

Pentest com Kali Linux





Instrutor: Vitor Mazuco

<http://facebook.com/vitormazuco>

Email: vitor.mazuco@gmail.com

WebSite: <http://vmzsolutions.com.br>



O OWASP ZAP (Zed Attack Proxy) é uma ferramenta muito versátil para testes de segurança na web. Ele tem um proxy, passivo e ativo que acha as vulnerabilidade por scanners, fuzzer, spider, HTTP solicitação remetente e alguns outros recursos interessantes.

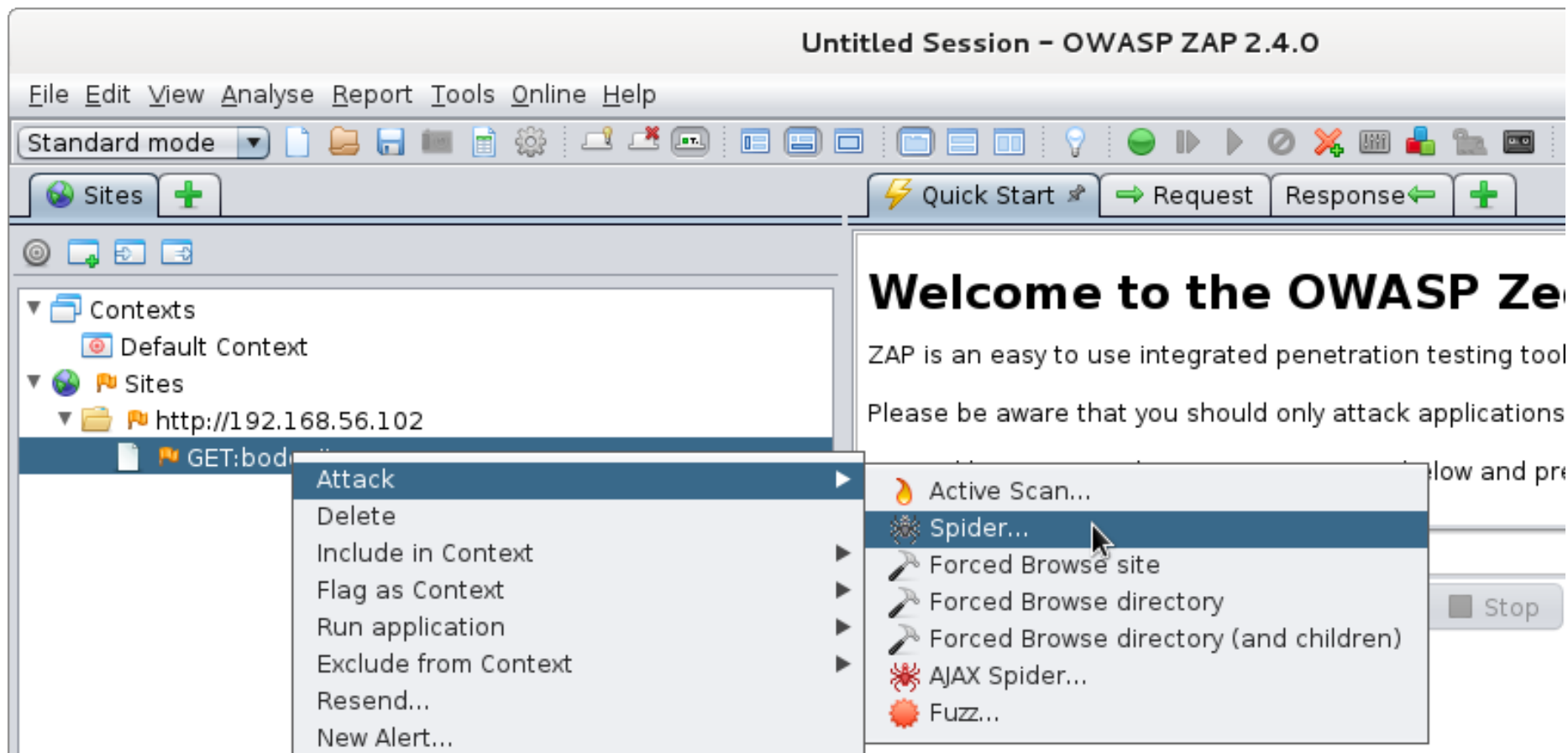


Para que o ZAP seja executado configure o navegador com o proxy na porta 8080. Agora, entre no ip da sua OWASP:

<http://192.168.1.163/bodgeit/>

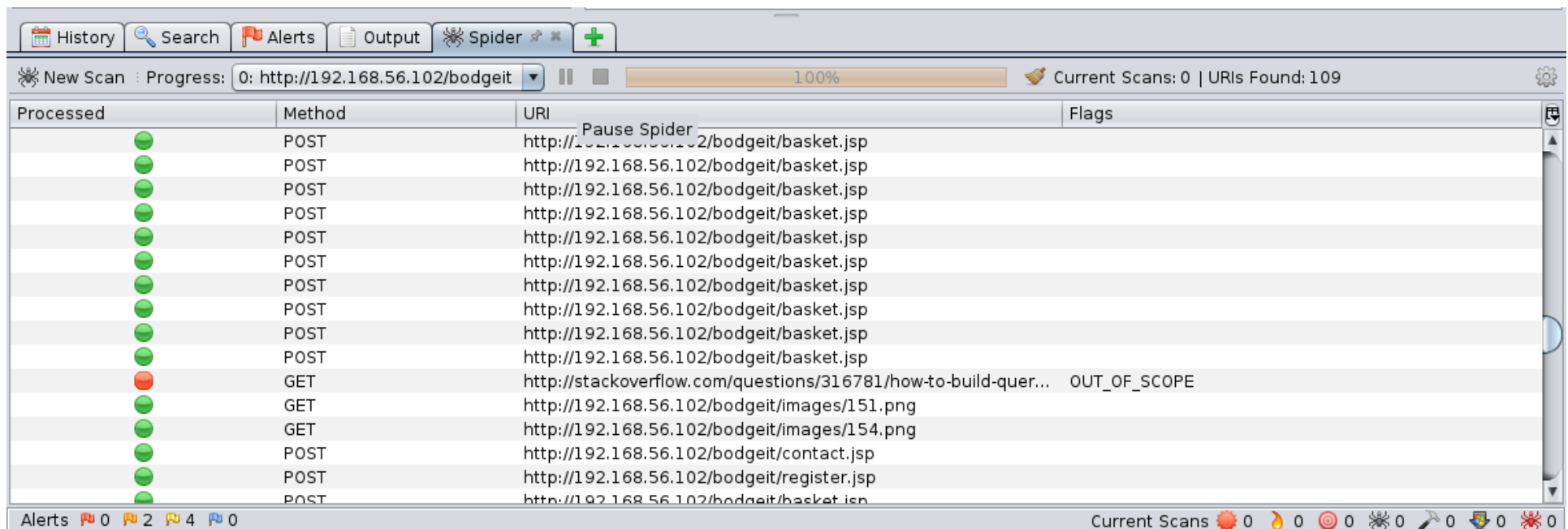


ZAP Spider





Os resultados serão exibidos no painel inferior da guia Spider:



The screenshot shows the ZAP Spider tool interface. At the top, there are tabs for History, Search, Alerts, Output, and Spider. The Spider tab is active, showing a progress bar at 100% and a status bar indicating 'Current Scans: 0 | URIs Found: 109'. Below the progress bar is a table with columns: Processed, Method, URI, and Flags. The table lists various URIs and their corresponding HTTP methods. Most entries are marked with a green circle in the 'Processed' column, indicating they were successfully processed. One entry, 'http://stackoverflow.com/questions/316781/how-to-build-quer...', is marked with a red circle and has the flag 'OUT_OF_SCOPE'.

