Revista Facultad de Ingeniería N.° 34. pp. 101-113. Septiembre, 2005

Sistema de seguridad en redes locales utilizando sistemas multiagentes distribuidos. Net-Mass

Daniel Horfan Álvarez, Andrew Mark Bailey***
*y Lucas Adrián Gómez Blandón****

(Recibido el 2 de julio de 2004. Aceptado el 17 de mayo de 2005)

Resumen

La seguridad en las redes de datos es uno de los elementos neurálgicos dentro de una organización, ya que requiere de diferentes estrategias de protección para salvaguardar puntos vulnerables, y de la coordinación y distribución de esfuerzos para cercar todas las posibles formas de ataques informáticos. En este artículo se propone un sistema multiagente distribuido como herramienta de protección para redes con diferentes sistemas operativos y susceptibilidad de diversos ataques.

El sistema propuesto está conformado por un grupo de agentes autónomos heterogéneos con características particulares en cuanto a sus estrategias de detección de intrusos y protección de los sistemas primordiales de la red, se tiene planeado que estos agentes basen sus decisiones en la aplicación de diferentes técnicas de inteligencia artificial como heurística de sistemas expertos, algoritmos evolutivos y redes neuronales.

---------- *Palabras clave:* seguridad informática, sistemas multiagentes, JAVA, JINI.

Distributed network multi agent security system. Net-Mass

Abstract

Network security is one of the weakest and most sensitive areas within an organization, because it requires different strategies to protect vulnerable points as well as the coordination and distribution of efforts in order to encompass all the possible forms and points of computer attacks. In this article a distributed multiagent system is proposed as a tool to protect security in networks with different operating systems and which consequently are susceptible to diverse attacks.

* Universidad de San Buenaventura, sede San Benito, carrera 56C N.º 51-90. Teléfono: (574) 511 36 00. Fax: (574) 513 82 94. Medellín. daniel.horfan@usbmed.edu.co.

** Universidad de San Buenaventura, carrera 56C N.º 51-90. Teléfono: (574) 511 36 00. Fax: (574) 513 82 94. Medellín. andy@hazlorealidad.com.

*** Estudiante de Ingeniería de Sistemas, Universidad de San Buenaventura, carrera 56C N.º 51-90. Teléfono: (574) 511 36 00. Fax: (574) 513 82 94. Medellín. lgomez@kpmg.com.co.

The system proposed is composed of heterogeneous autonomous agents with different characteristics in terms of their strategies of intruder detection and protection of primary network systems. It is planned that these agents will base their decisions on the application of different artificial intelligence techniques, such as expert system heuristics, evolutionary algorithms, and neural networks.

---------- *Key words:* information security, multiagent systems, JAVA, JINI.

Introducción

En las organizaciones de hoy la seguridad sobre la red de datos es fundamental, ya que ésta es como el sistema circulatorio que se encarga de llevar la información a cada una de sus partes, de tal forma que funcione como un todo. Organizaciones como SANS (Sys-admin, Audit, Network, Security) [1] dentro de sus estadísticas [2] muestran niveles de ataques en países como Estados Unidos de América que superan la cifra de 2.000.000 diarios (11.239.366 para el 31 de marzo de 2005), para el caso de Colombia, se tienen reportes que superan los 2.500 ataques diarios (2.911 para el 6 de marzo de 2004); es importante recordar que no todas las organizaciones efectivamente reportan los intentos de ataque y menos aún aquellos materializados para no comprometer la imagen de su organización.

Por su parte CERT (Carnegie Mellon Software Engineering Institute) [3] muestra en su artículo "State of the practice of intrusion detection technologies" [4] cómo los sistemas de detección de intrusos más eficaces corresponden a aquellos que funcionan en tiempo real.

En la seguridad en redes se tienen en cuenta muchas variables como son: intrusos, virus, control de tráfico en la red, ataques, entre otras. En este proyecto se diseña un sistema multiagente distribuido de monitoreo y alarma de eventos de seguridad en la red denominado Net-Mass. El sistema está concebido bajo características de estabilidad y robustez que impidan que él mismo pueda convertirse en el objeto de ataque. En particular, en este artículo se presentan las características del modelo de agentes distribuidos en cuanto al modelo interno del agente, la estructura organizacional y los protocolos de comunicación y notificación. En el proyecto se ha terminado el análisis y el diseño, y se encuentra en la fase de desarrollo y pruebas preliminares.

En el numeral uno se presentan los aspectos teóricos generales acerca de los sistemas multiagentes y la seguridad en redes; en el segundo numeral se presentan las características del sistema propuesto y el diseño e implementación del prototipo; por último en el numeral tres se presentan las conclusiones y perspectivas de trabajos futuros en el área.

Marco conceptual

A continuación se presenta un pequeño marco teórico acerca de los sistemas multiagentes y la seguridad en redes de datos para que sirva como referente conceptual de los temas por tratar en el cuerpo del presente artículo.

Agentes y sistemas multiagentes [5]

Agentes

En la literatura relacionada con el tema de agentes de *software* se han planteado diversas definiciones, que en resumen permiten definir un agente como un sistema computacional que opera de forma autónoma, con capacidad de reaccionar dinámicamente a estímulos internos y externos, que puede interactuar mediante la comunicación con sistemas afines y no afines y que posee aptitudes, conocimientos e intencionalidad para resolver un problema específico [6, 7, 8].

