Shjonathan Tan

21112183

s32tan

CS458 System Security Assignment 2

Written questions

Written Question 1

Qn 1a) Bell La Padula Confidentiality model has no read up, only read down access while write access is write up, no write down.

Formally, **read access** to file f is granted to user x if:

$$security\_level_x \geq security\_level_f \ \boldsymbol{AND}\ compartments_f \subseteq compartments_x$$

, **write access** file f is granted to user x if:

$$security\_level_x \leq security\_level_f \ \boldsymbol{AND}\ compartments_x \subseteq compartments_f$$

Terry can read readme but he cannot write down to readme

1b)

Terry cannot read or write Business Plan

1c)

Terry cannot read equipment registry and cannot write to equipment registry

1d)

terry cannot read draft and cannot write to draft

1e)

Terry can read software requirements specification

Terry can write to software requirements specification

2a) Biba Integrity Model can read up, no read down. Can write down, no write up.

Low watermark property is integrity of an obj is the lowest lvl of all objs tat contributed to its creation

Kelly cannot read file1 as she does not have the compartment “project plan”

2b)

Kelly is able to write to file2 as it has a lower integrity than Manager

2c) Kelly is unable to read file3 as she does not have comments compartment

2d)

Kelly is unable to write to director as she does not have the same integrity and compartments

2e)

Kelly is able to write to file5 as the integrity level is the same.

Written Question 2

- K-Company website owns IP range 16.35.125.0/24 (all ip addr btwn 16.35.125.0 and 16.35.125.255) uses a default deny policy
- All employees can browse http and https from within company network
- Kim work Remotely, ssh to work device, device IP address 16.35.125.25
- Malicious activity from IP address 85.63.28.0/24, block all incoming traffic from this range
- Webserver at 16.35.125.13, http and https , accessible from anywhere in the world
- K-Company trusts a DNS server (22.95.33.101) hosted by an allied organization to handle all its DNS lookups. DNS server listens for DNS requests on port 53 and also expects the clients to send these request from the ports in range 5000 to 5100. Assume DNS goes over UDP
- Rules
    - Drop or allow, Source ip addr, Dest ip addr, Source port, Dest port, Tcp or udp or both
    - For tcp a set of tcp flags must be set {}, SYN, ACK, or {SYN ,ACK}
    - Eg to allow access to HTTP pages from a server with IP addr 5.5.5.5
        - ALLOW 16.35.125.0/24 => 5.5.5.5 FROM PORT all TO 80 BY TCP
    - For multiple ports 80 and 443
        - ALLOW 5.5.5.5 => 16.35.125.0/24 FROM PORT all TO {80,443} BY TCP
    - For a range of ports from 80 to 443:
        - ALLOW 5.5.5.5 => 16.35.125.0/24 FROM PORT all TO [80,443] BY TCP

- To allow anyone to access the server at any port:
  - ALLOW 0.0.0.0/0 => 5.5.5.5 FROM PORT all to all BY TCP

1)

Default deny policy:

DROP * => * FROM PORT all to all

allows employee to browse http and https from within company network to outside internet

ALLOW 16.35.125.0/24 => * FROM PORT {80, 443} BY TCP {}

allows employee working remote to ssh connect to company network

ALLOW 16.35.125.25 => 16.35.125.0/24 FROM PORT {22} BY TCP { }

blocks all incoming traffic from sus ip addr to company network

DENY 85.63.28.0/24 => * FROM PORT all to all BY TCP

Webserver accessible from anywhere for http and https:

ALLOW 0.0.0.0/0 => 16.35.125.13 FROM PORT {80, 443} BY TCP {}

allows clients to connect to dns server

ALLOW * => 22.95.33.101 FROM PORT all TO [5000, 5100] BY UDP

2)

this is ip address spoofing where the headers are using the company's ip address as their source address. To stop this, we can use ingress filtering on the network and firewalls.

Ingress filtering involves establishing an access control list that contains IP addresses of permitted source addresses. It uses data link layer IP address filtering capability of a router and blocks traffic that is most likely to be malicious. It checks the IP packet header and checks if the source IP address is invalid or if it does not match the originating network, the filter will determine that this address is fake and drop the packet.

3) Exposing a web server to the public and keeping network secure, we implement a DMZ. A DMZ is a separate network segment that is isolated from the internet network and acts as a buffer zone between public internet and internal network.

This allows the company to provide access to specific services like a web server without exposing the internal networks to potential attacks.

1. Create a DMZ: set up a separate physical or logical network segment within your organization's network infrastructure. The DMZ should be isolated from internal company network and only accessible from the public internet and internal network through the firewall
2. Position the web server in the DMZ to act as an intermediate zone between the public internet and internal company network
3. Configure firewall rules to allow incoming traffic from public internet to the web server. Use port 80 and 443 as well as port forwarding or destination NAT rules to direct traffic to the public web server. For outgoing traffic, restrict traffic from DMZ to internal network and limit services that the web server in the DMZ can access in the internal network.
4. Implement strict access control policies and firewall rules between dmz, public internt and internal network.

