CIS 4360
Secure Computer Systems

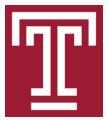
Applied Cryptography

Professor Qiang Zeng

Spring 2017

Symmetric vs. Asymmetric Cryptography

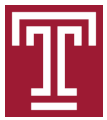
- Symmetric cipher is much faster
- With asymmetric ciphers, you can post your Public Key to the world and then the world can communicate with your secretly without having to meet you first
- Non-repudiation can only be achieved through asymmetric cryptography
  - Digital Signature
- Key establishment with Asymmetric Crypto is easier *(to be covered this lecture)*



Previous class…

How do Digital Signatures assure non-repudiation?

A Digital Signature is generated by one's private key; nobody else can generate the signature

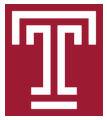


Previous class…

Since Asymmetric Cryptography is so versatile, can it replace Symmetric Cryptography completely?

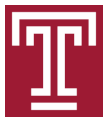
No, Symmetric Cryptography has its advantage on speed.

Plus, all Asymmetric Ciphers are established on some computationally difficult number-theory problems, which are never mathematically proven to be difficult. Advances in number theory or Quantum Computing may one day render all asymmetric ciphers ineffective. But Symmetric Cryptography will keep safe (you only need to increase the key size when the computing power advances)



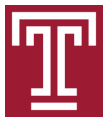
Outline

- Authentication Protocols
- Data Integrity Checking Protocols
- Forward Secrecy
  - Diffie-Hellman Key Agreement



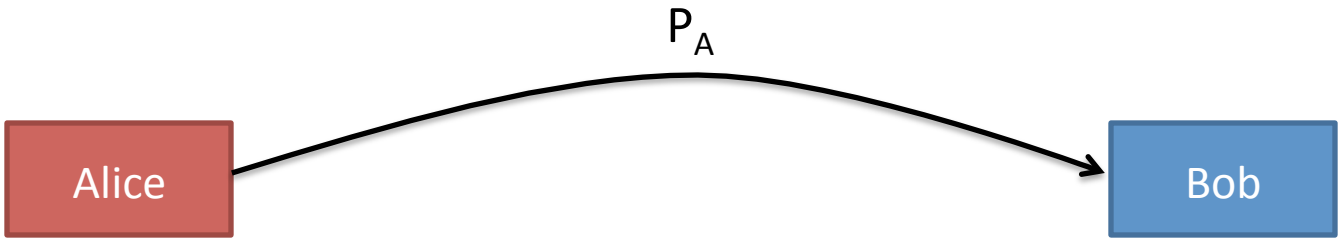
User Authentication

- User authentication: to verify the identity of the communicating participant
  - E.g., the participant shows evidence about the knowledge of some password

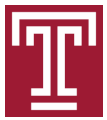


User Authentication – Password as Plaintext (Use it only over a secure channel)

- Bob wants to authenticate Alice's identity
- Assumption: Bob knows Alice's password is P_A
  - P_A is Alice's credential

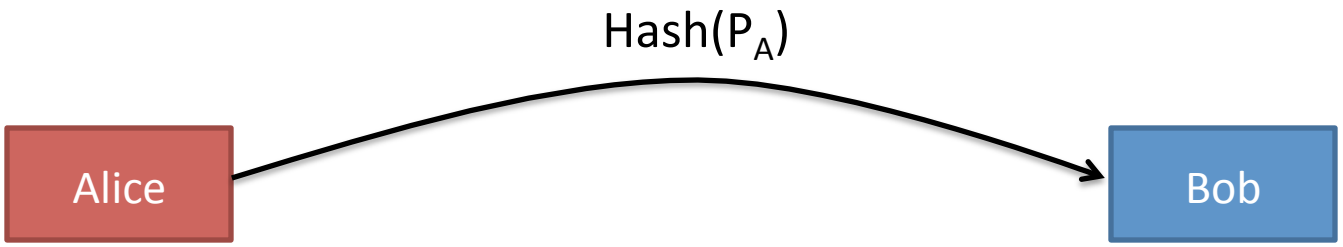


The password is transmitted as plaintext, which can be intercepted by the adversary; this scheme is insecure for network-based communication