A los agentes de *software* se les asocian características como: intencionalidad, entendida como la declaración explícita de metas y medios para llegar a ellas; racionalidad, que es la capacidad de evaluar y seleccionar acciones para ejecutar; compromiso, consistente en la planificación de las tareas por medio de la coordinación y negociación entre agentes; adaptabilidad, referida al control de sus aptitudes y comportamientos de acuerdo con las funciones que ejecuta. Así mismo, los agentes pueden clasificarse desde diferentes puntos de vista en estáticos o móviles según su movilidad; en deliberativos y reactivos de acuerdo con su capacidad de reacción; y en autónomos o no, si operan sin necesidad de intervención humana [9].

Los agentes de mayor interés al hablar de inteligencia artificial son los autónomos, pues presentan características que los acercan al comportamiento humano, tales como: racionali-

dad, habilidad social, cooperación, capacidad de aprendizaje, reactividad y proactividad.

Aquellos agentes que combinan características propias de varias de las clasificaciones enunciadas anteriormente son denominados híbridos. Éstos se utilizan en aquellas aplicaciones en donde se considera que trae más beneficios tener un sólo agente con varias propiedades, en lugar de varios con cada una de ellas. Por ejemplo, se puede diseñar un agente que posea al mismo tiempo las características de deliberativo y de reactivo.

Sistemas multiagentes (SMA)

Desde que surgió la inteligencia artificial distribuida (IAD), se han abordado los temas relacionados con el estudio de los modelos y del comportamiento de varios agentes que cooperan entre sí para la resolución de un problema o desarrollo de una tarea. En particular, el área de los SMA estudia el comportamiento y la interacción de las entidades que integran el sistema. En los SMA, las entidades inteligentes que lo conforman se denominan agentes.

Dentro de las características principales de un SMA se tienen: la *no existencia de benevolencia,* lo cual hace a los agentes menos vulnerables a desviaciones del objetivo principal por la influencias externas; la *obtención de múltiples metas,* por la capacidad de resolver varios problemas e inclusive enfrentar el problema de varias maneras; la *autonomía* pues los agentes que conforman el SMA deben tener autonomía que les permita seleccionar la mejor manera para cumplir con sus objetivos y la *heterogeneidad,* ya que los agentes que conforman un SMA tienen características muy particulares, lo cual les permite intervenir en el problema de diferentes maneras. Esta heterogeneidad hace necesaria la existencia de un sistema de comunicación que permita la interacción entre los diferentes agentes.

En la arquitectura de los sistemas multiagentes existen ciertas características que definen el funcionamiento del sistema:

Estructura organizacional. Está relacionada con la estructura de las componentes funcionales del sistema, sus características, sus responsabilidades, sus necesidades y de la forma como realizan sus comunicaciones. En general, se pueden distinguir cuatro tipos de configuraciones: *centralizada*, donde un agente controla la interacción entre los otros agentes; *horizontal*, donde todos los agentes están en el mismo nivel; *jerárquica* donde existen diferentes niveles de agentes que trabajan distintos niveles de abstracción; y la estructura ad hoc que es una mezcla de las anteriores.

Cooperación. Se refiere a la forma como los agentes trabajan conjuntamente para lograr un objetivo global, la cual depende de la configuración organizacional del sistema. Entre los modelos de cooperación se tienen: la *cooperación compartiendo tareas y resultados,* donde con base en los resultados generados por los demás agentes, un agente realiza sus propias tareas; la *cooperación por delegación,* donde un agente divide una tarea en subtareas, las asigna a sus agentes para su ejecución, y por ultimo, las integra para hallar la solución global; y la *cooperación por ofrecimiento,* donde un agente divide una tarea en subtareas y las ubica en una lista esperando que los agentes del sistema se ofrezcan a realizarlas dependiendo de sus habilidades.

Coordinación. Entre un grupo de agentes permite considerar todas las tareas que se van a realizar por medio de una planificación de acciones para la resolución de tareas. Esta planificación lleva a que se conozca con anticipación el comportamiento de los agentes y por tanto, a través de la coordinación, permitir que un grupo de agentes analice las tareas por realizar y se coordinen para ejecutarlas. Cuando el SMA determina y planifica globalmente las acciones de los diferentes agentes se dice que tiene una *coordinación global* y cuando cada agente decide qué hacer y resuelve localmente los conflictos que detecte con otros agentes se dice que tiene una *coordinación individual.*

Control. Es un instrumento que provee apoyo en la implementación de mecanismos de coordinación. El control puede ser considerado desde el punto de vista *local* o *global*. Se debe mantener un balance entre el control *global* (basándose en la información producida por todos los agentes del sistema) y *local* (solamente con base en datos locales), pues si se emplea mucho control global se puede aumentar considerablemente el tiempo de cómputo, debido al cambio dinámico de la información producida por los agentes, por el contrario si se desborda el control local, se pierde eficiencia en la cooperación, pues se pueden realizar tareas no deseables.

