# ECE/CS578 Final Exam

Fall-21 / December 5-6, 2021

Name: Justin Cabral

| Problem | 1 | 2 | 3 | 4 | 5 | Total |
|---|---|---|---|---|---|---|
| | | | | | | |

Exam rules:

- Deadline: December 7, 2021, 6pm.
- Submission: on Canvas
- Individual test: *No team work!*
- You can ask your questions to the instructor at 3pm on December 5on Zoom (https://wpi.zoom.us/my/koksalmus)

## Good luck and have fun!

1. *Explain the followings:*

(a) What is Kerckhoffs' Principle? Why is it important?

**Answer:** Kirchhoff's Principle is that in a secure cryptographic system, the encrypted messages and encryption algorithms are public knowledge but don't compromise its overall security. The only thing that should remain private is the secret key. The principle is important because even if an attacker knows how the cryptographic system works at a fundamental level, there is nothing they can do to retrieve the message. The cryptographic system is sound at such a fundamental level that revealing the secret key is the only reason it should be compromised. This allows the researchers to test claims made about cryptographic systems as well.

(b) Define perfect secrecy and computational security. Explain why popular practical encryption schemes achieve computational security but not perfect secrecy.

**Answer:** Perfect secrecy is when a given ciphertext conveys 0% information about the plaintext used to generate it. Meaning if given 1 plaintext and 2 ciphertexts, you could not distinguish which ciphertext was generated by the plaintext making it a 50/50 decision. Computational security on the other hand is when a ciphtertext conveys a very, very small amount of information about the plaintext that generated it, but it will take too much time computationally to distinguish that difference. Meaning if given 1 plaintext and 2 ciphertexts there would be a 49.99999/50.00001 chance of either being correct, but it would take too much computation time to tell which.

The Popular encryption schemes achieve computational security and not perfect secrecy because it is much more practical for the world we live in. For one, to be perfectly secure means that the key space must match the size of the message, making it slow in practice. Also perfect secrecy only solves encryption of the transmission of a message, but leaves the problem of sending the secret key needed to decrypt the message. Therefore asymmetric, computationally secure algorithms such as the RSA family are most widely used today because it solves the problem of needing to send the secret key.

(c) What is a "one-time pad (OTP) / Vernam Cipher" and why are they not used in practice?

**Answer:** The one-time pad is an encryption scheme that cannot be cracked and requires the single use of pre-shared key that is no smaller than the message that is being sent. They're not used in practice because the one-time-pad only solves the problem of encrypting the message but does not account for the sharing of the single use key. The need to encrypt the single use key to be shared is redundant and defeats the purpose of using the system.

(d) Explain the similarities and differences between public key encryption schemes and digital signature schemes?

**Answer:** The similarities between public key encryption and digital signature schemes is that they both use an asymmetric, public/private key pair system as their foundation. However, with public key encryption, you're using a person's public key to encrypt a

message to send to them, which they will decrypt with their private key. In digital signature schemes, a person can validate who they are by using their private key to sign a given message to prove authenticity.

2. *LFSR-based Stream cipher*

We conduct a known-plaintext attack on an LFSR-based stream cipher. We know that the plaintext sent was:
1001 0010 0110 1101 1001 0010 0110 By tapping the
channel we observe the following stream:
1011 1100 0011 0001 0010 1011 XXXX Assume that the chosen LFSR generates a maximum-length sequence.

(a) What is the degree $m$ of the key stream generator?

**Answer:** The degree m is defined as $m = \text{Log}_2(P + 1)$ therefore $\text{Log}_2(7 + 1) = 3$

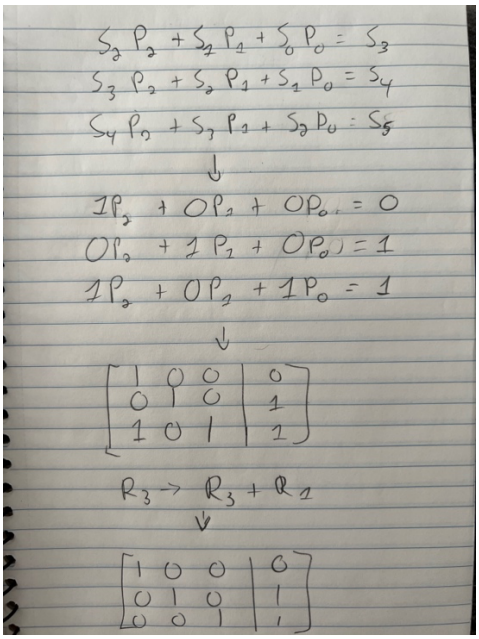
The degree m of the key stream generator is 3.

