SIM Global Education

UOW AUSTRALIA

CSCI262 – System Security

Tutorial Set A – User Authentication

24 October 2023

Question 1a

Is there any harm in revealing old passwords? Why or why not?

Question 1a

Is there any harm in revealing old passwords? Why or why not?

Yes, there might be harm. There could possibly being a pattern or theme common to the passwords that you choose, making it possibly to determine more likely what passwords are currently being used. For example, if my old password is m00dleP@ssword2016, then an attacker might guess that my current password might be m00dleP@ssword2017. In addition, with the old password revealed, and attacker may use information from the password to perform a social engineering attack.

Question 1b

What is the entropy associated with a password chosen with uniform randomness from the set of length 8 strings with symbols taken from the lowercase alphabet {a,…,z}?

Question 1b

What is the entropy associated with a password chosen with uniform randomness from the set of length 8 strings with symbols taken from the lowercase alphabet {a,…,z}?

The entropy for L number of symbols taken from N possible symbols is given by the formula:

$$\begin{aligned} Entropy &= log_2 N^L \\ &= L\, log_2 N \end{aligned}$$

Question 1b

What is the entropy associated with a password chosen with uniform randomness from the set of length 8 strings with symbols taken from the lowercase alphabet {a,…,z}?

The entropy for L number of symbols taken from N possible symbols is given by the formula:

$$\begin{aligned} Entropy &= \log_2 N^L \\ &= L \log_2 N \end{aligned}$$

Thus, for $L = 8$ characters and $N = 26$ lowercase alphabets, the entropy is

$$\begin{aligned} Entropy &= L \log_2 N \\ &= 8 \times \frac{\log_{10} 26}{\log_{10} 2} \\ &\approx 37.60 \end{aligned}$$

Question 1b

What is the entropy associated with a password chosen with uniform randomness from the set of length 8 strings with symbols taken from the lowercase alphabet {a,…,z}?

The entropy for L number of symbols taken from N possible symbols is given by the formula:

$$\begin{aligned} Entropy &= \log_2 N^L \\ &= L \log_2 N \end{aligned}$$

Thus, for $L = 8$ characters and $N = 26$ lowercase alphabets, the entropy is

$$\begin{aligned} Entropy &= L \log_2 N \\ &= 8 \times \frac{\log_{10} 26}{\log_{10} 2} \\ &\approx 37.60 \end{aligned}$$

How strong is the password?

Question 1b

What is the entropy associated with a password chosen with uniform randomness from the set of length 8 strings with symbols taken from the lowercase alphabet {a,…,z}?

The entropy for L number of symbols taken from N possible symbols is given by the formula:

$$\begin{aligned} Entropy &= \log_2 N^L \\ &= L \log_2 N \end{aligned}$$

Thus, for $L = 8$ characters and $N = 26$ lowercase alphabets, the entropy is

$$\begin{aligned} Entropy &= L \log_2 N \\ &= 8 \times \frac{\log_{10} 26}{\log_{10} 2} \\ &\approx 37.60 \end{aligned}$$

How strong is the password?

$$2^{37.6} \approx 2.1 \times 10^{11}$$

Question 1c

How much additional entropy is added by the inclusion of a uniformally random digit on the end of that password?

Question 1c

How much additional entropy is added by the inclusion of a uniformally random digit on the end of that password?

By adding a uniformally random digit on the end of that password, an additional of $\log_2 10 = \frac{\log_{10} 10}{\log_{10} 2}$ = 3.32 bits of information will be added to the original entropy; that is, 37.6 + 3.32 = 40.92. This is because a uniformally random digit can be any one of the ten (10) possible digits.

With an entropy of 40.92 bits, the strength of the password equals $2^{40.92} \approx 2.08 \times 10^{12}$.

Question 1c

Alternatively, we can work out the total number of possible passwords and calculate the entropy for that total space. For example:

- 8 lowercase characters → $26^8 = 2.088 \times 10^{11}$ possible combinations
- 1 random digit → $10^1 = 10$ possible combinations
- Thus, in total there are $2.088 \times 10^{11} \times 10 = 2.088 \times 10^{12}$ possible combinations

Question 1c

$With\ 2.088 \times 10^{12}\ possible\ combinations, the\ entropy\ of\ the\ password\ is$:

$$
\begin{aligned}
Entropy &= L \log_2 N \\
&= 1 \log_2 2.088\ \times 10^{12} \\
&= \frac{log_{10} 2.088\ \times 10^{12}}{log_{10} 2} \\
&= 40.92
\end{aligned}
$$

Why L is 1?

Basically there is only 1 (one) set of password with 2.088 x 10^{12} possible combinations.

Question 1d

How much entropy is there associated with a typical ATM Pin?

Question 1d

How much entropy is there associated with a typical ATM Pin?

In Singapore, the size of a typical ATM Pin is 6 digits long. Hence the entropy is:

$$
\begin{aligned}
Entropy &= L\, Log_2\, N \\
&= 6 \times \frac{log_{10} 10}{log_{10} 2} \\
&\approx 19.93 \text{ bits}
\end{aligned}
$$

…and the strength of a 6-digit PIN is about $2^{19.93} = 998{,}913$.

Question 1e

Is fDtk53$e3W22eSDmvfFp-4F a good password? Why or why not?

Question 1e

Is fDtk53$e3W22eSDmvfFp-4F a good password? Why or why not?

It is a ‘yes’ and a ‘no’. The password described above consists of 23 characters long drawn from a possible of 92 symbols (10 digits, 26 lowercase alphabets, 26 uppercase alphabets, and 30 symbols), we have:

$$\begin{aligned} Entropy &= L \, log_2 N \\ &= 23 \times \frac{log_{10} 92}{log_{10} 2} \\ &= 150.04 \, bits \end{aligned}$$

Question 1e

150.04 bits of information forms 1.47 x 10^{45} combinations. With a typical computer that can find 1000 combination in a second, for example, a brute-form attempt for 1.47 x 10^{45} combinations would need

$$\frac{(1.47 \times 10^{45})}{1000} = 1.47 \times 10^{42}\ seconds$$

$$\frac{1.47 \times 10^{45}}{3600} = 4.08 \times 10^{38}\ hours$$

$$\frac{4.08 \times 10^{38}}{24} = 1.7 \times 10^{37}\ days$$

$$\frac{1.7 \times 10^{37}}{365} = 4.7 \times 10^{34}\ years$$

Question 1e

From the entropy points of view, the password is very strong, and hence ‘yes’ it is a good password. Unfortunately, this is not usually the case in real-life because a randomly generated 23-symbol long password is hard to remember. People tend to generate password with some template or pattern, hence ‘no’ this password is too difficult to remember, and people would unlikely use it.

Question 1f

- Without writing down your password, or the method of choosing your password, estimate the entropy associated with the password you use most.

Question 1f

- Without writing down your password, or the method of choosing your password, estimate the entropy associated with the password you use most.

For a typical 12 characters long password drawn from a possible of 62 symbols (10 digits, 26 lowercase alphabets, and 26 uppercase alphabets), for example, we have:

$$\begin{aligned} Entropy &= L \log_2 N \\ &= 12 \times \frac{\log_{10} 62}{\log_{10} 2} \\ &\approx 71.45 \text{ bits} \end{aligned}$$

Question 1f

- Without writing down your password, or the method of choosing your password, estimate the entropy associated with the password you use most.

For a typical 12 characters long password drawn from a possible of 62 symbols (10 digits, 26 lowercase alphabets, and 26 uppercase alphabets), for example, we have:

$$\begin{aligned} Entropy &= L \log_2 N \\ &= 12 \times \frac{\log_{10} 62}{\log_{10} 2} \\ &\approx 71.45 \text{ bits} \end{aligned}$$

How strong is the password?

Question 1f

- Without writing down your password, or the method of choosing your password, estimate the entropy associated with the password you use most.

For a typical 12 characters long password drawn from a possible of 62 symbols (10 digits, 26 lowercase alphabets, and 26 uppercase alphabets), for example, we have:

$$Entropy = L\ log_2\ N$$
$$= 12 \times \frac{log_{10}\ 62}{log_{10}\ 2}$$
$$\approx 71.45 \text{ bits}$$

How strong is the password?

$$2^{71.45} \approx 3.2 \times 10^{21}$$

Question 1f

- The entropy calculated for the example mentioned in the previous slides is based on the assumption that the 12-symbol long password is **randomly** generated from the 62 symbols.

- As discussed in Question 1f, unfortunately, this is not usually the case in real-life because a randomly generated 12-symbol long password is hard to remember. Assuming my password has the following pattern ☺ (Of course I would not tell you the truth about how I construct my password ha..ha..)

Question 1f

For example, if I generate my password using a template with the following rules:

- Always start with 2 digits, follow with
- 3 lowercase alphabet, follow with
- 1 uppercase alphabet, and follow with
- A fixed 6-letter code ‘UOWSIM’

An example of my password is 09jctXUOWSIM.

In this case, although the length of my password remain as 12, the entropy of my password would be lower and is calculated as follow:

Question 1f

- 2 digits → 10^2 = 100 possible combinations
- 3 lowercase letters → 26^3 = 17576 possible combinations
- 1 uppercase letter → 26 possible combinations
- A fixed 'UOWSIM' code → 1 possible combinations
- Thus, in total there are 100 x 17576 x 26 x 1 = 45697600 possible combinations

$$Entropy = L\ log_2\ N$$
$$= 1 \times \frac{log_{10}\ 45697600}{log_{10}\ 2}$$
$$\approx 25.4456\ bits$$

Question 1f

- So, does it make much different for a password strength of $2^{71.45}$ = 3.2 x 10^{21} (first example of Question 1h) and $2^{25.4456}$ = 4,558,096 (second example of Question 1h)?

Question 1f

- So, does it make much different for a password strength of $2^{71.45}$ = 3.2 x 10^{21} (first example of Question 1h) and $2^{22.12}$ = 4,558,096 (second example of Question 1h)?

- Yes, very much different. If an attacker knows the way I construct my password, with a computer that can find 1000 combination in a second, for example, a brute-force attempt for 4,558,096 combination can be done in 1.26 hours.

- For a randomly generated password, the attacker would need about 1.02 x 10^{11} years.

Question 1g

- How much confidence do you have in the method of choosing your password not being guessed?

Question 1g

- How much confidence do you have in the method of choosing your password not being guessed?

Please refer to the first example given in Question 1h and discussion in previous slide.

Question 1h

- How much confidence do you have in your password under the assumption the method of choosing your password was known by an attacker?

Question 1h

- How much confidence do you have in your password under the assumption the method of choosing your password was known by an attacker?

Please refer to the example of generating a password if a template is used shown in previous question.

Question 2

Does taking *H(M)*, for H a cryptographic hash function, provide confidentiality for *M*?

Question 2

Does taking *H(M)*, for H a cryptographic hash function, provide confidentiality for *M*?

No, $H(M)$ does not provide confidentiality for M. Why?

Reason being:

i. Cryptography hash function must satisfy the following properties:
   - One-way or pre-image resistant - **It is computationally infeasible that for a given message digest Y, we can find an X such that $\boldsymbol{H(X) = Y}$.**
   - Collision-resistant

Question 2

- If cryptographic hash function is one-way, then it means we cannot get back the original message from the digest, and hence cannot meet the requirement for confidentiality.

- In cryptography, confidentiality is the concept of ensuring that data is not made available or disclosed to unauthorized people. A cryptographic hash function achieves the first part of confidentiality, that is, ensuring that data is not readable by unauthorized people, but a cryptographic hash function cannot meet the second requirement of confidentiality, that is made the data available to its intended recipient.

Question 3

Hashing “produces a fingerprint” of a message. In what way does this misrepresent the relationship between hash and message, relative to the relationship between human fingerprint and human?

Question 3

Hashing “produces a fingerprint” of a message. In what way does this misrepresent the relationship between hash and message, relative to the relationship between human fingerprint and human?

The uniqueness between hash and message may not be 100% correct; in other words, there is certain uncertainty involve due to possibility of a collision.

Question 4…1/4

2) Describe in detail how the one-time password system of Lamport works.

Question 4…1/4

2) Describe in detail how the one-time password system of Lamport works.

One-time password refers to a password that can be used only for one session or one transaction. Lamport’s one-time password is one example of such password. Lamport’s one-time password consists of two parts, the setup and the process as follows:

Setup:

- In the setup process, a user is selecting a password that is secret to him/her.
- The system will then use this password, together with some value, say n, generate a sequence of passwords $p_1, p_2, \dots p_n$.

Question 4 ...2/4

Process:

- A user, let’s say Alice, request for connection to a server.
- The server issues a challenge n;
- The user responds with one-time password which is generated as $h^{n-1}(password)$
- The server checks if $h\left(h^{n-1}(password)\right) = h^{n}(password)$
- If it matches, then server accepts the communication request. If it does not, the server rejects the communication request.
- Once the user has been authenticated, the server needs to update its information.

Question 4 ...3/4

Process: (cont…)

- The system will then replace $x_n = h^n(password)$ with the one-time password sent by the user's, that is, $x_{n-1} = h^{n-1}(password)$.
- The value n is replaced by $n - 1$.
- When n reaches 0, the system will have run out of passwords in the hash chain and will have to run a new setup process, with a new base password.

Question 4 ...4/4

- Lamport's one-time password works because the system define p_i to be $H^{n-1}(p)$ where H is a hash function known to all, e.g., MD5() in our Assignment 1. In this way, attacker cannot derive future password from a past password. For example, after p_6, which is equals $H^{n-6}(p)$, the attacker can compute $H(p_6)$, which equals $H^{n-5}(p)$, the already used password p_5. The attacker cannot compute p_7 because p_7 equals $H^{n-7}(p)$, and computing $H^7(p)$ from $H^6(p)$ would require the attacker to compute the inverse of H or to know p, but H is a cryptographic hash function.

