Cruzando Fronteras:
Tendencias de Contabilidad Directiva para el Siglo XXI

DISEÑO DE UN SISTEMA BORROSO PARA LA DETECCIÓN DE INTRUSOS

Enrique López González (*)
Angela Díez Díez (**)
Francisco J. Rodríguez Sedano (**)
Cristina Mendaña Cuervo (*)
Jesús Calabozo Morán (**)
(*) Departamento de Dirección y Economía de la Empresa
(**) Departamento de Ingeniería Eléctrica y Electrónica
Universidad de León,
Campus de Vegazana, s/n. 24071. León (España).

Resumen

En los últimos años, debido al incremento en el uso de los ordenadores y la emergencia del comercio electrónico, la detección de intrusos se ha convertido en una prioridad importante, pues no resulta técnicamente factible construir un sistema invulnerable.

La hipótesis de la investigación es que la Lógica Borrosa es capaz de producir "mejores" reglas que incrementen la flexibilidad y robusted de los sistemas de auditoría informática. De hecho, los Sistemas Basados en Reglas Borrosas (SBRBs) han demostrado ser una herramienta muy efectiva para el diseño de Sistemas Inteligentes de Análisis de Datos para problemas de control, clasificación, modelado etc., en aquellos contextos donde la información y/o los datos están afectados de imprecisión no probabilística. De esta forma, el principal propósito del presente del trabajo consistirá en elaborar un modelo de Sistema Experto Borroso aplicado a la Detección de Intrusos, para ello nuestra propuesta consiste en profundizar en los dos aspectos siguientes: el diseño de un SBRB y estudiar la adaptación de estos modelos en "minería de datos" relativos a la detección de intrusos.

Palabras clave: Auditoría Informática; Detección de Intrusos; Minería de Datos; Sistemas Basados en Reglas Borrosas (SBRBs).

1. Introducción

Nuestra sociedad depende cada vez más del acceso y procesamiento rápido de información, lo que ha supuesto la proliferación del empleo de los ordenadores y de las redes de comunicaciones y con ello la existencia de problemas de acceso desautorizado y manipulación de datos: en multitud de ocasiones los intrusos de la red han superado los mecanismos de autentificación diseñados para proteger los sistemas, pues con el aumento en el entendimiento sobre cómo funcionan los sistemas, los intrusos se han vuelto expertos en determinar las debilidades, empleando diversos niveles de engaño antes de irrumpir en un sistema determinado, intentando cubrir sus huellas para que su actividad en el sistema no se descubra fácilmente. De ahí que, la detección de intrusos se haya convertido en una prioridad importante, pues no resulta técnicamente factible construir un sistema totalmente invulnerable: el propio concepto de seguridad es en si mismo “borroso”.

La hipótesis principal que orienta nuestro trabajo consiste en que las denominadas “técnicas inteligentes” son capaces de producir "mejores" reglas que incrementen la flexibilidad y robusted de los sistemas de auditoría informática. De hecho, los Sistemas Basados en Reglas Borrosas (SBRBs) han demostrado ser una herramienta muy efectiva para el diseño de Sistemas Inteligentes de Análisis de Datos para problemas de control, clasificación, modelado etc., en aquellos contextos donde la información y/o los datos están afectados de imprecisión no probabilística. El sustrato teórico de manejo de estos sistemas es la Lógica Borrosa, cuyas reglas de inferencia permiten construir algoritmos eficientes para la gestión de los SBRBs [Bardossy, Duckstein, 1995].

El diseño de un SBRBs contempla dos aspectos principales. Por un lado, y principal centro de atención del presente trabajo, la selección del modelo de inferencia en un SBRBs es un aspecto que ha sido ampliamente estudiado tanto para problemas de modelado y control como para problemas de clasificación. A este respecto, en [Cordón, del Jesús, Herrera y López, 1997] se puede encontrar un estudio de las diferentes propuestas existentes en la literatura especializada, así como nuevas propuestas que permiten inferir a partir de la base de reglas de forma cooperativa. Por otro lado, la construcción de la base de reglas Borrosas, inicialmente se carecía de procedimientos sistemáticos generales para obtener una base de reglas lo que hizo que muchas de las aplicaciones que se desarrollaron siguiesen un enfoque de ensayo-error. Actualmente, se estudian diferentes técnicas para el aprendizaje de bases de reglas Borrosas, tanto a partir de ejemplos como a partir del modelo del sistema. Aunque la construcción de las bases de reglas es un aspecto crucial para el diseño de SBRBs, también es el mas difícil, por lo que no es sorprendente el gran número

de intentos que se han hecho para automatizar esta tarea, y parece claro que esta línea de aprendizaje automático es la adecuada.

En el ámbito de la detección de intrusos cabe encontrarse con frecuencia con grandes volumenes de datos, que incluyen un número grande de variables y de ejemplos. Resulta bastante claro que el valor de todos esos datos recae en la capacidad de los usuarios para extraer reglas útiles, descubrir hechos y tendencias, etc. Este es el terreno de acción de lo que se ha dado en denominar "minería de datos" [Cios, Pedrycz, Swiniarski 1998]. La obtención de reglas de asociación para estos problemas entraña una mayor dificultad, los algoritmos de obtención necesitan ser muy eficientes por cuanto el conjunto de variables y ejemplos son muy grandes. En ocasiones, el aprendizaje es dirigido porque se conoce la variable-clasificación a caracterizar, pero en otros muchos casos la búsqueda no puede ser dirigida, lo cual dificulta la precisión en la misma. De ahí que resulte común encontrarse con los problemas de complejidad para la obtención de bases de reglas, lo que supone la necesidad de intentar tener reglas genéricas en cuanto al numero de variables e información asociada a las mismas.

