Revision

CS5285 Sem A 2024/25

Gerhard Hancke

Lecture Plan

- Basic revision/overview (very quickly!)
  - What I hope you know…in theory
  - Study the original slides not the revision lecture!
- Some course context (e-commerce)
  - You did learn useful things you can apply…in practice
- Exam Information

Revision
Lecture 1
Basic Security

Security Services and Mechanisms

- A security threat is a possible means by which your security goals may be breached (e.g. loss of integrity or confidentiality).
- A security **service** is a measure which can be put in place to address a threat (e.g. provision of confidentiality).
- A security **mechanism** is a means to provide a service (e.g. encryption, digital signature).

Data Confidentiality and Integrity

- Protection against unauthorised disclosure of information.
- Integrity is protection against unauthorised modification of data

- Think back: What is ‘protection’ in each case?
  – Prevent, Detect, Recover?

Authentication

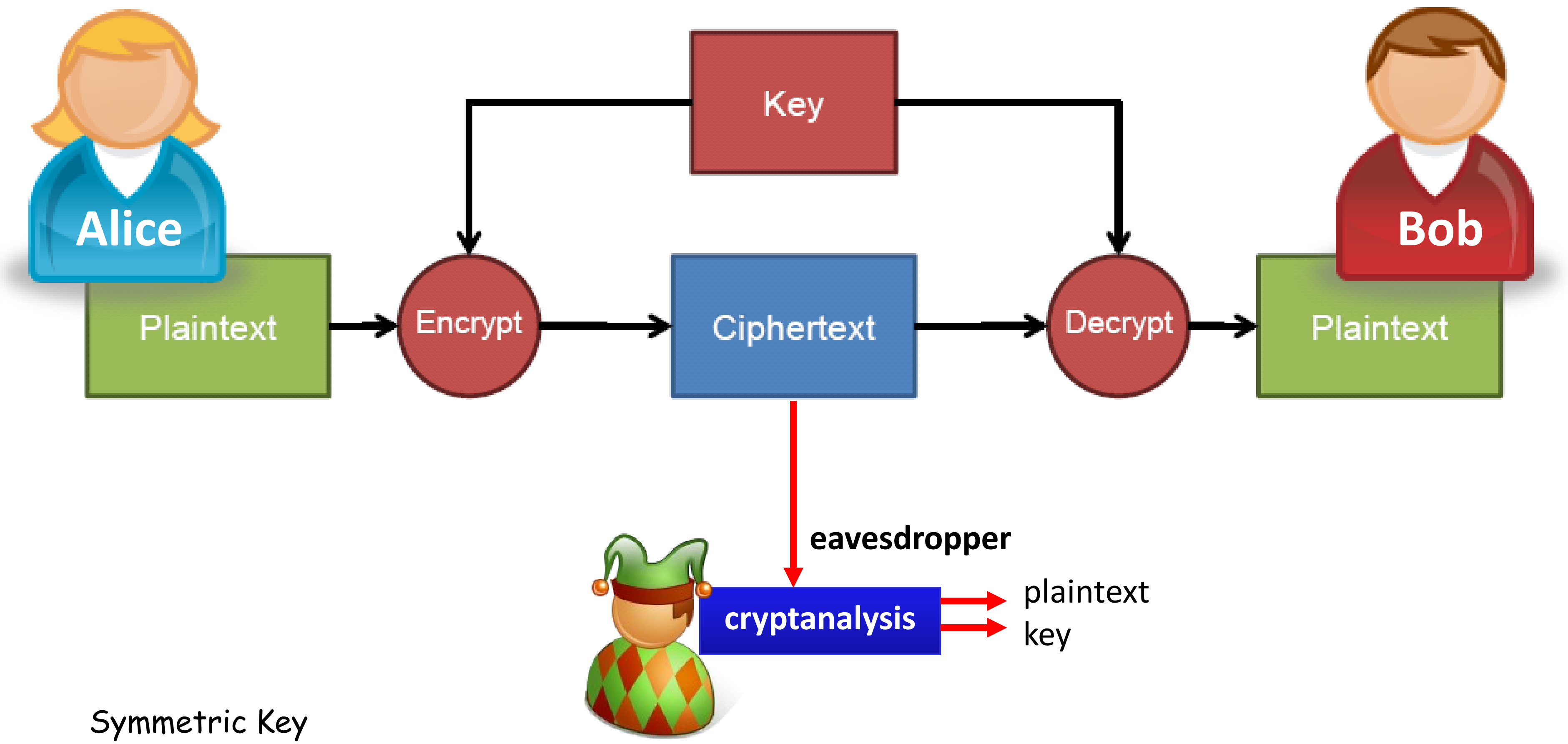
- **Entity authentication** provides checking of a claimed identity at a point in time.
  - Typically used at start of a connection.
  - Addresses masquerade and replay threats.
- **Origin authentication** provides verification of source of data.
  - Does not protect against replay or delay.

Non-repudiation

- Protects against a sender of data denying that data was sent (**non-repudiation of origin**).
- Protects against a receiver of data denying that data was received (**non-repudiation of delivery**).
- Example: analogous to signing a letter and sending via recorded delivery.

Revision
Lecture 2
Symmetric Encryption

- A symmetric-key cipher or cryptosystem is used for encrypting/decrypting a plaintext/ciphertext
- The same key is used for encrypting and decrypting



Cryptanalysis

Basic assumptions

- The system is completely known to the attacker
- Only the key is secret
- Also known as **Kerckhoffs Principle**
- Crypto algorithms are not secret
- No "security through obscurity"

Objective of an attacker

- Identify secret key used to encrypt a ciphertext
- (OR) recover the plaintext of a ciphertext without the secret key

- Simple substitution and shift ciphers!
- A secret key (in Simple Substitution) is a *random permutation* of the alphabetic characters.
- E.g.

a	*b*	*c*	*d*	*e*	*f*	*g*	*h*	*i*	*j*	*k*	*l*	*m*
X	*N*	*Y*	*A*	*H*	*P*	*O*	*G*	*Z*	*Q*	*W*	*B*	*T*

n	*o*	*p*	*q*	*r*	*s*	*t*	*u*	*v*	*w*	*x*	*y*	*z*
S	*F*	*L*	*R*	*C*	*V*	*M*	*U*	*E*	*K*	*J*	*D*	*I*

- Each permutation is a potential candidate of the secret key

Statistical Attack / Frequency Analysis

- An interesting observation on simple substitution: the relative letter frequencies do not change during encryption

- Average letter frequencies in English (Beker and Piper, 1982)

letter	frequency	letter	frequency
A	.082	N	.067
B	.015	O	.075
C	.028	P	.019
D	.043	Q	.001
E	.127	R	.060
F	.022	S	.063
G	.020	T	.091
H	.061	U	.028
I	.070	V	.010
J	.002	W	.023
K	.008	X	.001
L	.040	Y	.020
M	.024	Z	.001

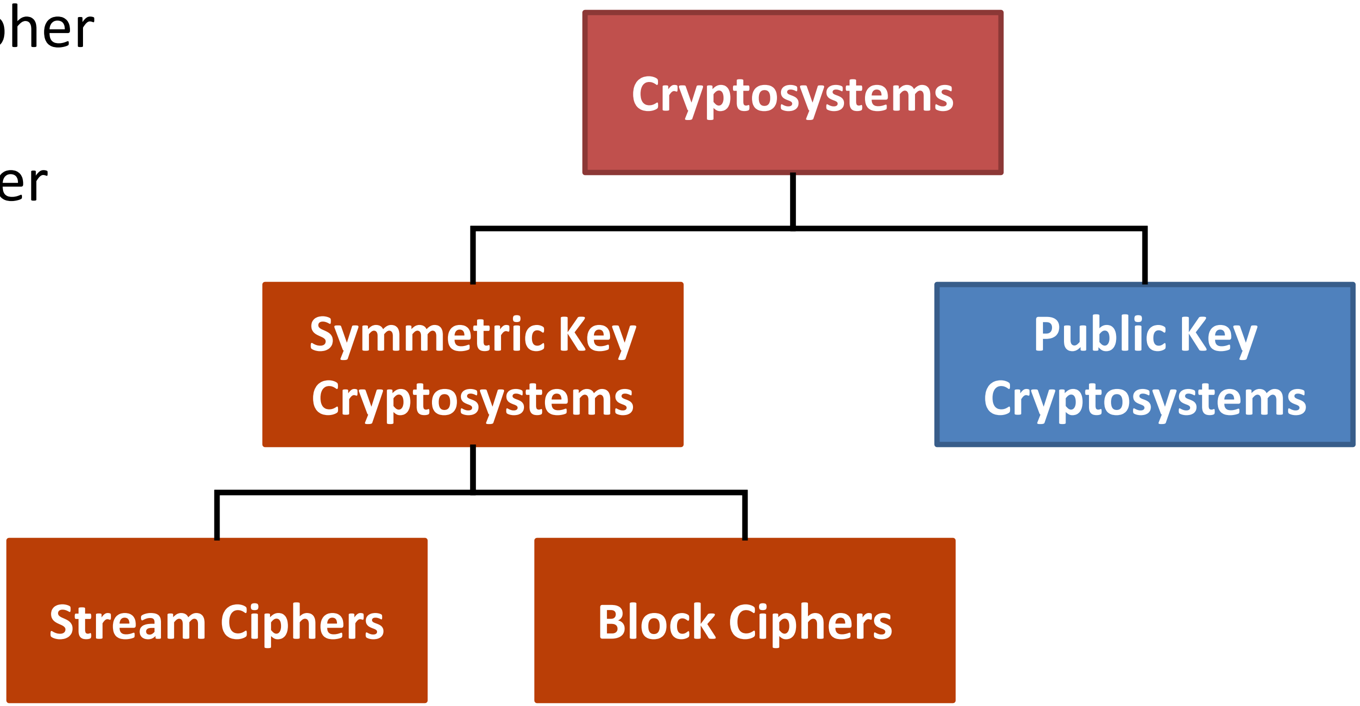
Questions: What are the current symmetric key cryptosystems?

