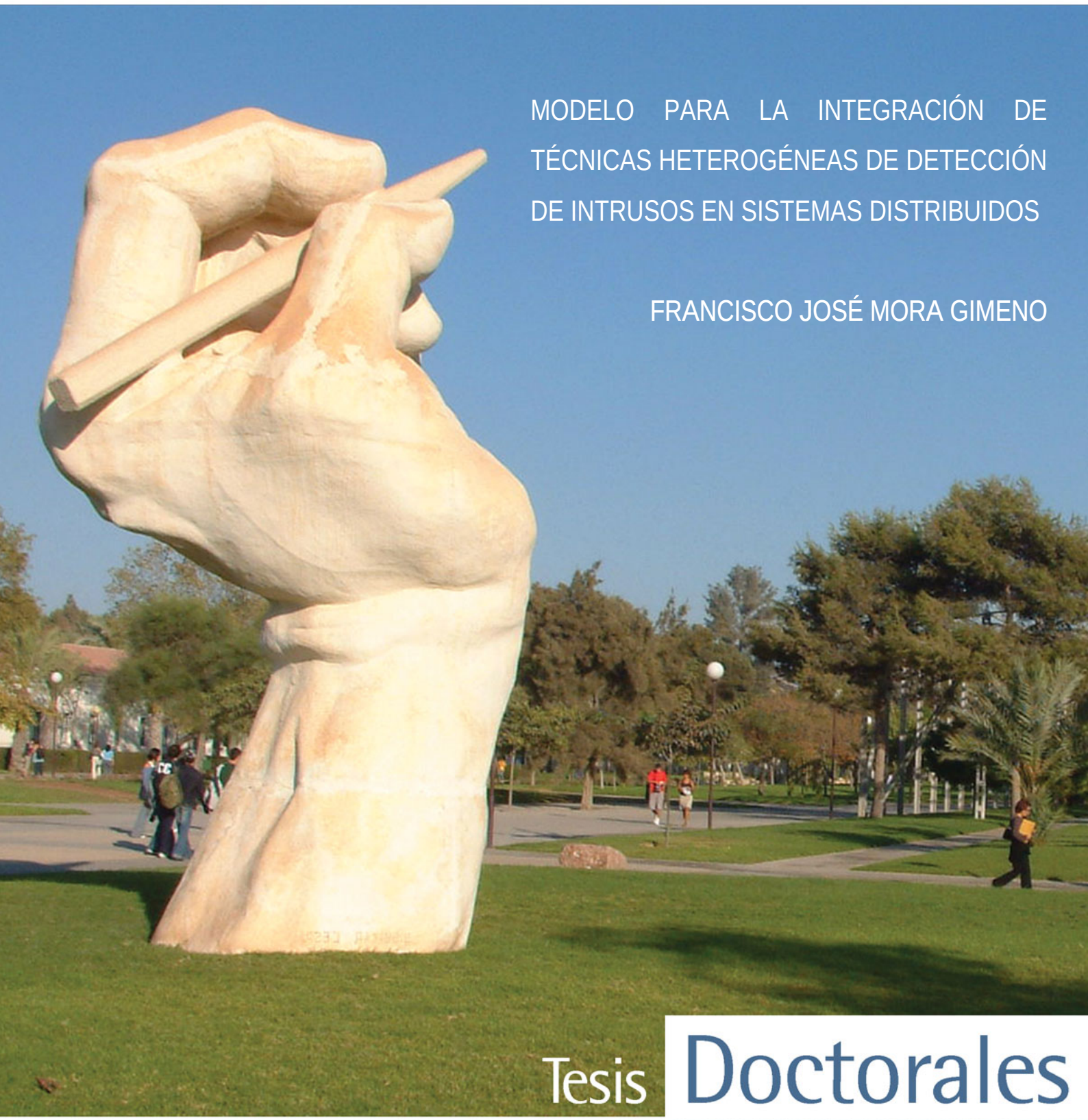
Universitat d'Alacant
Universidad de Alicante

MODELO PARA LA INTEGRACIÓN DE TÉCNICAS HETEROGÉNEAS DE DETECCIÓN DE INTRUSOS EN SISTEMAS DISTRIBUIDOS

FRANCISCO JOSÉ MORA GIMENO



Tesis Doctorales

www.eltallerdigital.com

UNIVERSIDAD de ALICANTE

Tesis Doctoral

Modelo para la Integración de Técnicas Heterogéneas de Detección de Intrusos en Sistemas Distribuidos

UNIVERSIDAD DE ALICANTE

TESIS DOCTORAL

MODELO PARA LA INTEGRACIÓN DE TÉCNICAS HETEROGÉNEAS DE DETECCIÓN DE INTRUSOS EN SISTEMAS DISTRIBUIDOS

Presentada por
FRANCISCO JOSÉ MORA GIMENO

Dirigida por
DR. FRANCISCO MACIÁ PÉREZ

DEPARTAMENTO DE TECNOLOGÍA INFORMÁTICA Y COMPUTACIÓN

DICIEMBRE DE 2009

Para Rosa, Sara y Jordi

Sólo sé que no sé nada

Sócrates *(470 AC-399 AC) Filósofo griego.*

Agradecimientos

La intención de estas palabras es mostrar mi más sincero agradecimiento a todas las personas que han contribuido a la realización de esta humilde investigación y, aunque pudieran parecer sólo palabras, son mucho más que eso, porque os aseguro que las escribo con el corazón. Me gustaría pensar que no voy a olvidar a nadie, pero ya sabéis que soy el *tonto del pueblo* y va a ser imposible. Por esta razón, os pido disculpas por adelantado.

En primer lugar, con todo el cariño del mundo, mi más sincero agradecimiento a mi familia, a mi mujer Rosa y a mis hijos Sara y Jordi. A Rosa, por su paciencia infinita y apoyo constante a lo largo de más de 20 años. Para una persona como yo, que nunca juega a la lotería, el azar me ha otorgado la mejor de las fortunas. Gracias. A Sara y Jordi, porque al verlos cada mañana comprendo que incluso alguien como yo puede hacer algo bien en la vida. Sois mi razón de vivir. Os quiero. A mi madre Antonia, por creer siempre en mí. A los ausentes: mi padre José y mi hermano Antonio, porque estéis donde estéis sé que estáis conmigo.

En segundo lugar, mis agradecimientos de todo corazón a mi amigo, mi director de tesis. El presente trabajo lleva más aportación tuya que mía, pero, qué le vamos a hacer, soy un egoísta. Gracias, por el apoyo, las ideas, los consejos, la visión, las críticas constructivas, tus conocimientos, infinitamente

mayores que los míos, tu accesibilidad y muchísimas cosas más que sería imposible enumerar. Escribiste en tu tesis que esperabas seguir trabajando conmigo, han pasado 9 años y así ha sido, que continúe la fiesta. Moltes gràcies Paco.

Al grupo de investigación José Vicente, Joan Carles, Héctor, Jorge Selva, Felipe, Antonio, Jorge Gea y la sección cubana. Por vuestra ayuda, apoyo, aportaciones, discusiones y, en definitiva, por todas las sesiones en las que me habéis soportado estoicamente. A Diego y Virgilio, por ser las personas del grupo a quien más he dado la vara mientras comíamos en esos restaurantes con encanto y, pese a ello, entre broma y broma, me habéis aportado ideas constructivas. A Juan, por tus conocimientos sobre la gestión de redes que tienes a bien enseñarme, pero, también, por las largas charlas sobre política, problema sin solución que no impide el debate entre las buenas gentes de izquierda y derecha. Gracias a todos.

A todo el departamento, porque no hay nadie que en algún momento no me haya dado apoyo, consejo y conocimiento. A José, amigo y compañero de despacho, por el ánimo y apoyo constante en todo tipo de momentos y por ser el sabio que resuelve todas mis dudas. Gracias. A todos, muchas gracias.

A todos. Gracias.

Castalla, 8 de diciembre de 2009

Francisco José Mora Gimeno

Resumen

En la presente tesis se ha llevado a cabo una investigación detallada sobre los problemas de seguridad de las redes de computadores, dentro del ámbito de los Sistemas de Detección de Intrusos Distribuidos o DIDS y centrada fundamentalmente en el campo de la correlación e integración de las alertas generadas por este tipo de sistemas. El principal resultado de este trabajo ha sido la creación de un modelo general de detección de intrusos distribuido que permite de manera sistemática la integración de múltiples métodos de correlación de alertas.

El trabajo aborda los siguientes aspectos fundamentales:

- Una revisión del estado del arte sobre los aspectos relacionados con los sistemas de detección de intrusos distribuidos y los mecanismos empleados por éstos para correlacionar las alertas producidas.
- Desarrollo de un marco formal que constituye el contexto en el cual definir y especificar formalmente el modelo.
- Creación y formulación de un modelo general de detección distribuido que permite la integración de múltiples métodos de correlación de alertas.
- Definición de un método de integración que, basándose en técnicas de aprendizaje de máquina y empleando una métrica de evaluación de IDS, permite la integración de múltiples métodos de correlación, mejorando el rendimiento

de estos métodos por separado gracias al aprovechamiento de la información sobre las capacidades de las técnicas de correlación que forman parte de dicha integración.

- Diseño de una arquitectura completamente distribuida del sistema con criterios de escalabilidad, flexibilidad y adaptabilidad que permite implantar el modelo en entornos reales.
- Diseño y realización de un conjunto de experimentos cuyos resultados demuestran la validez de la propuesta y, por ende, de las hipótesis de partida.

Resum

En la present tesi s'ha portat a terme una investigación detallada sobre els problemes de seguretat de les xarxes de computadors, dins de l'àmbit dels Sistemes de Detecció d'Intrusos Distribuïts o DIDS i centrada fonamentalment en el camp de la correlació i integració de les alertes generades per aquest tipus de sistemes. El principal resultat d'aquest treball ha segut la creació d'un model general de detecció d'intrusos distribuït que permet de manera sistemática la integració de múltiples mètodes de correlació d'alertes.

El treball aborda els següents aspectes fonamentals:

- Una revisió de l'estat de l'art sobre els aspectes relacionats amb els sistemes de detecció d'intrusos distribuïts i els mecanismes emprats per aquests per a correlacionar les alertes produïdes.
- Desenvolupament d'un marc formal que constitueix el context en el qual definir i especificar formalment el model.
- Creació i formulació d'un model general de detecció distribuït que permet la integració de múltiples mètodes de correlació d'alertes.
- Definició d'un mètode d'integració que, basant-se en tècniques d'aprenentatge de màquina i emprant una mètrica d'avaluació de IDS, permet la integració de múltiples mètodes de correlació, millorant el rendiment d'aquests mètodes per

separat gràcies a l'aprofitament de la informació sobre les capacitats de les tècniques de correlació que formen part d'aquesta integració.

- Disseny d'una arquitectura completament distribuïda del sistema amb criteris de escalabilitat, flexibilitat i adaptabilitat que permet implantar el model en entorns reals.
- Disseny i realització d'un conjunt d'experiments els resultats dels quals demostren la validesa de la proposta i, per tant, de les hipòtesis de partida.

Abstract

In this thesis a detailed investigation on the security problems of computer networks has been carried out within the field of Intrusion Detection Systems Distributed or DIDs and focused primarily on the field of correlation and integration of warnings generated by such systems. The main result of this work has been the creation of a general model of distributed intrusion detection that allows a systematic way to integrate multiple methods of correlating of alerts.

The work addresses the following aspects:

- A review of the state of the art on system aspects of distributed intrusion detection mechanisms used by them to correlate the alerts produced.
- Develop a formal framework is the context in which to define and specify the model formally.
- Creating and developing a general model of distributed detection that allows the integration of multiple methods of correlating of alerts.
- Define a method of integration based on machine learning techniques and using an evaluation metric of IDS, enabling integration of multiple methods of correlation, improving the performance of these methods separately by leveraging information about the capabilities of correlation techniques as part of this integration.

- Design of a fully distributed architecture of the system with criteria of scalability, flexibility and adaptability that allows us to deploy the model in real environments.
- Design and broad of a set of experiments whose results demonstrate the validity of the proposal and hence the initial hypotheses.

Resumen del Contenido

Contenido

Figuras y Tablas

Figuras

Tablas

Capítulo 1

Introducción

El incremento exponencial del uso de las tecnologías de la información y las comunicaciones alcanza todos los ámbitos de la sociedad. Hoy en día, estamos conectados casi de forma permanente a Internet, tanto en casa como en el trabajo. En los computadores desde los que realizamos estas conexiones tenemos información privada que no debe ser conocida sin nuestro consentimiento (Pollach, 2007) y es muy probable que empleemos nuestro acceso a Internet para realizar operaciones sensibles que requieren seguridad, como transferencias bancarias, compra-venta de acciones o transacciones de cualquier tipo.

Para las empresas, Internet se ha convertido en un nuevo canal de comercialización de sus bienes y servicios, incluso en un generador de nuevos modelos de negocio (Karagiannis *et al.*, 2007). Los modelos e-Business se han introducido en empresas de la más diversa índole: desde grandes empresas hasta administraciones de lotería, hoy venden sus productos en todo el mundo sin importar su ubicación.

También los Gobiernos ponen sus servicios electrónicos a disposición de los ciudadanos y de las organizaciones. Los particulares realizamos la declaración del IRPF, las empresas presentan sus modelos fiscales a la Administración y las

Universidades solicitan proyectos de investigación, todo ello a través de Internet.

Pero, en este nuevo escenario, la interconexión y acceso a un mundo globalizado de servicios electrónicos tiene sus riesgos. Uno de los más importantes radica en la seguridad de nuestras redes y sistemas. Los riesgos de seguridad vienen determinados por distintos factores, entre los que destacan los ataques o actividades maliciosas llevadas a cabo por un atacante y las propias vulnerabilidades de los sistemas. En cuanto al primer factor, el número medio de computadores atacados por día en el segundo semestre del año 2007 fue de 62.000, un incremento del 17% respecto al primer semestre del mismo año (Turner *et al.*, 2008). Por otra parte, como se puede comprobar en la figura 1.1, el número de nuevas vulnerabilidades descubiertas cada año se ha incrementado sustancialmente en la última década, pasando de 200 en el año 1998 a casi 7.000 en el año 2007. Estos datos nos recuerdan, provocando cierta alarma social, la vulnerabilidad y los riesgos de las redes informáticas y de comunicaciones.

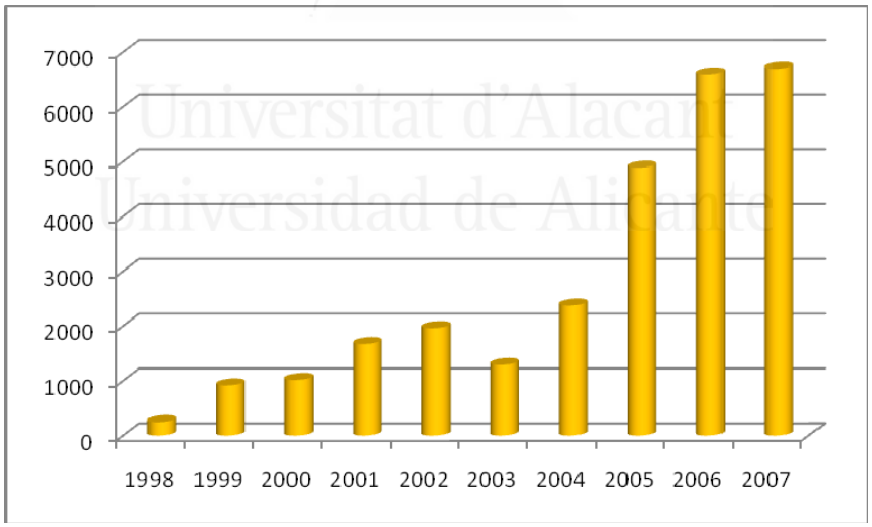


Figura 1.1. Número de vulnerabilidades por año (Nist, 2007)

Existen varios aspectos que dan idea de la preocupación e importancia social de la seguridad de redes en el ámbito

institucional: según la consultora Ernst & Young, ha habido un aumento importante de las inversiones en seguridad de la información en los últimos años (Ernst & Young, 2006); el 93% de las empresas europeas invierten en la seguridad de sus redes (Fundetec, 2007); por último, como se puede observar en la figura 1.2, el número de instituciones de EEUU que reconocen más de 10 incidentes de seguridad al año se incrementó en un 200% el año 2007 (CSI, 2008).

Figura 1.2. Instituciones que reconocen incidentes de seguridad

Por otra parte, instituciones científicas y empresas de todo el mundo están dedicando gran esfuerzo a la investigación de todos los aspectos relacionados con la seguridad de redes, como lo demuestra la existencia de una enorme cantidad de publicaciones y congresos centrados exclusivamente en temas de seguridad informática y otros que, aun perteneciendo a otros campos de las tecnologías de la información, incluyen aspectos relacionados con la seguridad computacional. Además, existen congresos dedicados exclusivamente a alguno de los aspectos o herramientas de seguridad como los sistemas de detección de intrusos (IDS —Intrusion Detection System). La figura 1.3 muestra el número de congresos de seguridad que organizan o

con los que colaboran las principales asociaciones internacionales relacionadas con la informática y la electrónica.

Figura 1.3. Incremento del número de congresos de seguridad de IEEE y ACM

Los datos anteriores muestran la relevancia del problema de la seguridad informática como campo de investigación. Para minimizar los riesgos, pues asumimos que la seguridad total no existe, es imprescindible que utilicemos todas las herramientas de seguridad a nuestro alcance. De esta manera, cortafuegos, sistemas de detección de intrusos, mecanismos de control de acceso y demás utilidades, se convierten en aliados dentro de una buena política de seguridad (Qin y Lee, 2003). La figura 1.4 muestra las tecnologías más utilizadas por las empresas de EEUU.

Las herramientas como el control de acceso y los cortafuegos son muy importantes en la seguridad porque impiden el acceso no autorizado y muchos de los ataques externos. No obstante, debido a las vulnerabilidades de los propios sistemas, es posible acceder sin autorización, incluso aunque el mecanismo de control de acceso esté perfectamente diseñado (Josang *et al.*, 2007). Por otra parte, el 70% de los ataques informáticos provienen de la

propia red interna de la organización, por lo que no pueden ser filtrados por los cortafuegos.

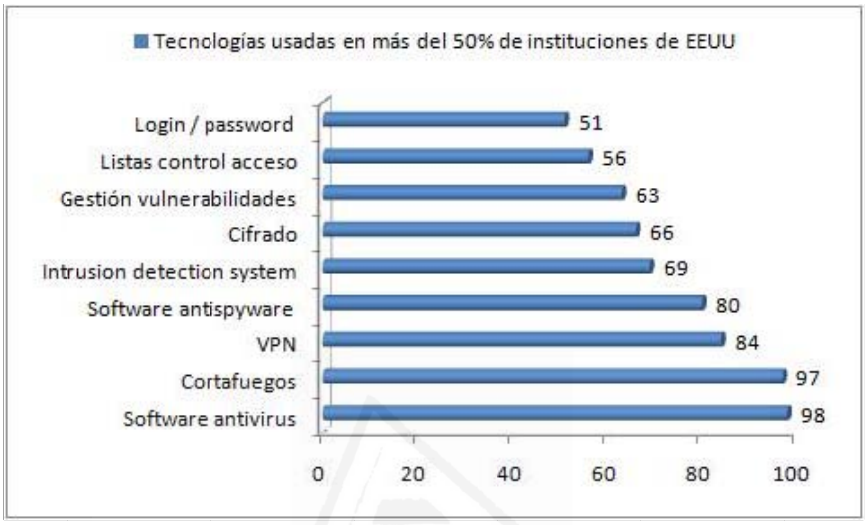


Figura 1.4. Tecnologías de seguridad más utilizadas

Por lo tanto, para detectar tanto los ataques que se producen dentro de la red como los provenientes del exterior que no hayan sido detenidos por el cortafuegos, es necesario utilizar mecanismos de seguridad activos como los sistemas de detección de intrusos (IDS —Intrusion Detection System). Los IDS monitorizan continuamente la red y los sistemas con el objetivo de detectar cualquier ataque o indicio de actividad maliciosa (Braun *et al.*, 2008).

La tarea del administrador de seguridad en el problema de la detección de intrusos no es sencilla debido a numerosos factores, entre los que se puede destacar los siguientes:

- Los sistemas y servicios son cada vez más sofisticados y complejos. Debido a ello, son cada vez más difíciles de diseñar, por lo que es más probable que puedan tener algún fallo de seguridad en el propio diseño (Proctor, 2001).
- La configuración de los sistemas y servicios es cada vez más complicada, por lo que algún error en la configuración puede

dejar al descubierto nuestras redes y sistemas (Northcutt y Novak, 2003).

- Constantemente aparecen nuevos ataques y vulnerabilidades para los cuales el sistema tiene que ser actualizado con la mayor celeridad posible (Kruegel y Vigna, 2003).
- El número de nodos interconectados crece vertiginosamente, nodos que también deben ser considerados en la política de seguridad global del sistema.

Cuando un sistema de detección descubre un ataque, o cualquier otra actividad maliciosa, informa al administrador de seguridad mediante una *alerta*. Un ataque no se suele producir de manera aislada, sino que pertenece a un escenario mayor formado por un conjunto de ataques. La conexión lógica existente entre distintas alertas pertenecientes al mismo escenario, lo inabordable de su tratamiento manual debido al gran volumen de las mismas (Qin y Lee, 2003) y la tendencia a incluir en el análisis alertas de otros sistemas de seguridad distintos a los IDS, constituyen tres de las principales razones que han motivado la aparición hace pocos años de mecanismos de correlación de alertas. La correlación de alertas es un proceso que toma como entrada las alertas producidas por uno o más sistemas de detección de intrusos y proporciona como salida una descripción de alto nivel de la actividad maliciosa de la red (Kruegel *et al.*, 2005).

Puesto que uno de los aspectos que diferencia las distintas técnicas de correlación de alertas es la cantidad de información que es necesario codificar previamente para su análisis, los métodos van desde los que requieren todo el conocimiento del dominio, hasta los que no necesitan ninguna información previa para su funcionamiento.

Los métodos de correlación mediante la *especificación de escenarios* utilizan todo el conocimiento disponible con el fin de especificar patrones de escenarios completos y modelar la correlación como un problema de reconocimiento de patrones (Qin y Lee, 2004). La mayoría de trabajos que utilizan esta técnica lo hacen basándose en algún lenguaje de especificación

que permita definir los patrones de búsqueda. Ejemplos de estos lenguajes son LAMBDA (Cuppens y Ortalo, 2000), Chronicles (Morin y Debar, 2003) y STATL (Eckmann *et al.*, 2002).

Los sistemas basados en el método de especificación de escenarios presentan resultados muy favorables en cuanto a capacidades de detección, tienen una altísima probabilidad de reconocer aquellos escenarios almacenados en la base de datos. Dado que saben perfectamente lo que buscan, son extremadamente fiables y no suelen cometer errores, por lo que prácticamente carecen de *falsos positivos*, es decir, etiquetar o clasificar como ataque un comportamiento lícito del sistema. No obstante, a pesar de las ventajas que presentan, tienen limitaciones como el tiempo necesario para codificar los escenarios y, sobre todo, su incapacidad para detectar escenarios nuevos (Qin y Lee, 2004).

Para reducir la cantidad de información que es necesario codificar explícitamente, un segundo tipo de sistemas establece los *prerrequisitos y consecuencias* de cada ataque individual con el objetivo de identificar relaciones en el proceso de correlación. La idea principal es que el éxito de un ataque posibilita la ejecución de otro ataque posterior, por lo tanto, el mecanismo de correlación asocia consecuencias de un ataque anterior con prerrequisitos de otro posterior (Templeton y Levit, 2000), (Ning *et al.*, 2002) y Cuppens y Miege, 2002).

Este tipo de sistemas presentan las ventajas de necesitar menos información y, por tanto, menos tiempo de codificación de las precondiciones y las consecuencias. También poseen cierta capacidad para detectar pequeñas variaciones sobre escenarios conocidos. Por el contrario, tiene el inconveniente de no detectar escenarios nuevos o grandes variaciones y, además, introduce el problema de los *falsos positivos* o errores en el diagnóstico.

Con el objetivo de dotar al sistema de correlación de alertas de la capacidad de detección de escenarios nuevos, un tercer tipo de métodos abordan el problema sin tener en cuenta en el proceso de correlación información a priori de los escenarios o los

ataques. Este tipo de técnicas se denomina *similitud entre los atributos de las alertas o clustering*. Estos sistemas se basan en buscar similitudes o relaciones entre los atributos de las alertas, de tal forma que aquéllas que tengan valores similares o relacionados en sus atributos pertenecerán al mismo grupo. La idea es que las alertas agrupadas en la misma clase forman parte del mismo escenario (Cuppens, 2001), (Dain y Cunningham, 2001), (Valdes y Skinner, 2001), (Staniford *et al.*, 2002) y (Qin y Lee, 2003).

Los métodos de clustering presentan la ventaja de detectar escenarios nuevos o desconocidos por no basarse en conocimiento previo. Sin embargo, tienen el problema grave de obtener un índice muy elevado de falsos positivos. Para conseguir sistemas de correlación que mejoren el rendimiento, se ha propuesto en la literatura enfoques que integran de manera ad hoc pares de las técnicas comentadas. Así, en (Qin y Lee, 2003) ya se aborda la necesidad de integración de algoritmos de correlación y en [Ning *et al.*, 2002] se recomienda utilizar su técnica junto con otras técnicas complementarias.

Basándose en el enfoque de integración, (Qin y Lee, 2004) complementan un motor de correlación de prerrequisitos y consecuencias con análisis estadístico. Por otra parte, (Ning *et al.*, 2004) utilizan un enfoque de integración de prerrequisitos y consecuencias con técnicas de clustering.

Estos enfoques que integran dos métodos de correlación de forma ad hoc presentan mejor rendimiento que aquéllos que únicamente usan una técnica. Aun así, los resultados siguen siendo débiles: muchos casos muestran ratios de detección del orden del 60%, notoriamente bajos, pero, sobre todo, índices de falsos positivos del 7% en el mejor de los casos, claramente excesivos. Tales resultados alejan la posibilidad de que los mecanismos de correlación de alertas sean utilizados en entornos reales de producción.

Los sistemas de detección actuales adolecen de problemas como la incapacidad de detectar escenarios nuevos o el elevado índice

de falsos positivos (Northcutt y Novak, 2003). Además, los métodos utilizados para correlacionar escenarios completos no son del todo efectivos y deberían ser usados junto con otras técnicas complementarias (Ning *et al.*, 2002). Los nuevos sistemas de detección deben detectar ataques tanto a nivel de red como de nodo y aplicación, tener la capacidad de correlacionar alertas de distintos IDS con el objetivo de descubrir escenarios completos y estrategias de alto nivel e, incluso, poder integrar información proveniente de herramientas de seguridad distintas como los cortafuegos. Además, estas tareas se tienen que llevar a cabo de la manera más eficiente posible, es decir, maximizando la capacidad de detección tanto de escenarios conocidos como nuevos y minimizando el índice de falsos positivos o errores.

Según lo anterior, el campo de acción de esta investigación se enmarca en el ámbito de los sistemas de detección de intrusos distribuidos, más concretamente en aquéllos que llevan a cabo procesos de correlación para obtener el estado de seguridad global de la red.

El objeto de estudio de la presente tesis se puede ubicar en el desarrollo de metodologías de integración de múltiples métodos de correlación complementarios con el objetivo de conseguir sistemas de detección eficientes, en términos de capacidades de detección y reducción de errores a niveles manejables por el administrador de seguridad.

La revisión anterior muestra el interés social, económico y científico del problema de la seguridad informática. Prueba de ello es el esfuerzo inversor de empresas e instituciones en tecnologías de seguridad, así como el notable incremento del número de congresos de seguridad organizados por los principales organismos internacionales.

La investigación en los sistemas de detección ha evolucionado desde la detección de ataques aislados hasta la correlación de alertas en busca de escenarios completos de mayor nivel. Los avances producidos en el proceso de correlación, aun siendo evidentes, distan mucho de poder ser utilizados en entornos de

producción. No obstante, los resultados obtenidos por la integración ad hoc de pares de técnicas muestran que este enfoque es una vía de investigación prometedora y factible para conseguir sistemas de correlación que resuelvan el problema de su implantación en entornos reales.

Identificación del Problema

Tal y como se puede deducir de lo expuesto en los apartados previos, todos los métodos revisados tienen sus propias ventajas e inconvenientes. Unos consiguen grandes capacidades de detección a costa de no reconocer escenarios nuevos y otros permiten la detección de escenarios nuevos pero presentan el problema del alto número de falsos positivos o errores en la correlación. En este sentido, los enfoques que integran pares de métodos de manera ad hoc tienen mejor rendimiento que aquellos que únicamente usan una técnica. No obstante, los resultados continúan siendo claramente insuficientes y las mejoras conseguidas son cada vez más ajustadas sin llegar a una solución real factible.

De forma sintetizada, se puede definir el problema a abordar en la presente Tesis de la siguiente manera: la carencia de una metodología general que permita integrar de forma sistemática múltiples técnicas de correlación impide el aprovechamiento de las sinergias de cada una de ellas y, por lo tanto, la obtención de rendimientos mejores que cada una por separado. La falta de una solución general provoca que se tenga que optar por soluciones ad hoc que precisan mucho mantenimiento, sintonización constante y producen resultados pobres.

En cualquier caso, tomando como referencia los trabajos anteriores, se puede observar que los enfoques que utilizan integración ad hoc de dos métodos complementarios presentan mejores resultados en términos de rendimiento, obtención de un ratio de detección elevado, junto con un índice de falsos positivos manejable. Esta observación apoya la vía de la integración de métodos complementarios como el camino a seguir para obtener

rendimientos plenamente satisfactorios. No obstante, el hecho de que integre únicamente dos métodos complementarios y lleve a cabo el proceso de integración de manera ad hoc limita sus posibilidades, en cuanto a rendimiento y a la posible incorporación de nuevos métodos.

Hipótesis y Objetivos

Definimos la función $Perform()$ como el rendimiento obtenido por un sistema de detección de intrusos. Este resultado se debe poder evaluar mediante la utilización de alguna métrica. Así mismo, mediante $Integ()$ especificamos la operación de integración de los resultados de un conjunto de métodos de correlación.

A partir de las definiciones anteriores, establecemos las siguientes hipótesis de partida:

1. La generalización de la idea de la integración y su formalización permiten la creación de modelos que integran múltiples métodos de correlación.

2. La integración de varios métodos de correlación proporciona resultados superiores a los obtenidos individualmente por esos mismos métodos.

   Siendo C el conjunto de todos los posibles métodos de correlación y $C_a \subset C$ un subconjunto de funciones de correlación $C_1, C_2, \dots, C_n$, donde cada $C_i \in C_a$, entonces:

   $$Perform\binom{Integ(C_i)}{\forall C_i \in C_a} \geq Perform(C_i), \forall C_i \in C_a \qquad [1.1]$$

3. La ordenación de las técnicas de correlación en función de su rendimiento permite establecer un método capaz de obtener el mejor resultado del modelo de integración.

   Si, dadas dos funciones de correlación C_i y C_j:

   $$Perform(C_i) \geq Perform(C_j) \qquad [1.2]$$

Entonces

$$Perform\big(Integ(C_i \circ C_j)\big) \geq Perform\big(Integ(C_j \circ C_i)\big) \quad [1.3]$$

Por lo tanto, el objetivo general de la investigación es la creación de un modelo general y formal para la detección de intrusos que automatice la integración de múltiples técnicas de correlación. El modelo propuesto estará compuesto por:

- La definición de los elementos involucrados.
- Una metodología de integración que maximice el rendimiento del sistema basada en una ordenación de los métodos de correlación en función de la información que cada uno de ellos proporciona.
- Una arquitectura distribuida que permita la implantación en entornos de producción.

El objetivo general enunciado en el párrafo anterior dará cumplimiento a las hipótesis de partida, aspecto central del presente trabajo. No obstante, para el desarrollo de la investigación se establecen los siguientes objetivos específicos:

- Crear un modelo de IDS que aborde las limitaciones de los sistemas actuales, en especial, la detección y correlación de ataques nuevos con criterios de eficiencia. Es decir, la obtención de ratios de detección elevados tanto de escenarios nuevos como conocidos, pero manteniendo un índice de falsos positivos reducido.
- Diseñar una arquitectura que haga viable el modelo propuesto en escenarios realistas. Una de las características deseables de todo IDS es que imponga la mínima sobrecarga posible sobre el sistema en el que se está ejecutando. No obstante, para llevar a cabo el proceso de detección y correlación es necesario recoger, analizar y procesar grandes cantidades de información, aspecto que origina un consumo elevado de recursos. Por lo tanto, otro aspecto fundamental para que este tipo de investigaciones puedan tener una viabilidad realista será plantear

arquitecturas, organizaciones y componentes basadas en las tecnologías existentes.

- Desarrollar herramientas que implementen el modelo y permitan su utilización por parte de los administradores de seguridad en entornos de producción. Las herramientas, eminentemente gráficas, permitirán el control y la gestión de todo el sistema, aportando una visión del estado de seguridad global de la red.

Metodología y Plan de Trabajo

La metodología de trabajo tiene en cuenta distintos aspectos del método científico como el análisis de otras propuestas, síntesis y creación de modelos y, por supuesto, experimentación. Entre los métodos de trabajo científico utilizados merece la pena destacar los siguientes:

- Se propone una revisión del estado del conocimiento mediante la observación y el análisis de los trabajos relacionados con los sistemas de detección de intrusos, los métodos de correlación de alertas y las propuestas de integración ad-hoc de técnicas. Los *métodos analítico* y de *observación científica* permiten ver las carencias de los sistemas actuales.
- Partiendo del análisis del estado del arte y, concretamente, del estudio de los trabajos que utilizan integración de métodos de correlación de manera ad-hoc, se inducen las hipótesis de integración general de técnicas mediante el *método lógico inductivo*, que permite generalizar conocimiento basándose en casos particulares.
- A partir de las hipótesis definidas y siguiendo el *método sintético* se formula un modelo general de detección que unifica todos los elementos involucrados en el proceso.
- Con el objetivo de establecer un método de integración general que permita obtener el máximo rendimiento, se

propone el uso del *método de medición* para definir métricas basadas en la teoría de la información que evalúen la relevancia de los distintos métodos de correlación.

- Después de formular el modelo de detección y establecer el método general de integración se diseña una arquitectura capaz de aportar viabilidad en entornos realistas. Para ello, se utiliza el *método sistémico* que permite modelar un sistema mediante la determinación de su estructura, componentes y las relaciones entre ellos.
- Finalmente, con el objetivo de probar las hipótesis mediante la experimentación, se emplean *métodos empíricos* en el desarrollo y aplicación de un prototipo sobre escenarios de prueba, y *métodos estadísticos* para la evaluación cuantitativa de los resultados obtenidos.

Las tareas de investigación que se proponen para llevar a cabo estos objetivos son las siguientes:

1. Formalizar y generalizar un modelo de detección que permita integrar cualquier método de correlación existente.
2. Estudiar las capacidades de las redes neuronales artificiales y de la aplicación de algoritmos de clustering a la integración de distintos métodos de correlación de manera general.
3. Definir y proponer un algoritmo que, basándose en la cantidad de información que proporciona cada método de correlación, mida la información aportada por cada método para permitir la ponderación de los distintos métodos de forma que, en consecuencia, se maximice el rendimiento de los métodos integrados.
4. Analizar las tecnologías existentes con el objetivo de proponer una arquitectura que haga viable el modelo dentro del marco tecnológico existente en la actualidad.

5. Implementar un prototipo que incluya los distintos componentes del modelo general de detección y evaluar la propuesta en distintos escenarios de prueba.

Las principales aportaciones de la investigación se enmarcan en el campo de la generalización y formalización de modelos globales de detección, con la propuesta de un modelo que permitirá la utilización, dentro de un mismo sistema, de distintos tipos de IDS y múltiples métodos de correlación, todo ello con los mecanismos de integración apropiados. Esta generalización favorece la creación de sistemas de detección con capacidades para abordar potencialmente todos los problemas: detección de nuevos escenarios, ratios bajos de falsos positivos, etc. Otra contribución relevante es la mejora del proceso de integración mediante la ponderación de las técnicas de correlación en función de la cantidad de información que proporcionan o ganancia, lo que permitirá incrementar el rendimiento de los sistemas de detección en términos de aumento de la capacidad de detección y reducción de falsos positivos.

A partir de este momento y para una mejor comprensión de los aspectos fundamentales tratados en la presente tesis, hemos estructurado la memoria en distintos capítulos que son un reflejo bastante fiel del plan de trabajo seguido.

El capítulo 2 aborda el estudio del estado del conocimiento en el ámbito de la detección de intrusos, analizando de manera general aspectos como: el concepto, las técnicas de detección, los tipos y las organizaciones de los IDS actuales. Además se profundiza en los temas relacionados con el proceso de correlación de alertas y sus métodos.

El capítulo 3 está dedicado a establecer un marco formal a partir del cual se diseñará el modelo general de detección e integración de alertas y se desarrollará todo el formalismo asociado.

En el capítulo 4 se realiza el estudio profundo de los métodos de correlación de alertas y, especialmente, de los mecanismos de integración de los resultados de las correlaciones. Se desarrollará la metodología que permitirá la integración de múltiples métodos

de manera general, basándonos en la ordenación de las distintas técnicas de correlación en función de la cantidad de información que aportan al resultado final y en aspectos relacionados con la teoría de grafos.

Mientras que los dos capítulos anteriores desarrollan respectivamente las hipótesis de partida, en el capítulo 5 se aborda la viabilidad real planteando una arquitectura del sistema que soporte el modelo.

En el capítulo 6 se llevará a cabo la validación de la propuesta a través del rendimiento del modelo mediante la evaluación, sobre distintos escenarios, de la implementación de un prototipo. Describiremos los escenarios utilizados, las pruebas realizadas y, principalmente, los resultados empíricos obtenidos.

Finalmente, dedicaremos el capítulo 7 se dedica a la exposición de las principales conclusiones y aportaciones del trabajo así como a plantear las líneas futuras de investigación que se desprenden del mismo.

Capítulo 2

Estado del Arte

En el capítulo anterior se estableció como objetivo de la investigación crear un modelo general de detección de intrusos que permita la integración de múltiples métodos de correlación en pos de obtener mejores rendimientos que los sistemas actuales y superar las carencias intrínsecas de las que adolece cada uno de ellos. Es por esta razón que el estudio del estado del conocimiento se ha centrado fundamentalmente en el campo de los IDS, su organización, los diferentes mecanismos de correlación empleados, y los trabajos más actuales en este área.

El análisis de los IDS nos permite comprender, desde distintos puntos de vista, la gran variedad de sistemas de detección que existen en la actualidad, las diferentes técnicas utilizadas en la detección e, incluso, las debilidades y fortalezas de cada uno de ellos. El estudio de su organización mostrará las diversas arquitecturas, junto con sus ventajas e inconvenientes. Examinando los métodos de correlación de alertas abordaremos en profundidad las técnicas empleadas para relacionar varias alertas que pertenecen al mismo escenario de ataque. Finalmente, la exploración de los trabajos que utilizan integración nos proporcionará una visión más general de las ventajas de esta idea, pero también de sus limitaciones, vislumbrando la necesidad de seguir investigando en la búsqueda de modelos que permitan la integración general de múltiples métodos, así como

de metodologías que posibiliten un aprovechamiento máximo del proceso de integración.

