Problem 1

Confidentiality: Limitation of access to obtaining texts, files, or any information that was supposed to be confidential from unauthorized parties.
Example: Zoom privacy policy gave the right to hosts Zoom users' personal data.
Source:https://objective-see.com/blog/blog_0x56.html

Integrity: Protection from unauthorized modification of texts, files, or any protected information. Assurance of accuracy, integrity, trustworthiness.
Example: Use of encryption under ECB mode. Potential confidentiality and integrity hazard.
Quote from citizen-lab, *"Zoom documentation claims that the app uses "AES-256" encryption for meetings where possible. However, in our testing, a single AES-128 key was used in ECB mode by all meeting participants to encrypt and decrypt audio and video".*
Source:https://citizenlab.ca/2020/04/faq-on-zoom-security-issues/

Availability: Guarantee of reliable service to authorized access.
Example: Blizzard Entertainment Gaming Company responded to customers about the possibility of experiencing DDoS attacks during PvP game matches. Quote from Blizzard developers on gaming forum, "*…a cheater is able to identify the IP address of a specific player by hacking the information from the console network's social features and begin a small-scale DDoS attack to jam out that player's connection to the internet, causing them to drop*".
Source:https://us.forums.blizzard.com/en/overwatch/t/can-you-ddos-on-overwatch/415343/5

PROBLEM 2

One-wayness: Given hash value(output value of hash function), it is hard to obtain the inverse of the it(input value), the original parameter passed to the hash function. “One-wayness”, as in it is only possible doing the “image mapping” way, yet computationally impossible to perform the reverse way.
Example: PBKDF2 (Password-Based Key Derivation Function 2) applies a certain degree of randomness through the “salting process” and performs “stretching” over an agreed upon amount of hashing iterations.

Weak collision: Given input value “x” that has hash image “y”, it is computationally impossible for one to find another input value “x’” that also results in the same hash value “y”.
Example: MD5 keys used to providing proof of purchasing software products.

Strong collision: It is hard to find any arbitrary pair (x, x’) that produces same hash value “y”.
Example: Feasibility of digital signatures on hash values of documents. The rationale behind this is that it is impossible to produce/ find two documents with the same hash.

PROBLEM 3

We can prove by contradiction.

Assuming the adversary has advantage in ordinary sense, so the adversary can determine some information with a non-negligible higher probability.

Precisely, **there exists message M** such that **for a non-negligible amount of random message M' and ciphertext c, there is probability p(M',c) that Pr[E(k,M) = c] is p(M',c)** (p.s. conditionally on one of Pr[E(k,M) = c] and Pr[E(k,M') = c] holds, and |p(M', c)- 1/2| > 0).

If p(M',c) > 1/2, then the probability:
Pr[b = 1]
*= 1/2 * Pr[E(k,M) = c] / (1/2 * Pr[E(k,M) = c] + 1/2 * Pr[E(k,M') = c])*
= p(M', c)/ (p(M',c) + (1-p(M',c)))
That is, we guess b = 1 we get probability p(M',c) = 1/2 + |p(M',c)-1/2| correct.

Similarly, if p(M', c) < 1/2, then:
Pr[b = 0]
*= 1/2 * (1 - Pr[E(k,M) = c]) / (1/2 * Pr[E(k,M) = c] + 1/2 * Pr[E(k,M') = c])*
= (1 -p(M', c))/ (p(M',c) + (1-p(M',c)))
So by guessing b = 0, we get probability (1-p(M',c)) = 1/2 + |p(M', c) - 1/2| correct.

Therefore, we can always get 1/2 + |p(M',c) - 1/2| probability correct in both cases. As claiming that averaging over all possible p(M',c) it is non-negligible, we are done.

On the other hand, if there is a non-negligible probability P[b = b'] > 1/2, then:
P[b = b']
*= P[b = b' | b = 0] * P[b = 0] + P[b = b' | b = 1] * P[b = 1] (conditional probability)*
*= P[b = b' | b = 0] * 1/2+ P[b = b' | b = 1] * 1/2,*
it must be true that one of P[b = b' | b = 0] or P[b = b' | b = 1] is greater than 1/2.

While when Pr[b = b' | b = 1] > 1/2, according to pigeonhole principle, there exists a non-neglible ratio of message M' and c such that with (Pr[b = b' | b = 1] + 1/2) / 2 > 1/2 probability, it is E(k,M) = c rather than E(k,M') = c. Similarly, by pigeonhole principle, there exists non-negligible message M' such that (Pr[b = b' | b = 0] + 1/2) / 2 > 1/2 probability, it is E(k,M') = c over {E(k,M) = c or E(k,M') = c}.

Therefore we have shown the equivalence due to prove by contradiction.

PROBLEM 4

How to run code:

1. Run code4.py with python
2. During runtime, input ‘y’ when the appearing line of text turns meaningful.
3. Flag will be shown on screen after finishing previous step.
4. Flag is in problem4.txt.

Solution:
Basic Caesar attack.

Flag: CNS{CRYPT0_1S_S0_$IMP13}

PROBLEM 5

How to run code:

1. Run code5.py with python
2. Wait until D1, D2, D3 appears patiently haha.
3. The a0 secret is in problem5.txt, along with D1, D2, D3.

Solution:
Use Lagrange polynomials to find weight of D1, D2, D3 that constructs a0, a1, a2.
Note that there exist fast algorithm for computing power in modular arithmetics.
Note that g^D1 = c1 * c2 * c3, similar to g^D2 and g^D3.

a0 = 22903031829579284005874185426584358429053

PROBLEM 6

How to run code:

1. Run code6.py with python
2. Wait until “done”. Then the flag appears
3. Flag is in problem6.txt.
4. Only able to solve for one flag.

Solution:
Reveal original plaintext by CBC padding oracle attack from the last block.
Turns out that part of plaintext is one of the flags.

Flag: CNS{A_lonely_cat_wan||isvip:0||isadmin:0||desc:ts_friends.Meow~meow~(=^._.^=)}