Computer System Administration

Homework 3 – File Server

ychsiao

Overview (1/2)

- Image that you are a TA of a course, the professor wants you to build a file server that students can submit their homework to
- To prevent your colleagues from accidentally deleting files on the server, the snapshot and rollback features are needed

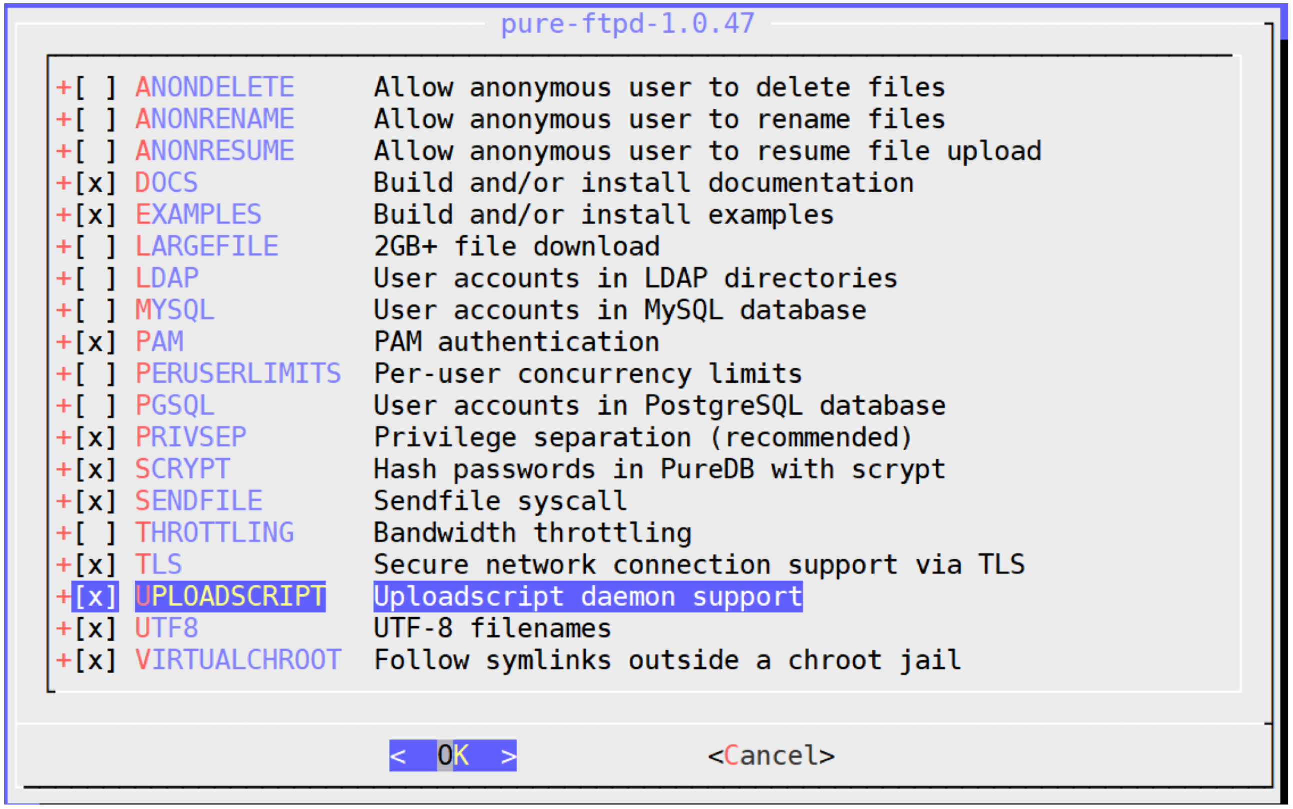
Overview (2/2)

- File Server (100% + 15%)
  - FTP Server (50%)
  - ZFS on /home/ftp (25% + 10% Bonus)
  - Pure-ftpd uploadscript with rc (25%+5% Bonus)

