For these exercises, you only need to provide what is asked in the description. (There is no need to describe a vulnerability, the exploit, and prevention techniques)

**Indirect reconaissance**
- Run https://www.pdx.edu and https://oregonctf.org/ through the sites below
  - https://observatory.mozilla.org/
  - https://builtwith.com/
- Answer the questions below
  - How do the two sites differ in what they find out about each site?
  - Provide a screenshot of the https://www.pdx.edu output

**Direct reconaissance**
- Set up pentest-vm
  - Log into console.cloud.google.com
  - Create a new project labeled (cs495winter19)
  - On Menu, find “Marketplace” and “Compute Engine” and pin them to the top
  - Click on “Compute Engine” and wait for it to be enabled
  - Click on “Create”
  - Create a new instance
    - Name: pentest-vm
    - Zone: us-west1-b
    - Machine type: micro
    - Boot disk: Ubuntu 18.04
  - Click on “Create” and wait
  - ssh into new instance and perform the following
    
    sudo apt update
sudo apt install libcurl4-openssl-dev
libssl-dev python-pip wfuzz nmap -y

- Use Google Marketplace to set up several web server VMs
  - Zone: us-west1-b
  - Machine type: micro
  - Deselect “Allow HTTPS traffic”
  - Visit the landing page for each VM to ensure it has been deployed properly
  - Note the “Internal IP address” of each instance
  - VMs to bring up
  - ssh into each instance
    - Find where each server pulls its configuration from via a "ps -ef | grep apache" or "ps -ef | egrep nginx"
    - Examine the conf files, the DocumentRoot (apache2) or / (nginx) resides within and cd into it
      - apache2: /opt/bitnami/apache2/htdocs
      - nginx: /opt/bitnami/nginx/html
    - On the lampstack and nginxstack VMs, create directories named secret, files, admin (via sudo mkdir).
    - Then create index.html files in each directory
      sudo touch {secret,files,admin}/index.html
    - Web content must be readable and executable by the Linux account the web server is run from (typically www-data). Since you create these files with a different account, ensure that all files are readable.
and all directories are readable and executable by running this command in the location of the web server’s document root: `sudo chmod go+rX`.

- Use Google Compute Engine, bring up a web server on a Windows Server 2012 R2 instance
  - [https://cloud.google.com/compute/docs/quickstart-windows](https://cloud.google.com/compute/docs/quickstart-windows)
  - To connect to your instance, use an RDP client
    - `remmina` on `linuxlab` machines. Enter the external IP address of your Windows instance
    - ![Remmina Client](image)

  - Google Chrome’s RDP for Google Cloud Platform extension
    - Add via More Tools => Extensions => Get Extensions
    - Click on RDP button in console
      - Skip the clean-up step
      - Connect to your Windows Server 2012 instance and install the IIS component
        - [https://cloud.google.com/compute/docs/tutorials/basic-webserver-iis](https://cloud.google.com/compute/docs/tutorials/basic-webserver-iis)
        - Note: Windows PowerShell can be accessed in the upper right corner under “Tools”
■ Within PowerShell, change directories into the webroot folder (cd) given in the echo command in the instructions

■ Create directories named secret, files, admin using mkdir
  ● Note that you may need to do "mkdir ..." to be able to make the directory

■ Then, in PowerShell copy the index.html file you created in the echo command into each of the directories
  ● cp .\index.html admin
  ● cp .\index.html secret
  ● cp .\index.html files

■ Create directories of your own using words of your choice and copy the index.html file into them as well
● Use wfuzz and nmap to automatically scan directories on each of the 4 web servers using their INTERNAL IP address
  ○ wfuzz -c -w
    /usr/share/wfuzz/wordlist/general/common.txt
    --hc 404 http://10.x.y.z/FUZZ
  ○ nmap --script http-enum 10.x.y.z
  ○ Answer the following questions
    ■ Do the nmap and wfuzz tools get similar results for each site?
    ■ Provide screenshots of each tool’s output on the Windows web server VM
● Stop all VM instances when complete