CENTROS DE PROCESSAMENTO DE DADOS

2020/2021

Gonçalo Miguel Conceição Vicente[1]

[1] 2172131@my.ipleiria.pt, Engenharia Informática, Diurno

Resumo

O presente documento descreve os trabalhos desenvolvidos no âmbito da unidade curricular Centros de Processamento de Dados pertence à licenciatura em Engenharia Informática e descreve a configuração e administração do serviço HTTP.

Para este trabalho foi utilizado o software GNS3, para a implementação de todo o cenário. Existem 3 redes distintas, uma delas (10.0.0.0/24) onde é simulado uma zona Área Apache2, que é composta por dois computadores com o sistema operativo Windows 7, um VPCS e um servidor HTTP Apache2. Na zona Área Nginx existe a rede 192.168.1.0/24, está alojado um servidor HTTP Nginx, um computador com o sistema operativo Windows 7 e um VPCS. Existe também a rede 172.16.1.0/30 que interliga as duas redes anteriormente mencionadas.

Palavras-chave: HTTP, Apache2, Nginx, GNS3.

1. Introdução inicial

O HTTP, Hypertext Transfer Protocol, é um protocolo de comunicação base para a comunicação de dados WWW, World Wide Web, sendo responsável pelo tratamento de pedidos e respostas entre clientes e servidores. Foi lançado em 1990 com a primeira versão, chamada por HTTP/0.9 onde era um protocolo muito simples para transferência de dados no formato de texto ASCII. Entre 1992 e 1996, e tendo em vista a necessidade de não transferir apenas texto foi desenvolvido o HTTP/1.0, passando assim a transferir mensagens do tipo MIME44. Em 1997 surge a versão HTTP/1.1 onde foram implementadas, entre outras, o uso de conexões persistentes e uso de servidores de proxy. Em março de 2015 foi divulgado o lançamento de uma nova versão, HTTP/2 que deixará o browser com o tempo de resposta ao pedido melhor e mais seguro. Uma nova versão, HTTP/3 encontra-se em fase de testes por grandes empresas.

O presente documento encontra-se dividido em 2 capítulos, que apresentam o processo de pesquisa, desenvolvimento e testes do projeto.

No primeiro capítulo é abordado o serviço implementado, quais as suas funcionalidades bem como as soluções que existem para se poder utilizar o serviço HTTP. Ao longo deste capítulo é documentado todo o processo de desenvolvimento, desde as opções tomadas a imagens exemplificativas de algumas configurações que foram realizadas.

O segundo capítulo é direcionado para os testes, onde são apresentados exemplos que comprovam o correto funcionamento do projeto desenvolvido.

2. Serviço HTTP

O que é e o que faz?

O serviço HTTP é um protocolo de comunicação que é o responsável pelos pedidos e respostas entre os servidores e os clientes, sendo a base para o WWW. Até aos dias de hoje existiram 4 versões em pleno funcionamento, sendo que a quinta versão ainda se encontra em fase de testes. Na tabela 1 é possível visualizar os anos de lançamento das diferentes versões do protocolo HTTP.

Ano	Versão HTTP
1990	0.9
1996	1.0
1997	1.1
2015	2.0

2018	3.0

Tabela 1 - Versões do HTTP por ano de lançamento

Nginx vs Apache2

O Nginx, lançado em 2004, e o Apache, lançado em 1995, são servidores web bastante populares que servem para responder aos pedidos dos seus clientes, maioritariamente browsers, com páginas web.

Ambos os servidores são utilizados por grandes empresas no mundo e, durante muitos anos foi o Apache a ser o servidor web mais utilizado, mas a fatia de mercado do Nginx tem vindo a crescer bastante ao longo dos últimos anos, uma vez que, em alguns casos este apresenta uma vantagem competitiva em relação ao seu desempenho.

Uma grande vantagem do Apache deve-se ao seu sistema de configuração e ao ficheiro .htaccess, uma vez que é um ficheiro que oferece uma grande flexibilidade, além de que cada pasta pode ter o seu próprio ficheiro com as suas próprias configurações o que, para sistemas que fornecem alojamento é uma grande vantagem uma vez que uma máquina pode servir para vários utilizadores onde cada qual pode configurar o seu website sem interferir com a configuração global do servidor.

