**UNIVERSIDAD COMPLUTENSE DE MADRID**

FACULTAD DE INFORMÁTICA

Departamento de Arquitectura de Computadores y Automática



# TESIS DOCTORAL

## Desarrollo de algoritmos eficientes para identificación de usuarios en accesos informáticos

## Development of efiicient algorithms for identifying users in computer access

MEMORIA PARA OPTAR AL GRADO DE DOCTOR

PRESENTADA POR


**César Byron Guevara Maldonado**

Directoras

**Matilde Santos Peñas**
**María Victoria López López**


**Madrid, 2018**

# DESARROLLO DE ALGORITMOS EFICIENTES PARA IDENTIFICACIÓN DE USUARIOS EN ACCESOS INFORMÁTICOS

# DEVELOPMENT OF EFFICIENT ALGORITHMS FOR IDENTIFYING USERS IN COMPUTER ACCESS


Memoria para optar al grado de Doctor en Informática

presentada por

**César Byron Guevara Maldonado**

Directoras

**Matilde Santos Peñas**

**María Victoria López López**

*Departamento de Arquitectura de Computadores y Automática*

*Facultad de Informática*

*Universidad Complutense de Madrid*

Madrid, 2017

# DESARROLLO DE ALGORITMOS EFICIENTES PARA IDENTIFICACIÓN DE USUARIOS EN ACCESOS INFORMÁTICOS

# DEVELOPMENT OF EFFICIENT ALGORITHMS FOR IDENTIFYING USERS IN COMPUTER ACCESS

Thesis by

**César Byron Guevara Maldonado**

In Partial Fulfillment of the Requirements for the Degree of

Doctor por la Universidad Complutense de Madrid en el

Programa de Doctorado en Ingeniería Informática

Advisors

**Matilde Santos Peñas**

**María Victoria López López**

*Departamento de Ingeniería del Software e Inteligencia Artificial*

*Facultad de Informática*

*Universidad Complutense de Madrid*

*Madrid, April 2017*

# DESARROLLO DE ALGORITMOS EFICIENTES PARA IDENTIFICACIÓN DE USUARIOS EN ACCESOS INFORMÁTICOS

# DEVELOPMENT OF EFFICIENT ALGORITHMS FOR IDENTIFYING USERS IN COMPUTER ACCESS

Thesis by

**César Byron Guevara Maldonado**

In Partial Fulfillment of the Requirements for the Degree of

Doctor por la Universidad Complutense de Madrid en el

Programa de Doctorado en Ingeniería Informática

Advisors

**Matilde Santos Peñas**

**María Victoria López López**

*Departamento de Ingeniería del Software e Inteligencia Artificial*

*Facultad de Informática*

*Universidad Complutense de Madrid*

Madrid, April 2017

# Agradecimientos

Quisiera agradecer en primer lugar a Dios por darme la oportunidad de llegar paso a paso a las metas propuestas y aprender cada día que la vida es un regalo con continuos retos que ganar.

Agradezco a mi madre, que siempre ha confiado en mí a pesar de todos los problemas, enfermedades y vicisitudes que han existido durante toda mi vida. Tú que me has enseñado que rendirse no es una opción y seguir adelante hasta llegar al objetivo es una forma de vida. Gracias por ser todo para mí y apoyarme con tus palabras y tu amor.

También tengo que agradecer a mi padre el Dr. Oscar Byron Maldonado Torres, que ha inspirado muchos de los estudios que he realizado y espero algún día poder ser una persona de bien como usted. Nunca olvidaré sus palabras de aliento y todos los esfuerzos para mi superación.

Gracias también a mis directoras Matilde Santos y Victoria López por ser mis guías, maestras y amigas en este largo camino que es la investigación. Sin ustedes y su dedicación no habría sido posible llegar a este objetivo de finalizar este trabajo.

Gracias a Milton Chérrez, Glenda Ramos, amigos de LIAAD y Talma que han sido un apoyo fundamental para mi estancia en Europa y que los he considerado verdaderos amigos.

Agradezco a las instituciones públicas del Ecuador por darme la oportunidad de estudiar y alcanzar este objetivo. Gracias al SENESCYT por otorgarme la beca para realizar mi doctorado y financiar mi investigación de manera completa. También a la Unidad de Gestión Inmobiliaria del Sector Público Inmobiliar y a la Universidad de las Fuerzas Armadas ESPE por brindarme el auspicio para realizar mis estudios doctorales.

# Contenido

# Resumen

Actualmente los ciberataques son un problema serio y cada vez más frecuente en organizaciones, empresas e instituciones de todo el mundo. Se pueden definir como el acceso, transferencia o manipulación no autorizada de información de un ordenador o centro de datos. Los datos confidenciales en empresas y organizaciones incluyen propiedad intelectual, información financiera, información médica, datos personales de tarjetas de crédito y otros tipos de información dependiendo del negocio y la industria involucrada.

En esta tesis se realizan varias contribuciones dentro del campo de Detección de Anomalías (AD), Sistema de Detección de Intrusos (IDS) y Detección de Fugas de Información (DLD). Una de las principales aportaciones común a los tres campos mencionados es el desarrollo de una estructura dinámica de datos para representar el comportamiento real y único de los usuarios, lo que permite que cada uno tenga una huella digital que lo identifica.

Otras aportaciones están en la línea de la aplicación de técnicas de inteligencia artificial (IA), tanto en el procesamiento de los datos como en el desarrollo de meta clasificadores (combinación de varias técnicas de IA), por ejemplo: árboles de decisión C4.5 y UCS, máquinas de vectores soporte (SVM), redes neuronales, y técnicas como vecinos cercanos (K-NN), entre otras. Se han aplicado con buenos resultados a la detección de intrusos y han sido validadas con bases de datos públicas como Unix, KDD99, y con una base de datos gubernamental de la república del Ecuador.

Dentro del campo de detección de anomalías, se han usado algoritmos bio-inspirados para la identificación de comportamientos anómalos de los usuarios, como los sistemas inmunes artificiales y la selección negativa, además de otros algoritmos de alineamiento de secuencias, como el de Knuth Morris Pratt, para identificar subsecuencias posiblemente fraudulentas.

Finalmente, en el ámbito de detección de fugas de información, se han desarrollado algoritmos aplicando técnicas estadísticas como las cadenas de Markov a la secuencia de ejecución de tareas de un usuario en un sistema informático, obteniendo buenos resultados que han sido comprobados con bases de datos secuenciales públicas y privadas.

**Palabras clave:** Detección de Anomalías, Sistema de Detección de Intrusos, Detección de Fugas de Información, Inteligencia Artificial.

# Abstract

Cyber-attacks are currently a serious problem and are becoming increasingly frequent in organizations, companies and institutions worldwide. It can be defined as the unauthorized access, transfer or manipulation of a computer or data center. Confidential data in companies and organizations include intellectual property, financial information, medical information, personal credit card information and other information depending on the business and industry involved.

In this thesis, various contributions are made within the field of Anomaly Detection (AD), Intruder Detection Systems (IDS) and Data Leak Detection (DLD). One of the main contributions common to the three aforementioned fields is the development of a dynamic data structure to represent the real and unique user behaviour, which allows each user to have a digital fingerprint that identifies them.

Other contributions are related to the application of artificial intelligence (AI) techniques, both in data processing and in the development of meta-classifiers (combination of various AI techniques), for example C4.5, UCS, SVM, neural networks and K-NN, among others. They have been successfully applied to the detection of intruders and have been validated against public data bases such as UNIX, KDD99 and against a government database of the Republic of Ecuador.

In the field of anomaly detection, bioinspired algorithms have been used in the detection of anomalous behaviours, such as artificial immune systems and negative selection, in addition to other sequence alignment algorithms, such as the Knuth-Morris-Pratt (KMP) string matching algorithm, to identify potentially fraudulent subsequences.

Lastly, in the field of data leak detection, algorithms have been developed applying statistical techniques such as Markov chains to a user's job execution sequence in an information system, obtaining good results which have been verified against sequential databases.

**Keywords:** Anomaly Detection, Intruder Detection System, Data Leak Detection, Artificial Intelligence.

# Parte I: Memoria de la Tesis en español

# Capítulo I: Introducción

En las últimas décadas las redes de información han crecido de manera exponencial debido al gran número de computadores, servidores y equipos de comunicación conectados a internet, lo que ha convertido a estos sistemas en instrumentos imprescindibles en la vida de las personas. Internet se ha convertido en una herramienta que ha revolucionado la sociedad y, especialmente, la informática y las telecomunicaciones.

En la actualidad la mayoría de las personas e instituciones hacen uso de las redes para realizar operaciones y gestiones financieras, tributarias, publicitarias, sociales, de salud, etc.

Debido a este acceso tan abierto, los datos que se manejan deben ser protegidos para no ser mal utilizados. La información es uno de los activos más importantes de las empresas y negocios. De hecho los ciberataques son un problema de gran envergadura que afecta a la mayoría de personas y empresas alrededor del mundo. Este problema se puede describir como la transmisión o manipulación de datos privados de forma no permitida desde una organización a un agente externo (persona u organización no autorizada). En la actualidad es un problema realmente preocupante y una amenaza a la seguridad. Los datos confidenciales de instituciones y personas incluyen propiedad intelectual, información financiera, información de carácter personal y gran variedad de contenidos en función de los negocios y las industrias involucradas (Kumar, 2013). Un estudio realizado por Insight Express en EEUU, encargado por Cisco Systems en 2008, revela que el comportamiento de los empleados supone una de las formas más comunes de amenazas de pérdida de información sensible (Cisco, 2008). Estudios actuales revelan que las amenazas internas causan mayores pérdidas financieras que los ataques realizados desde fuera de la organización (Takebayashi y Tomoyoshi, 2010).

Aunque hay muchos ejemplos, el caso más conocido de fuga de información secreta fue la divulgación de documentos internos de la Agencia de Seguridad Nacional de los Estados Unidos (NSA) a través de "Wikileaks". El problema consistió en revelar unos 250.000 cables diplomáticos y 400.000 informes militares. Este incidente ha recibido un alto nivel de atención y la seguridad informática de EEUU se ha enfrentado a numerosas críticas de otros países y organizaciones de derechos civiles. De hecho este tipo de delitos pueden causar grandes pérdidas financieras o dañar gravemente la reputación de una organización o gobierno (Karhula y Paivikki, 2011).

Por todo esto es necesario implementar nuevas políticas de seguridad que aborden la prevención, detección de intrusos y la fuga de información sensible en las instituciones. Las intrusiones de forma intencionada hacen que la seguridad de la información sea vulnerada. Además los principales agentes de fuga de datos en las instituciones son muchas veces los usuarios con acceso autorizado a los sistemas.

En este trabajo de tesis se presenta el desarrollo de algoritmos basados en la inteligencia artificial para la detección de intrusos en sistemas computacionales y la detección de anomalías en el comportamiento de usuarios para localizar fugas de información. El método se basa en las operaciones, actividades y tareas realizadas por los usuarios dentro de los sistemas de información.

# 1.1. Motivación

Actualmente la seguridad en redes y sistemas informáticos es de gran relevancia debido al valor que posee la información en las instituciones públicas o privadas. Además, el gigantesco volumen de datos que manejan las personas y equipos a los que acceden es un gran problema debido a la posibilidad de filtraciones de forma accidental o intencional, pudiendo causar pérdidas millonarias para personas o instituciones.

El sector público de la república del Ecuador posee una gran red de sistemas de información que permiten realizar actividades y tareas internas de forma automatizada, pero resulta cada vez más difícil salvaguardar la integridad y confidencialidad de la información. Esta red es objeto de constantes y variados tipos de ciberataques para la sustracción y modificación de información reservada de las instituciones gubernamentales. Esta gran red de información comparte documentación importante entre diversas instituciones, lo que fomenta la suplantación de usuarios y la hace vulnerable a ataques de agentes no autorizados. Por eso se ha visto la necesidad de implementar un sistema eficiente para tener un adecuado control de acceso y evitar este tipo de ciberataques.

Los esfuerzos que se han realizado hasta ahora para evitar este tipo de ataques han dado como resultado sistemas de seguridad complejos y poco eficientes, ya que la mayoría de las suplantaciones e intrusiones se realizan desde dentro de las mismas entidades, lo que dificulta la identificación de estos usuarios autorizados. Esta es la principal razón por la cual se ha propuesto realizar un estudio del comportamiento de los usuarios aplicando técnicas inteligentes al reconocimiento de patrones, que permitan identificar de una forma eficiente y automática los accesos fraudulentos. Es decir, se han desarrollado modelos para la detección de anomalías, intrusiones y fugas de información utilizando el comportamiento único de cada usuario en los sistemas.

# 1.2. Objetivos

## *1.3.1. Objetivo General*

El objetivo de este trabajo es el diseño, desarrollo e implementación de un conjunto de algoritmos eficientes, aplicando técnicas de inteligencia artificial, para la detección de anomalías, detección de fugas de información e identificación de accesos fraudulentos en sistemas informáticos.

## *1.2.2. Objetivos Específicos*

- Analizar conjuntos de datos reales para identificar ataques informáticos tanto en red como en sistemas.
  La investigación tendrá como base fundamental el uso de información de ataques a redes y sistemas informáticos, utilizando múltiples conjuntos de datos para desarrollar modelos de clasificación y otras soluciones para la detección de intrusos y fugas de datos. En concreto se ha trabajado con la base de datos como el KDD99, Amazon, Unix Comands y el conjunto de

datos llamado Ecuador que contiene información codificada del comportamiento de varios usuarios en una institución real de gobierno ecuatoriano.

- Desarrollar una estructura secuencial de datos dinámica para describir el patrón de comportamiento de los usuarios.
  Esta estructura de datos proporciona una información detallada sobre orden, frecuencia y manera de realizar las actividades dentro de un sistema u ordenador personal, lo que brinda una herramienta de conocimiento ideal para determinar el comportamiento normal y anormal de cada uno de los usuarios. Esta nueva estructura ha sido implementada en varios modelos y algoritmos propuestos para la detección de intrusos.
- Aplicar técnicas de la inteligencia artificial de manera conjunta para identificar intrusiones en equipos informáticos.
- Esta sinergia de técnicas de clasificación inteligente ha conducido a desarrollar detectores más eficientes, rápidos y dinámicos. Con soluciones interesantes y novedosas para el campo de la ciberseguridad en la detección online de intrusos.
- Desarrollar algoritmos basados en técnicas estadísticas para calcular la probabilidad de un ataque según el comportamiento de los usuarios.
  Ya que el comportamiento secuencial y rutinario de los usuarios permite el cálculo de su probabilidad de con otra fiabilidad. Este algoritmo ha sido probado con conjuntos de datos reales obteniendo resultados muy buenas.
- Desarrollar algoritmos basados en técnicas bio-inspiradas para la detección de fugas de información.
  Mediante la identificación de anomalías en la conducta de los usuarios. Estos algoritmos han utilizado el histórico del comportamiento normal para generar el comportamiento fraudulento

# 1.3. Estructura de la tesis

Esta tesis está organizada de la siguiente forma. Tras este capítulo introductorio, en el capítulo 2 se presenta el estado del arte en el área de seguridad informática, con una breve historia, definición, elementos que la constituyen y las amenazas existentes. Además se muestra la clasificación de los ataques informáticos y los diferentes aspectos en los que interactúan, debilidades de los sistemas de gestión, y defectos de diseño.

En el capítulo 3 se describen las bases de datos utilizadas, sistemas de detección de anomalías, intrusos y fuga de información junto con la estrategia en la que se basa cada uno de ellos. Se comentan brevemente las técnicas de inteligencia artificial y algoritmos utilizados en este trabajo para la detección de ataques informáticos, como son las redes neuronales artificiales, árboles de decisión, algoritmos genéticos, etc.

En el capítulo 4 se presentan los artículos y contribuciones publicadas en congresos nacionales, internacionales y en revistas indexadas. Estos artículos fueron desarrollados con diferentes bases de datos y algoritmos para generar resultados comparativos en el ámbito de la detección de anomalías, detección de intrusos y fuga de información. Estas publicaciones y sus índices de calidad avalan el interés de las aportaciones presentadas, así como los buenos resultados obtenidos.

En el capítulo 5 se presentan las principales conclusiones que han resultado de la investigación y las futuras líneas de trabajo en esta línea.

# Capítulo II. Seguridad informática

La seguridad informática es un área que se enfoca específicamente a la protección de la infraestructura computacional (equipos de comunicación, almacenamiento de datos, servidores, computadores, dispositivos móviles, etc.), salvaguardando la información contenida o que se comparte en medios electrónicos. Para cumplir este objetivo se han creado leyes, normativas, estándares, protocolos, equipos, herramientas, aplicaciones, etc. que ayudan a evitar en lo posible el riesgo de que la información pueda caer en manos de personas no autorizadas.

La seguridad informática es, por lo tanto, la disciplina que se ocupa de diseñar las normas, procedimientos, métodos, técnicas, herramientas o aplicaciones destinadas a conseguir confidencialidad, integridad y disponibilidad en un sistema de información (Stavroulakis y Stamp, 2010).

La seguridad informática se basa en tres pilares fundamentales: la confidencialidad, es decir, que la información sea accesible sólo a aquellas personas que posean autorización; la integridad de la información, que se conserve inalterada ante eventos maliciosos, accidentes o desastres; y la disponibilidad, que permite la utilización de la información de un sistema informático en todo momento por los usuarios autorizados, sin sufrir ninguna degradación en cuanto a accesos. Por último, la autenticidad es la propiedad que permite identificar el generador de la información (Russell y Gangemi, 1991).

## 2.1. Elementos de la seguridad informática

La seguridad informática fue desarrollada para proteger los principales elementos informáticos, los cuales son:

- La infraestructura computacional: consiste en buscar el óptimo funcionamiento y administración de los equipos tecnológicos para prevenir cualquier evento nocivo (fraudes, fallos, desastres naturales, robos, etc.), que no permita el normal desempeño de los equipos.
- Los usuarios: personas que utilizan la infraestructura computacional y que administran la información a ser protegida. La tarea principal es garantizar que el manejo de la información por parte de los usuarios sea adecuada y segura.
- La información: es el principal activo de una persona o institución. La información permanece en la infraestructura computacional y es manejada por los usuarios.

Las agresiones a la seguridad de un sistema o de una red de información describen un flujo de información desde un origen (disco duro, fichero, base de datos, etc.) hasta un destino, como puede ser otro fichero o un usuario. Existen cuatro tipos posibles agresiones a la información, que son:

- Interrupción: se describe como destrucción o inutilización de un recurso, de una red o sistema informático que deja de estar disponible para los usuarios. Ésta es una agresión a la disponibilidad.
- Intercepción: se describe como un agente (persona, programa o equipo) no autorizado que consigue acceder a un recurso para realizar actividades fraudulentas (intervenciones de las líneas para capturar datos, copia ilícita de ficheros o programas, etc.). Ésta es una agresión a la confidencialidad.
- Modificación: el deterioro del recurso (modificación de la información, alteración de un programa, edición de contenido de mensajes, etc.) reeditado de forma ilegítima. Ésta es una agresión a la integridad.
- Fabricación: creación de objetos falsos (archivos, programas, datos) por parte de un agente no autorizado el cual los inserta en los equipos o sistemas de información. Ésta es una agresión a la autenticidad.

### 2.1.1. Ataques informáticos

Un ataque informático es un método por el cual un agente (persona, equipo o programa no autorizado), mediante un sistema o equipo informático, intenta tomar el control, desestabilizar o dañar otro sistema informático (ordenador, red privada, etc.) (Burke y Christiansen, 2009). Los ataques informáticos se agrupan en dos tipos, pasivos y activos.

a) Ataques pasivos: se realizan monitorizaciones de las transmisiones realizadas en una red o sistema de información. El objetivo es obtener información que está siendo transmitida para ser divulgada y utilizada para acciones fraudulentas. Para realizar este tipo de ataque el agente no autorizado realiza un análisis del tráfico de la red para extraer la información confidencial. El agente fraudulento podría determinar la localización y la identidad de los computadores que se están comunicando y observar la frecuencia y la longitud de los mensajes intercambiados. Esta información puede ser útil para extraer la naturaleza de la comunicación que se está realizando. Las agresiones pasivas son muy difíciles de detectar ya que no implican la alteración de los datos. Sin embargo, es factible impedir el éxito de estos ataques. Así, el énfasis para tratar estas agresiones está en la prevención antes que en la detección.
b) Ataques activos: modifican el flujo de datos o crean flujos falsos. Se clasifican en cuatro categorías: enmascaramiento, repetición, modificación de mensajes y denegación de un servicio.

Tanto los ataques activos como pasivos requieren un análisis técnico para prevenir cualquier intento de acción no autorizada en la red o sistema de información.

# 2.2. Sistemas de seguridad

Internet y los sistemas informáticos han planteado numerosos problemas de seguridad debido al uso continuo de las redes. Estudios realizados muestran que el número de intrusiones ha aumentado exponencialmente cada año. Toda intrusión o ataque malintencionado a la red, servidores, ordenadores o sistemas de información pueden causar pérdidas millonarias o desastres graves a las personas o instituciones, además de violar las políticas de seguridad de la información. La ciberseguridad es un área importante de investigación para mejorar la detección y prevenir todo tipo de amenazas a las redes y sistemas donde se encuentra la información almacenada (Kleinberg et al., 2001).

Para identificar mejor los diversos aspectos del problema de la intrusión en informática se habla de sistemas de detección y sistemas de prevención (Deepa y Kavitha, 2012).

Se describe como una intrusión al intento de comprometer la información, evitar o vulnerar los mecanismos de seguridad de una red o computador. La detección de intrusiones es el proceso de monitoreo de los eventos que ocurren en un sistema informático o red, analizándolos para poder identificar signos de posibles ataques (Kantzavelou y Katsikas, 2010). Un sistema de detección de intrusos (IDS), es un software o hardware que permite automatizar el proceso de detección de intrusiones e identificar la entrada no autorizada (Kabiri y Ghorbani, 2005; Chung y Wahid, 2012).

Dentro de la detección existe un contexto más específico que es la detección de fugas de información, posterior a la detección de intrusos. Ocurre con usuarios autorizados en los que el IDS no ha detectado ninguna amenaza, pero que realizan actividades fraudulentas al filtrar información sensible.

Por otro lado, la prevención de intrusiones posee las características y habilidades del IDS, pero su objetivo principal es intentar detener posibles incidentes limitando el uso de sus recursos, es decir, cambia el contenido del ataque y/o cambia el entorno de seguridad y evita que tenga éxito. También puede variar la configuración de otros controles de seguridad para interrumpir un ataque, como reconfigurar un dispositivo de red para bloquear el acceso del atacante o la víctima, o alterar un firewall basado en host con el objetivo de bloquear los ataques entrantes. Pero se producen altas tasas de falsas alarmas y se puede identificar incorrectamente una actividad normal (no intrusiva) como malintencionada y responder a esa actividad de manera incorrecta.

## 2.2.1. Sistema de detección de intrusos

Un sistema de detección de intrusos (IDS) es un programa de detección de accesos no autorizados a un computador o a una red. El IDS normalmente posee sensores (*sniffer* de red) dentro del núcleo del IDS, los cuales monitorizan las actividades (generalmente sobre el tráfico de la red, comportamiento de los usuarios, etc.). El IDS detecta las anomalías que pueden ser indicio de la presencia de ataques (Lazarevic et al., 2005).

En la actualidad la seguridad de la información es un área muy importante para cualquier persona o institución alrededor del mundo, ya que los datos se han convertido en el activo más importante el cual debe ser salvaguardado de una manera eficiente y adecuada. La gran mayoría de los datos debe mantenerse seguros de cualquier intruso o actividad no permitida, de modo que la seguridad tiene una importancia crítica. El término de ”intrusión” como lo presenta (Heady et al., 1990) se puede definir como cualquier conjunto de acciones que tratan de comprometer la integridad, confidencialidad o disponibilidad de un recurso. Por ello es necesario utilizar una herramienta que pueda detectar estas actividades y mantener la información accesible solo a las personas autorizadas. Dicha herramienta se la denomina como Sistema de Detección de Intrusos (IDS), el mismo que analiza eventos que suceden en un sistema informático en busca de signos de intrusiones. El principal objetivo de un IDS es monitorear la actividad en un servidor, red o un equipo informático (PC, Tablet, móvil, etc.) de tal forma que permita identificar de manera eficiente posibles ataques o intentos de violación a la seguridad basados en patrones de comportamiento, firmas de código o análisis de protocolos, para luego alertar al administrador del equipo. El gran auge en el desarrollo e implementación de múltiples IDS han surgido algunas maneras en la recolección y utilización de la información para el sistema de detección, los cuales se describen continuación:

- Basado en el Host: Este tipo de sistema de detección utiliza la información de tareas o actividades realizadas en el equipo donde se encuentra funcionando el IDS.

- Basado en la Red: Este sistema de detección utiliza toda la información del tráfico de la red que se genera en el ambiente que se encuentra funcionando, es decir, recolecta la información de red interna como información de red externa (conexiones entrantes y salientes).

Con los tipos anteriores de recolección de datos permiten una variedad de IDS como la detección de usos indebidos (*misuse detection*) la cual compara la información recogida con descripciones (o firmas) de ataques conocidos. Por otra parte, la detección de anomalías (*anomaly detection*) utiliza los datos históricos sobre la ejecución de tareas o actividades en el sistema y detalla el comportamiento deseado de usuarios como de las aplicaciones, para construir un perfil que representa la operación normal del sistema monitorizado, e identifica patrones de actividades que se desvían del perfil definido.

En la actualidad los Sistemas de Detección de Intrusos (IDS) son de gran relevancia para la seguridad en las redes y sistemas informáticos ya que cada vez y con mayor frecuencia aparecen nuevas formas de ataques, algunos muy complejos (Debar y Viinikka, 2005). Los IDS en su mayoría analizan el tráfico de la red e impiden cualquier intrusión previa al ingreso de un usuario al sistema de información. El análisis realizado por el IDS en general es a bajo nivel, generando alertas aisladas y manejando una inmensa cantidad de información. Otros tipos de IDS realizan un filtrado de anomalías de la información recopilada de los equipos host, así como también de las bases de datos de los sistemas con los que interactúan (Zurutuza y Uribeetxeberria, 2005).

El campo de detección de intrusos sigue siendo un frente abierto a la investigación para desarrollar metodologías dinámicas que puedan adaptarse a la evolución de ataques informáticos, cada vez más sofisticados y complejos. Las estrategias que utilizan los IDS se clasifican en dos grupos: detección de uso indebido y detección de anomalías. La metodología de Detección de Intrusos basada en Anomalías, la cual es utilizada en este trabajo, es extensa y compleja, y ha permitido

obtener resultados favorables pero no enteramente adaptables a las necesidades actuales de la tecnología.

La detección de uso indebido de un sistema informático utiliza el conocimiento de las secuencias de actividades que constituyen un ataque, las cuales están almacenadas en una base de datos. La información recopilada es comparada con los patrones de ataque almacenados y, si coinciden, originan una alarma. Esta estrategia es una de las más utilizadas de forma comercial (Bace y Mell, 2001). Sus ventajas son la correspondencia con un patrón de intrusión de base de datos ya almacenado (rapidez), y que el número de falsos positivos sea muy bajo (fiabilidad y precisión). Sin embargo algunas de sus desventajas son, por ejemplo, la incapacidad de detectar nuevos ataques al no ser completamente dinámico, y además el requisito fundamental de mantener actualizada de forma continua la base de datos de los patrones.

Por otro lado, la detección de anomalías utiliza la información de un comportamiento normal del usuario y todo lo que sea diferente de aquella conducta es identificado como un intruso. Sus desventajas son que genera una gran cantidad de falsos positivos y que el comportamiento normal de los usuarios es bastante difícil de modelar, por la necesidad de contar con un periodo amplio de recopilación de información para el aprendizaje del comportamiento (Wang y Gombault, 2007).

Además existen una variedad de técnicas, metodologías y algoritmos para el desarrollo de un IDS. El área más utilizada para los IDS es la aplicación de Inteligencia Artificial con las técnicas de Machine Learning y Minería de Datos (Data Mining). En esta área se han realizado múltiples investigaciones y artículos, obteniendo grandes avances de sobre la aplicación distintas técnicas como Arboles de Decisión, Redes Neuronales, Algoritmos Genéticos, Support Vector Machines, Sistemas Inmunes Artificiales entre otras (Scarfone y Mell, 2007). Los IDS también utilizan técnicas y modelos estadísticos con el propósito de automatizar completamente la detección de ataques maliciosos distinguiendo del uso normal de los sistemas. Las más utilizadas son Redes Bayesianas, Cadenas de Markov, etc.

Un sistema de detección de intrusos IDS también se puede definir como un sistema de software o hardware automatizado para realizar un proceso de monitoreo y análisis de datos del medio informático para la detección de intrusiones (Bace y Mell, 2001; Stavroulakis y Stamp, 2010; Garcia-Teodoro, 2009). Algunos trabajos sobre IDS centran su atención en los sistemas inmunes artificiales como una ´optima alternativa para identificar comportamientos anómalos los presenta en (Debar et al., 1999; Debar et al., 2000; Kumar et al., 2006; Murali y Rao, 2005). Estos han validado que dicha técnica al obtener resultados fiables.

Un creciente número de investigadores aplicar técnicas basadas en la evolución y para la solución de diversos problemas, incluyendo diagnóstico de fallos, la detección de virus y detección de fraude hipotecario, entre otros ejemplos (DasGupta, 1993; Kephart, 1995; Kim y Bentley, 2001; Hofmeyr y Forrest, 2000; Forrest y Hofmeyr, 2001).

Uno de los trabajos más recientes sobre la aplicación de sistemas inmunes artificiales es el publicado por (Gong et al., 2009), que propone un algoritmo de selección negativa que ha demostrado ser eficaz para problemas de detección de anomalías. Presenta una nueva estrategia en

la etapa de entrenamiento para la reducción de muestras y así reducir el coste computacional en fase de pruebas. Este algoritmo puede obtener altas tasa de detección y reduce falsas alarmas en la mayoría de los casos.

Por otro lado la investigación realizada por (Aziz et al., 2012) presenta un nuevo enfoque para la detección de anomalías en el tráfico de red utilizando detectores generados por un algoritmo genético. Este trabajo utiliza el algoritmo de selección negativa en un sistema inmune para detectar patrones anómalos. Muestra una comparativa con otros experimentos realizados con un conjunto de datos llamado NSL-KDD. El algoritmo propuesto obtiene resultados muy buenos en el análisis en comparación con otros métodos de aprendizaje automático.

Otro trabajo muy interesante que aplica el sistemas inmunes con selección negativa, es el de (Peng et al., 2013). Detalla los excelentes mecanismos de auto-aprendizaje, la capacidad de adaptación del sistema inmunológico humano, y además, muestra los conceptos y las definiciones formales de antígeno, anticuerpos y células de memoria en el dominio de seguridad de la red. Un aspecto muy importante de la investigación es cómo se define la evolución de los perfiles de detección (incluida la generación de perfiles, de aprendizaje dinámico, transformación dinámica, y la auto-organización dinámica), que logrará que los perfiles de detección dinámica se puedan sincronizar con el entorno de red real. Este trabajo obtiene resultados experimentales buenos y aportan una alternativa tener en cuenta para la detección de anomalías en red.

Otro enfoque que ha tenido gran éxito ha sido el uso del sistema inmune natural en la protección de los seres vivos contra una amplia variedad de patógenos como lo presenta (Tizard, 1992). Específicamente, el sistema inmunológico utiliza dos principios fundamentales que se incluyen en la teoría de redes inmunes: los mecanismos de selección negativa y los principios de selección clonal.

En todos estos ámbitos, la detección de anomalías es un área de investigación importante en la aplicación de un sistema inmune artificial (AIS) que ha sido probada con excelentes resultados.

Existen otros trabajos en esta línea de investigación como de los (Lazarevic et al., 2003) y (Chandola et al., 2009), que buscan objetos anormales que son diferentes de los objetos normales. El principal problema para detectar anomalías en secuencias de tareas es la gran cantidad de datos para el procesamiento y el desarrollo del modelo dinámico adaptable a la conducta humana. En el artículo de (Javitz et al., 1993) se utilizan las distribuciones estadísticas para definir el comportamiento normal y anormal. Por otra parte, los trabajos dirigidos a la detección de anomalías presentados por (Mukherjee et al., 1994) describen los muchos perfiles de las actividades normales de los usuarios, los sistemas, los recursos del sistema, tráfico de red, servicios y detecta intrusiones mediante la identificación de desviaciones significativas de los patrones de comportamiento normales observados a partir de perfiles.

Existe gran variedad de aspectos y puntos de vista para clasificar las técnicas, métodos y formas de abordar el problema de los IDS. En la figura 8 se presentan cuatro aspectos generales para su clasificación: la implementación del sistema (System Deployment), fuente de datos (Data Source), secuencia temporal (timeline) y la estrategia de detección (Detection Strategy).

En la rama del sistema realiza un análisis de la arquitectura del host y la red a ser protegidos. Se estudia tres aspectos: tecnología (Technology Type), tipo de red (Networking Type) y arquitectura de red (Network Architecture) (Figura 1). En nuestra investigación nos hemos centrado en el tipo de tecnología, la cual se ha clasificado en cuarto clases según el lugar donde se han presentado las actividades sospechosas y qué clases de eventos hay que reconocer. Las cuatro clases son: IDS basados en host (HIDS), IDS basado en red (NIDS), IDS basado en Wireless (WIDS), e IDS mixtos (MID) (Mukherjee et al., 1994; Stavroulakis y Stamp, 2010; Sabahi y Movaghar, 2008; Modi et al., 2012). El HIDS monitoriza y recopila las características y actividades de los equipos host que contienen información sensible. Un NIDS captura el tráfico de la red en segmentos específicos por medio de sensores, que posteriormente analizan las actividades de las aplicaciones y sus protocolos. El WIDS es similar al NIDS, pero en este tipo sólo monitoriza el tráfico de red inalámbrica, como las redes “ad hoc”, sensores de red y redes inalámbricas mesh. Por otra parte también se pueden se puede aplicar varios de estos tipos de IDS de forma conjunta para mejorar el porcentaje de detección y que sean más completos, precisos y eficientes en la identificación de posibles ataques.

Respecto a la fuante de datos se analizan  en el host como en la red. Para este tipo de IDS hay cuatro tipos de análisis de datos (Data Type), repositorios de datos (Data Collection) y de componentes (Collection Component).

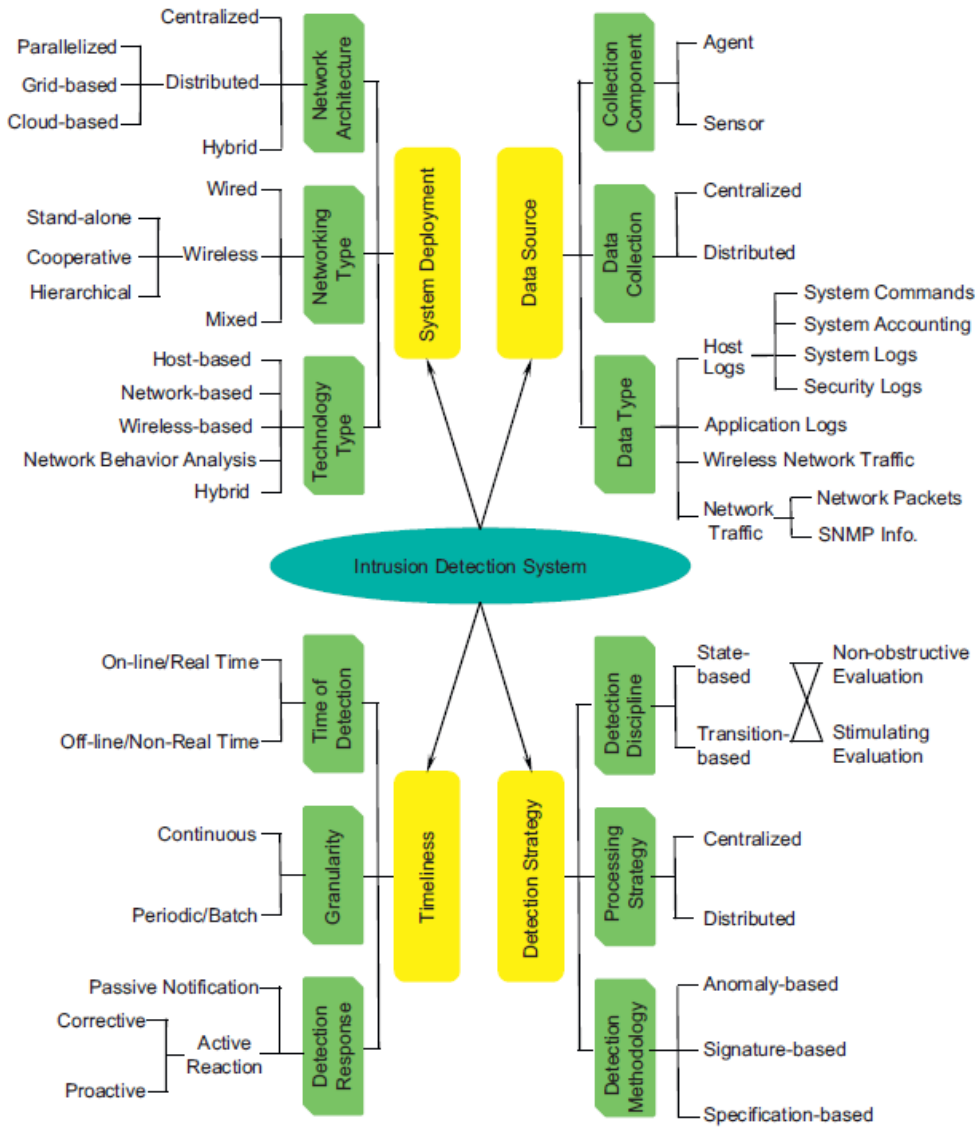



Figura 1. Una visión general de los tipos de IDS (Liao et al., 2013).

Las metodologías de detección aplicadas a los sistemas de detección de intrusos son muy variadas, por ejemplo la detección basada en la firma (SD), en anomalías (AD) (Alomari y Othman, 2012), en análisis de protocolo con estado (SPA), etc. (Debar et al., 2000). En este trabajo nos hemos centrado en la detección basada en anomalías.

Una anomalía es una desviación de un comportamiento conocido (normal), basado en un perfil o patrón. Este comportamiento es derivado del monitoreo de actividades regulares, conexiones de red, hosts o usuarios durante un período de tiempo. Los perfiles pueden ser estáticos o dinámicos, y desarrollarse para muchos atributos, por ejemplo, intentos fallidos de inicio de sesión, uso del procesador, el recuento de correos electrónicos enviados, etc. A continuación se comparan los perfiles normales con los eventos observados para reconocer ataques significativos. Este enfoque también se denomina detección basada en comportamiento en algunos artículos.

En el área de detección de intrusos se pueden desarrollar dos puntos de vista, como se ha comentado antes: detección de anomalías y detección de mal uso (que incluirá las fugas de

información). Un ejemplo de la detección por mal uso es la investigación que se centran en el control de acceso basado en roles (RBAC), empleada en entornos de bases de datos distribuidas, pero siguen siendo propensas a intentos de acceso ilegítimo. En el RBAC los usuarios ponen en riesgo la seguridad mediante intentos de acceso por medio de credenciales falsas o uso incorrecto, intencional o no, según el rol del usuario. Este esquema combina componentes de seguridad informática como comprobación de credenciales, análisis de consultas, monitorización de registros de historial, y una formulación probabilística del riesgo. Por otro lado, un ejemplo de detección de anomalías es el método de inspección profunda de paquetes de red, que también se usa ampliamente para detección de intrusiones (Roesch, 1999). Este trabajo describe de forma detallada el diseño e implementación de un algoritmo de coincidencia de cadenas para manejar los patrones cortos y flexibles en el tráfico de la red (Hao, 2005). La gran desventaja de este algoritmo es que no está diseñado para los diversos tipos de datos, lo que hace difícil distinguir información sensible de la normal, lo que puede causar problemas en la detección.

En el trabajo de (Stavroulakis y Stamp, 2010), se propuso una clasificación de los IDS basada en métodos de detección estadística, patrones y en heurísticas.

A continuación se enumeran los métodos estudiados:

1. Basado en estadística
    a) Métodos estadísticos (Debar et al., 2000; Patcha et al., 2007; García-Teodoro et al., 2009; Xie, 2011).
    b) Métodos bayesianos (Kabiri y Ghorbani, 2005; Murali y Rao, 2005; Sabahi y Movaghar, 2008; Lazarevic et al., 2005.).
2. Basado en patrones
    a) Reconocimiento de patrones (Tsai et al., 2009; Ye et al., 2001).
    b) Máquinas de vectores de soporte (Modi et al., 2012; Kolias et al., 2011; Li et al., 2012; Horng et al., 2011).
    c) Cadenas de Markov (Couture, 2012; Gao et al., 2002).
3. Basado en heurísticas
    a) Redes neuronales (Murali y Rao, 2005; Mar, 2012; Modi et al., 2012; Kenkre et al., 2015).
    b) Algoritmos genéticos (Sen y Clark, 2011; Gong et al., 2005).
    c) Sistemas inmunes (Debar et al., 2000; Stavroulakis y Stamp, 2010).

Finalmente, respecto al tiempo analiza el momento de la detección que puede ser "fuera de línea " (outline) o" detección en tiempo real " (online). Este tipo de IDS estudia la forma periódica de la transmisión de datos, es decir, "transformación continua '', y busca signos de ataques a través del tiempo (granularidad).

Teniendo en cuenta estas premisas, nuestra propuesta es desarrollar un método dinámico para la detección de intrusos mediante el análisis de anomalías en el tráfico de la red. Como un primer paso para alcanzar el objetivo en esta investigación se utilizan de manera conjunta varias estrategias de clasificación que provienen de la Inteligencia Artificial, en concreto los árboles de decisión, las máquinas de vectores de soporte y las redes neuronales. Se puede probar así que la sinergia de varias técnicas hace más eficiente y fiable el sistema de detección.

## 2.2.2. Sistemas de detección de anomalías

La detección de anomalías en actividades, tareas o comportamientos realizados por usuarios, no es una tarea fácil ya que implica el análisis de una gran cantidad de datos, y la obtención de perfiles de comportamiento únicos para cada comportamiento, de forma que se puedan detectar posibles irregularidades (Chandola et al., 2012; Guevara et al., 2014b).

En la literatura se encuentran distintos enfoques para desarrolla sistemas de detección de anomalías. Por ejemplo, en (Gao et al., 2002) se usa HMM (Hidden Markov Models) para aprender los patrones de comportamiento normal y anómalo de usuarios que realizan procesos UNIX. El trabajo presentado en (Helman y Bhangoo, 1997) trata con datos históricos de comandos ejecutados por usuarios, y establece un ranking para seleccionar de entre ellos los más relevantes. Propone la clasificación de cada secuencia según la frecuencia de lo que se denominan las trazas normales y la de los procesos raros o no esperados, donde se encuentran las intrusiones. En (Lazarevic et al., 2003) se presenta una revisión comparada de varios métodos de detección de anomalías para detección de intrusos en sistemas informáticos, focalizada en redes con un gran número de conexiones. Más actuales son las revisiones de (Patcha et al., 2007; Chandola et al., 2009; García-Teodoro et al., 2009; Bhuyan et al., 2014). En ellas se clasifican por las técnicas aplicadas, la complejidad computacional, el dominio, etc. Sin embargo, en esos trabajos citados no tienen en cuenta la estructura secuencial de ejecución de las operaciones sino la cantidad de las operaciones ejecutadas, la transferencia de información, etc.

La aplicación de técnicas de selección negativa, que emulan el Sistema Inmune Artificial (AIS), para detección de anomalías no es nueva (Laurentys et al., 2010). El AIS ha inspirado el desarrollo de modelos computacionales para la resolución de diversos problemas, incluyendo el diagnóstico de fallos, detección de virus, malware y detección de fraude, entre otras aplicaciones (García et al., 2015; Dasgupta, 1999; Aickelin et al., 2014; Dasgupta et al., 2004). En concreto existen en la actualidad sistemas de detección de intrusos (IDS) que utilizan esta aproximación como se refleja en las revisiones de (Bhuyan et al., 2014). Algunos ejemplos de detección de comportamiento anómalos mediante selección negativa se pueden ver en (González y Dasgupta, 2003), donde se parte sólo de muestras positivas para entrenar el sistema (y no de muestras de comportamiento anómalo, que se generan mediante este algoritmo); y en (Kim y Bentley, 2001; Aziz et al., 2014), para el tráfico de una red de información.

Un artículo muy interesante en la línea de nuestro trabajo es el de (Chandola et al., 2012), ya que está centrado en la detección de anomalías en secuencias discretas. Formula el problema en tres pasos: identificación de las secuencias anómalas respecto a una base de datos que contiene las secuencias normales; identificación de las subsecuencias anómalas dentro de secuencias más largas; identificación del patrón en una secuencia cuya frecuencia de ocurrencia es anómala. Para cada uno de estos tres procesos agrupa las técnicas que se han venido aplicando en diferentes dominios, glosando sus puntos fuertes y sus desventajas. Como allí se sugiere, la tendencia actual es la hibridación de varias de esas metodologías para adaptarlas a las distintas formulaciones del problema. En este sentido está dirigida nuestra propuesta, donde la selección negativa nos permite general detectores de anomalías que luego serán utilizados por el algoritmo de Knuth-Morris-Pratt para detectar subsecuencias anómalas dentro de la base de datos de secuencias del usuario informático.

En el presente trabajo se propone una estructura de datos dinámica que permite codificar las actividades de los usuarios. La identificación de las tareas más y menos frecuentes, específicamente la secuencia de tareas realizadas, basada en el histórico del cada usuario, lleva a la creación de un perfil de comportamiento para cada uno. En esto se va a basar la detección de anomalías. Por otro lado, se han aplicado técnicas de selección negativa para obtener un conjunto de dimensiones reducidas, de forma que sea computacionalmente manejable, de detectores anómalos. Posteriormente se ha aplicado el algoritmo de alineamiento de secuencias Knuth-Morris-Pratt para la localización de estos detectores anómalos definidos como indicadores de comportamientos fraudulentos en secuencias de actividades de usuarios.

### 2.2.3. Sistemas de detección de fugas de información

La detección y alerta de la fuga de datos por usuarios autorizados se encuentra entre los problemas más difíciles a los que se enfrentan los investigadores y profesionales de la ciberseguridad. Hace algunos años los expertos en seguridad identificaban las posibles amenazas internas mediante el examen de las actividades sospechosas en el comportamiento físico de los usuarios. Si bien esto sigue siendo relevante en la actualidad, ahora la tendencia es detectar actividades sospechosas por las operaciones que se ejecutan en los sistemas informáticos. A pesar de ello, la información confidencial continúa filtrándose de manera alarmante.

En la década de 1990, organizaciones como el Defense Personnel and Security Research Center (PERSEREC) y la Corporación Research and Development (RAND) de los EEUU, centran su discurso académico sobre el tema de las amenazas internas a través de publicaciones y talleres (Herbig y Wiskoff, 2002; Anderson, 1999; Anderson, 2000; Anderson y Brackney, 2004). A partir de 2001, el CERT Insider Threat Center también comenzó a realizar investigaciones sobre este tema, iniciando el desarrollo de una base de datos de amenazas de información privilegiada. Durante este período de tiempo, los investigadores también han desarrollado modelos tanto para la identificación de las diferentes características de usuarios internos (las principales víctimas de la organización), como del incidente en sí. El objetivo de estas investigaciones era centrarse en aspectos del comportamiento del delincuente al sustraer la información confidencial (Shaw et al., 1998): características psicológicas como la dependencia del ordenador, la flexibilidad ética, y la falta de empatía como indicadores potenciales de un riesgo para la conducta destructiva y potencialmente ilegal. En el enfoque más específico del ámbito socio-técnico se analizan las características, rasgos de personalidad, comportamientos técnicos (incluidos los patrones de uso y ejecución de errores significativos) para identificar las fugas de datos. Estos indicadores son una medida ponderada que facilita la tarea de detección (Schultz, 2002).

Posteriormente, se realizaron estudios que presentan un enfoque dirigido al desarrollo de algoritmos y procedimientos de acceso a la información, donde se permite que los usuarios autorizados tengan acceso exclusivo a los datos sensibles a través de políticas de control (Bonatti et al., 2002; Jajodia et al., 2001). Estos trabajos se centran en impedir la fuga de información restringiendo el uso de los datos sólo a usuarios o sistemas de confianza. Sin embargo estas políticas son restrictivas y pueden hacer que sea imposible satisfacer las peticiones de los usuarios.

Otras investigaciones en años posteriores plantean un método para mitigar las fugas de datos mediante la detección del mal uso de la información almacenada. Esa información se

encuentra en bases de datos donde se analiza la sintaxis de las sentencias SQL ejecutadas, para construir perfiles de usuario (Kamra et al., 2008).

La detección de fuga de datos propone soluciones basadas en host y en red. Los basados en host analizan las actividades que puede realizar el atacante en el equipo víctima cuando intenta adueñarse de la información. Posteriormente, previene su salida bloqueando recursos de hardware o de software. Un ejemplo de este tipo de sistemas es el presentado por (Jiang et al., 2010), el mismo que aplica soluciones para detectar malware sigiloso utilizando el escaneo del sistema operativo mediante un antivirus. Otro trabajo de detección de fugas es el realizado por (Borders y Prakash, 2009), que plantea incluir un cifrado de datos y la aplicación de políticas para restringir la transferencia de datos sensibles. Estos métodos pueden complementarse y ser implementados de forma simultánea para mejorar la detección de fugas de información. Uno de los trabajos más conocidos sobre este tema es (Papadimitriou y García-Molina, 2011). Investiga el problema de un distribuidor que ha proporcionado información sensible a un conjunto de agentes supuestamente de confianza (terceros). Algunos de estos datos se han filtrado y se han hallado en un lugar no autorizado (internet, PC, dispositivo de almacenamiento, etc.). El distribuidor debe evaluar la probabilidad de la procedencia de los datos filtrados y de los agentes que lo realizaron. Se propone una estrategia de asignación de datos (a través de los agentes) que mejoran la probabilidad de identificar las fugas. En algunos casos también se puede inyectar datos a los registros, información falsa, para mejorar aún más las probabilidades de detección de fugas y la identificación del agente culpable.

Por otra parte, la solución basada en red detecta filtraciones en las comunicaciones entre los equipos interconectados. Su interfaz debe funcionar como monitor de actividades, el cual captura todo el tráfico de la red analizando el acceso a la información sensible de forma mal intencionada. Un ejemplo es el presentado por (Stann y Heidemann, 2003), donde se realiza un análisis utilizando sensores en redes inalámbricas, explorando la fiabilidad en la capa de transporte, que proporciona una entrega garantizada de la información a su destinatario. Otros trabajos en esta misma área se centran en el análisis de tráfico de red saliente sin cifrar, mediante la inspección profunda de paquetes (Shu y Yao, 2013), o, por ejemplo, un análisis teórico de la información por medio de la entropía (Fawcett, 2010). Sin embargo, requieren el almacenamiento de los datos sensibles en el sistema de detección. En la actualidad también existen investigaciones sobre el filtrado de datos de forma accidental por un usuario legítimo (Jung et al., 2008). Este tipo de fugas puede ser debido a errores humanos, como olvidarse de utilizar el cifrado, el reenvío descuidado de un correo electrónico interno o de archivos adjuntos a agentes externos, etc. El enfoque se basa en la monitorización y análisis de una parte específica del tráfico de red con altas regularidades, el cual identifica qué información se repite generando reglas de comportamiento. El trabajo propone una solución eficiente, que permite localizar un aumento sustancial de la cantidad de nueva información. Este método de detección de fugas de datos identifica desviaciones de los escenarios dentro del flujo de datos normales (Borders y Prakash, 2009).

Investigaciones más recientes se aproximan al problema de fuga de información de diferentes formas. Muchas de estas investigaciones proponen soluciones sofisticadas, usando métodos de "Big Data" (Tene y Polonetsky, 2012), algoritmos de inteligencia artificial, o análisis estadísticos o gráficos. Otras tendencias intentan distinguir y recrear el comportamiento del usuario

a través de una semántica lingüística, el motivo de la comunicación, el correo electrónico o la mensajería instantánea.

Para la evitar la fuga de información actualmente existen dos enfoques para abordar el problema: el prevención de fugas de información (Data Leak Prevention) y la detección de fugas de información (Data leakage Detection).

# 2.2.3.1. Prevención de fugas de información

El primer enfoque evita la fuga de información creando sistemas de prevención de fugas de información o también denominados DLP (Data Leak Prevention). Los DLP monitorizan los recursos informáticos para prevenir la fuga de la información confidencial que se encuentra en discos duros, bases de datos, etc. Los datos confidenciales de las instituciones y de la organización incluyen propiedad intelectual, información financiera, información personal y una variedad de otra información en función de los negocios y la industria como lo presenta.

Los DLP pueden trabajar en dos tipos de identificación de fugas de datos: detectar información sensible que va a ser alterada de alguna manera (ITSELF) y localizar los comportamientos anómalos como muestra en su trabajo (Kim y Kim, 2010) y (Kumar, 2013). Cuando se detecta una posible fuga de información relevante se procede a cifrarla o bien no se permite grabarla en un dispositivo extraíble, o se bloquea el PC, como expone (Raj et al., 2013). Muchos de estos DLP se han instalado en los front ends y utilizan herramientas del propio sistema operativo para escanear las actividades de los usuarios en el ordenador. Estas herramientas son también llamadas agentes. Los agentes tienen identificada la información importante (por ejemplo, números de tarjetas de crédito, códigos de la seguridad social, etc.) y la detectan en los correos electrónicos (si este es el sistema de fuga, por ejemplo), bloqueándola para que no pueda ser enviada.

Por otra parte, existen trabajos como los de (Agarwal y Kiernan, 2002), que utilizan la información que se fuga de las aplicaciones por el correo electrónico, mensajería instantánea y otros canales de Internet. El objetivo es evitar el filtrado de los email utilizando "finger prints" en el cuerpo de los mensajes y generar listas blancas y negras de las direcciones de correo electrónico y además palabras específicas para identificarlos como spam.

En resumen, el control del acceso a la información debe estar garantizado para los usuarios autorizados pero a la vez hay que evitar que la información confidencial pueda ser filtrada. Esto supone la dificultad de tener que discernir la información sensible que se debe monitorizar, o cuándo bloquear la salida de la información. Los actuales DLP de control de información son más bien limitados en términos de flexibilidad por las políticas de seguridad establecidas por los gobiernos e instituciones, las cuales son de obligatorio cumplimiento como lo menciona (Balinsky et al., 2011).

# 2.2.3.2. Detección de fugas de información DLD

El segundo enfoque es el llamado detección de fugas de información DLD (Data Leakage Detection), su función es detectar al agente autorizado el mismo que realiza una transmisión de datos privados (sensibles) o información confidencial a un tercero (receptor no autorizado) de forma no permitida. Los DLD son los encargados de monitorizar el comportamiento de cada uno de

los agentes que interactúan con la información y envía una alerta al momento que se está fugando información como lo presenta (Papadimitriou y Garcia-Molina, 2011).

Uno de los trabajos interesantes sobre esta área es el de (Patil y Bhavsar, 2012), que han desarrollado un sistema de vigilancia de datos y detector de fugas para identificar y prevenir la fuga de información. El trabajo describe dos modelos, el primero interviene cuando un empleado de la organización tiene acceso a datos confidenciales sin el consentimiento del propietario. El modelo se denomina Data Watcher y se utiliza para identificar la fuga de datos. El segundo modelo evalúa el porcentaje de culpabilidad entre las dos partes involucradas de la fuga de la información. Este modelo se centra cuando la fuga se realiza desde fuera de la organización. Además, este modelo utiliza objetos falsos con una herramienta de maracas de agua (Watermark) para mejorar la probabilidad de identificar terceros culpables.

En el trabajo de (Gharpande y Deshmukh, 2013) propone varias estrategias de asignación de datos utilizando a los agentes para mejorar la probabilidad de identificar las fugas de información. Este método no se basa en alteraciones de los datos publicados (por ejemplo, marcas de agua), en algunos casos también pueden inyectar registros de datos "realistas pero falsos" a mejorar aún más nuestras posibilidades de detección de fugas e identificar al culpable. Una idea similar en la propuesta por (Gade y Munde, 2012), propone nuevas estrategias de asignación de datos para mejorar la probabilidad de detectar fugas con precisión. En este trabajo no envían las alteraciones de los datos, sino, realizan una inyección de datos que "parecen genuinos pero son falsos", con el fin de mejorar la probabilidad de detectar fugas y localizar la parte que realmente lo filtró.

Finalmente, cuando la detección se basa en el comportamiento de los usuarios el problema se hace más complejo. Se trata de modelar el comportamiento “normal” de los usuarios, y detectar si las actividades de un usuario autorizado en un momento determinado, son las que cabría esperar o no. Esto puede provocar un número alto de falsos positivos debido a la similitud entre el comportamiento de un usuario autorizado que realiza sus tareas o que pretende filtrar información. Para obtener el patrón de comportamiento de los usuarios se pueden aplicar técnicas de reconocimiento de patrones y clasificación que aprendan de forma dinámica (Torgo, 2011).

# 2.3. Técnicas de inteligencia artificial aplicadas a la seguridad informática

En este apartado se presentan algunas de las técnicas de inteligencia artificial utilizadas en esta investigación; en concreto, las redes neuronales artificiales, árboles de decisión y algoritmos genéticos, entre otras.

## *2.3.1. Redes neuronales artificiales*

Las redes neuronales artificiales son un método computacional que permite resolver problemas de forma individual o conjunta, realizando tareas de clasificación, predicción, identificación, optimización y diagnóstico (Demuth et al., 2014). La unidad elemental de una red neuronal es un procesador local definido como neurona, que posee la característica de calcular una suma de datos de entrada para posteriormente aplicar una función de activación y así obtener una señal que se

transmitirá a las siguientes neuronas. Las neuronas se agrupan en capas (niveles), con un alto grado de conectividad entre cada una de ellas, con pesos en sus conexiones. Las redes neuronales artificiales utilizan un algoritmo de aprendizaje supervisado o no supervisado, según el tipo de red, para ajustar sus pesos y de esta manera minimizar una función de error que indique el grado de ajuste y generalización de los datos.

La principal característica de las redes neuronales es su capacidad de aprender a partir de un conjunto de datos de entrenamiento. Es decir, puede hallar un modelo que se ajuste a los datos aprendidos.

En nuestra investigación hemos aplicado redes neuronales para la detección de intrusos, como se presenta en las publicaciones (Guevara et al., 2013a; Guevara et al., 2013b), donde se ha diseñado un meta clasificador constituido por redes neuronales, máquinas de vectores de soporte y árboles de decisión, para identificar el comportamiento de los usuarios obteniendo buenos resultados y una alta fiabilidad en la detección (Figura 2).

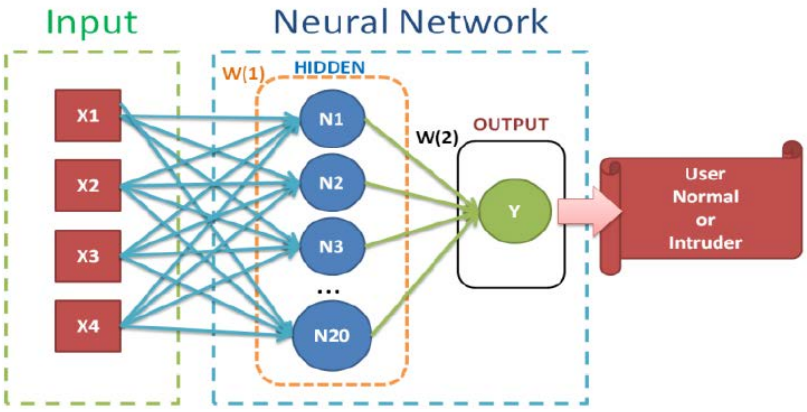



Figura 2. Estructura de una red neuronal para la detección de intrusos (Guevara, 2016b).

## *2.3.2. Árboles de decisión*

Un árbol de decisión es un modelo que mediante aprendizaje inductivo con datos estructurados y construcciones lógicas puede predecir y clasificar. Los árboles de decisión son un modelo muy popular para la clasificación (Chasmer et al., 2014). Para construir un árbol de decisión es necesario describir los elementos que intervienen: los nodos, vectores, arcos y etiquetas. El nodo se puede definir como el momento en el que se ha de tomar una decisión de entre muchas posibles, lo que va haciendo que a medida que aumenta el número de nodos, aumenta el número de posibles finales a los que se puede llegar. Esto hace que un árbol con muchos nodos sea complicado de analizar debido a la existencia de numerosos caminos que se pueden seguir. Los vectores son la solución final a la que se llega. Los arcos son las uniones entre un nodo y otro, y representan cada acción. Las etiquetas se encuentran en cada nodo y cada arco y dan nombre a cada acción.

Dentro de los árboles de decisión existen dos algoritmos muy utilizados para la clasificación, son el C4.5 y el ID3. Los árboles de decisión C4.5 utilizan aprendizaje basado en similaridades. Estos algoritmos generan un árbol de decisión de forma recursiva al considerar el criterio de la mayor proporción de ganancia de información, es decir, elige al atributo que mejor clasifica a los datos. Los

árboles de decisión ID3 utilizan la búsqueda de hipótesis o reglas, a partir de un conjunto de ejemplos. Este conjunto de ejemplos deberá estar conformado por una serie de tuplas de valores, cada una de ellas denominadas atributo, en el que en uno de ellos, el atributo a clasificar, es el objetivo y suele ser de tipo binario: positivo o negativo, sí o no, válido o inválido, etc. De esta forma el algoritmo trata de obtener las hipótesis que clasifiquen, ante nuevas instancias, si dicho ejemplo va a ser positivo o negativo.

Los árboles de decisión han sido utilizados en varias publicaciones en nuestra investigación (Guevara et al., 2013a; Guevara et al., 2013b; Guevara et al., 2014a; Guevara et al., 2014c; Guevara et al., 2017), donde se han desarrollado múltiples modelos para la detección de intrusos. Se han diseñado meta clasificadores, (Guevara et al., 2013a; Guevara et al., 2014a; Guevara et al., 2014c), donde la combinación de varias técnicas de clasificación mejora el resultado de la detección y baja la tasa de falsos positivos. Por otra parte otras publicaciones, como (Guevara et al., 2013b; Guevara et al., 2017), presentan la aplicación de árboles como una técnica de comprobación de la detección, obteniendo resultados satisfactorios en tu porcentaje de precisión.