2.- Descubriendo las amenazas en auditoría informática: la detección de intrusos

La mayoría de sistemas de computación proporcionan un mecanismo de control de acceso como su primera línea de defensa. Sin embargo, esto sólo limita si el acceso a un objeto en el sistema se permite pero no planea o restringe lo que un sujeto puede hacer con el propio objeto si tiene el acceso para manipularlo. El control de acceso, por consiguiente, no planee y no pueda prevenir el flujo de información sin autorización a través del sistema porque tales flujos pueden tener lugar con los accesos autorizados.

Es más, en sistemas dónde el control de acceso es discrecional, la responsabilidad de la protección de los datos recae sobre el usuario final. Esto requiere a menudo que los usuarios entiendan el mecanismo de protección ofrecido por el sistema y cómo lograr la seguridad deseada usando estos mecanismos.

Pueden controlarse los flujos de información para reforzar la seguridad aplicando a modelos tal como la Campana (Bell) y el modelo LaPadula para proporcionar la confidencialidad, o el modelo Biba para proporcionar la integridad. Sin embargo, la seguridad se consigue a costa de la comodidad. Ambos modelos son conservadores y restringen las operaciones de lectura y escritura para asegurar la confidencialidad e integridad de los datos en el sistema. Si ambos modelos se usa conjuntamente, el modelo resultante sólo permite los accesos a los objetos en el mismo nivel de la clasificación de seguridad que el sujeto. Así, un sistema completamente seguro puede que no sea muy útil.

Los controles de acceso y los patrones de protección no son útiles contra las amenazas internas o las concesiones del módulo de autentificación. Si una contraseña es débil y se compromete, las medidas de control de acceso no pueden prevenir la pérdida o corrupción de la información que el usuario fue autorizado a acceder. En general, los métodos estáticos para asegurar las propiedades de seguridad en un sistema simplemente pueden ser insuficientes, o hace los sistema demasiado restrictivo a sus usuarios. Por ejemplo, las técnicas estáticas no pueden prevenir la violación de la política de seguridad que es el resultado de examinar ficheros de datos; y los controles de acceso obligatorios que sólo permiten el acceso de usuarios a datos para que ellos tengan la autorización apropiada hacen el sistema engorroso al uso. Un método dinámico, tal como rastreo del comportamiento, se necesita descubrir y quizás prevenir las brechas en la seguridad.

Las dificultades de diseñar software complejo, libre de errores es improbable que se resuelvan en un futuro cercano. Los fallos en el software del sistema son a menudo ponen de manifiesto las debilidades en la seguridad. Es más, los ciclos de vida del software están acortándose continuamente la causa del aumento de la competitividad del mercado aumentada. Esto produce a menudo diseños pobres o pruebas inadecuadas, agravando más el problema.

> Nosotros debemos tener las medidas necesarias para descubrir las brechas de seguridad, es decir, identificar intrusos e intrusiones. Los sistemas de detección de la intrusión cumplen este papel y normalmente forman la última línea de defensa en el esquema de la protección global de un sistema de computación. Ellos no sólo son útiles para descubrir con éxito brechas en la seguridad, sino también para supervisar los esfuerzos para romper la seguridad, que mantiene información importante para tomar las contramedidas oportunas.

Así, los sistemas de detección de intrusos incluso son útiles cuando se toman medidas preventivas fuertes para proteger el sistema de computación con un alto grado de confianza en su seguridad. Además, las medidas preventivas tales como las reparaciones de fallos del software de sistema no siempre pueden ser preferibles al descubrimiento de su utilización desde una práctica consideración del costo-beneficio. Reparar los virus no es posible sin la fuente del software y conocimientos técnicos precisos, y el gran despliegue de parches puede requerir procedimientos de instalación más difíciles que poner al día el banco de datos de la detección de la intrusión, sobre todo cuando el software se personaliza para el uso local a los sitios individuales. En el caso de programas grandes y complejos, como el envío de correo electrónico, no puede ser posible la reparación de todos sus posibles defectos igual que cuando su código fuente está disponible. Los métodos genéricos de supervisión de explotación de las vulnerabilidades pueden ser muy útiles estos casos.

De acuerdo con lo anterior, se puede definir una intrusión como aquellas acciones que intentan poner en peligro la integridad, confidencialidad, o disponibilidad de un recurso". También, se puede entender por intrusión a una violación de la política de seguridad del sistema.

Estas definiciones sobre este término son en general suficientes para abarcar todas las amenazas mencionadas en anteriormente, pero conviene poner de manifiesto que cualquier definición de intrusión es, por necesidad, imprecisa, como los requisitos de política de seguridad no siempre se traducen en un conjunto bien definido de acciones. Mientras que la política define las metas que deben satisfacerse en un sistema, los detectores de brechas de esta política requieren conocimiento de pasos o acciones que pueden producir su violación.

La detección de intrusos puede ser dividida en dos categorías principales: la detección de intrusión de anomalías y la detección de intrusión de mal uso (abuso). La primera se refiere a intrusiones que pueden descubrirse basadas en la conducta anómala y uso de recursos de computación. Por ejemplo, si el X usuario sólo usa la computadora de su oficina entre las 9:00 a.m. y las 5 p.m., una actividad en su cuenta fuera de ese horario es anómalo y, por lo tanto, puede ser una intrusión. Posteriormente, considérese a otro usuario Y, que siempre pueda conectarse fuera de las horas de trabajo a través del servidor de la compañía. Una sesión del "login" remota nocturna a su cuenta podría ser considerada extrana, anomala o simplemente "rara". La detección de la anomalía intenta cuantificar la conducta usual o aceptable y señala cualquier conducta irregular como un intruso potencial.

