



SÃO
PAULO
TECH
SCHOOL

Arquitetura de soluções 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;

Velocidade: suas páginas da web devem carregar o mais rápido possível. Os usuários desejam que sua solicitação seja atendida imediatamente;

Simultaneidade: refere-se ao tratamento de várias solicitações recebidas ao mesmo tempo;

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;

Setup fácil: colocar um projeto em funcionamento rapidamente é fundamental para a iteração do seu projeto;

Documentação: os servidores Web são complexos. As configurações mais comuns ajudarão você a se recuperar rapidamente, mas suas necessidades aumentarão com o tempo.

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

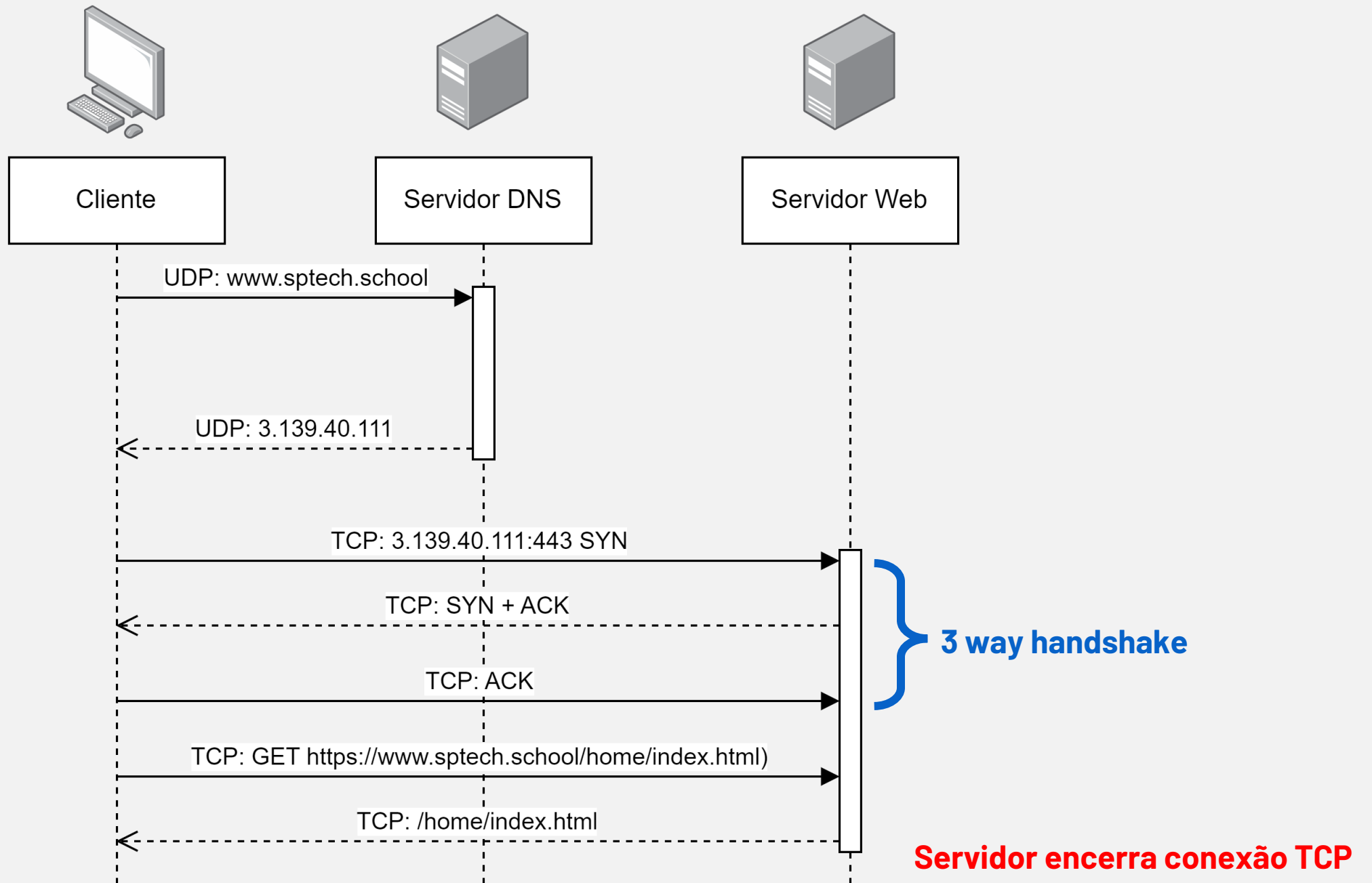
O lado servidor

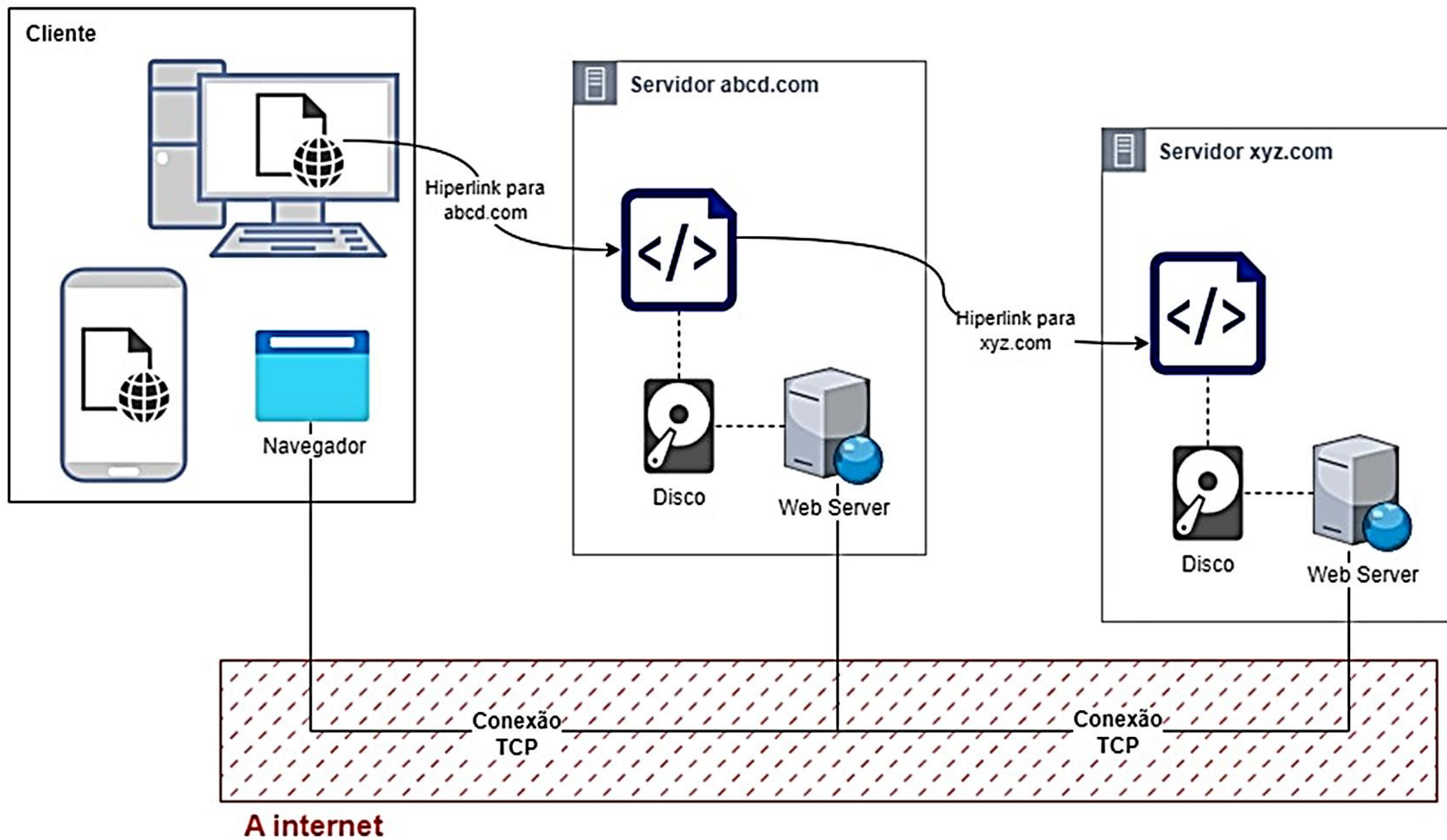
Quando o usuário digita um URL ou clica em uma linha de hipertexto, o navegador analisa o URL e interpreta a parte entre `http://` e a barra seguinte como um nome de DNS a ser pesquisado.

