Ανάλυση και εμπειρική αξιολόγηση του προτύπου **IEEE 802.1AE (MAC Security)**

Η Μεταπτυχιακή Διατριβή κατατέθηκε στο τμήμα
Μηχανικών Πληροφοριακών & Επικοινωνιακών Συστημάτων
του Πανεπιστημίου Αιγαίου
σε μερική εκπλήρωση των απαιτήσεων για το
Μεταπτυχιακό Δίπλωμα Ειδίκευσης στις
Τεχνολογίες και Διοίκηση Πληροφοριακών και
Επικοινωνιακών Συστημάτων

Χαλβατζή Νικολέτα

Εαρινό Εξάμηνο 2019-2020

Analysis and empirical evaluation of IEEE 802.1AE (MAC Security) standard

A dissertation
submitted to the Department of Information & Communication Systems Engineering
of the University of the Aegean
in partial fulfilment of the requirements
for the degree of
Master of Science

Chalvatzi Nikoleta

Committee

Supervisor: Dr. Georgios Kambourakis - Associate Professor
Member: Dr. Dimosthenis Vougioukas - Assistant Professor
Member: Dr. Marios Anagnostopoulos - Research Fellow

Spring Semester 2019-2020

Statement of Authenticity

I declare that this thesis is my own work and was written without literature other than the sources indicated in the bibliography. Information used from the published or unpublished work of others has been acknowledged in the text and has been explicitly referred to in the given list of references. This thesis has not been submitted in any form for another degree or diploma at any university or other institute of tertiary education.

Karlovasi, 10-2-2018

Chalvatzi Nikoleta

(Signature)

Περίληψη

Οι διάφορες εφαρμογές ενός τοπικού δικτύου (LAN - Local Area Network) το καθιστούν αναμφισβήτητα ένα σημαντικό και αναπόσπαστο μέρος πολλών σύγχρονων εγκαταστάσεων, όπως σχολείων, πανεπιστημίων και κτιρίων γραφείων. Σήμερα, πολλές απειλές προσπαθούν να υποβαθμίσουν την ασφάλεια του δικτύου, ενώ η τρέχουσα προσοχή των επιχειρήσεων κατευθύνεται σε εσωτερικές παραβιάσεις δεδομένων. Σημαντικές ερευνητικές μελέτες πραγματοποιούνται για να κατανοήσουμε πόσο αποτελεσματικά είναι τα πρωτόκολλα ασφαλείας έναντι αυτών των απειλών. Η παρούσα διατριβή εστιάζει στο σχετικά νέο πρωτόκολλο MACSec με στόχο (α) να ερευνήσει το πώς το MACSec προστατεύει τη συνεδρεία δύο μηχανών από έναν εσωτερικό εισβολέα, (β) να διερευνήσει τη σχέση του με το IPSec, και (γ) να αξιολογήσει εμπειρικά το φόρτο που προσθέτει το πρωτοκόλλο MACSec σε υπηρεσίες όπως η μεταφορά αρχείων.

Για τους σκοπούς της μελέτης, δημιουργείται μια συνεδρία MACSec μεταξύ δύο εικονικών μηχανών (VMs). Στη συνέχεια, ορισμένα πακέτα δεδομένων ανταλλάσσονται μεταξύ των VMs ενώ ένα τρίτο VM με ρόλο εισβολέα όντας συνδεδεμένος στο ίδιο LAN επιχειρεί να συλλάβει την κυκλοφορία των πακέτων. Αργότερα, ενεργοποιούμε το FTP και το HTTP στα μηχανήματα και καταγράφουμε ξανά τα ανταλλαγμένα πακέτα χρησιμοποιώντας τη μηχανή του επιτιθέμενου. Τελικώς, αντιγράφουμε τρία αρχεία με διαφορετικά μεγέθη μεταξύ των εικονικών μηχανών και καταγράφουμε τον χρόνο της ανταλλαγής όταν η MACSec συνεδρεία ειναι απενεργοποιημένη και ενεργοποιημένη.

Τα αποτελέσματα αποκαλύπτουν ότι το πρωτόκολλο MACSec διασφαλίζει την συνεδρία των δύο VM και ο εισβολέας δεν μπορεί να έχει πρόσβαση σε ιδιωτικά δεδομένα. Επιπλέον, το FTP λειτουργεί μέσω της ασφαλούς συνεδρίας του πρωτοκόλλου MACSec, ενώ το HTTP δεν σχετίζεται με τη συνεδρία του πρωτοκόλλου MACSec και τα πακέτα δεδομένων είναι ευάλωτα σε εσωτερικές επιθέσεις. Τελικώς παρατηρούμε ότι η MACSec συνεδρεία επιμηκύνει τον χρόνο για την μεταφορά αρχείων. Καταλήγουμε στο συμπέρασμα ότι το MACSec είναι αποτελεσματικό στην προστασία από έναν εσωτερικό εισβολέα που επιχειρεί να παρακολουθήσει την κυκλοφορία πακέτων και ότι διαφορετικά πρωτόκολλα είτε υποστηρίζουν είτε είναι ανεξάρτητα από το πρωτόκολλο MACSec. Μια πλήρης αξιολόγηση πρέπει να εκπληρωθεί σε συνδυασμό με τη διερεύνηση εξωτερικών επιθέσεων, άμεσων επιθέσεων, και τη συνεργασία του MACSec με άλλα πρωτόκολλα. Η προτεινόμενη προσέγγιση αποτελεί ένα σημαντικό βήμα προς μια πλήρη αξιολόγηση της αποτελεσματικότητας του πρωτοκόλλου MACSec για να αποκτήσουμε βαθύτερη κατανόηση σχετικά με τα κενά ασφαλείας των δικτύων τύπου LAN.

Abstract

The various applications of a Local Area Network (LAN) have incontestably render it an important and integral part of many contemporary facilities including schools, universities, and office buildings. Nowadays, a lot of threats attempt to downgrade the network security, while enterprises' current attention is directed to internal data breaches. Considerable research studies are carried out in order to apprehend how effective are the security protocols against these threats. In this thesis, a study about MACSec protocol, a promising relatively new security protocol is conducted to (a) explore how MACSec protects the session of two machines against an internal attacker, (b) investigate its relation with IPSec, and (c) empirically assess the overhead of MACSec protocol in common services like file transfer.

For the purpose of the study, a MACSec session is established between two virtual machines (VMs). Subsequently, some packets of data are exchanged between them and a third virtual machine with a role of attacker connected to the same LAN attempts to eavesdrop and capture the network traffic. Later, we enable the FTP and HTTP on the machines and capture the exchanged packets again using the attacking machine. Finally, we copy three files with different sizes between the VMs and record the time of the transfer when the MACSec session is enabled or not.

The results reveal that the MACSec protocol effectively secures the session of the two VMs and the attacker cannot access any private data. In addition, the FTP functions through the secure session of MACSec protocol, while naturally the HTTP is not related to the session of the MACSec protocol and the packets of data are vulnerable to internal attacks. Finally, we notice that the MACSec session prolongs the time of files transfer. We conclude that the MACSec is efficient in protecting against an internal attacker attempting to monitor packets traffic and that different protocols either support or are independent of the MACSec protocol. A complete assessment must be fulfilled in tandem with investigating external attacks, immediate attacks and the cooperation of the MACSec with other protocols. The proposed approach is a first step towards a complete evaluation of the efficacy of the MACSec protocol to gain insight in security gaps of LAN.

Acknowledgements

Working in the accomplishment of this thesis was an interesting let alone demanding journey. A lot of dedication, persistence and knowledge pursuit was required, but the efforts eventually bear fruit. Primarily I would thank God for being able to complete this project with success.

Then I would like to express my sincere thanks to my supervisor Professor Georgios Kambourakis whose teaching skills, kind guidance, encouragement, patience, suggestion and very constructive criticism have contributed immensely to the evolution of my ideas on this thesis and shaped the present work in its current form.

I express my deep sense of gratitude to all the teaching staff for supplying me with all the necessary knowledge during my undergraduate and postgraduate studies.

One more person I would like to especially acknowledge with thanks is Ms Anastasia Douma, which I had the honor to collaborate with during my postgraduate studies. Both her knowledge and her experience were important in helping and inspiring me along the various requirements of the master.

In addition, I am immensely obliged to my family for their perpetual support, encouragement and moral values.

Last but not least, I would like to personally thank Nikolaos Chaikalis, Charalampos Katsis, Panagiotis Pachymanolis, Stefanos Plastras, Olena Tymkiv, Flora Simou, Elena Chatzoula who made the demanding time joyful, but always efficient.

List of Figures

List of Acronyms

- **ARP**: Address Resolution Protocol
- **CAK**: Connectivity Association Key
- **CKN**: Connectivity Association Key Name
- **DHCP**: Dynamic Host Configuration Protocol
- **EAP**: Extensible Authentication Protocol
- **FQDN**: Fully Qualified Domain Name
- **FTP**: File Transfer Protocol
- **GCM-AES**: Galois/Counter Mode - Advanced Encryption Standard
- **GPL**: GNU General Public License
- **HTTP**: Hypertext Transfer Protocol
- **HTTPS**: Hypertext Transfer Protocol Secure
- **IANA**: Internet Assigned Numbers Authority
- **ICMP**: Internet Control Message Protocol
- **ICV**: Integrity Check Value
- **IEEE**: Institute of Electrical and Electronics Engineers
- **IPSec**: Internet Protocol Security
- **ISO**: International Organization for Standardization
- **LACP**: Link Aggregation Control Protocol
- **LANs**: Local Area Networks
- **LACP**: Link Aggregation Control Protocol
- **LLDP**: Link Layer Discovery Protocol
- **LTE**: Long-Term Evolution
- **MACSec**: Media Access Control Security
- **MKA**: MACSec Key Agreement Protocol

- **MKPDU**: MACSec Key Agreement Protocol Data Unit
- **OSI**: Open Systems Interconnection model
- **PING**: Packet Internet Grouper
- **SA**: Security Association
- **SAK**: Secure Association Key
- **SCs**: Secure Channels
- **SCP**: Secure Copy
- **SCI**: Secure Channel Identifier
- **SDBN**: Software-Defined Bridged Networks
- **SecTAG**: Security TAG
- **SFTP**: SSH File Transfer Protocol
- **SSH**: Secure Shell protocol
- **SSL**: Secure Sockets Layer
- **STP**: Spanning Tree Protocol
- **TCP**: Transmission Control Protocol
- **TLS**: Transport Layer Security
- **TTL**: Time To Live
- **UDLD**: Unidirectional Link Detection
- **VM**: Virtual Machine
- **VPN**: Virtual Private Network
- **VXLAN**: Virtual Extensible LAN

Contents

Chapter 1

Introduction

1.1 Background and Context

The increasing necessity for use of numerous computers in residences as well as in facilities such as schools, universities, laboratories, libraries and office buildings generated the demand for high-speed interconnections between computer systems. A Local Area Network (LAN) is a computer network that interconnects computers within a limited area facilitating the connection of computers and other devices that are physically close to each other. So far, LAN has proved to be very useful in providing every device with access to the Internet through a single broadband connection, security in transmission of data between devices since packets of data transmitted locally never leave the LAN, and finally access to shared resources such as printers, scanners, file servers, mail servers and network attached storage [1].

However, as all computer networks, LAN is not risk-free. Until recently, in enterprises high priority has been given to network security aiming to prevent external threats to data. Even though firewalls can protect effectively against external attacks, what is gaining concern today is that a major source of a large number of threats has been redirected to data and privacy breaches internal to the enterprise [2]. Internal attacks with access in sensitive information, knowing how that information is protected and without leaving evidence of intrusion, make it difficult to be detected and prevented using one-size-fits-all security measures. If we take into account that these insider threats can be accidental and unintentional and may result in release of secure, private, confidential information to an untrusted environment, facilitating an outsider attacker to benefit from these vulnerabilities and gaining access to these information, as occured in the Target data breach [3] it is easily inferred that there are a lot of internal threats [4] that limit the security of the system, greatly necessitating a network security behind external firewalls.

Up to date there a lot of cryptographic protocols serving this purpose, acting in different ways and at different stages. The Open Systems Interconnection model (OSI model) is a conceptual model proposed by the International Organization for Standardization (ISO) that divides the network communication into seven layers, implementing in each layer these protocols. Yet, in the Internet many of the existing cryptographic protocols are limited, one of them, the TCP/IP protocol that has prevailed, doesn't offer any security guarantees, while the majority of the protocols currently used are private, except for the TLS and the IPSec, the later predominately used in VPNs [5]. Instead, a protocol arisen that is gaining popularity is the Media Access Control Security (MACSec) protocol, a standard IEEE 802.1AE security technology that provides security communication for all network traffic over Ethernet connections. It belongs to the Layer 2 of OSI and is responsible to provide secure network access, verify the origin of data through protected channels and optionally encrypt traffic between the host and the access switch. Since encryption-based security is increasingly applicable to organizations and governments, MACSec providing such a network-wide encryption-based security stands out as the most relevant means to achieve such a level of security [6], [7], [8], [9].

1.2 Motivation

The inspiration for writing this thesis was drawn primarily from the increasing demand of enterprises for high level security that ensures safe packets traffic against insider threats. Emphasis has been given in security protocols while a great deal of interest has been aroused about the MACSec protocol, a promising cryptographic protocol that ensures communication to authorized endpoints on the local network. We mainly wanted to examine the efficacy of MACSec protocol security in packets traffic against an internal attacker, the behavior of the protocols FTP and HTTP on machines with a MACSec session as well as the interference of the MACSec protocol with the time for transferring files of different sizes.

1.3 Scope and Objectives

The first goal is to determine the level of security in the session of packets traffic between two machines when a third internal attacker monitors and captures the packets.

The second goal is to examine as a case study if both the FTP and HTTP take advantage of the MACSec protocol and are secured against an internal attacker.

The third goal is to evaluate the time of files transfer between two machines when the MACSec is enabled and not.

1.4 Thesis Structure

The thesis at hand is structured as follows. In the following chapter, we provide a thorough description of the MACSec protocol. In Chapter 3, we go into details about the FTP and HTTP. In Chapter 4 we explicate the methodology we follow. In the last chapters we provide an analysis of the results of our investigation, discuss our findings, and draw a conclusion.

Chapter 2

MACSec Protocol

2.1 Introduction

The Media Access Control Security (MACSec) protocol as defined by IEEE 802.1AE, allows authorized systems to connect LANs to a network, maintain metadata confidentiality and take action against frameworks transmitted or modified by unauthorized devices. It belongs to Level 2 of OSI, is a hop-by-hop[10] encryption and provides encryption in wired networks using out-of-band[11] methods for key encryption. The MACSec Key Agreement (MKA) protocol, as specified in IEEE Std 802.1X, provides the required key management. Only links that belong to hosts can be secured using MACSec [12].

In more detail, to ensure confidentiality and integrity of a network traffic, MACSec is based on GCM-AES (Galois/Counter Mode - Advanced Encryption Standard)[13] and provides point-to-point security over Ethernet connections between nodes that are in the same LAN. It can protect the IP traffic, the Address Resolution Protocol (ARP), the Dynamic Host Configuration Protocol (DHCP) and is capable of detecting and preventing network security threats, including denial of service (DoS), intrusion, man-in-the-middle, masquerading, passive wiretapping, and playback attacks. One of its main functions is to secure communication from and to the endpoints at the access edge and is usually used with the 802.1X protocol which provides port-based authentication and transmits the necessary key to the host and switch. The 802.1X is a protocol that locates MACSec-enabled users and dynamically distributes the key to them [12], [6], [7], [8].

The MACSec protocol can be used in cooperation with other security protocols, such as Internet Protocol Security (IPSec) and Secure Sockets Layer (SSL), to provide end-to-end network security and it can provide Ethernet link for almost all traffic, including Link Layer Discovery Protocol (LLDP), Link Aggregation Control Protocol (LACP), DHCP, ARP and other protocols that are usually not secured on Ethernet due to security policies [7].

In MACSec, packets are channeled through “secure channels”, which are supported by “secure links” and each of the secure links uses a separate, randomly generated key. The IEEE 802.1X defines a MKA companion protocol, which makes the exchange of keys and allows the mutual recognition of nodes wishing to join in a MACSec association connection. On Linux machines, MKA is applied to wpa_supplicant. The wpa_supplicant uses a unique authentication token, which is a shared key or a username/password pair similar to the authentication mechanisms used in IEEE 802.11 (WiFi), in order to create a session with the authentication server, which can be a switch or a Linux host running hostapd. After the authentication is completed, the keys are generated and exchanged via the encrypted channel and are used to configure a secure MACSec link [6].

2.2 Architecture of MACSec network

Each node in a network that is protected by MACSec, has at least one secure transmission channel connected to an identifier called Secure Channel Identifier (SCI). This secure channel stores various configuration parameters, such as whether it will be protected by repetitions or whether encryption will be enabled [6].

Each node that has a transmit secure channel must also form a corresponding receive secure channel. The receive secure channel should have an SCI that corresponds to the SCI of the transmit secure channel of the peer [6].

Within any secure transmission and reception channel, some secure correlations are identified. Secure associations retain the encryption keys and are identified by their connection number. One other important parameter, which is associated with any secure association, is the packet number. On the transmission side, this number of packets is placed in the MACSec header and is used in the encryption process. On the receiving side, the packet number from the MACSec header can be checked with the packet number who is locally stored in the corresponding secure connection to perform replication protection [6].

2.3 How MACSec Works

The MACSec provides security via the use of secure point-to-point Ethernet interfaces between MACSec-enabled interfaces. If the user wants to enable MACSec on multiple Ethernet connections, they must configure MACSec on each point-to-point Ethernet connection individually. These interfaces are ensured by exchanging and verifying the security keys corresponding to the interfaces at each end of the point-to-point Ethernet connector. The key can be configured by the user or can be dynamically generated, depending on the security function used to enable the MACSec. Other parameters which can be configured by the user, such as MAC address or port, must match to the interfaces on each side of the link so as to enable the MACSec. Once MACSec is enabled on a point-to-point Ethernet connection, all the traffic crossing the connection is MACSec-secured through the use of data integrity checks and it is encrypted [7].

The integrity of data is verified by the data integrity checks. The MACSec adds to all Ethernet frames that cross the MACSec-secured point-to-point Ethernet link, a 8-byte header and a 16-byte tail. The header and tail are checked by the receiving interface to ensure that the data was not compromised while traversing the link. If any irregularities are found in the data integrity check, the traffic is reduced [7].

The MACSec can be used to ensure that all network traffic on the Ethernet frame is encrypted, so the data in the Ethernet box cannot be viewed by anyone who can monitor the traffic on the connector. The encryption of MACSec is optional and can be configured by the user [7].

2.4 How MACSec handles data and control traffic

On an active MACSec port, all traffic is controlled and the data is encrypted or the integrity is protected or both. In case a MACSec session is not secured, then all data and the control traffic are omitted [9].

If MACSec is enabled on a port, then the port blocks data traffic and data traffic is not forwarded until a MACSec session is secured. If a continuous session is interrupted, the traffic on the port is blocked until a new secure session is created [9].

The traffic control, including Spanning Tree Protocol (STP), Link Aggregation Control Protocol (LACP), or Unidirectional Link Detection (UDLD) traffic is not transmitted from an active MACSec port until a MACSec session is ensured. If a session is created, from the port only 802.1X packets protocol will be transmitted and once a secure session is created, the flow is normally controlled through the port [9].

2.5 MACSec Key Agreement protocol

The MKA is installed on one device based on an IEEE 802.1X Extensible Authentication Protocol (EAP) framework in order to create communication [12], [9].

The MACSec peers who are located on the same local network, belong to a unique connectivity association. Since they constitute members of the same connectivity association, they can identify themselves with a shared Connectivity Association Key (CAK) and a Connectivity Association Key Name (CKN). The CAK is a static key that is predefined in every MACSec-enabled interface. The MACSec authentication is based on the mutual ownership and acknowledgment of the preconfigured CAK and Connectivity Association Key Name (CKN) [12], [9].