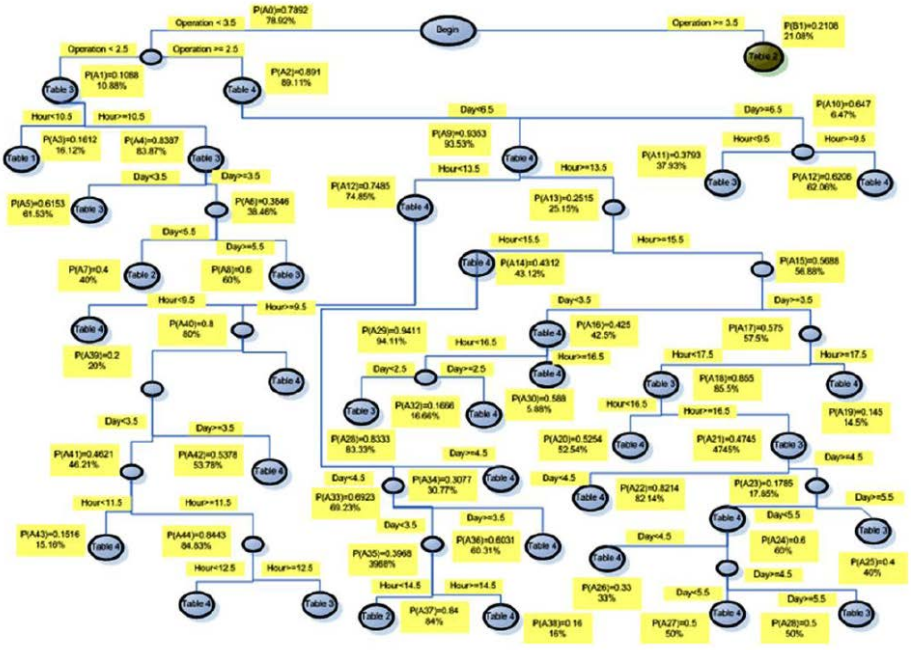



Figura 3. Árbol de decisión para la detección de intrusos (Guevara, 2016b).

## *2.3.3. Algoritmos genéticos*

Los algoritmos genéticos son métodos adaptativos de búsqueda y optimización. Están basados en la evolución genética de los organismos vivos. A lo largo de las generaciones, las poblaciones evolucionan en la naturaleza acorde con los principios de la selección natural y la supervivencia de los más fuertes. Tomando como base esta idea, los algoritmos genéticos son capaces de ir creando soluciones para problemas del mundo real. La evolución de dichas soluciones lleva a valores óptimos del problema, que depende en buena medida de una adecuada configuración del algoritmo. Los principios básicos de los algoritmos genéticos fueron establecidos por (Goldberg and Holland, 1988). Es necesaria una codificación o representación de las soluciones. Además se requiere una función de

ajuste o adaptación del problema, la cual asigna una evaluación a cada posible solución (individuo) codificada. Durante la ejecución del algoritmo, algunos individuos son seleccionados para la reproducción, que se cruzarán generando hijos, sobre los cuales puede actuar un operador de mutación. El resultado de la combinación de los anteriores operadores genéticos será un nuevo conjunto de individuos (posibles soluciones al problema) supuestamente mejores que los anteriores, los cuales en la evolución del algoritmo genético formarán parte de la siguiente población.

En nuestra investigación se utilizan los algoritmos genéticos, específicamente los sistemas de clasificación supervisada UCS. En (Guevara et al., 2014c) se combinan las técnicas de árboles de decisión C4.5 y UCS y se aplican a una base de datos Amazon para identificar accesos informáticos de los usuarios, con buenos resultados.

# Capítulo III. Descripción de las bases de datos de los experimentos

En esta investigación se han utilizado tres bases de datos que han sido el pilar fundamental para comprobar nuestras propuestas y su eficiencia. Estas bases de datos son tanto públicas y privadas. Las bases de datos públicas utilizadas son: Unix Comands, KDD 99 y Amazon; por otro lado, la base de datos privada pertenece al gobierno ecuatoriano. A continuación describimos cada una de ellas de forma más detallada.

## 3.1 Base de datos Unix comands

Se ha utilizado la base de datos de Unix (Aeberhard et al., 1994) para algunos experimentos. Contiene 9 conjuntos de datos de usuarios extraídos de los históricos de ejecución de comandos de 8 usuarios con ordenadores UNIX en un periodo de 2 años (UNIX User Data Set). La información que contiene son los nombres de los comandos, banderas y meta-caracteres de shell. Además, existen etiquetas como ** SOF ** y ** EOF ** las cuales se han insertado al inicio y al final de las sesiones de usuario para identificar su dimensión. Las sesiones están ordenadas por fecha y los símbolos aparecen en el orden de ejecución dentro de cada sesión, pero no se incluyen marcas de tiempo.

Para obtener buenos resultados en nuestros experimentos se utilizaron dos tipos distintos de conjuntos: uno para entrenar el modelo propuesto y otro para realizar pruebas. También trabaja con un tercer grupo de datos para validar cada una de nuestras propuestas. Los conjuntos de datos se detallan de la siguiente manera:

**Training data:** Este conjunto de datos está formado por el 60% de las sesiones de cada uno de los usuarios. Son empleados para entrenar el algoritmo.

**Testing data:** Este conjunto de datos se utiliza para evaluar funcionamiento (detección) del sistema de anomalías entrenado anteriormente con datos de cada uno de los usuarios.

**Validation data:** El conjunto de datos de validación es creado a partir de información de testing de un usuario específico y de otro usuario diferente. Esta información permite comprobar el funcionamiento del algoritmo propuesto.

Tabla 1. Distribución de sesiones por usuarios en la base de datos UNIX.

| Usuarios | Sesiones | Comandos | Longitud Máxima Sesión |
|---|---|---|---|
| USR_0 | 567 | 7840 | 214 |
| USR_1 | 515 | 18851 | 429 |
| USR_2 | 1069 | 16600 | 353 |
| USR_3 | 501 | 15864 | 1865 |
| USR_4 | 955 | 35905 | 957 |
| USR_5 | 582 | 33657 | 1400 |
| USR_6 | 3419 | 57314 | 434 |
| USR_7 | 1522 | 14285 | 446 |
| USR_8 | 1983 | 50076 | 564 |
| **TOTAL** | 11113 | 250392 | |

En la tabla 1 se resumen algunas de las características de la base de datos Unix. Los usuarios 0 y 1 son realmente la misma persona. Ejemplos de algunos comandos son: cd, ls, more, cat, vi, etc. La sesión más corta es de un solo comando. La media en longitud de una sesión es de 700, aunque depende de cada usuario. Nosotros hemos trabajado con 5 comandos distintos para realizar nuestra base de datos para el entrenamiento del modelo. La base de datos contiene comandos, banderas y otros tipos de información, pero nos hemos orientado a los comandos más ejecutados por el usuario en el sistema operativo Unix para determinar su comportamiento.

Para nuestra investigación se utiliza toda la base de datos, pero un 60% para entrenamiento y un 40% para pruebas del modelo propuesto. El número de registros en la base de datos es de 400000, con 42 atributos diferentes. Con tanta información el entrenamiento de nuestros modelos ha requerido un procesamiento bastante difícil ya que quiere de mucho tiempo para realizarlo.

En nuestra propuesta para la detección de comportamiento anómalo en comportamientos de usuario nosotros hemos visto necesario aplicar una estructura de datos secuencial, donde se puede evidenciar de mejor manera los comportamientos fraudulentos. Esta nueva estructura de datos contempla un inicio de sesión etiquetado como "** SOF **", seguida de una ejecución secuencial de comandos UNIX (de tamaño variable). Para terminar, el fin de la sesión se ha etiquetado como "** EOF **", como se puede apreciar en la figura 4.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| ** SOF ** | f | <1> | f | <1> | f | <1> | ** EOF ** |
| ** SOF ** | elm | telnet | <1> | f | <1> | ** EOF ** | |
| ** SOF ** | cd | <1> | cd | <1> | ls | cd | ** EOF ** |
| ** SOF ** | telnet | <1> | lo | ** EOF ** | | | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| ** SOF ** | f | <1> | date | telnet | <1> | f | ** EOF ** |
| ** SOF ** | f | <1> | telnet | <1> | elm | ** EOF ** | |
| ** SOF ** | elm | vt100 | elm | telnet | <1> | telnet | ** EOF ** |
| ** SOF ** | f | <1> | date | elm | ls | ** EOF ** | |
| ** SOF ** | elm | f | <1> | date | elm | fg | ** EOF ** |
| ** SOF ** | elm | f | <1> | date | telnet | <1> | ** EOF ** |
| ** SOF ** | lo | ** EOF ** | | | | | |
| ** SOF ** | elm | lo | ** EOF ** | | | | |

Figura 4. Muestra de estructura secuencial de datos de la base de datos UNIX.

Para nuestro estudio vamos a partir del trabajo realizado por (Aeberhard, 1994), el cual utiliza 7 usuarios de la base de datos UNIX, además, toma como atributos para la clasificación cinco comandos, los cuales son "finger", "man", "who", "date" y "more". Pero hemos utilizado otros comandos más para que la clasificación sea más precisa, esos comandos fueron seleccionados debido al número de veces que fueron ejecutados por los usuarios, lo comandos adicionales son "cd", "ls", "elm", "rm", "grep", "make", "mv", "cp", "less" y "cat". Por otro lado la estructura de la tabla para la clasificación va ser basada en la estructura descrita en la figura 4. Como se muestra en la tabla 5, se puede apreciar la distribución de frecuencia de ejecución de comandos realizados por el usuario #1. Adicionalmente, se va a realizar una tabla de datos para cada uno de los usuarios, donde las sesiones serán identificadas como "normal" si pertenece al usuario respectivo y al resto de usuarios serán etiquetadas como "anormal".

Tabla 5. Tabla de datos de usuario 1, con 5 comandos básicos y 11 comandos adicionales.

| | Comandos básicos | | | | | Comandos adicionales | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Sesión | finger | man | who | date | more | cd | ls | elm | rm | grep | make | mv | cp | less | cat | chmod | label |
| 1 | 2 | 0 | 0 | 0 | 0 | 3 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | normal |
| 2 | 0 | 0 | 0 | 4 | 0 | 0 | 5 | 1 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | normal |
| 3 | 1 | 1 | 0 | 0 | 3 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | normal |
| 4 | 0 | 2 | 0 | 0 | 0 | 1 | 2 | 1 | 0 | 0 | 0 | 6 | 0 | 0 | 0 | 0 | normal |
| 5 | 0 | 0 | 2 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 7 | 0 | 0 | 0 | 0 | anormal |
| 6 | 0 | 0 | 3 | 0 | 6 | 0 | 0 | 0 | 7 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | anormal |
| 7 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 3 | 0 | 0 | 0 | 0 | 0 | 5 | 0 | 0 | anormal |
| 8 | 0 | 3 | 0 | 0 | 1 | 5 | 4 | 0 | 0 | 0 | 0 | 4 | 0 | 0 | 0 | 0 | anormal |
| 9 | 1 | 4 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 3 | 0 | 0 | 1 | 0 | anormal |

# 3.2 Base de datos KDD Cup 1999

Este conjunto de datos fue utilizado para el “The Third International Knowledge Discovery and Data Mining Tools Competition”, que se realize dentro del “Fifth International Conference on Knowledge Discovery and Data Mining”. El objetivo del concurso era construir un detector de intrusiones en la red, es decir, un modelo predictivo capaz de distinguir entre conexiones fraudulentas, llamadas intrusivas, y conexiones normales. El Programa de Evaluación de Detección de Intrusiones de DARPA

de 1998 fue preparado y administrado por MIT Lincoln Labs. El objetivo fue estudiar y evaluar la investigación sobre detección de intrusos. Se proporcionó un conjunto estándar de datos a ser auditados, que incluye una amplia variedad de intrusiones simuladas en un entorno de red militar. El concurso KDD de 1999 utiliza una versión de este conjunto de datos. Lincoln Labs creó un entorno para generar nueve semanas de datos de volcado TCP sin procesar para una red de área local (LAN) de la Fuerza Aérea de Estados Unidos. Ellos operaban la LAN como si fuera de la Fuerza Aérea, pero simulaban múltiples ataques. Los datos de entrenamiento en bruto eran aproximadamente cuatro gigabytes de datos binarios comprimidos de volcado TCP de siete semanas de tráfico de red. Esto se procesó en unos cinco millones de registros de conexión. Del mismo modo, las dos semanas de datos de prueba produjeron alrededor de dos millones de registros de conexión. Una conexión es una secuencia de paquetes TCP que comienzan y terminan en unos tiempos bien definidos, entre los cuales los datos fluyen hacia y desde una dirección IP de origen a una dirección IP de destino bajo algún protocolo bien definido. Cada conexión se etiqueta como normal o como ataque, con un tipo de ataque específico. Cada registro de conexión consta de aproximadamente 100 bytes. Los ataques se dividen en cuatro categorías principales:

- DOS: denegación de servicio.
- R2L: acceso no autorizado desde una máquina remota.
- U2R: acceso no autorizado a privilegios de super usuario (root) locales.
- Probing (sondeo): vigilancia y otros sondeos, por ejemplo, exploración de puertos.

Es importante señalar que los datos de prueba no tienen de la misma distribución de probabilidad que los datos de entrenamiento e incluyen tipos de ataque específicos que no están en los datos de entrenamiento. Esto hace que la tarea sea más realista. Las características de cada uno de los atributos de la base de datos están detalladas en una lista completa que se presenta en las tres siguientes tablas:

Tabla 2. Características básicas de las conexiones individuales TCP.

| ***Nombre del atributo*** | ***Descripción*** | ***Tipo*** |
|---|---|---|
| Duration | Longitud (número de segundos) de la conexión | continuo |
| protocol_type | Tipo del protocolo, e.g. tcp, udp, etc. | discreto |
| Service | Servicio de red sobre el destino, e.g., http, telnet, etc. | discreto |
| src_bytes | Número de bytes de datos de origen a destino | continuo |
| dst_bytes | Número de bytes de datos de destino a fuente | continuo |
| Flag | Estado normal o de error de la conexión | discreto |
| Land | 1 si la conexión es al mismo host / de lo contrario es 0 | discreto |
| wrong_fragment | Número de fragmentos erróneos | continuo |

| | | |
|---|---|---|
| Urgent | Número de paquetes urgentes | continuo |

Tabla 3. Características de contenido dentro de una conexión sugerida.

| **Nombre del atributo** | **Descripción** | **Tipo** |
|---|---|---|
| Hot | Número de indicadores "hot" | continuo |
| num_failed_logins | Número de intentos de inicio de sesión fallidos | continuo |
| logged_in | 1 si ha iniciado sesión correctamente, de lo contrario 0. | discreto |
| num_compromised | Número de condiciones "comprometidas" | continuo |
| root_shell | 1 si se obtiene root shell, caso contrario 0 | discreto |
| su_attempted | 1 si se ha intentado el comando "su root '', de lo contrario 0 | discreto |
| num_root | Número de accesos "root '' | continuo |
| num_file_creations | Número de operaciones de creación de archivos | continuo |
| num_shells | Número de solicitudes de shell | continuo |
| num_access_files | Número de operaciones en archivos de control de acceso | continuo |
| num_outbound_cmds | Número de comandos de salida en una sesión de ftp | continuo |
| is_hot_login | 1 si el inicio de sesión pertenece a la lista "hot", caso contrario 0 | discreto |
| is_guest_login | 1 si el inicio de sesión es un "guest'", caso contrario 0 | discreto |

Tabla 4. Características de tráfico utilizando una ventana de tiempo de dos segundos.

| **Nombre del atributo** | **Descripción** | **Tipo** |
|---|---|---|
| Count | Número de conexiones al mismo host que la conexión actual en los últimos dos segundos | continuo |

| | | |
|---|---|---|
| Nota: Las siguientes características se refieren las mismas conexiones de host. | | |
| serror_rate | % de conexiones que tienen errores "SYN'' | continuo |
| rerror_rate | % de conexiones que tienen errores "REJ'' | continuo |
| same_srv_rate | % de conexiones con el mismo servicio | continuo |
| diff_srv_rate | % De conexiones con el diferente servicio | continuo |
| srv_count | Número de conexiones al mismo servicio que la conexión actual en los últimos dos segundos | continuo |
| Nota: Las siguientes características se refieren a conexiones de servicio. | | |
| srv_serror_rate | % de conexiones que tienen errores "SYN'' | continuo |
| srv_rerror_rate | % de conexiones que tienen errores "REJ'' | continuo |
| srv_diff_host_rate | % de conexiones a diferentes hosts | continuo |

# 3.3 Base de datos Amazon access

Se trata de un conjunto de datos reducido, menos del 10% de los atributos se utilizan para cada muestra. Este conjunto de datos está dividido en dos archivos: uno que contiene el acceso para los usuarios y el otro que contiene el historial de acceso de un usuario específico. Contiene usuarios y su acceso asignado (Amazon Access Samples Data Set). El archivo tiene 4 categorías de atributos:

1. [PERSON_{ATTRIBUTE}]: Esta categoría describe el 'usuario' al que se le dio acceso. La columna [PERSON_ID] es la columna de clave principal del archivo. Hay una fila por usuario.
    - PERSON_ID: Id del usuario
    - PERSON_MGR_ID: Id del administrador del usuario
    - PERSON_ROLLUP_1: ID de agrupación de usuarios
    - PERSON_ROLLUP_2: ID de agrupación de usuarios
    - PERSON_ROLLUP_3: ID de agrupación de usuarios
    - PERSON_DEPTNAME: ID de la descripción del departamento
    - PERSON_LOCATION: Id de la región
    - PERSON_BUSINESS_TITLE: Id del título
    - PERSON_BUSINESS_TITLE_DETAIL: Id de la descripción
    - PERSON_JOB_CODE: Id del código de trabajo
    - PERSON_COMPANY: Id de la compañía
    - PERSON_JOB_FAMILY: Id de la familia de trabajos

2. [RESOURCE_ {ID}]: Esta categoría de atributos corresponde a los recursos a los que un usuario puede tener acceso. Un usuario tendrá un 1 en esta columna si tiene acceso a ella, 0 de lo contrario será 0.
3. [GROUP_ {ID}] - Esta categoría de atributos son los grupos a los que un usuario puede tener acceso. Un usuario tendrá un 1 en esta columna si tiene acceso a ella, de lo contrario será 0.
4. [SYSTEM_SUPPORT_ {ID}] - Esta categoría de atributos es el sistema del que un usuario puede tener soporte. Un usuario tendrá un 1 en esta columna si el usuario tiene soporte del sistema, de lo contrario será 0.

Existen múltiples columnas en este conjunto de datos, una breve descripción de una de ellas es:

- ACTION: o bien 'remove_access' o 'add_access'.
- TARGET_NAME: {RESOURCE_ID} o {GROUP_ID}.
- LOGIN: el id del usuario que está obteniendo o perdiendo acceso.
- REQUEST_DATE: (Fecha/hora) formato año/mes/día/hora: minuto: segundo.
- AUTHORIZATION_DATE: fecha y hora de autorización, formato año/mes/día/hora:minuto: segundo.

Este conjunto de datos contiene 30000 registros y 20000 atributos en todo el conjunto de datos. En nuestra investigación se ha utilizado solo el 10% para nuestro modelo.

# 3.4 Base de datos Ecuador

La información se ha tomado de una base de datos de una entidad pública de la República del Ecuador, posee múltiples usuarios que interactúan con un sistema donde realizan una variedad de tareas y operaciones. El esquema de la base de datos se presenta en la figura 5.

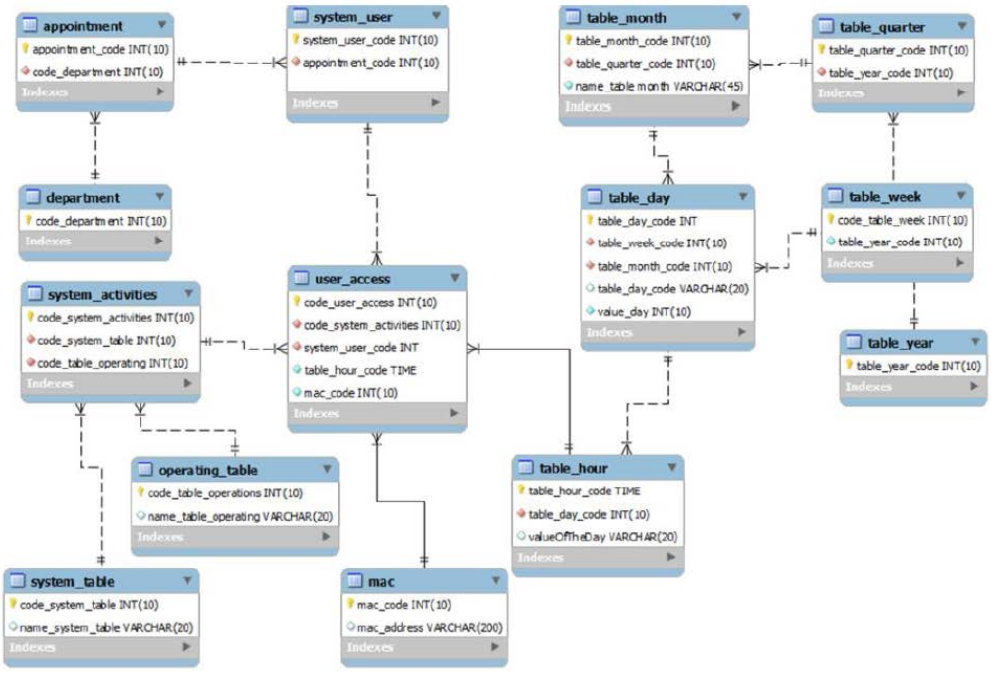



Figura 5. Esquema de la base de datos Ecuador (Guevara, 2016b).

En la base de datos existen varios atributos que permitirán definir un patrón de comportamiento del usuario. Esta información describirá las actividades que realiza dentro del sistema, como también fecha, hora, y la estación de trabajo que utiliza regularmente en su acceso. Los siguientes son algunos de los atributos que se pueden obtener de dicha base de datos tabla 6.

Tabla 6. Tabla de atributos de la base de datos Ecuador.

| Nombre atributo | Tipo de dato | Descripción |
|---|---|---|
| Fecha de ingreso | Date Time | Indica la fecha y hora en la cual el usuario ingresa al sistema. |
| IP | Varchar | Proporciona la IP de la estación de trabajo en la cual realiza el ingreso al sistema. |
| País | Varchar | Muestra el país desde cual realiza el ingreso al sistema. |
| Fecha de Actividad | Date Time | Fecha y hora en la cual el usuario realiza las tareas de Insertar, Actualizar y Eliminar en la base de datos. |
| Tabla afectada | Varchar | Muestra la tabla en la cual realiza operaciones. |
| Operación | Varchar | Operaciones que realiza el usuario con los registros. |
| Usuario | Varchar | Presenta el usuario que realiza las actividades en la base de datos. |

Para este estudio se han tomado una muestras de cada una de las actividades y tareas nombradas anteriormente, y se han creado gráficas que brindan una visualización de la información de una manera más simple y sencilla. Como ejemplo se presenta, para un periodo de tres meses y de un solo usuario, la variable operación (figura 6).

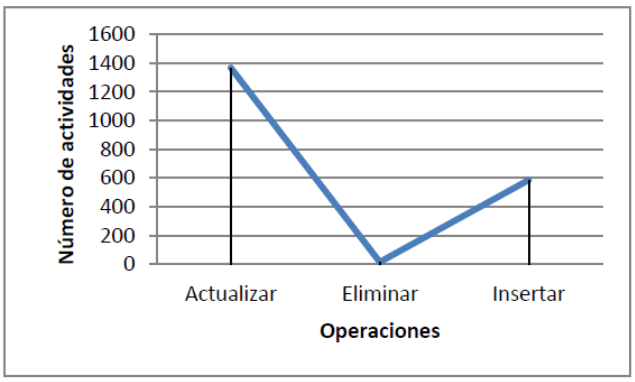



Figura 6. Gráfica de la variable operación de la base de datos Ecuador.

En la figura 7 se presentan las operaciones que el usuario #6 ha realizado sobre la base de datos Ecuador. Como se puede observar, la mayor de este usuario es realizar las operaciones de actualizar y de insertar que las de eliminar, pero no se descarta ésta última ya que la combinación de estas tres operaciones puede ser muy importante para un futuro patrón de comportamiento.

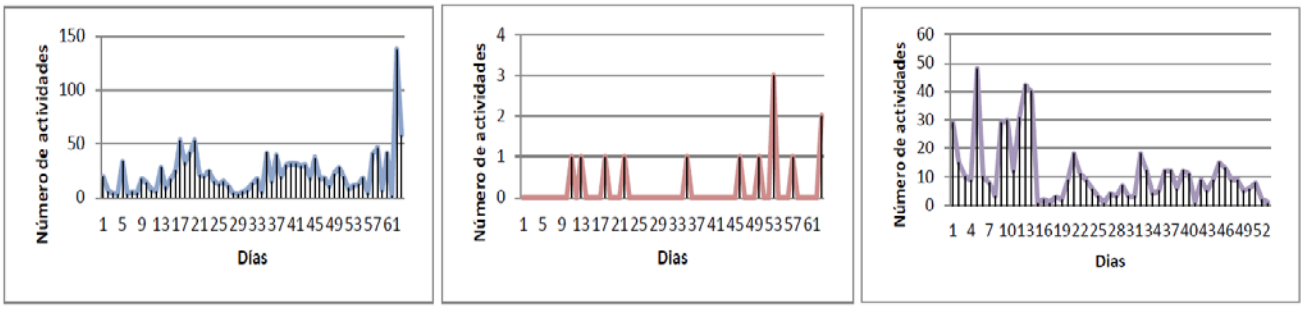



Figura 7. Gráfica de usuario #6 en un periodo de 2 meses con las operaciones de actualizar, eliminar e insertar, de izquierda a derecha.

Otra variable es tabla accedida: describe las tablas sobre las que se realizan actividades dentro del sistema y se han denominado Tabla 1, 2, 3 y 4 para mantener la confidencialidad de la información que está registrada en cada una de ellas. En la figura 8 se presenta la distribución de la información de los usuarios por tablas a los que han accedido. En la figura 9 se presenta el comportamiento del usuario #6 respecto a cada una de las tablas en un periodo de tiempo.

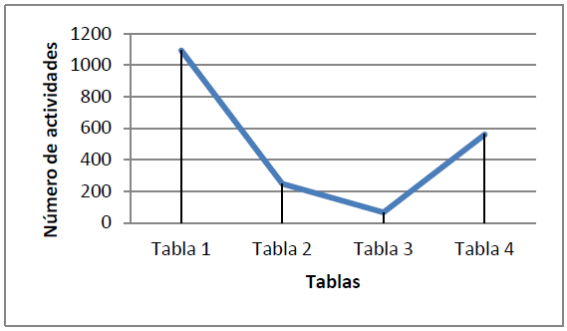



Figura 8. Gráfica de acceso a las diferentes tablas de la base de datos.

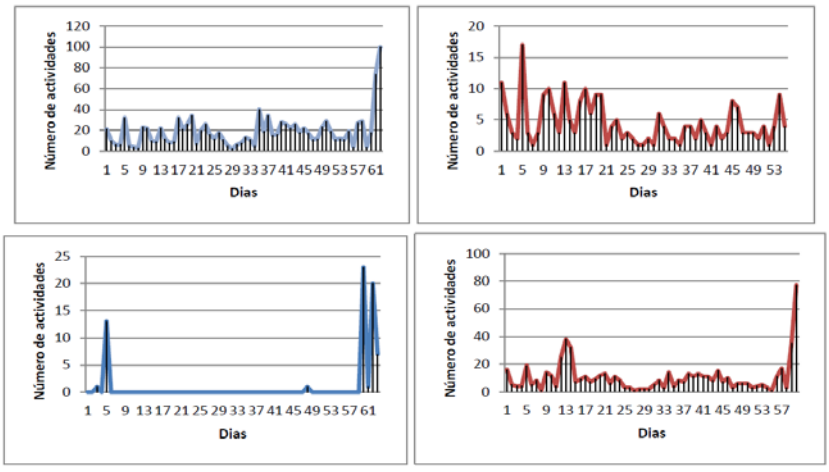



Figura 9. Gráfica del usuario #6 en 2 meses a las tablas 1, 2, 3 y 4, de izquierda a derecha, de arriba y abajo.

La variable fecha: proporciona informaciones de los días de la semana en los que se realizan operaciones dentro del sistema. Se representa en la figura 10.

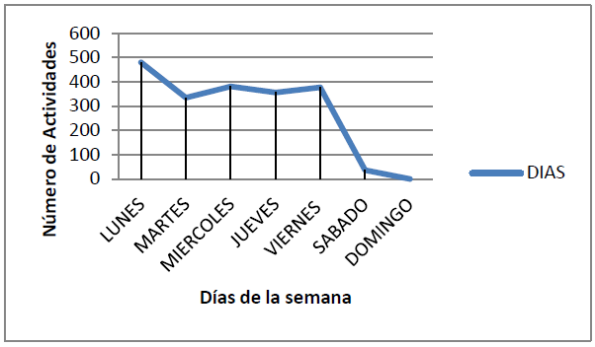



Figura 10. Operaciones realizadas por los usuarios por día de la semana.

La variable hora, de igual forma, muestra los accesos durante la jornada laboral a las estaciones de trabajo (Figura 11).

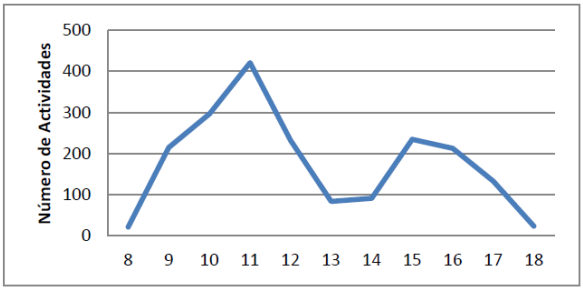



Figura 11. Operaciones realizadas por los usuarios durante la jornada laboral.

Por último, la variable estación de trabajo (Figura 12) representa la dirección IP de la estación de trabajo desde la que realizan operaciones.

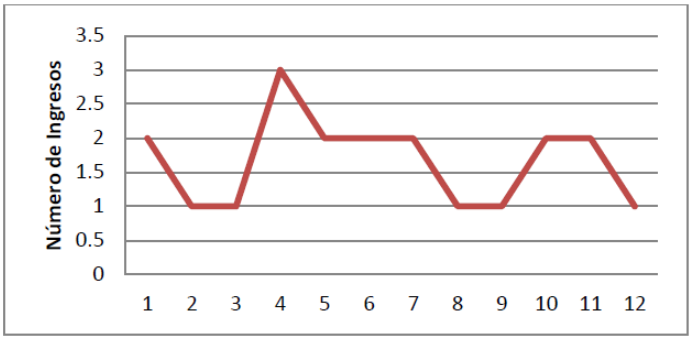



Figura 12. IP de acceso por los usuarios.

Para desarrollar una nueva estructura de datos dinámica que describa el comportamiento del usuario, es necesario realizar una fusión de información, la que presenta en la figura 13. Para posteriormente obtener una estructura secuencial y dinámica del comportamiento del usuario (figura 14).

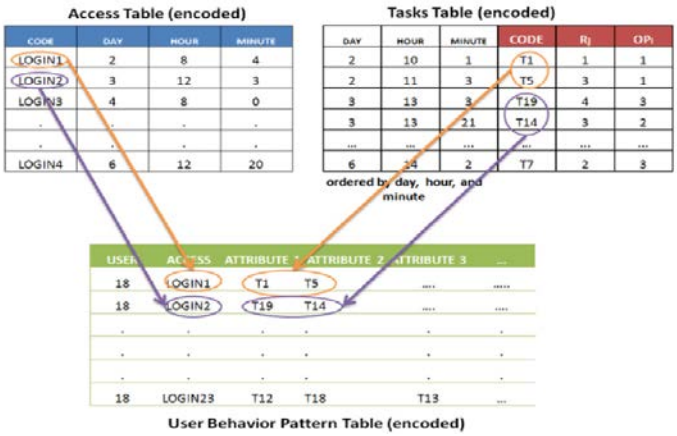



Figura 13. Estructura de datos de tareas realizadas por el usuario en la base de datos Ecuador (Guevara, 2017).

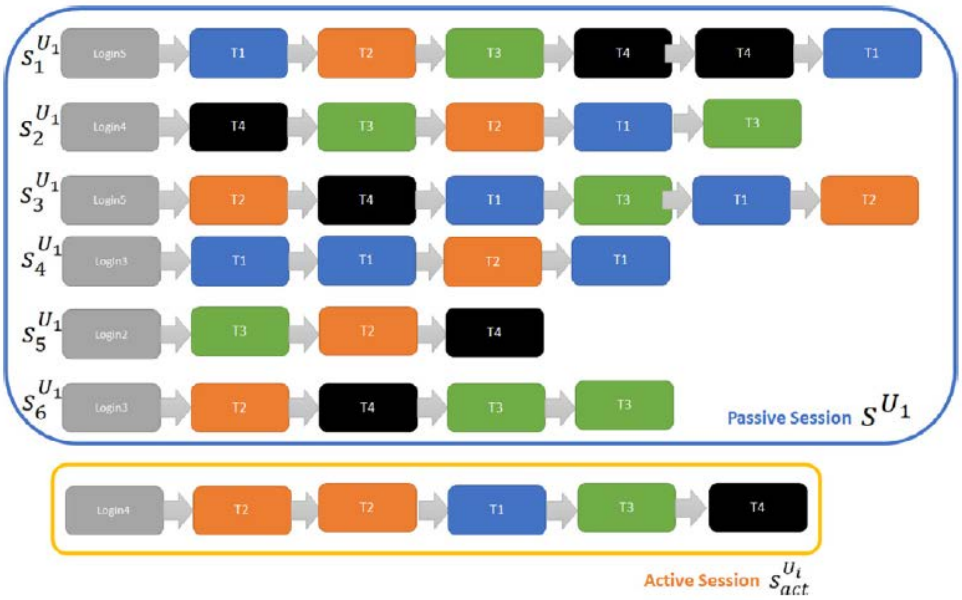



Figura 14. Diagrama de propuesta de estructura de datos secuencial con la base de datos Ecuador (Guevara, 2017).

# Capítulo IV. Relación de publicaciones y discusión integradora

En este capítulo se presentan las publicaciones asociadas a la investigación doctoral aunadas según las aportaciones relativas a los sistemas de seguridad implementados.

También en esta investigación se ha trabajado con varios conjuntos de datos reales, tanto de red como de sistemas informáticos. Este análisis de cada uno de los conjuntos de datos se ha desarrollado, en el capítulo anterior. Este estudio de la información de los usuarios permitió desarrollar diversos modelos para la detección de intrusos y de anomalías en el comportamiento de los usuarios. Además, fue la base fundamental para la creación de la nueva estructura de datos secuencial, la cual permite describir de forma mucho más real y dinámica el comportamiento de un usuario. Dicha estructura fue utilizada en los artículos (Guevara et al., 2016a; Guevara et al., 2017), con resultados bastante buenos y un funcionamiento eficiente cumpliendo con dos de los principales objetivos planteados en el trabajo doctoral.

Otro de los objetivos que se ha abordado en este trabajo doctoral ha sido el análisis, diseño y desarrollo de meta clasificadores para la detección de intrusos, (Guevara et al., 2013a; Guevara et al., 2014ª; Guevara et al., 2014c). La combinación de una variedad técnicas de clasificación supervisada ha mejorado la tasa de precisión de la detección, con un porcentaje muy bajo de la tasa de falsos positivos y una eficiencia del clasificador bastante buena. Por ejemplo, en la publicación (Guevara et al., 2014c) se combinan las técnicas de árboles de decisión C4.5 y UCS y se aplican a una base de datos Amazon para identificar accesos informáticos de los usuarios, con buenos resultados.

Otro de los objetivos de esta investigación doctoral fue el desarrollo de un algoritmo que aplique teoremas estadísticos para la detección de ataques en un sistema informático. Se propuso un algoritmo de detección de anomalías y fugas de información en la publicación (Guevara et al., 2017). Ahí se describe la aplicación de cadenas de Markov y el teorema de Bayes para identificar anomalías en la ejecución secuencial de tareas. Este algoritmo es la principal aportación al trabajo doctoral debido a su impacto, originalidad y aplicación en el mundo real.

Finalmente, se ha desarrollado un algoritmo de detección de anomalías en el comportamiento de los usuarios aplicando técnicas bio-inspiradas como la selección negativa. Este algoritmo fue publicado en el artículo (Guevara et al., 2016a). Utiliza el histórico de la ejecución de tareas de cada uno de los usuarios con comportamiento normal para generar secuencias de tareas anómalas (intrusivas). Esto permitió que la detección de posibles desviaciones del comportamiento habitual de los usuarios pueda ser identificada de una forma más rápida y eficiente, cumpliendo con el último objetivo planteado en la tesis doctoral.

# 4.1. Publicaciones y premios derivados de la Tesis

## Congresos

| | |
|---|---|
| 1) Autores: | C. Guevara, M. Santos, J. A. Martín |
| Título: | Método para la detección de intrusos basado en la sinergia de técnicas de Inteligencia Artificial |
| Congreso: | Congreso Español de Informática CEDI 2013. |
| Lugar y fecha: | Madrid-España; 17 al20 septiembre 2013. |
| Índice de calidad: | Congreso nacional |

| | |
|---|---|
| 2) Autores: | C. Guevara, M. Santos, J. A. Martín-H |
| Título: | Identification of computer information system intruders by decision trees and artificial neural networks |
| Congreso: | 2013 ISKE 8th Int. Conference on Intelligent System and Knowledge Engineering |
| Lugar y fecha: | ShenZhen, China, 20 al 23 de noviembre de 2013 |
| Índice de calidad: | Congreso "Core B" |

| | |
|---|---|
| 3) Autores: | C. Guevara, M. Santos, J. Martin, V. López |
| Título: | Reaching a consensus on access detection by a making decision system |
| Congreso: | International Conference on Progress in Informatics and Computing PIC 2014 DOI: 10.1109/PIC.2014.6972308 |
| Lugar y fecha: | Shaghai - China, 16 al 18 mayo de 2014. |
| Índice de calidad: | Congreso IEEE |

| | |
|---|---|
| 4) Autores: | C. Guevara, M. Santos, V. López |
| Título: | Training strategy to improve the efficiency of an intelligent detection system |
| Congreso: | International FLINS Conference on Decision Making and Soft Computing |
| Lugar y fecha: | João Pessoa (Paraíba), Brasil, 17 al20 de agosto de 2014. |
| Índice de calidad: | Congreso "Core B" |

| | |
|---|---|
| 5) Autores: | C. Guevara, M. Santos, V. López |
| Título: | Data leakage detection algorithm based on sequences of activities |
| Congreso: | International Symposium on Research in Attacks RAID 2014 |
| Lugar y fecha: | Gothenburg-Sweden, 17 al 19 de septiembre de 2014. |
| Índice de calidad: | Congreso "Core A" |

| | |
|---|---|
| 6) Autores: | C. Guevara, M. Santos, V. López |
| Título: | Sistema de detección de intrusos aplicando selección negativa en perfiles de usuario |
| Congreso: | I Jornadas Nacionales de Investigación en Ciberseguridad |
| Lugar y fechas: | León- España ,14 al 16 de septiembre de 2015. |

| | |
|---|---|
| ISBN: | 978-84-9773-742-5; RIASC, Universidad de León, INCIBE |
| Volumen, pág. (año): | 8-13, 2015 |
| Índice de calidad: | Congreso nacional |

| | |
|---|---|
| 7) Autores: | C. Guevara, M. Santos, V. López |
| Título: | Sistema inmune artificial para la detección de comportamientos anómalos de usuarios en sistemas informáticos |
| Congreso: | CAEPIA 2015 XVI Conferencia de la Asociación Española para la Inteligencia Artificial |
| Lugar y fechas: | Albacete - España, 09 al 12 de noviembre de 2015. |
| Publicación: | Actas de la XVI Conferencia de la Asociación Española para la Inteligencia Artificial |
| ISBN: | 978-84-608-4099-2 |
| Volumen, pág. (año): | 209-218, 2015 |
| Índice de calidad: | Congreso nacional |

| | |
|---|---|
| 8) Autores: | C. Guevara, M. Santos, V. López |
| Título: | Intrusion detection with neural networks based on knowledge extraction by decision trees |
| Congreso: | 9th International Conference on Computational Intelligence in Security for Information Systems CISIS 2016. DOI: 10.1007/978-3-319-47364-2_49 |
| Lugar y fecha: | San Sebastián - España, 19 al 21 de octubre de 2016. |
| Índice de calidad: | Congreso "Core B" |

## Revistas

| | |
|---|---|
| 9) Autores: | C. Guevara, M. Santos, V. López |
| Título: | Negative selection and Knuth Morris Pratt algorithm for anomaly detection |
| Revista: | IEEE Latin America Transactions. ISSN: 1548-0992. Doi: 10.1109/TLA.2016.7459637 |
| Volumen, pág. (año): | 2016 |
| Índice de Calidad: | JCR 2015 : 0.436 (2015) Engineering, Electrical & Electronic 219/257 (Q4) |

| | |
|---|---|
| 10) Autores: | C. Guevara, M. Santos, V. López |
| Título: | Data leakage detection algorithm based on task sequences and probabilities |
| Revista: | Knowledge-Based Systems Elsevier, ISSN: 0950-7051. Doi: 10.1016/j.knosys.2017.01.009 |
| Volumen, pág. (año): | in press, 2017 |
| Índice de calidad: | JCR 2015: 3.325 (2015) Computer Science, Artificial Intelligence 17/130 (Q1) |

### 4.1.1. Premios

1. “Sistema de detección de intrusos aplicando selección negativa en perfiles de usuario", Ganador del premio al mejor artículo de estudiante en las Jornadas Nacionales de Investigación en Ciberseguridad JNIC 2015, del 14 al 16 de septiembre de 2015. León, España.
2. "Sistema de detección de fugas de información ILLALPA", Finalista en la presentación de proyectos Cyber Camp 2014, 1 al 4 diciembre 2014. Madrid, España.
3. "Identification of computer information system intruders by decision trees and artificial neural networks", Premio al mejor poster de artículo científico en 8th International Conference on Intelligent Systems and Knowledge Engineering (ISKE 2013), noviembre 2013. ShenZhen, China

## 4.2. Publicaciones sobre sistema de detección de intrusos

En esta investigación doctoral se marcó como objetivo prioritario la disminución del porcentaje de falsos positivos en la detección de intrusos. Esto se ha conseguido con la aplicación de técnicas de inteligencia artificial y la nueva estructura de datos propuesta. En esta sección se describirán de forma breve cada uno de los artículos publicados y las aportaciones realizadas en cada uno de ellos.

Así, en el artículo publicado en el congreso **CEDI 2013** (Guevara et al., 2013a), se presenta la creación de un meta-clasificador con la combinación de varias técnicas inteligentes de clasificación como son: la red neuronal multicapa, máquinas de vectores soporte y árboles de decisión. Se desarrolló un sistema de detección de intrusos para la base de datos KDD 99. El sistema consigue una alta eficiencia y fiabilidad en la detección de intrusos en las redes o sistemas informáticos. En concreto se obtuvieron buenos resultados, con una precisión del 99% y una tasa de falsos positivos del 1%, lo que valida la eficiencia de la propuesta.

En la misma línea, en el artículo del congreso **ISKE 2013** (Guevara et al., 2013b), se presentó un sistema de detección de intrusos que utiliza una red neuronal para clasificar las tareas que ha realizado el usuario en la jornada laboral (día, hora y operación), en un periodo de tiempo de 2 años (base de datos Ecuador). Posteriormente, aplicando árboles de decisión y el teorema de Bayes, se detecta si es una intrusión o un comportamiento normal del usuario. Además se desarrolló una interfaz gráfica que muestra el comportamiento de cada uno de los usuarios como el resultado de la detección, con buenos resultados. El propósito de este trabajo era mostrar cómo se pueden utilizar diferentes técnicas inteligentes para identificar a los usuarios que acceden a la información y detectar intrusiones. Al aplicar técnicas avanzadas e inteligentes de análisis y de minería de datos se han modelado patrones de comportamiento de los usuarios. Estos perfiles de usuario son dinámicos.

El artículo publicado en el congreso **PIC 2014** (Guevara et al., 2014a), desarrolla un meta-clasificador con la combinación de cuatro técnicas de clasificación: ID3, Naive Bayes, UCS y C4.5, para clasificar ataques de red utilizando la base de datos KDD 99. Para validar la detección el resultado del meta-clasificador es finalmente interpretado por un sistema dinámico de reglas, lo que proporciona una identificación muy eficiente. La tasa de accesos correctamente clasificados como normales o ataques es del 99.99% con un porcentaje de falsos positivos muy bajo.

Otro artículo, publicado en el congreso **FLINS 2014** (Guevara et al., 2014c), crea un modelo basado en la sinergia de dos técnicas de clasificación, C4.5 y UCS para identificar accesos no permitidos a recursos, utilizando la base de datos Amazon (el desafío Kaggle). En esta propuesta se divide el conjunto de datos en tres partes e se ingresa al clasificador, que detecta si es un comportamiento normal o intrusivo. Esta estrategia ha resultado más eficaz que si toda la base de datos se utiliza como un conjunto único. La precisión del clasificador es del 98.4 % en el entrenamiento y del 98.1% en pruebas, lo que es un buen resultado para la detección.

En el congreso nacional **JNIC 2015** (Guevara et al., 2015b), se desarrolló un IDS aplicando selección negativa, con la base de datos Ecuador para el desarrollo de la propuesta, y una estructura de datos secuencial de cada sesión de usuario presentada en el artículo (Guevara et al., 2014c), que se ha extendido para su aplicación en este trabajo. Se trabaja con el comportamiento normal de cada uno de los usuarios para generar comportamientos anómalos y de esta manera identificar en las sesiones futuras intrusiones. Las contribuciones específicas son: modelo de datos, la identificación de un perfil de usuario y la aplicación de algoritmos de selección negativa y de búsqueda de secuencias de forma local, como Knuth Morris Pratt (KPM), para identificar tareas intrusivas de forma eficiente. La precisión de la propuesta es de 99.31% y la tasa de falsos positivos de 0.44%.

En el congreso **CAEPIA 2016** (Guevara et al., 2015b), se presenta un IDS que aplica el sistema inmune artificial humano para detectar comportamientos anómalos dentro de un sistema informático. Las principales aportaciones de este trabajo son: el proceso de desarrollo de un modelo de datos dinámico, clasificación de las tareas más ejecutadas y las poco ejecutadas por cada usuario, presentado en el artículo (Guevara et al., 2014c; Guevara et al., 2015a) y ampliado en este trabajo. Además, la aplicación de un algoritmo de sistemas inmunes artificiales, como la selección negativa, para la generación de detectores de secuencias binarias de tareas anómalas para la posterior detección de comportamientos peligrosos de los usuarios en el algoritmo de búsqueda local Knuth Morris Pratt (KMP). Este trabajo ha obtenido buenos resultados, con un 95.5% de precisión, pero con una media de tiempo de detección de 74.99 segundos, que pasa un sistema real es muy lento.

También en el congreso **CISIS 2016** (Guevara et al., 2016b) se aborda de nuevo el problema de la detección de intrusos o el acceso no autorizado a computadoras, que ha sido siempre crítico cuando se trata de sistemas de información, donde la seguridad, la integridad y la privacidad son cuestiones clave. Aunque se están desarrollando y aplicando estrategias cada vez más sofisticadas y eficaces de detección, tanto de hardware como de software, todavía existe la necesidad de mejorarlas para erradicar completamente el acceso ilegítimo. El propósito de este trabajo es mostrar cómo las técnicas informáticas se pueden utilizar para identificar el acceso no autorizado a las computadoras. Se aplica primero un análisis avanzado de los datos para obtener un enfoque cualitativo. Los árboles de decisión se utilizan para obtener patrones de comportamiento de los

usuarios, y las redes neuronales se usan como clasificadores para identificar los intrusos. El resultado obtenido es alentador cuando se ha aplicado la combinación de técnicas inteligentes sobre datos reales.

Finalmente, el artículo publicado en la revista **IEEE Transactions in Latinoamérica** (Guevara et al., 2016a), utiliza el algoritmo de Knuth Morris Pratt (alineamiento de secuencias) y selección negativa para generar subsecuencias anómalas basadas en el comportamiento normal de los usuarios y buscarlas en una nueva sesión de usuario. El trabajo se basa en información del comportamiento de usuarios autorizados que han realizado varias tareas en un sistema informático durante dos años. El estudio utiliza una estructura de datos dinámica que puede codificar las actividades actuales de los usuarios y sus comportamientos, presentado en los artículos (Guevara et al., 2015a; Guevara et al., 2015b), pero mejorado y extendido para su aplicación en este trabajo. La identificación de las tareas más y menos frecuentes, basadas en la base de datos histórica de cada usuario, proporciona una forma sencilla de crear un perfil de comportamiento. Con este perfil aplicamos técnicas de selección negativa para obtener un conjunto razonable de detectores anómalos. A continuación se aplica el algoritmo de Knuth-Morris-Pratt para localizar detectores de anomalías como indicadores de comportamiento fraudulento. Este procedimiento para detectar el comportamiento anómalo ha sido probado en datos reales y los resultados demuestran la efectividad de la propuesta y motivan investigaciones adicionales para mejorar el sistema de detección existente. Los resultados han obtenido una media de 98% de precisión y un 5% de falsos positivos.

## 4.3. Publicaciones sobre sistemas de detección de fugas de información

De las publicaciones de la investigación doctoral, en el artículo del congreso **RAID del 2015** (Guevara et al., 2014c), se plantea la búsqueda de subsecuencias anómalas para detección de fugas de información en sesiones de usuario en la base de datos Ecuador. En este artículo se utiliza una estructura de datos secuencial de las tareas realizadas por un usuario en una sesión. Se emplea el algoritmo de page Rank (Page, 1999) para agrupar las tareas ejecutadas frecuentemente por cada usuario, para posteriormente calcular su probabilidad aplicando el teorema de Bayes. Este algoritmo trabaja con datos históricos de las actividades de los usuarios autorizados en un sistema informático. Esta información reúne datos de la hora de los accesos, duración, día de la semana, operación, tabla a la que se ha accedido, etc. Los datos han sido proporcionados por una institución gubernamental del Ecuador. El procedimiento tiene dos fases: la primera se basa en el cálculo de la probabilidad de cada actividad que realiza un usuario. Estas actividades son, por ejemplo, modificar un archivo, eliminar, copiar, etc. Las diferentes actividades en diferentes momentos están codificadas. El algoritmo Page Rank se utiliza para calcular la probabilidad de cada actividad.

Pero las actividades forman secuencias, es decir, durante una sesión (tiempo entre el inicio sesión y desconexión), el usuario realiza diferentes actividades, una tras otra. Estas secuencias de actividades pueden tener diferentes longitudes. Se calcula la probabilidad de cada secuencia el teorema de Bayes y el mínimo de estas probabilidades condicionales se establece como umbral. La probabilidad de cada actividad se compara con el umbral. Si no lo supera, se genera un mensaje de advertencia

de posible fuga. De lo contrario, la secuencia de actividades pasa a la segunda fase del procedimiento. La secuencia de actividades que se está evaluando se compara con todas las secuencias de actividades de ese usuario almacenados en su histórico. Aplicando el algoritmo de Smith-Waterman, se obtiene una puntuación de similitud para esta subsecuencia con respecto al resto de las secuencias realizadas previamente por el usuario. Este algoritmo determina regiones similares entre dos cadenas comparando segmentos de todas las longitudes posibles y optimiza la medida de similitud. Si la puntuación es mayor que un segundo umbral, la actividad puede considerarse correcta; de lo contrario puede ser una fuga. Aunque este algoritmo todavía está en desarrollo, su principal contribución es la forma en que maneja diferentes longitudes de las secuencias y cómo funciona con un patrón de comportamiento de cada usuario.

Otro artículo de la tesis doctoral ha sido publicado en la revista **KNOSYS** (Guevara et al., 2017). Presenta un algoritmo para la detección de fugas de información utilizando las bases de datos Ecuador y Unix Comands. En este trabajo se propone un nuevo algoritmo para detectar el comportamiento anómalo de los usuarios en sesiones de trabajo en sistemas informáticos, con la estructura de datos desarrollada para los artículos (Guevara et al., 2014c; Guevara et al., 2015a; Guevara et al., 2015b; Guevara et al., 2016a). Primero se identifica el perfil de comportamiento de cada usuario autorizado y las tareas computacionales que habitualmente realiza en los archivos del sistema. A continuación, se codifica una nueva sesión como una secuencia de tareas (con subsecuencias de longitud 2) y se aplica un algoritmo basado en la probabilidad de dichas secuencias. Las actividades clasificadas como posibles anomalías se comprueban dos veces mediante la aplicación de cadenas de Markov. El procedimiento ha demostrado ser eficiente en términos de alta precisión de detección y baja tasa de falsos positivos. Se ha validado en una base de datos real proporcionada por una institución gubernamental de Ecuador y también en un conjunto de datos público de comandos Unix. Además, se ha demostrado que el algoritmo es eficiente con respecto al tiempo computacional y la sobrecarga de este software de monitoreo es baja. En concreto, la tasa de correctamente clasificados es del 94.78% y la tasa de falsos positivos es 4.73%.

# Capítulo V. Conclusiones y trabajos futuros

## 5.1. Conclusiones

La seguridad de la información se ha desarrollado y ha evolucionado considerablemente en las últimas décadas, debido a que es un área de con gran importancia para personas y organizaciones en todo el mundo. Esta área se centra en diseñar e implementar sistemas tecnológicos preventivos y reactivos que permiten resguardar y proteger la información de las organizaciones para mantener la confidencialidad, la disponibilidad e integridad de datos de la misma. Debido al elevado número de ciberataques a sistemas informáticos públicos y privados, se ha visto la necesidad de desarrollar nuevas estrategias, modelos, algoritmos y sistemas para analizar e identificar posibles actividades fraudulentas en sistemas de información.

En esta tesis se aportan distintas contribuciones en diferentes áreas.

Se ha implementado una nueva estructura de datos de secuencias de tareas (comandos y actividades), que permite la aplicación de técnicas de aprendizaje máquina y algoritmos de alineamiento de secuencias. Esta estructura fue aplicada a dos bases de datos distintas, una pública que es la de UNIX Comands y otra privada que pertenece a un sistema del sector de gobierno de la República del Ecuador.

- Se han implementado modelos de meta-clasificadores para la mejora de la eficiencia en la detección e identificación de intrusos con buenos resultados en término de porcentaje de aciertos y reducción de la tasa de falsos positivos. Se ha propuesto varios métodos para la detección de intrusos mediante la sinergia de técnicas de clasificación que provienen de la inteligencia artificial. Se ha evaluado la aplicación de varias técnicas de clasificación a un problema bien definido, y se ha probado como la combinación de varias de ellas mejora el porcentaje de aciertos. Se ha utilizado como base de conocimientos con diversas bases de datos.

  En concreto se han aplicado árboles de decisión C4.5 e ID3, redes neuronales, algoritmos genéticos UCS, máquinas de vectores de soporte SVM, Naive Bayes y vecinos más cercanos, para la clasificación de comportamientos normales y anormales de los sistemas informáticos estudiados.

- Además otra aportación es la aplicación de sistemas inmunes artificiales, selección negativa y Knuth Morris Pratt (alineamiento de secuencias de ADN), que basándose en el comportamiento esperado de los usuarios (base de datos del histórico de los usuarios), generan comportamientos anómalos y buscan la secuencia de ejecución de tareas realizadas por el usuario una sub-secuencia fraudulenta, con una detección precisa y eficiente, obteniendo resultados con una baja tasa de falsos positivos en la fase de pruebas. El estudio lo que permitió que el trabajo pueda describir de una forma aproximada la creación de perfiles de usuario representados por el modelo de datos. La

ventaja de nuestro modelo es la generación de comportamientos anómalos, aplicando selección negativa, a partir de comportamientos normales de cada uno de los usuarios. Esto permite que el perfil de cada usuario sea único y dinámico. El IDS se aplicó para comprobar el análisis de sensibilidad y definir el mejor rendimiento de parámetros de detección. Esta forma ofrece un mejor equilibrio entre la tasa de detección y tasa de falsas alarmas, también comprueba su adaptabilidad al comportamiento humano para la detección aplicando el enfoque propuesto.

- Por último, otra de las aportaciones de nuestro trabajo fue el desarrollo de un nuevo algoritmo de detección de fugas de información basado en la probabilidad de ejecución de tareas de los usuarios. El algoritmo está estructurado en múltiples pasos consecutivos, el último paso es comprobar la decisión de si es o no un intento de ataque. El algoritmo trabaja con la probabilidad de las secuencias de tareas que han sido ejecutadas por un usuario en una computadora. Se ha demostrado que es eficaz en términos de exactitud y con una tasa de falsos positivos muy baja, que es una de sus principales ventajas.

  Además se ha realizado un análisis del tiempo computacional que requiere en entrenamiento y aplicación del algoritmo, llegando a la conclusión de que puede utilizarse como monitor de software, ya que no sobrecarga el tiempo de trabajo. En otras palabras, cuando un usuario está ejecutando una secuencia de actividades, el sistema de detección supervisa su actividad al mismo tiempo pero sin ralentizar el ritmo de trabajo. Además, se puede implementar en cualquier sistema informático y es escalable.
  La principal limitación de las propuestas es que son algoritmos supervisados, que funcionan con datos históricos, y por lo tanto, necesitan información de los usuarios para generar su patrón de comportamiento (en nuestro caso los menos los datos de los últimos tres meses). De hecho, si la secuencia de tareas realizadas por un usuario en una sesión es demasiado corta, el sistema no podría detectar si es normal o no.
  Otra limitación de nuestro enfoque es que se centra en la detección y no en la prevención.

# 5.2. Trabajos futuros

A partir de las contribuciones conseguidas en este trabajo de investigación, planteamos algunas líneas de trabajo futuro.

- En el algoritmo desarrollado con cadenas de Markov se puede mejorar la identificación de sub-secuencias anómalas aplicando Deep Learning para mejorar la precisión del clasificador y también el tiempo de detección.

- Para los sistema de detección de anomalías y de detección de intrusos proponemos, en primer lugar, la aplicación de algoritmos de procesamiento de Big Data (Ramírez, 2017). También se podría unificar toda la información de distintos sistemas informáticos, redes sociales, dispositivos móviles, etc., para definir un perfil más real de cada uno de los

usuarios, y de esta manera que sea más eficiente la detección de comportamientos anómalos.

- Finalmente, otro avance que planteamos es la implementación de sistemas biométricos basados en ritmo cardiaco, movimiento de pupilas, ritmo de escritura, entre otras, para que el perfil del usuario sea único y así más eficiente para detectar un posible ciberataque.

# Part II: Thesis in English

# Chapter 1: Introduction

Over the last few decades, information networks have grown exponentially due to the great number of computers, servers, and communication equipment connected to the Internet. Thus, these systems have become an essential part of people's lives. The Internet has become a tool which has revolutionized society and, in particular, information technology and telecommunications.

Currently, most people and institutions make use of networks to carry out operations and dealings concerning finance, tax, advertising, corporations, health, etc.

Due to this open access, the data handled should be protected so they are not misused. Information is one of the most important assets for companies and businesses. In fact, cyberattacks are a major problem which affects most people and businesses around the world. This problem can be described as the transfer of private data in a prohibited manner from an organization to an external agent (an unauthorized person or organization). It is currently a real concern and a threat to security. Confidential data belonging to people and institutions include intellectual property, financial information, personal information, and a wide variety of information regarding the business and industry involved (Kumar, 2013). A study carried out by Insight Express in the United States, commissioned by Cisco Systems in 2008, revealed that employee behavior is one of the most significant threats of loss of sensitive information. Recent studies report that internal threats cause greater financial loss than attacks made from outside the organization (Takebayashi and Tomoyoshi, 2010).

Though there are many examples, the most well-known case of leakage of confidential information was the disclosure of internal documents belonging to the National Security Agency (NSA) of the United States via "WikiLeaks". The problem consisted in the disclosure of 250,000 diplomatic cables and 400,000 military reports. This incident received a lot of attention and the computer security of the United States was criticized by many other countries and civil rights organizations. In fact, these crimes can cause great financial losses or seriously damage the reputation of an organization or government (Karhula and Paivikki, 2011).

Due to all of the aforementioned, it is necessary to implement new security policies that include the prevention and detection of intrusions and the leakage of sensitive information in institutions. Intentional information leakages compromise the security of information. Furthermore, the main actors in the leakage of data in institutions are often users who are authorized to access internal systems.

This dissertation will present the development of algorithms based on artificial intelligence for the detection of intrusions in computer systems and the detection of anomalies in user behavior to identify information leakages. The method is based on the operations, activities, and tasks carried out by the user within the information systems.

# 1.1. Purpose

Currently, the security of IT networks and systems is of great importance given the value of information in public or private institutions. Furthermore, the phenomenal amount of data handled by people and the computers they use is a major problem given the possibility of accidental or intentional leakages, which may cause massive losses for people or institutions.

The public sector of the Republic of Ecuador has a large network of information systems which allow to automatically carry out internal activities and tasks. However, safeguarding the integrity and confidentiality of this information is increasingly more difficult. This network suffers constant and varied types of cyberattacks aimed at stealing or modifying the classified information of government institutions. Important documents are shared across different institutions through this large information network. This leads to user impersonation and makes the network vulnerable to attacks from unauthorized parties. Hence the need to implement an efficient system in order to properly control access and prevent these cyberattacks.

The efforts made until now to prevent these attacks have resulted in complex and inefficient security systems, given that most impersonations and intrusions are made within government entities, which complicate the identification of authorized users. This is the main reason why a study of user behavior has been proposed, by applying intelligent techniques to pattern recognition, in order to efficiently and automatically identify fraudulent access. In other words, models have been developed for the detection of anomalies, intrusions, and information leakages by using the unique behavior of each user in the information systems.

# 1.2. Objectives

## *1.3.1. General Objective*

The objective of this dissertation is to design, develop, and implement a set of efficient algorithms by applying artificial intelligence techniques for the detection of anomalies and information leakages, and the identification of fraudulent access in information systems.

## *1.2.2. Specific Objectives*

- Analyze sets of real data to identify computer attacks both on networks and systems.
- Develop a dynamic sequential data structure to describe the behavioral pattern of users.
- Apply artificial intelligence techniques jointly to identify intrusions in computer equipment.
- Develop algorithms based on statistical techniques to calculate the probability of an attack according to user behavior.
- Develop algorithms based on bio-inspired techniques to detect information leakages.

# 1.3. Dissertation structure

This dissertation is organized in the following manner: after this introductory chapter, chapter 2 presents the state-of-art in the field of computer security, with a brief history, definition, the

elements of which it consists, and the existing threats. It also includes the classification of computer attacks and the different aspects with which they interact, weaknesses of management systems, and design defects.

