COMP3632 A3

20306527

SIM, Kyu Doun

kdsim@connect.ust.hk

1. Data Privacy

(a)

K-anonymity	Yes. Company could anonymize the data.
Differential Privacy	No. There is hard to guarantee that queries will be differentially private.
SMC	No. The query submitting it would be a private query of my location information.
PIR	No. The location data has great privacy. If a user is allowed to download such database the query it personally, he or she could compromise privacy.

(b)

K-anonymity	No. Data is meither private nor sensitive.
Differential Privacy	No. Cannot ensure that the query will be differentially private.
SMC	No. Data is not private. Only query is private.
PIR	Yes. The data here is a nutritional information of foods. Therefore users can download the data and privately query it.

(c)

K-anonymity	No. It is not sure what attributes to anonymize, attributes might be too imprecise to be anonymized.
Differential Privacy	No. The query is private and cannot be ensured that it could be constructed in a differentially private way.
SMC	No. The query is also private.
PIR	Yes. The query can be constructed in a privacy-preserving manner to stop the DNS server from knowing what you asked for. The problem states that the DNS servers will cooperate too. Therefore PIR is suitable.

(d)

K-anonymity	Yes. Sensitive information could be anonymized.
Differential Privacy	No. The query is private as it contains my personal information.
SMC	No. The query would be private therefore unsuitable.
PIR	No. The data is quite sensitive, so if a user were to download the database of the user habits, it would compromise privacy.

2 Multi-level Security Model

(a)

1. Finn tries to read the file.

Failed operation. Finn can only read all Finance and Basic files.

Level: Stays in Accounting

2. Batu tries to write to the file.

Successful operation. Batu is a lower security subject and can write on higher security level objects.

Level: Stays in Accounting

3. Finn tries to write to the file.

Successful operation. According to the definition of the High-water mark Bell-Lapadula Model, "*If a person at security level A writes to a file at security level B, then the security of the file must be raised to a level that is allowed to read both A and B; if there are multiple such levels, choose the lowest one."* **Therefore, when Finn writes to the file, the security level elevates to Management.**

Level: Elevated to Management

4. Ace tries to write to the file.

Successful operation. The security level of the file stays at the Management level.

Level: Management

5. Ace tries to read the file.

Fails. Ace can only read files on Accounting and Basic level.

Level: Management

6. Ma tries to read the file

Successful operation. Ma can read all files. Stays at management.

Level: Management

2 (b)

Proof: After Finn writes a file, regardless of its original security level, Acc can never read the file.

If Finn was able to successfully write on a file, this means the file was either on Finance level or placed in a lower security level that Finn was allow to write on. According to the high-water mark Bell-Lapuadula, the file's security level must be raised to a level that can read files on the Finance level and the level that the certain file is under. If there are multiple levels that can achieve this, it will be elevated to the lowest possible level.

Therefore, regardless of any file, being it on the Basic, Accounting, or Finance, if Finn successfully writes on the file, the level of security will elevate to each Basic to Finance, Accounting to Management, and Finance will stay in Finance. No matter what operations come or go after, Ace cannot read the file since the file is under either the Finance level or Management level; both levels that Ace cannot read. For the case when Finn writes a file on the Management level, the security level will stay in Management and Ace originally cannot read files on the Management level.

2 (c)

1. Finn tries to read the file.

Failed operation. Finn can only read all Finance and Basic files.

Level: Stays in Accounting

2. Batu tries to write to the file.

Successful operation. Batu is a lower security subject and can write on higher security level objects.

Level: Stays in Accounting

3. Finn tries to write to the file.

Successful operation. According to the definition of the High-water mark Bell-Lapadula Model, "*If a person at security level A writes to a file at security level B, then the security of the file must be raised to a level that is allowed to read both A and B; if there are multiple such levels, choose the lowest one."* **Therefore, when Finn writes to the file, the security level elevates to Management.**

Level: Elevated to Management

4. Ace tries to write to the file.

Successful operation. The security level of the file stays at the Management level.

Level: Management

5. Ace tries to read the file.

Successful operation. Ace can read all Accounting, Basic, and Management files.

Level: Management

6. Ma tries to read the file

Successful operation. Ma can read all files. Stays at management.

Level: Management

2 (d)

Let Y be the statement that "It is always true that the security of any file can only increase or stay the same, no matter what operations are done in what order".

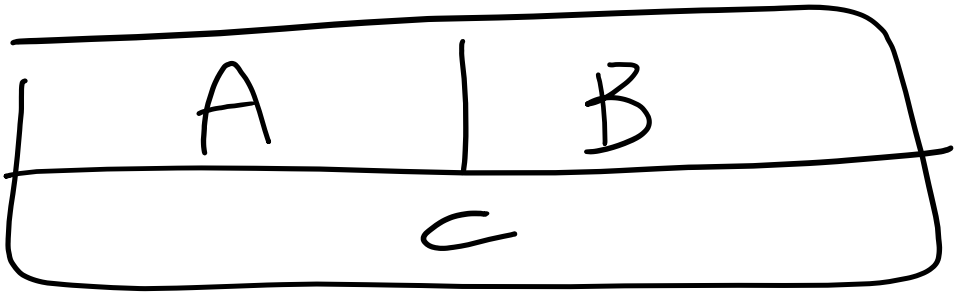
Let X be the necessary and sufficient condition for the security levels of a Multi-level Security system using High-water mark Bell-Lapadula model for Y.

