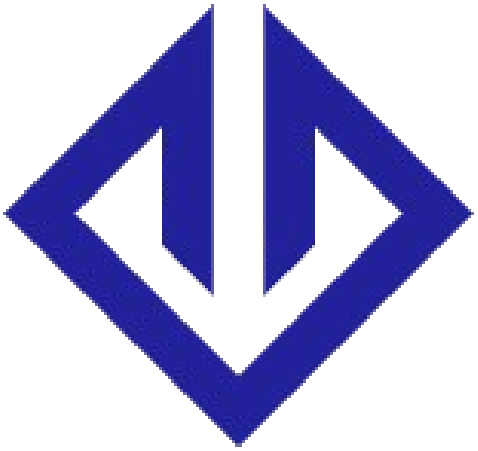
Institut Puig Castellar
Santa Coloma de Gramenet



Kontor Tek Inc.

(Projecte de desenvolupament)
CFGS Administració de Sistemes Informàtics i Xarxes

Autors: Yang Yun Chen, Di Lu, Hoayang Xu
Grup: 2
Curs acadèmic: CFGS ASIX

B) GNU Free Documentation License (GNU FDL)

C) Copyright

Resum del projecte:

Kontor Tek Inc ofrece dos servicios: un programa de seguridad que escanea la red, bloquea IPs y gestiona listas, y un servicio de copias de seguridad controlable desde terminal o Telegram.

Paraules clau:

Seguretat informàtica

Còpies de seguretat

Servidor web

Xarxa local

Vulnerabilitats

Telegram Bot

Abstract:

Kontor Tek Inc offers two services: a security program that scans the network, blocks IPs and manages lists, and a backup service that can be controlled from a terminal or Telegram.

Keywords:

Computer Security

Backups

Web Server

Local Area Network

Vulnerabilities

Telegram Bot

Índex

Llista de figure

Introducción

El presente proyecto tiene como finalidad el diseño, desarrollo y lanzamiento de una plataforma web comercial que ofrecerá servicios de seguridad informática y almacenamiento en la nube mediante un modelo de suscripción. La solución integral busca dar respuesta a la creciente necesidad de protección de datos y seguridad de red, tanto para usuarios particulares como para pequeñas y medianas empresas.

La plataforma ofrecerá:

- Un programa de seguridad que permitirá a los usuarios escanear sus redes locales para identificar dispositivos conectados, detectar posibles vulnerabilidades y gestionar el acceso a la red mediante un bloqueo de direcciones IP sospechosas en una lista negra automatizada.

- Un sistema que realizará copias de seguridad periódicas de los datos de los usuarios, almacenándolos de forma segura y encriptada en servidores remotos. Los clientes tendrán control total sobre la frecuencia de las copias y podrán acceder y restaurar sus datos en cualquier momento.

El objetivo es crear un servicio unificado, fácil de utilizar y accesible, que permita a los usuarios sin conocimientos técnicos avanzados mejorar significativamente su postura de ciberseguridad y garantizar la integridad y disponibilidad de su información crítica.

a. Contexto

En el contexto tecnológico actual, la ciberseguridad se ha convertido en una necesidad fundamental para cualquier organización, independientemente de su tamaño. El constante aumento de ataques a redes locales, intrusiones no autorizadas y pérdida de datos críticos ha puesto de manifiesto la importancia de disponer de herramientas de protección accesibles y eficientes. Sin embargo, la mayoría de soluciones profesionales del mercado tienen un coste elevado o requieren conocimientos técnicos avanzados para ser implementadas, lo que las hace poco accesibles para pequeñas empresas o entornos educativos.

Ante esta situación, el proyecto Kontor Tek Inc. nace con el objetivo de desarrollar una plataforma propia de seguridad y backup que sea funcional, accesible y fácil de usar. La solución integra un detector de malware capaz de escanear la red y bloquear dispositivos no autorizados, combinado con un servicio de copias de seguridad local y un sistema de notificaciones y control remoto mediante Telegram. Todo esto se presenta a través de una página web que centraliza el acceso a los servicios, la descarga del software y la documentación necesaria para su instalación y uso.

b. Justificación

La principal motivación para desarrollar este proyecto es la carencia de soluciones de ciberseguridad asequibles y adaptadas a entornos con recursos limitados. Las herramientas existentes en el mercado suelen ser costosas, complejas de configurar o dependen de servicios externos que no están siempre disponibles. Nosotros pretendemos cubrir ese vacío ofreciendo una solución autónoma, instalable en cualquier máquina Linux, que no requiere infraestructura adicional ni suscripciones a servicios de terceros para funcionar.

Además, la integración del control remoto mediante Telegram representa un valor añadido significativo, ya que permite gestionar la seguridad de la red desde cualquier dispositivo móvil de forma inmediata, sin necesidad de acceder físicamente a la máquina. Esto es especialmente relevante en situaciones donde es necesario actuar rápidamente ante una amenaza detectada, como el bloqueo de una IP sospechosa o la consulta del estado del sistema en tiempo real.