Each peer device through the connectivity association establishes a single unidirectional secure channel where it transmits the MACSec frames, i.e., Ethernet frames with MACSec headers that usually carry encrypted data, to each peer. A connectivity association consists of two secure channels, one for inbound and one for outbound traffic. All peers within the connectivity association use the same encryption suite, currently either Galois/Counter Mode Advanced Encryption Standard 128 or 256 (GCM-AES-128 or GCM-AES-256), for MACSec-certified security features [12], [9].

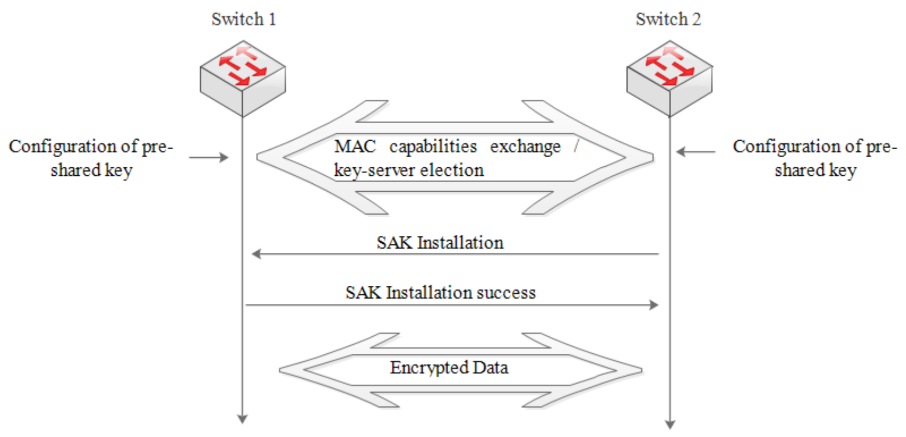
The MACSec Key Agreement employs the Connectivity Association Key to derive transient session keys called Secure Association Keys (SAKs). The SAKs and other MKA parameters are necessary to keep the communication secure through the secure channel and to perform encryption and other MACSec security functions. The SAKs, along with other basic control information are distributed in MKA protocol control packets, also referred to as MKPDUs [12], [9].

2.6 MKA message exchange between two switches

The MKA protocol will start key-server elections as soon as two MACSec peers confirm that they own a shared CAK and CKN [9].

To determine the type of data encryption as well as whether MACSec encryption is used, the key-server is important. Also, it is responsible for creating SAKs as well as distributing them to the connected device. Once a SAK is successfully installed, both devices can exchange data securely [9].

The following Figure 2.1 illustrates the message flow between two switches during MACSec communication [9].


Switch 1
Switch 2
Configuration of pre-shared key
MAC capabilities exchange / key-server election
Configuration of pre-shared key
SAK Installation
SAK Installation success
Encrypted Data

key.jpg key.jpg

Figure 2.1: MKA pre-shared key and key name exchange between two network switches. [9]

2.7 Secure channels

Every channel that wishes to communicate safely over MACSec, implements temporary sessions namely correlations. These sessions can only be created with a unique SAK assigned to the session [9].

After the transmission of a certain number of frames or after peer disconnection and reconnection, the secure associations expires and need to be re-established [9].

The SCI in combination with an Association Number (AN), forms the Secure Association Identifier (SAI), which is responsible for determining a secure association. When a peer interface receives a MACSec frame, the device from the SAI recognizes the session key that is transferred to the MACSec frame and uses the key to decrypt and authenticate the received frame [9].

2.8 Scenarios of MACSec

2.8.1 MACSec LAN setup for multiple secure channels

One of the main uses of MACSec is the assurance of standard LAN. In this setting, multiple machines connected to the same LAN have been configured in a way that all the packets exchanged between them are secured and can be received only from these nodes [6], [14].

In Figure 2.2, the red link indicates that packets are not protected by MACSec. Instead, blue links are protected by MACSec [6], [14].

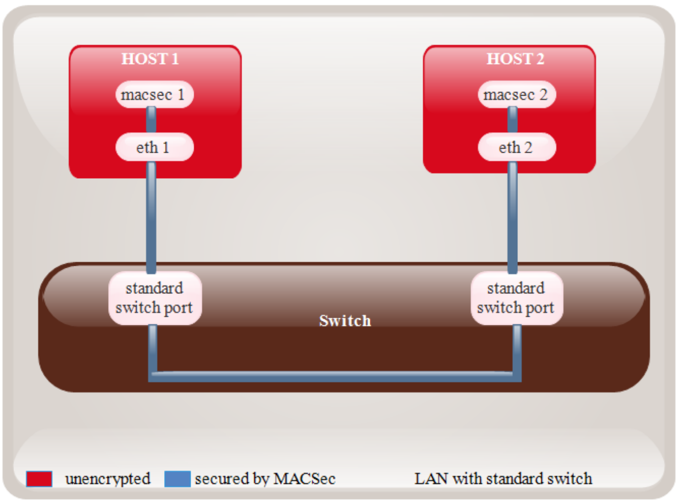


Figure 2.2: Example LAN setup with a standard switch. [6]

In this first installation, the switch is not capable of encrypting frames, but can forward the MACSec frames which are protected between the ports. The MACSec terminates on the hosts [6], [14].

An alternative solution is to use a switch that supports MACSec. In this case, MACSec is enabled on the client machines as well as on the switch ports these machines are connected to. These access switches often provide 802.1X services to allow robust certification, authorization, and accounting before finally allowing the client to join the network [6], [14].

In Figure 2.3, the red link illustrates the packets that are not protected by MACSec. Instead, blue links are provided by MACSec [6], [14].

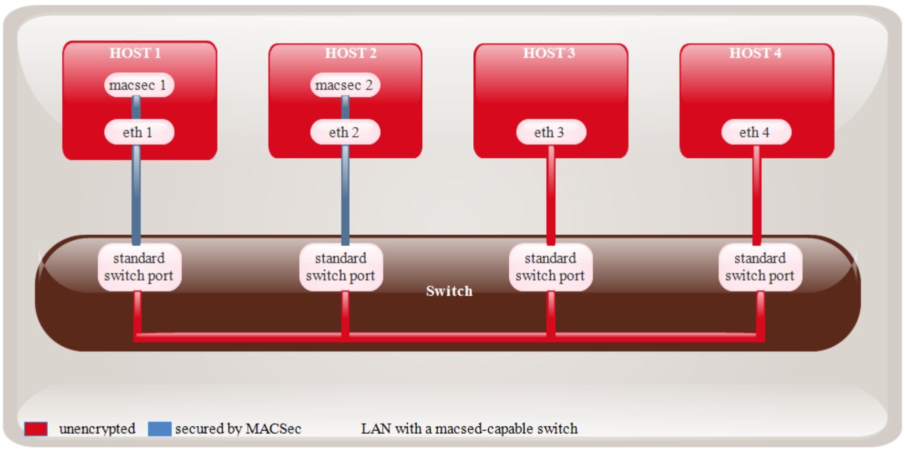


Figure 2.3: Example LAN setup with a MACSec-capabe switch. [6]

The second of these LAN settings uses a MACSec-enabled switch, but hosts 3 and 4 do not use MACSec. The MACSec terminates on the switch ports. [6], [14]

2.8.2 MACSec Virtual Extensible LAN (VXLAN) setup

The MACSec is compatible with Virtual Extensible LAN (VXLAN) and other tunneling technologies such as GENEVE[15] and GRETAP[16]. A cloud client with a virtual private LAN can use MACSec to encrypt all internal traffic before leaving the VM. In this way, the cloud provider cannot access the communication between the VMs as it is shown in Figure 2.4 [6], [14].

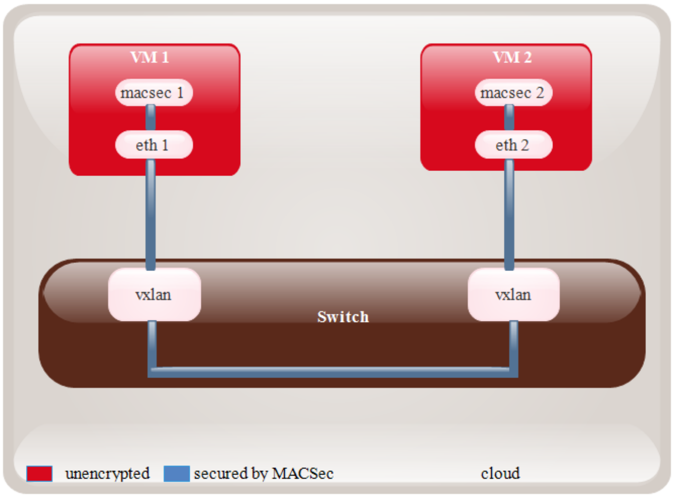


Figure 2.4: Example Cloud topology with VXLAN. [6]

2.9 Benefits and Limitations

The MACSec offers services over wired networks and some of the benefits are the follow [17]:

- **Confidentiality**: The MACSec is responsible for ensuring the confidentiality of data at the network level (Layer 2 of OSI) by providing strong encryption.

- **Integrity**: The MACSec provides data integrity checks, ensuring that data cannot be modified while in transit.
- **Flexibility**: The user can selectively enable MACSec by using a centralized policy, thereby helping to ensure that MACSec is enforced where required, while also allowing access to a network that is not MACSec enabled.
- **Network intelligence**: The MACSec encrypts packets on a hop-by-hop basis at the network level (Layer 2 of OSI), enabling the network to monitor traffic in line with the existing policies.

Although MACSec offers excellent data security, it has some limitations as listed below [17]:

- **Endpoint support**: Not all endpoints support MACSec.
- **Hardware support**: Line-rate encryption usually requires updated hardware on the access switch.
- **Technology integration**: Enabling MACSec may affect the functions of other technologies also connected to the access edge, such as IP telephony. Understanding and adapting these technologies is essential for successful development.

2.10 Comparison to IPSec security protocol

The MACSec is a security protocol gaining more and more use in computer machines. In contrast, the IPSec is a security protocol suite which has been utilised and tested for more than 20 years [18], with a currently extensive bibliography concerning its efficacy and limitations. Therefore, in this thesis we decided to make a comparison between the MACSec and IPSec protocols.

The IPSec, known as Internet Protocol Security or IP Security Protocol, is a secure protocol suite that operates at the network level (Layer 3 of OSI) and practically certifies and encrypts data packets that are transmitted over the Internet Protocol network. It has been designed to provide security at the IP level and this is done through the certification and encryption of IP network packets. It includes protocols that define the cryptographic algorithms that are used to encrypt, decrypt and authenticate packets, as well as protocols which are required for secure key exchange and key management [19]. IPSec may be used in three different security domains: virtual private networks, application-level security and routing security. At this time, IPSec is predominately used in VPNs [5].

With the MACSec and IPSec, the user applications do not need any modification to benefit from the security guarantees provided by these standards. However, the IPSec is preferred for data encryption on WAN/MAN[20] networks as its allows the transfer of a large set of options from the private services VPN of Layer 3 to the public Internet. Although it resolves most of the encryption requirements, one of its limitations is its performance. This is due to the fact that IPSec is associated with high-speed links that require line-rate performance regardless of the packet size and network speed [6], [21].

One main difference is that the MACSec belongs to the Layer 2 of the OSI model and makes link encryption, while the IPSec belongs to the Layer 3 of the OSI model and offers IP encryption. In the MACSec, the link encryption in Layer 2 is done in the same Layer of the OSI. In the Layer 2, the link encryption is used in any of the selected technologies. The applications and protocols above Layer 2 (Layers 3-7 of OSI) are transparent and part of the payload being encrypted. Because the encryption of the Layer 2 is known for the link-layer aware, it is dependent on the Layer 2 transport chosen for the network design, thus utilising the forwarding of the encrypted frame performed in the transport of choice [21].

In the IPSec, the encryption in the Layer 3 operates at the IP layer of the OSI model, encrypting the data packets in the IP protocol. The IPSec offers packet-based encryption. With the Layer 3, the encryption between endpoints is not limited to Layer 2 technology, but uses IP allowing the transfer of IP for connection to the endpoints. The encryption at the Layer 3 is much more flexible because it can take advantage of any packet-based IP transfer. Instead, the encryption at the Layer 2 is limited to the specific Layer 2 transport and does not normally cover a wide footprint network. The encryption to the Layer 3 does not know the Layer 2 transport networks/links that the packet has to cross [21].

The MACSec can protect all DHCP and ARP network traffic, while IPSec cannot ensure it. However, IPSec can operate on routers, while MACSec is limited to a local area network [6].

The IPSec provides security using end-to-end tunnels and is complex, while MACSec supports easy upgrades and high-speed connectivity up to 100G at low power consumption and low cost [22].

The IPSec provides security using end-to-end tunnels that are encrypted. One major disadvantage of IPSec is its complexity. Except that it includes a dedicated encryption machine, it greatly enlarges the size of the Ethernet header [22].

The MACSec can be scaled linearly with the number of links in hop-by-hop scenarios as well as with the number of endpoints in end-to-end applications, while IPSec can support a certain amount of total capacity and a specific number of tunnels per network port [22].

However, the two protocols are compatible with each other and can be complementary. The MACSec with tag and flow can enhance the IPSec on two levels. Firstly, in network equipment that can be extremely expensive, it can be converted to something that is based on MACSec only. Secondly, considering wireless network security at the small cell level, the last mile-link between the small cell and the central office should no longer be IPSec, it could be purely MACSec [22].

Chapter 3

Protocols

3.1 FTP and HTTP protocols

As already pointed out, the MACSec protocol operates at layer 2, on ethernet frames. The data Link Layer is responsible for the final encapsulation of higher-level messages into frames that are sent over the network at the physical layer [2]. Both the FTP and HTTP function at Layer 7.

3.1.1 FTP

One of the first uses of the Internet was to transfer files between computers. The File Transfer Protocol (FTP) is an early form created to move files from one device to another quickly and instantly. It has its roots back in 1971, when the first version was created and published by Abhay Bhushan. In the 1980s, the FTP form was updated to the server-associated TCP/IP version. The FTP is used to remotely access computers, register shared files and upload or download files between local and remote computers. The FTP operates on ports 20, 21 [23], [24].

The FTP is based on the client-server model architecture using separate control and data connections between the client and the server. The FTP users can authenticate themselves with a direct connection protocol, usually in the form of a username and password, but can login anonymously if the server is configured to allow it. For secure transmission that protects the username and password and encrypts the content, FTP is often secured with TLS (FTPS) or replaced with the SSH File Transfer Protocol (SFTP) [25].

The FTP uses two basic channels in order to operate. The one channel is the command channel that transmits information about the task itself (what files are to be accessed if commands are entered, etc.) and the second channel is the data channel that transfers the actual file data between of devices [24].

The FTP connections can have both active and passive modes. The active modes are the most common since they allow open communication between the server and the device through both channels. The role of server to the communication is to establish the connection by approving data requests. However, this operation can be altered due to firewalls and similar issues, so there is a passive mode where the server pays attention but does not keep the connections active, allowing the other device to do all the work [24].

Because FTP transfers are not encrypted, it is easy for the attacker to eavesdrop on the traffic. For this reason, many users use FTPS. This practically works in the same way as FTP, but it encrypts application layer information, meaning that malicious users cannot read any files, even if they could watch them. At this point, many servers refuse to offer unencrypted access and instead offer only FTPS [24].

Communication and data transfer

The FTP can be executed in active or passive mode, which determines how the data connection is established. In both cases as it is illustrated in Figure 3.1, the client establishes a TCP control connection from a random, usually non-privileged, N port to the FTP server's command port 21 [25].

- In active mode, a client who want to start listening to incoming data connections from the server on port M, sends the FTP command PORT M to inform the server on which port it will be listening. Then, the server enters a data channel to the client from port 20, the FTP server data port.
- Passive mode can be used in cases where a client is behind a firewall and is unable to accept incoming TCP connections. The client uses the control connection to send a PASV command to the server and then receives a server IP address and a server port number from the server, which the client then uses to open a data connection from an arbitrary port client in the server IP address and server port number received.

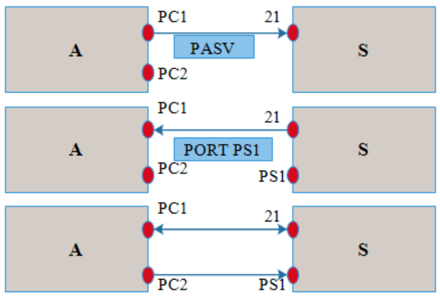


Figure 3.1: Starting an FTP passive connection using port 21. [25]

3.1.2 HTTP

The Hypertext Transfer Protocol (HTTP) which frequently called internet protocol, is an application protocol designed in the context of the internet protocol suite and used in distributed, collaborative and hypermedia information systems. Practically, it is the database of web communication, where hypertext documents include hyperlinks to other resources that the user can easily access. The HTTP has been given this definition because the most traffic to the Internet is HTTP-based and furthermore, it requires a reliable transport layer protocol, namely the Transmission Control Protocol (TCP) [23], [26].

When a user requests a Web resource, it is requested using HTTP. When a client enters this address in a Web browser, DNS is prompted to resolve the Fully Qualified Domain Name (FQDN) to an IP address. When the address is resolved, an HTTP request is sent to the Web server. The Web server responds by sending an HTTP response. This communication is often done in a single session on a Web site. The HTTP uses the TCP to communicate between clients and servers. Typically, the HTTP listens to port 80 [23], [26].

The HTTP protocol operates as a request-response between client-server on the computer model. The client can be a web browser and the server an application that runs on a computer hosting a site. The client submits an HTTP request message to the server which provides various resources (HTML files and other content or performs other functions on behalf of the client) and returns a response message to the client. The response may contain the content requested in the message body as well as information on completing the status of the request [26].

The HTTP is designed to allow network interfaces to improve or enable communication between clients and servers. The high-traffic websites often benefit from web cache web servers that provide content for upstream servers to improve response time. The web browser cache has previously accessed internet resources and reused them, where possible, to reduce network traffic [26].

Table of OSI		
7	Aplication	ECHO, ENRP, FTP, Gopher, HTTP, NFS, RTSP, SIP, SMTP, SNMP, SSH, Telnet, Whois, XMPP, etc.
6	Presentation	XDR, ASN.1, SMB, AFP, NCP, TLS, SSL, SSH, etc.
5	Session	ASAP, ISO 8327 / CCITT X.225, RPC, NetBIOS, ASP, etc.
4	Transport	TCP, UDP, RTP, SCTP, SPX, ATP, IL
3	Network	IP, ICMP, IGMP, IPX, OSPF, RIP, IGRP, EIGRP, IPSec, ARP, RARP, X.25, etc.
2	Data Link(Device driver)	Ethernet, Token ring, MAC, HDLC, Frame, relay, ISDN, ATM, 2802.11, FDDI, PPP, etc.
1	Physical(Network adapter)	10BASE-T, 100BASE-T, 10000BASE-T, SONET/SDH, G.7-9, T-carrier/E-carrier, various 802.11 physical layers, etc.

Chapter 4

Methodology and Experimentation

4.1 Context of the study

From a theoretical point of view, the work described in this thesis is carried out to explore the way the MACSec protocol protects the session of packets traffic between computers connected to the LAN against an internal attacker also by definition connected to the same LAN and the way the MACSec protocol influences the time of transferring files. From the experimental side, the evaluation process is conducted in two VMs simulating two computers that both support the MACSec protocol and communicate with each other. A Third VM simulating the attacker is connected to the same LAN and attempts to monitor and capture the packets of data transferred between the MACSec - enabled VMs (called First and Second VMs in the following).

4.2 Configuring the MACSec protocol

For the purpose of the study Ubuntu 18.04.1 is installed on two VMs generated by the VMware Workstation Player, a virtualization software package distributed free of charge by VMware, Inc. allowing to run one or more isolated operating systems on a single computer. In addition, Kali Linux 2018.4 is installed on a Third VM also generated by the VMware Workstation Player.

Ubuntu is a complete operating system based on various versions of Linux. It is freely available and is extremely advanced in terms of technology and design. Its most important advantage is simplicity and ease of use. In addition to the pre-installed basic applications it includes, it provides many programs that cover specialized needs [27], [28].