(b) What is the initialization vector?

**Answer:** Given that the first bits of the keystream are s0 = 0, s1 = 0, and s2 = 0, the initialization vector is (s2 = 1, s1 = 0, s0 = 0)

(c) Determine the feedback coefficients of the LFSR.

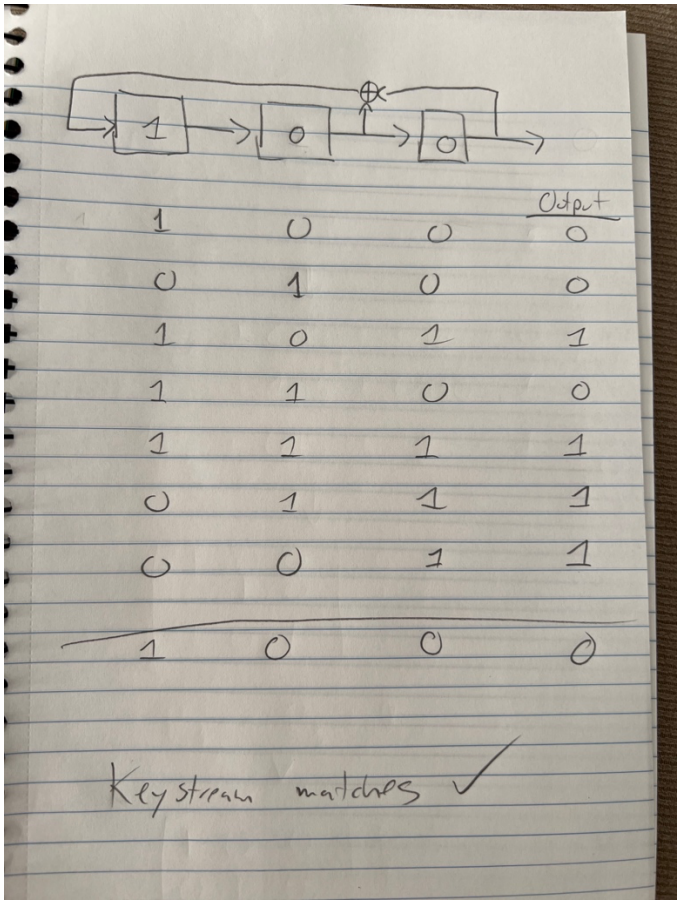
**Answer:** Knowing the degree of the key stream is 3, we can use a set of equations produce the next 3 values using the feedback coefficients as the unknowns. After finding the next 3 values we can do a Gauss-Jordan Elimination to find the feedback coefficients.




This gives us the feedback coefficients of ($P_0 = 1$, $P_1 = 1$, $P_2 = 0$)

(d) Draw a circuit diagram and verify the output sequence of the LFSR.

