16-4284 (NS-B)

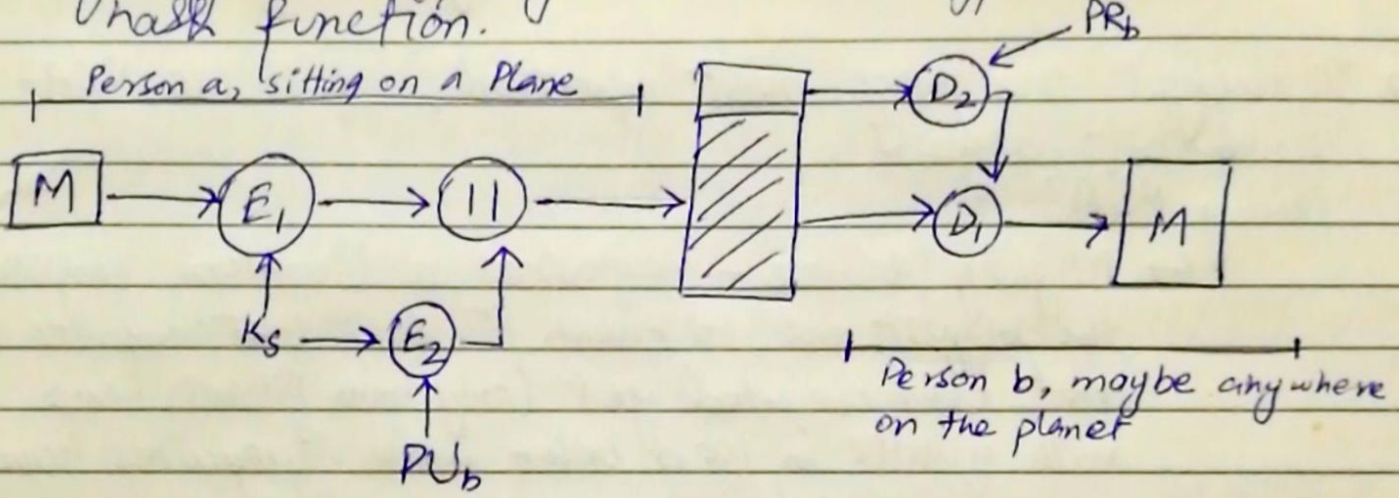
Mon Tue Wed Thu Fri Sat Sun

Date 30/6/20

Q1a.

1) Confidentiality and authentication

To ensure confidentiality, will use a combination of symmetric/Asymmetric encryption with a hash function.



→ Basically, here we are encrypting the message using a randomly generated 256-bit key using a True-random no. generator to ensure maximum protection and reduce guesses.

→ It will Symmetric key encryp/decryp so have to encrypt the randomly generated key to protect msg. Using the Public key of recipient, we'll encrypt the session key. In the end, combine them & send it.

Important thing is encryption algos used so, here are the details:

→ To encrypt message, AES is used
Why AES?
↳ Faster in processing.
↳ Less prone to cryptanalysis/attacks compared to other symm. encryp algos
↳ Allows to use various length keys.

→ To encrypt the key, most important, we'll use RSA-2048-bit algo.
Why RSA?
↳ Mighty secure - requires a Quantum computer, the biggest one, to break it. And that computer is not even created yet (only have Quant. Comp. with qubits in 100 or below while it requires 4000 or so qubits in a quantum computer.
↳ Only the person having the private key will be able to decrypt it.

Authentication:

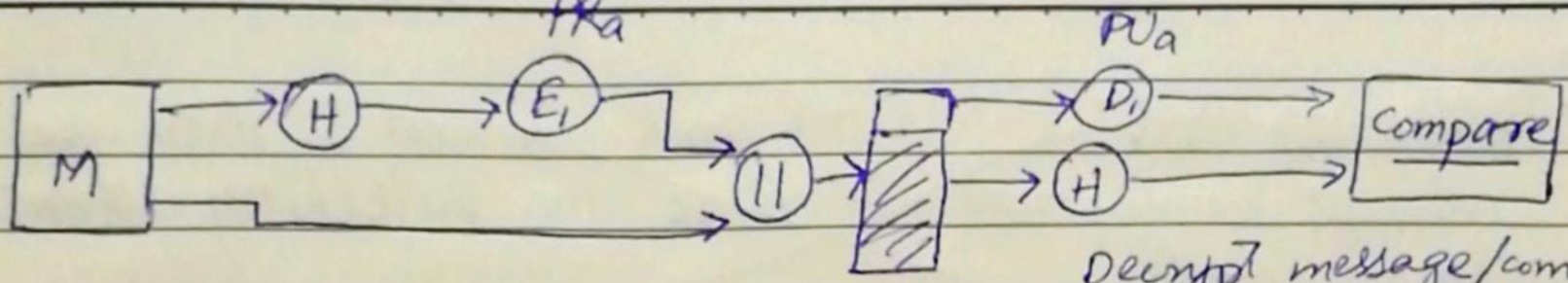
For authentication, using the concept of hash and original data comparison.
→ In this, using SHAKe-256 hashing algo to get the 256-bit hash of the message.
→ Using RSA to encrypt the hash generated.

