

OCS-WAF:

un Web Application Firewall basado en anomalías con clasificadores One-Class SVM

Nico Epp Ralf Funk



Universidad Nacional de Asunción
Facultad Politécnica
Ingeniería en Informática

Defensa Pública de Trabajo Final de Grado

San Lorenzo, Paraguay
20 de Noviembre 2017

Contenido

Introducción

- Motivación de la investigación
- Objetivos para la investigación

Implementación de OCS-WAF

- Arquitectura general
- Fase de entrenamiento
- Fase de detección

Pruebas y resultados

- Conjuntos de datos de prueba
- Análisis de la eficacia de detección
- Análisis del tiempo de respuesta de las aplicaciones
- Comparación con otros trabajos

Conclusiones

- Resumen de la investigación
- Logro de los objetivos
- Trabajos futuros
- Publicaciones

Contenido

Introducción

Motivación de la investigación

Objetivos para la investigación

Implementación de OCS-WAF

Arquitectura general

Fase de entrenamiento

Fase de detección

Pruebas y resultados

Conjuntos de datos de prueba

Análisis de la eficacia de detección

Análisis del tiempo de respuesta de las aplicaciones

Comparación con otros trabajos

Conclusiones

Resumen de la investigación

Logro de los objetivos

Trabajos futuros

Publicaciones

Aplicaciones web



- ▶ Propiedades:
 - ▶ Ubicuidad
 - ▶ Acceso anónimo
 - ▶ Código escrito por no expertos
 - ▶ Vulnerabilidades presentes

Vulnerabilidades en aplicaciones web¹



- Severidad alta: *Cross-Site Scripting* (XSS), Inyección SQL, entre otros



- Severidad media: CSRF, Denegación de servicio (DoS), entre otros

¹ Acunetix Web Application Vulnerability Report 2016 (*Datos entre abril 2015 y marzo 2016*)

Problema:

- ▶ Las aplicaciones web contienen vulnerabilidades y están expuestas a un elevado riesgo de ataques.

Problema:

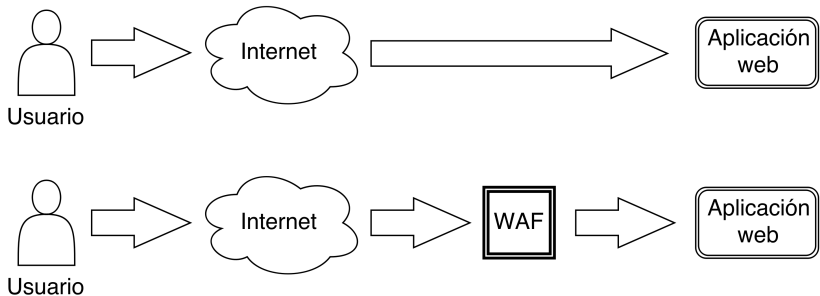
- ▶ Las aplicaciones web contienen vulnerabilidades y están expuestas a un elevado riesgo de ataques.

Nuestra investigación:

- ▶ Mitigación del riesgo de ataques

Nuestra investigación

- ▶ Web Application Firewall (WAF)
 - ▶ Finalidad: mitigar riesgo de ataques



Sistemas de Detección de Intrusiones (IDS)

Sistemas de Detección de Intrusiones (IDS)

- ▶ Modo de respuesta:
 - ▶ Pasivo - detección (IDS)
 - ▶ Activo - prevención (IPS)

Sistemas de Detección de Intrusiones (IDS)

- ▶ Modo de respuesta:
 - ▶ Pasivo - detección (IDS)
 - ▶ Activo - prevención (IPS)
- ▶ Fuente de datos:
 - ▶ *Host-based systems* (HIDS)
 - ▶ *Network-based systems* (NIDS)
 - ▶ Mensajes HTTP → WAF

Sistemas de Detección de Intrusiones (IDS)

- ▶ Modo de respuesta:
 - ▶ Pasivo - detección (IDS)
 - ▶ Activo - prevención (IPS)
- ▶ Fuente de datos:
 - ▶ *Host-based systems* (HIDS)
 - ▶ *Network-based systems* (NIDS)
 - ▶ Mensajes HTTP → WAF
- ▶ Método de detección:
 - ▶ Por firmas de ataques
 - ▶ Por anomalías

WAF con detección de anomalías

- ▶ Dos fases:
 - ▶ Fase de entrenamiento: construcción de modelos que describen mensajes HTTP normales
 - ▶ Fase de detección: comparación de nuevos mensajes con modelos construidos

WAF con detección de anomalías

- ▶ Dos fases:
 - ▶ Fase de entrenamiento: construcción de modelos que describen mensajes HTTP normales
 - ▶ Fase de detección: comparación de nuevos mensajes con modelos construidos
- ▶ Ventaja: detección de ataques nuevos sin re-entrenar
- ▶ Desventaja: dificultad de construir modelos útiles para la detección

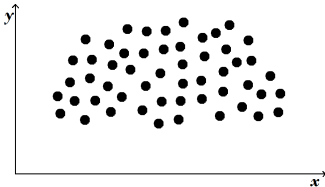
WAF con detección de anomalías

- ▶ Dos fases:
 - ▶ Fase de entrenamiento: construcción de modelos que describen mensajes HTTP normales
 - ▶ Fase de detección: comparación de nuevos mensajes con modelos construidos
- ▶ Ventaja: detección de ataques nuevos sin re-entrenar
- ▶ Desventaja: dificultad de construir modelos útiles para la detección
- ▶ Estrategías de abordaje:
 - ▶ Métodos estadísticos (definición de *threshold*)
 - ▶ Problema de clasificación utilizando herramientas de *Machine Learning*

Problemas de clasificación y *Machine Learning*

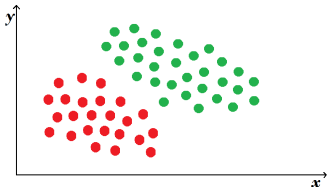
Problemas de clasificación y *Machine Learning*

