CS590 DSP: Data Security and Privacy **Duc Viet Le**
CS590-Data Privacy

Homework #3

Problem 1 (6 pts) Levels of security

- (2 pts) Assume that you are doing an exhaustive key search attack, you have access to a machine with 64 cores where each core can independently check 2^{32} (or about 4G) keys per second, about how many years would it take to for you to search through all keys while attacking ciphers of the following key lengths: 64, 80?

  Answer: since 64 cores can run in parallel, we can search $2^{32} \times 2^6 = 2^{38}$ keys per second
  - For key length of 64, the size of the key space will be $|\mathcal{K}| = 2^{64}$. Thus, it will take:

    $$2^{64}/2^{38} = 2^{26} \text{ seconds} \approx 2.128 \text{ years}$$

  - For key length of 80, the size of the key space will be $|\mathcal{K}| = 2^{80}$. Thus, it will take:

    $$2^{80}/2^{38} = 2^{42} \text{ seconds} \approx 139461.14 \text{ years}$$

- (2 pt) Still assuming that you have a computer with 64 cores where each core can check 2^{32} keys per second, further assume that the age of universe is 13.7 billion years, how many ages of the universe does it take to search through a key space of 128 bits?

  Answer: For key length of 128, the size of the key space will be $|\mathcal{K}| = 2^{128}$. Thus, it will take:

  $$2^{128}/2^{38} = 2^{90} \text{ seconds} \approx 3.925 \times 10^{19} \text{ years} \approx 2.865 \times 10^{9} \text{ universe age}$$

- (2 pts) Does this mean that a cipher of 128 bit can never be broken by brute-force? Justify your answer.

  Answer: If brute-force attack means exhaustive key search attack, then a cipher of 128 bit cipher may not be broken in the near future because the key space is huge.
  However, if we are talking about AES-CTR implementation with an initial vector picked uniformly at random, then due to birthday paradox, the chance of reusing IV is 1/2 after 2^{64}
  Also, other attacks like side channel attack, padding attack are also effective.

Problem 3 (10 pts)

a (5 pts) Password hashing and salting. Let us consider that a system administrator is thinking about how to store the user passwords of a system. For defeating dictionary or rainbow table attacks, he decides to store the hash values of the salted-passwords. He chooses a well-known hash function $\mathsf{H}(\cdot)$ for this purpose. He then comes across the following attack on the hash function $\mathsf{H}(\cdot)$: Given a hash value h and any arbitrary value x, the attacker can efficiently calculate another value y such that $\mathsf{H}(x||y) = h$. He is stringent on using the hash function $\mathsf{H}(\cdot)$. Given a password p and a random salt r, which of the following values should he store in the password file: $\mathsf{H}(p||r)$ or $\mathsf{H}(r||p)$. Justify your answer.

Answer: He would store $\mathsf{H}(p||r)$ in the password file.

Consider the attacker $\mathcal{A}$ that has access to $\mathsf{H}(\cdot)$, $\mathcal{A}(x, h)$ outputs y such that $\mathsf{H}(x||y) = h$.

If the system admin stores $\mathsf{H}(r||p)$, then $\mathcal{A}(r, H(r||p)) \rightarrow p'$ such that $\mathsf{H}(r||p) = \mathsf{H}(r||p')$. Therefore, the attacker can use p' to log in.

However, if the system admin stores $\mathsf{H}(p||r)$, then $\mathcal{A}(r, H(p||r)) \rightarrow p'$ such that $\mathsf{H}(p||r) = \mathsf{H}(r||p')$. This time the attacker is not able to use p' to login and he cannot change r. Therefore, he will not able to log in.

b (5 pts) Weird hash function. Let us assume a customer has given you the responsibility of designing a "*weird hash function*" $\mathsf{WH} : \{0,1\}^* \rightarrow \{0,1\}^d$ that takes as input a binary string of arbitrary (finite) length and returns a d-length random looking binary string. The customer requires WH to have the following properties: (a) WH has to be deterministic, i.e., for the same input it has to generate the same output; (b) WH has to ensure that for the first 2^d calls to WH with **distinct** values there will be no collision after which there can be collisions. Give a design of WH. Note that, efficiency is not a requirement. Justify why such a function will not work well as a typical hash function does in practice.

Hint: The problem is not as difficult as it looks. What if you can have some auxiliary space, can you design it?

