Departamento de Teoría de la Señal, Telemática y Comunicaciones

Programa de posgrado en Sistemas Multimedia

UNIVERSIDAD DE GRANADA

Tesis Doctoral

Sistema de detección de intrusos mediante modelado de URI

Doctorando:
Rolando Salazar Hernández

Director:
Jesús E. Díaz Verdejo

Granada, 2015

Editor: Universidad de Granada. Tesis Doctorales
Autor: Rolando Salazar Hernández
ISBN: 978-84-9125-731-8
URI: http://hdl.handle.net/10481/43353

El doctorando D. Rolando Salazar Hernández y el director D. Jesús Esteban Díaz Verdejo, Catedrático de Universidad del Departamento de Teoría de la Señal, Telemática y Comunicaciones de la Universidad de Granada

GARANTIZAMOS AL FIRMAR ESTA TESIS DOCTORAL

que el trabajo ha sido realizado por el doctorando bajo mi dirección y, hasta donde nuestro conocimiento alcanza, en la realización del trabajo se han respetado los derechos de otros autores a ser citados, cuando se han utilizado sus resultados o publicaciones.

Granada, 30 de octubre de 2015

Director de la tesis

Dr. D. Jesús Esteban Díaz Verdejo

Doctorando

D. Rolando Salazar Hernández

Dedicatoria

A Clarisa
A mis hijos Rolando y Rodrigo

Agradecimientos

Mi agradecimiento a todas las personas que han colaborado para hacer posible esta tesis.

Al Dr. Díaz Verdejo por la dirección y apoyo incondicional para la realización de la tesis.

A los profesores del Doctorado de Tecnologías Multimedia de la Universidad de Granada, España por compartir sus conocimientos con un servidor.

A las autoridades de la Universidad Autónoma de Tamaulipas por brindar las facilidades para el desarrollo y conclusión de la tesis.

A mis amigos J.J. Ramos, María Ángeles, Guillermo, Luz, gracias por animarme y alentarme a culminar esta tesis.

A mis compañeros del departamento de Teoría de la Señal, Telemática y Comunicaciones de la Universidad de Granada, España. J.J. Ramos, Paco, Jorge, Erika, Pablo Almeigeiras, Povedano, Santi, Carlos, Rosa, Salcedo, Gabriel, Sonia, Pablo Padilla, Ángel, Isaac.

A mis compañeros de piso, Regibel (tod@s), Cofré y familia, Hugo, Erik, Ruete y familia, Raúl y familia.

A mis compañeros del piso ONU, Manu (Español), Donald (Gabón), Mustafá (Marruecos) y Rolando (México).

A mis compañeros de Café, agradecido estoy por compartir momentos inolvidables.

A Dios por darme la fuerza espiritual.

Glosario

A-HIDS	Anomaly-based Host-based Intrusion Detection System (sistema de detección de intrusos basado en host y en anomalías)
A-IDS	Anomaly-based Intrusion Detection System (sistema de detección de intrusos basado en anomalías)
A-NIDS	Anomaly-based Network-based Intrusion Detection System (sistema de detección de intrusos basado en red y en anomalías)
ANN	Artificial Neural Network (red neuronal artificial)
ASCII	American Standard Code for Information Interchange (código estándar estadounidense para el intercambio de información)
CERT	Computer Emergency Response Team (equipo de respuesta a emergencias informáticas)
CIDF	Common Intrusion Detection Framework (marco común para la detección de intrusiones)
CISL	Common Intrusion Specification Language (lenguaje común de especificación de intrusión)
CNN	Chaotic Neural Network (redes neuronales caóticas)
DARPA	Defense Advanced Research Projects Agency (Agencia de investigación de proyectos avanzados para la defensa)
DDoS	Distributed Denial of Service (denegación de servicio distribuida)
DNI	Documento Nacional de Identidad
DNS	Domain Name System (sistema de nombres de dominio)
DoS	Denial of Service (denegación de servicio)
FIRE	Fuzzy Intrusion Recognition Engine (Motor difuso de reconocimiento de intrusiones)
FN	False Negative (falso negativo)
FP	False Positive (falso positivo)
FTP	File Transfer Protocol (protocolo de transferencia de archivos)
HIDS	Host-based Intrusion Detection System (sistema de detección de intrsusos basado en host)
HMM	Hidden Markov Model (modelo oculto de Markov)
HTTP	Hypertext Transfer Protocol (protocolo de transferencia de hipertexto)
ITU	International Telecommunication Union (Unión Internacional de Telecomunicaciones)
ICMP	Internet Control Message Protocol (protocolo de mensajes de control de Internet)

IDES	Intrusion Detection Expert System (Sistema experto para la detección de intrusiones)
IDEVAL	Intrusion Detection EVALuation (Proyecto para la evaluación de los sistemas de detección de intrusiones)
IDS	Intrusion Detection System (sistema de detección de intrusos)
IDWG	Intrusion Detection Working Group (Grupo de trabajo para la detección de intrusiones)
IETF	Internet Engineering Task Force (Equipo de ingeniería de Internet)
INCIBE	Instituto Nacional de Ciberseguridad
IP	Internet Protocol (protocolo de Internet)
ITSEC	Information Technology Security Evaluation Criteria (criterio de evaluación de la seguridad de las tecnologías de la información).
KDD	Knowledge Data Discovery (proceso de extracción de conocimiento)
LARIAT	Lincoln Adaptable Real-time Assurance Test-bed (Banco de pruebas adaptable y en tiempo real para la garantía de la seguridad Lincoln)
MIB	Management Information Base (base de datos de información de gestión)
MIT	Massachusetts Institute of Technology (Instituto Tecnológico de Massachusetts)
MLP	MultiLayer Perceptron (perceptrón multicapa)
NCSC	National Computer Security Center.(Centro Nacional de Seguridad Computacional)
NIDES	New Intrusion Detection Expert System (Nuevo sistema experto de detección de intrusos)
NIDS	Network-based Intrusion Detection System (sistema de detección de intrusos basado en red)
OOV	Out of Vocabulary (fuera de vocabulario)
OSI	Open Systems Interconnection (modelo de interconexión de sistemas abiertos)
OSVDB	Open Sourced Vulnerabilities Data Base (base de datos de dominio público de vulnerabilidades)
OWASP	Open Web Application Security Project (Proyecto abierto de seguridad en aplicaciones web)
PDU	Protocol Data Unit (unidad de datos de protocolo)
PGP	Pretty Good Privacy (privacidad bastante buena)
PSM	Partition-based Stochastic Modeling (modelo estocástico basado en particionamiento)
RFC	Request for Comments (solicitud de comentarios)
RNA	Red neuronal artificial
ROC	Receiver Operating Characteristic (característica operativa del receptor)
S-HIDS	Signature-based Host-based Intrusion Detection System (sistema de detección de intrusos basado en host y en firmas)
S-IDS	Signature-based Intrusion Detection System (sistema de detección de intrusos basado en firmas)

S-NIDS	Signature-based Network-based Intrusion Detection System (sistema de detección de intrusos basado en firmas y en red)
SMM	Safety System Monitor (monitor de seguridad del sistema)
SNMP	Simple Network Management Protocol. (Protocolo de monitoreo de red simple)
SOG-IS	Seniors Officials Group - Information Systems Security (Grupo de oficiales senior – seguridad de los sistemas de información)
SPADE	Smart Python multi-Agent Development Environment (Entorno de desarrollo multiagente inteligente en Python)
SSL	Secure Sockets Layer (capa de conexión segura)
SSM	Stochastic Structural Model (Modelo estructural estócastico)
TCP	Transmission Control Protocol (protocolo de control de la transmisión)
TIC	Tecnologías de la Información y las Comunicaciones
TN	True Negative (verdadero negativo)
TP	True Positive (verdadero positivo)
UDP	User Datagram Protocol (protocolo de datagrama de usuario)
URI	Uniform Resource Identifier (identificador de recursos uniforme)
URL	Uniform Resource Locator (localizador uniforme de recursos)
VRT	Vulnerability Research Team (Equipo de investigación de vulnerabilidades)

Contenido

Índice de figuras

Índice de tablas

Resumen

Cada vez son más las actividades diarias que dependen del uso de las redes de computadoras, en especial del uso del Internet y sus servicios. El acceso a los datos y a la información cada vez cobra mayor relevancia, desde la lectura de los periódicos, hasta las compras de diversos productos y servicios por la red. Esto ha dado pie a que cada día surjan nuevas amenazas o ataques cibernéticos que pueden alcanzar elevados niveles de peligrosidad y con un potencial alto impacto. Así, acciones como el robo de datos, la suplantación de identidad, la intrusión a equipos de cómputo y redes de computadoras, al igual que otras de muy diversa naturaleza, ponen en riesgo las operaciones diarias de cualquier persona o institución.

En este contexto se desarrollan herramientas informáticas y procedimientos cuya finalidad es mitigar o anular cualquier amenaza que ponga en riesgo las operaciones en la red y/o la seguridad de los sistemas y los usuarios. Entre estas podemos mencionar algunas como los analizadores de vulnerabilidades, los antivirus, los cortafuegos o los sistemas de detección de intrusos, que abordan la seguridad desde diferentes enfoques tanto preventivos, como de detección y respuesta.

Motivados por ayudar a minimizar el riesgo asociado a las intrusiones o ataques a los recursos de una red de computadores, surge el presente trabajo, centrado en los sistemas de detección de intrusiones. Así, este tiene como objetivo principal el desarrollo de mejoras a un sistema de detección de intrusos en red basado en el modelado de los mensajes intercambiados por un protocolo de comunicaciones. Este sistema, denominado SSM (del inglés, *Structural Stochastic Model*), utiliza el modelado de Markov para representar las cargas útiles asociadas a protocolos basados en el paso de mensajes. En particular, sus autores mostraron su operación con éxito para la detección de ataques basados en web, es decir, que utilicen el protocolo HTTP para transportar cargas útiles maliciosas. El sistema resultante es un detector de intrusos basado en anomalías, ya que la detección se realiza a partir del análisis de las desviaciones de dichas cargas útiles respecto del modelo de normalidad establecido.

Los buenos resultados obtenidos por este IDS en los entornos de laboratorio considerados para la experimentación motivan la exploración de potenciales modificaciones orientadas a mejorar sus prestaciones para su operación en escenarios reales e incluso en servicios web en explotación. En este sentido, en el presente trabajo se proponen y evalúan diversas propuestas de diferente calado en el sistema SSM.

Así, inicialmente, y tras analizar las limitaciones operativas de la técnica original, se desarrollan modificaciones cuya finalidad es mejorar el rendimiento y aplicabilidad de la misma, aunque sin afectar a la esencia de dicho sistema de detección de intrusiones. Estas mejoras están relacionadas con la aplicación del sistema en

escenarios con grandes vocabularios, es decir, con una variabilidad significativa en los posibles valores de las cadenas que pueden observarse, lo que está asociado a sitios web complejos y con gran dinamicidad en sus contenidos. Así, a diferencia de como se aplica en el sistema original, se propone un tratamiento diferenciado de los vocabularios en cada uno de los estados del modelo, lo que afecta a la implementación, al suavizado de las probabilidades de observación utilizadas por el sistema y a la/s probabilidad/es de observaciones fuera del vocabulario. En particular, se propone y evalúa la utilización de probabilidades de fuera de vocabulario diferenciadas por estado a fin de tener en cuenta los diferentes tamaños de los vocabularios asociados a cada uno de ellos.

Previamente, en relación al escenario experimental a utilizar, se propone y aplica una metodología para la adquisición de trazas de tráfico, tanto normal como de ataques, que garanticen la obtención de un modelado suficientemente representativo y una evaluación adecuada de las capacidades del detector en escenarios reales. Para ello se establecen varias particiones del conjunto de datos de diversa naturaleza y con diferente finalidad. Adicionalmente, a fin de evitar los problemas de privacidad asociados al análisis de las cargas útiles de un protocolo, se propone una técnica de anonimización, desarrollada el efecto, que preserva la información necesaria para la aplicación del modelado y la evaluación de las secuencias. Utilizando las bases de datos obtenidas, se constata una evidente mejora de los resultados con las modificaciones propuestas, que es especialmente relevante para el caso de grandes vocabularios.

Seguidamente, dado que la técnica de modelado utilizada es independiente de la naturaleza maliciosa o no de las cadenas a modelar, se presenta y evalúa una propuesta basada en la utilización del modelado tanto para representar las cadenas normales como las de ataque. Así, el sistema de detección de intrusiones resultante evolucionaría de ser basado en anomalías, a partir de la detección mediante un umbral de anormalidad, a un sistema híbrido que determina la naturaleza de cada cadena a partir del modelo (normal o ataque) que proporciona mayor probabilidad. El sistema de reconocimiento inicialmente propuesto es modificado para mejorar sus prestaciones mediante la incorporación de nuevo de un detector de umbral. Este detector únicamente se utilizaría en aquellos casos considerados dudosos por el reconocedor, es decir, aquellos casos en los que las probabilidades de generación por el modelo normal y de ataque son similares. Finalmente, y utilizando como medida objetiva a minimizar durante el entrenamiento de los modelos la diferencia de dichas probabilidades, se propone la utilización de una técnica de entrenamiento discriminativa que mejore las prestaciones del sistema resultante.

Las diversas propuestas presentadas son convenientemente evaluadas y los resultados son comparados con los que se obtienen mediante el sistema SSM original, constatándose una mejora en las prestaciones del sistema de detección de intrusiones. Estos resultados son también validados mediante la utilización de particiones de las bases de datos que habían sido establecidas con anterioridad y que no se había utilizado previamente para evaluar el sistema.

Abstract

Nowadays, more and more daily activities depend on the use of computer networks, particularly on the use of the Internet and its services. Access to data and information becomes increasingly relevant, from reading the newspapers, to purchases of various products and services for the network. This has given rise to new threats or cyberattacks that can achieve high levels of risk and high potential impact arising every day. Thus, actions such as data theft, identity theft, computer equipment and computer networks intrusions and other very diverse in nature, are threatening the daily operations of any person or institution.

In this context, tools and procedures are developed which aims to mitigate or nullify any threat that jeopardizes the network operations and / or the security systems and users. Among these we can mention some as vulnerabilities analyzers, antivirus, firewalls or intrusion detection systems (IDSs), addressing security from different approaches as preventive, detection and response.

Thus, the present thesis has as its main objective the development of improvements to network-based intrusion detection system based on the modeling of the messages exchanged by a communication protocol. This so-called SSM (Structural Stochastic Model) uses Markov modeling to represent the payload contents for protocols based in the exchange of messages. In particular, the authors showed their successful operation for detecting web-based attacks, that is, attacks using the HTTP protocol to transport malicious payloads. The resulting system is an anomaly-based intrusion detector, as the detection is performed based on the analysis of the deviations of those payloads from the established model of normality.

The good results obtained by the IDS in laboratory environments motivate exploring potential modifications aimed at improving its performance for operation in real scenarios and even in deployed web services. In this sense, this paper proposes and evaluates various different proposals on the SSM system.

So, initially, and after analyzing the operating limitations of the original technique, the proposed modifications aimed at improving the performance and applicability of the method, but without affecting the essence of the intrusion detection system. These improvements are related to the implementation of the system in scenarios with large vocabularies, i.e., with significant variability in the possible string values that can be observed, which is associated with complex web sites with great dynamism in its contents. So, unlike as applied in the original system, a different treatment of the vocabularies in each state of the model is proposed, which affect the implementation, the smoothing of the observation probabilities used by the system and the probability for observations out of the vocabulary (OOV). In particular, it is proposed and evaluated using OOV probabilities differentiated by state to take into account the different sizes of each associated vocabularies.

Previously, in relation to the experimental framework, it is proposed and applied a methodology for the acquisition of traffic traces, both normal and attacks, which ensure the provision of a sufficiently representative modeling and a proper assessment of the capabilities of the detector in real settings. For this, multiple partitions of the dataset of various kinds and for different purposes are established. Additionally, to avoid the privacy concerns related to analyzing the payloads of a protocol, an anonymization technique is proposed. This technique preserves the information necessary for implementing the modeling and the evaluation of sequences. Using the obtained databases, a clear improvement in the results when using the proposed amendments, which is especially relevant in the case of large vocabularies, is found.

Then, as the modeling technique used is independent on the malicious or normal nature of the instances to model, we present and evaluate a proposal based on the use of the modeling to represent both the normal and attack sequences. Thus, the resulting IDS evolves from being an anomaly-based one, from detection by a threshold of abnormality, to a hybrid system which determines the nature of each payload from the model (normal or attack) that provides the greater probability. The initially proposed recognition system is modified to improve their performance by incorporating a new threshold-based detector. This detector would be used only in those borderline cases considered by the recognizer, that is, those cases where the probability of generation from the normal and attack models are similar. Finally, using the difference of these probabilities as the objective measure to minimize during the training of the models, a discriminative training technique is applied to improve the performance of the resulting system.

The various proposals are properly assessed and the results are compared to those obtained using the original SSM system, confirming an improvement in the performance of the IDS. These results are also validated using the validation partitions in the databases. They had been established previously and were not previously used to evaluate the system.

1 Introducción: sistemas de detección de intrusos

El desarrollo de las TIC (Tecnologías de la Información y las Comunicaciones) ha supuesto toda una revolución social. De la mano de estas tecnologías la penetración de Internet ha alcanzado cuotas inimaginables no hace demasiado tiempo. Servicios como la web, de interfaz atractiva, fácil uso y enormes capacidades, hacen que la adquisición de libros y música, las transferencias bancarias o la reserva de vuelos de avión sean actividades cada vez más habituales en el ámbito electrónico.

Este hecho, sin embargo, tiene sus riesgos. El trasvase y almacenamiento de información, especialmente la relativa a datos sensibles (cuentas y claves bancarias, datos personales, historiales médicos, etc.), deben ser cuidadosamente implementados y vigilados de cara a impedir accesos no autorizados y, en suma, a garantizar la seguridad de la información.

Es en este contexto de la necesidad de proporcionar unos mecanismos de seguridad mínimos que permitan la confianza de los usuarios en los sistemas y servicios involucrados en las TIC, en el que se enmarca la presente tesis doctoral. A lo largo de este capítulo se presenta una visión general de la seguridad en redes y comunicaciones, particularizando la consecución de ésta a través del empleo de los denominados *sistemas de detección de intrusos*, o IDS (del inglés, *Intrusión Detection Systems*). Objeto central en el que se incardina este trabajo, se llevará a cabo una revisión del estado del arte de esta tecnología y se concluirá el capítulo con los objetivos específicos de la tesis y la organización y estructura de esta memoria.

1.1 Seguridad en redes y sistemas

Las comunicaciones constituyen una parte vital de cualquier actividad individual o colectiva en la actualidad. Así, la disposición de una red de computadores para distribuir información permite una toma de decisiones más ágil y efectiva.

Por otra parte, son cada vez más los servicios que se prestan basados en la red, especialmente Internet, por lo que ésta se convierte en un recurso de gran importancia actual para la sociedad en general. En consecuencia, cualquier fallo en la infraestructura de red, software o hardware, puede tener graves repercusiones para los recursos y servicios ofertados, con los consiguientes perjuicios de ello derivados. Estos fallos se

deben en gran medida al alto volumen de software malicioso (*malware*) que se distribuye por la red; en muchas ocasiones, de manera automática sin que el usuario sea consciente de ello. Para mitigar los efectos de estos programas es necesario el despliegue de herramientas que permitan en primera instancia su detección temprana y, a partir de ello, la puesta en marcha de procedimientos orientados a evitar los posibles efectos perniciosos derivados de su ejecución.

En todo este contexto surgen conceptos relativos a la seguridad de las redes de comunicaciones como son *vulnerabilidad* y *ataque*. La diferencia básica entre el uno y el otro es que el primero se refiere a un error en el diseño o la configuración del software o del hardware, lo que posibilita acciones no autorizadas, como que un intruso pueda acceder a un sistema. En cambio, un ataque es un intento de explotar una vulnerabilidad. Por lo tanto, para mejorar la seguridad de un sistema es necesario estar al tanto de las nuevas vulnerabilidades que se publican en los foros dispuestos al efecto, como CERT [CERT, 1988], en los que se informa sobre las vulnerabilidades detectadas y se indican las posibles contramedidas a aplicar [McHugh, 2001].

A medida que ha ido creciendo la dependencia de la sociedad con las tecnologías de la información y las comunicaciones, también lo ha hecho el número de incidentes de seguridad reportados contra este tipo de entornos y sistemas [PWC, 2015] [INCIBE, 2013]. Es por ello que en las últimas décadas ha crecido significativamente el interés general por las tecnologías de la seguridad [Asenova et al., 2015]. En este sentido, ya en 2002 Maiwald realizó un trabajo acerca de cómo ha evolucionado la seguridad en sus diferentes áreas; desde aspectos físicos, p.e. control de accesos o custodia de soportes de información, hasta llegar a la seguridad de la información en el sentido actual, p.e. cifrado de la información o protocolos de comunicación seguros [Maiwald, 2002]. En un trabajo más reciente, Krutz, Conley y Cole realizan una recopilación de los conceptos de seguridad en redes [Cole et al., 2009].

En los años 70 del siglo pasado, Bell y La Padula desarrollaron un modelo para la operación segura en computadores. Éste estaba basado en varios niveles de clasificación de la información, desde “sin clasificar” hasta “alto secreto”, dependiendo de a qué persona en el nivel jerárquico dentro de la organización iba dirigida. Este modelo fue adoptado por los militares estadounidenses para el envío de mensajes dentro de sus jerarquías. Posteriormente, este trabajo fue retomado en el año 1983 y bautizado con el nombre de Libro Naranja [NCSC, 1983], el cual define los requerimientos de funcionalidad de acuerdo a la seguridad. En 1991 nace, así, el criterio de evaluación de la seguridad de las tecnologías de la información (o ITSEC, por sus siglas en inglés) [SOG-IS, 1991], y el criterio Federal en 1992. El Libro Naranja (*Orange Book*) e ITSEC no sólo clasifican la información en función del destino, sino que también establecen criterios y normas que deben cumplir los sistemas, tanto hardware como software, para alcanzar un determinado nivel de seguridad.

Toda esta problemática se ve agravada en el contexto de las redes de computadores y las comunicaciones globales, por lo que se hace necesario el desarrollo de una política de seguridad de redes general para toda la estructura de una entidad.

Para que la prestación de los servicios de red no se vea afectada parcial o totalmente por incidentes de seguridad, existen modelos de referencia para evaluar las políticas de seguridad aplicadas en un entorno de red. Así, por ejemplo, la recomendación X.800 de la I.T.U. (*International Telecommunication Union*) [ITU-T, 1991] establece servicios de seguridad para las capas del modelo de referencia OSI que permitan garantizar la confianza adecuada en los sistemas y/o en las transferencias de datos. Dichos servicios se concretan en cinco principales: *confidencialidad, integridad, autenticación, no repudio y disponibilidad* [ITU-T, 1991] [Huidobro M. & Roldan M., 2005] [Carracedo, 2004] [Stallings, 2003]. El primero se refiere a la inaccesibilidad de la información a terceros. Por su parte, la integridad es la capacidad de evitar que la información pueda ser modificada por personas no autorizadas. A través de la autenticación se persigue garantizar que las partes involucradas en una comunicación son quienes dicen ser. El no repudio evita que un participante en una comunicación pueda negar dicha participación o la autoría de los mensajes intercambiados. Finalmente, y no menos importante, la disponibilidad se refiere a la propiedad de que un sistema esté accesible y utilizable siempre que así se precise por parte de los usuarios.

Diversos son los mecanismos específicos propuestos para proporcionar uno o varios de los distintos servicios de seguridad antes citados. Entre ellos cabe destacar la utilización de esquemas criptográficos, bien sean de clave privada o de clave pública. En todo caso, seguidamente se proporciona un estudio más pormenorizado de los más habituales.

1.2 Mecanismos de seguridad

Mediante una administración adecuada de un sistema y el uso de herramientas de seguridad de la información, se puede reducir significativamente el riesgo existente en este tipo de entornos. Uno de los aspectos básicos en la administración de la seguridad es el establecimiento de una *política de seguridad* que permita manejar eficientemente los recursos disponibles, con un costo razonable.

1.2.1 Políticas de seguridad

Una política de seguridad es un conjunto de normas que establecen qué está (o no) permitido hacer en un sistema o entorno dado y, en su caso, quién está (o no) autorizado a hacerlo. Una política de seguridad aplicada a una organización permitirá a los empleados conocer la manera de conducirse dentro de ella [Cole, 2005]. Una *política explícita* consiste en un conjunto de reglas bien documentadas, mientras que una *política implícita* comprende reglas no documentadas, aunque asumidas por la mayoría [Maiwald, 2002].

Para comprender mejor el concepto y funcionalidad de las políticas de seguridad, definamos más formalmente dos conceptos aparecidos con anterioridad [Anderson, 1980] y que Shirey posteriormente actualiza [Shirey, 2000] [Shirey, 2007]:

- **Vulnerabilidad**: Es un fallo o debilidad en el diseño, funcionamiento, gestión o ejecución de un sistema que puede ser aprovechado para violar la política de seguridad de un sistema.
- **Ataque**: Es un asalto a la seguridad de un sistema que se deriva de una amenaza inteligente, es decir, de un acto inteligente que es un intento deliberado para evadir los servicios de seguridad y violar la política de seguridad de un sistema.

Si un sistema carece de políticas de seguridad resulta imposible definir cuándo un evento observado es o no un ataque, y más aún si éste es ilegal o no. En este contexto, una política de seguridad débil (o incluso inexistente en algunos casos) es la causa principal de ocurrencia de incidentes de seguridad. En este sentido, existen numerosas herramientas que permiten robustecer la seguridad de un entorno en base a la definición de las políticas de seguridad a adoptar. Por mencionar algunas de estas herramientas, citemos los cortafuegos (o *firewalls*) y los anti-virus, los cuales son de generalizada adopción en la actualidad. A continuación se hace un análisis más detallado de este tipo de herramientas.

1.2.2 Herramientas de seguridad

A la par que han evolucionado las redes de computadoras lo han hecho las herramientas de seguridad. Existe en la actualidad un variado conjunto de herramientas para proteger la información de un entorno, ya sean los datos que viajan por la red o la infraestructura misma de ésta.

A continuación se describen brevemente algunas de las herramientas más habituales para la protección de redes y sistemas:

- **Cifrado de datos**: Es un método utilizado para la protección de datos. Consiste en transformar los datos, de forma que una persona no autorizada no sea capaz de entenderlos. Existen diferentes tipos de cifrado, como el asimétrico o el simétrico, entre otros.
- **Protocolos de comunicación segura**: Son mecanismos que permiten establecer canales de comunicación de una manera segura. Por mencionar algunos de los más usuales como *Secure Sockets Layer* (SSL de inglés), este sirve para establecer un canal de comunicación cifrado entre el cliente y el servidor. Un ejemplo es la comunicación cifrada entre el navegador y el servidor web [Oppliger, 2009]. IPsec [Kent & R., 1998] es un mecanismo de seguridad que provee protección basada en cifrar los datos enviados entre dos equipos [Chakrabarti & Manimaran, 2002]. Otro protocolo de comunicación segura es *Pretty Good Privacy* también conocido por sus siglas PGP del inglés. Este protocolo es comúnmente utilizado para el cifrado de mensajes de correo electrónico [Zimmermann, 1995].

- **Cortafuegos:** Un cortafuegos (*firewall*) es básicamente un dispositivo hardware o software encargado de filtrar el tráfico de red, posibilitando así la prevención de accesos no autorizados.
- **Sistemas de detección de vulnerabilidades:** Es una herramienta usada para buscar agujeros, debilidades y vulnerabilidades de seguridad en una computadora o red de computadoras. Una vez realizada la prueba indican las posibles vulnerabilidades encontradas [Stewart, 2008].
- **Sistemas de detección de intrusos (IDS, *Intrusion Detection System*):** Es una herramienta de seguridad que monitoriza eventos dentro de una computadora o red de computadoras, que posteriormente se analizan en busca de intrusiones o intento de ellas [Debar, 2002].
- **Anti-*spamming***: Son herramientas que permiten el filtrado de correos electrónicos no deseados enviados masivamente con contenidos publicitarios o maliciosos que pueden afectar al funcionamiento de una computadora o la red de computadoras [Androutsopoulos et al., 2006] [Salcedo-Campos et al., 2011].
- **Anti-*malware:*** Son programas que detectan códigos maliciosos o mal intencionados que pueden dañar el funcionamiento de una computadora [Kramer & Bradfield, 2009]. Se consideran software malicioso los virus, troyanos y gusanos, entre otros, que alteran el funcionamiento de la computadora.

1.3 Sistemas de detección de intrusos

Anderson publicó un trabajo titulado "*Computer Security Threat Monitoring and Surveillance*" [Anderson, 1980] donde se establecen las bases de la detección de intrusos en sistemas de computadores, principalmente mediante la consulta de archivos de registros de sucesos. Entre 1984 y 1996, Denning y Neumann desarrollaron el primer modelo de IDS descrito en la bibliografía, denominado IDES ("*Intrusion Detection Expert System*") [Denning & Neumann, 1985]. A partir de este momento se han ido proponiendo y creando nuevos sistemas y técnicas de detección de intrusos.

Un sistema de detección de intrusos, o IDS, es una herramienta de seguridad para la detección de ataques a sistemas o redes de computadores. Los IDS analizan la información recolectada con el fin de encontrar incidentes de seguridad. Además, en algunos casos, los IDS permiten el despliegue de respuestas a las violaciones de seguridad [Garcia-Teodoro et al., 2009].

De esta forma, un IDS ha de ser capaz de distinguir entre un acceso "normal" al sistema, que puede surgir de la puesta en marcha de servicios ofertados, y un intento de vulnerar de algún modo dichos servicios.

Por lo tanto, un IDS deberá ser capaz de, al menos, generar alertas en todas aquellas situaciones que puedan ser consideradas como eventos de intrusión. En suma, el objetivo de los IDS es detectar y alertar sobre el mal uso realizado por usuarios

legítimos de los sistemas de información, el abuso de privilegios o la explotación de vulnerabilidades [Debar et al., 1999].

1.3.1 Arquitectura de los IDS

A continuación se presentan diferentes arquitecturas adoptadas para el desarrollo de IDS, describiéndose brevemente cada uno de sus componentes, para posteriormente pasar a su clasificación de acuerdo a sus características.

Existen varias propuestas sobre la arquitectura de los IDS, las cuales se han ido modificando y mejorando con el paso del tiempo [Northcutt & Novak, 2001] [Joycee et al., 2013]. En 1998 DARPA creó el grupo de trabajo CIDF ("*Common Intrusion Detection Framework*"), con el propósito de coordinar diversos proyectos en el campo de los IDS y desarrollar un marco de trabajo común para asegurar la compatibilidad de los mismos. Básicamente, las arquitecturas consideradas en el desarrollo de IDS se refieren al uso de agentes autónomos y la exploración de datos en tiempo real.

CIDF ("Common Intrusion Detection Framework")

La primera de las arquitecturas propuestas se describe en las siguientes líneas como un intento de estandarización de los IDS. Aunque no logró su aceptación como estándar, sí se estableció como un modelo de referencia sobre las intrusiones. Actualmente existe un grupo de trabajo involucrado en este modelo, el IDWG ("*Intrusion Detection Working Group*") de la IETF.

La arquitectura propuesta en el CIDF se desarrolló para solventar algunas de las carencias de los primeros IDS, donde los datos eran recogidos por un solo *host*. Algunas soluciones pasaban por recoger datos de forma distribuida, pero siendo analizados por una única consola. Los principales componentes de esta arquitectura son cuatro [Porras & Neumann, 1997]:

1) Generadores de eventos, también llamados sensores (bloques E), cuyo trabajo es monitorizar eventos en el sistema a proteger.
2) Analizadores de eventos (bloques A), encargados de recibir los eventos y realizar un análisis sobre los mismos. Los bloques A generarán los informes resultantes del análisis realizado, pudiendo ofrecer una acción recomendada.
3) Bases de datos de eventos (bloques D), donde se pueden almacenar temporalmente eventos e informes.
4) Módulos de respuesta (bloques R), que toman los resultados de los componentes antes descritos y responden al evento o eventos detectados con objeto de proceder a su solución.

En la Figura 1.1 se ilustran los distintos bloques constituyentes de un IDS y los posibles flujos entre los mismos. El número de bloques que integran un IDS y las

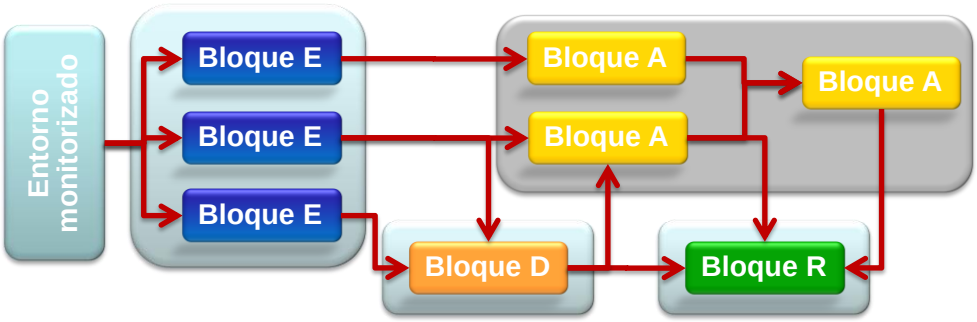


Figura 1.1: Modelo CDIF de un sistema IDS

relaciones entre ellos dependen del entorno en el que éstos se implementan. La contribución del grupo de trabajo CIDF ha sido la definición básica de las interfaces para la comunicación entre los diferentes bloques. Esta característica permite la construcción de sistemas IDS complejos a partir de bloques individuales que pueden integrar diversos mecanismos de captura y almacenamiento de eventos, algoritmos de detección y mecanismos de respuesta.

Para que los distintos componentes se comuniquen es necesario un lenguaje, el denominado CISL ("*Common Intrusion Specification Language*") [Feiertag et al., 1999]. CISL permite la transmisión de información de eventos en bruto; el tráfico de red y la auditoria de registros; la generación de los resultados del análisis, en los cuáles se describen las anomalías del sistema y los ataques detectados; y la prescripción de posibles respuestas.

Arquitectura en tiempo real

Lee y otros desarrollaron la arquitectura de exploración de datos en tiempo real. Las características que aporta esta arquitectura son la eficiencia, la exactitud y la facilidad de uso, así como la veracidad con la que se procesan los datos [Lee et al., 2001].

La característica de eficiencia está relacionada con la necesidad de que el IDS detecte los ataques sin error, y en el menor tiempo posible. Sin embargo, cuando hay grandes flujos de datos a analizar, el tiempo para el procesamiento de cada paquete y el cálculo de las estadísticas relativas a sus características es demasiado elevado. Dicho retraso puede posibilitar la ocurrencia de otros ataques.

La característica de veracidad se refiere a los ataques que puede detectar el IDS y a los falsos positivos, que es el porcentaje de eventos normales que el IDS determina erróneamente como intrusiones.

Las principales ventajas de la arquitectura en tiempo real (Figura 1.2) son sus altas prestaciones y escalabilidad, debido a que los componentes pueden residir en alguna red local, distribuyendo la carga de trabajo entre todos los componentes, o en diferentes redes, lo cual puede permitir la colaboración de otros sistemas IDS [Lee et al., 2001].

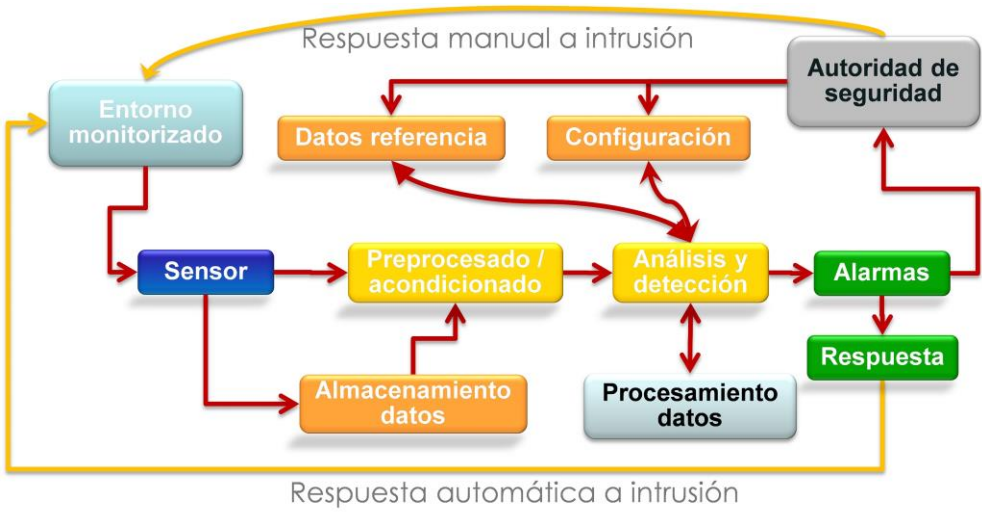


Figura 1.2: Arquitectura IDS en tiempo real [Axelsson, 1998], revisada

1.3.2 Clasificación de los IDS

Los IDS se pueden clasificar de acuerdo a varios criterios [Axelsson, 1998] [Estévez-Tapiador et al., 2004] [Patcha & Park, 2007]. Atendiendo a la fuente de información considerada para la recopilación de los datos sobre los que llevar a cabo el proceso de detección, un IDS puede ser *de host* (HIDS, *Host-based IDS*) o *de red* (NIDS, *Network-based IDS*). En cambio, si atendemos al tipo de procesamiento llevado a cabo en la detección, un IDS puede ser *basado en firmas* (S-IDS, *Signature-based IDS*) o *basado en anomalías* (A-IDS, *Anomaly-based IDS*).

En lo que sigue se describe en mayor profundidad cada uno de los distintos tipos de IDS mencionados.

Detección de intrusos basada en host y en red

Cuando las primeras herramientas de detección fueron diseñadas, el entorno de trabajo era una computadora y los usuarios locales al sistema considerado. Por ello, la detección de intrusos basada en *host* fue la primera área explorada en el campo de los IDS. En este caso, la detección se basa en el uso de archivos de auditoría locales, los cuales contienen llamadas al sistema, con órdenes del sistema operativo [Debar et al., 1999].

Con el paso de sistemas individuales a sistemas en red, las primeras investigaciones en el ámbito de NIDS se orientaron a tratar de conseguir que los sistemas de detección de intrusos basados en *host* se comunicaran entre sí. Este intercambio de información se puede llevar a cabo a varios niveles, estando principalmente referido a datos propiamente dichos o a alarmas locales.

Con el uso generalizado de Internet, los sistemas de detección de intrusos se han focalizado hacia los ataques a la red misma o a través de la red. Para ello, las herramientas específicas desarrolladas monitorizan los paquetes de red en tiempo real en busca de posibles ataques. Estas herramientas resultan atractivas para los administradores de sistemas, ya que se pueden instalar en puntos estratégicos de la red.

IDS basados en host (HIDS)

Las primeras propuestas para la detección de intrusiones se basaban en el uso de datos de auditoría de una máquina [McHugh, 2001]. En ITESEC [SOG-IS, 1991] se exponen los criterios a considerar para el análisis de una máquina, comenzando con la clase C2. Entre éstos se incluye la identificación y autenticación (inicios de sesión, etc.), el espacio de direcciones del programa (apertura de archivo y ejecuciones de programa), la eliminación de objetos, las acciones relativas a la administración y otros eventos de seguridad que se consideren pertinentes.

La mayoría de los HIDS recogen los datos de forma continua a partir del sistema operativo. Estos datos se refieren principalmente a los siguientes aspectos [Debar et al., 1999] [Sobh, 2006]:

a) **Sistema**: El propio sistema donde se encuentra instalado el IDS provee datos acerca de su evolución: llamadas al *kernel*, ejecución de comandos del sistema operativo, etc.; es decir, comandos que se están procesando en cada instante en la computadora. Es de resaltar a este respecto la dificultad de darle continuidad a la detección basada en estos eventos, debido a la ausencia de una vía estructurada para el almacenamiento de este tipo de información.
b) **Contabilidad** (*accounting*): Constituyendo una de las fuentes de información más habituales acerca del comportamiento de un sistema, a través del *accounting* se obtiene información de los recursos utilizados por los usuarios del sistema: tiempo de procesamiento, uso de la red, aplicaciones compartidas, etc.
c) **Ficheros de trazas:** La ejecución de procesos en una máquina puede dar lugar a ficheros de traza (o *logs*), donde se almacena información diversa acerca de las circunstancias de la ejecución: marcas de tiempo, resultados parciales y finales, etc. No obstante su versatilidad, el uso de *logs* ralentiza el funcionamiento general del sistema sobre él sustentado.

La toma de decisiones a partir de la (ingente) información anteriormente citada supone un gran reto para el funcionamiento en tiempo real de un HIDS. Por el contrario, una recolección insuficiente de datos aumentaría el riesgo de obviar las manifestaciones de posibles ataques.

IDS basados en red (NIDS)

Una alternativa a la detección basada en *host* es observar el tráfico que va hacia y desde el sistema o sistemas conectados en red en busca de signos de intrusión. Este enfoque tiene la ventaja de que un solo sensor, correctamente colocado, puede supervisar varias máquinas simultáneamente. Por el contrario, presenta la desventaja de que no puede detectar ataques como los realizados desde la consola de un sistema o aquellos que no atraviesan el segmento de red monitorizado. En esta misma línea se sitúa la dificultad de monitorizar comunicaciones cifradas. A pesar de todo ello, la detección basada en red es la opción más usada en las soluciones IDS actuales.

A continuación se presentan algunas alternativas por lo que se refiere a la recolección de datos por parte de un sensor de red:

a) **Información de gestión SNMP** ("*Simple Network Management Protocol*") [Debar et al., 1999]: La base de datos de información de gestión (MIB, "*Management Information Base*") es un repositorio que contiene información diversa relacionada con las capas y los protocolos de la red, desde la configuración de ésta (tablas de rutas, direcciones de interfaces, etc.) hasta datos de *accounting* de distintos protocolos (paquetes ICMP emitidos/recibidos, etc.).

   Inicialmente se optó por examinar los contadores a nivel de interface, por ser el lugar donde se puede diferenciar entre la información de red y la información del sistema operativo. El sistema recogía incrementos en el número de *bytes* y paquetes enviados y recibidos en cada interfaz cada cierto tiempo. El resultado del estudio de la variación en estos números no fue muy satisfactorio.

   Por su parte, los contadores MIB relacionados con capas más altas de la red no contienen mucha más información discriminativa. Así, en [Debar et al., 1999] se especifica que los contadores en capa TCP/UDP no permiten obtener correlaciones de interés.

b) **Información de red**: La popularidad de los rastreadores de red (*sniffers*) para la recopilación de información sobre los eventos que suceden en una red ha crecido de modo significativo en los últimos años. Esto es consistente con la adopción generalizada de sistemas distribuidos frente a centralizados; y el ritmo de ésta se ha incrementado con la penetración de Internet. La mayoría de los accesos a los equipos sensibles tiene lugar actualmente en la red, por lo que la captura de la información antes de "entrar" en el *host* es probablemente la forma más eficaz de seguimiento de la operación del mismo.

   No obstante el uso generalizado de *sniffers* para la captura de tráfico de red, su utilización plantea algunos problemas que hay que tener presentes:

- Algunas de estas herramientas analizan la carga útil de los paquetes, lo que permite la detección de ataques en base a la observación de patrones conocidos. Sin embargo, un análisis eficiente requiere el conocimiento del tipo de máquina o aplicación a las que están destinados los paquetes.
- En línea también con el análisis de la carga útil de los paquetes, el uso cada vez más generalizado de cifrado en las comunicaciones dificulta (cuando no impide) el proceso de análisis y detección.
- El empleo de redes conmutadas hace de la localización específica de los sensores de tráfico un asunto de gran transcendencia. Una disposición "inadecuada" de los mismos puede restringir la información capturada a la propia de cada *host*, lo que limita la búsqueda de correlaciones de eventos entre máquinas de un mismo entorno y, consecuentemente, la capacidad de detección del IDS a desplegar.

Finalmente, hemos de mencionar que los paquetes de red son la fuente de información de numerosos productos IDS comerciales, como *IBM Security Network Intrusion Prevention System* de IBM, *Next Generation Intrusion Prevention System* de Cisco o *Network Edge Services* de Juniper, existiendo además diversos proyectos de investigación en esta línea con una larga trayectoria que aún siguen activos [Paxson, 1999] [Staniford-Chen et al., 1996].

Detección de intrusos basada en firmas y en anomalías

Como se ha indicado al inicio de la Sección 1.3.2, el tipo de detección llevada a cabo por un IDS da lugar a la clasificación de éstos como basados en firmas (S-IDS, sean éstos de *host*, S-HIDS, o de red, S-NIDS) o basados en anomalías (A-IDS, bien A-HIDS o A-NIDS). Seguidamente se comentan las características de cada uno de estos tipos.

IDS basados en firmas (S-IDS)

La operación de los S-IDS se sustenta en la consideración de un conjunto predefinido de firmas de ataques. De este modo, los sistemas de detección de intrusos basados en firmas buscan la aparición de patrones establecidos a priori en una base de datos dentro de los datos analizados. En otras palabras, las decisiones se toman sobre la base de un conocimiento previo adquirido.

Uno de los principales beneficios del uso de la detección basada en firmas es que los ataques conocidos pueden detectarse de manera fiable y rápida, con una baja tasa de falsos positivos (eventos legales detectados como intrusivos). Para los administradores es fácil determinar en estos casos exactamente cuándo está siendo atacado el sistema y obtener las secuencias de ataque específicas. Por el contrario, uno de los mayores inconvenientes presentes en este tipo de sistemas es la necesidad de

Ventajas	Desventajas
• Alta tasa de detección. • Muy baja tasa de falsas alarmas.	• Coste asociado a la definición y actualización de la base de reglas (firmas). • No pueden detectar intrusiones desconocidas.

Tabla 1.1: Ventajas y desventajas de los IDS basados en firmas [Sobh, 2006]

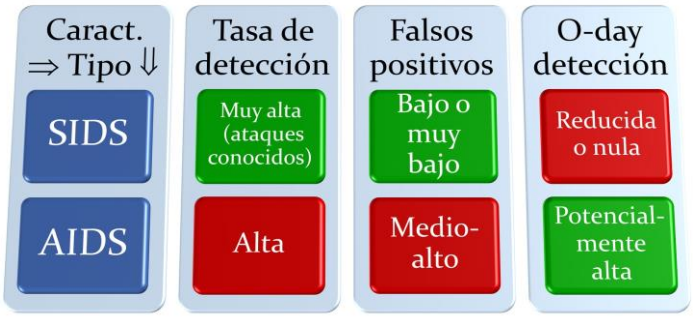


Figura 1.3: Comparación de prestaciones de los SIDS y AIDS

mantener debidamente actualizada la base de datos de firmas. Más allá de este hecho, las firmas suelen ser excesivamente rígidas, de manera que cualquier ataque no contenido en la base de datos de firmas, aunque se trate de una simple variante de alguno/s ya conocido/s, será inevitablemente obviado por el S-IDS.

A modo de resumen, en la Tabla 1.1 se indican los pros y contras más relevantes que presentan este tipo de sistemas.

IDS basados en anomalías (A-IDS)

Un sistema de detección de intrusos basado en anomalías (A-IDS) parte de la estimación de un modelo de referencia acerca del comportamiento normal del entorno a proteger. A partir de ese punto, cualquier actividad observada que se desvíe un cierto grado de dicho perfil de normalidad se considerará como una posible intrusión. Como los S-IDS, los A-IDS presentan ciertas ventajas y desventajas. En primer lugar, cabe mencionar que los A-IDS tienen la capacidad teórica de detectar ataques no conocidos a priori, frente a lo que sucede con los S-IDS. Por el contrario, es de destacar que el modelo de normalidad en los sistemas A-IDS actualmente existentes suele ser excesivamente genérico, lo que suele provocar altas tasas de falsos positivos.

A modo de conclusión, en la Tabla 1.2 se resumen las características principales de los sistemas A-IDS. A este respecto, es de mencionar que sus ventajas son, básicamente, las desventajas de los S-IDS, y viceversa. En la Figura 1.3 se comparan las prestaciones de ambos tipos de sistemas en relación a tres propiedades relevantes: la

Ventajas	Desventajas
• No es necesario ningún conocimiento a priori de los ataques. • Capacidad de detección de nuevos ataques.	• Altas tasas de falsos positivos. • Susceptible a cambios de perfiles y pérdida de la efectividad en entornos dinámicos.

Tabla 1.2: Ventajas y desventajas de los IDS basados en Anomalías [Sobh, 2006]

tasa de detección, la tasa de falsos positivos y la capacidad de detección de nuevos ataques (ataques de día cero, o *0-day*).

Detección de intrusos híbrida

Los IDS anteriores pueden combinarse entre sí para dar lugar a lo que se conoce como *IDS híbridos*. Así, podemos combinar un IDS, de *host* o de red, basado en firmas o en anomalías, con otros IDS, de *host* o de red, basados en firmas o en anomalías, de cara al robustecimiento de la detección en un entorno dado. En suma, en un IDS híbrido se toman las fortalezas de unos esquemas para cubrir las debilidades de otros [Aydin et al., 2009]. Se persigue así un doble objetivo: por una parte mejorar el rendimiento del IDS global y por otra reducir la tasa de falsos positivos.

Díaz-Verdejo y otros han desarrollado una aproximación que ejemplifica claramente los IDS híbridos. Así, en [Díaz-Verdejo et al., 2007] se presenta un NIDS híbrido basado en firmas y en anomalías a partir del S-NIDS de dominio público Snort [Roesch, 1998-2015]. La aproximación no solo no degrada el rendimiento del sistema de detección de intrusos basado en firmas, sino que mejora la detección global de ataques en base al análisis de la carga útil de las peticiones HTTP. Adicionalmente, se mejora la tasa de detección de Snort o, en el peor de los casos, se iguala esta.

Otro ejemplo del uso de la combinación de los métodos de detección basados en firmas y en anomalías es el presentado por Ding y otros [Ding et al., 2009]. En él se utilizan tres sub-modelos: uno basado en firmas, un módulo de anomalías y un módulo de generación de firmas, usando como base *Snort*. Evaluado el sistema híbrido con la base de datos DARPA 1999, se consigue una tasa de ataques detectados del 94%. Las pruebas realizadas con el sistema híbrido demuestran un mejor rendimiento cuando estas se llevan a cabo fuera de línea, empeorando cuando se lleva a cabo un procesamiento en línea.

Otro trabajo similar en cuanto a los bloques constitutivos es [García-Teodoro. et al., 2015], en el que se propone la combinación de un AIDS y un SIDS para extraer firmas de nuevos ataques tras ser detectados por el AIDS.

Para concluir, es también importante señalar que algunos investigadores han implementado IDS combinando HIDS y NIDS [Garcia-Teodoro et al., 2009]. Esto resulta en la detección del uso indebido de una máquina junto con posibles intrusiones “externas” a esta.

Figura 1.4: Clasificación de ataque / lícito

1.4 Evaluación de IDS

Una cuestión de especial relevancia en el campo de la detección de intrusos es la que se refiere a la evaluación de un IDS. Es decir, la determinación de si una cierta metodología IDS es mejor, y porqué, que otras.

Diversos son los aspectos que hay que considerar a este respecto. Por una parte, en trabajos como [Porras & Porras, 1998] y [Lazarevic et al., 2005] se propone usar los siguientes dos parámetros para evaluar la eficiencia de un IDS:

- **Eficiencia**: La eficiencia se refiere a la detección correcta de los ataques y la ausencia de falsas alarmas. Por el contrario, un IDS resulta ineficiente cuando detecta una acción legítima como anómala o intrusiva, o una ilegítima como "normal".
- **Rendimiento**: El rendimiento de un sistema de detección de intrusos se refiere a su capacidad de procesamiento. Si el rendimiento de un IDS es pobre, la detección no será posible en tiempo real. Así, un IDS ha de realizar su análisis y generar el resultado lo más rápido posible, de manera que el evento o eventos intrusivos detectados puedan ser convenientemente tratados antes de que el posible daño se haga efectivo.

Adicionalmente a las anteriores, ciertos autores proponen otras propiedades a evaluar en un IDS. Concretamente, en [Debar et al., 1999] se plantea la tolerancia a fallos como un aspecto relevante. Un IDS debe ser resistente a los ataques, especialmente a los de denegación de servicio. Esto resulta de suma importancia, ya que un IDS opera sobre un sistema operativo y, como tal, puede verse afectado por las vulnerabilidades y/o ataques asociados al mismo.

Sin subestimar la importancia de otras propiedades, el aspecto más relevante de un IDS es su eficiencia en la detección. En este sentido, cuatro son las posibles situaciones que pueden aparecer como resultado de un proceso de detección (Figura 1.4):

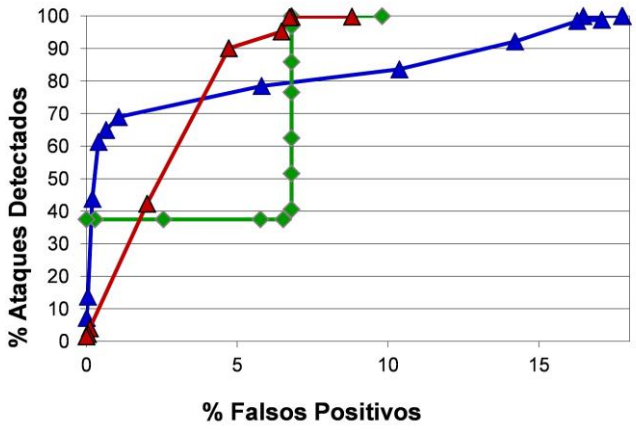


Figura 1.5: Ejemplo de curva ROC

- Evento intrusivo detectado correctamente como tal. Es lo que se conoce como verdadero positivo o TP, del inglés "*True Positive*".
- Evento legítimo detectado correctamente como tal. Es lo que se conoce como verdadero negativo o TN, del inglés "*True Negative*".
- Evento intrusivo detectado incorrectamente como legítimo. Es lo que se conoce como falso negativo o FN, del inglés "*False Negative*".
- Evento legítimo detectado incorrectamente como intrusivo. Es lo que se conoce como falso positivo o FP, del inglés "*False Positive*".

Es evidente que los valores deseados para la consecución de un IDS de altas prestaciones son alta tasa de TP y TN y bajos valores de FP y FN. Esos valores permitirán medir el rendimiento del IDS en términos de capacidad de detectar eventos intrusivos. La curva ROC (*Receiver Operating Characteristic* del inglés) [Hanley & McNeil, 1982] permitirá comparar la capacidad de detección de intrusos frente a la tasa de falsos positivos. Esta es una gráfica paramétrica en la que el parámetro a ajustar es la sensibilidad del sistema a lo que percibe como un comportamiento inseguro. Así, la curva ROC (Figura 1.5) representa la probabilidad de detectar un evento intrusivo frente a la probabilidad de que un evento normal sea mal clasificado variando un parámetro en particular, como puede ser un umbral de detección. Mediante los valores representados en la curva ROC se puede visualizar y evaluar el rendimiento del IDS en términos de precisión [Tong et al., 2009].