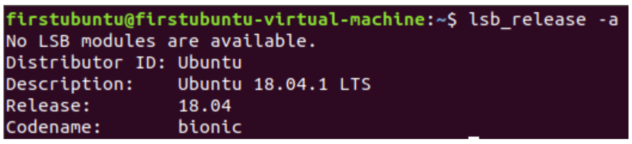
Kali Linux is a Debian-based Linux distribution that contains over 300 security testing tools aimed at numerous information security tasks including penetration testing, computer forensics, security research and reverse engineering. This advanced penetration testing and security auditing tool was released in 2013 as a rebuild of BackTrack Linux and is maintained by Offensive Security, a leader in information security training. It is regarded as the one of the most Ethical Hacking/Pentesting distribution available. Kali Linux allows the user to utilise similar tools and techniques that a hacker would use to check the security of the network so that they can find and correct these issues before a real hacker finds them [29], [30].

Regarding the First and Second VMs, in order for the MACSec protocol to be enabled in the Ubuntu operating system of both VMs it is necessary that kernel supports it. Some versions of Ubuntu have the kernel, but the MACSec field is either not enabled or not included, thus it is important to upgrade the kernel. However, in the Ubuntu version 18.04.1 the kernel had already the field MACSec enabled right from the start.

Specifically, the kernel is a program that forms the central core of a computer's operating system and has the complete control all over the systems. Usually, it is one of the programs which is loaded first at startup (after bootloader) on most systems and deals with startup management as well as software input/output requests, converting them into data processing instructions for the central processing unit. Furthermore, it manages memory, keyboards, monitors, printers, and speakers [31]. MACSec is part of the Linux kernel since version 4.6 [6] so that even switch-to-host or host-to-host links can be protected.

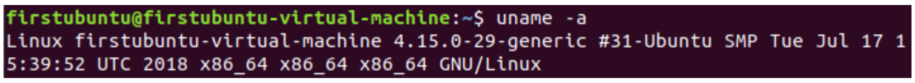
In Figure 4.1 and Figure 4.2 it is shown the versions of Ubuntu and kernel respectively we used.

- **lsb_release -a**
- **uname -a**



```
firstubuntu@firstubuntu-virtual-machine:~$ lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 18.04.1 LTS
Release:        18.04
Codename:       bionic
```

Figure 4.1: Ubuntu 18.04.1 version



```
firstubuntu@firstubuntu-virtual-machine:~$ uname -a
Linux firstubuntu-virtual-machine 4.15.0-29-generic #31-Ubuntu SMP Tue Jul 17 1
5:39:52 UTC 2018 x86_64 x86_64 x86_64 GNU/Linux
```

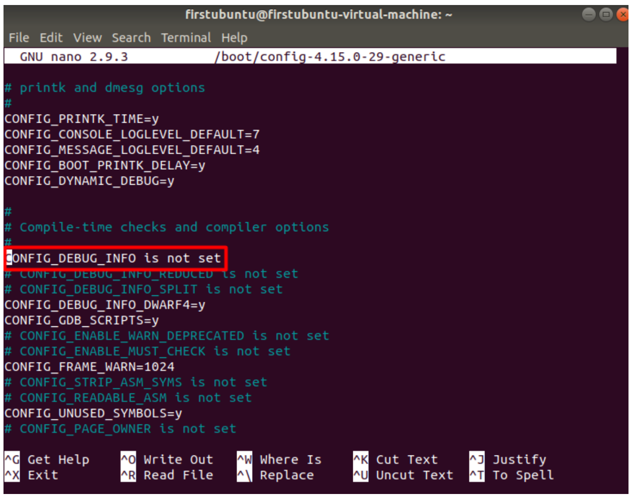
Figure 4.2: Kernel version supported by Ubuntu 18.04.1 version

Later, on both the First and Second VM we need to enable the fields **'CONFI_DEBUG_INFO'** and **'CONFIG_MACSec'** to the .config file **'config-4.15.0-29-generic'** that is located in the path **/boot/**. This is important in order to compile the MACSec module support. In detail, the steps used to enable the fields are the following, also illustrated in Figures 4.3 and 4.4.

- **sudo nano /boot/config-$(uname -r)**

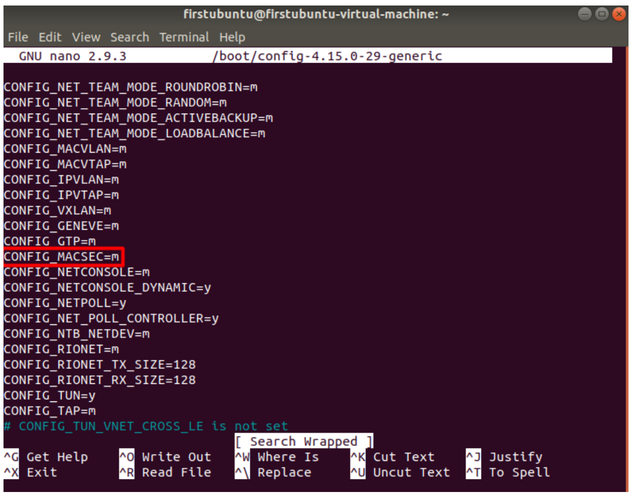
Press ctrl + W to search the fields into the .config file
# speed up building
CONFIG_DEBUG_INFO is not set
# enable MACSec support
CONFIG_MACSec=m

- **ctr+X**
- **y**
- **Enter**


firstubuntu@firstubuntu-virtual-machine: ~
File Edit View Search Terminal Help

```
  GNU nano 2.9.3             /boot/config-4.15.0-29-generic

# printk and dmesg options
#
CONFIG_PRINTK_TIME=y
CONFIG_CONSOLE_LOGLEVEL_DEFAULT=7
CONFIG_MESSAGE_LOGLEVEL_DEFAULT=4
CONFIG_BOOT_PRINTK_DELAY=y
CONFIG_DYNAMIC_DEBUG=y

#
# Compile-time checks and compiler options
#
CONFIG_DEBUG_INFO is not set
# CONFIG_DEBUG_INFO_REDUCED is not set
# CONFIG_DEBUG_INFO_SPLIT is not set
CONFIG_DEBUG_INFO_DWARF4=y
CONFIG_GDB_SCRIPTS=y
# CONFIG_ENABLE_WARN_DEPRECATED is not set
# CONFIG_ENABLE_MUST_CHECK is not set
CONFIG_FRAME_WARN=1024
# CONFIG_STRIP_ASM_SYMS is not set
# CONFIG_READABLE_ASM is not set
CONFIG_UNUSED_SYMBOLS=y
# CONFIG_PAGE_OWNER is not set

^G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify
^X Exit       ^R Read File  ^\ Replace    ^U Uncut Text ^T To Spell
```

Figure 4.3: Unset CONFIG_DEBUG_INFO to speed up time and decrease size on disk.


firstubuntu@firstubuntu-virtual-machine: ~
File Edit View Search Terminal Help

```
  GNU nano 2.9.3             /boot/config-4.15.0-29-generic

CONFIG_NET_TEAM_MODE_ROUNDROBIN=m
CONFIG_NET_TEAM_MODE_RANDOM=m
CONFIG_NET_TEAM_MODE_ACTIVEBACKUP=m
CONFIG_NET_TEAM_MODE_LOADBALANCE=m
CONFIG_MACVLAN=m
CONFIG_MACVTAP=m
CONFIG_IPVLAN=m
CONFIG_IPVTAP=m
CONFIG_VXLAN=m
CONFIG_GENEVE=m
CONFIG_GTP=m
CONFIG_MACSEC=m
CONFIG_NETCONSOLE=m
CONFIG_NETCONSOLE_DYNAMIC=y
CONFIG_NETPOLL=y
CONFIG_NET_POLL_CONTROLLER=y
CONFIG_NTB_NETDEV=m
CONFIG_RIONET=m
CONFIG_RIONET_TX_SIZE=128
CONFIG_RIONET_RX_SIZE=128
CONFIG_TUN=y
CONFIG_TAP=m
# CONFIG_TUN_VNET_CROSS_LE is not set
                              [ Search Wrapped ]
^G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify
^X Exit       ^R Read File  ^\ Replace    ^U Uncut Text ^T To Spell
```

Figure 4.4: Enabling the MACSec

Finally, we also need to install the iproute2 in the First and Second VMs. The iproute2 is an open source project that is based on the terms of version 2 of the GNU General Public License (GPL) and its development depends on the development of the Linux kernel networking components. It consists of a collection of utilities that are used to control and monitor various types of networking in the Linux kernel such as routing, network interfaces, tunnels, traffic control and network device drivers [32].

Iproute2 is used for traffic control and the command line utilities are: IP, ss, bridge, rtacct, rtmon, tc, ctstat, lnstat, nstat, routef, routel, rtstat, tipc, arpd and devlink. tc. The iproute2 utilities communicate with the kernel of Linux using the netlink protocol [32].

The steps for the installation are the following.

- **sudo apt-get update**
- **apt-get install iproute2**
- **reboot**

4.3 Steps for establishing a secure session

After having the MACSec protocol enabled, to ensure that a secure session has been established between the First and Second VM, we have to create and exchange MKA keys. To configure the MACSec protocol so as to allow the creation of keys, a series of important steps for the First VM needs to be made:

- we hide the real IP address,
- we load the MACSec kernel,
- we create a variable 'macsec0' of type macsec on top of the physical address (MAC) of the First VM,
- we create a random key,
- we load the physical address of the Second VM that the First VM talks to,
- we load again the physical address of the Second VM that the First VM talks to, securing this session with the key of the Second VM,
- we enable the macsec0,
- we define a fake IP address for the First VM,
- we handle the machine so that the process is encrypted with MACSec.

We repeat the same process for the Second VM.

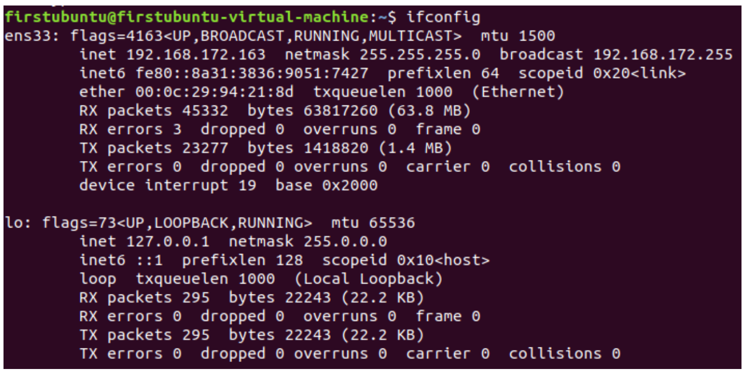
We noticed that this version of Ubuntu did not have some tools installed as it did not support the ifconfig command. So, before we proceed we needed to get them installed:

- **sudo apt install net-tools**

On the First Virtual Machine

Before configuring the MACSec, we need to see the real IP address of the First VM, as it is illustrated in Figure 4.5. The real IP address of the First VM is: 192.168.172.163.

- **ifconfig**



```
firstubuntu@firstubuntu-virtual-machine:~$ ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.172.163  netmask 255.255.255.0  broadcast 192.168.172.255
        inet6 fe80::8a31:3836:9051:7427  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:94:21:8d  txqueuelen 1000  (Ethernet)
        RX packets 45332  bytes 63817260 (63.8 MB)
        RX errors 3  dropped 0  overruns 0  frame 0
        TX packets 23277  bytes 1418820 (1.4 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 19  base 0x2000

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 295  bytes 22243 (22.2 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 295  bytes 22243 (22.2 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Figure 4.5: Real IP address of the First VM before configuring the MACSec.

# Clear IP configuration on the Host-Only adaptor between the VMs

- **sudo ifconfig ens33 0.0.0.0**
- **ifconfig**

# Load the MACSec kernel

- **sudo modprobe macsec**

# Create the MACSec device on top of the physical one

- **sudo ip link add link ens33 macsec0 type macsec**

# Configure the Transmit SA and keys

- **sudo ip macsec add macsec0 tx sa 0 pn 100 on key 01 11111111111111111111111111111111**

# Configure the Receive Channel and SA:
# Physical address (MAC) of the peer (Second VM)
# port number, packet number and key (Second VM)

- **sudo ip macsec add macsec0 rx address 00:0c:29:8e:22:19 port 1**
- **sudo ip macsec add macsec0 rx address 00:0c:29:8e:22:19 port 1 sa 0 pn 100 on key 02 22222222222222222222222222222222**

# Bring up the interface

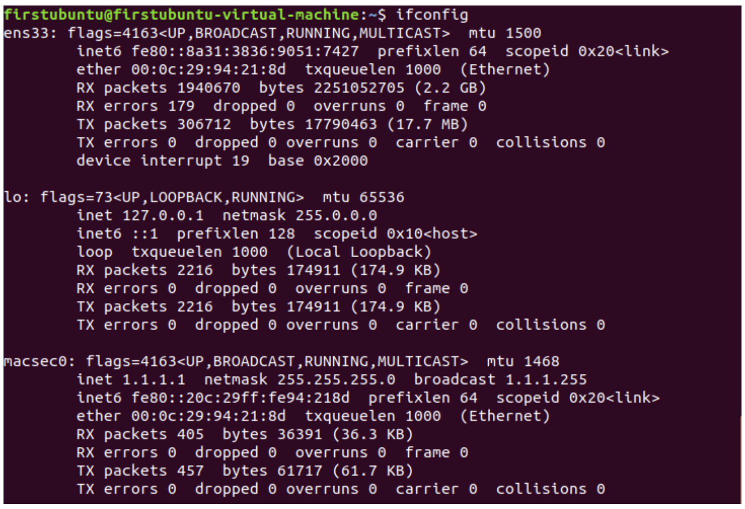
- **sudo ip link set dev macsec0 up**

# Configure a fake IP address on it for connectivity between the hosts

- **sudo ifconfig macsec0 1.1.1.1/24**

# We type the command ‘ifconfig’ to see the fake IP address of the MACSec protocol as it is illustrated in Figure 4.6.

- **ifconfig**



```
firstubuntu@firstubuntu-virtual-machine:~$ ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::8a31:3836:9051:7427  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:94:21:8d  txqueuelen 1000  (Ethernet)
        RX packets 1940670  bytes 2251052705 (2.2 GB)
        RX errors 179  dropped 0  overruns 0  frame 0
        TX packets 306712  bytes 17790463 (17.7 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 19  base 0x2000

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 2216  bytes 174911 (174.9 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2216  bytes 174911 (174.9 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

macsec0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1468
        inet 1.1.1.1  netmask 255.255.255.0  broadcast 1.1.1.255
        inet6 fe80::20c:29ff:fe94:218d  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:94:21:8d  txqueuelen 1000  (Ethernet)
        RX packets 405  bytes 36391 (36.3 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 457  bytes 61717 (61.7 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Figure 4.6: Fake IP 1.1.1.1 address of MACSec in the First VM.

# Enable encryption:

- **sudo ip link set macsec0 type macsec encrypt on**

<u>**On the Second Virtual Machine**</u>

We follow exactly the same steps as in the First VM.

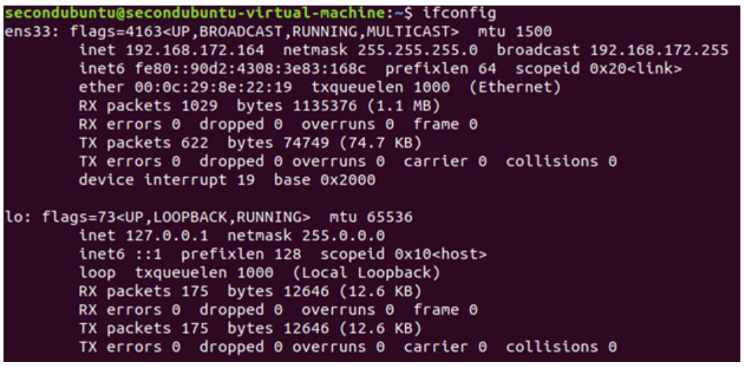
Before configuring the MACSec, we need to see the real IP address of the Second VM, as it is illustrated in Figure 4.7. The real IP address of the First VM is: 192.168.172.164.

- **ifconfig**

# Clear IP configuration on the Host-Only adaptor between the VMs

- **sudo ifconfig ens33 0.0.0.0**
- **ifconfig ens33**

# Load the MACSec kernel



```
secondubuntu@secondubuntu-virtual-machine:~$ ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.172.164  netmask 255.255.255.0  broadcast 192.168.172.255
        inet6 fe80::90d2:4308:3e83:168c  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:8e:22:19  txqueuelen 1000  (Ethernet)
        RX packets 1029  bytes 1135376 (1.1 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 622  bytes 74749 (74.7 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 19  base 0x2000

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 175  bytes 12646 (12.6 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 175  bytes 12646 (12.6 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Figure 4.7: Real IP address of the Second VM before configuring the MACSec.

- **sudo modprobe macsec**

# Create the MACSec device on top of the physical one

- **sudo ip link add link ens33 macsec0 type macsec**

# Configure the Transmit SA and keys

- **sudo ip macsec addmacsec0 tx sa 0 pn 100 on key 02 222222222222222222222222222222**

# Configure the Receive Channel and SA:
# Physical address (MAC) of the peer (First VM)
# port number, packet number and key (First VM)

- **sudo ip macsec add macsec0 rx address 00:0c:29:94:21:8d port 1**

- **sudo ip macsec add macsec0 rx address 00:0c:29:94:21:8d port 1 sa 0 pn 100 on key 01 111111111111111111111111111111**

# Bring up the interface

- **sudo ip link set dev macsec0 up**

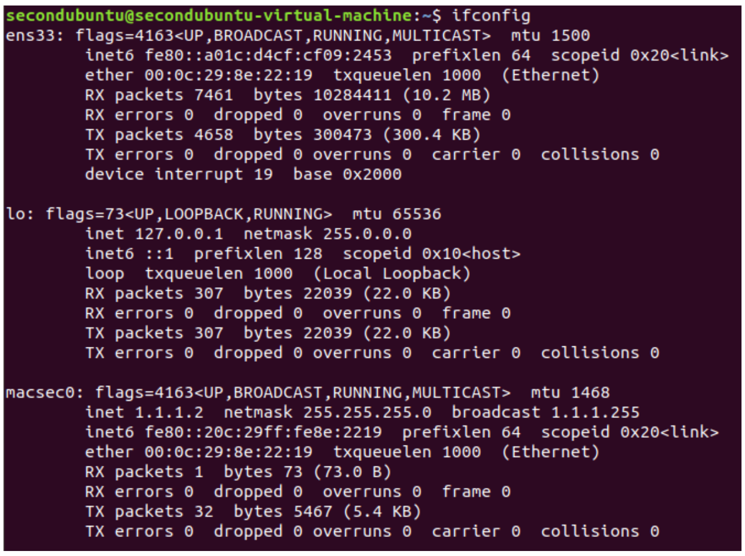
# Configure a fake IP address on it for connectivity between the hosts

- **sudo ifconfig macsec0 1.1.1.2/24**

# We type the command 'ifconfig' to see the fake IP address of the MACSec protocol as it is illustrated in Figure 4.8.

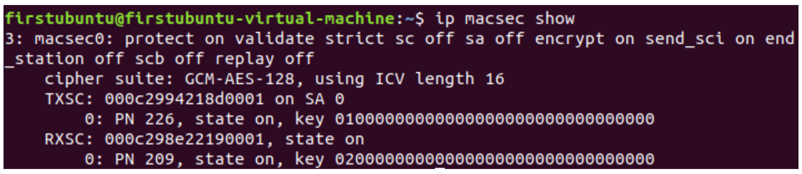
- **ifconfig**

# Enable encryption:



```
secondubuntu@secondubuntu-virtual-machine:~$ ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::a01c:d4cf:cf09:2453  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:8e:22:19  txqueuelen 1000  (Ethernet)
        RX packets 7461  bytes 10284411 (10.2 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 4658  bytes 300473 (300.4 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 19  base 0x2000

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 307  bytes 22039 (22.0 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 307  bytes 22039 (22.0 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

macsec0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1468
        inet 1.1.1.2  netmask 255.255.255.0  broadcast 1.1.1.255
        inet6 fe80::20c:29ff:fe8e:2219  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:8e:22:19  txqueuelen 1000  (Ethernet)
        RX packets 1  bytes 73 (73.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 32  bytes 5467 (5.4 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Figure 4.8: Fake IP 1.1.1.2 address of MACSec in the Second VM.



```
firstubuntu@firstubuntu-virtual-machine:~$ ip macsec show
3: macsec0: protect on validate strict sc off sa off encrypt on send_sci on end
_station off scb off replay off
    cipher suite: GCM-AES-128, using ICV length 16
    TXSC: 000c2994218d0001 on SA 0
        0: PN 226, state on, key 01000000000000000000000000000000
    RXSC: 000c298e22190001, state on
        0: PN 209, state on, key 02000000000000000000000000000000
```

Figure 4.9: Connection between the First and Second VM with the keys.