To prove that it is a necessary and sufficient condition, we must prove that X -> Y and Y -> X.

Proof 1) Y -> X

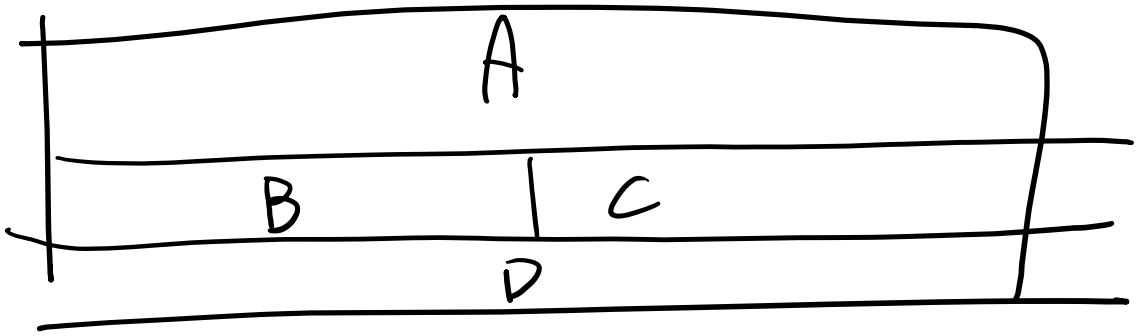
If the security of any file can only increase or stay the same, whatever operation comes before or after, the very top layer of the multi-level should be a level that can read all files. Such structure guarantees that after an operation in the high-water mark Bell-Lapadula model, the file could either stay or get elevated security status.

Let's say that the top layer has a security level that cannot read all files at any security level. We can image the following lattice structure.



Level A can read files on A and C. Level B can read files on B and C. If A writes to B, there is no higher security level than A and B, therefore there will be information leakage. The high-water mark Bell Lapadual model does not stop from A writing to B, even if the security level is not clearly defined as high or low between A and B.

If we had a following security level structure



Level B can read files on B and D, and level C can read on file C and D. If the person in B wrote to C, then the level will get elevated to A, a level that can read all files in B, C, and D.

The fact that all files' security level can stay or get elevated, guarantees the fact that the top most layer can read all files, meaning for any operation, there is a security level that a file can go regardless of the operation.

Proof 2) X -> Y

If the top most layer can read all files, this means that any file in any level can either get its level elevated or remained the same. If the person in the top most writes to any lower level, the security level of the file should get elevated to the top most layer. The person in the top most layer can read or write any files on its own level, therefore the security level will remain. If this is true for the top most layer, it will be true for any lower level than that because we can construct a new multi-level security with having the top most layer that can read all files lower than it.

3 Bitcoin

(a)

1 block per 10 minute = 1 block per 600 seconds

number of bitcoins per second is 6/600 = 0.01.

Total running cost is $\frac{10^{20}}{10^{16}} * \$\ 0.002 = \$20$ per second.

Therefore, $20/0.01 = $2000 is the minimum price of Bitcoin to cover the running costs.

(b)

1MB = 1048576 Bytes

1MB / 166 bytes = 6316.72289157, round it to 6316 transactions per block.

Then divide it by 600 seconds per blocks = 10.52666… transactions per second.

Therefore 10.53 transactions per second.

(c)

Number of bitcoins per second is 3/600 = 0.005, so 0.005 bitcoin per second is lost.

Monetary loss needed to be covered by transaction fees = $2000∗0.005 = $100 per second.

Therefore, the mean transaction fee needs to 100/10.53 = $9.496676 = $9.50

(d) 152 devices.

Assume that x machines are needed.

The total share of the bitcoin revenue will be $\frac{x*10^{16}}{x*10^{16}+10^{20}}$

The number of bitcoins per second is 0.01 and the price is fixed to $2,500.

Therefore, for each second, the revenue would be 0.01 * 2500 = $25 per second.

Multiply the (share of revenue) * (revenue per second) will give

$$\$25 * (\text{ratio above}) = 25 * \frac{x*10^{16}}{x*10^{16}+10^{20}}$$

Profit is revenue minus loss. Loss in this case would be the cost of running the miner, which is $ 0.002 per second per machine.

$$25 * \frac{x * 10^{16}}{x * 10^{16} + 10^{20}} - 0.002x$$

There are about 365 * (60 * 60 * 24) = 31536000 seconds in a year.

$$31536000 * \left(25 * \frac{x * 10^{16}}{x * 10^{16} + 10^{20}} - 0.002x\right)$$

Each machine costs $8000.

$$31536000 * \left(25 * \frac{x * 10^{16}}{x * 10^{16} + 10^{20}} - 0.002x\right) - 8000x$$

We are therefore solving the following inequality:

$$31536000 * \left(25 * \frac{x * 10^{16}}{x * 10^{16} + 10^{20}} - 0.002x\right) - 8000x \geq 1000000$$

The solution for above is

$$151.758 \leq x \leq 927.148$$

Therefore, the minimum required number of machines are 152 machines.