1.5 Técnicas A-NIDS

Dado que la presente tesis doctoral centra su atención en esquemas de detección de intrusiones en red basados en anomalías, A-NIDS, se considera oportuna una presentación más en detalle de este tipo de sistemas. Este es el objetivo de la presente

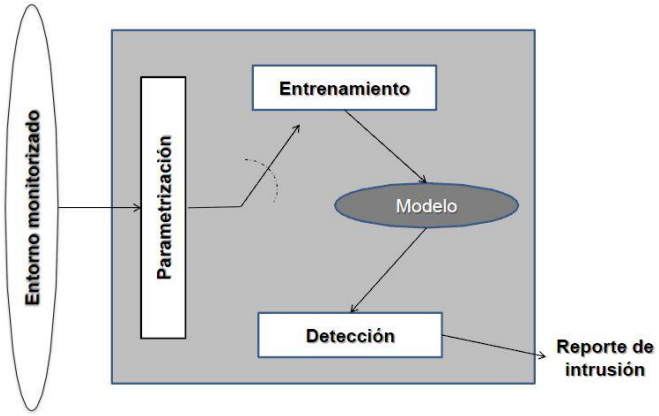


Figura 1.6: Arquitectura de un A-NIDS

sección, donde se trata de reflejar brevemente el estado del arte actual para estas tecnologías.

La complejidad actual de las redes de computadores, así como la amplia tipología del tráfico que circula por las mismas, hace que las técnicas de detección basadas en anomalías para este tipo de entornos sean de naturaleza compleja. Como se indica en [Garcia-Teodoro et al., 2009], tres son las etapas básicas implicadas en el funcionamiento de un A-NIDS (Figura 1.6):

1) **Parametrización**: En esta etapa, las instancias observadas del sistema objetivo son representadas en una forma pre-establecida.
2) **Entrenamiento**: El comportamiento normal (o anormal) del sistema es caracterizado mediante un modelo derivado de un conjunto de observaciones (parametrizadas) de partida. Este modelo puede ser estimado de distintas formas dependiendo del tipo de A-NIDS considerado.
3) **Detección**: Una vez construido el modelo del sistema, éste es usado para estimar el "comportamiento" del tráfico observado (parametrizado). Si la desviación (*score*) encontrada con respecto al modelo excede un cierto umbral, se generará una alarma notificando un incidente intrusivo. A partir de este punto, se podrán poner en marcha las actuaciones que se consideren necesarias para la solución del problema.

De acuerdo con el "tipo de comportamiento" reflejado en el modelo, es decir, al método de modelado, las técnicas A-NIDS pueden ser clasificadas en tres categorías principales (Figura 1.7):

1) A-NIDS estocásticos.
2) Técnicas basadas en especificaciones.
3) Esquemas basados en aprendizaje.

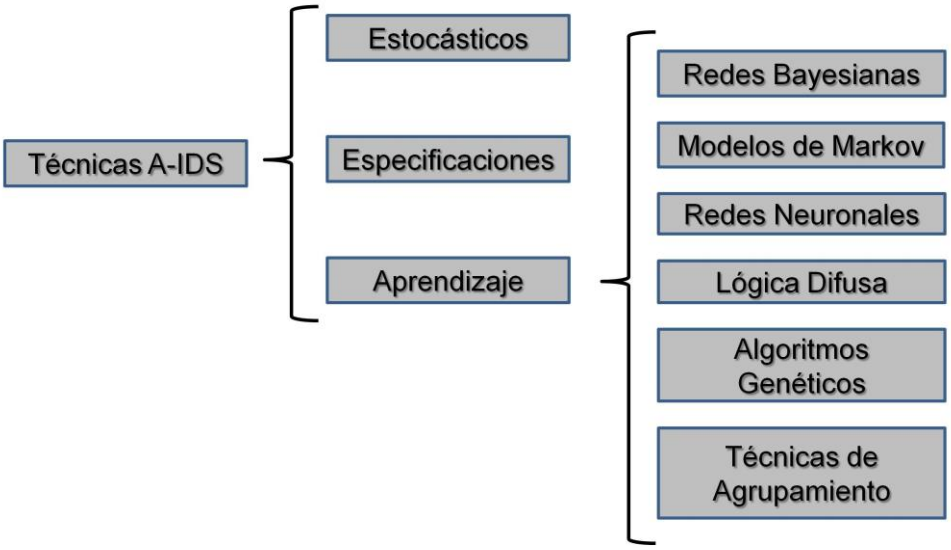


Figura 1.7: Técnicas usadas por los sistemas de detección de intrusos basados en anomalías

Son varios los trabajos en los que se lleva a cabo una clasificación de los sistemas de detección de intrusos. Por mencionar alguna, en [Sobh, 2006] se propone una clasificación de los IDS desde dos perspectivas: la primera basada en la observación de las actividades del sistema, y la segunda en los componentes del sistema de detección.

También en [Garcia-Teodoro et al., 2009] se presenta un listado de las diferentes técnicas utilizadas para la detección de intrusos de red basadas en anomalías (A-NIDS). Además de la clasificación propiamente dicha, en este trabajo se presentan los pros y contras generales de cada una de las técnicas (Tabla 1.3).

A continuación describiremos brevemente las 3 aproximaciones indicadas.

1.5.1 A-NIDS estocásticos

En este tipo de A-NIDS el comportamiento del sistema se modela en base a la estimación de un perfil estadístico. Para ello se recurre a medidas como el número de paquetes correspondientes a ciertos protocolos, la tasa de conexiones, la tasa de tráfico que circula, los números de direcciones IP, etc.

Un ejemplo de los primeros IDS desarrollados con métodos estocásticos es Haystack [Smaha, 1988], el cual utiliza como estrategia de detección de anomalías el comportamiento de los usuarios y grupos de usuarios dentro de un perfil. Este IDS define un rango de valores que se consideran normales para cada función, calcula la distribución de probabilidad de los resultados y genera una alarma si el grado de desviación (*score*) es demasiado elevado. Este IDS fue diseñado para analizar seis tipos

de intrusiones: intentos de suplantación de identidad, ataques de enmascaramiento, penetración del control de accesos, ataques de denegación de servicio, fugas de información y ataques de uso malicioso. Sin embargo, el inconveniente principal de este sistema es su operación sin conexión y la necesidad de un equipo de altas prestaciones para realizar los análisis estadísticos de detección en tiempo real. Adicionalmente, otro problema con el que cuenta es la actualización de los perfiles de usuario.

Otro IDS que utiliza técnicas estadísticas como aproximación a la detección de anomalías fue el desarrollado por el instituto de investigación de Stanford (*Stanford Research Institute*). Fue propuesto a principios de 1980 y se denominó *sistema experto de detección de intrusos* (IDES, "*Intrusion Detection Expert System*") [Lunt et al., 1990]. IDES es un sistema de seguimiento continuo del comportamiento del usuario y de los eventos detectados sospechosos ya ocurridos. A partir de la metodología de IDES, los autores desarrollaron una versión mejorada denominada NIDES ("*New Intrusion Detection Expert System*") [Anderson et al., 1994] [Anderson et al., 1995]. NIDES trabaja en tiempo real y se basa en la consideración de frecuencias de aparición de ciertas variables relativas al perfil de comportamiento. De esta manera, la distribución de frecuencias se representa en forma de histograma con probabilidades asociadas a cada uno de los rangos de valores posibles que puede tomar una variable.

Otro de los ejemplos de uso de técnicas estadísticas para la detección de intrusos es el sistema SPADE [Biles, 2001]. Éste, complemento de SNORT [Roesch, 1998-2015], es utilizado para la detección automática de ataques de escaneo de puertos. Al igual que otros sistemas A-NIDS, SPADE presenta el problema de altos índices de falsos positivos, debido a que clasifica como ataque todos los paquetes no observados previamente.

Krügel y otros muestran en [Krügel et al., 2002] que es posible describir un sistema a partir de la distribución de los datos en el *payload* de los paquetes y de las cabeceras de éstos. De este modo, los caracteres ASCII observados son ordenados en base a su frecuencia de aparición y agrupados en seis grupos.

García-Teodoro y otros han realizado en [Garcia-Teodoro et al., 2009] un estudio y clasificación de las técnicas estadísticas usadas para la detección de intrusos en red. Las primeras aproximaciones estaban orientadas a generar modelos uni-variantes; es decir, se definía un rango de valores aceptables para cada variable considerada en el modelo. Con posterioridad se generaron modelos multi-variantes, donde se tenía en cuenta la correlación de dos o más variables. Otras propuestas, por su parte, consideran modelos basados en series temporales.

El uso de técnicas estadísticas para la detección de anomalías presenta varias ventajas. En primer lugar, no se requiere del conocimiento previo de los ataques y fallas de seguridad. Además, se pueden obtener mediciones precisas de las actividades maliciosas que ocurren típicamente durante largos periodos de tiempo, resultando en buenos detectores de ataques de denegación de servicio (DoS, del inglés "*Denial of Service*"). Otro tipo de actividades intrusivas, como el escaneo de puertos, son detectables mediante este tipo de técnicas debido a que un comportamiento normal difiere sustancialmente de uno intrusivo.

Técnica	• Pros • Contras	Subtipos
A) Estocásticas	• **Pros:** No requiere conocimiento a priori del comportamiento normal. Notificación acertada de actividades maliciosas. • **Contras:** Susceptibles a ser entrenados por atacantes. Dificultad para fijar la parametrización y configuración. Suposición del proceso cuasi-estacionaria poco realista.	A1) Modelos uni-variantes (independiente de variables gausianas aleatorias) A2) Modelos multi-variantes (correlación entre varias métricas) A3) Series temporales (intervalos de tiempo, contadores y algunas otras métricas relacionadas con el tiempo)
B) Basados en especificaciones	• **Pros:** Robustez, flexibilidad y escalabilidad. Habilidad para anticipar conocimiento / datos • **Contras:** Dificultad y alto consumo de tiempo para alta calidad de conocimiento / datos.	B1) Máquinas de estados finitos (estados y transiciones) B2) Lenguajes de descripción (UML, N-gramms, etc.) B3) Sistemas expertos (clasificación basada en reglas)
C) Máquinas basadas en aprendizaje	• **Pros:** Flexibilidad y adaptabilidad. Captura de interdependencias. Categorización de patrones. • **Contras:** Alta dependencia sobre suposiciones acerca del comportamiento aceptado por el sistema. Alto consumo de recursos.	C1) Redes bayesianas (probabilidad relacionada con variables) C2) Modelos de Markov C3) Redes neuronales (fundamentado en el cerebro humano) C4) Algoritmos genéticos (inspirado en la evolución biológica) C5) Agrupamiento

Tabla 1.3: Clasificación de técnicas detección de intrusos en red basadas en anomalías [Garcia-Teodoro et al., 2009]

Sin embargo, los esquemas de detección de intrusos estocásticos también presentan algunos inconvenientes. La posible consideración de eventos anómalos a la hora de estimar el modelo de normalidad podría variar la estadística considerada, lo que provocaría errores de detección. También puede ser difícil determinar los umbrales de detección de modo que se establezca un equilibrio entre las tasas de falsos positivos y de falsos negativos. Por otra parte, no todos los sistemas pueden ser modelados mediante procesos estadísticos puros, sino que en ocasiones se requiere la consideración de procesos cuasi-estacionarios.

1.5.2 A-NIDS basados en especificaciones

El uso de sistemas expertos es ampliamente considerado en el desarrollo de esquemas A-NIDS basados en especificaciones. En éstos se lleva a cabo un proceso de detección basado en reglas que implica tres pasos:

- A partir de la base de datos de entrenamiento se obtienen diferentes atributos y clases.
- De ellos se deduce un conjunto de reglas, parámetros o procedimientos.
- Finalmente, los eventos observados son clasificados a partir de la comprobación de dichas reglas.

Este tipo de A-NIDS pasa pues por la especificación de un conjunto de reglas por parte de un experto humano. La eficiencia de estos modelos depende de la completitud de las reglas dadas para especificar el comportamiento legítimo o no del sistema, de modo que este tipo de esquemas presenta una baja tasa de falsos positivos, al tiempo que no reporta alarmas ante eventos no observados con anterioridad.

Una de las herramientas más ampliamente utilizadas para especificar las reglas es el empleo de autómatas de estados finitos asociados a protocolos/servicios de red [Estévez-Tapiador et al., 2004]. La utilización de lenguajes como UML para abordar esta descripción puede encontrarse en trabajos como [Zhu & Zulkernine, 2007].

Una de las ventajas principales de los esquemas A-NIDS basados en especificaciones es su robustez y flexibilidad. Sin embargo, su principal desventaja, común a otras técnicas A-NIDS, es la dificultad de disponer de modelos de calidad.

1.5.3 A-NIDS basados en aprendizaje

Una de las maneras de mejorar el funcionamiento de un sistema de IDS es el uso de las técnicas denominadas *machine-learning*, o basadas en aprendizaje. En estas técnicas se parte de la disposición de un modelo o perfil de normalidad que es establecido durante una fase de entrenamiento o aprendizaje. A partir de ello, se pretende mejorar sus prestaciones desde una doble vertiente: la reducción de la tasa de falsos positivos y la reducción del coste computacional involucrado. Para ello se puede reestimar el sistema a partir de la disposición de nuevos datos de entrenamiento.

Seguidamente se mencionan las variantes más importantes de las técnicas de *machine-learning*.

Redes bayesianas

Una red bayesiana es un modelo gráfico que codifica relaciones probabilísticas entre las variables de interés. Cuando las redes bayesianas se utilizan junto con técnicas estadísticas tienen varias ventajas para el análisis de datos [Heckerman, 1996]. Entre éstas se encuentra la capacidad de codificar interdependencias entre las variables y de

predecir eventos, así como la capacidad de incorporar tanto conocimiento como datos. Dado que las redes bayesianas pueden representar sistemas causales, son susceptibles de ser usadas para predecir las consecuencias de una acción.

Algunos investigadores han usado estadística bayesiana para crear modelos para la detección de anomalías, como [Kruegel et al., 2003] y [Valdes & Skinner, 2000] que presentan un modelo basado en redes bayesianas para la detección de ataques con un alto desempeño, llamado eBayesTCP. Otro sistema basado en redes bayesianas es EMERALD [Porras & Neumann, 1997], el cual tiene la capacidad de detectar ataques distribuidos cuando cada sesión individual de ataque no es lo suficientemente sospechosa para generar una alerta.

Las técnicas bayesianas han sido frecuentemente utilizadas para la clasificación y supresión de falsas alarmas [Altwaijry & Algarny, 2011]. En esta línea, la cualidad más citada de las redes bayesianas para su uso en IDS es su capacidad para incluir información a priori [Scott, 2004].

Modelos de Markov

Desde hace algún tiempo, los comportamientos estadísticos de las redes de computadores han dado lugar a la propuesta de múltiples modelos del tráfico. Sin embargo, este no es un proceso sencillo. Una aproximación exitosa en esta dirección ha sido la utilización de modelos de Markov, los cuales han demostrado su alta eficiencia en la detección de intrusos debido a la capacidad de capturar características relevantes que no es posible determinar mediante otro tipo de aproximaciones.

Ye y otros llevaron a cabo en [Ye et al., 2004] una investigación realizada con la técnica de cadenas de Markov a fin detectar ciberataques. El sistema propuesto demuestra un buen desempeño, aunque conforme aumenta el tamaño de la ventana del umbral considerado, aumenta la cantidad de ruido y, consecuentemente, empeora la detección.

Otro ejemplo del uso de la técnica de las cadenas de Markov con el fin de la detección de intrusos es [Estevez-Tapiador et al., 2005]. En este trabajo se construye un modelo de comportamiento normal a partir de la observación de símbolos en la carga útil de los paquetes correspondientes al protocolo HTTP. Siendo este sistema evaluado con la base datos DARPA [Lippmann et al., 2000], los resultados de detección obtenidos son buenos, resultando ser eficiente y escalable a redes de alta velocidad.

Por su parte, Jha y otros presentan en [Jha et al., 2001] un marco teórico para el desarrollo de sistemas de detección de intrusos usando modelos de Markov. Además de los buenos resultados de detección conseguidos, los autores presentan algunas directrices futuras en el uso de este tipo de modelado.

Una variante de los modelos de Markov es la relativa a los modelos ocultos de Markov (HMM, "*Hidden Markov Models*"). Esta aproximación tiene propiedades similares a los modelos de Markov, si bien las transiciones entre los estados se desconocen, de ahí el término "oculto"; sólo se conocen los resultados de las transiciones. Como ejemplo del uso de esta variante, Yeung y Ding usan en [Yeung &

Ding, 2003] una aproximación dinámica basada en los modelos ocultos de Markov y el principio de máxima probabilidad.

Los inconvenientes generales del uso de esta técnica para la detección de intrusos se refieren a su, por lo general, elevado coste computacional y a la dificultad de disponer de suficientes muestras representativas para el entrenamiento. Así, habida cuenta del enorme flujo de datos que circula actualmente por las redes, difícilmente se puede desplegar un modelo escalable en tiempo real para este tipo de entornos. A pesar de sus inconvenientes, las técnicas que incluyen el modelado de Markov se continúan utilizando con éxito en la detección de intrusiones [Sharma & Manoria, 2015].

Redes neuronales

Debido principalmente a su flexibilidad y adaptabilidad, la técnica de redes neuronales ha sido adoptada por los investigadores como una técnica más para la detección de intrusos.

Una red neuronal se construye sin ningún conocimiento de inicio, pero puede ser entrenada para tomar decisiones por asignación de pares de prototipos de entrada-salida, ajustando las interconexiones y los pesos de los elementos de la red (conocidos como *neuronas*) a fin de que cada prototipo de entrada corresponda aproximadamente con el prototipo de salida correspondiente [Hecht-Nielsen, 1988].

Las redes neuronales representan, así, un conocimiento básico distribuido en forma de interconexiones ponderadas, utilizándose un algoritmo de aprendizaje para modificar la base de conocimiento dispuesto.

La principal ventaja de las redes neuronales es su tolerancia a la imprecisión de los datos e información incierta, así como su capacidad de deducción a partir de los datos sin tener conocimiento de regularidades en los mismos. Esto, combinado con su posibilidad de generalización a partir de los datos aprendidos, ha demostrado su capacidad de uso en la detección de intrusiones [Patcha & Park, 2007]. Sin embargo, las soluciones IDS basadas en redes neuronales presentan algunos inconvenientes. Por una parte, que es posible no lograr una solución satisfactoria por falta de datos o porque no existe una función para llevar a cabo el aprendizaje. Por otro lado, que las redes neuronales son lentas y de alto coste en su entrenamiento.

En todo caso, hay grupos de investigación actualmente tratando de dar solución a las limitaciones antes mencionadas de cara a la mejora de esta tecnología [Yao et al., 2006] [Al-Jarrah & Arafat, 2015]. Así, en uno de los primeros trabajos en este contexto, Ghosh y otros presentan en [Ghosh & Schwartzbard, 1999] una aplicación de la técnica de redes neuronales para la detección de futuras intrusiones a sistemas en red. En este trabajo se utiliza el modelo clásico multicapa para el diseño de una red neuronal de retro-propagación.

Otro ejemplo de uso de las redes neuronales en sistemas de detección de intrusos es [Yao et al., 2006]. En este trabajo se presenta un sistema de detección híbrido que hace uso de un esquema perceptrón multicapa (MLP, del inglés *MultiLayer*

Perceptron) y redes neuronales caóticas (CNN, del inglés *Chaotic Neural Network*), el cual consigue una alta tasa de detección y una baja tasa de falsos positivos.

Bivens y otros también desarrollaron un A-NIDS usando redes neuronales [Bivens et al., 2002]. En este caso, los autores abordan la detección de tres tipos de ataques: denegación de servicio (DoS), denegación de servicio distribuido (DDoS, del inglés *Distributed Denial of Service*) y escaneo de puertos. El proceso completo llevado a cabo por el sistema es el siguiente: (a) captura de tráfico con *tcpdump*, (b) pre-procesado para la extracción de parámetros tales como el *timestamp* (marcas temporales) de los paquetes, y (c) clasificación. En una primera instancia, el clasificador descarta el tráfico "normal", haciendo eficiente el sistema cuando los grupos de entrada son uniformes. Posteriormente, los eventos dudosos son pasados a través de una red neuronal de tipo MLP para la detección.

Lógica difusa

La lógica difusa (*fuzzy logic*) es apropiada para el problema de detección de intrusos por dos razones principales. En primer lugar, porque en la detección de intrusiones están involucradas muchas características cuantitativas [Bridges et al., 2000], de tipo ordinal (p.e. tiempo de CPU) y lineal (p.e. número de servicios TCP/UDP iniciados desde algún *host*). En segundo lugar, porque precisamente la seguridad involucra procesos difusos.

En la investigación realizada por Bridges anteriormente citada se indica que se puede usar un rango para representar un cierto valor/comportamiento normal. Por su parte, los valores fuera de ese intervalo serán considerados anómalos.

En este contexto, en [Dickerson & Dickerson, 2000] se propone el sistema de detección de intrusos basado en anomalías FIRE ("*Fuzzy Intrusion Recognition Engine*"). Éste hace uso de técnicas de minería de datos para procesar los datos de entrada a la red, empleando métricas significativas para la detección. Estas métricas son evaluadas como conjuntos difusos. FIRE puede detectar una amplia variedad de ataques comunes, permitiendo la combinación de métricas simples de tráfico de red con reglas difusas para la detección de ataques específicos o generales de la red. El sistema ha demostrado ser escalable en un entorno distribuido con múltiples *host*s y/o redes. Shanmugavadivu y otros han desarrollado un NIDS utilizando la técnica de lógica difusa para detectar comportamientos anómalos en la red a partir de las reglas con resultados prometedores [Shanmugavadivu & Nagarajan, 2011].

Algoritmos genéticos

Los algoritmos genéticos fueron introducidos en la década de 1970 por J. Holland. Este tipo de procedimientos son reglas empíricas basadas en los principios de evolución natural y genética, requiriendo que se especifique la codificación de los individuos y una función de evaluación para medir la aptitud de cada uno.

Un aspecto importante de los algoritmos genéticos es que pueden operar concurrentemente sobre un espacio de búsqueda, evitando soluciones locales.

En [Chittur, 2001] se propone un enfoque novedoso para detectar intrusiones haciendo uso de algoritmos genéticos. El algoritmo fue diseñado para que cada individuo de la población representase una posible conducta. Los atributos analizados corresponden a los protocolos TCP, UDP e ICMP, donde cada resultado obtenido fue etiquetado como normal o anómalo. El resultado final fue la obtención de una tasa de detección del 97,8% y una extremadamente baja tasa de falsos positivos.

Banković y otros [Bankovic et al., 2009] presentan un sistema IDS basado en dos algoritmos genéticos. Con el uso de este tipo de técnica demuestran una reducción del tiempo en la detección. También demuestran una disminución en el costo computacional debido a la reducción de los datos procesados. Desafortunadamente, debido a que el conjunto de datos utilizados para el entrenamiento y las pruebas presenta una distribución de datos que no es muy realista (sólo el 20% de los datos corresponde a conexiones normales), no es posible demostrar la capacidad de detectar eventos raros mediante este tipo de técnicas.

En trabajos más recientes, Hoque y otros han usado algoritmos genéticos en un sistema de detección de intrusos para detectar diferentes tipos de intrusiones en la red [Hoque et al., 2012].

Técnicas de agrupamiento

Las técnicas de agrupamiento (*clustering* en inglés) han sido estudiadas en diversas aproximaciones como solución al reconocimiento de voz, identificación de tráfico, etc. Aunque existen múltiples variantes y técnicas, dos son los métodos de agrupamiento principales usados: el primero, el denominado *K-medias* (*K-means* en inglés) y el segundo, el llamado *agrupamiento de conexión simple* (*simple linkage clustering*).

K-medias consiste en dos pasos iterativos: en primer lugar se reasignan todos los puntos a sus *centroides* más cercanos, y en segundo lugar se recalculan los *centroides* de los nuevos grupos creados en el paso anterior. El proceso continúa hasta que los *centroides* no sufran un cambio significativo en la siguiente iteración [Laskov et al., 2005]. El algoritmo de agrupamiento K-medias es el referente principal entre los diversos métodos usados para seleccionar grupos representativos de datos.

En lo que respecta a la detección de intrusiones, Portnoy y otros presentan en [Portnoy et al., 2001] un trabajo donde utilizan métodos de agrupamiento usando el algoritmo K-medias. En su investigación describen dos ventajas principales del uso de este método frente a los clasificadores basados en especificaciones o en algoritmos de aprendizaje. La primera es que no se precisa una clasificación manual de los datos en la fase de entrenamiento. La segunda es que no es necesario conocer a priori los nuevos tipos de intrusiones con objeto de su detección posterior. El sistema planteado presenta una tasa de detección baja en comparación con los sistemas que utilizan algoritmos basados en datos etiquetados, pero consideran que aun así el sistema sigue siendo muy útil.

En [Wang et al., 2008] se desarrolla un algoritmo donde se combinan el número de atributos y el número de atributos de clasificación para la detección de intrusos basada en anomalías. En el trabajo se hace uso de técnicas de agrupamiento con diversidad mínima, permitiendo la detección tanto de ataques conocidos como no conocidos. Los resultados obtenidos de las pruebas realizadas resultan más efectivos que los conseguidos mediante K-medias.

En [Vaarandi & Podi, 2010] se propone un sistema de detección de intrusos basado en las técnicas de agrupamiento.

1.6 Objetivos y estructura de la tesis

Tras la exposición del contexto general en el que se desarrolla el presente trabajo de tesis, en este apartado se describen los antecedentes, los objetivos y las contribuciones originales del mismo así como la organización de la presente memoria.

1.6.1 Antecedentes y objetivos

En investigaciones previas desarrolladas en el Departamento de Teoría de la Señal, Telemática y Comunicaciones de la Universidad de Granada se ha desarrollado y evaluado un sistema de detección de intrusos basado en red y en anomalías (ANIDS) que usa el modelado de Markov para representar las cargas útiles de protocolos basados en el paso de mensajes [Estevez-Tapiador et al., 2005]. El sistema, denominado SSM (*Structural Stochastic Model*), permite también incorporar las especificaciones del protocolo y ha sido evaluado utilizando la base de datos DARPA99 [Lippmann et al., 2000], habiendo proporcionado excelentes resultados.

El presente trabajo constituye una continuación de dicho trabajo en el que se pretende evaluar las capacidades de dicho sistema en entornos más realistas y proponer y evaluar mejoras al IDS. En este sentido, el sistema de detección de intrusos previo presenta, como se ha mencionado, un rendimiento bastante aceptable con unas altas tasas de detección manteniendo bajas tasas de falsos positivos. Sin embargo, el sistema es susceptible de incorporar algunas mejoras en el modelado que permitan disminuir las tasas de falsos positivos sin afectar las tasas de ataques detectados e incluso mejorar estas últimas. Por otra parte, el sistema de detección fue evaluado originalmente con la base de datos DARPA 99, que, aunque fue una referencia al trabajar con detección de intrusiones durante mucho tiempo, presenta múltiples e importantes limitaciones [McHugh, 2000] [Sommer, 2010] que desaconsejan su uso. Aunque se analizarán con más detalle dichas limitaciones en el Capítulo 2, es relevante indicar que es una base de datos que no ha sido actualizada desde 1999 y, por ende, los ataques y el tráfico en ella contenidos distan de los que se observarían en la actualidad en una red en explotación. Adicionalmente, el tráfico de ataques contenido en ella es insuficiente en número y en tipos de ataques. Debido a estas razones, se ha propuesto e implementado una metodología para obtener una base de datos que permita evaluar adecuadamente el sistema de detección de intrusos. Entre las características que debe reunir el tráfico

adquirido para este fin se encuentra el requisito de que sea tráfico real procedente de una red en explotación. Sin embargo, es difícil que las entidades privadas o públicas proporcionen una base de datos con tráfico real para evaluar un IDS debido a cuestiones de privacidad y las implicaciones legales asociadas. Por tanto, se hace necesario desarrollar una metodología que permita anonimizar las trazas de red a la vez que se preserva la información necesaria para la aplicación de la técnica.

Por tanto, el objetivo principal de esta tesis es el desarrollo y evaluación de mejoras al modelado SSM. Para la consecución de este objetivo global, será necesario considerar los siguientes objetivos adicionales:

- Adquirir una base de datos de tráfico real y un conjunto de instancias de ataque, en ambos casos en número suficiente para posibilitar el entrenamiento y evaluación de las técnicas.
- Desarrollar un método de anonimización de las trazas de tráfico adquiridas para posibilitar su uso sin violar la privacidad pero preservando las propiedades e información necesarias para el sistema SSM y las variantes que se propongan.
- Evaluar las capacidades de la técnica SSM con una base de datos adecuada y que responda a las características del tráfico actual.
- Proponer, implementar y evaluar mejoras al modelado SSM.

1.6.2 Contribuciones de la tesis

Contextualizada en la detección de intrusiones en red basada en anomalías, y de acuerdo a los objetivos planteados, la presente tesis doctoral aporta las siguientes contribuciones originales en el desarrollo y mejora de este tipo de sistemas:

- Se propone una metodología para la generación de tráfico de ataques de cara a la evaluación de las prestaciones de sistemas A-NIDS. La base de datos así obtenida incorpora un número elevado de instancias de ataque.

- Se diseña e implementa una metodología para la anonimización del tráfico de red capturado en entornos reales, para que pueda ser usada para evaluar los sistemas de detección de intrusos.

- Partiendo de la disposición de un sistema A-NIDS mixto estocástico y basado en especificaciones (SSM), desarrollado en el equipo de investigación del director de la tesis, se propone la mejora del mismo en diversos aspectos:

    - Suavizado de las probabilidades para grandes vocabularios.
    - Uso de un valor para las observaciones fuera de vocabulario (OOV) dependiente del estado del modelo.

- Desarrollo de un sistema NIDS híbrido a partir del modelado mediante SSM de los ataques y el tráfico normal
- Propuesta de una medida de la confianza en la clasificación realizada por el sistema y entrenamiento discriminativo de los modelos a partir de la misma.

1.6.3 Estructura y organización de la memoria

La presente memoria de tesis se estructura en cinco capítulos adicionales al presente de introducción. Su contenido se indica a continuación.

En primer lugar, para evaluar el rendimiento de un sistema de detección de intrusiones en red son necesarias bases de datos con tráfico de red "normal" (capturado en una red real y libre de ataques) y "de ataque". En este sentido, el Capítulo 2 propone una metodología para la generación de tráfico de ataque y la captura de tráfico en entornos reales que permita evaluar el rendimiento de un sistema IDS. Utilizando esta, y una vez adquirido el tráfico correspondiente, se describen las características más relevantes de las bases de datos así obtenidas.

Para finalizar este capítulo, se ha desarrollado una metodología con el fin de anonimizar la carga útil del protocolo HTTP del tráfico real capturado.

En el Capítulo 3 se describe la técnica básica considerada para la detección de intrusiones, esto es, SSM. Así, en este capítulo se desarrolla el marco teórico de la técnica SSM, procediéndose seguidamente a su evaluación con las bases de datos previamente capturadas. El sistema así obtenido será considerado el sistema de referencia para los posteriores desarrollos y mejoras a realizar, utilizándose para comparar los resultados obtenidos.

Una vez obtenido el sistema de referencia y los resultados correspondientes, en el Capítulo 4 se presentan y evalúan modificaciones de la técnica básica. Para ello se realiza un análisis de las limitaciones y debilidades del sistema. A partir de este, se proponen modificaciones relacionadas con el suavizado de las probabilidades de observación para su adecuación al caso de usar grandes vocabularios. A continuación se propone y evalúa la utilización de probabilidades de fuera de vocabulario diferenciadas por estado a fin de tener en cuenta los diferentes tamaños de los vocabularios asociados a cada uno de ellos. En ambos casos se obtienen resultados satisfactorios de la aplicación de las mejoras propuestas.

En el Capítulo 5 se propone la extensión de la utilización del modelado SSM a los ataques, de forma que se presenta un sistema híbrido que incorpora un modelo de normalidad y un modelo genérico obtenido a partir de los ataques. De esta forma, se sustituye la detección basada en umbral por un sistema de clasificación en el que se asigna la clase más probable a partir de los modelos correspondientes. En este escenario, es posible definir una medida de confianza en la clasificación obtenida a partir de las citadas probabilidades, así como realizar un reentrenamiento del sistema que permita maximizar la probabilidad de la clase correcta.

Finalmente, en el Capítulo 6 se presentan las conclusiones y líneas de trabajo futuro.

2 Escenario experimental

Tal como se ha indicado en el Capítulo 1, existen diferentes técnicas que son utilizadas para la detección de intrusiones en red. Todas ellas necesitan disponer de información previa en base a la que se decide si un evento o conjunto de eventos constituyen una intrusión. Así, en los sistemas basados en firmas, es preciso disponer de las firmas correspondientes a los eventos intrusivos. Por el contrario, en los sistemas basados en anomalías es preciso disponer de un modelo del comportamiento del sistema. Dicho modelo es habitualmente obtenido a partir de una fase de entrenamiento del detector durante la que es necesario contar con un conjunto adecuado y suficiente de datos representativos de la operación normal del sistema.

Adicionalmente, la evaluación del rendimiento del detector desarrollado requiere también de la disponibilidad de datos que permitan determinar sus capacidades, tanto en situación normal, es decir, en ausencia de ataques, como ante la aparición de ataques. En consecuencia, el desarrollo y evaluación de sistemas de detección de intrusiones basados en red y en anomalías requiere la utilización de tráfico de red para las fases de entrenamiento, prueba y validación. Este tráfico debe contener patrones de comportamiento normal y anómalo y representar adecuadamente el tráfico real.

En este contexto surgen habitualmente dos problemas de diferente naturaleza. El primero está relacionado con la cantidad y calidad de los datos necesarios tanto para la estimación del modelo como para su evaluación. El segundo, de índole legal, está relacionado con la privacidad de las comunicaciones, ya que los datos necesarios para las fases anteriormente mencionadas pueden contener información sensible, tanto a nivel individual como corporativo o institucional.

En este capítulo se describe el escenario experimental considerado para el desarrollo de esta tesis, abordándose los problemas asociados a la adquisición del tráfico de red necesario. En este sentido, se propondrán y describirán metodologías para la adquisición y preprocesamiento del tráfico de red, a partir de las cuales se obtendrán bases de datos de tráfico convenientemente etiquetado y categorizado para su uso durante el desarrollo y evaluación de los detectores de intrusiones.

De esta forma, el presente capítulo se estructura en cinco apartados. El primero está dedicado al análisis de las propiedades y requerimientos que deben reunir las bases de datos a utilizar, así como a la evaluación de las técnicas y bases de datos descritas en la bibliografía en el contexto de los sistemas IDS. A continuación, en el Apartado 2.2, se presentará una metodología para la adquisición y etiquetado de tráfico de red que posibilita el uso de los datos capturados para el entrenamiento, evaluación y validación

de los IDS. La aplicación de esta metodología se describe a continuación, obteniéndose las bases de datos que serán utilizadas en este trabajo. Especial relevancia presenta la adquisición de los ataques, que deben ser generados de acuerdo a diferentes aproximaciones que serán descritas en el Apartado 2.3. El tratamiento de los datos obtenidos a fin de evitar los problemas de privacidad será descrito en el Apartado 2.4, en el que se propondrá y aplicará una técnica de anonimización que preserva la información necesaria para el desarrollo de los detectores, a la vez que oculta la información de carácter sensible en ellos contenida. Finalmente, en el Apartado 2.5 se describirán las propiedades más relevantes de los conjuntos de datos capturados.

2.1 Características de los conjuntos de datos

Un elemento fundamental para el desarrollo y evaluación de un sistema de detección de intrusos es disponer de conjuntos de datos que contengan la información necesaria para realizar la detección. Estos datos se usan tanto durante la fase de entrenamiento, para la estimación de los modelos o la determinación de los parámetros de los métodos de detección, como durante la evaluación de su rendimiento. De esta forma, las propiedades o características que presente dicho conjunto de datos resultan de extrema importancia. En particular, resulta relevante que dicho conjunto de datos, que denominaremos base de datos en lo que sigue, cumpla los siguientes requisitos:

- **Representatividad**: El tráfico normal utilizado debe ser suficiente como para obtener un modelo de normalidad adecuado, mientras que el tráfico de ataques debe incluir el mayor número de variantes de los mismos.

  Por otra parte, a fin de que tanto el AIDS resultante como su evaluación sean válidos y representativos de cara a la detección de intrusiones, los datos deben haber sido obtenidos en entornos reales en explotación [McHugh, 2000] [Athanasiades et al., 2003] [Bermúdez-Edo et al., 2006] [Sommer, 2010]. En caso contrario, el IDS no incorporará un modelo adecuado del comportamiento normal del sistema, por lo que su rendimiento no sería el esperado. Adicionalmente, se introducirán sesgos significativos durante la evaluación del rendimiento y, en consecuencia, en la comparación entre las diferentes técnicas desarrolladas o existentes, al falsearse la relación entre la tasa de detección (ataques correctamente detectados) y de falsos positivos (eventos normales detectados como ataques). Sin embargo, y afortunadamente, la proporción de tráfico de ataques en el tráfico circulante es muy baja, lo que dificulta su recopilación. Así, se hace necesaria la adquisición de ingentes volúmenes de datos para obtener un número suficiente de ataques y sus variantes.

  De esta forma, el problema de la adquisición de un número suficiente y representativo de ataques y de tráfico normal resulta de gran dificultad.

- **Etiquetado**: Los datos a utilizar deben estar convenientemente etiquetados en función de su naturaleza (normal / ataque), a fin de evitar que datos de ataque sean utilizados durante el entrenamiento del sistema y la consiguiente inutilización del modelo de normalidad. De esta forma, se debe disponer de dos conjuntos diferenciados de eventos. El primero de ellos estará compuesto por todo el tráfico normal libre de ataques que haya sido capturado, por lo que lo denominaremos *limpio* o *normal*, indistintamente. El segundo conjunto estará compuesto por todos los eventos intrusivos considerados, por lo que lo denominaremos de *ataque*.

  De esta forma, el tráfico adquirido directamente de la red, que denominaremos tráfico *sucio* ya que puede contener ataques, debe ser etiquetado de forma fehaciente. Esto representa un problema práctico cuando, de acuerdo al requisito de representatividad, el tráfico a utilizar se adquiere de una red real en explotación, debido a la dificultad inherente a la detección de los ataques contenidos en dicho tráfico sucio.
- **Particionado**: Finalmente, dado que se va a entrenar un sistema de detección que debe ser evaluado y validado, deben considerarse varios conjuntos de datos que pueden ser obtenidos a partir del particionado de la captura obtenida. De esta forma, se necesitará, al menos, un conjunto de tráfico limpio para el entrenamiento, otro para la evaluación y otro para la validación, en su caso. En general, no será necesario establecer un conjunto de entrenamiento para el tráfico de ataques, ya que no se estima ningún modelo de ataques en los A-IDS, aunque sí se requerirán las correspondientes particiones para evaluación y validación.

Una vez establecidos los requisitos que debe cumplir la base de datos a utilizar, a continuación se presenta una revisión de las bases de datos disponibles que han sido utilizadas por la comunidad científica para el desarrollo de IDS.

Los primeros sistemas IDS fueron evaluados con conjuntos de datos generados o adquiridos por los propios investigadores, sin asignársele una especial relevancia a la misma. En 1998, bajo el auspicio de la agencia DARPA (*Defense Advanced Research Project Agency*), los laboratorios MIT desarrollaron una base de datos con la finalidad de que los resultados obtenidos por cada sistema pudiesen ser comparados. Tras la primera adquisición en 1998, se revisó el proyecto y se realizó una segunda adquisición en 1999. Las bases de datos resultantes, denominadas DARPA/Lincoln Laboratory IDEVAL [Mahoney & Chan, 2003], constituyen una de las bases de datos más ampliamente utilizadas como marco de referencia para evaluar el rendimiento de un IDS. Esta base de datos suele considerarse en sus versiones de 1998, DARPA'98 y de 1999, DARPA'99, por separado. Adicionalmente, dado el elevado volumen de datos que contenía y la dificultad de la tarea, DARPA'98 fue utilizada como base para el desafío KDD'99 [Bay et al., 2000]. La adquisición de los datos se realizó en un entorno simulado representativo de la red de una base aérea norteamericana, en la que se introdujo tráfico sintético tanto en lo relativo a la operación normal de la red como a los

ataques. En este sentido, aunque la utilización de un entorno controlado posibilita el etiquetado del tráfico, la naturaleza sintética del mismo introduce graves problemas en relación a la representatividad y a las proporciones entre tráfico normal y de ataque. En particular, algunas contribuciones de McHugh [McHugh, 2000] y Mahoney [Mahoney & Chan, 2003] cuestionan la exactitud del escenario en el que fue generada la base de datos identificando deficiencias que invalidarían los datos obtenidos para la evaluación de detectores de intrusiones. Por otra parte, se han identificado dos problemas adicionales en relación a su aplicación en la actualidad. En primer lugar, dada la gran evolución en los tipos y naturaleza de los ataques, los ataques considerados no resultan adecuados en la actualidad. En particular, los ataques basados en web, que constituyen uno de los vectores de ataque actualmente más utilizados, resultan escasos y claramente obsoletos. En segundo lugar, los volúmenes y tipos de tráfico utilizados, que podrían ser válidos cuando se realizó la adquisición, resultan igualmente inadecuados en el contexto de las redes actuales.

A pesar de estas limitaciones, la base de datos IDEVAL continua siendo muy referenciada en la literatura acerca de la evaluación de los sistemas de detección de intrusos.

En 2001, DARPA, en colaboración con otras instituciones, inició el programa LARIAT (*Lincoln Adaptable Real-time Assurance Test-bed*) [Rossey et al., 2002]. Desafortunadamente es de uso restringido para equipos militares de Estados Unidos de América y algunas instituciones educativas bajo condiciones especiales.

En el contexto mencionado, la mayor parte de los trabajos descritos en la bibliografía relativos al desarrollo y evaluación de IDS optan por utilizar IDEVAL, con los problemas de representatividad que ello implica, o generan sus propias bases de datos. En este caso, los datos se obtienen a partir de capturas en redes reales, generándolos artificialmente o mediante una combinación de ambas aproximaciones [Antonatos et al., 2004]. Hemos de reseñar que, en algunas investigaciones, las bases de datos utilizadas están bastante alejadas de la realidad o resultan inadecuadas para una evaluación de los detectores con el rigor científico necesario.

Una vez expuesto lo anterior, y teniendo en cuenta que los conjuntos de datos existentes resultan inadecuados, se ha decidido obtener tráfico siguiendo una metodología que se ha diseñado para este propósito. Esta metodología tiene como objetivo final obtener tráfico que pueda ser utilizado para el desarrollo y evaluación de sistemas IDS en las mejores condiciones posibles. Para ello, y atendiendo a los requisitos anteriormente mencionados, se procederá a obtener tráfico real en una red en explotación, a partir del que se obtendrá un conjunto de datos sucio. Dado que resulta enormemente complejo obtener un repertorio adecuado y suficiente de ataques a partir del tráfico real, se procederá a la generación y adquisición de tráfico de ataques, que será incorporado al conjunto de datos. Evidentemente, esta metodología altera las proporciones relativas de tráfico normal y de ataques, por lo que introducirá un sesgo en los resultados que impide una comparación en igualdad de condiciones con otros sistemas. En cualquier caso, la metodología resulta adecuada en el presente trabajo dado que su objetivo es la mejora de una técnica previa, no siendo necesaria la comparación

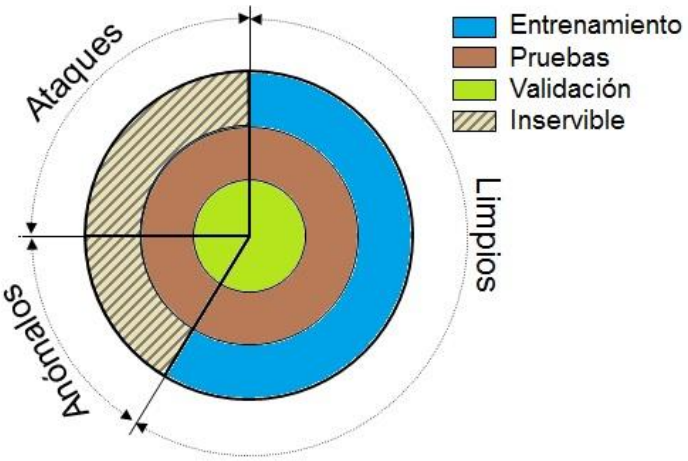


Figura 2.1: Particiones de la base de datos de tráfico según su naturaleza y uso

con otras descritas en la bibliografía. La metodología y su aplicación se describen a continuación.

2.2 Adquisición de conjuntos de datos

El cumplimiento de todos los requisitos expuestos en el apartado anterior resulta extremadamente complejo, requiriendo de una aproximación que permita obtener y clasificar el tráfico de una red en explotación. A este efecto, en el seno del grupo de investigación se ha propuesto una metodología [Bermúdez-Edo et al., 2006] para la captura y acondicionamiento de los datos que permite el entrenamiento, evaluación y validación de los sistemas de forma adecuada. Esta metodología se basa en el establecimiento de particiones en la base de datos de tráfico capturado de acuerdo a dos criterios principales: su carácter normal, anómalo o de ataque y su uso en el desarrollo del sistema, esto es, entrenamiento, test y validación. Dicha metodología resulta aplicable en el caso de combinar detección basada en firmas y en anomalías, al diferenciar entre ataques y anomalías. En la Figura 2.1 se muestran las particiones o bloques de datos resultantes.

Como se observa en la figura anterior, se establecerán un total de 9 particiones diferentes, resultantes de las combinaciones posibles entrenamiento/test/validación y limpio/ataque/anómalo. Evidentemente, según el sistema a desarrollar, algunas de las particiones pueden resultar inútiles (p.e., la partición de entrenamiento de ataques o la de entrenamiento de anomalías en la Figura 2.1), aunque es necesario definirlas para preservar la representatividad de los resultados obtenidos (problema de sesgos debidos a falta de proporcionalidad [McHugh, 2000] [Athanasiades et al., 2003]). En cualquier caso, pueden usarse técnicas de tipo *Leaving-k-out* [Duda & Hart, 1973] para aumentar la representatividad de los datos y posibilitar un mayor aprovechamiento de los mismos, al poder usarse todos los datos obtenidos.

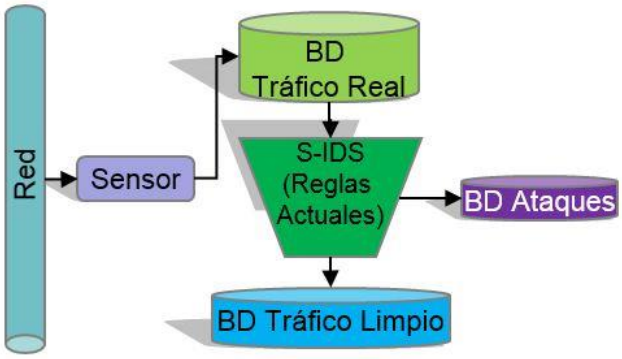


Figura 2.2: Filtrado de la traza real para separar el tráfico normal del de ataques

En el ámbito de este trabajo, dado que únicamente se realizará detección basada en anomalías, no se considerará la diferenciación anómalo/ataque, por lo que, en lo que sigue, se particularizará la metodología al caso considerado.

En cualquier caso, una vez capturado el tráfico real como primer paso de la aplicación de la metodología, se plantea un problema asociado a su categorización como normal o ataque. Evidentemente, una clasificación manual resultaría totalmente inviable, debido tanto a los elevados volúmenes de datos como al conocimiento necesario por parte del etiquetador. Por tanto, en [Bermúdez-Edo et al., 2006] se propone el uso de un IDS basado en firmas para filtrar y clasificar cada uno de los eventos de acuerdo a su naturaleza (Figura 2.2). Posteriormente será necesario establecer las particiones de entrenamiento, test y validación.

La aplicación de este método en un entorno real requiere, por tanto, de la disposición de un IDS basado en firmas que sea capaz de clasificar y separar el tráfico en dos bloques con suficiente fiabilidad, de acuerdo a las firmas de ataques. De esta forma, el detector de ataques seleccionado desempeña un papel clave, ya que es el responsable de eliminar todos los ataques del conjunto de datos sucio obtenido a fin de obtener el conjunto limpio. Sin embargo, aunque resulta imposible garantizar la inexistencia de ataques en dicho conjunto, su número debería ser extremadamente reducido si las firmas usadas por el detector están actualizadas. Por otra parte, dado que los ataques que potencialmente puedan existir serán, fundamentalmente, ataques de día cero (*0-day*) que aún no estén incluidos en las firmas, el problema se verá mitigado si la clasificación se realiza con suficiente retardo respecto del instante de la captura como para que se hayan actualizado las firmas de los posibles ataques.

Una vez determinado el método a utilizar para adquirir el tráfico limpio, nos centraremos en el tráfico de ataques. Como resultado del análisis propuesto, se obtendrá un conjunto de datos correspondientes a los ataques detectados por el IDS utilizado. Sin embargo, para obtener un conjunto representativo y suficiente de los mismos sería necesaria la captura de ingentes volúmenes de tráfico real, lo que resulta inviable en la práctica. Por tanto, en el presente trabajo se propone una aproximación diferente para

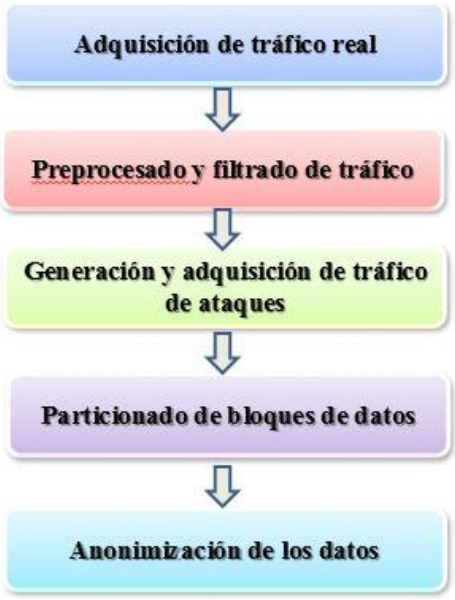


Figura 2.3: Metodología para la adquisición de tráfico

su obtención, que consiste en la generación, inyección y captura de ataques por los investigadores en un entorno controlado. De esta forma, resulta posible obtener una importante representatividad y volumen de ataques, sin más que considerar todos los descritos en la bibliografía y fuentes especializadas. Evidentemente, este procedimiento presenta el grave inconveniente de sesgar la proporcionalidad entre el volumen de ataques y de tráfico normal, dificultando la comparación de los métodos que se desarrollen con otros métodos de detección. Por otra parte, deben determinarse procedimientos que permitan generar los ataques de forma que se consiga la cobertura deseada con el menor esfuerzo posible.

Finalmente, la metodología que se utilizará para la generación de las bases de datos constará de los siguientes pasos (Figura 2.3):

- Adquisición de tráfico real, a partir de la colocación de sensores de tráfico (*sniffers*) en la/s red/es a monitorizar. El resultado de este proceso será un conjunto de datos sucio.
- Preprocesado y filtrado del tráfico obtenido mediante un IDS basado en firmas actualizado para eliminar los ataques. El resultado será una base de datos que consideraremos limpia.
- Generación y adquisición de tráfico de ataques, a partir de la recopilación de la información necesaria. Al finalizar este proceso se dispondrá de un conjunto de datos de ataque.

- Particionado. Los dos bloques de datos obtenidos, limpio y de ataques, deben ser particionados de acuerdo a su finalidad. Así, se establecerán 3 particiones para entrenamiento, evaluación y validación sobre el bloque de datos limpio, mientras que el de ataque será inicialmente dividido en evaluación y validación. Sin embargo, a fin de mejorar la significatividad y representatividad de los resultados obtenidos, se utilizará la técnica de validación cruzada (*leaving-k-out*) [Duda & Hart, 1973] para lo que se establecerán diferentes particiones que se irán utilizando consecutivamente con diferente finalidad (entrenamiento/test).
- Anonimización. Aunque no considerado en la metodología propuesta originalmente en [Bermúdez-Edo et al., 2006], se aplicará un procedimiento de anonimización de los datos obtenidos a fin de evitar los problemas de privacidad que pudiesen presentarse. Esta anonimización puede también realizarse con anterioridad al particionado.

En líneas generales, la aplicación práctica de la metodología se inicia con el despliegue de sensores (comúnmente denominados *sniffers*) que realicen la captura del tráfico de forma pasiva en entornos reales, esto es, sin interferir en las actividades propias del entorno a monitorizar. Estos sensores deben ser configurados para adquirir únicamente el tráfico que resulte de interés. En nuestro caso, únicamente se capturará el tráfico HTTP.

Una vez capturado el tráfico, la siguiente tarea sería su preprocesamiento para analizar las propiedades de los eventos capturados, realizar las operaciones de normalización o parametrización que se consideren necesarias y descartar el tráfico que no resulte de interés. El conjunto de datos obtenido será sucio, ya que podría contener ataques. Por tanto, a continuación, es necesario separar el tráfico que pudiera contener algún ataque. Para esta tarea se utiliza un sistema de detección de intrusos basado en firmas, que debe reunir algunos requisitos relacionados con su capacidad de detección y actualización. Un sistema que reúne las características necesarias (fiabilidad y disponibilidad de las reglas) es Snort [Roesch, 1998-2015].

La etapa de adquisición de tráfico sintético de ataques resulta de mayor complejidad, ya que es necesario recopilar los ataques existentes. Para ello se ha comenzado con la tarea de investigar las diversas fuentes de información sobre ataques disponibles en Internet. Durante este proceso se han localizado diversas fuentes de información de reconocido prestigio que almacenan información de las vulnerabilidades a servicios o servidores web, así como los *exploits* correspondientes. Sin embargo, el elevado número de vulnerabilidades y posibles ataques hace que sea necesario desarrollar diversos métodos que permitan su generación de la forma menos costosa posible y con las máximas garantías respecto de la cobertura de los ataques existentes. En este sentido, y a partir del análisis de las fuentes de información disponibles, se han realizado varias aproximaciones a fin de obtener baterías de ataques que puedan ser de utilidad.

Así, se ha evaluado como primera aproximación el uso de detectores de vulnerabilidades para la generación de tráfico de ataques, ya que durante el proceso de detección se genera tráfico cuyo objetivo es determinar si existe una determinada vulnerabilidad o no y, por consiguiente, será tráfico de ataque. También se ha utilizado una aproximación de *ingeniería inversa* a partir del conjunto de reglas de *Snort* asociadas al tipo de tráfico de interés (URI en nuestro caso). Utilizando estas reglas se deberían poder generar todos los ataques potencialmente detectables por dicho IDS. Finalmente, se complementaron las estrategias previas mediante la generación supervisada y/o manual, según los casos, de los ataques descritos en diversas fuentes de información. Dada la mayor complejidad de esta etapa del proceso de adquisición de datos, se dedicará el Apartado 2.3 a su descripción con mayor detalle.

