Lecture 11: Authentication Protocols (cont'd)

CS 181S

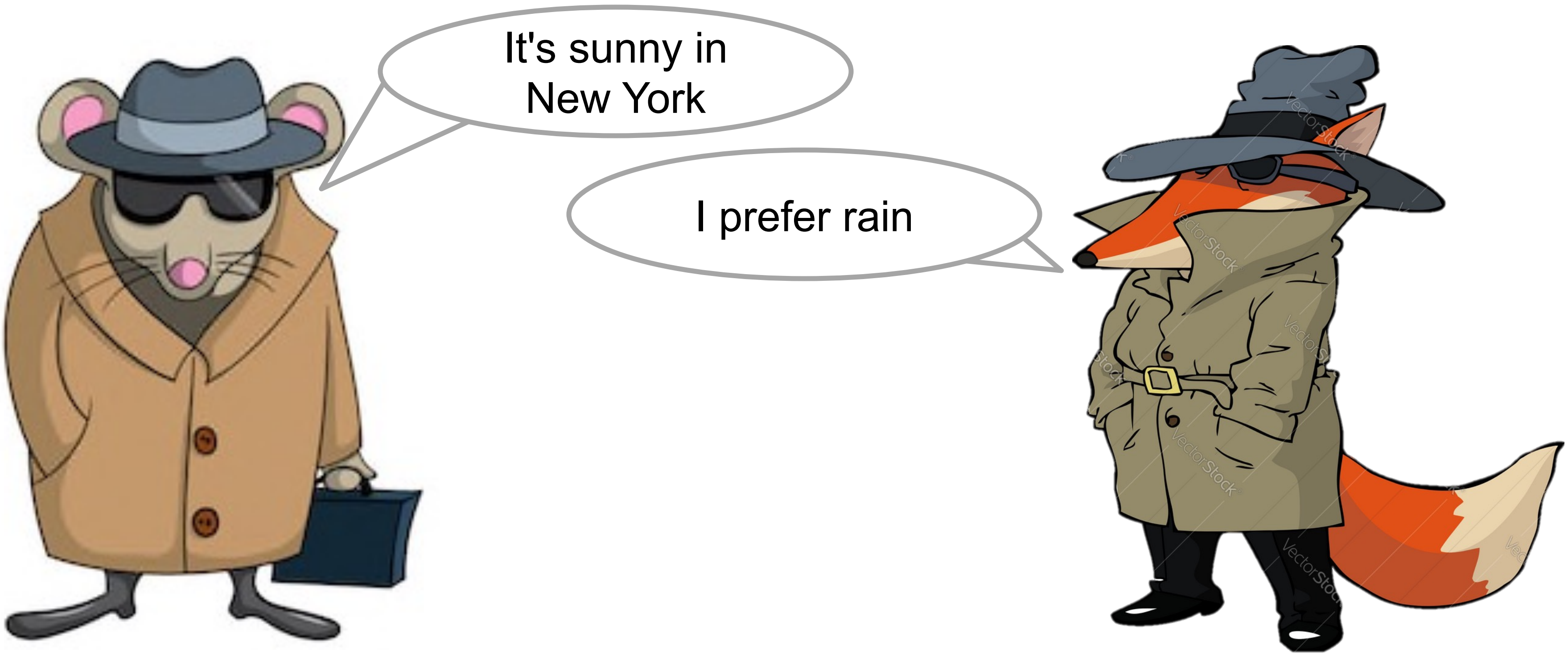
Spring 2024

Review: Authentication

- **Threat:** attacker who controls the network
  - Dolev-Yao model: attacker can read, modify, delete messages
- **Vulnerability:** communication channel between sender and receiver can be controlled by other principals
- **Harm:** attacker can pretend to be someone else (violating security goals)
- **Countermeasure:** authentication protocols

Review: Authentication Protocols

- An **authentication protocol** allows a principal receiving a message to verify the identity of the principal that sent that message



