Homework 3
File Server & Backup

phlin, rzhung, hslin

國立陽明交大資工系資訊中心

Computer Center of Department of Computer Science, NYCU

Outline

- HW 3-1: File server
- HW 3-2: Pure-ftpd uploadscript with advance logging
- HW 3-3: ZFS & Backup

HW 3-1: File Server (25%)

HW 3-1: Requirement (1/4)

Use **pure-ftpd** to build a file server; create 3 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***

HW 3-1:

Create users

1. Create a system user “sysadm”
   - Can login by SSH & FTP
   - Password is your IP without dots, e.g. password=1011301 when IP=10.113.0.1
   - Full access to */home/ftp* and subdirectories under “ftp”
2. Create two virtual users “ftp-vip1”, “ftp-vip2”
   - Password is your IP without dots, e.g. password=1011301 when IP=10.113.0.1
   - Can only delete files in */home/ftp/**upload*** which are created by themselves
   - Other file permissions are same as sysadm
3. Anonymous with no password
   - Can’t create any directories, and can’t delete any files & directories
   - Can’t list */home/ftp/hidden* but can enter *hidden/treasure* and show *hidden/treasure/**secret***

HW 3-1:

Other requirements

- Your ftp server should support Explicit FTP over TLS (FTPES)
- All accounts except sysadm are chrooted (***/home/ftp*** is the root directory)

HW 3-1:

	sysadm		**ftp-vip**		**anonymous**	
	public/	*upload/*	*public/*	*upload/*	*public/*	*upload/*
upload	✔	✔	✔	✔	✔	✔
download	✔	✔	✔	✔	✔	✔
mkdir	✔	✔	✔	✔	✘	✘
rmdir	✔	✔	✔	▲	✘	✘
delete	✔	✔	✔	▲	✘	✘

✔: full access ▲: only the owner has permission ✘: permission denied

HW 3-1: Log

- Save all ftp log (login, operate … etc) into /var/log/pureftpd/pureftpd.log
- Filter only ftp login log into /var/log/pureftpd/login.log

Hint: syslog, VerboseLog

```
[rzhung@nfs ~]$ cat /var/log/pureftpd/pureftpd.log
Nov 10 11:40:40 nfs pure-ftpd[33316]: (?@192.168.202.250) [INFO] New connection from
192.168.202.250
Nov 10 11:40:40 nfs pure-ftpd[33316]: (?@192.168.202.250) [INFO] sysadm is now logged in
Nov 10 11:40:40 nfs pure-ftpd[33316]: (sysadm@192.168.202.250) [NOTICE]
/home/ftp//public/non.exe uploaded  (15062 bytes, 41.09KB/sec)
Nov 10 11:41:34 nfs pure-ftpd[33316]: (sysadm@192.168.202.250) [NOTICE]
/home/ftp//public/pureftpd.viofile downloaded  (102 bytes, 967.11KB/sec)
Nov 10 11:42:34 nfs pure-ftpd[33316]: (sysadm@192.168.202.250) [INFO] Logout.
```

```
[rzhung@nfs ~]$ cat /var/log/pureftpd/login.log
Nov 10 10:22:46 nfs pure-ftpd[33150]: (?@10.113.0.254) [INFO] sysadm is now logged in
Nov 10 11:36:09 nfs pure-ftpd[33290]: (?@192.168.202.250) [INFO] sysadm is now logged in
```

HW 3-1: Grading (14%)

- FTP over TLS (3%)
- sysadm login
  - login from ssh (1%)
  - Full access to "***public***", "***upload***", "***hidden***" (3%)
- ftp-vip1, ftp-vip2 login
  - Full access to "***public***", "***hidden***" (3%)
  - Full access to "***upload***", but can only delete their own files and directories. (4%)

HW 3-1: Grading (11%)

- Chrooted (***/home/ftp***) all accounts except sysadm (2%)
- Anonymous login
  - Can only upload and download from "***public***", "***upload***" (2%)
  - Hidden directory "***/home/ftp/hidden***" problem:
    can enter but can't retrieve directory listing (3%)
- Log
  - /var/log/pureftpd/pureftpd.log (1%)
  - /var/log/pureftpd/login.log (3%)

HW 3-1: 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

HW 3-2: pure-ftpd uploadscript With Adv. Logs (35%)