There are many…

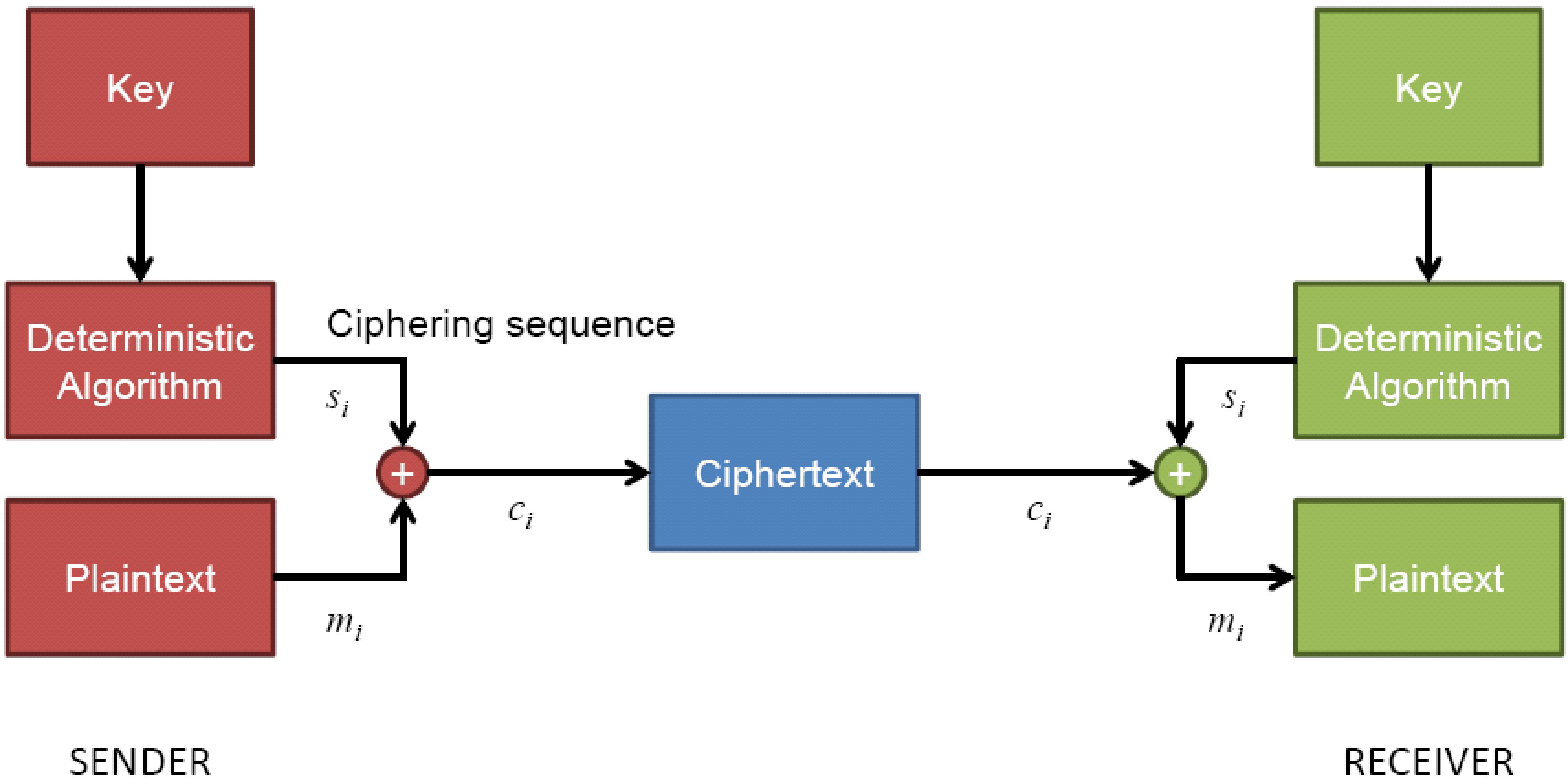
They can be categorized into two types:

1. Stream Cipher

2. Block Cipher

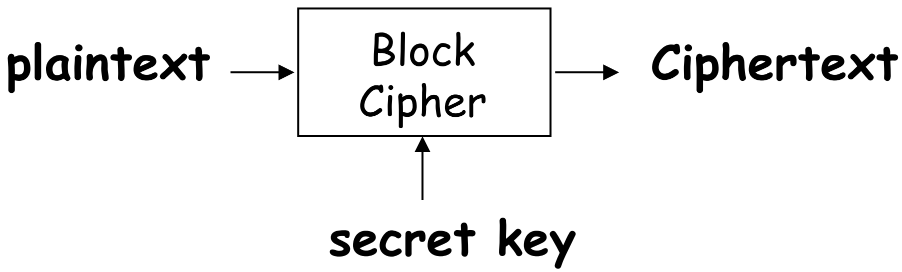


Stream Ciphers



- Deterministic Algorithm a.k.a. **Keystream Generator**
- Ciphering Sequence a.k.a. **Keystream**
- **We looked in a little more detail at RC4**

Block Ciphers

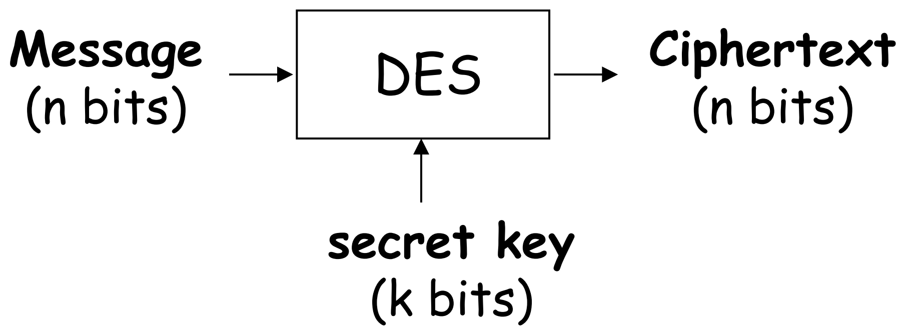


- A block cipher takes a *block* of **plaintext** and a **secret key**, produces a *block* of **ciphertext**.
- The key is **reused** for different plaintext blocks
- Typical block sizes: 64 bits, 128 bits, 192 bits, 256 bits
- Key sizes: 56 bits (DES), 128/192/256 bits (AES)
- Popular block ciphers: DES, 3DES, AES

Bruteforce Attack | Exhaustive Key Search

- **An algorithm is secure when the easiest way of attacking it is by bruteforce attack.**
  - i.e. check all possible key combinations one by one (could be done in parallel)
  - For a key of *n* bits, the total number of possible keys (or the entire key space) is 2^n.
  - An average of half the combinations should be tried in order to find the key, i.e. 2^{n-1}.
- Nowadays the minimum key size is 80 bits to make it impossible for a bruteforce attack.
- To give a better security margin, the key size is recommended to be at least 128 bits.

Multiple Blocks



- How to encrypt multiple blocks?
- A new key for each block?
  - As bad as (or worse than) the one-time pad!
- Encrypt each block independently?
- Make encryption depend on previous block(s), i.e., "chain" the blocks together?
- How to handle partial blocks?

Modes of Operation

- Many modes of operation — we discuss four
- Electronic Codebook (**ECB**) mode
  - Obvious thing to do
  - Encrypt each block independently
  - There is a serious weakness
- Cipher Block Chaining (**CBC**) mode
  - Chain the blocks together
  - More secure than ECB
- Counter Mode (**CTR**) mode
  - Acts like a stream cipher
  - Popular for random access
- Also looked at Cipher Feedback (**CFB**) Mode

Type of transmission errors

- **Transmission errors** are **errors** (a 1 becomes a 0 or a 0 becomes a 1) that occur in the communication channel.
- **Transmission losses** are bits that get lost (they never arrive) in the communication channel.

Error Propagation

- A decryption process involves **error propagation** if a ciphertext input that has one incorrect bit produces a plaintext output that has more than one incorrect bit.

Revision
Lecture 3
Number Theory

Nothing…

- No direct questions on number theory…
- The lecture is for reference only
- Only look at modulo mathematics as related to public key cryptography!