Sistemas de Detección de Intrusos

Aunque el concepto de detección de intrusos se remonta a (Anderson, 1980), el primer modelo de detección no se presentó hasta 1986 (Denning, 1986). Este modelo expuso las hipótesis en las cuales se basan los IDS actuales. Las violaciones de seguridad podían ser detectadas observando patrones de uso anormal del sistema. Esta idea llevó a la construcción de IDS con arquitectura monolítica, aplicados a un único computador.

Con la proliferación de las redes de computadores, los investigadores aplicaron la tecnología usada en los IDS de un único computador a pequeñas redes. Desafortunadamente, la extensión directa de los sistemas locales a pequeñas redes no resulta suficiente en los actuales entornos fuertemente interconectados, heterogéneos y complejos (Proctor, 2001). Tales sistemas deberían ser capaces de colaborar para la detección de ataques distribuidos. Por ejemplo, un evento en un computador puede no significar nada; el mismo evento en varios o todos los computadores de una red, durante un corto espacio de tiempo, puede significar que se ha producido o se está produciendo un ataque (Carzaniga *et al.*, 2007).

Además de la capacidad para colaborar, en los entornos actuales, sería deseable que los sistemas de detección contaran con las siguientes características (Northcutt y novak, 2003):

- *Ejecución continua*. El sistema debe ejecutarse continuadamente, sin interrupción, con la mínima supervisión humana.
- *Tolerante a fallos*. Debe ser capaz de recuperarse de una caída del sistema, sea esta accidental o producida por alguna actividad maliciosa. Una vez reiniciado, debe ser capaz de recuperar inmediatamente su estado previo.

- *Resistente a la subversión*. El IDS debe poder monitorizarse a si mismo y detectar si ha sido modificado por un atacante.
- *Mínima sobrecarga*. Debe provocar la menor sobrecarga posible en el sistema donde se está ejecutando para no interferir en las tareas normales del computador.
- *Políticas de seguridad*. Se tiene que poder adecuar a las políticas de seguridad del sistema que está monitorizando.
- *Escalable*. El sistema de detección debe ser capaz de adaptarse a la previsible evolución de los servicios e infraestructuras de forma que pueda seguir monitorizando grandes cantidades de información sin perder eficiencia y con los mismos tiempos de respuesta.
- *Reconfiguración dinámica*. Es impracticable reiniciar el sistema completo cada vez que se realiza un cambio, máxime si el número de computadores que están siendo monitorizados es grande.

Tipos de IDS

Los IDS son generalmente clasificados en función de diferentes factores no excluyentes. Según esto, atendiendo al origen de la información, es decir, al lugar donde se recogen los datos que serán utilizados en el análisis (Cretu *et al.*, 2008), encontramos la siguiente clasificación:

- *IDS basados en host (HIDS —Host Intrusion Detection System)*. Sistemas que basan su decisión en información obtenida del computador, generalmente a partir de los históricos del sistema.
- *IDS basados en aplicación*. Trabajan con información producida por las aplicaciones específicas.
- *IDS basados en red (NIDS —Network Intrusion Detection System)*. Sistemas que basan su decisión en información obtenida de la propia red, mediante la monitorización del tráfico de la red en la cual están conectados los equipos.

Los NIDS tienen la ventaja de no afectar el rendimiento de las computadoras que protegen debido a que utilizan como fuente el tráfico de red (Zanero y Savaresi, 2004). No obstante, tienen una gran limitación en las actuales redes conmutadas ya que su ubicación es un elemento clave para su buen funcionamiento. Además, entre sus desventajas se encuentra su incapacidad para detectar intrusiones en paquetes cifrados (Wagner, 2004), puesto que pueden ser únicamente descifrados en los nodos extremos de la comunicación y no en un elemento intermedio (Frincke *et al.*, 2007).

Mientras que los NIDS operan en toda una red, protegiendo a un conjunto de máquinas, los HIDS protegen únicamente a la máquina en la que son instalados (Liang y Sekar, 2005). Utilizan como fuente de información los datos generados por la computadora en la que operan (Sharif *et al.*, 2007), especialmente a nivel de sistema operativo: archivos de auditoría del sistema, archivos *logs* o cualquier archivo que el usuario desee proteger. Los HIDS tienen como ventaja su capacidad para detectar situaciones como los intentos fallidos de acceso o modificaciones en archivos considerados críticos (Kruegel y Vigna, 2003). Las desventajas principales de este tipo de IDS se encuentran en el consumo de recursos del propio equipo que desea proteger y el hecho de que él mismo está expuesto a ser víctima de algún ataque.

Otra clasificación muy utilizada en la literatura es la que se obtiene según la estrategia o filosofía de análisis, donde se pueden llevar a cabo dos tipos de análisis distintos (Kruegel *et al.*, 2005b):

- *IDS de abusos*. Se define el comportamiento anormal del sistema, usando conocimiento específico de los ataques para detectarlos. El análisis se basa en buscar patrones de ataques perfectamente conocidos. Su funcionamiento es similar al de los antivirus.
- *IDS de anomalías*. El análisis en este tipo de sistemas consiste en modelar el comportamiento normal del objeto que

está siendo monitorizado. Después se interpretará cualquier desviación sobre el comportamiento normal como anómala, siendo clasificada como ataque.

Dentro de los sistemas de abusos, los primeros sistemas utilizaban algún lenguaje de reglas o sistema experto para definir abstracciones de ataques denominadas *firmas* y almacenarlas en una base de conocimiento (Roesch, 1999), (Neumann y Porras, 1999). Los mecanismos de transición entre estados también permiten describir firmas de ataques complejos (Eckmann *et al.*, 2002) y (Kemmerer, 2005). Actualmente, existen trabajos que permiten generar firmas automáticamente a partir de vulnerabilidades (Brumley *et al.*, 2006) y (Cui *et al.*, 2007), creando paquetes anómalos aleatoriamente (Wang *et al.*, 2006), basándose en el análisis forense de la memoria del servidor víctima (Liang y Sekar, 2005) o partiendo del polimorfismo de los gusanos para generar automáticamente firmas que los detecten (Perdisci *et al.*, 2006) y (Van Gundy *et al.*, 2007).

Los IDS basados en anomalías se pueden construir mediante muchas técnicas distintas para modelar el comportamiento normal y llevar a cabo el proceso de análisis (Tan y Maxion, 2002) y (Majorczyk *et al.*, 2008). La herramienta más utilizada es el análisis estadístico, donde el modelo normal está formado por un conjunto de variables estadísticas (Robertson *et al.*, 2006). Por ejemplo, una combinación lineal de varias medidas (Kruegel y Vigna, 2003) y (Balzarotti *et al.*, 2007) o una métrica basada en la distribución de caracteres para detectar anomalías en el contenido de los paquetes de red (Wang y Stolfo, 2004).

No obstante, existen muchos sistemas de anomalías que usan técnicas de aprendizaje de máquina para aprender, mediante entrenamiento, los parámetros que modelan el comportamiento normal del sistema. Las redes neuronales son la herramienta de aprendizaje más utilizada en los IDS. De esta manera, podemos encontrar su aplicación a la detección de anomalías en programas a partir de las llamadas al sistema (Han *et al.*, 2004), en protocolos de red basándose en distintas variables de sesión (Zanero y Savaresi, 2004), (Ramadas *et al.*, 2003) y (Mora *et al.*,

2006), y en la capa de aplicación de los paquetes a partir de palabras clave (Lippmann y Cunningham, 2000). Otro enfoque combina redes neuronales en el diseño de un IDS que detecta tanto abusos como anomalías (Zhang *et al.*, 2005).

Otras técnicas de aprendizaje usadas en la detección de anomalías son las máquinas de soporte vectorial (Chen *et al.*, 2005) y (Sung y Mukkamala, 2003), los modelos de Markov ocultos (Liu y Qiao, 2006) y los autómatas finitos deterministas (Ingham *et al.*, 2007).

Los IDS de abusos tienen como ventaja principal una gran capacidad de detección, manteniendo un índice bajo de falsos positivos. Su principal problema es que son incapaces de detectar ataques nuevos e, incluso, pequeñas variaciones de ataques bien conocidos. Por el contrario, los IDS basados en anomalías poseen como ventaja principal la capacidad para detectar ataques nuevos para los cuales no se tiene ninguna información previa. Su principal problema es el alto índice de falsos positivos que presentan estos sistemas.

Una tercera clasificación agrupa los IDS en función de la respuesta que llevan a cabo después de detectar una intrusión (Northcutt *et al.*, 2001):

- *Pasivos*. Son los sistemas que únicamente lanzan una alerta informativa cuando se detecta un ataque, será el administrador quien tome las medidas adecuadas.
- *Activos*. Aquellos sistemas que después de detectar una intrusión ejecutan acciones proactivas. Por ejemplo, desconectar usuarios, cerrar conexiones de red o insertar una nueva regla en el firewall.

Los sistemas pasivos presentan como desventaja la existencia de un retraso en el tratamiento de la intrusión (Debar *et al.*, 2007). Sin embargo, las contramedidas automáticas pueden causar problemas de denegación de servicio del propio sistema y de usuarios legítimos (Wang *et al.*, 2007).

Finalmente, aunque en menor medida que las anteriores, los IDS se pueden clasificar atendiendo a la frecuencia de uso del sistema (Kruegel *et al.*, 2005):

- *Dinámicos*. Aquéllos que analizan continuamente la actividad de los sistemas que protegen. Los datos son analizados tan pronto como se producen.
- *Estáticos*. Llevan a cabo el análisis cada cierto periodo de tiempo. En lugar de procesar de forma continua analizan el estado del sistema en un momento dado.

La ventaja de los dinámicos es que, al procesar la información inmediatamente, pueden llevar a cabo acciones de respuesta en el caso de detectar una intrusión. Sin embargo, la recogida y análisis permanente introduce una sobrecarga importante. La única ventaja de los sistemas estáticos es que no introducen sobrecarga en el sistema monitorizado, pero sólo son útiles para análisis forense.

La tabla 2.1 muestra los distintos tipos de sistemas de detección, junto con sus fortalezas y debilidades (Kruegel *et al.*, 2005).

Tabla 2.1. Clasificación y características de los IDS.

	Fortalezas						Debilidades			
	Ratio detección	Detección nuevos	Protección de red	Protección interna	Respuesta inmediata	Protección continua	Falsos positivos	Consumo recursos	Cifrado	Denegación legítima
NIDS			+					+	−	
HIDS				+				−	+	
IDS abusos	+	−					+			
IDS anomalías	=	+					−			
Pasivos					−					+
Activos					+					−
Dinámicos						+		−		
Estáticos						−		+		

Organización de los IDS

Con el incremento del tamaño y la heterogeneidad de las redes, ha crecido también la complejidad estructural de los IDS. Los sistemas de detección tradicionales están situados en cualquiera de los nodos o dispositivos de red y realizan la detección de manera autónoma. Desafortunadamente, son incapaces de aportar una visión global de la seguridad de la red o de detectar ataques que ocurran simultáneamente en varios nodos de la red (Ganame *et al.*, 2008). Para abordar las deficiencias de los sistemas tradicionales, se han desarrollado los sistemas de detección de intrusos distribuidos (DIDS —Distributed Intrusion Detection Systems), formados por IDS locales o sensores que cooperan con el resto en el proceso de detección, mediante el intercambio de información y adoptando algún tipo de organización (Huang *et al.*, 1999).

En cuanto al intercambio de información, unos enfoques utilizan mecanismos de aprendizaje de máquina para coordinar la cantidad de información a distribuir, optimizando el compromiso entre el tiempo necesario para detectar el ataque y el volumen de información a compartir (Peng *et al.*, 2007). Otros, abordan el problema mediante la definición de estados internos de los sensores locales y la utilización de un marco que permite gestionar, compartir y migrar estos estados internos (Colajanni *et al.*, 2007).

Desde el punto de vista de la organización, podemos encontrar arquitecturas centralizadas donde se tienen muchos nodos sensores distribuidos que recogen la información y la transfieren a un nodo central que realiza el análisis (Ullrich, 2004). En (Zhang *et al.*, 2005) se muestra un sistema que realiza dos procesos de clustering distintos: uno en los nodos distribuidos para elegir eventos anómalos candidatos y otro en un IDS central para determinar el verdadero ataque.

Debido a la carencia absoluta de escalabilidad en el enfoque anterior, surgen las arquitecturas jerárquicas donde se distribuyen nodos intermedios que se encargan de filtrar la

información irrelevante, con lo que el nodo raíz tendrá menos información a procesar (Neumann y Porras, 1999), (Spafford y Zamboni, 2000), (Xiaoping y Yu, 2004) y (Chu *et al.*, 2005).

Un ejemplo clásico de arquitectura jerárquica es AAFID (Agentes Autónomos para la Detección de Intrusos) (Spafford y Zamboni, 2000). La estructura del sistema se basa en tres componentes esenciales que son agentes autónomos (aunque no móviles): *agents*, *transceivers* y *monitors*. Los agentes recogen datos de determinados aspectos del computador e informan al transceiver apropiado. Puede haber varios agentes por computador pero un solo transceiver. Los transceivers, a su vez, informarán al monitor del que dependan. Puede haber varios transceivers dependiendo del mismo monitor. Así mismo, los monitores pueden estar organizados en una jerarquía de monitores. Sólo los transceivers y los monitores pueden generar una alarma, los primeros sobre el estado del computador que controlan y los segundos de la red que están monitorizando. La figura 2.1 muestra la arquitectura del sistema AAFID.

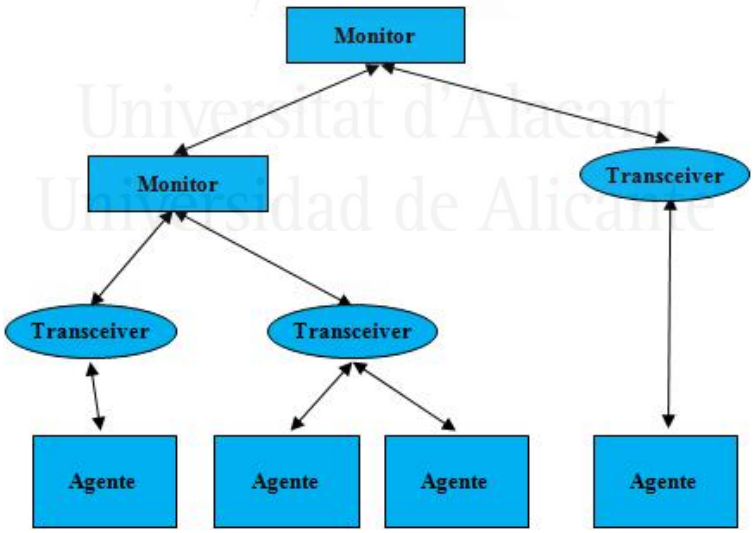


Figura 2.1. Arquitectura jerárquica de AAFID.

En (Xiaoping y Yu, 2004) se presenta una arquitectura distribuida jerárquica formada principalmente por tres

componentes: *centros de control, proxys, sensores*. Los sensores recogen la información realizan cierta reducción y pasan los datos relevantes a los centros de control a través de los proxys. Otro trabajo presenta un DIDS aplicable a redes de gran escala y basado en una arquitectura jerárquica con tres componentes: *un nodo raíz, varios nodos intermedios y muchos nodos hoja* (Chu *et al.*, 2005). En este caso los nodos hoja son IDS locales y no se limitan únicamente a realizar reducción de datos, sino que transfieren información de agregación hasta el nodo raíz.

Los enfoques centralizados o jerárquicos presentan dos problemas fundamentales (Hofmann *et al.*, 2007):

- *Escalado limitado*. Uno de los principales problemas de llevar a cabo el análisis y control del sistema de forma centralizada es que proporciona poca posibilidad de escalado. Procesar toda la información en un único computador implica limitar el tamaño de la red que puede monitorizar. De sobrepasar dicho límite, el analizador central es incapaz de procesar y mantener todo el flujo de información. Además, suele causar una sobrecarga del tráfico de información en la red, principalmente en los nodos cercanos al del analizador. Este problema se suele reducir, aunque no eliminar, estableciendo varios niveles en la jerarquía y realizando una reducción de datos en los niveles intermedios con técnicas de minería de datos.
- *Baja tolerancia a fallos*. Los dos tipos de sistemas presentan el problema de un único punto de fallo, o lo que es lo mismo, si falla el nodo central, dejará de funcionar todo el sistema. Si un atacante consigue suspender el analizador central (denegación de servicio, etc.), toda la red quedará sin protección. Éste es un problema grave, desde el punto de vista de la seguridad, que se suele mitigar introduciendo redundancia.

Para solucionar los problemas fundamentales de los enfoques anteriores es necesario usar técnicas de análisis de datos

descentralizadas o distribuidas (Kruegel y Toth, 2002) y (Yeggeswaran *et al.*, 2004).

Para llevar a cabo un análisis distribuido se están utilizando distintas tecnologías. Un enfoque sugiere que sólo los nodos donde tenga lugar realmente la intrusión colaboren o cooperen en su detección (Arora *et al.*, 2004). Otro enfoque plantea el uso de código móvil donde los agentes viajan por la red recogiendo información relevante del ataque y, por qué no, realizando ellos mismos el análisis (Kannadiga y Zulkernine, 2005). Finalmente, una perspectiva completamente descentralizada utiliza un esquema de cooperación peer-to-peer (Vlachos *et al.*, 2004).

Un DIDS descentralizado es el proyecto CARDS, un prototipo que usa árboles de ataque para representar secuencias predefinidas de pasos de ataques como firmas distribuidas. Descompone el ataque distribuido en unidades más pequeñas que corresponden con eventos distribuidos indicadores de ataques. Este sistema ejecuta y coordina la detección en el lugar donde se producen estos eventos (Ning *et al.*, 2002).

En (Kruegel *et al.*, 2005) se propone un sistema P2P que reconoce ataques de forma distribuida y presenta un lenguaje para expresar los ataques como una secuencia de pasos que relacionan computadores. El lenguaje tiene la ventaja de que sólo necesita información local para decidir que mensajes debe pasar y está diseñado para evitar el crecimiento exponencial de la información a compartir. Muchos de los DIDS basados en P2P sólo permiten el intercambio de información entre nodos vecinos, limitando la exactitud de las decisiones de detección, por lo que existen marcos de detección distribuida que recogen información de cualquier nodo de la red (Ye *et al.*, 2008). Estos sistemas son interesantes ya que confirman que el enfoque P2P para la detección de intrusos distribuida es factible y apropiado (Locasto *et al.*, 2005).

Finalmente, el uso de la tecnología de agentes en los DIDS también está muy extendido. Un enfoque denominado IDIAS (Intrusion Detection Intelligent Agent System) integra agentes

inteligentes de distintos tipos en el mismo entorno para proporcionar una estrategia de defensa en profundidad (Berquia y Nacsimento, 2004). En (Shyu *et al.*, 2007) se muestra un DIDS de dos niveles basándose en un Sistema Multi-Agente (MAS —Multi-Agent System). En un nivel los agentes detectan anomalías mientras en el otro nivel llevan a cabo detección de abusos. Por último, (Zhao-wen *et al.*, 2007) presentan un modelo cooperativo distribuido basado en agentes que implementa el proceso de detección mediante agentes específicos que intercambian mensajes de eventos entre dominios de detección lógicos.

La tabla 2.2 muestra las distintas organizaciones de los DIDS y sus características más importantes.

Tabla 2.2. Características y tipos de DIDS.

	Características				
	Escalabilidad	Tolerancia a fallos	Resistencia a subversión	Reconfiguración	Flexibilidad
Centralizados	−	−	−	=	−
Jerárquicos	=	=	=	=	=
Descentralizados	+	+	+	+	+

Correlación de Alertas

Hasta hace pocos años los Sistemas de Detección de Intrusos (IDS) únicamente procesaban información de auditoría local o tráfico de red monitorizado para detectar patrones específicos —es el caso de los basados en firmas o abusos—, o desviaciones sobre un modelo de comportamiento normal —es el caso de aquellos basados en anomalías—, pero siempre de forma aislada. Sin embargo, hay varias razones por las que las alertas de los IDS no deberían ser tratadas independientemente:

- Muchas de estas alertas forman parte de algún escenario de intrusión multi-estado único y el administrador de seguridad debería analizar el incidente completo en vez de las alertas individuales (Zhou *et al.*, 2007).
- Está generalmente aceptado que combinando varios detectores de intrusiones podemos proporcionar mejor rendimiento (Gu *et al.*, 2008).
- El enorme volumen de datos de seguridad desde diferentes dispositivos puede desbordar a los administradores de seguridad y evitar que lleven a cabo un análisis efectivo, e incluso, hacer el análisis imposible de abordar (Li *et al.*, 2007).
- Los IDS convencionales a menudo lanzan una gran cantidad de alertas, la mayoría de las cuales son redundantes, y falsos positivos. Por lo tanto, es difícil comprender el estado del sistema así como emprender las acciones apropiadas (Haibin y Jian, 2007).

Teniendo en cuenta lo anterior, es necesario desarrollar sistemas de correlación de alertas automáticos capaces de eliminar redundancias, aumentar el rendimiento, descubrir los escenarios de alto nivel y mostrar a los administradores información que permita un análisis efectivo del estado de seguridad de la red.

La correlación de alertas es un proceso que analiza las alertas producidas por uno o más IDS y proporciona una visión de alto nivel de los ataques intentados u ocurridos (Valeur *et al.*, 2004). Aunque la correlación se presenta a menudo como un único paso, el análisis se suele llevar a cabo mediante varios componentes. Desafortunadamente, existen muchos enfoques que se centran exclusivamente en algún componente. (Kruegel *et al.*, 2005) exponen un enfoque comprensivo de correlación de alertas en el que muestran el proceso como una secuencia de componentes, entre los que destacan la fusión de alertas, la reconstrucción de sesiones de ataque y la correlación de ataques de múltiples pasos.

Aunque existen trabajos en la literatura que se centran exclusivamente en la fusión de alertas, debemos distinguir entre la fusión y la correlación de alertas (Siaterlis y Maglaris, 2004). La fusión de alertas es una parte de la correlación y tiene el objetivo de combinar alertas que representan la detección independiente de la misma ocurrencia de ataque por distintos IDS (Valeur *et al.*, 2004). En (Siraj *et al.*, 2004) se propone un motor de decisión que fusiona información de diferentes sensores usando técnicas de inteligencia artificial como los mapas cognitivos difusos (Kosko, 1986). (Gu *et al.*, 2008) muestran cómo fusionar alertas mediante el test del ratio de probabilidad (Huelsenbeck y Rannala, 1997), obteniendo mejores resultados que otras técnicas existentes como la *votación por mayoría* o la *votación ponderada*.

Otros trabajos han abordado como objetivo no la correlación de alertas sino la reducción de la cantidad de alertas que se presentan al administrador (Julisch y Dacier, 2002). Así, en (Soleimani y Ghorbani, 2008) abordan el problema de manejar grandes cantidades de alertas mediante un proceso de múltiples capas que consigue, finalmente, identificar sólo las alertas críticas y reducir alrededor del 90% de las alertas. En (Lincoln *et al.*, 2004) se analiza el problema de la privacidad en los sistemas de correlación de alertas.

Independientemente de si los trabajos han abordado aspectos concretos de la correlación de alertas o del proceso completo, las distintas técnicas empleadas se pueden clasificar en tres grandes grupos en función de la cantidad de información previa que necesita cada método. En este sentido, los métodos van desde aquéllos en los que es necesario aportar todo el conocimiento del dominio, hasta los que no necesitan ninguna información previa para su funcionamiento.

Los métodos que más cantidad de información previa necesitan son los de especificación de escenarios: utilizan todo el conocimiento disponible de los escenarios con el fin de especificar patrones de escenarios completos. Funcionan de forma similar a los IDS de firmas donde el patrón a buscar es un escenario de

alto nivel, formado por varios ataques secuenciales, en lugar de un evento aislado. La figura 2.2 muestra un ejemplo de escenario de ataque con múltiples pasos (ataques), concretamente, un montaje ilegal de un sistema de archivos de red NFS.

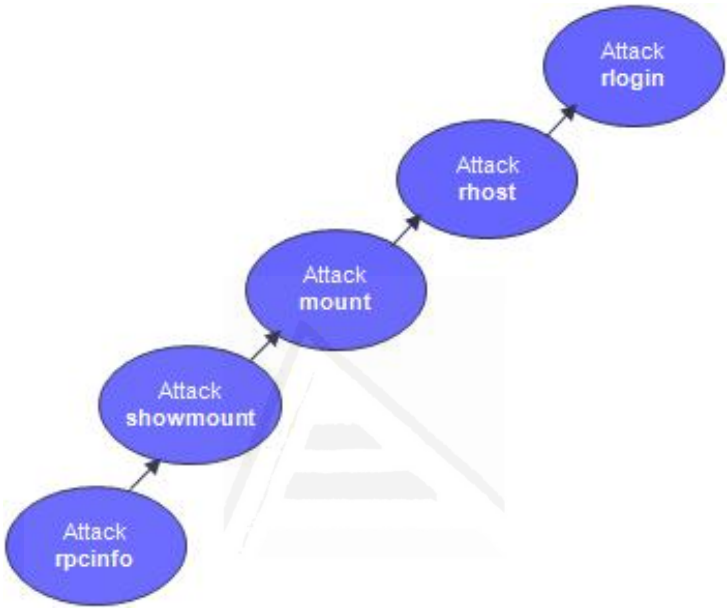


Figura 2.2. Ejemplo de escenarios de múltiples pasos.

La mayoría de trabajos que utilizan esta técnica lo hacen basándose en algún lenguaje de especificación que permita definir los patrones de búsqueda. Uno de los primeros lenguajes que permitía especificar ataques, que no escenarios, es LAMBDA (Cuppens y Ortalo, 2000). En este caso se utilizan cinco campos para la definición de un ataque y permite expresar tanto la combinación de eventos que el atacante genera en el ataque como aquellos eventos que es necesario detectar y verificar. Este lenguaje se utilizó como parte de la plataforma de IDS cooperativa MIRADOR (Cuppens y Miege, 2002).

Otro trabajo que se basa en un lenguaje para especificar escenarios lo podemos encontrar en (Morin y Debar, 2003), donde los autores proponen un componente de correlación de múltiples

alarmas basado en un marco denominado *Chronicles*. *Chronicles* proporciona un lenguaje declarativo de alto nivel y un sistema de reconocimiento que se utiliza en áreas en las que se monitorizan sistemas dinámicos.

STATL es un lenguaje extensible de descripción de ataques y escenarios basado en mecanismos de transición entre estados diseñado para soportar detección de intrusos (Eckmann *et al.*, 2002). El lenguaje permite describir los escenarios como secuencias de acciones o ataques que el atacante lleva a cabo para comprometer los sistemas. Tiene la ventaja de definir las características de los escenarios independientes del dominio y proporcionar construcciones para extender la descripción a entornos y dominios particulares, lo que permite especificar escenarios basados en host o en red y en sistemas como Windows, Linux y Solaris. También basado en estados, en (Cheung *et al.*, 2003) se propone el lenguaje de correlación de ataques CAML para modelar intrusiones de múltiples estados. Este enfoque usa predicados para abstraer los escenarios de ataque en términos de estados del sistema.

En (Debar y Wespi, 2001) se describe el algoritmo de agregación y correlación usado en el diseño e implementación de la consola de detección de intrusos Tivoli Enterprise Console de IBM. El algoritmo lleva a cabo la correlación basándose en un conjunto de reglas explicitas programadas previamente y se usa en la detección de *duplicados* (alertas lanzadas por distintos sensores pero que pertenecen al mismo ataque) y *consecuencias* (alertas relacionadas temporalmente). Este trabajo, aun perteneciendo al mismo grupo de técnicas, necesita menos codificación de información.

Para reducir la cantidad de información que es necesario codificar explícitamente, el segundo tipo de sistemas utiliza conocimiento de los ataques individuales o de tipos de ataques (Ning y Xu, 2003). Así, esta clase de técnicas establece los prerrequisitos y consecuencias de cada ataque o tipo con el objetivo de identificar relaciones en el proceso de correlación. La idea principal es que el éxito de un ataque posibilita la ejecución

de otro ataque posterior, por lo tanto, el mecanismo de correlación asocia consecuencias de un ataque anterior con prerrequisitos de otro posterior. El primer trabajo que utilizó esta idea fue (Templeton y Levitt, 2000). Su propuesta necesita que se cumplan todas las precondiciones de un ataque para considerar sus consecuencias. Aunque satisfacer todas las condiciones es teóricamente correcto, tiene un impacto negativo en la práctica al impedir continuar el análisis cuando existe relación entre los ataques si dicha relación no es completa.

En (Ning *et al.*, 2002) se presenta una extensión del trabajo anterior que aborda sus limitaciones y permite aplicar la técnica de correlación de forma práctica. El enfoque correlaciona *prerrequisitos* (condiciones necesarias para que una intrusión tenga éxito) y *consecuencias* (el beneficio de una intrusión cuando tiene éxito) de las intrusiones. El sistema permite la agregación de alertas y la satisfacción parcial de prerrequisitos, mejorando el rendimiento práctico del enfoque.

El trabajo anterior genera grafos de correlación a partir de los ataques detectados, sin embargo, suelen obtenerse grafos muy complejos y difíciles de entender. Para conseguir grafos más simples, (Al-mamory y Zhang, 2007) construyen grafos de ataques, pero definen reglas asociadas a clases de ataques en lugar de ataques individuales, lo que reduce la complejidad de los grafos resultantes y la cantidad de reglas a codificar. (Sheyner *et al.*, 2002) proponen algoritmos para generar y analizar automáticamente grafos de ataque de alto nivel, pero utilizan como entrada al sistema información de las vulnerabilidades en lugar de alertas de los IDS.

(Cuppens y Miege, 2002) proponen el mismo método pero con la nomenclatura *pre-condiciones* y *post-condiciones*. Utilizan el lenguaje Labmda que está basado en la lógica de predicados y correlacionan post-condiciones de un ataque anterior con pre-condiciones de otro posterior si tienen al menos un predicado común.

Siguiendo un enfoque similar a las nociones de consecuencias y post-condiciones, en (Zhou *et al.*, 2007) usan el concepto de *capacidad* para abstraer los niveles de acceso obtenidos por un atacante en cada uno de los pasos. Después, definen las relaciones lógicas entre las distintas capacidades.

Finalmente, dentro de este tipo de técnicas, (Cheng *et al.*, 2007) proponen un sistema para generar una base de datos de escenarios de ataque completa, a partir del emparejamiento de pre y post-condiciones.

Los trabajos comentados, a pesar de presentar resultados favorables en cuanto a capacidades de detección y reducción de falsos positivos, tienen limitaciones como el tiempo necesario para codificar los escenarios y su incapacidad para detectar ataques nuevos (Qin y Lee, 2004). Tales limitaciones se pueden abordar mediante técnicas que no tengan en cuenta en el proceso de correlación información a priori de los escenarios o de los ataques. Éste es el caso de los sistemas basados en similitud entre los atributos de las alertas o clustering. Estos sistemas buscan similitudes o relaciones entre los atributos de las alertas, de tal forma que aquéllas que tengan valores similares o relacionados en sus atributos pertenecerán al mismo grupo (Julisch, 2003). La idea es que las alertas agrupadas en el mismo grupo forman parte del mismo escenario.

En (Cuppens, 2001) se muestra un módulo de cooperación para la correlación de alertas basado en el clustering de alertas similares. En el algoritmo de clustering se utiliza un sistema experto en el que los requerimientos de similitud se especifican usando un conjunto de reglas. Las reglas permiten expresar la semejanza entre alertas a partir de los campos *Detecttime*, *Source*, *Target* y *Classification*.

(Dain y Cunninghan, 2001) proponen un enfoque probabilístico para agrupar en escenarios las alertas producidas por distintos sensores heterogéneos. Cuando se recibe una nueva alerta, se calcula la probabilidad de que pertenezca a cada uno de los escenarios existentes y se inserta en el que obtenga la mayor

probabilidad. Dicha probabilidad se calcula basándose en tres factores: fortaleza entre dos alertas, relación de tiempo entre ellas y correlación entre direcciones IP. Los resultados muestran que el sistema toma decisiones parecidas a las que tomaría un analista humano.

Otro trabajo que utiliza un enfoque probabilístico se puede encontrar en (Haibin y Jian, 2007). Los autores proponen un mecanismo de correlación de alertas basándose en la *teoría de la evidencia de Dempster-Shafer* (Dempster, 1967), que considera las alertas como evidencias de ataques y combina todas las evidencias de acuerdo a la regla de combinación de Dempster. El sistema tiene en cuenta el grado de exactitud de los IDS en la detección de los distintos tipos de ataque.

Hay métodos sofisticados de clustering que utilizan un enfoque probabilístico a la hora de agrupar las alertas (Valdes y Skinner, 2001). El algoritmo usado utiliza una función de similitud para cada característica de las alertas y un valor de similitud global obtenido como media ponderada de las semejanzas de los atributos. En los cálculos se utilizan los conceptos de valor esperado y mínimo; el esperado expresa la esperanza de que las características se emparejen si las alertas están relacionadas, mientras que el mínimo define el umbral por debajo del cual no se pueden agrupar dos alertas. Basándose en los conceptos anteriores se define una función que devuelve el valor o la probabilidad de que dos alertas, o una nueva alerta y una hiper-alerta existente, pertenezcan al mismo cluster y puedan fusionarse. Como en el trabajo anterior, las características principales sobre las que se fundamenta el análisis son: el origen del ataque, el destino, la clase y la información de tiempo.

Algunos trabajos aplican las técnicas de clustering en aspectos parciales o para la detección de tipos de ataque particulares. Éste es el caso de Spice (Staniford *et al.*, 2002), un motor de correlación de intrusiones diseñado para detectar escaneos de puertos furtivos. En esta propuesta se ha utilizado *recorrido simulado* como mecanismo de agrupación de paquetes anómalos que forman parte del mismo ataque.

(Porras *et al.*, 2002) muestran un algoritmo de clustering que agrupa las alertas basándose en la proximidad temporal. La intención del artículo es proporcionar herramientas automáticas para reducir el tiempo y coste de gestión de múltiples dispositivos de seguridad. La implementación del prototipo del sistema correlaciona alertas de seguridad de diferentes dispositivos heterogéneos distribuidos espacialmente: IDS, firewalls, servicios de autenticación y software antivirus. La estrategia seguida en este trabajo se basa en el análisis de la topología y la priorización y, principalmente, en la agregación de alertas basadas en atributos comunes.

En (Qin y Lee, 2003) se muestra el uso de técnicas de clustering para procesar alertas de bajo nivel formando alertas agregadas de alto nivel y se lleva a cabo un análisis causal basado en un test estadístico para descubrir nuevas relaciones entre los ataques. Utiliza el *test de causalidad de Granger* (Granger, 1969) para obtener relaciones entre alertas que deben pertenecer al mismo grupo o escenario. Este test se basa en el uso de series temporales. Como sus autores indican en el artículo, el método tiene un ratio de falsos positivos relativamente alto, inconveniente que presentan todos los trabajos cuya correlación está basada en clustering o similitud entre los atributos de las alertas. (Maggi y Zanero, 2007) demuestran que el trabajo anterior depende fuertemente de una buena elección de los parámetros del test, parámetros que son difíciles de estimar. Además, proponen un enfoque distinto basándose en un conjunto de test estadísticos más simples que no requieren parámetros de configuración complejos.

La minería de datos también se ha utilizado como mecanismo de correlación para reconocer estrategias de ataque de alto nivel (Li *et al.*, 2007). El sistema se basa en establecer un orden temporal de todas las alertas de la base de datos y usa el concepto de *correlatividad* entre los atributos para buscar las alertas que pueden ser agrupadas en el mismo escenario.

Los métodos de clustering presentan la ventaja de detectar escenarios nuevos por no basarse en conocimiento previo, pero

presentan un índice elevado de falsos positivos. Con el objetivo de conseguir sistemas de correlación que puedan mejorar la exactitud del análisis de escenarios, pero con un índice de falsas alarmas manejable, se han propuesto en la literatura enfoques que integran de manera ad hoc dos de las técnicas comentadas. Así, en (Qin y Lee, 2003) ya se comenta la necesidad de integración de algoritmos de correlación existentes.

Uno de los primeros trabajos en los que se empleó la técnica de prerrequisitos y consecuencias fue (Ning *et al.*, 2002). Posteriormente, en (Ning *et al.*, 2004) diseñan un sistema que utiliza el enfoque de integración de dos técnicas complementarias: aquéllas basadas en prerrequisitos y consecuencias de los ataques y las basadas en clustering. El objetivo es mejorar el rendimiento de los sistemas de correlación y reducir el impacto de ataques perdidos, para lo cual correlacionan escenarios de ataques aislados, obtenidos mediante la técnica de prerrequisitos y consecuencias, utilizando la información proporcionada por el clustering.

El primer trabajo obtenía, para distintos conjuntos de datos de prueba, unos ratios de detección del 57% y una tasa de falsos positivos del 19%; resultados mediocres en ambos casos. El segundo mejora el rendimiento al permitir integrar grafos de correlación aislados y ataques perdidos por los IDS locales, sin embargo, no presenta una evaluación cuantitativa de las capacidades del sistema.

Basándose en el enfoque de integración, (Qin y Lee, 2004) presentan un sistema que utiliza dos mecanismos de correlación complementarios basados en dos hipótesis sobre las relaciones entre los pasos de los ataques. La primera es que los distintos pasos de un ataque están relacionados directamente cuando un ataque anterior posibilita otro posterior. En este caso usa un motor de correlación bayesiano basado en la técnica de precondiciones y consecuencias. La segunda hipótesis es que para algunos pasos de ataque, aunque no estén relacionados directamente, existe una similitud temporal y estadística. Para este supuesto utilizan el enfoque estadístico analizado en (Qin y

Lee, 2003). En el proceso de integración, primero aplican la red bayesiana a las alertas de entrada y después emplean el *test de causalidad de Granger* para integrar grafos de correlación aislados.

La integración o incorporación de dos motores de correlación mejora los resultados de (Qin y Lee, 2003) tanto en capacidades de detección como en reducción de falsos positivos. No obstante, aunque reduce del 13% al 7% el ratio de falsos positivos, continua siendo un índice de error claramente insuficiente en entornos reales de producción.

Analizando detenidamente el proceso de integración en los dos trabajos anteriores se observa que no se concede la misma importancia a cada uno de los métodos de correlación. En (Ning *et al.*, 2004), primero se aplican paralelamente las dos técnicas a las alertas de entrada, después, partiendo de los grafos de ataque aislados obtenidos por el método de prerrequisitos y consecuencias, se utiliza la información de la técnica de clustering como apoyo para la integración, pero únicamente integra los subgrafos obtenidos con el método basado en información del dominio. Es más, después de integrar realiza hipótesis sobre posibles ataques perdidos buscando una correlación directa, es decir, que exista en la base de datos de ataques un camino que permita inferir una relación causa-efecto sobre el ataque perdido.

En (Qin y Lee, 2004) se integran los métodos de prerrequisitos y consecuencias con técnicas estadísticas de la siguiente forma: primero se aplica el motor de correlación bayesiano (prerrequisitos y consecuencias) a las alertas, el resultado de este paso es un conjunto de grafos de correlación aislados; después se utiliza un test estadístico para descubrir más relaciones entre las alertas y enlazar los grafos aislados de la etapa anterior.