Lenguaje de comunicación KQML [10]. La comunicación en los sistemas multiagentes es primordial, ya que es el medio por el cual los agentes comparten conocimiento y se sincronizan para llevar a cabo sus tareas conjuntamente. KQML se planteó en función de un modelo de interacción dinámica entre agentes inteligentes llamado KBS (Knowledge Based System). Este modelo permite evitar las limitaciones que imponían los modelos distribuidos, y a la vez hacer más fácil la interacción entre verdaderos agentes inteligentes. El modelo más simple es el cliente-servidor, en el cual uno de los agentes actúa como cliente formulando una petición a otro agente, el cual actúa como servidor proporcionando la respuesta.

La comunicación entre agentes utilizando KQML puede ser sincrónica o asincrónica. La primera de ellas se refiere a que el mensaje no se envíe de forma completa sino fraccionada, estableciéndose una comunicación entre los dos agentes de forma que el cliente vaya pidiendo al servidor sucesivamente las respuestas; la asincrónica consiste en que el cliente se suscribe a un determinado servidor que, de manera asincrónica, envía la respuesta.

En la estructura de comunicación de KQML se distinguen los siguientes niveles estructurados de forma similar al modelo OSI:

Nivel de mensaje. Constituye el núcleo del mensaje y es el que determina la clase de interacciones que puede tener un agente con otro.

Nivel de contenido. Constituye el conocimiento o petición que se desee comunicar en un lenguaje determinado, por ejemplo una pregunta, una respuesta, una orden, entre otros.

Nivel de transporte o comunicación. Contiene los parámetros básicos de la comunicación como la información del emisor y el receptor.

Seguridad en redes

Si bien es cierto que al nivel de la estructura de red se han generado cambios importantes en seguridad en los últimos diez años, también lo es que los avances en técnicas para comprometerla a ella y a sus elementos han avanzado tan rápidamente como estos. Técnicas como *Man in the middle* la cual también se conoce como *TCP hijacking* en conjunto con las llamadas *lluvias ARP* o *ARP poison* han logrado comprometer el tráfico de red. Soluciones como SSL/TLS, certificados asimétricos, IPSec, PPTP y otros nacen a partir de estas necesidades y evolucionan en la medida en que se hacen nuevos hallazgos en la formulación de sus especificaciones iniciales que los hacen vulnerables nuevamente a ataques.

Siendo éste el propósito de una solución de identificación de intrusos (ya que se tendría la capacidad de identificar este tipo de ataque a través de un sistema de estos) no es suficiente cuando habla del contenido en la red. Los nuevos ataques se han orientado no solo a tomar información de la red sino a colocar (también conocido como *inyectar*) contenidos en ella, para el caso en que el atacante tiene un ataque dirigido y bien estructurado, como parte del tráfico, mas esta situación no es común: el nivel de dificultad que tiene este tipo de ataque lo limita en quienes pueden ejecutarlo mientras que para el caso de un código viral es posible incluso hacerlo sin tener ningún nivel de experiencia.

La detección de código viral en el tráfico de red se hace posible en la medida en que se tiene la capacidad de analizar los paquetes y en ellos detectar las firmas de éste. Sabiendo que no es un método totalmente efectivo, sí puede reducir enormemente el riesgo para el promedio de casos presentados.

Detección de intrusos [11, 12]

Un sistema de identificación de intrusos (IDS, Intruder Identification System) tiene por objetivo identificar posibles ataques. Un IDS genera un proceso de alertas, tomando una base de datos con condiciones que tipificarían los diferentes tipos de vulnerabilidades conocidas. Estas alertas deben estar ajustadas a los usos reales de la red (operaciones normales) y así detectar los comportamientos anómalos casuales o intencionados (también llamados ataques dirigidos). Entre los de su tipo, existen los *Network IDS*, los cuales usan sensores de red para la recolección de datos; los *Host IDS*, que se orientan a la protección de un sistema particular y monitorean las acciones que se realizan en éste; y los Application IDS, similares a los host IDS pero orientados a un servicio específico, por ejemplo, bases de datos (ORACLE, MS SQL Server, MySQL, etc.), servicios de red (servicio de correo, navegación de usuarios, sitios web, etc.).

Los NIDS (Network IDS) tienen la capacidad de actuar en forma pasiva y activa; en la primera, se generan las alertas y se informa la situación que se presenta, en el segundo caso, el NIDS genera el proceso de interrumpir la conexión y rechazar las solicitudes de conexión generadas desde el origen detectado.

Detección de virus

Los detectores de virus en el tráfico de red son similares a los sistemas de detección de intrusos con la diferencia que su base de conocimiento contiene las firmas de patrones de virus identificados. Algunos de los sistemas de detección de virus utilizan heurística y otras técnicas de inteligencia artificial como medios de optimización de su trabajo.

Diseño e implementación del sistema de seguridad basado en sistemas multiagentes distribuidos. Net-Mass

El Net-Mass es una aplicación de apoyo al proceso de gestión de la seguridad en redes de datos. Su desarrollo está fundamentado en la filosofía de agentes inteligentes que operan de forma coordinada y autónoma, con características de seguridad y estabilidad frente ataques externos y con conocimientos y aptitudes que les permiten evidenciar una posible violación a las políticas de seguridad de la red y notificarlo al área administrativa para su solución. A continuación se presentan las principales características del sistema en sus etapas de diseño e implementación.

