Tutorial 2 - Solutions

Computer Security
School of Informatics
University of Edinburgh

In this second tutorial for the Introduction to Computer Security course we cover Cryptography and Cryptographic protocols. The tutorial consists of questions from past years exams.

You are free to discuss these questions and their solutions with fellow students also taking the course, and also to discuss in the course forum. Bear in mind that if other people simply tell you the answers directly, you may not learn as much as you would by solving the problems for yourself; also, it may be harder for you to assess your progress with the course material

1 Hash functions

Let $\mathcal{M} = \{0,1\}^*$ and $\mathcal{T} = \{0,1\}^n$ for some integer n.

1. Explain what does it mean for a hash function $h : \mathcal{M} \to \mathcal{T}$ to be one-way.

   Solution

   A function h is a one-way function if for all x there is no efficient algorithm which given $h(x)$ can compute x

2. Explain what does it mean for a hash function $h : \mathcal{M} \to \mathcal{T}$ to be collision resistant.

   Solution

   A function h is collision resistant if there is no efficient algorithm that can find two messages m_1 and m_2 such that $h(m_1) = h(m_2)$.

3. Suppose $h : \mathcal{M} \to \mathcal{T}$ is collision resistant. Is h also one-way? If so, explain why. If not, give an example of a collision resistant function that is not one-way.

   Solution

   Let g be a hash function which is collision resistant and maps arbitrary-length inputs to $n-1$-bit outputs. Consider the function h defined as:

   $$h(x) = \begin{cases} 1||x & \text{if } x \text{ has bitlength } n-1 \\ 0||g(x) & \text{otherwise} \end{cases}$$

   where $||$ denotes concatenation. Then h is an n-bit hash function which is collision resistant but not one-way. As a simpler example, the identity function on fixed-length inputs is collision resistant but not one way.

4. Suppose $h : \mathcal{M} \to \mathcal{T}$ is one-way. Is H also collision resistant? If so, explain why. If not, give an example of a one-way function that is not collision resistant.

Solution

Let h be the function $h(x) = \bar{0}$ (where $\bar{0}$ denotes 0 encoded over n bits). This function is trivially one way since it maps any input to the same value $\bar{0}$, but for the same reason it is also not collision resistant.

5. Let p be a prime number and g a generator of $\mathbb{Z}_p^*$. Consider the function $h : \mathbb{Z} \to \mathbb{Z}_p^*$ where $h(m) = g^m \bmod p$.

   (a) Is h collision resistant? Explain your answer.

   Solution

   No. We know that for all k $g^{m+k(p-1)} \equiv g^m \pmod{p}$

   (b) If we assume the difficulty of the discrete logarithm problem in $\mathbb{Z}_p^*$, can you explain why this function is one way?

   Solution

   It can be shown that h satisfies one wayness with a reduction from the discrete logarithm problem. The reduction is trivial in that they are almost exactly the same problem. If you have an algorithm which can produce preimages, you need only reduce them modulo p to produce the correct answer for the discrete logarithm problem.

6. Bob is on an under cover mission for a week and wants to prove to Alice that he is alive each day of that week. He has chosen a secret random number, s, which he told to no one (not even Alice). But he did tell her the value $H = h(h(h(h(h(h(h(s)))))))$, where h is a cryptographic hash function. During that week Bob will have access to a broadcast channel, so he knows any message he sends to Alice will be received by Alice. Unfortunately Bob knows that Eve was able to intercept message H. Explain how Bob can broadcast a single message everyday that will prove to Alice that he is still alive. Note that your solution should not allow anyone (and in particular Eve) to replay any previous message from Bob as a (false) proof that he still is alive.

Solution

Let d range from 1 to 7 and denote the day of the week. On day d, Bob broadcasts message $h^{7-d}(s)$. Because of one wayness of h, from previous seen messages $h^7(s), \ldots, h^{7-d+1}(s)$ no one else can compute $h^{7-d}(s)$ but Bob. But anyone (and in particular Alice) can verify that $h^{7-d}(s) = h(h^{7-d+1}(s))$ that is the message received on day $d-1$ is the hash of the message received on day d, proving that Bob is alive.