O Nginx não possui um sistema de configurações como o do Apache e, apesar de ser mais eficiente e rápido acaba por não ser implementado em sistemas que fornecem alojamento. Por outro lado, ao não permitir configurações por diretoria apresenta uma vantagem em relação ao Apache.

No gráfico seguinte é possível visualizar que o Nginx se tem vindo a aproximar bastante do Apache ao longo dos últimos anos.

.

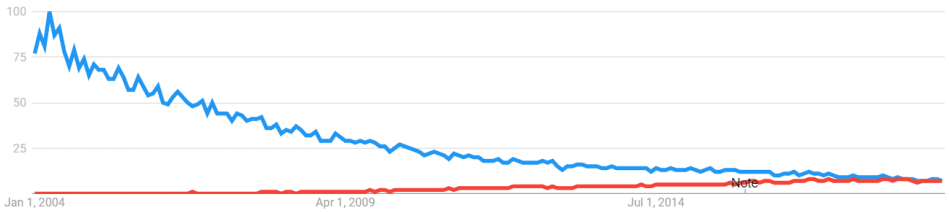

Figura 1 - Apache vs Nginx

Cenário apresentado

Na elaboração deste trabalho foi desenvolvido o serviço HTTP, *Hypertext Transfer Protocol*, que é um protocolo de comunicação e é a base para a comunicação através do WWW. De forma a implementar este serviço, utilizou-se dois servidores de HTTP distintos: Nginx e Apache2. Para simular que estes servidores se encontravam em áreas locais diferentes, existiu a necessidade de configurar 2 routers, interligados entre si. A figura 2 mostra o cenário do projeto implementado, e na tabela 2 podemos ver em detalhe o esquema de endereçamento.

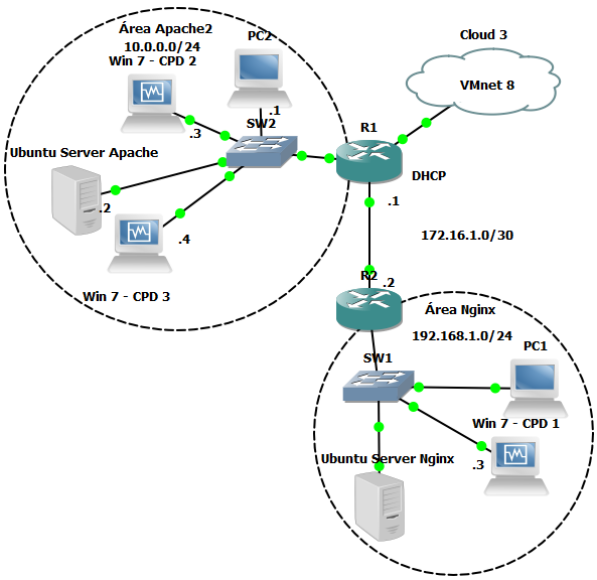

Figura 2 - Cenário do projeto

	Equipamento	Interface	IP	Máscara de rede	Gateway
Área Apache2	RApache2	fa0/0	172.16.1.1	255.255.255.252	
		fa1/0	DHCP	DHCP	
		fa0/1	10.0.0.254	255.255.255.0	
	PC2	e0	10.0.0.1	255.255.255.0	10.0.0.254
	Win 7 - CPD 2	e0	10.0.0.2	255.255.255.0	10.0.0.254
	Ubuntu Server Apache	e0	10.0.0.2	255.255.255.0	10.0.0.254
		e0	10.0.0.8	255.255.255.0	10.0.0.254
	Win 7 - CPD 3	e0	10.0.0.4	255.255.255.0	10.0.0.254
Área Nginx	RNginx	fa0/0	172.16.1.2	255.255.255.252	
		fa0/1	192.168.1.24	255.255.255.0	192.168.1.24
	Ubuntu Server Nginx	e0	192.168.1.1	255.255.255.0	192.168.1.24
	PC1	e0	192.168.1.2	255.255.255.0	192.168.1.24
	Win 7 - CPD 1	e0	192.168.1.3	255.255.255.0	192.168.1.24

Tabela 2 - Esquema de endereçamento

Nginx

No cenário do projeto é possível verificar que uma das áreas é destinada ao servidor HTTP Nginx, onde foi desenvolvida uma máquina virtual Linux, com o sistema operativo Ubuntu Server 18.04 LTS, de forma a ser possível instalar este servidor e fazer todas as configurações para o seu correto funcionamento.