HW 3-2: Requirements-Script (1/6)

- Create an “uploadscript.sh” for filtering every file uploaded. (3%)
  - Files with extension .exe is violated
    - Move these files to *hidden/.exe* created at HW3-1

```
[rzhung@nfs ~]$ sudo ls /home/ftp/hidden/.exe/
test.exe
```

- Log violation of uploadscript into */home/ftp/public/pureftpd.viofile* (2%)
  - Format -

    timestamp **hostname** **programname**: **filename** violate file detected. Uploaded by **upload_user**.

```
[rzhung@nfs ~]$ cat /home/ftp/public/pureftpd.viofile
Nov 10 11:41:29 nfs ftpuscr[33351]: /home/ftp/public/test.exe violate file detected. Uploaded by
sysadm.
```

HW 3-2: Requirements-Service (2/6)

- Create a service “ftp-watchd” which enables running a command after a successful upload (5%)
  - The name of the service should be exactly the same as “ftp-watchd”
  - Execute uploadscript.sh when a file is successfully uploaded to the FTP server
  - Automatically start ftp-watchd when host boots up

HW 3-2: Requirements-Service (3/6)

- `uploadscript` feature in pure-ftpd should be activated `(5%)`
- You should write an rc or systemd 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
- Your service must support these operation: `(5%)`
  - `$ service ftp-watchd start`
  - `$ service ftp-watchd stop`
  - `$ service ftp-watchd restart`
  - `$ service ftp-watchd status`

HW 3-2: Requirements-Service (4/6)

- Require a pid file to indicate which process to stop

```
[rzhung@nfs ~]$ cat /var/run/pure-uploadscript.pid
20878
```

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

```
[rzhung@nfs ~]$ sudo service ftp-watchd start
Starting ftp-watchd.
```

  - Service stop

```
[rzhung@nfs ~]$ sudo service ftp-watchd stop
Kill: 20878
```

HW 3-2: Requirements-Service (5/6)

- Service restart

```
[rzhung@nfs ~]$ sudo service ftp-watchd restart
Kill: 3458
Starting ftp-watchd.
```

- Service status

```
[rzhung@nfs ~]$ sudo service ftp-watchd status
ftp-watchd is running as pid 3477.
```

HW 3-2: Requirements-Syslog (6/6)

- Please use syslog to handle logs of uploadscript. (10%)
  - “Processname” should be ftpuscr
  - Syslog settings for uploadscript should be written in *syslog.d/ftpuscr.conf*
  - To reduce the chance of conflicting with other service logs, please use the “local facility” - `local0` to handle logs.
- All logs should be labeled with proper level (5%)
  - Conditions that are not error conditions, but should possibly be handled specially.

```
[rzhung@nfs ~]$ cat /home/ftp/public/pureftpd.viofile
Nov 10 11:41:29 nfs ftpuscr[33351]: /home/ftp/public/test.exe violate file detected. Uploaded by
sysadm.
```

HW 3-2: Grading (Bonus +5%)

- You should finish all basic features before the bonus
- Using the HW3-1 Log section (pureftpd/login.log)
  - Add IP information for logs.

```
[rzhung@nfs ~]$ cat /home/ftp/public/pureftpd.viofile
Nov 10 11:41:29 nfs ftpuscr[33351]: /home/ftp/public/test.exe violate file detected. Uploaded by
sysadm. From 192.168.202.250.
```

  - Using the latest login log if there is more than one records in pureftpd/login.log

```
[rzhung@nfs ~]$ cat /var/log/pureftpd/login.log
Nov 10 10:22:46 nfs pure-ftpd[33150]: (?@10.113.0.254) [INFO] sysadm is now logged in
Nov 10 11:36:09 nfs pure-ftpd[33290]: (?@192.168.202.250) [INFO] sysadm is now logged in
```

HW 3-2: 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)

HW 3-3: ZFS & Backup (40%)

Requirements (1/8)

- Enable ZFS service
  - Reboot and everything is fine (ZFS still mounted)

- Add two new hard disks and create a mirror pool called "mypool"
  - Mount **mypool** on */home/ftp*

- Create ZFS datasets
  - Set lz4 compression, atime=off to all datasets
  - Create **mypool/public**, **mypool/upload**, **mypool/hidden**

