Exercises - Week 9
More attacks

1. Are the following statements True or False:

a) **Eliminating buffer overflows would completely prevent the problem of Internet worms**
b) **Some viruses add their code to that of existing executables residing on disk.**
c) **Having a centralized botnet with one C&C is a good idea. You do not need to manage many computers since all report to the same C&C.**
d) **We cannot trust that executing a program will not do damage just by inspecting the program's source code**
e) **Trojans are self-contained programs that replicate themselves to infect other machines**
f) **Ransomware is a type of malware focused on network denial of service**
g) **Signature-based antiviruses are very effective against known viruses but not new**

a) F - The worms can exploit other vulnerabilities to gain control over systems.
b) T - This is one of the modus operandi of viruses. They can also add themselves in macros or in the booting files
c) F - Having a unique point of failure is **never** a good idea. The more spread is the C&C the more difficult it is to shut down the botnet.
d) T - The source code needs to be compiled in order to get executed. That there is no problem in the source code does not mean that the compiler will not introduce vulnerabilities or malicious code in the executable.
e) F - Trojans are not self-contained, and they do not replicate themselves. Only when combined with a worm they spread on their own.
f) F - Ransomware does not aim at shutting down a network. It aims at temporarily locking a computer to obtain money for releasing it.
g) T - The antivirus does not have the signature of the new virus, so the antivirus won't be able to detect it. This is the reason why signature based antiviruses require frequent updates.

2. You got access to an encrypted version of the final exam of COM-301 which is a file of 100KB encrypted with AES-128 (symmetric encryption with a key length of 128 bits). You really want to decrypt the file to get a very good score. It turns out that you have become a millionaire. For some reason instead of going to Hawaii, you decide that you are going to finish your degree and need to pass this course. Therefore, you really, really want to decrypt the exam. Assume you have three options:

a) **Buy a botnet that costs you 100CHF for every 2^{10} bots. Each bot can brute force 2^{40} keys a day (fictitious number!). You can buy at most 2^{24} bots.**
b) **Pay one of the assistants to leak you the key using a covert channel so that the professor does not realize. The covert channel leaks 4 bits of the key per day. The assistant does want to go to Hawaii for vacation every summer of his life, and since he has learned that you have became a millionaire, asks you for 25,000 CHF per bit.**
c) **Buy the latest coolest ever technology by Dr Nefario that can use a side channel attack from your seat in the classroom to get the cleartext version of the exam from the professor computer that can extract 1KB of the file every week. This coolest tech costs 1M CHF, but you are a millionaire and you do not care.**

Which option is the fastest? And which is the cheapest? Justify.

First, we analyze each approach to approximate the cost and delay of decrypting the exam.

a) The attacker can buy between 2^{10} to 2^{24} bot nodes. We study the two extremes:
   i) 2^{10} nodes: Cost = 100 CHF. Computation power = 2^{50} key per day. Expected time to break the decryption: $2^{128}/2^{50} = 2^{78} \cong 3 * 10^{23}$ days.
   ii) 2^{24} nodes: Cost = $2^{14} * 100$ = 1,638,400 CHF. Computation power = 2^{64} key per day. Expected time to break the decryption: $2^{128}/2^{64} = 2^{64} \cong 2 * 10^{19}$ days.
b) Cost = 128 * 25,000 = 3.2 Million CHF. Time = 128/4 = 32 days.
c) Cost = 1 Million CHF. Time = 100KB / 1KB = 100 days.

Getting a small 1024 machine botnet with 100 CHF is the cheapest way of getting the exam, but it takes 2^{21} years.

Asking TAs is the fastest way of getting good grades!!! It only requires 32 days.

3. Malware can use different approaches to choose to which machines in the network to spread. Compare each of the following approaches for the following points: i) how well they work to spread faster, ii) how effective they are if the malware has a pre-defined target, iii) how well they work for remaining undetected, iv) how well they work to obtain the widest coverage.

a) **Random (i.e., try random nodes in the network) vs Targeted (i.e., select new targets from a pre-defined set)**
b) **Full (send to every reachable node every time it spreads) vs Limited (restrict the number of nodes in each spreading step)**

a) Random vs targeted.
   i) The speed of random choice depends on the number of nodes tried every time. If the malware uses full, this is the fastest way. Targeted is fast if the attack is targeted (i.e., the set of victims is predefined).
   ii) This mode is not good if the target is pre-defined. There is no guarantee that the random search will find it fast. Targeted is optimal for this task.
   iii) Random search is more likely to be detected, since it will try many options increasing the possibility of hitting a monitored victim. Target is more likely to not be detected as it only touches a limited amount of nodes.
   iv) Random is good for widest coverage. It ensures that, over time, all possible victims will be infected. The targeted mode will likely not obtain large coverage, but it does not matter since it is usually selected when the victim set is pre-defined.

