

CS 6355/4355: Cryptanalysis and Database Security

Topic: Introduction to Blockchain Technology

Lecturer: Rongxing LU

Email: RLU1@unb.ca Office: GE 114

Website: <http://www.cs.unb.ca/~rlu1/>

Faculty of Computer Science, University of New Brunswick

Outline

- Technical Foundation of Blockchain
 - Focus on BitCoin Technique
- Current Situations of Blockchain Research



What is Blockchain Technology?

- Blockchain technology is a digital innovation that is poised to significantly alter financial markets within the next few years, within a cryptographic ecosystem that has the potential to also significantly impact trusted computing activities and therefore cybersecurity concerns as a whole.

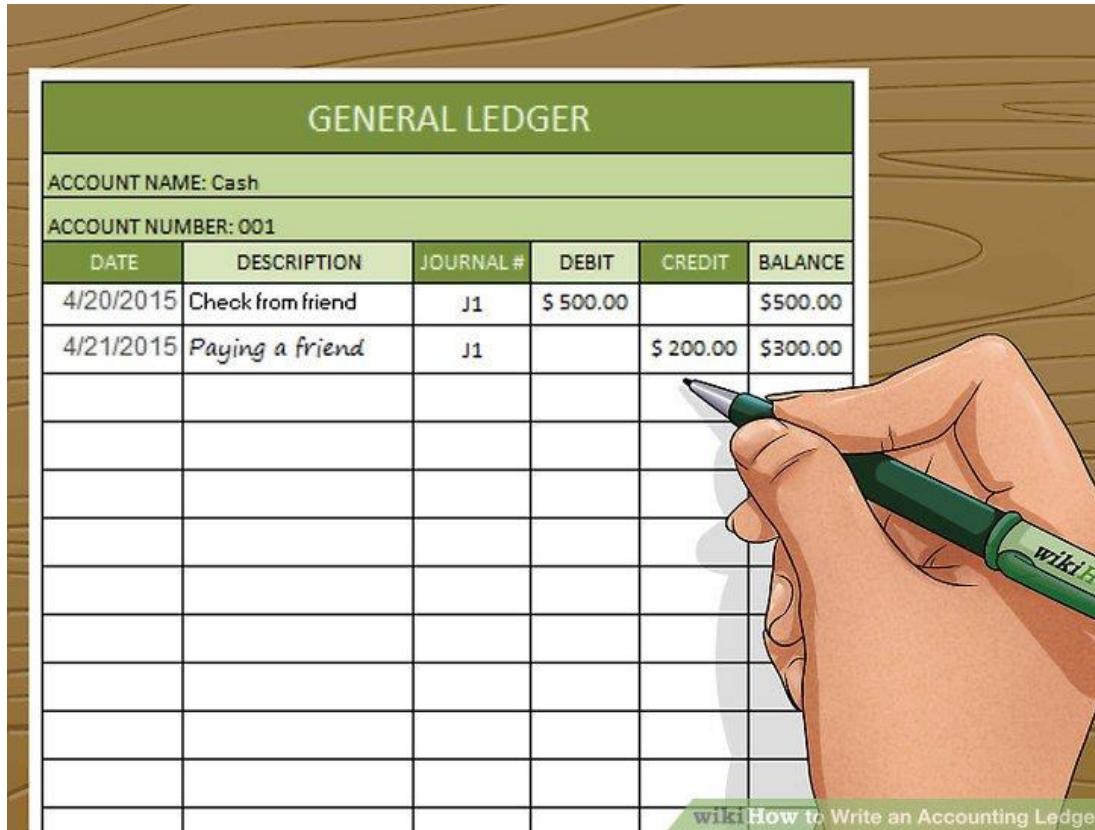


The “Human Body” of Blockchain



Skeleton:	Distributed ledger	Blood:	Cryptography
Muscle:	Game theory	Brain:	Smart contract

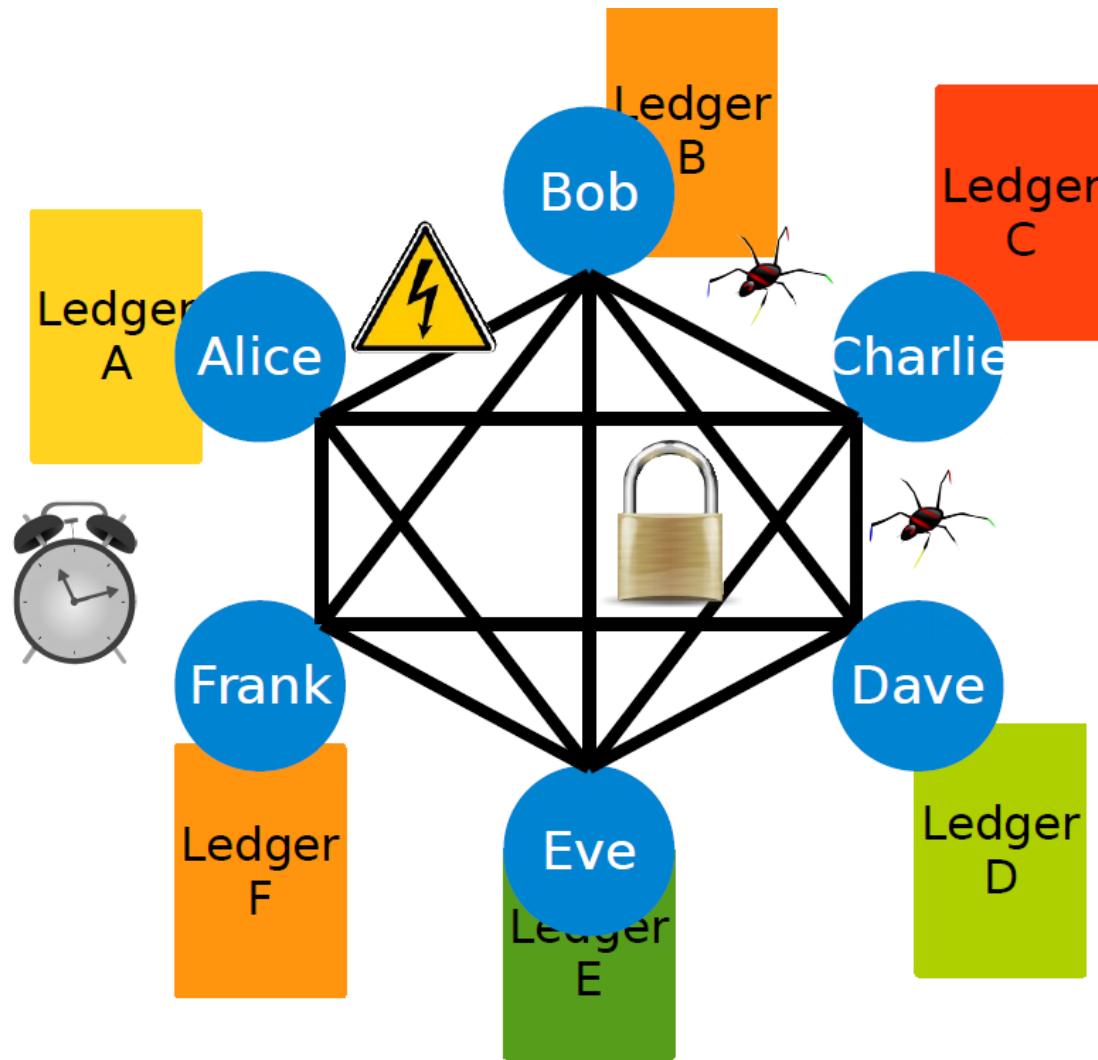
What is Ledger ?



- Ledger records all business activity as transactions (in databases)
- Every market and network defines a ledger
- Ledger records asset transfers between participants

Datum	Entnahme und Abhebungen M€	Lieferungen, Gutsabholen, Gutsabliefern M€	Befand der Schuld M€	Befand des Guthabens M€
19.2.			10.9.81	
Beginn				
12. Au	2.000,- M€	Bankdepot	89,-	11.13.81
23.	- 2.100,- M€	Sparschwein	102.90	8.6.24
24.	- 2.50,- M€	Bauspardepot	3.259	5.2.33
9.	- 10.00,- M€	Bankdepot	6.50	4.3.10.80
12.	1.500,- M€	Bankdepot	4.6.50	13.5.9.31
21.	- 500,- M€	Zinselzweckdepot	72.50	18.5.9.30
20.	- 1.250,- M€	Bankdepot	6.250	1.3.12.30
24.	- 1.350,- M€	Konto	6.78.55	1.3.13.55
29.	1.000,- M€	Bankdepot	4.6.50	2.3.1.15
18.	- 2.500,- M€	Konto	152.50	8.3.3.95
31.	Zinselzweckdepot gez. 21.12.92		30.05	9.4.3.90
19.3.				
Jan.	31.5,- M€	Postkasse	4.80	
	51,- M€	Reise - Postkasse	32.2,-	1.05.7.93
	1.20,- M€	Handelsgeld		
	585,- M€	Lebensmittel		
	50,- M€	Reisekosten, 50,- M€		
	52,- M€	Magnetrührer		
			135.48	1.1.3.81

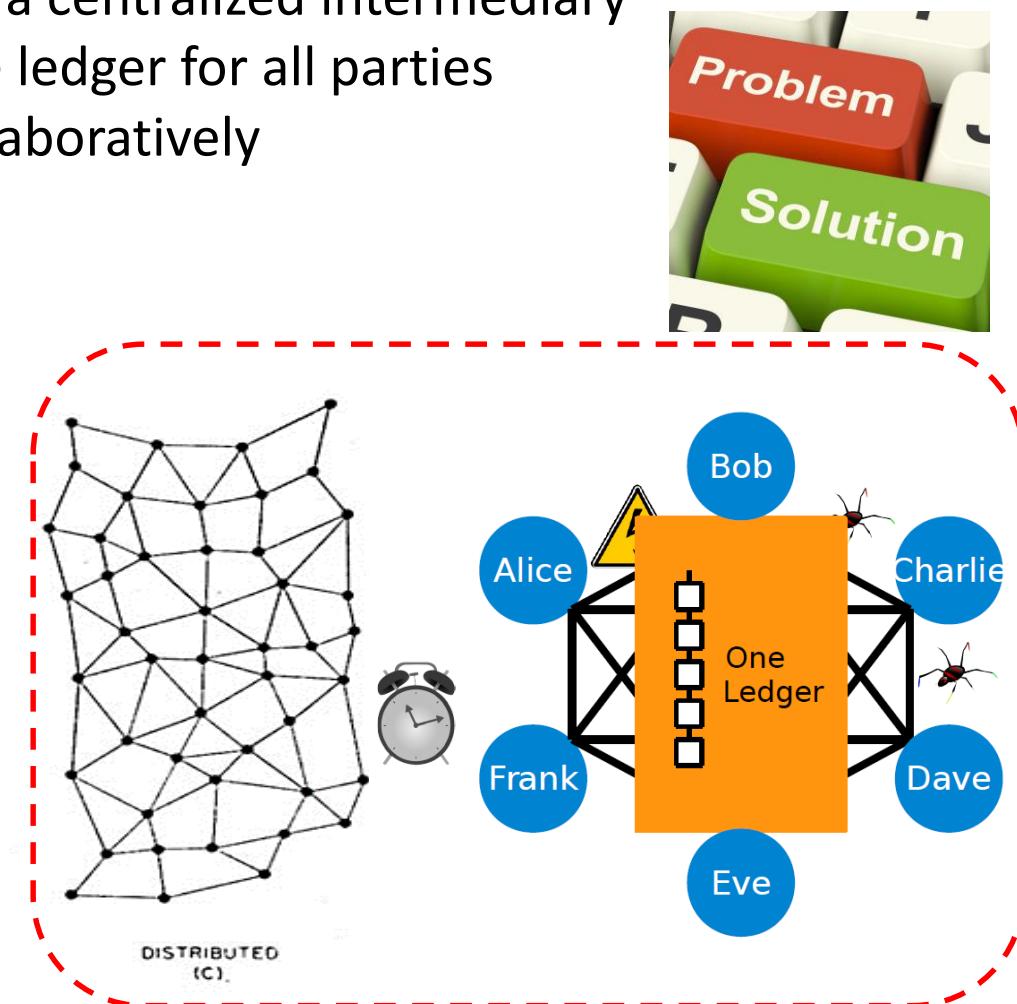
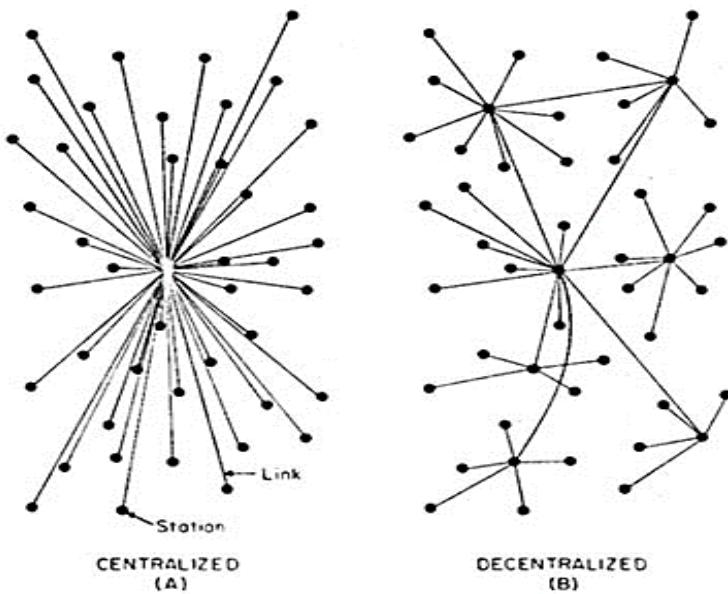
Multiple Ledgers



- Problems
 - Every party keeps its own ledger and state
 - Problems, incidents, faults
 - Diverging ledgers

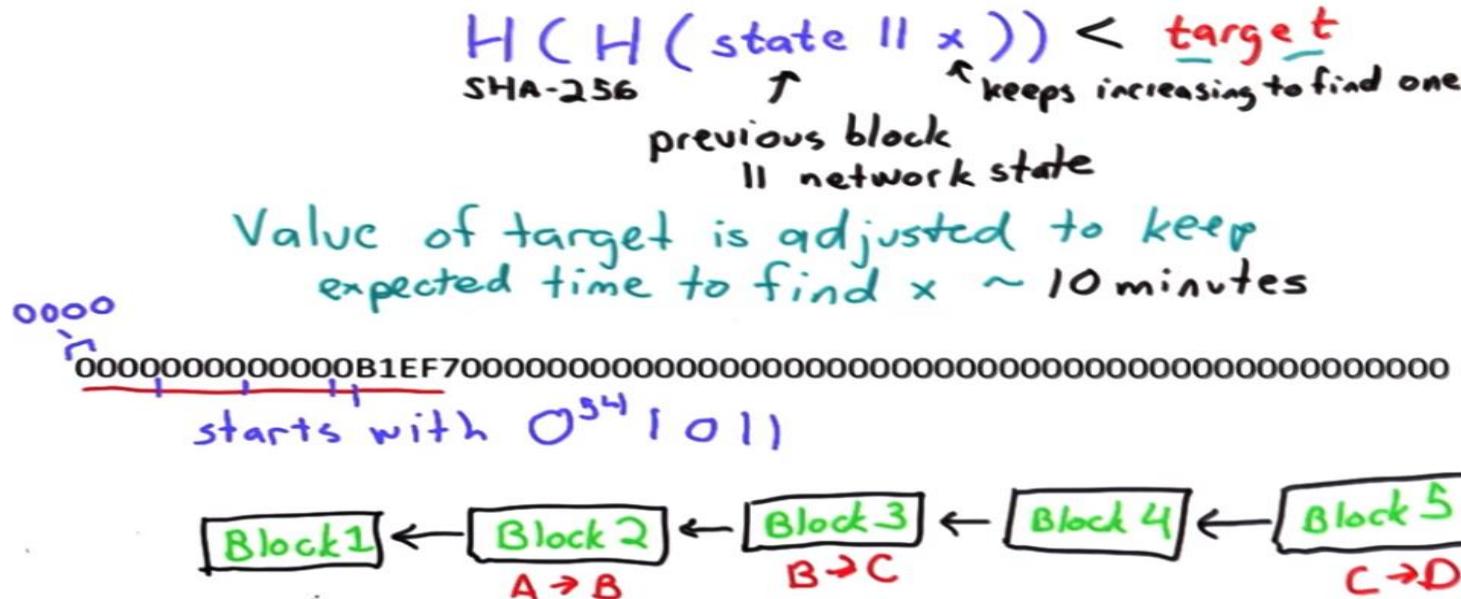
Blockchain provides one virtual ledger

- One common trusted ledger
- Today often implemented by a centralized intermediary
- Blockchain creates one single ledger for all parties
- Replicated and produced collaboratively
- Trust in ledger from
 - **Cryptographic protection**
 - **Distributed validation**



Cryptography in Blockchain

- Integrity of ledger
- Authenticity of transactions
- Privacy of transactions
- Identity of participants



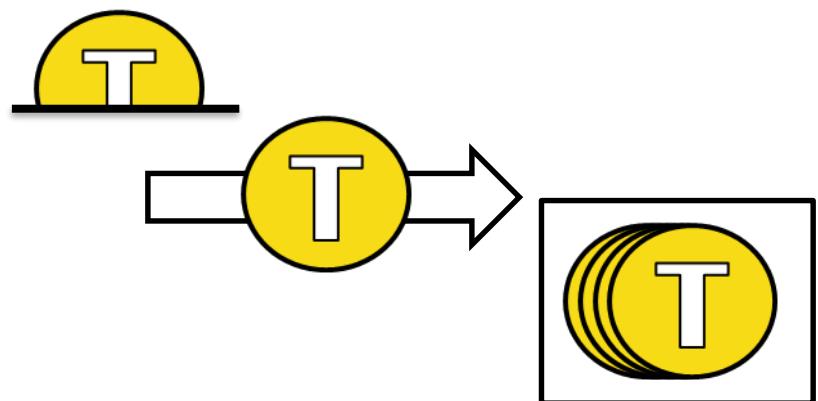
Cryptography in Bitcoin

- Cryptography has been a key technology in the financial world for decades
 - Payment networks, ATM security, smart cards, online banking ...
- **Bitcoin** started in 2009
 - First prominent use of cryptography for a new trust model (= trust no entity)
- The promise of Blockchain – Reduce trust and replace it by technology
 - Exploit advanced cryptographic techniques



Algorithms in Bitcoin

- TheoryCoin
 - How to ***create*** coins
 - How to ***transfer*** coins
 - How to ***store*** coins



- Difference between TheoryCoin and BitCoin
- Problems



TheoryCoin (TC): How to create money

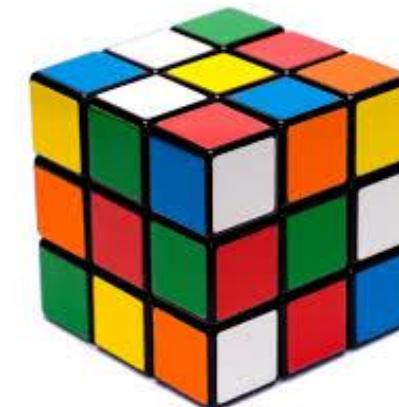


1. Everyone
tries to solve a puzzle

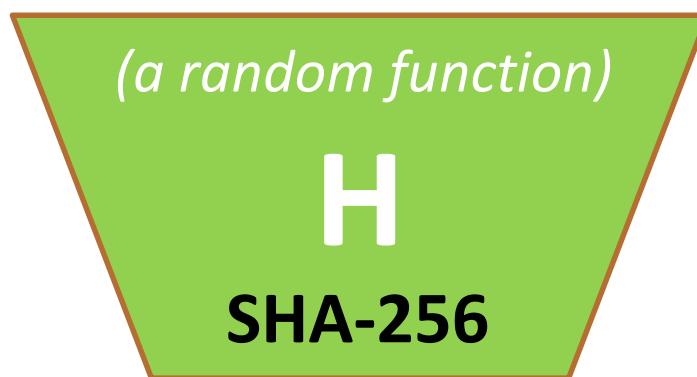
2. The **first one** to solve
the puzzle **gets 1 TC**

3. The solution of **puzzle i**
defines puzzle $i+1$

3		2	4		6		
	4					5	3
1	8	9	6	3	5	4	
			8		2		
	7	4	9	6	8		1
8	9	3	1	5		6	4
			1	9	2	5	
2			3			7	4
9	6		5		3		2



TheoryCoin (TC): How to create money

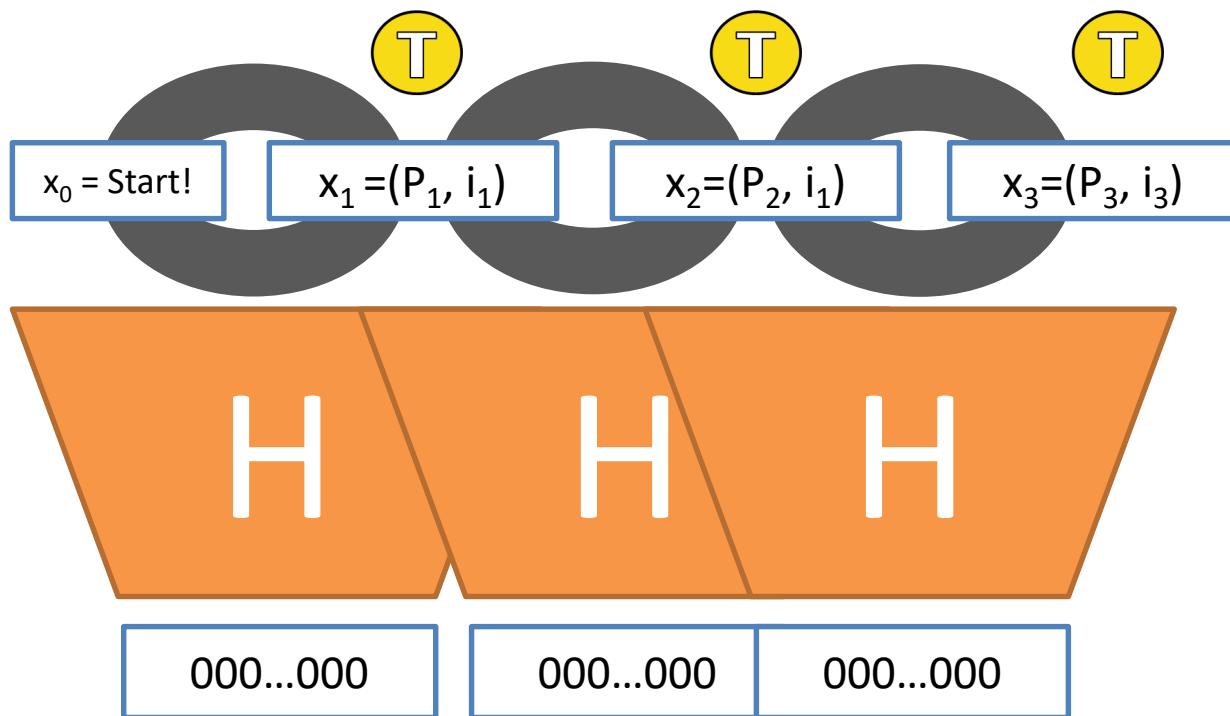
 $L \in \{0,1\}^*$ $R \in \{0,1\}^*$  $T \in \{0,1\}^d$

The puzzle:
given L , find R
such that $T=0^d$

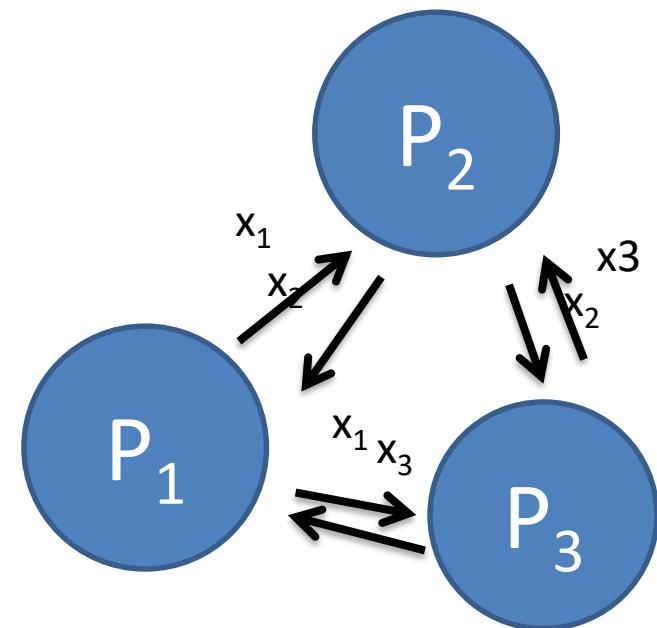
```
SolvePuzzle (L) {  
    repeat {  
        R = my_name || i++  
        T = H(L, R)  
    } while (T ≠ 0d)  
    return R  
}
```

Proof-of-Work

TheoryCoin (TC): How to create money

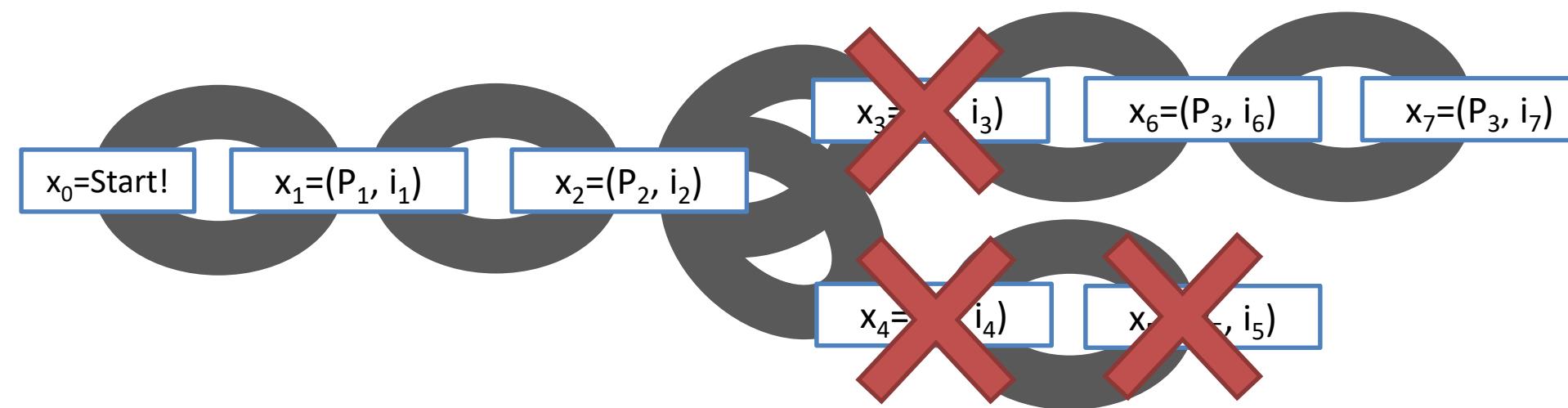


```
SolvePuzzle(L) {  
    repeat{  
        R = my_name || i++  
        T = H(L, R)  
    }while(T ≠ 0d)  
    return R  
}
```



* aka *the blockchain*

TheoryCoin (TC): How to create money



* aka *the 51% attack*

TheoryCoin (TC): How to create money



Recap:

Solve the next puzzle → get a coin

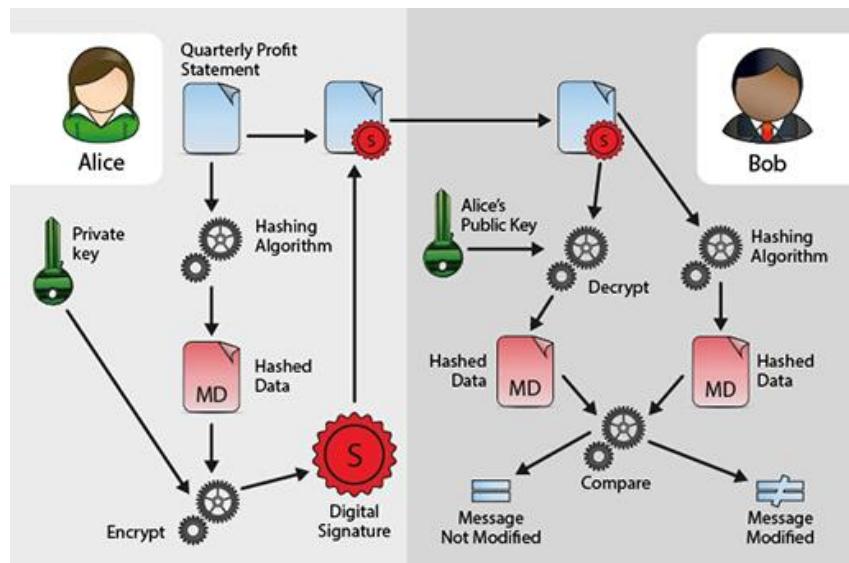
- To “**solve**” puzzle i find x_i s.t $H(x_{i-1}, x_i) = 0^d$
- The longest chain defines “**next puzzle**”
- The name in block x_i “**gets**” coin i.



TheoryCoin (TC): How to transfer money



- (Digital) Signatures
 - Only you can sign
 - Everyone can verify
 - You cannot deny



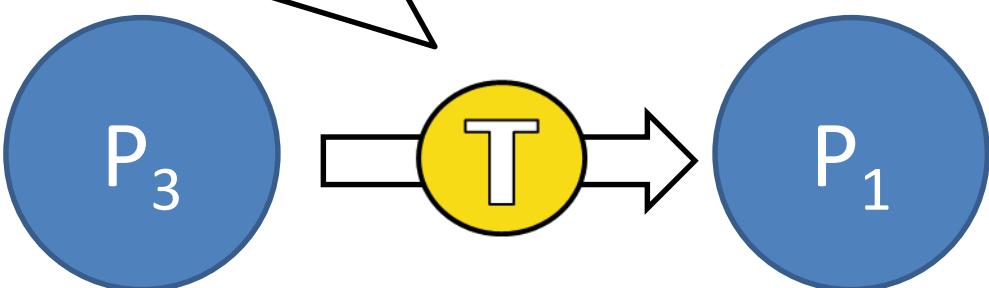
ECDSA

CRYPTOGRAPHY

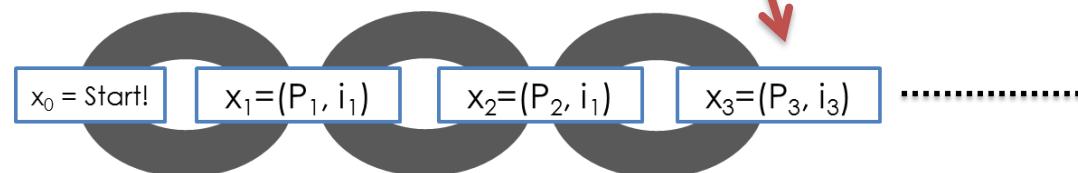
TheoryCoin (TC): How to transfer money



$m = "P_3 \text{ gives coin 3 to } P_1"$
 $s = \text{Sig}(sk_3, m)$



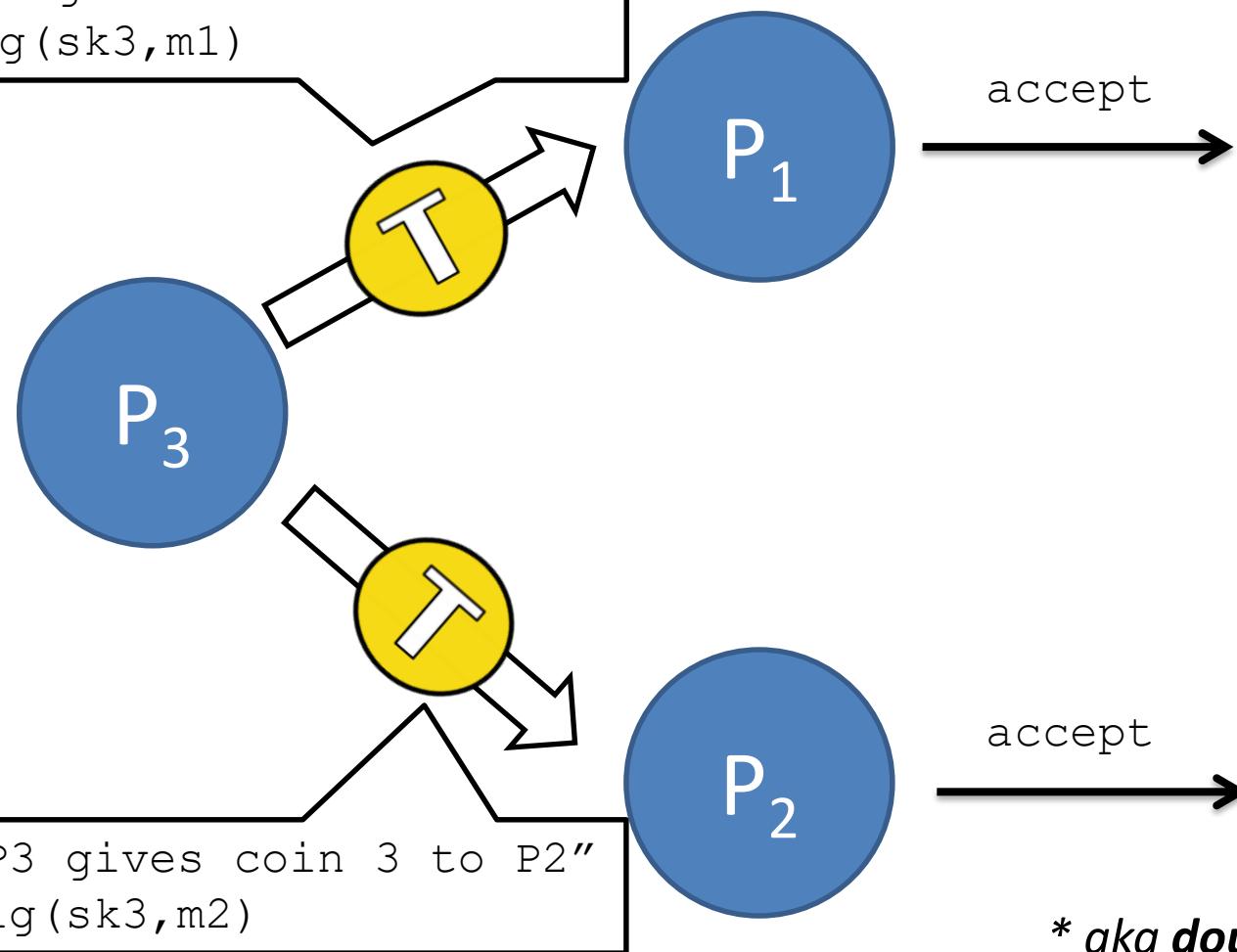
If
 $\text{Ver}(pk_3, m, s) = \text{accept}$
and
 P_3 owns coin 3
then
return accept



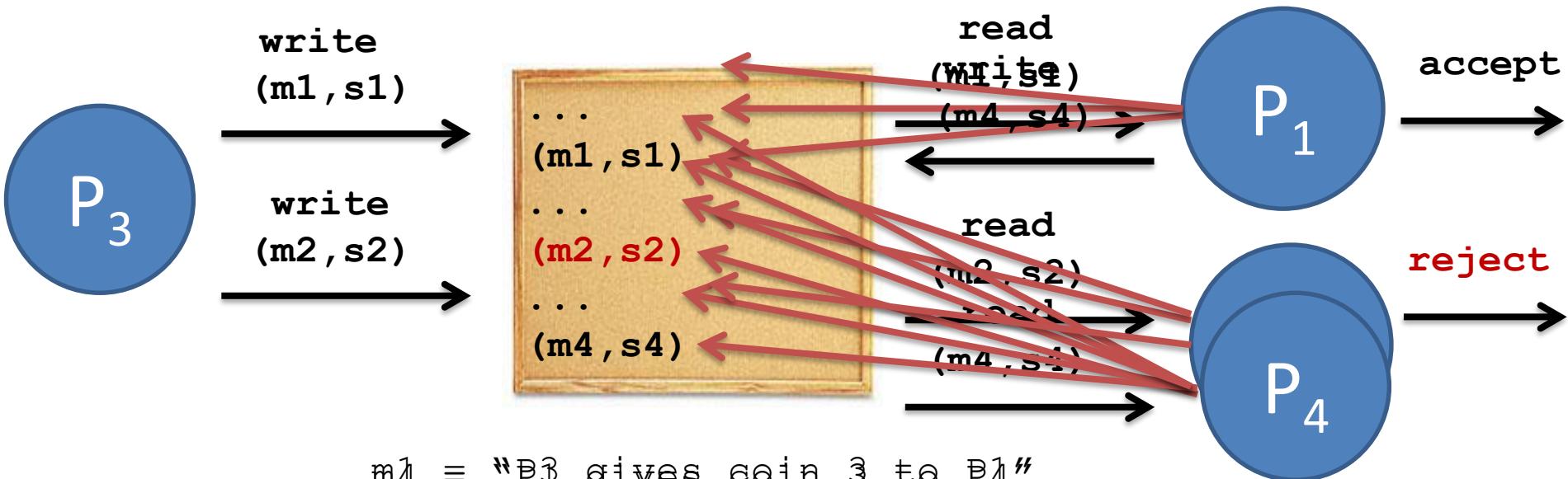
TheoryCoin (TC): How to transfer money



$m1 = \text{"P3 gives coin 3 to P1"}$
 $s1 = \text{Sig}(\text{sk}_3, m1)$



TheoryCoin (TC): How to transfer money



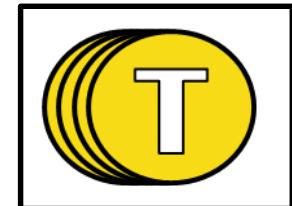
$m_4 \equiv \text{"P3 gives coin 3 to P4"}$

$s_4 \equiv \text{Sig}(\text{sk}_3, m_4)$

$m_2 = \text{"P3 gives coin 3 to P2"}$

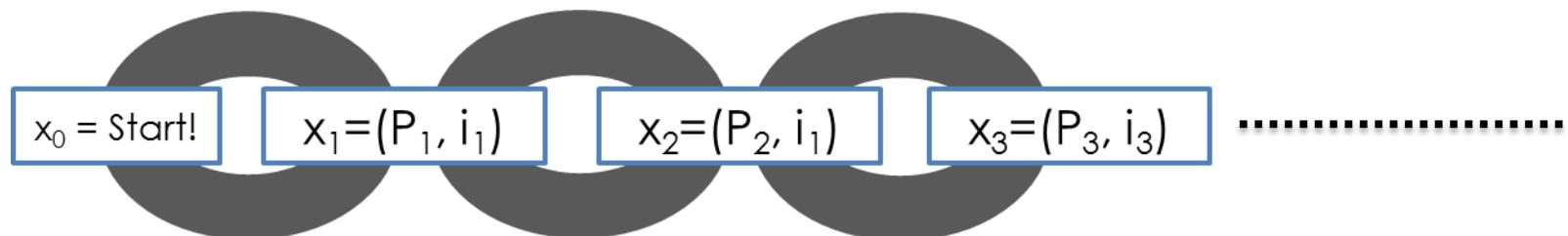
$s_2 = \text{Sig}(\text{sk}_3, m_2)$

TheoryCoin (TC): How to store money



Main Idea:

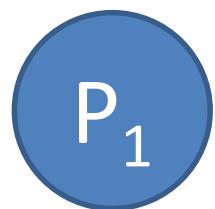
Record **transfers** in the **blockchain**



TheoryCoin (TC): How to store money



```
SolvePuzzle(L) {  
    repeat{  
        R = my_name || i++  
        T = H(L,R)  
    }while(T ≠ 0d)  
    return R  
}
```



(m, s)

```
SolvePuzzle(L) {  
    repeat{  
        R = my_name || i++  
        T = H(L,R)  
    }while(T ≠ 0d)  
    return R  
}
```

$x_0 = \text{Start!}$

$x_1 = (P_1, i_1)$

$x_2 = (P_2, i_1)$

$x_3 = (P_3, i_3)$

$x_4 = (P_4, (m,s), i_4)$

```
SolvePuzzle(L) {  
    repeat{  
        R = my_name  
        T = H(L,R)  
    }while(T ≠ 0d)  
    return R  
}
```

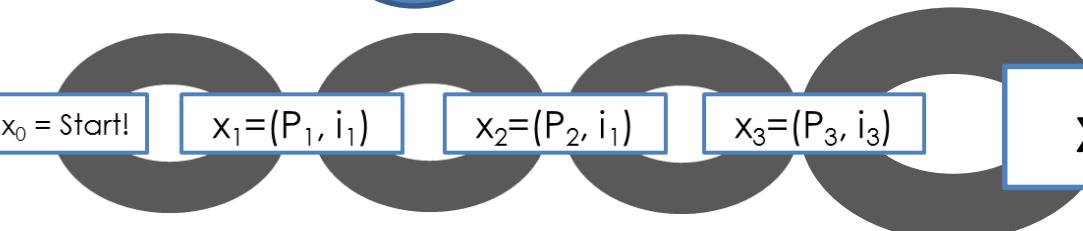
```
SolvePuzzle(L) {  
    repeat{  
        R = my_name || i++  
        T = H(L,R)  
    }while(T ≠ 0d)  
    return R  
}
```

```
SolvePuzzle(L, . . .) {  
  
    repeat{  
  
        R = my_name || (m, s) || i++  
        T = H(L, R)  
  
    }while(T ≠ 0d)  
  
    return R  
}
```

}



(m, s)



diff(, )

How is money created in Bitcoin?

- New block **every ~10 mins**
 - \mathbf{d} adjusted every ~2000 blocks
- $H = \text{2-SHA2}$
- Initial reward: **50 BTC**
 - Halved every ~4 years (now **25 BTC**)

$$L \in \{0,1\}^*$$

$$R \in \{0,1\}^*$$



$$T \in \{0,1\}^d$$

diff(, )

How is money transferred in Bitcoin?

Example: P1 wants to give 60 to P2

... gives 50 to P1

... gives 25 to P1



P1 gives 60 to P2

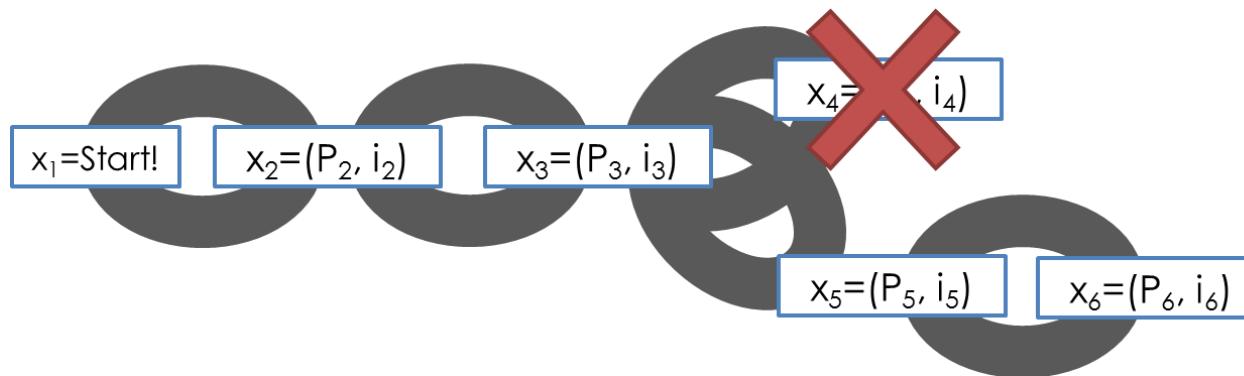
P1 gives 14 to P1

Transaction fee 1



How is money stored in Bitcoin?

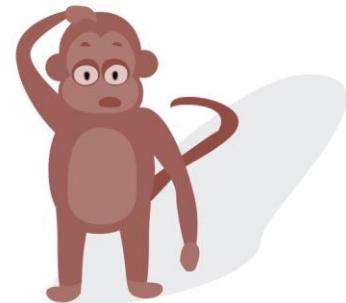
- Transaction in **orphaned blocks** are invalid
 - Wait 6 blocks (~1 hour) before accepting transaction.
 - Checkpoints to prevent complete history rollback.



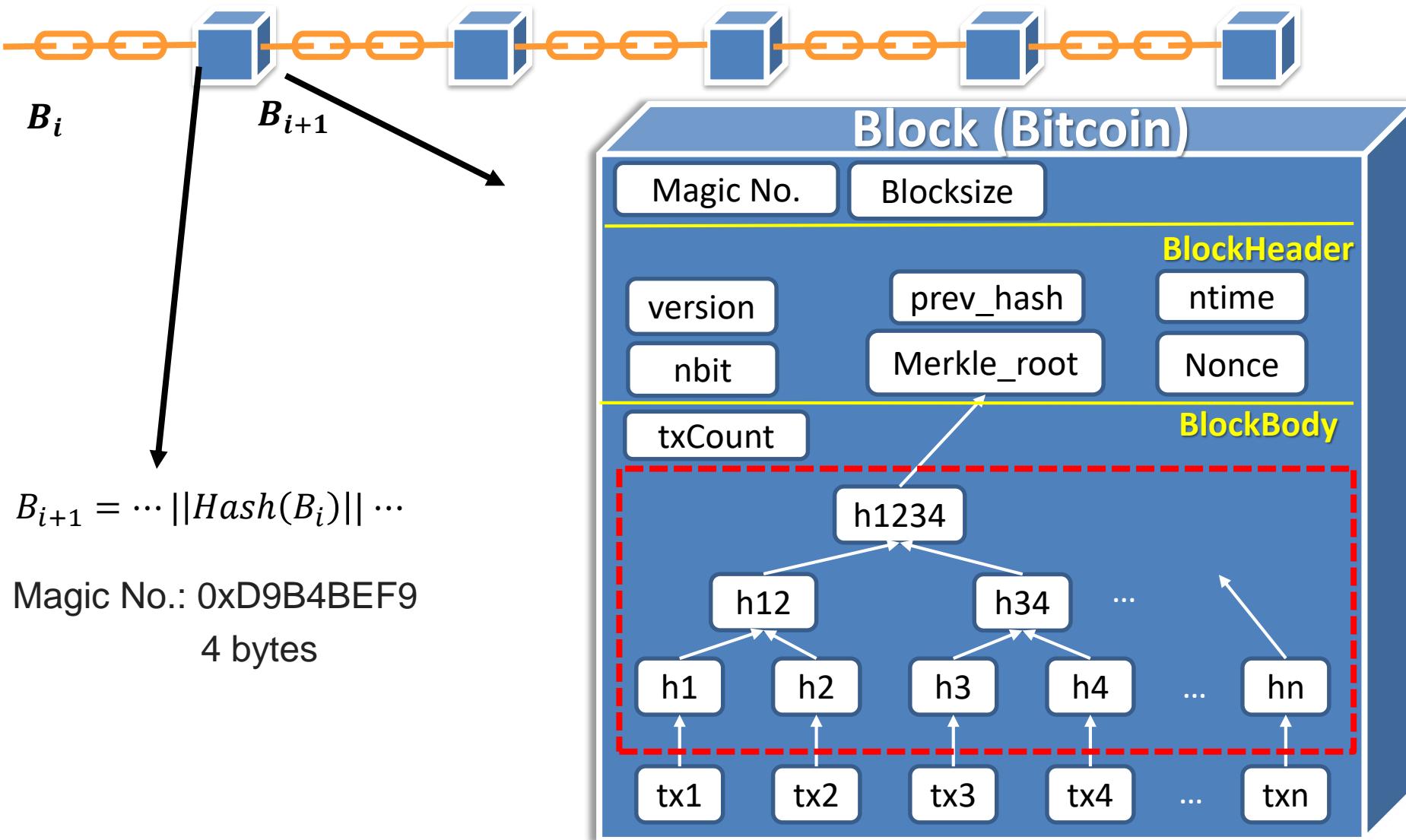
- **All transaction** are stored in the blockchain
 - (Currently ~xx GB)

Problems

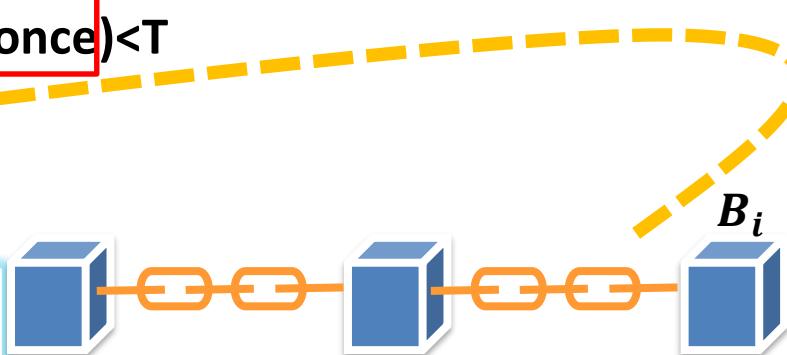
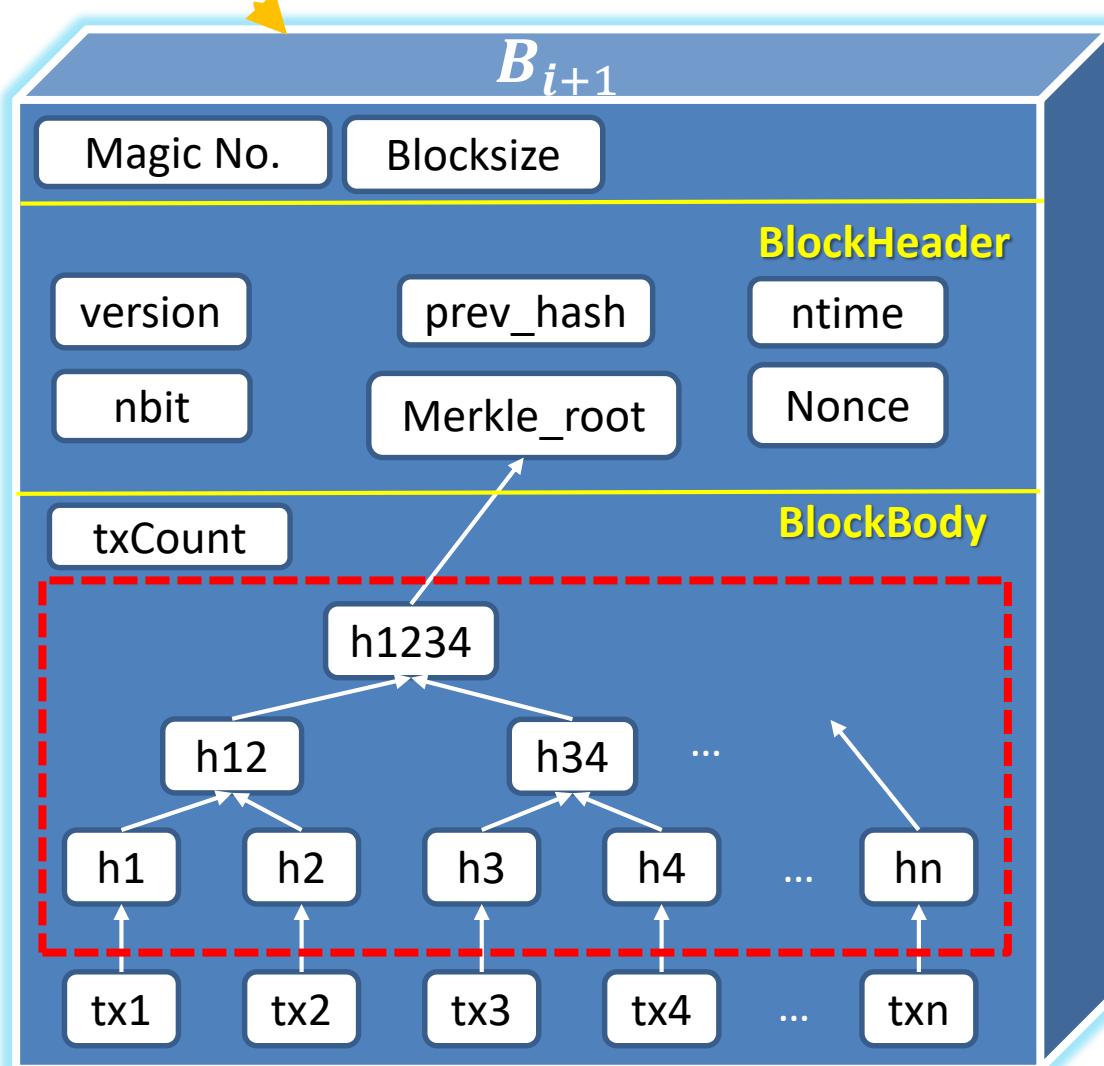
- Anonymity
 - Every transaction ever made is **recorded forever**
 - Use **new identity** for each transaction
- Users? (*and their devices*)
 - 1HKywxiL4JziqXrzLKhmB6a74ma6kxbSDj
- Mining pools
 - **Solving puzzles (mining) is hard!**
 - Miners join pools and share work/reward
 - **How to optimally split work?**
 - **Mechanism design?**
 - rational miner?
 - how to allocate reward?



