Protección de Componentes en una Plataforma para la Prevención de Ataques Coordinados

Sergio Castillo, Joaquín García, Guillermo Navarro y Joan Borrell

Dept. d'Enginyeria de la Informació i de les Comunicacions

Escola Tècnica Superior d'Enginyeries (ETSE)

Universitat Autònoma de Barcelona, 08193, Bellaterra

{SCastillo, JGarcia, GNavarro, JBorrell}@ccd.uab.es

Resumen

Se presenta en este artículo un conjunto de mecanismos de control de acceso para la protección de componentes de una plataforma preventiva en la que actualmente estamos trabajando. Estos mecanismos de control de acceso suponen una barrera que impide a un posible atacante conseguir una posición privilegiada dentro del sistema, con el objetivo de desactivar sus componentes o manipular sus elementos de forma deshonesta.

1. Introducción

Los ataques contra redes informáticas acostumbran a beneficiarse del uso de técnicas coordinadas y distribuidas, ya que ofrece la posibilidad de ejecutar tareas más complejas, como puede ser la realización de ataques de denegación de servicio distribuidos o la exploración de puertos coordinados [6].

Estas técnicas son también de utilidad para dificultar su detección, por lo que a menudo dichos ataques no podrán ser detectados considerando exclusivamente la información aislada de cada uno de los nodos del sistema.

De modo similar, el uso de técnicas distribuidas son también beneficiosas para la implementación de sistemas para la detección y reacción ante ataques contra redes informáticas. La distribución de los componentes de un sistema de detección de intrusos (IDS, *Intrusion Detection System*), por ejemplo, permitirá una mejora en la recogida de información, para su posterior análisis y correlación en distintos nodos de procesamiento.

Actualmente estamos trabajando en el desarrollo de una plataforma para la prevención de ataques coordinados que trata de alcanzar dicha distribución [3]. Nuestra propuesta utiliza un paradigma publicador/subscriptor para la comunicación de los distintos componentes del sistema. Cada uno de los nodos del sistema, llamado *celda de prevención*, consta de un conjunto de sensores, analizadores, gestores y unidades de respuesta.

Los sensores, o analizadores de bajo nivel, se encargan de recoger información sospechosa (como, por ejemplo, establecimiento de conexiones anómalas, cambios repentinos en el tráfico de la red, etc.). Tal información es publicada sobre la infraestructura de comunicaciones de la plataforma (en forma de alertas). Por otro lado, el conjunto de analizadores y gestores estarán subscritos a la plataforma de prevención, a la espera de poder consumir las alertas y realizar un proceso de correlación y selección de contramedidas a ejecutar por parte de las unidades de respuesta.

Los distintos componentes de cada celda de prevención tratarán de detectar si los recursos donde está albergada forman parte activa de un ataque coordinado. En caso de detectar tal situación, las celdas de prevención actuarán de la forma apropiada sobre cada uno de sus recursos asociados para, finalmente, eludir su participación en el ataque detectado.

Si un atacante descubre la presencia de una celda de prevención, intentará desactivar-

la previamente a la ejecución del ataque con el objetivo de evadir la detección y cancelación del mismo. De igual modo, el usuario ilegítimo podría intentar interactuar con los elementos que conforman las celdas de prevención con la finalidad de provocar un mal funcionamiento de la plataforma. Así, una manipulación adecuada permitiría al intruso generar alertas falsas que, enviadas a otras celdas de prevención, podrían causar denegaciones de servicio (a través de las unidades de respuesta) o bien ocultar el ataque real a los gestores de correlación.

La ejecución de las celdas de prevención se realiza en modo privilegiado (administrador), lo que garantiza evitar que usuarios *normales* puedan interactuar con éstas. Esta situación llevaría al atacante a intentar conseguir una posición privilegiada dentro del sistema, no solo con el objetivo de desactivar la celda de prevención o manipular sus elementos de forma deshonesta, sino también porque acciones involucradas en ataques coordinados, tales como *IP Spoofing* o exploración de puertos silenciosa, requieren de privilegio dentro del sistema. Así pues, es necesario la inclusión de mecanismos que mitiguen o eliminen cualquier tipo de acción que atente contra los elementos de la plataforma y su correcto funcionamiento.

