SÃO
PAULO
TECH
SCHOOL

Computação e sistemas distribuídos em nuvem

Servidores Web

Eduardo Verri

eduardo.verri@sptech.school

Introdução à Web Servers

A função principal de um servidor web é servir páginas web de um site.

Uma página da web pode ser renderizada a partir de um único arquivo HTML ou de uma variedade complexa de recursos agrupados.

Se você deseja hospedar sua aplicação web na internet, em muitos casos você precisará de um servidor web.

Um dos casos de uso mais comuns para servidores web é servir arquivos necessários para renderizar um site em um navegador.

Um servidor web lida com solicitações na Internet por meio dos protocolos HTTP e HTTPS e também é chamado de servidor HTTP.

Um servidor web é diferente de outros tipos de servidores porque é especializado em lidar com essas solicitações HTTP e HTTPS, diferenciando-se de servidores de aplicativos e servidores para outros protocolos

- Serve arquivos HTML, CSS e JavaScript
- Serve imagens e vídeos.
- Lida com mensagens de erro HTTP
- Lida com solicitações de usuários, muitas vezes simultaneamente
- Direciona a correspondência e reescrita de URL
- Processa e fornece conteúdo dinâmico
- Compacta conteúdo para uso e velocidade de dados otimizados
- Ativa o cache do navegador para seu conteúdo estático.

Objetivos de um servidor Web

Uptime (Tempo de atividade): refere-se ao tempo que um servidor web está online e operacional. Os sites precisam estar online o tempo todo para atender aos usuários, portanto, o objetivo é um alto tempo de atividade. Isso também se traduz em estabilidade e previsibilidade. Quando um usuário insere um URL ou clica em um link para seu site, a página esperada deve carregar sempre e a qualquer momento. As únicas exceções devem ser os tempos de inatividade planejados para atualizações ou manutenção.

Velocidade: suas páginas da web devem carregar o mais rápido possível. Os usuários desejam que sua solicitação seja atendida imediatamente, caso contrário você corre o risco de perdê-los. Em uma página da web de carregamento lento, mesmo que o usuário aguarde o primeiro carregamento, cada carregamento longo subsequente diminuirá exponencialmente sua disposição de permanecer ou visitar novamente

Objetivos de um servidor Web

Simultaneidade: refere-se ao tratamento de várias solicitações recebidas ao mesmo tempo. Ter muitas pessoas tentando visitar seu site ao mesmo tempo parece uma coisa boa, mas isso se torna um problema real quando o tempo de carregamento fica lento e todo o servidor trava. Seu servidor físico ou virtual possui apenas alguns recursos, como RAM e capacidade de computação da CPU, e os servidores web devem usar esses recursos com eficiência.

Escalabilidade: refere-se a tornar seus servidores existentes mais poderosos por meio de escalonamento vertical ou adicionar mais servidores à sua configuração por meio de escalonamento horizontal. À medida que seu público aumenta, você pode chegar a um ponto em que precisará de mais de um ou dois pequenos servidores web.

Objetivos de um servidor Web

Setup fácil: colocar um projeto em funcionamento rapidamente é fundamental para a iteração do seu projeto. Um processo de instalação simples e repetível é importante para o primeiro servidor web que você configurar e para os vários servidores web posteriormente, quando você aumentar a escala..

Documentação: os servidores Web são complexos. As configurações mais comuns ajudarão você a se recuperar rapidamente, mas suas necessidades aumentarão com o tempo. Muitas vezes você precisará de recursos que não são tão usados. Quando chegar esse momento, uma boa documentação é essencial para criar soluções personalizadas para suas necessidades

Objetivos de um servidor Web

Suporte ao desenvolvedor: se os desenvolvedores principais não estiverem comprometidos com seus próprios projetos, você não deve comprometer seu projeto com os deles. Isso inclui planos de suporte de longo prazo para seu software, juntamente com suporte imediato de curto prazo que eles fornecem na forma de correções de bugs e patches.

Suporte da comunidade: uma equipa central de desenvolvimento tratará da maior parte do trabalho pesado, mas uma comunidade próspera contribui para preencher as lacunas. Com projetos de código aberto, isso pode significar contribuições para a base de código real, mas uma comunidade forte também responderá às suas perguntas e ajudará com seus problemas específicos.