Question 4 ...4/4

For example:

- Secret = s
- $H^5\left(H^4\left(H^3\left(H^2\left(H^1(s)\right)\right)\right)\right)$
- $H^1(s) = pw1$
- $H^2\left(H^1(s)\right) = pw2$
- $H^3\left(H^2\left(H^1(s)\right)\right) = pw3$
- $H^4\left(H^3\left(H^2\left(H^1(s)\right)\right)\right) = pw4$
- $H^5\left(H^4\left(H^3\left(H^2\left(H^1(s)\right)\right)\right)\right) = pw5$

Question 4 ...4/4

Client request for connection	
	Server send challenge
Client respond according to challenge	Server verifies that the challenge is correct.
	If challenge is correct, the connection is established.

n = 5 (Number of one-time passwords)

Challenge (n-c)	Counter (c)	Respond $H^{n-c}(S) = pw_{n-i}$	Last used pw (lpw)	Verification $H\left(H^{n-c}(s)\right) = lpw$
4	1	$H^4(S) = pw4$	$pw5$	$H\left(H^4(S)\right) = pw5$

Question 4 ...4/4

Client request for connection	
	Server send challenge
Client respond according to challenge	Server verifies that the challenge is correct.
	If challenge is correct, the connection is established.

n = 5 (Number of one-time passwords)

Challenge (n-c)	Counter (c)	Respond $H^{n-c}(S) = pw_{n-i}$	Last used pw (lpw)	Verification $H\left(H^{n-c}(s)\right) = lpw$
4	1	$H^4(S) = pw4$	$pw5$	$H\left(H^4(S)\right) = pw5$
3	2	$H^3(S) = pw3$	$pw4$	$H\left(H^3(S)\right) = pw4$

Question 4 ...4/4

Client request for connection	
	Server send challenge
Client respond according to challenge	Server verifies that the challenge is correct.
	If challenge is correct, the connection is established.

n = 5 (Number of one-time passwords)

Challenge (n-c)	Counter (c)	Respond $H^{n-c}(S) = pw_{n-i}$	Last used pw (lpw)	Verification $H\left(H^{n-c}(s)\right) = lpw$
4	1	$H^4(S) = pw4$	$pw5$	$H\left(H^4(S)\right) = pw5$
3	2	$H^3(S) = pw3$	$pw4$	$H\left(H^3(S)\right) = pw4$
2	3	$H^2(S) = pw2$	$pw3$	$H\left(H^2(S)\right) = pw3$

Question 4 ...4/4

Client request for connection	
	Server send challenge
Client respond according to challenge	Server verifies that the challenge is correct.
	If challenge is correct, the connection is established.

n = 5 (Number of one-time passwords)

Challenge (n-c)	Counter (c)	Respond $H^{n-c}(S) = pw_{n-i}$	Last used pw (lpw)	Verification $H(H^{n-c}(s)) = lpw$
4	1	$H^4(S) = pw4$	$pw5$	$H(H^4(S)) = pw5$
3	2	$H^3(S) = pw3$	$pw4$	$H(H^3(S)) = pw4$
2	3	$H^2(S) = pw2$	$pw3$	$H(H^2(S)) = pw3$
1	4	$H^1(S) = pw1$	$pw2$	$H(H^1(S)) = pw2$

Question 4 ...4/4

n = 5 (Number of one-time passwords)

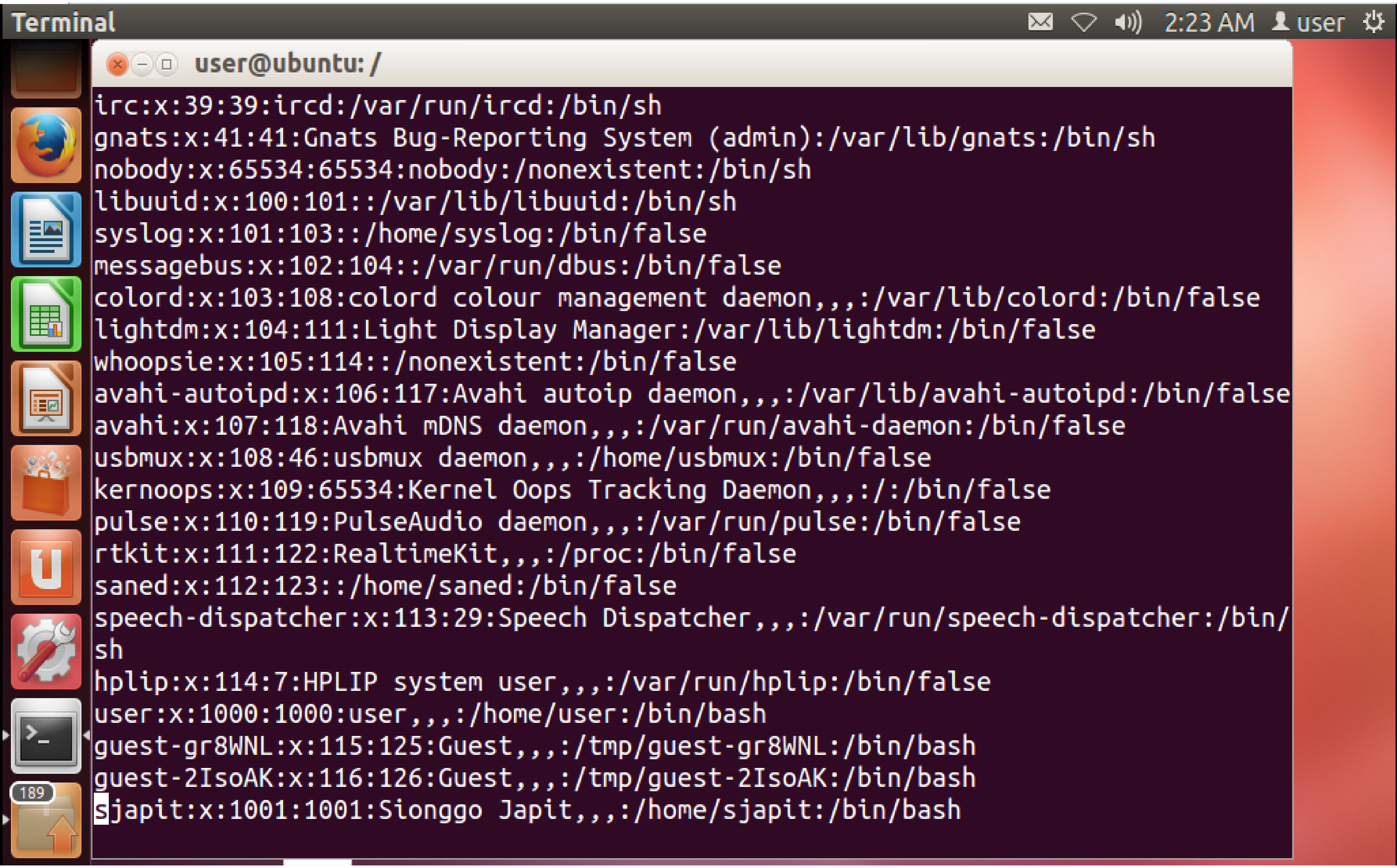
Challenge (n-c)	Counter (c)	Respond $H^{n-c}(S) = pw_{n-i}$	Last used pw (lpw)	Verification $H\left(H^{n-c}(s)\right) = lpw$
4	1	$H^4(S) = pw4$	$pw5$	$H\left(H^4(S)\right) = pw5$
3	2	$H^3(S) = pw3$	$pw4$	$H\left(H^3(S)\right) = pw4$
2	3	$H^2(S) = pw2$	$pw3$	$H\left(H^2(S)\right) = pw3$
1	4	$H^1(S) = pw1$	$pw2$	$H\left(H^1(S)\right) = pw2$
0	5		$pw1$	

Question 5

- Find where passwd and shadow are located. (This is in the lecture notes!)

a) Look inside them. Find your own entry. Identify your user identification number and your native group number.

b) How large are the passwd and shadow files?



```
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
syslog:x:101:103::/home/syslog:/bin/false
messagebus:x:102:104::/var/run/dbus:/bin/false
colord:x:103:108:colord colour management daemon,,,:/var/lib/colord:/bin/false
lightdm:x:104:111:Light Display Manager:/var/lib/lightdm:/bin/false
whoopsie:x:105:114::/nonexistent:/bin/false
avahi-autoipd:x:106:117:Avahi autoip daemon,,,:/var/lib/avahi-autoipd:/bin/false
avahi:x:107:118:Avahi mDNS daemon,,,:/var/run/avahi-daemon:/bin/false
usbmux:x:108:46:usbmux daemon,,,:/home/usbmux:/bin/false
kernoops:x:109:65534:Kernel Oops Tracking Daemon,,,:/:/bin/false
pulse:x:110:119:PulseAudio daemon,,,:/var/run/pulse:/bin/false
rtkit:x:111:122:RealtimeKit,,,:/proc:/bin/false
saned:x:112:123::/home/saned:/bin/false
speech-dispatcher:x:113:29:Speech Dispatcher,,,:/var/run/speech-dispatcher:/bin/
sh
hplip:x:114:7:HPLIP system user,,,:/var/run/hplip:/bin/false
user:x:1000:1000:user,,,:/home/user:/bin/bash
guest-gr8WNL:x:115:125:Guest,,,:/tmp/guest-gr8WNL:/bin/bash
guest-2IsoAK:x:116:126:Guest,,,:/tmp/guest-2IsoAK:/bin/bash
sjapit:x:1001:1001:Sionggo Japit,,,:/home/sjapit:/bin/bash
```

/etc/passwd …1

Format of passswd (/etc/passwd)

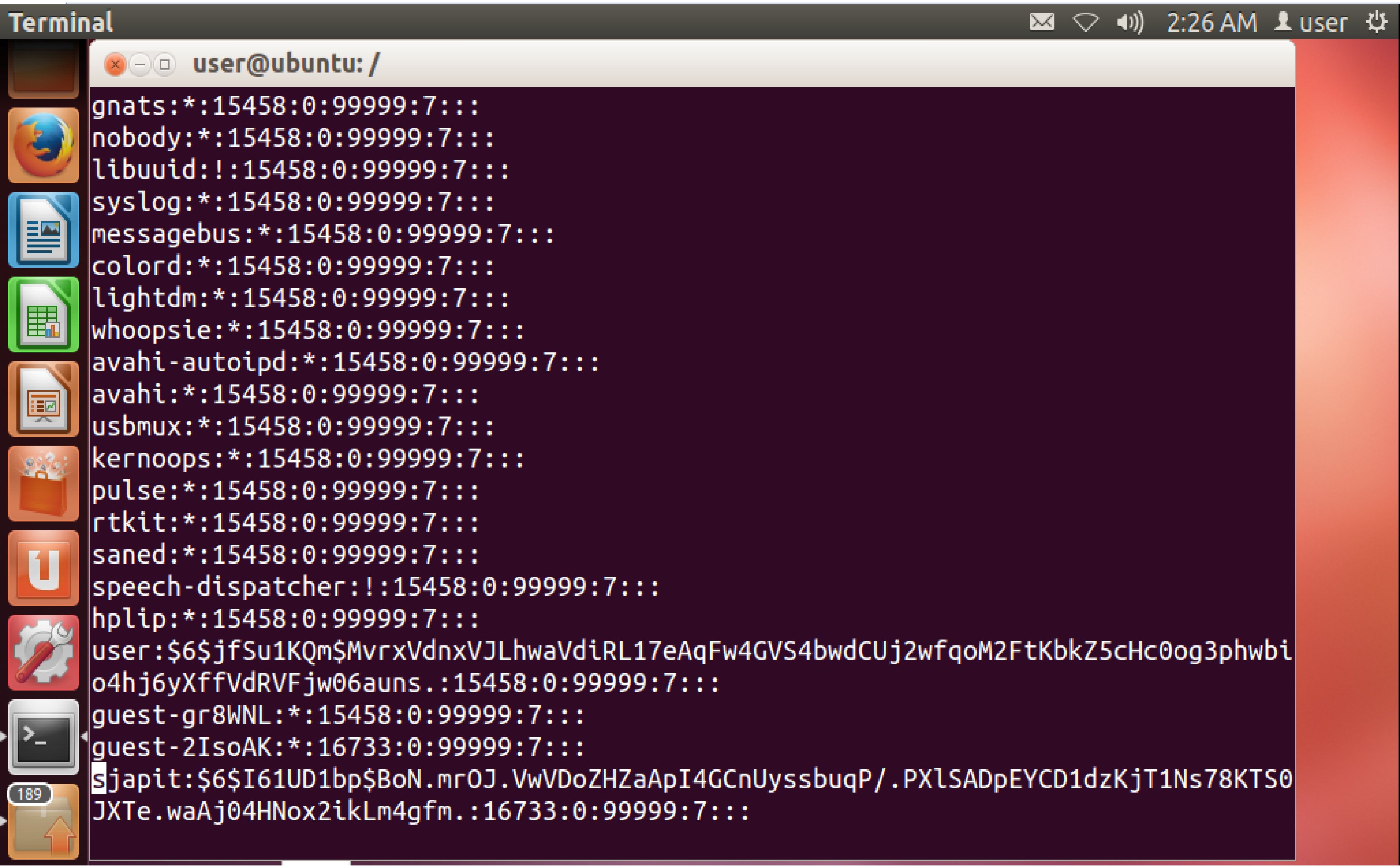
Username:x:NumericUserid:NumericGroupid:FullName:HomeDirectoryOfUser:User’sShellAccount

- **Username**: Up to 8 characters. Case-sensitive
- An ‘**x**’ indicate passwords are stored in the shadow file at /etc/shadow
- **Numeric userid**: userid to identify a user. UID 0 is reserved for root and UID 1 – 99 are reserved for other predefined accounts. Further UID 100 – 999 are reserved by system for administrative and system accounts/groups.

/etc/passwd …2

Username:x:NumericUserid:NumericGroupid:FullName:HomeDirectoryOfUser:User’sShellAccount

- **Numeric group id**: The primary group id, stored in /etc/group file
- **FullName**: This is a comment field. It allows additional information about users to be added here.
- **Home director**: The absolute path to the directory the user will be in when they log in.
- **User’s Shell account**: the absolute path of a command or shell (/bin/bash).



```
gnats:*:15458:0:99999:7:::
nobody:*:15458:0:99999:7:::
libuuid:!:15458:0:99999:7:::
syslog:*:15458:0:99999:7:::
messagebus:*:15458:0:99999:7:::
colord:*:15458:0:99999:7:::
lightdm:*:15458:0:99999:7:::
whoopsie:*:15458:0:99999:7:::
avahi-autoipd:*:15458:0:99999:7:::
avahi:*:15458:0:99999:7:::
usbmux:*:15458:0:99999:7:::
kernoops:*:15458:0:99999:7:::
pulse:*:15458:0:99999:7:::
rtkit:*:15458:0:99999:7:::
saned:*:15458:0:99999:7:::
speech-dispatcher:!:15458:0:99999:7:::
hplip:*:15458:0:99999:7:::
user:$6$jfSu1KQm$MvrxVdnxVJLhwaVdiRL17eAqFw4GVS4bwdCUj2wfqoM2FtKbkZ5cHc0og3phwbi
o4hj6yXffVdRVFjw06auns.:15458:0:99999:7:::
guest-gr8WNL:*:15458:0:99999:7:::
guest-2IsoAK:*:16733:0:99999:7:::
sjapit:$6$I61UD1bp$BoN.mrOJ.VwVDoZHZaApI4GCnUyssbuqP/.PXlSADpEYCD1dzKjT1Ns78KTS0
JXTe.waAj04HNox2ikLm4gfm.:16733:0:99999:7:::
```

/etc/shadow ..1

Format of shadow (/etc/shadow)

Username:password:theNumberOfDaysLastChange:theNumberOfDaysPasswordMayBeChange:theNumberOfDaysPasswordMustBeChange:theNumberOfDaysToWarnUserOfAnExpiringPassword:

theNumberOfDaysAfterPasswordExpiresDisable:theNumberOfDaysAccountIsDisable:reservedForPossibleFutureUse.

/etc/shadow ..2

Format of shadow (/etc/shadow)

- **Username**, up to 8 characters. Case-sensitive, usually all lowercase. A direct match to the username in the /etc/passwd file.
- **Password**, 13 character encrypted. A blank entry (eg. ::) indicates a password is not required to log in (usually a bad idea), and a “*‘” entry (eg. :*:) indicates the account has been disabled.
- The number of days since January 1, 1970 or since the password was last changed.

/etc/shadow ..3

Format of shadow (/etc/shadow)

- The number of days before password may be changed (0 indicates it may be changed at any time)
- The number of days after which password *must* be changed (99999 indicates user can keep his or her password unchanged for many, many years)
- The number of days to warn user of an expiring password (7 for a full week)

/etc/shadow ..4

Format of shadow (/etc/shadow)

- The number of days after password expires that account is disabled
- The number of days since January 1, 1970 that an account has been disabled
- A reserved field for possible future use.

Question 6

Describe the use of a rainbow table to attack hash-/salt-based password systems.

Question 6

Describe the use of a rainbow table to attack hash-/salt-based password systems.

- Pre-compute potential hash values by:
  - Generating a large dictionary containing possible passwords,
  - For each possible password, the hash value associated with the possible salt value is generated.
  - This large dictionary is called the rainbow table.
  - The rainbow table is then used as look-up table.

Question 6a

- How can protection against such an approach be provided?

Question 6a

- How can protection against such an approach be provided?

To protect against attack from attacker using rainbow table, one can use sufficiently large salt value and sufficiently large hash length.

Question 6b

In rainbow tables, reduction functions as well as the hashing function in generating a rainbow table. How is a reduction function used?

Rainbow table: Example

- The following example demonstrates how rainbow table is constructed and how a pre-image searching is done using rainbow table.
- A small, 15 passwords are randomly selected from the word.txt file provided by Dr. Dung Duong; the password.txt file is shown in the next slide.
- For demonstration purpose, I included the hashed value of each password together with the corresponding reduction value from a typical reduction function. The reduction function I am using is a simple modulus function using the size of the password.txt file as the modulo.