En este artículo presentamos el diseño de un conjunto de mecanismos de control de acceso para la protección de los distintos componentes de cada celda de prevención. Estos mecanismos tratan de reforzar las deficiencias de seguridad presentes en nuestra plataforma de prevención, tratando de impedir a un posible atacante conseguir una posición privilegiada dentro del sistema, con el objetivo de desactivar sus componentes o manipular sus elementos de forma deshonesta, una vez tenga bajo su poder el nodo asociado a dicha celda de prevención. Se presenta también en este artículo el desarrollo de una primera versión de los mecanismos de control propuestos, integrados en el núcleo del sistema operativo utilizado para el desarrollo de nuestra plataforma.

El resto del documento está organizado de la siguiente manera. Nuestra plataforma, y los componentes de cada una de las entidades que la conforman, son presentados en la sección 2. Los mecanismos de control para la protección de componentes son descritos a lo largo de la sección 3, y una breve descripción del desarrollo actual se encuentra en la sección 4. Las conclusiones y el trabajo futuro se podrán encontrar en la última sección.

2. Sistema de celdas de prevención

En esta sección presentamos nuestra plataforma para la prevención de ataques coordinados. Por medio de la utilización de un conjunto de entidades cooperativas, la plataforma pretende evitar la participación de los recursos donde dichas entidades están siendo albergadas para la realización de las diferentes etapas.

El diseño de nuestra plataforma tiene tres objetivos principales. El primero de ellos es obtener una arquitectura modular compuesta por un conjunto de entidades que cooperen. Estas entidades colaborarán entre sí para detectar si el equipo donde están instaladas participa de forma activa en la elaboración de un ataque coordinado contra la red donde están conectados o contra una red de terceras partes. Una vez el ataque, o una de sus acciones, haya sido detectado, estas entidades deberán ser capaces de prevenir la utilización de los recursos asociados para evitar su participación en el ataque detectado.

El segundo objetivo es conseguir una relación completa e independiente entre los diferentes componentes que conforman cada una de estas entidades cooperativas. De esta forma, será posible distribuir tales componentes de acuerdo con las necesidades de cada recurso que queramos desarmar, de una forma escalable y eficiente.

Por último, el tercer objetivo es obtener un sistema tolerante a ataques contra el propio sistema. Así, aún en caso de compromiso por parte del nodo que alberga a cada celda de prevención, ésta debe mantener, de forma aceptable, sus mecanismos de detección y reacción.

2.1. Componentes de una celda

Los componentes internos a cada celda de prevención son integrados en nuestra plata-

forma de acuerdo con los elementos básicos de cualquier sistema de detección y reacción (sensores, analizadores, gestores y unidades de respuesta). Los mensajes intercambiados entre estos componentes serán básicamente tres: *eventos* (entre sensores y analizadores), *alertas* (entre analizadores y gestores), y *acciones* (entre gestores y unidades de respuesta). Estos componentes, y los distintos mensajes intercambiados entre ellos, son descritos a continuación.

En primer lugar tenemos los sensores, o recolectores de información. Estos componentes obtienen información sospechosa, tanto a nivel de red como a nivel del propio equipo donde se encuentran instalados, para publicarla en un canal específico (donde algunos analizadores asociados estarán a la escucha). Nuestra plataforma utiliza sensores basados en red (en busca de tráfico ilegal o peligroso como, por ejemplo, desbordamientos de buffer, datagramas con suplantación de IP, inundación de tráfico, etc) y sensores basados en equipo (en busca de acciones sospechosas o peligrosas originadas en el propio equipo y que pueden suponer un riesgo como, por ejemplo, utilización abusiva de la CPU, ejecución de comandos potencialmente hostiles, etc.).