2 Symmetric encryption

Let $(\mathcal{E}_{32}, \mathcal{D}_{32})$ be a secure (deterministic) block cipher with 32-bits key size and 32-bits message size. We want to use this cipher to build a new (deterministic) block cipher $(\mathcal{E}_{64}, \mathcal{D}_{64})$ that will encrypt 64-bits messages under 64-bits keys. We consider the following encryption algorithm. To encrypt a message M under a key K, we split M into two parts M_1 and M_2, and we also split K into two parts K_1 and K_2. The ciphertext C is then computed as $\mathcal{E}_{32}(K_1, M_1)||\mathcal{E}_{32}(K_2, M_2)$. In other words we concatenate the encryption of M_1 under K_1 using $\mathcal{E}_{32}$, with the encryption of M_2 under K_2 using $\mathcal{E}_{32}$.

1. What is the corresponding decryption algorithm? To justify your answer prove that the consistency property is satisfied.

   Solution

   We just split C into two parts C_1 and compute the underlying plaintext as $\mathcal{D}_{32}(K_1, C_1)||\mathcal{D}_{32}(K_2, C_2)$. The proof of consistency is trivial given the consistency of $(\mathcal{E}_{32}, \mathcal{D}_{32})$. Indeed

$$\begin{aligned}\mathcal{D}_{64}(K_1||K_2, \mathcal{E}_{64}(K_1||K_2, M_1||M_2)) &= \mathcal{D}_{64}(K_1||K_2, \mathcal{E}_{32}(K_1, M_1)||\mathcal{E}_{32}(K_2, M_2)) \\ &= \mathcal{D}_{64}(K_1, \mathcal{E}_{32}(K_1, M_1))||\mathcal{D}_{64}(K_2, \mathcal{E}_{32}(K_2, M_2)) \\ &= M_1||M_2\end{aligned}$$

2. Consider the following game.

   - In a first phase, the attacker choses a few plaintext messages $M_1, \ldots, M_n$ and gets back the corresponding ciphertexts $C_1, \ldots, C_n$ under some key K that he does not know. The attacker gets to know that C_1 is the ciphertext corresponding to $M_1, \ldots,$ C_n is the ciphertext corresponding to M_1.
   - In a second phase the attacker builds two messages M and M' and gets back C which is the encryption under K either of M or M'. But now, the attacker doesn't know if the plaintext underlying C is M or M' and has to guess it.

   Informally, a symmetric cipher is said to be subject to a chosen plaintext attack if the attacker can guess (with high probability) which of M or M' is the plaintext corresponding to C. Show that the new cipher $(\mathcal{E}_{64}, \mathcal{D}_{64})$ is subject to a chosen plaintext attack even though $(\mathcal{E}_{32}, \mathcal{D}_{32})$ is not.

   Solution

   Let $M_1 = 0^{32}||0^{32}$ and $M_2 = 1^{32}||1^{32}$. Let $C_1 = \mathcal{E}_{32}(K_1, 0^{32})||\mathcal{E}_{32}(K_2, 0^{32})$ and $C_2 = \mathcal{E}_{32}(K_1, 1^{32})||\mathcal{E}_{32}(K_2, 1^{32})$, and let $M = 0^{32}||1^{32}$ and $M' = 1^{32}||0^{32}$. Given C_1 and C_2 the attacker can trivially compute $\mathcal{E}_{64}(0^{32}||1^{32}) = \mathcal{E}_{32}(K_1, 0^{32})||\mathcal{E}_{32}(K_2, 1^{32})$ and $\mathcal{E}_{64}(1^{32}||0^{32}) = \mathcal{E}_{32}(K_1, 1^{32})||\mathcal{E}_{32}(K_2, 0^{32})$, and thus win the game with probability 1. Thus this new scheme is not secure under chosen plaintext attack.

3. A symmetric cipher is said to be vulnerable to a know plaintext attack if given a plaintext message M and its corresponding ciphertext C under some key K not known to the attacker, the attacker can recover the key K in a reasonable amount of time (that is significantly less than by a brute force-attack). Show that $(\mathcal{E}_{64}, \mathcal{D}_{64})$ is subject to a known plaintext attack.

Solution

(a) A brute force attack consists in trying all the possible keys in the key space. So for the new algorithm the brute-force attack has complexity 2^{64}

(b) First, use brute-force to recover K_1. We know that $C_1 = \mathcal{E}_{32}(K_1, M1)$, and we know M_1, C_1, so try all possibilities for K_1 and see which one is consistent with this equation. Next, use brute force to recover K_2, by a similar method. This requires $2^{32}+2^{32} = 2^{33}$ trial decryptions in total, which is easily feasible. That means our attack has complexity 2^{33}.