Password.txt

Sno	Password	hashed value	Reduction Function
1	10th	515da2caf582ac4801cbb5d876c73c90	14
2	Ababa	bbf12b95db10da96472e2e019ffa4659	6
3	TWA	47221236d3df2a4cca11b1d7512faf7d	13
4	Abater	d48f58d9dc9af4b68b860e71f7336b44	1
5	Aaron	1c0a11cc4ddc0dbd3fa4d77232a4e22e	11
6	mundane	147e19efcaca65ee9f16ac703514b374	6
7	bake	a6ecfad3e0f9a51c6335848449a91bed	9
8	zoo	d2cbe65f53da8607e64173c1a83394fe	4
9	zombie	0eda241fc65ccf35d9743309ac395215	6
10	freehold	47ebf781047c3340fd5b0363b10c82aa	8
11	abalone	6e1ba55b046f7d62bbd6dc33b63d5ec7	4
12	sun	ebd556e6dfc99dbed29675ce1c6c68e5	15
13	heel	649be85da19882e6335962b2842385ea	11
14	insect	dce41a93f7edb175dfc59a4d52105847	7
15	prosecute	c18ac77dbe4b7211c616667e4f8fc526	11

Reduction function

- I am using a simple reduction function as follow:

$$r = MD5(password)\ mod\ sizeOfPasswordFile$$

  where:

  - r – the reduction value
  - Password – the current password to be hashed using MD5 hash function.
  - sizeOfPasswordFile – the total number of passwords contains in the password.txt file.
- In this example, the sizeOfPasswordFile is 15 (15 different passwords are in the password.txt file.)

Hash-chain

Create a hash-chain to generate entries for the rainbow table:

- Read in the list of possible passwords.

  sizeOfPasswordFile = 15

10th
Ababa
TWA
Abater
Aaron
mundane
bake
zoo
zombie
freehold
abalone
sun
heel
insect
prosecute

Hash-chain

- For each previously unused password, mark it as used. For example, the first unused password is $10\mathrm{th}$.
  - Apply MD5 hash function to the password to get a hashed-value. For example,

$$\begin{aligned} hv &= MD5(10th) \\ &= 515da2caf582ac4801cbb5d876c73c90 \end{aligned}$$

  - Next, convert the digest (hexadecimal value) into long number before we apply the reduction function by taking modulus of the size of the password file. For example,

$$\begin{aligned} r &= (2888488213\ mod\ 15\ ) + 1 \\ &= 14 \end{aligned}$$

Hash-chain

- The reduction function returns a value 14. The value 14 indicates that the 14th password in the password file will be the next password to be chained into the list. Mark the 14th password as used and repeat the previous described steps 4 more times.

- After the 4th repeat, store the first password in the list and the last hashed value into the rainbow table.

Construction of Rainbow table:

Sno	Password	hashed value	Reduction Function
1	10th	515da2caf582ac4801cbb5d876c73c90	14
2	Ababa	bbf12b95db10da96472e2e019ffa4659	6
3	TWA	47221236d3df2a4cca11b1d7512faf7d	13
4	Abater	d48f58d9dc9af4b68b860e71f7336b44	1
5	Aaron	1c0a11cc4ddc0dbd3fa4d77232a4e22e	11
6	mundane	147e19efcaca65ee9f16ac703514b374	6
7	bake	a6ecfad3e0f9a51c6335848449a91bed	9
8	zoo	d2cbe65f53da8607e64173c1a83394fe	4
9	zombie	0eda241fc65ccf35d9743309ac395215	6
10	freehold	47ebf781047c3340fd5b0363b10c82aa	8
11	abalone	6e1ba55b046f7d62bbd6dc33b63d5ec7	4
12	sun	ebd556e6dfc99dbed29675ce1c6c68e5	15
13	heel	649be85da19882e6335962b2842385ea	11
14	insect	dce41a93f7edb175dfc59a4d52105847	7
15	prosecute	c18ac77dbe4b7211c616667e4f8fc526	11

Hash-list:	
10th	515da2caf582ac4801cbb5d876c73c90
insect	dce41a93f7edb175dfc59a4d52105847
bake	a6ecfad3e0f9a51c6335848449a91bed
zombie	0eda241fc65ccf35d9743309ac395215
mundane	147e19efcaca65ee9f16ac703514b374

Rainbow Table:	
10th	147e19efcaca65ee9f16ac703514b374

Construction of Rainbow table:

Sno	Password	hashed value	Reduction Function
1	10th	515da2caf582ac4801cbb5d876c73c90	14
2	Ababa	bbf12b95db10da96472e2e019ffa4659	6
3	TWA	47221236d3df2a4cca11b1d7512faf7d	13
4	Abater	d48f58d9dc9af4b68b860e71f7336b44	1
5	Aaron	1c0a11cc4ddc0dbd3fa4d77232a4e22e	11
6	mundane	147e19efcaca65ee9f16ac703514b374	6
7	bake	a6ecfad3e0f9a51c6335848449a91bed	9
8	zoo	d2cbe65f53da8607e64173c1a83394fe	4
9	zombie	0eda241fc65ccf35d9743309ac395215	6
10	freehold	47ebf781047c3340fd5b0363b10c82aa	8
11	abalone	6e1ba55b046f7d62bbd6dc33b63d5ec7	4
12	sun	ebd556e6dfc99dbed29675ce1c6c68e5	15
13	heel	649be85da19882e6335962b2842385ea	11
14	insect	dce41a93f7edb175dfc59a4d52105847	7
15	prosecute	c18ac77dbe4b7211c616667e4f8fc526	11

Hash-list:	
Ababa	bbf12b95db10da96472e2e019ffa4659
mundane	147e19efcaca65ee9f16ac703514b374
mundane	147e19efcaca65ee9f16ac703514b374
mundane	147e19efcaca65ee9f16ac703514b374
mundane	147e19efcaca65ee9f16ac703514b374

Rainbow Table:	
10th	147e19efcaca65ee9f16ac703514b374
Ababa	147e19efcaca65ee9f16ac703514b374

Construction of Rainbow table:

Sno	Password	hashed value	Reduction Function
1	10th	515da2caf582ac4801cbb5d876c73c90	14
2	Ababa	bbf12b95db10da96472e2e019ffa4659	6
3	TWA	47221236d3df2a4cca11b1d7512faf7d	13
4	Abater	d48f58d9dc9af4b68b860e71f7336b44	1
5	Aaron	1c0a11cc4ddc0dbd3fa4d77232a4e22e	11
6	mundane	147e19efcaca65ee9f16ac703514b374	6
7	bake	a6ecfad3e0f9a51c6335848449a91bed	9
8	zoo	d2cbe65f53da8607e64173c1a83394fe	4
9	zombie	0eda241fc65ccf35d9743309ac395215	6
10	freehold	47ebf781047c3340fd5b0363b10c82aa	8
11	abalone	6e1ba55b046f7d62bbd6dc33b63d5ec7	4
12	sun	ebd556e6dfc99dbed29675ce1c6c68e5	15
13	heel	649be85da19882e6335962b2842385ea	11
14	insect	dce41a93f7edb175dfc59a4d52105847	7
15	prosecute	c18ac77dbe4b7211c616667e4f8fc526	11

Hash-list:

TWA	47221236d3df2a4cca11b1d7512faf7d
heel	649be85da19882e6335962b2842385ea
abalone	6e1ba55b046f7d62bbd6dc33b63d5ec7
Abater	d48f58d9dc9af4b68b860e71f7336b44
10th	515da2caf582ac4801cbb5d876c73c90

Rainbow Table:

10th	147e19efcaca65ee9f16ac703514b374
Ababa	147e19efcaca65ee9f16ac703514b374
TWA	515da2caf582ac4801cbb5d876c73c90

Construction of Rainbow table:

Sno	Password	hashed value	Reduction Function
1	10th	515da2caf582ac4801cbb5d876c73c90	14
2	Ababa	bbf12b95db10da96472e2e019ffa4659	6
3	TWA	47221236d3df2a4cca11b1d7512faf7d	13
4	Abater	d48f58d9dc9af4b68b860e71f7336b44	1
5	Aaron	1c0a11cc4ddc0dbd3fa4d77232a4e22e	11
6	mundane	147e19efcaca65ee9f16ac703514b374	6
7	bake	a6ecfad3e0f9a51c6335848449a91bed	9
8	zoo	d2cbe65f53da8607e64173c1a83394fe	4
9	zombie	0eda241fc65ccf35d9743309ac395215	6
10	freehold	47ebf781047c3340fd5b0363b10c82aa	8
11	abalone	6e1ba55b046f7d62bbd6dc33b63d5ec7	4
12	sun	ebd556e6dfc99dbed29675ce1c6c68e5	15
13	heel	649be85da19882e6335962b2842385ea	11
14	insect	dce41a93f7edb175dfc59a4d52105847	7
15	prosecute	c18ac77dbe4b7211c616667e4f8fc526	11

Hash-list:

sun	ebd556e6dfc99dbed29675ce1c6c68e5
prosecute	c18ac77dbe4b7211c616667e4f8fc526
abalone	6e1ba55b046f7d62bbd6dc33b63d5ec7
Abater	d48f58d9dc9af4b68b860e71f7336b44
10th	515da2caf582ac4801cbb5d876c73c90

Rainbow Table:

10th	147e19efcaca65ee9f16ac703514b374
Ababa	147e19efcaca65ee9f16ac703514b374
TWA	515da2caf582ac4801cbb5d876c73c90
sun	515da2caf582ac4801cbb5d876c73c90

Construction of Rainbow table:

Sno	Password	hashed value	Reduction Function
1	10th	515da2caf582ac4801cbb5d876c73c90	14
2	Ababa	bbf12b95db10da96472e2e019ffa4659	6
3	TWA	47221236d3df2a4cca11b1d7512faf7d	13
4	Abater	d48f58d9dc9af4b68b860e71f7336b44	1
5	Aaron	1c0a11cc4ddc0dbd3fa4d77232a4e22e	11
6	mundane	147e19efcaca65ee9f16ac703514b374	6
7	bake	a6ecfad3e0f9a51c6335848449a91bed	9
8	zoo	d2cbe65f53da8607e64173c1a83394fe	4
9	zombie	0eda241fc65ccf35d9743309ac395215	6
10	freehold	47ebf781047c3340fd5b0363b10c82aa	8
11	abalone	6e1ba55b046f7d62bbd6dc33b63d5ec7	4
12	sun	ebd556e6dfc99dbed29675ce1c6c68e5	15
13	heel	649be85da19882e6335962b2842385ea	11
14	insect	dce41a93f7edb175dfc59a4d52105847	7
15	prosecute	c18ac77dbe4b7211c616667e4f8fc526	11

Hash-list:

zoo	d2cbe65f53da8607e64173c1a83394fe
Abater	d48f58d9dc9af4b68b860e71f7336b44
10th	515da2caf582ac4801cbb5d876c73c90
insect	dce41a93f7edb175dfc59a4d52105847
bake	a6ecfad3e0f9a51c6335848449a91bed

Rainbow Table:

10th	147e19efcaca65ee9f16ac703514b374
Ababa	147e19efcaca65ee9f16ac703514b374
TWA	515da2caf582ac4801cbb5d876c73c90
sun	515da2caf582ac4801cbb5d876c73c90
zoo	a6ecfad3e0f9a51c6335848449a91bed

Construction of Rainbow table:

Sno	Password	hashed value	Reduction Function
1	10th	515da2caf582ac4801cbb5d876c73c90	14
2	Ababa	bbf12b95db10da96472e2e019ffa4659	6
3	TWA	47221236d3df2a4cca11b1d7512faf7d	13
4	Abater	d48f58d9dc9af4b68b860e71f7336b44	1
5	Aaron	1c0a11cc4ddc0dbd3fa4d77232a4e22e	11
6	mundane	147e19efcaca65ee9f16ac703514b374	6
7	bake	a6ecfad3e0f9a51c6335848449a91bed	9
8	zoo	d2cbe65f53da8607e64173c1a83394fe	4
9	zombie	0eda241fc65ccf35d9743309ac395215	6
10	freehold	47ebf781047c3340fd5b0363b10c82aa	8
11	abalone	6e1ba55b046f7d62bbd6dc33b63d5ec7	4
12	sun	ebd556e6dfc99dbed29675ce1c6c68e5	15
13	heel	649be85da19882e6335962b2842385ea	11
14	insect	dce41a93f7edb175dfc59a4d52105847	7
15	prosecute	c18ac77dbe4b7211c616667e4f8fc526	11

Hash-list:	
Aaron	1c0a11cc4ddc0dbd3fa4d77232a4e22e
abalone	6e1ba55b046f7d62bbd6dc33b63d5ec7
Abater	d48f58d9dc9af4b68b860e71f7336b44
10th	515da2caf582ac4801cbb5d876c73c90
insect	dce41a93f7edb175dfc59a4d52105847

Rainbow Table:	
10th	147e19efcaca65ee9f16ac703514b374
Ababa	147e19efcaca65ee9f16ac703514b374
TWA	515da2caf582ac4801cbb5d876c73c90
sun	515da2caf582ac4801cbb5d876c73c90
zoo	a6ecfad3e0f9a51c6335848449a91bed
Aaron	dce41a93f7edb175dfc59a4d52105847

Construction of Rainbow table:

Sno	Password	hashed value	Reduction Function
1	10th	515da2caf582ac4801cbb5d876c73c90	14
2	Ababa	bbf12b95db10da96472e2e019ffa4659	6
3	TWA	47221236d3df2a4cca11b1d7512faf7d	13
4	Abater	d48f58d9dc9af4b68b860e71f7336b44	1
5	Aaron	1c0a11cc4ddc0dbd3fa4d77232a4e22e	11
6	mundane	147e19efcaca65ee9f16ac703514b374	6
7	bake	a6ecfad3e0f9a51c6335848449a91bed	9
8	zoo	d2cbe65f53da8607e64173c1a83394fe	4
9	zombie	0eda241fc65ccf35d9743309ac395215	6
10	freehold	47ebf781047c3340fd5b0363b10c82aa	8
11	abalone	6e1ba55b046f7d62bbd6dc33b63d5ec7	4
12	sun	ebd556e6dfc99dbed29675ce1c6c68e5	15
13	heel	649be85da19882e6335962b2842385ea	11
14	insect	dce41a93f7edb175dfc59a4d52105847	7
15	prosecute	c18ac77dbe4b7211c616667e4f8fc526	11

Hash-list:

freehold	47ebf781047c3340fd5b0363b10c82aa
zoo	d2cbe65f53da8607e64173c1a83394fe
Abater	d48f58d9dc9af4b68b860e71f7336b44
10th	515da2caf582ac4801cbb5d876c73c90
insect	dce41a93f7edb175dfc59a4d52105847

Rainbow Table:

10th	147e19efcaca65ee9f16ac703514b374
Ababa	147e19efcaca65ee9f16ac703514b374
TWA	515da2caf582ac4801cbb5d876c73c90
sun	515da2caf582ac4801cbb5d876c73c90
zoo	a6ecfad3e0f9a51c6335848449a91bed
Aaron	dce41a93f7edb175dfc59a4d52105847
freehold	dce41a93f7edb175dfc59a4d52105847

The rainbow table

10th	147e19efcaca65ee9f16ac703514b374
Ababa	147e19efcaca65ee9f16ac703514b374
TWA	515da2caf582ac4801cbb5d876c73c90
sun	515da2caf582ac4801cbb5d876c73c90
zoo	a6ecfad3e0f9a51c6335848449a91bed
Aaron	dce41a93f7edb175dfc59a4d52105847
freehold	dce41a93f7edb175dfc59a4d52105847

Construction of Rainbow table:

Sno	Password	hashed value	Reduction Function
1	10th	515da2caf582ac4801cbb5d876c73c90	14
2	Ababa	bbf12b95db10da96472e2e019ffa4659	6
3	TWA	47221236d3df2a4cca11b1d7512faf7d	13
4	Abater	d48f58d9dc9af4b68b860e71f7336b44	1
5	Aaron	1c0a11cc4ddc0dbd3fa4d77232a4e22e	11
6	mundane	147e19efcaca65ee9f16ac703514b374	6
7	bake	a6ecfad3e0f9a51c6335848449a91bed	9
8	zoo	d2cbe65f53da8607e64173c1a83394fe	4
9	zombie	0eda241fc65ccf35d9743309ac395215	6
10	freehold	47ebf781047c3340fd5b0363b10c82aa	8
11	abalone	6e1ba55b046f7d62bbd6dc33b63d5ec7	4
12	sun	ebd556e6dfc99dbed29675ce1c6c68e5	15
13	heel	649be85da19882e6335962b2842385ea	11
14	insect	dce41a93f7edb175dfc59a4d52105847	7
15	prosecute	c18ac77dbe4b7211c616667e4f8fc526	11

Rainbow Table:	
10th	147e19efcaca65ee9f16ac703514b374
Ababa	147e19efcaca65ee9f16ac703514b374
TWA	515da2caf582ac4801cbb5d876c73c90
sun	515da2caf582ac4801cbb5d876c73c90
zoo	a6ecfad3e0f9a51c6335848449a91bed
Aaron	dce41a93f7edb175dfc59a4d52105847
freehold	dce41a93f7edb175dfc59a4d52105847

Example: Successful search of a password in a chain.
User enter 6e1ba55b046f7d62bbd6dc33b63d5ec7.