Diseño del sistema

Modelo del agente

Un agente tiene una estructura interna compuesta por una serie de componentes que interactúan entre sí para brindarle autonomía y habilidades propias, que le permitan reaccionar de forma proactiva frente a estímulos externos, producidos en este caso particular en la red de datos. Dentro del Net-Mass los agentes tienen una estructura constituida de los siguientes componentes.

Componente de ejecución. Es la parte del agente encargada de llevar a cabo las labores o trabajos que le han sido asignados. En esta componente se tienen desarrolladas diferentes utilidades dependiendo de la especialidad del agente, usando para ello técnicas de inteligencia artificial como redes neuronales, sistemas expertos y algoritmos evolutivos, entre otros, y rutinas o subprogramas que realizan cálculos definidos.

Componente funcional. Esta es una parte fundamental del agente, ya que en ella se encuentran las utilidades que le dan su autonomía y comportamiento. Este componente se desarrolla usando técnicas de sistemas expertos, donde se encuentran las reglas que se disparan para activación de la clonación, modificaciones en la planificación y el control del agente.

Componente de comunicación. Es una serie de métodos que permiten la publicación de las habilidades de un agente y la interacción de éste con los demás miembros del sistema. Esta componente utiliza un protocolo de comunicación híbrido el cual es explicado en el numeral *Protocolo de comunicación.*

Agentes del sistema

En el sistema Net-Mass se tienen 6 tipos de agentes a saber:

Agentes de escucha. Son los encargados de captar los paquetes que llegan a la tarjeta de red y realizar un preanálisis del encabezado de capa transporte, de red y de enlace de datos (capas 4, 3 y 2 del modelo OSI) basado en una rutina de reglas estructuradas en un sistema experto que producen un diagnóstico en cuanto a la confiabilidad del paquete. Luego de realizar dicho análisis, el agente decide, de acuerdo con el grado de confiabilidad, si el paquete requiere de evaluación más profunda, y en tal caso lo delega a los agentes de detección de virus e intrusos. La configuración del sistema de reglas de calificación se puede realizar en forma paramétrica o se puede colocar al agente a trabajar en una etapa de aprendizaje, en la cual con la ayuda del administrador genera su base de reglas.

Agentes de detección de intrusos de red (NIDS). Estos se encargan de realizar un análisis profundo en cuanto a la identidad del origen del paquete de datos para detectar posibles intrusos, para su implementación se tiene estimado conformar una red neuronal y un sistema de clasificación genética. Una de las características es su capacidad de identificar cambios y con base en las decisiones del administrador reestructurar las reglas en las que basa la caracterización del tráfico.

Agentes de detección de virus. Éstos se encargan de realizar un análisis profundo en cuanto a la presencia de firmas de virus; para su implementación se tiene estimado conformar un sistema experto y un sistema de clasificación genética.

Agentes de coordinación. Éstos se encargan de realizar la coordinación y control de los agentes de los demás agentes; los agentes de coordinación realizan una planificación de los recursos y de acuerdo con estos realizan la distribución óptima de los agentes dentro de un segmento de red.

Agentes de reporte. Éstos agentes proveen al administrador una serie de utilidades donde se pueden observar las estadísticas del tráfico de red y los eventos de seguridad sucedidos. Todos los agentes del sistema, durante su ejecución envían estadísticas parciales al agente de reportes y éste los organiza para presentarlos al administrador de la red. A este agente se le puede asociar un servidor web para que su información pueda ser consultada desde un punto remoto.

Agentes de alarma y auditoría: estos son los encargados de notificar cuando un evento de seguridad es identificado. Este tipo de agente solo se ejecuta en el área administrativa.

En la figura 1 se presenta el esquema de interacción entre los agentes y su ubicación dentro de la red, y se observa cómo los agentes se distribuyen al interior con el fin de monitorear todo el tráfico ante posibles ingresos de intrusos o ataque de un virus.

Arquitectura del sistema

En la arquitectura del sistema se involucran: la estructura organizacional, los métodos de cooperación y las técnicas de coordinación y control del sistema.

Estructura organizacional. El Net-Mass utiliza una estructura organizacional ad-hoc, pues al interior del sistema se utilizan diferentes esquemas de organización según el nivel de abstracción y el tipo de comunicación que se presente entre los agentes. La estructura organizacional del sistema se esquematiza en la figura 2.