Una vez se dispone de los conjuntos de datos limpios y de ataques, se procede al establecimiento de varias particiones en ambos conjuntos. Una de las particiones de cada uno de los conjuntos se reserva para validación, mientras que las restantes serán utilizadas, a partir de la aplicación de la técnica de validación cruzada, para entrenamiento y evaluación.

La última fase considerada consiste en la anonimización del tráfico adquirido en entornos reales, a fin de ofuscar la información sensible que pudieran contener. En este sentido se ha realizado una revisión de las herramientas disponibles que permiten ofuscar, cambiar o borrar información sensible dentro del tráfico de red. Sin embargo, se ha constatado que ninguna resulta de utilidad para los fines del presente trabajo, ya que eliminan la información contenida en las cargas útiles que es utilizada por los detectores a desarrollar. De esta forma, se propone una técnica de anonimización en el Apartado 2.4. Esta se basa en reemplazar la información contenida en las cargas útiles por datos neutros que no contengan información sensible, pero respetando las relaciones entre los diferentes elementos de todas las cargas útiles, con lo que se preserva la información necesaria para el desarrollo de los IDS.

A continuación se describen en detalle los procedimientos empleados para la adquisición de los diversos conjuntos de datos limpios que se utilizan en el presente trabajo, indicando las características de cada uno de ellos. Como se ha mencionado, la adquisición de los conjuntos de ataque se describirá en el Apartado 2.3.

2.2.1 Conjuntos de datos limpios

La adquisición del tráfico real necesario es una tarea nada trivial, no sólo por cuestiones técnicas, sino también debido a los datos sensibles que puedan circular por la red. Por este motivo, resulta difícil que una institución privada o pública permita la obtención de datos mediante la colocación de un sensor. Por otra parte, el acceso a la red, preferiblemente al enlace de acceso de la institución o al segmento donde se ubique el servidor web, también plantea algunas dificultades.

Afortunadamente, algunas instituciones de carácter público nos han permitido la colocación del sensor o, en otros casos, nos han proporcionado los datos. A

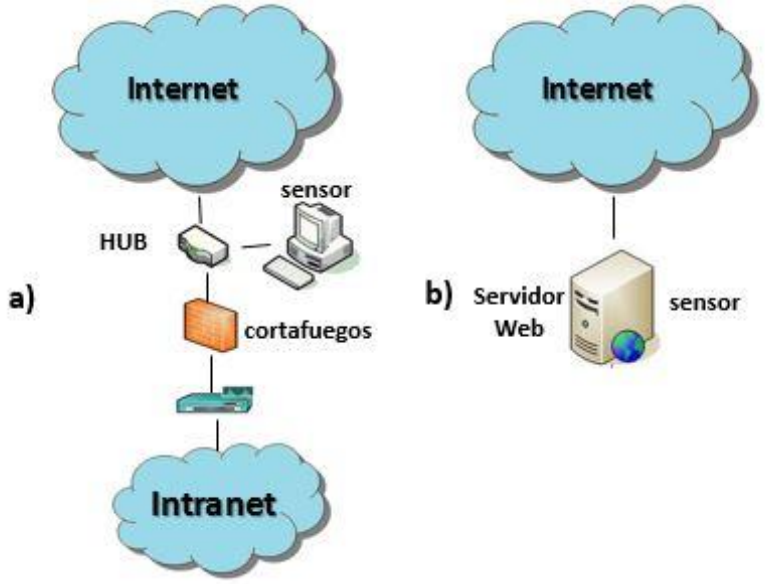


Figura 2.4: Topologías utilizadas para la colocación del sensor de tráfico: a) sensor en el segmento de acceso, b) sensor en el servidor

continuación describimos la manera en que fueron capturados los datos en las instituciones que nos permitieron esta tarea.

Captura de tráfico real

Para la obtención del tráfico es necesario el despliegue de sensores o herramientas que permitan capturar el tráfico en tiempo real en una red en explotación. La adecuada ubicación del sensor y sus capacidades (carga computacional, memoria, formato de salida, etc.) resultan relevantes de cara a evitar o minimizar la pérdida de datos durante la captura. También resulta importante la disponibilidad de filtros en los sensores que permitan capturar únicamente el tráfico de interés.

A partir de estas consideraciones, y teniendo en cuenta que el tráfico a capturar son las peticiones a los servicios HTTP, se utilizará la herramienta *tcpdump* [Van & McCanne, 1991] junto con *Snort* [Roesch, 1998-2015] y *Wireshark* [Sharpe & Warnicke, 2004] para la adquisición del tráfico y la supervisión del proceso de captura. Estos son programas del dominio público disponibles en Internet que permiten la captura y el análisis del tráfico en una red o en un segmento de red de computadoras. El tráfico es capturado en formato *pcap* [Van & McCanne, 1994], que es un formato común de amplio uso. Adicionalmente, los tres programas permiten la utilización de filtros para seleccionar el tráfico de interés para el análisis y pueden operar en tiempo real directamente sobre el tráfico circulante en la red o sobre los archivos de traza obtenidos.

En la Figura 2.4 se muestran las ubicaciones de los sensores en las diversas topologías utilizadas. La situación concreta en cada caso dependerá, obviamente, de la

infraestructura de red de la institución en la que se realice la captura, así como de la accesibilidad de la misma. A continuación se describen las diferentes instituciones colaboradoras y la manera en que fueron colocados los sensores en cada una de ellas.

Un primer escenario considerado es la red del Departamento de Teoría de la Señal, Telemática y Comunicaciones de la Universidad de Granada, en la que se colocó un sensor para la adquisición del tráfico dirigido a un servidor web en producción, utilizándose la topología mostrada en la Figura 2.4b). En este caso, el sensor utilizado fue *tcpdump* [Van & McCanne, 1991]. La base de datos así obtenida se ha denominado CERES, debido al nombre del servidor monitorizado.

El segundo escenario utilizado corresponde a la escuela preparatoria Valle Hermoso, que es una institución de enseñanza media ubicada en la ciudad del mismo nombre en México. La población estudiantil, administrativa y docente es del orden de 700 usuarios. Se ha colocado como sensor *Wireshark* [Sharpe & Warnicke, 2004], utilizando la topología en la Figura 2.4a). La base de datos capturada se ha denominado PVHDB (Preparatoria de Valle Hermoso DB).

Una vez almacenadas las bases de datos, se procederá a su preprocesamiento. A continuación se presenta la manera en que fue realizado y los aspectos considerados.

Preprocesado y filtrado del tráfico

Los datos capturados deben ser preprocesados y preparados para permitir un adecuado entrenamiento y evaluación del sistema A-IDS.

En primer lugar, deben abordarse algunas cuestiones previas relativas a la extracción y normalización de los URI. En particular, en el presente trabajo se utilizarán únicamente los URI contenidos en paquetes GET del protocolo HTTP. Por este motivo, se hace necesaria la separación de estos paquetes del conjunto capturado. Adicionalmente, es conveniente realizar algunas operaciones de normalización del contenido de las URI que, si bien no afectan su naturaleza ni la información transportada, sí podrían introducir algunas ineficiencias en el procesamiento posterior. A modo de ejemplo, el carácter ‘ ‘ (espacio) debe aparecer como ‘%20’ de acuerdo a las especificaciones del protocolo. Sin embargo, en el tráfico capturado se observa la utilización del citado carácter, siendo conveniente su sustitución.

Por otra parte, dado que, de acuerdo a la metodología propuesta, se requiere que el tráfico capturado (sucio) se encuentre libre de ataques, se hace necesaria la utilización de un detector IDS basado en firmas que permita la obtención de la base de datos limpia a partir de la sucia (Figura 2.5). Como se ha mencionado con anterioridad, el detector elegido es Snort.

Las reglas utilizadas por este IDS son completamente parametrizables, lo que permite seleccionar el conjunto de reglas aplicables en cada caso e incluso definir reglas adicionales por parte del usuario. Existen disponibles públicamente varios conjuntos de reglas, que pueden ser agrupadas dos categorías: las reglas comunitarias y las verificadas u oficiales. Estas últimas son validadas por el *Vulnerability Research Team* perteneciente al grupo de desarrollo de *Snort*. Por tanto, se denominan habitualmente

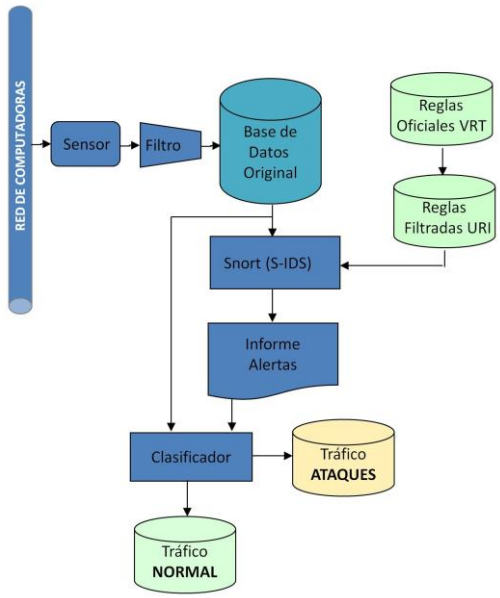


Figura 2.5: Separación del tráfico normal y ataques

VRT por sus siglas en ingles. Las reglas comunitarias, como su propio nombre indica, son reglas que han sido generadas y actualizadas por la comunidad de usuarios en Internet, pero que no han sido convenientemente evaluadas o contrastadas por expertos en seguridad. Las reglas VRT son actualizadas periódicamente, lo cual garantiza su validez y la inclusión de todos los ataques conocidos en una determinada fecha.

En el presente trabajo se han utilizado varios conjuntos de reglas VRT a lo largo de la investigación. A partir de estos conjuntos de reglas VRT, se han considerado sólo las reglas que afectan el contenido del URI [Berners-Lee et al., 2005], debido a que es la información a utilizar durante el desarrollo del sistema de detección de intrusos propuesto.

En la Tabla 2.1 se puede observar el número de reglas incluidas en cada uno de los conjuntos VRT considerados, así como el número de ellas que afectan al URI contenido en los paquetes. Como se puede apreciar en la tabla, el número de reglas ha disminuido en algunos casos. Esto se debe a que algunas de ellas han sido descartadas y otras han sido incluidas en los propios preprocesadores de Snort.

Cada regla de Snort está formada por dos partes (Figura 2.6). Por un lado se tiene la cabecera de la regla, que determina los flujos a los que debe aplicarse, así como algunos parámetros relativos a la activación. La segunda parte de la regla, llamada cuerpo de la regla, está formada por diferentes campos e incluye la información necesaria para que se active la regla, esto es, para que se produzca la detección.

Para su aplicación en el filtrado del tráfico HTTP ha sido necesario introducir algunas modificaciones en las reglas. En particular, se ha tenido que modificar el umbral de detección debido a que solo se evalúa la cabecera URI y el umbral de detección toma

Conjunto Fecha	VRT1 27-07-05	VRT3 29-03-06	VRT9 15-06-06	VRT20 4-10-06	VRT22 18-10-06
No. Reglas	3.923	4.822	5.750	9.280	7.566
Reglas URI	924	902	1.247	2.839	1.373

Conjunto Fecha	VRT43 16-08-07	VRT46 26-10-07	VRT52 23-12-08	VRT61 15-07-09	VRT69 18-09-09
No. Reglas	8.850	9.160	9.871	6.283	6.096
Reglas URI	1.416	1.448	1.620	1.641	1.639

Tabla 2.1: Conjuntos de reglas VRT usados con Snort

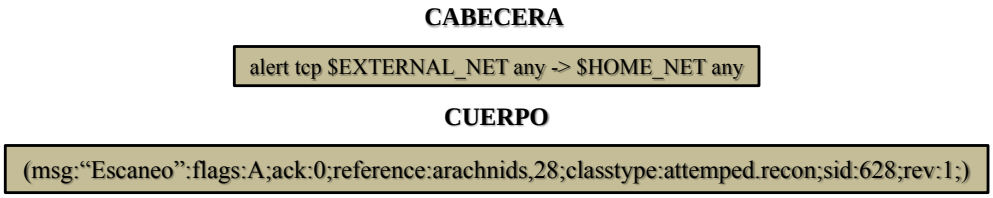


Figura 2.6: Ejemplo de regla de Snort

en cuenta los establecimientos de conexión del protocolo TCP. De manera análoga, se ha descartado el uso de los campos *flow* y *flowbits* de cada regla.

Cuando Snort es usado como S-IDS (Figura 2.5), emitirá reportes sobre los eventos detectados como intrusivos, aunque también emite alertas sobre paquetes malformados. En casos especiales, los paquetes malformados pueden representar un ataque. A su salida, con la configuración adecuada, Snort puede generar un archivo en formato *pcap* [Van & McCanne, 1991] conteniendo todos los paquetes asociados a los eventos encontrados.

A partir de este archivo y de la captura original, y mediante el uso de un programa desarrollado al efecto, se eliminan de la base de datos original (sucia) los eventos detectados como intrusivos, resultando un archivo que, de acuerdo al detector, únicamente contiene tráfico limpio de ataques y malformaciones.

Por otra parte, y en relación al preprocesamiento antes mencionado, la utilización de Snort reporta algunas ventajas ya que incluye un módulo que permite la normalización del tráfico HTTP entrante, que ha sido activado durante el procesamiento realizado.

Particionado y anonimización

Una vez obtenidos los conjuntos de datos limpios, se procede a su anonimización de acuerdo al procedimiento descrito en el Apartado 2.4. Finalmente, se realiza un particionado en tres bloques para el entrenamiento, evaluación y validación del sistema. Por tanto, como resultado del procedimiento de adquisición relativo a los

datos limpios se obtienen tres bloques diferenciados convenientemente anonimizados para cada una de las bases de datos consideradas.

2.3 Adquisición del tráfico sintético de ataques

Una vez adquirido y acondicionado el tráfico real que constituirá el conjunto de datos limpio, y continuando con la metodología descrita, a continuación se presentan algunas aproximaciones utilizadas a fin de obtener una batería de ataques. Esta permitirá generar el tráfico de ataques necesario para evaluar y validar el sistema de detección de intrusos.

El objetivo de las aproximaciones evaluadas es determinar un procedimiento que permita generar y capturar tráfico de ataques con el menor esfuerzo posible y en condiciones controladas para evitar la inclusión de otros flujos de tráfico en la captura. Como se mostrará más adelante, se ha utilizado *Snort* como herramienta de apoyo para la validación del tráfico capturado. En el Apartado 2.5 se presentarán los resultados de clasificación y evaluación de los conjuntos de ataques recopilados mediante los métodos analizados.

2.3.1 Escenario experimental

Los objetivos planteados aconsejan el establecimiento de un escenario de experimentación que reúna las características adecuadas. En particular, resulta relevante que el entorno esté controlado en todo momento y que se encuentre aislado a fin de evitar interferencias de otros sistemas. Por otra parte, se requiere de la existencia de diferentes equipos con conectividad a nivel de red y que desempeñarán el papel de atacantes o víctimas. Dado que será necesario ejecutar *exploits* para generar el tráfico de ataque, será de interés disponer de equipos operando en diferentes sistemas operativos para posibilitar su ejecución. Es importante reseñar que, cuando se trabaja con *exploits*, puede ser delicado hacer uso de ellos debido al daño que pueden causar a otros equipos o aplicaciones, pudiendo difundirse por la red a otros equipos. En este contexto, en [Massicotte et al., 2006] y [Laureano et al., 2007] se utiliza una infraestructura basada en máquinas virtuales con la finalidad de aislar el sistema experimental de entornos reales. Análogamente, para prevenir los posibles daños y aislar la red de trabajo, se propone la utilización de máquinas virtuales, en particular, del producto *VMware Workstation* [Vmware, 2015]. Mediante este software se puede simular un grupo de equipos con diferentes sistemas operativos compartiendo un mismo hardware, cada uno de los cuales opera de manera independiente y simultánea. Adicionalmente, se conseguirá el asilamiento requerido, tanto desde el punto de vista de evitar tráfico adicional como de realizar el control de posibles daños.

El escenario propuesto se compone de varios equipos, todos simulados mediante *VMware* y dispuestos dentro de un mismo segmento de red. Los equipos virtuales podrán desempeñar dos roles: víctimas o atacantes (Figura 2.7). Se ha dispuesto un único equipo víctima junto con varios equipos atacantes. Adicionalmente,

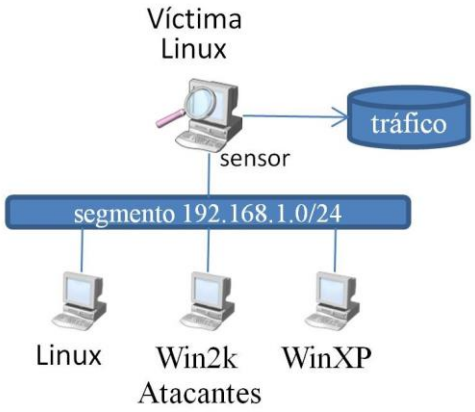


Figura 2.7: Escenario experimental con máquinas virtuales (Modo de recopilación de ataques)

al equipo víctima se le ha asignado también el papel de monitor, habilitándose como sensor para capturar todo el tráfico de red que fluye por el segmento considerado. Las máquinas virtuales con el rol de atacantes utilizan diferentes sistemas operativos para la ejecución de diferentes tipos de *exploits*. Los ataques son ejecutados lanzándolos desde la máquina o máquinas atacantes hacia el equipo víctima. Con la infraestructura formada por máquinas virtuales se aislará de cualquier posible tráfico generado por otras aplicaciones que se estén ejecutando durante la experimentación.

Este escenario será utilizado tanto para capturar los ataques como para evaluar los resultados de su ejecución. Así, en primer lugar se generarán varias baterías de ataques, mediante los programas o técnicas adecuados, que serán capturados por el sensor. En este modo, el modo de recopilación de ataques, se utiliza *Wireshark* [Wireshark, 2008-2015] como sensor para la captura del tráfico y los equipos atacantes generarán el tráfico en tiempo real. Idealmente, una batería de ataques será, por tanto, la recopilación de URI generados durante la fase de recopilación de ataques (escenario de la Figura 2.7).

Una vez dispuestas las baterías de ataques, se procederá a evaluar su comportamiento y sus características relevantes mediante la retransmisión de los paquetes contenidos en la captura realizada. En este modo, el modo de evaluación, se utilizará *Snort* [Roesch, 1998-2015] como sensor y clasificador del tráfico. Por otra parte, dado que únicamente será necesario replicar el tráfico contenido en dicha batería, sólo se considerarán dos equipos: un atacante y una víctima (Figura 2.8). Para ello, en las máquinas Linux consideradas en ambos escenarios se usa la distribución de *Linux Backtrack* [Aharoni, 2007] que cumple con todos los requisitos para este rol. Esta es una distribución que cuenta con compiladores, interpretes así como herramientas que permiten la compilación y ejecución de diferentes tipos de *exploits* desarrollados en diferentes plataformas operativas como *Linux* y *Windows*®.

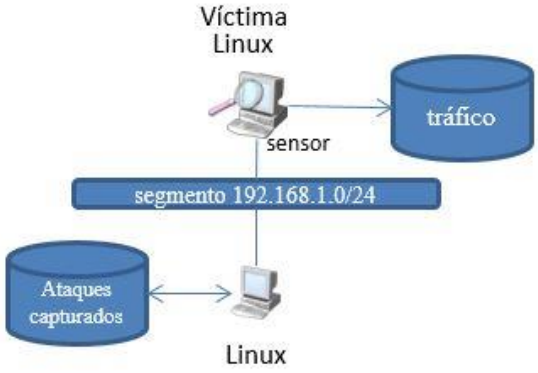


Figura 2.8: Escenario experimental con máquinas virtuales (Modo de evaluación)

2.3.2 Generación de ataques usando detectores de vulnerabilidades

En una primera aproximación se evaluará el uso de detectores de vulnerabilidades para la generación de tráfico de ataques. Estos constituyen un método para la revisión de posibles fallos o la búsqueda vulnerabilidades en una aplicación.

Esta aproximación se describe en [Wang & Abdel-Wahab, 2005] y [Yamada et al., 2005], donde se utilizan buscadores de vulnerabilidades como *Nessus* [Nessus, 2004] a fin generar tráfico de ataques para evaluar los IDS en el caso de aplicaciones o servicios web. En nuestro caso, se procede a utilizar algunos detectores de vulnerabilidades como *nikto* [Nikto, 2004], *wikto* [Wikto, 2004], *w3af* [Riacho, 2008] y *Nstealth* [N-Stalker Co., 2015], incluidos en la distribución *Backtrack*, para generar tráfico desde las máquinas virtuales con el rol de atacantes (Figura 2.7).

El tráfico generado en la red es filtrado para seleccionar únicamente los paquetes con peticiones GET y almacenado como batería de ataques. La evaluación de la base de datos obtenida mediante *Snort* muestra que se genera tráfico adicional durante la búsqueda de la vulnerabilidad, resultando, en consecuencia, en la existencia de tráfico normal entremezclado con el de ataques. Por ejemplo, de 1.181 peticiones GET con sospecha de ataques, *Snort* solo alerta de 957 [Salazar-Hernandez & Diaz-Verdejo, 2007].

Como conclusión, una vez analizada la base de datos se ha observado que en el tráfico generado por los detectores de vulnerabilidades existe tráfico del cual no tenemos la certeza de que sea tráfico de ataques, sino que es tráfico generado durante las pruebas que se realizan. Aunque los resultados de esta primera aproximación son buenos, en cuanto a la cantidad del tráfico de ataques generado y la relativa facilidad de su obtención, no son adecuados para evaluar la actuación de los sistemas de detección de intrusos. Dado que es necesaria la certeza sobre la caracterización del tráfico, esta aproximación debe ser descartada.

2.3.3 Recopilación de ataques mediante ingeniería inversa de las reglas de Snort

Como aproximación alternativa se ha considerado como origen de la información el conjunto de reglas de *Snort* asociado a las URI. Esta aproximación presenta, a priori, la ventaja de disponer del listado completo de ataques conocidos a la fecha considerada. Por otra parte, a partir del propio conjunto de reglas de *Snort*, se deberían poder generar todos los ataques potencialmente detectables por dicho IDS. Para la generación de los ataques será necesario, en función de su naturaleza, reproducir el URI, si no necesita ningún tipo de parametrización, o localizar un *exploit* que lo genere. Por tanto, esta aproximación presenta el inconveniente, frente al uso de detectores de vulnerabilidades, de requerir la recopilación manual de los *exploits* necesarios. Por el contrario, por construcción, debe presentar una mayor cobertura de los tipos de ataque conocidos.

Cada una de las reglas incluidas en *Snort* incluye un campo en el que se proporcionan referencias e identificadores de los ataques o vulnerabilidades asociados. Para la recopilación de los *exploits* se utilizan estos identificadores, a partir de los que es necesario acceder a las diversas fuentes de información consideradas en la confección del conjunto de reglas.

Se han considerado las reglas VRT del 16 de agosto del 2007, que incluyen 2.375 reglas que afectan al protocolo HTTP (caso de estudio) de un total de 8.262 reglas. En este punto es importante reseñar la diferencia entre regla y firma. La firma constituye el elemento diferenciador del ataque, es decir, la secuencia característica que determina la existencia del ataque. La detección se realiza mediante reglas que establecen los criterios que debe aplicar el detector para determinar la existencia o no de la firma y, en consecuencia, del ataque. De esta forma, una misma firma (un mismo ataque) podrá ser detectada mediante una o más reglas diferentes.

Una vez clasificadas las reglas por el campo de referencia, se generan diferentes scripts que automatizan la recopilación de información más extensa de cada uno de los *exploits* que se pueden utilizar para atacar cada vulnerabilidad.

Sin embargo, a partir del análisis de los ataques obtenidos, que se describirá en el Apartado 2.5, se constata un comportamiento inconsistente en esta aproximación, poniéndose en duda la calidad de las reglas y de los *exploits* recopilados. En particular, una inspección somera de los *exploits* muestra que muchos de ellos tienen una finalidad descriptiva de la naturaleza del ataque, no estando parametrizados ni generando instancias reales del mismo, sino plantillas que muestran la estructura de los paquetes.

2.3.4 Recopilación supervisada de ataques a partir de bases de datos de vulnerabilidades

Una vez descartada la generación de ataques mediante detectores de vulnerabilidades y puesta en duda la aproximación basada en los conjuntos de reglas de Snort, se procederá a la generación manual de los ataques. Para ello se procederá a

recopilar la información de los ataques existentes descritos en las fuentes más relevantes.

En Internet existen múltiples fuentes de información respecto de ataques y vulnerabilidades. Entre ellas, resultan relevantes *Bugtraq* de *Securityfocus* [Bugtraq, 2005] y *Open Sourced Vulnerabilities Data Base* (OSVDB) [OSVDB, 2009], en las que se han encontrado claramente documentadas las vulnerabilidades que afectan al protocolo HTTP así como referencias a los correspondientes *exploits*.

De las fuentes de recolección de ataques, la base de datos *Bugtraq* de *Securityfocus* cuenta con una descripción más explícita de la vulnerabilidad. Por otra parte, es una fuente que contiene de una manera organizada, fiable y clara los datos relacionados con las vulnerabilidades en general. Por estas razones y, teniendo en cuenta además que en las aproximaciones anteriores ya se ha recopilado parte de la información de esta fuente, así como los *exploits* asociados, se ha decidido tomar esta fuente de información como punto de partida para la realización de esta etapa.

Una de las ventajas de usar las fuentes de información de las vulnerabilidades web antes descritas es que se mantienen actualizadas con las últimas vulnerabilidades encontradas. La mayoría de estas se encuentran documentadas, lo que permite la reproducción del *exploit* a fin de obtener una batería de ataques web. Otra de las ventajas es que la batería que se recopile con estas fuentes será más realista y contendrá los ataques actualizados.

En relación a la disponibilidad de los *exploit*, se han usado diversas fuentes adicionales a las ya mencionadas. Una de ellas es *Milw0rm* [milw0rm, 1998], que cuenta con una gran variedad de tipos de *exploit*, aunque desafortunadamente no consideran ninguna clasificación de los mismos. La diversidad de *exploits* incluidos, que van desde una simple URL hasta complejos programas desarrollados en lenguajes de alto nivel, la hacen una fuente de información a considerar. Asimismo, *Packetstorm* [Packetstorm, 2002] es otra fuente de *exploits* que complementa a las anteriores.

En cualquier caso, respecto de los objetivos del presente trabajo, se han encontrado los problemas que a continuación se detallan:

a) Parametrización del ataque

Se han encontrado casos en los que el *exploit* correspondiente a la vulnerabilidad no presenta una adecuada parametrización. Así, a modo de ejemplo, el *exploit* (consistente en el URI que genera el ataque)

http://ejemplo.com/showphoto.php?photo=[query]

incluye el parámetro *query*, que representa las posibles variaciones del ataque, pero no se incluye ningún valor concreto que lo implemente.

Sin embargo, en otros casos se encuentra claramente el URI que explota una vulnerabilidad, como en el siguiente ejemplo:

http://ejemplo.com/cgi-bin/hello.bat?&dir+c:

donde podemos apreciar que el uso directo del URI constituye un ataque.

Este problema ya había sido observado en las aproximaciones previamente realizadas y había sido el causante, en parte, de los malos

resultados obtenidos. En este caso se va a proceder a la revisión manual de los mismos para su adecuación.

b) Búsqueda de los *exploits*

En otros casos no se indica el URI ni su formato, siendo necesario localizar el *exploit* adecuado a partir de las referencias proporcionadas. Para ello se ha realizado una búsqueda de los *exploits* en las diversas fuentes disponibles siguiendo el diagrama de flujo representado en la Figura 2.9. Obviamente, se ha tomado como punto de partida la información contenida en *Bugtraq*. Siguiendo el flujo mostrado, si el *exploit* es encontrado se parametriza, adecuándolo según el lenguaje de programación en el que se encuentre. Una vez realizada esta tarea, y haciendo uso del escenario descrito previamente (Figura 2.7), se ejecuta el *exploit*, capturándose y almacenándose las trazas correspondientes. Cuando el *exploit* no sea encontrado se continúa la búsqueda en las fuentes de información en la secuencia mostrada en la Figura 2.9. Es importante notar que todas fuentes de información consultadas están disponibles en Internet. Como último recurso, la búsqueda del ataque se realizó en foros de discusión, listas de correos o mediante una búsqueda general en Internet.

Finalmente, para terminar con la preparación de las bases de datos, tanto limpias como de ataques, es necesario considerar la privacidad. Así, cuando se proporcionan bases de datos con tráfico real, estas pueden contener datos sensibles. Que pueden comprometer la seguridad de la aplicación e inclusive el servidor donde está instalada la aplicación. A fin de mitigar el problema, y formando parte de la metodología, se ha desarrollado un procedimiento en el que se aplica un método a fin de anonimizar los datos sensibles. A continuación se presenta la aproximación propuesta para la anonimización del tráfico real.

2.4 Anonimización de tráfico de red

Como se ha comentado previamente, uno de los principales problemas que se plantean cuando se manejan bases de datos con tráfico real es que pueden contener datos sensibles en cuanto a la seguridad y privacidad de las comunicaciones. En este sentido, si estos datos sensibles son manejados sin una autorización expresa de los implicados, se puede incurrir en acciones tipificadas como delito de acuerdo a las legislaciones vigentes en cada país. En particular, la Ley de Protección de Datos vigente en la Unión Europea tipifica este tipo de actividades. Por otra parte, además de las cuestiones relacionadas con la privacidad de los usuarios, también se plantean problemas desde el punto de vista de la seguridad de la infraestructura de la red, ya que permiten inferir información que puede ser utilizada para atacar dicha infraestructura [Biskup & Flegel, 2009] [Yurcik et al., 2007].

Para solventar estos problemas se suele plantear una aproximación basada en la anonimización de la información sensible contenida en la base de datos de tráfico. De

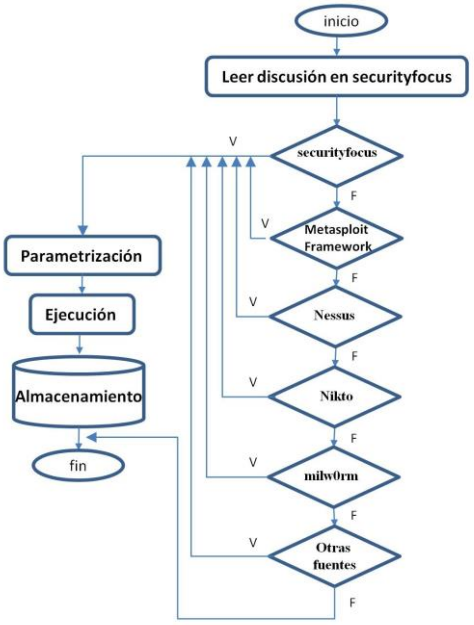


Figura 2.9: Diagrama de flujo de búsqueda de *exploits*

esta forma, se posibilita, además, el uso de las mismas bases de datos por diferentes equipos investigadores, lo que permite comparar los rendimientos de las técnicas desarrolladas. Evidentemente, estas técnicas deben preservar la información relevante, de cara a la aplicación perseguida, sin que sea posible inferir ningún tipo de información sensible a partir de la misma.

Sin embargo, las técnicas descritas en la bibliografía, y que serán comentadas brevemente en los apartados siguientes, se centran en la anonimización de la información relativa a las cabeceras, como pueden ser las direcciones IP, los números de puertos, etc. Esto es debido a que, en el escenario habitual, las cargas útiles (*payloads*) de los paquetes son directamente eliminados o sólo se incluyen los primeros bytes. Consecuentemente, estas técnicas no serán de aplicación cuando la información sensible se encuentre en dichas cargas útiles, como es el caso que nos ocupa.

En consecuencia, a continuación se presenta una técnica para la anonimización de la información contenida en las cargas útiles asociadas a protocolos basados en el intercambio de mensajes de texto, como es el caso de HTTP o DNS. En particular, la técnica propuesta se aplica al caso de peticiones GET del protocolo HTTP, dado que es el escenario considerado. En esencia, el proceso de anonimización se basa en el mapeo y sustitución, una a una, de las diferentes cadenas de texto que aparezcan en la carga útil, a la vez que se obtienen algunos datos relativos a propiedades genéricas de las cadenas sustituidas y que pueden ser relevantes en el proceso de detección, como puede ser la frecuencia de aparición de la cadena. Esta técnica presenta dos características importantes: es de fácil implementación y bajo coste computacional y, adicionalmente,

preserva la información en la que se basan las técnicas de detección asociadas a la identificación de cadenas en las cargas útiles.

2.4.1 Técnicas de anonimización del tráfico de red

Como se ha mencionado con anterioridad, la anonimización del tráfico de red tiene como objetivo fundamental preservar la seguridad y la privacidad de las comunicaciones en lo que respecta a los datos sensibles implicados. Estos datos sensibles pueden corresponder tanto a datos relativos a los individuos implicados (por ejemplo el DNI, los números de cuenta, las páginas web accedidas, los perfiles y preferencias, etc.), como a datos relativos a la infraestructura de red (direcciones y máscaras de red, ubicación de cortafuegos, etc.). De esta forma, es necesario ocultar cualquier tipo de dato que permita inferir información de dicha naturaleza, pero sin alterar la información asociada al proceso que se realizará seguidamente. Es decir, se trata de ocultar la información sensible sin alterar las características que serán analizadas en las fases de tratamiento posterior. Por ello, en la aplicación de las técnicas de anonimización resulta extremadamente relevante la finalidad a la que se destinan los datos. Así, por ejemplo, si se pretende caracterizar los flujos de tráfico entre diferentes subredes será necesario que la transformación a aplicar a los datos (IP en este caso) no altere las relaciones entre los diferentes equipos en cuanto a su pertenencia a la misma subred.

Las técnicas habituales de anonimización se pueden agrupar en las siguientes categorías [Yurcik et al., 2007]:

- **Filtrado**: el valor de los campos con información sensible es borrado.
- **Reemplazo**: se cambian los valores de los campos sensibles bien mediante la permutación con otros valores contenidos en los datos (pseudo-anonimato) o por valores diferentes (anonimato total).
- **Reducción de exactitud**: se sustituyen los valores de los datos por una aproximación de los mismos o se realiza un mapeo de los valores de los datos por grupos.
- **Agregación de ruido**: se agrega ruido para perturbar los valores de los campos.
- **Agregación**: se sustituye el valor de los campos con estadísticas acumulativas de dichos valores.

Estas técnicas proveen diferentes niveles de protección, dependiendo la necesidad de utilizar unas u otras de las políticas de privacidad que cada entidad tenga. Por otra parte, en función de la finalidad de los datos, será necesario aplicar transformaciones con propiedades diferentes.

En el presente trabajo se pretende ofuscar la información contenida en las cargas útiles de los paquetes asociados a los protocolos de aplicación a analizar sin perder la información referente a la estructura y composición de los mismos. En particular, se

pretende ocultar la información relativa a las páginas accedidas, los nombres de las variables y sus valores en el caso de URI contenidos en peticiones GET. La información relativa a direcciones IP, puertos y restantes cabeceras resulta irrelevante y, por tanto, pueden ser anonimizadas mediante borrado o reemplazo.

Herramientas de anonimización

Existen varias herramientas disponibles para realizar la anonimización, entre las que cabe mencionar *SCRUB-tcpdump* [Yurcik et al., 2008], *TCPdpriv* [Minshall, 1996], *Anonymizer API* [Koukis et al., 2006] y Anonym [Farah & Trajkovic, 2013].

SCRUB-tcpdump permite la anonimización de las trazas de red en formato *tcpdump*, en campos que puedan contener datos sensibles. La herramienta trabaja en diferentes formas eliminando la información, añadiendo ruido, reemplazando o realizando permutaciones de los datos en los campos seleccionados. Sin embargo, trabaja únicamente hasta el nivel de transporte, por lo que la única operación posible para ofuscar las cargas útiles transportadas en los paquetes TCP o UDP consiste en el borrado de las mismas. En la Figura 2.10 se muestra un ejemplo del resultado obtenido mediante la aplicación de esta herramienta, resultando en la anonimización de la dirección IP.

TCPdpriv es una de las herramientas disponibles en Internet con mayor difusión. Su operación es análoga a *SCRUB-tcpdump*, eliminando información sensible de las cabeceras de las trazas de red. En capas inferiores a TCP, dispone de una amplia gama de posibilidades de parametrización. De igual manera que ocurría con *SCRUB-tcpdump*, permite borrar toda la carga útil de los protocolos TCP y UDP. Provee diferentes niveles de anonimización, desde dejar los campos sin cambios hasta conseguir el más estricto anonimato, como el cambio completo de rangos de direcciones IP.

Anonymizer API (AAPI) es una herramienta desarrollada en C por Koukis y otros autores, presentando como principal ventaja su rapidez de procesamiento de las trazas. Utiliza funciones diversas dependiendo del nivel de anonimato a aplicar a las trazas. AAPI provee una variedad de funciones de anonimización como el uso de compendios (*hashing*) con diferentes algoritmos (MD5, SHA, CRC32, etc.), aleatorización (*random*) para campos genéricos, mapeo de nombres de archivos y URI (*mapping*) para valores secuenciales sobre algún tipo de distribución (uniforme, gausiana, etc.), reemplazo (*replacing*) con constantes enteras o cadenas de caracteres, preservación (*prefix-preserving*) para direcciones IP, sustitución de expresiones regulares y borrado de campos. Esta herramienta permite anonimizar parte de las cabeceras en los protocolos de la capa de aplicación como son HTTP y FTP, si bien esta se realiza reemplazando el contenido de la cabecera por la que se indique.

Las tres herramientas mencionadas han sido analizadas y evaluadas con la finalidad de determinar su aplicabilidad en el contexto considerado. Sin embargo, únicamente AAPI permite operaciones diferentes del borrado sobre los campos de la

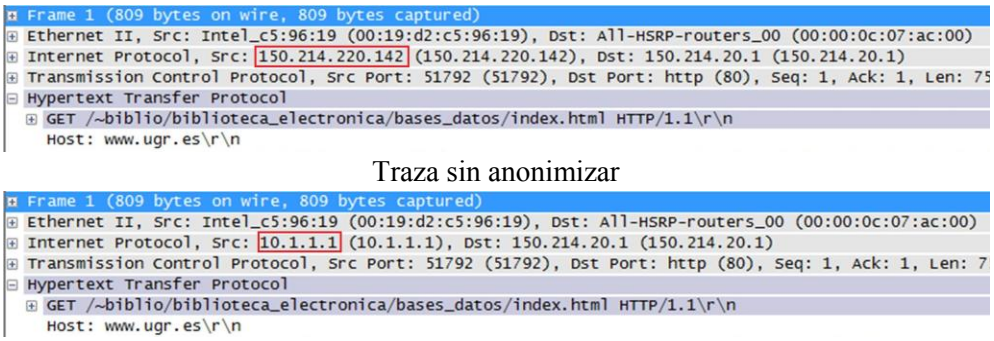


Figura 2.10: Anonimización de traza mediante SCRUB-tcpdump

capa de aplicación. Además, la única operación disponible es la sustitución por cadenas seleccionadas, lo que resulta inadecuado.

En consecuencia, se ha desarrollado una metodología que permita el anonimato de los campos con datos sensibles de las cabeceras del protocolo HTTP que será descrita a continuación. La metodología propuesta es aplicable a otros protocolos basados en el paso de mensajes de texto.

2.4.2 Antecedentes y especificación de requisitos

La técnica de anonimización a utilizar debe reunir un conjunto de requisitos relacionados con las técnicas de detección a utilizar y la naturaleza de los mensajes. A fin de determinarlos, consideraremos la estructura típica de los mensajes intercambiados mediante un protocolo basado en el paso de mensajes, particularizándolo al caso de URI del protocolo HTTP, así como la técnica de detección desarrollada, denominada SSM [Estevez-Tapiador et al., 2005]. Esta técnica, que será detallada en el Capítulo 3, se basa en el análisis de los elementos que constituyen cada mensaje (cada URI en el caso concreto considerado) tanto en cuanto a la secuencialidad de los mismos como a los diferentes valores que pueden adoptar. De esta forma, cada mensaje se descompone en un conjunto de valores o cadenas que pueden aparecer en cada uno de los campos del mensaje, definiéndose un vocabulario asociado a cada campo, esto es, un conjunto de valores posibles para las cadenas en dicho campo.

En este contexto, la técnica de anonimización a aplicar debe preservar la información asociada a los diferentes valores (cadenas) que pueden aparecer en los campos de los mensajes. Esta información presenta dos facetas diferenciadas desde el punto de vista del detector. Por una parte, debe preservarse la información relativa a la posibilidad de aparición de una cadena concreta (pertenencia al vocabulario) en cada uno de los campos y, por otra parte, resulta relevante la frecuencia de aparición de cada uno de estos valores a fin de determinar su probabilidad. Adicionalmente, y en función de las variantes de la técnica de detección considerada, puede resultar de interés disponer de información relativa a la naturaleza de las cadenas (numérica, alfanumérica,

alfabética) y algunas de sus características más relevantes, como por ejemplo la longitud en número de caracteres de dichos valores.

La información sensible se encontrará, obviamente, en los valores de los parámetros, como por ejemplo, el valor del campo DNI de una consulta, el nombre de un usuario o la página accedida. Por tanto, la ofuscación necesaria debe eliminar los valores de las cadenas, no siendo válidas, en consecuencia, las técnicas de reemplazo basadas en permutaciones. La única técnica viable en este contexto será la de reemplazo en base a un conjunto de valores diferente del conjunto original.

Finalmente, otro de los requisitos de la técnica de anonimización a emplear es que debe permitir la reposición de los valores originales. A pesar de que esto pueda parecer contradictorio, hemos de reseñar que el objetivo de la anonimización es disponer de trazas de tráfico susceptibles de ser utilizadas durante el desarrollo y evaluación de los detectores. Es evidente que la puesta en explotación de los mismos requerirá que la transformación realizada pueda ser invertida en los modelos resultantes a fin de que puedan operar en la red real. Si se realiza adecuadamente, esta reposición puede ser realizada por el administrador de la red, de forma análoga a la anonimización inicial, por lo que los investigadores o desarrolladores no tendrán acceso en ningún momento a la información sensible. Esta restricción puede relajarse sin más que considerar la aplicación de la transformación a todo el tráfico entrante cuando el detector se encuentre en operación, aunque de esta forma se introduciría una sobrecarga computacional innecesaria.

2.4.3 Metodología propuesta

La metodología propuesta se basa en reemplazar cada uno de los valores de cada uno de los campos que constituyen el mensaje a partir de un diccionario de equivalencias. Para ello se considera como punto de partida la traza de red capturada en bruto, es decir, tal como se obtiene a partir de la red en explotación. A continuación detallaremos el proceso y metodología a seguir que consta de varias fases (Figura 2.11):

- **Preparación y acondicionamiento**: durante esta fase se extrae la información relevante de cada uno de los paquetes de la traza de red. También se consideran procedimientos adicionales relacionados con la calidad de los datos obtenidos.
- **Obtención de vocabulario**: se obtiene el vocabulario asociado al tráfico monitorizado.
- **Caracterización del vocabulario**: opcionalmente, si el método de detección así lo requiere, se caracterizan cada uno de los elementos del vocabulario.
- **Generación del nuevo vocabulario**: se genera un nuevo vocabulario con el mismo número de elementos que el original para reemplazarlo. No debe existir ningún tipo de relación entre los elementos del nuevo vocabulario y los del original.

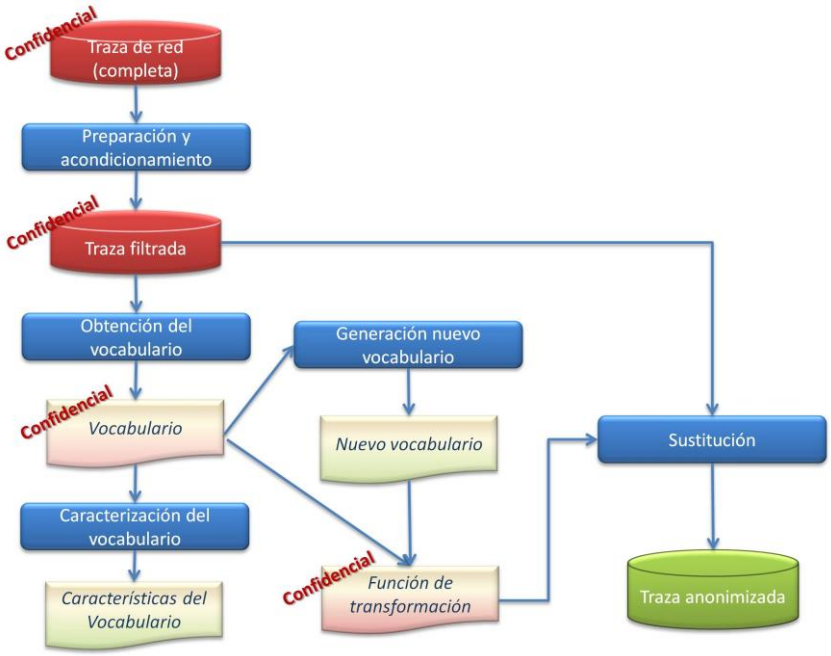


Figura 2.11: Fases y elementos del proceso de anonimización propuesto

- **Sustitución**: se genera una función de transformación que asocie a cada elemento del vocabulario original un elemento del nuevo vocabulario. Usando esta función, se reemplazan todas las cadenas correspondientes al vocabulario original en la traza de red por su correspondiente elemento del nuevo vocabulario.

Como resultado de estas fases, que serán detalladas a continuación, se obtiene un conjunto de datos que preserva la información útil de la traza de red y que se encuentra convenientemente anonimizada. La información confidencial se encuentra en el vocabulario original y en la función de transformación, por lo que para preservar la privacidad únicamente se requiere la ocultación de esta información. Adicionalmente, es posible revertir la operación sin más que aplicar la función de transformación inversa.

Preparación y acondicionamiento

La metodología propuesta se inicia sobre el conjunto de datos capturados de la red tras eliminar los ataques, es decir, sobre la traza de tráfico limpia,

$$T_{orig}=\{t_{orig,i},\ 1<i<L_{orig}\},$$

que debe haber sido adquirida en función de las características a analizar. Se considera, por tanto, el uso de los filtros que se estimen oportunos sobre los datos durante el proceso de captura. Evidentemente, esta traza contendrá la información sensible que se pretende anonimizar, por lo que deberá haber sido adquirida y procesada por personal

autorizado. Adicionalmente, es posible que incluya cabeceras con información relativa a direcciones IP, puertos, etc. Esta información también puede resultar sensible, por lo que habrá que considerar la aplicación de técnicas de anonimización también a estos elementos. Dado que únicamente estamos analizando la ofuscación de la información contenida en las cargas útiles, este aspecto no será considerado. Por tanto, en lo que sigue consideraremos que dicha información de cabecera ha sido convenientemente anonimizada mediante la técnica que se estime oportuna y de acuerdo a la política de privacidad que sea de aplicación.

Otro aspecto a considerar en esta fase es la extracción de la información que se considera relevante para las fases posteriores, lo que puede requerir un filtrado adicional de la traza T_{orig}, En nuestro caso particular, únicamente resulta de interés el URI contenido en las peticiones GET del protocolo HTTP, por lo que se descartarán los paquetes que no incluyan peticiones GET y se extraerá dicho URI.

Como resultado del proceso de preparación y acondicionamiento se obtendrá, a partir de la traza original, T_{orig}, una nueva traza,

$$T_{fil}=\{t_i,\ 1<i<L_{fil}\};T_{fil}\subseteq T_{orig}$$

únicamente con los datos a considerar en las fases posteriores. Típicamente, estos serán cargas útiles, o segmentos de la misma, de los protocolos considerados. Evidentemente, la información contenida en esta traza continuará siendo sensible, por lo que todo el proceso anterior debe haber sido realizado por personal autorizado al que se habrán suministrado las especificaciones de captura y los filtros a aplicar, en su caso.

Obtención del vocabulario

La siguiente fase consiste en la obtención del vocabulario original, V_{orig}, a partir de T_{fil}. Para ello se requiere la segmentación de las cargas útiles contenidas en T_{fil} de acuerdo a las especificaciones correspondientes.

Sean *nseg(p)* y *segmento(p,i)* las funciones que obtienen, respectivamente, el número de segmentos de la carga útil *p* y el segmento *i*-ésimo de dicha carga útil. A partir de éstas, la obtención del vocabulario se puede realizar de acuerdo al siguiente procedimiento:

- Inicialización:

$$V_{orig} = \varnothing$$

- Recursión: $\forall\ t_i \in T_{fil}$

$$n_i = nseg(t_i)$$
$$S_i = \left\{ s_j = segmento(t_i, j),\ \forall j, 1 \le j \le n_i \right\}$$
$$V_{orig} = V_{orig} \bigcup S_i$$

- $T_{\{orig \mid fil \mid anon\}}$: Traza de tráfico
  (Conjunto de cargas útiles)
  Original, filtrada o anonimizada.
- t_i: elemento de una traza de tráfico
- $L_{\{orig \mid fil \mid anon\}}$: tamaño de una traza (núm. elementos)
- $V_{\{orig \mid nuevo\}}$: vocabulario (original o nuevo)
- M: tamaño del vocabulario
- o_k: elemento del vocabulario original (palabras)
- p_k: elemento del vocabulario anonimizado (pseudo-palabras)
- S: conjunto de segmentos de un elemento de la traza
- s_i: segmento i-ésimo de un elemento de la traza
- C: conjunto de vectores de características del vocabulario
- c: vector de características de un elemento del vocabulario

Tabla 2.2: Notación utilizada para el método de anonimización

Finalmente, se obtendrá el vocabulario compuesto por todos los valores diferentes de las cadenas en cada uno de los segmentos

$$V_{orig}=\{o_k,\ 1\leq k\leq M\}$$

siendo $M\geq L_{fil}$. Este vocabulario contiene todas las cadenas de texto encontradas en la traza, por lo que contiene información sensible. Para su obtención será necesario suministrar al personal autorizado los programas correspondientes incluyendo las funciones *nseg()* y *segmento()*.

Por motivos meramente operativos puede resultar conveniente ordenar alfabéticamente los elementos del vocabulario, a fin de facilitar el proceso de búsqueda y sustitución. Esta ordenación no introduce ninguna modificación conceptual en el proceso de anonimización.

Caracterización del vocabulario

En algunos casos la técnica de detección puede incorporar información relativa a la propia estructura de las cadenas de texto que pueden aparecer en los diferentes elementos constitutivos de la carga útil. A modo de ejemplo, en el caso de los URI puede utilizarse la naturaleza de la cadena (texto, alfanumérica, numérica) y su longitud. Dado que se va a realizar una sustitución de dichas cadenas, será necesario preservar la información de estructura que se considere necesaria, siempre con las limitaciones que determine la política de privacidad en uso. Es decir, se podrían considerar características globales como las mencionadas, mientras que sería delicado incluir aspectos de mayor detalle como por ejemplo, el número de vocales que contiene.

En cualquier caso, para esta fase se considera el vocabulario previamente extraído, V_{orig}, para el que se obtiene el vector de características, C_k, de dimensión D,

asociado a cada elemento del vocabulario, o_k, a partir de la función que obtiene la característica i-ésima, como

$$C_k = [c_{ki}]_{1 \le i \le D} \quad \text{siendo } c_{ki} = \text{característica}(o_k, i)$$

La información resultante carece de información sensible si las características extraídas son adecuadas, por lo que puede ser utilizada por personal no autorizado. Sin embargo, durante el proceso de extracción habrá que manejar el vocabulario original, por lo que su obtención deberá realizarse por parte de personal autorizado al que se suministrarán los programas necesarios.

Generación del nuevo vocabulario

El nuevo vocabulario será utilizado para sustituir al original, por lo que debe contener el mismo número de elementos (M). Dado que no debe existir ninguna relación entre cada uno de los elementos de ambos vocabularios, sin pérdida de generalidad, se propone la generación de elementos del vocabulario de acuerdo al esquema pn , con n un número en el rango $[1,M]$. Dado que no serán palabras válidas en el protocolo considerado, y a fin de clarificar la nomenclatura, denominaremos en lo que sigue a estos elementos como pseudo-palabras.

Por tanto, el nuevo vocabulario, V_{nuevo}, estará compuesto por

$$V_{nuevo}=\{p_k=\text{“pk”}, 1 \le k \le M\}$$

Evidentemente, no existe ningún tipo de información sensible en este vocabulario.

Sustitución

La sustitución constituye el núcleo de la técnica de anonimización que, como se ha comentado, consiste en el reemplazo de cada una de las apariciones de las palabras en el vocabulario por pseudo-palabras. Para ello se define una función de reemplazo,

$$\Re : V_{orig} \rightarrow V_{nuevo}$$
$$\Re(o_k) = p_k$$

La función de reemplazo debe ser biyectiva para no alterar las frecuencias de aparición de cada palabra ni sus posiciones relativas dentro de las cargas útiles. La obtención de la traza anonimizada, T_{anon}, se realizará mediante la aplicación de la función de reemplazo a cada uno de los segmentos de cada uno de los elementos de la traza filtrada, T_{fil}, de tal forma que

$$segmento(t_{anon,i}, j) = \Re(segmento(t_{fil,i}, j)) \quad 1 \le j \le nseg(t_{fil,i})$$

siendo

$$T_{anon}=\{t_{anon,i}, 1 \le i \le L_{fil}\}; card(T_{fil})=card(T_{anon})$$

L	Host	URI
123	http://www.ugr.es	~biblio/bibl_electronica/bases_datos/index.html

Figura 2.12: Formato de los registros de la traza filtrada

En el caso de que se hubiesen ordenado alfabéticamente los elementos de V_{orig} resulta conveniente realizar una reordenación aleatoria de los elementos de V_{nuevo} o asignar el número de secuencia incluido en la pseudo-palabra al azar a fin de ocultar la información de orden en el resultado. En caso contrario, de la relación $p_k<p_{k+1}$ podría extraerse información, especialmente en el caso de grandes vocabularios. Consecuentemente, habrá que aplicar la misma reordenación a las características C_k.

Dado que se utilizan tanto T_{fil} como V_{orig}, ambos conteniendo información sensible, el proceso de reemplazo debe ser realizado por personal autorizado. El resultado, T_{anon}, carece de información sensible.

Finalmente, tanto la traza anonimizada, T_{anon}, como los vectores de características, C_k, pueden ser distribuidos sin restricciones.

Restitución

La aplicación de la técnica de detección desarrollada, en su caso, puede realizarse sin más que considerar la función de reemplazo en todos los paquetes a analizar de forma análoga a la realizada durante el proceso de anonimización. Sin embargo, esto supone una carga computacional que resulta conveniente eliminar. Dado que la función de reemplazo es biyectiva, existirá la función inversa que puede ser aplicada al modelo obtenido, si este incluye información sobre las pseudo-palabras, recuperándose el vocabulario original en dicho modelo.

Evidentemente, también puede recuperarse la traza filtrada original a partir de la traza anonimizada y de la función de reemplazo inversa, si fuese necesario.

