SECURITY • NETWORKING • MANAGEMENT

Guidelines for Web applications protection with dedicated Web Application Firewall

Prepared by: dr inż. Mariusz Stawowski, CISSP
Bartosz Kryński, Imperva Certified Security Engineer

INTRODUCTION

Security incident statistics and Web applications penetration tests prove that over 70% of Web applications around the world suffer from serious vulnerabilities which may compromise the company's data and reputation. Unsecure websites are commonly used by the criminals to distribute malware (so called drive-by download attacks). Most companies are unaware of this fact and fall victim to criminal and malware attacks.

Web applications are developed mainly to fulfill the company's specific business requirements. Very often the developers have limited resources and time, as well as insufficient competencies in the field of IT security. The developers main focus is to fulfill the functional requirements of an application, not to ensure its security. As a consequence, deployed application might contain vulnerabilities.

In such a case the application can be secured by implementing a dedicated security solution in the form of Web Application Firewall, as well as by employing a competent auditor[1] who can identify the application's vulnerabilities, which can be then corrected by the developer according to the audit results.

These recommendations for Web applications have been included in the PCI DSS[2] standard, which underlines the necessity to implement Web Application Firewall protections. The following part of the study describes in detail the guidelines for protecting Web applications.

WHY CONVENTIONAL SAFEGUARDS ARE INSUFFICIENT?

Network firewalls and Intrusion Prevention Systems (IPS) are foundations for developing the network security architecture (among other the security zones), as well as preventing against numerous network attacks (e.g. exploits of the network services and operating systems, network worms distribution, etc.). In reality it is impossible to ensure the security of Web applications using the conventional firewall and IPS protections (including UTM). These methods deploy techniques based on the signatures (i.e. the attacks patterns). Web applications, which are usually developed on business request, contain unique vulnerabilities not known to the authors of IPS protections. For this reason they are unable to create adequate signatures. Moreover, other techniques used in conventional IPS solutions, such as heuristics and protocol anomaly detection are also ineffective for Web applications protection as most of the attacks are conducted in legitimate HTTP traffic.

[1] More information about security audits performed by CLICO Professional Services is available at: http://www.clico.pl/services/clico_audit.pdf

[2] PCI Data Security Standard (PCI DSS), security standard published by PCI Security Standards Council, more information available at: https://www.pcisecuritystandards.org

HOW DOES DEDICATED WEB APPLICATION FIREWALL WORK?

An effective protection for Web applications is provided by Web Application Firewall (WAF), that is dedicated security solution. One of the most advanced WAF solutions is IMPERVA SecureSphere. The IMPERVA WAF is mainly based on the automatically created and updated profile of the protected Web application. WAF "learns" the application's structure, URLs, parameters, cookies, etc.

The purpose for creating the profile is to define the expected and proper users activity as observed when a Web application is accessed. Figure 1 presents a fragment of the Web application profile managed by WAF.

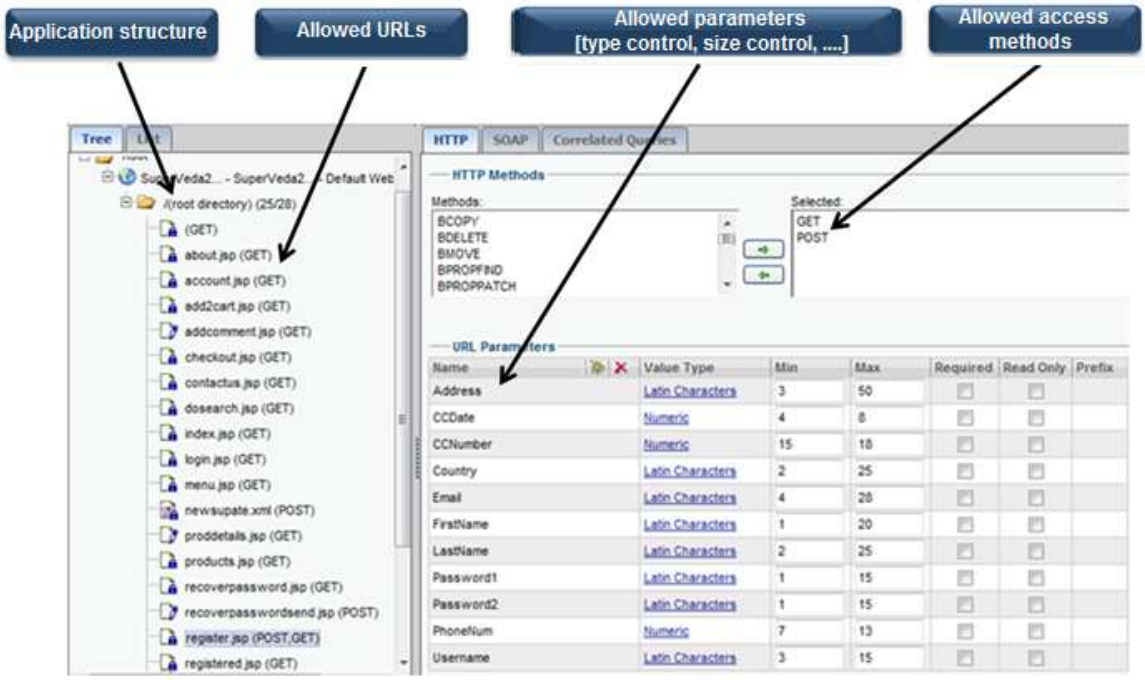


Figure 1. Web application protection profile built by IMPERVA WAF.

Main difference between the IPS and WAF is the approach to the network traffic control. IPS uses the blacklist approach, accepting whole network traffic except the packets identified as illegitimate. As a consequence all attacks not identified by IPS reach the Web application. WAF uses the whitelist approach, accepting the network traffic identified as legitimate, based on the created Web application profile. Thus the traffic not identified as legitimate is blocked by WAF and attacks cannot reach the Web application.

Selection of the adequate strategy for Web application protection is conditioned by specific requirements of the project (e.g. network and application structure, other deployed protections). IMPERVA SecureSphere WAF is deployed as a dedicated protection layer for Web application. In case of numerous business portals and other Web applications (e.g. e-commerce, e-banking) not only the Web application, but also the database require the same level of protection.

IMPERVA SecureSphere WAF system is a single-platform solution providing complete WAF functionality as well as Database Firewall (DBF). All of the Web application and database safeguards are managed by a dedicated management system, provisioned with adequate monitoring and reporting tools. Figure 2 presents the structure of IMPERVA WAF deployment in such an architecture.

