SHELL SCRIPT

Prof. Sérgio Rodrigues

Instituto Federal Sul-rio-grandense

Curso de Tecnologia em Desenvolvimento de Sistemas para Internet

Disciplina: Gerenciamento de Redes de Computadores

Introdução

Introdução

- O shell é o ”prompt”da linha de comando do Unix e Linux, é o que recebe os comandos digitados pelo usuário e os executa. O shell é a ligação entre o usuário e o kernel.

Introdução

- Para os usuários do Windows, é fácil pensar no shell como um MSDOS melhorado, mas o shell oferece inúmeros recurso que vão desde comandos básicos para navegar entre diretórios e manipular arquivos, ele também possui todas as estruturas de uma linguagem de programação, como IF, FOR, WHILE, variáveis e funções;
- Com isso também é possível usar o shell para fazer scripts e automatizar tarefas sendo extremamente util ao Gerenciamneto de Redes de Computadores.

SHELL SCRIPT

SHELL SCRIPT

- Um script é um arquivo que guarda vários comandos e pode ser executado sempre que preciso. Os comandos de um script são exatamente os mesmos que se digita no prompt, é tudo shell.

Trabalhando com SHELL SCRIPT

Trabalhando com SHELL SCRIPT

- # date
- # df
- # w

Trabalhando com SHELL SCRIPT

- É melhor fazer um script chamado ”sistema” e colocar estes comandos nele. O conteúdo do arquivo ”sistema”seria o seguinte:
  - #!/bin/bash
  - date
  - df
  - w

Trabalhando com SHELL SCRIPT

- Não use o sistema como usuário administrador (root), saia e entre como um usuário normal;
- Use apenas letras minúsculas e evite acentos, símbolos e espaço em branco;
- Salve os arquivos dentro de seu diretório pessoal ($HOME).

Trabalhando com SHELL SCRIPT

- Tornar o script um arquivo executável. Use o seguinte comando para que seu script seja reconhecido pelo sistema como um comando executável:
- $ chmod +x sistema

Trabalhando com SHELL SCRIPT

- Se o script estiver no diretório corrente, chame-o com um ”./” na frente, deste modo:
- $ ./sistema

Trabalhando com SHELL SCRIPT

- O comando ”echo” serve para mostrar mensagens na tela. Altere o arquivo sistema conforme abaixo:
- #!/bin/bash
- echo ”Data e Horário:”
- date
- echo
- echo ”Uso do disco:”
- df
- echo
- echo ”Usuários conectados:”
- w

Trabalhando com SHELL SCRIPT

- Para o script ficar mais completo, vamos colocar uma interação mínima com o usuário, pedindo uma confirmação antes de executar os comandos:
- #!/bin/bash
- echo ”Vou buscar os dados do sistema. Posso continuar? s/n”
- read RESPOSTA
- test ”$RESPOSTA” = ”n” && exit
- echo ”Data e Horário:”
- date
- echo
- echo ”Uso do disco:”
- df
- echo
- echo ”Usuários conectados:”
- w

Ver próximo slide

Trabalhando com SHELL SCRIPT

- O conteúdo da variável é acessado colocando-se um cifrão ”$” na frente
- O comando test é útil para fazer vários tipos de verificações em textos e arquivos
- O operador lógico ”&&”, só executa o segundo comando caso o primeiro tenha sido OK. O operador inverso é o ”ll”(pipe)

Trabalhando com SHELL SCRIPT

- Para colocar cometários basta iniciar a linha com um ”#” e escrever o texto do comentário em seguida;
- Também é possível colocar comentários no meio da linha # como este.

Trabalhando com SHELL SCRIPT

- As variáveis são a base de qualquer script. É dentro delas que os dados obtidos durante a execução do script serão armazenados. Para definir uma variável, basta usar o sinal de igual ”=” e para ver seu valor, usa-se o ”echo”: (linha de comando)
- $ VARIAVEL=”um dois tres”
- $ echo $VARIAVEL
- um dois tres
- $ echo $VARIAVEL $VARIAVEL
- um dois tres um dois tres
- $

ATENÇÃO - Não podem haver espaços ao redor do igual ”=”

Trabalhando com SHELL SCRIPT

- Ainda é possível armazenar a saída de um comando dentro de uma variável. Ao invés de aspas, o comando deve ser colocado entre ”$(...)”, veja: (linha de comando)
- $ HOJE=$(date)
- $ echo ”Hoje é: $HOJE”
- Hoje é: Sáb Abr 24 18:40:00 BRT
- $ unset HOJE
- $ echo $HOJE
- $

E finalmente, o comando ”unset” apaga uma variável.

Trabalhando com SHELL SCRIPT

- Diferente de outras linguagens de programação, o shell não usa os parênteses para separar o comando de seus argumentos, mas sim o espaço em branco. O formato de um comando é sempre:
- COMANDO OPÇÕES PARÂMETROS
- O comando ”cat -n sistema” mostra o nosso script, com as linhas numeradas
- Exemplo $ cat-n sistema

Trabalhando com SHELL SCRIPT

- O ”read”é um comando do próprio shell, já o ”date” é um executável do sistema. Dentro de um script, não faz diferença usar um ou outro, pois o shell sabe como executar ambos.
- Há vários comandos que foram feitos para serem usados com o shell (ver pág. 09 da Apostila de Introdução ao Shell Script).