Como se puede observar, en los dos casos se concede más importancia al método que utiliza información de los ataques, usando la técnica de similitud entre atributos o clustering como información de apoyo o complemento en el proceso de

integración, pero integrando únicamente los grafos obtenidos con el primer método. La idea es análoga a la detección de intrusos, donde el analista de seguridad generalmente aplica primero detección de abusos o patrones y, después, complementa con detección de anomalías para cubrir el espacio de ataques nuevos que el método de patrones no puede descubrir (Qin y Lee, 2004).

Conclusiones

Analizando la principal clasificación de los IDS, observamos que los sistemas de abusos tienen como ventaja principal su gran capacidad de detección, manteniendo un índice bajo de falsos positivos. Su principal problema es que son incapaces de detectar ataques nuevos e, incluso, pequeñas variaciones de ataques bien conocidos. Por el contrario, los IDS basados en anomalías poseen como ventaja principal la capacidad para detectar ataques nuevos para los cuales no se tiene ninguna información previa. Su principal problema radica en el alto índice de falsos positivos que presentan estos sistemas. Atendiendo a las ventajas e inconvenientes de cada tipo y dado que no son excluyentes, es recomendable el uso de varios IDS de distintos tipos complementarios en pos de obtener mejores rendimientos combinando sus resultados.

El estudio de la evolución de las distintas organizaciones de los sistemas de detección distribuidos nos ha permitido comprender las limitaciones de los sistemas centralizados y jerárquicos, y las ventajas de escalabilidad y tolerancia a fallos de los sistemas completamente distribuidos.

El examen de los distintos trabajos sobre el proceso de correlación de alertas muestra que todos los métodos propuestos tienen sus propias ventajas e inconvenientes. Sin embargo, hasta el momento no existe ninguna técnica que permita aportar una solución con los criterios de rendimiento que un sistema real requeriría:

- Los métodos de especificación de escenarios obtienen ratios de detección de escenarios conocidos elevados con índices mínimos de falsos positivos pero carecen de la capacidad de detectar situaciones nuevas. Además, necesitan de la codificación previa de toda la información relativa a los escenarios, hecho que consume mucho tiempo y puede dar lugar a errores.
- Los trabajos que utilizan el enfoque de prerrequisitos y consecuencias necesitan codificar menos cantidad de información que el caso anterior, pueden detectar pequeñas variaciones de escenarios conocidos, pero introducen el problema de los falsos positivos al obtener ratios medios.
- Los métodos que trabajan buscando similitudes entre los atributos de las alertas o técnicas de clustering tienen la ventaja de detectar escenarios nuevos y no necesitar la codificación de ninguna información previa, pero el gran inconveniente de presentar ratios de falsos positivos muy elevados.
- Los enfoques que integran dos métodos de manera ad hoc tienen mejor rendimiento que aquéllos que únicamente usan una técnica. No obstante, los resultados continúan siendo claramente insuficientes, pero muestran que la vía de la integración de métodos complementarios es una de las mejores opciones para seguir trabajando con el objetivo de conseguir rendimientos óptimos.

El análisis en profundidad de los trabajos de integración indica que se concede distinta importancia a los diferentes métodos de correlación. Esta observación constituye uno de los pilares en los que se basa nuestra propuesta: si somos capaces de medir el rendimiento o aportación de cada método, podríamos establecer una ordenación que permitiera obtener los mejores resultados de la aplicación simultánea de múltiples técnicas de correlación.

Capítulo 3

Modelo General de Detección

Los sistemas de detección siguen modelos basados en aspectos concretos de la detección de intrusos o se centran en determinados tipos de IDS. De esta manera, es sencillo encontrar trabajos que aborden, por ejemplo, sólo la seguridad de las aplicaciones Web (McAllister *et al.*, 2008) o que únicamente lleven a cabo análisis de anomalías (Ashfaq *et al.*, 2008), pero no se suelen encontrar sistemas que utilicen simultáneamente sensores de firmas y de anomalías.

De igual manera, es muy sencillo encontrar sistemas de correlación que utilizan sólo un método de análisis de las relaciones entre las alertas, pero hay muy pocos trabajos que utilicen enfoques de integración de técnicas complementarias, y los que existen usan sólo dos métodos y realizan la integración de manera ad hoc (Ning *et al.*, 2004).

Si los sistemas de detección sólo utilizan sensores de alguno de los distintos tipos existentes, no serán capaces de detectar todo el espacio de ataques (Qin y Lee, 2004). Además, si los enfoques de correlación usan únicamente un método, no podrán cubrir todo el abanico de escenarios, ni averiguar todas las relaciones existentes entre las alertas.

Los sistemas actuales deben tener la capacidad de detectar todo el espacio de ataques (Northcutt y Novak, 2003), la habilidad de utilizar varios métodos de correlación complementarios para ser más efectivos (Ning *et al.*, 2002) e, incluso, la posibilidad de integrar información de seguridad proveniente de herramientas distintas a los IDS (Qin y Lee, 2003). En definitiva, no deben estar centrados en aspectos concretos y tienen que ser más generales, permitiendo el uso simultáneo tanto de distintos tipos de sensores como de diferentes mecanismos de correlación de alertas.

Descripción General de la Propuesta

Si bien el presente apartado está dedicado a la descripción del modelo, es necesario considerar previamente ciertas características que presentan los sistemas de detección de intrusos, éstas peculiaridades permitirán obtener una visión global y una contextualización de la propuesta.

Existen muchos problemas donde a la hora de modelarlos únicamente es necesario tener en cuenta dos elementos fundamentales: el entorno y el sistema que aportará la solución. Sin embargo, en los sistemas de detección debemos considerar un elemento adicional formado por los sistemas maliciosos que provocan o son la causa de los problemas que debemos solucionar. Éstos sistemas atacantes, aunque no se van a modelar en la propuesta, conviene tenerlos presentes para contextualizar la problemática global. En nuestro caso, el entorno estará formado por la red, incluyendo tanto los dispositivos de red como los computadores que la integran y el tráfico que generan.

Aunque se tenga una buena política de actualizaciones, las redes son vulnerables debido principalmente a los siguientes factores: errores en la implementación del software y deficiencias en la configuración de los servicios. Las vulnerabilidades son la causa principal de los problemas de seguridad asociados con las redes y constituyen el punto de entrada de los atacantes.

Tanto los sistemas de detección como los atacantes intentan cumplir sus objetivos ejecutando acciones sobre la red. No obstante, los objetivos y las acciones de unos y otros son claramente contrarios. Los atacantes intentan aprovechar las vulnerabilidades de las redes para llevar a cabo ataques en pos de conseguir intrusiones. Por el contrario, los IDS ejecutan acciones de defensa para detectar y responder a las intrusiones. Ataque contra defensa e intrusión frente a detección constituyen la dinámica general del problema. La figura 3.1 muestra una visión global de la problemática de la detección de intrusiones.

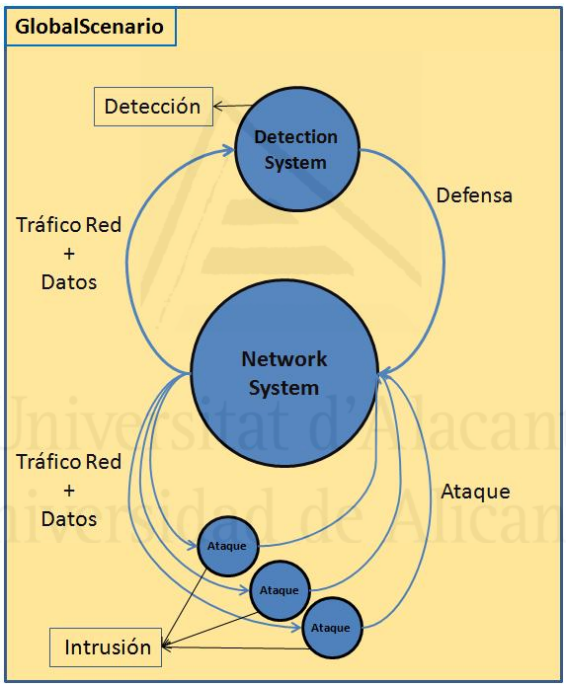


Figura 3.1. Contexto del problema de detección de intrusiones que representa el escenario global.

Después de mostrar la visión global del contexto en el que se desenvuelven los sistemas de detección, vamos a analizar con mayor nivel de detalle la descripción general del modelo de detección propuesto en esta tesis. Como se puede observar en la figura 3.2, el modelo está formado por la red y el sistema de

detección, donde el sistema se ha dividido en cuatro etapas claramente diferenciadas: percepción, correlación, integración y respuesta.

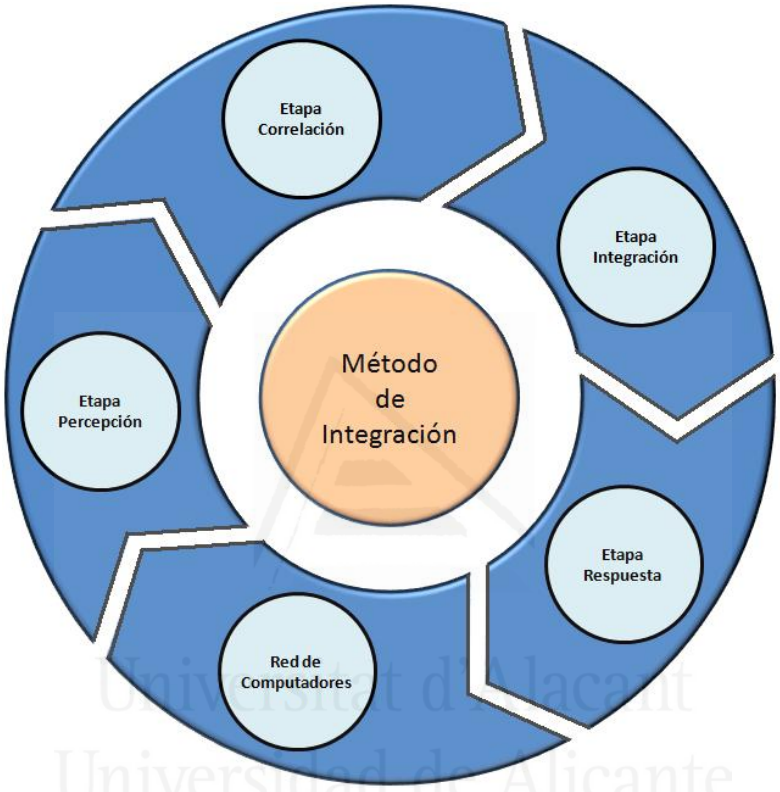


Figura 3.2. Modelo general del Sistema de Detección (DIDS).

La red de comunicaciones constituye el elemento a proteger y está formada por todos aquellos recursos susceptibles de ser atacados, desde dispositivos de red como *routers* y *switches* hasta computadores y servidores, pasando por cualquier otro recurso de red como puedan ser los dispositivos embebidos. La red estará permanentemente monitorizada para obtener toda la información posible, a partir de la cual se llevará a cabo el posterior análisis y detección. En el caso de detectar actividad maliciosa, el sistema podrá actuar sobre la red con el objetivo de mejorar la configuración de seguridad de ésta.

Inicialmente, se considerará al sistema como una entidad que está permanentemente monitorizando la red, analizando el estado de seguridad de ésta y ejecutando las acciones de respuesta necesarias, en el caso de que se encuentre en un estado inseguro. Mediante las acciones de respuesta, el sistema reconfigura y realimenta la red. Esta realimentación está basada en la experiencia y el conocimiento adquirido por el sistema. Ésta es la razón por la cual en la figura 3.2 la red se muestra como si de una etapa más del sistema se tratara.

Etapa de Percepción

La etapa de percepción permite reconocer todos los posibles estados del entorno que se puedan considerar representativos de un ataque. Mediante la monitorización de todos los recursos de la red se obtiene la información de entrada al sistema, sobre la que se puede realizar un análisis desde el punto de vista de la seguridad.

Como se ha observado en el capítulo anterior, se pueden tener tres fuentes de datos principales (Kruegel *et al.*, 2005): los paquetes de red, los host y las aplicaciones. Cuando la entrada procede de la red, los datos a tener en cuenta serán todos los que forman el paquete de red, desde los campos de las cabeceras para realizar un análisis de protocolos hasta el contenido para llevar a cabo detección a nivel de aplicación. Las entradas procedentes del host son todos aquellos datos que permiten obtener una visión de su estado en un momento dado. Ejemplos de datos de host sobre los que basar el análisis son los registros de auditoría del sistema como *syslog* o información de acceso, modificación y borrado de archivos del sistema críticos. Finalmente, cuando los datos de entrada provienen de las aplicaciones, éstos permiten el seguimiento o monitorización de su ejecución. Lo más común es utilizar *logs* de las propias aplicaciones pero, en ocasiones, se pueden usar las llamadas al sistema que realizan e, incluso, sus argumentos.

Para detectar los eventos que se consideran ataques de bajo nivel, la etapa de percepción utiliza IDS que realizarán el análisis

basándose en la información obtenida de la monitorización de los elementos de la red.

Dada la generalidad del modelo, la etapa de percepción, que modela la detección de bajo nivel, no se limita al uso de un IDS concreto, todo lo contrario, expresa la utilización simultanea de muchos IDS. Incluso, con el objetivo de alcanzar mayor nivel de detección global, los IDS deberían tener mucha variabilidad de tipos, tanto desde el punto de vista de la información de entrada como desde los mecanismos de análisis.

Utilizando como entrada los paquetes de red, los NIDS de firmas pueden abordar la detección de patrones de ataques de red perfectamente conocidos obteniendo óptimas capacidades de detección (Northcutt el al, 2001). Por otra parte, los NIDS de anomalías permiten detectar, estableciendo modelos de comportamiento normal de protocolos, ataques nuevos contra protocolos de red (Zanero y Savaresi, 2004). Por lo tanto, usando los dos tipos de NIDS se cubre potencialmente todos los posibles ataques detectables de red.

Sin embargo, existen ciertos inconvenientes a la hora de emplear NIDS: el cifrado de las comunicaciones, el ensamblaje de paquetes y las redes de alta velocidad, entre otros. Este tipo de sistemas se suele desplegar en los dispositivos de red, tales como los routers. El primer problema radica en que si se utiliza cifrado, el paquete de red no se descifrará hasta el nodo destinatario de la comunicación, limitando enormemente el análisis de los NIDS. La segunda desventaja es que realizar ensamblaje de paquetes fragmentados en el NIDS tiene un coste muy grande, pero si no se ensambla se corre el riesgo de recibir ataques de inserción y evasión que eluden la detección (Ptacek *et al.*, 1998). Por último, el tercer inconveniente está relacionado con la escalabilidad, dificultad de realizar todo el análisis necesario de los paquetes en redes sobrecargadas o de alta velocidad.

Además, los NIDS tienen grandes capacidades para detectar ataques externos a la red pero, en general, carecen de habilidades para detectar ataques internos, por lo que la utilización

simultanea de NIDS y HIDS proporciona beneficios importantes (Proctor, 2001).

Así, empleando como entrada información de host, ya sea del propio sistema o de las aplicaciones, los HIDS de firmas detectan con gran exactitud patrones de ataques que explotan vulnerabilidades tanto de los sistemas operativos como de las aplicaciones que se ejecutan sobre ellos. Además, el uso de HIDS de anomalías permite detectar, estableciendo modelos de comportamiento normal de usuarios, servicios y aplicaciones, ataques desconocidos contra cualquiera de ellos (Kruegel *et al.*, 2005).

Etapa de Correlación

Como se ha comentado en el capítulo del estado del arte, hasta hace pocos años los IDS únicamente procesaban la información de entrada y emitían de forma aislada, en el caso de detectar algún ataque, la alerta correspondiente. Se pueden llevar a cabo diferentes respuestas pero, generalmente, la mayoría de sistemas de detección de intrusos son pasivos, limitándose a informar al administrador mediante la correspondiente alarma. La función de percepción definida en los párrafos anteriores permite modelar o expresar por completo este funcionamiento de los IDS.

No obstante, también se ha comentado en el estado del arte, existen razones por las que las alertas de los IDS no deberían ser tratadas independientemente: muchas de estas alertas aisladas forman parte de algún escenario de intrusión multi-estado único que debería ser analizado de forma completa (Zhou *et al.*, 2007); está aceptado que combinando varios IDS podemos proporcionar mejor rendimiento (Gu *et al.*, 2008); la enorme cantidad de alarmas desborda a los administradores y hace el análisis imposible de abordar (Li *et al.*, 2007) y; finalmente, la mayoría de las alertas aisladas que emiten distintos IDS son redundantes y deben ser reducidas para comprender el verdadero estado de seguridad de la red (Haibin y Jian, 2007). Todas estas razones muestran la necesidad de llevar a cabo un proceso de correlación de alertas que relacione las alertas aisladas eliminando

redundancias, detectando escenarios de alto nivel e incrementando el rendimiento.

La etapa de correlación del sistema permite llevar a cabo automáticamente el proceso de correlación de las alertas. Como se puede deducir, la información de entrada a la función de correlación estará formada por todas las alertas aisladas producidas por los distintos IDS de la etapa de percepción.

Como en el caso de la etapa de percepción, dado que también existen distintos métodos o mecanismos no excluyentes para realizar la correlación de alertas, la etapa de correlación del modelo estará formada potencialmente por el empleo simultáneo de todos los métodos de correlación de alertas. Será competencia de la etapa de integración aprovechar al máximo las sinergias de cada uno de ellos para obtener los mejores resultados.

Los métodos de *especificación de escenarios* codifican, mediante algún tipo de lenguaje, todo el conocimiento de los patrones, es decir, todos los eventos y ataques que se tienen que producir para considerar detectado un escenario multi-estado o de alto nivel determinado (Eckmann *et al.*, 2002). Este tipo de técnicas presenta como ventajas la obtención de altos índices de detección de escenarios conocidos y la no existencia del problema de los falsos positivos, pero tienen el inconveniente de necesitar gran cantidad de tiempo en la codificación de la base de datos de patrones y que es un proceso que puede producir errores.

Los métodos de *prerrequisitos y consecuencias* codifican menos información previa que los anteriores. En vez de describir el patrón entero del escenario de alto nivel, definen los prerrequisitos y las consecuencias de cada uno de los ataques aislados o de bajo nivel conocidos. Posteriormente, el algoritmo busca coincidencias entre las consecuencias de un ataque anterior y los prerrequisitos de otro ataque posterior, estableciendo la relación entre los dos ataques y construyendo, de esta manera, el escenario completo (Cuppens y Miege, 2002). Las ventajas de esta técnica es que necesita menos información previa para la ejecución del algoritmo y puede detectar pequeñas

variaciones de escenarios conocidos (Ning *et al.*, 2002), el inconveniente es que presenta el problema de los falsos positivos.

Finalmente, otra clase de métodos utiliza la *similitud entre los atributos de las alertas* para, sin necesidad de ningún tipo de información previa, agrupar aquellas alertas similares indicando que pertenecen al mismo escenario (Qin y Lee, 2003). Los mecanismos utilizados para agrupar las alertas pueden ser de diversa índole, desde relaciones estadísticas y probabilísticas hasta algoritmos de clustering. Las ventajas de estos métodos es que no necesitan información previa y que pueden detectar, potencialmente, escenarios completamente nuevos, el problema principal es que presentan índices de falsos positivos extremadamente elevados.

Cada uno de los métodos tiene sus propias ventajas e inconvenientes, pero nada impide la utilización simultanea de los distintos métodos, porque son totalmente complementarios; más bien todo lo contrario, se recomienda utilizar varios métodos para incrementar el rendimiento (Qin y Lee, 2003). La etapa de correlación permite el uso simultáneo de todos y cada uno de los métodos expuestos. Sin embargo, con el objetivo de conseguir un rendimiento óptimo, es responsabilidad de la etapa de integración llevar a cabo el proceso de manera que maximice las ventajas y sinergias de cada uno de ellos y minimice los inconvenientes.

Etapa de Integración

La etapa de integración del modelo define el proceso de integración de los resultados obtenidos por los distintos métodos de correlación de la etapa anterior. Los resultados obtenidos por cada método de correlación representan escenarios de alto nivel detectados pero, dado que ningún método es perfecto, es conveniente emplear varios métodos complementarios e integrar sus resultados (Ning *et al.*, 2002).

Como se ha analizado en el capítulo anterior, los enfoques existentes que utilizan simultáneamente varios métodos de correlación complementarios (Ning *et al.*, 2004) y (Qin y Lee,

2004), aquéllos que mejor rendimiento proporcionan, emplean únicamente dos métodos. Además, llevan a cabo la integración de los resultados de la correlación de manera ad hoc, impidiendo la integración generalizada de más métodos y el aprovechamiento de las sinergias de cada uno de ellos.

No obstante, el hecho de que los sistemas que utilizan integración ad hoc y aprovechan las ventajas de dos métodos de correlación sean los que consiguen mejores resultados, muestra que la integración es un enfoque apropiado para optimizar los resultados.

Por lo tanto, nuestra propuesta profundiza en esta idea e incorpora la etapa de integración, pero en lugar de integrar únicamente dos métodos, tiene la capacidad de integrar cualquier técnica de correlación. Además, realiza la integración utilizando una metodología basada en los conceptos de la teoría de la información (Shannon, 1948), con el objetivo de aprovechar al máximo las ventajas y minimizar los inconvenientes de cada método existente en el sistema. Concretamente, la propuesta de esta investigación es que la etapa de integración se apoye en la noción de *entropía* para caracterizar y medir la cantidad y calidad de la información que proporciona cada uno de los métodos de correlación. De esta manera, una medida basada en la entropía guiará el algoritmo de integración con el objetivo de que aquellas técnicas de correlación que más información aporten al resultado final sean las que predominen en el proceso de integración.

Etapa de Respuesta

Finalmente, la etapa de respuesta modela la capacidad del sistema de influir en el entorno llevando a cabo algún tipo de respuesta activa o simplemente informando de forma pasiva al administrador del sistema para que éste tome las acciones apropiadas. La experiencia y el conocimiento adquirido por el sistema son determinantes en la elección de las acciones de respuesta adecuadas que permiten aumentar el nivel de seguridad realimentando y reconfigurando la red.

El problema de utilizar únicamente respuesta pasiva, es decir, informar al administrador y que éste manualmente inspeccione las alertas y tome las acciones apropiadas, es que puede haber un retraso importante entre la detección de la intrusión y su respuesta (Kruegel *et al.*, 2005). Por el contrario, el uso de respuestas activas o automáticas eliminan el retraso anterior, pero introducen otros inconvenientes que abordamos a continuación.

La cantidad de acciones de respuesta activa que se pueden llevar a cabo es amplia y variada: modificar permisos de archivos, incluir nuevas reglas en los cortafuegos, detener procesos, terminar conexiones de red, desconectar usuarios sospechosos, deshabilitar cuentas de usuario e, incluso, detener máquinas sospechosas (Northcutt y Novak, 2003). No obstante, existe el mito de que se puede emplear una respuesta activa para detener a los intrusos antes de que el ataque completo ocurra pero, es importante observar que, las contramedidas automáticas presentan el inconveniente de que pueden ser usadas por los atacantes como mecanismos excelentes de denegación de servicio en la propia red que se pretende proteger (Proctor, 2001).

Modelado del Entorno

Antes de desarrollar el modelo de detección propuesto es conveniente definir todos los elementos que conformen el marco sobre el cual actuará. El objetivo es definir una base formal sobre la que poder especificar el modelo con el mayor rigor posible. Como se puede observar en la figura 3.2, se deben caracterizar tanto los componentes del entorno como los del sistema. Puesto que el entorno estará compuesto por la red de comunicaciones sobre la que se implante el sistema, en los apartados siguientes se utilizará indistintamente entorno o red.

La Red de Comunicaciones

La red de comunicaciones constituye el entorno en el que se desarrolla toda la actividad y se define basándose en dos conceptos principales: el conjunto de los distintos estados posibles en los que puede encontrarse y el conjunto de acciones que se pueden llevar a cabo sobre cualquiera de los estados anteriores.

Supongamos que es posible caracterizar el conjunto Σ de estados posibles de la red como:

$$\Sigma = \{\sigma_1, \sigma_2, \dots, \sigma_n\} \quad [3.1]$$

Utilizando notación algebraica, cada estado de la red puede definirse mediante la estructura:

$$\sigma = \langle D, R, F, C \rangle \quad [3.2]$$

Donde:

- D Conjunto de elementos que constituyen el dominio de la estructura.
- R Conjunto de relaciones definidas entre los elementos de la estructura.
- F Conjunto de funciones definidas en la estructura.
- C Conjunto de elementos de D diferenciados. En general, constantes.

Según la anterior definición se dice que un predicado con la forma $p(b_1,b_2,\dots,b_n)$, se satisface en un estado σ si y sólo si la estructura $\langle I(b_1), I(b_2), \dots, I(b_n) \rangle$ es un elemento de R. Donde $I()$ es la función de interpretación de la fórmula p.

$$\sigma \vDash p(b_1, b_2, \dots, b_n) \Leftrightarrow \langle I(b_1), I(b_2), \dots, I(b_n) \rangle \in R \quad [3.3]$$

Si se tiene en cuenta que el dominio y las constantes permanecen inalterados para todos los estados, cada estado de la red se puede definir mediante la estructura:

$$\sigma_i = \langle D, R_i, F_i, C \rangle \qquad [3.4]$$

Como se ha analizado al principio del capítulo, las acciones que realiza el sistema sobre el entorno se encuentran en situación de conflicto con las llevadas a cabo por los sistemas maliciosos o atacantes. Por lo tanto, no es posible asegurar que la ejecución de una acción vaya a tener el efecto perseguido. Este problema se puede subsanar considerando una acción como un modo de intentar influir en el entorno, modificándolo según el objetivo del sistema. Sin embargo, las consecuencias de dicha acción no tienen por qué verse reflejadas en la red acorde con sus intenciones; es decir, se debe separar las acciones que realizan los sistemas (defensa y atacante) del efecto que realmente producen sobre los estados del entorno.

Según lo anterior, para modelar las acciones y sus consecuencias sobre la red de comunicaciones utilizaremos el modelo denominado *la acción como respuesta a las influencias*, propuesto por (Ferber y Müller, 1996). El modelo considera que el efecto que produce la ejecución de una acción sobre un estado concreto del entorno representa una *influencia* sobre el nuevo posible estado y que será la combinación de todas las influencias aportadas por todas las acciones de todos los sistemas la que realmente provoque un cambio en el estado del entorno. Según esto, una acción es el resultado de las reacciones del mundo ante las influencias de los distintos sistemas.

Supongamos que existe un conjunto finito P con todas las posibles acciones que se pueden llevar a cabo en un determinado entorno.

$$P = \{\rho_1, \rho_2, \dots, \rho_n\} \qquad [3.5]$$

Cada subconjunto $\wp(P) \subset P$ se denomina *plan*. Puesto que un plan podría estar compuesto por una simple acción o por un conjunto

de las mismas, a partir de ahora se hará referencia a acción o plan según convenga. Igualmente, cada plan podría estar constituido por todas las tareas de *P*, por lo que ambos conjuntos son totalmente intercambiables en la formulación. Por motivos de generalidad, se utilizará el conjunto *P*.

Empleando *operadores STRIP* (Waldinger, 1977) para definir las acciones, cada una de las tareas que caracterizan las acciones que se pueden realizar sobre la red, puede describirse mediante la estructura:

$$p = \langle nombre, pre, post \rangle \qquad [3.6]$$

En la que:

nombre	Expresión con la forma $f(x_1, \dots, x_k)$, donde cada x_i son variables autorizadas para aparecer en las fórmulas *pre* y *post*.
pre y *post*	Conjuntos de fórmulas de la forma $g(a_1, \dots, a_k)$, donde g es un predicado n-ario y cada a_i son constantes o variables.

Este modelo diferencia y posibilita la descripción por separado de las acciones que encapsulan los objetivos deseados y el efecto real que producen en su entorno. Por lo tanto, añadimos la definición del conjunto de las posibles influencias o intentos de acción de los distintos sistemas sobre el estado actual del mundo.

$$\Gamma = \{\gamma_1, \gamma_2, \dots, \gamma_n\} \qquad [3.7]$$

La ejecución de una tarea $p \in P$ representa una acción sobre la red y provoca una modificación del estado de ésta, pero lo modelamos como una aplicación parcialmente definida, en la que el resultado no es un nuevo estado del entorno, sino una influencia $\gamma \in \Gamma$ sobre el mismo:

$$Exec : P \times \Sigma \rightarrow \Gamma \qquad [3.8]$$

Según esta aplicación, una tarea $p \in P$ se puede ejecutar en Σ si y sólo si la aplicación está definida para un cierto estado$\sigma \in \Sigma$ de la red. Este hecho lo expresamos formalmente mediante el predicado:

$$\gamma = Exec(p, \sigma) \qquad [3.9]$$

Donde la función $Exec$ actúa de la siguiente manera:

$$Exec(\langle nombre, pre, post\rangle, \sigma) = \begin{cases} post & si\ pre(\sigma) se\ verifica \\ \{\} & si\ no\ se\ verifica \end{cases} \qquad [3.10]$$

Puesto que puede haber simultaneidad de las acciones llevadas a cabo en un cierto estado del entorno por el sistema de detección y los sistemas atacantes, debemos extender la función $Exec$ para contemplar este hecho. Para ello, definimos el operador de simultaneidad denotado como $\|$. Dicho operador combina acciones simultáneas y reúne de forma sencilla sus influencias. Realizamos ahora la extensión mediante un morfismo del espacio de acciones equipado con el operador de simultaneidad $\|$, actuando sobre el conjunto de influencias Γ. Formalmente:

$$Exec : (P, \|)\ x\ \Sigma\ \rightarrow \Gamma \qquad [3.11]$$

Por ejemplo, dadas p_1 y $p_2 \in P$, dos tareas del espacio de acciones y $\sigma \in \Sigma$, un estado cualquiera del entorno:

$$Exec(p_1 \| p_2, \sigma) = Exec(p_1, \sigma) \cup Exec(p_2, \sigma) \qquad [3.12]$$

Donde vemos que la ejecución simultánea de ambas acciones es equivalente a la unión de las influencias provocada por su ejecución aislada.

Sin embargo, la finalidad que se persigue con la ejecución de las tareas es la transición de un estado del entorno a otro. En este caso, las acciones son el resultado de la combinación de las distintas aportaciones en influencias y dicha transformación se contempla como una reacción del entorno, es decir, de la red de

comunicaciones, ante éstas influencias, hecho que se describe mediante la función:

$$ModRed : \Sigma \; x \; \Gamma \; \rightarrow \Sigma \qquad [3.13]$$

Utilizando esta función, el resultado de la ejecución simultánea de n acciones se define de la forma:

$$\sigma_2 = ModRed(\sigma_1, Exec(p_1 \| \ldots \| p_n, \sigma_1)) \qquad [3.14]$$

Finalmente, utilizando las definiciones planteadas hasta el momento se puede describir el entorno a partir de las acciones mediante la estructura:

$$NetworkSystem = \langle \Sigma, P, \Gamma, Exec, ModRed \rangle \qquad [3.15]$$

Es decir, mediante los conjuntos de los estados posibles de la red, de las acciones que se pueden ejecutar sobre el entorno y de las influencias provocadas por dichas acciones, junto con las funciones de ejecución de estas acciones y de reacción del entorno ante las influencias.

Modelado del Sistema de Detección

Una vez establecido el marco formal para la red de comunicaciones basado en el modelo de influencia y reacción, definiremos el sistema de detección como la entidad capaz de provocar influencias. Recordemos que los sistemas atacantes, aunque también pueden llevar a cabo acciones, no se van a especificar en este trabajo y sólo se han tenido en cuenta para contextualizar la propuesta.

Elementos del Sistema de Detección

Si en el apartado anterior se han analizado los principales elementos que componen el entorno: sus estados y las acciones e influencias que se pueden realizar para cambiar dicho estado, en este apartado se presenta la base formal del sistema de detección

de intrusos, es decir, del componente que va a interactuar con el entorno ejecutando las distintas acciones, provocando las influencias y posibilitando el cambio de estado para alcanzar un determinado conjunto de estados que constituyen su objetivo.

Como se puede observar en la figura 3.2, el sistema se ha dividido en cuatro etapas claramente diferenciadas. A continuación se define detenidamente la base formal presente en cada una de ellas.

Etapa de Percepción

La etapa de percepción permite obtener y analizar el estado de seguridad de la red en todo momento mediante un conjunto de elementos independientes que facilitarán su implantación en un sistema distribuido que pueda ser más fácilmente escalable.

Cada elemento independiente es un IDS con capacidad para percibir y discernir sobre el estado de seguridad en el que se encuentra la red. La estructura y comportamiento de cada IDS dependerá de su tipo, principalmente, del mecanismo de análisis empleado.

Estos IDS detectan los eventos que se consideran ataques a bajo nivel y realizan el análisis basándose en los distintos estados del entorno comentados en el apartado anterior. Si después de realizar el proceso de análisis, la percepción es que la red se encuentra en un estado inseguro, el IDS lanzará una alarma alertando sobre éste hecho.

Las alertas lanzadas por los IDS estarán en un formato para el intercambio de información entre sistemas de detección denominado IDMEF (Intrusion Detection Message Exchange Format) (Debar *et al.*, 2007b), muy extendido en éste área. La figura 3.3 muestra un ejemplo de alerta en formato IDMEF. Como podemos observar en dicha figura, la información de la alerta suele incluir las direcciones origen y destino del ataque, la hora y el tipo de ataque (si es posible), entre otras informaciones.

```
<IDMEF-Message version="0.3">
<Alert alertid="1" impact="unknown" version="1">
<Time>
        <date>03/07/2000</date>
        <time>09:51:36</time>
        <sessionduration>00:00:00</sessionduration>
</Time>
<Analyzer ident="tcpdump_dmz">
        <name>tcpdump_dmz</name>
</Analyzer>
<Source spoofed="unknown">
        <Node>
                <Address category="ipv4-addr">
                        <address>202.77.162.213</address>
                </Address>
        </Node>
</Source>
<Target>
        <Node>
                <Address category="ipv4-addr">
                        <address>172.16.115.1</address>
                </Address>
        </Node>
        <Service>
                <name>icmp-echo-request</name>
        </Service>
</Target>
</Alert>
</IDMEF-Message>
```

Figura 3.3. Ejemplo de alerta en formato IDMEF.

Teniendo en cuenta lo anterior, definimos Φ como el conjunto de todas las posibles alarmas que se pueden activar por cualquier IDS.

$$\Phi = \{\varphi_1, \varphi_2, \dots, \varphi_n\} \qquad [3.16]$$

Una vez definido el conjunto de alertas, cualquier IDS se modela como una función especializada en la detección de intrusiones; especializada porque cada tipo de IDS está mejor preparado para la detección de determinadas clases de ataques. Por ejemplo, los HIDS detectan mejor ataques internos a la red, mientras los NIDS

presentan mejores capacidades en la detección de intrusiones externas. De esta manera, podemos definir F_{IDS} como el conjunto de todos los posibles IDS o, dicho de otra manera, como el conjunto de todas las posibles funciones especializadas en la detección a bajo nivel.

$$F_{IDS} = \{IDS_1, IDS_2, \dots, IDS_n\} \quad [3.17]$$

Donde, teniendo en cuenta las definiciones 3.1 y 3.16, cada IDS es una función que, partiendo de un estado de la red obtiene una alarma, indicando si este estado se corresponde con una intrusión.

$$IDS_i : \Sigma \rightarrow \Phi, \text{donde } IDS_i \in F_{IDS} \quad [3.18]$$

Por supuesto, un mismo evento de entrada puede provocar la activación de varias alarmas generadas, probablemente, por distintos IDS. Generalmente, si el ataque es externo puede ser detectado tanto por el sensor de red, al analizar los paquetes, como por el IDS de Host, cuando examina los estados del sistema. Por otra parte, distintos eventos de entrada pueden lanzar una única alarma en un único IDS; este es el caso, por ejemplo, de los escaneos de puertos o de los mapeos de red.

Dado que el sistema puede estar constituido por varios IDS ejecutándose simultáneamente, la etapa de percepción puede proporcionar un conjunto de alarmas ante un mismo estado de la red, indicando la percepción que tiene el sistema de detección de intrusos del estado de red. Este hecho se denota mediante la siguiente función:

$$Percept: \Sigma \rightarrow \Phi^n \quad [3.19]$$

La figura 3.4 muestra gráficamente la etapa de percepción, donde, en un momento dado, la salida estará formada por la unión o agrupación de las alarmas generadas por todos los IDS en un vector de alertas.

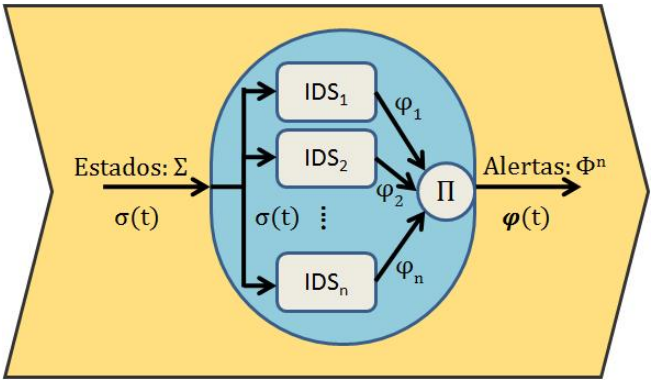


Figura 3.4. Etapa de percepción.

Etapa de Correlación

La etapa de correlación es la responsable de establecer las relaciones existentes entre las diferentes percepciones consideradas anómalas desde el punto de vista de la seguridad que hayan podido conseguir los elementos independientes en la fase anterior obteniendo, de esta forma, una visión mucho más global del estado del entorno.

Además, al igual que en la etapa de percepción, en esta fase es conveniente emplear múltiples componentes independientes que utilicen métodos de correlación distintos, lo que favorecerá su despliegue en entornos distribuidos mejorando la escalabilidad y aportará distintas vistas o perspectivas sobre el estado global del entorno.

Independientemente de la cantidad de elementos distintos que formen la etapa de correlación, cada uno de ellos recibirá como entrada todas y cada una de las alertas generadas por la etapa de percepción y devolverá como resultado *hiperalertas* donde, cada una de ellas está formada por una secuencia de alertas de bajo nivel relacionadas entre sí según el método de correlación concreto empleado por el elemento.

Definimos Z como el conjunto de todas las posibles hiperalertas que se pueden generar por cualquiera de los métodos de correlación.

$$\mathrm{Z} = \{\mu_1, \mu_2, \ldots, \mu_n\} \quad [3.20]$$

Teniendo en cuenta lo anterior, cualquier método se modela como una función especializada en la correlación de alertas generadas por los IDS de la etapa de percepción. De esta manera, se puede definir F_{CORREL} como el conjunto de todas las posibles funciones de correlación asociadas a cada uno de los métodos.

$$F_{CORREL} = \{Correl_1, Correl_2, \ldots, Correl_n\} \quad [3.21]$$

Donde, teniendo en cuenta las definiciones 3.16 y 3.20, cada método de correlación es una función que, partiendo de un conjunto de alertas aisladas de bajo nivel, obtiene una hiperalerta, indicando las alertas de bajo nivel que están relacionadas entre sí formando un escenario de alto nivel.

$$Correl_i : \Phi^n \rightarrow \mathrm{Z}, \text{ donde } Correl_i \in F_{CORREL} \quad [3.22]$$

Una misma alerta lanzada por cualquiera de los IDS de la etapa de percepción puede pertenecer, después de la función de correlación, a varias hiperalertas. Primero, porque la alerta puede formar parte de distintos escenarios y, segundo, porque puede ser correlacionada en el mismo escenario, pero por diferentes métodos de correlación.