- Clasificación no supervisada

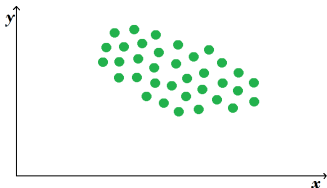


Problemas de clasificación y *Machine Learning*

- ▶ Clasificación no supervisada
- ▶ Clasificación supervisada



Problemas de clasificación y *Machine Learning*



- ▶ Clasificación no supervisada
- ▶ Clasificación supervisada
- ▶ Clasificación semi-supervisada
 - ▶ Caso especial: entrenamiento con muestras de una sola clase
OCC: *One-Class Classification*
 - ▶ Una herramienta posible:
One-Class SVM

Objetivo general

Objetivo general

- ▶ Detectar mensajes HTTP anómalos entre las aplicaciones web y sus usuarios con el fin de mitigar los riesgos de ataques contra dichas aplicaciones, utilizando un WAF basado en clasificadores One-Class SVM.

Objetivos específicos

Objetivos específicos

1. Diseñar procesos de extracción de características (*features*) específicamente para mensajes HTTP, basado en aportes de otros investigadores de la literatura.

Objetivos específicos

1. Diseñar procesos de extracción de características (*features*) específicamente para mensajes HTTP, basado en aportes de otros investigadores de la literatura.
2. Implementar un WAF basado en anomalías, utilizando los procesos de extracción de *features* diseñados junto con clasificadores One-Class SVM.

Objetivos específicos

1. Diseñar procesos de extracción de características (*features*) específicamente para mensajes HTTP, basado en aportes de otros investigadores de la literatura.
2. Implementar un WAF basado en anomalías, utilizando los procesos de extracción de *features* diseñados junto con clasificadores One-Class SVM.
3. Evaluar la eficacia del WAF implementado en cuanto a la detección de mensajes HTTP anómalos.

Objetivos específicos

1. Diseñar procesos de extracción de características (*features*) específicamente para mensajes HTTP, basado en aportes de otros investigadores de la literatura.
2. Implementar un WAF basado en anomalías, utilizando los procesos de extracción de *features* diseñados junto con clasificadores One-Class SVM.
3. Evaluar la eficacia del WAF implementado en cuanto a la detección de mensajes HTTP anómalos.
4. Analizar la viabilidad de utilizar el WAF implementado para detección de ataques en tiempo real.

Contenido

Introducción

Motivación de la investigación

Objetivos para la investigación

Implementación de OCS-WAF

Arquitectura general

Fase de entrenamiento

Fase de detección

Pruebas y resultados

Conjuntos de datos de prueba

Análisis de la eficacia de detección

Análisis del tiempo de respuesta de las aplicaciones

Comparación con otros trabajos

Conclusiones

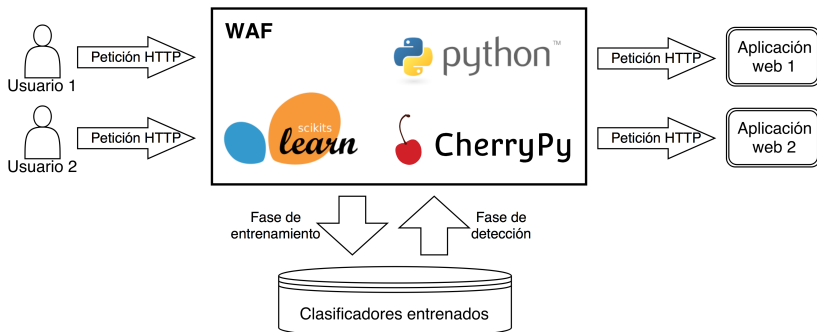
Resumen de la investigación

Logro de los objetivos

Trabajos futuros

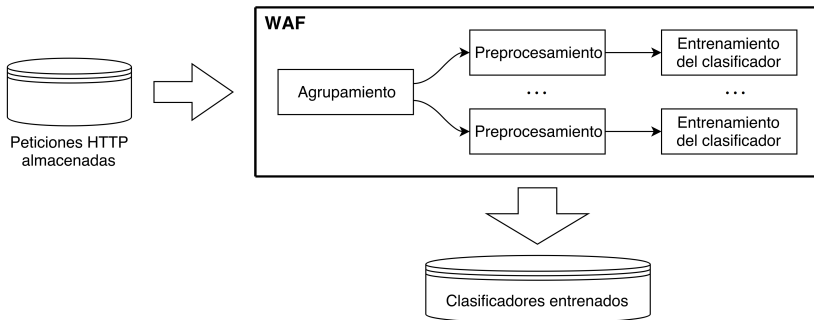
Publicaciones

Detector de anomalías OCS-WAF



<https://github.com/nico-ralf-ii-fpuna/tfg>

Fase de entrenamiento de OCS-WAF



Estructura de peticiones HTTP

Método HTTP URL *query string*

GET http://localhost:8080/ver.jsp?email=juan%40gmail.com&full=y HTTP/1.1
Accept-Charset: utf-8, utf-8;q=0.5
Accept-Language: en

p₁ v₁ p₂ v₂

Método HTTP URL

POST http://localhost:8080/tienda1/publico/pagar.jsp HTTP/1.1
Accept-Charset: utf-8, utf-8;q=0.5
Accept-Language: en

cantidad=5&precio=330

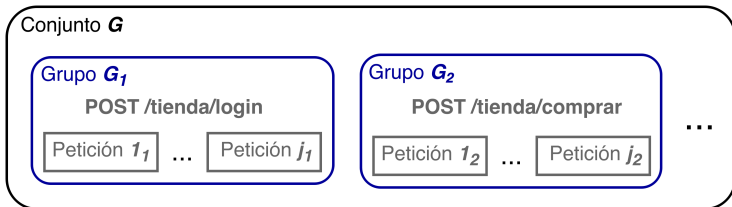
p₁ v₁ p₂ v₂

} cabeceras
}
} cuerpo

1. Paso de agrupamiento

1. Paso de agrupamiento

- ▶ Agrupación por método HTTP y URL



- ▶ Similitud entre peticiones de un mismo grupo G_i
 - ▶ $\forall i = 1, 2, \dots, |G|$
- ▶ Descripciones más precisas del comportamiento normal dentro de cada grupo G_i