Data Structure of BitCoin



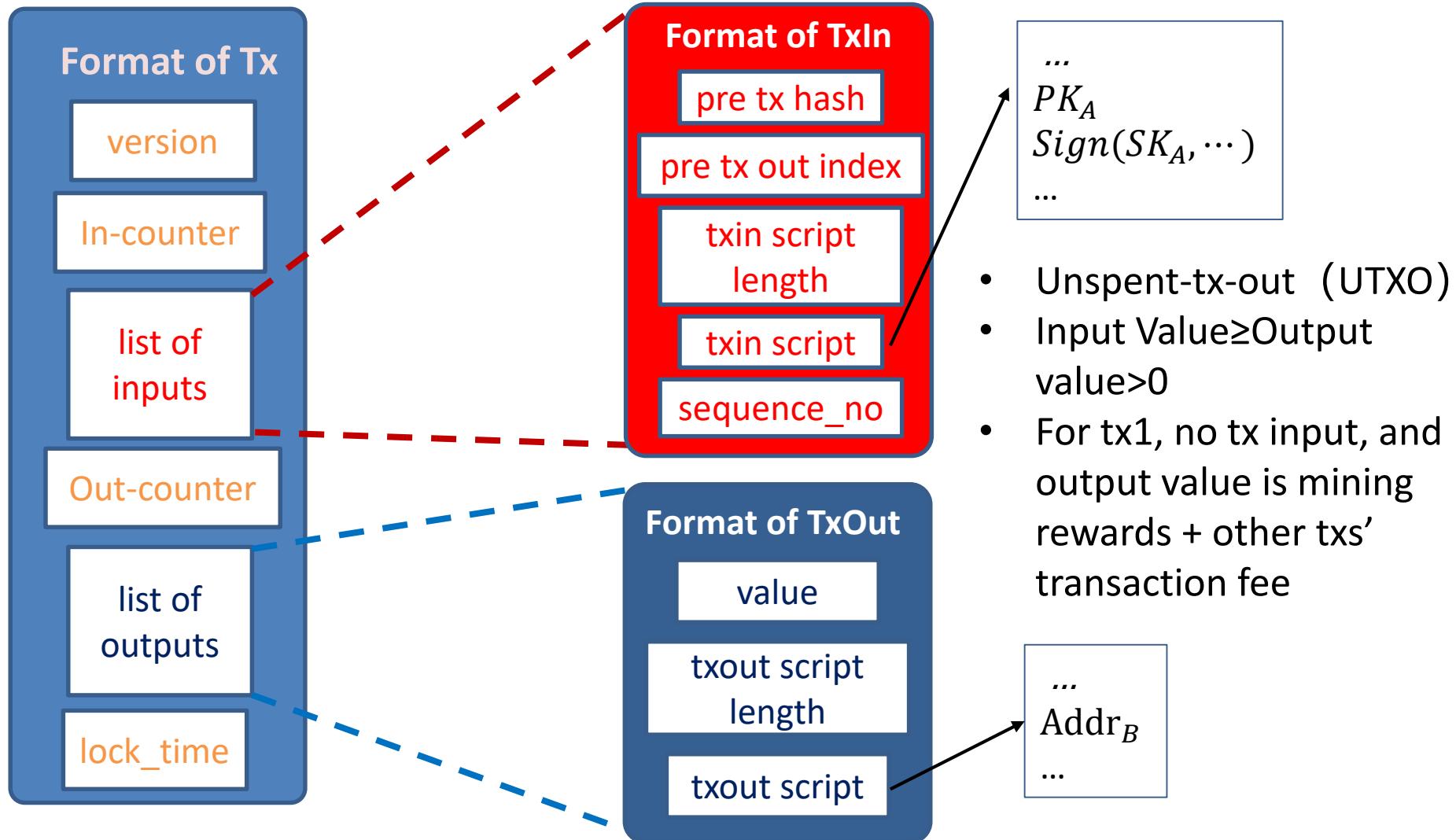
Hash(version,pre_hash,ntime,nbit,merkle_root,**Nonce**)<T



- Pow
- $nbit \rightarrow T$, bitcoint 10 min
- Miner will get 50, and half every 4 years
- Except tx1, transcation fee for other tx ≥ 0

Mining in
BitCoin

BitCoin Transactions



Disadvantages of BitCoin

- Slow
 - Generate one block / 10 minutes
 - Each block 1M (2M)
 - 7 transactions/second
 - VISA average value is 1,667^[1]



[1] <https://altcointoday.com/bitcoin-ethereum-vs-visa-paypal-transactions-per-second/>

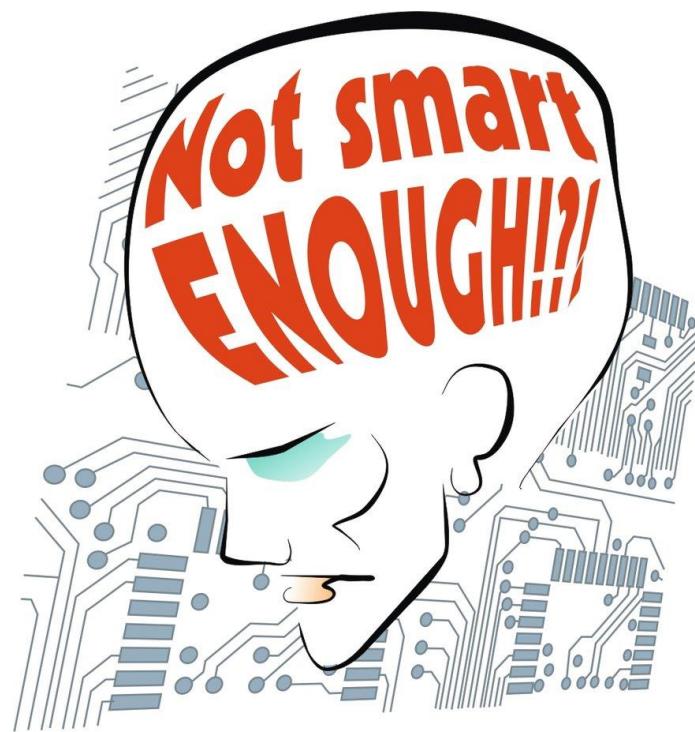
Disadvantages of BitCoin

- Not sufficient in Privacy Preservation
 - Transactions are publicly open
 - Input Address, Output Address, Transaction Content
 - No link between real identity and address, but they can be possibly linked

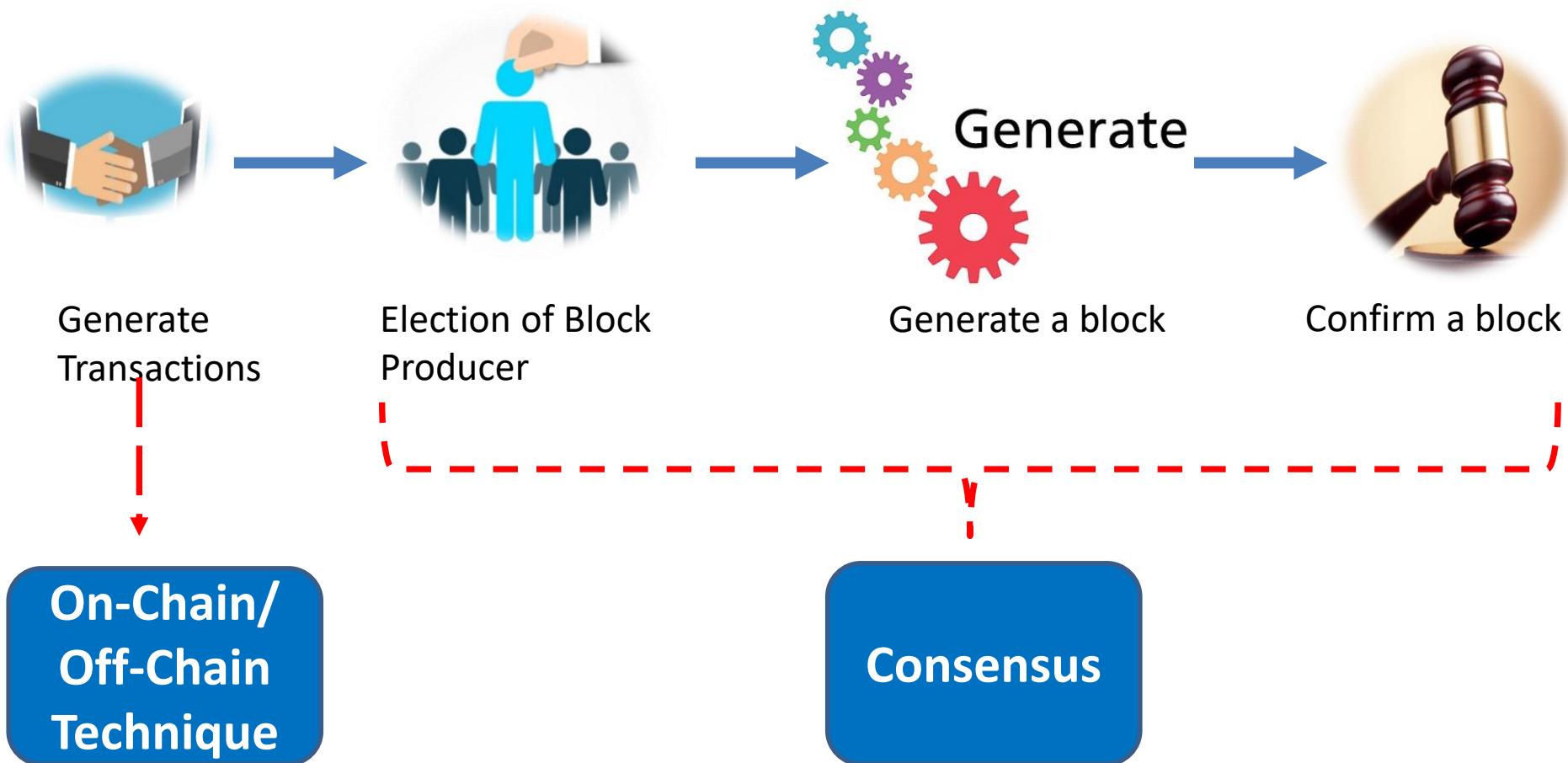


Disadvantages of BitCoin

- Not smart enough
 - Block only includes the transactions
 - Cannot run complex program



How to deal with the speed challenge?

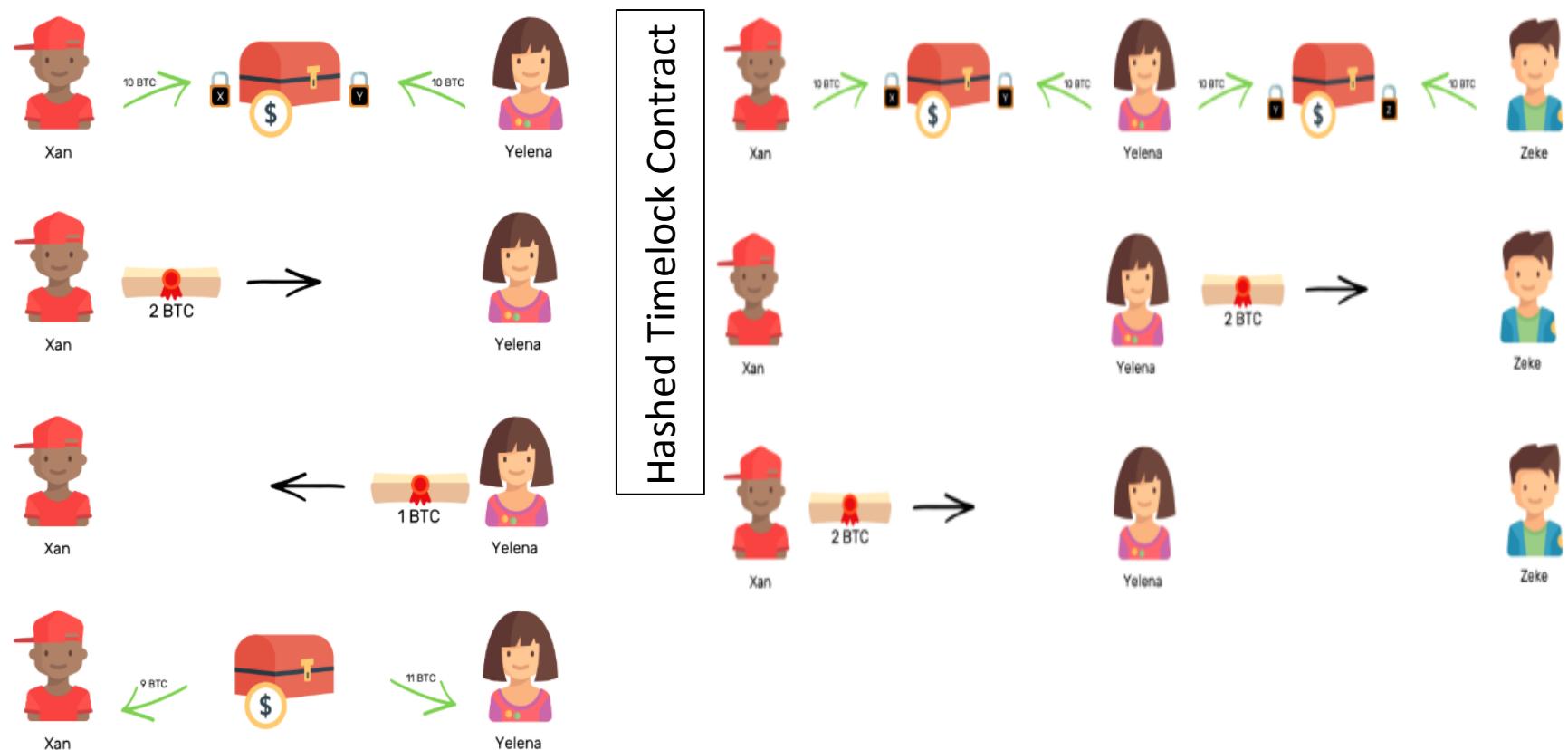


How to deal with the speed challenge?

Off-Chain Technique

- Lightning Network

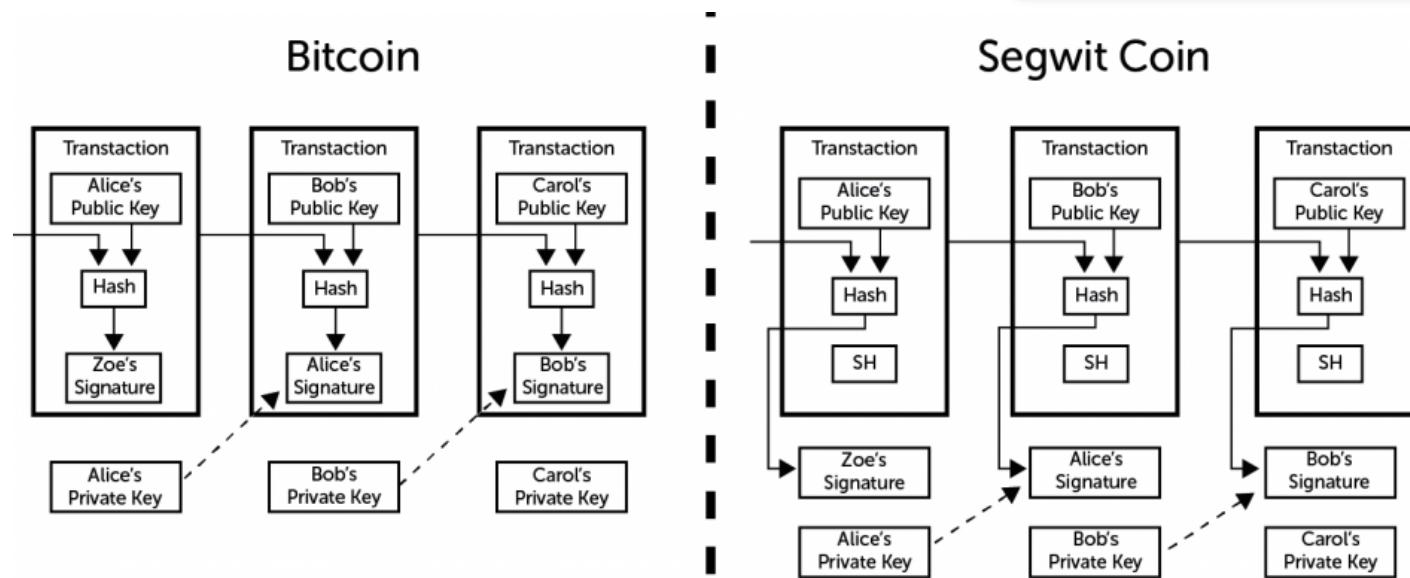
Recoverable Sequence Maturity Contract



How to deal with the speed challenge?

On-Chain Technique

- Increase the size of block
 - 2M, 4M, 8M
- Reduce the size of transactions
 - SegWit Coin



How to deal with the speed challenge?

Consensus

- Consensus
 - dBFT (Delegated Byzantine Fault Tolerance)
 - Reduce the Candidate Number of Block Producers
 - dPoS (Delegated Proof of Stake)
 - Reduce the Candidate Number of Block Producers
 - Sharding
 - Increase the Number of Block Producers
 - directed acyclic graph (DAG)
 - Increase the Number of Block Producers



How to deal with the Privacy challenge?



Ring Signature
+ Commitment
+ Encrypted Address



zerocoin



CASH

Zero Knowledge Proof
+ Commitment

Pedersen Commitment

Setup:

p, q, g, y

Commit:

$c = g^r y^m \bmod p$

Open:

$c = g^r y^{m'} \bmod p$

Ring Signature: Protect the initiator of the transactions + link them

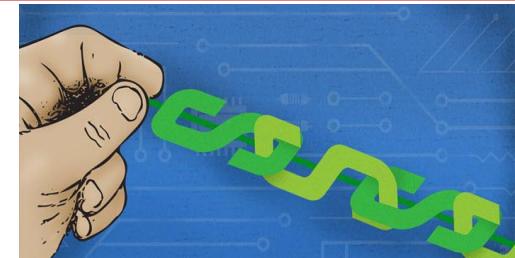
Commitment: Protect the content of transactions

Encrypted Address: The address can be received by the transaction's receiver; the privacy key related to the address can be received by the transaction's receiver.

$$A = g^a, B = g^b$$

$$g^r, g^{H(A^r)} \cdot B$$

Smart Contract



- A **smart contract** is a computer protocol intended to digitally facilitate, verify, or enforce the negotiation or performance of a contract. Smart contracts allow the performance of credible transactions without third parties. These transactions are trackable and irreversible.
- In 2018, a US Senate report said: "While smart contracts might sound new, the concept is rooted in basic contract law."
- With smart contracts, a program enforces the contract built into the code.



Smart Contract

Ethereum Account Type (Just like User Account)



Address



Balance



Code
State



0x16E0022b17B...

0 Ether

```
contract Counter {  
    uint counter;  
  
    function Counter() public {  
        counter = 0;  
    }  
    function count() public {  
        counter = counter + 1;  
    }  
}
```

Thank
you



CS 6355/4355: Cryptanalysis and Database Security

Topic 1: Foundations of Security and Classical
Encryption Techniques

Lecturer: Rongxing LU

Email: RLU1@unb.ca Office: GE 114

Website: <http://www.cs.unb.ca/~rlu1/>

Faculty of Computer Science, University of New Brunswick

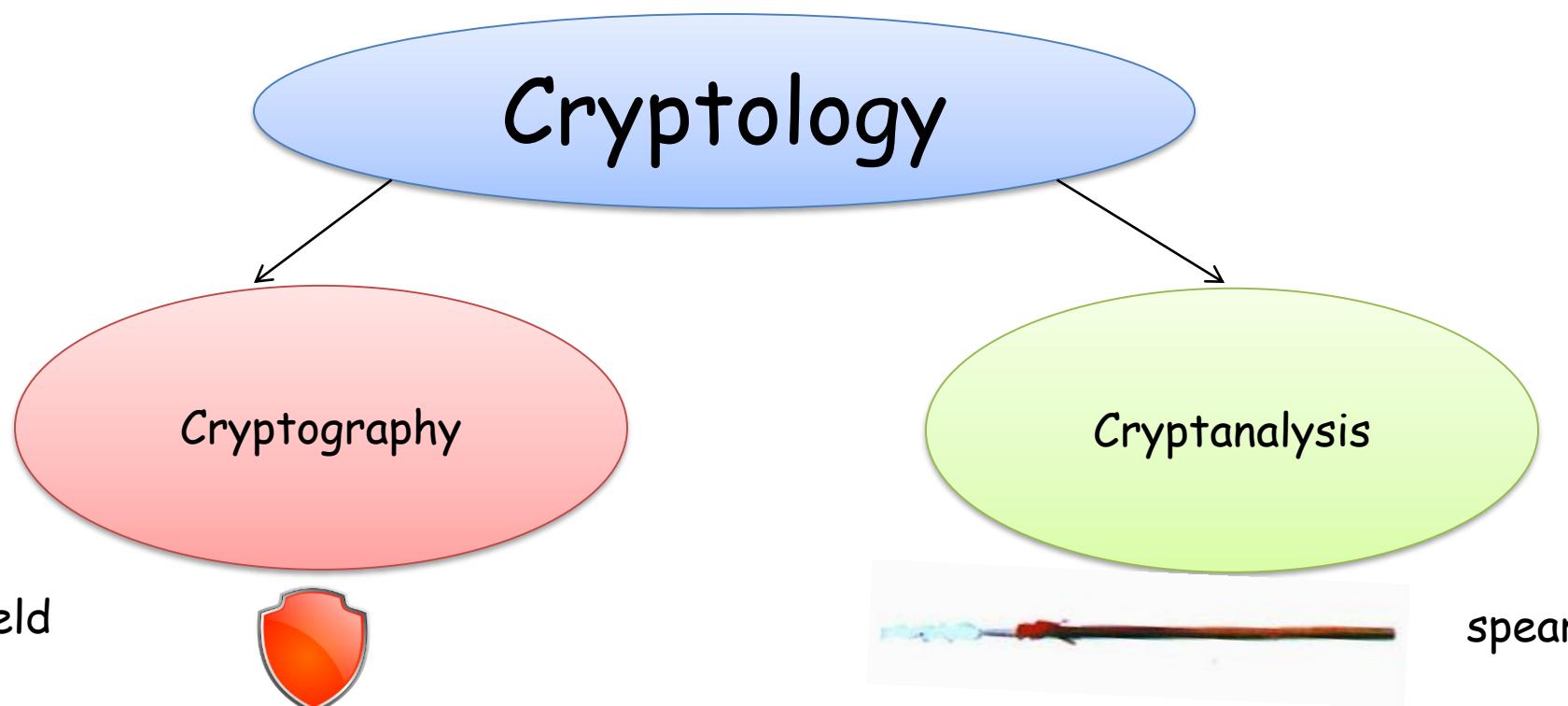
Introduction

- What is Cryptology?
 - The term **cryptology**, is derived from the Greek words “kryptÓs”, standing for “hidden”, and “lÓgos”, standing for “word”. Consequently, the meaning of the term cryptology is best paraphrased as “hidden word”. This paraphrase refers to the original intent of cryptology, namely to hide the meaning of specific words and to protect their confidentiality accordingly.



Cryptology = Cryptography + Cryptanalysis

- Cryptology refers to the mathematical science and field of study that comprises both cryptography and cryptanalysis.

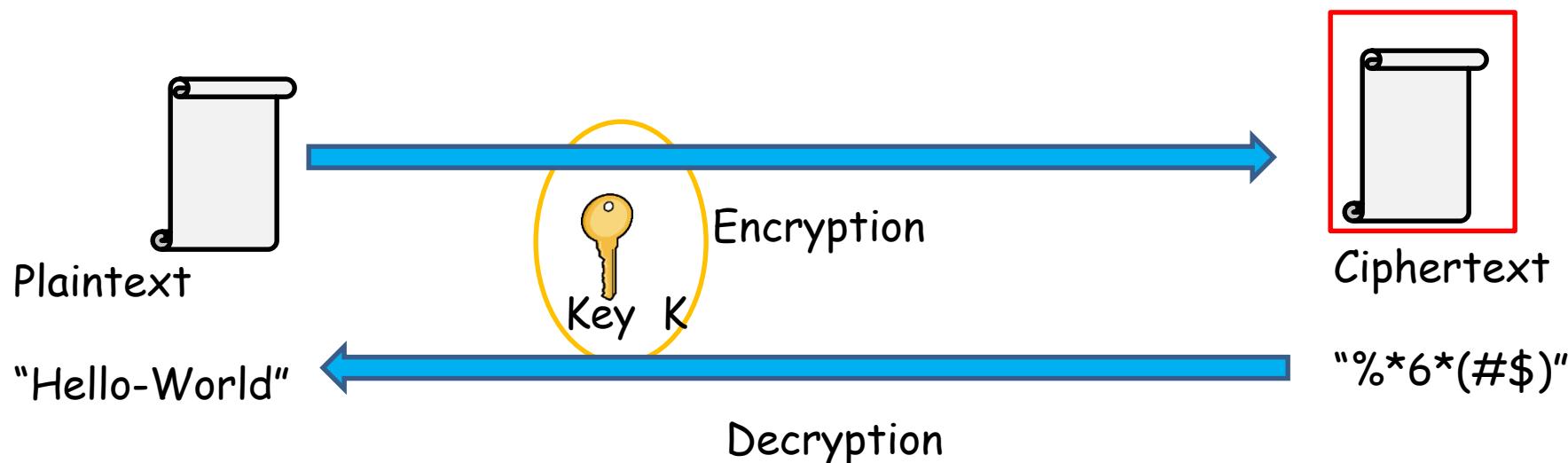




shield

Cryptography

- The science of “Secret” writing
 - A cipher is a function which transforms a plaintext message into a ciphertext (cryptogram) by the process of encipherment
 - Plaintext is recovered from the ciphertext by the process of deciphering



Cryptanalysis

- The science and study of breaking ciphers, i.e., the process of determining the plaintext message from the ciphertext

Ciphertext



Plaintext



spear

Without knowing Key K



Cryptographic algorithms and protocols can be grouped into four main areas:

- Symmetric encryption
 - Used to conceal the contents of blocks or streams of data of **any size**, including messages, files, encryption keys, and passwords
- Asymmetric (public key) encryption
 - Used to conceal **small** blocks of data, such as encryption keys and hash function values, which are used in digital signatures
- Data integrity algorithms
 - Used to protect blocks of data, such as messages, from alteration
- Authentication protocols
 - Schemes based on the use of cryptographic algorithms designed to authenticate the identity of entities

Cryptographic algorithms and protocols can be applied to Network Security

- The field of network and Internet security consists of:
 - measures to deter, prevent, detect, and correct security violations that involve the transmission of information



Cryptographic algorithms and protocols can be applied to Computer Security

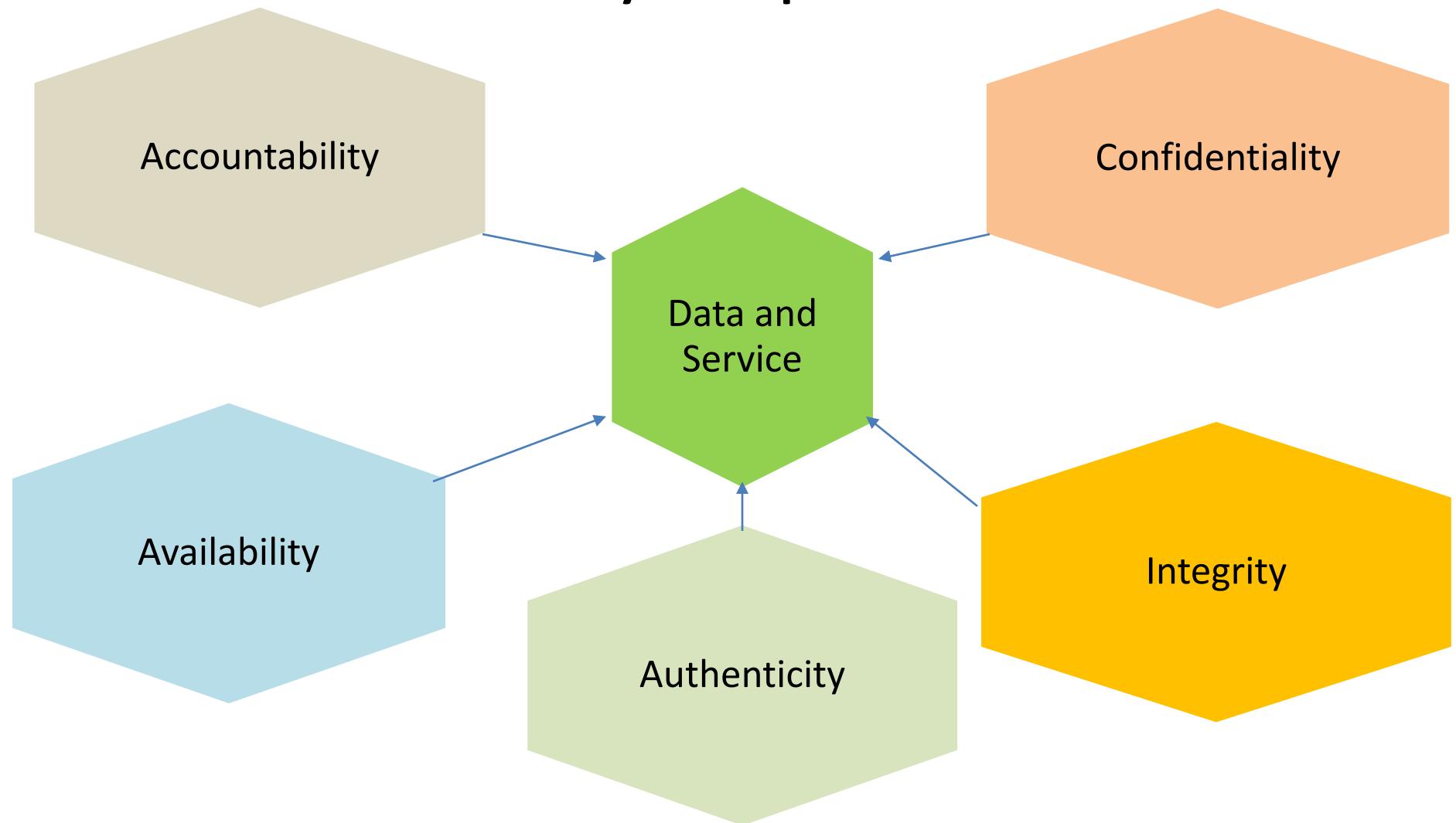
- The NIST Computer Security Handbook defines the term computer security as:
 - “the protection afforded to an automated information system in order to attain the applicable objectives of preserving the integrity, availability and confidentiality of information system resources” (includes hardware, software, firmware, information/data, and telecommunications)



Computer Security Objectives

- Confidentiality
 - Data confidentiality
 - Assures that private or confidential information is not made available or disclosed to unauthorized individuals
 - Privacy
 - Assures that individuals control or influence what information related to them may be collected and stored and by whom and to whom that information may be disclosed
- Integrity
 - Data integrity
 - Assures that information and programs are changed only in a specified and authorized manner
 - System integrity
 - Assures that a system performs its intended function in an unimpaired manner, free from deliberate or inadvertent unauthorized manipulation of the system
- Availability
 - Assures that systems work promptly and service is not denied to authorized users

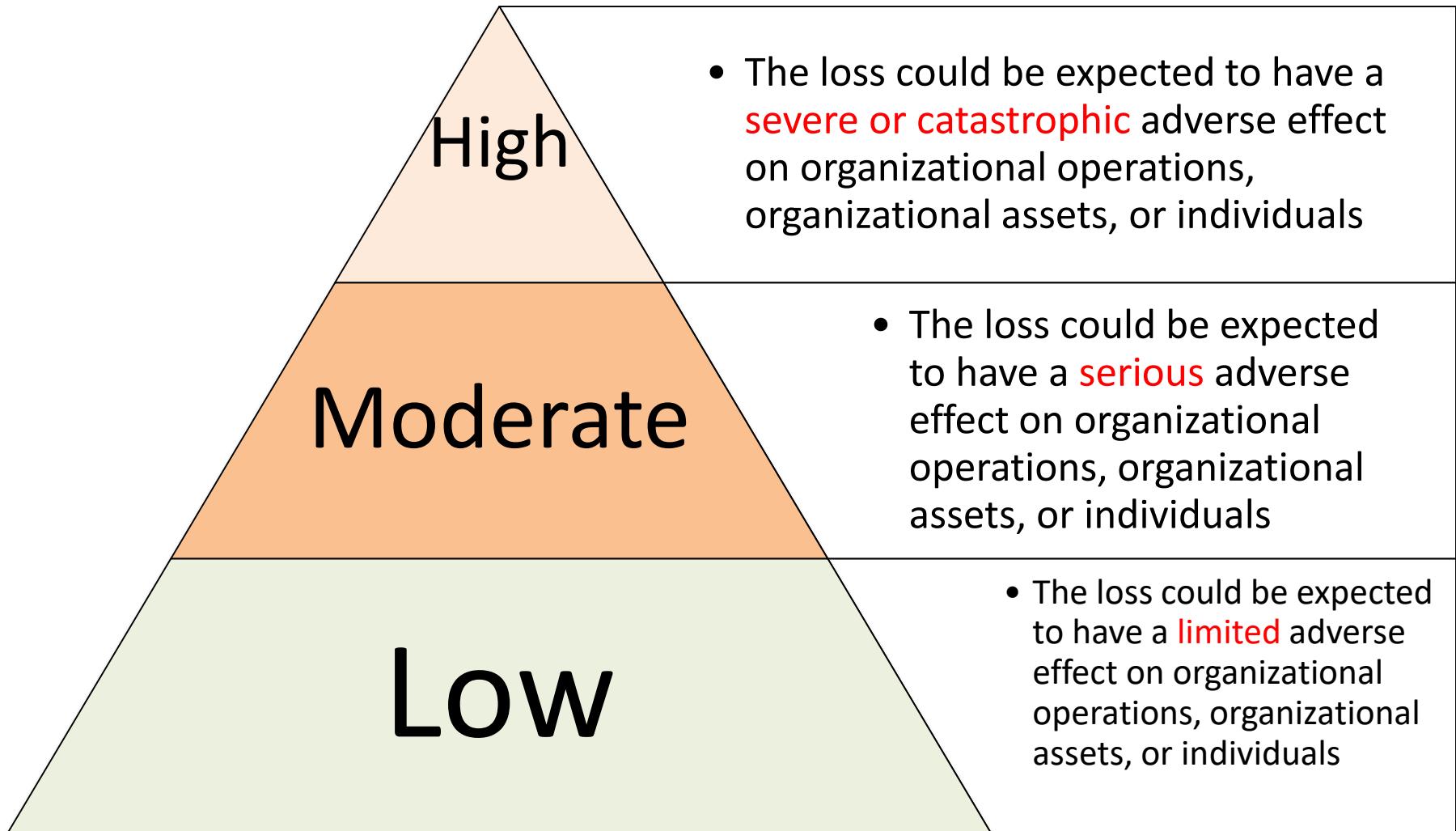
Essential Network and Computer Security Requirement



Authenticity & Accountability

- Authenticity:
 - The property of being genuine and being able to be verified and trusted; confidence in the validity of a transmission, a message, or a message originator.
- Accountability:
 - The security goal that generates the requirement for actions of an entity to be traced uniquely to that entity.
 - Nonrepudiation, intrusion detection and prevention, legal action

Breach of Security Levels of Impact



Computer Security Challenges

- Security is not simple
- Potential attacks on the security features need to be considered
- Procedures used to provide particular services are often counter-intuitive
- It is necessary to decide where to use the various security mechanisms
- Requires constant monitoring
- Is too often an afterthought
- Security mechanisms typically involve more than a particular algorithm or protocol
- Security is essentially a battle of wits between a perpetrator and the designer
- Little benefit from security investment is perceived until a security failure occurs
- Strong security is often viewed as an impediment to efficient and user-friendly operation



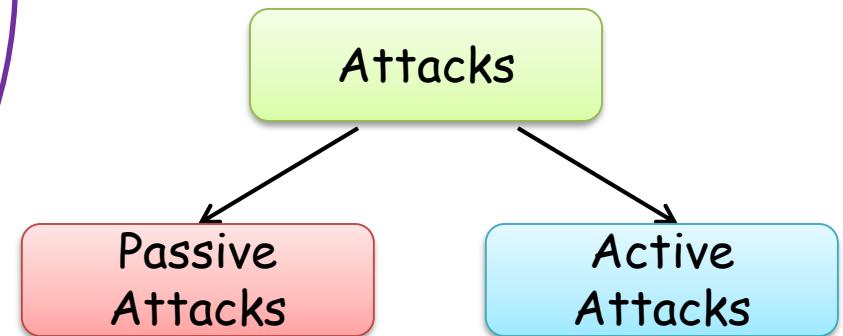
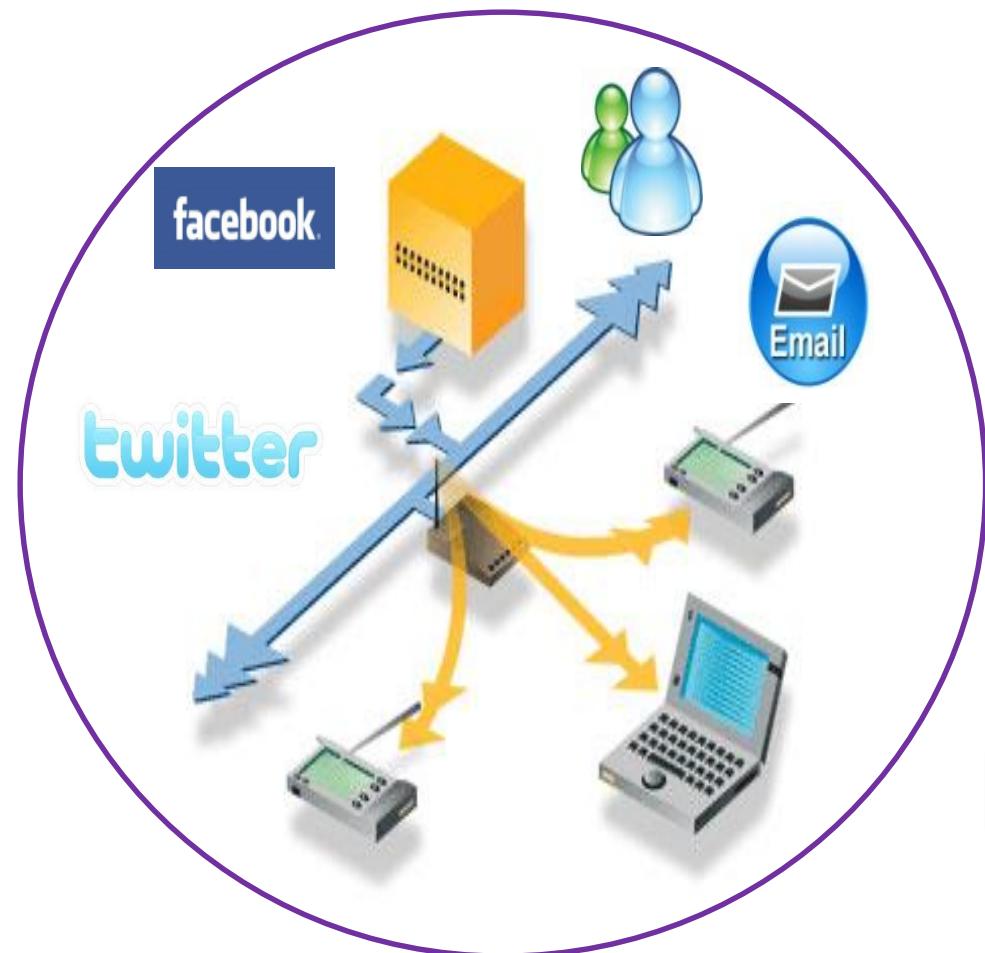
OSI Security Architecture

- Security attack
 - Any action that compromises the security of information owned by an organization
- Security mechanism
 - A process (or a device incorporating such a process) that is designed to detect, prevent, or recover from a security attack
- Security service
 - A processing or communication service that enhances the security of the data processing systems and the information transfers of an organization
 - Intended to counter security attacks, and they make use of one or more security mechanisms to provide the service

Threat & Attack

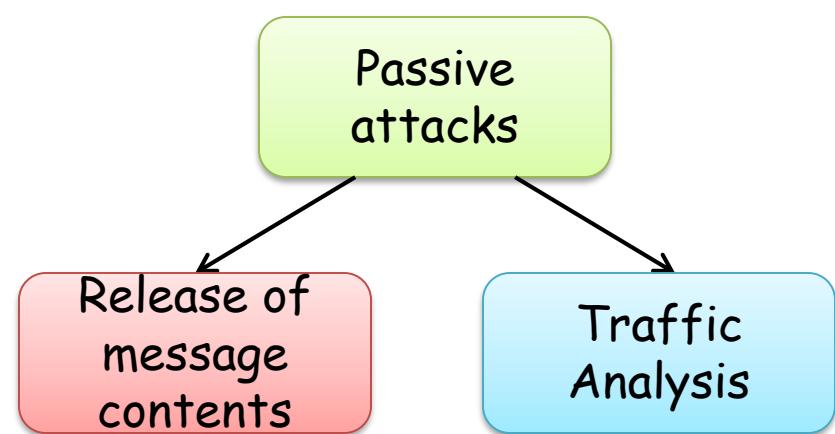
- Definitions taken from RFC 4949, Internet Security Glossary.
- **Threat:**
 - A potential for violation of security, which exists when there is a circumstance, capability, action, or event that could breach security and cause harm. That is, a threat is a possible danger that might exploit a vulnerability.
- **Attack:**
 - An assault on system security that derives from an intelligent threat; that is, an intelligent act that is a deliberate attempt (especially in the sense of a method or technique) to evade security services and violate the security policy of a system.

Security Attacks



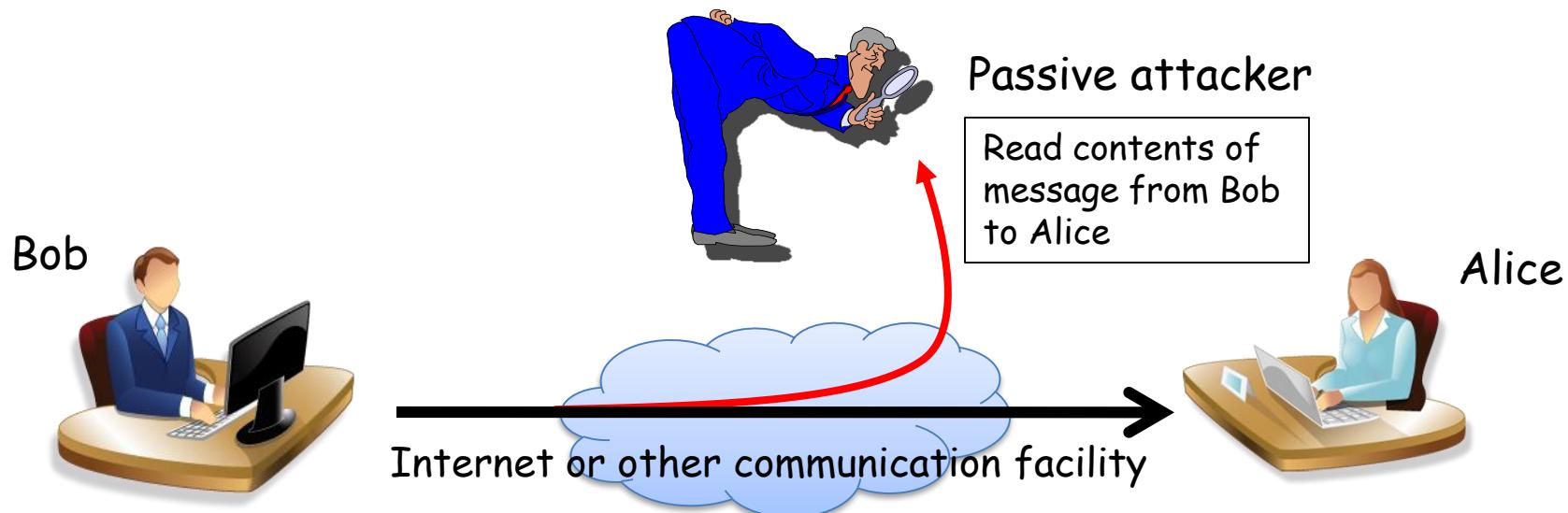
Passive Attacks

- **Passive attacks** are in the nature of eavesdropping on, or monitoring of, transmissions. The goal of the opponent is to obtain information that is being transmitted.
- Two types of passive attacks are **release of message contents** and **traffic analysis**.



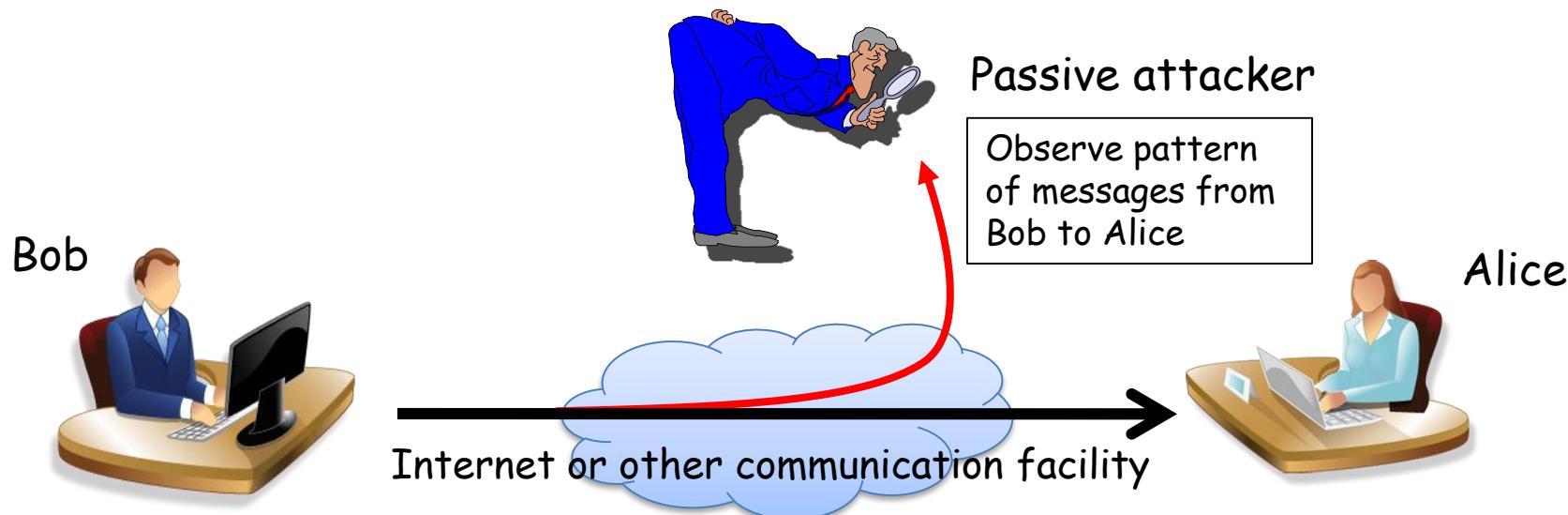
Release of message contents

- The release of message contents: A telephone conversation, an electronic mail message, and a transferred file may contain sensitive or confidential information.
- Our goal is to prevent a passive attacker from learning the contents of these transmissions.



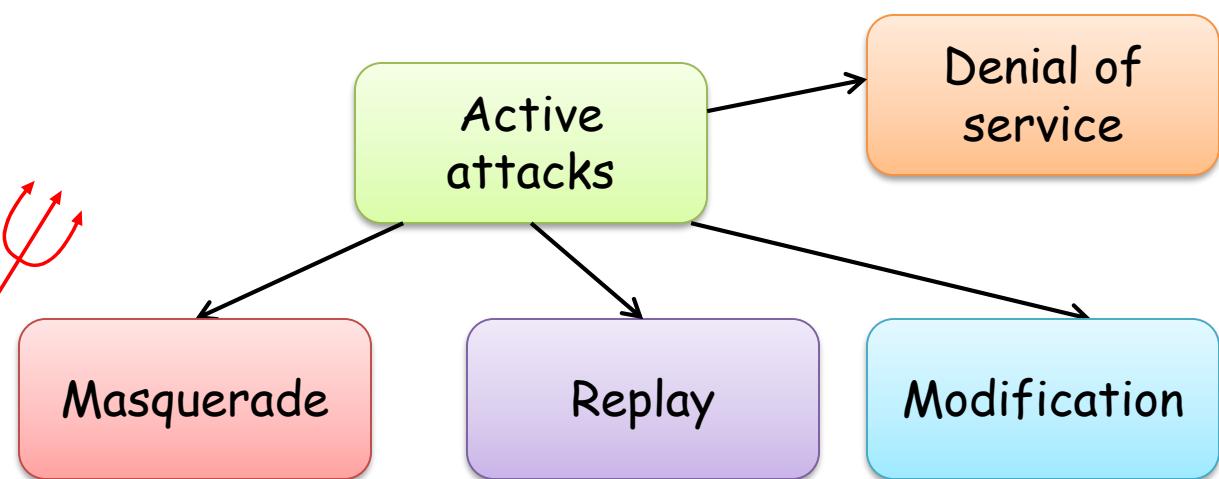
Traffic Analysis

- **Traffic Analysis:** If we had encryption protection in place, a passive attacker might still be able to observe the pattern of these messages. The attacker could determine the location and identity of communicating hosts and could observe the frequency and length of messages being exchanged, where the information might be useful in guessing the nature of the communication that was taking place.



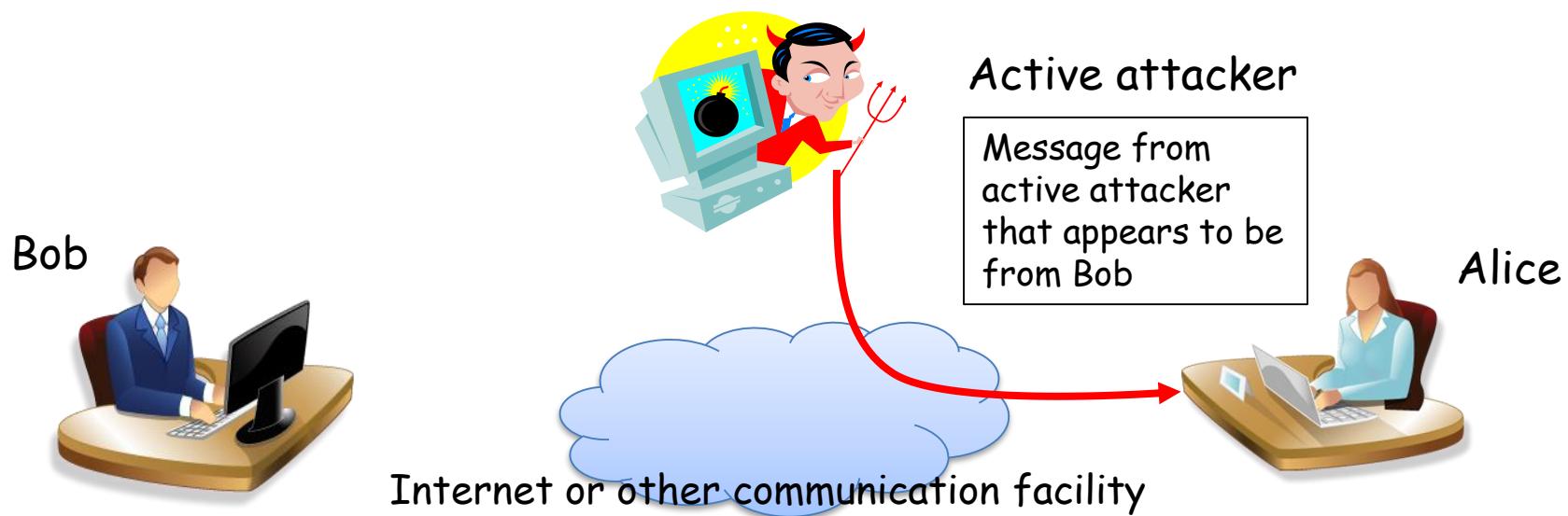
Active Attacks

- **Active attacks** involve some modification of the data stream or the creation of a false stream and can be subdivided into four categories: masquerade, replay, modification of messages, and denial of service.



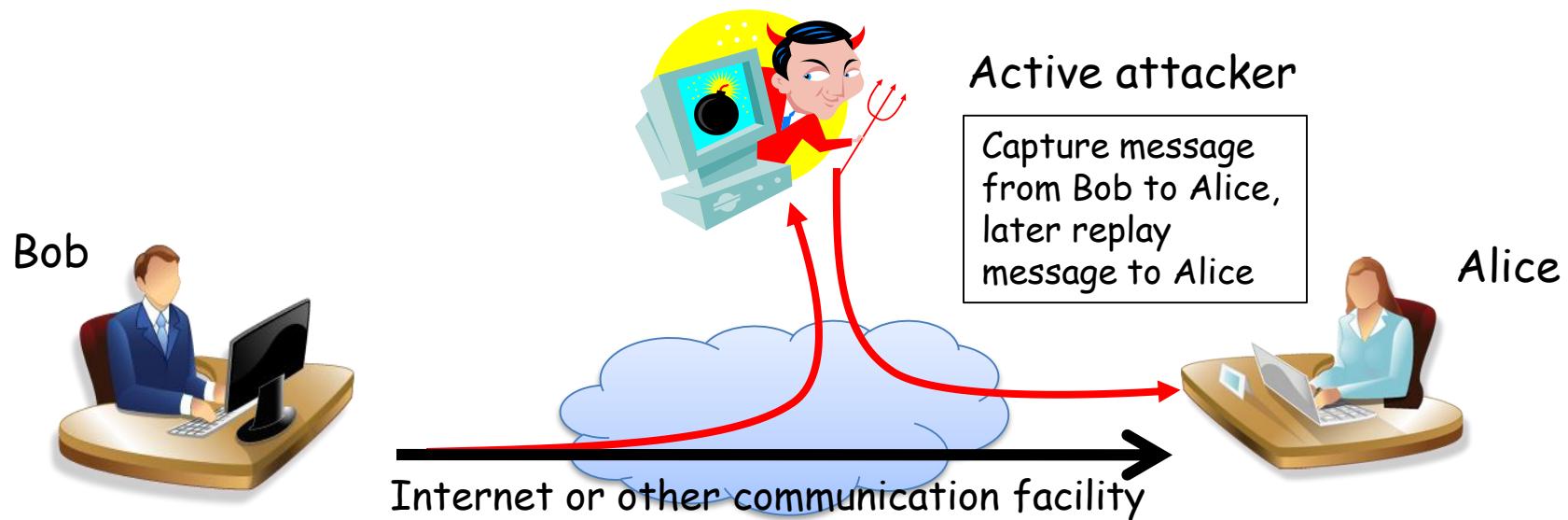
Masquerade

- **Masquerade:** takes place when one entity pretends to be a different entity



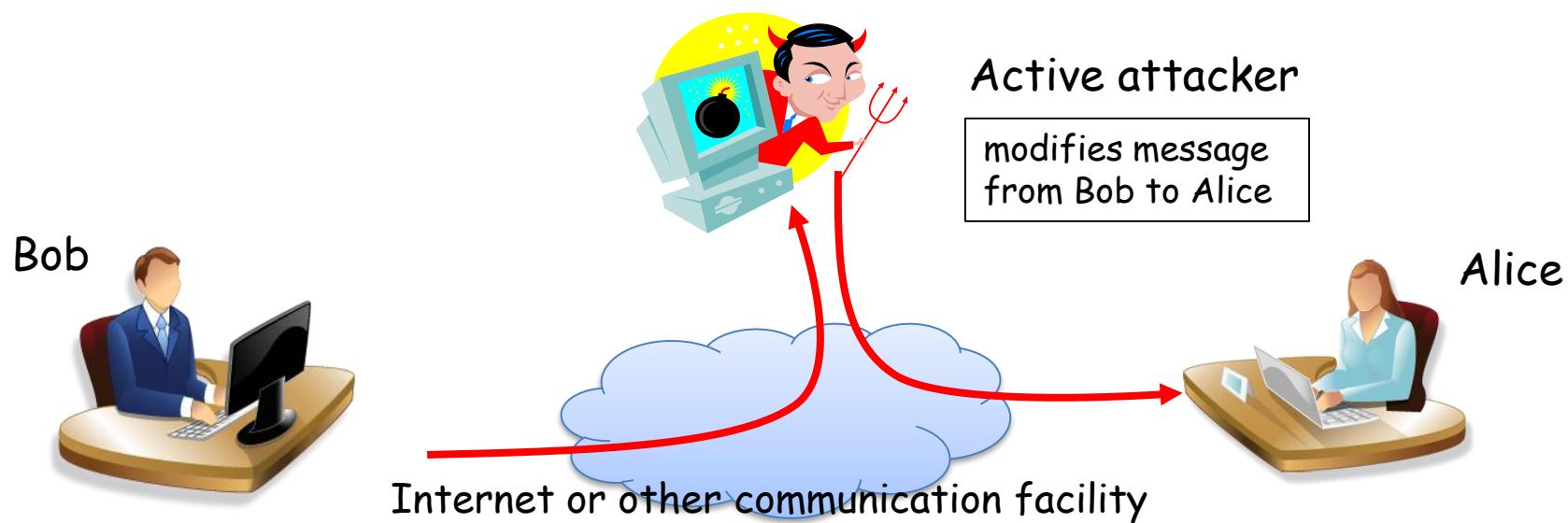
Replay

- **Replay:** involves the passive capture of a data unit and its subsequent retransmission to produce an unauthorized effect



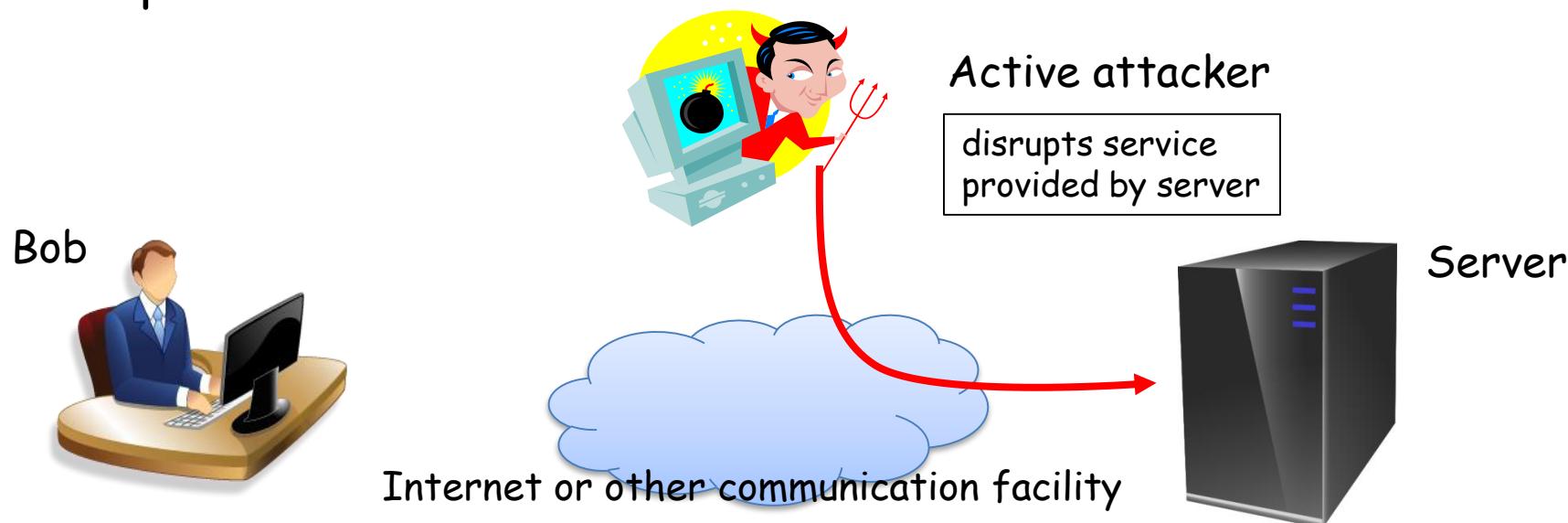
Modification

- **Modification:** means that some portion of a legitimate message is altered, or that messages are delayed or reordered, to produce an unauthorized effect. For example, a message meaning "Allow Alice to read confidential file accounts" is modified to mean "Allow Alice to delete confidential file accounts."