Algunos autores clasifican las amenazas como incursiones externas, incursiones internas, y engaños (*misfeasance*) y usan esta clasificación para desarrollar un sistema de vigilancia basado en la detección de anomalías en la conducta de los usuarios. Se definen las incursiones externas como intrusiones que se llevan a cabo por los usuarios del sistema de computación sin autorización; las incursiones internas son aquéllas que se llevan a cabo por los usuarios autorizados de sistemas de computación que no están autorizados para utilizar ciertos datos; y el engaño se define como el mal uso de datos y otros recursos por parte de usuarios autorizados.

En el contraste, la detección de intrusión de mal uso se refiere a intrusiones que siguen modelos bien definidos de ataque que explotan las debilidades en el sistema y en el software de aplicación. Precisamente tales modelos pueden escribirse por adelantado. Por ejemplo, la explotación de los virus del envío de correo electrónico usados en ataques por Internet estarían catalogados bajo esta categoría. Esta técnica representa el conocimiento sobre la conducta mala o inaceptable y busca descubrirlo directamente, en contraposición a

la detección de intrusión de anomalías que busca descubrir el complemento de la conducta normal.

3.- Construcción de un Sistema Borroso para la Detección de Intrusos

El sistema elaborado esta encaminado a la determinación del nivel de intrusión que se puede producir en un sistema informático a partir del análisis de una serie de variables utilizadas en la detección de intrusos. Para ello, se utilizan como variables de entrada las siguientes:

- *Horario*: indica si la intrusión se produce en horario laboral o fuera de dicho horario. Será calificado como horario laboral o no laboral.
- *Directorios de sistema y/o datos*: indica si el intruso accede a directorios de sistema y/o directorios de datos. Será calificado como accede (a dichos directorios) o no accede (a los directorios).
- *Uso de comandos peligrosos*: indica si el intruso utiliza determinados comandos considerados como peligrosos (ej. copiar, mover, modificar, borrar, crear,....). Será calificado como utiliza (dichos comandos) o no utiliza (los comandos).
- *Tiempo CPU*: tiempo uso de la CPU. Permitirá su clasificación como bajo, medio o alto.
- *Comandos usados*: número de comandos diferentes utilizados. Será calificado como bajo, medio o alto.
- *Tiempo de Uso de I/O*: tiempo de uso de los distintos dispositivos de I/O. En función del mismo se procederá a su clasificación como bajo, medio o alto.

Las tres primeras las podemos considerar como variables binarias ya que solo miden si la variable se utiliza o no. Las tres siguientes se pueden considerar como variables ordinales ya que miden algunos comportamientos cuantificables numéricamente.

Como variable de salida se utilizará el nivel de intrusión, previa calificación del mismo como informativo, sospechoso, serio o crítico.

3.1.- Determinación de la relación entre las variables

La primera actividad a realizar consiste en la determinación de las relaciones existentes entre las diferentes variables del modelo, es decir, establecer el expertizaje que

permite dar una salida a partir de los diferentes valores de las entradas. En este caso, dichas relaciones se pueden observar en la siguiente Figura 1.

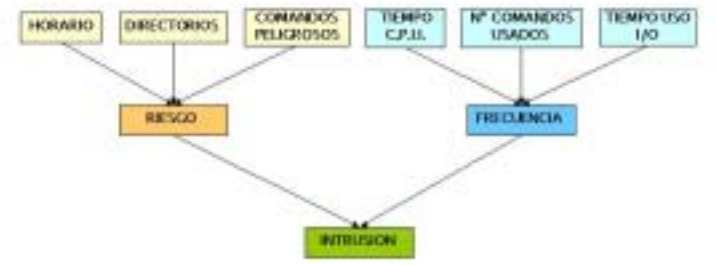


Figura 1. Esquema asociativo entre variables

A la vista del esquema anterior, puede observarse lo siguiente:

- Las tres primeras variables (horario, directorios y comandos peligrosos) se agrupan para dar lugar a otra variable intermedia llamada riesgo que determinará el riesgo que supone para el sistema la posible intrusión. Esta variable se califica como muy bajo, bajo, medio, alto o muy alto.
- Las tres últimas variables (tiempo C.P.U., número de comandos usados y tiempo uso I/O) se agrupan a su vez en otra variable intermedia llamada frecuencia que determinará la periodicidad con que se produce la posible intrusión. Esta variable se califica como muy baja, baja, media, alta y muy alta.
- Para determinar el nivel de intrusión se tendrán en cuenta estas dos variables intermedias.

3.2. Fuzzificación de las variables

Una vez que conocemos las variables que vamos a utilizar y las relaciones existentes entre las mismas, el siguiente paso a seguir es determinar el dominio de cada uno de los números borrosos representativos de las diferentes etiquetas lingüísticas en las que se divide el universo de discurso de cada variable. Obviamente, no tiene sentido hablar del dominio de las tres primeras variables ya que, al ser variables binarias, solo pueden tener los valores 1 ó 0.

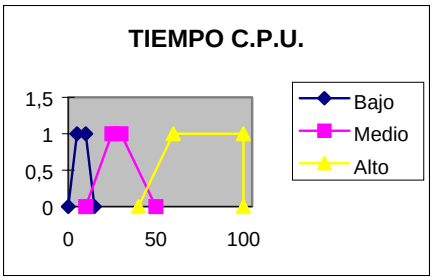
3.2.1.- Variable Tiempo C.P.U.