2.4.4 Ejemplo de aplicación

A modo de ejemplo, se describe la aplicación de la metodología a la base de datos denominada PVHDB.

Preparación y acondicionamiento

La traza original consta de 1.176.765 paquetes de tráfico GET “limpio”. La anonimización de las cabeceras se ha realizado mediante borrado, ya que la información a utilizar se encuentra únicamente en las cargas útiles. Mediante los programas adecuados, se ha extraído el URI de cada uno de los paquetes, que ha sido almacenado en un archivo de traza filtrada, T_{fil}, en el que también se incluye la longitud del URI a fin de facilitar el procesamiento de los datos. El formato resultante de cada uno de los registros del archivo de traza filtrada se muestra en la Figura 2.12.

Obtención del vocabulario

A partir del archivo de traza se ha extraído el vocabulario tras la segmentación de cada uno de los URI. Para ello se ha utilizado la librería *URIparser* [Pipping, 2007], que responde a las especificaciones del RFC2396.

El tamaño del vocabulario resultante, V_{orig}, es de 28.025 palabras diferentes.

Caracterización del vocabulario

Dicho vocabulario ha sido analizado para obtener el tipo de cadena correspondiente a cada una de las palabras así como la longitud de las mismas.

En cuanto a la clasificación, se han etiquetado 30 palabras como alfabéticas, 481 palabras como numéricas y 27.014 palabras como alfanuméricas. En la Figura 2.13 se muestra el histograma correspondiente a la distribución de las longitudes de las palabras analizadas.

Generación del nuevo vocabulario

De acuerdo al tamaño del vocabulario original, se ha generado un nuevo vocabulario compuesto por 28.025 pseudo-palabras con el formato *p[n]*, siendo *n* un número de 0 a 28.024.

Sustitución

Para realizar la sustitución de las cadenas en la traza filtrada se ha definido una función de reemplazo (Figura 2.14) que asocia cada palabra del vocabulario original, V_{orig}, con una pseudo-palabra de V_{nuevo}.

Obsérvese que, a fin de facilitar la búsqueda de las palabras a reemplazar durante la sustitución, se han ordenado alfabéticamente las palabras del vocabulario original. En consecuencia, también se ha realizado la reordenación correspondiente en el conjunto de características. Por ello, se ha considerado la generación de un archivo de características indexado en el que se detallan las características de cada una de las pseudo-palabras (Figura 2.15).

Finalmente, se realiza la sustitución de todas las palabras contenidas en los URI de la traza filtrada (Figura 2.16). En este proceso se respetan las posiciones relativas de las cadenas en el URI, por lo que la información de estructura (sintaxis) se mantiene. Por otra parte, dado que la sustitución es uno a uno, se preserva el número de veces que aparece cada cadena, por lo que no se introducirán cambios en las frecuencias relativas de aparición. Se consiguen cumplir, por tanto, los requisitos asociados a la técnica de detección. La traza resultante, así como la relación de características de los elementos del vocabulario, carecen de información sensible.

Restitución

Una vez estimado el modelo a utilizar para la detección de anomalías se puede revertir la anonimización sin más que usar la transformación inversa sobre el modelo resultante, si este incluye la información relativa al vocabulario.

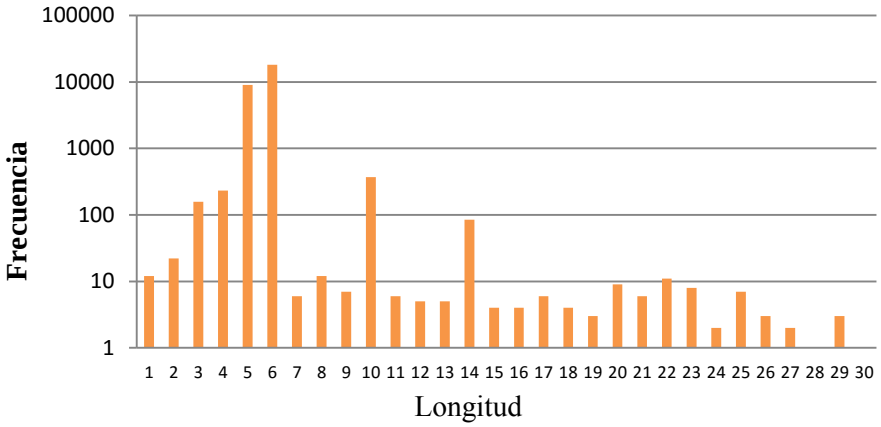


Figura 2.13: Distribución de longitudes de las palabras en el vocabulario

Pseudopalabra	Palabra
p0	bases_datos
p1	bibl_electronica
p2	index.html
p3	www.ugr.es
p4	~biblio

Figura 2.14: Función de reemplazo: asociación entre palabras y pseudo-palabras de los vocabularios

Pseudopalabra	Longitud y tipo	
p0	11	A
p1	16	X
p2	10	A
p3	10	A
p4	6	A

Figura 2.15: Índice de características

En nuestro caso particular, el modelado considerado se basa en el modelado de Markov, en el cual se incluye un vocabulario con las cadenas que pueden ser observadas en el proceso a modelar. Evidentemente, durante la experimentación, este vocabulario estará asociado con las pseudo-palabras del nuevo vocabulario. Sin embargo, una vez obtenido el modelo y estimadas las probabilidades de las pseudo-palabras, se puede obtener el modelo correspondiente a la traza original por parte del personal autorizado sin más que aplicar la función de reemplazo en el sentido inverso (Figura 2.14) al vocabulario asociado al modelo. Evidentemente, tras esta restitución, el modelado

L	Host	URI
123	http://www.ugr.es/	~biblio/bibl_electrónica/bases_datos/index.html
21	http://p3/	p4/p1/p0/p3

Figura 2.16: URI antes y después de la anonimización

Base Datos	DARPA '99			CERES	PVHDB
	HUME	MARX	Ataques		
Tráfico total	32.435	41.648	1455	437.957	1.716.781
Tráfico http	12.154	16.539	1455	143.826	1.360.515
Peticiones GET	**12.154**	**16.539**	**1455**	**143.826**	**1.360.515**

Tabla 2.3: Paquetes (en bruto) contenidos en los conjuntos de datos adquiridos

resultante será idéntico al que se hubiese obtenido sin realizar el proceso de anonimización.

2.5 Conjuntos de datos para experimentación

A continuación se describirán las características más relevantes de los conjuntos de datos que serán utilizados durante la experimentación a realizar, tanto para los conjuntos limpios como los de ataque.

2.5.1 Conjuntos de datos limpios

Como se ha indicado en el Apartado 2.3, se han considerado 2 escenarios para su utilización y/o adquisición de los conjuntos de datos. Adicionalmente, se ha incluido también el conjunto de datos utilizado en los trabajos precursores de esta tesis, en los que se hacía uso de la base de datos DARPA'99 junto con un conjunto de ataques sintético.

El volumen de datos considerado en cada escenario se resume en la Tabla 2.3. A continuación se describirán con mayor detalle los aspectos más relevantes de cada uno de los conjuntos de datos.

DARPA

El proyecto IDEVAL fue objeto de revisión en 1999, teniendo como propósito refinar su funcionamiento. La versión de 1999 (DARPA'99) resulta aún hoy en día una de las herramientas de evaluación y comparación de IDS más ampliamente usada, a pesar de sus graves limitaciones, basándose en el manejo de tráfico de tipo sintético. Esta versión contiene el tráfico capturado durante 5 semanas en una red real que simula la red de una base aérea. Además de las trazas de red, también se obtuvieron archivos de traza y monitorización de cada uno de los equipos conectados, posibilitando así una

Base datos	L1	L2	L3	LV	A1	A2	A3	AV
HUME	2,837	2,837	2,837	3,647	351	349	351	451
MARX	3,860	3,860	3,860	4,963	351	349	351	451

Tabla 2.4: Particiones de tráfico normal (L) y ataques (A) consideradas para DARPA'99

detección basada en host. El volumen total de tráfico capturado es de 743 MB, estando dividido en días y semanas, siendo generado con características diferentes durante cada una de las semanas consideradas.

Así, durante las dos primeras semanas, el tráfico circulante en dicha red era tráfico limpio, es decir, libre de ataques. Durante la 3ª semana se incluyeron instancias de ataques, que se encontraban debidamente etiquetadas y clasificadas. Finalmente, durante la 4ª semana se generó tráfico de ataques, no estando identificados.

Para los fines de nuestro trabajo, se ha tomado el tráfico HTTP limpio (semanas 1 y 2) de la versión DARPA'99 con destino a los servidores web Hume y Marx existentes en el entorno de evaluación. Debido a la antigüedad y bajo número de los ataques existentes, el uso del tráfico de la 3ª semana fue descartado. Análogamente, al no estar etiquetados los ataques, la 4ª semana tampoco fue incluida en el conjunto de datos. En consecuencia, para disponer del tráfico de ataques necesario, en trabajos previos [Estevez-Tapiador et al., 2003] se generaron sintéticamente varios de ellos en un entorno equivalente al del proyecto IDEVAL a partir de los ataques descritos en *ArachNIDS* [ArachNids, 2003]. Como resultado se obtuvo la base de datos denominada "Ataques", que incluye 1.477 instancias de 65 ataques diferentes. En consecuencia, en las bases de datos que hemos denominado HUME y MARX sólo se incluyen peticiones HTTP de dos semanas, siendo tráfico sintético sin ataques.

El análisis de las URI contenidas en esta base de datos proporciona un total de 12.155 palabras, con un vocabulario asociado de 354 palabras diferentes, para la base de datos HUME y, para MARX, un total de palabras de 16.539 con 362 palabras diferentes.

Finalmente, de acuerdo a la metodología propuesta, se considerará la aplicación de la técnica *leave-one-out* a partir del establecimiento de varias particiones de datos (Tabla 2.4). Se ha usado la notación L para designar datos limpios y A para ataques. Las particiones L1, L2 y L3, junto con A1, A2 y A3, serán utilizadas para entrenamiento y evaluación, mientras que las particiones LV y AV se reservan para la validación de los sistemas desarrollados. Por tanto, únicamente se utilizarán tras el ajuste y evaluación de los métodos propuestos.

CERES

El conjunto de datos CERES fue obtenido a partir del tráfico generado hacia el servidor departamental entre los días 14 de diciembre del 2007 y 6 de enero de 2008. El volumen total de tráfico capturado es de 609 MB, existiendo un total de 437.957

paquetes HTTP. De estos, 143.826 corresponden a peticiones GET. Tras el correspondiente filtrado mediante *Snort*, no se detectaron paquetes de ataque. Finalmente, el conjunto de datos contiene 143.826 paquetes.

El análisis de las peticiones URI proporcionó un total de 492.096 palabras, con 4.533 palabras diferentes (vocabulario).

Para la aplicación de la técnica *leave-one-out* se han establecido 5 particiones (Tabla 2.5). Las particiones L1 a L4 se usarán para entrenamiento y test, mientras que la partición LV se reserva para validación.

PVHDB

Esta traza se ha capturado mediante la herramienta *tcpdump* del 28 de noviembre al 7 de diciembre del 2009 en un centro de enseñanza superior. En la Figura 2.17 se muestra el volumen de tráfico diario obtenido. De los datos adquiridos, 1.177.005 paquetes contienen peticiones GET, habiéndose filtrado el tráfico restante en primera aproximación.

A continuación se utiliza un sistema de detección de intrusos basados en firmas (*Snort*) para descartar el tráfico anómalo y/o ataques que la base de datos pudiera contener. *Snort* se ha configurado y parametrizado con las reglas VRT del 18 de septiembre del 2009 que afectan el contenido del URI. Después de evaluar la base de datos con *Snort*, se han detectado 16 paquetes con instancias de ataque. El número de alertas emitidas por *Snort* es poco significativo comparado con el volumen de tráfico evaluado. Continuando con la metodología, los paquetes marcados como alertas por *Snort* son descartados de la base de datos, quedando 1.176.989 paquetes de tráfico GET "limpio". Las características más relevantes de la traza se resumen en la Tabla 2.3.

Continuando con la metodología, se han extraído las cabeceras URI de cada uno de los paquetes de la base de datos. Durante esta fase se han detectado y descartado 208 paquetes en cuya cabecera se han detectado errores en el proceso de extracción. La base de datos con tráfico después de esta fase consta de 1.176.781 URI.

A continuación se segmentan los URI para extraer las palabras existentes. De este análisis se obtienen 65.068 palabras, estando el vocabulario compuesto por 28.025 palabras diferentes.

Finalmente, se han establecido las particiones a utilizar, de forma análoga a la realizada para CERES (Tabla 2.5). En la Tabla 2.6 se muestra la clasificación del conjunto de datos por tipo de tráfico para las bases de datos disponibles.

Particiones de entrenamiento, evaluación y validación

Como se ha mencionado en cada caso, las bases de datos utilizadas deben ser particionadas, en primera aproximación, en dos bloques para su uso en entrenamiento y evaluación, por una parte, y validación, por otra. Los conjuntos de validación deben contener tanto tráfico limpio como ataques, mientras que los conjuntos que se vayan a utilizar para entrenamiento deben corresponder a tráfico limpio. De esta forma, se han categorizado cada una de estas bases de datos en dos conjuntos, de manera aleatoria,

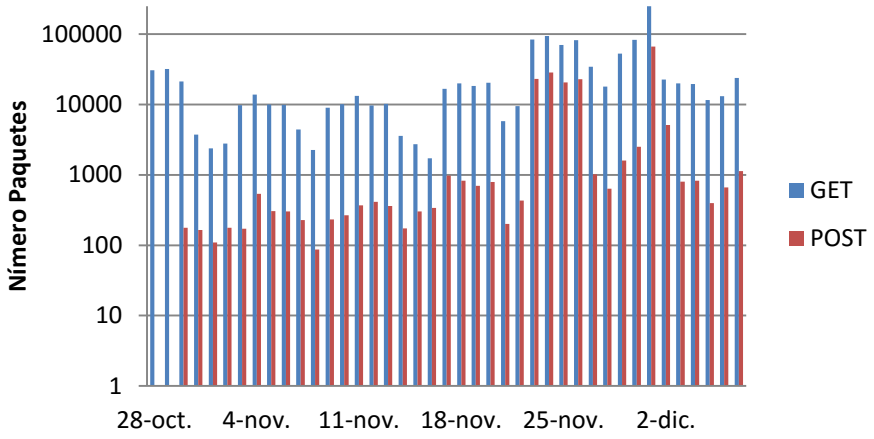


Figura 2.17: Histograma de la captura de la base de datos PVHDB

Base datos	L1	L2	L3	L4	LV
CERES	25.518	25.518	25.518	25.518	43.746
PVHDB	205.895	205.896	205.895	205.896	352.964

Tabla 2.5: Particiones de tráfico normal para bases de datos CERES y PVHDB

Base Datos	DARPA '99		CERES	PVHDB
	HUME	MARX		
Tráfico limpio	12.138	16.505	143.826	1.176.781
Tráfico ataques	1455	1455	0	16

Tabla 2.6: Tráfico normal y tráfico de ataques en los conjuntos de tráfico considerados

asignando el 70% para el entrenamiento y pruebas del sistema y el 30% restante para la validación. Los volúmenes de datos resultantes se muestran en la Tabla 2.7.

Adicionalmente, para la aplicación de la metodología *leave-one-out*, se han establecido particiones disjuntas en el bloque de entrenamiento y evaluación de forma que se puedan realizar combinaciones de las mismas (Tabla 2.8a y Tabla 2.8b). Así, se considera el reagrupamiento de las particiones para obtener los conjuntos finales que se usarán en entrenamiento y evaluación con o sin tráfico de ataques, según el caso, y que se muestran en la Tabla 2.8. Los experimentos a realizar al aplicar la técnica *leaving-one-out* corresponden, por tanto, a lo indicado en el Apartado 2.5.3.

2.5.2 Conjunto de datos de ataques

Como se indicó en el Apartado 2.4, la adquisición del conjunto de datos de ataque se realiza según diversas aproximaciones que se suceden secuencialmente.

Base datos	Entrenamiento y Pruebas (70%)	Validación (30%)
HUME	8.511 paquetes	3.647 paquetes
MARX	11.580 paquetes	4.963 paquetes
CERES	102.073 paquetes	43.746 paquetes
PVHDB	823.582 paquetes	352.964 paquetes

Tabla 2.7: Particionado de tráfico limpio para entrenamiento, pruebas y validación

Entrenamiento	Evaluación	
	Tráfico Limpio	Tráfico Anómalo
L1 u L2	L3	A3
L1 u L3	L2	A2
L2 u L3	L1	A1

a) Usando 3 particiones (DARPA)

Entrenamiento	Evaluación	
	Tráfico Limpio	Tráfico Anómalo
L1 u L2 u L3	L4	BD-Ataques
L1 u L2 u L4	L3	BD-Ataques
L1 u L3 u L4	L2	BD-Ataques
L2 u L3 u L4	L1	BD-Ataques

b) Usando 4 particiones (CERES y PVHDB)

Tabla 2.8: Combinaciones de particiones para la realización de experimentos mediante la técnica *leaving-one-out* (no se muestra la validación)

Uso de analizadores de vulnerabilidades

La primera aproximación consiste en la generación de tráfico de ataques a partir de diversos analizadores de vulnerabilidades. En la Tabla 2.9 se muestran los detectores utilizados, los resultados de la captura y su clasificación mediante Snort, utilizando las reglas que afectan al contenido de las URI (*uricontent*) incluidas en el conjunto de reglas oficial (VRT) de fecha 28/12/08. La columna "Únicas" corresponde a los tipos de ataque detectados (identificadores de ataque únicos detectados), mientras que la columna "Total" representa el número total de instancias de ataques detectados. A este respecto, es importante notar que un mismo ataque puede presentar varias instancias resultantes de alguna pequeña variación de los parámetros del mismo. Un ejemplo podría ser el ataque *http://192.168.1.2/ataque.php?variable=[SQL injection]*, siendo las instancias (variantes) del ataque las que surgen de modificar el valor del código *[SQL injection]*. Como se puede observar, no todo el tráfico capturado (tráfico GET) es etiquetado como ataque, siendo significativas las discrepancias. Por consiguiente, se procede a aplicar el procedimiento de ingeniería inversa a partir de las firmas de *Snort*.

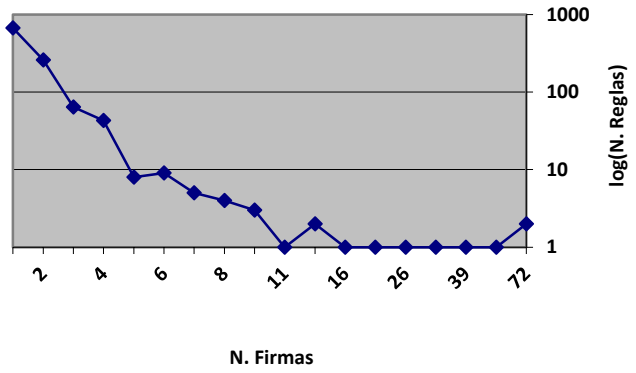


Figura 2.18: Histograma de la relación entre firmas y reglas

	Paquetes		Alertas	
Detector	Total	GET	Únicas	Total
wikto	10.944	2.157	237	491
Nikto (ev.7)	12.176	2.085	102	462
Nikto (ev.8)	15.987	2.706	312	983
W3af	14.127	2.678	317	975
Nstealth	10.9659	10.977	482	4.711
Total	162.893	20.603	1.450	7.622

Tabla 2.9: Tráfico generado a partir de buscadores de vulnerabilidades. La clasificación se ha realizado mediante Snort con las reglas VRT de 28/12/08

Ingeniería inversa

Se han considerado las reglas VRT de 16/08/2007, con 8.262 reglas de las que 2.375 afectan al contenido de los paquetes HTTP. Como se indicó en el Apartado 2.4, la correspondencia entre reglas y firmas no es unívoca, existiendo habitualmente más reglas que firmas. Así, las 2.375 reglas se utilizan para proporcionar cobertura a 1.076 firmas diferentes, existiendo algunas firmas que se han traducido en un número elevado de reglas (hasta 72 reglas por firma). En la Figura 2.18 se muestra un histograma de la distribución entre firmas y reglas en el conjunto considerado. Como puede observarse, la mayoría de las firmas se instancian en una o dos reglas. En particular, 669 firmas tienen una única regla asociada, mientras que otras 260 firmas se instancian en 2 reglas.

El análisis de las firmas resultantes en función del origen de la referencia proporciona los datos mostrados en la Tabla 2.10. La fila URL corresponde a las firmas asociadas a un URL fijo que, por tanto, no necesita parametrización y para el que no es necesario localizar un *exploit*. La fila "Sin referencia" muestra el número de reglas que no presentaban el campo "*reference*", lo cual significa que son reglas aportadas por el personal certificado de *Snort*. Éstas no serán consideradas en lo sucesivo. Las reglas con

Reglas Snort por referencia		Núm. de firmas
URL		347
Sin referencia		234
Bugtraq		553
Nessus		45
Common Vulnerabilities and Exposures (CVE)		73
ARACHNIDS		61
	Bugtraq	49
	CVE	3
	URL	1
	Sin referencia	7

Tabla 2.10: Distribución de las firmas de Snort según la referencia

Fuente	Total paq.	Paq.ataque
ArachNIDS	1.345	96
Nessus	916	73
Bugtraq	38.120	953
CVE	817	59
Total	41.198	1.181

Tabla 2.11: Tráfico de ataques recopilados mediante el uso de *exploits*

referencia de *Arachnids* tuvieron que ser revisadas debido a que el portal asociado, *www.whitehats.com*, ha sido dado de baja. Las 61 firmas incluidas, tras su revisión, fueron redistribuidas entre las restantes fuentes como se muestra en la Tabla 2.10. Las vulnerabilidades descritas en *Bugtraq* (www.securityfocus.com), *Nessus* (www.nessus.org) y *CVE* (cve.mitre.org) fueron consideradas para la recopilación de los *exploits* correspondientes.

A partir de esta información y, en su caso, de los *exploits* recopilados de cada fuente, se lanza una batería de ataques de la que resultan 1.181 paquetes GET (Tabla 2.11). El análisis de esta batería de ataques mediante Snort usando el mismo conjunto de reglas considerado como punto de partida proporciona únicamente 957 alertas. Se constata, en consecuencia, un comportamiento inconsistente en esta aproximación, poniéndose en duda la calidad de las reglas y de los *exploits* recopilados. Por tanto, se procede a complementar los ataques detectados siguiendo la tercera aproximación descrita en el Apartado 2.4.

Generación supervisada

Mediante la aplicación de los procedimientos descritos en el Apartado 2.4, se ha procedido a recopilar dos conjuntos de ataques, denominados RDB y OSVDB. El primero se ha obtenido a partir de la recopilación de 338 ataques, de los que se han

generado 707 instancias. La segunda fue obtenida a partir del proceso de recopilación y procesado de la base de datos disponible en internet en la dirección http://www.osvdb.org [OSVDB, 2009], obteniéndose un total de 6.896 ataques.

Finalmente, se ha realizado un análisis de las bases de datos de ataque resultantes a fin de determinar sus características y potencialidades de cara a su uso en el desarrollo de sistemas IDS. Como paso previo a este análisis, se ha procedido a la clasificación de los ataques contenidos en RDB, que constituirán el núcleo de la experimentación.

Clasificación de los ataques

La evaluación del rendimiento de los IDS se realiza, fundamentalmente, en base a las tasas de FP y TP. Sin embargo, a fin de determinar la respuesta ante ataques de diversa naturaleza, lo que puede facilitar la incorporación de mejoras y el análisis de resultados, puede resultar conveniente que el conjunto de ataques utilizado se encuentre categorizado en función del tipo al que pertenecen.

A este fin se han considerado dos taxonomías disponibles en Internet: la taxonomía OSVDB [OSVDB, 2009], que incluye una clasificación de todas las vulnerabilidades existentes debidamente actualizada; y la taxonomía *Open Web Application Security Project* (OWASP) [OWASP, 2015], que realiza una clasificación únicamente de los ataques a las aplicaciones web. Ambas taxonomías se complementan una a otra, debido a que OSVDB hace una clasificación muy general y OWASP es más específica. Por tanto, se ha decidido hacer uso de ambas para la clasificación de los ataques.

En consecuencia, cada uno de los ataques contenidos en la base de datos RDB ha sido categorizado atendiendo a ambas taxonomías, obteniéndose los resultados mostrados en la Tabla 2.12. Como puede observarse, ambas clasificaciones son incompatibles, en el sentido de que, en general, ataques de un mismo tipo según cualquiera de las clasificaciones son incluidos en clases diferentes de acuerdo a la otra. La excepción la constituyen los ataques de denegación de servicio (DoS), que, como se constata en la Tabla 2.12, son clasificados de igual forma de acuerdo a ambas taxonomías.

Se puede observar que el mayor número de ataques recopilados, así como las instancias de estos, corresponden a los de gestión de autenticación, de acuerdo a OSVDB, con un total de 137 ataques con 313 instancias de ataques. En contraste, la clasificación usando OSWAP muestra un notorio cambio en el que la mayor representación se encuentra en la inyección de comandos (*command injections*), con un total de 102 ataques y 223 instancias de ataques.

2.5.3 Análisis de los ataques

La base de datos de ataques RDB ha sido contrastada mediante *Snort*, a fin de determinar sus características y potencialidades respecto del proceso de detección de intrusiones. Por otra parte, al ser *Snort* uno de los IDS más ampliamente utilizados, los

OSVDB / OWASP	6 Input Manip.	8 DoS	10 Inform. disclosure	12 Auth. Manag.	18 Lost Integrity	29 Web Related	TOTAL
1 Absolute path traversal	6/8		34/66	24/55		6/11	70/140
2 Full path disclosure	2/5		3/8	4/12			9/25
3Account lockout attack	3/5			3/11			6/11
4 Path manipulation	5/5		4/16	1/1	1/2		11/24
5 Relative path manipulation			10/19	19/28		1/1	30/48
6 Forced browsing	7/12		2/2	7/13		3/7	19/34
7 Denial of service		18/33		1/1			19/34
8 XSS using script in attributes	4/4		4/7	12/28			20/39
9 XSS cross-site scripting	1/1			2/5	1/1		4/7
10 Buffer overflow attack	2/4		1/1	13/22			16/27
11 Command injection	6/6		50/122	43/88		3/7	102/223
12 Resource Injection			2/2	1/2			3/4
13 Double encoding	5/5		6/16	5/39			16/60
14 Setting manipulation	3/5		3/3			2/3	8/11
15 SQL injection			3/7	2/8			5/15
Total	**44/60**	**18/33**	**122/269**	**137/313**	**2/3**	**15/29**	**338/707**

Tabla 2.12: Clasificación de los ataques en RDB según taxonomías OSVDB y OWASP. Se indican en formato A/B el número de ataques (A) y de instancias (B)

resultados obtenidos constituirán el sistema de referencia respecto al que se compararán los desarrollos de sistemas basados en anomalías que se realicen en nuestro equipo de investigación.

Para esta etapa se han tomado las reglas VRT del 23 de diciembre del 2008, que era el conjunto de reglas más recientes al realizar la experimentación. Una de las características de este tipo de reglas (reglas VRT) es la verificación y validación de cada una de ellas realizada por los expertos de *Sourcefire*.

Es importante reseñar que, para la realización de los experimentos, se ha deshabilitado la consideración del flujo de las conexiones (opción *flow* de las reglas), debido a que se usa *Snort* en modo off-line y únicamente se someten a análisis los paquetes GET, sin considerar la sesión en la que se incardinan (actividad de red generada, p.e., por el establecimiento de conexión TCP).

El conjunto de reglas considerado contiene 9.871 reglas, que han sido preprocesadas para seleccionar sólo las que afectan el protocolo HTTP, resultando un total de 2.635 reglas. De éstas, se han elegido y considerado únicamente las que afectan al contenido del URI, mediante el uso del campo *uricontent* en la regla, por ser este el objeto de la investigación en curso. Finalmente, el conjunto de reglas utilizado contiene 1.620 reglas (Figura 2.19).

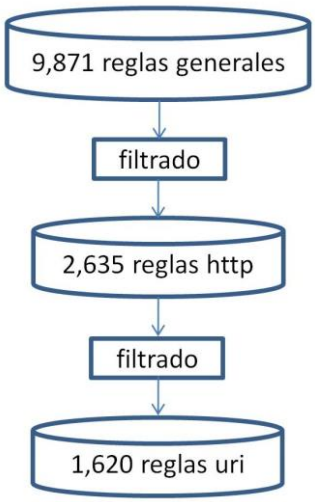


Figura 2.19: Selección de las reglas de *Snort*

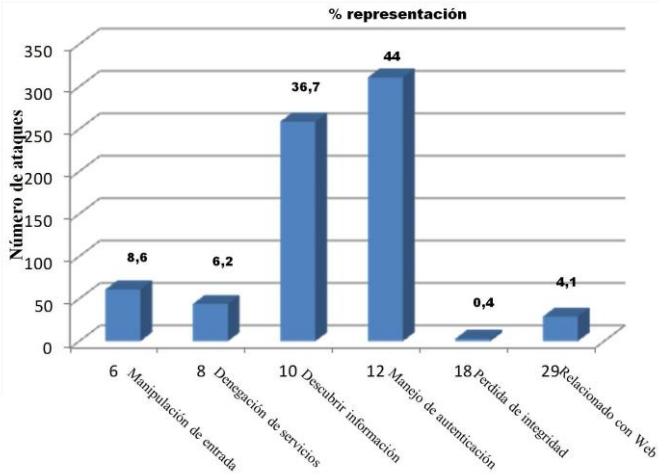


Figura 2.20: Ataques detectados por categoría, de acuerdo a OSVDB

Una vez preprocesado el conjunto de reglas a utilizar, se ha configurado adecuadamente Snort para que utilice este conjunto de reglas, deshabilitando los preprocesadores que pudieran alterar los URI.

Los resultados obtenidos se detallan en la Figura 2.20, en la que se muestra el número de ataques detectados de cada categoría, según la taxonomía OSVDB, así como el porcentaje que representan dentro de la muestra total de ataques detectados. Se puede observar que los ataques del tipo manejo de autenticación representan el 44% del total de ataques detectados, habiéndose detectado 265 de ellos. Por el contrario, los ataques de pérdida de integridad representan un 0,4 por ciento del universo total de ataques

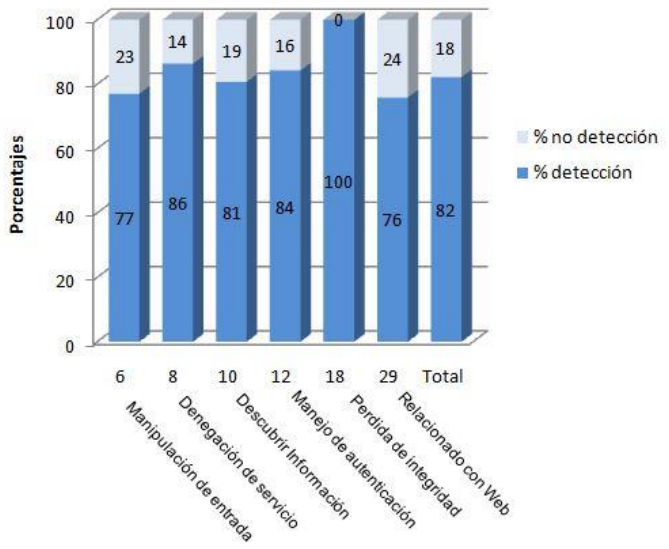


Figura 2.21: Tasas de detección en función de la categoría (OSVDB) utilizando Snort

detectados. Evidentemente, estas estadísticas corresponden a la distribución de los ataques detectados por categoría, siendo poco ilustrativas respecto de la capacidad de detección tanto global como por categoría.

En la Figura 2.21 se muestran los porcentajes relativos de ataques detectados y no detectados según su categoría. Como se puede observar, los ataques de pérdida de integridad son detectados con una muy alta eficiencia (100%), mientras que los ataques de manipulación de entrada y los relacionados con web presentan una tasa de detección relativamente pobre (en torno al 75%). Globalmente, se comprueba que Snort presenta un 17,83 por ciento de trazas de ataque no detectadas, por lo que su rendimiento ante esta batería de ataques puede calificarse como insuficiente.

Para mejorar la capacidad de detección se hace necesario mejorar la calidad de las reglas (recuérdese que las firmas consideradas proporcionarían una cobertura adecuada de los ataques) o desarrollar técnicas de detección alternativas, como puede ser la detección basada en anomalías o la combinación de ambas aproximaciones, objeto de investigación en nuestro grupo de trabajo y en la presente tesis.

3 El sistema de referencia

La evaluación y validación de las técnicas y/o mejoras que se desarrollen requiere de un escenario de trabajo y de unos conjuntos de datos que posibiliten, no sólo el análisis de los resultados, sino también su comparación con otras técnicas. De esta forma, para evaluar las posibles mejoras en el rendimiento de las técnicas propuestas será necesario comparar los resultados obtenidos en un entorno controlado con los proporcionados por otras técnicas de referencia.

En el Capítulo 2 se han descrito los conjuntos de datos a utilizar, así como las metodologías para la evaluación y validación que resultan de aplicación en el presente trabajo, es decir, se ha presentado el escenario de trabajo.

Este capítulo se centrará en la descripción del sistema de detección de intrusiones que ha sido utilizado como base para el desarrollo de las mejoras que se plantearán a lo largo del presente trabajo y que constituirá el sistema de referencia con el que se compararán los resultados obtenidos. De esta forma, en este capítulo se describirán las técnicas básicas utilizadas en el sistema IDS de referencia, sus fundamentos y los resultados de detección proporcionados por dicho sistema de referencia para las bases de datos consideradas.

El sistema de detección de intrusiones considerado como base es el denominado *Stochastic Structural Model* (SSM), propuesto en [Estevez-Tapiador, 2004a], que utiliza el modelado de Markov para representar el contenido de los mensajes intercambiados por un protocolo dado. La aplicación de este modelado fue descrita y analizada en detalle por sus autores en el caso del protocolo HTTP, debido tanto a su importancia como a su frecuente uso como vector de ataque en Internet. Por tanto, se presentarán también los aspectos más relevantes del protocolo HTTP para su utilización en el sistema de detección de intrusiones.

Finalmente, se detallarán los resultados relativos al rendimiento del detector SSM para las bases de datos consideradas. Para ello, se realizarán los experimentos necesarios para el ajuste de los parámetros del detector. Los resultados finales obtenidos en el presente capítulo corresponderán, en consecuencia, a los del sistema de referencia con el que se compararán los obtenidos a partir de las técnicas que se desarrollen.

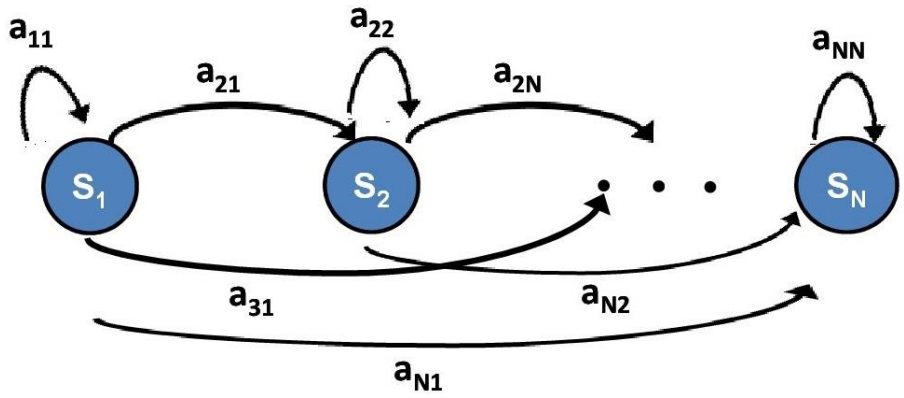


Figura 3.1 Grafo de un autómata de estados finitos

3.1 Modelado de protocolos mediante modelos de Markov

Los protocolos de comunicaciones presentan una estructura y funcionamiento claramente definidos y establecidos que posibilitan su descripción mediante técnicas formales. En particular, pueden ser descritos mediante autómatas de estados finitos [Sekar et al., 2002]. De esta forma, a partir de las especificaciones del protocolo, es posible inferir un autómata de estados finitos que represente su operación. En particular, los protocolos en uso en Internet son descritos en los denominados RFC (del inglés, *Request for Comments*), que incluyen toda la información necesaria para la especificación formal de los mismos y, en consecuencia, la definición del autómata de estados finitos asociado.

3.1.1 Autómatas de estados finitos

Un autómata de estados finitos es un modelo matemático que puede ser utilizado para describir la evolución de algunos sistemas. Un autómata, *M*, queda definido mediante una tupla compuesta por los siguientes elementos [Hopcroft, 2002]:

$$M = (\Gamma, \theta, \delta, S, F) \tag{3.1}$$

siendo:

- $\Gamma = \{S_1, S_2, \ldots, S_N\}$ un conjunto finito de estados, S_i, en los que se puede encontrar el sistema.
- $\Theta = \{\upsilon_1, \upsilon_2, \ldots, \upsilon_L\}$ un conjunto finito de símbolos u observaciones, υ_i, llamado *alfabeto* del autómata.

- $\delta : \Gamma \times \theta \rightarrow \Gamma$ una función de transición entre estados.
- $S \subseteq \Gamma$ el conjunto de posibles estados iniciales del sistema.
- $F \subseteq \Gamma$ el conjunto de estados finales que puede alcanzar el sistema.

El funcionamiento del autómata puede ser descrito mediante un grafo orientado (Figura 3.1) en el que los nodos se corresponden con los estados del autómata y los arcos con las transiciones posibles. El funcionamiento del autómata es tal que, a partir de un estado inicial dado, se producirán transiciones entre el estado actual y el siguiente de acuerdo a la función de transición entre estados. Cada transición está asociada a un símbolo del alfabeto, de forma que se producirán dos secuencias de eventos: la secuencia de estados que sigue el sistema y la secuencia de símbolos generados/observados durante dicha evolución del sistema.

Los autómatas de estados finitos se pueden clasificar en dos grandes tipos: deterministas o no deterministas [Brookshear, 1989]. En los autómatas deterministas, la evolución del sistema queda determinada, como su propio nombre indica, de forma unívoca por la secuencia de símbolos observados. Equivalentemente, la secuencia de símbolos observados queda determinada unívocamente a partir de la secuencia de estados. Por el contrario, en los autómatas no deterministas, dada una secuencia de símbolos de entrada o una secuencia de estados, existe más de una única secuencia de estados o de símbolos posibles, respectivamente.

Los autómatas deterministas pueden ser utilizados para describir la operación de los protocolos de comunicación en base a los mensajes intercambiados por los mismos [Brookshear, 1989], que serán considerados los elementos del alfabeto. Un ejemplo de autómata asociado a un protocolo es el autómata de TCP [Sekar et al., 2002], ampliamente utilizado para mostrar su operación y elementos (Figura 3.2).

La obtención del autómata asociado a un protocolo dado se puede realizar considerando dos aproximaciones: a partir de la propia especificación del protocolo o mediante la inferencia de dicho autómata a partir de la observación de secuencias de intercambios de mensajes entre entidades que utilicen dicho protocolo. A partir de la primera de las aproximaciones se obtendrá un autómata completo que represente todos los estados y secuencias permitidas, de acuerdo a dicho protocolo. Sin embargo, presenta el grave inconveniente de necesitar intervención manual para su obtención. Por el contrario, la inferencia a partir de secuencias observadas puede ser realizada de forma automática mediante diversas técnicas ampliamente descritas en la bibliografía [Mahoney & Chan, 2001].

En cualquier caso, la utilidad de los autómatas deterministas en el contexto del presente trabajo resulta muy limitada, ya que únicamente permitirían determinar, dado un autómata asociado a un protocolo, si una secuencia de mensajes intercambiados puede ser decodificada mediante dicho autómata. Esto es, si la secuencia de mensajes corresponde a la especificación del protocolo y, por tanto, es gramaticalmente correcta. Como se discutirá más adelante, una parte importante de los ataques son estructuralmente correctos de acuerdo al protocolo, es decir, siguen la gramática

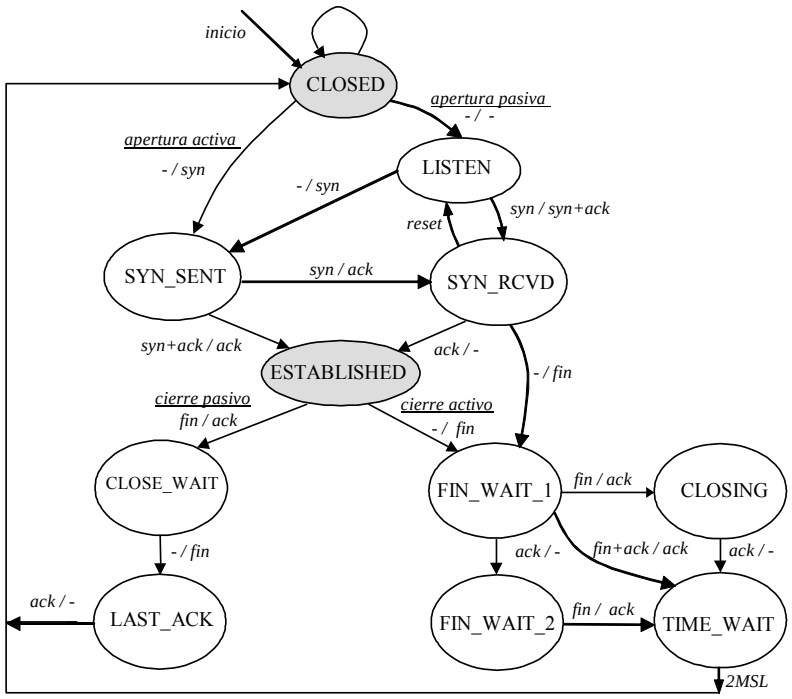


Figura 3.2 Diagrama del autómata determinista del protocolo TCP

correcta, residiendo su naturaleza maliciosa en aspectos relacionados con la semántica. En este sentido, los autómatas no deterministas permiten incorporar información adicional que puede estar relacionada con la semántica o el contexto (el contenido del mensaje) y que, en consecuencia, permitirán discriminar ataques cuya estructura sintáctica sea correcta.

Un tipo de autómata no determinista de especial interés para los objetivos del presente trabajo son los autómatas de estados finitos probabilísticos [Brookshear, 1989]. En éstos se consideran probabilidades asociadas a los estados y/o a los símbolos, de forma que se atribuye una naturaleza probabilística tanto a la secuencia de estados que sigue el sistema como a la de símbolos observados.

En particular, resultan especialmente relevantes los autómatas asociados a las cadenas y modelos de Markov, por lo que, a continuación, se describirán brevemente los fundamentos y elementos de un modelo de Markov.

3.1.2 Modelos de Markov

En los procesos de Markov [Ching & Ng, 2005], los eventos futuros dependen de los inmediatamente anteriores. Así, en un modelado de Markov de orden *n*, la probabilidad de que se produzca un evento depende de los *n* eventos anteriores. Son, por tanto, procesos con memoria en los que el estado en que se encuentra el sistema en un instante determinado está directamente relacionado con la historia previa.

En el caso de los modelos de Markov discretos, se considera un conjunto de estados en los que puede encontrarse el sistema y un conjunto de eventos observables asociados a dichos estados[1]. El sistema evoluciona a lo largo del tiempo (discreto) mediante transiciones entre estados, generándose un símbolo en cada una de dichas transiciones. Tanto las transiciones entre estados como la producción de los símbolos observables se rigen por distribuciones de probabilidad.

Formalmente un modelo de Markov discreto, λ, se define como la quíntupla:

$$\lambda = (\Gamma, \theta, A, B, \Pi) \tag{3.2}$$

siendo

- $\Gamma = \{S_1, S_2, \ldots, S_N\}$ es el conjunto de *N* estados del modelo. Se denotará mediante q_t el estado del sistema en el instante de tiempo *t*.
- $\theta = \{\upsilon_1, \upsilon_2, \ldots, \upsilon_M\}$ es el vocabulario del modelo, esto es, el conjunto de los *M* posibles símbolos o eventos observables del modelo. Se denotará mediante O_t al símbolo observado en el instante *t*.
- **A** es una matriz *NxN* de probabilidades de transición entre estados, de forma que:

$$A = [a_{ij}], 1 \leq i, j \leq N$$
$$a_{ij} = P(q_t = S_j | q_{t-1} = S_i) \tag{3.3}$$

  esto es, a_{ij} es la probabilidad de transitar al estado S_j cuando el sistema se encuentra en el estado S_i en el instante de tiempo anterior.
- **B** es una matriz *NxM* de probabilidades de generación u observación de los símbolos, de forma que:

$$B = [b_{ik}], 1 \leq i \leq N, 1 \leq k \leq M$$
$$b_{ik} = P(O_t = v_k | q_t = S_i) \tag{3.4}$$

  es decir, b_{ik} es la probabilidad de observar a la salida el símbolo v_k cuando el sistema se encuentra en el estado S_i.
- Π es el vector de probabilidades del estado inicial:

$$\Pi = [\pi_i], 1 \leq i \leq N$$
$$\pi_i = P(q_1 = S_i) \tag{3.5}$$

[1] Diversos autores utilizan interpretaciones diferentes para la relación entre los observables y los estados. Algunos consideran que éstos están directamente asociados a los estados, siendo generados por el sistema al encontrarse en dicho estado, originando simultáneamente una transición hacia otro estado. Alternativamente, otros autores relacionan los observables directamente con las transiciones, de forma que éstos se generan durante las transiciones entren estados. Ambas interpretaciones son formalmente equivalentes y serán utilizadas indistintamente en esta memoria.

es decir, π_i es la probabilidad de que el sistema se encuentre en el estado S_i en el instante inicial ($t=1$).

Dada la naturaleza probabilística de los elementos del modelo de Markov, se debe cumplir:

$$\sum_{j=1}^{N} a_{ij} = 1, \forall\, i = 1, 2, \dots N$$

$$\sum_{k=1}^{M} b_{ik} = 1, \forall\, i = 1,2, \dots, N \qquad (3.6)$$

$$\sum_{i=1}^{N} \Pi_i = 1$$

Un modelo de Markov puede ser utilizado para modelar un proceso de generación de secuencias o símbolos que, a su vez, puede ser utilizado como parte de un sistema de reconocimiento o clasificación. Para ello es necesario, en primer lugar, determinar el modelo capaz de simular el proceso objeto de análisis que, a su vez, quedará determinado por los valores de los diferentes parámetros que lo constituyen. La obtención de estos valores constituye el denominado problema de *estimación* o *entrenamiento*. En segundo lugar, se requiere también un procedimiento para la obtención de la probabilidad de generación por el modelo de las secuencias observadas. Este problema es el denominado de *evaluación* de secuencias. Ambos problemas y sus soluciones se encuentran ampliamente descritos en la bibliografía [Rabiner, 1989], por lo que continuación se describirán brevemente las soluciones comúnmente adoptadas.

Evaluación

Dada una secuencia de observaciones, $O = o_1, o_2 \dots, o_T$, la secuencia de estados por los que evoluciona el sistema durante la generación de dicha secuencia, $\mathrm{Q} = \mathrm{q}_1, \mathrm{q}_2 \dots, \mathrm{q}_\mathrm{T}$ y un modelo, λ, el problema de la evaluación puede ser descrito como la obtención de la probabilidad condicional de que dicha secuencia de observaciones haya sido generada por el modelo considerado al evolucionar de acuerdo a la secuencia de estados dada, $P(O|\lambda, Q)$. Esta probabilidad viene dada por [Ching & Ng, 2005]:

$$P(O|\lambda, Q) = \pi_{q_1} b_{q_1 o_1} \prod_{t=1}^{T-1} a_{q_t q_{t+1}}\, b_{q_{t+1} o_{t+1}} \qquad (3.7)$$

Habitualmente, para evitar problemas de desbordamiento asociados a la representación interna de los datos, se evalúa el logaritmo de la probabilidad en lugar de la probabilidad, por lo que la expresión a utilizar resulta:

$$\begin{aligned} logP(O|\lambda,Q) &= log(\pi_{q_1}) + log(b_{q_1 o_1}) + \sum_{t=1}^{T-1} log(a_{q_t q_{t+1}}) \\ &+ \sum_{t=1}^{T-1} log(b_{q_{t+1} o_{t+1}}) \end{aligned} \tag{3.8}$$

Entrenamiento

Dada una secuencia de observaciones $O = o_1, o_2 \dots, o_T$ correspondiente al proceso que se desea modelar, y su correspondiente secuencia de estados, $Q = q_1, q_2 \dots, q_T$, el problema de la estimación o entrenamiento puede ser descrito como el ajuste de los parámetros de un modelo, $\lambda = (\Gamma, \theta, A, B, \Pi)$, de forma que se maximice la probabilidad de generación de dicha secuencia de observaciones por el modelo, dada la secuencia de estados observada.

En general, para una estimación correcta de los parámetros del modelo, será necesario disponer de un conjunto representativo y suficiente de secuencias de observaciones junto con las secuencias de estados asociados a cada uno de ellos. De esta forma, el entrenamiento no se realizará en base a una única pareja de secuencias de estados y observaciones, sino a un conjunto de ellos (entrenamiento multisecuencia).

Así, consideremos un conjunto de entrenamiento Ω de *L* pares de secuencias de observaciones y estados,

$$\Omega = \{(O^1, Q^1), (O^2, Q^2), \dots, (O^L, Q^L)\} \tag{3.9}$$

tal que

$$\begin{aligned} O^i &= \{o_1^i, o_2^i, \dots, o_{T_i}^i\} \\ Q^i &= \{q_1^i, q_2^i, \dots, q_{T_i}^i\} \end{aligned} \tag{3.10}$$

donde hemos denotado mediante superíndices la secuencia del conjunto de entrenamiento y mediante subíndices el elemento dentro de la secuencia.

Los parámetros del modelo de Markov se obtendrán sin más que hacer un recuento de las frecuencias de aparición relativas de los símbolos y los estados [Estevez-Tapiador, 2004a]. Así

$$\begin{aligned} a_{ij} &= \frac{\sum_{s=1}^{L}\sum_{t=1}^{T_s-1}\delta(q_t^s = s_i \ , q_{t+1}^s = s_j)}{\sum_{s=1}^{L}\sum_{t=1}^{T_s-1}\delta(q_t^s = s_i)} \\ b_{ij} &= \frac{\sum_{s=1}^{L}\sum_{t=1}^{T_s}\delta(o_t^s = v_j \ , q_t^s = s_i)}{\sum_{s=1}^{L}\sum_{t=1}^{T_s-1}\delta(q_t^s = s_i)} \\ \pi_i &= \frac{\sum_{s=1}^{L}\delta(q_1^s = s_i)}{L} \end{aligned} \tag{3.11}$$

siendo *δ( )* una función que toma el valor 1 cuando todos sus argumentos sean verdaderos y 0 en caso contrario. Básicamente, es una función de recuento de observaciones cumpliendo una determinada condición.

3.1.3 Parametrización

La aplicación de las técnicas de modelado previamente mencionadas a un protocolo de comunicaciones requiere de la identificación de los diferentes elementos que forman parte del protocolo con los elementos constitutivos de los autómatas. Habitualmente, las observaciones corresponden a símbolos o mensajes pertenecientes al repertorio del protocolo, mientras que los estados se identifican con elementos abstractos correspondientes a situaciones o contextos en los que puede encontrarse el sistema y de los que dependen las posibles actuaciones posteriores. Como ya se ha mencionado, el autómata correspondiente a un protocolo puede inferirse a partir de su especificación. Dada la naturaleza de los protocolos de comunicaciones, un protocolo bien definido debe quedar caracterizado por un autómata de estados finitos determinista [Estevez-Tapiador et al., 2005].

Siguiendo con el ejemplo correspondiente a TCP mostrado en la Figura 3.2, se puede observar que los mensajes son los diferentes *flags* que pueden ser utilizados en el protocolo TCP (e.g. RST, SYNC, etc.) y los estados se identifican con las posibles situaciones o contextos por los que puede ir atravesando el sistema cuando se hace uso del protocolo. En este ejemplo se puede observar, asimismo, que el autómata es determinista y que el repertorio de posibles observaciones en cada instante depende del estado en el que se encuentra el sistema.

A fin de establecer los elementos a analizar y, en su caso, el preprocesamiento necesario para modelar un protocolo a partir del tráfico observado en la red, hemos de considerar los bloques utilizados en los intercambios de datos. Estos bloques, denominados unidades de datos del protocolo (PDU, del inglés *Protocol Data Unit*) se forman por la concatenación (encapsulado) de la información a transmitir con la correspondiente a cada uno de los protocolos implicados en la comunicación. Dado que el modelado se realizará para la operación de uno de dichos protocolos, se hace necesaria una primera etapa de preprocesado de las PDU para extraer la información correspondiente a la capa/protocolo requerido. Dada la composición de las PDU, esta operación consistirá, básicamente, en la extracción del campo correspondiente, que vendrá convenientemente delimitado en la estructura de la PDU.

Tras el preprocesado, será necesario analizar los parámetros y/o mensajes que se encuentren codificados en el campo extraído y, en su caso, establecer una representación simbólica de dichos valores que pueda ser utilizada en el modelado. Este proceso será denominado parametrización en lo que sigue.

3.2 El protocolo HTTP

Un volumen bastante significativo del tráfico en Internet va dirigido a servicios y aplicaciones web. Por otra parte, las vulnerabilidades basadas en web representan una parte sustancial de los riesgos de seguridad de las redes de computadoras [Kruegel et al., 2005]. En consecuencia, se han realizado numerosos trabajos de investigación usando los diferentes paradigmas existentes de sistemas de detección de intrusos para

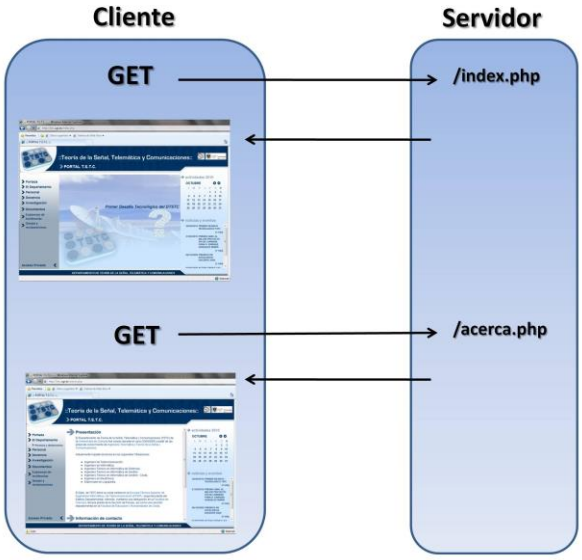


Figura 3.3: Cómo trabaja el protocolo HTTP

detectar ataques que sean transportados mediante el protocolo HTTP utilizado en los sistemas web. A continuación se detallarán los aspectos más relevantes de dicho protocolo a fin de obtener un modelo del mismo que pueda ser incorporado en sistemas de detección de intrusiones.