Answer: One solution is to keep track of the first 2^d queries asked along with a counter. For example, we can construction WH as follow:

- Choose some fixed key $k \leftarrow \{0,1\}^d$. This is for random look string.
- For first 2^d distinct queries, $\mathsf{WH}(q_i) = (i-1) \oplus k$ where q_i denotes query i. Store $(q_i, (i-1) \oplus k)$ into a look up table.
- For all other queries q_j where $j > 2^d$, if q_j is in the look up table (i.e q_j has been queried before), thus $q_j = q_i$ for some i; return $((i-1) \oplus k)$. Else, return $(q_j \bmod 2^d) \oplus k$

However, this construction requires to store a look up table of 2^d rows. Therefore, it's not suitable for real world application for a large d

Question 4 (10 pts) Identity-Based Encryption (IBE) is a type of public-key encryption in which the public key of a user is some unique information about the identity of the user (e.g. a user's email address). Identity-based systems allow any party to generate a public key from a known identity value such as an ASCII string. A trusted third party, called the Private Key Generator (PKG), generates the corresponding private keys. To operate, the PKG first publishes a master public key, and retains the corresponding master private key (referred to as master key). Given the master public key, any party can compute a public key corresponding to the identity ID by combining the master public key with the identity value. To obtain the private key corresponding to an identity ID string, the party authorized to use the identity ID contacts the PKG, which uses the master private key to generate the private key for identity ID.

Consider Kerberos, Public key infrastructure, and IBE systems. Each of the three systems can enable two users who did not know each other to communicate securely. All of them use a Trusted Third

Party: the Key Distribution Center (KDC) in Kerberos, Certificate Authorities (CAs) in public key certificates, and PKGs in IBE systems.

Compare these three systems along the following dimensions (1) System availability and communication overhead for large number of users; (2) The ease of having multiple TTPs in an Internet-scale system (consider how users under different TTPs can communicate). (3) The degree of trust one has to place in the TTPs for confidentiality.

Answer:

1. System availability and communication overhead for large number of users:
   - Kerberos: It can not scale in term of both availability and communication overhead because it has to maintain list of privates key for each parties to communicate with TTP.
   - CA: It can scale as the higher level CA can sign for lower level CA. The user relies on chain of signatures. Communication overhead is low.
   - PKG: It can scale in term of communication but not in term of availability. The reason is that we relies on single PKG to maintain private keys, IDs, and authorized users.
2. The ease of having multiple TTPs in an Internet-scale system
   - Kerberos: It's not easy to have multiple TTPs to communicate each other as it have to keep track list of private keys for different users.
   - CA: It's easy because root CA can sign certificate for other CA. Therefore, having multiple CAs does not affect because users only need to verify chain of signature.
   - PKG: It's not easy. It's not clear how different TTPs can communicate and maintain list of IDs and Private keys.
3. The degree of trust one has to place in the TTPs for confidentiality.
   - Kerberos: The degree of trust is high. Since there is no way to verify if the TTP does not maliciously generate and maintain private keys.
   - CA: The degree of trust is low. Because users can always verify with the true owner of the public key to make sure if CAs are malicious or not.
   - PKG: The degree of trust is high. Because we relies on PKG to generate private keys and authenticate authorized users to obtain secret keys, and there is not way to check if the PKG misbehave or not.

Problem 5 (8 pts) Given three boolean input variables x, y, and z, each taking values in $\{$TRUE, FALSE$\}$. For each of following, answer whether there is information flow in the non-deducibility sense. (Circle yes or no.)

- $w = (x \text{ OR } y) \text{ AND } z$

  yes / no there is information flow between x and w

Answer: No. There is no information flow between x and w. Consider $(y, z) = (\mathsf{false}, \mathsf{true})$, then for $\forall x, w \in \{f(x, \mathsf{false}, \mathsf{true})\} = \{\mathsf{true}, \mathsf{false}\}$

yes / no there is information flow between $\{x, y\}$ and w

Answer: Yes. There is information flow between $\{x, y\}$ and w. Consider $w = \mathsf{true}$, then we know that $(x, y) \neq (\mathsf{false}, \mathsf{false})$

- $w = x$ XOR y XOR z

  yes / no there is information flow between $\{x, y\}$ and w

  Answer: No. There is no information flow between $\{x, y\}$ and w. It's one-time pad. Given any value of w, we cannot rule out any posibility for $\{x, y\}$

  yes / no there is information flow between $\{y, z\}$ and w

  Answer: No. There is no information flow between $\{y, z\}$ and w. It's one-time pad. Given any value of w, we cannot rule out any posibility for $\{y, z\}$

Problem 6 (10 pts) Assume that x, y, z are variables that take values either 0 or 1. Answer the following questions.