Para que um servidor HTTP possa alojar mais do que um website, existe um mecanismo que permite que tal seja possível, recorrendo ao uso de blocos do servidor. Assim, de forma a demonstrar as potencialidades deste mecanismo, foram configuradas 4 websites num único servidor Nginx [1].

Quando se acede via browser ao IP 192.168.1.1, o utilizador é automaticamente redirecionado para uma página que utiliza a versão segura do protocolo HTTP, o protocolo HTTPS, *Hyper Text Transfer Protocol Secure*, onde os dados são transmitidos através de uma ligação criptografada e onde a autenticidade do servidor e do cliente são verificadas através de certificados digitais [2] [3]. Ao ser escolhido o porto a que o utilizador se quer ligar, por exemplo, 192.168.1.1:8080, o utilizador tem acesso a um website que utiliza a versão segura do protocolo HTTP, porém a ligação é feita através de um porto diferente do convencional. Por exemplo, através do IP 192.168.1.1:7080 é efetuada uma ligação HTTPS e são pedidas credenciais de acesso ao website [4]. Existe ainda a possibilidade de aceder a um outro website através do IP 192.168.1.1:9090, onde é usando apenas o protocolo HTTP.

Uma vez que a administração de um servidor web não se trata apenas da configuração inicial dos serviços, e num servidor web, como o Nginx, os logs tem informações muito importantes sobre cada tentativa de acesso aos recursos que são disponibilizados pelo servidor. Tendo em conta esta situação, foi configurado diferentes ficheiros de log de acordo com o website em questão.

```
server {
    listen 443 ssl;
    listen [::]:443 ssl;
    include snippets/self-signed.conf;
    include snippets/ssl-params.conf;

    root /var/www/html/website1;

    index index.html;

    server_name example.com www.example.com;

    access_log /var/log/nginx/example.com.access.log timed;
    error_log /var/log/nginx/example.com.error.log;

    location / {
        try_files $uri $uri/ =404;
    }
}

server {
    listen 80;
    listen [::]:80;

    server_name example.com www.example.com;

    return 301 https://192.168.1.1$request_uri;
}
```

```
server {
    listen 7080 ssl;
    listen [::]:7080 ssl;
    include snippets/self-signed.conf;
    include snippets/ssl-params.conf;

    root /var/www/html/website4;

    index index.html;

    server_name example.com www.example.com;

    access_log /var/log/nginx/website4.access.log;
    error_log /var/log/nginx/website4.error.log;

    location / {
        try_files $uri $uri/ =404;
        auth_basic "Restricted Content";
        auth_basic_user_file /etc/nginx/.htpasswd;
    }
}
```

```
server {
    listen 9090;
    listen [::]:9090;

    root /var/www/html/website3;

    index index.html;

    server_name example.com www.example.com;

    access_log /var/log/nginx/website3.access.log;
    error_log /var/log/nginx/website3.error.log;

    location / {
        try_files $uri $uri/ =404;
    }
}
```

```
server {
    listen 8080 ssl;
    listen [::]:8080 ssl;
    include snippets/self-signed.conf;
    include snippets/ssl-params.conf;

    root /var/www/html/website2;

    index index.html;

    server_name example.com www.example.com;

    access_log /var/log/nginx/website2.access.log;
    error_log /var/log/nginx/website2.error.log;

    location / {
        try_files $uri $uri/ =404;
    }
}
```

Figura 3 - Configurações no servidor Nginx

Apache2

No cenário do projeto, além de uma área destinada ao Nginx, existe também uma outra área direcionada para o servidor HTTP, Apache2. Neste cenário, recorreu-se também ao desenvolvimento de uma outra máquina virtual com o sistema operativo Linux, Ubuntu Server 18.04 LTS, para que seja possível implementar da melhor forma um servidor HTTP Apache2.

Também no Apache2 existe a possibilidade que um único servidor possa alojar vários websites, tal como acontece no Nginx. Assim, de forma a demonstrar que também é uma ferramenta existente, foram implementados 4 websites no servidor Apache2 do cenário [5].