Por último, el proyecto también tiene una dimensión formativa importante. Su desarrollo ha permitido profundizar en áreas clave de la informática como la seguridad en redes, la programación en Python, la administración de sistemas Linux y el desarrollo web con PHP y MySQL. Esto le convierte en un proyecto de síntesis que integra de forma práctica los conocimientos adquiridos a lo largo del curso.

c. Objetivos

i. Generales

- Desarrollar el servidor web con diferentes páginas (uno de presentación donde explicaremos sobre nuestro servicio, y otro donde están las opciones de suscripción).
- Desarrollar el programa de seguridad con las funciones mencionadas y que funcione correctamente.

ii. Específicos

- Diseñar una base de datos para registrar a los usuarios con suscripción.
- Implementar las funciones de escaneo, búsqueda de información y bloqueo de IPs en el programa de seguridad.
- Implementar un sistema de detección automática de comportamientos sospechosos con bloqueo inmediato.
- Diseñar y maquetar las páginas web.
- Configurar el sistema de copias de seguridad.
- Crear los paquetes de instalación (install.sh) y desinstalación (uninstall.sh) para desplegar toda la infraestructura.

d. Estratègia i planificación del projecte

Para realizar el proyecto se consideraron diversas estrategias. La primera opción era adaptar una solución de ciberseguridad ya existente, personalizándolas según las necesidades del proyecto. La segunda opción, finalmente elegida, era desarrollar un producto completamente nuevo y propio, diseñado desde cero para integrar de forma coherente la detección de malware, el control remoto por Telegram y el servicio de backup.

La estrategia de desarrollo propio se consideró la más adecuada por diversas razones. En primer lugar, permite tener un total control sobre el funcionamiento de cada componente, facilitando la integración entre módulos y la adaptación a requisitos específicos que las soluciones existentes no cubrían de forma conjunta. En segundo lugar, construir la herramienta desde cero tiene un valor formativo muy superior, puesto que obliga a comprender y resolver cada problema técnico en profundidad, desde la detección de paquetes en la red como el despliegue de un servidor web con base de datos.

En cuanto a la viabilidad, el proyecto se planificó en fases progresivas: primero el desarrollo y pruebas de los módulos principales en local, después la integración entre componentes y finalmente el despliegue de la plataforma web conectada a la base de datos. Esta planificación incremental permitió detectar y corregir errores en cada fase antes de avanzar a la siguiente, reduciendo el riesgo de bloqueos mayores. Los recursos necesarios, como una máquina virtual Ubuntu, Visual Studio Code y herramientas de código abierto como Python y PHP, eran totalmente accesibles, confirmando la viabilidad técnica y económica del proyecto.

e. Metodologia de trabajo

Para el desarrollo de este proyecto hemos adoptado una metodología de trabajo incremental e iterativa, adaptada al contexto educativo del ciclo formativo. Cada semana registramos las actividades realizadas, las tareas pendientes, las dificultades encontradas y las soluciones aplicadas.

Las características principales de la metodología empleada son:

- Reuniones y puestas en común semanales del grupo para hacer seguimiento del estado de las tareas y resolver los problemas encontrados.
- Uso de Git para el control de versiones del código fuente.
- División de tareas entre los miembros del equipo por áreas de conocimiento: servidor web y base de datos, programa de seguridad en Python, y sistema de instalación e integración.
- Revisión y adaptación del proyecto a mitad de camino tras una reunión con los tutores, donde se redefinió el alcance y se priorizaron las funcionalidades.
- Uso de máquinas virtuales para un entorno de pruebas aislado y reproducible.
- Documentación progresiva a lo largo de todas las fases del proyecto.

Descripción del proyecto

f. Anàlisis de requisitos

i. Funcionales

- El sistema permitirá la selección y gestión de planes de suscripción (mensual/anual) y el procesamiento de pagos.
- El sistema permitirá el bloqueo y desbloqueo de direcciones IP por la Red.
- El sistema implementará un programa de seguridad capaz de escanear la red para detectar posibles vulnerabilidades.
- El sistema generará informes de seguridad.
- El sistema permitirá la configuración de la frecuencia y el alcance de las copias de seguridad.
- El sistema permitirá a los usuarios acceder y gestionar sus copias de seguridad almacenadas.
- El sistema implementará un trigger que se active automáticamente al detectar vulnerabilidades, bloqueando la IP y poniéndola en una lista negra.

ii. No Funcionales

- La interfaz será accesible desde dispositivos móviles.
- Las páginas deben cargarse en menos de tres segundos.
- El sistema debe ser compatible con los navegadores web más utilizados (Chrome, Firefox, Edge, Safari, Brave).
- El sistema debe utilizar conexiones seguras (HTTPS) y aplicar medidas de protección contra ataques comunes
- El paquete de instalación debe ser compatible con sistemas Linux (formato Unix, no Windows).
- La aplicación debe funcionar con privilegios de superusuario (root) para garantizar el acceso a las interfaces de red ya nftables.

g. Tecnologies

i. Comparativa de les tecnologies valorades

Para el servidor web, valoramos tres opciones principales: Apache, Nginx y Node.js. Apache resultó la opción más adecuada para el proyecto, ya que es el servidor web estudiado en el ciclo ASIX, dispone de una documentación muy amplia y ofrece una integración nativa y sencilla con PHP. Por su parte, Nginx también lo estudiamos durante el curso y es una alternativa potente con buen rendimiento, pero su configuración es más compleja, por eso escogimos Apache. Por último, Node.js se descartó porque no tenemos mucho conocimiento sobre él y porque no es compatible de forma nativa con PHP, que era uno de los requisitos del proyecto.