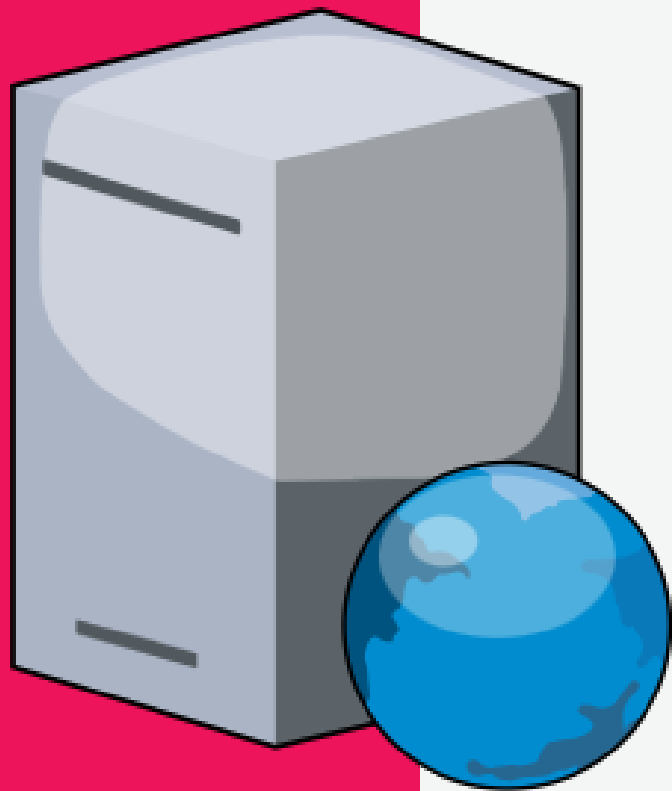
Munido do endereço IP do servidor, o navegador estabelece uma conexão TCP para a porta 80 desse servidor. Em seguida, ele envia um comando contendo o restante do URL, que é o nome de um arquivo nesse servidor.

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





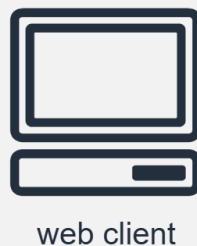


Escolhendo um servidor Web

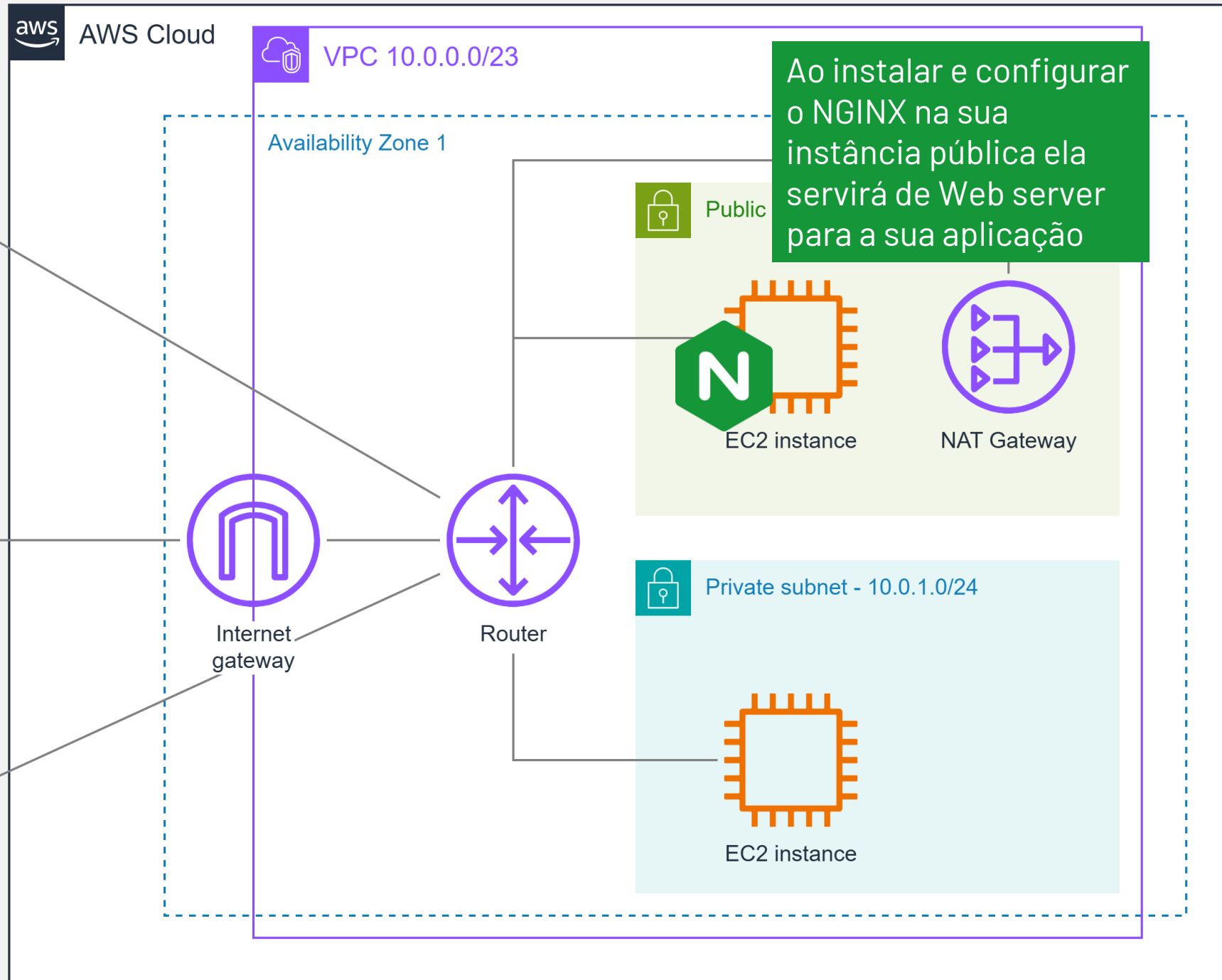
Os servidores web de código aberto mais populares atualmente são Apache e Nginx. Juntos, eles são responsáveis por atender mais de 50% do tráfego da internet. Ambas as soluções são capazes de lidar com diversas cargas de trabalho e trabalhar com outros softwares.

