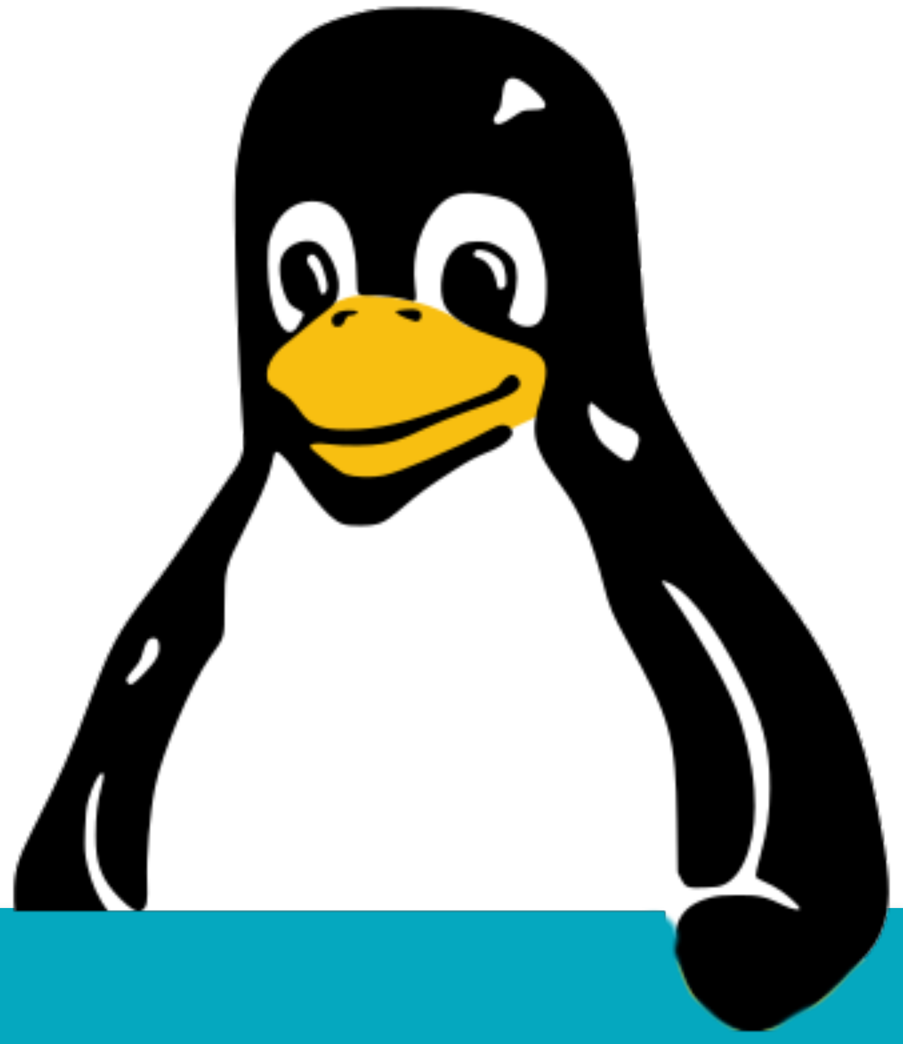
Linux, day 3



Objectives covered

Objective	Summary	Book
2.5	File permissions	15
3.1	Common scripting utilities	4
3.3	Git operations	27
3.4	Advanced Git operations	27

LAB: Files and directories

Command hints

sudo	Switch User and DO
mkdir	MaKe DIRectory
nano	Friendly editor
vi	Less-than-friendly editor
cp	CoPy
mv	MoVe
rm	ReMove
man	MANual (documentation)

Assignment

- Create two new directory trees:
  - "*~/staff/files*" and "*~/dummies/files*"
- Use "nano" or "vi" to put some text into:
  - *~/staff/files/staff-demo.txt*
  - *~/dummies/files/dummy-demo.txt*
- Move:
  - *~/staff/* to */home/staff/*
  - *~/dummies* to */home/dummies*

Spoilers

```
$ cd ~
$ mkdir -p staff/files dummies/files
$ vi staff/files/staff-demo.txt
$ vi dummies/files/dummies-demo.txt
$ sudo mv staff /home/
$ sudo mv dummies /home/
```

LAB: File permissions

Command hints

chmod	CHange MODe
chown	CHange OWNer
chgrp	CHange GRouP

Assignment

- */home/staff* and contents should have group "staff".
  - New files should automatically get group "staff".
  - Files should only be deletable by their creator.
  - Group "staff" should have full rights on all contents.

- Apply similarly for "dummies" on */home/dummies*.