Chapter 3 describes, regarding publications resulting from this dissertation, the databases used, anomaly, intrusion and leakage of information detection systems, as well as the strategy on which each of them is based. It also contains a brief discussion on artificial intelligence techniques and algorithms used in the dissertation for the detection of computer attacks, such as artificial neural networks, decision trees, genetic algorithms, etc.

Chapter 4 features the papers and contributions published in national and international congresses and indexed journals. These papers were developed with different databases and algorithms to generate comparative results in the field of anomaly, intrusion, and leakage of information detection. These publications and their quality scores confirm the interest of the submitted contributions as well as the good results obtained.

Chapter 5 presents the main conclusions drawn from the research findings and future lines works in this area.

# Chapter 2: Computer security

Computer security is a field specifically focused on the protection of computer infrastructure (communication equipment, data storage, servers, computers, mobile devices, etc.), safeguarding the information which is contained or shared on electronic media. In order to achieve this objective, laws, regulations, standards, protocols, equipment, tools, applications, etc., have been created, with the aim of minimizing the possible risk of information ending up in the hands of unauthorized individuals.

Therefore, computer security is the discipline of designing rules, procedures, methods, techniques, tools or applications to achieve confidentiality, integrity, and availability in an information system (Stavroulakis and Stamp, 2010).

Computer security is based on three key pillars: confidentiality, that is, that information is accessible only to authorized individuals; data integrity, meaning it remains unchanged in case of malicious events, accidents or disasters; and availability, allowing the use of information on a computer system at all times by authorized users, with no degrading regarding access. Finally, authenticity is the element that allows to identify the information generator (Russell and Gangemi, 1991).

## 2.1. Computer security elements

Computer security was developed to protect the main computer elements, which are:

- Computer infrastructure: it entails seeking optimal functioning and administration of technological equipment to prevent any harmful event (fraud, malfunctions, natural disasters, theft, etc.), that impedes the normal performance of the computer equipment.
- Users: people who use the computer infrastructure and manage the information to be protected. The main task is to guarantee that the handling of information by the users is appropriate and secure.
- Information: it is the main asset of a person or institution. Information is part of the computer infrastructure and is handled by the users.

Attacks on the security of an information system or network consist in a flow of information from a source (hard drive, file, database, etc.) to a destination, such as another file or user. There are four types of possible attacks on information, which are:

- Interruption: the destruction or deactivation of a resource in an information system or network, which ceases to be available to users. This is an attack on availability.
- Interception: an unauthorized actor (person, program or equipment) that manages to access a resource to carry out fraudulent activities (wire tapping to acquire data, illegal copying of files or programs, etc.). This is an attack on confidentiality.

- Modification: the impairment of the resource (modifying information, altering a program, editing the content of messages, etc.) that is unlawfully re-edited. This is an attack on integrity.
- Fabrication: the creation of false objects (files, programs, data) by an unauthorized actor that inserts them in information equipment or systems. This is an attack on authenticity.

# 2.1.1. Computer attacks

A computer attack is a method through which an actor (unauthorized person, equipment or program) attempts to take control of, disrupt or damage another computer system (computer, private network, etc.) by using computer equipment or systems. (Burke and Christiansen, 2009). Computer attacks may be categorized into two groups: passive and active.

c) Passive attacks carry out observations of the transmissions made in an information system or network. The objective is to obtain information which is being transmitted in order to disclose it and use it for fraudulent activities. In order to carry out such an attack, the unauthorized actor analyses the traffic of the network in order to extract confidential information. The fraudulent actor may determine the location and identity of the computers that are communicating and observe the frequency and the length of the exchanged messages. This information can be useful to identify the nature of the communication that is being made. Passive attacks are very difficult to detect, given that they do not alter the data. However, it is possible to impede their success. Therefore, in dealing with these attacks, the focus is on prevention rather than on detection.
d) Active attacks modify the flow of data or create false flows. They are categorized in four groups: masking, repetition, modification of messages, and denial of a service.

Both active and passive attacks require a technical analysis for the optimum prevention of any type of attempt of unauthorized action in the information system or network.

## 2.2. Security systems

The Internet and computer systems have raised many security problems due to the continuous use of networks. Studies show that the number of intrusions has risen exponentially every year. All intrusions or malicious attacks to networks, servers, computers or information systems may cause massive losses or serious disasters to people or institutions, as well as breach information security policies. Cybersecurity is an important area of research to improve detection and prevent all types of threats to the networks and systems where information is stored (Kleinberg et al., 2001).

In order to better identify the different aspects of the problem of intrusion in computing, detection and prevention systems are used (Deepa and Kavitha, 2012).

An intrusion is described as an attempt to compromise information, and avoid or violate the security mechanisms of a network or computer. The detection of intrusions is the process of monitoring the events that occur in a computer system or network, analyzing them in order to be able to identify possible signs of attacks (Kantzavelou and Katsikas, 2010). An intrusion detection system (IDS) is software or hardware that automates the intrusion detection process and identifies unauthorized access (Kabiri and Ghorbani, 2005; Chung and Wahid, 2012).

Within the area of detection there is a more specific context, the detection of information leakages, which is subsequent to the detection of intrusions. It occurs with authorized and legitimate users who go undetected by the IDS, but carry out fraudulent activities to leak sensitive information.

Furthermore, the prevention of intrusions has the characteristics and capacities of IDS, but its main aim is attempting to prevent possible incidents by limiting the use of their resources, i.e, it changes the content of the attack and/or the security environment and prevents the attack from being successful. It can also change the settings of other security controls to interrupt attacks, such as resetting a network device to block the attacker's or victim's access, or modify a host-based firewall with the aim of blocking inbound attacks. However, this method produces a high number of false alarms and can incorrectly classify a normal activity (non-intrusive) as malicious and respond to this activity incorrectly.

### 2.2.1. Intrusion detection systems

An intrusion detection system (IDS) is a program that detects unauthorized access on a computer or network. IDS normally have sensors (network sniffers) within the IDS nucleus, which monitor activities (generally regarding network traffic, user behavior, etc.). IDS detect anomalies that may be a sign of existing attacks (Lazarevic et al., 2005).

There is a variety of aspects and perspectives for categorizing the techniques, methods and ways of addressing the issue of IDS. That is why figure 1 presents four general aspects for their categorization: System Deployment, Data Source, Timelines, and Detection Strategy.

The branch of System Deployment carries out a varied analysis into the host architecture and network to be protected. This branch analyses three types of study, which are: Technology Type,

Networking Type, and Network Architecture. In our research we have addressed the group Technology Type to analyze the network traffic. We have focused on the type of technology, classified into four categories depending on the place where the suspecting activities were detected and the type of events that can be recognized. The four types are: Host-based IDS (HIDS), network-based IDS (NIDS), wireless-based IDS (WIDS), Network Behavior Analysis (NBA) and Multics IDS (MIDS) (Mukherjee et al., 1994; Stavroulakis and Stamp, 2010; Sabahi and Movaghar, 2008; Modi et al., 2012). HIDS monitor and collect the characteristics and activities of host computers that contain sensitive information. These computers are analyzed to identify any suspicious activity. NIDS capture the network traffic in specific segments via sensors, in order to later analyze the activities of the applications, such as their used protocols. This way, possible attacks can be recognized. WIDS are similar to NIDS, but only monitor wireless network traffic, such as ad hoc networks, wireless sensor networks and wireless mesh networks. NBA inspect the traffic of the network to recognize flows of unexpected attacks compared to expected normal behavior. Furthermore, many of these IDS can be applied together to improve the detection rate and carry out a more complete, precise and efficient identification of possible attacks. This type of technology is implemented by MIDS.

Data Source studies the sources of data collected from the host and the network to be protected. This type of IDS features three types of analyses, which are: Data Type, Data Collection, and Collection Component.

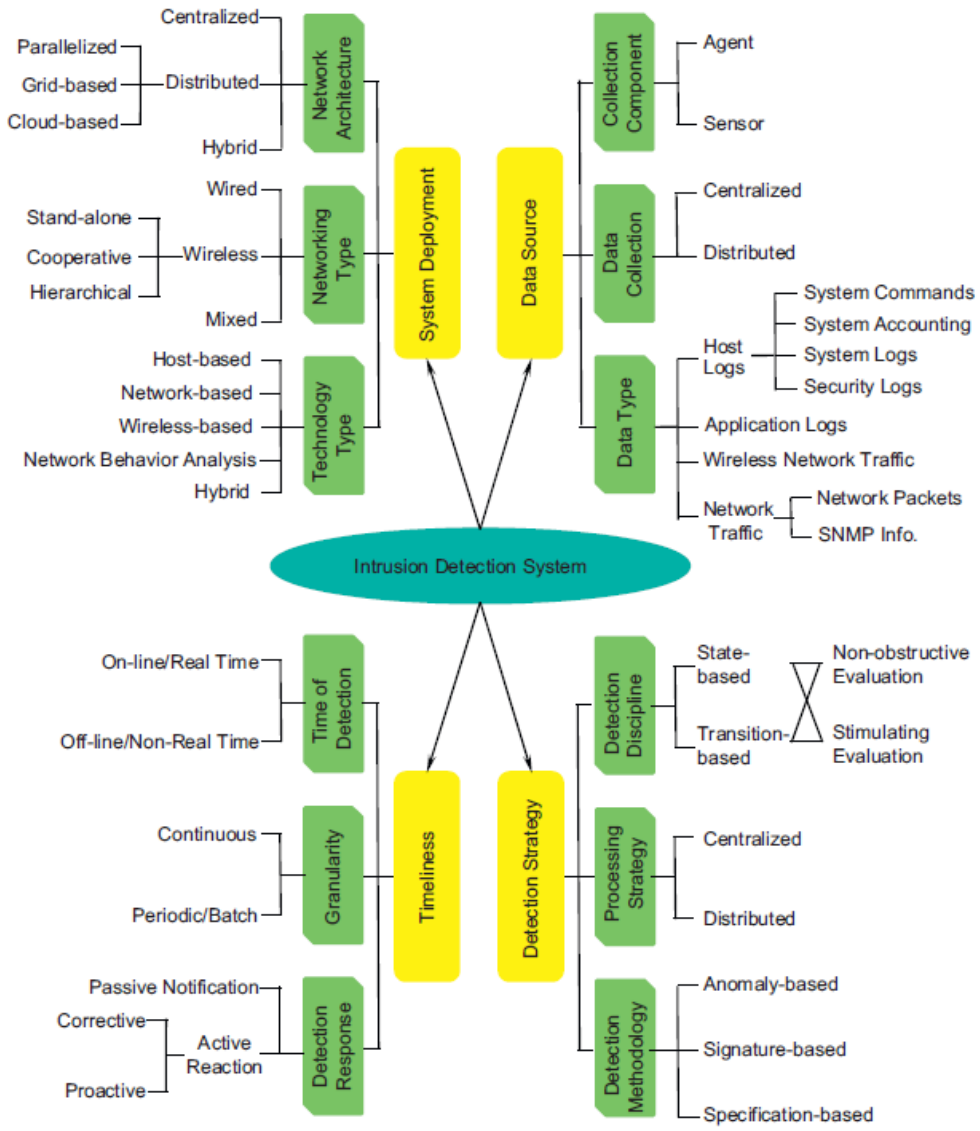



Figure 1. An overview of the types of IDS (Liao et al., 2013).

In the field of Detection Strategy, the detection methodologies applied to intrusion detection systems are widely varied. They include Signature-based Detection (SD), Anomaly-based detection (AD) (Alomari and Othman, 2012), Stateful Protocol Analysis (SPA), etc. (Debar et al., 2000). This paper focuses on Anomaly-based Detection.

An anomaly is a deviation from a known (normal) behavior, based on a profile or pattern. This behavior comes from monitoring regular activities, network connections, hosts or users over a period of time. Profiles can be either static or dynamic, and developed for many attributes, e.g., failed login attempts, processor usage, the number of e-mails sent, etc. Subsequently, normal profiles are compared to observed events in order to recognize significant attacks. This approach is also called behavior-based detection in some papers.

As previously mentioned, two points of view may be developed in the area of intrusion detection: detection of anomalies and detection of incorrect use (which includes information leakages). An example of incorrect use detection is research which concentrates on Role-based

Access Control (RBAC). This detection is used in databases which have been distributed but are still susceptible to unauthorized access attempts. In RBAC, users potentially pose the risk of unauthorized access attempts by using false credentials or an intentional or unintentional incorrect use of the permissions authorized via a user role. This method features computer security components such as verification of credentials, analysis of queries, monitoring of history records, and the expected use for a probabilistic risk assessment. Incidentally, an example of anomaly detection is the method of deep network packet inspection, which is also widely used in the intrusion detection system (Roesch, 1999). This paper provides an in-depth description of the design and implementation of a string matching algorithm to control short and flexible patterns in network traffic (Hao, 2005). The major drawback of this algorithm is that it is not designed for different types of data, which makes it difficult to distinguish between sensitive and normal information. This could cause detection problems.

In (Stavroulakis and Stamp, 2010) paper proposed classifying IDS according to methods of statistic, pattern, and heuristic detection.

The methods which were studied are as follows:

4. Statistics-based methods
    c) Statistical methods (Debar et al., 2000; Patcha et al., 2007; García-Teodoro et al., 2009; Xie, 2011).
    d) Bayesian methods (Kabiri and Ghorbani, 2005; Murali and Rao, 2005; Sabahi and Movaghar, 2008; Lazarevic et al., 2005.).
5. Pattern-based methods
    d) Pattern recognition (Tsai et al., 2009; Ye et al., 2001).
    e) Support vector machines (Modi et al., 2012; Kolias et al., 2011; Li et al., 2012; Horng et al., 2011).
    f) Markov chains (Couture, 2012; Gao et al., 2002).
6. Heuristic-based methods
    d) Neural networks (Murali and Rao, 2005; Mar, 2012; Modi et al., 2012; Kenkre et al., 2015).
    e) Genetic algorithms (Sen and Clark, 2011; Gong et al., 2005).
    f) Immune systems (Debar et al., 2000; Stavroulakis and Stamp, 2010).

Finally, the Timelines branch analyses the time of detection, which may be offline or in real time, i.e., online. This type of IDS analyzes the transmission of data on a rolling basis, that is, "continuous transformation", looking for signs of attacks over time (granularity).

# 2.2.2. Data leakage detection systems

Data leakage detection and alert by authorized users is one of the most difficult problems that cybersecurity researchers and professionals face. Some years ago, security experts used to identify possible internal threats by examining suspicious activities in the physical behavior of users. Although this is still relevant nowadays, the current trend consists in detecting suspicious activities through the operations executed in computer systems. Despite this, confidential information is still being leaked at an alarming rate.

In the nineties, organizations such as the US Defense Personnel and Security Research Center (PERSEREC) and the US Research and Development (RAND) Corporation focus on internal threats with publications and workshops (Herbig and Wiskoff, 2002; Anderson, 1999; Anderson, 2000; Anderson and Brackney, 2004). As of 2001, the CERT Insider Threat Center also started to carry out research into this topic and began to develop a database of threat cases concerning insider information leakage. During this time, researchers were also developing models to identify the different characteristics of internal users (the main victims of the organization) and the incident itself. The objective of this research was focusing on aspects of the criminal's behavior when they extracted confidential information (Shaw et al., 1998), psychological characteristics such as computer addiction and lack of ethics and empathy as potential indicators of destructive and potentially illegal behavior. The most specific approach of the socio-technical field analyzes characteristics, personality features, and technical behavior (including usage patterns and execution of significant mistakes) to identify data leakages. These indicators are a weighted measure that makes the detection task easier (Schultz, 2002).

Afterwards, studies were carried out that show an approach focused on the development of algorithms and information access procedures that allow authorized users to have exclusive access to sensitive data through control policies (Bonatti et al., 2002; Jajodia et al., 2001). These studies are aimed at preventing information from being leaked by limiting the use of data to trusted users or systems. However, these policies are restrictive and may make it impossible to fulfill user requests.

Previous studies consider a method to reduce data leakage through the detection of an improper use of stored information. This information is located in databases where syntax of executed SQL statements is analyzed by focusing on existent data. Syntax is based on the representation of an SQL expression to build user profiles (Kamra et al., 2008).

Data leakage detection uses host-based and network-based solutions. Host-based solutions analyze the activities that the attacker may carry out in the victim's system when they try to steal information. Afterwards, these solutions prevent the loss of information by blocking hardware and software resources. An example of this kind of system is the project presented by (Jiang et al., 2010), the same which applies solutions to detect stealthy malware by scanning the operative system with an antivirus. Another project that utilizes this kind of leakage detection is that by (Borders and Prakash, 2009), that considers including data encryption when information is used and applying policies to restrict the transfer of sensitive data. These methods can be complemented and implemented simultaneously to improve leakage detection. One of the most well-known projects on data leakage is that presented by (Papadimitriou and Garcia-Molina, 2011). He studied the problem of a distributor that provided sensitive information to different allegedly trusted (third) parties. Some of these data were leaked and found in unauthorized places (Internet, PC, storing device, etc.). The distributor had to assess the most likely origin of the leaked data and the actors responsible. This project suggested a data allocation strategy (through actors) that improves the likelihood of identifying leakages. In some cases, data may be added to registers, false information to improve leakage detection possibilities and the identification of the actor responsible.

The network-based solution detects leakages in communications among interconnected systems. Its interface works as an activity monitor that captures all the network traffic and analyzes malicious access to sensitive information. An example of this kind of leakage detection is Stann's

paper presented in 2003, where he analyzed the network using sensors in wireless networks and explored the reliability of the transport layer, which guarantees information delivery to its addressee in the network. Other papers in this area focus on analyzing unencrypted outgoing network traffic through an in-depth packet inspection (Shu and Yao, 2013) or a theoretical analysis of information using entropy (Fawcett, 2010). However, these methods require confidential data to be stored in the detection system. Currently, there are studies about accidental leakage of confidential data in network traffic by an authorized user (Jung et al., 2008). The problem is detecting these kind of leakages, that may be caused by human errors, such as forgetting to use encryption, careless forwarding an internal e-mail or attached files to external parties, etc. The approach is based on periodically monitoring and analyzing a specific part of the network traffic, which identifies repeated information by generating rules of behavior. The paper suggests an optimal and efficient solution that enables to locate a substantial increase in the quantity of new information. This method of data leakage detection identifies scenario variations within the normal data flow (Borders, 2006).

More recent studies address the problem of information leakage in a variety of different ways. Many of these studies propose sophisticated solutions using Big Data methods (Tene and Polonetsky, 2012), artificial intelligence algorithms or statistical or graphical analysis. Other trends try to distinguish and reproduce the user's behavior through linguistic semantics, the reason for the communication, such as e-mail or instant messaging.

# 2.3. Artificial intelligence techniques applied to computer security

This section features some of the techniques used in this research, particularly, artificial neural networks, decision trees, and genetic algorithms, among others.

## *2.3.1. Artificial neural networks*

Artificial neural networks are a mathematical method that individually or jointly solves problems, by carrying out data classification, forecasting, identification, optimization and diagnosis tasks. The elementary unit of an artificial neural network is a local processor defined as a neuron capable of calculating an amount of input data in order to apply an activation function and obtain a signal that will be transmitted to the next neuron. Neurons are grouped in layers (levels), which are highly connected and present weights with weighted values. Artificial neural networks use a supervised or unsupervised learning algorithm, according to the type of network, to adjust their architecture, thus minimizing an error function that shows the level of data adjustment generalization.

The main characteristic of neural networks is that they are capable of learning from a group of training data. This means that they can find a model that adjusts to the learned data.

Supervised learning consists in training the neural network by using a group of training data or patterns made of input and output data, and applying an algorithm that adjusts network weights, thus creating an output.

In our research, we have applied neural networks to detect intrusions, as shown in papers (Guevara et al., 2013a; Guevara et al., 2013b), where a metaclassifier made of neural networks,

support vector machines, and decision trees has been designed to identify intrusive behavior. Good results have been obtained, with a highly reliable detection of intrusions in computer networks or systems (Figure 2).

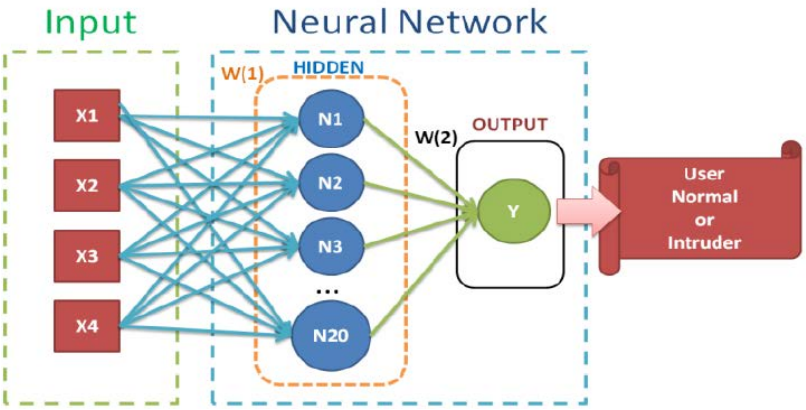



Figure 2. Neural network structure for the intrusion detection(Guevara, 2016b).

## *2.3.2. Decision trees*

A decision tree is a model that can make forecasts and classifications through inductive learning with structured data and logical constructions. Decision trees are a very popular classification model. In order to build a decision tree using information, we must describe the elements involved: nodes, vectors, arcs and labels. Nodes can be defined as the moment when a decision must be made among many other possible decisions. Therefore, the higher the number of nodes, the higher number of possible endings we can achieve. Thus, it is difficult to analyze a tree with many nodes, since there are many paths that can be followed. Vectors are the final solution we reach depending on the different possibilities available. Arcs are the links between nodes, and represent each different action. Labels are found in every node and arc and name every action.

Decision trees present two of the most used algorithms in classification, i.e, C4.5 and ID3. C4.5 decision trees use similarity-based learning. These algorithms recursively create a decision tree by taking into account the criterion of the largest information gain, that is, they choose the attribute that best classifies data. ID3 decision trees use the search for hypothesis or rules relating to a given group of examples. The group of examples must be made up of different value tuples. Each of them, an attribute. The attribute to be classified in one of them is a binary objective: positive or negative, yes or no, valid or invalid, etc. Thus, the algorithm tries to obtain hypotheses that determine, in the event of new situations, if the said example will be positive or negative.

We have used decision trees in different papers in our research (Guevara et al., 2013a; Guevara et al., 2013b; Guevara et al., 2014a; Guevara et al., 2014c; Guevara et al., 2017) where multiple intrusion detection models have been developed. Metaclassifiers (Guevara et al., 2013a; Guevara et al., 2014a; Guevara et al., 2014c) have been designed, where the combination of different classification techniques improves detection results and reduces the false positive rate. Other papers, such as (Guevara et al., 2013b; Guevara et al., 2017), feature the application of decision trees as a detection verification technique. Satisfactory results have been obtained in terms of accuracy (Figure 3).

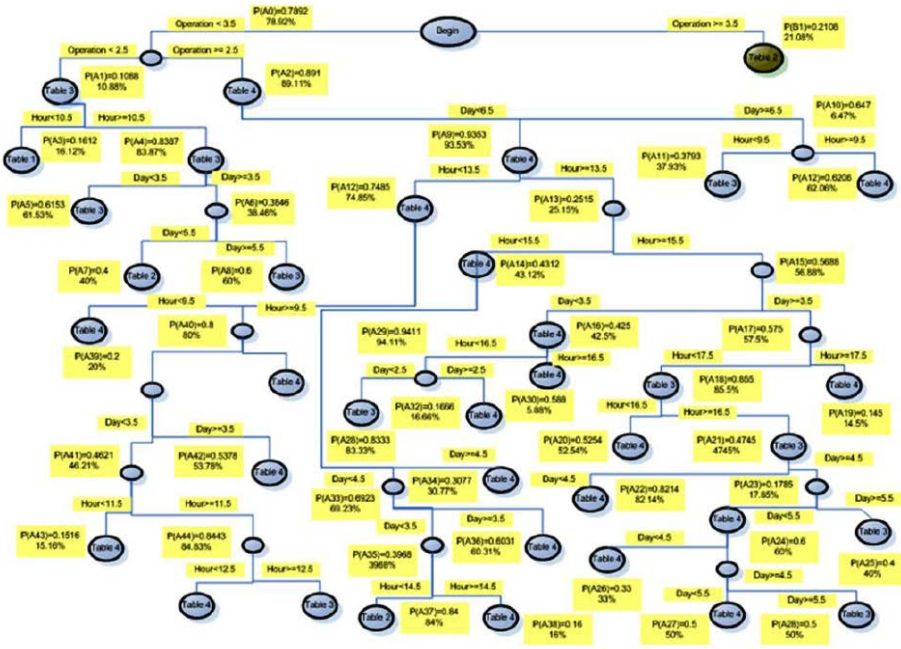



Figure 3. Decision Tree for Intrusion Detection (Guevara, 2016b).

## *2.3.3. Genetic algorithms*

Genetic algorithms are adaptive methods that can be used to solve search and optimization issues. They are based on the genetic process of living organisms. Throughout generations, populations evolve in nature according to the principles of natural selection and survival of the fittest. Taking this idea as a base, genetic algorithms are capable of creating solutions for real-life problems. The evolution of the said solutions leads to these issues showing optimal results, which largely depend on the appropriate configuration of the algorithm. Goldberg and Holland established the basic principles of genetic algorithms in 1988. It is necessary to encrypt and represent solutions. Furthermore, a problem adjustment or adaptation function is required. This function assigns an evaluation to each possible encrypted solution. During the execution of the algorithm, the parents are selected for reproduction. Afterwards, the selected individuals will be crossed and create children, over which a mutation operator may act. The result of the combination of the previous functions will be a group of individuals (possible solutions to the problem), which will be part of the next population in the evolution of the genetic algorithm.

In our research we use genetic algorithms, specifically UCS (supervised classifier systems). In paper (Guevara et al., 2014c), we combined combine the C4.5 and UCS decision tree techniques, which are subsequently applied to an Amazon database to identify computer access by users. Good results were obtained.

# Chapter III: Description of experiment databases

We have used three databases in our research, the main basis upon which we have verified our proposals and their efficiency. These databases are public and private. Therefore, we can check the efficiency of the models in each publication. Public databases are: UNIX Commands, KDD 99 and Amazon. The private database belongs to the Ecuadorian government. Find below a detailed description of each of the said databases.

## 3.1 UNIX Command data Set

We have used the UNIX database (Aeberhard et al., 1994) for experiments in our study. The data set features 9 sets of user data obtained from the history of commands executed by 8 users with UNIX computers over a 2-year period (UNIX User Data Set). The information included in this data set consists in command names, flags, and shell metacharacters. Furthermore, there are labels, such as ** SOF ** and ** EOF **, that have been inserted at the beginning and at the end of user sessions to identify their dimension. Sessions are organized by date and symbols appear in the order they were executed in the shell session, but this information does not include timestamps.

In order to achieve an optimal result in our experiments, we have used two different types of data set: one to train the proposed model and another to carry out tests. We may also work with a third data set to validate each of our proposals. Data sets are detailed as follows:

**Training data:** This data set is formed by 60% of the sessions of each user included in the data set. These data are used by the expert to train the algorithm.

**Testing data:** This data set is used to assess performance (detection) of the anomaly system that was previously trained with data of each of the users.

**Validation data:** The Validation data set was created using the testing information of a given user and of a different user. This information allows tests to show the optimal performance of the proposed algorithm.

Table 1. Session distribution per user in the UNIX database.

| USERS | NORMAL SESSIONS | COMMANDS | Maximum Session Length |
|---|---|---|---|
| USR_0 | 567 | 7840 | 214 |
| USR_1 | 515 | 18851 | 429 |
| USR_2 | 1069 | 16600 | 353 |

| | | | |
|---|---|---|---|
| USR_3 | 501 | 15864 | 1865 |
| USR_4 | 955 | 35905 | 957 |
| USR_5 | 582 | 33657 | 1400 |
| USR_6 | 3419 | 57314 | 434 |
| USR_7 | 1522 | 14285 | 446 |
| USR_8 | 1983 | 50076 | 564 |
| **TOTAL** | 11113 | 250392 | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| ** SOF ** | f | <1> | f | <1> | f | <1> | ** EOF ** |
| ** SOF ** | elm | telnet | <1> | f | <1> | ** EOF ** | |
| ** SOF ** | cd | <1> | cd | <1> | ls | cd | ** EOF ** |
| ** SOF ** | telnet | <1> | lo | ** EOF ** | | | |
| ** SOF ** | f | <1> | date | telnet | <1> | f | ** EOF ** |
| ** SOF ** | f | <1> | telnet | <1> | elm | ** EOF ** | |
| ** SOF ** | elm | vt100 | elm | telnet | <1> | telnet | ** EOF ** |
| ** SOF ** | f | <1> | date | elm | ls | ** EOF ** | |
| ** SOF ** | elm | f | <1> | date | elm | fg | ** EOF ** |
| ** SOF ** | elm | f | <1> | date | telnet | <1> | ** EOF ** |
| ** SOF ** | lo | ** EOF ** | | | | | |
| ** SOF ** | elm | lo | ** EOF ** | | | | |

Figure 4. Example of a sequential data structure of the UNIX database.

Table 2. Data table of User# 1, with 5 basic commands and 11 additional commands.

| | Basic Commands | | | | | Additional Commands | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Session | finger | man | who | date | more | cd | ls | elm | rm | grep | make | mv | cp | less | cat | chmod | label |
| 1 | 2 | 0 | 0 | 0 | 0 | 3 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | normal |
| 2 | 0 | 0 | 0 | 4 | 0 | 0 | 5 | 1 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | normal |
| 3 | 1 | 1 | 0 | 0 | 3 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | normal |
| 4 | 0 | 2 | 0 | 0 | 0 | 1 | 2 | 1 | 0 | 0 | 0 | 6 | 0 | 0 | 0 | 0 | normal |
| 5 | 0 | 0 | 2 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 7 | 0 | 0 | 0 | 0 | abnormal |
| 6 | 0 | 0 | 3 | 0 | 6 | 0 | 0 | 0 | 7 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | abnormal |
| 7 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 3 | 0 | 0 | 0 | 0 | 0 | 5 | 0 | 0 | abnormal |
| 8 | 0 | 3 | 0 | 0 | 1 | 5 | 4 | 0 | 0 | 0 | 0 | 4 | 0 | 0 | 0 | 0 | abnormal |
| 9 | 1 | 4 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 3 | 0 | 0 | 1 | 0 | abnormal |

# 3.2 KDD Cup 1999 data Set

This is the data set used for The Third International Knowledge Discovery and Data Mining Tools Competition, which was held in conjunction with KDD-99 The Fifth International Conference on Knowledge Discovery and Data Mining (KDD Cup 1999 Data Set). The competition's task was to build a network intrusion detector, a predictive model that was able to distinguish between fraudulent connections, called intrusions, and normal connections. The 1998 DARPA Intrusion Detection Evaluation Program was prepared and managed by MIT Lincoln Labs. The objective was to study and evaluate the research into intrusion detection. A standard data set was submitted for auditing purposes. It included a wide range of simulated intrusions in a military network environment. The KDD 1999 intrusion detection competition used a version of this data set. Lincoln Labs created an environment to generate nine weeks of TCP dump data for a local area network (LAN) that simulated a typical LAN of the U.S. Air Force. They operated the LAN as if it were a real environment of the U.S. Air Force, though simulating multiple attacks. Gross training data were approximately four gigabytes of TCP dump compressed binary data obtained from 7 weeks of network traffic. This was processed in approximately five million logins. Likewise, the 2-week testing data resulted in approximately two million logins. A login is a TCP packet sequence that begins and ends in specific periods of time, during which data flow to and from a source IP address to a destination IP address according to a clearly defined protocol. Each login is labeled as normal or as an attack with exactly one type of specific attack. Each login comprises approximately 100 bytes. Attacks are classified into four main categories:

- DoS: Denial of Service.
- R2L: unauthorized access from a remote machine.
- U2R: unauthorized access to root privileges.
- Probing: surveillance and other probings, for example, port exploration.

It must be noted that testing data do not arise from the same probability distribution as training data, and that they include specific types of attacks that are not included in training data. This makes the task more realistic. The characteristics of each of the attributes of the data set are detailed in a complete list which is illustrated in the three following tables:

Table 3. Basic characteristics of individual TCP connections.

| ***Name of the attribute*** | ***Description*** | ***Type*** |
|---|---|---|
| duration | Duration (number of seconds) of the connection | continuous |
| protocol_type | Type of protocol, eg, tcp, udp, etc. | discrete |
| Service | Destination network service, eg, http, telnet, etc. | discrete |
| src_bytes | Number of origin to destination data bytes | continuous |
| dst_bytes | Number of destination to source data bytes | continuous |

| | | |
|---|---|---|
| Flag | Normal or error connection status | discrete |
| land | 1 if the connection is to the same host/ 0 if otherwise | discrete |
| wrong_fragment | Number of wrong fragments | continuous |
| urgent | Number of urgent packets | continuous |

Table 4. Characteristics of the content within a suggested connection.

| **Name of the attribute** | **Description** | **Type** |
|---|---|---|
| hot | Number of "hot" indicators | continuous |
| num_failed_logins | Number of failed login attempts | continuous |
| logged_in | 1 if logged in correctly, 0 if otherwise. | discrete |
| num_compromised | Number of "compromised" conditions | continuous |
| root_shell | 1 if root shell is obtained, 0 if otherwise | discrete |
| su_attempted | 1 if the command "su root" has been attempted, 0 if otherwise | discrete |
| num_root | Number of "root" accesses | continuous |
| num_file_creations | Number of file creation operations | continuous |
| num_shells | Number of shell requests | continuous |
| num_access_files | Number of operations in access control files | continuous |
| num_outbound_cmds | Number of outbound commands in one ftp session | continuous |
| is_hot_login | 1 if the login session belongs to the "hot" list, 0 if otherwise | discrete |
| is_guest_login | 1 if it is a "guest" login session, 0 if otherwise | discrete |

Table 5. Traffic characteristics using a time window of two seconds.

| **Name of the attribute** | **Description** | **Type** |
|---|---|---|

| | | |
|---|---|---|
| count | Number of connections to the same host as the current connection over the last two seconds | continuous |
| Note: The following characteristics refer to the same host connections. | | |
| serror_rate | % of connections with "SYN" errors | continuous |
| rerror_rate | % of connections with "REJ" errors | continuous |
| same_srv_rate | % of connections with the same service | continuous |
| diff_srv_rate | % of connections with a different service | continuous |
| srv_count | Number of connections to the same service as the current connection over the last two seconds | continuous |
| Note: The following characteristics refer to service connections. | | |
| srv_serror_rate | % of connections with "SYN" errors | continuous |
| srv_rerror_rate | % of connections with "REJ" errors | continuous |
| srv_diff_host_rate | % of connections to different hosts | continuous |

# 3.3 Amazon access data Set

It is a limited data set. Less than 10% of the attributes are used for each sample. This data set is divided into two files. One contains user access and the other contains the access history of a given user. This is a limited data set that contains users and their assigned access (Amazon Access Samples Data Set). The file contains 4 attribute categories:

5. [PERSON_{ATTRIBUTE}]: This category describes the "user" to whom access was granted. The column [PERSON_ID] is the column for the file's main key. There is one row per user.
    - PERSON_ID: User ID
    - PERSON_MGR_ID: User administrator ID
    - PERSON_ROLLUP_1: Group of users ID
    - PERSON_ROLLUP_2: Group of users ID
    - PERSON_ROLLUP_3: Group of users ID
    - PERSON_DEPTNAME: Department description ID
    - PERSON_LOCATION: Region ID
    - PERSON_BUSINESS_TITLE: Title ID
    - PERSON_BUSINESS_TITLE_DETAIL: Description ID
    - PERSON_JOB_CODE: Job code ID
    - PERSON_COMPANY: Company ID

- PERSON_JOB_FAMILY: Job family ID

6. [RESOURCE_ {ID}]: The attributes in this category are the resources to which the user may have access. A user will have a 1 in this column if they have access to it, and a 0 if otherwise.
7. [GROUP_ {ID}] - The attributes in this category are groups to which the user may have access. A user will have a 1 in this column if they have access to it, and a 0 if otherwise.
8. [SYSTEM_SUPPORT_ {ID}] - This attribute category is the system for which a user receives support. A user will have a 1 in this column if they receive system support, and a 0 if otherwise.

There are many columns in this data set. Find below a brief description of one of them:

- ACTION: "remove_access" or "add_access".
- TARGET_NAME: {RESOURCE_ID} or {GROUP_ID}.
- LOGIN: the ID of the user obtaining or losing access.
- REQUEST_DATE: (Date/time) format year/month/day time: minute: second.
- AUTHORIZATION_DATE: date and time of authorization, format year/month/day time: minute: second.

Furthermore, this data set contains 30 thousand logins and 20 thousand attributes in the whole set of data.

# 3.4 Ecuador database

The information has been taken from a database of a public entity of the Republic of Ecuador. This same entity has multiple users that interact with a system where they carry out different tasks. The schema of the database is presented in figure 5.

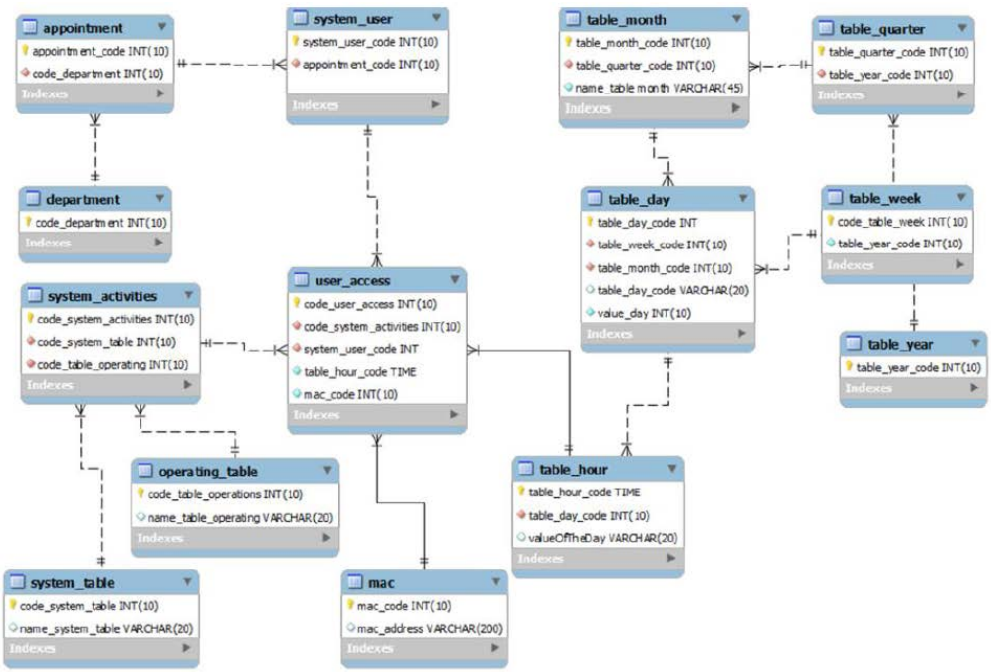



Figure 5. Schema Database Ecuador (Guevara, 2016b).

There are different attributes in the database that will allow us to define a user behavior pattern. This information will describe the activities users carry out in the system, as well as their access date, time and their usual workstation. Table 6 below shows the attributes that may be obtained from the said database.

Table 6. Table of attributes

| Name of the attribute | Type of data | Description |
|---|---|---|
| Date of access | Date Time | Shows the date and time when the user accessed the system. |
| IP | Varchar | Provides the IP of the workstation used to access the system. |
| Country | Varchar | Shows the country in which the user accessed the system. |
| Date of Activity | Date Time | Specifies the date and time when the user carried out Insertion, Updating and Deletion tasks in the database. |
| Affected Table | Varchar | Shows the table where the user interacted with the database. |
| Operation | Varchar | Provides the operations performed on the database. |
| User | Varchar | Shows the user who carried out activities in the database. |

In order to prepare this analysis, a sample of the above mentioned activities and tasks was taken, and charts that illustrate the information in an easier and simpler way were prepared. The sample corresponds to a three-month period and a single user.

Operation Variable: Figure 6 presents an overview of the operations carried out in the system.

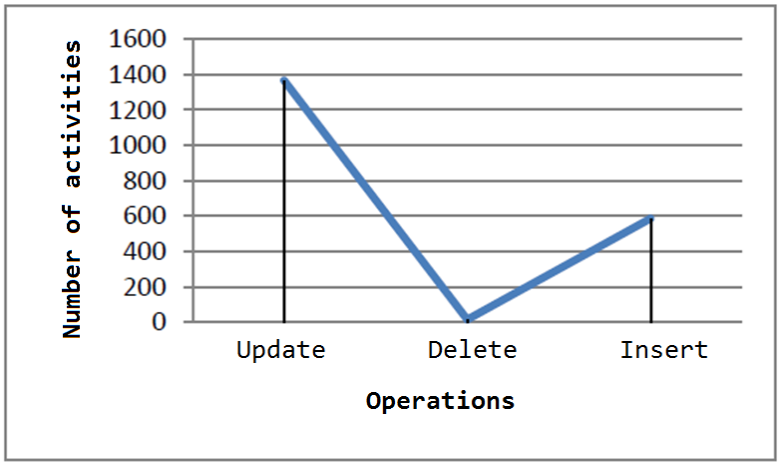



Figure 6. Operation Variable chart for the whole Ecuador data set.

Figure 7 shows the operations that user #6 carried out on the Ecuador database. As it can be seen, the trend that is more common to this user consists in performing more updating and insertion

tasks than deletion tasks. However, deletion tasks are not disregarded in the study, since the combination of the three tasks may be very important for a future behavior pattern.

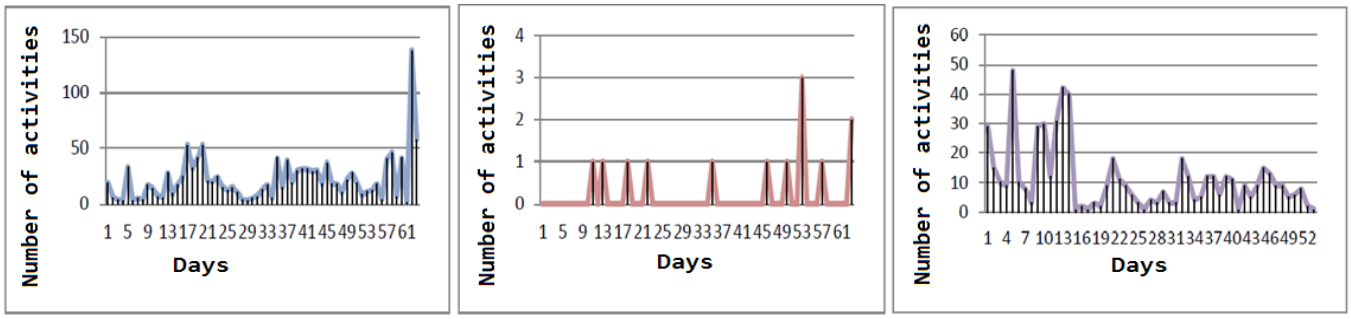



Figure 7. Charts of user #6 in multiple periods of time that show the updating, deletion and insertion operations.

Affected Table Variable: This variable describes tables where activities are performed in the system. They have been named Table 1, 2, 3, and 4 in order to keep the information recorded in each one of them confidential. Figure 8 outlines the distribution of user information according to the tables they have accessed. Figure 9 features the behavior of user #6 in each of the tables during a period of time.

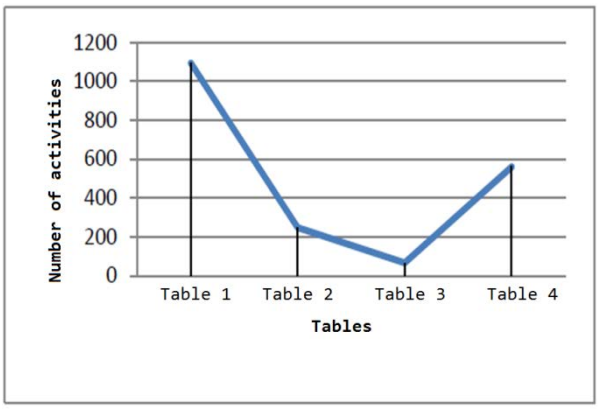



Figure 8. Chart of the distribution of information according to user tables.

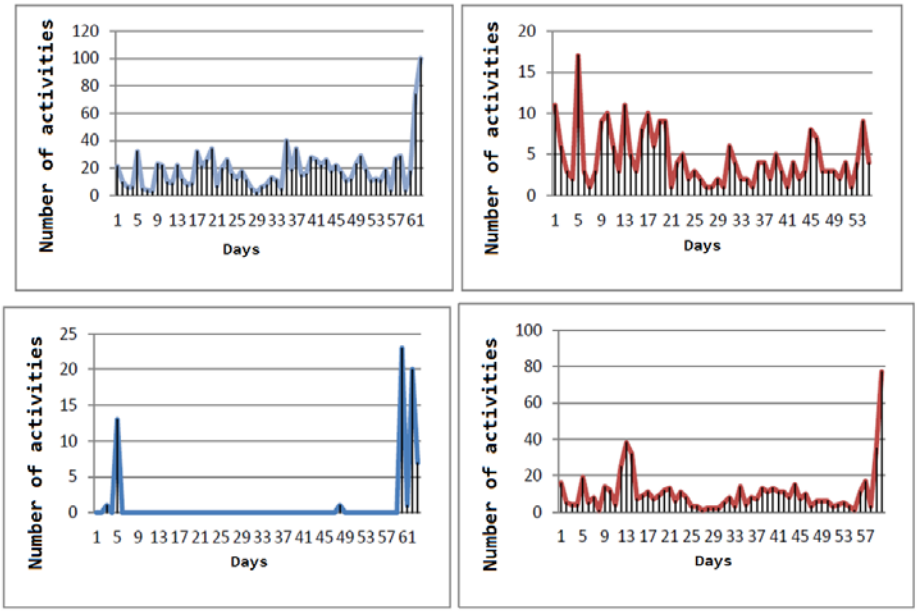



Figure 9. Charts of user #6 in multiple periods of time with Tables 1, 2, 3, and 4.

Date Variable: Figure 10 shows weekdays during which operations are carried out in the system.

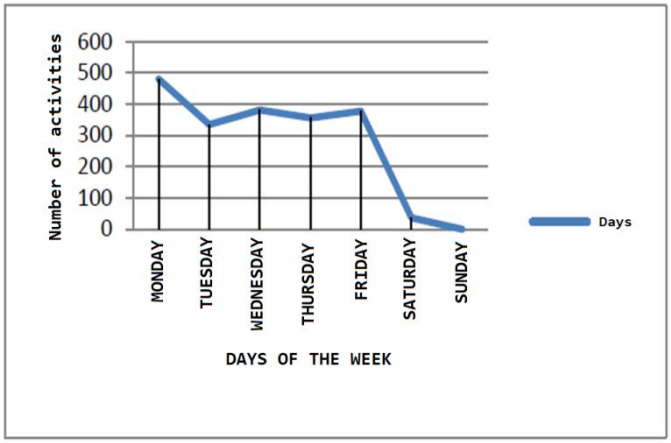



Figure 10. Chart of the operations carried out by users per weekday.

Time Variable: Likewise, figure 11 features accesses using workstations during working hours.

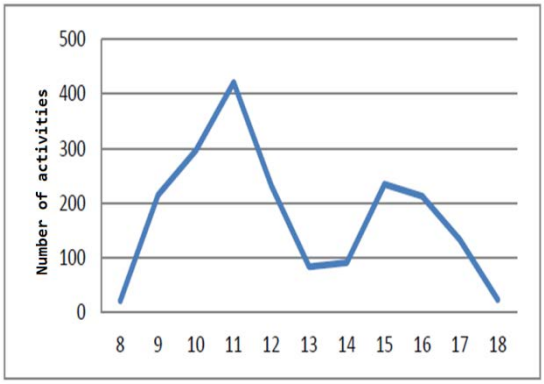


Figure 11. Chart of the operations carried out by users during working hours.

Workstation Variable: Figure 12 represents the IP address of the workstation used to carry out operations in the system.

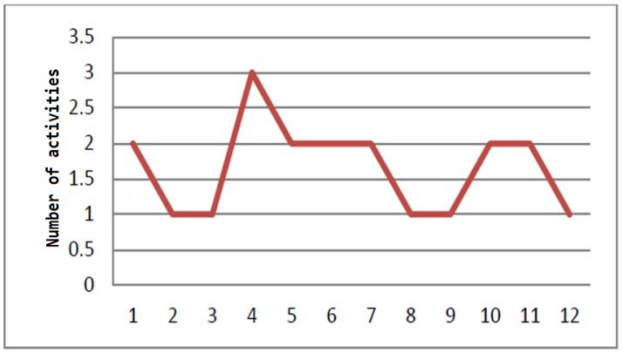



Figure 12. Chart of the operations carried out by users in each of their workstations.

To develop a new dynamic data structure that describes the user behavior is required a fusion of information, it is presented in Figure 13. In the figure 14 shown the schema dynamic and sequential user behavior.

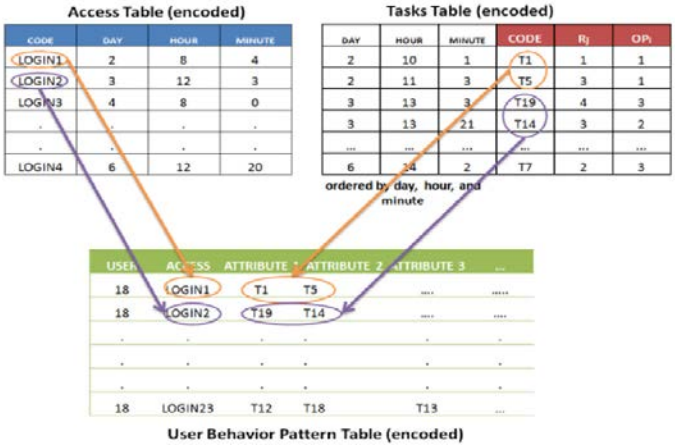



Figure 13. Data structure tasks performed by the user in the database Ecuador (Guevara, 2017).

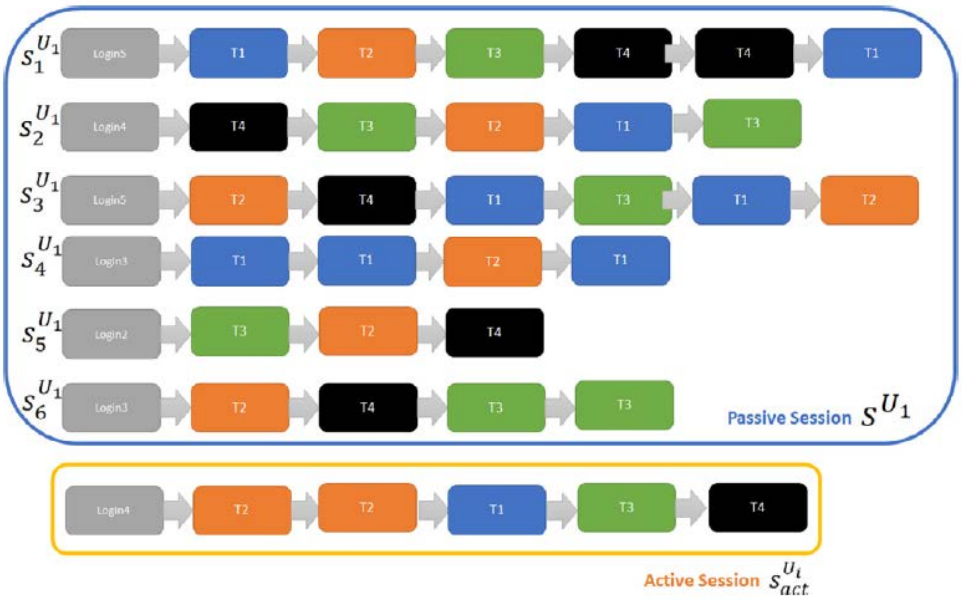



Figure 14. Proposed structure diagram sequential data with the database Ecuador (Guevara, 2017).

# Chapter IV. List of publications and inclusive discussion

This chapter features the publications related to the doctoral research carried out, brought together according to the contributions made to the implemented security systems.

## 4.1. Publications and awards

### Congresses

| | |
|---|---|
| 1) Authors: | C. Guevara, M. Santos, J. A. Martín |
| Title: | Método para la detección de intrusos basado en la sinergia de técnicas de Inteligencia Artificial |
| Congress: | CEDI 2013 Spanish Congress on Computer Science. |
| Place and date: | Madrid, Spain, September 17-20, 2013. |
| Quality score: | National congress |

| | |
|---|---|
| 2) Authors: | C. Guevara, M. Santos, J. A. Martín-H |
| Title: | Identification of computer information system intrusions by decision trees and artificial neural networks |
| Congress: | ISKE 2013 ISKE 8th Int. Conference on Intelligent System and Knowledge Engineering |
| Place and date: | Shenzhen, China, November 20-23, 2013 |
| Quality score: | CORE B Congress |

| | |
|---|---|
| 3) Authors: | C. Guevara, M. Santos, J. Martín, V. López |
| Title: | Reaching a consensus on access detection by a decision-making system |
| Congress: | PIC 2014 International Conference on Progress in Informatics and Computing. DOI: 10.1109/PIC.2014.6972308 |
| Place and date: | Shanghai, China, May 16-18, 2014. |
| Quality score: | IEEE Congress |

| | |
|---|---|
| 4) Authors: | C. Guevara, M. Santos, V. López |
| Title: | Training strategy to improve the efficiency of an intelligent detection system |
| Congress: | International FLINS Conference on Decision Making and Soft Computing |
| Place and date: | João Pessoa (Paraíba), Brazil, August 17-20, 2014. |
| Quality score: | CORE B Congress |

| | |
|---|---|
| 5) Authors: | C. Guevara, M. Santos, V. López |
| Title: | Data leakage detection algorithm based on sequences of activities |
| Congress: | RAID 2014 International Symposium on Research in Attacks. |

| | |
|---|---|
| Place and date: | Gothenburg, Sweden, September 17-19, 2014. |
| Quality score: | CORE A Congress |

| | |
|---|---|
| 6) Authors: | C. Guevara, M. Santos, V. López |
| Title: | Sistema de detección de intrusos aplicando selección negativa en perfiles de usuario |
| Congress: | I Jornadas Nacionales de Investigación en Ciberseguridad |
| Place and date: | León, Spain, September 14-16, 2015. |
| ISBN: | 978-84-9773-742-5; RIASC, University of León, INCIBE |
| Volume, page (year): | 8-13, 2015 |
| Quality score: | National congress |

| | |
|---|---|
| 7) Authors: | C. Guevara, M. Santos, V. López |
| Title: | Sistema inmune artificial para la detección de comportamientos anómalos de usuarios en sistemas informáticos |
| Congress: | CAEPIA 2015 16th Conference of the Spanish Association for Artificial Intelligence |
| Place and date: | Albacete, Spain, November 9-12, 2015. |
| Publication: | Minutes of the 16th Conference of the Spanish Association for Artificial Intelligence |
| ISBN: | 978-84-608-4099-2 |
| Volume, page (year): | 209-218, 2015 |
| Quality score: | National congress |

| | |
|---|---|
| 8) Authors: | C. Guevara, M. Santos, V. López |
| Title: | Intrusion detection with neural networks based on knowledge extraction by decision trees |
| Congress: | CISIS 2016 9th International Conference on Computational Intelligence in Security for Information Systems. DOI: 10.1007/978-3-319-47364-2_49 |
| Place and date: | San Sebastián, Spain, October 19-21, 2016. |
| Quality score: | CORE B Congress |

## Journals:

| | |
|---|---|
| 9) Authors: | C. Guevara, M. Santos, V. López |
| Title: | Negative selection and Knuth-Morris-Pratt algorithm for anomaly detection |
| Journal: | IEEE Latin America Transactions. ISSN: 1548-0992. DOI: 10.1109/TLA.2016.7459637 |

| | |
|---|---|
| Volume, page (year): | 2016 |
| Quality score: | JCR 2015 : 0.436 (2015) Engineering, Electrical & Electronic 219/257 (Q4) |

| | |
|---|---|
| 10) Authors: | C. Guevara, M. Santos, V. López |
| Title: | Data leakage detection algorithm based on task sequences and probabilities |
| Journal: | Knowledge-Based Systems<br>Elsevier, ISSN: 0950-7051. DOI: 10.1016/j.knosys.2017.01.009 |
| Volume, page (year): | in press, 2017 |
| Quality score: | JCR 2015: 3.325 (2015) Computer Science, Artificial Intelligence 17/130 (Q1) |

### 4.1.1. Awards

4. "Intrusion detection system based on the negative selection of user profiles". Awarded the prize for the best student paper in the JNIC 2015 Cybersecurity Research National Conferences, held on September 14-16, 2015, in León, Spain.
5. "ILLALPA Information leakage detection system". Finalist in the 2014 Cyber Camp project presentation, held on December 1-4, 2014, in Madrid, Spain.
6. "Identification of computer information system intrusions by decision trees and artificial neural networks". Awarded the prize for the best scientific poster at the ISKE 2013 8th International Conference on Intelligent Systems and Knowledge Engineering, held in November 2013 in Shenzhen, China.

## 4.2. Publications on intrusion detection systems

The application of artificial intelligence techniques and the new data structure proposed has lead to the significant reduction of false positives in intrusion detection in this doctoral research.

The paper published in the **CEDI 2013 (**Guevara et al., 2013a) presents the creation of a metaclassifier with a combination of various classification techniques, such as the multilayer neural network, support vector machines, and decision trees, in order to develop an intrusion detection system that uses the KDD 99 database. The developed system allows for the application of the said classification techniques and their combination to increase efficiency and reliability in the detection of users in IT networks or systems. A scenario has been designed to test this model, in order to demonstrate the validity of the proposal and to confirm future lines of action. Good results have been obtained with an accuracy of 99% and a false positive rate of 1%, which shows the efficiency of the proposal.

The paper of the congress **ISKE 2013** (Guevara et al., 2013b) features an intrusion detection system, which is the same as that which uses a neural network to classify the tasks carried out by the user during working hours (day, time and operation), within a period of 2 years (Ecuador database),

to subsequently confirm, by using decision trees and Bayes' theorem, whether it is an intrusion or normal user behavior. Moreover, the paper presents a graphical interface that shows the behavior of each user as the result of the detection by obtaining acceptable results. The purpose of this paper is to show how different intelligent techniques can be applied to identify the users who have access to critical information and to verify whether access is allowed or not. Advanced and intelligent data mining and analysis techniques have been applied, such as decision trees and artificial neural networks, in order to obtain user behavior patterns. Dynamic user profiles have been obtained. The main contribution of this paper is to show effective solutions to detect intrusions in computer systems.

The paper published at the congress **PIC 2014** (Guevara et al., 2014a) develops a metaclassifier by combining the following four classification techniques: ID3, Naive Bayes, UCS and C4.5, to classify network attacks by using the KDD 99 database. In order to obtain a proper detection, the result of the metaclassifier is controlled by a dynamic system of rules, which leads to a more effective identification. The rate of correctly classified attacks is of 99.99%, with a very low false positive rate.

Another paper published in the congress **FLINS 2014** (Guevara et al., 2014c) creates a model based on the combination of two classification techniques, C4.5 and UCS, to identify unauthorized accesses to resources, by using the Amazon database (Kaggle challenge). This proposal divides the data set into three parts and includes the classifier, the same which detects if a behavior is normal or intrusive. This training strategy has proven to be more effective than if the whole database had been used as a single set. The accuracy of the classifier is of 98.4% in the training stage, and of 98.1% in the testing stage, which is a good detection result.

The paper presented at the national congress **JNIC 2015** (Guevara et al., 2015a) presents an IDS by applying negative selection, using the Ecuador database to explain the proposal, and a sequential data structure of each of the user sessions detailed in paper (Guevara et al., 2014c), broadened in order to be used in this research. The normal behavior of each user is used to generate abnormal behavior, in order to identify future intrusions in new sessions. The specific contributions are: data model, user profile identification, and the application of negative selection algorithms and local sequential search algorithms such as KPM to efficiently identify intrusive tasks. The proposal is 99.31% accurate, and presents a false positive rate of 0.44%.

The paper published in the congress **CAEPIA 2016** (Guevara et al., 2015b) presents an IDS that applies the artificial human immune system to detect abnormal behavior within a computer system. The main contributions of this paper are: the development process of a dynamic data model, the classification of the most and least frequent tasks per user, presented in papers (Guevara et al., 2014c; Guevara et al., 2015b) and broadened in this paper, as well as the application of an algorithm of artificial immune systems, such as negative selection, to generate binary sequence detectors of abnormal tasks to subsequently detect dangerous user behavior by applying the Knuth-Morris-Pratt (KMP) local searching algorithm. This paper has obtained good results, with an accuracy of 95.5%, which is quite positive. However, the average detection time is 74.99 seconds, which is very slow for a real system.

The congress **CISIS 2016** (Guevara et al., 2016b) also tackles the issue of detecting intrusions or unauthorized access to computers, which has always been critical regarding information systems, where security, integrity, and privacy are key issues. Even though more and more sophisticated and effective detection strategies are being developed and implemented, both in hardware and software, they still have to be improved in order to completely eradicate illegitimate access. The purpose of this paper is to show how IT techniques can be used to identify unauthorized access to computers. The advanced analysis of data is first applied to obtain a qualitative approach. Decision trees are used to obtain user behavior patterns, and neural networks are applied as classifiers to identify intrusion detection. The application of combined intelligent techniques to real data has shown encouraging results.

Ultimately, the paper published in the journal **IEEE Latin America Transactions** (Guevara et al., 2017) uses the Knuth-Morris-Pratt algorithm (sequence alignment) and negative selection to generate abnormal subsequences based on normal user behavior and look for them in a new user session. The paper is based on information about the behavior of authorized users who have carried out multiple tasks in a computer system for two years. The analysis uses a dynamic data structure that can encrypt users' current activities and behavior, as shown in papers (Guevara et al., 2014c; Guevara et al., 2015a), improved and broadened to be used in this research. Identifying the most and least frequent tasks, according to the historical database of each user, is an easy way to create a single behavior profile. With this profile, we apply negative selection techniques in order to obtain a reasonable set of abnormal detectors of computational dimensions. Afterwards, the Knuth-Morris-Pratt algorithm is applied to locate anomaly detectors as an indication of a fraudulent behavior. This procedure, aimed at detecting abnormal behavior, has been tested with real data. The results show the effectiveness of the proposal and encourage further research to improve the current detection system. The results present an average accuracy of 98% and an average false positive rate of 5%.