**Answer:**



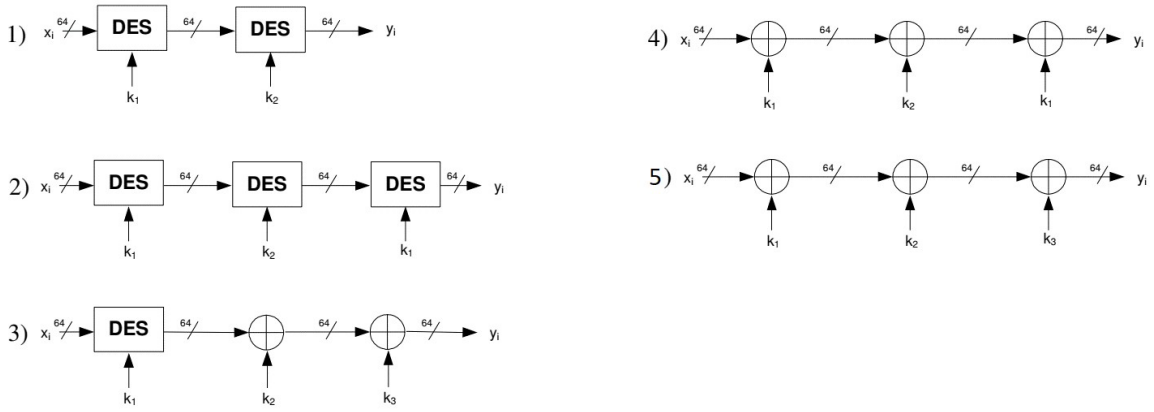

(e) Recover the missing 4 bits of the ciphertext stream.

**Answer:** The missing 4 bits of the ciphertext stream are 0001

3. *DES*

There are different ways to increase the security of block ciphers. This problem proposes different methods to increase the security against brute force attacks. Your task is to assess the security of these methods.

Assume that the adversary knows two message-ciphertext pairs $(m_1,c_1)$ and $(m_2,c_2)$. Furthermore, the adversary is able to break a simple DES instance via a brute-force attack. The key lengths are $|k_1| = |k_2| = |k_3| = 64$ bit. The following schemes are given:




(a) Explain how an adversary can efficiently attack the encryption scheme (i.e.,explain the most efficient attack on the scheme brief and concisely)?

**Answer: Scheme 1)** For 2DES, the most efficient way to attack the scheme is to split the cipher in half and attack both sides separately and do a meet-in-the-middle attack. This would leave us with 2 halves and $2^{56}$ operations which to solve separately. Since our attacker can brute force a 64bit key space, this would be his most efficient route to go.

**Answer: Scheme 2)** For 3DES with 2 unique keys the most efficient way to attack the scheme is to treat the outer DES as 2DES with a single key, and the inner function as a simple DES. This allows us to perform the meet in the middle attack to the 2DES the same way in which we performed it in the example for the answer above. Once solving that we can move on to the single DES and brute force it.

**Answer: Scheme 3)** For DES with 2 keys XOR'd we can simply brute-force k3 using our ciphertext, and then brute-force k2 using our found k3, and finally brute force the final DES with our result from k2. We can do this because each key space is only $2^{64}$ which our attacker has the capability to defeat.

**Answer: Scheme 4)** For this triple XOR scheme with 2 unique keys, the attacker can brute force the 64-bit key space for k1, and k2. Since k1 is used twice, this reduces the overall computation time.

**Answer: Scheme 5)** For this triple XOR scheme with 3 unique keys, the attacker can simply do a brute force attack on each key separately since the effective key space is only 64-bits.

(b) What is the effective key length of the scheme, i.e., how many bits of the key does an attacker have to guess to break the scheme?

**Answer: Scheme 1)** Effective key length is 112. The attacker needs to find the 64 bits in each k1 and k2 in order to break the scheme.

**Answer: Scheme 2)** Effective key length is 112. The attacker needs to find the 64 bits in k1 and k2 in order to break the scheme.

**Answer: Scheme 3)** Effective key length is 120. The attacker needs to find the 64 bits in each k1 and single k2 in order to break the scheme.

**Answer: Scheme 4)** Effective key length is 64. The attacker needs to find the 64 bits in each k1 and single k2 in order to break the scheme.

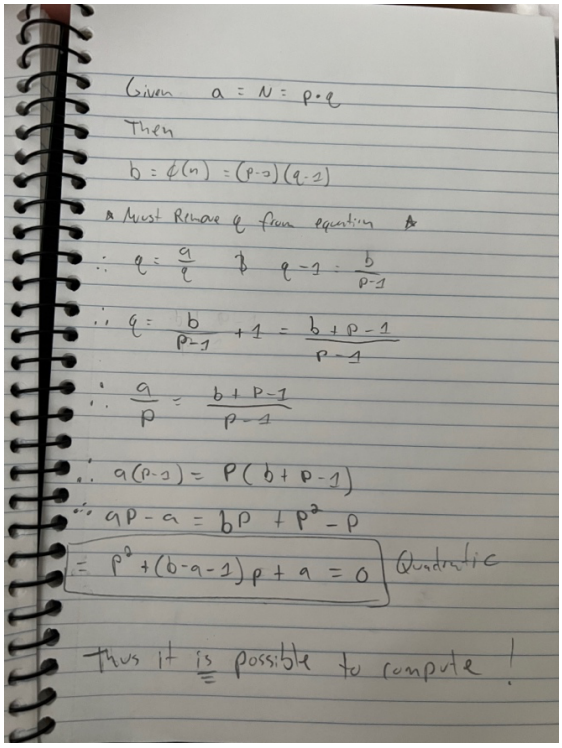
**Answer: Scheme 5)** Effective key length is 64. The attacker needs to find the 56 bits in DES, 64 bits for key 2 and key 3.