FTP server

Installation

- You can install pure-ftpd with pkg or port
- If you install with port, remember to compile it with "upload script" support



```
pure-ftpd-1.0.47

+[ ] ANONDELETE     Allow anonymous user to delete files
+[ ] ANONRENAME     Allow anonymous user to rename files
+[ ] ANONRESUME     Allow anonymous user to resume file upload
+[x] DOCS           Build and/or install documentation
+[x] EXAMPLES       Build and/or install examples
+[ ] LARGEFILE      2GB+ file download
+[ ] LDAP           User accounts in LDAP directories
+[ ] MYSQL          User accounts in MySQL database
+[x] PAM            PAM authentication
+[ ] PERUSERLIMITS  Per-user concurrency limits
+[ ] PGSQL          User accounts in PostgreSQL database
+[x] PRIVSEP        Privilege separation (recommended)
+[x] SCRYPT         Hash passwords in PureDB with scrypt
+[x] SENDFILE       Sendfile syscall
+[ ] THROTTLING     Bandwidth throttling
+[x] TLS            Secure network connection support via TLS
+[x] UPLOADSCRIPT   Uploadscript daemon support
+[x] UTF8           UTF-8 filenames
+[x] VIRTUALCHROOT  Follow symlinks outside a chroot jail

            <  OK  >            <Cancel>
```

Requirements (1/6) – Directories

Create three directories under */home/**ftp***

1. */home/ftp/**public*** :
   - ❑ Everyone can download & upload file
   - ❑ Everyone can mkdir, rmdir, delete except anonymous.
2. */home/ftp/**upload*** :
   - ❑ Everyone can upload & download
   - ❑ Everyone can mkdir except anonymous
   - ❑ Everyone can only delete & rmdir their own file or directory except anonymous and sysadm.
3. */home/ftp/**hidden*** :
   - ❑ Create a directory called "treasure" inside *hidden*
   - ❑ Create a file called "secret" inside *hidden/**treasure***
   - ❑ Anonymous can't list */home/ftp/hidden* but can enter *hidden/treasure* and show *hidden/treasure/**secret***

Requirements (2/6)

Create four users

1. Create a system user "sysadm"
   - ❑ Can login by SSH
   - ❑ Password is your student ID
   - ❑ Full access to */home/ftp* and subdirectories under *"ftp"*
2. Create two virtual users "ftp-vip1", "ftp-vip2"
   - ❑ Password is your student ID
   - ❑ Can only delete files in */home/ftp/**upload*** which are created by **themselves**.
   - ❑ Other permissions are same as sysadm.

Requirements (3/6)

3. Anonymous
   - ❑ Can’t create any directories
   - ❑ Can’t delete any files & directories
   - ❑ Can’t list */home/ftp/hidden* but can enter *hidden/treasure* and show *hidden/treasure/**secret***

Other requirements

1. Your ftp server should support Explicit FTP over TLS (FTPES)
2. All accounts are chrooted (***/home/ftp*** is the root directory)

Requirements (4/6)

sysadm	*public/*	*upload/*
upload	**V**	**V**
delete	**V**	**V**
mkdir	**V**	**V**
rmdir	**V**	**V**
download	**V**	**V**

V : full access   △ : only the owner has permission
X : permission denied

Requirements (5/6)

ftp-vip	*public/*	*upload/*
upload	**V**	**V**
delete	**V**	△
mkdir	**V**	**V**
rmdir	**V**	△
download	**V**	**V**

V : full access △ : only the owner has permission
X : permission denied

Requirements (6/6)

Anonymous	*public/*	*upload/*
upload	**V**	**V**
delete	**X**	**X**
mkdir	**X**	**X**
rmdir	**X**	**X**
download	**V**	**V**

V : full access　　△ : only the owner has permission

X : permission denied

Grading (1/3)

- ❑ FTP over TLS (5%)
- ❑ sysadm (15%)
  - login from ssh (4%)
  - Full access to ***"public"*** (3%)
  - Full access to ***"upload"*** (4%)
  - Full access to ***"hidden"*** (4%)

Grading (2/3)

- ftp-vip1, ftp-vip2 (15%)
  - Chrooted ***(/home/ftp)*** (4%)
  - Full access to ***"public"*** (3%)
  - Full access to ***"upload"***, but can only delete their own files and directories. (4%)
  - Full access to ***"hidden"*** (4%)

Grading (3/3)