Desta forma, quando o utilizador acede ao IP 10.0.0.2, sem destinar qual o porto em que deseja fazer a ligação, o servidor irá utilizar a versão segura do protocolo HTTP, de forma a fazer uma ligação segura, além de que este pedido irá usar a versão HTTP/2, uma vez que também foi configurado neste servidor [6]. Porém, existe também a possibilidade de o utilizador aceder ao mesmo website tendo apenas de informar qual o porto em que deseja fazer a ligação, bastando então digitar 10.0.0.2:80. Este servidor apresenta também configurada um página pessoal, acessível tanto por HTTP como por HTTPS, neste caso para o utilizador ubuntu, sendo para tal necessário inserir no browser 10.0.0.2/~ubuntu [7]. Em semelhança ao servidor Nginx, existe também uma área reservada a diferentes utilizadores, sendo que para tal é necessário que o utilizador se autentique através de um login e password [8].

Uma vez que este servidor possui dois endereços IP diferentes, quando o utilizador acede ao IP 10.0.0.8, o browser devolve um website daquele que seria devolvido pelo IP 10.0.0.2, que corresponde ao mesmo servidor. Esta situação é possível, uma vez que o Apache2 tem a possibilidade de apresentar diferentes websites consoante o endereço IP que o utilizador está a solicitar [9].

```
Listen 8080
<VirtualHost *:8080>
        ServerAdmin webmaster@localhost
        DocumentRoot /var/www/html/website2
        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log combined

 <Directory "/var/www/html/website2">
   AuthType Basic
   AuthName "Restricted Content"
   AuthUserFile /etc/apache2/.htpasswd
   Require valid-user
 </Directory>

</VirtualHost>
```

```
<IfModule mod_ssl.c>
        <VirtualHost _default_:443>
                ServerAdmin webmaster@localhost
                Protocols h2 http/1.1
                DocumentRoot /var/www/html/website1/

                ErrorLog ${APACHE_LOG_DIR}/error.log
                CustomLog ${APACHE_LOG_DIR}/access.log combined

                SSLEngine on
                SSLCertificateFile      /etc/ssl/certs/apache-selfsigned.crt
                SSLCertificateKeyFile /etc/ssl/private/apache-selfsigned.key

                <FilesMatch "\.(cgi|shtml|phtml|php)$">
                                SSLOptions +StdEnvVars
                </FilesMatch>
                <Directory /usr/lib/cgi-bin>
                                SSLOptions +StdEnvVars
                </Directory>
        </VirtualHost>
</IfModule>
```

```
Listen 80
 <VirtualHost 10.0.0.8>
        ServerAdmin webmaster@localhost
        DocumentRoot /var/www/html/website3
        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

```
 <VirtualHost *:80>
        ServerAdmin webmaster@localhost
        DocumentRoot /var/www/html/website1
        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

```
<IfModule mod_userdir.c>
        UserDir public_html
        UserDir disabled root

        <Directory /home/*/public_html>
                AllowOverride FileInfo AuthConfig Limit Indexes
                Options MultiViews Indexes SymLinksIfOwnerMatch IncludesNoExec
                Require method GET POST OPTIONS
        </Directory>
</IfModule>
```

Figura 4 - Configurações no servidor Apache2

Ligações entre routers

Para simular que ambos os servidores se encontravam em áreas locais diferentes e para que os computadores da rede local do Nginx conseguissem ter acesso aos websites disponibilizados pelo servidor Apache2 e vice-versa foi necessário proceder à configuração de dois routers.

De forma a interligar os dois routers foi utilizada a rede 172.16.1.0/30, ficando atribuído o IP 172.16.1.1 ao router correspondente à área Apache2 e o endereço IP 172.16.1.2 ao router da área Nginx. Foi ainda utilizado o protocolo RIPv2, que foi o protocolo que melhor se ajustava às necessidades da rede, uma vez que é uma rede simples composta apenas por dois routers.

Na figura 5, estão apresentadas as principais configurações implementadas nos dois routers do cenário.

```
hostname RApache2

interface FastEthernet0/0
 ip address 172.16.1.1 255.255.255.252
 duplex auto
 speed auto

interface FastEthernet0/1
 ip address 10.0.0.254 255.255.255.0
 duplex auto
 speed auto

interface FastEthernet1/0
 ip address dhcp
 duplex auto
 speed auto

router rip
 version 2
 network 10.0.0.0
 network 172.16.0.0
 default-information originate
```

```
hostname RNginx

interface FastEthernet0/0
 ip address 172.16.1.2 255.255.255.252
 duplex auto
 speed auto

interface FastEthernet0/1
 ip address 192.168.1.254 255.255.255.0
 duplex auto
 speed auto

router rip
 version 2
 network 172.16.0.0
 network 192.168.1.0
 no auto-summary
```

Figura 5 - Configurações principais nos routers

3. Testes e resultados

De forma a demonstrar que as configurações implementadas surtiram efeito, podemos analisar através das figuras seguintes que tudo está a funcionar como suposto.

