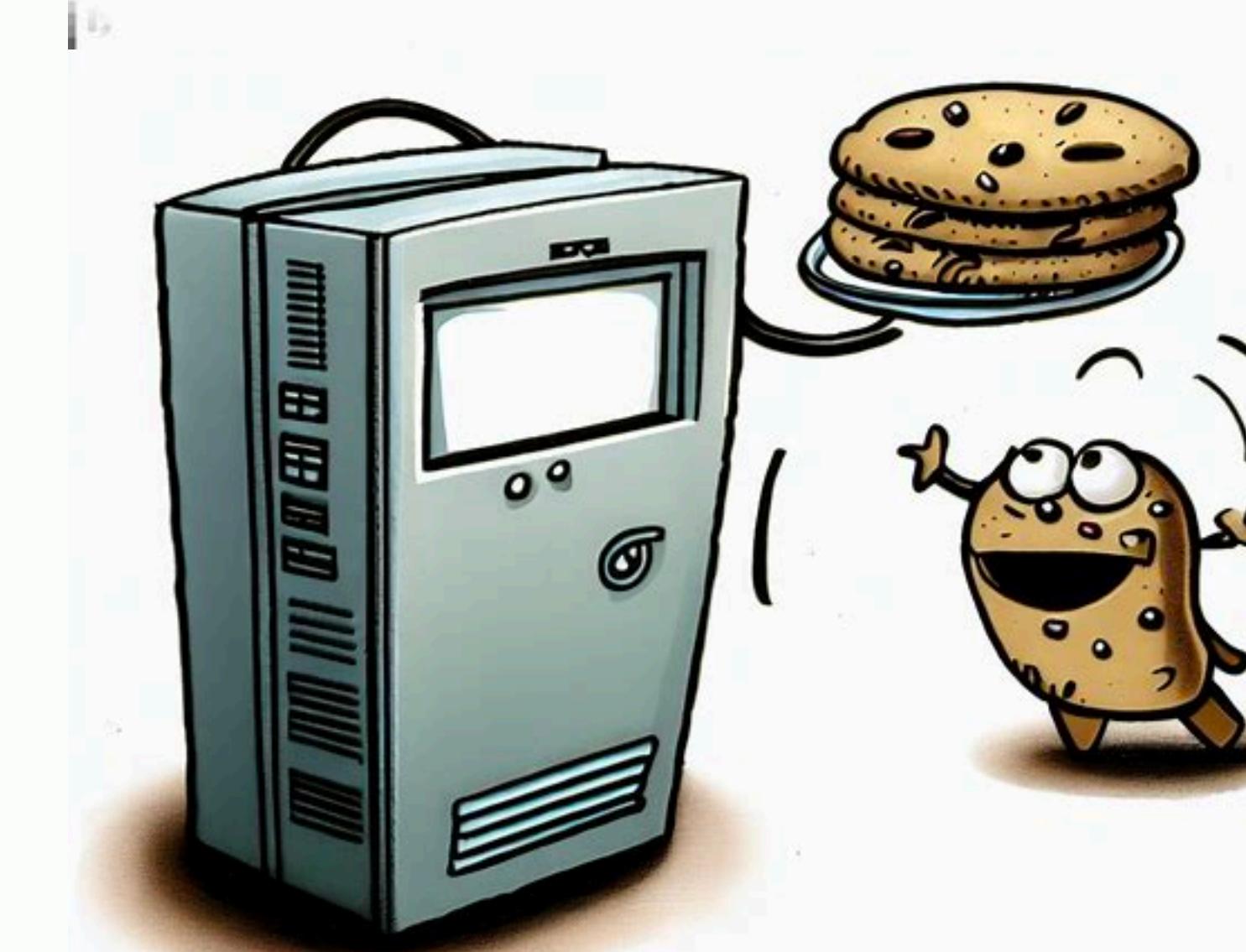