Se han diferenciado dentro de esta variable los siguientes estados: tiempo bajo, tiempo medio y tiempo alto. Los dominios asignados a los diferentes números borrosos relacionados con esta variable son:

- Tiempo C.P.U. Bajo = (0, 5, 10, 15)
- Tiempo C.P.U. Medio = (10, 25, 30, 50)
- Tiempo C.P.U. Alto = (40, 60,100, 100)

De acuerdo con lo anterior, los valores asignados a esta variable se recogen el Cuadro 1.

	Bajo	Medio	Alto
No menor que	0	10	40
Igual que	5	25	60
Igual que	10	30	100
No mayor que	15	50	100

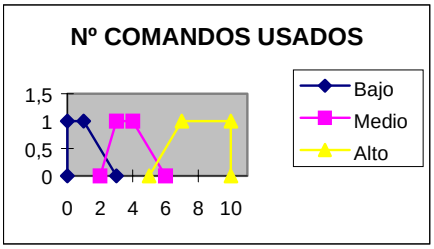


Cuadro 1.

3.2.2.- Variable Número de comandos usados

La variable "número de comandos usados" se ha modelizado a través de tres etiquetas lingüísticas: bajo, medio y alto. Al igual que en el caso anterior, se determinaron los dominios de cada uno de los subconjuntos borrosos representativos de tales estados; siendo los valores establecidos y su representación gráfica la que se muestra en el Cuadro 2.

	Bajo	Medio	Alto
No menor que	0	2	5
Igual que	0	3	7
Igual que	1	4	10
No mayor que	3	6	10

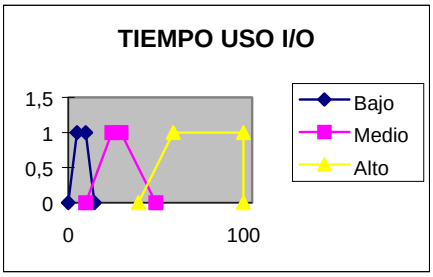


Cuadro 2.

3.2.3. Variable Tiempo uso I/O

La tercera de las variables consideradas como entradas al sistema es el tiempo de uso del sistema de I/O. En este sentido, se consideraron tres posibles estados de esta variable: bajo, medio y alto. Los números borrosos asociados a estas tres etiquetas y la representación gráfica de los mismos son los recogidos en el Cuadro 3.

	Bajo	Medio	Alto
No menor que	0	10	40
Igual que	5	25	60
Igual que	10	30	100
No mayor que	15	50	100

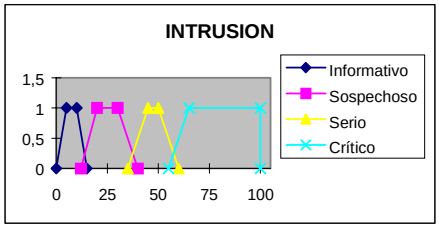


Cuadro 3.

3.2.4. Variable Intrusión

Además de las variables de entrada, se ha establecido como única variable de salida el nivel de intrusión que indicará en qué grado podemos considerar que la conexión al sistema la ha establecido un intruso. Dicha variable de salida la calificamos según cuatro etiquetas lingüísticas: informativo, sospechoso, serio y critico, siendo la representación gráfica y los números borrosos asociados a las mismas los que se recogen en el Cuadro 4.

	Inform.	Sosp.	Serio	Crítico
No menor que	0	12	35	55
Igual que	5	20	45	65
Igual que	10	30	50	100
No mayor que	15	40	60	100



Cuadro 4.

3.3. Establecimiento de las reglas

Una vez establecidas las definiciones de cada una de las etiquetas lingüísticas en las que se dividen las variables borrosas establecidas (tanto de entrada como de salida), y conocidas las relaciones entre las mismas el siguiente paso es el establecimiento de reglas borrosas del tipo *SI* ..... *ENTONCES* .....

3.3.1. Variable intermedia "Riesgo"

Esta variable es el resultado de la combinación de las tres variables binarias (horario, directorios y comandos peligrosos), de ahí que sea necesario establecer las reglas de funcionamiento operativas para su formación, las cuales se muestran en el Cuadro 5.

Horario	Directorios	Comandos Peligrosos	Riesgo
Laboral	Accede	Utiliza	MEDIO
Laboral	Accede	No utiliza	BAJO
Laboral	No accede	Utiliza	MEDIO
Laboral	No accede	No utiliza	MUY BAJO
No laboral	Accede	Utiliza	MUY ALTO
No laboral	Accede	No utiliza	ALTO
No laboral	No accede	Utiliza	ALTO
No laboral	No accede	No utiliza	MEDIO

Cuadro 5.

3.3.2. Variable intermedia "Frecuencia"

De forma análoga a la variable anterior, esta variable es resultado de la combinación de las tres variables ordinales (tiempo CPU, nº comandos usados y tiempo uso I/O), siendo preciso, por tanto, establecer las reglas de funcionamiento operativas para su formación. Dichas reglas se recogen en el Cuadro 6.

3.3.3. Variable final "Intrusión"

La variable de salida es el resultado de la combinación de las dos variables intermedias (riesgo y frecuencia), lo que implica asimismo establecer las reglas de funcionamiento operativas para su formación, las cuales se muestran recogidas en el Cuadro 7.

