CSIT 495/595 - Introduction to Cryptography
Key Distribution and Management

Bharath K. Samanthula
Department of Computer Science
Montclair State University

Recap of Number Theory

- Prime and Composites
- Modular Arithmetic
- Groups and Fermat's theorem
- Euler's Generalization
- Intractable Problems

Key Distribution and Management

- In private-key cryptography, parties communicate using a shared secret key
- Key Question: How can the parties share a secret key in the first place?
- Naive approaches:
  1. two parties can meet and exchange the key
  2. parties use a trusted courier service as secure channel
- The above approaches were used earlier to share keys in many government, diplomatic and military contexts (e.g., the "red phone" connecting Moscow and Washington in 1960s)

How Convenient is this Secure Channel?

Example: Consider a large multinational company X

- Suppose every pair of employees at X want to securely communicate
- It is difficult for each pair of employees to meet to share the key (often, this may become impossible for employees working in different continents)
- Also, how to share keys with new employees?

Key Management Issues

- If N employees are able to share secret keys, then each employee will have to manage and store $N - 1$ keys
- There may be other keys the employee has to manage (for connecting to printers, customers, etc.)
- The more keys there are, the harder it is to protect them
- Of course, all the keys should be stored securely
- In case of fewer keys, keys can be stored on *secure hardware* such as a *smartcard*

Issues with Private-Key Cryptography

Broadly, three main issues:

- key distribution
- storing and managing large number of secret keys
- inapplicability of private-key cryptography to open systems

In open systems, users may not be aware of each others' existence until they want to communicate (e.g., sending an email to someone whom you have never met)

Key Distribution Techniques

- Key-Distribution Center (KDC)
- Diffie-Hellman Key Exchange

Key-Distribution Center (KDC) 1

- Consider the previous example where all pairs of employees must be able to communicate securely
- Assume there is a key distribution center (a trusted party such as system administrator) who can establish shared keys
- A naive approach
  1. For any new employee i, the KDC can share a key with that employee on his first day of work (at a secret location)
  2. KDC generates $i - 1$ keys, say $k_1, \ldots, k_{i-1}$, and sends k_j to j^{th} employee
- Is the above approach practical?

Key-Distribution Center (KDC) 2

A better Approach

- KDC generates keys on an on-demand basis
- Suppose KDC shares k_A and k_B with Alice and Bob, resp.
- If Alice wants to send a message to Bob, she informs the KDC
- KDC generates a session key and sends this to Alice encrypted using k_A and Bob using k_B
- After finishing the conversation, Alice and Bob can erase the session key

