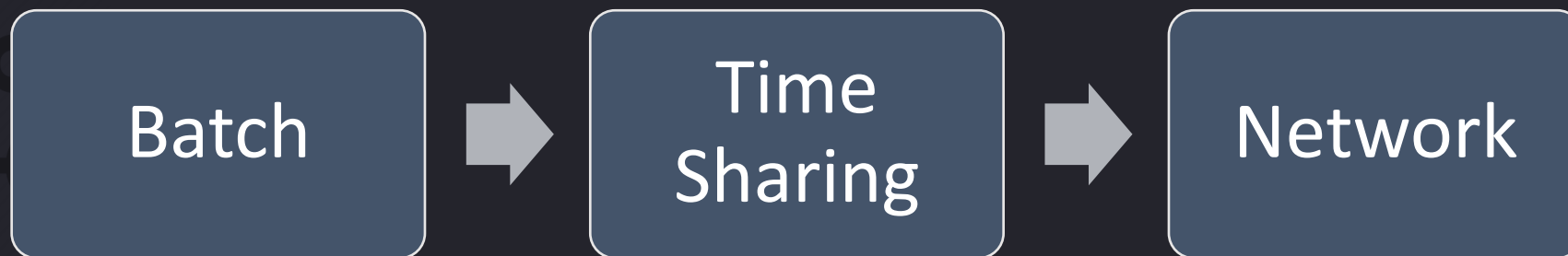


Fundamentos de Redes de Computadores



Curiosidades Históricas

- Até o fim da década de 1950, os computadores eram operados de forma individual
- Não havia interatividade entre usuário e máquina
- A partir da década de 1960, aparece o conceito de Time Sharing (TS)
 - Compartilhamento de tempo de execução entre usuário e terminal
 - Inicou-se o processo de troca de pacotes de informações no MIT, substituindo circuitos físicos na comunicação.



Curiosidades Históricas

- Em 1969, surge a ARPANET (tataravó da nossa internet atualmente)
- ARPA = Advanced Research Project Agency.
- Em 1974, Xerox cria o protótipo do Ethernet
- Em 1981, TCP/IP é testado pelos militares, formando a internet dos dias atuais
- Em 1983, ARPANET é dividida entre civil e militar.
- Década de 1980, começam a surgir os PCs.
- 1990 é a década do Boom da internet.

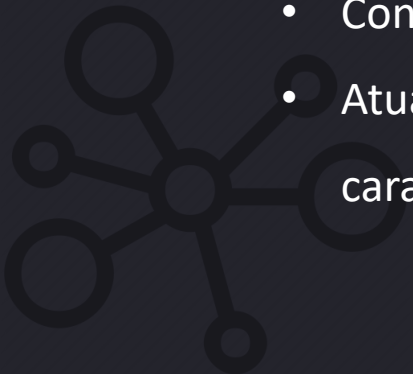


Conceitos Básicos

Conjunto de computadores autônomos interligados por um sistema de comunicação, capazes de trocar informações de forma confiável e compartilhar recursos (hardware, software e dados).

Objetivos

- Compartilhamento de recursos.
- Construção de sistemas com maior desempenho e confiabilidade.
- Atuação como meio de compatibilização entre usuários de características heterogêneas



Arquitetura de Rede

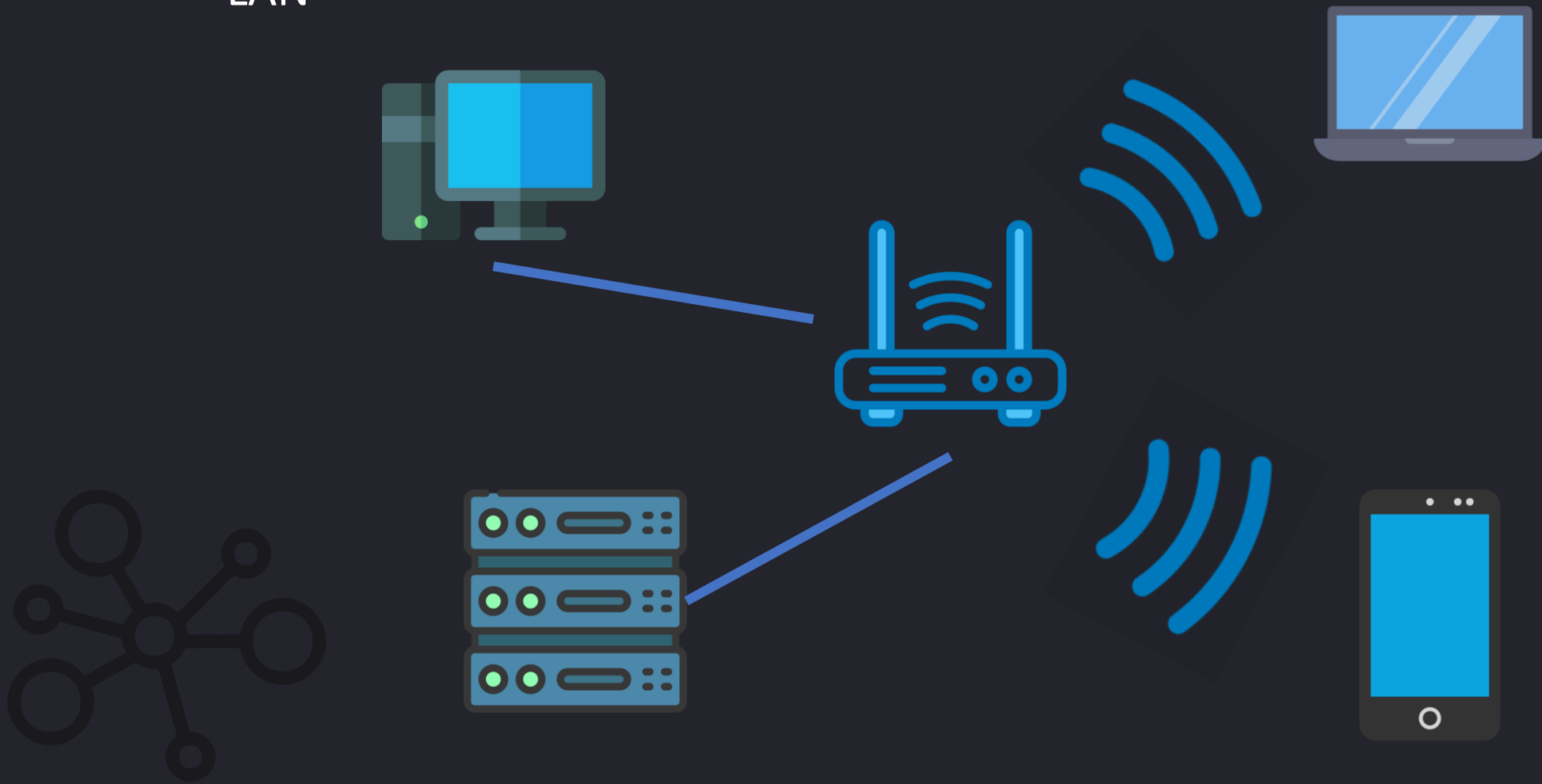
Classificação por grupos:

- WAN – Wide Area Network
 - Propriedade Pública
- MAN – Metropolitan Area Network
 - Híbrida – utiliza WAN mas é gerenciada de forma privada
- LAN – Local Area Network
 - Propriedade Privada



Arquitetura de Rede

LAN



Arquitetura de Rede

MAN

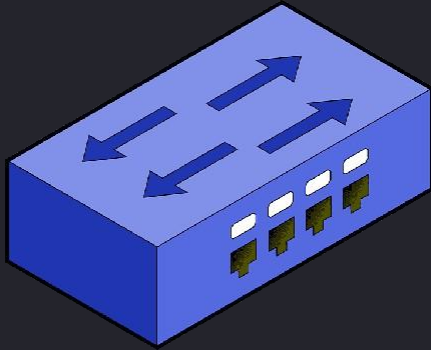


Arquitetura de Rede

WAN



Equipamentos de uma rede



Switches

Responsáveis pela interconexão dos equipamentos de uma rede

8, 16, 24 ou 48 portas são os mais comuns

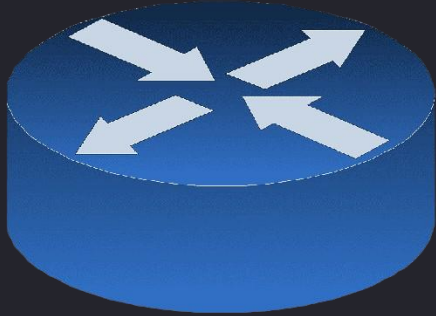
Gerenciáveis e não gerenciáveis

Segmentação de rede

Trabalha na camada 2 do modelo OSI



Equipamentos de uma rede



Roteadores

Responsáveis pela interconexão de redes distintas

Normalmente modulares (cisco, aruba e afins)

Gerenciáveis

Trabalha na camada 3 do modelo OSI



Equipamentos de uma rede



Access Points

Conversores de mídia

Equipamentos corretos para se estruturar uma rede wifi

Gerenciáveis

Trabalha na camada 1, 2 e 3 do modelo OSI



CSFI
S
n

Equipamentos de uma rede



Fibras Ópticas

Interconexão física para longas distâncias

Throughput de até 100Tbits

Gerenciáveis

Camada 1 do modelo OSI



Protocolos de Rede

Conjunto de regras e padrões que compõe uma espécie de “linguagem universal” entre computadores e dispositivos, propiciando a comunicação, conexão ou transferência de dados entre quaisquer máquinas conectas à internet.

Elementos de protocolos

- Sintaxe – Formato e ordem pelos quais os dados são apresentados
- Semantica – Conferência do significado dos conjuntos sintáticos
- Timing – Definição de velocidade da comunicação



Protocolos de Rede

MODELO OSI

Aplicação	Geração dos dados
Apresentação	Tradução e conversão dos dados a serem transportados
Sessão	Negociação da conexão de transporte
Transporte	Encapsulamento do pacote de acordo com o protocolo
Rede	Encapsulamento de endereçamento (origem/destino)
Ligação (Enlace)	Segmentação dos frames
Física	Conversão de dados lógicos para físicos



Protocolos de Rede

MODELO TCP

Aplicação

Geração dos dados

Transporte

Encapsulamento do pacote de acordo com o protocolo

Rede

Encapsulamento de endereçamento (origem/destino)

Física

Conversão de dados lógicos para físicos



Endereçamentos de Rede

Classes de IPs

Reservas de endereços de acordo com a quantidade e o range de endereços

- Classe A – 10.0.0.0/8 – 16+ milhões de endereços
- Classe B – 172.16.0.0/16 – 65536 endereços
- Classe C – 192.168.0.0/24(16) – 256 endereços (65536 endereços)
- Classe D (Multicast) – 224.0.0.0/4
- Classe E (Link Local) – 169.254.0.0/16



Endereçamentos de Rede

Máscara de Sub-Redes

Usada para definir o que é endereço de rede e o que é endereço de host.

R	E	D	E	HOSTS
192	168	0	200	
255	255	255	0	
11111111	11111111	11111111	00000000	



Endereçamentos de Rede

Máscara de Sub-Redes

Usada para definir o que é endereço de rede e o que é endereço de host.