3 Encryption

One-time pads Inspired by the one-time pad, Alice decides to design her own protocol to confidentially send messages to Bob. Alice's protocols works as follows:

- When Alice is ready to send her message $M \in \{0,1\}^\ell$, she randomly selects $K_A \in \{0,1\}^\ell$, and sends to Bob the message $M_1 = M \oplus K_A$.
- Bob then randomly selects $K_B \in \{0,1\}^\ell$ and sends to Alice the message $M_2 = M_1 \oplus K_B$.
- Next, Alice computes $M_3 = M_2 \oplus K_A$ and sends it to Bob.
- Bob may now retrieve the message M.

1. Show that $M = M_3 \oplus K_B$.

Solution

The relies only on commutativity and associativity of $\oplus$:

$$\begin{array}{rl} & M_3 \oplus K_B \\ = & (M_2 \oplus K_A) \oplus K_B \\ = & ((M_1 \oplus K_B) \oplus K_A) \oplus K_B \\ = & (((M \oplus K_A) \oplus K_B) \oplus K_A) \oplus K_B \\ = & M \end{array}$$

2. This protocol is insecure. Show that Eve can retrieve any message intended for Bob.

Solution

The attacker can mount a MITM attack pretending to be Bob since messages coming from either Alice or Bob are not authenticated.

$$
\begin{array}{ccc}
A & & E \\
 & \xrightarrow{\quad M \oplus K_A \quad} & \\
 & \xleftarrow{\quad (M \oplus K_A) \oplus K_e \quad} & \\
 & \xrightarrow{\quad ((M \oplus K_A) \oplus K_E) \oplus K_A \quad} &
\end{array}
$$

As we just saw the last message received by Eve is nothing more than $M \oplus K_E$ which Even can decrypt since she knows K_E

ElGamal

3. Recall the details of the ElGamal encryption scheme seen in class.

Solution

- Fix prime p, and generator $g \in (\mathbb{Z}_p)^*$
- $\mathcal{M} = \{0, \ldots, p\text{-}1\}$ and $\mathcal{C} = \mathcal{M} \times \mathcal{M}$
- $G_{EG}() = (pk, sk)$ where $pk = g^d \pmod p$ and $sk = d$ and $d \xleftarrow{r} \{1, \ldots, p\text{-}2\}$
- $E_{EG}(pk, x) = (g^r \pmod p),\ m \cdot (g^d)^r \pmod p)$ where $pk = g^d \pmod p$ and $r \xleftarrow{r} \mathbb{Z}$
- $D_{EG}(sk, x) = e^{\text{-}d} \cdot c \pmod p$ where $x = (e, c)$
- Consistency: $\forall (pk, sk) = G_{EG}(), \forall x, D_{EG}(sk, E_{EG}(pk, x)) = x$

  Proof: Let $pk = g^d \pmod p$ and $sk = d$

$$
\begin{aligned}
D_{EG}(sk, E_{EG}(pk, x)) &= (g^r)^{\text{-}d} \cdot m \cdot (g^d)^r \pmod p \\
&= m \pmod p
\end{aligned}
$$

4. Assume you are given an ElGamal public key pk (but not the corresponding private key). Assume you are also given the ciphertexts $c_a = E(pk, m_a)$ and $c_b = E(pk, m_b)$ corresponding to the encryption using ElGamal of messages m_a and m_b under pk respectively. But you are not given m_a nor m_b. Show that how you can construct a ciphertext which is a valid ElGamal encryption under the key pk of the message $m_a \cdot m_b$ (mod p).

Solution

By definition of ElGamal, there exists r_a and r_b such that

$$\begin{array}{rclcl} c_a & = & (c_a^1, c_a^2) & = & (g^{r_a} \pmod p),\ m_a \cdot (g^d)^{r_a} \pmod p)) \\ c_b & = & (c_b^1, c_b^2) & = & (g^{r_b} \pmod p),\ m_b \cdot (g^d)^{r_b} \pmod p)) \end{array}$$

But then by the properties of modular arithmetic we can compute

$$\begin{array}{rcl} c_a^1 \cdot c_b^1 & = & (g^{r_a + r_b} \pmod p) \\ c_b^1 \cdot c_b^2 & = & m_a \cdot m_b \cdot (g^d)^{r_b + r_b} \pmod p)) \end{array}$$

And thus the ciphertext $c = (c_a^1 \cdot c_b^1, c_b^1 \cdot c_b^2)$ which corresponds to the ElGamal encryption of $m_a \cdot m_b \pmod p$ under pk.