1. Check if the hash-value is found in the rainbow table. No, 6e1ba55b046f7d62bbd6dc33b63d5ec7 is not found in the rainbow table.
2. Apply the reduction function to the hash-value until a match is found in the rainbow table.

6e1ba55b046f7d62bbd6dc33b63d5ec7 → Abater → D48f58d9dc9af4b68b860e71f7336b44 → 10th → 515da2caf582ac4801cbb5d876c73c90.

Construction of Rainbow table:

Sno	Password	hashed value	Reduction Function
1	10th	515da2caf582ac4801cbb5d876c73c90	14
2	Ababa	bbf12b95db10da96472e2e019ffa4659	6
3	TWA	47221236d3df2a4cca11b1d7512faf7d	13
4	Abater	d48f58d9dc9af4b68b860e71f7336b44	1
5	Aaron	1c0a11cc4ddc0dbd3fa4d77232a4e22e	11
6	mundane	147e19efcaca65ee9f16ac703514b374	6
7	bake	a6ecfad3e0f9a51c6335848449a91bed	9
8	zoo	d2cbe65f53da8607e64173c1a83394fe	4
9	zombie	0eda241fc65ccf35d9743309ac395215	6
10	freehold	47ebf781047c3340fd5b0363b10c82aa	8
11	abalone	6e1ba55b046f7d62bbd6dc33b63d5ec7	4
12	sun	ebd556e6dfc99dbed29675ce1c6c68e5	15
13	heel	649be85da19882e6335962b2842385ea	11
14	insect	dce41a93f7edb175dfc59a4d52105847	7
15	prosecute	c18ac77dbe4b7211c616667e4f8fc526	11

Rainbow Table:	
10th	147e19efcaca65ee9f16ac703514b374
Ababa	147e19efcaca65ee9f16ac703514b374
TWA	515da2caf582ac4801cbb5d876c73c90
sun	515da2caf582ac4801cbb5d876c73c90
zoo	a6ecfad3e0f9a51c6335848449a91bed
Aaron	dce41a93f7edb175dfc59a4d52105847
freehold	dce41a93f7edb175dfc59a4d52105847

Example: Successful search of a password in a chain.
User enter 6e1ba55b046f7d62bbd6dc33b63d5ec7.

3. Starting with the password TWA a search is done. After a few reduction is done, the hash-value 6e1ba55b046f7d62bbd6dc33b63d5ec7 is found.
4. The password (preimage of the hash-value 6e1ba55b046f7d62bbd6dc33b63d5ec7) is abalone.

Construction of Rainbow table:

Sno	Password	hashed value	Reduction Function
1	10th	515da2caf582ac4801cbb5d876c73c90	14
2	Ababa	bbf12b95db10da96472e2e019ffa4659	6
3	TWA	47221236d3df2a4cca11b1d7512faf7d	13
4	Abater	d48f58d9dc9af4b68b860e71f7336b44	1
5	Aaron	1c0a11cc4ddc0dbd3fa4d77232a4e22e	11
6	mundane	147e19efcaca65ee9f16ac703514b374	6
7	bake	a6ecfad3e0f9a51c6335848449a91bed	9
8	zoo	d2cbe65f53da8607e64173c1a83394fe	4
9	zombie	0eda241fc65ccf35d9743309ac395215	6
10	freehold	47ebf781047c3340fd5b0363b10c82aa	8
11	abalone	6e1ba55b046f7d62bbd6dc33b63d5ec7	4
12	sun	ebd556e6dfc99dbed29675ce1c6c68e5	15
13	heel	649be85da19882e6335962b2842385ea	11
14	insect	dce41a93f7edb175dfc59a4d52105847	7
15	prosecute	c18ac77dbe4b7211c616667e4f8fc526	11

Rainbow Table:	
10th	147e19efcaca65ee9f16ac703514b374
Ababa	147e19efcaca65ee9f16ac703514b374
TWA	515da2caf582ac4801cbb5d876c73c90
sun	515da2caf582ac4801cbb5d876c73c90
zoo	a6ecfad3e0f9a51c6335848449a91bed
Aaron	dce41a93f7edb175dfc59a4d52105847
freehold	dce41a93f7edb175dfc59a4d52105847

Example: Successful search of a password in a chain that involve collision. User enter d2cbe65f53da8607e64173c1a83394fe.

1. Check if the hash-value is found in the rainbow table. No, d2cbe65f53da8607e64173c1a83394fe is not found in the rainbow table.
2. Apply the reduction function to the hash-value until a match is found in the rainbow table.

d2cbe65f53da8607e64173c1a83394fe → Abater → D48f58d9dc9af4b68b860e71f7336b44 → 10th → 515da2caf582ac4801cbb5d876c73c90.

Construction of Rainbow table:

Sno	Password	hashed value	Reduction Function
1	10th	515da2caf582ac4801cbb5d876c73c90	14
2	Ababa	bbf12b95db10da96472e2e019ffa4659	6
3	TWA	47221236d3df2a4cca11b1d7512faf7d	13
4	Abater	d48f58d9dc9af4b68b860e71f7336b44	1
5	Aaron	1c0a11cc4ddc0dbd3fa4d77232a4e22e	11
6	mundane	147e19efcaca65ee9f16ac703514b374	6
7	bake	a6ecfad3e0f9a51c6335848449a91bed	9
8	zoo	d2cbe65f53da8607e64173c1a83394fe	4
9	zombie	0eda241fc65ccf35d9743309ac395215	6
10	freehold	47ebf781047c3340fd5b0363b10c82aa	8
11	abalone	6e1ba55b046f7d62bbd6dc33b63d5ec7	4
12	sun	ebd556e6dfc99dbed29675ce1c6c68e5	15
13	heel	649be85da19882e6335962b2842385ea	11
14	insect	dce41a93f7edb175dfc59a4d52105847	7
15	prosecute	c18ac77dbe4b7211c616667e4f8fc526	11

Rainbow Table:

10th	147e19efcaca65ee9f16ac703514b374
Ababa	147e19efcaca65ee9f16ac703514b374
TWA	515da2caf582ac4801cbb5d876c73c90
sun	515da2caf582ac4801cbb5d876c73c90
zoo	a6ecfad3e0f9a51c6335848449a91bed
Aaron	dce41a93f7edb175dfc59a4d52105847
freehold	dce41a93f7edb175dfc59a4d52105847

Example: Successful search of a password in a chain that involve collision. User enter d2cbe65f53da8607e64173c1a83394fe.

3. Starting with the password TWA a search is done. After 4 reductions are done, the hash-value d2cbe65f53da8607e64173c1a83394fe is still not found in the rainbow table.

Construction of Rainbow table:

Sno	Password	hashed value	Reduction Function
1	10th	515da2caf582ac4801cbb5d876c73c90	14
2	Ababa	bbf12b95db10da96472e2e019ffa4659	6
3	TWA	47221236d3df2a4cca11b1d7512faf7d	13
4	Abater	d48f58d9dc9af4b68b860e71f7336b44	1
5	Aaron	1c0a11cc4ddc0dbd3fa4d77232a4e22e	11
6	mundane	147e19efcaca65ee9f16ac703514b374	6
7	bake	a6ecfad3e0f9a51c6335848449a91bed	9
8	zoo	d2cbe65f53da8607e64173c1a83394fe	4
9	zombie	0eda241fc65ccf35d9743309ac395215	6
10	freehold	47ebf781047c3340fd5b0363b10c82aa	8
11	abalone	6e1ba55b046f7d62bbd6dc33b63d5ec7	4
12	sun	ebd556e6dfc99dbed29675ce1c6c68e5	15
13	heel	649be85da19882e6335962b2842385ea	11
14	insect	dce41a93f7edb175dfc59a4d52105847	7
15	prosecute	c18ac77dbe4b7211c616667e4f8fc526	11

Rainbow Table:

10th	147e19efcaca65ee9f16ac703514b374
Ababa	147e19efcaca65ee9f16ac703514b374
TWA	515da2caf582ac4801cbb5d876c73c90
sun	515da2caf582ac4801cbb5d876c73c90
zoo	a6ecfad3e0f9a51c6335848449a91bed
Aaron	dce41a93f7edb175dfc59a4d52105847
freehold	dce41a93f7edb175dfc59a4d52105847

Example: Successful search of a password in a chain that involve collision. User enter d2cbe65f53da8607e64173c1a83394fe.

4. Using the next chain starting with the password sun, a search is done. Similarly, after 4 reductions, the hash-value d2cbe65f53da8607e64173c1a83394fe is still cannot be found.

Construction of Rainbow table:

Sno	Password	hashed value	Reduction Function
1	10th	515da2caf582ac4801cbb5d876c73c90	14
2	Ababa	bbf12b95db10da96472e2e019ffa4659	6
3	TWA	47221236d3df2a4cca11b1d7512faf7d	13
4	Abater	d48f58d9dc9af4b68b860e71f7336b44	1
5	Aaron	1c0a11cc4ddc0dbd3fa4d77232a4e22e	11
6	mundane	147e19efcaca65ee9f16ac703514b374	6
7	bake	a6ecfad3e0f9a51c6335848449a91bed	9
8	zoo	d2cbe65f53da8607e64173c1a83394fe	4
9	zombie	0eda241fc65ccf35d9743309ac395215	6
10	freehold	47ebf781047c3340fd5b0363b10c82aa	8
11	abalone	6e1ba55b046f7d62bbd6dc33b63d5ec7	4
12	sun	ebd556e6dfc99dbed29675ce1c6c68e5	15
13	heel	649be85da19882e6335962b2842385ea	11
14	insect	dce41a93f7edb175dfc59a4d52105847	7
15	prosecute	c18ac77dbe4b7211c616667e4f8fc526	11

Rainbow Table:	
10th	147e19efcaca65ee9f16ac703514b374
Ababa	147e19efcaca65ee9f16ac703514b374
TWA	515da2caf582ac4801cbb5d876c73c90
sun	515da2caf582ac4801cbb5d876c73c90
zoo	a6ecfad3e0f9a51c6335848449a91bed
Aaron	dce41a93f7edb175dfc59a4d52105847
freehold	dce41a93f7edb175dfc59a4d52105847

Example: Successful search of a password in a chain that involve collision. User enter d2cbe65f53da8607e64173c1a83394fe.

4. Similarly, continue with the next chain starting with password zoo. This time a match is found, and the password (preimage of the hash-value d2cbe65f53da8607e64173c1a83394fe) is zoo.

Question 6b

- How large is a rainbow table? What is it a function of?

Question 7

Alice can read and write with respect to the file x, can read the file y, and can execute the file z. Bob can read x, and can read and write with respect to y.

a) Draw up an access control matrix for this situation.

b) Write a set of access control lists for this situation. Which list is associated with which file?

c) Write a set of capability lists for this situation. With what is each list associated?

Question 7

a) Access control matrix:

	x	y	z
Alice	rw	r	e
Bob	r	rw	

b) Access control list:

x : (Alice, rw), (Bob, r)

y: (Alice, r), (Bob, rw)

z: (Alice, x)

Question 7

c) Capabilities:

Alice: (x, rw), (y, r), (z, e)

Bob: (x, r), (y, rw)

Question 8

Consider that Alice can perform actions x_2 and x_4 on objects O_1 and O_2, that Bob can perform actions x_2 and x_3 on objects O_2 and O_3, and the Carol can perform actions x_1 and x_3 on objects O_3 and O_1.

a) Consider that two users can collaborate to perform an action on an object if and only if they are each independently allowed to perform that action on the object. What collaborations are possible?

Question 8

a.

	O1	O2	O3
Alice	X2, X4	X2, X4	
Bob		X2, X3	X2, X3
Carol	X1, X3		X1, X3

	O1	O2	O3
Alice	X2, X4	**X2**, X4	
Bob		**X2**, X3	X2, **X3**
Carol	X1, X3		X1, **X3**

Two collaborations – Alice and Bob can collaborate to perform activity X2, and Bob and Carol can collaborate to perform activity X3.

Question 8 (…continue…)

b) We want to ensure that each action can be performed collaboratively on at least one object, and that every object can have at least one collaborative action occurring on it.

   i. Describe a set of changes to the permissions that would allow this.

   ii. If adding a single permission for a single object counts as a single change, what is the minimal number of changes required to obtain the desired property?

   iii. If you added an additional user, without cost, would the answer to the previous question be the same? Explain

Question 8

b(i).

	O1	O2	O3
Alice	X2, X4	X2, X4	
Bob		X2, X3	X2, X3
Carol	X1, X3		X1, X3

	O1	O2	O3
Alice	**X1**, X4	X2, X4	
Bob		X2, X3	X2, X3
Carol	X1, **X4**		X1, X3

2 changes – Replace X2 with X1 for Alice on O1 and replace X3 with X4 for Carol on O1.

- Action X1 can be collaborated between Alice and Carol on O1.
- Action X2 can be collaborated between Alice and Bob on O2.
- Action X3 can be collaborated between Bob and Carol on O3.
- Action X4 can be collaborated between Alice and Carol on O4.

Question 8

b(ii).

	O1	O2	O3
Alice	X2, X4	X2, X4	
Bob	**X1**	X2, X3	X2, X3
Carol	X1, X3	**X4**	X1, X3

- 2 changes – Add X1 to Bob and add X4 to Carol.
- Action X1 can be collaborated among Bob and Carol on object O1.
- Action X2 can be collaborated among Alice and Bob on object O2.
- Action X3 can be collaborated among Bob and Carol on object O3.
- Action X4 can be collaborated among Alice and Carol on object O4.

Question 8

b(iii).

	O1	O2	O3
Alice	X2, X4	**X2**, **X4**	
Bob		**X2**, X3	X2, **X3**
Carol	**X1**, X3		X1, **X3**
Danny	**X1**	**X4**	

- 2 changes – Give X1 on O1 and X4 on O2 to Danny.
- With additional subject added, the mandatory constraint can be handled easier.

Question 8 (…continue…)

c) If, in addition to the property described in the last question, we want each user to share in exactly one collaboration, what restrictions apply?

Question 8

	O1	O2	O3
Alice	X2, X4	X2, X4	X1, X4
Bob	X2, X3	X2, X3	X2, X3
Carol	X1, X3	X3	X1, X3
Danny	X1, X4	X4	X2, X4

With 4 users, it is not possible to achieve the requirement without overlapping. We need to have a minimum of 8 users to ensure that each user shares exactly one collaboration. No overlapping process in each role (for each user); in other words, no multiple related activities in each role (user).

Question 9

Suppose a user wishes to edit the file F_1 in a capability-based system.

a) How is he sure that the editor cannot access any other files?

b) Could this be done in an ACL based system? If so, how? If not, why not?

Question 9

a) Check the capability of the editor to know which files it can access.

Capabilities:
editor: (F_1, w), (F_2, r), (F_3, e)

The system just needs to check the capability of the editor.

