Sample Questions
(50.20 Security Mid-term Exam)

Question 1
Please name four different types of malware, and briefly comment on the difference between them.
Solution:

- Virus: spreads by injecting code into an executable. External force is needed for spreading.
- Worm: spreads by actively exploiting vulnerabilities on remote hosts. No need external force for spreading (automatic spreading, for example, if connected to Internet)
- Trojan: hidden functionality in shared binaries. Essentially malicious while disguised as legitimate software.
- Rootkit: Gains privileged access to a computer while actively hiding its presence.

Question 2
Please describe one attack which is prevented by the use of salt, and why. For an n-bit salt, by which factor is the cost of that attack increased?

Solution: Salt can be used to make rainbow table attack harder. Salt is usually too long to be included in table generation process. Each salt would need own table. Infeasible in practise. For n-bit salt, 2^n increase in cost for attacker.

Question 3
Please describe what social engineering attacks refer to, and give an example.
Solution: In social engineering attacks, the attacker tries to trick a victim into performing an action with malicious consequences. For example, a phone call could be used to trick someone to re-set or disclose a password.

Question 4
Consider the following minimal program on 64-bit Linux.

```
int main()
{
char input[8];
printf("Please type your name");
gets(input)
return 0;
}
```

(a) Describe what kind of attack is possible if an attacker can provide the input, for example over a network connection. Provide a short input that will reliably crash the application, and comment on why it will crash the process.

Solution: Buffer overflow attacks are possible. One example of minimal input: 'A'*24 as string. This will overwrite RBP and RIP, causing a return into AAAAAAAA and thus crash due to memory access violation.

(b) Briefly describe three common countermeasures to the proposed attack.

Solution:
NX stack: Non-executable stack .
Canary: Value stored between stack frame and stored RBP. Is verified before returning.
ASLR: Randomize locations of executable and libraries in memory.

(c) Assume that the attacker knows that the printf(const char * format) function is at memory address 0xff00aabb, and wants to call it to print a string stored at 0x55667788. Given the following shellcode (i.e. the assembly instructions) to call printf():

```
mov rdi, 0x55667788
call 0xff00aabb
```

For the given application and shellcode, the array input is located at 0xaabbccdd. Assume that no countermeasures are in place. Please describe what needs to be sent to inject and execute the shellcode.

Solution: One example solution:

- Inject the shellcode to the address 0xaabbccdd+0x18.

Question 5

Researchers has demonstrated a collision attack on SHA1. Please summarize

(a) what kind of cryptographic algorithms or applications are threatened by collision attacks
Threatened: crypto hash functions and everything using them

(b) what collision attack is
A Hash Collision Attack is an attempt to find two input strings of a hash function that produce the same hash result.

(c) what the expected cost of collision attack is (assuming no other weaknesses are exploited).
Cost: $2^{n/2}$

Question 6

You are designing a web application that will allow the user to create an image with two text strings in it. The user provides those strings via HTTP POST with form data. The web application extracts the strings from the form data, and then passes them

as arguments to a bash script to generate the image. The generated image is returned to the user (without embedding the user-provided strings in the webpage). The strings are stored in a database for documentation purposes.
Given that usecase, please list two(2) web security attacks that we discussed that might be possible, and one that should not be possible. For the two possible attacks, briefly give an example how they work and their effects. For the attack which should not be possible, please explain why you think they will not work here.
Solution: SQL injection+Command injection. Not working: XSS
(because strings are explicitly not embedded in server response).

Question 7
Alice and Bob use a modified Vigenère cipher to encrypt a message (of size 200 Bytes).

The plaintext of the messages is consisting of 8-bit ASCII characters. The modified Vigenère cipher works similar to Vigenère cipher introduced in the lecture: encryption by adding the key character to the plaintext character, decryption by subtracting the cyphertext character from the key character, but operates on bytes (i.e. 256 different input characters). The key has length 20 Bytes. It is used byte-wise as well. For example, if the first key character is 0x01 (in hex), and the first plaintext character is 0x61, the ciphertext character would be 0x62.

Eve knows that the key is 20 characters long (but not knowing the actual key), and that the message plaintext contains a 20 character secret string (lowercase, English language) at a random offset within the 200 characters. All other characters are filled with random decimal number characters. An example plaintext would be:

123412312156this is a secret bob9080709872 [...]

Eve obtains the ciphertext by eavesdropping, and wants to know the secret string contained within.

a) Can Eve determine at which position in the ciphertext the secret string is? If yes, describe how this could be achieved. If no, argue why not.

Solution: Yes, because the digits are drawn from a small set of 10 plaintext characters. This (modified) Vigenère ciphers is a period shift cipher, for a key of length 20, every 20-th plaintext character is shifted by same amount (offset).
This results 9 ciphertext characters with small mutual distance, and one with larger distance.
The area in the ciphertext with those larger distances is the area of the (hidden) plaintext.

b) Can Eve decrypt the ciphertext message partially or fully? If yes, describe how this could be achieved. If no, argue why not.

Solution: As discussed above, every 20-th character can be put into a set, and a mutual distance of ciphertext characters can be derived. The hidden alphabetic plaintext will result in an outlier. Frequency analysis on each set uniquely (in most cases) identify the exact shift applied for this set, which

can then be undone to decrypt the message (or get a couple of candidate messages).

hex	0x0	0x1	0x2	0x3	0x4	0x5	0x6	0x7	0x8	0x9	0xA	0xB	0xC	0xD	0xE	0xF
0x30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
0x60	‘	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
0x70	p	q	r	s	t	u	v	w	x	y	z	{	|	}	-	-

Figure 1. ASCII Table in hex for Question 7.

decimal	0	1	2	3	4	5	6	7	8	9
48	0	1	2	3	4	5	6	7	8	9
96	‘	a	b	c	d	e	f	g	h	i
106	j	k	l	m	n	o	p	q	r	s
116	t	u	v	w	x	y	z	{	|	}

Figure 2. ASCII Table in decimal for Question 7