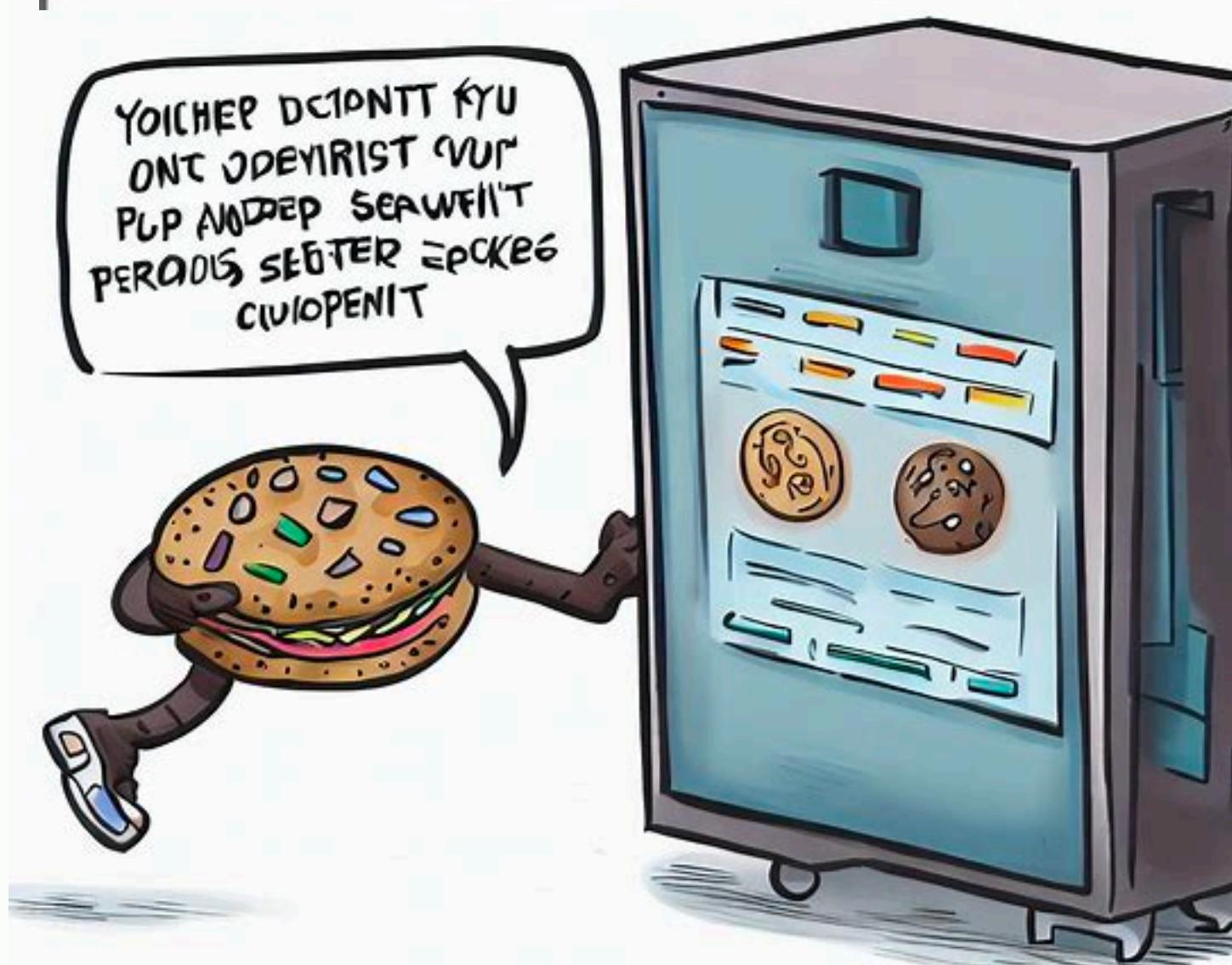
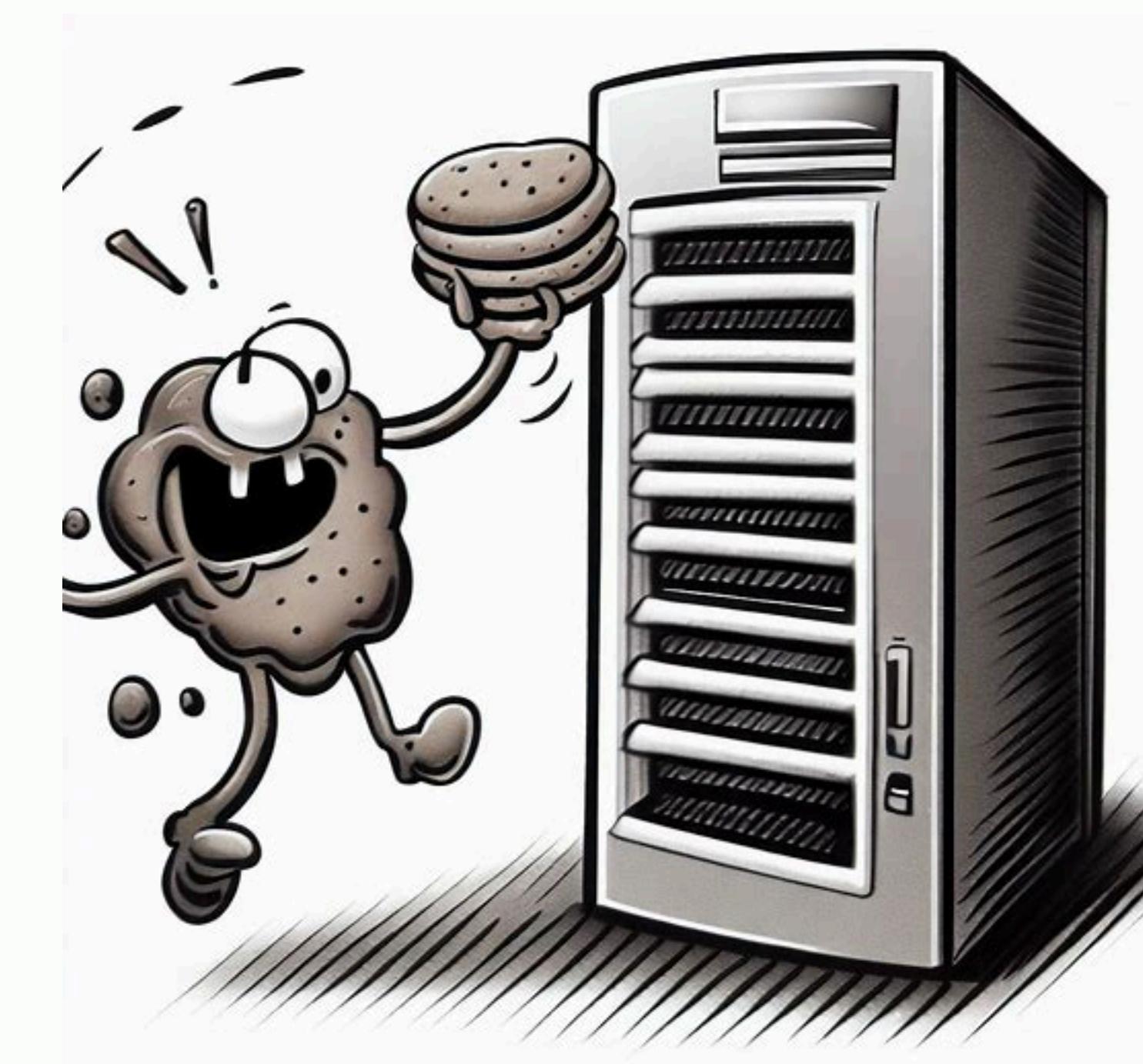