Cooperación. En el del sistema multiagente propuesto se utilizan dos modelos de cooperación, uno orientado a compartir tareas y resultados el cual se presenta entre los agentes de mismo tipo, y

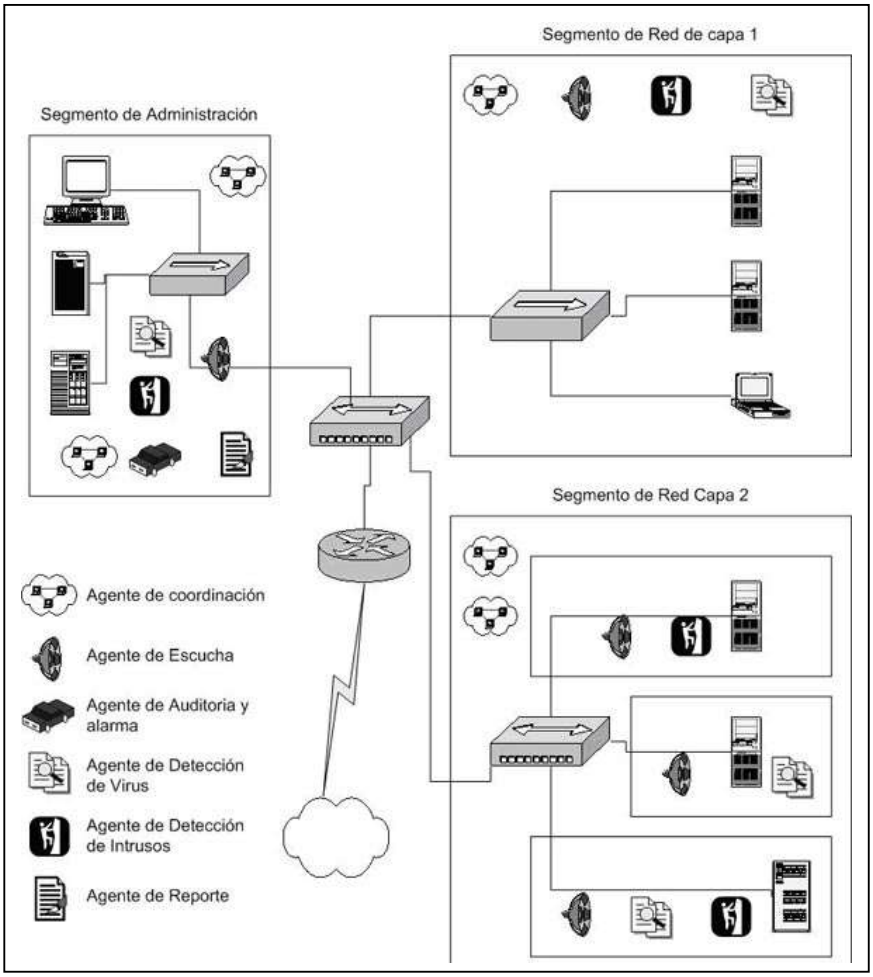

Figura 1 Distribución de los agentes en la red en Net-Mass

otra por delegación entre los agentes, los agentes de escucha y detección.

Coordinación y control. En el sistema multiagente la coordinación global está a cargo de los agentes de coordinación los cuales realizan tareas de organización y control sobre un dominio de colisión o segmento de red. Sin embargo, al interior de cada agente se incluye un componente de control y coordinación individual, que le permita tener autonomía para sus decisiones.

Protocolo de comunicación

Como componente de comunicación hemos evaluado varios protocolos sincrónicos como: UDP[1], UDP Multicast, TCP[2], RMI[3], RMI-IIOP y protocolos asíncronos como JAVA Spaces y

1 UDP: User Datagram Protocols. Protocolo de comunicación no orientado a la conexión, basado en el envío de datagramas entre dos computadores de una red. Multicast se refiere a la comunicación entre varios PC en la red.

2 TCP: Transmission Control Protocols. Protocolo de comunicación orientado a la conexión, basado en el envío de segmentos entre computadores de la red.

3 RMI: Remote Method Invocation. AP que permite la invocación de métodos en aplicaciones que se ejecutan en forma remota. IIOP se refiere al protocolo de RMI sobre Corba [13].

