SEGURANÇA WEB

Ataques e Vulnerabilidades

Uma vulnerabilidade é um vazio ou fraqueza numa aplicação, o qual pode ser uma falha de desenho ou de implementação, que permite a um atacante causar danos aos intervenientes de uma aplicação

Ataques são as técnicas que os atacantes utilizam para explorar as vulnerabilidades em aplicações

SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

- Open Web Application Security Project (OWASP)
  - O projeto OWASP representa um consenso alargado acerca das falhas mais críticas das aplicações web
  - Os membros do projeto incluem uma variedade de peritos de todo o mundo que partilham a sua experiência para produzir uma lista

SEBASTIAO@DI.UBI.PT
FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

OWASP Top 10 (2013)

1. Injeção
2. Autenticação Quebrada ou Gestão de Sessão
3. Cross-Site Scripting (XSS)
4. Controlo de acesso pouco restrito
5. Má-configuração de servidores
6. Exposição de dados sensíveis
7. Proteção insuficiente a ataques
8. Cross-Site Request Forgery (CSRF)
9. Utilização de componentes com vulnerabilidades conhecidas
10. Utilização de APIs sub-protegidas

Source: https://www.owasp.org/index.php/Category:OWASP_Top_Ten_2013_Project

SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

OWASP Top 10 (2017)

1. Injeção
2. Autenticação Quebrada
3. Exposição de dados sensíveis
4. Entidades XML Externas (XXE)
5. Controlo de acesso quebrado
6. Má-configuração de servidores
7. Código cruzado de sites - Cross-Site Scripting (XSS)
8. Deserialização insegura
9. Utilização de componentes com vulnerabilidades conhecidas
10. Monitorização e registo insuficientes

Source: https://www.owasp.org/index.php/Category:OWASP_Top_Ten_2017_Project

SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

- Impacto na segurança
  - Fraude
  - Exposição legal
  - Perdas financeiras
  - Furto de propriedade intelectual
  - Comprometimento da reputação de marcas
  - Extorsão

SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

- Ataque de caminho transversal
  - Usando os caracteres .. e / para ganhar acesso a ficheiros e diretórios que não são supostos serem acedidos

    `http://www.example.com/../../foo.txt`

  - Os servidores normalmente estão bem protegidos contra estes tipos de ataque, mas a aplicação também pode ser alvo:

    `http://www.example.com/page.php?page=../../foo.txt`

    `http://www.example.com/viewimage.php?path=viewimage.php`

SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

Ataque de caminho transversal- Prevenção

```
http://www.example.com/index.php?page=news
```

Substituir:

```
include('header.php');
include($_GET['page']);
include('footer.php');
```

Por:

```
include('header.php');
  if ($page == 'news')
  include('news.php');
if ($page == 'login')
include('login.php');
include('footer.php');
```


SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

- Injeção SQL
  - Inserção de um comando SQL pelos dados e entrada do cliente para a aplicação
  - Os ataques de injeção SQL permitem os atacantes:
    - Roubar identidade
    - Adulterar dados existentes
    - Permitir a divulgação total de todos os dados do sistema
    - Tornar-se administradores da base de dados do servidor

SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

Injeção SQL – Roubar identidade

```
// verifies if username e password are correct
$conn->query("SELECT * FROM users WHERE username = '" . $username . "' AND
password ='" . $password . "'");
http://www.example.com/login.php?username=johndoe&password=' OR 1 = 1; --
SELECT * users WHERE username = 'johndoe' AND password = '' OR 1 = 1; --'
```


SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

Injeção SQL – Divulgar dados

```
// $username has the name of the logged in user
$conn->query("SELECT * FROM items WHERE owner = '" . $username . "'");
```

Create account with username: johndoe' OR 1 = 1

```
SELECT * items WHERE owner= 'johndoe' OR 1 = 1; --'
```


SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

Injeção SQL – Ganhar privilégios

```
// searches for specific item
$conn->query("SELECT * FROM items WHERE title= '" . $title . "'");
http://www.example.com/search.php?title='; INSERT INTO users VALUES
('johndoe', 'password', true); --
SELECT * items WHERE title= '' ; INSERT INTO users VALUES
('johndoe', 'password', true); --'
```


SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

Injeção SQL – Prevenção

- Utilização de comandos pré-preparados (consultas parametrizadas)
- Utilização de Stored Procedures
- Filtrar todas as entradas do utilizador

```
$stmt = $conn->prepare('SELECT * FROM items WHERE title = ?');
$stmt->execute(array($title));
$items = $stmt->fetchAll();
```


SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

- Bloqueio de conta
  - A aplicação web contém um mecanismo de proteção, mas o mecanismo é demasiado restritivo e pode ser despoletado facilmente. Estes mecanismos são utilizados contra ataques de força bruta.
  - Esta técnica permite aos atacantes negar o servido a utilizadores legítimos, causando o bloqueio nas suas contas
    - Exemplo: tentativas sucessivas de autenticação errada no ebay com o nome de utilizador daquele que está a ganhar o leilão, deixando-o bloqueado e fazendo depois licitações à vontade

SEBASTIAO@DI.UBI.PT
FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

- Bloqueio de conta - Prevenção
  - Implementar mecanismos melhores de autenticação, como aqueles que têm em conta o endereço IP de uma conta em adição ao seu nome de utilizador
  - Considerar alternativas a um bloqueio de conta pode ser também efetivo, como por exemplo apresentar ao utilizador um puzzle para resolver

SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

- Cross-site Scripting (XSS)
  - Ataques XSS são um tipo de injeção, na qual o código é injetado em sítios originalmente fiáveis
  - Normalmente são categorizados em três tipos: persistente, refletido e baseado em DOM

SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

- Cross-site Scripting (XSS)
  - O XSS persistente geralmente acontece quando os dados de utilizador são armazenados no servidor alvo, como uma base de dados, numa mensagem de fórum, num registo de visitante, comentário, etc.
  - O XSS refletido acontece quando os dados do utilizador são imediatamente devolvidos pela aplicação web numa mensagem de erro, como num resultado de pesquisa ou outras respostas
  - Os XSS baseados em DOM normalmente ocorrem quando o fluxo de dados nunca deixa o navegador
    - Por exemplo, o código malicioso pode estar no URL e a página insere-o no DOM tornando-o malicioso

SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

Cross-site Scripting (XSS) - Persistente

```
<?php
  $stmt = $conn->prepare("INSERT INTO comment VALUES (NULL, ?, ?, ?)");
  $stmt->execute(array($_POST['post_id'], $_POST['text'], $_SESSION['username']));
?>
<?php
  $stmt = $conn->prepare("SELECT * FROM comment WHERE post_id = ?");
  $stmt->execute(array($_POST['post_id']));
  $comments = $stmt->fetchAll();
  foreach($comments as $comment) {
    echo "<div class=\"comment\">" . $comment['text'] . "</div>"; }
?>
```

O texto comentado pode ter código malicioso que fica guardado permanentemente na base de dados e é mostrado a todos

SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

Cross-site Scripting (XSS) - Refletido

```
<?php
  echo "You searched for: " . $_GET["query"];
// List search results
?>

http://example.com/search.php?query=<script>alert("hacked")</script>
```


SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

Cross-site Scripting (XSS) – baseado em DOM

```
var pos=document.URL.indexOf("language=") + 8;
var language = document.URL.substring(pos,document.URL.length));
$("#language").html(language);
```

```
http://www.example.com/index.php?language=<script>alert("hacked")</script>
```

Ou

```
http://www.example.com/index.php#language=<script>alert("hacked")</script>
```

O identificador não é enviado ao servidor, tornando impossível detetar o ataque remotamente

SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

Cross-site Scripting (XSS) - Prevenção

- Nunca colocar dados não fiáveis:
  - Diretamente num código
  - Dentro de um comentário HTML
  - No nome de um atributo
  - No nome de uma etiqueta
  - Diretamente no CSS
  - Dentro de valores inseguros de atributos

SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

- Cross-site Scripting (XSS) - Prevenção
  - Algumas regras para utilizar dados inseguros :
    - Texto dentro do corpo HTML: codificar entidades HTML
    - HTML dentro do corpo HTML Body: filtrar HTML
    - Atributos HTML seguros: codificar entidades HTML
    - Parâmetros GET: codificação de URL
  - Mas isto pode não ser suficiente

SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

Cross-site Scripting (XSS) - Prevenção

Validar

Se a entrada contém caracteres inesperados, rejeitá-la :

```
if ( !preg_match ("/^[a-zA-Z\s]+$/",
$_GET['name'])) {
// ERROR: Name can only contain letters e spaces
}
```


SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

Cross-site Scripting (XSS) - Prevenção

Codificar

Codificação de entidades - Codificar caracteres especiais tais como entidades HTML:

```
<script>alert("hacked")</script>
&#x3C;script&#x3E;alert(&#x22;hacked&#x22;)&#x3C;/script&#x3E;
```

Codificação de URL - Codificar URLs:

```
http://example.com/search.php?q=<script>alert("h")</script>
http://example.com/search.php?q%3D%3Cscript%3Ealert(%22h%22)%3C%2Fscript%3E
```


SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

Cross-site Scripting (XSS) – Prevenção em Javascript

Filtrar o HTML antes de inserir dados inseguros dentro de um elemento HTML

```
var entityMap = { "&": "&amp;", "<": "&lt;", ">": "&gt;", '"': '&quot;',
"'": '&#39;', "/": '&#x2F;'};

function escapeHtml(string)
{
 return String(string).replace(/[&<>"'\/]/g, function (s) {
   return entityMap[s];
});
}
```


SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

Cross-site Scripting (XSS) – Prevenção em PHP

- Utilizar as funções strip_tags e htmlentities pode não ser suficiente
- Adicionalmente, utilizar codificadores cientes do contexto
  - Por exemplo: OWASP ESAPI para PHP
- Se alguns dados devem ser autorizados, pode-se utilizar, por exemplo, um purificador HTML

SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

- Cross-site Request Forgery (CSRF)
  - A aplicação permite ao utilizador submeter um pedido de alteração de estado que não inclui nada secreto

```
http://example.com/transferFunds.php?amount=1500&destination=4673243243
```

  - O atacante constrói um pedido que irá transferir dinheiro da conta da vítima para a conta do atacante, e depois inclui este ataque num pedido de imagem armazenado em vários sítios sob controlo do atacante:

```
<img src =
"http://example.com/transferFunds.php?amount=1500&destination=4673243243"
width="0" height="0" />
```


SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

Cross-site Request Forgery (CSRF) - Prevenção

- Gerar uma token aleatória em cada sessão
- Armazenar esta token como variável de sessão
- Enviar esta token como parte de cada pedido
- Verificar se o token está correto em cada página

SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática

Cross-site Request Forgery (CSRF) - Prevenção

```
session_start();
if (!isset($_SESSION['csrf_token'])) {
  $_SESSION['csrf']) = generate_random_token();
}
<form action="transferfunds.php">
  <input type="hidden" name="csrf" value="<?=$_SESSION['csrf_token'])?>">
</form>
session_start();
if ($_SESSION['csrf'] !== $_POST['csrf']) {
  // ERROR: Request does not appear to be legitimate
}
```


SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

Man in the Middle Attack (MITM)

- Interceta uma comunicação entre dois sistemas
- Utilizando técnicas diferentes, o atacante divide a ligação TCP original em 2 ligações novas, uma entre o cliente e o atacante e outra entre o atacante e o servidor
- Sendo intercetada uma vez, o atacante atua como uma proxy, sendo capaz de ler, inserir e modificar os dados da comunicação

SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

- Man in the Middle Attack (MITM) – Prevenção
  - Estabelecer uma ligação segura
  - Encriptar a transmissão
    - A encriptação pode não ser suficientes, porque cada método de encriptação exige uma transmissão de informação adicional sobre um canal seguro
  - A solução é utilizar chaves públicas que tenham sido assinadas por uma autoridade de certificação (CA)

SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

- Man in the Middle Attack (MITM) – Prevenção
  - Criptografia de chave pública
    - Também conhecida como criptografia assimétrica, é uma classe de algoritmos criptográficos que requer duas chaves distintas, uma das quais é privada e outra das quais é pública
    - Se o remetente assina a mensagem com a sua chave pública, qualquer recetor pode verificar que a mensagem foi enviada por ele
    - Se o remetente encripta a mensagem com a chave privada, apenas o recetor com a chave privada pode ler essa mensagem

SEBASTIAO@DI.UBI.PT
FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

- Man in the Middle Attack (MITM) – Prevenção
  - Certificados
    - Os navegadores web confiam em sítios baseando-se em CA que vêm pré-instaladas (Verisign, Microsoft,...)
    - O utilizador confia na CA que dá a sua anuência aos seus sítios legítimos
    - Se um sítio fornece um certificado válido, significa que foi assinado por uma autoridade confiável
    - O certificado identifica positivamente o sítio
    - O utilizador confia que o nível de encriptação do protocolo (TLS/SSL) é suficientemente seguro contra interceções