Los eventos publicados por los sensores del sistema son recibidos por una serie de analizadores, encargados de realizar un proceso de análisis y de correlación de bajo nivel. Así, estos componentes consumen eventos y producen alertas locales en el interior de cada celda de prevención. Estas alertas son publicadas en el canal correspondiente (en este caso, en el canal de alertas locales). La plataforma hace uso tanto de analizadores con detección basada en usos indebidos como de analizadores con detección basada en anomalías.

Un gestor de correlación, que está a la escucha de alertas locales y externas a través de los canales correspondientes, es el responsable de contrastar dicha información contra sus escenarios de ataque asociados. Este componente realiza un proceso de correlación a partir de las alertas recibidas. Está, por tanto, involucrado en la parte del proceso de correlación relativo a la celda de prevención donde se encuentra albergado. Puesto que la correlación de un ataque coordinado es un conjunto de alertas internas y externas reportadas por elementos internos y externos del sistema, todas estas alertas son analizadas en un proceso de alto nivel ejecutado por el gestor de correlación. Este elemento es también el responsable de publicar alertas de correlación y alertas de contramedida.

La comunicación con el resto de celdas de prevención es responsabilidad de un gestor de cooperación, que está a la escucha de alertas cooperativas provenientes del exterior de la celda de prevención donde se encuentra instalado. Estas alertas son publicadas como alertas externas en su interior. A la vez, consume las alertas de correlación producidas en el interior de la celda de prevención donde se encuentra, y las publica en el exterior en forma de alertas cooperativas.

La parte reactiva del sistema se basa en un conjunto de gestores de contramedidas, encargados de recibir alertas de contramedida publicadas por el gestor de correlación de la celda de prevención que les alberga. Estos gestores son los responsables de consumir contramedidas y transformarlas en las acciones correspondientes. Una vez la alerta de contramedida es transformada, las acciones asociadas son publicadas en el canal asociado a las unidades de respuesta de la celda de prevención.

Finalmente, las unidades de respuesta toman las acciones producidas por los gestores de contramedidas para llevarlas a cabo. Cada acción es generada para prevenir uno de los distintos pasos del ataque coordinado que ha sido detectado, y es ejecutada contra el recurso donde la celda de prevención se encuentra alojada. Al igual que los sensores, la plataforma hace uso de unidades de respuesta basadas en red (para, por ejemplo, bloquear conexiones, cerrar puertos TCP/UDP, etc.) y unidades de respuesta basadas en equipo (para, por ejemplo, finalizar la ejecución de procesos, realizar un cambio de permisos, bloquear cuentas de usuario, etc).

3. Mecanismos de protección

La mayor parte de la investigación actual en el ámbito de los sistemas de prevención contra ataques informáticos focalizan sus esfuerzos en mejorar sus mecanismos de detección y reacción, sin considerar elementos de seguridad que permitan su protección. En esta linea, cuando un atacante compromete la seguridad de un sistema de detección de intrusos (IDS, *Intrusion Detection System*), por ejemplo, tratará de desactivar los mecanismos de detección, así como eliminar los ficheros de registro que delaten su entrada en el sistema. En este contexto, podemos considerar el concepto de protección basándonos en la definición introducida por Butler Lampson en 1974, quien describió *protección* como la habilidad de controlar el acceso de un programa o un usuario a otros elementos de un sistema [5].

Tal y como se ha expuesto en la sección anterior, las entidades de nuestra plataforma cooperan entre sí para detectar cuando los recursos donde están albergadas forman parte activa de un ataque coordinado. Al igual que en un IDS tradicional, una manipulación adecuada de los procesos asociados a cada celda de prevención permitiría a un atacante evadir la detección de su presencia. De esta forma, el intruso sería libre de ocultar la parte del ataque local a la celda de prevención donde se encuentra, o incluso podría generar alertas falsas que, enviadas a otras celdas de prevención, provocaran un mal funcionamiento de la plataforma. Esta situación nos lleva a la necesidad de incorporar mecanismos de protección sobre los distintos componentes de cada celda de prevención, mitigando o eliminando cualquier tipo de acción que atente contra la plataforma y su correcto funcionamiento.