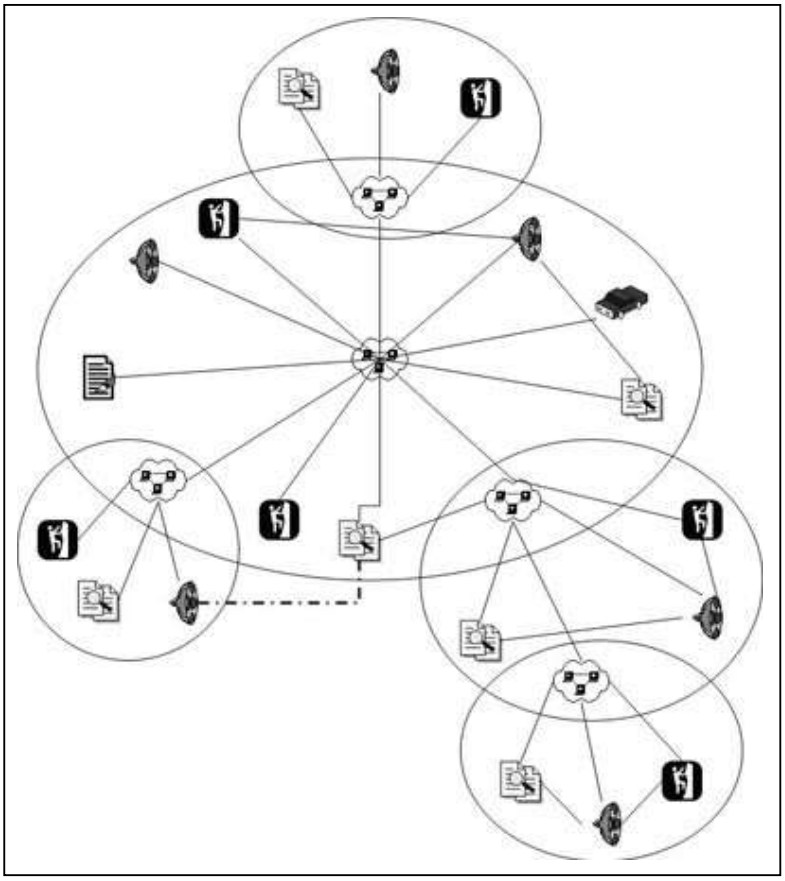

Figura 2 Estructura organizacional del Net-Mass

JAVA Messaging Service—JMS—,[4] cada uno de los cuales brinda características particulares con sus respectivas ventajas y desventajas. Hemos optado por una solución de comunicación híbrida extensible en la cual se combina las técnicas de UDP *Multicast*, RMI y JMS.

Entre los agentes que se encuentran en el mismo segmento de red se utilizan métodos sincrónicos soportados sobre RMI y UDP Multicast utilizando el *framework* de JINI[5] y entre las comunicaciones hacia los agentes de administración se realizan utilizando JMS [15].

Debido al alto flujo de información que se puede presentar en el sistema se optó por un protocolo de comunicación entre los agentes basado en los conceptos de KQML, pero utilizando llamadas RMI si el servicio es remoto (es decir en otro computador) y llamadas locales, si el servicio es local (dentro de la misma máquina virtual de JAVA en el mismo computador). Esto tiene la ventaja de no requerir la codificación, transmisión y decodificación de un mensaje en formato texto como lo propone KQML, lo cual tendría alto costo computacional y de transporte en la red.

4 JMS: JAVA Message Service. Es una API desarrollada por Sun en colaboración con otras empresas que permite el acceso a sistemas de mensajería de terceros [14].

5 JINI: Sistema que permite la publicación y descubrimiento de servicios brindados por sistemas distribuidos [16, 17].

Sin embargo, en el futuro se le puede adaptar un facilitador con formato KQML para comunicarse con otros agentes, lo cual es posible dadas las características de modularidad con la que es construido cada componente del agente.

En un sistema distribuido es esencial que haya una forma de encontrar servicios disponibles, de ahí que en el proceso de comunicación, se ha involucrado una etapa de búsqueda de los agentes dentro de la red. Ésta es realizada por el protocolo de descubrimiento de JINI, el cual permite encontrar servicios disponibles en aplicaciones distribuidas de forma fácil, mediante el registro de servicios de *hardware* o *software* en una red, y su posterior búsqueda y utilización. Las características de este protocolo permiten que el servicio de registro y búsqueda pueda caerse sin interrumpir el correcto funcionamiento del sistema.

Entre el sistema de agentes y los agentes de coordinación y administración se utiliza un sistema asíncrono de mensajes, de tal forma que los mensajes sobre la seguridad de la red no se pierdan, aun cuando la aplicación de administración no esté en ejecución o si existe un problema en la comunicación de la red. Este esquema de comunicación utiliza la interfaz del JAVA Message Service —JMS— y su implementación JAVA System Message Queue.

Adicionalmente se ha configurado el sistema de comunicación para que sea tolerante a fallas, y en caso de que el sistema de mensajes falle se pasa el control a otro igual de respaldo. Entre los predicados que propone KQML se seleccionaron los que se muestran en la tabla 1 para la comunicación entre agentes.

Monitoreo de agentes

Para que la aplicación administrativa pueda monitorear el sistema de agentes se ha utilizado JAVA Management Extensión —JMX— [18] que permite administrar de forma remota los agentes.

Aunque el objetivo principal de los agentes es ser autónomos, se ha empleado JMX que permite la comunicación con los agentes y su monitoreo, con el propósito de depuración, lo cual brinda la confianza que el sistema está funcionando adecuadamente y opcionalmente intervenir en la interacción de los agentes.

La arquitectura JMX esta compuesta por un modelo de tres capas:

- *Nivel de instrumentación*, que brinda una interfaz para manejar cualquier objeto de JAVA.
- *Nivel de agentes*, el cual contiene agentes JMX para manejar los objetos en el nivel de instrumentación. Los agentes JMX son contenedores de objetos de JAVA que brindan servicios de manejo.
- *Nivel de manejo* para este caso incluye la aplicación administrativa, este nivel brinda componentes para manejar agentes.

Control de seguridad de los agentes

Para los servicios se utiliza la tecnología RMI, la cual además brinda la seguridad necesaria frente a ataques al propio sistema. Estas características de los agentes se presentan a continuación:

Tabla 1 Predicados de KQML utilizados

Acceso y estado	*Pregunta*	*Respuesta*	*Organización*	*Comunicación*
Deny	Evaluate	Sorry	Recommend one	Advertise
Standby	Reply	Error	Recommend all	Subscribe
Ready	Ask-if	Response	Recruit one	
	Ask-about		Recruit all	
	Ask-one			

Comunicaciones. El diseño del modelo de la seguridad para la tecnología de JINI se construye en las nociones de un elemento *principal* y de una *lista de control de acceso* (Access Control List, ACL). Los servicios de JINI están orientados a nombre de un elemento principal y contienen generalmente las entradas para cada nuevo usuario particular del sistema (en este caso, cada agente). Gracias a esto, puede solicitar el acceso a otros servicios basados en la identidad del objeto que pone el servicio en ejecución. Si el acceso a un servicio está permitido dependerá del contenido de la lista de control de acceso que se asocie al objeto.