Denial of Service

- **Denial of Service (DoS):** prevents the normal use or management of communications facilities.
 - DoS attack may have a specific target; for example, an entity may suppress all messages directed to a particular destination (e.g., the security audit service).
 - Another form of DoS is the disruption of an entire network, either by disabling the network or by overloading it with messages so as to degrade performance.



Comparisons



- **Passive Attacks**

- Passive attacks are very difficult to detect because they do not involve any alteration of the data.
- Typically, the message traffic is sent and received in an apparently normal fashion and neither the sender nor receiver is aware that a third party has read the messages or observed the traffic pattern.
- However, it is feasible to prevent the success of these attacks, usually by means of encryption. Thus, the emphasis in dealing with passive attacks is on **prevention** rather than detection.



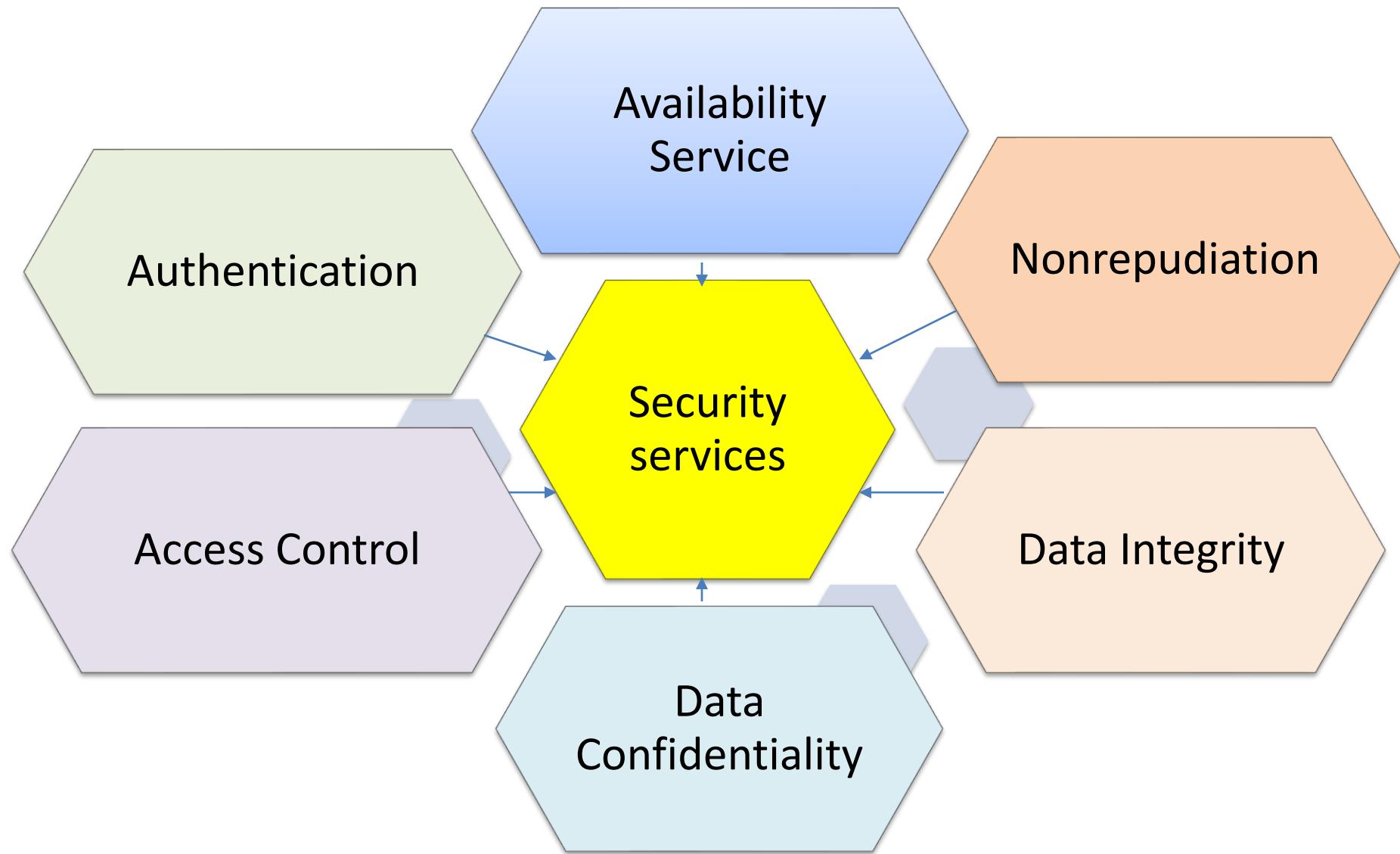
- **Active Attacks**

- Active attacks present the opposite characteristics of passive attacks. Whereas passive attacks are difficult to detect, measures are available to prevent their success.
- On the other hand, it is quite difficult to prevent active attacks absolutely, because of the wide variety of potential physical, software, and network vulnerabilities.
- Instead, the goal is to **detect** active attacks and to recover from any disruption or delays caused by them. If the detection has a deterrent effect, it may also contribute to prevention.

Security Services

- Defined by X.800 as:
 - A service provided by a protocol layer of communicating open systems and that ensures adequate security of the systems or of data transfers
- Defined by RFC 4949 as:
 - A processing or communication service provided by a system to give a specific kind of protection to system resources

Security Services



Authentication

- Concerned with assuring that a communication is authentic
 - In the case of a single message, assures the recipient that the message is from the source that it claims to be from
 - In the case of ongoing interaction, assures the two entities are authentic and that the connection is not interfered with in such a way that a third party can masquerade as one of the two legitimate parties

Two specific authentication services are defined in X.800:

- Peer entity authentication
- Data origin authentication

Access Control

- The ability to limit and control the access to host systems and applications via communications links
- To achieve this, each entity trying to gain access must first be identified, or authenticated, so that access rights can be tailored to the individual

Data Confidentiality

- The protection of transmitted data from passive attacks
 - Broadest service protects all user data transmitted between two users over a period of time
 - Narrower forms of service includes the protection of a single message or even specific fields within a message
- The protection of traffic flow from analysis
 - This requires that an attacker not be able to observe the source and destination, frequency, length, or other characteristics of the traffic on a communications facility

Data Integrity

- Can apply to a stream of messages, a single message, or selected fields within a message
- Connection-oriented integrity service, one that deals with a stream of messages, assures that messages are received as sent with no duplication, insertion, modification, reordering, or replays
- A connectionless integrity service, one that deals with individual messages without regard to any larger context, generally provides protection against message modification only

Nonrepudiation

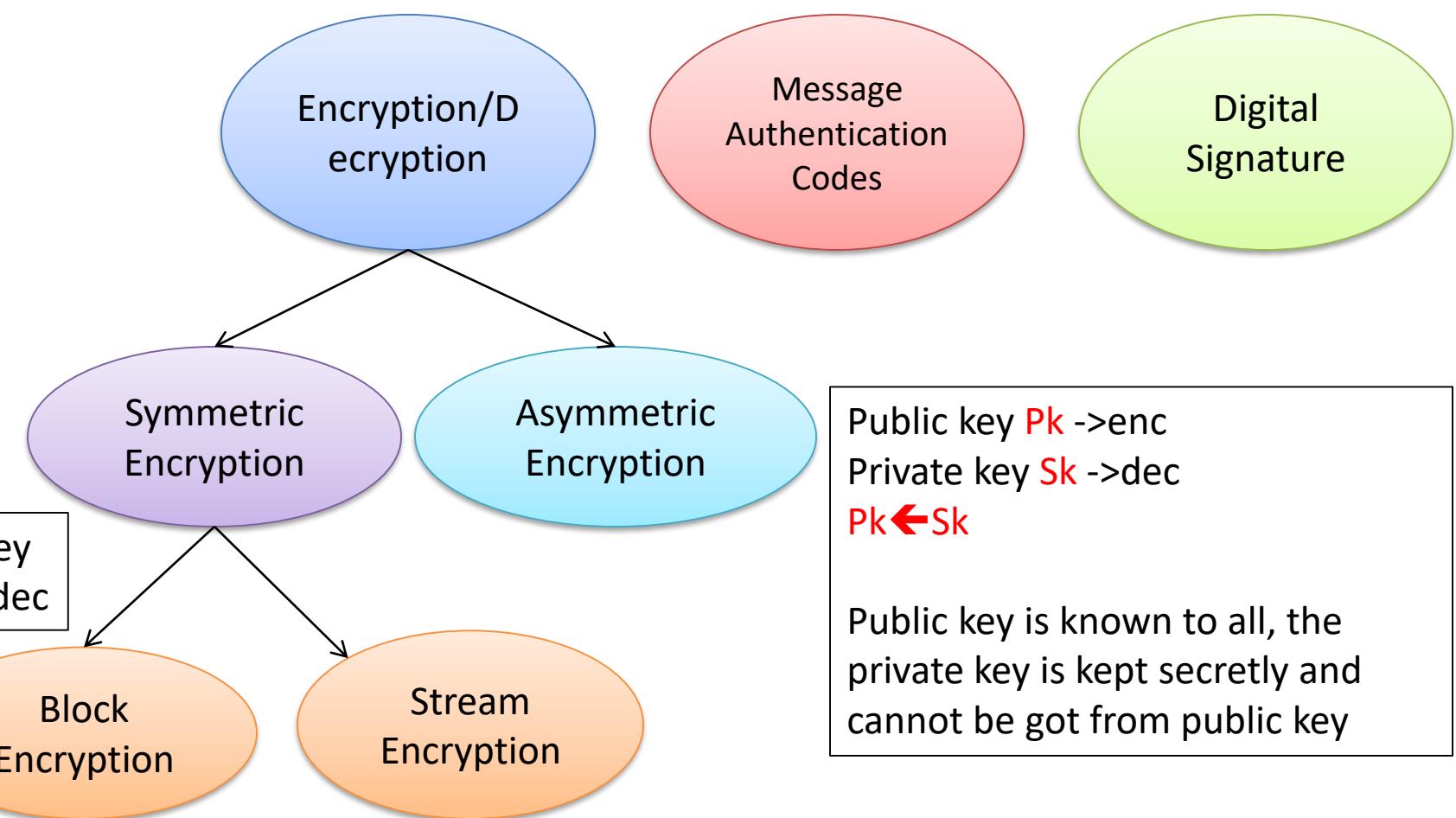
- Prevents either sender or receiver from denying a transmitted message
- When a message is sent, the receiver can prove that the alleged sender in fact sent the message
- When a message is received, the sender can prove that the alleged receiver in fact received the message

Availability Service

- Protects a system to ensure its availability
- This service addresses the security concerns raised by denial-of-service attacks
- It depends on proper management and control of system resources and thus depends on access control service and other security services

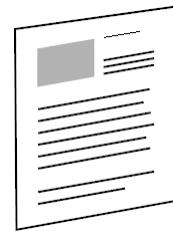
Security Mechanisms

- **Cryptographic Tools**



Encryption/Decryption

Symmetric Encryption



message M

$$C = \text{Enc}(M, K)$$



$$M = \text{Dec}(C, K)$$

Key K



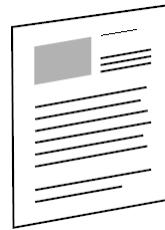
ciphertext C

recover M or K/Sk in
a **reasonable** time

Message Space

Key Space

Asymmetric Encryption



message M

Public key Pk



$$C = \text{Enc}(M, Pk)$$



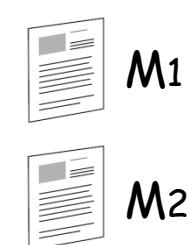
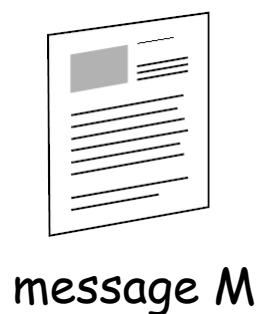
$$M = \text{Dec}(C, Sk)$$

Private key Sk



ciphertext C

Message Authentication/Hashing

 $H(M_1)$ $H(M_2)$

With 1-bit difference

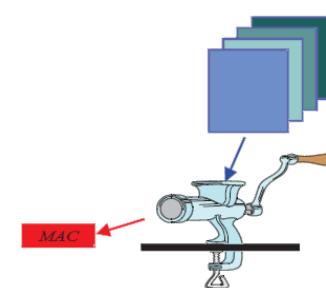
With unpredictable 50% change

 $H(M_1)$ 

No-collision

Given $H(M_1)$, it is impossible to find another message M_2 , whose hash value $H(M_2)=H(M_1)$

MD5: Message Digest 5
SHA1: Secure Hash Algorithm 1

 $H(M)$ 

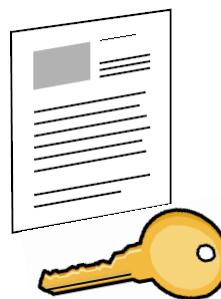
digest
Allow to detect any modification of M

No source authentication

Digital Signatures

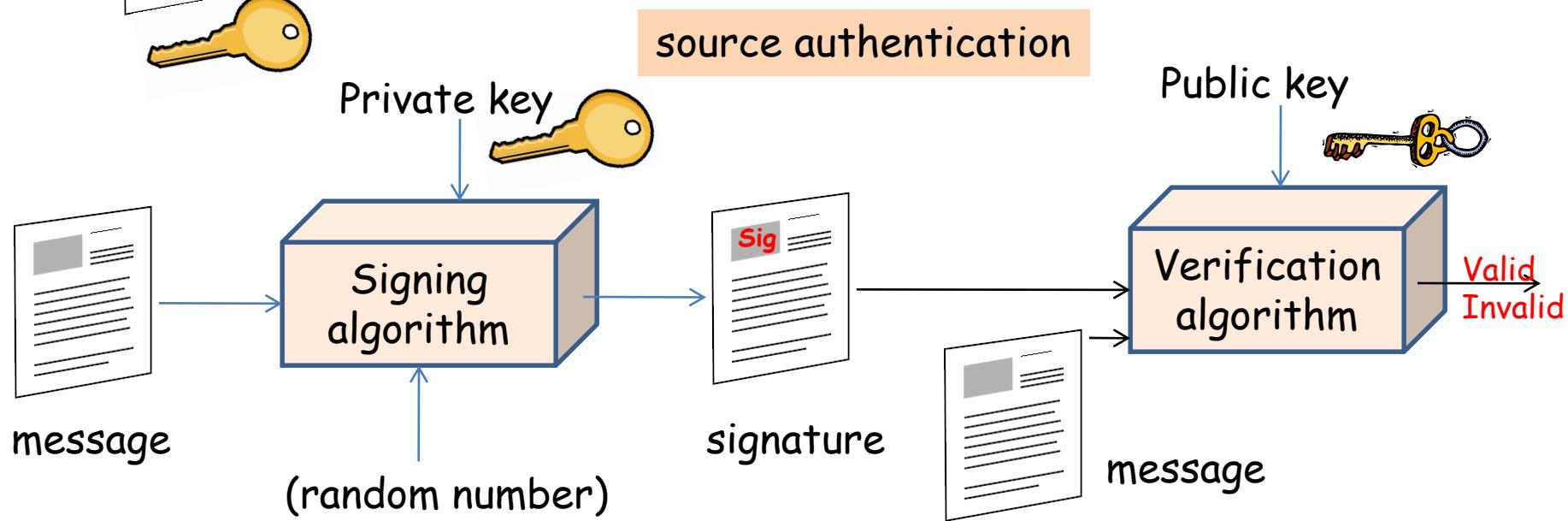


Handwriting signature: prove the authors and content of a message to a third party in a later time

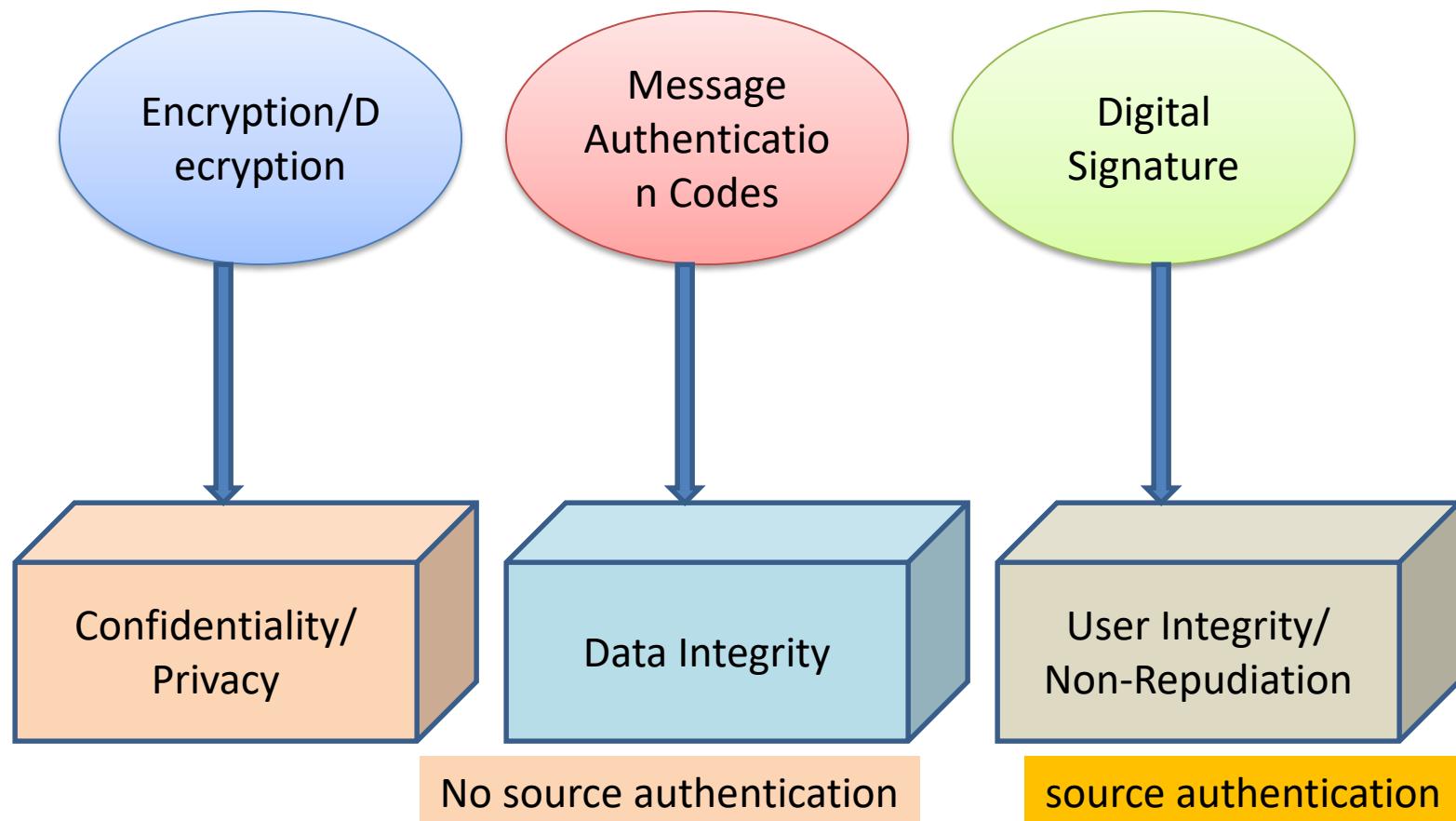


Digital Analogy of handwriting signature

Digital signature uses private key **Sk** to sign an electronic file



Cryptographic Objectives



History of Cryptography

Ancient period

Technical period

Paradoxical period

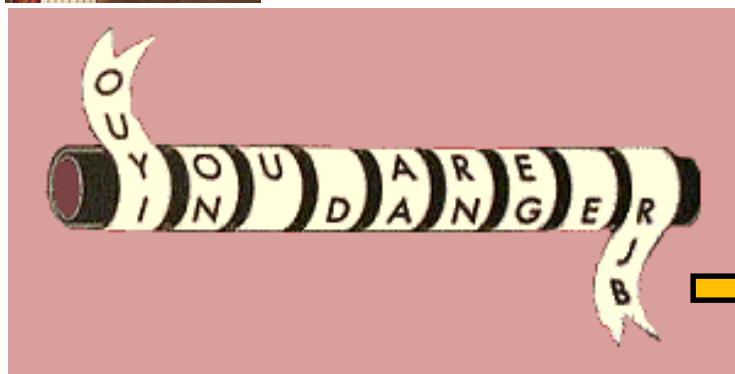


- **Ancient period:** (- until 1918)
 - with relatively simple algorithms that were designed and implemented **manually**.
- **Technical period:** (from 1919 until 1975)
 - Extensive use of encrypting electro-mechanical machines, especially in the period of the Second World War (**cipher machine**).
- **Paradoxical period:** (from Mid-1970s until -)
 - More pervasive use of computers in recent decades, supported by solid mathematical basis (number theory, group, ring, field theory, ...) (**cryptography on computer**)

Ancient period



transposition of letters
substitution (replace letters)



Historical Story 1

In 405 BC the Greek general LYSANDER OF SPARTA was sent a coded message written on the inside of a servant's belt. When LYSANDER wound the belt around a wooden baton the message was revealed. The message warned LYSANDER that Persia was about to go to war against him. He immediately set sail and defeated the Persians.

Historical Story 2

1	2	3	4	5	
1	A	B	C	D	E
2	F	G	H	W	K
3	L	M	N	O	P
4	Q	R	S	T	U
5	V	W	X	Y	Z

The Greeks also invented a code which changed letters into numbers. A is written as 11, B is 12, and so on. So WAR would read 52 11 42.

This kind of code cipher was still being used two thousand years later during the First World War.

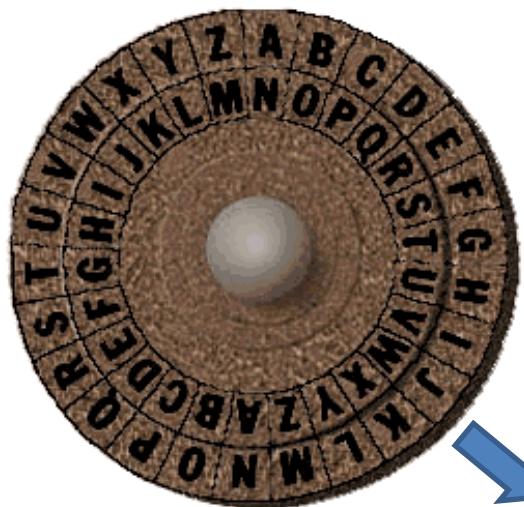
Plaintext	H	E	L	L	O	W	O	R	L	D
Ciphertext	23	15	31	31	34	52	34	42	31	14

Historical Story 3



Caesar Shift Cipher:

The Roman Emperor CAESAR invented his own simple code:
Each letter substituted by shifting $n=3$ places
His famous phrase VENI, VIDI, VICI ("I came, I saw, I conquered") would have read YHQL YLGL YLFL
 $V-Y, E-H, N-Q, I-L, D-G, C-I$



Cipher Disk

Plaintext

Ciphertext

H	E	L	L	O	W	O	R	L	D
U	R	Y	Y	B	J	B	E	Y	Q

Technical Period



Machine used for substitution

Historical Story 4 During the Second World War

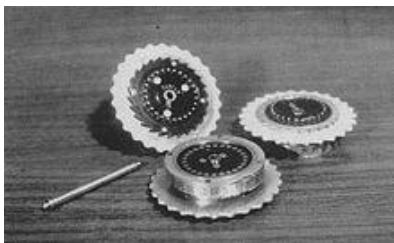
Before war broke out in 1939 the Germans had planned a special way of keeping their communications secret. The army, navy and air force were told to encode their messages using cipher machines called ENIGMA



ENIGMA

<https://www.youtube.com/watch?v=mXZNayEPFKc>

ENIGMA



Performing substitutions

key space = a total of 10^{17} combinations, i.e., ENIGMA can put a message into code in around Million, Million, Million different ways.

- The Enigma cipher machine looked like a traditional typewriter in a wooden box. An electric current went from the keyboard through a set of rotors and a plugboard to light up the 'code' alphabet.
- At least once a day the Germans changed the order of the rotors, their starting positions and the plugboard connections.
- To decipher a message sent using Enigma, you had to work out exactly how all of these had been set.
- In the 1930's Polish cipher experts secretly began to try to crack the code. Obtained information on its usage: 1) daily code book indicated rotors and orientation; 2) a different orientation key for each message. Just before war broke out they managed to pass models and drawings of Enigma to British and French code-breakers. Finally, Enigma was broken.

Paradoxical Period



Symmetric
Cryptography

Asymmetric
Cryptography

- The draft *Data Encryption Standard (DES)* was published in the U.S. Federal Register on 17 March 1975, which is an effort to develop secure electronic communication facilities for businesses such as banks and other large financial organizations. DES was adopted and published as a Federal Information Processing Standard Publication in 1977. The release of its specification stimulated an explosion of public and academic interest in cryptography.
- In 1976, *New Directions in Cryptography* by Whitfield Diffie and Martin Hellman was published, which is a very significant step in the history of cryptography. In the paper, it introduced a radically new method of distributing cryptographic keys, which went far toward solving one of the fundamental problems of cryptography, key distribution, and has become known as Diffie-Hellman key exchange. In addition, it also stimulated the birth of a new class of enciphering algorithms, the asymmetric cryptography.

Kerckhoffs Principles

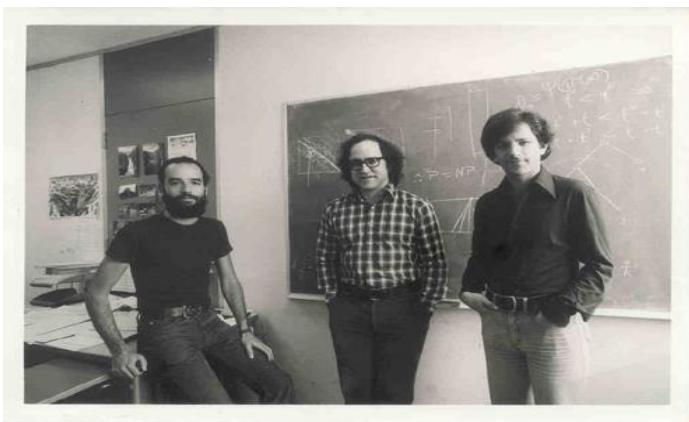


1. The system must be substantially, if not mathematically, undecipherable;
2. The system must not require secrecy and can be stolen by the enemy without causing trouble;
3. It must be easy to communicate and remember the keys without requiring written notes, it must also be easy to change or modify the keys with different participants;
4. The system ought to be compatible with telegraph communication;
5. The system must be portable, and its use must not require more than one person;
6. Finally, regarding the circumstances in which such system is applied, it must be easy to use and must neither require stress of mind nor the knowledge of a long series of rules.

- A cryptosystem should be secure even if everything about the system, except the key, is public knowledge.
- Kerckhoffs' principle was reformulated by Claude Shannon as "The enemy knows the system." In that form, it is called Shannon's maxim.

RSA Public Key Cryptosystem

- R. Rivest, A. Shamir, L. Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. Communications of the ACM, Vol. 21 (2), pp.120-126. 1978.
- **History:** Famous Paper Rejection (anonymous reviewer's comments)
 - "R.L. RIVEST, A. SHAMIR, AND L. ADELMAN" *A Method for Obtaining Digital Signatures and Public-Key Cryptosystems.*" According to the (very short) introduction, this paper purports to present a practical implementation of Diffie and Hellman's public-key cryptosystem for applications in the electronic mail realm. If this is indeed the premise, the paper should be rejected both for a failure to live up to it and for its irrelevance. I doubt that a system such as this one will ever be practical. The paper does a poor job of convincing the reader that practicality is attainable.



$P \in Q \text{ PRIME}$
 $N = PQ$
 $ED \equiv 1 \pmod{(P-1)(Q-1)}$
 $C = M^e \pmod{N}$
 $M = C^d \pmod{N}$

Simple Classical Encryption Techniques

- Transposition
- Substitution
- Polyalphabetic Substitution
- Vigenere

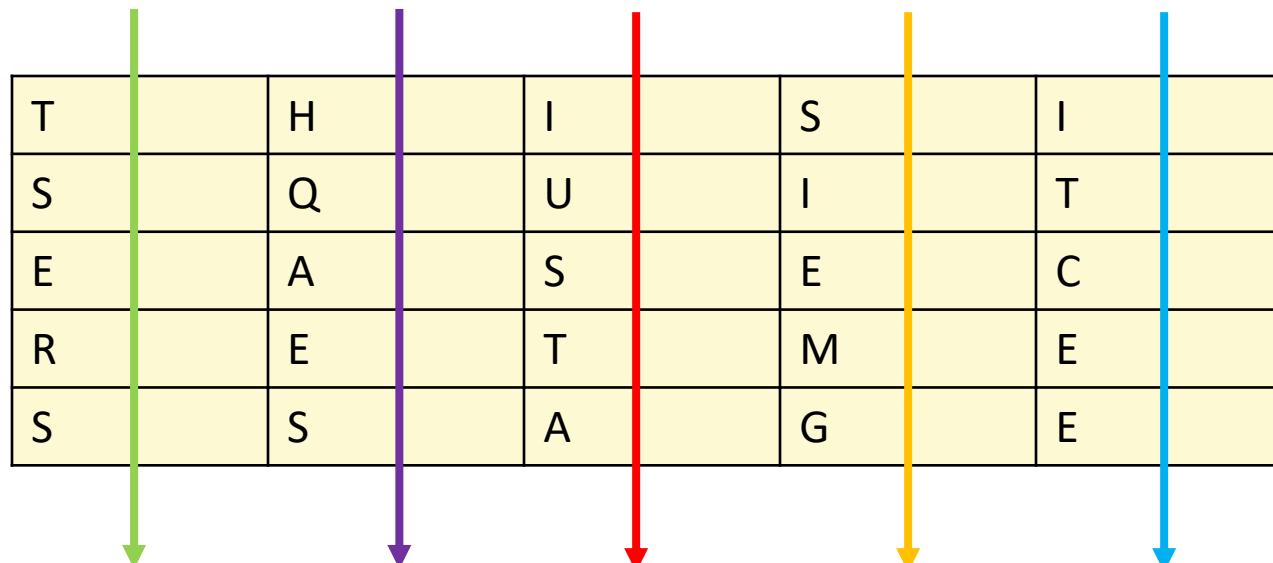
Transposition

- Transposition Ciphers rely on rearranging the order of letter according to some predetermined pattern
- Common method is Columnar Transposition - Write message in a matrix then rearrange columns
- Example:
 - "THIS IS QUITE A SECRET MESSAGE"
 - Represent as 5 X 5 matrix

T	H	I	S	I
S	Q	U	I	T
E	A	S	E	C
R	E	T	M	E
S	S	A	G	E

Columnar Transposition

- “Key” is the order in which columns are read, — choose 3-1-5-4-2
- Ciphertext is now
“IUSTATSERSTCEESIEMGHQAES”



Cryptanalysis of Cipher

- “IUSTATSER SITCEESIEMGHQAES” –
“Looks” complex, but ...
- Observation - letters do not appear
equally in English text
- Frequency Analysis

Single Letter Frequency

- Frequency Analysis
- In English

a	8.2%	j	0.2	s	6.3
b	1.5	k	0.8	t	9.1
c	2.8	l	4.0	u	2.8
d	4.3	m	2.4	v	1.0
e	12.7	n	6.7	w	2.4
f	2.2	o	7.5	x	0.2
g	2.0	p	1.9	y	2.0
h	6.1	q	0.1	z	0.1
i	7.0	r	6.0		

- Thus, letters ciphering **e**, **t**, and **a** are easily discovered
- Subsequently can look for the rest of the letters and letter pairs

Diagram Frequency and Trigram Frequency

The most frequent diagrams in English on a relative scale of 1 to 10:

Diagram	Frequency	Diagram	Frequency
TH	10.00	HE	9.05
IN	7.17	ER	6.65
RE	5.92	ON	5.70
AN	5.63	EN	4.76
AT	4.72	ES	4.24
ED	4.12	TE	4.04
TI	4.00	OR	3.98
ST	3.81	AR	3.54
ND	3.52	TO	3.50
NT	3.44	IS	3.43
OF	3.38	IT	3.26
AL	3.15	AS	3.00

The most frequent trigrams in English:

ENT	ION	AND	ING	IVE	TIO
FOR	OUR	THI	ONE		

Substitution Ciphers

- Message symbols are mapped into permuted set of symbols

A => B	B => W	C => E	D => K	E => Q
F => F	G => M	H => V	I => Y	J => A
K => L	L => U	M => C	N => O	O => N
P => P	Q => H	R => S	S => I	T => D
U => X	V => T	W => R	X => G	Y => Z

Cryptanalysis of Substitutions

- Brute force attacks: try all 26!
decipherments - if one decipherment per microsecond, it would take more than 10³ years!
- Analyze a large volume of ciphertext for letter frequency
- If frequencies are close to natural English only mapped to different letters, try replacement
- Digram and Trigram frequencies will also be preserved.

Shifted Alphabets

- Cipher formed by shifting letters of the alphabet k positions modulo $|l|$ (the size of the alphabet)
- For English

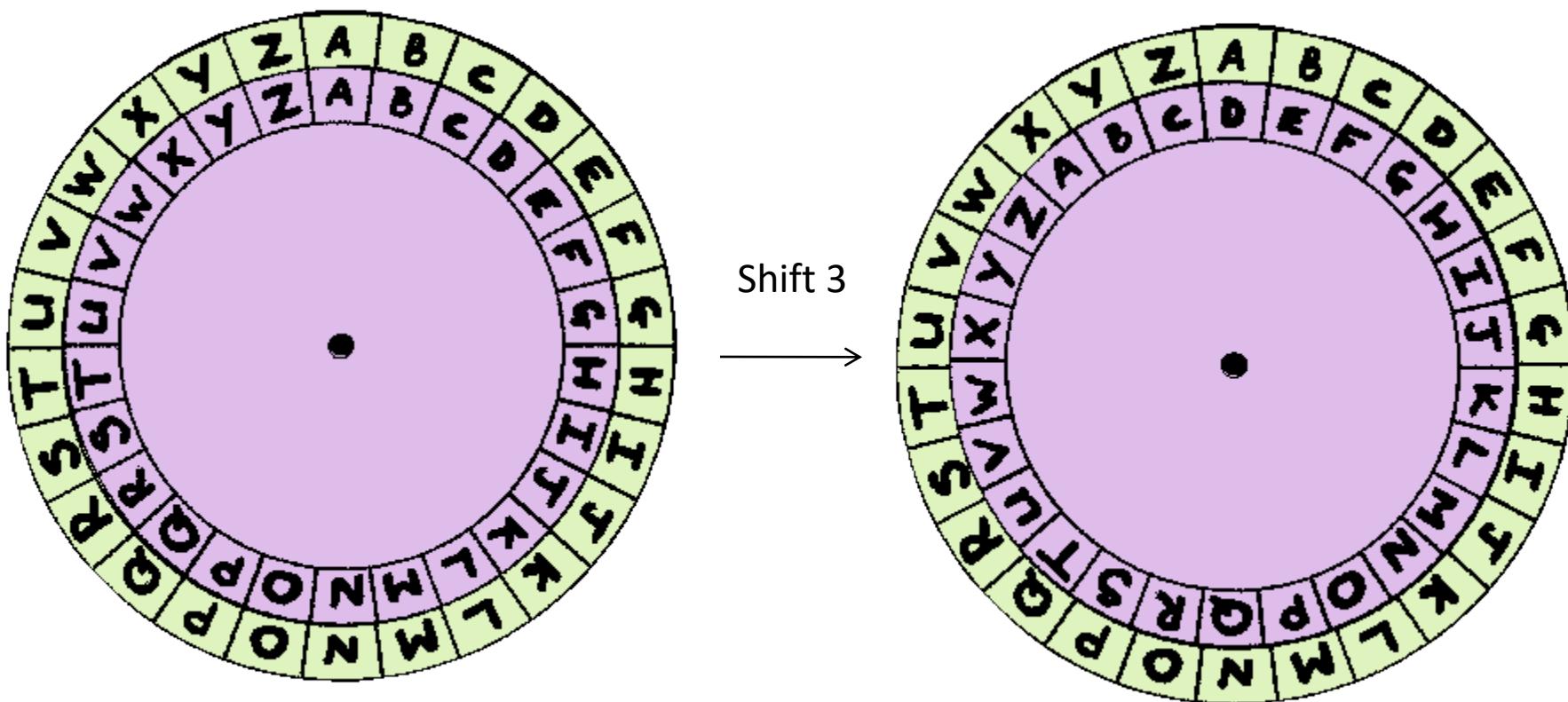
$$f(\blacksquare) = (\blacksquare + k) \bmod 26$$

- Multiplicative functions could also be used if k and l are relatively prime

$$f(\blacksquare) = \blacksquare \times k \bmod 26$$

Example: Caesar Cipher

- Very simple method is simply to shift alphabets — Caesar Ciphers (shift of 3)



T	H	I	S	I	S	A	C	A	E	S	A	R	C	I	P	H	E	R
W	K	L	V	L	V	D	F	D	H	V	D	U	F	L	S	K	H	U

Other Transforms

- Affine Transforms (Linear plus a constant)

$$f(\blacksquare) = (\blacksquare \cdot k + j) \bmod n$$

- Higher Order (define a poly of degree l)

$$f(\blacksquare) = \blacksquare^l \cdot k_l + \blacksquare^{l-1} \cdot k_{l-1} + \cdots + \blacksquare \cdot k_1 + k_0 \bmod n$$

Cryptanalysis

- All previous methods create a one-to-one mapping of plaintext to ciphertext.
- This is vulnerable to frequency analysis even for relatively small amounts of intercepted ciphertext
- Objective is to “flatten” symbol distribution in ciphertext.

Polyalphabetic Substitutions

- Reduce probability of successful correlation attacks by smearing statistics
- Use multiple substitutions
- $E_K(M) = f_1(m_1), f_2(m_2), \dots, f_d(m_d), f_{d+1}(m_{d+1}), \dots$

Vigenere Example

- Vigenere Cipher uses a repeated “Key Word” to perform substitutions

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
0	1	2	3	4	5	6	7	8	9	1	1	1	1	1	1	1	1	1	1	2	2	2	2	2	2

T	H	I	S		I	S		A		V	I	G	E	N	E	R	E		C	I	P	H	E	R	Plaintext
I	C	R	I		C	R		I		C	R	I	C	R	I	C	R		I	C	R	I	C	R	Key
B	J	Z	A		K	J		I		X	Z	O	G	E	M	T	V		K	K	G	P	G	I	Ciphertext

$$\text{Ciphertext} = \text{Plaintext} + \text{Key} \mod 26$$

Coding in Cryptography



Bit, Byte and the ASCII code

- Definition: A bit is either 0 or 1.
- Definition: A byte is a string of 0's and/or 1's with length 8. So one byte equals 8 bits.
- Example:
 - 00000000, 11111111, 11110000, 10101010
- Example:
 - 1001111000011101 is a data of two bytes

Modulo-2 Addition

- Definition: The exclusive-or (XOR) (also called as modulo-2 addition) is defined as follows:

x	y	$x \oplus y$
0	0	0
1	0	1
0	1	1
1	1	0

- Remark: $x \oplus x = 0$ for any $x \in \{0,1\}$
- Remark: if $x \oplus y = z$ then $x = z \oplus y$

Modulo-2 Addition

- Bitwise exclusive-or: Let $x = x_1x_2 \cdots x_n$ and $y = y_1y_2 \cdots y_n$. The bitwise exclusive-or of x and y is
$$x \oplus y = (x_1 \oplus y_1)(x_2 \oplus y_2) \cdots (x_n \oplus y_n)$$
- Example:

	1	0	0	1	1
\oplus	1	0	1	1	0
	0	0	1	0	1

Encoding Message: ASCII Code

- America Standard Code for Inform. Interchanges
 1. Blank 1,
 2. letters (capital + lower) $26+26=52$,
 3. Digits (0,1,...,9) 10,
 4. Symbols 32 [punctuation symbols, accents, brackets, operators]
 5. Controls 33 (non-printable characters)
- Altogether 128 characters. Each is encoded into a string of 7 bits.
 - Example: e= 01100101, "=00100010
- Remark: The ASCII encoding rule is a 1-to-1 function F from the set of 128 characters to a subset of the set $\{0,1\}^8$, i.e., consisting of all binary strings of length 8 whose first bit is always 0.
- Remark: There are other ways to encode English messages and data. (Huffman coding)

Binary Representation of Numbers

$$i = i_{t-1}i_{t-2}\cdots i_1i_0 = i_{t-1} \times 2^{t-1} + i_{t-2} \times 2^{t-2} + \cdots + i_1 \times 2 + i_0$$

where each $i_j \in \{0,1\}$

Example: $1011 = 2^3 + 2^1 + 2^0 = 11$

Decimal Representation of Numbers

$$i = i_{t-1}i_{t-2} \cdots i_1i_0 = i_{t-1} \times 10^{t-1} + i_{t-2} \times 10^{t-2} + \cdots + i_1 \times 10^1 + i_0$$

where each $i_j \in \{0,1,2,\dots,9\}$

Example: $7019 = 7 \times 10^3 + 1 \times 10^1 + 9 \times 10^0 = 7019$

Hexadecimal Representation of Numbers

Define $a = 10, b = 11, c = 12, d = 13, e = 14, f = 15$

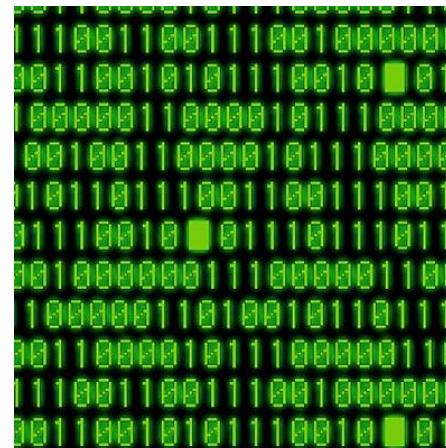
$$i = i_{t-1}i_{t-2} \cdots i_1i_0 = i_{t-1} \times 16^{t-1} + i_{t-2} \times 16^{t-2} + \cdots + i_1 \times 16 + i_0$$

where each $i_j \in \{0, 1, 2, \dots, 9, a, b, c, d, e, f\}$

Example: $8ad2 = 8 \times 16^3 + a \times 16^2 + d \times 16 + 2 \times 16^0 = 35536$

Encoding Message

- The purpose of encoding is not for data confidentiality, but to represent data or a message as binary string for information processing and interchanging.
- We can always encode messages to binary strings for processing.



Thank
you



CS 6355/4355: Cryptanalysis and Database Security

Topic 2: Symmetric Encryption

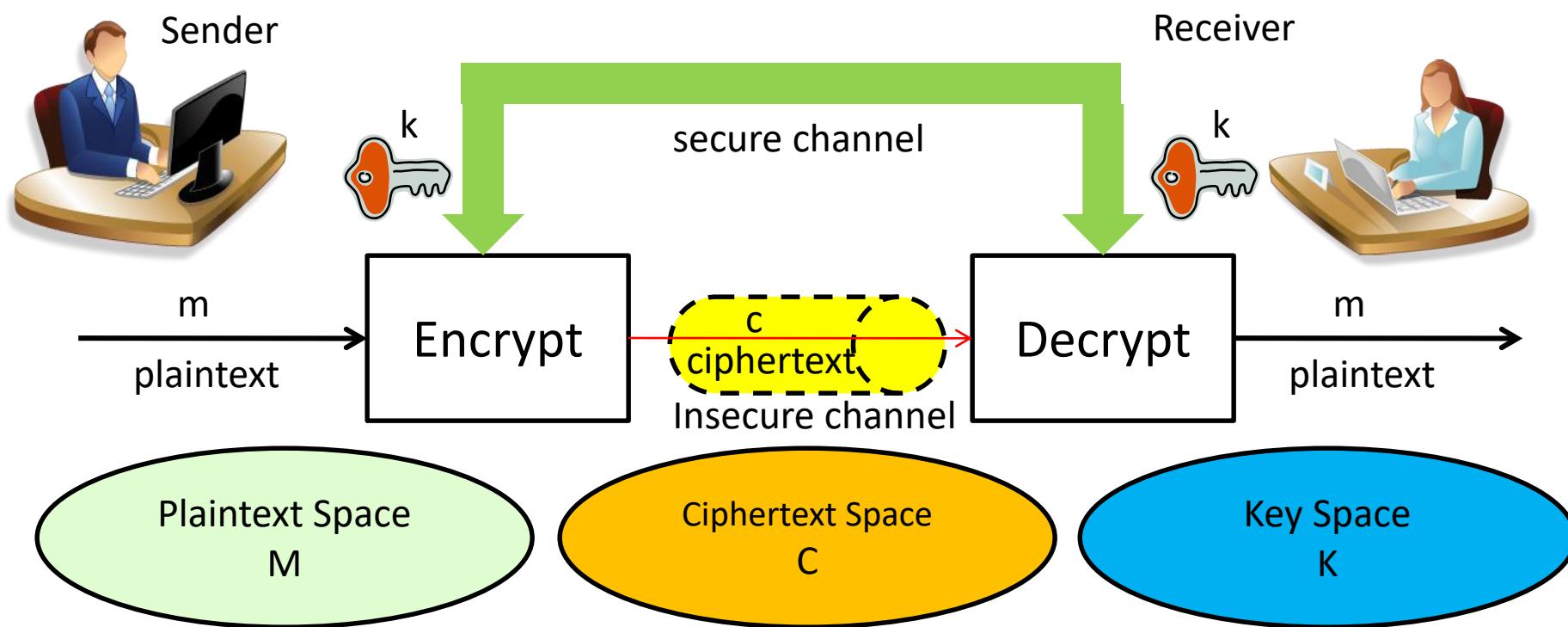
Lecturer: Rongxing LU

Email: RLU1@unb.ca Office: GE 114

Website: <http://www.cs.unb.ca/~rlu1/>

Faculty of Computer Science, University of New Brunswick

Symmetric Encryption



Symmetric Encryption

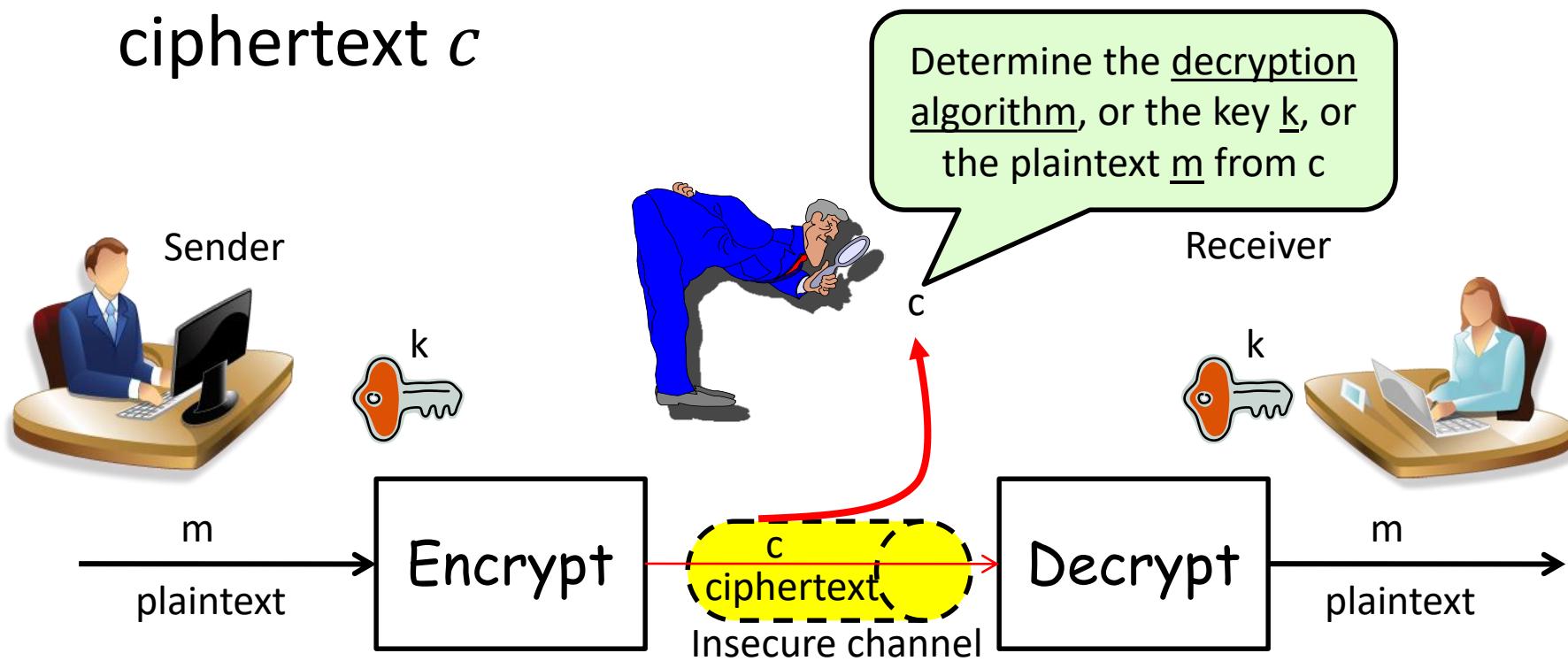
- A 5-tuple (M, C, K, Enc_k, Dec_k) , where
 - M, C, K are the message space, ciphertext space, and key space, respectively.
- Any key $k \in K$ could be the encryption and decryption key,
- Encryption: $c = Enc_k(m)$, where Enc_k is usually applied to blocks of or characters of a plaintext $m \in M$
- Decryption: $m = Dec_k(c)$, where Dec_k is usually applied to blocks or characters of the ciphertext $c \in C$
- Enc_k, Dec_k are encryption and decryption algorithms, with $Dec_k(Enc_k(m)) = m$ for each $m \in M$

Brute-Force Attack and Cryptanalysis on Symmetric Encryption

- Brute-force attack
 - Attacker tries every possible key on a piece of ciphertext until an intelligible translation into plaintext is obtained
 - On average, half of all possible keys must be tried to achieve success
 - To supplement the brute-force approach, some degree of knowledge about the expected plaintext is needed, and some means of automatically distinguishing plaintext from garble is also needed
- Cryptanalysis
 - Attack relies on the nature of the algorithm plus some knowledge of the general characteristics of the plaintext
 - Attack exploits the characteristics of the algorithm to attempt to deduce a specific plaintext or to deduce the key being used

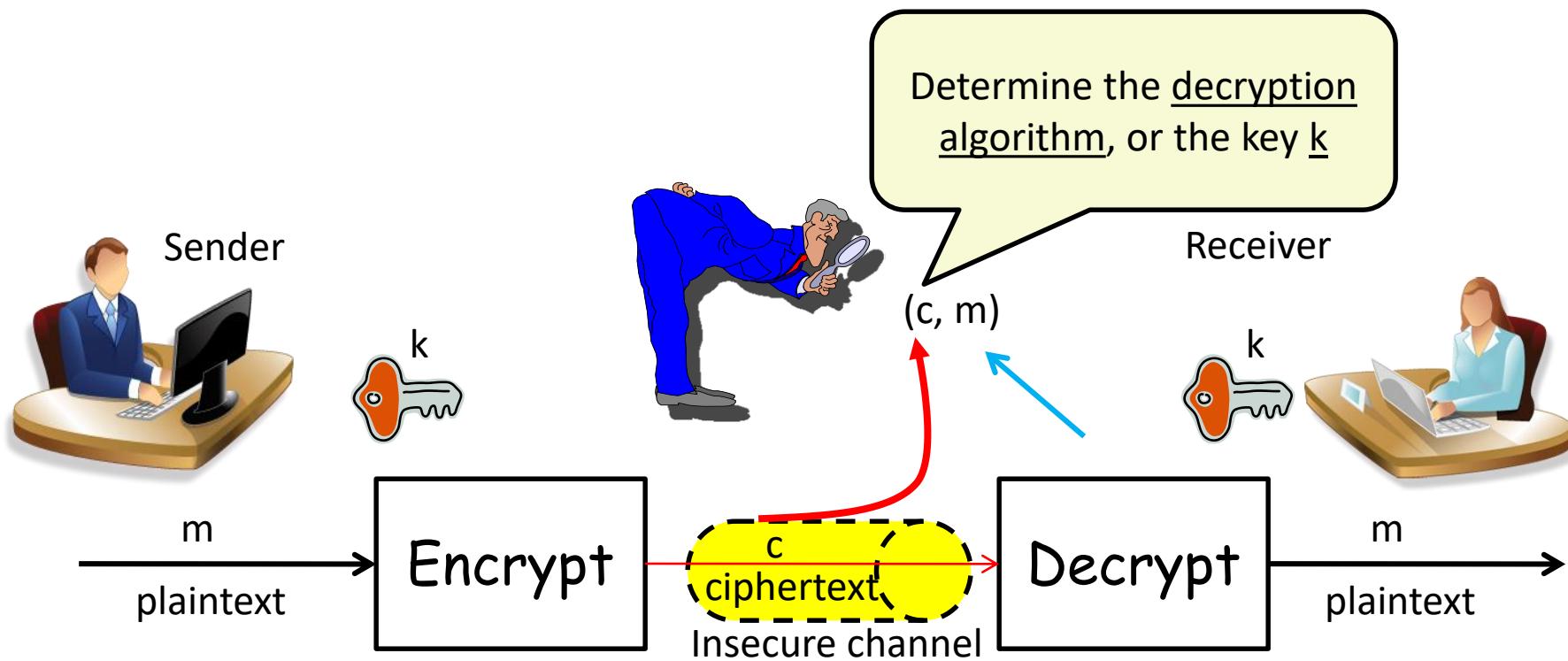
Attacks on Symmetric Encryption

- **Ciphertext-only attack:** An adversary determines the decryption algorithm Dec_k or key k , or the plaintext from intercepted ciphertext c



Attacks on Symmetric Encryption (2)

- **Known-plaintext attack:** An adversary determines the decryption algorithm Dec_k or key k , from a ciphertext-plaintext (c, m)

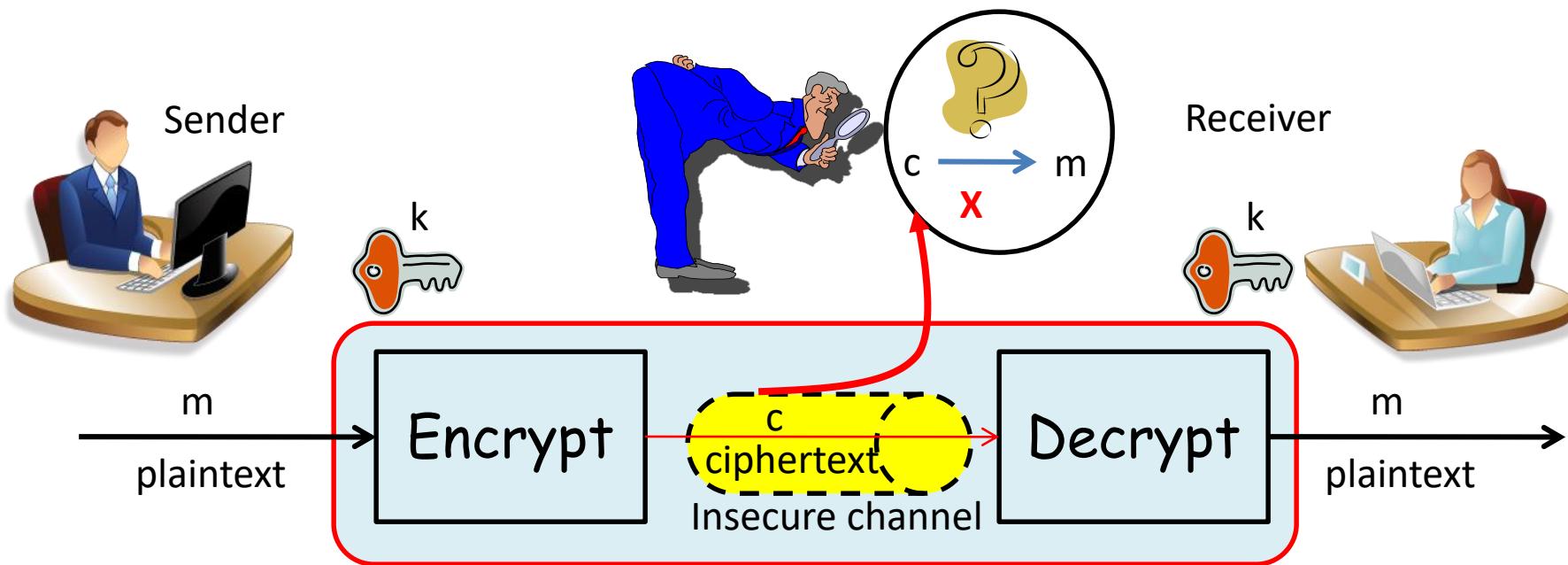


More Attacks on Symmetric Encryption

Type	Known to the Cryptanalyst
Ciphertext Only	<ul style="list-style-type: none">Encryption algorithm; Ciphertext
Known Plaintext	<ul style="list-style-type: none">Encryption algorithm; CiphertextOne or more plaintext-ciphertext pairs formed with the secret key
Chosen Plaintext	<ul style="list-style-type: none">Encryption algorithm; CiphertextPlaintext message chosen by cryptanalyst, together with its corresponding ciphertext generated with the secret key
Chosen Ciphertext	<ul style="list-style-type: none">Encryption algorithm; CiphertextCiphertext chosen by cryptanalyst, together with its corresponding decrypted plaintext generated with the secret
Chosen Text	<ul style="list-style-type: none">Encryption algorithm; CiphertextPlaintext message chosen by cryptanalyst, together with its corresponding ciphertext generated with the secret keyCiphertext chosen by cryptanalyst, together with its corresponding decrypted plaintext generated with the secret

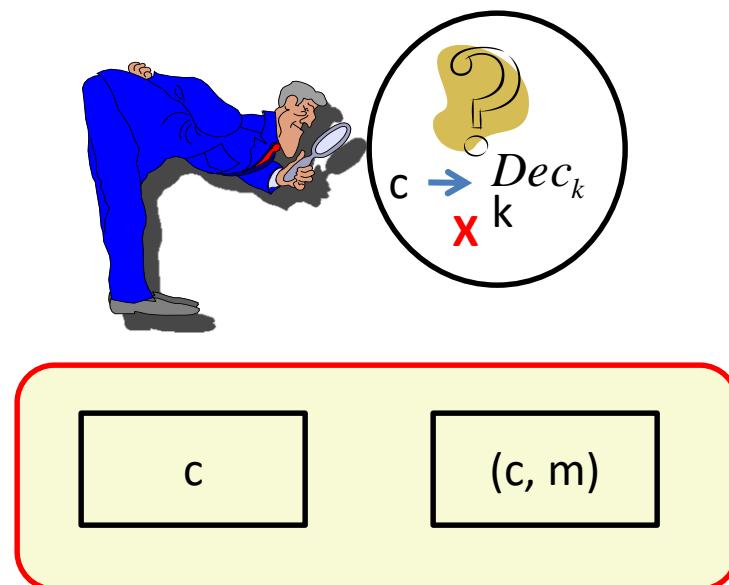
Security Requirements

- According to Kerckhoffs Principles, the security should depend on the confidentiality of the key, so it is usually assumed that the algorithms Enc_k and Dec_k are known to an adversary.
- It should be computationally infeasible for an adversary to determine the plaintext $m \in M$, given a ciphertext $c \in C$.



