Verificación de integridad de procesos a nivel kernel.

Gómez R., Saucedo E.
ITESM-CEM
{rogomez, esaucedo}@campus.cem.itesm.mx

Actualmente la Seguridad Computacional es un elemento importante en cualquier organización. Las propiedades de confidencialidad, integridad y disponibilidad de los datos son elementos presentes en casi todas las organizaciones y las amenazas a estas propiedades son cada día mayores y más sofisticadas. Los ataques son cada día mas sofisticados y la responsabilidad de llevar a cabo acciones ante alguna contingencia recae en el administrador; sin embargo esta tarea además de ser tediosa es bastante insegura por la responsabilidad humana que esta conlleva.

Casi todos los sistemas actuales se basan en una interacción GUI entre los administradores y los usuarios. En muchos sistemas, los administradores tienen la idea errónea que una vez que un sistema ha sido configurado de una forma correcta este permanecerá en dicho estado de manera permanente [4]. Han surgido muchas ideas acerca de la automatización de sistemas; la mayoría de ellas basadas en el uso de scripts; sin embargo, muy pocas ponen especial atención en la degradación operacional que estos pueden causar al sistema. Idealmente, la intervención del administrador debería ser eliminada en lo posible.

En nuestro enfoque proponemos que, en caso de un ataque, el responsable de llevar a cabo las acciones necesarias sea el sistema operativo y la degradación del sistema sea la mínima. Bajo esta propuesta estamos utilizando Linux como sistema operativo debido a la ventaja que proporciona al tener los fuentes del sistema operativo de forma libre y que es posible modificarlo sin cargo alguno. Aunado a lo anterior consideramos una de las propiedades de un sistema seguro: integridad; un sistema tiene la propiedad de integridad si los datos manipulados no se alteran o destruyen por usuarios, entidades o procesos no autorizados.

Como es sabido, la criptografía es utilizada principalmente para mantener la propiedad de confidencialidad de los datos pero también puede utilizarse para asegurar la integridad de los datos debido a que la modificación de un mensaje que ha sido cifrado es fácilmente detectable. Usando criptografía es posible autenticar un mensaje ya que si dos entidades conocen una llave secreta cuando una de estas entidades recibe el mensaje cifrado con dicha llave esto implica que el mensaje recibido fue enviado por un origen conocido. Finalmente, la criptografía otorga también control de acceso a solo aquellas entidades autorizadas que conocen la llave secreta. Con estas bases se puede utilizar la criptografía para verificar la integridad de los procesos que se ejecutan en un sistema.

Nuestro trabajo es parte de un proyecto de protección de redes a través de agentes móviles. Cuando se habla de seguridad de agentes móviles se debe considerar la protección de hosts de procesos maliciosos, la protección de la comunicación entre procesos y la protección de hosts maliciosos. Los primeros dos puntos pueden resolverse

con técnicas convencionales de seguridad pero el último punto es el que detallamos a continuación.

Nuestro trabajo se centra en verificar la integridad de los procesos que protegen un sistema computacional o una red. Consideramos un proceso denominado **proceso de defensa** este, es el proceso encargado de verificar la integridad de las aplicaciones de seguridad y su integridad propia. Una vez que esta verificación se lleva a cabo el **proceso de defensa** se envía a otra máquina de la red. En la siguiente máquina el **proceso de defensa** llevará a cabo la misma verificación que en el sistema anterior; es decir, la verificación de integridad de las aplicaciones de seguridad y la verificación del **proceso de defensa** propiamente. El llevar a cabo la verificación del mismo proceso permite que el sistema sepa si el **proceso de defensa** está llevando a cabo las operaciones para las que fue diseñado. Lo que significa que no haya sido modificado en forma maliciosa.

Nuestro primer paso fue crear un campo nuevo en algún lugar en los fuentes de Linux, lo que permite llevar a cabo la verificación de integridad a nivel kernel. Todos los sistemas operativos tienen información acerca de cada proceso que se está ejecutando en el sistema. Esta información es almacenada en una tabla llamada tabla de procesos. Las entradas contenidas en esta tabla tienen información acerca del estado del proceso, el estado de la memoria y el estado de la memoria y el estado de los recursos para cada proceso. En Linux, la información almacenada por proceso es definida por la estructura task_struct.

En base a lo anterior, decidimos añadir el nuevo campo en la estructura task_struct. Este campo actúa como una huella digital del proceso, particularmente usamos el algoritmo MD5 en los campos que no deben cambiar durante la vida del proceso debido a que cualquier cambio causa un cambio en la firma digital, característica que aprovechamos para garantizar la integridad de los procesos. Por tanto, añadimos un campo de 16 bytes donde almacenamos una huella MD5 y se ha verificado durante el proceso de verificación. Después de esto es necesario recompilar el kernel para que los cambios se lleven a cabo.

Durante el proceso de verificación se tuvo que tomar en cuenta que era necesario acceder al cuarto Mb de la memoria, y que este solo puede ser accedido por el sistema. Decidimos crear una nueva llamada al sistema que nos permita acceder a la información que necesitamos. Esta llamada al sistema ejecuta el algoritmo MD5 sobre los campos que seleccionamos después de examinar todos los campos que componen la estructura task_struct. Después que la llamada al sistema lleva a cabo la verificación de integridad del proceso de defensa y ejecuta el proceso que lleva a cabo el cambio de máquina de acuerdo al algoritmo que proponemos.

Se han realizado pruebas a nivel local con cierto tipo de procesos obteniendo resultados satisfactorios. El siguiente paso es ampliar las pruebas a un nivel de red.