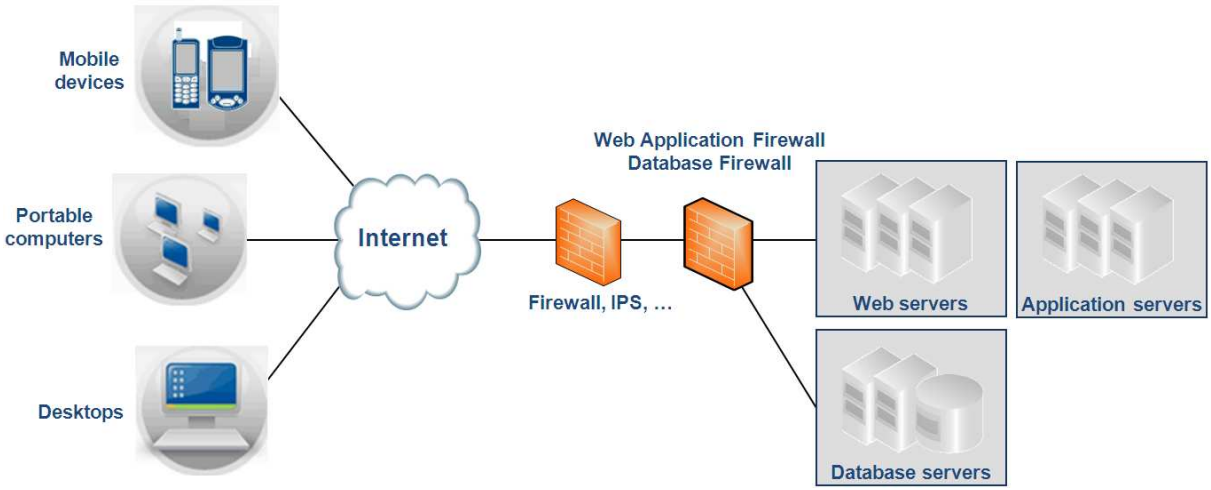


Figure 2. IMPERVA WAF as a dedicated protection layer for Web application.

HOW IS WEB APPLICATION SECURED?

In order to verify in details the effectiveness of IPS and WAF solutions in the scope Web application protection, the tests were carried out. Web application was targeted with attacks consisted of real exploits against existing application's vulnerabilities. The tests were performed on the IMPERVA WAF solution and an IPS solution from one of the world leading vendors (IPS tested with all available detection techniques and signatures activated).

During the tests the Web application was targeted with the following attacks:

1. Simple Reflected XSS.
2. Reflected XSS with character encoding.
3. Simple Stored XSS.
4. Stored XSS with character encoding.
5. Simple SQL-Injection.
6. SQL-Injection with SQL query combination.
7. Web application cookie manipulation.
8. Forceful Browsing.
9. Information Leakage.

Test 1. Simple Reflected XSS attack

-- XSS code is part of URL address (e.g. link sent in email or presented on Web page):
`http://192.168.53.90/dosearch.asp?string=<script>alert("test")</script>`

1a. IPS reaction

-- IPS detects script elements in URL parameter, detailed data is unavailable.

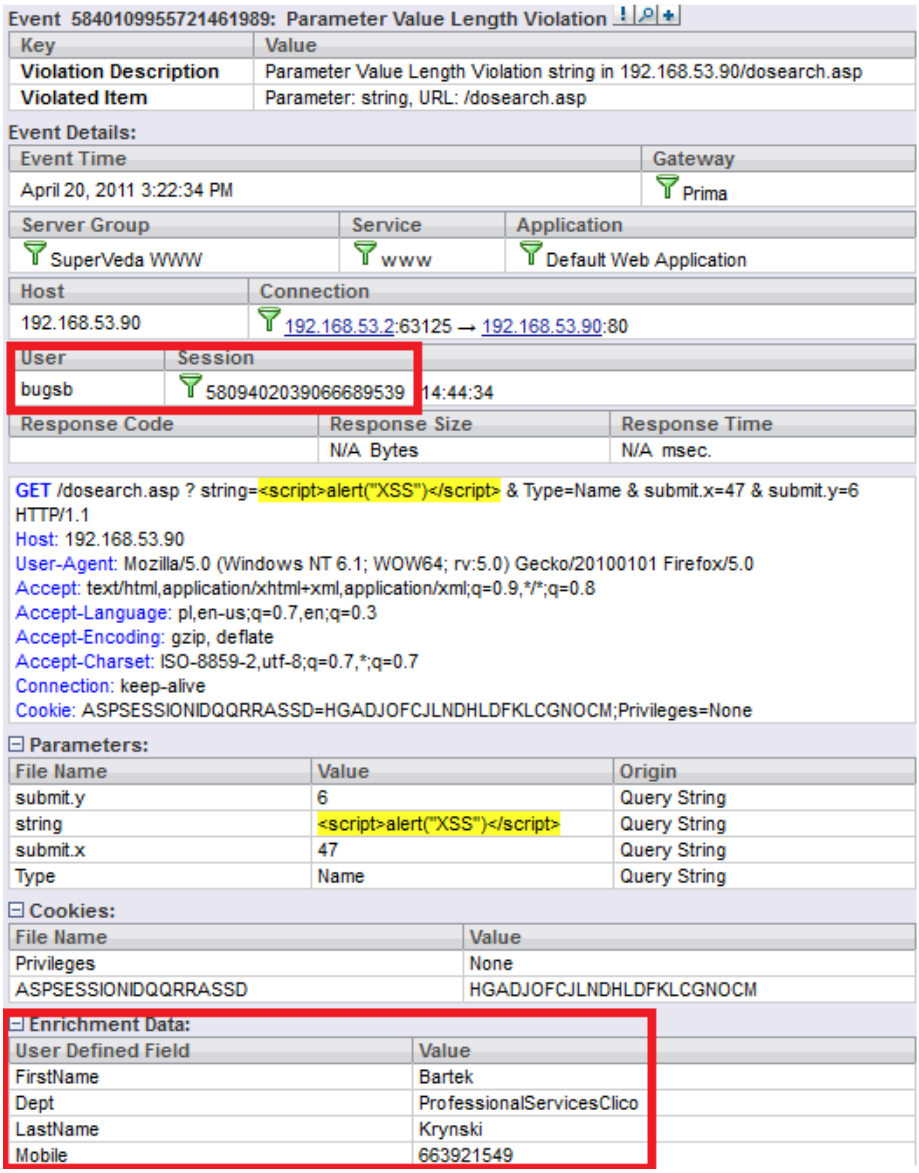
Category	Predefined
Subcategory	HTTP: HTML Script Tag Embedded in URL Parameters
Severity	Major
Device	IPS

1b. WAF reaction

-- WAF detects and displays illegitimate content and the size of parameter within URL

5010	⛔		15:12:28	1	Parameter Type Violation string in 192.168.53.90/dosearch.asp
5009	⛔		15:12:14	4	Parameter Value Length Violation string in 192.168.53.90/dosearch.asp
5008			**15:06:29**	**4**	**Multiple XSS - Basic-5 from 192.168.53.2**
5002	⛔		15:04:31	41	Distributed XSS - Basic-5(+)
5004	⛔		**14:56:23**	**19**	**Multiple Cross-site scripting from 192.168.53.2**

Having access to accurate information about the attack, such as the name of authenticated user and other user information (e.g. obtained from Active Directory, LDAP, databases or Human Resources system), telephone number, department, etc. The company's security administrator can thoroughly analyze the incident.



