06

Agendando tarefas com o crontab

Transcrição

O script criado já é capaz de fazer a requisição para o `localhost` e consegue verificar qual é o status HTTP que obtemos. E a nossa próxima estratégia é verificar qual é o conteúdo da variável `resposta_http` , e ver se ela é 200.

Como sabemos, o número 200 indica que está tudo bem. Caso o status seja diferente de 200, então será necessário reiniciar o servidor.

```
#!/bin/bash

    resposta_http=$(curl --write-out %{http_code} --silent --output /dev/null http://localhost)
if [ $resposta_http -eq 200 ]
then
    echo "Tudo esta ok com o servidor"
else
    echo "Houve um problema no servidor. Tentando reinicializar"
fi
```

Para reinicializar o servidor, utilizamos o comando `systemctl restart apache2` :

```
#!/bin/bash

resposta_http=$(curl --write-out %{http_code} --silent --output /dev/null http://localhost)
if [ $resposta_http -eq 200 ]
then
    echo "Tudo esta ok com o servidor"
else
    echo "Houve um problema no servidor. Tentando reinicializar"
    systemctl restart apache2
    echo "...Servidor reinicializado"
fi
```

Saímos, e testaremos. Pela URL (localhost), temos acesso ao conteúdo web do servidor. Se pararmos o servidor Apache, tentaremos realizar novas requisições, porém o servidor não vai estar disponível. Vamos pará-lo, a fim de ver como ele se comporta:

```
$ sudo service apache2 stop
```

Depois, voltemos ao navegador, e teclamos o "F5", onde uma nova requisição será feita. Como podemos ver, de fato a requisição não foi feita. Vamos então ver se o nosso script é capaz de fazer a verificação do status http, e mostrar um status diferente de 200, visto que a requisição foi mau sucedida.

Como precisaremos reinicializar o servidor, precisamos ter o privilégio do usuário root:

```
$ sudo bash monitoracao-servidor.sh
```

A mensagem mostrada é a seguinte:

```
Houve um problema no servidor. Tentando reinicializar
...Servidor reinicializado
```

Será que ele realmente foi reinicializado?

Voltemos ao navegador, e clicamos no "F5" para fazer uma nova requisição.

Após dar "F5", conseguimos ter acesso ao conteúdo web a ser acessado pelos usuários da *Multillidae*. Então, o script é capaz de verificar o status HTTP, alertar caso ele esteja diferente de 200, e reinicializar o servidor. Mas, qual é a chance de lembrarmos de rodar esse script, por exemplo, a cada cinco minutos, ou dez minutos..? Provavelmente vamos nos esquecer! E o ideal seria configurar esse script para que ele seja executado de tempos em tempos, dependendo da nossa necessidade.

Para isso, precisaremos da ajuda do **crontab**!

O **crontab** é um arquivo que terá uma lista de comandos para serem executados em um período de tempo. Para que o script seja executado pelo crontab, precisamos mudar o status de permissão dele, e dizer que ele pode ser executado. Com o comando `chmod +x monitoracao-servidor.sh` , damos o **status de permissão** para esse script ser executado pelo crontab.

Precisamos fazer a edição no crontab reinicializando o servidor, para isso, é necessário utilizar o crontab via usuário **root**.

```
$ sudo crontab -e
```

Dentro desse arquivo, temos a possibilidade de configurá-lo para executar o script de acordo com a nossa necessidade. Algumas informações são mostradas para nós, por exemplo, é mostrado um modelo de como configurar um script para que ele seja executado toda semana, às 5 horas da manhã:

```
# m h  dom mon dow      command
```

- m : minutos
- h : horas
- dom : dia do mês
- mon : mês
- dow : dia da semana
- command : caminho completo para o script

Faremos um exemplo. Se quisermos que o script seja executado a cada dois minutos, faríamos dessa forma:

```
*/2 * * * * /home/rafael/Scripts/monitoracao-servidor.sh
```

Sabemos que esse script será executado a cada dois minutos. Se o servidor parar de funcionar, esse script continuará sendo executado, e então ele irá reparar que o status vai estar diferente de 200, e reinicializará para nós. Vamos ver se isso realmente vai acontecer.

Salvamos o arquivo com "Ctrl + X" e "Y", e paramos o servidor com `sudo service apache2 stop`. Se atualizarmos a página, receberemos a mensagem de que não é possível acessar aquele conteúdo.

Aguardaremos dois minutos no relógio, pois o nosso script deve ser executado a cada dois minutos!

Ótimo! Depois que se passaram os dois minutos, atualizaremos novamente a página de login do servidor, e com certeza, temos novamente o acesso à esse conteúdo!

Agora, o usuário não precisa ficar lembrando de rodar esse script a cada 5 minutos, a cada 10 minutos, por exemplo. Entretanto, ainda temos aquela outra etapa. Como o servidor parou de funcionar por um período, precisamos mandar um e-mail para o administrador do sistema para que ele possa verificar depois o que pode ter ocorrido para que o servidor tenha parado de funcionar.