


Internet

CAROL E MATHEUS

Sumário

- 
- 1- O que é Internet e Como funciona
 - 2 - Nomes de Domínio e DNS
 - 3 - URI - URL - URN
 - 4 - Protocolos HTTP, Métodos/Verbos, Cabeçalhos, Corpo e Código de Status
 - 5 - HTTPS e SSL/TS
 - 6 - Servidores Web e Hosting
 - 7 - Navegadores e User Agents
 - 8 - Cookies e Sessions
 - 9 - CDN (Content Delivery Network)

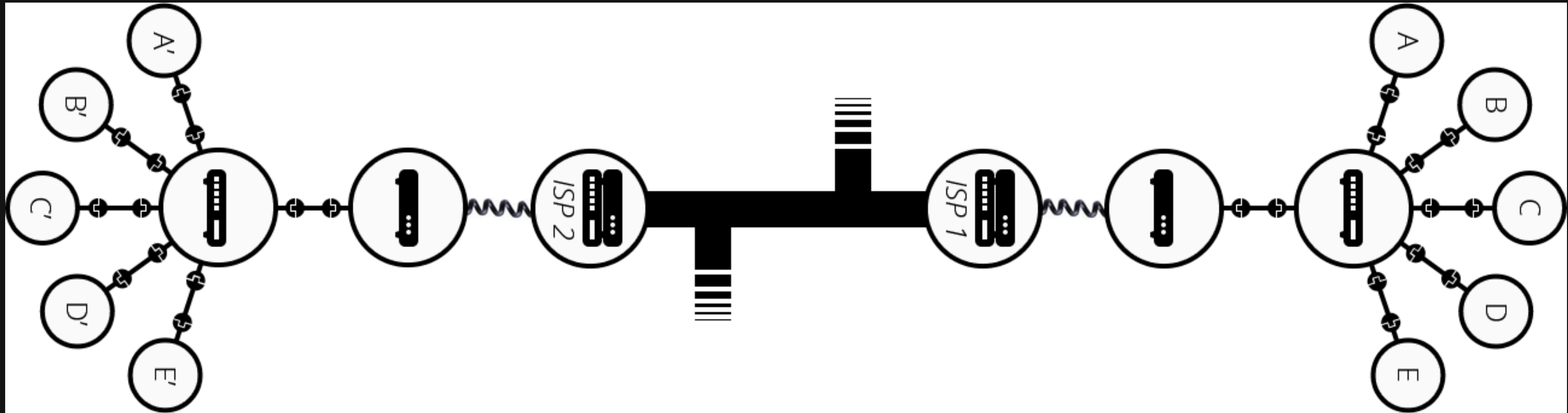
O que é a Internet?

É uma rede global de computadores conectados entre si que se comunicam por meio de um conjunto padronizado de protocolos;

Sua história começou na década de 1960 como um projeto de pesquisa financiado pelo exército dos EUA;

Na década de 1980 evoluiu para uma infraestrutura pública com o apoio de universidades públicas e empresas privadas;

Como a Internet Funciona?



Nomes de Domínio e DNS

Acessamos informações on-line por meio de nomes de domínio, como **google.com**;

No entanto, os navegadores interagem por meio de endereços de Protocolo de Internet (IP);

Já o DNS (Domain Name System) converte os nomes de domínio em endereços de IP para que os navegadores possam carregar os recursos da internet;

Domínio: terra.com IPV4: 208.84.244.50

Domínio: google.com IPV6: 2800:3f0:4001:83b::200e

Estrutura de Nomes de Domínio



TDL - DOMÍNIO DE NÍVEL SUPERIOR

Informa aos usuários o propósito geral do serviço por trás do nome de domínio.

Ex: .com, .net, .org, .edu, .gov.

RÓTULOS (LABELS)

São uma sequência de caracteres de 1 a 63;

Label 1: Chamado de Domínio de Nível Secundário (SLD).

Proprietário de um domínio?

Não é possível ser proprietário de um domínio pra sempre;

Um domínio pode ser comprado por 1 ou mais anos;

Se for renovado o proprietário anterior possui prioridade se comparado a outras pessoas;

Como encontrar um domínio disponível?

Recorrer a um site de um registrador de nome de domínio;

Fornecem um serviço "whois" que informa se um nome de domínio está disponível;

<https://registro.br/tecnologia/ferramentas/whois/>

Como funciona uma solicitação DNS?