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.

Public route table	
Destination	Target
10.0.0.0/23	local
0.0.0.0/0	<i>igw-id</i>



Private route table	
Destination	Target
10.0.0.0/23	local
0.0.0.0/0	<i>nat-gateway-id</i>



Instalando e configurando o NGINX

Liberação das portas 80 e 8080 [http]

Vá em **Rede e segurança > Security groups**

Grupos de segurança (1/4) [Informações](#)

[Ações](#) [Exportar grupos de segurança para CSV](#) [Criar grupo de segurança](#)

<input type="checkbox"/>	Name	ID do grupo de segurança	Nome do grupo de segurança	ID da VPC	Descrição	Proprietário
<input checked="" type="checkbox"/>	-	sg-077ddcd31473214f5	lw-public01	ypc-0167a8d360959e406	launch-wizard-1 created 2025-03-05T1...	3446i
<input type="checkbox"/>	-	sg-08f3a15e3061ec938	default	ypc-0167a8d360959e406	default VPC security group	3446i
<input type="checkbox"/>	-	sg-0cf6e8c9ac23b206b	lw-private01	ypc-0167a8d360959e406	launch-wizard-1 created 2025-03-05T1...	3446i
<input type="checkbox"/>	-	sg-078b6b47c2a29feb8	default	ypc-066b9051804110561	default VPC security group	3446i

sg-077ddcd31473214f5 - lw-public01

[Regras de entrada](#) [Regras de saída](#) [Compartilhamento - novo](#) [Associações da VPC - novo](#) [Tags](#)

Regras de entrada (1)

<input type="checkbox"/>	Name	ID da regra do grup...	Versão do IP	Tipo	Protocolo	Intervalo de portas	Origem
--------------------------	------	------------------------	--------------	------	-----------	---------------------	--------

[Editar regras de entrada](#)

Liberação das portas 80 e 8080 [http]

EC2 > Grupos de segurança > sg-077ddcd31473214f5 - lw-public01 > Editar regras de entrada

Editar regras de entrada [Informações](#)

As regras de entrada controlam o tráfego de entrada que tem permissão para acessar a instância.