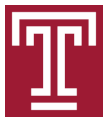


User Authentication – Hash of Password (don't use it)

- Bob wants to authenticate Alice's identity
- Assumption: Bob knows Alice's password is P_A

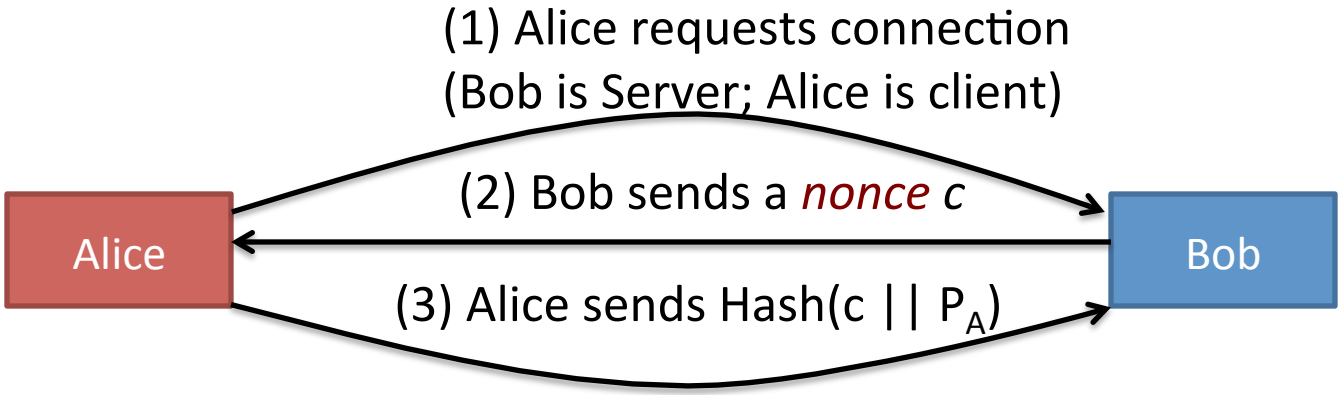


It is insecure as the adversary can record Hash(P_A) for Replay Attacks



Challenge-response based User Authentication

- Bob wants to authenticate Alice's identity
- Assumption: Bob knows Alice's password is P_A



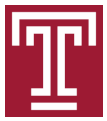
This is called the Digest Access Authentication. Replay Attacks will not work, why?

As Bob (Server) makes sure *c* is never reused.

A Nonce is a number that is only used once

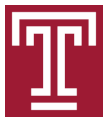
Challenge-response based User Authentication

- The scheme is still vulnerable to the Chosen Plaintext Attack
  - The adversary may intercept the request from Alice and impersonate the server
  - The fake server then sends Alice a pre-selected "challenge" c
  - Alice then returns Hash(c || password)
  - If the adversary has pre-computed Hash(c || password) for all possible passwords and the pre-selected c value, then a rainbow table attack can be launched
- Countermeasure: Alice sends a client nonce (cnonce) along with Hash(cnonce || c || password)



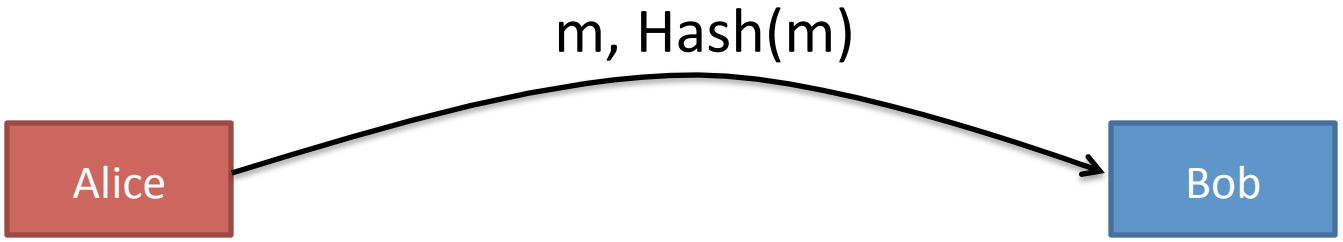
User Authentication vs. Data Integrity

- User authentication: the identity of the communicating participant can be verified
  - E.g., the participant shows evidence about the knowledge of some password
- Data integrity: the receiver can check whether the message has been manipulated
  - Data integrity implies that the data comes from the right origin

