

Postmortem 13/04/2020: nginx-proxy & novo site do IMEsec

Métricas

- **Tempo fora do ar:** 31 minutos
- **Serviços afetados:** O site do imesec (<https://imesec.ime.usp.br>).
- **O quão foi afetado?:** Queda total

O que aconteceu

Estávamos felizes em colocar o novo site em produção. No momento, estávamos em uma infra um pouco bamba: todos os containers estavam rodando em um `docker-compose` na `torradeira`, nosso servidor com 4 CPU/8 GB RAM.

Fizemos uma simples mudança no nosso ambiente de deploy. Apenas duas linhas que causaram o problema todo. Estava assim antes:

```
nginx-website:
  image: richarvey/nginx-php-fpm:latest
  deploy:
    replicas: 2
    restart_policy:
      condition: any
      delay: "0"
  environment:
    GIT_EMAIL: imesec@ime.usp.br
    GIT_NAME: imesec
    GIT_REPO: https://github.com/IMEsec-USP/website.git
    VIRTUAL_HOST: imesec.ime.usp.br
    LETSENCRYPT_HOST: imesec.ime.usp.br
    WEBROOT: /var/www/html/website/
```

E depois da mudança:

```
nginx-website:
  image: imesec/website # mudança aqui!
  deploy:
    replicas: 2
    restart_policy:
      condition: any
      delay: "0"
  environment:
    GIT_EMAIL: imesec@ime.usp.br
    GIT_NAME: imesec
    GIT_REPO: https://github.com/IMEsec-USP/website.git
    VIRTUAL_HOST: imesec.ime.usp.br
    LETSENCRYPT_HOST: imesec.ime.usp.br
    # Tiramos a variável webroot, já que não precisaríamos mais dela.
```

Após a aplicação dessa mudança, a *ada* logo apitou nos seus logs:

```
[2020-04-13 15:33:14.018261] IMEsec Website returned 502
[2020-04-13 15:35:14.726066] IMEsec Website returned 502
[2020-04-13 15:37:18.408670] IMEsec Website returned 502
[2020-04-13 15:37:18.408754] IMEsec Website was considered HealthStatus.DOWN
[2020-04-13 15:37:18.408830] [TelegramHandler] broadcasting message in conversations
```

[131845033, -1001284501077, 211525815]

O site estava fora, e a ada apitou sobre:



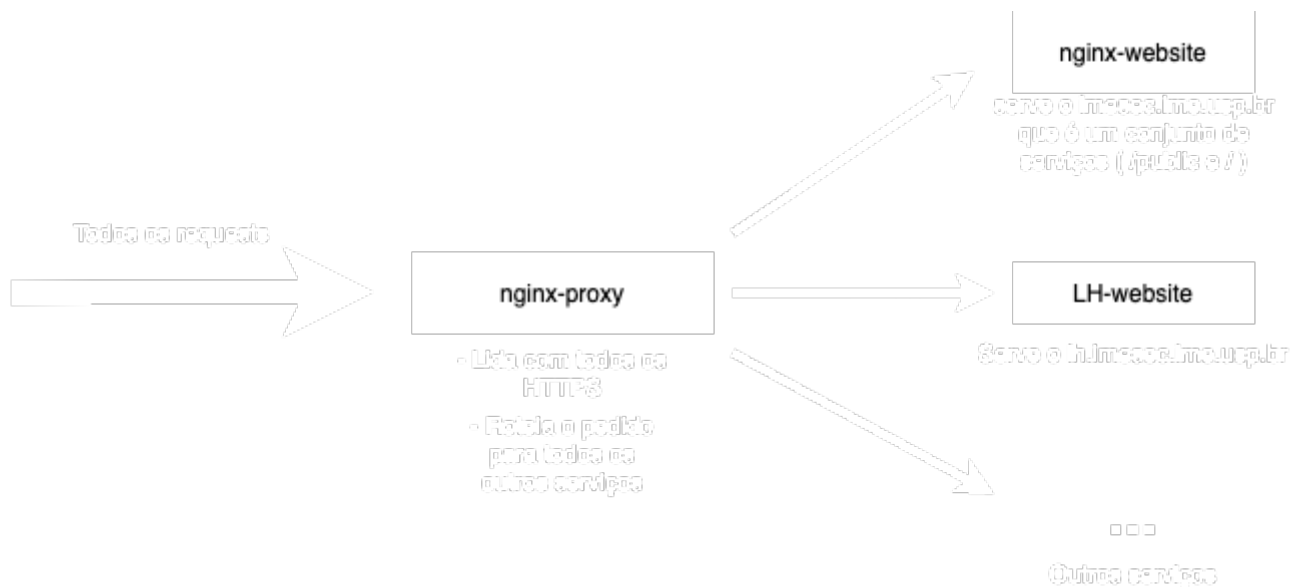
nota: o log está em GMT (UTC) e o horário da mensagem está em GMT+1 (British Summer Time).