(c) Which of the schemes show a significantly improved security compared to a single encryption?

**Answer:** Scheme 1 with 2DES is a significant improvement over a single encryption due to the increase in effective key space. As well as Scheme 2 with 3DES and 2 unique keys because it also increases the effective key space. Scheme 3 also has improved security due to the larger effective key space.

4. RSA

(a) Let $N = pq$ be the product of two distinct primes. Show that if $\varphi(N)$ and $N$ are known, then it is possible to compute $p$ and $q$ in polynomial time.
(*Hint:* Derive a quadratic equation (over the integers) in the unknown $p$)



It is possible to compute p and q in poly time.

(b) Eve records the transmission of an RSA-encrypted message. The recorded ciphertext is $c = 2032$. Eve also knows the public key to be $pk = (n,e) = (7979,1879)$. Your goal is to recover a message $m$ that has been encrypted with RSA.

- Give the equation for the decryption of $c$. Which variables are not known to Eve? Can Eve recover $m$? If so, how? If not, why not?

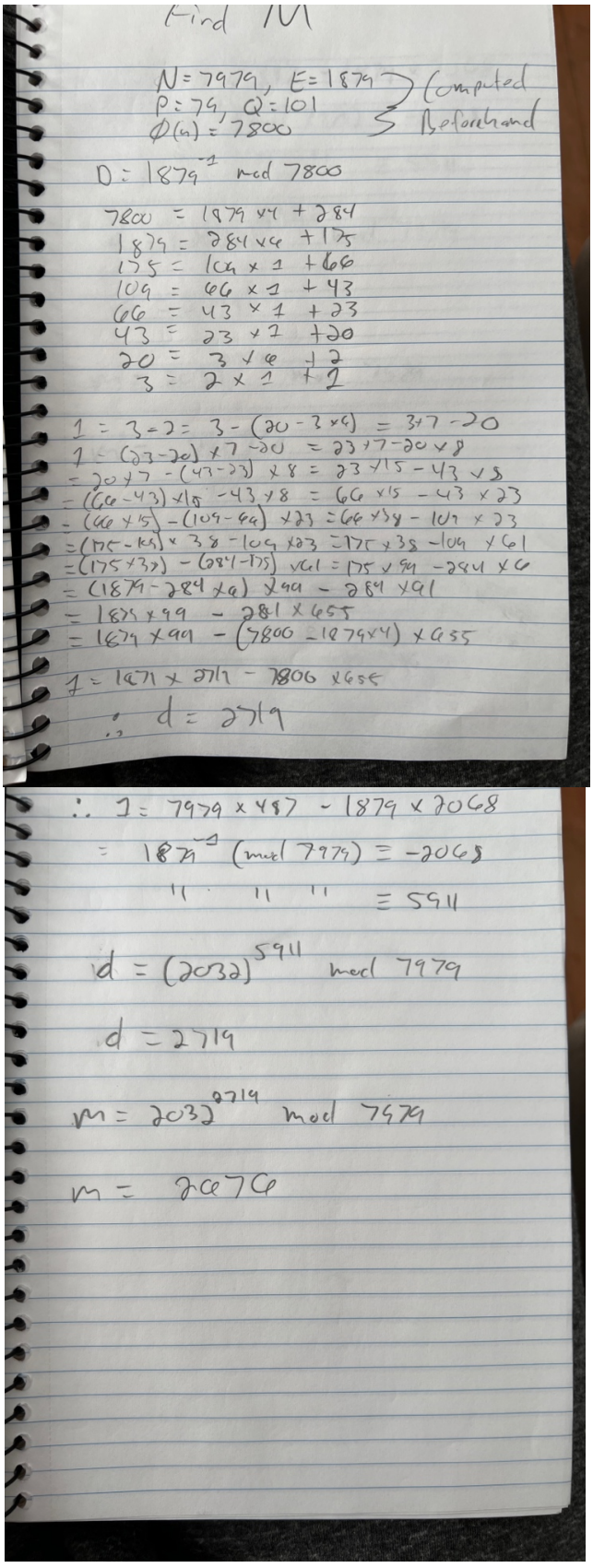
  **Answer:** The variables that are now known to Eve are p, q which are prime factors of N. As well as phi(n) which is derived from using p and q. Given that our public N = 7979 we can perform a prime factorization which gives us p = 79 and q = 101. We can now calculate phi(n) = 78 * 100 = 7800. From here this allows us to compute the modular inverse of D and recover m.