The process: Send message hash encrypted along with original data. Then do reverse computation on other side & compare!

16-4284 (B)

Mon Tue Wed Thu Fri Sat Sun

Date ___/___/___



Why SHAKE-256? (Based on SHA-3)

→ Allows variable bits in output. (Long output)
→ Allows 256-bit security against the attacks like preimage, 2nd-preimage and collision. SHA-3 provided 128-bit security against collision attacks.
→ Impossible to have another hash like we generate using SHAKE-256.

Note: We can combine both to get the required functionality.

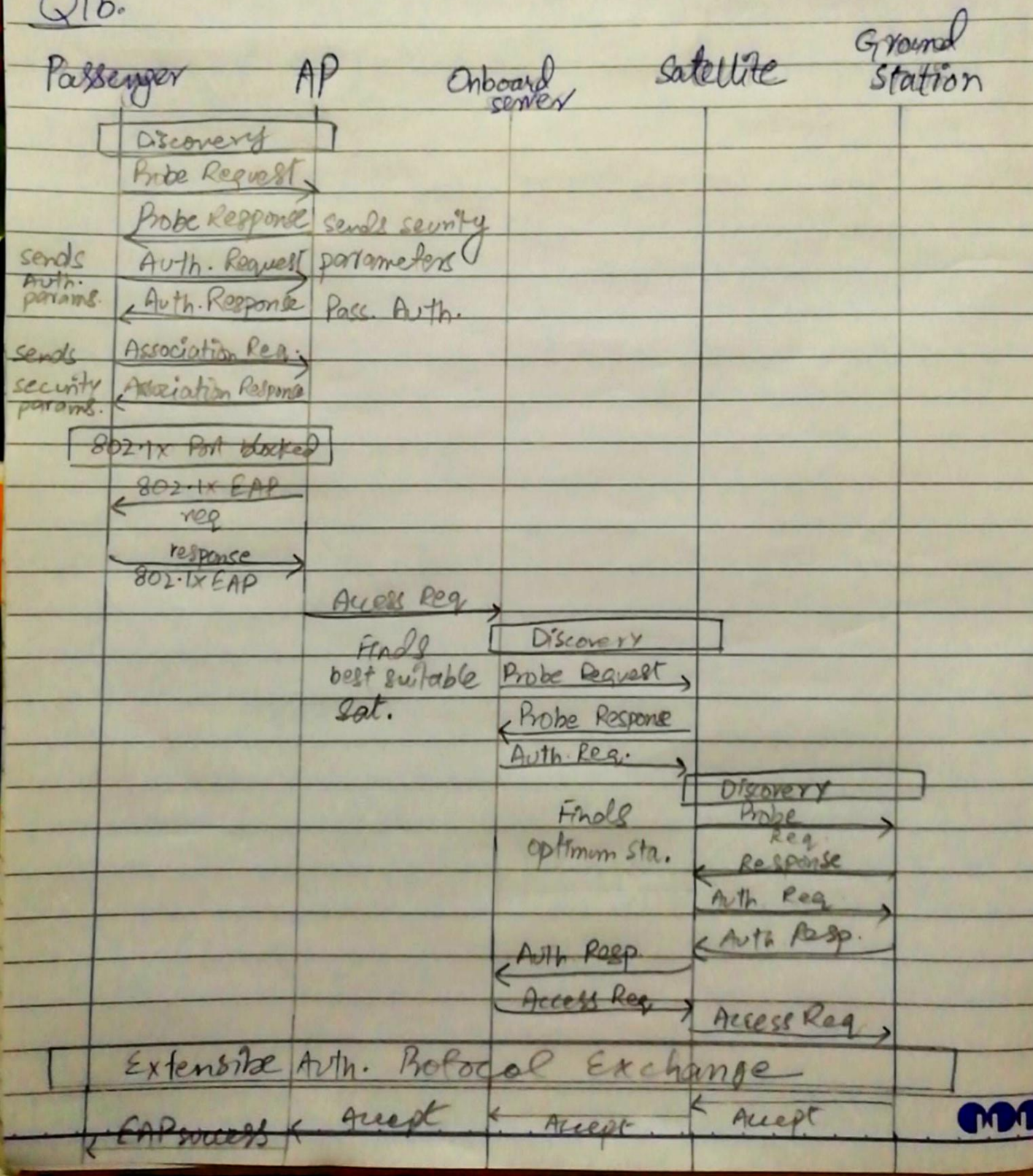
16-4284(B)

(4)

Mon Tue Wed Thu Fri Sat Sun

Date ___/___/___

Q1b.



