File Integrity Monitoring

Transcribed on July 7, 2025 at 3:48 PM by Minutes AI

Speaker 1 (00:08)

Bienvenidos a esta nueva sesión en la cual vamos a realizar un nuevo ejemplo de uso de elementos criptográficos, en este caso orientados a la integridad, para ver utilizar cuándo se modifica ciertos archivos o elementos en un sistema operativo.

Lo que estamos haciendo es una especie de worksheet que va a estar observando ciertos elementos en busca de cambios.

Si nos paramos a pensar cómo funciona un sistema de detección de intrusiones a nivel de host, precisamente el tema de la integridad de los binarios o de los ficheros es algo fundamental, es decir, detectar que hay cambios en ficheros que no deben cambiar nunca o que tienen poca frecuencia de cambio.

Esto es lo que haría un hids, por ejemplo, a la hora de evaluar este tipo de anomalías dentro de un sistema.

Nosotros lo que vamos a hacer con bas es un pequeño ejemplo de ello.

Como he comentado en la demo, en esta demostración lo que vamos a hacer es escribir un script de bash, el cual va a leer un directorio y va a sacar el hash de todos los ficheros que hay en ese directorio.

Después lo que vamos a hacer es implementar ese watcher, el cual va a proporcionarnos un mecanismo para detectar cuando hay un cambio en un fichero y de esa forma poder notificarlo.

Bien, vamos a pasar ahora a la parte de la máquina virtual y vamos a comenzar a escribir nuestro script.

Vamos a llamarlo watcher.

Tenemos el intérprete por defecto que queremos utilizar, que en este caso para que en caso de que no se indique con qué shell queremos ejecutar el script, pues en sistema se buscará la existencia de Ovas para poder ejecutar este script.

Bien, lo que vamos a hacer es como hicimos en otro caso, es utilizar un control de argumentos.

En este caso nuestro script va a funcionar de la siguiente manera, es decir, nuestro script va a necesitar al menos un argumento, el uso sería el nombre del script y luego un parámetro el cual marcará o indicaremos el directorio en el que queremos monitorizar los elementos, es decir, ese directorio analizaremos todos los archivos que hay dentro de ese directorio, sacaremos el hash de ellos y en caso de en caso de que haya un cambio sobre

Como digo, son soluciones que están implementadas a nivel real, con otro tipo de complejidad lógicamente, pero que nos hace ver la idea un poco de estos elementos que están monitorizando los cambios de integridad.

Bien, aquí vamos a indicarle el uso del script, en este caso workset sh y luego el uso que tenemos que mandar, en este caso directory y tendríamos el exit.

Después si nos pasa un argumento tendríamos que valorar luego que sea un directorio, etc.

Aunque eso también con operadores unarios que existen en Bas podemos hacerlo fácilmente.

Vamos a ir directos al grano, vamos a declarar una variable directory y vamos a decir argumento un lo van pasando en un argumento, es una ruta que es un directorio, lo damos por tratado.

Vamos a declarar también hash table, una tabla hash, un diccionario, fijaos aquí ponen los hases, nuestra idea es tener almacenado los hases, el listado de hases en esta tabla y la clave, es decir, la clave será el nombre del fichero y el valor será el hash correspondiente.

De esa forma luego podremos evaluar si hay algún cambio en ese ese directorio sobre los elementos que tenemos en nuestra tabla hash o en nuestro directorio, nuestro diccionario, como queramos llamarlo.

Bien, entonces vamos a recordar que esto es nuestra hash table for file has.

Bien, vamos a sacarlos con esta instrucción, vamos a sacar los directorios que se encuentran, perdón, los ficheros que se encuentran dentro del directorio que nos han pasado como parámetro sobre que queremos generar esos hashes y almacenarlos para tenerlos evaluados, es decir, para controlar la integridad de estos archivos.

Para ello generamos una variable que se llama files, hacemos un LS sobre la ruta del directorio que estamos monitorizando y vamos a obtener la ruta a todos los ficheros.

Bien, ahora vamos a utilizar un bucle for para ir recorriendo por ejemplo este listado de ficheros que hemos obtenido de ese directorio.

Vamos a utilizar en este caso para ir almacenando en la variable de hashes nuestro diccionario, vamos indicándole oye pues file está clave y el valor que va a tomar, en este caso vamos a tardar por el md, los hashes van a ser de md, ejecutamos sobre el propio fichero cut f, recordad que esto ya lo hemos trabajado en otro ejemplo, me quedo con el hash, solamente con el hash, es lo que me interesa.

De esta forma vamos a ir guardando y vamos a ir diciendo que queremos utilizar el watcher de esta forma, el directorio i love it, let's fucking go bro.

Ejemplo, vamos a poner aquí por ejemplo algorithm algún fichero, por ejemplo carpeta, tenemos esta carpeta y tenemos ese fichero, tenemos esta carpeta y tenemos ese fichero.