Event 5840109955721461989: Parameter Value Length Violation

Key	Value
Violation Description	Parameter Value Length Violation string in 192.168.53.90/dosearch.asp
Violated Item	Parameter: string, URL: /dosearch.asp

Event Details:

Event Time	Gateway
April 20, 2011 3:22:34 PM	Prima

Server Group	Service	Application
SuperVeda WWW	www	Default Web Application

Host	Connection
192.168.53.90	192.168.53.2:63125 → 192.168.53.90:80

User	Session	
bugsb	5809402039066689539	14:44:34

Response Code	Response Size	Response Time
	N/A Bytes	N/A msec.

GET /dosearch.asp ? string=<script>alert("XSS")</script> & Type=Name & submit.x=47 & submit.y=6
HTTP/1.1
Host: 192.168.53.90
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:5.0) Gecko/20100101 Firefox/5.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: pl,en-us;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Accept-Charset: ISO-8859-2,utf-8;q=0.7,*;q=0.7
Connection: keep-alive
Cookie: ASPSESSIONIDQQRRASSD=HGADJOFCJLNDHLDFKLCGNOCM;Privileges=None

Parameters:

File Name	Value	Origin
submit.y	6	Query String
string	<script>alert("XSS")</script>	Query String
submit.x	47	Query String
Type	Name	Query String

Cookies:

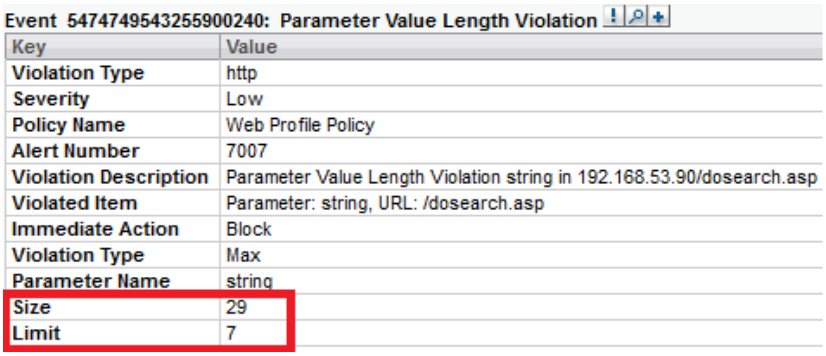
File Name	Value
Privileges	None
ASPSESSIONIDQQRRASSD	HGADJOFCJLNDHLDFKLCGNOCM

Enrichment Data:

User Defined Field	Value
FirstName	Bartek
Dept	ProfessionalServicesClico
LastName	Krynski
Mobile	663921549

-- WAF administrator can thoroughly analyze the underlying reasons for blocking a URL query.

---- violation of the size of the entry data,



Event 5474749543255900240: Parameter Value Length Violation

Key	Value
Violation Type	http
Severity	Low
Policy Name	Web Profile Policy
Alert Number	7007
Violation Description	Parameter Value Length Violation string in 192.168.53.90/dosearch.asp
Violated Item	Parameter: string, URL: /dosearch.asp
Immediate Action	Block
Violation Type	Max
Parameter Name	string
Size	29
Limit	7

---- violation of the type of the entry data,

Event 5474749543255900240: Parameter Type Violation

Key	Value
Violation Type	http
Severity	Medium
Policy Name	Web Profile Policy
Alert Number	7005
Violation Description	Parameter Type Violation string in 192.168.53.90/dosearch.asp
Violated Item	Parameter Type Violation
Immediate Action	Block
Parameter Name	string
Parameter Value	<script>alert("XSS")</script> ;
Unexpected Groups	Double Quote,Parenthesis,Angled Brackets,Slash

---- identification of the attack using signatures,

Event 5840109955721461957: Signature Violation

Key	Value
Violation Description	XSS - Basic-5
Violated Item	Location: parameters, Position: 8

If required, WAF administrator can customize learned profile, either from the log console or within the profile itself.

Event 5840109955721461989: Parameter Value Length Violation

Key	Value
Violation Description	Parameter Value Length Violation string in 192.168.5 ... asp
Violated Item	Parameter: string, URL: /dosearch.asp

Add to profile

URL Parameters

Name	Value Type	Min	Max	Required	Read Only	Prefix
Type	Latin Characters	0	1000			
string	Foreign Language Characters (UTF-8)	0	7			
submit.x	Numeric					
submit.y	Numeric					

Configure Value Type:

Custom Value Type:

Primary Value Type: Foreign Language Characters (UTF-8)

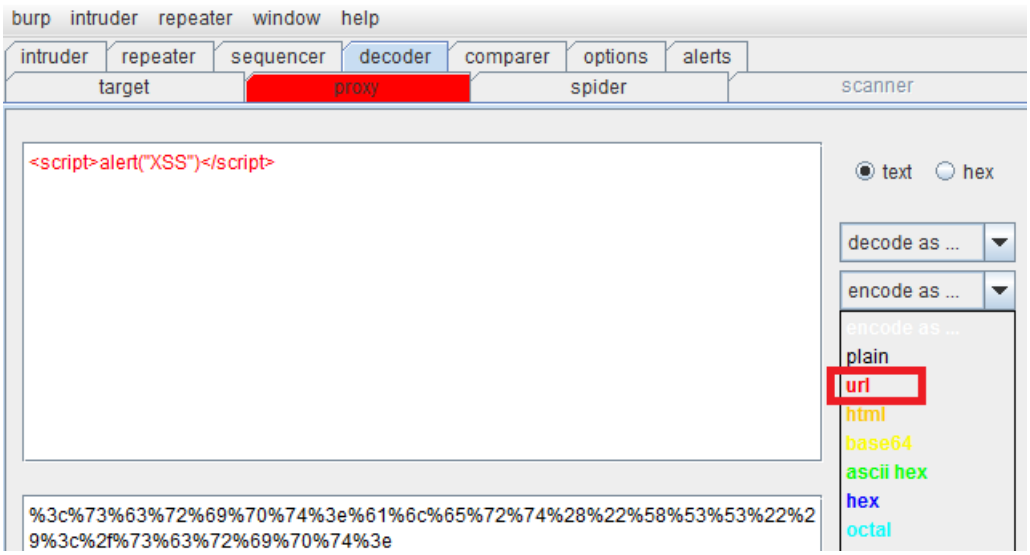
Additional Allowed Character Groups:

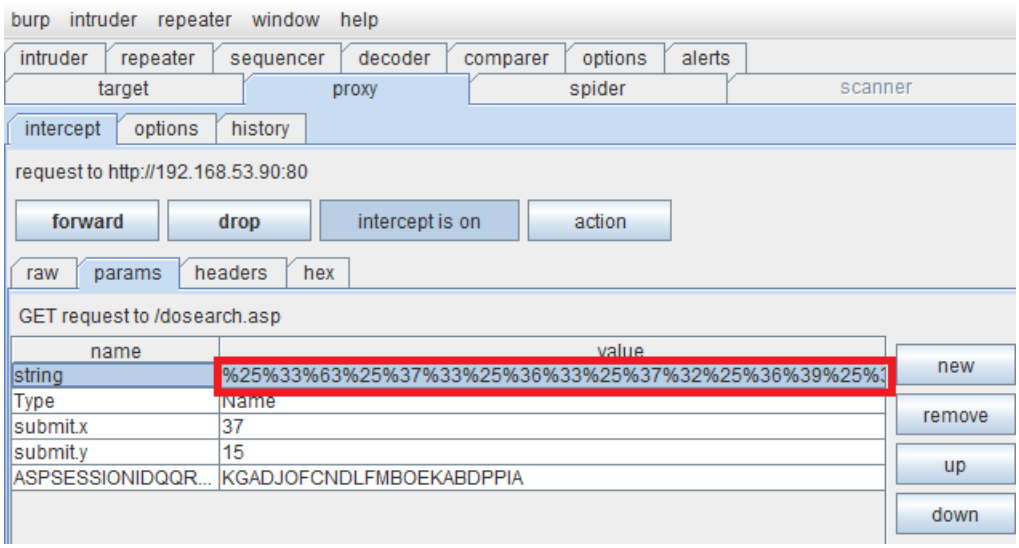
Semicolon	;	Dash	-
Ampersand	&	Others	# $: @ ^ _
Null	[NULL]	Concatenation	|
Plus Sign	+	Period	.
Double Quote	"	Asterix	*
Comma	,	OS Related Separators	! ' ~
Parenthesis	()	Brackets	[] {}
Percent Sign	%	HTTP Query String Separators	= ?
Angled Brackets	> <	ASCII Control Characters	
Slash	\ /	Quote	'

Select All | Unselect All | Save | Cancel

Test 2. Reflected XSS with character encoding

-- Using intercepting Proxy (e.g. Burp, Paros), XSS code is encoded within URL and inserted to the Web application.
Note: The configuration of Burp Proxy is presented in appendix 1.





2a. IPS reaction

-- Does not detect the attack.

2b. WAF reaction

-- Detects the attack and displays used encoding.

Event 5840109955721462013: Double URL Encoding

Key	Value
Violation Description	Double URL Encoding - parameter: string in 192.168.53.90/dosearch.asp
Violated Item	Parameter: string, URL: /dosearch.asp

Event Details:

Event Time	Gateway
April 20, 2011 4:00:37 PM	Prima

Server Group	Service	Application
SuperVeda WWW	www	Default Web Application

Host	Connection
192.168.53.90	192.168.53.2:52374 → 192.168.53.90:80

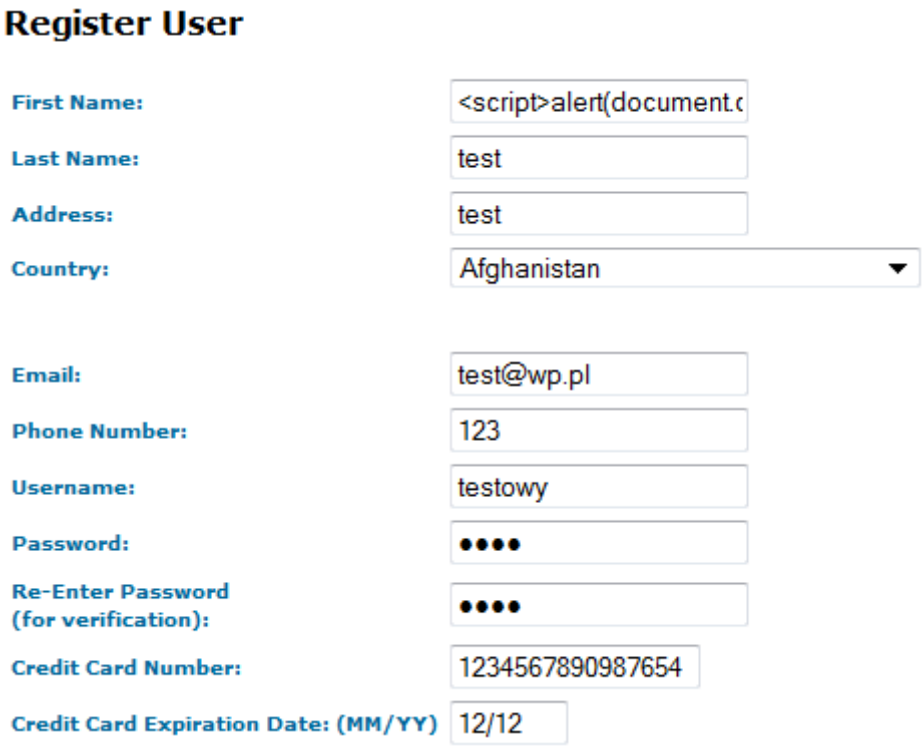
User	Session
	5809402039066689543 15:54:34

Response Code	Response Size	Response Time
	N/A Bytes	N/A msec.

GET /dosearch.asp ? string=%3c%73%63%72%69%70%74%3e%61%6c%65%72%74%28%22%58%53%53%22%29%3c%2f%73%63%72%69%70%74%3e & Type=Name & submit.x=32 & submit.y=14 HTTP/1.1

Test 3. Simple Stored XSS

-- 'First Name' field in Web registration form is filled with XSS code **`<script>alert(document.cookie)</script>`** which is then displayed on each instance of the user authentication.



-- As the attack result, the users are prompted with the value of their Cookie immediately after logging on to the Web application.

3a. IPS reaction

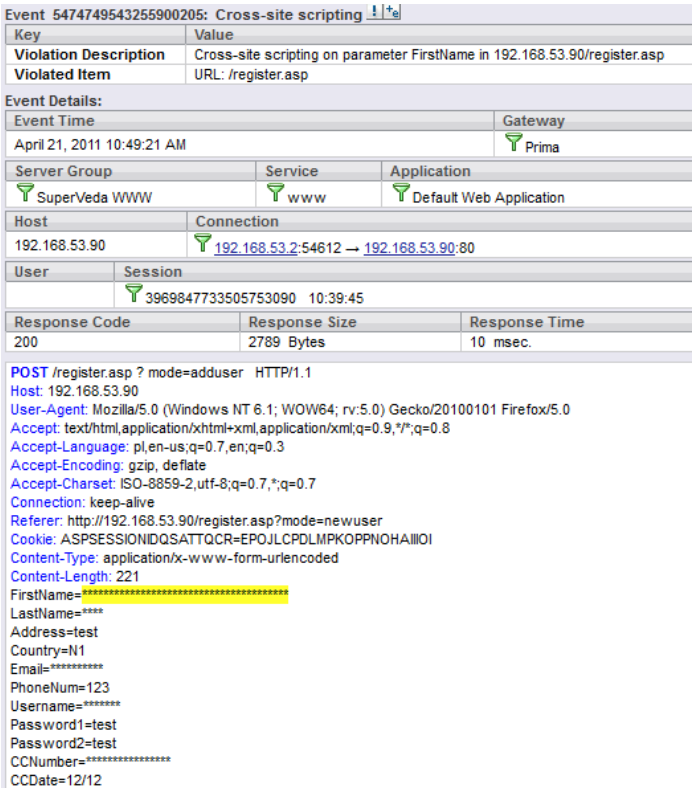
-- Does not detect the attack.