Spoilers

```
$ sudo chgrp -R staff /home/staff

$ sudo chmod g+s /home/staff
$ sudo chmod +t /home/staff
$ sudo chmod g+rwx /home/staff
```

Spoilers

```
$ sudo chgrp -R dummies /home/dummies

$ sudo chmod g+s /home/dummies
$ sudo chmod +t /home/dummies
$ sudo chmod g+rwx /home/dummies
```

You try…

```
$ git config --global user.name \
"John Doe"

$ git config --global user.email \
johndoe@example.com
```

You try…

```
$ mkdir ~/Documents/gitdemo
$ cd ~/Documents/gitdemo
$ git init
$ ls -al
$ ls -al .git
```

You try…

```
$ git status

$ echo "Hallo" > readme.txt
$ git status
```

You try…

```
$ git add readme.txt
$ git status

$ git commit -m "My first file"
$ git status
```

You try…

```
$ git log
```

You try…

```
$ git branch bugfix
$ git checkout bugfix

$ vi readme.txt    # make some changes
$ git add readme.txt
$ git commit -m "Fixed it!"
```

You try…

```
$ ls -al; cat readme.txt

$ git checkout master        # … back
$ ls -al; cat readme.txt

$ git checkout bugfix        # … and forth
$ ls -al; cat readme.txt
```

You try…

```
$ git checkout master
$ echo "Hallo" > newfile
$ git add newfile

$ git commit -m "New file made."
```

You try…

```
$ git checkout bugfix
$ ls newfile

$ git merge master
$ ls newfile
```

You try…

```
$ git checkout master
$ cat readme.txt


$ git merge bugfix
$ cat readme.txt
```

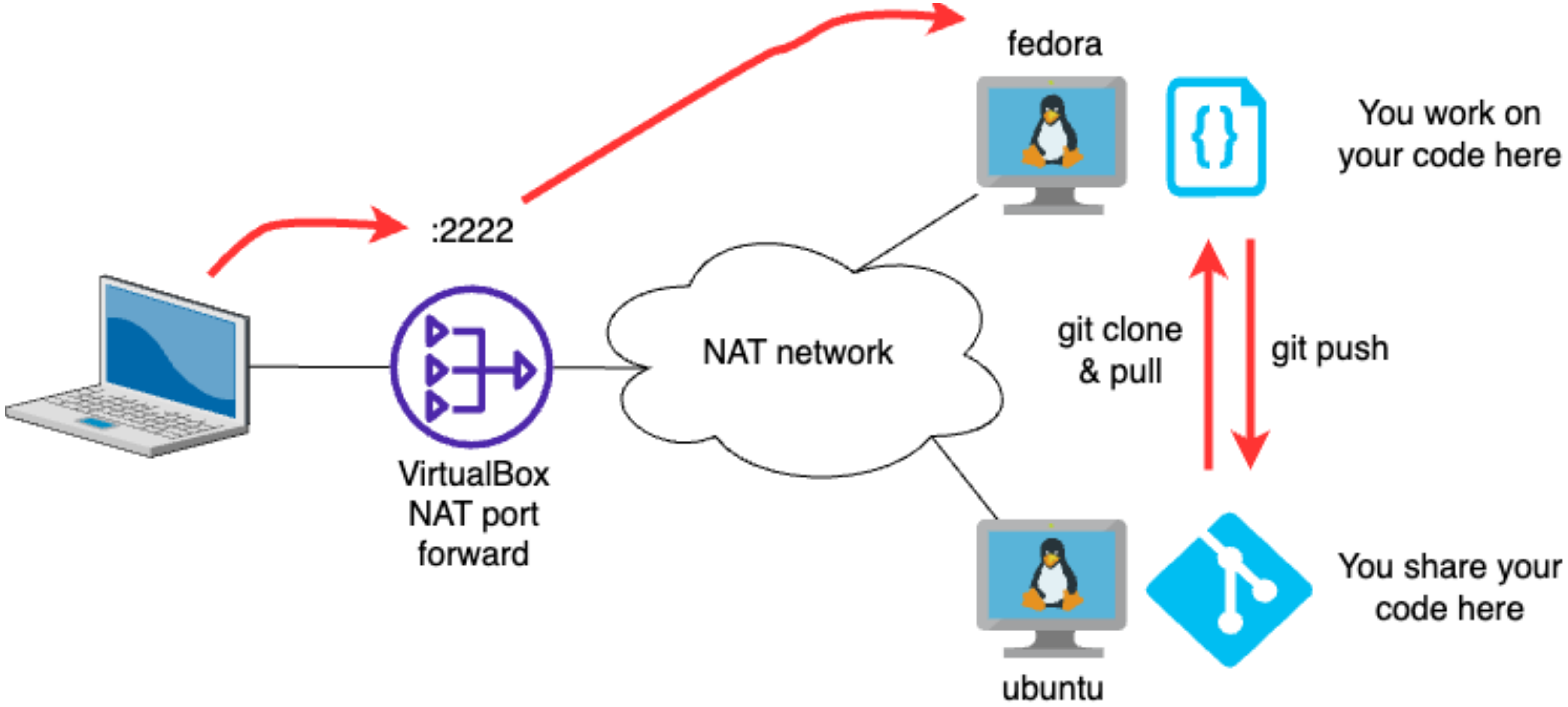
You try…

```
$ cd ~/Documents

$ git clone \
https://github.com/tsluyter/exploits

$ cd exploits; git log
```