Restart 15:00

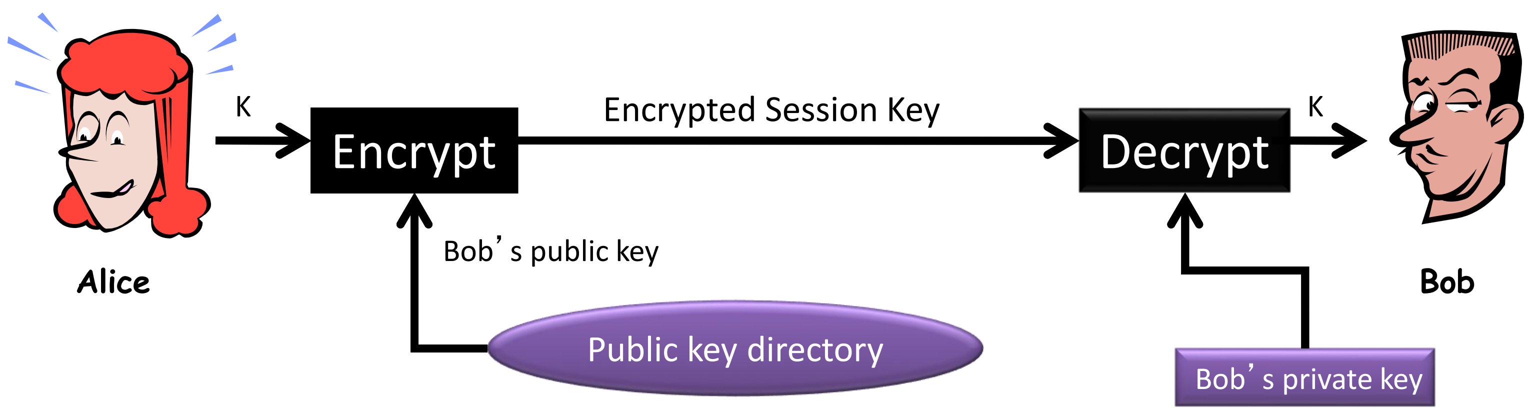
Revision

Lecture 4

Asymmetric Encryption

Public key Encryption

- Receiver Bob has a key pair: public and private
  - **publish** the public key such that the key is publicly known
  - Bob keeps the private key secret
- Other people use Bob's public key to encrypt messages for Bob
- Bob uses his private key to decrypt



- Security requirement 1: difficult to find private key or plaintext from ciphertext
- Security requirement 2: difficult to find private key from public key

What is public key crypto based on?

- Public key crypto is based on mathematical one way functions
  - Easy to compute output given the inputs
  - Difficult to compute input given the output

- Factorisation problem
  - Multiplying two prime numbers
  - Given prime x and y it is easy to compute $x.y = z$
  - Given z it is not easy to compute x and y

- Discrete logarithm problem
  - Exponentiation of a number
  - Given a, b and prime n is it easy to calculate $z= a^b \bmod n$
  - Given z, a and n it is not easy to compute b

- ‘Not easy’ means it is currently not computationally feasible…

Rivest, Shamir, and Adleman (RSA)

- Randomly choose two large and roughly equal-length prime numbers, p and q.
  - E.g. $|p| = |q| = 512$ bits
- Sets $n = pq$ (n is called the public modulus)
- Randomly choose e such that $\gcd(e, \phi(n)) = 1$.
  - e is called the public exponent.
  - $\phi(n) = \phi(pq) = (p\text{-}1)(q\text{-}1)$
- Compute d such that $de \equiv 1 \pmod{\phi(n)}$.
  - In other words, d is the modular inverse of e modular $\phi(n)$.
  - d is called the private exponent.

- Public Key: $PK = (n, e)$
- Private Key: $SK = d$

- Encryption: $C = M^e \bmod n$
- Decryption: $M = C^d \bmod n$

ElGamal Encryption Scheme

- Let p be a large prime.
- Let $Z_p^* = \{ 1, 2, 3, \ldots, p-1 \}$
- Let $Z_{p-1} = \{ 0, 1, 2, \ldots, p-2 \}$
- $a \in_R S$ means that a is randomly chosen from the set S
- Let $g \in Z_p^*$ such that none of $g^1 \bmod p$, $g^2 \bmod p$, …, $g^{p-2} \bmod p$ is equal to 1.

Public Key Pair:

- Private key: $x \in_R Z_{p-1}$
- Public key: $y = g^x \bmod p$

Encryption:

1. $r \in_R Z_{p-1}$
2. $A = g^r \bmod p$
3. $B = My^r \bmod p$ where $M \in Z_p^*$ is the message.

Ciphertext $C = (A, B)$.

Decryption:

1. $K = A^x \bmod p$
2. $M = B\,K^{-1} \bmod p$

Diffie-Hellman Key Exchange

$g^a \bmod p$ →

← $g^b \bmod p$

Alice, a

Bob, b

- Alice computes $(g^b)^a = g^{ba} = g^{ab} \bmod p$
- Bob computes $(g^a)^b = g^{ab} \bmod p$
- Could use $K = g^{ab} \bmod p$ as symmetric key

- This key exchange scheme is secure against eavesdroppers if Diffie-Hellman Problem is assumed to be hard to solve.
- However, it is insecure if the attacker in the network is active: man-in-the-middle attack. "Active" means that the attacker can intercept, modify, remove or insert messages into the network.

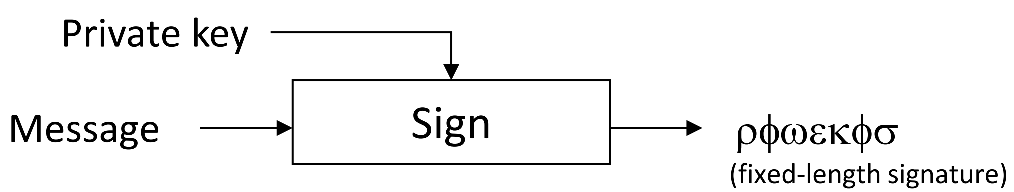
Revision
Lecture 5
Integrity

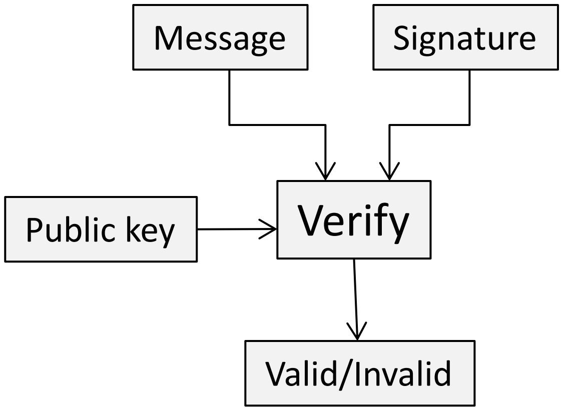
Integrity

- Can encryption also provide integrity services? Does encrypting a message prevent:
  - Changing part of a message
  - Deletion of part of a message
  - Insertion of a false message
  - Falsifying the origin of a message
- Levels of integrity
  - Detect (accidental)modification
  - Data origin authentication (verify origin/no modification)
  - Non-repudiation (only one person generated this message)

Digital Signature

- Use asymmetric cryptography
- Only one party should be able to sign
  - Sign using Alice's private key (signing key)
  - Verify using Alice's public key (verification key)





- ❑ Only the signer (who has a private key) can generate a valid signature
- ❑ Anyone (since the corresponding public key is published) can verify if a signature with respect to a message is valid

Hash Function

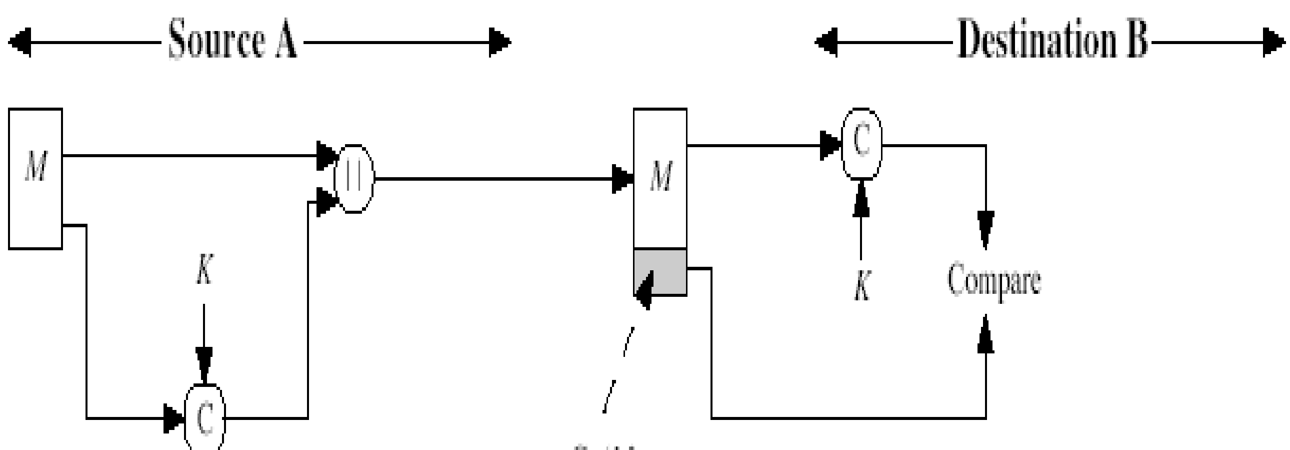
- A cryptographic hash function h(x) should provide
  - Two functional properties
    - Compression – arbitrary length input to output of small, fixed length
    - Easy to compute – expected to run fast
  - Three security properties
    - One-way – given a hash value y it is infeasible to find an x such that h(x) = y (also called pre-image resistance)
    - Second pre-image resistance – given y and h(y), cannot find x where h(x)=h(y)
    - Collision resistance – infeasible to find x and y, with $x \neq y$ such that h(x) = h(y)
- Note: As h is a compression algorithm, there should theoretically be collisions. Collision resistance require that it is hard to find any collision
- Who can search for a collision?