5. Assume you are given an ElGamal public key pk (but not the corresponding private key) and a ciphertext $c = E(pk, m)$ which is the ElGamal encryption of some unknown message m under pk. You are furthermore given access to an oracle that will decrypt any ciphertext other than c. ElGamal is said to be vulnerable to a chosen ciphertext attack if you can retrieve m. Show that ElGamal is indeed vulnerable to a chosen ciphertext attack.

Solution

Let $c = (c_1, c_2)$ be the ElGamal encryption of some unknown message m under pk. We can compute the ElGamal encryption of 2 under pk. Let $c' = (c'_1, c'_2)$ be the encryption of 2 under pk. We just saw to the previous question that we can compute $c'' = (c_1 \cdot c'_1, c_2 \cdot c'_2)$ without knowing m and which is the encryption of $m \cdot 2$ (mod p). Now using the decryption oracle we can obtain $m \cdot 2$ (mod p). Finally since 2 and p are coprime, 2 admits an inverse mod p which we can compute and devide $m \cdot 2$ (mod p) by 2 to retrive m.

4 The Diffie-Hellman protocol

In class, we saw the Diffie-Hellman protocol, which is a two-party key establishment protocol secure against passive attackers. However, as we saw, the Diffie-Hellman protocol is insecure against active attackers. Indeed, a malicious agent can mount a man-in-the-middle attack to learn a key not intended for him. This attack is possible because their is no mechanism to authenticate the two parties to one another. We consider the following extension of the Diffie-Hellman protocol to thwart this attack. We assume that the parties A and B have a private signing key sk_A and sk_B respectively, and a certificate on the corresponding public key CERT_A and CERT_B respectively signed by a common Trusted Third Party.

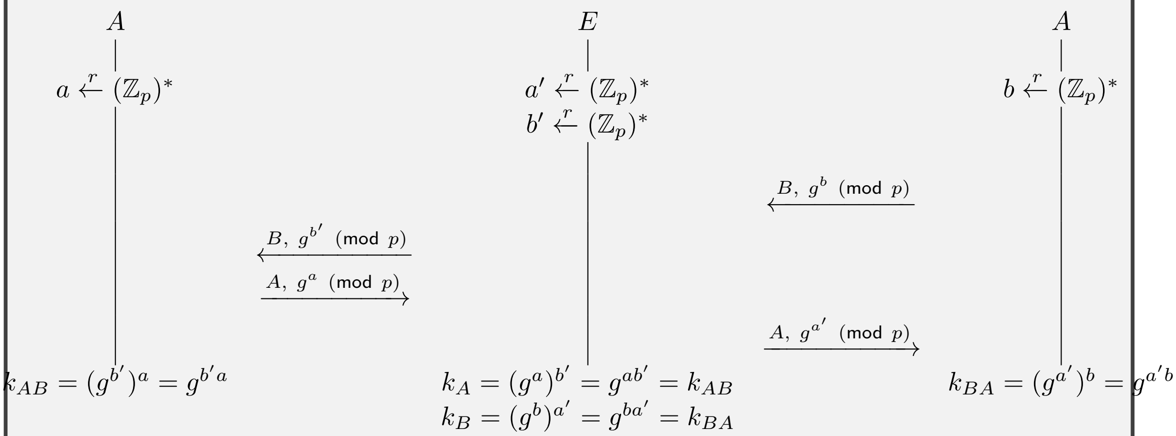
$$\begin{array}{ccc} A & & B \\ & \xrightarrow{\quad\quad\quad g^x \quad\quad\quad} & \\ & \xleftarrow{g^y,\ B,\ \mathsf{CERT}_B, \mathsf{sig}(\mathsf{sk}_B, (g^x, g^y))} & \\ & \xrightarrow{A,\ \mathsf{CERT}_A, \mathsf{sig}(\mathsf{sk}_A, (g^x, g^y))} & \end{array}$$

The result is a shared secret $K_{AB} = g^{xy}$ from which the parties derive a session-key.