Las características inherentes al diseño de componentes de nuestra plataforma nos llevan rápidamente a dos posibles mecanismos de protección: *auto-protección* por parte de los propios elementos de cada celda de prevención, o protección a nivel del *núcleo del sistema operativo* que los alberga.

En el primer caso, cada componente es responsable de su propia protección utilizando mecanismos como podrían ser la ocultación de procesos, los sistemas basados en agentes móviles, o técnicas criptográficas asociadas a los registros del sistema. Actualmente, la mayor parte de las propuestas basadas en esta solución, tales como [1, 9, 10], se muestran ineficientes contra ataques en los que el intruso, en posesión de una posición de privilegio, interactúa sin restricciones con los componentes a través del propio sistema operativo. De esta manera, la cancelación de los procesos asociados al sistema de detección, o el borrado de registros, demuestran la insuficiencia de estas metodologías. El problema de estas propuestas reside en dos hechos. En primer lugar, la existencia de usuarios privilegiados (administrador) en la mayoría de sistemas operativos actuales, los cuales pueden interactuar con el sistema con plena libertad. Y, en segundo lugar, en la delegación de parte de la protección al control de acceso proporcionado por el sistema operativo, sin considerar que un atacante podría, a partir de una deficiencia de seguridad, adquirir los permisos de un usuario privilegiado.

En el segundo caso, el núcleo del sistema operativo es el responsable de proporcionar los mecanismos de protección adecuados, separándolos del propio sistema de detección y/o reacción. La protección se traduce en la incorporación de un sistema de control de acceso a las llamadas del núcleo del sistema operativo. De esta manera, se puede autorizar o denegar una llamada en función de diversos criterios, tales como el identificador del proceso que la realiza, parámetros de la llamada, etc. El control de acceso del núcleo permite, por tanto, eliminar el concepto de confianza asociado a usuarios privilegiados, relegando la autorización de ejecución de una llamada al control de acceso interno. Adicionalmente, y a diferencia de los métodos de *auto-protección* anteriores, proporciona un método unificado, evitando la implementación de diversas técnicas específicas para cada componente.

Para proteger los componentes de nuestra plataforma proponemos el uso del control de acceso integrado en el núcleo del sistema operativo. Así, aunque un atacante escale privile-

gios dentro de un sistema y consiga adquirir los permisos de un administrador, no será capaz de lanzar acciones que atenten contra la celda de prevención, ya que cualquier llamada ilegítima hacia los componentes será interceptada y cancelada por el control de acceso del núcleo.

A su vez, esta metodología nos permite un segundo nivel de protección. El mecanismo incorporado en el núcleo y la modularidad basada en componentes permite aplicar el principio de compartimentalización [11]. Este principio de seguridad intenta minimizar el daño que puede ser ocasionado a un sistema, segmentándolo en varios componentes que pueden ser protegidos independientemente. De esta manera, aunque uno de los elementos del sistema sea comprometido, el resto funcionarían de forma confiable. En nuestro caso particular, diversos componentes de la celda de prevención son ejecutados como procesos. Definiendo los permisos adecuados en el control de acceso basándonos en el identificador de proceso, podemos limitar el entorno de interacción de cada elemento de la celda de prevención con el resto. Con esto conseguimos que, si un intruso se hace con el control de un proceso asociado a un componente a través de un desbordamiento de buffer, por ejemplo, estará limitado a realizar aquellas llamadas definidas para éste.

Dentro de nuestra propuesta, podemos ejemplificar el principio de la compartimentalización de la siguiente manera. En cada celda de prevención, gracias al control de acceso, podemos considerar que los analizadores se ejecutan de forma independiente y aislada respecto al gestor de cooperación. La protección incluida a nivel del núcleo evita así que llamadas peligrosas (como podría ser la responsable de la cancelación de un proceso) sea ejecutada por un proceso asociado a un componente hacia otro elemento. De esta manera, si un usuario ilegítimo adquiere el control de un proceso ligado a un analizador, no podrá cancelar la ejecución del gestor de cooperación.