→ Here, first phase is discovery phase in which usual security parameters' configuration will be set.

→ Secondly, AP is basically one of the routers on the plane. Assuming that, its always connected to on-board server.

→ Once the connection b/w Passenger & AP is is about to set, that is after auth. A 802.1x EAP access request is sent across.

→ This access request will make onboard sever to find best satellite and connect with it, Then the sat. will further connect/discover a suitable ground station.

↳ At the end, EAP exchange will be established, allowing the user to surf the internet.

Secure Data transfer:

Now, once connection established we can use multiple secure data transfer techniques. But we can also make the channel, via which we connected, secure. Assuming that usual WPA2/WPA protection is employed. Furthermore, VPN is being used to get a secure tunnel/channel of communication. Assuming VPN is installed on the plane.

16-4284(B)

(7) Wrongly Numbered

Mon Tue Wed Thu Fri Sat Sun

Q1C

Majorly, Q1a is dealing with heavy computations so, in Q1a:

↳ First thing we can do is to zip the message before encryption and zip hash before transferring

↳ Secondly, reduce SHAKE-256 to SHAKE-128; the output will be 128 bit hash and still secure.

↳ In RSA, used for session key encryption, probably most resource hungry, used 1024-bit algo. for encryption.

↳ Lastly, can use 128-bit session key instead of 256-bit.

Why?

→ Zipping will help reduce size and in turn transfer computations will be reduced.

→ 128-bit hashing requires less resources

→ Similarly AES is already fast, using 128-bit key will help reduce resource consumption without affecting performance.

For Q1b.

Not much computations are being used. But the data transfer encryption protocols can be kept in-check.

↳ VPN might need to be reduced address-switching but we are using in-house VPN, supposedly, so it'll not be resource hungry.

→ Also, reduce no. of Handshakes. (In some way)

16-4284(B)

(8)

Mon Tue Wed Thu Fri Sat Sun

Date ___/___/___

Q2a.

→ Requirements Engg. :

1) Define functional requirements
   ↳ In a secure dev. environment, func. req. are mandatory because it tells / helps you analyze the functions you need to secure.

2) Specify Abuse cases & model threats.
   ↳ helps you identify scenarios in which an attacker can exploit your software.

3) Analyze risks
   ↳ helps you find probability of any vulnerability in the software.

5) Define Security mechanism for the security req.
   ↳ In order to ~~be req~~ implement req., mechanisms need to be defined early to be able to cope with threats.

4) Specify security requirements
   ↳ Important to mitigate threats and define the mechanisms to stop exploits.

6) Assess Security Index
   ↳ to measure the security of the system specs. according to the standards. Can find vulnerabilities.

7) Specify functional req. for the security mechanism
   ↳ Need it to effectively implement the mentioned mechanisms.

8) Perform Inspections on the Specs
↳ check the minimum level of security is met. To find if it is acceptable or not

→ Design

9) Make a detailed design.
↳ to get the high-level idea of the system.

10) Inspect design
↳ In order to find loopholes in the system.

11) Remove vulnerabilities
↳ to get better security of the system.

12) Assess the design-level
↳ important because we need to match the security-level req. with design.

13) Design a security monitor
↳ helps you find any attack being perform on your system / informs of any security breach.

→ Implementation

14) Select a Programming language
↳ Should select secure language because it helps us reduce vulnerabilities. e.g. → high-level languages.

16-4284 (B)

(10)

Mon Tue Wed Thu Fri Sat Sun

Date ___/___/___

15) Follow standards & guidelines
↳ Defined by state-of-art security org., helps you get rid of silly design/coding mistakes, which usually being done by people.

→ Assurance

16) Testing
↳ Different types of testing helps in finding vulnerabilities and their origin.

17) Code Inspection
↳ one can find any type of vulnerability induced due to bad coding practices.

→ Maintenance

18) Observe working of the software
↳ To check for any kind of deviations from the defined spec.

19) Find vulnerabilities based on 18).
↳ helps you make the system secure and less exploitable; less loopholes.

20) Keep observing & resolving vulnerabilities
↳ In order to keep the system performing at its best without any major flaw.

Q2b.

Reduced Budget; Cut / Reduce Activities:

1) Skip (8) as budget is reduced; expectations must also be reduced so it does not matter whether or not the Standards are met

2) Skip (16); It is not as important as others we can assume that we have best possible software / Network architects.

3) Skip (13); Security monitor is not mandatory as one can also find breaches through static analysis not as efficient though.

4) Skip (15); we can assume that we have the best coders / devs and they do not need any guidelines.

5) Skip (20); reduced budget so cannot continually monitor the security. Also, we can just resolve any issue whenever it arises.

Rationale:

↳ Since budget is limited; security-level cannot be that good.