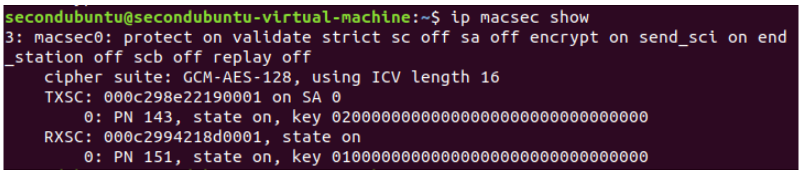
- **sudo ip link set macsec0 type macsec encrypt on**

<u>In the First VM Figure 4.9</u>

- **ip macsec show**

<u>In the Second VM Figure 4.10</u>

- **ip macsec show**



```
secondubuntu@secondubuntu-virtual-machine:~$ ip macsec show
3: macsec0: protect on validate strict sc off sa off encrypt on send_sci on end
_station off scb off replay off
    cipher suite: GCM-AES-128, using ICV length 16
    TXSC: 000c298e22190001 on SA 0
        0: PN 143, state on, key 02000000000000000000000000000000
    RXSC: 000c2994218d0001, state on
        0: PN 151, state on, key 01000000000000000000000000000000
```

Figure 4.10: Connection between Second and First VM with the keys.

4.4 Sniffing on LAN traffic

For the course of this investigation and in order to capture the packets data transmitted between the First and the Second VMs so as to test if the machines support the MACSec, if the communication in both of them is encrypted and therefore, if they are secured against the attacker, we utilise an important separate tool, wireshark.

Wireshark is a network packet analyzer used to capture packet data thoroughly in the network medium. This tool can record traffic from many different types of network media such as Ethernet, wireless LAN, Bluetooth, USB and many more. The specific types of media who are supported may be limited by a variety of factors including the hardware and the operating system [33], [34].

In the past, many packet analysis tools were either very expensive or proprietary or both. However, with the appearance of wireshark this changed. The original name coined was Ethereal, but in May of 2006 it changed to Wireshark for trademark reasons. Wireshark is one of the best packet analysts available today. It is an open source software project released by the GNU General Public License and it is available for free. It can be used on any number of computers without the user having to worry about keys or license fees. In addition, all source code is freely available within the GPL. Because of this, it is very easy for different users to add new protocols to wireshark, either as plugins or embedded in the source [33], [34].

Below we present some of the features that wireshark offers:

- Available for Windows, Linux, Mac OS X, Solaris and BSD software.
- Record in real time packets data from a network interface.
- Uses GTK + for the desktop and Pcap for packets capture.
- Opening files containing packets data recorded with tcpdump/WinDump, wireshark and many other packet capture programs.
- Display packets with very detailed protocol information.
- Save packet data captured.
- Export some or all of the packets to various record file formats.
- Multiple criteria filtering.
- Multiple criteria packets search.

4.5 Research Design

For the context of this study, we estimated the security of MACSec protocol in three consecutive steps: i) Verification of the established session between the First and Second VM, ii) Capturing of the exchanged packets by the attacking Third VM, iii) Installation of FTP and HTTP on First and Second VM and capturing of the exchanged packets by the attacking Third VM. Later, we proceeded to an assessment of the time of files transfer between the First and Second VM when the MACSec is disabled and enabled.

4.5.1 Verification of established session between the First and Second VM

First off, before we proceed to any assessment of the level of security provided by MACSec protocol against an internal attacker, we wanted to confirm that the MACSec session has been properly established.

For this reason we install the wireshark tool in the First VM.

- **sudo apt-get update**
- **sudo apt install wireshark**

Subsequently, from the side of the First VM, we execute the command 'ping' via terminal to check if the Second VM is active.

- **ping 1.1.1.2**

Then, we open a new terminal from the side of the First VM and run the wireshark so as to see the packets traffic in both VMs.

- **sudo wireshark**

4.5.2 Capturing of the exchanged packets by the attacking Third VM

In the next step, we make use of a Third VM with Kali Linux operating system performing the role of an internal attacker by capturing the packets traffic between the First and the Second VM.

At first, we install the wireshark on the Third VM.

- **sudo apt install wireshark**

In the First VM we execute for one more time the command 'ping'.

- **ping 1.1.1.2**

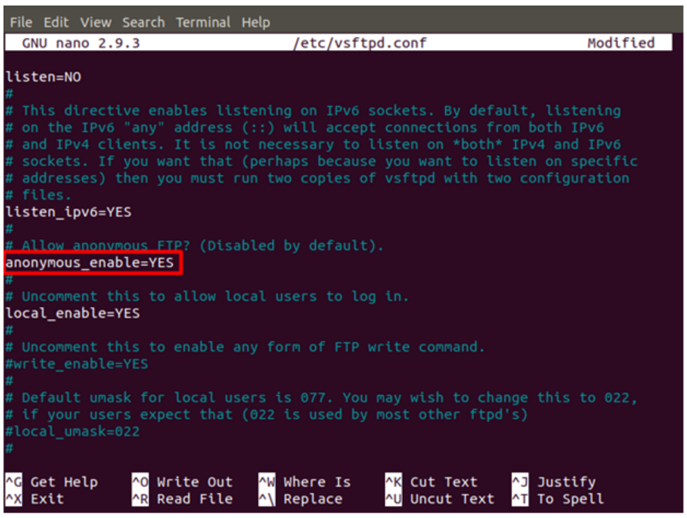
Afterwards from the side of the Third VM, we open a new terminal and run the wireshark in order to see the packets traffic between the First and Second VM.

4.5.3 Installation of FTP and HTTP on the First and Second VM and capturing of the exchanged packets by the attacking Third VM

File Transfer Protocol (FTP)

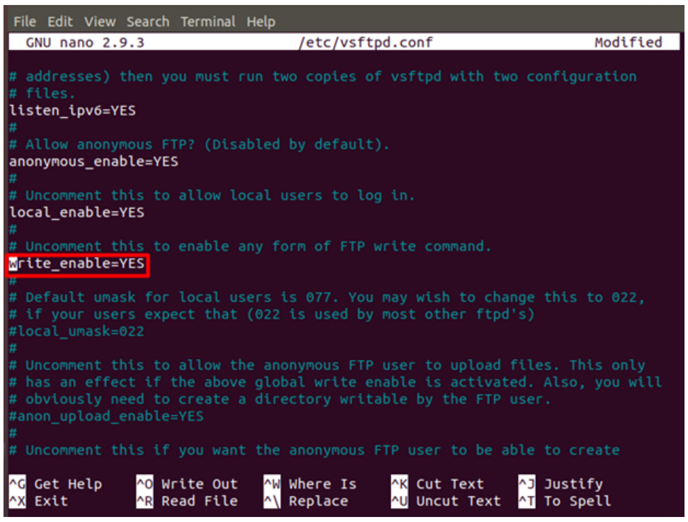
Installation of FTP:

- **sudo apt-get update**
- **sudo apt-get install vsftpd**



```
File  Edit  View  Search  Terminal  Help
  GNU nano 2.9.3                    /etc/vsftpd.conf                    Modified

listen=NO
#
# This directive enables listening on IPv6 sockets. By default, listening
# on the IPv6 "any" address (::) will accept connections from both IPv6
# and IPv4 clients. It is not necessary to listen on *both* IPv4 and IPv6
# sockets. If you want that (perhaps because you want to listen on specific
# addresses) then you must run two copies of vsftpd with two configuration
# files.
listen_ipv6=YES
#
# Allow anonymous FTP? (Disabled by default).
anonymous_enable=YES
#
# Uncomment this to allow local users to log in.
local_enable=YES
#
# Uncomment this to enable any form of FTP write command.
#write_enable=YES
#
# Default umask for local users is 077. You may wish to change this to 022,
# if your users expect that (022 is used by most other ftpd's)
#local_umask=022
#

^G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify
^X Exit       ^R Read File  ^\ Replace    ^U Uncut Text ^T To Spell
```

Figure 4.11: Enable in file vsftpd.conf the field anonymous_enable.



```
File  Edit  View  Search  Terminal  Help
  GNU nano 2.9.3                    /etc/vsftpd.conf                    Modified

# addresses) then you must run two copies of vsftpd with two configuration
# files.
listen_ipv6=YES
#
# Allow anonymous FTP? (Disabled by default).
anonymous_enable=YES
#
# Uncomment this to allow local users to log in.
local_enable=YES
#
# Uncomment this to enable any form of FTP write command.
write_enable=YES
#
# Default umask for local users is 077. You may wish to change this to 022,
# if your users expect that (022 is used by most other ftpd's)
#local_umask=022
#
# Uncomment this to allow the anonymous FTP user to upload files. This only
# has an effect if the above global write enable is activated. Also, you will
# obviously need to create a directory writable by the FTP user.
#anon_upload_enable=YES
#
# Uncomment this if you want the anonymous FTP user to be able to create

^G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify
^X Exit       ^R Read File  ^\ Replace    ^U Uncut Text ^T To Spell
```

Figure 4.12: Enable in file vsftpd.conf the field write_enable.

#We wish to enable anonymous mode in FTP, so we need to edit /etc/vsftpd.conf by changing as illustrated in Figures 4.11, 4.12.

- **sudo nano /etc/vsftpd.conf**

- **sudo systemctl restart vsftpd**
- **sudo systemctl enable vsftpd**
- **sudo systemctl status vsftpd**
- **/etc/init.d/vsftpd start**

At the Third VM, we use the wireshark tool to capture the packets traffic between the First and the Second VM as long as the FTP has been enabled in the First and Second VMs

- **sudo wireshark**

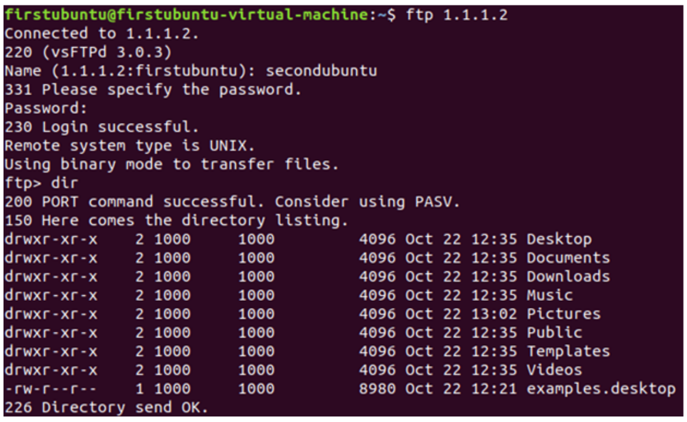
At side of the First VM, we connect via the FTP with the Second VM. The fake IP of the Second VM is the 1.1.1.2, as already mentioned.

- **ftp 1.1.1.2**

- secondubuntu
- 123456

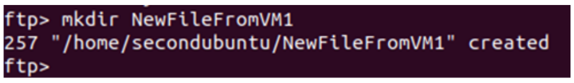
After the connection has been achieved, in the First VM having remotely access to the space of the Second VM, we type the command "dir" to see the list of files and directories included in the Second VM. Later, in this way we create a directory in the Second VM with the name NewFileFromVM1, and then, we type again the command dir so as to see the directory created.

- dir
- mkdir NewFileFromVM1
- dir



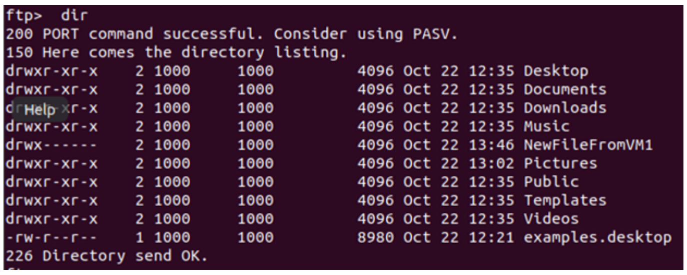
```
firstubuntu@firstubuntu-virtual-machine:~$ ftp 1.1.1.2
Connected to 1.1.1.2.
220 (vsFTPd 3.0.3)
Name (1.1.1.2:firstubuntu): secondubuntu
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> dir
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
drwxr-xr-x    2 1000     1000         4096 Oct 22 12:35 Desktop
drwxr-xr-x    2 1000     1000         4096 Oct 22 12:35 Documents
drwxr-xr-x    2 1000     1000         4096 Oct 22 12:35 Downloads
drwxr-xr-x    2 1000     1000         4096 Oct 22 12:35 Music
drwxr-xr-x    2 1000     1000         4096 Oct 22 13:02 Pictures
drwxr-xr-x    2 1000     1000         4096 Oct 22 12:35 Public
drwxr-xr-x    2 1000     1000         4096 Oct 22 12:35 Templates
drwxr-xr-x    2 1000     1000         4096 Oct 22 12:35 Videos
-rw-r--r--    1 1000     1000         8980 Oct 22 12:21 examples.desktop
226 Directory send OK.
```

Figure 4.13: Connection of the First to the Second VM via the FTP service.



```
ftp> mkdir NewFileFromVM1
257 "/home/secondubuntu/NewFileFromVM1" created
ftp>
```

Figure 4.14: Create a dir in the Second VM using the First VM via the FTP service.



```
ftp>  dir
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
drwxr-xr-x    2 1000     1000         4096 Oct 22 12:35 Desktop
drwxr-xr-x    2 1000     1000         4096 Oct 22 12:35 Documents
d[illegible]r-x    2 1000     1000         4096 Oct 22 12:35 Downloads
drwxr-xr-x    2 1000     1000         4096 Oct 22 12:35 Music
drwx------    2 1000     1000         4096 Oct 22 13:46 NewFileFromVM1
drwxr-xr-x    2 1000     1000         4096 Oct 22 13:02 Pictures
drwxr-xr-x    2 1000     1000         4096 Oct 22 12:35 Public
drwxr-xr-x    2 1000     1000         4096 Oct 22 12:35 Templates
drwxr-xr-x    2 1000     1000         4096 Oct 22 12:35 Videos
-rw-r--r--    1 1000     1000         8980 Oct 22 12:21 examples.desktop
226 Directory send OK.
```

Figure 4.15: All dirs in the Second VM, both the previously existed and the new one created, as observed using the First VM.

At the Second VM, we now type the command 'ls' to check the list with all the files/directories in the Second VM as it is illustrated in Figure 4.16.

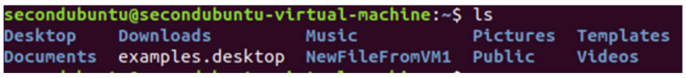

```
secondubuntu@secondubuntu-virtual-machine:~$ ls
Desktop    Downloads          Music          Pictures  Templates
Documents  examples.desktop   NewFileFromVM1 Public    Videos
```

Figure 4.16: All dirs in the Second VM, both the previously existed and the new one created, as observed using the Second VM.

Finally, the Third VM running in parallel with the aforementioned process captures the packets traffic using the wireshark tool.

Hypertext Transfer Protocol (HTTP)

Installation of HTTP:

- **sudo apt install apache2**
- **/etc/init.d/apache2 start**

At the First VM, we execute the 'ping' command on an online website, specifically google.com.

- **ping google.com**

Simultaneously, the Third VM captures the packets traffic between the First VM and the website.

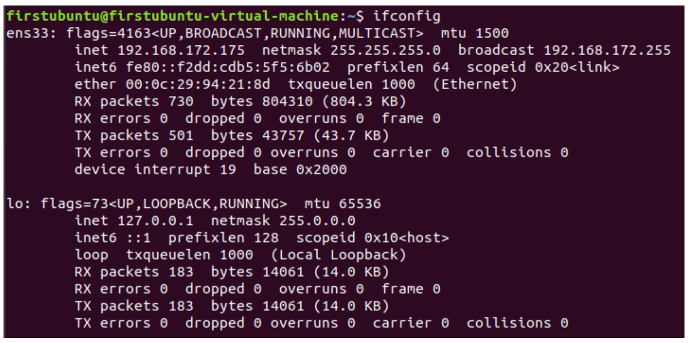
- **sudo wireshark**

In the next step, we download a website using the First VM and check the web traffic as captured by the Third VM via wireshark tool.

- **cd Download**
- **mkdir everydaylinuxuser**
- **cd everydaylinuxuser**
- **wget -r www.everydaylinuxuser.com**

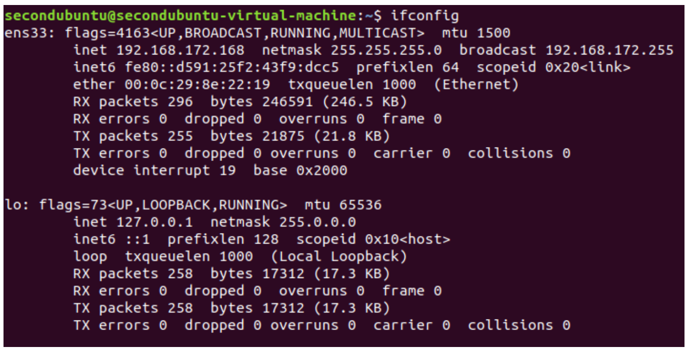
4.5.4 Time of files transfer between the First and Second VM when the MACSec is disabled and enabled

In this step, we decided to follow the same process for creating new VMs from the beginning. Hence, the new real IP address for the First VM is 192.168.172.175 and for the Second VM is 192.168.168 as it is illustrated in Figures 4.17, 4.18.



```
firstubuntu@firstubuntu-virtual-machine:~$ ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.172.175  netmask 255.255.255.0  broadcast 192.168.172.255
        inet6 fe80::f2dd:cdb5:5f5:6b02  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:94:21:8d  txqueuelen 1000  (Ethernet)
        RX packets 730  bytes 804310 (804.3 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 501  bytes 43757 (43.7 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 19  base 0x2000

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 183  bytes 14061 (14.0 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 183  bytes 14061 (14.0 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Figure 4.17: IP of the First VM



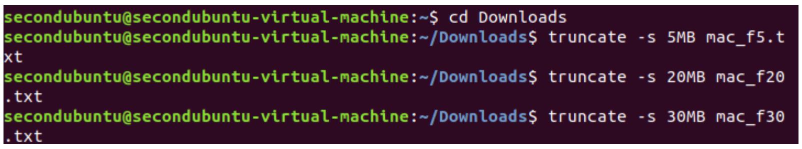
```
secondubuntu@secondubuntu-virtual-machine:~$ ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.172.168  netmask 255.255.255.0  broadcast 192.168.172.255
        inet6 fe80::d591:25f2:43f9:dcc5  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:8e:22:19  txqueuelen 1000  (Ethernet)
        RX packets 296  bytes 246591 (246.5 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 255  bytes 21875 (21.8 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 19  base 0x2000

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 258  bytes 17312 (17.3 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 258  bytes 17312 (17.3 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Figure 4.18: IP of the Second VM

At first, we create in the Second VM three consecutive files with sizes 5MB, 20MB and 30MB.

For the creation of the files, we type the following commands:

- **cd Downloads**
- **truncate -s 5MB mac_f5.txt**
- **truncate -s 20MB mac_f20.txt**
- **truncate -s 30MB mac_f30.txt**



```
secondubuntu@secondubuntu-virtual-machine:~$ cd Downloads
secondubuntu@secondubuntu-virtual-machine:~/Downloads$ truncate -s 5MB mac_f5.t
xt
secondubuntu@secondubuntu-virtual-machine:~/Downloads$ truncate -s 20MB mac_f20
.txt
secondubuntu@secondubuntu-virtual-machine:~/Downloads$ truncate -s 30MB mac_f30
.txt
```

Figure 4.19: Creation of files with size 5MB, 20MB, 30MB.