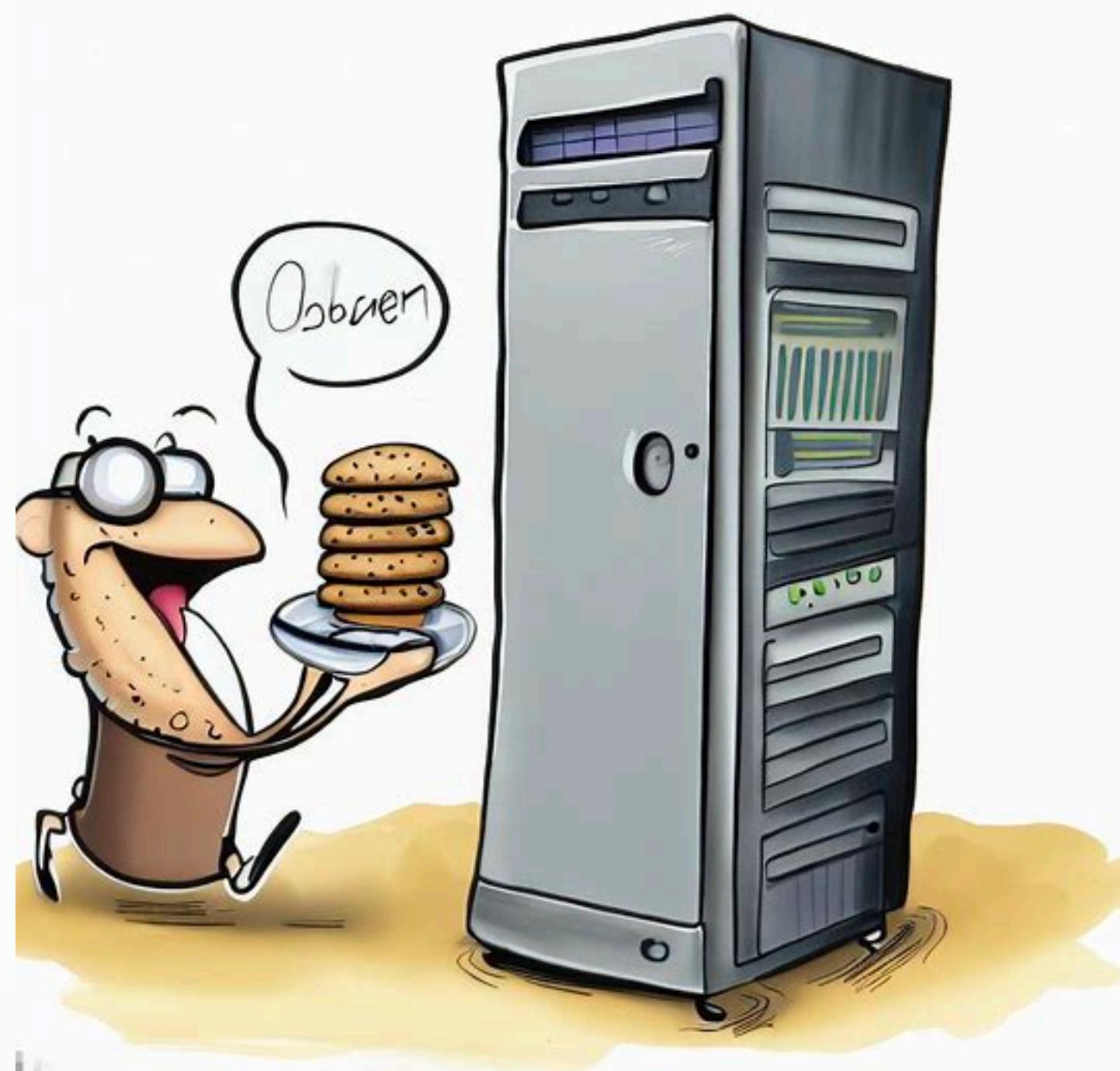