- Anonymous login (15%)
  - chroot ***(/home/ftp)*** (4%)
  - Can only upload and download from ***"public"*** (3%)
  - Can only upload and download from ***"upload"*** (4%)
  - Hidden directory ***"/home/ftp/hidden"*** problem: can enter but can't retrieve directory listing (4%)

Hint

- README
  - */usr/local/share/doc/pure-ftpd/**
- Accounts related
  - Virtual user
  - pure-pw(8)
  - pure-pwconvert(8)
  - README.Virtual-Users
- If `pure-ftpd` is not working
  - Check your pure-ftpd.conf

ZFS on /home/ftp

Requirements (1/8)

- ❑ Enable ZFS service
  - Reboot and everything is fine (ZFS still mount)
- ❑ Add two new hard disks to create a mirror (RAID 1) storage called “mypool”
  - Use the added hard disk to create a mirror storage pool using the zpool command
- ❑ Create ZFS datasets
  - Create a dataset **mypool** mount on /home/ftp
  - Set lz4 compression, atime=off to all datasets
  - Create mypool/public, mypool/upload, mypool/hidden

Requirement (2/8): Zbackup

❑ Automatic Snapshot Script

- Usage:
  - Create: `zbackup DATASET [ROTATION_CNT]`
  - List: `zbackup -l|--list [DATASET|ID|DATASET ID]`
  - Delete: `zbackup -d|--delete [DATASET|ID|DATASET ID]`
  - Export: `zbackup -e|--export [DATASET|ID|DATASET|ID]`
  - Import: `zbackup -i|--import [DATASET|ID|DATASET ID] FILENAME`

Requirement (3/8): Zbackup

❑ Specification - Create (Default)

- No more than **rotation count** snapshots per dataset
- If **rotation count** has reached, delete the oldest one
- If no **rotation count** specified, the default rotation shoud be 20
- While creating a new snapshot, print log message to stdout
- You’re snapshot should include the dataset name and date

```
[ychsiao@tSA ~]$ zbackup -l
ID  DATASET         TIME
[ychsiao@tSA ~]$ sudo zbackup mypool/public
Snap mypool/public@2019-10-31-13:52:01
[ychsiao@tSA ~]$ sudo zbackup mypool/public
Snap mypool/public@2019-10-31-13:52:20
[ychsiao@tSA ~]$ sudo zbackup mypool/public 1
Snap mypool/public@2019-10-31-13:52:47
Destroy mypool/public@2019-10-31-13:52:01
Destroy mypool/public@2019-10-31-13:52:20
[ychsiao@tSA ~]$
```

Requirement (4/8): Zbackup

❑ Specification - List

- The list should include id, dataset and time. Sorted by time.
- If the variable is **DATASET**, list the snapshot of that dataset
- If the variable is **ID**, list only the snapshot with that **id**
- Otherwise, list all of the snapshot

```
[ychsiao]@[tSA][~]$ zbackup -l
ID  DATASET        TIME
1   mypool/public  2019-10-26-18:27:39
2   mypool/public  2019-10-26-18:27:41
3   mypool         2019-10-26-18:30:59
[ychsiao]@[tSA][~]$ zbackup -l 3
ID  DATASET  TIME
3   mypool   2019-10-26-18:30:59
[ychsiao]@[tSA][~]$ zbackup -l mypool/public
ID  DATASET        TIME
1   mypool/public  2019-10-26-18:27:39
2   mypool/public  2019-10-26-18:27:41
[ychsiao]@[tSA][~]$ zbackup -l mypool/public 2
ID  DATASET        TIME
2   mypool/public  2019-10-26-18:27:41
[ychsiao]@[tSA][~]$
```

