**Previniendo ataques**



La prevención (en cualquier ámbito de la vida) es muy importante, ya que nos permite estar preparados frente a determinados eventos que puedan ocurrir y nos generen problemas serios.

Cuando pensamos en proteger un servidor, no solamente vamos a conocer nuestras debilidades, sino que ademas podemos saber como actuar cuando se nos presenten logrando corregirlas rápidamente.

Los puntos a considerar como “débiles” son muchos, abarcando desde permisos de archivos (propietarios, ejecución, etc), servicios en ejecución que no son necesarios, configuraciones por defecto consideradas inseguras y muchísimas cosas más, con lo cual realizar un control o chequeo punto por punto puede llegar a ser una tarea tediosa, corriendo además el riesgo de equivocarnos en algún de ellos, quedando sin hacerse y por ende terminamos dejando agujeros de seguridad.

Es por ello que existen muchas herramientas para realizar “hardening” (o aseguramiento) del sistema operativo, automatizando gran cantidad de estos controles, permitiendo tener muy buenos tiempos en su ejecución y además contar con una lista amplia de pasos a realizar.

Analizando la cantidad de herramientas disponibles y sus funcionalidades, nos encontramos con dos problemas que son los siguientes:

1.- Herramientas discontinuadas

2.- Herramientas en actividad pero estas cuentan con un repositorio de chequeos propio. Esto significa que para poder agregar un nuevo control hay que esperar que el desarrollador libere el mismo (en muchos casos ocurre que determinada funcionalidad o chequeo que estamos necesitando hacer no es liberado), con lo cual existe una limitante importante.

Por todos esto mencionado anteriormente, creamos una herramienta de muy simple uso, la cual tiene la capacidad de ser extendida a demanda, es decir cada uno le puede cargar los controles necesarios mediante cualquier lenguaje de scripting o programa propio, sin esperar por el desarrollador y hacerla de esta manera crecer a demanda.

Para ello, implementamos tres cosas:

.- Una base de datos sqlite con una tabla conteniendo los chequeos a realizar

.- Un pequeño motor en python que recorre esa tabla y realiza dicho chequeo

.- Scripts en bash que son los que se encargan realmente de los chequeos. Al momento de liberada la herramienta, se cuentan con 46 scripts. Es importante mencionar que los scripts que se encargan de los controles pueden ser cualquier programa (sea scripting o compilado), lo importante es el retorno “0” OK y “1” ERROR

El funcionamiento es muy simple, a la hora de definir un nuevo control, simplemente debemos crear el script que realizara el mismo y darlo de alta en la tabla “Tasks”, la cual es leída por el programa principal “tfm11.py” y se encarga de su ejecución.

Los resultados siempre deben ser en caso exitoso “0” y en caso de error “1”. Si se desea mayor detalle, se puede pasar al script el flag “D” (Detalle) y en lugar de retornar un “1” retornará un mensaje detallado del problema encontrado.

Entendemos que de esta manera, es posible expandir la herramienta a controles de cualquier tipo (chequeos de permisos, revisión de bases de datos, transacciones, verificación de servicios en ejecución, control de información en logs) y una cantidad enorme de controles específicos del ambiente a utilizar.

La herramienta y su documentación, se encuentran en el siguiente repositorio de github ( <https://github.com/alejandroguardian/TFM11>), el mismo de público y se encuentra disponible para su uso.

Esperamos que este trabajo aporte un granito de arena en este apasionante mundo, colaborando en hacer mas seguros los servidores.

Estamos en contacto !