2. Paso de preprocesamiento

2. Paso de preprocesamiento

- ▶ Representación de características de las peticiones mediante vectores numéricos de *features*
 - ▶ Petición HTTP $\rightarrow \vec{f} \in \mathbb{R}^n$
 - ▶ dimensiones distintas para cada grupo G_i

2. Paso de preprocesamiento

- ▶ Representación de características de las peticiones mediante vectores numéricos de *features*
 - ▶ Petición HTTP $\rightarrow \vec{f} \in \mathbb{R}^n$
 - ▶ dimensiones distintas para cada grupo G_i
- ▶ Características analizadas por OCS-WAF:
 - ▶ Distribución de caracteres
 - ▶ Entropía
 - ▶ Cantidad de caracteres

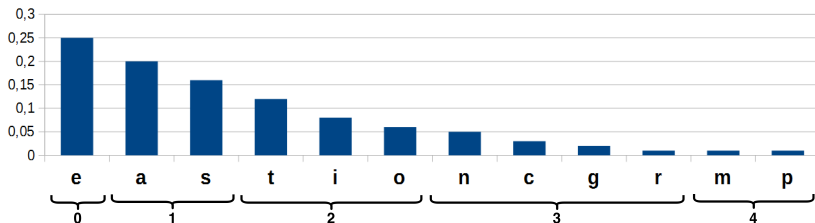
2. Paso de preprocesamiento

- ▶ Distribución de caracteres
 - ▶ Aporte de Kruegel y Vigna²
 - ▶ Distribución de frecuencias relativas de caracteres
 - ▶ Suma de frecuencias relativas en cinco intervalos

²Kruegel and Vigna (2003) *Anomaly detection of web-based attacks*.

2. Paso de preprocesamiento

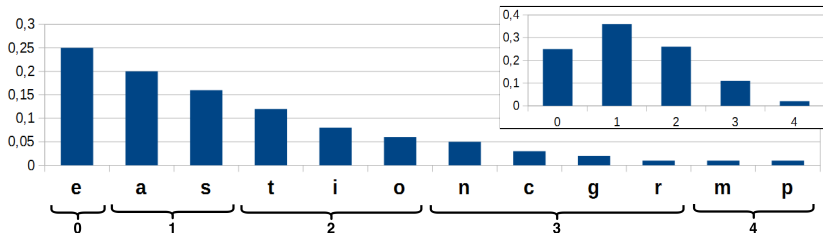
- Distribución de caracteres
 - Aporte de Kruegel y Vigna²
 - Distribución de frecuencias relativas de caracteres
 - Suma de frecuencias relativas en cinco intervalos



²Kruegel and Vigna (2003) *Anomaly detection of web-based attacks*.

2. Paso de preprocesamiento

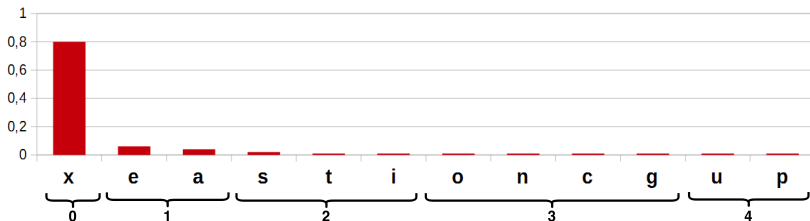
- ▶ Distribución de caracteres
 - ▶ Aporte de Kruegel y Vigna²
 - ▶ Distribución de frecuencias relativas de caracteres
 - ▶ Suma de frecuencias relativas en cinco intervalos



²Kruegel and Vigna (2003) *Anomaly detection of web-based attacks*.

2. Paso de preprocesamiento

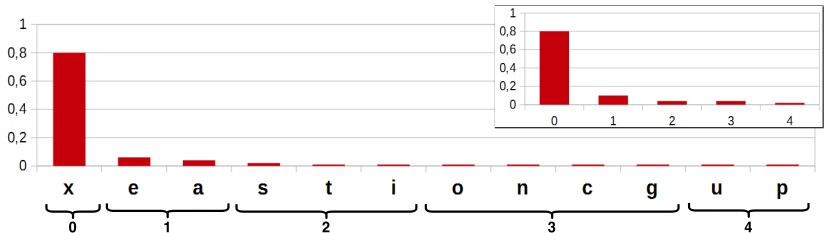
- Distribución de caracteres
 - Aporte de Kruegel y Vigna²
 - Distribución de frecuencias relativas de caracteres
 - Suma de frecuencias relativas en cinco intervalos



²Kruegel and Vigna (2003) *Anomaly detection of web-based attacks*.

2. Paso de preprocesamiento

- Distribución de caracteres
 - Aporte de Kruegel y Vigna²
 - Distribución de frecuencias relativas de caracteres
 - Suma de frecuencias relativas en cinco intervalos



²Kruegel and Vigna (2003) *Anomaly detection of web-based attacks*.

2. Paso de preprocesamiento

- ▶ Entropía (Teoría de la información)
 - ▶ Relación entre longitud del valor y cantidad de caracteres distintos
 - ▶ Aporte de Nguyen et. al.³
 - ▶ Fórmula propuesta por Claude Shannon⁴

$$H(x) = - \sum_{i=1}^{|c|} \left(\frac{c_i}{|x|} \times \log_2 \frac{c_i}{|x|} \right)$$

Ejemplo

$x = \text{aabbcc}$

$|x| = 5$

$|c| = 3$

$c_1 = 2 \rightarrow a$

$c_2 = 2 \rightarrow b$

$c_3 = 1 \rightarrow c$

³Nguyen et. al. (2011) *Application of the generic feature selection measure in detection of web attacks*.

⁴Shannon (1948) *A Mathematical Theory of Communication*.