O lado cliente

Em essência, um navegador é um programa que pode exibir páginas Web e clicar com o mouse em itens na página exibida. Quando um item é selecionado, o navegador segue o hiperlink e busca a página selecionada.

Suponha que um usuário clique em um link na Internet que aponta para *home page* da ITU, http://www.itu.org/home/index.html:

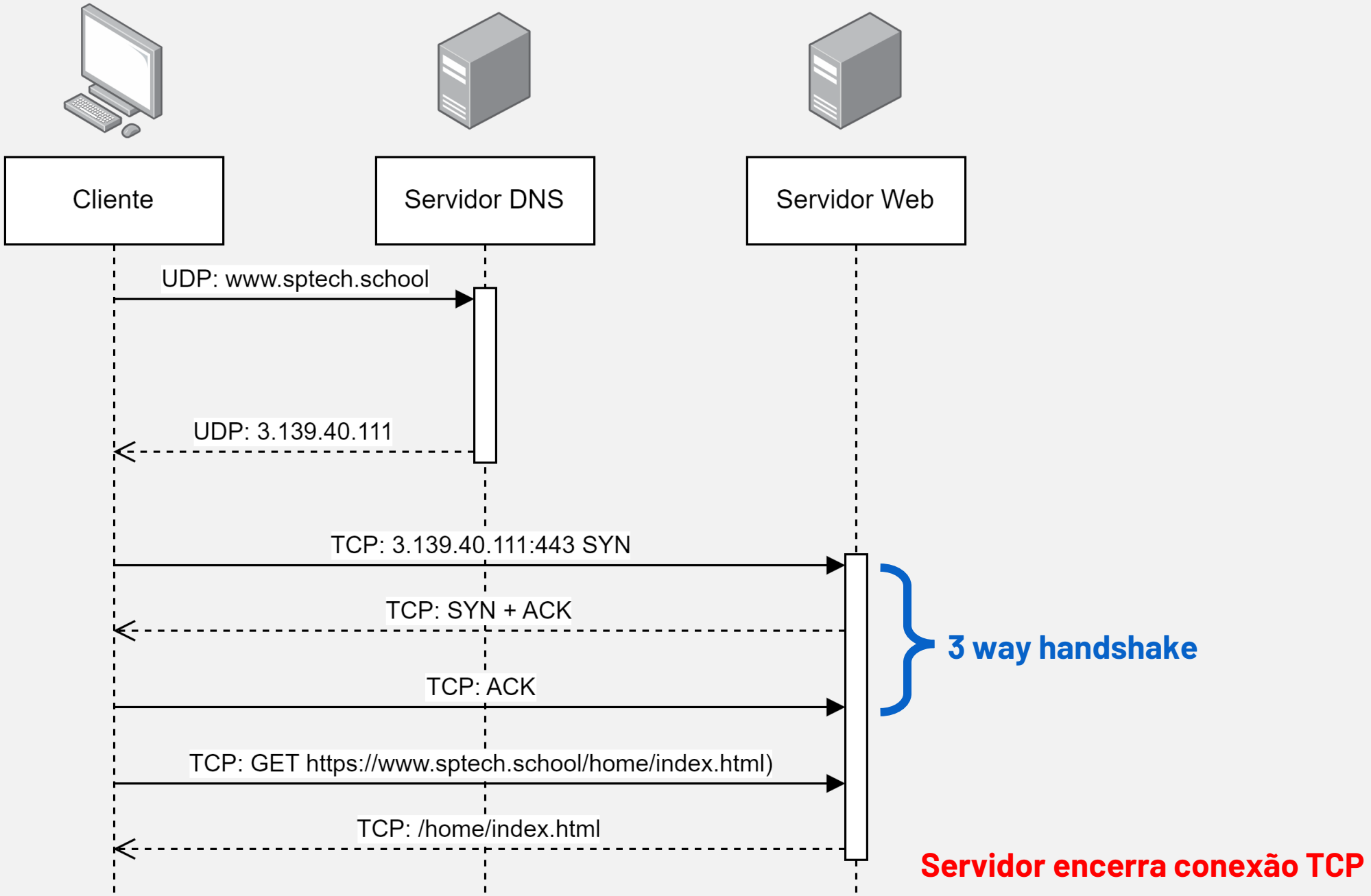
I. O navegador determina o URL
II. O navegador pergunta ao DNS qual é o endereço IP de www.itu.org
III. O DNS responde com 156.106.192.32
IV. O navegador estabelece uma conexão TCP com a porta 80 em 156.106.192.32
V. Em seguida, o navegador envia um comando solicitando o arquivo /home/index.html
VI. O servidor www.itu.org envia o arquivo /home/index.html
VII. A conexão TCP é encerrada
VIII. O navegador exibe todo o texto de /home/index.html
IX. O navegador busca e exibe todas as imagens que o arquivo contém