El protocolo HTTP (*HyperText Transfer Protocol*, protocolo de transferencia de hipertexto) es un protocolo de la capa de aplicación que opera sobre la pila de protocolos TCP/IP, cuya especificación original (versión 1.0) se encuentra detallada en el RFC 1945 [Berners-lee et al., 1996]. La versión 1.1 del protocolo se establece en el RFC 2068 [Fielding et al., 1999], que fue revisado en el RFC 2616 [Berners-Lee et al., 2005]. Un aspecto importante en este protocolo son los URI (*Uniform Resource Identifier*, identificador uniforme de recurso), que se encuentran especificados en el RFC 2396 [Berners-Lee, 1998]. La última versión del protocolo es la 1.2, recogida en el RFC 2774 [Nielsen et al., 2000].

HTTP es un protocolo cliente-servidor orientado a transacciones que utiliza el esquema petición-respuesta. La información transmitida mediante este protocolo se denomina *recurso*, quedando identificada, en la especificación original, mediante un localizador uniforme de recursos (URL, *Uniform Resource Locator*). El recurso puede ser un documento en formato HTML, un archivo, el resultado de la ejecución de un programa, una consulta a una base de datos, etc. El concepto de URL fue generalizado al de URI, ya mencionado, para incluir otros tipos de recursos asociados a otros protocolos. En este sentido, los términos URL y URI son equivalentes en el contexto del protocolo HTTP y serán usados indistintamente en lo que sigue. En la Figura 3.3 se muestra un esquema de la operación habitual del protocolo. En esta, el cliente envía un mensaje con la petición de un recurso disponible al servidor (mensaje GET) y este le

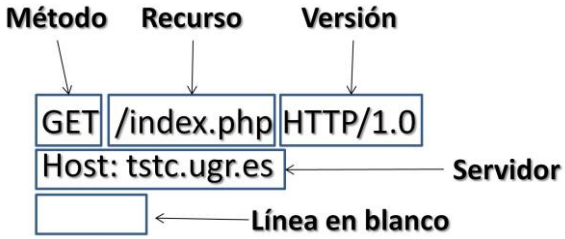


Figura 3.4: Estructura de una solicitud GET del protocolo HTTP

regresa el recurso para ser mostrado por el cliente a través de un programa denominado navegador (*browser* en inglés).

El protocolo HTTP usa en capas inferiores el protocolo TCP, operando habitualmente el servidor en el puerto 80. Es relevante que el servidor no mantiene información acerca del cliente entre peticiones sucesivas. En la versión original, sólo un objeto puede ser enviado en cada conexión, abriéndose y cerrándose la misma para cada intercambio de datos. A partir de la versión 1.1, la conexión es persistente, es decir, múltiples objetos pueden ser enviados sobre una única conexión TCP. De los métodos de solicitud de recursos, los más usuales son GET, POST y HEAD. GET es un método utilizado para la solicitud de un recurso, mientras que POST es utilizado para el envío de datos para ser procesados por el recurso identificado. Finalmente, el método de solicitud HEAD opera de forma análoga al método GET con la única diferencia de que únicamente se solicita la cabecera del recurso. Otros métodos son PUT, OPTIONS, TRACE, DELETE y CONNECT.

El formato de los mensajes de solicitud y respuesta son similares, estando basados en secuencias de caracteres y orientados al idioma inglés. Ambos consisten en:

- Una línea inicial,
- cero o más líneas de cabecera,
- una línea en blanco (CRLF) y
- el cuerpo del mensaje (opcional), por ejemplo un archivo, datos de la consulta o salida de una consulta.

La línea inicial es diferente para la solicitud que para la respuesta. En el caso de la solicitud, presenta 3 partes separadas por espacios:

1. El nombre del método invocado,
2. el identificador del recurso solicitado y
3. la versión del protocolo HTTP utilizada, convenientemente formateada.

Un ejemplo de mensaje de solicitud se puede ver en la Figura 3.4.

El protocolo HTTP es un protocolo sin estado, por lo que no es posible establecer un autómata de estados finitos a partir de sus especificaciones que modele su operación. Sin embargo, el carácter transaccional y la estructura de los mensajes intercambiados, basados en caracteres alfanuméricos organizados en campos, posibilita el modelado individual de cada uno de dichos mensajes mediante un autómata. En particular, resulta relevante para el presente trabajo la estructura de los URI contenidos en mensajes de petición tipo GET considerados en el protocolo, por lo que será analizada a continuación.

3.2.1 Estructura de los URI

La estructura y contenido de los URI quedan definidos en el RFC 2396 [Berners-Lee, 1998], que, como se ha comentado, incorpora y generaliza el concepto original de URL. De acuerdo al mencionado documento, un URI es una cadena de caracteres que identifica de forma global y unívoca a un recurso disponible en una red. Se estructura en una serie de *campos*, cada uno de los cuales tiene asociado un significado en función de su posición en la cadena, y cuyos valores están asociados al recurso en cuestión.

El nombre asignado a cada uno de los campos de un URI varía en los diferentes RFC que hacen referencia al mismo, por lo que pueden encontrarse en la bibliografía diversas formas de describir la composición de un URI. De acuerdo a los objetivos del presente trabajo, consideraremos específicamente los URI que pueden ser transportados en peticiones GET del protocolo HTTP, cuya estructura puede ser descrita [Berners-Lee, 1998] como compuesta de los siguientes campos:

- **Protocolo** (*esquema* en el RFC2396): Protocolo de comunicaciones a utilizar. En nuestro caso, este campo siempre tendrá el valor “http”.
- ***Host***: nombre (recomendado) o dirección del equipo en el que se encuentra el recurso. El puerto (*:port*), en caso de estar presente, será considerado como parte de este campo.
- **Segmento de ruta**: cada uno de los elementos de la ruta que especifica la ubicación del recurso en el equipo (host). Pueden existir ninguno, uno o varios segmentos de ruta en un URI. La secuencia formada por todos los segmentos de ruta es denominada *referencia* del URI en el RFC. En caso de estar presente, el fragmento (*#fragment*), será considerado como parte de la referencia.
- **Consulta**: Una cadena con información que debe ser interpretada por el recurso. En el contexto considerado, la sintaxis de la consulta corresponde a la concatenación por parejas de dos tipos de campos:
  - **Atributo**: Conteniendo el nombre de una variable o una cadena.
  - **Valor**: Conteniendo el valor asignado al atributo.

Se puede definir, en consecuencia, un conjunto, *F*, de tipos de campos que pueden formar parte del URI,

$$F = \{P, H, S, A, V\} \tag{3.12}$$

donde se ha representado mediante:

- *P* el protocolo,
- *H* el host,
- *S* el segmento de ruta,
- *A* el atributo y
- *V* el valor.

El URI se construye por la concatenación de los diferentes campos que lo componen, y que se encuentran separados entre sí por caracteres o conjuntos de caracteres especiales. Un URI consta de un *protocolo* opcional, un *host* opcional, una secuencia de uno o más *segmentos de ruta*, que constituyen el denominado *path absoluto* (RFC2616) y, opcionalmente, una *consulta* compuesta por una secuencia de *atributos*, cada uno de ellos con un *valor* opcional. Por tanto, de acuerdo al RFC2616, un URI asociado al protocolo HTTP o URL responde al esquema general:

"http:" "//" host [":" puerto] [path_absoluto ["?" consulta]]

A modo de ejemplo, consideremos el URI

http://ceres.ugr.es/it/index.php?sec=100&tema=enlace

que podría ser incluido en una petición GET. Este URI se utilizaría para solicitar el documento *index.php* ubicado en el directorio *it* del servidor *http* en el ordenador *ceres.ugr.es*, al que se le pasan los parámetros *sec* con valor *100* y *tema* con valor *enlace* para la generación del documento resultante.

3.2.2 Segmentación de los URI

La interpretación de un URI requiere de la separación de sus elementos constitutivos (segmentación) y la determinación de los valores de cada uno de los campos. Debido a la forma en que se construyen, los URI pueden ser fácilmente segmentados en una secuencia de campos con sus valores asociados (*tokens*) a partir de la consideración de un conjunto de delimitadores, *D*. El valor de un campo en un URI corresponderá a la secuencia de caracteres entre dos delimitadores consecutivos, quedando el tipo de campo determinado por los dichos delimitadores.

Así, un URI *U* puede ser representado, en términos generales, por

$$U = \|_1\sigma_1\|_2\sigma_2 \cdots \|_s\sigma_s\,, \qquad \|_i \in D \tag{3.13}$$

siendo σ_i una cadena de caracteres ASCII de longitud variable correspondiente al valor de un campo.

En el caso de los URI asociados al protocolo HTTP, los delimitadores posibles son, de acuerdo al estándar:

$$D_{HTTP} = \{D_1, D_2, D_3, D_4, D_5, D_6\} \quad (3.14)$$

donde los elementos son:

D_1="/", delimitador de recurso,
D_2="?", delimitador de parámetros,
D_3="=", delimitador de asignación de atributo,
D_4="&", delimitador entre parámetros,
D_5= blanco (ASCII 32), delimitador de fin de recurso (EOR),
D_6="://", delimitador de protocolo.

De esta forma, es posible definir dos funciones de segmentación, $G_1(\ )$ y $G_2(\ )$, que obtengan, dado un URI *U*, la secuencia de los valores de los campos que componen dicho URI y la secuencia de tipos de campo, respectivamente

$$G_1(U) = \{\sigma_1, \sigma_2, \cdots, \sigma_s\}$$
$$G_2(U) = \{f_1, f_2, \cdots, f_s\} \; con \; f_i \in F \quad (3.15)$$

siendo *s* el número de campos que componen el URI considerado.

A modo de ejemplo, para ilustrar el funcionamiento del proceso de segmentación, considérense los siguientes URI válidos:

U_1=http://ceres.ugr.es/tareas/feb/descri.html
U_2=/observa/feb/maestro.php?mat=17&alumno=11
U_3=/tareas?unidad1=tres

Utilizando las funciones de segmentación $G_1()$ y $G_2()$, se obtendrían las siguientes segmentaciones:

$$G_1(U_1) = \{http, ceres.ugr.es, tareas, feb, descri.html\}$$
$$G_2(U_1) = \{P, H, S, S, S\}$$
$$G_1(U_2) = \{observa, feb, maestro.php, mat, 17, alumno, 11\}$$
$$G_2(U_2) = \{S, S, S, A, V, A, V\}$$
$$G_1(U_3) = \{tareas, unidad1, tres\}$$
$$G_2(U_3) = \{S, A, V\}$$

3.3 El IDS basado en SSM

La información contenida en las cabeceras de los paquetes que circulan por una red puede ser utilizada para la detección de intrusiones. A modo de ejemplo, Sekar y otros [Sekar et al., 2002] han utilizado algunos campos como las marcas de tiempo (*timestamp* en inglés) del protocolo IP para realizar la detección de intrusiones. Otros autores, como Mahoney y Chan [Mahoney & Chan, 2001] han utilizado campos de diferentes protocolos como Ethernet, IP y TCP para desarrollar sus detectores de intrusos. En el caso del protocolo HTTP, Kruegel y otros [Kruegel et al., 2005] han

utilizado las cabeceras de dicho protocolo para desarrollar un sistema de detección de intrusiones basado en anomalías.

En este contexto, Estévez [Estevez-Tapiador, 2004a] ha desarrollado y probado la efectividad de tres técnicas de detección de intrusos basadas en el uso de autómatas de estados finitos para modelar la información contenida en las cabeceras de algunos protocolos basados en el paso de mensajes. En particular, se ha mostrado su efectividad en el caso del protocolo HTTP.

La primera de estas técnicas, denominada *modelado estocástico básico* (BSM, *Basic Stochastic Modeling* en inglés) se basa en el uso de la función de densidad de probabilidad de los caracteres presentes en una unidad de datos del protocolo modelado, en este caso de HTTP, para crear un modelo de normalidad. El modelo utiliza dichas funciones de probabilidad en el contexto de un modelado basado en cadenas de Markov o, equivalentemente, en autómatas de estados finitos estocásticos.

La técnica anterior fue mejorada a partir de su evolución a otra técnica denominada *modelado estocástico segmentado* (PSM, *Partition-based Stochastic Modeling* en inglés). En esta se sigue utilizando un modelado basado en cadenas de Markov, pero se consideran los diferentes elementos que componen un URI como el elemento básico del modelado, en lugar de las secuencias de caracteres. La aproximación se fundamenta en la constatación de que todas las solicitudes realizadas a un mismo servidor tienen una estructura similar, compartiéndose elementos entre ellas. En este caso, se realiza la segmentación del URI, de acuerdo a la sintaxis asociada al protocolo, y se codifican cada uno de los valores o cadenas observadas (símbolos).

Las técnicas antes descritas han demostrado un incremento en la eficiencia cuando se va incorporando información al modelado de las estructuras del protocolo HTTP. Finalmente, Estévez [Estevez-Tapiador, 2004a] propuso otra mejora en el modelado en la técnica denominada *modelado estructural estocástico* (SSM, *Structural Stochastic Modeling* en inglés). Esta será la considerada en el presente trabajo como referencia para la introducción de posibles mejoras, ya que era esta técnica la que proporcionaba mejores resultados.

En consecuencia, a continuación se presentan los detalles de la técnica denominada SSM.

3.3.1 Modelado mediante SSM

La técnica SSM se basa en la teoría de los modelos de Markov. De acuerdo a esta, se define un autómata de estados finitos estocástico basándose en la estructura y sintaxis del protocolo modelado (HTTP). Este autómata permite el análisis de los mensajes y la evaluación de la probabilidad de que un mensaje haya sido producido por el modelo.

El elemento básico del modelado es la identificación de cada uno de los campos que componen una PDU con los estados del modelo, que son los que generarán los diferentes observables (valores de los campos). Por tanto, en el modelado SSM se

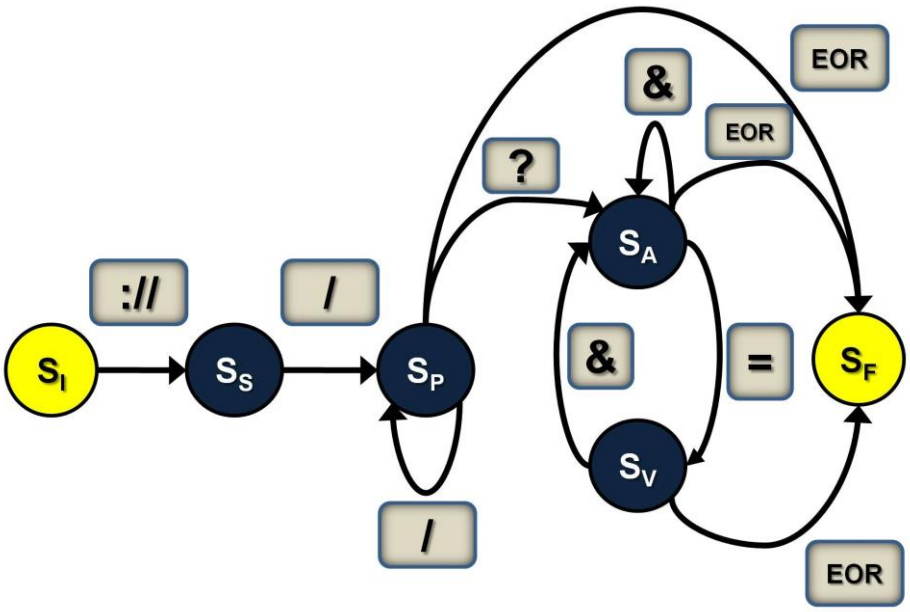


Figura 3.5 Grafo del autómata de estados finito del modelo SSM

considera que una PDU es una observación generada por la interacción de dos procesos estocásticos entrelazados:

- el de transición entre los diferentes estados (campos de la PDU), y
- la generación de símbolos (cadenas de caracteres) en cada una de las transiciones de acuerdo a distribuciones de probabilidad dependientes de los estados.

A partir de los campos descritos en el Apartado 3.2.2 para los URI del protocolo HTTP, y obviando el campo protocolo por tener en este caso siempre el mismo valor, se puede inferir la topología del autómata de estados finitos que permite modelar la producción de un URI. El autómata resultante se muestra en la Figura 3.5, en la que se han considerado 6 estados:

- un estado inicial, S_I,
- un estado servidor/host, S_S,
- un estado segmento de ruta (o *path*), S_P,
- un estado atributo, S_A,
- un estado valor, S_V, y
- un estado final, S_F.

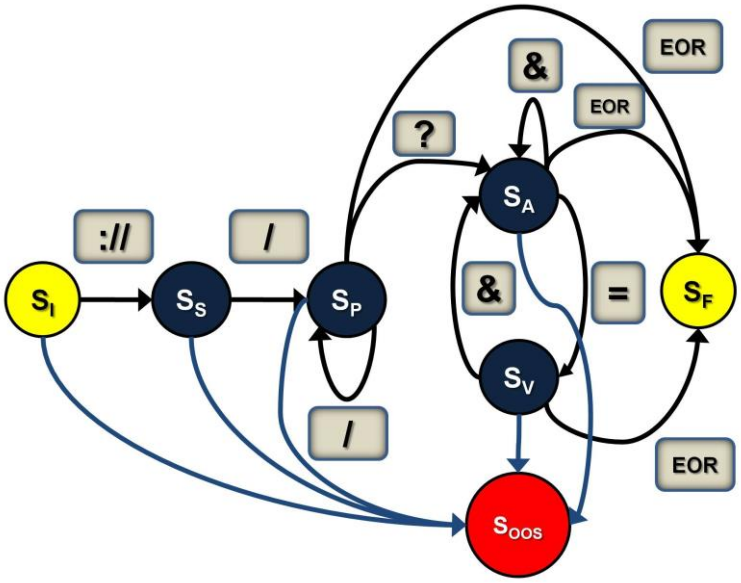


Figura 3.6 Autómata utilizado por la técnica SSM para modelar URI

Las transiciones posibles entre estados vienen determinadas por la especificación del protocolo. A su vez, dado un URI, y de acuerdo al proceso de segmentación, estas transiciones quedan unívocamente determinadas por el delimitador utilizado. En consecuencia, cada arco del modelo se ha etiquetado con el delimitador asociado a la transición correspondiente. Por otra parte, el autómata considerado representa a un único servicio, esto es, a una única pareja protocolo/host.

Como se ha mencionado previamente, la topología del modelo se ha deducido directamente a partir de la especificación del protocolo, pudiéndose utilizar el modelo para analizar las URI recibidas. De esta forma, una URI puede considerarse legítima, en el sentido de que cumple las especificaciones, si el modelo alcanza el estado final, S_F, cuando se analiza dicho URI. Para facilitar la aplicación de este modelo en un detector de anomalías, se puede añadir un estado adicional, S_{OOS} (Figura 3.6). Este estado será alcanzado por el sistema desde cualquier otro estado cuando el delimitador observado no esté permitido en el estado original. De esta forma, el estado S_{OOS} es un estado de "fuera de especificación", lo que implica que, si se alcanza durante la evolución del sistema, el URI que esté siendo analizado es incorrecto. Por este motivo, a las probabilidades de transición a dicho estado se les asigna un valor nulo y, en consecuencia, la probabilidad de observación de una cadena que alcance el estado de fuera de especificación tendrá un valor igualmente nulo.

Finalmente, dado que los delimitadores sólo pueden aparecer en determinados estados y su aparición determina el estado siguiente, se puede construir una tabla de transiciones en función de los delimitadores (Figura 3.7).

El modelo descrito permite únicamente representar las URI asociadas a peticiones GET. Sin embargo, se pueden generar modelos válidos para otros métodos

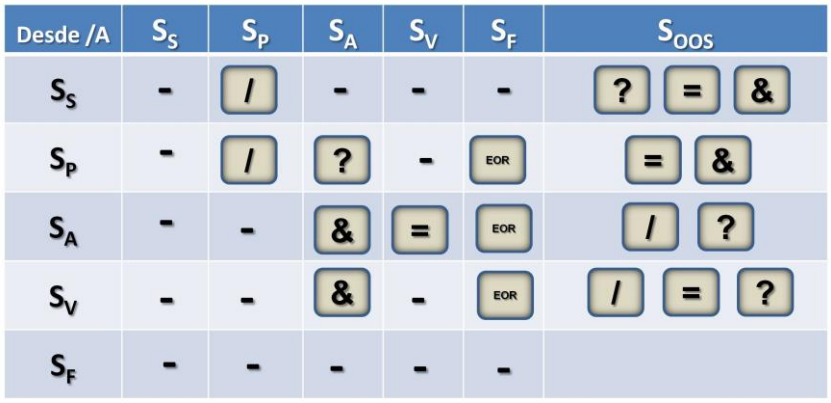


Figura 3.7 Transiciones posibles entre estados en función del delimitador observador en el URI

de solicitud, como POST y HEAD del protocolo HTTP, e incluso extender este tipo de modelado a otros protocolos diferentes que estén basados en el paso de mensajes. En particular, Estévez mostró en [Estevez-Tapiador et al., 2005] la aplicación de este modelado a peticiones DNS.

Una vez identificados los estados y la topología del modelo, esto es, el primero de los procesos estocásticos considerado por el modelado SSM, procederemos a detallar el modelado del segundo de los procesos. Este segundo proceso es el asociado a la producción de símbolos (observaciones) en cada uno de los estados.

Como se ha indicado previamente, dado un URI, es posible establecer dos secuencias emparejadas correspondientes a la secuencia de campos (estados en el modelado SSM) y sus valores, sin más que aplicar el procedimiento de segmentación. La secuencia de campos ya ha sido incluida como el primero de los procesos considerados en el modelado. Los diferentes valores obtenidos en cada uno de los campos constituirán las observaciones a modelar por el autómata, esto es, el segundo de los procesos.

Cada una de las observaciones corresponderá, en el caso de URI de HTTP, a cadenas de caracteres, que serán denominadas palabras en lo que sigue. De esta forma, es posible construir un *vocabulario*, *V*, compuesto por todas las posibles palabras que pueden aparecer en el URI, a partir de la observación de múltiples peticiones a un servidor dado. Idealmente, podría determinarse un vocabulario completo que incluyese todas las posibles palabras si se dispusiese de todas las posibles peticiones que se puedan realizar al servidor.

Adicionalmente, dado que los valores de las palabras dependen o pueden depender del campo asociado, se puede determinar también el campo o campos asociados a cada elemento del vocabulario. Esto es, cada palabra tendrá asociados los campos en los que puede ser observada.

Una vez establecido el vocabulario, a partir de la observación de peticiones dirigidas al servidor a modelar, es posible inferir distribuciones de probabilidad

asociadas a cada palabra para cada estado del modelo, sin más que aplicar las ecuaciones de estimación correspondientes (Ec. (3.11) y (3.17)).

En consecuencia, la técnica SSM es una combinación de dos paradigmas de la detección de intrusos [Estevez-Tapiador et al., 2003]: aprendizaje y especificación. La especificación es utilizada para determinar la topología del modelo, mientras que el aprendizaje es utilizado para estimar tanto el vocabulario como las probabilidades asociadas a las palabras del vocabulario.

Finalmente, el modelo usado por SSM para el proceso de generación/análisis de un URI está formado por [Estevez-Tapiador et al., 2005]:

- Γ: conjunto de estados (S_I, S_S, S_P, S_A, S_V, S_F y S_{OOS}),
- Θ: un conjunto de M símbolos observables,
- ***A***: la matriz de probabilidad de transición entre estados,
- ***B***: la matriz de probabilidad de observación de los símbolos y
- ∏: el vector de probabilidades iniciales, cuyos valores están determinados por la topología del modelo.

De esta forma, dado un modelo, λ, y un URI, *U*, es posible estimar la probabilidad de que dicho URI haya sido producido mediante dicho modelo de acuerdo a (3.9):

$$logP(U|\lambda) = log\left(b_{q_1 o_1}\right) + \sum_{t=1}^{T-1} log\left(a_{q_t q_{t+1}}\right) + \sum_{t=1}^{T-1} log\left(b_{q_{t+1} o_{t+1}}\right) \qquad (3.16)$$

donde hemos tenido en cuenta que las probabilidades iniciales son nulas excepto para S_I y que la secuencia de estados queda determinada de forma unívoca por los delimitadores.

3.3.2 Arquitectura del IDS

El modelo SSM puede ser utilizado como el núcleo de un detector de intrusiones sin más que considerar las probabilidades generadas por el mismo y utilizar un detector de umbral. A continuación se describirán la arquitectura y funcionamiento del detector y algunos problemas de implementación del modelado que es necesario resolver para su aplicación efectiva como detector.

El detector basado en el modelado SSM se compone de los siguientes módulos o bloques (Figura 3.8):

- Un módulo de parametrización, encargado de extraer las URI de los paquetes a analizar y su posterior segmentación. También, en su caso, deberá realizar la cuantización o representación de los diferentes segmentos de acuerdo a los requerimientos del sistema.

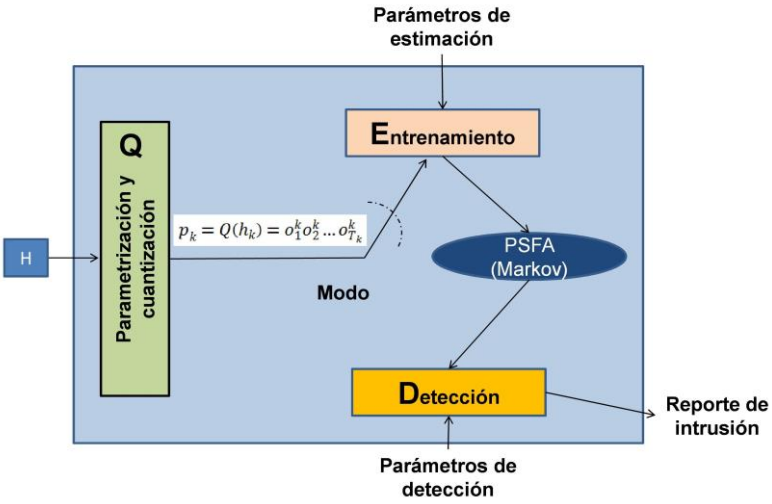


Figura 3.8 Diagrama del detector basado en la técnica SSM

- Un módulo de entrenamiento, encargado de estimar los parámetros del modelo cuando se opere en el modo de entrenamiento.
- Un modelo, que representa el modo de operación normal del sistema.
- Un módulo de evaluación y detección, que obtendrá la probabilidad del URI analizado, de acuerdo al modelo, y determinará si es anómalo o normal (detección).

El procesamiento realizado para una carga útil, p_k, puede ser descrito como sigue. En primer lugar, el módulo de parametrización proporcionará a su salida dos secuencias emparejadas correspondientes a la secuencia de estados y la secuencia de símbolos que componen el URI contenido en dicha carga útil:

$$O^k = G_1\big(U(p_k)\big) = o_1^k o_2^k \dots o_{T_k}^k$$
$$Q^k = G_2\big(U(p_k)\big) = q_1^k q_2^k \dots q_{T_k}^k \qquad (3.17)$$

siendo *U( )* la función que obtiene el URI incluido en la carga útil.

El procesamiento realizado a continuación depende del modo de operación activo. El sistema puede operar en modo de entrenamiento o en modo de detección. En el primer caso, se activará el módulo de entrenamiento, que analizará los URI, una vez parametrizados, a fin de estimar los parámetros del modelo. En este proceso se considerarán las secuencias de entrenamiento y los parámetros de entrenamiento (Figura 3.8), operando en modo *batch*, es decir, se estimará el modelo una vez procesadas todas las secuencias de entrenamiento. Para ello se considerarán las ecuaciones (3.11) detalladas en el Apartado 3.2.2.

En el modo de evaluación, cada URI a la entrada del sistema será procesado individualmente, procediéndose a su parametrización (módulo de parametrización) y la

evaluación de la probabilidad de generación de dicho URI a partir del modelo, tras lo cual se clasificará como normal o anómalo (módulo de detección). El proceso de detección opera en función de algunos parámetros externos al modelo, como puede ser el umbral de detección. Estos parámetros serán descritos con posterioridad.

3.3.3 Evaluación y clasificación de URI

La clasificación de un URI como normal o anómalo se realiza a partir del denominado *índice de anormalidad*, que se define en función de la probabilidad de generación del URI de acuerdo al modelo establecido.

Así, dado un URI, U, y un modelo, λ, se define el *índice de anormalidad* del URI, $N_S(U)$, como

$$N_S(U) = -log(P(U|\lambda) \tag{3.18}$$

siempre que el sistema alcance el estado final, S_F, en la decodificación del URI U. En caso contrario, se le asignará un valor ∞ (correspondiente a considerar $P(U|\lambda) = 0$). Esto es,

$$N_S(U) = \begin{cases} -log(P(U|\lambda) & si\ q_T = S_F \\ \infty & si\ q_T \neq S_F \end{cases} \tag{3.19}$$

El valor del índice de anormalidad será positivo y tanto mayor cuanto menor sea la probabilidad de la secuencia observada. De esta forma, se podrá clasificar un URI como normal o anómalo de acuerdo al *umbral de detección*, θ, como

$$Clase(U) = \begin{cases} Normal & si\ N_S(P) < \theta \\ Anómalo & si\ N_S(P) \geq \theta \end{cases} \tag{3.20}$$

Este umbral de detección puede ser ajustado experimentalmente para seleccionar el punto óptimo de operación del sistema. Mediante este parámetro se puede establecer un compromiso entre el porcentaje de ataques detectados y la tasa de falsos positivos, dado que, como es habitual en la aproximación basada en detección de anomalías, se identificarán como ataques todas las instancias anómalas.

3.3.4 Elementos adicionales del modelado

La utilización del modelado propuesto plantea algunas dificultades prácticas relacionadas con la representatividad y comparabilidad de los valores obtenidos que es necesario abordar. Los problemas más relevantes son el de entrenamiento insuficiente y el de la comparación de secuencias.

Entrenamiento insuficiente

El primero de los problemas consiste en el denominado *problema del entrenamiento insuficiente* [Díaz-Verdejo et al., 2002], ampliamente descrito en la

bibliografía. Está relacionado con la posible aparición durante la operación en modo de evaluación de palabras que no han sido observadas durante el proceso de entrenamiento y, en consecuencia, no han sido incluidas en el vocabulario. Por ello, este problema también se denomina de fuera de vocabulario (OOV, del inglés *Out-of-vocabulary*). La observación de una palabra fuera de vocabulario puede ser debida al hecho de que el conjunto de entrenamiento no incluya dicha palabra, a pesar de que sea legítima, o a que la palabra no sea válida o legítima. En consecuencia, se pueden considerar dos aproximaciones para solucionar el problema:

- eliminar la palabra y asignarle probabilidad cero durante la evaluación y, en consecuencia, hacer nula la probabilidad de la secuencia en la que aparezca, o, alternativamente,
- asignarle una probabilidad fija de valor pequeño.

El segundo método es el denominado *suavizado*, que también puede extenderse a las palabras incluidas en el vocabulario con una probabilidad inferior a una dada. Este caso será descrito a continuación. Obviamente, la elección de una de las dos estrategias dependerá, en general, de la aplicación considerada, esto es, de la existencia de palabras lícitas/ilícitas y del tamaño del conjunto de entrenamiento.

En el caso del modelado de las URI, dado que, a efectos prácticos, es imposible incluir en el vocabulario todas las palabras que pueden ser observadas en las peticiones normales, será necesario optar por la segunda de las estrategias.

En nuestro caso, diferenciaremos dos mecanismos para afrontar el problema del entrenamiento insuficiente, aunque ambos pueden ser descritos de forma unificada.

El primero de ellos, que denominaremos simplemente *suavizado*, consistirá en la asignación de un valor fijo, el *umbral de suavizado*, μ, a todas aquellas probabilidades de observación de valor inferior al umbral de suavizado. Esto es:

$$b'_{ij} = \begin{cases} b_{ij} & si\ b_{ij} \geq \mu \\ \mu & si\ b_{ij} < \mu \end{cases} \tag{3.21}$$

Dado que es necesario preservar la interpretación probabilística, tras la aplicación del umbral de suavizado será necesario renormalizar los valores de la probabilidad de observación por estados.

El segundo de los métodos consistirá en la asignación de una probabilidad fija, la *probabilidad de fuera de vocabulario*, p_{OOV}, o simplemente OOV, a cualquier palabra observada durante el proceso de decodificación y que no se encuentre incluida en el vocabulario.

Ambos procedimientos son complementarios y, por coherencia, se debería utilizar un valor del umbral de suavizado mayor o igual que el de la probabilidad de fuera de vocabulario, esto es:

$$\mu \geq p_{OOV} \tag{3.22}$$

Comparación de secuencias

La comparación directa de las probabilidades de generación asignadas a dos secuencias en el caso de que tengan diferentes longitudes puede resultar inadecuada. Esto es debido a que, dado que se están evaluando probabilidades que se van acumulando en cada elemento de la secuencia, la probabilidad de generación de la secuencia es una función decreciente con su longitud, esto es, con el número de veces que se ha acumulado la probabilidad. En consecuencia, se hace necesario establecer algún mecanismo que compense este decrecimiento y permita comparar secuencias de diferentes longitudes.

La solución adoptada es análoga a la propuesta por Estévez [Estevez-Tapiador, 2004a], en la que se establecen dos factores de compensación asociados a las componentes probabilísticas del modelo: las probabilidades de estado inicial, las probabilidades de transición y las probabilidades de observación, a fin de evaluar una probabilidad normalizada de observación. El factor de compensación utilizado es el valor esperado de la probabilidad en cada una de las transiciones, de forma que se establecen dos factores de compensación:

$$\begin{aligned} \varepsilon_t &= E[A] = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N} a_{ij} \\ \varepsilon_O &= E[B] = \frac{1}{NM}\sum_{i=1}^{N}\sum_{j=1}^{M} b_{ij} \end{aligned} \tag{3.23}$$

correspondientes, respectivamente, a los valores esperados de las probabilidades de transición y de observación.

Usando estos factores de ajuste, la evaluación de la probabilidad normalizada de una secuencia de longitud *T*, esto es, con *T* palabras y *T* transiciones entre estados, resulta [Estevez-Tapiador et al., 2003]:

$$\begin{aligned} log P_{norm}(U|\lambda) \quad &= log\left(\varepsilon_i b_{q_1 o_1}\right) + \sum_{t=1}^{T-1} log\left(\varepsilon_t a_{q_t q_{t+1}}\right) \\ &+ \sum_{t=1}^{T-1} log\left(\varepsilon_O b_{q_{t+1} o_{t+1}}\right) \end{aligned} \tag{3.24}$$

Agrupando términos, la comparación de las diferentes secuencias cuando se consideran sus longitudes puede realizarse sin más que añadir un factor corrector, ד, aditivo en la evaluación de la probabilidad logarítmica de producción, de valor:

$$ד = log\left(\frac{1}{\varepsilon_i \varepsilon_t^{(T-1)} \varepsilon_o^T}\right) = -(log\varepsilon_i + (T-1)log\varepsilon_t + Tlog\varepsilon_o) \tag{3.25}$$

Este factor ד representa el valor esperado logarítmico de la probabilidad en el reconocimiento de una secuencia típica de longitud *T*. De esta forma, tras la

compensación, se obtiene un valor que debe interpretarse como el nivel de adecuación de la secuencia al modelo en relación al obtenido por una secuencia típica de la misma longitud.

A continuación se presentan los resultados experimentales con el uso de la técnica SSM que servirán como resultados de referencia para los próximos capítulos.

3.4 Evaluación experimental

Con objeto de disponer de unos resultados de referencia, se ha implementado y evaluado la técnica SSM, tal como se propone en [Estevez-Tapiador, 2004a], con las bases de datos descritas en el Capítulo 2. Los resultados que se obtengan servirán para evaluar y contrastar las mejoras y modificaciones que se proponen en esta tesis.

DARPA'99

Así, la primera de las bases de datos consideradas es DARPA'99, junto con la base de datos de ataques A. Esta es la misma que fue utilizada por [Estevez-Tapiador, 2004a] y servirá, además de para comparar las mejoras, para comprobar la implementación correcta de la técnica en las mismas condiciones que en el trabajo original.

Las características más relevantes de esta base de datos se detallan en la Tabla 2.6, mientras que en la Tabla 2.4 se muestra la composición de cada una de las particiones establecidas.

Como se ha mencionado, este primer grupo de experimentos pretende obtener un resultado comparable con los obtenidos por los autores de la técnica SSM [Estevez-Tapiador, 2004a] y determinar el mejor valor del parámetro OOV, que debe ser ajustado durante el entrenamiento y evaluación del sistema. Evidentemente, el ajuste debe ser finalmente comprobado mediante la partición de validación, aunque este aspecto no es de interés en este punto de la experimentación.

Para obtener las curvas ROC correspondientes se va modificando el umbral de detección, θ, —Ec. (3.20)— a fin de determinar los diferentes valores de TP y FP para cada valor de dicho umbral.

Tras la aplicación de la técnica *leave-one-out* para las particiones consideradas, se obtienen las curvas ROC globales mostradas en la Figura 3.9, en las que se introduce la notación A / B para indicar las bases de datos normal (A) y de ataques (B) utilizadas en cada experimento. En las ROC obtenidas se puede apreciar la bondad del modelado, alcanzándose tasas del 100% de detección para tasas de FP muy reducidas. De hecho, tal como se puede constatar en dichas figuras, no ha sido posible obtener tasas de detección inferiores al 100% ajustando el valor del umbral de detección, salvo, obviamente, para el punto 0,0 (no mostrado en las ROC).

En la Tabla 3.1 se muestran algunas características del vocabulario, indicándose el número de palabras diferentes observadas en cada segmento del URI (estado). En este caso, se constata que el número de palabras diferentes correspondientes al *path* es

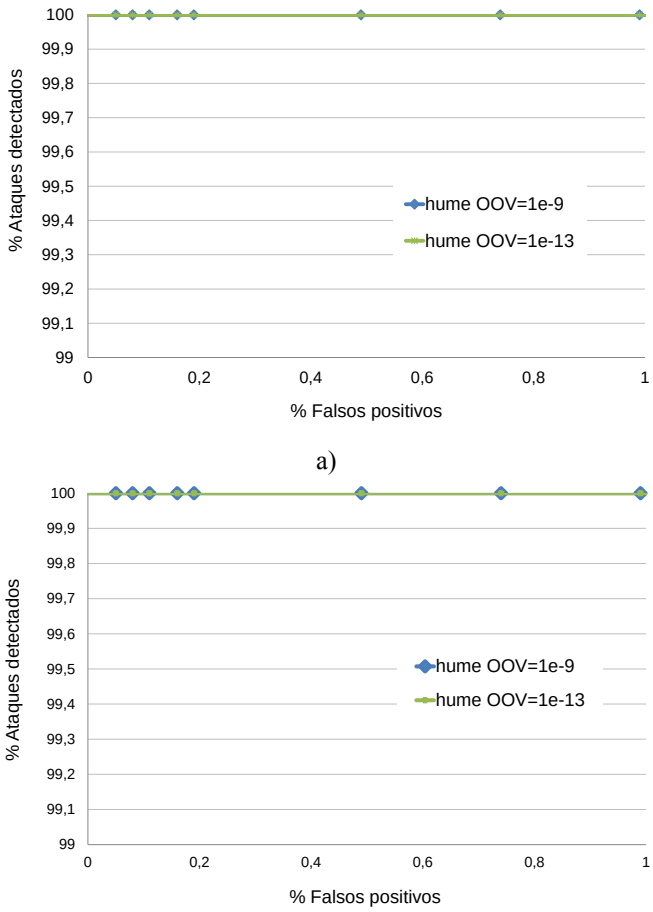


Figura 3.9: Curvas ROC obtenidas para el detector SSM original: a) HUME / A y b) MARX /A para diferentes valores de OOV

claramente mayor que el número de atributos diferentes y que el de los valores correspondientes a dichos atributos. Esta situación no parece, a priori, suficientemente representativa de los sitios web dinámicos actuales, donde sería razonable esperar un elevado número de valores diferentes para los parámetros existentes. En cualquier caso, resulta relevante notar que el vocabulario es global al modelo, no tratándose de forma diferenciada para cada uno de dichos estados. Consecuentemente, el modelo, además de la matriz de transiciones, **A**, incorpora un vocabulario, *V*, de tamaño 348, junto con una matriz de probabilidades de observación, **B**, de dimensión 348x3.

A la vista de los resultados obtenidos, se concluye la adecuación de la implementación realizada y se elige el valor 10^{-9} para la probabilidad de fuera de vocabulario, OOV, como referente para la experimentación subsiguiente con esta base de datos.

	HUME			MARX		
Estado	L1	L2	L3	L1	L2	L3
S_P	342	346	345	351	352	343
S_A	1	1	1	1	1	1
S_V	1	1	1	1	1	1
TOTAL	344	348	347	353	354	345

Tabla 3.1: Tamaño del vocabulario para cada estado en las particiones de las bases de datos HUME y MARX

	CERES12		CERES13		CERES23	
Estado	N_t	N	N_t	N	N_t	N
S_P	239.546	3.370	238.193	3.335	242.231	3.331
S_A	4.780	60	5.203	71	1.601	74
S_V	4.773	423	5.187	450	1.588	224
TOTAL	247.999	3.853	248.573	3.856	245.420	3.529

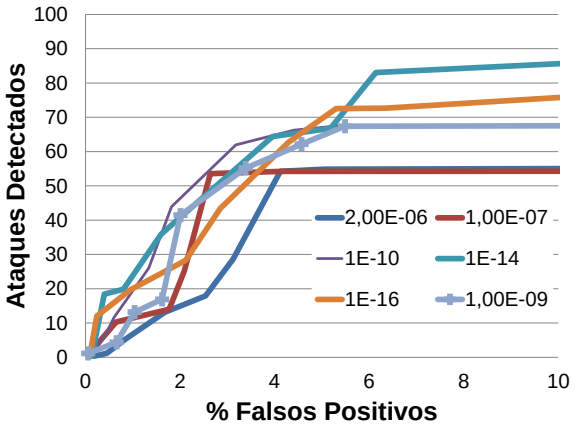
Tabla 3.2: Número total de palabras (N_t) y tamaño del vocabulario (N) para cada estado y globales en las diferentes particiones consideradas en la base de datos CERES

CERES

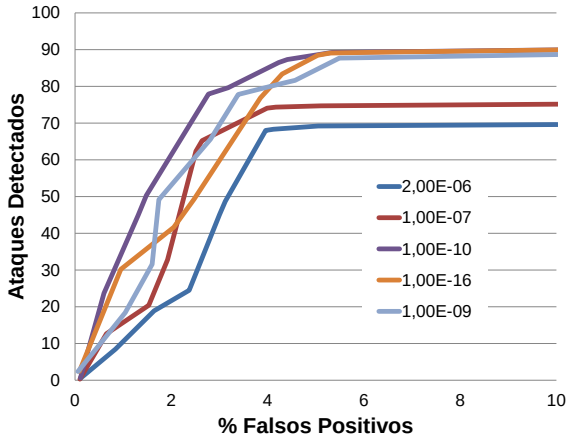
La segunda base de datos considerada es CERES, junto con las bases de datos de ataques RDB y OSVDB.

Las características más relevantes de esta base datos se detallan en la Tabla 2.6. Esta presenta un mayor vocabulario que DARPA y, adicionalmente, incluye un mayor número de elementos de tipo *consulta*, con sus correspondientes parejas *atributo-valor* (Tabla 3.2). En este sentido, ni el número de atributos diferentes ni el número de valores observados son muy elevados, lo que es indicativo de que la dinamicidad del sitio web es limitada.

Los resultados globales obtenidos para el ajuste del valor OOV más representativos, aplicando la metodología *leave-one-out*, se muestran en la Figura 3.10. En este caso se observa un rendimiento inferior al caso de DARPA'99, ya que, aunque se pueden alcanzar tasas de detección altas (en torno al 90%) con un reducido número de FP (en torno al 5%), el sistema es claramente mejorable en prestaciones. Por otra parte, se observa una aparente saturación en la tasa de detección en la zona de tasas de FP inferiores al 10%. Adicionalmente, el comportamiento para diferentes valores de OOV resulta inconsistente, lo que motiva un análisis detallado de la matriz de probabilidades de observación a fin de determinar el origen de este comportamiento. Se constata así la existencia de un elevado número de posiciones cuyo valor es OOV, de acuerdo al procedimiento de suavizado, lo que se identifica como el potencial causante del problema. Aunque esta cuestión se discutirá con más detalle en el Capítulo 4, ya que estará relacionada con la primera propuesta de mejora, parece evidente que la existencia de un elevado número de valores fijados artificiosamente al valor OOV para evitar el



a)



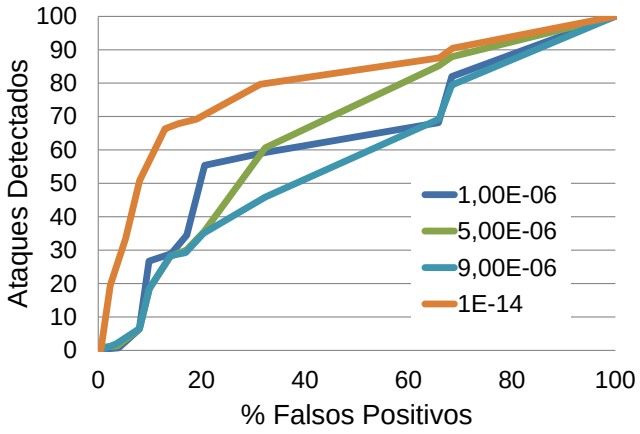
b)

Figura 3.10: Curvas ROC obtenidas para el detector SSM original para: a) CERES / RDB y b) CERES /OSVDB para diferentes valores de OOV

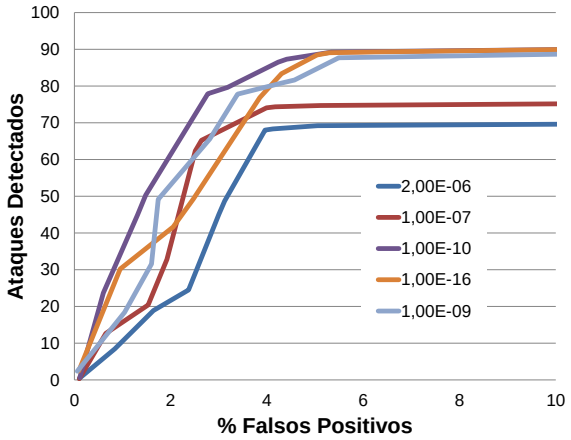
problema de entrenamiento insuficiente genera un efecto distorsionador sobre las probabilidades estimadas en el entrenamiento. Obviamente, este efecto será mayor cuanto mayor sea el valor OOV establecido y el número de palabras del vocabulario a las que afecta.

PVHDB

Para confirmar las observaciones anteriores, se procede a repetir la experimentación con PVHDB y las dos bases de datos de ataque. Las características de PVHDB se muestran también en la Tabla 2.6, mientras que los detalles sobre los tamaños del vocabulario se indican en la Tabla 3.3. En este caso, el tamaño del vocabulario es un orden de magnitud superior al de CERES, existiendo un elevado número de valores de parámetros.



a)



b)

Figura 3.11: Curvas ROC obtenidas para el detector SSM original para: a) PVHDB / RDB y b) PVHDB /OSVDB para diferentes valores de OOV

	PVHDB12		PVHDB13		PVHDB23	
Estado	N_t	N	N_t	N	N_t	N
S_P	851.899	1.033	846.973	1.053	850.962	1.033
S_A	506.867	57	446.118	55	313.753	47
S_V	506.796	12.017	446.061	12.422	313.665	14.011

Tabla 3.3: Número total de palabras (N_t) y tamaño del vocabulario (N) para cada estado en las diferentes particiones consideradas en la base de datos PVHDB

Los resultados más representativos obtenidos al intentar ajustar el valor de OOV se muestran en la Figura 3.11. De nuevo se observa un comportamiento inconsistente con el valor de OOV y una degradación del rendimiento, que ahora resulta claramente insuficiente. De acuerdo a la mejor ROC obtenida, para alcanzar valores en torno al 80% de TP se necesita una tasa de en torno al 30% de FP, lo que es inasumible.

Globalmente, la experimentación realizada utilizando el sistema SSM original muestra un excelente comportamiento en el escenario con un vocabulario reducido. Este se va degradando a medida que se aumenta el tamaño del vocabulario, lo que evidencia limitaciones de dicho sistema que es necesario resolver.

Por tanto, a la vista de los problemas surgidos, de la experimentación realizada únicamente se considerarán los resultados para DARPA'99 como de referencia para desarrollos posteriores, siendo necesario establecer unos resultados de referencia adecuados para las otras dos bases de datos (CERES y PVHDB), lo que se aborda como primera propuesta de mejora en el Capítulo 4.

4 Mejoras al sistema de referencia

En el capítulo anterior se ha contrastado la eficacia en la detección de la técnica SSM, evaluándose los resultados experimentales obtenidos. Así, la combinación de los modelos de Markov y la especificación del protocolo correspondiente al servicio objetivo consiguen unas prestaciones altamente aceptables, tanto en términos de capacidad de detección como en tiempo de procesamiento, en escenarios con vocabularios reducidos. En este sentido, aunque la experimentación llevada a cabo para el sistema ha sido *off-line*, recientemente se ha realizado su despliegue y puesta en marcha en entornos reales.

No obstante lo anterior, no debemos obviar que el sistema de detección propuesto presenta ciertas limitaciones. Éstas serán discutidas a lo largo del Apartado 4.1 del presente capítulo. Así, a partir del sistema original, se plantearán y analizarán algunas posibles mejoras con objeto de conseguir incrementar el rendimiento general del sistema desarrollado. La primera está relacionada con aspectos de implementación en el caso de grandes vocabularios. Como se ha comentado en el capítulo anterior, el suavizado necesario de la matriz de probabilidades de observación puede plantear algunos inconvenientes en el caso de grandes vocabularios, por lo que se propondrá y evaluará un tratamiento de dicha matriz en base a vectores correspondientes a las columnas (estados), que será descrito y analizado en el Apartado 4.2. Este tratamiento consigue una reducción de la dimensionalidad de los vectores que es necesario utilizar, a la vez que elimina el problema del suavizado de los valores de dicha matriz, al obviar los valores nulos. Persiste, sin embargo, la necesidad de realizar un suavizado en base a la utilización de la probabilidad de observaciones fuera de vocabulario (OOV). La segunda propuesta de mejora se refiere a la inclusión efectiva de las probabilidades de transición en el modelado de Markov del SSM, de acuerdo a la teoría subyacente. Tanto la discusión sobre este aspecto como los resultados experimentales derivados se desarrollarán en el Apartado 4.3.

Con posterioridad, en los Apartados 4.3 y 4.4 se evaluarán mejoras al esquema usualmente considerado para el suavizado de las probabilidades de observación. En primer lugar, en base a la propuesta de una estimación más racional (a priori) del parámetro de suavizado. Posteriormente, en lo que constituye una propuesta de mayor calado, a través de la sustitución del esquema usual de suavizado para observaciones

fuera de vocabulario por un procedimiento en el que se hace un tratamiento dependiente del estado para los casos OOV.

4.1 Análisis del sistema SSM: propuesta de mejoras

En la técnica SSM se establece un modelo de Markov tomando como observaciones las cargas útiles de los paquetes correspondientes al servicio objetivo; en nuestro caso, HTTP. A partir de dichas cargas útiles, se implementa el siguiente procedimiento general para la obtención del modelo de normalidad correspondiente:

1) Se define un autómata de estados finitos general derivado de la especificación del protocolo en cuestión, donde cada estado corresponde a una parte específica de los URI del servicio.
2) Fijado el autómata, se asigna una probabilidad igual a 1 para todas las transiciones entre estados contempladas en las especificaciones del protocolo/servicio. Es decir:

$$a_{ij} = \begin{cases} 1 & si\ la\ transición\ de\ S_i\ a\ S_j\ es\ posible \\ 0 & en\ otro\ caso \end{cases} \quad (4.1)$$

3) Se segmenta cada uno de los URI disponibles en el conjunto de datos de entrenamiento de acuerdo a las reglas que rigen el autómata general, obteniéndose el conjunto de observaciones, símbolos o palabras, que define el "diccionario" o "vocabulario" del entorno considerado.
4) Sobre este grupo de observaciones, o diccionario, y para el conjunto de entrenamiento, se estima la probabilidad de aparición de cada una de las palabras en cada uno de los estados del modelo (matriz **B**).
5) Como resultado se obtendrá un modelo de normalidad para el protocolo/servicio HTTP. Dicho modelo quedará definido por un conjunto de estados (S), las transiciones entre ellos (matriz **A**) y las probabilidades de observación de las palabras de un diccionario, Θ, en cada uno de los estados del modelo (matriz **B**).

Si bien la aproximación del modelado de Markov es fácilmente comprensible desde un punto de vista teórico, existen algunos problemas prácticos en la implementación realizada y que son habituales en este contexto.

Un primer aspecto de relevancia en el modelado considerado se refiere a la matriz de observaciones **B**. Como se ha discutido a lo largo del Capítulo 3, el cálculo y uso de estas probabilidades debe hacerse en base a dos procedimientos complementarios:

1) **Suavizado de las probabilidades**: Existe la posibilidad real de que una palabra dada del diccionario no aparezca asociada a uno o más estados del

modelo en el conjunto de entrenamiento. Esto implicará una probabilidad de observación nula, lo que se traducirá en la imposibilidad de permitir en detección dicha observación.

Para solucionar este problema se trata de evitar la situación de "suceso imposible" mediante la utilización de valores de probabilidad de observación mínimos distintos de cero en estos casos.

2) **Probabilidad de fuera de vocabulario** (OOV, "*Out Of Vocabulary*"): Adicionalmente al hecho de que una palabra del diccionario pueda no haberse observado en un estado del modelo en la fase de entrenamiento, es bastante probable que en detección se observe una palabra no incluida en el diccionario obtenido en entrenamiento.

Como antes, para evitar la asignación de una probabilidad nula a una observación de estas características en detección, lo que equivaldría de nuevo a un "suceso imposible", se suele recurrir también al uso de un término de probabilidad pequeño distinto de cero.

Ambas situaciones son, como se ha indicado, complementarias y están derivadas del problema conocido como *entrenamiento insuficiente*. Es decir, el conjunto de datos usado para entrenar el sistema no es suficientemente representativo del entorno que se desea modelar. Evidentemente, se puede tratar de reducir este problema aumentando el tamaño del conjunto de entrenamiento, pero de esta forma tampoco se garantiza que se evite el problema. Consecuentemente, ambas técnicas, suavizado y OOV, deben ser tenidas en cuenta en todo sistema real.

Sin embargo, como se ha evidenciado en el Capítulo 3, la consideración del suavizado de las probabilidades de observación plantea algunos problemas en el caso de grandes vocabularios, es decir, de una alta dimensionalidad, debido a un elevado número de filas de la matriz **B**. En estos casos, es posible la existencia de un alto número de valores a cero en dicha matriz por no haberse observado la palabra correspondiente a la fila en algunos de los estados. Esta situación es especialmente relevante para sitios web dinámicos en los que se usen variables y el número de valores posibles para las mismas sea elevado.

