Lecture 6 (10/20/97) part 1

Dolev-Yao

Presenter: Jeanette Wing
Scribe: Ken Tew

Overview

Approach

Uses an algebraic approach. Defines the protocol in terms of strings and operations on strings. The strings represent the operations of the principals and intruder(s).

Protocols

- Ping-Pong: 2 parties.
  1st party will send N messages, 2nd party will send either N or N-1 messages. Each party will send a message then receive a response, then send again.
  - Cascade
    Cascade protocols simply allows users to apply encryption- decryption operations to form messages.
  - Name-stamp
    Name-stamp protocols allow users to append, delete, and check names encrypted together with the plaintext.
- Public-Key cryptography
  The Dolev-Yao protocol assumes all encryption is done with public/private keys. The following properties are assumed to hold:
  E_X is the encryption function for user X. D_X is the decryption function for user X.
  There is a trusted public directory that contains all the (X, E_X) pairs and is accessable by anyone. D_X is known only to X.
  $E_X D_X \ = \ D_X E_X \ = \ 1$
  Knowing $E_X(M)$ and the public directory does not reveal anything about the value of M.

Property

"Secure" means secrecy. A protocol is secure if the encrypted messages can only be seen by the principal with the decryption key.

Example 1 (Cascade)

a. A $\to$ B: (A, $E_B(M)$, B)

b. B $\to$ A: (B, $E_A(M)$, A)

Breaking Ex. 1.

1. A $\to$ Z(B): (A, $E_B(M)$, B)
2. Z $\to$ B: (Z, $E_B(M)$, B)
3. B $\to$ Z: (B, $E_Z(M)$, Z)
4. Z decodes $E_Z(M)$ to get M.

Since Z can get M this protocol is insecure.

Example 2 (Needham-Schroeder variation, Name-stamp)

a. A $\to$ B: (A, $E_B(MA)$, B)

b. A $\to$ B: (B, $E_A(MB)$, A)

According to the Dolev-Yao paper this protocol is secure. The proof is given in the paper.

Example 3 (Name stamp)

a. A $\to$ B: (A, $E_B(E_B(M)A)$, B)

b. B $\to$ A: (B, $E_A(E_A(M)B)$, A)

The intention here was to make the previous encryption harder to break by using double encryption. However, it actually makes the protocol insecure.

Breaking Example 3

Note. Z takes $M' = E_A(M)B$ so $E_A(M') = E_A(E_A(M)B)$ which Z can see.

1. B $\to$ Z(A): (B, $E_A(E_A(M)B)$, A) Z gets $E_A(M')$
2. Z $\to$ A: (Z, $E_A(E_A(M')Z)$, A)
3. A $\to$ Z: (A, $E_Z(E_Z(M')Z)$, Z) Z can now decrypt twice to get $E_A(M)B$
4. Z $\to$ A: (Z, $E_A(E_A(M)Z)$, A)
5. A $\to$ Z: (A, $E_Z(E_Z(M)Z)$, Z)

Z can now get M.

Cascade Protocols (Modeled by Dolev-Yao)

The protocol is represented by a string of E_X, E_Y, D_X, D_Y (which are applied to a message M). At each step of the protocol, the string is prepended with the appropriate E_X, E_Y, D_X, D_Y based on the operations that the principal performs on the encrypted message.

The 2 parties (X and Y) can add to the string as follows:

X can apply E_X, E_Y, and D_X
Y can apply E_X, E_Y, and D_Y

Let γ represent a string, then $\bar{\gamma}$ represents the reduced string after applying the algebraic laws: $E_X D_X = D_X E_X = 1$.

Example 1 revisited and its string notation.

a. A $\rightarrow$ B: (A, $E_B(M)$, B) E_B

b. B $\rightarrow$ A: (B, $E_A(M)$, A) $E_A D_B E_B$ which reduces to E_A

Notation N_i(X,Y) represents the string that is applied to M at step i of the protocol. So for the above example:

N_1(A,B) = E_B
N_2(A,B) = $E_A D_B E_B$

Definition of Secure (Cascade Protocol)

$\Sigma(Z) = E \bigcup \{D_Z\}$
The strings Z can generate consist of all possible encryptions and decryptions with D_Z.

Σ_X = All the strings that X will apply during the protocol

Σ_Y = All the strings that Y will apply during the protocol

A protocol is insecure if there exists some string γ in $(\Sigma(Z) \bigcup \Sigma_X \bigcup \Sigma_Y)^*$ such that $\overline{\gamma N_i(X,Y)} = \lambda$ for some N_i(X,Y).

Essentially, this says that if at any point of the protocol Z can take the message encrypted at that point and generate (or cause to be generated by running thru the protocol any number of times) a string of operations (γ) such that all of the encryptions/decryptions cancel (produce a null string λ), then Z will be able to see the plaintext message M.

Otherwise the protocol is secure.

Example 1 break. again.

1. A $\rightarrow$ Z(B): (A, $E_B(M)$, B)   Z gets N_1(A,B)
2. Z $\rightarrow$ B: (Z, $E_B(M)$, B)
3. B $\rightarrow$ Z: (B, $E_Z(M)$, Z)   Z gets B to apply $E_Z D_B$ to N_1
4. Z decodes $E_Z(M)$ to get M   Z applies D_Z to $E_Z D_B N_1$

Result: Z has generated the string $D_Z E_Z D_B$ and applied it to N_1 (E_B) which gives $D_Z E_Z D_B E_B \ = \ \lambda$

Name-Stamp Protocol

Notation
$\gamma \ = \ head(\gamma)tail(\gamma)$, where tail is a suffix of m bits.