Security Requirements (2)

- It should be computationally infeasible for an adversary to systematically determine the decryption algorithm Dec_k or key k from intercepted ciphertext c , even if the corresponding plaintext m is known.



Properties of Symmetric Encryption

- How to design a symmetric cryptosystem to meet the above requirement?
- Two properties are desirable
 - Confusion
 - Diffusion

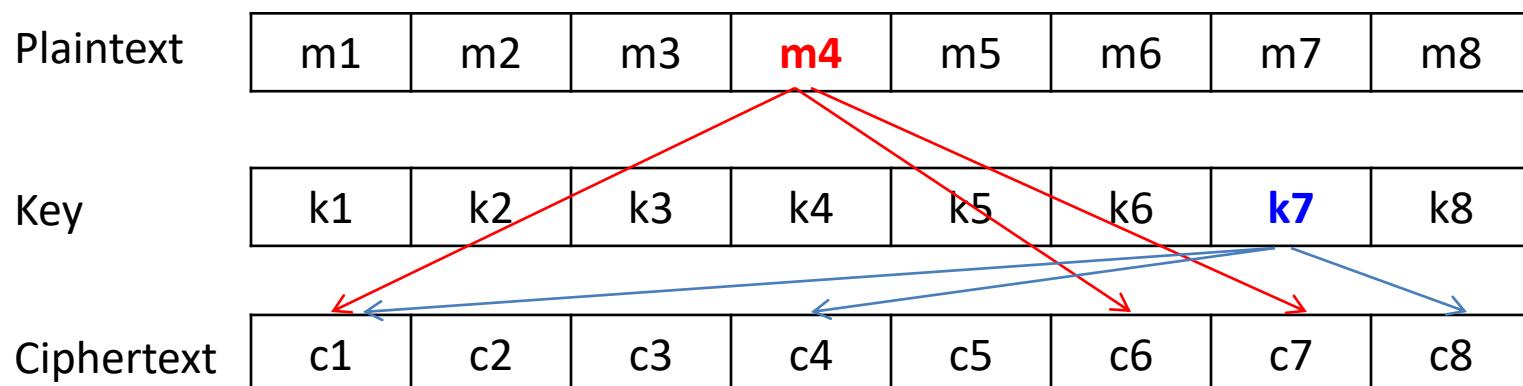
Confusion

- **Confusion:**
 - Process of substituting characters or symbols to make the relationship between ciphertext and key as complex as possible.
 - Attackers' uncertainty as to the contents of a message or the key used for encryption and decryption.

Plaintext	1	1	0	1	1	0	1	0
Key	0	1	0	1	1	1	0	1
Ciphertext = Plain \oplus Key	1	0	0	0	0	1	1	1

Diffusion

- **Diffusion:**
 - Process of spreading effect of plaintext or key as widely as possible over ciphertext.
 - Dispersion of the effect of individual key or message bits over the plaintext



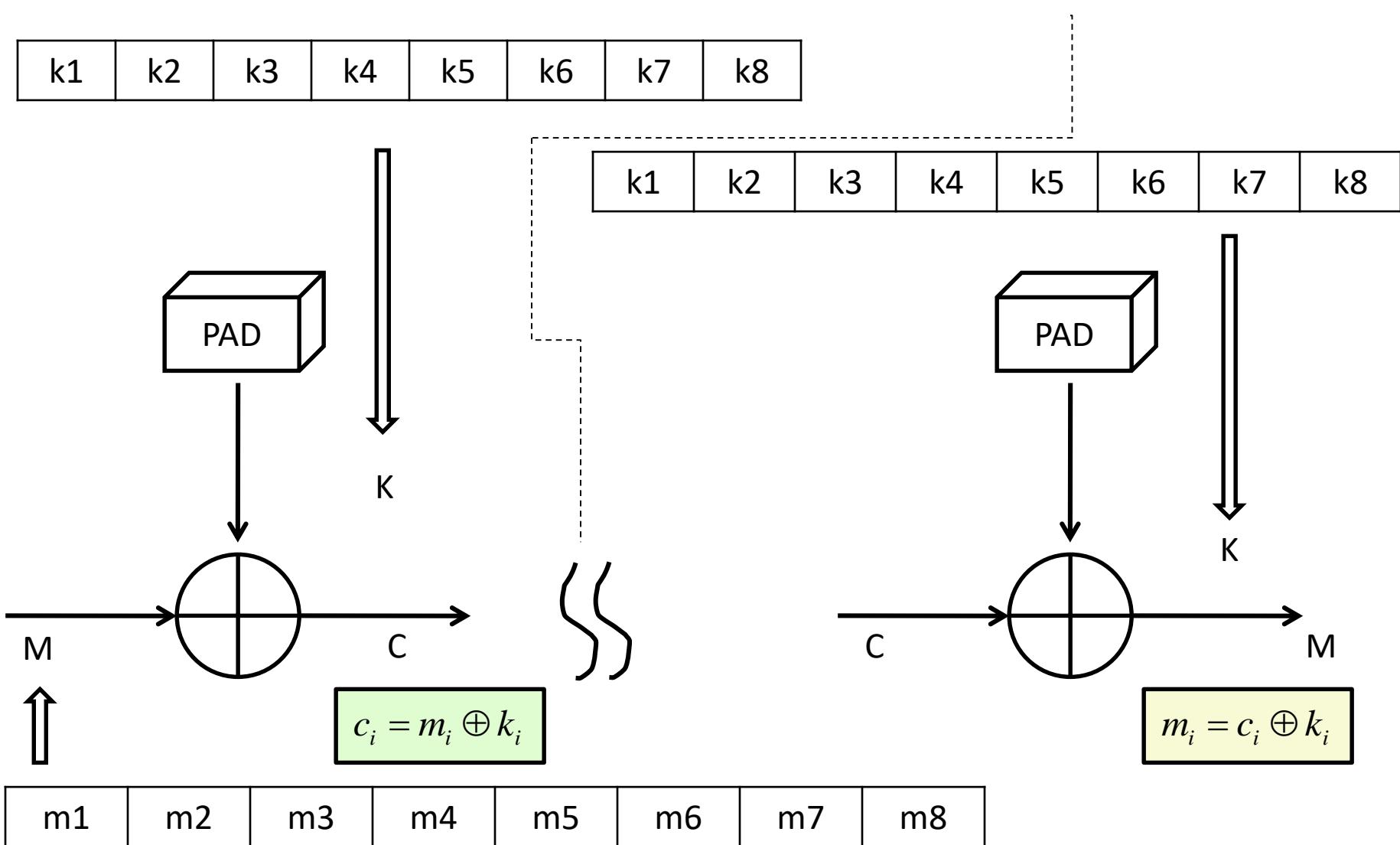
Classification of Secure Ciphers

- Unconditionally secure
 - No matter how much time an opponent has, it is impossible for him or her to decrypt the ciphertext simply because the required information is not there
 - **Cannot be broken regardless of attackers computational abilities**
- Computationally secure
 - The cost of breaking the cipher exceeds the value of the encrypted information
 - The time required to break the cipher exceeds the useful lifetime of the information
 - **Secure against attacker with “reasonable” resources**

Unconditionally secure

- Only system known to be provably unconditionally secure are “one-time pads”
- Each message is encoded into a binary string using the ASCII code;
- The secret key is a random binary string with the same length as the message;
- A secret key is used only for one message and is then discarded.

One-Time Pad



Why One-Time Pad is provably secure?

- The security depends on the randomness of the key, but it is hard to define randomness.
- In cryptographic context, we seek two fundamental properties in a binary random key sequence:
 - **Unpredictability:** Independent of the number of the bits of a sequence observed, the probability of guessing the next bit is not better than $1/2$. Therefore, the probability of a certain bit being 1 or 0 is exactly equal to $1/2$.
 - **Balanced (Equal Distribution):** The number of 1 and 0 should be equal.

Mathematical Proof

- the probability of a key bit being 1 or 0 is exactly equal to $\frac{1}{2}$
- The plaintext bits are not balanced. Let the probability of 0 be x and then the probability of 1 turns out to be $1 - x$.
- We can calculate the probability of ciphertext bits.

m_i	Prob. m	k_i	Prob. k	c_i	Prob. c
0	x	0	$1/2$	0	$x/2$
0	x	1	$1/2$	1	$x/2$
1	$1 - x$	0	$1/2$	1	$(1 - x)/2$
1	$1 - x$	1	$1/2$	0	$(1 - x)/2$

- We find out the probability of a ciphertext bit being 1 or 0 is equal to $\frac{1}{2} \cdot x + \frac{1}{2} \cdot (1 - x) = \frac{1}{2}$, and the ciphertext looks like a random sequence.

Computationally Secure

- “Useful” Cryptosystem
- Observe the message have only limited “secret” lifetime;
- Balance between size of cipher (speed and efficiency) and strength;
- Evolution driven by improvements in computational power;
- An encryption scheme is said to be computationally secure if:
 - The cost of breaking the cipher exceeds the value of the encrypted information or
 - The time required to break the cipher exceeds the useful lifetime of the information.

Average time required for exhaustive key search

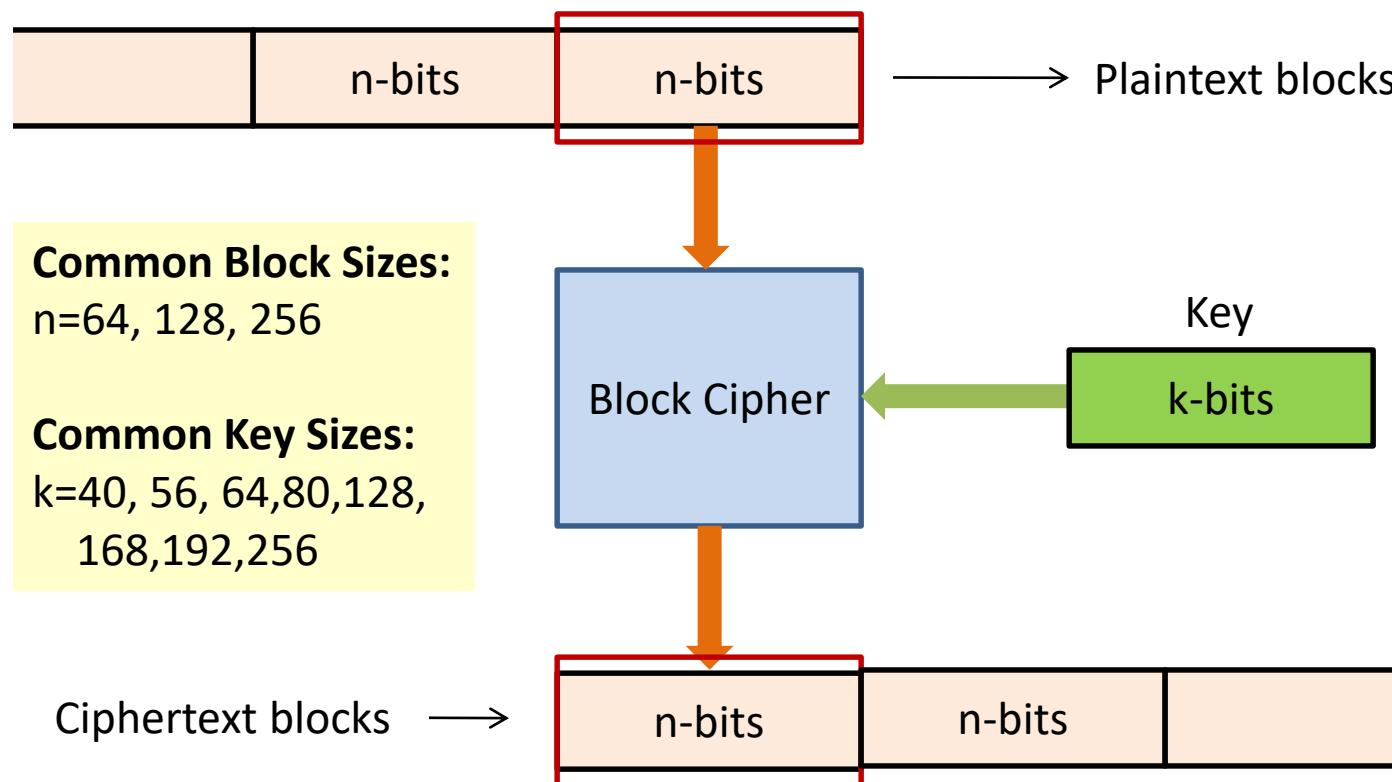
Key Size (bits)	Number of Alternatives Keys	Time required at 10^6 Decryption / μs
32	$2^{32} = 4.3 \times 10^9$	2.15 milliseconds
56	$2^{56} = 7.2 \times 10^{16}$	10 hours
128	$2^{128} = 3.4 \times 10^{38}$	5.4×10^{18} years
168	$2^{168} = 3.7 \times 10^{50}$	5.9×10^{30} years

Type of Ciphers

- Two basic formats:
 - Block ciphers
 - Block ciphers break messages into fixed length blocks, and encrypt each block using the same key.
 - The Data Encryption Standard (DES) is an example of a block cipher, where blocks of 64 bits are encrypted using a 56-bit key.
 - Stream ciphers
 - Stream ciphers, like block ciphers, break message into fixed length blocks, but use a sequence of keys to encrypt the blocks.
 - The Vigenere cipher is an example of a stream cipher.
 - Key = $k_1 k_2 k_3 k_4$ (random, used one-time only)
 - Plaintext = $m_1 m_2 m_3 m_4$; Ciphertext = $c_1 c_2 c_3 c_4$, where $c_i = m_i \oplus k_i$

Block Ciphers

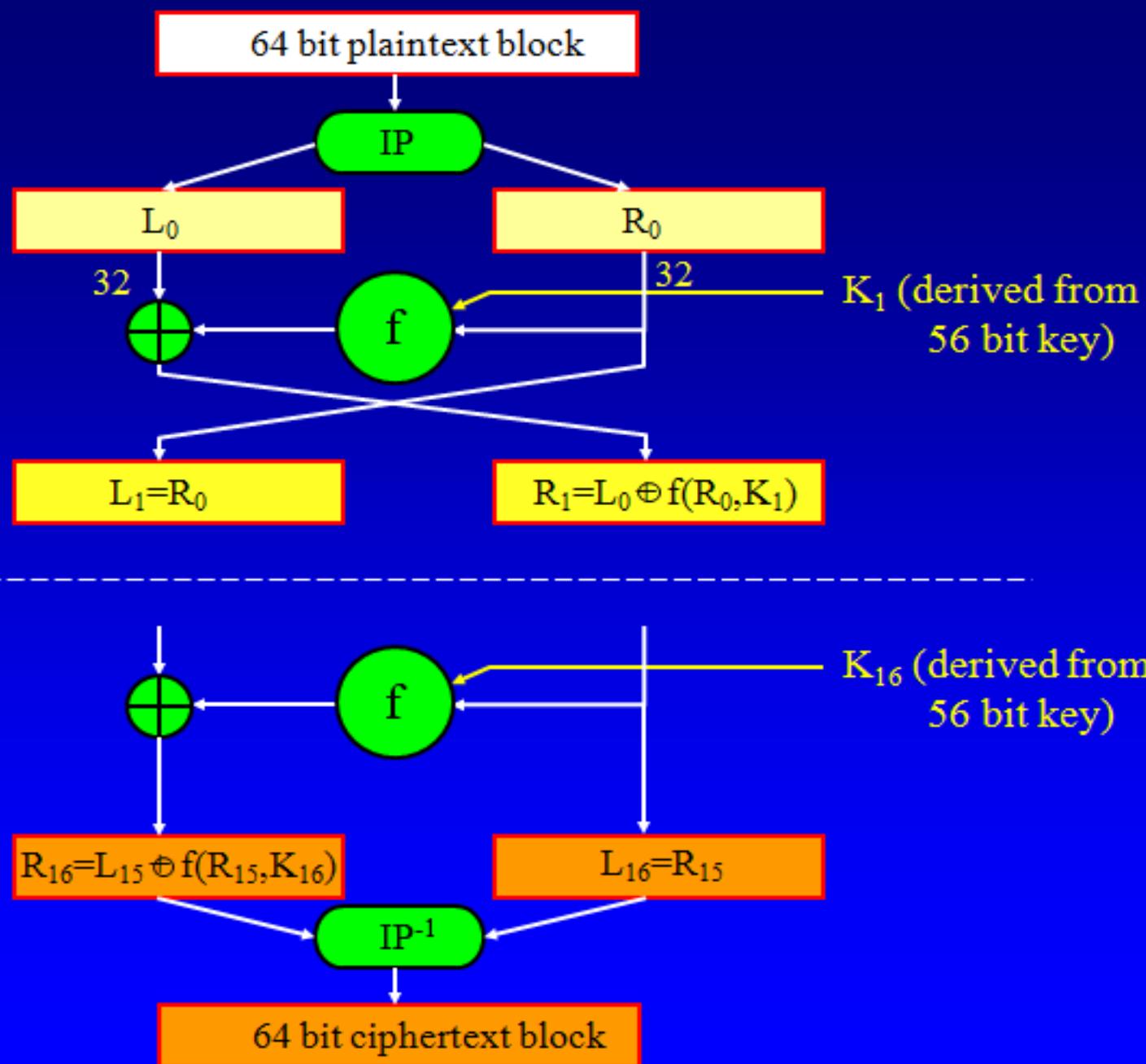
- Message is divided into fixed size blocks (block size) using padding if necessary
- Ciphertext is block of (usually) the same size



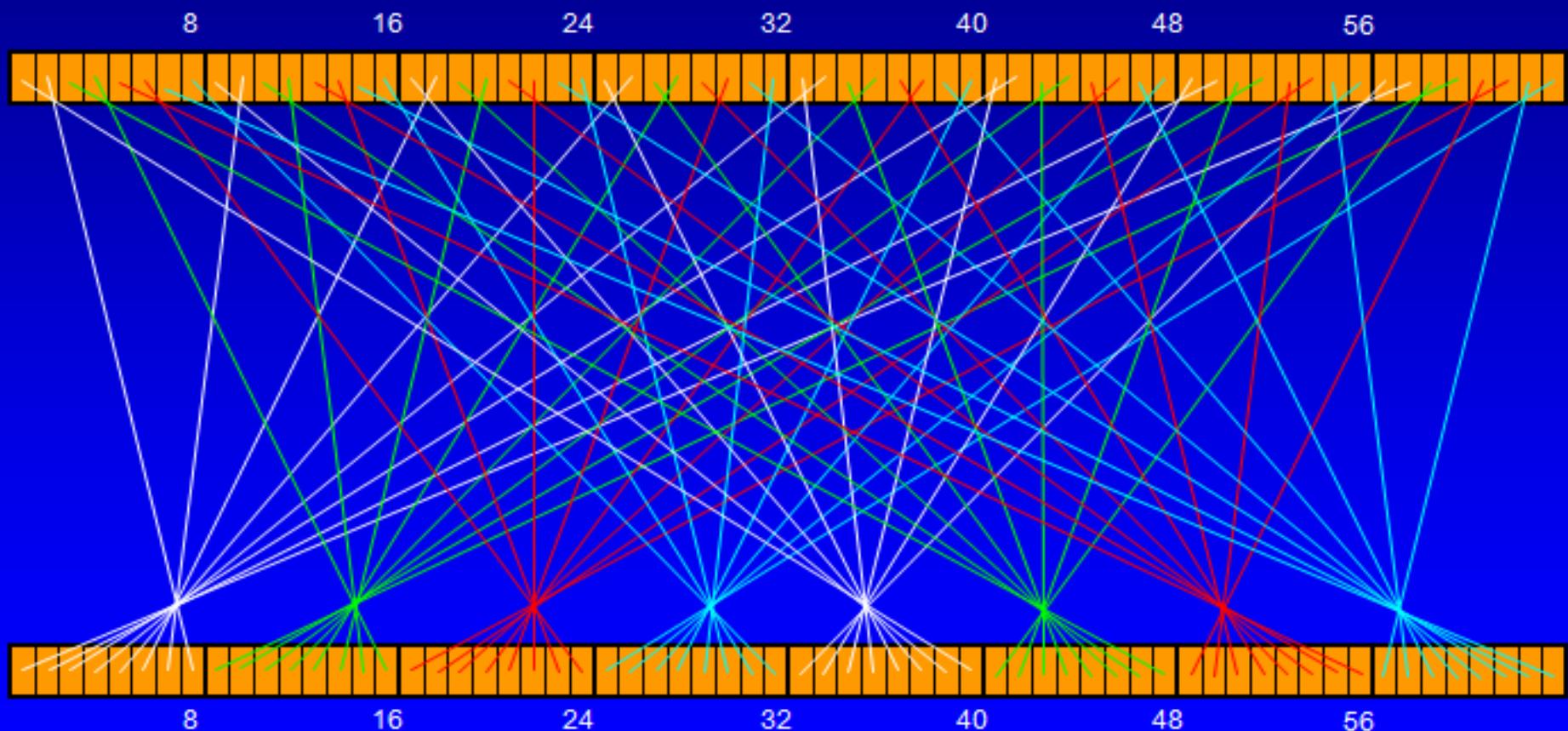
Block Ciphers (2)

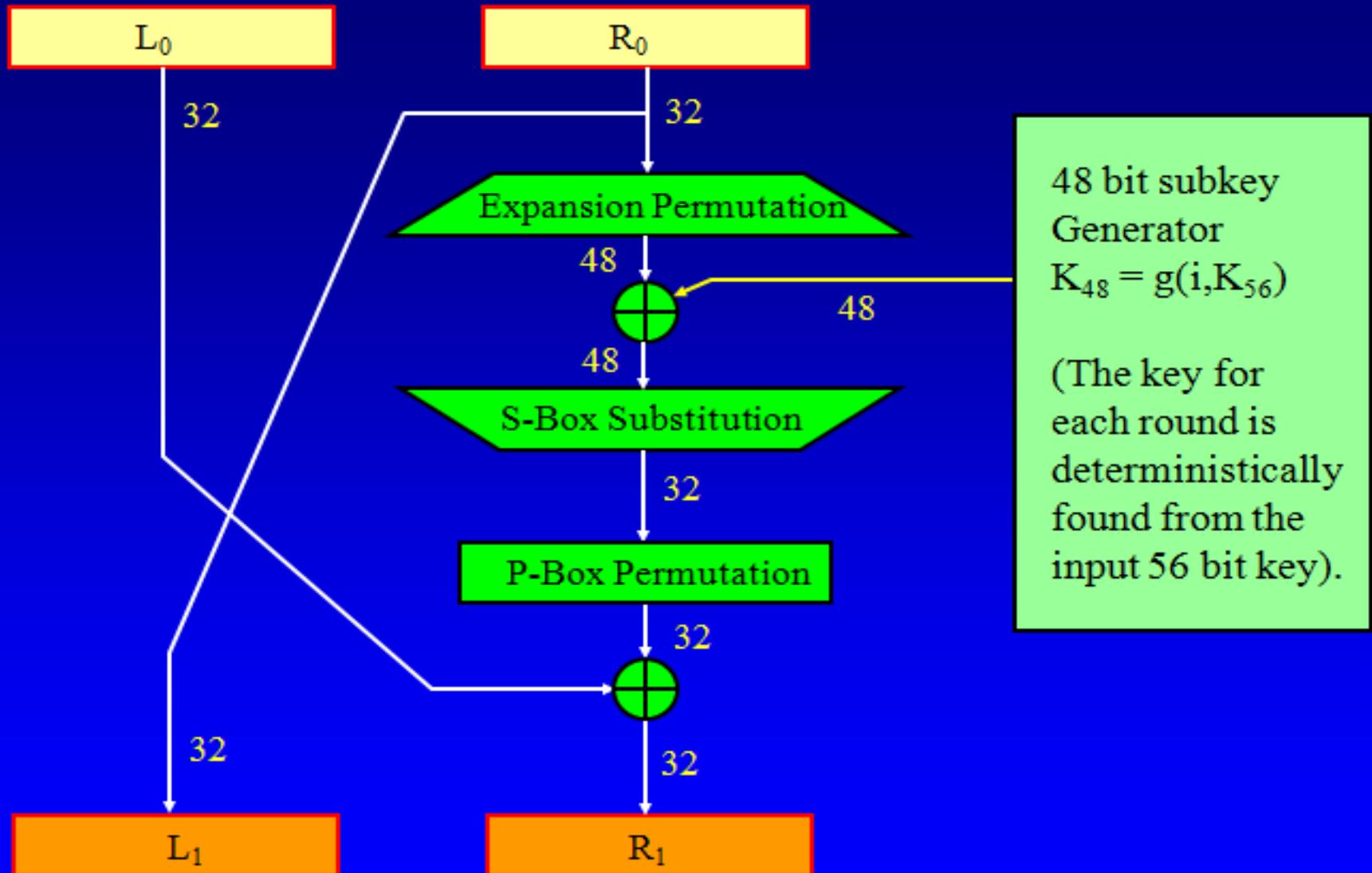
- Formal definition of Block Cipher
 - Let E be an encryption algorithm, and let $E_k(b)$ be the ciphertext of the message b with key k .
 - Let a message $m = b_1 b_2 \dots$ where each b_i is of a fixed length.
 - A block cipher is a cipher for which $E_k(m) = E_k(b_1)E_k(b_2) \dots$
- Properties of Block Ciphers
 - Adds Confusion about message and key
 - Should have good Diffusion Properties
 - Single bit change in input plaintext should produce changes in approx. 50% of output bits (at random)
 - Single bit change in key should produce changes in approx. 50% of output bits (at random)
- Example: DES encrypts 64-bit blocks
 - Key size 56 bits plus 8 parity bits

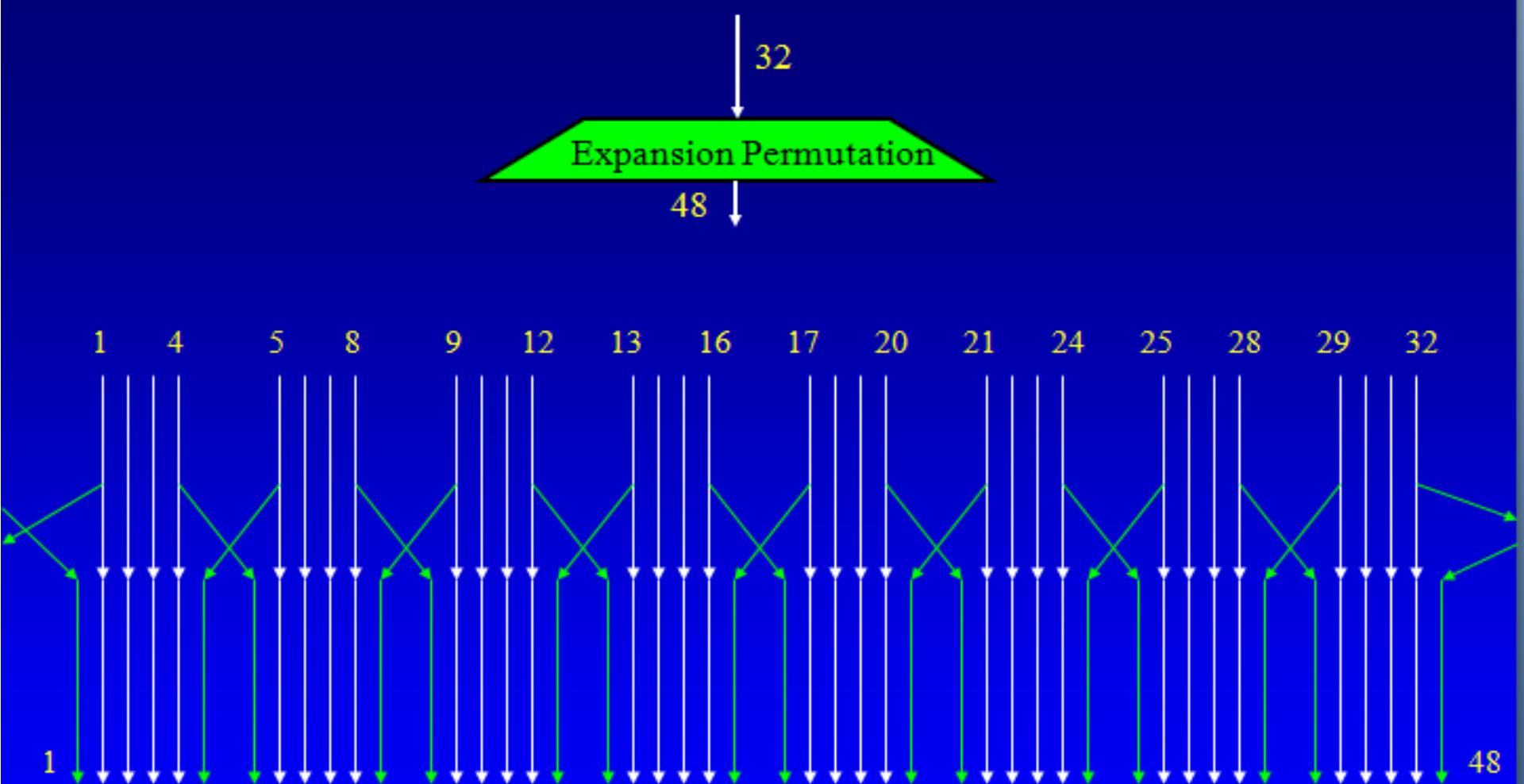
DES

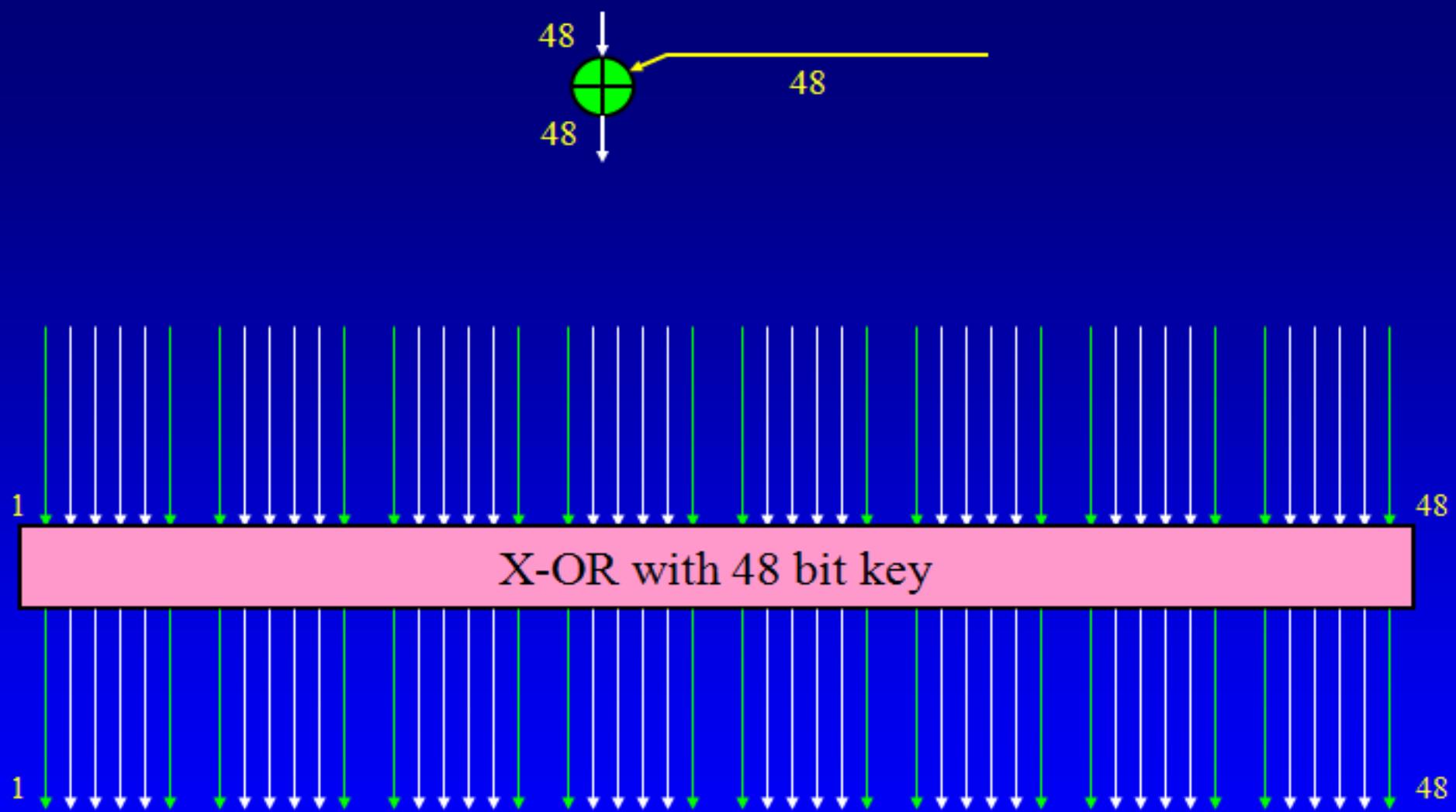


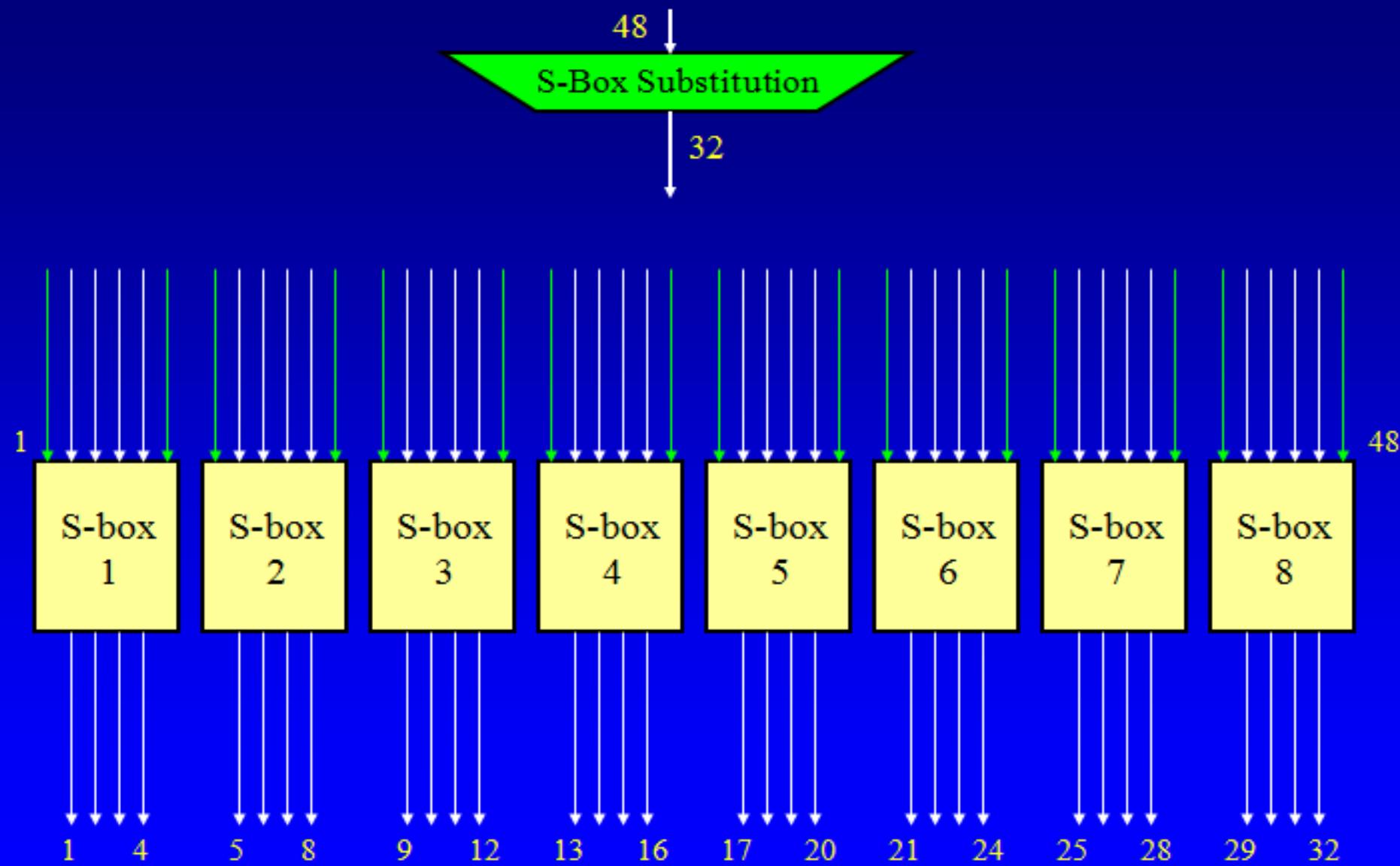
IP (Initial Permutation):



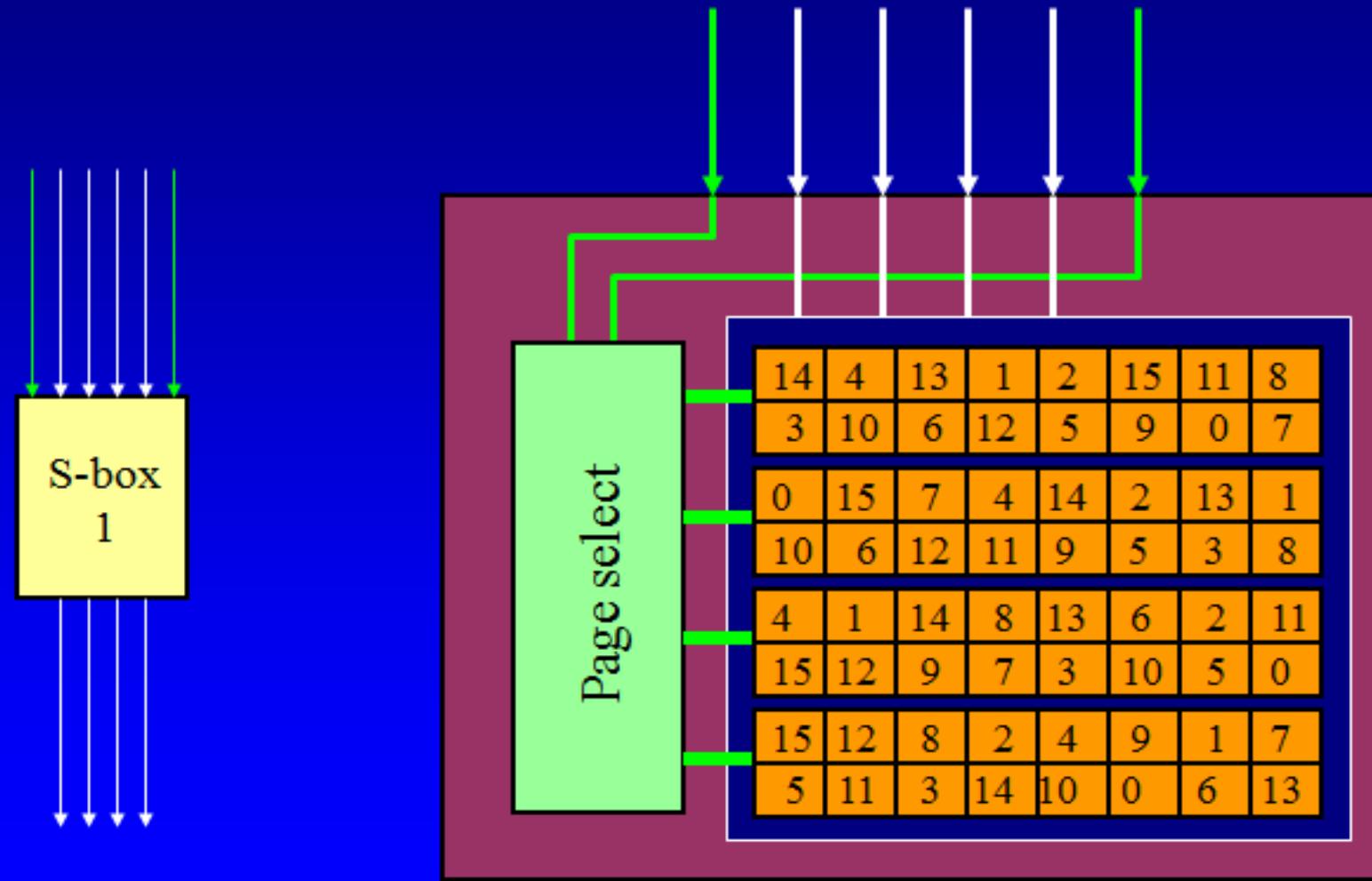








How an S-Box works

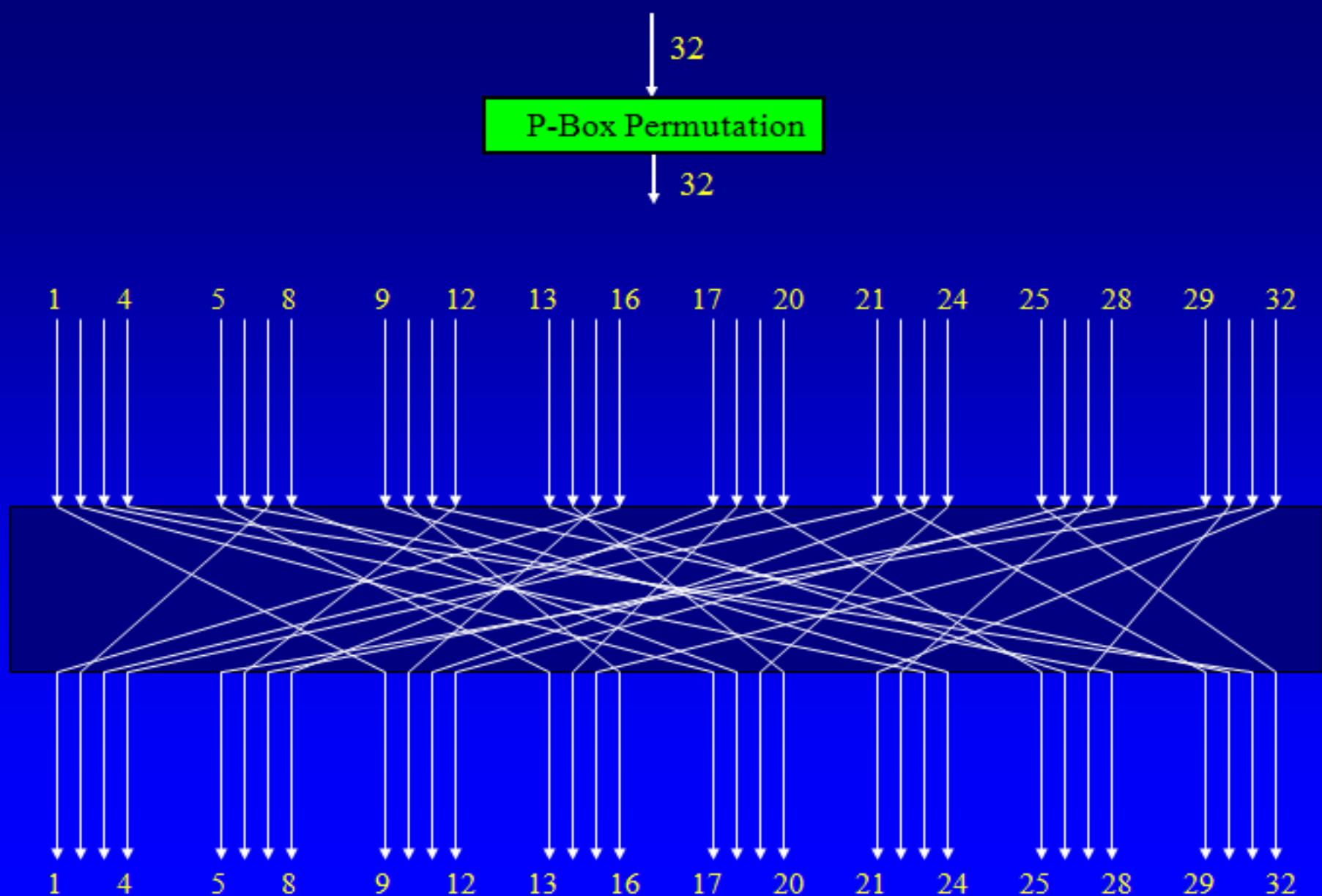


How an S-Box works

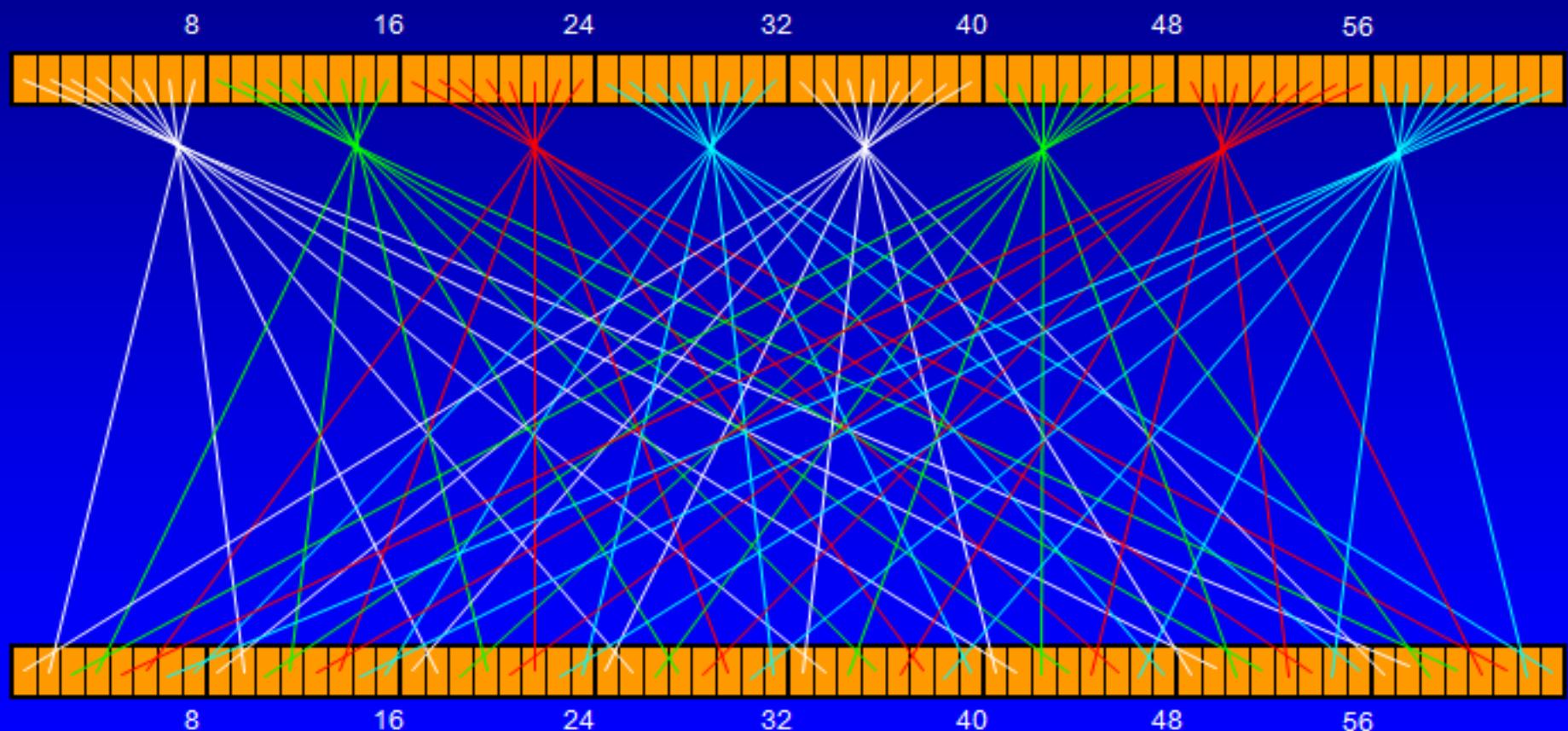
The diagram illustrates the operation of an S-box, specifically S_5 . It shows a 12-bit input word being processed by the S-box to produce a 4-bit output word. The input is divided into two parts: the "Outer bits" (the first 4 bits) and the "Middle 4 bits of input". The output is also divided into two parts: the first 3 bits and the last bit.

The S-box S_5 takes a 12-bit input and produces a 4-bit output. The input is shown as a sequence of 12 bits, grouped into four sets of three bits each. The first set (outermost) is labeled "Outer bits". The second set is labeled "Middle 4 bits of input". The third and fourth sets are grouped together. The output is shown as a sequence of 4 bits, grouped into three sets of two bits each. The first set is the first three bits of the output. The second set is the last bit of the output. The third set is the last two bits of the output.

