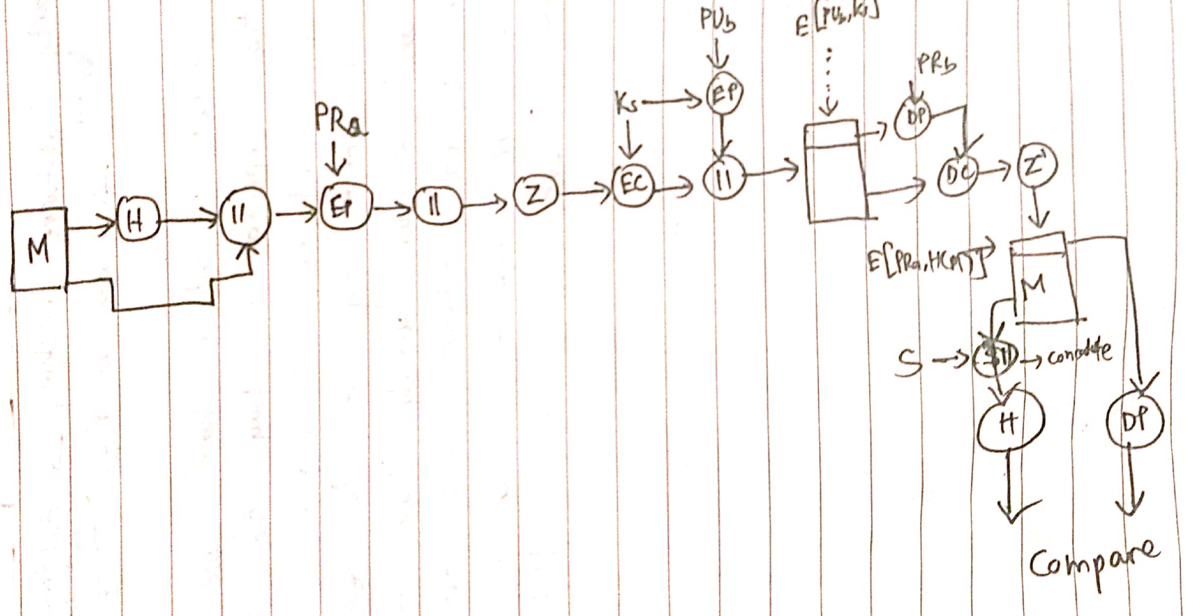
Q1 a)

