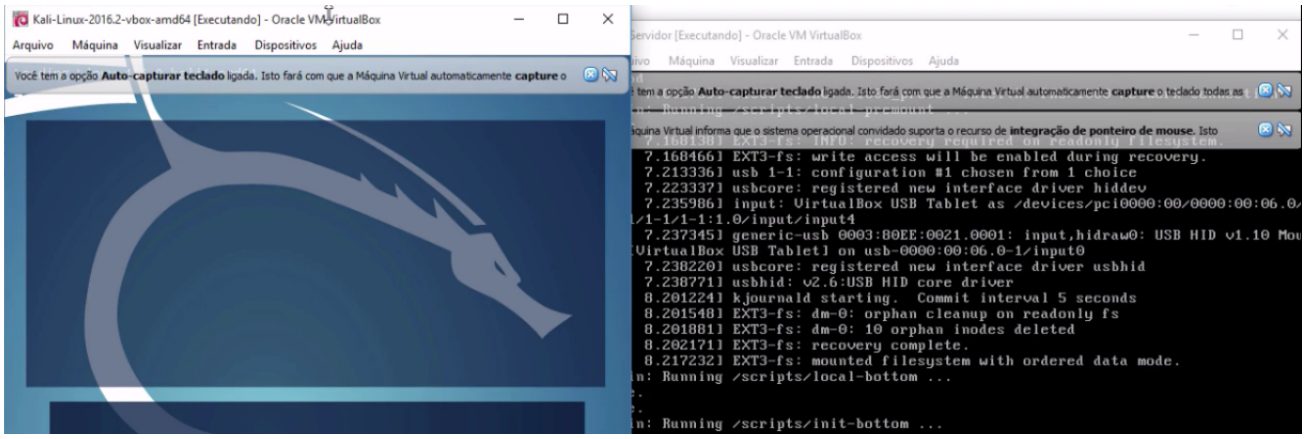
05

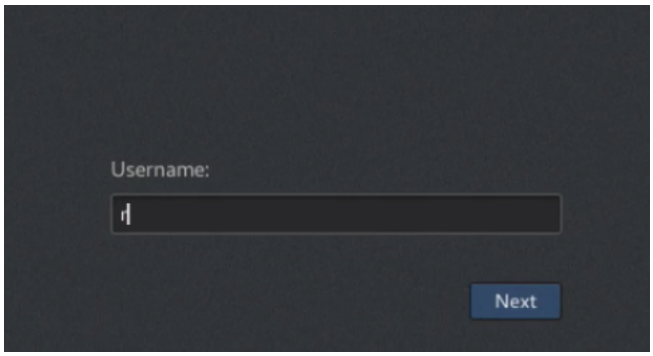
SQL Injection

Transcrição

Chegou o momento de inicializar as máquinas para realizar os primeiros ataques. Para fazer isso precisamos selecionar os dois ambientes e clicar em "Iniciar". Fazendo isso teremos duas abas com os respectivos ambientes rodando:

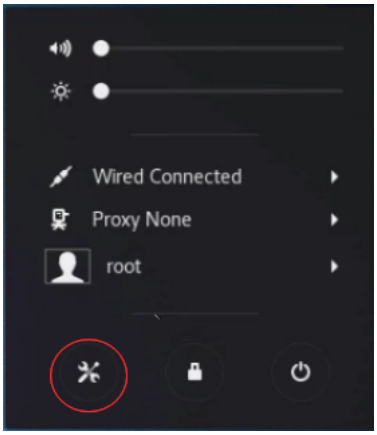


O Kali Linux (o ambiente referente ao hacker) demora um pouco para carregar. Ao finalizar o *boot* do Kali Linux aparece a seguinte tela:



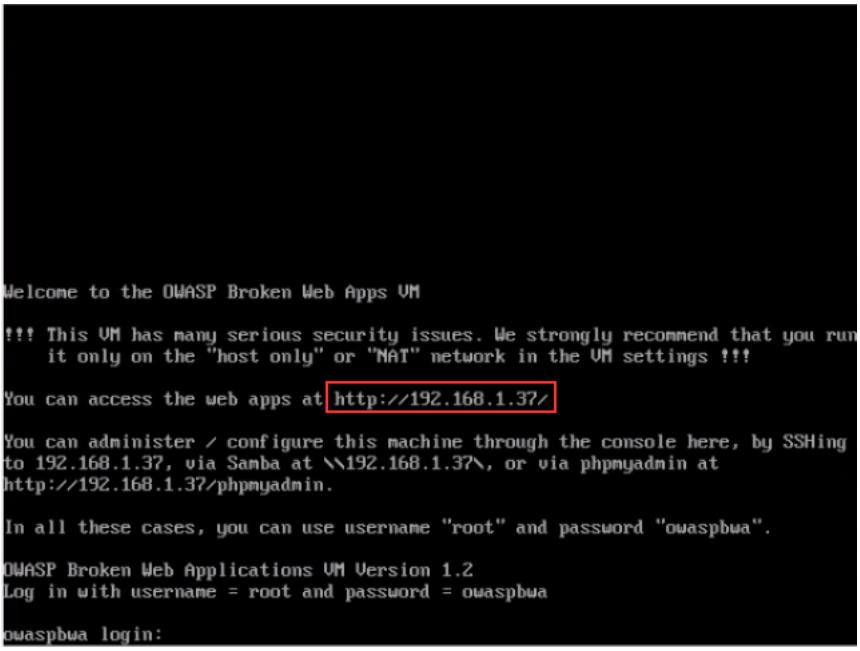
Basta preencher um *Username*, criar a *password* e dar um "Sign up", assim, aparecerá na tela o Kali Linux.

Antes de simular ataques vamos alterar uma pequena configuração do teclado. Nós selecionamos a seta na parte inferior direita da tela do Kali Linux e clicaremos no símbolo das ferramentas:

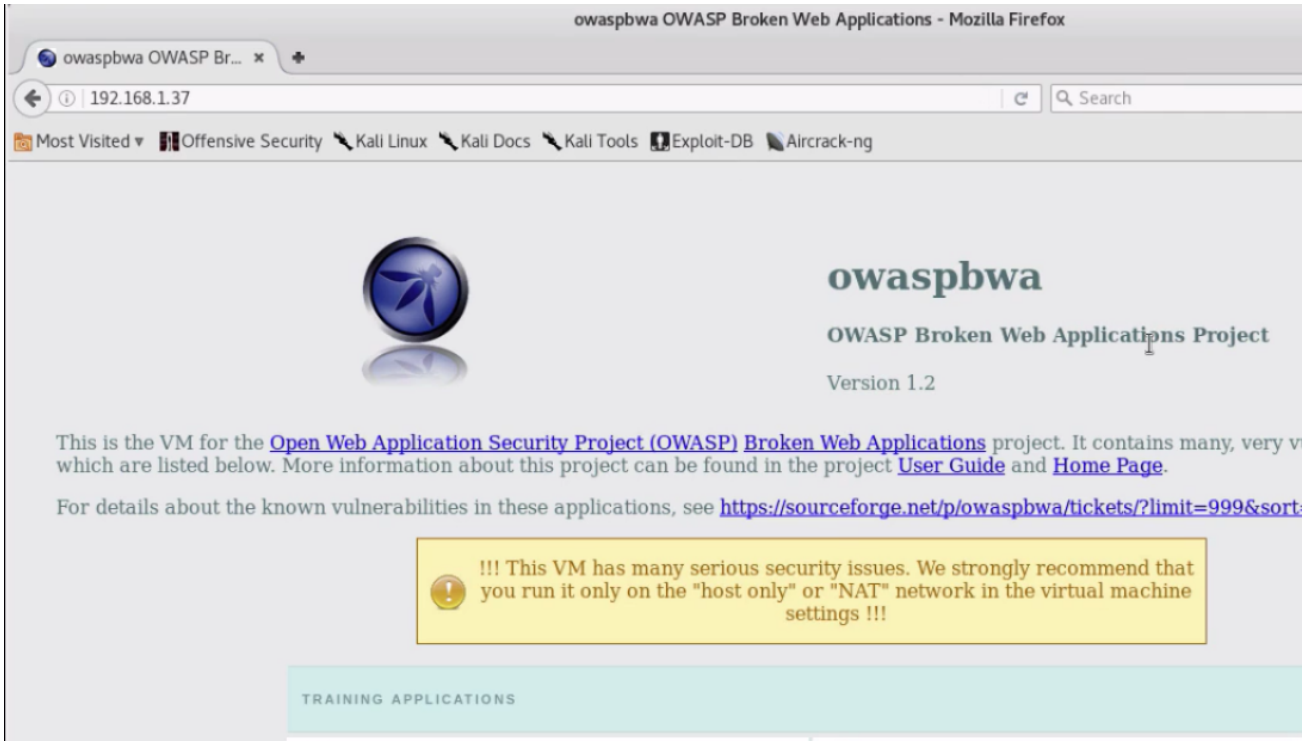


Abrirá uma janela com várias áreas e nós selecionamos o "Region & Language". Surgirá um formulário, o *Input sources*, e nele vamos clicar no símbolo de "+" e escolhemos o "Português(Brazil)". Por fim, basta clicar em "Add" e a opção escolhida será adicionada. Com o teclado configurado da maneira correta podemos acessar as páginas do servidor.