Regras de entrada [Informações](#)

ID da regra do grupo de segurança	Tipo Informações	Protocolo Informações	Intervalo de portas Informações	Origem Informações	Descrição - opcional Informações	
sgr-00c237c488bddce98	SSH ▼	TCP	22	Persona... ▼	<input type="text"/>	<input type="button" value="Excluir"/>
-	1 → HTTP ▼	TCP	80	Qualqu... ▼	<input type="text"/> 0.0.0.0/0 ✕	<input type="button" value="Excluir"/>
-	2 → TCP personalizado ▼	TCP	8080	Qualqu... ▼	<input type="text"/> 0.0.0.0/0 ✕	<input type="button" value="Excluir"/>

[Adicionar regra](#)

[Cancelar](#)

[Visualizar](#)

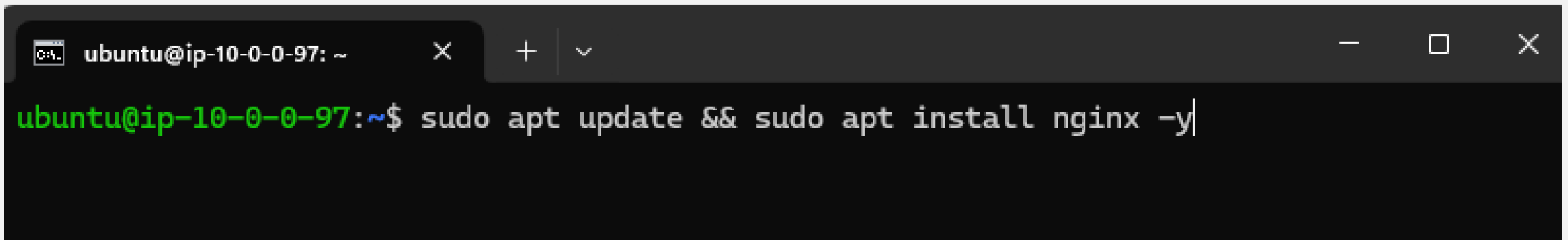
3 →

[Salvar regras](#)

Instalando o NGINX

Acesse via SSH sua instância e execute

```
$ sudo apt update && sudo apt install nginx -y
```

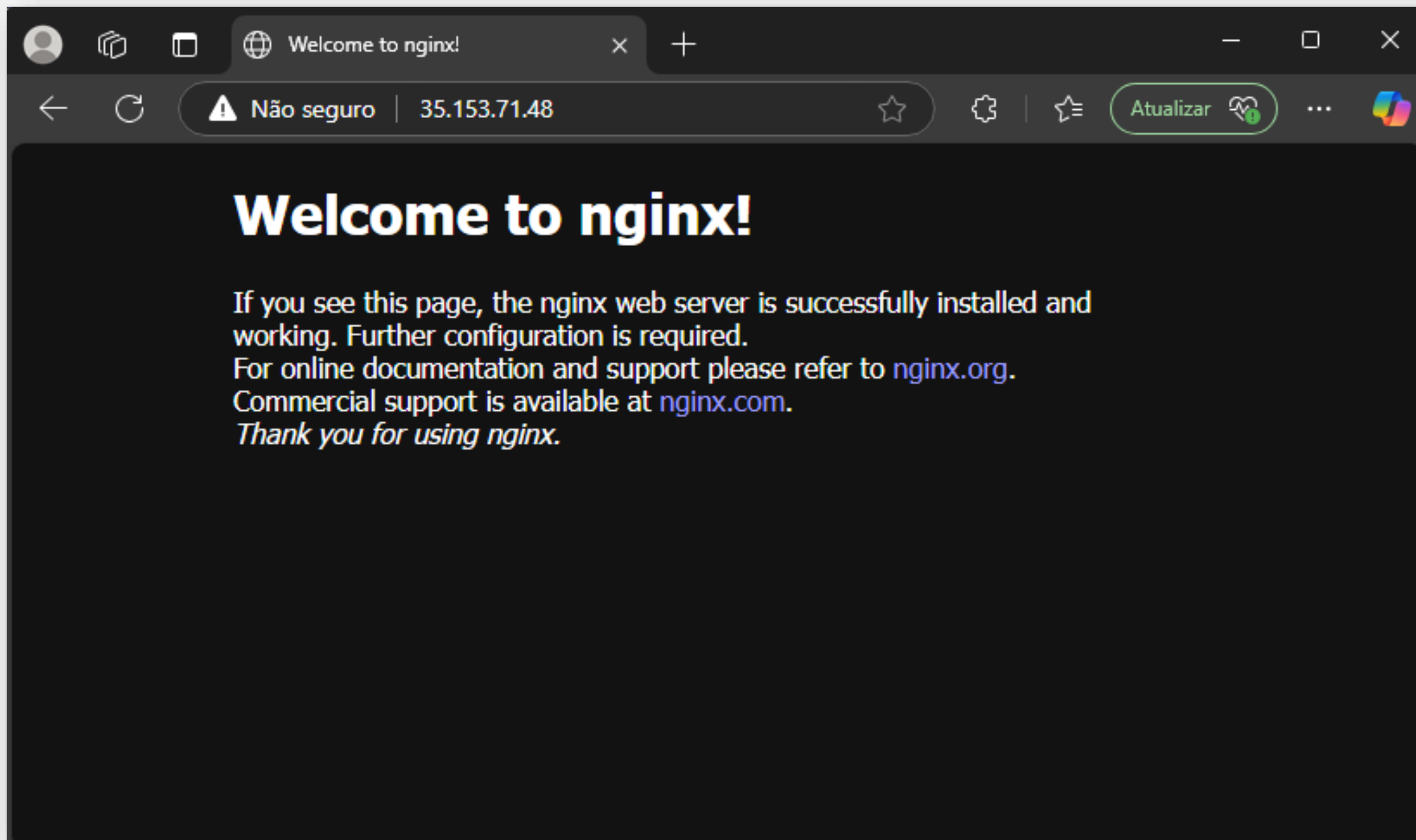
A terminal window with a dark background. The title bar shows a tab labeled 'ubuntu@ip-10-0-0-97: ~' with standard window controls (close, maximize, zoom). The terminal content shows the command 'ubuntu@ip-10-0-0-97:~\$ sudo apt update && sudo apt install nginx -y' being entered. The prompt 'ubuntu@ip-10-0-0-97:~\$' is in green, and the command is in white. A blue cursor is at the end of the command.