3b. WAF reaction

-- WAF detects the violation of the value of the URL query.

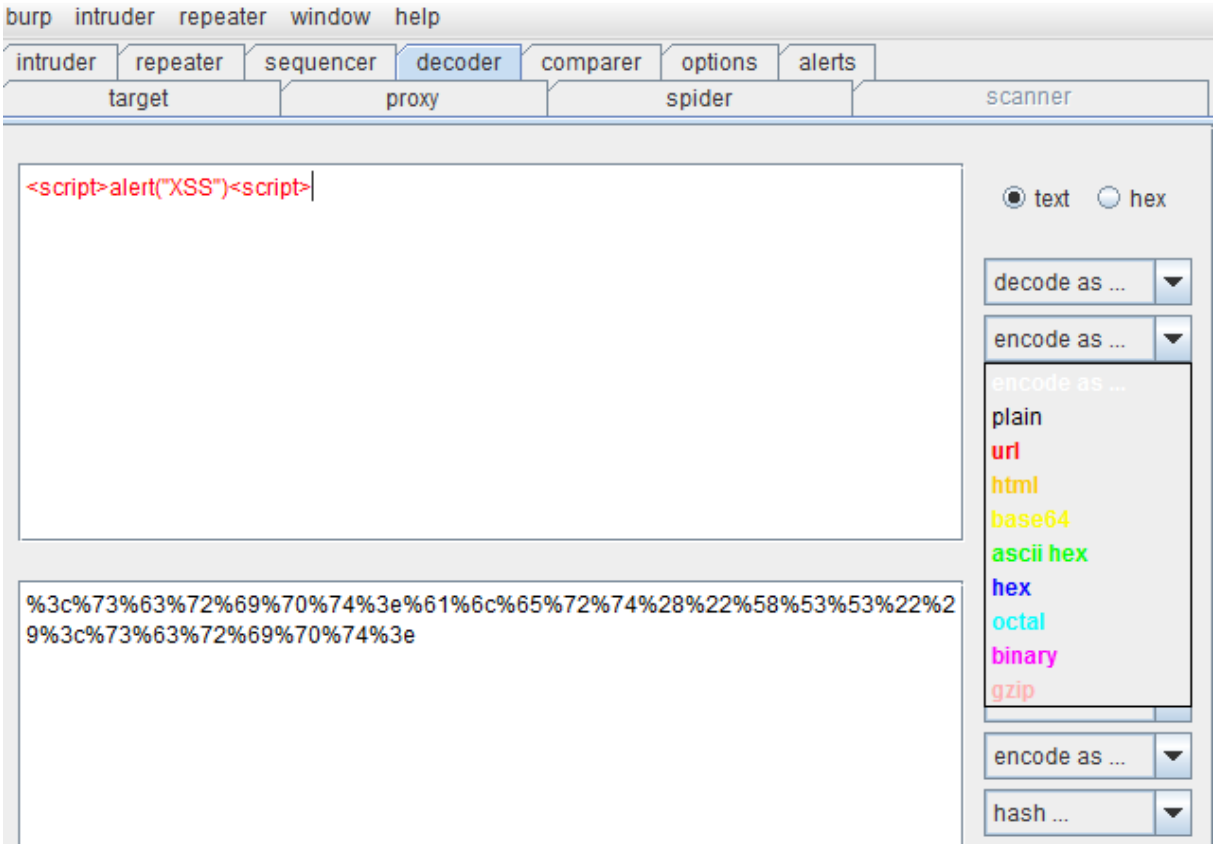
7003	⛔		10:49:21	9	Cross-site scripting from 192.168.53.2 in /register.asp - FirstName
7002			**10:49:21**	**9**	**Unauthorized Method POST for 192.168.53.90/register.asp**
7001			**10:49:21**	**42**	**Multiple XSS - Basic-5 from 192.168.53.2**
7004			**10:49:00**	**6**	**Suspicious Response Code**

Note: Additionally WAF offers the functionality of concealing the client data, therefore user names, passwords, etc. can be masked within logs.



Test 4. Reflected XSS with character encoding

-- The script (XSS code) is encoded as URL, and using intercepting Proxy (e.g. Burp, Paros) inserted in 'First Name' field in Web application form:

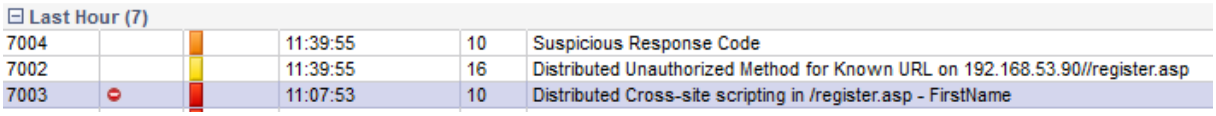


4a. IPS reaction

-- Does not detect the attack.

4b. WAF reaction

-- Detects the attack and displays used encoding (as in the case of previous attack) additionally pointing to other suspicious activities, such as using an unexpected http method (through an intercepting Proxy) and suspicious server response.



Test 5. Simple SQL-Injection

-- SQL query `' OR ''='` is typed into the Web application logon field, granting unauthorized access to the application.

5a. IPS reaction

-- IPS detects SQL-Injection attack within URL, detailed information on the attack is unavailable.

Category	Predefined
Subcategory	HTTP: SQL Injection In URL
Severity	Major
Device	IPS

5b. WAF reaction

-- Detects the attack and displays detailed information (as in the case of previous attacks).

No.			Updated	#	Alert Description
Last Hour (5)					
7009			12:05:39	1	SQL injection on parameter username in 192.168.53.90/login.asp
7008			**12:05:39**	**1**	**Parameter Type Violation username in 192.168.53.90/login.asp**

Event 5474749543255900370: SQL injection

Key	Value
Violation Type	http
Severity	High
Policy Name	Web Correlation Policy
Alert Number	7009
Violation Description	SQL injection on parameter username in 192.168.53.90/login.asp
Violated Item	URL: /login.asp
Immediate Action	Block
Input Type	parameter
Parameter Name	username
Parameter Value	********

Event Details:

Event Time	Gateway
April 21, 2011 12:05:38 PM	Prima

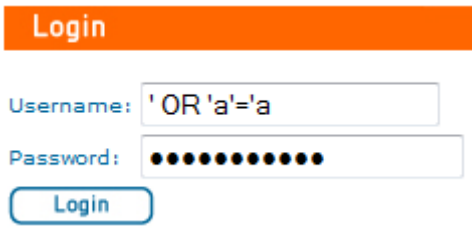
Server Group	Service	Application
SuperVeda WWW	www	Default Web Application

Host	Connection
192.168.53.90	192.168.53.2:55492 → 192.168.53.90:80

User	Session
' OR ''='	3969847733505753095 12:05:29

Test 6. SQL-Injection with SQL query combination

-- SQL query is provided in a modified form (e.g. `' OR 'a'='a`, `' OR '1000'='1000`, `' OR '1000'>'100`, etc).