2. Paso de preprocesamiento

- ▶ Entropía (Teoría de la información)
 - ▶ Relación entre longitud del valor y cantidad de caracteres distintos
 - ▶ Aporte de Nguyen et. al.³
 - ▶ Fórmula propuesta por Claude Shannon⁴

Ejemplo: petición normal

▶ $H(x) = 5,092$

Ejemplo: petición con *buffer overflow*

▶ $H(x) = 0.901$

³Nguyen et. al. (2011) *Application of the generic feature selection measure in detection of web attacks.*

⁴Shannon (1948) *A Mathematical Theory of Communication.*

2. Paso de preprocesamiento

- ▶ Cantidad de caracteres
 - ▶ Aporte de Kruegel y Vigna⁵, y Nguyen et. al.⁶
 - ▶ Conjuntos de caracteres
 1. Todos
 2. Dígitos
 3. Letras
 4. Otros caracteres

⁵Kruegel and Vigna (2003) *Anomaly detection of web-based attacks*.

⁶Nguyen et. al. (2011) *Application of the generic feature selection measure in detection of web attacks*.

2. Paso de preprocesamiento

- ▶ Cantidad de caracteres
 - ▶ Aporte de Kruegel y Vigna⁵, y Nguyen et. al.⁶
 - ▶ Conjuntos de caracteres
 1. Todos
 2. Dígitos
 3. Letras
 4. Otros caracteres

Ejemplo: petición normal

Todos	=	100
Dígitos	=	9
Letras	=	74
Otros	=	17

Ejemplo: petición con *code injection*

Todos	=	132
Dígitos	=	21
Letras	=	78
Otros	=	33

⁵Kruegel and Vigna (2003) *Anomaly detection of web-based attacks*.

⁶Nguyen et. al. (2011) *Application of the generic feature selection measure in detection of web attacks*.

2. Paso de preprocesamiento

- 10 *features* extraídos

<i>Features</i>	Tipo de dato	Rango de valores
Dist. de caract. - intervalo 0	núm. reales	$[0, 1]$
Dist. de caract. - intervalo 1	núm. reales	$[0, 1]$
Dist. de caract. - intervalo 2	núm. reales	$[0, 1]$
Dist. de caract. - intervalo 3	núm. reales	$[0, 1]$
Dist. de caract. - intervalo 4	núm. reales	$[0, 1]$
Entropía	núm. reales	$[0, \infty)$
Longitud o cantidad total	núm. enteros	$[0, \infty)$
Cantidad de dígitos	núm. enteros	$[0, \infty)$
Cantidad de letras	núm. enteros	$[0, \infty)$
Cantidad de otros caracteres	núm. enteros	$[0, \infty)$

2. Paso de preprocesamiento

- ▶ Análisis de valores de parámetros
 - ▶ Obtención de parámetros presentes en peticiones de G_i
 - ▶ $Q_i = [\text{"email"} , \text{"full"}]$
 - ▶ $B_i = []$
 - ▶ Extracción de *features* de cada valor de los parámetros

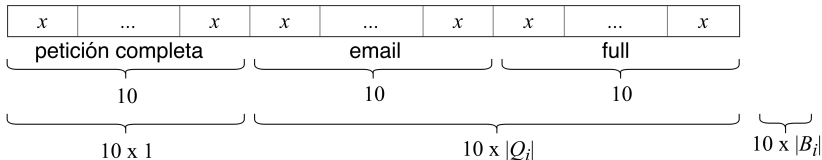
Diagram illustrating the extraction of parameters from HTTP requests:

- p_1 (parameter name: email)
- v_1 (parameter value: juan%40gmail.com)
- p_2 (parameter name: full)
- v_2 (parameter value: y)

GET http://localhost:8080/ver.jsp?email=juan%40gmail.com&full=y HTTP/1.1
GET http://localhost:8080/ver.jsp?email=pedro%40gmail.com HTTP/1.1
GET http://localhost:8080/ver.jsp?email=jose%40gmail.com HTTP/1.1

2. Paso de preprocesamiento

- Composición del vector de *features*
 - Dimensiones según los parámetros en cada grupo G_i
 - $n_i = 10 \times (1 + |Q_i| + |B_i|)$



GET http://localhost:8080/ver.jsp?email=juan%40gmail.com&full=y HTTP/1.1
GET http://localhost:8080/ver.jsp?email=pedro%40gmail.com HTTP/1.1
GET http://localhost:8080/ver.jsp?email=jose%40gmail.com HTTP/1.1

Parameters p_1 , v_1 , p_2 , and v_2 are indicated above the corresponding parts of the URLs.

2. Paso de preprocesamiento

- Composición de la matriz de vectores de *features*
 - Filas: vectores de peticiones del grupo G_i
 - Columnas: *features* del grupo G_i

$$M_i = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,n_i} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,n_i} \\ \vdots & \vdots & \ddots & \vdots \\ x_{|G_i|,1} & x_{|G_i|,2} & \cdots & x_{|G_i|,n_i} \end{bmatrix}$$

2. Paso de preprocesamiento

- ▶ Escalamiento de *features* (normalización)
 - ▶ Problema:
 - ▶ Distintas importancias de *features* debido a rangos diferentes

⁷Rieck (2009) Machine Learning for Application-Layer Intrusion Detection

2. Paso de preprocesamiento

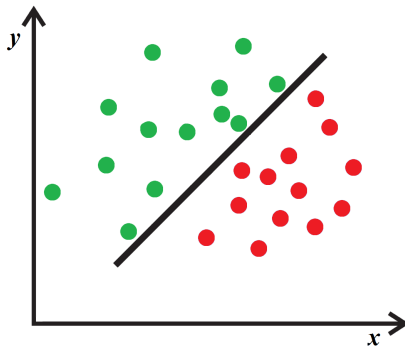
- ▶ Escalamiento de *features* (normalización)
 - ▶ Problema:
 - ▶ Distintas importancias de *features* debido a rangos diferentes
 - ▶ Finalidad del escalamiento estándar⁷:
 - ▶ Promedio cercano a 0 y una varianza cercana a 1 en cada *feature* (cada columna de M_i)

$$x_{\text{nuevo}} = \frac{x_{\text{actual}} - \mu_{\text{de la columna}}}{\sigma_{\text{de la columna}}}$$