Para la base de datos comparamos entre MySQL, PostgreSQL y SQLite, y escogimos MySQL porque es el sistema de bases de datos que hemos utilizado durante el segundo año del ciclo ASIX, tiene una integración muy buena con PHP a través de PDO, y ofrece herramientas gráficas como phpMyAdmin que facilitan su gestión. PostgreSQL es un sistema más potente y escalable, y también lo hemos trabajado durante estos dos años, pero su configuración es más compleja. SQLite se descartó por su baja escalabilidad y por no ser adecuado para un entorno de servidor web con múltiples usuarios concurrentes.

Para el programa de seguridad, Python fue el lenguaje de programación escogido de forma clara, ya que dispone de bibliotecas específicas para la manipulación de redes como Scapy y para la integración con la API de Telegram, además de ser un lenguaje estudiado en el ciclo. Para las notificaciones se valoró el uso de correo electrónico o de Telegram, y finalmente se optó por Telegram por su facilidad de integración mediante API y por la inmediatez de las notificaciones.

ii. Tecnologies escollides

- **Visual Studio Code:** IDE principal para todo el código del proyecto (HTML, CSS, PHP, Python, Shell Script).
- **Python:** Desarrollo del programa de seguridad (escaneo de red, detección de vulnerabilidades, bloqueo de IPs).
- **Shell Script:** Para aplicar las instalaciones y desinstalaciones del programa.
- **HTML, CSS y JS:** Creación y diseño de las páginas web.
- **PHP:** Generación de páginas web dinámicas y conexión con la base de datos.
- **MySQL:** Base de datos para gestionar los usuarios suscritos y los datos de la aplicación.
- **Apache:** Servidor web que gestiona las peticiones HTTP/HTTPS entre el servidor y los clientes.

- **python-telegram-bot:** Integración con la API de Telegram para alertas y control remoto.

h. Estructura del projecte

El projecte s'organitza en quatre blocs principals interconnectats que conformen una solució integral de seguretat i còpies de seguretat per a sistemes Linux.

El primer bloc és el **Servidor Web**, basat en Apache sobre Ubuntu Server amb PHP i MySQL, que actua com a punt d'accés central per als usuaris: gestiona el registre, les subscripcions i la descàrrega del programari.

El segon bloc és el **Programa de Seguretat** (malware_detector.py), executat com a servei de sistema (systemd) amb privilegis de root. Monitoritza el tràfic de xarxa en temps real, detecta comportaments sospitosos (escaneig de ports SYN, ràfegues DDoS, atacs ARP) i aplica regles de bloqueig a nftables, desant les IPs a blacklist.txt.

El tercer bloc és el **Bot de Telegram** (telegram_bot.py), també executat com a servei systemd, que proporciona una interfície de control remot. Permet a l'administrador autoritzat consultar l'estat del sistema i gestionar les llistes negra i blanca directament des del mòbil.

El quart bloc és l'**Eina de Còpies de Seguretat** (malware_backup_tool.py), una aplicació gràfica d'escriptori (Python + Tkinter) que permet crear i restaurar còpies comprimides en format .back de tots els components crítics del sistema.

Tots aquests components es despleguen mitjançant els scripts install.sh i uninstall.sh, que automatitzen la instal·lació de dependències, la configuració dels serveis i la creació de les comandes de terminal.

i. Descripció de components

i. Servidor Web

La página web de Kontor Tek Inc. está formada por varios componentes que trabajan conjuntamente para ofrecer una experiencia completa al usuario. A continuación se describe cada componente de forma individual.

Frontend (HTML, CSS, JavaScript)

Es la capa visible de la aplicación, la parte con la que interactúa el usuario directamente. Está compuesta por tres páginas principales: index.html (página principal), login.html (acceso y registro) y manual.html (documentación). El diseño sigue una estética cyberpunk con tipografías monoespacio, paleta de colores en cian y verde neón sobre fondo oscuro, y elementos visuales como cuadrículas animadas y efectos de cursor. JavaScript se encarga de la interactividad: gestión de modales, validación de formularios en tiempo real, comunicación con el backend mediante fetch y control del estado de sesión del usuario.

Backend (PHP)

El backend gestiona toda la lógica de servidor a través de cinco endpoints:

- login.php — Autentica al usuario comparando las credenciales con la base de datos. Utiliza password_verify() para comprobar el hash bcrypt e inicia la sesión PHP.
- register.php — Registra nuevos usuarios en la base de datos. Valida el formato del email, comprueba duplicados y almacena la contraseña cifrada con password_hash().
- check-session.php — Verifica si hay una sesión activa y si el usuario tiene una suscripción vigente. Es llamado por el botón de descarga y por los botones de selección de plan.
- subscribe.php — Gestiona la contratación de un plan de suscripción por parte de un usuario autenticado.
- download.php — Sirve el archivo de instalación solo si el usuario tiene sesión activa y suscripción vigente.

Base de datos (MySQL)

Almacena toda la información persistente de la aplicación. Las tablas principales son:

- Usuari — Datos de los usuarios registrados (nombre, email, contraseña hash, fecha de registro, estado de suscripción).
- Subscripcio — Registro de las suscripciones activas (tipo de plan, fechas de inicio y fin, precio, estado).
- Backup, InformeSeguretat, MACBloqueada, Vulnerabilitat — Tablas relacionadas con el programa de seguridad.