A pesar de poder ejecutar los elementos de una celda como si se realizase en espacios separados gracias a nuestra propuesta, no siempre es posible considerar una independencia total entre componentes. Es necesario permitir la comunicación entre algunos de éstos para el correcto funcionamiento global de la celda. Sin embargo, esta comunicación puede ser restringida entre componentes de manera que, por ejemplo, analizadores puedan publicar mensajes dirigidos al gestor de correlación a través del canal correspondiente, pero no hacia las unidades de respuesta.

La protección de componentes adoptada para nuestra plataforma plantea algunos problemas desde el punto de vista de la implementación. En primer lugar, es necesario definir de forma específica cuales deben ser las llamadas consideradas como una amenaza cuando son lanzadas hacia un elemento de la celda de prevención. Esto requiere un estudio meticuloso sobre cada una de las llamadas que proporciona el núcleo y de que manera particular pueden ser aprovechadas de forma maliciosa. Y, en segundo lugar, es preciso definir el control de acceso para cada una de estas llamadas. A pesar de la dificultad en relación al estudio sobre las llamadas del núcleo, y la definición de sus respectivos controles de acceso, podemos considerar tres niveles básicos de protección donde podemos catalogar las diversas llamadas:

- Protección de procesos críticos: engloba toda aquella acción que pueda atentar contra la correcta ejecución de los procesos asociados a la celda de prevención, ya sea por la interacción sobre estos a través de señales, o la manipulación de su espacio de memoria. Algunos ejemplos serían: ejecución de nuevos programas ya en memoria, cancelación, manipulación del espacio de direcciones y trazado de procesos, etc.

- Protección de mecanismos de comunicación: recoge todo aquel proceso por el cual un atacante consigue alterar, generar o eliminar cualquier tipo de mensaje intercambiado entre los elementos de la celda de prevención.

- Protección de archivos asociados a los componentes: considera cualquier acción

malintencionada dirigida hacia los archivos que utilizan o manipulan los componentes de la celda de prevención, tales como archivos ejecutables, de configuración o de registro.

4. Implementación

En esta sección presentamos brevemente el desarrollo que se está llevando a cabo actualmente para realizar el control de acceso presentado en la sección anterior. Para tal implementación proponemos el uso del marco de trabajo de módulos seguros de Linux (LSM, *Linux Security Modules*) [12], integrado de serie en las versiones del núcleo 2.6.x.

LSM no proporciona un mecanismo de control de acceso específico, sino que provee un marco de trabajo a través de un conjunto de puntos de intercepción a lo largo del núcleo del sistema operativo. De esta manera, es posible implementar cualquier tipo de control de acceso. Básicamente distingue los siguientes puntos de intercepción: *Task hooks, Program Loading Hooks, Filesystems Hooks* y *Network hooks*.

Estos puntos permitirán establecer protección en los tres niveles que enunciábamos en la sección anterior. Adicionalmente, LSM otorga un conjunto de beneficios a nuestra implementación que afianzan nuestros objetivos. En primer lugar, introduce una carga mínima en el sistema en comparación a un núcleo sin soporte para LSM, no influyendo considerablemente en los tiempos de los procesos de detección y reacción. En segundo lugar, el mecanismo de control de acceso puede ser integrado en el sistema como un módulo, sin necesidad de recompilar el núcleo del sistema operativo, reforzando así la modularidad e independencia entre componentes que enunciábamos inicialmente. Por último, y a diferencia de otras propuestas para Linux [7, 8], cuyas implementaciones requieren la modificación de algunas funcionalidades de la versión original 2.6.x del núcleo, proporciona un alto grado de flexibilidad y portabilidad a nuestra implementación a través de la concepción de un marco genérico para el control de acceso.