6a. IPS reaction

-- Does not detect the attack.

6b. WAF reaction

-- Detects the attack and displays detailed information (as in the case of previous attacks) utilizing many security policy rules.

7009			12:09:45	5	Distributed SQL injection in /login.asp - username
7011			12:09:45	3	Parameter Value Length Violation password in 192.168.53.90/login.asp
7008			12:09:45	5	Parameter Type Violation username in 192.168.53.90/login.asp

Test 7. Manipulation of the Web application cookie

Response directed to the Web application is intercepted and the user role is modified.

-- Using intercepting Proxy, Cookie value is modified from ‘None’ to ‘Discount’.

raw | params | headers | hex

GET request to /add2cart.asp

type	name	value
URL	ProdID	6
URL	sale	no
cookie	ASPSESSIONIDQSA...	PPOJLCPDIJMDGNJOAEDDMBBL
cookie	Privileges	Discount

-- As a result, the user received an additional 10% discount.

Products Total: $91.85
Shipping: $0
Discount (10%): $9.185
==========
Total: $82.665

An application vulnerability could be analogically exploited if the application does not verify the values transmitted within the Discount parameter concealed from the browser. Using an intercepting proxy the intruder can replace the parameter value, so that he is granted the desired discounts.

7a. IPS reaction

-- Does not detect the attack.

7b. WAF reaction

-- Detects the Cookie Tampering attack and displays detailed information (as in the case of previous attacks).

Event 5474749543255900807: Cookie Tampering

Key	Value
Violation Type	http
Severity	Medium
Policy Name	Web Profile Policy
Alert Number	7014
Violation Description	Cookie Tampering on cookie Privileges: Expected NA, Observed Discount
Violated Item	Cookie: Privileges
Immediate Action	None
Cookie Name	Privileges
Observed Value	Discount
Allowed Value	NA

```
GET /add2cart.asp ? ProdID=13 & sale=no  HTTP/1.1
Accept: */*
Accept-Language: pl-PL
User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1; WOW64; Trident/4.0; SLCC2;
   .NET CLR 2.0.50727; .NET CLR 3.5.30729; .NET CLR 3.0.30729; Media Center PC 6.0;
   .NET4.0C)
Accept-Encoding: gzip, deflate
Proxy-Connection: Keep-Alive
Host: 192.168.53.90
Cookie: ASPSESSIONIDQSATTQCR=PPOJLCPDIJMDGNJOAEDDMBBL;Privileges=Discount
```

Parameters:

File Name	Value	Origin
sale	no	Query String
ProdID	13	Query String

Cookies:

File Name	Value
ASPSESSIONIDQSATTQCR	PPOJLCPDIJMDGNJOAEDDMBBL
Privileges	Discount

Enrichment Data:

User Defined Field	Value
FirstName	Bartek
Mobile	663921549
LastName	Krynski
Dept	ProfessionalServicesClico

Test 8. Forceful Browsing

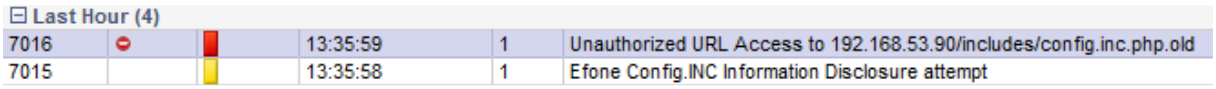
URL of the application configuration page **`/includes/config.inc.php.old`** is directly accessed without logging to the application.

8a. IPS reaction

-- Does not detect the attack.

8b. WAF reaction

-- Detects the attack and displays detailed information (as in the case of previous attacks).

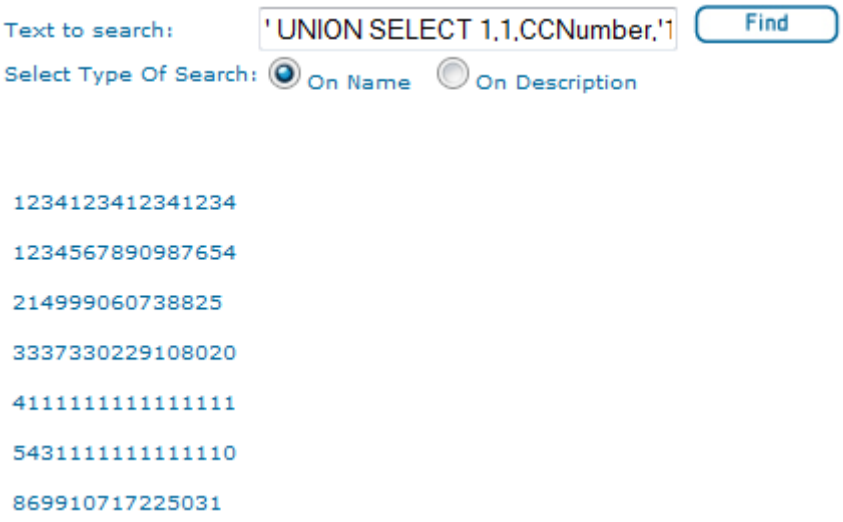


Test 9. Sensitive Information Leakage

Credit card numbers are read from the database by employing an SQL Injection attack within the search field:

```
' UNION SELECT 1,1,CCNumber,'1',1,1,'1' from users where '%'='
```

Apart from the products the search results also list credit card numbers of all registered users.

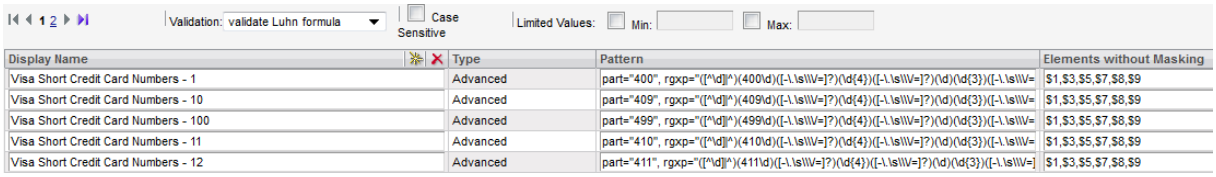


9a. IPS reaction

-- Does not detect the attack.