Dado que pueden existir distintos métodos de correlación en el sistema simultáneamente, la salida de la etapa de correlación estará formada potencialmente por varias hiperalertas. Así, se define la función que describe la etapa de correlación como:

$$Correl: \Phi^n \rightarrow Z^m \quad [3.23]$$

La figura 3.5 muestra gráficamente la etapa general de correlación, donde, en un momento dado, la salida estará formada por la unión o agrupación de las hiperalertas generadas por todos los métodos de correlación complementarios en un vector de hiper-alertas $\boldsymbol{\mu}$.

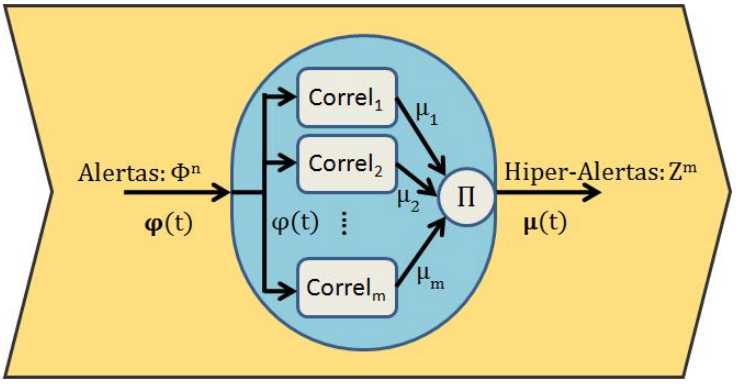


Figura 3.5. Etapa de correlación.

Etapa de Integración

La etapa de integración aprovechará los resultados obtenidos por los distintos métodos que pueden utilizarse en la etapa de correlación para proporcionar un diagnóstico más fiable sobre el estado de seguridad de la red de comunicaciones. En este sentido, integra las distintas vistas proporcionadas por la fase anterior, consiguiendo una visión global y más fiel del estado del entorno.

Teniendo en cuenta lo anterior, la etapa de integración del modelo define el proceso de integración de las hiperalertas generadas por los distintos métodos de la etapa de correlación. El capítulo siguiente abordará en profundidad el análisis y especificación de la estructura interna y comportamiento de la función de integración.

En este caso, a diferencia de las etapas anteriores, esta etapa está compuesta por una única función, denominada función de integración que define el proceso de integración de las hiperalertas generadas por los distintos métodos de la etapa de correlación. Produce como salidas nuevas hiperalertas representando escenarios completos detectados, donde cada escenario estará formado por un conjunto de hiperalertas obtenidas en la etapa de correlación. La figura 3.6 muestra las entradas y salidas de la etapa de integración.

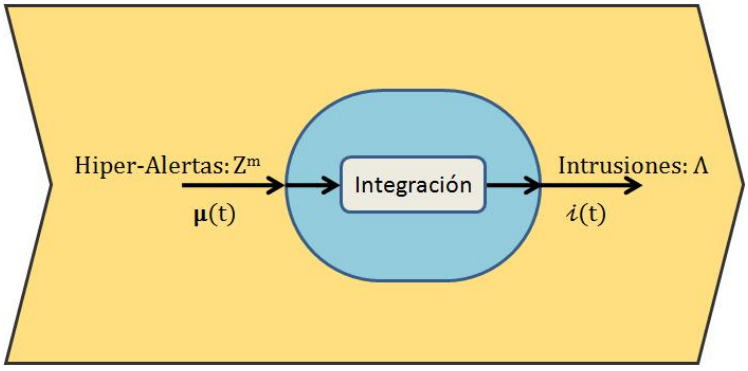


Figura 3.6. Etapa de integración.

Teniendo en cuenta lo indicado en los párrafos anteriores, podemos definir Λ como el conjunto de todas las posibles intrusiones o escenarios de alto nivel que es posible detectar.

$$\Lambda = \{i_1, i_2, \ldots, i_n\} \quad [3.24]$$

Donde la función que describe la etapa de integración se puede definir como:

$$Integ: Z^m \rightarrow \Lambda \quad [3.25]$$

En el presente apartado se ha realizado una breve descripción general de la etapa de integración. No obstante, el capítulo siguiente está íntegramente dedicado al análisis y especificación de la estructura interna y comportamiento de la función de integración, abordando aspectos como los distintos algoritmos de clustering para la integración, conceptos de teoría de la información como medida de la cantidad de información de cada método y proceso final de integración.

Etapa de Respuesta

Finalmente, la etapa de respuesta representa la capacidad del sistema para influir en el entorno mediante la ejecución de acciones que pueden cambiar el estado del mismo. Las respuestas a llevar a cabo son el resultado de la capacidad de

deliberación y toma de decisiones del sistema de detección y hacen uso de la experiencia y el conocimiento adquirido por éste con el objetivo de llevar al entorno a un estado seguro.

La etapa de respuesta recibe como información de entrada las intrusiones detectadas en la fase de integración y produce como salidas las respuestas adecuadas según el tipo de intrusión, además de un informe al administrador informando del estado de seguridad global de la red. La elección de las acciones dependerá de la intrusión de entrada o escenario detectado, de una función distancia y de un subconjunto de estados objetivo $\sigma^* \in \Sigma$.

$$Response: \Lambda \rightarrow P \qquad [3.26]$$

La figura 3.7 muestra gráficamente la función general de respuesta.

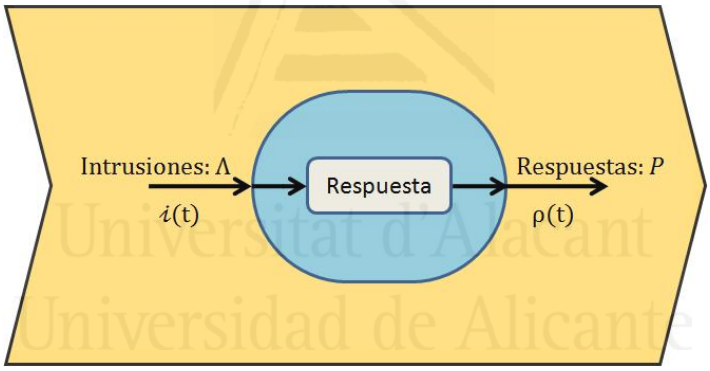


Figura 3.7. Etapa de respuesta.

Las respuestas serán la ejecución de las acciones que provocarán las influencias sobre el entorno modelado mediante la función *ModRed()*, posibilitando el cambio de estado. En cuanto a la capacidad de ejecución, ya ha sido definida anteriormente [3.8] y consiste en la producción de influencias que serán combinadas y que actuarán en el entorno de acuerdo con las leyes que lo caracterizan. Dicha ejecución puede ser llevada a cabo automáticamente por el sistema de detección o realizada por un administrador humano.

Finalmente, una vez definidos todos los elementos, se puede describir el sistema de detección mediante la estructura:

$$DetectionSystem = \langle \Phi, Z, \Lambda, Percept, Correl, Integ, Response \rangle \quad [3.27]$$

Donde:

- Φ Es el conjunto de todas las posibles alertas emitidas por los distintos IDS de la etapa de percepción, resultado de las percepciones que dichos elementos tienen sobre los estados del entorno.
- Z Es el conjunto de todas las posibles hiperalertas obtenidas por los distintos componentes en la etapa de correlación, representan relaciones existentes entre distintas alarmas o percepciones.
- Λ Es el conjunto de todas las posibles intrusiones detectadas en la etapa de integración, representan de manera muy fiable escenarios completos detectados.
- $Percept$ Función que define la capacidad del sistema para percibir y clasificar los estados el entorno.
- $Correl$ Función que aporta la habilidad de relacionar conjuntos de percepciones.
- $Integ$ Función que permite la integración de distintas vistas en una visión global y fiable del estado de seguridad del entorno.
- $Response$ Función que representa la capacidad de deliberación y toma de decisión del sistema en la selección de las acciones a ejecutar para influir en el entorno.

Todos ellos definidos en apartados anteriores.

Modelo de Detección

En el apartado anterior se han definido los conjuntos y funciones que componen el marco formal sobre el que describir el modelo

general de detección. Basándose en dicho marco formal, en este apartado se analiza formalmente la visión global del modelo de detección, identificando tanto los elementos que lo forman como el comportamiento y evolución a lo largo del tiempo.

La definición de las funciones básicas del marco formal se ha realizado basándose en los dominios de entrada y salida. En este apartado, partiendo de éstos dominios y, puesto que el comportamiento del sistema viene dado por la evolución de los estados de la red en función del tiempo, se van a definir las funciones del modelo basándose en un parámetro de tiempo *t*. La figura 3.8 muestra el conjunto de funciones del método.

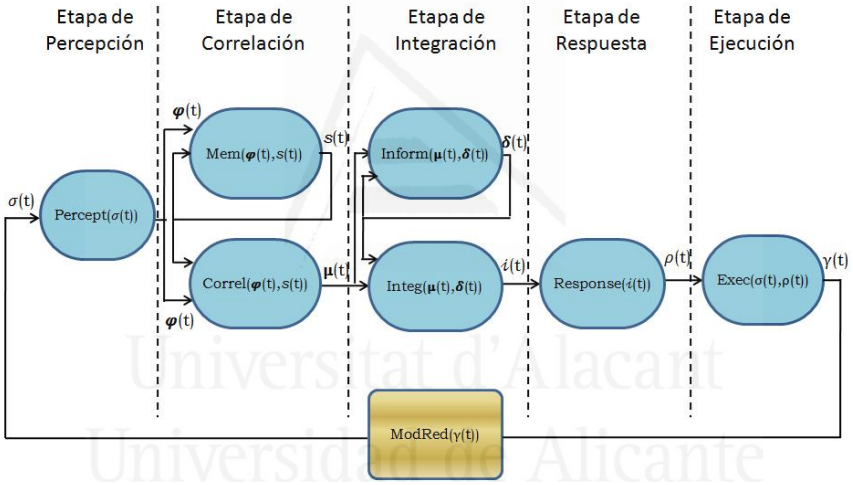


Figura 3.8. Etapas y funciones del modelo de detección.

La etapa de percepción está formada únicamente por la función *Percept* definida en el apartado anterior que, a partir de un estado de la red, es capaz de proporcionar la percepción que el sistema tiene del estado concreto. La percepción dota al sistema de la facultad de clasificar y distinguir entre los distintos estados posibles del entorno. Desde el punto de vista de los sistemas de detección, la percepción estará formada por los ataques que es posible detectar y, por lo tanto, por el conjunto de posibles alarmas.

Dado que se ha dotado al modelo de la posibilidad, incluso la conveniencia, de incorporar múltiples elementos de percepción, la función *Percept* es el producto de la actuación en paralelo de un conjunto de funciones especializadas en la detección de intrusiones aisladas IDS_i, pertenecientes al conjunto de funciones de detección F_{IDS} definidas en el apartado anterior. Según esto, podemos definir formalmente la percepción o función *percept* en un instante t como:

$$\boldsymbol{\varphi}(t) = Percept(\,\sigma(t)\,) = \prod_{\forall\, IDS_i \in F_{IDS}} IDS_i(\,\sigma(t)\,) \qquad [3.28]$$

Es decir

$$\boldsymbol{\varphi}(t) = (\varphi_1, \varphi_2 \ldots \varphi_x) \,/\, \forall \varphi_i \in \boldsymbol{\varphi}(t), \varphi_i = IDS_i(\,\sigma(t)\,) \qquad [3.29]$$

La detección de intrusiones aisladas por sí mismas no aporta información suficiente para determinar sin ambigüedad la actividad maliciosa de la red, por lo que es necesaria una etapa de correlación de alertas. Esta etapa de correlación debe relacionar unas alarmas con otras pertenecientes al mismo escenario de alto nivel, pero las alarmas a relacionar no tienen necesariamente que pertenecer al mismo instante de tiempo, sino que la mayoría seguramente pertenecerán a instantes anteriores. Por lo tanto, es necesario disponer de un histórico que facilite una panorámica más amplia de la actuación sobre el sistema y que permita una correlación posterior de estas intrusiones aisladas.

En definitiva el modelo debe disponer de un estado interno que le aporte la capacidad de adquirir conocimiento y, por lo tanto, desarrollar una función de correlación más rica y compleja basada en la experiencia. Para ello, definimos S como el conjunto de los estados internos del sistema:

$$S = \{s_1, s_2, \ldots, s_n\} \qquad [3.30]$$

Tras definir el conjunto de estados internos, la adquisición de experiencia se llevará a cabo mediante el paso de un estado

interno a otro. Para conseguir dicho conocimiento definimos la función de memorización *Mem* como una función que relaciona un estado interno del sistema con un conjunto de percepciones de la red en un determinado estado interno del sistema.

$$Mem: \Phi^n \, x \, S \rightarrow S \qquad [3.31]$$

Formalmente, la experiencia y el conocimiento del sistema en un instante t estará basada en la función de memorización y la función de percepción del instante inmediatamente anterior (*t-1*). Es decir

$$s(t+1) = Mem(\, \varphi(t), s(t)\,) \qquad [3.32]$$

A partir de la percepción actual del estado de la red, representada por una serie de alarmas, junto con la panorámica obtenida gracias a la capacidad de memorización del modelo, se intenta detectar mediante diferentes técnicas de correlación si existe alguna relación entre las distintas alertas individuales detectadas a lo largo del un periodo de tiempo determinado. Este proceso se lleva a cabo mediante la función *Correl* definida en el marco formal (ecuación 3.23), pero modificada para que tenga en cuenta la experiencia según:

$$Correl: \Phi^n x \, S \rightarrow Z^m \qquad [3.33]$$

El resultado de la etapa de correlación será un conjunto de hiperalertas que representan los distintos escenarios aislados de alto nivel obtenidos por los distintos métodos de correlación. Dado que los métodos emplean herramientas de análisis diferentes, es muy probable que sobre un escenario tengamos varias hiperalertas (una por cada método), por lo que el modelo poseerá la facultad de analizar los escenarios desde múltiples puntos de vista o perspectivas.

La función de correlación en un instante t estará basada en el conocimiento adquirido por el sistema y el conjunto de percepciones sobre entorno en ese mismo instante *t*; es decir, en

la función de memorización y la función de percepción de ese preciso instante. Formalmente:

$$\boldsymbol{\mu}(t) = Correl(\,\boldsymbol{\varphi}(t), s(t)\,) \,/\, \boldsymbol{\mu}(t) \in Z^m \qquad [3.34]$$

La función $Correl$ es el resultado de la conjunción de diferentes técnicas $Correl_i \in F_{CORREL}$. Por lo tanto, podemos definir la función de correlación como la unión de las diferentes técnicas $Correl_i$ de correlación. Es decir

$$\boldsymbol{\mu}(t) = Correl(\,\boldsymbol{\varphi}(t), s(t)\,) = \prod_{\forall\, Correl_i \in F_{CORREL}} Correl_i(\,\boldsymbol{\varphi}(t), s(t)) \qquad [3.35]$$

Una vez analizado el estado de la red mediante las diferentes técnicas de correlación, el sistema toma conciencia y obtiene una visión mucho más global del estado de seguridad de ésta e, incluso, posee la habilidad de observar y analizar dicho estado desde distintos puntos de vista. No obstante, con el objetivo de incrementar el rendimiento y la fiabilidad, la etapa de integración realiza un proceso de comparación entre los distintos resultados o vistas μ_i obtenidos, para relacionar escenarios aislados y obtener escenarios completos.

Para conseguir los objetivos de rendimiento, la función de integración debe maximizar la contribución al resultado final de las ventajas de los métodos y minimizar la aportación de los inconvenientes. Para conseguir dicho fin se utiliza una métrica basada en conceptos de la teoría de la información como *entropía*, de tal manera que, la cantidad de información que aporta cada método o la exactitud en los resultados de cada uno de ellos, influya o modifique el funcionamiento de la función de integración.

Definimos I como el conjunto de todas las posibles aportaciones de los métodos de correlación:

$$I = \{\delta_1, \delta_2, \ldots, \delta_n\} \qquad [3.36]$$

La aportación o cantidad de información que aporta cada método de correlación varía con el tiempo en función de los aciertos y

errores que comete, por lo que el sistema debe tener la capacidad de aprender de la experiencia con el objetivo de adquirir el conocimiento real de la aportación de cada método en un momento dado. Para ello, definimos la función de información o *inform* que relacionará la aportación de un método con su resultado partiendo de una aportación previa.

$$Inform: \mathrm{Z}\ x\ I \rightarrow\ I \qquad [3.37]$$

En un momento dado, la salida de la función *Inform* estará formada por la agrupación de las aportaciones de cada método de correlación en un vector de aportaciones $\boldsymbol{\delta}$. Formalmente, definimos la función de información en un instante t como:

$$\boldsymbol{\delta}(t) = Inform(\,\boldsymbol{\mu}(t), \boldsymbol{\delta}(t-1)\,) \qquad [3.38]$$

Después de definir la función de información que aportará al modelo la capacidad de conocer, basándose en su experiencia, la aportación real de cada método de correlación, podemos redefinir la función de integración expresada en el modelo formal (ecuación 3.25) como:

$$Integ: Z^m x\ I\ \rightarrow\ \Lambda \qquad [3.39]$$

Formalmente, expresamos que se ha detectado una intrusión en un instante dado t como:

$$i(t) = Integ(\,\boldsymbol{\mu}(t), Inform(\,\boldsymbol{\mu}(t), \boldsymbol{\delta}(t-1)\,)\,) \qquad [3.40]$$

Tanto los aspectos concretos de la metodología utilizada para condicionar la función de integración, como el propio mecanismo o proceso de integración serán analizados en profundidad en el siguiente capítulo.

La etapa de respuesta del modelo estará compuesta únicamente por la función *Response* definida en el marco formal. La función dota al modelo de la capacidad de deliberar y tomar decisiones

sobre la acción a llevar a cabo a partir del estado inseguro de la red o intrusión detectada.

Después de especificar todos los elementos, se puede redefinir el modelo (ecuación 3.27) mediante la estructura:

$$DetectionSystem = \quad [3.41]$$

$$\langle \Phi, Z, \Lambda, S, I, Percept, Mem, Correl, Inform, Integ, Response \rangle$$

Donde Φ, Z, Λ, S, I, $Percept$, Mem,$Correl$, $Inform$, $Integ$ y $Response$ se corresponden con los elementos definidos en el marco formal y en los párrafos anteriores.

Definimos la función de comportamiento *Behave* como una función que a partir de un estado de la red, un estado interno del sistema de detección y una aportación interna de los métodos de correlación proporciona una acción de respuesta, un nuevo estado interno del sistema y una nueva aportación. Esta función especifica el comportamiento global del modelo de detección de intrusos, y se define como:

$$Behave: \Sigma \; x \; S \; x \; I \; \rightarrow \; P \; x \; S \; x \; I \quad [3.42]$$

Donde la acción de respuesta del sistema de detección es el resultado de todo el complejo proceso de deliberación y decisión que involucra las funciones principales de las distintas etapas definidas en el modelo, el nuevo estado interno del sistema dependerá de la experiencia adquirida mediante la función de memorización a partir de las distintas percepciones y la nueva aportación se basará en el conocimiento real aprendido mediante la función de información a partir de los resultados de cada método de correlación. Es decir:

$$Behave(\sigma, s, \boldsymbol{\delta}) = \quad [3.43]$$

$$\langle Response\big(Integ(Correl(\boldsymbol{\varphi}, s), \boldsymbol{\delta})\big), Mem(\boldsymbol{\varphi}, s), Inform(Correl(\boldsymbol{\varphi}, s), \boldsymbol{\delta}) \rangle$$

$$con \; \boldsymbol{\varphi} = \; Percept(\sigma)$$

Finalmente, dado que las dinámicas de la red y del sistema se determinan mediante la evolución de sus respectivos estados internos en relación con el tiempo, podemos definir el comportamiento global mediante tres ecuaciones: la primera para describir el estado de la red según el tiempo y el comportamiento del sistema, la segunda para definir la evolución del estado interno del sistema y la tercera para especificar la modificación de la aportación de los métodos de correlación:

$$\sigma(t+1) = ModRed(\sigma(t), Exec(Response\big(Integ(Correl\big(\boldsymbol{\varphi}(t), s(t)\big), \boldsymbol{\delta}(t))\big), \sigma(t))$$

$$s(t+1) = Mem(\boldsymbol{\varphi}(t), s(t))$$

$$\boldsymbol{\delta}(t+1) = Inform(Correl(\boldsymbol{\varphi}(t), s(t)), \boldsymbol{\delta}(t)) \qquad [3.44]$$

$$con\ \boldsymbol{\varphi}(t) = Percept(\sigma(t))$$

Conclusión

En este capítulo se ha realizado una formalización rigurosa para evitar ambigüedades del escenario global en el que se contextualiza la problemática de los sistemas de detección de intrusos. En este sentido, después de especificar los distintos elementos, se puede definir el escenario global mediante la estructura:

$$GlobalScenario = \langle NetworkSystem, DetectionSystem \rangle \qquad [3.45]$$

A continuación se muestra un pequeño resumen de la formulación empleada a lo largo del capítulo, con el objetivo de facilitar una visión global de la misma. La tabla 3.1 muestra las definiciones principales de la red de comunicaciones.

Tabla 3.1. Resumen formulación del entorno.

$\boldsymbol{NetworkSystem = \langle \Sigma, P, \Gamma, Exec, ModRed \rangle}$		**[3.15]**
$\Sigma = \{\sigma_1, \sigma_2, \dots, \sigma_n\}$	$\sigma_i = \langle D, R_i, F_i, C \rangle$	[3.1],[3.4]
$P = \{\rho_1, \rho_2, \dots, \rho_n\}$	$p = \langle nombre, pre, post \rangle$	[3.5],[3.6]
$\Gamma = \{\gamma_1, \gamma_2, \dots, \gamma_n\}$		[3.7]
$Exec : P\ x\ \Sigma \rightarrow \Gamma$	$Exec : (P, \|\|)\ x\ \Sigma \rightarrow \Gamma$	[3.8],[3.11]
$ModRed : \Sigma\ x\ \Gamma \rightarrow \Sigma$		[3.13]

Finalmente, en la tabla 3.2 muestra el resumen de la formulación empleada en el modelado del sistema de detección.

Tabla 3.2. Resumen formulación del sistema de detección.

$\boldsymbol{DetectionSystem =}$ $\boldsymbol{\langle \Phi, Z, \Lambda, S, I, Percept, Mem, Correl, Inform, Integ, Response \rangle}$	**[3.41]**
$\Phi = \{\varphi_1, \varphi_2, \dots, \varphi_n\}$	[3.16]
$Z = \{\mu_1, \mu_2, \dots, \mu_n\}$	[3.20]
$\Lambda = \{i_1, i_2, \dots, i_n\}$	[3.24]
$S = \{s_1, s_2, \dots, s_n\}$	[3.30]
$I = \{\delta_1, \delta_2, \dots, \delta_n\}$	[3.36]
$Percept: \Sigma \rightarrow \Phi^n$	[3.19]
$Mem: \Phi^n\ x\ S \rightarrow S$	[3.31]
$Correl: \Phi^n x\ S \rightarrow Z^m$	[3.33]

$Inform: \mathrm{Z}\ x\ I \rightarrow\ I$	[3.37]
$Integ: Z^m x\ I\ \rightarrow\ \Lambda$	[3.39]
$Response: \Lambda\ \rightarrow\ P$	[3.26]

Capítulo 4

Método General de Integración

En este capítulo se desarrolla con mayor detalle la etapa de integración definida en el modelo general (ver figura 3.2). Más concretamente, se propondrá un método y un algoritmo para resolver la función de integración (*integ()*) propuesta en [3.40].

En la literatura se puede encontrar varios métodos de correlación para construir escenarios de ataque de alto nivel a partir de alertas de intrusiones de bajo nivel emitidas por los IDS. Cada uno de estos métodos tiene sus propias fortalezas y debilidades. Dado que los métodos correlacionan alertas usando mecanismos distintos, su integración puede proporcionar, potencialmente, mejores resultados (Ning *et al.*, 2004).

Es necesario realizar la etapa de integración de manera que, en el resultado final, se maximicen las ventajas y sinergias de los métodos integrados y se minimicen sus inconvenientes. Para alcanzar el objetivo anterior, esta tesis propone como hipótesis la ordenación de los métodos de correlación en función de su rendimiento para establecer un método de integración capaz de obtener el mejor resultado, donde aquéllos métodos con mejor rendimiento tendrán mayor aportación. Es decir, se debe establecer alguna medida de calidad o fiabilidad de los distintos

métodos de correlación, con el objetivo de que los más fiables condicionen el proceso de integración y tengan más peso en el resultado final.

Puesto que gran parte de la responsabilidad para la efectiva integración de los métodos de correlación recae sobre el conocimiento que se tiene sobre el comportamiento que cada una de ellas tuvo en el pasado, también será responsabilidad de este capítulo, proponer el mecanismo para valorar dicho comportamiento. En términos prácticos, se trata de definir con más detalle la función de información (*Inform()*) propuesta en la eq. [3.37].

Para establecer la metodología se debe responder a una serie de cuestiones clave: ¿Cuál será la medida de calidad de cada una de las técnicas de correlación?, ¿Qué método se va a utilizar para llevar a cabo la integración generalizada de múltiples técnicas? y, finalmente, ¿Cómo se utilizarán las medidas de calidad para modificar el método de integración con el fin de obtener los mejores resultados?

Los apartados siguientes darán respuesta a las cuestiones planteadas describiendo los distintos métodos que guiarán el proceso de integración. Si bien lo más importante es la descripción de un método de integración ponderado, previamente se debe definir tanto el propio método de integración general como la métrica que permitirá ponderar y modificar posteriormente su comportamiento.

Medida de Calidad

En este apartado se especifica la estructura interna de la función *Inform* definida en la etapa de integración del modelo de detección (eq. 3.38). Esta función proporciona al modelo la capacidad de aprender de la experiencia con el objetivo de adquirir el conocimiento real de la aportación de cada sistema de correlación al método de integración.

La aportación de cada método de correlación se basará en la evaluación de su rendimiento, aspecto fundamental en el campo de la detección de intrusos. Esta evaluación se centrará en medir la efectividad de los distintos sistemas en términos de su habilidad para clasificar o correlacionar correctamente. Otros objetivos del rendimiento como bajo consumo de recursos, resistencia a ataques contra el propio sistema de detección, etc., no se tienen en cuenta. Por lo tanto, es necesario disponer de una métrica que permita evaluar y comparar la calidad de cada uno de los métodos de correlación de manera objetiva (Gu *et al.*, 2006). Por supuesto, la medida de calidad de cada método puede variar con el tiempo en función de sus aciertos y errores.

Existen varias métricas que permiten evaluar distintos aspectos de los sistemas de detección. Generalmente, las más aceptadas son el ratio de *verdaderos positivos* (TP —True Positive) y el ratio de *falsos positivos* (FP —False Positive). Los dos ratios definen la probabilidad de que la salida del sistema de detección sea una alarma, pero, en el caso de los TP cuando verdaderamente ha ocurrido una intrusión y en el caso de los FP cuando la intrusión no se ha producido.

Los falsos positivos son posiblemente el principal problema de los sistemas de detección. Cuando se llevan a cabo respuestas ante una detección y ésta es un falso positivo, se producen frecuentemente denegaciones de servicios de usuarios o conexiones legítimas al cerrar la sesión de usuario o la conexión abierta. Además, cuando el número de falsos positivos es elevado, el sistema de detección deja de utilizarse, porque el administrador de seguridad pierde la confianza en el sistema, ya que malgasta mucho tiempo analizando si es un ataque verdadero antes de aplicar la respuesta oportuna.

Los enfoques que emplean los ratios TP y FP para evaluar el rendimiento de los distintos sistemas de detección suelen mostrarlos combinados mediante el uso de curvas ROC (Receiver Operating Characteristic) (Hancock y Wintz, 1966), (Lippmann y Fried, 2000). Una curva ROC muestra la relación entre TP y FP, donde la curva superior denota mejor rendimiento del sistema de

detección asociado ya que indica que por cada FP se consigue mayor ratio de TP. Sin embargo, cuando las curvas de los distintos IDS se cruzan es difícil decidir cuál es mejor.

Otro enfoque denominado análisis basado en coste también utiliza conjuntamente los ratios TP y FP (Stolfo *et al.*, 2000). En este caso la evaluación se considera en términos de medidas de coste: daño causado por una intrusión no detectada en el caso de los TP e inconveniente provocado por una falsa alarma en el caso de los FP.

(Gaffney y Ulvila, 2001) integran los enfoques de curvas ROC con análisis de coste para calcular el coste esperado de cada sistema de detección. El problema de los mecanismos de evaluación basados en análisis de coste es que las medidas de coste se determinan subjetivamente, por lo que no se pueden utilizar para evaluar y comparar objetivamente sistemas de detección (Gu *et al.*, 2006).

Otras métricas empleadas para la evaluación del rendimiento de los IDS son los ratios de detección bayesianos *valor predictivo positivo* y *valor predictivo negativo*, PPV y PPN respectivamente (Axelsson, 1999). Estas medidas dependen de los índices TP y FP pero, principalmente, están condicionadas por el concepto de probabilidad de intrusión inicial o *ratio base* (B).

Para buscar una medida que determine la cantidad de información que proporciona cualquier método de detección, también se han analizado aspectos relacionados con la teoría de la información. Un concepto básico en esta teoría, que indica la cantidad de información de un mensaje, es el de *entropía* (Shannon, 1948). Este concepto ha sido empleado dentro del campo de la seguridad de redes en aplicaciones relacionadas con la autenticación (Jakobsson *et al.*, 2008), la generación de claves robustas (Azimi-Sadjadi *et al.*, 2007) y la detección de anomalías (Yin *et al.*, 2004) y (Brauckhoff *et al.*, 2006).

Los aspectos relacionados con la teoría de la información también se han empleado para evaluar sistemas de detección. En (Lee y Xiang, 2001) se propone un enfoque donde los conceptos como

entropía (ecuación 4.1), *entropía condicional* (ecuación 4.2) y *entropía condicional relativa* son empleados para medir el rendimiento de los modelos de detección.

$$H(X) = -\sum_{i=1}^{n} p(x_i)\, log\, p(x_i) \qquad [4.1]$$

$$H(X|Y) = -\sum_{i=1}^{n}\sum_{j=1}^{m} p(x_i, y_j)\, log\, p(x_i, y_j) \qquad [4.2]$$

Donde X e Y son variables aleatorias discretas que representan respectivamente las entradas y salidas del sistema de detección.

Todos los trabajos anteriores proponen métricas que valoran distintos aspectos del rendimiento de un sistema de detección. Sin embargo, no aportan una métrica de evaluación global del sistema sino de aspectos parciales e, incluso, algunas dependen de parámetros subjetivos. Por lo tanto, es necesaria una métrica de evaluación objetiva y que tenga en cuenta todos los aspectos parciales anteriormente analizados.

En este sentido, en (Gu *et al.*, 2006) se propone una métrica que de alguna manera combina o integra los enfoques anteriores. La medida denominada *capacidad de detección de intrusos* (C_{ID} — Intrusion Detection Capability) se define como el ratio entre la información mutua de la entrada y la salida de un sistema de detección y la entropía de la entrada. La información mutua mide la reducción de la incertidumbre de la entrada mediante el conocimiento de la salida del sistema de detección. C_{ID} normaliza esta reducción empleando la entropía de la entrada.

$$C_{ID} = \frac{\mathrm{I(X;Y)}}{\mathrm{H(X)}} \qquad [4.3]$$

Donde la información mutua entre la entrada y la salida se define según la siguiente ecuación:

$$I(X;Y) = H(X) - H(X|Y) \qquad [4.4]$$

Por un lado, C_{ID} está basada en conceptos de la teoría de la información utilizados anteriormente como entropía, entropía condicional e información mutua y no depende de ningún parámetro subjetivo. Por otro lado, tiene en cuenta todos los aspectos determinantes en la evaluación de las capacidades de detección: TP, FP, PPV, NPV y B. Esto es así, ya que todas las ecuaciones que componen C_{ID} se pueden expandir y expresar en función de los cinco ratios anteriores (Gu *et al.*, 2006).

Los experimentos llevados a cabo en el trabajo anterior demuestran que la métrica C_{ID} no sólo es válida para evaluar, comparar y ordenar el rendimiento de los sistemas de detección sino que presenta los mejores resultados al compararse con el resto de métricas. Por lo tanto, utilizaremos esta medida de calidad para obtener la aportación de los distintos sistemas de correlación al método de integración. De manera que, cuanto mayor capacidad de detección de intrusos C_{ID}, mayor ponderación en el método de integración.

Trasladando la métrica anterior a nuestro modelo, tenemos que la entrada al sistema es $X = \sigma(t)$ y la salida de los métodos de correlación es $Y = \boldsymbol{\mu}(t)$. Según esto, la función *inform* propuesta en [3.37] y [3.38] que devuelve un vector de aportaciones con los valores asociados a la calidad de la información proporcionada por cada uno de los métodos se puede expresar como:

$$\boldsymbol{\delta}(t) = Inform(\, Y, \boldsymbol{\delta}(t-1)\,) = \frac{\mathrm{I}(\sigma(t); \boldsymbol{\mu}(t))}{\mathrm{H}(\sigma(t))} \qquad [4.5]$$

Método de Integración

Después de definir una métrica que permite evaluar la fiabilidad o calidad de la información que proporcionan cada uno de los métodos de correlación y antes de utilizarla en la modificación del proceso de integración, es necesario determinar cuál va a ser el método de integración utilizado para combinar el conjunto de hiperalertas procedentes de las distintas técnicas de correlación.

La salida de cada uno de los métodos de correlación de alertas estará formada por hiperalertas $\mu \in Z$ que indican los potenciales escenarios $i \in \Lambda$ detectados. Estos escenarios se suelen denominar *aislados* (Qin y Lee, 2004) por ser poco probable la detección de todos y cada uno de los ataques que componen el escenario completo, de ahí la ventaja de integrarlos con los resultados de otros métodos. La integración permitirá combinar escenarios parciales detectados por los distintos métodos en la etapa de correlación, con el objetivo de construir el escenario completo, aumentando las capacidades de detección y rendimiento con respecto al uso de cada una de las técnicas de manera aislada.

El método debe permitir integrar múltiples técnicas de correlación de manera general, independientemente del número de técnicas y de los distintos mecanismos de correlación empleados. Por lo tanto, la integración no puede tener en cuenta información a priori de los distintos métodos de correlación involucrados. En caso contrario, el método de integración dependería de las técnicas de correlación subyacentes y no pasaría de ser una integración ad hoc basada en las técnicas empleadas, impidiendo la incorporación automática de nuevos métodos de correlación al sistema o, incluso, obligando a adaptar todo el modelo para ello.

En este sentido, si se analizan detenidamente los trabajos que integran pares de métodos de correlación (Ning *et al.*, 2004) y (Qin y Lee, 2004), se apreciará que ambos combinan el método de *prerrequisitos y consecuencias* con un método de *clustering o similitud entre los atributos de las alertas* (este método no utiliza información previa). Además, ambos trabajos utilizan los resultados de la técnica de clustering para integrar las hiperalertas generadas por el método de prerrequisitos y consecuencias. Es decir, mediante el método de clustering se integran escenarios aislados sólo del método de prerrequisitos y consecuencias.

Basándose en los trabajos que integran pares de métodos, esta investigación propone como método de integración una extensión

del enfoque anterior. Es decir, la propuesta va a utilizar un método de clustering para llevar a cabo la integración, pero donde los escenarios aislados (hiperalertas) pueden provenir de cualquier método de correlación.

Existen distintos mecanismos que permiten obtener clusters o clasificar datos de entrada, entre los que se pueden encontrar desde métodos probabilísticos hasta algoritmos de clustering de grafos y redes neuronales.

Una hiperalerta de salida de cualquiera de los métodos de correlación se suele representar gráficamente mediante un grafo dirigido, indicando los distintos pasos del escenario aislado detectado. Por lo tanto, basándose en esa similitud, una primera aproximación para llevar a cabo la integración mediante la clasificación de hiperalertas en clusters es pensar en conceptos relacionados con la teoría de grafos, más concretamente, en algoritmos de clustering de grafos, muy estudiados y utilizados por la literatura en distintos ámbitos de aplicación (Foggia *et al.*, 2007) y (Flake *et al.*, 2003). De esta manera, algoritmos bien conocidos como el Clustering de Markov o MCL (van Dongen, 2000), Iterative Conductance Cutting o ICC (Vempala *et al.*, 2000) y Geometric Minimum Spanning Tree Clustering o GMC (Gaertler, 2002) se podrían emplear como mecanismos para llevar a cabo la integración. Sin embargo, el problema es que este tipo de algoritmos presentan unas necesidades de computación excesivamente altas, siendo en la mayoría de los casos problemas de complejidad NP-completo (Brandes, 2007), por lo que se deben descartar debido a las restricciones temporales inherentes a los sistemas de detección.

Otra alternativa a estudiar son las redes neuronales, uno de los métodos más utilizados en el campo de los IDS debido tanto a sus capacidades de clustering o clasificación (Han y Cho, 2006) como a sus habilidades para generalizar y detectar ataques nuevos a partir de otros observados previamente (Ghosh *et al.*, 2000). Ejemplos de tales aplicaciones se pueden encontrar tanto en la detección de ataques a nivel de aplicación (Lippmann y Cunningham, 2000) como a nivel de protocolos de red

(Lichodzijewski *et al.*, 2002), (Ramadas *et al.*, 2003) y (Mora *et al.*, 2006). Estas características las convierten en un mecanismo idóneo para su aplicación en el campo de la detección de intrusos (Zanero y Savaresi, 2004), solucionando el problema de la complejidad computacional.

No obstante, aunque las redes neuronales se han empleado extensamente en el campo de la detección de intrusos, el método de integración tiene dos necesidades específicas que se deben cubrir: se ha optado por el clustering como mecanismo de integración y, como se ha analizado en el capítulo anterior, la etapa de integración debe tener la capacidad de adquirir conocimiento. Por lo tanto, la red neuronal idónea presentará características de clustering y de aprendizaje continuo.

De entre el gran número de redes neuronales existentes, las que utilizan un mecanismo de aprendizaje no supervisado tienen la capacidad de agrupar los datos de entrada estableciendo distintas categorías o clusters (Clare y Cohen, 2001). Estas redes se denominan *autoorganizativas*, ya que es la propia red quien debe encontrar los clusters apropiados a partir de correlaciones entre las informaciones presentadas.

Dentro de los modelos autoorganizativos se pueden diferenciar dos grandes grupos desde el punto de vista estructural: aquéllos que tienen una dimensionalidad fija y topología de la red preestablecida, y los que modifican su dimensionalidad y topología durante el aprendizaje. Entre los primeros se encuentran los mapas autoorganizativos de Kohonen (Kohonen, 1982). Entre los segundos cabe destacar la red Growing Neural Gas (Fritzke, 1995).