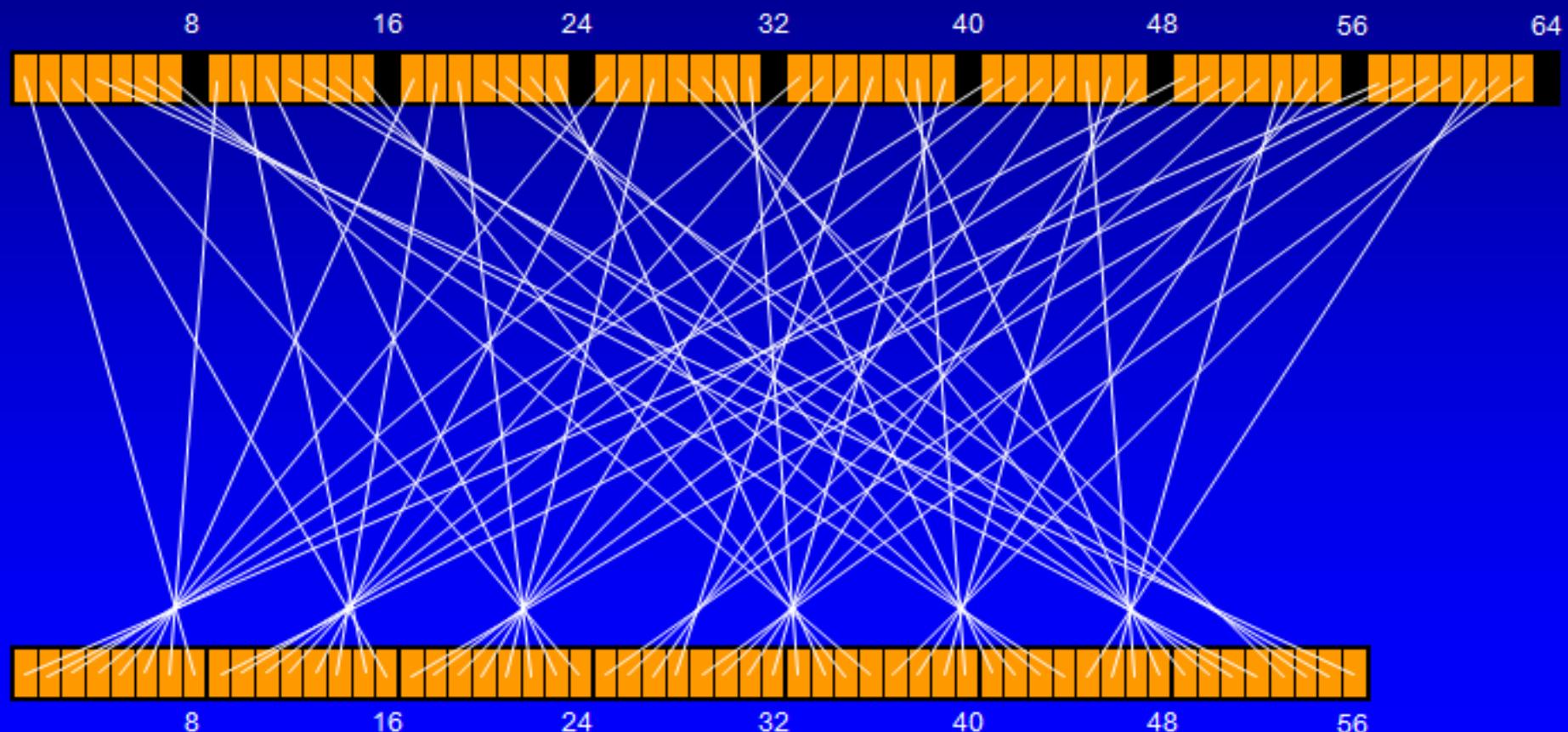
S_5	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
00	0010	1100	0100	0001	0111	1010	1011	0110	1000	0101	0011	1111	1101	0000	1110	1001
01	1110	1011	0010	1100	0100	0111	1101	0001	0101	0000	1111	1010	0011	1001	1000	0110
10	0100	0010	0001	1011	1010	1101	0111	1000	1111	1001	1100	0101	0110	0011	0000	1110
11	1011	1000	1100	0111	0001	1110	0010	1101	0110	1111	0000	1001	1010	0100	0101	0011



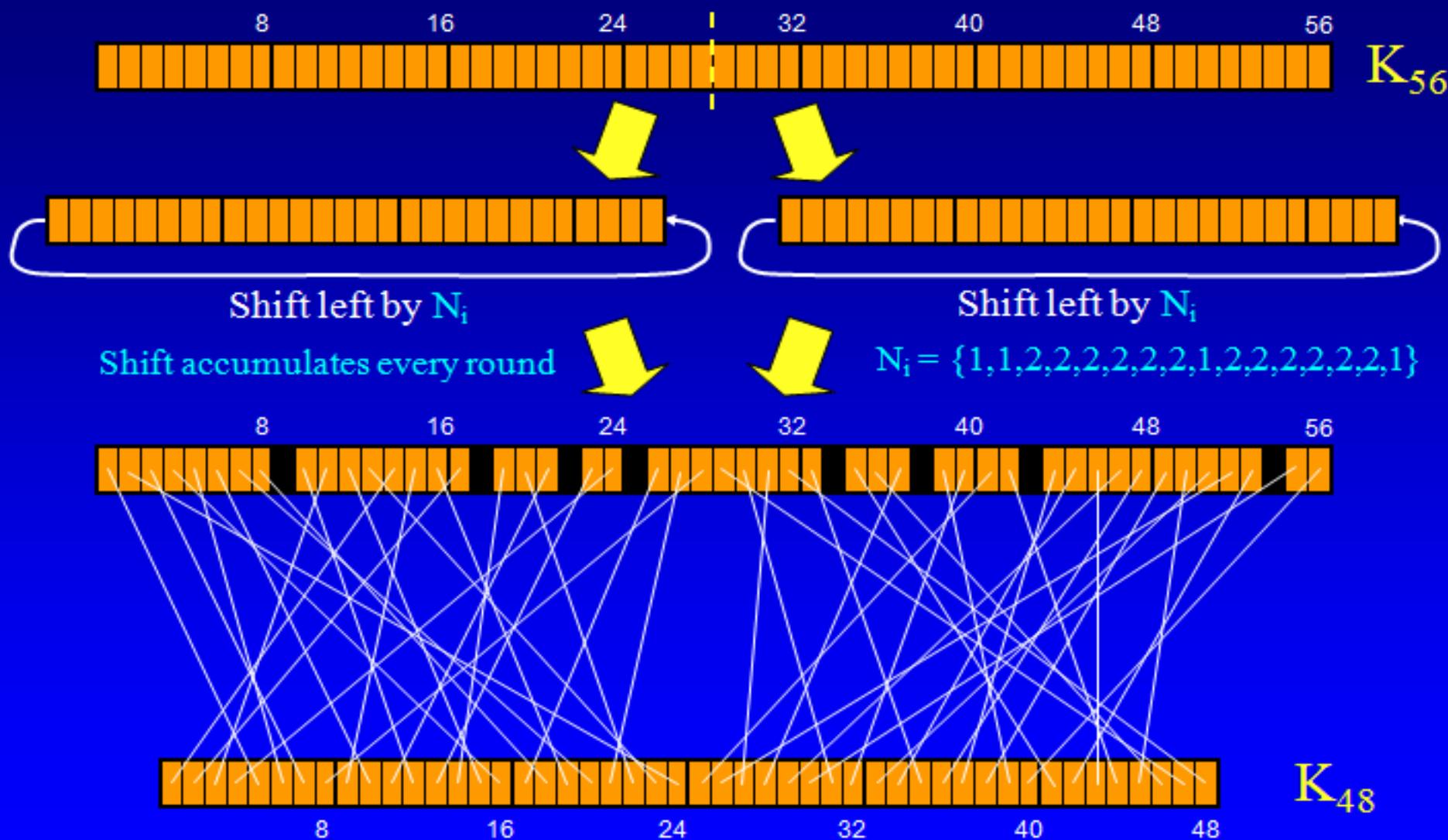
IP⁻¹ (Final Permutation):



Initial Key Permutation



Key Split & Shift & Compress



Avalanche Effect

- **key desirable property** of encryption algorithm
 - where a change of one input or key bit results in changing approx half output bits
 - making attempts to “home-in” by guessing keys impossible
- DES exhibits strong avalanche



Strength of DES – Key Size

- 56-bit keys have $2^{56} = 7.2 \times 10^{16}$ values
- brute force search looked hard
- advances have shown is possible
 - in 1997 on Internet in a few months
 - in 1998 on dedicated h/w (EFF) in a few days
 - in 1999 above combined in 22hrs!
 - still must be able to recognize plaintext
 - Otherwise, have no idea on the correct key
- Forced to consider alternatives to DES

Strength of DES – Analytic Attacks

- Currently, we have several analytic attacks on DES
 - utilize some deep structure of the cipher by gathering information about encryptions
 - can eventually recover some/all of the sub-key bits
 - if necessary then exhaustively search for the rest
- generally these are statistical attacks
 - differential cryptanalysis
 - linear cryptanalysis
 - related key attacks

Strength of DES – Timing Attacks

- attacks actual implementation of cipher
- use knowledge of consequences of implementation to derive information about some/all subkey bits
- specifically use fact that calculations can take varying times depending on the value of the inputs to it
- particularly problematic on smartcards

Differential Cryptanalysis

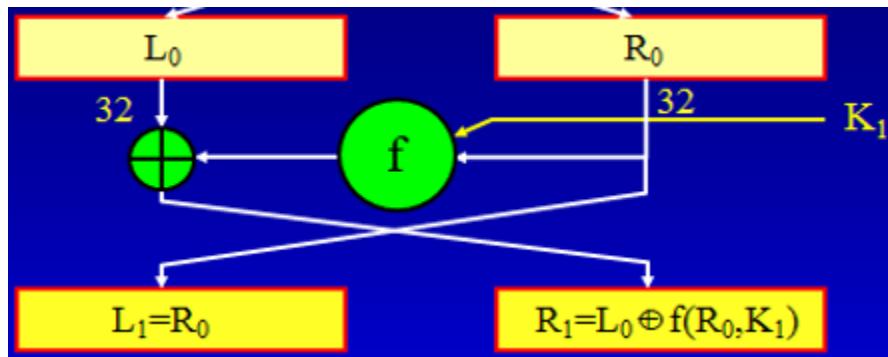
- one of the most significant recent (public) advances in cryptanalysis
- known by NSA in 70's cf DES design
- Murphy, Biham & Shamir published in 90's
- powerful method to analyse block ciphers
- used to analyze most current block ciphers with varying degrees of success
- DES reasonably resistant to it, cf Lucifer

Differential Cryptanalysis (2)

- a statistical attack against **Feistel ciphers**
- uses cipher structure not previously used
- design of **S-P** networks has output of function **f** influenced by both input & key
- hence cannot trace values back through cipher without knowing value of the key
- differential cryptanalysis compares two related pairs of encryptions (differential)

Differential Cryptanalysis (3)

- Differential cryptanalysis compares two related pairs of encryptions
- with known difference in the input $m_0 || m_1$
- searching for a known difference in output
- when same subkeys are used



$$\begin{aligned}\Delta m_{i+1} &= m_{i+1} \oplus m'_{i+1} \\ &= [m_{i-1} \oplus f(m_i, K_i)] \oplus [m'_{i-1} \oplus f(m'_i, K_i)] \\ &= \Delta m_{i+1} \oplus [f(m_i, K_i) \oplus f(m'_i, K_i)]\end{aligned}$$

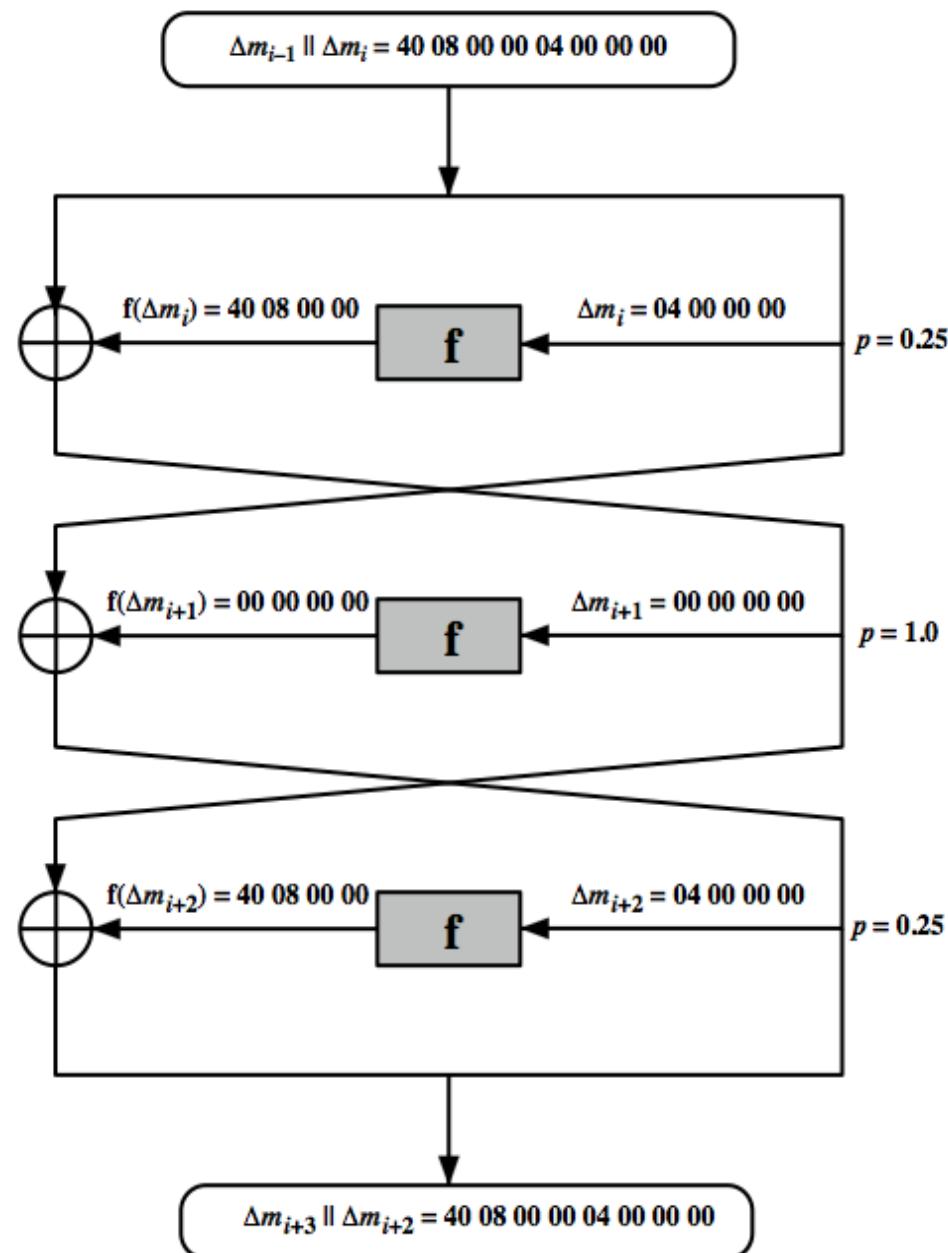
$$\begin{aligned}\Delta R_1 &= R_1 \oplus R'_1 \\ &= [L_0 \oplus f(R_0, K_1)] \oplus [L'_0 \oplus f(R'_0, K_1)] \\ &= \Delta L_0 \oplus [f(R_0, K_1) \oplus f(R'_0, K_1)]\end{aligned}$$

Differential Cryptanalysis (4)

- have some input difference giving some output difference with probability p
- if find instances of some higher probability input / output difference pairs occurring
- can infer subkey that was used in round
- then must iterate process over many rounds (with decreasing probabilities)
- This procedure must be repeated many times to determine all the key bits.

Differential Cryptanalysis (5)

Overall probability
of given output
difference is
 $0.25 \times 1.0 \times 0.25$
 $= 0.0625$



Linear Cryptanalysis

- another fairly recent development
- also a statistical method
- must be iterated over rounds, with decreasing probabilities
- developed by Matsui et al in early 90's
- based on finding linear approximations
- can attack DES with 2^{43} known plaintexts, easier but still in practice infeasible

Block Cipher Design Principles:

Number of Rounds

- The greater the number of rounds, the more difficult it is to perform cryptanalysis
- In general, the criterion should be that the number of rounds is chosen so that known cryptanalytic efforts require greater effort than a simple brute-force key search attack
- If DES had 15 or fewer rounds, differential cryptanalysis would require less effort than a brute-force key search

Block Cipher Design Principles: Design of Function F

- The heart of a Feistel block cipher is the function F
- The more nonlinear F, the more difficult any type of cryptanalysis will be
- The algorithm should have good avalanche properties
 - Strict avalanche criterion (SAC): States that any output bit j of an S-box should change with probability $1/2$ when any single input bit i is inverted for all i, j
 - Bit independence criterion (BIC) : States that output bits j and k should change independently when any single input bit i is inverted for all i, j, k
- The SAC and BIC criteria appear to strengthen the effectiveness of the confusion function

Block Cipher Design Principles: Key Schedule Algorithm

- With any Feistel block cipher, the key is used to generate one subkey for each round
- In general, we would like to select subkeys to maximize the difficulty of deducing individual subkeys and the difficulty of working back to the main key
- It is suggested that, at a minimum, the key schedule should guarantee key/ciphertext Strict Avalanche Criterion and Bit Independence Criterion

Advanced Encryption Standard (AES)

- NIST (National Institute of Standards and Technology) created a program for the development of Advanced Encryption Standard (AES) (first call Sept. 97)
- “Winner” – **Rijndael** announced Oct. 2000
- Rijndael (Daemen and Rijmen) supports keys of 128, 192, or 256 bits and messages of 128, 192, or 256 bits (AES uses only 128 bit blocks)
- Designed to be resistant to linear or differential cryptanalysis
- Fast and efficient in hardware and software implementations

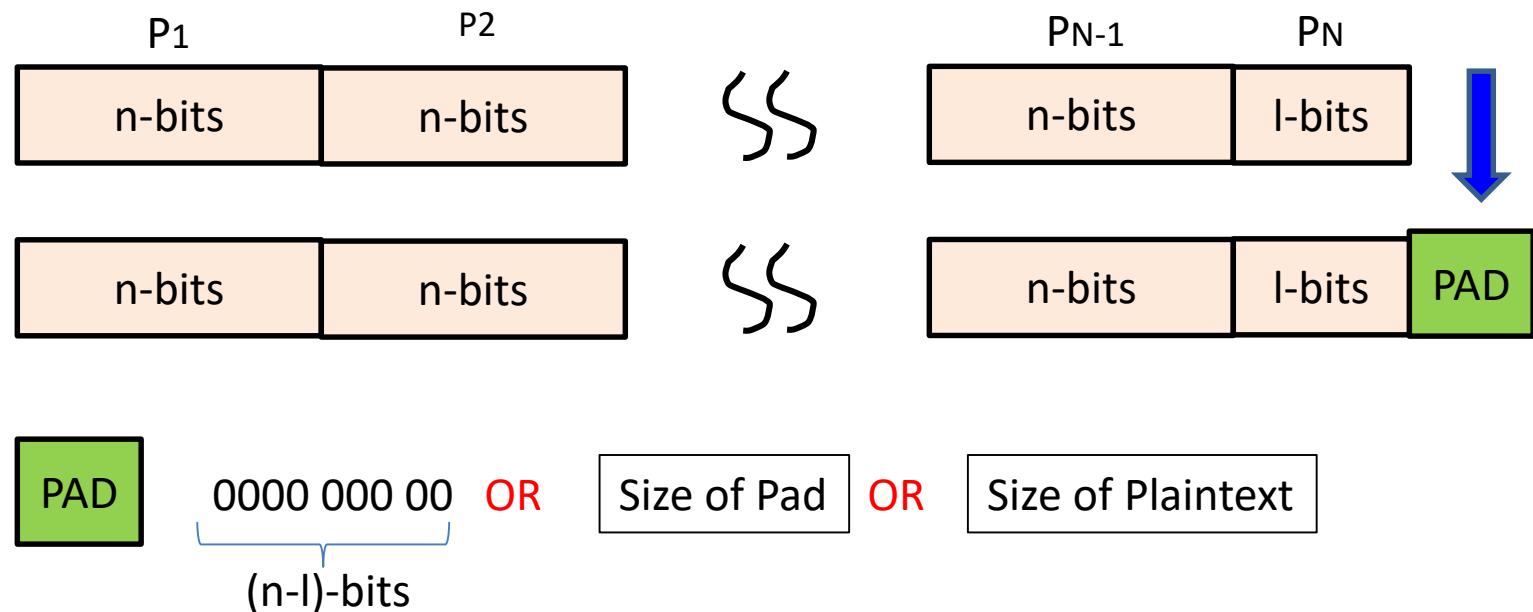


How to use a block cipher?

- Block ciphers encrypt fixed-size blocks
 - e.g. DES encrypts 64-bit blocks
- We need some way to encrypt a message of arbitrary length
 - e.g. a message of 1000 bytes
- NIST (National Institute of Standards and Technology) defines several ways to deal with it, called modes of operation
 - ECB (Electronic Code Book)
 - CBC (Cipher Block Chaining)

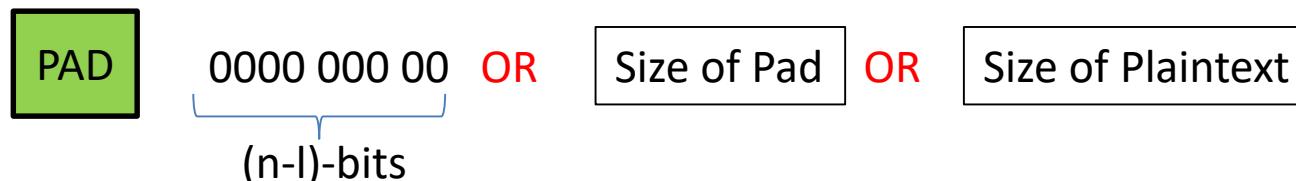
Message Padding

- The plaintext message is broken into blocks, P_1, P_2, P_3
- The last block may be short of a whole block and needs padding.

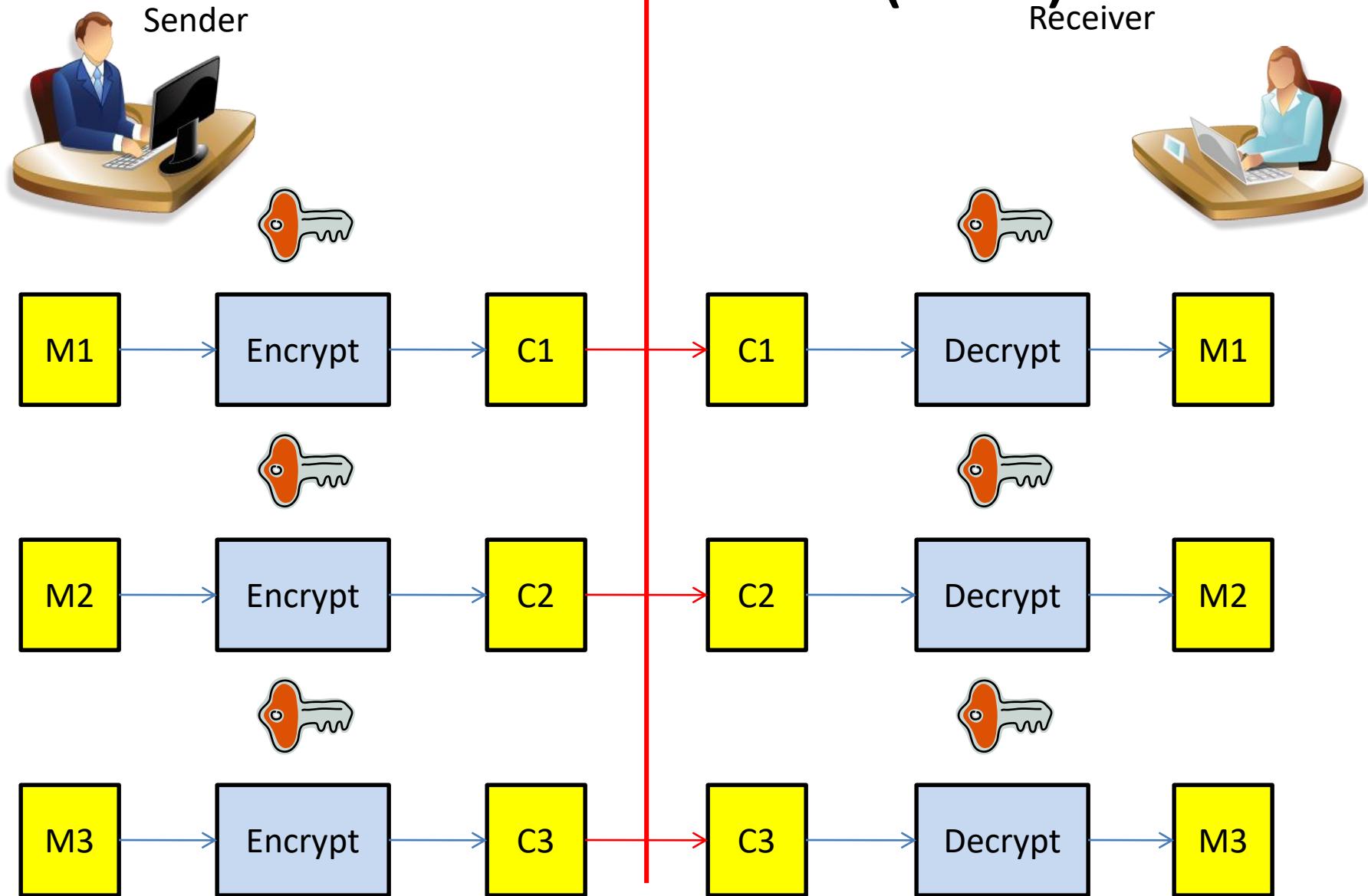


Message Padding (2)

- Possible padding:
 - Known non-data values (e.g. nulls)
 - Or a number indicating the size of the pad
 - Or a number indicating the size of the plaintext
 - The last two may require an extra block.



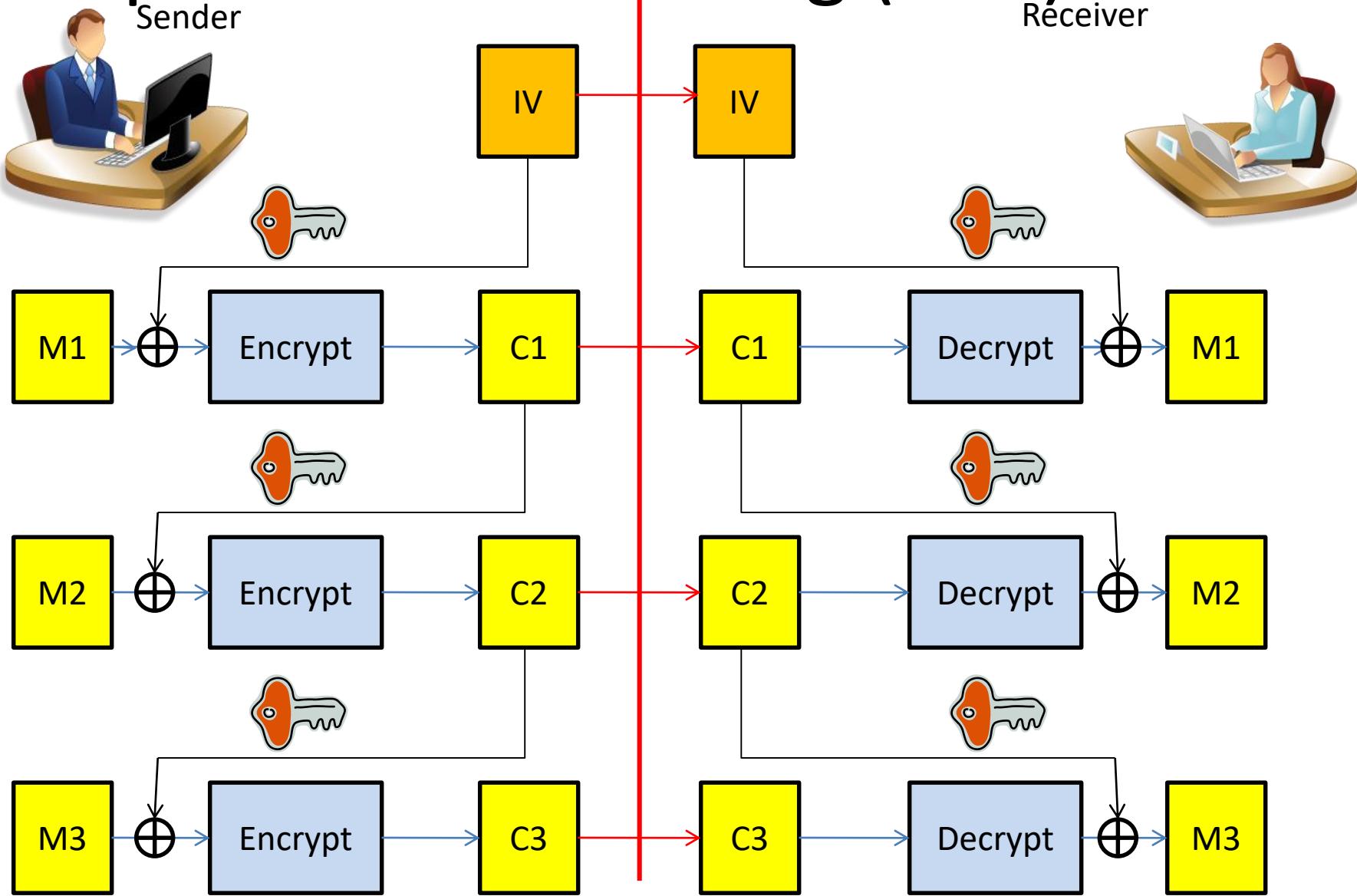
Electronic Code Book (ECB) Mode



Electronic Code Book (ECB) Mode (2)

- Cipher acts as simple block substitution determined by key
- For a given key, this mode behaves like we have a gigantic codebook, in which each plaintext block has an entry, hence the name Electronic Code Book
- Fast and simple but repeated input block creates repeated ciphertext block
- Vulnerable to replay attacks: if an attacker thinks block C_2 corresponds to \$ amount, then substitute another C_k
- Attacker can also build a codebook of $\langle C_k; \text{guessed } M_k \rangle$ pairs
- Application: secure transmission of short pieces of information (e.g. a temporary encryption key)

Cipher Block Chaining (CBC) Mode



Cipher Block Chaining (CBC) Mode (2)

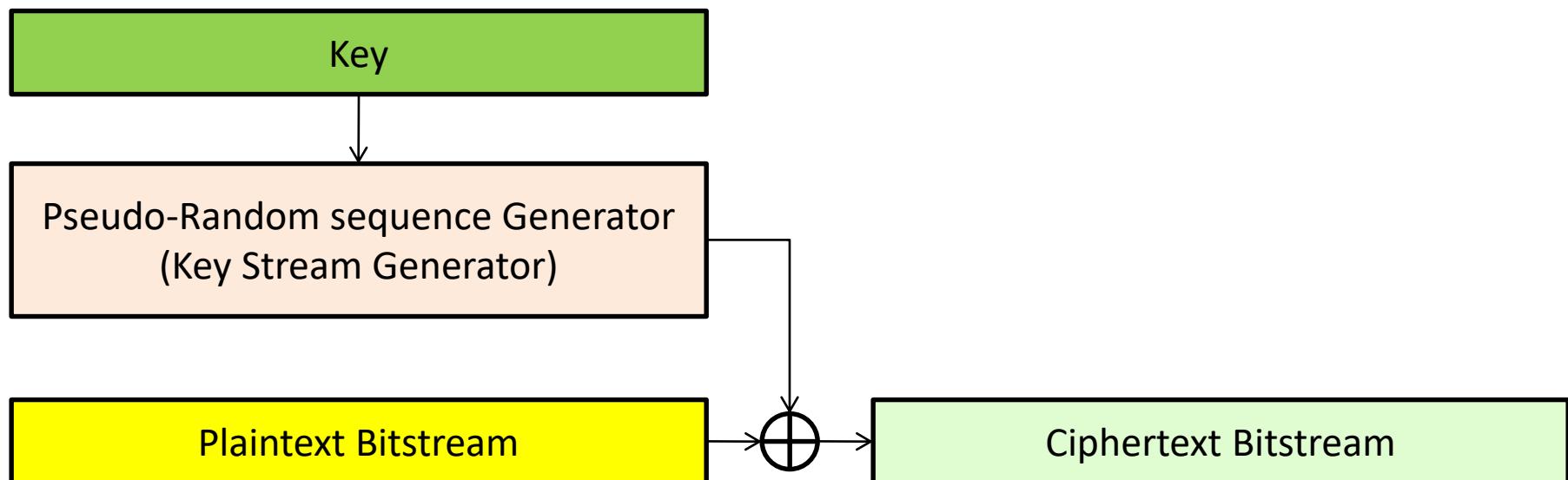
- The plaintext is broken into blocks M_1, M_2, M_3
- Each plaintext block is XORed (chained) with the previous ciphertext block before encryption (hence the name CBC)

$$C_i = E_k(C_{i-1} \oplus M_i); \quad C_0 = IV$$

- The encryption of a block depends on the current and all blocks before it. Then, the input plaintext $M_i = M_k$ will not result in the same output code due to memory-based chaining
- Use an Initial Vector (IV) (Use only once) to start the process
- Decryption: $M_i = C_{i-1} \oplus D_k(C_i)$
- Application: general block-oriented transmission.

Stream Ciphers

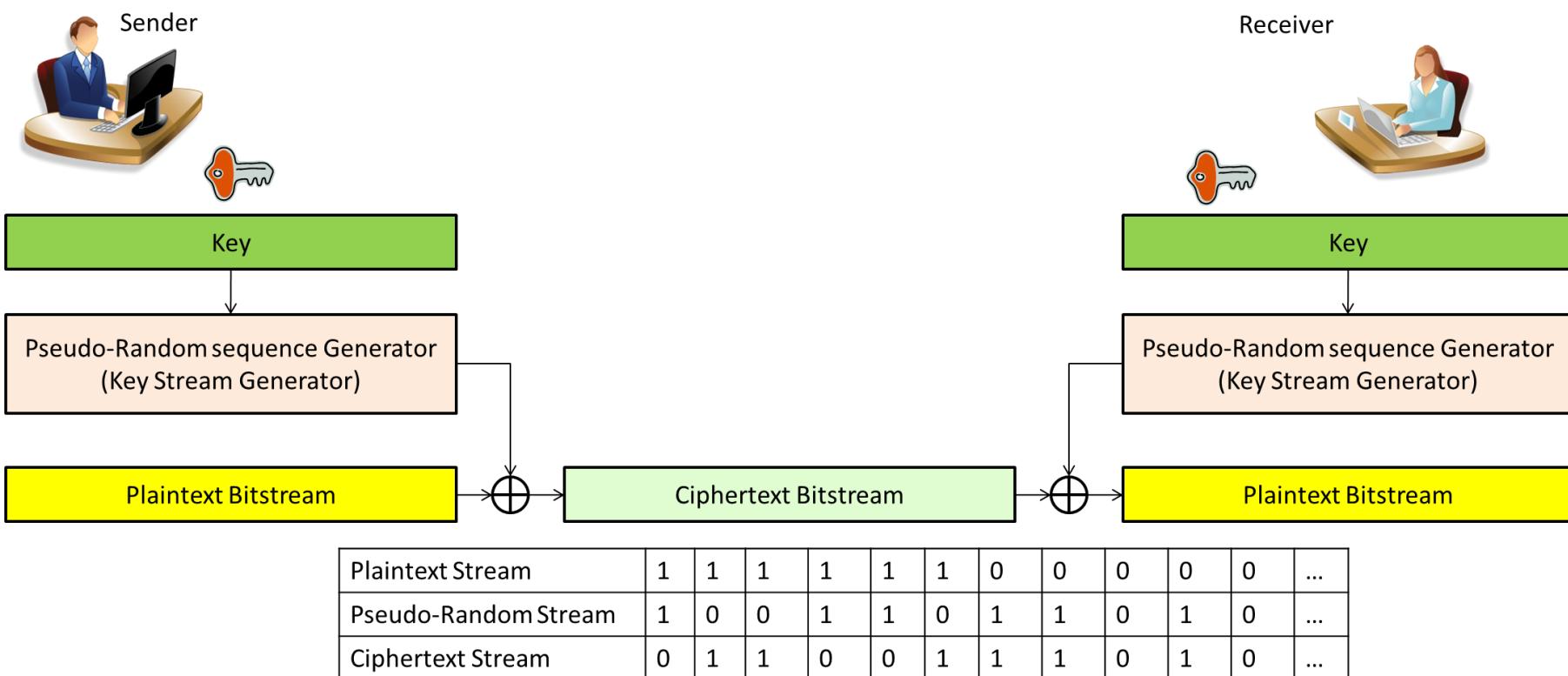
- Many times data is transmitted in serial form (one bit at a time)
- Cipher generates “Key-Stream” which is combined with “Message-Stream” to produce “Cipher-Stream”



Plaintext Stream	1	1	1	1	1	1	0	0	0	0	0	...
Pseudo-Random Stream	1	0	0	1	1	0	1	1	0	1	0	...
Ciphertext Stream	0	1	1	0	0	1	1	1	0	1	0	...

Stream Ciphers (2)

- Inverted at receiver by combining the same Key Stream
- Better than Block Ciphers for Serial Communication Channels
- Repeated input patterns do not produce repeated cipher stream sequences



Stream Ciphers (3)

- Only adds Confusion (no Diffusion)
- If Synchronization lost between sender and receiver (Cryptosync) – must resync
- “no Diffusion” → What shall we know about the key?

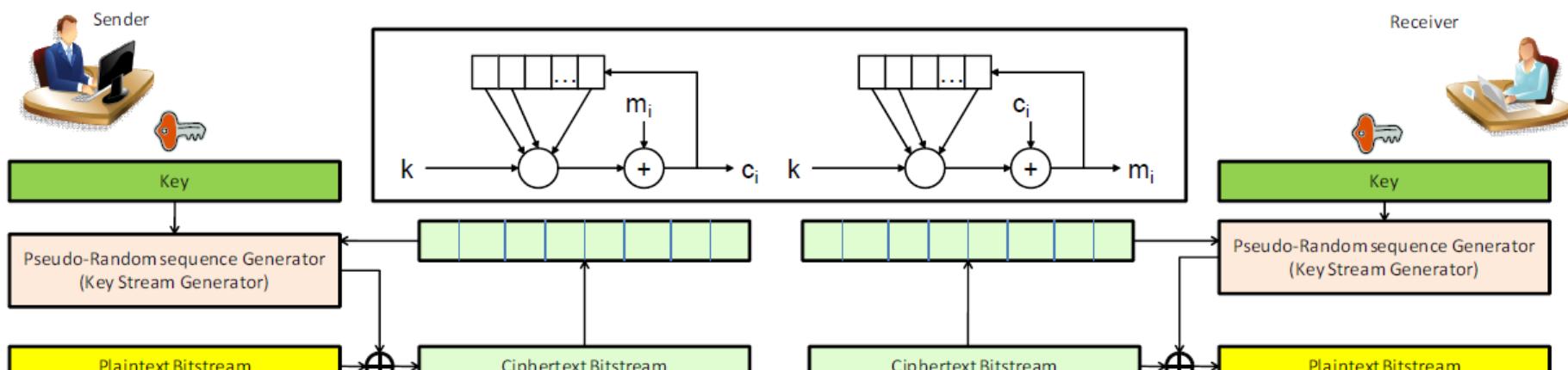


Dangers of Keystream Reuse in Stream Cipher

- If we have a random stream, why don't we just reuse it?
$$(M_1 \oplus K) \oplus (M_2 \oplus K) = M_1 \oplus M_2$$
- If M_1 or M_2 is known or discovered, then the other one will also be known.

Self-synchronous Stream Cipher

- Use ciphertext as feedback into the stream generation process. Key-stream is generated as a function of the key and a fixed number of previous ciphertext digits.
- If Cryptosync lost, resynchronization will be re-established once bad bits have passed through feedback register



Plaintext Stream	1	1	1	1	1	1	0	0	0	0	0	...
Pseudo-Random Stream	1	0	0	1	1	0	1	1	0	1	0	...
Ciphertext Stream	0	1	1	0	0	1	1	1	0	1	0	...

Thank
you



CS 6355/4355: Cryptanalysis and Database Security

Topic 3: Finite Fields and Number Theory

Lecturer: Rongxing LU

Email: RLU1@unb.ca Office: GE 114

Website: <http://www.cs.unb.ca/~rlu1/>

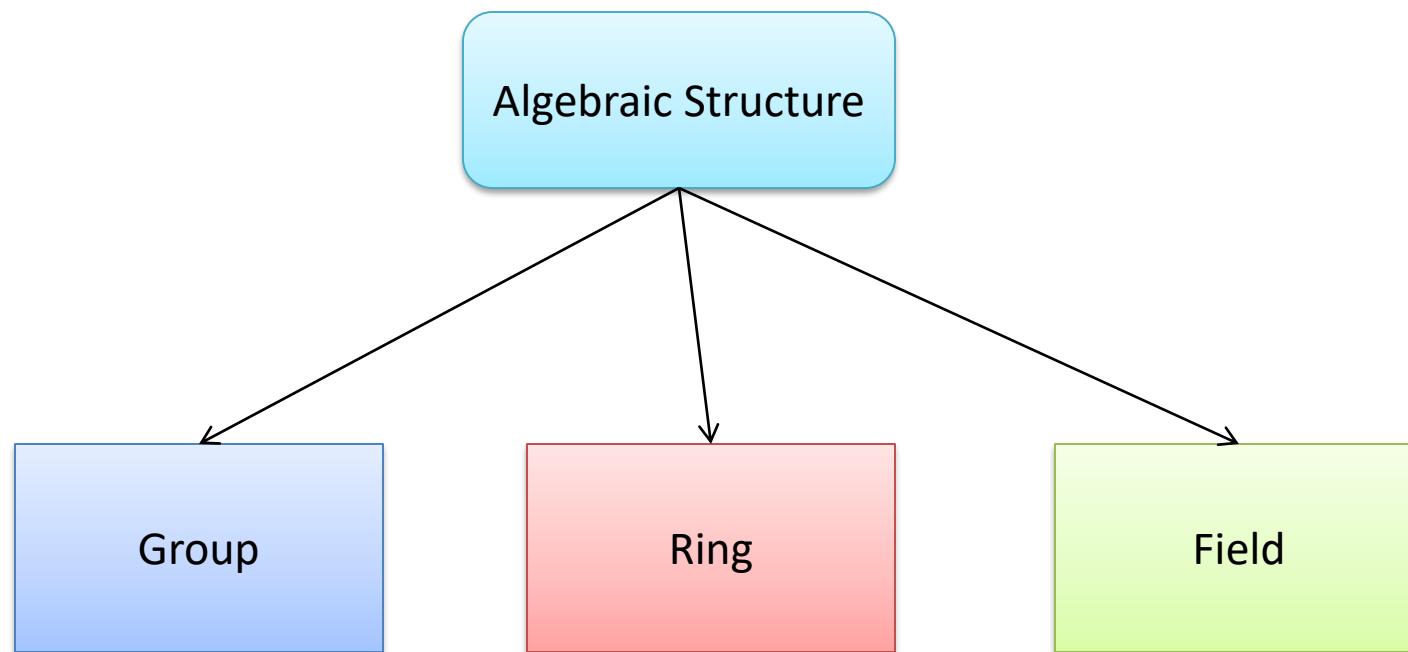
Faculty of Computer Science, University of New Brunswick

Mathematical Fundamentals

- Algebraic Structure (Group, Ring, and Finite Fields)
- Number Theory

Algebraic Structure

- Public key cryptography requires sets of integers and specific operations that are defined for those sets, including Group, Ring and Field.



Basic Modular Arithmetic

- define **modulo operator** “ $a \bmod n$ ” to be remainder when a is divided by n
- use the term **congruence** for: $a \equiv b \bmod n$
 - when divided by n , a and b have same remainder
 - eg. $100 \equiv 34 \bmod 11$
 - 1:00 and 13:00 hours are the same $13 \equiv 1 \bmod 12$
- b is called a **residue** of $a \bmod n$
 - since with integers can always write: $a = qn + b$
 - usually chose smallest positive remainder as residue
 - ie. $0 \leq b < n-1$
 - process is known as **modulo reduction**
 - eg. $-12 \bmod 7 = -5 \bmod 7 = 2 \bmod 7 = 9 \bmod 7$

Modulo 8 Addition Example

+ \ X	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7	0
2	2	3	4	5	6	7	0	1
3	3	4	5	6	7	0	1	2
4	4	5	6	7	0	1	2	3
5	5	6	7	0	1	2	3	4
6	6	7	0	1	2	3	4	5
7	7	0	1	2	3	4	5	6

↓
Y

→ X

Z=X+Y mod 8

Group

- A group (G) is a set of elements with an operation (\bullet) that satisfies four properties (or axioms). A commutative group satisfies an extra property, commutativity:
 - **Closure:** with some operation (\bullet) whose result is also in the set
 - **Associativity:** $(a.b).c = a.(b.c)$
 - **Existence of identity:** $e: e.a = a.e = a$
 - **Existence of inverse:** $a^{-1}: a.a^{-1} = e$
 - **Commutativity:** $a.b = b.a$
 - \Rightarrow forms an abelian (commutative) group

Group (cont.)

- A group (G, \bullet) involves a single operation, the properties imposed on the operation allow the use of a pair of operations as long as they are inverses of each other.
- Example 1.
 - The set of residue integers with the addition operator, $G = \langle Z_n, + \rangle$, is a commutative group. We can perform addition and subtraction on the elements of this set without moving out of the set.
 - $Z_8 = \{0, 1, 2, 3, 4, 5, 6, 7\}$, $Z_3 = \{0, 1, 2\}$

+	0	1	2
0	0	1	2
1	1	2	0
2	2	0	1

Z_3

Group (cont.)

+	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7	0
2	2	3	4	5	6	7	0	1
3	3	4	5	6	7	0	1	2
4	4	5	6	7	0	1	2	3
5	5	6	7	0	1	2	3	4
6	6	7	0	1	2	3	4	5
7	7	0	1	2	3	4	5	6

 Z_8

Group (cont.)

- **Example 2.**

- The set Z_n^* with the multiplication operator, $G = \langle Z_n^*, \times \rangle$, is also an abelian group.
- $Z_7^* = \{1, 2, 3, 4, 5, 6\}$, $Z_3^* = \{1, 2\}$, $Z_8^* ?$

\times	1	2	3	4	5	6
1	1	2	3	4	5	6
2	2	4	6	1	3	5
3	3	6	2	5	1	4
4	4	1	5	2	6	3
5	5	3	1	6	4	2
6	6	5	4	3	2	1

\times	1	2
1	1	2
2	2	1

Z_3^*

$Z_8^* = \{1, 3, 5, 7\}$

Since 2, 4, 6 have no inverses in Z_8^*

Z_7^*

Group (cont.)

- **Example 3.**

- Let us define a set $G = \langle \{a, b, c, d\}, \bullet \rangle$ and the operation as shown in

•	a	b	c	d
a	a	b	c	d
b	b	c	d	a
c	c	d	a	b
d	d	a	b	c

- It is an **abelian group**
 - Closure, Associativity, Existence of identity, Existence of inverse, Commutativity

Discussion

- Check whether the following sets can form group under the given operation?
- Case 1: the set of real numbers R, for the operation $a^{\circ}b = 2(a + b)$

Case 2

- $G=\{1, -1\}$, for the ordinary multiplication operation.

Case 3

- Non-Zero Real Number Set R^* , for operation
 $a^\circ b = 2ab$

Case 4

- Let $G=\{(a,b) \mid a, b \text{ are real numbers and } a \neq 0\}$, for the operation $(a,b)^o(c,d)=(ac,ad+b)$.

Case 4...

Exercise

- Let $G=\{e, a, b\}$, the operation is defined as

	e	a	b
e	e	a	b
a	a	b	e
b	b	e	a

- Prove G is a group for the operation.

Finite Group

- Finite Group: A group having a finite number of elements.
 - Infinite Group: A group having an infinite number of elements. Some infinite groups, such as the integers or rationals.
- Finite Groups, such as \mathbb{Z}_7^* , \mathbb{Z}_3^* , and $G = < \{a, b, c, d\}, \bullet >$
- The number of elements in a group
is called the **order** of the group.

•	a	b	c	d
a	a	b	c	d
b	b	c	d	a
c	c	d	a	b
d	d	a	b	c

Subgroup

- A subset H of $\langle G, * \rangle$ is called a subgroup of G if H also forms a group under the same operation *. More precisely, H is a subgroup of G if the restriction of $*$ to $H \times H$ is a group operation on H . This is usually represented by $H \leq G$, read as "H is a subgroup of G".
- Trivial subgroup $G \leq G$
- Nontrivial subgroups:
 $J=\{0,4\}$ and $H=\{0,2,4,6\}$,
where J is also a
subgroup of H
- **operation:**
addition modulo 8

+	0	2	4	6	1	3	5	7
0	0	2	4	6	1	3	5	7
2	2	4	6	0	3	5	7	1
4	4	6	0	2	5	7	1	3
6	6	0	2	4	7	1	3	5
1	1	3	5	7	2	4	6	0
3	3	5	7	1	4	6	0	2
5	5	7	1	3	6	0	2	4
7	7	1	3	5	0	2	4	6

Z_8

Subgroup (Cont.)

- Example.
 - Is the group $H = \langle \mathbb{Z}_{10}, + \rangle$ a subgroup of the group $G = \langle \mathbb{Z}_{12}, + \rangle$?

Cyclic Subgroups

- Cyclic subgroups
 - If a subgroup of a group can be generated using the power of an element, the subgroup is called the **cyclic subgroup**.

$$a^n \rightarrow \underbrace{a \bullet a \bullet \cdots \bullet a}_{n-times}$$

Cyclic Subgroups (Cont.)

- **Example.**

➤ Four cyclic subgroups can be made from the group $G = \langle Z_6, + \rangle$. They are $H_1 = \langle \{0\}, + \rangle$, $H_2 = \langle \{0, 2, 4\}, + \rangle$, $H_3 = \langle \{0, 3\}, + \rangle$, and $H_4 = G$.

$$0^0 \bmod 6 = 0$$

$$2^0 \bmod 6 = 0$$

$$4^0 \bmod 6 = 0$$

$$2^1 \bmod 6 = 2$$

$$4^1 \bmod 6 = 4$$

$$2^2 \bmod 6 = (2 + 2) \bmod 6 = 4$$

$$4^2 \bmod 6 = (4 + 4) \bmod 6 = 2$$

$$3^0 \bmod 6 = 0$$

$$3^1 \bmod 6 = 3$$

$$1^0 \bmod 6 = 0$$

$$1^1 \bmod 6 = 1$$

$$1^2 \bmod 6 = (1 + 1) \bmod 6 = 2$$

$$1^3 \bmod 6 = (1 + 1 + 1) \bmod 6 = 3$$

$$1^4 \bmod 6 = (1 + 1 + 1 + 1) \bmod 6 = 4$$

$$1^5 \bmod 6 = (1 + 1 + 1 + 1 + 1) \bmod 6 = 5$$

$$5^0 \bmod 6 = 0$$

$$5^1 \bmod 6 = 5$$

$$5^2 \bmod 6 = 4$$

$$5^3 \bmod 6 = 3$$

$$5^4 \bmod 6 = 2$$

$$5^5 \bmod 6 = 1$$

Cyclic Subgroups (Cont.)

- Example.
 - Three cyclic subgroups can be made from the group $G = \langle \mathbb{Z}_{10}^*, \times \rangle$.

Cyclic Groups

- Cyclic Group.
 - A cyclic group is a group that is its own cyclic subgroup.

$$\{e, g, g^2, \dots, g^{n-1}\}, \text{ where } \sim g^n = e$$

Cyclic Groups

- Cyclic Group.
 - A cyclic group is a group that is its own cyclic subgroup.
 - Example. $\{e, g, g^2, \dots, g^{n-1}\}$, where $\sim g^n = e$
 - Three cyclic subgroups can be made from the group $G = \langle Z_{10}^*, \times \rangle$. G has only four elements: 1, 3, 7, and 9. The cyclic subgroups are $H_1 = \langle \{1\}, \times \rangle$, $H_2 = \langle \{1, 9\}, \times \rangle$, and $H_3 = G$.
 - The group $G = \langle Z_{10}^*, \times \rangle$ is a cyclic group with two generators, $g = 3$ and $g = 7$.

x	1	3	7	9
1	1	3	7	9
3	3	9	1	7
7	7	1	9	3
9	9	7	3	1

$7^0 \bmod 10 = 1$	$3^0 \bmod 10 = 1$
$7^1 \bmod 10 = 7$	$3^1 \bmod 10 = 3$
$7^2 \bmod 10 = 9$	$3^2 \bmod 10 = 9$
$7^3 \bmod 10 = 3$	$3^3 \bmod 10 = 7$

Cyclic Groups (Cont.)

- Cyclic Group.

➤ Example.

□ The group $G = \langle \mathbb{Z}_6, + \rangle$ is a cyclic group with two generators, $g = 1$ and $g = 5$.

+	0	1	2	3	4	5
0	0	1	2	3	4	5
1	1	2	3	4	5	0
2	2	3	4	5	0	1
3	3	4	5	0	1	2
4	4	5	0	1	2	3
5	5	0	1	2	3	4

	$g = 1$	$g = 5$
g	1	5
$g + g$	2	4
$g + g + g$	3	3
$g + g + g + g$	4	2
$g + g + g + g + g$	5	1
$g + g + g + g + g + g$	0	0

Lagrange's Theorem

- Assume that G is a group, and H is a subgroup of G . If the orders of G and H are $|G|$ and $|H|$, respectively, then, based on Lagrange's theorem, $|H|$ divides $|G|$.
- Example.**
 - Four cyclic subgroups can be made from the group $G = \langle Z_6, + \rangle$. They are $H_1 = \langle \{0\}, + \rangle$, $H_2 = \langle \{0, 2, 4\}, + \rangle$, $H_3 = \langle \{0, 3\}, + \rangle$, and $H_4 = G$.
- Order of an Element**
 - The order of an element is the order of the cyclic group it generates.

$$0^0 \bmod 6 = 0$$

$$2^0 \bmod 6 = 0$$

$$4^0 \bmod 6 = 0$$

$$2^1 \bmod 6 = 2$$

$$4^1 \bmod 6 = 4$$

$$2^2 \bmod 6 = (2 + 2) \bmod 6 = 4$$

$$4^2 \bmod 6 = (4 + 4) \bmod 6 = 2$$

$$3^0 \bmod 6 = 0$$

$$3^1 \bmod 6 = 3$$

$$1^0 \bmod 6 = 0$$

$$1^1 \bmod 6 = 1$$

$$1^2 \bmod 6 = (1 + 1) \bmod 6 = 2$$

$$1^3 \bmod 6 = (1 + 1 + 1) \bmod 6 = 3$$

$$1^4 \bmod 6 = (1 + 1 + 1 + 1) \bmod 6 = 4$$

$$1^5 \bmod 6 = (1 + 1 + 1 + 1 + 1) \bmod 6 = 5$$

$$5^0 \bmod 6 = 0$$

$$5^1 \bmod 6 = 5$$

$$5^2 \bmod 6 = 4$$

$$5^3 \bmod 6 = 3$$

$$5^4 \bmod 6 = 2$$

$$5^5 \bmod 6 = 1$$

Ring

- A ring, $R = \langle \{a,b,c,\dots\}, \bullet, \blacksquare \rangle$, is an algebraic structure with two operations.

Set	Operation \bullet	Operation \blacksquare
$\{a,b,c,\dots\}$	<ol style="list-style-type: none">1. Closure2. Associativity3. Commutativity4. Existence of identity5. Existence of inverse	<ol style="list-style-type: none">1. Closure2. Associativity3. Commutativity (The third property is only satisfied for a commutative ring)

distributivity

$$a \blacksquare (b \bullet c) = a \blacksquare b + a \blacksquare c , \quad (b \bullet c) \blacksquare a = b \blacksquare c + c \blacksquare a$$

Ring (Cont.)

- The set \mathbb{Z} with two operations, addition and multiplication, is a commutative ring. We show it by $R = \langle \mathbb{Z}, +, \times \rangle$.
Addition satisfies all of the five properties; multiplication satisfies only three properties.
- $\mathbb{Z} = \{\dots, -2, -1, 0, 1, 2, \dots\}$

+	...	-2	-1	0	1	...
...
-2	...	-4	-3	-2	-1	...
-1	...	-3	-2	-1	0	...
0	...	-2	-1	0	1	...
1	...	-1	0	1	2	...
...

\times	...	-2	-1	0	1	...
...
-2	...	4	2	0	-2	...
-1	...	2	1	0	-1	...
0	...	0	0	0	0	...
1	...	-2	-1	0	1	...
...

Example

- Let R be a ring with identity (denoted as 1). Prove R is also a ring with identity under the operations $a \oplus b = a + b - 1$, $a \circ b = a + b - ab$

Field

- A field, denoted by $F = \langle \{a,b,c,\dots\}, \bullet, \blacksquare \rangle$, is a commutative ring in which the second operation satisfies all five properties defined for the first operation except that the identity of the first operation has no inverse.

Set	Operation \bullet	Operation \blacksquare
$\{a,b,c,\dots\}$	<ol style="list-style-type: none">1. Closure2. Associativity3. Commutativity4. Existence of identity5. Existence of inverse	<ol style="list-style-type: none">1. Closure2. Associativity3. Commutativity4. Existence of identity5. Existence of inverse*

* The identity element of the first operation has no inverse with respect to the second operation

Finite Fields

- Galois showed that for a field to be finite, the number of elements should be p^n , where p is a prime and n is a positive integer.
- A Galois field, $\text{GF}(p^n)$, is a finite field with p^n elements.
- When $n = 1$, we have $\text{GF}(p)$ field. This field can be the set \mathbb{Z}_p , $\{0, 1, \dots, p - 1\}$, with two arithmetic operations.
- A very common field in this category is $\text{GF}(2)$ with the set $\{0, 1\}$ and two operations, addition and multiplication.

$\text{GF}(2)$	
$\{0, 1\}$	$(+, \times)$

+	0	1
0	0	1
1	1	0

x	0	1
0	0	0
1	0	1

a	0	1
-a	0	1

a	0	1
a^{-1}	--	1

Finite Fields (Cont.)

- We can define GF(5) on the set \mathbb{Z}_5 (5 is a prime) with addition and multiplication operators

GF(5)	
{0, 1, 2, 3, 4}	(+, \times)

+	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

x	0	1	2	3	4
0	0	0	0	0	0
1	0	1	2	3	4
2	0	2	4	1	3
3	0	3	1	4	2
4	0	4	3	2	1

a	0	1	2	3	4
-a	0	4	3	2	1
a	0	1	2	3	4
a^{-1}	--	1	3	2	4

Finite Fields (Cont.)

Arithmetic in $GF(2^3)$ -- Addition

		000	001	010	011	100	101	110	111
+		0	1	2	3	4	5	6	7
000	0	0	1	2	3	4	5	6	7
001	1	1	0	3	2	5	4	7	6
010	2	2	3	0	1	6	7	4	5
011	3	3	2	1	0	7	6	5	4
100	4	4	5	6	7	0	1	2	3
101	5	5	4	7	6	1	0	3	2
110	6	6	7	4	5	2	3	0	1
111	7	7	6	5	4	3	2	1	0

(a) Addition

Finite Fields (Cont.)

Arithmetic in $GF(2^3)$ - Multiplication

	000	001	010	011	100	101	110	111
x	0	1	2	3	4	5	6	7
000	0	0	0	0	0	0	0	0
001	1	0	1	2	3	4	5	6
010	2	0	2	4	6	3	1	7
011	3	0	3	6	5	7	4	1
100	4	0	4	3	7	6	2	5
101	5	0	5	1	4	2	7	3
110	6	0	6	7	1	5	3	2
111	7	0	7	5	2	1	6	4

Finite Fields (Cont.)