LAB: Git

Your own, "remote" repo

- Let’s make the Ubuntu VM our **Git server**.



Setting up the server

- On the Ubuntu VM, make user account "git".
  - With homedir "*/home/git*".
  - And a password you won’t mind typing.

- Test that you can SSH from Fedora,
  - To the user "git" on the new VM.

Making a repo

- On the **Ubuntu VM**, login as user "git".
  - Configure their name and email (slide 70).

- Make the dir "*/home/git/firstrepo*".
- "*cd*" into "*firstrepo*" and init a Git repo.
  - Use: "*git init --bare*"!!
- See: [Bare vs non-bare repositories]

Cloning the repo

- On the **Fedora VM**, login as yourself.
- "*cd*" into your home directory.
- Clone the repository from the new VM:

```
$ git clone ssh://git@ubuntu:/home/git/firstrepo
```

Making a change

- On the **Fedora VM**, "*cd*" into the Git repo.
- Make a new file and commit the change.
- Then "*git push*" the update.

Comparing

- Compare the contents of:
  - The cloned git repo on your Fedora box.
  - The bare repo on the Ubuntu VM.
  - "*git log*" on the two repository locations.

- Research question: where are the files on Ubuntu?!

Co-working

- On the **Fedora VM**, login as user “dummy”.
- Make sure you're in the homedir of "dummy".
- Clone the repository from the new VM:

```
$ git clone ssh://git@ubuntu:/home/git/firstrepo
```

Co-working

- Do you see the file(s) you just pushed?
- Now make another file, as dummy.
- Commit and push it.

- Then switch back to your own account.
- And “*git pull*”. Does the changed file show up?

Spoilers - on Ubuntu

- With your own account …

```
$ sudo useradd -m -s /bin/bash git

$ sudo passwd git

$ su - git
```

Spoilers - on Ubuntu

- You are now the “git user”…

```
$ git config --global user.name "Git"
$ git config --global user.email "git@ubuntu"

$ mkdir firstrepo; cd firstrepo
$ git init --bare
$ exit
```

Spoilers - on Fedora

- With your own account.

```
$ cd ~

$ git clone ssh://git@ubuntu:/home/git/firstrepo

$ cd firstrepo
```

Spoilers - on Fedora

- With your own account.

```
$ echo "Hoi." > readme.txt
$ git add readme.txt
$ git commit -m "My first file."
$ git push

$ su - dummy
```

Spoilers - on Fedora

- Now you are user "dummy".

```
$ git config --global user.name "Dummy"
$ git config --global user.email "dummy@fedora"

$ git clone ssh://git@ubuntu:/home/git/firstrepo
$ cd firstrepo
```

Spoilers - on Fedora

- You are still user "dummy".

```
$ echo "Dummy wrote this." > dummy.txt
$ git add dummy.txt
$ git commit -m "Dummy file."
$ git push

$ exit
```

Spoilers - on Fedora

- You are now using your own account again.

```
$ cd ~/Documents/firstrepo

$ git pull

$ ls -al
$ cat dummy.txt
```

Closing

Homework

- Reading:
  - Chapter 4
  - Chapter 25

Homework

- Go do:
  - Download the free book "Pro Git".
  - Complete the labs from today's class.
  - Make a "Scripts" repository on Ubuntu.
    - Clone it to your homedir on Fedora.
    - Prove that it works.

Reference materials

Resources

- Linux file paths
- FHS on Wikipedia
- Identifying file types in Linux
- Graphical vi cheatsheet
- Vim Adventures! (game to practice hotkeys)
- Nano cheatsheet

Resources

- Git internals
- Free book: Pro Git
- Intro to Git for security professionals
- Bare vs non-bare repositories
- Stop making shell aliases for SSH!