Como se ha comentado en párrafos anteriores, es necesario que la red sea capaz de aprender durante su funcionamiento habitual, lo que le permitirá seguir incorporando conocimiento continuamente, este tipo de aprendizaje se denomina *on line*. De no ser así, con la aparición de ataques nuevos habría que volver a entrenar la red neuronal con estos ataques más los que se reconocía previamente. Por el contrario, el aprendizaje *off line*

está basado en una fase de entrenamiento que proporciona el aprendizaje y, posteriormente, una fase de funcionamiento que lleva a cabo la clasificación pero sin posibilidad de aprender.

El inconveniente de las redes neuronales con aprendizaje *on line* es que deben enfrentarse al problema denominado *dilema de la estabilidad y plasticidad del aprendizaje* (Grossberg, 1980). Este dilema plantea el siguiente interrogante: ¿cómo una red puede aprender nuevos patrones (plasticidad), sin olvidar los patrones previamente aprendidos (estabilidad)?. El aprendizaje off line no presenta problemas de estabilidad en su operación, gracias a que son estáticas en la fase de funcionamiento. Sin embargo, en las redes on line, debido al carácter dinámico de las mismas, el estudio de la estabilidad suele ser un aspecto a tener en cuenta.

En el mapa autoorganizativo desarrollado por Kohonen, las neuronas están unidas entre sí formando una rejilla, normalmente bidimensional. Tiene serias limitaciones a la hora de establecer el mapa del espacio de los vectores de entrada, ya que durante el aprendizaje no puede variar la estructura de la rejilla. Además, desde el punto de vista de la detección de intrusos, tal como se ha comentado anteriormente, es importante que el mecanismo de aprendizaje sea on line, característica que no tienen los mapas autoorganizativos, ya que se trata de una red con aprendizaje off line que no puede aprender nada durante el funcionamiento. Debido a las razones anteriores, y aunque se trata de una red cuyo uso está muy extendido, se ha descartado el empleo de los mapas autoorganizativos.

Las redes Growing Cell Structures (Fritzke, 1993), Neural Gas (Martinetz y Schulten, 1991) y Growing Neural Gas (Fritzke, 1995) son válidas para los requisitos del problema. Sin embargo, las dos primeras presentan algunas desventajas con respecto a la tercera, las GCS necesitan establecer la dimensión topológica a priori y las NG requieren definir previamente el tamaño máximo de la red. Además, tienen una complejidad del aprendizaje mayor que las GNG, por lo que precisan tiempos de procesamiento muy superiores.

Se ha optado por utilizar la red GNG por ser la más adecuada, al no tener ninguna restricción previa, ni topológica ni de tamaño. La GNG por una parte mantiene todas las características de las redes autoorganizativas como la capacidad de clustering y, además, permite seguir aprendiendo mientras funciona, por lo que presenta la capacidad de clasificar adecuadamente escenarios nuevos sin necesidad de volver a reentrenar la red con todos los anteriores.

Algoritmo de la Función de Integración

En los párrafos anteriores se han analizado los posibles algoritmos de clustering que se utilizarán en el proceso de integración de las salidas de los múltiples métodos de correlación, concluyendo que el que más se adecua a nuestros intereses es el de la red neuronal Growing Neural Gas. Además; también se ha establecido una métrica basándose en conceptos de la teoría de la información que permite evaluar mediante un único valor objetivo la bondad o rendimiento de cada uno de los métodos de correlación; finalmente, en este apartado, para optimizar el resultado final de la etapa de integración y teniendo en cuenta todo lo anterior, se propone una modificación del algoritmo de aprendizaje de la red neuronal, utilizando la métrica definida en apartados anteriores, con el objetivo de que los métodos de correlación más fiables condicionen el proceso de integración y tengan más peso en el resultado final. En este sentido, una intrusión $i \in \Lambda$ se define como:

$$i = Integ(\boldsymbol{\mu}, Inform(\boldsymbol{\mu}, \boldsymbol{\delta})) = GNG^{*}(\boldsymbol{\mu}, \boldsymbol{C_{ID}}) \qquad [4.6]$$

Donde GNG^{*} es una red neuronal GNG modificada.

Si se analiza detenidamente el algoritmo de aprendizaje de la red Growing Neural Gas, se puede observar que la red neuronal aprende fundamentalmente mediante dos aspectos distintos: la inserción de nuevas neuronas y la adaptación de los vectores de referencia.

En el primer caso, el incremento de un error local acumulado en cada neurona vencedora permitirá posteriormente la inserción de nuevas neuronas cerca de aquéllas con mayor error. El incremento será el cuadrado de la distancia existente entre el vector de entrada a la etapa de integración y el vector de referencia de la neurona ganadora (ecuación 4.7).

$$\Delta E_{s_1} = \|\boldsymbol{\mu} - w_{s_1}\|^2 \quad [4.7]$$

En el segundo, y principal, la adaptación de los vectores de referencia se realiza para la neurona vencedora y sus vecinas con respecto a los patrones de entrada, con el fin de aprender de dichos patrones. Las modificaciones dependerán de dos factores de aprendizaje: el de la neurona ganadora ε_1 y el de sus vecinas ε_2, siendo mayor ε_1 y, por consiguiente, mayor el aprendizaje de la neurona ganadora (ecuación 4.8 y 4.9).

$$\Delta w_{s_1} = \varepsilon_1(\boldsymbol{\mu} - w_{s_1}) \quad [4.8]$$

$$\Delta w_i = \varepsilon_2(\boldsymbol{\mu} - w_i), \forall i \in \mathcal{N}_{s_1} \quad [4.9]$$

Es precisamente en estas fases del algoritmo de aprendizaje donde se va a introducir la métrica *capacidad de detección de intrusos* o C_{ID} definida en apartados anteriores, modificando el comportamiento del aprendizaje y beneficiando a los métodos con mayor rendimiento.

En primer lugar, se modifica la ecuación 4.7 introduciendo la medida de calidad C_{ID} del método de correlación correspondiente, la cual se multiplicará por la distancia al cuadrado entre los vectores de referencia del patrón de entrada y la neurona ganadora:

$$\Delta E_{s_1} = C_{ID_i}\|\boldsymbol{\mu} - w_{s_1}\|^2 \quad [4.10]$$

Donde $\boldsymbol{C_{ID}} = Inform(\boldsymbol{\mu}, \boldsymbol{\delta}(t-1))$ y C_{ID_i} es la componente i del vector $\boldsymbol{C_{ID}}$ que representa la medida de calidad del método de correlación $Correl_i \in F_{CORREL}$.

El incremento del error local de la neurona vencedora será mayor cuanto mayor sea la medida de calidad (rendimiento) del método de correlación asociado al patrón de entrada. El mayor incremento del error provocará, en etapas posteriores del algoritmo, la inserción de más neuronas cerca de las vencedoras de los patrones asociados a métodos fiables, aumentando la resolución y eficiencia de esa zona de la red y, por lo tanto, ponderando positivamente la clasificación de estos patrones.

Se debe notar que, dado que el factor de fiabilidad es un valor entre 0 y 1, los errores locales de las neuronas se incrementarán más lentamente. Sin embargo, este hecho no afecta a la inserción de nuevas neuronas, ya que no se trata de alcanzar un determinado valor máximo; simplemente, cuando llega el momento de insertar neuronas, se mira aquélla que tenga mayor error, independientemente de su valor.

En segundo lugar, se modifica la ecuación 4.8 introduciendo la medida de calidad C_{ID_i}, la cual se multiplicará por la distancia euclídea entre los vectores de referencia del patrón de entrada y la neurona ganadora y por el factor de aprendizaje de la neurona vencedora ε_1:

$$\Delta w_{s_1} = C_{ID_i}\, \varepsilon_1 (\boldsymbol{\mu} - w_{s_1}) \qquad [4.11]$$

El incremento del vector de referencia de la neurona ganadora será mayor cuanto mayor sea la medida de rendimiento del método de correlación asociado al patrón de entrada. El mayor incremento provocará que la topología de la red neuronal se ajuste (aprendizaje o autoorganización) más hacia los patrones con mayor fiabilidad. Por lo tanto, son los métodos de correlación más fiables los que tienen mayor peso o condicionan más el aprendizaje y, por consiguiente, el proceso de integración.

El aprendizaje será más lento que el algoritmo original. Sin embargo, una de las características de la red Growing Neural Gas es que obtiene tiempos de aprendizaje muy bajos, por ejemplo, tres veces inferior a los tiempos de los mapas SOM (Flórez, 2001), por lo que la modificación seguirá teniendo tiempos razonables y asumibles en el ámbito de los problemas de detección de intrusos.

En tercer lugar, se modifica la ecuación 4.9 introduciendo la métrica C_{ID_i}, la cual se multiplicará por la distancia euclídea entre los vectores de referencia del patrón de entrada y los vecinos de la neurona ganadora, y por el factor de aprendizaje de las neuronas vecinas ε_2:

$$\Delta w_i = C_{ID_i}\, \varepsilon_2(\boldsymbol{\mu} - w_i), \forall i \in \mathcal{N}_{s_1} \qquad [4.12]$$

Este último paso permite que cuando la medida de calidad C_{ID_i} es alta, no sólo se ajuste más rápidamente la neurona vencedora al patrón de entrada (ecuación 4.11), sino también la zona de vecindad asociada a la neurona ganadora.

El algoritmo GNG con las modificaciones anteriores permite la integración mediante clustering de múltiples métodos de correlación de alertas complementarios. Por una parte, beneficia a los métodos con mayores capacidades de detección, generalmente, aquéllos que utilizan todo el conocimiento de los escenarios. Por otra parte, al permitir integrar también métodos basados en similitud entre atributos, añade la capacidad de detectar escenarios nuevos, pero penalizando su aportación, ya que estos métodos suelen producir peores rendimientos que pueden perjudicar el resultado final.

Finalmente, una vez introducidas las modificaciones a la red neuronal GNG, el método o algoritmo de la función de integración [eq. 3.40] se puede expresar de la siguiente manera:

1. Se crea el conjunto $\mathcal{A}$ con únicamente dos neuronas c_1 y c_2

$$\mathcal{A} = \{c_1, c_2\} \quad [4.13]$$

con sus respectivos vectores de referencia w_1 y w_2 inicializados aleatoriamente, siguiendo generalmente la función de densidad de probabilidad $p(\boldsymbol{\mu})$.

Se inicializa el conjunto de conexiones al conjunto vacio

$$\mathcal{C} = \phi \quad [4.14]$$

Se inicializan los índices de calidad de los métodos de correlación según la ecuación [4.5].

$$C_{ID} = Inform(\boldsymbol{\mu}, \delta(t-1)) \quad [4.15]$$

2. Se obtiene una señal de entrada $\boldsymbol{\mu}$.
3. Se localizan la neurona ganadora s_1 y la segunda neurona más cercana s_2 obtenidas como

$$s_1 = arg\ min_{c \in \mathcal{A}} \|\boldsymbol{\mu} - w_c\| \quad [4.16]$$

$$s_2 = arg\ min_{c \in \mathcal{A} - \{s_1\}} \|\boldsymbol{\mu} - w_c\| \quad [4.17]$$

4. Si la conexión entre ambas neuronas s_1 y s_2 no existe, entonces es creada

$$\mathcal{C} = \mathcal{C} \cup \{(s_1, s_2)\} \quad [4.18]$$

Se fija a 0 la edad de la conexión entre ellas

$$edad(s_1, s_2) = 0 \quad [4.19]$$

5. Se modifica la variable del error acumulado local de la neurona ganadora según se ha comentado en párrafos anteriores. Concretamente, se añade el cuadrado de la distancia existente entre la señal de entrada y el vector de referencia de la neurona ganadora, modificado mediante la medida de calidad del método de correlación.

$$\Delta E_{s_1} = C_{ID_i} \|\boldsymbol{\mu} - w_{s_1}\|^2 \quad [4.20]$$

6. Se adaptan los vectores de referencia de la neurona ganadora s_1 así como de todas sus neuronas vecinas. Esta adaptación se pondera según ε_1, ε_2 y C_{ID} .

$$\Delta w_{s_1} = C_{ID_i}\, \varepsilon_1(\boldsymbol{\mu} - w_{s_1}) \quad [4.21]$$

$$\Delta w_i = C_{ID_i}\, \varepsilon_2(\boldsymbol{\mu} - w_i), \forall i \in \mathcal{N}_{s_1} \quad [4.22]$$

7. Se incrementan las edades de todas las aristas que salen de s_1

$$edad(s_1, i) = edad(s_1, i) + 1, \forall i \in \mathcal{N}_{s_1} \quad [4.23]$$

8. Se eliminan todas las aristas cuya edad sea mayor que una cierta cantidad $edad_{max}$. Si al producirse la eliminación una neurona se queda aislada, es decir, sin aristas que salgan de ella, esta neurona es eliminada.

9. Cada cierto número λ de señales de entrada generadas se inserta una neurona según el siguiente proceso:

   - Se determina la neurona q con el mayor error local acumulado:

$$q = arg\ max_{c \in \mathcal{A}}\, E_c \quad [4.24]$$

   - Se localiza la neurona f vecina de q con mayor error local acumulado:

$$f = arg\ max_{c \in \mathcal{N}_q}\, E_c \quad [4.25]$$

   - Se inserta una nueva neurona r entre f y q:

$$\mathcal{A} = \mathcal{A} \cup \{\mathrm{r}\} \quad [4.26]$$

$$w_r = \frac{(w_q + w_f)}{2} \quad [4.27]$$

- Se insertan nuevas conexiones entre la neurona r y las neuronas f y q, eliminando la conexión que existía entre éstas:

$$\mathcal{C} = \mathcal{C} \cup \{(r,q),(r,f)\}$$
$$\mathcal{C} = \mathcal{C} - \{(q,f)\} \quad [4.28]$$

- Se reduce el error de las neuronas f y q por una fracción α:

$$\Delta E_q = -\alpha E_q$$
$$\Delta E_f = -\alpha E_f \quad [4.29]$$

- Se interpola el error de la neurona r entre los errores de f y q:

$$E_r = \frac{(E_q + E_f)}{2} \quad [4.30]$$

10. Se reduce el error de todas las neuronas

$$\Delta E_c = -\beta E_c, \forall c \in \mathcal{A} \quad [4.31]$$

11. Si se cumple una determinada condición (tamaño máximo de la red, error cuadrático medio o cualquier condición definida por el usuario) se finaliza el proceso. Si no es así, se continúa con el paso 2. El proceso puede continuar indefinidamente dado que los parámetros que intervienen en el método de aprendizaje son constantes, en otras redes los parámetros disminuyen con el número de iteraciones, por lo que al llegar a cero ya no es posible que la red continúe aprendiendo.

Capítulo 5

Arquitectura Distribuida Propuesta

El modelo general de detección desarrollado en los capítulos anteriores define conceptual y formalmente la metodología empleada para posibilitar que múltiples métodos de correlación puedan funcionar juntos de manera integrada e, incluso, que obtengan el mejor aprovechamiento del método de integración. Sin embargo, ¿Cómo podemos llevar a la práctica el modelo?, es decir, ¿Cuál es la arquitectura que da soporte y hace viable dicho modelo en entornos realistas?.

El presente capítulo se dedica a la propuesta de una arquitectura de sistema de detección de intrusos distribuida que permita la viabilidad del modelo general de detección desarrollado en capítulos anteriores utilizando la tecnología existente. Se realiza una justificación de la conveniencia de utilizar enfoques distribuidos y se describen los módulos funcionales junto con su estructura organizativa y los mecanismos de comunicación entre ellos.

Modelo Conceptual de la Arquitectura

Se ha analizado en el capítulo del estado del arte que el tamaño y heterogeneidad de las redes ha incrementado la complejidad estructural de los sistemas de detección. Estos sistemas han pasado de IDS locales, incapaces de aportar una visión global de la seguridad de la red o detectar ataques simultáneos en varios nodos de la red (Ganame *et al.*, 2008), a IDS distribuidos formados por sensores locales que cooperan en el proceso de detección global.

No obstante, no todos los sistemas distribuidos tienen las mismas ventajas, éstas dependen de su arquitectura. Las arquitecturas centralizadas que recogen información en todos los nodos pero sólo analiza un nodo central (Ullrich, 2004), presentan una carencia absoluta de escalabilidad. Las organizaciones jerárquicas que distribuyen nodos intermedios encargados de filtrar la información irrelevante, disminuyen la cantidad de datos a procesar por el nodo raíz (Chu *et al.*, 2005), pero siguen acusando el problema de la escalabilidad, aunque en menor medida.

Además del problema de escalabilidad analizado en el párrafo anterior, estos tipos de arquitecturas presentan otro problema importante desde el punto de vista de la seguridad: la baja tolerancia a fallos. Los dos tipos de sistemas presentan el problema de un único punto de fallo, si se compromete o falla el nodo central, dejará de funcionar todo el sistema. Si un atacante consigue detener el analizador central (denegación de servicio, etc.), toda la red estará sin protección. Éste es un problema grave, desde el punto de vista de la seguridad, que se suele mitigar introduciendo redundancia.

Para solucionar los problemas fundamentales de las arquitecturas anteriores, es necesario usar organizaciones donde tanto la recogida de la información como el análisis de datos se lleve a cabo de forma descentralizada o distribuida (Kruegel *et al.*, 2005). Las arquitecturas completamente distribuidas son

perfectamente escalables y no presentan el problema de seguridad del punto único de fallo.

Dentro de estas arquitecturas completamente distribuidas se han utilizado distintos mecanismos o tecnologías: un enfoque sugiere que sólo los nodos donde tenga lugar realmente la intrusión colaboren o cooperen en su detección (Arora *et al.*, 2004); otro enfoque plantea el uso de código móvil donde los agentes viajan por la red recogiendo información relevante del ataque y, por qué no, realizando ellos mismos el análisis (Kannadiga y Zulkernine, 2005); finalmente, una perspectiva completamente descentralizada utiliza un esquema de cooperación peer-to-peer (Vlachos *et al.*, 2004).

La propuesta de esta tesis se basa en una arquitectura completamente distribuida debido a las ventajas de escalabilidad y tolerancia a fallos comentadas en los párrafos anteriores. Además, se apoya en las Arquitecturas Orientadas a Servicios (Service Oriented Architecture —SOA) como base a partir de la cual construir los componentes principales del sistema de detección de intrusos distribuido.

Casos de Uso

La arquitectura de un sistema puede describirse a través de cinco vistas interrelacionadas (figura 5.1), donde cada vista es una proyección de la organización y la estructura del sistema, centrada en un aspecto particular de éste (Booch y Rumbaugh, 2001).

Las vistas de diseño y procesos comprenden los requisitos funcionales y funcionamiento del sistema, las dos vistas se pueden describir con los mismos mecanismos: diagrama de clases para la faceta estática y diagrama de actividades para la componente dinámica. La vista de implementación está formada por los componentes que se utilizan para ensamblar y hacer disponible el sistema físico. La vista de despliegue contiene la configuración de los nodos que forman la topología hardware de la red sobre la que se ejecuta el sistema. Finalmente, la vista de

casos de uso define el comportamiento del sistema tal y como es percibido por los distintos usuarios. Los casos de uso definen *el qué* del sistema, mientras el resto de vistas especifican *el cómo*, de ahí que se diga que los casos de uso guían la arquitectura (Booch y Rumbaugh, 2001).

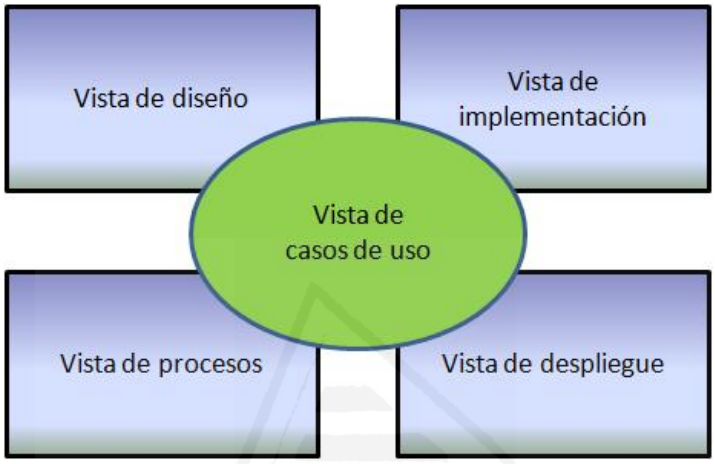


Figura 5.1. Modelado de la arquitectura de un sistema (Booch y Rumbaugh, 2001).

Según lo anterior, antes de abordar la estructura y organización concreta de esta arquitectura, es conveniente analizar los casos de uso principales y las características más destacables de la arquitectura SOA. Posteriormente, se presentará el modelo conceptual de la arquitectura junto con las distintas vistas tanto estáticas como dinámicas o de comportamiento.

La figura 5.2 muestra los cinco casos de uso principales: *suscripción, cancelar suscripción, iniciar detección, detener detección* y *consultar resultados*. *Iniciar detección* constituye el caso de uso principal, pone en marcha el sistema distribuido de detección y encapsula toda la funcionalidad descrita en las distintas etapas de percepción, correlación, integración y respuesta definidas del modelo general propuesto en el capítulo 3, este caso se analizará con detalle a continuación. Por el contrario, *detener detección* para el sistema.

Dado que el sistema de detección distribuido basa su funcionamiento en la arquitectura SOA, el caso de uso *suscripción* permitirá suscribirse a los distintos servicios que ofrece el sistema para recibir las alertas o hiper-alertas generadas por estos servicios. La suscripción la podrán realizar tanto los componentes de nuestro propio sistema como los de otros sistemas externos gracias al uso de protocolos estándar. *Cancelar suscripción* se empleará para anular la suscripción realizada previamente y evitar la recepción de información de seguridad desde el sistema.

Finalmente, consultar resultados posibilita la realización de informes y estadísticas elaboradas sobre el estado de seguridad de la red. Estos informes serán empleados por el administrador de seguridad, pero también podrán ser utilizados por otros sistemas externos, principalmente, aquéllos que recogen información sobre la seguridad global de grandes redes como Internet.

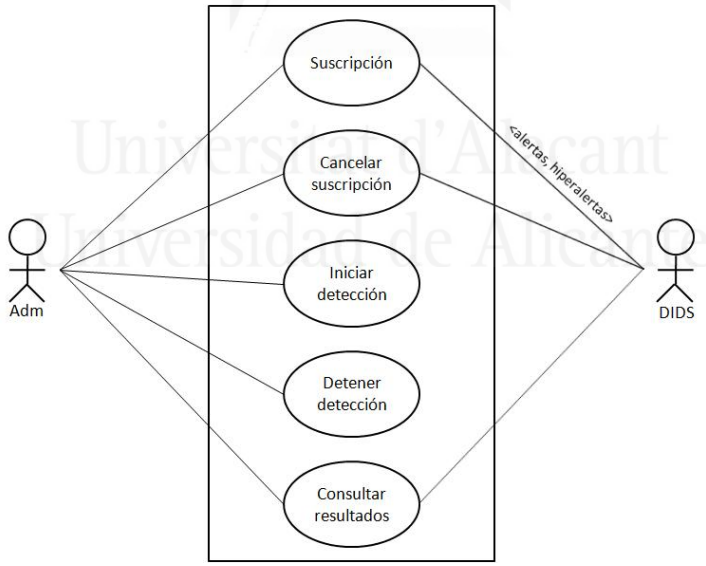


Figura 5.2. Diagrama de casos de uso generales.

Dado que los casos de uso principales se pueden descomponer a su vez en otros de distintas granularidad, se va a estudiar con

mayor nivel de detalle el caso de uso principal *iniciar detección*. La figura 5.3 muestra su descomposición en otros casos de menor nivel de abstracción.

El caso de uso *iniciar detección* permite iniciar los sensores que realizarán la función de percepción e iniciar las etapas de correlación, integración y respuesta, invocar la publicación de los distintos servicios y activar la notificación de alertas a los sistemas o componentes que se hayan suscrito a los servicios. Sin embargo, principalmente, lanza el motor de detección que se encargará de ejecutar y controlar todo el proceso de detección, desde la detección de ataques, pasando por la correlación y la integración, hasta la respuesta a las intrusiones. Como se puede observar, la invocación de los cuatro casos de uso anteriores por parte del motor de detección es opcional, este aspecto posibilita la ejecución de todo el modelo descrito en el capítulo 3 o sólo alguna de sus partes.

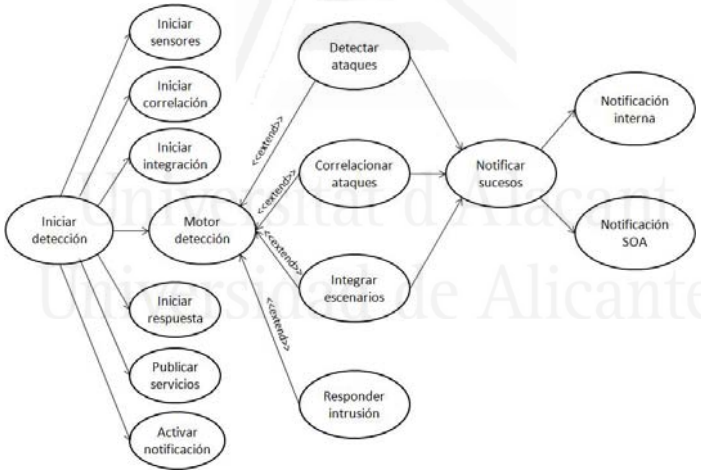


Figura 5.3. Caso de uso iniciar detección.

Finalmente, *notificar sucesos* es invocado para informar de las alertas o hiper-alertas encontradas a aquellos sistemas o componentes que se hayan suscrito a los servicios correspondientes. Será común que un componente de correlación se suscriba para recibir las alertas detectadas por los sensores de detección, así como que un elemento de integración esté suscrito

para recibir las hiper-alertas generadas por los sistemas de correlación.

Es importante destacar que la funcionalidad del sistema sigue generalmente la especificación de distintos patrones de diseño. En este sentido, iniciar detección implementará el patrón de diseño *fachada* ya que presentará una interfaz común que encapsulará el hecho de poder ejecutar opciones muy distintas: desde el modelo completo a sólo una etapa. El motor de detección será el *orquestador* que controlará la ejecución, posiblemente opcional, de las principales funciones del sistema. Finalmente, la publicación de servicios, junto con el caso de uso de mayor nivel suscripción, sigue el patrón *publicación-suscripción* u observador.

Una vez definido el comportamiento del sistema a través de sus casos de uso, se va a analizar la arquitectura distribuida SOA que va a permitir acceder a dichas funcionalidades. El estudio incluye los elementos principales tanto desde el punto de vista estructural como funcional, así como las principales características que SOA aportará al sistema de detección distribuido.

Arquitectura SOA

La arquitectura del sistema de detección debe utilizar una organización completamente distribuida, ir más allá de protocolos de comunicación ad-hoc, incorporar tecnologías capaces de ofrecer su funcionalidad de manera abierta y normalizada, facilitando la interoperabilidad entre los componentes del sistema y entre los componentes y otras aplicaciones. En este sentido, las arquitecturas orientadas a servicios proporcionan un modelo en el que cada elemento de la red ofrece su funcionalidad a través de servicios independientes y accesibles de forma estandarizada (Douglas, 2003).

Desde el punto de vista estructural, SOA es frecuentemente clasificado de diferentes maneras, dependiendo de la tecnología de implementación utilizada para construir los servicios. El principal modelo, desde el punto de vista estructural, inspirado

en el conjunto inicial de Servicios Web estándar, define SOA como una arquitectura modelada alrededor de tres componentes básicos: el solicitante de servicio, el suministrador de servicio y el servicio de registro (Erl, 2005).

Desde el punto de vista funcional, la arquitectura puede describirse también en función de las diferentes fases del modelo SOA en la que se encuentren los elementos que participan en el mismo: publicación, búsqueda, descubrimiento y consumo.

1) La *publicación* es la primera de las fases en las que entra todo servicio. En dicha fase, el servicio deberá localizar un registro o servidor de publicación y, mediante el protocolo UDDI (Universal Description, Discovery and Integration), enviarle, en forma de hojas WSDL (Web Services Description Language), toda la documentación que describe el servicio que pretende prestar.

2) En la fase de *búsqueda* cualquier consumidor que desee obtener un servicio deberá previamente conocerlo. Para ello, localizará un servidor de publicación al que solicitar la documentación del servicio. Para esta fase se emplea, nuevamente, el protocolo UDDI.

3) Durante la *fase de descubrimiento* el servidor de publicación enviará toda la documentación de servicio requerido al consumidor, dicha información serán las hojas WSDL registradas por el servicio en la fase de publicación y que incluyen todos los datos necesarios para localizarlo y consumirlo.

4) El *consumo* es la fase más importante de todas pues es la que otorga verdadero sentido a todo el sistema. En esta fase, los consumidores, una vez realizado el descubrimiento de servicios, se encuentran en disposición de consumirlos o, lo que es lo mismo, de dirigirse directamente a los computadores y dispositivos de red y solicitar los servicios que ofrecen. Toda la comunicación entre servicios y consumidores en esta fase se realizará mediante solicitudes y respuestas SOAP.

En la actualidad, existe un conjunto de tecnologías estándar que hacen posible el desarrollo de aplicaciones basadas en la arquitectura SOA: el protocolo HTTP, el lenguaje XML, y sobre ellos, el lenguaje WSDL y los protocolos SOAP y UDDI. La conjunción de estas tecnologías constituye la base de los Servicios Web (WS), principal implementación de la arquitectura SOA, aunque se puede desarrollar con cualquier tecnología.

La utilización de la arquitectura conceptual SOA proporcionará al sistema de detección las siguientes características (Erl, 2005):

- Acoplamiento débil. La relación entre los servicios minimiza las dependencias y sólo necesita tener conciencia de la existencia de los servicios. En las arquitecturas distribuidas tradicionales, la interacción esperada entre los componentes se tiene en cuenta en la fase de diseño. Esta dependencia del diseño es una forma de acoplamiento que, aunque eficiente en la localización de los componentes, conduce a un ajuste estricto de los componentes que una vez implementados no es fácil alterar.

- Abstracción. Más allá de lo que se describe en el contrato de servicio, los servicios ocultan o abstraen lógica subyacente al mundo exterior. La funcionalidad se puede proporcionar a través de componentes, de sistemas heredados o mediante cualquier otra fuente.

- Reusabilidad y composición. La lógica completa se divide en servicios con la intención de promover su reusabilidad. Cuando se diseña apropiadamente, colecciones de servicios pueden ser coordinadas y ensambladas para formar servicios compuestos de mayor nivel.

- Interoperabilidad. Aunque la reusabilidad está presente generalmente en los enfoques distribuidos tradicionales, SOA alcanza niveles más profundos al fomentar la interoperabilidad entre aplicaciones. Mediante SOA los servicios se relacionan y operan intrínsecamente a través del

descubrimiento, permitiendo oportunidades de integración imprevistas.

El acoplamiento débil y la interoperabilidad son posiblemente las características que más diferencian SOA de arquitecturas distribuidas como CORBA, COM/DCOM, DCE y otros esquemas propietarios (Marks y Bell, 2006). Estas características se consiguen en gran medida debido a la utilización de protocolos estándar para Servicios Web, principalmente SOAP para los mensajes, WSDL para la descripción de los servicios y UDDI para el descubrimiento.

El acoplamiento débil aportará al sistema de detección de la escalabilidad y flexibilidad necesarias para establecer cualquier tipo de configuración de la arquitectura del sistema, al no tener restricciones de la fase de diseño. La interoperabilidad dotará al sistema de detección de la capacidad para utilizar, trabajar y cooperar con otros sistemas de detección externos.

Atendiendo a los aspectos expuestos en los párrafos anteriores, la arquitectura propuesta tiene una organización completamente distribuida que le aporta escalabilidad y evita el problema del punto único de fallo. Además, se basa en los conceptos y estándares de la arquitectura SOA que permite incorporar aspectos como acoplamiento débil, interoperabilidad, abstracción, reusabilidad y composición de servicio, no sólo entre los distintos elementos del sistema, sino también con otros sistemas distintos, mediante el uso de protocolos estándar y abiertos.

Finalmente, dado que esta investigación aborda aspectos del problema de la seguridad de redes, es necesario tener en cuenta que la adopción de la arquitectura SOA genera nuevos riesgos. No obstante, aunque caen fuera del alcance de esta tesis, estos riesgos están siendo abordados intensamente por la comunidad científica, muestra de ello son los estudios sobre métricas de seguridad SOA (Magott y Woda, 2008), seguridad inherente de XML (Phan, 2007), seguridad de servicios Web (Bertino y Martino, 2006), configuraciones de seguridad SOA (Satoh *et al.*, 2008) y

seguridad de servicios de orquestación externos (Chen y Lukkien, 2007).

Arquitectura Conceptual

La arquitectura de un sistema simplemente describe los elementos de alto nivel y sus relaciones (Harmon *et al.*, 2001). Cualquier discusión debe empezar haciendo una distinción entre arquitecturas abstractas y de implementación. El modelo abstracto permite que el mismo elemento sea implementado en más de un modelo de componentes y proporciona protección frente a cambios en la tecnología.

La arquitectura de cualquier sistema debe ser independiente de la implementación para poder acomodar la elección de distintas tecnologías existentes y aquéllas que aparecerán en el futuro. De esta manera, la tecnología se puede cambiar sin afectar al núcleo del sistema e, incluso, se pueden soportar múltiples tecnologías simultáneamente.

Una arquitectura distribuida se divide en subarquitecturas (Harmon *et al.*, 2001):

- Arquitectura de aplicaciones. Se refiere al conjunto de sistemas de una compañía y sus relaciones.
- Arquitectura técnica. Describe el diseño global de los sistemas de la compañía. Establece las capas funcionales y los niveles en los que se puede dividir una aplicación. También define los marcos, servicios y patrones sobre los que se confiará el desarrollo de aplicaciones basadas en componentes.
- Arquitectura de implementación. Se derivará de los requisitos impuestos por la tecnología utilizada.
- El modelo de negocio. Determina los componentes que se utilizarán en un proceso de negocios específico. Se suele dividir en dos: modelo de negocio de dominio, aplicable a un conjunto de aplicaciones, y modelo de negocio de aplicación, específico de una aplicación concreta.

- Arquitectura operacional. Se deriva de los requerimientos operacionales específicos.

La arquitectura técnica es la que verdaderamente describe el diseño global o estructura de los sistemas. Está formada por un modelo conceptual del sistema y una infraestructura. Dicho modelo conceptual establece los conceptos principales de la arquitectura mediante la definición de los distintos elementos, su organización y relación, para lo cual se emplean generalmente patrones arquitectónicos y de diseño.

El modelo propuesto en esta tesis se basa en la combinación de tres tipos de arquitecturas distribuidas: arquitecturas de N-niveles, orientadas a servicios o SOA y arquitecturas de componentes software distribuidos. Concretamente, la arquitectura estará formada por componentes software distribuidos, organizados según patrones arquitectónicos de n-niveles y que ofrecen su funcionalidad o comportamiento bajo un paradigma orientado a servicios.

Nuestro modelo conceptual de la arquitectura técnica se basa en una arquitectura distribuida de componentes de negocio electrónico definida por (Harmon *et al.*, 2001), pero adaptada a los elementos principales del modelo de detección de intrusos definido en el capítulo 3, a los casos de uso generales y a la arquitectura SOA subyacente empleada.

La figura 5.4 muestra el modelo conceptual donde se pueden observar, entre otros, los elementos principales del modelo que constituyen a su vez el caso de uso más importante y los componentes que permitirán ofrecer la funcionalidad mediante SOA. A continuación analizaremos con mayor nivel de detalle la arquitectura conceptual.

El modelo conceptual permite describir los detalles de dos conceptos básicos:

- Las capas funcionales. Definen la responsabilidad de los componentes y abordan aspectos de escalabilidad, distribución e independencia de la tecnología.

- Niveles de distribución. Describen cómo se correlacionan los distintos componentes en un sistema de computación distribuido; representan distribuciones lógicas del modelo y proporcionan patrones de propósito general para sistemas distribuidos.

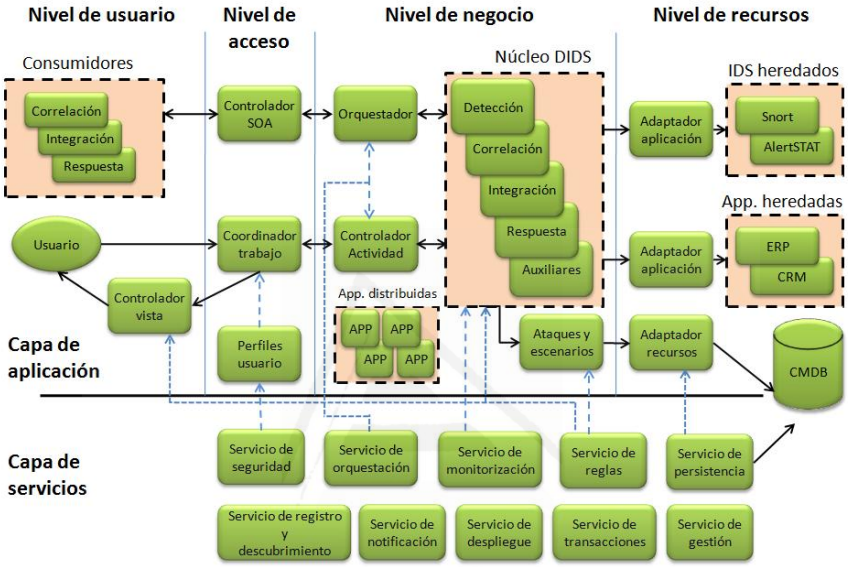


Figura 5.4. Modelo conceptual de la arquitectura.

Como se puede observar en la figura 5.4, la arquitectura está dividida en dos capas funcionales: capa de servicios y capa de aplicación. La primera está formada por los servicios comunes que debe proporcionar la infraestructura a todas las aplicaciones. La segunda se compone de los distintos elementos que constituyen el patrón a seguir en todas las aplicaciones de la organización y, por consiguiente, en el sistema de detección de intrusos distribuido. En este sentido, el modelo conceptual sigue un patrón arquitectónico denominado layers (Fowler, 2002) que permite segmentar las responsabilidades del sistema en distintos subsistemas o capas, y que aborda aspectos de escalabilidad, distribución e independencia de la tecnología.

La capa de servicios está formada por los siguientes servicios:

- Seguridad. Componente que permitirá llevar a cabo todas las tareas relacionadas con la seguridad, como los aspectos de atenticación, control de acceso y autorización.
- Orquestación. Módulo que proporcionará la coordinación e integración de los servicios que describen un proceso de negocio. Suelen ser scripts que definen el mapa de servicios, generalmente, en formato XML. Este componente será un motor de workflow para la interpretación en tiempo de ejecución de secuencias de servicios.
- Monitorización. Servicio que realizará la monitorización de todos los eventos necesarios desde el punto de vista de la seguridad.
- Persistencia. Elemento encargado de prestar soporte de almacenamiento para los componentes de la capa de aplicación.
- Notificación. Módulo empleado para la notificación de eventos entre los componentes software distribuidos, particularmente empleado por el caso de uso *notificar sucesos* con el objetivo de informar a otros componentes suscritos a los servicios de las alertas encontradas.
- Despliegue. Servicio que dará soporte al despliegue físico de los componentes distribuidos que forman parte del sistema.
- Registro y descubrimiento. Unidad que aportará la funcionalidad básica de la arquitectura orientada a servicios analizada en el apartado anterior. Concretamente, permitirá que los componentes publiquen sus propios servicios y busquen, descubran y consuman otros.
- Reglas. Este servicio permitirá el establecimiento de las políticas y estrategia global de la organización. Además, se empleará para definir configuraciones de los distintos componentes.