SEBASTIAO@DI.UBI.PT
FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

- Boas práticas de palavras-chave
  - Utilizar sal
    - O sal deve ser gerado com um gerador de números pseudoaleatório criptograficamente seguro (Cryptographically Secure Pseudo-Random Number Generator - CSPRNG)
    - O sal necessita de ser único por utilizador
    - O sal necessita de ser grande

SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

- Boas práticas de palavras-chave
  - Geração
    - Adicionar o sal como prefixo à palavra-chave e fazer hash com uma função criptográfica padrão como a bcrypt
    - Guardar ambos o sal e o hash no registo do utilizador na base de dados

SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

- Boas práticas de palavras-chave
  - Validação
    - Obter o sal do utilizador e hash da base de dados
    - Adicionar o sal como prefixo da palavra-passe indicada pelo utilizador e utilizar a função de hash
    - Comparar o resultado final com o guardado na base de dados

SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

Palavras-passe em PHP

- O método recomendado para fazer hash e validar palavras-passe em PHP é utilizar as funções password-hash e password-verify

```
string password_hash (string $password , integer $algo [,array $options ])
boolean password_verify (string $password , string $hash)
```

- Estas funções geram o seu próprio sal
- A função hash retorna o algoritmo utilizado, custo e sal como parte do hash
- Isto permite à função verificar hash sem ter necessidade de armazenamento adicional para o sal ou o algoritmo

SEBASTIAO@DI.UBI.PT
FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

Palavras-passe em PHP

```
<?php
  $options = ['cost' => 12];
  $stmt = $db->prepare(INSERT INTO users VALUES (?, ?))';
  $stmt->execute(array($username, password_hash($password, PASSWORD_DEFAULT,
$options)));
<?php
  $stmt = $db->prepare('SELECT * FROM users WHERE username = ?');
  $stmt->execute(array($username));
  $user = $stmt->fetch();
  if ($user !== false && password_verify($password, $user['password'], )) {
    $_SESSION['username'] = $username;
  }
```


SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

Palavras-passe em PHP

- Garantir que os nomes de utilizador/id utilizador não são distinguíveis entre maiúsculas/minúsculas (mesmo e-mails)
- Implementar controlos de força de palavra-passe
- Não aplicar limites de tamanho ou de caracteres no registo de credenciais
- Desenhar armazenagem de palavras-passe de modo a ser robusta num eventual corrompimento

SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

Fixação de Sessão

Este ataque consiste em obter um id de sessão válido, induzindo o utilizador a autenticar-se com aquele id de sessão e depois roubar essa sessão validada

```
http://example.com/<script>document.cookie="sessionid=abcd";</script>
http://example.com/<meta http-equiv=Set-Cookie
content="sessionid=abcd">
```


SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

Fixação de Sessão - Prevenção

Regenerar o id de sessão em cada pedido (ou a cada *n* pedidos):

```
<?php
  session_start();
  session_regenerate_id(true);
```

Destruir a sessão se a origem é suspeita

```
<?php
  if (strpos($_SERVER['HTTP_REFERER'], 'http://example.com/') !== 0)
    session_destroy();
```


SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

- Roubo de Sessão
  - Ganhar controlo da sessão do utilizador, roubando o id da sessão
    - Cross-site scripting
    - Session Sniffing
    - Man-in-the-middle

SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

Roubo de Sessão - Prevenção

- Medidas anti XSS
- Atributo HttpsOnly
- HTTPS

SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

Denial of Service (DoS)

A negação de serviço é uma tentativa de tornar os recursos de máquina ou de rede indisponíveis para os utilizadores alvo

Existem várias maneiras de tornar o serviço indisponível para utilizadores legítimos, manipulando os pacotes de rede, programação, lógica ou exploração de vulnerabilidades de recursos, entre outras

SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática

SEGURANÇA WEB

Denial of Service (DoS) - Prevenção

- Bloquear os IPs que fazem muitos e sucessivos pedidos
- Instalar uma proxy invertida
- Utilizar cookies SYN

SEBASTIAO@DI.UBI.PT

FACULDADE ENGENHARIA
Departamento de Informática