1. Briefly explain the purpose of the signatures in the protocol above. How does it defend against the attack discussed in class?

Solution

The original Diffie-Hellman has no authentication mechanism to ensure the two parties that they are indeed talking to each other. In class, we saw that the DH protocol is subject to the following man in the middle attack



where Eve has caused

- A to think that she is communicating securely with B and that they have both agreed to the key k_{AB};
- B to think that she is communicating securely with A and that they have both agreed to the key k_{BA};
- Eve has learned the keys k_{AB} and k_{BA} which were intended to remain secret from her.

In the variant proposed in the statement of Problem 2, A and B sign their view on k_{AB} and k_{BA}. Now, because Eve cannot forge A or B's signature she cannot mount the attack on the original DH protocol on this variant of the protocol. In particular, she cannot sign with the secret signing key of A the message $(g^{a'}, g^b)$. In other words she cannot build message $\mathsf{sign}(\mathsf{sk}_A, (g^{a'}, g^b))$. Similarly, she cannot sign with the secret signing key of B the message $(g^a, g^{b'})$. In other words she cannot build message $\mathsf{sign}(\mathsf{sk}_A, (g^a, g^{b'}))$.

2. Show that an active man-in-the-middle, Eve, can cause:
   - A to think that she is communicating securely with B (as required),
   - but B to think he is communicating securely with Eve.

   In other words, B is fooled into thinking that the subsequent encrypted messages he is receiving (from A) are coming from Eve. Note that Eve cannot eavesdrop on the resulting encrypted channel.

Solution

If Eve intercepts the third message in an honest execution of the protocol, and replaces it with the following message:

$$E,\ \mathsf{CERT}_E,\ \mathsf{sig}(\mathsf{sk}_E, (g^x, g^y))$$

which she can because she can obtain g^x and g^y from the first to messages of the session, then

- A will think that she is communicating securely with B (as required),
- but B will think he is communicating securely with Eve.

This is possible because in the first two messages g^x and g^y are not linked to A and B in a secure way.

3. Describe how Eve can use this attack to steal money from A. For example, suppose A gives expert advice in a private chat room run by B, and that she gets paid for that.

Solution

Eve could also register as an expert on Bob's private chat to sell her advice. Then she could just relay to A the messages sent from B to her. A will accept these messages as coming from B for her and will reply with her advice. Now Eve, will intercept A's responses and relay them to B as if coming from herself and will get paid for the advice in place of A.

4. Propose a way to fix the protocol to defend against this attack. Explain why your fix prevents the attack from Question 2.

Solution

To fix this problem, A and B need to link g^x and g^y to the two parties of this protocol. This could be achieved as follows

$$\begin{array}{ccc}
A & & B \\
 & \xrightarrow{\quad g^x \quad} & \\
 & \xleftarrow{\quad g^y,\ B,\ \mathsf{CERT}_B,\ \mathsf{senc}(g^x \cdot g^y, \mathsf{sig}(\mathsf{sk}_B, (g^x, g^y))) \quad} & \\
 & \xrightarrow{\quad A,\ \mathsf{CERT}_A,\ \mathsf{senc}(g^x \cdot g^y, \mathsf{sig}(\mathsf{sk}_A, (g^x, g^y))) \quad} &
\end{array}$$

Note that the resulting protocol is the Station-to-Station protocol seen in class.

5 Authentication and key-agreement protocol

Consider the following two-party authentication and key agreement protocol. Alice (A) and Bob (B) want to establish a session key using a long-term symmetric key K_{AB}. First Alice generates a nonce N_A and sends it along with her identity to Bob. Bob generates his own nonce N_B and sends it together with the encryption of Alice's nonce under the long-term key K_{AB}.

Alice acknowledge receipt of this message by sending the encryption of Bob's nonce under the long-term key. Finally Bob generates the session key k and sends it to Alice encrypted under K_{AB}.

$$\begin{array}{ccc} A & & B \\ & \xrightarrow{\quad A,\ N_A \quad} & \\ & \xleftarrow{\{N_A\}_{K_{AB}},\ N_B} & \\ & \xrightarrow{\quad \{N_B\}_{K_{AB}} \quad} & \\ & \xleftarrow{\quad \{k\}_{K_{AB}} \quad} & \end{array}$$

1. This protocol is flawed. Show how Eve could learn a session key that Alice thinks she has securely established with Bob. (You will assume that nonces and keys have the same length)