Teniendo en cuenta estos aspectos, a lo largo del Apartado 4.2 se propone una mejora basada en el tratamiento diferenciado por estados de la matriz **B**. De acuerdo a la propuesta, se consideran vectores de probabilidades independientes para cada estado, eliminándose en consecuencia las probabilidades correspondientes a palabras no observadas durante el entrenamiento, es decir, obviando los valores nulos de la matriz original. De esta forma, se hace innecesario el suavizado de las probabilidades de observación, aunque no el uso de la probabilidad OOV. Adicionalmente, se realiza una propuesta alternativa a la usualmente considerada en la bibliografía y en el sistema de referencia para la estimación de un posible valor de suavizado de los vectores resultantes.

La segunda propuesta de mejora se refiere al hecho de que, como se ha indicado en la Ec. (4.1), la matriz de transición, **A,** asociada al modelo de Markov obtenido, no satisface la condición impuesta en la teoría correspondiente en relación a que la suma de las probabilidades de transición desde un estado dado a cualquier otro deber ser 1, es decir, $\sum_{j=1}^{N} a_{ij} = 1$. El principal inconveniente de este incumplimiento es que aquellas partes del autómata que presenten una menor ramificación se verán "perjudicadas", desde el punto de vista probabilístico, frente a las de mayor ramificación en la fase de detección. En otras palabras, la constatación de una transición entre dos estados dados debe proporcionar más información cuanto mayor sea el número de transiciones posibles desde el estado origen en cuestión.

La no consideración de este aspecto en la implementación actual de SSM aconseja estudiar el comportamiento del sistema ante la estimación e inclusión efectiva de una matriz A "más realista" en el modelado IDS. Esta cuestión será abordada en el Apartado 4.3.

Finalmente, también relacionado con el modelado de las probabilidades de observación, en el Apartado 4.4 del presente capítulo se plantea otra posible mejora al sistema de referencia consistente en la consideración de una probabilidad OOV dependiente del estado.

Cada una de las posibles mejoras al sistema de referencia anteriormente expuestas en relación a las matrices **A** y **B** se analiza en mayor detalle a continuación. Para la evaluación experimental de las mismas se han considerado las bases de datos DARPA, PVHDB y CERES, además de RDB y OSVDB como bases de datos de ataques. Dichas bases de datos se describen en el Capítulo 2. Sobre todas ellas se han tomado, por un lado, particiones de entrenamiento para la obtención del modelo de normalidad correspondiente y, por otro, de test para la obtención y el subsiguiente análisis de los resultados de detección, a partir de la unión de varias de ellas de acuerdo a un procedimiento *leave-one-out*.

4.2 Implementación para grandes vocabularios

La posible aparición de términos nulos en la matriz de observaciones, **B**, previamente comentada, genera problemas de índole práctica relacionados con la aplicación de la técnica de suavizado. Estos problemas, aunque presentes en todos los casos, adquieren una dimensión relevante en el caso de que el tamaño del vocabulario sea grande. Para ilustrar la problemática asociada consideraremos un caso práctico, aunque de vocabulario reducido, como ejemplo. Así, en la Tabla 4.1 se muestra el vocabulario asociado a un posible modelo estimado durante la fase de entrenamiento sobre un subconjunto reducido de URI tomados de PVHDB (véase el ejemplo del Apartado 3.3). En este caso, se considera un vocabulario compuesto por 11 palabras cuya frecuencia de aparición en el conjunto de entrenamiento se indica también en la tabla. Consecuentemente, las probabilidades de observación para cada uno de los estados del autómata son las indicadas, evidenciándose la existencia de varios términos a valor 0.

Palabra	Estado *n.obs./probabilidad*		
	S_P	S_A	S_V
tareas	2/0,286	0/0	0/0
feb	2/0,286	0/0	0/0
descri.html	1/0,143	0/0	0/0
unidad1	0/0	1/0,333	0/0
tres	0/0	0/0	1/0,333
observa	1/0,143	0/0	0/0
maestro.php	1/0,143	0/0	0/0
mat	0/0	1/0,333	0/0
17	0/0	0/0	1/0,333
alumno	0/0	1/0,333	0/0
11	0/0	0/0	1/0,333

Tabla 4.1: Ejemplo de diccionario, frecuencias de aparición y probabilidades de observación asociadas

Una solución habitual para solventar el problema de los elementos nulos de la matriz **B**, que es la considerada en el sistema de referencia, es realizar un suavizado de las mismas de acuerdo al procedimiento siguiente. Para cada probabilidad de observación de un símbolo i en un estado j, b_{ij}, se obtiene un valor suavizado de la probabilidad, b'_{ij}, de acuerdo a:

$$b'_{ij} = \begin{cases} b_{ij} - \dfrac{z_j}{n_j}.\varepsilon & si\ b_{ij} > \varepsilon \\ \varepsilon & si\ b_{ij} \leq \varepsilon \end{cases} \qquad (4.2)$$

donde ε es el valor de suavizado (mayor que cero), z_j el número de palabras con probabilidad igual a cero o inferior a ε para el estado j, y n_j el número de palabras con probabilidad mayor que ε en dicho estado. En otras palabras, a los símbolos no observados en las secuencias de entrenamiento se les asigna un valor ε pequeño (p.e. 10^{-3}), ajustándose proporcionalmente el resto de probabilidades para que la suma global por estado se mantenga normalizada y de valor mínimo ε, según $\sum_{i=1}^{M} b'_{ij} = \sum_{i=1}^{M} b'_{ij} = 1$.

Así, suponiendo un valor del suavizado, ε, igual a 10^{-3}, para las observaciones de la Tabla 4.1, se obtendrían los valores mostrados en la Tabla 4.2.

En este contexto, resulta relevante que el esquema de suavizado implementado presenta como característica principal el hecho de que el parámetro ε debe determinarse de forma experimental e independiente de las observaciones utilizadas en la fase de entrenamiento.

Al margen de cómo se defina el parámetro ε, el esquema de suavizado utilizado es de implementación simple y efectiva ante el problema de entrenamiento insuficiente.

Palabra	Prob. de observación		
	S_P	S_A	S_V
tareas	0,286	0,001	0,001
feb	0,286	0,001	0,001
descri.html	0,143	0,001	0.001
unidad1	0,001	0,333	0,001
tres	0,001	0,001	0,333
observa	0,143	0,001	0,001
maestro.php	0,143	0,001	0,001
mat	0,001	0,333	0,001
17	0,001	0,001	0,333
alumno	0,001	0,333	0,001
11	0,001	0,001	0,333

Tabla 4.2: Probabilidades de observación suavizadas para cada estado

Sin embargo, este suavizado de **B** tiene una serie de implicaciones que deben ser consideradas:

1) Dado que el vocabulario manejado es global, resulta previsible que, a medida que crezca el tamaño de éste, más elevado será el número de palabras a las que habrá que aplicar en cada estado el umbral ε. Esto, evidentemente, afectará también en mayor medida a las palabras que superen dicho umbral de observación.

   En consecuencia, el proceso de suavizado, si bien de aplicación necesaria, lleva implícita una afectación aleatoria en la matriz de observaciones del modelo. Este efecto puede llegar a ser importante especialmente en el caso de grandes vocabularios.

2) Adicionalmente, es posible la aparición en detección de palabras no incluidas en el vocabulario global (observaciones *fuera de vocabulario,* OOV), por lo que la utilización del suavizado resulta insuficiente por sí sola para resolver el problema del entrenamiento insuficiente. En este caso, la probabilidad de observación asociada sería nula, lo que nos lleva de nuevo al problema de partida respecto del entrenamiento insuficiente.

   Como ya se ha mencionado, esta situación se resuelve habitualmente mediante de la incorporación ad-hoc, totalmente artificial y al margen de la matriz **B**, de una probabilidad de observación muy pequeña. Sin embargo, es evidente que este valor OOV debe determinarse de forma experimental e independiente de cualquier otra consideración relacionada con el entrenamiento del sistema.

Así, el sistema SSM ha sido implementado, siguiendo el procedimiento habitual para los autómatas de estados finitos, para que, a partir de los segmentos de los URI utilizados en entrenamiento, se genere un vocabulario único compuesto por todas las palabras observadas. De esta forma, será necesario estimar una matriz **B** de tamaño *MxN*, siendo *M* el número de palabras en el vocabulario y *N* el de estados del autómata. En el caso de grandes vocabularios, no sólo esta matriz tendrá una elevada dimensionalidad, sino que adicionalmente es bastante probable que exista un alto número de valores nulos. Esto es debido a que una proporción importante de las palabras aparecerán únicamente en uno de los estados. Es más, en escenarios como los sitios web dinámicos en los que se consideren diferentes atributos, el número de palabras diferentes asociadas a cada uno de los estados del modelo presentará una gran variabilidad. A modo de ejemplo, y considerando las bases de datos utilizadas en la experimentación (Capítulo 2), podemos observar que en el estado S_P, el vocabulario asociado incluye alrededor de un centenar de palabras, mientras que en el caso del estado S_A existen decenas de miles de ellas. Consecuentemente, se obtendrán decenas de miles de valores nulos para la probabilidad de observación en el estado S_A. El procedimiento de suavizado modificará estos valores para que tomen el valor ε establecido. En este escenario, incluso si se considera un valor pequeño para ε, por ejemplo, $\varepsilon=10^{-5}$, se obtendrá un factor corrector para los valores no nulos de las probabilidades de alrededor de 0,1. El resultado será que las probabilidades correspondientes a las palabras realmente observadas en el estado S_A serán modificadas significativamente. La aproximación más obvia en este caso sería utilizar valores aún más pequeños para el factor de suavizado, lo que invalidaría parcialmente el efecto deseado de paliar el entrenamiento insuficiente. Adicionalmente, a partir del ejemplo mostrado puede observarse que existirán limitaciones en los valores posibles del factor de suavizado a fin de mantener la condición de normalización de las probabilidades.

Para solucionar este problema se propone la utilización de un vocabulario diferenciado por cada estado del autómata. Es decir, un vocabulario para el estado de segmento de *path* (S_P), otro para el estado de atributo (S_A) y otro más para el estado de valor (S_V). Así se podrán tratar de forma diferenciada las observaciones correspondientes a cada segmento a ser evaluado y tener un mejor control de cada uno de los vocabularios por cada estado. Las consecuencias derivadas de esta aproximación se analizarán más adelante cuando se detalle el procedimiento propuesto, si bien resulta relevante mencionar que, de esta forma, se eliminan los valores nulos para las probabilidades de observación, lo que hace innecesario el suavizado de las mismas.

A partir de las cuestiones previas, en este apartado se propone una modificación de la técnica para resolver el problema de entrenamiento insuficiente en lo que respecta a la matriz de observaciones, **B**, que resulta especialmente útil en el caso de grandes vocabularios. Dicha propuesta se basa en dos modificaciones complementarias:

1) Se trata de forma independiente cada columna de la matriz **B**, es decir, cada estado, a efectos de determinar el vocabulario asociado. De esta forma, no se realiza suavizado alguno de la matriz **B**, ya que ésta se limitará a

Diccionario	
Índice	Palabra
0	tareas
1	feb
2	descri.html
3	unidad1
4	tres
5	observa
6	maestro.php
7	mat
8	17
9	alumno
10	11

S_P		S_A		S_V	
Índice	B_P	Índice	B_A	Índice	B_V
0	0,286	3	0,333	4	0,333
1	0,286	7	0,333	8	0,333
2	0,143	9	0,333	10	0,333
5	0,143				
6	0,143				

Tabla 4.3: Diccionario global y probabilidades de observación por estado, donde se obvian las palabras no observadas en entrenamiento para cada estado

incorporar aquellas palabras cuya probabilidad de observación en entrenamiento sea distinta de cero. En otras palabras, los símbolos no observados no formarán parte de la matriz **B**.

En consecuencia, la matriz **B** será considerada a partir de vectores asociados a las columnas. Además de evitar así posibles problemas por la modificación "artificial" de B, el hecho de que el tamaño de dichos vectores sea más reducido permitirá una mayor optimización en los recursos implicados, tanto desde el punto de vista computacional como desde el de almacenamiento. El único coste añadido es que, en el proceso de búsqueda del índice de una palabra, habrá que tener en cuenta el estado en el que se encuentra el modelo. De esta forma, habrá un diccionario por estado, en lugar de uno global.

2) La posible observación en detección de palabras no aparecidas en entrenamiento, se encuentren éstas en el vocabulario global o no, se tratará como situación OOV.

Siguiendo con el ejemplo considerado, como se mostró en la Tabla 4.1, son varias las palabras no observadas en entrenamiento para cada uno de los estados del modelo. De esta forma era necesario suavizar la matriz **B** de modo similar al indicado en la Tabla 4.2. Sin embargo, en la aproximación de mejora se propone que, en lugar de disponer una matriz de dimensiones $M \times N$, se utilice un vector de observación distinto por estado, $\boldsymbol{B_i}$, $i \in \{1,\ldots,N\}$, cada uno de ellos de dimensión menor o igual que M en función del número de palabras observadas en entrenamiento en cada caso.

De este modo, para el caso planteado como ejemplo, **B** resultará como se indica en la Tabla 4.3, donde se observa que aquellas palabras no aparecidas en entrenamiento no se consideran en el vector correspondiente al estado en cuestión. Así, el vector **B**

	CERES12			CERES13			CERES23		
Estado	N_t	N	P_{min}	N_t	N	P_{min}	N_t	N	P_{min}
S_P	239.546	3.370	4,17e-6	238.193	3.335	4,2e-6	242.231	3.331	4,13e-6
S_A	4.780	60	0,000209	5.203	71	0,000192	1.601	74	0,000625
S_V	4.773	423	0,00021	5.187	450	0,000193	1.588	224	0,00063

Tabla 4.4: Número total de palabras (N_t), tamaño del vocabulario (N) y probabilidad mínima (P_{min}) para cada estado en las diferentes particiones consideradas en la base de datos CERES

asociado al estado S_P, $\mathbf{B_P}$, es sólo de longitud 5, frente al original de 11; mientras que los de los estados S_A y S_V, $\mathbf{B_A}$ y $\mathbf{B_V}$, respectivamente, son sólo de tamaño 3, frente, de nuevo, al original de 11. Dada la variabilidad en el tamaño de **B** para cada estado, será preciso identificar la palabra del diccionario a la que corresponde cada entrada en el vector. La reducción así conseguida en el tamaño de **B** supondrá, adicionalmente, un decremento proporcional en el almacenamiento y los tiempos de cómputo asociados en detección.

Una vez expuesta la aproximación planteada como posible mejora al sistema de referencia presentado en el Capítulo 3, a continuación se presentan y discuten los resultados experimentales obtenidos a partir de la implementación de la misma. En este estudio se han realizado diferentes experimentos variando los valores del parámetro de fuera de vocabulario para cada estado entre 10^{-20} y 10^{-3}. Seguidamente se muestran y analizan las curvas ROC obtenidas más significativas.

4.2.1 Resultados experimentales

En la experimentación realizada se consideran las bases de datos CERES y PVHDB (véase Capítulo 2), ya que DARPA no se veía afectada por este problema.

Evaluación con la base de datos CERES

El primer escenario considerado corresponde a CERES. Las características más relevantes de esta base datos se detallan en la Tabla 2.6. Como se puede observar, esta base de datos presenta un mayor vocabulario que DARPA, que es la utilizada como referencia (Capítulo 3), y, adicionalmente, incluye un número adecuado de elementos de tipo *consulta*, con sus correspondientes parejas *atributo-valor*.

En la Tabla 4.4 se muestra el número de palabras y las probabilidades mínimas obtenidas en el modelado en cada uno de los estados, para las diferentes particiones consideradas. Es reseñable que el estado con mayor número de palabras es el estado de segmento de path, S_P. Esto se debe a que en el servidor donde se obtuvieron las trazas la mayoría de los recursos eran páginas web estáticas.

Como se ha mencionado en el Capítulo 3, fue durante la experimentación con esta base de datos cuando se constató el efecto distorsionador del suavizado en el caso de grandes vocabularios, dando lugar a la propuesta de modificación que está siendo

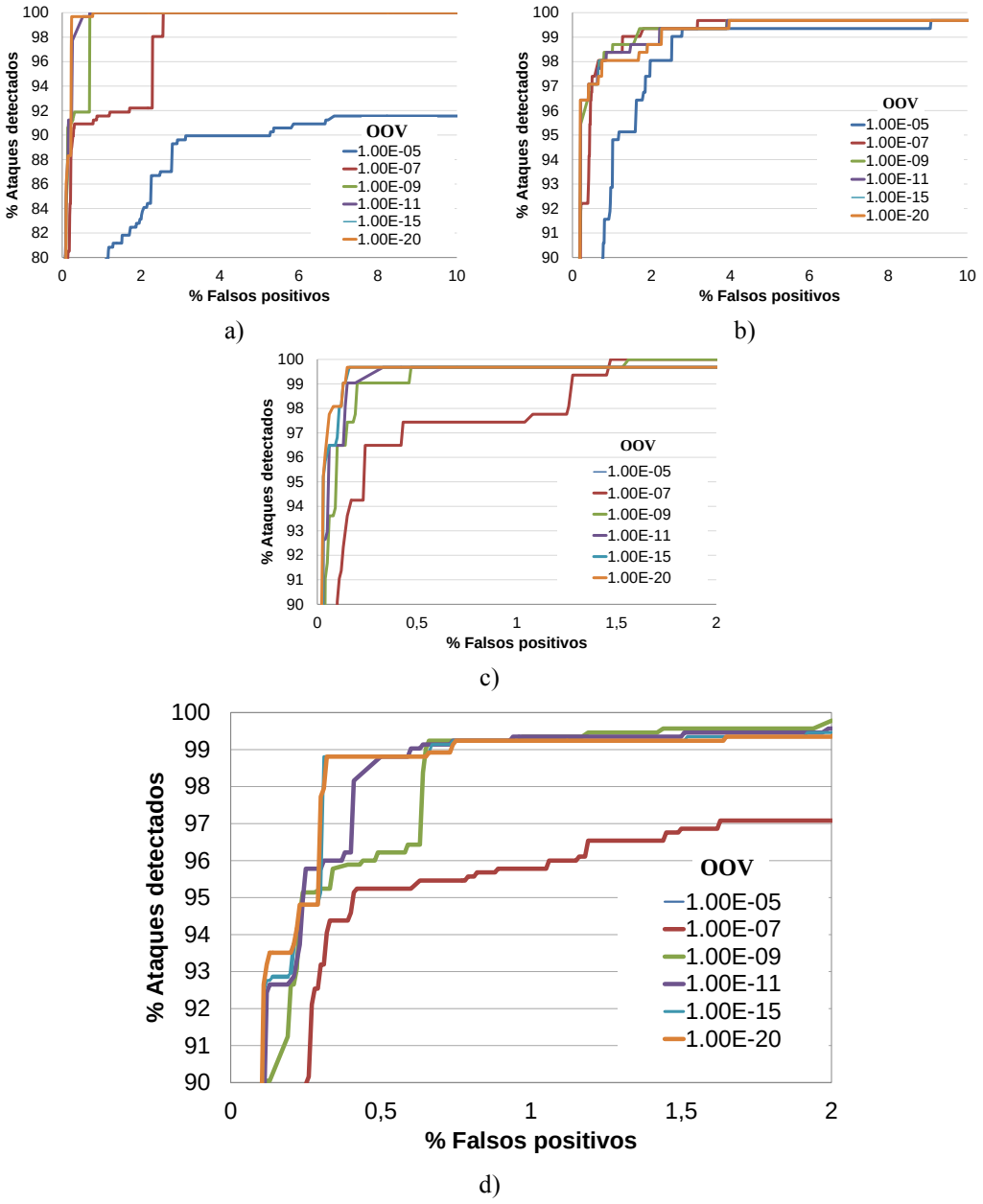


Figura 4.1: Curvas ROC para CERES / RDB para diferentes valores de OOV, para las particiones: a) CERES23, b) CERES13, c) CERES12 y d) resultados globales

evaluada. En este sentido, la comparación de los resultados obtenidos en la implementación original y la actual servirán para constatar la mejora introducida con la consideración de vectores diferenciados por estado para la matriz B.

La experimentación con esta base de datos, una vez obtenidos los modelos, se ha realizado para las dos bases de datos de ataque disponibles. Los parámetros a ajustar

OOV	% DET	% FP
$1e^{-5}$	81,08	1,04
$1e^{-7}$	95,78	1,05
$1e^{-9}$	99,35	1,00
$1e^{-11}$	99,35	1,00
$1e^{-15}$	99,24	1,02
$1e^{-20}$	99,24	1,00

Tabla 4.5: Resultados globales para CERES / RDB

experimentalmente son la probabilidad OOV y el umbral de detección. En consecuencia, se han evaluado valores de OOV en el rango 10^{-5} a 10^{-20}, mostrándose a continuación las correspondientes curvas ROC obtenidas mediante la variación del umbral de detección. En primer lugar se consideró la base de datos RDB, obteniéndose los resultados mostrados en la Figura 4.1, donde se utiliza la notación CERES / RDB para especificar las bases de datos normal y de ataques utilizadas. Esta notación será utilizada en lo sucesivo.

En la Tabla 4.5 se detallan los valores numéricos obtenidos para la tasa de detección cuando se fija una tasa de falsos positivos en torno al 1%. Se puede observar que el mejor comportamiento se consigue para valores del parámetro de fuera de vocabulario en torno a 10^{-9}, degradándose ligeramente para valores menores. Este efecto se debe a que valores más pequeños de OOV implican que las palabras no observadas durante el entrenamiento serán fuertemente penalizadas, por lo que la tasa de falsos positivos debería aumentar al disminuir OOV. Sin embargo, es de esperar que en los ataques se incluyan palabras no permitidas o no observadas durante el entrenamiento, por lo que, idealmente, el valor de OOV debería ser bajo para facilitar su detección. En consecuencia, dado que el conjunto de evaluación incluye nuevos elementos en el vocabulario en relación al usado en el modelo, es necesario encontrar un compromiso para dicho valor.

A continuación se procedió a evaluar la base de datos con los ataques en OSVDB, obteniéndose los resultados mostrados en la Figura 4.2.

El análisis de las curvas obtenidas muestra que valores de OOV en torno a 10^{-9} seleccionados previamente resultan adecuados. En este caso se alcanza un 95,78% de ataques detectados con sólo un 0,4% de falsos positivos. Aunque valores de fuera de vocabulario más pequeños (10^{-15} y 10^{-20}) presentan una ligera mejoría, ésta se considera irrelevante y presenta el inconveniente de incrementar la especificidad del modelado ante la observación de nuevas palabras.

Evaluación con la base de datos PVHDB

A continuación se ha evaluado la técnica SSM incorporando la mejora propuesta con la base de datos PVHDB, igualmente descrita en el Capítulo 2, y cuyas características más relevantes se detallan en la Tabla 2.5.

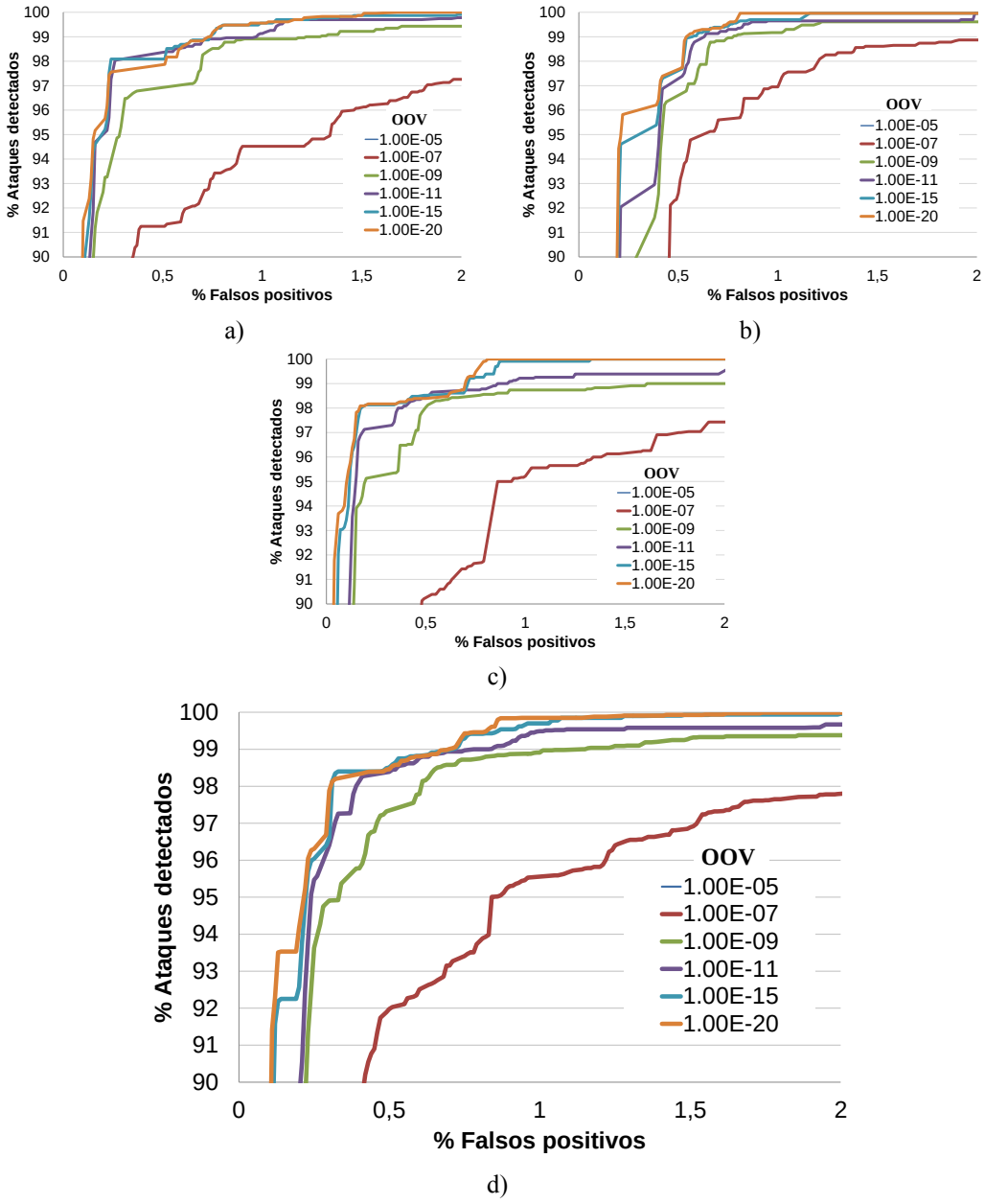


Figura 4.2: Curvas ROC para CERES / OSVDB para diferentes valores de OOV, para las particiones: a) CERES23. b) CERES13, c) CERES12 y d) resultados globales

En la Tabla 4.6 se muestra el número de palabras y la probabilidad mínima obtenida en el modelado en cada uno de los estados, para las diferentes particiones consideradas. Es reseñable que el estado con mayor número de palabras, a diferencia de la base de datos CERES, es el estado de valor, S_v. Esta situación es la más habitual en la actualidad, al considerarse páginas web dinámicas y la consiguiente utilización de variables para su generación.

	PVHDB12			PVHDB13			PVHDB23		
Est.	N_t	N	P_{min}	N_t	N	P_{min}	N_t	N	P_{min}
S_P	851.899	1.033	1,17e-6	846.973	1053	1,18e-6	850.962	1.033	1,18e-6
S_A	506.867	57	1,97e-6	446.118	55	4,48e-6	313.753	47	6,37e-6
S_V	506.796	12.017	1,97e-6	446.061	12.422	2,24e-6	313.665	14.011	3,19e-6

Tabla 4.6: Número total de palabras (N_t), tamaño del vocabulario (N) y probabilidad mínima (P_{min}) para cada estado en las diferentes particiones consideradas en la base de datos PVHDB

En primer lugar se realizaron experimentos con los ataques contenidos en RDB. Las curvas ROC obtenidas se muestran en la Figura 4.3 para valores de OOV en el rango 10^{-5} a 10^{-20}. Se puede constatar que en todas las curvas ROC más del 90% de los ataques fueron detectados con una tasa de falsos positivos del 0,4%. El rendimiento observado es ligeramente diferente para cada partición, observándose rendimientos excelentes en algunos casos. Así, se puede observar en la Figura 4.3c) que, para la partición considerada, el sistema presenta una tasa de aciertos superior al 99% con una tasa de falsos positivos menor que el 0,1%. Los resultados globales se pueden observar en la Figura 4.3d), donde se constata que el sistema tiene una tasa de aciertos del 99,89% con una tasa de falsos positivos de 0,1% cuando el valor asignado a la probabilidad de fuera de vocabulario es de 10^{-9}.

Continuando con esta base de datos, se realizaron experimentos con la base de datos de ataque OSVDB, mostrándose los resultados obtenidos en la Figura 4.4.

Se puede apreciar que los mejores resultados del sistema se obtienen cuando el valor del parámetro de fuera de vocabulario es superior a 10^{-10}.

Los mejores resultados obtenidos del ajuste de la probabilidad de OOV tras la implementación de las propuestas para grandes vocabularios serán considerados en lo que sigue como resultados de referencia de cara a la evaluación de las subsiguientes mejoras o técnicas que se propongan. De esta forma, las curvas ROC de referencia se muestran en la Figura 4.5. Consecuentemente, el valor de OOV seleccionado para todos los experimentos realizados es de 10^{-9} debido al comportamiento observado del sistema.

4.3 Suavizado de los vectores de observación

Una vez implementado el sistema en base a vectores de probabilidades de observación por estados, sería posible tratar de establecer un valor de suavizado ε más acorde con la base de datos de entrenamiento manejada. Dado que ya no existen valores nulos en estos vectores, el suavizado se realizará sobre los valores más pequeños existentes en los mismos. El razonamiento que lleva a la propuesta de utilización de un método de suavizado en este contexto está también relacionado con el problema de entrenamiento insuficiente y con una posible falta de representatividad de los valores obtenidos en el caso de grandes vocabularios. En particular, si el número de palabras

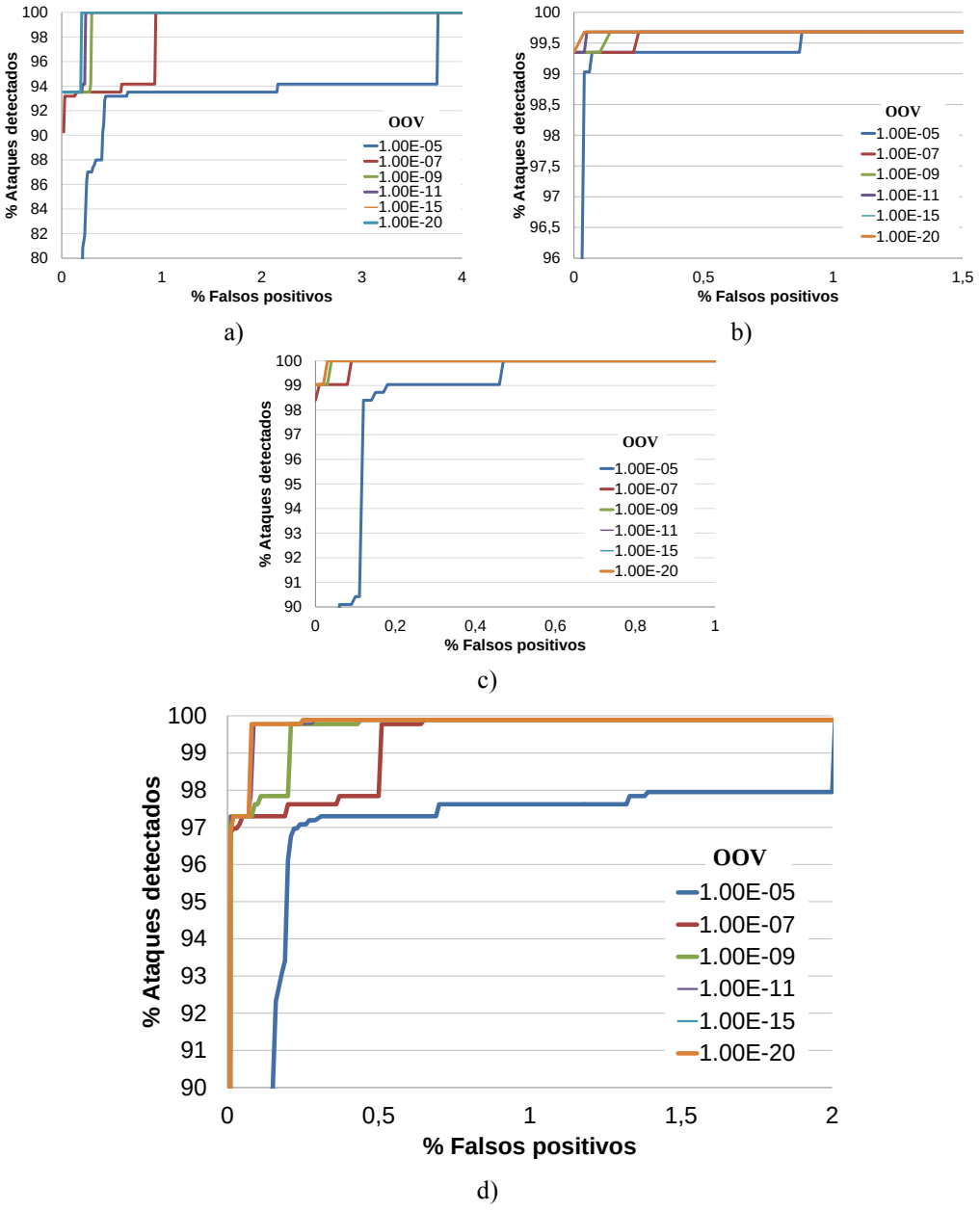


Figura 4.3: Curvas ROC para PVHDB/RDB para diferentes valores de OOV, para las particiones: a) PVHDB23, b) PVHDB13, c) PVHDB12 y d) resultados globales

diferentes es elevado y la base de datos de entrenamiento suficientemente grande, la confianza estadística que presentará el valor asignado a la probabilidad de una palabra observada una única vez será baja, en comparación a la de las palabras más frecuentes. Es decir, es posible que, debido al tamaño finito de la base de datos, se esté asignando probabilidades diferentes a palabras poco frecuentes que deberían presentar valores idénticos.

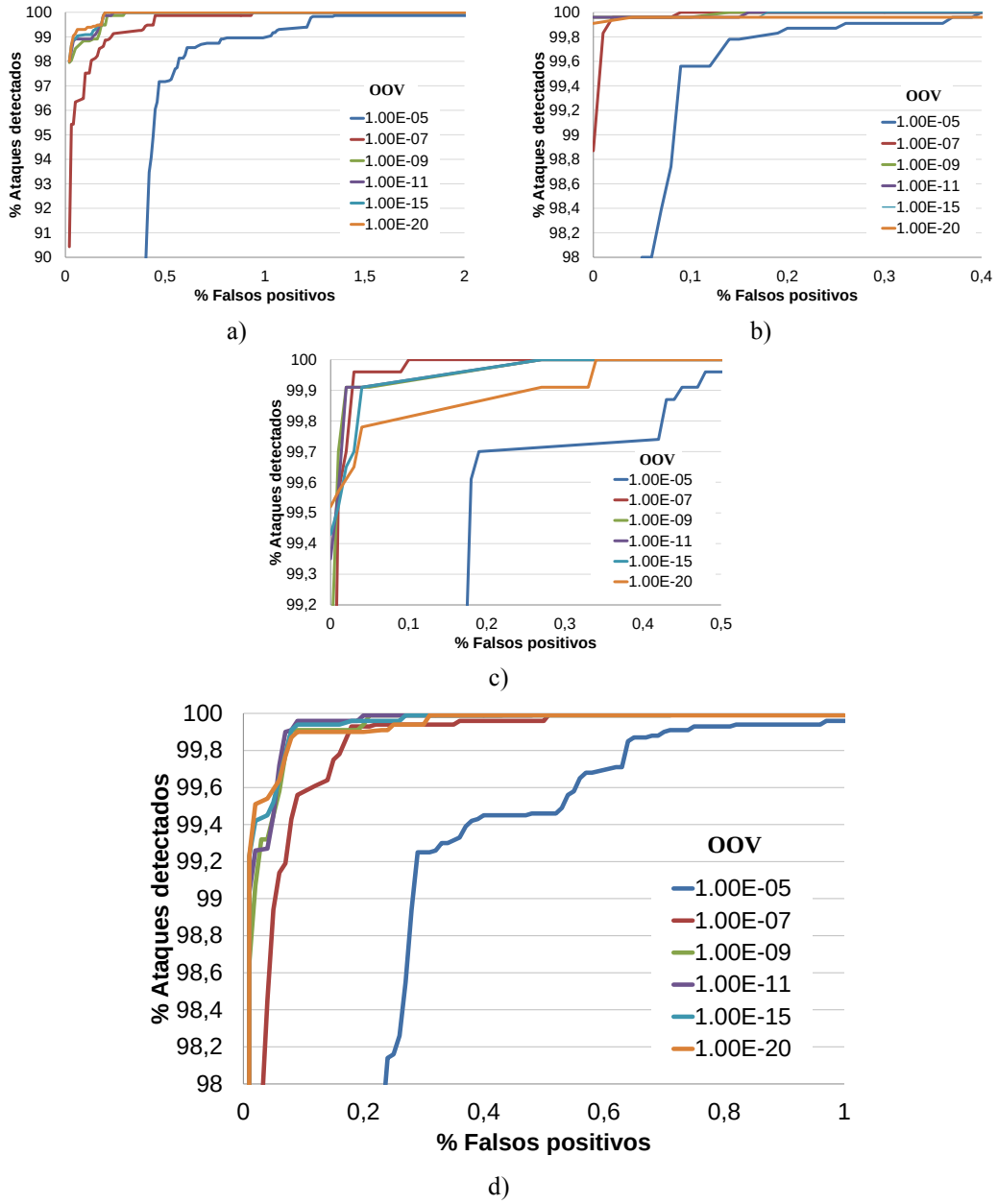


Figura 4.4: Curvas ROC para PVHDB/ OSVDB para diferentes valores de OOV, para las particiones: a) PVHDB23, b) PVHDB13, c) PVHDB12 y d) resultados globales

Con el objetivo de minimizar este problema, se propone el siguiente procedimiento de estimación para ε:

1) Se segmentan todos los URI en la base de datos de entrenamiento y se contabiliza, a través de un mero recuento, el número de veces que se observa una palabra j en un estado i. Éste será el parámetro n_{ij}.

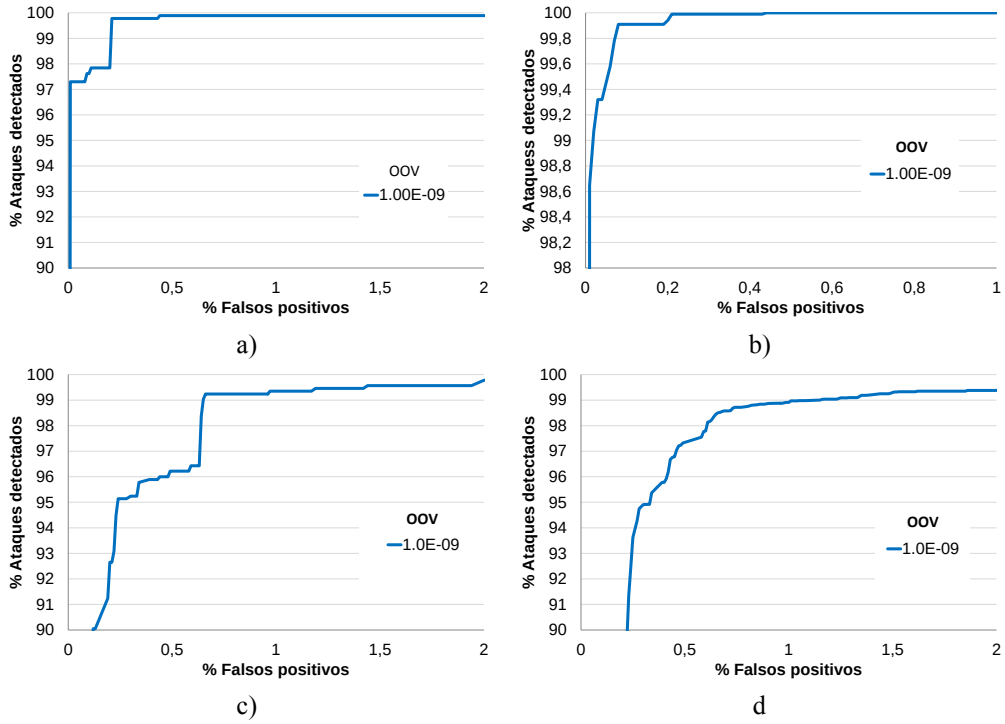


Figura 4.5: Curvas ROC de referencia: a) PVHDB/ RDB b) PVHDB/ OSVDB, c) CERES / RDB y d) CERES / OSVDB

2) Se calculan, del modo usual, las probabilidades de observación asociadas:

$$b_{ij} = \frac{n_{ij}}{\sum_{j=1}^{M} n_{ij}} \tag{4.3}$$

3) Se obtiene la probabilidad de observación global mínima, mayor que cero, aparecida en entrenamiento, b_m:

$$b_m = \min_{\forall i,j}\{b_{ij} \mid b_{ij} > 0\,, 1 \le i,j \le N\} \tag{4.4}$$

4) Se fija un valor mínimo aceptado para la probabilidad de observación:

$$\varepsilon = \mu . b_m \tag{4.5}$$

tomando μ valores mayores que cero.

A partir de este punto se procede al suavizado las probabilidades de observación b_{ij} tal como se indica en la expresión (4.2), donde ahora hemos de tener en cuenta que z_j representa el número de palabras con probabilidad menor o igual al ε establecido de acuerdo a (4.5) para el estado j, y n_j el número de palabras con probabilidad mayor que ε en dicho estado.

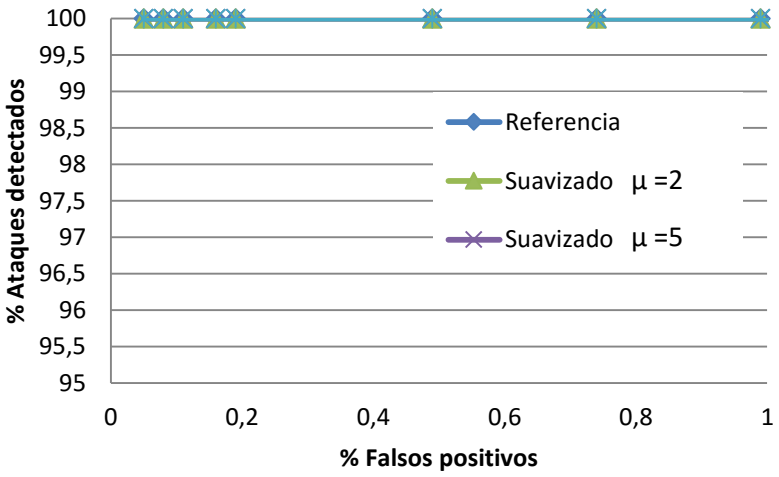


Figura 4.6: Curvas ROC obtenidas en experimentación para HUME / A usando la técnica de suavizado estándar y la propuesta como mejora

La técnica propuesta se basa en la suposición de que la menor probabilidad obtenida corresponde a la observación de un número muy reducido de instancias de la palabra asociada durante la fase de entrenamiento, lo que es típicamente cierto en el caso de grandes vocabularios.

De cara a la evaluación de la técnica de suavizado aquí planteada y su comparación con la considerada en el sistema de referencia, en un primer grupo de experimentos se ha tomado la base de datos DARPA referida a HUME y MARX. De modo similar a experimentaciones anteriores, se considera el esquema *leaving-one-out* con un 70% de los datos para entrenamiento y el 30% para test (ver Tabla 2.4 y Tabla 2.6). En lo que respecta al tráfico etiquetado como ataque, se toma el conjunto A (Tabla 2.6) compuesto por 1.455 ataques. Sobre esta base de trabajo se obtienen los resultados que se indican a continuación.

En una primera experimentación se han evaluado 2 valores distintos para el parámetro μ: 2,0 y 5,0. Los resultados obtenidos han sido comparados con los proporcionados por el sistema elegido como referencia. En la Figura 4.6 se muestran las curvas ROC de los resultados de la evaluación del sistema con la base de datos HUME, y en la Figura 4.7 para la base de datos MARX. En ambos casos se puede observar que los resultados obtenidos son los mismos tanto si se considera la técnica de suavizado estándar como la nueva aquí propuesta. Dado que la técnica estándar consigue unos resultados óptimos y que el tamaño del vocabulario es reducido, es lógico que el esquema de suavizado alternativo aquí planteado no consiga mejoras. En todo caso, sí es importante destacar como buen síntoma del esquema propuesto que los nuevos resultados no empeoran.

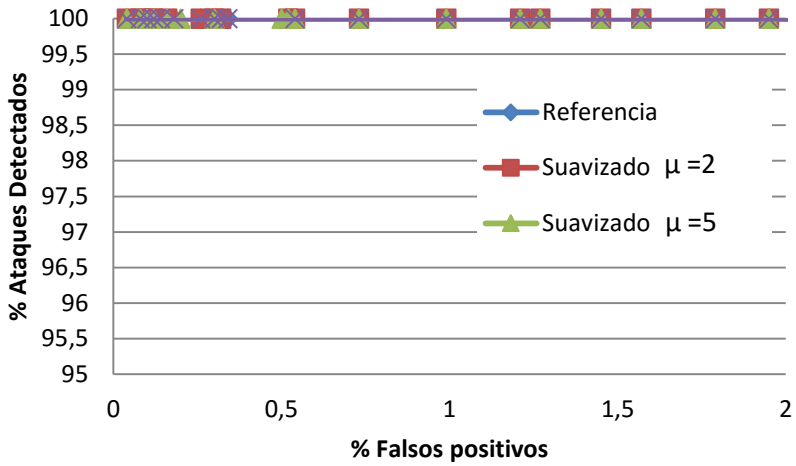


Figura 4.7: Curvas ROC obtenidas en experimentación para MARX / A usando la técnica de suavizado estándar y la propuesta como mejora

Dada, pues, la escasa representatividad de la base de datos DARPA, en la experimentación que sigue a continuación se han llevado a cabo pruebas adicionales similares con PVHDB y CERES. Aunque se han evaluado diferentes valores para μ, los resultados más relevantes se han obtenido para μ=5, por lo que en lo que sigue, a fin de permitir una mejor visualización de los resultados, solo se mostrarán las curvas correspondientes a este valor.

En un primer grupo de experimentos se utiliza la base de datos con tráfico normal PVHDB y el tráfico de ataques de la base de datos RDB. Los resultados obtenidos se muestran en la Figura 4.8. Como se puede observar, se alcanza el 99,89% de ataques detectados para un 0,44% de falsos positivos si no se aplica el suavizado propuesto. Análogo resultado para la tasa de detección se obtiene para los valores de μ evaluados. Sin embargo, se observa una disminución en la tasa de falsos positivos necesaria para μ=5, que se reduce 0,3%.

La aplicación del método de suavizado propuesto para PVHDB / OSVDB proporciona los resultados mostrados en la Figura 4.9. En esta se puede observar que se alcanza un 100% de ataques detectados para un 0,44% de falsos positivos en el caso de referencia, empeorando con la aplicación del método de suavizado propuesto (0,85% de FP para μ=5).

Los resultados experimentales obtenidos con estas bases de datos no son concluyentes, por cuanto que resultan contradictorios en los dos experimentos, si bien no se observan grandes diferencias en el comportamiento del sistema.

A continuación se evalúa la base de datos CERES con las correspondientes bases de ataques. Así, en la Figura 4.10 se muestran los resultados para CERES / RDB, mientras que en la Figura 4.11 se muestran los obtenidos para CERES / OSVDB. En la primera se puede observar que se alcanza un 99,35% de ataques detectados para un 0,99% de FP sin aplicar la propuesta, mientras que con el suavizado se alcanza dicha

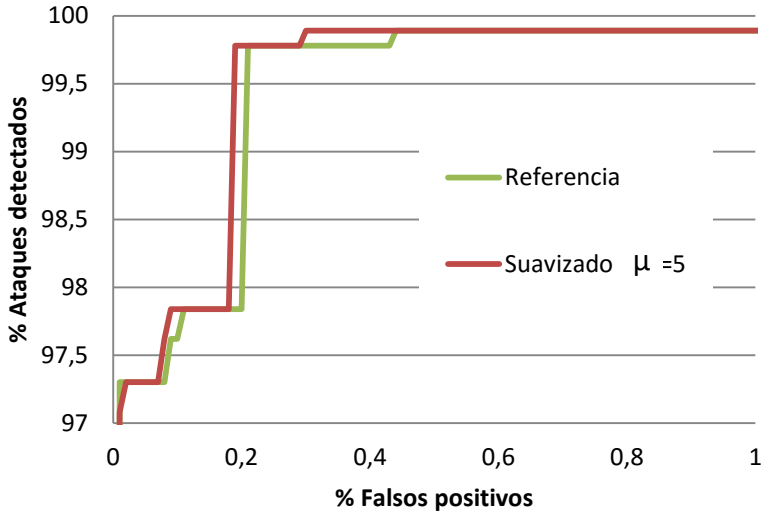


Figura 4.8: Curvas ROC para PVHDB /RDB usando la técnica de suavizado estándar y la propuesta como mejora

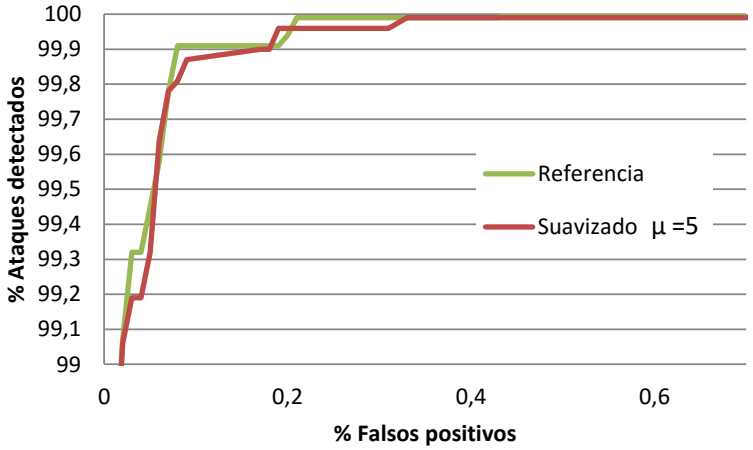


Figura 4.9: Curvas ROC para PVHDB /OSVDB usando la técnica de suavizado estándar y la propuesta como mejora

tasa de detección para una tasa de 1,01% de FP. En el segundo caso, el sistema sin suavizado alcanza un 98,9% de ataques detectados con un 0,99% de falsos positivos, mientras que con suavizado se llega a esa tasa de detección con un 1,05% de falsos positivos.

Una vez realizados todos los experimentos incorporando la mejora propuesta al sistema de detección de intrusos se observa en los resultados un solapamiento en general con los resultados obtenidos sin la técnica de suavizado, con variaciones mínimas respecto de este. Se concluye, en consecuencia, que la propuesta de suavizar las

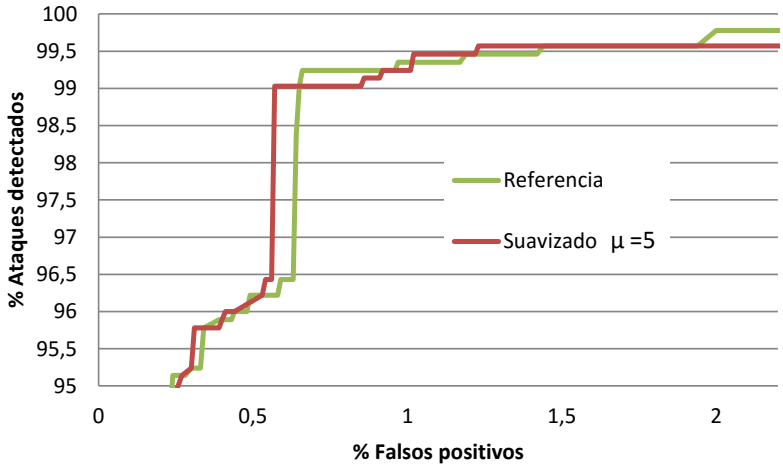


Figura 4.10: Curvas ROC para CERES/RDB usando la técnica de suavizado estándar y la propuesta como mejora

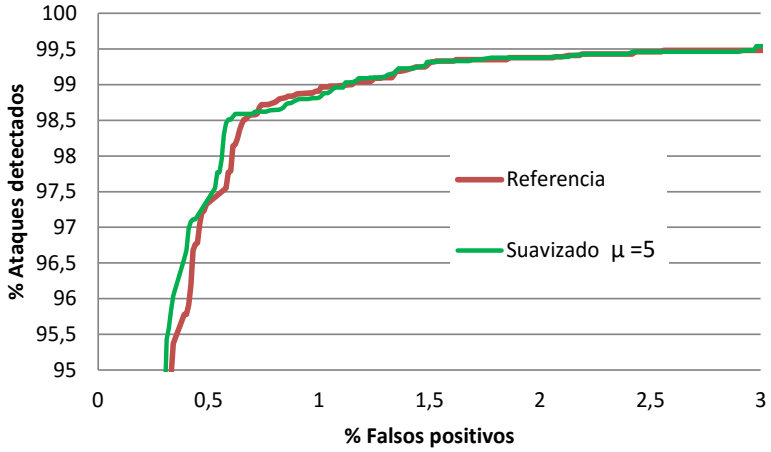


Figura 4.11: Curvas ROC para CERES/OSVDB usando la técnica de suavizado estándar y la propuesta como mejora

probabilidades de observación para las palabras menos frecuentes no aporta ninguna información que mejore significativamente el rendimiento y actuación del sistema de detección de intrusos con la técnica SSM en los escenarios considerados. Este comportamiento, inesperado a priori, puede ser debido a diversos efectos relacionados con el reducido número de palabras a las que afecta junto con la utilización de la probabilidad OOV, que puede enmascarar las posibles mejoras. Por este motivo, queda descartado su uso para mejorar el rendimiento.

Origen \ Destino	S_S	S_P	S_A	S_V	S_F	S_{OOS}
S_S	0	1	0	0	0	0
S_P	0	1	1	0	1	1
S_A	0	0	0	1	0	0
S_V	0	0	0	0	1	0
S_F	0	0	0	0	0	0

Tabla 4.7: Probabilidades de transiciones (A) para SSM según las especificaciones del protocolo

4.4 Estimación e inclusión de la matriz de transiciones

Como se describió en el Capítulo 3 y se ha vuelto a señalar en el Apartado 4.1, la aproximación IDS en nuestro caso implementada contempla el uso de probabilidades igual a 1 para todas las transiciones entre estados aceptadas en las especificaciones del protocolo/servicio. Así, en la Tabla 4.7 se muestran las probabilidades de transición entre estados permitidos según la especificación del protocolo objeto de estudio. Como se observa, todas son de valor 1, si están permitidas, o 0, si no lo están.