- To recover the private key $d$, Eve has to compute $d = e^{-1} \bmod \varphi$. Can Eve recover $\varphi(N)$?

  **Answer:** Yes Eve can compute phi(n) by doing a prime factorization of N = 7979 which gives us (79,101) for p and q respectively. Phi(n) = (79-1)(101-1) = 7800.

- Compute the message $m$.
  (*Hint:* Start by factoring $N = pq$. Then use $\varphi(N)$ to compute $d$)

  **Answer:**







- Can Eve do the same message recovery attack (as in (c)) for *large* $N$, e.g., $|N|$ = 1024bit?

  **Answer:** No, Eve cannot do the same recovery attack as in C because the key space of 1024bits is far too large and is computationally secure from an attempt at prime factorization.

- Eve recovers a message-ciphertext pair $(m,c)$. Can she recover the private key $d$?

  **Answer:** No, because in order to recover the private key we would need to know at least n and e. If the key space is too large, then knowing p & q would allow us to recover d as well.

(c) In class, we have seen that multi-encryption, i.e. cascading encryption functions, e.g.triple DES $TDES_{k1k2k3}(x) = DES_{k1}(DES_{k2}(DES_{k3}(x)))$ may be used to increase the security level of block ciphers. Now consider the case, where we use multi-encryption on a public-key algorithm: RSA encrypt data multiple times with different keys. Can we come to the same conclusion for RSA encryption? Mathematically justify your answer for double RSA.

**Answer:** Using RSA to encrypt data multiple times with different keys does not increase the security of RSA. Here's why:

In RSA we know that

*Encryption* = $D^e \bmod n$

*Decryption* = $E^d \bmod n$

Since $e_i d_i$ is congruent to 1 mod phi(n) we know that finding any double encrypted message is

$(m^{e1e2})^{d1d2}$ which simplifies to = m

Therefore, we know $d_1 d_2$ must be brute forced beforehand instead of d1 and d2 separately. However, this does not mean that RSA is more secure because if one were to solve the prime factorization of any public key for e1 or e2, it would break the encryption since RSA at its fundamental level uses prime factors as the foundation for its strength of encryption. No matter how many times you use RSA to encrypt, if you're able to solve prime factors, you can break any length or amount of RSA encryptions.

5. *Hash functions:*

A company stores credentials for user authentication in a single file. The file contains a salt and the hash output of a user-chosen password appended with the salt. The used hash functions fulfills all requirements, but its output is only 128 bits long.

(a) Name the three properties required for a cryptographic hash functions.

**Answer:** The three properties of secure cryptographic hash functions are:

1. Collision resistance
2. Preimage resistance
3. Second preimage resistance

(b) Is the output bit length sufficient for the application? Explain your answer.

**Answer:** No, the big length is not sufficient for the application because a hash function digest of n-bits has a collision resistance of n/2 bits. So, in this example our collision resistance is only 128/2 which gives us $2^{64}$ of computational security, which can be brute forced with today's hardware. In order to achieve the intended 128-bit security the output bit length would have to be 256 bits.

(c) Explain the security benefit of the salt value in the example.

**Answer:** The security benefit of the salt value is that it guards against the collision-based vulnerabilities such as the birthday paradox where two users with the same password would produce the same hash. By placing a randomly generated SALT value at the beginning of each password, it helps protect against all potential collisions in the future.