We install afterwards the SSH protocol in the First and Second VM in order to enable the copy of files from the Second to the First VM.

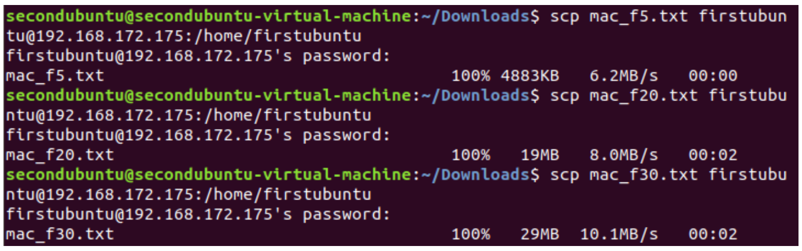
- **sudo apt-get update**
- **sudo apt-get install openssh-server**

At the First VM, we use the wireshark tool to record the time needed for files transfer.

- **sudo wireshark**

Then, when the MACSec is not enabled, we copy the files to the First VM typing the following commands:

- **scp mac_f5.txt firstubuntu@192.168.172.175:/home/firstubuntu**
- **scp mac_f20.txt firstubuntu@192.168.172.175:/home/firstubuntu**
- **scp ma_f30.txt firstubuntu@192.168.172.175:/home/firstubuntu**

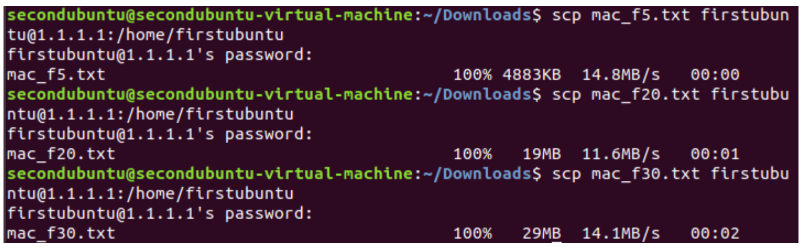


```
secondubuntu@secondubuntu-virtual-machine:~/Downloads$ scp mac_f5.txt firstubun
tu@192.168.172.175:/home/firstubuntu
firstubuntu@192.168.172.175's password:
mac_f5.txt                                    100% 4883KB   6.2MB/s   00:00
secondubuntu@secondubuntu-virtual-machine:~/Downloads$ scp mac_f20.txt firstubu
ntu@192.168.172.175:/home/firstubuntu
firstubuntu@192.168.172.175's password:
mac_f20.txt                                   100%   19MB   8.0MB/s   00:02
secondubuntu@secondubuntu-virtual-machine:~/Downloads$ scp mac_f30.txt firstubu
ntu@192.168.172.175:/home/firstubuntu
firstubuntu@192.168.172.175's password:
mac_f30.txt                                   100%   29MB  10.1MB/s   00:02
```

Figure 4.20: File transfer when the MACSec session is disabled.

Finally, we repeat the same steps when the MACSec is enabled, copying the same files from the Second to the First VM using the following commands:

- **scp mac_f5.txt firstubuntu@1.1.1.1:/home/firstubuntu**
- **scp mac_f20.txt firstubuntu@1.1.1.1:/home/firstubuntu**
- **scp mac_f30.txt firstubuntu@1.1.1.1:/home/firstubuntu**



```
secondubuntu@secondubuntu-virtual-machine:~/Downloads$ scp mac_f5.txt firstubun
tu@1.1.1.1:/home/firstubuntu
firstubuntu@1.1.1.1's password:
mac_f5.txt                                    100% 4883KB  14.8MB/s   00:00
secondubuntu@secondubuntu-virtual-machine:~/Downloads$ scp mac_f20.txt firstubu
ntu@1.1.1.1:/home/firstubuntu
firstubuntu@1.1.1.1's password:
mac_f20.txt                                   100%   19MB  11.6MB/s   00:01
secondubuntu@secondubuntu-virtual-machine:~/Downloads$ scp mac_f30.txt firstubu
ntu@1.1.1.1:/home/firstubuntu
firstubuntu@1.1.1.1's password:
mac_f30.txt                                   100%   29MB  14.1MB/s   00:02
```

Figure 4.21: File transfer when the MACSec session is enabled.

Chapter 5

Analysis and Results

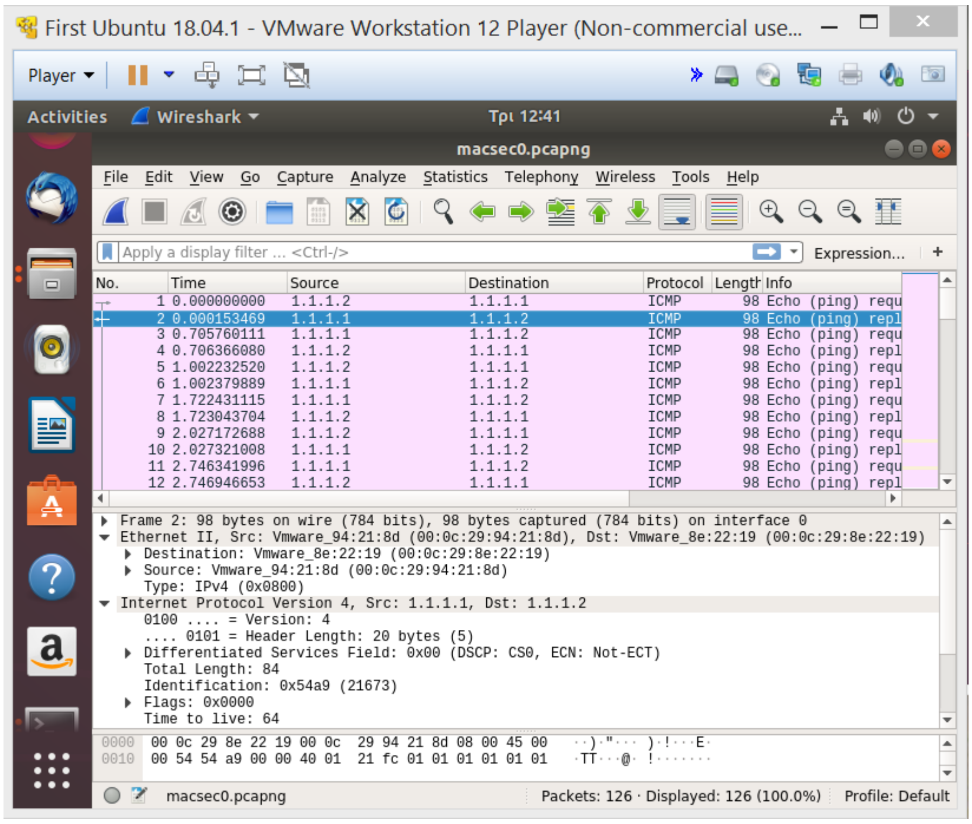
The results of the commands and recording of packets capture are presented below and commented as following:

In the first step of packets exchange between the First and Second VM, it is confirmed that the session between the First and Second VM is established, since as it is illustrated in Figure 5.1 the 'ping' command is executed successfully using the fake IPs 1.1.1.1 for the First VM and 1.1.1.2 for the Second VM as defined in the MACSec session. The ping utility however makes use of the ICMP, and thus we verify that the ICMP supports the session of the MACSec protocol. In addition, the Figure 5.2 and 5.3 indicate in each VM that both the fake and real IP addresses are converted via ARP to a common physical address (MAC), namely Vmware_94:21:8d for the First VM and Vmware_8e:22:19 for the Second VM. Therefore, we verify that the ARP also supports the MACSec protocol.

In the second step of capturing the exchanged packets between the First and Second VM by an attacking Third VM, as it is illustrated in Figure 5.4 the internal attacker is able to track only the physical addresses of the First and Second VMs and not the 'ping' command between those VMs or the IP addresses, and thus the communication between them is protected by the MACSec protocol. The same applies when the real IP addresses are retrieved automatically after a while and we repeat the same process, with the attacking Third VM being able to capture only the physical addresses as illustrated in Figure 5.5.

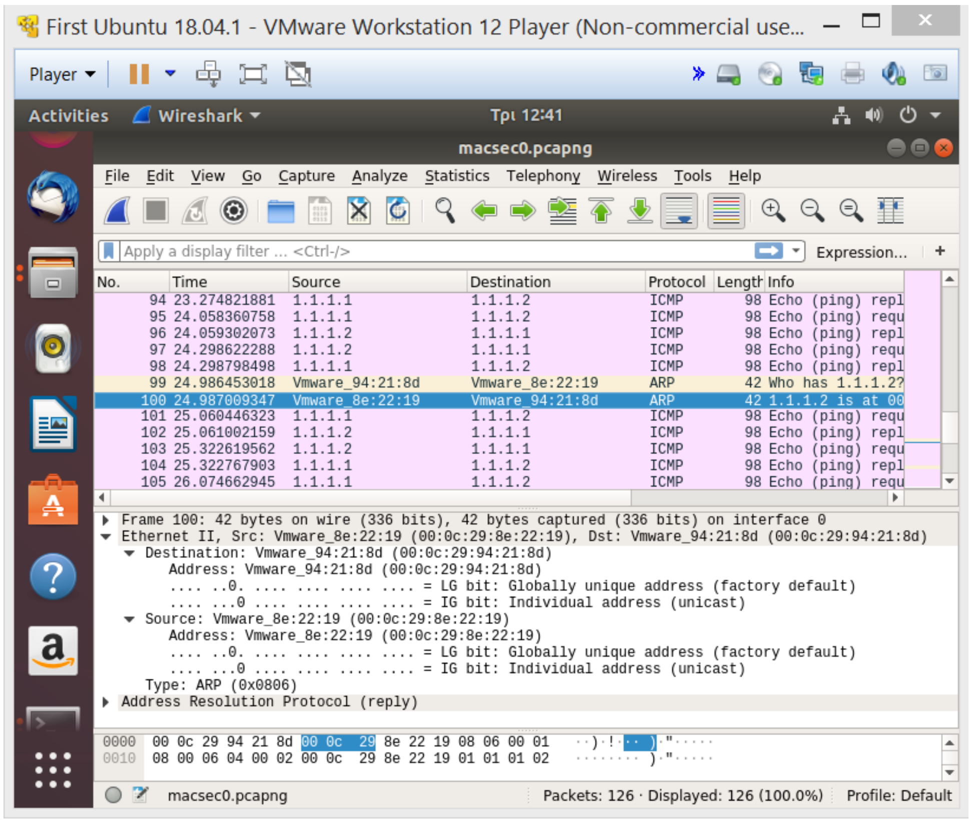
In the third step of installation of the FTP on the First and Second VM and capture of the exchanged packets by the attacking Third VM, in Figure 5.7 the packets tracked by the attacker reveal only the physical addresses of both the First and Second VMs and no information about the creation of directory in the space of the Second VM or about the IP addresses. The same occurs when the real IP addresses are retrieved automatically after a while and we repeat the same process, with the attacking Third VM being able to capture only the physical addresses as illustrated in Figure **??**

In the next step, after the installation of the HTTP on the First VM, when we execute the 'ping' command to the website google.com with the fake IP address, while the real IP of the First VM is hidden, we are not able to access the Internet.


First Ubuntu 18.04.1 - VMware Workstation 12 Player (Non-commercial use...
Player
Activities
Wireshark
Τρι 12:41
macsec0.pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...

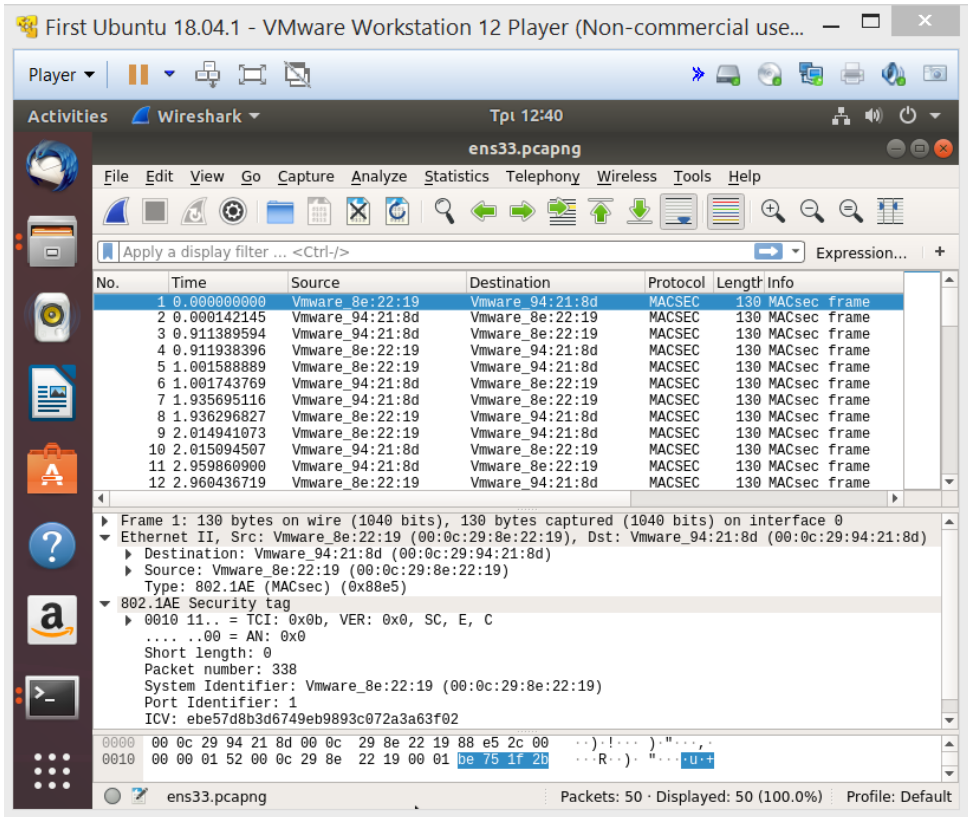
```
No.  Time         Source    Destination  Protocol Length Info
   1 0.000000000  1.1.1.2   1.1.1.1      ICMP     98 Echo (ping) requ
   2 0.000153469  1.1.1.1   1.1.1.2      ICMP     98 Echo (ping) repl
   3 0.705760111  1.1.1.1   1.1.1.2      ICMP     98 Echo (ping) requ
   4 0.706366080  1.1.1.2   1.1.1.1      ICMP     98 Echo (ping) repl
   5 1.002232520  1.1.1.2   1.1.1.1      ICMP     98 Echo (ping) requ
   6 1.002379889  1.1.1.1   1.1.1.2      ICMP     98 Echo (ping) repl
   7 1.722431115  1.1.1.1   1.1.1.2      ICMP     98 Echo (ping) requ
   8 1.723043704  1.1.1.2   1.1.1.1      ICMP     98 Echo (ping) repl
   9 2.027172688  1.1.1.2   1.1.1.1      ICMP     98 Echo (ping) requ
  10 2.027321008  1.1.1.1   1.1.1.2      ICMP     98 Echo (ping) repl
  11 2.746341996  1.1.1.1   1.1.1.2      ICMP     98 Echo (ping) requ
  12 2.746946653  1.1.1.2   1.1.1.1      ICMP     98 Echo (ping) repl

▸ Frame 2: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface 0
▾ Ethernet II, Src: Vmware_94:21:8d (00:0c:29:94:21:8d), Dst: Vmware_8e:22:19 (00:0c:29:8e:22:19)
  ▸ Destination: Vmware_8e:22:19 (00:0c:29:8e:22:19)
  ▸ Source: Vmware_94:21:8d (00:0c:29:94:21:8d)
    Type: IPv4 (0x0800)
▾ Internet Protocol Version 4, Src: 1.1.1.1, Dst: 1.1.1.2
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 84
    Identification: 0x54a9 (21673)
  ▸ Flags: 0x0000
    Time to live: 64

0000  00 0c 29 8e 22 19 00 0c  29 94 21 8d 08 00 45 00   ··)·"···  )·!···E·
0010  00 54 54 a9 00 00 40 01  21 fc 01 01 01 01 01 01   ·TT···@·  !·······

macsec0.pcapng          Packets: 126 · Displayed: 126 (100.0%)    Profile: Default
```

Figure 5.1: Captured packets macsec0 ICMP from the First and Second VMs.


First Ubuntu 18.04.1 - VMware Workstation 12 Player (Non-commercial use...
Player
Activities
Wireshark
Τρι 12:41
macsec0.pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...

```
No.  Time          Source           Destination      Protocol Length Info
  94 23.274821881  1.1.1.1          1.1.1.2          ICMP     98 Echo (ping) repl
  95 24.058360758  1.1.1.1          1.1.1.2          ICMP     98 Echo (ping) requ
  96 24.059302073  1.1.1.2          1.1.1.1          ICMP     98 Echo (ping) repl
  97 24.298622288  1.1.1.2          1.1.1.1          ICMP     98 Echo (ping) requ
  98 24.298798498  1.1.1.1          1.1.1.2          ICMP     98 Echo (ping) repl
  99 24.986453018  Vmware_94:21:8d  Vmware_8e:22:19  ARP      42 Who has 1.1.1.2?
 100 24.987009347  Vmware_8e:22:19  Vmware_94:21:8d  ARP      42 1.1.1.2 is at 00
 101 25.060446323  1.1.1.1          1.1.1.2          ICMP     98 Echo (ping) requ
 102 25.061002159  1.1.1.2          1.1.1.1          ICMP     98 Echo (ping) repl
 103 25.322619562  1.1.1.2          1.1.1.1          ICMP     98 Echo (ping) requ
 104 25.322767903  1.1.1.1          1.1.1.2          ICMP     98 Echo (ping) repl
 105 26.074662945  1.1.1.1          1.1.1.2          ICMP     98 Echo (ping) requ

▸ Frame 100: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface 0
▾ Ethernet II, Src: Vmware_8e:22:19 (00:0c:29:8e:22:19), Dst: Vmware_94:21:8d (00:0c:29:94:21:8d)
  ▾ Destination: Vmware_94:21:8d (00:0c:29:94:21:8d)
      Address: Vmware_94:21:8d (00:0c:29:94:21:8d)
      .... ..0. .... .... .... .... = LG bit: Globally unique address (factory default)
      .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
  ▾ Source: Vmware_8e:22:19 (00:0c:29:8e:22:19)
      Address: Vmware_8e:22:19 (00:0c:29:8e:22:19)
      .... ..0. .... .... .... .... = LG bit: Globally unique address (factory default)
      .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
    Type: ARP (0x0806)
▸ Address Resolution Protocol (reply)

0000  00 0c 29 94 21 8d 00 0c  29 8e 22 19 08 06 00 01   ··)·!···  )·"·····
0010  08 00 06 04 00 02 00 0c  29 8e 22 19 01 01 01 02   ········  )·"·····

macsec0.pcapng          Packets: 126 · Displayed: 126 (100.0%)    Profile: Default
```

Figure 5.2: Captured packets macsec0 ARP from the First and Second VMs.

