Scripts

Contains all the script I used for my administration

- Script - Backup Script for Home Directories and MySQL Databases
- Script - MySQL Dump Databases Separated by DB Name
- Script - Move Infected Emails to Quarantine and Notify Users
- Delete Old Emails and Notify User

Script - Backup Script for Home Directories and MySQL Databases

```
#!/bin/sh

# Home directory to backup must be absolute path, with trailing slash
home_dir='/home/'
# Target backup directory, must be absolute path, with trailing slash
backup_dir='/backups/'

# Database user
db_user='root'
# Database Password
db_pwd=''

cd "$home_dir"
# Get list of users based on home dir
users=`find . -maxdepth 1 -type d \( -iname "*" ! -iname "backups" ! -iname "lost+found" \) -
exec echo {} \; | sed "s#./##" | grep -v '^/home$'`
for user in $users; do
    # Skip if user string is empty
    if [ $user == "" -o $user == ".." -o $user == "." ]; then
        continue
    fi
    # Archive all files in directory
    archive="$backup_dir`date +%Y%m%d`.$user.tar.gz"
    tar czf "$archive" "$user"
done

# Database backup script
if [ ! -z "$db_pwd" ]; then
    databases=`mysql -u$db_user -p$db_pwd -e "SHOW DATABASES;" | tr -d "| " | grep -v
```

```
Database`
else
    databases=`mysql -u$db_user -e "SHOW DATABASES;" | tr -d "| " | grep -v Database`
fi
cd $backup_dir
for db in $databases; do
    if [[ "$db" != "information_schema" ]] && [[ "$db" != "performance_schema" ]] && [[ "$db"
!= "mysql" ]] && [[ "$db" != _* ]] ; then
        sql="`date +%Y%m%d`.$db.sql"
        echo "Dumping database: $db"
        if [ -z "$db_pwd" ]; then
            mysqldump -u$db_user $db > $sql
        else
            mysqldump -u$db_user -p$db_pwd $db > $sql
        fi
        tar -czf "`date +%Y%m%d`.$db.sql.tar.gz" $sql
        rm $sql
    fi
done
```

NOTE: Add this to `cronjob` to delete backups older than 90 days: `find . -type d -mtime +90 -exec rm {}\;`

Script - MySQL Dump Databases Separated by DB Name

```
#! /bin/bash

DUMP_EXEC="mysqldump" #path to mysqldump
MYSQL_EXEC="mysql" #path to mysql

MYSQL_USER="root" #db user
MYSQL_PASSWORD="" #db password

databases="$MYSQL_EXEC -u$MYSQL_USER"
if [ "$MYSQL_PASSWORD" ]; then
    databases="$databases -p$MYSQL_PASSWORD"
fi

eval "$databases -e 'show databases'" | while read dbname
do
    if [ "$dbname" ]; then
        echo "Dumping database: $dbname"
        dumpScript="$DUMP_EXEC --max_allowed_packet=1G -u$MYSQL_USER"
        if [ "$MYSQL_PASSWORD" ]; then
            dumpScript="$dumpScript -p$MYSQL_PASSWORD"
        fi
        eval "$dumpScript --complete-insert '$dbname' > '$dbname.sql'"
    fi
done
```

Script - Move Infected Emails to Quarantine and Notify Users

The following script will do the following:

1. Parse email headers from ClamScan Results
2. Move infected email to `$QUARANTINE` folder
3. Construct email messages
4. Email the users who has any infected emails in their mailbox

```
#! /bin/bash

ADMIN="admin@domain.com" # Admin email
QUARANTINE="/quarantine/directory/" # Quarantine folder with trailing slash
HEADER="The emails listed has been moved to quarantine and will be deleted after 30 days. If you have any concerns, please contact the server administrator"
FOOTER="This is an automated email through ClamScan results, please find the script details at 'https://wiki.twcloud.tech/books/linux/page/script---move-infected-emails-to-quarantine-and-notify-users'"

# Getting email information
[ -z "$1" ] && echo "File parameter missing" && exit 1
[ ! -f "$1" ] && echo "File not found / not a regular file" && exit 1
declare -A emails
while read i; do
    file=`echo "$i" | sed -e 's/\: \ .*FOUND//'`
    if [ ! -f "$file" ]; then
        continue
    fi
    infection=`echo "$i" | sed -n 's/\: \ .*FOUND//'`
    to=`cat "$file" | grep -m 1 "^Envelope-to\: \s\+" | sed 's/Envelope-to\: \ //' | grep -EiEio '\b[A-Z0-9._%+-]+@[A-Z0-9.-]+\.[A-Z]{2,4}\b'`
    # Try find To: header if Envelope-to: not found
```

```
    [[ -z "$to" ]] && to=`cat "$file" | grep -m 1 "^To\:\s\+" | sed 's/To\:\ //' | grep -EiEio
'\b[A-Z0-9._%+-]+@[A-Z0-9.-]+\.[A-Z]{2,4}\b'`
    from=`cat "$file" | grep -m 1 "^From\:\s\+" | sed 's/From\:\ //'`
    d=`cat "$file" | grep -m 1 "^Delivery-date\:\s\+" | sed 's/Delivery-date\:\ //'`
    subject=`cat "$file" | grep -m 1 "^Subject\:\s\+" | sed 's/Subject\:\ //'`

    # Send empty "$to" to admin
    [[ -z "$to" ]] && to="$ADMIN"

    # Construct email message
    [[ -z "${emails[$to]}" ]] && emails[$to]="$HEADER"
    emails[$to]="${emails[$to]}\n\nFrom: $from\nDate: $d\nSubject: $subject"

    # Move emails to quarantine
    mv "$file" "$QUARANTINE"
done < "$1"

# Notify email users that the emails are sent to quarantine
for k in "${!emails[@]}"; do
    echo -e "${emails[$k]}\n----------\n$FOOTER" | mail -s "Infected emails quarantined" -c
"$ADMIN" $k
done
```

Delete Old Emails and Notify User

```
ADMIN="admin@domain.com" # Admin email
DOMAIN="domain.com" # Domain name
HEADER="The emails listed has been moved to trash, and will be deleted on the 31st December every year"
FOOTER="This is an automated email generated through a script, please find the script details at 'https://wiki.twcloud.tech/books/linux/page/delete-old-emails-and-notify-user'"
REMOVE_FILE_AGE=60 # File age to remove in days
USER="user" # Username used to login to the hosting account
TRASH_FOLDER="/home/$USER/trashed_emails/" # Trash folder with trailing slash

# Getting email information
[ -z "$1" ] && echo "Email user parameter missing" && exit 1
[ ! -d "/home/$USER/mail/$DOMAIN/$1/cur" ] && echo "Email not found" && exit 1
# Declarations
declare -A emails

for file in $(find "/home/$USER/mail/$DOMAIN/$1/cur" -type f -mtime +${REMOVE_FILE_AGE} -print); do
    if [ ! -f "$file" ]; then
        continue
    fi
    to="$1@$DOMAIN"
    from=`cat "$file" | grep -m 1 "^From\:\s\+" | sed 's/From\:\ //'`
    d=`cat "$file" | grep -m 1 "^Delivery-date\:\s\+" | sed 's/Delivery-date\:\ //'`
    subject=`cat "$file" | grep -m 1 "^Subject\:\s\+" | sed 's/Subject\:\ //'`

    # Send empty "$to" to admin
    [[ -z "$to" ]] && to="$ADMIN"

    # Construct email message
    [[ -z "${emails[$to]}" ]] && emails[$to]="$HEADER"
    emails[$to]="${emails[$to]}\n\nFrom: $from\nDate: $d\nSubject: $subject"
```

```
    # Move emails to trash
    mv "$file" "$TRASH_FOLDER"
done

# Notify email users that the emails are sent to trash
for k in "${!emails[@]}"; do
    echo -e "${emails[$k]}\n----------\n$FOOTER" | mail -s "Inbox Cleared" -c "$ADMIN" $k
done
```