CD and linux

Admin

- Recovering from snow closures
  - Schedule
    - Quizzes
    - Midterm Exam
    - Project1 Sprint4
  - Podcast discussion
- Other things we should discuss?

CI/CD

- So far we have done a lot of the CI in CI/CD
  - So what is the CD part?
  -

CI/CD

- So far we have done a lot of the CI in CI/CD
  - So what is the CD part?
  - Continuous Deployment.
  - Every-time we pass all the automated tests and linting, we automatically send the updated program to the server.
- So what OS do servers mostly run?

CI/CD

- So far we have done a lot of the CI in CI/CD
  - So what is the CD part?
  - Continuous Deployment.
  - Every-time we pass all the automated tests and linting, we automatically send the updated program to the server.
- So what OS do servers mostly run?
  - Linux (has 44.8% of all computing, and you don’t see much on desktop or phones)

Logging into linux

- If we want to log into a linux machine to install or run a program what are our options to do so?

Logging into linux

- If we want to log into a linux machine to install or run a program what are our options to do so?
  - Connect from a trusted MAC/IP combo (older and mostly phased out)
  - Use a username and password.
    - How useful is this for an automated system?

Logging into linux

- If we want to log into a linux machine to install or run a program what are our options to do so?
  - Connect from a trusted MAC/IP combo (older and mostly phased out)
  - Use a username and password.
    - How useful is this for an automated system?
    - Probably not very

Logging into linux

- If we want to log into a linux machine to install or run a program what are our options to do so?
  - Connect from a trusted MAC/IP combo (older and mostly phased out)
  - Use a username and password.
  - Use an ssh key
    - If both machines have the same public/private key pair, then you can get one to connect to the other and run programs on that second machine.
    - Most common approach for CI/CD at the moment.
  - Anything else you came up with?

First: create an SSH key on your desk/laptop

- On Linux/Unix (including MacOS)
- Run ssh-keygen
  - Eg:
    - `ssh-keygen -t rsa -b 4096 -C "your_email@domain.com"`
    - `When prompted for a passphrase, keep it empty`
    - `Ref: https://wiki.debian.org/Setup%20SSH%20Passwordless%20Login`

For windows,

- You can either run ssh-keygen from powershell
- Or use the “Apps and Features” control panel as per:
  - https://www.purdue.edu/science/scienceit/ssh-keys-windows.html
  - standard disclaimer that I don’t have a windows machine to practice on.

Second: setup server

- On the linux server
  - Create a non-root user (let's call the user `deploy`)
  - Login as that `deploy` user
  - Create a hidden folder called .ssh in the `deploy` user's home folder
  - Copy your public key to the server
    - Either by using scp (linux/unix/mac)
    - or maybe by using something like vim/nano and pasting the public key onto the server

Third setup github secrets

- You need to add three new secrets to your github secrets
  - Your `deploy` username
  - Your private key that matches the server public key
  - Your server IP address
- Make sure they are only in the secrets and not in your code anywhere.
- Strong suggestion: use a new public/private key pair for this – don't reuse as some tutorials suggest you might.

Fourth Setup Github Actions

- Now to deploy you need
  - A new section below your tests
  - to have the three secrets loaded into environment variables
  - To connect to the server
    - Could do an rsync
    - I’ve had good luck with the appleboy/ssh-action script

Using appleboy/ssh-action

- Add a section after tests
- - name: Deploy
- uses: appleboy/ssh-action@master
- with:
- host: ${{ secrets.DEPLOY_SERVER }}
- username: ${{ secrets.DEPLOY_USER }}
- key: ${{ secrets.DEPLOY_KEY }}
- script: |
- cd WheresMyJob
- git pull
- <more linux commands here to run program>

- The ssh-action script wants the three environment variables to be named exactly this
  - It will ssh to the server then run each command in the ‘script’ section on the server
  - This one assumes that the project exists on your server as a git repo

References

- Various references:
  - Short and sweet: https://www.linkedin.com/posts/muhoroedwin_how-i-automate-my-deployments-tired-of-activity-7362513474884689920-xSB7/
  - Some of these do extra steps, but they do explain things more
  - https://medium.com/@balazskocsis/deploying-to-a-server-with-github-actions-a-deep-dive-e8558e83a4d7
  - https://dev.to/nasrulhazim/automating-code-deployment-with-github-actions-via-ssh-2b36
  - https://www.redhat.com/en/blog/passwordless-ssh
  - http://www.linuxproblem.org/art_9.html