Requirements (2/8)

- Automatic Snapshot Script: **zfsbak**
  - Add your script to $PATH
    - Allow to execute zfsbak with command ‵zfsbak ″, not ‵./zfsbak ″
  - Usage:
    - Create: zfsbak DATASET [ROTATION_CNT]
    - List: zfsbak –l|--list [DATASET|ID|DATASET ID]
    - Delete: zfsbak –d|--delete [DATASET|ID|DATASET ID]
    - Export: zfsbak –e|--export DATASET [ID]
    - Import: zfsbak –i|--import FILENAME DATASET

```
phlin@nfs:~ % zfsbak
Usage:
- create: zfsbak DATASET [ROTATION_CNT]
- list: zfsbak -l|--list [DATASET|ID|DATASET ID]
- delete: zfsbak -d|--delete [DATASET|ID|DATASET ID]
- export: zfsbak -e|--export DATASET [ID]
- import: zfsbak -i|--import FILENAME DATASET
```

Requirements (3/8)

- Specification - Create (Default)
  - Must specify **dataset**
  - If no rotation count is specified, use 20 as default
  - No more than rotation count snapshots per dataset
  - If rotation count is reached, delete the oldest one
  - Your snapshot should include the dataset name and date
  - When snapshot is the same as previous one, not to do this snapshot {Hint : zfs diff}

```
phlin@nfs:~ % zfsbak -l
ID DATASET TIME
phlin@nfs:~ % sudo zfsbak mypool/public
Snap mypool/public@2021-09-25-17:39:36
phlin@nfs:~ % sudo zfsbak mypool/public
Snap mypool/public@2021-09-25-17:39:40
phlin@nfs:~ % sudo zfsbak mypool/public 1
Snap mypool/public@2021-09-25-17:39:45
Destroy mypool/public@2021-09-25-17:39:36
Destroy mypool/public@2021-09-25-17:39:40
```

```
phlin@nfs:~ % zfsbak -l
ID DATASET TIME
phlin@nfs:~ % sudo zfsbak mypool/public
Snap mypool/public@2021-09-25-17:30:12
phlin@nfs:~ % sudo zfsbak mypool/public
Snapshot is the same as latest one!
phlin@nfs:~ % zfsbak -l
ID  DATASET         TIME
1   mypool/public   2021-09-25-17:30:12
```

Requirements (4/8)

- Specification – List
  - List snapshots created by zfs. **Sorted by time**
  - If only **ID** is specified, list only the snapshot with that **id**
  - If only **DATASET** is specified, list all snapshots of that dataset
  - If **DATASET** and **ID** are specified, list only the snapshot with that **dataset** and **id**
  - Otherwise, list all snapshots

```
phlin@nfs:~ % zfsbak -l
ID  DATASET          TIME
1   mypool/public    2021-09-25-17:38:12
2   mypool/public    2021-09-25-17:38:45
3   mypool/upload    2021-09-25-18:39:06
4   mypool/upload    2021-09-25-18:39:15
5   mypool/upload    2021-09-25-18:39:22
```

```
phlin@nfs:~ % zfsbak -l 3
ID  DATASET          TIME
3   mypool/upload    2021-09-25-18:39:06
phlin@nfs:~ % zfsbak -l mypool/upload
ID  DATASET           TIME
1   mypool/upload    2021-09-25-18:39:06
2   mypool/upload    2021-09-25-18:39:15
3   mypool/upload    2021-09-25-18:39:22
phlin@nfs:~ % zfsbak -l mypool/upload 2
ID  DATASET           TIME
2   mypool/upload    2021-09-25-18:39:15
```

Requirements (5/8)