⁷Rieck (2009) Machine Learning for Application-Layer Intrusion Detection

Support Vector Machine (SVM)

- ▶ Clasificadores binarios de aprendizaje supervisado
- ▶ Separación de clases mediante un hiperplano

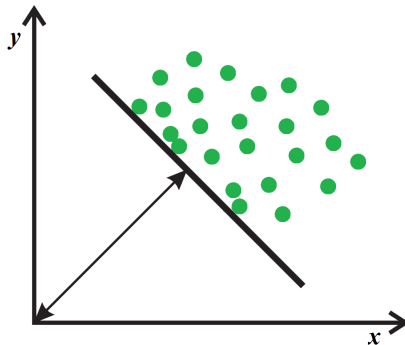


Support Vector Machine (SVM)

- ▶ Clasificadores binarios de aprendizaje supervisado
- ▶ Separación de clases mediante un hiperplano

One-Class SVM

- ▶ Versión modificada para problemas OCC
- ▶ Separación de única clase del origen

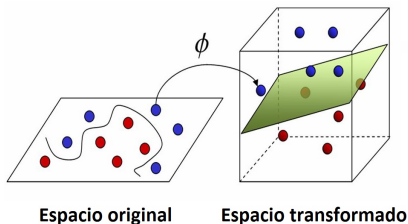


Support Vector Machine (SVM)

- ▶ Clasificadores binarios de aprendizaje supervisado
- ▶ Separación de clases mediante un hiperplano

One-Class SVM

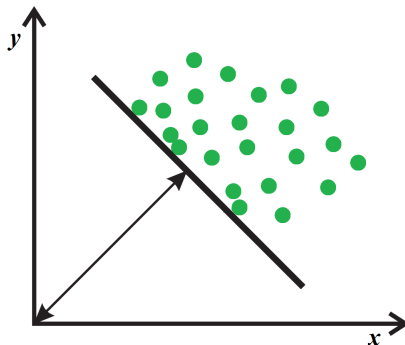
- ▶ Versión modificada para problemas OCC
- ▶ Separación de única clase del origen
- ▶ Transformación a otros espacios
 - ▶ *Radial Basis Function* (RBF) kernel



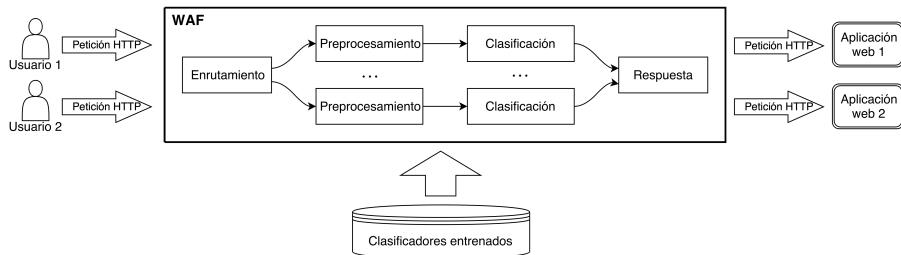
3. Paso de entrenamiento de clasificadores

3. Paso de entrenamiento de clasificadores

- ▶ Un One-Class SVM por cada grupo G_i
- ▶ Entrenamiento con los vectores de *features* construidos
 - ▶ Matrices M_i
- ▶ Gran cantidad de *features*
- ▶ Almacenamiento persistente de clasificadores entrenados



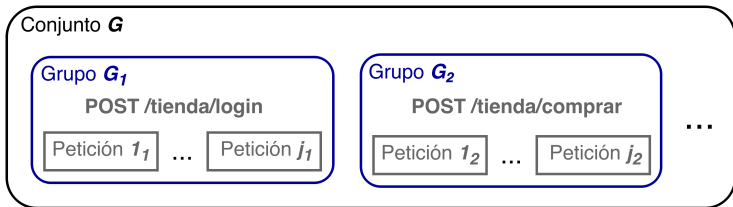
Fase de detección



1. Paso de enrutamiento

1. Paso de enrutamiento

- Identificación del grupo G_i de las nuevas peticiones según su método HTTP y URL



- Delegación al proceso de extracción de *features* del grupo y al clasificador correspondiente

2. Paso de preprocesamiento

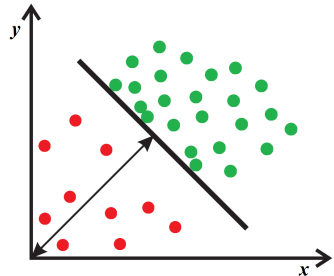
2. Paso de preprocesamiento

- ▶ Construcción de vectores de *features* para nuevas peticiones
- ▶ Parámetro de Q_i o B_i que no aparece en una nueva petición
 - ▶ Componentes correspondientes llevan 0
- ▶ Nuevos parámetros que no están en Q_i o B_i
 - ▶ No agregan nuevos *features* al vector
 - ▶ Están contenidos en la petición completa
- ▶ Vectores nuevos con misma dimensión n_i del grupo

3. Paso de clasificación

3. Paso de clasificación

- ▶ Uso del hiperplano obtenido para el grupo G_i
- ▶ Análisis de la posición de nuevas peticiones:
 - ▶ Lado opuesto al origen:
 - ▶ Petición normal
 - ▶ Mismo lado que el origen:
 - ▶ Petición anómala o ataque



4. Paso de respuesta

4. Paso de respuesta

- ▶ Distintas acciones como respuesta al resultado de clasificación
- ▶ Peticiones normales:
 - ▶ Reenvío a las aplicaciones web destino
- ▶ Peticiones anómalas:
 - ▶ Registro en un *log*
 - ▶ Opcionalmente: bloqueo de la petición