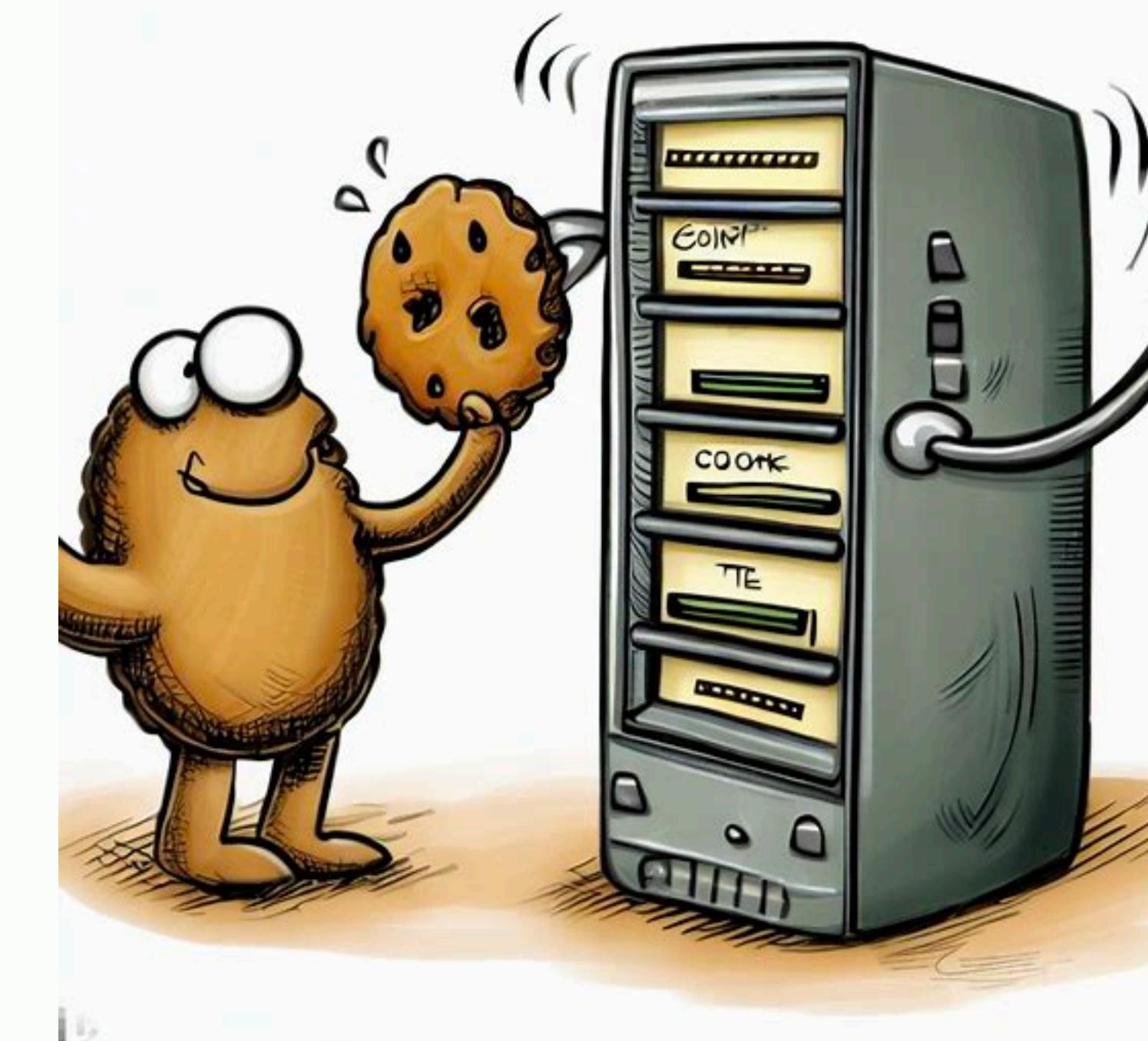
