ECE596C: Handout #15

Key Distribution

Electrical and Computer Engineering, University of Arizona,
Loukas Lazos

Abstract. In this lecture we describe different key distribution schemes for establishing a shared secret among two or multiple parties. Particularly we present the Diffie-Hellman key distribution scheme and the Blom's key predistribution scheme.

1 Key Distribution Problem

The key distribution problem (KDP) is the problem of establishing and maintaining security associations (shared secrets) between a network of n users connected via insecure links. The KDP can be classified into the following two stages.

- *Initialization*: During the initialization phase, a trusted party also referred to as *Trusted Authority* (TA) distributes keying information to relevant parties in a secure manner. The keying material can be either used directly to encrypt data, or it can be used to establish new security associations.
- Maintenance: During the maintenance phase, the TA, or the network users themselves renew relevant cryptographic secrets to maintain secrecy of the communications (How is maintenance different from the initialization phase?)

For the initialization phase, key material can be either preloaded to the network entities before deployment, or can be issued using a temporary secure channel. Using the secure channel, the TA assigns long lived keys to network entities, that are usually only used to derive other keys and not to encrypt data (they are also called key encryption keys). Nodes who wish to securely communicate use their long lived keys to establish a session key that has a much shorter expiration time.

Question What determines the expiration time of a key? What are the advantages of having a session key?

Parameters considered when designing a key distribution scheme

- Resistance to node compromise and collusion.
- Communication cost for the TA and users.
- Storage cost for the users.
- Flexibility in establishing different types of keys.

1.1 Adversarial Model

Various types of attacks are considered, including both passive (inferring keying material by eavesdropping) and active ones. An adversary can:

- Alter messages while in transit.

- Inject fake messages into the network.
- Save transmitted messages and replay them at a later time.
- Attempt to impersonate one or more users in the network.

The goal of the adversary may be mutli-fold. To mention a few examples the adversary can:

- Fool two users U, V into accepting an invalid key as valid.
- Fool two users U, V into believing they have successfully established a shared key, while in reality they have not.
- Partially or fully recover a session, or a long-term key so as to compromise the confidentiality of the communications.

1.2 Diffie-Hellman Key Predistribution

The Diffie-Hellman key predistribution scheme relies on the difficulty of the Decision Diffie-Hellman problem we describe below.

The Diffie-Hellman Problems Computational Diffie-Hellman Problem (CDH)–Consider a mutliplicative group $(G, \cdot)$, an element $\alpha \in G$ of order n, and two elements $\beta, \gamma \in < \alpha >$ (recall that $< \alpha >$: $\{\alpha^i,\ 0 \leq i \leq n-1\}$). Find $\delta \in < a >$ such that $\log_\alpha \delta \equiv \log_\alpha \beta \log_\alpha \gamma \pmod n$. Equivalently given α^b and α^c find α^{bc}.

Decision Diffie-Hellman Problem (DDH)–Consider a mutliplicative group $(G, \cdot)$, an element $\alpha \in G$ of order n, and two elements $\beta, \gamma \in < \alpha >$. Is $\delta \in < a >$ such that $\log_\alpha \delta \equiv \log_\alpha \beta \log_\alpha \gamma \pmod n$. Equivalently given α^b and α^c, α^d, determine if $d \equiv bc \pmod n$.

It is easy to see that the DDH problem can be reduced to the CDH. Let an algorithm solve the CDH problem, i.e., find δ' such that $\log_\alpha \delta' \equiv \log_\alpha \beta \log_\alpha \gamma \pmod n$. Then check to see if $\delta' = \delta$.

Also the CDH problem can be reduced to the DLP as follows. Let an algorithm solve the DLP, i.e., for some α, β, γ it is easy to find $b = log_\alpha \beta$ and $c = \log_\alpha \gamma$ given only α^β and α^γ. Then it is easy to compute $d = bc \pmod n$ and $\delta = \alpha^d$.

