UNIVERSIDADE FEDERAL DO PARANÁ
ANÁLISE E DESENVOLVIMENTO DE SISTEMAS
DS010 - ADMINISTRAÇÃO DE SISTEMAS

ANA KAROLINA DOS SANTOS

PROGRAMAS IMPLEMENTADOS

CURITIBA
21 DE FEVEREIRO DE 2023

SUMÁRIO

1. DOCUMENTAÇÃO DO CÓDIGO SHELL SCRIPT PARA REALIZAR BACKUP NA MÁQUINA COM SISTEMA OPERACIONAL LINUX.

O código foi pensado para realizar backups de diretórios inteiros, como transição de sistemas no ambiente de produção, que necessitam do backup antes qualquer alteração dentro do diretório ou cultura empresarial de impossibilitar a perda de arquivos de diretórios importantes para o desenvolvimento de produtos ou serviços.

Para executar o script necessita que sua máquina use como sistema operacional principal o Linux, se estiver usando outro sistema operacional como principal e usar a distribuição Ubuntu, mesmo usando WSL, o script não executará como deve por causa de conflitos e permissões de comandos.

Contudo, deve-se baixar o arquivo shell script em sua máquina, preferencialmente no diretório destino dos arquivos de backup, mas poderá também instalar no diretório de origem, abrir o terminal de comandos e dar permissão de uso para o código e o iniciar com o seguinte comando:

$chomd\ xxx\ backup.sh$

$./backup.sh$

Sendo xxx do comando $chmod$ representado pela opção de acesso, se deseja que o script seja usado apenas pelo usuário root (700), pelo super usuário e pelo grupo (770), por qualquer usuário (777).

1.1. O CÓDIGO

Inicia-se o código com o indicador de programa bash:

$\#!\ /bin/bash$

Segue-se as funções que organizam o código em fragmentos e configuram o comando de backup.

1.1.1. FUNÇÃO MENU PRINCIPAL

A função Menu Principal tem o objetivo de validar se o usuário realmente quer realizar o backup ou o invocou por engano, precisando da confirmação do usuário para rodar o código shell de backup ou sair do programa. A função é re-invocada caso o usuário digite algo que não dê finalidade ao programa. A função Menu Principal:

```
MenuPrincipal(){
    echo "Realizar backup de diretórios e/ou arquivos?"
```

```
        echo "1 - Sim"
        echo "2 - Não. Sair"
        read fazerBackup
        case $fazerBackup in
                1) functionBackup ;;
                2) exit;;
                *) echo "Opção inexistente.";  MenuPrincipal ;;
        esac
}
```

1.1.2. FUNÇÃO DIRETÓRIO DE DESTINO

A função Diretório De Destino usa uma variável global dentro do código (diretorioDeDestino) e confere se o diretório passado pelo usuário existe dentro da máquina.

A priori existe a pergunta se o diretório para guardar o backup é o que contém o programa, caso não seja, o programa pede o caminho desde a raiz até a pasta almejada como destino do backup. Verifica se a pasta existe nesse percurso e caso não exista, entrará em um loop exigindo o caminho correto de um diretório existente da máquina.

A função Diretório De Destino:

```
        functionDiretorioDeDestino(){
        clear
        echo "Deseja realizar o backup no diretorio atual? (s/n)"
          read verificaBackupNoDiretorioAtual
          if [ $verificaBackupNoDiretorioAtual = "s" ]; then
                diretorioDeDestino=$(pwd)
             echo "Os arquivos irão para: $diretorioDeDestino"
          else
                echo "Digite o caminho (desde a raiz) do diretório de destino para backup:"
                read diretorioDeDestino
                if [ -d $diretorioDeDestino ]; then
                        echo $diretorioDeDestino
                else
                until [ -d $diretorioDeDestino ]; do
                                echo "Este diretório não existe. Digite novamente o caminho:"
```

```
            read diretorioDeDestino
        done
                echo "Os arquivos irão para: $diretorioDeDestino"
    fi
fi
}
```

1.1.3. FUNÇÃO DIRETÓRIO DE ORIGEM

A função Diretório De Origem usa uma variável global dentro do código (diretorioDeOrigem) e confere se o diretório passado pelo usuário existe dentro da máquina.