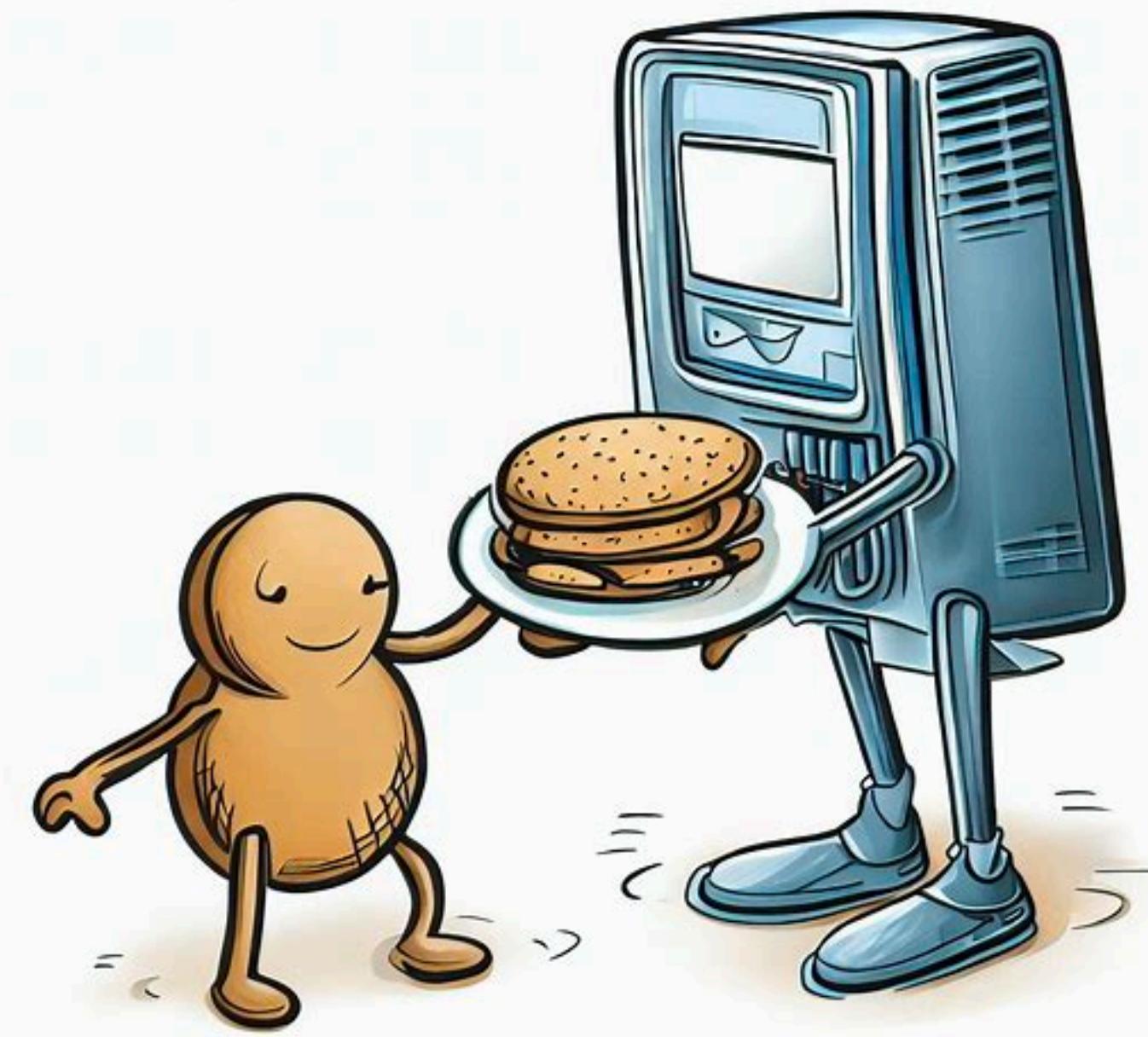
Cookies og sessions