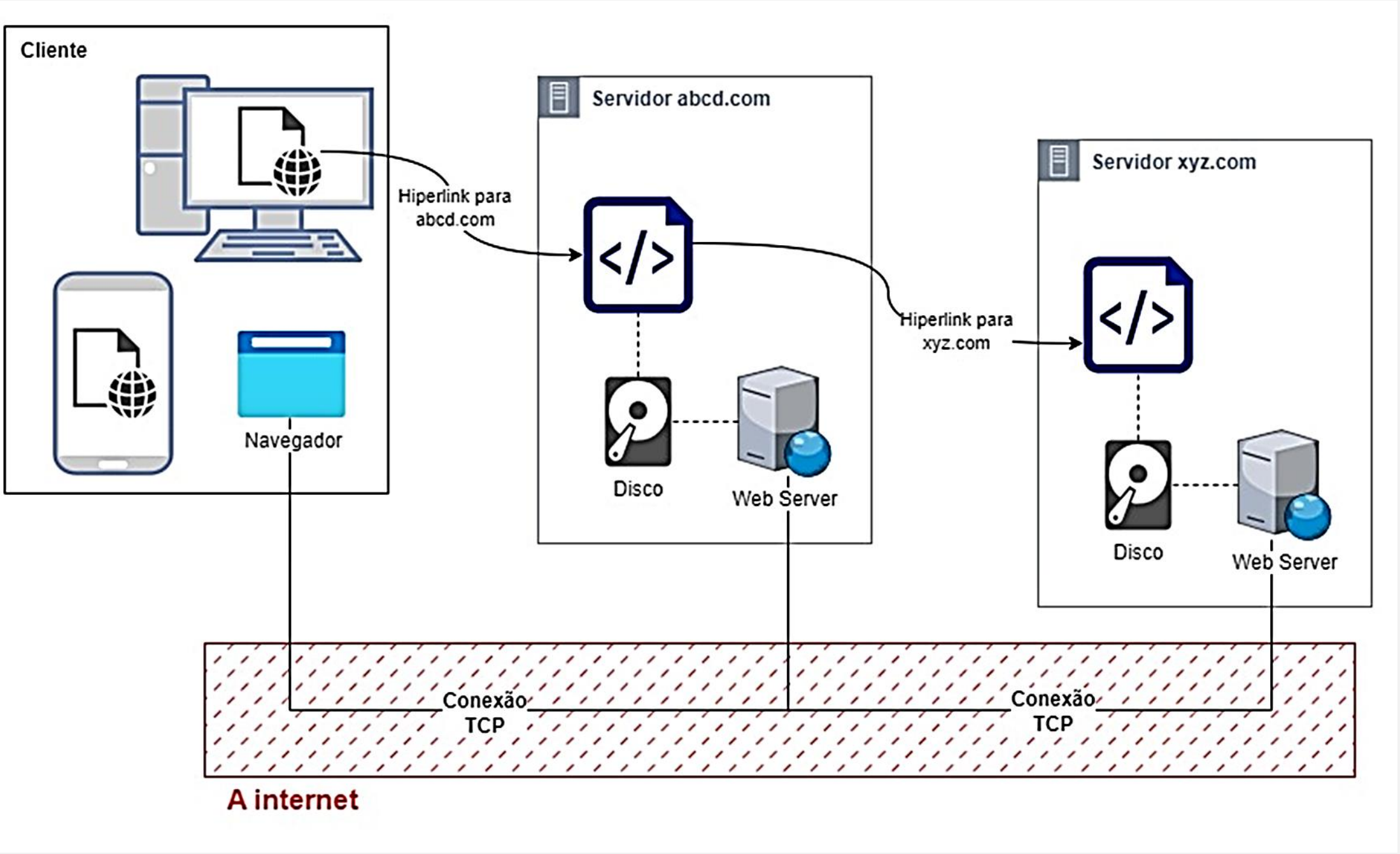
O lado servidor

Quando o usuário digita um URL ou clica em uma linha de hipertexto, o navegador analisa o URL e interpreta a parte entre http:// e a barra seguinte como um nome de DNS a ser pesquisado. Munido do endereço IP do servidor, o navegador estabelece uma conexão TCP para a porta 80 desse servidor. Em seguida, ele envia um comando contendo o restante do URL, que é o nome de um arquivo nesse servidor.