1. Give a deterministic function $f(x, y, z)$ such that $w = f(x, y, z)$ satisfies the following conditions: There **exists no** information flow in the non-deducibility sense between x and w, between y and w, between z and w, between $(x+y)$ and w, between $(x+z)$ and w, and between $(y+z)$ and w. But there **exists** information flow in the non-deducibility sense between $(x + y + z)$ and w.

   Answer: We let $w = f(x, y, z) = x + y + z \mod 2$. It's not difficult to see that:
   - There is no information flow between x and w, between y and w, between z and w, between $(x + y)$ and w, between $(x + z)$ and w, and between $(y + z)$ and w. This is similar to the second part of problem 5.
   - There is information flow between $(x + y + z)$ and w: For example, $w = 1$,then we learn that $x + y + z \neq 0$ and 2

2. Give a deterministic function $f(x, y, z)$ such that $w = f(x, y, z)$ satisfies the same conditions as above, except that now we require that there **exists** information flow in the non-deducibility sense between $(y + z)$ and w.

   Answer: We let $w = f(x, y, z) = y + z \mod 2$. It's not difficult to see that:
   - There is no information flow between x and w, between y and w, between z and w, between $(x + y)$ and w, between $(x + z)$ and w, and between $(y + z)$ and w..
   - There is information flow between $(x + y + z)$ and w: For example, $w = 1$, then we learn that $x + y + z \neq 0$

- There is information flow between $(y + z)$ and w: For example, $w = 1$, then we learn that $y + z \neq 0$ and 2

Problem 7 (10 pts) The Woo-Lam Protocol is an authentication protocol using symmetric encryption and trusted third party Trent.

Alice and Trent share a symmetric key K_{AT};

Bob and Trent share a symmetric key K_{BT}.

The protocol is as follows:

1. Alice $\rightarrow$ Bob: $Alice$;
2. Alice $\leftarrow$ Bob: N_B;
3. Alice $\rightarrow$ Bob: $E_{K_{AT}}[N_B]$;
4. Trent $\leftarrow$ Bob: $Bob, E_{K_{BT}}[Alice, E_{K_{AT}}[N_B]]$;
5. Trent $\rightarrow$ Bob: $E_{K_{BT}}[N_B]$;
6. Bob decrypts what he receives in step 5 using K_{BT}, and accepts if the encryption returns his nounce sent in step 2 correctly; he rejects otherwise.

Assume that Carl and Trent also share a symmetric key K_{CT}.

Describe a parallel-session attack in which Carl starts two sessions with Bob (one as Carl and one faking as Alice) and can eventually make the faking session with Bob succeeds, i.e., Bob believes that he is talking with Alice in that session. Describe the message sequences in the attack.

Hint: Assume that the communication between Trent and Bob is connection-less (e.g., through UDP); in other words, when Bob sends two messages in two sessions to Trent and receives two replies, Bob cannot link a reply with a particular session; he can only try to decrypt and see whether the reply is meaningful for that session. In this case, Bob will accept in a session when one of the replies is correct for that session.

Answer: Here is the attack:

1. Carl $\rightarrow$ Bob: $Alice$;
2. Carl $\rightarrow$ Bob: $Carl$;
3. Carl $\leftarrow$ Bob: N_{AB};
4. Carl $\leftarrow$ Bob: N_{CB};
5. Carl $\rightarrow$ Bob: $E_{K_{CT}}(N_{CB})$;
6. Carl $\rightarrow$ Bob: $E_{K_{CT}}(N_{AB})$;
7. Trent $\leftarrow$ Bob: $Bob, E_{K_{BT}}(Alice, E_{K_{CT}}(N_{CB}))$;
8. Trent $\leftarrow$ Bob: $Bob, E_{K_{BT}}(Carl, E_{K_{CT}}(N_{AB}))$;
9. Trent $\rightarrow$ Bob: $E_{BT}(D_{K_{AT}}(E_{K_{CT}}(N_{CB})))$; // After decrypting, Bob will discard this message
10. Trent $\rightarrow$ Bob: $E_{BT}(N_{AB})$; // Bob will accept this message.

Thus, in the end of the protocol, Carl succeeds in impersonating Alice because Bob can only decrypt the challenge N_{AB}.

Fix: In step 5 of the proposed protocol, Trent should include $E_{K_{BT}}(Alice, N_B)$. The same attack will not be possible.

Question 8 (20 pts) Notions for confidentiality.