Stuxnet used a targeted spreading to target Iran's nuclear infrastructure while WannaCry used a random spreading since they didn't care about the identity of the victim.

b) Full vs limited
   i) Full spread is likely the fastest. Using this every node will try all of the possible available targets, getting to them faster. Limited nodes is not a fast way to cover all the network
   ii) Full mode should be better than limited if the target is pre-defined, since one would find it faster even using random search.
   iii) A limited spreading tries to keep a constant number of infected machines while full spreading aims to infect as many machines as possible. Security experts and antimalware softwares inspect systems to find malwares. Having less infected nodes makes the detection of the malware harder.
   iv) Full spreading is likely to provide wider coverage, giving the attacker more computational power. Limited spread can also obtain wide coverage, but at a much slower pace.

Slammer did a full spreading to infect a large set of machines. Stuxnet restricted itself to a small number of infected machine to remain undetected.

4. Explain what type of Intrusion Detection System (IDS) would you choose for (justify your answer):

a) **Detecting a well-known worm that rarely mutates**
b) **Detecting unknown worm**
c) **Detecting a known worm that randomizes its operation all the time (i.e., it never repeats the same pattern).**
d) **Detecting this particular worm that acts by sending 5 SYN-ACK at the beginning of the attack**

a) Signature based. If the worm is known and never mutates it must be possible to learn a signature that can be used to identify it.
b) Anomaly-based. If you do not know how the threat is going to look like, best try to detect anything that is not in the universe of good things.
c) Anomaly-based. Even being known, it is not possible to find a signature. Only anomaly detection is possible.
d) The 5 SYN-ACK are a signature. Just add it to your Signature-based IDS.

Previous concepts

1. This problem concerns the impact of weak cryptographic key generation on the security of messages exchange. The threats are Eve, an eavesdropper, and/or Mallory, a MITM attacker. [Assume neither Eve nor Mallory can conduct massive brute-forcing attacks].

Consider a scenario in which Bob and Alice use RSA cryptography. Alice uses BuggyLib to generate her public/private key pair, which always generates e = 1 as public key which implies that the secret key is d=1. Bob uses SecureLib which generates secure public/private key pairs.

Recall that: RSA has three parameters: e, d, and n, where e = d-1 (mod phi(N)). Encryption of a message m is m^e*, and decryption of a ciphertext c is* $c^d = (m^e)^d \bmod N = m^{ed} \bmod N = m^1 \bmod N = m$*. Digital signatures work conversely: Sig(m)=*$m^d \bmod N$ *and it can be verified using e.*

a) **Alice sends a message to Bob using RSA to obtain CONFIDENTIALITY, can Eve or Mallory perform an attack? What attack?**
b) **Bob sends a message to Alice using RSA to obtain CONFIDENTIALITY, can Eve or Mallory perform an attack? What attack?**
c) **Alice sends a message to Bob using RSA to obtain AUTHENTICITY, can Eve or Mallory perform an attack? What attack?**

a) No, they cannot. The message is encrypted with Bob's public key which was generated with SecureLib which produces secure keys.

b) Alice can read Bob's message, but so can Eve. The message M sent by Bob is encrypted with Alice's public key e=1. That means that the encryption is computed as M^1=M mod N (assuming M<N). Effectively, Alice has sent her message in the clear!!

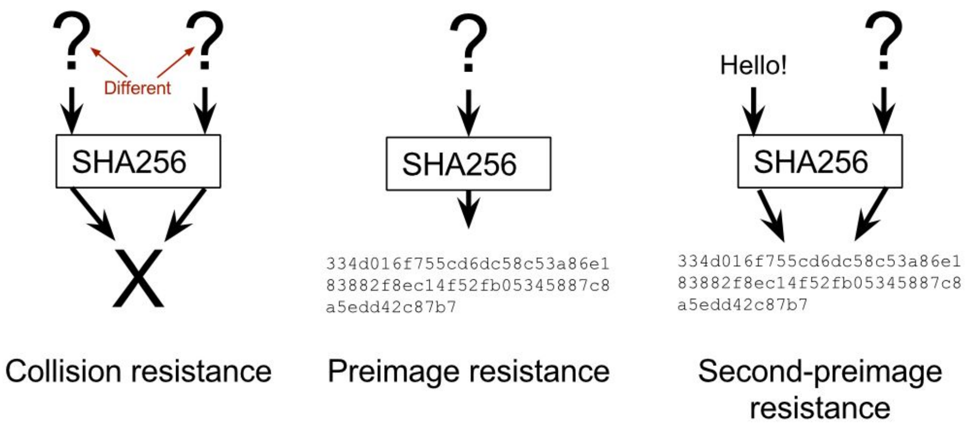
c) To obtain authenticity Alice signs her message as s=(M^d mod N), and sends to Bob the tuple (M,s) composed by the message and the signature. Since d=1, s=M. Mallory can see this, interpret that d=1 and learn that he can send any message to Bob as a tuple (Mx,Mx). Any tuple with two same values will verify correctly!