Question 9

b) It is hard to achieve in an ACL based system, because an ACL shows authorization of a single file. To find out the privileges the editor possesses, the system has to go through all ACLs.

Access control list:

F_1 : (editor, rw), (Bob, r)

F_2: (editor, r), (Bob, rw)

F_3: (editor, x)

The system needs to go through all the ACL for all objects.

Question 10

Assume you have 3 ordinary staff and a manager in one office.

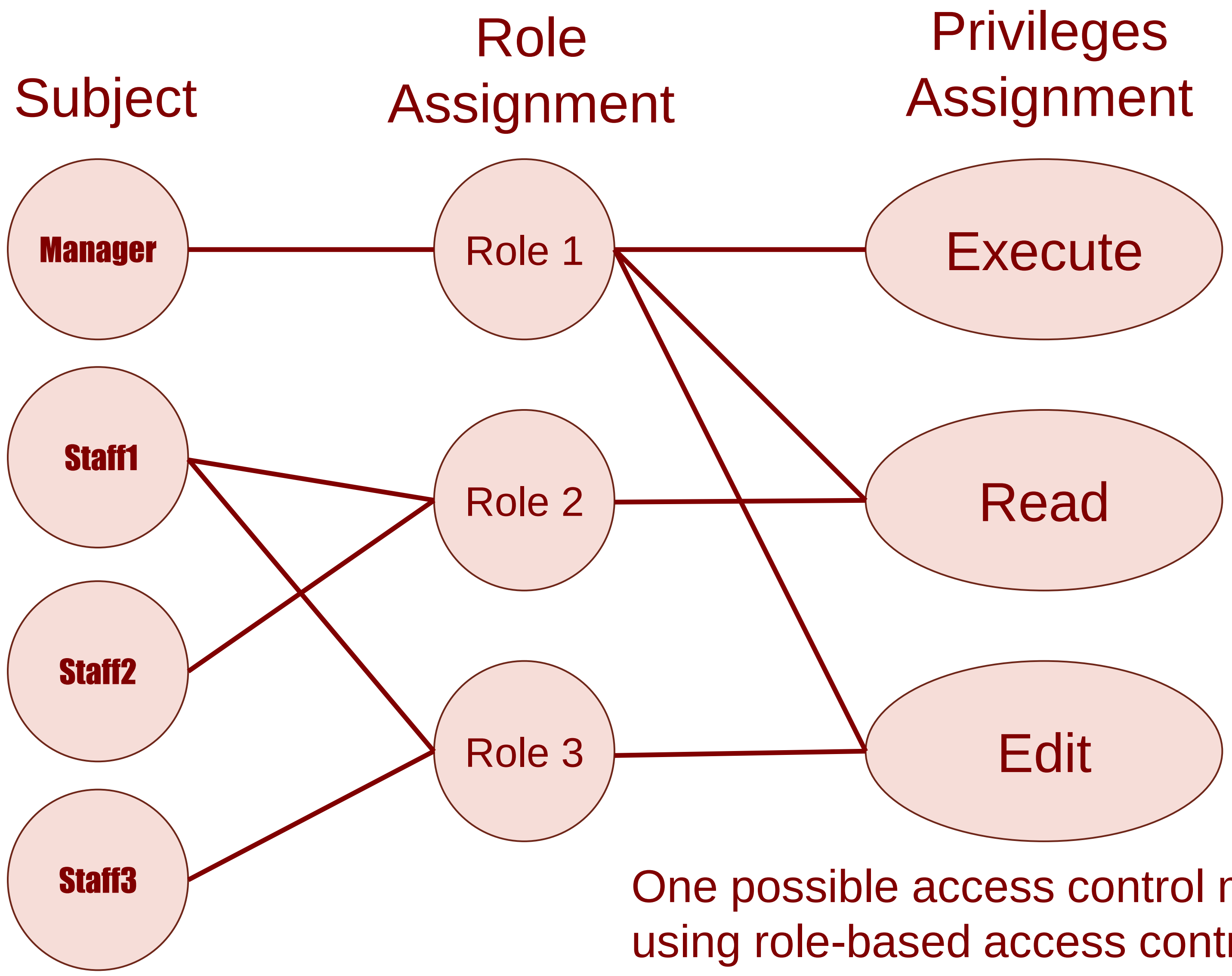
a) What sort of access control model might be suitable for representing staff permissions with respect to a staff in the office?

b) How much such access control be implemented?

Question 10

a) One popular access control model suitable for this case is a role-based access control model (RBAC). RBAC associates permissions with roles, and users are made members of appropriate role. The relationship of users to roles is many to many, as is the relationship of roles to resources, or system objects. This model greatly simplifies management of complex permissions.

Question 10



One possible access control manages using role-based access control model.

Question 10

b) Role-based access control lends itself to an effective implementation of the principle of least privilege.

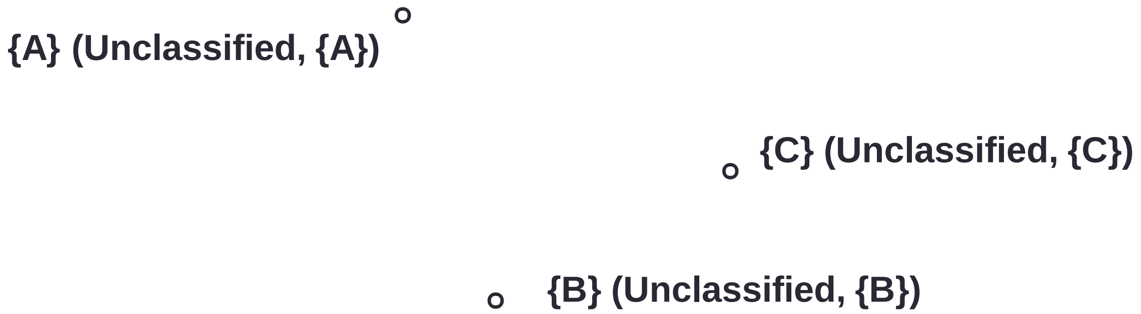
- Each role should contain the minimum set of access rights needed for that role.
- A user is assigned to a role that enables him or her to perform only what is required for that role.
- Multiple users assigned to the same role, enjoy the same minimal set of access rights.

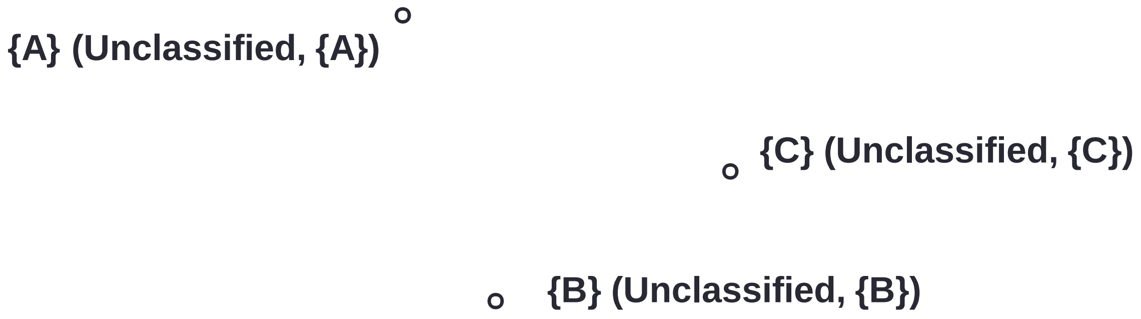
Question 11

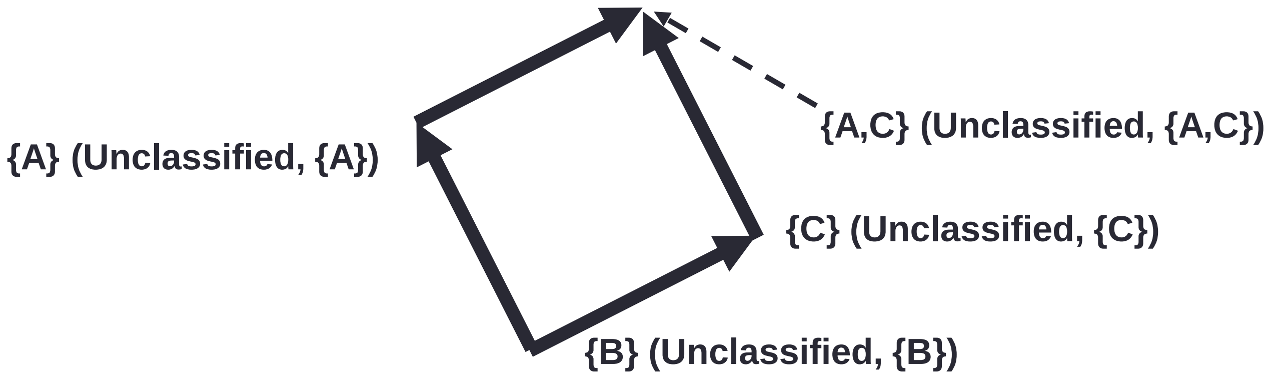
In a lattice based system a security level is composed of a clearance/classification and a composite category. So, given classifications Top Secret, Secret, Confidential, and Unclassified (ordered from highest to lowest), and the categories A, B, and C, we have security levels like (Top Secret, {A, C}). Such a level implies Top Secret clearance in {A} AND in {C}, and that composite {A, C} dominates both {A} and {C}. The composite category part of the clearance {A, C}, is higher than having both A and C clearance independently. In other words, the classification of something at (Top Secret, {A,C}) implies it is not enough to have (Top Secret, {A}) OR (Top Secret, {C}) to read this, both are needed. For each of the following specify what type of access (read, write, read & write or none) is allowed. Use BLP rules to determine access.

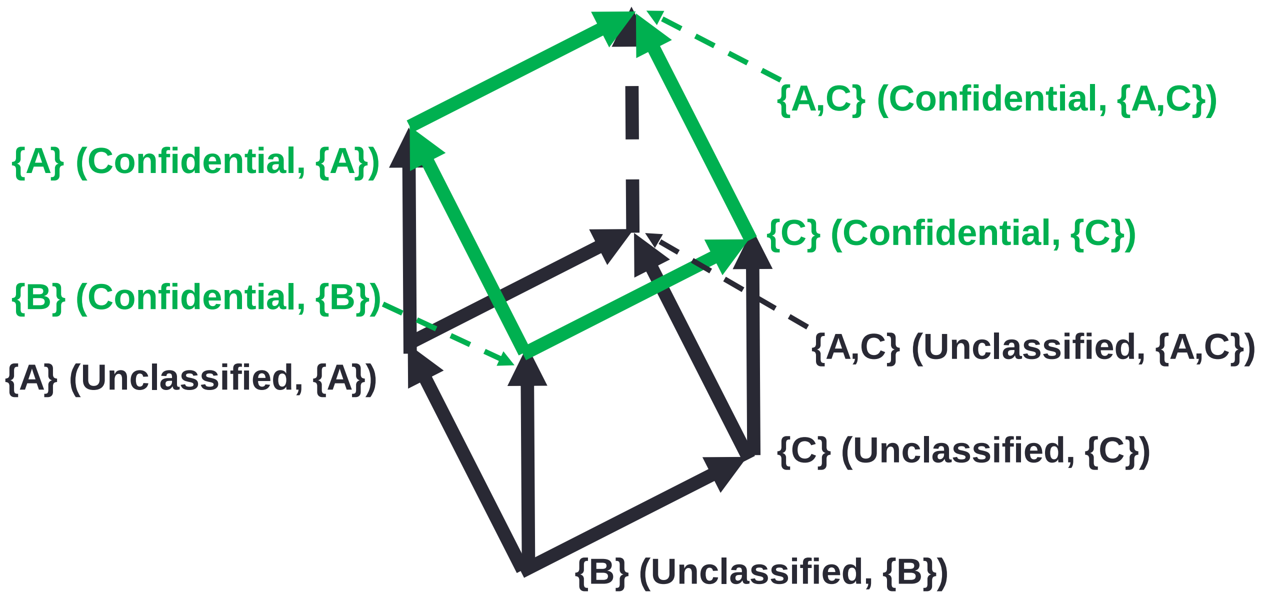
Question 11

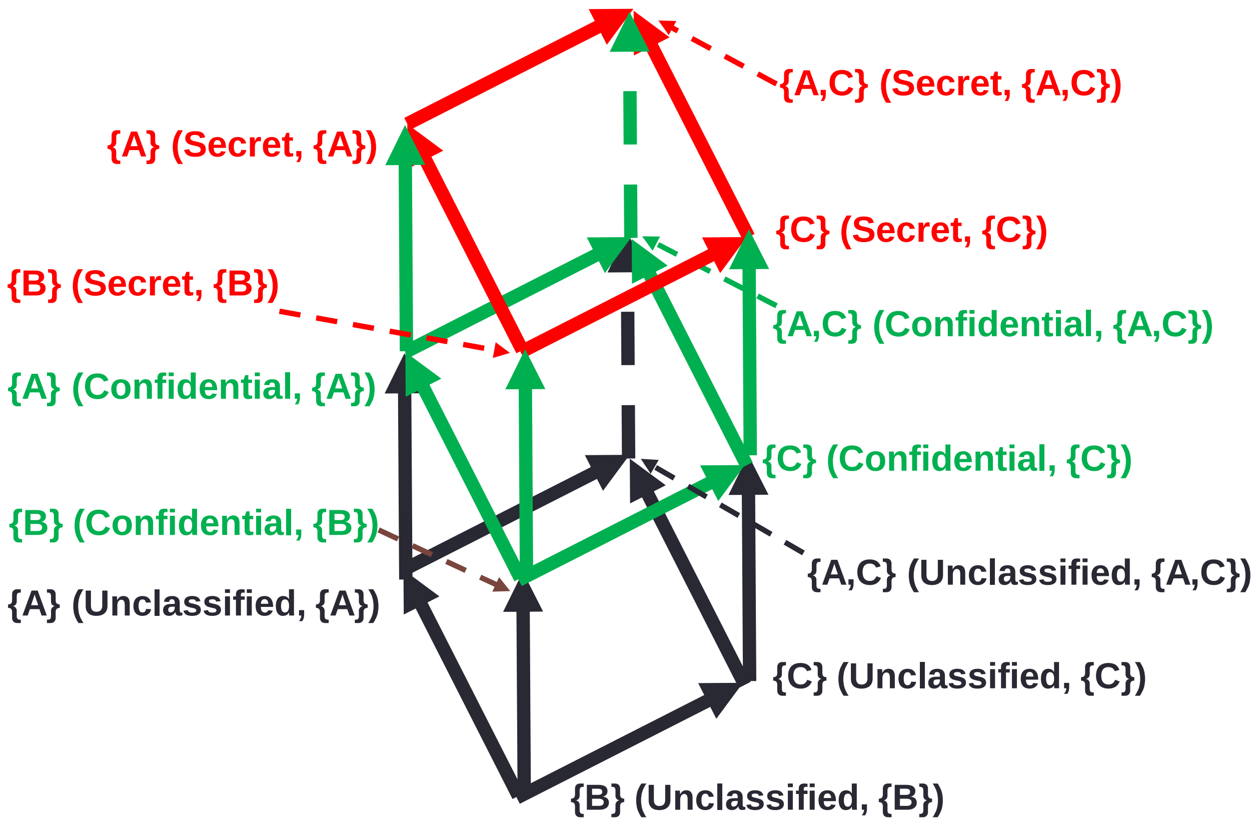
a) Alice, cleared for (Top Secret, {A,C}), wants to access a document classified (Secret, {B,C}).
b) Bob, cleared for (Confidential, {C}), wants to access a document classified (Confidential, {B})
c) Chris, cleared for (Secret, {C}), wants to access a document classified (Confidential, {A})
d) Dan, cleared for (Top Secret, {A,C}), wants to access a document classified (Confidential, {A})
e) Eve, who has no clearances (and so works at the Unclassified level), wants to access a document classified (Confidential, {B})