Tiempo Cpu	Nº Comandos Utilizados	Tiempo Uso I/O	Frecuencia
Bajo	Bajo	Bajo	MUY BAJA
Bajo	Bajo	Medio	MUY BAJA
Bajo	Bajo	Alto	BAJA
Bajo	Medio	Bajo	BAJA
Bajo	Medio	Medio	BAJA
Bajo	Medio	Alto	BAJA
Bajo	Alto	Bajo	MEDIA
Bajo	Alto	Medio	MEDIA
Bajo	Alto	Alto	MEDIA
Medio	Bajo	Bajo	BAJA
Medio	Bajo	Medio	MEDIA
Medio	Bajo	Alto	MEDIA
Medio	Medio	Bajo	MEDIA
Medio	Medio	Medio	MEDIA
Medio	Medio	Alto	MEDIA
Medio	Alto	Bajo	MEDIA
Medio	Alto	Medio	ALTA
Medio	Alto	Alto	ALTA
Alto	Bajo	Bajo	MEDIA
Alto	Bajo	Medio	ALTA
Alto	Bajo	Alto	ALTA
Alto	Medio	Bajo	ALTA
Alto	Medio	Medio	ALTA
Alto	Medio	Alto	ALTA
Alto	Alto	Bajo	ALTA
Alto	Alto	Medio	MUY ALTA
Alto	Alto	Alto	MUY ALTA

Cuadro 6.

Riesgo	Frecuencia	Intrusión
Muy Bajo	Muy Baja	INFORMATIVO
Muy Bajo	Baja	INFORMATIVO
Muy Bajo	Media	INFORMATIVO
Muy Bajo	Alta	SOSPECHOSO
Muy Bajo	Muy Alta	SOSPECHOSO
Bajo	Muy Baja	INFORMATIVO
Bajo	Baja	INFORMATIVO
Bajo	Media	SOSPECHOSO
Bajo	Alta	SOSPECHOSO
Bajo	Muy Alta	SOSPECHOSO
Medio	Muy Baja	SOSPECHOSO
Medio	Baja	SOSPECHOSO
Medio	Media	SERIO
Medio	Alta	SERIO
Medio	Muy Alta	SERIO
Alto	Muy Baja	SERIO
Alto	Baja	SERIO
Alto	Media	SERIO
Alto	Alta	CRITICO
Alto	Muy Alta	CRITICO
Muy Alto	Muy Baja	CRITICO
Muy Alto	Baja	CRITICO
Muy Alto	Media	CRITICO
Muy Alto	Alta	CRITICO
Muy Alto	Muy Alta	CRITICO

Cuadro 7.

3.4. Funcionamiento del sistema

3.4.1. Variables de entrada

Una vez establecido el esquema general del sistema, es necesario determinar el funcionamiento del mismo. De esta forma, ante unos determinados datos de entrada ("***crisp***") en primer lugar será necesario determinar el grado de activación o grado de verdad de cada una de las etiquetas integrantes de las diferentes variables de entrada horario, directorios, comandos peligrosos, tiempo CPU, número de comandos usados y tiempo de uso I/O. Así, por ejemplo, para la variable *tiempo CPU* existen tres diferentes variables lingüísticas: Tiempo CPU medio y Tiempo CPU alto. El grado de activación de cada una ante una determinada entrada se determina como sigue:

– **Tiempo CPU bajo (0, 5, 10, 15):**

- Si el dato del ejemplo es menor o igual que 0, entonces el grado de pertenencia al subconjunto será 0.
- Si el dato es mayor o igual que 15, entonces el grado de pertenencia será 0.
- Si el dato es mayor o igual que 5 y menor o igual que 10, entonces el grado de pertenencia es 1.
- Si el dato es mayor que 0 y es menor que 5, entonces el grado de pertenencia es (dato-0)/(0-0).
- Si el dato es mayor que 10 y menor que 15, entonces el grado de pertenencia es (15-dato)/(15-10).

Lógicamente, el grado de pertenencia total al subconjunto se calcula como la suma de los grados de pertenencia resultantes.

Se procedería de la misma forma con todas las etiquetas lingüísticas de cada una de las variables de entrada.

3.4.2. Variable intermedia "Riesgo"

Para determinar el grado de pertenencia de esta variable, será necesario conocer el grado de verdad de la regla utilizada, o peso de la regla (que coincidirá con el grado de pertenencia buscado).

Dado que se trata de reglas compuestas por más de un antecedente, el grado de verdad de la regla determinará a partir del grado de verdad de cada uno de los antecedentes, relacionados a través de una T-norma. En este caso la T-norma que va a utilizarse será el producto, de tal forma que:

μ_t (RIESGO) = μ_x (HORARIO)*μ_y (DIRECTORIOS)*μ_z (COMANDOS)

Si observamos la tabla nos encontramos con que existen diferentes reglas con el mismo consecuente como por ejemplo:

SI Tiempo Horario laboral **y** directorios accede **y** comandos peligrosos utiliza **ENTONCES** Riesgo medio.
SI Tiempo Horario laboral **y** directorios no accede **y** comandos peligrosos utiliza **ENTONCES** Riesgo medio.

Para determinar del grado de pertenencia total al subconjunto Riesgo medio, tenemos que elegir una de las opciones propuesta por Sugeno o Mamdani; en nuestro caso hemos elegido la opción propuesta por Sugeno, ya que mantiene la unicidad en el grado de pertenencia total a los diferentes subconjuntos borrosos de una variable.