## 4.3. Publications on data leakage detection systems

The paper of the congress **RAID 2015** (Guevara et al., 2014c) was prepared as part of the publications of this doctoral research. It builds on the search of abnormal subsequences to detect information leakages in user sessions in the Ecuador database, presented in section 3.1. This algorithm uses the proposed sequential data structure based on user tasks carried out during a user session. The PageRank algorithm (Page, 1999) is used to group the most frequent tasks of each user, to subsequently calculate the probability by applying Bayes' theorem. Good results have been obtained. This algorithm works with historical data of the activities of authorized users within a computer system. This information gathers data regarding access, such as the time, duration, day of the week, operation, table which was accessed, etc. The data are provided by a government institution of Ecuador called Inmobiliar. The process has two stages: the first one is based on the calculation of the probability of each activity carried out by each user, such as modifying, deleting or copying a file, etc. Different activities at different times are encrypted with an integer or a character. The PageRank algorithm is used to calculate the probability of each activity. However, activities create sequences. That is, users perform different activities, one after the other, during a session (time lapsed from login to logout). These activity sequences may vary in length. The probability of

each activity sequence is calculated by applying Bayes' theorem. The minimum of these conditional probabilities is obtained and defined as a threshold, $rmin$. A new activity chain, $s_i$, is implemented in the leakage detection system. The PageRank algorithm is applied first, followed by Bayes' theorem. Thus, we obtain the probability for such activity sequence for a specific user $p_i$ . This probability, $p_i$, is compared to the threshold, $rmin$. If the threshold is not exceeded, a message warning of a possible leakage appears. Otherwise, the activity sequence enters the second stage of the process. The activity sequence being evaluated is compared to all the activity sequences of the user in question that are stored in the historical database. By applying the Smith-Waterman algorithm, we obtain a similarity score for this subsequence compared to the rest of the sequences previously carried out by the user. This algorithm determines similar regions between two chains by comparing segments of all possible lengths and maximizes similarity. If the score is higher than a second threshold, the activity may be considered correct. Otherwise, it may be a leakage. Although the said algorithm is still under development, its main contribution is the way it manages different sequence lengths and how it works with each user's behavior pattern.

Another paper of this doctoral dissertation has been published in the journal **KNOSYS in 2017** (Guevara et al., 2017). It features an algorithm to detect information leakages by using the Ecuador database and UNIX Commands. This research proposes a new algorithm to detect the abnormal behavior of users during working sessions in computer systems. This algorithm is the same as that used for the data structure developed for papers (Guevara et al., 2014c; Guevara et al., 2015a; Guevara et al., 2015b; Guevara et al., 2017), broadened in detail in this paper. First, we identify each authorized user's behavior profile and the computational tasks they usually perform in the system files. Then, a new session is encrypted as a task sequence (with length 2 subsequences) and an algorithm is applied according to the probability of the said sequences. The activities classified as potential anomalies are verified twice by using Markov chains. The process has proven to be effective, as it presents a high detection accuracy and a low false positive rate. It has been validated in a real database provided by a government institution of Ecuador, as well as in a public data set of UNIX Commands. Furthermore, it has been shown that the algorithm is effective with regard to the computational time, and that the overloading of this monitoring software is low. The accuracy of correct classifications is of 94.78%, and the false positive rate is of 4.73%.

# Chapter V: Conclusions and future works

## 5.1. Conclusions

Information security has developed and evolved significantly over the last few decades, given that it is a very important research area for people and organizations around the world. This area focuses on designing and implementing many preventive and reactive technological systems that allow organizations to safeguard and protect their information in order to uphold their data confidentiality, availability, and integrity. Due to the increase in the number of cyberattacks to public and private IT systems, there has been a need to develop new strategies, models, algorithms, and systems to analyze and identify possible fraudulent activities in information systems.

- This dissertation provides different contributions to different areas. Firstly, a new sequential data structure of tasks (commands and activities) has been implemented, which allows the application of machine learning and sequence alignment algorithm techniques. This structure was applied to two different databases, a public one of UNIX Commands, and a private one which belongs to a system of the government of the Republic of Ecuador.
- Secondly, metaclassifier models have been implemented to improve the efficiency of detection and identification of intrusions. They show good results in terms of accuracy. Specifically, C4.5 and ID3 decision trees, neural networks, UCS genetic algorithms, SVM support vector machines, Naive Bayes, and other similar methods have been applied, for the classification of normal and abnormal behavior of each of the users of the analyzed IT systems.
- Another contribution is the application of artificial immune systems, negative selection, and Knuth-Morris-Pratt (sequence alignment), which based on the expected user behavior (according to the user history database), generate abnormal behavior and look for a fraudulent subsequence in the execution sequence of tasks carried out by the user. The detection is more accurate and efficient. Test phase results show a low false positive rate.
- Finally, another of the contributions of our research has been the development of a new information leakage detection algorithm based on the probability of users carrying out certain tasks. Markov chains were applied in this research to perform the second verification of the result obtained by the anomaly detection algorithm.

## 5.2. Future lines works

We pose future lines of action based on the contributions achieved as a result of this research.

Concerning the algorithm developed with Markov chains, the identification of abnormal subsequences can be improved by applying Deep Learning in order to improve the accuracy of the classifier and the detection time.

For anomaly and intrusion detection systems we propose, firstly, to apply Big Data processing algorithms (Ramírez, 2017). We could also group all the information of different computer systems, social networks, mobile devices, etc., in order to define a more real profile for each user. Thus, the detection of abnormal behavior would be more efficient.

Ultimately, we also suggest the implementation of biometric systems based on heart rate, pupil movement, writing speed, etc., for the user profile to be unique and more efficient in order to detect potential cyberattacks.

# Referencias


1. Aeberhard, S., Coomans, D., & de Vel, O. (1994). The performance of statistical pattern recognition methods in high dimensional settings. In IEEE signal processing workshop on higher order statistics, Ceasarea (Vol. 4, pp. 14-6).
2. Aickelin, U., Dasgupta, D., & Gu, F. (2014). Artificial immune systems. In Search Methodologies (pp. 187-211). Springer US.
3. Alomari O, Othman ZA. (2012)."Bees algorithm for feature selection in network anomaly detection". Journal of Applied Sciences Research; 8:1748–56.
4. Anderson, R. H. (1999). "Research and Development Initiatives Focused on Preventing, Detecting, and Responding to Insider Misuse of Critical Defense Information Systems". (No. RAND-CF-151-OSD).
5. Anderson, R. H., Bozek, T., Longstaff, T., Meitzler, W., Skroch, M., &VanWyk, K. (2000). "Research on mitigating the insider threat to information systems # 2". (RAND CF-163-DARPA). Technical report, RAND Corporation.
6. Anderson, R. H., &Brackney, R. (2004). "Understanding the insider threat". RAND Corporation.
7. Agrawal, R., & Kiernan, J. (2002). Watermarking relational databases. In Proceedings of the 28th international conference on Very Large Data Bases (pp. 155-166). VLDB Endowment.
8. Aziz, A. S. A., Salama, M., ella Hassanien, A., & Sanaa, E. L. (2012, September). Detectors generation using genetic algorithm for a negative selection inspired anomaly network intrusion detection system. In Computer Science and Information Systems (FedCSIS), 2012 Federated Conference on (pp. 597-602). IEEE.
9. Aziz, A. S. A., Azar, A. T., Hassanien, A. E., & Hanafy, S. E. O. (2014). Negative Selection Approach Application in Network Intrusion Detection Systems. arXiv preprint arXiv:1403.2716.
10. Bace, R., & Mell, P. (2001). NIST special publication on intrusion detection systems. BOOZ-ALLEN AND HAMILTON INC MCLEAN VA.
11. Balinsky, H., Perez, D. S., & Simske, S. J. (2011). System call interception framework for data leak prevention. In Enterprise Distributed Object Computing Conference (EDOC), 2011 15th International (pp. 139-148). IEEE.
12. Bhuyan, M. H., Bhattacharyya, D. K., & Kalita, J. K. (2014). Network anomaly detection: methods, systems and tools. Ieee communications surveys & tutorials, 16(1), 303-336.
13. Bonatti, P., De Capitani di Vimercati, S., & Samarati, P. (2002). "An algebra for composing access control policies". ACM Transactions on Information and System Security (TISSEC), 5(1), 1-35.
14. Borders, K., & Prakash, A. (2009). "Quantifying information leaks in outbound web traffic". In Security and Privacy, 30th IEEE Symposium on (pp. 129-140).IEEE.

15. Burke, B. E., & Christiansen, C. A. (2009). "Insider risk management: A framework approach to internal security". RSA, the Security Division of EMC.
16. Cisco Systems. (2008). "Data leakage worldwide: Common risks and mistakes employees make". Web.
17. Couture M. (2012)."Real time intrusion prediction based on optimized alerts with hidden Markov model". Journal of Networks; 7:311–21.
18. Chasmer, L., Hopkinson, C., Veness, T., Quinton, W., & Baltzer, J. (2014). A decision-tree classification for low-lying complex land cover types within the zone of discontinuous permafrost. Remote Sensing of Environment, 143, 73-84.
19. Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. ACM computing surveys (CSUR), 41(3), 15.
20. Chandola, V., Banerjee, A., & Kumar, V. (2012). Anomaly detection for discrete sequences: A survey. IEEE Transactions on Knowledge and Data Engineering, 24(5), 823-839.
21. Chung YY, Wahid N. (2012)."A hybrid network intrusion detection system using simplified swarm optimization (SSO)". Applied Soft Computing; 12:3014–22.
22. DasGupta, D. (1993). An overview of artificial immune systems and their applications. In Artificial immune systems and their applications (pp. 3-21). Springer Berlin Heidelberg.
23. Dasgupta, D., KrishnaKumar, K., Wong, D., & Berry, M. (2004, September). Negative selection algorithm for aircraft fault detection. In International Conference on Artificial Immune Systems (pp. 1-13). Springer Berlin Heidelberg.
24. Debar, H., Dacier, M., & Wespi, A. (1999). Towards a taxonomy of intrusion-detection systems. Computer Networks, 31(8), 805-822.
25. Debar, H., Dacier, M., & Wespi, A. (2000). "A revised taxonomy for intrusion detection systems". In Annales des telecommunications, Vol. 55, No. 7-8, pp. 361-378. Springer-Verlag.
26. Debar, H., & Viinikka, J. (2005). Intrusion detection: Introduction to intrusion detection and security information management. In Foundations of security analysis and design III (pp. 207-236). Springer Berlin Heidelberg.
27. Deepa A. and V. Kavitha. (2012) "A comprehensive survey on approaches to intrusion detection system". Procedia Engineering, vol. 38, pp. 2063–2069.
28. Demuth, H. B., Beale, M. H., De Jess, O., & Hagan, M. T. (2014). Neural network design. Martin Hagan.
29. Fawcett, T. (2010). "ExFILD: A tool for the detection of data exfiltration using entropy and encryption characteristics of network traffic". (Doctoral dissertation, University of Delaware).
30. Forrest, S., & Hofmeyr, S. A. (2001). Immunology as information processing. In SANTA FE INSTITUTE STUDIES IN THE SCIENCES OF COMPLEXITY-PROCEEDINGS VOLUME- (pp. 361-388). Reading, Mass.; Addison-Wesley.

31. Gade, S., Munde, K. K., & RV, K. (2012).Data Allocation Strategies for Leakage Detection.
32. Gharpande, A. O., & Deshmukh, M. V. (2013). Data Leakage Detection. International Journal of computer science and applications, 6(2).
33. García-Teodoro, P., Diaz-Verdejo, J., Maciá-Fernández, G., & Vázquez, E. (2009). "Anomaly-based network intrusion detection: Techniques, systems and challenges. Computers & security".28(1), 18-28.
34. García Villalba, L. J., Orozco, A. L. S., & Vidal, J. M. (2015). Malware Detection System by Payload Analysis of Network Traffic. IEEE Latin America Transactions, 13(3), 850-855.
35. Gao, B., Ma, H. Y., & Yang, Y. H. (2002). "Hmms (hidden Markov models) based on anomaly intrusion detection method". In Machine Learning and Cybernetics, 2002. Proceedings. 2002 International Conference on (Vol. 1, pp. 381-385). IEEE.
36. Goldberg, D. E., & Holland, J. H. (1988). Genetic algorithms and machine learning. Machine learning, 3(2), 95-99.
37. Gong, R. H., Zulkernine, M., & Abolmaesumi, P. (2005, May). "A software implementation of a genetic algorithm based approach to network intrusion detection". In Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, 2005 and First ACIS International Workshop on Self-Assembling Wireless Networks. SNPD/SAWN 2005. Sixth International Conference on (pp. 246-253). IEEE.
38. Gong, M., Zhang, J., Ma, J., & Jiao, L. (2012). An efficient negative selection algorithm with further training for anomaly detection. Knowledge-Based Systems, 30, 185-191.
39. González, F. A., & Dasgupta, D. (2003). Anomaly detection using real-valued negative selection. Genetic Programming and Evolvable Machines, 4(4), 383-403.
40. Guevara, C., Santos, M., & Martín, J. A. (2013a). Método para la Detección de Intrusos basado en la Sinergia de Técnicas de Inteligencia Artificial. In Proceedings of the IV Congreso Español de Informática CEDI (pp. 963-972).
41. Guevara, C., Santos, M., & Martín, J. A. (2013b). Identification of computer information system intruders by decision trees and artificial neural networks. In 8th Int. Conference on Intelligent System and Knowledge Engineering ISKE 2013.
42. Guevara, C., Santos, M., Martin, J. A., & López, V. (2014a). Reaching a consensus on access detection by a decision system. In Progress in Informatics and Computing (PIC), 2014 International Conference on (pp. 119-122). IEEE.
43. Guevara, C., Santos, M., & López, V. (2014b). Training strategy to improve the efficiency of an intelligent detection system. In Decision Making and Soft Computing: Proceedings of the 11th International FLINS Conference (pp. 544-549).
44. Guevara, C., Santos, M., & López, V. (2014c). Data leakage detection algorithm based on sequences of activities. In Proceedings of the 17th International Symposium Research in Attacks, Intrusions and Defenses RAID (Vol. 8688, pp. 477-478).

45. Guevara, C. B. , Santos, M., & López, V. L. (2015a). Sistema de Detección de Intrusos aplicando Selección Negativa en Perfiles de Usuario. In Actas de las primeras Jornadas Nacionales de Investigación en Ciberseguridad: León, 14, 15, 16 de septiembre de 2015: I JNIC2015 (pp. 8-13). Servicio de Publicaciones.
46. Guevara, C. B. G., Santos, M., & López, V. (2015b). Sistema Inmune Artificial para la Detección de Comportamientos Anómalos de Usuarios en Sistemas Informáticos.
47. Guevara, C. B., Santos, M., & López, M. V. (2016a). Negative selection and knuth morris pratt algorithm for anomaly detection. IEEE Latin America Transactions, 14(3), 1473-1479.
48. Guevara, C., Santos, M., & López, V. (2016b). Intrusion Detection with Neural Networks Based on Knowledge Extraction by Decision Tree. In International Conference on European Transnational Education (pp. 508-517). Springer International Publishing.
49. Guevara, C., Santos, M., & López, V. (2017). Data Leakage Detection Algorithm based on Task Sequences and Probabilities. Knowledge-Based Systems.
50. Hao, F., Kodialam, M., Lakshman, T. V., & Zhang, H. (2005, October)."Fast payload-based flow estimation for traffic monitoring and network security". In Architecture for networking and communications systems, 2005. ANCS 2005. Symposium on (pp. 211-220).IEEE.
51. Heady, R., Luger, G. F., Maccabe, A., & Servilla, M. (1990). The architecture of a network level intrusion detection system (pp. 3-6). University of New Mexico. Department of Computer Science. College of Engineering.
52. Helman, P., & Bhangoo, J. (1997). A statistically based system for prioritizing information exploration under uncertainty. IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans, 27(4), 449-466.
53. Herbig, K. L., & Wiskoff, M. F. (2002). "Espionage against the United States by American citizens 1947-2001". (No. PERSEREC-TR-02-5). Defense Personnel Security Research Center Monterey CA.
54. Hofmeyr, S. A., & Forrest, S. (2000). Architecture for an artificial immune system. Evolutionary computation, 8(4), 443-473.
55. HorngSJ, Su MY, Chen YH, Kao TW, Chen RJ, Lai JL, Perkasa CD. "A novel intrusion detection system based on hierarchical clustering and support vector machines". Expert Systems with Applications 2011; 38:306–13.
56. Info Watch Research Center. (2012). "Global Data Leakages & Insider Threats Report, 2012".
57. Jajodia, S., Samarati, P., Sapino, M. L., & Subrahmanian, V. S. (2001). "Flexible support for multiple access control policies". ACM Transactions on Database Systems (TODS), 26(2), 214-260.
58. Javitz, H. S., Valdes, A., & NRaD, C. (1993). The NIDES statistical component: Description and justification. Contract, 39(92-C), 0015.

59. Jiang, X., Wang, X., & Xu, D. (2010). "Stealthy malware detection and monitoring through VMM-based out of the box semantic view reconstruction". ACM Transactions on Information and System Security (TISSEC), 13(2), 12.
60. Jung, J., Sheth, A., Greenstein, B., Wetherall, D., Maganis, G., & Kohno, T. (2008, October). "Privacy oracle: a system for finding application leaks with black box differential testing". In Proceedings of the 15th ACM conference on Computer and communications security (pp. 279-288). ACM.
61. Kabiri P, Ghorbani AA. (2005)."Research on intrusion detection and response: a survey". International Journal of Network Security; 1:84–102.
62. Kamra, A., Terzi, E., & Bertino, E. (2008). "Detecting anomalous access patterns in relational databases". The VLDB Journal, 17(5), 1063-1077.
63. Kantzavelou I, Katsikas S. (2010). "A game-based intrusion detection mechanism to confront internal attackers". Computers & Security; 29:859–74.
64. Karhula, Paivikki. (2011)."What is the effect of WikiLeaks for Freedom of Information?" FAIFE Spotlight [online: http://www.ifla.org/publications/what-is-the-effect-of-wikileaks-for-freedom-of-information].
65. Kenkre, P. S., Pai, A., & Colaco, L. (2015). "Real time intrusion detection and prevention system". In Proceedings of the 3rd International Conference on Frontiers of Intelligent Computing: Theory and Applications (FICTA) (pp. 405-411). Springer International Publishing.
66. Kephart, J. O., Sorkin, G. B., Arnold, W. C., Chess, D. M., Tesauro, G. J., White, S. R., & Watson, T. J. (1995, August). Biologically inspired defenses against computer viruses. In IJCAI (1) (pp. 985-996).
67. Kim, J., & Bentley, P. J. (2001). An evaluation of negative selection in an artificial immune system for network intrusion detection. In Proceedings of the 3rd Annual Conference on Genetic and Evolutionary Computation (pp. 1330-1337). Morgan Kaufmann Publishers Inc.
68. Kim, J., & Kim, H. J. (2010). Design of internal information leakage detection system considering the privacy violation. In Information and Communication Technology Convergence (ICTC), International Conference on (pp. 480-481). IEEE.
69. Kleinberg, J., Papadimitriou, C. H., & Raghavan, P. (2001). "On the value of private information". In Proceedings of the 8th conference on Theoretical aspects of rationality and knowledge (pp. 249-257). Morgan Kaufmann Publishers Inc.
70. Kolias C, Kambourakis G, Maragoudakis M. (2011)."Swarm intelligence in intrusion detection: a survey". Computers& Security; 30:625–42.
71. Kumar, V., Srivastava, J., & Lazarevic, A. (Eds.). (2006). Managing cyber threats: issues, approaches, and challenges (Vol. 5). Springer Science & Business Media.
72. Kumar, A. (2013)."Comparative evaluation of algorithms for effective data leakage detection". Information & Communication Technologies (ICT), 2013 IEEE Conference on. IEEE.

73. Laurentys, C. A., Ronacher, G., Palhares, R. M., & Caminhas, W. M. (2010). Design of an artificial immune system for fault detection: a negative selection approach. Expert Systems with Applications, 37(7), 5507-5513.
74. Lazarevic, A., Ertoz, L., Kumar, V., Ozgur, A., & Srivastava, J. (2003, May). A comparative study of anomaly detection schemes in network intrusion detection. In Proceedings of the 2003 SIAM International Conference on Data Mining (pp. 25-36). Society for Industrial and Applied Mathematics.
75. Lazarevic, A., Kumar, V., &Srivastava, J. (2005). "Intrusion detection: A survey". In Managing Cyber Threats (pp. 19-78). Springer US.
76. Li, Zhang G, Nie J, Niu Y, Yao A. (2012). "The application of genetic algorithm to intrusion detection in MP2P network". In: Third international conference on advances in swarm intelligence, Shenzhen, China, pp. 390–397.
77. Liao, H. J., Lin, C. H. R., Lin, Y. C., & Tung, K. Y. (2013). Intrusion detection system: A comprehensive review. Journal of Network and Computer Applications, 36(1), 16-24.
78. Modi, D Patel, B Borisaniya, H Patel, A Patel, M Rajarajan, (2012)."A survey of intrusion detection techniques in Cloud". Journal of Network and Computer Applications.
79. Mukherjee B, Heberlein LT, Levitt KN. (1994).Network intrusion detection. IEEE Network; 8:26–41.
80. Murali A, Rao M."A survey on intrusion detection approaches". (2005). In: First international conference information and communication technologies, Karachi, Pakistan, pp. 233–240.
81. Page, L., Brin, S., Motwani, R., & Winograd, T. (1999). The PageRank citation ranking: Bringing order to the web. Stanford InfoLab.
82. Papadimitriou, P., & Garcia-Molina, H. (2011)."Data leakage detection". Knowledge and Data Engineering, IEEE Transactions on, 23(1), 51-63.
83. Patcha, A., & Park, J. M. (2007). "Network anomaly detection with incomplete audit data". Computer Networks, 51(13), 3935-3955.
84. Patil, S. J., & Bhavsar, A. K. (2012). Implementation of data watcher in data leakage detection system.
85. Peng, L., Chen, W., Xie, D., Gao, Y., & Liang, C. (2013). Dynamically real-time anomaly detection algorithm with immune negative selection. Applied Mathematics & Information Sciences, 7(3), 1157.
86. Raj, S. R., Cherian, A., & Abraham, A. (2013). A Survey on Data Loss prevention Techniques. International Journal of Science and Research, 2(4), 240-241.
87. Ramírez Gallego, S., Lastra, I., Martínez Rego, D., Bolón Canedo, V., Benítez, J. M., Herrera, F., & Alonso-Betanzos, A. (2017). "Fast mRMR: Fast Minimum Redundancy Maximum Relevance Algorithm for High-Dimensional Big Data". International Journal of Intelligent Systems, 32(2), 134-152.
88. Roesch, M. (1999, November). "Snort: Lightweight Intrusion Detection for Networks". In LISA (Vol. 99, No. 1, pp.229-238).

89. Russell, D., & Gangemi, G. T. (1991). "Computer security basics". O'Reilly Media, Inc.
90. Sabahi F, Movaghar A. "Intrusion detection: a survey". In: Third international conference on system and network communication, Sliema, Malta, 2008, pp. 23–26.
91. Scarfone, K., & Mell, P. (2007). Guide to intrusion detection and prevention systems (idps). NIST special publication, 800(2007), 94.
92. Sen S, Clark JA. "Evolutionary computation techniques for intrusion detection in mobile ad hoc networks". Computer Networks2011; 55:3441–57.
93. Stann, F., & Heidemann, J. (2003). RMST: Reliable data transport in sensor networks. In Sensor Network Protocols and Applications, 2003. Proceedings of the First. International Workshop on (pp. 102-112). IEEE.
94. Stavroulakis, P., & Stamp, M. (2010). "Handbook of information and communication security". Springer Science & Business Media.
95. Shaw, E., Ruby, K., & Post, J. (1998). "The insider threat to information systems: The psychology of the dangerous insider". Security Awareness Bulletin, 2(98), 1-10.
96. Shu, X., & Yao, D. D. (2013). "Data leak detection as a service". In Security and Privacy in Communication Networks (pp. 222-240). Springer Berlin Heidelberg.
97. Schultz, E. E. (2002). "A framework for understanding and predicting insider attacks". Computers & Security, 21(6), 526-531.
98. Takebayashi, T. (2010)."Data loss prevention technologies". Fujitsu Scientific and Technical Journal 46.1:47-55.
99. Tene, O., & Polonetsky, J. (2012). Big data for all: Privacy and user control in the age of analytics. Nw. J. Tech. & Intell. Prop., 11, xxvii.
100. Tizard, I. R. (1992). Immunology, an introduction. Brooks Cole.
101. Torgo, L., & Lopes, E. (2011, July). Utility-based fraud detection. In IJCAI Proceedings-International Joint Conference on Artificial Intelligence (Vol. 22, No. 1, p. 1517).
102. Tsai, C. F., Hsu, Y. F., Lin, C. Y., & Lin, W. Y. (2009). "Intrusion detection by machine learning: A review". Expert Systems with Applications, 36(10), 11994-12000.
103. Wang, W., & Gombault, S. (2007). Distance Measures for Anomaly Intrusion Detection. In Security and Management (pp. 17-23).
104. Warrender, C., Forrest, S., & Pearlmutter, B. (1999). Detecting intrusions using system calls: Alternative data models. In Security and Privacy, 1999. Proceedings of the 1999 IEEE Symposium on (pp. 133-145). IEEE.
105. Xie, M., Han, S., Tian, B., & Parvin, S. (2011). "Anomaly detection in wireless sensor networks: A survey". Journal of Network and Computer Applications, 34(4), 1302-1325.
106. Ye, N., Li, X., Chen, Q., Emran, S. M., & Xu, M. (2001). "Probabilistic techniques for intrusion detection based on computer audit data". IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans, 31(4), 266-274.
107. Zurutuza, U., & Uribeetxeberria, R. (2005). Revisión del estado actual de la investigación en el uso de data mining para la detección de intrusiones. Escuela Politécnica Superior de Mondragón, Departamento. Informática.

**Websites**

108. KDD Cup 1999 Data Set, UCI Machine Learning Repository. https://archive.ics.uci.edu/ml/datasets/KDD+Cup+1999+Data.
109. Amazon Access Samples Data Set, UCI Machine Learning Repository. https://archive.ics.uci.edu/ml/datasets/Amazon+Access+Samples.
110. UNIX User Data Set, UCI Machine Learning Repository. https://archive.ics.uci.edu/ml/datasets/UNIX+User+Data.

# Parte III: Publicaciones

# Lista de publicaciones


1. Guevara, C., Santos, M., & Martín, J. A. (2013a). Método para la Detección de Intrusos basado en la Sinergia de Técnicas de Inteligencia Artificial. In Proceedings of the IV Congreso Español de Informática CEDI (pp. 963-972).
2. Guevara, C., Santos, M., & Martín, J. A. (2013b). Identification of computer information system intruders by decision trees and artificial neural networks. In 8th Int. Conference on Intelligent System and Knowledge Engineering ISKE 2013.
3. Guevara, C., Santos, M., Martin, J. A., & López, V. (2014a). Reaching a consensus on access detection by a decision system. In Progress in Informatics and Computing (PIC), 2014 International Conference on (pp. 119-122). IEEE.
4. Guevara, C., Santos, M., & López, V. (2014b). Training strategy to improve the efficiency of an intelligent detection system. In Decision Making and Soft Computing: Proceedings of the 11th International FLINS Conference (pp. 544-549).
5. Guevara, C., Santos, M., & López, V. (2014c). Data leakage detection algorithm based on sequences of activities. In Proceedings of the 17th International Symposium Research in Attacks, Intrusions and Defenses RAID (Vol. 8688, pp. 477-478).
6. Guevara, C. B. , Santos, M., & López, V. L. (2015a). Sistema de Detección de Intrusos aplicando Selección Negativa en Perfiles de Usuario. In Actas de las primeras Jornadas Nacionales de Investigación en Ciberseguridad: León, 14, 15, 16 de septiembre de 2015: I JNIC2015 (pp. 8-13). Servicio de Publicaciones.
7. Guevara, C. B. G., Santos, M., & López, V. (2015b). Sistema Inmune Artificial para la Detección de Comportamientos Anómalos de Usuarios en Sistemas Informáticos.
8. Guevara, C. B., Santos, M., & López, M. V. (2016a). Negative selection and knuth morris pratt algorithm for anomaly detection. IEEE Latin America Transactions, 14(3), 1473-1479.
9. Guevara, C., Santos, M., & López, V. (2016b). Intrusion Detection with Neural Networks Based on Knowledge Extraction by Decision Tree. In International Conference on European Transnational Education (pp. 508-517). Springer International Publishing.
10. Guevara, C., Santos, M., & López, V. (2017). Data Leakage Detection Algorithm based on Task Sequences and Probabilities. Knowledge-Based Systems.

# Método para la Detección de Intrusos basado en la Sinergia de Técnicas de Inteligencia Artificial

César Guevara, Matilde Santos y José Antonio Martín H.

Facultad de Informática, Universidad Complutense de Madrid, C/ Profesor García Santesmases s/n 28040-Madrid, España. Teléfono: +34 91 394 76 20; Fax: +34 91 394 7527

cesar.guevara@gmail.com, msantos@dacya.ucm.es, jamartinh@fdi.ucm.es

**Abstract.** Las técnicas de clasificación basadas en Inteligencia Artificial son herramientas computacionales que han sido aplicadas en la detección de intrusos (IDS), pues resuelven de manera eficiente muchos de los problemas en el área de la seguridad informática. La detección de intrusos conlleva utilizar gran cantidad de información para una adecuada aproximación a la identificación de intrusos, por eso es necesario emplear técnicas basadas en el conocimiento. En este trabajo se identifica el comportamiento de los usuarios autorizados y de los intrusos mediante la sinergia de varias de las técnicas de IA: redes neuronales, máquinas de vectores soporte y árboles de decisión. El sistema desarrollado permite la aplicación de esas técnicas de clasificación y su combinación para obtener una mayor eficiencia y fiabilidad en la detección de usuarios en las redes o sistemas informáticos. Para probar este modelo se ha diseñado un escenario que sirve para demostrar la validez de la propuesta y para acreditar líneas futuras de trabajo.

**Keywords:** Inteligencia Artificial, Detección Intrusos, Redes Neuronales, Arboles de Decisión, Máquinas de Vectores Soporte

## 1 Introducción

En la actualidad los Sistemas de Detección de Intrusos (IDS) son de gran relevancia para la seguridad en las redes y sistemas informáticos ya que cada vez y con mayor frecuencia aparecen nuevas formas de ataques, algunos muy complejos [1].

Los IDS en su mayoría analizan el tráfico de la red e impiden cualquier intrusión previa al ingreso de un usuario al sistema de información. El análisis realizado por el IDS en general es a bajo nivel, generando alertas aisladas y manejando una inmensa cantidad de información. Otros tipos de IDS realizan un filtrado de anomalías de la información recopilada de los equipos host, así como también de las bases de datos de los sistemas con los que interactúan [2].

El campo de detección de intrusos sigue siendo un frente abierto a la investigación para desarrollar metodologías dinámicas que puedan adaptarse a la evolución de

ataques informáticos, cada vez más sofisticados y complejos. Las estrategias que utilizan los IDS se clasifican en dos grupos: detección de uso indebido y detección de anomalías. La metodología de Detección de Intrusos Basada en Anomalías, la cual es utilizada en este trabajo, es extensa y compleja, y ha permitido obtener resultados favorables pero no enteramente adaptables a las necesidades actuales de la tecnología.

La detección de uso indebido de un sistema informático utiliza el conocimiento de las secuencias de actividades que constituyen un ataque, las cuales están almacenadas en una base de datos. La información recopilada es comparada con los patrones de ataque almacenados y, si coinciden, originan una alarma. Esta estrategia es una de las más utilizadas de forma comercial [3]. Sus ventajas son la correspondencia con un patrón de intrusión de base de datos ya almacenado (rapidez), y que el número de falsos positivos sea muy bajo (fiabilidad y precisión). Sin embargo algunas de sus desventajas son, por ejemplo, la incapacidad de detectar nuevos ataques al no ser completamente dinámico, y además el requisito fundamental de mantener actualizada de forma continua la base de datos de los patrones.

Por otro lado, la detección de anomalías utiliza la información de un comportamiento normal del usuario y todo lo que sea diferente de aquella conducta es identificado como un intruso. Sus desventajas son que genera una gran cantidad de falsos positivos y que el comportamiento normal de los usuarios es bastante difícil de modelar, por la necesidad de contar con un periodo amplio de recopilación de información para el aprendizaje del comportamiento [4].

Teniendo en cuenta estas premisas, nuestra propuesta es desarrollar un método dinámico para la detección de intrusos mediante el análisis de anomalías en el tráfico de la red. Como un primer paso para alcanzar el objetivo en esta investigación se utilizan de manera conjunta varias estrategias de clasificación que provienen de la Inteligencia Artificial, en concreto los árboles de decisión, las máquinas de vectores de soporte y las redes neuronales. Se puede probar así que la sinergia de varias técnicas hace más eficiente y fiable el sistema de detección.

Para esta propuesta se han evaluado diferentes metodologías de clasificación con la misma base de datos, para luego aplicar una sinergia de aquellas técnicas que dan mejores resultados y obtener así un resultado mejor en la detección de intrusos que utilizándolas por separado. La metodología propuesta debería ser capaz de procesar la información en tiempo real, filtrando anomalías que pudiesen estar originadas por intentos de intrusión. Debe además generar alertas en caso de encontrar anomalías y almacenarlas de forma dinámica para ser utilizadas en el futuro como aprendizaje para la detección de intrusiones.

La base de datos utilizada es un benchmark en la literatura de seguridad [9], con el fin de probar la validez de esta propuesta para en un futuro aplicarla a bases de datos reales con mayor complejidad.

El artículo está estructurado de la siguiente manera: en la siguiente sección se presenta la evaluación de las técnicas de clasificación que van a ser utilizadas en el estudio; posteriormente se detalla el diseño y la descripción del método propuesto que combina varias de ellas. En la siguiente sección se presentan y se discuten los resultados obtenidos. Y finalmente se comentan las conclusiones con la información relevante del estudio y líneas futuras de investigación.

## 2 Evaluación de Técnicas de Clasificación

Para el desarrollo de este método de detección de intrusos se ha realizado un análisis previo de un determinado número de técnicas de clasificación, para seleccionar las más adecuadas para nuestro estudio. Las técnicas que se han aplicado son las siguientes:

- Red Neuronal Multicapa [5]
- Arboles de Decisión [6]
- Máquinas de Vectores de Soporte [7] [14]
- Red Neuronal de Hopfield [5]
- Redes de Bayes [5]

Para realizar la evaluación de las técnicas de clasificación se ha empleado el Dataset NSL-KDD [8, 9], el cual contiene información de patrones de comportamiento de usuarios en redes informáticas, tanto intrusos como autorizados. El Dataset de NSL-KDD contiene 40 atributos o variables que proporcionan distinto tipo de información sobre los accesos (protocolos, tiempos, tipos de acceso, etc.). Para este estudio se emplearon 25.192 registros para el entrenamiento (el 20% de la base de datos) y 17.102 registros para las pruebas (el 20% del dataset), como se sugiere en [9].

Todos los atributos que incluye el dataset, que describe el comportamiento completo del usuario, son significativos por lo que no es recomendable dividirlo en pequeños grupos de datos ya que se corre el riesgo de perder información relevante para la clasificación.

Para valorar el desempeño de las técnicas de clasificación se realizó una validación cruzada, sirviéndose de la técnica de K-Fold [10], utilizando diez particiones de los datos (k =10) para cada una de las técnicas utilizadas [11]. Estas pruebas permitieron conocer el porcentaje de casos clasificados correctamente y la precisión del clasificador para cada una de las cinco técnicas de clasificación (Figura 1).

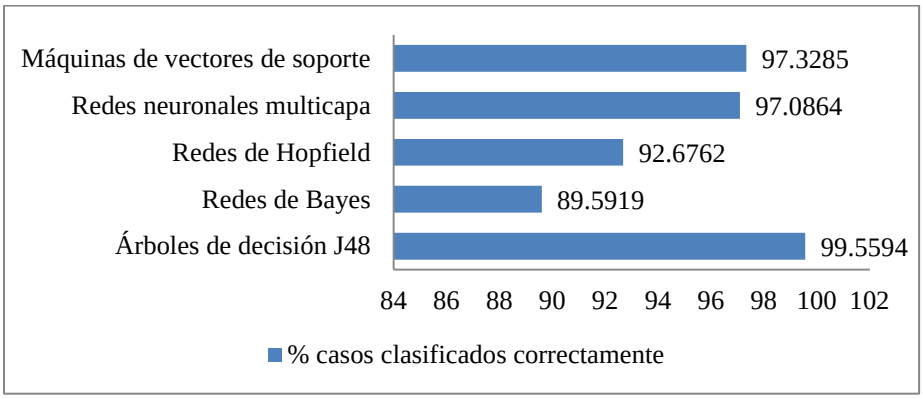



**Fig. 1.** Porcentaje de casos clasificados correctamente con cada técnica

En la tabla 1 se muestran varios indicadores que facilitan la selección de las técnicas más adecuadas y eficientes para nuestro estudio, así como el porcentaje de casos (registros) clasificados correctamente. Además presenta los indicadores de los comportamientos intrusivo y normal. El error medio absoluto del clasificador permite conocer la diferencia entre el valor de la medida y el valor tomado como exacto del fallo del clasificador. La precisión del clasificador permite conocer la eficiencia del clasificador. La matriz de error muestra la información de las pruebas realizadas a cada uno de los clasificadores, proporcionando información de los verdaderos positivos y los falsos positivos.

| | | | | MATRIZ DE ERROR | | | |
|---|---|---|---|---|---|---|---|
| **Nombre técnica de clasificación** | **% casos clasificados correctamente** | **Error medio absoluto** | **Precisión del clasificador** | **Verdaderos positivos Normal** | **Verdaderos positivos Intrusos** | **Falsos Positivos Normal** | **Falsos Positivos Intrusos** |
| Árboles de decisión J48 | 99.5594 | 0.006 | 0.996 | 13389 (53.14%) | 11692 (46.42%) | 51 (0.20%) | 60 (0.24%) |
| Redes de Bayes | 89.5919 | 0.103 | 0.896 | 12272 (48.71%) | 10298 (40.88) | 1445 (5.74%) | 1177 (4.67%) |
| Redes de Hopfield | 92.6762 | 0.123 | 0.927 | 12694 (50.39%) | 10653 (42.29%) | 1090 (4.33%) | 755 (2.99%) |
| Redes neuronales multicapa | 97.0864 | 0.03 | 0.971 | 13064 (51.86%) | 11394 (45.23%) | 349 (1.38%) | 385 (1.53%) |
| Máquinas de vectores de soporte | 97.3285 | 0.027 | 0.974 | 13261 (52.64%) | 11258 (44.69%) | 485 (1.92%) | 188 (0.75%) |

**Tabla 1.** Comparación de técnicas de clasificación con el Dataset NSL-KDD

El número de aciertos es elevado en cada una de las técnicas ya que se ha utilizado un número muy alto de ejemplos para el entrenamiento.

Después del resultado obtenido de la evaluación se observa que las tres técnicas de clasificación que han alcanzado un mejor resultado son: Redes Neuronales Multicapa, Máquinas de Vectores de Soporte y Árboles de Decisión.

## 3 Modelo Propuesto

En este artículo se propone un modelo para la detección de intrusos como el que se muestra en la figura 2. Resulta de la sinergia de las tres técnicas de clasificación anteriormente citadas que daban mejores resultados: Redes Neuronales, Máquinas de Vectores de Soporte y Árboles de Decisión.

La utilización de estas técnicas es la base fundamental para la disminución de detecciones fallidas y para mejorar el rendimiento del filtrado de intrusos. Con este esquema se incrementará porcentualmente la detección de intrusos y disminuirá la tasa de falsos positivos de manera significativa, lo que beneficiará la seguridad de los sistemas.

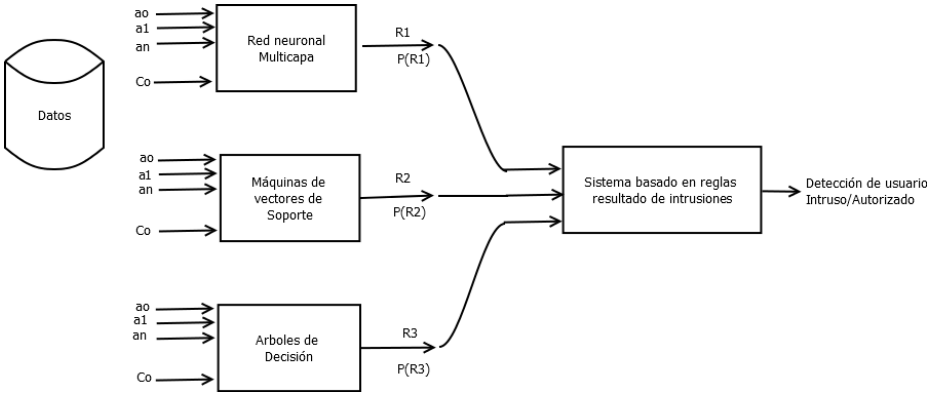



**Fig. 2.** Diagrama de la sinergia de Técnicas de Clasificación para Detección de Intrusiones

El modelo se basa en un proceso de uso simultáneo de la información con las tres técnicas de clasificación. Cada una de las partes de este proceso realiza tareas distintas con la información recopilada en la tarea anterior, lo que es importante para obtener un resultado eficiente. El flujo de operaciones se muestra en la figura 3.

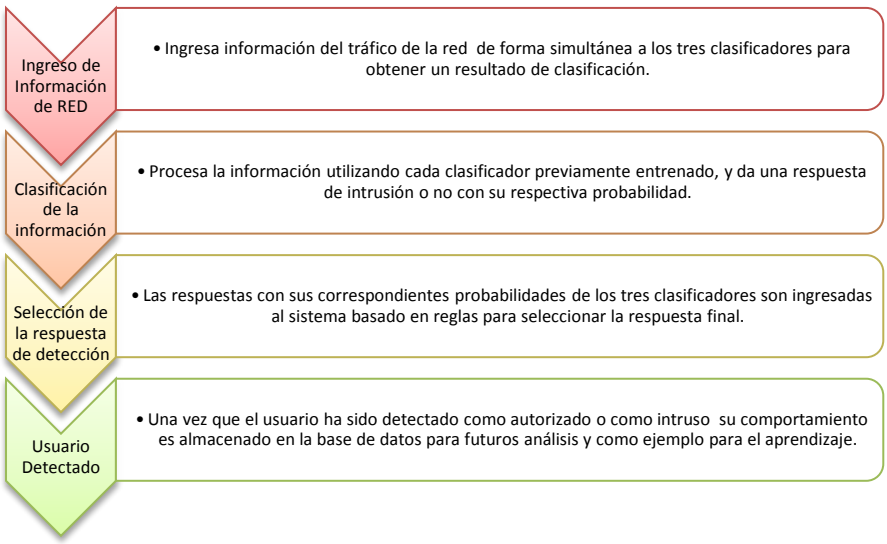



**Fig. 3.** Flujo de la información del modelo de sinergia de clasificadores para detección de intrusos.

A continuación se detallan cada una de las fases del proceso de detección de intrusos del modelo propuesto.

### 3.1 Configuración de técnicas de clasificación

Previamente a la aplicación de las técnicas de clasificación es necesario describir la configuración de cada una de ellas.

**Red Neuronal Multicapa:** se ha construido una red neuronal con 40 neuronas de entrada, tantas como atributos de los usuarios de la red; 2 capas ocultas, la primera con 20 neuronas y la segunda con 16; la salida de esta red neuronal es una sola neurona que da como resultado una de las dos clases posibles, accesos permitido o acceso intrusivo.

**Máquinas de vectores de soporte (SVM):** se ha utilizado como kernel la función RBF (Radial Basis Function), entrenando las SVM con el algoritmo SMO (Sequential Minimal Optimazer). El entrenamiento fue realizado con la librería LIBSVM [14] utilizando los 40 atributos del trafico de la red.

**Árboles de decisión:** el árbol de decisión utiliza los 40 atributos del dataset del tráfico de red y aplica el algoritmo J48 para la clasificación.

Todos los clasificadores se entrenaron con 25.912 registros para el entrenamiento y 17102 registros para pruebas.

### 3.2 Descripción del modelo

**Ingreso de información de la red:** se ingresa de manera simultánea la información de tráfico de la red a los tres clasificadores, con sus cuarenta atributos por registro, normalizados [12].

**Clasificación de la información:** se clasifica la información de la red dando como salida si un usuario es un intruso o está autorizado, con un porcentaje de fiabilidad y precisión.

**Fase de Selección de resultados mediante sistema basado en reglas:** los resultados de los tres clasificadores anteriores se introducen en el sistema de reglas como entradas. Estas entradas se denominan R1, R2 y R3, con sus respectivas probabilidades P(R1), P(R2) y P(R3). Luego el sistema de decisión basado en reglas recibe estos seis valores, y proporciona el resultado final de la clasificación. La figura 4 presenta las 8 reglas que conforman este sistema. La salida será un valor binario, "1" si es intruso y "0" si está autorizado.

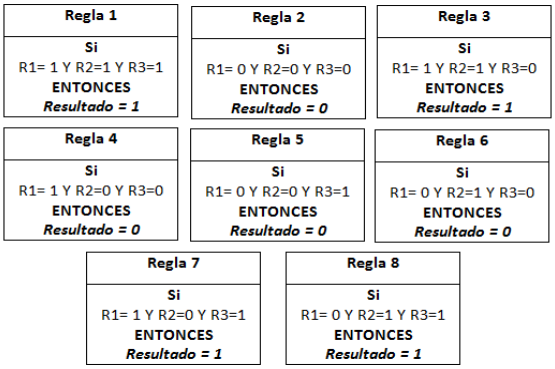



**Fig. 4.** Reglas para la identificación final del usuario.

Después de obtener el resultado de la detección es necesario conocer la probabilidad de la respuesta. Para ello que se realiza un cálculo matemático utilizando la media de los datos que cumplen las reglas anteriormente descritas. Para las reglas 3, 4, 5, 6, 7 y 8 se calcula la media de las dos probabilidades correspondientes a los resultados que verifican las reglas anteriormente descritas, ya que dos de los clasificadores proporcionan el mismo resultado y el tercero uno distinto (1).

$$\mathbf{Med= P(Ra)+P(Rb)/2} \quad (1)$$

En el caso de las reglas 1 y 2, donde la salida de los tres clasificadores es la misma, la media de probabilidades es:

$$\mathbf{Med=P(R1)+P(R2)+P(R3)/3} \quad (2)$$

Es imprescindible utilizar la media de las probabilidades de los resultados de las distintas técnicas de clasificación para obtener una probabilidad conjunta. Además, esta probabilidad resultante será registrada dentro del histórico del comportamiento del usuario, lo que permite que el modelo sea dinámico al contar con nueva información.

Para validar este sistema de reglas se utilizaron registros de la base de datos [9]. En la tabla 2 se presenta una pequeña muestra de cómo se presentan los datos para las pruebas.

| R1 | P(R1) | R2 | P(R2) | R3 | P(R3) | Clasificador |
|---|---|---|---|---|---|---|
| 0 | 0.99995 | 0 | 0.99968 | 0 | 0.99945 | 0 |
| 0 | 0.99995 | 0 | 0.99968 | 0 | 0.99032 | 0 |
| 1 | 0.99924 | 1 | 0.99055 | 1 | 0.99032 | 1 |
| 1 | 0.99924 | 1 | 0.99055 | 1 | 0.99945 | 1 |

**Tabla 2.** Muestra de información para la decisión final de la respuesta.

**Usuario Detectado:** finalmente se obtiene un resultado (Intruso, 1, o Autorizado, 0), el cual se almacena en la base de datos para que pueda utilizado para futuros análisis de patrones de comportamiento dentro de la red.

## 4 Discusión de Resultados

En la tabla 3 se presentan los resultados para los indicadores de eficiencia utilizados y los falsos positivos y falsos negativos obtenidos con la técnica propuesta.

| % casos clasificados correctamente | % Error medio absoluto | % Precisión del clasificador | % Verdaderos positivos autorizados | % Verdaderos positivos Intrusos | % Falsos Positivos autorizados | % Falsos Positivos Intrusos |
|---|---|---|---|---|---|---|
| 99.9% | 0.1 | 99.9 | 99.9% (2997) | 99.9% (2997) | 1% (3) | 1% (3) |

**Tabla 3.** Indicadores utilizados para probar la propuesta de sinergia de Técnicas de Clasificación para Detección de Intrusiones.

Como ya se comentó, la precisión de las técnicas de clasificación para esta base de datos es muy elevada, con un valor de 99.9% (**Precisión del Clasificador**) en todos los casos de clasificación. En la aplicación de la sinergia de estrategias inteligentes se obtiene un resultado más elevado que al utilizar las técnicas de manera individual. De hecho se alcanza el 99.9% de aciertos en verdaderos positivos y verdaderos negativos (2.997 casos); por lo tanto hay sólo un 1% de falsos positivos y de falsos negativos, lo que significa que el modelo realiza la clasificación de forma muy eficiente.

La información con la que se ha realizado el estudio es muy completa y extensa, se ha contado con muchos ejemplos para entrenar los sistemas de clasificación, y por eso los resultados obtenidos con las técnicas seleccionadas ya eran buenos.

Lo que se ha probado en este trabajo es cómo la fusión de varias de estas técnicas mejora el resultado, con el fin de poder aplicar esta sinergia a problemas más complejos donde no haya tanta información o ésta esté incompleta [13].

## 5 Conclusiones y Trabajos Futuros

En este estudio se propuso un método para la detección de intrusos mediante la sinergia de técnicas de clasificación que provienen de la Inteligencia Artificial.

Se ha evaluado la aplicación de varias técnicas de clasificación a un problema bien definido, y se ha probado como la combinación de varias de ellas mejora el porcentaje de aciertos. Se ha utilizado como base de conocimientos el dataset NSL-KDD.

Las líneas de trabajo futuro son principalmente dos. Por un lado, habría que aplicar esta estrategia de fusionar técnicas a problemas más complejos, con menos ejemplos de entrenamiento, o señales ruidosas, con información incompleta, etc. Por otro, estamos trabajando para realizar pruebas con información real de usuarios que acceden a bases de datos de información y realizan una serie de operaciones sobre los registros.

## Referencias

1. Debar, H., Viinikka, J.: Introduction to Intrusion Detection and Security Information Management. In: Foundations of Security Analysis and Design III FOSAD 2005. LNCS, 3655, pp. 207--236. Springer (2005).
2. Zurutuza, U.: Revisión del estado actual de la investigación en el uso de data mining para la detección de intrusiones. Escuela Politécnica Superior de Mondragón, Departamento. Informática 2005.
3. Bace, R.G, Mell, P.. Intrusion detection systems. Gaithersburg, MD: U.S. Dept. of Commerce, Technology Administration, National Institute of Standards and Technology, 2001.
4. Wang, W., Gombault, S.: Distance measures for anomaly intrusion detection. In: Security and Management (2007) 17–23
5. Haykin, S. Neural networks: a comprehensive foundation. New York: Macmillan, 2004.
6. Quinlan, J.R.. Induction of decision trees. Broadway, N.S.W., Australia: New South Wales Institute of Technology, School of Computing Sciences, 1986.
7. Abe, S. Support vector machines for pattern classification. London: Springer, 2005.
8. MIT: Mit Lincoln laboratory-Darpa intrusion detection evaluation (retrieved march 2009). http://www.ll.mit.edu/mission/communications/ist/corpora/ideval/index.html (1999)

9. Nsl-kdd data set for network-based intrusion detection systems." Available on: http://nsl.cs.unb.ca/NSL-KDD/, March 2009.
10. Moore, A.W., Lee, M.S. 1993. Efficient Algorithms for Minimizing Cross Validation Error. In Machine Learning: Proceedings of the Eleventh International Conference, Morgan Kaufmann.
11. Esposito, M., Mazzariello, C., Oliviero, F., Romano S.P., Sansone, C. "Evaluating pattern recognition techniques in intrusion detection systems." in Proceedings of the 5th International Workshop on Pattern Recognition in Information Systems (PRIS) 2005, May 2005, pp. 144 -153.
12. Wang W., Zhang X., Gombault S., Knapskog S.J., Attribute Normalization in Network Intrusion Detection, IEEE, 2009 10th International Symposium on Pervasive Systems, Algorithms, and Networks. 978-0-7695-3908-9/09, pp-448-453.
13. Lee, W., Stolfo, S.J. and Mok, K. (1999). Data Mining in work flow environments: Experiments in intrusion detection. In Proceedings of the 1999 Conference on Knowledge Discovery and Data Mining.
14. Chang C.-C., Lin C.-J., LIBSVM: a library for support vector machines. ACM Transactions on Intelligent Systems and Technology, 2-27, 2011. Available at http://www.csie.ntu.edu.tw/~cjlin/libsvm

# Identification of Computer Information System Intruders by Decision Trees and Artificial Neural Networks

**César Byron Guevara, Matilde Santos, José Antonio Martín**[1]

**Abstract.** Detection and control of intruders or unauthorized access to computer systems has always been important when dealing with information systems where security, integrity and privacy are key issues. Although computer devices are more sophisticated and efficient, there is still the necessity of establishing safety procedures to avoid illegitimate accesses. The purpose of this paper is to show how different intelligent techniques can be used to identify users accessing critical information and to check whether or not access is allowed. Advanced and intelligent analysis and data mining techniques such as decision trees and artificial neural networks have been applied to obtain patterns of users' behavior. Dynamic user profiles are obtained. The main contribution of this work is to show effective solutions for the detection of intruders in computer information systems.

**Keywords:** Intrusion, pattern recognition, behavioural profile, security, intelligent techniques, decision tree, neural networks.

## 1. Introduction

Nowadays, computer systems form a large information network. Users who have access to this network generate a lot of information about their behaviour without realizing it. In fact, they are continuously providing information, for example, their preferences, location, ways of acting, etc. Such personal information describes a pattern that could be use as a signature or a unique identifier of each one and thus it provides an effective way to detect intruders.

This personal information is systematically recorded in different databases. However, its analysis is quite complex due to its huge size. Intelligent data mining [4] strategies can be applied for the analysis of this information in a faster and

[1] César Guevara, Matilde Santos, José Antonio Martín (✉)
Computer Science Faculty, University Complutense of Madrid
email: cesargue@ucm.es, msantos@dacya.ucm.es, jamartinh@fdi.ucm.es

more efficient way, which will produce a behavioural pattern and will allow us to generate a user profile.

The public sector of the Republic of Ecuador, as other governments, has information systems of utmost importance, so it is necessary to have an adequate control of the users who have access to such information. The efforts that have been made to improve the confidentiality of this information has resulted in sophisticated security systems, making user access more complicated and becoming more inconvenience. Despite that, intrusion of unauthorized agents and leak of classified information are still reported. To solve this problem we propose to obtain behaviour patterns of the authorized users in an automatic way in order to control the access to those systems. The information that has been used in this study comes from activities carried out by public servants who have access to the governmental databases.

In this work we have applied decision trees [2, 5, 7] and artificial neural networks (ANN) [1,8] for the identification of these patterns has allows us to obtain dynamic profiles of information systems users, and thereby to establish a way of detecting intrusions or malicious agents in the computer systems. This modelling demands a previous pre-processing and analysis phases of the available information that with the automatic mechanisms of classification are the main contributions of this research work.

The system has been tested with real data that have been provided by an official institution of the Ecuador government.

## 2. Pre-processing and Data analysis

The information has been taken from the database of a public institution of the Republic of Ecuador, which has multiple users that interact with the system and perform a variety of tasks within that database.

The first step involves the selection of the most relevant variables and before the information that is stored in different databases must be integrated and gathered into one. This will make easier the tasks of sorting, cleaning and retrieving the information that will be used for the classification. The subsequent analysis and the success of the detection strongly depend on this initial phase [4].

Several databases have been grouped into one call “tfmdata”. This database is multidimensional and every dimension describes a set of relevant facts such as regarding the access of each user to that information system. The result is what is called OLAP (Online Analytical Processing). OLAP is used to focus on important attributes, identify exceptions, o finding iterations.

## *2.1. Analysis of variables*

The variables should reflect the users' behavior when they are using the system and describe their activities in a more detailed manner. In the database there are already several attributes that allow us the definition a behavioral pattern of the user such as day of the week, device used to the access, etc.

For this analysis, a sample was taken for each of the activities and tasks listed above. The following figures show these variables for a user during a period of three months on a specific database. Figure 1 shows the activity of the user, that is, number of times he has performed each of the three possible operations, Update, Delete, Insert. Figure 2 a, b, c presents the times he has carried out every operation along the two months [8].

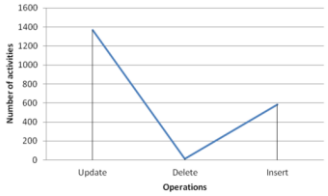



**Fig. 1** Operation performed by the user


**Fig. 2** Operation performed by the user in the DB; a)Update, b)Delete, c)Insert

The same analysis has been carried out with the other variables as follows. Table: four databases have been named as 1, 2, 3 and 4 to keep the confidentiality although they represent official information. Times a user has accessed to a specific database each day of those two months have been recorded. The number of times the user performs an operation on a database each day of the week, the hour of access and time he spends doing that operation, number of accesses, IP address of the workstation from which operations are made, etc.

Once this analysis has been made some conclusions can be drawn, at least in a general way, on the actions the user performs more often, database more used, etc. This allows us to find the most significant variables, those which better identify each user.

## *2.2. Multidimensional model of the data repository*

Four dimensions or variables have been taken into account in the OLAP model: time, user, activity, and workstation. Each of them has been generated in the data repository. Different tables are associated to everyone. User can be group or single user, affiliation, department, ranking, position. Activities entry is connected to operation and database. Workstation is just the IP and the device. All of these variables are related. The fact table has been created under the entry "*user_access*" which is the main one. The OLAP generated is shown in Figure 3.

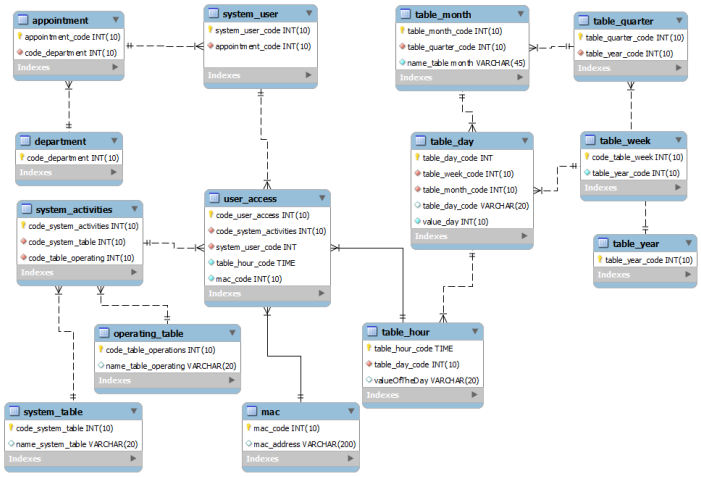



**Fig. 3** Diagram of the variables of the data repository OLAP

The next step is to dump the real data into the OLAP repository, after debugging and processing the information to ensure it is correct. In fact, several inconsistencies were detected and solved. The information was decomposed into the variables of the only database to integrate and unify it. This allowed us to save space as some attributes had a very large format but little information.

## *2.3. Exploratory data analysis*

Once all the data are integrated in the repository, the following step is to summarize the main characteristics of the attributes. It may also be interesting to obtain some features such as the maximum, minimum and average values and to represent them graphically. To perform this analysis in an easy, efficient and automated way the tool WEKA was used. The main result will be what it is called mining view and it will be very useful to perform the visual data mining and therefore to get an idea of the possible patterns, the relevant data, etc.

This tool scans the data analysis algorithms and gives exact and efficient information for each variable. We have worked with 18 different users, each one represented by a different color. As can be seen in Figure 4, the values of the different attributes are shown for each variable. Combining all the information of every variable for each user it will be possible to obtain a pattern of the behavior.

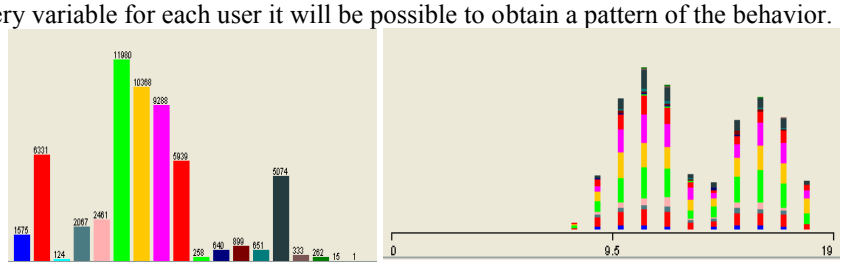



**Fig. 4** a) Variable “entering information”, b) working time, for each user

Once the data has been gathered, cleaned and integrated, an exploratory analysis must be done to extract the maximum possible information. It should identify the information that will be meaningful to obtain patterns. The following questions have been used as guideline to define a decision making scenario:

**What do you want from the information collected?**

Get a behavioral pattern of each user of a information system in order to detect unauthorized access.

**What rules are to be defined to model the data entry?**

The information is collected for each system. That is, user identification and password, but they are different for each system and are not linked together.

**Is there any documentation for collecting this information?**

There is no documentation to support this task. The information comes from various databases and different systems, so we must define a pattern of each user from the available data.

At this stage, the objective is, as already noted, to get a view of information that facilitates the data exploration and subsequently the application of pattern recognition techniques, in order to detect patterns, trends, anomalies, and a better understanding of the data. Some data mining techniques such as generalization, aggregation and filtering of the information are used.

To show this exploratory analysis, although we have worked with 18 users, we will use just three to simplify the study. We have chosen users #6, #8 and #11. For instance, Figure 5 shows how the behavior of these three users is quite different regarding variable time (Figure 5a) and variable operation they perform (2, 3 or 4, meaning introducing data, deleting or updating) (Figure 5b). Other variables that have been taken into account are day and table. These have been included in the final data base that has as inputs day, time, operation, table and user. For example, at a specific hour of a week day, let say 5 (Friday), each of the user is doing a different operation on a different table.

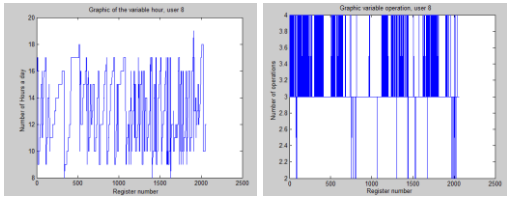



**Fig. 5** a) Variable “time”, b) Variable “operation”, for each user

This has been done for all the variables and we could learn that variable year does not give relevant information in order to describe a user, as we are working with three years, 2010, 2011 and 2012 and on average the work load is mainly the same every year for all the users. The graphs of this variable for the three users are quite similar. The same happened with the variables week and month. So it was decided that the variables year, month and week will not be used as they do not give relevant information in order to obtain a behavioral pattern for the users.

