Exercises - Week 1
Principles of computer security

For any questions that ask for a justification, the justification is needed to get credit for the answer.

***Note**: Some of this exercises could have multiple acceptable answers. We are giving examples. If you have another answer and would like to check use the forum or send us an email or ask us during the next exercise session.*

1. Write a security policy for protecting examination results kept on a computer system. Your policy should at least consider the access requirements of students, lecturers, and school administrators.

This is one of the very open questions. This answer is a possible one. Other answers can be correct. If you want a correction just send it via email/forum.

Principals: students, lecturers, school administrators
Assets: examination grades
Properties:

Students must be able to read their own grades, not others', to keep integrity they cannot write

Lecturers must be able to read and write all grades for their own courses (i.e., cannot read/write on other lecturers' courses)

Administrators must be able to read all courses to generate final reports. It could be argued that they also need to write (e.g., to fix grades)

2. Are these threats, harms or vulnerabilities? (justify)

2a. Thieves can enter into the lab to steal equipment
2b. Credit card numbers are stolen
2c. Users choose weak passwords
2d. The backup system stops working
2e. A hacker can install malware
2f. A botnet sends many packets to the server
2g. The students can see the exam questions before the test
2h. The cryptographic keys are 56 bits (hint: https://en.wikipedia.org/wiki/Data_Encryption_Standard#History_of_DES)

In green, the answer we were thinking about when writing the question. In blue, valid alternatives heard in class. (More alternatives may be also correct. If yours is not there and you want to check use email/forum.) In red, wrong answers.

2a. **Threat***: this sentence expresses something that can happen to the system: who can attack it and with what goal. It is a feared event.*

Harm*: this sentence describes the result of the thieves exploiting a vulnerability, entering the lab and stealing material. In other words can be seen as the result of an attack.*

2b. **Harm***: this sentence expresses the result of an attack (e.g., exfiltration of data, for instance using malware)*

Threat*: this sentence describes a feared event, that the credit cards numbers are stolen and the thieves can use them to commit fraud.*

2c. **Vulnerability***: this opens the door for an exploit (guessing the password is easier)*

Harm*: while this sentence establishes a fact, the fact that users have weak passwords in an on itself is not a harm yet. It only becomes a fact if an adversary exploits the vulnerability.*

2d. **Harm***: the sentence describes damage that happened on a system as a result of an attack (e.g., denial of service on the backup server).*

Vulnerability*: the sentence describes a situation, when the backup is down, that can be exploited by an adversary to delete or modify files in the system without the possibility to recover them.*

*2e.***Threat***: the sentence describes a feared event. Who might attack the system and how would this attack be performed.*

2f. **Threat***: the sentence describes a feared event. Who might attack the system and how would this attack be performed.*

2g. **Harm***: the sentence describes a damage made to the system as a result of an attack (e.g., students steal the professor's office keys)*

Threat*: the sentence describes a feared event, that the students may have access to the exam questions.*

2h. **Vulnerability***: the sentence describes a weakness of the cryptographic algorithm DES, namely that it uses a small key: 56 bits. Such small space can nowadays be brute forced (i.e., it is possible to find the key trying all the possibilities via exhaustive search) rendering the encryption useless.*

3. Explain why is testing hopelessly inadequate for showing the absence of bugs?

"Showing absence of bugs" implies that one is able to prove that there is ***NO*** *bug in a program. However, it is hard, and arguably impossible, to test for all possible failure conditions. Also, many problems are caused by combinations of bugs. Thus, even when one finds a problem it may not be possible to identify all the bugs that cause the error.*

As a side note (and as it is dictated by the economy of mechanism principle), the more complex a system is, the more difficult it is to identify bugs.

4. A company publishes the design of its security software product in a manual that accompanies the executable software. In what ways does this satisfy the principle of open design? In what ways does it not?

Satisfies the principle: *The publication of a manual explaining the design is indeed open, as it is possible for everyone to understand the functionality that the software is intended to provide.*

Does not satisfy the principle: *However, the software itself is not open. Thus, one cannot fully apply the Linus law: one cannot look at the code, thus cannot find bugs in the implementation. Therefore, it cannot take advantage of the positive aspects promoted by the principle of open design.*

5. A common technique for inhibiting password guessing is to disable an account after three consecutive failed login attempts.
Discuss how this technique might prevent legitimate users from accessing the system. Why is this situation a violation of the principle of least common mechanism?

In order to prevent legitimate users from accessing the system an adversary can attempt to log in with the legitimate user's username but incorrect credentials so that this user gets blocked.

This attack is enabled by the fact that legitimate users and potential adversaries share the same interface (i.e., the attack is enabled because the system does not respect the principle of least common mechanism).

Note: this does not mean that systems should be built with users having different authentication interface. It only means that the fact that they share a common interface opens the door to attacks.

6. When the execution of a piece of software overwrites memory that was not reserved for the program, is it a good idea to revert the execution and reserve more memory? What principle is this solution violating? What is the good option in this case?