- Specification – Delete
  - Delete snapshots created by zfs
  - If only **ID** is specified, delete the dataset with that **id**
  - If only **DATASET** is specified, delete all snapshots of that dataset
  - If **DATASET** and **ID** are specified, delete only the snapshot with that **dataset** and **id**
  - Otherwise, delete all snapshots
  - Can specify multi snapshot and delete

```
phlin@bsdzfs:~ % zfsbak -l
ID  DATASET          TIME
1   mypool/public    2021-09-25-17:55:17
2   mypool/public    2021-09-25-17:55:33
3   mypool/upload    2021-09-25-19:39:27
4   mypool/upload    2021-09-25-19:39:29
5   mypool/upload    2021-09-25-19:39:32
6   mypool/upload    2021-09-25-19:40:37
```

```
phlin@bsdzfs:~ % zfsbak -d 3
Destroy mypool/upload@2021-09-25-19:39:27
phlin@bsdzfs:~ % zfsbak -d mypool/upload 2
Destroy mypool/upload@2021-09-25-19:39:32
phlin@bsdzfs:~ % zfsbak -d mypool/public
Destroy mypool/public@2021-09-25-17:55:17
Destroy mypool/public@2021-09-25-17:55:33
phlin@bsdzfs:~ % zfsbak -d
Destroy mypool/upload@2021-09-25-19:39:29
Destroy mypool/upload@2021-09-25-19:40:37
```

```
phlin@bsdzfs:~ % zfsbak -d mypool/public 1 3 4
Destroy mypool/public@2021-09-25-19:54:49
Destroy mypool/public@2021-09-25-19:55:31
Destroy mypool/public@2021-09-25-19:55:37
```

Requirements (6/8)

- Specification – Export (Bonus)
  - Must specify **dataset**
  - **ID** defaults to 1
  - Compress with gzip
  - Encrypt with aes256 (Hint: Use openssl; Ask user to input password)
  - A filename example: `dataset@2021-09-25-20:05:27.gz.enc`
  - Put the export file at the user's home directory.

```
phlin@bsdzfs:~ % sudo zfsbak -e mypool/public 1
enter aes-256-cbc encryption password:
Verifying - enter aes-256-cbc encryption password:
Export mypool/public@2021-09-25-20:05:27 to ~/mypool_public@2021-09-25-20:05:27.gz.enc
```

Requirements (7/8)

- Specification – Import (Bonus)
  - Must specify **filename** and **dataset**
  - **filename** is the file exported by zfsbak
  - Ask user to input password
  - Load the snapshot to the dataset

```
phlin@bsdzfs:~ % sudo zfsbak -i ~/mypool_public@2021-09-25-20:05:27.gz.enc mypool/public2
enter aes-256-cbc encryption password:
Import ~/mypool_public@2021-09-25-20:05:27.gz.enc to mypool/public2
phlin@bsdzfs:~ % zfsbak -l
ID  DATASET         TIME
1   mypool/public   2021-09-25-21:03:27
2   mypool/public   2021-09-25-21:03:35
3   mypool/public2  2021-09-25-21:12:22
phlin@bsdzfs:~ % ls /home/ftp
hidden public public2 upload
```

Requirements (8/8)

- Log
  - Must contain the action (e.g. snap), dataset name and time
    - Print "**Snap `dataset@create_time`**" after creating the new snapshot, e.g.,
      - Snap mypool/public@ 2021-09-25-20:05:27
    - Print "**Destroy `dataset@create_time`**" after destroying the deleted snapshot, e.g.,
      - Destroy mypool/public@ 2021-09-25-20:05:27
    - **(Bonus)** Print "**Export `dataset@create_time` to `file_location`**" after exporting the target snapshot, e.g.,
      - Export mypool/public@2021-09-25-20:05:27 to ~/mypool_public@2021-09-25-20:05:27.gz.enc
    - **(Bonus)** Print "**Import `target_file` to `dataset`**" after importing the target file, e.g.,
      - Import mypool_public@2021-09-25-20:05:27.gz.enc to mypool/public2
  - For any undefined operation, just print the error message and exit

Grading (40/40%, Bonus + 6%)

- Create a mirror storage (2%)
- Create all dataset and set up correctly (2%)
- zfsbak
  - Usage (3%)
  - Create (8%)
  - List (12%)
  - Delete (8%)
  - Log (5%)
  - Export, Import (include log) (Bonus +6%) (Both complete)

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

Attention!

- Due date: 2021-12-06T23:59:59+08:00
- Online Judge open date: 2021-11-22
- Email us if you finish bonus, we will judge manually
  - ta@nasa.cs.nctu.edu.tw

Help me!

- TA time: 3 GH at EC 324 (PC Lab)
- Questions about this homework
  - Ask them on https://groups.google.com/g/nctunasa
  - We MIGHT give out hints on google group
    - Be sure to join the group :D
  - Do not use E3 to email us

Good Luck!