## 3. Application of intelligent techniques to model the users

Knowledge extraction from the processed data is the basis to define an only behavioral pattern of each user. This user profile has to unique, dynamic and close to reality. For that reason it is important to work with techniques that keep the models understandable such as the decision trees.[6]

Besides, when applying the classification, we are not only determining if the user is an intruder but also the degree of certainty of that prediction.We have combined two soft computing techniques, decision trees and neural networks. In the decision tree the probability of each event (leave) has been added. The neural network carries out the final step, the classification [3].

The merging the two techniques that allows a more efficient detection of frauds, failures or intrusions into the systems is one of the main contributions of this paper. The application has been developed using the software Matlab and the corresponding toolboxes.

### *3.1 Decision tree implementation*

Decision trees allow to express rules that relate multiple attributes and organize the available information in a hierarchical, orderly way. Four decision trees have been generated for periods of one, three, six and twelve months for each user to find which one was more useful for the discrimination. Figure 6 shows the decision tree for a user in a period of one year.

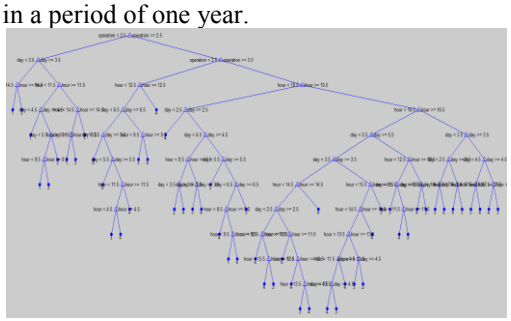


**Fig. 6** Decision tree, user #6, a year

Comparing the trees generated for the different users we realized that selecting the period of three months was enough to obtain a quite different structure and at the same time to keep it simple. The probability associated to each branch and leaf of the tree data was calculated. It was defined by the following formula,

$$P(A)=val_A/N, \qquad P(B)=val_B/N$$

$$P(T)= P(A_1)*P(A_2)*P(A_3).....*P(A_n)$$

Where N the total number of cases, *valJ* is the number of events in branch i, P(i) the probability of the event in the i node, P(T) the final probability at n, and n the total number of nodes at that point of the tree. As an example, one of the two main branches will be shown (Figure 7). Each node or joint has its probability and percentage assigned. So it indicates the likelihood of an action of an authorized user.

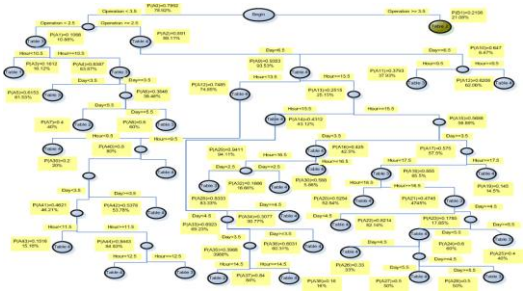


**Fig. 7** Right branch of the decision tree, user #6, three months

For instance, the following formulas have been applied to obtain the corresponding probabilities. User #6 performs operation 3 on table 1 at 9:15h on day 1 (Monday).

$$P1(T)=P(A0)*P(A1)*P(A3)$$

$$P1(T)=(0.7892)*(0.1088)*(0.1612)=0.0138\ (1.38\%)$$

User #6 performs operation 2 on table 3 at 16:00h on day 4 (Thursday).

$$P2(T)=P(A0)*P(A1)*P(A4)*P(A6)*P(A8)$$

$$P2(T)=(0.7892)*(0.1088)*(0.8387)*(0.3646)*(0.6)=0.0157\ (1.575\ \%)$$

User #6 performs operation 4 on table 4 at 12:20h on day 7 (Sunday).

$$P3(T)=P(B1)*P(B3)*P(B20)*P(B28)*P(B26)$$

$$P3(T)=(0.2108)*(0.3599)*(0.573)*(0.8431)*(0.7674)=0.0281\ (2.81\%)$$

In the three previous cases, user 6 performed different activities and each one has its own probability. The most probable is the third one. But it depends on the time, day, etc. That means that if someone logs the system and performs that task at that time, he will be an authorized user and not an intruder. If on the other hand somebody logs the system at that day and time and performs any other activity with probability smaller than that one, it would mean that if could be an intruder.

This representation displays the behaviors of the users in detail, but it does not describe the behavior that the user does not normally do. That is, if the behavior of the user does not follow any of those branches, how can it be detected if it is an intrusion or an authorized user doing something different from usual? So it needs to be complemented by another technique such as neural networks in order to give a final classification.

## *3.2. Neural Networks for final detection of intrusions*

With the aim to identify if a user is authorized or is an intruder, a neural network has been applied. It has been designed using Matlab.

After trying several configurations, the final neural network is as follows. There are four inputs: day, time, operation, and table. These are the most significant variables and the ones that allow us to discriminate between users, as explained in section 2. The output is a single neuron that represents the user, that is, a number between 0 and 18 (0 stands for intruder and the rest of the numbers identify the user). There is a hidden layer with 20 neurons.

Samples have been generated from real data (authorized users) and others that have created to simulate the behavior of intruders. The available samples have been then divided into training and test sets. Learning (performance) is measured in terms of the mean square error. The network is trained until the error is smaller than 0.001.

Once the neural network has learnt to classify the user that is working on the system, in order to validate it we have tested it with the test samples. Then we have represented the results in what is called a confusion matrix. The confusion matrix shows the percentages of correct and incorrect classifications. Correct clas-

sifications are the green squares of the diagonal. Incorrect classifications are the red squares.

In the confusion matrix for users 1 and 2. The users identified by the classifier are in rows, and the right users are represented in columns. That is shows the number of users that our classifier identified as user 1 and they were user 1 (221 out of 222 samples). That is, one of the inputs was wrongly classified as user 2 when it was user 1.

## 4. Simulation tool for intrusions detection

A simulation tool that implements these techniques for intruders' detection was developed. The input is the set of activities of a user that logs the system, and the output is the identification of the user or the probability of being an intrusion. The simulation tool gives three different classifications in accordance with the inputs.

- *Authorized access*: the user and the information of his behavior match, so it seems to be an authorized user and has been rightly identified.
- *Unauthorized access*: the information collected from the behavior of the user is different from expected for the user that has logged the system. A message of Intruder is displayed.
- *Unrecognized pattern*: an X is displayed when the system is not able to identify the behavioral pattern with any of the user, but it could still be an authorized user. In this case the result in a warning of possible intrusion.

An interface has been designed to insert the data. For example, a user that has just logged the system has introduced his identification code and password that corresponds to user 8. The system automatically detects day, time, and it records the operation is being performed and the table that has been accessed. In this case, Tuesday, 13.00h, operation 4, Table 3. The corresponding variables are displayed on the left bottom window (day, blue line; time, green line; operation, blue; table, red). The corresponding decision tree is generated. This profile will allow the neural network to identify the user pattern. In this case, according to the action performed by the user, it has been identified as the correct one, that is, the authorized user 8. In this case, the second tree corresponds to the pattern of user 13. As they do not match, it is an intruder (Figure 8).

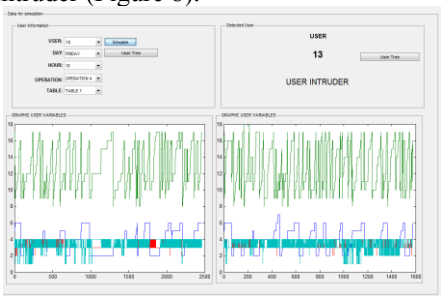



**Fig. 8** Intrusion detection

Finally, the application can also detect an access of a user who does not present an expected behavior, so it could be an intrusion but it can also be the right user with an out of the ordinary behavior. That means that is performing some actions with a low probability for that user but still that match the flow of actions of the corresponding decision tree. In this case the application will give as result "non-identified user" and the corresponding alert in case it is an intruder.

## 5. Conclusions

Data mining techniques such as decision trees and neural networks have been proved very efficient when applied to pattern recognition and classification. In this paper we have developed a simulation tool that implement both of them to detect unauthorized accesses to information systems.

The decision tress allowed us to obtain a representation of the behavior of the user close to reality, in the sense of understandable. On the other hand, neural networks identify a behavioral pattern based on the description given by the decision trees.

Previously, an exploratory analysis of the data was carried out in order to extract the key knowledge and to identify the significant variables that allow the discrimination of different users' behavior.

This has been applied to real data that come from an official institution of the Republic of Ecuador Government.

## 6. References

1. Anderson, James A., and Joel Davis. An introduction to neural networks. Vol. 1. Cambridge, MA: MIT press, 1995.
2. Brodley, Carla E., and Paul E. Utgoff. "Multivariate decision trees." Machine learning 19.1 (1995): 45-77.
3. Santos, M., and A. J. Cantos. "Classification of plasma signals by genetic algorithms." Fusion Science and Technology 58.2 (2010): 706.
4. Fayyad, Usama M., et al. "Advances in knowledge discovery and data mining." (1996).
5. Quinlan, John Ross. C4. 5: programs for machine learning. Vol. 1. Morgan kaufmann, 1993.
6. Russell, Stuart. Artificial intelligence: A modern approach, 2/E. Pearson Education India, 2003.
7. Murthy, Sreerama K. "Automatic construction of decision trees from data: A multi-disciplinary survey." Data mining and knowledge discovery 2.4 (1998): 345-389.
8. Witten, Ian H., and Eibe Frank. Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann, 2005.

# Reaching a Consensus on Access Detection by a Decision System

César Guevara, Matilde Santos, José Antonio Martín, Victoria López
Dep. Computer Architecture and Automatic Control
Complutense University of Madrid
Madrid, Spain
cesargue@ucm.es

***Abstract*—Classification techniques based on Artificial Intelligence are computational tools that have been applied to detection of intrusions (IDS) with encouraging results. They are able to solve problems related to information security in an efficient way. The intrusion detection implies the use of huge amount of information. For this reason heuristic methodologies have been proposed. In this paper, decision trees, Naive Bayes, and supervised classifying systems UCS, are combined to improve the performance of a classifier. In order to validate the system, a scenario based on real data of the NSL-KDD99 dataset is used.**

***Keywords—artificial intelligence; heuristic methodologies; intrusion detection (IDS); decision trees; supervised classifying system UCS; naive Bayes***

## I. INTRODUCTION

Currently, Intrusion Detection Systems (IDS) are commonplace for network security and computer systems, and more and more frequently new forms of attacks, some considerably complex, appear [1].

The majority of IDS analyzes the traffic of the network and rejects whatever intrusion of a user in the information system. The analysis completed by the IDS takes places generally at a low level, generating isolated alarms and handling an immense quantity of information. Other types of IDS utilize what is called an anomalies filter that is applied to the information of the server and the databases.

The field of intrusion detection continues as an open research line concerning the developing of dynamic methodologies that are able to adapt themselves to the evolution of the computer attacks, each time more sophisticated and complicated. The strategies that IDS utilizes can be classified into two groups: detection of incorrect use, and detection of anomalies. The methodology of the Intrusion Detection Based on Anomalies, the strategy which is discussed in this article, is demanding and complex. Although it has reached good results, it is not entirely adaptable to the needs of the current technologies.

The detection of the incorrect use of a computer system requires the knowledge of the sequence of activities that constitutes an attack, which must have been stored in a database. The stored information is compared with the patterns of previous attacks. If they coincide, an alarm or warning is set off. This is the most commonly used strategy and in fact there are even commercial software that facilities it. Its main advantages lies in the speed, as it is just to find the similarity with the pattern of intrusion (already uploaded) and that the number of false positives is generally low (reliability and precision). However, some disadvantages are the incapacity to detect new attacks in a dynamic way, and furthermore the necessity of being continually updating the patterns´ databases with new cases [2, 3]. To summarize, it is based on the intrusion behaviour and tries to identify this pattern.

On the other hand, the strategy detection of anomalies is based on the information of the normal behavior of a user. Every other different behavior is identified as an intrusion. Therefore it is based on the normal behaviour pattern. Some of the disadvantages are that this method generates a considerable amount of false positives and besides the normal behavior of the users is quite difficult to be modeled, mainly due to the necessity of storing the information and the learning of the users´ behavior [4].

With these premises in mind, our proposal is to develop a dynamic method for the detection of intrusion by means of the analysis of the anomalies in the network traffic. The first step to reach this objective is to apply different classification strategies that come from the Artificial Intelligence field. Intrusion detection has been approached before using data mining techniques, classification in particular [5, 6].

Specifically we have tried decision trees, Naive Bayes, and supervised classification systems. Then, we combine some of these techniques. The main contribution of this paper is that the fusion of some classifications techniques improves the detection as it has been proved in comparison to these strategies and other when applied independently [7].

The different classification methodologies have been evaluated for the same database independently. The best ones have been chosen to be applied in the final classifier. As we will show, the synergy of some of them provides better results than using each technology separately.

The methodology proposed should be able of processing the information in real time, filtering anomalies which could be originated by intruders. Furthermore, it should be able not only to generate alerts in the case of finding anomalies but to store

This work has been partially supported by the Ministry of Higher Education, Science, Technology and Innovation (SENESCYT) of the Government of the Republic of Ecuador.

them in a dynamic way to be used in the future. This way, the system learns dynamically with new cases.

The database utilized is a benchmark in the literature of security [4, 8]. Once the decision system has been proved on it, it will be applied to more complex real databases.

This article is structured as follows. The following section presents an evaluation of some classification technologies that have been analyzed. In section III a new decision making system is designed by the fusion of the ones that have provided better results. Section IV is devoted to the discussion of the results obtained by this classifier. Finally, conclusions and future work end the paper.

## II. Application of Heuristic Methodologies to Intrusion Detection

In order to choose some classification techniques that allow us to design a better classifier, they have been applied to a benchmark database. We have tried some methodologies form different fields of the Artificial Intelligence discipline.

The intelligent technologies that have been applied are the following:

- Neural Networks: Multilayered Networks, Hopfield´s Neuronal Networks, Bayes Networks [9];
- Decision Trees: C4.5, ID3 [10, 11];
- Support Vector Machines (SVM) [12]
- Supervised Classifier System (UCS) [13];
- Naive Bayes (NB) [14].

In the work we have used different types of classification techniques that can be used to take advantage of the strong characteristics of each of these techniques. Neural networks are efficient using multiple numbers of variables. The decision trees are algorithms that are also efficient with a lot of information and they perform the classification using all data available during learning. Moreover, the "supervised classification systems" use rules based on every data for the classification and its main advantage is that it would not rule out any information. The Naive Bayes is a learning algorithm that runs continuously until it obtains the better result in the classification with multiple numeric variables.

These classification techniques have been applied to the Dataset NSL-KDD[1]. It is a real database that contains information of the behavior of computer system users, both intruders and authorized ones. The NSL-KDD Dataset is made of 40 attributes or variables which provide information about the protocol, date, time, entry type, etc. In this work 25,192 registers were used for training (20% of the database NSL-KDD) and 17,102 registers for the test (13.5% of the dataset NSL-KDD), as suggested in [15].

All of the attributes included in the dataset, which describe the user´s behavior, are supposed to give relevant information and should be taken into account to improve the efficiency of a classifier based on them.

[1] NSL-KDD: http://nsl.cs.unb.ca/NSL-KDD/, 2014-03-28

To evaluate the performance of the classification technologies, a cross validation was carried out, making use of the K-fold technique [16]. We use 10 partitions of the data (k = 10) for each of the techniques. These tests gave the percentage of correctly classified cases. The accuracy of the classifier was also computed as the percentage of correctly classified users over the total number of cases for each of the eight classification techniques tested.

These previous results are shown in Figure 1. As it is possible to see, all of them present a high ratio of hits. That is because the number of examples used for the training is quite big, and therefore all the techniques give good results.

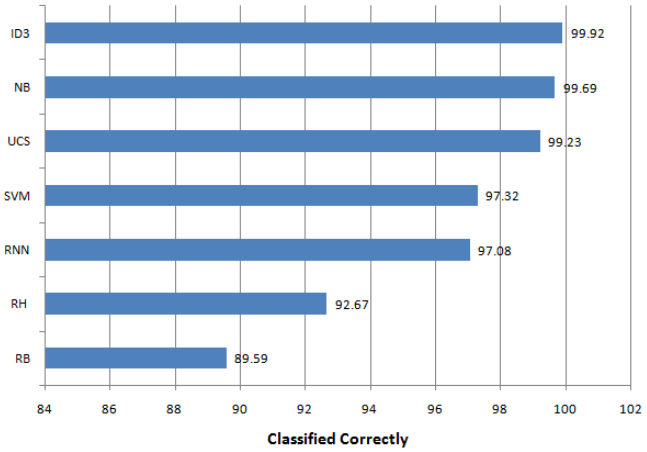



Fig. 1. Percentage of correctly classified user with each technique for the dataset NSL-KDD99

More detailed information is presented in Table I. It shows different indicators of the performance of each technique, that will allow the selection of the most appropriate and efficient technique for the final decision system. In this table, the accuracy of the classification technique is also given, with the EMAE (absolute margin of error of the classifier), defined as the difference between the validity of the measurement and the validity taken exactly as an error.

Furthermore, this matrix of error shows true positives (TP) and false positives (FP) cases obtained by each technique, both for intruders and authorized user. This will help to finally make a decision on the best techniques.

TABLE I. Comparison of Classification Techniques on the Dataset KSL-KDD.

| Algorithm | Classified Correctly | EMAE | Accuracy | TP Genuine | TP Intruder | FP Genuine | FP Intruder |
|---|---|---|---|---|---|---|---|
| C4.5 | 99.96% | 0.04% | 0.9996 | 13444 | 11740 | 0 | 12 |
| ID3 | 99.92% | 0.08% | 0.9992 | 13439 | 11735 | 1 | 17 |
| NB | 99.69% | 0.31% | 0.9969 | 13408 | 11708 | 32 | 44 |
| UCS | 99.23% | 0.77% | 0.9923 | 13346 | 11654 | 94 | 98 |
| SVM | 97.32% | 2.68% | 0.9732 | 13261 | 11258 | 485 | 188 |
| RNN | 97.08% | 2.92% | 0.9708 | 13064 | 11394 | 349 | 385 |
| RH | 32.67% | 7.33% | 0.9267 | 12694 | 10653 | 1090 | 755 |
| RB | 89.59% | 10.41% | 0.8959 | 12272 | 10298 | 1445 | 1177 |

After analyzing the performance of these techniques, four of them have been chosen for the design of the final classifier:

two types of decision trees, C4.5 e ID3, Naive Bayes, and Supervised system classifier UCS as these give the best results in terms of hits.

## III. Fusion of Classification Techniques

A decision system for the detection of intrusions in an information system has been design based on the four classification techniques mentioned before. Figure 2 shows the classifier system that merges these methodologies.

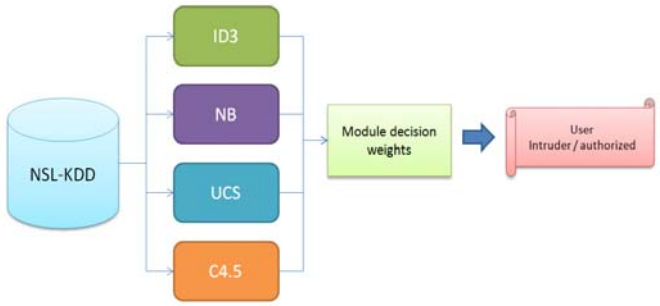



Fig. 2. Combination of the four classification techniques in a detection system

As it is possible to see in Figure 2, the information on the actions carried out for any user is introduced in the four classifiers. Each of them gives a result. These classifications outputs will be merged applying a rule based system that weighted them and will give the final result.

### A. Configuration of the Tools

Each classification system has been previously trained. All of them were trained using 25,192 registers and 17,102 for the tests.

The configuration of these techniques is given below.

- **Decision Trees C4.5 and ID3:** Decision trees utilize the 40 attributes of the network´s traffic dataset and apply the algorithms J48 and ID3 for the classification.

  Before the application of the ID3 decision tree, the Iterative Dicotomizer 3 Discretizer algorithm (ID3-D)[2][8, 9], was applied to preprocessing the data of the benchmark and in order to improve its performance.

- **Naive Bayes:** The Naive Bayes technique has used all the attributes of the dataset NSL-KDD. Again a discretization algorithm, the Bayesian Discretizer (Bayesian-D) was previously used.

- **UCS:** The supervised classifier system has worked with the 40 attributes of the NSL-KDD dataset. The genetic algorithm was configured using the following values:

  - Number of explorations: 100,000;
  - Size of the initial population: 6,400;
  - Delta: 0.1;
  - Selection type: RWS; nu: 10.0;
  - Tournament size: 0.4;
  - Mutation type: free; probability: 0.8;
  - Cross type: 2PT; probability: 0.04;

[2] KEEL: http://sci2s.ugr.es/keel/index.php, 2014-03-28

### B. Generation of the Making Decision System

The flow of the final classification system, as shown in Figure 2, is as follows.

1) **Data Input:** the network traffic information feeds the four classification systems. The 40 attributes have been previously normalized to increase the discrimination capability of each classifier [17].

2) **Classification of the inputs:** each classifier gives as result if the user is an intruder or an authorized user, with a confidence percentage.

3) **Weighting the previous results:** The results of the four previous classifiers (R1, R2, R3 and R4) are the inputs of a rule-based system.

   These results are combined according to expression (1).

$$\text{Resul} = R1 * 0.3 + R2 * 0.28 + R3 * 0.25 + R4 * 0.17 \quad (1)$$

   Different weights have been assigned to each one of the classification techniques, based on the result of Table I.

   The weights and techniques are as follows. C4.5 (weight = 0.3), ID3 (weight = 0.28), NB (weight = 0.25) and UCS (weight = 0.17). These values are obtained from tests carried out with each of the techniques, by trial and error. The percentage is related to the number of correct detections over the total number of tests.

4) **Final decision**. A threshold is applied to the result given by expression (5). If it is greater than 0.5, it will be considered an authorized user (0); any value under 0.5 will be detected as an intruder (1).

   This value will be included in the database as a new example in order to be used in the future. So the database is dynamically increased.

## IV. Results and Discussion

The indicators of the efficiency of the final making decision system are presented in Table II. The EMAE error, the accuracy of the classifier, and the true and false positives and negatives results are shown.

TABLE II. Results of the Final Classifier for the Detection of Intrusions.

| Correctly Classified | EMAE | Accuracy | True Pos Authorized | True Pos Intruder | False Pos Authorized | False Pos Intruder |
|---|---|---|---|---|---|---|
| 99.99% | 0.1% | 99.99% | 10254 | 6835 | 7 | 6 |

As previously mentioned, the precision of the different classification techniques for this database was very high, due to

the high number of available examples to train the methodologies. Anyway, the best ratio was 99.96 %.

With the synergy of the techniques, the new decision system has a hit ratio of 99.99%. That is, the fusion of different techniques improves the final classification ratio.

Besides, the number of false positives and false negatives values has significantly decreased. That is an interesting and useful result. Only the 0.1 % of the cases was false positives and false negatives. The information used in this study is complex and large. The classification systems were tested using different criterions, and therefore the obtained results can be considered reliable.

This study demonstrates how the fusion of different techniques may improve the results of a classification system. Once it has been proved, this synergy can be applied to more complex real problems, with uncertain or incomplete information.

## V. Conclusions and Future Work

This work proposes a method for the detection of intrusions based on the fusion of four classification techniques. All of them come from the Artificial Intelligence field.

The database NSL-KDD was used as a benchmark, in order to get some knowledge of the classification process of intrusions, and to prove the validity of the proposal. The results obtained by each technique separately are worse than when merging them in a final classifier that takes all of them into account. Therefore, the detection of the intruders has improved by the synergy of different classification techniques. Not only the percentage of intrusion detection is higher but the ratio of false positives is smaller.

This is a direct benefit to the security of any computer systems. As an immediate future work, this making decision system will be applied to a real database of a governmental institution, where more variables are considered.

Although the proposed method is slightly more complex than using an independent technique, we believe that when applied to more complex databases and large data, the proposal may be worth it as it is more efficient.

## Acknowledgments

Authors would like to thank the reviewers for their helpful comments.

## References

[1] H. Debar and J. Viinikka. "Intrusion detection: Introduction to intrusion detection and security information management." Foundations of security analysis and design III. LNCS vol. 3655, 207-236 2005.

[2] R. Bace and P. Mell. NIST special publication on intrusion detection systems. BOOZ-ALLEN AND HAMILTON INC MCLEAN VA, 2001

[3] M. Esposito, C. Mazzariello, F. Oliviero, S.P. Romano and C. Sansone “Evaluating pattern recognition techniques in intrusion detection systems” in PRIS, pp. 144 -153, 2005.

[4] W. Lee, S.J. Stolfo and K.W Mok, “Data Mining in work flow environments: Experiments in intrusion detection." in Proc. fifth ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 1999.

[5] V. Jaiganesh, S. Mangayarkarasi and P. Sumathi, "Intrusion detection systems: a survey and analysis of classification techniques." vol, 2, 1629-1635, 2013.

[6] A. Mitrokotsa and C. Dimitrakakis, "Intrusion detection in MANET using classification algorithms: The effects of cost and model selection.", Ad Hoc Networks, vol. 11, no. 1, 226-237, 2013.

[7] C. Guevara, M. Santos and J.A. Martín-H, "Identification of computer information system intruders by decision trees and artificial neural networks" in International Conference on Intelligent Systems and Knowledge Engineering ISKE 2013.

[8] W. Wang, X. Zhang, S. Gombault and S.J. Knapskog, "Attribute Normalization in Network Intrusion Detection", in 10th IEEE Int. Symp. on Pervasive Systems, Algorithms, and Networks, pp-448-453, 2009.

[9] S. Haykin, "Neural networks: a comprehensive foundation.", New York: Macmillan, 2004.

[10] J.R. Quinlan, "Induction of decision trees”. Machine learning, vol. 1, no 1, p. 81-106, 1986.

[11] C. X. Ling, Q. Yang, J. Wang, and S. Zhang, "Decision trees with minimal costs." in Proc. twenty-first Int. Conf. on Machine Learning, ACM, p. 69, 2004.

[12] S. Abe, "Support vector machines for pattern classification.", London: Springer, 2005.

[13] E. Bernadó-Mansilla and J.M. Garrell., "Accuracy-Based Learning Classifier Systems: Models, Analysis and Applications to Classification Tasks." Evolutionary Computation, vol. 11, no. 3, pp. 209-238, 2003.

[14] P. Domingos and M. Pazzani, "On the optimality of the simple Bayesian classifier under zero-one loss." Machine Learning vol. 29, no. 2-3, pp. 103-13, 1997.

[15] "NSL-KDD data set for network-based intrusion detection systems.” Available: http://nsl.cs.unb.ca/NSL-KDD/, March 2014.

[16] A.W. Moore and M.S. Lee, "Efficient algorithms for minimizing cross validation error.", in Proc. Int. Conf. Machine Learning, pp. 190-198, 1993.

[17] M. Santos , J.A. Martín-H, V. López and G. Botella, "Dyna-H: A heuristic planning reinforcement learning algorithm applied to role-playing game strategy decision systems", Knowledge-Based Systems, vol. 32, pp. 28-36, 2012.

# TRAINING STRATEGY TO IMPROVE THE EFFICIENCY OF AN INTELLIGENT DETECTION SYSTEM

CÉSAR GUEVARA[†] MATILDE SANTOS[†], VICTORIA LOPEZ[†]

*School of Computing, Department of Computer Architecture and Automation,*
*Complutense University of Madrid*
*Madrid, Spain*

Detection systems of computer accesses are essential for information security. In this article, we propose a classification system that combines two intelligent algorithms, those being: Supervised Classification Systems, UCS, and Decision Trees, C4.5. The experiments were carried out using a dataset provided by Amazon, the Kaggle challenge. The system has been trained by dump the dataset into subgroup. This training strategy has resulted more efficient than if the whole database is used as an only set.. The results prove the use of the proposed detection system provides higher classification accuracy and reduces the percentage of false positives in comparison to other classification techniques.

**Keywords**— Training Strategy, Supervised Classification Systems, Decision Trees, Computer Access, Detection Systems.

## 1. INTRODUCTION

The problem security in accessing information systems is important and concerns public and private companies worldwide. This problem occurs when employees make incorrect use of the access to a resource, because they steal confidential information or conduct operations that are not allowed.

The leaking of confidential information is one of the greatest threats for any country. It is necessary to develop efficiency methods to prevent these attacks and furthermore determine the resources available by each user, according to the tasks that he or she has to develop in their own working environment.

The problem is complex because it involves the creation of a behavioral profile of each employee, and this profile changes dynamically. Furthermore, it can be uncertain at some stages. Therefore the application of intelligent techniques has been proved to be very useful for these problems that require knowledge and

[†] César Guevara, cesargue@ucm.es.
[†] Matilde Santos, msantos@ucm.es.
[†] Victoria López, vlopezlo@ucm.es.

works with a lot of information that sometimes is incomplete or uncertain and may vary throughout time.

Some of the most used methods for dealing with this problem are classification methods such as: data mining, pattern recognition, classification systems based on genetic algorithms and supervised or non-supervised classification techniques. Although there many intrusion detection systems, it is still an open issue because the number of false positives when detecting fraud is usually high. The detection system proposed in this paper is based on two classification techniques, Decision Trees (C4.5) [14,16], and Supervised Classification Systems (UCS) [2,3]. It has been proven to be more efficient than their application separately.

Moreover, to improve the efficiency of this detection system, this work proposes a learning strategy that consists of dividing the training set into subgroups. These subgroups are formed according to some attributes that provide relevant information for the classification.

The database used for testing this proposal is an Amazon dataset, the Kaggle Challenge, which contains accesses of employees during the period of 2010 and 2011. The data consists of real, historic data [9]. Users were allowed or denied access to the different system resources manually.

Therefore, the objectives of this work are:

1. To build a classifier system that combines two intelligent classification techniques.
2. To identify relevant attributes in the Amazon Kaggel base data for detection of intruders.
3. To apply a learning algorithm that is based on the division of the training set.
4. To validate the detection system with the database.

To summarize, the final objective of this work is to generate an automatic method that analyzes the data in order to predict the rejection or acceptance of the accesses to the system resources by an employee.

The structure of the paper is as follows:

In the second section, the materials and methods that have been used are described. Section 3 is devoted to the design and generation of the intelligent decision system. Results and comparisons with other classification techniques are presented in Section 4. Finally, the paper ends with the conclusions.

## 2. MATERIALS AND METHOD

In this section, we present the phases that take part in the development of the classifier for system accesses, describing in thorough detail the data structure, pre-processing, the selection of classifying techniques, the creation of the target classifier model, and the tests that confirm its optimal execution.

Figure 1 below shows the process that was developed in this study for the classification of accesses to computer system resources. The process initiates with the input of information, with the permitted and denied users' accesses to be pre-processed. Later, the now refined datasets are divided into groups according to one or other relevant attributes that grants older information for classification utilizing filters [11] and wrappers [7]. In the construction the target classifier creates multiple instances of the target classifier C4.5 [14] and UCS [3, 8, 12] following the data groups created in the previous step. In the Analysis of the target classifier, the data is classified with techniques C4.5 and UCS in permitted and denied accesses of the dataset. Finally, in the evaluation of the execution, the results of the performance are obtained with error meters of the target classifier.

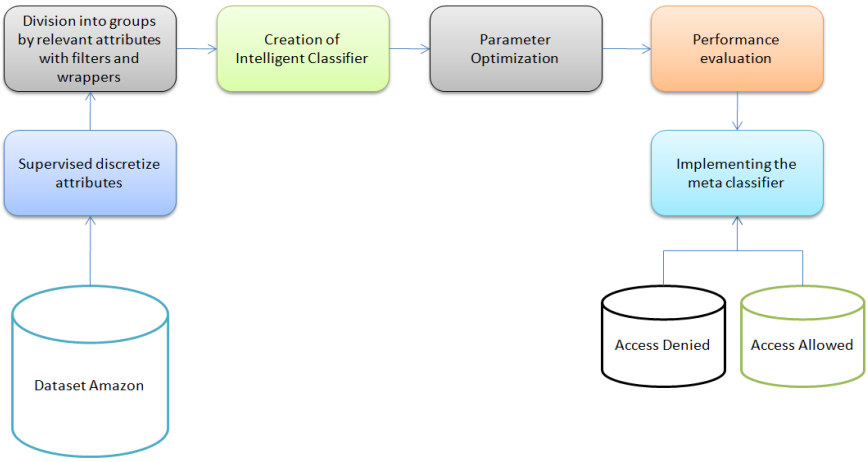



Figure 1. Classifier System Accesses.

### 2.1. *Dataset Kaggel Amazon*

The dataset consists of 10 attributes that describe the resources to those users who can have access, the identifier of the user, the user's role within the institution, and other information concerning the category and place wherever

developing one's work. Also, it possesses a discriminating attribute, that which permits or denies access to system resources, utilizing a 0 in order to deny access, and a 1 to grant access. The data for the training consists of 32,769 information lines, and 58,921 lines for the tests.

The data consists of real, historical data collected from 2010 and 2011. The employees were permitted or denied access to the resources in manual time. The following attributes are those that construct the dataset:

- **ACTION**: The action of assigning a "1" to the allowed access, and the denied a "0."
- **RESOURCE**: The identification code for the resource.
- **MGR_ID**: The identification code of the administrator who keeps the registry of access to a resource.
- **ROLE_ROLLUP_1**: Grouping code for categories id 1.
- **ROLE_ROLLUP_2**: Grouping code for categories id 2.
- **ROLE_DEPTNAME**: Department code of the company.
- **ROLE_TITLE**: Title code of the company.
- **ROLE_FAMILY_DESC**: The group's extended code to which a user within the company belongs.
- **ROLE_FAMILY**: The group code for a user within the company.
- **ROLE_CODE**: The company's senior code. This code is unique for each role.

The dataset from Amazon had to be analyzed and processed in order to complete any task with classification techniques, and furthermore, the comparison of the multiple techniques to select the most accurate for this study.

### 2.2. *Pre-Processing of Data*

The pre-processing of data lies in the task of eliminating data efficiently, as well as any other inconsistency, such as redundancy, noise, etc. The fundamental purpose of the preparation of the data is to manipulate and transform crude data in a manner which, the content of the enclosed information in the joint information can be exposed and easily accessible." [1,4]. Due to this, it is necessary to try to evaluate the performance of the dataset, which are the most relevant attributes, and eliminate any type of noise or inconsistency it encounters in the dataset using filters and wrappers in a joint manner to obtain a better reliability directly following the prior steps. This step allows the obtainment of trustworthy data for the generation of the most efficient classification models for this study [13].

## 3. CREATION OF INTELLIGENT CLASSIFIER

For the construction of the target classifier we utilized the techniques of Decision Trees C4.5, and the Supervised Classification System UCS, which uses a genetic algorithm in order to create classification rules. For the selection of these techniques, we compared multiple options, those that demonstrated their optimal execution with the Amazon dataset, which will be presented in the following section [6, 15].

### 3.1. *Division of Data Groups*

In this step, it's necessary to divide the refined dataset in pre-processing. Its size and complexity requires that they be studied independently. For this, it was necessary to subdivide it by the most relevant criteria of the attributes in equitable groups. Each of the data groups must contain all of the denied accesses as a base and assign the granted accesses equitably. This way we preserve the integrity of the information and we will be able to classify user accesses in the most efficient manner.

At the initial task of pre-processing of the data, it must distinguish the complete characteristics of the attributes that are contained in the Amazon dataset. The dataset contains 10 attributes of a complete set and an attribute of a classifier with 2 classes, those that assign an action of granting access "1" or denying it "0" to each of the users, as well a whole set. The dataset for the training possesses 32,769 instances of data and that of the tests contains 58,921.

The common task of eliminating redundancy and noise in the data was completed with the analysis tool Wakaito Environment for Knowledge Analysis (WEKA)[*]. For a deeper analysis of the variables in the Amazon dataset, it was necessary to use a wrapper and recognize that if a correlation between them was found between them [7], as indicated in table 1, we would apply the Gain Ranking Filter [17], which ranks the attributes that are evaluated for each value of the attribute by means of the measurement of the profit of information in respect to the class ACTION. This indicates that the attribute RESOURSE supplies the information better, which will be utilized in order to create the subgroups of data in the following phase.

[*] WEKA: http://www.cs.waikato.ac.nz/ml/weka/

Table 1. Ranking of attributes information (Gain Ranking Filter Weka).

| RANKING | ATTRIBUTES |
|---|---|
| 5 | ROLE_DEPTNAME |
| 4 | ROLE_ROLLUP_2 |
| 6 | ROLE_FAMILY_DESC |
| 3 | ROLE_ROLLUP_1 |
| 7 | ROLE_CODE |
| 9 | ROLE_TITLE |
| 8 | ROLE_FAMILY |
| 2 | MGR_ID |
| 1 | RESOURCE |

For the task of the division of the sub groups, we separated the permitted accesses and the denied into two groups. Previously, it was used as a base for all the information with denied accesses to be entered as constant en each of the 13 divisions that will be completed in the Amazon dataset. The permitted accesses must be arranged descending by the RESOURCE attribute (that supplies information in the classification better), and provides an accurate delimitation for the division of the information. Last, the permitted accesses were divided in groups of registries. This way we obtained 15 data groups for the training of the techniques of C4.5 and UCS, as shown in figure 2.

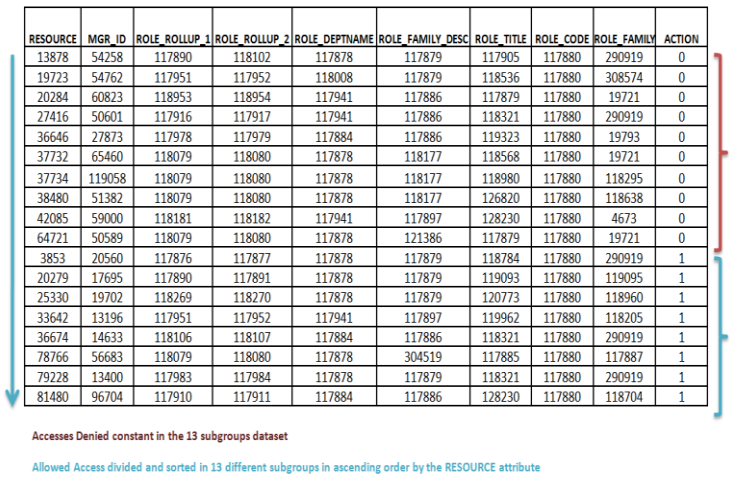



Figure 2. Division of Amazon dataset in sub groups.

### 3.2. *DEVELOPMENT OF CLASSIFIER*

Beyond utilizing the techniques C4.5 and UCS, the target classifier requires systems of rules in order to be able to direct the data of the entrance in order to be trained and analyzed. We have created a system of rules that directs the entered information by one of the sub models of the target classifier that have been trained with the dataset (d1, d2, d3…dn), to be classified [5, 10]. The rules have been outlined based on the attributes that supply more information in the pre-processing stage. In the calculations section, we will discuss the creation of these rules. For this reason figure 3 shows the model of the target classifier and its respective tools that it constitutes.

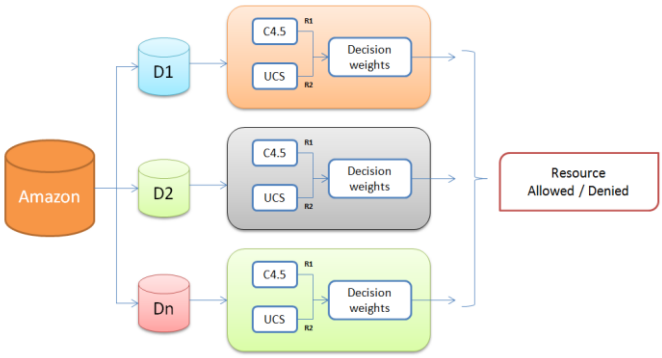



Figure 3. Design of intelligent classifier.

With the dataset now divided into sub groups, the following step is to evaluate the various classification techniques and finally select the most efficient one for the target classifier. Table 2 exhibits the performance of each of the various techniques with 15 sub groups from the Amazon dataset. This stated the techniques C4.5 and UCS demonstrated optimal performances, with an accuracy of 0.983 and 0.976, respectively.

Table 2. Classification Techniques Accuracy for each of the subgroups.

| | ACCURACY | | | | | | |
|---|---|---|---|---|---|---|---|
| **DATASET** | **ID3** | **XCS** | **BAYES Net** | **RNM** | **C4.5** | **UCS** | **SVM** |
| D1 | 0.970 | 0.950 | 0.969 | 0.974 | 0.982 | 0.981 | 0.950 |
| D2 | 0.978 | 0.945 | 0.981 | 0.952 | 0.988 | 0.981 | 0.927 |
| D3 | 0.994 | 0.956 | 0.992 | 0.972 | 0.997 | 0.994 | 0.920 |
| D4 | 0.976 | 0.869 | 0.961 | 0.972 | 0.985 | 0.981 | 0.896 |
| D5 | 0.963 | 0.912 | 0.928 | 0.948 | 0.973 | 0.967 | 0.780 |
| D6 | 0.969 | 0.951 | 0.974 | 0.973 | 0.983 | 0.974 | 0.765 |
| D7 | 0.967 | 0.950 | 0.965 | 0.939 | 0.976 | 0.973 | 0.691 |
| D8 | 0.976 | 0.913 | 0.921 | 0.890 | 0.990 | 0.974 | 0.784 |
| D9 | 0.968 | 0.886 | 0.975 | 0.923 | 0.982 | 0.975 | 0.797 |
| D10 | 0.976 | 0.838 | 0.975 | 0.901 | 0.982 | 0.974 | 0.795 |
| D11 | 0.963 | 0.910 | 0.964 | 0.922 | 0.971 | 0.973 | 0.884 |
| D12 | 0.982 | 0.897 | 0.984 | 0.924 | 0.990 | 0.977 | 0.884 |
| D13 | 0.975 | 0.972 | 0.956 | 0.924 | 0.979 | 0.972 | 0.654 |
| D14 | 0.967 | 0.944 | 0.979 | 0.926 | 0.983 | 0.977 | 0.654 |
| D15 | 0.973 | 0.927 | 0.978 | 0.926 | 0.981 | 0.976 | 0.654 |
| **AVERAGE** | **0.973** | **0.921** | **0.967** | **0.938** | **0.983** | **0.976** | **0.802** |

With the most efficient techniques selected, we proceeded to the training phase and testing each one individually from the dataset, as indicated in figure 4.

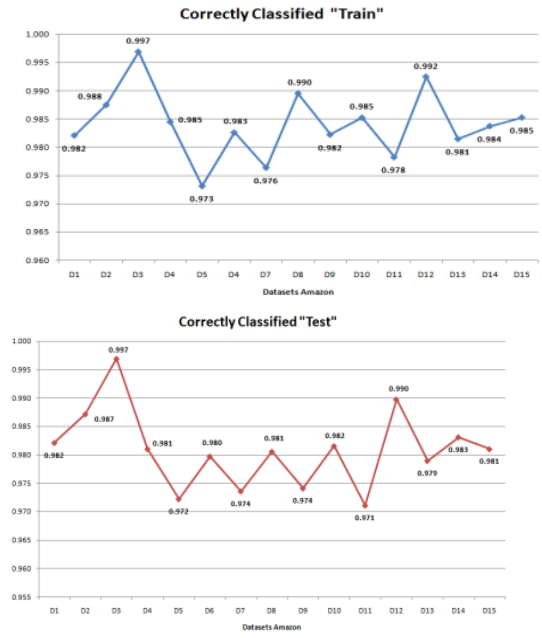



Figure 4. Accuracy of classifier.

At the end, the target classifier utilizes a decision system to validate the answers obtained from the classification algorithms C4.5 with R1, and UCS with R2. Due to this, the selection was completed by weighing the results utilizing formula 1, granting the weight of 0.6 to algorithm C4.5 and a weight of 0.4 to the UCS algorithm. This optimized the classification for the detection of system accesses.

$$\textbf{Result= 0.6*R1+0.4*R2} \quad \textbf{(1)}$$

## 4. Experiment Results

In general the target classifier obtained an accuracy of 0.984 in training and 0.981 in the tests, using all of the Amazon dataset with the propose methodology, as show in table 3, in comparison to other techniques without utilizing the proposed methodology.

Table 3. Comparison of results of the classification algorithms with the dataset Amazon.

| | **ACCURACY** | |
|---|---|---|
| **ALGORITHM** | **TRAIN** | **TEST** |
| **Meta Classifier System UCS and C4.5** | **0.984** | **0.981** |
| Naive Bayes | 0.912 | 0.901 |
| UCS | 0.949 | 0.939 |
| XCS | 0.942 | 0.936 |
| Multilayer perceptron | 0.942 | 0.94 |
| J48 | 0.945 | 0.941 |

In the study the KEEL (Knowledge Extraction bases on Evolutionary Learning)[†] tool for training and testing classification techniques are used.

## 5. Conclusions

In conclusion, we managed to obtain results for the evaluation, where the executions of the proposed target classifier in training and testing demonstrate an increased performance in the classification in comparison to other algorithms. In regards to accuracy, the target classifier managed to efficiently detect permitted and denied accesses of Amazon users, a critical factor concerning the objective of this present study.

[†] KEEL: http://sci2s.ugr.es/keel/index.php

On the basis of this study's results, the proposed method for the creation and application of this target classifier offers an efficient prediction. The proposed process for the execution of this study, from the pre-processing of data (with the division of the Amazon dataset in multiple sup groups) up to the utilization of algorithms of C4.5 and UCS combined confirmed that this is a trustworthy classifier for these types of processes.

The main goal of this study is to propose a method for the detection of permitted and denied accesses within a computer system, utilizing evolutionary algorithms together as well as other classification techniques in order to improve the performance and execution of detection. In summary, the completed experiments with the Amazon dataset and the creation of a model based on the measures of execution and error obtained a high accuracy rate, and conversely, a reduced rate in classification errors.

The experiment's results exhibit that classification with the model of the target classifier with multiple datasets improve the prediction rate's percentage, which suggests that the use of computer resources would be considerably low as well.

Considering that this method proved efficient with this data, a future line of investigation would be to develop a method for the detection of accesses to computer systems and intrusion detection that allows for the utilization of various evolutionary techniques in order to be adaptable to the always changing behavior of users and intrusions in the future.

**Acknowledgments**

This work has been supported by the Ministry of Higher Education, Science, Technology and Innovation SENESCYT of Government of the Republic of Ecuador with the scholarship "**Convocatoria abierta 2011 y 2012 segunda etapa**".

## References

1. Al-Hamami, A., Mohammad, A., Hassan, S. (2006). Applying data mining techniques in intrusion detection system on web and analysis of web usage. Information Technology Journal, 5(1), 1-4.
2. Bernadó-Mansilla, E., Garrell-Guiu, J. M. (2003). Accuracy-based learning classifier systems: models, analysis and applications to classification tasks. Evolutionary computation, 11(3), 209-238.
3. Butz, M. V., Lanzi, P. L., Wilson, S. W. (2008). Function approximation with XCS: Hyper ellipsoidal conditions, recursive least squares, and compaction. Evolutionary Computation, IEEE Transactions on, 12(3), 355-376.

4. Cheng, J., Greiner, R. (1999, July). Comparing Bayesian network classifiers. In Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence (pp. 101-108). Morgan Kaufmann Publishers Inc..

5. Friedman, N., Geiger, D., Goldszmidt, M. (1997). Bayesian network classifiers. Machine learning, 29(2-3), 131-163.

6. Guijarro, M., Pajares, G., Abreu, R., Garmendia, L., Santos, M. (2007, October). Design of a hybrid classifier for natural textures in images from the Bayesian and Fuzzy paradigms. In Intelligent Signal Processing, 2007. WISP 2007. IEEE International Symposium on (pp. 1-6). IEEE.

7. Gutlein, Martin, et al. "Large-scale attribute selection using wrappers." Computational Intelligence and Data Mining, 2009. CIDM'09. IEEE Symposium on. IEEE, 2009.

8. Holland, J. H., Holyoak, K. J., Nisbett, R. E., Thagard, P. R. (1989). Induction: Processes of inference, learning, and discovery.

9. Kaggel, Amazon.com - Employee Access Challenge Retrieved from http://www.kaggle.com/c/amazon-employee-access-challenge

10. Kantardzic, Mehmed. Data mining: concepts, models, methods, and algorithms. John Wiley & Sons, 2011.

11. Koren, Yehuda. "Factorization meets the neighborhood: a multifaceted collaborative filtering model." Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2008.

12. Orriols-Puig, A., Casillas, J., Bernadó-Mansilla, E. (2007, July). Fuzzy-UCS: preliminary results. In Proceedings of the 2007 GECCO conference companion on Genetic and evolutionary computation (pp. 2871-2874). ACM.

13. Phua, Clifton, et al. "A comprehensive survey of data mining-based fraud detection research." arXiv preprint arXiv:1009.6119 (2010).

14. Polat, Kemal, and Salih Günes. "A novel hybrid intelligent method based on C4. 5 decision tree classifier and one-against-all approach for multi-class classification problems." Expert Systems with Applications 36.2 (2009): 1587-1592.

15. Smith-Miles, Kate A. "Cross-disciplinary perspectives on meta-learning for algorithm selection." ACM Computing Surveys (CSUR) 41.1 (2008): 6.

16. Ture, Mevlut, Fusun Tokatli, and Imran Kurt. "Using Kaplan–Meier analysis together with decision tree methods (C&RT, CHAID, QUEST, C4. 5 and ID3) in determining recurrence-free survival of breast cancer patients." Expert Systems with Applications 36.2 (2009): 2017-2026.

# Poster Abstract: Data Leakage Detection Algorithm Based on Sequences of Activities

César Guevara, Matilde Santos, and Victoria López

Complutense University of Madrid
Madrid, Spain
{cesargue,msantos}@ucm.es, vlopez@fdi.ucm.es

**Abstract.** In this paper we propose an algorithm for data leakage detection. This algorithm works with historical data of the activities of authorized users in a computer system. This information gathers data of the hour of the accesses, duration, day of the week, operation, table that has been accessed, etc. They have been provided by a governmental institution at Ecuador. The procedure has two phases. The first one is based on the calculation of the probability of each activity that is carried out by each user. These activities are for instance to modify a file, delete, copy, etc. The different activities at different times are codified by an integer or character. The Page Rank algorithm is used to calculate the probability of every activity. But the activities form sequences, that is, during a session (time between the user logs in and logs out), the user carries out different activities, one after another. These sequences of activities may have different length. The probability of each sequence of activities is then calculated by applying the Bayes' theorem. The minimum of these conditional probabilities is obtained and set as a threshold, $r_{min}$. If a new chain of activities, $s_i$, is introduced in the detection leakage system, first of all the page rank is applied and then the Bayes' law, so a probability of that particular sequence of activities for that user is obtained, let's say $p_i$. This probability, $p_i$, is compared to the threshold, $r_{min}$. If it does not surpass it, a leakage warning message is generated. Otherwise, the sequence of activities goes to the second phase of the procedure. The sequence of activities that is being tested is then compared to all the sequences of activities of that particular user that are stored in the historical database. Applying the Smith and Waterman algorithm, a similarity score is obtained for this sentence regarding the rest of the sequences previously carried out by the user. This algorithm determines similar regions between two strings comparing segments of all possible lengths and optimizes the similarity measure. If the score is higher than a second threshold, let' say that there are more than ***n*** characters (codified activities) that are the same than another sequence of the same user, the activity can be considered right; otherwise it may be a leakage. Although this algorithm is still being developing, its main contribution is the way it handles different lengths of the sequences and how it works with a behavioral pattern of each user.

**Keywords:** Data Leakage Detection, Page Rank algorithm, Bayes' Theorem, Smith and Waterman algorithm, behavioral pattern.

A. Stavrou et al. (Eds.): RAID 2014, LNCS 8688, pp. 477–478, 2014.

# Poster Abstract: Data Leakage Detection Algorithm including Sequences of Activities

César Guevara (cesargue@ucm.es), Matilde Santos (msantos@ ucm.es), Victoria López (vlopez@fdi.ucm.es)
Complutense University of Madrid, School of Computing, Department of Computer Architecture and Automatic Control
28040-Madrid, Spain

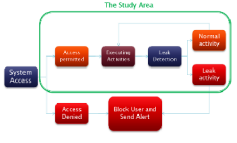
## ABSTRACT

Information leakage is a current problem that worries most public and private institutions around the world. Data is one of the most important assets for an institution. Therefore it is important to design and implement computational tools in order to avoid leakage, loss or modification of crucial information. This problem is especially critical when data leakage is caused by authorized people.

In this paper we propose an algorithm for data leakage detection. The algorithm is based on the one hand on the probability of certain activities that authorized users usually do (such as modify a file, delete, copy, etc.). On the other hand, it also provides a ranking of similarity between the sequence of activities of particular users, that is, the frequency and order of these activities.

## CURRENT SCENARIO (MATERIALS)

The public sector of the Republic of Ecuador, as any other government, has information systems of utmost importance. It is necessary to have an adequate control even of the authorized users' activities who have access to such information.



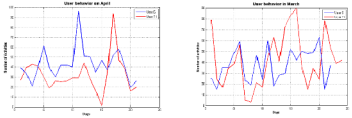

The information used in this study is the sequences of activities carried out by public servants who have authorized access to the governmental databases (modify a file, copy, insert, etc). This information gathers data of the hour of the accesses, duration, day of the week, operation, table that has been accessed, etc. It has been codified as a string of characters.

## METHODS

- **Page Rank algorithm** allows us to obtain the probabilities of each of the activities undertaken by the users.
- **Bayes' Theorem** gives the conditional probability of a sequence of activities of a particular user.
- **Smith and Waterman algorithm** determines similar regions between two codified sequences of activities comparing segments of all possible lengths and giving a score.

## USER BEHAVIOUR

The user behavior is dynamic and the number of activities during a session is variable




Activities within a user session in a computer system has the following structure

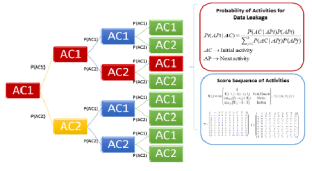

## ALGORITHM

**Phase 1:**

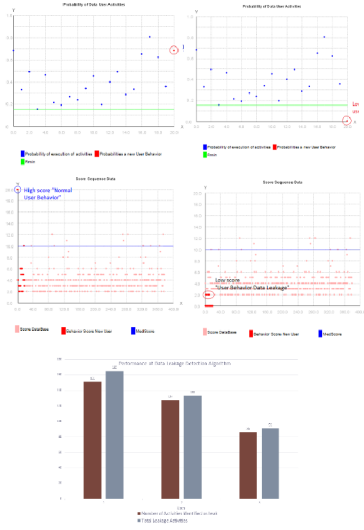
1) Page Rank: activity's probability $P_i(a_i)$
2) Bayes' Law: conditional probability of a sequence of activities of an user $P_i(s_i)$
   - Set a threshold as a minimum ($r_{min}$)
3) Evaluation of a new sequence of activties $s_e$
   - If $P_i(S_e) < r_{min}$ warning leakage otherwise go to Phase 2

**Phase 2:**

1) Smith & Waterman algorithm: score of similarity of that sequence $S_i(s_e)$
   - If $S_i(s_e) < n$ warning leakage otherwise O.K.




## RESULTS




## CONCLUSIONS

- The proposed algorithm handles different lengths of the sequences
- It works with a behavioral pattern of each user
- Preliminary results are encouraging

# Sistema de Detección de Intrusos aplicando Selección Negativa en Perfiles de Usuario

César Guevara, Matilde Santos, Victoria López
Facultad de Informática, Universidad Complutesnse de Madrid
cesargue@ucm.es,msantos@ucm.es,vlopez@fdi.ucm.es

*Resumen*— **Esta trabajo propone un modelo de datos y la aplicación de Sistemas Inmunes Artificiales para el desarrollo e implementación de un Sistema de Detección de Intrusos, que permita identificar actividades anómalas e intrusivas dentro de un sistema de información de gobierno. El IDS propuesto utiliza un conjuntos de datos del comportamiento de los usuarios que han ejecutado múltiples tareas en un período considerable de tiempo dentro del sistema real. Las contribuciones específicas son: modelo de datos, la identificación de un perfil de usuario y la aplicación de algoritmos de selección negativa y de búsqueda de secuencias de forma local como KPM para identificar tareas intrusivas de forma eficiente. Los resultados son óptimos y con una tasa de falsas alarmas bajo.**

**Palabras Clave:** Sistema inmune artificial, Sistema de Detección de Intrusos, comportamiento de usuario, Selección negativa

## I. Introducción

En la actualidad la seguridad de la información es un área muy importante para cualquier persona o institución alrededor del mundo, ya que los datos se han convertido en el activo más importante el cual debe ser salvaguardado de una manera eficiente y adecuada. La gran mayoría de los datos deben mantenerse seguros de cualquier intruso o actividad no permitida, de modo que la seguridad tiene una importancia crítica. El termino de "intrusión" como lo presenta [1] se puede definir como cualquier conjunto de acciones que tratan de comprometer la integridad, confidencialidad o disponibilidad de un recurso. Por ello es necesario utilizar una herramienta que pueda detectar estas actividades y mantener la información accesible solo a las personas autorizadas. Dicha herramienta se la denomina como Sistema de Detección de Intrusos (IDS), el mismo que analiza eventos que suceden en un sistema informático en busca de signos de intrusiones. El principal objetivo de un IDS es monitorear la actividad en un servidor, red o un equipo informático (PC, Tablet, móvil, etc.) de tal forma que permita identificar de manera eficiente posibles ataques o intentos de violación a la seguridad basados en patrones de comportamiento, firmas de código o análisis de protocolos, para luego alertar al administrador del equipo. El gran auge en el desarrollo e implementación de múltiples IDS han surgido algunas maneras en la recolección y utilización de la información para el sistema de detección, los cuales se describen continuación:

- Basado en el Host: Este tipo de sistema de detección utiliza la información de tareas o actividades realizadas en el equipo donde se encuentra funcionando el IDS.
- Basado en la Red: Este sistema de detección utiliza toda la información del tráfico de la red que se genera en el ambiente que se encuentra funcionando, es decir, recolecta la información de red interna como información de red externa (conexiones entrantes y salientes).

Con los tipos anteriores de recolección de datos permiten una variedad de IDS como la detección de usos indebidos (misuse detection) la cual compara la información recogida con descripciones (o firmas) de ataques conocidos. Por otra parte, la detección de anomalías (anomaly detection) utiliza los datos históricos sobre la ejecución de tareas o actividades en el sistema y detalla el comportamiento deseado de usuarios como de las aplicaciones, para construir un perfil que representa la operación normal del sistema monitorizado, e identifica patrones de actividades que se desvían del perfil definido.

Además existen una variedad de técnicas , metodologías y algoritmos para el desarrollo de un IDS. El área más utilizada para los IDS es la aplicación de Inteligencia Artificial con las técnicas de Machine Learning y Minería de Datos (Data Mining). En esta área se han realizado múltiples investigaciones y artículos, obteniendo grandes avances de sobre la aplicación distintas técnicas como Árboles de Decisión, Redes Neuronales, Algoritmos Genéticos, Support Vector Machines, Sistemas Inmunes Artificiales entre otras [2]. Los IDS también utilizan técnicas y modelos estadísticos con el propósito de automatizar completamente la detección de ataques maliciosos distinguiendo del uso normal de los sistemas. Las más utilizadas son Redes Bayesianas, Cadenas de Markov, etc.

En este trabajo se presenta una propuesta de investigación en torno a la seguridad informática, y más específicamente en el área de los sistemas de detección de intrusos (IDS) basados en anomalías de comportamiento de usuarios. El sistema propuesto utiliza la técnica de Sistemas Inmunes artificiales aplicando el algoritmo de selección negativa, además, un algoritmo de búsqueda de secuencias en forma local llamado Knuth-Morris-Pratt (KMP).

El artículo está organizado de la siguiente forma: en la Sección 2 se detalla trabajos realizados anteriormente presentado sus fortalezas y comparándolos con nuestra propuesta. En la Sección 3 se detalla el objetivo de la investigación y la aportación científica, además se presenta los materiales y métodos utilizados para la construcción del IDS. Posteriormente, en la Sesión 4 se presentan desarrollo del algoritmo. En las sessión 5 se presentan los resultados del sistema propuesto. Finalmente, en la Sección 6 se pre-

sentan las conclusiones y trabajos futuros que han surgido de esta investigación.

## II. Trabajos Relacionados

Un sistema de detección de intrusos IDS es un sistema de software o hardware automatizado para realizar un proceso de monitoreo y análisis de datos del medio informático para la detección de intrusiones como lo presenta [3][4] [14]. En múltiples trabajos en los cuales los IDS centran su atención en la técnica de Sistema inmunes artificial como una óptima alternativa para identificar comportamientos anómalos los presenta en [5] [6] [7] [8] . Los cuales validan que dicha técnica la cual proporciona resultados fiables y que son una gran área de investigación para el futuro.

Es por eso que con un creciente número de investigadores informáticos que seleccionan y estudian esta técnica, las cuales, obtienen un gran éxito como un mecanismo natural para la solución de diversos problemas, incluyendo diagnóstico de fallos, la detección de virus y detección de fraude hipotecario como lo presenta [9] [10] [11] [12] [13].

De los trabajos más recientes en la aplicación de sistemas inmunes artificiales es el presentado por [15], el cual propone un algoritmo de selección negativa que ha demostrado ser eficaz para los problemas de detección de anomalías y que presenta una nueva estrategia en la etapa de entrenamiento, además, un continuo entrenamiento para la reducción de muestras self para reducir el coste computacional en fase de pruebas. Este algoritmo puede obtener la tasa de detección más alta y una tasa más baja de falsas alarmas en la mayoría de los casos.

Por otro lado la investigación realizada por [16] presenta un nuevo enfoque para la detección de anomalías en el tráfico de red utilizando detectores generados por un algoritmo genético. Este trabajo utiliza el algoritmo de selección negativa en un sistema inmune que puede detectar patrones anómalos. Este trabajo, muestra una comparativa con varios otros experimentos realizados con un conjunto de datos conocidos llamado NSL-KDD. El algoritmo propuesto muestra resultados muy buenos en el análisis, en comparación con otros métodos de aprendizaje automático.