Contenido

Introducción

Motivación de la investigación

Objetivos para la investigación

Implementación de OCS-WAF

Arquitectura general

Fase de entrenamiento

Fase de detección

Pruebas y resultados

Conjuntos de datos de prueba

Análisis de la eficacia de detección

Análisis del tiempo de respuesta de las aplicaciones

Comparación con otros trabajos

Conclusiones

Resumen de la investigación

Logro de los objetivos

Trabajos futuros

Publicaciones

Conjuntos de datos utilizados

- ▶ Conjuntos utilizados:
 - ▶ CSIC 2010⁸
 - ▶ CSIC TORPEDA 2012⁹
- ▶ Peticiones HTTP simuladas a una aplicación de comercio electrónico
- ▶ Distintos tipos de ataques
 - ▶ Inyección SQL, *buffer overflow*, *cross-site scripting* (XSS), entre otros
- ▶ Datos utilizados:
 - ▶ 18 grupos de peticiones según método HTTP y URL
 - ▶ 40 130 peticiones normales y 42 444 anomalías

⁸<http://www.isi.csic.es/dataset/>

⁹<http://www.tic.itefi.csic.es/torpeda>

Pruebas de eficacia de detección

		Clasificación real	
		N Negatives (peticiones normales)	P Positives (peticiones anómalas)
Detección obtenida	Negatives (peticiones detectadas como normales)	TN True Negatives (peticiones normales detectadas correctamente como normales)	FN False Negatives (peticiones anómalas detectadas incorrectamente como normales)
	Positives (peticiones detectadas como anómalas)	FP False Positives (peticiones normales detectadas incorrectamente como anómalas)	TP True Positives (peticiones anómalas detectadas correctamente como anómalas)

$$TPR = \frac{TP}{P} \quad , \quad FPR = \frac{FP}{N} \quad , \quad F_1\text{-score} = \frac{2TP}{2TP + FP + FN}$$

- $F_1\text{-score}$ es más robusto frente a datos no balanceados

Pruebas de eficacia de detección

- ▶ Mejoras obtenidas por el análisis de valores de parámetros
 - ▶ Promedio de los 18 grupos
 - ▶ 3 iteraciones con distintos subconjuntos de entrenamiento
 - ▶ Entrenamiento:
 - ▶ 1 500 peticiones normales en cada grupo
 - ▶ $\approx 75\%$ de los datos normales
 - ▶ Detección:
 - ▶ $\approx 2\,000$ peticiones normales en cada grupo
 - ▶ $\approx 1\,300$ peticiones anómalas en cada grupo

	TPR	FPR	F_1 -score
Solo petición completa	0.77 ± 0.28	0.11 ± 0.22	0.79 ± 0.23
Con análisis de parámetros	0.93 ± 0.11	0.03 ± 0.03	0.95 ± 0.08

Pruebas de tiempo de respuesta

- ▶ Medición del impacto de OCS-WAF sobre las aplicaciones protegidas
 - ▶ Promedio de los 18 grupos
 - ▶ 100 peticiones de cada grupo

	Tiempo de respuesta promedio (en milisegundos)
Directo a la aplicación web	$4,0 \pm 0,6$
A través de OCS-WAF	$8,7 \pm 1,3$

Trabajos relacionados de otros autores

- Utilización del mismo conjunto de datos CSIC 2010

Propuesta	Observaciones	TPR	FPR	F_1 -score
OCS-WAF	One-Class SVM	0,93	0,03	0,95
ModSecurity ^{10 11}	Detección por firmas	0,56	0,00	0,71
HTTP-WS-AD ¹²	Métodos estadísticos	0,99	0,02	0,99
Torrano-Giménez ¹³	Árboles de decisión	0,95	0,05	-
OC-WAD ¹⁴	One-Class SVM	0,96	0,03	-

¹⁰<https://www.modsecurity.org/>

¹¹Giménez (2015) HTTP-WS-AD: Detector de anomalías orientada a aplicaciones web y web services.

¹²Giménez (2015) HTTP-WS-AD: Detector de anomalías orientada a aplicaciones web y web services.

¹³Torrano-Giménez (2015) Study of stochastic and machine learning techniques for anomaly-based detection.

¹⁴Parhizkar y Abadi (2015) OC-WAD: A one-class classifier ensemble approach for anomaly detection.

Contenido

Introducción

Motivación de la investigación

Objetivos para la investigación

Implementación de OCS-WAF

Arquitectura general

Fase de entrenamiento

Fase de detección

Pruebas y resultados

Conjuntos de datos de prueba

Análisis de la eficacia de detección

Análisis del tiempo de respuesta de las aplicaciones

Comparación con otros trabajos

Conclusiones

Resumen de la investigación

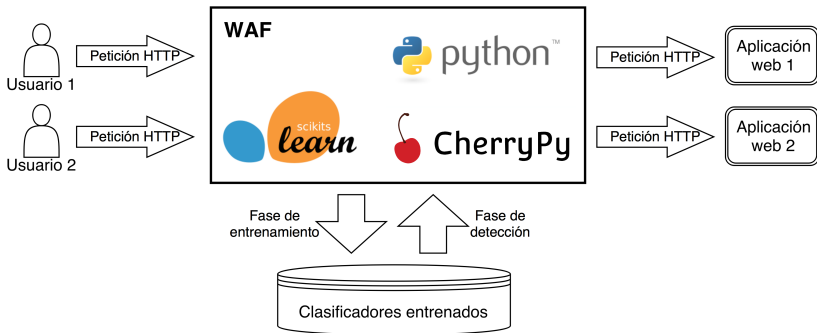
Logro de los objetivos

Trabajos futuros

Publicaciones

OCS-WAF

- ▶ Implementación de un WAF para protección de múltiples aplicaciones web
- ▶ Detección de anomalías en mensajes HTTP mediante clasificadores One-Class SVM



Aporte de nuestra investigación

- ▶ Combinación de procesos de extracción de *features* para peticiones HTTP con clasificadores One-Class SVM
 - ▶ Análisis de valores de parámetros
 - ▶ Descripciones más precisas de los valores normales

Aporte de nuestra investigación

- ▶ Combinación de procesos de extracción de *features* para peticiones HTTP con clasificadores One-Class SVM
 - ▶ Análisis de valores de parámetros
 - ▶ Descripciones más precisas de los valores normales
- ▶ Código fuente de OCS-WAF
 - ▶ Incentivar la reproducción de nuestros resultados
 - ▶ Facilitar la extensión de nuestro trabajo