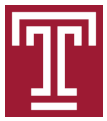


Data Integrity with Symmetric Crypto

- Alice wants to make sure Bob can verify the integrity of the message received

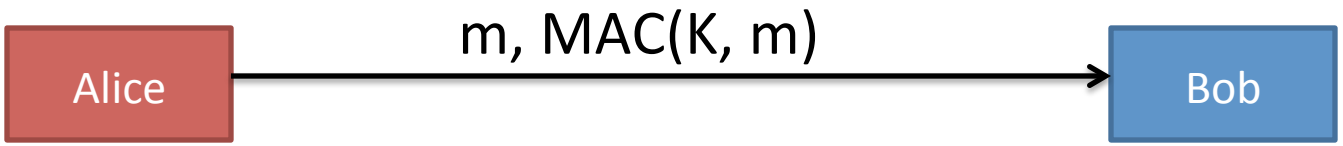


We already covered that the adversary may replace both m and the hash; thus, this scheme is insecure if the message and the hash are both transmitted through an insecure channel

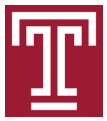


Data Integrity with Symmetric Crypto

- Alice wants to make sure Bob can verify the integrity of the message received
- K is the shared key between Alice and Bob

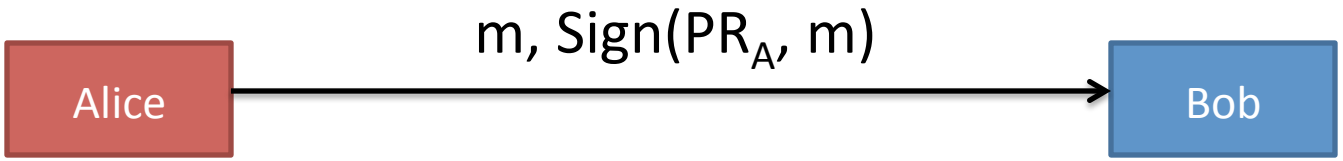


It is critical that the message should contain timestamp or sequence number; otherwise, it is vulnerable to Replay Attacks

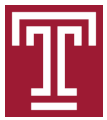


Data Integrity with Asymmetric Crypto

- Alice wants to make sure Bob can verify the integrity of the message received
- PR_A is the private key of Alice

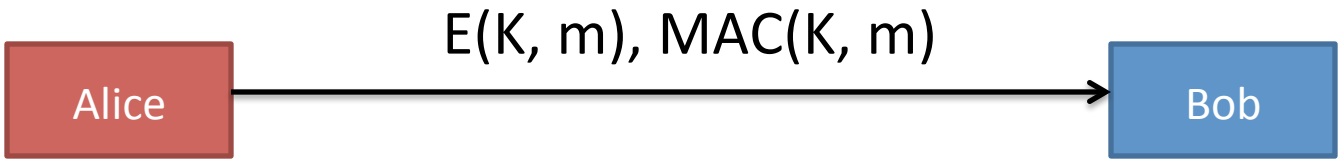


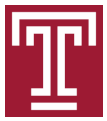
- Bob recovers the digest from the signature
- Then, Bob regenerates the digest independantely and compares it against the recovered digest
- Can this scheme achieve non-repudiation?
  - Yes, everyone can verify that the signature was generated by Alice, and only Alice has the private key to generate it



Data Integrity + Confidentiality with Symmetric Crypto

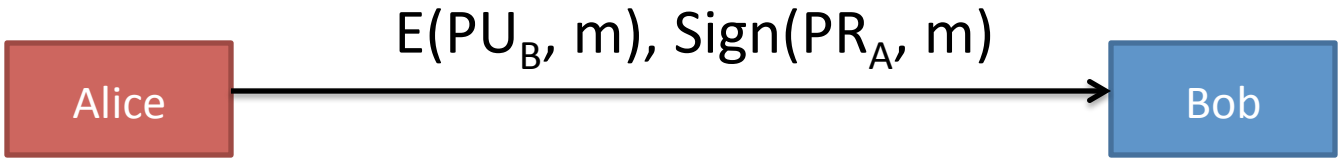
- Alice wants to make sure Bob can verify the integrity of the message received
- In addition, Alice wants to achieve confidentiality