Endereçamentos de Rede

Calculando quantidade de endereços válidos

Em uma rede /24, temos 254 endereços válidos.

Mas porque, se fizermos as contas existem 256 números possíveis?

192.168.0.0

11000000.10101000.00000000.00000000
11111111.11111111.11111111.00000000

255.255.255.0



Endereçamentos de Rede

192.168.0.0

11000000.10101000.00000000.00000000

11111111.11111111.11111111.00000000

255.255.255.0

Toda rede tem 2 endereços reservados

O primeiro que representa o CIDR ou ID da rede;

O último, utilizado para broadcast.



Endereçamentos de Rede

192.168.0.0

11000000.10101000.00000000.00000000

11111111.11111111.11111111.00000000

255.255.255.0

Portanto, toda rede, independentemente de sua máscara,
terá sempre **2 endereços a menos** utilizáveis em hosts



Endereçamentos de Rede

192.168.0.0

11000000.10101000.00000000.00000000

11111111.11111111.11111111.00000000

255.255.255.0

Na rede com máscara /24 temos então 254 endereços válidos
192.168.0.1 – 192.168.0.254



Endereçamentos de Rede

Calculando quantidade de endereços válidos

Em uma rede /25, temos então 2 redes de 128 endereços válidos.

192.168.0.0

11000000.10101000.00000000.00000000
11111111.11111111.11111111.10000000

255.255.255.128



Endereçamentos de Rede

Calculando quantidade de endereços válidos

Em uma rede /25, temos então 2 redes de 128 endereços válidos.

Rede 1

CIDR – 192.168.0.0/25

Primeiro válido – 192.168.0.1

Último válido – 192.168.0.126

Broadcast – 192.168.0.127

Rede 2

CIDR – 192.168.0.128/25

Primeiro válido – 192.168.0.129

Último válido – 192.168.0.254

Broadcast – 192.168.0.255



Endereçamentos de Rede

Calculando quantidade de endereços válidos

Se precisarmos de mais de 256 endereços de rede em uma rede, podemos utilizar uma máscara de rede maior.

Uma rede /23, temos então 1 rede de 512 endereços válidos.

Rede 1

CIDR – 172.16.0.0/23

Primeiro válido – 172.16.0.1

Último válido – 172.16.1.254

Broadcast – 172.168.1.255



MÃO NA MASSA



CSFTS

Softwares para emulação de redes



Cisco Packet Tracer



Endereçamentos de Rede

Geramos, com redes grandes, um problema comum do protocolo TCP/IP chamado gargalo de rede.

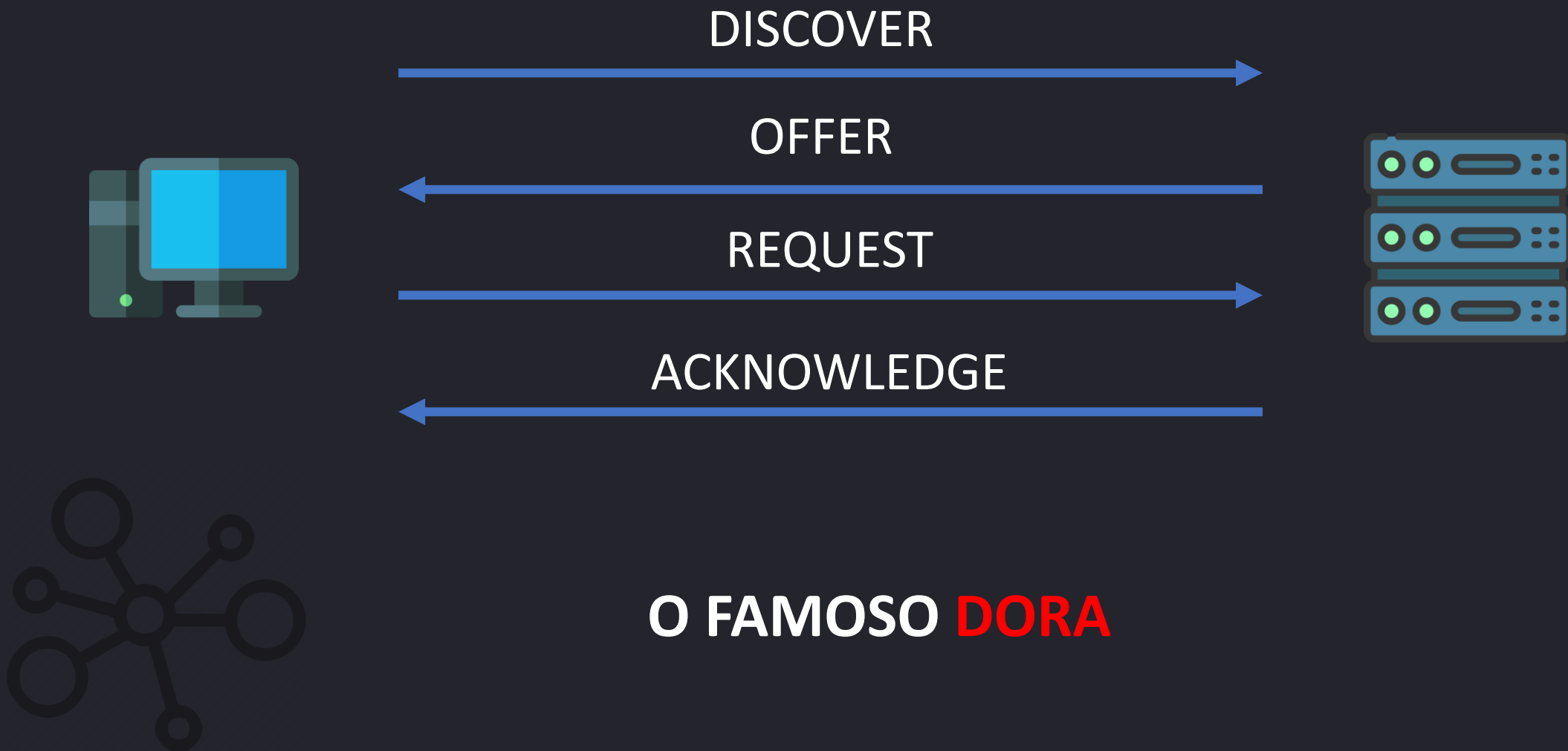
Broadcasting é o modelo de transmissão de dados a todos os nós da rede.

