Information Security

Assignment # 2

Question # 01:

Let us see what goes wrong when a stream cipher key is used more than once. Below are eleven hex-encoded ciphertexts that are the result of encrypting eleven plaintexts with a stream cipher, all with the same stream cipher key. Your goal is to decrypt the last ciphertext, and submit the secret message within it as solution.

Hint: XOR the ciphertexts together, and consider what happens when a space is XORed with a character in [a-zA-Z].

Ciphertext # 01

260f02174c1f094918070c030c5315070601530c0d15180901131e430d1057170810081952050a1f120
20a430b170442202649141e1806165715194a0419104a180f07114b11000309550d53040e440b0a0d0
158190c4e01180f1c01460703041b4e02141a42061455040b1f480406154b1a041a11001d421f1e481d
03051543070542

Ciphertext # 02

214a061e4c1c14001b1c0b10491e1f48180110130d124b0515114d0d171b15170b451d021d490b1e1b
5116430616191549020116065 91c450419191f0612441d01081c034b09014f0c1101124607011e48020
b0814431d1907034810030b1f41101c0c155616010355030a1f0d141d041f4e170d1c4e0a03051e0d4a
0210410b0b0407

Ciphertext # 03

294a0f1a080f03074f1800041a12010d4b0d004101080d07141b0c170b1919520d0d080152001c511b1
9164304151b070d1c080317150c45191e0203091305081f0448070502491b0d1410530b1e171e48030
158090a1d1505100d1603174b0e044e1616150d1f03070f004b09081741020002060a1e0807020f0c4a
0901070c1c1342111a

Ciphertext # 04

3c020253050500061d180403001c0848020a530901020f0d08560006110516151c16491c0149011e015
60b0e001d120b08010c1b0b591b0a0318150f0b14080f484101124b0b1c1c11550616460f0d190b0e12
1d1f060a560514481108100417131c061c5a420808114a0d051c030111190b02061c4e180710051a0f4
b0d15430d170c580c03

Ciphertext # 05

294a05120f0011081d11451a0c0015090c01530806460a06461718070b1957001c0606071600011655
1f114302161a1b49131c1b1e005504070117180f18104a04090d084b12010a45070110091900030606
44111e431e1a0b1f0d0046010e17131c101d124c49351a07014b0a07100a1c0f040758031f11050b0f0f
184400110b56120a01021d10010e

Ciphertext # 06

2a0b0418010a1502061b0257000046094b16160207140f0108114d1707151f1c10141c1052000151021
e0b000558174204101a04131e10451e0256180f150b1817040c4609070a0412141617154b0b041c0e4
4194d171c17090d48100e121f411f1d431513030712551e0b4b0a035311070f0f061c4e1c0d041d0918
0f174f432702420f0f1548

Ciphertext # 07

071847030419071a0a0645110601460b0701120f48140e0403171e0611561814591606 1b151a4f3314
15090e0c0b1d0b0712491f130a55071214184a0b5607051d151a091d031b1c0c140853120414030b41
0d164d1706134a33060d12160f41251a020c131149151c04070e48121b044b191e06164e1b0e1a0f0f0
b1f0d0e0d1d560d1e4e0f1c00

Ciphertext # 08

1d1902530a0414493c141116071a05481b11011107150e1b4601081107561a131d0049141508061f06
0242131f171b0b07100703520b1a061c511b1f191f0703120f1b4a4b0a0c0e011c0a14461f0b4a1a0407
171f0743141f14060d08141841170007580610061 61a19010f48071d15024314021b051703050101040
c440d06091f11140f12011c0a

Ciphertext # 09

0a134700180a120c4f140b134915030c0e16120d4801041e0304030e07180301592c0755160c0d10011
342021f1d560006010157061110451 2091f191e130a09164107004b04080c0e1805000d0e004a3b001
019030a0d5607031b1707140e12560f0d1c561601031c18441b1d14030e191a1307580f180b1a031c13
4b100e431d030014070b011d05060711

Ciphertext # 10

090c01160f1f46050606111 20716141b4b0b1d1507460a4812040c000956031a1811491c01490214141
81643191756000c55191b1300100157171918 1d17160e004f482f1f461e0e16551316140e4407090501
580c040f1f04151c441601040c1f00061602421b091601440f0d0412150e4e17111d4e180d0202481e0
3014106161f110c0b080b1644

Ciphertext # 11

200f0b1f034b0b104f1b041a0c530f1b4b10120f01074b1b071a08060f56161c1d45005513044f06071f1
60a031f560349060c14001c01451a1405190b11014a150e1a462235490c0914170046024402071 10158
190b0b0f4a1101080a530904560f011413421d09550e01081a1f03154b0702431b01081013091c06124
41802170f421c0b051a0a141e0e0c

Question # 02:

Multicast MACs. Suppose user A wants to broadcast a message to n recipients $B_1, \ldots,$ B_n. Privacy is not important but integrity is. In other words, each of $B_1, \ldots, B_n$ should be assured that the message he is receiving were sent by A. User A decides to use a MAC.

a) Suppose user A and $B_1, \ldots, B_n$ all share a secret key k. User A computes the MAC tag for every message she sends using k. Every user Bi verifies the tag using k. Using at most two sentences explain why this scheme is insecure, namely, show that user B1 is not assured that messages he is receiving are from A.

b) Suppose user A has a set $S = \{k_1, \ldots, k_m\}$ of m secret keys. Each user B_i has some subset $S_i \subseteq S$ of the keys. When A transmits a message she appends m MAC tags to it by MACing the message with each of her m keys. When user B_i receives a message he accepts it as valid only if all tags corresponding to keys in S_i are valid. Let us assume that the users $B_1, \ldots, B_n$ do not collude with each other. What property should the sets $S_1, \ldots, S_n$ satisfy so that the attack from part (a) does not apply?

c) Show that when n = 10 (i.e. ten recipients) it suffices to take m = 5 in part (b). Describe the sets $S_1, \ldots, S_{10} \subseteq \{k_1, \ldots, k_5\}$ you would use.

d) Show that the scheme from part (c) is completely insecure if two users are allowed to collude.