Ahora quedaría cero esto y lo que nosotros vamos a decirle a nuestro script es bien, el tema es que fijaros, el problema que tenemos aquí es que nos da este error porque no tenemos tenido en cuenta que estamos hablando de que los ficheros están dentro de una carpeta, entonces en este caso lo que vamos a hacer es, lo que tenemos que indicar es acuérdate de poner la ruta al Directory File, de esa forma sí que podremos trabajar con rutas, con rutas relativas.

Bueno, ahora veis que no nos ha fallado, está bien, entonces se ha almacenado, lo único que tenemos que ver cómo hacer una previsualización, cómo ver qué hashes son los que tenemos ahí.

Para ello vamos a recorrer for, vamos aquí, vamos a decirle que recorremos todos los hases, todos los elementos, mejor dicho, de nuestra estructura de datos hases, y aquí vamos a meterle, vamos a coger la clave, aquí estamos cogiendo la clave, estamos viendo el listado de claves, le vamos a decir cada clave que tiene la estructura hashes que nos dé el valor, luego que nos dé el valor, como ya vamos a ver, ahí lo tenemos, fijaos como ya tenemos en la estructura de datos una clave, que es el nombre del fichero que se encuentra dentro de pokdir, y tenemos los ficheros y estamos sacando el Has de sus ficheros y lo estamos metiendo dentro de la estructura de datos, que es la estructura de datos Has, como se puede ver aquí.

Bien, esta previsualización está bien, ya viene la parte del watcher.

Ahora lo que queremos implementar es hagamos que una parte del código, una vez que ya tenemos esto, sería como, bueno, pues tengo mi parte de obtener los ficheros que yo tengo que monitorizar.

Podríamos haberlo hecho de otra forma, podríamos haber dicho Oye, pásame el listado de ficheros con las notas absolutas que quieres monitorizar para que no cambien nunca y ya está.

Pero bueno, no es así, lo que hemos hecho es un directorio, todo lo que tú metas en ese directorio no puede cambiar, luego hacemos una previsualización de los hashes, es decir, lo leo la primera vez, lo meto memoria y a partir de entonces sí que no va a poder cambiar y es la parte del Watcher.

El Watcher lo que va a hacer, lo que vamos a hacer aquí es monitorizar que ningún momento, en ningún momento, en ningún momento esto va a poder cambiar.

Entonces lo que vamos a hacer es, el watcher va a estar para siempre con un buen tú, hasta que hagamos un control c, leyendo ficheros, dormirá 5 s, va a leer los ficheros, va a sacar los hases de todos esos ficheros que hay dentro de pogdir, va a sacar los hases, va a comparar los hases con los que yo ya conocía y si alguno ha cambiado me va a decir este fichero ha cambiado, 5 s después va a dormir y luego 5 s después, mejor dicho, va a volver a arrancar y va a hacer el proceso otra vez.

Entonces aquí tenemos un bucle para volver a recorrer todos los ficheros.

Sacaremos hashes parciales, sacamos el hash M sum file, más que Dollar file, el Dollar Directory, utilizamos CAT para quedarnos con el primer campo.

Ahora lo que vamos a hacer es una comprobación de, bueno, si es distinto de dólar valor de hashes, la clave, si esto es distinto, esto es distinto, generamos alerta, generamos la alerta de Oye, cuidado que ha habido algo que se ha modificado o que no es igual.

Bien, el fichero dólar file tiene un hash antiguo, que es lo que estamos diciendo aquí, es oye, el fichero tal que estamos monitorizando antes tenía este hash y ahora tiene este hash, eso significa que ha sido modificado.

Y luego ya simplemente cuando hemos hecho esta evaluación de todo, si no hay ninguna alerta, lo que vamos a hacer es un sleeping por five seconds 5 s y continuaremos.

Esto si no entra nunca aquí, significa que los ficheros nunca han cambiado, pero en el momento que cambien, pues lo vamos a alertar.

Esa alerta luego lo podemos cambiar por automatizar algo, cualquier cosa, otro tipo de mensajería, lo que queramos, pero aquí por pantalla nos va a decir oye, aquí ha habido un cambio y está generando alerta.

Podríamos utilizar Syslog para generar un evento a nivel sistema donde quede registrado.

Al final se puede hacer cosas bastante potentes, pero fijaros, la idea y la base está de nuevo en los algoritmos, en este caso de hashing, en este caso de ficheros, y está pensado en cómo también parte de los hids que funcionan en una máquina a nivel de host, cómo funcionan realmente es un poco lo que estamos mostrando aquí.

Así que vamos a vamos a probar de nuevo levantarlo.

Fijaros, va durmiendo esta parte de previsualización, estos son los valores de los hashes y va durmiendo hasta que nosotros en algún momento entremos aquí y modifiquemos, por ejemplo este.

Guardamos y entonces fijaros cómo nos ha dicho esto y ahí está.

Y nos va a estar alertando constantemente.