- Transacciones. Módulo encargado del soporte de transacciones y de la coordinación de las actividades realizadas por los componentes distribuidos. Suele haber dos tipos de coordinación: transacción atómica para operaciones individuales y de negocios para grandes transacciones.
- Gestión. Componente para la administración de los elementos que componen la capa de servicios.

En cuanto a la capa de aplicación, emplea una arquitectura de n-niveles de distribución que define la responsabilidad de cada nivel, representa distribuciones lógicas del modelo, proporciona patrones de uso general y describe cómo se mapean distintos componentes en un sistema de computación distribuido. Concretamente, esta capa está formada por cuatro niveles: usuario, encargado de la presentación de la información a los clientes; acceso, responsable de controlar e intermediar en el acceso a los servicios de manera segura; negocio, donde se localizarán todos los elementos que componen el núcleo principal de las aplicaciones, en este caso del sistema de detección; recursos, cuya función es conectar las aplicaciones con los recursos disponibles como almacenamiento y sistemas heredados.

Es necesario tener en cuenta que un nivel lógico de distribución se puede desplegar sobre uno o más sistemas o nodos físicos. Inversamente, varios niveles también se pueden desplegar en un único sistema. Por lo tanto, la separación en varios niveles lógicos no condiciona el despliegue del sistema de detección sobre uno o múltiples servidores físicos.

El primer nivel establece los distintos puntos de entrada o interfaces de interacción con el sistema. En este sentido, se distinguen dos tipos de entidades o elementos externos diferentes: por una parte los usuarios que realizarán un uso interactivo del sistema y, por otro lado, otros sistemas o subsistemas que interactuarán con el éste de manera automática siguiendo estándares bien definidos.

El nivel de acceso, dado que existen dos elementos de entrada distintos, proporciona mecanismos de seguridad diferentes para cada uno de los tipos de entidades. De esta manera, la autenticación de los usuarios se puede basar en aspectos biométricos o de posesión (llave o tarjeta), mientras que los sistemas emplearán mecanismos de clave. Por otra parte, los usuarios pueden emplear autenticación simple mientras los sistemas deberían utilizar protocolos más complejos como Kerberos o RADIUS. Los aspectos de autorización también presentan diferencias, mientras que los usuarios podrán realizar tareas de gestión (los que estén autorizados), los sistemas sólo pueden consumir servicios.

Los dos tipos de entidades externas no sólo acceden al sistema a través de mecanismos de seguridad distintos, sino que emplean patrones de comportamiento o interacción diferentes. Mientras los usuarios interaccionan con el sistema según el patrón modelo-vista-controlador (MVC —Model View Cotroller) (Fowler, 2002), los sistemas se comunican con éste a través del modelo orientado a servicios o SOA.

Dado que estos dos patrones constituyen la principal forma de comunicación del sistema, vamos a analizarlos con mayor nivel de detalle. La figura 5.5 muestra la arquitectura del patrón MVC empleado para la interacción del usuario.

En nuestra arquitectura el *nivel de negocio* es el *modelo*, el *controlador de vista* realiza las funciones de la *vista* del patrón y el *coordinador de trabajo* ejerce de *controlador* entre los dos anteriores, cumpliendo el objetivo de desacoplar vista y modelo. Este patrón permite separar responsabilidades entre los componentes, lo que ayuda a la escalabilidad y la reutilización.

El coordinador de trabajo o *controlador* permitirá controlar el acceso y personalizar la presentación en función del usuario o el rol, para lo cual, empleará el servicio de perfiles. Es responsable de la navegación del usuario para lo que recogerá y almacenará temporalmente información sobre múltiples presentaciones.

Realiza las funciones de controlador del patrón MVC basándose a su vez en dos patrones: *Intercepting Filter* y *Service to Worker*.

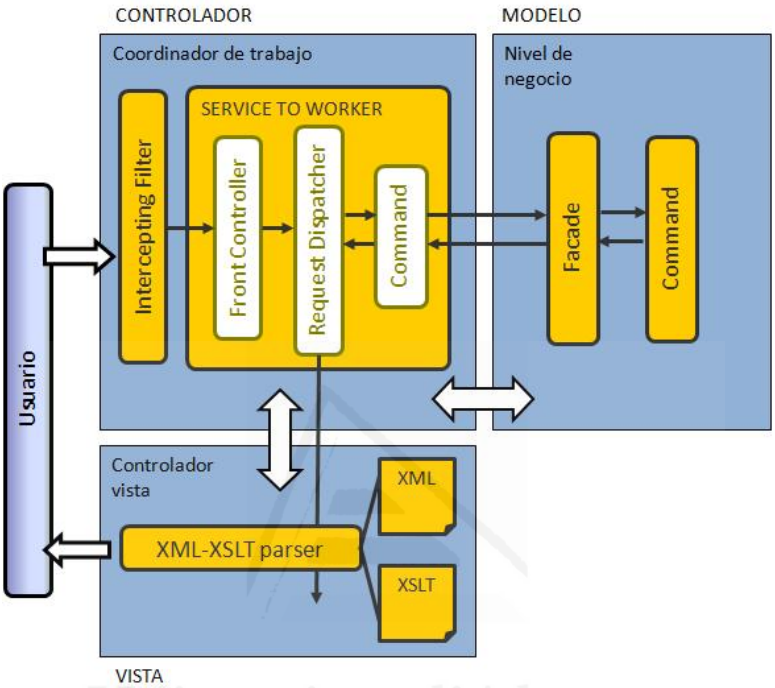


Figura 5.5. Patrón de interacción MVC.

El patrón interceptor permite recoger la petición del usuario y aplicarle un conjunto de filtros. Si se acepta la petición, se pasa al *service to worker* que identificará la operación a realizar, invocará el servicio de negocio asociado y controlará el flujo a la siguiente vista. Este patrón combina un conjunto de patrones más pequeños ofreciendo una solución flexible para cumplir con los requerimientos de un controlador: *Front Controller,* como punto central que gestiona las peticiones; *Request Dispatcher,* selecciona la vista y la acción a realizar; finalmente, *Command* se encarga de encapsular la información de la petición y enviarla al modelo.

El *modelo*, en este caso el nivel de negocio, representa los datos e implementa la lógica que maneja dichos datos. El modelo se

basará en el empleo de dos tipos de patrones: *Facade* y *Command*. El primero ofrecerá una interfaz común para la invocación de los distintos servicios y permitirá desacoplar el modelo de la vista y el controlador haciendo transparente los detalles de implementación. En este caso, el patrón *Facade* será utilizado por el *Controlador de Actividad*. El segundo encapsulará la lógica de la aplicación y será implementado por los componentes del nivel de negocio que ofrecen sus servicios.

El *Controlador de Actividad* es el encargado de presentar la funcionalidad del caso de uso de mayor nivel y de ejecutar la secuencia de procesos de negocio involucrados. Al existir un servicio de orquestación en la arquitectura, el controlador básicamente recuperará una descripción XML que define la secuencia de servicios que integran el caso de uso y se lo pasará al motor de orquestación. Como se ha comentado, este componente empleará un interfaz común, independientemente del conjunto de acciones y opciones que se vayan a desplegar, implementando el patrón de diseño *Facade* (Gamma *et al.*, 1995).

La *vista*, en este caso el *controlador de vista*, realiza la función de interfaz de usuario mostrando los datos procedentes del modelo, es responsable de la presentación y dependiente del dispositivo físico y la tecnología empleada. Este componente reduce el procesamiento en el nivel de negocio y disminuye la sobrecarga de la comunicación. El uso de XML/XSLT permite separar los datos de la presentación e independizarla, en la medida de lo posible, de la tecnología.

Por otra parte, los sistemas o subsistemas externos se comunicarán con el modelo siguiendo el patrón de interacción SOA, cuya arquitectura se muestra en la figura 5.6.

Anteriormente se ha analizado la arquitectura básica SOA tanto desde el punto de vista estructural con sus componentes proveedor, consumidor y registro, como desde el lado de su funcionamiento con las etapas de publicación, búsqueda, descubrimiento y consumo. Vamos a analizar ahora los elementos intermedios que intervienen en la interacción entre el

proveedor y el consumidor. Como se observa en la figura el sistema provee los servicios de detección, correlación e integración, mientras que los consumidores potenciales pueden ser otros servicios de correlación, integración y respuesta.

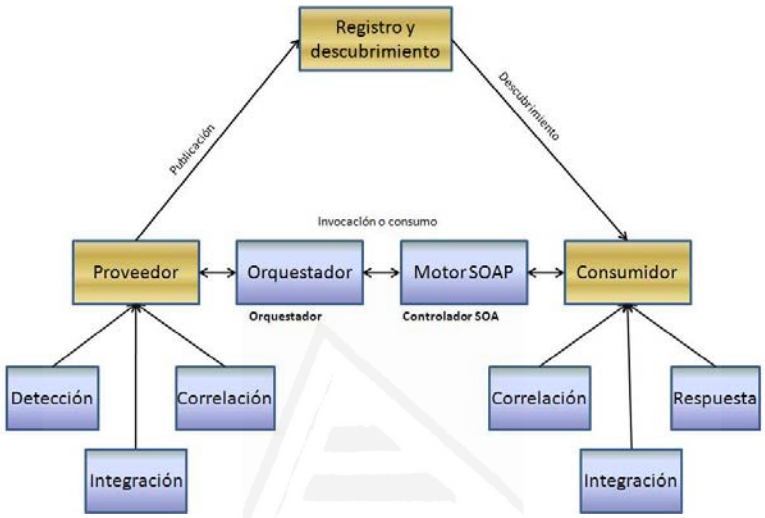


Figura 5.6. Modelo de interacción SOA.

Cuando un consumidor invoca un servicio lo hace a través de SOAP, por lo que el primer elemento intermedio en la arquitectura será un *motor SOAP*, en nuestro caso estará integrado en el *controlador SOA*. Éste analiza y verifica el mensaje SOAP, realiza funciones de seguridad y control de acceso, y envía la petición de servicio al nivel de negocio. Por otra parte, es responsable de remitir las respuestas de los servicios a los clientes.

La petición de servicio es recibida por el orquestador que la evalúa para ver si se trata del consumo de un único servicio o de un caso de uso de mayor nivel y, por lo tanto, de una integración de servicios. En el primer caso, enviará la petición al servicio o proveedor correspondiente. En el segundo, dado que se trata de un proceso verdaderamente orquestado, obtendrá la descripción XML asociada con dicha integración y se la pasará al servicio de orquestación para que la ejecute.

Los procesos de negocio son unidades o componentes que llevan a cabo la funcionalidad soportada por el sistema de detección de intrusos. Estos procesos pueden ser nuevos o encapsular aplicaciones heredadas a través de un adaptador de aplicación. En este sentido, el nivel de negocio estará compuesto principalmente por aquéllos componentes que serán responsables de la realización de las distintas etapas del modelo de detección presentado en el capítulo 3: detección, correlación, integración, respuesta y componentes auxiliares de soporte a los cuatro anteriores.

Los componentes de entidad representan unidades de información o datos que son necesarios para llevar a cabo las funciones de negocio. En este sentido, el elemento ataques y escenarios se comportará como un componente de entidad que contiene, entre otros datos, las firmas.

Por otra parte, los procesos de negocio se comportarán según el patrón Observer (Gamma *et al.*, 1995) que permite que un componente cliente se suscriba a un componente observable y quede a la espera de recibir notificaciones de información. Por ejemplo, un elemento de correlación se suscribirá a varios de detección, posteriormente le serán notificadas las alertas que se produzcan.

Un objetivo de la arquitectura es aislar la funcionalidad del sistema de detección de los aspectos específicos de cómo estos procesos y datos son implementados y almacenados. Esto se lleva a cabo mediante adaptadores que actúan como niveles de aislamiento especiales. En el caso de las aplicaciones heredades se emplearán adaptadores de aplicación, mientras que en el caso de los datos se utilizarán adaptadores de recursos. Estos adaptadores se comportarán, como es lógico, según el patrón de diseño Adapter o Proxy (Gamma *et al.*, 1995).

Después de analizar todos los componentes que forman el modelo conceptual de la arquitectura, se van a describir con mayor nivel de detalle los distintos componentes del nivel de negocio por

corresponder éstos con los módulos principales del sistema de detección.

La figura 5.7 muestra un diagrama con las distintas clases que forman los componentes del nivel de negocio. El diagrama permite observar la parte estática de la arquitectura y se corresponde con las vistas de diseño y procesos.

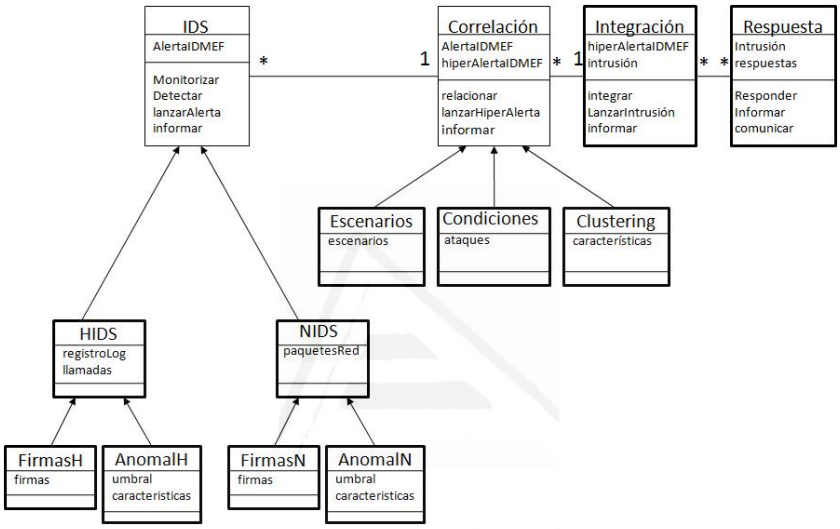


Figura 5.7. Diagrama de las clases principales del modelo.

El componente de detección se realizará mediante la clase genérica IDS. Esta clase presenta un primer nivel de especialización distinguiendo entre IDS de host y de red. A su vez, estas clases tendrán un segundo nivel de especialización según realicen el análisis mediante anomalías o a través de firmas. Se puede observar que existe una relación de dependencia uno a muchos entre la clase Correlación e IDS, ya que el componente de correlación relacionará muchas alertas en una única hiper-alerta.

La clase de correlación se especializará en tres clases correspondientes a los tres tipos de métodos de correlación existentes: especificación completa de patrones de escenarios, definición de prerrequisitos y consecuencias de los ataques

individuales y aquéllas técnicas que no emplean ningún tipo de información denominadas de clustering. Como en el caso anterior, existe una relación de dependencia uno a muchos entre las clase Integración y Correlación, debido a que se integrarán las hiper-alertas procedentes de distintos métodos de correlación.

Las clases Integración y Respuesta realizarán la funcionalidad correspondiente a las etapas de mismo nombre definidas en el modelo. Presentan una relación de dependencia muchos a muchos porque una misma intrusión puede desencadenar varias respuestas y una misma respuesta se puede dar en muchas intrusiones.

El diagrama de clases de los elementos del nivel de negocio permite describir la visión estática de los componentes principales del DIDS. Con el objetivo de aportar una perspectiva más completa, la figura 5.8 muestra la dinámica de estos elementos mediante un diagrama de secuencia.

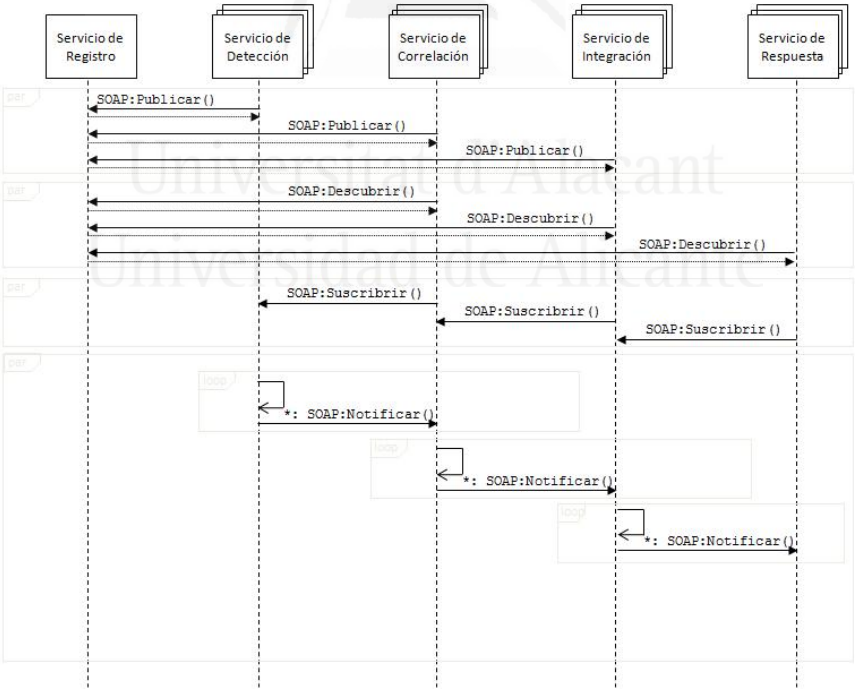


Figura 5.8. Dinámica de los componentes principales.

Dado que los distintos componentes van a ofrecer su funcionalidad mediante un paradigma orientado a servicios, la primera fase consiste en la publicación por parte de los elementos del sistema de los servicios que soportan. De esta manera, los proveedores de servicios, concretamente, las distintas entidades de detección, correlación e integración, publicarán la descripción WSDL de sus servicios en el servidor de registro y descubrimiento.

La segunda etapa la llevan a cabo los consumidores, en este caso los distintos componentes de correlación, integración y respuesta. Esta fase consiste en la búsqueda y descubrimiento de los servicios que se ofrecen en el DIDS, para ello los tres tipos de elementos consultarán al servidor de descubrimiento y obtendrán la descripción WSDL de los servicios en los que estén interesados.

Después de conseguir las descripciones WSDL, los consumidores están en condiciones de invocar o consumir los servicios ofrecidos por los proveedores. En este caso consistirá en la suscripción para el envío y la recepción de alertas, generadas por los elementos de la etapa anterior del modelo general de detección. Así, los componentes de correlación se suscriben a los de detección, los de integración a los de correlación y los de respuesta a los de integración. Nótese que las entidades de correlación e integración se comportan simultáneamente como proveedor y consumidor.

Finalmente, una vez publicados, descubiertos e invocados los servicios, la última etapa es la operación normal del DIDS. Los distintos servicios de detección estarán continuamente monitorizando los elementos a su cargo y realizando el proceso de detección. En el caso de detectar alguna alerta la enviarán o notificarán a todos los servicios de correlación que se hayan suscrito. Éstos, que estarán siempre esperando recibir alertas, la recogerán, procesarán y, en caso de correlacionar algún escenario, enviarán la hiper-alerta correspondiente a los servicios de integración suscritos. Los componentes de integración integrarán la hiper-alerta recibida con otras anteriores y, si encuentran un escenario completo, notificarán a los servicios de

respuesta suscritos que llevarán a cabo las acciones de réplica definidas para devolver al sistema a un estado seguro.

Modelo Físico

El modelo físico de la arquitectura de un sistema se suele mostrar mediante las vistas de implementación y despliegue. El diagrama de componentes expresa la vista de implementación y comprende los componentes que se utilizan para ensamblar y hacer disponible el sistema físico (Booch y Rumbaugh, 2001). Un componente es una parte física y reemplazable de un sistema que exporta un conjunto de interfaces y proporciona la realización de esas interfaces. La figura 5.9 muestra los componentes físicos principales del nivel de negocio del modelo conceptual de la arquitectura.

Los componentes *detectorH* y *detectorN* empaquetan físicamente las clases que llevan a cabo la etapa de detección en los nodos y dispositivos de red respectivamente. Existe una relación de dependencia entre los componentes anteriores y las clases HIDS y NIDS, debido a que los componentes son la implementación física de las clases. También presentan dependencia del archivo de firmas para los IDS del tipo abusos.

El componente *correlación* constituye la implementación física de las clases que realizan dicha etapa del modelo general. En este caso, el componente presenta un conjunto más extenso de dependencias; depende de las clases *escenarios*, *condiciones* y *clustering* por ser las que realiza, de los archivos *escenarios* y *ataques* empleados en varios de los métodos de correlación y de los elementos de detección por ser el origen de las alertas.

Los componentes *integración* y *respuesta* encapsulan las clases de mismo nombre que efectúan la etapa de integración de escenarios y ejecutan la respuesta a las intrusiones detectadas. El segundo presenta una dependencia respecto al primero que, al mismo tiempo, depende del elemento de correlación.

Finalmente, dado que la arquitectura propuesta en el presente trabajo se basa en componentes software distribuidos que ofrecen su funcionalidad bajo un paradigma orientado a servicios, es necesario que los componentes registren los servicios que ofrecen, para lo que se empleará el componente *registro*, del cual dependerán el resto y que a su vez depende de los archivos WSDL que describen estos servicios.

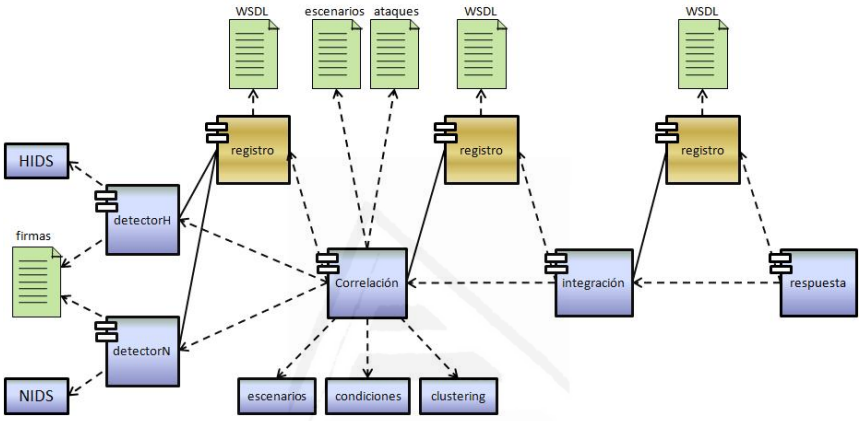


Figura 5.9. Diagrama de componentes principales.

Contenedor de Componentes

Una vez analizado los componentes que constituyen el nucleo del sistema y antes de abordar el despliegue físico de la red, vamos a analizar el elemento principal de la arquitectura física: el contenedor de componentes. La figura 5.10 muestra la estructura interna de un contenedor de componentes software distribuidos.

La estructura interna del contenedor sigue el esquema básico de la arquitectura conceptual empleada. Por esta razón, se distinguen dos capas diferenciadas: servicios y componentes. Todos los contenedores deben tener un conjunto de servicios necesarios o genéricos que son, en principio, aquéllos identificados en la capa de servicios de la arquitectura. Además pueden proveer de otros concretos como, por ejemplo, servicios para acceder a bases de datos o para comunicarse con componentes CORBA.

Como se puede observar en la figura, la capa de componentes también estará formada por componentes genéricos y concretos. En cuanto a los genéricos, se identifican tres tipos distintos: *Entidad*, que serán empleados para el almacenamiento de datos; *Sesión*, que proporcionarán un patrón común reutilizable a los componentes controladores y coordinadores concretos; y *Auxiliar*, que encapsulan la lógica de negocio que será compartida y empleada por los componentes concretos. Éstos últimos serán aquéllos que pertenecen al modelo de detección.

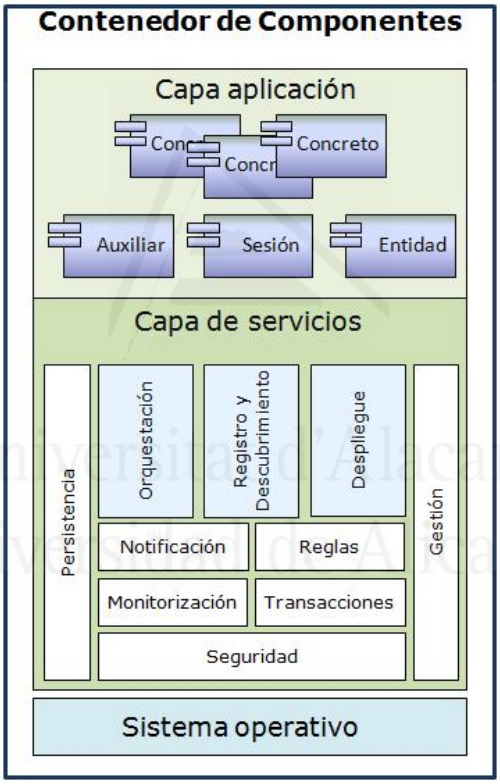


Figura 5.10. Estructura de un contenedor de componentes.

Si bien los servicios y componentes genéricos estarán presentes en todos los contenedores, y por tanto no permitirán diferenciarlos, componentes concretos permitirán caracterizar al contenedor. En este sentido, los elementos concretos dotarán al

contenedor de la capacidad de especializarse y, por lo tanto, de la posibilidad de tener distintos tipos de contenedores.

En nuestro caso vamos a distinguir cuatro tipos de contenedores distintos que coinciden con las cuatro etapas del modelo general de detección (figura 5.11): contenedor de detección, correlación, integración y respuesta.

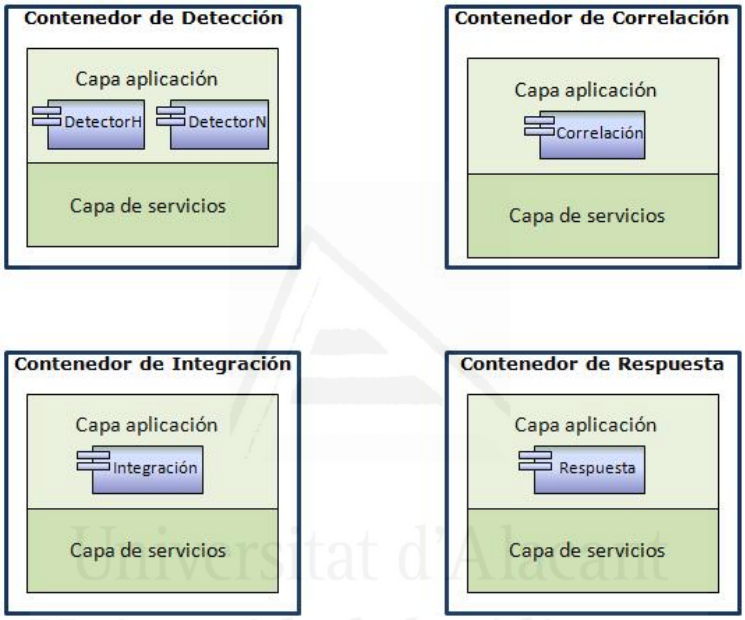


Figura 5.11. Tipos de contenedores especializados.

El contenedor de detección está especializado en la etapa de mismo nombre del modelo y puede tener tantos componentes concretos como mecanismos de detección existen. En este caso, nótese que se podría especializar todavía más, concretamente, en un contenedor de detección de host y otro de red. El contenedor de correlación también puede tener tantos componentes como métodos distintos de correlación de alertas. Los contenedores de integración y respuesta llevarán a cabo las dos últimas etapas del modelo y están formados por componentes únicos.

El hecho de tener contenedores independientes especializados en cada una de las etapas posibilita el despliegue físico del modelo

general de detección de manera modular. Por ejemplo, se puede realizar sólo la función de detección desplegando únicamente componentes de detección. Si más adelante se decide correlacionar alertas, bastará con conectar a la red un contenedor de correlación que descubrirá mediante SOA los servicios de detección y los consumirá relacionando sus alertas.

Finalmente, después de describir los distintos componentes físicos de los que consta el sistema de detección y la estructura interna de los tipos de contenedores que alojarán dichos componentes, vamos a abordar el despliegue físico de la red.

La distribución de estos contenedores de componentes por la red es lo que constituye el despliegue físico que muestra la configuración de los nodos y de los componentes que residen en ellos. El empleo de contenedores especializados e independientes permite establecer distintas configuraciones distribuidas del sistema de detección. La figura 5.12 muestra una posible arquitectura distribuida jerárquica.

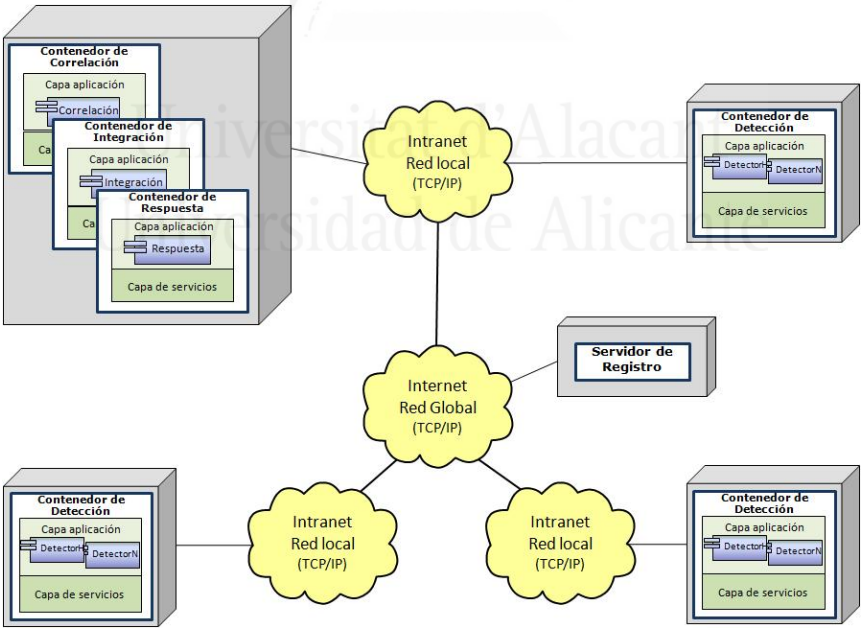


Figura 5.12. Diagrama desplegué con arquitectura jerárquica.

La figura muestra el despliegue de un dominio de red con tres sitios o redes locales interconectadas a través de Internet. Como se puede observar, se han desplegado contenedores de detección por las tres redes locales. En cambio, sólo se han empleado contenedores de correlación, integración y respuesta en una de las redes que actuará como red raíz de la jerarquía. El nodo con los tres contenedores de la red raíz recibirá las alertas de detección generadas en todas las redes locales y llevará a cabo las funciones de correlación, integración y respuesta de manera centralizada jerárquica.

Esta arquitectura lleva a cabo un proceso de análisis completo de grano grueso a nivel de toda la red del dominio, en cambio ejecuta sólo el proceso de detección de grano fino en cada una de sus tres redes locales. Sin embargo, esta arquitectura adolece de dos problemas principales: escalabilidad y tolerancia a fallos. Escalabilidad porque al realizar el análisis principal de manera centralizada en la raíz de la jerarquía, en redes con mucha actividad maliciosa o simplemente si se añaden más redes locales al dominio, se sobrecarga o satura el proceso de análisis. Tolerancia a fallos debido a que presenta el problema del único punto de fallo, por lo que si un atacante compromete la raíz del sistema de detección, toda la red quedará desprotegida, aspecto crucial desde el punto de vista de la seguridad.

Con el objetivo de solventar los problemas asociados a la arquitectura jerárquica anterior, de entre todas las posibles, nuestra propuesta consiste en utilizar los contenedores de componentes especializados dentro de una arquitectura completamente distribuida. Este enfoque aumenta la escalabilidad, adaptabilidad, flexibilidad y tolerancia a fallos del sistema de detección. La figura 5.13 muestra el despliegue físico de la red para el mismo dominio que en caso anterior.

Se han desplegado los cuatro tipos de contenedores especializados en cada una de las redes locales del dominio, de esta manera cada red local realizará todas las funciones de detección del modelo general de forma completamente autónoma y distribuida. En este sentido, se lleva a cabo el análisis completo

de grano fino en cada subred, no sólo la detección como en el caso anterior. Además, si se quiere tener una visión global bastará con establecer una red maestra y las otras esclavas.

Desde el punto de vista de la escalabilidad, si se añade una nueva subred al dominio sólo tendremos que desplegar en ella los mismos contenedores que el resto de redes, sin que su funcionamiento afecte al resto. Con respecto a la tolerancia a fallos, si una red local es comprometida quedará desprotegida sólo ésta, el resto continuarán funcionado perfectamente por no presentar el problema del punto de fallo único.

La arquitectura distribuida es muy adaptable y nada impide añadir en cada intranet nuevos contenedores de componentes especializados en la función que se pretenda potenciar. Por ejemplo, se podrían incluir más contenedores de detección en el caso de una intranet con muchos nodos, sitios con mucho tráfico o redes de alta velocidad e, incluso, más de un contenedor de correlación si se producen un número excesivo de alertas.

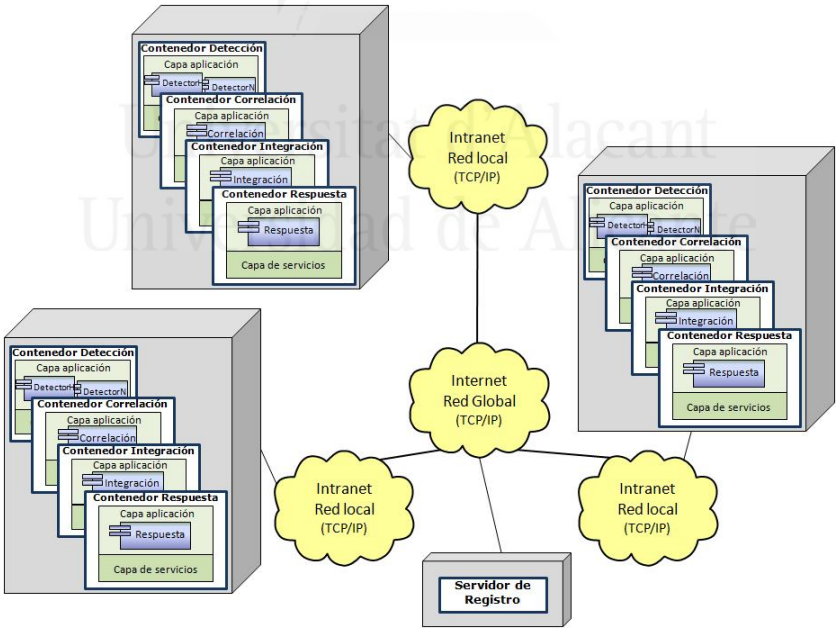


Figura 5.13. Arquitectura completamente distribuida propuesta.

Esta arquitectura presentada en este capítulo es un fiel reflejo del enfoque general del modelo de detección, pero centrado en las arquitecturas orientadas a servicios. Este enfoque favorece la implantación de DIDS basados en una red de sensores o contenedores distribuidos ya que permite definir los diferentes componentes involucrados como la composición ordenada de actividades o funciones del modelo, cada una de las cuales es ofrecida por un servicio SOA.

Dentro de las arquitecturas completamente distribuidas de los DIDS se han utilizado distintos mecanismos o tecnologías, pero en la mayoría de los casos la relación entre los componentes del sistema se establece de forma explícita en el diseño, como por ejemplo una relación de vecindad (Arora *et al.*, 2004) y (Vlachos *et al.*, 2004). Al contrario que los enfoques anteriores, la arquitectura totalmente distribuida propuesta es mucho más flexible ya que permite establecer la relación entre los servicios en tiempo de ejecución, gracias a la característica de interoperabilidad de la arquitectura SOA que le aportan los mecanismos de publicación, descubrimiento y consumo.

Capítulo 6

Evaluación y Validación

Llegados a este punto, se ha alcanzado el objetivo general de la investigación, en el que se planteaba la creación de un modelo general y formalmente definido que permitiera la integración de múltiples técnicas de correlación, además del establecimiento de un método de integración que mejorara el rendimiento del sistema basado en la ponderación de los métodos de correlación según la cantidad de información que éstos proporcionan en función de una medida objetiva de calidad. Además, se ha propuesto una arquitectura distribuida que hace viable el despliegue del modelo en entornos de producción reales y con las tecnologías actuales. No obstante, falta una de las tareas más importantes, la validación del modelo comprobando que se comporta como se espera que lo hiciera.

Para llevar a cabo esta tarea nos apoyamos en el método de experimentación que nos permitirá probar el modelo y analizar los resultados obtenidos para demostrar su validez. En este sentido, este capítulo aborda en primer lugar el diseño de los experimentos que permitan avalar las hipótesis de partida, en segundo lugar su realización o implementación y, finalmente, el análisis y evaluación de las pruebas llevadas a cabo que muestran los beneficios de la propuesta.

Diseño de Experimentos

Es necesario diseñar experimentos que validen las capacidades del modelo en el cumplimiento de los objetivos e hipótesis planteados en el presente trabajo. Tales pruebas deben mostrar principalmente que la solución planteada es capaz de generalizar la integración de cualquier número de métodos de correlación, mejorar el rendimiento obtenido por cualquiera de las técnicas involucradas en dicha integración y maximizar el resultado de la integración aprovechando el conocimiento adquirido sobre la eficiencia de los métodos que la componen.

Para medir las capacidades del modelo como método de integración general se debe diseñar un experimento que permita demostrar que el sistema es capaz de integrar de la misma manera dos y más métodos de correlación. En este sentido, se seguirá una metodología de demostración inductiva.

Teniendo en cuenta lo anterior los experimentos diseñados para la validación de la generalidad del método de integración se llevan a cabo según el diagrama de actividad de la figura 6.1.

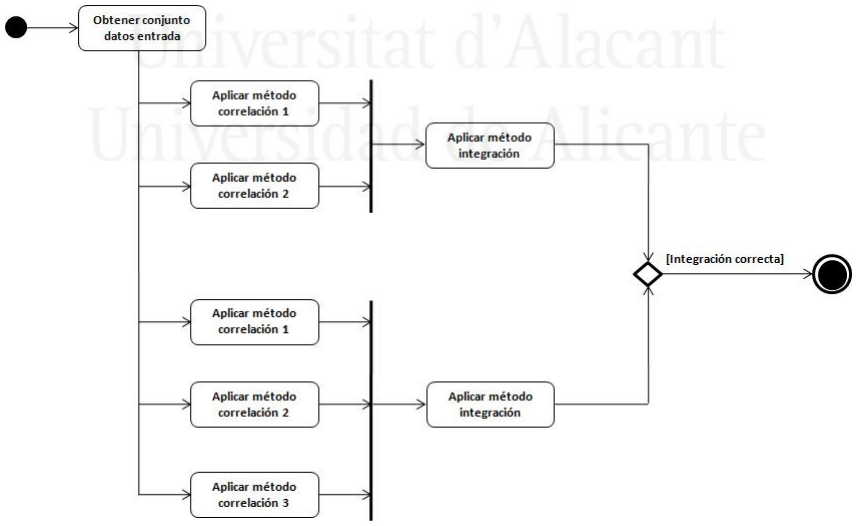


Figura 6.1. Diagrama de actividad del experimento 1.

En primer lugar se obtiene el conjunto de datos de entrada que, en este caso, serán las alertas generadas por los elementos de detección en la etapa de percepción del modelo. En segundo lugar, aplicar en paralelo dos métodos de correlación distintos y, posteriormente, ejecutar el método de integración general empleando como entrada el resultado de los dos métodos anteriores. En tercer lugar, ejecutar en paralelo tres métodos de correlación distintos y llevar a cabo el proceso de integración general sobre el resultado de los tres métodos.