Servidor web (Apache)

Apache actúa como intermediario entre el cliente y la aplicación, recibiendo las peticiones HTTP de los navegadores y sirviendo los archivos estáticos (HTML, CSS, JS) o ejecutando los scripts PHP correspondientes. Está instalado en una máquina virtual Ubuntu y aloja todos los archivos del proyecto bajo el directorio /var/www/html/.

ii. Programa de Seguretat (malware_detector.py)

El programa de seguridad es el núcleo de la plataforma. Ejecutado como servicio de sistema (systemd) con privilegios de root, monitoriza de forma continua el tráfico de red en la interfaz configurada.

Los mecanismos de detección implementados son:

1. **Detección de escaneo SYN:** Si una IP envía más de 100 paquetes SYN a puertos diferentes en una ventana de 5 segundos, se considera un intento de escaneo de puertos y la IP será bloqueada automáticamente.
2. **Detección de ráfagas de paquetes grandes:** Si una IP envía 5 o más paquetes de tamaño superior a 1.500 bytes en 10 segundos, se clasifica como un posible ataque DDoS o exfiltración de datos y es bloqueada.
3. **Ataques tipo ARP**

Los comandos disponibles son:

Comanda	Descripció
sudo md-blocklist	Muestra la lista de IPs actualmente bloqueadas.
sudo md-block <ip>	Bloquea la IP especificada
sudo md-unblock <ip>	Desbloquea la IP especificada
sudo md-whitelist <ip>	Muestra la lista de IPs autorizadas
sudo md-whitelist-add <ip>	Agrega una IP específica a la whitelist (la eliminará del blacklist si dicha IP se encuentra en él).
sudo md-whitelist-del <ip>	Elimina una IP específica de la whitelist.

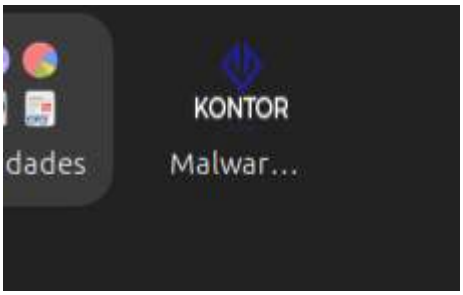
iii. Bot de telegram (telegram_bot.py)

El bot de Telegram proporciona una interfaz de control remoto para el administrador. Únicamente el Chat ID autorizado puede enviar comandos.
Los comandos disponibles son:

Comanda	Descripció
/start	Inicia el servicio y muestra los botones para gestionar el sistema
/status	Muestra las IPs actualmente bloqueadas
/block <ip>	Bloquea la IP especificada
/unblock <ip>	Desbloquea la IP especificada
/whitelist_add	Agrega una IP específica a la whitelist.
/whitelist_del	Elimina una IP específica de la whitelist

iv. Sistema de Còpies de Seguretat (malware_backup_tool.py)

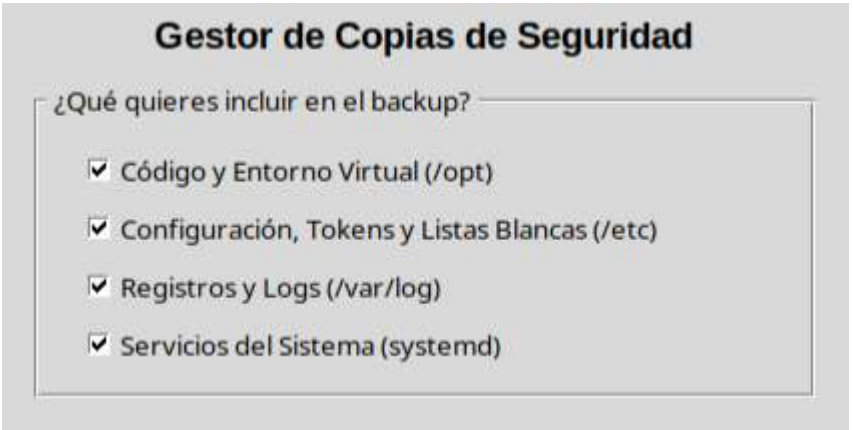
Esta herramienta de copias de seguridad requiere autenticación de root para operar.



Es una aplicación gráfica de escritorio (python + Tkinter) que permite crear y restaurar copias comprimidas en formato back (tar.gz) de todos los componentes del sistema.

Los componentes que se pueden incluir en la copia de seguridad son:

- Código y entorno virtual Python (/opt/malware_detector)
- Configuración, tokens y listas blanca/negra (/etc/malware_detector)
- Registros y logs (/var/log/malware_detector)
- Servicios del sistema systemd (malware-detector.service, telegram-bot.service)



v. Paquet d'instalació i de desinstalació (install.sh i uninstall.sh)

El script `install.sh` automatiza completamente el despliegue de toda la infraestructura. Se ejecuta con privilegios de root y realiza las siguientes operaciones de forma secuencial:

- Verificación de privilegios de root.
- Instalación de dependencias del sistema (python3, scapy, nftables, tcpdump, etc.).
- Creación de la estructura de directorios del sistema.
- Copia y configuración de todos los archivos del proyecto.
- Creación de un entorno virtual Python e instalación de las dependencias (scapy, python-telegram-bot, requests).
- Configuración interactiva del token de Telegram, el Chat ID y la interfaz de red.
- Generación automática de la lista blanca con la IP de la interfaz y la IP del administrador (SSH).
- Creación y activación de los servicios systemd (malware-detector y telegram-bot).
- Instalación de los comandos de terminal (md-block, md-unblock, md-blocklist, md-whitelist, md-whitelist-add, md-whitelist-del).
- Creación de un acceso directo gráfico para la herramienta de copias de seguridad.