A priori existe a pergunta se o diretório para realizar o backup é o que contém o programa, caso não seja, o programa pede o caminho desde a raiz até a pasta a realizar backup. Verifica se a pasta existe nesse percurso e caso não exista, entrará em um loop exigindo o caminho correto de um diretório existente da máquina.

A função Diretório De Origem:

```
functionDiretorioDeOrigem() {
clear
echo "Fará backup do diretório atual? (s/n)"
read confirmacaoDeOrigem
if [ $confirmacaoDeOrigem = "s" ]; then
    diretorioDeOrigem=$(pwd)
    echo "Os arquivos irão para: $diretorioDeOrigem"
  else
      echo "Digite o caminho (desde a raiz) do diretório de origem para backup:"
      read diretorioDeOrigem
    if [ -d $diretorioDeOrigem ]; then
      echo "diretório de origem: $diretorioDeOrigem"
      else
            until [ -d $diretorioDeOrigem ]; do
                  echo "Este diretório não existe. Digite novamente o caminho:"
        read diretorioDeOrigem
            done
            echo "Os arquivos irão para: $diretorioDeOrigem"
```

```
        fi
    fi
}
```

1.1.4. FUNÇÃO CONFIGURAR BACKUP

A função Configurar Backup perguntará ao usuário se ele deseja transferir até os sub diretórios vazios da pasta de origem e se deseja deixar algum arquivo do diretório sem backup, seria uma opção para realizar backup do diretório que o script está alocado, ou cumprir alguma politica de documentos exclusivos sem possibilidade de cópias.

Se o usuário quiser excluir algum arquivo do backup, deverá escrever o nome do arquivo e sua extensão, exemplo: texto.txt. Para excluir o arquivo com o nome de texto e sendo uma extensão de arquivo de texto, senão o rsync irá ignorar o comando de excluir o arquivo do backup.

A função Configurar Backup:

```
functionConfigurarBackup(){
   clear
echo "Deseja excluir subdiretórios vazios para a pasta de backup? (s/n)"
read confirmaExclusaoDeSubdiretoriosVazios
if [ $confirmaExclusaoDeSubdiretoriosVazios = "s" ]; then
        opcaoDeBackup="-hamP"
else
        opcaoDeBackup="-haP"
fi
echo "Deseja excluir algum arquivo do backup? (s/n)"
read confirmaExclusaoDeArquivos
if [ $confirmaExclusaoDeArquivos = "s" ]; then
        echo "Escreva o nome dos arquivos e se existir mais de um arquivo os separe com vírgula e espaço."
        echo "  Exemplo: arquivo1, arquivo2, arquivo3, ..., arquivoN"
        read arquivosAExcluirNoBackup
        opcaoDeBackup="$opcaoDeBackup --exclude=$arquivosAExcluirNoBackup"
fi
}
```

A função usa combinação de comandos do comando rsync, os usados no código foram:

- a (--archive): para copiar os arquivos com seus respectivos metadados e permissões;
- m (--prune-empty-dirs): para excluir subdiretórios vazios;
- h (--human-readable format): para os dados sobre o backup serem apresentados em tamanhos compreensíveis para o usuário;
- P (--progress e --partial): para aparecer a barra de progresso do backup e caso o backup seja interrompido, quando a máquina poder rodar normalmente, continuará o backup;
- --exclude: para excluir arquivos e/ou da origem do backup.

1.1.5. FUNÇÃO PERIODICIDADE DOS BACKUPS

A função Periodicidade Dos Backups perguntará ao usuário se ele deseja realizar backups com determinada periodicidade ou o agendar, caso o usuário não tenha interesse será direcionado para confirmação de backup. A função Periodicidade Dos Backups:

```
functionPeriodicidadeDosBackups() {
clear
echo "Deseja realizar backups com periodicidade, ou o deixar agendado? (s/n)"
read opcaoDeControleDeTempo
if [ $opcaoDeControleDeTempo = "s" ]; then
        functionCrontab ;
else
        echo "Backup será realizado em breve"
fi
}
```