Começando por analisar os resultados do servidor de Nginx, figura 6, onde podemos verificar que ao inserirmos o endereço IP 192.168.1.1 será feita uma ligação HTTPS, sendo que apenas está assinalado a vermelho pelo facto de o certificado gerado não ter sido emitido por entidade certificadora.

Figura 6 - Website 1 do servidor Nginx

Na figura 7, podemos verificar que ao inserimos o IP anterior mas a definirmos o porto, ou seja, 192.168.1.1:8080, a ligação é feita utilizando a versão segura do HTTP apesar de não ser feito no porto por defeito, o porto 443.

Figura 7 - Website 2 do servidor Nginx

Na figura 8, é feito um pedido ao servidor através do endereço 192:168.1.1:9090 e a ligação é feita utilizando apenas o protocolo HTTP.

Figura 8 - Website 3 do servidor Nginx

Por fim, no servidor Nginx, é feito um pedido através do endereço 192.168.1.1:7080 podemos ver que em primeiro lugar o é apresentada ao utilizador uma janela onde é necessário inserir um nome de utilizador e password (figura 10) válidos para que tenham acesso ao website que pretendem aceder. Caso as credencias inseridas sejam válidas serão então redirecionados para o website disponivel (figura 9).

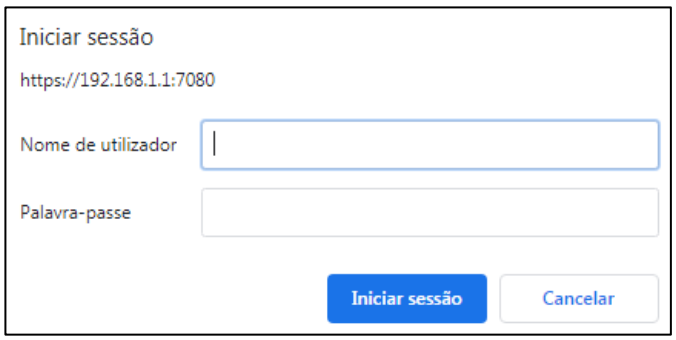

Figura 10 - Janela para inserção de credencias

Figura 9 - Website 4 do servidor Nginx

Analisando o servidor Apache2, podemos verificar que ao inserir o endereço IP 10.0.0.2 sendo que quando o pedido é HTTP o protocolo utilizado é o HTTP/1.1, figura 13, porém quando o pedido é HTTPS o protocolo utilizado é o HTTP/2 figura (11).

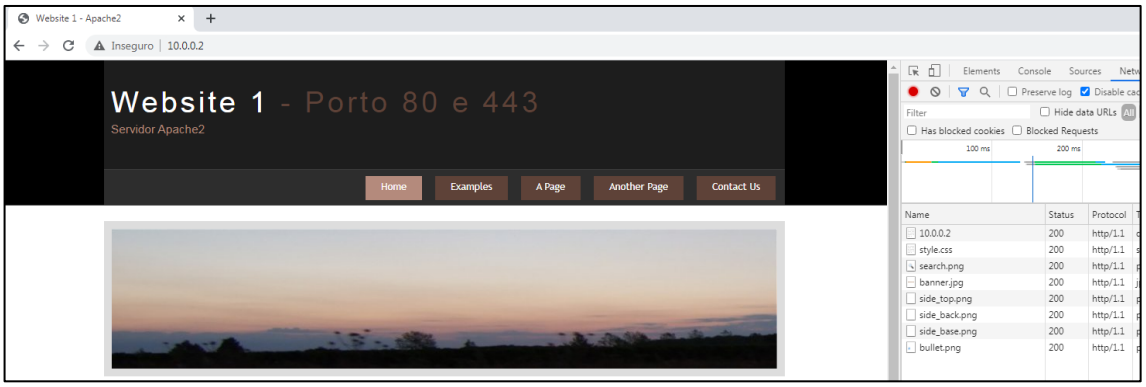

Figura 11 - Website 1 do servidor Apache2 (HTTP)

