10

Enviando email para o administrador

Transcrição

Conseguimos configurar o script para que ele fosse executado a cada dois minutos pelo **crontab**. Agora falta fazermos aquela última requisição para os diretores da *mutillidae*, que é enviar um e-mail para o administrador de sistemas, para que ele possa entender por que o servidor parou de funcionar.

A fim de realizar algumas mudanças em nosso script, pausaremos a linha no crontab que executa o script a cada dois minutos. Para acessar o crontab, precisamos do privilégio do usuário root:

```
$ sudo crontab -e
```

Para fazer com que o comando pare de funcionar, podemos colocar uma `#` antes do comando, assim essa linha passará a ser um comentário, dessa forma:

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

Vamos sair e salvar com "Ctrl + X" e "Y".

Enquanto o crontab está pausado momentaneamente, vamos precisar fazer o download de duas ferramentas para enviar o email: o **SSMTP** e o **Mail utils**. Nessa etapa, é importante você possuir uma conta no GMAIL (https://www.google.com/gmail/), pois aqui no curso utilizaremos o *GMAIL* para enviar informações para o administrador da *mutillidae*.

Faremos o download do primeiro pacote **SSMTP**:

```
$ sudo apt-get install ssmtp
```

Uma vez que o download foi concluído, precisaremos configurar o arquivo do SSMTP para indicar qual será o e-mail responsável por enviar essas informações. Para fazer essas alterações, precisaremos ter privilégios do usuário **root**, e pelo fato de que teremos que fazer a edição de vários campos, utilizaremos o `gedit`. O arquivo está localizado no diretório `/etc/ssmtp/ssmtp.conf`:

```
$ sudo gedit /etc/ssmtp/ssmtp.conf
```

Vamos apagar todo o conteúdo que há dentro desse arquivo, deixando-o limpo para fazermos a configuração do e-mail responsável por enviar a mensagem para o administrador.

Bom, criamos duas contas de e-mail: a `servidor.mutillidae@gmail.com` sendo esta a conta principal, e a conta `adm.mutillidae@gmail.com`. Para realizar a configuração de que as mensagens sejam enviadas a partir do e-mail `servidor.mutillidae@gmail.com`, devemos colocar desta maneira:

```
root=servidor.mutillidae@gmail.com
```

Agora temos que indicar a porta de comunicação que o *GMAIL* utiliza para enviar as mensagens:

```
root=servidor.mutillidae@gmail.com
mailhub=smtp.gmail.com:587
```

Agora, colocaremos o **usuário** e a **senha** para que a gente possa enviar a mensagem.

```
root=servidor.mutillidae@gmail.com
mailhub=smtp.gmail.com:587
AuthUser=servidor.mutillidae@gmail.com
AuthPass=
```

Usaremos também, o protocolo de criptografia para enviar essas mensagens para o administrador da *Mutillidae*.

```
root=servidor.mutillidae@gmail.com
mailhub=smtp.gmail.com:587
AuthUser=servidor.mutillidae@gmail.com
AuthPass=
UseSTARTTLS=yes
```

O que nos resta agora é colocar a senha da sua conta, clicar em "Salvar" no canto superior direito, e a configuração do arquivo SMTP já está concluída.

Agora que configuramos o arquivo `ssmtp.conf`, colocaremos o envio do e-mail, para que o script passe a mensagem para o administrador da *Mutillidae*. Agora faremos o download do **mailutils**!

```
$ sudo apt-get install mailutils
```

Após a conclusão do download, vamos até o script para fazer as alterações, com o comando `nano monitoracao-servidor.sh`.

Bom, qual é a nossa preocupação aqui? Nós não queremos ver essas mensagens no terminal. Só queremos mandar a mensagem para o administrador, para que ele possa ver depois o que ocorreu.

Apagaremos as mensagens que serão impressas no terminal, e vamos nos preocupar em enviar somente a mensagem para o administrador da *Mutillidae*. No momento, como é mais importante enviar essa mensagem para o administrador, podemos alterar um pouco a lógica que está nesse script.

Antes, verificávamos o conteúdo da variável `resposta_http`. Se ela fosse 200, seria impresso no terminal e tudo estava ok. Se não fosse 200, mostraria uma mensagem dizendo que estava tentando reinicializar o servidor.

Ao mudar a lógica, fizemos também algumas mudanças no código. Caso o conteúdo da variável `resposta_http` **não seja** igual a 200, é enviado um e-mail para o administrador e o servidor reinicializa. Como estávamos validando um conteúdo que não é igual a 200, então trocaremos o comando `-eq` para `-ne` que significa (*not equals*).

```
#!/bin/bash

resposta_http=$(curl --write-out %{http_code} --silent --output /dev/null http://localhost)
```

```
if [ $resposta_http -ne 200 ]
then

    systemctl restart apache2
fi
```

Se não for igual a 200, é porque existe algum problema lá e é necessário fazer a reinicialização do servidor. Para mandarmos o e-mail, faremos o seguinte:

```
if [ $resposta_http -ne 200 ]
then
mail -s "Problema no servidor"
    systemctl restart apache2
fi
```

Após o `mail` , configuramos o assunto da nossa mensagem, cujo o título da mesma, colocamos entre as aspas. Depois de ter colocado o título, especificaremos o e-mail para o qual estaremos enviando essa mensagem:

```
if [ $resposta_http -ne 200 ]
then
mail -s "Problema no servidor" adm.mutillidae@gmail.com
    systemctl restart apache2
fi
```

Podemos também, colocar o conteúdo interno nesse e-mail, uma mensagem. Faremos o uso do comando `<<del` , sendo ele um delimitador, e fecharemos com o mesmo comando `del` :

```
if [ $resposta_http -ne 200 ]
then
mail -s "Problema no servidor" adm.mutillidae@gmail.com<<del
Houve um problema no servidor e os usuarios pararam de ter acesso ao conteúdo web.
del
    systemctl restart apache2
fi
```

O script foi configurado para que, caso o conteúdo da `resposta_http` **não seja** igual a 200, será mandado um e-mail para o administrador da *Mutillidae,* e o servidor será reinicializado logo em seguida. Saímos com o "Ctrl + X" e "Y" para salvar.

A fim de testar as novas alterações no script, vamos parar o servidor apache com o comando `sudo service apache2 stop` .

Legal, agora precisamos tirar o crontab do *pause,* para que ele volte a executar o script a cada dois minutos.

```
$ sudo crontab -e
```

Então nós retiramos o `#` do comando `*/2 * * * * /home/rafael/Scripts/monitoracao-servidor.sh` . Salvamos, e pronto! O crontab voltará a executar o script a cada dois minutos!

Passado os dois minutos, podemos testar o acesso ao servidor, com o "F5", e vemos que esta funcionando novamente. Vamos abrir a conta de e-mail do administrador para ver se ele recebeu o e-mail.

Legal! O administrador recebeu o e-mail enviado pelo crontab.

Conseguimos fazer não somente a verificação do servidor, mas também conseguimos enviar um e-mail automaticamente para o administrador, assim ele está ciente que houve problemas com o servidor e depois, ele pode analisar o que pode ter ocorrido.