Written Question 3)

1)

It is vulnerable to brute force attack as the password length is only 8 letters long.

The shared office space may increase risk of unauthorized access to the file containing hashed passwords.

The use of md5 hashed passwords accessible to untrusted individuals can allow a rainbow table attack which will allow untrusted individuals.

2)

No. The updated file is still accessible to untrusted individuals and the usage of md5 hashing is still prone to rainbow table attack using files such as the released RockYou2021 password list which has 8.4 billion password entries.

Also terry's passwords are

25d55ad283aa400af464c76d713c07ad -> 12345678

25f9e794323b453885f5181f1b624d0b -> 123456789

e807f1fcf82d132f9bb018ca6738a19f -> 1234567890

terry only updates by 1 letter each time which can be easily brute forced if we are doing rainbow attack

3)

K-Software can include a salt to the hashing, this will cause users to have same password to have different md5 hash configurations due to the unique salt and this will cause rainbow attacks to become ineffective.

K-software can also use stronger and more secure cryptographic hashing algorithms such as bcrypt or Argon2. They include salting passwords as well as adaptive hashing. They are computationally intensive and makes it much harder and slower for attackers to crack hashed passwords through brute force or dictionary attacks. The salting used is a unique salt which adds randomness to the hashing process. Thus rendering rainbow tables much harder.

4)

Using 2-factor authentication is useful only if the 2nd authentication is of a different class as passwords.

Using an additional pin is the same class as user password, which means if you can break the first password, you probably can break the additional pin with brute force attack. However it is easy to use and has low cost.

Using a security key generated by an authenticator phone app is better as it is a different class as user password. This is something the user has which is a smart phone app, which will produce a random code every few minutes and will allow the code to be invalid after a few minutes. this means the attacker is unable to guess the random code within that few minutes. this allows it to be more secure. However it is more troublesome to setup the authenticator app as it is more complex.

It is more secure to use the phone mobile application together with the password thus it is more effective authentication than using the password and additional pin.

5)

MD5 is not useful because it is not secure because it is susceptible to collisions attacks where 2 different inputs can produce the same MD5 hash which allows attackers to forge data and gain access into the system. MD5 also does not have salting, thus it is prone to rainbow table or brute force attack. MD5 also does not have many iterations as it was designed to do hashing fast.

Alternative uses of hash function include

Bcrypt – computationally intensive and makes brute force attacks much slower

Argon2 – designed to be secure against attacks such as brute force and side channel attacks

They both have built in mechanisms for adding salts to password as well as having multiple iterations which slows down hashing which means attackers take a much longer time for brute force or rainbow table attacks

Programming questions

1a)

Using inspect tool and network tab, we copy paste the post request used to login on the browser. We get “ curl 'http://ugster504.student.cs.uwaterloo.ca/s32tan/post.php' -X POST -H 'Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8' -H 'Referer: http://ugster504.student.cs.uwaterloo.ca/s32tan/login.php' -H 'Cookie: CS458=vc7ubq6ml1bk6ssg007cbkhq05' -H 'Upgrade-Insecure-Requests: 1' --data-raw 'username=alice&password=MySecretPassword&form=login' “

We replicate this curl command

Exploit.sh has a command curl -c ‘cookieinfo.txt’ –compressed -X POST --data-raw 'username=alice&password=MySecretPassword&form=login'

-c saves cookie information into a file called cookieinfo.txt

--compressed Request a compressed response using one of the algorithms curl supports, and automatically decompress the content. Response headers are not modified when saved, so if they are "interpreted" separately again at a later point they might appear to be saying that the content is (still) compressed; while in fact it has already been decompressed.

-X Change the method to use when starting the transfer.

-I, --head (HTTP FTP FILE) Fetch the headers only!

We make a post using

curl -b cookieinfo.txt "$URL/post.php" --compressed -X POST --data-raw "title=$TITLE&content=sample&type=1&form=content&submit=";

we make a comment using

curl -b cookieinfo.txt "$URL/post.php" --compressed -X POST --data-raw "comment=firstcomment&form=comment&parent=$ID&uid=7&submit=' --referer 'http://ugster504.student.cs.uwaterloo.ca/s32tan/view.php?id=$ID";

we upvote the post created

curl -b cookieinfo.txt "$URL/vote.php?id=$ID&vote=1" --compressed -X GET --referer "$URL/index.php";

2) We guess the password is MountainGoat because of the various posts relating to mountain goats
3) Examining the post about the confirmation url, we are given the clue that it is a URL % encoding at the end of the url, and we find out there is a binary to text encoding of the username. We use this to find out thetravellers confirmation code.
4) We go into the login page and use this input pvuser' OR '1'='1 which overwrites the password check in the sql code, allowing us to enter as the user as it returns true for both
5) We find a post relating to robots.txt so we enter the url/robots.txt which gives us a clue to go to url/docs and then we download the database. From the database we find out the passwords are hashed in SHA1. We also examine which tables have users and user relations.