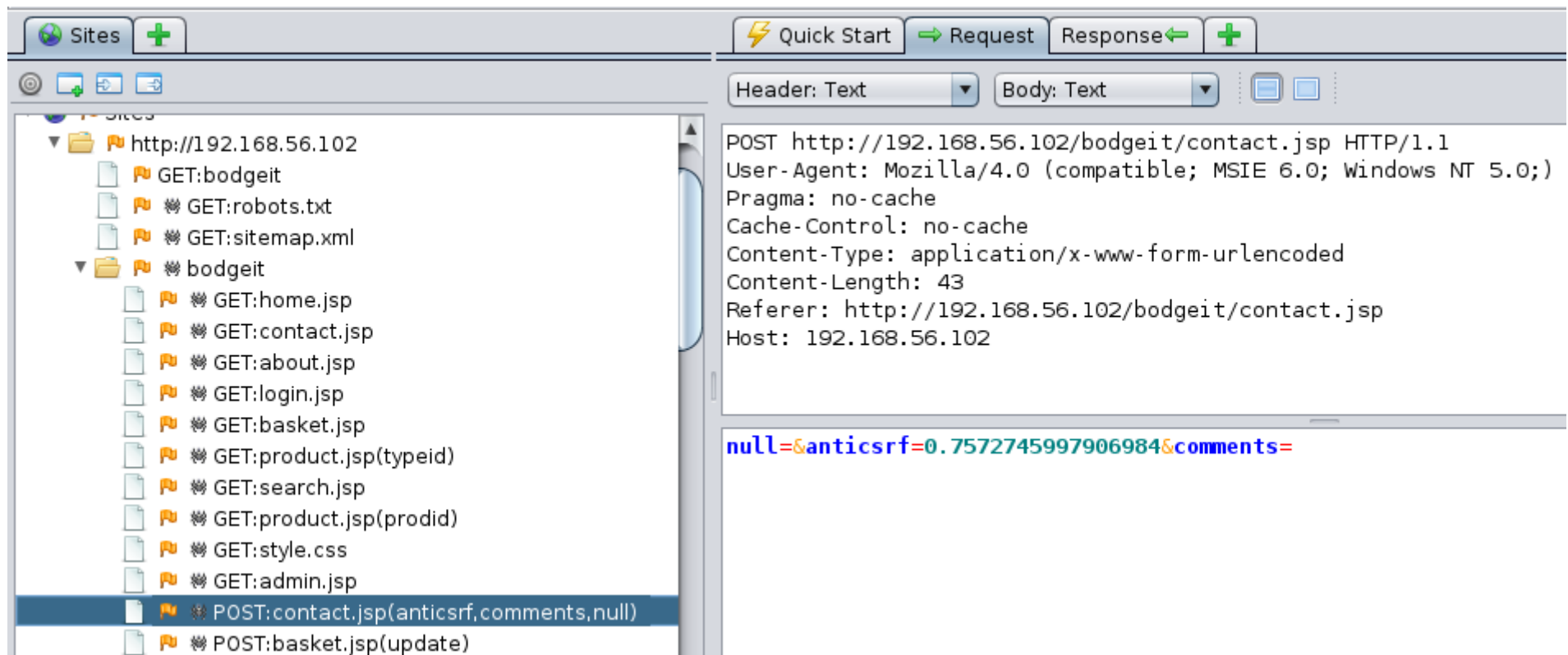
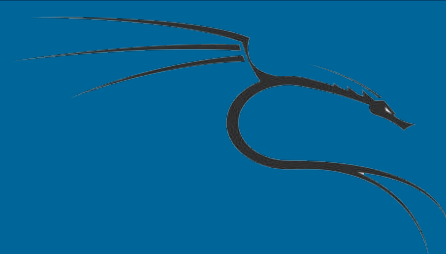
Processed	Method	URI	Flags
●	POST	http://192.168.56.102/bodgeit/basket.jsp	
●	POST	http://192.168.56.102/bodgeit/basket.jsp	
●	POST	http://192.168.56.102/bodgeit/basket.jsp	
●	POST	http://192.168.56.102/bodgeit/basket.jsp	
●	POST	http://192.168.56.102/bodgeit/basket.jsp	
●	POST	http://192.168.56.102/bodgeit/basket.jsp	
●	POST	http://192.168.56.102/bodgeit/basket.jsp	
●	POST	http://192.168.56.102/bodgeit/basket.jsp	
●	POST	http://192.168.56.102/bodgeit/basket.jsp	
●	POST	http://192.168.56.102/bodgeit/basket.jsp	
●	POST	http://192.168.56.102/bodgeit/basket.jsp	
●	GET	http://stackoverflow.com/questions/316781/how-to-build-quer...	OUT_OF_SCOPE
●	GET	http://192.168.56.102/bodgeit/images/151.png	
●	GET	http://192.168.56.102/bodgeit/images/154.png	
●	POST	http://192.168.56.102/bodgeit/contact.jsp	
●	POST	http://192.168.56.102/bodgeit/register.jsp	
●	POST	http://192.168.56.102/bodgeit/basket.jsp	

Alerts 0 2 4 0 Current Scans 0 0 0 0 0 0 0 0

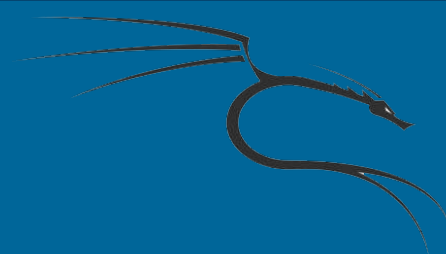


Se queremos analisar as requests e responses de arquivos individuais, vamos para a guia Sites e abrimos a pasta do site e a pasta bodgeit dentro dela. Vamos dar uma olhada no: `POST:contact.jsp(anticsrf,comments,null)`

ZAP Spider



No lado direito, podemos ver o pedido completo feito, incluindo os parâmetros utilizados (metade inferior).



Na metade superior, podemos ver o cabeçalho da resposta, incluindo o banner do servidor e o cookie de sessão, e na metade inferior temos a resposta completa em HTML. Com isso poderemos obter um cookie de um usuário autenticado pode ser usada para seqüestrar a sessão do usuário e executar ações que as personificam.



ZAP Spider

The screenshot displays the ZAP Spider tool's interface. At the top, there are four buttons: 'Quick Start' (lightning bolt icon), 'Request' (green arrow icon), 'Response' (green arrow icon), and a green plus icon. Below these buttons, there are two dropdown menus: 'Header: Text' and 'Body: Text'. To the right of these are two small square icons. The main area is divided into two sections. The top section shows the raw HTTP response text, and the bottom section shows the HTML body content with syntax highlighting.

HTTP/1.1 200 OK
Date: Sun, 12 Jul 2015 20:09:43 GMT
Server: Apache-Coyote/1.1
Content-Type: text/html
Content-Length: 2436
Set-Cookie: JSESSIONID=A2033EB7AF934FCDB1BF75CE25524271; Path=/
Via: 1.1 127.0.1.1
Vary: Accept-Encoding

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2//EN">
<html>
<head>
<title>The BodgeIt Store</title>
<link href="style.css" rel="stylesheet" type="text/css" />
<script type="text/javascript" src="./js/util.js"></script>
</head>
<body>

<center>
<table width="80%" class="border">
<tr bgcolor="#C0C0C0>
```