Si yo vuelvo a dejar, yo vuelvo a dejar el fichero como está, lógicamente esta alerta va a dejar de existir porque ya de nuevo, cuidado, cuando hay un cambio de has yo no actualizo los hases.

Yo tengo mis hashes marcados a fuego, jargo deados, por así decirlo, marcados a fuego.

Estos son los hases de estos ficheros, estos ficheros no pueden cambiar.

¿Cómo funciona un hids?

Realmente el hids, hay ficheros que están clasificados con esto es raro que cambie, esto no puede cambiar, esto es como que cambie.

Entonces claro, a nivel de alerta no es lo mismo.

Nosotros hemos hecho más básico, pero la idea se puede entender, la idea se puede entender y es bastante potente.

Luego claro, si nosotros vamos a pararlo, vamos a meternos de nuevo aquí en Pogbir, vamos a meter por ejemplo el p y vamos a poner otra cosa, por ejemplo, tenemos ya 13 ficheros, tres hashes diferentes y ahí vamos, vamos dejándolo.

Y bueno, es una buena praxis, es una buena praxis para decir oye, para detectar anomalías cuando esto no tiene que haber cambiado o esto no tiene que tal, se puede hacer a nivel de atributos, estos ficheros no se tocan, no se modifican a mis modificaciones también se puede mirar.

Pero esto es algo bastante interesante porque fijaos qué nos aporta los algoritmos de hashing en este caso.

Bueno, sencillo, como puede verse.

Vamos a hacer el pequeño resumen a modo de conclusiones, vamos a visualizar de nuevo el código y vamos a entenderlo.

Control de los argumentos que nos pasan.

Nosotros como desarrolladores decimos mi skip va a funcionar con solamente un un argumento, que es el directorio donde queremos hacer variaciones.

Oye, pásame un listado y que no hubiera este control, sino que pásame un listado.

Si pasas un argumento solamente, pues es un directorio, tengo que entender que es un directorio y puedo comprobar que es un directorio con operadores unarios que hay para esas comprobaciones.

O si me pasas un listado de dos, tres, cuatro, x, son ficheros, pásame a rutas completas, completas, si existe pinchero, etc.

Bien, luego la estructura clave es la estructura de tabla hash.

La tabla hash, el diccionario python es la clave, porque nosotros lo que vamos a hacer, voy a leer todos los ficheros que hay en ese directorio, han pasado por el funcionamiento que tiene script, podríamos saber si podríamos decir, oye, no, en vez de un directorio te vas a un listado de archivos, también lo has podido recorrer, porque al final con bas podemos recorrer como si fuera una lista los argumentos que me pasan al script, por lo cual ahí tendríamos también esa posibilidad.

Bueno, deciros también que luego tenemos la parte de ficheros, donde, bueno, aquí está, perdón, ficheros, la parte de preview simplemente es, oye, la tabla hash tiene estos elementos las claves son los nombres de los ficheros a los que estamos almacenando el hash, que es el valor, en este caso asociado a la clave de esa estructura.

Y luego con el watcher, un funcionamiento que como veis es sencillo, lo que hacemos es nos ponemos en modo bucle infinito, nos ponemos a leer, o mejor dicho, sacar el hash de todos los ficheros que estamos trabajando y comparando ese hash con los que tenemos almacenados en la estructura de datos anterior.

De modo que en el caso de que alguno de esos hashes haya cambiado, nosotros alertamos la alerta.

¿Cómo podemos hacer?

Como he comentado antes, vía email, vía mensajería, vía syslog, lo que queráis.

La alerta se genera, nosotros lo imprimimos por pantalla y ya está.

Y luego 5 S.

5 s para no estar infinitamente, para no estar constantemente, dejo un tiempo, cinco, que pueden ser 20, que pueden ser 30, que puede ser 1 min.

No hace falta que sea cada 5 s, eso ya lo decidiríamos.

Pero lo importante de todo esto es, vale, tenemos un ejemplo práctico y que se entiende que tiene sentido y que además es algo muy básico en HIDs, pero aquí lo interesante es, usamos algoritmos de hashing para poder detectar cuando un elemento cambia en un sistema y poder alertar de que hay una anomalía.

Esto puede ser que haya un intruso, puede ser que haya un atacante que está dentro del sistema y que está modificando ficheros o que está haciendo algo en los ficheros y yo me entero por eso al final es lo que está haciendo un hids o una de las funcionalidades principales de un hids.

Aquí tenemos un mini ejemplo de esto.

Bien, pues dicho esto, vamos a volver a las slides y como conclusiones, de nuevo hemos hecho un ejemplo con algoritmos de hashing y hemos visto un caso diferente a otros que hemos ido comentando, mostrando, donde podemos implementar funcionalidades básicas a nivel de anomalía, porque hay modificaciones de ciertos sistemas y podemos hacerlo con BAS y con los elementos de hashing que hemos ido revisando.

Así que nada, con esto finalizamos la sesión.

Espero que se haya entendido, que os parezca interesante el ejemplo y nos vemos en la siguiente sesión.