Key-Distribution Center (KDC) 3

Pros

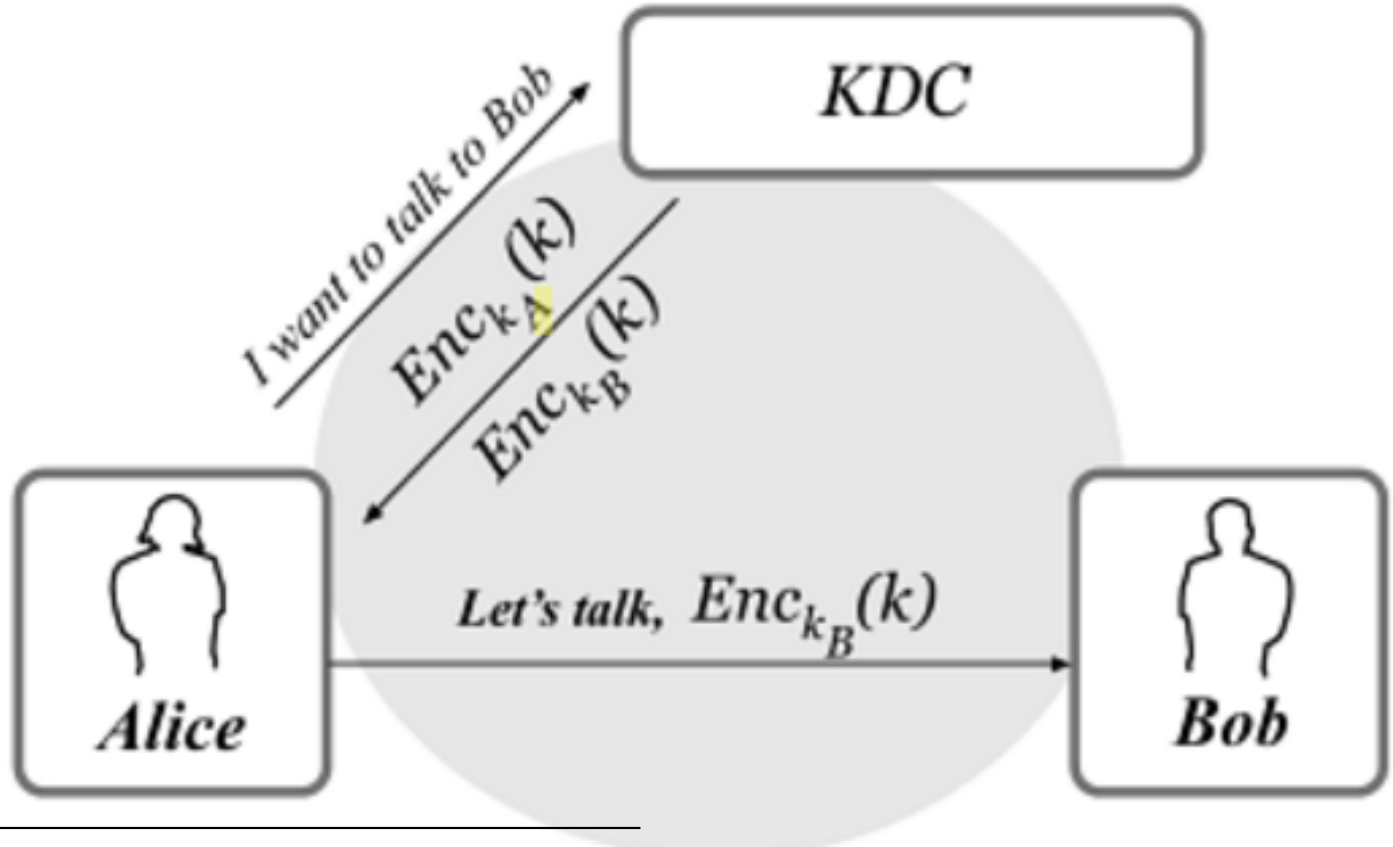
- Each employee need to store only one long-term key (the one they share with KDC)
- If a new employee joins the company, the work is minimal - a key must be set up between this employee and KDC

Cons

- A successful attack on KDC can compromise the whole system
- KDC acts as a single point of failure (if KDC is down due to over load, secure communication may be temporarily impossible)

Needham-Schroeder Protocol 1

- Forms the core of Kerbos - a widely used service[1] in many universities for authentication and supporting secure communication



[1] Kerberos is the default mechanism for supporting secure networked communication in Windows and many UNIX systems.

Needham-Schroeder Protocol 2

- The second ciphertext, $Enc_{k_B}(k)$, is sometimes called a ticket
- In this protocol, KDC need not initiate a second connection to Bob
- Also, if Bob is offline, KDC need not to worry
- Once Alice retains the ticket, she can re-initiate a secure communication with Bob without the involvement of Bob