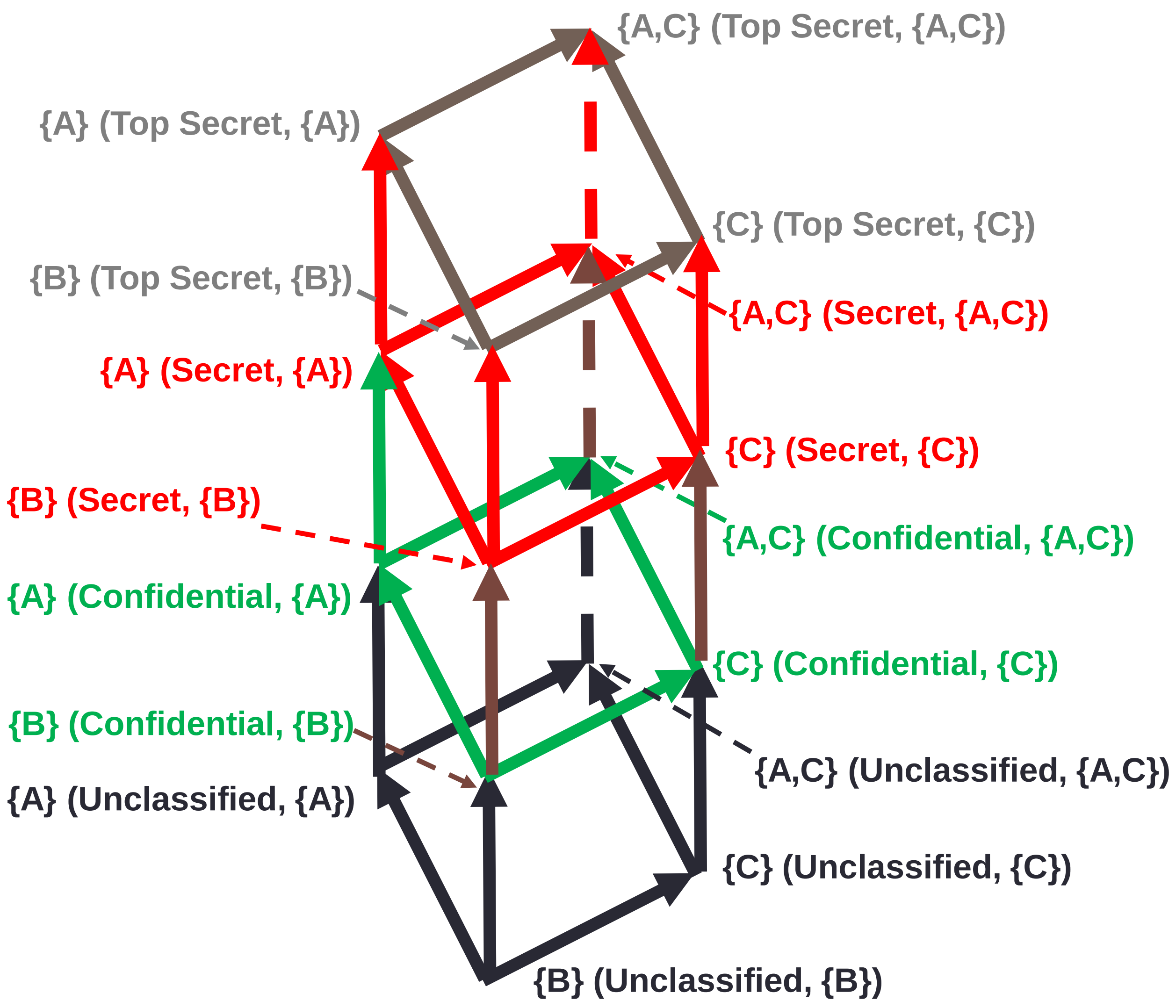

{A} (Unclassified, {A})
{C} (Unclassified, {C})
{B} (Unclassified, {B})


{A} (Unclassified, {A})
{C} (Unclassified, {C})
{B} (Unclassified, {B})


{A,C} (Unclassified, {A,C})
{A} (Unclassified, {A})
{C} (Unclassified, {C})
{B} (Unclassified, {B})


{A,C} (Confidential, {A,C})
{A} (Confidential, {A})
{C} (Confidential, {C})
{B} (Confidential, {B})
{A,C} (Unclassified, {A,C})
{A} (Unclassified, {A})
{C} (Unclassified, {C})
{B} (Unclassified, {B})

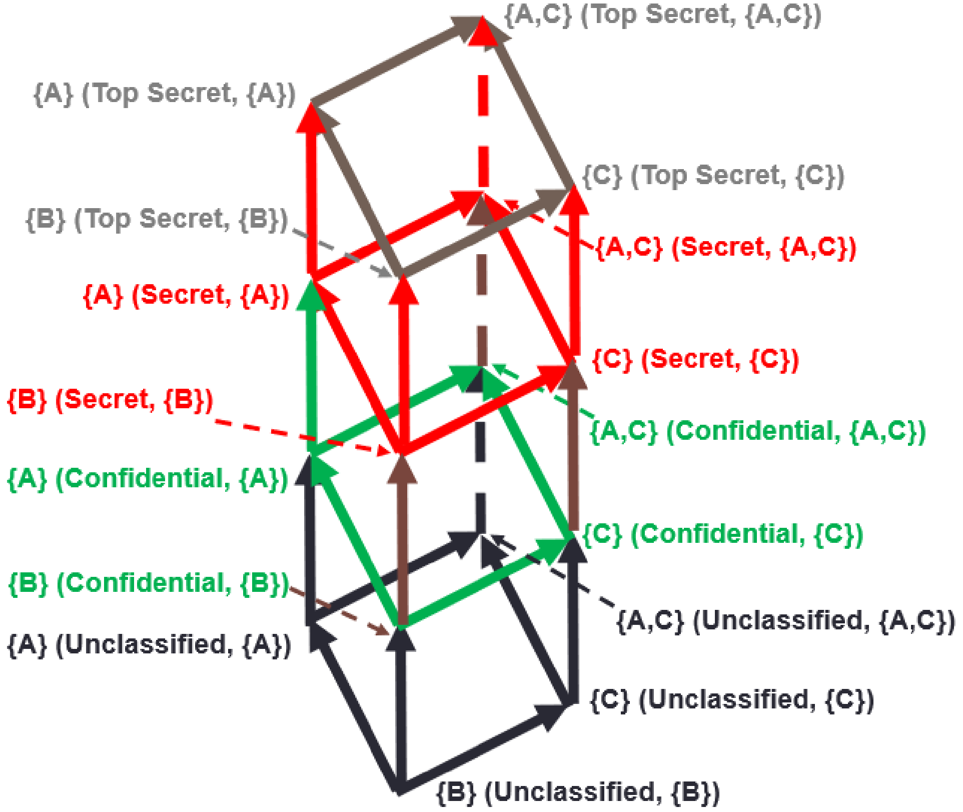

{A,C} (Secret, {A,C})
{A} (Secret, {A})
{C} (Secret, {C})
{B} (Secret, {B})
{A,C} (Confidential, {A,C})
{A} (Confidential, {A})
{C} (Confidential, {C})
{B} (Confidential, {B})
{A,C} (Unclassified, {A,C})
{A} (Unclassified, {A})
{C} (Unclassified, {C})
{B} (Unclassified, {B})


{A,C} (Top Secret, {A,C})
{A} (Top Secret, {A})
{C} (Top Secret, {C})
{B} (Top Secret, {B})
{A,C} (Secret, {A,C})
{A} (Secret, {A})
{C} (Secret, {C})
{B} (Secret, {B})
{A,C} (Confidential, {A,C})
{A} (Confidential, {A})
{C} (Confidential, {C})
{B} (Confidential, {B})
{A,C} (Unclassified, {A,C})
{A} (Unclassified, {A})
{C} (Unclassified, {C})
{B} (Unclassified, {B})

Question 11

a) Alice, cleared for (Top Secret, {A,C}), wants to access a document classified (Secret, {B,C}).

Time allocated: 5 minutes



Question 11

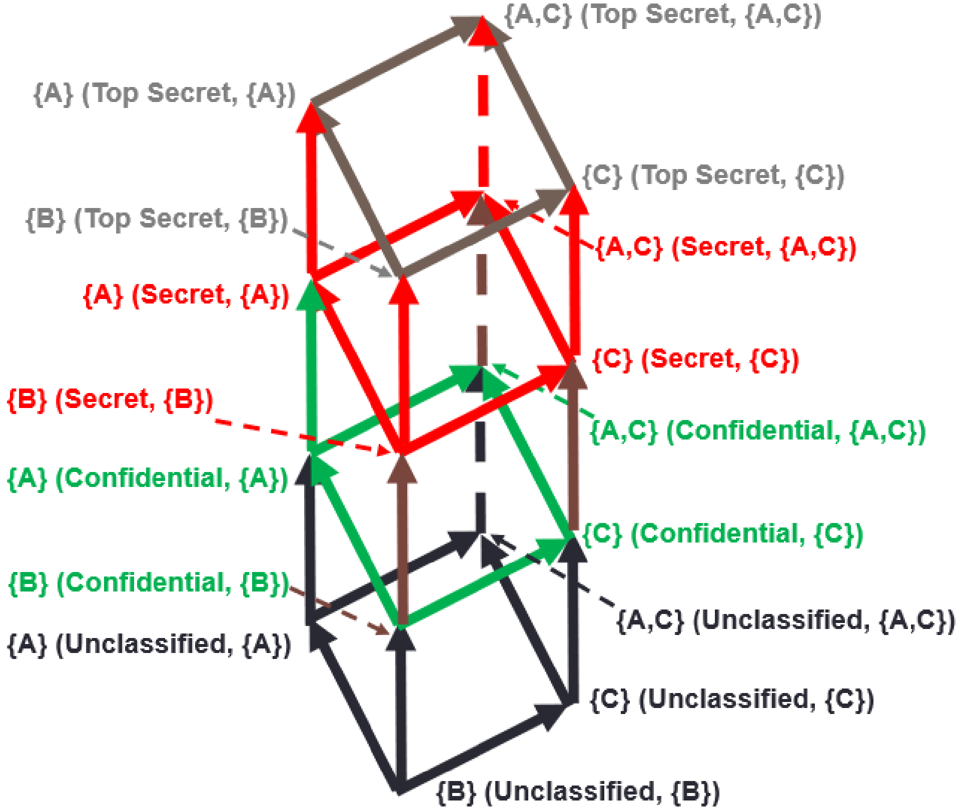
a) Alice, cleared for (Top Secret, {A,C}), wants to access a document classified (Secret, {B,C}).

Use BLP

- Alice can read C (ss-property)

  ss-property (simple security property) states that a subject can only read an object of less or equal security level.
- Alice cannot write to C (*-property)

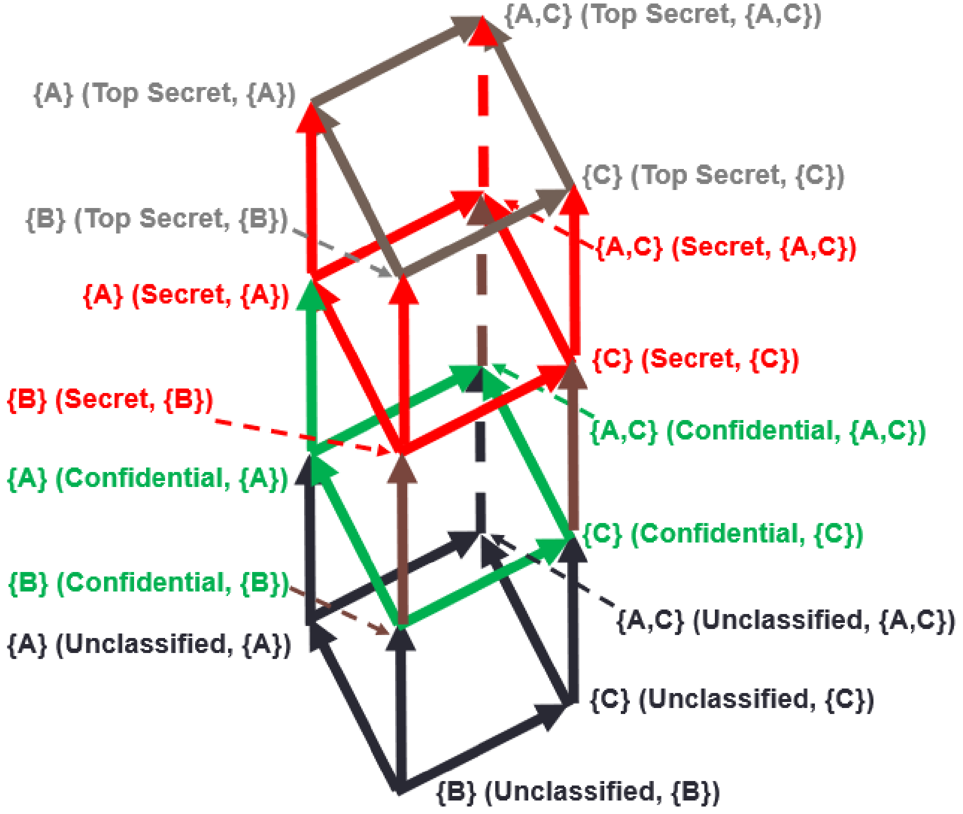
  According to ***-property**, a subject can only write into an object of higher or equal security level.
- Alice has no authorization to B.



Question 11

b) Bob, cleared for (Confidential, {C}), wants to access a document classified (Confidential, {B}).

Time allocated: 5 minutes



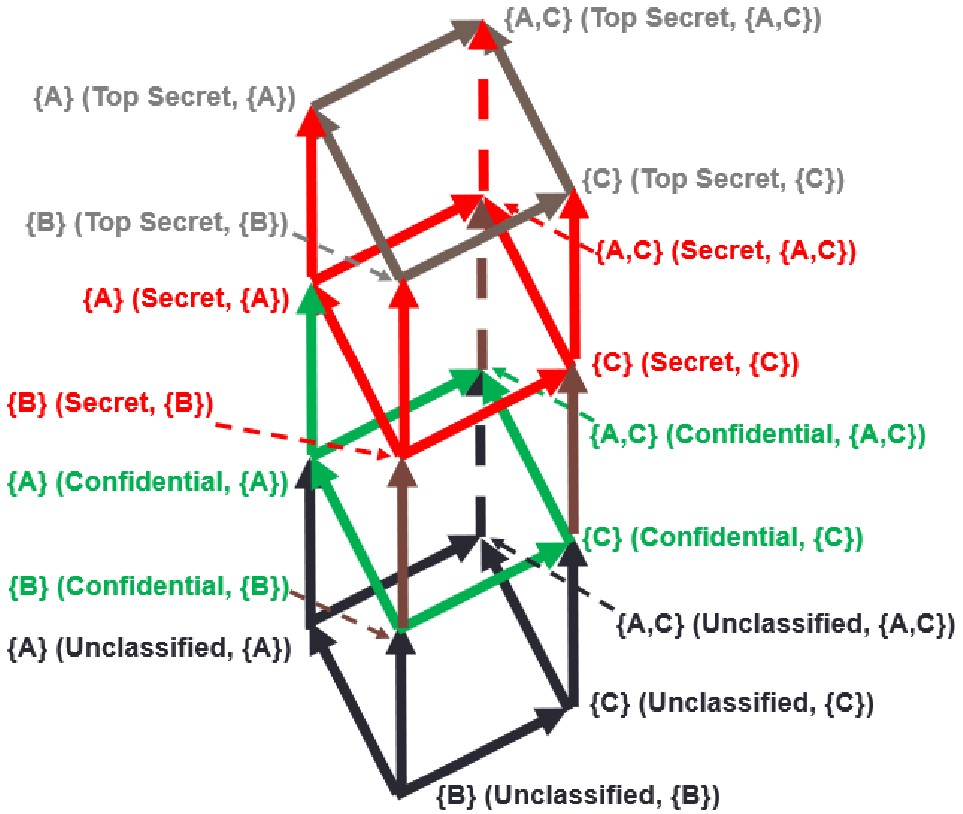
Question 11

b) Bob, cleared for (Confidential, {C}), wants to access a document classified (Confidential, {B}).