MAC

- How MAC Works
  - A MAC is a symmetric cryptographic mechanism
  - Sender and receiver share a secret key K

1. Sender computes a **MAC tag** using the message and K; then sends the MAC tag along with the message
2. Receiver computes a MAC tag using the message and K; then compares it with the MAC tag received. If they are equal, then the receiver concludes that the message is not changed

  - Note: only sender and receiver can compute and verify a MAC tag



(a) Message authentication

Revision
Lecture 6
Authentication

Entity authentication

Unilateral authentication:

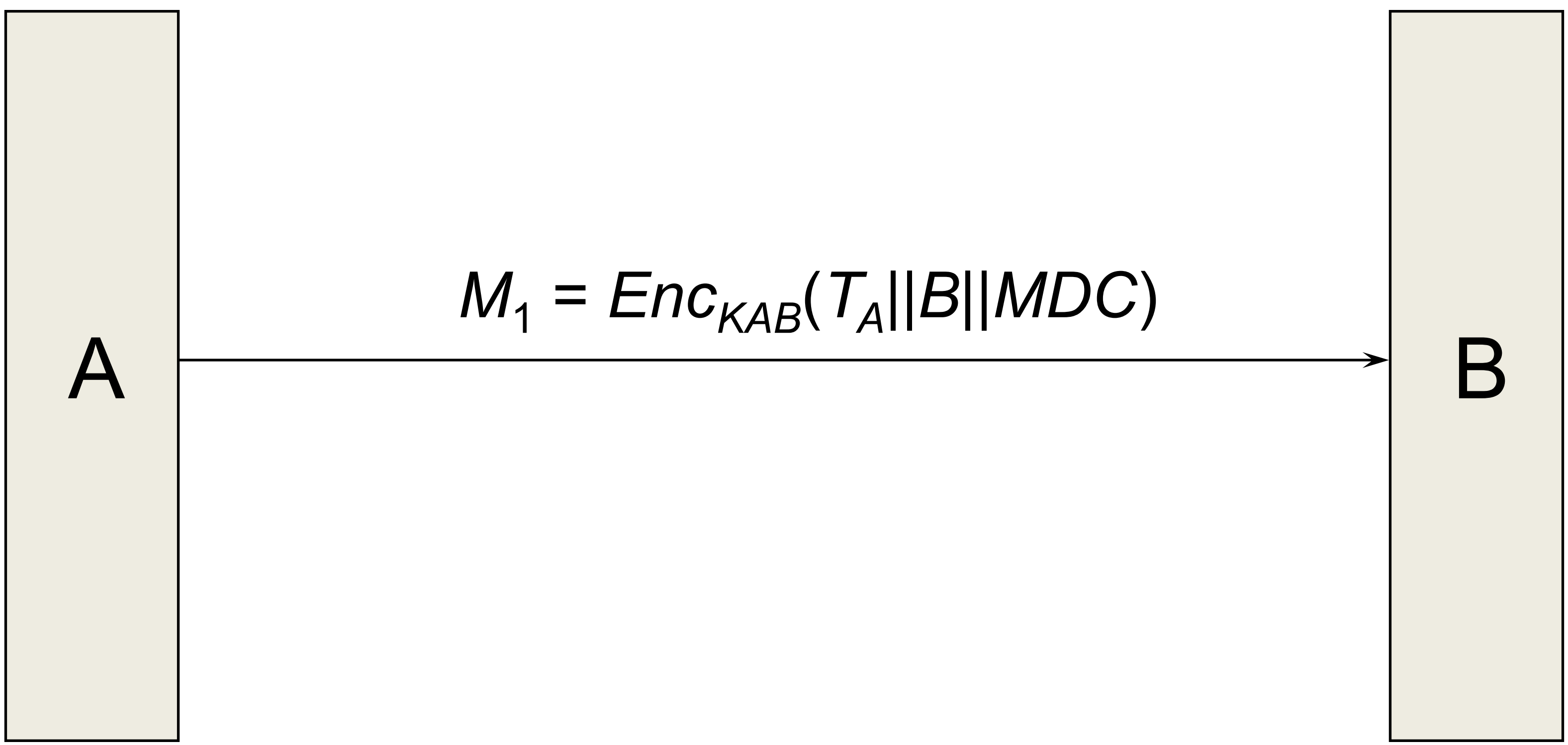
- entity authentication which provides one entity with assurance of the other’s identity but not vice versa.

- Mutual authentication:
  - entity authentication which provides both entities with assurance of each other’s identity.

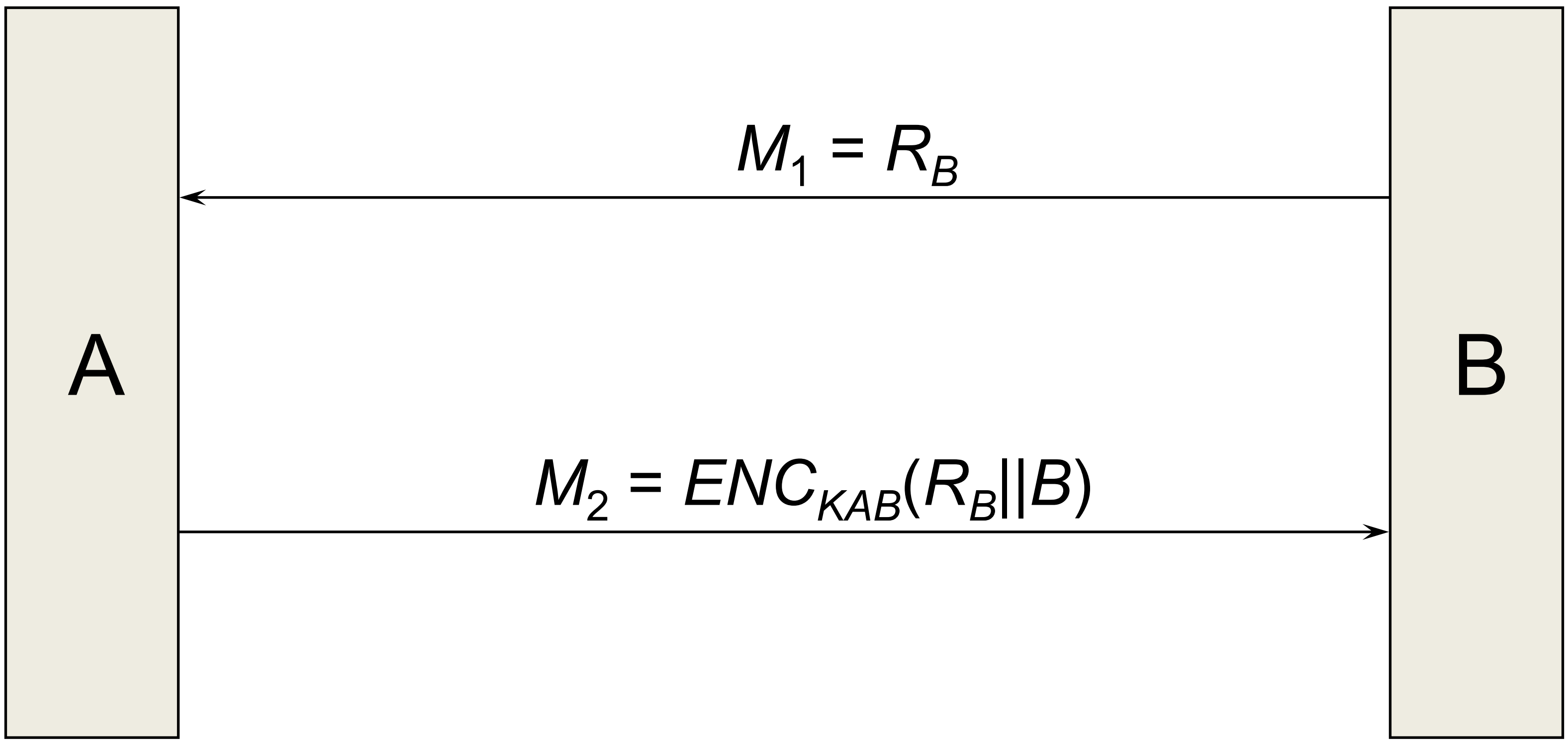
Entity authentication

- A verifier only sends/receives messages, i.e. digital data.
- To check that the principal is online the verifier need to establish:
  - that the messages came from the principal (origin authentication),
  - and that the messages have been recently generated (freshness).
- If both conditions are satisfied then we have authenticated the claimant.

Example 1 (timestamp & encryption)



Example 2 (nonce & encrypt)



Should be able to design similar protocols for ENC, MAC, SIG(Timestamp or nonce, mutual or unilateral).

Revision
Lecture 7
Key Management

Terms for key management

- *Key establishment*: Process of making a secret key available to multiple entities.
- *Key agreement*: Process of establishing a key in such a way that neither entity has key control.
- *Key transport*: Process of securely transferring a key from one entity to another.
- *Key control*: Ability to a choose a key's value.
- *Key confirmation*: Assurance that another entity has a particular key.