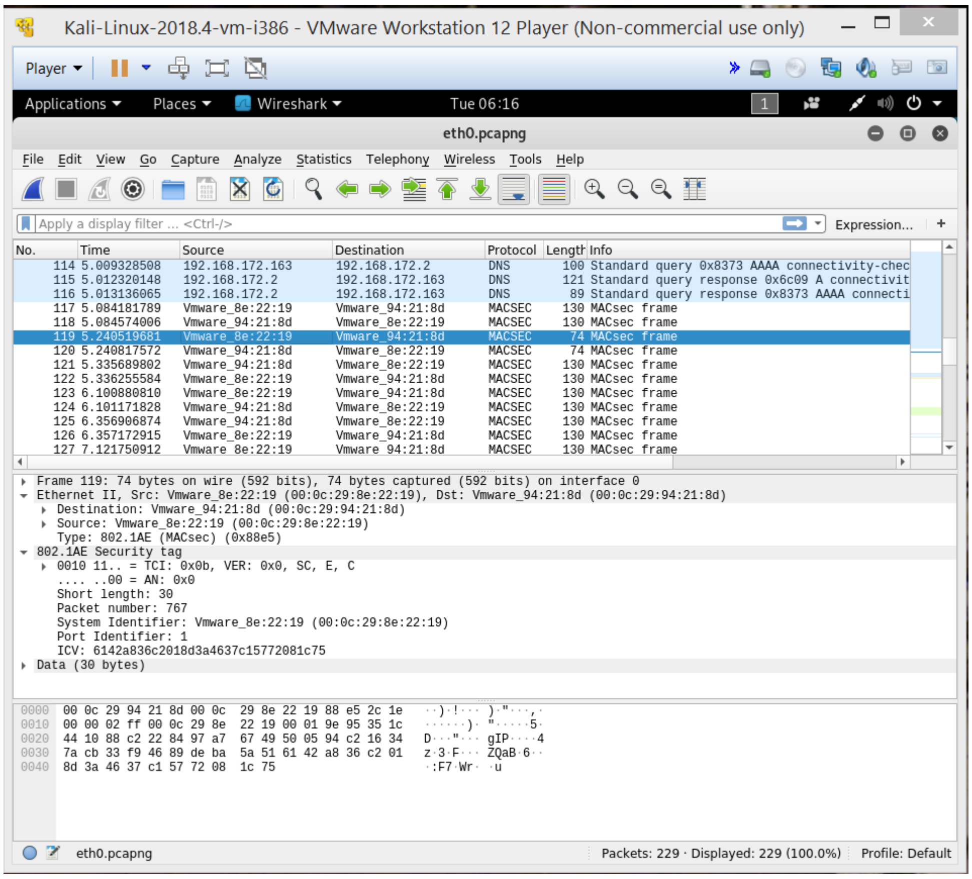


```
No.  Time         Source             Destination        Protocol Length Info
   1 0.000000000  Vmware_8e:22:19    Vmware_94:21:8d    MACSEC   130 MACsec frame
   2 0.000142145  Vmware_94:21:8d    Vmware_8e:22:19    MACSEC   130 MACsec frame
   3 0.911389594  Vmware_94:21:8d    Vmware_8e:22:19    MACSEC   130 MACsec frame
   4 0.911938396  Vmware_8e:22:19    Vmware_94:21:8d    MACSEC   130 MACsec frame
   5 1.001588889  Vmware_8e:22:19    Vmware_94:21:8d    MACSEC   130 MACsec frame
   6 1.001743769  Vmware_94:21:8d    Vmware_8e:22:19    MACSEC   130 MACsec frame
   7 1.935695116  Vmware_94:21:8d    Vmware_8e:22:19    MACSEC   130 MACsec frame
   8 1.936296827  Vmware_8e:22:19    Vmware_94:21:8d    MACSEC   130 MACsec frame
   9 2.014941073  Vmware_8e:22:19    Vmware_94:21:8d    MACSEC   130 MACsec frame
  10 2.015094507  Vmware_94:21:8d    Vmware_8e:22:19    MACSEC   130 MACsec frame
  11 2.959860900  Vmware_94:21:8d    Vmware_8e:22:19    MACSEC   130 MACsec frame
  12 2.960436719  Vmware_8e:22:19    Vmware_94:21:8d    MACSEC   130 MACsec frame

▸ Frame 1: 130 bytes on wire (1040 bits), 130 bytes captured (1040 bits) on interface 0
▾ Ethernet II, Src: Vmware_8e:22:19 (00:0c:29:8e:22:19), Dst: Vmware_94:21:8d (00:0c:29:94:21:8d)
   ▸ Destination: Vmware_94:21:8d (00:0c:29:94:21:8d)
   ▸ Source: Vmware_8e:22:19 (00:0c:29:8e:22:19)
     Type: 802.1AE (MACsec) (0x88e5)
▾ 802.1AE Security tag
   ▸ 0010 11.. = TCI: 0x0b, VER: 0x0, SC, E, C
     .... ..00 = AN: 0x0
     Short length: 0
     Packet number: 338
     System Identifier: Vmware_8e:22:19 (00:0c:29:8e:22:19)
     Port Identifier: 1
     ICV: ebe57d8b3d6749eb9893c072a3a63f02

0000  00 0c 29 94 21 8d 00 0c  29 8e 22 19 88 e5 2c 00   ··)·!···  )·"···,·
0010  00 00 01 52 00 0c 29 8e  22 19 00 01 be 75 1f 2b   ···R··)·  "····u·+

ens33.pcapng          Packets: 50 · Displayed: 50 (100.0%)    Profile: Default
```

Figure 5.3: Captured packets ens33 MACSEC from the First and Second VMs.

After some time has passed and the real IP address has retrieved automatically, we gain access to the Internet. Since this step required the real IP to access the internet, the attacking Third VM recorded the packets as it is illustrated in Figure 5.8: the real IP address 192.168.172.163 of the First VM and the IP 172.217.169.174 of the website google.com. In an attempt to verify, we hided the real IP address of the First VM with the command 'sudo ifconfig ens33 0.0.0.0' and later pinged to google and we could not connect. In addition, the same happens when we fail to download a web page using the First VM with fake IP address, while its real IP is hidden, but when the real IP address reappears after a while we succeed. This is also witnessed by the Third VM, capturing the packets traffic as it is illustrated in Figure 5.9: the real IP address 192.168.172.163 of the First VM and the IP 160.153.154.142 of the downloaded web page. Again, we attempted to verify hiding the real address of the First VM with the command 'sudo ifconfig ens33 0.0.0.0' and downloading the web page and we had not access to the Internet. Even in the case when the real IP of the VM is revealed, the attacker obtained no information about the action of downloading, except interaction of the First VM with a website whose IP was tracked. Therefore, as expected, we verify that the HTTP needs the real IP address of the VM in order to gain access to the Internet while the MACSec protocol does not function due to a luck of established session, and thus it does not protect the packets data and the machine becomes vulnerable to an internal attacker who can track the real IP addresses.

In the final step, we create from the side of the Second VM three files with different sizes to measure the time needed for the files to be copied to the First VM when the


Kali-Linux-2018.4-vm-i386 - VMware Workstation 12 Player (Non-commercial use only)
Player
Applications
Places
Wireshark
Tue 06:16
eth0.pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...

```
No.  Time           Source            Destination       Protocol Length Info
 114 5.009328508    192.168.172.163   192.168.172.2     DNS       100 Standard query 0x8373 AAAA connectivity-chec
 115 5.012320148    192.168.172.2     192.168.172.163   DNS       121 Standard query response 0x6c09 A connectivit
 116 5.013136065    192.168.172.2     192.168.172.163   DNS        89 Standard query response 0x8373 AAAA connecti
 117 5.084181789    Vmware_8e:22:19   Vmware_94:21:8d   MACSEC    130 MACsec frame
 118 5.084574006    Vmware_94:21:8d   Vmware_8e:22:19   MACSEC    130 MACsec frame
 119 5.240519681    Vmware_8e:22:19   Vmware_94:21:8d   MACSEC     74 MACsec frame
 120 5.240817572    Vmware_94:21:8d   Vmware_8e:22:19   MACSEC     74 MACsec frame
 121 5.335689802    Vmware_94:21:8d   Vmware_8e:22:19   MACSEC    130 MACsec frame
 122 5.336255584    Vmware_8e:22:19   Vmware_94:21:8d   MACSEC    130 MACsec frame
 123 6.100880810    Vmware_8e:22:19   Vmware_94:21:8d   MACSEC    130 MACsec frame
 124 6.101171828    Vmware_94:21:8d   Vmware_8e:22:19   MACSEC    130 MACsec frame
 125 6.356906874    Vmware_94:21:8d   Vmware_8e:22:19   MACSEC    130 MACsec frame
 126 6.357172915    Vmware_8e:22:19   Vmware_94:21:8d   MACSEC    130 MACsec frame
 127 7.121750912    Vmware_8e:22:19   Vmware_94:21:8d   MACSEC    130 MACsec frame

Frame 119: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
Ethernet II, Src: Vmware_8e:22:19 (00:0c:29:8e:22:19), Dst: Vmware_94:21:8d (00:0c:29:94:21:8d)
  Destination: Vmware_94:21:8d (00:0c:29:94:21:8d)
  Source: Vmware_8e:22:19 (00:0c:29:8e:22:19)
  Type: 802.1AE (MACsec) (0x88e5)
802.1AE Security tag
  0010 11.. = TCI: 0x0b, VER: 0x0, SC, E, C
  .... ..00 = AN: 0x0
  Short length: 30
  Packet number: 767
  System Identifier: Vmware_8e:22:19 (00:0c:29:8e:22:19)
  Port Identifier: 1
  ICV: 6142a836c2018d3a4637c15772081c75
Data (30 bytes)

0000  00 0c 29 94 21 8d 00 0c  29 8e 22 19 88 e5 2c 1e   ··)·!··· )·"···,·
0010  00 00 02 ff 00 0c 29 8e  22 19 00 01 9e 95 35 1c   ······)· "·····5·
0020  44 10 88 c2 22 84 97 a7  67 49 50 05 94 c2 16 34   D···"··· gIP····4
0030  7a cb 33 f9 46 89 de ba  5a 51 61 42 a8 36 c2 01   z·3·F··· ZQaB·6··
0040  8d 3a 46 37 c1 57 72 08  1c 75                     ·:F7·Wr· ·u

eth0.pcapng    Packets: 229 · Displayed: 229 (100.0%)    Profile: Default
```

Figure 5.4: Captured packets traffic between the First and Second VMs from Kali Linux.

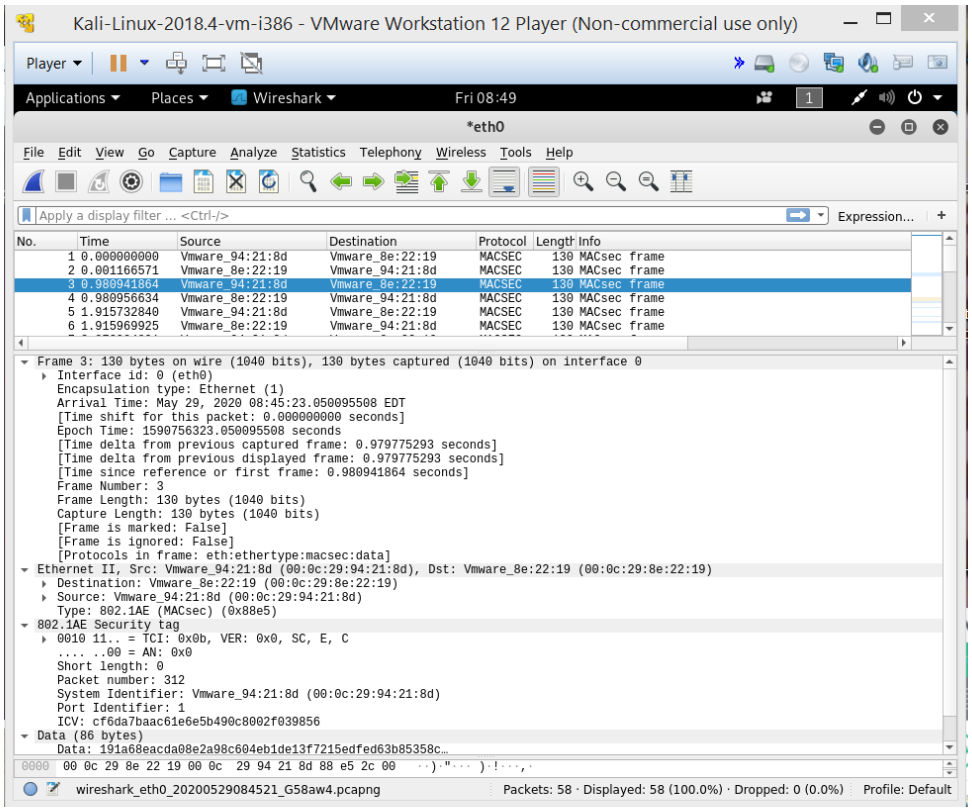
MACSec session is disabled. The IP: 192.168.172.168 is the real IP address of the Second VM, while the IP: 192.168.172.175 is the real IP address of the First VM. As it is illustrated in Figure 4.20, the time of files transfer is 4.733 s for the 5MB file, 6.696 s for the 20MB file and 6.849 s for the 30MB file.

Later, after enabling the MACSec session, we used again the Second VM to copy the files to the First VM. The IP: 1.1.1.1 is the fake IP address of the First VM, while the IP: 1.1.1.2 is the fake IP address of the Second VM. As it is depicted in Figure 4.21 the time of files transfer is 4.750 s for the 5MB file, 8.889 s for the 20MB file and 10.447 s for the 30MB file.

Below, we present a table with the different times of files transfer depending on when the MACSec is disabled or enabled.

Time of files transfer from the Second to the First VM (in seconds)			
	First File 5MB	Second File 20MB	Third File 30MB
MACSec is not Enabled	4.733	6.696	6.849
Enabled MACSec	4.750	8.889	10.447

As it is observed, the time of files transfer between the two VMs is prolonged when the MACSec is enabled and the bigger the size of the file, the bigger gets the


Kali-Linux-2018.4-vm-i386 - VMware Workstation 12 Player (Non-commercial use only)
Player
Applications
Places
Wireshark
Fri 08:49
*eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...

```
No.  Time         Source            Destination       Protocol Length Info
   1 0.000000000  Vmware_94:21:8d   Vmware_8e:22:19   MACSEC    130 MACsec frame
   2 0.001166571  Vmware_8e:22:19   Vmware_94:21:8d   MACSEC    130 MACsec frame
   3 0.980941864  Vmware_94:21:8d   Vmware_8e:22:19   MACSEC    130 MACsec frame
   4 0.980956634  Vmware_8e:22:19   Vmware_94:21:8d   MACSEC    130 MACsec frame
   5 1.915732840  Vmware_94:21:8d   Vmware_8e:22:19   MACSEC    130 MACsec frame
   6 1.915969925  Vmware_8e:22:19   Vmware_94:21:8d   MACSEC    130 MACsec frame
```

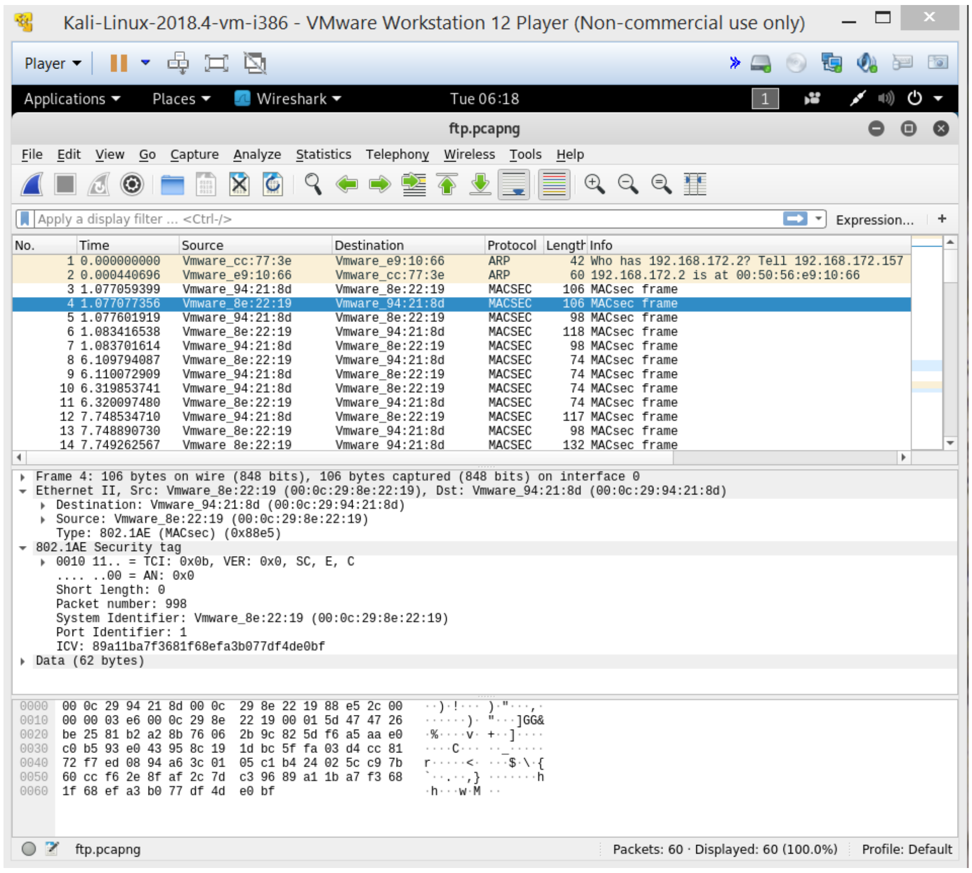
```
▾ Frame 3: 130 bytes on wire (1040 bits), 130 bytes captured (1040 bits) on interface 0
  ▸ Interface id: 0 (eth0)
    Encapsulation type: Ethernet (1)
    Arrival Time: May 29, 2020 08:45:23.050095508 EDT
    [Time shift for this packet: 0.000000000 seconds]
    Epoch Time: 1590756323.050095508 seconds
    [Time delta from previous captured frame: 0.979775293 seconds]
    [Time delta from previous displayed frame: 0.979775293 seconds]
    [Time since reference or first frame: 0.980941864 seconds]
    Frame Number: 3
    Frame Length: 130 bytes (1040 bits)
    Capture Length: 130 bytes (1040 bits)
    [Frame is marked: False]
    [Frame is ignored: False]
    [Protocols in frame: eth:ethertype:macsec:data]
▾ Ethernet II, Src: Vmware_94:21:8d (00:0c:29:94:21:8d), Dst: Vmware_8e:22:19 (00:0c:29:8e:22:19)
  ▸ Destination: Vmware_8e:22:19 (00:0c:29:8e:22:19)
  ▸ Source: Vmware_94:21:8d (00:0c:29:94:21:8d)
    Type: 802.1AE (MACsec) (0x88e5)
▾ 802.1AE Security tag
  ▸ 0010 11.. = TCI: 0x0b, VER: 0x0, SC, E, C
    .... ..00 = AN: 0x0
    Short length: 0
    Packet number: 312
    System Identifier: Vmware_94:21:8d (00:0c:29:94:21:8d)
    Port Identifier: 1
    ICV: cf6da7baac61e6e5b490c8002f039856
▾ Data (86 bytes)
    Data: 191a68eacda08e2a98c604eb1de13f7215edfed63b85358c…
```

```
0000  00 0c 29 8e 22 19 00 0c  29 94 21 8d 88 e5 2c 00   ··)·"··· )·!···,·
```

```
wireshark_eth0_20200529084521_G58aw4.pcapng    Packets: 58 · Displayed: 58 (100.0%) · Dropped: 0 (0.0%)    Profile: Default
```

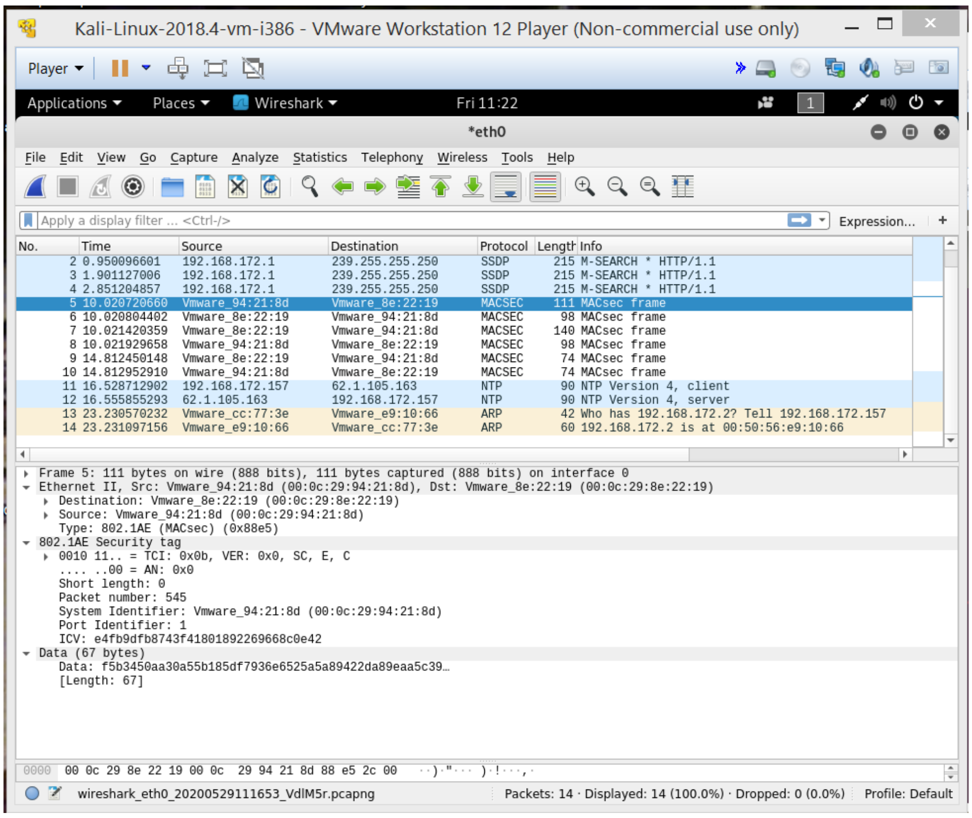
Figure 5.5: Captured packets traffic between the First and Second VMs from Kali Linux when the real IP addresses are retrieved.

difference. So, when the MACSec is enabled, for the first file with size of 5MB the difference in time transfer is negligible, for the second file of 20MB size the time is increased by more than 2 ec and for the third file with 30MB size the time is increased by less than 4 ec.