El script `uninstall.sh` es un desinstalador completo e interactivo. Pregunta al usuario qué componentes desea eliminar y cuáles conservar, y se encarga de detener los servicios, eliminar las reglas de nftables, borrar los archivos y limpiar las dependencias de Python si corresponde.

j. Definició de funcionalitats

i. Funció 1

Detección de malware y notificación vía Telegram

El programa Python monitoriza el sistema y la red de forma continua. Al detectar una anomalía o amenaza, el sistema realiza las siguientes acciones:

- Identifica la IP origen del comportamiento sospechoso no autorizado.
- Aplica inmediatamente una regla de bloqueo en nftables (INPUT y FORWARD).
- Guarda la IP en el archivo blacklist.txt para garantizar la persistencia entre reinicios.
- Envía una notificación automática al usuario vía bot de Telegram con la IP, el motivo del bloqueo y la hora del incidente, preguntando además si se desea autorizarla o no en caso de que fuera un malentendido.

ii. Funcio 2

Gestión de copias de seguridad

La herramienta de copias de seguridad permite al administrador proteger la configuración del sistema de forma sencilla desde una interfaz gráfica. Las operaciones principales disponibles son:

- **Crear Backup (.back):** El usuario selecciona los componentes a incluir y elige la ruta de destino. La herramienta genera un archivo comprimido (.back / tar.gz) con todos los archivos seleccionados.
- **Restaurar desde .back:** El usuario selecciona un archivo de copia de seguridad previamente creado. El sistema detiene los servicios, extrae los archivos a sus ubicaciones originales y reinicia los servicios automáticamente.

Desenvolupament

k. Programa de Seguretat “Malware detector y Telegram”

La herramienta utilizada para desarrollar el programa fue Visual Studio Code, desde donde gestionamos todo el proyecto. Primero empezamos creando el módulo malware_detector.py, definiendo las reglas de detección de red para diferentes tipos de comportamiento de ataques. Después creamos telegram_bot.py, implementando las funciones básicas y la comunicación entre dispositivos móviles y la terminal.

A partir de aquí, realizamos pruebas de conectividad entre ambos módulos para verificar que cuando telegram_bot.py enviaba un comando, malware_detector.py pudiera identificarlo correctamente. También configuramos un entorno con Systemd para monitorizar el estado de cada servicio. Luego creamos el ejecutable install.sh, encargado de instalar todas las dependencias necesarias, crear la estructura de directorios con los archivos requeridos (whitelist.txt, blacklist.txt y detector.log) y los scripts de comandos para la terminal. De forma complementaria, desarrollamos uninstall.sh para desinstalar todos los archivos y directorios generados por install.sh.

En el siguiente paso, configuramos telegram_bot.py para que sincronizará los comandos de la terminal con el chat de Telegram, asegurándonos de que ambos programas ejecutarán el mismo tipo de comandos sin conflictos entre ellos. Cuando se usa un comando del script configurado, este ejecuta directamente la instrucción correspondiente en nftables para bloquear o desbloquear, y devuelve el estado del cambio de forma inmediata.

Por último, una vez establecida la estructura base, realizamos una serie de mejoras: cualquier comando ejecutado devuelve un mensaje indicando si se completó correctamente o no; se sincronizó nftables con blacklist.txt para facilitar la comprobación de bloqueos y con whitelist.txt para permitir automáticamente todas las IPs autorizadas; y se añadieron más interacciones en Telegram, incluyendo botones, un mini manual de comandos, alertas de seguridad, validaciones en casos específicos y visualización del estado de las listas de blacklist y whitelist.

Problemas encontrados en el proceso de creación:

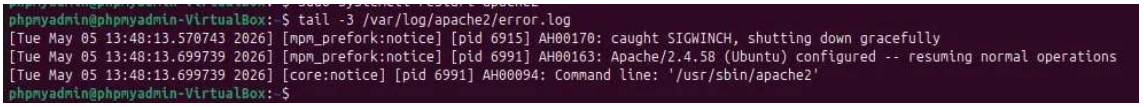
- **Desincronización:** Al desbloquear una IP desde Telegram, md-blocklist en la terminal seguía mostrando la IP como bloqueada, indicando una falta de sincronización entre el estado en memoria, el archivo y nftables. Para solucionarlo, implementamos un manejador de señales (SIGHUP) en malware_detector.py para recargar las listas al instante. Los comandos md-block y md-unblock se actualizaron para notificar al servicio y asegurar que el archivo de IPs bloqueadas (blacklist.txt) se mantuviera sincronizado.
- **Detección errónea del usuario de Telegram:** Al usar comandos desde Telegram en una red diferente, el sistema detectaba al usuario como intruso y saltaba la alarma. Para solucionarlo, se añadió una regla para no bloquear al usuario de Telegram enlazado al token, de forma que el sistema pudiera diferenciar entre tráfico legítimo y amenazas reales.