Arithmetic in $\text{GF}(2^3)$ – Identity, Inverse

w	0	1	2	3	4	5	6	7
$-w$	0	1	2	3	4	5	6	7
w^{-1}	---	1	5	6	7	2	3	4

Finite Fields (Cont.) $GF(2^3)$

Polynomial Arithmetic Modulo $(x^3 + x + 1)$

		000	001	010	011	100	101	110	111
		0	1	x	$x + 1$	x^2	$x^2 + 1$	$x^2 + x$	$x^2 + x + 1$
+		0	1	x	$x + 1$	x^2	$x^2 + 1$	$x^2 + x$	$x^2 + x + 1$
000	0	0	1	x	$x + 1$	x^2	$x^2 + 1$	$x^2 + x$	$x^2 + x + 1$
	1	1	0	$x + 1$	x	$x^2 + 1$	x^2	$x^2 + x + 1$	$x^2 + x$
010	x	x	$x + 1$	0	1	$x^2 + x$	$x^2 + x + 1$	x^2	$x^2 + 1$
	$x + 1$	$x + 1$	x	1	0	$x^2 + x + 1$	$x^2 + x$	$x^2 + 1$	x^2
x^2	x^2	$x^2 + 1$	$x^2 + x$	$x^2 + x + 1$	0	1	x	$x + 1$	
	$x^2 + 1$	x^2	$x^2 + x + 1$	$x^2 + x$	1	0	$x + 1$	x	
$x^2 + x$	$x^2 + x$	$x^2 + x + 1$	x^2	$x^2 + 1$	x	$x + 1$	0		1
	$x^2 + x + 1$	$x^2 + x$	$x^2 + 1$	x^2	$x + 1$	x	1		0

(a) Addition

		000	001	010	011	100	101	110	111
		0	1	x	$x + 1$	x^2	$x^2 + 1$	$x^2 + x$	$x^2 + x + 1$
×		0	0	0	0	0	0	0	0
000	0	0	0	0	0	0	0	0	0
	1	0	1	x	$x + 1$	x^2	$x^2 + 1$	$x^2 + x$	$x^2 + x + 1$
x	x	0	x	x^2	$x^2 + x$	$x + 1$	1	$x^2 + x + 1$	$x^2 + 1$
	$x + 1$	0	$x + 1$	$x^2 + x$	$x^2 + 1$	$x^2 + x + 1$	x^2	1	x
x^2	x^2	0	x^2	$x + 1$	$x^2 + x + 1$	$x^2 + x$	x	$x^2 + 1$	1
	$x^2 + 1$	0	$x^2 + 1$	1	x^2	x	$x^2 + x + 1$	$x + 1$	$x^2 + x$
$x^2 + x$	$x^2 + x$	0	$x^2 + x$	$x^2 + x + 1$	1	$x^2 + 1$	$x + 1$	x	x^2
	$x^2 + x + 1$	0	$x^2 + x + 1$	$x^2 + 1$	x	1	$x^2 + x$	x^2	$x + 1$

(b) Multiplication

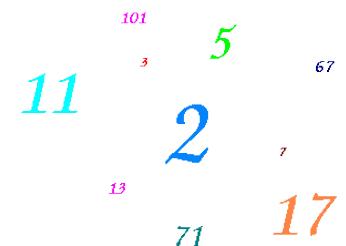
Group, Ring, Field

Algebraic Structure	Supported Typical Operations	Supported Typical Sets of Integers
Group	$(+, -)$ or (\times, \div)	\mathbb{Z}_n or \mathbb{Z}_n^*
Ring	$(+, -)$ and (\times)	\mathbb{Z}
Field	$(+, -)$ and (\times, \div)	\mathbb{Z}_p

Number Theory

- Fundamental Number Theorem
- GCD, Euclid's algorithm
- Extended Euclid's Algorithm
- Modular Arithmetic
- Euler's Totient Function
- Fermat Theorem
- Euler's Theorem

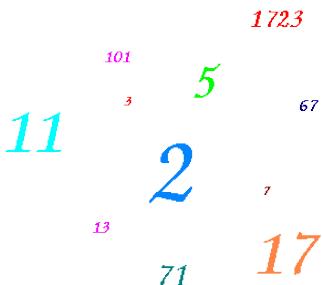
1723



Prime Numbers

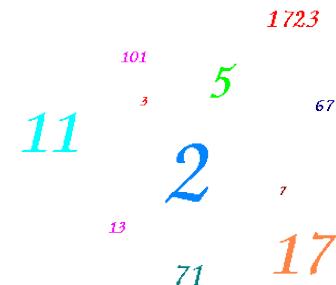
- prime numbers only have divisors of 1 and self
- they cannot be written as a product of other numbers
- note: 1 is prime, but is generally not of interest
- eg. 2,3,5,7 are prime, 4,6,8,9,10 are not
- prime numbers are central to number theory
- list of prime number less than 200 is:

2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83
89 97 101 103 107 109 113 127 131 137 139 149 151 157 163 167
173 179 181 191 193 197 199



Fundamental Theorem of Arithmetic

- All numbers can be expressed as a unique products of primes
 - $10 = 2 * 5$, $20 = 2 * 2 * 5$, $60 = 2 * 2 * 3 * 5$
- Proof in two parts
 - 1. All numbers are expressible as products of primes
 - 2. There is only one such product sequence per number



Fundamental Theorem of Arithmetic

- First part of proof
 - All numbers are products of primes

Let $S = \{x \mid x \text{ is not expressible as a product of primes}\}$

Let $c = \min\{S\}$. c cannot be prime

Let $c = c_1 \cdot c_2$

$c_1, c_2 < c \Rightarrow c_1, c_2 \notin S$ (because c is $\min\{S\}$)

$\therefore c_1, c_2$ are products of primes $\Rightarrow c$ is too

$\therefore S$ is an empty set

1723

11 2

71

17

13

67

101

3

5

7

Fundamental Theorem of Arithmetic

- Second part of proof
 - The product of primes is unique

Let $n = p_1 p_2 p_3 p_4 \dots = q_1 q_2 q_3 q_4 \dots$

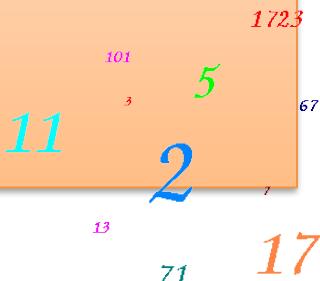
Cancel common primes. Now unique primes on both sides

Now, $p_1 \mid p_1 p_2 p_3 p_4$ “|” divide

$\Rightarrow p_1 \mid q_1 q_2 q_3 q_4 \dots$

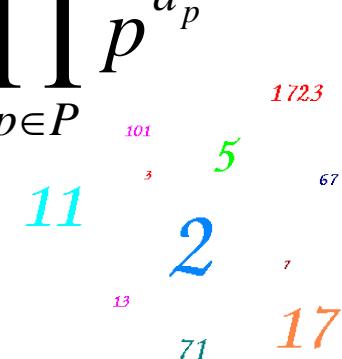
$\Rightarrow p_1 \mid \text{one of } q_1, q_2, q_3, q_4 \dots$

$\Rightarrow p_1 = q_i$ which is a contradiction



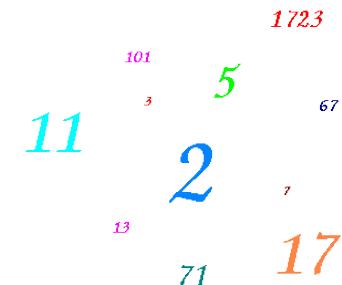
Prime Factorization

- to **factor** a number n is to write it as a product of other numbers: $n=a \times b \times c$
- note that factoring a number is relatively hard compared to multiplying the factors together to generate the number
- the **prime factorisation** of a number n is when its written as a product of primes $a = \prod_{p \in P} p^{a_p}$
 - eg. $91=7 \times 13$; $3600=2^4 \times 3^2 \times 5^2$

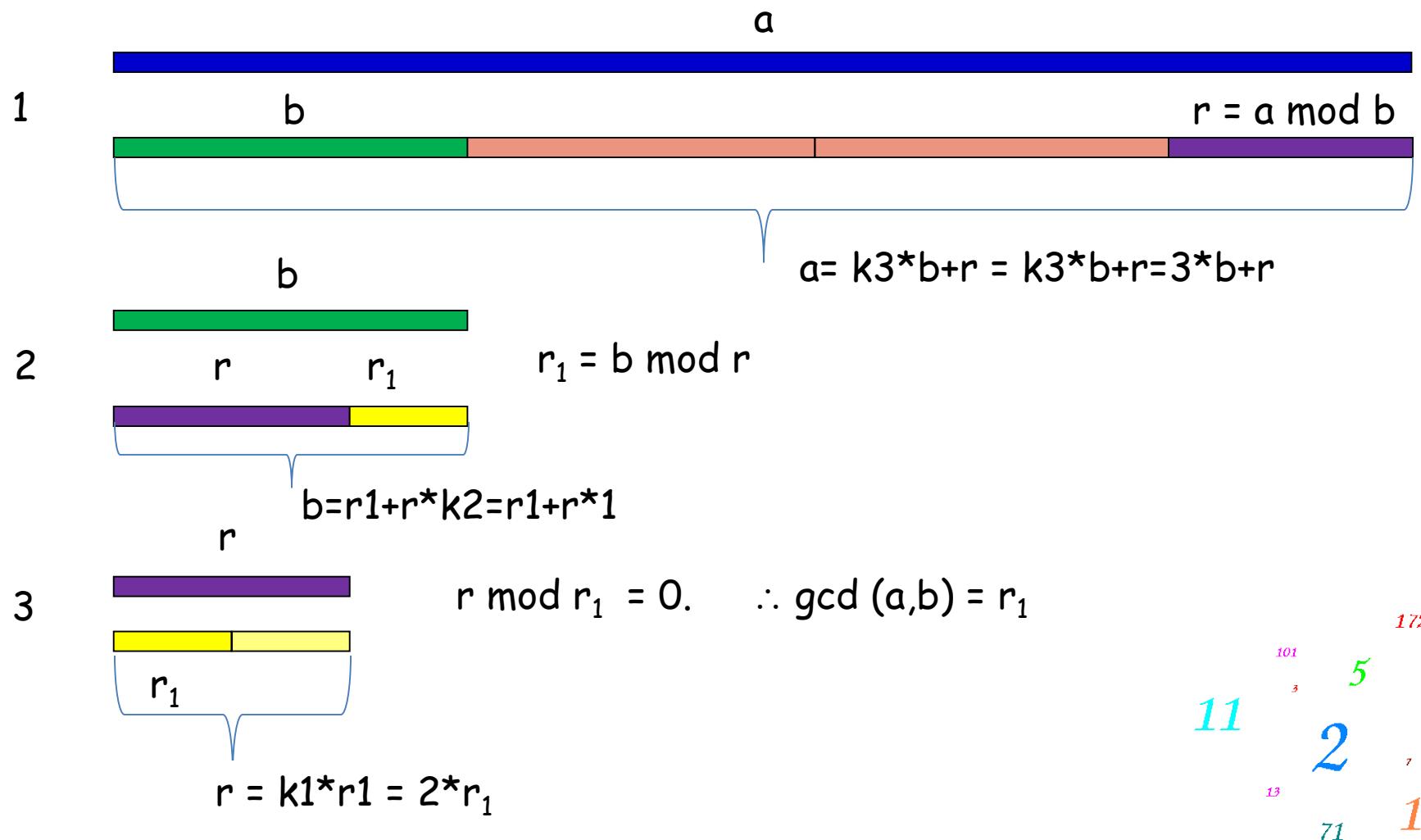


Relatively Prime Numbers & GCD

- two numbers a, b are **relatively prime** if have **no common divisors** apart from 1
 - eg. 8 & 15 are relatively prime since factors of 8 are 1,2,4,8 and of 15 are 1,3,5,15 and 1 is the only common factor
- conversely can determine the greatest common divisor (GCD) by comparing their prime factorizations and using least powers
 - eg. $300=2^1 \times 3^1 \times 5^2$ $18=2^1 \times 3^2$ hence $\text{GCD}(18,300)=2^1 \times 3^1 \times 5^0=6$



GCD(a,b)- Euclid's algorithm



Euclid's algorithm proof

- Proof that r_1 divides a and b
- Proof that r_1 is the greatest divisor

Since $r_1|r$, we have

$$r = k_1 \cdot r_1 \text{ for some } k_1,$$

Since $b=r_1+r \cdot k_2$ for some k_2 ,
we have

$$b=r_1+k_1 \cdot r_1 \cdot k_2=r_1 \cdot (1+k_1 \cdot k_2)$$

As a result, we have $r_1|b$

Since $a= k_3 \cdot b + r$ for some k_3

$$a= k_3 \cdot b + r = k_3 \cdot r_1 \cdot (1+k_1 \cdot k_2) + k_1 \cdot r_1 = r_1(k_3 \cdot (1+k_1 \cdot k_2) + k_1)$$

We have $r_1|a$

If there exists a number $c \leq a$,
 $c|a$ and $c|b$

we have $c|a \rightarrow c|k_3 \cdot b + r$

Since $c|b$, we have $c|r$

we have $c|b \rightarrow c|(r_1 + r \cdot k_2)$

Since $c|r$, we have $c|r_1$

1723

101

5

67

11

2

3

17

13

71

Euclid's algorithm

EuclidGCD(a,b)

Assume a and b are nonnegative integers

```
if (b == 0)
    gcd(a,b) = a;                                // stopping condition.
else
    gcd(a,b) = gcd(b, a% b)                      // recursive step (%=mod)
```

Examples

1. Let a = 54, b = 30

$$\text{gcd}(54, 30) = \text{gcd}(30, 54 \% 30) = \text{gcd}(30, 24)$$

$$\text{gcd}(30, 24) = \text{gcd}(24, 30 \% 24) = \text{gcd}(24, 6)$$

$$\text{gcd}(24, 6) = \text{gcd}(6, 24 \% 6) = \text{gcd}(6, 0)$$

$$\text{gcd}(6, 0) = 6 \quad // \text{stop: gcd}(54, 30) = 6$$

2. Let a = 45, b = 16

$$\text{gcd}(45, 16) = \text{gcd}(16, 45 \% 16) = \text{gcd}(16, 13)$$

$$\text{gcd}(16, 13) = \text{gcd}(13, 16 \% 13) = \text{gcd}(13, 3)$$

$$\text{gcd}(13, 3) = \text{gcd}(3, 13 \% 3) = \text{gcd}(3, 1)$$

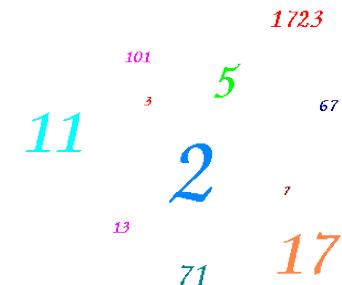
$$\text{gcd}(3, 1) = \text{gcd}(1, 3 \% 1) = \text{gcd}(1, 0)$$

$$\text{gcd}(1, 0) = 1 \quad // \text{stop: gcd}(45, 16) = 1$$

1723
101
3
5
67
11
2
7
13
71
17

The $\text{gcd}(a,b)$ as a Linear Combination of a & b

- $ax + by =$ “linear combination” of a and b
 - $12x + 20y = \{\dots, -12, -8, -4, 0, 4, 8, 12, \dots\}$
- The **minimum positive linear combination of a & b** = $\text{gcd}(a,b)$
- Proof in two steps:
 - If $d = \min(ax+by)$ and $d > 0$, then $d \mid a$, $d \mid b$
 - d is the greatest divisor.



$$d \mid a, d \mid b$$

By contradiction

Let $S = \{z = ax + by \mid z > 0\}$

Let $d = \min\{S\} = ax_1 + by_1$

Let $a = qd + r$. $0 \leq r < d$

$$r = a - qd = a - q(ax_1 + by_1)$$

$$r = a(1 - qx_1) + (-qy_1)b$$

If $r > 0$, $r \in S$

But $r < d$, which is a contradiction, because $d = \min\{S\}$

$$\therefore r = 0 \Rightarrow d \mid a$$

\therefore Similarly, we have $d \mid b$

1723

101
3
5
2

67

13

71

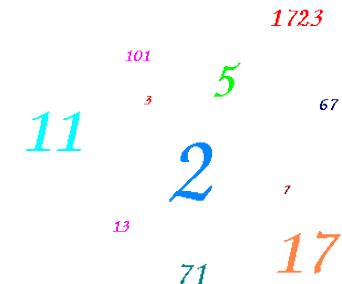
17

Extended Euclidean Algorithm

Given two integers a and b , we often need to find other two integers, u and v , such that

$$u \times a + v \times b = \gcd(a, b)$$

The extended Euclidean algorithm can calculate the $\gcd(a, b)$ and at the same time calculate the value of u and v .



Extended Euclidean Algorithm

Dividend	Divisor	Quotient	Reminder
a=60	= b=13	× 4	+ 8
b=13	= 8	× 1	+ 5
8	= 5	× 1	+ 3
5	= 3	× 1	+ 2
3	= 2	× 1	+ 1

$$1 = 3 - 2 \times 1$$

$$1 = 3 - (5 - 3 \times 1) \times 1 = 3 \times 2 - 5 \times 1$$

$$1 = (8 - 5 \times 1) \times 2 - 5 \times 1 = 8 \times 2 - 5 \times 3$$

$$1 = 8 \times 2 - (13 - 8 \times 1) \times 3 = 8 \times 5 - 13 \times 3$$

$$1 = (60 - 13 \times 4) \times 5 - 13 \times 3 = \underline{60 \times 5} - \underline{13 \times 23}$$

$$GCD(a=60,b=13)$$

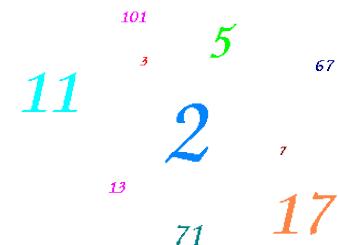
$$GCD(60,13)=1$$

$$1 = 60 \times 5 + 13 \times (-23)$$

$$U=5$$

$$V=-23$$

1723



Modular Arithmetic

- $a \equiv b \pmod{n}$
 - n is the modulus
 - a is “congruent” to b , modulo n
 - $a - b$ is divisible by n $n|(a-b)$
 - $a \% n = b \% n$ $7\%12 = 19\%12=7$
- $a \equiv b \pmod{n}$, $c \equiv d \pmod{n}$
 - Addition
 - $a + c \equiv b + d \pmod{n}$
 - Multiplication
 - $ac \equiv bd \pmod{n}$

$$a - b = jn$$

$$c - d = kn$$

$$a + c - (b + d) = (j + k) n$$

1723

101

5

67

11

2

17

3

7

13

71

Modular Arithmetic (Cont.)

- Power

➤ $a \equiv b \pmod{n} \Rightarrow a^k \equiv b^k \pmod{n}$

If $a^k \equiv b^k \pmod{n}$,

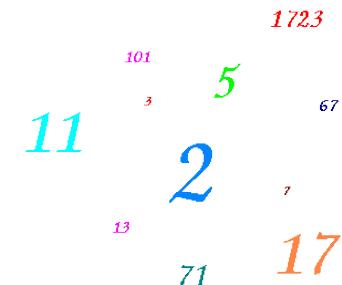
According to multiplication rule

$$a \cdot a^k \equiv b \cdot b^k \pmod{n},$$

$$\therefore a^{k+1} \equiv b^{k+1} \pmod{n}$$

- Going n times around the clock

➤ $a + kn \equiv b \pmod{n}$



Modular Arithmetic (Cont.)

- If a, b have no common factors, there exists a^i such that $a \cdot a^i \equiv 1 \pmod{b}$
 - a^i is called the “multiplicative inverse”
 - We can calculate a^i with extended Euclidean algorithm

Dividend	Divisor	Quotient	Remainder
$a=60$	$= b=13$	$\times 4$	$+8$
$b=13$	$= 8$	$\times 1$	$+5$
8	$= 5$	$\times 1$	$+3$
5	$= 3$	$\times 1$	$+2$
3	$= 2$	$\times 1$	$+1$

For example:

$$13 \times ? \equiv 1 \pmod{60}$$

$$1 = 3 - 2 \times 1$$

$$1 = 3 - (5 - 3 \times 1) \times 1 = 3 \times 2 - 5 \times 1$$

$$1 = (8 - 5 \times 1) \times 2 - 5 \times 1 = 8 \times 2 - 5 \times 3$$

$$1 = 8 \times 2 - (13 - 8 \times 1) \times 3 = 8 \times 5 - 13 \times 3$$

$$1 = (60 - 13 \times 4) \times 5 - 13 \times 3 = 60 \times 5 - 13 \times 23$$

$$1 = 60 \times 5 - 13 \times 23 \pmod{60} = -13 \times 23 \pmod{60}$$

$$1 = 13 \times (-23) = 13 \times 37 \pmod{60}$$

$$13 \times 37 = 481 = 8 \times 60 + 1 \pmod{60} = 1 \pmod{60}$$

Since $37 + 23 = 60 \pmod{60}$, $37 = -23 \pmod{60}$

1723

101

5

67

11

2

17

13

7

71

Exercises

- 1. If $p|10a - b, p|10c - d$, then $p|ad - bc$
- 2. if n is odd, then $3|2^n + 1$
- 3. $k = 0,1,2, \dots$ for $n \in \mathbb{Z}$, we have $2n + 1|1^{2k+1} + 2^{2k+1} + \dots + (2n)^{2k+1}$
- 4. if $m - p|mn + pq$ then $m - p|mq + np$
- 5. if $x \equiv 1 \pmod{m^k}$ then $x^m \equiv 1 \pmod{m^{k+1}}$

Euler Totient Function $\phi(n)$

- when doing arithmetic modulo n
- **complete set of residues** is: 0..n-1
- **reduced set of residues** is those numbers (residues) which are relatively prime to n
 - E.g., for n=10,
 - complete set of residues is {0,1,2,3,4,5,6,7,8,9}
 - reduced set of residues is {1,3,7,9}
- number of elements in reduced set of residues is called the **Euler Totient Function $\phi(n)$**

1723
101
3
5
67
11
2
7
13
71
17

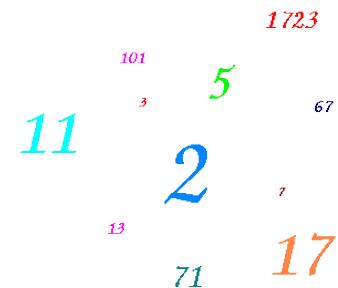
Euler Totient Function $\phi(n)$

- to compute $\phi(n)$ need to count number of residues to be excluded
- in general need prime factorization, but
 - for p (p prime) $\phi(p) = p-1$
 - for $p \cdot q$ (p, q prime)
 $\phi(pq) = \phi(p) \times \phi(q) = (p-1) \times (q-1)$
- eg.
 - $\phi(37) = 36, \phi(11) = 10$
 - $\phi(21) = (3-1) \times (7-1) = 2 \times 6 = 12,$
 - $\phi(10) = (2-1) \times (5-1) = 1 \times 4 = 4 \quad \{1, 3, 7, 9\}$

11 101
2 3
5
7
13
17
71
1723
67

Fermat's Theorem

- $a^{p-1} = 1 \pmod{p}$
 - where p is prime and $\gcd(a,p)=1$
- also known as Fermat's Little Theorem
- also $a^p = a \pmod{p}$
- useful in public key and primality testing
- $\phi(p)=p-1$



Euler's Theorem

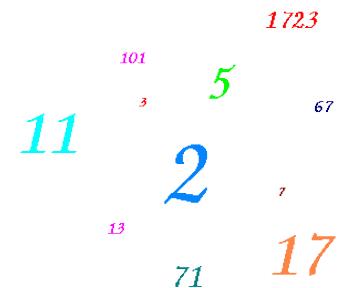
- a generalisation of Fermat's Theorem
- $a^{\phi(n)} = 1 \pmod{n}$
 - for any a, n where $\gcd(a, n) = 1$
- eg.

$$a=3; n=10; \phi(10)=4;$$

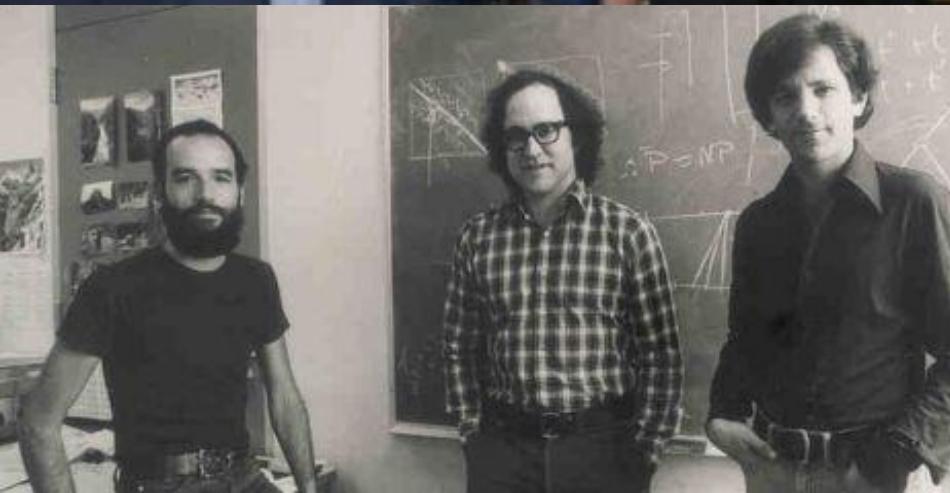
$$\text{hence } 3^4 = 81 = 1 \pmod{10}$$

$$a=2; n=11; \phi(11)=10;$$

$$\text{hence } 2^{10} = 1024 = 1 \pmod{11}$$



RSA Public Key Cryptosystem



P, q ARE PRIME
 $n = p \cdot q$
 $\phi(n) = (p-1)(q-1)$
RELATIVELY PRIME:
 $e \perp \phi(n)$
 $(d \cdot e) \bmod \phi(n) = 1$
PUBLIC KEY:
 $P = (e, n)$
SECRET KEY:
 $S = (d, n)$
CIPHER TEXT:
 $P(M) = M^e \bmod n = C$
MESSAGE:
 $S(C) = C^d \bmod n = M$
 $\therefore S(P(M)) = M$

Thank
you



CS 6355/4355: Cryptanalysis and Database Security

Topic 4: RSA Public Key Encryption

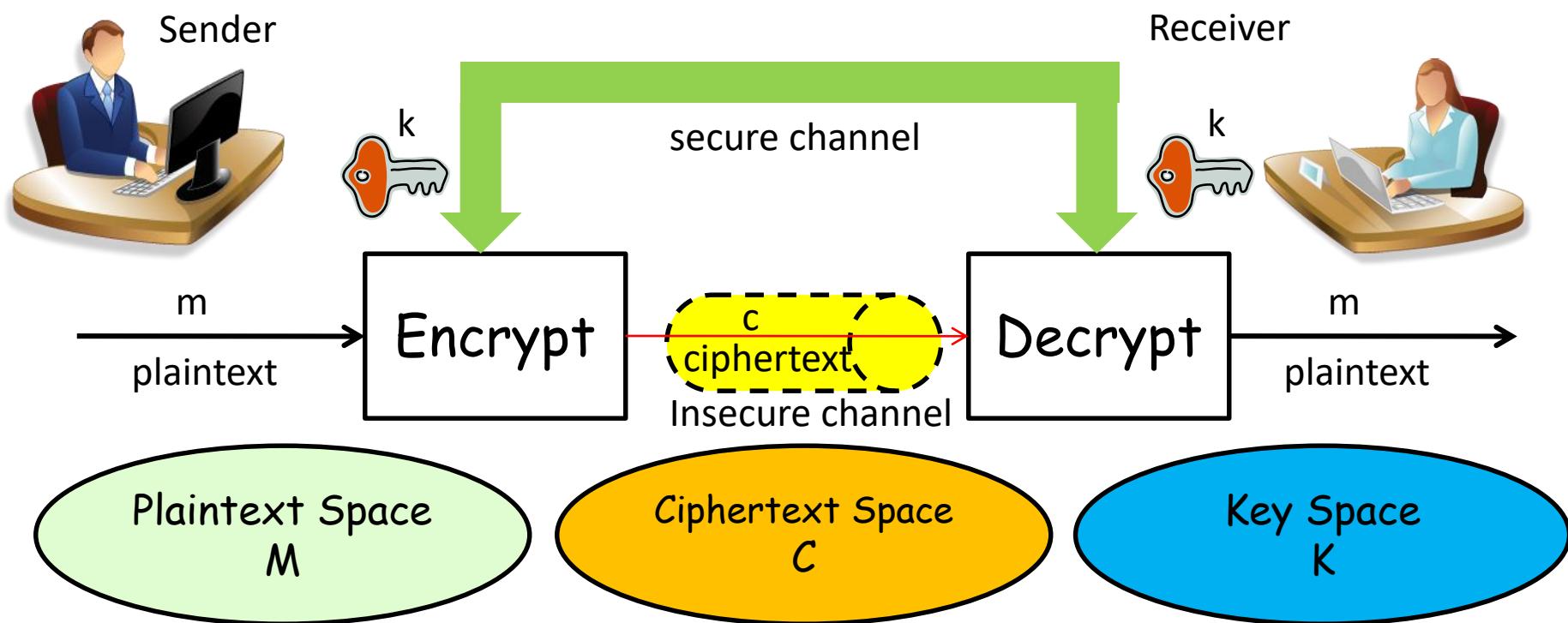
Lecturer: Rongxing LU

Email: RLU1@unb.ca Office: GE 114

Website: <http://www.cs.unb.ca/~rlu1/>

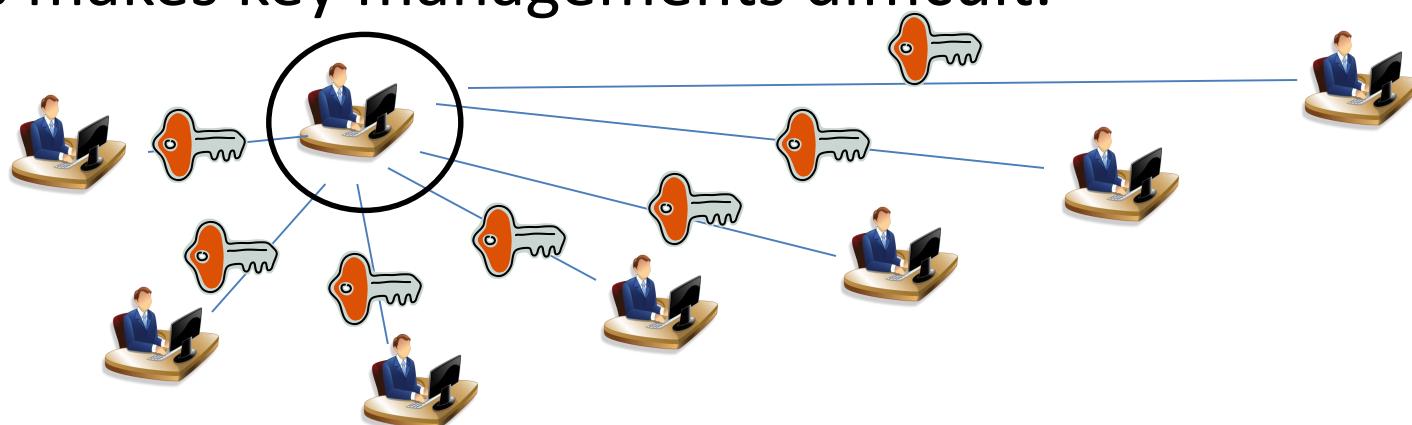
Faculty of Computer Science, University of New Brunswick

Review of Symmetric Cryptography

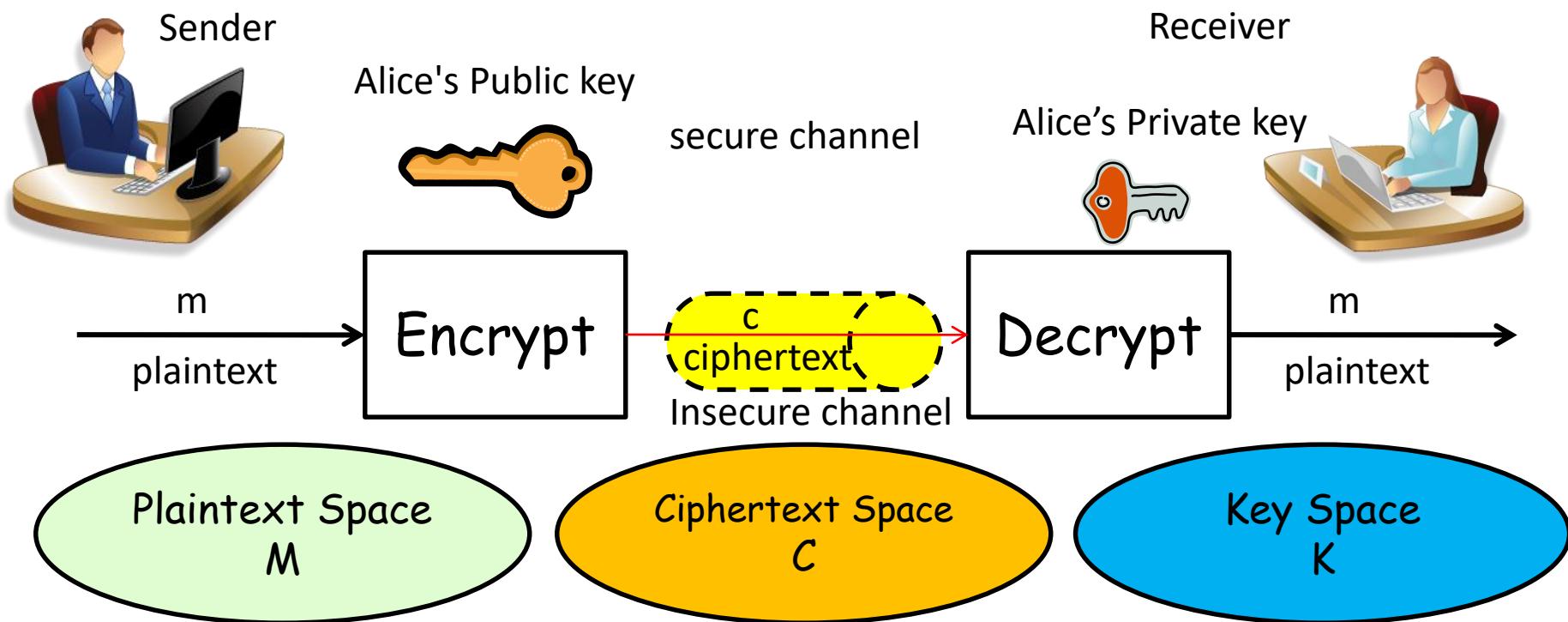


Disadvantages of Symmetric Cryptography

- The sender and receiver must share the same secret key. Key distribution is a must.
- If 1000 people want to communicate (two and two, in all possible ways), each must keep 999 secret keys, and the system requires a total of $(999*1000)/2=499500$ secret keys.
- This makes key managements difficult.



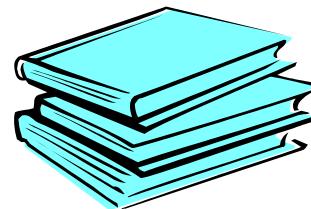
Asymmetric Cryptography



Asymmetric Cryptography

=>Public key Cryptography

Public key Directory



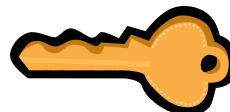
- 1. Simplify the key management
- 2. Make the digital signature possible



Public Key Cryptography

- **Public Key/Asymmetric** cryptography involves the use of **two keys**:

➤ **public-key**, which may be known by anybody, and can be used to **encrypt messages**, and **verify signatures**

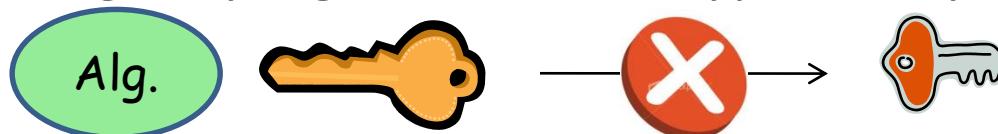


➤ **private-key**, known only to the recipient, used to **decrypt messages**, and **sign (create) signatures**



Characteristics of Public Key

- Public-Key algorithms rely on two keys where:
 - it is computationally infeasible to find decryption key knowing only algorithm & encryption key

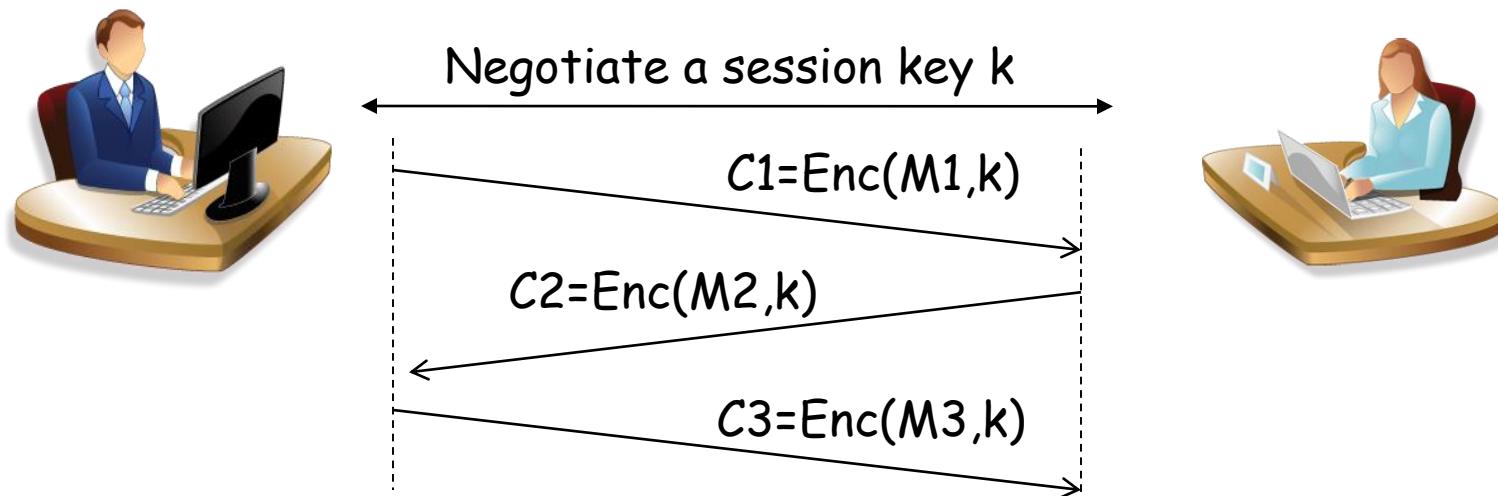


- it is computationally easy to en/decrypt messages when the relevant (en/decrypt) key is known
- either of the two related keys can be used for encryption, with the other used for decryption (for some algorithms)



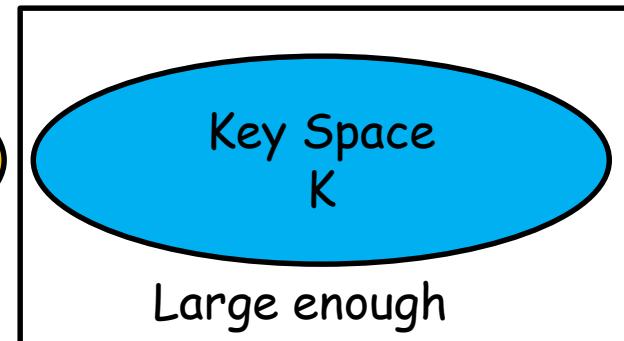
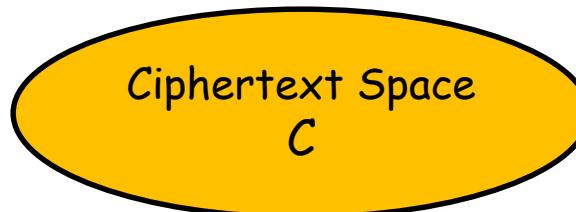
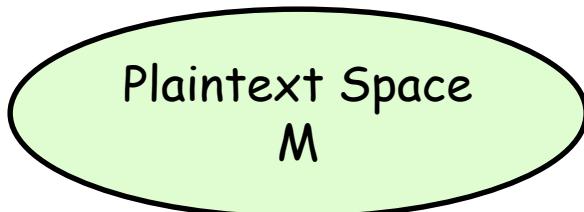
Applications of Public Key Cryptography

- Public-Key algorithms
 - encryption/decryption (provide confidentiality)
 - digital signatures (provide authentication)
 - key exchange (of session keys)



Security of Public Key Algorithms

- keys used are too large (>512bits, 1024 bits, ...)
- security relies on the known hard problem, it is hard enough to be impractical to break
- requires the use of very large numbers
- hence is slow compared to private key schemes

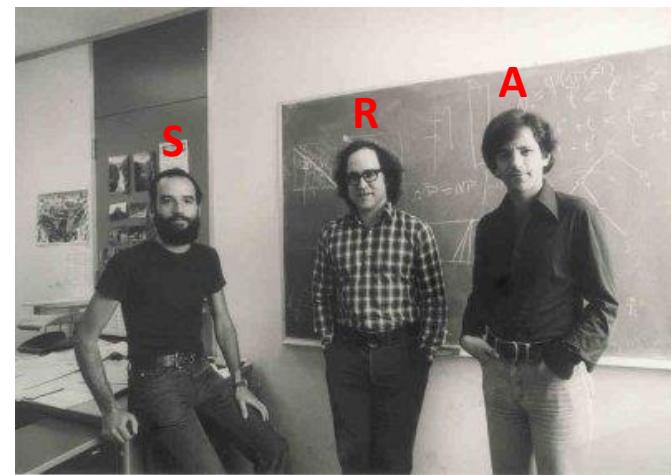


Introduction of RSA

- Mathematic Background of RSA
- RSA cryptosystem
- RSA cryptanalysis



P & Q PRIME
 $N = PQ$
 $ED \equiv 1 \pmod{(P-1)(Q-1)}$
 $C = M^e \pmod{N}$
 $M = C^d \pmod{N}$
RSA Algorithm



https://www.youtube.com/watch?v=f_8s451zYYE

Fermat's Little Theorem

- $a^{p-1} = 1 \pmod{p}$
 - where p is prime and $\gcd(a,p)=1$

Proof

- List all elements $< p$, e.g., a set $A=\{1,2,3,\dots, p-1\}$
- Choose any element $a < p$, and multiple a with all elements in the set A , i.e., a set $B = \{a^1 \pmod{p}, a^2 \pmod{p}, a^3 \pmod{p}, \dots, a^{(p-1)} \pmod{p}\}$
- Suppose two element (r, s) in A such that $r \neq s$, but $a^r = a^s \pmod{p}$. In this case, we have $a(r - s) = 0 \pmod{p}$. Because $\gcd(a, p)=1$, we have $(r - s) = 0 \pmod{p}$, i.e., $r = s$, which contradicts with $r \neq s$. Therefore, the set B also includes $p-1$ elements, which are less than p .



Fermat's Little Theorem

- $a^{p-1} = 1 \pmod{p}$
 - where p is prime and $\gcd(a,p)=1$

Proof



- Now, since both the set $A = \{1, 2, 3, \dots, p-1\}$ and the set $B = \{a^1 \pmod{p}, a^2 \pmod{p}, a^3 \pmod{p}, \dots, a^{p-1} \pmod{p}\}$ contains $p-1$ elements less than p , we can multiply all elements in A and B as follows,

$$1 * 2 * 3 * \dots * (p-1) = \underline{a^1} * \underline{a^2} * \underline{a^3} * \dots * \underline{a^{p-1}} \pmod{p}$$

$$- 1 * 2 * 3 * \dots * (p-1) = 1 * 2 * 3 * \dots * (p-1) * \underline{a^{p-1}} \pmod{p}$$

$$- 1 = a^{p-1} \pmod{p} \quad \blacksquare$$

Exercises

$$4^6 \bmod 7 = ?$$

$$19^{22} \bmod 23 = ?$$

$$89^{100} \bmod 101 = ?$$

$$4^6 \bmod 7 = 1$$

$$19^{22} \bmod 23 = 1$$

$$89^{100} \bmod 101 = 1$$

$$4^{37} \bmod 7 = ?$$

$$19^{45} \bmod 23 = ?$$

$$89^{1001} \bmod 101 = ?$$

$$4^{37} \bmod 7 = 4^{6 \times 6 + 1} \bmod 7 = (4^6)^6 \cdot 4 \bmod 7 = 1^6 \cdot 4 \bmod 7 = 4$$

$$19^{45} \bmod 23 = 19^{22 \times 2 + 1} \bmod 23 = (19^{22})^2 \cdot 19 \bmod 23 = 19$$

$$89^{1001} \bmod 101 = 89^{100 \times 10 + 1} \bmod 101 = (89^{100})^{10} \cdot 89 \bmod 101 = 89$$

For a large prime p , and any element a , $\gcd(a, p) = 1$, we have $a^{p-1} \bmod p = 1$, $a^{(p-1)k+1} \bmod p = a$, $a^p \bmod p = a$

Euler's Theorem

- If n and a are coprime positive integers, i.e., $\gcd(a, n)=1$, then $a^{\varphi(n)} = 1 \pmod{n}$, where $\varphi(n)$ is Euler's totient function.

- for p (p prime) $\varphi(p) = p-1$

- for p,q (p,q prime)

$$\varphi(pq) = \varphi(p) \times \varphi(q) = (p-1) \times (q-1)$$



➤ eg.

$$\varphi(37) = 36, \varphi(11) = 10$$

$$\varphi(21) = (3-1) \times (7-1) = 2 \times 6 = 12,$$

$$\varphi(10) = (2-1) \times (5-1) = 1 \times 4 = 4 \quad \{1, 3, 7, 9\}$$

Euler's Theorem

- $\varphi(n)$ denotes the number of positive integers less than n which are relatively prime to n , where $n=pq$
- Example to show why

$$\varphi(pq) = \varphi(p) \times \varphi(q) = (p-1) \times (q-1)$$

- $p=7, q=5, n=pq=7 \times 5=35$

0	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31	32	33	34

$$\gcd(a, 35) = 1$$

$$a = 1, 2, 3, 4, 6, 8, 9, 11, 12, 13, 16, 17, 18, 19, 22, 23, 24, 26, 27, 29, 31, 32, 33, 34$$

$$p=7, 7*x, x=\{1, 2, 3, 4\}, (q-1) \text{ elements } \gcd(7*x, 35)=7$$

$$q=5, 5*y, y=\{1, 2, 3, 4, 5, 6\}, (p-1) \text{ elements } \gcd(5*y, 35)=5$$

$$\varphi(n) = p*q - (q-1) - (p-1) = (p-1)(q-1) \quad \varphi(35) = \underline{35-1} - 4-6 = 24$$

Euler's Theorem

Proof of $a^{\phi(n)} = 1 \pmod{n}$, where $n=pq$

- First, $\phi(n)$ denotes the number of positive integers less than n which are relatively prime to n
- Let $A=\{r_1, r_2, \dots, r_{\phi(n)}\}$ be the set of $\phi(n)$ integers less than n and relatively prime to n , where any $r_i \neq r_j$.
- For any element a in A , $\gcd(a, n)=1$, then, $a^*r_1 \pmod{n}$, $a^*r_2 \pmod{n}$, ... $a^*r_{\phi(n)} \pmod{n}$ are distinct, otherwise if $r_i^*a = r_j^*a \pmod{n}$ then $r_i = r_j \pmod{n}$ (**Contradiction**).
- Since each r_i^*a is relatively prime to n and there are only $\phi(n)$ distinct numbers relatively prime to n , therefore, the set $B=\{a^*r_1 \pmod{n}, a^*r_2 \pmod{n}, \dots, a^*r_{\phi(n)} \pmod{n}\}$ is identical to the set $A=\{r_1, r_2, \dots, r_{\phi(n)}\}$.

Euler's Theorem

Proof of $a^{\varphi(n)} = 1 \pmod{n}$, where $n=pq$

- Since the set $B=\{a^*r_1 \pmod{n}, a^*r_2 \pmod{n}, \dots a^*r_{\varphi(n)} \pmod{n}\}$ is identical to the set $A=\{r_1, r_2, \dots r_{\varphi(n)}\}$.

$$r_1 * r_2 * \dots * r_{\varphi(n)} = \underline{a^*r_1} * \underline{a^*r_2} * \dots * \underline{a^*r_{\varphi(n)}} \pmod{n}$$

$$r_1 * r_2 * \dots * r_{\varphi(n)} = r_1 * r_2 * \dots * r_{\varphi(n)} * \underline{a^{\varphi(n)}} \pmod{n}$$

$$1 = \underline{a^{\varphi(n)}} \pmod{n} \quad ■$$

$$25^{120} \pmod{143} = ?$$

$$19^{60} \pmod{77} = ?$$

Exercise

$$25^{120} \pmod{143} = 25^{(11-1)(13-1)} \pmod{11 \times 13} = 1$$

$$19^{60} \pmod{77} = 19^{(7-1)(11-1)} \pmod{7 \times 11} = 1$$

Exercises

$$25^{1201} \bmod 143 = ?$$

$$19^{481} \bmod 77 = ?$$

$$25^{1201} \bmod 143 = 25^{120 \times 10 + 1} \bmod 11 \times 13 = (25^{120})^{10} \cdot 25 \bmod 11 \times 13 = 25$$

$$19^{481} \bmod 77 = 19^{60 \times 8 + 1} \bmod 7 \times 11 = (19^{60})^8 \cdot 19 \bmod 7 \times 11 = 19$$

For a large number $n = pq$, where p, q are primes,
and any element a , $\gcd(a, n) = 1$,
we have $a^{\varphi(n)} \bmod n = 1$, $a^{\varphi(n) \cdot k + 1} \bmod n = a$, $a^{\varphi(n) + 1} \bmod n = a$

Testing for Primality

- For many cryptographic algorithms including RSA, it is necessary to select one or more very large prime numbers at random. Thus we are faced with the task of determining whether a given large number is prime. There is no simple yet efficient means of accomplishing this task.

65	64	63	62	61	60	59	58	57
66	37	36	35	34	33	32	31	56
67	38	17	16	15	14	13	30	55
68	39	18	5	4	3	12	29	54
69	40	19	6	1	2	11	28	53
70	41	20	7	8	9	10	27	52
71	42	21	22	23	24	25	26	51
72	43	44	45	46	47	48	49	50
73	74	75	76	77	78	79	80	81

Testing for Primality

Proof: "There are Infinitely Many Primes"

- Assume that the primes are finite, and we can list them as $L=\{p_1, p_2, \dots, p_r\}$
- Let p be any common multiple of these primes plus one, i.e., $p = p_1 * p_2 * \dots * p_r + 1$. Then, p is either a prime or not.
- If p is a prime, then p is a prime that was not in L .
- If p is not a prime, then p is divisible by some prime, call p_0
- Here, p_0 can not be any of p_1, p_2, \dots, p_r , otherwise p_0 would divide 1, which is impossible. So this prime p_0 is some prime that was not in list $L=\{p_1, p_2, \dots, p_r\}$. In other words, the $L=\{p_1, p_2, \dots, p_r\}$ was incomplete, and there are infinite many primes. ■

Testing for Primality

Naive Try

```
boolean isPrime(integer n)
    if ( n < 2 ) return false
    for(i = 2 to n -1)
        if( i |n ) // "i divides n"
            return false
    return true
```

Question: What is the running time of this algorithm?

Answer: Assuming divisibility testing is a basic operation - so $O(1)$ (this is an invalid assumption)- then above primality testing algorithm is $O(n)$.

Testing for Primality

Second Try

- Don't try number bigger than $n/2$.
- After trying 2, don't try any other even numbers, because know n is known odd if $2 \nmid n$ by trying 2.
- In general, try only smaller prime numbers.
- In fact, only need to try to divide by prime numbers no larger than \sqrt{n}

```
boolean isPrime(integer n)
    if ( n < 2 ) return false
    for each prime number i no larger than  $\sqrt{n}$ 
        if( i | n ) // "i divides n"
            return false
    return true
```

Testing for Primality

Theorem:

If n is a composite, then its smallest prime factor is $\leq \sqrt{n}$

Proof. (By contradiction).

Suppose the smallest prime factor is $> \sqrt{n}$. Then we can decompose $n = p^*q^*x$ where p and q are primes $> \sqrt{n}$ and x is some integer. Therefore

$$n = p^*q^*x > \sqrt{n} * \sqrt{n} * x = nx,$$

which implies that $n > n$.

Therefore, it shows the supposition (the smallest prime factor is $> \sqrt{n}$) was false.

As a result, the theorem is proved.

```
boolean isPrime(integer n)
    if ( n < 2 ) return false
    for each prime number i no larger than  $\sqrt{n}$ 
        if( i | n )      // "i divides n"
            return false
    return true
```

Running time $< O(\sqrt{n})$

Testing for Primality

Examples. Test if 139 and 143 are primes.

List all primes up to \sqrt{n} and check if they divide the numbers.

{2,3,5,7,11}, since $13^2 = 169 > 143$

	139	143
2	2 ∤ 139	2 ∤ 143
3	3 ∤ 139	3 ∤ 143
5	5 ∤ 139	5 ∤ 143
7	7 ∤ 139	7 ∤ 143
11	11 ∤ 139	11 143

$$11 \times 13 = 143$$

Therefore, 139 is a prime number, but 143 is composite.

Testing for Primality

Direct application of Fermat's Little Theorem for testing primality of n $a^{p-1} = 1 \pmod{p}$

If we want to test if p is prime, then we can pick random a 's in the interval and see if the equality holds. If the equality does not hold for a value of a , then p is composite. If the equality does hold for many values of a , then we can say that p is probably prime.

Inputs: n : a value to test for primality;

k : a parameter that determines the number of times to test for primality

Output: composite if n is composite, otherwise probably prime
repeat k times:

 pick a randomly in the range $(1, n - 1]$

 if $a^{n-1} \neq 1 \pmod{n}$, then return composite

 return probably prime

Testing for Primality

Example.

For $n = 221$, we test whether it is a prime.

1. We randomly pick $1 \leq a < 221$, say $a = 38$. Check

$$a^{n-1} = 38^{220} = 1 \pmod{221}$$

2. From the first test, 221 could be a prime.
3. We take another $a = 26$ for test

$$a^{n-1} = 26^{220} = 169 \neq 1 \pmod{221}$$

4. Therefore, 221 is not a prime.
 $(221=17*13)$

repeat k times

Testing for Primality

Miller-Rabin algorithm

Observation: Given an odd integer $n > 2$, $n - 1$ can be expressed as $n - 1 = 2^k q$, with $k > 0$, and q odd.

Fact. If n is a prime number, for some $a < n$, $\gcd(a, n) = 1$, we have $a^{n-1} \equiv 1 \pmod{n}$. Then,

$$a^{n-1} - 1 \equiv 0 \pmod{n} \Rightarrow a^{2^k q} - 1 \equiv 0 \pmod{n}$$

$$a^{2^k q} - 1 = (a^{2^{k-1} q} - 1)(a^{2^{k-1} q} + 1) \equiv 0 \pmod{n}$$

And we have

$$a^{2^{k-1} q} - 1 \equiv 0 \pmod{n}$$

$$a^{2^{k-1} q} = (a^q)^{2^{k-1}} \equiv 1 \pmod{n}$$

$$a^q \equiv 1 \pmod{n}$$

$$a^{2^{k-1} q} + 1 \equiv 0 \pmod{n}$$

$$a^{2^{k-1} q} \equiv -1 \equiv n - 1 \pmod{n}$$

Testing for Primality

Therefore, if n is a prime number, we have

$$a^q \equiv 1 \pmod{n} \quad a^{2^{k-1}q} \equiv -1 = n-1 \pmod{n}$$

But if $a^q \equiv 1 \pmod{n}$ we can say n is probably prime.

