

*Grado en Ingeniería Informática-Ingeniería del Software*

*2021-2022*

*Curso de Seguridad en Sistemas Informáticos y en Internet*

# **PAI 3.**

## **BYODSEC**

### **BRING YOUR OWN DEVICE SEGURO PARA UNA UNIVERSIDAD PÚBLICA USANDO ROAD WARRIOR VPN**

**SECURITY TEAM 11**

*Matilde Ghidini*

*Matteo Halilaga*

*Gabriele Petroni*

# ÍNDICE

<b>INTRODUCCIÓN</b>	<b>3</b>
Objetivos del proyecto	<b>3</b>
<b>TECNOLOGÍAS Y LIBRERÍAS UTILIZADAS</b>	<b>3</b>
Lenguaje de programación: Python	<b>3</b>
Socket	<b>3</b>
SSL	<b>3</b>
Sqlite3	<b>3</b>
Algoritmo SHA256	<b>3</b>
JSON	<b>3</b>
Logging	<b>3</b>
Uuid	<b>4</b>
Hamachi	<b>4</b>
<b>SOLUCIÓN</b>	<b>4</b>
TAREA 1 : Arquitectura Cliente-Servidor Segura con SSL	<b>4</b>
TAREA 2: Análisis de tráfico de red en comunicaciones – Wireshark	<b>4</b>
TAREA 3: Selección e implementación del conjunto de cipher suite más adecuado para la seguridad con SSL/TLS	<b>4</b>

## INTRODUCCIÓN

En este proyecto se pide la realización de un sistema que pueda implementar de la Política de Seguridad Bring your Own Device (BYOD), que consiste en que los empleados utilicen sus propios dispositivos para realizar sus trabajos, pudiendo tener acceso a recursos de la Universidad tales como correos electrónicos, bases de datos y archivos en servidores corporativos usando una VPN SSL. Para la transmisión de todos estos elementos es fundamental la implementación de canales de comunicación seguros.

### Objetivos del proyecto

1. Desarrollar/seleccionar cómo llevar a la práctica de forma lo más eficiente posible los canales de comunicación segura para la transmisión de credenciales (usuario, contraseñas) y un mensaje con el Protocolo SSL/TLS (autenticidad, confidencialidad e integridad). Tener en cuenta que el número de empleados que usarán la aplicación son aproximadamente 300.
2. Utilizar alguna herramienta de análisis de tráfico que permita comprobar la confidencialidad e integridad de los canales de comunicación seguros.
3. Establecer los Cipher Suites que serán usados en la versión TLS 1.3. Además, el cliente nos solicita pruebas sobre la capacidad para soportar a los 300 empleados por la VPN SSL desarrollada. Nota: Es muy importante para el cliente que la implementación de la VPN que se implemente sea lo más eficiente posible. Considerando eficiente aquella solución con bajo overhead, baja

## TECNOLOGÍAS Y LIBRERÍAS UTILIZADAS

### Lenguaje de programación: Python

Hemos decidido desarrollar nuestro proyecto utilizando el lenguaje Python.

### Socket

Hemos utilizado la librería Python socket, para implementar construir nuestra arquitectura cliente-servidor.

### SSL

Hemos utilizado la librería Python ssl para implementar una conexión segura entre cliente y servidor.

### Sqlite3

Para crear una base de datos donde almacenar usernames y passwords de los usuarios, hemos utilizado la librería Python Sqlite3, que es un base de datos SQL self-contained y file-based.

### JSON

Hemos utilizado la librería Python json, para la serialización y deserialización de los datos enviados del cliente hasta el servidor.

### Hamachi

Hamachi es un software de virtualización de redes que permite emular una red local (LAN) a los dispositivos conectados por WAN. Con Hamachi se puede generar una red local aunque los dispositivos se encuentren en distintos lugares repartidos por el mundo. Para ello, Hamachi hace uso de redes privadas virtuales (VPN). Hemos utilizado Hamachi para simular el funcionamiento de nuestro programa en diferentes dispositivos.



No.	Time	Source	Destination	Protocol	Length	Info
1	0.00000	127.0.0.1	127.0.0.1	TCP	66	52061 -> 10035 [FIN] Seq=6155535 Len=0 MSS=16344 WS=64 TSval=155551569 TSecr=0 Seq_Perm=1
2	0.00034	127.0.0.1	127.0.0.1	TCP	66	10035 -> 52061 [FIN, ACK] Seq=6155535 Len=0 MSS=16344 WS=64 TSval=130253049 TSecr=155551569 Seq_Perm=1
3	0.000116	127.0.0.1	127.0.0.1	TCP	56	52061 -> 10035 [ACK] Seq=61548216 Len=0 TSval=155551569 TSecr=130253049
4	0.000125	127.0.0.1	127.0.0.1	TCP	56	[TCP Window update] 10035 -> 52061 [ACK] Seq=61548216 Len=0 MSS=16344 WS=64 TSval=130253049 TSecr=155551569
5	0.000729	127.0.0.1	127.0.0.1	TLSv3.0	573	Client Hello
6	0.000872	127.0.0.1	127.0.0.1	TCP	56	10035 -> 52061 [ACK] Seq=61538 Len=0 TSval=130253049 TSecr=155551569
7	0.000471	127.0.0.1	127.0.0.1	TLSv3.0	1529	Server Hello, Change Cipher Spec, Application Data, Application Data, Application Data
8	0.000532	127.0.0.1	127.0.0.1	TCP	56	52061 -> 10035 [ACK] Seq=61548 Len=0 TSval=155551578 TSecr=130253049
9	0.000578	127.0.0.1	127.0.0.1	TLSv3.0	138	Change Cipher Spec, Application Data
10	0.000656	127.0.0.1	127.0.0.1	TCP	56	10035 -> 52061 [ACK] Seq=61548 Len=0 TSval=130253049 TSecr=155551578
11	0.000658	127.0.0.1	127.0.0.1	TLSv3.0	131	Application Data
12	0.000665	127.0.0.1	127.0.0.1	TCP	56	10035 -> 52061 [ACK] Seq=615474 Len=0 TSval=130253049 TSecr=155551578
13	0.000734	127.0.0.1	127.0.0.1	TLSv3.0	511	Application Data
14	0.000801	127.0.0.1	127.0.0.1	TCP	56	52061 -> 10035 [ACK] Seq=615720 Len=0 TSval=155551578 TSecr=130253049
15	0.000810	127.0.0.1	127.0.0.1	TLSv3.0	311	Application Data
16	0.000813	127.0.0.1	127.0.0.1	TCP	56	10035 -> 52061 [ACK] Seq=615804 Len=0 TSval=155551578 TSecr=130253049
17	0.000953	127.0.0.1	127.0.0.1	HTTP	87	Application Data
18	0.000975	127.0.0.1	127.0.0.1	TCP	56	52061 -> 10035 [ACK] Seq=615815 Len=0 TSval=155551578 TSecr=130253049
19	0.01000	127.0.0.1	127.0.0.1	TCP	56	10035 -> 52061 [FIN, ACK] Seq=615805 Len=0 TSval=155551578 TSecr=130253049
20	0.01026	127.0.0.1	127.0.0.1	TCP	56	10035 -> 52061 [ACK] Seq=615464 Len=0 TSval=130253049 TSecr=155551578

Timestamp echo reply: 155551579

A la hora de realizar la comunicación entre cliente y servidor, se puede ver en la captura de tráfico la lista de todos los posibles cipher suites que se pueden utilizar.