1. Digite "domínio do site" na barra de localização do seu navegador;
2. O navegador pergunta ao computador se este reconhece o IP (usando um cache DNS local). Se reconhecer, o nome é traduzido para o IP e o navegador solicitar o conteúdo ao servidor web;
3. Se o computador não souber qual IP está por trás do domínio ele vai perguntar a um servidor DNS, que irá informar o endereço IP correspondente a este domínio;
4. Agora que o computador conhece o endereço IP solicitado, o navegador pode solicitar o conteúdo ao servidor web.

URI - Uniform Resource Identifier

É uma string que identifica um recurso;

Se uma pessoa disser que mora na única casa amarela da sua cidade, ela não está dando instruções sobre como chegar lá. No entanto, essa informação identifica essa casa entre as outras da sua cidade.

Um exemplo de URI é: <https://sig.ifc.edu.br/sigaa/verTelaLogin.do>, que identifica a página de acesso ao sigaa;

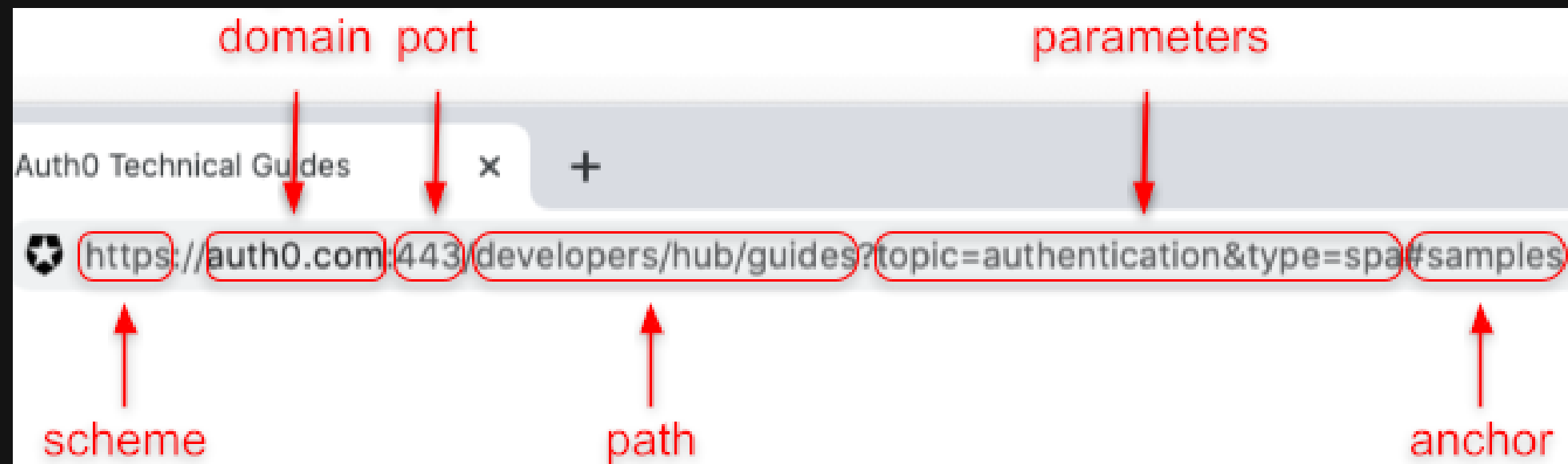
URL - Uniform Resource Locator

Uma URL é uma sequência de caracteres que denota a localização de um determinado recurso na Internet e também pode ser chamada de address;

Uma URL é como se fosse um endereço residencial que contém todas as informações para encontrar uma casa;

Um exemplo de URL é: **<https://www.microsoft.com/>**

Estrutura de uma URL



Scheme: Em uma URL, este é o protocolo que deve ser usado para acessar o recurso.

Domain: Indica o servidor que hospeda o recurso, pode ser um nome de domínio ou um endereço IP.

Port: É a porta do protocolo para a qual enviar a solicitação de acesso ao recurso.

Path: Este é o caminho para o recurso no servidor de hospedagem.

Parameters: São informações extras opcionais fornecidas ao servidor de hospedagem.

Anchor: Representa uma parte específica dentro do recurso. Também é chamada de fragmento .