a = sender
b = receiver

16L-4005 CS-B



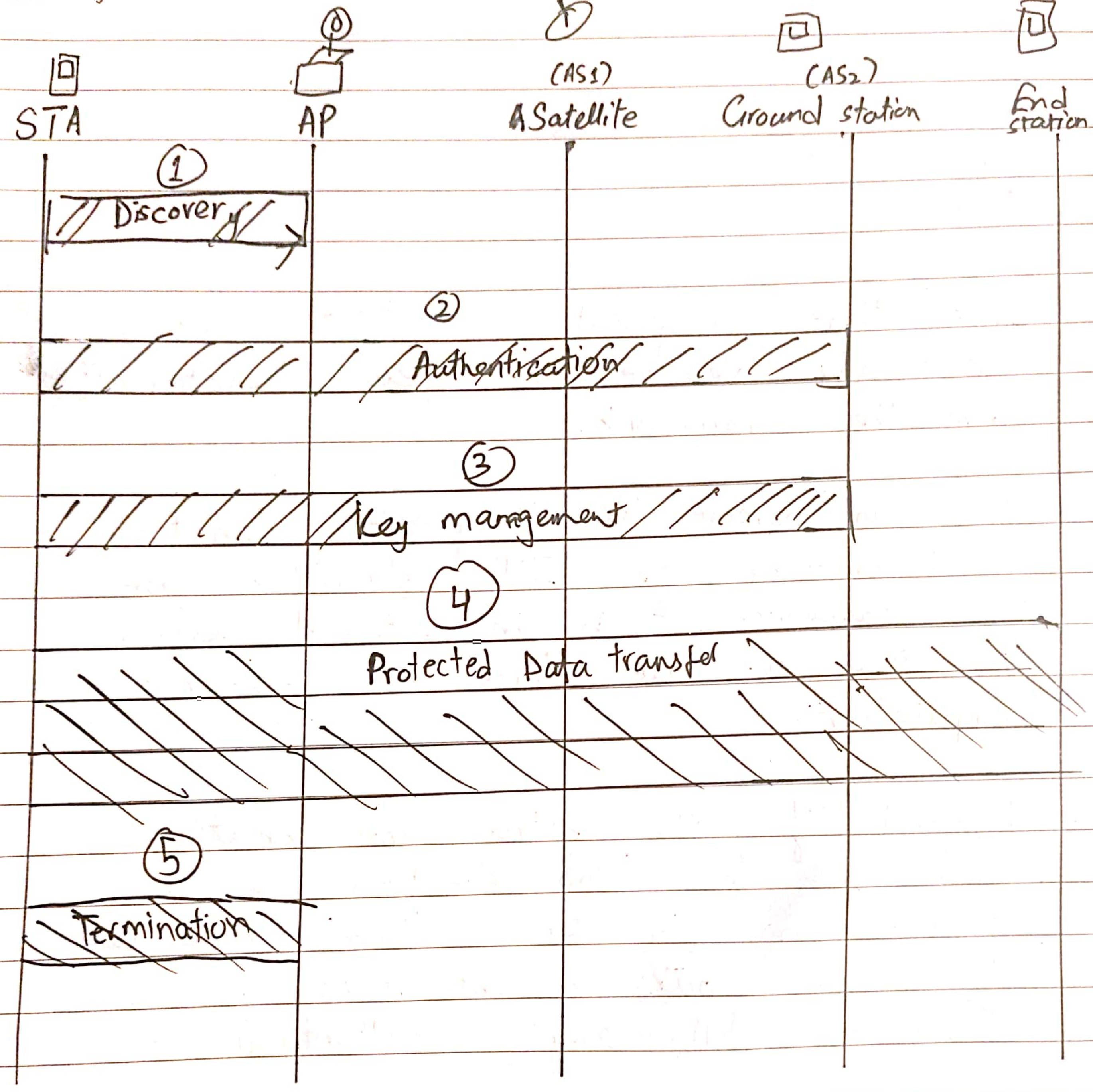
H: Hash function
Z = ZIP algorithm

Q1a:

16L-4005 CS-B

Date 30th June, 20

Q1 b):



RSA algorithm is used for the message encryption and digital signature processes involved for message authentication and confidentiality. During 2 STA and ground station prove their identities to each other. The AP blocks non authentication traffic b/w both entities unless transaction is successful. AP only forwards traffic to ground station.

Q1c:

Since there computation restrictions now, one may remove a few functionalities from the framework.

→ We can remove the compression function by removing the ZIP algorithm from the framework. It is an optional service and since we are tight on resource computation.

→ Removal of the extra concatenations used for the Hash functions in our solution in 1a since those were used to enhance our authentication process. There is still ample authentication.

→ Change of algorithm. Since RSA uses a lot of memory and we already are short on computation. Use of ~~AES~~ ~~or~~ DES or AES can save resource computation since the algorithms use less memory and are more optimal due average entropy value

→ Removal of decompression algorithm

16L-4005 CS-B

to save computation. Same reason as the reason ~~for~~ like compression algo.

→ Less number of encryption rounds to be done to increase efficiency

→ Combination of algorithms used for encryption may just increase computational problems.

5 16L-4005 CS-B

Date 30th June, 20

Answer (s) :

Q2a.

Planning

1) Security awareness training
- The sessions allow the project members to have a basic knowledge about the security threats and creates a set mindset for the future activities.

2) SDL discovery
- By having a thorough plan of the SDL, it ensures the fact that the team is going to tackle the issues as early as possible.

Requirement

3) Specify security requirements.
Having a list of security requirements helps to easily identify and fix the suspected non complicant areas of the project.

4) Specify abuse cases
It helps one to think like an attacker and helps the members to stay a step or two ahead of the attack.

5) Threat identification and Risk analysis
It gives a better idea of the threats on how likely it is going to succeed.

Date 30th June, 20

6) Select appropriate security mechanisms
This is to cater the security requirements making sure it does not add a new vulnerability giving a false sense of security

7) Asses security index
It gives the level of security of the requirements one has proposed. Hence, it gives a clear understanding of the existing and potential vulnerabilities

8) Perform inspection
It gives a better idea of the whole requirements that include functional, non-functional and security requirements. It might tell if there's anything still need to be done.

Design

9) Architecture Risk analysis
It ranks the technical risks on the basis of severity. It lets the team know about certain flaws that may allow attacks to succeed with help of dependency and attack analysis.

10) Threat modeling
It uncovers the potential threats early on decreasing the associated costs and providing a basis for the future responses.

11) Perform design inspection
It helps to remove vulnerabilities and makes the design more structured.

12) Third-party software tracking
Regular checks of third-party software helps to target areas threatened by compromised components and fills the loopholes.

Implementation

13) Follow secure coding guidelines / standards
It makes the project more authentic as it is following a set ~~rate~~ no. of parameters or rules defined. It has to be up-to the standards mentioned.

14) Select secure programming language
It allows appropriate level of functionality according to the project requirements. Usually, high-level languages are preferred as they are more secure than the lower one.

15) Code review
It helps the developers to flag and fix potential issues before diverting the attention to something else.

Testing

16) Penetration testing
It gives idea of certain attack scenarios that might have been overlooked by the project team.

17) Fuzz testing
It improves protection from attacks that use malinformed inputs like SQL injection

16L-4005 8

18) Testing plan formation
It provides the basis for formally testing any of the software product.

19) Make test cases
Testers have a clear picture of what needs to be achieved. It helps in convincing quality of product

Maintenance

20) Have a response plan
It gives idea to the incident team of the security breaches that may occur It helps in repairing the breaches if executed correctly.

21) Environment management
Security monitoring covers the entire system therefore it improves overall security of the project

22) Perform rework
It protects the project or app from newly discovered vulnerabilities by reviewing the previous phases.

Date

Q2b):

It is a clear fact that removal of bugs and errors or any kind of vulnerabiliy ~~should be~~ in the earlier phases of the SDLC as it is more cost effective. The one I may skip are:

- Fuzz testing → Use of automated tools are expensive
- Third-party software tracking → Additional need might just increase our cost
- Penetration testing → Same as fuzzing. Explained below.

Since the budget is cut in half, we may not have the expense to buy the static tools for the aforementioned techniques / activities. Also, it is more expensive to remove defects later on and as we already have budget constraints. Extra software costs are just going to increase our ~~burden~~ financial burden even more. Therefore, these additional set

Scanned with CamScanner