PROYECTO FINAL

PROGRAMACIÓN PARA LA ADMINISTRACIÓN DE REDES

LOPEZ HERRERA JUAN LUIS

REDES Y SERVICIOS DE CÓMPUTO

CRUZ LOPEZ JOSAFAT
FUENTES MENDOZA ANDREA

LRSC
Licenciatura en Redes y Servicios de Cómputo

CONTENIDO

Instrucciones 1
Programa principal de PowerShell 1
Programa principal de Bash 6
Línea con la que se monta los recursos compartidos 9

PROYECTO FINAL PROGRAMACIÓN EN LA ADMINISTRACION DE REDES

INSTRUCCIONES

Realizar una implementación de un sistema en Linux y Windows, usando Bash y Powershell que implemente las siguientes acciones:

- Alta, Modificación y Eliminación de Usuarios, Directorios en cada equipo al que se tiene acceso, Llaves públicas y privadas de cada usuario
- Subir archivos por cada usuario a los directorios a los que se tiene permiso,
- Crear directorios por cada usuario dentro de los que ya se tiene permiso
- Listar los procesos de el mismo usuario en el otro equipo, desde Windows a Linux y viceversa
- Detener procesos que corren en el otro equipo que sean del mismo usuario
- Listar archivos y directorios en el otro equipo del usuario
- Enviar y recibir archivos comprimidos, encriptados y firmados a otro usuario o a sus directorios en el otro equipo
- Enviar y recibir mensaje encriptados a otro usuario de otro equipo
- Poder revisar los mensajes recibidos por otros usuarios desde el otro equipo