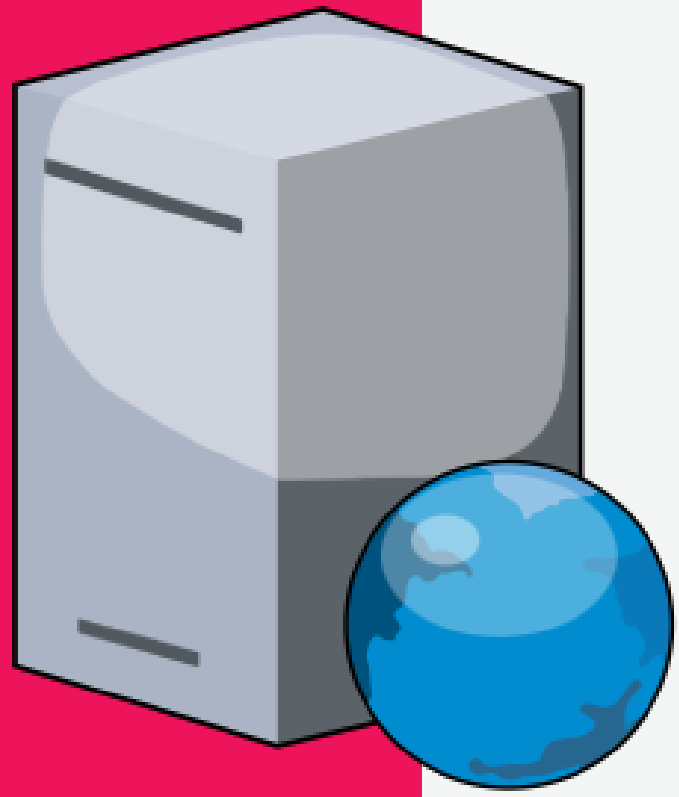
Em linhas gerais as etapas que o servidor executa em seu loop principal são:

I. Aceitar uma conexão TCP de um cliente (um navegador)
II. Obter o nome do arquivo solicitado
III. Obter o arquivo (do disco)
IV. Retornar o arquivo ao cliente
V. Encerrar a conexão TCP


Cliente
Servidor DNS
Servidor Web
UDP: www.sptech.school
UDP: 3.139.40.111
TCP: 3.139.40.111:443 SYN
TCP: SYN + ACK
TCP: ACK
3 way handshake
TCP: GET https://www.sptech.school/home/index.html)
TCP: /home/index.html
Servidor encerra conexão TCP


Cliente
Navegador
Servidor abcd.com
Hiperlink para
abcd.com
Disco
Web Server
Hiperlink para
xyz.com
Servidor xyz.com
Disco
Web Server
Conexão
TCP
Conexão
TCP
A internet

Escolhendo um servidor Web



Os servidores web de código aberto mais populares atualmente são Apache e Nginx. Juntos, eles são responsáveis por atender mais de 50% do tráfego da internet. Ambas as soluções são capazes de lidar com diversas cargas de trabalho e trabalhar com outros softwares.