Essentially, $tail(\gamma)$ represents the name part of the message.

A user Y can do any of the following operations on a string γ:

a. E_X encrypt with any public key

b. D_Y decrypt only with their own private key

c. i_X append a name to a message: $i_X \ \gamma \ = \ \gamma X$

d. d delete a name from a message: $d\gamma \ = \ head(\gamma)$

e. d_X name-match - verify that the name is correct and remove it:
$d_X\gamma \ = \ head(\gamma)$ if $tail(\gamma) \ = \ X$; otherwise $\perp$ Note on a name match if the name doesn't match then the protocol will halt.

Algebraic Laws:

$$E_X D_X \ = \ D_X E_X \ = \ 1$$

$$d_X i_X \ = \ d i_X = 1 \quad \text{but } i_X d_X \ \neq \ 1$$

Note for normal users X and Y a protocol must insure that no d_X or d_Y remains after reducing.

Definition of a Secure Name-Stamp Protocol

$\Sigma(Z) \ = \ (E_X,\ i_X,\ d_x,\ d \mid all\ X) \bigcup \{D_Z\}$

Σ_X = all strings applied by X (including new Name Stamp operations defined above)

Σ_Y = all strings applied by Y (including new Name Stamp operations defined above)

A protocol is insecure if there exists some string γ in $(\Sigma(Z) \ \bigcup \ \Sigma_X \ \bigcup \ \Sigma_Y)^*$ such that $\overline{\gamma N_i(X,Y)} \ = \ \lambda$ for some N_i(X,Y).

Note this is the same definition for the Cascade Protocol, only the names have changed.

A protocol is secure otherwise.

Example 2 again

a. A $\rightarrow$ B: (A, $E_B(MA)$, B) A applies $E_B i_A$
N_1(A,B) = $E_B i_A$

b. A $\rightarrow$ B: (B, $E_A(MB)$, A) B applies $E_A i_B d_A D_B$
N_2(A,B) = $E_A i_B d_A D_B E_B i_A$ reduced $E_A i_B$

Example 3 (normal)

a. A $\rightarrow$ B: (A, $E_B(E_B(M)A)$, B) A applies $E_B i_A E_B$
N_1(A,B) = $E_B i_A E_B$

b. B $\rightarrow$ A: (B, $E_A(E_A(M)B)$, A) B applies $E_A i_B E_A D_B d_A D_B$
N_2(A,B) = $E_A i_B E_A D_B d_A D_B E_B i_A E_B$ reduced $E_A i_B E_A$

Breaking Example 3

1. B $\rightarrow$ Z(A): (B, $E_A(E_A(M)B)$, A) Z starts with N_2(A,B) = $E_A i_B E_A$

2. Z $\rightarrow$ A: (Z, $E_A(E_A(M')Z)$, A) Z applies $E_A i_Z$

   $\gamma\overline{N_2(A,B)} \ = \ E_A i_Z E_A i_B E_A$ (Note Z isn't required to apply the protocol correctly).

3. A $\rightarrow$ Z: (A, $E_Z(E_Z(M')Z)$, Z) A applies $E_Z i_Z E_Z D_A d_Z D_A$

   $\gamma\overline{N_2(A,B)} \ = \ E_Z i_Z E_Z D_A d_Z D_A E_A i_Z E_A i_B E_A$

4. Z $\rightarrow$ A: (Z, $E_A(E_A(M)Z)$, A) Z applies $E_A i_Z d_B D_Z d_Z D_Z$

$$\gamma\overline{N_2(A,B)} \ = \ E_A i_Z d_B D_Z d_Z D_Z E_Z i_Z E_Z D_A d_Z D_A E_A i_Z E_A i_B E_A$$

5. A $\rightarrow$ Z: (A, $E_Z(E_Z(M)Z)$, Z) A applies $E_Z i_Z E_Z D_A d_Z D_A$

$$\gamma\overline{N_2(A,B)} \ = \ E_Z i_Z E_Z D_A d_Z D_A E_A i_Z d_B D_Z d_Z D_Z E_Z i_Z E_Z D_A d_Z D_A E_A i_Z E_A i_B E_A$$

6. Z applies $D_Z d_Z D_Z$

$$\gamma\overline{N_2(A,B)} = D_Z d_Z \underline{D_Z E_Z} i_Z E_Z D_A d_Z \underline{D_A E_A} i_Z d_B D_Z d_Z \underline{D_Z E_Z} i_Z E_Z D_A d_Z \underline{D_A E_A} i_Z E_A i_B E_A$$

Reducing all the $D_X E_X$ and $E_X D_X$ pairs gives:

$D_Z \underline{d_Z i_Z} E_Z D_A \underline{d_Z i_Z} d_B D_Z \underline{d_Z i_Z} E_Z D_A \underline{d_Z i_Z} E_A i_B E_A$

Reducing all the $d_X i_X$ pairs gives:

$\underline{D_Z E_Z} D_A d_B \underline{D_Z E_Z D_A E_A} i_B E_A$

Reducing all the $D_X E_X$ and $E_X D_X$ pairs gives:

$D_A d_B i_B E_A$

Reducing the above gives:

$\overline{\gamma \ N_2(A,B)} \ = \ \lambda$

Summary of Dolev-Yao Results.

Cascade protocols are secure iff:

The messages transmitted between X and Y always contain some layers of encryption function E_X and E_Y;

In generating a reply message, each participant A never applies D_A without also apply E_A.

Name-stamp protocols

There is a polynomial-time ($O(n^8)$) algorithm for checking whether it is secure.