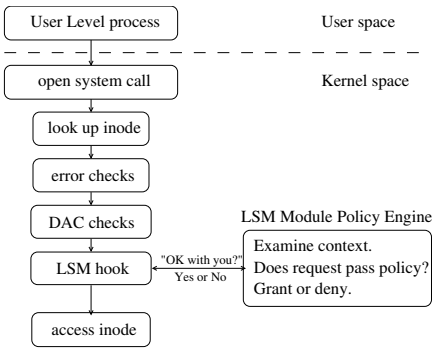

Figura 1: Puntos de intercepción de LSM

La abstracción que proporciona LSM a través de su interfaz, permite que los módulos medien entre los usuarios y los objetos internos del núcleo del sistema operativo. Para esto, previo acceso al objeto interno, el punto de intercepción llama a una función que el módulo debe proveer y que será la responsable de conceder o denegar el acceso. En la figura 1 podemos observar esta idea, en la que un módulo registra una función para realizar un control sobre los *inodos* del sistema de ficheros. A su vez, LSM permite conservar el control de acceso discrecional (DAC, *Dicretionary Access Control*) que Linux proporciona, al interponerse entre dicho DAC y el objeto en si. De esta manera, si un usuario no tiene permisos en relación a un determinado archivo, por ejemplo, el DAC del sistema operativo le denegará el acceso sin llamar a la función que registró el módulo LSM. Esta arquitectura reduce la carga que se introduciría en el sistema en comparación a un control de acceso para la protección de los componentes centralizado en la interfaz de llamadas del sistema operativo. En este último caso, las correspondientes comprobaciones asociadas a la protección siempre se realizarían y, posteriormente se cedería el control al mecanismo DAC del sistema operativo, lo que supondría realizar, en la mayoría de veces, un trabajo innecesario.

Los componentes de la celda de prevención deberán poder realizar operaciones solo permi-

tidas para el administrador del sistema (como, por ejemplo, filtrado de paquetes, cancelación de cualquier proceso, etc). Esto implicará que los procesos de sistema asociados a los componentes se lanzarán como si fuese el administrador del sistema quien realizase tal operación. De lo contrario, si asociáramos un usuario no privilegiado para cada componente, el mecanismo DAC de Linux no permitiría la ejecución de determinadas llamadas. El control de acceso interno al núcleo se basará, por tanto, en el identificador de proceso (PID, *Process Identifier*) que realiza la llamada y que estará asociado a un componente específico. Cada función registrada por un módulo LSM, determinará qué componente realiza la llamada a partir de su PID y, a partir de esto, aplicará el control de acceso basándose también en los parámetros de la llamada. Así, por ejemplo, un componente podrá acceder a sus archivos de configuración, pero no a los de otro.

Un aspecto a tener en cuenta desde el punto de vista de la implementación, es la propia administración de los controles de acceso y la gestión de cada una de las celdas de prevención. Como exponíamos en la sección anterior, los administradores no deben ser capaces de lanzar ninguna llamada del sistema que atente contra la celda de prevención, evitando así que también lo haga un intruso en caso que escale sus privilegios a los del administrador. Este hecho se contrapone con la gestión de la celda ya que, si un administrador no puede interactuar con los componentes, tampoco podrá realizar ningún proceso de gestión y configuración sobre la celda. Para solventar esta problemática, hemos introducido un mecanismo de autenticación temporal basado en el identificador del vendedor de un dispositivo USB, así como su identificador de producto. Así, mientras el dispositivo esté conectado al sistema, el administrador gozará de los privilegios única y exclusivamente necesarios para la manipulación y gestión de la celda. En el momento que el dispositivo sea retirado, las restricciones volverán a su normalidad. En la figura 2 se muestra como una función registrada por un módulo LSM habilita la modificación de un archivo de configuración.