Esta aproximación implica obviar la información que pueda contenerse en las observaciones reales obtenidas del entorno. Con objeto de hacer coherente, pues, el empleo de las probabilidades de transición con la teoría de Markov subyacente, al tiempo que se trata de incorporar la información real proporcionada por el modelo ante la ocurrencia de dichas transiciones, en lo que sigue se plantea la estimación real de las probabilidades a partir de los datos de entrenamiento. Para ello, de acuerdo al método de entrenamiento descrito en el Apartado 3.1.2, bastará con seguir el siguiente procedimiento:

1) Determinar la secuencia de estados atravesada por el modelo al decodificar cada uno de los URI dispuestos en entrenamiento.
2) Contabilizar del número de veces que se produce una transición desde un estado i a otro j, $n_{ij} \forall i,j \in \{1,\ldots,N\}$.
3) Normalizar los valores obtenidos para las observaciones de transición:

$$a_{ij} = \frac{n_{ij}}{\sum_{j=1}^{N} n_{ij}} \tag{4.6}$$

A modo de ejemplo sencillo de lo expuesto, consideremos los 3 URI del ejemplo del Apartado 3.2.2, cuya decodificación en estados es:

$URI_1 = S_S\ S_P\ S_P\ S_P\ S_F$

$URI_2 = S_S\ S_P\ S_P\ S_P\ S_{OOS}$

Origen \ Destino	S_S	S_P	S_A	S_V	S_F	S_{OOS}
S_S	0	3/3=1	0	0	0	0
S_P	0	6/9=0,67	1/9=0,11	0	1/9=0,11	1/9=0,11
S_A	0	0	0	1/1=1	0	0
S_V	0	0	0	0	1/1=1	0
S_F	0	0	0	0	0	0

Tabla 4.8: Matriz A de transiciones "real" derivada de los datos de entrenamiento

URI_3= S_S S_P S_P S_P S_A S_V S_F

En este caso, es evidente que

n_{SP}=3,siendo $\sum_{j=1}^{N} n_{Sj} = 3$;

n_{PP}=6, n_{PA}=1, n_{PF}=1 y n_{POOS}=1, siendo $\sum_{j=1}^{N} n_{Pj} = 9$;

n_{AV}=1, siendo $\sum_{j=1}^{N} n_{Aj} = 1$; y

n_{VF}=1, siendo $\sum_{j=1}^{N} n_{Vj} = 1$

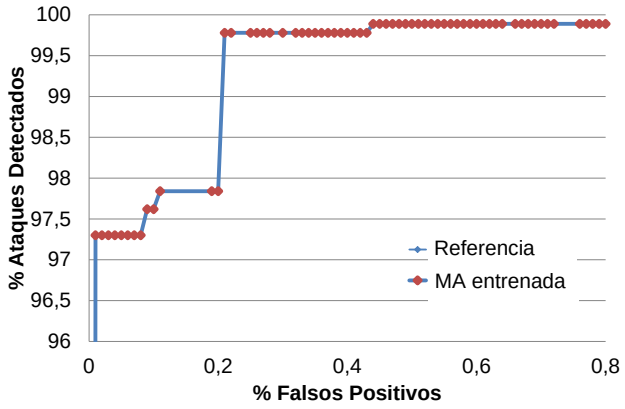
En consecuencia, las probabilidades de transición estimadas quedarían como se muestra en la Tabla 4.8.

Puesta en práctica la estimación real de la matriz de probabilidades de transición y su uso posterior en detección para el entorno de experimentación considerado, en la Figura 4.12 se muestran los resultados obtenidos. Las distintas gráficas muestran las ROC proporcionadas por el sistema de referencia y cuando se considera la matriz **A** estimada en entrenamiento para cada una de las bases de datos consideradas.

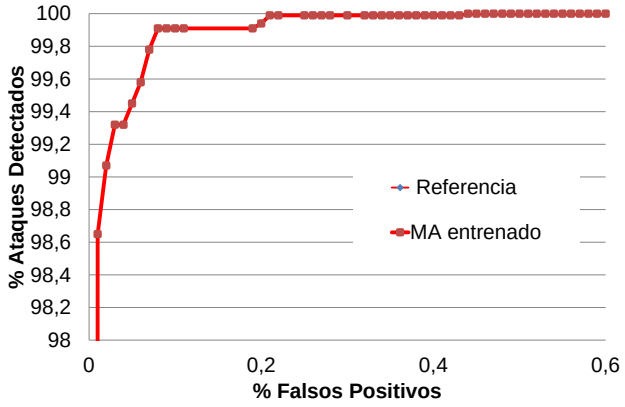
Puede comprobarse que en ninguno de los casos estudiados se consigue una mejora en detección cuando se incluye la matriz **A** "real" en el modelado. En definitiva, lamentablemente, la mejora propuesta al sistema de referencia no es tal.

4.5 Esquema OOV dependiente del estado

Como se ha apuntado anteriormente, una vez asumida la reducción de la matriz de observaciones **B**, la posible aparición en detección de palabras no incluidas en **B** se trata como una situación OOV. Frente al caso estándar implementado en el sistema de referencia, en la presente propuesta de mejora se plantea que el valor correspondiente de observación OOV no sea único, sino dependiente del estado. La motivación de esta propuesta es simple. El modelo considerado dispone de 3 estados principales: S_P, de ruta, S_A, de atributo, y S_V, de valor. De todos ellos, es evidente que el estado que resulta más probable para que ocurra la situación OOV es S_V, seguido de S_A y, finalmente, S_P. En consecuencia, proponemos que la penalización OOV en el caso del estado S_P sea superior a los dos restantes, mientras que la ocurrida en S_V sea la inferior. En otras palabras, el valor OOV asociado a S_P será más pequeño que el asociado a S_A y S_V, siendo el correspondiente a S_v el mayor de todos.



a)



b)

Figura 4.12: Curvas ROC entrenando y sin entrenar probabilidad de transición: a) PVHBD / RDB, b) PVHDB / OSVDB

A continuación se discuten los resultados experimentales obtenidos con la implementación de la mejora propuesta. En nuestro estudio se han realizado diferentes experimentos variando los valores del parámetro de fuera de vocabulario para cada estado entre 10^{-20} y 10^{-3}. Seguidamente se muestran y comentan las curvas ROC más representativas conseguidas.

En relación a la base de datos DARPA, en la Figura 4.13 y en la Figura 4.14 se muestran los resultados para HUME y MARX, respectivamente. Además de la ROC del sistema de referencia, se indican las obtenidas con los siguientes valores para fuera de vocabulario por estado:

- $S_P \rightarrow 10^{-9}$, $S_A \rightarrow 10^{-7}$ y $S_V \rightarrow 10^{-5}$.
- $S_P \rightarrow 10^{-13}$, $S_A \rightarrow 10^{-3}$ y $S_V \rightarrow 10^{-3}$.

habiéndose utilizado la notación A_B_C para indicar los exponentes, cambiados de signo, de las probabilidades OOV correspondientes, respectivamente, a S_P , S_A y S_V.

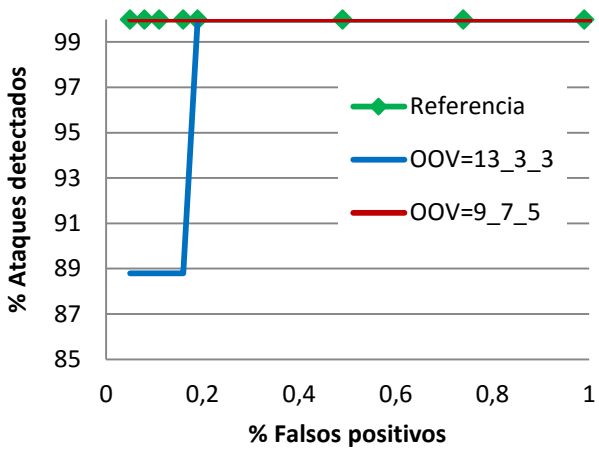


Figura 4.13: ROC para HUME / A con parámetro de fuera de vocabulario dependiente del estado

Puede observarse que, como sucedía en el caso del suavizado de la matriz **B** planteado en el Apartado 4.3, las prestaciones conseguidas no consiguen mejorar las del sistema de referencia. Evidentemente, porque éstas son óptimas. En todo caso, es significativo indicar que, al tiempo que éstas no empeoran, con una aproximación más coherente en la selección del parámetro de fuera de vocabulario, sí se obtiene una mayor eficiencia computacional en la detección, motivada por la reducción de la matriz **B**. Si bien no puede concluirse que dicha mejora sea crítica, tampoco puede despreciarse sin más. Así, las mediciones realizadas arrojan una mejora de en torno al 12% de reducción en el tiempo de cómputo involucrado en detección respecto del sistema de referencia.

Dada lo poco conclusiva que resulta la experimentación sobre DARPA, una vez más se han efectuado pruebas adicionales similares sobre las bases de datos de tráfico limpio PVHDB y CERES, considerando en ambos casos las bases de ataques RDB y OSVDB para la obtención de las ROC correspondientes. Los resultados conseguidos en cada caso se detallan a continuación.

En la Figura 4.15 se muestra la experimentación PVHDB / RDB, donde se evidencia que los mejores resultados corresponden a los valores OOV siguientes: $S_P \rightarrow 10^{-9}$, $S_A \rightarrow 10^{-12}$ y $S_V \rightarrow 10^{-7}$. En este caso, se alcanzan tasas de detección superiores al 99,5% con una tasa de falsos positivos en torno al 0,1%. Unos resultados muy similares son los proporcionados por $S_P \rightarrow 10^{-8}$, $S_A \rightarrow 10^{-11}$ y $S_V \rightarrow 10^{-7}$. En todo caso, ambos resultan claramente mejores que los obtenidos por el sistema de referencia, el cual, como puede observarse, alcanza una tasa de detección inferior al 98% para una tasa de falsos positivos del 0,1%.

En la Figura 4.16 se muestran los resultados obtenidos en la experimentación PVHDB / OSVDB, de donde puede concluirse que los valores OOV por estado que mejores resultados proporcionan son los mismos que en el caso de PVHDB / RDB. Es

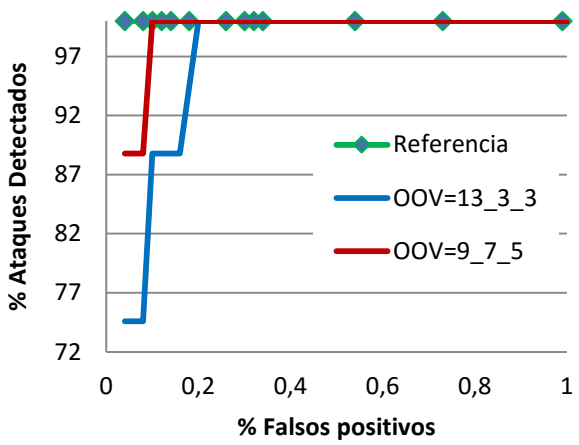


Figura 4.14: ROC para MARX /A con parámetro de fuera de vocabulario dependiente del estado

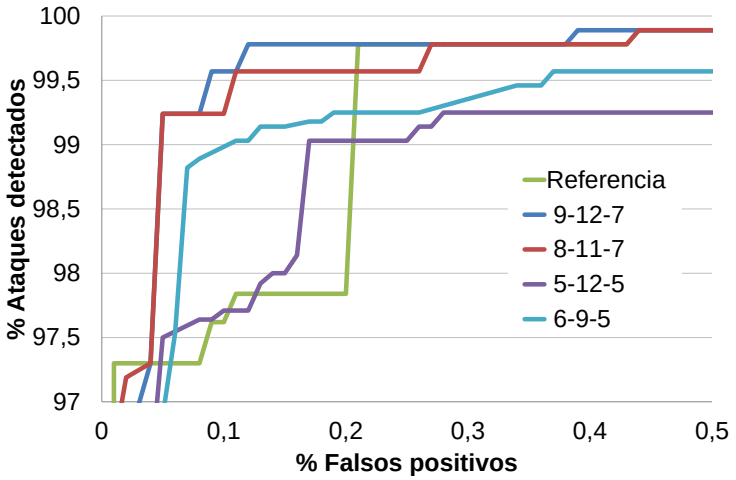


Figura 4.15: Curvas ROC para PVHDB/ RDB, para el sistema de referencia y haciendo uso de valores OOV dependientes del estado

de destacar, sin embargo, que en este caso las prestaciones evidenciadas por el sistema de referencia no son muy inferiores, como sí sucedía en la experimentación anterior.

El siguiente grupo de experimentos realizados se refieren a la base de datos con el tráfico normal CERES. De nuevo, para la consecución de las ROC se ha hecho uso de las bases de datos de ataques RDB y OSVDB. En la Figura 4.17 se muestran los resultados para el primer caso, y en la Figura 4.18 para el segundo.

En el caso CERES / RDB los resultados mejores corresponden a los mismos valores OOV dependientes del estado que los de la base de datos PVHDB, alcanzándose una tasa de detección del 99,78% con una tasa de falsos positivos del 2%.

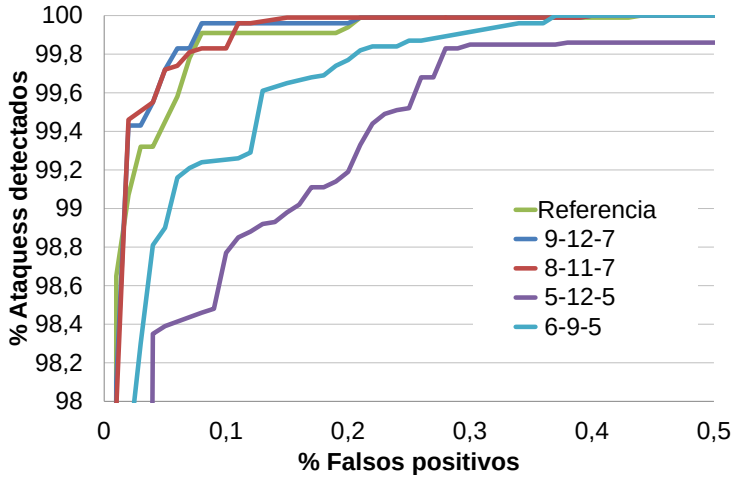


Figura 4.16: Curvas ROC para PVHDB/ OSVDB, para el sistema de referencia y haciendo uso de valores OOV dependientes del estado

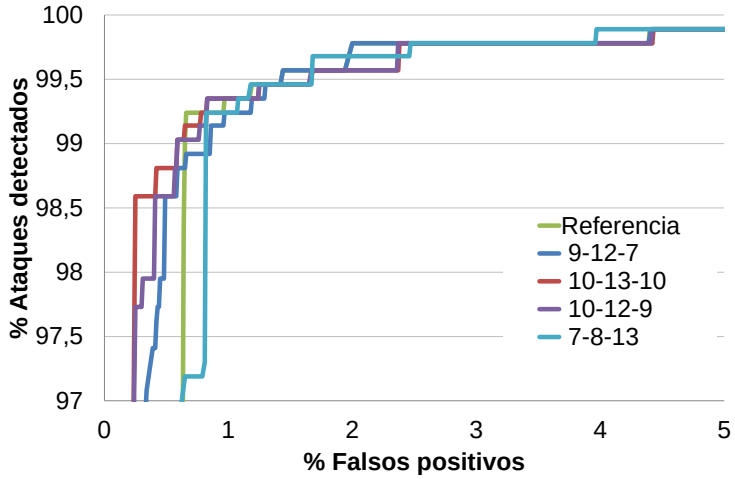


Figura 4.17: Curvas ROC para CERES / RDB, para el sistema de referencia y haciendo uso de valores OOV dependientes del estado

Por su parte, para CERES / OSVDB los mejores resultados corresponden a los valores 10^{-7}, 10^{-8} y 10^{-13} para los estados S_P, S_A y S_V, respectivamente. Con ellos, el sistema alcanza una tasa de falsos positivos del 2,97% con una tasa de ataques detectados del 99,86%.

En ambas experimentaciones se evidencia la consecución de un mejor comportamiento del sistema IDS haciendo uso de valores OOV dependientes del estado que la obtenida por el sistema de referencia, donde se adopta un único OOV global para todas las situaciones.

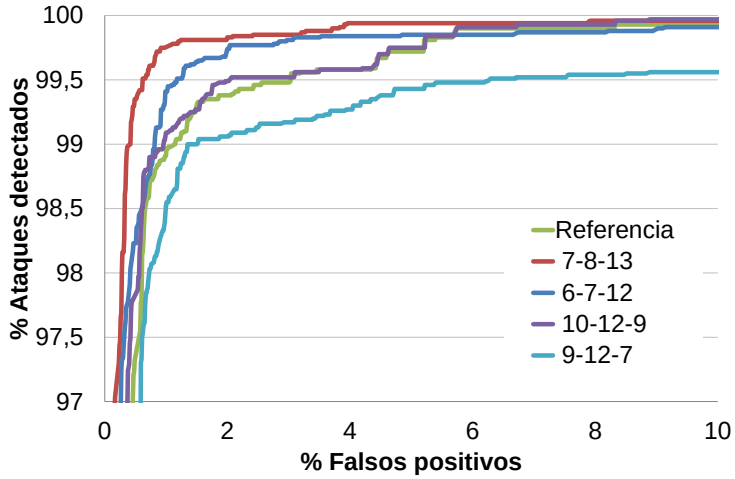


Figura 4.18: Curvas ROC para CERES / OSVDB, para el sistema de referencia y haciendo uso de valores OOV dependientes del estado

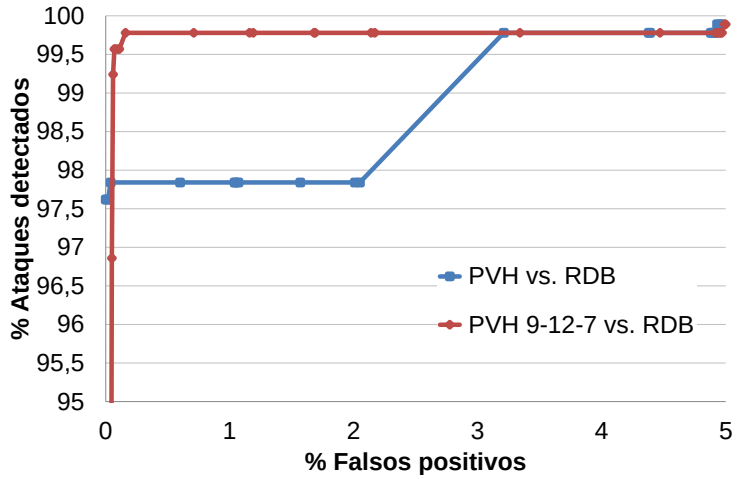
4.5.1 Validación de los resultados

Vistos los buenos resultados obtenidos en test por la mejora planteada, seguidamente se procederá a validar los mismos. Es decir, la experimentación antes realizada ha tratado de ser exhaustiva en la búsqueda de los mejores valores OOV a especificar por estado; sin embargo, estas pruebas han sido llevadas a cabo sobre las particiones de test de las respectivas bases de datos. Por ello, en lo que sigue se repetirán los experimentos sobre las particiones establecidas para validar los resultados conseguidos. Así, los 205.880 URI existentes en PVHDB y los 35.458 de CERES, correspondientes a las particiones de validación, no han sido considerados en modo alguno hasta este punto para entrenar o sintonizar el sistema.

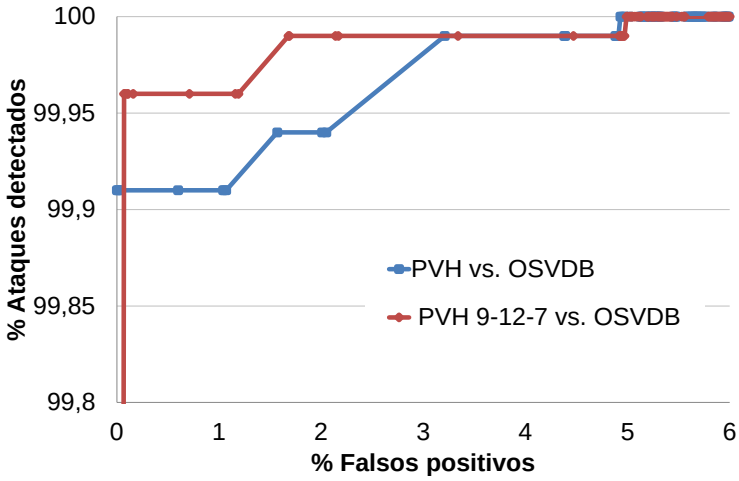
En la Figura 4.19 se muestran las curvas ROC para la base de datos PVHDB, considerándose como bases de datos de ataques RDB (subfigura a) y OSVDB (subfigura b). En dicha figura se esquematiza tanto la ROC del sistema de referencia (con valor OOV único igual a 10^{-9}) como la proporcionada por nuestra propuesta de OOV dependiente del estado con los mejores valores por estado obtenidos en la experimentación de test: $S_P \rightarrow 10^{-9}$, $S_A \rightarrow 10^{-12}$ y $S_V \rightarrow 10^{-7}$.

Se puede observar en la gráfica que nuestra propuesta OOV dependiente del estado alcanza, para el caso PVHDB / RDB, una tasa de detección superior a 99% con una tasa de falsos positivos por debajo del 1%. Por el contrario, el sistema de referencia también consigue una tasa superior a 99%, pero con una tasa de falsos positivos igual a 3%.

Un comportamiento análogo se obtiene para el caso PVHDB / OSVDB. Con los mismos valores de OOV por estado, se consigue una tasa de detección superior a 99,9% con un valor de falsos positivos inferior al 2%. Sin embargo, el sistema de



a)

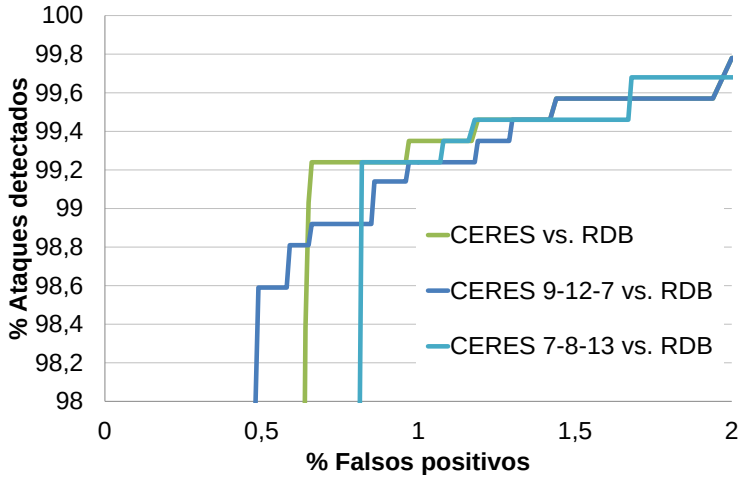


b)

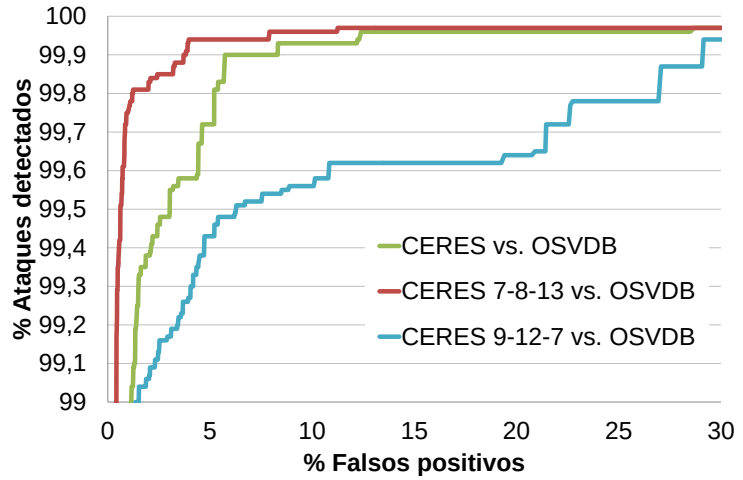
Figura 4.19: Sistema de referencia y técnica OOV dependiente del estado para: a) PVH/RDB y B) PVH/OSVDB

referencia alcanza esta misma tasa de detección con una tasa de falsos positivos superior al 3%.

Finalmente, en la Figura 4.20 se muestran las curvas ROC de detección para la base de datos CERES, considerándose también como bases de datos de ataques RDB (subfigura a) y OSVDB (subfigura b). Como antes, se representa tanto la ROC del sistema de referencia (con valor OOV único igual a 10^{-9}) como la proporcionada por nuestra propuesta de OOV dependiente del estado con los mejores valores por estado obtenidos anteriormente en la experimentación de test: $S_P \rightarrow 10^{-9}$, $S_A \rightarrow 10^{-12}$ y $S_V \rightarrow 10^{-7}$.



a)



b)

Figura 4.20: Sistema de referencia y técnica OOV dependiente del estado para: a) CERES/RDB y b) CERES/OSVDB

Se puede observar en la gráfica que la propuesta OOV dependiente de estado alcanza para el caso CERES/RDB una tasa superior al 99,8% con una tasa de falsos positivos del 2%. Frente a esto, el sistema de referencia también consigue una tasa superior a 99,3% con una tasa de falsos positivos al 2,2%.

Para el caso de CERES / OSVDB, con los mismos valores de OOV por estado, se consigue una tasa de detección superior al 99,8% con un valor de falsos positivos superior al 10%. Sin embargo, el sistema de referencia supera el 12% en el porcentaje de los falsos positivos.

Como conclusión final de este apartado, puede afirmarse que las propuestas de mejora planteadas, uso de la matriz **B** por estados y consiguiente no suavizado de la

misma y uso de valores OOV dependientes del estado, resulta exitosa desde el punto de vista de las prestaciones conseguidas por el sistema de detección.

5 Modelado explícito de ataques

Una vez mejorado el sistema SSM para su operación en escenarios con grandes vocabularios, en el presente capítulo se plantea mejorar sus prestaciones modificando el modo en el que se utiliza la técnica subyacente. En su formulación original, y en las mejoras propuestas y evaluadas hasta ahora, se ha usado el sistema como detector, basándose la clasificación de los eventos analizados en la comprobación del índice de anormalidad y en la superación o no de un umbral de detección. Sin embargo, SSM permite modelar el comportamiento del sistema, por lo que podría utilizarse como clasificador, esto es, no para determinar el grado de desviación de las observaciones respecto del modelo, sino para determinar cuál de entre varios modelos es el que mejor se ajusta a dichas observaciones. En este sentido, a continuación se propone y evalúa la utilización de SSM para modelar también los ataques, lo que permite el diseño de un sistema híbrido en el que se determina la categoría a la que pertenece cada evento en base a discernir si queda mejor representado mediante el modelo de normalidad o el de ataques.

La utilización de métodos o sistemas híbridos ha sido descrita en trabajos previos de numerosos autores (véase el Capítulo 1) en los que se proporcionan resultados que mejoran los obtenidos por las técnicas individuales utilizadas, al combinar las potencialidades de ambas. En este sentido, la aproximación más habitual consiste en combinar la detección basada en firmas con la detección basada en anomalías, como es el caso de [Depren et al., 2005] [Reis et al., 2002] [Tombini et al., 2004] y de trabajos previos del equipo de investigación [Díaz-Verdejo et al., 2007]. Otra aproximación, que es la considerada en el presente trabajo, se basa en el modelado de los ataques y del comportamiento normal del sistema [Fontenelle et al., 2007].

De esta forma, a continuación se describirá la arquitectura del sistema, en la que se consideran dos modelos diferentes para representar las instancias normales y las de ataque, respectivamente. Inicialmente, se considerará la operación como reconocedor, asignando a cada evento la clase que proporcione mayor probabilidad. A partir del análisis de los resultados obtenidos, se propone una medida de la confianza en la clasificación de cada evento, que puede ser utilizada para mejorar el rendimiento del sistema en base a un modo de operación híbrido en dos etapas en el que se combina el funcionamiento como reconocedor con la detección basada en anomalías, que se usa

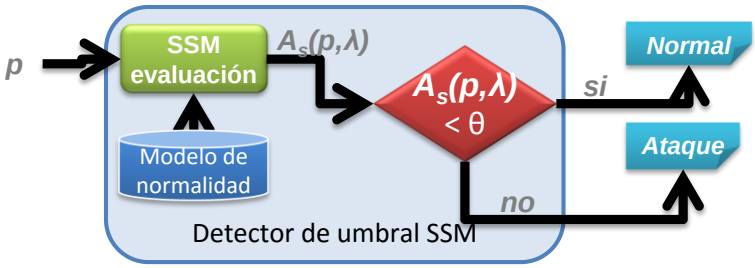


Figura 5.1: IDS basado en detección de umbral mediante SSM

para aquellos casos en los que el reconocedor proporciona un bajo índice de confianza. Finalmente, tras validar el sistema propuesto, se evalúa la posibilidad de aplicar una técnica de entrenamiento discriminativo a los dos modelos considerados a partir de la medida de confianza.

5.1 Sistema basado en reconocimiento normal/ataque

El sistema de detección de intrusos basado en SSM clasifica los eventos, p, como anómalos (Apartado 3.3.3) en base al *índice de anormalidad,* $N_s(U)$ —Ec. (3.18)—, y a un umbral de detección, θ, según:

$$Clase(p) = \begin{cases} Normal & si\ N_S(p) < \theta \\ Anómalo & si\ N_S(p) \geq \theta \end{cases} \quad (5.1)$$

Su funcionamiento es, por tanto, el de un detector de umbral (Figura 5.1). Sin embargo, el modelado de Markov subyacente, esto es, el modelo SSM, es susceptible de ser utilizado para representar cualquier tipo de mensaje asociado al servicio, independientemente de su naturaleza. En particular, en el caso considerado en el presente trabajo, podría utilizarse para modelar tanto los URI normales como los URI de ataque.

Consecuentemente, la propuesta a evaluar consiste en usar el modelado SSM para representar ambos tipos de peticiones normales y de ataque. Por lo tanto, se entrenan por separado un modelo para las cargas útiles normales (λ_N) y un modelo a partir de las peticiones contenidas en los ataques (λ_A), que serán, respectivamente, el modelo de normalidad y el de ataque.

Así, la clasificación de una carga útil, p, será realizada por un reconocedor (Figura 5.2) que asignará la clase del modelo que proporcione la probabilidad más alta, esto es:

$$Clase(p) = \begin{cases} normal, & si\ P(p|\lambda_N) \geq P(p|\lambda_A) \\ ataque, & en\ otro\ caso \end{cases} \quad (5.2)$$

El rendimiento de esta nueva aproximación se evalúa con las bases de datos disponibles con cargas útiles normales y de ataque y utilizando el mismo procedimiento

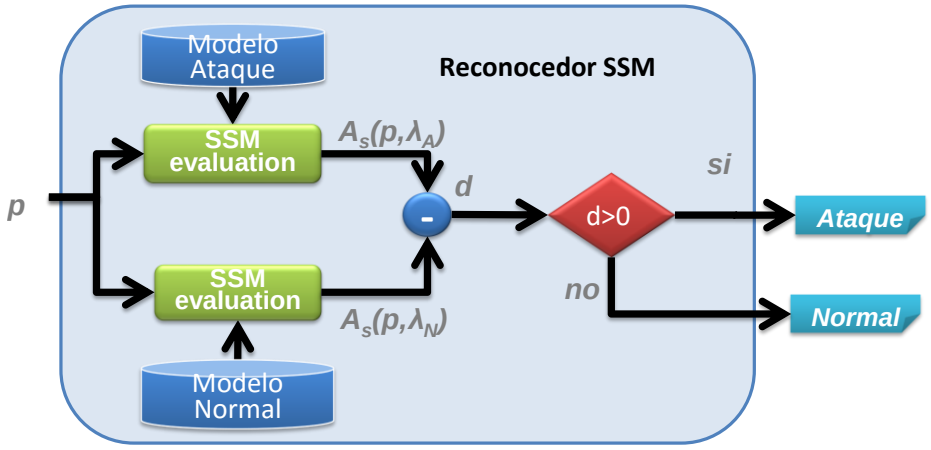


Figura 5.2: IDS basado en reconocimiento mediante SSM

Experimento	Tasa de detección	Falsos Positivos
PVHDB / RDB	95,5 %	2,0 %
PVHDB / OSVDB	90,0 %	0,05 %

Tabla 5.1: Resultados obtenidos con el reconocedor SSM

de *leaving-one-out* utilizado en anteriores grupos de experimentaciones. Los resultados se comparan con los establecidos como referencia (Apartado 4.1), usándose el mismo valor para el parámetro fuera de vocabulario OOV=10^{-9} a fin de poder realizar la comparación. Los resultados obtenidos se muestran en la Tabla 5.1. Es importante notar que, como la decisión se basa en el modelo que proporciona la probabilidad más alta, sólo es posible un punto de operación para el reconocedor. Por lo tanto, no se puede establecer un compromiso entre la tasa de detección y la tasa de falsos positivos a través de ningún parámetro de ajuste (umbral), como se realizó para el sistema de referencia. Es evidente que los resultados muestran un rendimiento inferior al obtenido por el sistema de referencia, ya que el punto de operación está por debajo de la curva ROC (Figura 4.5) para el sistema de referencia.

A continuación, y a la vista de los resultados, se procede a realizar un análisis más profundo a fin de determinar posibles mejoras y las causas de estos resultados.

5.2 Índice de confianza de la clasificación

Para evaluar el comportamiento del reconocedor, se realiza un análisis del área de confusión, es decir, de las cargas útiles para las que se producen errores de clasificación y sus probabilidades asociadas. Para ello se define la medida $S(p)$ como la diferencia entre las probabilidades de la carga útil analizada, p, según los modelos de normalidad y de ataque, esto es:

$$S(p) = P(p|\lambda_A) - P(p|\lambda_N) \tag{5.3}$$

Esta medida se puede interpretar de la siguiente manera. Un alto valor positivo para una carga útil significa que la probabilidad de que esta sea un ataque es significativamente mayor que la de ser normal. Por el contrario, un alto valor negativo implica que la probabilidad de ser normal es mayor que la de ser un ataque. Por tanto, las cargas útiles para las que esta medida es pequeña (tanto positiva como negativa) se encuentran cerca del límite de decisión y, por tanto, se consideran dentro de la zona de confusión. En consecuencia, denominaremos a $S(p)$ *índice de confianza*, puesto que valores absolutos altos de este índice de confianza están asociados a una clara discriminación entre ambas categorías, de acuerdo a los modelos utilizados. Por el contrario, valores pequeños del índice de confianza implicarán que los dos modelos proporcionan valores similares y, en consecuencia, la confianza en la clasificación será baja.

El criterio de decisión normal/ataque se puede expresar en función del índice de confianza de la forma:

$$Clase(p) = \begin{cases} normal, & si\ S(p) < 0 \\ ataque, & en\ otro\ caso \end{cases} \tag{5.4}$$

Podemos analizar el comportamiento del reconocedor a partir de la representación de los valores de $S(p)$ obtenidos para las cargas útiles normales y de ataques en ambos experimentos (Figura 5.3 y Figura 5.4). En estas figuras se puede observar que la zona de confusión, que corresponde a la zona en la que se solapan las distribuciones de las cargas útiles normales (en azul) y de ataques (en rojo), es una zona relativamente pequeña y con un número reducido de cargas útiles en torno al valor 0.

Una primera observación derivada de este análisis es que es posible modificar el punto de operación del reconocedor desplazando el origen mediante un parámetro aditivo. Esto es, se puede introducir un offset, σ, en la Ec. (5.3) de forma que el criterio de decisión sea:

$$Clase(p) = \begin{cases} normal, & si\ S(p) < \sigma \\ ataque, & en\ otro\ caso \end{cases} \tag{5.5}$$

A la vista de las gráficas obtenidas, un valor positivo para σ reducirá la tasa de falsos positivos, aunque a costa de reducir la tasa de detección. El efecto contrario se conseguirá asignando valores negativos a dicho offset. Sin embargo, de esta forma no se conseguirá mejorar la confianza en la clasificación obtenida.

Una aproximación alternativa consistiría en no clasificar aquellas cargas útiles para las que el índice de confianza es pequeño, de forma que se puede establecer un umbral, μ, que denominaremos *umbral de confusión*, a partir del que se realiza la clasificación (Figura 5.5). Así, el criterio de decisión resulta:

$$Clase(p) = \begin{cases} normal & si\ S(P) \leq -\mu \\ ataque & si\ S(P) \geq \mu \\ desconocido & si\ |S(P)| < \mu \end{cases} \tag{5.6}$$

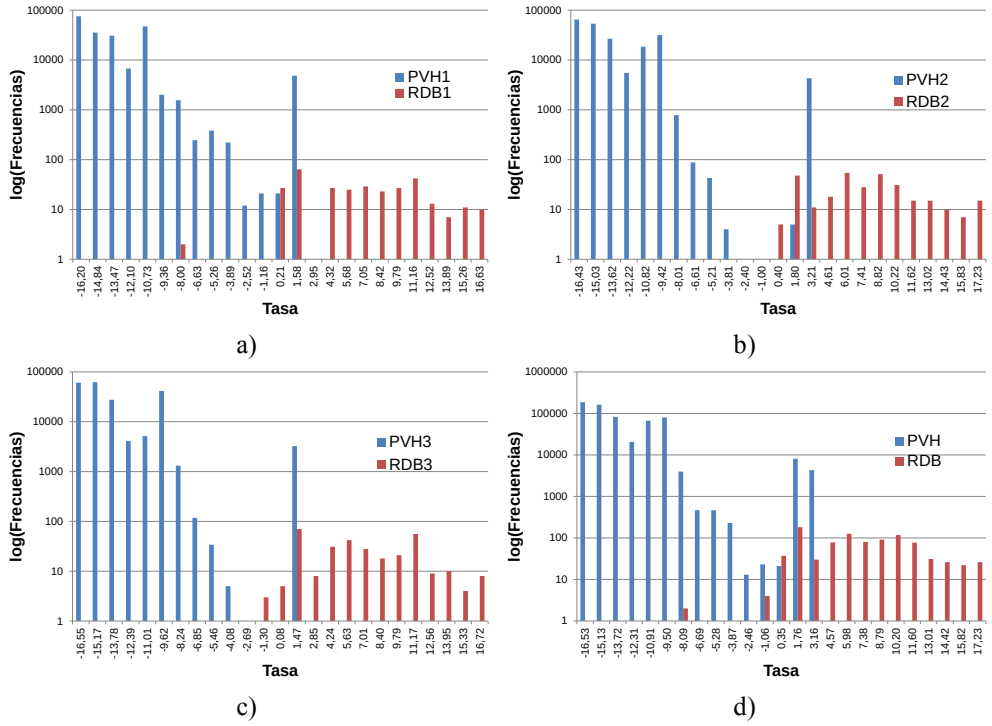


Figura 5.3: Distribución de S(p) para el experimento PVHDB / RDB. a) PVHDB23, b) PVHDB13, c) PVHDB12 y d) Resultados globales

La evaluación de esta aproximación sobre las mismas bases de datos proporciona los resultados mostrados en la Tabla 5.2. Los resultados evidencian que es posible clasificar correctamente la mayoría de las cargas útiles si se selecciona un valor del umbral de confusión alrededor de 3,0. En este caso, alrededor del 25% de los ataques y menos del 5% de las cargas útiles normales se quedan sin clasificar. De esta manera, la confianza en la decisión para las peticiones clasificadas será alta, pero a costa de un número inaceptable de ellas sin clasificar. Por tanto, el reto se centra en el procesamiento de estas cargas útiles dudosas con el fin de aumentar el rendimiento en la clasificación.

5.3 Detector híbrido

A fin de categorizar las cargas útiles cuyo índice de confianza queda en la zona de confusión, se propone la utilización de un detector híbrido que combine el reconocedor SSM y el detector de umbral SSM, actuando en dos pasos (Figura 5.6). De esta forma, las observaciones son analizadas en primer lugar por el reconocedor que las etiquetará si el índice de confianza queda fuera de la zona de confusión establecida. En caso contrario, serán clasificadas de acuerdo al detector. Se utilizan, en consecuencia, dos umbrales: el umbral de confusión, μ, y el umbral de detección, θ, resultando:

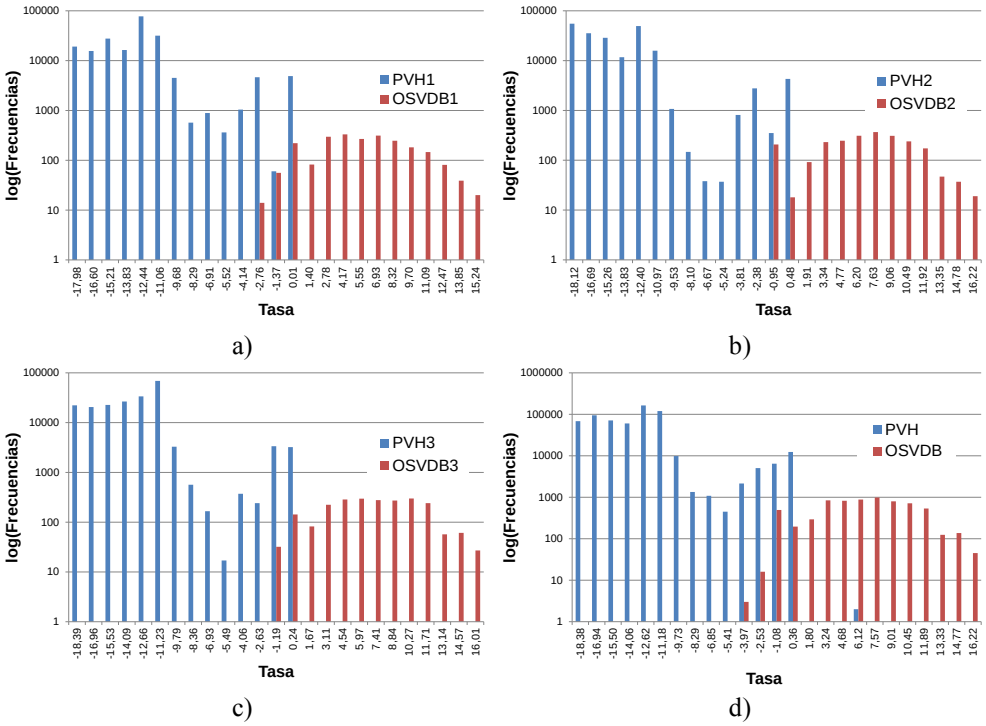


Figura 5.4: Distribución de S(p) para el experimento PVHDB / OSVDB. a) PVHDB23, b) PVHDB13, c) PVHDB12 y d) Resultados globales

$$Clase(p) = \begin{cases} normal, & si \begin{cases} P(p|\lambda_A) - P(p|\lambda_N) \leq -\mu \\ o \\ |P(p|\lambda_A) - P(p|\lambda_N)| < \mu \; y \; P(p|\lambda_N) \geq \theta \end{cases} \\ ataque, & si \begin{cases} P(p|\lambda_A) - P(p|\lambda_N) \geq \mu \\ o \\ |P(p|\lambda_A) - P(p|\lambda_N)| < \mu \; y \; P(p|\lambda_N) < \theta \end{cases} \end{cases} \quad (5.7)$$

El razonamiento detrás de este enfoque es considerar el sistema original basado en detección de anomalías cuando las probabilidades de acuerdo al modelo de normalidad y de ataques son muy similares. En este caso, la información discriminativa proporcionada comparando ambas probabilidades no se considera suficiente para tomar una decisión adecuada y es, consecuentemente, descartada. Por tanto, la clasificación se realiza de acuerdo al modelo de normalidad. Implícitamente, esta aproximación considera más fiable el modelo de normalidad que el de ataque, lo que parece una buena hipótesis si se considera el número de muestras de entrenamiento utilizadas en ambos modelados.

Experimento	Umbral	Normal				Ataque			
		FP		Sin Clasificar		Sin Detectar		Sin Clasificar	
		N.	%	N.	%	N.	%	N.	%
PVH vs. RDB	3,0	0	0.00	12.425	2,01	2	0,22	243	26,15
PVH vs. OSVDB	1,0	2	3.10^{-4}	12.369	2,01	650	9,42	134	1,94
	2,0	2	3.10^{-4}	12.649	2,05	61	0,88	1089	15,79
	3,0	2	3.10^{-4}	23.537	3,82	14	0,20	1689	24,50

Tabla 5.2: Rendimiento del reconocedor con umbrales de clasificación

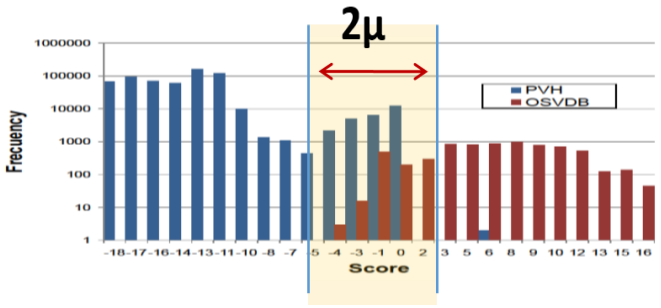


Figura 5.5: Umbral de clasificación para la zona de confusión

Los resultados experimentales obtenidos (Figura 5.7) muestran una mejora con respecto al sistema SSM original. Por otra parte, en comparación con el reconocedor (sección anterior), esta variante no sólo mejora el rendimiento sino que también permite la selección del punto de operación del detector a través de la elección de los parámetros μ (umbral de confusión) y Θ (umbral de detección). La complejidad introducida en la detección no es relevante y, adicionalmente, a partir del índice de confianza, se podría establecer una confianza en la clasificación en forma de probabilidad de clasificación correcta. Para ello bastaría establecer una transformación a partir del índice de confianza mediante, por ejemplo, una función sigmoide.

Como se puede observar en la Figura 5.7.a), el comportamiento del sistema híbrido para PVHDB / RDB supera el 99,5% de detección con una tasa de falsos positivos inferior a 0,1%, mientras que el sistema de referencia necesita de una tasa de falsos positivos superior al 0,2% para conseguir rendimientos similares. Mejores resultados se observan en el caso de PVHDB / OSVDB, en los que la tasa de detección alcanza el 100% con sólo un 0,05% de falsos positivos. En particular, este experimento constituye un indicio relevante respecto de la bondad del sistema híbrido, ya que el tamaño de la base de datos de ataques es superior para OSVDB que para RDB, por lo que es de suponer que el modelado realizado sea más fidedigno. Esto es coherente con el razonamiento que justifica la utilización del sistema de detección basado en umbral en caso de bajo índice de confianza.

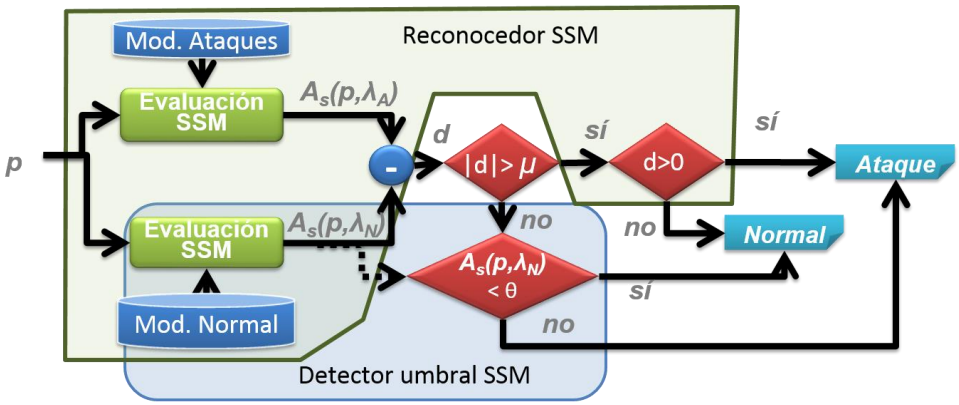


Figura 5.6: Diagrama del detector híbrido basado en el sistema SSM

Una vez mejorado el rendimiento del sistema mediante diversas aproximaciones, resulta conveniente realizar un proceso de validación que permita comprobar de manera fehaciente la mejora obtenida. Por tanto, a continuación se procede a validar el sistema híbrido propuesto.

5.3.1 Validación del sistema híbrido

Con el objetivo de corroborar el comportamiento y actuación del sistema híbrido propuesto, a continuación se realizan y analizan experimentos de validación mediante la utilización de las particiones establecidas al efecto. En la Figura 5.8 se comparan los resultados obtenidos para la partición de validación de PVHDB para el sistema de referencia y el sistema híbrido con ambas bases de datos de ataques. En el caso de RDB — Figura 5.8.a) —, la tasa de falsos positivos obtenida es inferior al 0,5% para el sistema híbrido, mientras que es superior al 5% para el sistema de referencia, para alcanzar una tasa de detección superior al 95%, lo que supone una clara mejora del comportamiento del sistema. Análogamente, para OSVDB — Figura 5.8.b) —, se puede observar que en ambos casos se alcanza el 100% de detección aunque con diferentes tasas de falsos positivos. En particular, esta tasa se reduce del 5% para el sistema de referencia a menos del 0,2% para el sistema híbrido.

A la vista de los resultados, se concluye que el rendimiento, en términos de detección y falsos positivos, del sistema híbrido propuesto es claramente superior al del sistema original basado en la detección de anomalías mediante umbral.

5.4 Entrenamiento discriminativo

El nuevo enfoque utilizado para el detector híbrido no cambia los fundamentos de la técnica utilizada, que es SSM en ambos casos, sino su uso para modelar tanto los ataques como las instancias normales. Por lo tanto, se usan dos modelos (ataque y normal) como base de un reconocedor, lo que hace de este un IDS híbrido. En

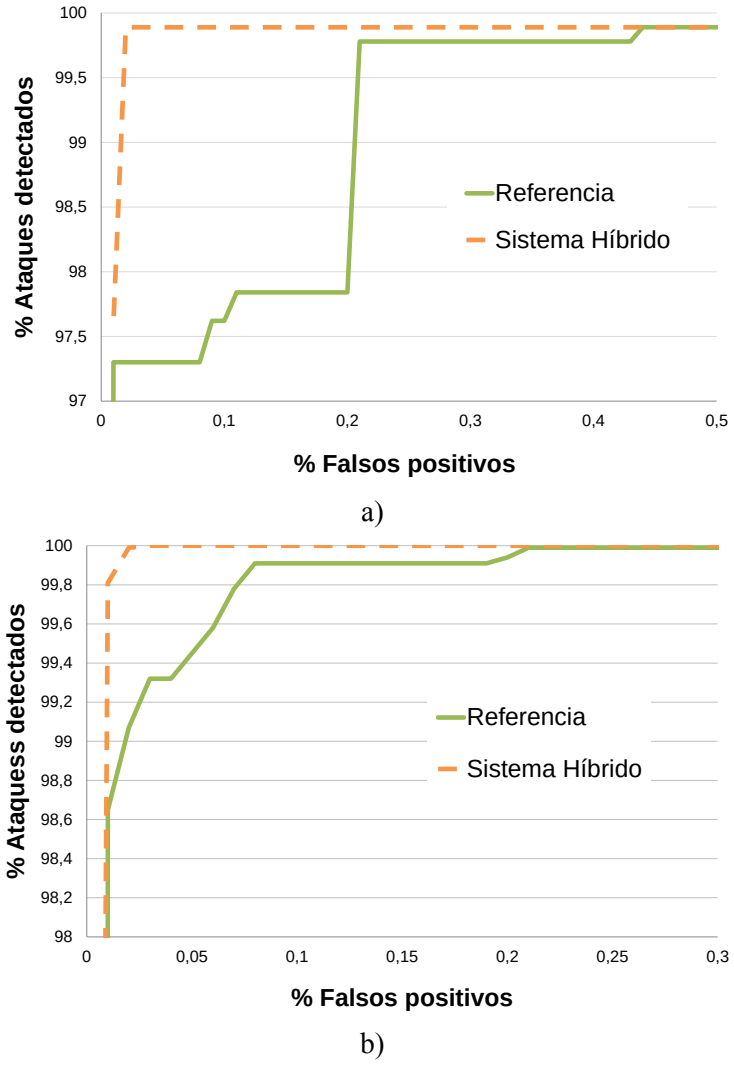
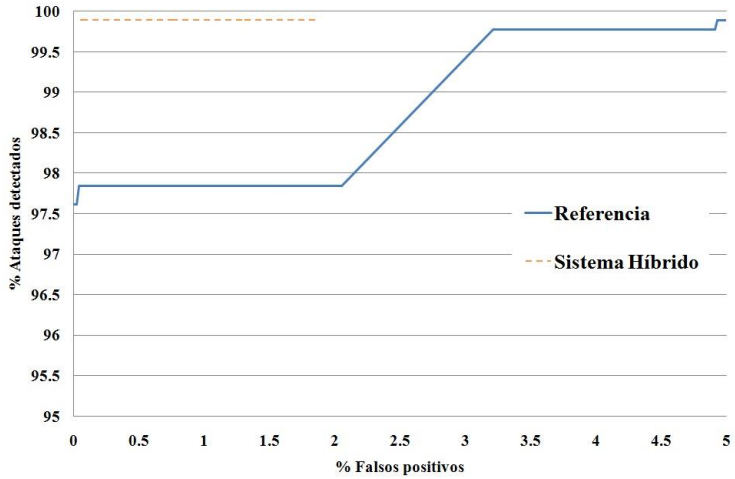


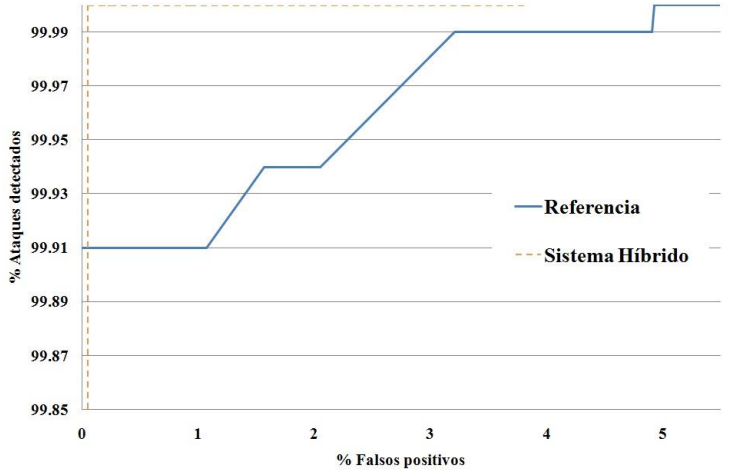
Figura 5.7: Comparación de los resultados para el sistema básico SSM y el sistema híbrido propuesto: a) PVH / RDB b) PVH / OSVDB

consecuencia, la capacidad de detección combina las habilidades de un S-IDS y un A-IDS. Así, los ataques conocidos pueden ser explícitamente modelados y, en su caso, detectados. Por otra parte, dada la capacidad de generalización asociada al uso de modelos, es razonable asumir que algunas de las nuevas variantes de dichos ataques podrán ser también detectadas. Adicionalmente, el sistema tiene la capacidad de detectar nuevos ataques a partir del modelo de normalidad, ya que es de esperar que estos sean diferentes de las instancias normales y, consecuentemente, sus índices de normalidad serán reducidos.