We model an encryption scheme as a deterministic function enc that takes three inputs: (key, IV, plaintext) and outputs a ciphertext, i.e., $CT = \mathsf{enc}(K, IV, PT)$. We assume that all inputs are binary strings of certain lengths, and the key and IV are always chosen at uniform random. Intuitively, when one wants to encrypt a plaintext, one randomly chooses a key and an IV, and then feeds the three inputs into the function enc, and obtains the output ciphertext. It is required that there exists a deterministic decryption function dec such that

$$\forall_K \, \forall_{IV} \, \forall_{PT} \, \mathsf{dec}(K, \mathsf{enc}(K, IV, PT)) = PT.$$

We have studied four security notions for confidentiality of the plaintext given that the ciphertext is public, and we limit our attention to the case that each key is used to encrypt a single message: (1) perfect secrecy (information theoretical security); (2) computational indistinguishability as in IND-CPA security (though Chosen Plaintext Attack is not applicable here because one key encrypts a single message); (3) non-interference between plaintext and ciphertext; and (4) non-deducibility between plaintext and ciphertext.

1. **[4 pts]** Provide the definition the the above four notions of security. (Note that for computational indistinguishability, you need to adapt IND-CPA to match the current situation in which one key encrypts a single message.)

Answer: $\mathcal{M}, \mathcal{C}, \mathcal{K}, \mathcal{IV}$ are message space, ciphertext space, key space, and IV space.

(a) Perfect secrecy:

$$\forall pt, pt' \in \mathcal{M}, \forall ct \in \mathcal{C} : \mathbf{Pr}[\mathsf{enc}(K, IV, pt) = ct] = \mathbf{Pr}[\mathsf{enc}(K, IV, pt') = ct]$$

or

$$\forall pt \in \mathcal{M}, \forall ct \in \mathcal{C} \text{ for which } \mathbf{Pr}[C = ct] \neq 0 : \mathbf{Pr}[M = pt | C = ct] = \mathbf{Pr}[M = pt]$$

where the probabilities are over choice of K and IV.
Also, given a ciphertext, it can be an encryption of any plaintext with equal probability.

(b) IND-CPA:

$$\forall pt, pt' \in \mathcal{M}, \forall ct \in \mathcal{C} : |\mathbf{Pr}[\mathsf{enc}(K, IV, pt) = ct] - \mathbf{Pr}[\mathsf{enc}(K', IV', pt') = ct]| \leq \mathsf{negl}$$

where the probabilities are over choice of K and IV, negl is a negligible probability.
In otherword, there is some small probability that an adversary can distinguish the ciphertexts.

(c) Non-deducibility:

$$\forall pt \in \mathcal{M}, \forall ct \in \mathcal{C} \text{ for which } \mathbf{Pr}[C = ct] \neq 0 : \mathbf{Pr}[M = pt | C = ct] \neq 0$$

or

$$\forall pt \in \mathcal{M}, \forall ct \in \mathcal{C}, \forall K \in \mathcal{K}, \forall IV \in \mathcal{IV} : \mathbf{Pr}[\mathsf{enc}(K, IV, pt) = ct] \neq 0$$

It also means that given ciphertext, the adversary cannout rule out any plaintext.

(d) Non-interference:

$$\forall pt, pt' \in \mathcal{M}, \forall K \in \mathcal{K}, \forall IV \in \mathcal{IV} : \mathsf{enc}(K, IV, pt) = \mathsf{enc}(K, IV, pt')$$

Also, all encryptions lead to same ciphertext.

2. **[3 pts]** One of these four notions is impossible to achieve for the purpose of encryption. Which one is it? Why is it impossible to achieve?

Answer: Non-interference is impossible to achieve because it's impossible to obtain the same ciphertext for any plaintext with random key and iv.

3. **[5 pts]** For each of the remaining 3 notions, in order to satisfy it, is the key length required to be at least as long as the plaintext length? If so, give a proof that it is required.

Answer:

- Perfect Secrecy: Yes. the key length required to be at least as long as the plaintext length. We want to show that $|\mathcal{M}| > |\mathcal{K}|$, then we cannot achieve perfect secrecy. Given some ciphertext $ct \in \mathcal{C}$, we can obtain a set $\mathcal{M}(ct)$ to be the set of all plaintext by applying $\mathsf{dec}(K, IV, ct)$ for all $K \in \mathcal{K}$, we can ignore IV. Thus, $|\mathcal{M}(ct)| < |\mathcal{M}|$, therefore, there exists $pt \in \mathcal{M}$ and $pt \notin \mathcal{M}(ct)$. Thus, it means:

  $$\mathbf{Pr}[M = pt | C = ct] = 0 \neq \mathbf{Pr}[M = pt]$$

  Therefore, not perfect secrecy.
- IND-CPA: No. if there exists a pseudorandom function, then we can construct IND-CPA scheme. Schemes like AES-CTR mode are IND-CPA, and the key is fixed length.
- Non-Deducibility: Yes. Using the same argument as we did for perfect secrecy, one can show that $|\mathcal{K}| > |\mathcal{M}|$ is necessary for non-deducability. Otherwise, there exists a plaintext pt and ciphertext ct such that:

  $$\mathbf{Pr}[M = pt | C = ct] = 0$$

4. **[4 pts]** Give an encryption function that satisfies computational indistinguishability, but violates non-deducability.