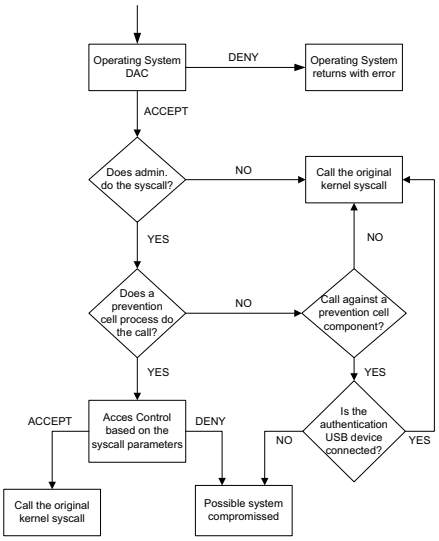

Figura 2: Control de acceso en la modificación de un archivo de configuración

5. Conclusiones

Hemos presentado una arquitectura para la prevención y protección de ataques coordinados que proporciona protección a los componentes que la conforman. Dicha protección se basa en un control de acceso integrado en el núcleo del sistema operativo, reforzando nuestro objetivo de modularidad e independencia entre componentes. Así mismo, esta metodología permite la inclusión de elementos de terceras partes de forma confiable sin realizar grandes cambios en la infraestructura.

Aunque nuestra propuesta basada en el control de acceso a nivel del núcleo del sistema operativo ofrece una protección a los componentes de nuestra plataforma, nada garantiza que una vulnerabilidad en éste no permita a un usuario hacerse con el control de los componentes de una celda de prevención. Por ello, estamos estudiando metodologías, como la tolerancia a intrusiones [2, 4, 13], que nos conduzcan a reforzar la seguridad de la plataforma en general, evitando que una celda comprometida por una deficiencia afecte al funcionamiento global de la arquitectura.

Referencias

[1] S. Bhattacharya and N. Ye. Design of robust, survivable intrusion detection agent. In *1st Asia-Pacific Conference on Intelligent Agent Technology*, Hong Kong, December 1999.

[2] P. Esteves-Verissimo, N. Ferreira Neves, and M. Pupo-Correia. Intrusion-Tolerant Architectures: Concepts and Design. Technical report, Computer Science Department, University of Lisboa, Portugal, April 2003.

[3] J. García, F. Autrel, J. Borrell, S. Castillo, F. Cuppens, and G. Navarro. Decentralized publish-subscribe system to prevent coordinated attacks via alert correlation. In *6th International Conference on Information and Communications Security*, October 2004.

[4] F. Gong, K. Goseva-Popstojanova, K. Vaidyanathan, K. Trivedi, F. Wang, R. Wang, and B. Muthusamy. Characterizing Intrusion Tolerant Systems Using a State Transition Model. In *DARPA Information Survivability Conference and Exposition II (DISCEX-II)*, Anaheim, California, June 2001.

[5] B. W. Lampson. Protection. In *ACM SIGOPS Operating Systems Re-view*, volume 8(1), pages 18–24, January 1974.

[6] D. Moore, G. Voelker, and S. Savage. Inferring Internet denial of service activity. In *Usenix Security Symposium*, Washington, D.C, 2001.

[7] T. Onabuta, T. Inoue, and M. Asaka. A Protection Mechanism for an Intrusion Detection System Based on Mandatory Access Control. In *13th Annual Computer Security Incident Handling Conference*), Toulouse, France, June 2001.

[8] A. Ott. The Role Compatibility Security Model. In *7th Nordic Workshop on Secure IT Systems*, Karlstad, Sweden, November 2002.

[9] B. Schneier, and J. Kelsey. Cryptographic Support for Secure Logs on Untrusted Machines. In *7th USENIX Security Symposium Proceedings*, San Antonio, Texas, January 1998.

[10] T. Takada, and H. Koike. NIGELOG: Protecting Logging Information by Hiding Multiple Backups in Directories In *10th International Workshop on Database and Expert Systems Applications (DEXA99)*, Florence, Italy, September 1999.

[11] J. Viega, and G. McGraw. *Building Secure Software - How to Avoid Security Problems the Right Way.* Addison-Wesley, September 2002.

[12] C. Wright, C. Cowan, S. Smalley, J. Morris, and G. Kroah-Hartman. Linux Security Modules: General Security Support for the Linux Kernel. In *11th USENIX Security Symposium*, San Francisco, California, August 2002.

[13] Y. Zhang, H. Vin, L. Alvisi, W. Lee, and S. K. Dao. Heterogeneus Networking: A New Survivability Paradigm In *2001 workshop on New security paradigms*, New Mexico, 2001.