9b. WAF reaction

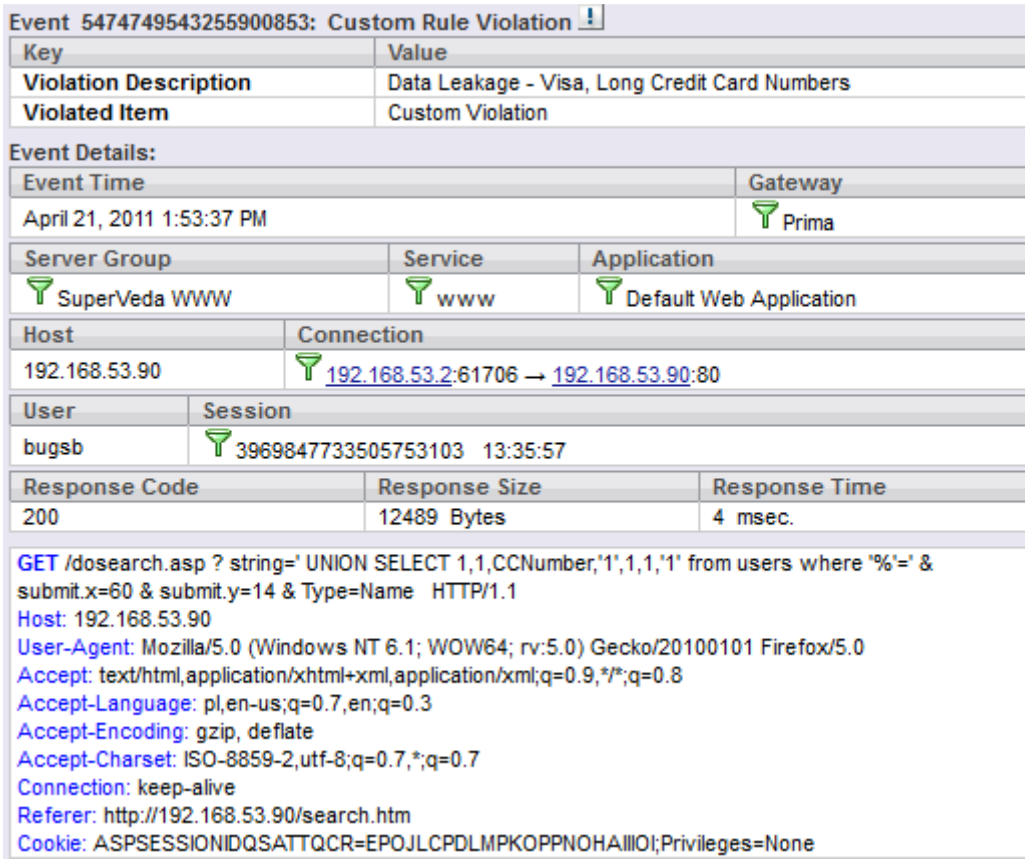
-- WAF identifies credit card numbers using defined regular expressions, optionally enabling verification with the Luhn algorithm.



1 2 | Validation: validate Luhn formula | Case Sensitive | Limited Values: Min: Max:

Display Name	Type	Pattern	Elements without Masking
Visa Short Credit Card Numbers - 1	Advanced	part="400", rgxp="([^\d]|^)(400\d)([-\.\s|\/=]?)(\d{4})([-\.\s|\/=]?)(\d)(\d{3})([-\.\s|\/=	$1,$3,$5,$7,$8,$9
Visa Short Credit Card Numbers - 10	Advanced	part="409", rgxp="([^\d]|^)(409\d)([-\.\s|\/=]?)(\d{4})([-\.\s|\/=]?)(\d)(\d{3})([-\.\s|\/=	$1,$3,$5,$7,$8,$9
Visa Short Credit Card Numbers - 100	Advanced	part="499", rgxp="([^\d]|^)(499\d)([-\.\s|\/=]?)(\d{4})([-\.\s|\/=]?)(\d)(\d{3})([-\.\s|\/=	$1,$3,$5,$7,$8,$9
Visa Short Credit Card Numbers - 11	Advanced	part="410", rgxp="([^\d]|^)(410\d)([-\.\s|\/=]?)(\d{4})([-\.\s|\/=]?)(\d)(\d{3})([-\.\s|\/=	$1,$3,$5,$7,$8,$9
Visa Short Credit Card Numbers - 12	Advanced	part="411", rgxp="([^\d]|^)(411\d)([-\.\s|\/=]?)(\d{4})([-\.\s|\/=]?)(\d)(\d{3})([-\.\s|\/=	$1,$3,$5,$7,$8,$9

WAF detects the attack and displays detailed information (as in the case of previous attacks). Moreover, if WAF has been deployed in the reverse proxy mode it offers the ability to mask sensitive information.



Event 5474749543255900853: Custom Rule Violation

Key	Value
Violation Description	Data Leakage - Visa, Long Credit Card Numbers
Violated Item	Custom Violation

Event Details:

Event Time	Gateway
April 21, 2011 1:53:37 PM	Prima

Server Group	Service	Application
SuperVeda WWW	www	Default Web Application

Host	Connection
192.168.53.90	192.168.53.2:61706 → 192.168.53.90:80

User	Session
bugsb	3969847733505753103 13:35:57

Response Code	Response Size	Response Time
200	12489 Bytes	4 msec.

```
GET /dosearch.asp ? string=' UNION SELECT 1,1,CCNumber,'1',1,1,'1' from users where '%'=' &
submit.x=60 & submit.y=14 & Type=Name  HTTP/1.1
Host: 192.168.53.90
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:5.0) Gecko/20100101 Firefox/5.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: pl,en-us;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Accept-Charset: ISO-8859-2,utf-8;q=0.7,*;q=0.7
Connection: keep-alive
Referer: http://192.168.53.90/search.htm
Cookie: ASPSESSIONIDQSATTQCR=EPOJLCPDLMPKOPPNOHAIIIOI;Privileges=None
```

The table below summarizes the results of basic tests of the effectiveness of IPS and WAF safeguards in the scope of Web application protection.

No.	Attack on Web application	IPS	WAF
1.	Simple Reflected XSS	• Attack detected as a script within URL parameter • Detailed information about the attack is unavailable	• Detection of an violation of HTTP GET parameter value • Provides detailed information about the attack
2.	Reflected XSS with character encoding	• Does not detect the attack[3]	• Detection of an violation of HTTP GET parameter value • Provides detailed information about the attack, including used encoding
3.	Simple Stored XSS	• Does not detect the attack	• Detection of violation of HTTP POST query value • Provides detailed information about the attack
4.	Stored XSS with character encoding	• Does not detect the attack.	• Detection of violation of HTTP POST query value • Provides detailed information about the attack, including used encoding
5.	Simple SQL-Injection	• SQL-Injection attack detected within URL • Detailed information about the attack is unavailable	• Detection of an violation of HTTP GET parameter value • Provides detailed information about the attack
6.	SQL-Injection with SQL query combination	• Does not detect the attack	• Detection of an violation of HTTP GET parameter value • Provides detailed information about the attack
7.	Manipulation of the Web application cookie	• Does not detect the attack	• Detection of an violation of cookie value • Provides detailed information about the attack
8.	Forceful Browsing	• Does not detect the attack	• Detection of the access to an unknown URL
9.	Information Leakage	• Does not detect the attack	• Identifies the attack and conceals sensitive data

The attacks launched during the tests belong to the most popular, basic and easiest to perform. Web applications may be targeted by numerous other attacks which cannot be prevented by deploying conventional protections (FW, IPS, UTM). Such attacks include Cross Site Request Forgery (CSRF), unauthorized file and Web server access (file enumeration), cookie poisoning, manipulation of masked fields and others.