1.1.6. FUNÇÃO CRONTAB

A função Crontab permitirá capturar informações de tempo do usuário com variáveis globais para configurar o comando crontab na função de Backup e realizar os backups com determinada periodicidade, caso seja o desejo do usuário. A função Crontab:

```
functionCrontab() {
```

```
echo "Deseja realizar backups em dias da semana? (s/n)"
read verificaDiasDaSemana
if [ $verificaDiasDaSemana = "s" ]; then
        echo;
        echo "  Digite o dia da semana. Exemplo de digitação:"
        echo "  Para segunda, digite: 1"
        echo "  Para terça, digite: 2"
        echo "  ..."
        echo "  Para domingo, digite: 7"
        echo "  1, 2, 3"
        read diasDaSemana
else
            diasDaSemana=*
fi

echo "Deseja realizar backups em dias do mês especifico? (s/n)"
read verificaDiasDoMes
if [ $verificaDiasDoMes = "s" ]; then
        echo;
        echo "  Digite o dia do mês (1, 2, 3, ..., 31)"
        read dia
else
            dia=*
fi

echo "Deseja realizar backups em algum mês especifico? (s/n)"
read verificaMes
if [ $verificaMes = "s" ]; then
        echo;
        echo "  Digite o mês (1, 2, 3, ..., 12)"
        read mes
else
            mes=*
fi
```

```
echo "Deseja realizar backups em algum horário especifico? (s/n)"
read verificaHoras
if [ $verificaHoras = "s" ]; then
        echo;
        echo "  Digite a hora e apenas a hora (0 a 23, meia-noite=0)"
        read h
        echo "  Digite os minutos (0 a 59)"
        read min
else
            h=*
            min=*
fi
echo "Backup agendado para o/os dia/dias: $dia, do/dos mês/meses: $mes,"
echo " às: $h:$min, do/dos dia/dias semanal/semanais $diasDaSemana"
}
```

Onde as variáveis correspondem:

- diaDaSemana aos dias da semana;
- dia aos dias do mês;
- mês aos meses;
- h a hora;
- min aos minutos.

Além de especificações únicas, o usuário poderá optar por backups em espaços de tempos ou em mais de um único momento durante o dia, a semana, o mês ou o ano. O usuário deve usar os operadores vírgula, ou hífen, ou barra para concatenar os momentos que deverá ocorrer o backup.

- a vírgula especificará os momentos, como backup nos nas segundas-feiras e terças-feiras, será: 1, 2;
- o hífen indicaria o espaço de tempo, como backup em dias úteis, será: 1-5;
- a barra dividirá o tempo em realizar a tarefa, como backup a cada 2 dias entre os dias 1 e 10 de um mês, será: 1-10/2.

Caso o usuário não opte por determinar alguma das variáveis, a variável obterá o valor * que corresponde por indeterminação, dando poder a máquina para realizar periodicamente nesse espaço de tempo. Ou seja:

- para dia da semana a qualquer dia da semana que respeite as outras configurações de tempo;
- para dia a qualquer dia que respeite as outras configurações de tempo;
- para mês a qualquer mês que respeite as outras configurações de tempo;
- para hora a qualquer hora que respeite as outras configurações de tempo;
- para minutos a qualquer minuto que respeite as outras configurações de tempo;

Se o usuário não determinar nenhuma destas variáveis, o backup ocorrerá em um loop infinito dentro da máquina, por isso deverá responder negativamente à pergunta de backups periódicos ou cancelar o backup na função Confirma Backup que será evocada após a Crontab.

1.1.7. FUNÇÃO CONFIRMAR BACKUP

A função Confirmar Backup tem o mesmo objetivo da função Menu Principal, ela deverá validar se o usuário realmente quer realizar o backup, porém é para caso o usuário tenha respondido de forma equivocada alguma das perguntas de agendamento de horário, ou destino, ou origem do backup. A função tem o objetivo de ser direta ao objetivo do usuário: realizar o backup ou sair do programa. A função Confirmar Backup:

```
functionConfirmaBackup(){
   clear
if [ ! $dia ]; then
        echo "Confirma backup do $diretorioDeOrigem para $diretorioDeDestino? (s/n)"
else
        echo "Confirma backup do $diretorioDeOrigem para $diretorioDeDestino,"
        echo " agendado para dia: $dia, mês: $mes, dia da semana: $diasDaSemana, horário: $h:$min? (s/n)"
fi
read confirma
if [ $confirma = "s" ] || [ $confirma = "S" ]; then
```

```
        echo "Backup sendo realizado..."
    else
        exit ;
    fi
    }
```

1.1.8. FUNÇÃO BACKUP

A função Backup resgata as variáveis globais com seus valores definidos através das primeiras funções evocadas e configura os comandos rsync e crontab.