Requirement (5/8): Zbackup

- Specification - Delete
  - Delete snapshots created by zfs
  - If **DATASET** is specified, delete the whole dataset
  - If **ID** is specified, delete the dataset with that id
  - Otherwise, delete all snapshot of the dataset

```
[ychsiao]@[tSA][~]$ zbackup -l
ID  DATASET        TIME
1   mypool/public  2019-10-26-18:27:39
2   mypool/public  2019-10-26-18:27:41
3   mypool         2019-10-26-18:50:30
4   mypool         2019-10-26-20:11:32
5   mypool         2019-10-26-20:11:34
[ychsiao]@[tSA][~]$ sudo zbackup -d 1
Destroy mypool@2019-10-26-18:50:30
[ychsiao]@[tSA][~]$ sudo zbackup -d mypool 2
Destroy mypool@2019-10-26-20:11:34
[ychsiao]@[tSA][~]$ sudo zbackup -d mypool/public
Destroy mypool/public@2019-10-26-18:27:39
Destroy mypool/public@2019-10-26-18:27:41
[ychsiao]@[tSA][~]$ sudo zbackup -d
Destroy mypool@2019-10-26-20:11:32
```

Requirement (6/8): Zbackup

- Specification - Export
  - Must specify **dataset**
  - **ID** defaults to 1
  - Compress with gzip
  - Encrypt with aes256 (Hint: Use openssl; Ask user to input password)
  - The filename for example: `dataset@2019-10-26-16:20:48.gz.enc`
  - You can put your export file at anywhere, as long as you can find it

```
[ychsiao@tSA ~]$ sudo zbackup -e mypool/public 1
enter aes-256-cbc encryption password:
Verifying - enter aes-256-cbc encryption password:
Export mypool/public@2019-10-29-13:07:36 to
/tmp/mypool/public@2019-10-29-13:07:36.gz.enc
```

Requirement (7/8): Zbackup

- ❑ Specification
  - Import
    - Must specify **filename** and **dataset**
    - **filename** is the file exported by zbackup
    - Ask user to input password
    - Load the snapshot to the dataset

```
[ychsiao@tSA ~]$ sudo zbackup -i \
/tmp/mypool/public\@2019-10-29-13\:07\:36.gz.enc\
mypool/public2
enter aes-256-cbc decryption password:
[ychsiao@tSA ~]$ zbackup -l
ID  DATASET         TIME
1   mypool/public   2019-10-29-13:07:36
2   mypool/public2  2019-10-29-17:43:55
[ychsiao@tSA ~]$ ls /home/ftp/
hidden  public  public2 upload
```

Requirement (8/8): Zbackup

- Error detection
  - Snap dataset@create_time of the new snap, e.g.,
    - Snap mypool@2019-10-25-09:36:02
  - Destroy dataset@create_time of the deleted snap, e.g.,
    - Destroy mypool@2019-10-25-09:31:55
  - Export dataset@create_time to your storage directory, e.g.
    - Export mypool/public@2019-10-25-09:31:55 to /tmp/mypool/public@2019-10-25-09:31:55.gz.enc
  - Import dataset@create_time.gz.enc to a new dataset@create_time
    - Import /tmp/mypool/public@2019-10-25-09:31:55.gz.enc to mypool/public2
  - Log must contain the action (e.g. snap), time and dataset name, but the format has no requirement
  - For any non-define operation, just print an error message and exit

Grading

- ZFS on /home/ftp (25% + 10% Bonus)
  - Create a mirror storage (2%)
  - Create all dataset and set up correctly (3%)
  - Zbackup (20%+10%)
    - Create (5%)
    - List, Delete (10%)
    - Export, Import (+10%)
    - Error detection (5%)