Trabalhando com SHELL SCRIPT

- É possível combinar comandos, aplicando-os em sequência, para formar um comando completo. Usando o pipe ”|” é possível canalizar a saída de um comando diretamente para a entrada de outro, fazendo uma cadeia de comandos. (linha de comando)
- Exemplo:
- $ cat /etc/passwd | grep root | cut -c1-10
- root:x:0:0
- $

Exercícios

Exercício

1. Crie um script denominado ”path” que mostre o path corrente, e qual o usuário que esta logado.
2. Crie um script denominado ”backup” que faça uma cópia do arquivo ”path”para o diretório /tmp e logo após fazer o desligamento do seu PC após 2 minutos.
3. Crie um script denominado ”cadastro” que permite criar um usuário determinar sua senha e cadastras as informações pessoais deste usuário.
4. Crie um script denominado ”verificacao” que mostre o tempo que o sistema esta em uso, a versão do kernel, o uso dos discos, o estado da memória e se a rede está funcionando.

Trabalhando com SHELL SCRIPT (parte 2)

Trabalhando com SHELL SCRIPT (parte 2)

- O canivete suíço dos comandos do shell é o ”test”, que consegue fazer vários tipos de testes em números, textos e arquivos. Ele possui várias opções para indicar que tipo de teste será feito (ver pág. 10).

Trabalhando com SHELL SCRIPT (parte 2)

- Assim como os comandos do sistema que possuem e opções e parâmetros, os scripts também podem ser preparados para receber dados via linha de comando.
- Dentro do script, algumas variáveis especiais são definidas automaticamente, em especial, ”$1”contém o primeiro argumento recebido na linha de comando, ”$2”o segundo, e assim por diante.

Trabalhando com SHELL SCRIPT (parte 2)

Digite esse script abaixo com o nome de ”argumento”

- #!/bin/bash
- # argumentos - mostra o valor das variáveis especiais
- echo ”O nome deste script é: $0”
- echo ”Recebidos $# argumentos: $*”
- echo ”O primeiro argumento recebido foi: $1”
- echo ”O segundo argumento recebido foi: $2”

Trabalhando com SHELL SCRIPT (parte 2)

Ele serve para demonstrar o conteúdo de algumas variáveis especiais. (linha de comando)

- $ ./argumentos um dois três
- O nome deste script é: ./argumentos
- Recebidos 3 argumentos: um dois três
- O primeiro argumento recebido foi: um
- O segundo argumento recebido foi: dois

Trabalhando com SHELL SCRIPT (parte 2)

O shell também sabe fazer contas. A construção usada para indicar uma expressão aritmética é ”$((...))”, com dois parênteses. (linha de comando)

- $ echo $((2*3))
- 6
- $ echo $((2*3-2/2+3))
- 8
- $ NUM=44
- echo $((NUM*2))
- 88
- $ NUM=$((NUM+1))
- $ echo $NUM
- 45

Trabalhando com SHELL SCRIPT (parte 2)

- Assim como qualquer outra linguagem de programação, o shell também tem estruturas para se fazer condicionais e loop. As mais usadas são if, for e while.

Trabalhando com SHELL SCRIPT (parte 2)

Estrutura do IF

- if COMANDO
- then
- comandos
- else
- comandos
- fi

Trabalhando com SHELL SCRIPT (parte 2)

Exemplo de IF (Salve o script com nome ”teste”)

- #!/bin/bash
- echo ”Digite um Numero”
- read VARIAVEL
- if test ”$VARIAVEL” -gt 10
- then
- echo ”é maior que 10”
- else
- echo ”é menor que 10”
- fi

Trabalhando com SHELL SCRIPT (parte 2)

Há um atalho para o test , que é o comando ” [” Ambos são exatamente o mesmo comando, porém usar o ” [” deixa o if mais parecido com o formato tradicional de outras linguagens (tem que ter espaços antes e depois [ ])

- if [”$VARIAVEL” -gt 10]
- then
- echo ”é maior que 10”
- else
- echo ”é menor que 10”
- fi

Trabalhando com SHELL SCRIPT (parte 2)

Altere o script ”teste”para que ele pegue o 1 argumento.

- $ ./teste 20
- e maior que 10

Trabalhando com SHELL SCRIPT (parte 2)

Estrutura do FOR

- for VAR in LISTA
- do
- comandos
- done

Trabalhando com SHELL SCRIPT (parte 2)

Exemplo do FOR (Salve o script com nome ”contando”)

- for numero in um dois três quatro cinco
- do
- echo ”Contando: $numero”
- done

Trabalhando com SHELL SCRIPT (parte 2)

Exemplo do FOR (Salve o script com nome ”passo”)

- for passo in $(seq 10)
- do
- echo ”Numero $passo”
- done

Trabalhando com SHELL SCRIPT (parte 2)

Estrutura do WHILE

- while COMANDO
- do
- comandos
- done

Trabalhando com SHELL SCRIPT (parte 2)

Exemplo do WHILE (Salve o script com o nome ”contador”)

- i=0
- while test $i -le 10
- do
- i=$((i+1))
- echo ”Contando: $i”
- done

Trabalhando com SHELL SCRIPT (parte 2)

Exemplo do WHILE (Salve o script com o nome ”loop”)

- while :
- do
- if test -f /tmp/lock
- then
- echo ”Aguardando liberação do lock...”
- sleep 1
- else
- break
- fi
- done

Exercícios

Outras Informações

Dicas de Shell Script

http://www.dicas-l.com.br/cantinhodoshell/