Hence, solving the DDH problem implies that one can solve the CDH problem which in turn implies that one can solve the DLP problem. We now describe the Diffie-Hellman Key Predistribution Scheme (KPS).

1.3 Diffie-Hellman KPS

1. Consider a mutliplicative group $(G, \cdot)$, an element $\alpha \in G$ of order n, being publicly known.
2. User V computes a shared key $K_{U,V}$ as follows:

$$K_{U,V} = \alpha^{\alpha_U \alpha_V} = b_U^{\alpha_V}, \tag{1}$$

   where b_U denotes the public key of U and α_V denotes V's private key.
3. User U computes a shared key $K_{U,V}$ as follows:

$$K_{U,V} = \alpha^{\alpha_U \alpha_V} = b_V^{\alpha_U}, \tag{2}$$

   where b_V denotes the public key of V and α_U denotes U's private key.

Example: Let $G = \mathbb{Z}^*$, $p = 12987461$, $q = 1291$, primes with $p - 1 \equiv 0 \pmod{q}$ and $\alpha = 3606738$ denote the public key parameters. Then $\mathbb{Z}_p^*$ has order of q. Let U select $a_U = 357$ as the private key and compute the corresponding public key as:

$$b_U = \alpha^{\alpha_U} \mod p = 3606728^{357} \mod 12987461 = 7317197. \tag{3}$$

Let also V select $\alpha_V = 199$ as its private key and compute its public key as $b_V = \alpha^{\alpha_V} \mod p = 138432$. The two users U, V can individually compute a commonly shared key without the need for the exchange of any information via

$$K_{U,V} = b_V^{\alpha_U} \mod p = b_U^{\alpha_V} \mod p = 11829605. \tag{4}$$

Security of the Diffie-Hellman KPS Assume that a user $W \neq U, V$ wants to compute $K_{U,V} = \alpha^{\alpha_U \alpha_V}$, by knowing b_V, b_U but without knowing α_V, α_U. This is the definition of the CDH problem which with the right choices of public parameters and group G is considered to be intractable.

1.4 The Blom Key Predistribution Scheme

We first describe the Blom scheme resistant to the compromise of one user and then we extend to the compromise of k users.

1. Let p be a large prime publicly known and let the TA select an element $r_U \in \mathbb{Z}_p$ for each user U which also made public with $r_U \neq r_V$ for $U \neq V$.
2. The TA chooses three random elements $a, b, c \in \mathbb{Z}_p$ and forms the polynomial:

$$f(x, y) = a + b(x + y) + cxy \mod p. \tag{5}$$

   The polynomial f is symmetric, i.e., $f(x, y) = f(y, x)$.
3. For each user U, the TA computes $g_U(x) = f(x, r_U)$, and sends it to U over a secure channel. Note that g_U is linear in x and hence can be written in the form

$$g_U(x) = a_U + b_U x, \; a_U = a + br_U \mod p, \; b_U = b + cr_U \mod p. \tag{6}$$

4. If U, V want to communicate, they individually compute the common key

$$K_{U,V} = K_{V,U} = f(r_U, r_V) = a + b(r_U + r_V) + cr_U r_V \mod p. \tag{7}$$

   with user U computing

$$K_{U,V} = g_U(r_V) \tag{8}$$

   and V computing

$$K_{U,V} = g_V(r_U) \tag{9}$$

Security of Blom's scheme The security of the scheme relies on the fact that no individual user can derive the original polynomial f from its polynomial g.