If $a^q \not\equiv 1 \pmod{n}$ we can confirm n is NOT a prime number

MILLER-RABIN(n)

Repeat for Primality Testing

1. find integers $k > 0$, q odd, so that $n - 1 = 2^k q$
2. select a random integer a , $1 < a < n - 1$
3. if $a^q \pmod{n} \equiv 1$ then return "probably prime"
4. for $j = 0$ to $k - 1$ do
5. if $a^{2^j q} \pmod{n} \equiv n - 1$ then return "probably prime"
6. return "composite"

Testing for Primality

Determine If $n = 221$ is prime. We can write

$$n - 1 = 220 = 2^2 \cdot 55$$

We have $k = 2$ and $q = 55$. Randomly select a number a such that $a < n$, e.g., 174. Then,

$$a^{2^0 \cdot q} \bmod n = 174^{55} \bmod 221 = 47 \neq 1, n - 1$$

$$a^{2^1 \cdot q} \bmod n = 174^{110} \bmod 221 = 220 = n - 1$$

Since $220 \equiv -1 \pmod{n}$, 221 is probably prime. We try another random a , this time choosing $a=137$:

$$a^{2^0 \cdot q} \bmod n = 137^{55} \bmod 221 = 88 \neq 1, n - 1$$

$$a^{2^1 \cdot q} \bmod n = 137^{110} \bmod 221 = 205 \neq n - 1$$

Hence 221 is not a prime number. Actually, $221=13 \cdot 17$

repeat k times

Integer Factorization Problem

- Input:
 - N: odd composite integer with at least two distinct prime factors. For example, $N=pq$.
- Output:
 - Primes p and q.
- Examples.
 - $21=7\times3$, $77=7\times11$, $143=11\times13$, $483=21\times23$
 - $11797055816053=3554179\times3319207$
 - $6962917758703=1836059\times3792317$

Integer Factorization Problem

- Examples.
 - $21=7\times 3$, $77=7\times 11$, $143 =11\times 13$, $483=21\times 23$
 - $11797055816053=3554179\times 3319207$
 - $6962917758703=1836059\times 3792317$
 - Given p, q , to compute $N=pq$ (easy)
 - Given $N=pq$, to compute p or q
 - ❖ if N is small, easy
 - ❖ else if N is large, difficult

The RSA Challenge Numbers

- $N=pq$
 - RSA-576
 - Decimal Digits: 174
 - ❖ $N=18819881292060796383869723946165$
0439807163563379417382700763356422
9888597152346654853190606065047430
45317388011303396716199692321205734
0318795506569962213051687593076502
57059

The RSA Challenge Numbers

- RSA-576
 - On December 3, 2003, a team of researchers reported a successful factorization of the RSA-576
 - Decimal Digits: 174
 - ❖ P=39807508642406493739712550055038
6491199064362342526708406385189575
946388957261768583317
 - ❖ q=472772146107435302536223071973048
22463291469530209711645985217113052
0711256363590397527

The RSA Challenge Numbers

- RSA-896 (Not factored)
 - Decimal Digits: 270
 - ❖ N=41202343698665954385553136533257
5948179811699844327982845455626433
87644556524842619809887042316184187
9261420247188869492560931776375033
42113098239748515094490910691026986
10318627041148808669705649029036536
58867433731720813104105190864254793
282601391257624033946373269391

RSA Cryptosystem

- p and q are large primes.
- $n = p \cdot q$
- $\varphi(n) = (p - 1)(q - 1)$
- e is an integer, $1 < e < \varphi(n)$, $\gcd(e, \varphi(n)) = 1$
- d is an integer, $1 < d < \varphi$, $e \cdot d \equiv 1 \pmod{\varphi}$
(d is inverse of e in modulo $\varphi(n)$, calculated by the extended Euclidean algorithm)
- (n, e) is public key
- (n, d) is private key

RSA Cryptosystem (Cont.)

- Encryption : $c=m^e \text{ mod } n$
- Decryption : $m=c^d \text{ mod } n$
- m is plaintext, c is ciphertext.

$$\begin{aligned}m &= c^d \text{ mod } n = m^{ed} \text{ mod } n \\&= m^{1+k\varphi(n)} \text{ mod } n = m \cdot (m^{\varphi(n)})^k \text{ mod } n \\&= m \cdot 1 \text{ mod } n = m\end{aligned}$$

RSA Example



$$m = 15$$

$$C = m^e \bmod n = 15^{13} \bmod 77 = 64$$

$$PK : (n, e)$$

$$SK : d$$

$$\begin{aligned} p &= 7, q = 11, n = pq = 7 \times 11 = 77 \\ \phi(n) &= (p-1)(q-1) = 6 \times 10 = 60 \\ e &= 13, d = ? \\ \therefore \gcd(60, 13) &= 1 \end{aligned}$$

C

$$\begin{aligned} m &= C^d \bmod n = 64^{37} \bmod 77 = 15 \\ &= 15^{13 \times 37} \bmod 77 = 15^{1+8 \times 60} \bmod 77 = 15 \end{aligned}$$

$$1 = 3 - 2 \times 1$$

$$1 = 3 - (5 - 3 \times 1) \times 1 = 3 \times 2 - 5 \times 1$$

$$1 = (8 - 5 \times 1) \times 2 - 5 \times 1 = 8 \times 2 - 5 \times 3$$

$$1 = 8 \times 2 - (13 - 8 \times 1) \times 3 = 8 \times 5 - 13 \times 3$$

$$1 = (60 - 13 \times 4) \times 5 - 13 \times 3 = \underline{60 \times 5} - \underline{13 \times 23}$$

$$1 = 60 \times 5 - 13 \times 23 \bmod 60 = -13 \times 23 \bmod 60$$

$$1 = 13 \times (-23) = 13 \times 37 \bmod 60$$

$$d = 37$$

Since $37 + 23 = 60 \bmod 60$, $37 = -23 \bmod 60$

Dividend	Divisor	Quotient	Remainder
$\phi(n) = 60$	$= e = 13$	$\times 4$	$+ 8$
$e = 13$	$= 8$	$\times 1$	$+ 5$
8	$= 5$	$\times 1$	$+ 3$
5	$= 3$	$\times 1$	$+ 2$
3	$= 2$	$\times 1$	$+ 1$

Security of RSA Cryptosystem

- Since n and e are in public key, in order to recover $m=c^d \bmod n$ from c , we need to know the private key d .
- While in order to compute d , we need to factorize $n=p \cdot q$ to compute $\varphi(n)=(p-1)(q-1)$.
- Therefore, the security of RSA cryptosystem is dependent upon the hardness of factor large integer.
- Currently, 1024 bit-RSA is commonly used.
- For other applications, like military application, 4096 bit-RSA is preferred.

The difference value $|p-q|$ should be large

Why

large

- If $n (>0)$ is an odd integer n and written as $n=x^*y$. Then, n can be also written as $u^2 - v^2$, where $u=(x+y)/2$, $v=(x-y)/2$. If $x > y$, both u and v are >0 .
- If $n=x^*y$, and x, y are close, then v is a small number, and u is a little larger than \sqrt{n} .
- In this case, we can guess u by trying $(\sqrt{n}+1, \sqrt{n}+2, \sqrt{n}+3, \dots)$ to satisfy $u^2 - n = v^2$.
- Since u is a little larger than \sqrt{n} , we can quickly determine u , and then v .
- Finally, we can factor $n=x^*y$ as $x=u+v$, $y=u-v$.

Example

- If $p=401$, $q=409$, then $n=p*q=164009$
- We can calculate $\sqrt{n}=404.98 \approx 405$.
- Since $405^2 - n = 4^2$, $(405^2 = 164025)$
- We can determine $v=4$, and then $u=405$.
- Finally $x=u+v=409$, $y=u-v=401$
- Therefore, the difference value $|p-q|$ should be large, but p, q should be of the same length. $|p| \approx |q|$.

The modulus $n=p*q$ can not be common in the system

Why

- Suppose two entities share the common modulus $n=p*q$, but have different public-private key pairs (e_1, d_1) and (e_2, d_2) with $\gcd(e_1, e_2)=1$. If the same message m is encrypted with $c_1=m^{e_1} \pmod{n}$, $c_2=m^{e_2} \pmod{n}$, then the message m can be recovered without knowing the private keys.
- Since $\gcd(e_1, e_2)=1$, we can use the extended Euclidean algorithm to calculate r, s such that $r*e_1+s*e_2=1$.
- Then, $c_1^r * c_2^s = m^{r*e_1+s*e_2} = m \pmod{n}$, we can get m
- If $r < 0$, we should calculate $(c_1^{-1})^{-r} * c_2^s = m \pmod{n}$

Example

- If $p=7$, $q=11$, $n=p^*q=77$, $\varphi(n)=(p-1)(q-1)=60$.
- $(e_1=13, d_1=37)$, $(e_2=17, d_2=53)$ extended Euclidean algorithm
- For $m=15$, $c_1=m^{e_1} \text{ mod } n=15^{13} \text{ mod } 77=64$, $c_2=m^{e_2} \text{ mod } n=15^{17} \text{ mod } 77=71$
- Since $\gcd(e_1, e_2)=1$, we have $4*13 + (-3)*17 = 1$
extended Euclidean algorithm
- Because $s=-3$, we compute $(c_2)^{-1} \text{ mod } 77=-13$
 $\text{mod } 77=64$ extended Euclidean algorithm $-13*71 + 12*77 = 1$
- $c_1^{e_2} * (c_2^{-1})^s = 64^4 * 64^3 \text{ mod } 77 = 15$

common modulus attack

Example (2)

- If $p=23$, $q=29$, $n=p*q=667$, $\varphi(n)=(p-1)(q-1)=616$.
- $(e_1=13, d_1=273)$, $(e_2=41, d_2=601)$ extended Euclidean algorithm
- For $m=20$, $c_1=m^{e_1} \text{ mod } n=20^{13} \text{ mod } 667=451$,
 $c_2=m^{e_2} \text{ mod } n=20^{41} \text{ mod } 667=132$
- Since $\gcd(e_1, e_2)=1$, we have $19*13 + (-6)*41 = 1$
extended Euclidean algorithm
- Because $s=-6$, we compute $(c_2)^{-1} \text{ mod } 667=-96$
 $\text{mod } 667=571$ extended Euclidean algorithm $-96*132 + 19*667 = 1$
- $c_1^{r_s} * (c_2^{-1})^s = 451^{19} * 571^6 \text{ mod } 667 = 20$

common modulus attack

The padding is required in RSA


$$PK : (n, e)$$
$$SK : d$$


$m = \text{"hello world"}$

$$C = m^e \bmod n$$

 C C' 

$$R = r^e \bmod n$$

$$C' = C \cdot R \bmod n$$

$$C'^d \bmod n = m \cdot r \bmod n$$

Since m is meaningful, If the recovered is m^*r , which becomes meaningless. Then, the receiver can detect it.

However, if m is a session key (a random string), the receiver cannot identify the recovered m^*r is correct or not.

The padding is required (Cont.)



m

$$C = (\underbrace{m \parallel H(m)}_{l\text{-bits}})^e \bmod n$$

C

$PK : (n, e)$

$SK : d$



C'



$$C' = C \cdot R \bmod n$$

$$C'^d \bmod n = a \parallel \underbrace{b}_{l\text{-bits}}$$

check $H(a) = b$

With the padding, the receiver can detect whether the message has Modified during the transmission.

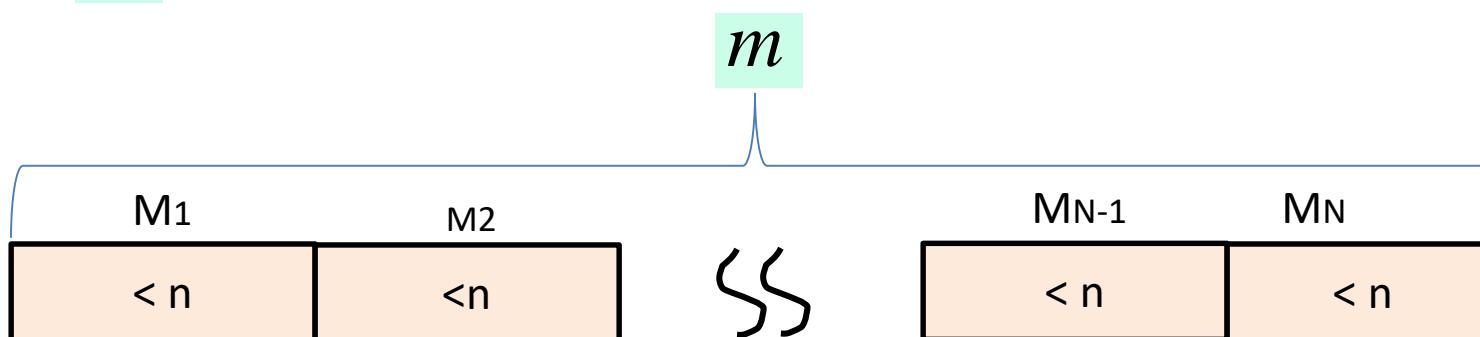
How to encrypt a long message



m

$PK : (n, e)$

$SK : d$



- If m is large than n , then m should be first divided into $m=m_1m_2m_3\dots$, each $m_i < n$.
- However, the efficiency is big problem.

How to encrypt a long message (Cont.)

Encryption algorithm	Comparison1	Comparison 2
<u>Symmetric Encryption Algorithm</u> , e.g., AES	fast	Need sharing a secret key in advance
<u>RSA Algorithm</u>	Slow, usually 100-1000 times slower than AES, (usually used for encrypting small message)	Knowing the receiver's public key, do not need sharing a secret key in advance



You

$\text{pk } (n, e)$
no shared key



Your friend

 $\text{sk } d$

Discussion 1: how to send a short text message m ($<1k$) secretly and efficiently?

Discussion 2: how to send a file M of large size secretly and efficiently?



$$c = m^e \pmod{n}$$

\xrightarrow{c}



$$m = c^d \pmod{n}$$



choose a key k

$$c_1 = k^e \pmod{n}$$

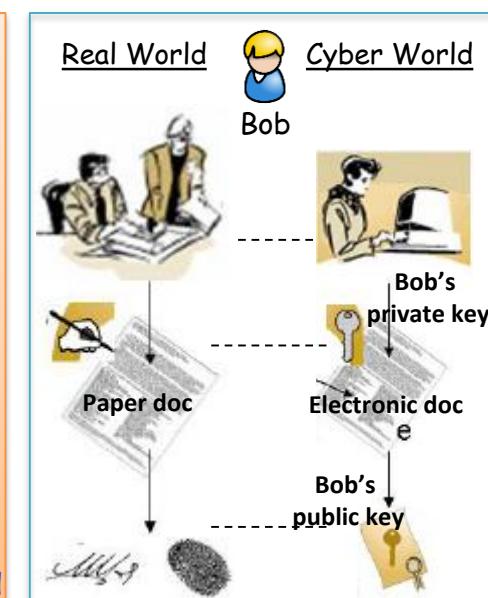
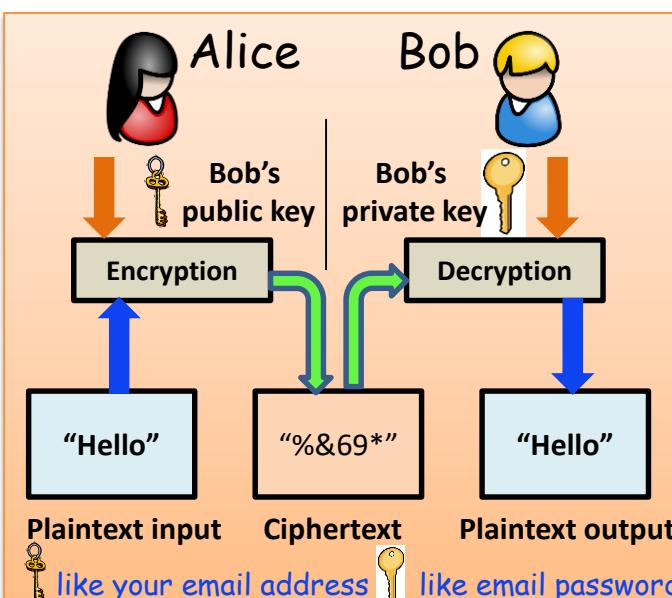
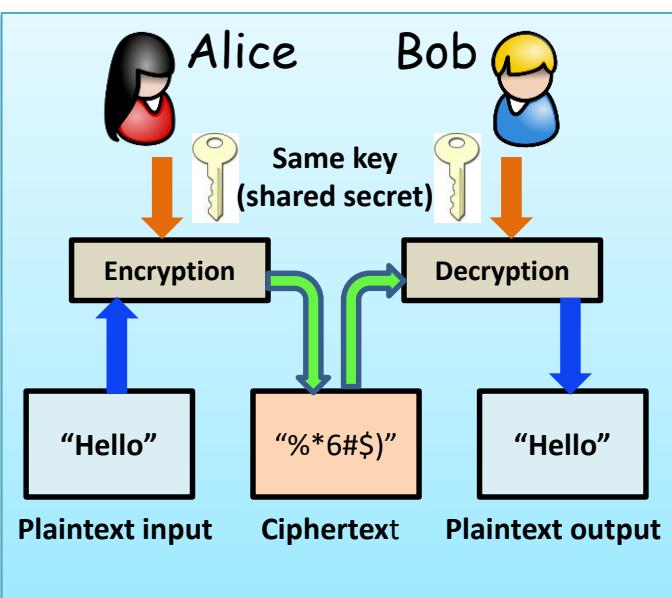
$$C_2 \leftarrow \text{AES}(M, k) \xrightarrow{(c_1, C_2)}$$



$$k = c_1^d \pmod{n}$$

$$M \leftarrow \text{AES}(C_2, k)$$

Comparison Symmetric Encryption and Public Key Encryption



Symmetric Encryption (AES, DES, ...)

- Fast
- Need share the same key
- Achieve Confidentiality

Public key Encryption (RSA)

- Slow (due to exponential operation)
- Do not need share the same key
- Achieve Confidentiality

Digital Signature

- + Use private key to sign
- + Use public key to verify
- + Achieve authentication, data integrity, non repudiation

Thank
you



CS 6355/4355: Cryptanalysis and Database Security

Topic 5: Key Exchange and ElGamal Public Key
Encryption

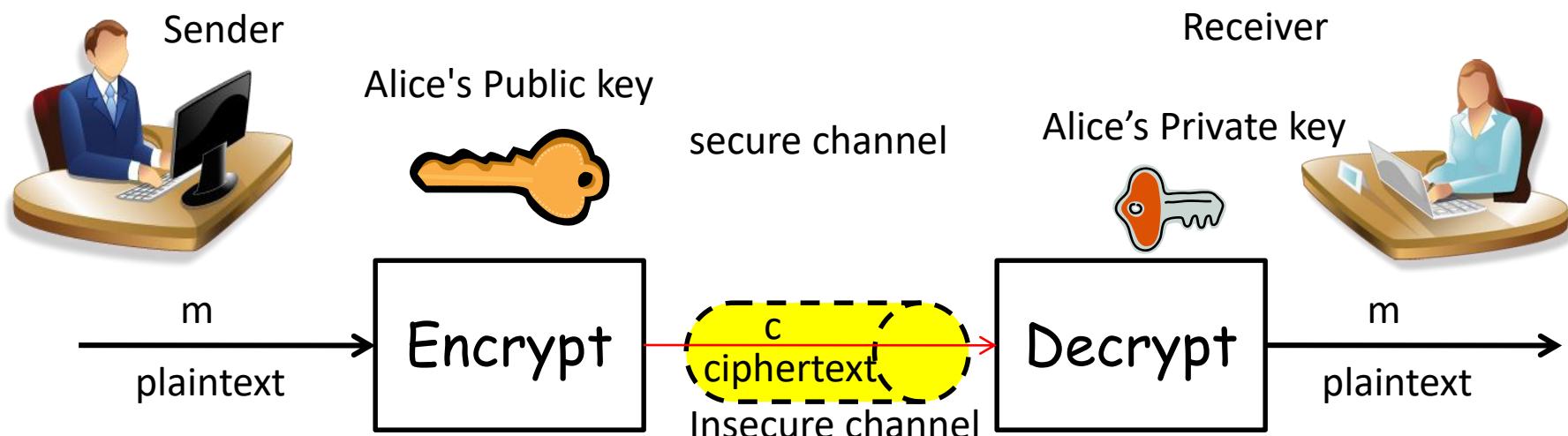
Lecturer: Rongxing LU

Email: RLU1@unb.ca Office: GE 114

Website: <http://www.cs.unb.ca/~rlu1/>

Faculty of Computer Science, University of New Brunswick

Review of RSA Cryptosystem



- p and q are large primes.
- $n = p \cdot q$
- $\varphi(n) = (p - 1)(q - 1)$
- e is an integer, $1 < e < \varphi(n)$, $\gcd(e, \varphi(n)) = 1$
- d is an integer, $1 < d < \varphi$, $e \cdot d \equiv 1 \pmod{\varphi}$
- (n, e) is public key
- (n, d) is private key

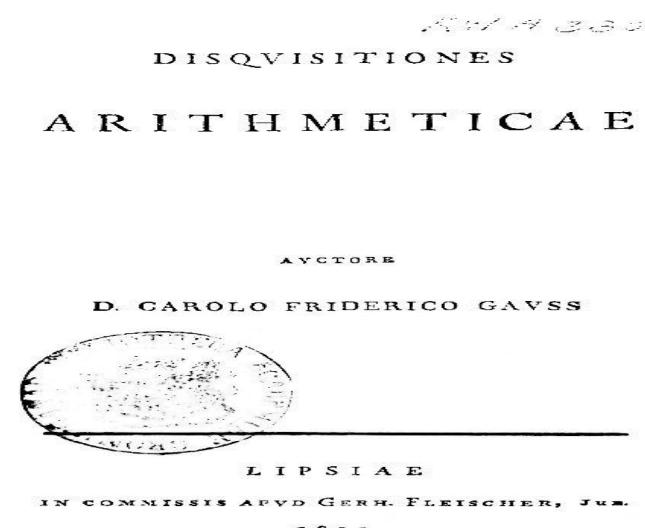
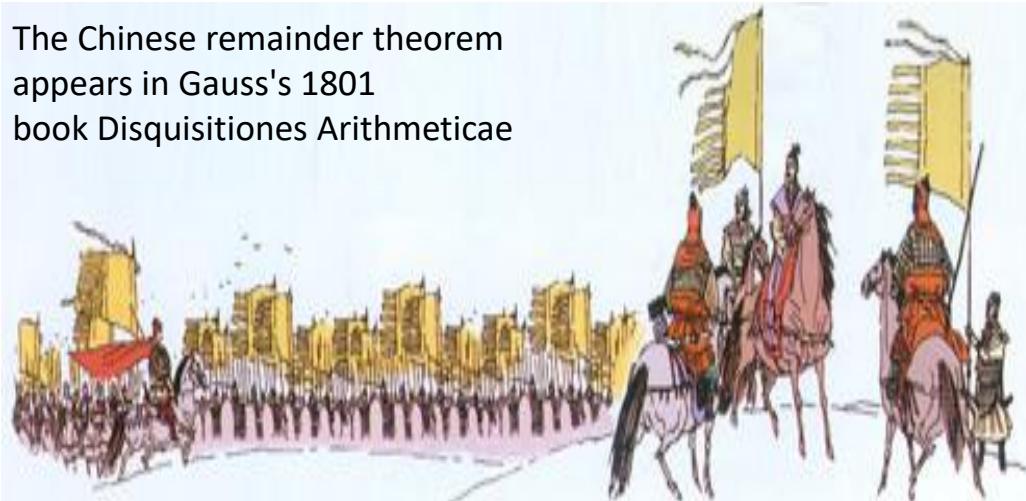
- Encryption : $c = m^e \pmod{n}$
- Decryption : $m = c^d \pmod{n}$
- m is plaintext, c is ciphertext.

$$\begin{aligned}m &= c^d \pmod{n} = m^{ed} \pmod{n} \\&= m^{1+k\varphi(n)} \pmod{n} = m \cdot (m^{\varphi(n)})^k \pmod{n} \\&= m \cdot 1 \pmod{n} = m\end{aligned}$$

RSA Decryption is Slow

- p and q are large primes.
 - $n = p \cdot q$ $\varphi(n) = (p - 1)(q - 1)$
 - e is an integer, $1 < e < \varphi(n)$, $\gcd(e, \varphi(n)) = 1$
 - d is an integer, $1 < d < \varphi$, $e \cdot d \equiv 1 \pmod{\varphi}$
 - (n, e) is public key (n, d) is private key
 - Encryption : $c = m^e \pmod{n}$
 - Decryption : $m = c^d \pmod{n}$
 - m is plaintext, c is ciphertext.
- $p=179, q=197$
 - $n = 35263 \quad \varphi(n) = 34888$
 - $e = 17$
 - $d = 8209$
- $c=m^e \pmod{n} = 168^{17} \pmod{35263} = 28657$
 - $m=c^d \pmod{n} = 28657^{8209} \pmod{35263} = 168$
-
- In RSA cryptosystem, the public key e is small, then the private key d will become large. Then, the RSA decryption is relatively slow.
 - Can we accelerate the RSA decryption?
 - The answer is Yes, we can use Chinese Remainder Theorem

The Chinese remainder theorem appears in Gauss's 1801 book *Disquisitiones Arithmeticae*



A general Han leads 1500 soldiers for a battle. After the battle, it is estimated that 400-500 soldiers are dead, then for the rest soldiers, if stand 3 for row, left 2; stand 5 for row, left 4; stand 7 for row, left 6. Then, Han quickly get the accurate number 1049.

$$\begin{cases} x \equiv 2 \pmod{3} \\ x \equiv 4 \pmod{5} \\ x \equiv 6 \pmod{7} \end{cases} \quad x \equiv 1049$$

The **Chinese remainder theorem** is a result about [congruences in number theory](#) and its generalizations in [abstract algebra](#). It was first published in the 3rd to 5th centuries by the Chinese mathematician [Sun Tzu](#).

Chinese Remainder Theorem (CRT)

- CRT is used to speed up modulo computations
- Let m_1, m_2, \dots, m_k be pairwise relatively prime integers, i.e., $\gcd(m_i, m_j) = 1$ for $1 \leq i < j \leq k$. Let $a_i \in \mathbb{Z}_{m_i}$ for $1 \leq i \leq k$ and set $M = m_1 m_2 \dots m_k$. There exists a unique $A \in \mathbb{Z}_M$ such that $A \equiv a_i \pmod{m_i}$ for $i = 1 \dots k$.
- A can be computed as:

$$A \equiv \left(\sum_{i=1}^k a_i c_i \right) \pmod{M}$$

- Where $c_i = M_i^{-1} \pmod{m_i}$ and $M_i = \frac{M}{m_i}$ for $1 \leq i \leq k$

CRT Proof

- Since $M_i = \frac{M}{m_i} = \frac{m_1 \times m_2 \times \cdots \times m_k}{m_i} = m_1 \times m_2 \times \cdots \times m_{i-1} \times m_{i+1} \times \cdots \times m_k$

$$c_i = [M_i \times (M_i^{-1} \bmod m_i)] \bmod m_i = [M_i \bmod m_i \times (M_i^{-1} \bmod m_i)] \bmod m_i = 1 \bmod m_i$$

$$c_i = [M_i \times (M_i^{-1} \bmod m_i)] \bmod m_j$$

$$= [m_1 \times m_2 \times \cdots \times m_{i-1} \times m_{i+1} \times \cdots \times m_k \times (M_i^{-1} \bmod m_i)] \bmod m_j = 0$$

- We have $c_i = M_i \times (M_i^{-1} \bmod m_i) = \begin{cases} 1 \bmod m_i \\ 0 \bmod m_j \text{ where } j \neq i \end{cases}$

- Therefore, $A \equiv \left(\sum_{i=1}^k a_i c_i \right) \bmod M$

$$= a_1 c_1 + a_2 c_2 + \cdots + a_k c_k + l \times M = a_i \bmod m_i$$

- where $1 \leq i \leq k$

CRT Proof (Cont.)

- A is unique in \mathbb{Z}_M
- Suppose A is not the unique solution, there must exist another answer $A' \equiv a_i \pmod{m_i}$ in \mathbb{Z}_M .

$$\begin{cases} A \equiv a_1 \pmod{m_1} \\ \vdots \\ A \equiv a_i \pmod{m_i} \\ \vdots \\ A \equiv a_k \pmod{m_k} \end{cases}$$

$$\begin{cases} A' \equiv a_1 \pmod{m_1} \\ \vdots \\ A' \equiv a_i \pmod{m_i} \\ \vdots \\ A' \equiv a_k \pmod{m_k} \end{cases}$$

- Then $A \equiv A' \pmod{m_i}$
 - $A - A' = r_1 * m_1 = r_2 * m_2 = \dots r_k * m_k$
 - $r_i | m_j$ where $i \neq j$ (since m_i 's are relatively prime)
 - $m_1 | A - A', m_2 | A - A', \dots, m_k | A - A'$ ----(1)
 - $M | A - A'$. Suppose $M \nmid A - A'$, there exist at least one $m_i \nmid A - A'$, which contradicts with (1). Therefore,
 - $M | A - A' \Rightarrow A \equiv A' \pmod{M}$, which shows A is unique in \mathbb{Z}_M

Example

- Find a number x such that have remainders of 1 when divided by 3, 2 when divided by 5 and 3 when divided by 7. i.e.

$$\begin{cases} x \equiv 1 \pmod{3} \\ x \equiv 2 \pmod{5} \\ x \equiv 3 \pmod{7} \end{cases}$$

- First, we know $M=3*5*7=105$, $M_1=M/m_1=35$, $M_2=M/m_2=21$, $M_3=M/m_3=15$

$$x = \sum_{i=1}^3 a_i \cdot [M_i \cdot (M_i^{-1} \pmod{m_i})] \pmod{M}$$

$$\begin{aligned} &= 1 \times 35 \times 2 + 2 \times 21 \times 1 + 3 \times 15 \times 1 \pmod{105} \\ &= 52 \end{aligned}$$

Example (2)

- Given $1813 = 37 * 49$, Represent 973 in \mathbb{Z}_{1813} as a 2-tuple
- $M=1813, m_1=37, m_2=49$
- $A=973$
- $A \bmod m_1 = 973 \bmod 37 = 11$
- $A \bmod m_2 = 973 \bmod 49 = 42$
- Therefore, $A = (11 \bmod 37, 42 \bmod 49)$

Example (3)

- Give $x = 11 \bmod 37$ & $x = 42 \bmod 49$, find x .
- $m_1=37$, $m_2=49$, $M=m_1*m_2=37*49=1813$,
 $M_1=M/m_1=49$, $M_2=M/m_2=37$
- $M_1^{-1} \bmod m_1 = 49^{-1} \bmod 37 = 34$
 $M_2^{-1} \bmod m_2 = 37^{-1} \bmod 49 = 4$

$$\begin{aligned}x &= \sum_{i=1}^2 a_i \cdot [M_i \cdot (M_i^{-1} \bmod m_i)] \bmod M \\&= 11 \times 49 \times 34 + 42 \times 37 \times 4 \bmod 1813 \\&= 18326 + 6216 \bmod 1813 \\&= 24542 \bmod 1813 = 973\end{aligned}$$

CRT Speed Up RSA Decryption

- p and q are large primes.
 - $n = p \cdot q$ $\varphi(n) = (p - 1)(q - 1)$
 - e is an integer, $1 < e < \varphi(n)$, $\gcd(e, \varphi(n)) = 1$
 - d is an integer, $1 < d < \varphi$, $e \cdot d \equiv 1 \pmod{\varphi}$
 - (n, e) is public key (n, d) is private key
 - Encryption : $c = m^e \pmod{n}$
 - Decryption : $m = c^d \pmod{n}$
 - m is plaintext, c is ciphertext.
- $p=179, q=197$
 - $n = 35263 \quad \varphi(n) = 34888$
 - $e = 17, d = 8209$
 - $d_1 = d \pmod{p-1} = 8209 \pmod{178} = 21$
 - $d_2 = d \pmod{q-1} = 8209 \pmod{196} = 173$
 - $c = m^e \pmod{n} = 168^{17} \pmod{35263} = 28657$
 - $m = c^d \pmod{n} = 28657^{8209} \pmod{35263} = 168$

$$\begin{aligned}c &= 28657 \quad c \pmod{p} = 28657 \pmod{179} = 17 \\&\quad c \pmod{q} = 28657 \pmod{197} = 92\end{aligned}$$

$$\left\{ \begin{array}{l} m \equiv c^{d_1} \equiv 17^{21} \pmod{179} \equiv 168 \pmod{179} \\ m \equiv c^{d_2} \equiv 92^{173} \pmod{197} \equiv 168 \pmod{197} \end{array} \right. \quad \left\{ \begin{array}{l} c \equiv 17 \pmod{179} \\ c \equiv 92 \pmod{197} \end{array} \right.$$

CRT Speed Up RSA Decryption

$$\begin{aligned} c &= 28657 \quad c \bmod p = 28657 \bmod 179 = 17 \\ &\quad c \bmod q = 28657 \bmod 197 = 92 \end{aligned} \quad \left\{ \begin{array}{l} c \equiv 17 \pmod{179} \\ c \equiv 92 \pmod{197} \end{array} \right.$$

$$\left\{ \begin{array}{l} m \equiv c^{d_1} \equiv 17^{21} \pmod{179} \equiv 168 \pmod{179} \\ m \equiv c^{d_2} \equiv 92^{173} \pmod{197} \equiv 168 \pmod{197} \end{array} \right. \xrightarrow{\text{Coincidence}}$$

$$m_1 = 179, m_2 = 197, M = m_1 * m_2 = 35263$$

$$M_1 = M / m_1 = 197, M_2 = M / m_2 = 179$$

$$M_1^{-1} \bmod m_1 = 197^{-1} \bmod 179 = 10$$

$$M_2^{-1} \bmod m_2 = 179^{-1} \bmod 197 = 186$$

$$x = \sum_{i=1}^3 a_i \cdot [M_i \cdot (M_i^{-1} \bmod m_i)] \bmod M$$

$$= 168 \times 197 \times 10 + 168 \times 179 \times 186 \bmod 35263$$

$$= 168$$

RSA is one-way secure, but is not semantic secure under Chosen Plaintext attack

- p and q are large primes.
- $n = p \cdot q$
- $\phi(n) = (p - 1)(q - 1)$
- e is an integer, $1 < e < \phi(n)$, $\gcd(e, \phi(n)) = 1$
- d is an integer, $1 < d < \phi$, $e \cdot d \equiv 1 \pmod{\phi}$
- (n, e) is public key
- (n, d) is private key
- Encryption : $c = m^e \pmod{n}$
- Decryption : $m = c^d \pmod{n}$
- m is plaintext, c is ciphertext.



1. The adversary chooses two messages m_0, m_1
2. Someone randomly flips a bit $b \in \{0,1\}$, and encrypts $c = m_b^e \pmod{n}$, returns c to the adversary

If the adversary cannot tell the plaintext of c , we call the encryption algorithm is semantic secure.

3. However, by computing $c_1 = m_0^e \pmod{n}$ and $c_2 = m_1^e \pmod{n}$, and compare c with (c_1, c_2) , the adversary can know the plaintext. Therefore, RSA is not semantic secure Under chosen plaintext attack.

Question: Is there a semantic secure algorithm? Yes

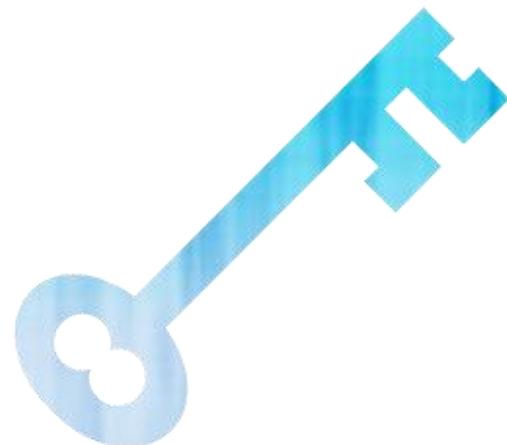
$$c = m^e \pmod{n}$$



(one-way secure) Given $c = m^e \pmod{n}$, without Knowing the private key d , the adversary cannot recover m from c .

Introduction of ElGamal

- Mathematic Background of ElGamal
- ElGamal Cryptosystem



Taher Elgamal

Taher El Gamal. 1985. A public key cryptosystem and a signature scheme based on discrete logarithms. In *Proceedings of CRYPTO 84 on Advances in cryptology*, G R Blakley and David Chaum (Eds.). Springer-Verlag New York, Inc., New York, NY, USA, 10-18.

Primary Root

- Consider $Z_7^* = \{1, 2, 3, 4, 5, 6\}$, we know $3^1 \equiv 3 \pmod{7}$, $3^2 \equiv 2 \pmod{7}$, $3^3 \equiv 6 \pmod{7}$, $3^4 \equiv 4 \pmod{7}$, $3^5 \equiv 5 \pmod{7}$, $3^6 \equiv 1 \pmod{7}$. We obtain all the nonzero elements modulo 7 as powers of 3. This means that 3 is a primitive root modulo 7.
- However, $3^3 \equiv 1 \pmod{13}$, so only 1, 3, 9 are the powers of 3. Therefore, 3 is not a primitive root modulo 13.
- In general, when p is a prime, a primitive root modulo p is a number α whose powers yield every nonzero elements modulo p .

$\alpha^m \equiv n \pmod{p}$, for $1 \leq n < p$
- There are $\varphi(p-1)$ primitive root modulo p .

Discrete Logarithm Problem (DLP)

- Let p a large prime, g is primitive root of \mathbb{Z}_p^*
- DLP: Given p , g and $(g^a \bmod p)$, determine a .
- DLP in \mathbb{Z}_p^* is hard, that is, there is no an efficient algorithm which can determine a in a polynomial time.

Given $(p, g, x \in [1, p-1])$, it is easy to compute
 $y = g^x \bmod p$

However, given $(p, g, y = g^x \bmod p)$, it is hard to compute

x such that $y = g^x \bmod p$

ElGamal Cryptosystem

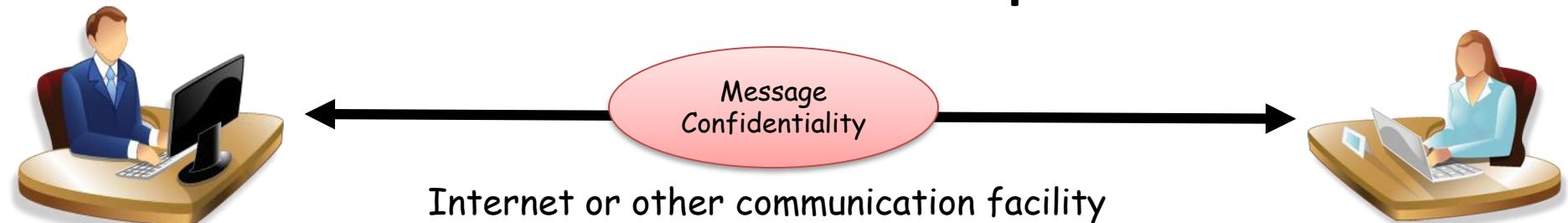
- p is a large prime, g is a generator (primitive root) of \mathbb{Z}_p^*
- x is a random number in $[1, p-1]$
- $y = g^x \text{ mod } p$
- (p, g) are public parameters
- y is a public key
- x is a private key (Because of the hardness of DLP, it is hard to get the private key x from the public key y)

ElGamal Cryptosystem (Cont.)

- Encryption :
 - Choose a random number k in $(1, p-1)$, and compute $K = Y^k \bmod p$
 - Encrypt a message M in Z_{p^*} as $C = (C_1, C_2)$, where ($C_1 = g^k \bmod p$, $C_2 = M * K \bmod p$)
- Decryption :
 - Use the private key x to compute $C_1^x = g^{kx} = Y^k = K \bmod p$
 - Recover $M = C_2 / K \bmod p$

Because of the hardness of DLP, it is hard to get x, k from $Y = g^x \bmod p$ and $K = Y^k \bmod p$

ElGamal Example



$$m = 7$$

$$p = 13, g = 2$$

$$SK : x = 5$$

$$PK : Y = g^x \bmod 13 = 2^5 \bmod 13 = 6$$

$$k = 4, K = Y^k \bmod 13 = 6^4 \bmod 13 = 9$$

$$C_1 = g^k \bmod 13 = 2^4 \bmod 13 = 3$$

$$C_2 = m \cdot K \bmod 13 = 7 \cdot 9 \bmod 13 = 11$$

$$C = (C_1, C_2) = (3, 11)$$

In real system, p
should be large

$$C_1^x \bmod 13 = 3^5 \bmod 13 = 9 = K$$

$$\begin{aligned} m &= C_2 / K \bmod 13 = 11 \times 9^{-1} \bmod 13 \\ &= 11 \times 3 \bmod 13 = 7 \end{aligned}$$

ElGamal Cryptosystem is Semantic secure under chosen plaintext attack



1. The adversary chooses two messages (m_0, m_1) in Z_p^*
2. Someone randomly flips a bit $b \in \{0,1\}$, and encrypts m_b as,

$$C = (C_1 = g^k \bmod p, C_2 = Y^k \cdot m_b \bmod p)$$

returns C to the adversary

3. For the same message m_b , different random numbers k will produce the different ciphertexts. Without knowing the random number k , it is impossible for the adversary to produce the same ciphertext. Therefore, the adversary cannot tell the corresponding plaintext of C . As a result, ElGamal cryptosystem is semantic secure.

Message
Confidentiality

The padding is required (Cont.)

 (p, g)  $m = \text{"hello world"}$ $SK : x$ $PK : Y = g^x \bmod p$

$$C = (C_1 = g^k \bmod p, C_2 = Y^k \cdot m \bmod p)$$

 C (C_1, C_2') 

$$C_2' = C_2 \cdot r \bmod p$$

$$\frac{C_2'}{C_1} = m \cdot r$$

Since m is meaningful, If the recovered is $m \cdot r$, which becomes meaningless. Then, the receiver can detect it.

However, if m is a session key (a random string), the receiver cannot identify the recovered $m \cdot r$ is correct or not.

The padding is required (Cont.)

 (p, g)  m $SK : x$ $PK : Y = g^x \bmod p$

$$C = (C_1 = g^k \bmod p, C_2 = Y^k \cdot m \parallel H(m) \bmod p)$$

 $C \quad (C_1, C_2')$ 

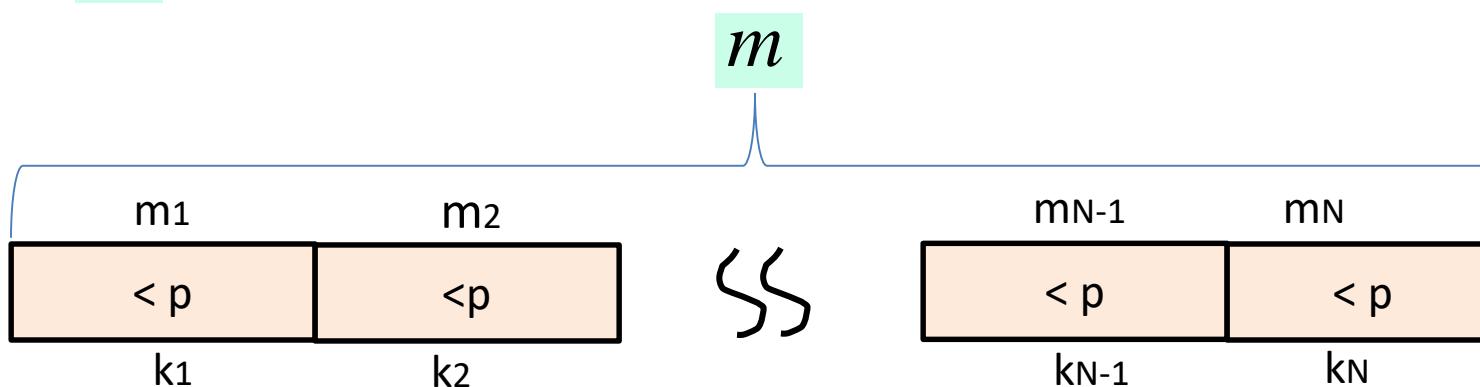
$$C_2' = C_2 \cdot r \bmod p$$

$$\frac{C_2'}{C_1} = m \parallel b$$

check $H(m) = b$

With the padding, the receiver can detect whether the message has Modified during the transmission.

How to encrypt a long message?

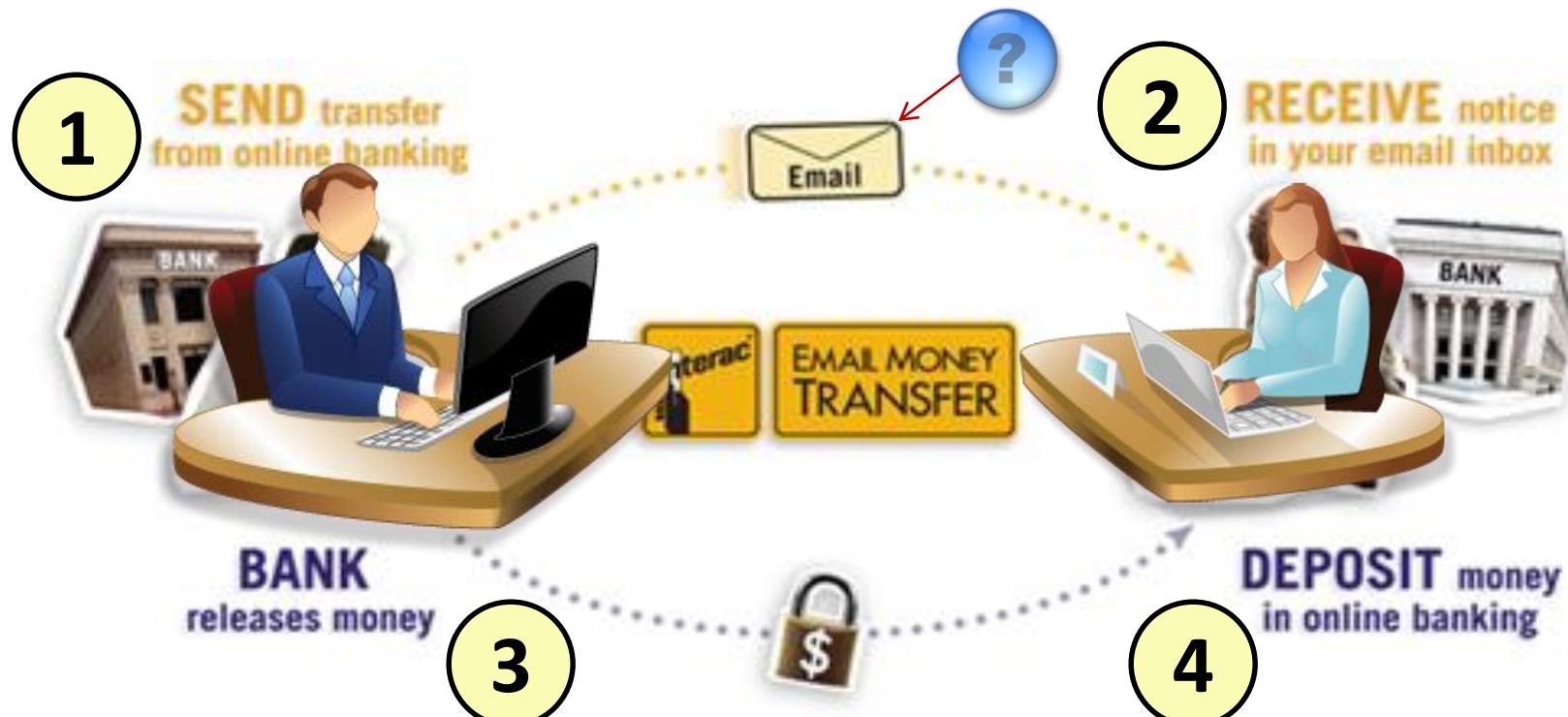
 (p, g)  m $SK : x$ $PK : Y = g^x \bmod p$ 

- If m is large than p , then m should be first divided into $m=m_1m_2m_3\dots$, each $m_i < p$.
- Each block should use different random number k . Why?
- Any other solution?

Diffie-Hellman Key Exchange

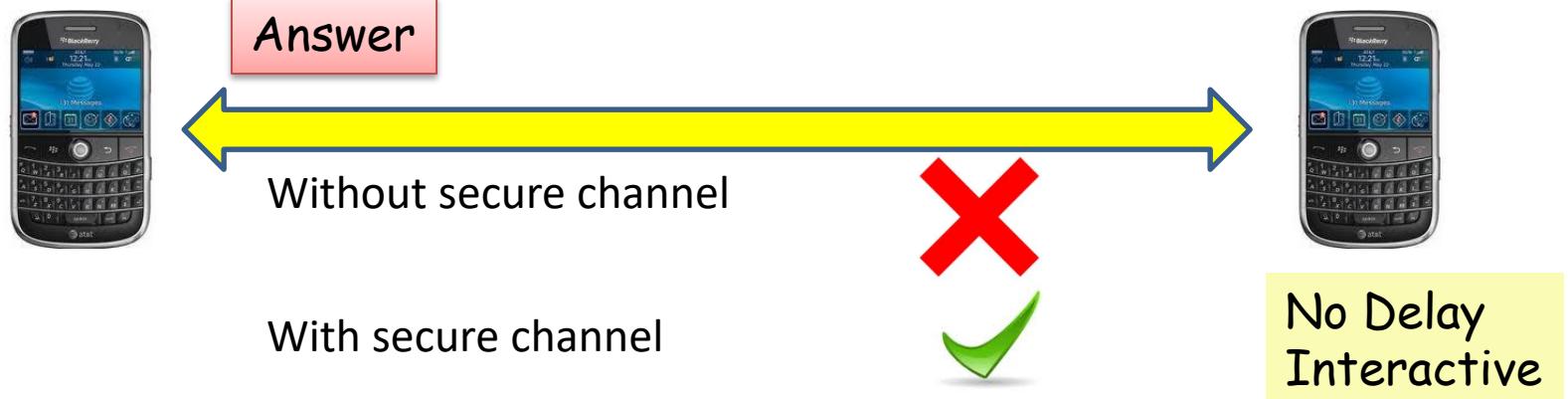
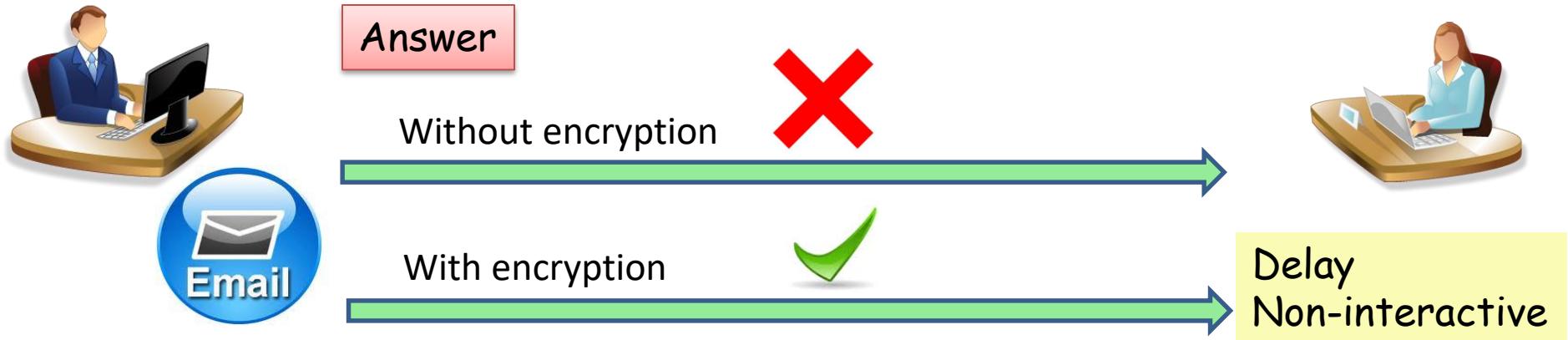


Secure Email Money Transfer



- Before Bank releases money, the receiver should first answer a question listed in the email. If the answer is correct, Bank will release money. Otherwise, stop.
- Then, how does the sender share the answer to the receiver?

Secure Email Money Transfer



Question: How to establish such a secure channel?

Answer: Key Exchange

Diffie-Hellman Key Exchange



Ralph Merkle, Martin Hellman,
Whitfield Diffie (1977)



p : a large prime, g : a generator of order $p-1$
 (p, g)



Choose a random
number

$$x \in \{1, \dots, p-1\}$$

Compute

$$g^x \bmod p$$



Choose a random
number

$$y \in \{1, \dots, p-1\}$$

Compute

$$g^y \bmod p$$

$$K = (g^y)^x = g^{xy} \bmod p$$

$$K = (g^x)^y = g^{xy} \bmod p$$

W. Diffie and M. E. Hellman, New Directions in Cryptography, IEEE Transactions on Information Theory, vol. IT-22, Nov. 1976, pp: 644-654.

D-H Key Exchange Example



$$(p = 11, g = 7)$$

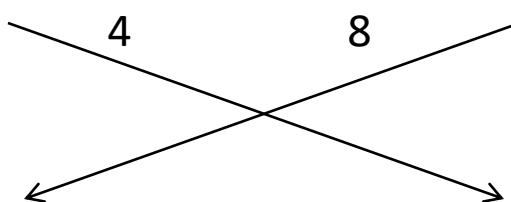


Choose a random number

$$x = 6$$

Compute

$$g^x \bmod p = 7^6 \bmod 11 = 4$$



$$K = (g^y)^x \bmod p = 8^6 \bmod 11 = 3$$

Choose a random number

$$y = 9$$

Compute

$$g^y \bmod p = 7^9 \bmod 11 = 8$$

$$K = (g^x)^y \bmod p = 4^9 \bmod 11 = 3$$

$$K = g^{xy} \bmod p = 7^{6 \times 9} \bmod 11 = 3$$

Is D-H Key Exchange Secure?

p : a large prime, g : a generator of order $p-1$



$$(p, g)$$

Choose a random number

$$x \in \{1, \dots, p-1\}$$

Compute

$$g^x \bmod p$$



Choose a random number

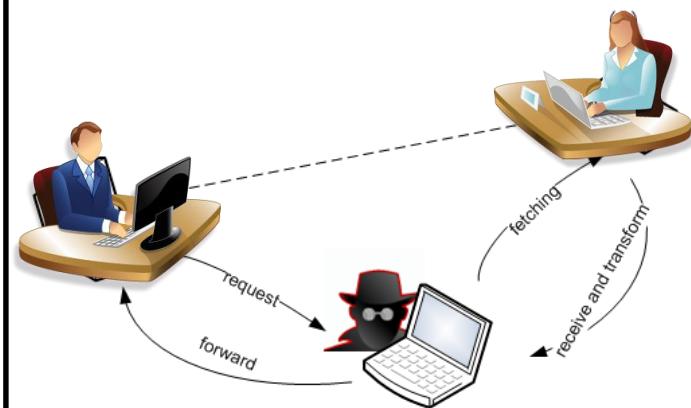
$$y \in \{1, \dots, p-1\}$$

Compute

$$g^y \bmod p$$

$$K = (g^y)^x = g^{xy} \bmod p$$

$$K = (g^x)^y = g^{xy} \bmod p$$



Man-in-the-Middle attack

- When p is a large prime, the Discrete Logarithm Problem is hard. Therefore, given $g^x \bmod p$, $g^y \bmod p$, the adversary cannot compute x, y . Then, the key $K=g^{xy} \bmod p$ is secure.
- However, the key is not authenticated. (Man-in-the-Middle attack)

Man-In-The-Middle Attack

p : a large prime, g : a generator of order $p-1$
 (p, g)



Choose a random number

$$x \in \{1, \dots, p-1\}$$

Compute

$$g^x \bmod p$$



Choose two random numbers

$$y', x' \in \{1, \dots, p-1\}$$



Choose a random number

$$y \in \{1, \dots, p-1\}$$

Compute

$$g^y \bmod p$$

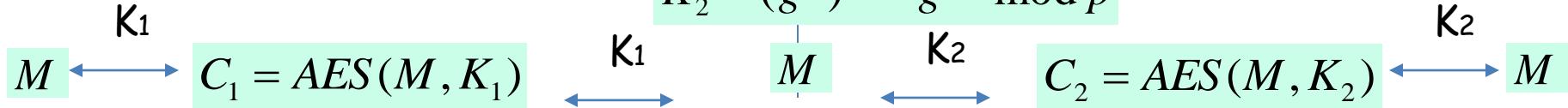
$$K_1 = (g^{y'})^x = g^{xy'} \bmod p$$

$$g^{y'} \bmod p, g^{x'} \bmod p$$

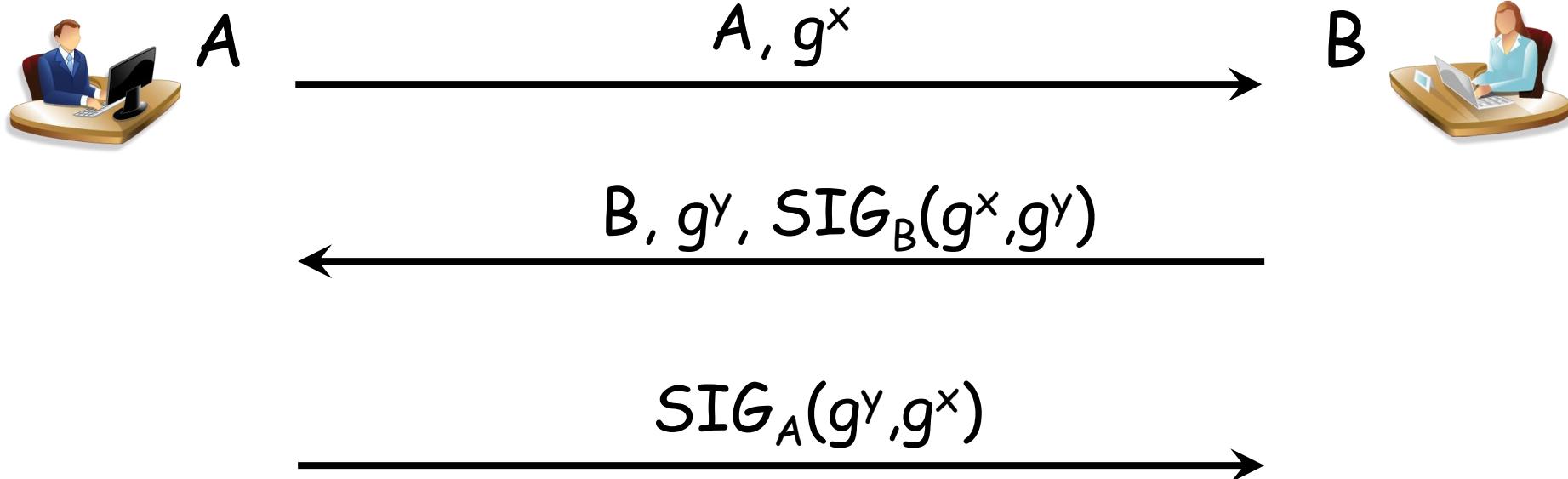
$$K_2 = (g^{x'})^y = g^{x'y} \bmod p$$

$$K_1 = (g^x)^{y'} = g^{xy'} \bmod p$$

$$K_2 = (g^y)^{x'} = g^{x'y} \bmod p$$



Basic Authenticated D-H Key Exchange



Each party signs its own DH value to prevent man-in-the-middle attack
(and the peer's DH value as a freshness guarantee against replay)

A: "Shared K= g^{xy} with B" ($K \leftrightarrow B$) B: "Shared K= g^{xy} with A" ($K \leftrightarrow A$)



Looks fine, but is it really secure ?