Como toda rede possui um endereço reservado para tal recurso, quanto maior a rede, mais *hosts* a comunicação atinge, mais respostas ao broadcast, mais tráfego de rede e chegamos ao gargalo de rede causado por domínio de broadcast.

Vamos usar o DHCP para exemplificar isso...



Endereçamentos de Rede



Endereçamentos de Rede



Protocolos e Portas

Existem 2 tipos de protocolos de transporte:

- TCP – orientado a conexão (mais lento porém mais confiável)
- UDP – baseado em transmissão (mais rápido e comum nos streammings)

A partir deles que é possível os protocolos de sessão estabelecerem a comunicação de dados usando portas (sockets) no S.O.

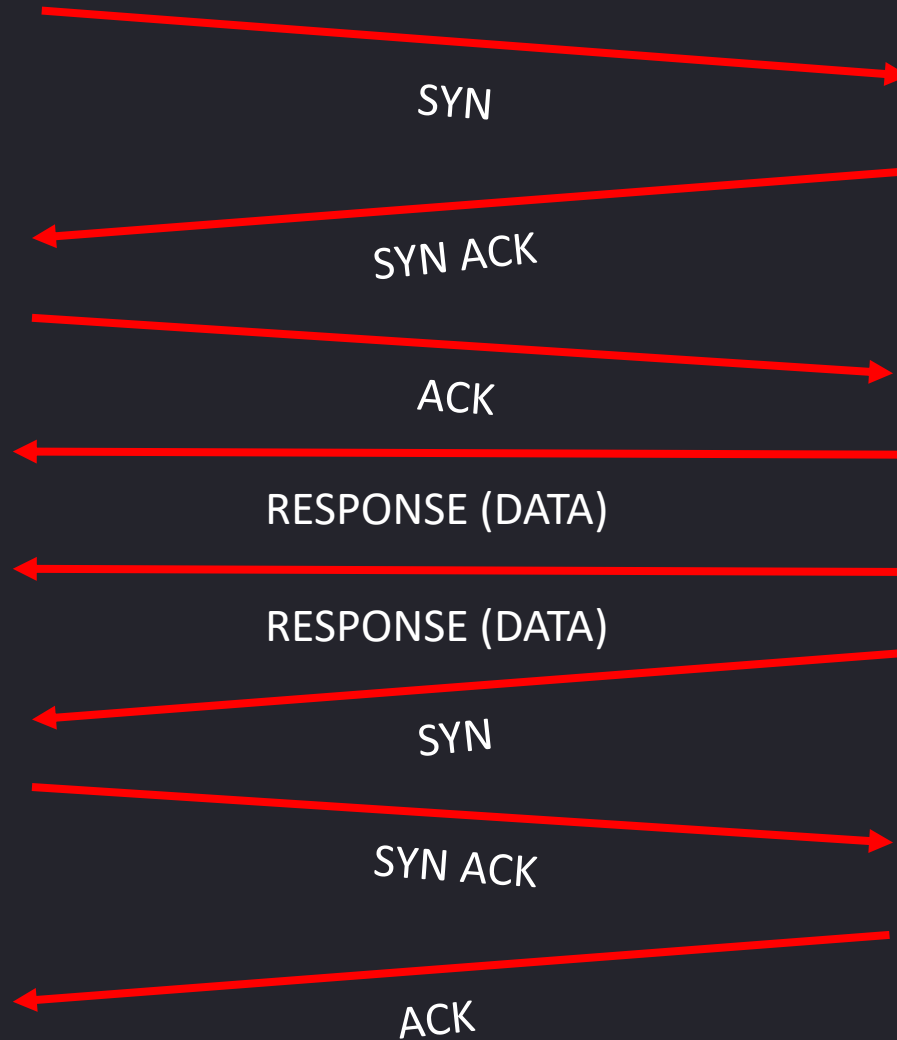
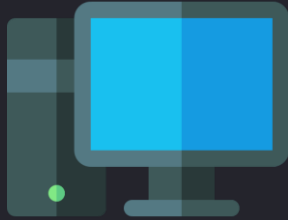
Todo protocolo de sessão usa uma porta padrão para estabelecer essa comunicação.

Temos 65535 portas de comunicação disponíveis.



Protocolos e Portas

TCP



Protocolos e Portas

UDP



REQUEST

RESPONSE (DATA)

RESPONSE (DATA)

RESPONSE (DATA)

RESPONSE (DATA)



Tabela NAT

Network Address Translation é usado para encaminhamento de portas de host público para um host privado.

É usado quando precisamos expor serviços internos para uso na internet.

