Question 1.1 Before computer ”machines”, what was a computer?

Machine operator was called the computer before computers were ‘machines’.

Question 1.2 Who was the first programmer?

Ada Lovelace was the first programmer. She wrote notes from summary of Babbage’s ideas which had been written by Luigi Federico Manabrea. In her notes there was a way for the first “computer” to calculate Bernulli numbers. This description itself is regarded as a first computer program, which means that Ada Lovelace was the first programmer.

Question 1.3

• What was the first programming language?

• Is it still used today? Why?

First computer language was the machine code, so it can be considered as a first programming language (but FORTRAN was the world’s first computer language to use language (e.g. English) to specify particular commands). Modern day programmers still occasionally use machine level code, especially when programming lower level functions of the system, such as drivers, interfaces with firmware and hardware devices.

Question 1.4 What was the first use of the ENIAC?

ENIAC was supposed to be used for the calculation of ballistics tables to help the artillery fire their weapons at the correct angles, but it ran its first program to make calculations in the development of the H-Bomb.

Question 1.5 Describe each of the following vulnerabilities (type of vulnerability, specific conditions to exploit the vulnerability) and explain if they can have an impact on the confidentiality, integrity and/or availability:

1. CVE-2014-0160

2. CVE-2016-9079

3. CVE-2018-20343

1. Vulnerability type – Buffer Error, this vulnerability allows remote attackers to obtain sensitive information from process memory via crafted packets that trigger a buffer over-read. This particular vulnerability don’t have impact on integrity or availability, but partially impacts confidentiality, because it allows unauthorized disclosure of information. CVE-2014-0160 does not allow unrestricted access to memory on the targeted host, a successful exploit does leak information from memory locations which have the potential to contain particularly sensitive information. It can be exploited on the TLS and DTLS implementations in OpenSSL 1.0.1 before the version 1.0.1g.

2. Vulnerability type - Use After Free (use of previously-freed memory). The use of this vulnerability (referencing memory after it has been freed) can cause a program to crash,

use unexpected values, or execute code, it can have any number of consequences, ranging from the corruption of data to the execution of code. But this particular vulnerability has no effect on integrity and availability, but allows unauthorized disclosure of information (confidentiality). This vulnerability affects Firefox < 50.0.2, Firefox ESR < 45.5.1, and Thunderbird < 45.5.1.

3. Vulnerability Type – Buffer Error. With this vulnerability you can trigger a buffer overflow and overwrite the stack. It can be used in Build Engine, in games such as Duke Nukem 3D, Shadow Warrior and Redneck Rampage.

Question 1.6

1. Does this article describe someone or an organization using non-disclosure?
2. What is ”EternalBlue”?

1. Shadow Brokers organization used non-disclosure for profit (selling information on the black market).
2. EternalBlue is a tool created by NSA, that exploited a vulnerability in Microsoft Windows, that later has been stolen and sold on black market for the profit.

Question 1.7 Under which circumstances is a vulnerability not reported to the vendor?

Vulnerability can be temporarily known only by United States Government, and potentially other partners, if it can be used for national security and law enforcement purposes, such as intelligence collection, military operations, and/or counterintelligence.

Question 1.8 What kinds of WEP flaws does the paper mention? Are these design or implementation flaws?

WEP have issues in security protocol design, not implementation, for instance 24-bit IVs are too short.

It does not meet its fundamental goals of confidentiality. It also fails to meet the expected goals for integrity and authentication. WEP uses a single shared key common to all users of a WLAN, and this common key is often stored in software-accessible storage on each device.

Since WEP does not include a key management protocol, distributing the new secret to all users is an unwieldy process. As a result, key compromises are often ignored.

The first octet encrypted under WEP is a known fixed value, and the required ciphertext packets can be readily obtained after eavesdropping on a network for a few hours, as well as encrypted messages could be modified freely by an attacker

Integrity protection for source and destination addresses is not provided, it is insecure and does not prevent adversarial modification of intercepted packets.

Question 1.9 Suppose we are 100 years into the future and we can finally write code without any implementation bug. Can an implementation of WEP be secure? Why or why not?

WEP can not be secure since it has a lot off flaws in the design itself, and it is a protocol which can be only replaced by another (more secure) protocols, In this case, cryptographic algorithms were used inappropriately: designers selected well-regarded algorithms, but used them in insecure ways.

WEP protocol can be wrapped with a set of algorithms, that can extend and improve security, but it can never be safe itself.