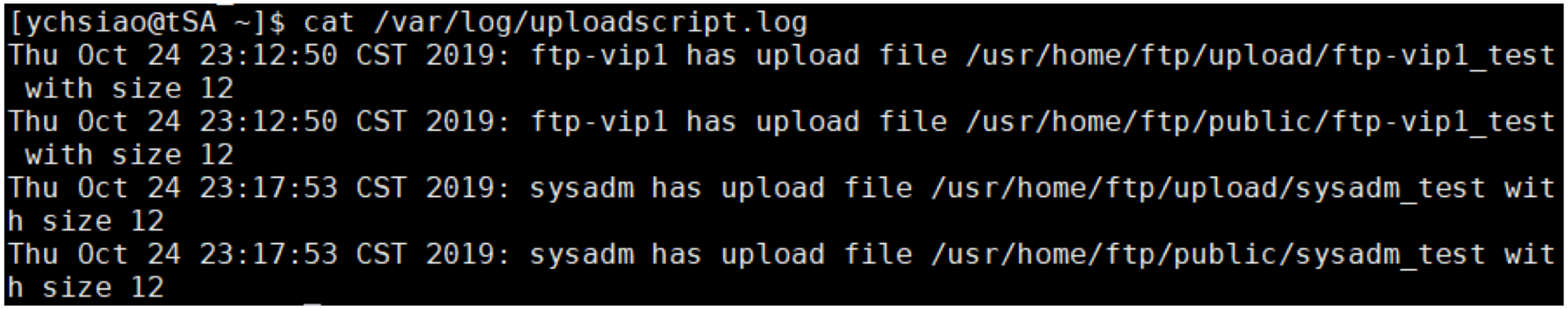
Hint

- It will be much easier if you implement `Delete`, `Export`, `Import` with a well coding `List`
- Check handbook first
  - https://www.freebsd.org/doc/en/books/handbook/zfs-zfs.html
  - https://www.freebsd.org/doc/en/books/handbook/zfs-term.html

Pure-ftpd uploadscript with RC

Requirements (1/5) : uploadscript

- Create a uploadscript.sh for recording every uploading into */var/log/uploadscript.log*
- The log message should required upload time, upload user, upload file name, and file size

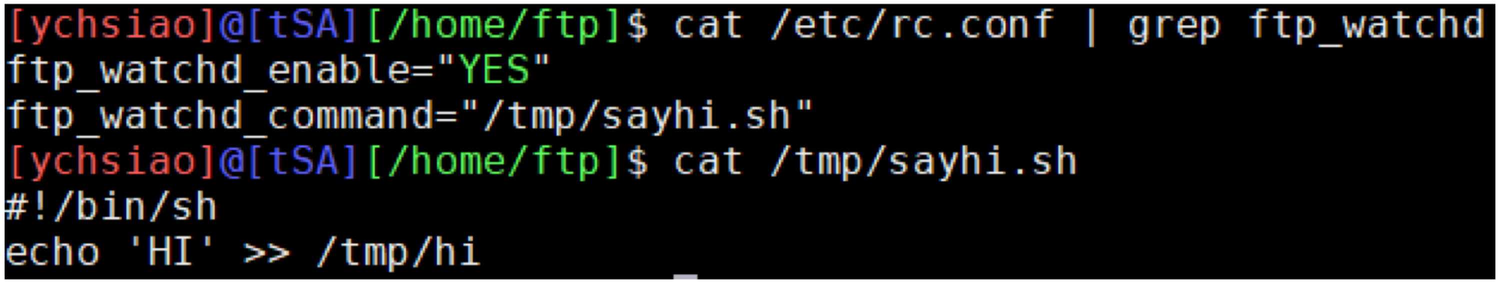


```
[ychsiao@tSA ~]$ cat /var/log/uploadscript.log
Thu Oct 24 23:12:50 CST 2019: ftp-vip1 has upload file /usr/home/ftp/upload/ftp-vip1_test
 with size 12
Thu Oct 24 23:12:50 CST 2019: ftp-vip1 has upload file /usr/home/ftp/public/ftp-vip1_test
 with size 12
Thu Oct 24 23:17:53 CST 2019: sysadm has upload file /usr/home/ftp/upload/sysadm_test wit
h size 12
Thu Oct 24 23:17:53 CST 2019: sysadm has upload file /usr/home/ftp/public/sysadm_test wit
h size 12
```

- Create the service `ftp-watchd` which enable to run the command after a successful upload
  - Execute uploadscript.sh when a file is successfully uploaded to the FTP Server
  - Passing arguments described in rc.conf
    - Don’t hardcore the command, the command can be specified in rc.conf