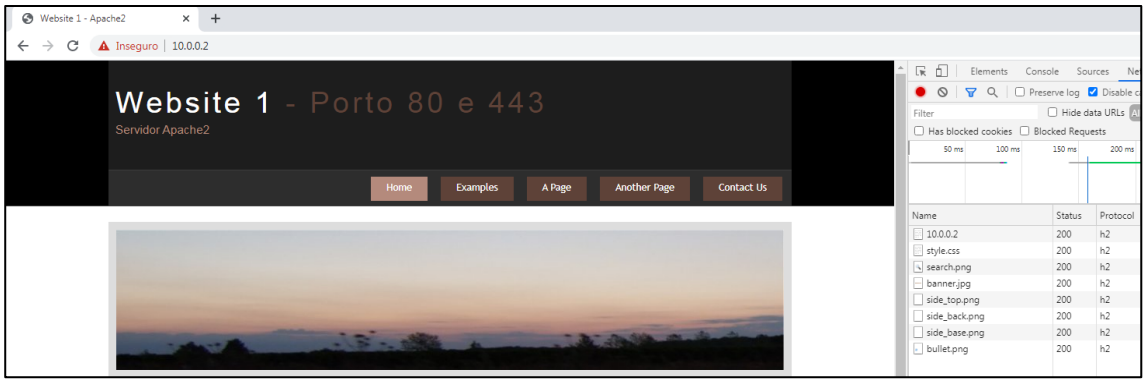

Figura 12 - Website 1 do servidor Apache2 (HTTPS)

Na figura 12, quando o pedido é feito através do IP e do porto, ou seja, 10.0.0.2:8080, é utilizado o protocolo HTTP, num porto diferente do porto definido por omissão, ou seja, o porto 80.

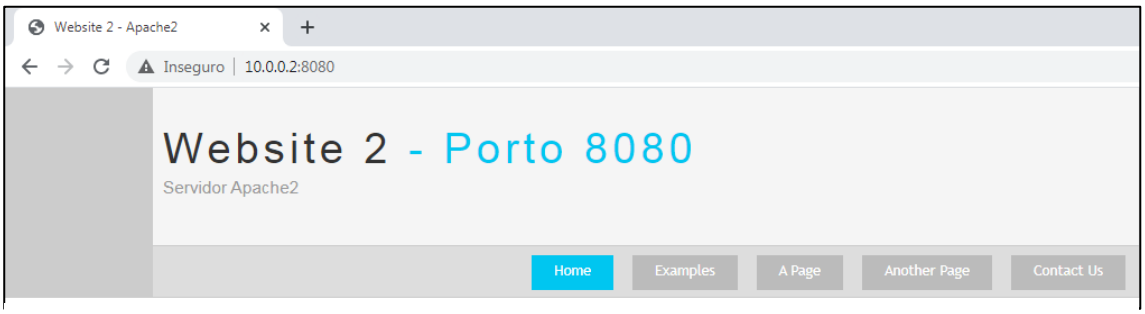

Figura 13 - Website 2 do servidor Apache2

No servidor Apache2, existe a particularidade da placa de rede possuir 2 endereços IP, o 10.0.0.2 e o 10.0.0.8. Assim, acedendo ao endereço IP 10.0.0.8 é devolvido, através de HTTP, o website demonstrado na figura 14.

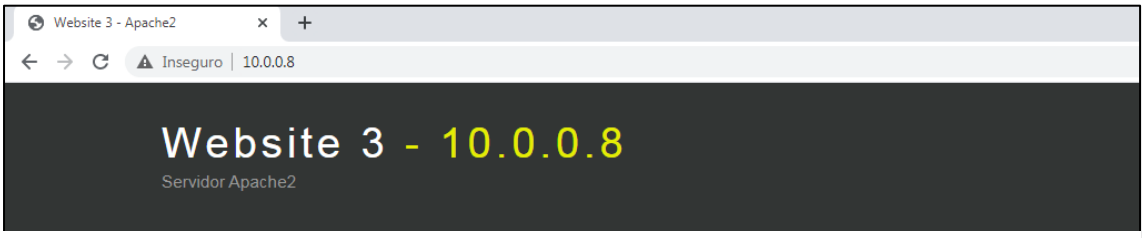

Figura 14 - Website 3 do servidor Apache2

Por fim, existe uma página web direcionada para o utilizador ubuntu, onde é possível aceder através do endereço 10.0.0.2/~ubuntu onde, neste caso, ubuntu representa o nome do utilizador corresponde à página à qual se pretende aceder (figura 15).