Symmetric-key protocols

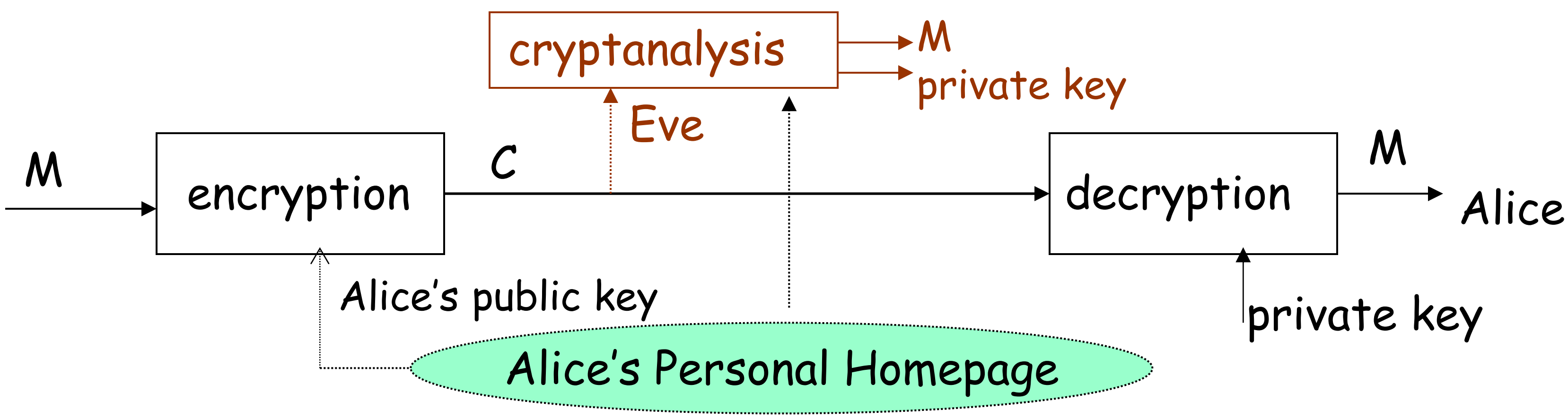
- The use of symmetric-key cryptography to produce a shared symmetric secret key.
- The protocols can be classified as:
  - Directly communicating entities
  - Use of a Key Distribution Centre (KDC) (Agreement)
  - Use of a Key Translation Centre (KTC) (Transport)

Public-key protocols

- The use of public-key cryptography to produce a shared symmetric secret key.
- The protocols can be classified as:
  - Key transport protocols (typically involving public-key encryption and digital signatures)
  - Key agreement protocols (indirectly specified by mostly based on the Diffie-Hellman protocol)

Digital Certificates

- Public key encryption: encrypt using receiver's public key
  - sender **has to be sure** that the public key used for encryption is indeed the receiver's public key
- Digital signature: verify a signature
  - Verifier **has to be sure** that the public key used for signature verification is indeed the signer's public key
- **How can the encryptor / verifier be sure that the public key is authentic?**

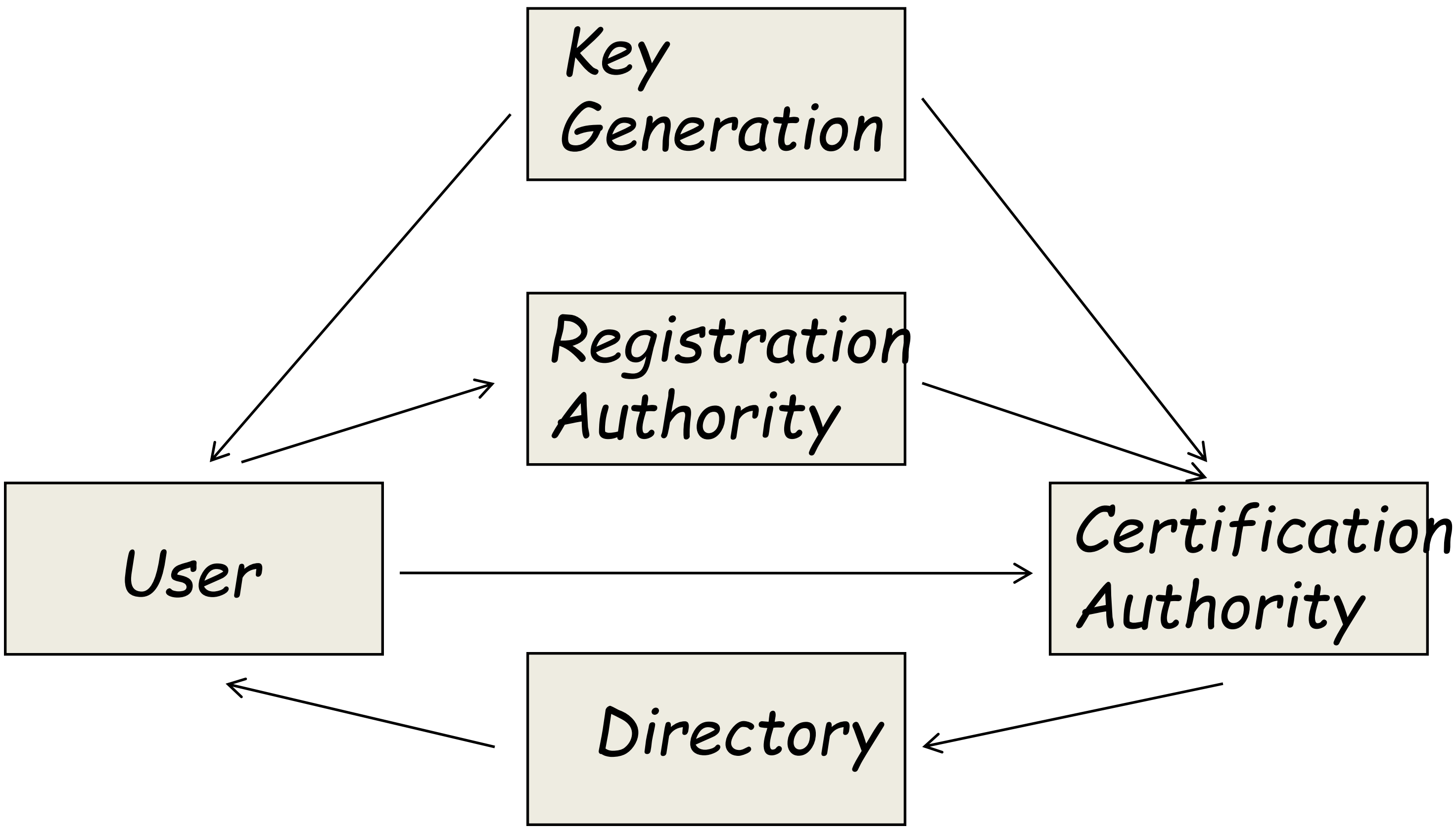


- How about posting the public key at a personal homepage?
- How about sending the public key to the encryptor / verifier using email?

The Certification Authority

- The “CA” is responsible for:
  - identifying entities before certificate generation
  - Generating user key or verifying user key
  - ensuring the quality of its own key pair,
  - keeping its private key secret.
- The CA, before generating a certificate, ought to check that a user knows the private key corresponding to its claimed public key.

Who is involved?



Revision
Lecture 8
Computer Security

Who Goes There?

- How to authenticate a human to a machine?
- Can be based on…
  - Something you **know**
    - For example, a password
  - Something you **have**
    - For example, a smartcard
  - Something you **are**
    - For example, your fingerprint

2-factor Authentication

- Requires 2 out of 3 of
  1. Something you know
  2. Something you have
  3. Something you are
- Examples
  - Password + Security Token
  - ATM: Card and PIN
  - Password + Cellphone (e.g. SMS)

Password File

- Bad idea to store passwords in a file
- But need a way to verify passwords
- Cryptographic solution: **hash** the passwords
  - Store $y = h(\text{password})$
  - Can verify entered password by hashing
  - If attacker obtains password file, he does not obtain passwords
  - But attacker with password file can guess x and check whether $y = h(x)$
  - If so, attacker has found password!

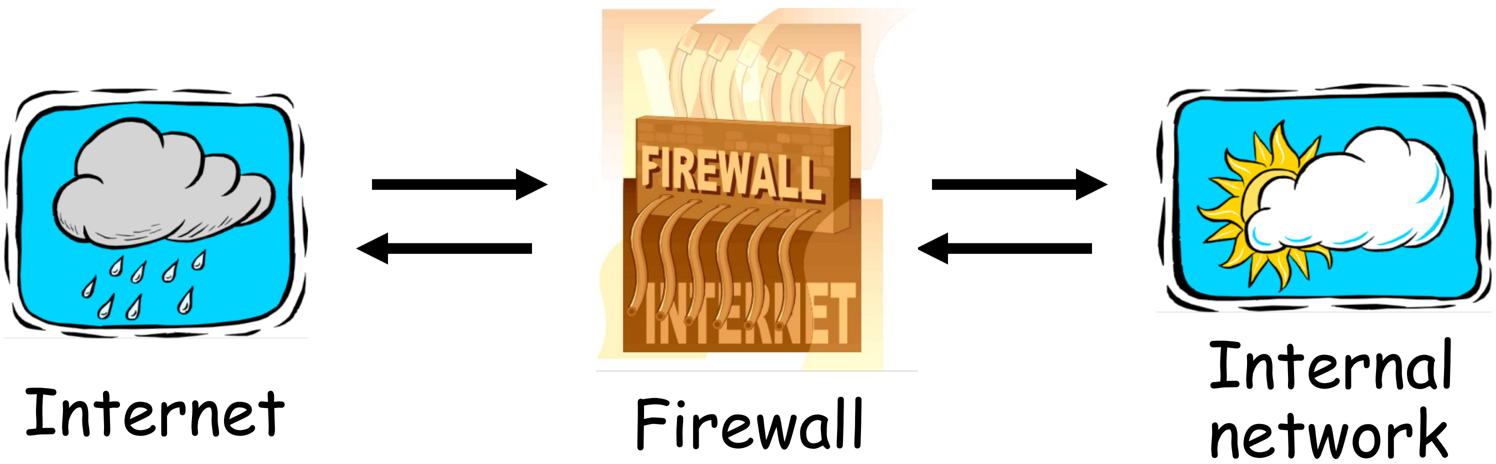
Password File

- Store hashed passwords
- Better to hash with **salt**
- Given password, choose random s, compute

$$y = h(\text{password}, s)$$

  and store the pair (s,y) in the password file
- Note: The salt s is **not secret**
- Easy to verify password
- Attacker must recompute dictionary hashes for each user — lots more work!