Protocolos de Rede

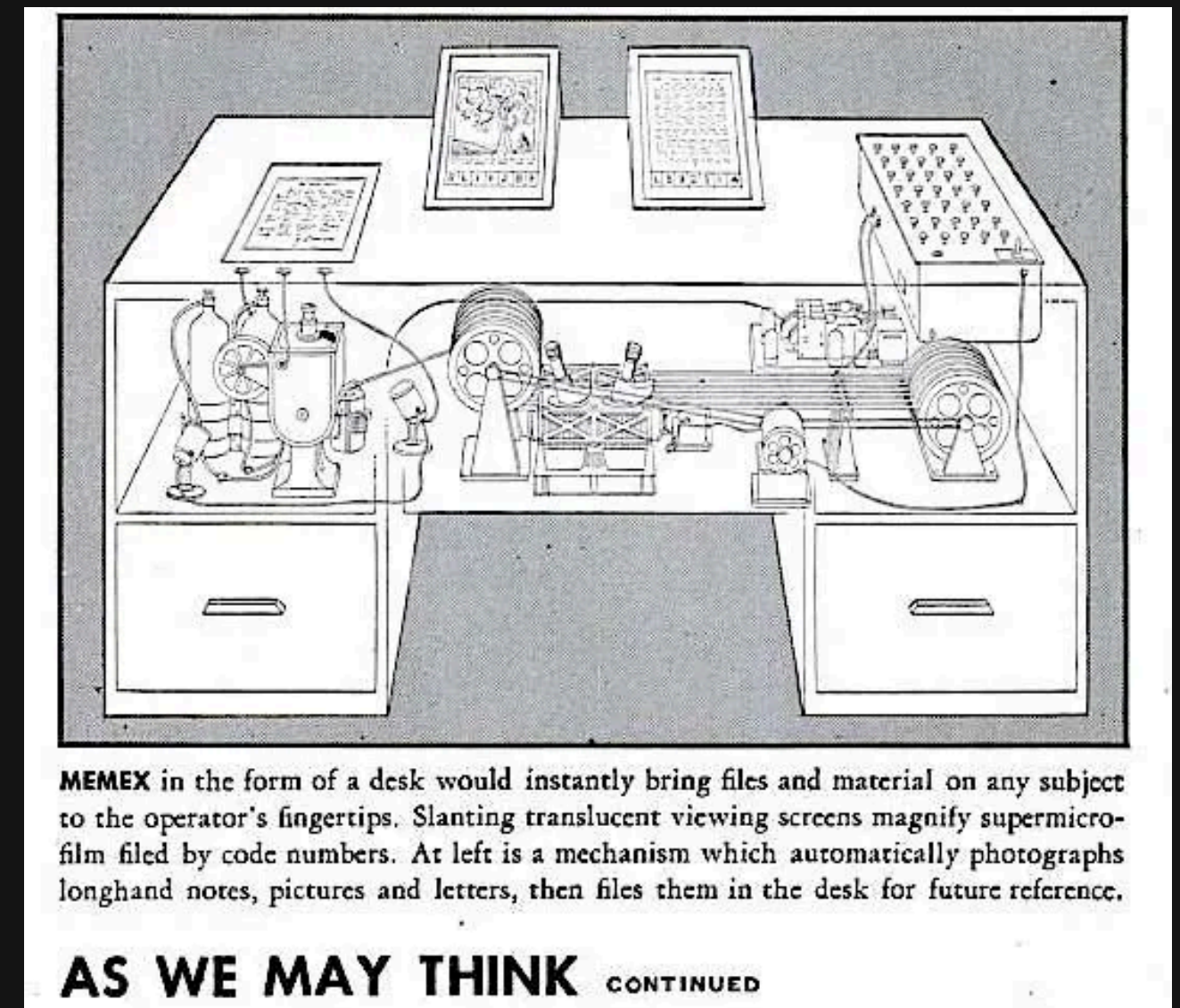
São padrões para realizar determinadas ações e manipular dados para que dois dispositivos em rede possam se comunicar, trocar arquivos, etc.

Exemplos:

TCP/IP, HTTP, HTTPS, FTP, SFTP, SSH, etc

Protocolos HTTP (Hyper Text Transfer Protocol)

- Texto legível sem conexão, usado para transferência de arquivos entre navegadores e servidores
- 1965 idealizado por Ted Nelson inspirado pelo sistema MEMEX
- 1989 Berners-Lee propôs o projeto WorldWideWeb hoje conhecido pela sigla WWW
- 1995 Dave Raggett liderou a primeira versão documentada
- 1996 amplamente aceito pelos desenvolvedores de navegadores
- Janeiro de 1997 lançado oficialmente como o padrão HTTP / 1.1