DOB





"a humorous cartoon drawing of a webserver sending a cookie to a browser"



"a humorous cartoon drawing of a web browser receiving a cookie from a web server"



"a futuristic dystopic photorealistic image of an anthropomorphic web server eating lots of cookies"

HTTP er stateless

- HTTP er designet til at være en stateless protokol, hvilket betyder, at hver anmodning behandles uafhængigt af tidligere anmodninger.
- Når en server modtager en HTTP-anmodning, behandles den udelukkende baseret på de oplysninger, der er indeholdt i anmodningen.

Nyt request, ...Who dis?

- På grund af HTTP's stateless natur mangler protokollen indbyggede mekanismer til at gemme tilstand (state) mellem anmodninger.
- Serveren har ingen indbygget måde at genkende en bruger eller huske tidlige interaktioner uden yderligere hjælpemidler.

Cookies, nom nom nom

- Cookies blev introduceret (ca. 1995) som en løsning på HTTP's stateless problem.
- Ved hjælp af cookies kan serveren gemme og hente oplysninger på klienten, hvilket giver mulighed for at tilføre en form for state til HTTP-anmodninger.

cookies er bare sma tekstfiler på din computer

- Ved at sende og modtage cookies kan serveren gemme brugerspecifikke data, f.eks. brugeridentifikatorer, præferencer eller indkøbskurvoplysninger.

I know what you did last summer

- Cookies giver mulighed for at opretholde tilstand flere anmodninger fra samme klient.
- Serveren kan bruge oplysninger gemt i cookies til at genkende brugere, huske indstillinger og tilpasse indholdet baseret på tidligere handlinger.

Cookie quiz

Hvad er en HTTP-cookie?

- A. En lille tekstfil, der gemmes på klientens computer
- B. Et serverside-scriptingssprog
- C. En type database, der bruges til at gemme webdata
- D. En sikkerhedsprotokol til datakryptering

Hvad er en HTTP-cookie?

- A. En lille tekstfil, der gemmes
på klientens computer
- B. Et serverside-scriptingssprog
- C. En type database, der bruges
til at gemme webdata
- D. En sikkerhedsprotokol til
datakryptering

Hvordan bruges HTTP-cookies primært?