$\mu_{MEDIO\ TOTAL}$ (RIESGO) = μ_{MEDIO1} (RIESGO)+$\mu_{MEDIO\ 2}$ (RIESGO)+$\mu_{MEDIO\ 3}$ (RIESGO)

3.4.3. Variable intermedia "Frecuencia"

La variable borrosa *frecuencia* se obtiene a partir de los datos relativos a tres variables iniciales, tiempo CPU, número de comandos usados y tiempo uso I/O; esta variable tiene cuatro estados diferenciados: Frecuencia muy baja, Frecuencia baja, Frecuencia media, Frecuencia alta y Frecuencia muy alta.

Lógicamente, para determinar cada uno de estos estados será preciso utilizar reglas lógicas del tipo:

Para determinar el grado de pertenencia de FRECUENCIA, será necesario conocer el grado de verdad de la regla utilizada, o peso de la regla (que coincidirá con el grado de pertenencia buscado).

Dado que se trata de reglas compuestas por más de un antecedente, el grado de verdad de la regla determinará a partir del grado de verdad de cada uno de los antecedentes, relacionados a través de una T-norma. En este caso la T-norma que va a utilizarse será el producto, de tal forma que:

μ_t (FRECUENCIA) = μ_x (TIEMPO CPU)*μ_y (Nº COMANDOS)*μ_z (TIEMPO USO I/O)

Si se analiza la tabla se puede observar que existen diferentes reglas con el mismo consecuente, como por ejemplo:

SI Tiempo CPU bajo **y** Nº comandos bajo **y** Tiempo uso I/O bajo **ENTONCES** Frecuencia muy baja.
SI Tiempo CPU bajo **y** Nº comandos bajo **y** Tiempo uso I/O medio **ENTONCES** Frecuencia muy baja.

El grado de pertenencia total de cada consecuente se determinará a partir de la regla de Sugeno, es decir, como suma de los grados de pertenencia parciales de cada una de las reglas que lo definen.

$\mu_{MUY\ BAJA\ TOTAL}$ (FRECUENCIA) = $\mu_{MUY\ BAJA1}$ (FRECUENCIA)+$\mu_{MUY\ BAJA\ 2}$ (FRECUENCIA)

3.4.4. Variable final "Intrusión"

Para determinar el grado de pertenencia de la variable de salida "Riesgo", será necesario conocer el grado de verdad de la regla utilizada, o peso de la regla (que coincidirá con el grado de pertenencia buscado).

Dado que se trata de reglas compuestas por más de un antecedente, el grado de verdad de la regla determinará a partir del grado de verdad de cada uno de los antecedentes, relacionados a través de una T-norma. En este caso la T-norma que va a utilizarse será el producto, de tal forma que:

μ_z (INTRUSION) = μ_x (RIESGO)*μ_y (FRECUENCIA)

Dado el conjunto de normas definido previamente, se puede observar que al igual que en el caso de las variables intermedias (riesgo y frecuencia), existen diferentes reglas que dan lugar al mismo consecuente, por tanto, el grado de pertenencia total de cada consecuente se determinará a partir de la regla de Sugeno, es decir, como suma de los grados de pertenencia parciales de cada una de las reglas que lo definen.

3.5. Defuzzificación

Una vez obtenido el grado de pertenencia borroso de cada ejemplo a cada subconjunto borroso de la variable final (en este caso la evaluación del proveedor), el paso siguiente es determinar, a partir de dicha evaluación borrosa, la cantidad ***crisp*** (no borrosa) a pedir del mismo.

Para ello es necesario proceder a la defuzzificación de la figura borrosa obtenida, cuya representación gráfica se muestra en la Figura 2, en la que se puede observar que la misma responde al tipo de evaluación de reglas borrosas que se conoce con el nombre de ***producto aritmético***, y que se utiliza para la elaboración de sistemas expertos en incertidumbre.

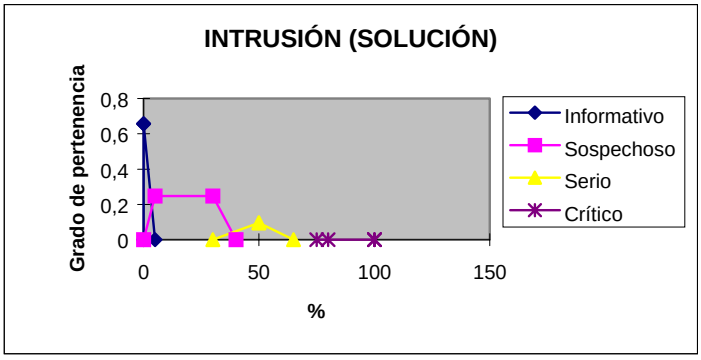


Figura 2.

Para conocer el nivel de intrusión, es necesario proceder a la defuzzificación de la figura anterior. Aunque existen múltiples métodos de defuzzificación, generalmente se aplica el método del centroide o centro de gravedad, que en el caso de trabajar con distintos subconjuntos borrosos de tipo trapezoidal se traduce en:

$$\text{Centroide} = \frac{\Sigma[a_1 + a_2 * (1 + \mu_A(y)) + a_3 * (1+\mu_A(y)) + a_4]_i}{\Sigma[1 + (1 + \mu_A(y)) + 1 + \mu_A(y)]_i}$$

Donde *i* hace referencia a los distintos subconjuntos borrosos existentes en la variable final (en este caso, la variable final es el nivel de intrusión con cuatro subconjuntos borrosos definidos: informativo, sospechoso, serio y critico).

De esta forma puede llegarse a determinar el nivel de intrusión de una conexión realizada al sistema.