O `razgrizone` e o `r0zbot` foram tentar resolver a situação.

processo de resolução

0. O que sabíamos

- Nossa infraestrutura desejada no momento era essa:



- Todos esses serviços estavam declarados na stack `imesec-full-stack.yml` declarada aqui.
- O Dockerfile da imagem nova era esse.
- Tudo descrito anteriormente.

1. desescalando o problema

Primeira ideia: **Rollback, depois pense**. Ou seja, primeiro tente voltar para a versão anterior e resolver a crise imediata. Só **depois**, com tudo estável, resolveríamos o problema.

Fomos o que tentamos. Mas a situação não normalizou. O website voltou com um 404:



Page not Found



Nessa hora, paramos pra **pensar**. Em uma avaliação rápida, achamos que resolver o problema e colocar o site novo no ar seria ainda mais rápido.

2. Analisando o ponto de falha

Já que conhecíamos a infraestrutura que estávamos mexendo, a primeira coisa foi tentar identificar o **ponto de falha** nela. Afinal, não sabíamos se o problema estava *dentro* da nova imagem `imesec/website` (logo do serviço `nginx-website` no gráfico) ou se era a comunicação entre a proxy de borda e o serviço (a seta que vai do `nginx-proxy` até o `nginx-website`).

De dentro do servidor, tentamos fazer o mesmo fluxo que um navegador faria:

```
curl -k -H 'Host: imesec.ime.usp.br' https://localhost
```

Isso nos devolveu o mesmo problema: `502 Bad Gateway`.

Tentamos então abrir uma porta do servidor e conectar nela diretamente, sem passar pelo proxy de borda. Fizemos isso com essa configuração:

```
nginx-website:
  image: imesec/website
  deploy:
    # ...
  environment:
    # ...
  ports:
    - "9001:5000"
```

Ou seja, a porta do servidor 9001 redirecionaria para a porta 5000 dentro do container. Fizemos

isso pois a porta 5000 é a que está exposta na imagem.

Fizemos isso, e o site ficou visível para todos.



O problema estava então entre a conexão entre a proxy de borda e o serviço. Tentamos então acessar o site normalmente algumas vezes pra que isso ficasse óbvio nos logs. E ficou mesmo. Apareceu isso nos logs:

```
nginx.1 | imesec.ime.usp.br 172.19.0.1 - - [13/Apr/2020:16:01:39 +0000] "GET / HTTP/2.0"
502 559 "-" "Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/80.0.3987.118 Safari/537.36"
nginx.1 | 2020/04/13 16:01:44 [error] 58#58: *44 connect() failed (111: Connection
refused) while connecting to upstream, client: 172.19.0.1, server: imesec.ime.usp.br,
request: "HEAD / HTTP/2.0", upstream: "http://172.19.0.25:80/", host: "imesec.ime.usp.br"
```

Tomamos atenção especial ao fato de não conseguir se conectar em `http://172.19.0.25:80/`. A porta estava como porta 80 e não 5000!

tentamos fazer os dois comandos:

```
curl -I HEAD http://172.19.0.25:80/
curl -I HEAD http://172.19.0.25:5000/
```

O de cima não funcionou (ele deu `Connection refused`), mas o segundo funcionou. Era o que esperávamos, afinal, nosso serviço está configurado pra rodar na porta 5000.

Tínhamos então que configurar a proxy pra acessar o serviço na porta 5000. Isso foi feito com uma variável de ambiente:

```
nginx-website:
# ...
```

```
environment:
```

```
VIRTUAL_PORT: 5000
```

Ao restartarmos o container do website, tudo voltou a funcionar normalmente.

Descobrendo a causa raíz

Depois de consertarmos da forma mais rápida possível, demos uma olhada mais a fundo no que tinha acontecido.

A imagem que usamos na proxy de borda utiliza o manifesto dos outros containers para automaticamente se configurar. Ela faz isso analisando as variáveis de ambiente e dockerfiles dos outros containers.

A imagem do nginx que usamos como base do nosso serviço também exporta a porta 80. Logo, nosso proxy não conseguiu identificar automaticamente qual delas teria que usar, e como fallback usou a primeira porta exportada: a porta 80, onde não tinha nada rodando.

Follow-ups

O follow up foi a resolução do problema: adicionar a porta 5000 no manifesto da nossa stack.

Nós também entendemos o porque do rollback não funcionar: a imagem antiga (`richarvey/nginx-php-fpm`) usa a variável de ambiente `GIT_REPO: https://github.com/IMEsec-USP/website.git` para clonar o repositório antes de rodar. Como a `master` do nosso repositório não tinha mais as pastas que ele esperava, ele não funcionou.

Uma lição aprendida é: quando possível, **não usar imagens que se constroem em runtime e não em tempo de build**. Uma imagem, quando funciona da primeira vez, é esperada a funcionar em todas as vezes seguintes. Como mudamos o ambiente em volta do novo container da imagem `richarvey/nginx-php-fpm`, ele parou de funcionar. Se tivéssemos construído uma imagem estática com o nosso site antes (e é o que temos agora), isso não teria acontecido.

Revision #7

Created Mon, Apr 13, 2020 4:07 PM by razgrizone

Updated Mon, Apr 13, 2020 7:30 PM by razgrizone