- **Bloqueo accidental de IPs en la whitelist:** Al intentar bloquear una IP que ya estaba en la lista blanca a través de Telegram, se bloqueaba directamente sin advertencia. Para solucionarlo, se implementó una confirmación interactiva en telegram_bot.py. Si la IP está en la whitelist, el bot pregunta al usuario si realmente desea moverla a la blacklist, ofreciendo los botones "Sí" y "No".
- **Vulnerabilidad a ataques ARP Spoofing:** El sistema no detectaba ni defendía contra ataques de suplantación ARP en la red local. Para solucionarlo, se añadió un nuevo motor de detección en malware_detector.py para monitorizar paquetes ARP. Este motor detecta cambios de MAC sospechosos para IPs conocidas, especialmente el Gateway, y bloquea al atacante. Las alertas de Telegram se actualizaron para incluir la dirección MAC del atacante.

I. Pagina Web

La web se estructuró en varias páginas separadas: una página principal (index.html) con secciones de presentación, programa, planes de suscripción y descarga; una página de acceso (login.html) con sistema de login y registro por pestañas; y una página de manual de instalación (manual.html). El diseño sigue una estética cyberpunk/hacker con fondo oscuro, cuadrícula animada, tipografías monoespacio y una paleta de colores basada en cian y verde neón. Para la interactividad se implementaron modales de descarga con tres estados según el estado de sesión del usuario, modales de confirmación de plan de suscripción, e indicador de fortaleza de contraseña en tiempo real.

Las tecnologías empleadas fueron HTML, CSS y JavaScript vanilla en el frontend, con especial atención a la compatibilidad entre navegadores. En el backend se utilizó PHP 8 con PDO para la conexión a la base de datos MySQL alojada en una máquina virtual Ubuntu con servidor Apache. La base de datos KontorTekInc contiene tablas para usuarios, suscripciones, backups, informes de seguridad, MACs bloqueadas y vulnerabilidades. Los endpoints desarrollados cubren login, registro, verificación de sesión, suscripción a planes y descarga protegida del software, todos comunicados mediante fetch con JSON.



Durante el desarrollo se encontraron varios errores relevantes. El más recurrente fue la corrupción de caracteres especiales en los archivos JavaScript al subirlos al servidor Apache, ya que los comentarios con caracteres UTF-8 rompían completamente la ejecución del JS. La solución fue mover todo el JS inline dentro del HTML y usar exclusivamente ASCII en el código. También se detectaron problemas de permisos en Apache (Permission denied) que se resolvieron con chown www-data sobre los archivos PHP, y un fallo de credenciales en MySQL que requirió crear un usuario nuevo desde phpMyAdmin al no tener acceso al root por contraseña. Finalmente, los modales no se mostraban correctamente en Firefox por el uso de la propiedad inset no soportada en versiones antiguas, que se sustituyó por top/left/width/height, y las llamadas a check-session.php no enviaban cookies de sesión al usar Access-Control-Allow-Origin: *, problema resuelto eliminando las cabeceras CORS al ser frontend y backend del mismo origen.

m. Copies de Seguretat (Backups)

Para desarrollar el sistema de copias de seguridad, creamos una aplicación de escritorio utilizando Python y la librería gráfica Tkinter. El objetivo era permitir al administrador guardar y restaurar toda la configuración y datos del programa de seguridad de forma visual e intuitiva, sin necesidad de interactuar directamente con la consola. Esta herramienta resultó necesaria porque el sistema de seguridad modifica archivos críticos del sistema operativo, como las reglas de nftables, las listas negras y blancas, los tokens de la API de Telegram y los servicios de systemd. En caso de una actualización fallida o un cambio de servidor, el administrador necesitaba una forma rápida y segura de devolver el sistema a su estado funcional.

La aplicación empaqueta todos estos componentes en un único archivo comprimido con extensión .back (formato tar.gz), permitiendo seleccionar qué elementos incluir en la copia: código y entorno virtual (/opt/malware_detector), configuración y listas (/etc/malware_detector), registros (/var/log/malware_detector) y los servicios de systemd. Para el manejo de archivos comprimidos se utilizó el módulo tarfile, y para la interacción con los servicios del sistema se empleó subprocess para ejecutar comandos systemctl.

El principal reto técnico fue garantizar que la aplicación solo se ejecutara con privilegios de root, ya que tanto la lectura de ciertos directorios como la gestión de los servicios de systemd lo requieren. Se añadió una comprobación inicial con os.geteuid() que muestra un mensaje de error si no se ejecuta con sudo. Además, se implementó una lógica crítica en el proceso de restauración: el sistema debe detener automáticamente los servicios antes de sobrescribir los archivos y volver a iniciarlos una vez finalizada la descompresión, para evitar conflictos en la memoria.

Otro reto importante fue la integración visual con el entorno de escritorio. En versiones anteriores, la aplicación se iniciaba como root = tk.Tk(), lo que provocaba que en sistemas Linux modernos no tuviera icono identificativo en el dock. En la versión final se cambió a root = tk.Tk(className='Malware-backup'), cumpliendo con el estándar de GNOME/Ubuntu y permitiendo que el sistema operativo asocie la ventana con su archivo .desktop e icono correspondiente. Este proceso iterativo permitió entregar una herramienta robusta que no solo cumple su función técnica, sino que también se integra correctamente en el entorno gráfico y ofrece una experiencia estable y profesional.

