1.

Confidentiality 保密性：保護資料不受到未經授權的揭露。如 encryption 便是做到保密性的一種方式，而如 Meltdown、Spectre 漏洞因為可以知道別的應用程式的運行資料，便違反了保密性。

Integrity 完整性：保障資料不會受到未經授權的篡改。如 checksum 可用來驗證是否經過被竄改，而 MitM 攻擊、連線劫持等便破壞了完整性。

Availability 可用性：確保目標使用者可存取欲獲得的資源。好的硬體與頻寬可維持伺服器的可用性，但如 DoS 或近期 Github 所遭受的 DDoS 便破壞了可用性。

2.

One-wayness: 很難找到某 hash 值的 input 是什麼。EX: sha256(x)= A665A45920422F9D417E4867EFDC4FB8A04A1F3FFF1FA07E998E86F7F7A27AE3，不易知道原先內容 x 為何。如挖礦的 proof of work 便是利用此性質來調整計算難度。

Weak collision resistance: 給定 X 的 hash 值，很難找到別的 input 之 hash 值與 X hash 值相等，For any given value h it is computationally infeasible to find y = x with H(y) = H(x)。X=456，sha256(X)= B3A8E0E1F9AB1BFE3A36F231F676F78BB30A519D2B21E6C530C0EEE8EBB4A5D0，不易找到另外一個輸入與之相等。如在檢查下載檔案完整性的 MD5 驗證碼便是利用此性質。

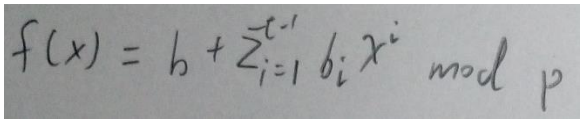
Strong collision resistance: 很難找到一組 input 的 hash 值相等，It is computationally infeasible to find any pair (x, y) such that H(x) = H(y)。如非利用他人密碼亦可登入他人帳號。

3.

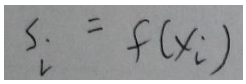
Setup:

Large prime: p
Generator: g
Random number: b

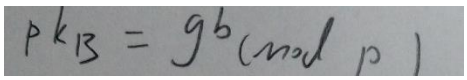
Compute a random number degree t-1 polynomial:



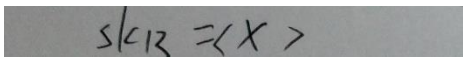
Compute n shares of b for each user i:



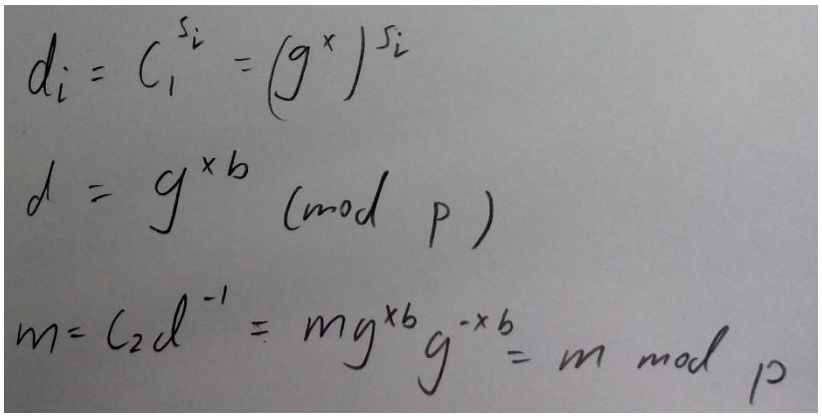
Public key:



Master secret key:



Decryption:



4.

第 0 題: 將所傳過來的字串回傳即可

第 1 題: 為 numbers to alphabet 題目，先將空白字元轉為 ascii code 32 後，兩個兩個一組讀，將值+96 成為對應的 ascii code 利用 chr()轉換回字母再回傳

第 2 題: 為凱撒密碼加密法，將所得到的密文丟入工具解開即可(https://www.xarg.org/tools/caesar-cipher)回傳

第 3 題: 為替換式密碼凱撒加密，也可將密文丟入工具(https://www.xarg.org/tools/caesar-cipher)嘗試解開再行回傳

第 4 題: 也為 Substitution cipher，利用頻率分析法，利用工具(https://quipqiup.com/)可協助嘗試出最有可能的答案並予以回傳

第 5 題: 換位加密法，將密文進行切割成不同長度，組成不同排列，接著看哪一種組合式有意義的明文便可回傳

第 6 題: Rail fence cipher 籬笆加密法，解法與波形應有關係，但尚未找到解密方

式 QQ

5.

