# ECE 560 Homework 3

Guanhua Chen

TOTAL POINTS

**102 / 100**

QUESTION 1

## 1 Accessing the Homework 0 / 0

✓ **- 0 pts** **Correct**

QUESTION 2

## User Authentication 8 pts

### 2.1 a 1 / 1

✓ **- 0 pts** **Correct**

### 2.2 b 1 / 1

✓ **- 0 pts** **Correct**

### 2.3 c 1 / 1

✓ **- 0 pts** **Correct**

**- 1 pts** Incorrect. Answer: Hashing, salting, and iteration count

### 2.4 d 1 / 1

✓ **- 0 pts** **Correct**

**- 1 pts** It's complexity

### 2.5 e 1 / 1

✓ **- 0 pts** **Correct**

**- 0.5 pts** Insufficient threat model details

### 2.6 f 1 / 1

✓ **- 0 pts** **Correct**

**- 0.5 pts** No descriptions...

**- 0.25 pts** Insufficient descriptions

**- 0.25 pts** Need a couple more methods

### 2.7 g 1 / 1

✓ **- 0 pts** **Correct**

### 2.8 h 1 / 1

✓ **- 0 pts** **Correct**

QUESTION 3

## File Permissions 7 pts

### 3.1 a 1 / 1

✓ **- 0 pts** **Correct**

**- 0.5 pts** Insufficient distinction between files and directories for interpreting RWX bits

**- 0.75 pts** Doesn't mention RWX at all

**- 0.5 pts** Doesn't mention owner/group/others

### 3.2 b 2 / 2

✓ **- 0 pts** **Correct**

**- 1 pts** No explanation for directory

**- 0.5 pts** Incorrect for directory

### 3.3 c 1 / 1

✓ **- 0 pts** **Correct**

### 3.4 d 1 / 1

✓ **- 0 pts** **Correct**

**- 0.25 pts** Doesn't explain permissions

**- 0.5 pts** Doesn't explain multiple elements in the line

### 3.5 e 1 / 1

✓ **- 0 pts** **Correct**

**- 1 pts** Incorrect. Answer: chmod+w filename

### 3.6 f 1 / 1

✓ **- 0 pts** **Correct**

**- 1 pts** Incorrect (chown)

QUESTION 4

## Data Processing 16 pts

### 4.1 Filter nmap output 4 / 4

✓ - 0 pts Correct

- 2 pts No output sample
- 2 pts Extraneous output present

### 4.2 Log analysis 4 / 4

✓ - 0 pts Correct

- 1.5 pts IPs not unique!

### 4.3 Web app needle in a haystack 10 / 8

✓ - 0 pts Correct

+ 4 pts Extra credit!

- 2 pts Didn't give actual click sequences but did give all intermediate pages

- 6 pts Found incorrect links, basic strategy right

- 4 pts No click sequences

+ 1 pts Extra credit, no click sequences

+ 3 pts Extra credit, no click sequences but correct intermediate pages

- 6 pts No work shown

+ 2 Point adjustment

1. Extra credit item found

QUESTION 5

## Regular Expressions 9 pts

### 5.1 NetID validator 2 / 3

- 0 pts Correct

✓ - 1 pts **No anchors (netid could be found in the middle of a non-netid string) , ^ for start anchor, $ for end anchor**

- 1 pts Wrong expression for letters. \\w includes digits and underscore. Use \[a-zA-Z]

- 1 pts Wrong expression for digits. Use \[0-9] or \\d

- 0 pts Not considering case sensitive for NEtID in this case

https://piazza.com/class/kd6irc9vkbl23b?cid=226

### 5.2 URL parser 2.5 / 3

- 0 pts Correct

✓ - 0.5 pts **Didn't escape dots**

- 0.5 pts Extra capture group

- 1 pts No capture groups

- 0.25 pts No beginning anchor

### 5.3 Pi digits 2.5 / 3

- 0 pts Correct

✓ - 0.5 pts **Forgot to get rid of greedy repetition (you may still get the correct hash w/o it, but there was no way for you to know that in advance)**

QUESTION 6

## 6 GNU Screen 2 / 2

✓ - 0 pts Correct

- 2 pts Wrong Screenshot

QUESTION 7

## 7 Cracking MD5 hashes with hashcat 8 / 6

✓ - 0 pts Correct

- 2 pts No Screenshot
- 2 pts Wrong password
- 2 pts No command

✓ + 2 pts Extra Credit

+ 4 pts Extra Credit

+ 6 pts Extra Credit

QUESTION 8

## 8 John the Ripper 10 / 10

✓ - 0 pts Correct

- 1 pts Description not detailed
- 3 pts Major defects on description
- 4 pts No description
- 1 pts Missing one password
- 6 pts No password cracked

QUESTION 9

## Evaluating MFA approaches 8 pts

### 9.1 a 2 / 2

✓ - 0 pts Correct

- 1 pts Did not mention SMS
- 2 pts No answer

9.2 b **2 / 2**

✓ **- 0 pts Correct**

**- 2 pts** No answer

9.3 c **2 / 2**

✓ **- 0 pts Correct**

**- 2 pts** No answer

9.4 d **2 / 2**

✓ **- 0 pts Correct**

**- 2 pts** No answer

**- 1.5 pts** The question asks for an attack event, not a method

QUESTION 10

## 10 SSH with MFA **5 / 5**

✓ **- 0 pts Correct**

QUESTION 11

## Hydra (Online SSH Dictionary Attack) 5 pts

### 11.1 Build user list **1 / 1**

✓ **- 0 pts Correct**

### 11.2 Attack **4 / 4**

✓ **+ 4 pts Correct**

**+ 5 pts** Extra-Credit

**+ 0 pts** Wrong answer

I don't feel ssh tunnel forwarding works for our purpose. We want an interactive shell before the ssh tunnel close itself.

QUESTION 12

## 12 Craft your prompt **3 / 3**

✓ **- 0 pts Correct**

**- 3 pts** Click here to replace this description.

QUESTION 13

## 13 Attack Recon **6 / 6**

✓ **- 0 pts Correct**

**- 1 pts** Correct solution but could have been shorter

QUESTION 14

## Keeping Up With the News 10 pts

### 14.1 Article summary with link **3 / 3**

✓ **- 0 pts Correct**

**- 3 pts** Click here to replace this description.

### 14.2 Aspects of CIA triad **2 / 2**

✓ **- 0 pts Correct**

**- 2 pts** Click here to replace this description.

### 14.3 Thread models **2 / 2**

✓ **- 0 pts Correct**

**- 2 pts** Click here to replace this description.

### 14.4 Describe attack/defense **3 / 3**

✓ **- 0 pts Correct**

**- 3 pts** Click here to replace this description.

QUESTION 15

## 15 Manipulating Binary File Formats **5 / 5**

✓ **- 0 pts Correct**

**- 5 pts** Click here to replace this description.

QUESTION 16

## 16 Late Penalty **0 / 0**

✓ **- 0 pts Correct**

## 1 Accessing the Homework 0 / 0

✓ **- 0 pts** **Correct**

## Question 0: Accessing the Homework (0 points, but necessary)

The Homework 3 pointer was a JPG/PDF polyglot, but if you're reading this, you've already figured this out. You can read more in PoC||GTFO 03:03 ("This PDF is a JPEG; or, This Proof of Concept is a Picture of Cats"). The final question of this assignment will require you to apply a similar technique to transform your submitted PDF.

## Question 1: Chapter 3 - User Authentication (8 points)

(Based in part on review questions from the textbook)

a. In general terms, what are the three (or four) categories of ways to authenticate a user's identity?

   Something you know(password), something you has(card), static biometrics(face), dynamic biometrics(voice)

b. List and briefly describe five possible threats to the secrecy of passwords.

   Offline dictionary attack(cracking hashed password using rainbow table)

   Specific dictionary attack(target specific accounts)

   Popular password attack(using popular password)

   Exploit user mistakes (use the password user post to login)

   Electronic monitoring(sniffing network)

c. What are three common techniques used to protect stored passwords?

   Hash, salt, iteration count

d. Which is more important: password complexity requirements or password expiration requirements? Why?

   Complexity.

   With the complex password and proper storage strategy, the password can be safe for long time. If the password is simple like 123, then no matter how often you update, it will always be cracked easily.

e. What is MFA? What threat model does it deal with?

   Multifactor authentication, authenticate user with composite tools.

   Fake login. For example if you only authenticate with password, once password is leaked then anyone can fake the user. However, with MFA, you need other info to login(SMS etc), adding extra protection.

f. List and briefly describe the principal characteristics used for biometric identification.

2.1 a **1 / 1**

✓ **- 0 pts** Correct

## Question 0: Accessing the Homework (0 points, but necessary)

The Homework 3 pointer was a JPG/PDF polyglot, but if you're reading this, you've already figured this out. You can read more in PoC||GTFO 03:03 ("This PDF is a JPEG; or, This Proof of Concept is a Picture of Cats"). The final question of this assignment will require you to apply a similar technique to transform your submitted PDF.

## Question 1: Chapter 3 - User Authentication (8 points)

(Based in part on review questions from the textbook)

a. In general terms, what are the three (or four) categories of ways to authenticate a user's identity?

   Something you know(password), something you has(card), static biometrics(face), dynamic biometrics(voice)

b. List and briefly describe five possible threats to the secrecy of passwords.

   Offline dictionary attack(cracking hashed password using rainbow table)

   Specific dictionary attack(target specific accounts)

   Popular password attack(using popular password)

   Exploit user mistakes (use the password user post to login)

   Electronic monitoring(sniffing network)

c. What are three common techniques used to protect stored passwords?

   Hash, salt, iteration count

d. Which is more important: password complexity requirements or password expiration requirements? Why?

   Complexity.

   With the complex password and proper storage strategy, the password can be safe for long time. If the password is simple like 123, then no matter how often you update, it will always be cracked easily.

e. What is MFA? What threat model does it deal with?

   Multifactor authentication, authenticate user with composite tools.

   Fake login. For example if you only authenticate with password, once password is leaked then anyone can fake the user. However, with MFA, you need other info to login(SMS etc), adding extra protection.

f. List and briefly describe the principal characteristics used for biometric identification.

2.2 b **1 / 1**

✓ **- 0 pts** Correct

## Question 0: Accessing the Homework (0 points, but necessary)

The Homework 3 pointer was a JPG/PDF polyglot, but if you're reading this, you've already figured this out. You can read more in PoC||GTFO 03:03 ("This PDF is a JPEG; or, This Proof of Concept is a Picture of Cats"). The final question of this assignment will require you to apply a similar technique to transform your submitted PDF.

## Question 1: Chapter 3 - User Authentication (8 points)

(Based in part on review questions from the textbook)

a. In general terms, what are the three (or four) categories of ways to authenticate a user's identity?

   Something you know(password), something you has(card), static biometrics(face), dynamic biometrics(voice)

b. List and briefly describe five possible threats to the secrecy of passwords.

   Offline dictionary attack(cracking hashed password using rainbow table)

   Specific dictionary attack(target specific accounts)

   Popular password attack(using popular password)

   Exploit user mistakes (use the password user post to login)

   Electronic monitoring(sniffing network)

c. What are three common techniques used to protect stored passwords?

   Hash, salt, iteration count

d. Which is more important: password complexity requirements or password expiration requirements? Why?

   Complexity.

   With the complex password and proper storage strategy, the password can be safe for long time. If the password is simple like 123, then no matter how often you update, it will always be cracked easily.

e. What is MFA? What threat model does it deal with?

   Multifactor authentication, authenticate user with composite tools.

   Fake login. For example if you only authenticate with password, once password is leaked then anyone can fake the user. However, with MFA, you need other info to login(SMS etc), adding extra protection.

f. List and briefly describe the principal characteristics used for biometric identification.

2.3 C **1 / 1**

✓ **- 0 pts** **Correct**

**- 1 pts** Incorrect. Answer: Hashing, salting, and iteration count

## Question 0: Accessing the Homework (0 points, but necessary)

The Homework 3 pointer was a JPG/PDF polyglot, but if you're reading this, you've already figured this out. You can read more in PoC||GTFO 03:03 ("This PDF is a JPEG; or, This Proof of Concept is a Picture of Cats"). The final question of this assignment will require you to apply a similar technique to transform your submitted PDF.

## Question 1: Chapter 3 - User Authentication (8 points)

(Based in part on review questions from the textbook)

a. In general terms, what are the three (or four) categories of ways to authenticate a user's identity?

   Something you know(password), something you has(card), static biometrics(face), dynamic biometrics(voice)

b. List and briefly describe five possible threats to the secrecy of passwords.

   Offline dictionary attack(cracking hashed password using rainbow table)

   Specific dictionary attack(target specific accounts)

   Popular password attack(using popular password)

   Exploit user mistakes (use the password user post to login)

   Electronic monitoring(sniffing network)

c. What are three common techniques used to protect stored passwords?

   Hash, salt, iteration count

d. Which is more important: password complexity requirements or password expiration requirements? Why?

   Complexity.

   With the complex password and proper storage strategy, the password can be safe for long time. If the password is simple like 123, then no matter how often you update, it will always be cracked easily.

e. What is MFA? What threat model does it deal with?

   Multifactor authentication, authenticate user with composite tools.

   Fake login. For example if you only authenticate with password, once password is leaked then anyone can fake the user. However, with MFA, you need other info to login(SMS etc), adding extra protection.

f. List and briefly describe the principal characteristics used for biometric identification.

2.4 d **1 / 1**

✓ **- 0 pts** **Correct**

**- 1 pts** It's complexity

## Question 0: Accessing the Homework (0 points, but necessary)

The Homework 3 pointer was a JPG/PDF polyglot, but if you're reading this, you've already figured this out. You can read more in PoC||GTFO 03:03 ("This PDF is a JPEG; or, This Proof of Concept is a Picture of Cats"). The final question of this assignment will require you to apply a similar technique to transform your submitted PDF.

## Question 1: Chapter 3 - User Authentication (8 points)

(Based in part on review questions from the textbook)

a. In general terms, what are the three (or four) categories of ways to authenticate a user's identity?

   Something you know(password), something you has(card), static biometrics(face), dynamic biometrics(voice)

b. List and briefly describe five possible threats to the secrecy of passwords.

   Offline dictionary attack(cracking hashed password using rainbow table)

   Specific dictionary attack(target specific accounts)

   Popular password attack(using popular password)

   Exploit user mistakes (use the password user post to login)

   Electronic monitoring(sniffing network)

c. What are three common techniques used to protect stored passwords?

   Hash, salt, iteration count

d. Which is more important: password complexity requirements or password expiration requirements? Why?

   Complexity.

   With the complex password and proper storage strategy, the password can be safe for long time. If the password is simple like 123, then no matter how often you update, it will always be cracked easily.

e. What is MFA? What threat model does it deal with?

   Multifactor authentication, authenticate user with composite tools.

   Fake login. For example if you only authenticate with password, once password is leaked then anyone can fake the user. However, with MFA, you need other info to login(SMS etc), adding extra protection.

f. List and briefly describe the principal characteristics used for biometric identification.

2.5 e **1 / 1**

✓ **- 0 pts** **Correct**

**- 0.5 pts** Insufficient threat model details

## Question 0: Accessing the Homework (0 points, but necessary)

The Homework 3 pointer was a JPG/PDF polyglot, but if you're reading this, you've already figured this out. You can read more in PoC||GTFO 03:03 ("This PDF is a JPEG; or, This Proof of Concept is a Picture of Cats"). The final question of this assignment will require you to apply a similar technique to transform your submitted PDF.

## Question 1: Chapter 3 - User Authentication (8 points)

(Based in part on review questions from the textbook)

a. In general terms, what are the three (or four) categories of ways to authenticate a user's identity?

   Something you know(password), something you has(card), static biometrics(face), dynamic biometrics(voice)

b. List and briefly describe five possible threats to the secrecy of passwords.

   Offline dictionary attack(cracking hashed password using rainbow table)

   Specific dictionary attack(target specific accounts)

   Popular password attack(using popular password)

   Exploit user mistakes (use the password user post to login)

   Electronic monitoring(sniffing network)

c. What are three common techniques used to protect stored passwords?

   Hash, salt, iteration count

d. Which is more important: password complexity requirements or password expiration requirements? Why?

   Complexity.

   With the complex password and proper storage strategy, the password can be safe for long time. If the password is simple like 123, then no matter how often you update, it will always be cracked easily.

e. What is MFA? What threat model does it deal with?

   Multifactor authentication, authenticate user with composite tools.

   Fake login. For example if you only authenticate with password, once password is leaked then anyone can fake the user. However, with MFA, you need other info to login(SMS etc), adding extra protection.

f. List and briefly describe the principal characteristics used for biometric identification.

Face: scan face and use certain point to detect whether you are the user or not

Fingerprint: fingerprint scanner

Signature: search pattern in your signature

Voice: find pattern in voice

Iris, retina: scan your eyes

g. Describe the general concept of a challenge-response protocol.