Secure Authentication Protocols

Multiple Keys

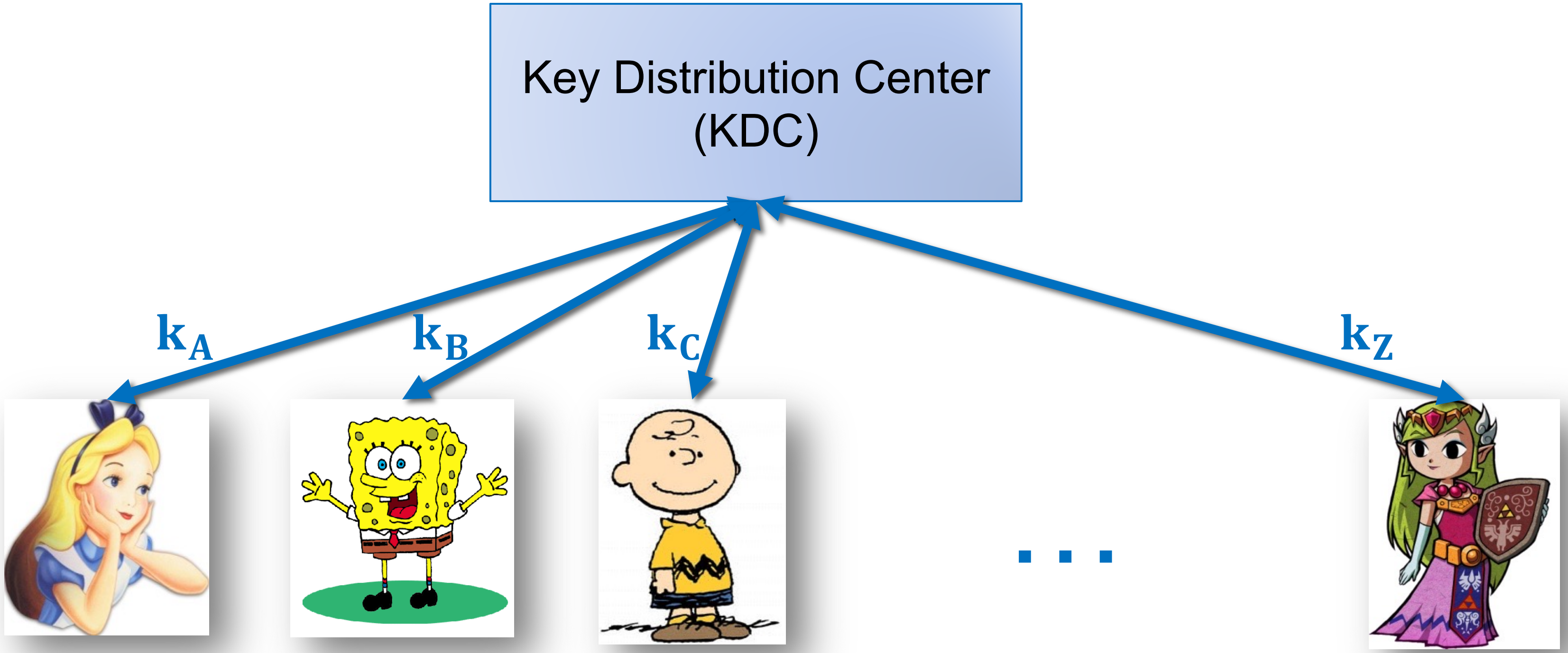
```
1. B->T: B, r
   1) T->B: A, r
   2) B->T: B, Enc(r; k_BA)
2. T->B: A, Enc(r; k_BA)
```

Included Identity

```
1. B->T: B, r
   1) T->B: A, r
   2) B->T: B, Enc(B, r; k)
2. T->B: A, Enc(B, r; k)
```

Assumptions

- ~~Assume Alice and Bob have a shared secret key k~~
- Assume that symmetric-key crypto works
- Assume there is a trusted **Key Distribution Center (KDC)** and that all principals have a shared key with the KDC



Goals

- Alice and Bob should acquire a shared key that they can use to securely communicate
- Alice should be convinced that she is talking to Bob
- Bob should be convinced that he is talking to Alice

Protocol 1

1. `A -> KDC: A, B`
2. `KDC -> A: A, B, Enc(k; k_A)`
3. `KDC -> B: A, B, Enc(k; k_B)`

Protocol 2

1. `A -> KDC: A, B`
2. `KDC -> A: A, B, Enc(k; k_A), Enc(k; k_B)`
3. `A -> B: A, B, Enc(k; k_B)`

Threat Model

- Dolev-Yao attacker
  - controls the network, can read, modify, create packets

- A **replay attack** occurs when an adversary repeats fragments of a previous protocol run
- ~~A **reflection attack** occurs when an adversary sends messages from an ongoing protocol back to the originator~~
- A **man-in-the-middle attack** occurs when an adversary secretly relays (and potentially changes) communications between two principals who believe they are communicating directly with eachother

Exercise 1: Replay Attacks

Is this protocol vulnerable to a replay attack?

```
1. A -> KDC: A, B
2. KDC -> A: A, B, Enc(k; k_A), Enc(k; k_B)
3. A -> B: A, B, Enc(k; k_B)
```

Protocol 3

```
1. A -> KDC: A, B, r
2. KDC -> A: A, B, Enc(k,r;k_A), Enc(k;k_B)
3. A -> B: A, B, Enc(k; k_B)
```

Is this protocol vulnerable to a replay attack?