Otro trabajo muy interesante que en el cual se aplica el sistemas inmunes con selección negativa, como lo presenta [17]. El trabajo propuesto detalla los excelentes mecanismos de auto-aprendizaje, la capacidad de adaptación del sistema inmunológico humano y además muestra los conceptos y las definiciones formales de antígeno, anticuerpos y células de memoria en el dominio de seguridad de la red. Un aspecto muy importante de la investigación es como se establecen las formulaciones evolución dinámica de los perfiles de detección (incluida la generación de los perfiles de detección, de aprendizaje dinámico, transformación dinámica, y la auto-organización dinámica), que logrará que los perfiles de detección dinámica se puedan sincronizar con el entorno de red real. Este trabajo obtiene resultados experimentales buenos en la cual se convierte en como una solución a tener en cuenta para la detección de anomalías en red.

Este trabajo propone la aplicación de un sistema inmune con selección negativa en las tareas que ejecutan los usuarios dentro de un sistema informático, donde, a partir del histórico de estas ejecuciones se genera un perfil de uso normal el cual permite identificar anomalías con resultados óptimos y con un coste computacional bajo. En las siguientes secciones se presentará las aportaciones de la propuesta y además todo el proceso de modelado de la solución para un IDS.

## III. Métodos y Materiales

### A. Objetivo y aportaciones de la Investigación

El objetivo principal de la investigación es desarrollar un algoritmo dinámico y eficiente el cual pueda identificar el comportamiento anómalo de un usuario o agente que realice tareas intrusivas en un sistema informático. Por otra parte las aportaciones de la investigación al estudio de IDS son:

- Utilización de datos reales de usuarios de un sistema informático real para el desarrollo del modelo de datos y el algoritmo de detección.
- Aplicación de un nuevo modelo de datos que represente el comportamiento de los usuario de forma eficiente.
- Aplicación del algoritmo de selección negativa de un sistema inmune artificial al modelo de datos para la identificación de comportamientos anómalos de los usuarios.
- Aplicación del algoritmo KMP para la identificación de comportamientos anómalos en nuevas secuencias de tareas.
- Desarrollar una nueva forma de detección de intrusos dinámica y eficiente dirigido a sistemas informáticos.

En la siguiente sección se describen los métodos y materiales utilizados para el desarrollo del algoritmo.

### B. Métodos

En esta sección se presentan los algoritmos aplicados en el presente trabajo. Los algoritmos utilizados son Algoritmo de selección negativa (NSA)y Algoritmo Knuth Morris Pratt (KMP).

#### B.1 Sistema Inmune Artificial con Selección Negativa

El algoritmo de selección negativa define el "self" mediante la construcción de modelos de comportamiento normales de un sistema monitorizado. Este proceso genera un número finito de patrones aleatorios que se comparan a cada modelo específico de self como lo presenta [11].

#### B.2 Algoritmo de Knuth Morris Pratt

El objetivo principal de este problema es encontrar una cadena dentro otra cadena. En un patrón P para cada posición i, spi(p) se dice que es la longitud del sufijo más largo de P[1; 2i], que coincide con el prefijo P. Es similar a navegar dentro de la cadena y que realiza sus comparaciones de izquierda a derecha. También calcula los desplazamientos máximos posibles de izquierda a derecha para el patrón P, [18].

### C. Materiales

Los datos con los que se ha realizado este trabajo fueron proporcionados de un sistema real del gobierno de la República de Ecuador. La información es confidencial y por esta razón los datos se han codificado para proteger la integridad de su información. El conjunto de datos se ha recolectado mediante la captura de las tareas ejecutadas por los usuarios en un período considerable de tiempo. Estos datos constan de diez conjuntos diferentes de información sobre la ejecución de las tareas de los usuarios. Esta información esta agrupada tanto por usuarios y por sesiones de usuario como se describe más adelante. El conjunto de datos constan del período de 2011 a 2013 y son 15.571 registros de sesiones 5312 de registros de sesiones anómalas. La información que contienen todo el comportamiento normal de cada usuario de inicio de sesión del sistema y las ejecuciones de tareas las cuales no tienen un tamaño definido. El problema principal es distinguir entre comportamiento normal y comportamiento anormal porque las tareas ejecutadas por los usuarios depende de la carga de trabajo, asignaciones o diferentes factores.

El sistema contiene siete tablas principales en la base de datos $(Tb_1, Tb_2, ..., Tb_7)$ y esta base de datos es posible ejecutar las cuatro operaciones sql que son Insertar, Modificar, Eliminar y Buscar. La estructura de datos de una sesión está conformada por un login (acceso al sistema) y una o varias tareas ejecutadas en forma secuencial. Las tareas están codificadas con 28 tareas genéricas más el inicio de sesión. Cada tarea tiene un código "$T$" más un número para identificar cada tarea. El comportamiento de cada usuario es dinámico y diferente de otro usuario. El conjunto de tareas se define como $T = \{T_1, T_2, T_3, ..., T_m\}$ donde $m$ es el número de la tarea y el conjunto de posibles tareas ejecutadas por el usuario $U$ se define como $T^U = \{T_1, T_2, T_3, ..., T_n\}$ donde $n$ es el número de tareas ejecutadas. El conjunto de posibles sesiones se define como $S = \{S_1, S_2, S_3, ..., S_n\}$ donde $S_n \in Hs$. Todas las sesiones grabadas del mismo usuario forman la base de datos histórica de ese usuario, estas sesiones se llama "Sesiones pasivas" $(Hs^U)$,y una nueva sesión de usuario se llama Sesión Activa $As^U$. Esta estructura no tiene tamaño fijo como se muestra en la figura 1.

Para identificar las tareas ejecutadas por el usuario se ha propuesto un modelo de datos en el cual se identifican las tareas T más ejecutadas y se crea una tabla que defina si existe esa tarea en el sesión realizada S. En este trabajo se ha comprobado que cada uno de los usuarios no realizan más de 13 tareas de las 28 existentes para su comportamiento normal. Por esta razón se ha tomado ese valor como máximo de tareas ejecutadas para este nuevo modelo. El modelo de datos identifica si existe una o varias tareas T en una sesión S y asigna un valor de "1" al casillero correspondiente de cada tarea y caso contrario "0" al no existir dicha tarea. El modelo de datos está presentado en la tabla 1.

Si en el caso de existiese un número mayor o menor de tareas al establecido el modelo de datos es adaptable al comportamiento de cada uno de los usuarios.

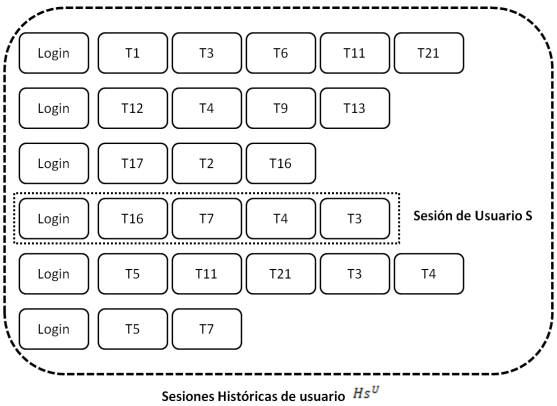



Fig. 1. Datos de sesiones Históricas del comportamiento de usuario en un sistema informático.

| T1 | T2 | T3 | T4 | T5 | T7 | T9 | T11 | T12 | T14 | T16 | T20 | T21 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |

TABLA I
MODELO DE DATOS PROPUESTO

## IV. Desarrollo del Algoritmo de Detección de Intrusos

Sobre el algoritmo de detección de intrusos el cual se basa en la nueva estructura de datos presentada anteriormente, además en el algoritmo de selección negativa y el algoritmo KMP que fue descrito en los apartados anteriores. La detección de tareas anormales podría resumir en los siguientes pasos:

- Paso 1. Generación de secuencias anormales: En esta sección utiliza el algoritmo de selección negativa y KMP para generar secuencias anómalas utilizando el conjunto de datos del modelo propuesto.
- Paso 2. Detección de Intrusos en Secuencias Activas: Finalmente en este paso se identifica si la secuencia activa $As^U$ es anómala o no utilizando el conjunto de datos resultante en el paso 1. De esta manera identificar de una forma eficiente y dinámica si es intrusiva o normal.

### A. Paso 1. Generación de secuencias anormales

En este paso lo que se propone utilizar un sistema inmune artificial aplicando el algoritmo de selección nega-

tiva, el cual a partir de un conjunto de datos del modelo propuesto genere sesiones anómalas para poder ser comparadas en el siguiente paso con una sesion activa. Este paso consta de tres fases las cuales son:

- Fase 1: Generar aleatoriamente la población inicial $B_0$ utilizando el modelo de datos propuesto con detectores $r$ que incluya los datos espaciales con las características, donde $r \in B_0$. Estos detectores se basan en las secuencias históricas de tareas $Hs^U$, cada figura azul representa un detector. Esta fase se ilustra en la figura 2a. En la generación de la población inicial número de elementos $B_0$ se calcula con la formula $B_0 = p^v$, donde $p$ es el número de bits de posibilidades y $v$ es la cantidad de tareas que ejecuta el usuario, este valor permite crear un número sesiones.
- Fase 2: Con los datos de entrenamiento $B_0$ una porción del espacio se define como normal (itself) utilizando el algoritmo KPM. Esta fase se ilustra en la figura 2b.El algoritmo de búsqueda local analiza cada registro de $B_0$ buscando similitudes con los las sesiones del modelo de datos de cada uno de los usuarios, si existe una similitud los marca como normales, caso contrario los marca como anómalos denominados $D_n$.
- Fase 3: Se elimina todos los detectores que se superponen con la región definida en itself, dejando el resto como detectores anormales $D_n$. Esta fase se ilustra en la figura 2c.El algoritmo de selección negativa obtiene detectores de comportamiento anómalo (rojo) para el siguiente paso del sistema de detección, como se muestra en la Figura 2d.

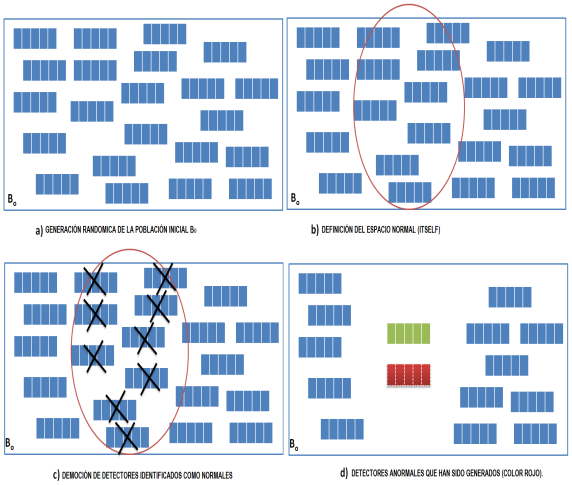



Fig. 2. Fases del sistema de inmune con selección negativa

En el siguiente paso se utilizan los detectores anómalos $D_n$ para identificar si la sesión activa $As^U$ es anómala o normal.

### *B. Paso 2. Detección de Intrusos en Secuencias Activas*

En este último paso se analiza la secuencia activa $As^U$ y se compara con todo el conjunto de detectores anómalos $D_n$. Se introduce nueva sesión activa $As^U$ para determinar la similitud con los detectores $D_n$ con el algoritmo KPM.Si la similitud (umbral de estimulación) $Sth > 0$ es igual a $TRUE$ el detector se activa y el antígeno se clasifica como anómalo, de lo contrario, se clasifica como normal.

Este paso es el que se repite de forma continua durante el uso del sistema informático determinando en que tarea el usuario se desvía a un comportamiento anómalo. La implementación del IDS es en una capa intermedia entre la interfaz del usuario y la ejecución de las operaciones en la Base de datos. Esto permite que las actividades del usuario no queden bloqueadas sino solamente las tareas que han sido detectadas como intrusivas, como se puede apreciar en la figura 3.

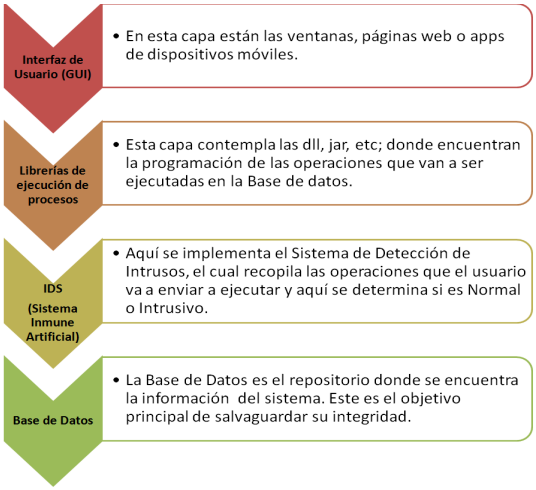



Fig. 3. Esquema de implementación del IDS en el sistema informático

En la siguiente sección se presentan los resultados de la propuesta de forma detallada.

## V. Resultados

En esta sección se presentan los resultados de las pruebas de detección del IDS en fase incial, ya que para obtener resultados definitivos es necesario tener pruebas más extensas para tener cifras definitivas. Para estas pruebas se utiliza la información diez usuarios de un sistema informático real.Durante el presente estudio se evidenció muchas características que hacian de la propuesta tener resultados de detección eficiente, una de estas es el modelo de datos y la aplicación de selección negativa del comportamiento normal de cada uno de los usuarios.

Los resultados de las pruebas de la aplicación del algoritmo IDS muestran la tasa de clasificado correctamente (CC), tasa clasificado incorrectamente (IC). También presenta la detección tiempo promedio (segundos) para cada usuario del estudio. Por otro lado se presentan valores de Verdaderos positivos (TP), Verdaderos negativos (TN), Falsos positivos (FP), Falsos negativos (FN). Para valorar el desempeño adecuado del IDS es necesario presentar

valores de Precisión, Tasa de Detección y Tasa de Falsas alarmas las cuales se presentan a continuación:

La precisión (PR)se refiere a la proporción de datos clasifica un tipo preciso en total de datos, a saber, la situación TP y TN, por tanto, la precisión esta daba por

Precisión= $\frac{(TP+TN)}{(TP+TN+FP+FN)} * 100\%$.

Tasa de detección (DR) se refiere a la proporción de comportamiento anómalo detectado entre todos los datos de comportamiento anómalo, es decir, la situación de los TP, la tasa de detección es por lo tanto esta dada por

Tasa de Detección= $\frac{(TP)}{(TP+FN)} * 100\%$.

Tasa de falsas alarmas (FA) se refiere a la proporción que los datos de comportamiento normal se detecta falsamente como un comportamiento anómalo, es decir, la situación de la FP, por lo tanto la tasa de falsas alarmas es

Tasa de falsas alarmas= $\frac{FP}{(FP+TN)} * 100\%$.

Este análisis de detección de intrusos se ilustran en las tabla 2 y 3.

| Usr | CC | IC | CC % | IC % | Tiempo Promedio |
|---|---|---|---|---|---|
| 1 | 1943 | 26 | 98.68 | 1.32 | 0.03 |
| 2 | 1833 | 8 | 99.565 | 0.435 | 0.012 |
| 3 | 2049 | 24 | 98.842 | 1.158 | 0.016 |
| 4 | 2563 | 15 | 99.418 | 0.582 | 0.028 |
| 5 | 1909 | 29 | 98.504 | 1.496 | 0.064 |
| 6 | 2057 | 7 | 99.661 | 0.339 | 0.045 |
| 7 | 2116 | 6 | 99.717 | 0.283 | 0.049 |
| 8 | 2080 | 8 | 99.617 | 0.383 | 0.087 |
| 9 | 2218 | 9 | 99.596 | 0.404 | 0.063 |
| 10 | 1973 | 10 | 99.496 | 0.504 | 0.067 |
| | | *Media* | *99.3096* | *0.6904* | *0.0461* |

TABLA II

Resultados de IDS propuesto en clasificado correctamente (CC), clasificado incorrectamente (CC).

| Usr | TP | FP | FN | TN | % PR | % DR | % FA |
|---|---|---|---|---|---|---|---|
| 1 | 689 | 25 | 1 | 1254 | 98.68% | 99.86% | 1.95% |
| 2 | 468 | 4 | 4 | 1365 | 99.57% | 99.15% | 0.29% |
| 3 | 456 | 13 | 11 | 1593 | 98.84% | 97.64% | 0.81% |
| 4 | 549 | 4 | 11 | 2014 | 99.42% | 98.04% | 0.20% |
| 5 | 347 | 2 | 27 | 1562 | 98.50% | 92.78% | 0.13% |
| 6 | 393 | 4 | 3 | 1664 | 99.66% | 99.24% | 0.24% |
| 7 | 571 | 1 | 5 | 1545 | 99.72% | 99.13% | 0.06% |
| 8 | 504 | 3 | 5 | 1576 | 99.62% | 99.02% | 0.19% |
| 9 | 582 | 2 | 7 | 1636 | 99.60% | 98.81% | 0.12% |
| 10 | 674 | 5 | 5 | 1299 | 99.50% | 99.26% | 0.38% |
| | | | | *Media* | *99.31%* | *98.29%* | *0.44%* |

TABLA III

Resultados del IDS propuesto en valores de Precisión, Tasa de Detección y Tasa de Falsas Alarmas.

Como se pudo apreciar los resultados obtenidos a través del Sistema de Detección de Intrusos propuesto son bastante eficientes para pruebas iniciales con una tasa clasificado correctamente alta y podemos ver que la precisión de 99,31%, la tasa de detección de 98,29%, y la tasa de falsas alarmas son inferiores al 2%, lo que significa que la detección tiene identificación aceptable. El coste computacional es uno de el factor más importante a considerar, tanto en el entrenamiento del algoritmo y en el proceso de la detección del comportamiento del usuario, como presenta la figura 4 y 5.

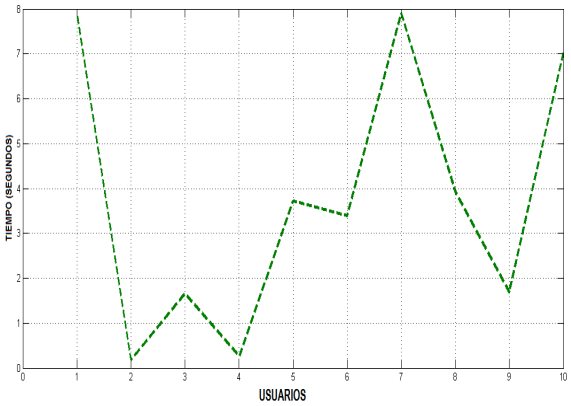



Fig. 4. Coste computacional en el entremamiento del IDS

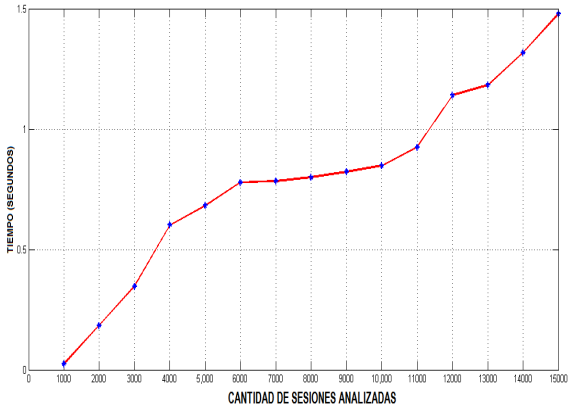



Fig. 5. Coste computacional en la detección del IDS

Dicho desempeño del IDS en una estación de trabajo es bajo lo que es un punto a favor a la propuesta. El tiempo promedio para detectar relativamente mínimo ya que con solo registros de 2 años facilita que la la rapidez de la detección pero no se puede obtener un perfil totalmente eficiente. El costo del entranemiento no es un factor de riesgo porque el entrenamiento se realiza solo una vez en períodos semanales o mensuales lo que hace al sistema se pueda adaptar al comportamiento del usuario.

## VI. Conclusiones y Trabajos Futuros

Este trabajo se ha propuesto y validado un sistema de detección de intrusos con un enfoque basado en la Sis-

temas inmunes artificiales implementado en un sistema informático, lo que permitió que el trabajo pueda describir de una forma aproximada la creación de perfiles de usuario representados por el modelo de datos. La ventaja de nuestro modelo es la generación de comportamientos anómalos, aplicando selección negativa, a partir de comportamientos normales de cada uno de los usuarios. Esto permite que el perfil de cada usuario sea único y dinámico. El IDS se aplicó para comprobar el análisis de sensibilidad y definir el mejor rendimiento de parámetros de detección. Esta forma ofrece un mejor equilibrio entre la tasa de detección y tasa de falsas alarmas, también comprueba su adaptabilidad al comportamiento humano para la detección aplicando el enfoque propuesto.

Como líneas de trabajos fututos serán la implementación de identificación de comportamientos anómalos en secuencias temporales de las tareas ejecutadas por el usuario aplicando reyes bayesianas u otras técnicas de inteligencia artificial. También,algoritmos de optimización y big data para mejorar el desempeno de las detecciones y reducir el tiempo de procesamiento y respuesta para ser implementado de manera online y con varios sistemas a la vez. Además de realizar la aplicación a diferentes áreas como la medicina y detección de problemas en sistemas industriales.

## Agradecimientos

Este trabajo ha sido parcialmente financiado por el Ministerio de Educación Superior, Ciencia, Tecnología e Innovación (SENESCYT) del Gobierno de la República del Ecuador bajo la beca "Convocatoria abierta 2011 y 2012".

## Referencias

[1] Heady, Richard, et al. The architecture of a network level intrusion detection system. Department of Computer Science, College of Engineering, University of New Mexico, 1990.

[2] Scarfone, Karen, and Peter Mell. "Guide to intrusion detection and prevention systems (idps)." NIST special publication 800.2007 (2007): 94.

[3] Bace, Rebecca, and Peter Mell. NIST special publication on intrusion detection systems. BOOZ-ALLEN AND HAMILTON INC MCLEAN VA, 2001.

[4] Stavroulakis, Peter, and Mark Stamp, eds. Handbook of information and communication security. Springer Science & Business Media, 2010.

[5] Debar, Hervé, Marc Dacier, and Andreas Wespi. "Towards a taxonomy of intrusion-detection systems." Computer Networks 31.8 (1999): 805-822.

[6] Debar, Hervé, Marc Dacier, and Andreas Wespi. "A revised taxonomy for intrusion-detection systems." Annales des telecommunications. Vol. 55. No. 7-8. Springer-Verlag, 2000.

[7] Kumar, Vipin, Jaideep Srivastava, and Aleksandar Lazarevic, eds. Managing cyber threats: issues, approaches, and challenges. Vol. 5. Springer Science & Business Media, 2006.

[8] Murali, Adithyavairavan, and Madhav Rao. "A survey on intrusion detection approaches." Information and Communication Technologies, 2005. ICICT 2005. First International Conference on. IEEE, 2005.

[9] DasGupta, Dipankar. An overview of artificial immune systems and their applications. Springer Berlin Heidelberg, 1999.

[10] Kephart, Jeffrey O., et al. "Biologically inspired defenses against computer viruses." IJCAI (1). 1995.

[11] Kim, Jungwon, and Peter J. Bentley. "Towards an artificial immune system for network intrusion detection: An investigation of clonal selection with a negative selection operator." Evolutionary Computation, 2001. Proceedings of the 2001 Congress on. Vol. 2. IEEE, 2001.

[12] Hofmeyr, Steven A., and Stephanie Forrest. "Architecture for an artificial immune system." Evolutionary computation 8.4 (2000): 443-473.

[13] Forrest, Stephanie, and Steven A. Hofmeyr. "Immunology as information processing." SANTA FE INSTITUTE STUDIES IN THE SCIENCES OF COMPLEXITY-PROCEEDINGS VOLUME-. Reading, Mass.; Addison-Wesley; 1998, 2001.

[14] Garcia-Teodoro, Pedro, et al. "Anomaly-based network intrusion detection: Techniques, systems and challenges." computers & security 28.1 (2009): 18-28.

[15] Gong, Maoguo, et al. "An efficient negative selection algorithm with further training for anomaly detection." Knowledge-Based Systems 30 (2012): 185-191.

[16] Aziz, Amira Sayed A., et al. "Detectors generation using genetic algorithm for a negative selection inspired anomaly network intrusion detection system." Computer Science and Information Systems (FedCSIS), 2012 Federated Conference on. IEEE, 2012.

[17] Peng, Lingxi, et al. "Dynamically real-time anomaly detection algorithm with immune negative selection." Applied Mathematics & Information Sciences 7.3 (2013): 1157-1163.

[18] Mandumula, Kranthi Kumar. "Knuth-Morris-Pratt Algorithm." Poslední zmena 18 (2011).

# Sistema Inmune Artificial para la Detección de Comportamientos Anómalos de Usuarios en Sistemas Informáticos

César Byron Guevara Maldonado, Matilde Santos P, Victoria López
cesargue@ucm.es, msantos@ucm.es, vlopez@fdi.ucm.es

Facultad de Informática, Universidad Complutense de Madrid

**Abstract.** Esta este trabajo presenta el desarrollo y aplicación inicial de un algoritmo la dinámico de detección de anomalías en sistemas de informacón gubernamental. Para el desarrollo del algoritmo due necesario utilizar información del comportamiento de varios usuarios que ejecutan multitud de tareas dentro del sistema durante un tiempo determinado. Las principales aportaciones de este trabajo son: el proceso de desarrollo de un modelo de datos dinámico, clasificación de tareas más ejecutadas y las poco ejecutadas por cada usuario, además, la aplicación de un algoritmo de sistemas inmunes artificiales como la selección negativa para la generación de secuencias detectores binarios de tareas anómalas para la posterior detección de comportamientos peligrosos de los usuarios aplicando el algoritmo de búsqueda local Knuth Morris Pratt (KMP).

## 1 Introducción

El sistema inmune natural ha tenido gran éxito en la protección de los seres vivos contra una amplia variedad de patógenos como lo presenta Tizard en [1]. El Sistema Inmune Artificial (AIS) siempre ha sido una inspiración para el desarrollo de modelos computacionales para resolver diversos problemas incluyendo el diagnóstico de fallas, detección de virus y detección de fraudes hipotecarios como lo presenta Dasgupta en[2]. Específicamente, el sistema inmunológico utiliza dos principios fundamentales que se incluyen en la teoría de redes inmunes que son los mecanismos de selección negativa y los principios de selección clonal. En todos estos ámbitos, la detección de anomalías es un área de investigación importante en la aplicación de un sistema inmune artificial (AIS) que ha sido probado con excelentes resultados. El objetivo más importante de la detección de anomalías es detectar comportamientos no frecuentes, mal uso y abuso de los sistemas informáticos por parte de los usuarios del sistema o intrusos internos o externos. En la actualidad existen muchos sistemas de detección de intrusiones basados en red (IDS) y en host (HIDS) que se han desarrollado utilizando diversos enfoques.

Este trabajo se centra en el estudio de la detección de anomalías en un sistema informático aplicando un conjunto de detectores anómalos a los datos de comportamiento de los usuarios. De ahí que la detección de anomalías de los

datos en secuencias de ejecución de tareas, ya que es un tema importante de investigación. Existen muchos trabajos sobre esta área de investigación de las técnicas de detección de anomalías como Lazarevic en [3] y Chandola en [4] que buscan objetos anormales que son diferentes de los objetos normales.

Estamos interesados en la detección de anomalías en secuencias discretas para encontrar posibles intrusiones, fraudes, fallos o la fuga de datos. La detección de anomalías para las secuencias discretas no es una tarea fácil, ya que implica el análisis de la secuencia normal de datos para detectar posibles anomalías. El principal problema para detectar anomalías en secuencias de tareas es la gran cantidad de datos para el procesamiento y el desarrollo del modelo dinámico adaptable a la conducta humana.

El documento está organizado de la siguiente manera: la sección 2 describe brevemente el AIS para la detección de anomalías, detección de fuga de datos y detección de intrusos que son las obras más importantes en esta área que han sido presentadas por varios autores. La sección 3, muestra los métodos y materiales utilizados como el algoritmo de selección negativa, secuencia algoritmo de búsqueda, además, la forma de aplicación para la detección de anomalías. En la sección 4, se detallan los objetivos de la investigación y las aportaciones al mismo. Sección 5 describe del experimento realizado en este trabajo y un análisis de los resultados. Por último, las conclusiones se han extraído de este artículo y trabajos futuros.

## 2 Trabajos relacionados

Varios trabajos han sido desarrollados para detectar anomalías en la operación sobre información de llamadas al sistema tal como se presenta en Forrest en[5] y Gao en [6].

En el trabajo presentado por Helman en [7] propone un ranking de cada secuencia comparando la frecuencia que se conoce va a ocurrir las trazas normales y la frecuencia con la que se espera que ocurran las intrusiones. En el artículo de Javitz en [8] utiliza las distribuciones estadísticas para definir el comportamiento normal y anormal.

Por otra parte, los trabajos dirigidos a la detección de anomalías presentado por Mykerjee [9] describe los muchos perfiles de las actividades normales de los usuarios, los sistemas, los recursos del sistema, tráfico de red, servicios y detecta intrusiones mediante la identificación de desviaciones significativas de los patrones de comportamiento normales observados a partir de perfiles.

## 3 Métodos y Materiales

En esta sección se presentan los principales algoritmos aplicados en nuestro trabajo. Los algoritmos utilizados son:

- Algoritmo de selección negativa (NSA).
- Algoritmo Knuth Morris Pratt (KMP).

### 3.1 Sistema Inmune Artificial con Selección Negativa

El AIS se basa en el funcionamiento del sistema inmune humano, que es capaz de reconocer en una forma muy eficiente cualquier agente patógeno. Esta es una teoría inmune clásica para entender el sistema inmunológico como un sistema que identifica a self (agente propio) o a non self(agente patógeno)[12].

### 3.2 Algoritmo de Knuth Morris Pratt

El objetivo principal de este algoritmo es encontrar una cadena dentro otra cadena. En un patrón $P$ para cada posición $i$, $spi(p)$ se dice que es la longitud del sufijo más largo de $P[1, 2i]$, que coincide con el prefijo $P$. Es similar a navegar dentro de la cadena y que realiza sus comparaciones de izquierda a derecha. También calcula los desplazamientos máximos posibles de izquierda a derecha para el patrón $P$, como lo presenta Gawrychowski en [11].

## 4 Algoritmo de Detección de Anomalías

### 4.1 Datos del estudio y estructura de datos

Los datos seleccionados para este trabajo fueron recogidos de un sistema de gobierno de la República de Ecuador. Esta información es confidencial y estos datos se han codificado para salvaguardar la integridad de su información. El conjunto de datos utilizados en este trabajo se ha generado mediante la captura de las tareas ejecutadas por los usuarios. Este conjunto de datos consta de diez conjuntos diferentes de información sobre la ejecución de las tareas en sesiones de cada usuario. Las sesiones grabadas durante el período de 2011 a 2013 son 15.571 y el conjunto de sesiones de pruebas con sesiones anómalas son 5312.

El problema principal es muy difícil distinguir entre comportamiento normal y comportamiento anormal porque las tareas ejecutadas por los usuarios depende de la carga de trabajo o asignaciones de cada usuario. Este sistema tiene siete tablas principales en la base de datos ($T1, T2..., T7$), en esta base de datos es posible ejecutar las cuatro operaciones sql que son: Insertar(1), Modificar (2), Eliminar (3) y Buscar(4). Las tareas están codificadas en 28 tareas genéricas más el inicio de sesión. Cada tarea tiene un código "Tsk" más un número para identificar cada tarea.

El comportamiento del usuario es dinámico y diferente de otro usuario, por esta razón tenemos que crear una estructura de datos que describa el comportamiento humano dentro de un sistema informático. Esta estructura está formada por un inicio de sesión (Tsk0) y una o más tareas ejecutadas en forma secuencial. Esta estructura no tiene tamaño fijo como se muestra en la figura 1.

El conjunto de tareas se define como $Tsk = \{Tsk_1, Tsk_2, Tsk_3, ..., Tsk_n\}$ donde $n$ es el $n$úmero de la tarea. El conjunto de posibles tareas ejecutadas por el usuario $U$ se define $Tsk^U = \{Tsk_1, Tsk_2, Tsk_3, ..., Tsk_m\}$, donde $m$ es el número de tareas ejecutadas por un usuario específico. Una sesión de usuario $S$ contiene una o varias tareas ejecutadas por el usuario y se define como $S^U =$

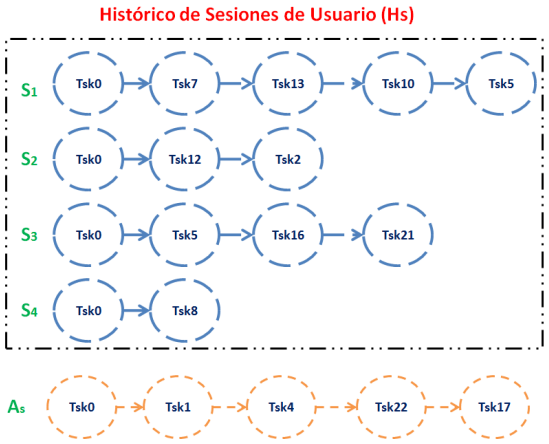



**Fig. 1.** Estructura de datos del comportamiento de usuario en un sistema informático.

$\{Tsk_1^U, Tsk_2^U, Tsk_3^U, ..., Tsk_m^U\}$. El conjunto de posibles sesiones se define como $Hs^U = \{S_1^U, S_2^U, S_3^U, ..., S_d^U\}$, donde $d$ es el número de sesiones realizadas por el usuario, estas sesiones se llama "Sesiones pasivas" donde $Hs^U \subset S^U$. Una nueva sesión de usuario se define como Sesión Activa $As^U$, la cual puede contener tareas $Tsk^U$ como tareas que no ha sido ejecutadas por el usuario $U$.

### 4.2 Objetivo del Experimento

La detección de anomalías AIS que se propone describe las siguientes contribuciones:

- Detectar comportamientos anómalos de los usuarios dentro de los sistemas informáticos.
- Creación de una estructura dinámica de tareas del comportamiento de los usuarios.
- Identificar de las tareas más ejecutadas por el usuario para detectar el comportamiento anormal.
- Generar secuencias de tareas anómalas utilizando el algoritmo de selección negativa como en lo presenta Kim, para detectar comportamientos anómalos utilizando la formulación de Forrest.
- Aplicar el algoritmo KMP para detectar el comportamiento anómalo de los usuarios.

En la siguiente sección se describe el proceso del algoritmo propuesto para aplicar la selección negativa con la nueva estructura de datos.

### 4.3 Proceso del algoritmo propuesto

La detección secuencial de tareas anormales podría resumir en los siguientes tres pasos:

**Paso 1. Ranking de usuarios tareas más ejecutadas** En esta etapa el objetivo es agrupar las tareas ejecutadas (Tsk) en 3 subgrupos, estos son: tareas más ejecutadas (ME), tareas ejecutadas por debajo de la media (MDE) y las tareas no ejecutadas (NE). Para hacer esta agrupación es necesario utilizar el conjunto de datos $Hs^U$, debido a que cada usuario tiene grupos específicos de tareas $Tsk^U$ acuerdo a su comportamiento. El grupo ME establece las tareas más populares ejecutados por el usuario. Estas tareas son buena parte del conjunto $Hs^U$.

El segundo grupo MDE son las tareas ejecutadas con poca frecuencia por el usuario, este grupo se identifica como tareas complementarias y no proporciona mucha información del comportamiento del usuario.

El último grupo NE son las tareas no ejecutadas por el usuario y estas tareas son probablemente las tareas anómalas.

Este procedimiento realiza un recuento de las tareas ejecutadas $Tsk^U$ de las sesiones $S^U$, definido como $XTask_n$, donde $n$ es el número de la tarea. Esos valores se deben ordenar en forma descendente. Después, el número total de ejecuciones de cada una de las tareas $Task_n$ se utiliza para calcular la media de los valores definidos como $Avg_{Med}$. Si $Xtask_n > Avg_{Med}$ entonces esta tarea $Task_n$ será etiquetada como ME, de lo contrario esta tarea será etiquetada como MDE. Si $Xtask_n$ es igual a cero, es decir que no contiene una sola ejecución de $Xtask_n$ se etiqueta como NE. Finalmente, la tabla obtenida se almacena en la memoria para llevar a cabo nuevas actividades en los siguientes pasos.

**Paso 2. Proporción de las tareas en las sesiones de usuario** En el segundo paso, el procedimiento es determinar el porcentaje de tareas ME, MDE y NE en las sesiones $S^U$ del conjunto de datos $Hs^U$, como también de la sesión a ser evaluada $As^U$.

Este conjunto de valores corresponde al porcentaje de cada tarea de usuario ejecutado en el conjunto de datos $Hs^U$ definidos como $Pr(Me^b_m)$ para las tareas más ejecutadas ME. Y $Pr(Med^b_m)$, para las tareas MDE, donde $b$ es el número de la tarea y $m$ es el número de la sesión a ser evaluada. El cálculo del valor de porcentaje de tareas ME y tareas MDE para una sesión $S^U$ o $As^U$ se aplica las siguientes eciaciones:

$$Pr(Me^b_m) = \frac{N_{Me}}{N_{tasks}} 100\%$$

$N_{Me}$ es el número de tareas ME dentro de una sesión ($S^U$ o $As^U$).

$$Pr(Med^b_m) = \frac{N_{Med}}{N_{tasks}} 100\%$$

$N_{Med}$ es el número de tareas MED dentro de una sesión ($S^U$ o $As^U$).

$N_{tasks}$es el número total de tareas dentro de una sesión ($S^U$ o $As^U$).

Posteriormente el procedimiento calcula un valor máximo y mínimo de $Pr(Me^b_m)$, definido como $Rmax_{Me}$ y $Rmin_{Me}$. También el valor máximo y mínimo de $Pr(Med^b_m)$ que está definido como $Rmax_{Med}$ y $Rmin_{Med}$.

Para determinar el porcentaje de tareas dentro $As^U$ es necesario aplicar la ecuaciones 1 y 2 obteniento los porcentajes $Pr(Med^b)^{As^U}$ y $Pr(Me^b)^{As^U}$, en el

caso de que existiese tareas NE se determina un umbral $Pr(Ne^b)^{As^U}$ mayor a cero y menor a 1 para identificar su presencia en la siguiente parte del algoritmo. Con estos umbrales en los siguientes pasos podemos evaluar una o más sesiones $As^U$ para conocer si están dentro de los parámetros de un comportamiento normal de usuario.

**Paso 3.Generación de secuencias anormales** La generación de secuencias anomalas propuesta consta de 3 fases principales:

- Fase 1: Generar aleatoriamente la población inicial $B_0$. Esta fase se ilustra en la figura 2a.
- Fase 2: Definición de la porción de espacio normal (itself). Esta fase se ilustra en la figura 2b.
- Fase 3: La eliminación de todos los detectores que se solapan en la región normal (itself). Esta fase se ilustra en la figura 2c y obtención de detectores anómalos $Dan$, como lo presenta la figura 2d.

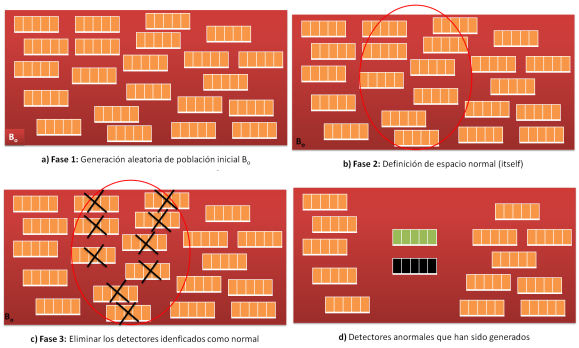



**Fig. 2.** Fases del sistema de detección de anomalías.

**Fase 1. Generación aleatoria de población inicial** $B_0$

En esta fase se presenta el uso de un sistema inmunológico artificial mediante la aplicación de selección negativa para generar secuencias de tareas anómalas de conjunto de datos históricos $Hs^U$ con las proporciones obtenidas en el apartado anterior que son $Pr(Me^b)^{As^U}$, $Pr(Med^b)^{As^U}$ y $Pr(Ne^b)^{As^U}$. Para generar la población inicial $B_0$ se debe cuenta las siguientes condiciones:

Si $Rmax_{Me} > Pr(Me^b)^{As^U} \geq Rmin_{Me}$, $Rmax_{Med} > Pr(Med^b)^{As^U} \geq Rmin_{Med}$ y $Pr(Me^b)^{As^U} == 0$ entonces esta información será utilizada para generar $Bo$ con el mismo porcentaje de datos ME y MDE, es decir $B_0 \leftarrow ME \cup MDE$.

Si $Pr(Me^b)^{As^U} == 1$ entonces esta información será utilizada para generar $B_0$ unicamente con los datos ME, es decir $B_0 \leftarrow ME$.

Si $Pr(Med^b)^{As^U} == 1$ entonces esta información será utilizada para generar $Bo$ únicamente con los datos MED, es decir $B_0 \leftarrow MED$.

En el caso de que $Rmax_{Me} > Pr(Me^b)^{As^U} \geq Rmin_{Me}$, $Rmax_{Med} > Pr(Med^b)^{As^U} \geq Rmin_{Med}$ y $Pr(Me^b)^{As^U} > 0$ entonces esta información será utilizada para generar $B_0$ con el mismo porcentaje de datos ME, MDE y NE, es decir $B_0 \leftarrow ME \cup MDE \cup NE$.

En la generación de población inicial $B_0$ el tamaño de la cadena de anticuerpos será binaria, por lo que es la cadena con un tamaño mínimo de tareas que pueden presentar información para la detección. Para determinar el número de detectores de $n_D$ en el espacio $Bo$ es necesario calcular el número de combinaciones con la siguiente ecuación $n_D = \frac{c!}{(c-2)!}$, donde $c$= number of elements of tasks $S_{An}$.

**Fase 2: Definición de la porción de espacio normal (itself)**

En esta fase compara los anticuerpos generados en la fase anterior con secuencias históricas $Hs^U$ utilizando el algoritmo KMP. En este proceso se determinan anticuerpos con comportamientos normales, es decir, se examina uno por uno la secuencias binarias de tareas $Tsk_n$ generadas en la poblacion inicial $B_0$ con toda la cadena de tareas de $S^U$ existentes en $Hs^U$, comparadolas con cada detector de $D_n$. Si dentro de las sesiones $Hs^U$ se determina que existe un detector de $D_n$ este detector será marcado como normal, de lo contrario se identifica como $D_{an}$ anormal. Con los detectores normales identificados continua a la siguiente fase.

**Fase 3: La eliminación de todos los detectores que se solapan en la región normal (itself)**

En esta fase se eliminan todos los detectores identificados como normales $Dn$ y detectores con un comportamiento anormal $D_{an}$ se conservan. Finalmente los detectores anómalos continuan al siguiente paso del sistema de detección, para identificar anomaías en secuencias activas $As^U$. El algoritmo de selección negativa que se utiliza en este trabajo emplea la función de emparejamiento r-contigua en una sesión activa $As^U$ del usuario. En el experimento, el umbral de coincidencia se define como un número de detectores que se fijan dependiendo del error de falsos negativos generados en el entrenamiento, como se muestra en el trabajo de [10].

**Paso 4. Detección de secuencias anómalas** En este último apartado, se analiza la secuencia activa $As^U$ y se compara con todo el conjunto de detectores anómalos $D_{an}$ aplicando el algoritmo KPM con un umbral de estimulación definido como $Sth$. Si $Sth$ es igual a TRUE el detector se activa y el antígeno lo clasifica como anómalo, de lo contrario, se clasifica como normal.

## 5 Resultados

Para la prueba del algoritmo de detección de anomalías se utiliza la información diez usuarios de un sistema informático real. Durante el estudio se observó que un factor principal para el óptimo funcionamiento de los algoritmos de AIS

es la estructura correcta de los datos reales de comportamiento del usuario. Estructura incorrecta impide la detección de algunos comportamientos anómalos la diferencia entre la secuencia de datos con y sin anomalías puede ser muy sutil y el algoritmo no será capaz de detectarlos. Los resultados de las pruebas de la aplicación del algoritmo AIS muestran la tasa de Clasificados Correctamente (CC), tasa Clasificados Incorrectamente (IC). También presenta la detección Tiempo promedio (segundos) para cada usuario del estudio. Estos resultados se ilustran en la tabla 1.

| USR | **CC %** | **IC %** | **Tiempo (s)** |
|---|---|---|---|
| 1 | 96.945 | 3.055 | 71.23 |
| 2 | 95.697 | 4.303 | 81.56 |
| 3 | 96.276 | 3.724 | 65.98 |
| 4 | 97.318 | 2.682 | 77.37 |
| 5 | 94.878 | 5.122 | 76.82 |
| 6 | 95.192 | 4.808 | 68.84 |
| 7 | 96.062 | 3.938 | 69.56 |
| 8 | 95.71 | 4.29 | 82.55 |
| 9 | 96.198 | 3.802 | 74.88 |
| 10 | 95.058 | 4.942 | 86.41 |
| **Media** | **95.927** | **3.911** | **74.995** |

**Table 1.** Tasa de detección y tiempo de detección para cada usuario utilizando el algoritmo AIS

Detección e identificación de comportamiento anómalo y conductas anómalas pueden generalizarse como lo siguiente: verdaderos positivos (TP), verdaderos negativos (TN), falsos positivos (FP) y falsos negativos (FN).

En la actualidad, el sistema de detección de anomalías requiere una precisón de la clasificación, tasa de detección y tasa de falsas alarmas para evaluar el desempaño del algoritmo. Este análisis de detección de anomalías se ilustran en la tabla 2.

Finalmente, los resultados medios obtenidos a través de este trabajo son buenos con una tasa clasificado correctamente alta y podemos ver que la precisión (95,928%), la tasa de detección (91,422%) y la tasa de falsas alarmas son inferiores a 2,483%, lo que significa que la detección tiene identificación aceptable. El coste computacional es uno de el factor más importante a considerar, tanto en el entrenamiento del algoritmo y en el proceso de la detección del comportamiento del usuario. El tiempo promedio para detectar es muy alto y este es el gran problema de aplicar en un sistema informático real. El costo del entranemiento no es un factor de riesgo porque el entrenamiento se realiza sólo una vez en cada sistema en la fase inicial. El costo en la detección es un factor de riesgo para AIS, como la detección se realiza varias veces mientras el usuario ejecuta tareas en el sistema informático.

| USR | TP | TN | FP | FN | Precisión | %Detección | False Alarms |
|---|---|---|---|---|---|---|---|
| 1 | 689 | 1215 | 45 | 15 | 96.945 | 97.869 | 3.571 |
| 2 | 448 | 1309 | 56 | 23 | 95.697 | 95.117 | 4.102 |
| 3 | 406 | 1585 | 36 | 41 | 96.276 | 90.828 | 2.220 |
| 4 | 539 | 1965 | 30 | 39 | 97.318 | 93.252 | 1.503 |
| 5 | 319 | 1515 | 52 | 47 | 94.878 | 87.158 | 3.318 |
| 6 | 332 | 1628 | 53 | 46 | 95.192 | 87.8307 | 3.1528 |
| 7 | 497 | 1552 | 36 | 48 | 96.062 | 91.193 | 2.267 |
| 8 | 471 | 1537 | 36 | 54 | 95.710 | 89.714 | 2.289 |
| 9 | 557 | 1594 | 31 | 54 | 96.198 | 91.162 | 1.908 |
| 10 | 644 | 1241 | 35 | 63 | 95.058 | 91.089 | 2.743 |
| | | | | **Media** | **95.928** | **91.422** | **2.483** |

**Table 2.** Resultados de la detección de comportamiento anómalo de los usuarios con el algoritmo de AIS.

## 6 Conclusiones y trabajos futuros

Este artículo se ha propuesto y validado un sistema inmune Artificial (AIS) con un enfoque basado en la detección de anomalías en el sistema informático inspirado en la selección negativa de las secuencias de tareas del usuario. El AIS se aplicó para comprobar el análisis de sensibilidad y definir el mejor rendimiento de parámetros de detección. La eficiencia de la AIS propuesto, que se muestra en la tabla 1 y 2, es notable y capaz de proporcionar una mejor tasa de detección con el mismo o en algunos casos menor tasa de falsas alarmas para varias tareas ejecutadas en el sistema informático. Los resultados presentan que el AIS propuesto automatiza y mejora la etapa de detección de anomalías con la aplicación de selección negativa en tareas específicas para generar secuencias de tareas anómalas. Esta forma ofrece un mejor equilibrio entre la tasa de detección y tasa de falsas alarmas, también comprueba su adaptabilidad al comportamiento humano para la detección aplicando el enfoque propuesto.

## References

1. Tizard, I. R. (1992). Immunology, an introduction. Saunders College Publishing.
2. Dasgupta, D. (1999). An overview of artificial immune systems and their applications. In Artificial immune systems and their applications (pp. 3-21). Springer Berlin Heidelberg.
3. Lazarevic, A., Ertoz, L., Kumar, V., Ozgur, A., and Srivastava, J. (2003, May). A Comparative Study of Anomaly Detection Schemes in Network Intrusion Detection. In SDM (pp. 25-36).
4. Chandola, V., Banerjee, A., and Kumar, V. (2009). Anomaly detection: A survey. ACM Computing Surveys (CSUR), 41(3), 15.
5. Warrender, C., Forrest, S., and Pearlmutter, B. (1999). Detecting intrusions using system calls: Alternative data models. In Security and Privacy, 1999. Proceedings of the 1999 IEEE Symposium on (pp. 133-145). IEEE.

6. Gao, B., Ma, H. Y., and Yang, Y. H. (2002). Hmms (hidden markov models) based on anomaly intrusion detection method. In Machine Learning and Cybernetics, 2002. Proceedings. 2002 International Conference on (Vol. 1, pp. 381-385). IEEE.
7. Helman, P., and Bhangoo, J. (1997). A statistically based system for prioritizing information exploration under uncertainty. Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on, 27(4), 449-466.
8. Javitz, H. S., Valdes, A., and NRaD, C. (1993). The NIDES statistical component: Description and justification. Contract, 39(92-C), 0015.
9. Mykerjee, B. L. Heberlein T., and K. Levitt N.,". Network Intrusion Detection, 14-26.
10. Forrest, S., Javornik, B., Smith, R. E., and Perelson, A. S. (1993). Using genetic algorithms to explore pattern recognition in the immune system. Evolutionary computation, 1(3), 191-211.
11. Gawrychowski, P., Jez, A., & Jez, L. (2014). Validating the Knuth-Morris-Pratt failure function, fast and online. Theory of Computing Systems, 54(2), 337-372.
12. Li, Dong, Shulin Liu, and Hongli Zhang. "Negative selection algorithm with constant detectors for anomaly detection." Applied Soft Computing 36 (2015): 618-632.

# Intrusion detection with neural networks based on knowledge extraction by decision tree

César Guevara, Matilde Santos, Victoria López

Department of Computer Architecture and Automatic Control
University Complutense of Madrid
28040-Madrid, Spain
cesargue@ucm.es, msantos@ucm.es, vlopez@fdi.ucm.es

**Abstract.** Detection of intruders or unauthorized access to computers has always been critical when dealing with information systems, where security, integrity and privacy are key issues. Although more and more sophisticated and efficient detection strategies are being developed and implemented, both hardware and software, there is still the necessity of improving them to completely eradicate illegitimate access. The purpose of this paper is to show how soft computing techniques can be used to identify unauthorized access to computers. Advanced data analysis is first applied to obtain a qualitative approach to the data. Decision tree are used to obtain users' behavior patterns. Neural networks are then chosen as classifiers to identify intrusion detection. The result obtained applying this combination of intelligent techniques on real data is encouraging.

**Keywords:** Intrusion detection, pattern recognition, behavioral profile, security, decision tree, neural networks.

## 1 Introduction

Information networks have become the principal communication channel around the world. Intrusion Detection Systems (IDS) have the function to monitor a network activity for any possible intrusions that could potentially endanger the integrity of communication or information of the network [1]. These systems must be equipped with important features: multiple attack stability, quick turnaround time, and maintenance of network connection[2].

To address this problem, our approach is to obtain behavioral patterns of the authorized users in an automatic way, in order to control the access to those systems. We have applied decision trees for the identification of the user dynamic profiles and then Artificial Neural Networks (ANN)to classify and detect intrusions. This behaviour modeling demands a previous pre-processing and analysis of the available information. These steps provide knowledge of the behavior of the users that will help to test the final results of the detection system [3].

In the literature this problem has been addressed by artificial intelligent techniques but, as far as we know, there are only two close related works. The paper by [4] compares the performance of decision trees (C4.5) and neural network (ANN model),

independently, in detecting network attacks in terms of detection accuracy, detection rate, and false alarm rate. They used KDD Cup 99 data as benchmark. According to the results, decision trees are more effective in intrusion detection than neural networks. The authors of [5] claim that most of the data mining methods applied to IDS, including rule-based expert systems, are not able to successfully identify the attacks which have different patterns from expected ones. Nevertheless, they propose a method based on the combination of decision tree algorithm and Multi-Layer Perceptron (MLP) neural network which is able to identify attacks with high accuracy and reliability. Other works have also applied only neural networks, such as [6] and [7], or only decision trees [8].

In this paper we take advantage of both methods, using decision trees to obtain the behavioral pattern of the users and then neural networks to classify the activity of the user into the computer system as normal or intrusion. Moreover, we have tested it on real data with good results.

The paper is organized as follows. In section 2we describe the pre-processing and analysis of the data and the generation of a multidimensional data model. In section 3, decision trees are applied to represent the behaviour of the users and neural networks to detect anomalous activities.Section4 shows a simulation tool developed to apply the IDS. Conclusions and future work end the paper.

## 2 Pre-processing and data analysis

The information has been provided by a public institution of the Republic of Ecuador, where different users interact with the system and perform a variety of tasks. We are dealing with 18 different users along three years, 2010, 2011 and 2012.The database is made up of 118.067 records (operations on files) where 116.952 are validated as authorized operations and 1.115 activities have been manually classified as anomalous [3, 11].

The first step involves the selection of the most relevant features of the users' access. Even more, as the information is usually stored in different databases we have first integrated the information and put it together into one repository. This will make easier the tasks of sorting, cleaning and retrieving the information that will be used for the classification. The subsequent analysis and the success of the detection strongly depend on these initial phases.

### 2.1 Analysis of variables

The variables should reflect the users' behavior when they are working on the computer system and describe their activities in a detailed manner. In the database there are already several attributes that allow us the definition of a behavioral pattern of a user, such as day of the week, computer used, time, session length, etc. Table 1 shows some of these features as follows: Day of the week, from 1 to 7; hour, from 8 to 20 (every hour); operation, from 1 to 4 (insert, update, delete and find); repository that is acceded, from 1 to 7.

| Attribute | Data Type | Description |
|---|---|---|
| ***Logging*** | DateTime | Day and hour of login |
| ***Ip*** | Varchar | IP address of the workstation |
| ***Country*** | Varchar | Country |
| ***Time*** | DateTime | Day and hour specific task is carried out |
| ***Database accessed*** | Varchar | Table of database that has been acceded |
| ***Operation*** | Varchar | Operation performed in the database |
| ***User*** | Varchar | User performing the activities in the database |

**Table 1.**Attributes of the database that describes the behaviour of the user.

Figure 1 shows, as an example, the activity of the user, that is, number of times he has performed each of the operations insert, update, and delete. The same analysis has been carried out with other variables. The behaviour of a user has been recorded: number of times the user performs a specific operation on a database every day of the week, hour of access, and time he spends doing that operation, number of accesses, IP address of the workstation from which operations are performed, etc.

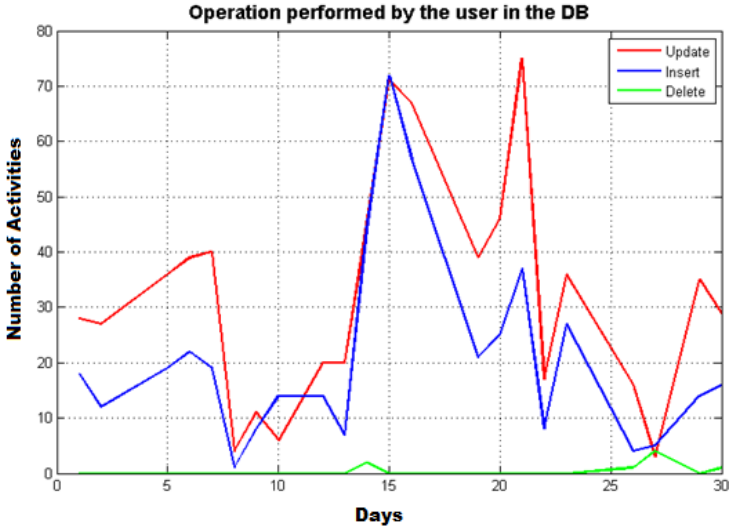



**Fig.1.**Operations performed by a user in the DB: Update, Delete and Insert.

## 2.2 Multidimensional model of the data repository

Several databases have been grouped into one call “tfmdata”. This database is multi-dimensional and every dimension describes a set of relevant attributes regarding the access of each user to that information system. The result is what is called OLAP

(Online Analytical Processing) [9]. OLAP is used to focus on important features, identify exceptions, o finding iterations.

Four dimensions or variables have been taken into account in the OLAP model: time, user, activity, and workstation (Figure 2). Different tables are also associated to every user with information regarding the affiliation, department, ranking, position, etc. An activity is defined by a pair: operation and database. Workstation is just the IP and the type of device. The information was decomposed into the variables of an only database to integrate and unify it. This allows us to save space as some attributes had a very large format but little information.

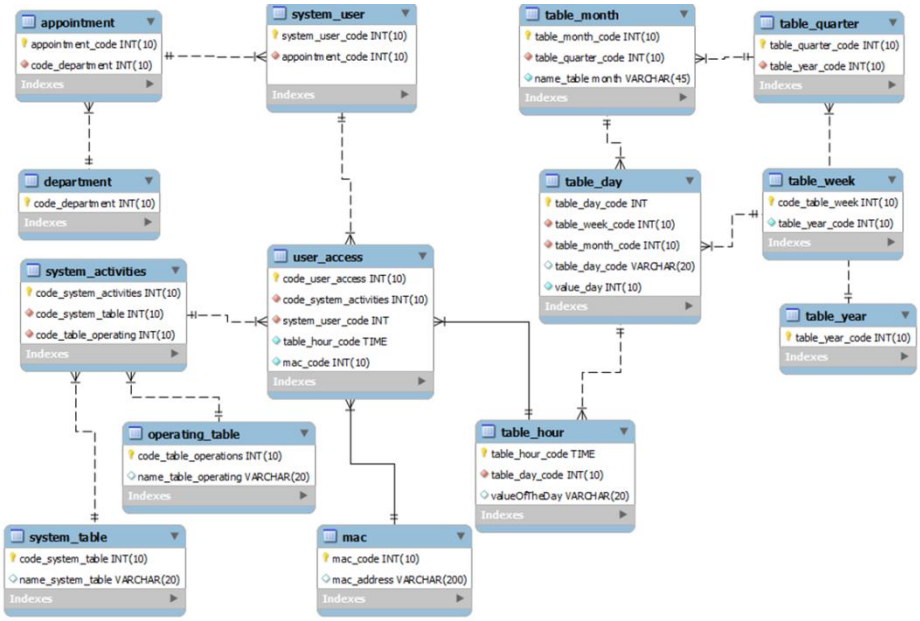



**Fig.** 2.OLAP model.

## 2.3 Exploratory data analysis

Once all the data are integrated into the OLAP repository, the following step is to analyze the characteristics of the attributes. It may be interesting to obtain some properties such as the maximum, minimum and average values and to represent them graphically. To perform this analysis in an easy, efficient and automated way the tool WEKA and Matlab were used. The result is called mining view and it will be very useful to perform the visual data mining and therefore to get an idea of the possible patterns, the relevant data, etc. This tool scans the data analysis algorithms and gives relevant information of each variable.

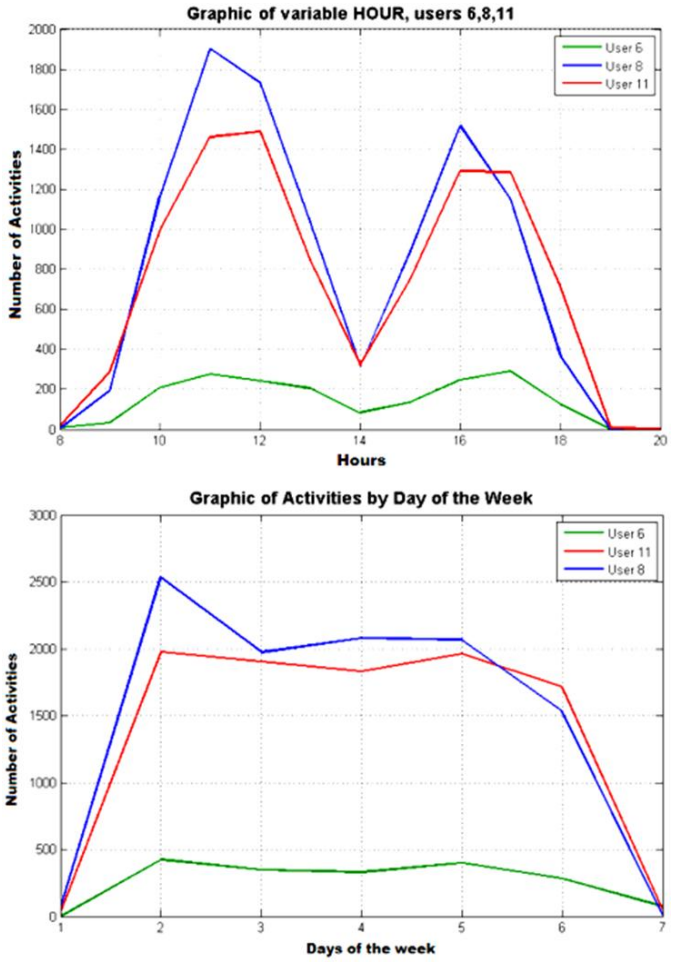



**Fig. 3.**a) Variable "hour", b) Variable "day of the week", for user 6 (green), 11 (red), 8 (blue).

Figure 3 shows how the behavior of users #6, #8 and #11 is quite different regarding hours, days, tables, number of operations they perform, etc. For example, at a specific hour of a week day, let say 5 (Friday), each user can be doing a different operation on a different table. Other features are mainly the same every year for all the users, such as the work load. That is why some features that do not give discriminant information have not been considered.

But the behaviour of each authorized users is not always so clear. Indeed, it is quite difficult to identify whether he/she is performing regular tasks or trying to leak information. Daily tasks can include sending out sensitive information. Even more, the same user can present very different behaviour depending on the tasks he is in charge of at that moment. The same user can show very different behaviour a month or another or even in the same period of time but different years.

# 3 Application of intelligent techniques to model users behaviour

Knowledge extraction from the processed data is the basis to define the behavioral pattern of each user [10]. The user profile has to be unique, dynamic and close to reality. For that reason it is important to work with techniques that keep the models interpretable such as with decision trees. The tree represents the expected behaviour of the user. The neural network carries out the final step, the classification. Besides, we are not only determining if the user is an intruder but also the degree of certainty of that prediction. Therefore we have combined these two knowledge-based techniques because the synergy of them allows a more efficient detection. They have been implemented using the software Matlab and the corresponding toolboxes.