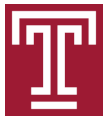


Data Integrity + Confidentiality with Asymmetric Crypto

- Alice wants to make sure Bob can verify the integrity of the message received
- In addition, Alice wants to achieve confidentiality

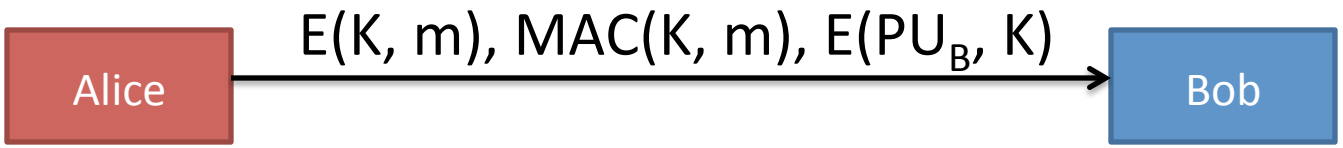


- What is the disadvantage of this scheme?
  - Asymmetric Crypto is quite expensive. It is not economic to use it to encrypt a large amount of data (Recall that when you sign a message, you do not sign the message directly but its digest, e.g., 256 bits)

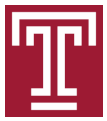


Data Integrity + Confidentiality with Asymmetric Crypto

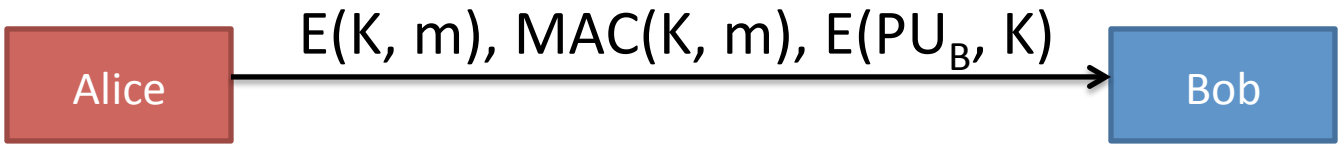
- Alice wants to make sure Bob can verify the integrity of the message received
- In addition, Alice wants to achieve confidentiality



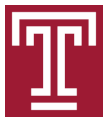
- Alice can pick a key
  - The key is used to encrypt and generate the MAC
  - The key is encrypted using Bob's public key and sent to Bob



Can We Do Better?

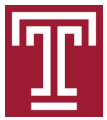


- The adversary may collect the traffic between Alice and Bob, even though the adversary does not understand the conversation
- It is possible that one day the adversary gets Bob's private key (e.g., the adversary is CIA)
- Is there countermeasure that protects the confidentiality of the past conversations even all the traffic has been collected and the long-time private key (of Bob) is leaked one day



Forward Secrecy

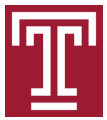
- Forward Secrecy (also called Perfect Forward Secrecy) protects past conversations against future compromises of long-time secret keys or passwords.
- It implies that even CIA has collected the traffic of all the past conversations and later obtains the key or password, you can deny the content about the conversation
- Forward Secrecy is usually built on Diffie-Hellman based key agreement