Para navegar nelas é preciso conhecer a URL de acesso, então, vamos retornar a aba do servidor:



Nessa tela podemos verificar o endereço fornecido. Voltamos no Kali Linux, abrimos o navegador e inserimos a URL para acessar a página vulnerável. Digitamos `192.168.1.37` e teremos o seguinte:

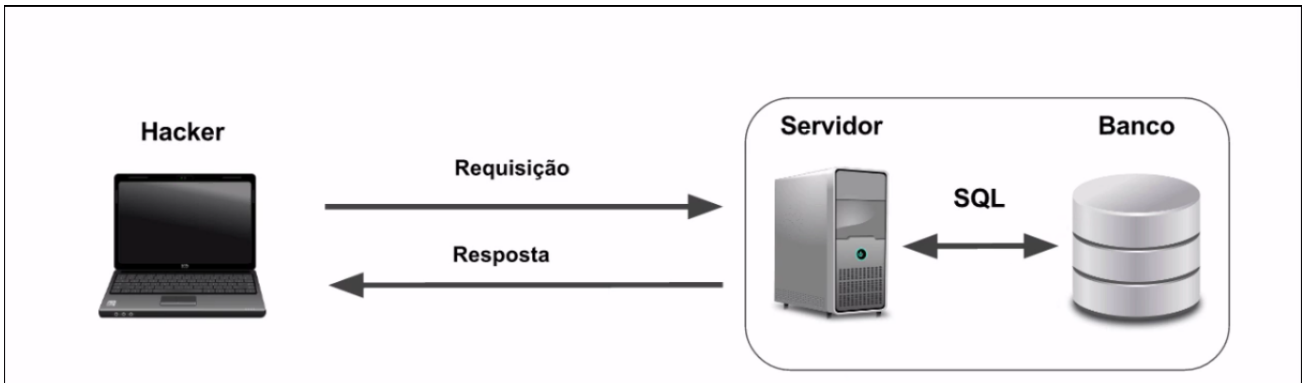


Dando um scroll na página, clicaremos no item `OWASP Multillidae II` que será o primeiro site vulnerável com que trabalharemos. Teremos o seguinte:

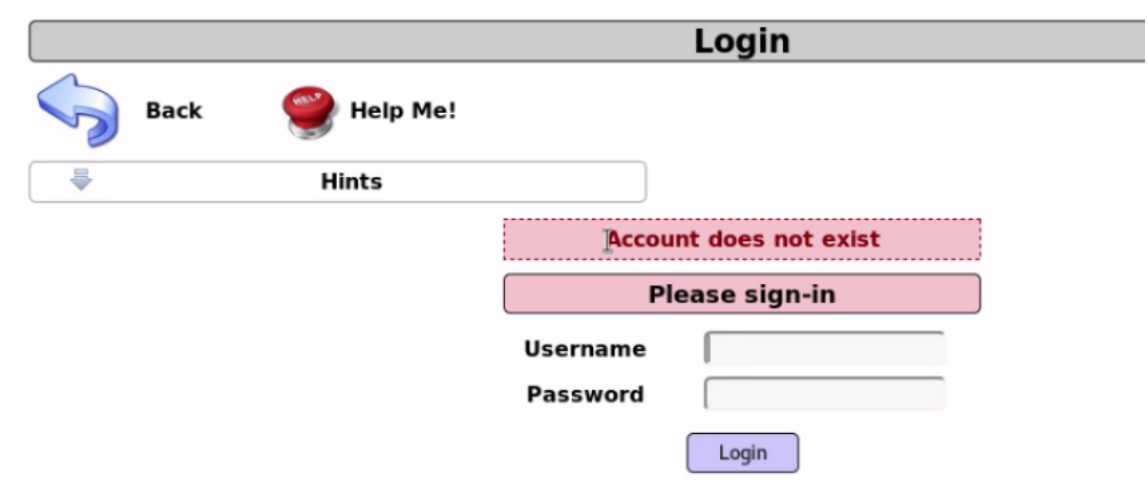
O *security level* dessa página é zero e nesse momento não estamos logados. Portanto, vamos clicar no "*Login register*" e aparecerá uma nova janela que pede usuário e senha. Podemos preencher isso com "Rafael" e "alura" (usuário e senha), por exemplo.