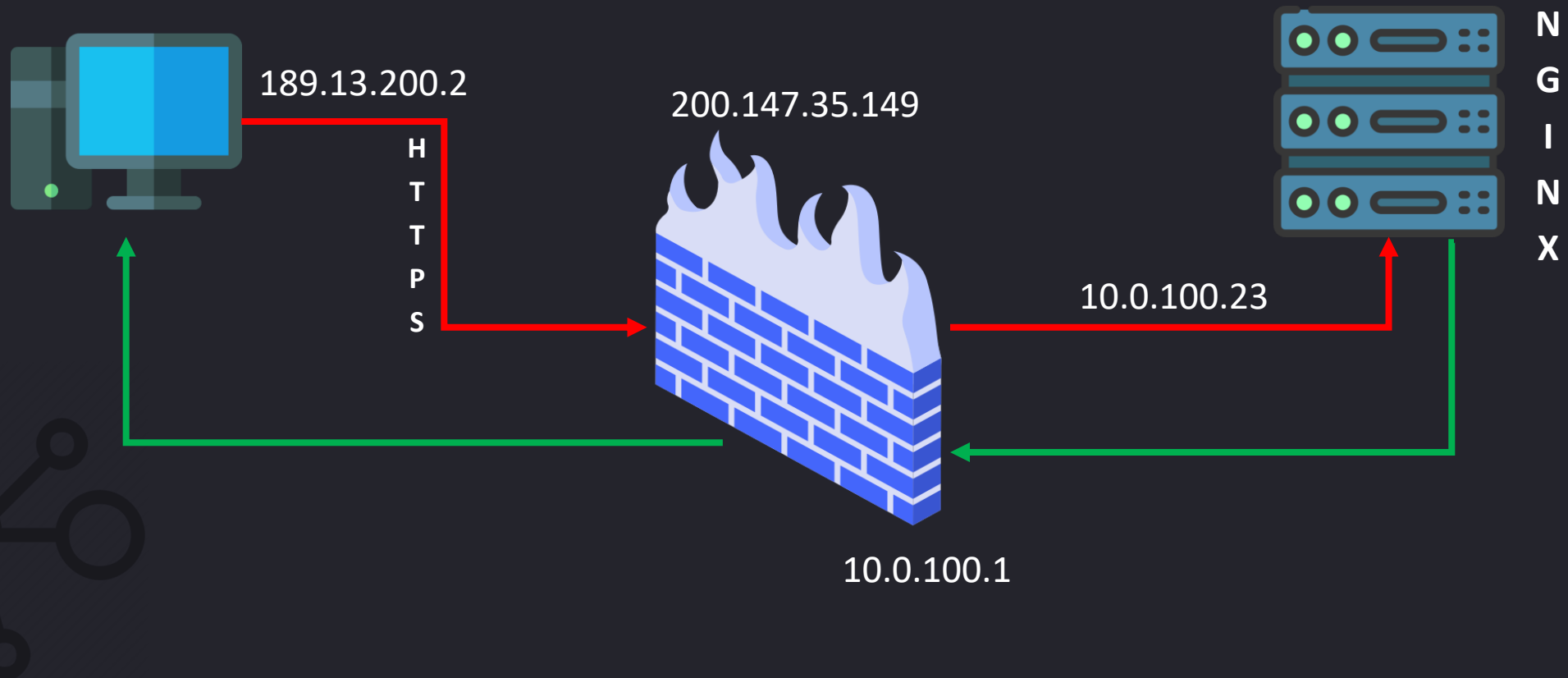
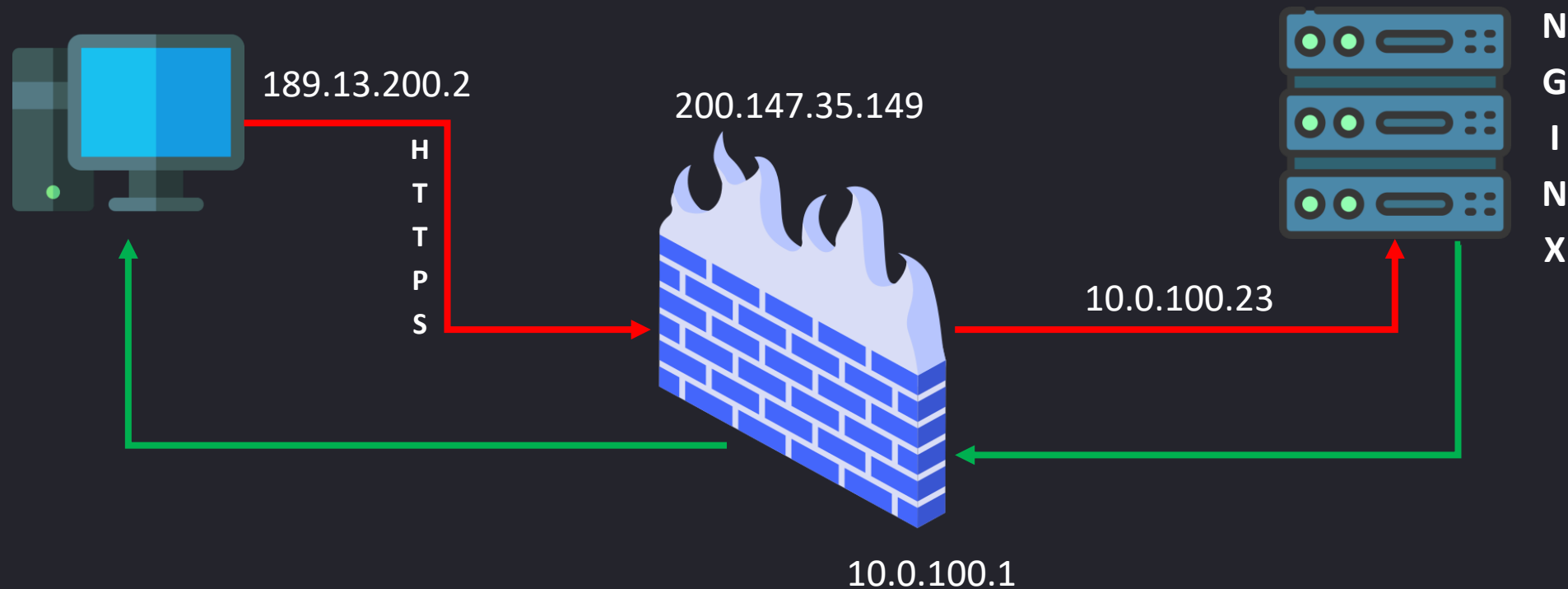