4. Anexo: Ejemplo de interface de la aplicación

El interface de la aplicación es muy sencillo, ya que en una sola pantalla se pueden visualizar todas las variables, tanto las de entrada, como las variables intermedias como la variable de salida. A modo ilustrativo se muestra la pantalla en la Figura 3.
En un primer bloque se encuentran las variables de entrada clasificadas como binarias: horario, directorios y comandos peligrosos. Solo hay que seleccionar el valor de dichas variables haciendo clic en la casilla correspondiente.

En el siguiente bloque se recogen las variables de entrada ordinales: tiempo CPU, nº de comandos peligrosos y tiempo de uso I/O. El valor de estas variables se puede elegir desplazando la barra con el ratón hasta visualizar a la derecha el valor correcto dentro del rango establecido para cada variable.

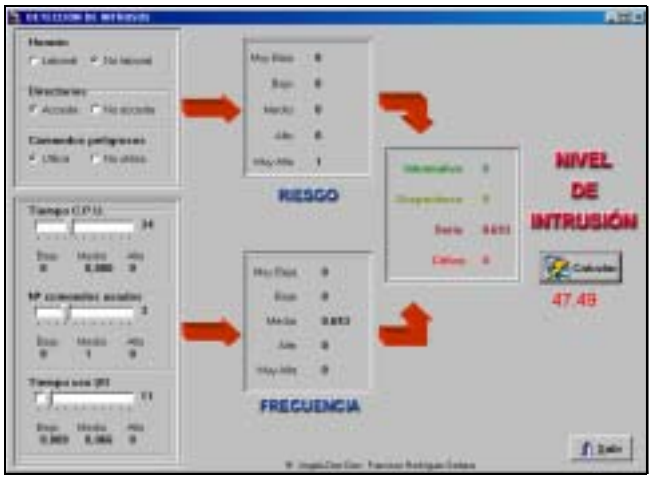


Figura 3.

En un segundo bloque tenemos las dos variables intermedias: riesgo y frecuencia. Cada vez que se cambie un valor de las variables de entrada correspondientes, el programa de aplicación actualiza los valores de las variables intermedias.

Por último se establece la variable de salida: nivel de intrusión, que también se actualiza automáticamente.

Para calcular el valor ***crisp*** de esta variable de salida, solo hay que pulsar el botón

De esta forma visualizamos el valor en la parte inferior de dicho botón.

5. Bibliografía

Agrawal, R., Imielinski, T. and Swami, A. (1993): "Mining association rules between sets of items in large databases". Proceedings of the ACM SIGMOD Conference on Management of Data, págs.207-216.

Axelsson, S. (1999): "On a Difficulty of Intrusion Detection". Departament of Computer Engineering. Chalmers University of Technology Göteborg, Sweden.

Axelsson, S., Lindqvist, U., Gustafson, U. and Jonsson, E. (1998): "An approach to UNIX security logging". Proceedings of the 21st National Information Systems Security Conference, págs. 62--75, Crystal City, Arlington, October 5--8, National Institute of Standards and Technology/National Computer Security Center.

Balasubramaniyan, J., Garcia-Fernández, J., Isaco, D., Spaord, E. and Zamboni. D. (1998): "An architecture for intrusion detection using autonomous agents". Technical Report 98/05, June. COAST Laboratory - Purdue University,

Bardossy, A. and Duckstein, L. (1995): "Fuzzy Rule-Based Modeling with Applications to Geophysical, Biological and Engineering Systems". CRC Press.

Cios, K., Pedrycz, W. and Swiniarski, R.W. (1998): "Data Mining. Methods for Knowledge Discovery". Kluwer

Cordón, O; Del Jesús, M; Herrera, F. and López, E. (1997): "Selecting fuzzy rule-based classification systems with specific reasoning methods usin genetic algorithms". 7th International Fuzzy Systems Association World Congress, Praga, junio, Vol. 2, págs. 424-429.

Chan, P. and. Stolfo, S. (1993): "Toward parallel and distributed learning by meta-learning". AAAI Workshop in Knowledge Discovery in Databases, págs. 227-240.

Cho, S. and Kim, J.H. (1995): "Multiple Network Fusion Using Fuzzy Logic", IEEE Trans. Neural Networks, Vol. 6, págs. 497-501.

Denning, D. (1987): "An Intrusion-Detection Model". IEEE Transactions On Software Engineering, Vol. 13, nºo. 2, págs. 222-232.

Denning, D.E. (1987): "An intrusion-detection model". IEEE Transaction on Software Engineering. Vol. 13 (2), págs. 222-232.

Fayyad, U., Piatetsky-Shapiro, G. and Smyth, P. (1996): "The KDD process of extracting useful knowledge from volumes of data". Communications of the ACM, Vol. 39(11), págs.27--34.

Flockhart, I. and Radclife, N. (1996): "A Genetic-Based Approach to Data Mining". Proceedings of the II International Conference on Knowledge Discovery and Data Mining (KDD-96), págs. 299-302.

Frank, J. (1994): "Artificial Intelligence and Intrusion Detection: Current and Future Directions". Division of Computer Science University.of California at Davis.

Halme, L. and Bauer, K. (1995): "AINT misbehaving. A taxonomy of anti-intrusion techniques". Proceedings of the 18th National Information Systems Security Conference, págs.163-172, Baltimore. National Institute of Standards and Technology/National Computer Security Cente

Ilgun, K., Kemmerer, R. and Porras, P. (1995): "State transition analysis: A rule-based intrusion detection approach". IEEE Transactions on Software Engineering, Vol. 21 (3), págs.181—199.