Diffie-Hellman Protocol 1

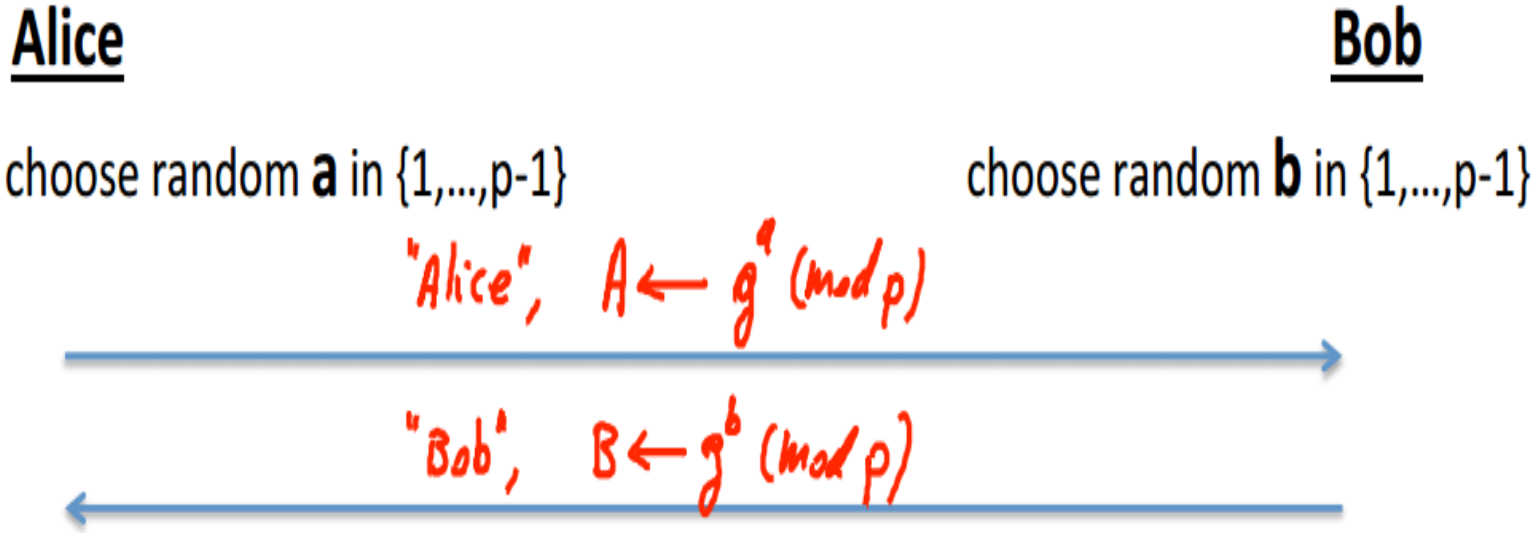
- KDCs and protocols like Kerberos still require, a private and authenticated channel at some point
- Key Question: Can we achieve private communication without ever communicating over a private channel?
- In 1976, *Whitfield Diffie* and *Martin Hellman* published a paper titled “New Directions in Cryptography”
  - Observed that certain things can be easily performed but not easily reversed
  - Example: it is easy to multiply two large prime numbers, but extremely hard to recover their primes from product (Factoring problem)

Diffie-Hellman Protocol 2

Based on the idea that parties can perform operations that they can reverse but that an eavesdropper cannot

Fix a large prime p (e.g. 600 digits)

Fix an integer g in {1, ..., p}



$$B^a \pmod p = (g^b)^a = k_{AB} = g^{ab} \pmod p = (g^a)^b = A^b \pmod p$$

Diffie-Hellman Protocol: Example

1. Alice and Bob agree on $p = 23$ and $g = 5$.
2. Alice chooses $a = 6$ and sends $5^6 \mod 23 = 8$.
3. Bob chooses $b = 15$ and sends $5^{15} \mod 23 = 19$.
4. Alice computes $19^6 \mod 23 = 2$.
5. Bob computes $8^{15} \mod 23 = 2$.

Then 2 is the shared secret.

Diffie-Hellman Protocol: Security

Eavesdropper sees: $p, g, A=g^a \pmod p$, and $B=g^b \pmod p$

Can she compute $g^{ab} \pmod p$??

The above DH problem (involve solving discrete logarithm) is considered to be hard for groups whose order is large enough

Useful References

- Chapter 10, Introduction to Modern Cryptography by Jonathan Katz and Yehuda Lindell, 2nd Edition, CRC Press, 2015.
- `https://www.math.brown.edu/~jhs/MathCrypto/SampleSections.pdf`