtualization support and compare the VNE results with the simulations presented in this work.

Acknowledgments

This work was supported by CNPq, FAPERJ and CAPES research agencies.

References

[1] Couto RS, Campista MEM, Costa LHMK. Network resource control for Xen-based virtualized software routers. Comput Netw 2014;64:71–88.
[2] Fischer A, Botero J, Till Beck M, de Meer H, Hesselbach X. Virtual network embedding: a survey. Commun Surv Tutor, IEEE 2013;15(4):1888–906.
[3] Trinh T, Esaki H, Aswakul C. Quality of service using careful overbooking for optimal virtual network resource allocation. In: Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), 8th International Conference; 2011. p. 296–9.
[4] Zhu F, Wang H. A modified ACO algorithm for virtual network embedding based on graph decomposition. Comput Commun 2016;80:1–15.
[5] Abdelwahab S, Hamdaoui B, Guizani M, Znati T. Efficient virtual network embedding with backtrack avoidance for dynamic wireless networks. IEEE Trans Wireless Commun 2016;15(4):2669–83.
[6] Rahman M, Aib I, Boutaba R. Survivable virtual network embedding. In: Networking, 9th International IFIP TC 6 Networking Conference. In: Lecture Notes in Computer Science, vol. 6091. Springer; 2010. p. 40–52.
[7] Guan X, Choi BY, Song S. Energy efficient virtual network embedding for green data centers using data center topology and future migration. Comput Commun 2015;69:50–9.
[8] Ogino N, Kitahara T, Arakawa S, Murata M. Virtual network embedding with multiple priority classes sharing substrate resources. Comput Netw 2017;112:52–66.
[9] Liao J, Feng M, Qing S, Li T, Wang J. Live: learning and inference for virtual network embedding. J Netw Syst Manage 2016;24(2):227–56.
[10] Mijumbi R, Serrat J, Gorricho JL, Boutaba R. A path generation approach to embedding of virtual networks. IEEE Trans Netw Serv Manage 2015;12(3):334–48.
[11] Bays LR, Oliveira RR, Buriol LS, Barcellos MP, Gaspary LP. Security-aware optimal resource allocation for virtual network embedding. In: Proceedings of the 8th International Conference on Network and Service Management; 2012. p. 378–84.
[12] Chowdhury NMK, Boutaba R. A survey of network virtualization. Comput Netw 2010;54(5):862–76.
[13] Coppolino L, D'Antonio S, Mazzeo G, Romano L. Cloud security: emerging threats and current solutions. Comput Electr Eng 2017;59:126–40.
[14] Xenakis C, Laoutaris N, Merakos L, Stavrakakis I. A generic characterization of the overheads imposed by IPsec and associated cryptographic algorithms. Comput Netw 2006;50(17):3225–41.

[15] Ronan J, Davy S, Malone P, Foghl M. Performance implications of IPsec deployment. In: 2nd International Workshop on Inter-Domain Performance and Simulation; 2004. p. 1–5.
[16] Kent S., Seo K. Security architecture for the internet protocol. 2005. RFC 4301.
[17] Kent S. IP authentication header. 2005a. RFC 4302.
[18] Kent S. IP encapsulating security payload (ESP). 2005b. RFC 4303.
[19] Stallings W. Cryptography and network security: principles and practice. 5th ed. Prentice Hall; 2010.
[20] Yu M, Yi Y, Rexford J, Chiang M. Rethinking virtual network embedding: substrate support for path splitting and migration. SIGCOMM Comput Commun Rev 2008;38(2):17–29.
[21] Frankel S., Glenn R., Kelly S. The AES-CBC cipher algorithm and its use with IPsec. 2003. RFC 3602; IETF.
[22] Madson C., Glenn R. The use of HMAC-SHA-1-96 within ESP and AH. 1998. RFC 2404; IETF.
[23] Sinha R, Papadopoulos C, Heidemann J. Internet packet size distributions: Some observations. University of Southern California; 2007. Tech. rep.
[24] Fischer A, Botero Vega JF, Duelli M, Schlosser D, Hesselbach Serra X, De Meer H. Alevin - a framework to develop, compare, and analyze virtual network embedding algorithms. In: Proceedings of the Workshops der Wissenschaftlichen Konferenz Kommunikation in Verteilten Systemen; 2011. p. 1–12.
[25] Moraes IM, Mattos DMF, Ferraz LHG, Campista MEM, Rubinstein MG, Costa LHMK, et al. FITS: A flexible virtual network testbed architecture. Comput Netw 2014;63:221–37.

Bernardo C.V. Camilo is a Master of Science student in the Department of Informatics at Pontifícia Universidade Católica do Rio de Janeiro and a software engineer at Globo.com, part of the largest media company in Brazil. His interests include distributed computing, computer networks and information security. Camilo has a Computer Engineer degree from Universidade Federal do Rio de Janeiro.

Rodrigo S. Couto is an Associate Professor with Universidade do Estado do Rio de Janeiro (UERJ). His research interests include data center networks, cloud computing, network reliability, and network virtualization. Couto has a Doctor of Science degree in Electrical Engineering from Universidade Federal do Rio de Janeiro (UFRJ). He is a member of IEEE.

Luís Henrique M.K. Costa is an Associate Professor with Universidade Federal do Rio de Janeiro (UFRJ). His research interests include routing, wireless networks, and future Internet. Costa has a Doctor degree in Computer Science from Université Pierre et Marie Curie (UPMC). He is a Senior Member of IEEE and Member of ACM.