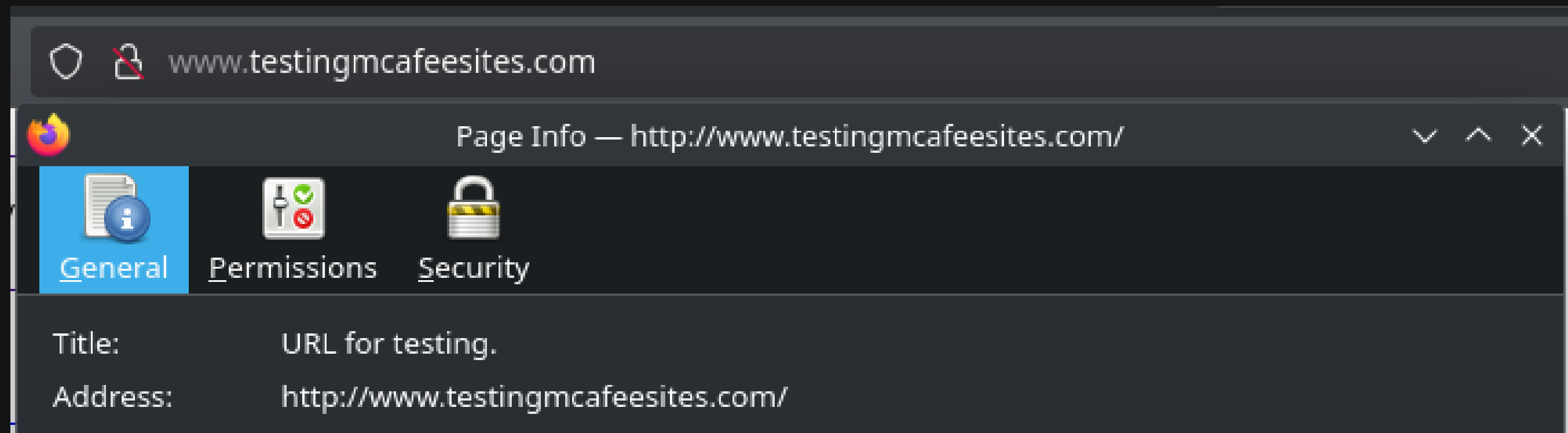


Como funciona?

Baseado em texto sem conexão

Texto nú: senha123

Criptografado MD5: e7d80ffeefa212b7c5c55700e4f7193e



Passo-a-passo

1. Se necessário usar servidores dns para recuperar o endereço IP;
2. Solicitação HTTP é enviada ao servidor da página web;
3. Solicitação recebida e página verificada, caso não exista erro 404 e conexão fechada;
4. Caso existe o servidor web mostrará, navegador requisita elementos necessários para total exibição;
5. Para cada requisição HTTP uma nova conexão;
6. Quanto todos os elementos tiverem sido carregados a página será carregada na janela do navegador {google};

Métodos/Verbos

1. **GET**: Solicita a representação de um recurso específico.
2. **HEAD**: GET exceto que na resposta não vem o corpo da resposta
3. **POST**: É usado para inserir dados em um recurso do servidor, ações em que o servidor alocará novos recursos
4. **PUT**: Quase igual ao POST porém este não necessita que o servidor aloque novos recursos
5. **DELETE**: Remove algum recurso do servidor
6. **CONNECT**: estabelece um túnel para o servidor
7. **OPTIONS**: Descreve as opções de comunicação
8. **TRACE**: executa um teste de chamada loop-back
9. **PATH**: Aplicar modificações parciais em um recurso do servidor

Cabeçalho

Campo de requisição/resposta pode ter parâmetros adicionais;

Chamamos a mensagem toda como cabeçalho genérico, porém nela podemos subdividir em três tipos:

1. **Cabeçalho de requisição:** Recurso requisitado e/ou cliente
2. **Cabeçalho de resposta:** Resposta e/ou sobre o servidor
3. **Cabeçalho de entidade:** Corpo da requisição/resposta

```
POST / HTTP/1.1
Host: localhost:8000
User-Agent: Mozilla/5.0 (Macintosh;... )... Firefox/51.0
Accept: text/html,application/xhtml+xml,..., */*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Content-Type: multipart/form-data; boundary=-12656974
Content-Length: 345

-12656974
(more data)
```

Request headers

General headers

Entity headers

Cabeçalho

1. **Cabeçalho de requisição:** Recurso requisitado e/ou cliente;
2. **Cabeçalho de resposta:** Resposta e/ou sobre o servidor;
3. **Cabeçalho de entidade:** Corpo da requisição/resposta;

The diagram shows an HTTP response structure with the following components and labels:

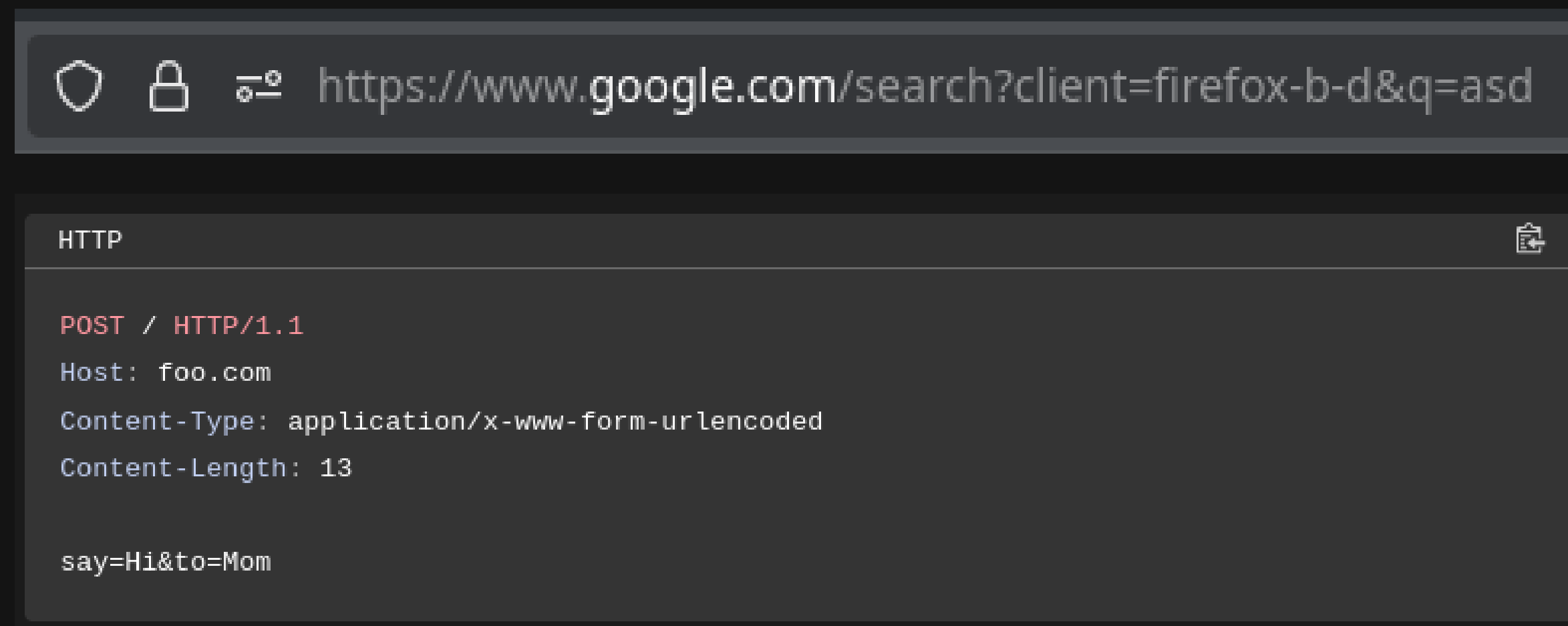
- Response headers:** Indicated by red arrows pointing to the following lines:
 - HTTP/1.1 200 OK
 - Access-Control-Allow-Origin: *
 - Etag: "d9b3b803e9a0dc6f22e2f20a3e90f69c41f6b71b"
 - Server: Apache
 - Set-Cookie: csrftoken=.....
 - X-Frame-Options: DENY
- Entity headers:** Indicated by black arrows pointing to the following lines:
 - Connection: Keep-Alive
 - Content-Encoding: gzip
 - Content-Type: text/html; charset=utf-8
 - Last-Modified: Wed, 10 Aug 2016 05:38:31 GMT
 - Transfer-Encoding: chunked
- General headers:** Indicated by green arrows pointing to the following lines:
 - Date: Wed, 10 Aug 2016 13:17:18 GMT
 - Keep-Alive: timeout=5, max=999

(body)

Corpo

Nem todas as requisições precisam de um corpo;

POST que envia um formulário de dados para o servidor utiliza o corpo para enviar os dados, ex:



Código de status

São indicadores de status finais de uma requisição, geralmente podemos dividi-los em 5 grupos:

1. Respostas Informativas (100 – 199)
2. Respostas bem-sucedidas (200 – 299)
3. Mensagens de redirecionamento (300 – 399)
4. Respostas de erro do cliente (400 – 499)
5. Respostas de erro do servidor (500 – 599)

Código de status

`{httpstats}`
`{http.cat}`

HTTPS (Hyper Text Transfer Protocol)