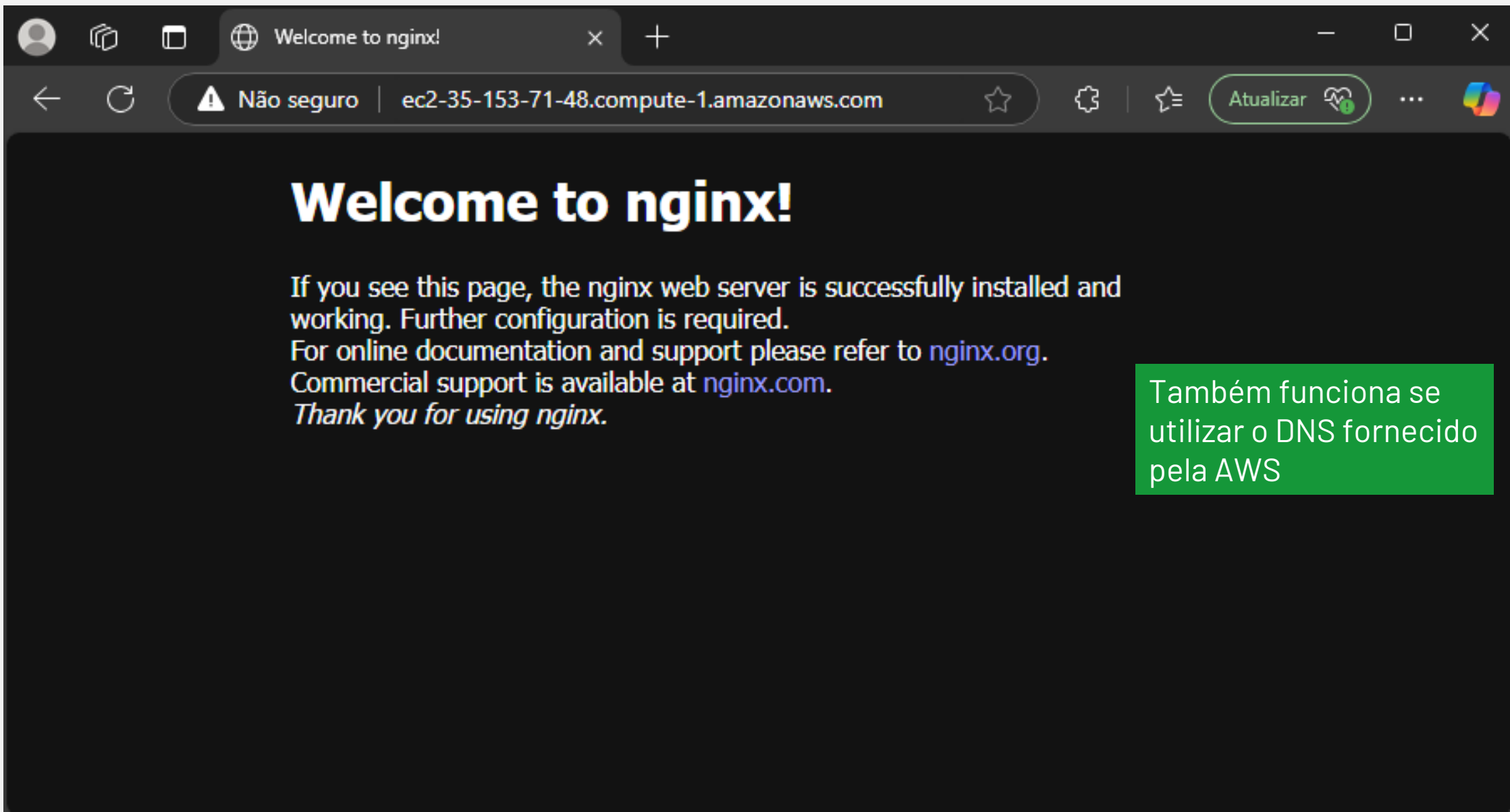
```
ubuntu@ip-10-0-0-97:~$ sudo apt update && sudo apt install nginx -y
```