A função confere se existiu a configuração de tempo através da condicional sobre a variável dia, pois a função Crontab retornará um valor a variável se o usuário optar por definir agendar o backup, mesmo que ele não opte por especificar o dia do backup.

Se a variável possuir valor definido, irá configurar o comando crontab com as exigências do usuário e usando como comando o rsync também configurado ao gosto do usuário. Senão, irá configurar apenas o comando rsync as exigências do usuário. A função Backup:

A função Confirma Backup:

```
    functionBackup(){
    clear
    dataDoBackup=$(date +%d-%m-%Y)
    functionDiretorioDeOrigem
    functionDiretorioDeDestino
    functionPeriodicidadeNosBackups
    functionConfirmaBackup
    pastaDeBackup="backup-$dataDoBackup"
    comandoBackup="rsync $opcaoDeBackup --backup --backup-dir=$pastaDeBackup
$diretorioDeOrigem $diretorioDeDestino/$pastaDeBackup"
    if [ ! $dia ]; then
        $comandoBackup
    else
        (crontab -l; echo "$min $h $dia $mes $diasDaSemana $comandoBackup") |
sort -u | crontab -
        echo "Backup agendado para: "
        crontab -l
    fi
    }
```

A variável comandoBackup representará o comando RSync e nela será convocada a variável global opcaoDeBackup que foi configurada na função Configurar Backup, entretanto, também existirá mais uma configuração do comando que são as opções --backup e --backup-dir, que devem ser usadas juntas. A opção indicará ao comando que ele está sendo usado para realizar um backup e a segunda opção que o backup deve ser realizado na pasta indicada.

O diretório indicado a opção --backup-dir é um diretório a ser criado com a data do dia que o script foi invocado com a palavra "backup" na frente da data. O nome da pasta é configurado através da variável pastaBackup.

1.2. AO FIM DO CÓDIGO

Por último, faço a chamada da função Menu Principal, senão o script não rodaria o programa, logo que ele foi fracionado em funções e para tais atuarem, necessitam ser invocadas por outra antes ou durante o script.

MenuPrincipal

REFERÊNCIAS

ALVES, Daphne Spier Moreira. PAULA, João Pedro Martins de. MORAES, Leonardo Xavier da Silva. Shell script para backup de arquivos. **HackMD**. 15 set, 2020. Disponível em: <https://hackmd.io/@DS010/B1GjzwCVD>. Acesso em: 23 fev, 2023.

CONTENT. Rock. Comando Chmod: descubra o que ele faz e como usar. **Rock Content Blog**. 27 nov, 21. Disponível em: <https://rockcontent.com/br/blog/chmod/>. Acesso em: 23 fev, 2023.

DELFINO, Pedro. Crontab: Entenda Como Agendar Tarefas No Linux De Uma Vez Por Todas**. Profissionais do Linux.** Disponível em: <https://e-tinet.com/linux/crontab/>. Acesso em: 21 fev, 2023.

DIGITAL OCEAN. **Como usar o Rsync para sincronizar diretórios locais e remotos**. 15 dez, 2020. Disponível em: <https://www.digitalocean.com/community/tutorials/how-to-use-rsync-to-sync-local-and-remote-directories-pt>. Acesso em: 21 fev, 2023.

G., Ariane. Como Usar o Comando Rsync Linux (Sincronização Remota). **Hostinger Tutoriais**. 03 jun, 2022. Disponível em: <https://www.hostinger.com.br/tutoriais/comando-rsync-linux>. Acesso em: 23 fev, 2023.

LUIZ, Henrique. Entendendo o Crontab: Agendamento de tarefas em Linux. **TOTVS Developers - Medium**. 9 abr, 2020. Disponível em: <https://medium.com/totvsdevelopers/entendendo-o-crontab-607bc9f00ed3>. Acesso em: 22 fev, 2023.

TORAB, Reyhaneh. How to Create a crontab: Through a Script. **Baeldung Linux**. 24, abr 2021. Disponível em: <https://www.baeldung.com/linux/create-crontab-script>. Acesso em: 23 fev, 2023.