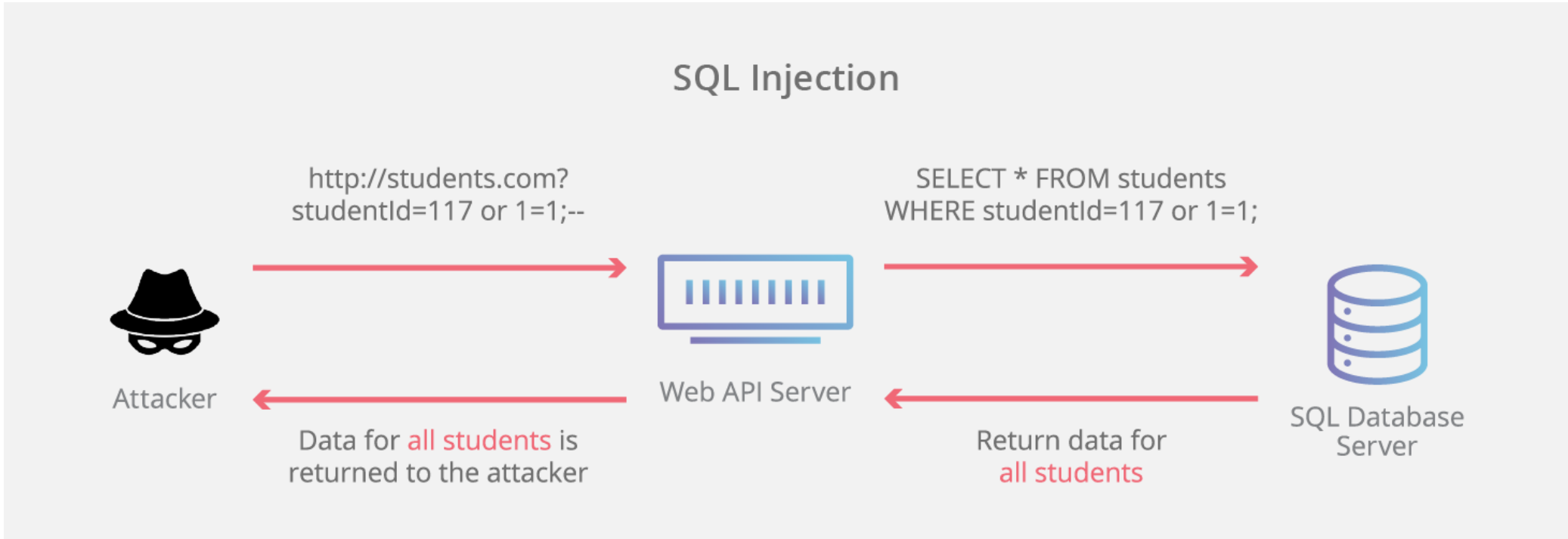
Conexão com banco de dados Mysql

SQL INJECTION

Sql injection

- É uma técnica de ataque que envolve a manipulação do código SQL.
- SQL Injection é uma classe de ataque onde o invasor pode inserir ou manipular consultas criadas pela aplicação, que são enviadas diretamente para o banco de dados relacional.
- Por que o SQL Injection funciona?
  - A aplicação aceita dados fornecidos pelo usuário;
    - Você pede para digitar o nome, mas quem garante que ele não digite código sql de forma maliciosa?


SQL Injection
http://students.com?
studentId=117 or 1=1;--
SELECT * FROM students
WHERE studentId=117 or 1=1;
Attacker
Web API Server
SQL Database
Server
Data for all students is
returned to the attacker
Return data for
all students

Sql injection na prática.

- Imagine o seguinte algoritmo:

$nome = $_GET[‘nome’];
$sql = “select * from Cliente Where nome = ‘$nome’”;

Se o usuário usar digitar o valor da variável $nome for igual a:Helio temos:
$sql = select * from Cliente Where nome = ‘helio’

Sql injection na prática.

- O problema é quando o usuário não digita o nome;
- Imagine que no lugar do nome o usuário digite: `' OR 1=1; #'`

$nome = $_GET[‘nome’];
$sql = “select * from Cliente Where nome = ‘$nome’”;

- Teríamos algo assim:

```
SELECT * FROM clientes WHERE nome ='' OR 1=1; #’
```

- *Depois do # é comentário, e serão apresentados todos os clientes.*
- *Esse é um exemplo simples, mas o usuário pode criar instruções que podem potencialmente excluir o banco de dados.*