Answer: Without reusing IV, AES-CTR or CBC encryption modes are proven to be IND-CPA secure; however, it violates non-deducability because adversary can brute-force the key space. Because the key space is smaller than the message space, given a ciphertext, a computational unbounded adversary can try to decrypt using all keys, and he knows for sure which message are not decryption of such ciphertext

5. **[4 pts]** Give an encryption function that satisfies non-deducability security, but violates computational indistinguishability.

Answer: Consider the encryption function such that $\mathsf{enc}(\mathsf{k}, \mathsf{iv}, \mathsf{pt}) = k \oplus pt$ where $\mathcal{K} = \mathcal{M} = \mathcal{C} = \{0,1\}^n$; we ignore iv in this case. However, we change that $\mathbf{Pr}[k = 0^n] = 0.999 \cdots = 1 - \mathsf{negl}$. Given the cipher text $ct \in \mathcal{C}$, any message has a nonnegative probability that it is the decryption of ct. However, it violates the IND-CPA property; consider pt'. Then, $\mathbf{Pr}[\mathsf{enc}(K, IV, pt') = pt'] = 1 - \mathsf{negl}$. Thus, one can easily distinguish between encryption of 2 different messages.

Problem 9 (16 pts) Read the article "New Directions in Cryptography" by Diffie and Hellman (available from the lectures & handouts page), and answer the following questions. [To get a good grade in this question make sure you answer is short and directly to the point. Bullet-points answers are recommended.]

a (4 pts) The paper gives rationales for building encryption schemes that are secure against known plaintext attacks and chosen plaintext attacks, by discussing how such schemes remove restrictions that are placed on the ways of using them. Discuss these rationale in your own words.

Answer:
Secure against known plaintext attacks:

- A system which is secure against a known plaintext attacks frees its users from the need to keep its past message secret.
- Such system is suitable for commercial situations where products or presses are sent in encrypted form for later public disclosure

Secure against chosen plaintext attacks:

- Such system frees its users from concerning over whether their opponent can plant messages in their system. For example, submitting a proposal to a competitor may result in his enciphering it for transmission to his headquarters.

b (4 pts) List all the limitations and shortcomings discussed in the paper about symmetric encryption schemes.

Answer:

- Require the existence of a secure channel for key exchange (e.g. weekly courier). This is not suitable because it's unrealistic to postpone a business contract until there is a secure channel.
- Not suitable for large number of users because there must exist a secure channel between each pair of users ($(n^2 - n)/2$ pairs for n users).

- Provable secure schemes like one time pad are not practical.

c (4 pts) List all the limitations and shortcomings discussed in the paper about symmetric message authentication schemes.

Answer:

- Similarly, traditional mac requires secure channel for key agreement.
- Symmetric MAC schemes do not protect against disputes between transmitter and receiver (non-repudiation).
- Symmetric MAC schemes cannot meet the need for a digital, unforgeable, message independent signature (i.e. similar to written signature).

d (4 pts) The paper establishes the relationships among (1) public-key encryption, (2) public key distribution, and (3) digital signature (referred to in
the paper as one-way authentication). By relationships, we mean it is possible to use one scheme to implement another. List these relationships, and explain the constructions involved to use one scheme to implement another.

Answer: Two relationships:

- $(1) \rightarrow (3)$: *A public key cryptosystem can be used to generate a one-way authentication system:*
  Intuitively, if the public key crypto system exists, it can be used to build one-way function. Using such one-way function, one can build a signature scheme like Lamport one-time signature construction. Also, it is mentioned in the paper that public-key system allows to construction trapdoor one-way functions. Those whose have knowledge of trapdoor information can efficiently compute the inverse. Thus, it can be used to build signature scheme.
- $(1) \rightarrow (2)$ *A trapdoor cryptosystem can be used to produce a public key distribution system:*
  As described above, a public key cryptosystem implies the existence of a trapdoor one-way function (can be inverted with the knowledge of trapdoor information). Thus, for A and B to establish a common key. A chooses a key at random and sends an arbitrary plaintext-cryptogram to B. B makes trapdoor cipher public but keep trapdoor information secret, uses the plaintex-cryptogram pair to solve for the key. A and B now can have key in common.