Existe un conjunto finito de tipos de métodos de correlación que pueden ser representados, que constituyen particiones finitas del conjunto de técnicas de correlación. Se elige un representante de cada clase y se realizan pruebas con combinaciones de dichos representantes. Si el método integra satisfactoriamente cualquier combinación, se puede inferir la capacidad del modelo a la hora de generalizar la integración.

Por otra parte, para validar que el método de integración obtiene mejor rendimiento que cada una de las técnicas empleadas en la integración, bastará con una simple comparación de los resultados logrados por el método de integración con respecto a cada una de las técnicas de correlación. La figura 6.2 muestra el diagrama de actividad del experimento.

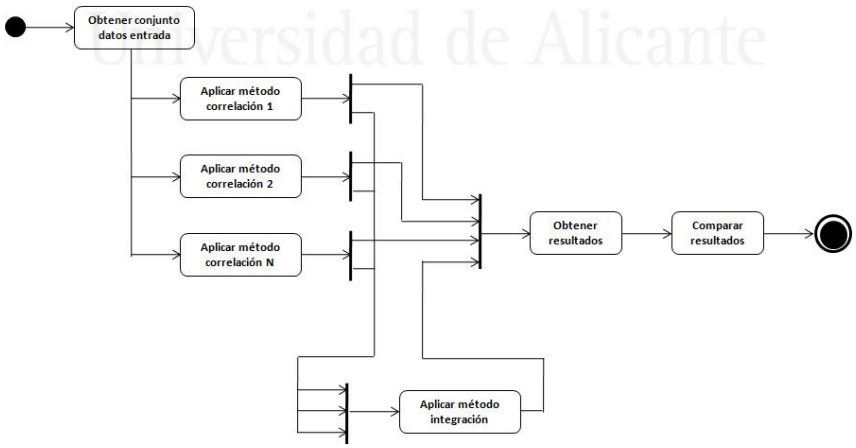


Figura 6.2. Diagrama de actividad del experimento 2.

Primero se obtienen los mismos datos de entrada del experimento anterior y se aplican en paralelo cada uno de los N métodos de correlación distintos. Después se lleva a cabo la integración de las hiperalertas de salida de los métodos de la etapa de correlación. En tercer lugar se obtienen los resultados tanto del proceso de integración como de cada uno de los métodos que lo componen y, finalmente, se comparan esperando que la función de integración haya obtenido el mejor rendimiento.

Por último, con el objetivo de mostrar que el modelo tiene la habilidad de mejorar el resultado de la integración aprovechando el conocimiento adquirido sobre la eficiencia de los métodos que la componen, también se empleará una comparación de distintos resultados. Por un lado, una etapa de integración que se beneficiará de la ordenación o ponderación de los métodos subyacentes en función del rendimiento de cada uno de ellos y, por otro lado, un método de integración sin ponderar que no tiene en cuenta ninguna información y trata a todos los métodos de correlación por igual, independientemente de sus capacidades. La figura 6.3 muestra el diagrama de actividad del experimento.

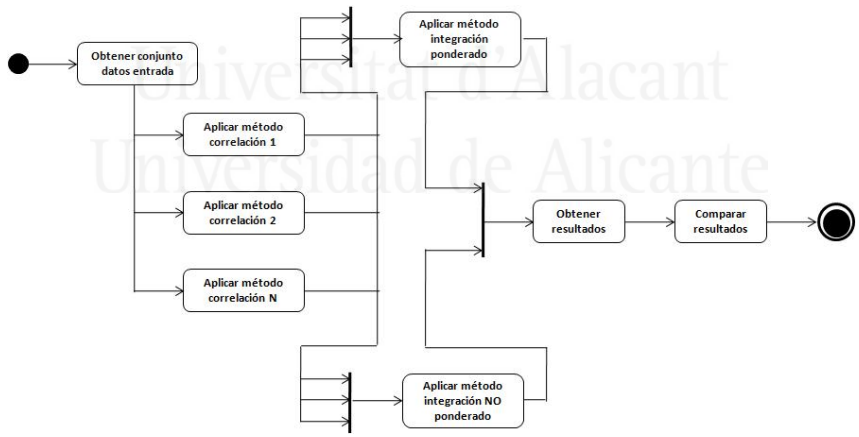


Figura 6.3. Diagrama de actividad del experimento 3.

Como se puede observar en la figura 6.3, después de aplicar los distintos métodos de correlación, se llevan a cabo dos variantes del método de integración distintas: una ponderada en función de

la eficiencia o calidad de cada uno de los métodos de correlación participantes y otra sin ponderar. Después se obtienen los resultados alcanzados por cada una de las dos variantes y se comparan para comprobar si la vertiente ponderada ha logrado mejores resultados que su homónima sin ponderar.

Implementación de los Escenarios

Tras definir el diseño de los experimentos que se llevarán a cabo para mostrar las capacidades del modelo general de detección de cara a resolver la problemática planteada, se aborda en el presente apartado la realización, es decir la tecnología empleada en la implementación de los escenarios de prueba de los experimentos.

El escenario de desarrollo debe contar con módulos pertenecientes a cada una de las etapas del modelo general. Además, se debe tener en cuenta que, dado que la solución planteada consiste principalmente en la integración de múltiples métodos de correlación, en la fase de correlación debe haber varios métodos. En este sentido, tal como se ha comentado en el capítulo del estado del arte, existen tres tipos de técnicas distintas: especificación de escenarios, prerrequisitos y consecuencias y métodos sin información previa del dominio, por lo que en el escenario habrá al menos un módulo por cada uno de los tipos de enfoque.

La figura 6.4 muestra los módulos empleados en la realización o implementación de los distintos elementos que componen las etapas del modelo. La etapa de percepción se realizará mediante el sensor o IDS de código abierto *Snort*. La fase de correlación empleará un componente por cada tipo de método: *alertSTAT* como técnica de especificación de escenarios, una implementación propia del método de prerrequisitos y consecuencias (Ning *et al.*, 2002) y *EMERALD* como módulo estadístico o sin información previa del dominio. La integración se llevará a cabo utilizando un módulo que implementa el algoritmo de la red neuronal GNG modificada que se ha diseñado.

Finalmente, la respuesta consistirá en un componente que ejecutará pequeñas acciones definidas en un archivo de reglas. Nótese que, dado que *Snort* y *alertSTAT* son sistemas heredados, se accede a ellos a través de un adaptador o proxy que encapsulará su funcionalidad.

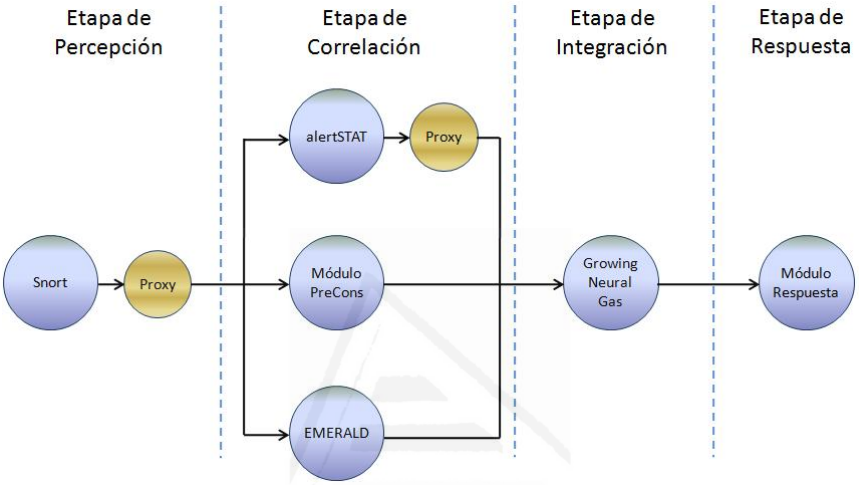


Figura 6.4. Módulos de la implementación del modelo.

Snort es un sistema de detección de intrusos de código abierto que se ha convertido en el estándar de facto en IDS. Combina los beneficios del análisis de firmas, de anomalías y de protocolos y constituye la tecnología de IDS más extendida del mundo.

El sistema tiene una base de datos de ataques con miles de firmas o patrones que se actualiza constantemente a través de Internet. Además, uno de los aspectos más destacables de *Snort* es que incorpora un lenguaje de creación de reglas o firmas flexible, potente y sencillo. Empleando este lenguaje se pueden añadir nuevos patrones o modificar los existentes para ajustarlos lo máximo posible a nuestra topología de red con el objetivo de aumentar la detección y reducir los falsos positivos.

Otra característica destacable de *Snort* es la capacidad de almacenar las alertas encontradas en una gran variedad de formatos de salida, desde archivos de texto o formatos binarios

como *tcpdump* hasta una enorme variedad de bases de datos como MySQL, Oracle o SQL Server. La flexibilidad a la hora de poder utilizar tantos formatos de salida convierten a Snort en un sensor ideal para conectarse a herramientas de seguridad de mayor nivel de abstracción como OSSIM o Prelude.

AlertSTAT es un paquete de software desarrollado por la Universidad de California que permite realizar correlación de alertas siguiendo el método de especificación de escenarios (Eckmann *et al.*, 2002). El sistema incorpora, para la especificación de los escenarios, un lenguaje de descripción de ataques basado en técnicas de transición entre estados denominado STATL. Este lenguaje tiene la característica de ser extensible, lo que posibilita la definición de nuevos tipos e instrucciones y, por lo tanto, la adaptación a una gran variedad de entornos y plataformas.

Dado que el sistema funciona como una máquina de estados, los escenarios se definen como una secuencia de estados y transiciones entre dichos estados, existiendo un estado inicial y uno final. El proceso de detección comienza por el estado inicial y se considera detectado el escenario completo si se alcanza el estado final. En caso de detectar algún escenario se obtiene como salida su alerta en formato IDMEF-XML.

El *Módulo PreCons* se ha implementado según el método de prerrequisitos y consecuencias definido en (Ning *et al.*, 2002) y, más concretamente, en la versión extendida (Ning *et al.*, 2002b). La técnica consiste en definir los prerrequisitos y las consecuencias de los ataques y, posteriormente, el proceso de correlación consistirá en relacionar las consecuencias de un ataque anterior con los prerrequisitos de otro posterior.

La ventaja de la versión extendida es que especifica detalladamente los prerrequisitos y consecuencias utilizados para los ataques que forman parte de los escenarios del conjunto de datos de prueba DARPA 2000, datos empleados por los autores para validad su propuesta y que se han usado en la evaluación de esta tesis como se indica en el apartado siguiente.

El último módulo de la etapa de correlación se ha implementado según el método de correlación de alertas probabilístico definido en (Valdes y Skinner, 2001) y que se emplea en EMERALD, uno de los sistemas de detección de intrusos distribuido más conocidos mundialmente, desarrollado por el instituto de investigación SRI International y que se basa en una arquitectura jerárquica.

El algoritmo correlaciona las alertas basándose en la similitud entre sus atributos o características. Define una función de similitud apropiada entre los distintos atributos que devuelve valores entre 0 y 1. La similitud total de dos alertas es una media ponderada de la semejanza de sus atributos y se define un umbral de emparejamiento mínimo.

Durante el proceso, para cada nueva alerta se calcula la similitud total con respecto a cada una de las alertas existentes, con el objetivo de obtener la alerta más parecida a la nueva y el valor de dicha semejanza. Si el valor de similitud total máximo es mayor que el umbral de emparejamiento mínimo, la nueva alerta se une al clúster al que pertenece la alerta más parecida. En caso de no superar el umbral mínimo, la nueva alerta será la primera de un nuevo clúster. La interpretación es que las alertas pertenecientes al mismo grupo forman parte del mismo escenario.

El módulo EMERALD emplea como atributos, a partir de los cuales establecer la similitud total, los mismos que se utilizan en (Valdes y Skinner, 2001); concretamente: direcciones IP de origen y destino del ataque, puertos origen y destino, clase de ataque y la hora. En cuanto a la similitud entre distintas clases de ataque, los autores detallan una tabla con los porcentajes, por lo que se han empleado exactamente los mismos valores.

Como se ha discutido en un capítulo anterior, la etapa de integración se lleva a cabo implementando el algoritmo de la red neuronal Growing Neural Gas. Partiendo de las alertas correlacionadas por los distintos métodos de la etapa anterior, la red neuronal realiza un proceso de entrenamiento y clasificación que permite obtener nuevos clústers o grupos donde, al igual que

en el módulo de EMERALD, las alertas que pertenecen al mismo grupo forman parte del mismo escenario.

El algoritmo GNG implementado es una modificación del original donde, en las principales fases del aprendizaje de la red neuronal, se ha introducido un nuevo parámetro que pondera dicho aprendizaje en función de las capacidades del método de correlación origen de la alerta o patrón de entrenamiento. El nuevo parámetro es la métrica *Intrusion Detection Capability* o C_{ID} (Gu *et al.*, 2006) que permite medir de manera objetiva la calidad o eficiencia de cualquier sistema de detección.

Para que las pruebas sean homogéneas y, por tanto, lo más válidas posible, se han empleado como características de entrada a la red neuronal los mismos atributos utilizados por el método de correlación probabilístico, es decir, direcciones y puertos de origen y destino, clase de ataque y hora. Por otra parte, los parámetros de aprendizaje de la red han sido $\lambda = 2000$, $\varepsilon_1 = 0.1$, $\varepsilon_2 = 0.01$, $\alpha = 0.5$, $\beta = 0.005$.

Finalmente, el módulo de respuesta consultará un archivo de reglas donde se asocia las acciones a tomar por cada escenario detectado y llevará a cabo dichas acciones. Aunque, en las pruebas realizadas se ha limitado a emitir un informe.

En cuanto a las tecnologías empleadas para la realización del paradigma SOA, los elementos anteriores se han implementado como *Servicios Web*. Para la publicación y el descubrimiento de los servicios se ha empleado el protocolo UDDI y como mecanismo de comunicación entre clientes y servidores a la hora de proporcionar servicios se ha utilizado SOAP. La tabla 6.1 resume las tecnologías y herramientas SOA usadas en el desarrollo del escenario.

Aunque otras tecnologías de implementación SOA como WS-* permiten emplear herramientas más avanzadas como WS-Discovery para la publicación y el descubrimiento, WS-Eventing para el envío de los eventos o alertas, WS-Security para la seguridad del propio sistema o WS-Management para su gestión, se ha optado por WS Basic Profile 1.0 por su simplicidad y por

cumplir con los requerimientos necesarios para validar de la arquitectura propuesta.

Tabla 6.1. Tecnologías SOA empleadas.

Tecnología	Descripción
Web Services	Tecnología de implementación SOA
UDDI	Protocolo para la publicación y el descubrimiento de servicios
SOAP	Protocolo de comunicación entre clientes y servidores
jUDDI v0.9rc4	Servidor de registro empleado
gSOAP v2.7	Servidor de aplicaciones
MTOM	Mecanismo de transferencia de alertas

Experimentación y Análisis de Resultados

Después de diseñar los experimentos y una vez que se han analizado las distintas tecnologías que se han empleado en la realización o implementación del modelo general de detección, es momento de analizar los resultados obtenidos. No obstante, previamente es necesario realizar ciertas consideraciones sobre los datos empleados para llevar a cabo las pruebas.

Un problema común a la hora de evaluar sistemas de detección y correlación es la falta de fuentes de datos fiables que se puedan emplear como banco de pruebas en los experimentos. Idealmente, se necesitan datos tanto de red como de host, completamente etiquetados, con escenarios de ataque complejos descritos en

detalle y, principalmente, que estén disponibles libremente para la comunidad científica (Maggi y Zanero, 2007).

El único conjunto de datos con los requisitos anteriores realmente disponible es la serie de datos para evaluación de sistemas de detección IDEVAL perteneciente a la Agencia de Proyectos de Investigación Avanzados de la Defensa de los EEUU (DARPA —Defense Advanced Research Projects Agency) y disponibles libremente en el Instituto Tecnológico de Massachusetts (MIT —Massachusetts Institute of Technology).

Aunque este conjunto de datos presenta ciertas deficiencias (Mahoney y Chan, 2003), es el conjunto de datos empleado principalmente por la comunidad científica en la evaluación de sistemas de detección (Ning *et al.*, 2002).

Existe otro conjunto de datos denominado DEFCON 9 CTF, pero no están disponibles libremente para experimentación. Además, los ataques no están etiquetados y, por tanto, no pueden emplearse para llevar a cabo una evaluación apropiada (Maggi y Zanero, 2007).

Los experimentos se han realizado con el conjunto de datos específico para la evaluación de sistemas de correlación DARPA 2000. Estos datos contienen una secuencia de ataques pertenecientes a todos los tipos de ataques principales e incluyen todo el tráfico de red generado durante la realización del escenario.

El escenario lleva a cabo ataques de *reconocimiento* de red para conseguir los nodos y servicios activos en la red, después ataques de *remoto a local* (R2L —Remote to Local) y *usuario a root* (U2R —User to Root) para conseguir una cuenta de usuario con privilegios de administración, a continuación ataques para instalar en los nodos comprometidos una herramienta que permita ejecutar un ataque de *denegación de servicio distribuido* (DDOS —Distributed Denial Of Service) y, finalmente, se realiza efectivamente el ataque DDOS.

Dado que uno de los parámetros que suelen caracterizar y distinguir los distintos métodos de correlación es su capacidad para correlacionar ataques nuevos o desconocidos, se han añadido ataques nuevos al escenario con el objetivo de realizar dos tipos de pruebas. Por una parte el escenario DARPA tal cual, donde los ataques son conocidos y, por otra, el mismo escenario pero con ataques nuevos. Nótese que es imposible incluir en la especificación de un escenario (primer tipo de métodos) ataque sobre los que no se tiene ningún conocimiento y, de manera similar, no se pueden definir los prerrequisitos y consecuencias de un ataque desconocido.

El primer conjunto de pruebas llevadas a cabo ha consistido en la aplicación de cada uno de los métodos de correlación implementados sobre la secuencia de ataques conocidos, es decir, teniendo en cuenta únicamente el escenario DARPA. Después se ha realizado el proceso de integración general tanto de manera ponderada como sin ponderar sobre las hiperalertas obtenidas de los tres métodos de correlación. Finalmente, dado que existen propuestas que integran los métodos de prerrequisitos y consecuencias con las técnicas sin información previa (Ning *et al.*, 2004) y (Qin y Lee, 2004), se ha ejecutado la integración de las hiperalertas generadas sólo por estos dos métodos con el objetivo de incluir este enfoque en la comparativa.

Previamente a la ejecución de los distintos métodos de correlación, se ha llevado a cabo la etapa de percepción empleando el IDS o sensor *Snort* y utilizando como entrada el tráfico de red proporcionado por el conjunto de datos de prueba. La figura 6.5 muestra gráficamente los resultados obtenidos por cada uno de los métodos sobre el escenario conocido.

Los métodos de prerrequisitos y consecuencias, y probabilístico son los que peores resultados obtienen con curvas de detección que van desde el 40% y 50% hasta algo más del 80%. La integración de estos dos métodos mejora la eficiencia con curvas desde el 60% hasta el 90%. Por último, existen tres curvas de detección que coinciden y que muestran las mejores capacidades,

concretamente AlertSTAT, integración de los tres métodos sin ponderar e integración ponderada.

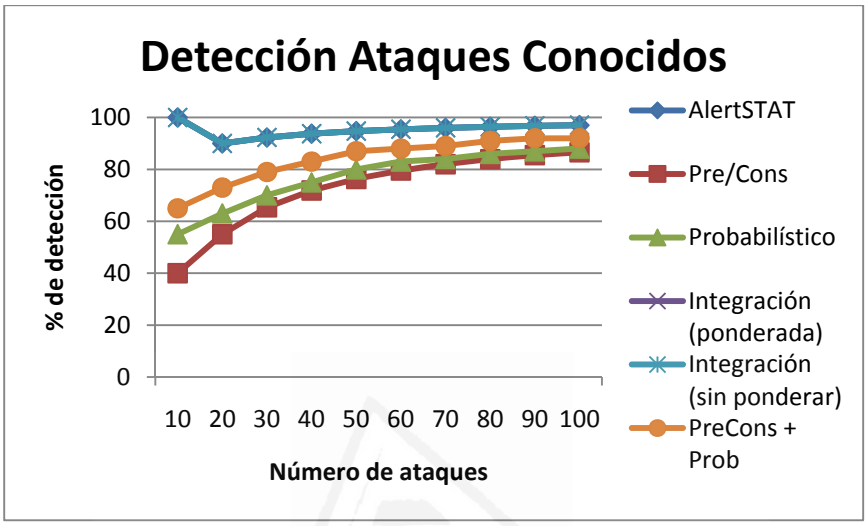


Figura 6.5. Resultado detección ataques conocidos.

Lo importante de las pruebas es que demuestra las capacidades del modelo a la hora de llevar a cabo la integración de múltiples métodos de manera general, en contra de las propuestas que integraban sólo dos técnicas de forma ad hoc. La misma función de integración se ha empleado para integrar prerrequisitos y consecuencias con el método probabilístico, y para integrar los dos anteriores con AlertSTAT.

Otro aspecto interesante es que el rendimiento o resultado de la integración en este caso no consigue mejorar la eficiencia de AlertSTAT. Esto es debido a que, dado que todos los ataques son conocidos y se han definido en la especificación del escenario, los otros dos métodos no pueden aportar ninguna información adicional que pueda mejorar la integración. Como se analiza a continuación, esto no ocurre cuando el escenario incluye ataques nuevos y, por lo tanto, no definidos en su especificación.

El segundo conjunto de pruebas llevadas a cabo han sido las mismas que en el caso anterior, pero sobre un escenario en el

cual se han añadido ataques desconocidos. La figura 6.6 muestra gráficamente los resultados obtenidos por cada uno de los métodos sobre el escenario DARPA con ataques desconocidos añadidos.

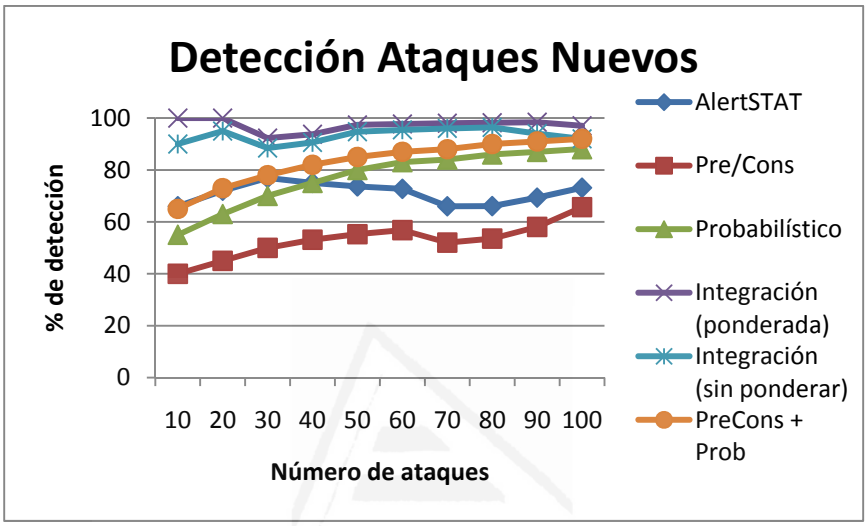


Figura 6.6. Resultado detección ataques desconocidos.

Lo primero que se observa en el gráfico es la pérdida de eficiencia de los métodos de correlación que emplean para su funcionamiento información o conocimiento del dominio de aplicación, concretamente AlertSTAT y prerrequisitos y consecuencias. Pre-Cons obtiene rendimientos más o menos en el rango del 40% al 60%, inferiores a los conseguidos en el primer caso. AlertSTAT presenta una curva de detección que se mueve alrededor del 70%, cuando en el primer caso estaba por encima del 90%, resultados mucho peores que con ataques conocidos. Es lógico que AlertSTAT sufra mayor deterioro en su rendimiento que Pre-Cons ya que necesita para su funcionamiento más información sobre los ataques.

Otro aspecto interesante es comprobar que el método probabilístico, dado que no necesita ningún tipo de información sobre los ataques, se comporta de manera similar en ambos casos.

No obstante, lo más importante de la figura 6.6 es que permite mostrar, por un lado, la capacidad de la integración para mejorar los resultados de cada uno de los métodos de correlación involucrados y, por otro lado, la habilidad de la integración ponderada para conseguir mayor rendimiento que la integración sin ponderar.

Así, mientras las curvas de detección de los tres métodos de correlación que componen la integración se mueven mayoritariamente entre el 50% y 85%, las capacidades de la función de integración están muy cercanas al 100% en todo momento. Estos resultados indican que el proceso de integración complementa adecuadamente las técnicas basadas en información previa con las que no requieren ningún conocimiento del dominio, principalmente cuando existen ataques nuevos o desconocidos, hecho que ocurre generalmente en la realidad donde aparecen nuevos ataques continuamente.

Observando la figura se puede comprobar la habilidad de la integración ponderada para obtener ventaja del conocimiento de la calidad de los métodos de correlación con respecto a la integración simple que no tiene en cuenta ninguna información de rendimiento de las técnicas que la componen. Si la integración simple obtiene resultados justo por encima del 90%, la versión ponderada es capaz de conseguir un rendimiento cercano al 100% a lo largo de la curva de detección.

Después de analizar el rendimiento de los distintos algoritmos implementados en dos escenarios distintos: escenario DARPA sólo con ataques conocidos y el mismo pero con inclusión de ataques nuevos, es conveniente considerar las capacidades medias de cada uno de los enfoques. Según esto, la figura 6.7 muestra los resultados medios obtenidos por cada uno de los métodos.

En términos medios se observa que PreCons obtiene los resultados más pobres, mientras que el método probabilístico se comporta algo mejor. Según esto, se puede comprobar que la integración de estos dos métodos mejora sensiblemente el

rendimiento de cada uno de ellos por separado. AlertSTAT presenta un comportamiento medio superior al 80% en todos los casos, resultado muy aceptable a la par que predecible. Por último, la integración de los tres métodos de correlación anteriores permite obtener los mejores resultados, siendo mejores en la versión ponderada.

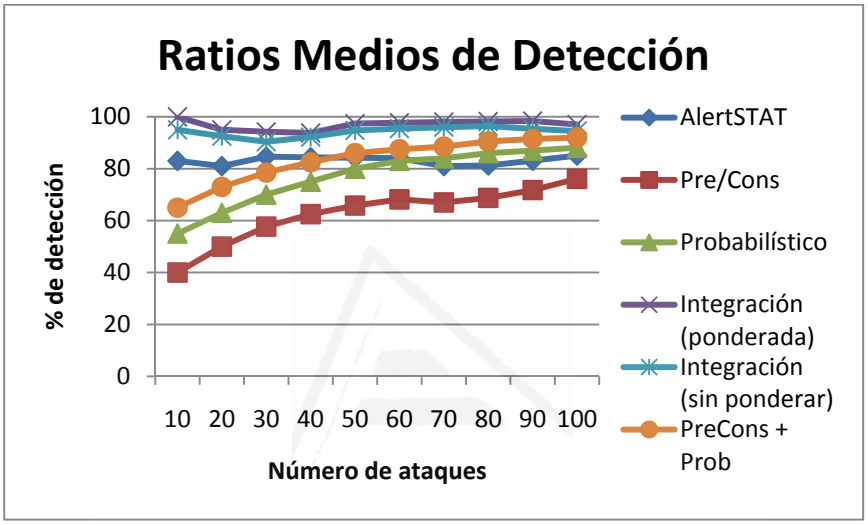


Figura 6.7. Resultados medios de detección.

A modo de resumen, se emplea un método de integración general que ha permitido integrar con éxito tanto dos como tres métodos de correlación y podría llevar a cabo el proceso de la misma manera para un mayor número de técnicas. Tanto en el caso de dos como el de tres, la integración ha obtenido mejores capacidades de detección que cualquiera de los métodos que la componen. Finalmente, la integración ponderada ha conseguido mejorar el rendimiento de su versión sin ponderar, cuando ésta última ya obtenía unos resultados más que aceptables.

Cuando se analizan los resultados de los sistemas de detección, la literatura existente examina principalmente dos características: la capacidad de detección del sistema y su habilidad para no cometer errores. Dado que se han examinado ya los aspectos de detección, se considera a continuación la

vertiente de los errores. La figura 6.8 muestra los ratios de falsos positivos obtenidos por los distintos métodos considerados.

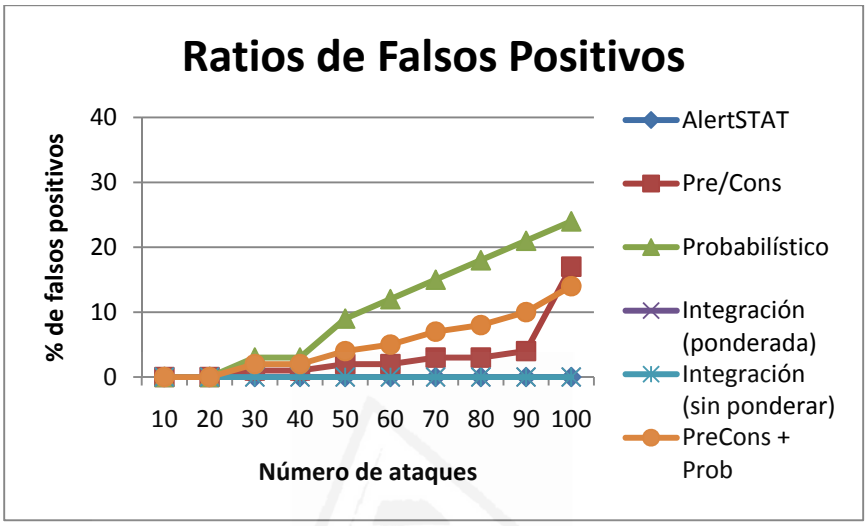


Figura 6.8. Ratios de falsos positivos.

Si es importante mostrar grandes capacidades de detección, también lo es poseer la habilidad de no cometer errores. En este sentido, el ratio de falsos positivos será tanto mejor cuanto menor sea su valor.

El aspecto más destacable de la figura es comprobar que tanto AlertSTAT como las dos variantes de la integración no cometen ningún error, mostrando en todo momento un ratio de falsos positivos del 0%. Esto es debido a que se ha realizado una especificación de grano fino del escenario en AlertSTAT, lo que minimiza o impide que se cometan errores pero imposibilita la detección de cualquier mínima variación del escenario. Además, el proceso de integración tiende a reducir las falsas alarmas emitidas por los métodos de correlación de la misma manera que éstos reducen los falsos positivos emitidos por los sensores (Qin y Lee, 2003).

Cabe mencionar que en este caso el método de prerrequisitos y consecuencias obtiene mejores resultados que la técnica

probabilística. En este sentido, PreCons presenta en general una curva de falsos positivos por debajo del 5%, mientras el método probabilístico crece de manera lineal hasta alcanzar valores por encima del 20% para 100 ataques. La integración de estos dos métodos muestra una curva ligeramente lineal pero con mucha menor pendiente que el método probabilísticos, consiguiendo reducir parcialmente los errores.

Se ha observado que el método probabilístico obtiene mejores ratios de detección que la técnica de prerrequisitos y consecuencias. En cambio, ésta última consigue un ratio de falsos positivos muy inferior a la primera. Por lo tanto, ¿qué método se comporta mejor de manera general?. Para responder a esta cuestión se suelen utilizar las curvas ROC, que relacionan las dos magnitudes y se considera que una curva, y por lo tanto el método al que representa, es mejor la que está siempre por encima de la otra. La figura 6.9 muestra las curvas ROC de todos los métodos empleados en las comparativas.

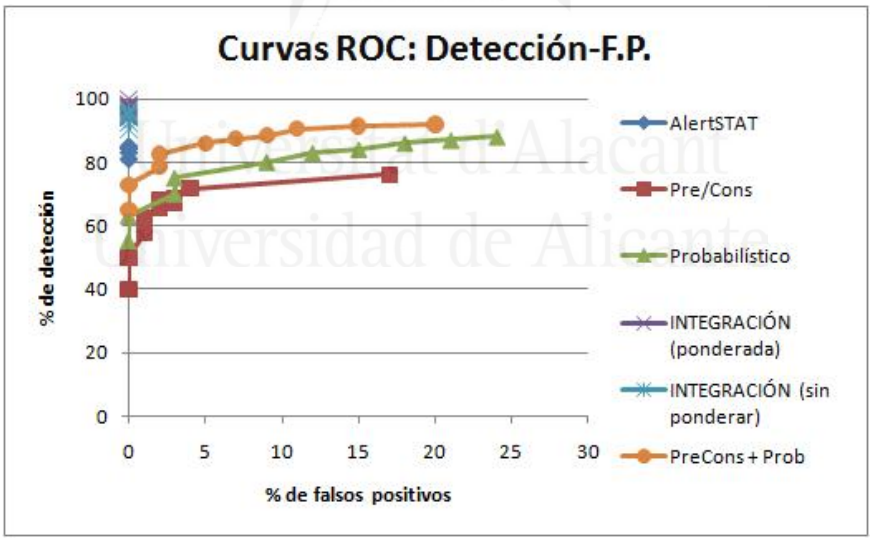


Figura 6.9. Curvas ROC de los métodos comparados.

Lo primero que se aprecia es que la integración de múltiples métodos, tres en este caso, obtiene los mejores resultados. De

hecho, sus curvas asociadas están en la parte superior izquierda de la gráfica por encima de cualquier otra curva. Se observa que el grupo de puntos de la integración ponderada está más arriba que su versión sin ponderar, indicando la habilidad de la ponderación para beneficiarse del conocimiento de la calidad de los métodos que integra.

La curva ROC de AlertSTAT también está claramente por encima de los métodos PreCons y probabilístico, así como de la integración de los dos últimos. Por lo tanto, se puede considerar que el método de especificación de escenarios obtiene, en general, mejores rendimientos que las otras dos técnicas. Aunque, es necesario recordar que se requiere mucho tiempo para codificar los escenarios en los que basa su funcionamiento y, principalmente, que no es capaz de detectar ataques desconocidos.

Finalmente, se puede observar que la curva ROC asociada al método probabilístico está en general por encima de su homónima de prerrequisitos y consecuencias, por lo que se puede decir que presenta mejor comportamiento o rendimiento. Así mismo, la curva que representa la integración de los dos métodos anteriores está siempre por encima de las asociadas a éstos, indicando la obtención de mejores resultados que sus componentes por separado.

Pese a que las curvas ROC permiten mostrar en general una comparativa para decidir la bondad de distintos métodos, en ocasiones es difícil determinar si uno se comporta mejor que otro, en especial cuando las curvas se cruzan. En nuestro caso, el gráfico discrimina perfectamente la mayoría de métodos excepto en el caso de las dos versiones de integración donde la diferencia, aun existiendo, no es tan contundente.

Por la razón anterior es conveniente emplear una medida objetiva que posibilite la comparación directa entre los métodos en función del valor de dicha medida. En este sentido, la métrica C_{ID} (Gu *et al.*, 2008) utilizada para evaluar el rendimiento de los métodos de correlación y ponderar la integración es idónea ya

que devuelve un valor real a partir de los ratios de detección y falsos positivos. La tabla 6.2 muestra los valores de C_{ID} obtenidos por todos los métodos de la comparativa ordenados de mayor a menor.

Tabla 6.2. Medidas de calidad de los métodos comparados.

Método	Valor C_{ID}
Integración ponderada	0.8787
Integración sin ponderar	0.8158
AlertSTAT	0.6311
PreCons + EMERALD	0.4636
EMERALD	0.2818
Prerrequisitos y consecuencias	0.2100

Llegados a este punto y según la tabla anterior, queda claro que los enfoques con integración mejoran los resultados obtenidos por cada uno de los métodos de correlación que componen dicha integración. Así, PreCons + EMERALD mejora el rendimiento conseguido separadamente por prerrequisitos y consecuencias o EMERALD. De la misma manera, los dos métodos que realizan integración múltiple alcanzan resultados muy superiores a cualquiera de sus componentes por separado.

Finalmente, la diferencia de rendimiento entre las dos versiones de integración múltiple, ponderada y sin ponderar, que no quedaba lo suficientemente clara en el gráfico de curvas ROC, aparece aquí perfectamente contrastada. En este sentido, el valor 0.8787 de la métrica C_{ID} del algoritmo de integración con ponderación es un 8% mejor que la versión sin ponderación, lo que indica que la habilidad para emplear el conocimiento sobre la

eficiencia de los métodos de correlación subyacentes produce los mejores resultados.

Conclusiones

Los experimentos realizados y los resultados obtenidos avalan la validez del modelo general de detección propuesto en la presente tesis, demostrando la eficiencia del enfoque de integración de múltiples métodos de correlación, la necesidad de que la integración sea general en lugar de ad hoc y la conveniencia de que se tenga en cuenta en el método de integración la calidad de las técnicas de correlación involucradas.

Los principales resultados obtenidos de la experimentación se pueden resumir como:

- Se ha generalizado la función de integración de manera que el mismo método de integración ha permitido integrar con éxito tanto dos como tres métodos de correlación y podría llevar a cabo el proceso de la misma manera para cualquier número de técnicas.
- El enfoque de integración de múltiples métodos de correlación mejora considerablemente el rendimiento obtenido con respecto a los resultados alcanzados de manera aislada por cada uno de dichos métodos.
- La realización del método de integración de manera ponderada, en función de las capacidades de los métodos de correlación subyacentes, permite obtener mejores resultados que el mismo método sin ponderación.

Capítulo 7

Conclusiones

En la presente tesis se ha llevado a cabo una investigación detallada sobre los problemas de seguridad de las redes de computadores, dentro del ámbito de los Sistemas de Detección de Intrusos Distribuidos o DIDS y centrada, fundamentalmente, en el campo de la correlación e integración de las alertas generadas por este tipo de sistemas.

El principal resultado de este trabajo ha sido la creación de un modelo general de detección de intrusos distribuido que permite y automatiza la integración de múltiples métodos de correlación de alertas. Con el objetivo de facilitar el análisis y la especificación de dicho modelo se ha creado un marco formal que responde a la metodología científica y constituye el contexto en el que se ha definido formalmente el modelo.

Se ha propuesto un método de integración que, basándose en técnicas de aprendizaje de máquina y empleando una métrica de evaluación de IDS objetiva, permite la integración de múltiples métodos de correlación y mejora el rendimiento que cada uno de estos métodos alcanza por separado, aprovechando la información sobre las capacidades de las técnicas de correlación que forman parte de dicha integración.

Así mismo, para llevar a cabo el modelo de la propuesta mediante tecnologías actuales, se ha diseñado una arquitectura completamente distribuida del sistema con criterios de escalabilidad, flexibilidad y adaptabilidad que ha permitido hacer viable el modelo en entornos reales de producción. Tomando como punto de partida la arquitectura propuesta, se ha diseñado un escenario de pruebas sobre el que se ha realizado un conjunto de experimentos que demuestran la validez de la propuesta y por ende de las hipótesis de partida.

Los siguientes apartados abordan con mayor nivel de detalle los elementos anteriores destacando las principales aportaciones de la investigación y las líneas de trabajo futuro.

Aportaciones

En este trabajo se ha profundizado en la investigación de los Sistemas de Detección de Intrusos Distribuidos, concretamente en los aspectos de correlación e integración de la enorme cantidad de alertas que generan este tipo de sistemas. Este estudio ha generado una serie de aportaciones relevantes que se resumen a continuación:

- Un Modelo general para la Detección de Intrusos Distribuido que proporciona una solución de carácter genérico a la integración de múltiples métodos de correlación, formulado rigurosamente, a partir del sistema acción-reacción, para evitar ambigüedades y asegurar la coherencia durante toda su definición.
- Método de Integración de técnicas de correlación, basado en una métrica de calidad y la red neuronal GNG, que permite que el Sistema de Detección de Intrusos pueda aprovechar el conocimiento sobre el rendimiento de cualquier función de correlación que se defina en el sistema sin importar a priori ni el mecanismo de correlación, ni el número de funciones de correlación definidas en el sistema.