Firewalls



- Firewall must determine what to let in to internal network and/or what to let out
- **Access control** for the network

Firewall Terminology

- No standard terminology
- Types of firewalls
  - **Packet filter** — works at network layer
  - **Stateful packet filter** — transport layer
  - **Application proxy** — application layer
  - Personal firewall — for single user, home network, etc.

Malicious Programs

Requires A Host Program

- Trapdoor/Backdoor
- Logic bombs
- Trojan horses
- **Viruses**

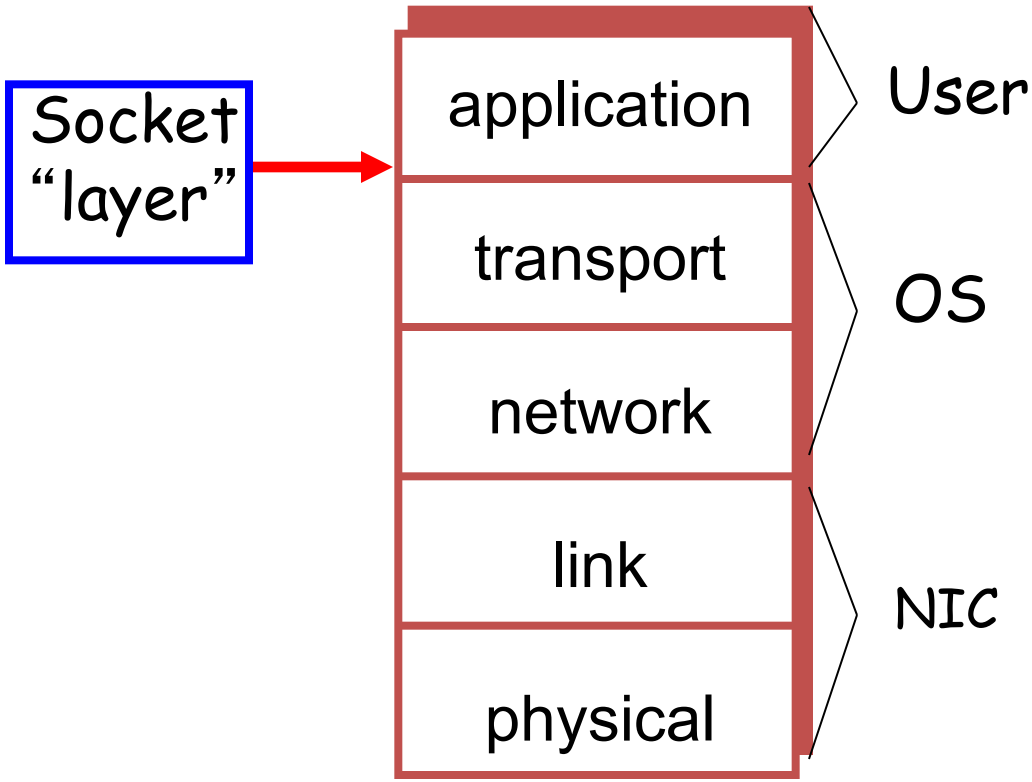
Independent of Host Programs

- Bacteria
- **Worms**

Revision
Lecture 9
Network Security

Socket layer

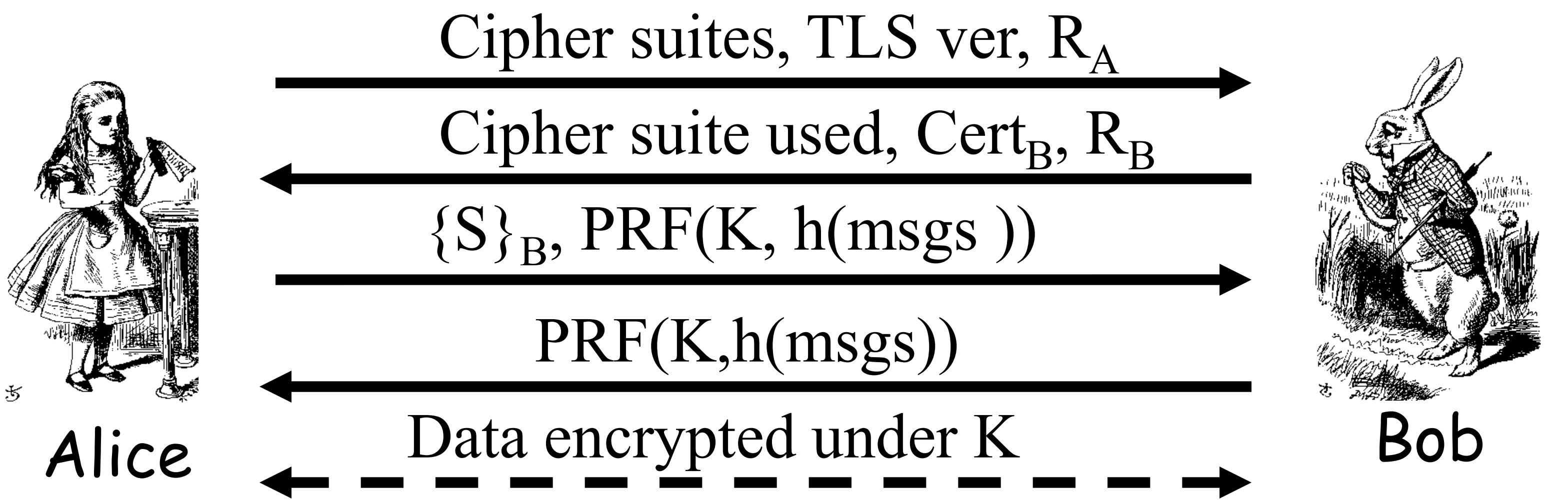
- "Socket layer" lives between application and transport layers
- SSL usually lies between HTTP and TCP



What is SSL?

- SSL is the protocol used for most secure transactions over the Internet
- For example, if you want to buy a book at amazon.com…
  - You want to be sure that you are dealing with Amazon (**one-way authentication**)
  - Your credit card information must be protected in transit (**data confidentiality**)
  - As long as you have money, Amazon doesn't care who you are (**authentication need not to be mutual**)

Simplified SSL Handshake Protocol

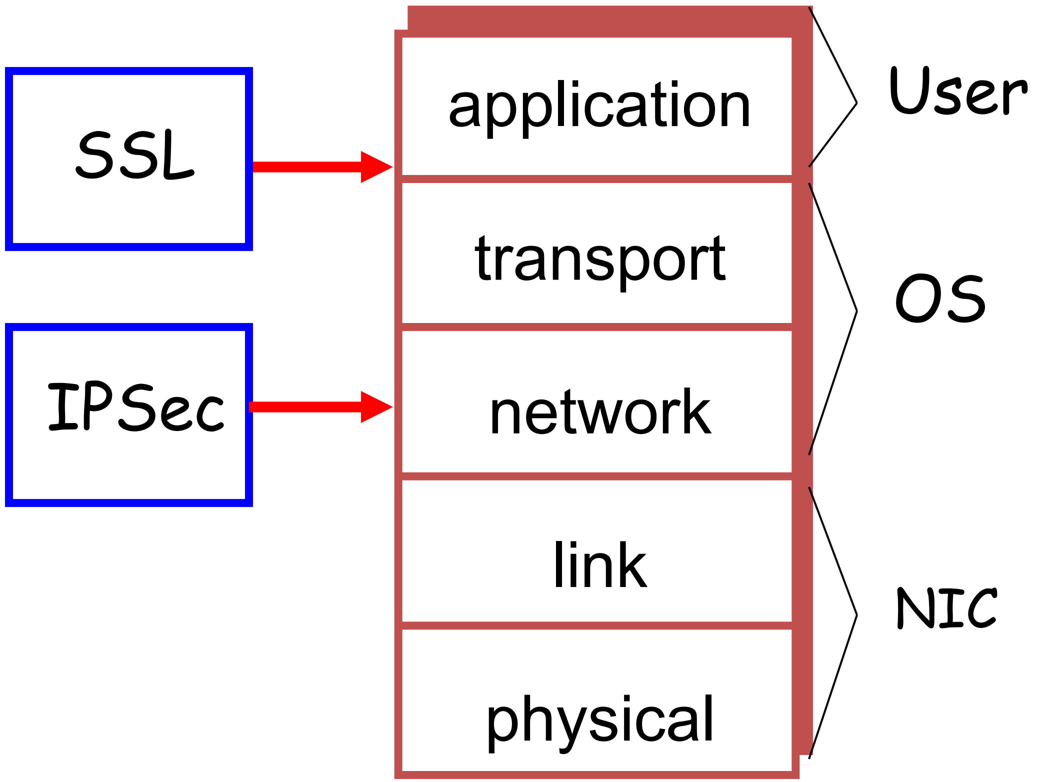


- S is **randomly chosen by Alice**
- $K = h(S,R_A,R_B)$
- msgs = all previous messages