Kali-Linux-2018.4-vm-i386 - VMware Workstation 12 Player (Non-commercial use only)
Player
Applications
Places
Wireshark
Tue 06:18
ftp.pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...

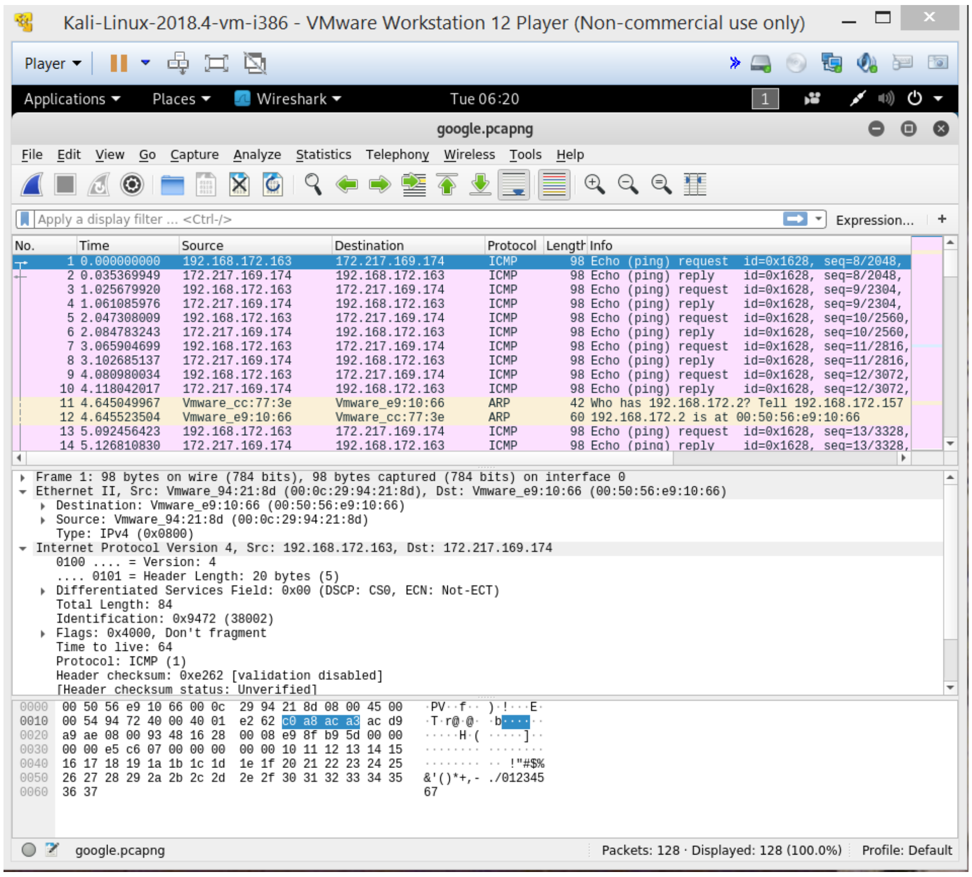
```
No.   Time          Source             Destination        Protocol Length Info
    1 0.000000000   Vmware_cc:77:3e    Vmware_e9:10:66    ARP        42 Who has 192.168.172.2? Tell 192.168.172.157
    2 0.000440696   Vmware_e9:10:66    Vmware_cc:77:3e    ARP        60 192.168.172.2 is at 00:50:56:e9:10:66
    3 1.077059399   Vmware_94:21:8d    Vmware_8e:22:19    MACSEC    106 MACsec frame
    4 1.077077356   Vmware_8e:22:19    Vmware_94:21:8d    MACSEC    106 MACsec frame
    5 1.077601919   Vmware_94:21:8d    Vmware_8e:22:19    MACSEC     98 MACsec frame
    6 1.083416538   Vmware_8e:22:19    Vmware_94:21:8d    MACSEC    118 MACsec frame
    7 1.083701614   Vmware_94:21:8d    Vmware_8e:22:19    MACSEC     98 MACsec frame
    8 6.109794087   Vmware_8e:22:19    Vmware_94:21:8d    MACSEC     74 MACsec frame
    9 6.110072909   Vmware_94:21:8d    Vmware_8e:22:19    MACSEC     74 MACsec frame
   10 6.319853741   Vmware_94:21:8d    Vmware_8e:22:19    MACSEC     74 MACsec frame
   11 6.320097480   Vmware_8e:22:19    Vmware_94:21:8d    MACSEC     74 MACsec frame
   12 7.748534710   Vmware_94:21:8d    Vmware_8e:22:19    MACSEC    117 MACsec frame
   13 7.748890730   Vmware_8e:22:19    Vmware_94:21:8d    MACSEC     98 MACsec frame
   14 7.749262567   Vmware_8e:22:19    Vmware_94:21:8d    MACSEC    132 MACsec frame

Frame 4: 106 bytes on wire (848 bits), 106 bytes captured (848 bits) on interface 0
Ethernet II, Src: Vmware_8e:22:19 (00:0c:29:8e:22:19), Dst: Vmware_94:21:8d (00:0c:29:94:21:8d)
  Destination: Vmware_94:21:8d (00:0c:29:94:21:8d)
  Source: Vmware_8e:22:19 (00:0c:29:8e:22:19)
  Type: 802.1AE (MACsec) (0x88e5)
802.1AE Security tag
  0010 11.. = TCI: 0x0b, VER: 0x0, SC, E, C
  .... ..00 = AN: 0x0
  Short length: 0
  Packet number: 998
  System Identifier: Vmware_8e:22:19 (00:0c:29:8e:22:19)
  Port Identifier: 1
  ICV: 89a11ba7f3681f68efa3b077df4de0bf
Data (62 bytes)

0000  00 0c 29 94 21 8d 00 0c  29 8e 22 19 88 e5 2c 00   ··)·!··· )·"···,·
0010  00 00 03 e6 00 0c 29 8e  22 19 00 01 5d 47 47 26   ······)· "···]GG&
0020  be 25 81 b2 a2 8b 76 06  2b 9c 82 5d f6 a5 aa e0   ·%····v· +··]····
0030  c0 b5 93 e0 43 95 8c 19  1d bc 5f fa 03 d4 cc 81   ····C··· ··_·····
0040  72 f7 ed 08 94 a6 3c 01  05 c1 b4 24 02 5c c9 7b   r·····<· ··$·\·{
0050  60 cc f6 2e 8f af 2c 7d  c3 96 89 a1 1b a7 f3 68   `··.··,} ·······h
0060  1f 68 ef a3 b0 77 df 4d  e0 bf                     ·h···w·M ··

ftp.pcapng        Packets: 60 · Displayed: 60 (100.0%)        Profile: Default
```

Figure 5.6: Packets between the First and Second VM using the FTP protocol as captured by the Third VM.


Kali-Linux-2018.4-vm-i386 - VMware Workstation 12 Player (Non-commercial use only)
Player
Applications
Places
Wireshark
Fri 11:22
*eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...

```
No.   Time           Source            Destination       Protocol Length Info
    2 0.950096601    192.168.172.1     239.255.255.250   SSDP      215 M-SEARCH * HTTP/1.1
    3 1.901127006    192.168.172.1     239.255.255.250   SSDP      215 M-SEARCH * HTTP/1.1
    4 2.851204857    192.168.172.1     239.255.255.250   SSDP      215 M-SEARCH * HTTP/1.1
    5 10.020720660   Vmware_94:21:8d   Vmware_8e:22:19   MACSEC    111 MACsec frame
    6 10.020804402   Vmware_8e:22:19   Vmware_94:21:8d   MACSEC     98 MACsec frame
    7 10.021420359   Vmware_8e:22:19   Vmware_94:21:8d   MACSEC    140 MACsec frame
    8 10.021929658   Vmware_94:21:8d   Vmware_8e:22:19   MACSEC     98 MACsec frame
    9 14.812450148   Vmware_8e:22:19   Vmware_94:21:8d   MACSEC     74 MACsec frame
   10 14.812952910   Vmware_94:21:8d   Vmware_8e:22:19   MACSEC     74 MACsec frame
   11 16.528712902   192.168.172.157   62.1.105.163      NTP        90 NTP Version 4, client
   12 16.555855293   62.1.105.163      192.168.172.157   NTP        90 NTP Version 4, server
   13 23.230570232   Vmware_cc:77:3e   Vmware_e9:10:66   ARP        42 Who has 192.168.172.2? Tell 192.168.172.157
   14 23.231097156   Vmware_e9:10:66   Vmware_cc:77:3e   ARP        60 192.168.172.2 is at 00:50:56:e9:10:66

Frame 5: 111 bytes on wire (888 bits), 111 bytes captured (888 bits) on interface 0
Ethernet II, Src: Vmware_94:21:8d (00:0c:29:94:21:8d), Dst: Vmware_8e:22:19 (00:0c:29:8e:22:19)
  Destination: Vmware_8e:22:19 (00:0c:29:8e:22:19)
  Source: Vmware_94:21:8d (00:0c:29:94:21:8d)
  Type: 802.1AE (MACsec) (0x88e5)
802.1AE Security tag
  0010 11.. = TCI: 0x0b, VER: 0x0, SC, E, C
  .... ..00 = AN: 0x0
  Short length: 0
  Packet number: 545
  System Identifier: Vmware_94:21:8d (00:0c:29:94:21:8d)
  Port Identifier: 1
  ICV: e4fb9dfb8743f41801892269668c0e42
Data (67 bytes)
  Data: f5b3450aa30a55b185df7936e6525a5a89422da89eaa5c39…
  [Length: 67]

0000  00 0c 29 8e 22 19 00 0c  29 94 21 8d 88 e5 2c 00   ··)·"··· )·!···,·

wireshark_eth0_20200529111653_VdlM5r.pcapng    Packets: 14 · Displayed: 14 (100.0%) · Dropped: 0 (0.0%)    Profile: Default
```

Figure 5.7: Packets between the First and Second VM using the FTP protocol as captured by the Third VM when the real IP addresses are retrieved.



Kali-Linux-2018.4-vm-i386 - VMware Workstation 12 Player (Non-commercial use only)
Player
Applications
Places
Wireshark
Tue 06:20
google.pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...
Packets: 128 · Displayed: 128 (100.0%)
Profile: Default

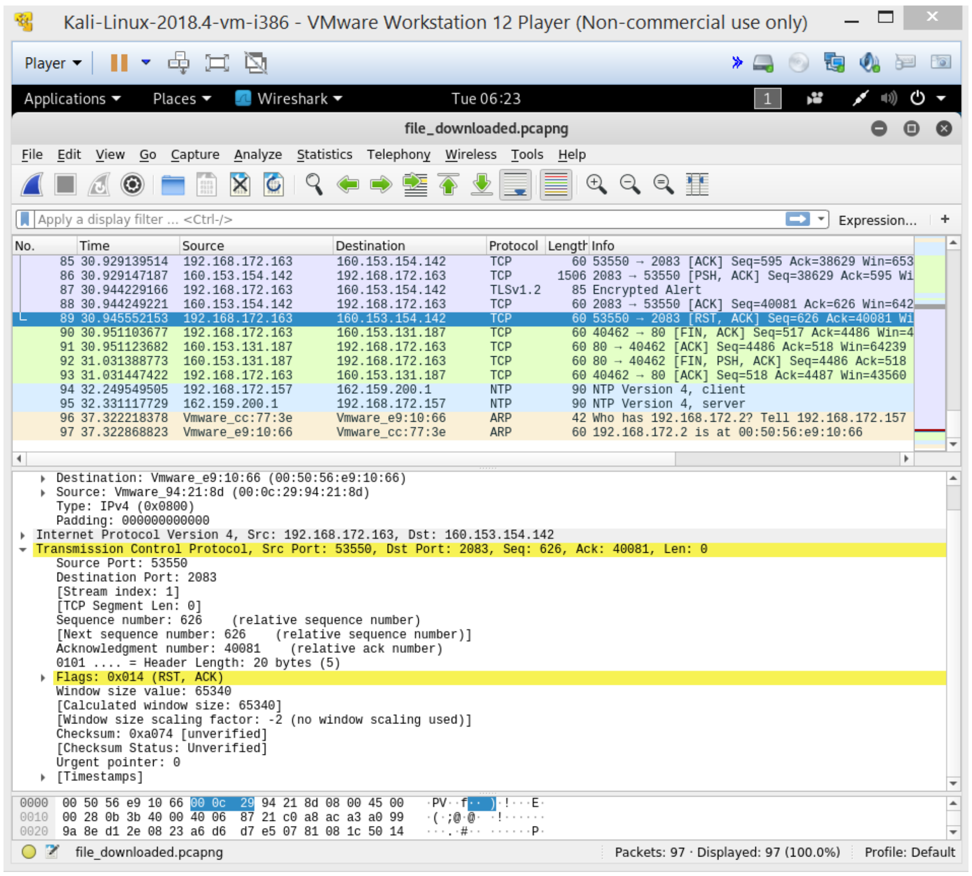
```
No.  Time          Source            Destination       Protocol Length Info
   1 0.000000000   192.168.172.163   172.217.169.174   ICMP      98 Echo (ping) request  id=0x1628, seq=8/2048,
   2 0.035369949   172.217.169.174   192.168.172.163   ICMP      98 Echo (ping) reply    id=0x1628, seq=8/2048,
   3 1.025679920   192.168.172.163   172.217.169.174   ICMP      98 Echo (ping) request  id=0x1628, seq=9/2304,
   4 1.061085976   172.217.169.174   192.168.172.163   ICMP      98 Echo (ping) reply    id=0x1628, seq=9/2304,
   5 2.047308009   192.168.172.163   172.217.169.174   ICMP      98 Echo (ping) request  id=0x1628, seq=10/2560,
   6 2.084783243   172.217.169.174   192.168.172.163   ICMP      98 Echo (ping) reply    id=0x1628, seq=10/2560,
   7 3.065904699   192.168.172.163   172.217.169.174   ICMP      98 Echo (ping) request  id=0x1628, seq=11/2816,
   8 3.102685137   172.217.169.174   192.168.172.163   ICMP      98 Echo (ping) reply    id=0x1628, seq=11/2816,
   9 4.080980034   192.168.172.163   172.217.169.174   ICMP      98 Echo (ping) request  id=0x1628, seq=12/3072,
  10 4.118042017   172.217.169.174   192.168.172.163   ICMP      98 Echo (ping) reply    id=0x1628, seq=12/3072,
  11 4.645049967   Vmware_cc:77:3e   Vmware_e9:10:66   ARP       42 Who has 192.168.172.2? Tell 192.168.172.157
  12 4.645523504   Vmware_e9:10:66   Vmware_cc:77:3e   ARP       60 192.168.172.2 is at 00:50:56:e9:10:66
  13 5.092456423   192.168.172.163   172.217.169.174   ICMP      98 Echo (ping) request  id=0x1628, seq=13/3328,
  14 5.126810830   172.217.169.174   192.168.172.163   ICMP      98 Echo (ping) reply    id=0x1628, seq=13/3328,
```

```
▸ Frame 1: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface 0
▾ Ethernet II, Src: Vmware_94:21:8d (00:0c:29:94:21:8d), Dst: Vmware_e9:10:66 (00:50:56:e9:10:66)
  ▸ Destination: Vmware_e9:10:66 (00:50:56:e9:10:66)
  ▸ Source: Vmware_94:21:8d (00:0c:29:94:21:8d)
    Type: IPv4 (0x0800)
▾ Internet Protocol Version 4, Src: 192.168.172.163, Dst: 172.217.169.174
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 84
    Identification: 0x9472 (38002)
  ▸ Flags: 0x4000, Don't fragment
    Time to live: 64
    Protocol: ICMP (1)
    Header checksum: 0xe262 [validation disabled]
    [Header checksum status: Unverified]
```

```
0000  00 50 56 e9 10 66 00 0c  29 94 21 8d 08 00 45 00   ·PV··f·· )·!···E·
0010  00 54 94 72 40 00 40 01  e2 62 c0 a8 ac a3 ac d9   ·T·r@·@· ·b······
0020  a9 ae 08 00 93 48 16 28  00 08 e9 8f b9 5d 00 00   ·····H·( ·····]··
0030  00 00 e5 c6 07 00 00 00  00 00 10 11 12 13 14 15   ········ ········
0040  16 17 18 19 1a 1b 1c 1d  1e 1f 20 21 22 23 24 25   ········ ·· !"#$%
0050  26 27 28 29 2a 2b 2c 2d  2e 2f 30 31 32 33 34 35   &'()*+,- ./012345
0060  36 37                                              67
```

google.pcapng

Figure 5.8: Captured packets by the Third VM between the First VM and the google.com website.


Kali-Linux-2018.4-vm-i386 - VMware Workstation 12 Player (Non-commercial use only)
Player
Applications
Places
Wireshark
Tue 06:23
file_downloaded.pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...

```
No.  Time          Source           Destination      Protocol Length Info
  85 30.929139514  192.168.172.163  160.153.154.142  TCP        60 53550 → 2083 [ACK] Seq=595 Ack=38629 Win=653
  86 30.929147187  160.153.154.142  192.168.172.163  TCP      1506 2083 → 53550 [PSH, ACK] Seq=38629 Ack=595 Wi
  87 30.944229166  192.168.172.163  160.153.154.142  TLSv1.2    85 Encrypted Alert
  88 30.944249221  160.153.154.142  192.168.172.163  TCP        60 2083 → 53550 [ACK] Seq=40081 Ack=626 Win=642
  89 30.945552153  192.168.172.163  160.153.154.142  TCP        60 53550 → 2083 [RST, ACK] Seq=626 Ack=40081 Wi
  90 30.951103677  192.168.172.163  160.153.131.187  TCP        60 40462 → 80 [FIN, ACK] Seq=517 Ack=4486 Win=4
  91 30.951123682  160.153.131.187  192.168.172.163  TCP        60 80 → 40462 [ACK] Seq=4486 Ack=518 Win=64239
  92 31.031388773  160.153.131.187  192.168.172.163  TCP        60 80 → 40462 [FIN, PSH, ACK] Seq=4486 Ack=518
  93 31.031447422  192.168.172.163  160.153.131.187  TCP        60 40462 → 80 [ACK] Seq=518 Ack=4487 Win=43560
  94 32.249549505  192.168.172.157  162.159.200.1    NTP        90 NTP Version 4, client
  95 32.331117729  162.159.200.1    192.168.172.157  NTP        90 NTP Version 4, server
  96 37.322218378  Vmware_cc:77:3e  Vmware_e9:10:66  ARP        42 Who has 192.168.172.2? Tell 192.168.172.157
  97 37.322868823  Vmware_e9:10:66  Vmware_cc:77:3e  ARP        60 192.168.172.2 is at 00:50:56:e9:10:66
```

```
    Destination: Vmware_e9:10:66 (00:50:56:e9:10:66)
    Source: Vmware_94:21:8d (00:0c:29:94:21:8d)
    Type: IPv4 (0x0800)
    Padding: 000000000000
 Internet Protocol Version 4, Src: 192.168.172.163, Dst: 160.153.154.142
 Transmission Control Protocol, Src Port: 53550, Dst Port: 2083, Seq: 626, Ack: 40081, Len: 0
    Source Port: 53550
    Destination Port: 2083
    [Stream index: 1]
    [TCP Segment Len: 0]
    Sequence number: 626    (relative sequence number)
    [Next sequence number: 626    (relative sequence number)]
    Acknowledgment number: 40081    (relative ack number)
    0101 .... = Header Length: 20 bytes (5)
    Flags: 0x014 (RST, ACK)
    Window size value: 65340
    [Calculated window size: 65340]
    [Window size scaling factor: -2 (no window scaling used)]
    Checksum: 0xa074 [unverified]
    [Checksum Status: Unverified]
    Urgent pointer: 0
    [Timestamps]
```

```
0000  00 50 56 e9 10 66 00 0c  29 94 21 8d 08 00 45 00   ·PV··f·· )·!···E·
0010  00 28 0b 3b 40 00 40 06  87 21 c0 a8 ac a3 a0 99   ·(·;@·@· ·!······
0020  9a 8e d1 2e 08 23 a6 d6  d7 e5 07 81 08 1c 50 14   ···.·#·· ······P·
```

Figure 5.9: Captured packets by the Third VM during the download of the web page.