Conclusions

n. Conclusió general

El desarrollo del proyecto Kontor Tek Inc. ha resultado ser una experiencia muy completa tanto a nivel técnico como formativo. Se ha conseguido diseñar, desarrollar y desplegar una plataforma funcional que integra un programa de detección de malware, un bot de control remoto por Telegram, un sistema de copias de seguridad y una página web con gestión de usuarios y suscripciones. El resultado final demuestra que es posible construir una solución de ciberseguridad real, operativa y accesible con herramientas de código abierto y los conocimientos adquiridos durante el ciclo formativo ASIX. El proyecto ha superado varios de los objetivos iniciales dejando solo un objetivo (Panel de controll para programa de seguridad) a medias por la falta de tiempo y errores encontrados durante el desarrollo de este.

o. Consecució dels objectius

Mayoritariamente todos los objetivos definidos al inicio del proyecto han sido alcanzados satisfactoriamente:

Servidor web: Se ha desarrollado y desplegado correctamente con Apache, PHP y MySQL, incluyendo las páginas de presentación, suscripción, acceso y manual de instalación.

Programa de seguridad: El módulo malware_detector.py detecta y bloquea automáticamente comportamientos sospechosos en la red mediante tres mecanismos: escaneo SYN, ráfagas DDoS y ataques ARP Spoofing.

Base de datos: Se ha diseñado e implementado el esquema de base de datos KontorTekInc con todas las tablas necesarias para gestionar usuarios, suscripciones y datos del programa.

Escaneo y bloqueo de IPs: Implementados y funcionales tanto desde la terminal como desde Telegram.

Sistema de detección automática: Operativo con bloqueo inmediato y notificación instantánea al administrador.

Diseño y maquetación web: Completado con una estética cyberpunk coherente y adaptada a los requisitos del proyecto.

Sistema de copias de seguridad: Desarrollada la aplicación gráfica con Tkinter, con funcionalidades de creación y restauración de backups.

Paquetes de instalación y desinstalación: Los scripts install.sh y uninstall.sh automatizan completamente el despliegue y la eliminación de toda la infraestructura.

p. Valoració de la metodologia i planificació

La metodología incremental e iterativa adoptada ha demostrado ser muy adecuada para un proyecto de esta envergadura. El hecho de dividir el trabajo en fases progresivas y revisar el estado del proyecto semanalmente ha permitido detectar y corregir errores de forma temprana, evitando bloqueos mayores en etapas avanzadas del desarrollo.

La división de tareas entre los miembros del equipo por áreas de conocimiento ha funcionado correctamente, aunque en algunos momentos fue necesaria una mayor coordinación, especialmente cuando los módulos desarrollados de forma independiente debían integrarse entre sí. El uso de Git para el control de versiones ha facilitado esta integración y ha evitado conflictos en el código.

En cuanto a la planificación temporal, se han producido algunos retrasos puntuales. A pesar de esto, el proyecto se ha completado dentro del plazo establecido gracias a la adaptación continua del plan de trabajo.

q. Visió de futur

El proyecto Kontor Tek Inc. tiene un amplio margen de mejora y continuidad. Entre las líneas de trabajo futuro más relevantes se identifican las siguientes:

- **Integración de un sistema de pagos real:** Actualmente los planes de suscripción se gestionan manualmente. En una versión futura se podría integrar una pasarela de pago como Stripe o PayPal para automatizar el proceso.
- **Panel de administración:** Desarrollar un panel web para que el administrador pueda gestionar usuarios, suscripciones y visualizar los informes de seguridad directamente desde el navegador.
- **Soporte para Windows:** El programa de seguridad actualmente solo funciona en sistemas Linux. Adaptar malware_detector.py para ser compatible con Windows ampliaría significativamente el número de usuarios potenciales.
- **Cifrado de las copias de seguridad:** Añadir cifrado AES a los archivos .back para garantizar la confidencialidad de los datos almacenados.
- **Despliegue en servidor cloud:** Migrar la plataforma web a un servidor en la nube con IP fija y certificado SSL, eliminando la dependencia de la máquina virtual local con DHCP.

Glossari

- **Apache:** Servidor web de código abierto que gestiona las peticiones HTTP entre el cliente y el servidor.
- **ARP Spoofing:** Ataque de red en el que un atacante envía mensajes ARP falsificados para asociar su dirección MAC con la IP de otro dispositivo legítimo.
- **Backup (.back):** Copia de seguridad de los archivos del sistema almacenada en formato comprimido tar.gz con extensión personalizada `.back`.
- **Blacklist:** Lista negra de direcciones IP que han sido bloqueadas por el sistema por comportamiento sospechoso.
- **Bot de Telegram:** Programa automatizado que interactúa con los usuarios a través de la plataforma de mensajería Telegram mediante comandos.
- **CORS (Cross-Origin Resource Sharing):** Mecanismo de seguridad de los navegadores que controla las peticiones HTTP entre dominios diferentes.
- **DDoS (Distributed Denial of Service):** Ataque que busca saturar un servidor o red enviando un volumen masivo de tráfico desde múltiples orígenes.
- **DHCP:** Protocolo que asigna direcciones IP de forma dinámica a los dispositivos de una red.
- **Endpoint:** Punto de acceso de una API, representado por una URL que acepta peticiones HTTP y devuelve una respuesta.
- **fetch:** Función de JavaScript que permite realizar peticiones HTTP asíncronas al servidor desde el navegador.
- **GNOME:** Entorno de escritorio gráfico utilizado en sistemas Linux como Ubuntu.
- **MySQL:** Sistema de gestión de bases de datos relacionales de código abierto.
- **nftables:** Framework del kernel de Linux para el filtrado de paquetes de red y el control del tráfico.
- **PDO (PHP Data Objects):** Extensión de PHP que proporciona una interfaz de acceso a bases de datos de forma segura mediante prepared statements.
- **PHP:** Lenguaje de programación de servidor utilizado para generar páginas web dinámicas.
- **Python:** Lenguaje de programación de alto nivel utilizado para el desarrollo del programa de seguridad y el sistema de backup.
- **Scapy:** Biblioteca de Python para la manipulación y análisis de paquetes de red.
- **SYN Scan:** Técnica de escaneo de puertos que envía paquetes SYN para detectar puertos abiertos en un sistema objetivo.
- **systemd:** Sistema de inicialización y gestión de servicios en Linux que controla los procesos en segundo plano.
- **Tkinter:** Biblioteca estándar de Python para el desarrollo de interfaces gráficas de escritorio.
- **Whitelist:** Lista blanca de direcciones IP autorizadas que el sistema no bloqueará bajo ninguna circunstancia.