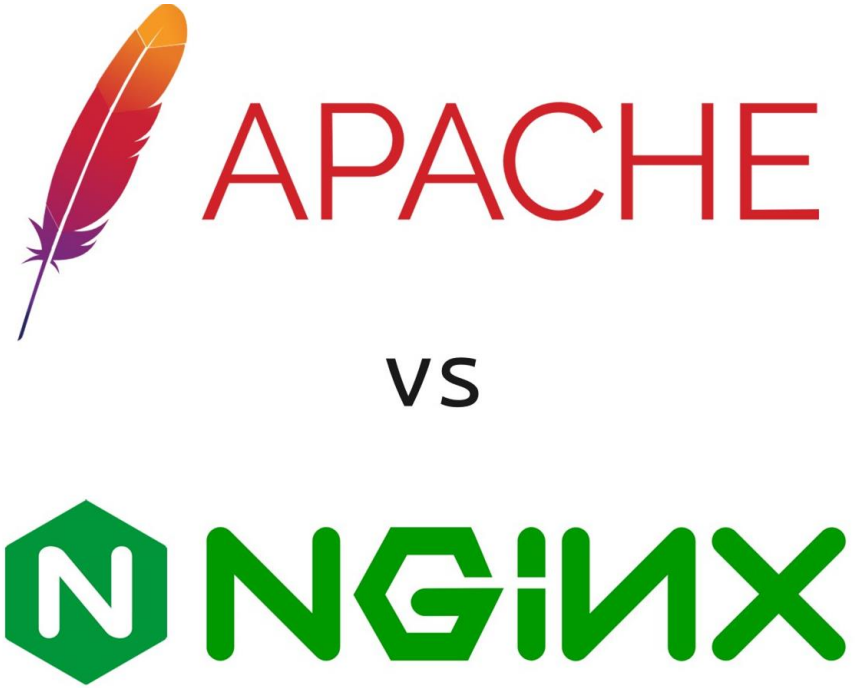
O Apache veio primeiro e foi construído em uma época em que era comum que vários sites com seus próprios arquivos de configuração individuais existissem em um único servidor web. O Nginx veio depois, em um momento em que as necessidades deixaram de servir vários sites de um servidor e passaram a servir um site de um servidor de maneira extremamente eficiente sob carga.