Checking someone has secret without sending the secret

Client require login as user Bob

Server issue a challenge only solved with bob secret

Client response answer

Server check match or not

h. In a challenge-response password protocol, why is a random nonce used?

To avoid the leak of hash.

If hash is leak, we can login with the hash. But with nonce, you can get nothing from the leak

## Question 2: File permissions (7 points)

From Problem 4.5: UNIX treats file directories in the same fashion as files; that is, both are defined by the same type of data structure, called an inode. As with files, directories include a nine-bit protection string. If care is not taken, this can create access control problems. For example, consider a file with the protection mode 644 (octal) contained in a directory with protection mode 773.

**Explain the interpretation of the 9-bit protection string, for both files and directories. (1)**

**644: 110 100 100 rw-r- - r- - user can read, write. group & others only read**
**773: 111 111 011 rwxrwx-wx user can read write execute. group can read write execute. other can write execute**

**Explain the meaning of the octal protection string of 644 for a file, and 750 for a directory. (2)**
**644: 110 100 100 rw-r- - r- - user can read, write. group & others only read**
**750: 111 101 000 rwxr-x- - - user can read write execute. group can read execute. other do nothing**

2.6 f **1 / 1**

✓ **- 0 pts** **Correct**

**- 0.5 pts** No descriptions...

**- 0.25 pts** Insufficient descriptions

**- 0.25 pts** Need a couple more methods

Face: scan face and use certain point to detect whether you are the user or not

Fingerprint: fingerprint scanner

Signature: search pattern in your signature

Voice: find pattern in voice

Iris, retina: scan your eyes

g. Describe the general concept of a challenge-response protocol.

Checking someone has secret without sending the secret

Client require login as user Bob

Server issue a challenge only solved with bob secret

Client response answer

Server check match or not

h. In a challenge-response password protocol, why is a random nonce used?

To avoid the leak of hash.

If hash is leak, we can login with the hash. But with nonce, you can get nothing from the leak

## Question 2: File permissions (7 points)

From Problem 4.5: UNIX treats file directories in the same fashion as files; that is, both are defined by the same type of data structure, called an inode. As with files, directories include a nine-bit protection string. If care is not taken, this can create access control problems. For example, consider a file with the protection mode 644 (octal) contained in a directory with protection mode 773.

**Explain the interpretation of the 9-bit protection string, for both files and directories. (1)**

**644: 110 100 100 rw-r- - r- - user can read, write. group & others only read**
**773: 111 111 011 rwxrwx-wx user can read write execute. group can read write execute. other can write execute**

**Explain the meaning of the octal protection string of 644 for a file, and 750 for a directory. (2)**
**644: 110 100 100 rw-r- - r- - user can read, write. group & others only read**
**750: 111 101 000 rwxr-x- - - user can read write execute. group can read execute. other do nothing**

2.7 g **1 / 1**

✓ **- 0 pts** Correct

Face: scan face and use certain point to detect whether you are the user or not

Fingerprint: fingerprint scanner

Signature: search pattern in your signature

Voice: find pattern in voice

Iris, retina: scan your eyes

g. Describe the general concept of a challenge-response protocol.

Checking someone has secret without sending the secret

Client require login as user Bob

Server issue a challenge only solved with bob secret

Client response answer

Server check match or not

h. In a challenge-response password protocol, why is a random nonce used?

To avoid the leak of hash.

If hash is leak, we can login with the hash. But with nonce, you can get nothing from the leak

## Question 2: File permissions (7 points)

From Problem 4.5: UNIX treats file directories in the same fashion as files; that is, both are defined by the same type of data structure, called an inode. As with files, directories include a nine-bit protection string. If care is not taken, this can create access control problems. For example, consider a file with the protection mode 644 (octal) contained in a directory with protection mode 773.

**Explain the interpretation of the 9-bit protection string, for both files and directories. (1)**

**644: 110 100 100 rw-r- - r- - user can read, write. group & others only read**
**773: 111 111 011 rwxrwx-wx user can read write execute. group can read write execute. other can write execute**

**Explain the meaning of the octal protection string of 644 for a file, and 750 for a directory. (2)**
**644: 110 100 100 rw-r- - r- - user can read, write. group & others only read**
**750: 111 101 000 rwxr-x- - - user can read write execute. group can read execute. other do nothing**

2.8 h **1 / 1**

✓ **- 0 pts** Correct

Face: scan face and use certain point to detect whether you are the user or not

Fingerprint: fingerprint scanner

Signature: search pattern in your signature

Voice: find pattern in voice

Iris, retina: scan your eyes

g. Describe the general concept of a challenge-response protocol.

Checking someone has secret without sending the secret

Client require login as user Bob

Server issue a challenge only solved with bob secret

Client response answer

Server check match or not

h. In a challenge-response password protocol, why is a random nonce used?

To avoid the leak of hash.

If hash is leak, we can login with the hash. But with nonce, you can get nothing from the leak

## Question 2: File permissions (7 points)

From Problem 4.5: UNIX treats file directories in the same fashion as files; that is, both are defined by the same type of data structure, called an inode. As with files, directories include a nine-bit protection string. If care is not taken, this can create access control problems. For example, consider a file with the protection mode 644 (octal) contained in a directory with protection mode 773.

**Explain the interpretation of the 9-bit protection string, for both files and directories. (1)**

**644: 110 100 100 rw-r- - r- - user can read, write. group & others only read**
**773: 111 111 011 rwxrwx-wx user can read write execute. group can read write execute. other can write execute**

**Explain the meaning of the octal protection string of 644 for a file, and 750 for a directory. (2)**
**644: 110 100 100 rw-r- - r- - user can read, write. group & others only read**
**750: 111 101 000 rwxr-x- - - user can read write execute. group can read execute. other do nothing**

**Do an experiment: Make a directory with permissions 773. Can non-owner non-group users list the contents of the directory? Can such users create new files in the directory? (1)**
Can list contents
Can't create new files

Consider the `ls` output below.

```
-rw-r--r--  1 root     root      1.6M Sep 25 16:58 gosh.tar.gz
```

**Explain in detail what each element this line means. (1)**

| - | rw-r—r-- | 1 | Root root | 1.6M | Sep 25 16:58 | gosh.tar.gz |
|---|---|---|---|---|---|---|
| **File type: regular file** | **User can read write, group can read, other can read** | **1 hard link to file** | **Owner is root, group is root** | **Size is 1.6M** | **Last modification timestamp** | **File name** |

**What command would you use to change the access rights to allow any user to edit the file? (1)**
**sudo chmod a+w** `gosh.tar.gz`
**What command would you use to take full ownership from root to yourself? (1)**
**Sudo chown my-uid** `gosh.tar.gz`

## Question 3: Data processing (16 points)

In this question, you'll be manipulating some text data. Beyond the tools covered in class, you may want to look into:

- AWK, a general purpose computer language that is designed for processing text-based data, either in files or data streams. The name AWK is derived from the surnames of its authors Alfred V. Aho, Peter J. Weinberger, and Brian W. Kernighan.
- Perl is also a general purpose computer language focused on text-based data developed by Larry Wall.

### Task 1: Filter nmap output (4 points)

**Using any combination of the tools described in class or listed above, on your Linux VM, write a *single command* with pipes that will parse out only the registered hostnames from an Nmap List scan of the 152.3.64.* subnet. Your output should not include those hosts without a hostname or any other extraneous output. Give the full command and a small**

3.1 a **1 / 1**

✓ **- 0 pts** **Correct**

**- 0.5 pts** Insufficient distinction between files and directories for interpreting RWX bits

**- 0.75 pts** Doesn't mention RWX at all

**- 0.5 pts** Doesn't mention owner/group/others

Face: scan face and use certain point to detect whether you are the user or not

Fingerprint: fingerprint scanner

Signature: search pattern in your signature

Voice: find pattern in voice

Iris, retina: scan your eyes

g. Describe the general concept of a challenge-response protocol.

Checking someone has secret without sending the secret

Client require login as user Bob

Server issue a challenge only solved with bob secret

Client response answer

Server check match or not

h. In a challenge-response password protocol, why is a random nonce used?

To avoid the leak of hash.

If hash is leak, we can login with the hash. But with nonce, you can get nothing from the leak

## Question 2: File permissions (7 points)

From Problem 4.5: UNIX treats file directories in the same fashion as files; that is, both are defined by the same type of data structure, called an inode. As with files, directories include a nine-bit protection string. If care is not taken, this can create access control problems. For example, consider a file with the protection mode 644 (octal) contained in a directory with protection mode 773.

**Explain the interpretation of the 9-bit protection string, for both files and directories. (1)**

**644: 110 100 100 rw-r- - r- - user can read, write. group & others only read**
**773: 111 111 011 rwxrwx-wx user can read write execute. group can read write execute. other can write execute**

**Explain the meaning of the octal protection string of 644 for a file, and 750 for a directory. (2)**
**644: 110 100 100 rw-r- - r- - user can read, write. group & others only read**
**750: 111 101 000 rwxr-x- - - user can read write execute. group can read execute. other do nothing**

**Do an experiment: Make a directory with permissions 773. Can non-owner non-group users list the contents of the directory? Can such users create new files in the directory? (1)**
Can list contents
Can't create new files

Consider the `ls` output below.

```
-rw-r--r--  1 root     root      1.6M Sep 25 16:58 gosh.tar.gz
```

**Explain in detail what each element this line means. (1)**

| - | rw-r—r-- | 1 | Root root | 1.6M | Sep 25 16:58 | gosh.tar.gz |
|---|---|---|---|---|---|---|
| **File type: regular file** | **User can read write, group can read, other can read** | **1 hard link to file** | **Owner is root, group is root** | **Size is 1.6M** | **Last modification timestamp** | **File name** |

**What command would you use to change the access rights to allow any user to edit the file? (1)**
**sudo chmod a+w** `gosh.tar.gz`
**What command would you use to take full ownership from root to yourself? (1)**
**Sudo chown my-uid** `gosh.tar.gz`

## Question 3: Data processing (16 points)

In this question, you'll be manipulating some text data. Beyond the tools covered in class, you may want to look into:

- AWK, a general purpose computer language that is designed for processing text-based data, either in files or data streams. The name AWK is derived from the surnames of its authors Alfred V. Aho, Peter J. Weinberger, and Brian W. Kernighan.
- Perl is also a general purpose computer language focused on text-based data developed by Larry Wall.

### Task 1: Filter nmap output (4 points)

**Using any combination of the tools described in class or listed above, on your Linux VM, write a *single command* with pipes that will parse out only the registered hostnames from an Nmap List scan of the 152.3.64.* subnet. Your output should not include those hosts without a hostname or any other extraneous output. Give the full command and a small**

3.2 b **2 / 2**

✓ **- 0 pts** **Correct**

**- 1 pts** No explanation for directory

**- 0.5 pts** Incorrect for directory

Face: scan face and use certain point to detect whether you are the user or not

Fingerprint: fingerprint scanner

Signature: search pattern in your signature

Voice: find pattern in voice

Iris, retina: scan your eyes

g. Describe the general concept of a challenge-response protocol.

Checking someone has secret without sending the secret

Client require login as user Bob

Server issue a challenge only solved with bob secret

Client response answer

Server check match or not

h. In a challenge-response password protocol, why is a random nonce used?

To avoid the leak of hash.

If hash is leak, we can login with the hash. But with nonce, you can get nothing from the leak

## Question 2: File permissions (7 points)

From Problem 4.5: UNIX treats file directories in the same fashion as files; that is, both are defined by the same type of data structure, called an inode. As with files, directories include a nine-bit protection string. If care is not taken, this can create access control problems. For example, consider a file with the protection mode 644 (octal) contained in a directory with protection mode 773.

**Explain the interpretation of the 9-bit protection string, for both files and directories. (1)**

**644: 110 100 100 rw-r- - r- - user can read, write. group & others only read**
**773: 111 111 011 rwxrwx-wx user can read write execute. group can read write execute. other can write execute**

**Explain the meaning of the octal protection string of 644 for a file, and 750 for a directory. (2)**
**644: 110 100 100 rw-r- - r- - user can read, write. group & others only read**
**750: 111 101 000 rwxr-x- - - user can read write execute. group can read execute. other do nothing**

**Do an experiment: Make a directory with permissions 773. Can non-owner non-group users list the contents of the directory? Can such users create new files in the directory? (1)**
Can list contents
Can't create new files

Consider the `ls` output below.

```
-rw-r--r--  1 root     root      1.6M Sep 25 16:58 gosh.tar.gz
```

**Explain in detail what each element this line means. (1)**

| - | rw-r—r-- | 1 | Root root | 1.6M | Sep 25 16:58 | gosh.tar.gz |
|---|---|---|---|---|---|---|
| **File type: regular file** | **User can read write, group can read, other can read** | **1 hard link to file** | **Owner is root, group is root** | **Size is 1.6M** | **Last modification timestamp** | **File name** |

**What command would you use to change the access rights to allow any user to edit the file? (1)**
**sudo chmod a+w** `gosh.tar.gz`
**What command would you use to take full ownership from root to yourself? (1)**
**Sudo chown my-uid** `gosh.tar.gz`

## Question 3: Data processing (16 points)

In this question, you'll be manipulating some text data. Beyond the tools covered in class, you may want to look into:

- AWK, a general purpose computer language that is designed for processing text-based data, either in files or data streams. The name AWK is derived from the surnames of its authors Alfred V. Aho, Peter J. Weinberger, and Brian W. Kernighan.
- Perl is also a general purpose computer language focused on text-based data developed by Larry Wall.

### Task 1: Filter nmap output (4 points)

**Using any combination of the tools described in class or listed above, on your Linux VM, write a *single command* with pipes that will parse out only the registered hostnames from an Nmap List scan of the 152.3.64.* subnet. Your output should not include those hosts without a hostname or any other extraneous output. Give the full command and a small**

3.3 C **1 / 1**

✓ **- 0 pts** Correct

Face: scan face and use certain point to detect whether you are the user or not

Fingerprint: fingerprint scanner

Signature: search pattern in your signature

Voice: find pattern in voice

Iris, retina: scan your eyes

g. Describe the general concept of a challenge-response protocol.

Checking someone has secret without sending the secret

Client require login as user Bob

Server issue a challenge only solved with bob secret

Client response answer

Server check match or not

h. In a challenge-response password protocol, why is a random nonce used?

To avoid the leak of hash.

If hash is leak, we can login with the hash. But with nonce, you can get nothing from the leak

## Question 2: File permissions (7 points)

From Problem 4.5: UNIX treats file directories in the same fashion as files; that is, both are defined by the same type of data structure, called an inode. As with files, directories include a nine-bit protection string. If care is not taken, this can create access control problems. For example, consider a file with the protection mode 644 (octal) contained in a directory with protection mode 773.

**Explain the interpretation of the 9-bit protection string, for both files and directories. (1)**

**644: 110 100 100 rw-r- - r- - user can read, write. group & others only read**
**773: 111 111 011 rwxrwx-wx user can read write execute. group can read write execute. other can write execute**

**Explain the meaning of the octal protection string of 644 for a file, and 750 for a directory. (2)**
**644: 110 100 100 rw-r- - r- - user can read, write. group & others only read**
**750: 111 101 000 rwxr-x- - - user can read write execute. group can read execute. other do nothing**

**Do an experiment: Make a directory with permissions 773. Can non-owner non-group users list the contents of the directory? Can such users create new files in the directory? (1)**
Can list contents
Can't create new files

Consider the `ls` output below.

```
-rw-r--r--  1 root     root      1.6M Sep 25 16:58 gosh.tar.gz
```

**Explain in detail what each element this line means. (1)**

| - | rw-r—r-- | 1 | Root root | 1.6M | Sep 25 16:58 | gosh.tar.gz |
|---|---|---|---|---|---|---|
| **File type: regular file** | **User can read write, group can read, other can read** | **1 hard link to file** | **Owner is root, group is root** | **Size is 1.6M** | **Last modification timestamp** | **File name** |

**What command would you use to change the access rights to allow any user to edit the file? (1)**
**sudo chmod a+w** `gosh.tar.gz`
**What command would you use to take full ownership from root to yourself? (1)**
**Sudo chown my-uid** `gosh.tar.gz`

## Question 3: Data processing (16 points)

In this question, you'll be manipulating some text data. Beyond the tools covered in class, you may want to look into:

- AWK, a general purpose computer language that is designed for processing text-based data, either in files or data streams. The name AWK is derived from the surnames of its authors Alfred V. Aho, Peter J. Weinberger, and Brian W. Kernighan.
- Perl is also a general purpose computer language focused on text-based data developed by Larry Wall.

### Task 1: Filter nmap output (4 points)

**Using any combination of the tools described in class or listed above, on your Linux VM, write a *single command* with pipes that will parse out only the registered hostnames from an Nmap List scan of the 152.3.64.* subnet. Your output should not include those hosts without a hostname or any other extraneous output. Give the full command and a small**

### 3.4 d 1 / 1

✓ **- 0 pts** **Correct**

**- 0.25 pts** Doesn't explain permissions

**- 0.5 pts** Doesn't explain multiple elements in the line

Face: scan face and use certain point to detect whether you are the user or not

Fingerprint: fingerprint scanner

Signature: search pattern in your signature

Voice: find pattern in voice

Iris, retina: scan your eyes

g. Describe the general concept of a challenge-response protocol.