Checando o status o NGINX

```
$ sudo systemctl status nginx
```

```
ubuntu@ip-10-0-0-97: ~  
● nginx.service - A high performance web server and a reverse proxy server  
   Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: enabled)  
   Active: active (running) since Wed 2025-03-05 18:13:00 UTC; 41s ago  
     Docs: man:nginx(8)  
  Process: 1930 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on; master_process on;  
  Process: 1932 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (code=ex  
Main PID: 1933 (nginx)  
   Tasks: 2 (limit: 1130)  
  Memory: 1.7M (peak: 1.9M)  
     CPU: 10ms  
   CGroup: /system.slice/nginx.service  
           └─1933 "nginx: master process /usr/sbin/nginx -g daemon on; master_proce  
             └─1934 "nginx: worker process"  
  
Mar 05 18:13:00 ip-10-0-0-97 systemd[1]: Starting nginx.service - A high performance  
Mar 05 18:13:00 ip-10-0-0-97 systemd[1]: Started nginx.service - A high performance w  
lines 1-16/16 (END)
```



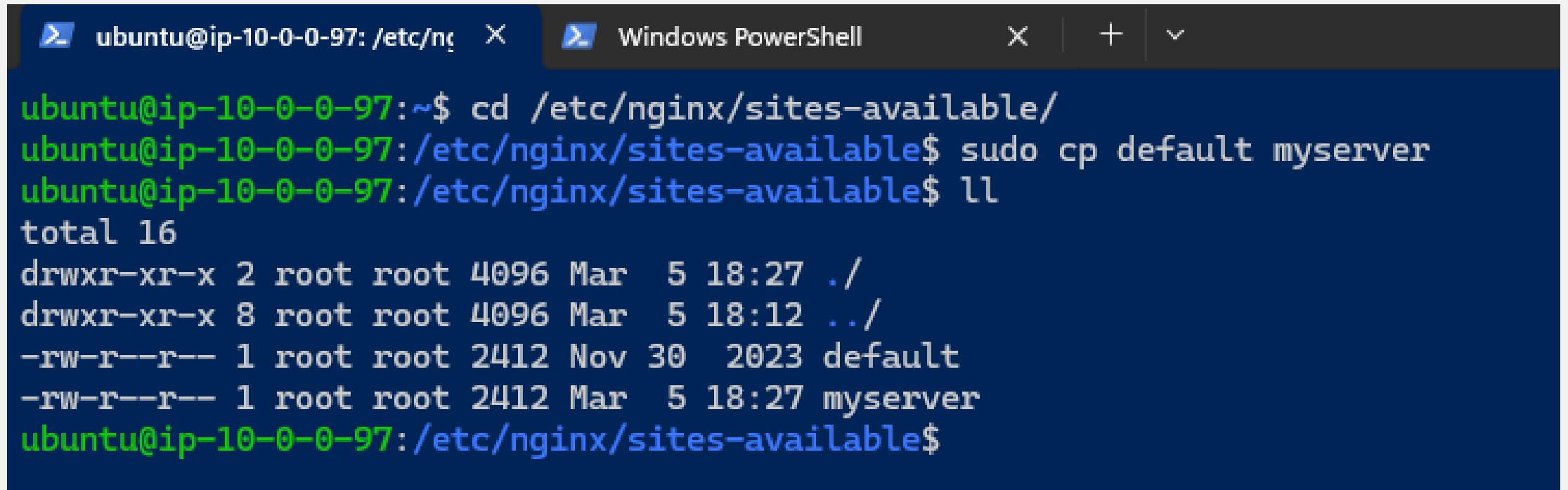
Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.
For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.
Thank you for using nginx.