Bibliografia

[1] Apache Software Foundation. **Apache HTTP Server Documentation**.
https://httpd.apache.org/docs/

[2] PHP Group. **PHP Manual**.

https://www.php.net/manual/es/

[3] Oracle Corporation. **MySQL 8.0 Reference Manual**.

https://dev.mysql.com/doc/refman/8.0/en/

[4] Python Software Foundation. **Python 3 Documentation**.

https://docs.python.org/3/

[5] Scapy Project. **Scapy Documentation.**

https://scapy.readthedocs.io/

[6] python-telegram-bot Contributors. **python-telegram-bot Documentation**.
https://docs.python-telegram-bot.org/

[7] Telegram. **Telegram Bot API**.

https://core.telegram.org/bots/api

[8] Netfilter Project. **nftables Documentation**.

https://wiki.nftables.org/

[9] Mozilla Developer Network. **MDN Web Docs — JavaScript fetch API**.
https://developer.mozilla.org/es/docs/Web/API/Fetch_API

Anexo

Manual de instalación: Guía completa paso a paso para instalar y configurar el programa de seguridad, incluyendo la obtención del Token de Telegram, el ChatID, la ejecución del script install.sh y la descripción de todos los comandos disponibles tanto en terminal como en el bot de Telegram. Disponible en la página web del proyecto en la sección *Manual*.

Diagramas

Diagrama Programa de Seguridad

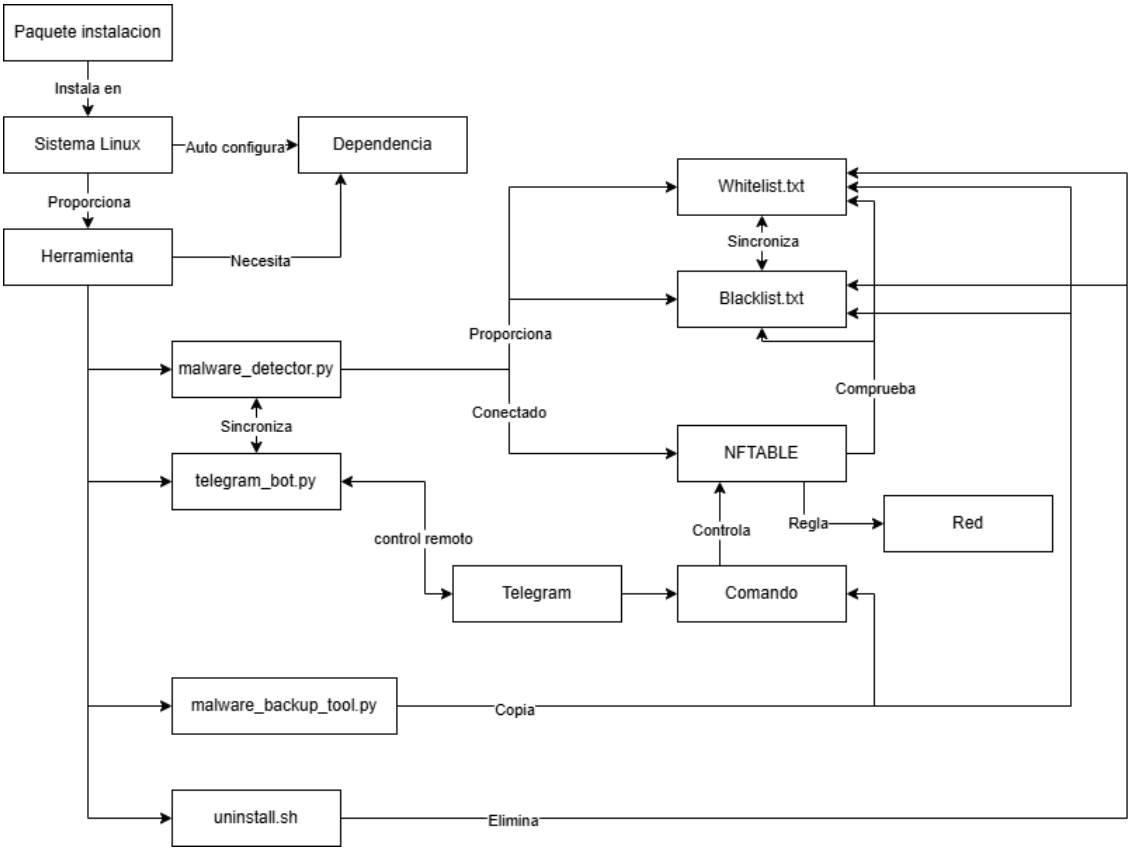


Diagrama Web