Theorem 1. *The Blom KPS with $k = 1$ is unconditionally secure against any individual user.*

Proof. Let W attempt to compute $K_{U,V}$ which is given by $K_{U,V} = a + b(r_U + r_V) + cr_U r_V \mod p$. User W knows the coefficients of its own polynomial $a_W = a + br_w \mod p$, $b_W = b + cr_W \mod p$, and the publicly known values r_U, r_V Assume that W conjectures that $K_{U,V} = K^* \in \mathbb{Z}_p$. Then we can write a system of three equations with all the unknowns in a matrix form

$$\begin{pmatrix} 1 & r_U + r_V & r_U r_V \\ 1 & r_W & 0 \\ 0 & 1 & r_W \end{pmatrix} \begin{pmatrix} a \\ b \\ c \end{pmatrix} = \begin{pmatrix} K^* \\ a_W \\ b_W \end{pmatrix}. \tag{10}$$

The determinant of the coefficient matrix is $D = (r_W - r_U)(r_W - r_V)$ with calculations being done in $\mathbb{Z}_p$. Since by assumption $r_W \neq r_V \neq r_U$ and p is a prime the coefficient determinant is non-zero and hence there is a unique solution in $\mathbb{Z}_p$ for a, b, c. Therefore for each choice of K^* we derive a new triplet of a, b, c and W cannot compute $K_{U,V}$

Theorem 2. *Any coalition of two users can compute the keys for all the users of the networks*

Proof. The two users can construct a system of four equations with three unknowns and solve for a, b, c. Hence they can recover $f(x, y)$ and given that r_U are public compute any key $K_{U,V}$.

Generalization of Blom's scheme to arbitrary k Blom's scheme can be generalized to sustain the compromise of k users by having the TA employ a polynomial of degree k.

1. Let p be a large prime publicly known and let the TA select an element $r_U \in \mathbb{Z}_p$ for each user U which also made public with $r_U \neq r_V$ for $U \neq V$.
2. for $0 \leq i, j \leq k$, the TA chooses random elements $a_{i,j} \in \mathbb{Z}_p$ such that $a_{i,j} = a_{j,i}$ and forms the polynomial:
$$f(x, y) = \sum_{i=0}^{k} \sum_{j=0}^{k} a_{i,j} x^i y^j \mod p. \tag{11}$$
The polynomial f is symmetric, i.e., $f(x, y) = f(y, x)$.
3. For each user U, the TA computes
$$g_U(x) = f(x, r_U) \mod p = \sum_{i=0}^{k} a_{U,i} x^i, \tag{12}$$
and sends the coefficient vector $a_U = \{a_{U,0}, \ldots, a_{U,k}\}$ to U, over a secure channel.
4. If U, V want to communicate, they individually compute the common key
$$K_{U,V} = K_{V,U} = f(r_U, r_V), \tag{13}$$
with user U computing
$$K_{U,V} = g_U(r_V) \tag{14}$$
and V computing
$$K_{U,V} = g_V(r_U), \tag{15}$$

Theorem 3. *The generalized Blom's KPS is unconditionally secure against the compromise of any set of k users, while a coalition of $k + 1$ can derive any key desired.*

The goal of the coalition of $k + 1$ users will be able to compute the key of any two nodes, if it can reconstruct the polynomial $f(x, y)$ from which all $g_U(x)$ are derived. The ability of the coalition to succeed is reflected in the following polynomial interpolation theorem.

Theorem 4. Lagrange Interpolation formula–*Let p be a prime and let $x_1, x_2, \ldots, x_{m+1}$ be distinct elements in $\mathbb{Z}_p$ and let $a_1, a_2, \ldots, a_{m+1}$ also be elements in $\mathbb{Z}_p$ not necessarily distinct. There is a unique polynomial $A(x) \in \mathbb{Z}_p[x]$ having a degree of at most m such that $A(x_i) = a_i$. This polynomial is given by*

$$A(x) = \sum_{j=1}^{m+1} a_j \prod_{1 \le h \le m+1,\ h \ne j} \frac{x - x_h}{x_j - x_h}. \tag{16}$$