Exemplo tabela usuarios

```
CREATE TABLE usuarios (
   id INT AUTO_INCREMENT PRIMARY KEY,
   nome VARCHAR(100),
   email VARCHAR(100)
);
```

```
INSERT INTO usuarios (nome,email) VALUES
('Carlos','carlos@email.com'),
('Ana','ana@email.com'),
('Marcos','marcos@email.com');
```

```php
class Database {
    public static function getConnection() {

        $host = "localhost";
        $banco = "empresa";
        $usuario = "root";
        $senha = "";

        try {

            $conn = new PDO(
                "mysql:host=$host;dbname=$banco;charset=utf8",
                $usuario,
                $senha
            );

            $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

            return $conn;

        } catch (PDOException $e) {
            throw new Exception("Erro ao conectar ao banco de dados", 0, $e);
        }
    }
}
```

```php
$app->get(
    '/teste',
    function (Request $request, Response $response): ResponseInterface {

        try {

            $conn = Database::getConnection();

            $response->getBody()->write("Conexão realizada com sucesso!");


        } catch (Exception $e) {

            $response->getBody()->write("Erro ao conectar: " . $e->getMessage());
        }


        return $response;
    }
);
```

Recupera uma conexão com banco de dados

```php
$app->get(
    '/usuarios',
    function (Request $request, Response $response): ResponseInterface {
        try {
            $conn = Database::getConnection();
            $sql = "SELECT * FROM usuarios";
            $stmt = $conn->query($sql);
            $usuarios = $stmt->fetchAll(PDO::FETCH_ASSOC);
            $json = json_encode($usuarios);
            $response->getBody()->write($json);
            return $response->withHeader('Content-Type', 'application/json');

        } catch (Exception $e) {

            $erro = [
                "erro" => "Falha ao buscar usuários",
                "mensagem" => $e->getMessage()
            ];
            $response->getBody()->write(json_encode($erro));
            return $response
                ->withHeader('Content-Type', 'application/json')
                ->withStatus(500);
        }
    }
);
```

Converte o resultado para um array associativo em php

$usuarios é um vetor associativo

Try tenta fazer, catch é executado caso aconteça algum erro

```php
$app->get(
    '/usuarios/{id}',
    function (Request $request, Response $response, $args): ResponseInterface {
        try {
            $conn = Database::getConnection();
            $sql = "SELECT * FROM usuarios WHERE id = :id";
            $stmt = $conn->prepare($sql);
            $stmt->execute([
                ':id' => $args['id']
            ]);
            $usuario = $stmt->fetch(PDO::FETCH_ASSOC);
            if (!$usuario) {
                $erro = ["erro" => "Usuário não encontrado"];
                $response->getBody()->write(json_encode($erro));
                return $response
                    ->withHeader('Content-Type', 'application/json')
                    ->withStatus(404);
            }
            $response->getBody()->write(json_encode($usuario));
            return $response->withHeader('Content-Type', 'application/json');
        } catch (Exception $e) {
            $erro = ["erro" => "Erro ao buscar usuário"];
            $response->getBody()->write(json_encode($erro));
            return $response
                ->withHeader('Content-Type', 'application/json')
                ->withStatus(500);
        }
    }
);
```

Converte o resultado para um array associativo em php

$usuarios é um vetor associativo

Try tenta fazer, catch é executado caso aconteça algum erro

```php
$app->post(
    '/usuarios',
    function (Request $request, Response $response): ResponseInterface {
        try {
            $conn = Database::getConnection();
            $nome = "helio";
            $email = "helioesperidiao@gmail.com";
            $sql = "INSERT INTO usuarios (nome,email)
                    VALUES (:nome,:email)";
            $stmt = $conn->prepare($sql);
            $stmt->execute([
                ':nome' => $nome ,
                ':email' => $email
            ]);
            $resposta = ["mensagem" => "Usuário inserido"];
            $response->getBody()->write(json_encode($resposta));
            return $response
                ->withHeader('Content-Type', 'application/json')
                ->withStatus(201);
        } catch (Exception $e) {
            $erro = ["erro" => "Erro ao inserir usuário"];
            $response->getBody()->write(json_encode($erro));
            return $response
                ->withHeader('Content-Type', 'application/json')
                ->withStatus(500);
        }
    }
);
```

Substitui os dados pelas variáveis e executa o sql

try tenta fazer, catch é executado caso aconteça algum erro

```php
$app->put(
    '/usuarios/{id}',
    function (Request $request, Response $response, $args): ResponseInterface {
        try {
            $conn = Database::getConnection();
            $nome = "helio";
            $email = "helioesperidiao@gmail.com";
            $sql = "UPDATE usuários SET nome = :nome, email = :email WHERE id = :id";
            $stmt = $conn->prepare($sql);
            $stmt->execute([
                ':nome' => $nome,
                ':email' => $email,
                ':id' => $args['id']
            ]);
            $resposta = ["mensagem" => "Usuário atualizado"];
            $response->getBody()->write(json_encode($resposta));
            return $response
                ->withHeader('Content-Type', 'application/json')
                ->withStatus(200);
        } catch (Exception $e) {
            $erro = ["erro" => "Erro ao atualizar usuário"];
            $response->getBody()->write(json_encode($erro));
            return $response
                ->withHeader('Content-Type', 'application/json')
                ->withStatus(500);
        }
    }
);
```