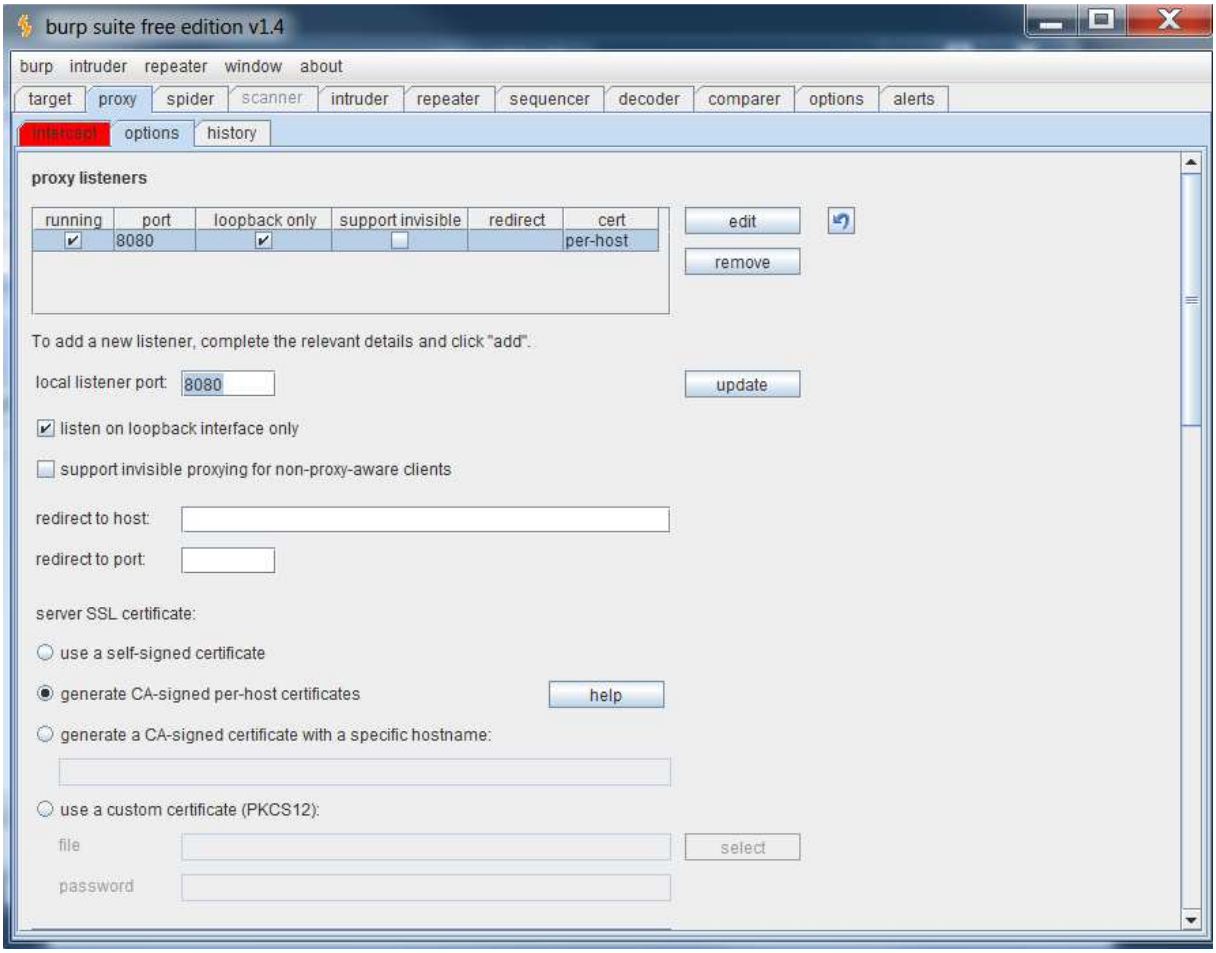
[3] This and other Web attacks included in the table could have been detected by IPS, if appropriate signatures had been formatted corresponding to specific attacks. Writing such signatures would require information about the method used to perform the attack. Gathering such data is practically impossible as IPS vendors have no knowledge about existing vulnerabilities of Web applications developed for the companies. Neither do they know the ways in which the vulnerabilities could be exploited. Various methods can be employed to conduct XSS, SQL-Injection and other attacks. It is not possible to provide a set of IPS signatures covering them all. Therefore, by means of using so called universal signatures, IPS can detect only basic Web attacks.

Appendix 1. How to configure Burp Proxy

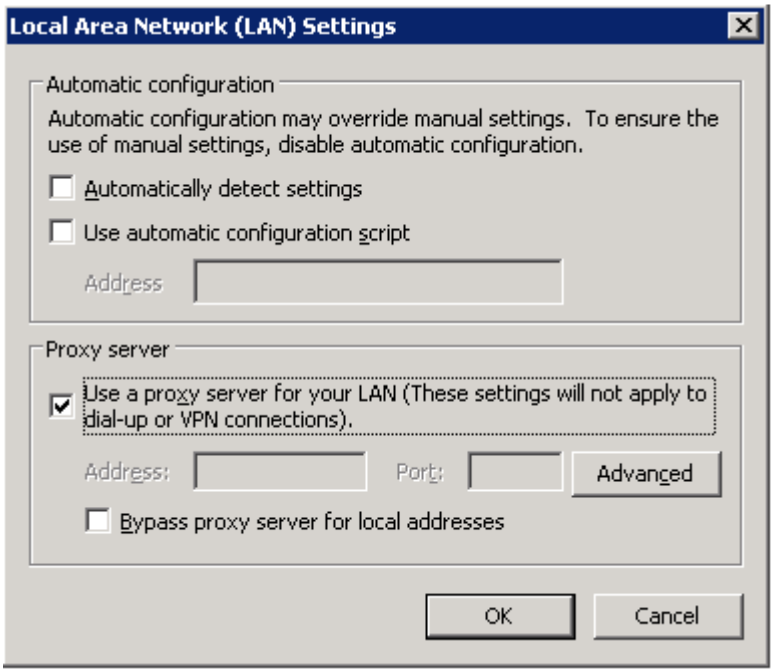
Burp is a suite of tools (like intercepting proxy, spider, scanner, sequencer, repeater, encoder etc.), helpful in performing security testing of web applications. BurpSuite Free Edition can be downloaded from: http://portswigger.net/burp/downloadfree.html

Burp Suite is a portable application, so there is no need to install it, to run we need to execute "suite.bat" file.

To configure proxy communication port we need to go to proxy -> options tab, and verify if proxy listener port is correct.



In the next step we need to redirect browser (Internet Explorer, Firefox or other) to direct analized traffic through proxy. In case of Internet Explorer we need to get to Tools -> Internet Options -> Connections -> Lan Settings -> Proxy Server and turn on option “Use a proxy server for your LAN”. For this setting address and port should be left in default (127.0.0.1/loopback, port: 8080).



From now on all communication between client browser and web application can be analized and modified by intercepting proxy.