This formula also has a bivariate form

Theorem 5. Bivariate Lagrange Interpolation formula–*Let p be a prime and let $x_1, x_2, \ldots, x_{m+1}$ be distinct elements in $\mathbb{Z}_p$ and let $a_1(x), a_2(x), \ldots, a_{m+1}(x)$ be polynomials in $\mathbb{Z}_p[x]$ of degree at most m. There is a unique polynomial $A(x, y) \in \mathbb{Z}_p[x, y]$ having a degree of at most m in both x and y such that $A(x, y_i) = a_i(x)$. This polynomial is given by*

$$A(x, y) = \sum_{j=1}^{m+1} a_j(x) \prod_{1 \le h \le m+1,\ h \ne j} \frac{y - y_h}{y_j - y_h}. \tag{17}$$

We now show how a coalition of k users can use the Bivariate Lagrange Interpolation formula to break Blom's scheme via an example.

Example: Let $p = 13, m = 2$, and pick $y_1 = 1$, $y_2 = 2$ and $y_3 = 3$

$$\begin{aligned} a_1(x) &= 1 + x = x^2, \\ a_2(x) &= 7 + 4x^2 \\ a_3(x) &= 2 + 9x. \end{aligned}$$

$$\begin{aligned} \frac{(y-2)(y-3)}{(1-2)(1-3)} &= 7y^2 + 4y + 3, \\ \frac{(y-1)(y-3)}{(2-1)(2-3)} &= 12y^2 + 4y + 10, \\ \frac{(y-1)(y-2)}{(3-1)(3-2)} &= 7y^2 + 5y + 1, \end{aligned}$$

The bivariate polynomial becomes:

$$\begin{aligned} A(x, y) &= a_1(x)(7y^2 + 4y + 3) + a_2(x)(12y^2 = 4y + 10) + a_3(x)(7y^2 + 5y + 1) \mod 13 \\ &= y^2 + 3y + 10 + 5xy^2 + 10xy + 12x + 3x^2y^2 + 7x^2y + 4x^2. \end{aligned}$$

One can verify that $A(x, y_i) = a_i(x)$.

It is fairly straightforward to see that a coalition of $k + 1$ will know $k + 1$ polynomials of degree k. Hence, using the bivariate Lagrange interpolation formula they can reconstruct $f(x, y)$ and derive any key they want. We also need to show that when the users in the coalition are less or equal to k then Blom's scheme is secure. This can be shown as follows:

Let K be the key corresponding to the user whose key needs to be computed, and let K^* be the key conjectured by the coalition. Then there is a symmetric polynomial $f^*(x, y)$ defined as follows.

$$f^*(x,y) = f(x,y) + (K^* - K) \prod_{1 \le i \le k} \frac{(x - r_{W_i})(y - r_{W_i})}{(r_U - r_{W_i})(r_V - r_{W_i})}. \tag{18}$$

It is easy to see that f^* is a symmetric polynomial. Also f^* is consistent with all the information known to the coalition cause the product contains a term $(y - r_{W_i})$. Hence it follows that regardless of the value of K^*

$$f^*(x, r_{W_i}) = f(x, r_{W_i}) = g_{W_i}(x). \tag{19}$$

Finally for computing $K_{U,V}$

$$f^*(r_U, r_V) = f(r_U, r_V) + K^* - K = K^* \tag{20}$$

So any value of K^* would actually be consistent with the information that the coalition holds, and the coalition cannot determine any other key in the network.

2 Session Key Distribution

In a session key distribution scheme, two users Alice and Bob are able to establish a shared secret K with the aid of the TA. For the purpose of establishing K the TA needs to generate and distribute K, and hence, needs to be online.

2.1 The Needham-Schroeder Scheme

The NS scheme was proposed in 1978 and it involves the following steps:

1. Alice selects r_A and sends $ID_A||ID_B||r_A$ to the TA.
2. The TA generates K and computes

$$\begin{aligned} t_{Bob} &= e_{K_B}(K||ID_A) \\ y_1 &= e_{K_A}(r_A||ID_B||K||t_{Bob}) \end{aligned}$$

   and sends y_1 to Alice.
3. Alice obtains K, t_{Bob}, and sends t_{Bob} to Bob.
4. Bob obtains K from t_{Bob}, chooses r_B and computes $y_2 = e_K(r_B)$ and sends it to Alice.
5. Alice replies with $y_3 = e_K(r_B - 1)$.