利用 factor DB 將 N 拆出 p 和 q 以後

並利用 libnum 套件，程式碼如下:

Import libnum

P=6864797660130609714981900799081393217269435300143305409394463459185543183397656052122559640661454554977296311391480858037121987999716643812574028291115057151

Q=5311379928167670986895882065524686273295931177270319231994441382004035598608522427391625022652292856688893294862465010153465793376527072394095199787665873519438312708353932190317281270804166349427306646525252846187992860040468618963138756087975061397

N=P*Q

C=581788861819028849604486057941619390966415254053077878653962134906354590013338110809005469428056022879067145295587559787690231746970635080581011868491636345452516660728072373182932320408042523797540667249887968750055421285011306277973114453319150410174840933885813769726830441663494273066465252528461879928600404686189631387560879750613977

E=65537

Phi=(p-1)*(q-1)

D=libnum.modular.invmod(e,phi)

Print libnum.n2s(pow(c,d,n))

BALSN{if_N_is_factorized_you_get_the_private_key}

6.

安裝 featherduster

（https://github.com/nccgroup/featherduster/blob/master/INSTALL.md）

利用程式碼將密文讀入 在利用套件的 `break_multi_byte_xor` 函式暴力搜索出有意義的 `english sentences` 即可從中獲得 `flag`

As suggested by the description we need to break the repeating pad, which is essentially a multibyte XOR. As a shortcut let's use cryptanalib for that. Unfortunately it does not output the found key, so we have to XOR again with ciphertext to get it:

BALSN{NeVer_U5e_0ne_7ime_PAd_7wIcE}

7.

將 plaintext 利用 key0 加密一次，也用 key1 解密 ciphertext 一次。而依據提示因為範圍落在 0~2**23 間。因此可利用暴力破解方式，尋得兩者一樣的時候之 key0 與 key1 分別個是什麼。再利用同樣 key1 與 key0 將 flag_en 解密，即可得到 flag。

※註: 解密須對助教所給的 ciphertext 值，利用 bytes.fromhex 轉換，才能進行解密

BALSN{so_2DES_is_not_used_today}

8.

利用 google 所找到的 sha1 collider (pdf header、jpg header 與不同的 collision block)可以組出不同的 prefix 但擁有相同的 sha1。接著利用 for 迴圈暴力跑 0~99999999 尋找哪個 prefix 的 sha1 後六碼與題目所給的相同，便可將該 prefix 的 hex 做為 X 傳回給伺服器。而將另外一種 prefix 的 hex 做為 Y 傳給伺服器便可得到 flag。

BALSN{P0W_1s_4_w4st3_0f_t1m3_4nd_3n3rgy}

9.

可利用 hashpump 工具協助進行 LEA 攻擊，-s 參數為 sha256(adminPassword || admin || Nc || login)，--data 參數為該 hash 所知道的明文(|| admin || Nc || login)，-a 參數為(||printflag)，-k 參數為所要猜測的 password 長度(1~30)，然而在 third 欲檢查的是 Ns，因此須利用 oracle padding attack 方式讓伺服器幫忙算，然而經嘗試多遍皆沒有成功 QQ。

10.

cut-and-paste attack 為一種針對完整性的攻擊，攻擊者替換掉一部分密文，被替換掉的密文與正常部分一起解密，但所產生出的明文與原先不同而可拿來利用嘗試修改訊息 EX:將 role 從 guest 變為 admin。

DSA 為數位簽章演算法，將公鑰密碼反過來使用，簽名者將訊息用私鑰加密，公布公鑰;驗證者使用公鑰將加密訊息解密並比對消息。
然而尚未想到該利用何種方法將兩者想法結合在一起來破解這題…

Reference:

https://quipqiup.com/
https://www.xarg.org/tools/caesar-cipher/
http://tholman.com/other/transposition/
https://web.archive.org/web//teconeq%20ehn%20ereq%20nolrechvirut%20eeodidita*
http://crypto.interactive-maths.com/rail-fence-cipher.html
https://www.dcode.fr/scytale-cipher