



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

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

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 filebased.

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.

SOLUCIÓN

TAREA 1: Arquitectura Cliente-Servidor Segura con SSL

El primer paso para poder implementar el protocolo SSL es la generación de claves y certificados auto firmados. Para eso, se instala openssl, se genera primero una clave privada, luego un CSR y finalmente un certificado.

El siguiente paso es la construcción de una arquitectura cliente-servidor, utilizando socket ssl y configurando respectivamente los certificados del cliente y del servidor.

Entonces, el cliente intenta conectarse al servidor.

Una vez que servidor y cliente están conectados, el cliente envía username y password.

El servidor comprueba que username y password sean correctos, buscándolos en una base de datos.

Si no están correctos, el servidor termina la conexión.

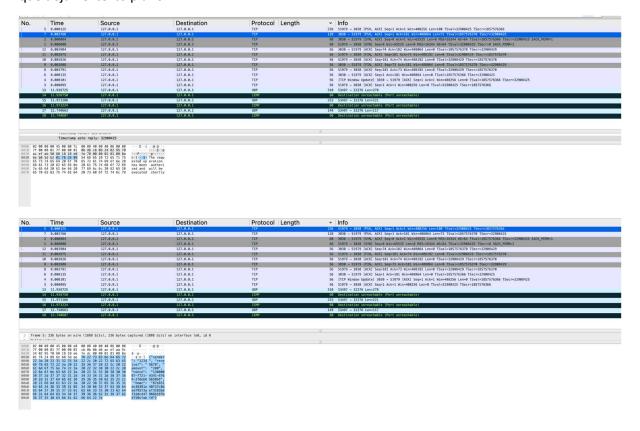
Si los datos son correctos el cliente puede empezar a enviar mensajes.

Basándose la comunicación en socket ssl, se garantiza la integridad, la confidencialidad y la autenticidad de los datos enviados.

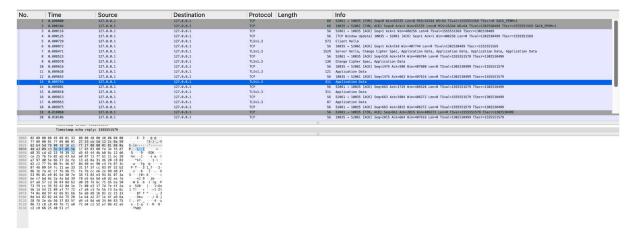
TAREA 2: Análisis de tráfico de red en comunicaciones – Wireshark

Hemos utilizado Wireshark para comprobar que los paquetes enviados en la red, están correctamente cifrados.

En las siguientes dos imágenes hemos capturado el tráfico entre el cliente y el servidor del PAI2, que no necesitaba ocultar los datos de los paquetes a través de ssl. Por eso, se utilizan socket sin cifrado que dejan el texto plano.

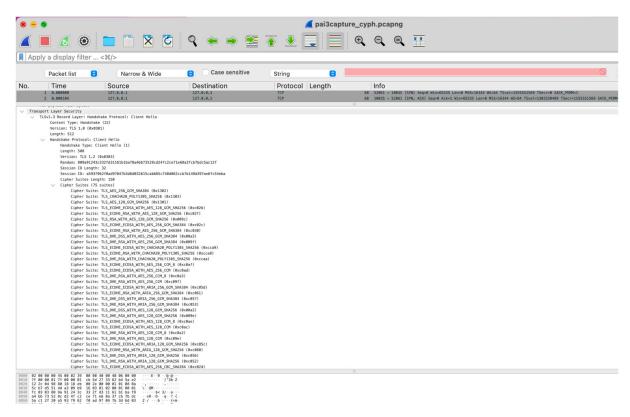


En la siguiente imagen, utilizando socket ssl (PAI3), se puede ver que los datos de los paquetes están ocultados y no se puede sacar la información de los paquetes que viajan en la red .



TAREA 3: Selección e implementación del conjunto de cipher suite más adecuado para la seguridad con SSL/TLS

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.



En este caso el servidor ha utilizado TLS_AES_256_GCM_SHA384 (0x1302).

Este cipher suite es uno de los más seguros, entonces no se necesita cambiarlo.