No, it is not a good idea because it is very difficult to guarantee that everything in the system will roll back correctly (the more difficult as the complexity of the system increases). This violates the fail-safe default principle. The safe solution is what programs already do: stop and exit with an error to inform the user that something has gone wrong, and to avoid creating more damage in the system.

7. Think about the following physical mechanisms. Do they serve a "security" purpose? What security policy do they serve? Under what threat model are they secure? What is the Trusted Computing Base?

- **A bank vault**
- **A non-electronic car lock**

Would the car key response be different if we consider an electronic car key? how?

[Other responses may be correct - check on forum or office hours if in doubt]
Bank vault

<u>Security purpose</u>: protect money from adversaries, i. e., non-authorized users - external (thieves) or internal (employees)

<u>Security policy</u>:

- *Principals: employees, customers, externals*
- *Assets: money and other valuables (e.g., jewels, papers)*
- *Properties:*
  - *Only authorized employees can see the content of the vault*
  - *Only authorized employees can alter the content of the vault*
  - *The authorized employees should always be able to enter in the vault*

<u>Threat Model</u>:

- *The bank vault is secure in a threat model in which the adversary:*
  - *Does not have access to the vault key*
  - *Cannot break the vault lock*
  - *Cannot corrupt authorized employees*
  - *Cannot disable the vault lock*

<u>TCB</u>: the security mechanism that locks the vault, vault's walls, floor, ceiling.

Non-electronic car lock

<u>Security purpose</u>: protect the vehicle from adversaries, i.e., unauthorized users (thieves)

<u>Security policy</u>:

- *Principals: car owner, authorized drivers/passengers (e.g., family), externals*
- *Assets: car itself, valuables inside*
- *Properties:*
  - *Only authorized drivers and the car owner can drive the car*
  - *Only owner, and authorized passengers can enter in the car*
  - *The owner should always be able to enter in the car*

<u>Threat Model</u>:

- *The car lock is secure in a threat model in which the adversary:*
  - *Does not have access to the car key*
  - *Cannot break the car lock*
  - *Cannot corrupt authorized drivers/passengers*
  - *Cannot bypass the car lock*

<u>TCB</u>: the car lock is the security mechanism

Electronic car lock

We have new actors:

the lock manufacturers that could leak the key to pirate the car or if the car is modern, they can unlock remotely!

The threat model is very different! It needs to consider the new actors, but also new means of using the electronic key. For instance:

- *No remote access to the key (key relay attacks: https://www.snopes.com/fact-check/code-grabbers-remote-keyless-entry/)*
- *Cannot break the crypto in the key (http://fortune.com/2016/08/11/volkswagen-car-remote-warning/, https://www.cosic.esat.kuleuven.be/keeloq/)*

8. Is this a security problem? (justify)

a) I need to send a wireless signal in an environment where there may be obstacles (walls, rain,...)
b) I need to keep my valuable laptop in my car to go shopping
c) I need to build a boat that floats under adverse conditions (storm)
d) I need to store the secret final exam on a server open to the internet
e) I need to make sure I am talking with my lawyer over the phone
f) I inadvertently added an infinite loop and takes down my server

a) **No**. *The environment is not adversarial, does not act purposefully to break the system.*
b) **Yes**. *In this scenario there is an adversary that wants to actively and purposedly steal my laptop.*
c) **No**. *The sea and the storm are not adversarial. They do not happen at the worse moment and their goal is not to sink the ship.*
d) **Yes**. *In this scenario there can be an adversary that purposedly will try to use the fact that the server is connected to the internet to try to steal the exam*
e) **Yes**. *In this scenario there can be an adversary that tries to impersonate my lawyer to, on purpose, try to extract confidential information from the conversation.*
f) **No**. *This is a bug, the attack is not intended, it is not adversarial and not a security problem.*

Yes. *This is a bug, as such it creates a vulnerability that could be exploited by an adversary to harm my system on purpose.*

9. An ad company wants to record how many times its users open the game Sandy Clash. A privacy-conscious user installs an app that opens Sandy Clash a random number of times taken from a uniform distribution between 0 and 2 (i.e., 0 times with probability 1/3, 1 time with probability 1/3, and 2 times with probability 1/3).

The ad company, which detects the use of the privacy-preserving app receives the following uses on a week: [3,5,1,2,4,3,4].

Should the ad company believe that on average the user has open the app Mean([3,5,1,2,4,3,4])=3.14? If not, what should the ad company do?

*If the company knows there is a privacy-preserving mechanism, as an **strategic adversary**, it should take the use of the mechanism into account. That is, it should compute the average number of fake accesses and subtract it from the observed values.*

*More concretely, given the mechanism, the ad company can know that on average each day the privacy-preserving app makes 1 fake access to Sandy Clash (1.0/3*0 + 1.0/3*1 + 1.0/3*2). Thus, it should consider that on average the user has accessed 2.14 times the app.*