Labores de coordinación de los agentes. Se propone la marcación de las operaciones por la selección de una semilla (SALT en inglés) inicial al momento de clonación y con ella la generación de mensajes críticos en el sistema encriptados bajo protocolos como MD5. La acción que permitirá el ingreso del clon al nuevo sistema se apoyará en el sistema de autenticación de cada sistema operativo como tal, es decir, para el caso de los sistemas operativos UNIX se contaría con usuarios y claves encriptadas en MD5, así como para el caso de sistemas operativos Windows estas estarían encriptadas en NTLMv2.

Predicados de comunicación entre los agentes (KQML). Similar a lo propuesto para la tarea de coordinación de los agentes, los predicados serán enviados adicionando la marcación para establecer la validez de los coordinadores. Cabe anotar que todo el sistema usará la misma semilla y en ciertos ciclos de tiempo como parte de los mensajes se enviará una nueva por parte de los coordinadores con un listado de padres para el coordinador final del agente que va a ser gobernado.

Implementación del sistema de captura de paquetes

Debido a que puede haber múltiples plataformas de ejecución como Windows, Linux y MacOs en la red se seleccionó JAVA como lenguaje de desarrollo dadas sus características de multiplataforma. Para permitir que JAVA tenga acceso a la información del tráfico en la red se desarrolló una interfaz a la librería de captura de paquetes denominada Libpcap. Esta interfaz está desarrollada utilizando la JAVA Native Interface (JNI) en C++.

Cuando un computador está conectado a una red ethernet mediante concentrador *(hub),* la tarjeta de red recibe todos los paquetes enviados en el segmento local de la red, en su funcionamiento normal dicha tarjeta solo envía los paquetes destinados para su dirección física al núcleo del sistema operativo. Se puede configurar la interfaz de red en un modo promiscuo, lo cual significa que todos los paquetes que recibe la tarjeta serán enviados al sistema operativo. Esto quiere decir que un solo agente puede monitorear un segmento de red, y para asegurar que se monitoree toda la red se debe tener un agente de escucha por cada segmento de la misma.

En la realidad utilizamos varios agentes por cada segmento de red porque cada agente tiene características específicas de funcionamiento y funcionalidad. En caso de que el segmento de red esté conectado con un *switch* en vez de un *hub*, sólo los dos equipos involucrados en la transmisión de los datos reciben los paquetes de la comunicación. En este caso, para monitorear la red completamente cada equipo debe tener un agente de escucha, lo cual presenta una complejidad extra en la coordinación de los agentes.

Libpcap es una librería escrita en C que coloca la interfaz de red en el modo promiscuo y cada vez que llega un paquete llama a un método en C++. Para la implementación del proyecto la mayor parte del código está escrito en JAVA, y utiliza la librería Libpcap a través de JNI, la interfaz nativa de JAVA, que permite que la aplicación JAVA invoque procedimientos en C y que el código C pueda invocar métodos JAVA.

Una vez el paquete es enviado desde el Libpcap al componente de JAVA, se carga una estructura de datos en la cual se almacenan los encabezados de capa 2, 3 y 4 del modelo OSI, y se analizan mediante un sistema experto.

Estado del proyecto

En el proyecto se han terminado con éxito las fases de análisis y diseño, actualmente se encuentra en su fase de desarrollo. Hasta el momento se tienen completamente desarrollados los siguientes componentes:

- Captura de paquetes de la interfaz.
- Desencapsulamiento de paquetes.
- Estructura funcional de agentes (comportamiento, clonación y control).
- Protocolo de comunicación basado en JINI (publicación y suscripción de servicios de cada agente).
- Detección de intrusos basado en tablas ARP.

En el funcionamiento del sistema se han involucrado comportamientos frente a eventos particulares entre los cuales se tienen los siguientes:

- Cuando dentro de un segmento de red existe mucho tráfico se crea un clon del agente de escucha y entre ambos negocian las direcciones MAC que cada uno atiende. El nuevo agente de escucha publica su interfaz mediante el mensaje "advertise" y se subscribe a su inmediato agente de coordinación con el mensaje "subscribe".
- Cuando un dispositivo de capa 1 (por ejemplo un *hub*-concentrador) es reemplazado por uno de capa 2 (por ejemplo un *switch*), los agentes de escucha detectan el cambio en la red y realizan una petición de clonación al agente coordinador para poblar los nuevos segmentos de red, generando nuevos agentes de escucha y detección.
- Cuando el agente de escucha pasa un paquete de datos al agente de detección para que realice una verificación y el agente no responde, se reporta el hecho al agente de coordinación, el cual mediante un mensaje de "recomend one" o "recomend all" le notifica al agente de escucha la ubicación del agente de otro segmento que puede realizar dicha labor.
- Cuando un agente de escucha o detección realiza el llamado a su agente de coordinación y éste no le responde, realiza un llamado de broadcast donde notifica la ausencia y esta es atendida por los agentes de coordinación de otros segmentos que crean un clon en el segmento y este recluta nuevamente los agentes del segmento mediante la utilización de mensajes "recruit one" y "recruit all".