Checking someone has secret without sending the secret

Client require login as user Bob

Server issue a challenge only solved with bob secret

Client response answer

Server check match or not

h. In a challenge-response password protocol, why is a random nonce used?

To avoid the leak of hash.

If hash is leak, we can login with the hash. But with nonce, you can get nothing from the leak

## Question 2: File permissions (7 points)

From Problem 4.5: UNIX treats file directories in the same fashion as files; that is, both are defined by the same type of data structure, called an inode. As with files, directories include a nine-bit protection string. If care is not taken, this can create access control problems. For example, consider a file with the protection mode 644 (octal) contained in a directory with protection mode 773.

**Explain the interpretation of the 9-bit protection string, for both files and directories. (1)**

**644: 110 100 100 rw-r- - r- - user can read, write. group & others only read**
**773: 111 111 011 rwxrwx-wx user can read write execute. group can read write execute. other can write execute**

**Explain the meaning of the octal protection string of 644 for a file, and 750 for a directory. (2)**
**644: 110 100 100 rw-r- - r- - user can read, write. group & others only read**
**750: 111 101 000 rwxr-x- - - user can read write execute. group can read execute. other do nothing**

**Do an experiment: Make a directory with permissions 773. Can non-owner non-group users list the contents of the directory? Can such users create new files in the directory? (1)**
Can list contents
Can't create new files

Consider the `ls` output below.

```
-rw-r--r--  1 root     root      1.6M Sep 25 16:58 gosh.tar.gz
```

**Explain in detail what each element this line means. (1)**

| - | rw-r—r-- | 1 | Root root | 1.6M | Sep 25 16:58 | gosh.tar.gz |
|---|---|---|---|---|---|---|
| **File type: regular file** | **User can read write, group can read, other can read** | **1 hard link to file** | **Owner is root, group is root** | **Size is 1.6M** | **Last modification timestamp** | **File name** |

**What command would you use to change the access rights to allow any user to edit the file? (1)**
**sudo chmod a+w** `gosh.tar.gz`
**What command would you use to take full ownership from root to yourself? (1)**
**Sudo chown my-uid** `gosh.tar.gz`

## Question 3: Data processing (16 points)

In this question, you'll be manipulating some text data. Beyond the tools covered in class, you may want to look into:

- AWK, a general purpose computer language that is designed for processing text-based data, either in files or data streams. The name AWK is derived from the surnames of its authors Alfred V. Aho, Peter J. Weinberger, and Brian W. Kernighan.
- Perl is also a general purpose computer language focused on text-based data developed by Larry Wall.

### Task 1: Filter nmap output (4 points)

**Using any combination of the tools described in class or listed above, on your Linux VM, write a *single command* with pipes that will parse out only the registered hostnames from an Nmap List scan of the 152.3.64.* subnet. Your output should not include those hosts without a hostname or any other extraneous output. Give the full command and a small**

3.5 e **1 / 1**

✓ **- 0 pts** **Correct**

**- 1 pts** Incorrect. Answer: chmod+w filename

Face: scan face and use certain point to detect whether you are the user or not

Fingerprint: fingerprint scanner

Signature: search pattern in your signature

Voice: find pattern in voice

Iris, retina: scan your eyes

g. Describe the general concept of a challenge-response protocol.

Checking someone has secret without sending the secret

Client require login as user Bob

Server issue a challenge only solved with bob secret

Client response answer

Server check match or not

h. In a challenge-response password protocol, why is a random nonce used?

To avoid the leak of hash.

If hash is leak, we can login with the hash. But with nonce, you can get nothing from the leak

## Question 2: File permissions (7 points)

From Problem 4.5: UNIX treats file directories in the same fashion as files; that is, both are defined by the same type of data structure, called an inode. As with files, directories include a nine-bit protection string. If care is not taken, this can create access control problems. For example, consider a file with the protection mode 644 (octal) contained in a directory with protection mode 773.

**Explain the interpretation of the 9-bit protection string, for both files and directories. (1)**

**644: 110 100 100 rw-r- - r- - user can read, write. group & others only read**
**773: 111 111 011 rwxrwx-wx user can read write execute. group can read write execute. other can write execute**

**Explain the meaning of the octal protection string of 644 for a file, and 750 for a directory. (2)**
**644: 110 100 100 rw-r- - r- - user can read, write. group & others only read**
**750: 111 101 000 rwxr-x- - - user can read write execute. group can read execute. other do nothing**

**Do an experiment: Make a directory with permissions 773. Can non-owner non-group users list the contents of the directory? Can such users create new files in the directory? (1)**
Can list contents
Can't create new files

Consider the `ls` output below.

```
-rw-r--r--  1 root     root      1.6M Sep 25 16:58 gosh.tar.gz
```

**Explain in detail what each element this line means. (1)**

| - | rw-r—r-- | 1 | Root root | 1.6M | Sep 25 16:58 | gosh.tar.gz |
|---|---|---|---|---|---|---|
| **File type: regular file** | **User can read write, group can read, other can read** | **1 hard link to file** | **Owner is root, group is root** | **Size is 1.6M** | **Last modification timestamp** | **File name** |

**What command would you use to change the access rights to allow any user to edit the file? (1)**
**sudo chmod a+w** `gosh.tar.gz`
**What command would you use to take full ownership from root to yourself? (1)**
**Sudo chown my-uid** `gosh.tar.gz`

## Question 3: Data processing (16 points)

In this question, you'll be manipulating some text data. Beyond the tools covered in class, you may want to look into:

- AWK, a general purpose computer language that is designed for processing text-based data, either in files or data streams. The name AWK is derived from the surnames of its authors Alfred V. Aho, Peter J. Weinberger, and Brian W. Kernighan.
- Perl is also a general purpose computer language focused on text-based data developed by Larry Wall.

### Task 1: Filter nmap output (4 points)

**Using any combination of the tools described in class or listed above, on your Linux VM, write a *single command* with pipes that will parse out only the registered hostnames from an Nmap List scan of the 152.3.64.* subnet. Your output should not include those hosts without a hostname or any other extraneous output. Give the full command and a small**

3.6 f **1 / 1**

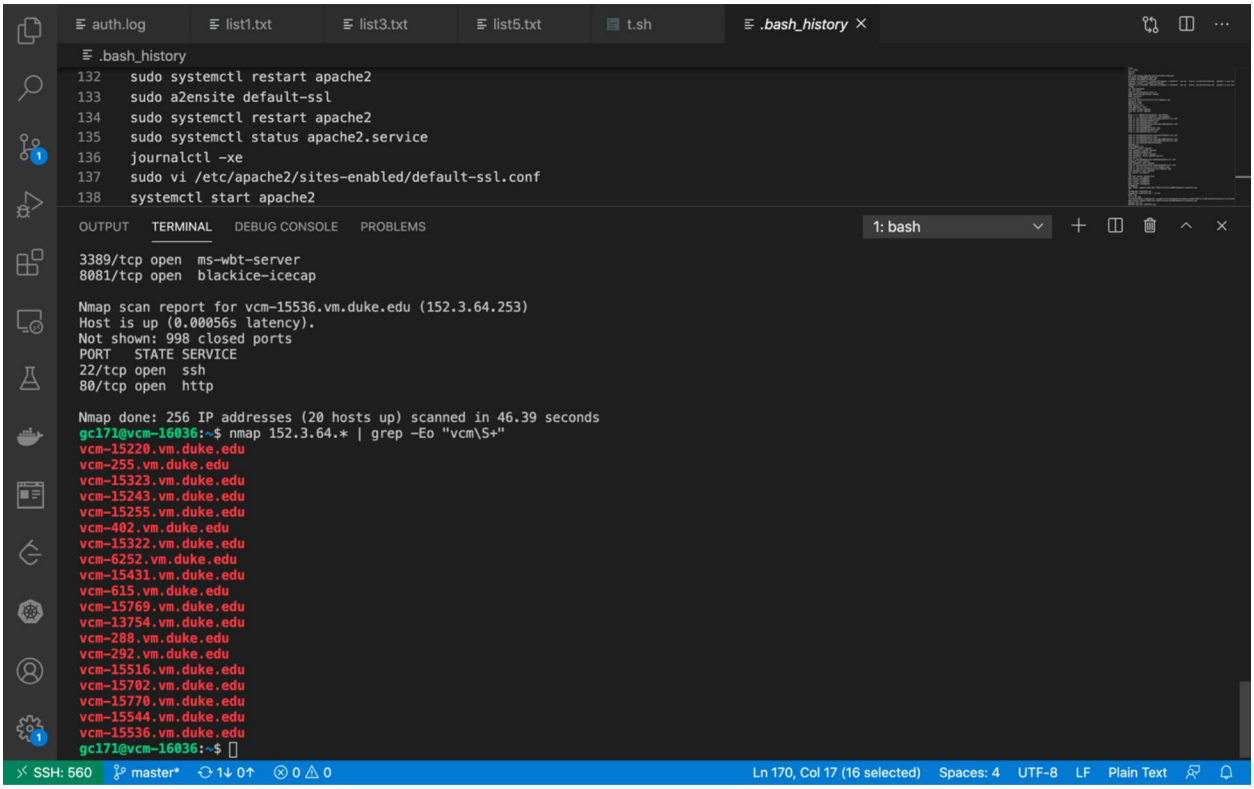
✓ **- 0 pts** **Correct**

**- 1 pts** Incorrect (chown)

**sample (5 hosts) of output. Do not use -sT or -sS in the Nmap Scan. A sample screenshot (with command blurred out of course) is shown below:**


An attacker might use output like the above to better understand a target environment, especially after they've gained a foothold to an internal network.

nmap 152.3.64.* | grep -Eo "vcm\S+"




## Task 2: Log analysis (4 points)

In Homework 1's question 2, we reviewed the output of Logwatch. That tool analyzes system logs such as the authorization log **auth.log**. Let's do a bit of similar analysis ourselves.

**Using wget, obtain this auth.log from a real production server on the internet.**

**Write a single command that identifies all the *unique* IP addresses that tried and failed to login using the invalid user 'admin'. Post a screenshot.**

### 4.1 Filter nmap output 4 / 4

✓ - 0 pts Correct

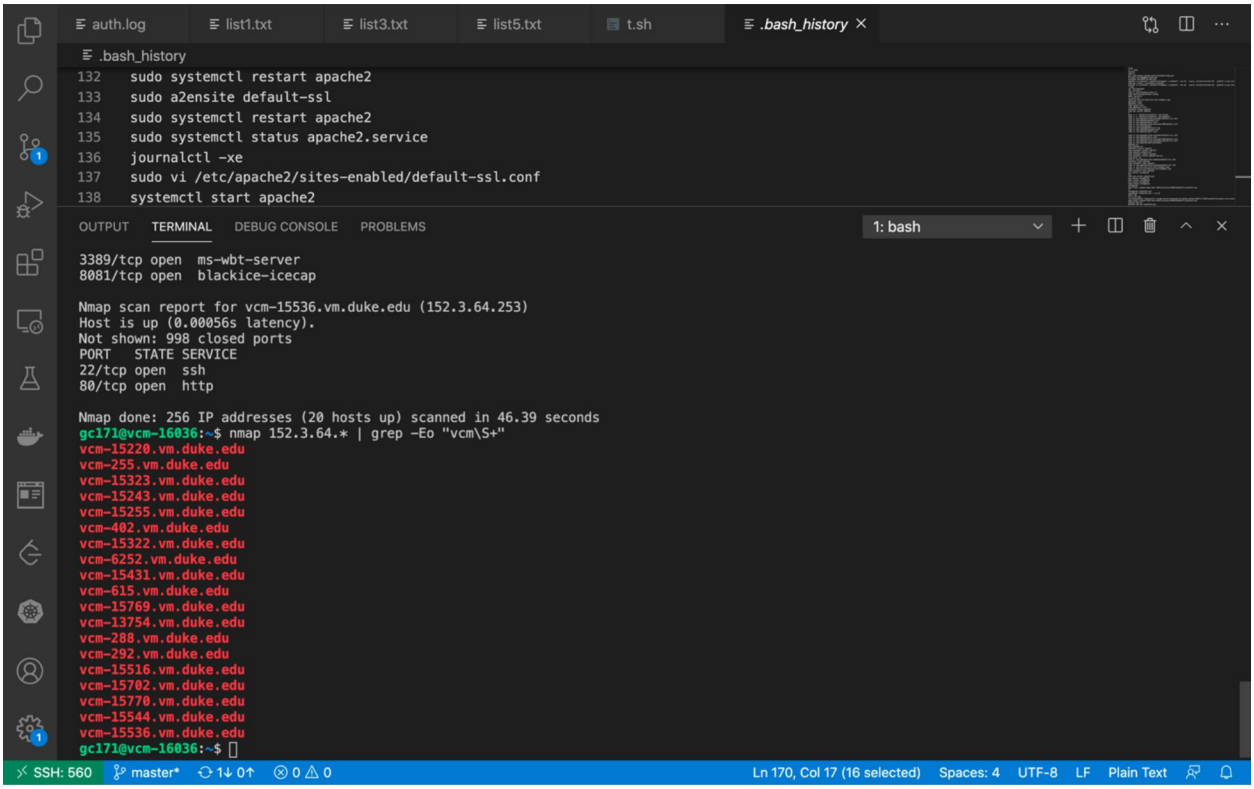
- 2 pts No output sample

- 2 pts Extraneous output present

**sample (5 hosts) of output. Do not use -sT or -sS in the Nmap Scan. A sample screenshot (with command blurred out of course) is shown below:**


An attacker might use output like the above to better understand a target environment, especially after they've gained a foothold to an internal network.

nmap 152.3.64.* | grep -Eo "vcm\S+"



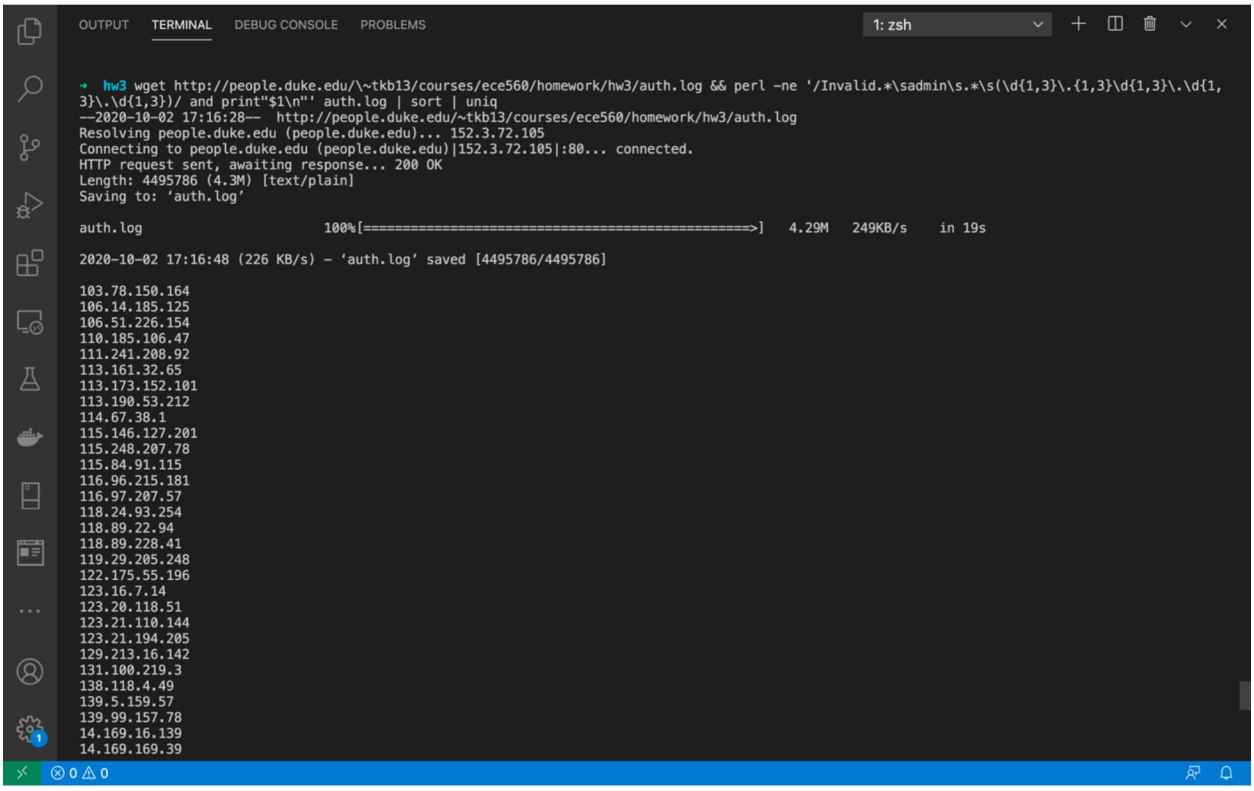

## Task 2: Log analysis (4 points)

In Homework 1's question 2, we reviewed the output of Logwatch. That tool analyzes system logs such as the authorization log **auth.log**. Let's do a bit of similar analysis ourselves.