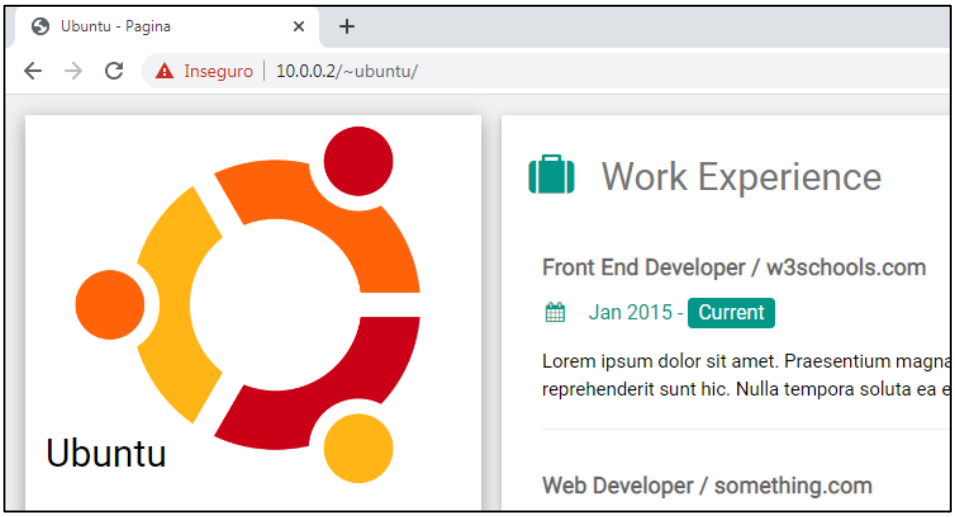

Figura 15 - Página pessoal do utilizador ubuntu

Referências

[1] J. Ellingwood, "How To Set Up Nginx Server Blocks (Virtual Hosts) on Ubuntu 16.04," 19 Maio 2016. [Online]. Available: www.digitalocean.com.

[2] J. Ellingwood, "How To Create Temporary and Permanent Redirects with Apache and Nginx," 12 Agosto 2013. [Online]. Available: www.digitalocean.com.

[3] B. Boucheron, "How To Create a Self-Signed SSL Certificate for Nginx in Ubuntu 18.04," 5 Julho 2018. [Online]. Available: www.digitalocean.com.

[4] J. Ellingwood, "How To Set Up Password Authentication with Nginx on Ubuntu 14.04," 10 Agosto 2015. [Online]. Available: www.digitalocean.com.

[5] L. Tagliaferri, "How To Set Up Apache Virtual Hosts on Ubuntu 18.04 [Quickstart]," 19 fevereiro 2020. [Online]. Available: www.digitalocean.com.

[6] B. Eres, "How to Enable HTTP/2 in Apache Web Server," 06 Março 2020. [Online]. Available: howtoforge.com.

[7] Student, "How to Enable Userdir for Apache2 / Nginx on Ubuntu 17.04 | 17.10," 2017 Setembro 2017. [Online]. Available: websiteforstudents.com.

[8] L. Tagliaferri, "How To Set Up Password Authentication with Apache on Ubuntu 18.04," 3 fevereiro 2020. [Online]. Available: www.digitalocean.com.

[9] "VirtualHost Examples," [Online]. Available: httpd.apache.org.

[10] Jithin, "Módulos Apache comumente usados," 8 setembro 2016. [Online]. Available: www.interserver.net.

[11] F. Timme, "Secure Your Apache With mod_security," [Online]. Available: www.howtoforge.com.

[12] K. Juell e E. Heidi, "How To Secure Apache with Let's Encrypt on Ubuntu 18.04," 6 Agosto 2020. [Online]. Available: www.digitalocean.com.

[13] M. Papiernik, "How To Add the log Module to Nginx on Ubuntu 16.04," 31 Outubro 2016. [Online]. Available: www.digitalocean.com.

[14] tremor, "Apache mod_auth_form & mod_session_crypto," 2013. [Online]. Available: msudol.com.

[15] T. Jankov, "Nginx vs Apache: Confronto Entre Servidores Web," 1 Setembro 2020. [Online]. Available: kinsta.com.