The protocol flows are illustrated in figure 1. In addition to verifying the reception of the correct messages, additional *validity checks* are performed on the decrypted messages as follows:

- Alice makes sure that the decryption of y_1 has the right form.
- Also Bob makes sure that the decryption of y_3 yields $r_B - 1$.

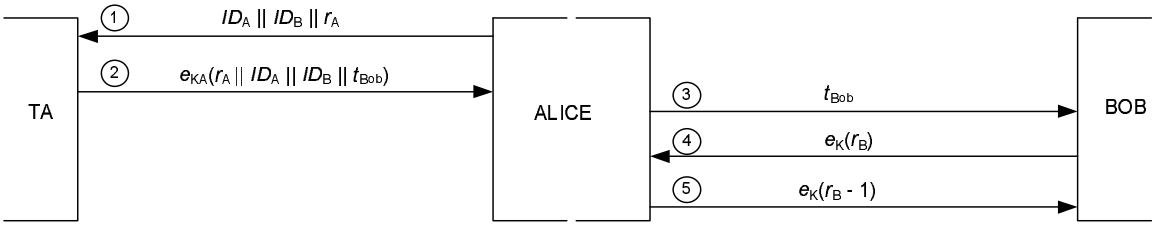

Fig. 1. The NS SKDS.

2.2 Attacks on the NS scheme

Denning and Sacco describe a known session key attack against the NS scheme that is as follows. Assume that Eve somehow obtained key K that was used during a session S between Alice and Bob. Assume now that Eve initiates a new session S' with Bob. Since Bob is not aware of the messages exchanged between Alice and the TA, eve can start from the third flow of the protocol and send t_{Bob} from session S to Bob. Bob will reply with $r_{B'}$ encrypted with K. Since Eve knows the key K from the previous session, it will be able to decrypt y_2 and reply with $r_{B'} - 1$ encrypted with K.

2.3 Kerberos Session Key Distribution Scheme

Kerberos refers to a series of SKDS developed at MIT during late 1980s. It is designed to provide strong authentication for client/server applications by using secret-key cryptography.

1. Alice selects r_A and sends $ID_A||ID_B||r_A$ to the TA.
2. The TA generates K chooses a lifetime L and computes Bob's ticket and y_1

$$t_{Bob} = e_{K_B}(K||ID_A||L)$$
$$y_1 = e_{K_A}(r_A||ID_B||K||L)$$

   The TA sends y_1, t_{Bob} to Alice.
3. Alice obtains K, and computes $y_2 = e_K(ID_A||time)$, and sends t_{Bob}, y_2 to Bob.
4. Bob obtains K from t_{Bob}, and $time$ from y_2, and replies to Alice with $y_3 = e_K(time + 1)$.

As in the case of the NS scheme several validity checks are being performed during the execution of the Kerberos protocol.

1. Alice checks the correct format of the decrypted y_1.
2. Bob checks the correct order of y_2 when decrypted, and also checks that $time < L$.
3. Alice checks that the decrypted y_3 is $time + 1$.

2.4 Comparison of NS scheme with Kerberos Scheme

Though the two schemes have similar structure the Kerberos scheme has certain advantages compared to the NS scheme. First although t_{Bob} is intended to reveal K to Bob, it is doubly encrypted with Alice's secret key. This double encryption adds unnecessary complexity without offering any