Versão segura do HTTP por implementar criptografia para as transferências de dados;

Utiliza TLS/SSL, com isso há a vantagem principal que é a segurança;

Desde julho de 2018 o Google definiu que o HTTP já não era mais seguro portanto teria um ranqueamento menor no seu SEO (Search Engine Optimization);

Para não entrar em conflito com o HTTP que usa a porta 80, o HTTPS usa a porta 443;

Geralmente é ofertado a utilização de TLS/SSL pelo provedor;

SSL (Secure Socket Layer)

Aplica a segurança em conexões entre servidores e clientes usando certificados digitais mitigando ataques hackers que possam interceptar esses dados (famosos sniffers)

handshake SSL:

1. Um navegador ou servidor tenta se conectar a um site protegido por um certificado SSL.
2. O navegador ou servidor solicita que esse servidor se identifique.
3. O servidor envia uma cópia do seu certificado SSL para o navegador ou servidor.
4. O navegador ou servidor verifica se o certificado SSL é de confiança. Se for, ele sinaliza isso ao servidor
5. O servidor retorna então uma confirmação assinada digitalmente para dar início a uma sessão criptografada por SSL.
6. Os dados criptografados são compartilhados entre o navegador ou servidor e o servidor

TLS (Transfer Layer Secure)

Evolução do SSL na segurança
handshake TLS 1.3:

1. Cliente envia um "olá" para o servidor contendo a versão do protocolo, um randomizador do cliente e um conjunto de criptografias.
2. Servidor utilizando as informações cedidas gera uma chave mestra usando um randomizador do servidor.
3. Servidor envia um "acabado" para o cliente contendo o certificado tls, a assinatura digital, o randomizado do servidor e o conjunto de criptografia escolhido.
4. Cliente envia um "concluído" assim que verifica a assinatura e o certificado, e gera a chave mestra utilizando as outras informações.

Servidores Web

Podem ser estáticos ou dinâmicos;

Estático: Consiste em um computador (hardware) com um servidor HTTP (software). É chamado de "estático" porque o servidor envia seus arquivos hospedados como estão para o navegador;

Dinâmico: Consiste em um servidor web estático mais software extra. É chamado de "dinâmico" porque o servidor de aplicação atualiza os arquivos hospedados antes de enviar conteúdo para o navegador;

Hosting

Hosting ou Hospedagem de site é como alugar um espaço na internet para armazenar todos os arquivos e dados de um site, tornando-o acessível para as outras pessoas;

Sem uma hospedagem, um site não pode ser visto por ninguém online;

A empresa de hospedagem mantém os servidores que hospedam seu site, garantindo que ele esteja sempre disponível e funcione corretamente;

Tipos de Hosting I

HOSPEDAGEM PADRÃO

Múltiplos usuários compartilham os mesmos recursos do servidor e esta é uma solução para pequenas empresas e sites pessoais;

HOSPEDAGEM DE SERVIDOR VPS

É semelhante a anterior, a diferença é que o provedor vai criar uma partição virtual para cada usuário, assim, este tipo de hospedagem é uma opção para sites de tamanho médio;

HOSPEDAGEM CLOUD

Usa um cluster de servidores virtuais para hospedar sites, quem se beneficia são sites de larga escala, como lojas de e-commerce e empresas com múltiplos sites;

Tipos de Hosting II

HOSPEDAGEM WORDPRESS

Oferece um ambiente de servidores otimizados para WordPress, tipicamente, vem com temas pré-instalados e plugins para funções-chave como cache e segurança, além de outras ferramentas;

HOSPEDAGEM DEDICADA

Designa um servidor físico para cada site, neste tipo é possível configurar o servidor, escolher o SO e o navegador desejado, este tipo é ideal para grandes empresas que precisam lidar com tráfego pesado.

Navegadores

São aplicativos de softwares que permitem o acesso à World Wide Web;

Neles é possível procurar uma resposta para qualquer pergunta que possa ter e também é possível acessar um site e depois ir para outro;

Exemplos:

1. Google Chrome
2. Safari
3. Microsoft Edge
4. Mozilla Firefox
5. Opera

User Agents

São uma cadeia de caracteres que o navegador envia ao servidor ao fazer uma requisição HTTP e isto permite aos servidores identificar o navegador utilizado, o tipo de dispositivo, o SO e a versão do agente de usuário requisitante.

Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:47.0) Gecko/20100101 Firefox/47.0 - **Firefox**

Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) - **Google Chrome**

Chrome/51.0.2704.106 Safari/537.36 OPR/38.0.2220.41 - **Opera**

Mozilla/5.0 (iPhone; CPU iPhone OS 13_5_1 like Mac OS X) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/13.1.1 Mobile/15E148 Safari/604.1 - **Safari**

Cookies

Dados armazenados no navegador do cliente mitigando a necessidade de requisições

Tamanho máximo de 4KB

“prazo de validade”

Session

Cookies só que armazenados no servidor

Dados sensíveis não precisam ficar transitando pela rede

Processo de “conectar” a sessão e o navegador que a está utilizando é feito usando os próprios cookies

CDN (Content Delivery Network)

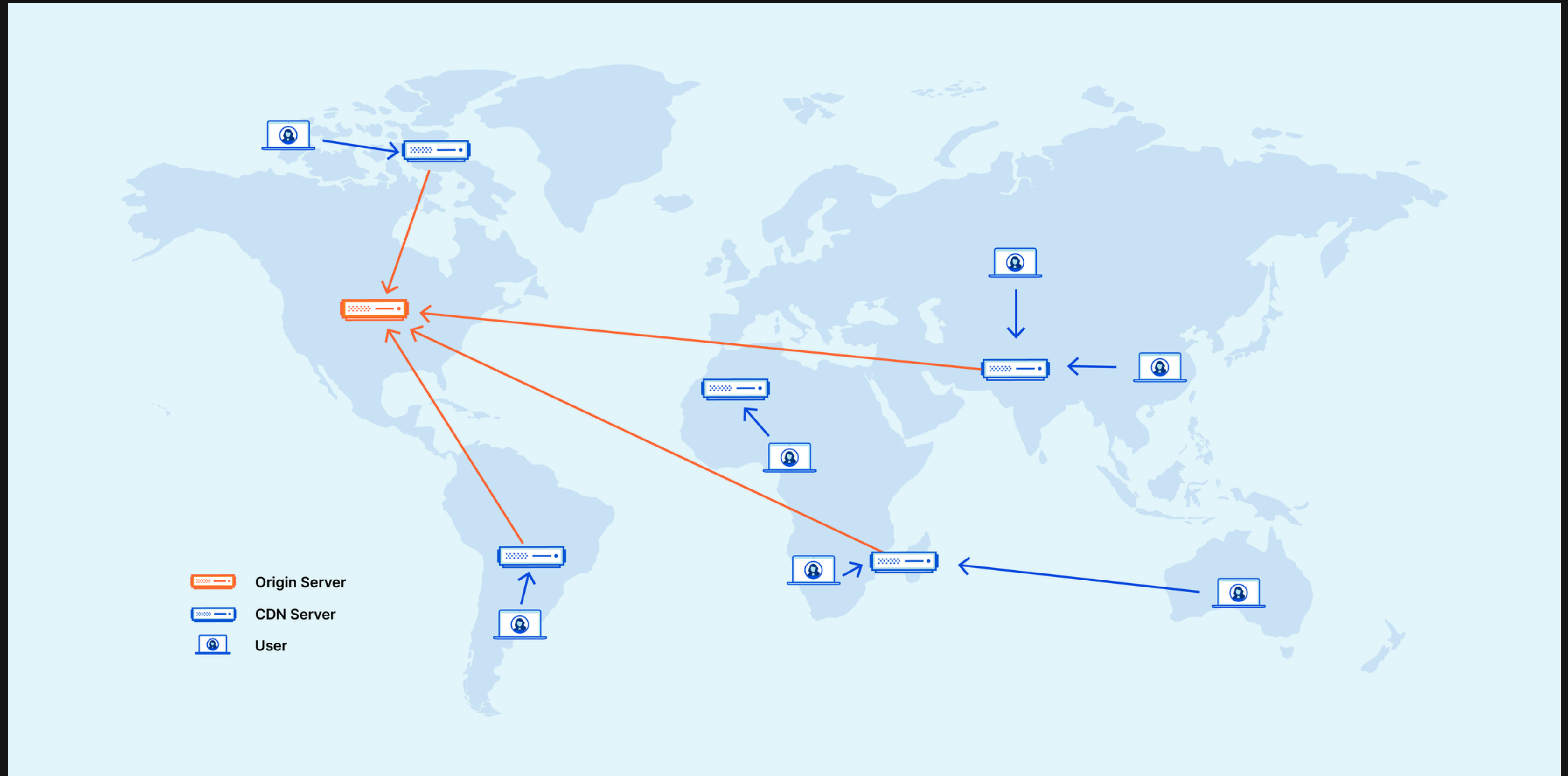
Acessos a um servidor de muitos lugares = latência de acesso maior

Espalhar servidores pelos principais centros populacionais

Porém em vez de servidores a ideia da CDN consiste em usar servidores não para hospedar mas para armazenar conteúdo em cache para os usuários geograficamente mais perto,

- Diminuindo a latência de acesso
- Reduzindo os custos de banda larga
- Redundância de dados
- Aumento da segurança.