Tabela NAT



SOURCE ADDRESS	SOURCE PORT	DESTINATION ADDRESS	DESTINATION PORT
200.147.35.149	443	10.0.100.23	443
200.147.35.139	1433	10.0.100.25	1433
200.147.35.139	8080	10.0.100.50	80

APIs

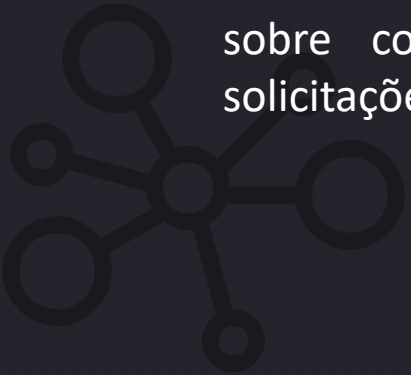
APIs são mecanismos que permitem que dois componentes de software se comuniquem usando um conjunto de definições e protocolos.

API significa Application Programming Interface (Interface de Programação de Aplicação).

No contexto de APIs, a palavra Aplicação refere-se a qualquer software com uma função distinta.

A interface pode ser pensada como um contrato de serviço entre duas aplicações. Esse contrato define como as duas se comunicam usando solicitações e respostas.

A documentação de suas respectivas APIs contém informações sobre como os desenvolvedores devem estruturar essas solicitações e respostas.



Tipos de APIs

APIs SOAP

Essas APIs usam o Simple Object Access Protocol (Protocolo de Acesso a Objetos Simples). Cliente e servidor trocam mensagens usando XML.

Esta é uma API menos flexível que era mais popular no passado.

APIs RPC

Essas APIs são conhecidas como Remote Procedure Calls (Chamadas de Procedimento Remoto).

O cliente conclui uma função (ou um procedimento) no servidor e o servidor envia a saída de volta ao cliente.

APIs WebSocket

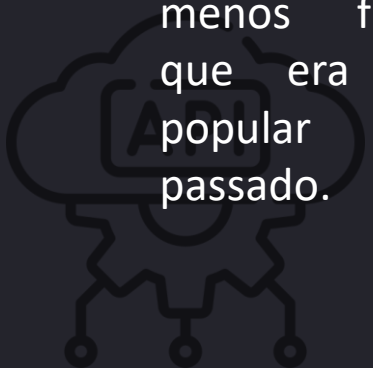
WebSocket é outro desenvolvimento de API da Web moderno que usa objetos JSON para transmitir dados.

Uma API WebSocket oferece suporte à comunicação bidirecional entre cliente e o servidor. O servidor pode enviar mensagens de retorno de chamada a clientes conectados, tornando-o mais eficiente que a API REST.

APIs REST

Essas são as APIs mais populares e flexíveis encontradas na Web atualmente.

O cliente envia solicitações ao servidor como dados. O servidor usa essa entrada do cliente para iniciar funções internas e retorna os dados de saída ao cliente.



APIs REST

REST significa Transferência Representacional de Estado. **REST** define um conjunto de funções como **GET**, **PUT**, **DELETE** e assim por diante, que os clientes podem usar para acessar os dados do servidor. Clientes e servidores trocam dados usando **HTTP**.

A principal característica da **API REST** é a ausência de estado. A ausência de estado significa que os servidores não salvam dados do cliente entre as solicitações.

As solicitações do cliente ao servidor são semelhantes aos URLs que você digita no navegador para visitar um site.

A resposta do servidor corresponde a dados simples, sem a renderização gráfica típica de uma página da Web.



Web APIs

O que é API Web?

Uma API Web ou API de serviço da Web é uma interface de processamento de aplicações entre um servidor da Web e um navegador da Web.

Todos os serviços da Web são APIs, mas nem todas as APIs são serviços da Web. A API REST é um tipo especial de API Web que usa o estilo de arquitetura padrão explicado acima.

Os diferentes termos que abrangem as APIs, como API C# ou APIs de serviço, existem porque, historicamente, as APIs foram criadas antes da World Wide Web.

As APIs Web modernas são APIs REST e os termos podem ser usados de forma intercambiável.



Protocolo HTTP

HTTP é um protocolo bem extensivo. Isso depende um pouco do conceito básico com noção de recursos e URIs, uma simples estrutura de mensagens, e uma estrutura de cliente-servidor para a comunicação ocorrer.