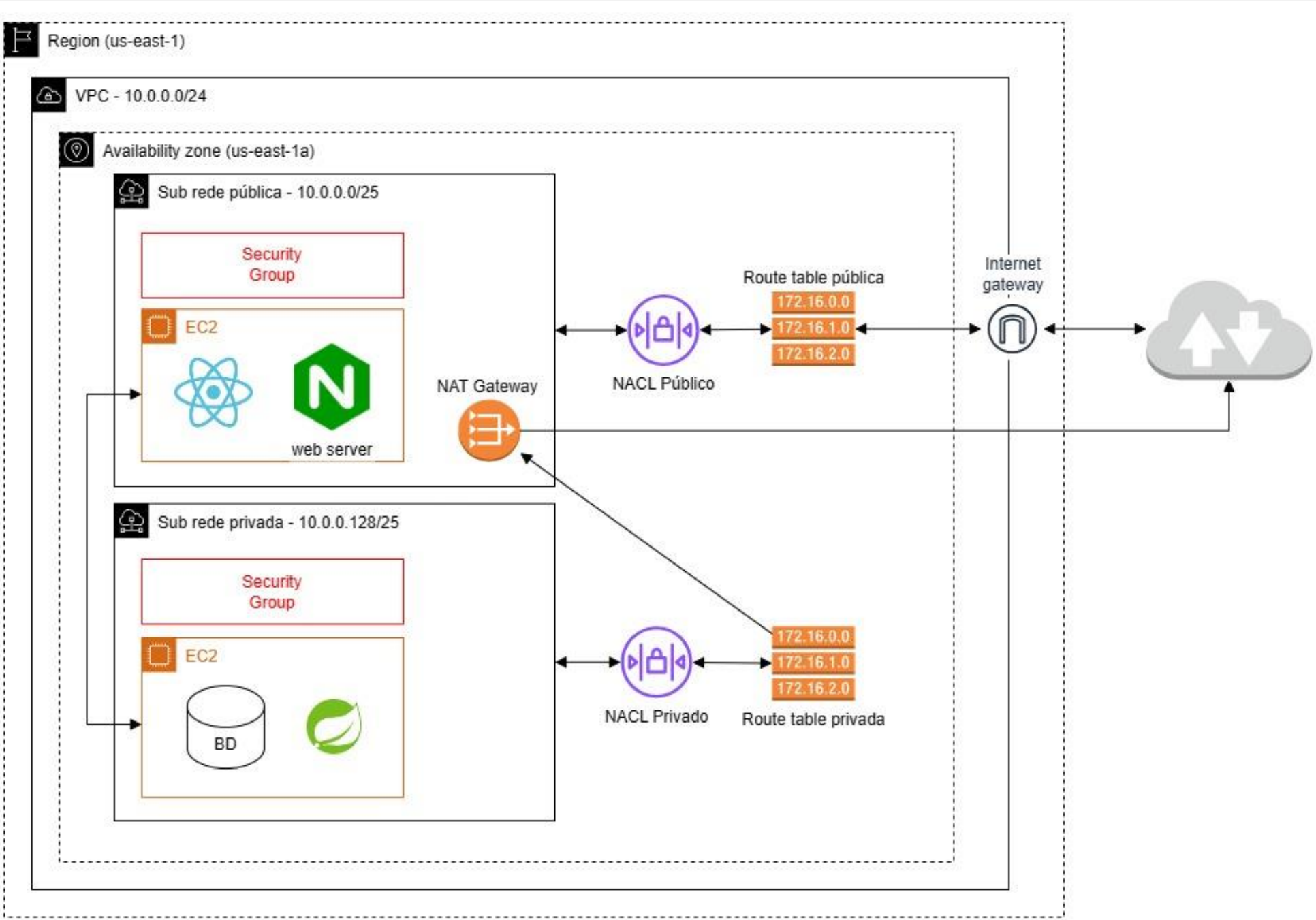
APACHE

- O servidor HTTP Apache foi criado por Robert McCool em 1995 e tem sido desenvolvido sob a direção da Apache Software Foundation desde 1999. Como o servidor web HTTP é o projeto original da fundação e é de longe o software mais popular, muitas vezes é referido simplesmente como “Apache”.
- O servidor web Apache foi o servidor mais popular na Internet pelo menos de 1996 a 2016. Devido a essa popularidade, o Apache se beneficia de excelente documentação e suporte integrado de outros projetos de software.
- O Apache é frequentemente escolhido pelos administradores por sua flexibilidade, potência e suporte quase universal. É extensível através de um sistema de módulos carregáveis dinamicamente e pode servir diretamente muitas linguagens de script, como PHP, sem a necessidade de software adicional.

- Em 2002, Igor Sysoev começou a trabalhar no Nginx como uma resposta ao problema C10K, que era um grande desafio para servidores web serem capazes de lidar com dez mil conexões simultâneas. O Nginx foi lançado publicamente em 2004 e atingiu esse objetivo ao contar com uma arquitetura assíncrona e orientada a eventos.
- Desde então, o Nginx ultrapassou o Apache em popularidade devido ao seu tamanho leve e à sua capacidade de escalar facilmente em hardware mínimo. O Nginx é excelente no fornecimento rápido de conteúdo estático, possui seu próprio sistema de módulos robusto e pode fazer proxy de solicitações dinâmicas para outros softwares, conforme necessário.
- O Nginx é frequentemente selecionado pelos administradores por sua eficiência de recursos e capacidade de resposta sob carga, bem como por sua sintaxe de configuração direta.

- O Nginx é muito mais leve que o Apache. O Apache deve criar um novo thread de processo para cada conexão. E embora possa processar 10 threads a uma velocidade comparável ao Nginx, quando é ampliado para centenas de conexões simultâneas, o Nginx assume uma liderança decisiva. O Nginx pode ser duas vezes mais rápido que o Apache ao servir conteúdo estático e muito menos intensivo em uso de CPU ao fazer isso.
- O Apache é mais configurável e está mais focado em ser um servidor web, possuindo alguns recursos úteis, como arquivos de configuração baseados em diretório e hosts virtuais. Isso torna muito fácil executar vários sites no mesmo servidor.
- Nginx e Apache suportam totalmente qualquer sistema Unix, incluindo FreeBSD. Embora o Nginx tecnicamente tenha uma versão que roda no Windows, não é a melhor. O Apache é totalmente compatível com Windows.




Region (us-east-1)
VPC - 10.0.0.0/24
Availability zone (us-east-1a)
Sub rede pública - 10.0.0.0/25
Security Group
EC2
web server
NAT Gateway
NACL Público
Route table pública
172.16.0.0
172.16.1.0
172.16.2.0
Internet gateway
Sub rede privada - 10.0.0.128/25
Security Group
EC2
BD
NACL Privado
Route table privada
172.16.0.0
172.16.1.0
172.16.2.0

Agradeço
a sua atenção!

SÃO
PAULO
TECH
SCHOOL