Solution

The following diagram depicts such an attack.

$$\begin{array}{ccccc} A & & E & & B \\ & \xrightarrow{\quad A,\ N_A \quad} & & & \\ & & & \xrightarrow{\quad A,\ N_A \quad} & \\ & & & \xleftarrow{\{N_A\}_{K_{AB}},\ N_B} & \\ & \xleftarrow{\{N_A\}_{K_{AB}},\ N_B} & & & \\ & \xrightarrow{\quad \{N_B\}_{K_{AB}} \quad} & & & \\ & & & \xrightarrow{\quad \{N_B\}_{K_{AB}} \quad} & \\ & & & \xleftarrow{\quad \{k\}_{K_{AB}} \quad} & \\ & \xleftarrow{\quad \{N_A\}_{K_{AB}} \quad} & & & \end{array}$$

At this point A thinks she has securely established the key N_A with B.

2. Propose a way to fix the protocol to defend against this attack. Explain why your fix prevents this attack.

Solution

Of course having nonces and keys be of different size would thwart this attack. But several other possibilities too to fix this protocol. For example include N_A and/or N_B in the message that contains the key

$$
\begin{array}{ccc}
A & & B \\
 & \xrightarrow{A,\ N_A} & \\
 & \xleftarrow{\{N_A\}_{K_{AB}},\ N_B} & \\
 & \xrightarrow{\{N_B\}_{K_{AB}}} & \\
 & \xleftarrow{\{N_A, N_B, k\}_{K_{AB}}} &
\end{array}
$$

Or include static tags to distinguish the different messages

$$
\begin{array}{ccc}
A & & B \\
 & \xrightarrow{A,\ N_A} & \\
 & \xleftarrow{\{1, N_A\}_{K_{AB}},\ N_B} & \\
 & \xrightarrow{\{2, N_B\}_{K_{AB}}} & \\
 & \xleftarrow{\{3, k\}_{K_{AB}}} &
\end{array}
$$

If Alice and Bob do not share a long-term symmetric key they could use the following three-party authentication and key agreement protocol that relies on a trusted third party (TTP). Alice and Bob both share a long-term symmetric key K_{AT} and K_{BT} respectively with the TTP.

$$
\begin{array}{ccccc}
A & & TTP & & B \\
A & \longrightarrow & & \xrightarrow{A,\ N_A} & B \\
 & & TTP & \xleftarrow{B,\ N_B,\ \{A, N_A\}_{K_{BT}}} & B \\
A & \xleftarrow{\{N_A,\ k\}_{K_{AT}},\ \{A,\ k\}_{K_{BT}},\ N_B} & TTP & & \\
A & \longrightarrow & & \xrightarrow{\{A,\ k\}_{K_{BT}},\ \{N_B\}_k} & B
\end{array}
$$

3. This protocol is flawed. Show how Eve could learn a session key that Alice thinks she has securely established with Bob. (You will assume that nonces and keys have the same length)

Solution

The following diagram depicts such an attack.

$A \qquad\qquad TTP \qquad\qquad E$

$$A \xrightarrow{A,\ N_A} E$$

$$TTP \xleftarrow{E,\ N_E,\ \{A, N_A\}_{K_{ET}}} E$$

$$A \xleftarrow{\{N_A,\ k\}_{K_{AT}},\ \{A,\ k\}_{K_{ET}},\ N_E} TTP$$

$$A \xrightarrow{\{A,\ k\}_{K_{ET}},\ \{N_E\}_k} E$$

At this point A thinks she has securely established the key k with B.

4. Propose a way to fix the protocol to defend against this attack. Explain why your fix prevents this attack.

Solution

The identity of B should be included in the ciphertext from the TTP to A

$A \qquad\qquad TTP \qquad\qquad B$

$$A \xrightarrow{A,\ N_A} B$$

$$TTP \xleftarrow{B,\ N_B,\ \{A, N_A\}_{K_{BT}}} B$$

$$A \xleftarrow{\{N_A,\ B,\ k\}_{K_{AT}},\ \{A,\ B,\ k\}_{K_{BT}},\ N_B} TTP$$

$$A \xrightarrow{\{A,\ k\}_{K_{BT}},\ \{N_B\}_k} B$$

Similarily, to avoid an attack on Bob's perspective the identity of A and B should be included in the ciphertext from the TTP to B.