Também funciona se
utilizar o DNS fornecido
pela AWS

Configurando o NGINX

```
$ cd /etc/nginx/sites-available/  
$ sudo cp default myserver  
$ sudo ln -s /etc/nginx/sites-available/myserver /etc/nginx/sites-enabled/  
$ sudo rm /etc/nginx/sites-available/default  
$ sudo rm /etc/nginx/sites-enabled/default
```



The image shows a terminal window with two tabs: 'ubuntu@ip-10-0-0-97: /etc/n...' and 'Windows PowerShell'. The terminal output shows the user navigating to the NGINX sites-available directory, copying the default configuration to 'myserver', and listing the directory contents. The directory listing shows the 'default' file being removed and 'myserver' being created.

```
ubuntu@ip-10-0-0-97:~$ cd /etc/nginx/sites-available/  
ubuntu@ip-10-0-0-97:/etc/nginx/sites-available$ sudo cp default myserver  
ubuntu@ip-10-0-0-97:/etc/nginx/sites-available$ ll  
total 16  
drwxr-xr-x 2 root root 4096 Mar  5 18:27 ./  
drwxr-xr-x 8 root root 4096 Mar  5 18:12 ../  
-rw-r--r-- 1 root root 2412 Nov 30  2023 default  
-rw-r--r-- 1 root root 2412 Mar  5 18:27 myserver  
ubuntu@ip-10-0-0-97:/etc/nginx/sites-available$
```

Configurando o NGINX

```
html --> /var/www/html
```

E o node? -> caso tenha utilizado o node, será necessário uma configuração de proxy reverso no NGINX. E o servidor Node rodando

Configurar tudo no arquivo **myserver**

```
server {
    listen 80 default_server;
    listen [::]:80 default_server;
    server_name localhost;

    location / {
        root /var/www/html/public/;
        index index.html index.htm index.nginx-debian.html;
        try_files $uri $uri/ =404;

        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header Host $host;
        proxy_set_header X-NginX-Proxy true;
        proxy_pass http://localhost:3000/;
    }
    location /login {
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header Host $host;
        proxy_set_header X-NginX-Proxy true;
        proxy_pass http://localhost:3000/login;
    }
}
```

```
server {  
    listen 80 default_server;  
    listen [::]:80 default_server;  
    server_name localhost;  
  
    location / {  
        root /var/www/html/public/;  
        index index.html index.htm index.nginx-debian.html;  
        try_files $uri $uri/ =404;  
  
        proxy_set_header X-Real-IP $remote_addr;  
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;  
        proxy_set_header Host $host;  
        proxy_set_header X-NginX-Proxy true;  
        proxy_pass http://localhost:3000/;  
    }  
    location /login {  
        proxy_set_header X-Real-IP $remote_addr;  
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;  
        proxy_set_header Host $host;  
        proxy_set_header X-NginX-Proxy true;  
        proxy_pass http://localhost:3000/login;  
    }  
}
```

Este é um arquivo exemplo!
Precisa ser ajustado conforme
sua aplicação!

No diretório **root** é onde ficam os
arquivos html, neste caso aqui
tinha uma pasta **public** com
todos os arquivos

Como temos o Node como
servidor, criamos um **proxy**
reverso para encaminhar as
requisições para uma porta
específica

Para cada endpoint criado que
não tenha a mesma raiz se faz
necessário um **location**. Caso
tenha uma raiz como /api/ só
precisaria de um location

Configurando o NGINX

```
$ sudo systemctl restart nginx
```

Reiniciar o serviço do NGINX, inicializar o servidor Node e testar o acesso ao site!

Agradeço
a sua atenção!



SÃO
PAULO
TECH
SCHOOL