**Using wget, obtain this auth.log from a real production server on the internet.**

**Write a single command that identifies all the *unique* IP addresses that tried and failed to login using the invalid user 'admin'. Post a screenshot.**

wget http://people.duke.edu/\~tkb13/courses/ece560/homework/hw3/auth.log && perl -ne '/Invalid.*\sadmin\s.*\s(\d{1,3}\.{1,3}\d{1,3}\.\d{1,3}\.\d{1,3})/ and print"$1\n"' auth.log | sort | uniq




## Task 3: Web app needle in a haystack (8 points)

A mock web application has been set up here:

http://target.colab.duke.edu/app/

The term “web application” here is a misnomer: for safety reasons, there’s no actual server-side code. Rather, there’s “lorem ipsum” content with hyperlinks that branch out to much of the same, with a few hidden things mixed in.

**Using shell-based tools, identify the two pages that have an HTML <form> tag and give the sequence of clicks needed to find these pages from the start. Show your work.**

**keep grep recursively**

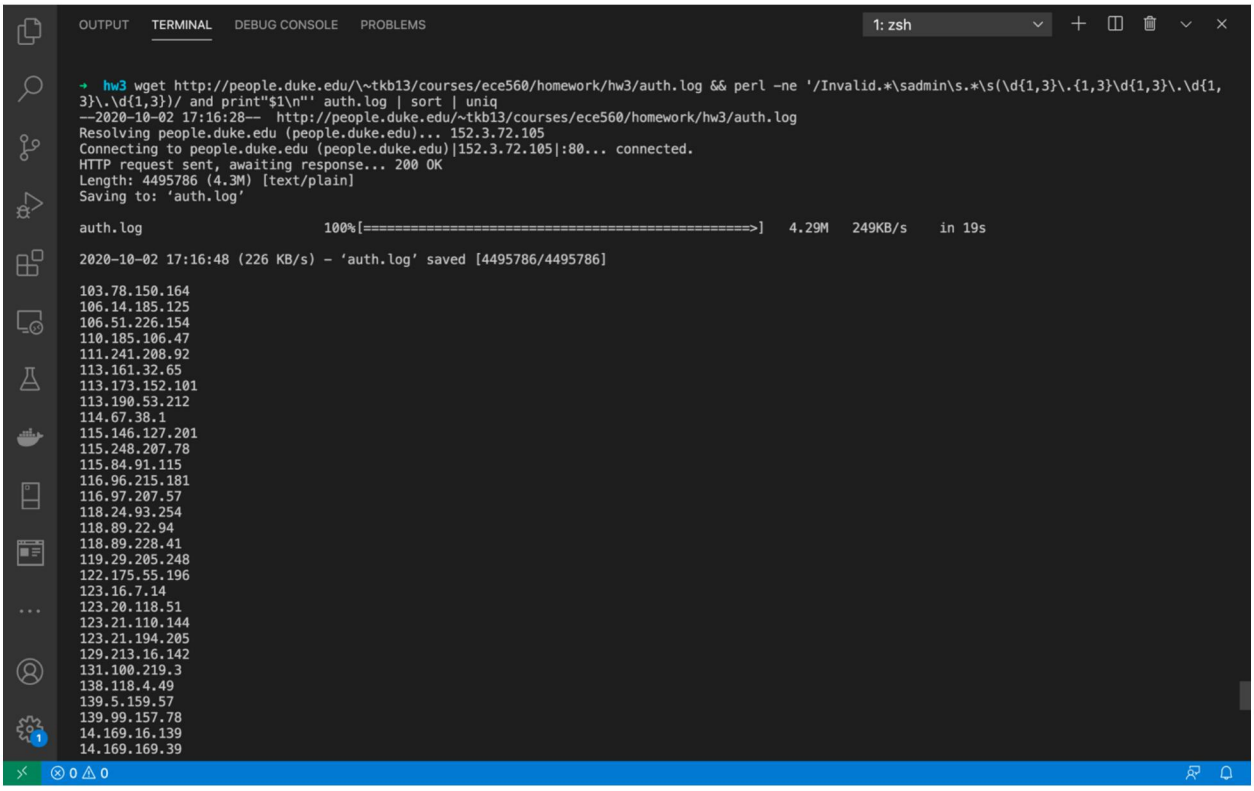
**porro -> magnam -> quisquam -> modi -> form**

### 4.2 Log analysis **4 / 4**

✓ **- 0 pts** **Correct**

**- 1.5 pts** IPs not unique!

wget http://people.duke.edu/\~tkb13/courses/ece560/homework/hw3/auth.log && perl -ne '/Invalid.*\sadmin\s.*\s(\d{1,3}\.{1,3}\d{1,3}\.\d{1,3}\.\d{1,3})/ and print"$1\n"' auth.log | sort | uniq




## Task 3: Web app needle in a haystack (8 points)

A mock web application has been set up here:

http://target.colab.duke.edu/app/

The term “web application” here is a misnomer: for safety reasons, there’s no actual server-side code. Rather, there’s “lorem ipsum” content with hyperlinks that branch out to much of the same, with a few hidden things mixed in.

**Using shell-based tools, identify the two pages that have an HTML <form> tag and give the sequence of clicks needed to find these pages from the start. Show your work.**

**keep grep recursively**

**porro -> magnam -> quisquam -> modi -> form**

```
→  hw3 grep '<form>' -r target.colab.duke.edu
target.colab.duke.edu/app/X-d1818c1ddd12f4455a79b7a74844e30d.html:<form><input type=text value='Sensitive vulnerable stuf
f'><input type=submit></form>
target.colab.duke.edu/app/X-8dbb314e8ce2364be7117ca36eba64a4.html:<form><input type=text value='Sensitive vulnerable stuf
f'><input type=submit></form>
→  hw3 grep 'X-d1818c1ddd12f4455a79b7a74844e30d.html' -r target.colab.duke.edu
target.colab.duke.edu/app/X-f1b68449cabb61422b31ba9988e05be8.html:<p>Modi tempora amet ut ipsum sed. Dolore quisquam cons
ectetur aliquam modi dolor modi numquam. Labore porro quaerat quiquia etincidunt amet quisquam dolorem. Dolore aliquam vo
luptatem <a href='X-d1818c1ddd12f4455a79b7a74844e30d.html'>modi</a> consectetur etincidunt dolorem dolorem. Numquam dolor
em dolorem numquam. Voluptatem neque velit aliquam porro velit.</p>
→  hw3 grep 'X-f1b68449cabb61422b31ba9988e05be8' -r target.colab.duke.edu
target.colab.duke.edu/app/X-71c4082a0a9ba00b7fb3344c78a53421.html:<html><body><p>Dolore quisquam ut consectetur velit qui
squam sit numquam. Labore dolor ipsum est labore quaerat est. Ipsum porro labore adipisci neque adipisci. Quiquia consect
etur sed quiquia ipsum velit. Eius quisquam est porro adipisci eius. Etincidunt dolor amet neque <a href='X-f1b68449cabb6
1422b31ba9988e05be8.html'>quisquam</a> dolorem ipsum sed.</p>
→  hw3 grep 'X-71c4082a0a9ba00b7fb3344c78a53421.html' -r target.colab.duke.edu
target.colab.duke.edu/app/X-6190da6200027751ba12b32e85d04bac.html:<p>Quiquia velit modi porro. Consectetur ut eius non ma
gnam. Dolore ut labore numquam amet consectetur ipsum voluptatem. Numquam non <a href='X-71c4082a0a9ba00b7fb3344c78a53421
.html'>magnam</a> non modi modi consectetur. Consectetur velit quisquam sed. Non non quiquia etincidunt. Neque velit magn
am non eius ut eius.</p>
→  hw3 grep 'X-6190da6200027751ba12b32e85d04bac' -r target.colab.duke.edu
target.colab.duke.edu/app/index.html:<p>Est ipsum etincidunt quisquam quiquia dolor quiquia. Quisquam non sit sit. Ut dol
or sit porro neque etincidunt quisquam adipisci. Tempora amet sit adipisci. Porro dolore quiquia numquam amet dolorem sed
. Neque labore adipisci velit neque quaerat adipisci quiquia. Etincidunt ipsum sed etincidunt neque aliquam est labore. U
t aliquam labore sed consectetur tempora tempora. Sed ipsum quiquia <a href='X-6190da6200027751ba12b32e85d04bac.html'>por
ro</a> porro.</p>
→  hw3
```

**magnam->ipsum -> voluptatem->quisquam->form**

```
→  hw3 grep '<form>' -r target.colab.duke.edu
target.colab.duke.edu/app/X-d1818c1ddd12f4455a79b7a74844e30d.html:<form><input type=text value='Sensitive vulnerable stuf
f'><input type=submit></form>
target.colab.duke.edu/app/X-8dbb314e8ce2364be7117ca36eba64a4.html:<form><input type=text value='Sensitive vulnerable stuf
f'><input type=submit></form>
→  hw3 grep 'X-8dbb314e8ce2364be7117ca36eba64a4' -r target.colab.duke.edu
target.colab.duke.edu/app/X-94867c7319f36f55e386516f6d06c419.html:<p>Ipsum quisquam tempora ipsum ut consectetur. Sed est
 numquam ut sed quiquia voluptatem. Sed eius est modi modi dolor. Tempora ipsum dolore velit. Velit numquam dolorem etinc
idunt porro eius dolorem porro. Modi porro dolor ut. Sit dolor numquam sed. Quiquia consectetur ut neque non dolor numqua
m labore. Voluptatem amet dolorem ut dolor. Neque tempora <a href='X-8dbb314e8ce2364be7117ca36eba64a4.html'>quisquam</a>
ipsum labore.</p>
→  hw3 grep 'X-94867c7319f36f55e386516f6d06c419' -r target.colab.duke.edu
target.colab.duke.edu/app/X-76547076d3aeff1a64770afee37a59a6.html:<p>Est est magnam est sit magnam neque neque. Dolor eiu
s labore est porro. Amet non quisquam tempora dolor modi. Quaerat porro neque labore labore etincidunt tempora numquam. N
umquam tempora ut quaerat velit. Neque sit labore velit. Voluptatem consectetur eius est amet consectetur. Quisquam non d
olor velit eius. Consectetur ipsum aliquam etincidunt modi eius sit. Numquam adipisci quisquam eius sed <a href='X-94867c
7319f36f55e386516f6d06c419.html'>voluptatem</a>.</p>
→  hw3 grep 'X-76547076d3aeff1a64770afee37a59a6' -r target.colab.duke.edu
target.colab.duke.edu/app/X-45fbf2013620801bdb44e119059bcaf7.html:<p>Velit quiquia labore adipisci magnam labore velit. C
onsectetur dolor labore <a href='X-76547076d3aeff1a64770afee37a59a6.html'>ipsum</a> sed dolor. Quisquam adipisci magnam c
onsectetur ipsum velit non. Dolorem voluptatem sed eius non porro velit tempora. Amet eius tempora eius ipsum quisquam es
t. Aliquam dolorem magnam quisquam neque labore quaerat. Magnam modi sit etincidunt sed. Etincidunt non amet velit labore
 non. Quisquam voluptatem adipisci ipsum adipisci dolore. Aliquam sit dolore quisquam tempora eius magnam.</p>
→  hw3 grep 'X-45fbf2013620801bdb44e119059bcaf7' -r target.colab.duke.edu
target.colab.duke.edu/app/index.html:<p>Amet est dolor tempora ipsum ipsum. Dolorem adipisci non dolore porro. Aliquam ne
que sit porro. Numquam dolorem eius ipsum dolore modi quaerat. Magnam ut quiquia quisquam. Quaerat tempora amet magnam ei
us aliquam porro tempora. Quisquam quaerat velit etincidunt consectetur amet amet eius. Aliquam quaerat <a href='X-45fbf2
013620801bdb44e119059bcaf7.html'>magnam</a> dolor eius adipisci velit. Magnam amet tempora eius voluptatem. Dolore volupt
atem quiquia aliquam tempora.</p>
→  hw3
```

**OPTIONAL: For up to 4 points of extra credit, find:**

1. A fat dog
2. A unicorn

As with the main question, give the page with the content and the click sequence needed to get there from the start. Show your work.

## fat dog

```
➜  hw3 grep 'fg' -r target.colab.duke.edu/app
Binary file target.colab.duke.edu/app/fd.jpeg matches
➜  hw3 grep 'img' -r target.colab.duke.edu/app
target.colab.duke.edu/app/X-a370360387b805b79d3378c26a8bcfe6.html:<p><img src="fd.jpg"></p>
➜  hw3 grep 'X-a370360387b805b79d3378c26a8bcfe6' -r target.colab.duke.edu/app
target.colab.duke.edu/app/X-acf700fb549363bf8a34b98ee18b072d.html:<html><body><p>Aliquam voluptatem quisquam velit quisquam. Dolore magnam quisqu
am quaerat. Numquam labore magnam eius magnam labore adipisci dolorem. Quaerat neque dolore neque labore labore aliquam. Numquam labore porro tem
pora ipsum. Quiquia aliquam <a href='X-a370360387b805b79d3378c26a8bcfe6.html'>consectetur</a> quaerat eius modi quisquam consectetur. Porro dolor
e porro labore aliquam voluptatem dolorem.</p>
➜  hw3 grep 'X-acf700fb549363bf8a34b98ee18b072d' -r target.colab.duke.edu/app
target.colab.duke.edu/app/X-0fefa92a30542ba7e95bc8a456b38658.html:<p>Sit eius sit est dolor non. Neque velit voluptatem tempora sit quaerat dolor
e. Adipisci modi labore magnam. Quiquia amet sit numquam. <a href='X-acf700fb549363bf8a34b98ee18b072d.html'>Ipsum</a> eius dolore tempora est sit
. Ut voluptatem amet dolorem consectetur eius numquam. Modi tempora ut velit modi.</p>
➜  hw3 grep 'X-0fefa92a30542ba7e95bc8a456b38658' -r target.colab.duke.edu/app
target.colab.duke.edu/app/X-c41a1e818940e17083e9cb6d2510f13c.html:<p>Tempora dolorem tempora magnam tempora. Numquam adipisci est labore quiquia.
 Magnam sed labore dolorem labore aliquam. Non modi magnam eius sed. Quaerat dolore aliquam amet quisquam. Neque quisquam dolorem non modi consec
tetur est. Non dolorem ut etincidunt ipsum ut neque etincidunt. Magnam est labore magnam dolor aliquam. Dolore porro <a href='X-0fefa92a30542ba7e
95bc8a456b38658.html'>tempora</a> consectetur amet. Aliquam adipisci quisquam quaerat.</p>
➜  hw3 grep 'X-c41a1e818940e17083e9cb6d2510f13c' -r target.colab.duke.edu/app
target.colab.duke.edu/app/index.html:<p>Labore <a href='X-c41a1e818940e17083e9cb6d2510f13c.html'>dolor</a> velit voluptatem etincidunt etincidunt
 adipisci. Velit sit consectetur ipsum quaerat. Amet sit modi voluptatem dolore sed quiquia numquam. Consectetur aliquam non quisquam neque volup
tatem. Quisquam non eius est labore ut. Magnam non non modi adipisci modi porro.</p>
➜  hw3
```

## dolor->tempora->ipsum->sonsectetur

## Question 4: Regular expressions (9 points)

### NetID validator (3 points)

Develop a regular expression that matches if and only if the input is a valid NetID (1-3 letters, 1 or more digits).

