Índice

1. Abstract

Mantener nuestra información en un lugar libre de ataques externos es una tarea difícil de realizar. Es por ello que existen numerosas técnicas de prevención de ataques informáticos. En este proyecto hemos desarrollado uno de ellos.

2. Introducción

Un sistema de detección de intrusos para host *(HIDS)* es capaz de monitorear y analizar los archivos internos de un sistema informático. Hemos desarrollado un *HIDS* capaz de comprobar la integridad de todo un sistema de directorios y restaurarlo en el caso de detectar alguna modificación.
Para ello, hemos usado el lenguaje de programación *Python*. *Hashlib* es un módulo de la librería estándar que nos facilita el cifrado mediante el algoritmo *SHA*, entre otros. Éste, aplicado a un fichero, devuelve una cadena alfanumérica de tamaño n, y cualquier modificación generará una nueva cadena totalmente distinta.

3. Objetivo

Los objetivos principales son los siguientes:

- Libre elección del tipo de algoritmo *SHA*, en todas sus variantes.
- Sistema automatizado. La verificación de integridad se realizará cada día.
- Posibilidad de poder restaurar el sistema en caso de añadir, modificar y/o eliminar algún fichero.
- Registro de las incidencias en un archivo *.log*, incluyendo la ruta del fichero y fecha de detección.
- Creación de un *script* de ataque automático para testear nuestro sistema.
- Creación de una función *plot* para mostrar la tasa de archivos modificados (eje y) y la fecha en que se ha detectado el ataque (eje x).

4. Desarrollo

Hemos desarrollado los siguientes *scripts*:

- *funciones-globales.py*: Aquí podemos encontrar funciones básicas.
  Destacaremos la creación de *hashes*, y su búsqueda en el archivo de configuración para su posterior ejecución.
- *attack.py*: Elimina, añade y modifica archivos de manera aleatoria. Realiza la comprobación del estado de los ficheros una vez se ha lanzado el ataque y guarda toda esta información en un .log. Este *.log* nos servirá para crear gráficas.

- *crear-backup.py*: Creación del backup de seguridad. Guardamos las cadenas alfanuméricas generadas por la función *SHA* en un diccionario dentro de un fichero *JSON* y copiamos todo el sistema a un directorio */backup*. También incluye la escritura de todos los *SHA* de todos los ficheros en el archivo de configuración.
- *borrar-backup.py*: Eliminación del backup de seguridad. Es necesario borrar el backup y volver a crearlo si se desea cambiar el algoritmo de cifrado.
- *comprueba-ficheros.py*: Compara todos los *SHA* del archivo *JSON* generado en el script anterior con los actuales. Si algún archivo es modificado, se escribirá en un *.log* con la fecha de detección.
- *comprueba-restaura-ficheros.py*: Además de hacer lo descrito en el *script* anterior, también comprueba si un archivo ha sido eliminado o añadido sin permiso. De forma automática, restaura el sistema a la fecha de la creación del *backup*.
- *plot.py*: Genera una gráfica con los datos obtenidos del *.log* generado por la función *attack.py*.

Todos estos *scripts* y directorios de prueba han sido alojados en un servidor externo (sistema *Linux*).
Se adjunta el directorio *primer-entregable* para ejecutar el HIDS en local para sistemas *Windows*. También se adjunta un fichero *.txt* con las instrucciones a seguir para el correcto funcionamiento del sistema, así como las credenciales de identificación para entrar en el servidor externo.

5. Resultados

Después de automatizar los ataques en el servidor mediante *crontab*, se han obtenido los siguientes resultados (ver 1).
Destacar que los ataques han sido completamente aleatorios y no se han incluido en los posteriores ataques los ficheros ya modificados anteriormente. Por cada ataque se modifican de 6 a 15 ficheros.
El gráfico es acumulativo. Se automatizó el ataque con *crontab* para que se realizara todos los días a la misma hora. El día *11 de octubre* el ataque finalizó, habiendo modificado todos los archivos existentes en el directorio.

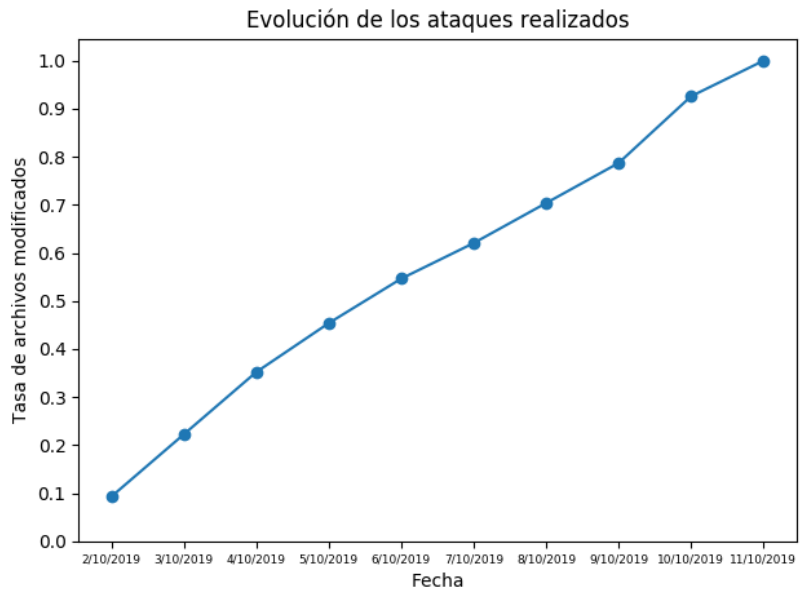

Figura 1: Gráfico de ataques

6. Conclusión

Tras la finalización del sistema, hemos aprendido a desarrollar un *HIDS* básico, que cumple las exigencias propuestas. Sería ideal que una vez ejecutados todos los *backups*, se protegiera tanto esta carpeta como el archivo *hash-backup.json* con contraseña. De esta manera, solo tendría acceso un grupo privilegiado. Destacar que es la primera vez que nos enfrentamos a un problema real de seguridad informática. El sistema podría mejorarse introduciendo algún *software* capaz de analizar el tráfico entrante y saliente y actuar en consecuencia a ello.

7. Aclaraciones

Todo el sistema se ha realizado desde cero. Se ha hecho uso de los siguientes módulos:

- *Matplotlib*, para generar gráficos.
- *Hashlib*, para la creación del *hash*.
- *Os, time, random* y *json* como módulos del sistema.
- *Shutil*, para realizar operaciones con directorios y ficheros.

No hemos hecho uso de ninguna función externa ni se ha compartido código con ningún otro grupo.