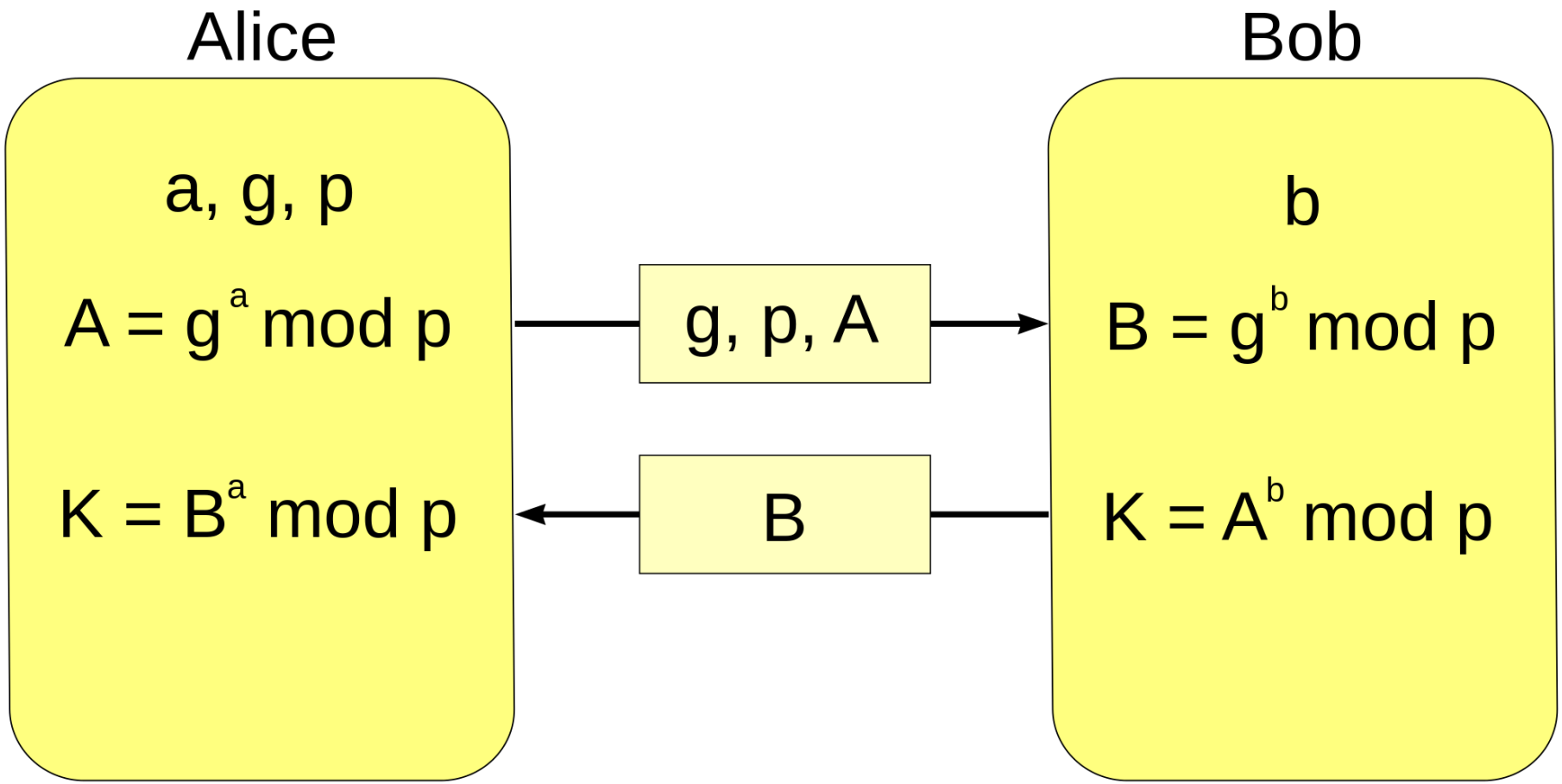
Diffie-Hellman Key Agreement

- While DH can also be used for encryption, the most wide use is to negotiate keys
- The most prominent property of DH is that even the adversary obtains all the traffic for key agreement in plaintext, the adversary cannot infer the key

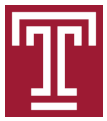
Seemingly *impossible*, but was achieved by Diffie and Hellman in 1976



DH Key Agreement



$K = A^b \bmod p = (g^a \bmod p)^b \bmod p = g^{ab} \bmod p = (g^b \bmod p)^a \bmod p = B^a \bmod p$



Agree upon two numbers:

P Prime Number 13

G Generator of P 6

Private = 5

Randomly generate a Private Key

Private = 4

(6^{5}) MOD 13

(7776) MOD 13

Public = 2

Calculate Public Key:

$(G^{Private})$ MOD P

(6^{4}) MOD 13

(1296) MOD 13

Public = 9

Exchange Public Keys

(9^{5}) MOD 13

(59049) MOD 13

Shared Secret = 3

Calculate the Shared Secret

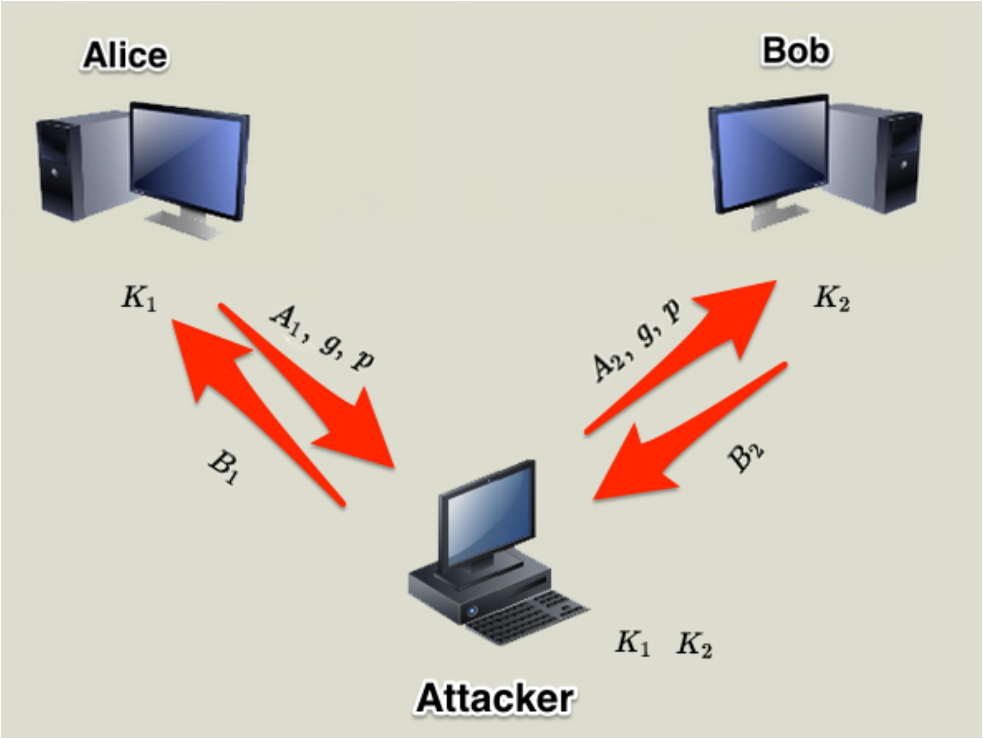
$(Shared\ Public^{Private})$ MOD P

(2^{4}) MOD 13

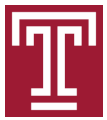
(16) MOD 13

Shared Secret = 3

Subject to Man-in-the-middle Attack

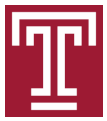


- The essential problem is that the schemes lacks authentication
  - Alice has no way to authenticate whether B is sent by Bob
  - Bob has no way to authenticate whether A is sent by Alice



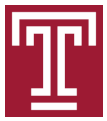
DH Key Agreement with Authentication

- Example: Station-to-Station protocol
- (1) Alice $\rightarrow$ Bob : *A, g, p*
- (2) Alice $\leftarrow$ Bob : *B*, $Cert_B$, $E_K(S_B(A, B))$ //Bob signs it
- (3) Alice $\rightarrow$ Bob : $Cert_A$, $E_K(S_A(A, B))$ // Alice signs it



Summary

- Important Applications of Crypto for
  - User Authentication
  - Data Integrity
  - Confidentiality
- Diffie-Hellman Key Agreement
  - Modular Logarithm
  - For Forward Secrecy



Writing Assignments

- How to achieve authentication and data integrity of communication over an insecure channel? What should the message sender do?
- Why is Diffie-Hellman Key Agreement subject to the man-in-the-middle Attack?