- A. Til at gemme og hente
brugerspecifikke data
- B. Til at forhindre cross-site
scripting-angreb
- C. Til at kryptere følsomme
oplysninger
- D. Til at forbedre websitets
ydeevne

Hvordan bruges HTTP-cookies primært?

- A. Til at gemme og hente
brugerspecifikke data
- B. Til at forhindre cross-site
scripting-angreb
- C. Til at kryptere følsomme
oplysninger
- D. Til at forbedre websitets
ydeevne

Hvilket af følgende er IKKE en almindelig anvendelse af cookies?

- A. Huske brugerloginoplysninger
- B. Spore brugeradfærd til analyseformål
- C. Gemme serversidekode i browseren
- D. Personalisere websiteindhold baseret på brugerpræferencer

Hvilket af følgende er IKKE en almindelig anvendelse af cookies?

- A. Huske brugerloginoplysninger
- B. Spore brugeradfærd til analyseformål
- C. Gemme serversidekode i browseren
- D. Personalisere websiteindhold baseret på brugerpræferencer

Hvilken HTTP-header bruges til at sætte en cookie fra serveren?

- A. Set-Cookie
- B. Cookie-Set
- C. Cookie
- D. Set-Cookie-Header

Hvilken HTTP-header bruges til at sætte en cookie fra serveren?

- A. Set-Cookie
- B. Cookie-Set
- C. Cookie
- D. Set-Cookie-Header

Hvad er en session cookie?

- A. En cookie, der vedvarer, selv efter at browseren er lukket
- B. En cookie, der gemmer følsomme brugeroplysninger
- C. En cookie, der udløber, når sessionen slutter
- D. En cookie, der kun kan tilgås af serveren

Hvad er en session cookie?

- A. En cookie, der vedvarer, selv efter at browseren er lukket
- B. En cookie, der gemmer følsomme brugeroplysninger
- C. En cookie, der udløber, når sessionen slutter
- D. En cookie, der kun kan tilgås af serveren



Sende cookies frem og tilbage



After receiving an HTTP request, a server can send one or more **Set-Cookie** headers with the response. The browser usually stores the cookie and sends it with requests made to the same server inside a **Cookie** HTTP header.

- ["Using HTTP cookies", MDN web docs](#)



Sætte og modtage cookies

Serveren fortæller klienten at den skal gemme et par cookies med **Set-Cookie** headeren i et

Response:

```
HTTP/2.0 200 OK
```

```
Content-Type: text/html
```

```
Set-Cookie: yummy_cookie=choco
```

```
Set-Cookie: tasty_cookie=strawberry
```

```
[page content]
```

For hvert efterfølgende request sender browseren så de samme cookies (og alle andre, der måtte være gemt for det domæne og evt sti) tilbage til serveren via **Cookie** headeren i et **Request**:

```
GET /sample_page.html HTTP/2.0
```

```
Host: www.example.org
```

```
Cookie: yummy_cookie=choco; tasty_cookie=strawberry
```



Session cookies vs. persistent cookies

Den første cookie herunder, "theme", kaldens en session cookie fordi den ikke har Expires eller Max-Age attributer. Session cookies slettes af browseren når den lukkes.

Den anden cookie, "sessionToken", er en persistent cookie fordi den indeholder en Expires attribut, som fortæller browseren at den først skal slettes på et senere angivet tidspunkt.

```
HTTP/1.0 200 OK
Content-type: text/html
Set-Cookie: theme=light
Set-Cookie: sessionToken=abc123; Expires=Wed, 20 Apr 2021 20:18:14 GMT
```

Ligesom med **Cache-Control** headeren på et Response, så kan en cookies levetid også sættes med en **Max-Age** attribut, som er en værdi i sekunder, hvorefter cookien skal slettes af browseren.

```
HTTP/1.0 200 OK
Content-type: text/html
Set-Cookie: sessionToken=abc123; Max-Age=3600
```

Cookie-attributter

Attribut	Betydning
Domain	Angiver domænet, som cookien er knyttet til, og begrænser dermed cookiens tilgængelighed til det specifikke domæne eller dets subdomæner.
Path	Bestemmer den URL-sti, hvor cookien er tilgængelig, og begrænser dermed cookiens anvendelse til specifikke stier på domænet.
Expires/Max-Age	Angiver udløbsdatoen for cookien, efter hvilken den ikke længere sendes til serveren.
Secure	Angiver, at en cookie kun skal sendes over en sikker HTTPS-forbindelse, og ikke over usikker HTTP.
HttpOnly	Forhindrer JavaScript-adgang til en cookie, hvilket hjælper med at beskytte mod cross-site scripting (XSS)-angreb.
SameSite	Bestemmer, hvornår en cookie skal sendes til serveren, og begrænser dermed dens tilgængelighed for tredjepartswebsteder.