IPSec
(Network Layer Security)

IPSec and SSL

- IPSec lives at the network layer
- IPSec is transparent to applications



IKE Phase 1

- Three ways to run phase 1 (generate proofs)
  - Public key encryption based
  - Signature based
  - Symmetric key based
- For each of these, there are two different "modes" to choose from
  - Main mode
  - Aggressive mode
- **There are 6 variants of IKE Phase 1!**

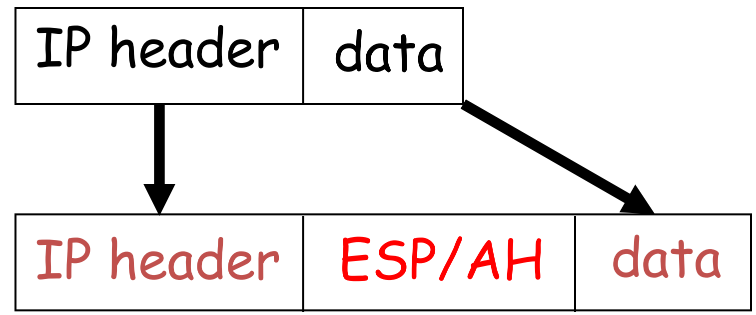
ESP and AH

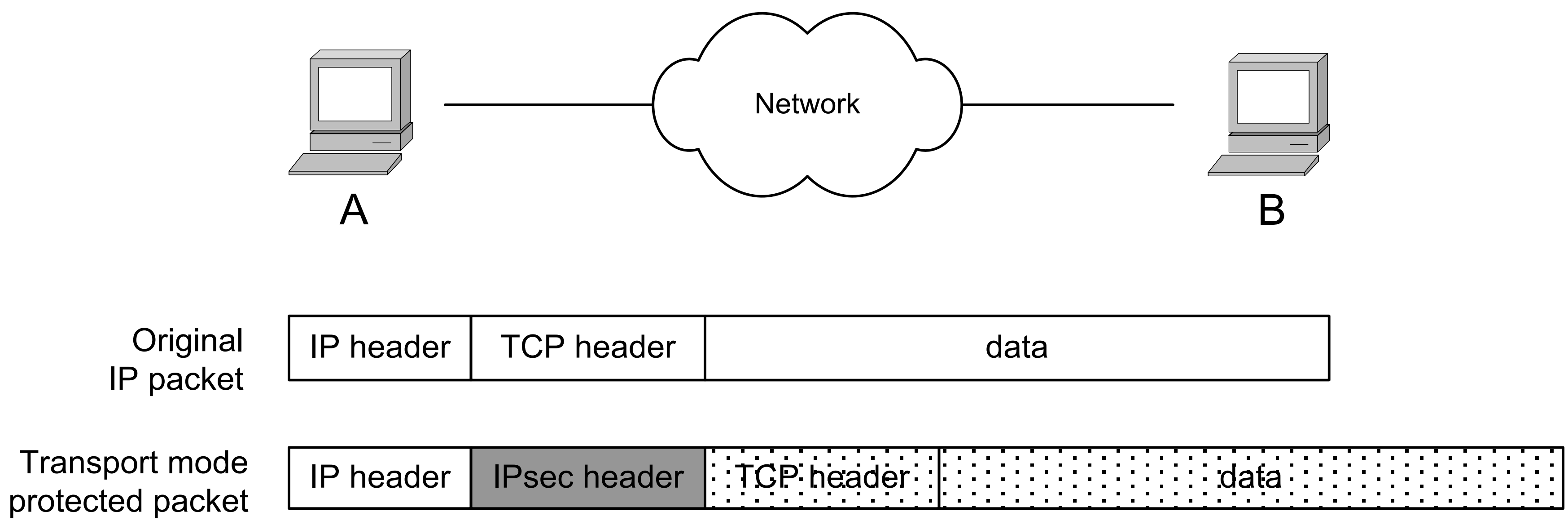
- Two Encapsulation modes
  1. Transport mode
  2. Tunnel mode

- Two Protocols
  - AH – Authentication Header – support message authentication only
  - ESP – Encapsulating Security Payload
    1. Encryption only
    2. Encryption with message authentication

IPSec Transport Mode

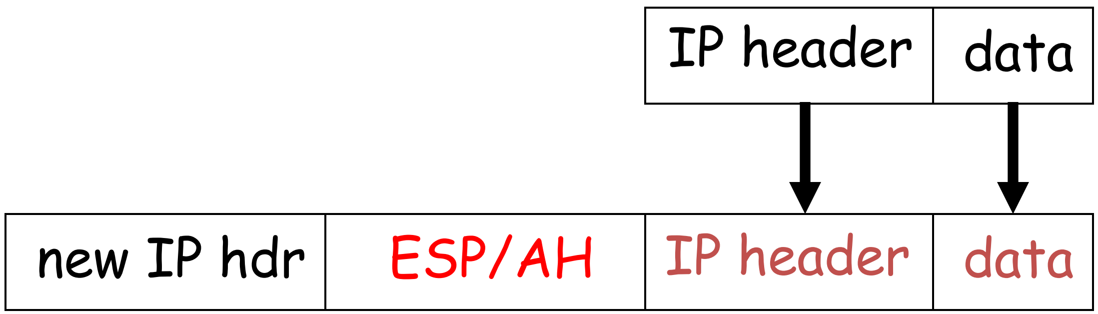
- Transport mode designed for host-to-host
- The original header remains
  - Passive attacker can see who is talking





IPSec Tunnel Mode

- IPSec Tunnel Mode



- Tunnel mode for gateway to gateway VPN
- Original IP packet encapsulated in IPSec
- Original IP header not visible to attacker
  - New header from firewall to firewall
  - Attacker does not know which hosts are talking

GSM: Authentication

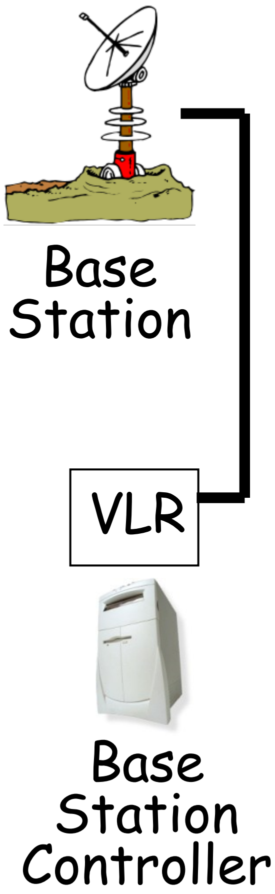
- Caller is authenticated to base station
- Authentication is **not** mutual
- Authentication via **challenge-response**
  - AuC generates RAND and computes XRES = A3(RAND, Ki) where A3 is a hash
  - Then (RAND,XRES) are sent to base station
  - Base station sends **challenge** RAND to mobile
  - Mobile's **response** is SRES = A3(RAND, Ki)
  - Base station verifies SRES = XRES
- **Note:** Ki never leaves AuC!
- The response length should be long enough to discourage online guessing. E.g. 32 bits
- Random challenge should be long enough to reduce the chance of generating repeated challenge numbers. E.g. 128 bits

GSM: Confidentiality

- Data encrypted with stream cipher, A5
- Encryption key Kc
  - AuC computes Kc = A8(RAND, Ki), where A8 is a hash
  - Then Kc is sent to base station with RAND
  - Mobile computes Kc = A8(RAND, Ki) after receiving RAND
  - The value of RAND is the same as the one used for authentication
  - Keystream generated from A5(Kc)
- **Note:** Ki never leaves home network!
- Ki is 128 bits long
- Kc is 64 bits long

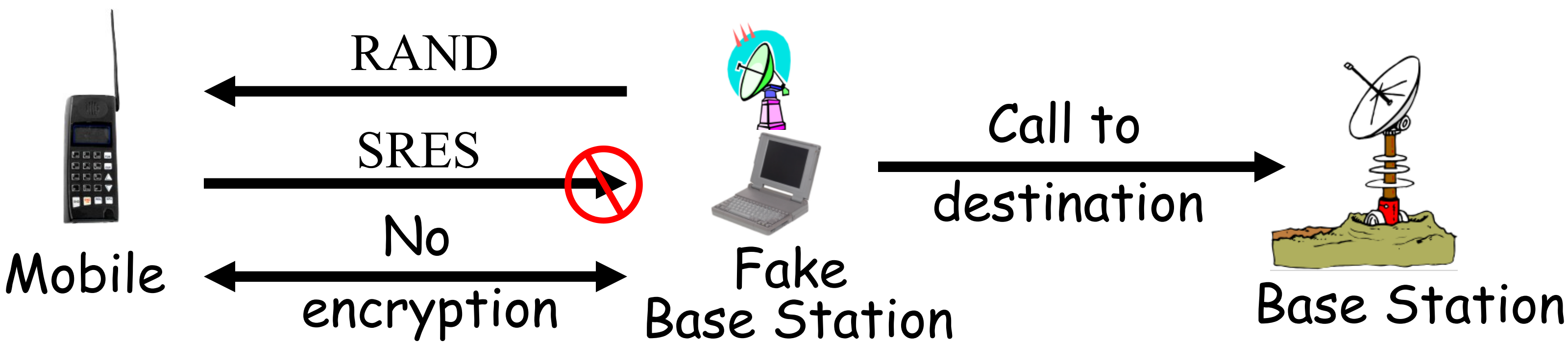
GSM Insecurity (1)