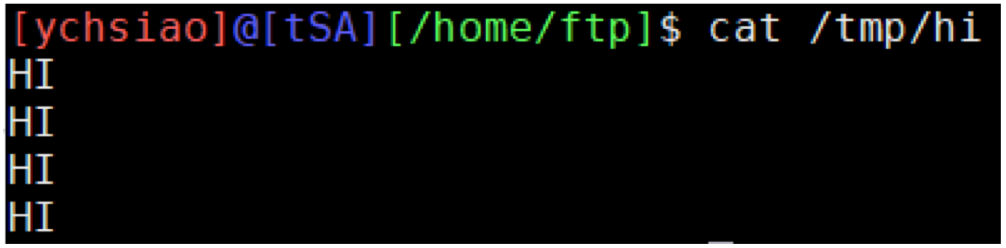
Requirements (2/5) : uploadscript

- Execute a command defined in rc.conf whenever a file is uploaded
- For example, set the following command in rc.conf :
  - echo “HI” and write to a file /tmp/hi



```
[ychsiao]@[tSA][/home/ftp]$ cat /etc/rc.conf | grep ftp_watchd
ftp_watchd_enable="YES"
ftp_watchd_command="/tmp/sayhi.sh"
[ychsiao]@[tSA][/home/ftp]$ cat /tmp/sayhi.sh
#!/bin/sh
echo 'HI' >> /tmp/hi
```

  - After four successful uploads, the command should be invoked as expected :



```
[ychsiao]@[tSA][/home/ftp]$ cat /tmp/hi
HI
HI
HI
HI
```

Requirements (3/5) : RC script

- You should write an rc script ftp-watchd as a daemon to start the pure-uploadscript program
  - pure-uploadscript should be run in the background when ftp-watchd is started

```
[ychsiao@tSA ~]$ service ftp-watchd status
ftp_watchd is running as pid 9021.
[ychsiao]@[tSA][~]$ ps aux | grep pure-uploadscript
root  9021  0.0  0.2  15560  6656  -  Is  01:16
0:00.00 /usr/local/sbin/pure-uploadscript
```

- Your service must support these operation
  - $ service ftp-watchd start
  - $ service ftp-watchd stop
  - $ service ftp-watchd restart
  - $ service ftp-watchd status
  - $ service ftp-watchd poll

Requirements (4/5) : RC script

- Requires a pid file to indicate which process to stop

```
[ychsiao]@[tSA][~]$ cat /var/run/pure-uploadscript.pid
8210
```

- You should display as following format while using each command
  - Service start

```
[ychsiao]@[tSA][~]$ sudo service ftp-watchd start
Starting ftp_watchd.
```

  - Service stop

```
[ychsiao]@[tSA][~]$ sudo service ftp-watchd stop
Kill: 8210
```

Requirements (5/5) : RC script

- Service restart

```
[ychsiao]@[tSA][~]$ sudo service ftp-watchd restart
Kill: 8191
Starting ftp_watchd.
```

- Service status

```
[ychsiao]@[tSA][~]$ sudo service ftp-watchd status
ftp_watchd is running as pid 8210.
```

- Server poll

```
[ychsiao]@[tSA][~]$ sudo service ftp-watchd poll
Waiting for PIDS: 8210
```

Hint

- ❑ Enable upload script under pure-ftpd.conf
  - CallUploadScript yes
- ❑ For pure-uploadscript, you can manually start the daemon by following command

```
pure-uploadscript -B -r /your/uploadscript/to/execute
```

- ❑ pure-uploadscript(8)

Grading

- pure-uploadscript (10%)
  - pure-uploadscript should be activated (5%)
  - Record should be written in log file after any successful upload (5%)
- ftp-watchd (15%+5%)
  - rc.d (5%)
    - Auto start on boot
  - Service operation work correctly (10%+5%)
    - User can specify command in rc.conf (5%)
    - start/status/restart (5%)
    - stop/poll (+5%)

Reminder

- ❑ Demo with root is not allowed
  - Please use sudo
- ❑ File/directory permissions are important
  - Owner, group, other
  - Read, write, execute
  - Set UID, set GID, sticky bit
- ❑ Our demo will run on our intranet server *savpn.nctu.me*. Make sure your connection is available to the server. You can test it by the link below:
  - http://savpn.nctu.me:8080/

  If the connect is not successful, check for your wireguard settings

Deadline

- You do not need to submit anything
- Due(Demo): 2019/11/27 (三)

Help!

- ❑ Email to ta@nasa.cs.nctu.edu.tw
- ❑ Do not send mail to New E3 https://e3new.nctu.edu.tw/ !!!
- ❑ Office hour: 3GH at EC318