PROGRAMA PRINCIPAL DE POWERSHELL

```
[void][System.Reflection.Assembly]::LoadWithPartialName("MySql.Data")
$global:Nombre=""
$global:Password=""
$global:Login=""

function conexion{
    param(
```

```
        [String[]]$Query

    )

    $Connection = New-Object MySql.Data.MySqlClient.MySqlConnection
    $ConnectionString = "server=" + "localhost" + ";port=3306;uid=" + "root"
+ ";pwd=1234" + ";database="+"usuarios"
    $Connection.ConnectionString = $ConnectionString
    $Connection.Open()

    $Command = New-Object MySql.Data.MySqlClient.MySqlCommand($Query,
$Connection)
    $DataAdapter = New-Object
MySql.Data.MySqlClient.MySqlDataAdapter($Command)
    $DataSet = New-Object System.Data.DataSet
    $RecordCount = $dataAdapter.Fill($dataSet, "data")
    if($global:Login -eq "Login"){
        $global:Nombre = $DataSet.Tables[0].Nombre
        $global:Password = $DataSet.Tables[0].Password
        $global:Login = ""
    }

    $Connection.Close()
}

function Ob-Cre {
    param(
        [String[]]$Usuario,
        [String[]]$Contraseña
    )
    $global:Login="Login"
    conexion -Query "Select * from usuario where Nombre = '$Usuario' "


    if($Contraseña -eq $Password) {
        Set-Content -Path Z:\Salida.txt -Value "Sesion iniciada
correctamente $global:Nombre"
        limpiar
    }else {
        $global:Nombre =""
        $global:Password =""
        Set-Content -Path Z:\Salida.txt -Value "Usuario Invalido o
Contraseña Incorrecta"
    }
```

```
}

function Agregar {
    param(
        [String[]]$Usuario,
        [String[]]$Contraseña
    )
    New-Item -ItemType Directory -Name "$Usuario" -Path Z:\
    conexion -Query "Insert into Usuario values ('$Usuario','$Contraseña')"
    conexion -Query "Insert into directorios values
('$Usuario','Z:\\$Usuario')"
    Set-Content -Path Z:\Salida.txt -Value "Usuario creado exitosamente"
    limpiar
}

function Eliminar {
    param(
        [String[]]$Usuario
    )

    conexion -Query "DELETE FROM Usuario WHERE Nombre='$Usuario'"
    conexion -Query "DELETE FROM directorios WHERE Usuario='$Usuario'"
    Remove-Item -Path "Z:\$Usuario"
    salida -Mensaje "$Usuario Eliminado"
}

function Modificar {
    param(
        [String[]]$Usuario,
        [String[]]$Usuario2,
        [String[]]$Contraseña2
    )

    conexion -Query "UPDATE Usuario SET Nombre = '$Usuario2', Password =
'$Contraseña2' WHERE Nombre = '$Usuario'"
    conexion -Query "UPDATE directorios SET Usuario = '$Usuario2',
Directorio = 'Z:\\$Usuario2' WHERE Usuario = '$Usuario'"

    Rename-Item -Path "Z:\$Usuario" -NewName "$Usuario2"

}

function limpiar {
```

```
    Set-Content Z:\Console.txt -Value "Powershell:Prompt:"
}

function salida {
    param(
        [String[]]$Mensaje
    )
    Set-Content Z:\Salida.txt -Value "$Mensaje"
}

function admin {
    param(
        [String[]]$Ejecutar
    )

    if($global:Nombre -eq "Josa"){
            Switch -Wildcard ("$Ejecutar"){
                "Agregar*" {
                    $User=$Ejecutar.Split(' ')[1]
                    $Contra=$Ejecutar.Split(' ')[2]
                    Agregar -Usuario $User -Contraseña $Contra
                }
                "Eliminar*" {
                    $User=$Ejecutar.Split(' ')[1]
                    Salida -Mensaje "Eliminando Usuario $User"
                    Eliminar -Usuario $User
                    limpiar
                }
                "Modificar*" {
                    $User=$Ejecutar.Split(' ')[1]
                    $User2=$Ejecutar.Split(' ')[2]
                    $Contra=$Ejecutar.Split(' ')[3]
                    Modificar -Usuario $User -Usuario2 $User2 -Contraseña2
$Contra
                }
            }
    }else {
            Switch -Wildcard ("$Ejecutar"){
            "Login*"{
                $User=$Ejecutar.Split(' ')[1]
                $Contra=$Ejecutar.Split(' ')[2]
                Ob-Cre -Usuario $User -Contraseña $Contra
            }
            "Exit*"{
                Salida -Mensaje "Sesion $global:Nombre Cerrada"
```

```
                $global:Nombre=""
                $global:Password=""
                limpiar
            }
            "Subir*" {
                $Archivo=$Ejecutar.Split(' ')[1]
                salida -Mensaje "Moviendo archivo $Archivo a carpeta
personal"
                Move-Item -Path Z:\$Archivo -Destination Z:\$global:Nombre
                limpiar
            }
            "Listar"{
                $result = Get-Process -IncludeUserName | Where-Object
UserName -EQ "DESKTOP-NUC4H85\$global:Nombre"
                Set-Content Z:\Salida.txt -Value $result.Name
            }
            "Detener"{
                salida -Mensaje "Deteniendo los procesos del Usuario
$global:Nombre"
                $Process = Get-Process -IncludeUserName |  Where-Object
UserName -EQ "DESKTOP-NUC4H85\$global:Nombre" | Select-Object -Property Id
                Stop-Process -Id $Process
            }
            "Files*"{
                $resul = Dir Z:\$global:Nombre
                salida -Mensaje "$resul"
                limpiar
           }
          }
    }

}

while(1){
        $Comando = Get-Content Z:\Console.txt
        $Prompt=$Comando.Split(":")[0]
        $Run=$Comando.Split(":")[2]
    if($Prompt -eq "Powershell"){
        Write-Host "$Run"
        admin -Ejecutar $Run
        Get-Content -Path Z:\Salida.txt
        sleep 3
        cls
    }
}
```

