DRAFT

IT 341: Introduction to System Administration
Project #9: Automating the Backup Process

---

1. **Make sure your Ethernet cable is plugged into the right-hand side**
   This is the jack that communicates with our server, `it20`, whose domain has the name `it.cs.umb.edu`. (If need be, review your notes about **Ethernet jacks**)

2. **Log in to your virtual machine as** `sysadmin`

3. **Create the directory** `/guests`
   You need to create a directory that other machines can use for backups. This directory should be in the root directory `/`

```
cd /
sudo mkdir guests
```

4. **Make this directory available to all users**

```
sudo chmod 777 guests
```

5. **Go to** `/usr/local/bin`

```
cd /usr/local/bin
```

6. **Create the script** `autobackup.sh`

```
sudo nano autobackup.sh
```

   Enter the following:

```
#!  /bin/bash
#
# shell script to perform daily backups of /etc

if [ $# -lt 2 ]
then
    echo Usage:  $(basename $0)  BACKUP_HOST  BACKUP_DIRECTORY
    exit 1
fi

backup_host=$1
backup_dir=$2
```

```
date=$(date +%F)

rsync -azvv -e ssh /etc $backup_host.it.cs.umb.edu:$backup_dir/$date
```

7. **Make this file executable:**

```
sudo chmod  755  autobackup.sh
```

8. **Logout** sysadmin

```
exit
```

9. ***ssh* into** another **virtual machine with your personal account**
   It should be a machine where you can log in without a password, as of Project 6.
   Also, the ***sysadmin*** of the other VM will need to have completed Step 4 above, so that you can do Step 10...

> **VIRTUAL_MACHINE_NAME** refers to your own VM. For example, **itvm24-2a**
>
> **OTHER_VIRTUAL_MACHINE** refers to the other VM where you are placing your backups. For example, **itvm26-2a**

10. **Create a directory for your backups**

```
cd  /guests
mkdir  VIRTUAL_MACHINE_NAME
```

11. **Exit the ssh connection**

```
exit
```

12. **Run autobackup.sh**

```
autobackup.sh  OTHER_VIRTUAL_MACHINE  /guests/VIRTUAL_MACHINE_NAME
```

13. **Check to see if the script worked:**

```
ssh  YOUR_UNIX_USERNAME@OTHER_VIRTUAL_MACHINE  'ls /guests/VIRTUAL_MACHINE_NAME'
```

Here, we are using a feature of ***ssh*** that allows you to run a Unix command on the remote machine without logging in. You should see a directory whose name is today's date in **YYYY-MM-DD** format.

14. **Remove the backup directory on the other virtual machine you just created:**

```
ssh  YOUR_UNIX_USERNAME@OTHER_VIRTUAL_MACHINE  'rm -rf /guests/VIRTUAL_MACHINE_NAME/TODAYS_DATE'
```

15. **Check to *make sure* the backup directory has been removed:**

```
ssh  YOUR_UNIX_USERNAME@OTHER_VIRTUAL_MACHINE  'ls /guests/VIRTUAL_MACHINE_NAME'
```

16. **Create a `cron` job to run `autobackup.sh`**

(BEFORE CONTINUING: Read the bullet points below, as well as my note at the bottom regarding "Making **`crontab`** work with **`ssh-agent`**")

Run ***`crontab`***

```
crontab -e
```

- If you have not created **`cron`** jobs on this machine yet, then there should be nothing in this file.
- Create an entry to run **`autobackup.sh`** in 5 minutes.
- To do this, you will have to look at the time and pick a time 5 minutes ahead.
- Remember that **`cron`** uses a 24-hour clock, so 1 PM is written at 13.
- (To help with this, research the **format of a `crontab` File**)
- The command part of the cron job should look like this:

```
/usr/local/bin/autobackup.sh  OTHER_VIRTUAL_MACHINE  /guests/VIRTUAL_MACHINE_NAME
```

17. **Check the other virtual machine to make sure the backup worked:**

```
ssh  YOUR_UNIX_USERNAME@OTHER_VIRTUAL_MACHINE  'ls /guests/VIRTUAL_MACHINE_NAME'
```

18. **Take a snapshot of the current state of your virtual machine:**

Select *VM* ⟶ *Snapshot* ⟶ *Take snapshot*
In the *Name* box type

```
Project 9
```

then click *Take Snapshot*.

Making `crontab` work with `ssh-agent`

I have seen that **`crontab`** and **`ssh-agent`** do not always play well together. Specifically, even if you have an **`ssh-agent`** running, it may not recognize it

when running autobackup.sh from crontab. To make this work, I recommend the following steps:

1. On your VM, as **sysadmin**, execute the following command:
   `sudo apt-get install keychain`

2. On your VM, as **sysadmin**, add the following line to the start of your script code:
   `source $HOME/.keychain/`/bin/hostname`-sh`

3. On your VM, as *yourself* (i.e., your personal account), execute the following command:
   `keychain $HOME/.ssh/YOUR_PRIVATE_KEY`
   (**`YOUR_PRIVATE_KEY`** will probably be either **`id_dsa`** or **`id_rsa`**)

4. Finally, as *yourself*, create the task in ***crontab***, if you have not done so already. It should work now because the previous steps created an ssh agent that should persist after you log off (so long as your VM is still up and running).

5. If these steps do not work, then see if you can troubleshoot the issue and get it to work. (If so, document this in your admin log.) If that fails, create a second set of keys (this time, no passphrase), place the public key into your **authorized_keys** file, and remove the *keychain* line from the script.