Como aportación adicional, el sistema híbrido también es capaz de proporcionar una medida de la confianza de la clasificación, a partir del índice de confianza. Esta



a)



b)

Figura 5.8: Resultados de validación para el sistema de referencia y el sistema híbrido propuesto: a) PVH / RDB, b) PVH / OSVDB

medida se puede utilizar para alertar al administrador del sistema o para realizar un análisis más pormenorizado de las cargas útiles con valores bajos de dicho índice. Aunque esta posibilidad resulta altamente interesante, en el presente trabajo nuestro interés se centra en mejorar el modelado. En este sentido, el índice de confianza puede ser utilizado para reentrenar los modelos mediante la estimación de correcciones a partir de las observaciones de clasificación dudosa, esto es, de aquellas que tengan un bajo valor del índice de confianza. Por tanto, el reentrenamiento se realiza considerando el índice de confianza como función a maximizar, en lugar de maximizar la probabilidad de observación proporcionada por el modelo correcto en cada caso. De esta forma, cada observación influirá en los valores de todos los modelos, aunque en sentido diferente

(maximizar o minimizar la probabilidad de generación asociada) y en una proporción que depende de las diferencias entre la probabilidad correcta y la/s errónea/s. Este tipo de aproximación se encuentra ampliamente descrito en la bibliografía [Collins, 2002], denominándose *entrenamiento discriminativo,* ya que debe incrementar la capacidad discriminativa de los modelos, al utilizar como función objetivo una medida relacionada con la confusión entre las categorías consideradas e intentar maximizar la diferencia entre la probabilidad proporcionada por el modelo correcto y las restantes.

5.4.1 Reentrenamiento discriminativo del sistema híbrido

De acuerdo a lo anterior, a continuación se evalúa la aplicación de las técnicas de entrenamiento discriminativo al sistema híbrido. En particular, se considerarán correcciones a los modelos inicialmente entrenados mediante la aproximación clásica de máxima probabilidad a partir de las observaciones próximas a la zona de confusión, es decir, al límite de decisión entre ambos. En este caso, por tanto, se reestimarán los modelos mediante la introducción de modificaciones aditivas en sus parámetros. Estas dependerán únicamente de las observaciones que calificaremos como dudosas, esto es, de aquellas que quedan dentro de la zona de confusión según el umbral de confusión considerado.

Por tanto, dado un modelo inicial, λ, y un conjunto de observaciones a utilizar para el reentrenamiento, *D*, se obtendrá un nuevo modelo, λ', a partir de la interpolación del modelo original y el correspondiente a las observaciones de reentrenamiento, λ_D, según:

$$\lambda' = \lambda + \alpha \cdot \lambda_D \tag{5.8}$$

siendo α un parámetro de ponderación que determina el grado de las modificaciones a efectuar a partir del nuevo modelo. El valor de este parámetro debe depender, a priori, de la confianza estadística en el nuevo modelo frente al antiguo, lo que, a su vez, depende del número de secuencias utilizadas para entrenar ambos modelos.

El entrenamiento discriminativo propuesto resultaría en un proceso iterativo (Figura 5.9) en el que, tras reestimar el modelo, se determina cuáles de las observaciones continúan en la zona de confusión y se vuelve a reestimar el modelo a partir de ellas.

Implementación de la técnica de re-entrenamiento

En el caso considerado, se establecen inicialmente los modelos de normalidad, λ_N, y de ataques, λ_A, a partir de todas las cargas útiles en las particiones de entrenamiento. Mediante dichos modelos y, utilizando el reconocedor, se evalúan los índices de anomalía de una partición de evaluación previamente etiquetada y se determinan los conjuntos de instancias dudosas en cada categoría, es decir:

$$\begin{aligned} D_N &= \{p \in N / S(p) < -\mu\} \\ D_A &= \{p \in A \,/\, S(p) < \mu\} \end{aligned} \tag{5.9}$$

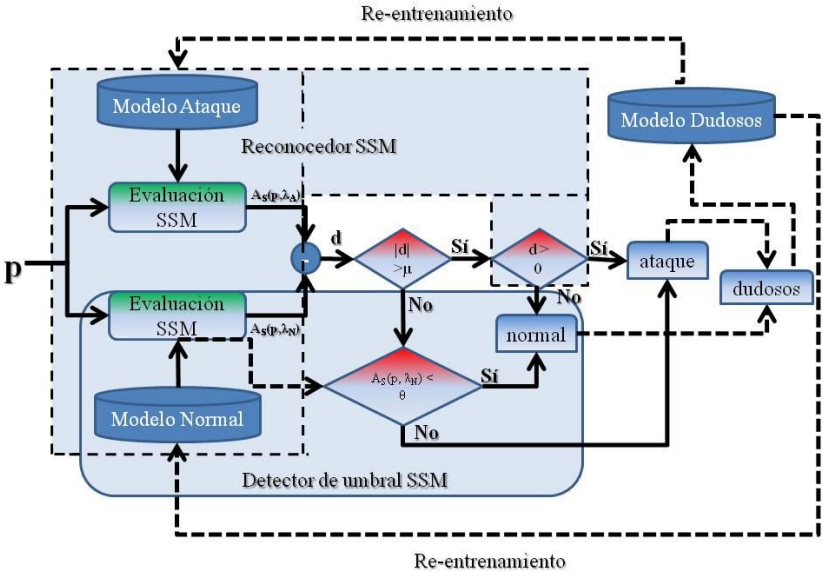


Figura 5.9: Esquema del procesamiento para el reentrenamiento de los modelos a partir de las instancias "dudosas"

siendo D_N y D_A, respectivamente, los conjuntos de cargas normales y de ataques de clasificación dudosa y N y A los conjuntos de evaluación normal y de ataques, respectivamente.

A partir de estos conjuntos de peticiones dudosas se obtienen los modelos correspondientes, λ_{D_N} y λ_{D_A}, según el algoritmo de entrenamiento habitual, esto es, según la Ec. (3.11). Por tanto, el nuevo modelo se obtendrá a partir de la adición, con un factor de ponderación, de los valores obtenidos en cada caso para el modelo de dudosos y la posterior normalización de las probabilidades. Dado que las probabilidades de transición se encuentran fijadas por la especificación del protocolo, únicamente se reestiman las probabilidades de observación, b_{ij}, en cada uno de los modelos, λ_{D_N} y λ_{D_A}.

A este respecto hemos de tener en cuenta que los vocabularios en el modelo de dudosos y el modelo original pueden ser diferentes, lo que, por otra parte, será bastante probable. Por tanto, será necesario que, durante la actualización de los valores, se establezca la correspondencia entre los índices en ambos vocabularios. Así, denotemos mediante V_i^{λ} al vocabulario asociado al estado *i* del modelo λ. Definimos $M_i^{\lambda_1\lambda_2}()$ como la función que realiza el mapeo del vocabulario del modelo λ_2 al del modelo λ_1 para el estado *i*:

$$M_i^{\lambda_1\lambda_2}(\quad): V_i^{\lambda_2} \rightarrow V_i^{\lambda_1}$$
$$M_i^{\lambda_1\lambda_2}(j) = k \Leftrightarrow V_i^{\lambda_2}[j] = V_i^{\lambda_1}[k] \tag{5.10}$$

Sin introducir ninguna diferencia desde el punto de vista teórico, en la práctica podemos diferenciar dos escenarios para cada una de las probabilidades de observación obtenidas para el modelo de dudosos, $b_{ij}^{\lambda_D}$. En el primero de ellos, la palabra asociada, $V_i^{\lambda}[\mathrm{j}]$, ya estaba incluida en el diccionario correspondiente a dicho estado en el modelo inicial, por lo que únicamente se sumará el valor obtenido para el modelo de dudosos, considerando el factor de ponderación, al valor del modelo inicial. En este caso, según se esté considerando el modelo normal o de ataque, tendremos:

$$\begin{aligned} Normal\colon b'_{iM_i^{\lambda_N\lambda_{D_N}}(j)} &= b_{iM_i^{\lambda_N\lambda_{D_N}}(j)} + \alpha \cdot b_{ij}^{\lambda_{D_N}},\ \ si\ \exists M_i^{\lambda_N\lambda_{D_N}}(j) \\ Normal\colon b'_{iM_i^{\lambda_A\lambda_{D_A}}(j)} &= b_{iM_i^{\lambda_A\lambda_{D_A}}(j)} + \alpha \cdot b_{ij}^{\lambda_{D_A}},\ \ si\ \exists M_i^{\lambda_A\lambda_{D_A}}(j) \end{aligned} \tag{5.11}$$

En el segundo escenario, dicha palabra no está incluida en el diccionario original para ese estado, por lo que será necesario incluirla y, de nuevo teniendo en cuenta el valor del factor de ponderación, asignarle el valor del modelo de dudosos. Así:

$$Normal\colon \begin{cases} k = \ card\left(V_i^{\lambda_N}\right) + 1 \\ V_i^{\lambda_N}[k] = V_i^{\lambda_{D_N}}[j];\ M_i^{\lambda_N\lambda_{D_N}}(j) = k \\ b'_{ik} = \alpha \cdot b_{ij}^{\lambda_{D_N}} \end{cases}$$

$$Ataque\colon \begin{cases} k = \ card\left(V_i^{\lambda_A}\right) + 1 \\ V_i^{\lambda_A}[k] = V_i^{\lambda_{D_A}}[j];\ M_i^{\lambda_A\lambda_{D_A}}(j) = k \\ b'_{ik} = \alpha \cdot b_{ij}^{\lambda_{D_A}} \end{cases} \tag{5.12}$$

Una vez actualizados todos los valores a partir del modelo de dudosos, deberá procederse a la normalización de las probabilidades, según la Ec. (3.6). De esta forma, la incorporación de la información procedente del modelo de dudosos modifica todas las probabilidades de observación, independientemente de que las palabras correspondientes se encuentren incluidas o no en el modelo de dudosos.

A la vista de la implementación realizada para el entrenamiento discriminativo, podemos concluir que las probabilidades de aquellas palabras observadas en las instancias dudosas se verán incrementadas en el modelo resultante correspondiente a la categoría de la observación, mientras que las restantes probabilidades se verán minoradas. Por otra parte, hemos de indicar que no se ha contemplado la disminución de las probabilidades de dichas palabras ni, obviamente, su inclusión en el caso de que no existiesen, en los modelos correspondientes a la categoría contraria. Aunque esta forma de actuar sería la indicada en el caso general de entrenamiento discriminativo, la naturaleza del problema considerado, en el que es de esperar que haya una gran diferenciación entre los vocabularios en los modelos de ataque y normal, aconseja modificar el procedimiento en el sentido realizado.

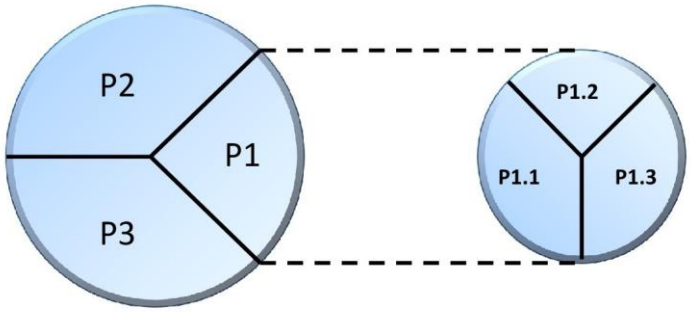


Figura 5.10 Esquema de reparticionado de las bases de datos usadas en experimentación de reentrenamiento en el caso de la partición de evaluación P1

Otra consideración práctica está relacionada con la posibilidad de que alguna de las probabilidades reestimadas tenga un valor inferior a la probabilidad OOV considerada. Esta situación sería posible en función del valor de α considerado y del valor de dicha probabilidad en el modelo de dudosos. En este caso, para evitar la incoherencia resultante, ya que se le asignaría menor valor a una palabra incluida en el vocabulario que a otra no incluida, se establece la probabilidad al valor OOV.

5.4.2 Experimentación con reentrenamiento discriminativo

A continuación se describe la preparación de las particiones correspondientes de las bases de datos de tráfico normal PVHDB y de ataques RDB con el fin de evaluar la técnica de re-entrenamiento de los modelos propuesta en este apartado. En este sentido, se han establecido nuevas subparticiones de las particiones previamente establecidas para posibilitar la evaluación de la aproximación de reentrenamiento en las condiciones adecuadas. Así, de acuerdo al procedimiento previamente descrito, se necesitará una partición de entrenamiento y una de evaluación, que será utilizada para reentrenar los modelos. Por tanto, para evaluar las mejoras obtenidas será necesaria otra nueva partición de evaluación que no haya sido utilizada previamente. De esta forma, y haciendo nuevamente uso de la técnica *leaving-one-out*, se han establecido tres subparticiones de similar tamaño en cada una de las particiones a utilizar durante la evaluación del sistema (Figura 5.10), tanto para el tráfico normal como el de ataques.

Usando estas tres subparticiones a partir de la partición de evaluación en cada caso original, se procede iterativamente utilizando la primera de ellas para evaluar los modelos y estimar los correspondientes modelos de dudosos. Una vez obtenido los modelos de dudosos, se reestiman los modelos y se vuelve a iterar la evaluación y reentrenamiento con la segunda partición, para realizar una última evaluación con la tercera partición. Así, a modo de ejemplo, la secuencia a seguir para el caso de usar la partición L1 de PVHDB como partición de evaluación se muestra en la Tabla 5.3. En este caso, para el modelo normal, se parte de las particiones L2 y L3 como particiones

Paso	Particiones (re)entrenamiento	Evaluación	Re-entrenamiento
A	L2 ∪ L3	P1.1	$D_N1.1$
	S2 ∪ S3	S1.1	$D_A1.1$
B	L2 ∪ L3 ∪ $D_N1.1$	P1.2	$D_N1.2$
	S2 ∪ S3 ∪ $D_A1.1$	S1.2	$D_A1.2$
C	L2 ∪ L3 ∪ $D_N1.1$∪ $D_N1.2$	P1.3	
	S2 ∪ S3 ∪ $D_A1.1$ ∪ $D_A1.2$		

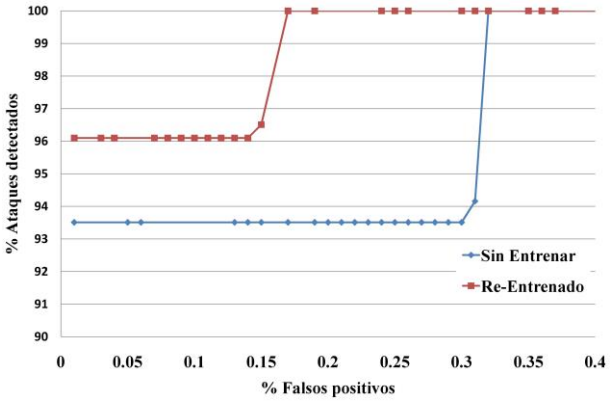
Tabla 5.3: Secuencia de particiones y modelos usados para el re-entrenamiento discriminativo en el caso de particiones de entrenamiento iniciales L2 y L3

de entrenamiento, mientras que se usan las subparticiones S2 y S3 para el modelo de ataques. El modelo resultante se evalúa con las subparticiones 1.1 de L1 (normal) y de S1 (ataques), a partir de los que se obtienen los modelos de dudosos normal y de ataques con los que se reestiman los modelos originales, según las Ecs. (5.11) y (5.12). Estos nuevos modelos son evaluados con la segunda subpartición de evaluación, 1.2, obteniéndose nuevos modelos de dudosos con los que se vuelve a reestimar los modelos. Finalmente, los modelos así obtenidos se evalúan con las subparticiones 1.3. Análogo procedimiento se aplica, de acuerdo al procedimiento *leave-one-out* utilizado, para los dos restantes experimentos considerando las combinaciones de particiones de entrenamiento.

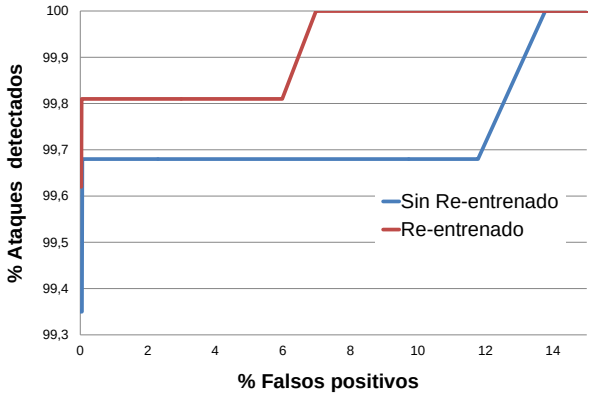
Para contrastar los resultados experimentales de esta aproximación, se usan todas las subparticiones de evaluación con los modelos originales y los modelos reentrenados. Los resultados obtenidos para cada partición de evaluación se muestran en la Figura 5.11. En estos se observa una mejora en el rendimiento del sistema tras el reentrenamiento en los dos primeros escenarios. En el tercero (particiones L3 y S3 para evaluación) no se observa dicha mejora dado que los resultados ya eran excelentes. Así, para el primer experimento, se obtiene una tasa de detección superior al 97% para tasas de FP reducidas y claramente inferiores a las del sistema original. Asimismo, se alcanza la tasa máxima de detección para una tasa de FP de 0,15%, que resulta ser la mitad de la obtenida con el sistema original. En estos casos, el área de confusión en la primera iteración incluía alrededor del 2% del tráfico normal y el 39% del tráfico de ataques. En el caso de las particiones L2 y S2 (segundo experimento) se constata también una clara mejora en los resultados al aplicar el reentrenamiento, reduciéndose también a la mitad (de 13,7% a 6,98%) la tasa de FP necesaria para alcanzar el máximo rendimiento, que es también del 100%.

5.4.3 Validación del re-entrenamiento

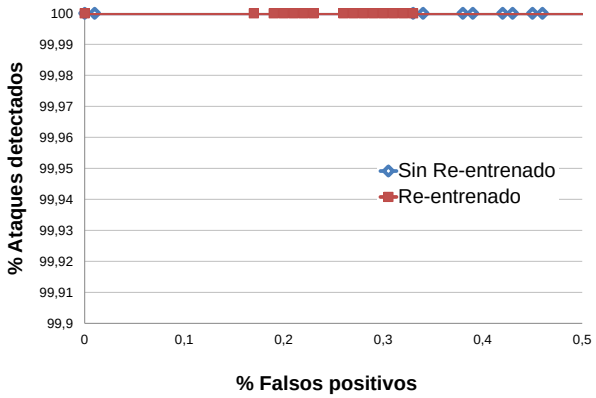
Finalmente, se validan los resultados anteriores utilizando la partición de validación de PVHDB, junto con la base de datos RDB, para evaluar el rendimiento del sistema.



a)



b)



c)

Figura 5.11: Curvas ROC para PVHDB / RDB con y sin reentrenamiento usando las particiones de evaluación: a) L1 y S1, b) L2 y S2, c) L3 y S3

Los resultados obtenidos para las tres combinaciones de particiones de entrenamiento inicial se muestran en la Figura 5.12. Así, en el primer experimento se alcanza un 100% de ataques detectados con una tasa del 0,04 % de falsos positivos tras reentrenar los modelos, frente a una tasa de falsos positivos inicial de 0,09%.

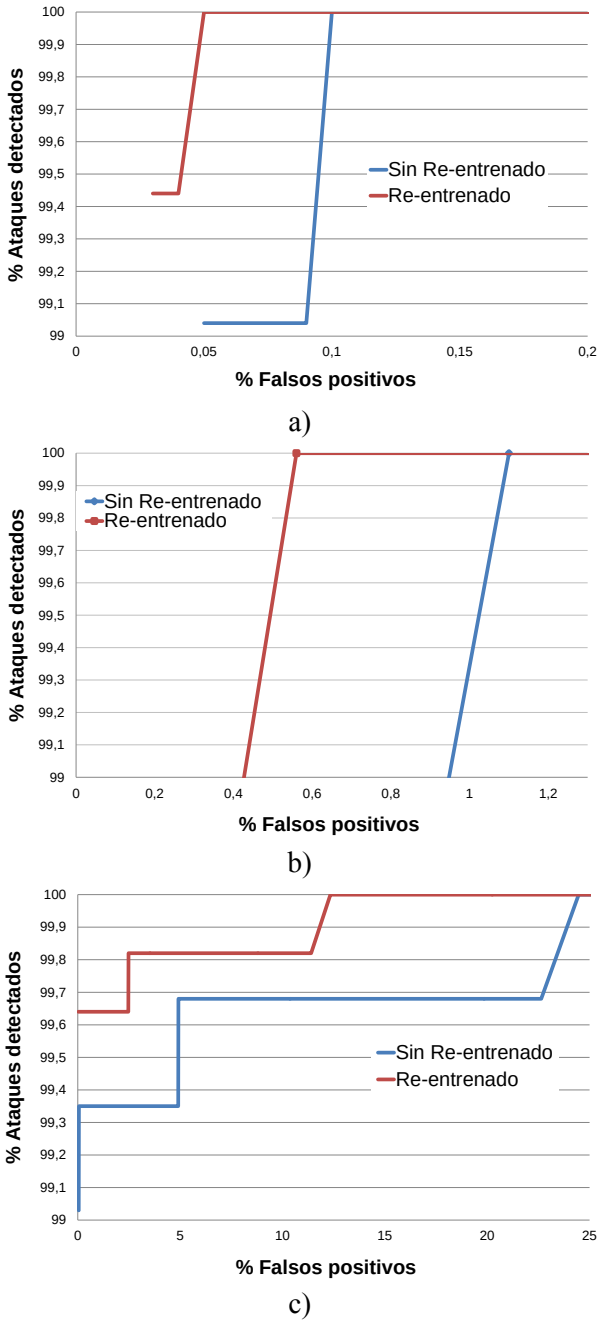


Figura 5.12 Curvas ROC para la partición de validación con / sin re-entrenamiento para PVHDB / RDB. a) Particiones de entrenamiento 1 y 2, b) ídem 2 y 3, c) ídem 1 y 3

Análogamente, en el segundo experimento se alcanza el 100% de detección con un 12,3% de FP para el sistema reentrenado y de 24,8% para el sistema inicial. Por último, en el tercer experimento también se alcanza el 100% de detección para un 0,5% y para un 1% para los modelos reentrenados y sin reentrenar, respectivamente. Por tanto, como

ocurría en la evaluación inicial, se observa una reducción en torno al 50% en la tasa de falsos positivos tras reentrenar el sistema.

Por tanto, a la vista de los resultados experimentales obtenidos, tanto en la evaluación inicial como en la validación, podemos concluir la bondad del reentrenamiento propuesto de los modelos para el sistema híbrido en base a considerar las cargas útiles que quedan dentro de la zona de confusión.

6 Conclusiones y trabajo futuro

Se describen a continuación las conclusiones y aportaciones del presente trabajo, así como las líneas futuras de actuación que pueden dar seguimiento a la investigación realizada.

6.1 Conclusiones

Se han propuesto y evaluado modificaciones al sistema de detección de intrusos basado en red y en anomalías denominado *Stochastic Structural Model* (SSM). Para ello ha sido necesario implementar un sistema de referencia siguiendo la arquitectura de dicho detector.

A fin de posibilitar el desarrollo y evaluación de las propuestas de mejora, así como establecer el marco con el que comparar los rendimientos obtenidos, se ha establecido un entorno experimental que incorpora diversas bases de datos de tráfico normal y de instancias de ataques.

En este sentido, se han evaluado las ventajas y desventajas de los diversos escenarios y las bases de datos disponibles para la evaluación del comportamiento de los sistemas de detección de intrusos, llegándose a la conclusión de su inadecuación para la experimentación a realizar.

En consecuencia, se ha desarrollado y aplicado una metodología para la adquisición y preparación de las bases de datos de tráfico necesarias, incluyendo el particionado de las mismas para posibilitar el entrenamiento, la evaluación y la validación final del sistema.

Aplicando dicha metodología, se ha capturado tráfico real en varias redes de computadoras en explotación mediante la colocación de sensores ubicados estratégicamente para tal efecto. Este tráfico ha sido procesado y acondicionado para su uso como base de datos de tráfico limpio, esto es, libre de ataques. Para ello se han desarrollado las herramientas y algoritmos necesarios, incluyendo filtros para eliminar los ataques detectados mediante detectores de intrusos basados en firmas, de acuerdo a la metodología propuesta.

La adquisición de las instancias de tráfico de ataques se ha realizado en un entorno controlado, habiéndose evaluado diversos métodos que permitan automatizar y

sistematizar en la medida de lo posible la generación de los mismos. Finalmente, y a partir de la combinación de diversos métodos y fuentes de información, se han generado dos bases de datos de tráfico de ataques con un elevado número de instancias de los mismos e incluyendo variantes de todos los ataques basados en web descritos hasta la fecha. Durante este proceso se han recopilado, seleccionado y adaptado los conjuntos de reglas VRT de Snort de interés para la detección de ataques basados en HTTP.

Asimismo, se ha desarrollado un método que permite anonimizar el tráfico real a fin de ocultar la información sensible contenida en él, sin pérdida de la información necesaria para la técnica de detección aplicada. Esta técnica posibilita la compartición de las bases de datos obtenidas con otros equipos de investigación sin invadir la privacidad de las comunicaciones.

Usando las bases de datos adquiridas, se evaluó el comportamiento del método de referencia, esto es, de SSM, obteniéndose resultados parcialmente satisfactorios debido a algunas limitaciones de la técnica relacionadas, especialmente, con su implementación efectiva. En particular, el comportamiento de SSM en el caso de grandes vocabularios resulta insuficiente, por lo que se han propuesto y evaluado dos modificaciones relacionadas con la gestión del vocabulario y el suavizado de las probabilidades de observación, tanto para palabras incluidas en el vocabulario como para las observaciones fuera de vocabulario. Así, además de la gestión del vocabulario de forma independiente por estado, se ha propuesto y justificado el uso de probabilidades de fuera de vocabulario diferenciadas por estado. Estas propuestas han proporcionado resultados satisfactorios que mejoran los proporcionados por el sistema de referencia.

A partir del esquema original del detector SSM, se ha propuesto una nueva arquitectura para el sistema mediante el modelado tanto de las instancias normales como de las de ataque mediante el modelado SSM. En este esquema se combina la detección basada en anomalías con el modelado explícito de los ataques, lo que puede considerase una detección basada en firmas. Consecuentemente, el sistema propuesto es un IDS híbrido.

Utilizando las probabilidades obtenidas por ambos modelos, se ha establecido un índice de confianza que permite definir una zona de confusión, asociada a los errores de clasificación. Mediante la aplicación de un reentrenamiento discriminativo usando las instancias en la zona de confusión, se ha conseguido mejorar el rendimiento del sistema, probándose la idoneidad de la arquitectura y método propuestos.

Los resultados obtenidos de todas las propuestas realizadas han sido validados mediante la utilización de particiones de validación de las bases de datos que no habían sido utilizadas previamente.

6.2 Líneas Futuras

Las modificaciones propuestas para la aplicación de la técnica SSM posibilitan su uso en entornos reales en explotación sin las limitaciones previas relacionadas con los tamaños de los vocabularios incluidos en el modelo. Adicionalmente, la arquitectura

híbrida propuesta permite mejorar aún más el rendimiento del sistema. Sin embargo, las limitaciones de las bases de datos utilizadas, en especial, las de las bases de datos de ataques, impiden la evaluación efectiva de propuestas que potencialmente pueden proporcionar mejores resultados. En particular, el número de ataques incluidos en la base de datos correspondiente resulta insuficiente para el entrenamiento de diversos modelos de ataque dependiendo de su tipo o naturaleza, así como de la implementación de un reentrenamiento discriminativo que modifique las probabilidades en todos los modelos. Así, el modelado de ataques realizado resulta algo genérico ya que debe incorporar todos los existentes.

En consecuencia, una de las líneas futuras a desarrollar consiste en aplicar la metodología para la obtención de mayores volúmenes de tráfico en entornos reales a fin de obtener tráfico con una mayor representatividad y que permita explorar el uso de diversos modelos tanto de normalidad como de ataque. En este sentido, se hace necesario sistematizar y automatizar en mayor medida la adquisición de instancias de ataque a fin de aumentar su volumen.

El coste computacional del modelado SSM, tanto en su propuesta original como en el sistema híbrido desarrollado, no resulta excesivo para su utilización efectiva en entornos reales en explotación. Por tanto, una actuación a realizar es la implementación del sistema como módulo para su incorporación en detectores de uso extendido, como Snort o Bro, o directamente en servidores de páginas web como Apache.

Finalmente, la técnica SSM permite la detección y discriminación de los segmentos anómalos en los URI analizados, por lo que se pueden desarrollar generadores automáticos de firmas para su utilización en detectores basados en firmas. Esta aproximación ya ha dado lugar a algún desarrollo con otros investigadores que ha sido recientemente publicado, aunque aún tiene suficiente recorrido para una investigación más profunda. En particular, puede resultar muy interesante la generación de firmas en el caso del sistema híbrido propuesto, lo que, nuevamente, requiere de bases de datos de tráfico adecuadas.

Bibliografía

Aharoni, M., 2007. *Bactrack Linux*. [Online] Available at: http://www.backtrack-linux.org.

Al-Jarrah, O. & Arafat, A., 2015. Network Intrusion Detection System Using Neural Network Classification of Attack Behavior. *Journal of Advances in Information Technology*, 6(1), pp.1-8.

Altwaijry, H. & Algarny, S., 2011. Multi-layer bayesian based intrusion detection system. In *Proc. World Congress on Engineering and Computer Science 2011*., 2011.

Anderson, J.P., 1980. *Computer security threatmonitoring and surveillance*. James P. Anderson Co.

Anderson et al., 1994. *NIDES: Software Users Manual*.

Anderson et al., 1995. *Detecting Unusual Program Behavior Using the Statistical Components of NIDES*. SRI International.

Androutsopoulos, I., Paliouras, G. & Michelakis, E., 2006. *Learning to Filter Unsolicited Commercial E-Mail*. National Centre for Scientific Research - Demokritos.

Anon., 2006. *VMware, Inc.* [Online] Available at: www.vmware.com.

Antonatos, S., Anagnostakis, K.G. & Markatos, E.P., 2004. Generating realistic workloads for network intrusion detection systems. In *Proc. 4th international workshop on Software and performance (WOSP 04)*., 2004.

ArachNids, 2003. *Advanced Reference Archive of Current Heuristics for Network Intrusion Detection Systems*. [Online] Available at: http://www.whitehats.com/ids/ [Accessed 2008].

Asenova, D., Bailey, S.J. & McCann, C., 2015. Public sector risk managers and spending cuts: mitigating risks. *Journal of Risk Research*, 18(5), pp.552-65.

Athanasiades, N. et al., 2003. Intrusion detection testing and benchmarking methodologies. In *Proc. First IEEE International Workshop on Information Assurance (IWIAS 2003)*., 2003.

Ax3soft, C., 2010. *Sax2 Expert NIDS*. [Online] Available at: http://www.ids-sax2.com.

Axelsson, S., 1998. *Research in Intrusion-Detection Systems: A Survey*. Chalmers University of Technology.

Aydin, M.A., Zaim, A.H. & Ceylan, K.G., 2009. A hybrid intrusion detection system design for computer network security. *Comput. Electr. Eng.*, 35(3), pp.517-26.

Bankovic, Z. et al., 2009. A Genetic Algorithm-based Solution for Intrusion Detection. *Journal of Information Assurance and Security*, 4, pp.192-99.

Bay, S.D., Kibler, D., Pazzani, M.J. & Smyth, P., 2000. The UCI KDD archive of large data sets for data mining research and experimentation. *ACM SIGKDD Explorations Newsletter*, 2(2), pp.81-85. http://kdd.ics.uci.edu.

Bermúdez-Edo, M., Salazar-Hernández, R., Díaz-Verdejo, J. & Gacía-Teodoro, P., 2006. Proposals on Assessment Environments for Anomaly-Based Network Intrusion Detection Systems. In *Proc. Critical Information Infrastructures Security*., 2006.

Berners-Lee, T., 1998. *Uniform Resource Identifiers (URI) - RFC 2396*. IEFT.

Berners-lee, T., Fielding, R. & Frustuk, H., 1996. *Hypertext Transfer Protocol - HTTP/1.0 - RFC 1945*. IETF.

Berners-Lee, T., Fielding, R. & Masinter, L., 2005. RFC 3986 *Uniform Resource Identifier: Generic Sintax (RFC 3986)*. IETF.

Biles, S., 2001. *Detecting the unknown with Snort and Statistical Packet Anomaly Detection Engine (SPADE)*. Computer Security Online Ltd.

Biskup, J. & Flegel, U., 2009. On Pseudonymization of Audit Data for Intrusion Detection. In *Designing Privacy Enhancing Technologies*. pp.161-80.

Bivens, A. et al., 2002. Network-based Intrusion Detection using Neural Networks. *Intelligent Engineering Systems Through Artificial Neural Networks*, 12, pp.579-84.

Bridges, S.M., Vaughn, R.B., Professor, A. & Professor, A., 2000. Fuzzy Data Mining And Genetic Algorithms Applied To Intrusion Detection. In *Proc. National Information Systems Security Conference (NISSC 00)*., 2000.

Brookshear, G.J., 1989. *Theory of computation: formal languages, automata, and complexity*. 1st ed. Benjamin-Cummings Publishing Co., Inc.

Bugtraq, 2005. *Security focus online*. [Online] Available at: http://www.securityfocus.com [Accessed 2015].

Cansian, A.M., Moreira, E., Carvalho, A. & Bonifacio, J.M., 1997. Network intrusion detection using neural networks. In *International Conference on Computational Intelligence and Multimedia Applications*., 1997. ICCIMA'97.

Carracedo, J., 2004. *Seguridad en redes telemáticas*. Mc. Graw Hill.

CERT, 1988. *Computer Emergency Response Team*. [Online] Available at: http://cert.org.

Chakrabarti, A. & Manimaran, G., 2002. Internet infrastructure security: a taxonomy. *IEEE Network*, 16, pp.13-21.

Ching, W.-K. & Ng, M.K., 2005. *Markov Chains: Models, Algorithms and Applications*. 1st ed. Springer.

Chittur, A., 2001. *Model Generation for an Intrusion Detection Systems using Genetic Algorithm*. Ossining High School.

Cisco Systems Inc., 1998. *Cisco Netranger*. [Online] Available at: http://www.cisco.com.

Cole, E., 2005. *Network Security Bible*. John Wiley & Sons.

Cole, E., Kruts, R.L. & Conley, J., 2009. *Network Security Biblie*. 2nd ed. John Wiley & Sons.

Collins, M., 2002. Discriminative training methods for hidden Markov models: theory and experiments with perceptron algorithms. In *Proc. of the ACL-02 conference on Empirical methods in natural language processing*., 2002.

Debar, H., 2002. An Introduction to Intrusion-Detection Systems. In *Proceedings of Connect'2000*., 2002.

Debar, H., Dacier, M. & Wespi, A., 1999. Towards a taxonomy of intrusion-detection systems. *Computer Networks*, 31, pp.805-22.

Denning, D.E. & Neumann, P., 1985. *Requirements and model for IDES - A real-time intrusion detection expert system*. SRI International.

Departament of Defense, 1985. *Trusted Computer System Evaluation Criteria "ORANGE BOOK"*. Departament of Defense.

Depren, O., Topallar, M., Anarim, E. & Ciliz, M., 2005. An intelligent intrusion detection system (IDS) for anomaly and misuse detection in computer networks. *Expert Systems with Applications*, 29, pp.713-22.

Díaz-Verdejo, J.E. et al., 2007. Una aproximación basada en Snort para el desarrollo e implantación de IDS híbridos. *IEEE Latin America Transactions* , 5, pp.386-92.

Díaz-Verdejo, J., Segura-Luna, J.C. & Rubio-Ayuso, A., 2002. *Reconocimiento de Voz Continua. Aproximaciones basadas en HMM y en redes neuronales recurrentes.* Universidad de Granada.

Dickerson, J.E. & Dickerson, J.A., 2000. Fuzzy network profiling for intrusion detection. In *Proc. 19th Int. Conf. North American Fuzzy Information Processing Society (NAFIPS 00)*., 2000.

Ding, Y.-X., Xiao, M. & Ai-Wu, L., 2009. Research and implementation on snort-based hybrid intrusion detection system. In *Proc. 2009 International Conference on Machine Learning and Cybernetics*., 2009.

Duda, R. & Hart, P., 1973. *Pattern Classification and Scene Analysis*. John Wiley & Sons.

Egan, J.P., 1975. *Signal Detection Theory and ROC analysis*. Academic Press.

Estevez-Tapiador, J.M., 2004a. *Detección de intrusiones en redes basada en anomalías mediante técnicas de modelado de protocolos*. Tesis doctoral. Universidad de Granada.

Estevez-Tapiador, J.M., Garcia-Teodoro, P. & Diaz-Verdejo, J.E., 2003. Stochastic protocol modeling for anomaly based network intrusion detection. In *Proc. First IEEE Int. Workshop on Information Assurance 2003 (IWIAS 2003)*., 2003.

Estévez-Tapiador, J.M., García-Teodoro, P. & Díaz-Verdejo, J.E., 2004. Anomaly detection methods in wired networks: a survey and taxonomy. *Computer Communications*, 27, pp.1569-84.

Estévez-Tapiador, J.M., García-Teodoro, P. & Díaz-Verdejo, J.E., 2004. Anomaly detection methods in wired networks: a survey and taxonomy. *Computer Communications*, 27(16), pp.1569-84.

Estevez-Tapiador, J.M., Garcia-Teodoro, P. & Diaz-Verdejo, J.E., 2005. Detection of Web-Based Attacks through Markovian Protocol Parsing. In *Proc. 10th IEEE Symposium on Computers and Communications (ISCC'05)*., 2005.

Estevez-Tapiador, J.M., Garcia-Teodoro, P. & Diaz-Verdejo, J.E., 2005. Detection of Web-Based Attacks through Markovian Protocol Parsing. In *Proc. 10th IEEE Symposium on Computers and Communications (ISCC'05)*., 2005.

Estevez-Tapiador, J.M., García-Teodoro, P. & Díaz-Verdejo, J.E., 2005. Detection of web-based attacks through Markovian protocol parsing. In *Proc. 10th IEEE Symposium on Computers and Communication (ISCC 2005)*., 2005.

Farah, T. & Trajkovic, L., 2013. Anonym: A tool for anonymization of the Internet traffic. In *Proc. IEEE Int. Conf on Cybernetics (CYBCONF 13)*., 2013.

Feiertag, R. et al., 1999. Intrusion Detection Inter-component Adaptive Negotiation. *Computer Networks*, 34, pp.605-21.

Fielding, R.T. et al., 1999. *Hypertext Transfer Protocol -- HTTP/1.1 - RFC 2068*. IETF.

Fontenelle, M.F. et al., 2007. Using Statistical Discriminators and Cluster Analysis to P2P and Attack Traffic. In *Proc. LANOMS*., 2007.

García-Teodoro., P., Díaz-Verdejo, J.E., Tapiador, J.M. & Hernandez-Salazar, R., 2015. Automatic Generation of HTTP Intrusion Signatures by Selective Identification of Anomalies. *Computers & Security*, p.En prensa.

Garcia-Teodoro, P., Diaz-Verdejo, J., Maciá-Fernández, G. & Vázquez, E., 2009. Anomaly-based network intrusion detection: Techniques, systems and challenges. *Computers & Security*, 28(1), pp.18-28.

Ghosh, A.K. & Schwartzbard, A., 1999. A study in using neural networks for anomaly and misuse detection. In *Proc. 8th conference on USENIX Security Symposium*., 1999.

Gonzalo, A., 2006. *Seguridad en Internet*. Madrid: Nowtilus, S.L.

Hanley, J.A. & McNeil, B.J., 1982. The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology*, 143(1), pp.29-36.

Hecht-Nielsen, R., 1988. Applications of counterpropagation networks. *Neural Networks*, 1, pp.131-39.

Heckerman, D., 1996. *A Tutorial on Learning With Bayesian Networks*. Microsoft.

Hopcroft, J.P., 2002. *introducción a la teoría de autómatas, lenguajes y computación*. Pearson Educación.

Hoque, M.S. et al., 2012. An implementation of intrusion detection system using genetic algorithm. *International Journal of Network Security & Its Applications*, 4(2), pp.109-20.

Huidobro M., J.M. & Roldan M., D., 2005. *Seguridad en Redes y Sistemas Informáticos*. Editorial Thompson.

Inc., N., 2010. *Niksun NetDetector*. [Online] Available at: http://www.niksun.com/product.php?id=4.

INCIBE, 2013. *Informe mensual. Red de sensores de Inteco*. INCIBE.

International Business Machine, 1997. *Internet Security Systems*. [Online] Available at: http://www.iss.net/.

ITRG & Group, I.-T.R., 2003. *Intrusion Detection: The Essential Buyer's Guide*. London, Canada.

ITU-T, 1991. *ITU-T recommendation X.800. Security Architecture for Open Systems Interconnection for CCITT applications*. ITU.

Jha, S., Tan, K. & Maxion, R.a., 2001. Markov chains, classifiers, and intrusion detection. In *Proc. 14th IEEE Computer Security Foundations Workshop, 2001.*, 2001.

Johnson, K., 2003. *base.secureideas.net*. [Online].

Joycee, K.A.M., Parkavi, R. & Senthikumari, R., 2013. Network Intrusion Detection & Prevention. *Automation and Autonomous System*, 5(6), p.244.

Kent, S. & R., A., 1998. *Security Architecture for the Internet Protocol (RFC 2401)*. IETF.

Kim, S. et al., 2009. *A Study of International Trend Analysis on Web Service Vulnerabilities in OWASP and WASC*. Springer Berlin / Heidelberg.

Koukis, D. et al., 2006. A Generic Anonymization Framework for Network Traffic. In *Proc. IEEE Int. Conf. on Communications (ICC 06).*, 2006.

Kramer, S. & Bradfield, J.C., 2009. A general definition of malware. *Journal in Computer Virology*, 6, pp.105-14.

Kruegel, C., Mutz, D., Robertson, W. & Valeur, F., 2003. Bayesian Event Classification for Intrusion Detection. In *Proc. 19th Computer Security Applications Conf. 2003.*, 2003.

Kruegel, C., Vigna, G., Robertson, W. & A, 2005. Multi-model approach to the detection of web-based attacks. *Computer Networks*, 48(5), pp.717-38.

Krügel, C., Toth, T. & Kirda, E., 2002. Service specific anomaly detection for network intrusion detection. In *Proc. 2002 ACM symposium on Applied computing - SAC '02.*, 2002.

Laskov, P., Düssel, P., Schäfer, C. & Rieck, K., 2005. Learning intrusion detection: supervised or unsupervised? In *Proc. Image Analysis and Processing (ICIAP 2005).*, 2005.

Laureano, M., Maziero, C. & Jamhour, E., 2007. Protecting host-based intrusion detectors through virtual machines. *Computer Networks*, 51, pp.1275-83.

Lazarevic, A., Kumar, V. & Srivastava, J., 2005. *Managing cyber threats: issues, approaches, and challenges*. Springer Verlag.

Lee, W. et al., 2001. Real time data mining-based intrusion detection. In *Proc. DARPA Information Survivability Conference and Exposition II (DISCEX'01).*, 2001.

Lippmann, R. et al., 2000. Analysis and Results of the 1999 DARPA Off -Line Intrusion Detection Evaluation. *Computers Networks*, 34(4), pp.579-95.

Lunt, T.F. et al., 1990. IDES: A Progress Report. In *Proc. Sixth Computer Security Applications Conf.*, 1990.

Mahoney, M.V. & Chan, P.K., 2001. *PHAD: Packet Header Anomaly Detection for Identifying Hostile Network Traffic*. Florida Institute of Technology.

Mahoney, M. & Chan, K., 2003. An Analysis of the 1999 DARPA/Lincoln Laboratory evaluation Data for Network Anomaly Detection. In *Proc. Recent Advances in Intrusion Detection (RAID 03)*., 2003.

Maiwald, E., 2002. *Network Security: A Beginner's Guide*. McGraw-Hill Professional.

Massicotte, F. et al., 2006. Automatic Evaluation of Intrusion Detection Systems. In *Proc. 22nd Annual Computer Security Applications Conference (ACSAC'06)*., 2006.

Maxion, R.a. & Feather, F.E., 1990. A case study of Ethernet anomalies in a distributed computing environment. *IEEE Transactions on Reliability*, 39, pp.433-43.

McHugh, 2000. Testing Intrusion detection systems: a critique of the 1998 and 1999 DARPA intrusion detection system evaluations as performed by Lincoln Laboratory. *ACM Transactions on Information and System Security* , 3(4), pp.262-94.

McHugh, J., 2001. Intrusion and intrusion detection. *International Journal of Information Security*, 1, pp.14-35.

milw0rm, 1998. *milw0rm*. [Online] Available at: http://www.milw0rm.com [Accessed 2011].

Minshall, G., 1996. *TCPdpriv Command Manual*.

NCSC, 1983. *Trusted Computer System Evaluation Criteria (Orange Book)*. Dept. of Defense - United States of America.

Nessus, 2004. *Tenable Network Security Nessus*. [Online] Available at: http://www.nessus.org [Accessed 2015].

Nielsen, H., Leach, P. & Lawrence, S., 2000. *An HTTP Extension Framework - RFC 2774*. IETF.

Nikto, 2004. *Nikto Web Server Vulnerability Detection Tool*. [Online] Available at: http://www.cirt.net/code/nikto.shtml [Accessed 2012].

Northcutt, S. & Novak, J., 2001. *Detección de Intrusos*. Prentice Hall / Pearson Education.

N-Stalker Co., 2015. *Nstealth Web Application Security Scanner*. [Online] Available at: http://www.nstalker.com/nstealth/ [Accessed 2015].

Oppliger, R., 2009. *SSL and TLS Theory and Practice*. Artech House.

OSVDB, 2008. *Open Source Vulnerabilities Data Base*. [Online] Available at: http://www.osvdb.org [Accessed 2015].

OSVDB, 2009. *Open Sourced Vulnerability Database*. [Online] Available at: http://osvdb.org [Accessed 2015].

OWASP, 2015. *OWASP*. [Online] Available at: http://www.owasp.org [Accessed 2015].

Packetstorm, 2002. *Packetstorm*. [Online] Available at: http://www.packetstormsecurity.nl [Accessed 2015].

Patcha, A. & Park, J.-m., 2007. An overview of anomaly detection techniques : Existing solutions and latest technological trends. *Computer Networks*, 51, pp.3448-70.

Paxson, V., 1999. Bro: a system for detecting network intruders in real-time. *Computer Networks*, 31, pp.2435-63.

Pipping, S., 2007. *Uriparser*. [Online] Available at: http://uriparser.sourceforge.net [Accessed 2015].

Porras, P.A. & Neumann, P.G., 1997. EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances. In *Proc. 1997 National Information Systems Security Conference*., 1997.

Porras, P.A. & Porras, A., 1998. Live Traffic Analysis of TCP/IP Gateways. In *Proc. The Internet Society's Symposium on Network & Distributed System Security*., 1998.

Portnoy, L., Eskin, E. & Stolfo, S., 2001. Intrusion detection with unlabeled data using clustering. In *Proc. ACM CSS Workshop on Data Mining Applied to Security (DMSA-2001}*., 2001.

Puketza, N.J. et al., 1996. A Methodology for Testing Intrusion Detection Systems. *IEEE Transactions on Software Engineering*, 22(10 October), pp.719-29.

PWC, 2015. *Turnaround and transformation in cibersecurity*. PWC.

Rabiner, L.R., 1989. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2), pp.257--286.

Reis, M., Paula, F., Fernandes, D. & Geus, P., 2002. A Hybrid IDS Architecture Based. In *Anais do Wseg 2002: Workshop em Seguranca de Sistemas Computacionais*., 2002.

Riacho, A., 2008. *w3af Web Application Attack and Audit Framework*. [Online] Available at: http://www.w3af.org/ [Accessed 2015].

Roesch, M., 1998-2015. *Snort The open source network intrusion system*. [Online] Available at: http://www.snort.org.

Rossey, L.M. et al., 2002. LARIAT: Lincoln Adaptable Real-time Information Assurance Testbed. In *Proc. IEEE Aerospace Conference*., 2002.

Salazar-Hernandez, R. & Diaz-Verdejo, J., 2007. Generación de tráfico de ataque para la evaluación de sistemas de detección de intrusos Jitel 2007. In *Actas VI Jornadas de Ingeniería Telemática, JITEL 2007*., #nov# 2007.

Salcedo-Campos, F.J., Díaz-Verdejo, J.E. & García-Teodoro, P., 2011. Segmental parameterisation and statistical modelling of e-mail headers for spam detection. *Information Sciences*, (195), pp.45-61.

Scott, S.L., 2004. A Bayesian paradigm for designing intrusion detection systems. *Computational Statistics & Data Analysis*, 45, pp.69-83.

Sebyala, A.A., Olukemi, T., Sacks, L. & Sacks, D.L., 2002. Active platform security through intrusion detection using naive bayesian network for anomaly detection. In *London Communications Symposium*., 2002.

Sekar, R. et al., 2002. Specification-based Anomaly Detection: A New Approach for Detecting Network Intrusions.

Sekar, R. et al., 2002. Specification-based Anomaly Detection: A New Approach for Detecting Network Intrusions. In *Proc. of the 9th ACM conf. on Computer and communications security (CCS 02)*., 2002.

Shanmugavadivu, R. & Nagarajan, N., 2011. Network intrusion detection system using fuzzy logic. *Indian Journal of Computer Science and Engineering (IJCSE)*, 2(1), pp.101-11.

Sharma, S.K. & Manoria, M., 2015. Intrusion Detection using Hidden Markov Model. *International Journal of Computer Applications*, 115(4).

Sharpe, R. & Warnicke, E., 2004. *Wireshark User's Guide*. NS Computer Software and Services.

Shirey, R., 2000. *Internet Security Glossary (RFC 2828)*. IETF.

Shirey, R., 2007. *Internet Security Glossary, version 2 (RFC 4949)*. IETF.

Smaha, S.E., 1988. Haystack: An Intrusion Detection System. In *Proc. Fourth Aerospace Computer Security Applications Conference 1988*., 1988.

Sobh, T.S., 2006. Wired and wireless intrusion detection system: Classifications, good characteristics and state-of-the-art. *Computer Standards & Interfaces*, 28, pp.670-94.

Sobh, T.S., 2006. Wired and wireless intrusion detection system: Classifications, good characteristics and state-of-the-art. *Computer Standards & Interfaces*, 28(6), pp.670-94.

SOG-IS, 1991. *Information Technology Security Evaluation Criteria*. Dept. of Trade and Industry.

Sommer, R., 2010. Outside the ClosedWorld: On Using Machine Learning for Network Intrusion Detection. In *Proc. IEEE Symposium on Security & Privacy*., 2010.

Stallings, W., 2003. *Fundamentos de seguridad en redes, aplicaciones y estándares. 2a. Edición*. Pearson Prentice Hall.

Staniford-Chen, S. et al., 1996. GrIDS - A Graph Based Intrusion Detection System For Large Networks. In *Proc. 19th National Information Systems Security Conference*., 1996.

Stanton, J., Stam, K., Mastrangelo, P. & Jolton, J., 2005. Analysis of end user security behaviors. *Computers & Security*, 24, pp.124-33.

Stewart, J.M., 2008. *CompTIA Security+ Review Guide*. Sybex.

Symantec, C., 2010. *Symantec Endpoint Protection*. [Online] Available at: http://www.symantec.com/business/endpoint-protection.

Tombini, E., Debar, H., Me, L. & Ducasse, M., 2004. A serial combination of anomaly and misuse IDSes applied to HTTP traffic. In *Proc. 20th Annual Computer Security Applications Conference*., 2004.

Tong, X., Wang, Z. & Yu, H., 2009. A research using hybrid RBF/Elman neural networks for intrusion detection system secure model. *Computer Physics Communications*, 180, pp.1795-801.

Vaarandi, R. & Podi, 2010. Network ids alert classification with frequent itemset mining and data clustering. In *Proc. 2010 Int. Conf. on Network and Service Management (CNSM 10)*., 2010.

Valdes, A. & Skinner, K., 2000. Adaptive, Model-based Monitoring for Cyber Attack Detection. In *Proc. 3rd Int. Workshop on Recent Advances in Intrusion Detection (TAID 00)*., 2000.

Van, C.L. & McCanne, S., 1991. *Tcpdump man pages*. Lawrence Berkeley National Laboratory.

Van, C.L. & McCanne, S., 1994. *Libpcap man pages*. Lawrence Berkeley National Laboratory.

Vázquez, J.G., 2010. *Wikipedia*. [Online] Available at: http://es.wikipedia.org/wiki/Lenguaje_de_programaci%C3%B3n.

Vmware, 2015. *Vmware*. [Online] Available at: http://www.vmware.com [Accessed 2015].

Wang, Y. & Abdel-Wahab, H., 2005. A Correlative Context-based Framework for Network Intrusion Detection System. In *Proc. 10th IEEE Symposium on Computers and Communications (ISCC 2005)*., 2005.

Wang, J., Zhang, K. & Ren, D.s., 2008. An Anomaly Intrusion Detection Algorithm Based on Minimal Diversity Semi-supervised Clustering. In *Proc. Int. Symposium on Computer Science and Computational Technology (ISCSCT 08)*., 2008.

Wikto, 2004. *Wikto Web Server Assessment Tool*. [Online] Available at: http://www.sensepost.com/research/wikto/ [Accessed 2011].

Wireshark, 2008-2015. *Wireshark*. [Online] Available at: http://www.wireshark.org.

Yamada, A., Miyake, Y., Takemori, K. & Tanaka, T., 2005. Intrusion detection system to detect variant attacks using learning algorithms with automatic generation of training data. In *Proc. Int. Conf. on Information Technology: Coding and Computing (ITCC'05)*., 2005.

Yao, Y., Wei, Y., Gao, F.-x. & Yu, G., 2006. Anomaly Intrusion Detection Approach Using Hybrid MLP/CNN Neural Network. In *Proc. 6th International Conference on Intelligent Systems Design and Applications*., 2006.

Yeung, D.-Y. & Ding, Y., 2003. Host-Based Intrusion Detection Using Dynamic and Static Behavioral Models. *Pattern Recognition*, 36, pp.229-43.

Ye, N., Zhang, Y. & Borror, C.M., 2004. Robustness of the Markov-chain model for cyber-attack detection. *IEEE Transactions on Reliability*, 53, pp.116-23.

Yurcik, W. et al., 2007. SCRUB-tcpdump: A Multi-Level Packet Anonymizer Demonstrating Privacy/Analysis Tradeoffs. In *Proc. 3rd IEEE Intl. Workshop on the Value of Security through Collab (SECOVAL 07)*., 2007.

Yurcik, W. et al., 2008. Privacy/Analysis Tradeoffs in Sharing Anonymized Packet Traces: Single-Field Case. In *Proc. 3rd Int. Conf. Availability, Reliability and Security (ARES 08)*., 2008.

Zhu, Z.J. & Zulkernine, M., 2007. Towards an Aspect-Oriented Intrusion Detection Framework. In *Proc. 31st Annual Int. Computer Software and Applications Conference, 2007 (COMPSAC 2007)*., 2007.

Zimmermann, H., 1980. OSI Reference Manual. *Architecture*, C.

Zimmermann, P.R., 1995. *The Official PGP User's Guide*. MIT Press.