Sikre cookies over HTTPS



A cookie with the **Secure** attribute is only sent to the server with an encrypted request over the HTTPS protocol. It's never sent with unsecured HTTP (except on localhost), which means man-in-the-middle attackers can't access it easily. Insecure sites (with **http:** in the URL) can't set cookies with the **Secure** attribute.

- "Using HTTP cookies", MDN web docs

```
Set-Cookie: id=a3fWa; Expires=Thu, 21 Oct 2021 07:28:00 GMT; Secure
```



HttpOnly cookies kan ikke læses fra JavaScript



A cookie with the `HttpOnly` attribute is inaccessible to the `JavaScript Document.cookie API`; it's only sent to the server. For example, cookies that persist in server-side sessions don't need to be available to JavaScript and should have the `HttpOnly` attribute. This precaution helps mitigate cross-site scripting (XSS) attacks.

- "Using HTTP cookies", MDN web docs

```
Set-Cookie: id=a3fWa; Expires=Thu, 21 Oct 2021 07:28:00 GMT; Secure; HttpOnly
```

Inspicér cookies i browserens DevTools:

Name	Value	Domain	Path	Expires / Max-Age	Size	HttpOnly	Secure	SameSite
_device_id	56fd76c63887f89afdb7388768b13115	github.com	/	2022-09-02T16:50:16.955Z	42	✓	✓	Lax
_ga	GA1.2.965282267.1523952449	.github.com	/	2022-05-29T08:34:49.000Z	29			
_gh_sess	DlwBbl9LA2liEE07TxqH8ILK1ea%2Bzr0YAS3v2...	github.com	/	Session	360	✓	✓	Lax
_octo	GH1.1.1952086774.1630601416	.github.com	/	2022-09-02T16:50:16.955Z	32		✓	Lax
logged_in	no	.github.com	/	2023-04-18T17:48:14.582Z	11	✓	✓	Lax
tz	Europe%2FCopenhagen	.github.com	/	Session	21		✓	Lax

▼ Request Headers

```
:authority: github.com
:method: GET
:path: /
:scheme: https
accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
accept-encoding: gzip, deflate, br
accept-language: en-GB,en-US;q=0.9,en;q=0.8
cache-control: no-cache
cookie: _ga=GA1.2.965282267.1523952449; _device_id=56fd76c63887f89afdb7388768b13115; _octo=GH1.1.1952086774.1630601416; logged_in=no; _gh_sess=DlwBbl9LA2liEE07TxqH8ILK1ea%2Bzr0YAS3v2t1U9aKeSrmkqFZgKhn9Y1o2j1i9kyDmP2S5wxYAV%2BQICekQt4bjWhtGJIJ03GjmBpW%2Fykd0hsrk040bvaYUxqdrevpfSXuZGshp%2FYAERn4pL4Y8KbWGKxVvl0r46S1Powpyr6UrZ9gvD28KmZP1g1ic0dW%2BCsX4TmpP2YvT7Hg1M9hPzUn9zGaMZhMkNb1%2FW7Gr7D1B%2Bw0PjJd4B2kMAwXY%2BRfdxcZyCXq8naW3eG%2B56Ja9w%3D%3D--QMr9l2zWPIM2h0I%2B--dRL0iBi9VMMc0SP2a6gCIg%3D%3D; tz=Europe%2FCopenhagen
pragma: no-cache
```



Brugeren kan filde med sine egne cookies



When you store information in cookies, keep in mind that all cookie values are visible to, and can be changed by, the end user.

Depending on the application, you may want to use an opaque identifier that the server looks up, or investigate alternative authentication/confidentiality mechanisms such as JSON Web Tokens.

- "Using HTTP cookies", MDN web docs

👉 Signing cookies in Remix