## 3.1 Decision tree implementation

Decision trees are useful to express rules that relate multiple attributes and organize the available information in a hierarchical, orderly way [5].Four decision trees have been generated for periods of one, three, six and twelve months for each user to find which one was more useful for the discrimination. Comparing the trees generated for the different users we realized that selecting the period of three months was enough to obtain a quite different structure for each user and at the same time to keep it simple.

The probability associated to each branch and leaf of the data tree is,

$$P(A) = \frac{valA}{N}; \quad P(B) = \frac{valB}{N} \tag{1}$$

$$P(T) = P(A1) * P(B2) * P(A2) * P(B2) * \ldots * P(An) * \mathrm{P}(Bn) \tag{2}$$

Where *N* is the total number of cases, *valJ* is the number of events in branch *i*, *P(i)* the probability of the event in the *i* node, *P(T)* the final probability at *n*, and *n* the total number of nodes at that point of the tree. As an example, one of the two main branches is shown in Figure 4 with the likelihood of an action of an authorized user.

This tree represents the behaviors of the users, but it does not describe the behavior that the user does not normally present. That is, if the behavior of the user does not follow any of those branches, how can it be detected if it is an intrusion or an authorized user doing something different than usual? So it needs to be complemented by another technique such as neural networks in order to give a final classification.

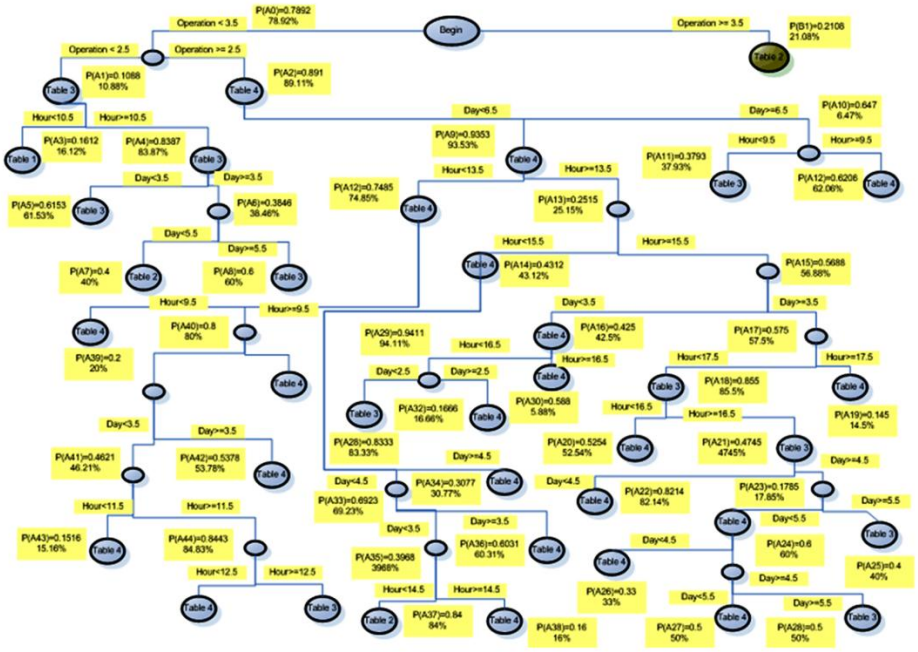



**Fig. 4.**Right branch of the decision tree, user #6, three months.

### 3.2 Final detection of intrusions by Neural Networks

To identify whether a user is an authorized one or an intruder, a Multi-layer Perceptron (MLP) neural network is applied. After trying several configurations, the final neural network is as follows (Figure 5). It has four inputs: day, time, operation, and table. These are the most significant variables and the ones that allow us to better discriminate between users, as explained in section 2. The output is a single neuron that represents the user, that is, a number between 0 and 18 (0 stands for intruder and the rest of the numbers identify the user). There is a hidden layer with 20 neurons.

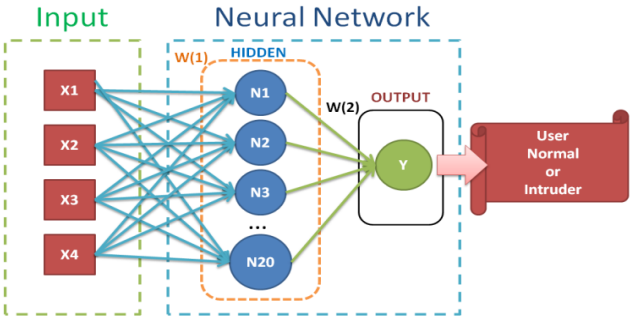



**Fig.** 5.General structure of the neural network of intrusion detection

Real data have been taken from the databases. The vast majority corresponds to authorized users. The available samples have been then divided into training and test sets. Learning (performance) is measured in terms of the mean square error. The network is trained until the error is smaller than 0.001.

Once the neural network has learned how to classify the user that is working on the system, in order to validate it we have tested it with the test samples. The results are summarized in Table 2. It shows the percentages of correct and incorrect classification of the 18 users. The classifier has correctly identified 99.92% of the samples (117.192 out of 118.067), that is, 1860 true positive plus 116.112 true negative samples. There were 35 false positive (normal behaviour detected as intrusion) and 60 false negative (activity identified as normal being intrusion).Besides, Table 2 shows the Attack Detection Rate (ADR) and False Alarm Rate (FAR) of the system.

| True Positive | True Negative | False Positive | False Negative | Accuracy | Error | ADR | FAR |
|---|---|---|---|---|---|---|---|
| 1860 | 116112 | 35 | 60 | 99.92% | 0.08% | 96.88% | 1.85% |

**Table 2.** Classification results for Normal and Intrusive activities of 18 users.

## 4 Simulation tool for intrusions detection

A simulation tool that implements these techniques for intrusion detection has been developed. The input dataset consists in the activities the user has performed into the computer system, and the output is the identification of the user who logged the system and the probability of being an intrusion. The simulation tool gives three different possible outputs.

- Authorized access: the sequence of activities of the user matches with his behavior pattern, so it seems to be an authorized user and has been rightly identified.
- Unauthorized access: the behavior of the user during that computer session is different from expected. A message is displayed warning it is an intrusion.
- Unrecognized pattern: an X is displayed when the system is not able to clearly identify any of the authorized users, but it could still be an authorized user. In this case the result is a warning of possible intrusion.

AGUI (Graphical User Interface) has been designed to introduce the data and show the final decision (Figure 6). On the left window, the information about the login of the system is shown (id of the user, day, hour, operation, table, etc.) In this case, it corresponds to user #9, Thursday, 06:00h, operation 2, on table 3. The decision tree has generated a profile for this user that is shown at the bottom left window (green, hour; light blue, operation; red, table; blue, day). The neural network is then applied to identify which user corresponds to the present activity. The profile of the user that corresponds to the activity is shown at the bottom of the right window.

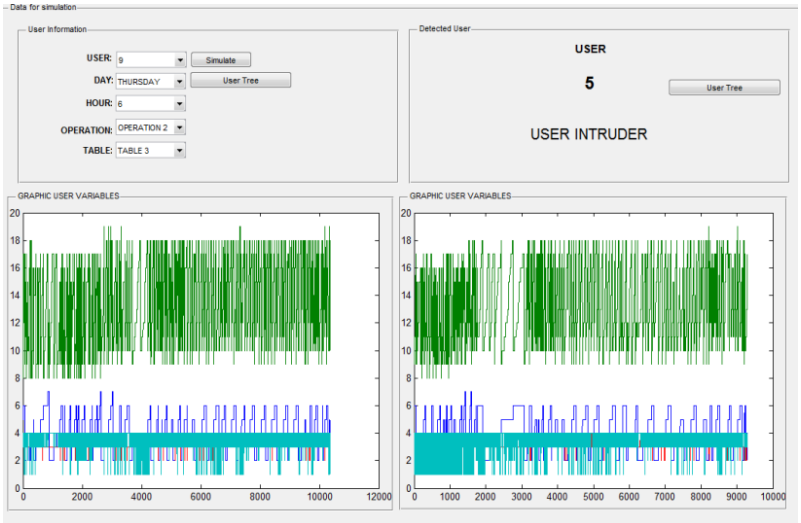



**Fig.** 6.Simulation tool of the intrusion detection system

As it is possible to see, it does not correspond to user #9. Furthermore, the activities' pattern corresponds to user 5. Therefore it is identified as an intrusion.

The application can also detect access of users who do not present the expected behavior but, although it could be an intrusion it is not clear that it is an intrusion. That means that the user is performing some actions with low probability regarding the corresponding decision tree but his activities somehow match the expected flow of actions. In this case the decision tool gives as result "non-identified user" and sends an alert but it does not block the access.

## 5 Conclusions

Intelligent techniques such as decision trees and neural networks have been proved very efficient when applied to pattern recognition and classification. In this paper we have developed a simulation tool that implements both of them to detect unauthorized accesses to information systems.

The decision tree allowed us to obtain a representation of the behavior of the user close to reality and interpretable. On the other hand, neural networks identify behavioral patterns based on the description given by the decision tree.

Previously, an exploratory analysis of the data was carried out in order to extract the key knowledge and to identify the significant variables that allow the discrimination of different users' behavior.

This decision-making system has been applied to real data provided by a governmental institution of the Republic of Ecuador.

As future work, other artificial techniques will be applied [12]. As it is well known, while many ensemble approaches exist, it remains, however, a difficult task to find a suitable ensemble configuration for a particular dataset [13].

**Acknowledgments**
This work has been partially supported by the Ministry of Higher Education, Science, Technology and Innovation (SENESCYT) of the Government of the Republic of Ecuador under the scholarship "Convocatoria Abierta 2011 y 2012".

## References

1. Haq, N. F., Onik, A. R., Avishek, M., Hridoy, K., Rafni, M., Shah, F. M., & Farid, D. M. (2015). Application of machine learning approaches in intrusion detection system: a survey. International Journal of Advanced Research in Artificial Intelligence, 4(3) 9-18.
2. Ahmed, M., Pal, R., Hossain, M. M., Bikas, M. A. N., & Hasa, M. K. (2009, April). A comparative study on the currently existing intrusion detection systems. In Computer Science and Information Technology-Spring Conference, 2009. IACSITSC'09. International Association of (pp. 151-154). IEEE.
3. Guevara, C. B., Santos, M., & López, M. V. (2016). Negative Selection and Knuth Morris Pratt Algorithm for Anomaly Detection. IEEE Latin America Transactions, 14(3), 1473-1479.
4. Jo, S., Sung, H., &Ahn, B. (2015). A comparative study on the performance of intrusion detection using Decision Tree and Artificial Neural Network models. Journal of the Korea Society of Digital Industry and Information Management, 11(4), 33-45.
5. Esmaily, J., Moradinezhad, R., & Ghasemi, J. (2015, May). Intrusion detection system based on Multi-Layer Perceptron Neural Networks and Decision Tree. In Information and Knowledge Technology (IKT), 2015 7th Conference on (pp. 1-5). IEEE.
6. Chen, Y., Abraham, A., & Yang, B. (2007). Hybrid flexible neural tree based intrusion detection systems. International Journal of Intelligent Systems, 22(4), 337-352.
7. Liu, G., Yi, Z., & Yang, S. (2007). A hierarchical intrusion detection model based on the PCA neural networks. Neuro computing, 70(7), 1561-1568.
8. Amudhavel, J., Brindha, V., Anantharaj, B., Karthikeyan, P., Bhuvaneswari, B., Vasanthi, M., ... & Vinodha, D. (March, 2016). A Survey on Intrusion Detection System: State of the Art Review. Indian Journal of Science and Technology, 9(11).
9. Thomsen, E. (2002). OLAP solutions: building multidimensional information systems. John Wiley & Sons.
10. Prakash, P. O., & Jaya, A. (2016). Analyzing and Predicting User Behavior Pattern from Weblogs. International Journal of Applied Engineering Research, 11(9), 6278-6283.
11. Guevara, C., Santos, M., & López, V. (2014, August). Data Leakage Detection Algorithm Based on Sequences of Activities. In Proc. 17th Int. Symp. Research in Attacks, Intrusions and Defenses RAID, 8688, 477-478, Springer.
12. Santos, M. (2011). An applied approach to intelligent control. Revista Iberoamericana de Automática e Informática Industrial RIAI, 8(4), 283-296.
13. Aburomman, A. A., & Reaz, M. B. I. (2016). A novel SVM-kNN-PSO ensemble method for intrusion detection system. Applied Soft Computing, 38, 360-372.

# Negative Selection and Knuth Morris Pratt Algorithm for Anomaly Detection

C. B. Guevara, M. Santos and V. López

***Abstract*— In this paper an algorithm for detecting anomalous behavior on computer systems is proposed. The work is based on information from the behavior of authorized users who have performed various tasks on a computer system over two years. The study uses a dynamic data structure that can encode the current activities of users and their behaviors. The identification of the most and least frequent tasks, based on the historical database of each user, provides a simple way of creating a single profile of behavior. With this profile, we apply negative selection techniques to obtain a reasonable computational size set of anomalous detectors. We then apply the Knuth-Morris-Pratt algorithm for locating detectors of anomalies as indicators of fraudulent behavior. This procedure for detecting anomalous behavior has been tested on real data and the results prove the effectiveness of the proposal and motivate further research to improve the existing detection system.**

***Keywords*— Anomaly Detection, Computer Systems, Behavior Profile, Negative Selection, Knuth Morris Pratt algorithm.**

## I. Introducción

EL PRESENTE trabajo propone un procedimiento para la detección de comportamientos no esperados en un sistema informático, con el fin de detectar sesiones anómalas, en las que pueda producirse una fuga de información. Es un tema importante, crítico para algunas instituciones y empresas. En concreto estamos interesados en la detección de anomalías en secuencias discretas de actividades, realizadas por usuarios autorizados. No es una tarea fácil ya que implica el análisis de una gran cantidad de datos, y la obtención de perfiles de comportamiento únicos para cada usuario, de forma que se puedan detectar posibles irregularidades [1, 2].

En la literatura se encuentran distintos enfoques para desarrollan sistemas de detección de anomalías. Por ejemplo, en Gao [3] se usa HMM (Hidden Markov Models) para aprender los patrones de comportamiento normal y anómalo de usuarios que realizan procesos UNIX. El trabajo presentado en [4] trata con datos históricos de comandos ejecutados por usuarios, y establece un ranking para seleccionar de entre ellos los más relevantes. Propone la clasificación de cada secuencia según la frecuencia de lo que se denominan las trazas normales y la de los procesos raros o no esperados, donde se encuentran las intrusiones. En [5] se presenta una revisión comparada de varios métodos de detección de anomalías para detección de intrusos en sistemas informáticos, focalizada en redes con un gran número de conexiones. Más actuales son las revisiones de [6, 7, 8, 9]. En ellas se clasifican por las técnicas aplicadas, la complejidad computacional, el dominio, etc. Sin embargo, en esos trabajos citados no tienen en cuenta la estructura secuencial de ejecución de las operaciones sino la cantidad de las operaciones ejecutadas, la transferencia de información, etc.

En el presente trabajo se propone una estructura de datos dinámica que permite codificar las actividades de los usuarios. La identificación de las tareas más y menos frecuentes, específicamente la secuencia de tareas realizadas, basada en el histórico del cada usuario, lleva a la creación de un perfil de comportamiento para cada uno. En esto se va a basar la detección de anomalías. Por otro lado, se han aplicado técnicas de selección negativa para obtener un conjunto de dimensiones reducidas, de forma que sea computacionalmente manejable, de detectores anómalos. Posteriormente se ha aplicado el algoritmo de alineamiento de secuencias Knuth-Morris-Pratt para la localización de estos detectores anómalos definidos como indicadores de comportamientos fraudulentos en secuencias de actividades de usuarios.

La aplicación de técnicas de selección negativa, que emulan el Sistema Inmune Artificial (AIS), para detección de anomalías no es nueva [10]. El AIS ha inspirado el desarrollo de modelos computacionales para la resolución de diversos problemas, incluyendo el diagnóstico de fallas, detección de virus, malware y detección de fraude, entre otras aplicaciones [11, 12, 13, 14]. En concreto existen en la actualidad sistemas de detección de intrusos (IDS) que utilizan esta aproximación como se refleja en las revisiones de [9]. Algunos ejemplos de detección de comportamiento anómalos mediante selección negativa se pueden ver en [15], donde se parte sólo de muestras positivas para entrenar el sistema (y no de muestras de comportamiento anómalo, que se generan mediante este algoritmo); y en [16, 17], para el tráfico de una red de información.

Un artículo muy interesante en la línea de nuestro trabajo es el de [18], ya que está centrado en la detección de anomalías en secuencias discretas. Formula el problema en tres pasos: identificación de las secuencias anómalas respecto a una base de datos que contiene las secuencias normales; identificación de las subsecuencias anómalas dentro de secuencias más largas; identificación del patrón en una secuencia cuya frecuencia de ocurrencia es anómala. Para cada uno de estos tres procesos agrupa las técnicas que se han venido aplicando en diferentes dominios, glosando sus puntos fuertes y sus desventajas. Como allí se sugiere, la tendencia actual es la hibridación de varias de esas metodologías para adaptarlas a las distintas formulaciones del problema. En este sentido está dirigida nuestra propuesta, donde la selección negativa nos permite generar detectores de anomalías que luego serán utilizados por el algoritmo de Knuth-Morris-Pratt para detectar subsecuencias anómalas dentro de la base de datos de secuencias del usuario informático.

El documento está organizado como sigue. En la sección 2 se describen brevemente los datos utilizados y los algoritmos

C. B. Guevara es Doctorando de la Facultad de Informática de la Universidad Complutense de Madrid (UCM), cesargue@ucm.es

M. Santos es Catedrática de la UCM, msantos@ucm.es

V. López es Profesora Contratada Doctora de UCM, vlopez@fdi.ucm.es

que se han aplicado. En la sección 3 se detalla la estructura de los datos utilizada para la codificación, así como el proceso propuesto para la detección de anomalías. La sección 4 presenta y analiza los resultados. El artículo termina con las conclusiones y trabajos futuros.

## II. Materiales y Métodos

En un sistema informático, cada usuario ejecuta una secuencia de tareas en función de su carga de trabajo o su asignación diaria. Sin conocimiento previo sobre el perfil del usuario, el problema principal es distinguir entre comportamiento normal y anormal, es decir, distinguir si las tareas ejecutadas son las habituales o si por el contrario son producto de un comportamiento fraudulento. Una forma de detectar posibles usos incorrectos de la información es la monitorización del sistema para obtener muestras con las que poder determinar patrones de comportamiento normal o anómalo.

Los datos seleccionados para este trabajo fueron recogidos de una institución oficial de la República de Ecuador. Esta información es confidencial y los datos se han codificado para salvaguardar su integridad. Este conjunto de datos se ha generado recopilando las actividades ejecutadas por usuarios autorizados en un sistema informático, en una serie de ficheros o repositorios. Se ha recogido información de 10 usuarios, que han realizado diversas sesiones de trabajo (*login*), y en cada una han realizado una serie de tareas (operaciones con ficheros). Las sesiones fueron grabadas durante el período de 2011 a 2013. Tras su procesamiento y depuración se cuenta con 20.883 registros de sesiones de usuario. Para generar el histórico de cada usuario se han considerado únicamente las actividades normales. Para comprobar la eficiencia del sistema de detección propuesto se han usado el conocimiento de un experto para comprobar que la clasificación era correcta.

Los algoritmos utilizados se describen a continuación.

### A. Algoritmo de Selección Negativa

Un Sistema Inmune Artificial (AIS) se basa en el funcionamiento del sistema inmune humano, que es capaz de reconocer de forma eficiente cualquier agente patógeno. La teoría clásica trata de analizar el sistema inmunológico como un sistema donde se pueden identificar dos tipos de agentes: propios y patógenos. Una forma de realizar esta clasificación en el sistema es mediante el algoritmo de Selección Negativa o NSA (Negative Selection Algorithm) [10]. Se trata de un algoritmo dinámico que se aplica como discriminador de objetos o elementos que son ajenos y dañinos a un sistema, por ejemplo para detección de virus computacionales, detección de intrusos, o detección de anomalías [14, 15, 16].

La clave de este método es que permite reducir el espacio de exploración, descartando las situaciones no posibles como tales, en vez de evaluarlas. Es decir, cuanto mayor sea la discriminación que se aplica, menor será el espacio de búsqueda de forma que se reducen drásticamente los tiempos computacionales de ejecución. El proceso se lleva a cabo sobre el espacio general de agentes (propios o patógenos), y dicho espacio se va reduciendo mediante la observación de la muestra donde se quiere realizar la identificación y un conjunto de muestras conocidas y sanas. Esta reducción es fundamental: el espacio general no puede ser considerado espacio de búsqueda debido al volumen de datos que ello supone. En nuestro ejemplo, el conjunto de combinaciones de actividades que puede realizar un usuario informático es prácticamente ilimitado. De esta forma no trabajaremos con todas las posibles secuencias de actividades, sino sólo con las que hayan aparecido alguna vez en el histórico.

Los sistemas inmunológicos pueden secuenciarse como cadenas de caracteres y, en ellas, cualquier subsecuencia de cualquier longitud puede representar un agente patógeno. Por lo tanto si $n$ es el número de caracteres en el alfabeto del que se trate y $m$ es la longitud de la secuencia a analizar, el espacio de búsqueda tendrá un tamaño correspondiente a las variaciones con repetición de $n$ elementos tomados de $i$ en $i$, siendo $1<=i<=m$, es decir, una complejidad computacionalmente no tratable de orden $n^m$. Con esta heurística de selección negativa conseguimos reducirlo a una complejidad de orden de $n^2$ manteniendo unas soluciones adecuadas.

### B. Algoritmo de Knuth-Morris-Pratt (KMP)

Una forma de detectar anomalías en el comportamiento de un usuario en un sistema informático es la búsqueda de patrones de comportamiento no esperado. Los algoritmos de alineamiento de secuencias son adecuados para este problema pues tratan de localizar subsecuencias anómalas dentro de secuencias más largas. Considerando que el comportamiento de un usuario puede modelizarse como una cadena de tareas realizadas secuencialmente, podemos aplicar alineamiento de secuencias para localizar subsecuencias sospechosas de fraude. Existen algunos trabajos previos en literatura sobre aplicación de alineamiento de secuencias a detección de intrusos [19].

El algoritmo de Knut-Morris-Pratt es un ejemplo clásico de alineamiento de secuencias para buscar secuencias [20]. El objetivo principal de este algoritmo es encontrar una palabra $P$ dentro de una cadena $S$. Para ello se compara cada carácter de $P$ con $S$. Si se encuentra, continúa la búsqueda. Si no es así, se desplaza P una posición a la derecha y se repite el proceso de búsqueda de coincidencia. La búsqueda se realiza mediante recorridos lineales simples pero la potencia de ejecución se consigue mediante la introducción de una tabla inicial para acelerar el proceso cuando se detecta un fallo. La complejidad del algoritmo resulta lineal respecto de la suma de las longitudes de la palabra y la cadena. Esta es $O(m,n)$, siendo $m$ la longitud del patrón P y $n$ la longitud de la cadena S. Como se supone, sin pérdida de generalidad, $m < n$, el tiempo de ejecución es $O(n)$.

En este trabajo se utilizará la versión de [20] este algoritmo para detectar subsecuencias anómalas.

## III. Algoritmo de Detección de Anomalías

El comportamiento de un usuario, entendiendo por tal las secuencias de actividades que realiza en un sistema informático, depende de la carga de trabajo o asignaciones que reciba, que puede ser muy variable. Por lo tanto es dinámico y diferente de los demás. El sistema que vamos a desarrollar en este trabajo se fundamenta en el análisis de una nueva sesión de usuario y su clasificación como correcta o fraudulenta en función de la similitud que la sesión en estudio posea con las del histórico.

Definimos el conjunto de tareas posibles (realizables por algún usuario del sistema) como Tsk $= \{t_1 \dots t_m\}$, donde $m$ es el número máximo de tareas diferentes ejecutables en el sistema. Cada tarea consiste en realizar una operación sobre un fichero. Se define una sesión *S* del usuario $u \in U$, siendo *U* el conjunto de usuarios del sistema, como una secuencia ordenada de tareas de tamaño variable *z,* es decir: $S^u = (t_1^u, t_2^u, \dots, t_z^u)$.

El conjunto de sesiones monitorizadas y almacenadas en el archivo histórico del usuario *u* se denota por $Hs^u = \{S_1^u, S_2^u, \dots, S_q^u\}$, donde $q$ es el número (también variable) de sesiones realizadas por ese usuario *u* en el periodo de tiempo considerado. En la Tabla I se muestra, a modo de ejemplo, parte del histórico del usuario #1, formado por 6 sesiones de diferente longitud cada una.

Una nueva sesión del usuario *u* ejecutada fuera del periodo de monitorización se denomina Sesión Activa y se denota ${S_A}^u$. Esta es la sesión que deseamos someter a evaluación para detección de sesiones anómalas.

El algoritmo propuesto para la detección de secuencias anómalas se realiza en los siguientes pasos:

### *A. Clasificación de tareas según su frecuencia de aparición*

En esta etapa se encuentran las tareas más, menos frecuentes, y de frecuencia media de aparición dentro del histórico de tareas de un usuario. Para ello se realiza un recuento del número de veces que cada tarea *t* ha sido realizada. Este valor, que denominamos $Ntask(t,u)$, se calcula mediante la expresión siguiente:

$$Ntask(t,u) = \sum_{t \in Tsk(Hs^u)} frec(t, Hs^u) \qquad (1)$$

siendo

$$Tsk(Hs^u) = \bigcup_{t_{ij} \in S_i} \bigcup_{S_i \in Hs^u} \{t_{ij}\}$$

Y el cálculo de la frecuencia de aparición se ha obtenido aplicando:

$$frec(t, Hs^u) = \sum_{i=1}^{q} frec(t, S_i);$$

$$frec(t, S_i) = \sum_{j=i_1}^{i_z} X_j \quad ; \qquad X_j = \begin{cases} 1 & Si\ t = t_j \\ 0 & Si\ t \neq t_j \end{cases}$$

A partir de ese valor se define un umbral, calculado mediante la media:

$$\mathrm{Mtask(u)} = \frac{\mathrm{Ntask(t,u)}}{|\mathrm{Tsk(Hs^u)}|} \qquad (2)$$

Este umbral se utiliza para clasificar las tareas del usuario según su frecuencia de aparición en:

- Tareas ME (más ejecutadas), si $Ntask(\mathrm{u}) > Mtask(u)$
- Tareas MDE (tareas por debajo de la media), si $Ntask(u) < Mtask(u)$
- Tareas NE (tareas no ejecutadas), es decir, $t \notin Tsk^U$.

Estos conjuntos se calculan de forma interactiva haciendo:

$$NE = Tsk - Tsk(Hs^u)$$

$$\forall t \in Tsk(Hs^u), \quad If\ Ntsk(u) < Mtsk(u) \quad then$$
$$MDE = MDE \cup \{t\}$$
$$else\ ME = ME \cup \{t\}$$

Por ejemplo, supongamos que el usuario #1 trabaja en un sistema donde existen 6 tareas, esto es, Tsk $= \{t_1 \dots t_6\}$. De su histórico, que se muestra en la Tabla I, se obtiene el número total de tareas (25), el de cada una de las tareas, *Ntask*, y la media *Mtask* (con y sin considerar las tareas sin ejecuciones, NE), así como su clasificación en los tres grupos anteriormente definidos.

TABLA I. HISTÓRICO DE SESIONES DE UN USUARIO Y CLASIFICACIÓN DE TAREAS.

| **Sesiones del Usuario 1:** | | | | | | |
|---|---|---|---|---|---|---|
| S1: T1—T3—T2—T4—T2; | | | | | | |
| S2: T3—T1—T1—T4—T3; | | | | | | |
| S3: T2—T3—T4—T4; | | | | | | |
| S4: T4—T2—T1; | | | | | | |
| S5: T1—T4—T3—T3; | | | | | | |
| S6: T3—T2—T4—T2; | | | | | | |
| **Tareas** | **T1** | **T2** | **T3** | **T4** | **T5** | **T6** |
| **$Ntsk_{ti}$** | 5 | 6 | 7 | 7 | 0 | 0 |
| **Mtsk (sin) = 6.25** | MDE | MDE | ME | ME | NE | NE |
| **Mtsk (con) = 4.17** | ME | ME | ME | ME | NE | NE |

En el ejemplo se obtienen los conjuntos de tareas MDE = {T1,T2}; ME={T3,T4} y NE = {T4,T5}. Se puede observar cómo varía la asignación a los grupos según se calcule el umbral incluyendo las tareas NE (media = 4.17) o sin considerarlas (media = 6.25).

Estos datos revelan información sobre el perfil del usuario concreto del cual se han extraído estos indicadores. Dicha información es útil para contrastar con sesiones activas del mismo usuario, por lo que debe almacenarse para posteriormente detectar anomalías de comportamiento en las sesiones activas y futuras.

### *B. Generación de detectores de anomalías*

Siguiendo con el ejemplo anterior, supongamos que el usuario #1 realiza una sesión en la que ejecuta las siguientes tareas: $\mathrm{As^U = T2 - T1 - T1 - T6 - T1 - T6 - T5}$.

En este paso del algoritmo se genera una población inicial mediante la unión los grupos de tareas presentes en la sesión activa del usuario. Así, en la sesión activa del ejemplo anterior, lo primero que habría que hacer es clasificar las tareas obteniéndose MDE – MDE – MDE – NE – MDE – NE – NE, ya que T1 y T2 aparecen en el histórico (MDE), y sin embargo T5 y T6 son tareas que no ha ejecutado nunca (NE). Por lo tanto la población inicial sería $B_0 = MDE \cup NE = \{T1, T2, T5, T6\}$. Es decir, está formada por las cuatro tareas distintas ejecutadas por el usuario en la sesión activa.

Formalmente,

$$\forall \mathrm{As^u} \quad , \quad \mathrm{S_A} = \bigcup_{\mathrm{t_i \in As^u}} \{\mathrm{t_i}\}$$

$$\forall \mathrm{B} \in \{\mathrm{ME, NE, MDE}\}, \mathrm{X_B} = \begin{cases} 0 & \mathrm{If\ B \cap S_A = 0} \\ \mathrm{B} & \mathrm{If\ B \cap S_A \neq 0} \end{cases}$$

y por lo tanto,

$$\mathrm{B_0 = (MDE \cap X_{MDE}) \cup (ME \cap X_{Me}) \cup (NE \cap X_{NE})}$$

A continuación se generan todas las posibles parejas ordenadas ($Ti$,$Tj$) con las tareas que ha realizado el usuario. Estos pares de tareas representan el tamaño mínimo de información para la detección (la subsecuencia). En total tendremos variaciones con repetición de $c$ elementos tomados de dos en dos, siendo $c$ el cardinal de la población inicial (en nuestro ejemplo $c$ = 4, por lo que tendremos 16 secuencias posibles). El conjunto de todas estas posibles parejas se denomina conjunto de detectores, $D$. Se puede representar como una matriz ($c$×$c$).

Mediante el algoritmo de Knut-Morris-Pratt (KMP) se comparan todos los detectores, $d$, con las secuencias del archivo histórico del usuario $Hs^U$. Un detector se clasifica como normal, $d \in D_n$, si esa subsecuencia ordenada se encuentra en el histórico (equivalente a una cadena). En caso contrario se clasifica como anómalo, $D_{an}$. El resultado será una partición $D = D_{an} \cup D_n$. Es decir, la construcción de los conjuntos de detectores normales y anómalos es la siguiente.

- Inicialización:

$$\mathrm{D} = \bigcup_{\mathrm{t_i t_j} \in \mathrm{Tsk(u)}} \{(\mathrm{t_i}, \mathrm{t_j})\}$$

$$\mathrm{D_n} = \emptyset \; ; \; \mathrm{D_{an}} = \emptyset$$

- Proceso:

$$\forall \mathrm{d} \in \mathrm{D}, \quad \text{If } \mathrm{KMP(d, Hs^u)} \text{ then } \mathrm{D_n} = \mathrm{D_n} \cup \{\mathrm{d}\}$$
$$\text{else } \mathrm{D_{an}} = \mathrm{D_{an}} \cup \{\mathrm{d}\}$$

En el ejemplo de la Tabla 1 se obtendría $D_n = \{(T1,T1),(T2,T1)\}$. Son las únicas subsecuencia de dos tareas ordenadas realizadas en la sesión actual y que se encuentran en el histórico. El resto son detectores anómalos que se obtendrían como el complementario al conjunto anterior, es decir:

$$D_{an} = \{(T1,T2),(T1,T5),(T1,T6),(T2,T2),(T2,T5),(T2,T6),(T5,T1),(T5,T2),(T5,T5),(T5,T6),(T6,T1),(T6,T2),(T6,T5),(T6,T6)\}$$

En este proceso se utilizan instancias a la función KMP que devuelve un resultado booleano igual a cierto si el detector d está en el histórico del usuario como subsecuencia. Es importante tener en cuenta que no es necesario ejecutar un KMP para este proceso por su sencillez, pero que al ejecutarse sobre un histórico de secuencias y siendo la búsqueda sobre una subsecuencia de tamaño 2, el coste del KMP es similar a otros procesos tradicionales.

Los detectores anómalos $d \in D_{an}$ pueden aparecer como subsecuencias en la sesión activa del usuario $\mathrm{As^U}$, lo que servirá como indicador de irregularidad en el comportamiento.

### C. Detección de subsecuencias anómalas

En esta última fase del proceso de detección se analiza la secuencia activa $As^U$ y se compara con el conjunto de detectores anómalos obtenidos anteriormente, $D_{an}$, aplicando nuevamente el algoritmo KMP. El objetivo es calcular el porcentaje $p$ de detectores anómalos en la sesión activa, que será comparado con un umbral de fiabilidad α fijado por el experto. Este umbral representa el porcentaje máximo aceptable de subsecuencias anómalas en la sesión que está siendo evaluada. Siendo muy conservadores, se puede tomar un umbral α=10%. Atendiendo a ese límite se va a clasificar la sesión activa en una de estas tres posibles situaciones: No Riesgo ($p<\alpha$), Riesgo ($\alpha<=p<=2\alpha$), y Alto Riesgo ($p> 2\alpha$).

La formulación de esta última parte del algoritmo es la siguiente,

$$num = \sum_{\mathrm{d} \in \mathrm{D_{an}}} (\mathrm{d}, \mathrm{As^u})$$
$$\mathrm{p} = (\mathrm{num}/|\mathrm{As^u}| - 1) * 100;$$
$$\text{If } (\mathrm{p} > 2\alpha) \text{ then } 'AltoRiesgo'$$
$$\text{elsif } (\alpha \leq p \leq 2\alpha) \text{ then } 'Riesgo'$$
$$\text{else } 'No\ Riesgo'$$

En el ejemplo, la sesión activa $\mathrm{As^U} = \mathrm{T2} - \mathrm{T1} - \mathrm{T1} - \mathrm{T6} - \mathrm{T1} - \mathrm{T6} - \mathrm{T5}$ presenta 4 parejas de detectores anómalos de las 6 secuencias de las que está compuesta (Fig. 1, en verde las dos primeras subsecuencias que pertenecen a $D_n$, y el resto en rojo porque corresponden a los detectores anómalos (T1,T6), (T6,T1), (T1,T6) y (T6,T5). En total hay un 66,6% de subsecuencias anómalas (p > 2α), por lo que esta sesión se clasifica como de *alto riesgo* de fraude.


Figura 1. Detección de subsecuencias anómalas

En general el umbral puede variar según la tolerancia, y se podría incrementar para evitar falsos positivos y que se ponga en marcha un protocolo de prevención que pueda obstaculizar el desarrollo del trabajo habitual de los operarios.

## IV. Resultados

La aplicación de este algoritmo tiene una fuerte dependencia de los datos del histórico de los usuarios. En ese sentido la fiabilidad de los resultados podría ser cuestionable dado que puede darse casos atípicos de comportamiento en los que un usuario modifique su rol debido a nuevas responsabilidades o a la existencia de estacionalidad en el sistema que no esté suficientemente reflejada en el histórico.

Para examinar la fiabilidad del procedimiento propuesto en este trabajo hemos realizado pruebas supervisadas con diez usuarios del sistema real descrito en la sección 2. Para cada uno de los 10 usuarios se han analizado las sesiones de sus respectivos históricos. El número de sesiones variaba entre 1 y 3 por día, dependiendo de cada usuario. La longitud de las sesiones está en un rango entre 2 y 65 tareas ejecutadas, con una media de 13 tareas distintas. De las sesiones de las bases de datos de cada usuario se ha utilizado un 70% para el entrenamiento del sistema, y un 30 % para validarlo (test).

Al trabajar con datos reales, el número de sesiones que se pueden clasificar como fraudulentas o anómalas es muy reducido, lógicamente. Es verdad que la experiencia nos dice que cuando un usuario realiza actividades de fuga de información, hasta que este comportamiento es detectado se acumulan una serie de sesiones, pero este número no es suficiente para obtener la información requerida sobre comportamiento anómalo. Por eso, para disponer de ejemplos tanto para entrenar el sistema como para luego probarlo se han generado sesiones anómalas mediante el método de selección

negativa. Este procedimiento no altera los resultados ya que al generar los detectores de anomalías, estos se reducen al mínimo. De hecho lo que queremos aportar con este método es una reducción de las tareas a analizar. Los usuarios no realizan todas las combinaciones de tareas posibles, que sería un número demasiado elevado para ser evaluado. Aplicando ciertas restricciones, la generación de un 25% de detectores anómalos del total de combinaciones existentes se ha demostrado suficiente para la correcta detección de anomalías.

A continuación se muestran algunos ejemplos del funcionamiento del sistema con datos reales.

$As_4^{U4}: T_{10} - T_{21} - T_3 - T_{12} - T_{10} - T_{21} - T_{12} - T_3 - T_3 - T_1 - T_{12} - T_{21} - T_{10} - T_3 - T_{12}$
ME= $\{T_{10}, T_{21}\}$; MDE=$\{T_1, T_3, T_{12}\}$; NE=0
$\text{num} = (D_{an}, As_4^{U4}) = 1$
$p = \left(\frac{1}{14}\right) * 100 = 7.14\% \quad \alpha = 10\%$
if $(p < \alpha)$ then'No Riesgo'
Experto: Normal

$As_2^{U4}: T_{10} - T_{10} - T_{19} - T_1 - T_{12} - T_{12} - T_1 - T_1 - T_3 - T_{21} - T_{10} - T_{21} - T_{10} - T_{10} - T_{15} - T_{12} - T_{21}$
ME= $\{T_{10}, T_{21}, T_{15}\}$; MDE=$\{T_1, T_3, T_{19}, T_{12}\}$; NE=0
$\text{num} = (D_{an}, As_2^{U4}) = 3$
$p = \left(\frac{3}{16}\right) * 100 = 18.75\% \quad \alpha = 10\%$
if $(\alpha \leq p \leq 2\alpha)$ then'Riesgo'
Experto: Fraudulenta

$As_3^{U4}: T_{15} - T_{10} - T_3 - T_1 - T_{10} - T_{10} - T_{19} - T_{19} - T_{19} - T_{19} - T_{10} - T_{10} - T_{21} - T_{10} - T_{10} - T_{16}$
ME=$\{T_{10}, T_{21}, T_{15}\}$; MDE=$\{T_1, T_3, T_{19}\}$; NE=$\{T_{16}\}$
$\text{num} = (D_{an}, As_3^{U4}) = 5$
$p = \left(\frac{5}{16}\right) * 100 = 31.25\% \quad \alpha = 10\%$
if $(p > 2\alpha)$ then'Alto Riesgo'
Experto: Fraudulenta

La clasificación del experto coincide con la del sistema, ya que se ha considerado que si el resultado del algoritmo para una sesión es "alto riesgo" o "riesgo", esas actividades se clasificaban como anómalas. Por lo tanto, para los ejemplos mostrados anteriormente, la clasificación del algoritmo es correcta en los tres casos. Se ha utilizado un valor de α = 10%.

La evaluación del rendimiento del sistema se ha hecho de la siguiente manera. Para un usuario, por ejemplo, el #4, hay 1998 sesiones almacenadas en su base de datos (histórico), todas ellas correctas. Se utilizarán un 70% para el entrenamiento (1400) y un 30 % para pruebas (598). Además se han generado 578 sesiones anómalas mediante el algoritmo de selección negativa. El conjunto de pruebas lo forman el 100% de las sesiones anómalas y el 30% de las normales, por lo tanto, hay un total de 1176 sesiones para pruebas.

Para este usuario, y con un valor del umbral de tolerancia al fraude de α = 10%, se han obtenido los datos de la Tabla II al aplicar el algoritmo de detección de sesiones anómalas.

TABLA II. CLASIFICACIÓN DE SESIONES DEL USUARIO #4 CON α = 10%.

| **Detectadas Correctamente** | | | |
|---|---|---|---|
| **# de Sesiones** | | **Algoritmo** | **Experto** |
| 587 | $p < \alpha$ | No Riesgo | Normal |
| 142 | $\alpha \leq p \leq 2\alpha$ | Riesgo | Fraude |
| 416 | $p > \alpha$ | Alto Riesgo | Fraude |
| **Detectados Incorrectamente** | | | |
| 20 | $p < \alpha$ | No Riesgo | Fraude |
| 2 | $\alpha \leq p \leq 2\alpha$ | Riesgo | Normal |
| 9 | $p > \alpha$ | Alto Riesgo | Normal |

Es decir, hay 1145 sesiones que se han detectado correctamente como normales o fraudulentas (97.36%), y 31 sesiones mal clasificadas (2.64%).

Para medir la eficiencia de la propuesta es importante calcular las detecciones de fraude consideradas verdadero negativo (TN), verdadero positivo (TP), falso negativo (FN) y falso positivo (FP), definidas como sigue:

- TN: sesión normal (experto) detectada como normal (algoritmo)
- TP: sesión anómala (experto) correctamente detectada como fraude (algoritmo)
- FP: sesión anómala (experto) incorrectamente detectada como normal (algoritmo)
- FN: sesión normal (experto) incorrectamente detectada como anómala (algoritmo).

Para el usuario #4, la Tabla III muestra los valores obtenidos para estos índices de funcionamiento.

TABLA III. TASAS DE CLASIFICACIÓN PARA EL USUARIO #4 CON α = 10%.

| **Algoritmo** | **Experto** | **Sesiones** | **Etiqueta** | **%** |
|---|---|---|---|---|
| Normal | Normal | 587 | Verdadero Negativo (TN) | 49.92 |
| Fraude | Fraude | 558 | Verdadero Positivo (TP) | 47.45 |
| Normal | Fraude | 20 | Falso Positivo (FP) | 1.70 |
| Fraude | Normal | 11 | Falso Negativo (FN) | 0.93 |

Es decir, el algoritmo funciona bien, con un porcentaje alto de detección correcta. Las tasas de falsos negativos y falsos positivos son bajas, siendo mayor el caso de las sesiones no detectadas como fraudulentas (FP). Esto se debe a la política conservadora que se ha seguido a la hora de elegir el umbral de tolerancia en la clasificación (α = 10%).

Si ahora aplicamos el detector con un α = 25%, para este mismo usuario, los resultados se muestran en las Tablas IV y V. En este caso, el número de sesiones correctamente detectadas es 874 (74.32%) y el de las mal clasificadas es 302 (25.681%). Como era de esperar ha aumentado el número de sesiones detectadas incorrectamente al aumentar el α. Igualmente, la tasa de falsos negativos y positivos también se ha incrementado notablemente (Tabla V).

TABLA IV. CLASIFICACIÓN DE SESIONES DEL USUARIO #4 CON α = 25%.

| **Detectadas Correctamente** | | | |
|---|---|---|---|
| **# de Sesiones** | | **Algoritmo** | **Experto** |
| 487 | $\mathbf{p < \alpha}$ | No Riesgo | Normal |
| 307 | $\mathbf{\alpha \leq p \leq 2\alpha}$ | Riesgo | Fraude |
| 80 | $\mathbf{p > 2\alpha}$ | Riesgo Alto | Fraude |
| **Detectados Incorrectamente** | | | |
| 191 | $\mathbf{p < \alpha}$ | No Riesgo | Fraude |
| 49 | $\mathbf{\alpha \leq p \leq 2\alpha}$ | Riesgo | Normal |
| 62 | $\mathbf{p > 2\alpha}$ | Riesgo Alto | Normal |

TABLA V. TASAS DE CLASIFICACIÓN PARA EL USUARIO #4 CON α = 25%.

| Algoritmo | Experto | Sesiones | Etiqueta | % |
|---|---|---|---|---|
| Normal | Normal | 487 | Verdadero Negativo (TN) | 41.41 |
| Fraude | Fraude | 387 | Verdadero Positivo (TP) | 32.91 |
| Normal | Fraude | 191 | Falso Positivo (FP) | 9.44 |
| Fraude | Normal | 111 | Falso Negativo (FN) | 16.24 |

Este algoritmo se ha aplicado para los 10 usuarios de los que se disponen de datos, obteniéndose los resultados de la Tabla 6 con un umbral de tolerancia al fraude de α = 10%. En la Tabla 7 se muestran las tasas de falsos positivos y negativos. La tasa de detección correcta, obtenida como la media, es del 97,28%.

TABLA VI. CLASIFICACIÓN DE SESIONES DE LOS USUARIOS POR EL SISTEMA CON α = 10%.

| | Detectados Correctamente | | | | Detectados Incorrectamente | | | |
|---|---|---|---|---|---|---|---|---|
| U | p<α | α≤p≤2α | p>2α | % | p<α | α≤p≤2α | p>2α | % |
| 1 | 345 | 121 | 614 | 96.948 | 14 | 5 | 15 | 3.052 |
| 2 | 374 | 234 | 264 | 95.614 | 16 | 6 | 18 | 4.386 |
| 3 | 466 | 239 | 197 | 96.265 | 15 | 11 | 9 | 3.735 |
| 4 | 587 | 142 | 416 | 97.364 | 20 | 2 | 9 | 2.636 |
| 5 | 450 | 87 | 244 | 94.897 | 15 | 9 | 18 | 5.103 |
| 6 | 477 | 56 | 308 | 95.136 | 15 | 14 | 14 | 4.864 |
| 7 | 442 | 196 | 346 | 96.094 | 10 | 17 | 13 | 3.906 |
| 8 | 450 | 329 | 149 | 95.769 | 11 | 14 | 16 | 4.231 |
| 9 | 457 | 474 | 114 | 96.136 | 9 | 18 | 15 | 3.864 |
| 10 | 357 | 461 | 186 | 95.076 | 10 | 24 | 18 | 4.924 |

TABLA VII. TASAS DE CLASIFICACIÓN CON α = 10%.

| Usuario | TP | TN | FP | FN | Tasa de Detección | Tasa Falsas Alarmas |
|---|---|---|---|---|---|---|
| 1 | 735 | 345 | 20 | 14 | 98.131 | 5.479 |
| 2 | 498 | 374 | 24 | 16 | 96.887 | 6.030 |
| 3 | 436 | 466 | 20 | 15 | 96.674 | 4.115 |
| 4 | 558 | 587 | 11 | 20 | 96.540 | 1.839 |
| 5 | 331 | 450 | 27 | 15 | 95.665 | 5.660 |
| 6 | 364 | 477 | 28 | 15 | 96.042 | 5.545 |
| 7 | 542 | 442 | 30 | 10 | 98.188 | 6.356 |
| 8 | 478 | 450 | 30 | 11 | 97.751 | 6.250 |
| 9 | 588 | 457 | 33 | 9 | 98.492 | 6.735 |
| 10 | 647 | 357 | 42 | 10 | 98.478 | 10.526 |

Si la tasa de error de falsos positivos fuera alta, el algoritmo de detección puede ser poco práctico ya que identifica erróneamente sesiones correctas como fraudulentas, bloqueando el acceso y perjudicando el ritmo de trabajo del usuario.

El tiempo computacional del algoritmo está en torno a 75 segundos de media, lo cual es un margen aceptable para enviar una alarma y bloquear el acceso a ese usuario si hiciera falta.

## V. CONCLUSIONES Y TRABAJOS FUTUROS

En este trabajo se ha desarrollado un algoritmo para la detección de comportamientos anómalos en sistemas informáticos, con el fin de detectar posibles anomalías en la ejecución de tareas en las sesiones de usuario. El trabajo utiliza información del comportamiento de usuarios autorizados que han ejecutado diversas tareas en un sistema informático a lo largo de dos años.

La identificación de las tareas más y menos frecuentes, basada en el histórico de cada usuario, proporciona un medio sencillo para la creación de un perfil único de comportamiento. Debido a la escasez de actividades fraudulentas con las que entrenar al sistema, éstas se han generado mediante un algoritmo de selección negativa inspirado en los sistemas inmunes. Posteriormente se ha aplicado un algoritmo de alineamiento de secuencias, el de Knuth-Morris-Pratt, para la localización de los detectores anómalos como indicadores de comportamientos fraudulentos.

Este algoritmo para la detección de comportamientos anómalos se ha probado con datos reales con resultados satisfactorios, avalando la eficiencia del procedimiento propuesto.

Como investigación futura se propone aplicar a este trabajo algoritmos de Big Data, que permitan reducir el tiempo de la detección, lo cual es crítico para este tipo de sistemas. También se plantea incrementar la eficiencia mediante la combinación con otras técnicas de clasificación como la de los vecinos más cercanos y las redes neuronales.

Sería muy deseable poder comprobar la robustez del método propuesto aplicándolo a otros datos.

## AGRADECIMIENTOS

Este trabajo ha sido parcialmente financiado por el Ministerio de Educación Superior, Ciencia, Tecnología e Innovación (SENESCYT) del Gobierno de la República del Ecuador bajo la beca "Convocatoria abierta 2011 y 2012".

## REFERENCIAS

[1] V. Chandola, A. Banerjee, and V. Kumar, “Anomaly detection for discrete sequences: A survey,” IEEE Trans. Knowl. Data Eng., vol. 24, no. 5, pp. 823–839, 2012.

[2] C. Guevara, M. Santos, and V. López, “Data Leakage Detection Algorithm Based on Sequences of Activities“. In Research in Attacks, Intrusions and Defenses: 17th International Symposium, RAID 2014, Gothenburg, Sweden, September 17-19, 2014, Proceedings (Vol. 8688, p. 477). Springer.

[3] B. Gao, H. Y. Ma, and Y. H. Yang, “HMMs (Hidden Markov models) based on anomaly intrusion detection method,” in Proceedings. International Conference on Machine Learning and Cybernetics, 2002, vol. 1, pp. 381–385.

[4] P. Helman and J. Bhangoo, “A statistically based system for prioritizing information exploration under uncertainty,” IEEE Trans. Syst. Man, Cybern. Part A Systems Humans., vol. 27, no. 4, pp. 449–466, 1997.

[5] A. Lazarevic, L. Ertoz, V. Kumar, A. Ozgur, and J. Srivastava, “A comparative study of anomaly detection schemes in network intrusion detection,” in Proceedings of the Third SIAM International Conference on Data Mining, 2003, vol. 3, pp. 25–36.

[6] A. Patcha and J. M. Park, “An overview of anomaly detection techniques: Existing solutions and latest technological trends,” Comput. Networks, vol. 51, no. 12, pp. 3448–3470, 2007.

[7] V. Chandola, A. Banerjee, and V. Kumar, “Anomaly detection: A survey,” ACM Comput. Surv., vol. 41, no. September, pp. 1–58, 2009.

[8] P. García-Teodoro, J. Díaz-Verdejo, G. Maciá-Fernández, and E. Vázquez, “Anomaly-based network intrusion detection: Techniques, systems and challenges,” Comput. Secur., vol. 28, no. 1–2, pp. 18–28, 2009.

[9] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, “Network Anomaly Detection: Methods, Systems and Tools,” IEEE Commun. Surv. Tutorials, vol. 16, no. 1, pp. 303–336, 2014.

[10] C. A. Laurentys, G. Ronacher, R. M. Palhares, and W. M. Caminhas, “Design of an Artificial Immune System for fault detection: A Negative Selection Approach,” Expert Syst. Appl., vol. 37, no. 7, pp. 5507–5513, 2010.

[11] L. J. Garcia Villalba, A. L. Sandoval Orozco, and J. Maestre Vidal, “Malware Detection System by Payload Analysis of Network Traffic,” IEEE Lat. Am. Trans., vol. 13, no. 3, pp. 850–855, Mar. 2015.

[12] D. Dasgupta, Ed., Artificial Immune Systems and Their Applications. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999.
[13] Aickelin, U., Dasgupta, D., and Gu, F. (2014). Artificial immune systems. In Search Methodologies (pp. 187-211). Springer, US.
[14] D. Dasgupta, K. Krishna Kumar, D. Wong, and M. Berry, "Negative Selection Algorithm for Aircraft Fault Detection," Artif. Immune Syst., vol. 7, no. 1, pp. 1–13, 2004.
[15] F. A. González and D. Dasgupta, "Anomaly Detection Using Real-Valued Negative Selection," Genet. Program. Evolvable Mach., vol. 4, no. 4, pp. 383–403.
[16] J. Kim and P. J. Bentley, "An evaluation of negative selection in an artificial immune system for network intrusion detection," in Proceedings of GECCO, 2001.
[17] A. S. A. Aziz, A. T. Azar, A. E. Hassanien, and S. E.-O. Hanafy, "Negative Selection Approach Application in Network Intrusion Detection Systems," Mar. 2014.
[18] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly Detection for Discrete Sequences: A Survey," IEEE Trans. Knowl. Data Eng., vol. 24, no. 5, pp. 823–839, May 2012.
[19] A. Kundu, S. Sural, and A. K. Majumdar, "Database intrusion detection using sequence alignment," Int. J. Inf. Secur., vol. 9, no. 3, pp. 179–191, Mar. 2010.
[20] P. Gawrychowski, A. Jez, and L. Jez, "Validating the Knuth-Morris-Pratt Failure Function, Fast and Online," Theory Comput. Syst., vol. 54, no. 2, pp. 337–372, Dec. 2013.

**César B Guevara Maldonado** es doctorando de la Facultad de Informática de la Universidad Complutense de Madrid (UCM). Máster en Investigación Informática en la UCM e Ingeniero en Sistemas e Informática de la Universidad de las Fuerzas Armadas ESPE. Actualmente becario de la Secretaría de Educación Superior, Ciencia, Tecnología e Innovación (SENESCYT) de la República del Ecuador.

**Matilde Santos** es licenciada y Doctora en Ciencias Físicas (1994) por la Universidad Complutense de Madrid (UCM). Catedrática de la Universidad Complutense de Madrid en la Facultad de Informática, en el Departamento de Arquitectura de Computadores y Automática. Sus principales líneas de investigación son la aplicación de técnicas de la Inteligencia Artificial a problemas reales en el ámbito de la ingeniería.

**Victoria López** es Doctora en Matemática Aplicada a la Computación e Inteligencia Artificial por la Universidad Politécnica de Madrid. Es licenciada en Matemáticas por la Universidad Complutense de Madrid, donde es Profesora de la Facultad de Informática en el Departamento de Arquitectura de Computadores y Automática.

Knowledge-Based Systems 000 (2017) 1–11

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

# Data leakage detection algorithm based on task sequences and probabilities

César Guevara, Matilde Santos, Victoria López*

*Computer Science Faculty, C/ Profesor García Santesmases, 9, Complutense University of Madrid, 28040-Madrid, Spain*

ARTICLE INFO

*Article history:*
Received 11 May 2016
Revised 4 January 2017
Accepted 5 January 2017
Available online xxx

*Keywords:*
Anomaly detection
Computer user behaviour
Markov chains
Data leakage
Knowledge-based decision system

ABSTRACT

In this paper we propose a novel algorithm to detect anomalous user behaviour in computer sessions. We first identify the behavioural profile of each authorized user from the computational tasks they usually carry out on the files of the information system. A new session is then codified as 2-length sequences and an algorithm based on the probability of those sequences is applied. The activities classified as possible anomalies are double-checked by applying Markov chains. The procedure has been proved efficient in terms of high detection accuracy and low false positive rate. It has been validate on a real database provided by a governmental institution of Ecuador and also on a public dataset of Unix commands. Besides, the algorithm has been shown efficient regarding computational time and the overhead of this monitoring software is low.



## 1. Introduction

Data Leakage (DL) is a serious and increasingly common problem for organizations, companies and institutions around the world. This problem can be defined as the unauthorized transfer of classified information from a computer or data centre to the outside world. That is, theft of computer based sensitive information. Sensitive data in companies and organizations include intellectual property, financial information, patient information, personal credit-card data, and any other information depending on the business and the industry involved [17,22].

The study presented in Cisco [7] reveals that organizations experienced about one threat per month. A significant proportion of computer and organizational security professionals believe insider threat is the greatest risk to their enterprise. Even more, internal fraud produces bigger financial losses in comparison to intrusions from outside the organization. These types of incidents, whether caused by malicious intent or an inadvertent mistake, can diminish a company's brand, reduce shareholder value, and damage the company's or organization goodwill and reputation [31]. But preventing theft by employees is difficult. The problem is that they may need to have access to sensitive or confidential information in order to perform their jobs, and they often copy the material they intend to take with them.

Being this the situation, the utility and necessity of any system that helps to alleviate this problem is more than justified. Therefore, the major goal of this paper is, on the one hand, to draw attention to this specific problem and, on the other hand, to propose an efficient knowledge-based algorithm for countering data leakage that, besides, reduces the number of false positive, which is the crucial problem of the data leakage detection methods. It is based on the user behaviour when working on a computational information system. The procedure is as follows. First we do anomaly detection, particularly, the algorithm we propose deals with the probability of sequences of operations performed by an authorized user on a workstation. If the output of the algorithm is that the actions carried out by the user corresponds to the expected ones, those sequences are identified as normal and removed from the system. Only sequences that are classified as possible anomaly by this algorithm are then evaluated using Markov chains, to double-check it. In case the information the user is working with is confidential or critical, we may then consider it a data leakage. The system has been proved reliable and efficient, with a low false positive rate.

Our work differs from existing research in the way we deal with the information about the computational operations of each user. We have developed a dynamic structure to codify the operations of the user in a computer system. This has allowed us to extract a user profile that can be obtained by the sequences of actions performed by the user from the historical database. We take into account not only the actions/commands or their relative frequency but the sequence of these operations, which gives us more relevant information and allows us to better detect this type of attacks.

* Corresponding author.
*E-mail addresses:* cesargue@ucm.es (C. Guevara), msantos@ucm.es (M. Santos), vlopezlo@ucm.es (V. López).

http://dx.doi.org/10.1016/j.knosys.2017.01.009
0950-7051/

This decision-making system has been validated on real data from a governmental institution of Equator with satisfactory results, and compared to another anomaly detection algorithm on a different public dataset, where it has been also applied, showing its potential adaptation to broader real-word application.

Another advantage of this detection system is that, once it has been trained, it can be applied on-line. The algorithm is able to analyse a sequence of operations within the current working session and, moreover, the users do not realize that they are being monitored as the overhead of the system is very low.

This rest of the paper is organized as follows. Section 2 summarizes related work on the topic of data leakage prevention and data leakage detection. In Section 3, the problem is described and the materials (dataset) and methods used are presented. Section 4 details the development of the data leakage detection algorithm and its application. Results are discussed in Section 5. The paper ends with the conclusions.

## 2. Background and related works

Data stored on the corporate networks is at risk because it is more accessible than ever. Organizations provide easy access to databases for information sharing, and storage and compression technology has allowed for more powerful (and risk-laden) operations. This increases the risk of confidential information falling into unauthorized hands. Information/Data Leakage poses a serious threat to companies and organizations, as the number of leakage incidents and the cost they inflict continues to increase.

We are assuming that whether caused by malicious intent [4], or an inadvertent mistake, by an insider or outsider [24], in any case, sensitive information lost can seriously hurt an organization. To reduce data leakage, two approaches have been proposed: Data Leak Prevention (DLP) and Data Leak Detection (DLD). Our work falls in the second category.

### 2.1. Data leak prevention

Data leakage prevention (DLP) has been studied both in academic research areas and in practical application domains. DLP is a suite of technologies aimed at safeguarding the computing resources to prevent leakage of confidential information found on hard drives, databases, etc. It focuses on the location, classification and monitoring of information at rest, in use and in motion. This solution can go far in helping in stopping the numerous leaks of information that occur each day on the globe.

DLP solutions can be classified according to various aspects such as: leakage source, data state, leakage channel, deployment scheme, preventive/detective approaches, and the action upon leakage [31]. In short, DLP techniques consist of detecting sensitive information that may be altered in anyway.

Most DLP tools have been installed at the front ends and are called agents. They use the operative system or other PC resources to scan the user activities. The agents identify sensitive information (e.g. credit card numbers, social security codes, etc.) by content - regardless of whether it is data in process of being transmitted over the network (data in motion, DIM), data on a server (data at rest, DAR), or data at an endpoint like a PC (data in use, DIU). Then these DLP proceed to encrypt or to block it [27]. But the information should be monitored more efficiently to reduce data loss statistics [11].

A major problem with DLP is that they rarely consider the flow of specified sensitive data items through the system in their decision process, but most often rely on the on-the-fly identification of sensitive data by using pattern matching or statistical algorithms. Besides, DLP techniques are often limited in terms of flexibility due to security policies established by governments and institutions.

As an example of the rates of detection that some developed tools can provide, in the work by Eesa et al. [9], the authors apply the cuttlefish algorithm to select some features on the KDD Cup 99 dataset and obtain a high detection (92.83% correctly classified) and accuracy (92.051%) rates with a low false alarm rate (1.41%).The false positive rate was high (5.3%) in Yeung and Ding [39], and in Chari and Cheng [5] (5%), with static behavioural models and rules (policies) for regulating access to system resources. The detection rate is good but not so high in other works such as the one by Liu and Lai [25], 80%, and Ye et al. [38], with 90%.

In summary, it is difficult to identify sensitive information and prevent leaking and, at the same time, ensuring the authorized users access to this information.

### 2.2. Data leakage detection

Incidents of data leakage in which personal or commercial sensitive data is leaked to untrusted or non-intended recipients are on the rise. Although researches on data leakage prevention are emerging, there is little research on the detection of information leakage from the perspective of user behaviour [36]. This is due to the fact that when detection is based on user behavioural patterns the problem becomes more complex. It requires, somehow, modelling the “normal” behaviour of authorized users. This can produce a high number of false positives because of the similarity between the expected behaviour of the authorized user that has to perform many different tasks or attempts to leak information. The efficiency of these systems is sometimes questionable [21].