/[a-zA-Z]{1,3}[\d]+/

## 4.3 Web app needle in a haystack 10 / 8

✓ **- 0 pts** **Correct**

**+ 4 pts** Extra credit!

**- 2 pts** Didn't give actual click sequences but did give all intermediate pages

**- 6 pts** Found incorrect links, basic strategy right

**- 4 pts** No click sequences

**+ 1 pts** Extra credit, no click sequences

**+ 3 pts** Extra credit, no click sequences but correct intermediate pages

**- 6 pts** No work shown

**+ 2 Point adjustment**

1. Extra credit item found

## fat dog

```
➜  hw3 grep 'fg' -r target.colab.duke.edu/app
Binary file target.colab.duke.edu/app/fd.jpeg matches
➜  hw3 grep 'img' -r target.colab.duke.edu/app
target.colab.duke.edu/app/X-a370360387b805b79d3378c26a8bcfe6.html:<p><img src="fd.jpg"></p>
➜  hw3 grep 'X-a370360387b805b79d3378c26a8bcfe6' -r target.colab.duke.edu/app
target.colab.duke.edu/app/X-acf700fb549363bf8a34b98ee18b072d.html:<html><body><p>Aliquam voluptatem quisquam velit quisquam. Dolore magnam quisqu
am quaerat. Numquam labore magnam eius magnam labore adipisci dolorem. Quaerat neque dolore neque labore labore aliquam. Numquam labore porro tem
pora ipsum. Quiquia aliquam <a href='X-a370360387b805b79d3378c26a8bcfe6.html'>consectetur</a> quaerat eius modi quisquam consectetur. Porro dolor
e porro labore aliquam voluptatem dolorem.</p>
➜  hw3 grep 'X-acf700fb549363bf8a34b98ee18b072d' -r target.colab.duke.edu/app
target.colab.duke.edu/app/X-0fefa92a30542ba7e95bc8a456b38658.html:<p>Sit eius sit est dolor non. Neque velit voluptatem tempora sit quaerat dolor
e. Adipisci modi labore magnam. Quiquia amet sit numquam. <a href='X-acf700fb549363bf8a34b98ee18b072d.html'>Ipsum</a> eius dolore tempora est sit
. Ut voluptatem amet dolorem consectetur eius numquam. Modi tempora ut velit modi.</p>
➜  hw3 grep 'X-0fefa92a30542ba7e95bc8a456b38658' -r target.colab.duke.edu/app
target.colab.duke.edu/app/X-c41a1e818940e17083e9cb6d2510f13c.html:<p>Tempora dolorem tempora magnam tempora. Numquam adipisci est labore quiquia.
 Magnam sed labore dolorem labore aliquam. Non modi magnam eius sed. Quaerat dolore aliquam amet quisquam. Neque quisquam dolorem non modi consec
tetur est. Non dolorem ut etincidunt ipsum ut neque etincidunt. Magnam est labore magnam dolor aliquam. Dolore porro <a href='X-0fefa92a30542ba7e
95bc8a456b38658.html'>tempora</a> consectetur amet. Aliquam adipisci quisquam quaerat.</p>
➜  hw3 grep 'X-c41a1e818940e17083e9cb6d2510f13c' -r target.colab.duke.edu/app
target.colab.duke.edu/app/index.html:<p>Labore <a href='X-c41a1e818940e17083e9cb6d2510f13c.html'>dolor</a> velit voluptatem etincidunt etincidunt
 adipisci. Velit sit consectetur ipsum quaerat. Amet sit modi voluptatem dolore sed quiquia numquam. Consectetur aliquam non quisquam neque volup
tatem. Quisquam non eius est labore ut. Magnam non non modi adipisci modi porro.</p>
➜  hw3
```

## dolor->tempora->ipsum->sonsectetur

## Question 4: Regular expressions (9 points)

### NetID validator (3 points)

Develop a regular expression that matches if and only if the input is a valid NetID (1-3 letters, 1 or more digits).

/[a-zA-Z]{1,3}[\d]+/

## 5.1 NetID validator 2 / 3

- **0 pts** Correct

✓ - **1 pts** **No anchors (netid could be found in the middle of a non-netid string) , ^ for start anchor, $ for end anchor**

- **1 pts** Wrong expression for letters. \\w includes digits and underscore. Use \[a-zA-Z]

- **1 pts** Wrong expression for digits. Use \[0-9] or \\d

- **0 pts** Not considering case sensitive for NEtID in this case

https://piazza.com/class/kd6irc9vkbl23b?cid=226

### URL parser (3 points)

Develop a regular expression that, for a given URL, matches if and only if it is a Wikipedia page, and it captures just the article part of the URL. It should work for HTTP and HTTPS. Examples:

| URL | Match? | Match group 1 |
|---|---|---|
| http://en.wikipedia.org/wiki/Computer_security | yes | Computer_security |
| https://en.wikipedia.org/wiki/Transport_Layer_Security | yes | Transport_Layer_Security |
| https://duke.edu/ | no | - |

/^(?:http|https):\/\/en.wikipedia.org\/wiki\/(.+)/

link to test: https://regex101.com/r/PHn6Yw/2

### Pi digits (3 points)

Develop a shell command to print the MD5 hash of pi (π) up to and including the first appearance of decimal digits "12345". To help check your work, the last byte of the answer is 0x02. You can download the value of pi, no need to compute it. For actually employing the regex, you can use grep with -E and other options, perl, Python, or other tools.

```
grep -Eo "(^3\.\d*12345)" pi.txt | tr -d '\n' | md5sum
```

## Question 5: GNU Screen (2 points)

Screen is a full-screen window manager that multiplexes a physical terminal (or GUI terminal window) between several processes, typically interactive shells. When screen is called, it creates a single terminal with a shell in it (or the specified command) and then gets out of your way so that you can use the program as you normally would. Then, at any time, you can create new (full-screen) terminals with other programs in them (including more shells), kill the current window, view a list of the active windows, turn output logging on and off, view the scrollback history, switch between terminals, etc. All terminals run their programs completely independent of each other. Especially useful for us is the fact you can *detach* a running screen session from your console and let all the associated terminals keep running in the background, then later *reattach* it to another console. Programs continue to run even when screen is detached.

Review this very brief video intro to screen. Here is a screen quick reference.

### 5.2 URL parser **2.5 / 3**

- **0 pts** Correct

✓ - **0.5 pts** **Didn't escape dots**

- **0.5 pts** Extra capture group
- **1 pts** No capture groups
- **0.25 pts** No beginning anchor

### URL parser (3 points)

Develop a regular expression that, for a given URL, matches if and only if it is a Wikipedia page, and it captures just the article part of the URL. It should work for HTTP and HTTPS. Examples:

| URL | Match? | Match group 1 |
|---|---|---|
| http://en.wikipedia.org/wiki/Computer_security | yes | Computer_security |
| https://en.wikipedia.org/wiki/Transport_Layer_Security | yes | Transport_Layer_Security |
| https://duke.edu/ | no | - |

/^(?:http|https):\/\/en.wikipedia.org\/wiki\/(.+)/

link to test: https://regex101.com/r/PHn6Yw/2

### Pi digits (3 points)

Develop a shell command to print the MD5 hash of pi (π) up to and including the first appearance of decimal digits "12345". To help check your work, the last byte of the answer is 0x02. You can download the value of pi, no need to compute it. For actually employing the regex, you can use grep with -E and other options, perl, Python, or other tools.

```
grep -Eo "(^3\.\d*12345)" pi.txt | tr -d '\n' | md5sum
```

## Question 5: GNU Screen (2 points)

Screen is a full-screen window manager that multiplexes a physical terminal (or GUI terminal window) between several processes, typically interactive shells. When screen is called, it creates a single terminal with a shell in it (or the specified command) and then gets out of your way so that you can use the program as you normally would. Then, at any time, you can create new (full-screen) terminals with other programs in them (including more shells), kill the current window, view a list of the active windows, turn output logging on and off, view the scrollback history, switch between terminals, etc. All terminals run their programs completely independent of each other. Especially useful for us is the fact you can *detach* a running screen session from your console and let all the associated terminals keep running in the background, then later *reattach* it to another console. Programs continue to run even when screen is detached.

Review this very brief video intro to screen. Here is a screen quick reference.

### 5.3 Pi digits **2.5 / 3**

- **0 pts** Correct

✓ **- 0.5 pts** **Forgot to get rid of greedy repetition (you may still get the correct hash w/o it, but there was no way for you to know that in advance)**

### URL parser (3 points)

Develop a regular expression that, for a given URL, matches if and only if it is a Wikipedia page, and it captures just the article part of the URL. It should work for HTTP and HTTPS. Examples:

| URL | Match? | Match group 1 |
|---|---|---|
| http://en.wikipedia.org/wiki/Computer_security | yes | Computer_security |
| https://en.wikipedia.org/wiki/Transport_Layer_Security | yes | Transport_Layer_Security |
| https://duke.edu/ | no | - |

/^(?:http|https):\/\/en.wikipedia.org\/wiki\/(.+)/

link to test: https://regex101.com/r/PHn6Yw/2

### Pi digits (3 points)

Develop a shell command to print the MD5 hash of pi (π) up to and including the first appearance of decimal digits "12345". To help check your work, the last byte of the answer is 0x02. You can download the value of pi, no need to compute it. For actually employing the regex, you can use grep with -E and other options, perl, Python, or other tools.

```
grep -Eo "(^3\.\d*12345)" pi.txt | tr -d '\n' | md5sum
```

## Question 5: GNU Screen (2 points)

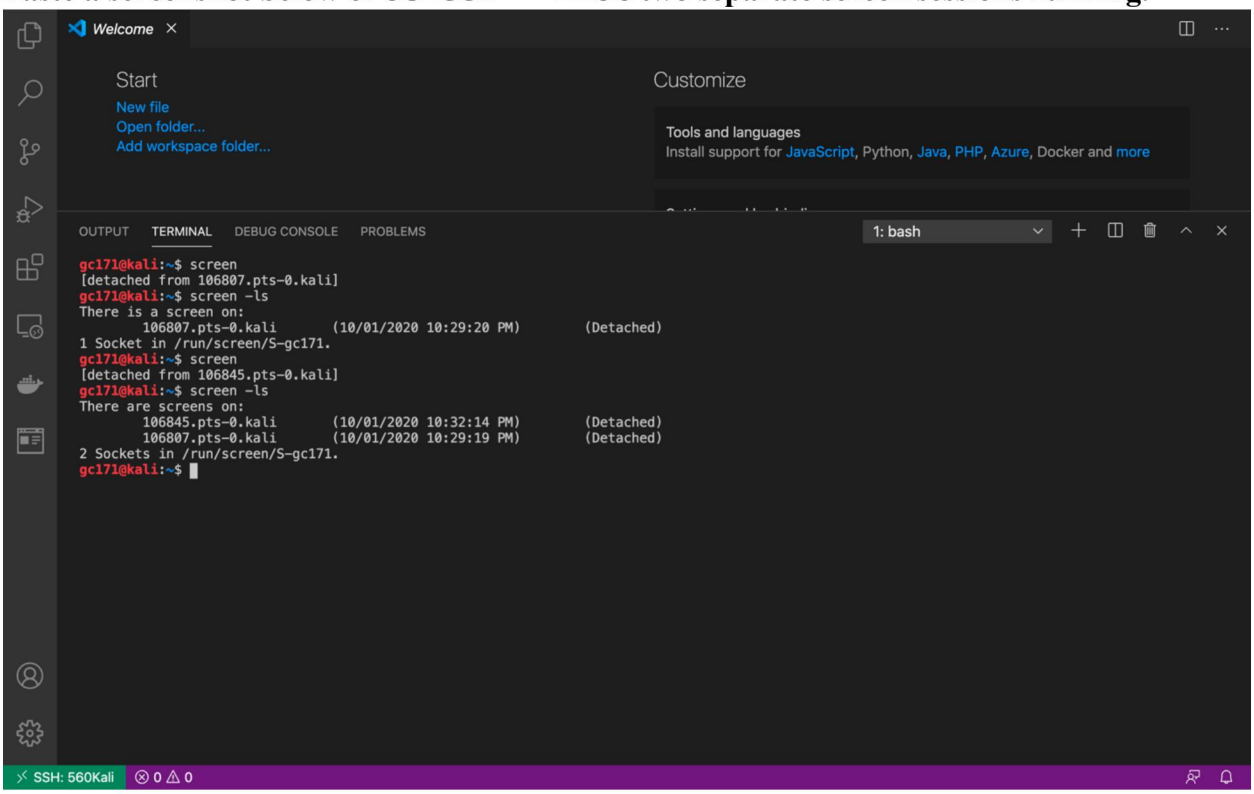
Screen is a full-screen window manager that multiplexes a physical terminal (or GUI terminal window) between several processes, typically interactive shells. When screen is called, it creates a single terminal with a shell in it (or the specified command) and then gets out of your way so that you can use the program as you normally would. Then, at any time, you can create new (full-screen) terminals with other programs in them (including more shells), kill the current window, view a list of the active windows, turn output logging on and off, view the scrollback history, switch between terminals, etc. All terminals run their programs completely independent of each other. Especially useful for us is the fact you can *detach* a running screen session from your console and let all the associated terminals keep running in the background, then later *reattach* it to another console. Programs continue to run even when screen is detached.

Review this very brief video intro to screen. Here is a screen quick reference.

This might be useful for the MD5 cracking question below, and it's essential for the problem "John the Ripper" after that. Start a new screen session on your Kali VM. Prove to yourself that you can detach, reattach, and are generally comfortable with screen.

*Note: If you prefer another terminal virtualizer, such as tmux, you may use that instead.*

**Paste a screenshot below of `screen --list` two separate screen sessions running.**




## Question 6: Cracking MD5 hashes with hashcat (6 points)

There is an FTP server called Serv-U, a commercial program written by CATsoft. Hackers sometimes install pirated copies on computers they compromise. This allows an attacker to access the file system and run commands on the compromised computer. The FTP server usernames and passwords (used by the attackers) are stored in a configuration file. The passwords are hashed with MD5 hashing algorithm. Here is a sample file from a compromised machine:

## 6 GNU Screen 2 / 2

✓ - 0 pts Correct

- 2 pts Wrong Screenshot

```
[USER=admin|1]
Password=ly5cf2ff593419bcf3d22c62e65a82128a
HomeDir=c:\
AlwaysAllowLogin=1
TimeOut=600
Maintenance=System
Access1=C:\|RWAMELCDP
```

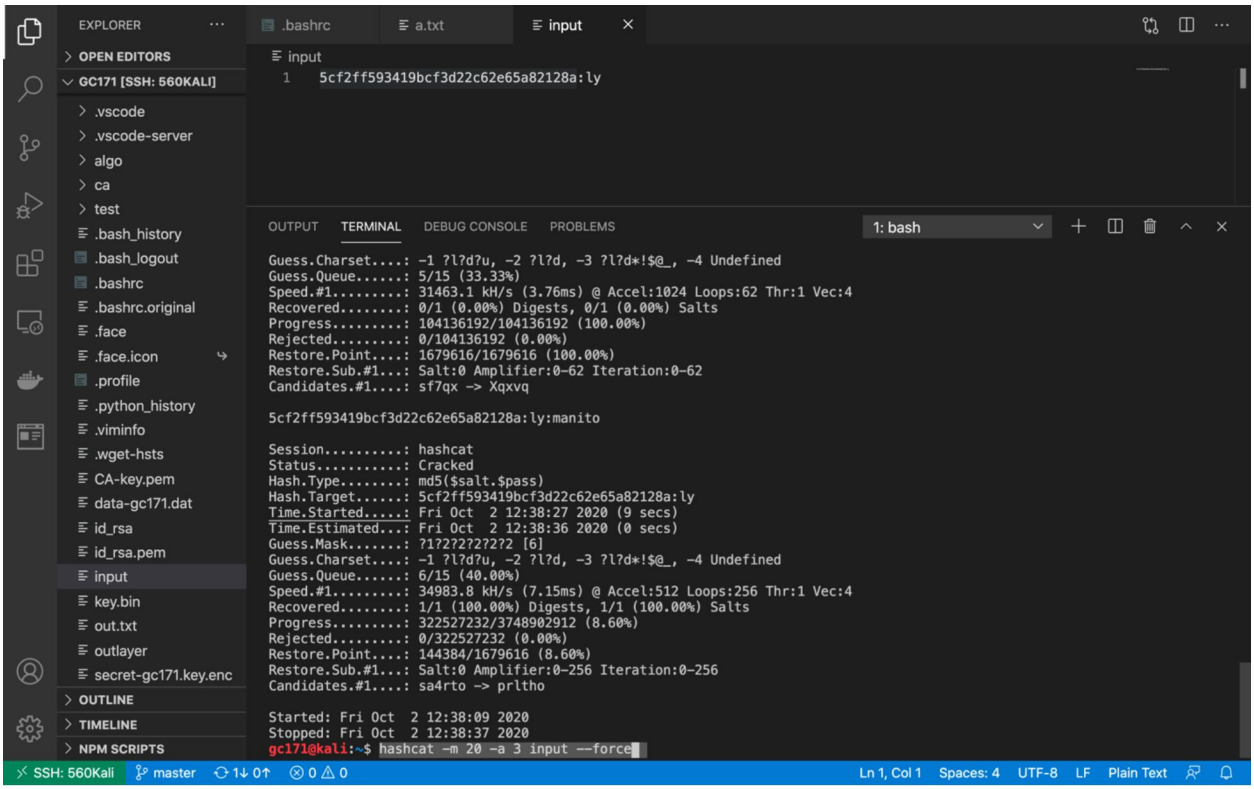
The MD5 hash of the password is on the line that starts with 'Password='. The two letters after the equal sign are the salt as ASCII text (shown in orange above) followed by the MD5 hash in hex (shown in blue above). This salt is prepended, i.e. the Serv-U password hash function is **`md5(salt+pass)`**.