Chapter 6

Discussion

The conduction of this study required that the MACSec protocol is supported in two VMs which would interact between them, testing the efficiency of the protocol in terms of secure communication. For the experimental setup, Ubuntu operating system needed to be installed on the First and Second VM. Even though all versions support the kernel's .config file[35], all the previous versions of Ubuntu before 2016 do not include the field 'CONFIG_MACSec'[8] so an installation of the latest version of kernel was necessary. This explains why a previous version of Ubuntu initially tested for the study, namely Ubuntu 16.04 LTS, did not include the 'CONFIG_MACSec' field in the kernel's .config file, something that inhibited the keys exchange in later steps and the session could not be established. In contrast, in the specific version of Ubuntu 18.04.1 installed on the First and Second VMs, the kernel's .config file had already enabled the MACSec field, so we did not have to upgrade the kernel. The only thing needed was to install the iproute2 and later reboot the systems.

For the course of this study for the sake of tracking the information in the packets data transmitted between the First and the Second VMs to check a potential vulnerabilities of MACSec, we made use of the wireshark tool, which has a graphical front-end and many more sorting and filtering options. It allows the attacker to monitor all network traffic on the network by setting the network card in promiscuous mode to see all traffic visible on this interface, not just the traffic directed to one of the configured interface addresses and of broadcast/multicast traffic. However, we recognise that the wireshark does not send packets to the network or do active things, thus can only be used by the attacking Third VM for tracking and capturing of packets and not for any intervntional action[33].

Although the process for establishing a secure session in the First and Second VMs was done properly and the real IP addresses were hidden.

Some difficulties arisen when the VMs were shutting down, i.e., each time we had to follow the steps of establishing the secure session from the beginning. This incidence is reasonable if we take into consideration the way the MACSec protocol functions, requiring an established session that if it is interrupted it will need to be repeated [9]. Yet, the MACSec session remained established even though the real IPs reappeared automatically after some time, as it is indicated in the case in which the attacking VM was unable to track the packets between the two VMs when executing ping command or using FTP when the real IPs revealed.

Concerning the ICMP and the ARP, both belong to the Layer 3 of OSI and were recorded by the wireshark tool when we executed the ping command in the first step of packets traffic between the two VMs. The Internet Control Message Protocol (ICMP) is one of the most important protocols on the Internet and is mainly used to test connectivity, gauge response time and exchange error messages (such as missing a service from a server or not having a computer on the network) from the operating systems of a network's computers. Most of the time it is not used by the applications that are running on a computer, but by its operating system. However, one of the exceptions to this rule is the ping utility, which sends an ICMP Echo Request Messages to a computer on the network to determine if that computer is active or not and how long it takes for the message to reach it. If this computer is active, it will respond with an Echo Response Messages [36]. The ping utility is used to ensure that basic Layer 2 Ethernet connectivity is established and verified before attempting to enable MACSec [21], [37]. On the other hand, the ARP helps the communication between two devices in the Internet. It is used to detect a link-level address or hardware address of a foreign computer based on a communication-level address. Its function is to translate the IP address into a physical address (MAC). The ARP transmits a packet to all devices on the source network. Then, network devices separate the header of the data link layer from the protocol data unit (PDU). The PDU, called frame, transports the packet to Layer 3, where the network ID of the packet is validated with the network ID of the packet of the destination IP. If it is equal then responds to the source with the MAC address of the destination, otherwise the packet reaches the network gateway and transmits the packet to the devices with which it is connected and validates the network ID[38]. Each device on a network has a unique number, namely a hardware address or MAC address, that is used by the Layer 2 of OSI protocols, including MACSec, to ensure that data intended for a specific machine gets to it properly [2]. Therefore, both the ICMP and ARP support in different ways the MACSec session.

Regarding the FTP and HTTP, both belong to the Layer 7 of OSI, nevertheless they interact in a different way with the MACSec protocol. The MACSec protects traffic at data link layer (Layer 2 of OSI). It is a standardized IEEE 802.1AE hop-by-hop encryption that enables confidentiality and integrity of data at layer 2 [12]. The FTP functions in a client-server communication [39] and this justifies why in the third step of our study the FTP functions through the secure session of MACSec protocol. In contrast, the HTTP functions when accessing a web page while a MACSec session is not established between the VM and the site. Therefore, the HTTP is not related to the session of the MACSec protocol. In addition, the First VM cannot access the internet with its real IP address hidden [40], and since the HTTP requires the real IPs to be retrieved, the VM cannot be secured against an internal attacker who can capture the real IP addresses, and thus it becomes vulnerable to attacks such as DDoS [41] and IP spoofing [42].

In the final step, as for the Secure Shell (SSH) protocol which is necessary for the enabling of file transfer, Transmission Control Protocol (TCP) Port Number 22 is the default de facto SSH Port assigned by Internet Assigned Numbers Authority (IANA). SSH is used for secure logins, port forwarding and it can transfer files using the associated SSH file transfer (SFTP) or secure copy (SCP) protocols [43], [44]. The evaluation of the interference of the MACSec protocol with the time of files transfer between the two VMs showed that during the MACSec session this time is augmented. This is explained by the fact that when the MACSec is enabled, some headers are added, and thus the file transfer is delayed. In detail, in the MACSec session a packet starts with an Ethernet header, the EtherType 88E5. This is followed by the MACSec SecTAG, which contains information that help the receiver recognize the decryption key, as well as a packet number for replay protection. After SecTAG, the payload ensues which can be encrypted and lastly the Integrity Check Value (ICV) which is produced by GCM-AES and guarantees that the package was actually created by a node that owned the key and that it was not modified during the sending [6].

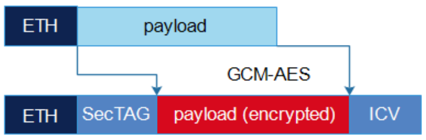


Figure 6.1: An Ethernet frame before and after MACSec processing and encryption. [6]

Chapter 7

Conclusions

In this thesis, we confirmed that after the MACSec session has been properly established between machines connected to the same LAN, they are protected against the internal attacker and the FTP supports the function of the MACSec. Yet, we found that this is not true with the HTTP which functions between the machine and the website residing in an external network and not in the MACSec session between the two machines, rendering the packets vulnerable to the internal attacker. Also, we observed that the time required for copying a file is prolonged when the MACSec is enabled. We conclude that the MACSec is a security protocol efficient in protecting against an internal attacker attempting to monitor packets traffic and that different protocols either support or are independent of the MACSec protocol, whereas it delays the files transfer.

To our acknowledgment, it should be noted that for the design of this thesis our main concern was to focus on internal threats in a LAN, which if unintentionally can be exploited by an outsider attacker. Yet, we did not investigate directly the efficacy of MACSec security against external attacks which have always posed a risk in enterprises. Moreover, the wireshark tool used for monitoring and capturing of the packets traffic was not able to perform any immediate attack such as the Metasploit Framework, THC Hydra, Aircrack-ng and others [45]. Finally, the protocols studied was only four and compared to the total number of protocols of the OSI model we cannot draw a conclusion about the interaction of the MACSec security with the other security protocols.

As a future work, we recommend that further research could consider the study of MACSec connections between real (physical) machines and machines residing in different LAN networks.

Chapter 8

Related Work

To date, there are limited works in the current literature that examine the application and efficacy of the MACSec protocol in encrypting network traffic.

The work by Ju-Ho Choi et al. [46] proposes a new virtual secure link over Software-Defined Bridged Networks (SDBN) in order to surpass the security limitation of MACSec in the communication among devices within only single LAN at bridged networks. In SDBN session, the proposed scheme follows the SDN approach, resulting in the central MACSec module recognizing a group of devices at the bridged networks regardless of their attached LANs and being treated as they are attached to a shared media LAN, called virtual secure link. Thus, the proposed scheme applies secure end-to-end communication in bridged networks.

Jun-Won Lee et al. [47] develop an enhanced, secure ARP on MACSec in order to protect IP and MAC address from external threats such as 'ARP spoofing' and 'MAC flooding' in data link layer in the LAN. The enhanced ARP will encrypt ARP Packet by SAK (Secure Association Key) for the authentication. By using the same cipher module and the same key management protocol (802.1af) as in MACSec, it will reduce the resource consumption of the system and will focus on the IPv4 security issues related to ARP attack. Adding the authentication process to an existing ARP packet, the access of unauthorized host is controlled from the beginning of ARP update. So, it can secure the LAN against attacks by unauthorized users or viruses.

Jesús Lázaro et al. [48] aimed to investigate the integration of MACSec in the electrical substation environment and presents three different approaches to securing High-availability Seamless Redundancy (HSR) rings in the substation automation communication. MACSec is integrated in LAN in industrial systems interconnected with automation networks in substations being part of the electric distribution smart grid network. They conclude that MACSec secures non end-to-end communication such as synchronization and is proposed as a complementary measure of proposed end-to-end security schemes.

Ruichun Gu et al. [49] examines the application of the MACSec security for the communication in a Software Defined LTE Core Network, in which a VxLAN is used for scalable tunnel establishment to realize network slicing for multi-tenant cellular networks in the cloud environment. According to their proposal, this enhances the confidentiality and integrity of transmitted data with little performance penalty and promote the agility and scalability of the core network.

Overall, after taking into consideration the above mentioned studies, we notice that only Ruichun Gu et al. [49] evaluate the MACSec session in LAN against an internal attacker who attempts to capture the packets traffic between VMs, oriented to mobile networks.

Bibliography

[1] Philip Brookes. *What is a Local Area Network, and do I need one?* 2018-01-28. URL: https://aktiv.digital/blog/2018/01/28/local-area-network/.

[2] guard. *SECURE NETWORKING MACsec Fundamentals.*

[3] Nena Giandomenico Juliana de Groot. *wikipedia.com.* URL: https://en.wikipedia.org/wiki/Data_breach.

[4] Ding-Zhu Du Yang Xiao Xuemin Shen. *Wireless Network Security.* URL: https://books.google.gr/books?id=efCKBrOOqXQC&pg=PA390&lpg=PA390&dq=when+an+attacker+in+connected+to+the+same+lan,+this+is+an+external+attack?&source=bl&ots=LMMb-uHcJW&sig=ACfU3U21QByvfoEJCpZDYgKsOqppkbippw&hl=en&sa=X&ved=2ahUKEwikj4X15sTpAhWJUcAKHdjyCR0Q6AEwAHoECAoQAQ#v=onepage&q=when%20an%20attacker%20in%20connected%20to%20the%20same%20lan%2C%20this%20is%20an%20external%20attack%3F&f=false.

[5] John Thomas and Adam J. Elbirt. *How IPsec works, why we need it, and its biggest drawbacks.* 2004-01-06. URL: https://www.csoonline.com/article/2117067/data-protection-ipsec.html.

[6] Sabrina Dubroca. *MACsec: a different solution to encrypt network traffic.* URL: https://developers.redhat.com/blog/2016/10/14/macsec-a-different-solution-to-encrypt-network-traffic/ (visited on 10/14/2016).

[7] Platform and Release Support. *Understanding Media Access Control Security (MACsec).* URL: https://www.juniper.net/documentation/en_US/junos/topics/topic-map/understanding_media_access_control_security_qfx_ex.html (visited on 09/19/2020).

[8] *MACsec Implementation on Linux.* URL: https://costiser.ro/2016/08/01/macsec-implementation-on-linux/#.XcVBAtUzapq.

[9] *MACsec.* URL: http://docs.ruckuswireless.com/fastiron/08.0.70/fastiron-08070-licenseguide/GUID-EBB8AA84-C558-4A12-82F5-3A947FD66CBE.html.

[10] wikipedia.com. *Hop-by-Hop Transport.* URL: https://en.wikipedia.org/wiki/Hop-by-hop_transport.

[11] *Out of Band.* URL: https://www.techopedia.com/definition/31507/out-of-band.

[12] Cisco. *Cisco Nexus 7000 Series NX-OS Security Configuration Guide.*

[13] wikipedia.com. *Galois/Counter Mode*. URL: https://en.wikipedia.org/wiki/Galois/Counter_Mode.

[14] Sabrina Dubroca. *MACsec: Encryption for the wired LAN*. URL: https://netdevconf.info/1.1/proceedings/slides/dubroca-macsec-encryption-wire-lan.pdf (visited on 2016).

[15] Russ White. *Geneve*. 2015-11-05. URL: https://packetpushers.net/geneve/.

[16] Hangbin Liu. *An introduction to Linux virtual interfaces: Tunnels*. 2019-17-05. URL: https://developers.redhat.com/blog/2019/05/17/an-introduction-to-linux-virtual-interfaces-tunnels/.

[17] CISCO. *Identity-Based Networking Services: MAC Security*. URL: https://www.cisco.com/c/en/us/products/collateral/ios-nx-os-software/identity-based-networking-services/deploy_guide_c17-663760.html#wp9000248 (visited on 05/10/2011).

[18] wikipedia.com. *IPsec*. URL: https://en.wikipedia.org/wiki/IPsec.

[19] Margaret Rouse. *IPsec (Internet Protocol Security)*. URL: https://searchsecurity.techtarget.com/definition/IPsec-Internet-Protocol-Security.

[20] Knowledge Base. *ARCHIVED: What is the difference between a LAN, a MAN, and a WAN?* URL: https://kb.iu.edu/d/agki (visited on 01/18/2018).

[21] CISCO WHITE PAPER. *WAN MACsec Deployment White Paper*. URL: https://www.cisco.com/c/dam/en/us/td/docs/solutions/CVD/Aug2016/WP-WAN-MACsecDep-Aug2016.pdf.

[22] Martin Nuss. *The Skinny on IPSec vs. MACsec*. URL: https://www.electronicdesign.com/communications/skinny-ipsec-vs-macsec (visited on 03/19/2015).

[23] Computer Networking Notes. *Types of Network Protocols Explained with Functions*. URL: https://www.computernetworkingnotes.com/networking-tutorials/types-of-network-protocols-explained-with-functions.html (visited on 05/06/2018).

[24] Jon Martindale. *What is FTP?* URL: https://www.digitaltrends.com/computing/what-is-ftp-and-how-do-i-use-it/ (visited on 08/21/2019).

[25] wikipedia.com. *File Transfer Protocol*. URL: https://en.wikipedia.org/wiki/File_Transfer_Protocol.

[26] wikipedia.com. *Hypertext Transfer Protocol*. URL: https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol.

[27] Adarsh Verma. *10 Reasons To Use Ubuntu Linux*. 2018-04-06. URL: https://fossbytes.com/reasons-to-use-ubuntu-linux-advantage/.

[28] Anurag Bansal. *Technomania*. 2008-04-07. URL: https://anuragbansal.wordpress.com/2008/04/07/25-reasons-to-use-ubuntu-instead-of-windows/.

[29] rohitkharat. *What Is Kali Linux and Why Do Hackers Use Kali Linux OS*. 2014-02-14. URL: https://www.cybrary.it/blog/0p3n/kali-linux-hackers-use-kali-linux-os/.

[30] g0tmi1k. *What is Kali Linux?* 2019-11-25. URL: https://www.kali.org/docs/introduction/what-is-kali-linux/.

[31] wikipedia.com. *Kernel (operating system).* URL: https://en.wikipedia.org/wiki/Kernel_(operating_system)#cite_note-Linfo-1.

[32] wikipedia.com. *iproute2.* URL: https://en.wikipedia.org/wiki/Iproute2.

[33] *Wireshark.* URL: https://www.wireshark.org/docs/wsug_html_chunked/ChapterIntroduction.html.

[34] wikipedia.com. *Wireshark.* URL: https://en.wikipedia.org/wiki/Wireshark (visited on 2019).

[35] wikipedia.com. *Ubuntu version history.* URL: https://en.wikipedia.org/wiki/Ubuntu_version_history.

[36] wikipedia.com. *ICMP.* URL: https://en.wikipedia.org/wiki/Internet_Control_Message_Protocol.

[37] CISCO. *MACSEC and MKA Configuration Guide, Cisco IOS XE Fuji 16.7.x.* URL: https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/macsec/configuration/xe-16-7/macsec-xe-16-7-book/wan-macsec-mka-support-enhance.html.

[38] VaibhavRai3. *How Address Resolution Protocol (ARP) works?* URL: https://www.geeksforgeeks.org/how-address-resolution-protocol-arp-works/.

[39] John Burke. *FTP (File Transfer Protocol).* URL: https://searchnetworking.techtarget.com/definition/File-Transfer-Protocol-FTP.

[40] wikipedia.com. *Internet Protocol.* URL: https://en.wikipedia.org/wiki/Internet_Protocol.

[41] wikipedia.com. *Denial-of-service attack.* URL: https://en.wikipedia.org/wiki/Denial-of-service_attack.

[42] CLOUDFLARE. *What is IP Spoofing?* URL: https://www.cloudflare.com/learning/ddos/glossary/ip-spoofing/.

[43] wikipedia.com. *Secure Shell.* URL: https://en.wikipedia.org/wiki/Secure_Shell.

[44] *Port 22 and Relation with SSH Protocol.* URL: https://www.poftut.com/port-22-and-relation-with-ssh-protocol/.

[45] Martins D. Okoi. *The Best 20 Hacking and Penetration Tools for Kali Linux.* 2019-04-22. URL: https://en.wikipedia.org/wiki/Data_breach.

[46] Pill-Won Park Ju-Ho Choi Sung-Gi Min. “Virtual Secure Link over Software-Defined Bridged Networks”. In: *https://gvpress.com/journals/IJCS/vol6$_n$o1/2.pdf* (2019-02-13).

[47] Ki-Ho Gum Jun-Won Lee Seon-Ho Park and Tai-Myoung Chung. “Design of Secure ARP on MACsec(802.1AE)”. In: *https://sci-hub.tw/https://ieeexplore.ieee.org/document/5677881* (2010-10-18).

[48] Jesús Lázaro et al. “MACsec Layer 2 Security in HSR Rings in Substation Automation Systems”. In: *https://addi.ehu.es/bitstream/handle/10810/30386/energies-10-00162.pdf?sequence=1isAllowed=y* (2017-10-31).

[49] Lei Yu Ruichun Gu Xiaolong Zhang and Junxing Zhang. “Enhancing Security and Scalability in Software Defined LTE Core Networks”. In: $https://www.researchgate.net/profile/Ruichun_Gu/publication/327490258_Enhancing_Security_an$ $Security - and - Scalability - in - Software - Defined - LTE - Core - Networks.pdf$ (2018-08-03).