MITM Attack

1. `A -> T: A, B, r`
   1) `T -> KDC: A, T, r`
   2) `KDC -> T: A, T, Enc(k, r; k_A), Enc(k; k_T)`
   1) `T -> KDC: T, B, r`
   2) `KDC -> T: T, B, Enc(k2, r; k_T), Enc(k2; k_B)`
2. `T -> A: A, B, Enc(k, r; k_A), Enc(k2; k_B)`
3. `A -> B: A, B, Enc(k2; k_B)`

Protocol 5

```
1. A -> KDC: A, B, r
2. KDC -> A: A, B, Enc(k, r, Enc(k; k_B); k_A)
3. A -> B: A, B, Enc(k; k_B)
```

Attack on Protocol 5

1. `T -> KDC: T, B, r`
2. `KDC -> T: T, B, Enc(k, r, Enc(k; k_B);k_T)`
3. `T -> B: A, B, Enc(k; k_B)`

Protocol 6

1. `A -> KDC: A, B, r`
2. `KDC -> A: A, B, Enc(k,r,Enc(A,B,k; k_B);k_A)`
3. `A -> B: A, B, Enc(A,B,k; k_B)`

Attack on Protocol 6

1. `A -> T: A, B, r`
   1. `T -> KDC: A, T, r`
   2. `KDC -> T: A, T, Enc(k, r, Enc(A,T,k; k_T);k_A)`
2. `T -> A: A, B, Enc(k, r, Enc(A,T,k; k_T);k_A)`
3. `A -> T: A, B, Enc(A,T,k; k_T)`

Protocol 7

1. `A -> KDC: A, B, r`
2. `KDC -> A: Enc(A,B,k,r,Enc(A,B,k; k_B);k_A)`
3. `A -> B: A, B, Enc(A,B,k; k_B)`

Protocol 8: Needham-Schroeder

```
1. A -> KDC: A, B, r
2. KDC -> A: Enc(A,B,k,r,Enc(A,B,k; k_B);k_A)
3. A -> B: A, B, Enc(A,B,k; k_B)
4. B -> A: A, B, Enc(r2; k)
5. A -> B: A, B, Enc(r2+1; k)
```

Exercise 2: MITM Attacks

Consider the following variant of Needham-Schroeder. Is this protocol vulnerable to a MITM attack?

```
1. A -> KDC: A, B, r
2. KDC -> A: Enc(A,B,r;k_A),Enc(r,k; k_A)
3. KDC -> B: Enc(A,B,r;k_B),Enc(r,k; k_B)
4. B -> A: A, B, Enc(r2; k)
5. A -> B: A, B, Enc(r2+1; k)
```

Exercise 2: MITM Attacks

Consider the following variant of Needham-Schroeder. Is this protocol vulnerable to a MITM attack?

Protocol 8: Needham-Schroeder

1. `A -> KDC: A, B, r`
2. `KDC -> A: Enc(A,B,k,r,Enc(A,B,k; k_B);k_A)`
3. `A -> B: A, B, Enc(A,B,k; k_B)`
4. `B -> A: A, B, Enc(r2; k)`
5. `A -> B: A, B, Enc(r2+1; k)`

Solution #1: More random numbers

1. A -> B: A, B
2. B -> A: A, B, r3
3. A -> KDC: A, B, r, r3
4. KDC -> A: Enc(A,B,k,r,Enc(A,B,k,r3; k_B);k_A)
5. A -> B: A, B, Enc(A,B,k,r3; k_B)
6. B -> A: A, B, Enc(r2; k)
7. A -> B: A, B, Enc(r2+1; k)

Solution #2: Timestamps

```
1. A -> KDC: A, B, r,
2. KDC -> A: Enc(A,B,k,r,Enc(A,B,k,t; k_B);k_A)
3. A -> B: A, B, Enc(A,B,k,t; k_B)
4. B -> A: A, B, Enc(r2; k)
5. A -> B: A, B, Enc(r2+1; k)
```

Solution #3: Otway-Rees

```
1. A -> B: n, A, B, Enc(r1,n,A,B;k_A)
2. B -> KDC: n, A, B, Enc(r1,n,A,B;k_A),
                          Enc(r2,n,A,B;k_B)
3. KDC -> B: n, Enc(r1,k;k_A),
                          Enc(r2,k;k_B)
4. B -> A: n, Enc(r1,k;k_A)
```

Type Attack

```
1. A -> B: n, A, B, Enc(r1,n,A,B;k_A)
2. B -> KDC: n, A, B, Enc(r1,n,A,B;k_A),
                      Enc(r2,n,A,B;k_B)
3. T -> B: n, Enc(r1,n,A,B;k_A),
                      Enc(r2,n,A,B;k_B)
4. B -> A: n, Enc(r1,n,A,B;k_A)
```

Exercise 3: Type Attacks

Consider the following variant of Otway-Rees

```
1. A -> B: n, A, B, Enc(r1,n,A,B;k_A)
2. B -> KDC: n, A, B, Enc(r1,n,A,B;k_A),
                         Enc(r2,n,A,B;k_B)
3. KDC -> B: n, Enc(r1+1,k;k_A),
                         Enc(r2+1, k;k_B)
4. B -> A: n, Enc(r1+1,k;k_A)
```

Would this protocol be vulnerable to a type attack?

Authentication in Practice

Microsoft
Active Directory

OpenLDAP™
http://www.OpenLDAP.org