- Hash used in A3/A8:
  - Broken after 160,000 chosen plaintexts
  - With SIM, can get Ki in 2 to 10 hours
- Encryption between mobile and base station but **no encryption** from base station to base station controller
  - When transmitted over microwave link…
- Encryption algorithm A5/1
  - Broken with 2 seconds of known plaintext



GSM Insecurity (2)

- **Fake base station** exploits two flaws
  - Encryption not automatic
  - Base station not authenticated



3GPP: 3rd Generation Partnership Project

- 3G fixes known GSM security problems
  - Mutual authentication
  - Keys (encryption/integrity) cannot be reused
  - Triples cannot be replayed
  - Strong encryption algorithm (AES)
  - Message authentication
  - Encryption extended to base station controller
- http://www.3gpp.org

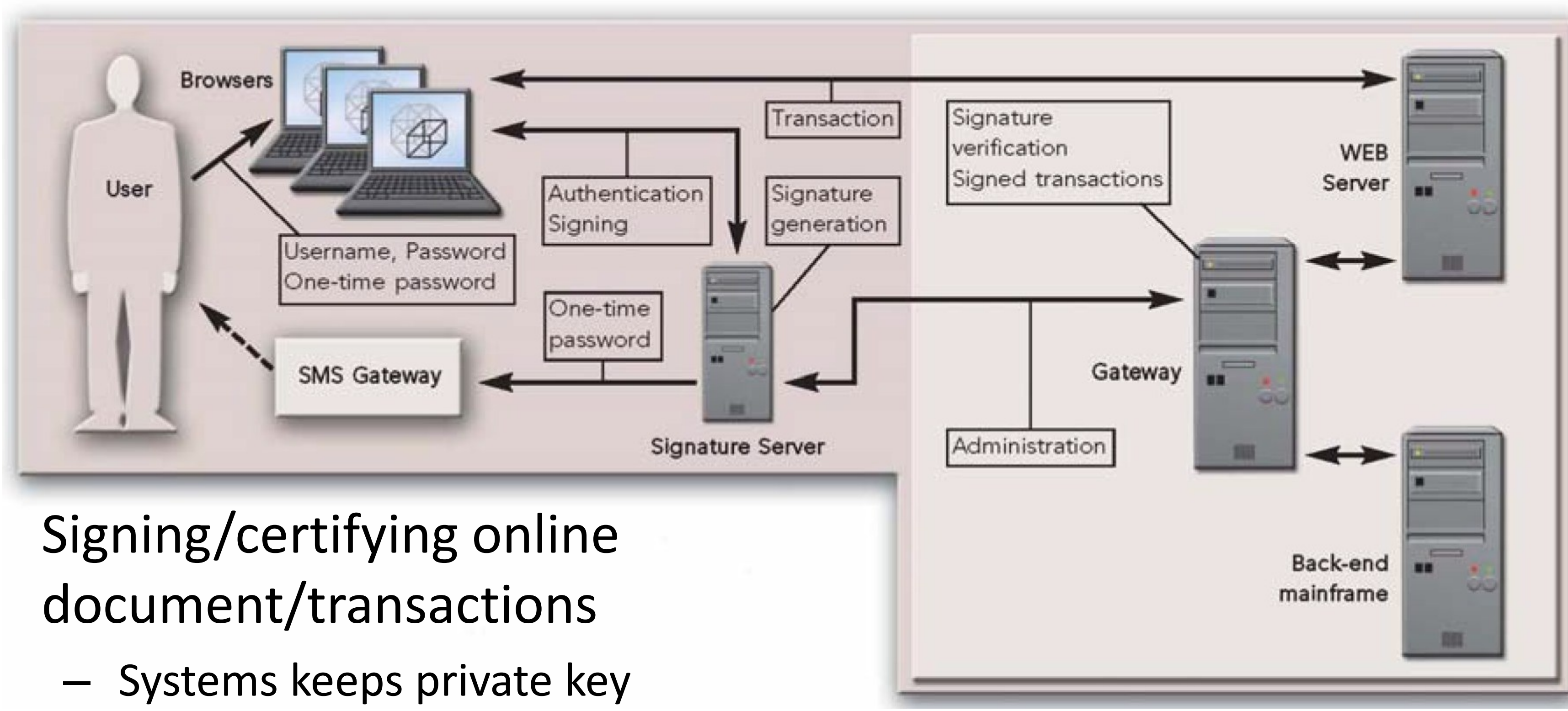
Course Context:
Real World

A long time ago…Lecture 1

- What are we going to do?
  - This is introductory course on information security
  - If you want deeper knowledge on specific topics
    - CS 5293 Topics on Information Security
    - CS 6290 Privacy Enhancing Technologies
  - We did say you will learn enough to meaningfully think about security of e-commerce systems….
  - So what have you learned anything for real world?

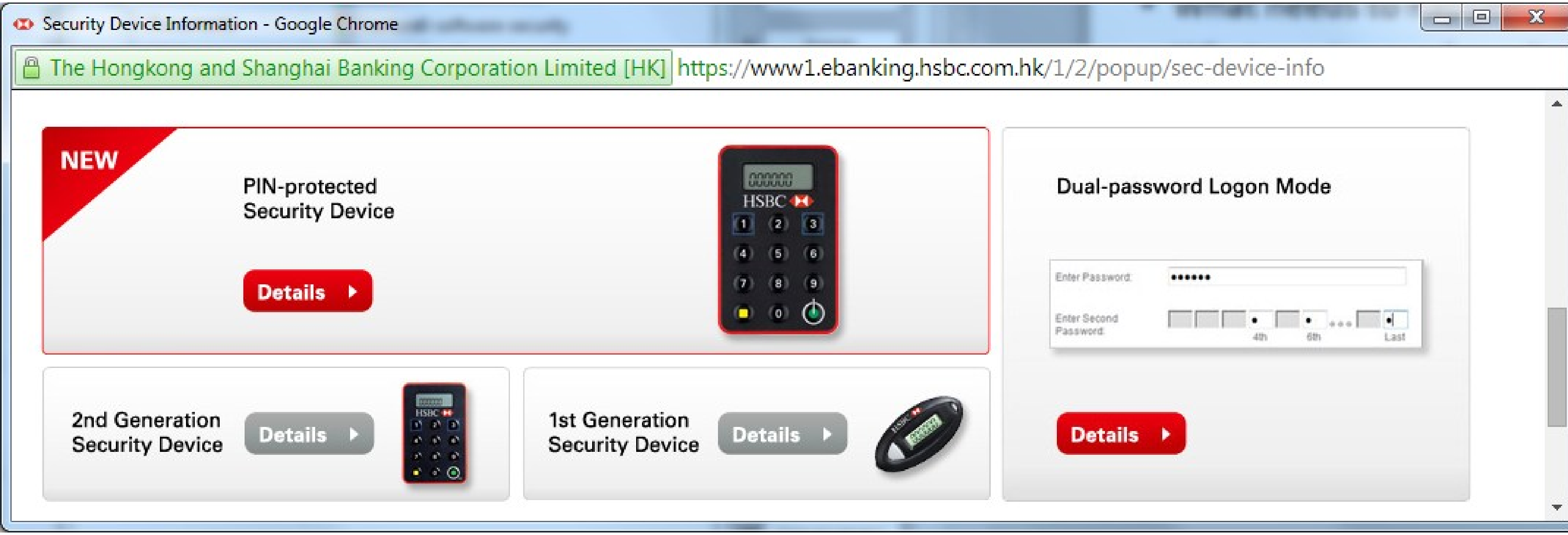
  Lets look at some examples…

Understand Product Advert? Cryptomathic Signer



- Signing/certifying online document/transactions
  - Systems keeps private key
- Description: Signatures, passwords, OTP, firewalls (not shown)

Online transactions?



- Authenticate the server and confidentiality? Lecture 9
  - TLS/SSL (see https and certificate valid green highlight)
- Authenticate the user? Lecture 8
  - Remember OTP (one-time password) generators
    - Time-based…..
  - Multi-factors authentication (1st gen and 2nd gen differs in factors)
  - Passwords!!!! How should we store passwords?
    - Look at dual-password method – how is phishing/shoulder surfing mitigated?
    - Partial password entry….

Understand Actual Secure Systems
Payment Card

- What needs to happen?
- What services and mechanisms are needed?
  - Authenticate the card? Lecture 6
  - Authenticate the user? Lecture 8
  - Key management (card/terminal+card/bank)? Lecture 7
  - Signature(integrity/non-repudiation)? Lecture 4/5
  - Encryption? Lecture 2

Payment Card

- Payment card issued by BankB
- User signature or PIN
- Card talks to payment terminal
  - Signs transaction information (card private key)
  - Sends card certificate (card public key)
  - Payment terminal verifies card signature (it permanently stores certificate of BankB)
- Card could talk to BankB directly
  - Shared symmetric key
  - Message sent to terminal, forwarded to bank

Two payment systems…

- Think critically about security…
- Card authentication
  - Proposal 1:

    Card >> Reader

    {'I am legitimate'}Sign_card
  - Proposal 2:

    Card >> Reader

    {'I am legitimate', R_term, transaction msg}Sign_card

Which one is better? Why?

Proposal 2 – freshness!

Thank you!

The end!