Quando preenchemos as informações e apertamos o botão de login, fazemos uma requisição ao servidor. Isto é, nós passamos para o servidor os parâmetros que estão no usuário e o servidor consulta no banco de dados verificando se o registro existe. Dependendo do que o banco verificar a resposta retornada ao servidor será: existe ou não existe!

A partir disso o servidor dá uma resposta ao usuário (no caso o hacker). Se o acesso for permitido, nós teremos o acesso ao site, caso contrário, receberemos uma mensagem negando. A comunicação que ocorre entre o Servidor e o Banco é feita usando a linguagem SQL. Resumindo:



O que nós faremos para resolver esse problema é induzir o banco a um comportamento indevido, portanto, vamos inserir códigos SQL. Feita essa introdução teórica, vamos retornar ao site. Se nós inserirmos o nome "Rafael" e "alura", como login e senha respectivamente, teremos o acesso negado:



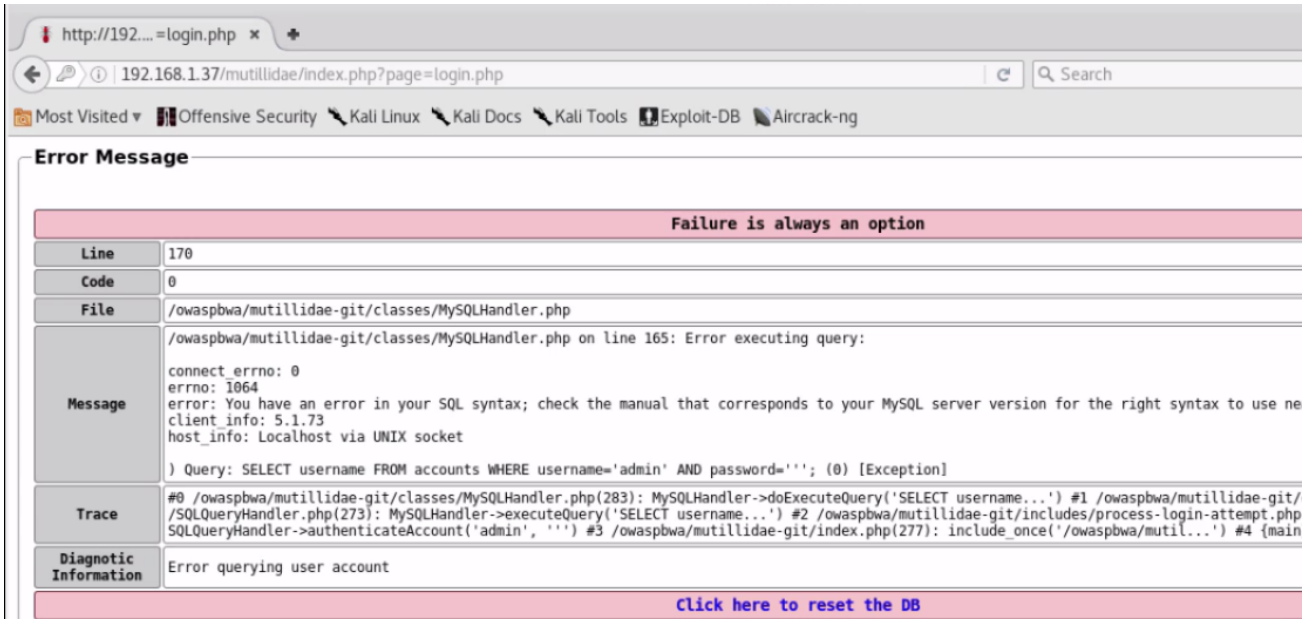
A mensagem devolvida afirma que a conta inserida não existe. Podemos interpretar essa mensagem da seguinte maneira: se inserirmos um usuário que existe, mas uma senha incorreta, é esperado uma resposta diferente, pois, a conta vai existir mesmo que a senha esteja errada.

Como podemos realizar o login na máquina se não temos nem a informação do login nem a do usuário? É preciso conhecer um ou outro. No caso do presente exercício, nós temos conhecimento de que o *Username* é `admin` . Ao preencher o usuário como *admin* e a senha como qualquer coisa, nós teremos uma mensagem avisando: *Password incorrect*.

A mensagem que recebemos nos fornece a certeza de que o `admin` existe!