Your Kali VM comes with a hash cracker called **`hashcat`**. Read up on how to use it, and have it crack the salted hash above. Tips:

- Hashcat is optimized for GPUs, but your VM doesn't have one. Override the resulting error message using the `--force` flag.
- You will need to create a "hash file" as input; the format of this file is "**`<hash>:<salt>`**".
- The default brute force mode will be fine, and should take around 30 seconds on the Kali VM.
- Found passwords are echoed in **`<hash>:<salt>:<pass>`** format and also saved to **`~/.hashcat/hashcat.potfile`**.

**Paste (1) the command executed, (2) a screenshot of the output, and (3) the password itself below.**

**hashcat -m 20 -a 3 input –force**




**manito**

**OPTIONAL: For up to 6 points of extra credit, crack each of the following Serv-U hashes.**

You may need to use more time, wordlists, and possibly even a combination wordlist/brute-force attack.

a. `qje39945197dbc380a5ac611fbbb7b5354`
b. `csa2eb704ec429728c1fe15814c006a4cc`
c. `qj847d6d5952baef48fa101d84a18b8de8`

**baseball**

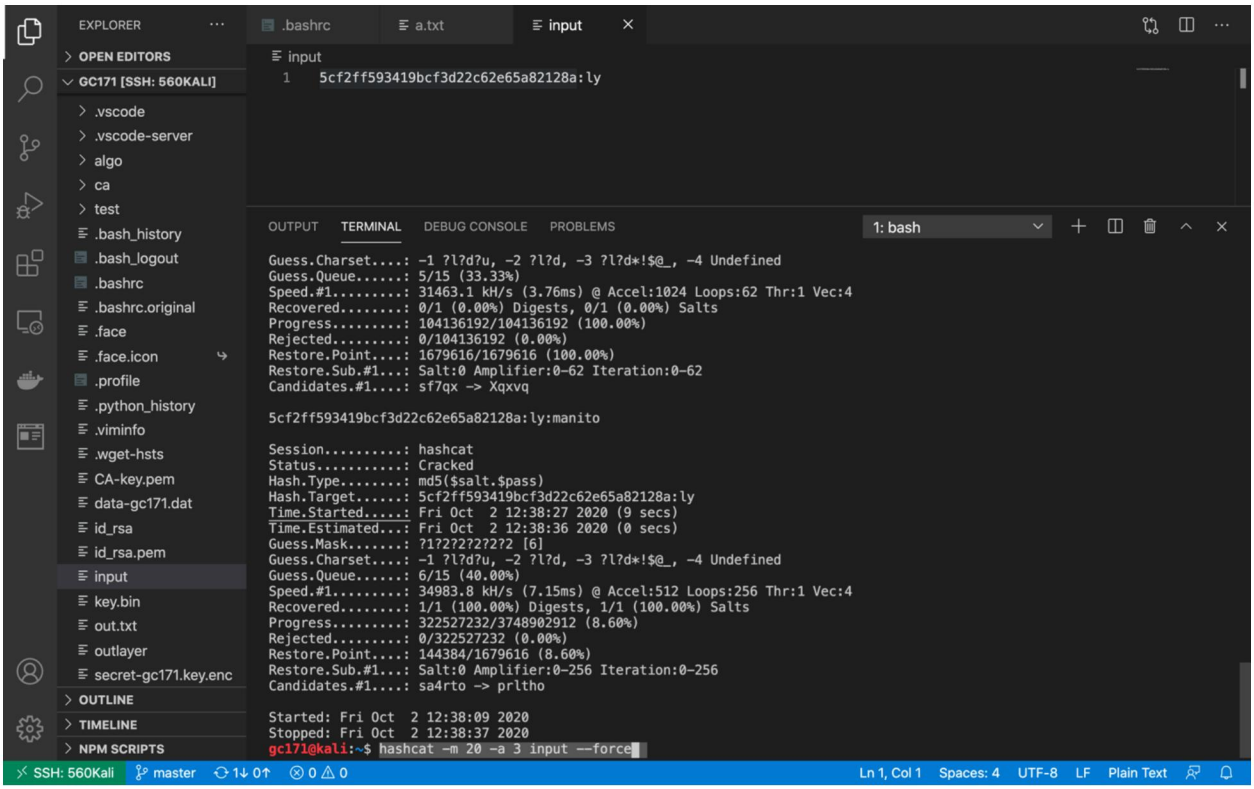
## Question 7: John the Ripper (10 points)

To start, do some research, and **describe in detail (purpose and content) the /etc/passwd and /etc/shadow files on a Linux system. (4 points)**

| | purpose | Content |
|---|---|---|

7 Cracking MD5 hashes with hashcat **8 / 6**

- ✓ **- 0 pts** **Correct**
- **- 2 pts** No Screenshot
- **- 2 pts** Wrong password
- **- 2 pts** No command
- ✓ **+ 2 pts** **Extra Credit**
- **+ 4 pts** Extra Credit
- **+ 6 pts** Extra Credit




**manito**

**OPTIONAL: For up to 6 points of extra credit, crack each of the following Serv-U hashes.**

You may need to use more time, wordlists, and possibly even a combination wordlist/brute-force attack.

a. `qje39945197dbc380a5ac611fbbb7b5354`
b. `csa2eb704ec429728c1fe15814c006a4cc`
c. `qj847d6d5952baef48fa101d84a18b8de8`

**baseball**

## Question 7: John the Ripper (10 points)

To start, do some research, and **describe in detail (purpose and content) the /etc/passwd and /etc/shadow files on a Linux system. (4 points)**

| | purpose | Content |
|---|---|---|

| | | |
|---|---|---|
| /etc/passwd | Store user account details | Attributes of each user, aims at user account details, world readable |
| /etc/shadow | Store User password details | Aims at user password, encrypted password store here, only read by root, |

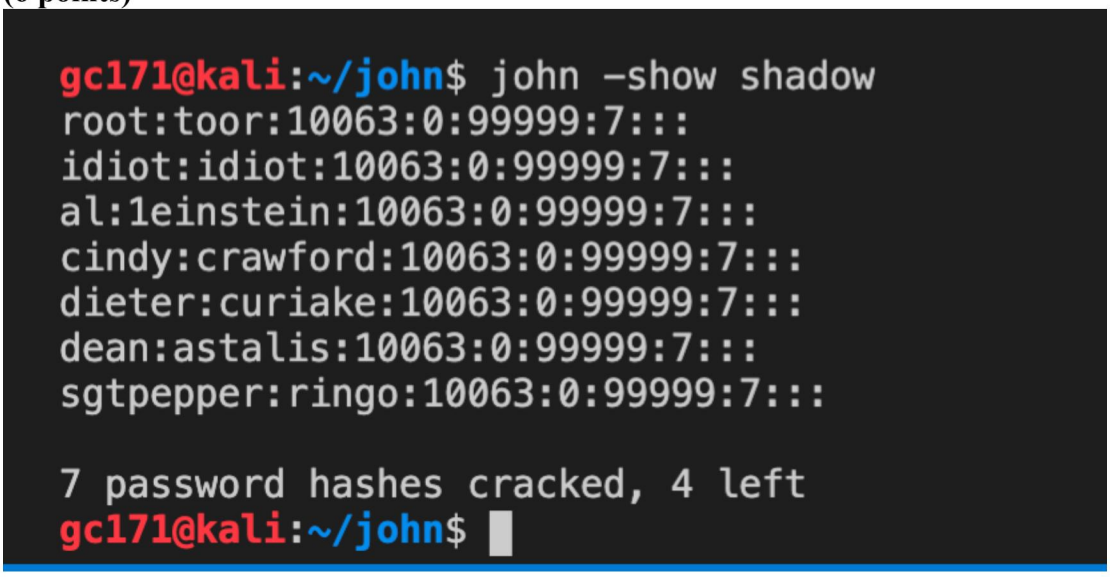
For this exercise, we will use John the Ripper, which is already installed on your Kali VM. I'll be supplying the shadow file for the exercise.

We will be using the shadow file here: http://people.duke.edu/~tkb13/courses/ece590-sec/homework/shadow

Download this shadow file to your Kali VM. Unlike hashcat, john has pretty smart defaults and wordlists, so you can just run it against the shadow file. To improve performance, you can specify **`--fork=<N>`** to run *N* instances in parallel; set *N* to the number of CPU cores on your system (you can check with `top`, `cat /proc/cpuinfo`, etc.).

Run `john` on the provided shadow file in a `screen` session (so you can detach, disconnect, then reattach later). Let it run for at least 24 hours. It is not expected to be able to crack all the passwords, but you should get at least 6.

**Paste the output your results with the following command "`john -show shadow`". (6 points)**




It should look something like the example below:

```
$ john -show shadow
abcuser:apple1:15358:0:99999:7:::
defuser:orange:15364:0:99999:7:::
ghiuser:gpgtest:15373:0:99999:7:::
```

8 John the Ripper **10 / 10**

✓ **- 0 pts** **Correct**

**- 1 pts** Description not detailed

**- 3 pts** Major defects on description

**- 4 pts** No description

**- 1 pts** Missing one password

**- 6 pts** No password cracked

```
3 password hashes cracked, 25 left
```

Note: Here, “abcuser” is the username and “apple1” is the password.

## Question 8: Evaluating MFA approaches (8 points)

Review this article on a data theft attack against content aggregator site Reddit.

a. What form of Multi-Factor Authentication (MFA) was deployed at Reddit?

   SMS for two-factor authentication

b. What are two ways in which the attacker may have gained access to the second factor?

   SIM-swap: hackers claim as the owner of sim card and require service provider to change the service to a new SIM card hacker controls

   mobile number port-out scams: hacker claims as the customer and requests mobile number transfer to another service provider

   in both attacks, the victims service gets shut off and SMS are sent to hackers

c. What are two alternative forms of MFA that could be deployed instead?

   app-based two-factor authentication

   two-factor authentication with hardware-based security keys

d. Identify another attack in the last 5 years in which SMS-based MFA was compromised.

   https://securityboulevard.com/2019/09/does-jack-dorseys-twitter-account-hack-mean-two-factor-authentication-is-waste-of-time/

   twitter CEO account was compromised. it is based on SMS

## Question 9: SSH with MFA (5 points)

On your Linux VM, you are going to add some extra protections to the SSH server by adding Multi-Factor Authentication (MFA) to your account.

Some research will tell you how to do this using Google Authenticator or the desktop app Authy.

**Show a screenshot of your SSH login with MFA code.**

9.1 a **2 / 2**

✓ **- 0 pts** **Correct**

**- 1 pts** Did not mention SMS

**- 2 pts** No answer

```
3 password hashes cracked, 25 left
```

Note: Here, “abcuser” is the username and “apple1” is the password.

## Question 8: Evaluating MFA approaches (8 points)

Review this article on a data theft attack against content aggregator site Reddit.

a. What form of Multi-Factor Authentication (MFA) was deployed at Reddit?

   SMS for two-factor authentication

b. What are two ways in which the attacker may have gained access to the second factor?

   SIM-swap: hackers claim as the owner of sim card and require service provider to change the service to a new SIM card hacker controls

   mobile number port-out scams: hacker claims as the customer and requests mobile number transfer to another service provider

   in both attacks, the victims service gets shut off and SMS are sent to hackers

c. What are two alternative forms of MFA that could be deployed instead?

   app-based two-factor authentication

   two-factor authentication with hardware-based security keys

d. Identify another attack in the last 5 years in which SMS-based MFA was compromised.

   https://securityboulevard.com/2019/09/does-jack-dorseys-twitter-account-hack-mean-two-factor-authentication-is-waste-of-time/

   twitter CEO account was compromised. it is based on SMS

## Question 9: SSH with MFA (5 points)

On your Linux VM, you are going to add some extra protections to the SSH server by adding Multi-Factor Authentication (MFA) to your account.

Some research will tell you how to do this using Google Authenticator or the desktop app Authy.

**Show a screenshot of your SSH login with MFA code.**

9.2 b **2 / 2**

✓ **- 0 pts** **Correct**

**- 2 pts** No answer

```
3 password hashes cracked, 25 left
```

Note: Here, “abcuser” is the username and “apple1” is the password.

## Question 8: Evaluating MFA approaches (8 points)

Review this article on a data theft attack against content aggregator site Reddit.

a. What form of Multi-Factor Authentication (MFA) was deployed at Reddit?

   SMS for two-factor authentication

b. What are two ways in which the attacker may have gained access to the second factor?

   SIM-swap: hackers claim as the owner of sim card and require service provider to change the service to a new SIM card hacker controls

   mobile number port-out scams: hacker claims as the customer and requests mobile number transfer to another service provider

   in both attacks, the victims service gets shut off and SMS are sent to hackers

c. What are two alternative forms of MFA that could be deployed instead?

   app-based two-factor authentication

   two-factor authentication with hardware-based security keys

d. Identify another attack in the last 5 years in which SMS-based MFA was compromised.

   https://securityboulevard.com/2019/09/does-jack-dorseys-twitter-account-hack-mean-two-factor-authentication-is-waste-of-time/

   twitter CEO account was compromised. it is based on SMS

## Question 9: SSH with MFA (5 points)

On your Linux VM, you are going to add some extra protections to the SSH server by adding Multi-Factor Authentication (MFA) to your account.

Some research will tell you how to do this using Google Authenticator or the desktop app Authy.

**Show a screenshot of your SSH login with MFA code.**

9.3 C **2 / 2**

✓ **- 0 pts** **Correct**

**- 2 pts** No answer

```
3 password hashes cracked, 25 left
```

Note: Here, “abcuser” is the username and “apple1” is the password.

## Question 8: Evaluating MFA approaches (8 points)

Review this article on a data theft attack against content aggregator site Reddit.

a. What form of Multi-Factor Authentication (MFA) was deployed at Reddit?

   SMS for two-factor authentication

b. What are two ways in which the attacker may have gained access to the second factor?

   SIM-swap: hackers claim as the owner of sim card and require service provider to change the service to a new SIM card hacker controls

   mobile number port-out scams: hacker claims as the customer and requests mobile number transfer to another service provider

   in both attacks, the victims service gets shut off and SMS are sent to hackers

c. What are two alternative forms of MFA that could be deployed instead?

   app-based two-factor authentication

   two-factor authentication with hardware-based security keys

d. Identify another attack in the last 5 years in which SMS-based MFA was compromised.

   https://securityboulevard.com/2019/09/does-jack-dorseys-twitter-account-hack-mean-two-factor-authentication-is-waste-of-time/

   twitter CEO account was compromised. it is based on SMS

## Question 9: SSH with MFA (5 points)

On your Linux VM, you are going to add some extra protections to the SSH server by adding Multi-Factor Authentication (MFA) to your account.

Some research will tell you how to do this using Google Authenticator or the desktop app Authy.

**Show a screenshot of your SSH login with MFA code.**

9.4 d **2 / 2**

✓ **- 0 pts** **Correct**

**- 2 pts** No answer

**- 1.5 pts** The question asks for an attack event, not a method

```
3 password hashes cracked, 25 left
```

Note: Here, “abcuser” is the username and “apple1” is the password.

## Question 8: Evaluating MFA approaches (8 points)

Review this article on a data theft attack against content aggregator site Reddit.

a. What form of Multi-Factor Authentication (MFA) was deployed at Reddit?

   SMS for two-factor authentication

b. What are two ways in which the attacker may have gained access to the second factor?

   SIM-swap: hackers claim as the owner of sim card and require service provider to change the service to a new SIM card hacker controls

   mobile number port-out scams: hacker claims as the customer and requests mobile number transfer to another service provider

   in both attacks, the victims service gets shut off and SMS are sent to hackers

c. What are two alternative forms of MFA that could be deployed instead?

   app-based two-factor authentication

   two-factor authentication with hardware-based security keys

d. Identify another attack in the last 5 years in which SMS-based MFA was compromised.

   https://securityboulevard.com/2019/09/does-jack-dorseys-twitter-account-hack-mean-two-factor-authentication-is-waste-of-time/

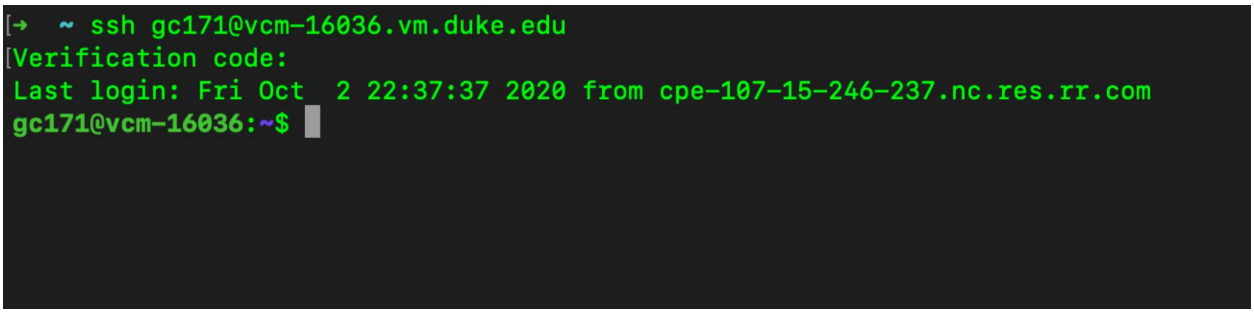
   twitter CEO account was compromised. it is based on SMS

## Question 9: SSH with MFA (5 points)

On your Linux VM, you are going to add some extra protections to the SSH server by adding Multi-Factor Authentication (MFA) to your account.