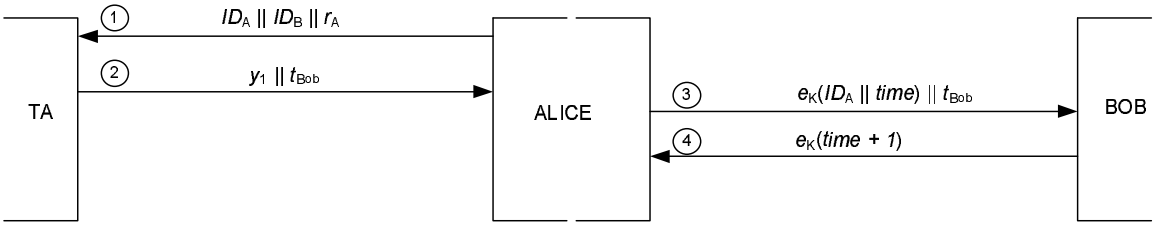

Fig. 2. The Kerberos V5 SKDS.

extra security and, hence, is removed in the Kerberos scheme. Furthermore, the use of the expiration time L limits the time that the scheme is vulnerable to the DS attack.

On the other hand, such a solution requires the use of synchronized clocks among the network participants. Security proofs involving timestamps are in general hard to be constructed and use of random nonces is preferred. Confirmation of the key by both parties (flows that use K as the encrypting key) is not necessarily an important attribute of SKDS and often is omitted. Most importantly in both schemes, confidentiality and authenticity is both provided via encryption. However, it is preferred to provide confidentiality via encryption and authenticity via a MAC. For example this can be done by having the TA send the following message to Alice on the second protocol flow.

$$y_1 = (e_{K_B}(K), MAC_{K_B}(ID_A||e_{K_B}(K))),$$
$$y_1' = (e_{K_A}(K), MAC_{K_A}(ID_A||r_A||e_{K_A}(K))),$$

Even this modification does not prevent the DS attack. To prevent this attack, Bob needs to be involved in the request from Alice and prior to the reception of the session key K.

2.5 The Bellare-Rogaway Scheme

In the BR scheme, both Alice and Bob interact with the TA. The steps executed are as follows.

1. Alice selects r_A and sends $ID_A||ID_B||r_A$ to the TA and r_A to Bob.
2. Bob selects r_B and sends $ID_A||ID_B||r_A||r_B$ to the TA.
3. The TA generates K and computes

$$y_B = (e_{K_B}(K), MAC_{K_B}(ID_A||ID_B||r_B||e_{K_B}(K)))$$
$$y_A = (e_{K_A}(K), MAC_{K_A}(ID_A||ID_B||r_A||e_{K_A}(K)))$$

The TA sends y_A to Alice, and y_B to Bob.

2.6 Formal Definition of Security for a SKDS

Definition 1. *A SKDS is assumed to be secure if the probability that a participant knows the session key K is very small, given that another participant has already accepted.*

Assumptions

First of all we make the following assumptions

1. Alice, Bob and the TA are honest
2. The encryption scheme and the MAC are secure.
3. The secret keys K_A, K_B are only known to the intended participants.
4. The nonces used in the scheme and the generation of the key K are assumed to be perfectly random.

We also define an adversarial model for Eve. We consider the following possibilities.

Adversarial Model

1. Eve is a passive eavesdropper
2. Eve is active and may try to impersonate Bob, or the TA.
3. Eve is active and may try to impersonate Eve or the TA.

With respect to the adversarial model we can prove the security of the BR scheme as follows.

1. In the case of an eavesdropper, Eve can obtain y_A, y_B but since the encryption scheme is assumed to be secure, Eve cannot obtain K. Also both Alice and Bob will accept since Eve is just a passive listener.
2. Let's now assume that Eve is impersonating Bob. When Alice receives y_A it can verify that this message was generated by the TA by validating the MAC, since only the TA is assumed to know the key K_A, the MAC incorporates the random nonce r_A, and the MAC is assumed to be secure. Furthermore, the inclusion of the encrypted K in the MAC prevents the alteration of the $e_{K_A}(K)$ in y_A. Hence, Alice can be sure that only Bob can obtain K.
3. Similar analysis as in the previous case can be done for the case where Eve impersonates Alice to Bob.