Time allocated: 5 minutes

Use BLP

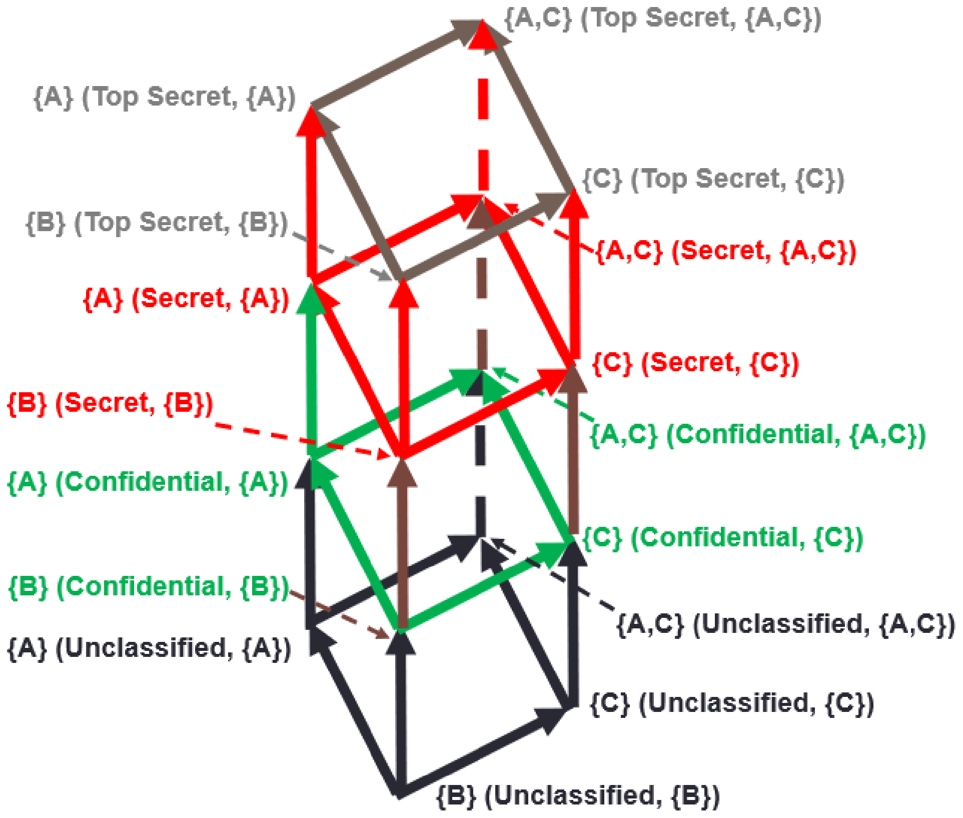
- Bob cannot access B. Even though Bob has the same clearance level as the level of the document, but Bob has no access to category {B} objects.



Question 11

c) Chris, cleared for (Secret, {C}), wants to access a document classified (Confidential, {A})

Time allocated: 5 minutes



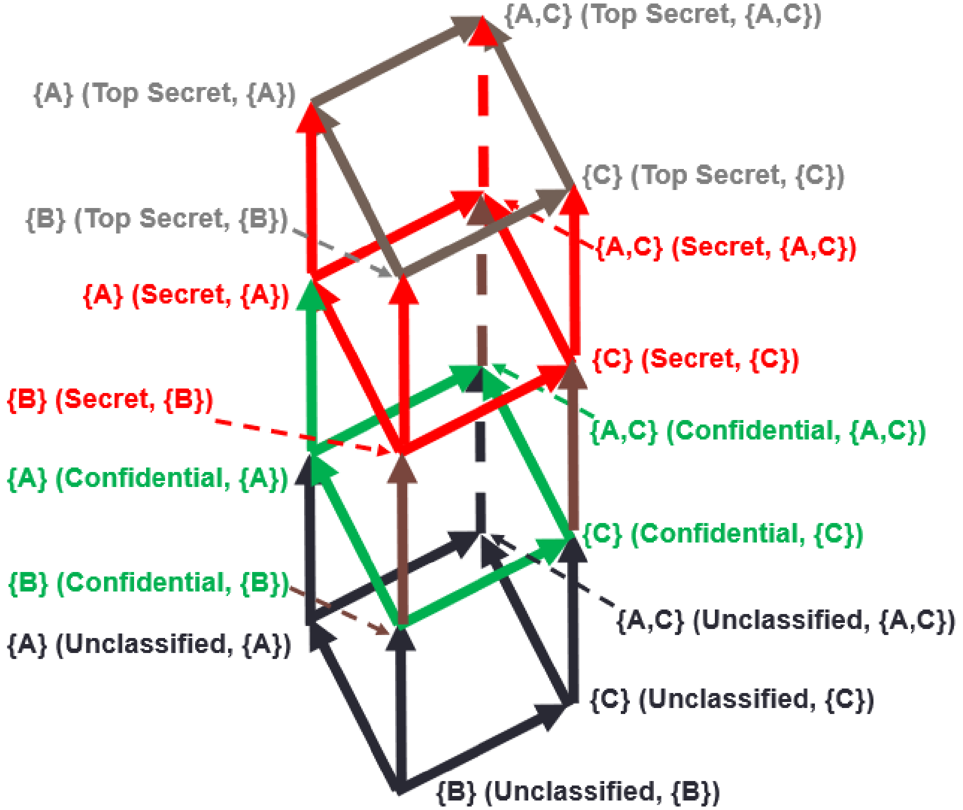
Question 11

c) Chris, cleared for (Secret, {C}), wants to access a document classified (Confidential, {A})

Time allocated: 5 minutes

Use BLP

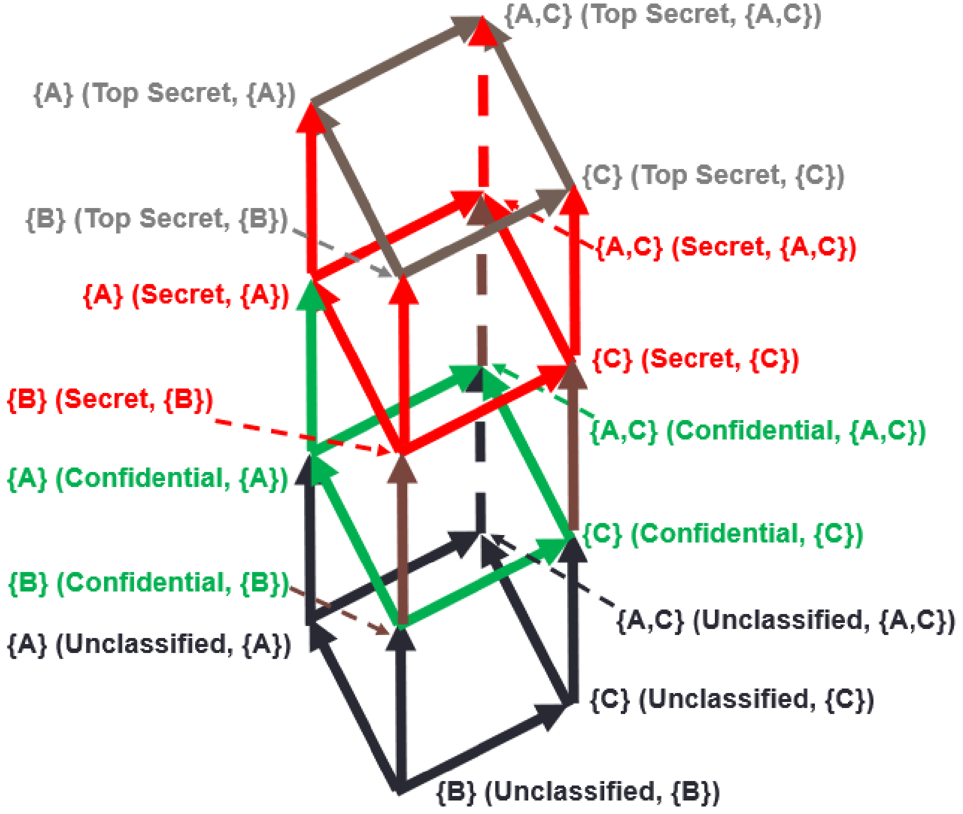
- Chris cannot access A.



Question 11

d) Dan, cleared for (Top Secret, {A,C}), wants to access a document classified (Confidential, {A})

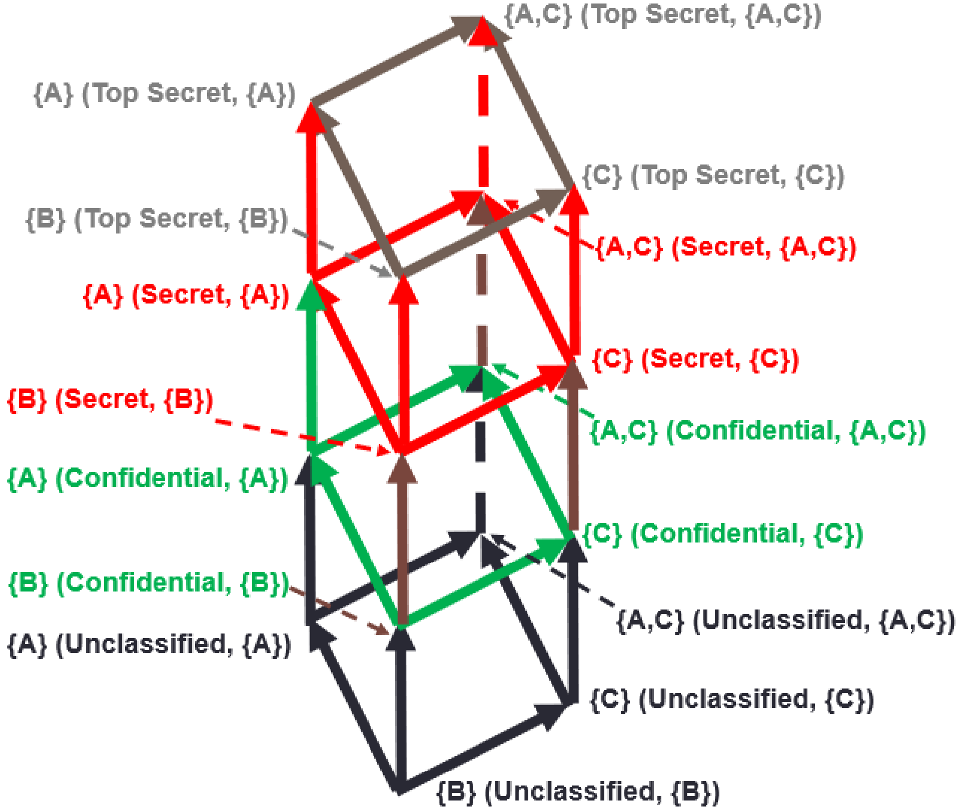
Time allocated: 5 minutes



Question 11

d) Dan, cleared for (Top Secret, {A,C}), wants to access a document classified (Confidential, {A})

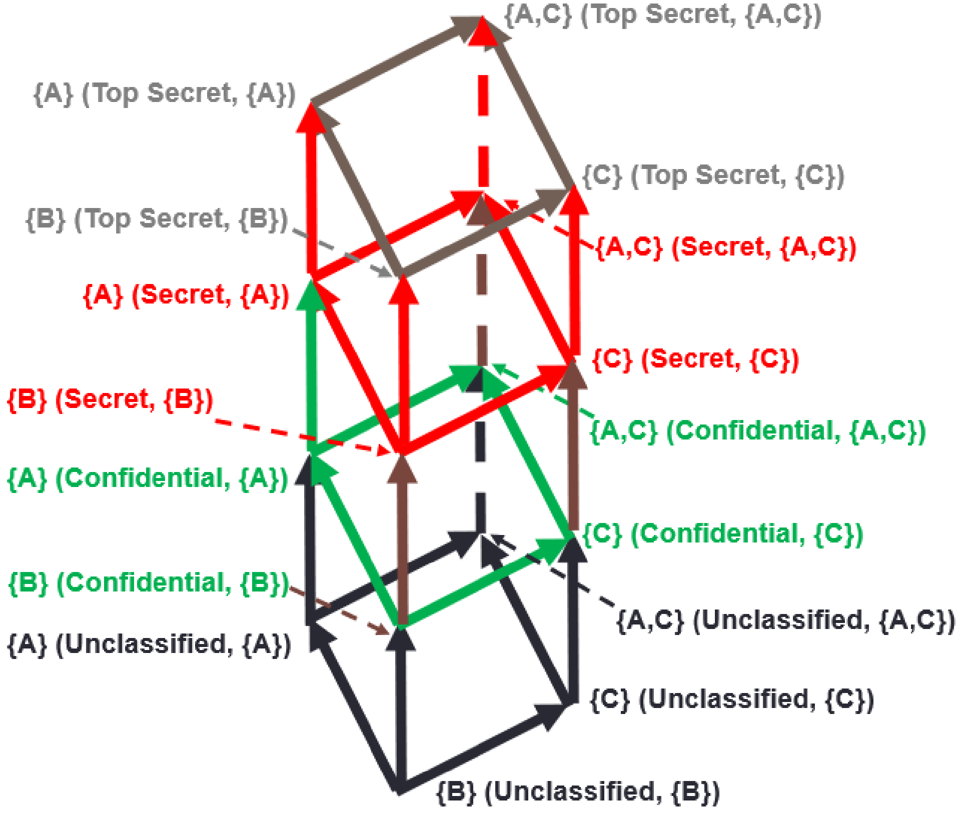
- Dan can read A (ss-property)
  ss-property (simple security property) states that a subject can only read an object of less or equal security level.
- Dan cannot write to A (*-property)
  According to ***-property**, a subject can only write into an object of higher or equal security level.



Question 11

e) Eve, who has no clearances (and so works at the Unclassified level), wants to access a document classified (Confidential, {B}).

Time allocated: 5 minutes



Question 11

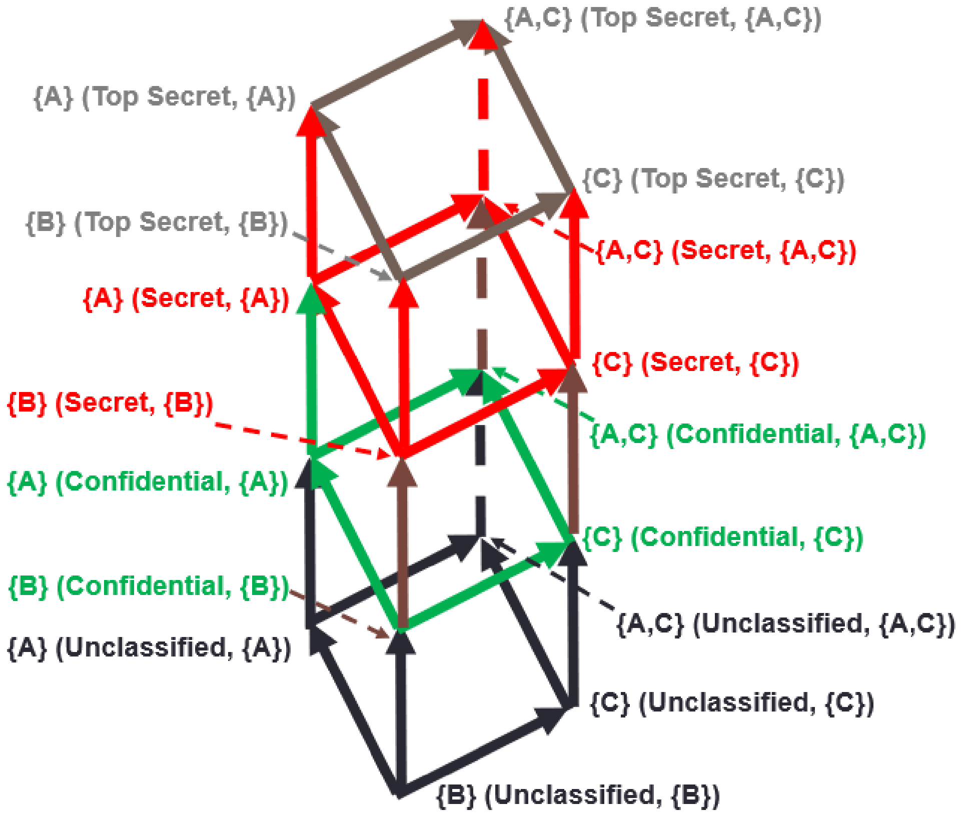
e) Eve, who has no clearances (and so works at the Unclassified level), wants to access a document classified (Confidential, {B}).