Some research will tell you how to do this using Google Authenticator or the desktop app Authy.

**Show a screenshot of your SSH login with MFA code.**




## Question 10: Hydra (Online SSH Dictionary Attack) (5 points)

Hydra is an online network-based dictionary attack tool for SSH and many other protocols. For this example, the tool will take 4 input values: username file, a password file, target list, and a service (ssh). With that information, it attempts to perform a network-based authentication to the remote machine(s) on port 22. If it successfully authenticates with the remote machine, it shows the results in the standard output with the username and passwords that worked.

For this exercise, you are going to use Hydra to perform an SSH dictionary attack against a machine specially prepared for this purpose, **target.colab.duke.edu**. Do not use the tool on any other targets!

Implementation note: Because I've intentionally put a guessable username/password combo on this machine, I've used the ufw software firewall (based on Linux iptables) to restrict SSH access to campus IP ranges.

On your Kali VM, do the following.

### Acquire a password list

A password list is a text file of likely passwords. There are *many* password lists out there, and Kali Linux ships with many of them pre-installed. For our purposes, we will use the Adobe "top 100" password list, which was derived from a 2013 breach of 150 million Adobe user accounts. You can find this password list in
**`/usr/share/wordlists/metasploit/adobe_top100_pass.txt`**

### Build user list (1 point)

If an attacker is doing a general scan, they will typically use a common username list, such as one of these. If an attacker is focused on a known server or organization, they will do research using public information to create a list of likely usernames.

In this scenario, let's assume we are targeting an organization's senior leadership, which includes Susan Hypothetical, Scott Samplesberg, and Jacob Exampleface. Based on this, produce a small text file of likely usernames.

**Paste your username list below.**

10 SSH with MFA **5 / 5**

✓ **- 0 pts** **Correct**

Susan Hypothetical
SH
SusanHypothetical
susanhypothetical
susan
susan0
susan1
susan2
susan3
Scott Samplesberg
SE
ScottSamplesberg
scottsamplesberg
scott
scott0
scott1
scott2
scott3
Jacob Exampleface
JE
JacobExampleface
jacobexampleface
jacob
jacob0
jacob1
jacob2
jacob3

## Attack (4 points)

Run hydra against target.colab.duke.edu using the resources gathered.

**Give the hydra command you used below.**

hydra -L user -P /usr/share/wordlists/metasploit/adobe_top100_pass.txt ssh://target.colab.duke.edu

**Display the successful results of your Hydra Attack below.** If you don't get any results, try more (and simpler!) usernames.

### 11.1 Build user list 1 / 1

✓ **- 0 pts** **Correct**

Susan Hypothetical
SH
SusanHypothetical
susanhypothetical
susan
susan0
susan1
susan2
susan3
Scott Samplesberg
SE
ScottSamplesberg
scottsamplesberg
scott
scott0
scott1
scott2
scott3
Jacob Exampleface
JE
JacobExampleface
jacobexampleface
jacob
jacob0
jacob1
jacob2
jacob3

## Attack (4 points)

Run hydra against target.colab.duke.edu using the resources gathered.

**Give the hydra command you used below.**

hydra -L user -P /usr/share/wordlists/metasploit/adobe_top100_pass.txt ssh://target.colab.duke.edu

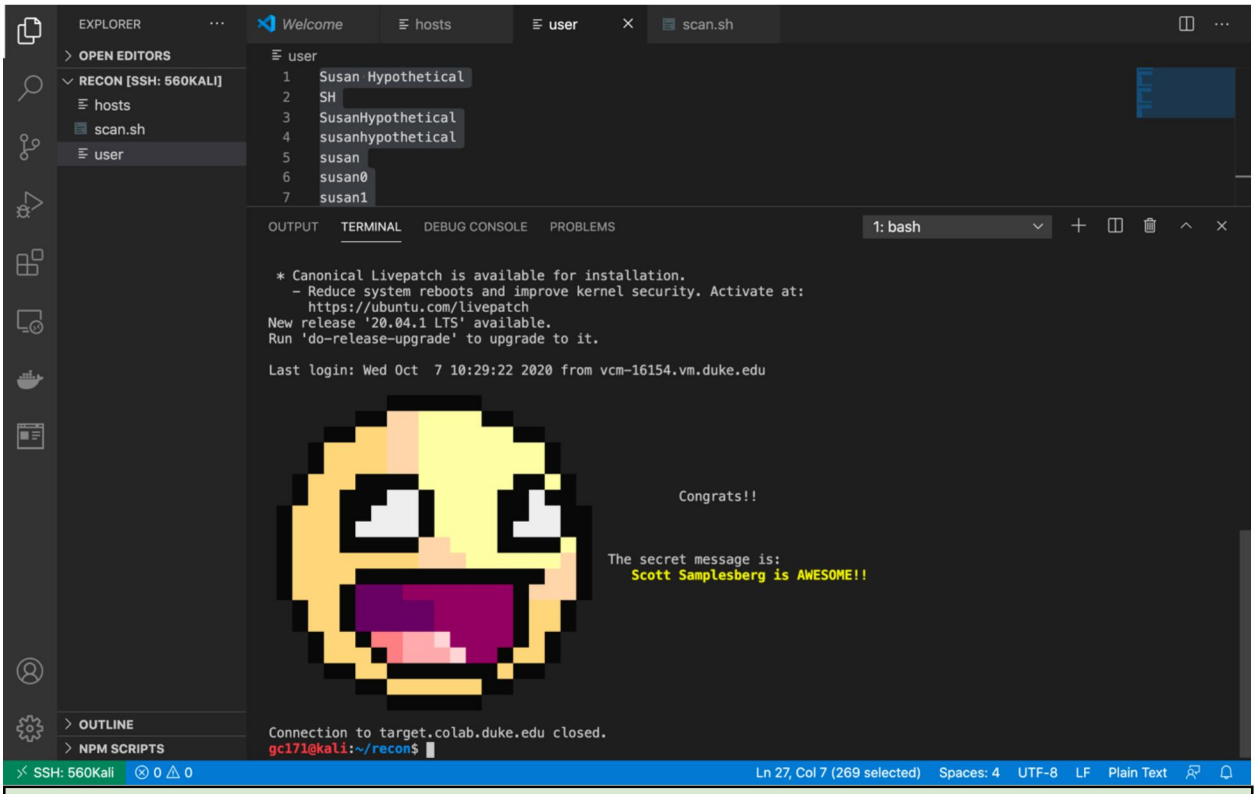
**Display the successful results of your Hydra Attack below.** If you don't get any results, try more (and simpler!) usernames.

```
gc171@kali:~/recon$ hydra -L user -P /usr/share/wordlists/metasploit/adobe_top100_pass.txt ssh://target.colab.duke.edu
Hydra v9.0 (c) 2019 by van Hauser/THC - Please do not use in military or secret service organizations, or for illegal pur
poses.

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2020-10-07 11:45:18
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[DATA] max 16 tasks per 1 server, overall 16 tasks, 2700 login tries (l:27/p:100), ~169 tries per task
[DATA] attacking ssh://target.colab.duke.edu:22/
[STATUS] 196.00 tries/min, 196 tries in 00:01h, 2507 to do in 00:13h, 16 active
[STATUS] 135.67 tries/min, 407 tries in 00:03h, 2296 to do in 00:17h, 16 active
[STATUS] 141.86 tries/min, 993 tries in 00:07h, 1710 to do in 00:13h, 16 active
[22][ssh] host: target.colab.duke.edu   login: scott   password: zxcvbnm
[STATUS] 141.33 tries/min, 1696 tries in 00:12h, 1007 to do in 00:08h, 16 active
[STATUS] 140.94 tries/min, 2396 tries in 00:17h, 309 to do in 00:03h, 16 active
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2020-10-07 12:04:52
gc171@kali:~/recon$
```

*Note: It is very likely that you will only be able to find one set of working credentials.*

**When you log into target.colab.duke.edu using the credentials found, a secret message is displayed. What is it?**




**OPTIONAL: For up to 5 points of extra credit, get a shell on target.colab.duke.edu.**

The vulnerable account on the server is configured to print a secret bit of text and exit without providing a prompt. **Show how you can get an interactive shell using these credentials.**

ssh tunnel forwarding

### 11.2 Attack 4 / 4

✓ **+ 4 pts** **Correct**

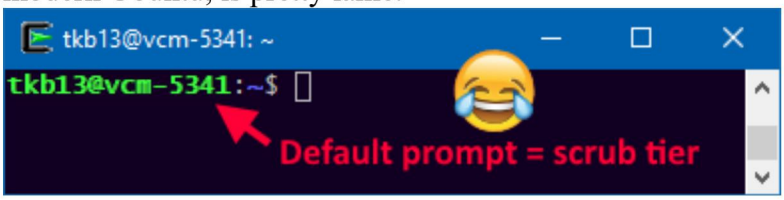
**+ 5 pts** Extra-Credit

**+ 0 pts** Wrong answer

I don't feel ssh tunnel forwarding works for our purpose. We want an interactive shell before the ssh tunnel close itself.

## Question 11: Craft your prompt (3 points)

A rite of passage of UNIX users is the customization of the bash prompt. The default, even on modern Ubuntu, is pretty lame.

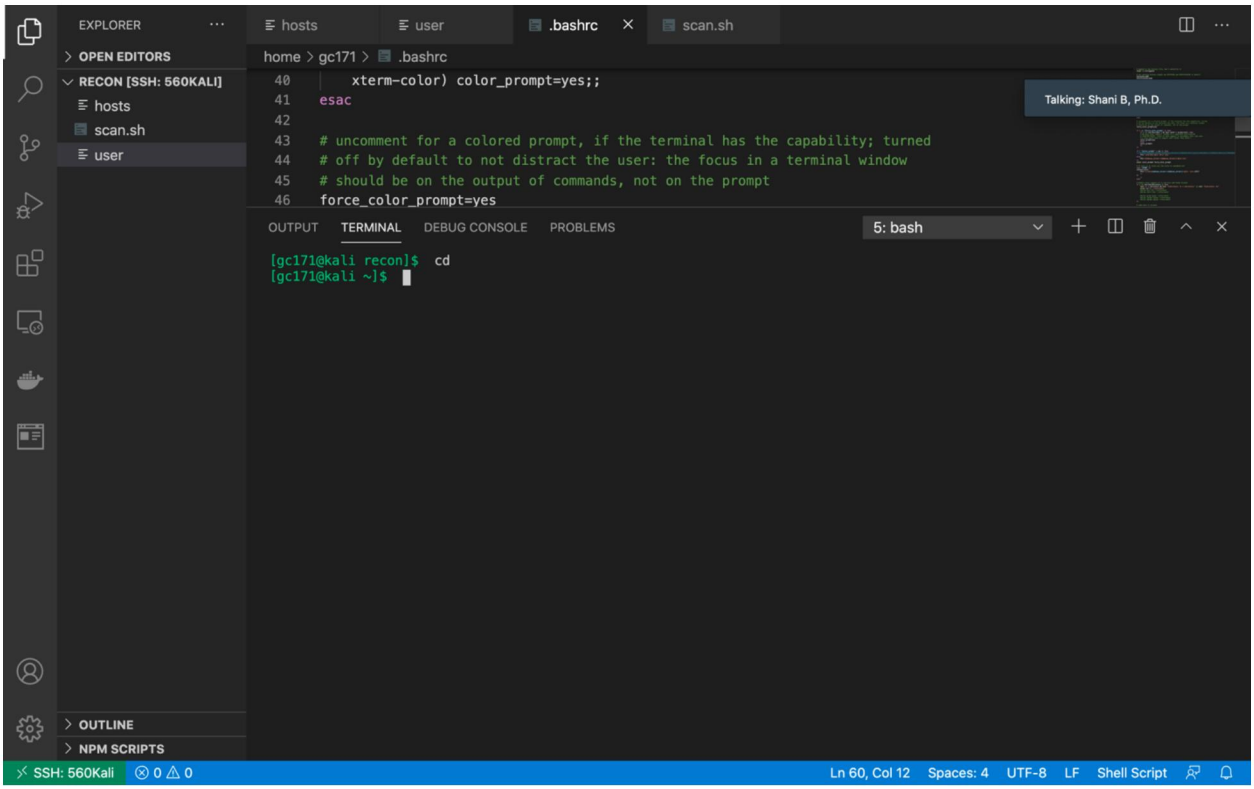



Using your understanding of shell color codes, develop a custom prompt by modifying the PS1 environment variable. You'll need to mark the escape codes for the shell using `\[` and `\]` indicators (this the shell needs to know what characters are printed vs. interpreted for cursor control purposes); details are here.

Edit the file `.bashrc` (sourced by the bash shell every time it is launched) in order to apply your prompt permanently.

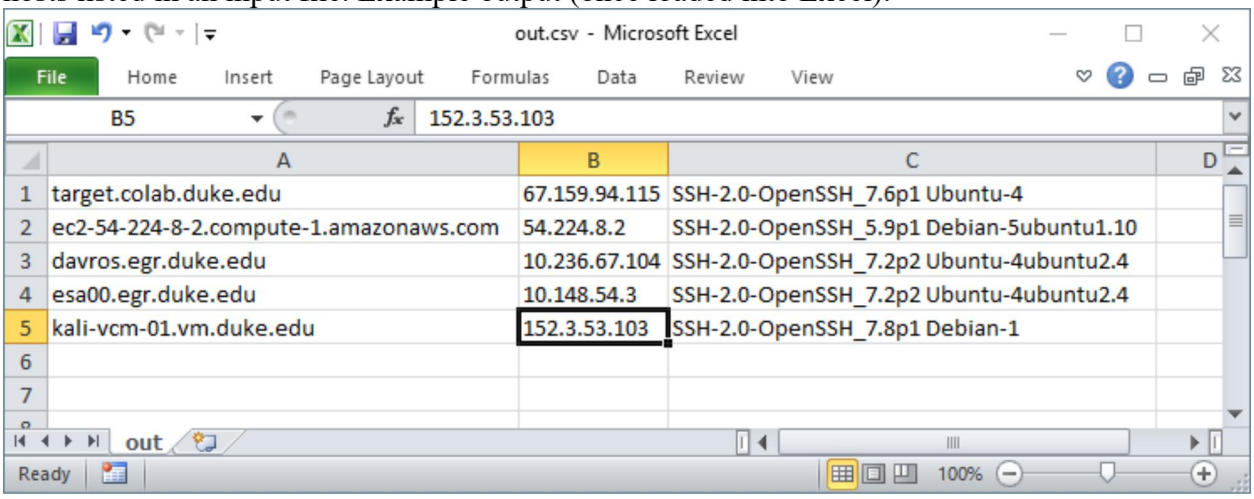
**Paste your PS1 environment variable command from `.bashrc` & a screenshot of the resulting prompt.**

```
PS1='\e[0;32m[\u@\h \W]\$ \e[m '
```




## Question 12: Attack recon (6 points)

One of the first things an attacker will do when focusing on a particular target organization is conduct reconnaissance. Using your choice of tools we've learned, develop a command or script that will produce a CSV file containing the hostname, IP address, and SSH version of all the hosts listed in an input file. Example output (once loaded into Excel):