2. Nowadays, it's common to download frequently used resources from CDNs (Content Delivery Networks[1]). Sites like Google, BBC, etc. have many users who need to download all scripts of the page. Additionally, some libraries and scripts like jQuery (a javascript library) are used in many different sites. A CDN node caches these resources and can serve these cached resources to the user. The distance between CDNs and users are less than end-sites and users. Thus, using CDNs reduces the latency and significantly reduces network traffic, benefiting the internet infrastructure and ISPs. On the other hand this creates an opportunity for a malicious CDN to add a backdoor to the resource. Sub Resource integrity[2] is a new security feature that enables browsers to verify that the integrity of the resources they fetch is maintained. It works by adding an integrity field with a hash value to the link that points to the resource to be downloaded. For example:

```
<script src="https://example.com/example-framework.js"

integrity="sha384-Li9vy3DqF8tnTXuiaAJuML3ky+er10rcgNR/VqsVpcw+ThHmYcwiB1pbO
xEbzJr7"
></script>
```

This hash can be verified when the resource arrives to make sure that there has been no manipulation. What properties of the hash function are needed (Hint: is not all properties ;)) in SRI?


?
?
Different
SHA256
X
Collision resistance
?
SHA256
334d016f755cd6dc58c53a86e1
83882f8ec14f52fb05345887c8
a5edd42c87b7
Preimage resistance
Hello!
?
SHA256
334d016f755cd6dc58c53a86e1
83882f8ec14f52fb05345887c8
a5edd42c87b7
Second-preimage resistance

[1] https://www.cloudflare.com/learning/cdn/what-is-a-cdn/
[2] **https://developer.mozilla.org/en-US/docs/Web/Security/Subresource_Integrity**

SRI uses a hash to enable browsers to verify the integrity of the files it downloads. It requires a trusted source that publishes the correct hash. Note that, as explained in the class, a hash function does not provide integrity on its own (it uses no key! It can be computed by anyone!). In SRI integrity is achieved by receiving a hash from a trusted source *before* the resource is downloaded. Note that it is crucial that the hash comes from a different source than the resource (this is called an *out of band channel*).

In this scenario, to provide integrity SRI hash must fulfill the following properties.

If the adversary is a third party:

- **2nd preimage resistance:** the site / adversary cannot provide a manipulated/new resource that hashes to the hash corresponding to the original resource.

If you also consider the site providing the resource as an adversary:

- **Collision:** the site cannot pre-compute two resources that hash to the same value. Otherwise they could provide different resources to different users.

3. Would the following mitigations work against the mentioned attacks (justify)

a) "Confirm origin of request" against "OS command injection"
b) "Sanitization of inputs" against "Cross-site scripting"
c) "Confirming origin of request" against "Buffer overflow" (memory safety bug)
d) "Sanitization of inputs" against "Cross-site request forgery"
e) "Fuzzing" against "Control-Flow Hijack"
f) "Stack canaries" against "code injection"
g) "Data Execution Prevention" against "Control-Flow Hijack"

a) No. Knowing that the origin is the actual page does not mean the bug cannot be exploited.
b) Yes! Checking the inputs can make sure that you are not passing wrong data to the server script
c) No. Same as a)
d) No. The problem is the origin, not the input (actually, inputs are correct).
e) Fuzzing is not a mitigation :) But it actually helps with all type of attacks
f) Yes. It avoids that one can write too much in the stack.
g) No. DEP avoids new data being written and executed, but CFH exploits already existing executable data!

4. Are the following statements True or False:

a) A stream cipher is a One Time Password scheme.
b) OTP can be used to encrypt a movie for secure broadcasting.

c) **Block ciphers don’t accept inputs with arbitrary length.**
d) **A symmetric encryption can either be stream cipher or block cipher.**
e) **Stream ciphers are insecure.**
f) **A MAC is a good way of proving who is the sender of an inappropriate message.**

a) False: A stream cipher has *one short* key that is used to creates a large and random string of bits that is xor-ed with the plaintext. The xor part is similar to an OTP, but an OTP requires *one new key every time as long as the plaintext message*.
b) False: OTP needs a key as long as the movie!! Producing such a key in a secure way is extremely difficult, and so is transmitting it to the receivers.
c) True: Input should be divisible by the block length (one can pad the input to make it of a suitable size).
d) True
e) False
f) False, the MAC does not guarantee that the message was sent by one particular person. It could have been sent by any of the two people with knowledge of the symmetric key. Thus, it does not allow to prove which one of them is the sender.