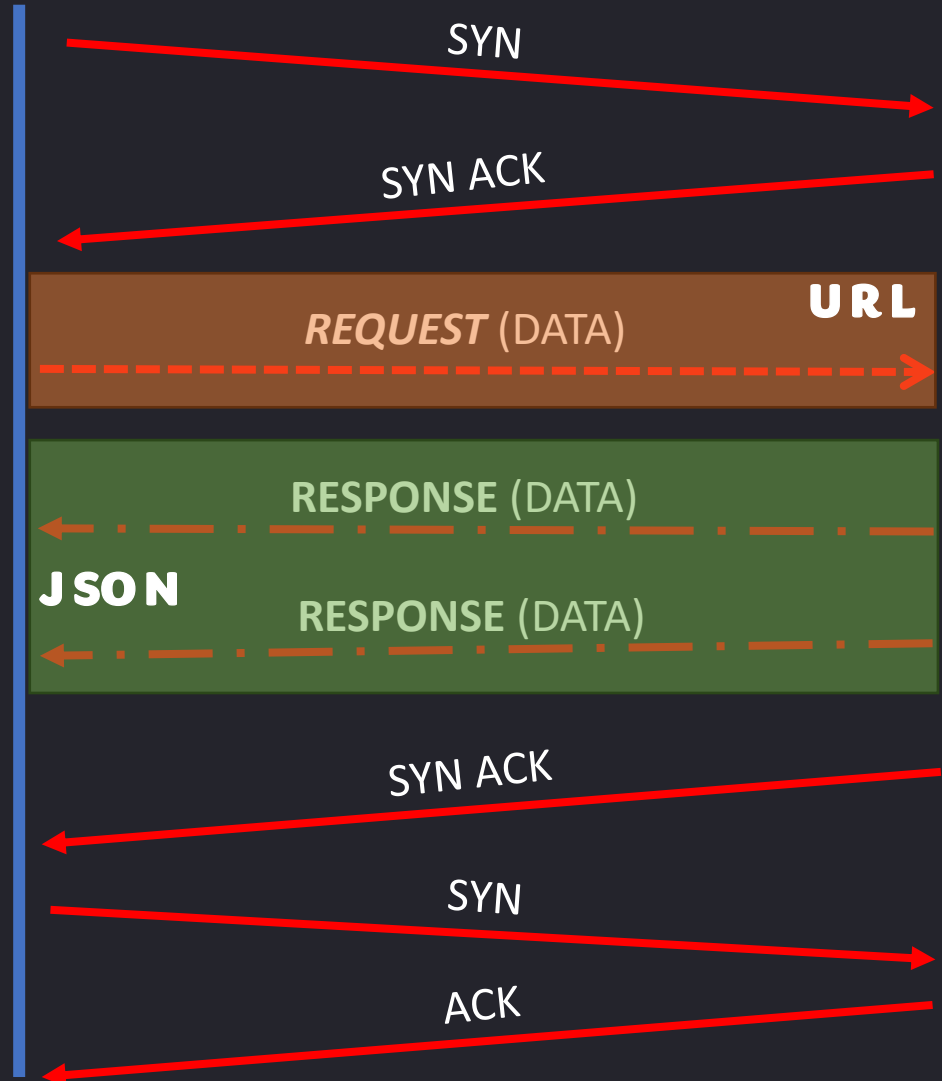
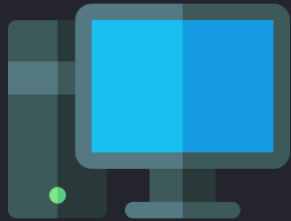
Em cima destes conceitos básicos, várias versões surgiram ao longo do tempo, adicionando novas funcionalidades e novas semânticas para criar novos métodos HTTP ou cabeçalhos.

Ele permite a obtenção de recursos, como documentos HTML. É a base de qualquer troca de dados na Web e um protocolo cliente-servidor, o que significa que as requisições são iniciadas pelo destinatário, geralmente um navegador da Web.

Um documento completo é reconstruído a partir dos diferentes sub-documentos obtidos, como por exemplo texto, descrição do layout, imagens, vídeos, scripts e muito mais.



TCP / HTTP









CS
FI
S

Verbos HTTP

O protocolo **HTTP** define um conjunto de **métodos de requisição** responsáveis por indicar a ação a ser executada para um dado recurso.

Embora esses métodos possam ser descritos como substantivos, eles também são comumente referenciados como **HTTP Verbs** (Verbos HTTP).

 GET	 POST	 PUT	 DELETE	 PATCH	 HEAD
Busca informações de um servidor	Adiciona dados a um recurso existente	Atualiza (substituindo) o recurso existente no servidor	Apaga informações de um servidor	Atualiza (parcialmente) o recurso existente no servidor	Busca informações de cabeçalho



Status Codes HTTP

Os códigos de status de resposta HTTP indicam se uma solicitação HTTP específica foi concluída com êxito.

As respostas são agrupadas em cinco classes:

Informational (100 – 199)	Successful (200 – 299)	Redirection (300 – 399)	Client error (400 – 499)	Server error responses (500 – 599)
100 Continue	200 Ok	300 Mult. Choice	400 Bad Req.	500 Internal Server Error
101 Sw Protocols	201 Created	301 Move Perm.	401 Unauth.	501 Not Implemented
102 Processing	202 Accepted	305 Use Proxy	403 Forbidden	503 Service Unavailable
103 Early Hints	204 No Content	307 Temp Redir	404 Not Found	504 Gw Timeout

<https://developer.mozilla.org/en-US/docs/Web/HTTP/Status>



API DATA

Os objetos de dados trafegados nas APIs, normalmente utilizam o JSON como padrão.

JSON é o nome dado ao *Java Script Object Notation*, tendo a estrutura padrão de chave / valor para sua criação.



```
{
  "name": "Luke Skywalker",
  "height": "172",
  "mass": "77",
  "hair_color": "blond",
  "skin_color": "fair",
  "eye_color": "blue",
  "birth_year": "19BBY",
  "gender": "male",
  "homeworld":
    "https://swapi.dev/api/planets/1/",
  "films": [
    "https://swapi.dev/api/films/2/",
    "https://swapi.dev/api/films/6/",
    "https://swapi.dev/api/films/3/",
    "https://swapi.dev/api/films/1/",
    "https://swapi.dev/api/films/7/"
  ],
  "species": [
    "https://swapi.dev/api/species/1/"
  ],
  "vehicles": [
    "https://swapi.dev/api/vehicles/14/",
    "https://swapi.dev/api/vehicles/30/"
  ],
  "starships": [
    "https://swapi.dev/api/starships/12/",
    "https://swapi.dev/api/starships/22/"
  ],
  "created": "2014-12-09T13:50:51.644000Z",
  "edited": "2014-12-20T21:17:56.891000Z",
  "url": "https://swapi.dev/api/people/1/"
}
```