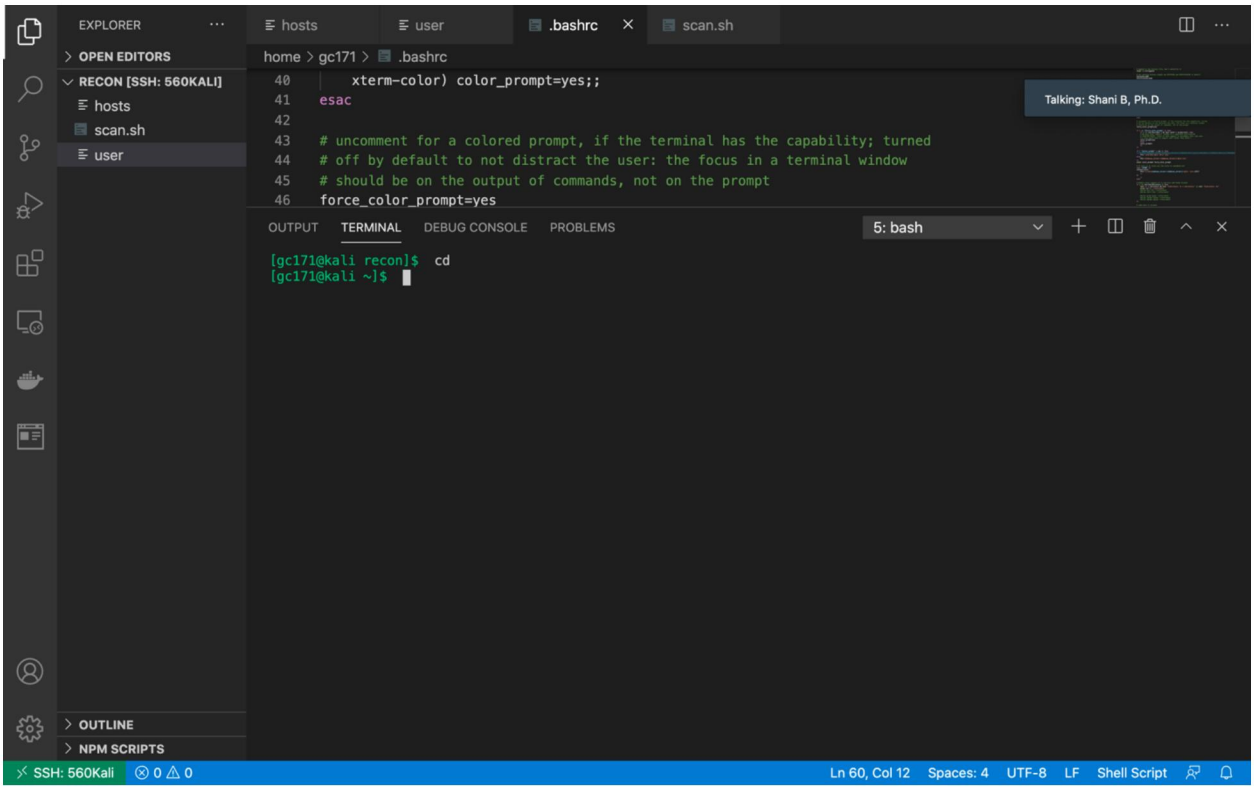

Apply this to the following hosts:

```
target.colab.duke.edu
96.30.196.58
139.180.157.201
```

## 12 Craft your prompt 3 / 3

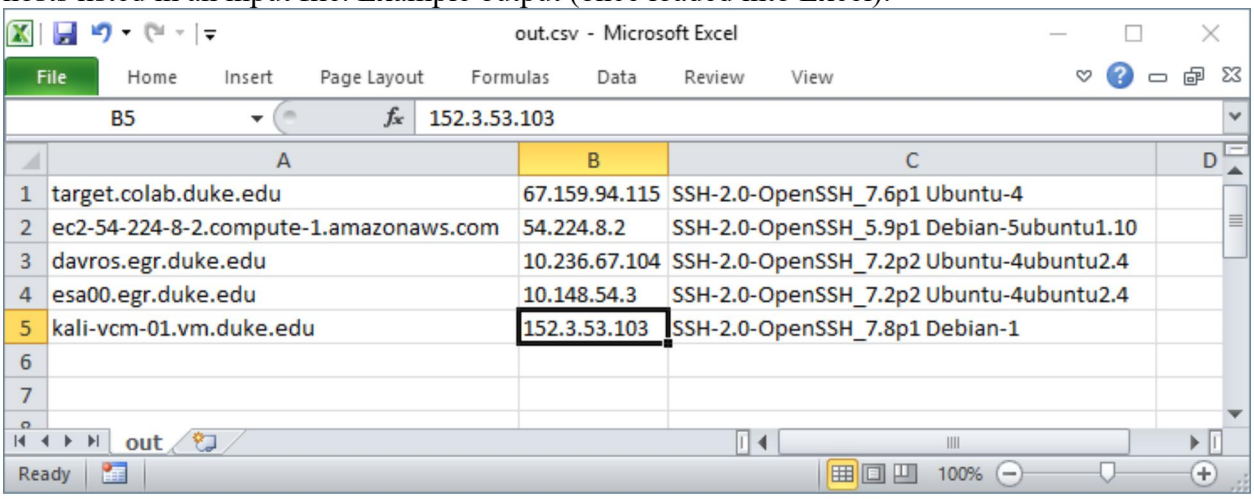
✓ **- 0 pts** **Correct**

**- 3 pts** Click here to replace this description.




## Question 12: Attack recon (6 points)

One of the first things an attacker will do when focusing on a particular target organization is conduct reconnaissance. Using your choice of tools we've learned, develop a command or script that will produce a CSV file containing the hostname, IP address, and SSH version of all the hosts listed in an input file. Example output (once loaded into Excel):




Apply this to the following hosts:

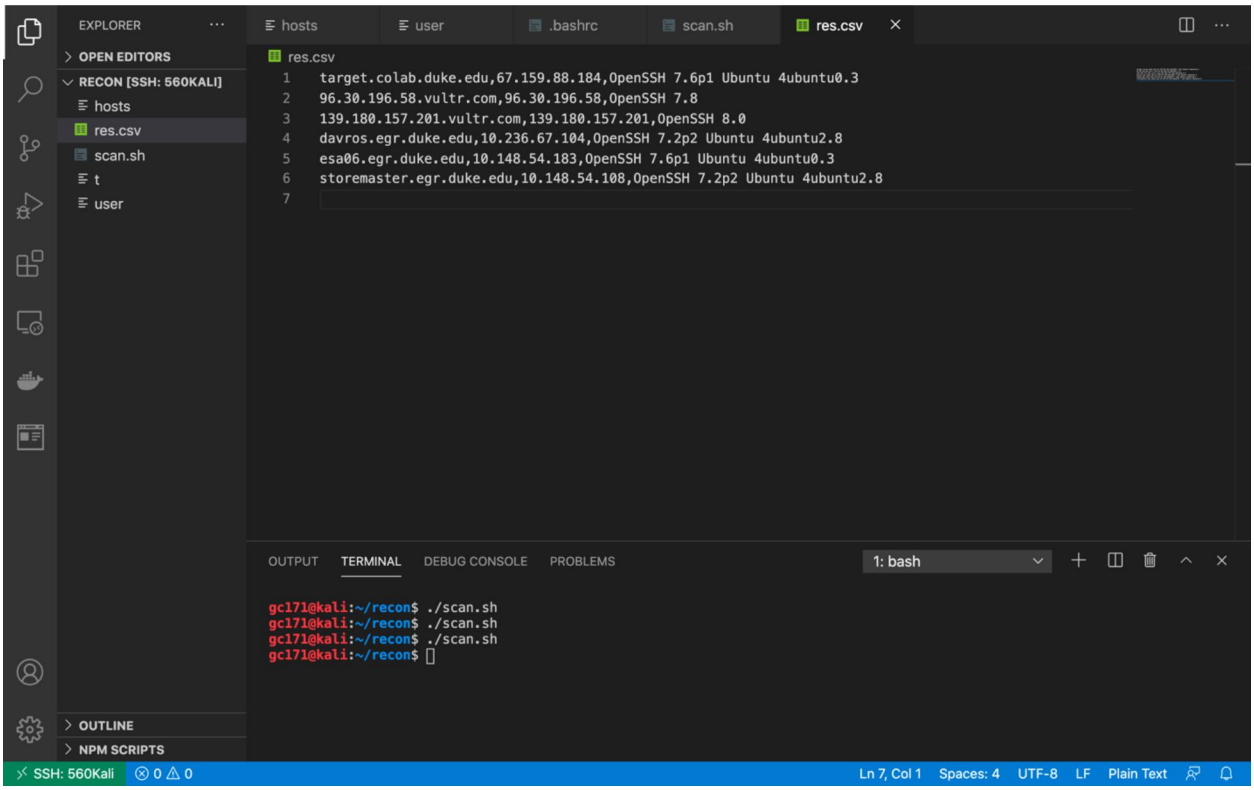
```
target.colab.duke.edu
96.30.196.58
139.180.157.201
```

```
davros.egr.duke.edu
esa06.egr.duke.edu
storemaster.egr.duke.edu
t-kali.colab.duke.edu
```

(Do not scan machines other than the above. All of the above are machines administered by me either at Duke or at a cloud hosting provider.)

**Give your script, code, or other artifacts inline in the answer PDF along with a screenshot of it in use. Your solution should be as small and simple as possible!**

```
input="hosts"
while IFS= read -r line
do
  nmap -sV -Pn "$line" | perl -ne '/Nmap.*for (.*) \((.*)\)/ and print "$1,$2,"' >>
res.csv
  nmap -sV -Pn "$line" | perl -ne '/22.*  ([\w .]+) \(/ and print "$1\n"' >> res.csv
done < "$input"
```




## Question 13: Keeping up with the news (10 points)

One absolutely essential attribute to the field of information security is that **it changes**. While most of the base theory we're talking about will remain true long into the future, attackers and

## 13 Attack Recon 6 / 6

✓ **- 0 pts** **Correct**

**- 1 pts** Correct solution but could have been shorter

defenders are locked into a continuous arms race of ever increasing sophistication. As such, a core skill for any security practitioner is to keep up to date with developments in the field.

Below are four IT security news sources:

- SANS Information Security News
- Security Week
- ThreatPost
- Reddit /r/netsec

Browse each, and pick out an article from the last 30 days related to one of the following topics: networking, cryptography, user authentication, malware, or denial of service attacks.

**From this article:**

- **Provide a brief summary, including a link to the article. (3)**
  **https://threatpost.com/boom-mobile-customer-data-fullz-house-magecart/159887/**

  service provider boom's website got injected some malicious code. the code will track the user input and stole credit info then sell it.

- **Identify which aspects of the CIA triad are at play and how. (2)**
  confidentiality: user credit card info is supposed to be well protected. hacker should not gain access to it.

  however, hacker successfully access to the info by injecting malicious code. it tracks user input and sell it.

- **Explain the threat model(s) at play (assets at risk, vulnerability at play, attacker's capabilities and knowledge). (2)**
  asset: user credit card info
  vulnerability: user info is in plaintext
  attacker capabilities: track the user input and grep the user info

- **If your article is explaining an attack, describe a defense that would have mitigated the attack and explain why it would be effective. Use the threat model in your analysis.**
  ***-or-***
  **If your article is explaining a defense, describe an attack that the defense would mitigate and explain why it would be effective. Use the threat model in your analysis. (3)**
  defense: rather than storing plaintext in client side, hash it in fly with salt & iteration counting. also ban js script from reading buffer.

  the asset we want to protect is user credit info. but attacker is capable of tracking the user input. therefore we should ban js from reading user input buffer.
  and once user input is done, hash it in fly. so even attacker can track it, they get a bunch of hash, which will take time to crack.

### 14.1 Article summary with link 3 / 3

✓ **- 0 pts** **Correct**

**- 3 pts** Click here to replace this description.

defenders are locked into a continuous arms race of ever increasing sophistication. As such, a core skill for any security practitioner is to keep up to date with developments in the field.

Below are four IT security news sources:

- SANS Information Security News
- Security Week
- ThreatPost
- Reddit /r/netsec

Browse each, and pick out an article from the last 30 days related to one of the following topics: networking, cryptography, user authentication, malware, or denial of service attacks.

**From this article:**

- **Provide a brief summary, including a link to the article. (3)**
  **https://threatpost.com/boom-mobile-customer-data-fullz-house-magecart/159887/**

  service provider boom's website got injected some malicious code. the code will track the user input and stole credit info then sell it.

- **Identify which aspects of the CIA triad are at play and how. (2)**
  confidentiality: user credit card info is supposed to be well protected. hacker should not gain access to it.

  however, hacker successfully access to the info by injecting malicious code. it tracks user input and sell it.

- **Explain the threat model(s) at play (assets at risk, vulnerability at play, attacker's capabilities and knowledge). (2)**
  asset: user credit card info
  vulnerability: user info is in plaintext
  attacker capabilities: track the user input and grep the user info

- **If your article is explaining an attack, describe a defense that would have mitigated the attack and explain why it would be effective. Use the threat model in your analysis.**
  ***-or-***
  **If your article is explaining a defense, describe an attack that the defense would mitigate and explain why it would be effective. Use the threat model in your analysis. (3)**
  defense: rather than storing plaintext in client side, hash it in fly with salt & iteration counting. also ban js script from reading buffer.

  the asset we want to protect is user credit info. but attacker is capable of tracking the user input. therefore we should ban js from reading user input buffer.
  and once user input is done, hash it in fly. so even attacker can track it, they get a bunch of hash, which will take time to crack.

## 14.2 Aspects of CIA triad 2 / 2

✓ **- 0 pts** **Correct**

**- 2 pts** Click here to replace this description.

defenders are locked into a continuous arms race of ever increasing sophistication. As such, a core skill for any security practitioner is to keep up to date with developments in the field.

Below are four IT security news sources:

- SANS Information Security News
- Security Week
- ThreatPost
- Reddit /r/netsec

Browse each, and pick out an article from the last 30 days related to one of the following topics: networking, cryptography, user authentication, malware, or denial of service attacks.

**From this article:**

- **Provide a brief summary, including a link to the article. (3)**
  **https://threatpost.com/boom-mobile-customer-data-fullz-house-magecart/159887/**

  service provider boom's website got injected some malicious code. the code will track the user input and stole credit info then sell it.

- **Identify which aspects of the CIA triad are at play and how. (2)**
  confidentiality: user credit card info is supposed to be well protected. hacker should not gain access to it.

  however, hacker successfully access to the info by injecting malicious code. it tracks user input and sell it.

- **Explain the threat model(s) at play (assets at risk, vulnerability at play, attacker's capabilities and knowledge). (2)**
  asset: user credit card info
  vulnerability: user info is in plaintext
  attacker capabilities: track the user input and grep the user info

- **If your article is explaining an attack, describe a defense that would have mitigated the attack and explain why it would be effective. Use the threat model in your analysis.**
  ***-or-***
  **If your article is explaining a defense, describe an attack that the defense would mitigate and explain why it would be effective. Use the threat model in your analysis. (3)**
  defense: rather than storing plaintext in client side, hash it in fly with salt & iteration counting. also ban js script from reading buffer.

  the asset we want to protect is user credit info. but attacker is capable of tracking the user input. therefore we should ban js from reading user input buffer.
  and once user input is done, hash it in fly. so even attacker can track it, they get a bunch of hash, which will take time to crack.

### 14.3 Thread models 2 / 2

✓ **- 0 pts** **Correct**

**- 2 pts** Click here to replace this description.

defenders are locked into a continuous arms race of ever increasing sophistication. As such, a core skill for any security practitioner is to keep up to date with developments in the field.

Below are four IT security news sources:

- SANS Information Security News
- Security Week
- ThreatPost
- Reddit /r/netsec

Browse each, and pick out an article from the last 30 days related to one of the following topics: networking, cryptography, user authentication, malware, or denial of service attacks.

**From this article:**

- **Provide a brief summary, including a link to the article. (3)**
  **https://threatpost.com/boom-mobile-customer-data-fullz-house-magecart/159887/**

  service provider boom's website got injected some malicious code. the code will track the user input and stole credit info then sell it.

- **Identify which aspects of the CIA triad are at play and how. (2)**
  confidentiality: user credit card info is supposed to be well protected. hacker should not gain access to it.

  however, hacker successfully access to the info by injecting malicious code. it tracks user input and sell it.

- **Explain the threat model(s) at play (assets at risk, vulnerability at play, attacker's capabilities and knowledge). (2)**
  asset: user credit card info
  vulnerability: user info is in plaintext
  attacker capabilities: track the user input and grep the user info

- **If your article is explaining an attack, describe a defense that would have mitigated the attack and explain why it would be effective. Use the threat model in your analysis.**
  ***-or-***
  **If your article is explaining a defense, describe an attack that the defense would mitigate and explain why it would be effective. Use the threat model in your analysis. (3)**
  defense: rather than storing plaintext in client side, hash it in fly with salt & iteration counting. also ban js script from reading buffer.

  the asset we want to protect is user credit info. but attacker is capable of tracking the user input. therefore we should ban js from reading user input buffer.
  and once user input is done, hash it in fly. so even attacker can track it, they get a bunch of hash, which will take time to crack.

### 14.4 Describe attack/defense 3 / 3

✓ **- 0 pts** **Correct**

**- 3 pts** Click here to replace this description.

## Question 14: Manipulating binary file formats (5 points)

Question 0 of this assignment involved detecting and decoding a polyglot PDF that was also a valid JPEG. **To receive credit for this question, the answers PDF you submit must also be a polyglot, but rather than a PDF+JPEG polyglot, it should be a PDF+ZIP polyglot. The ZIP aspect should contain (1) your public SSH key (the one submitted earlier to Encrypted Thing Giver in homework 2) and (2) a picture of a fat dog (<100kB).**

The construction of a PDF+ZIP polyglot is easier than you may guess because of the peculiar format of ZIP files. You may consult the ZIP file format article on Wikipedia, the napkin drawings by Julia Wolf in PoC||GTFO 01:05 ("This ZIP is also a PDF"), or this way-way-too-detailed presentation deck also by Julia Wolf.

## 15 Manipulating Binary File Formats 5 / 5

✓ **- 0 pts** **Correct**

**- 5 pts** Click here to replace this description.

16 Late Penalty 0 / 0

✓ - 0 pts Correct