Jackson, K., DuBois, D. and Stallings, C. (1991): "An expert system application for network intrusion detection". Proceedings of the 14th National Computer Security Conference, págs.215-225, Washington, October 1-4. National Institute of Standards and Technology /National Computer Security Center.

Kittler, J., Hatef, M., Duin, R. and Matas, J. (1998): "On Combining Classifiers". IEEE Trans. on Pattern Analysis and Machine Intelligence, nº 20, págs. 226-239.

Ko, C., Fink, G. and Levitt; K. (1994): "Automated detection of vulnerabilities in privileged programs by execution monitoring". Proceedings of the 10th Annual Computer Security Applications Conference, págs.134--144.

Kumar, S. (1995): "Classication and Detection of Computer Intrusions". PhD thesis, Purdue University, West Lafayette.

Kumar, S. and Spafford, E. (1995): "A software architecture to support misuse intrusion detection". Proceedings of the 18th National Information Security Conference, págs.194--204.

Lam, L. and Suen, C. (1997): "Application of Majority Voting to Pattern Recoginition and Analysis of its Behaviour and Performance". IEEE Transactions on Systms, Man, and Cybernetics, nº. 27, págs. 553-568.

Lane, T. and. Brodley, C. (1997): "Sequence matching and learning in anomaly detection for computer security". AAAI Workshop: AI Approaches to Fraud Detection and Risk Management, July, págs.43--49. AAAI Press.

Lane, T. and Brodley, C. (1998): "Temporal sequence learning and data reduction for anomaly detection". 5th ACM Conference on Computer & Communications Security, págs. 150-158, San Fransico (November).

Liu, H. and Motoda, H. (1998): "Feature Selection for Knowledge Discovery and Data mining". Kluwer.

López González, E. (1999): "La Construcción de Sistemas Inteligentes (Control Borroso) en Hoja Electrónica de Calculo para la Toma de Decisiones de Gestión en Ambiente de Incertidumbre". Curso Mestrado em Contabilidade e Auditoria, Universidade do Minho. Braga, Portugal.

López González, E. (2000): "Diseño de Sistemas Inteligentes de Gestión Empresarial". Material Docente para la asignatura SIGE. Dpto. de Dirección y Economía de la Empresa.

López González, E. (2001): "A Methodology for Building Fuzzy Expert Systems (FES) with Spreadsheet to Quality Function Deployment (QFD) of the Target Costing". Incluido en Gil Aluja, J. (ed.): "Handbook of Management under Uncertainty", Kluwer Academic Publishers, Dorchecht (En publicación).

Lunt, T. (1993): "Detecting Intruders in Computer Systems". Conference on Auditing and Computer Technology. Computer Science Laboratory SRI International.

Luo, J. (1999): "Integrating Fuzzy Logic with Data Mining Methods for Intrusion Detection", A Thesis Submitted to the Faculty of Mississippi State University in Partial Fulfillment of the Requirements for the Degree of Master of Science in Computer Science in the Department of Computer Science Mississippi State, Mississippi (August).

Me, L. (1998): "Gassata, a genetic algorithm as an alternative tool for security audit trails analysis". First international workshop on the Recent Advances in Intrusion Detection. (http://www.zurich.ibm.com/ dac/Prog RAID98/Table of content.html).

Me, L. and Alanoun V. (1996): "Detection d'intrusions dans un systeme informatique: methodes et outils". TSI. Vol.15, nº 4, págs. 29-45.

Mukherjee, B., Heberlein, L. and Levitt, K. (1994): "Network intrusion detection". IEEE Network, Vol. 8 (3), págs. 26-41.

Pedrycz, W. (1997): "Data Mining and Knowledge Discovery: a Fuzzy Set Perspective", R. Mesiar, B. Riecan (ed.), Fuzzy Structures: Current Trends. Tatra Mountains Mathematical Publications. Vol. 13, págs. 195-218.

Pedrycz, W. (1998): "Fuzzy Set Technology in Knowledge Discovery". Fuzzy Sets and Systems, nº. 98, págs. 279-290.

Porras, P. and Neumann, P. (1997): "EMERALD: Event monitoring enabling responses to anomalous live disturbances". Proceedings of the 20th National Information Systems Security Conference, págs. 353-365, Baltimore, National Institute of Standards and Technology/National Computer Security Center.

Sebring, M., Shellhouse, E., Hanna, M. and Whitehurst, R. (1988): "Expert systems in intrusion detection: A case study". Proceedings of the 11th National Computer Security Conference, págs. 74-81, Baltimore, National Institute of Standards and Technology/National Computer Security Center.

Sugeno, M. and Yasukawa, T. (1993): "A Fuzzy Logic-Based Approach to Linguistic Modeling". IEEE Transactions on Fuzzy Systems. Vol. 1, págs. 7-31.

Warrender, C., Forrest, S. and Perlmutter, B. (1999): "Detecting intrusions using system calls: Alternative data models". IEEE Symposium on Security and Privacy, Berkeley, págs.133-145.

White, G. and Pooch, V. (1996): "Cooperating security managers: Distributed intrusion detection systems". Computers & Security, Vol. 15 (5), págs. 441-450.

Wijsen, J. and Meersman, R. (1998): "On the Complexity of Mining Quantitative Association Rules". Data Mining and Knowledge Discovery. Vol. 2, págs. 263-281.

Zadeh, L.A. (1996): "Fuzzy Logic=Computing with Words", IEEE Transactions on Fuzzy Systems. Vol. 4, págs. 103-11.