Tipos dos dados em JSON

Os tipos de dados aceitos dentro de um JSON são os seguintes:

- Uma string
- Um número
- Um objeto (JSON)
- Um array
- Um boolean
- null

```
{  
  "name": "Luke Skywalker",  
  "height": 172,  
  "others": {  
    "species": [  
      "https://swapi.dev/api/species/1/"  
    ],  
    "vehicles": [  
      "https://swapi.dev/api/vehicles/14/",  
      "https://swapi.dev/api/vehicles/30/"  
    ],  
    "starships": [  
      "https://swapi.dev/api/starships/12/",  
      "https://swapi.dev/api/starships/22/"  
    ],  
    "created": "2014-12-09T13:50:51.644000Z"  
  },  
  "films": [  
    "https://swapi.dev/api/films/2/",  
    "https://swapi.dev/api/films/6/",  
    "https://swapi.dev/api/films/3/",  
    "https://swapi.dev/api/films/1/",  
    "https://swapi.dev/api/films/7/"  
  ],  
  "edited": false,  
  "url": null  
}
```

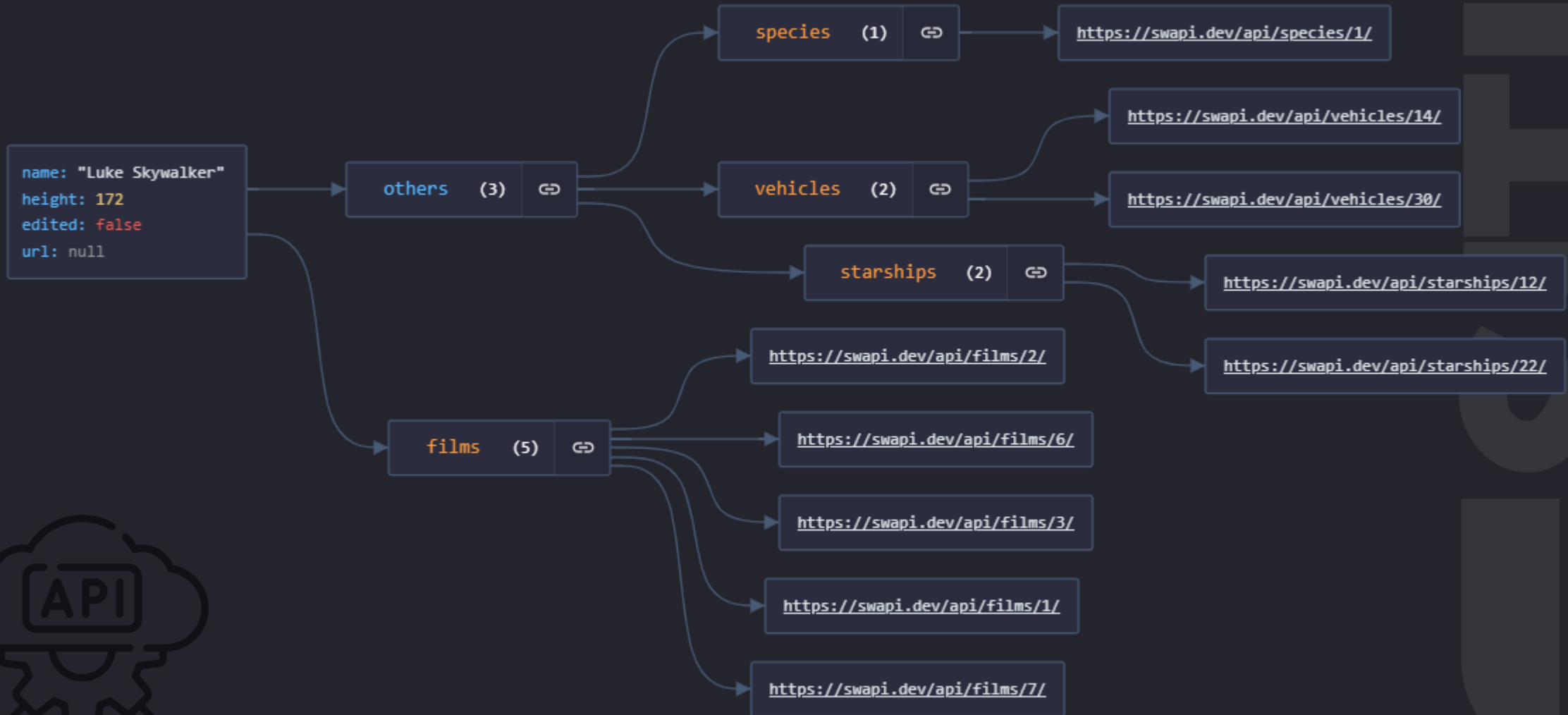


Tipos dos dados em JSON

```
{
  "name": "Luke Skywalker",
  "height": 172,
  "others": {
    "species": [
      "https://swapi.dev/api/species/1/"
    ],
    "vehicles": [
      "https://swapi.dev/api/vehicles/14/",
      "https://swapi.dev/api/vehicles/30/"
    ],
    "starships": [
      "https://swapi.dev/api/starships/12/",
      "https://swapi.dev/api/starships/22/"
    ],
    "created": "2014-12-09T13:50:51.644000Z"
  },
  "films": [
    "https://swapi.dev/api/films/2/",
    "https://swapi.dev/api/films/6/",
    "https://swapi.dev/api/films/3/",
    "https://swapi.dev/api/films/1/",
    "https://swapi.dev/api/films/7/"
  ],
  "edited": false,
  "url": null
}
```



Tipos dos dados em JSON



MÃO NA MASSA



CSFTS