Use BLP

- Eve can write to B (*-property)

  According to ***-property**, a subject can only write into an object of higher or equal security level.
- Eve cannot read B (ss-property)

  ss-property (simple security property) states that a subject can only read an object of less of equal security level.



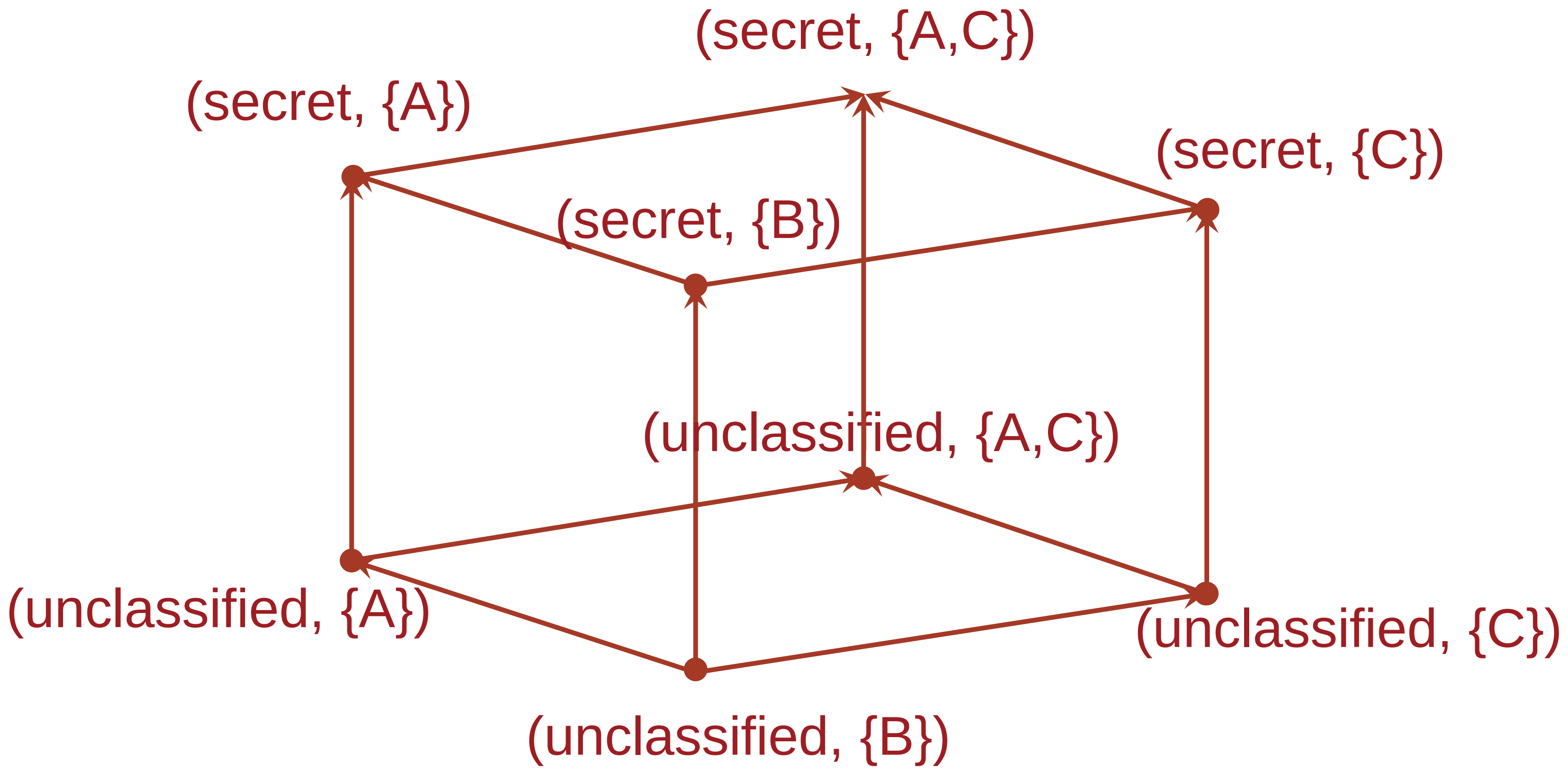
Question 11

Let us simplify the situation to classifications Secret and Unclassified and categories A, B, and C. Is it possible to have a partial order on the relationship between levels if we consider the composite categories like {A,C} to imply A or C?

Question 11

- It is not possible. The explanation are described as follow.
- In multilevel security (MLS), access control is based on a partial ordering (lattice) of security.
- Generally, categories as well as the security clearance are used in lattices.
- A, B, and C are different categories of objects.
- Secret and unclassified are security labels, and a pair (unclassified, {A}) indicates a security clearance on a classification of object.
- The partial ordering ≤ is defined by (unclassified, {A}) ≤ (secret, {A}) if and only if unclassified ≤ secret and {A} ≤ {A}.

Question 11

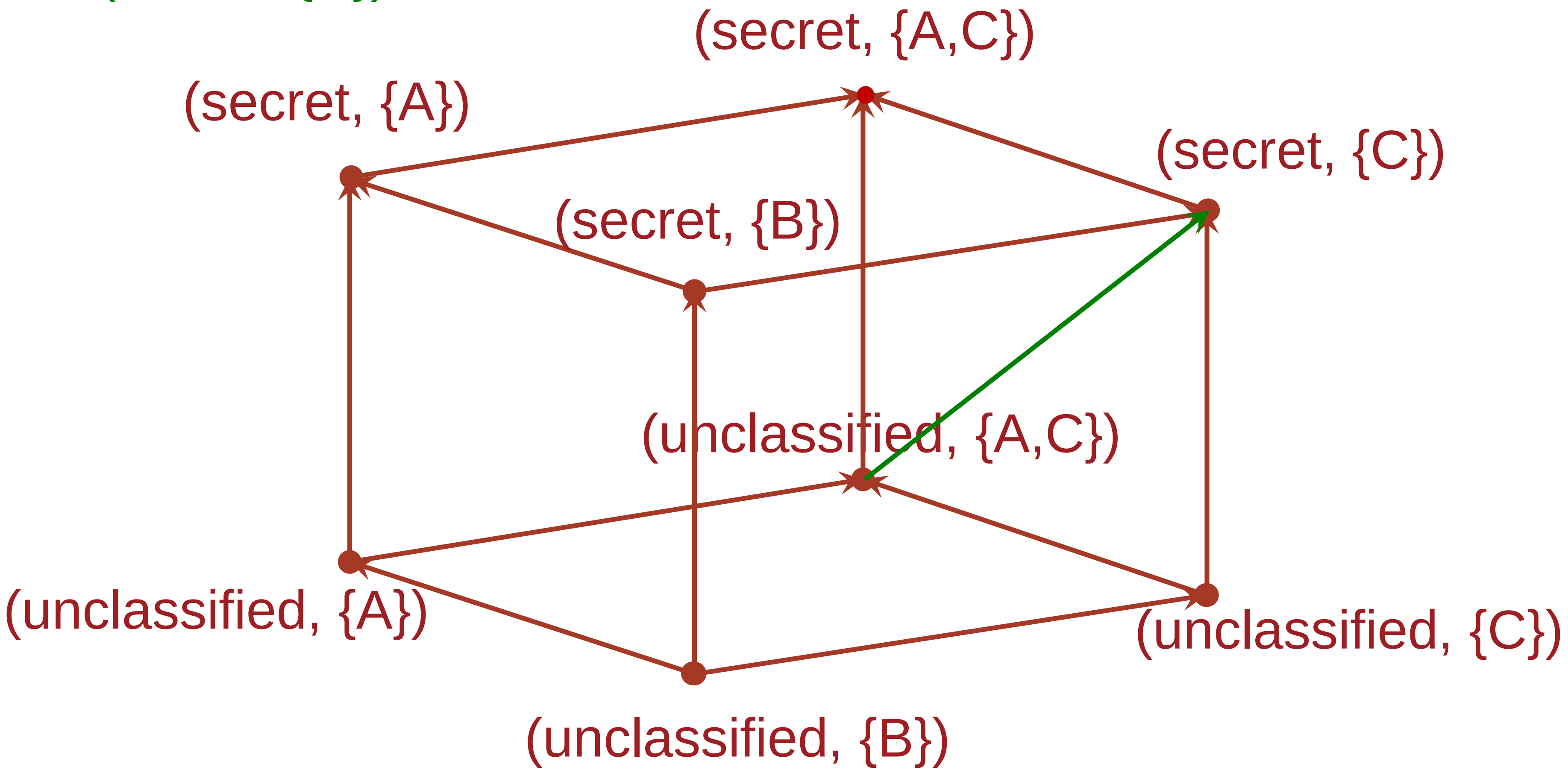

(secret, {A,C})
(secret, {A})
(secret, {C})
(secret, {B})
(unclassified, {A,C})
(unclassified, {A})
(unclassified, {C})
(unclassified, {B})

Question 11

- From the lattice shown above, the relations of the various (security clearance, categories) pairs are as follows:
  - (unclassified,{B}) ≤ (unclassified,{A})
  - (unclassified,{B}) ≤ (unclassified,{C})
  - (unclassified,{A}) ≤ (unclassified,{A,C})
  - (unclassified,{C}) ≤ (unclassified,{A,C})
  - (unclassified,{A}) ≤ (secret,{A})
  - (unclassified,{B}) ≤ (secret,{B})
  - (unclassified,{C}) ≤ (secret,{C})
  - (unclassified,{A,C}) ≤ (secret,{A,C})
  - (secret,{A}) ≤ (secret,{A,C})
  - (secret,{B}) ≤ (secret,{A})
  - (secret,{B}) ≤ (secret,{C})
  - (secret,{C}) ≤ (secret,{A,C})

Question 11

- Assuming that we have a relation (unclassified, {A,C}) ≤ (secret, {C})



Question 11

- If we consider the composite categories {A,C} to implies A or C, then according to the relation (unclassified, {A,C}) ≤ (secret, {C}) means a subject whose clearance is (secret, {C}) is able to read the object in category A because (unclassified, {A,C}) implies it is enough to have (unclassified, {A}) or (unclassified, {C}) to read this.

Question 12

The primary property provided by the Chinese Wall model of access is neither integrity nor confidentiality. Rather it is conflict of interest. Explain what is meant by a conflict of interest. Give an example to show how adding triplets to the current access set changes the access control matrix.

Question 12

- Conflict of interest of Chinese Wall Model is a concept that is used in Chinese Wall Model to control access to objects that might conflict the interest of the subject; it is similar to the code of conduct applies in financial and legal professions.

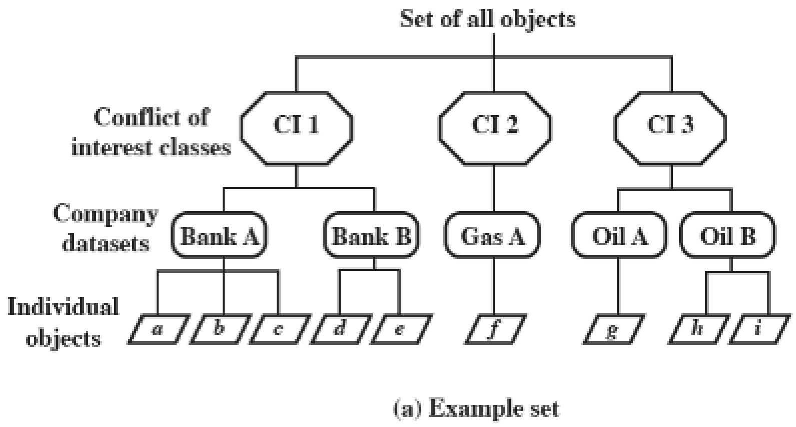
- The main idea in this concept is to use a logical wall (barrier) to prevent a subject that accesses data from one side of the wall from accessing data on the other side which has conflict of interest.

Question 12

- CWM consists of the following components:
  - **Subjects**: Active entities that may wish to access protected objects; includes users and processes
  - **Information**: Corporate information organized into a hierarchy with three levels:
    - Objects: Individual items of information, each concerning a single corporation
    - Dataset (DS): All objects that concern the same corporation
    - Conflict of interest (CI) class: All datasets whose corporations are in competition
  - **Access rules**: Rules for read and write access

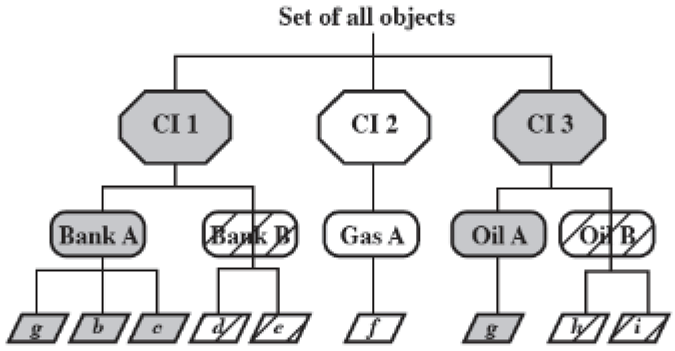
(William Stallings and Lawrie Brown, Computer Security: Principles and Practice, Pearson Education, 2008, page 317-319).

Question 12

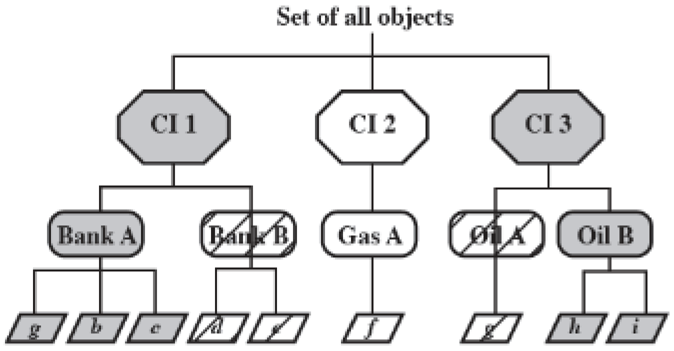


(a) Example set

In the diagram, there are datasets representing banks, oil companies, and gas companies. All bank datasets are in one CI, all oil company datasets in another CI, and so forth.



(b) John has access to Bank A and Oil A



(c) Jane has access to Bank A and Oil B

(William Stallings and Lawrie Brown, Computer Security: Principles and Practice, 3ed Pearson Education, 2015, page 474-475).

Question 12

- The basis of the Chinese wall policy is that subjects are only allowed access to information that is not held to conflict with any other information that they already possess.
- Once a subject accesses information from one dataset, a wall is set up to protect information in other datasets in the same CI.
- The subject can access information on one side of the wall but not the other side.
- When additional accesses are made in other CIs by the same subject, the shape of the wall changes to maintain the desired protection.
- Each subject is controlled by his or her own wall - the walls for different subjects are different.

Question 12

- To enforce the access control described earlier, two rules are needed:
  - **Simple security rule:** A subject S can read on object O only if: O is in the same DS as an object already accessed by S, **or** O belongs to a CI from which S has not yet accessed any information. Figure b and c in slide 72 illustrate the operation of this rule.
  - ***-property rule:** A subject S can write an object O only if: S can read O according to the simple security rule, **and** All objects that S can read are in the same DS as O. Thus, in the figures shown in slide 72, neither John nor Jane has writen access to any objects.

Reference

- William Stallings, Lawrie Brown 2015, *Computer Security: Principles and Practice*, 3ed, Pearson Education Limited, Edinburgh Gate, England
- Matt Bishop 2002, *Computer Security: Art and Science*, Addition-Wesley
- Dr Luke McAven, Lecture notes, 2015 Session 4