Comparativos con software *similares del mercado*

Herramientas de detección de intrusos tanto comerciales (CISCO IDS, RealSecure, Symantec ManHunt, entre otros) como libres (SNORT, MIDAS, entre otros) están limitados entre la disposición de sus agentes o sensores y su capacidad, en caso que el ataque no esté dirigido a la red sino a ellos como elementos de actualización, reorganización y generación de un proceso efectivo para enfrentar dicha situación. Más allá de la capacidad de operar en múltiples plataformas, el sistema espera poder generar esquemas adaptativos que representen incluso el estado actual de la red. En este momento, ninguna herramienta comercial lo ofrece como parte de sus cualidades.

Conclusiones y recomendaciones

Los sistemas multiagentes distribuidos pueden ser aplicados en diversas disciplinas permitiendo la optimización de resultados mediante aplicaciones de menor tamaño y que consumen menores recursos de computador, trabajando de forma distribuida.

Gracias al sistema de clonación dinámico involucrado en los agentes del sistema se garantiza el cubrimiento de todos los segmentos de una red frente ataques o entrada de intrusos.

Esta solución propuesta tiene por ventaja la independencia de un elemento particular para realizar el trabajo de "escuchar la red", de tal forma que los ataques orientados a los sensores (agentes de escucha) gracias a la estructura de multiagentes, no podrán comprometer el funcionamiento del sistema.

En el desarrollo del componente de ejecución de los agentes de detección se tiene planeado incorporar técnicas de inteligencia artificial que puedan mejorar la eficacia de las técnicas de detección actualmente usadas.

Referencias

1. http://www.sans.org. Consultado el 31 de marzo de 2004.
2. http://isc.sans.org/country report.html. Consultado el 18 de febrero de 2004.
3. http://www.cert.org. Consultado el 31 de marzo de 2004.
4. http://www.cert.org/archive/pdf/99tr028.pdf. Consultado el 31 de marzo de 2004.
5. D. Horfan et al. "Método para manejar el problema de la recarga de trabajo en los sistemas multiagentes". Memorias Tecnocom 2003.
6. B. Hayes-Roth. "An Architecture for Adaptive Intelligent System". Artificial intellignece: special issue on agents and interactivity. Vol. 72. N.º 1-2. Enero de 1995. pp. 72, 329-365. (Disponible electrónicamente en: https://portal.acm.org/poplogin.cfm?dl=ACM&coll=portal&comp_id=COMPONENT030&want_href=citation%2Ecfm%3Fid%3D201271&CFID=24619593&CFTOKEN=21127073. Consultado el 28 de marzo de 2004).
7. S. C. Smith et al. KidSim: programming agents without a programming language. Communications of the ACM. Julio de 1994. pp. 37, 55-67. (Disponible electrónicamente en: http://www.acypher.com/Publications/CACM/KidSimCACM.html. Consultado el 28 de marzo de 2004).
8. P. Maes. Artificial life meets entertainment: lifelike autonomous agents. Communications of the ACM. 1995. pp. 38, 108-114. (Disponible electrónicamente en: http://portal.acm.org/ft_gateway.cfm?id=219808&type=pdf&dl=portal&dl=ACM&CFID=11111111&CFTOKEN=2222222. Consultado el 28 de marzo de 2004).
9. N. Hyacinth. "Software agents: and overview". Knowledge Engineering Review. Vol. 11. N.º 3. Cambridge University Press. September 1996. pp. 1-40. (Disponible electrónicamente en: http://agents.umbc.edu/introduction/ao/. Consultado el 31 de marzo de 2004).
10. T. Finin et al. KQML as an agent communication language. Reporte técnico. Departamento de Ciencias Computacionales. Universidad de Mayfield Baltimore Country. Baltimore, MD. 1997. (Disponible electrónicamente en http://www.cs.umbc.edu/agents/introduction/kqmlacl.ps. Consultado el 27 de marzo de 2004).
11. P. Mell et al. An overview of issues in testing intrusion detection systems. National Institute of Standards and Technology ITL. 2002.
12. http://csrc.nist.gov/publications/nistir/nistir-7007.pdf. Consultado el 28 de marzo de 2004.
13. http://JAVA.sun.com/j2se/1.4.2/docs/guide/rmi-iiop/index.html. Consultado el 28 de marzo de 2004.
14. http://JAVA.sun.com/products/jms. Consultado el 28 de marzo de 2004.
15. W. K. Edwards. Core JINI. 2.ª ed. Prentice-Hall. 2000.
16. http://JAVA.sun.com/products/JINI/. Consultado el 31 de marzo de 2004.
17. http://www.JINI.org. Consultado el 31 de marzo de 2004.
18. http://JAVA.sun.com/products/JAVAManagement/. Consultado el 31 de marzo de 2004.