PROGRAMA PRINCIPAL DE BASH

```
#!/bin/bash
nombre=""
password=""

consultar (){
    cre=$(mysql -u root -p1234 Usuarios -e "select * from Usuario where
Nombre = '$1'")
    nombre=$(echo $cre | cut -d' ' -f3)
    password=$(echo $cre | cut -d' ' -f4)
    echo "Nombre de usuario: $nombre"
    echo "Contraseña del usuario: $password"
    if [[ "$password" -eq "$2" ]]
    then
        echo "Usuario logueado correctamente" >  /samba/Salida.txt
    fi
}

agregar (){
    mysql -u root -p1234 Usuarios -e "Insert into Usuario values
('$1','$2')"
    mysql -u root -p1234 Usuarios -e "Insert into directorios values
('$1','/samba/$1') "
    mkdir "/samba/$1"
    echo "Usuario $1 agregado correctamente" > /samba/Salida.txt
}

eliminar (){
    mysql -u root -p1234 Usuarios -e "Delete * from Usuario where Nombre =
'$1'"
    mysql -u root -p1234 directorios -e "Delete * from directorios where
usuario = '$1'"
    rm -r "/samba/$1"
    echo "Usuario $1 eliminado correctamente" > /samba/Salida.txt
}

modificar (){
    mysql -u root -p1234 Usuarios -e "Update Usuario set Nombre = '$2',
password = '$3' where Nombre = '$1'"
    mysql -u root -p1234 Usuarios -e "Update directorios set usuario = '$2',
directorio = '/samba/$2' where usuario = '$1'"
```

```
    cp "/samba/$1" "/samba/$2"
    rm -r "/samba$1"
    echo "Usuario $1 actualizado" > /samba/Salida.txt
}

limpiar (){
    echo "Bash:Prompt:" > /samba/Console.txt
    echo "Esperando nuevo comando" > /samba/Salida.txt
}

subir (){
    mv "/samba/$1" "/samba/$nombre"
    echo "El archivo $1 ya se encuentra en el directorio personal" >
/samba/salida.txt
}

admin (){
    ejecutar=$(echo "$1" | cut -d' ' -f1)
    case "$ejecutar" in
    "Login")
        echo "$1"
        user=$(echo "$1" | cut -d' ' -f2)
        contra=$(echo "$1" | cut -d' ' -f3)
        consultar "$user" "$contra"
        limpiar
    ;;
    "Agregar")
        echo "$1"
        user=$(echo "$1" | cut -d' ' -f2)
        contra=$(echo "$1" | cut -d' ' -f3)
        agregar "$user" "$contra"
        limpiar
    ;;
    "Eliminar")
        echo "$1"
        user=$(echo "$1" | cut -d' ' -f2)
        eliminar "$user"
        limpiar
    ;;
    "Modificar")
        echo "$1"
        user=$(echo "$1" | cut -d' ' -f2)
        user2=$(echo "$1" | cut -d' ' -f3)
        contra=$(echo "$1" | cut -d' ' -f4)
        modificar "$user" "$user2" "$contra"
```

```
        limpiar
    ;;
    "Listar")
        echo "$1"
        user=$(echo "$1" | cut -d' ' -f2)
        $(./procesos.sh "$user")
        echo procesos.txt > /samba/Salida.txt
        limpiar
    ;;
    "Detener")
        echo "$1"
        user=$(echo "$1" | cut -d' ' -f2)
        pkill -u "$user"
        limpiar
    ;;
    "Subir")
        echo "$1"
        archivo=$(echo "$1" | cut -d' ' -f2)
        subir "$archivo"
        limpiar
    ;;
    "Files")
        echo "$1"
        direc=$(echo "$1" | cut -d' ' -f2
        ls "/samba/$direc"
    ;;
    esac
}

#conexión de Samba
while(true)
do

    comando=$(cat /samba/Console.txt)
    salida=$(cat /samba/Salida.txt)
    prompt=$(echo "$comando" | cut -d ':' -f1)
    if [[ "$prompt" == "Bash" ]]
    then
    sleep 3
    clear
    echo "$salida"
    run=$(echo "$comando" | cut -d':' -f3 )
    admin "$run"
    fi
done
```

LÍNEA CON LA QUE SE MONTA LOS RECURSOS COMPARTIDOS

```
PS C:\Users\JrJos> New-PSDrive -PSProvider FileSystem -Name "Z" -Root \\192.168.100.8\Samba -Persist
New-PSDrive : Ya se está utilizando el nombre del dispositivo local
En línea: 1 Carácter: 1
+ New-PSDrive -PSProvider FileSystem -Name "Z" -Root \\192.168.100.8\Sa ...
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : InvalidOperation: (Z:PSDriveInfo) [New-PSDrive], Win32Exception
    + FullyQualifiedErrorId : CouldNotMapNetworkDrive,Microsoft.PowerShell.Commands.NewPSDriveCommand


PS C:\Users\JrJos>
```