These strategies consist of monitoring users rather than the information itself. That is, it is not based on alterations of the stored data but identifying the perpetrator [20,34]. Examples of intrusion detection systems can be found in Chavan and Desai [6] and Papadimitriou and Garcia-Molina [29]. In Kamra et al. [18], a mining process is used to form profiles that can model normal database access behaviour and identify intruders, based on the role of the employees. These methods usually estimate the likelihood of leakage and send an alert. In Mathew et al. [26] authors model users' access patterns by profiling the data points that users access, regarding query syntax. They apply statistical learning algorithms. Ye and Chen [37] present an anomaly detection technique based on a chi-square statistic. This technique builds a profile of normal events in an information system and detects anomalies.

But many pattern recognition techniques and clustering strategies can be applied to obtain the dynamic profile of user's behaviour [2,3,33]. Even more, no single technique can provide efficient intrusion detection [32]. An example of the application of different techniques to this very same dataset can be found in Guevara et al. [14], where Negative Selection and the Knuth Morris Pratt Algorithm are used. In that case many more data were used, most of them synthetically generated from the historical behaviour of the user in order to increase the number of negative samples of sequences. Similar results can be seen in Guevara et al. [13], where sub-sequences are generated from normal ones and the system is trained with those sequences. In that paper Bayes Networks and the Page Rank algorithm were applied. In Guevara et al. [15], a combination of neural networks and decision trees were applied to identify users and detect intrusions on the same database we are working here, with encouraging results.

Most of the papers that address similar problems deal with relative frequencies or other static characteristics of the information instead of sequence of activities. A notable exception is found in Costante et al [8], where authors monitor database activities and present a framework that automatically learns normal user behaviour, in terms of database activities, and detects anomalies as deviation from such behaviour.

Please cite this article as: C. Guevara et al., Data leakage detection algorithm based on task sequences and probabilities, Knowledge-Based Systems (2017), http://dx.doi.org/10.1016/j.knosys.2017.01.009

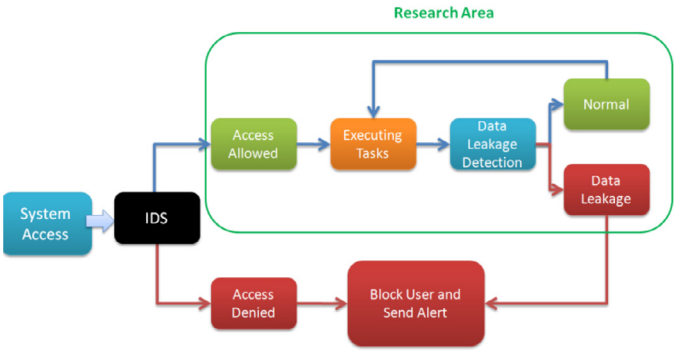



**Fig. 1.** Research area: data leakage detection.

We will compare our work with the one by Aeberhard et al. [1], who used a Unix command dataset. From it we have extracted the sequences of commands of different users and apply our algorithm to them. Results on this public real database are shown in Section 5 of this paper. Authors apply eight classifiers and their best result in terms of accuracy is 70.5% while ours gives 91.51%.

Our work applies anomaly detection but aiming at particularly reducing the number of false alarms, which is one of the main drawbacks of the detection methods.

## 3. Materials and methods

This study deals with authorized users who are allowed to access the computer information system (Fig. 1). It is assumed that some kind of intrusion detection system (IDS) has been previously applied. But once the user has login into the system, it is difficult to identify whether he/she is performing regular tasks or trying to leak information. Daily tasks can include sending out sensitive information.

Some of the difficulties come from the fact that the same user can present very different behaviour depending on the tasks he is in charge of at that moment. Only regarding the number of operations carried out by a user, the profile changes every month (Fig. 2), or it even varies during the same period of time but in different years (Fig. 3). This situation is even more different for the same user if we take into account the type of operation, the file he is working with, etc.

Because of that, the high ratio of false alarms that it is usually obtained by the data leakage detection systems is one of the main issues.

### 3.1. Dataset representation

The first step to approach the modelling of user behaviour is to use a flexible and efficient data structure. This representation must allow dynamic changes at least in the length of the registers or users [35].

In this work real data are provided (and anonymized) by a governmental institution of Ecuador. The database includes the tasks performed by 22 authorized users on different computers that belong to the institution. All of them had access to the same information resources. Only 10 of them stayed in the same institution the period of time we were interested in, from 2011 to 2013. Each user had an access profile that allowed him to execute only certain operations on specific files according to the position in the institution. The total number of access (computer sessions) over those three years was 12.727 and, since in each session several operation were executed, the number of operations registered those years is 573.731.

Among the characteristics of each access, the attributes that provide the most significant information about the behaviour of the agents are selected [12]. They are listed in the following tables.

*Access table (Login)*

This access table contains users' login information. Users $U_i$ are authorized people that work with the computer system, and a set of users is defined as $U = \{U_1, U_2, \ldots, U_m\}$, where in our case $m = 10$. Among others, this table contains the attributes *day, hour and minute*, which describes when the user logins into the system each time, and the duration of each session.

*Tasks table*

This table represents the activities carried out by the users into the computer system. The operations ($OP_i$) the user can perform with the files are: $OP_1$ = Update, $OP_2$ = Delete, $OP_3$ = Insert and $OP_4$ = Find. They work on 7 repositories or databases, $R_j$, each of them with different files which contain information about state properties, financial data, etc. Some of the repositories are considered to have confidential information.

A task $T_n$ is defined as an operation $OP_i$ executed on a repository, $R_j$, i.e. $T_n = (OP_i, R_j)$. We have considered a limited number of possible tasks, in this case 28 tasks (4 operations on 7 databases). The set of tasks executed by a user $U_i$ is represented as $T^{U_i} = \{T_1^{U_i}, T_2^{U_i}, T_3^{U_i}, \ldots, T_k^{U_i}\}$.

The information of the tasks carried out by the users is codified and unified as shown in Fig. 4 (User behaviour table). The database of tasks is sorted according to the day, hour and minute that every task was executed by each user $U_i$. In this way, it is possible to determine the sequence of tasks $T_k^{U_i}$ in a session and codified it to obtain a pattern. Sequence refers to a list of two linked task within a session, denoted as $sq(T_i, T_j)$or$(i, j)$.

For instance, Fig. 5 shows the database that is generated for a specific user, let us say $U_1$. Each row corresponds to a computer session, $s_i^{U_i}$. A session is a series of tasks executed by the user. The set of sessions each user has carried out is stored (historical database) and called Passive Session ($S^{U_i}$). It represents the historical behavior of the user, $S^{U_i} = \{s_1^{U_i}, s_2^{U_i}, \ldots, s_p^{U_i}\}$. In this example, user 1 has login the system six times ($p = 6$).

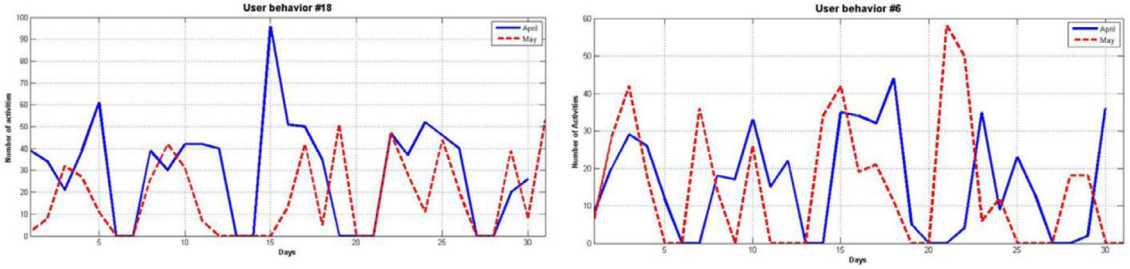



**Fig. 2.** Behaviour of users #18 (left) and #6 (right) in April (solid blue) and May (dotted red). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

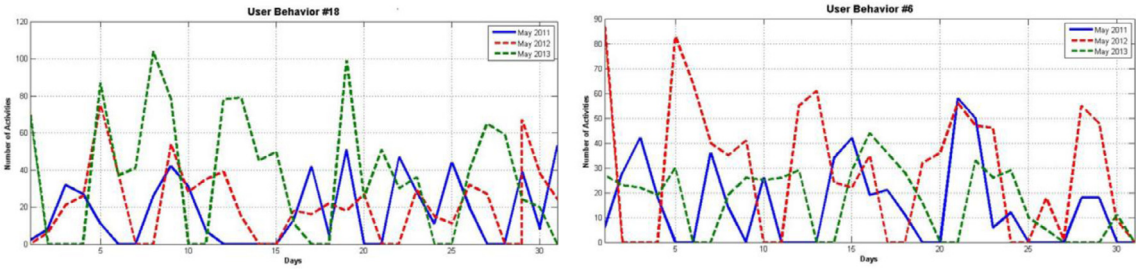



**Fig. 3.** Behaviour of users #18 (left) and #6 (right) the same month (May) in 2011(solid blue), 2012 (dotted red), 2013 (dotted green). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

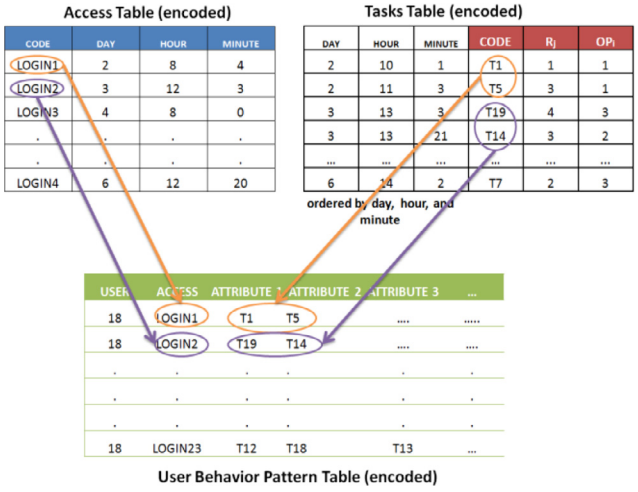



**Fig. 4.** Data structure of the tasks performed by a user $U$ in a computer system.

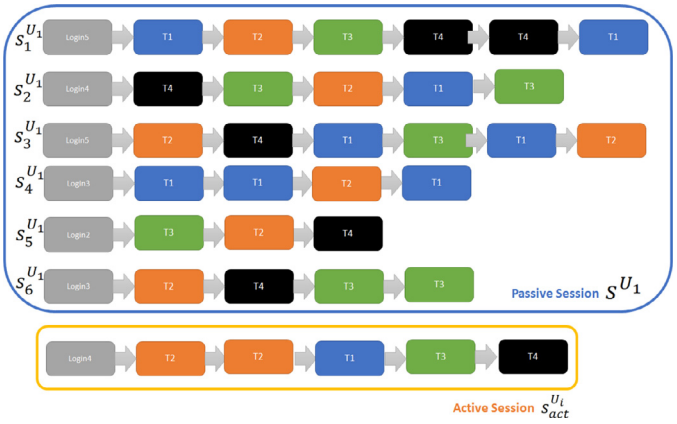



**Fig. 5.** Database of user sessions.

The length of a session depends on the number of tasks the user executes each time. For example, the first row represents $S_1^{U_1}$ and it consists of 6 tasks whereas the length of session 4th (fourth row) is 4. Any new session of the user, called Active Session, $s_{act}^{U_i}$ (Fig. 5, bottom), will be compared to the pattern obtained for that user, to determine if it corresponds to the expected behaviour of that user regarding the operation, files accessed, time, etc., or it could be a leakage.

### 3.2. Methods

In this paper, we have developed a novel algorithm to detect anomalous user behaviour in computer systems. It goes through a series of steps in order to double-check the identification of a leakage and to reduce the number of false alarms. Besides the calculation of some statistical characteristics and the likelihood of a sequence of commands, we finally apply Markov chains to make a final decision.

Markov chain is a random process that undergoes transitions from one state A to state B on a state space. The probability distribution of the state B depends only on the current state A and not on the sequence of previous events [19].

The formulation of this procedure is as follows, where the states are tasks in our case. Given a set of states $T = \{T_1, T_2, T_3, \ldots, T_m\}$, the probability of the transition between two of them in one step is defined as $a_{ij}$, i.e., the probability that the system switches from $T_i$ to $T_j$ is:

$$a_{ij} = Pr(T_m = j|/T_1 = t_1, T_2 = t_2, \ldots, T_{(m-1)} = i)$$
$$a_{ij} = Pr(T_m = j/T_{(m-1)} = i) \quad (1)$$

The probability of going from state $i$ to state $j$ in $m$ time steps is $P_{ij}^{(m)} = Pr(T_m = j/T_0 = i)$, and the single-step transition is $P_{ij} = Pr(T_1 = j/T_0 = i)$. In general, for a time-homogeneous Markov chain the transition matrix is:

$$P_{ij}^{(m)} = Pr(T_{(k+m)} = j/T_k = i) \quad (2)$$

and

$$P_{ij} = Pr(T_{(k+1)} = j/T_k = i) \quad (3)$$

The $m$-step transition probabilities satisfy the Chapman–Kolmogorov equation, $P_{ij}^m = \sum_{r \in S} P_{ir}^k P_{rj}^{n-k} 0 < k < m$, where S is the state space of the Markov chain. Let $X_k$ be the marginal random variable that sets the state at step k, then $X_0$ sets for the initial step.

$$P(X_k = j) = \sum_{r \in S} P_{rj}^k * P(X_0 = r) \quad (4)$$

Markov chains have many applications as statistical models of real-world processes, particularly, in the field of intrusion detection [28], anomaly detection [30], etc.

## 4. Data leakage detection algorithm based on sequences of tasks

As it was said in the introduction, first we apply anomaly detection and then do data leakage. The detection algorithm we propose consists of several steps in order to double check the final decision on whether it is an attempted attack or not. The steps of this algorithm are the following:

- Step 1: Calculation of sequence probability based on historical data.
- Step 2: Calculation of the more and less likely sequences within a session.
- Step 3: Checking anomalous sequences using Markov Chains.

Please cite this article as: C. Guevara et al., Data leakage detection algorithm based on task sequences and probabilities, Knowledge-Based Systems (2017), http://dx.doi.org/10.1016/j.knosys.2017.01.009

### 4.1. Step 1: calculation of sequence probability based on historical data

In this step, the algorithm uses information from the historical database of tasks to identify all 2-length sequences of tasks $(T_i, T_j)$ of a specific user and generates his profile.

The probability matrix of the sequences, $P_{sq}(i, j)$, for user $U_x$, is calculated as,

$$P\frac{U_x}{sq(i,j)} = \frac{Q_{ij}^{U_x}}{\sum_{i=1}^{m}\sum_{j=1}^{m} Q_{ij}^{Ux}} \tag{5}$$

Where each element $Q_{ij}^{Ux}$ of the square matrix is the number of times the sequence $(T_i, T_j)$ has been executed by user $U_x$. Note that $sq(T_i, T_j) \neq sq(T_i, T_j)$.

In the $P_{sq}(i, j)$ matrix, the sequences (i,j) are classified into two groups: the most (+) and less (-) likely sequences. The sequence with highest probability defines the $P_{Max}$ value and the sequence with the smallest probability gives the $P_{Min}$ value. A threshold, $\alpha \in (0, 1)$, is defined as the mean of these values, $\alpha = (P_{Max} + P_{Min})/2$.

A threshold for anomaly detection is defined as a value that classifies the actions of a specific user into two classes: anomalous and expected or normal ones, depending of the relative frequency the user carries out those actions. According to the each system one could choose a different threshold. In this work the thresholds are not fixed in advance as they are a function of the probabilities of the real sequences of the users. In this particular case, taking the mean of those probabilities to implement this threshold gave us good values while keeping it simple.

Let us see an example. For user $U_1$, according to Fig. 5, $Q_{11}$ is the number of times the sequence of task $(T_1, T_1)$ appears in that user's historical database. In this case is 1 (see Fig. 5, fourth row, $s_4^{U_1}$). The rest of the values of $Q_{ij}$ are calculated the same way. For instance, $(T_1, T_2) = 3$, $(T_2, T_1) = 2$, and so on. Therefore, the matrix $Q_{ij}$ for user $U_1$, with $k = 4$, is:

$$Q_{ij}^{U_1} = \begin{bmatrix} 1 & 3 & 2 & 0 \\ 2 & 0 & 1 & 3 \\ 1 & 2 & 1 & 1 \\ 2 & 0 & 2 & 1 \end{bmatrix} \tag{6}$$

As the probability matrix $P_{sq}{}^{U_1}(i, j)$ uses the sum of times any sequence appears in the $U_1$ database, in this case $\sum_i \sum_j Q_{ij} = 2$ Therefore, $P_{sq}^{U1}(i, j) = Q_{ij}/22$. The threshold is then $\alpha = 0.068$ because $P_{max} = 3/22$ and $P_{min} = 0$. Finally,

$$P_{sq}(i,j) = \begin{bmatrix} 0.045 & 0.136 & 0.090 & 0 \\ 0.090 & 0 & 0.045 & 0.136 \\ 0.045 & 0.090 & 0.045 & 0.045 \\ 0.090 & 0 & 0.090 & 0.045 \end{bmatrix} = \begin{bmatrix} - & + & + & - \\ + & - & - & + \\ - & + & - & - \\ + & - & + & - \end{bmatrix} \tag{7}$$

### 4.2. Step 2: calculation of the more and less likely sequences within a session

In this step the algorithm calculates the percentage of 2-length sequences $(T_i, T_j)$ labelled as (+) and (−) for each session. This percentage is called $Mp_{S_k}$ and $Lp_{S_k}$, respectively, where $S_k$ is the session that is being evaluated, which belongs to the set of historical sessions $S^{U_x}$. That is,

$$Mp_{S_k} = \frac{\sum Psq_i(+)}{|S_k| - 1}100\%, Lp_{sk} = \frac{\sum Psq_i(-)}{|S_k| - 1}100\% \tag{8}$$

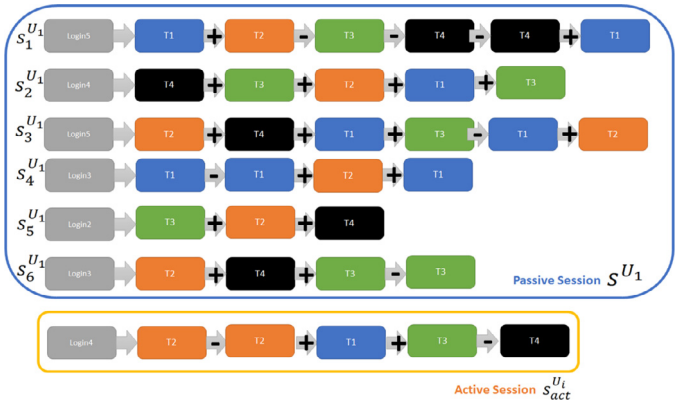



**Fig. 6.** More and less likely sequences in the user database.

**Table 1**
Percentage of more and less likely sequences for user $U_1$.

| $S_i$ | Nº sq | Nº of (+) | $Mp_{S_i}$ | Nº of (-) | $Lp_{S_i}$ |
|---|---|---|---|---|---|
| 1 | 5 | 2 | 40% | 3 | 60% |
| 2 | 4 | 4 | 100% | 0 | 0% |
| 3 | 5 | 4 | 80% | 1 | 20% |
| 4 | 3 | 2 | 66.66% | 1 | 33.33% |
| 5 | 2 | 2 | 100% | 0 | 0% |
| 6 | 3 | 2 | 66.66% | 1 | 33.33% |

The numerator is the number of sequences (+) or (−) in $S_k$ and the denominator is the number of 2-length sequences within $S_k$ (number of task minus 1).

Fig. 6 shows the values for user $U_1$ and Table 1 the corresponding indexes.

The minimum of each of these indexes will be set as other thresholds, that is, $Mp_{min}$ and $Lp_{min}$. In the example (Table 1), $Mp_{min} = 0.4$ and $Lp_{min} = 0$.

Once these thresholds have been obtained, they are applied to any new session of the user. Following the notation, for the actual session of $U_1$ (Fig. 6, last row):

$$Mp_{s_{act}} = \frac{\sum Psq_i(+)}{|S_{act}| - 1}100\% = \frac{2}{4}\text{x}100\%,$$

$$Lp_{s_{act}} = \frac{\sum Psq_i(-)}{|S_{act}| - 1}100\% = \frac{2}{4}\text{x}100\%$$

where the length of the session is 5 (number of tasks of that session).

Based on these indexes, it is possible to determine if so far the activity of the user can be considered an attempt of data leakage (if the indexes surpass the threshold). If so, we will continue the procedure to be sure of it. Otherwise it is not an anomaly and therefore this user activity is classified as normal.

In the example, $Mp_{s_{act}} = 50\% \geq$ Mpmin and $Lp_{s_{act}} = 50\% \geq 0\%$. The result of this step is true, meaning the decision is not clear since it could be an anomalous activity. Using a conservative policy we would identify it as a possible attack and generate an alarm. But if it is not, we would be generating false alarms and hindering the normal operation of the system.

Therefore we will try to improve these results by applying another technique, the Markov chain, to double-check this result.

### 4.3. Step 3: checking anomalous sequence using Markov chains

Once the step 2 of the algorithm has been evaluated, we obtain a first decision on the behaviour of the user (normal or possible attack). If the result is normal (expected behaviour), we do not need

to continue with the procedure and the system identifies those actions as corresponding to that authorized user. Otherwise, if the result of step 2 is that the user still has some probability of being trying to leakage information, in order to validate this statement we go on with step 3. This way, we get a second "opinion" on the user behaviour, and then the number of false positive can be reduced. It is worthy to remark that Markov chains are only applied to the possible anomalous sequences that result from step 2.

On the other hand, we have evaluated the results before and after applying the Markov chains to show the utility of this double-checking. This last step is just a verification of whether a probably anomalous action is a real attempted attack. Therefore, the Markov chains add extra information about the probability of being an attack. Indeed, as it will be shown, when we apply the Markov chain the number of false positive decreases and therefore, the accuracy of the classifier improves.

The Markov chains are applied to determine two types of anomalies:

- Anomaly detection type A: anomalies of 2-lenght sequences ($T_i$,$T_j$) within $S_{act}^{U_i}$ in one step.
- Anomaly detection type B: anomalies of 2-lenght sequences ($T_i$, $T_j$) within $S_{act}^{U_i}$ in $m$ steps.

#### 4.3.1. Anomaly detection type A

Let $U_x$ be a user and $T_i$, $T_j$ two tasks that belong to the set of tasks $\{T_1 \dots T_m\}$. The set $\Omega_i^{U_x} = \{(T_1, T_i) \dots (T_m, T_i)\}$ $(i = 1 \dots m)$ is the sample space of the activities performed by user $U_x$. The event $(T_i, T_j) \in \Omega_i^{U_x}$ means the user executes the sequence of tasks $T_i$-$T_j$. Normal activity of a user is related to frequent events and anomalous actions are linked to infrequent sequences of activities. In this sense, we can define the single step transition matrix (Eq. (3)), $P = (p_{ij})_{\ i,j=1\dots m}$, as the relative frequency matrix obtained from historical data of user $U_x$, where $p_{ij}$ is an estimation of the probability of user $U_x$ going from task $i$ to task $j$ in one step.

Each element of matrix $P$ is calculated by applying the Laplace rule, $p_{ij} = \frac{Q_{ij}^{U_x}}{\sum_{s=1}^{m} Q_{i,s}^{U_x}}$, where per each user $U_x$ and each task $i$, on the sample space $\Omega_i^{U_x}$, we define $Q_{ij}^{x}$ as the times the user $U_x$ has gone from $T_i$ to $T_j$ (from the historical data set, step 1 of the algorithm).

As shown in Eq. (6), the sum of the values of the first row of $Q_{ij}^{U_1}$ is $\sum_{i=1}^{4} Q_{1i} = 6$, therefore, $p_{11} = \frac{Q_{11}}{6} = \frac{1}{6} = 0.167$; $p_{12} = \frac{Q_{12}}{6} = \frac{3}{6} = 0.500$ and so on. The corresponding transition matrix $P$ for user $U_1$ is then:

$$P = \begin{bmatrix} 0.16 & 0.50 & 0.33 & 0 \\ 0.33 & 0 & 0.16 & 0.50 \\ 0.20 & 0.40 & 0.20 & 0.20 \\ 0.40 & 0 & 0.40 & 0.20 \end{bmatrix} \tag{9}$$

A new threshold $\delta$ is defined based on the values of the transition matrix P, that is, it depends on the user behaviour and the 2-task sequences performed in a session. To select it, we have taken into account that we want to eliminate the less likely sequences of tasks, i.e., with low probability of having been performed by that user, and of course remove those whose likelihood is zero. For this reason we cannot take the minimum probability as a threshold because it would be zero in most of the cases, but the smallest one different from zero. Even more, we have set an upper limit for this threshold, the first quartile, as it will reduce false positives ($\delta \leq 0.250$).

Let us see an example. First, sorting the values of P as follows, (0, 0, 0, 0.16, 0.16, 0.20, 0.20, 0.20, 0.20, 0.33, 0.33, 0.40, 0.40, 0.40, 0.50, 0.50), we obtain the threshold $\delta = 0.16$.

Assuming that $\hat{a}_{i,j} = p_{ij}$ is an estimation of $a_{i,j}$ (Eq. (1)), we will evaluate a new working session applying this procedure. If the new session is $s_{act}^{U_1} = (T_2, T_4, T_1, T_1, T_2, T_3)$, the sequences are -($T_2$, $T_4$), ($T_4$, $T_1$), ($T_1$, $T_1$), ($T_1$, $T_2$), ($T_2$, $T_3$)-, and applying the Markov probability matrix, their estimated probabilities are: $\hat{a}_{2,4} = p_{2,4} = 0.5$, $\hat{a}_{4,1} = p_{4,1} = 0.4$, $\hat{a}_{1,1} = p_{1,1} = 0.16$, $\hat{a}_{1,2} = p_{1,2} = 0.5$ and $\hat{a}_{2,3} = p_{2,3} = 0.16$. All of them are $p_{ij} \geq \delta$, thus the sequence is identified as normal and labelled as non-data leakage by the system.

Let us evaluate another new session, ($T_2$, $T_2$, $T_1$, $T_3$, $T_4$) (see Fig. 6, last row); the estimated values of the sequences are now: $\hat{a}_{2,2} = (T_2, T_2) = P_{2,2} = 0$, $\hat{a}_{2,1} = (T_2, T_1) = P_{2,1} = 0.33$, $\hat{a}_{1,3} = (T_1, T_3) = P_{1,3} = 0.33$, and $\hat{a}_{3,4} = (T_3, T_4) = P_{3,4} = 0.2$. In this case there is, at least, one sequence with probability $p_{ij} \leq \delta$ ($p_{22} = 0$). Thus, the binary sequence ($T_2$, $T_2$) is identified as anomaly.

Of course the results change if the threshold value varies. In the previous example, if $\delta = 0.250$ instead of 0.16, evaluating the same session $s_{act}^{U_1} = (T_2, T_2, T_1, T_3, T_4)$ (see Fig. 6, last row), the estimated values are now: $\hat{a}_{2,2} = P_{2,2} = 0$, $\hat{a}_{2,1} = P_{2,1} = 0.33$, $\hat{a}_{1,3} = P_{1,3} = 0.33$, and $\hat{a}_{3,4} = P_{3,4} = 0.2$. Then, there are now two sequences with probability $p_{ij} < \delta$ ($P_{2,2} = 0$ and $P_{3,4} = 0.2$). Thus, those binary sequences ($T_2$, $T_2$) and ($T_3$, $T_4$) are identified as possible anomaly.

We carried out some experiments using $\delta = 0.250$ and compared them with the values of Table 2 ($\delta < 0.250$). The total number of false positive (FP) was 2939 for $\delta < 0.250$ and 3552 for $\delta = 0.250$. Besides, the accuracy of the classifier was much better for $\delta < 0.250$ (48.07% vs. 37.18%), and the DR also bigger (47.44% vs. 34.64%). That is, it is better to use a smaller threshold $\delta$ because it better filters the cases with low probability. This proves the correctness of the choice of the threshold in order to reduce the false alarms.

#### 4.3.2. Anomaly detection type B

The algorithm calculates the probability of going from state $T_i$ to state $T_j$ in $m$ time steps, where $m$ is a finite number. In this case the future states depend on the past $m$ states.

$$p_{ij}^{m} = Pr(T_m = x_m / T_0 = x_0, T_1 = x_1, \dots, T_{(m-1)} = x_{(m-1)}) \tag{10}$$

As an example of anomaly detection type B for a new session of user $U_1$; we compute the probability of $s_{act}^{U_1} = (T_2, T_2, T_1, T_3, T_4)$, in $m$ steps, as shown in the following expressions. The threshold is now $\delta = 0.125$.

Step #1, $m = 1$

$p_{T_2,T_2}^{1} = P^1 = 0 < \delta$ then ($T_2$, $T_2$) is probably an anomalous sequence

Step #2, $m = 2$

$p_{T_2,T_1}^{2} = P^2 = 0.28 > \delta$ then ($T_2$, $T_1$) is probably a normal sequence

Step #3, $m = 3$

$p_{T_1,T_3}^{3} = P^3 = 0.28 > \delta$ then ($T_1$, $T_3$) is probably a normal sequence

Step #4, $m = 4$

$p_{T_3,T_4}^{4} = P^4 = 0.21 > \delta$ then ($T_3$, $T_4$) is probably a normal sequence

If any of the transitions has a probability smaller than $\delta$, it is then classified as possible anomaly. In this case it had not been necessary to calculate the steps from $m = 2$ to $m = 4$ because for $m = 1$, $P^1 < \delta$, therefore the sequence was classified as anomalous even if the rest of the transitions are identified as normal.

Let's see another example. For a given session, $s_{act}^{U_1} = (T_2, T_4, T_1, T_1, T_2, T_3)$, $m = 5$, the probability of the transition in $m$ steps is calculated as:

Step #1, $m = 1$

$p_{T_2,T_4}^{1} = P^1 = 0.5 > \delta$ then ($T_2$, $T_4$) is probably a normal sequence.

**Table 2**
Results of the algorithm (without Markov chain) for 10 users with dataset of Ecuador.

| User | #Normal sessions | #Fraudulent sessions | #Total sessions | TP | TN | FP | FN | %AR | %Error | %DR | %FPR |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 778 | 114 | 892 | 45 | 364 | 414 | 69 | 45.85 | 54.15 | 39.47 | 53.21 |
| 2 | 535 | 121 | 656 | 56 | 249 | 286 | 65 | 46.49 | 53.51 | 46.28 | 53.46 |
| 3 | 465 | 167 | 632 | 74 | 273 | 192 | 93 | 54.91 | 45.09 | 44.31 | 41.29 |
| 4 | 589 | 233 | 822 | 115 | 282 | 307 | 118 | 48.30 | 51.70 | 49.36 | 52.12 |
| 5 | 381 | 133 | 514 | 67 | 176 | 205 | 66 | 47.28 | 52.72 | 50.38 | 53.81 |
| 6 | 402 | 111 | 513 | 51 | 201 | 201 | 60 | 49.12 | 50.88 | 45.95 | 50.00 |
| 7 | 594 | 210 | 804 | 125 | 264 | 330 | 85 | 48.38 | 51.62 | 59.52 | 55.56 |
| 8 | 525 | 179 | 704 | 89 | 247 | 278 | 90 | 47.73 | 52.27 | 49.72 | 52.95 |
| 9 | 638 | 146 | 784 | 68 | 311 | 327 | 78 | 48.34 | 51.66 | 46.58 | 51.25 |
| 10 | 719 | 98 | 817 | 42 | 320 | 399 | 56 | 44.31 | 55.69 | 42.86 | 55.49 |
| Total | 5626 | 1512 | 7138 | 732 | 2687 | 2939 | 780 | | | | |
| Average | | | | | | | | 48.07 | 51.93 | 47.44 | 51.91 |

Step #2, $m = 2$

$p^2_{T_4,T_1} = P^2 = 0.224 > \delta$ then $(T_4, T_1)$ is probably a normal sequence.

Step #3, $m = 3$

$p^3_{T_1,T_1} = P^3 = 0.277 > \delta$ then $(T_1, T_1)$ is probably a normal sequence.

Step #4, $m = 4$

$p^4_{T_1,T_2} = P^4 = 0.252 > \delta$ then $(T_1, T_2)$ is probably a normal sequence.

Step #5, $m = 5$

$p^5_{T_2,T_3} = P^5 = 0.267 > \delta$ then $(T_2, T_3)$ is probably a normal sequence.

All the probabilities are bigger than $\delta = 0.125$, therefore this session is identified as normal and the detection process ends.

So far we have applied an anomaly detection algorithm. To do data leakage, we just check if any of the files that have been accessed during the task identified as anomalous contains confidential information. If any of the operations have been executed on files that belong to the set of critical activities, $Tc = \{T_1, T_2, \ldots, T_n\}$, defined as such by an expert, then a decision should be made on whether blocked the user to protect the information. Otherwise, and in any case, the decision-making system will send an alarm to the administrator.

That is, we distinguish between:

- Normal behaviour: the sequence of operations on the computer system matches the expected one for that user.
- Anomalous behaviour: the user is performing operations that do not match with the expected profile; therefore the algorithm detects an attempted attack.
- Data leakage: the anomalous behaviour involves the access to information that has been classified as confidential or critical.

Fig. 7 shows the diagram of the whole procedure that has been described.

#### 4.3.3. Enhancing the results

As the system is designed to monitoring users' activities, it must avoid introducing overhead and so to be used at real time. That is why it is important to decrease the number of false alarms that slows down the work pace of the user.

However, we are dealing with probabilities, and we could be wrong when identifying a normal or an anomalous sequence. With the step of the Markov chains, the probability of a normal sequence being detected as anomaly decreases. Thus, the probability of an anomalous sequence to be identified as such increases. We are therefore increasing the reliability of the true positives.

To express it in a more formal way: our method tries to improve the quality of detection by this double-checking. The model can be formulated as follows.

Let $sq_i$ be the $i$th sequence of the set of sequences of tasks $SQ = \{sq_1, \ldots, sq_n\}$ that inputs into the system.

Let $X_i$, $Y_i$ and $Z_i$ be binary variables defined as,

$X_i = 1 \Leftrightarrow Sq_i$ is an anomalous sequence
$Y_i = 1 \Leftrightarrow Sq_i$ is detected as an anomaly after step 2
$Z_i = 1 \Leftrightarrow Sq_i$ is detected as an anomaly after step 3
Otherwise they are zero. Then,
$P(X_i = 0/Y_i = 1) >= P(X_i = 0/Z_i = 1)$

Meaning that the probability after step 2 of being wrong (detecting a normal sequence as anomaly) is greater than the probability after step 3 of being wrong. That is, the probability of false positive (FP) is reduced by introducing step 3 in the detection algorithm.

We could say that we are surer when we say that a sequence is a possible attack after applying Markov chains. Due to this, the DR index will improve after applying this last step because the number of sequences identified as normal is greater and the number of possible anomalies has decreased.

## 5. Results and discussion

The algorithm has been applied to the real data described in Section 3.1. A number of 12.727 sessions were used, where only 512 were attacks, and the rest, 11.215, are considered normal. As it is well known, data containing authentic fraud scenarios is often not available for researchers but, on the other hand, classifiers need training data to successfully distinguish normal from fraudulent behaviour. That is why, in order to balance the sets of positive and negative examples, we have applied Negative Selection Algorithm to generate 1000 more negative samples. This is a common practice [23].

We have used 5589 sessions (44%) for training and 7138 for testing (56%), including the 1512 anomalous sessions available. We will show the influence of the number of examples on the effectiveness of the decision system because if you generated more fraudulent sessions the results do not improve much and, nevertheless, the computing is more demanding.

Results are given first in terms of percentage of correctly classified (CC) and incorrectly classified (IC) sessions, using the following definitions.

- TN (true negative): normal session detected as normal
- TP (true positive): data leakage successfully detected as such
- FN (false negative): data leakage incorrectly detected as normal session
- FP (false positive): normal session incorrectly detected as data leakage.

Please cite this article as: C. Guevara et al., Data leakage detection algorithm based on task sequences and probabilities, Knowledge-Based Systems (2017), http://dx.doi.org/10.1016/j.knosys.2017.01.009

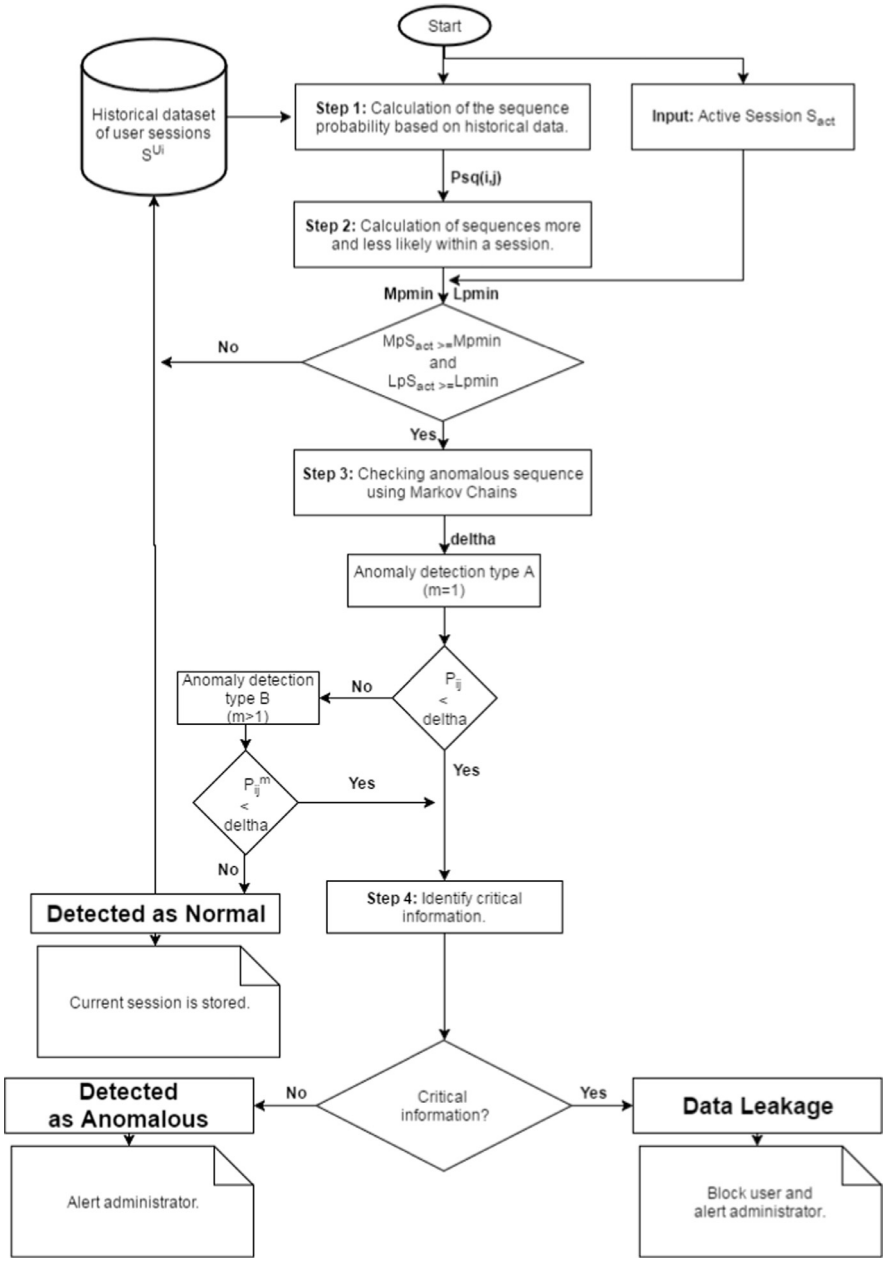



**Fig. 7.** Data leakage detection procedure.

The correctly classified CC value corresponds to the Detection Rate (DR),

$$DR = \frac{\text{No. of attacks that are correctly classified as attack (TP)}}{\text{Total No. of attacks in the test dataset (TP + FN)}}$$

The Accuracy Rate (AR), as defined by Eesa et al. [9], is defined as:

$$AR = \frac{\text{No. of correctly classified (TN + TP)}}{\text{Total No. of instances in the test dataset}}$$

The errors are given by the False Positive Rate (FPR) or False Alarm Rate (FAR) as defined by [10].

$$FPR = \frac{\text{No. of normal that are uncorrectly classified as attack (FP)}}{\text{Total No. of normal sessions (FP + TN)}}$$

Obviously, higher values of DR and AR, and lower values of FPR show better classification performance for the data leakage detection algorithm.

The algorithm has been tested for 10 different users. First, it was evaluated applying the first steps of the algorithm, that is, without applying the Markov chains. The results are summarized in Table 2.

The detection rate, that is, the percentage of sequences correctly classified, is around 47,5% and the accuracy of the decision system is 48%. For some of the users it gets values up to 59,5%. However, it is still very low. We then apply Markov chains in order to improve these results and mainly to reduce the number of false positive, particularly, the FPR index. In fact, once the Markov chains are applied to double-check the decision on whether a session is normal or anomalous, the results clearly improved, as it can be seen in Table 3.

The DR is now much better, 78% and the AR improves up to 91.5%, that it is indeed a good result taking into account we work with real data. The best performance of the algorithm gives AR = 93,03% and DR = 90,03%. Besides, the false positive rate is

Please cite this article as: C. Guevara et al., Data leakage detection algorithm based on task sequences and probabilities, Knowledge-Based Systems (2017), http://dx.doi.org/10.1016/j.knosys.2017.01.009

**Table 3**
Results of the algorithm (applying Markov chains) for 10 users with dataset of Ecuador.

| User | #Normal sessions | #Fraudulent sessions | #Total sessions | TP | TN | FP | FN | %AR | %Error | %DR | %FPR |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 778 | 114 | 892 | 78 | 743 | 35 | 36 | 92.04 | 7.96 | 68.42 | 4.50 |
| 2 | 535 | 121 | 656 | 96 | 501 | 34 | 25 | 91.01 | 8.99 | 79.34 | 6.36 |
| 3 | 465 | 167 | 632 | 104 | 465 | 0 | 63 | 90.03 | 9.97 | 62.28 | 0.00 |
| 4 | 589 | 233 | 822 | 203 | 548 | 41 | 30 | 91.36 | 8.64 | 87.12 | 6.96 |
| 5 | 381 | 133 | 514 | 106 | 359 | 22 | 27 | 90.47 | 9.53 | 79.70 | 5.77 |
| 6 | 402 | 111 | 513 | 88 | 378 | 24 | 23 | 90.84 | 9.16 | 79.28 | 5.97 |
| 7 | 594 | 210 | 804 | 181 | 567 | 27 | 29 | 93.03 | 6.97 | 86.19 | 4.55 |
| 8 | 525 | 179 | 704 | 142 | 507 | 18 | 37 | 92.19 | 7.81 | 79.33 | 3.43 |
| 9 | 638 | 146 | 784 | 116 | 606 | 32 | 30 | 92.09 | 7.91 | 79.45 | 5.02 |
| 10 | 719 | 98 | 817 | 67 | 685 | 34 | 31 | 92.04 | 7.96 | 68.37 | 4.73 |
| | 5626 | 1512 | 7138 | 1181 | 5359 | 267 | 331 | | | | |
| Average | | | | | | | | 91.51 | 8.49 | 76.95 | 4.73 |

**Table 4**
Results of the algorithm with real data (more anomalous sessions generated).

| | CC% | #CC sessions | Error % | #Error sessions |
|---|---|---|---|---|
| Data leak | 92.47% | 4198 | 7.53% | 342 |
| Normal | 97.10% | 5284 | 2.90% | 158 |
| Total | 94.78% | 9482 | 5.22% | 500 |

now quite low (FPR = 4.73%). If FPR were high, the detection algorithm would be inefficient because it would wrongly identify correct sessions as data leakage, unnecessarily blocking the access and slowing down the working pace of the users.

It is worthy to remark that the number of false positive (false alarms) has been greatly reduced from 2939 to 267. Even more, for some users we even get FPR = 0, meaning all the sessions were correctly classified. Besides, the number of attacks incorrectly detected as normal session is now 331, very low. The number of false negative is also small, meaning that the system is able of correctly detect anomalous behaviour.

We also did other experiments increasing the number of examples, i.e., generating more anomalous sessions. In Table 4 the results show the performance of the system when 4356 were anomalous sessions, most of them obtained synthetically.

As it is possible to see, the accuracy of the detection system is now 94,78%, no so far from the 91,51% when using only 1000 extra fraudulent sessions.

Admittedly, it is quite difficult to compare any detection algorithm to other procedures because there are many different types of data leakage and different approaches, depending on the objectives, the data usage, etc. Having said that, we found another real database in Aeberhard et al. [1] where to prove our proposal. It is based on the idea that the Unix commands used by a user in a workstation define his profile. From the database, available at https://archive.ics.uci.edu/ml/datasets/UNIX+User+Data, we were able to download not only the Unix commands but the sequence of commands used by each user.

The Unix command data summarizes the daily use of UNIX commands by users of a set of networked workstations. The cited paper deals with seven different users and the frequencies of use of five Unix commands in a day over 12 days. These commands are: “finger”, “man”, who”, “date” and “more”. Besides, authors generate synthetic data using different random distributions. In their case the classification does not takes into account the sequence of commands, but the relative frequency of use. We have download sessions of commands that are concatenated by date order to then apply our algorithm. This new experiment has allowed us to compare the results of the intrusion detection algorithms that are presented in that paper and ours on the same database. Besides, this example has allowed us also to compare the efficiency of the detection algorithm in terms of computational time and to extend the potential of our proposal to other real applications.

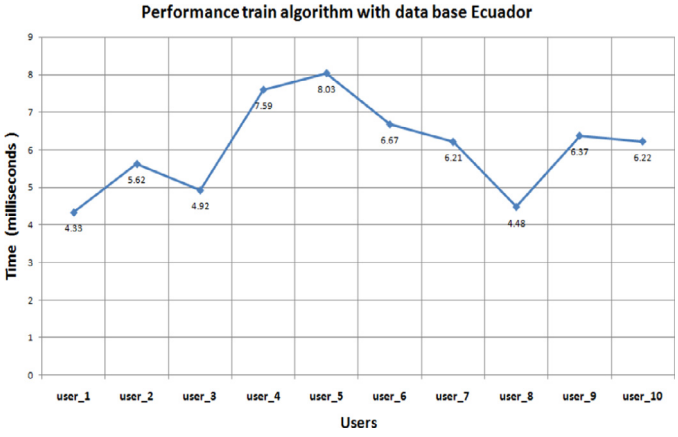



**Fig. 8.** Algorithm performance in milliseconds with the dataset from Ecuador.

The best classifier that is shown in that paper (Regularized Discriminant Analysis, RDA) gets 70.5% hits in the classification. In fact, among the 9 datasets used in that paper, the worst results are obtained for this Unix command database. For training, we used 500 sessions of other users as anomalous session samples. Again, we show the results without applying Markov chains (Table 5) and with this last step of the detection procedure (Table 6). We obtain a better classification of the nine users, identifying which of the users a particular sequence of operations belongs to or if it does not match any, in which case it could be a data leakage. The results were 68.84% of correctly classified sessions applying the algorithm up to step 3 and a detection rate of 96,03% with Markov chains.

The good result of our data leakage detection procedure on this public dataset gives us the conviction that it can also work well on other real databases, as it has been proved with the one we have presented in the paper.

### 5.1. Computational time considerations

The algorithm is quite efficient computationally; it takes it only between 4 to 8 milliseconds to give the result including training (Fig. 8) when applied to the dataset of real access in Ecuador.

That means that our decision system can be applied on-line. In fact, each time a user begins a new activity on the computer, our system is able to analyse the sequence of operations within the current working session.

Moreover, the detection algorithm may be running at same time employees are working because they do not realize that they are being monitored. The activity of this software could be detected by the users for two reasons:

i. Due to the fact that high overhead (processing time by monitoring) reduces the performance of the system itself and it may become unfriendly.

**Table 5**
Results of the algorithm (without Markov chain) for 9 users with the Unix dataset.

| Users | #Normal sessions | #Fraudulent sessions | Total sessions | TP | TN | FP | FN | AR | Error% | DR% | FPR% |
|---|---|---|---|---|---|---|---|---|---|---|---|
| user_0 | 567 | 500 | 1067 | 256 | 263 | 304 | 244 | 48.64 | 51.36 | 51.2 | 53.62 |
| user_1 | 515 | 500 | 1015 | 275 | 246 | 269 | 225 | 51.33 | 48.67 | 55 | 52.23 |
| user_2 | 1069 | 500 | 1569 | 402 | 821 | 248 | 98 | 77.95 | 22.05 | 80.4 | 23.2 |
| user_3 | 501 | 500 | 1001 | 468 | 327 | 174 | 32 | 79.42 | 20.58 | 93.6 | 34.73 |
| user_4 | 955 | 500 | 1455 | 314 | 723 | 232 | 186 | 71.27 | 28.73 | 62.8 | 24.29 |
| user_5 | 582 | 500 | 1082 | 347 | 386 | 196 | 153 | 67.74 | 32.26 | 69.4 | 33.68 |
| user_6 | 3419 | 500 | 3919 | 471 | 2481 | 938 | 29 | 75.33 | 24.67 | 94.2 | 27.43 |
| user_7 | 1522 | 500 | 2022 | 417 | 1124 | 398 | 83 | 76.21 | 23.79 | 83.4 | 26.15 |
| user_8 | 713 | 500 | 1213 | 329 | 540 | 173 | 171 | 71.64 | 28.36 | 65.8 | 24.26 |
| Total | 9843 | 4500 | 14,343 | Average | | | | 68.84 | 31.16 | 72.87 | 33.29 |

**Table 6**
Results of the algorithm (applying Markov chains) for 9 users with Unix dataset.

| Users | #Normal sessions | #Fraudulent sessions | Total sessions | TP | TN | FP | FN | AR% | Error% | DR% | FPR% |
|---|---|---|---|---|---|---|---|---|---|---|---|
| user_0 | 567 | 500 | 1067 | 481 | 500 | 19 | 67 | 91.94 | 8.06 | 96.2 | 3.66 |
| user_1 | 515 | 500 | 1015 | 484 | 508 | 16 | 7 | 97.73 | 2.27 | 96.8 | 3.05 |
| user_2 | 1069 | 500 | 1569 | 478 | 1033 | 22 | 36 | 96.3 | 3.7 | 95.6 | 2.09 |
| user_3 | 501 | 500 | 1001 | 490 | 496 | 10 | 5 | 98.5 | 1.5 | 98 | 1.98 |
| user_4 | 955 | 500 | 1455 | 488 | 902 | 12 | 53 | 95.53 | 4.47 | 97.6 | 1.31 |
| user_5 | 582 | 500 | 1082 | 485 | 543 | 15 | 39 | 95.01 | 4.99 | 97 | 2.69 |
| user_6 | 3419 | 500 | 3919 | 493 | 3219 | 7 | 200 | 94.72 | 5.28 | 98.6 | 0.22 |
| user_7 | 1522 | 500 | 2022 | 491 | 1498 | 9 | 24 | 98.37 | 1.63 | 98.2 | 0.6 |
| user_8 | 713 | 500 | 1213 | 481 | 685 | 19 | 28 | 96.13 | 3.87 | 96.2 | 2.7 |
| Total | 9843 | 4500 | 14,343 | Average | | | | 96.03 | 3.97 | 97.13 | 2.03 |

ii. If this overhead is detected by fraudulent users, they can change their *modus operandis*, and therefore cheat the system.

Therefore, overhead must be controlled and have to be very small. Regarding performance it is usually accepted a 10%overhead for exhaustive monitoring and less than 0.5% for a regular case. The most classical overhead formula in performance measurement of computer systems is the ratio "execution time of the monitor software/time between two activations" (in seconds) [16].

Assuming up to 25 tasks per session, the maximum execution time of our system is 0.00112 seconds for the worst case (longest tasks). The time between two activations in an exhaustive monitoring is the accumulative time of such sequence of tasks, which can be limited by 25 times the shortest execution time of the tasks, in our case $25\,x\,(0.00019 + Z)$, where Z is the reflection time of the user. Then, assuming $Z = 25$ s (1 s per task),

$$overhead <= \frac{0.00112}{25x(0.00019 + 25)} = 0.0000447$$

Therefore, the algorithm produces a maximum overhead of 0.00447% which is very acceptable as it is lower than 0.05%. Besides, for an automated monitoring that could be set every 30 seconds (instead monitoring at event 25), the decision system gets an overhead of 0.037%, still within an acceptable range.

## 6. Conclusions remarks

This paper presents an algorithm for data leakage detection based on user's anomalous behaviour identification. It is structured in consecutive steps, the last one to double-check the decision on whether or not it is an attempted attack. The algorithm works with the probability of sequences of the tasks performed by a user into a computer. It has been proved efficient in terms of accuracy and with a very low false positive rate, which is one of its main advantages.

Another contribution of this paper is the representation of the users' behaviour. The codification of the tasks (operations, files) of each user has been performed defining a data structure that allows us to implement different size of sequences and sessions, i.e., it is very flexible.

Besides, after the analysis of the computational time the algorithm requires, we have drawn the conclusion that it can be used as a software monitor since it does not overhead the working time. In other words, when a user is executing a sequence of activities, the detection system monitors his activity at the same time but without slowing down the pace of work. Moreover, it can be implemented in any computer system and it is scalable.

The algorithm has been validated using real data that come from a governmental institution of Ecuador. The test results prove the good performance of this decision system for data leakage detection (94,78% DR). In addition, this algorithm was not only applied to that institutional database but it has also been applied to a public database of Unix commands with very good results (96,03%). This proves the potential of this algorithm to be applied to any other real problems as long as the data can be codified as sequences.

We think that the main limitation of our proposal is that it is a supervised algorithm, it works with historical data, and therefore we need information from the users to generate their behavioural pattern (in our case at least data of the last three months). In fact, if the sequence of tasks carried out by a user in a session is too short, the system would be not able to detect whether it is normal or not. Another limitation of our approach is that it is focused on detection and not prevention.

As future work we should be able to analyze a higher number of subsequences of tasks at the same time in order to make it quicker. Another way of improving the detection system would be to add extra information from other sources that could help to define a more complete user behavioural profile and so to enhance the reliability of the system. In a near future we would like to apply fuzzy logic in order to make a decision on blocking the system or raising an alarm, and to improve the results of the first step of the algorithm using neural networks.

## Acknowledgements

This work has been partially supported by the Ministry of Higher Education, Science, Technology and Innovation (SENESCYT)

of the Government of the Republic of Ecuador under the scholarship “Convocatoria Abierta 2011 y2012”.

## References

[1] S. Aeberhard, D. Coomans, O. De Vel, Comparative analysis of statistical pattern recognition methods in high dimensional settings, Pattern Recognit. 27 (8) (1994) 1065–1077.
[2] M. Ahmed, A.N. Mahmood, M.R. Islam, A survey of anomaly detection techniques in financial domain, Future Gener. Comput. Syst. 55 (2016) 278–288.
[3] A.S.A. Aziz, E.L. Sanaa, A.E. Hassanien, Comparison of classification techniques applied for network intrusion detection and classification, J. Appl. Logic (2016).
[4] P. Chatziadam, I.G. Askoxylakis, N.E. Petroulakis, A.G. Fragkiadakis, Early warning intrusion detection system, in: Trust and Trustworthy Computing, Springer International Publishing, 2014, pp. 222–223.
[5] S.N. Chari, P.C. Cheng, BlueBox: a policy-driven, host-based intrusion detection system, ACM Trans. Inf. Syst. Secur. (TISSEC) 6 (2) (2003) 173–200.
[6] J. Chavan, P. Desai, Data leakage detection using data allocation strategies, Int. J. Adv. Eng. Technol. 6 (5) (2013) 2033.
[7] Cisco, Data Leakage Worldwide: Common Risks and Mistakes Employees Make, 2008 http://www.cisco.com/c/en/us/solutions/collateral/enterprise-networks/data-loss-prevention/white_paper_c11-499060.html accessed December, 2016.
[8] E. Costante, S. Vavilis, S. Etalle, J. den Hartog, M. Petkovic, N. Zannone, Database anomalous activities detection and quantification, in: Security and Cryptography (SECRYPT), 2013 International Conference on, IEEE, 2013, pp. 1–6.
[9] A.S. Eesa, Z. Orman, A.M.A. Brifcani, A novel feature-selection approach based on the cuttlefish optimization algorithm for intrusion detection systems, Expert Syst. Appl. 42 (5) (2015) 2670–2679.
[10] S. Elhag, A. Fernández, A. Bawakid, S. Alshomrani, F. Herrera, On the combination of genetic fuzzy systems and pairwise learning for improving detection rates on Intrusion Detection Systems, Expert Syst. Appl. 42 (1) (2015) 193–202.
[11] S. Ghorbanian, G. Fryklund, S. Axelsson, Do data loss prevention systems really work? in: Advances in Digital Forensics XI, Springer International Publishing, 2015, pp. 341–357.
[12] C. Guevara, M. Santos, J.A. Martin, V. Lopez, Reaching a consensus on access detection by a decision system, in: Progress in Informatics and Computing (PIC), 2014 International Conference on, IEEE, 2014, pp. 119–122.
[13] C. Guevara, M. Santos, V. López, Data leakage detection algorithm based on sequences of activities, in: Research in Attacks, Intrusions and Defences: 17th International Symposium, RAID 2014, Vol. 8688, Springer, 2014, p. 477.
[14] C.B. Guevara, M. Santos, M.V. Lopez, Negative selection and Knuth Morris Pratt algorithm for anomaly detection, IEEE Lat. Am. Trans. 14 (3) (2016) 1473–1479.
[15] C.B. Guevara, M. Santos, V. López, Intrusion detection with neural networks based on knowledge extraction by decision tree, in: Advances in Intelligent Systems and Computing, 527, Springer, 2016, pp. 508–517.
[16] X. Huang, J. Seyster, S. Callanan, K. Dixit, R. Grosu, S.A. Smolka, S.D. Stoller, E. Zadok, Software monitoring with controllable overhead, Int. J. Softw. Tools Technol. Trans. 14 (3) (2012) 327–347.
[17] C.L. Huth, D.W. Chadwick, W.R. Claycomb, I. You, Guest editorial: a brief overview of data leakage and insider threats, Inf. Syst. Front. 15 (1) (2013) 1.
[18] A. Kamra, E. Terzi, E. Bertino, Detecting anomalous access patterns in relational databases, VLDB J. 17 (5) (2008) 1063–1077.
[19] J.G. Kemeny, J.L. Snell, Finite Markov Chains, Vol. 356, van Nostrand, Princeton, NJ, 1960.
[20] J. Kim, H.J. Kim, Design of internal information leakage detection system considering the privacy violation, in: Information and Communication Technology Convergence (ICTC), 2010 Int. Conf. on, IEEE, 2010, pp. 480–481.
[21] A. Kumar, A. Goyal, N.K. Chaudhary, S. Sowmya Kamath, in: Comparative evaluation of algorithms for effective data leakage detection, In *Information & Communication Technologies (ICT), 2013 IEEE Conference on*, IEEE, 2013, pp. 177–182.
[22] T. Lewellen, A.P. Moore, D.M. Cappelli, R.F. Trzeciak, D. Spooner, R.M. Weiland, Carnegie-Mellon Univ Pittsburgh PA Software Engineering Inst, 2012.
[23] D. Li, S. Liu, H. Zhang, A negative selection algorithm with online adaptive learning under small samples for anomaly detection, Neurocomputing 149 (2015) 515–525.
[24] H.J. Liao, C.H.R. Lin, Y.C. Lin, K.Y. Tung, Intrusion detection system: a comprehensive review, J. Netw. Comput. Appl. 36 (1) (2013) 16–24.
[25] Z. Liu, Y. Lai, A data mining framework for building intrusion detection models based on IPv6, in: Advances in Information Security and Assurance, Springer Berlin Heidelberg, 2009, pp. 608–618.
[26] S. Mathew, M. Petropoulos, H.Q. Ngo, S. Upadhyaya, A data-centric approach to insider attack detection in database systems, in: Recent Advances in Intrusion Detection, Springer Berlin Heidelberg, 2010, pp. 382–401.
[27] C. Modi, D. Patel, B. Borisaniya, H. Patel, A. Patel, M. Rajarajan, A survey of intrusion detection techniques in cloud, J. Netw. Comput. Appl. 36 (1) (2013) 42–57.
[28] A. Nadeem, M. Howarth, Protection of MANETs from a range of attacks using an intrusion detection and prevention system, Telecommun. Syst. 52 (4) (2013) 2047–2058.
[29] P. Papadimitriou, H. Garcia-Molina, Data leakage detection, Knowl. Data Eng. IEEE Trans. 23 (1) (2011) 51–63.
[30] W. Sha, Y. Zhu, T. Huang, M. Qiu, Y. Zhu, Q. Zhang, A multi-order Markov chain based scheme for anomaly d, in: Computer Software and Applications Conference Workshops (COMPSACW), 2013 IEEE 37th Annual, IEEE, 2013, pp. 83–88.
[31] A. Shabtai, Y. Elovici, L. Rokach, A survey of data leakage detection and prevention solutions, Springer Brief in Computer Science, Springer, Science & Business Media, 2012.
[32] R. Sivakami, M Jaiganesh, R. Saravanan, An efficient fuzzy self-classifying clustering based framework for cloud security, Int. J. Comput. Intell. Syst. 10 (2017) 495–506.
[33] L. Torgo, E. Lopes, Utility-based fraud detection, in: In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, Vol. 22, 2011, p. 1517.
[34] S. Vavilis, A. Egner, M. Petković, N. Zannone, An anomaly analysis framework for database systems, Comput. Secur. 53 (2015) 156–173.
[35] S.Y. Wu, E. Yen, Data mining-based intrusion detectors, Expert Syst. Appl. 36 (3) (2009) 5605–5612.
[36] F. Xu, M. Kwan, H. Tse, K.P. Chow, A bayesian belief network for data leakage investigation, in: Proceedings of the 2nd International Workshop on Security and Forensics in Communication Systems, ACM, 2014, pp. 19–24.
[37] N. Ye, Q. Chen, An anomaly detection technique based on a chi-square statistic for detecting intrusions into information systems, Qual. Reliab. Eng. Int. 17 (2) (2001) 105–112.
[38] N. Ye, S.M. Emran, Q. Chen, S. Vilbert, Multivariate statistical analysis of audit trails for host-based intrusion detection, Comput. IEEE Trans. 51 (7) (2002) 810–820.
[39] D.Y. Yeung, Y. Ding, Host-based intrusion detection using dynamic and static behavioral models, Pattern Recognit. 36 (1) (2003) 229–243.

Please cite this article as: C. Guevara et al., Data leakage detection algorithm based on task sequences and probabilities, Knowledge-Based Systems (2017), http://dx.doi.org/10.1016/j.knosys.2017.01.009