- Desarrollo de una arquitectura completamente distribuida que propone una infraestructura tecnológica sobre la cual desplegar el modelo distribuido. Esta arquitectura se basa en un núcleo híbrido entre arquitecturas de N-niveles y arquitecturas de componentes software distribuidos bajo el paradigma del patrón arquitectónico orientado a servicios (SOA), que aporta las condiciones para lograr la escalabilidad y adaptabilidad deseables en este tipo de sistemas.
- Mejora de los rendimientos obtenidos por los actuales Sistemas de Detección de Intrusos Distribuidos gracias a la posibilidad de integrar sistemáticamente múltiples métodos de correlación de alertas y de tener en cuenta la aportación de cada uno de ellos.
- Modificación del algoritmo de la red neuronal GNG para incorporar la capacidad de establecer distintos grados de aprendizaje en función del tipo de patrón de entrada.

Publicaciones Relacionadas

- F.J. Mora; F. Maciá; J.M. García y H. Ramos. "Intrusion Detection System Based on Growing Grid Neural Network". *IEEE Mediterranean Electrotechnical Conference (MELECON 2006)*. 2006.
- V. Gilart; F. Maciá; F.J. Mora y J.V. Berná. "Normalization of industrial machinery with embedded devices and SOA". *IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2006)*. 2006.
- V. Gilart; F. Maciá; D. Marcos y F.J. Mora. "Industrial Machines as a Service: Modelling Industrial Machinery Processes". *IEEE Conference on Industrial Informatics*. 2007.
- I. Lorenzo; F. Maciá; F.J. Mora; R. Lau; J.A. Gil y D. Marcos. "Intrusion Detection Method Using Neural Networks Based on the Reduction of Characteristics". *Bio-Inspired Systems: Computational and Ambient Intelligence (IWANN'09)*, LNCS

5517, Ed. Springer Berlin/ Heidelberg, ISBN: 978-3-642-02477-1. 2009.

  - Indicios de calidad: clase B por el Conference Ranking - Computing Research and Education (CORE)

- F. Maciá; F.J. Mora; D. Marcos; J.A. Gil, H. Ramos y I. Lorenzo. "Network Intrusion Detection System Embedded on a Smart Sensor", *IEEE Transactions on Industrial Electronics*, 08-TIE-0885. (estado actual: 2ª revisión, cambios menores).

  - Indicios de calidad: índice de impacto JCR 2008: 5,468. Posición 1/52 dentro de la categoría "Automation & Control Systems"

Problemas Abiertos

- Optimizar la etapa de integración del modelo evaluando nuevas versiones de redes neuronales auto-organizativas que abren nuevas vías de mejora.
- Analizar diferentes métricas de evaluación de Sistemas de Detección de Intrusos con el objetivo de dotar al modelo de la capacidad para adaptarse dinámicamente a distintos requerimientos de rendimiento.
- Realizar comparativas del abanico de métodos de correlación existente para determinar los mejores rendimientos de cada tipo de métodos.

Líneas Futuras de Investigación

Como en todos los campos de la ciencia existen problemas o enfoques abiertos cuyo estudio es necesario continuar. En este sentido, las principales líneas de continuación tras esta investigación son las siguientes:

- Dotar al modelo de la capacidad de ser proactivo, con el objetivo de que sea capaz de anticiparse a la evolución de los

escenarios y permita detectarlos en sus primeras fases con una elevada probabilidad.

- Incorporar conocimiento semántico al modelo para posibilitar una colaboración de más alto nivel con otros Sistemas de Detección de Intrusos Distribuidos.

Referencias Bibliográficas

(Al-mamory y Zhang, 2007)
Al-mamory, S.O. y Zhang, H.L.: Scenario Discovery Using Abstracted Correlation Graph. *International Conference on Computational Intelligence and Security*. pp. 702–706. 2007.

(Anderson, 1980)
Anderson, J.P.: Computer Security Threat Monitoring and Surveillance. *Technical Report of Fort Washington*. 1980.

(Arora *et al.*, 2004)
Arora, A., Dutta, P., Bapat, S., Kulathumani, V., Zhang, H., Naik, V., Mittal, V., Cao, H., Demirbas, M., Gouda, M., Choi, Y., Herman, T., Kulkarni, S., Arumugam, U., Nesterenko, M., Vora, A. y Miyashita, M.: A Line in the Sand: a Wireless Sensor Network for Target Detection, Classification and Tracking. *Computer Networks*. vol 46. pp. 605–634. 2004.

(Ashfaq *et al.*, 2008)
Ashfaq, A.B., Robert, M.J., Mumtaz, A., Ali, M.Q. y Khayam, S.A.: A Comparative Evaluation of Anomaly Detectors under Portscan Attacks. *Recent Advances in Intrusion Detection*. LNCS 5230. pp. 351–371. 2008.

(Axelsson, 1999)
Axelsson, S.: The Base-rate Fallacy and its Implications for the Difficulty of Intrusion Detection. *ACM Conference on Computer and Communications Security*. pp. 1–7. 1999.

(Azimi-Sadjadi *et al.*, 2007)
Azimi-Sadjadi, B., Kiayias, A., Mercado, A. y Yener, B.: Robust key generation from signal envelopes in wireless networks. *ACM Conference on Computer and Communications Security*. pp. 401–410. 2007.

(Balzarotti *et al.*, 2007)
Balzarotti, D., Cova, M., Felmetsger, V. y Vigna, G.: Multi-Module Vulnerability Analysis of Web-Based Applications. *ACM Conference on Computer and Communications Security*. pp. 25–35. 2007.

(Berquia y Nacsimento, 2004)
Berquia, A. y Nacsimento, G.: A Distributed Approach for Intrusion Detection Systems. *International Conference on Information and Communiction Technologies*. pp. 493–494. 2004.

(Bertino y Martino, 2006)
Bertino, E. y Martino, E.: Security in SOA and Web Services. *IEEE International Conference on Web Services*. pp 41–41. 2006.

(Booch y Rumbaugh, 2001)
Booch, G. y Rumbaugh, J.: *El Lenguaje Unificado de Modelado*. Adisson-Wesley. 2001.

(Brandes *et al.*, 2007)
Brandes, U., Gaertler, M. y Wagner, D.: Engineering graph clustering: models and experimental evaluation. *ACM Journal of Experimental Algorithmics*. Vol. 12, no. 1. pp 1–26. 2007.

(Brauckhoff *et al.*, 2006)
Brauckhoff, D., Tellenbach, B., Wagner, A., May, M. y Lakhina, A.: Impact of packet sampling on anomaly detection metrics. *ACM Conference on Internet Measurement*. pp. 159–164. 2006.

(Braun *et al.*, 2008)
Braun, L., Dressler, F., Holz, T., Kirda, E., Kohlrausch, J., Kruegel, C., Limmer, T., Rieck, K. y Sterbenz, J.P.G.: Requirements for Network Monitoring from an IDS Perspective. *Workshop Network Attack Detection and Defense*. pp. 1–4. 2008.

(Brumley *et al.*, 2006)
Brumley, D., Newsome, J., Song, D.X., Wang, H. y Jha, S.: Towards Automatic Generation of Vulnerability-Based Signatures. *IEEE Symposium on Security and Privacy*. pp. 2–16. 2006.

(Carzaniga *et al.*, 2007)
Carzaniga, A., Picco, G.P. y Vigna, G.: Is Code Still Moving Arround? Looking Back at a Decade of Code Mobility. *International Conference on Software Engineering*. pp. 9–20. 2007.

(Chen *et al.*, 2005)
Chen, W.H., Hsu, S.H. y Shen, H.P.: Application of SVM and ANN for Intrusion Detection. *Computers and Operations Research*. Vol. 32. pp. 2617–2634. 2005.

(Chen y Lukkien, 2007)
Chen, S. y Lukkien, J.: A Service-Oriented Virtual Community Overlay Network for Secure External Service Orchestration. *ACM/IFIP/USENIX International Middleware Conference*. pp. 13–18. 2007.

(Cheng *et al.*, 2007)
Cheng, Y.C., Chen, C.H., Chiang, C.C., Wang, J.W. y Laih, C.S.: Generating Attack Scenarios with Causal Relationship. *IEEE International Conference on Granular Computing*. pp. 368–373. 2007.

(Cheung *et al.*, 2003)
Cheung, S., Lindqvist, U. y Fong, M.W.: Modeling Multistep Cyber Attacks for Scenario Recognition. *DARPA Information Survivability Conference and Exposition*. pp. 284–292. 2003.

(Chu *et al.*, 2005)
Chu, Y., Li, J. y Yang, Y.: The Architecture of the Large Scale Distributed Intrusion Detection System. *International Parallel and Distributed Computing, Applications and Technologies*. pp. 130–133. 2005.

(Clare y Cohen, 2001)
Clare, A. P. y Cohen, D. R.: A Comparison of Unsupervised Neural Networks and K-means Clustering in the Analysis of Multi-Element Stream Sediment Data. *Geochemictry: Exploration, Environment, Analysis*. Vol. 1, no. 2, pp. 119–134. 2001.

(Colajanni *et al.*, 2007)
Colajanni, M., Gozzi, D. y Marchetti, M.: Enhancing Interoperability and Stateful Analysis of Cooperative Network Intrusion Detection Systems. *ACM/IEEE Symposium on Architecture for Networking and Communications Systems*. pp. 165–174. 2007.

(Cretu *et al.*, 2008)
Cretu, G.F., Stavrou, A., Locasto, M.E., Stolfo, S.J. y Keromytis, A.D.: Casting out Demons: Sanitizing Training Data for Anomaly Sensors. *IEEE Symposium on Security and Privacy*. pp. 81–95. 2008.

(CSI, 2008)
CSI.: *The 13th Annual Computer Crime and Security Survey*. Computer Security Institute. 2008.

(Cui *et al.*, 2007)
Cui, W., Peinado, M., Wang, H.J. y Locasto, M.E.: ShieldGen: Automatic Data Patch Generation for Unknown Vulnerabilities with Informed Probing. *IEEE Symposium on Security and Privacy*. pp. 252–266. 2007.

(Cuppens, 2001)
Cuppens, F.: Managing Alerts in a Multi-Intrusion Detection. *17th Environment. Annual Computer Security Applications Conference*. pp. 22–31, 2001.

(Cuppens y Miege, 2002)
Cuppens, F. y Miege, A.: Alert Correlation in a Cooperative Intrusion Detection Framework. *IEEE Symposium on Security and Privacy*. pp. 202–215. 2002.

(Cuppens y Ortalo, 2000)
Cuppens, F. y Ortalo, R.: LAMBDA: A Language to Model a Database for Detection of Attacks. *Recent Advances in Intrusion Detection. LNCS 1907*. pp. 197–216. 2000.

(Dain y Cunningham, 2001)
Dain, O. y Cunningham, R.: Fusing a Heterogeneous Alert Stream into Scenarios. *ACM Workshop on Data Mining for Security Applications*. pp. 1–13. 2001.

(Debar *et al.*, 2007)
Debar, H., Thomas, Y., Cuppens, F. y Cuppens-Boulahia, N.: Enabling Automated Threat Response Through the Use of a Dynamic Security Policy. *Journal in Computer Virology*. Vol. 3, no. 3. pp. 251–266. 2007.

(Debar *et al.*, 2007b)
Debar, H., Curry, D. y Feinstein, B.: *The Intrusion Detection Message Exchange Format (IDMEF)*. RFC 4765. IETF Trust. 2007.

(Debar y Wespi, 2001)
Debar, H. y Wespi, A.: Aggregation and Correlation of Intrusion Detection Alerts. *Recent Advances in Intrusion Detection*. LNCS 2212. pp. 85–103. 2001.

(Dempster, 1967)
Dempster, A.: Upper and Lower Probabilities Induced by Multivalued Mapping. *Annals of Mathematical Statistics*. Vol. 38, no. 2. pp. 325–339. 1967.

(Denning, 1986)
Denning, D.E.: An Intrusion Detection Model. *IEEE Symposium on Security and Privacy*. pp. 118–133. 1986.

(Douglas, 2003)
Douglas, B.K.: *Web Services and Service-Oriented Architectures: The Savvy Manager's Guide*. Morgan Kaufmann Publishers. 2003.

(Eckmann *et al.*, 2002)
Eckmann, S.T., Vigna, G. y Kemmerer, R.A.: STATL: An Attack Language for State-Based Intrusion Detection. *Journal of Computer Security*. Vol. 10, no. 1. pp. 71–104. 2002.

(Ernst & Young, 2006)
Ernst & Young.: *Achieving Success in a Globalized World – 2006 Global Information Security Survey*. Ernst & Young. 2006.

(Flake *et al.*, 2003)
Flake, G.W., Tarjan, R.E. y Tsioutsiouliklis, K.: Graph Clustering and Minimum Cut Trees. *Internet Mathematics*. Vol. 1, no. 4. pp. 1–17. 2003.

(Flórez, 2001)
Flórez Revuelta, F.: *Modelo de Representación y Procesamiento de Movimiento para Diseño de Arquitecturas en Tiempo Real Especializadas*. Tesis doctoral. Universidad de Alicante. 2001.

(Foggia *et al.*, 2007)
Foggia, P., Percannella, G., Sansone, C. y Vento, M.: A Graph Based Clustering Method and Its Applications. International *Symposium on Advances in Brain, Vision and Artificial Intelligence*. pp. 277–287. 2007.

(Fowler, 2002)
Fowler, M.: *Patterns of Enterprise Application Architecture*. Adisson-Wesley Professional. 2002.

(Frincke *et al.*, 2007)
Frincke, D.A., Wespi, A. y Zamboni, D.: From Intrusion Detection to Self-Protection. *Computer Networks*. Vol. 51, no. 5. pp. 1233–1238. 2007.

(Fritzke, 1993)
Fritzke, B.: Growing Cells Structures – A Self-Organizing Network for Unsupervised and Supervised Learning. *Technical Report of International Computer Science Institute, Berkeley*. TR-93-026. 1993.

(Fritzke, 1995)
Fritzke, B.: A growing neural gas network learns topologies. *Advances in Neural Information Processing Systems*. Vol. 7. MIT Press. 1995.

(Fundetec, 2007)
Fundetec.: *Primer Informe Europeo de Seguridad Informática en las Pymes*. Fundetec. 2007.

(Gaertler, 2002)
Gaertler, M.: *Clustering with spectral methods*. Master's thesis. Universitat Konstanz. 2002.

(Gaffney y Ulvila, 2001)
Gaffney, J. E. y Ulvila, J. W.: Evaluation of Intrusion Detectors: A Decision Theory Approach. *IEEE Symposium on Security and Privacy*. pp. 50–61. 2001.

(Gamma *et al.*, 1995)
Gamma, E., Helm, R., Johnson, R. y Vlissides, J. M.: *Design Patterns: Elements of Reusable Object-Oriented Software*. Adisson-Wesley Professional. 1995.

(Ganame *et al.*, 2008)
Ganame, A.K., Bourgeois, J., Bidou, R. y Spies, F.: A Global Security Architecture for Intrusion Detection on Computer Networks. *Computers and Security*. Vol. 27, no. 1. pp. 30–47. 2008.

(García *et al.*., 2007)
García Rodriguez, J., Flórez Revuelta, F. y García Chamizo, J.M.: Image Compression Using Growing Neural Gas. *IJCNN International Joint Conference on Neural Networks*. pp. 366-370. 2007.

(Ghosh *et al.*, 2000)
Ghosh, A.K., Michael, C.C. y Schatz, M.: A Real-Time Intrusion Detection System Based on Learning Program Behavior. *Recent Advances in Intrusion Detection*. LNCS 1907. pp. 93–109. 2000.

(Granger, 1969)
Granger, C.W.J.: Investigating Causal Relations by Econometric Methods and Cross-Spectral Methods. *Econometrica*. Vol. 34. pp. 424–428. 1969.

(Grossberg, 1980)
Grossberg, S.: How does the brain build a cognitive code?. *Psychological Review*. Vol. 87, pp. 1–51. 1980.

(Gu *et al.*, 2006)
Gu, G., Fogla, P., Dagon, D., Lee, W. y Skoric, B.: Measuring Intrusion Detection Capability: An Information_Theoretic Approack. *ACM Symposium on Information, Computer and Communications Security*. pp. 90–101. 2006.

(Gu *et al.*, 2008)
Gu, G., Cárdenas, A.A. y Lee, W.: Principled Reasoning and Practical Applications of Alert Fusion in Intrusion Detection Systems. *ACM Symposium on Information, Computer and Communications Security*. pp. 136–147. 2008.

(Haibin y Jian, 2007)
Haibin, M. y Jian, G.: Intrusion Alert Correlation Based on D-S Evidence Theory. *International Conference on Communications and Networking in China*. pp. 377–381. 2007.

(Han *et al.*, 2004)
Han, S.J., Kim, K.J. y Cho, S.B.: Evolutionary Learning Program's Behavior in Neural Networks for Anomaly Detection. *International Conference on Neural Information Processing*. LNCS 3316. pp. 236–241. 2004.

(Han y Cho, 2006)
Han, S. y Cho, S.: Evolutionary Neural Networks for Anomaly Detection Based on the Behavior of a Program. *IEEE Transcations on Systems, Man, and Cibernetics*. Vol. 36, no. 3. pp. 559–570. 2006.

(Hancock y Wintz, 1966)
Hancock, J. y Wintz, P.: *Signal Detection Theory*. McGraw-Hill. 1966.

(Hofmann *et al.*, 2007)
Hofmann, A., Dedinski, I., Sick, B. y de Meer, H.: A Novelty-Driven Approach to Intrusion Alert Correlation Based on Distributed Hash Tables. *IEEE Symposium on Computers and Communictions*. pp. 71–78. 2007.

(Huang *et al.*, 1999)
Huang, M.Y., Jasper, R.J. y Wicks, T.M.: A Large Scale Distributed Intrusion Detection Framework Based on Attack Strategy Analysis. *Computer Networks*. Vol. 31, no. 23. pp. 2465–2475. 1999.

(Huelsenbeck y Rannala, 1997)
Huelsenbeck, J.P. y Rannala, B.: Phylogenetic Methods Come of Age: Testing Hypotheses in an Evolutionary Context. *Science*. Vol. 276. pp. 227–232. 1997.

(Hung y Wermter, 2003)
Hung, C y Wermter, S.: A Dynamic Adaptive Self-Organising Hybrid Model for Text Clustering. *IEEE International Conference on Data Mining*. pp. 75–82. 2003.

(Ingham *et al.*, 2007)
Ingham, K.L., Somayaji, A., Burge, J. y Forrest, S.: Learning DFA Representations of HTTP for Protecting Web Applications. *Computer Networks*. Vol 51, No. 5. pp. 1239–1255. 2007.

(Jakobsson *et al.*, 2008)
Jakobsson, M., Stolterman, E., Wetzel, S. y Yang, L.: Love and Authentication. *Annual Conference on Human Factors in Computing Systems*. pp. 197–200. 2008.

(Josang *et al.*, 2007)
Josang, A., Alfayyadh, B., Grandison, T., AlZomai, M. y McNamara, J.: Security Usability Principles for Vulnerability Analysis and Risk

Assessment. *Annual Computer Security Applications Conference*. pp. 269–278. 2007.

(Julisch, 2003)
Julisch, K.: Clustering Intrusion Detection Alarms to Support Root Cause Analysis. *ACM Transactions on Information and System Security*. Vol. 6, no. 4. pp. 443–471. 2003.

(Julisch y Dacier, 2002)
Julisch, K. y Dacier, M.: Mining Intrusion Detection Alarms for Actionable Knowledge. *International Conference on Knowledge Discovery and Data Mining*. pp. 366–375. 2002.

(Kannadiga y Zulkernine, 2005)
Kannadiga, P. y Zulkernine, M.: DIDMA – A Distributed Intrusion Detection System Using Mobile Agents. *International Conference on Software Engineering, Artificial Intellegence, Networking and Parallel/Distributed Computing*. pp. 238–245. 2005.

(Karagiannis *et al.*, 2007)
Karagiannis, D., Ronaghi, F. y Fill H.G.: Business-Oriented IT Management: Developing e-Business Applications with E-BMPS. *International Conference on Electronic Commerce*. pp. 97–100. 2007.

(Kemmerer, 2005)
Kemmerer, R.A.: Designing and Implementing a Family of Intrusion Detection Systems. *IEEE/ACM International Conference on Automated Software Engineering*. pp. 88–97. 2005.

(Kohonen, 1982)
Kohonen, T.: Self-organized formation of topologically correct feature maps. *Biological Cybernetics*. Vol. 43. pp. 59–69. 1982.

(Kosko, 1986)
Kosko, B.: Fuzzy Cognitive Maps. *International Journal of Man-Machine Studies*. Vol. 24. pp. 65–75. 1986.

(Kruegel *et al.*, 2005)
Kruegel, C., Valeur, F. y Vigna, G.: *Intrusion Detection and Correlation – Challenges and Solutions*. Springer. 2005.

(Kruegel *et al.*, 2005b)
Kruegel, C., Vigna, G. y Robertson W.K.: A Multi-Model Approach to the Detection of Web-based Attacks. *Computer Networks*. Vol. 48. pp. 717–738. 2005.

(Kruegel y Toth, 2002)
Kruegel, C. y Toth, T.: Distributed Pattern Detection for Intrusion Detection. *Network and Distributed System Security Symposium*. pp. 45–63. 2002.

(Kruegel y Vigna, 2003)
Kruegel, C. y Vigna, G.: Anomaly Detection of Web-based Attacks. *ACM Conference on Computer and Communications Security*. pp. 251–261. 2003.

(Lee y Xiang, 2001)
Lee, W. y Xiang, D.: Information-Theoretic Measures for Anomaly Detection. *IEEE Symposium on Security and Privacy*. pp. 130–143. 2001.

(Li *et al.*, 2007)
Li, W., Zhi-tang, L. y Jie, L.: Learning Attack Strategies Through Minig and Correlation of Security Alarms. *IFIP/IEEE International Symposium on Integrated Network Management*. pp. 713–716. 2007.

(Liang y Sekar, 2005)
Liang, Z. y Sekar, R.: Fast and Automated Generation of Attack Signatures: a Basis for Building Self-Protecting Servers. *ACM Conference on Computer and Communications Security*. pp. 213–222. 2005.

(Lichodzijewski *et al.*, 2002)
Lichodzijewski, P. Zincir-Heywood, A. y Heywood, M.: Dynamic Intrusion Detection Using Self-Organizing Maps. *Annual Canadian Information Technology Security Symposium*. 2002.

(Lincoln *et al.*, 2004)
Lincoln, P., Porras, P.A. y Shmatikov, V.: Privacy Preserving Sharing and Correlation of Security Alerts. *USENIX Security Symposium*. pp. 239–254. 2004.

(Lippmann y Cunningham, 2000)
Lippmann, R. y Cunningham, R.: Improving Intrusion Detection Performance Using Keyword Selection and Neural Networks. *Computer Networks*. Vol. 34. pp. 597–603. 2000.

(Lippmann y Fried, 2000)
Lippmann, R. P. y Fried, D. J.: Evaluating Intrusion Detection Systems: The 1998 Darpa Off-line Intrusion Detection

Evaluation. *DARPA Information Survivability Conference and Exposition*. pp. 12–26. 2000.

(Liu y Quiao, 2006)
Liu, Z.Y. y Qiao, H.: Hidden Markov Model Based Intrusion Detection. *Intelligence and Security Informatics*. Springer. 2006.

(Locasto *et al.*, 2005)
Locasto, M.E., Wang, K., Keromytis, A.D. y Stolfo, S.J.: FLIPS: Hybrid Adaptive Intrusion Prevention. *Recent Advances in Intrusion Detection*. LNCS 3858. pp. 82–101. 2005.

(Maggi y Zanero, 2007)
Maggi, F. y Zanero, S.: On the Use of Different Statistical Tests for Alert Correlation. *Recent Advances in Intrusion Detection*. LNCS 4637. pp. 167–177. 2007.

(Magott y Woda, 2008)
Magott, J. y Woda, M.: Evaluation of SOA Security Metrics Using Attack Graphs. *International Conference on Dependability of Computer Systems*. pp. 277–284. 2008.

(Mahoney y Chan, 2003)
Mahoney, M.V. y Chan, P.K.: An Analysis of the 1999 DARPA Evaluation Data for Network Anomaly Detection. *Recent Advances in Intrusion Detection*. LNCS 2820. pp. 220–237. 2003.

(Majorczyk *et al.*, 2008)
Majorczyk, F., Totel, E., Me, L. y Saidane, A.: Anomaly Detection with Diagnosis in Diversified Systems Using Information Flow Graph. *International Information Security Conference*. pp. 301–315. 2008.

(Marks y Bell, 2006)
Marks, E.A. y Bell, M.: *Service-Oriented Architecture. A Plan and Implementation Guide for Business and Technology*. John Wiley & Sons. 2006.

(Marsland *et al.*., 2000)
Marsland, S., Nehmzow, U. y Shapiro, J.: A Real-Time Novelty Detector for A Mobile Robot. *EUREL Advanced Robotics Conference*. pp. 67–75. 2000.

(Martinetz y Schulten, 1991)
Martinetz, T. y Schulten, K.: A Neural Gas Network Learns Topologies. *Artificial Neural Networks*. Vol. 1. pp. 397–402. 1991.

(McAllister *et al.*, 2008)
McAllister, S., Kirda, E. y Kruegel, C.: Leveraging User Interactions for In-Depth Testing of Web Applications. *Recent Advances in Intrusion Detection*. LNCS 5230. pp. 191–210. 2008.

(Mora *et al.*, 2006)
Mora, F.J., Maciá, F., García, J.M. y Ramos, H.: Intrusion Detection System Based on Growing Grid Neural Network. *IEEE Mediterranean Electrotechnical Conference*. pp. 839–842. 2006.

(Morin y Debar, 2003)
Morin, B y Debar, H.: Correlation of Intrusion Symptoms. *Recent Advances in Intrusion Detection*. LNCS 2820. pp. 94–112. 2003.

(Neumann y Porras, 1999)
Newmann, P.G. y Porras, P.A.: Experience with EMERALD to Date. *Workshop on Intrusion Detection and Network Monitoring*. pp. 73–80. 1999.

(Ning *et al.*, 2002)
Ning, P., Cui, Y. y Reeves, D.S.: Constructing Attacks Scenarios Through Correlation of Intrusion Alerts. *9th ACM Conference on Computer and Communications Security*. pp. 245–254. 2002.

(Ning *et al.*, 2004)
Ning, P., Xu, D., Healey, C.G. y Amant, R.: Building Attacks Scenarios Through Integration of Complementary Alert Correlation Method. *Network and Distributed System Security Symposium*. pp. 69–84. 2004.

(Ning y Xu, 2003)
Ning, P. y Xu, D.: Learning Attack Strategies from Intrusion Alerts. *ACM Conference on Computer and Communications Security*. pp. 200–209. 2003.

(Nist, 2007)
Nist.: *National Vulnerability Database – Automating Vulnerability Management, Security Measurement and Compliance Checking*. National Institute of Standards and Technology. 2007.

(Northcutt *et al.*, 2001)
Northcutt, S., Cooper, M., Fearnow, M. y Frederick, K.: *Intrusion Signatures and Analysis*. New Riders. 2001.

(Northcutt *et al.*, 2005)
Northcutt, S., Zeltser, L., Winters, S., Kent, K. y Ritchey R.W.: *Inside Network Perimeter Security*. New Riders Publishing. 2005.

(Northcutt y Novak, 2003)
Northcutt, S. y Novak, J.: *Network Intrusion Detection*. New Riders Publishing. 2003.

(Peng *et al.*, 2007)
Peng, T., Leckie, C. y Ramamohanarao, K.: Information Sharing for Distributed Intrusion Detection Systems. *Journal of Network and Computer Applications*. Vol. 30. pp. 877–899. 2007.

(Perdisci *et al.*, 2006)
Perdisci, R., Dagon, D., Lee, W., Fogla, P. y Sharif, M.I.: Misleading Worm Signature Generators Using Deliberate Noise Injection. *IEEE Symposium on Security and Privacy*. pp. 17–31. 2006.

(Phan, 2007)
Phan, C.: Service Oriented Architecture (SOA) – Security Challenges and Mitigation Strategies. *IEEE Military Communications Conference*. pp. 1–7. 2007.

(Pollach, 2007)
Pollach, I.: What's Wrong with Online Privacy Policies?. *Communications of the ACM*. Vol. 50, no. 9. pp. 103–108. 2007.

(Porras *et al.*, 2002)
Porras, P.A., Fong, M.W. y Valdes, A.: A Mission Impact Based Approach to INFOSEC Alarm Correlation. *Recent Advances in Intrusion Detection*. LNCS 2516. pp. 95–114. 2002.

(Proctor, 2001)
Proctor, P.E.: *The Practical Intrusion Detection Handbook*. Prentice Hall PTR. 2001.

(Ptacek *et al.*, 1998)
Ptacek, T.H., Newsham, T.N. y Simpson, H.J.: *Insertion, Evasion, and Denial of Service: Eluding Network Intrusion Detection*. Secure Networks Inc. 1998.

(Qin y Lee, 2003)
Qin, X y Lee, W.: Statistical Causality Analysis of INFOSEC Alert Data. *Recent Advances in Intrusion Detection*. LNCS 2820. pp. 73–93. 2003.

(Qin y Lee, 2004)
Qin, X y Lee, W.: Discovering Novel Attack Strategies from INFOSEC Alerts. *9th European Symposium on Research Computer Security*. LNCS 3193. pp. 439–456. 2004.

(Ramadas *et al.*, 2003)
Ramadas, M., Ostermann, S. y Tjaden, B.C.: Detecting Anomalous Network Traffic with Self-Organizing Maps. *Recent Advances in Intrusion Detection*. LNCS 2820. pp. 36–54. 2003.

(Robertson *et al.*, 2006)
Robertson, W.K., Vigna, G., Kruegel, C. y Kemmerer, R.A.: Using Generalization and Characterization Techniques in the Anomaly-based Detection of Web Attacks. *Network and Distributed System Security Symposium*. pp. 35–49. 2006.

(Roesch, 1999)
Roesch, M.: Snort: Lightweight Intrusion Detection for Networks. *13th Conference on Systems Administration*. pp. 229–238. 1999.

(Ruiz *et al.*, 2005)
Ruiz, L.B., Braga, T.R.M., Sikva, F.A., Assuncao, H.P., Nogueira, J.M.S. y Loureiro, A.A.F.: On the Design of a Self-Managed Wireless Sensor Network. *IEEE Communications Magazine*. Vol. 43, no. 8. pp. 95–102. 2005.

(Satoh *et al.*, 2008)
Satoh, F., Nakamura, Y., Mukhi, N.K., Tatsubori, M. y Ono, K.: Methodology and Tools for End-to-End SOA Security Configurations. *IEEE Congress on Services*. pp. 307–314. 2008.

(Shannon, 1948)
Shannon, C.E.: A Mathematical Theory of Communication. *Bell System Technical Journal*. Vol. 27, pp. 379–423, 1948.

(Sharif *et al.*, 2007)
Sharif, M.I., Singh, K., Giffin, J.T. y Lee, W.: Understanding Precision in Host Based Intrusion Detection. *Recent Advances in Intrusion Detection*. pp. 21–41. 2007.

(Sheyner *et al.*, 2002)
Sheyner, O., Haines, J.W., Jha, S., Lippmann, R. y Wing, J.M.: Automated Generation and Analysis of Attack Graphs. *IEEE Symposium on Security and Privacy*. pp. 273–284. 2002.

(Shyu *et al.*, 2007)
Shyu, M., Quirino, T., Xie, Z., Chen, S. y Chang, L.: Network Intrusion Detection Through Adaptive Sub-Eigenspace Modelling in Multiagent Systems. *ACM Transactions on Autonomous and Adaptive Systems*. Vol. 2, no. 3. pp. 9–45. 2007.

(Siaterlis y Maglaris, 2004)
Siaterlis, C. y Maglaris, B.: Towards Multisensor Data Fusion for DoS Detection. *ACM Symposium on Applied Computing*. pp. 439–446. 2004.

(Siraj *et al.*, 2004)
Siraj, A., Vaughn, R.B. y Bridges, S.M.: Intrusion Sensor Data Fusion in an Intelligent Intrusion Detection System Architecture. *Annual International Conference on System Sciences*. pp. 1–10. 2004.

(Soleimani y Ghorbani, 2008)
Soleimani, M. y Ghorbani, A.: Critical Episode Mining in Intrusion Detection Alerts. *Communication Networks and Services Research Conference*. pp. 157–164. 2008.

(Spafford y Zamboni, 2000)
Spafford, E.H. y Zamboni, D.: Intrusion Detection Using Autonomous Agents. *Computer Networks*. Vol. 34, no. 4. pp. 547–570. 2000.

(Staniford *et al.*, 2002)
Staniford, S., Hoagland, J. y McAlerney, J.: Practical Automated Detection of Stealthy Portscans. *Journal of Computer Security*. Vol. 1. pp. 105–136. 2002.

(Stolfo *et al.*, 2000)
Stolfo, S., Fan, W., Lee, W., Prodromidis, A. y Chan, P.: Cost-based Modeling for Fraud and Intrusion Detection. *DARPA Information Survivability Conference and Exposition*. pp. 130–144. 2000.

(Sung y Mukkamala, 2003)
Sung, A.H. y Mukkamala, S.: Identifiying Important Features for Intrusion Detection Using Support Vector Machines and Neural Networks. *Symposium on Applications and the Internet*. pp. 209–217. 2003.

(Tan y Maxion, 2002)
Tan, K.M.C. y Maxion, R.A.: Why 6? Defining the Operational Limits of Stide, an Anomaly-Based Intrusion Detector. *IEEE Symposium on Security and Privacy*. pp. 188–201. 2002.

(Templeton y Levitt, 2000)
Templeton, S. y Levitt, K.: A Requires/Provides Model for Computer Attacks. *New Security Paradigms Workshop*. pp. 31–38. 2000.

(Toop *et al.*, 2002)
Toop, U., Muller, P., Konnertz, J. y Pick, A.: Web based Service for Embedded Devices. *Workshop on Web, Web-Services, and Database Systems*. LNCS 2593. pp. 141–153. 2002.

(Turner *et al.*, 2008)
Turner, D., Fossi, M., Johnson, E., Mark, T., Blackbird, J., Entwisle, S., Low, M.K., McKinney, D. y Wueest, C.: *Symantec Global Internet Security Threat Report, Trends for July-December 07*. Vol. XIII. Symantec. 2008.

(Ullrich, 2004)
Ullrich, J.: *DSHIELD Intrusion Detection System*. Dshield.org. 2004.

(Valdes y Skinner, 2001)
Valdes, A. y Skinner, K.: Probabilistic Alert Correlation. *Recent Advances in Intrusion Detection*. LNCS 2212. pp. 54–68. 2001.

(Valeur et at, 2004)
Valeur, F., Vigna, G., Kruegel, C. y Kemmerer, R.A.: A Comprehensive Approach to Intrusion Detection Alert Correlation. *IEEE Transactions on Dependable and Secure Computing*. Vol. 1, no. 3. pp. 146–169. 2004.

(van Dongen, 2000)
Van Dongen, S.M.: *Graph clustering by Flow Simulation*. Ph.D. thesis. University of Utrecht. 2000.

(Van Gundy *et al.*, 2007)
Van Gundy, M., Chen, H., Su, Z. y Vigna. G.: Feature Omission Vulnerabilities: Thwarting Signature Generation for Polymorphic Worms. *Annual Computer Security Applications Conference*. pp. 74–85. 2007.

(Vempala, Kannan y Vetta, 2000)
Vempala, S., Kannan, R y Vetta, A.: On clusterings – good, bad and spectral. *41st Annual IEEE Symposium on Fundations of Computer Science*, pp. 367–378. 2000.

(Vlachos *et al.*, 2004)
Vlachos, V., Androutsellis-Theotokis, S. y Spinellis, D.: Security Applications of Peer-to-Peer Networks. *Computer Networks*. Vol. 45, no. 2. pp. 195–205. 2004.

(Wagner, 2004)
Wagner, D.: Cryptanalysis of a Provably Secure CRT-RSA Algorith. *ACM Conference on Computer and Communications Security*. pp. 92–97.2004.

(Waldinger, 1977)
Waldinger, R.: Achieving Several Goals Simultaneously. *Machine Intelligence*. Vol. 8, pp. 94–136. 1977.

(Wang *et al.*, 2006)
Wang, K., Parekh, J.J. y Stolfo, S.J.: Anagram: A Content Anomaly Detector Resistant to Mimicry Attack. *Recent Advances in Intrusion Detection.* LNCS 4219. pp. 226–248. 2006.

(Wang *et al.*, 2007)
Wang, L., Singhal, A. y Jajodia, S.: Toward Measuring Network Security Using Attack Graphs. *ACM Conference on Computer and Communications Security*. pp. 49–54. 2007.

(Wang y Stolfo, 2004)
Wang, K. y Stolfo, S.J.: Anomalous Payload-Based Network Intrusion Detection. *Recent Advances in Intrusion Detection.* pp. 203–222. 2004.

(Xiaoping y Yu, 2004)
Xiaoping, Y. y Yu, D.: An Auto-Configuration Cooperative Distributed Intrusion Detection System. *World Congress on Intelligent Control and Automation.* pp. 4375–4379. 2004.

(Ye *et al.*, 2008)
Ye, D., Bai, Q., Zhang, M. y Ye, Z.: P2P Distributed Intrusion Detections by Using Mobile Agents. *IEEE International Conference on Computer and Information Science*. pp. 259–265. 2008.

(Yegneswaran *et al.*, 2004)
Yegneswaran, V., Barford, P. y Jha, S.: Global Intrusion Detection in the DOMINO Overlay System. *Network and Distributed System Security Symposium*. pp. 57–74. 2004.

(Yin *et al.*, 2004)
Yin, Q., Zhang, R. y Li, X.: An new intrusion detection method based on linear prediction. *International Conference on Information Security*. pp. 160–165. 2004.

(Zanero y Savaresi, 2004)
Zanero, S y Savaresi, S.M.: Unsupervised Learning Techniques for an Intrusion Detection System. *ACM Symposium on Applied Computing*. pp. 412–419. 2004.

(Zhang *et al.*, 2005)
Zhang, Y.F., Xiong, Z.Y. y Wang, X.Q.: Distributed Intrusion Detection Based on Clustering. *International Conference on Machine Learning and Cybernetics*. pp. 2379–2383. 2005.

(Zhao-wen *et al.*, 2007)
Zhao-wen, L., Xing-tian, R. y Yan, M.: Agent-based Distributed Cooperative Intrusion Detection System. *International Conference on Communications and Networking in China*. pp. 17–22. 2007.

(Zhou *et al.*, 2007)
Zhou, J., Heckman, M., Reynolds, B., Carlson, A. y Bishop, M.: Modeling Network Intrusion Detection Alerts for Correlation. *ACM Transactions on Information and System Security*. Vol. 10, no. 1. pp. 77–107. 2007.