CDN (Content Delivery Network)



Referências I

ROADMAP.sh. How does the internet work? Disponível em: <https://roadmap.sh/>. Acesso em: 1 set. 2024.

MOZILLA DEVELOPER NETWORK. How does the Internet work? Disponível em: https://developer.mozilla.org/en-US/docs/Learn/Common_questions/Web_mechanics/How_does_the_Internet_work. Acesso em: 1 set. 2024.

MOZILLA DEVELOPER NETWORK. What is a domain name? Disponível em: https://developer.mozilla.org/en-US/docs/Learn/Common_questions/Web_mechanics/What_is_a_domain_name. Acesso em: 1 set. 2024.

AUTH0. URL, URI, and URN: Clarifying the differences. Disponível em: <https://auth0.com/blog/url-uri-urn-differences/>. Acesso em: 1 set. 2024.

Referências II

MOZILLA DEVELOPER NETWORK. What is a web server? Disponível em:

[https://developer.mozilla.org/en-](https://developer.mozilla.org/en-US/docs/Learn/Common_questions/Web_mechanics/What_is_a_web_server)

[US/docs/Learn/Common_questions/Web_mechanics/What_is_a_web_server](https://developer.mozilla.org/en-US/docs/Learn/Common_questions/Web_mechanics/What_is_a_web_server). Acesso em: 1 set. 2024.

HOSTINGER. O que é hospedagem de site? Entenda tudo sobre esse serviço. Disponível em:

<https://www.hostinger.com.br/tutoriais/o-que-e-hospedagem-de-sit>. Acesso em: 1 set. 2024.

AVAST. O que é um navegador de internet? Disponível em: <https://www.avast.com/pt-br/c-what-is-a-web-browser>. Acesso em: 1 set. 2024.

MOZILLA DEVELOPER NETWORK. HTTP headers: User-Agent. Disponível em:

<https://developer.mozilla.org/pt-BR/docs/Web/HTTP/Headers/User-Agent>. Acesso em: 1 set. 2024.

Referências III

NTI SOLUÇÕES. O que é protocolo de rede. Disponível em: <https://ntisolucoes.srv.br/o-que-e-protocolo-de-rede/>. Acesso em: 1 set. 2024.

ROCK CONTENT. HTTP: o que é, para que serve e como funciona. Disponível em: <https://rockcontent.com/br/blog/http/>. Acesso em: 1 set. 2024.

BYLEARN. Protocolo HTTP e HTTPS. Disponível em: <https://dojo.bylearn.com.br/tecnologia/protocolo-http-e-https/>. Acesso em: 1 set. 2024.

MOZILLA DEVELOPER NETWORK. Métodos HTTP. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Web/HTTP/Methods>. Acesso em: 1 set. 2024.

LINKEDIN. Qual a diferença entre PUT e POST? Disponível em: <https://www.linkedin.com/advice/3/what-difference-between-put-post-request-skills-sales-engineering-tujme?lang=pt&originalSubdomain=pt>. Acesso em: 1 set. 2024.

Referências IV

MOZILLA DEVELOPER NETWORK. Mensagens HTTP. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Web/HTTP/Messages>. Acesso em: 1 set. 2024.

CLOUDFLARE. O que é HTTPS? Disponível em: <https://www.cloudflare.com/pt-br/learning/ssl/what-is-https/>. Acesso em: 1 set. 2024.

ROCK CONTENT. O que é TLS e SSL? Disponível em: <https://rockcontent.com/br/blog/tls-ssl/>. Acesso em: 1 set. 2024.

KASPERSKY. O que é um certificado SSL. Disponível em: <https://www.kaspersky.com.br/resource-center/definitions/what-is-a-ssl-certificate>. Acesso em: 1 set. 2024.

GANESH ICMC. Cookies e sessões. Disponível em: https://gitbook.ganeshicmc.com/web/semana-1/11_cookies_e_sesoes. Acesso em: 1 set. 2024.

Referências V

CLOUDFLARE. O que é um CDN? Disponível em: <https://www.cloudflare.com/learning/cdn/what-is-a-cdn/>. Acesso em: 1 set. 2024.