A segunda pergunta que podemos fazer para burlar o sistema é: "será que o desenvolvedor filtra o que pode ser escrito nos campos?" Por exemplo, vamos escrever `admin` e o password nós colocaremos apenas uma aspas simples. A aspas simples delimita *strings* e caracteres permitindo a inserção de códigos SQL para serem interpretados no banco de dados. Então, se o desenvolvedor não previu o uso de aspas simples, podemos verificar a reação:

Essa mensagem avisa que uma exceção aconteceu. Isso significa que o sistema não se comportou da maneira como deveria. Acima, na mesma página, temos as informações do porquê da exceção:



Isso nos informa que ocorreu um erro na sintaxe. Essa falha deve-se às aspas simples que nós inserimos e que não são fechadas. Outra dica é verificar o Manual da SQL. Ainda, ele também nos informa a *query* que vai para o banco de dados.

Com essa *Exception*, foi possível mapear diversas informações! Com isso nós sabemos como a query é construída e qual o banco de dados é utilizado. Em seguida, vamos copiar a *query* e colá-la no banco de notas para analisar com mais tranquilidade. Teremos:

```
SELECT username FROM accounts WHERE username='admin' AND password='''
```

Nós havíamos preenchido de forma incorreta, portanto, vamos deixar representado como `x`. O que pode ser feito diante disso? Nós podemos utilizar as aspas simples para delimitar o caractere e acrescentar uma mensagem que é sempre verdadeira, por exemplo, `a=a`. Como podemos juntar essas duas informações o `x` e o `a=a` de maneira que ambas se tornem verdadeiras? Simples! Nós podemos adicionar o `or`, assim:

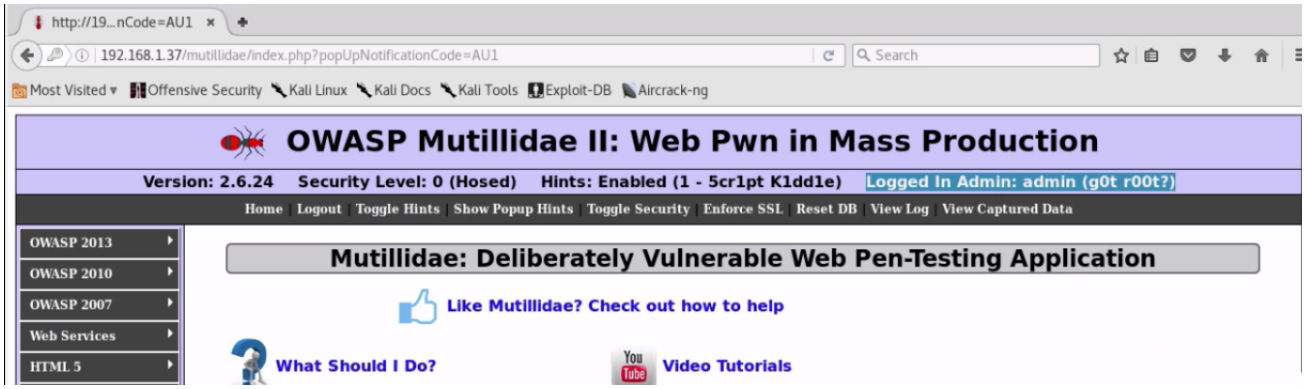
```
SELECT username FROM accounts WHERE username='admin' AND password='x' or a=a'
```

Com isso estamos dizendo: a password é x ou a=a. Desta maneira, não é preciso que as duas sejam verdadeiras, mas apenas uma. Portanto, toda a informação que acabamos de inserir junto a `password` é verdadeira. Ainda, se deixarmos o segundo `a` com apenas uma aspas parece que ele fica aberto e isso acusará um erro sintático. Portanto, é preciso inserir aspas no outro lado dele para que continue igual: `'a'='a'`.

Lembrando que a primeira aspas que está junto do `x` faz parte do sistema e a última que acompanha o segundo `a` também. Teremos:

```
SELECT username FROM accounts WHERE username='admin' AND password='x' or 'a'='a'
```

Vamos verificar se inserirmos `x' or 'a'='a` no lugar da senha da página se isso nos permite ter acesso à página. Pois, estaremos preenchendo um *username*, `admin`, que já sabemos ser verdadeiro e uma senha que também é. Vamos testar? Preencheremos o *username* como `admin` e a senha como `x' or 'a'='a` e teremos o seguinte:



Ou seja, estamos logados como admin.

O que fizemos nesse caso foi inserir código SQL para que o banco de dados se comportasse de maneira indevida. Resumindo, inserimos um usuário verdadeiro e uma senha onde o `'a'='a'` é verdadeiro. Portanto, com o *username* e o *login* sendo verídicos, nós temos acesso à página restrita do site!