Objetivos específicos

Objetivos específicos

1. Diseñar procesos de extracción de características (*features*) específicamente para mensajes HTTP, basado en aportes de otros investigadores de la literatura.
 - ▶ Diseño de nuevos procesos de extracción de *features* para mensajes HTTP

Objetivos específicos

1. Diseñar procesos de extracción de características (*features*) específicamente para mensajes HTTP, basado en aportes de otros investigadores de la literatura.
 - ▶ Diseño de nuevos procesos de extracción de *features* para mensajes HTTP
2. Implementar un WAF basado en anomalías, utilizando los procesos de extracción de *features* diseñados junto con clasificadores One-Class SVM.
 - ▶ Implementación de OCS-WAF

Objetivos específicos

3. Evaluar la eficacia del WAF implementado en cuanto a la detección de mensajes HTTP anómalos.
 - ▶ Pruebas de eficacia de detección con un TPR de 0,93, FPR de 0,03 y F_1 -score de 0,95

Objetivos específicos

3. Evaluar la eficacia del WAF implementado en cuanto a la detección de mensajes HTTP anómalos.
 - ▶ Pruebas de eficacia de detección con un TPR de 0,93, FPR de 0,03 y F_1 -score de 0,95
4. Analizar la viabilidad de utilizar el WAF implementado para detección de ataques en tiempo real.
 - ▶ Realización de pruebas de impacto de OCS-WAF sobre el tiempo de respuesta de las aplicaciones protegidas

Objetivo general

Objetivo general

- ▶ Detectar mensajes HTTP anómalos entre las aplicaciones web y sus usuarios con el fin de mitigar los riesgos de ataques contra dichas aplicaciones, utilizando un WAF basado en clasificadores One-Class SVM.
 - ▶ Detección de mensajes HTTP anómalos con OCS-WAF

Ideas para trabajos futuros

Ideas para trabajos futuros

- ▶ Realizar pruebas con otros conjuntos de datos.

Ideas para trabajos futuros

- ▶ Realizar pruebas con otros conjuntos de datos.
- ▶ Explorar otras características de los mensajes HTTP.

Ideas para trabajos futuros

- ▶ Realizar pruebas con otros conjuntos de datos.
- ▶ Explorar otras características de los mensajes HTTP.
- ▶ Explorar otras herramientas (en vez del One-Class SVM) para la detección.

Ideas para trabajos futuros

- ▶ Realizar pruebas con otros conjuntos de datos.
- ▶ Explorar otras características de los mensajes HTTP.
- ▶ Explorar otras herramientas (en vez del One-Class SVM) para la detección.
- ▶ Extender para incluir cuerpos de otros formatos, por ejemplo datos binario, JSON, XML, entre otros.

Ideas para trabajos futuros

- ▶ Realizar pruebas con otros conjuntos de datos.
- ▶ Explorar otras características de los mensajes HTTP.
- ▶ Explorar otras herramientas (en vez del One-Class SVM) para la detección.
- ▶ Extender para incluir cuerpos de otros formatos, por ejemplo datos binario, JSON, XML, entre otros.
- ▶ Extender para incluir mensajes HTTP/2.

Ideas para trabajos futuros

- ▶ Realizar pruebas con otros conjuntos de datos.
- ▶ Explorar otras características de los mensajes HTTP.
- ▶ Explorar otras herramientas (en vez del One-Class SVM) para la detección.
- ▶ Extender para incluir cuerpos de otros formatos, por ejemplo datos binario, JSON, XML, entre otros.
- ▶ Extender para incluir mensajes HTTP/2.
- ▶ Explorar la posibilidad de paralelizar el proceso de entrenamiento en OCS-WAF.

Publicación de nuestro trabajo

- ▶ **Título:** Anomaly-based Web Application Firewall using HTTP-specific features and One-Class SVM

WRSeg 2017

- ▶ Workshop Regional de Segurança da Informação e Sistemas Computacionais
- ▶ Santa María, Brasil
- ▶ 25 de Setiembre 2017



Revista REABTIC

- ▶ Revista Eletrônica Argentina-Brasil de Tecnologias da Informação e da Comunicação
- ▶ Enviado y en revisión



Gracias por su atención!



`https://github.com/nico-ralf-ii-fpuna/tfg`

Anexo: One-Class SVM

- Hiperplano del clasificador para el grupo G_i

$$\vec{w}_i \cdot \phi(\vec{x}) - \rho_i = \sum_{j=1}^{|G_i|} \left(a_{ij} K_i(\vec{f}_{ij}, \vec{x}) \right) - \rho_i = 0 \quad (1)$$

- Kernel Radial Basis Function (RBF)

$$K(\vec{x}_1, \vec{x}_2) = \phi(\vec{x}_1) \cdot \phi(\vec{x}_2) = \exp(-\gamma \|\vec{x}_1 - \vec{x}_2\|^2) \quad (2)$$

Anexo: One-Class SVM

- Entrenamiento: problema de minimización para el grupo G_i

$$\min_{\vec{w}_i, \rho_i, \xi_i} \frac{1}{2} \|\vec{w}_i\|^2 - \rho_i + \frac{1}{\nu_i |G_i|} \sum_{j=1}^{|G_i|} \xi_{ij} \quad (3)$$

$$\vec{w}_i \cdot \phi(\vec{f}_{ij}) \geq \rho_i - \xi_{ij}, \quad \xi_{ij} \geq 0, \quad \forall j = 1, 2, \dots, |G_i| \quad (4)$$

- Entrenamiento: formulación dual de la optimización

$$\min_{a_i} \frac{1}{2} a_i S_i a_i^T \quad (5)$$

$$\sum_{j=1}^{|G_i|} a_{ij} = 1, \quad 0 \leq a_{ij} \leq \frac{1}{\nu_i |G_i|} \quad \forall j = 1, 2, \dots, |G_i| \quad (6)$$

Anexo: One-Class SVM

- Detección: función de decisión del grupo G_i

$$g_i(\vec{x}) = \begin{cases} \vec{w}_i \cdot \phi(\vec{x}) - \rho_i \geq 0 & +1 \\ \text{caso contrario} & -1 \end{cases} \quad (7)$$

$$g_i(\vec{x}) = \begin{cases} \sum_{j=1}^{|G_i|} \left(a_{ij} K_i(\vec{f}_{ij}, \vec{x}) \right) - \rho_i \geq 0 & +1 \\ \text{caso contrario} & -1 \end{cases} \quad (8)$$

Anexo: Datos utilizados para las pruebas

► Información sobre los 18 grupos de peticiones

ID	Conjunto de datos CSIC	Método HTTP y URL	Cant. parám.	Peticiones normales	Peticiones anómalas
c00	2010	GET /tienda1/miembros/editar.jsp	13	2 000	1 362
c01	2010	POST /tienda1/miembros/editar.jsp	13	2 000	1 362
c02	2010	GET /tienda1/publico/anadir.jsp	5	2 000	1 380
c03	2010	POST /tienda1/publico/anadir.jsp	5	2 000	1 380
c04	2010	GET /tienda1/publico/autenticar.jsp	5	2 000	1 361
c05	2010	POST /tienda1/publico/autenticar.jsp	5	2 000	1 361
c06	2010	GET /tienda1/publico/caracteristicas.jsp	1	2 000	954
c07	2010	POST /tienda1/publico/caracteristicas.jsp	1	2 000	954
c08	2010	GET /tienda1/publico/entrar.jsp	1	2 000	897
c09	2010	POST /tienda1/publico/entrar.jsp	1	2 000	897
c10	2010	GET /tienda1/publico/pagar.jsp	3	2 000	1 343
c11	2010	POST /tienda1/publico/pagar.jsp	3	2 000	1 343
c12	2010	GET /tienda1/publico/registro.jsp	13	2 000	1 364
c13	2010	POST /tienda1/publico/registro.jsp	13	2 000	1 364
c14	2010	GET /tienda1/publico/vaciar.jsp	1	2 000	919
c15	2010	POST /tienda1/publico/vaciar.jsp	1	2 000	919
t00	2012	POST /tienda1/miembros/editar.jsp	12	5 608	10 121
t01	2012	POST /tienda1/publico/registro.jsp	12	2 522	13 163
	Total			40 130	42 444

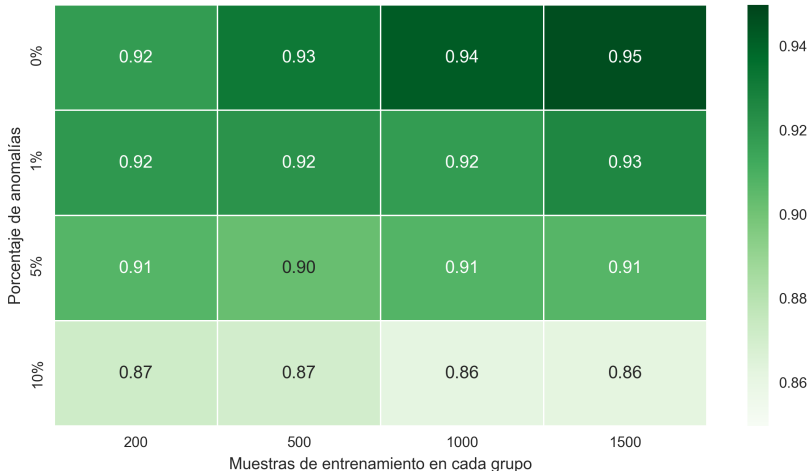
Anexo: Resultados de detección

► Resultados de los 18 grupos de peticiones

Grupo	TPR	FPR	F_1 -score
c00	$0,71 \pm 0,01$	$0,05 \pm 0,00$	$0,80 \pm 0,00$
c01	$0,72 \pm 0,01$	$0,05 \pm 0,01$	$0,80 \pm 0,00$
c02	$1,00 \pm 0,00$	$0,03 \pm 0,01$	$0,98 \pm 0,01$
c03	$1,00 \pm 0,00$	$0,03 \pm 0,00$	$0,98 \pm 0,00$
c04	$0,91 \pm 0,01$	$0,01 \pm 0,00$	$0,95 \pm 0,01$
c05	$0,92 \pm 0,01$	$0,01 \pm 0,00$	$0,95 \pm 0,00$
c06	$0,99 \pm 0,00$	$0,00 \pm 0,00$	$1,00 \pm 0,00$
c07	$0,99 \pm 0,00$	$0,00 \pm 0,00$	$1,00 \pm 0,00$
c08	$1,00 \pm 0,00$	$0,00 \pm 0,00$	$1,00 \pm 0,00$
c09	$1,00 \pm 0,00$	$0,00 \pm 0,00$	$1,00 \pm 0,00$
c10	$1,00 \pm 0,00$	$0,00 \pm 0,00$	$1,00 \pm 0,00$
c11	$1,00 \pm 0,00$	$0,01 \pm 0,00$	$0,99 \pm 0,00$
c12	$0,74 \pm 0,00$	$0,05 \pm 0,01$	$0,81 \pm 0,01$
c13	$0,74 \pm 0,00$	$0,05 \pm 0,01$	$0,81 \pm 0,01$
c14	$1,00 \pm 0,00$	$0,00 \pm 0,00$	$1,00 \pm 0,00$
c15	$1,00 \pm 0,00$	$0,00 \pm 0,00$	$1,00 \pm 0,00$
t00	$0,99 \pm 0,01$	$0,06 \pm 0,04$	$0,98 \pm 0,00$
t01	$1,00 \pm 0,00$	$0,09 \pm 0,06$	$0,99 \pm 0,01$
	$0,93 \pm 0,11$	$0,03 \pm 0,03$	$0,95 \pm 0,08$

Anexo: Resultados de detección

► Resultados con anomalías entre los datos de entrenamiento





`https://github.com/nico-ralf-ii-fpuna/tfg`