Identity-Misbinding Attack



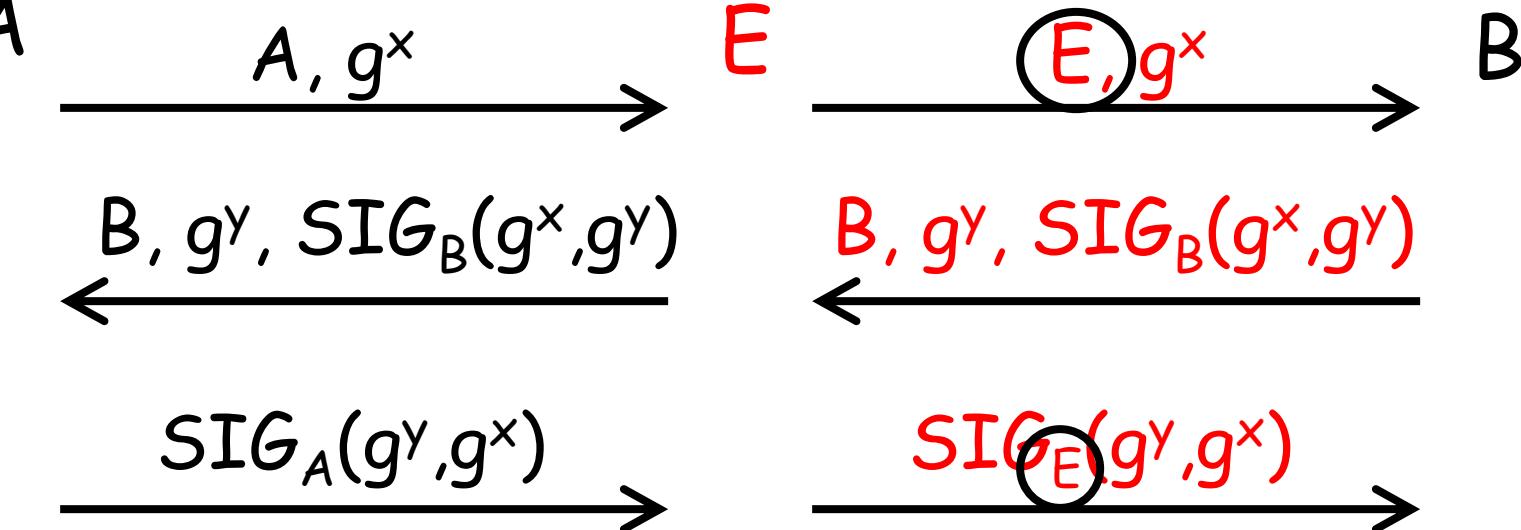
A



E



B



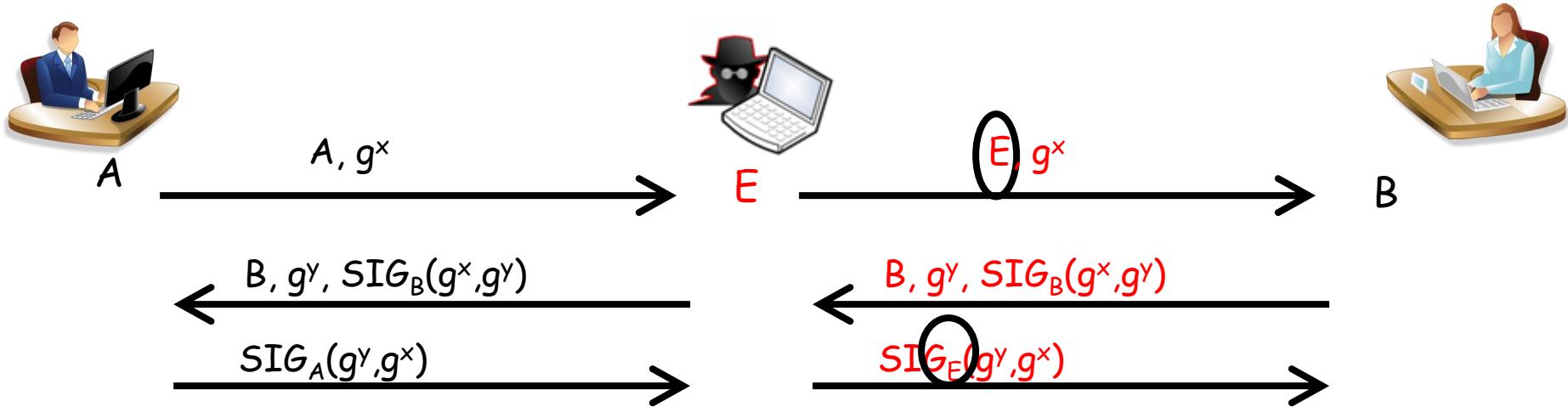
– Any damage? Wrong identity binding!

A: "Shared $K=g^{xy}$ with B" ($K \leftrightarrow B$)

B: "Shared $K=g^{xy}$ with E" ($K \leftrightarrow E$)

E doesn't know $K=g^{xy}$ but B considers anything sent by A as coming from E

Dangers of Identity-Misbinding Attack



- $B = \text{CIBC Bank}$ $A, E = \text{customers}$
- $A \rightarrow B: \{\text{"deposit \$2000 in my account"}\}_K$
- B deposits the money in " E "'s account!



- $B = \text{Central Command}$ $A = F-16$ $E = \text{small unmanned plane}$
- $B \rightarrow E: \{\text{"destroy yourself"}\}_K$
- E passes command to A A destroys itself



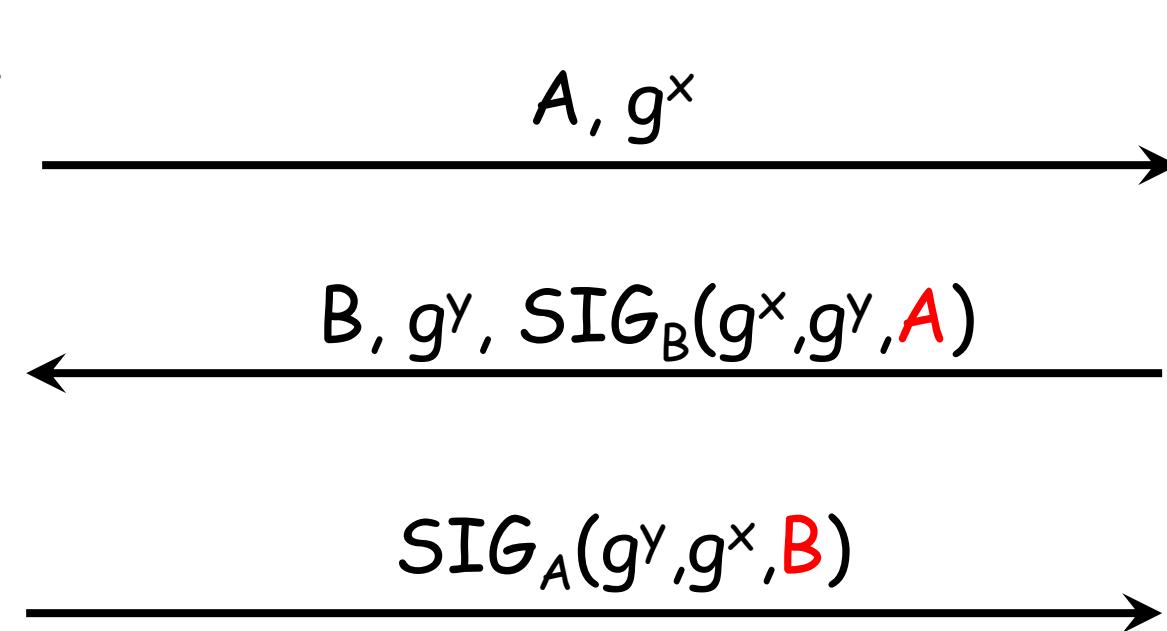
Solution for Thwarting Identity-Misbinding Attack



A

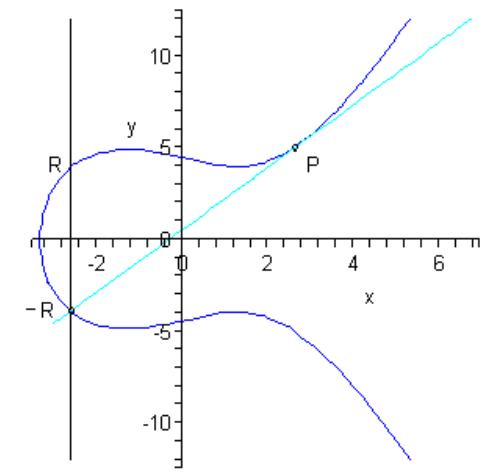
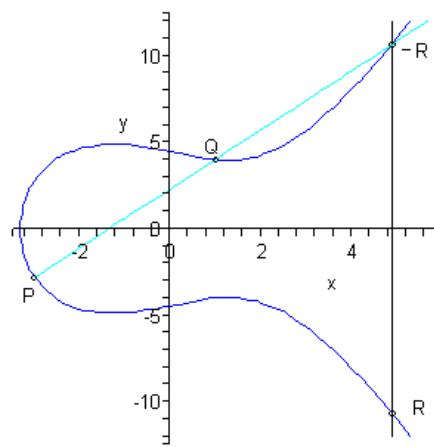
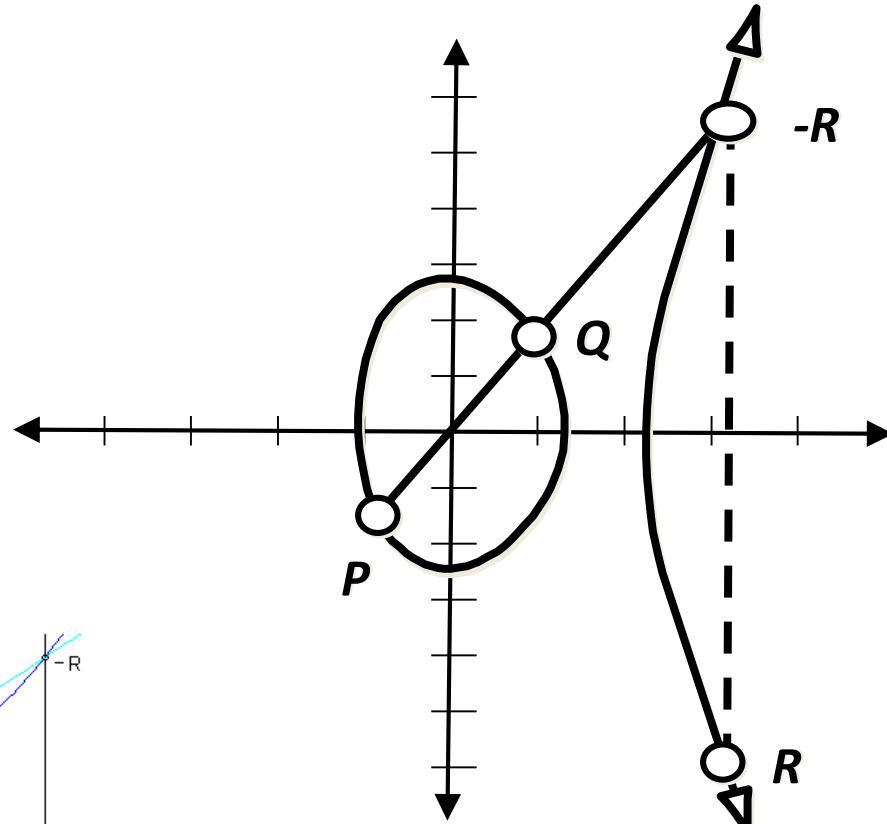


B



Including the identity of the peer under the signature to thwart the identity-Misbinding attack

Elliptic Curve Cryptography



Construction of Elliptic Curves

If p is an odd prime, where $p \geq 3$ and F_p is a group of non-zero elements under the operation of multiplication modulo p .

$$F_p = \{1, 2, \dots, p-1\}$$

Then an elliptic curve E over F_p is defined by the equation:

$$y^2 = x^3 + ax + b$$

where,

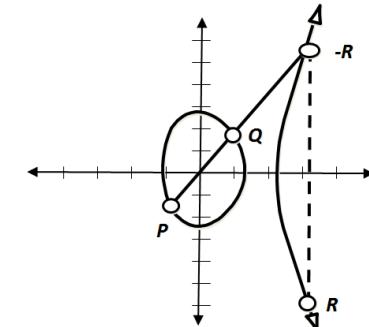
$a, b \in F_p$ and $4a^3 + 27b^2 \neq 0 \pmod{p}$, together with special point O , called as point at infinity.

Example

Let $p = 23$, $a = 1$, $b = 1$

We first check whether $4a^3 + 27b^2 = 8 \pmod{23} \neq 0$, then according to the elliptic curve $E : y^2 = x^3 + x + 1$, we calculate the points on $E(F_{23})$, which are $\textcolor{red}{O}$ and the followings:

- | | | | | | |
|--------|--------|---------|---------|---------|--------|
| (0,1) | (6,4) | (12,19) | (0,22) | (6,19) | (13,7) |
| (1,7) | (7,11) | (13,16) | (1,16) | (7,12) | (17,3) |
| (3,10) | (9,7) | (17,20) | (3,13) | (9,16) | (18,3) |
| (4,0) | (11,3) | (18,20) | (5,4) | (11,20) | (19,5) |
| | (5,19) | (12,4) | (19,18) | | |



Elliptic Curve Addition

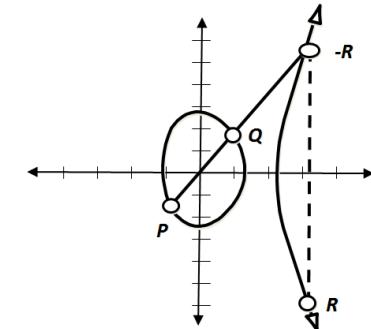
If $P, Q \in E(F_p)$ then,

1. $P + O = O + P = P$
2. If $P = (x, y)$ then $-P = (x, -y)$
and $P + (-P) = O$
3. If $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ and $P \neq -Q$ then,
 $P + Q = (x_3, y_3)$ where,

$$x_3 = \lambda^2 - x_1 - x_2 \text{ and } y_3 = \lambda(x_1 - x_3) - y_1$$

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1} \quad \text{if } P \neq Q,$$

$$\lambda = \frac{3x_1^2 + a}{2y_1} \quad \text{if } P = Q$$



Elliptic Curve Addition Group

If $P, Q \in E(F_p)$ then,

1. [Identity] $P + O = O + P = P$
2. [Inverse] If $P = (x, y)$ then $-P = (x, -y)$

and $P + (-P) = O$

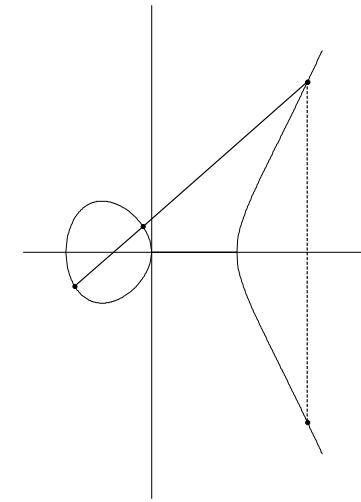
3. [Closure, Associativity, Commutativity]

If $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ and $P \neq -Q$ then,

$P + Q = (x_3, y_3)$ where,

$$x_3 = \lambda^2 - x_1 - x_2 \text{ and } y_3 = \lambda(x_1 - x_3) - y_1$$

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1} \text{ if } P \neq Q, \quad \lambda = \frac{3x_1^2 + a}{2y_1} \text{ if } P = Q$$



Scalar Multiplication in Elliptic Curve Addition Group

Scalar multiplication is repeated group addition:

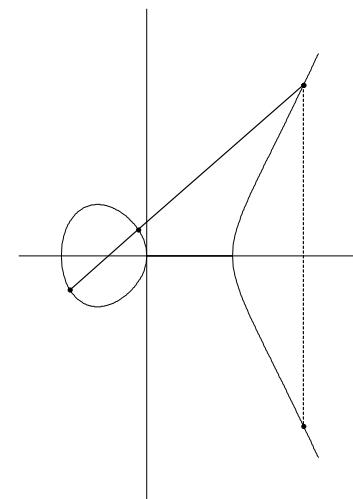
$$cP = P + \dots + P \text{ (c times)}$$

where c is an integer

For all $P \in E(F_q)$,

$$nP = O$$

where $n = \#E(F_q)$

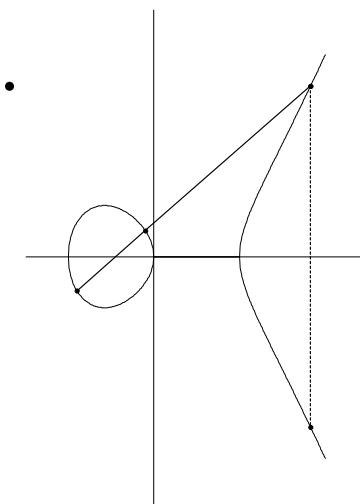


Elliptic Curve Discrete Logarithm Problem

- **ECDLP**: Given two points $Y = xP = P + \dots + P$, find x .

first suggested by Miller 1985,
Koblitz 1987

- With appropriate cryptographic restrictions, **ECDLP** is believed to take exponential time --- $O(\sqrt{r})$ time, where r is the order of Y



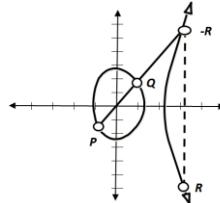
Hard Problem

Security Issues on ECDLP

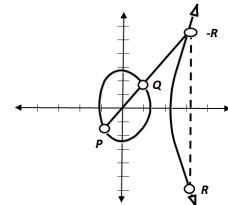
- The best algorithm known to date is Pollard rho-method which takes about $\sqrt{\pi n} / 2$ steps, where a step is an elliptic curve addition.
- Equivalent key sizes

Symmetric	ECC	DH/DSA/RSA
80	163	1024
128	283	3072
192	409	7680
256	571	15360

Advantages of ECC (Elliptic Curve Cryptography)



- Speed
 - EC operations are generally faster than Discrete Logarithm, Integer Factoring counterparts at comparable key sizes
 - Key pair generation is much faster than for Integer Factoring
- Data Size
 - EC data are shorter than Discrete Logarithm, Integer Factoring counterparts
 - Keys are shorter
 - Signatures with appendix are same size as for Discrete Logarithm, shorter than Integer Factoring
 - EC offers considerable bandwidth savings when being used to transform short messages.
- Disadvantage: ECC is mathematically more difficult to explain to client (i.e., Curve Generation)

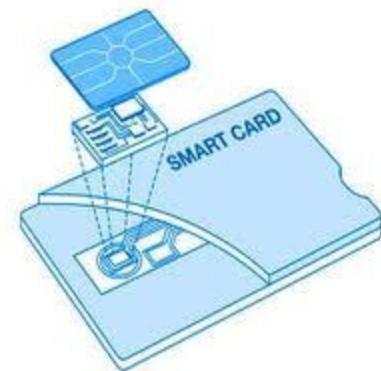


Applications of ECC

Implementations of ECC are particularly beneficial in applications where bandwidth, processing capacity, power availability, or storage is constrained.

Such applications include:

- Wireless transactions
- Handheld computing
- Broadcast and
- Smart card applications.



EC ElGamal Cryptosystem



$$y^2 = x^3 + ax + b \quad a, b \in F_p \text{ and } 4a^3 + 27b^2 \neq 0 \pmod{p}$$

P is a generator of order n in E(Fq)



$$SK : d_B \in [1, n-1]$$

$$PK : Q_B = d_B P$$

m Represent m as a point M in E(Fq)

$$r \in [1, n-1]$$

$$C = (C_1 = rP, C_2 = rQ_B + M)$$

$$C$$

$$M = C_2 - d_B C_1$$

Because

$$C_2 - d_B C_1 = rQ_B + M - d_B rP = rd_B P + M - d_B rP$$

Recover the message m from the Point M

EC Diffie Hellman Key Exchange

$E(F_p)$ point P in $E(F_p)$ of order n

$$(p, g)$$



Choose a random number

$$x \in [1, n-1]$$

Compute

$$xP$$

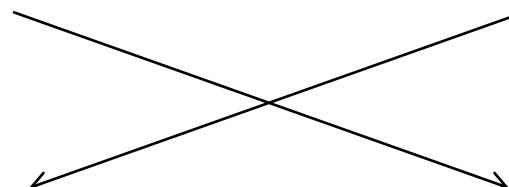


Choose a random number

$$y \in [1, n-1]$$

Compute

$$yP$$



$$K = x(yP) = xyP$$

$$K = y(xP) = xyP$$

Thank
you



CS 6355/4355: Cryptanalysis and Database Security

Topic 6: Hash Function, Message Authentication
Code, and Digital Signature

Lecturer: Rongxing LU

Email: RLU1@unb.ca Office: GE 114

Website: <http://www.cs.unb.ca/~rlu1/>

Faculty of Computer Science, University of New Brunswick

What are hash functions?

- Just a method of compressing strings
 - E.g., $H : \{0,1\}^* \rightarrow \{0,1\}^{160}$
 - Input is called "message", output is "digest"
- Why would you want to do this?
 - Short, fixed-size better than long, variable-size
 - ❖ True also for non-crypto hash functions
 - Digest can be added for redundancy
 - Digest hides possible structure in message

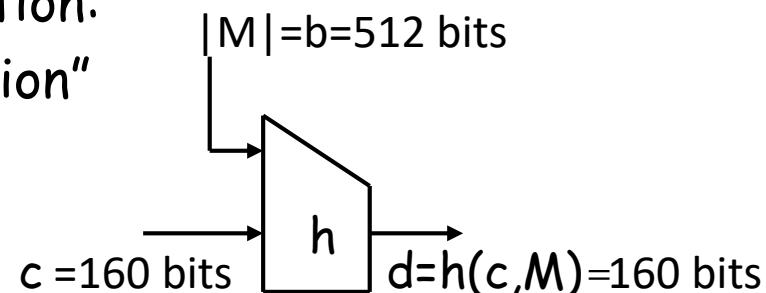


- Features of Hash Functions
 - Hash functions form an unique image of a message
 - The hash can be applied to any size of message to produce a fixed size output
 - The hash is easy and efficient to compute but is computationally infeasible to invert

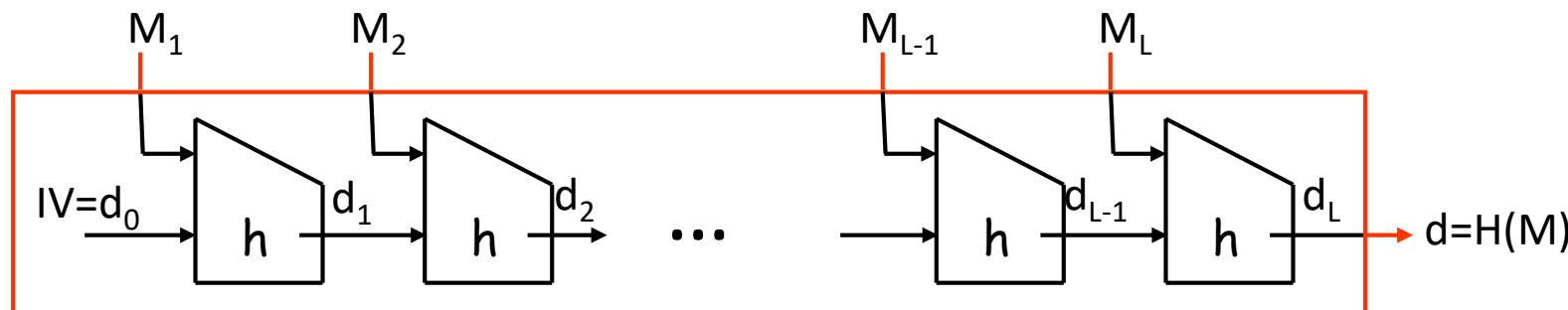
How are Hash functions built?

Typically using Merkle-Damgård iteration:

1. Start from a “compression function”
 - $h: \{0,1\}^{b+n} \rightarrow \{0,1\}^n$



2. Iterate it (Append to the message as few 0-bits as necessary to obtain a string to make its length a multiple of block-size)



3. Merkle-Damgård iteration: If an appropriate padding scheme is used and the compression function is collision-resistant, then the hash function will also be collision resistant

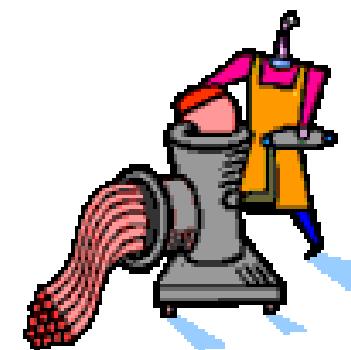
Classification of Hash Functions

- MIC (message integrity codes)
 - Unkeyed: no keys required, anyone can generate or verify
 - One-Way Hash Functions (OWHFs)
 - Given y , hard to find x such that $H(x)=y$
 - Collision Resistant Hash Functions (CRHFs)
 - Hard to find $x \neq x'$ such that $H(x)=H(x')$
- MAC (message authentication codes)
 - both authentication and integrity
 - Keyed: usually based on a cipher - users must exchange secret key in order to authenticate
 - requires no additional mechanism

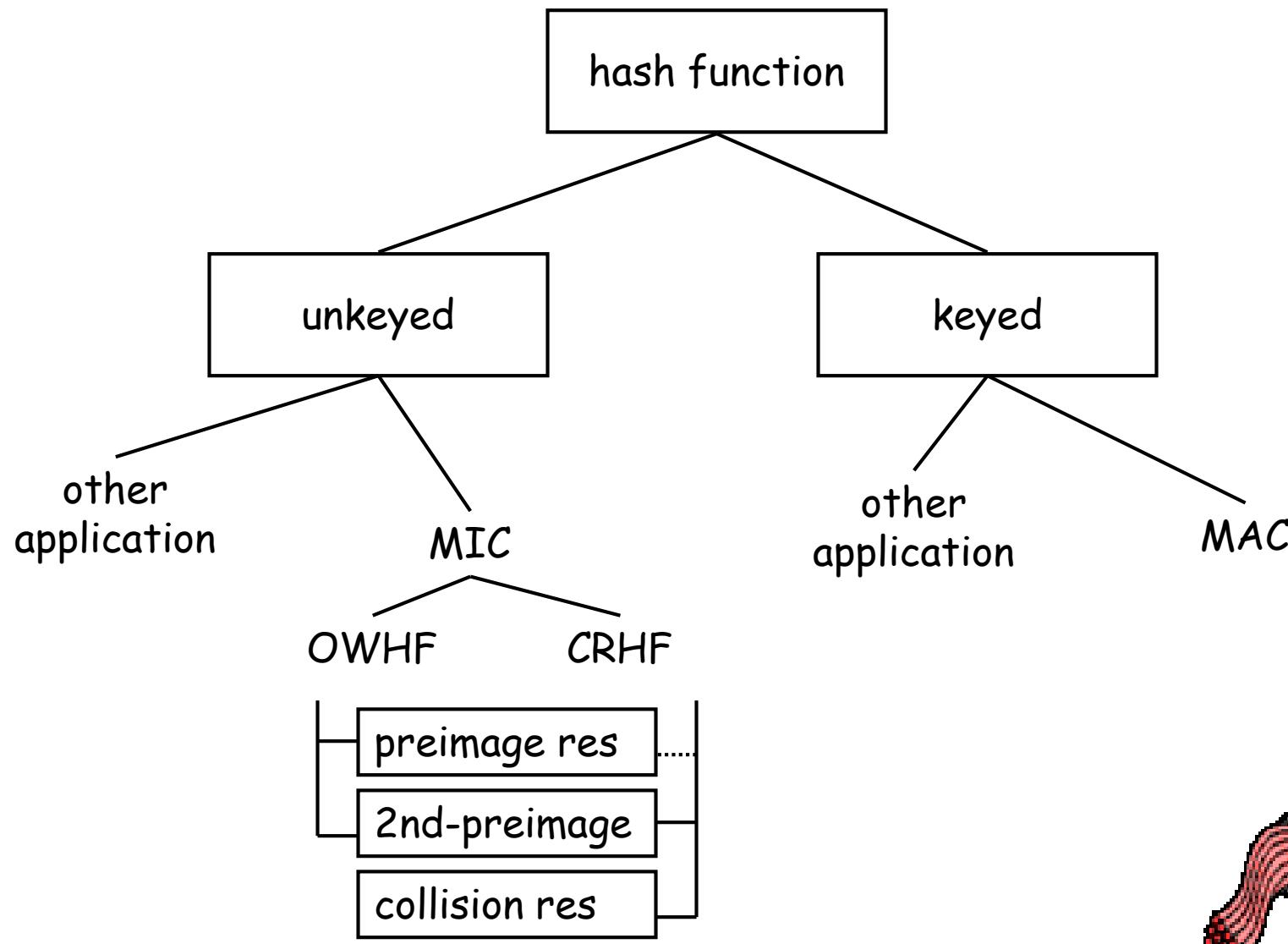


Properties of Hash Functions

- **Preimage resistance:** given y it's computationally infeasible to find a value x s.t. $h(x)=y$
- **2-nd preimage resistance:** given x and $y=h(x)$ it's computationally infeasible to find a value $x' \neq x$ s.t. $h(x')=h(x)$
- **Collision resistance:** it's computationally infeasible to find any two distinct values x',x s.t. $h(x')=h(x)$



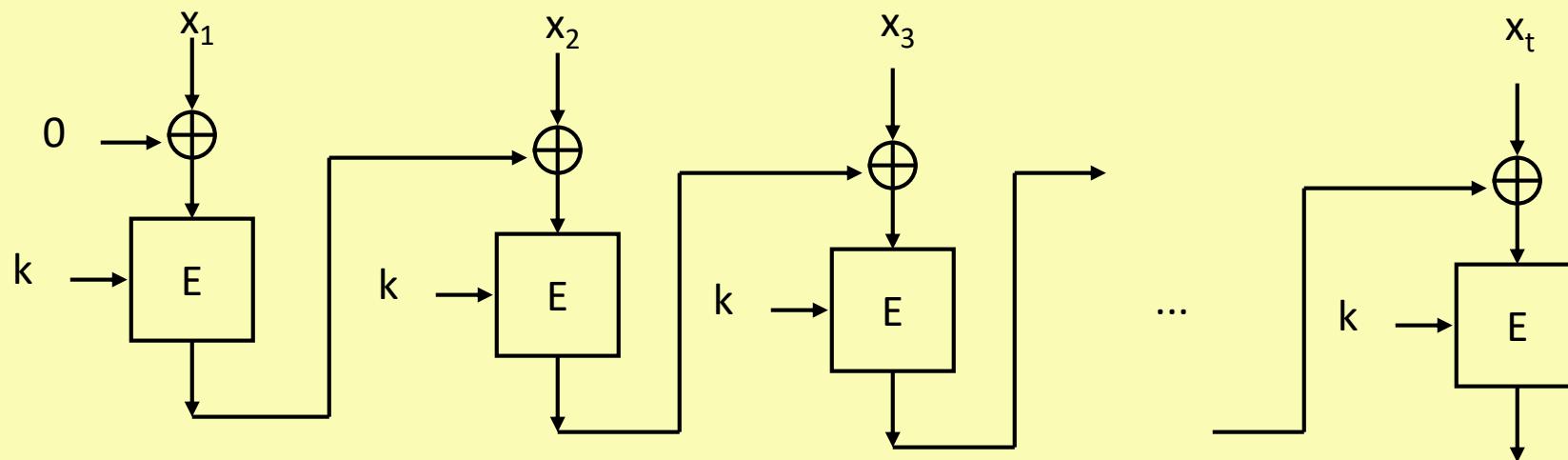
Classification of Hash Functions



Examples – DES MAC



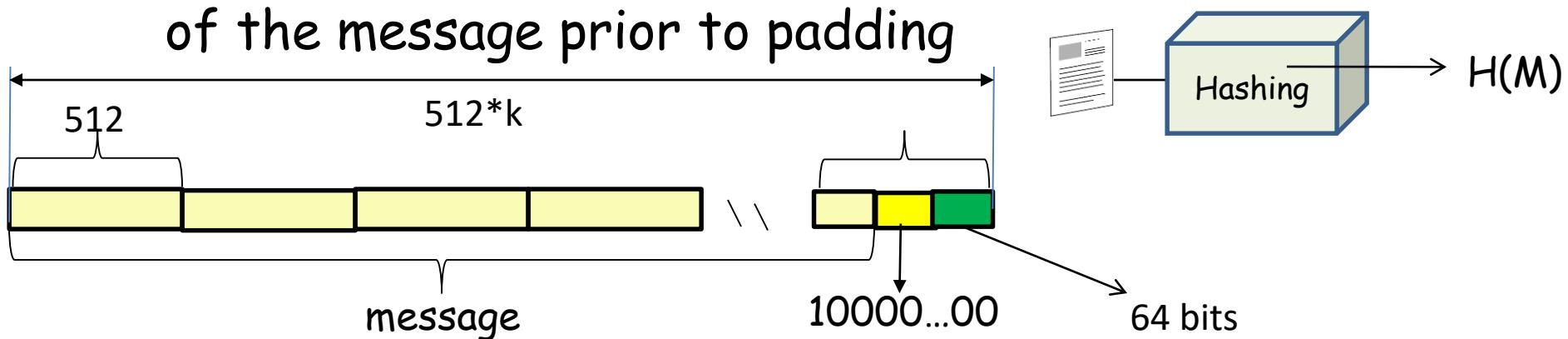
- DES can be used in Cipher Block Chaining (CBC) mode to produce a digest of the message
- This requires the exchange of a secret key between users
- Cannot prevent repudiation by one of the users



MD5



- Introduced in 1992 by Ron Rivest (RSA fame)
- Un-keyed hash
- Processes 512 bit input blocks - 128 bit output hash
- Initially, message is padded so that is 64 bits shorter than a multiple of 512 bits by adding a single "1" and then "0"s
- The last 64 bits are used to represent the length of the message prior to padding



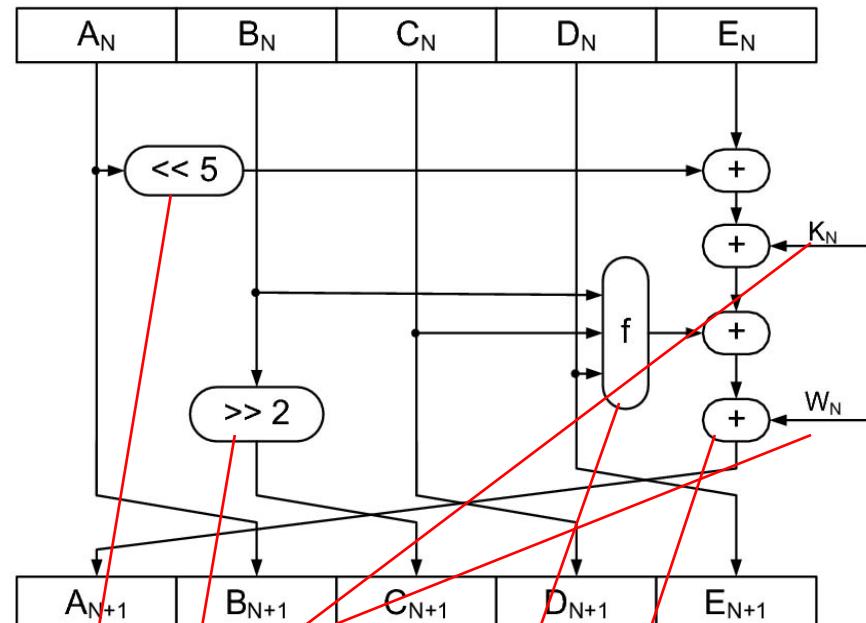
Secure Hash Algorithm (SHA)

- SHA-1(Secure Hash Algorithm)

- designed by the National Security Agency and published by the NIST as a U.S. Federal Information Processing Standard.
- iterated hash function
- 160-bit message digest
- word-oriented (32 bit) operation on bitstrings
- Padding scheme extends the input x by at most one extra 512-bit block
- The compression function maps 160+512 bits to 160 bits
- Make each input affect as many output bits as possible

expanded message word

Round constant



f is a nonlinear function

addition modulo 2^{32} .

a left bit rotation by 5 places

a right bit rotation by 2 places



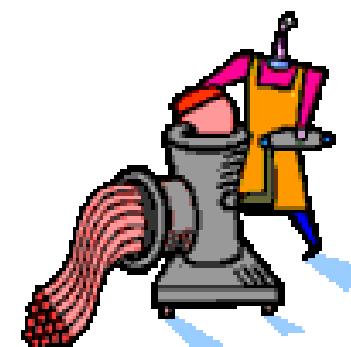
SHA

- SHA was modified to SHA-1 in 1992
- May 2001 - FIPS (Federal Information Processing Standards) 180-2 proposal for expansion to 256, 384, and 512 bit hash
- **Designed to match security of AES at 128, 192, and 256 bits**
- July 2007, NIST (National Institute of Standards and Technology) announced a call for the development of a new hash function -- SHA-3 function to replace the older SHA
 - The list of 14 candidates accepted to Round 2 was published on July 24, 2009
 - The announcement of the final round candidates occurred on December 10, 2010
 - and the proclamation of a winner and publication of the new standard are scheduled to take place in 2012.



Strength of Hash Functions

- If two messages can be found that create the same hash function, the validity of the real one could be disputed.
- Assume there is no weakness in the hash algorithm, how many tries would an attacker have to perform to find a message that produced the same MAC?



Birthday Attack

- Birthday paradox
 - In a group of 23 randomly chosen people, at least two will share a birthday with probability at least 50%. If there are 30, the probability is around 70%.
 - Finding two people with the same birthday is the same thing as finding a collision for this particular hash function.

$$P(n=23, m=365) = 1 - \frac{m^{(n)}}{m^n} = 1 - \frac{365^{(23)}}{365^{23}}$$

$$m^{(n)} = \frac{m!}{(m-n)!} = m \times (m-1) \times (m-2) \cdots (m-n+1)$$



Birthday Attack

- The probability that all 23 people have different birthdays is

$$1 \times \left(1 - \frac{1}{365}\right) \left(1 - \frac{2}{365}\right) \dots \left(1 - \frac{22}{365}\right) = 0.493$$

- Therefore, the probability of at least two having the same birthday is $1 - 0.493 = 0.507$
- Birthday Attack \rightarrow Square Root Attack

$$n \approx \sqrt{m} \Rightarrow 23 \approx \sqrt{365} \Rightarrow prob = 0.507$$



Birthday Attack



- The birthday attack (also called square root attacks) can be used to find collisions for hash functions if the output of the hash function is not sufficiently large.
 - Suppose h is an n -bit hash function. Then there are $N = 2^n$ possible outputs. We have the situation of list of length $r \approx \sqrt{N}$ "people" with N possible "birthdays," so there is a good chance of having two values with the same hash value
 - (**Old**) Banking standards used 32bit DES MACs - If you had the key, it would only require 2^{16} tries to find a message that would produce the same hash value
 - Today, hash value are at least 128 bits, preferably 160 bits to prevent square root attacks

Application 1: Secure Password

- password breaches

Morning Mix

Russian hackers steal more than 1 billion passwords. Security firm seizes opportunity.

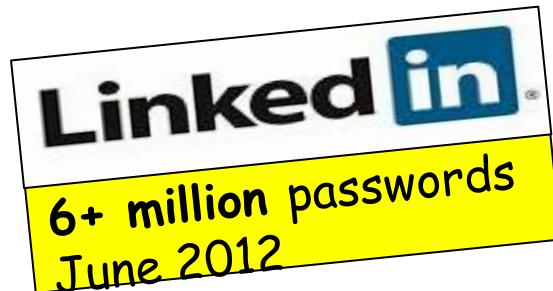


By Gail Sullivan August 6, 2014 Follow @g_forcewinds



A Wisconsin cybersecurity firm uncovered the theft of more than 1.2 billion usernames and passwords by Russian hackers. (Reuters)

<http://www.washingtonpost.com/news/morning-mix/wp/2014/08/06/russian-hackers-steal-a-billion-passwords-security-firm-seizes-opportunity/>



How to store a Password?

- Is a Password directly stored in Server?
- NO!

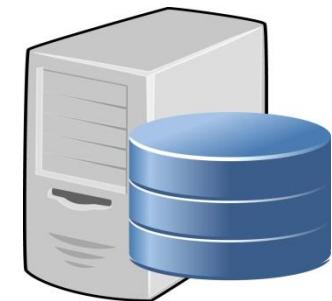
User Name	Password
Alice	qwert



Password Stored in Hashed Form

- P = Alice's password
- System stores mapping "Alice" $\rightarrow h(P)$ in database, for a suitable hash function h .
- When someone (perhaps Alice) tries to log in as Alice, system computes $h(P')$ of submitted password P' and compares it to $h(P)$. If equal, login is allowed.
- ``one-wayness'' of Hash function makes a stolen hash not so useful to adversary.

User Name	H(Password)
Alice	$H(P)=070jlsfd$



Defeat Against Pre-computation Attack

- To defeat precomputation attack, a per-user ``salt'' value s is used: system stores mapping "Alice" $\rightarrow (s, h(s, P))$. Hash $h(s, P')$ computed for submitted password P' and compared.
- Hashing with salting forces adversary who steals hashes and salts to find passwords by brute-force offline search: adversary repeatedly guesses P' until a P' is found such that $h(s, P') = h(s, P)$
- Ideally, it is great! However, in reality (real password space is not large enough)

Real passwords are often *weak* and *easily guessed*.

1. 123456
2. password
3. 12345
4. 12345678
5. qwerty
6. 123456789
7. 1234
8. baseball
9. dragon
10. football
11. 1234567
12. monkey
13. letmein
14. abc123
15. 111111
16. mustang
17. access
18. shadow
19. master
20. michael
21. superman
22. 696969
23. 123123
24. batman
25. trustno1

This List Of 2014's Worst Passwords, Including '123456,' Is Embarrassing

Posted Jan 20, 2015 by Jordan Crook (@jordanrcrook)

4,504
SHARES



A Wisconsin cybersecurity firm uncovered the theif passwords by Russian hackers. (Reuters)

<http://techcrunch.com/2015/01/20/this-list-of-2014s-worst-passwords-including-123456-is-embarrassing/>

35% of Users Have Weak Passwords; the Other 65% can be Cracked

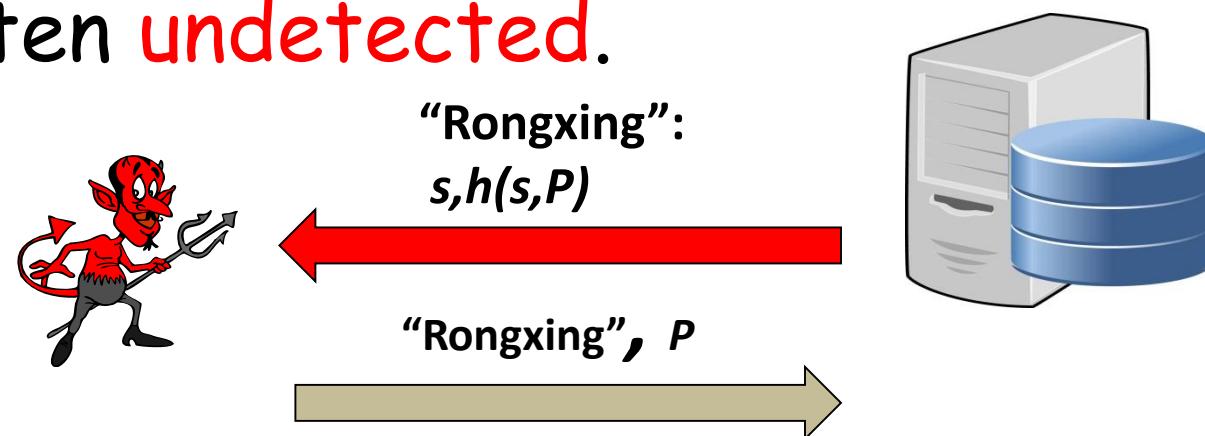
Posted by Eran Cohen on Mar 13, 2017 5:00:00 AM



Find me on: [in](#)

Password Stored in Hashed Form is insufficient!

- Adversary compromises system ephemerally, steals password hashes
- Adversary cracks hash, finding P
- Impersonate user(s) and logs in.
- Adversary almost always succeeds, and is often **undetected**.



“Honeywords” by Juels & Rivest

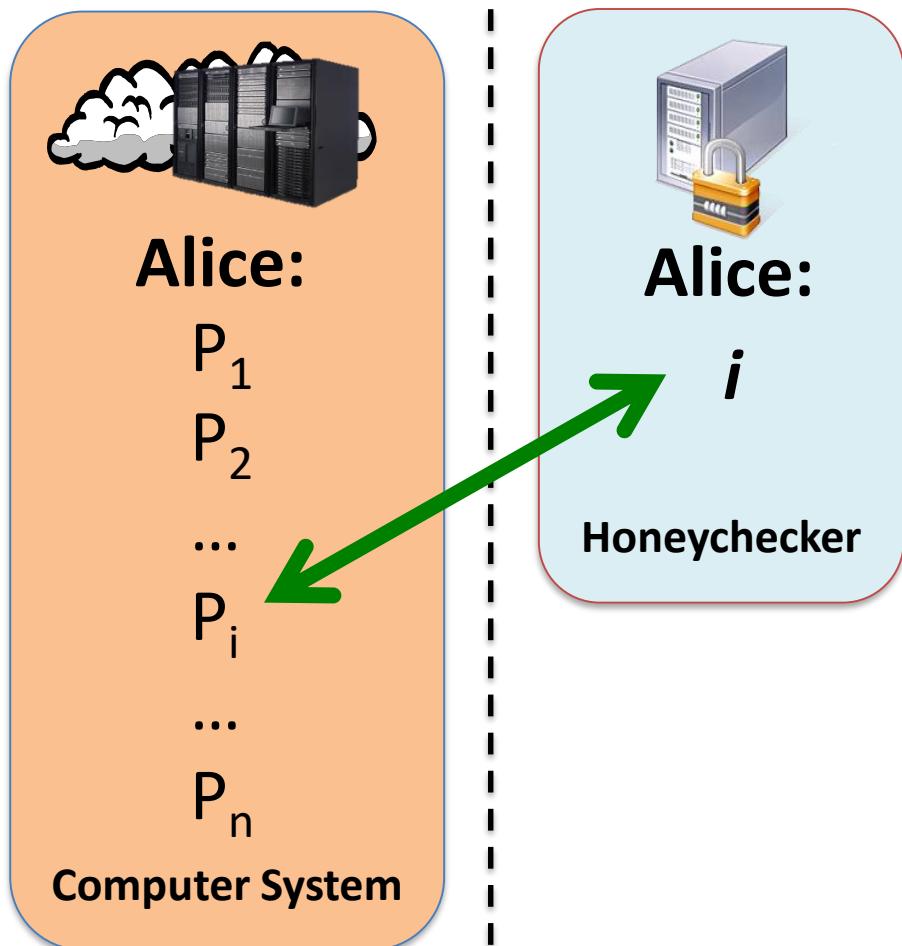
- ACM CCS 2013



Honeywords: Making Password
Cracking Detectable

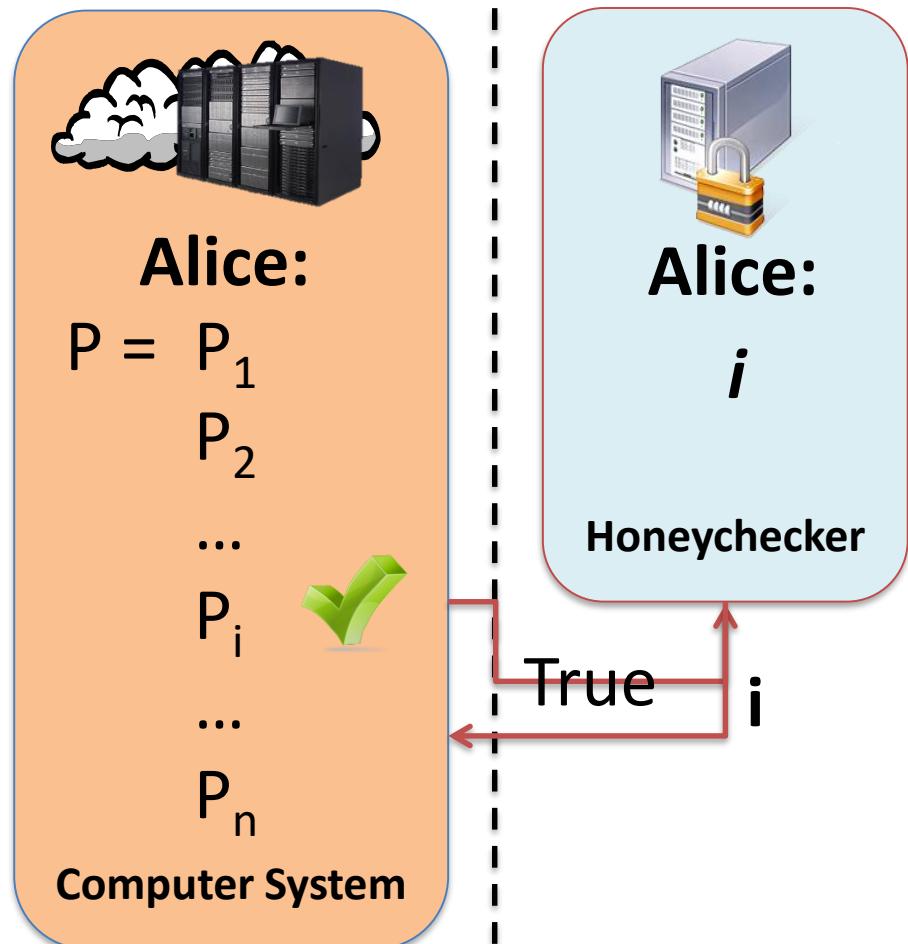
Honeywords: Verification

- The authentication system stores a mapping from Alice to her set of passwords
- A “honeychecker” stores the index of the correct password for Alice

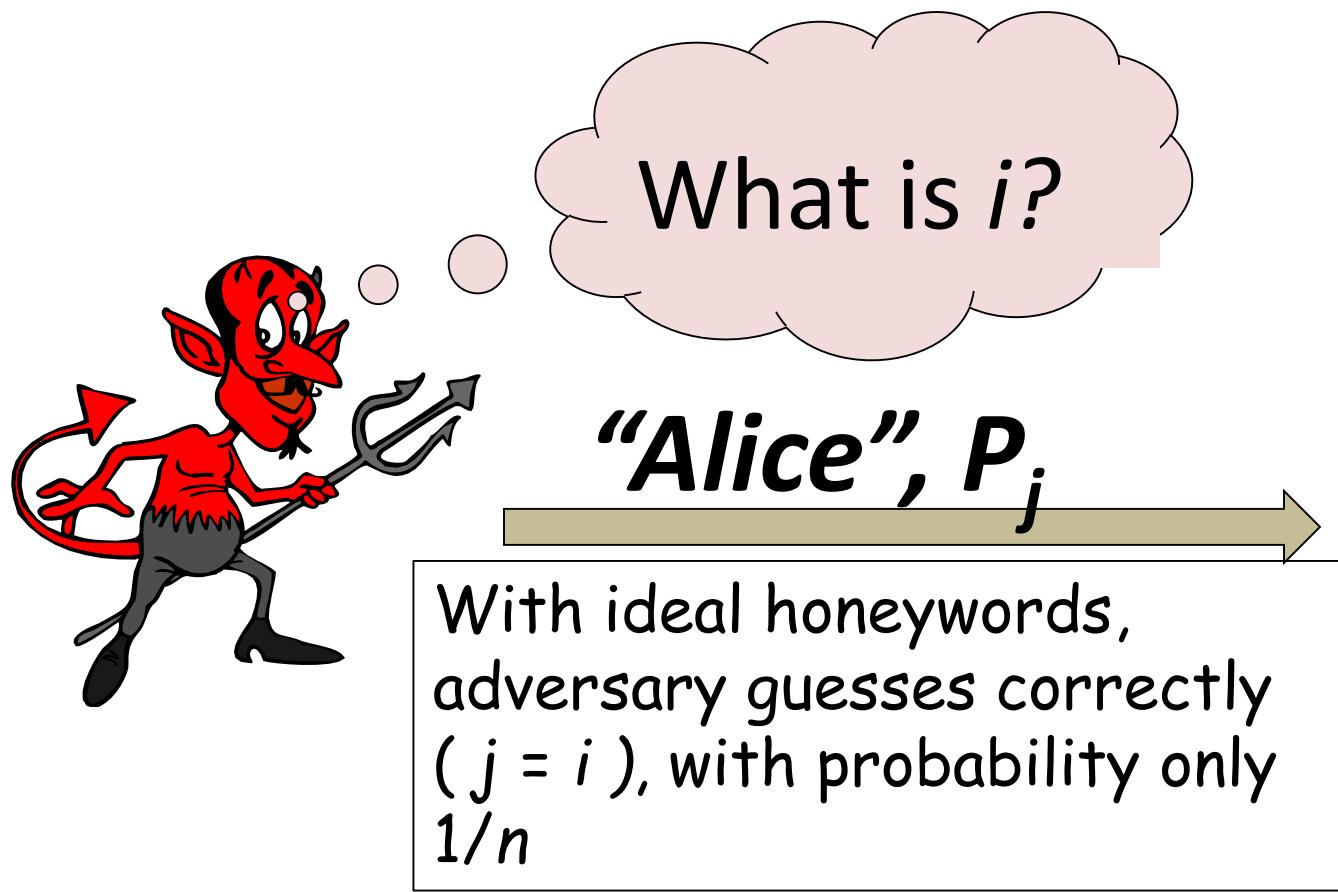


Honeywords: Verification

- Alice authenticates by submitting her password P
- The computer system checks her password against all those it stores
- If a match is found, the index of that match is sent to the honeychecker for verification
- If the index is correct, Alice is authenticated



Honeywords: Verification



Alice:

P_1
 P_2
...

P_i
...

P_n

Computer System

Application 2: Bitcoin



July 26, 2017

1 Bitcoin equals
3100.32 Canadian Dollar

1	Bitcoin
3100.32	Canadian Dollar



Bitcoin



- Bitcoin is an innovative payment network and a new kind of money, invented by Satoshi Nakamoto in 2008 and introduced as open-source software in 2009.
- Bitcoin uses peer-to-peer technology to operate with no central authority or banks; managing transactions and the issuing of bitcoins is carried out collectively by the network.



TheoryCoin vs. Bitcoin

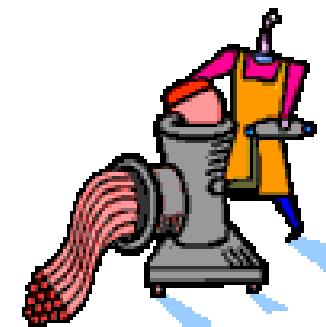
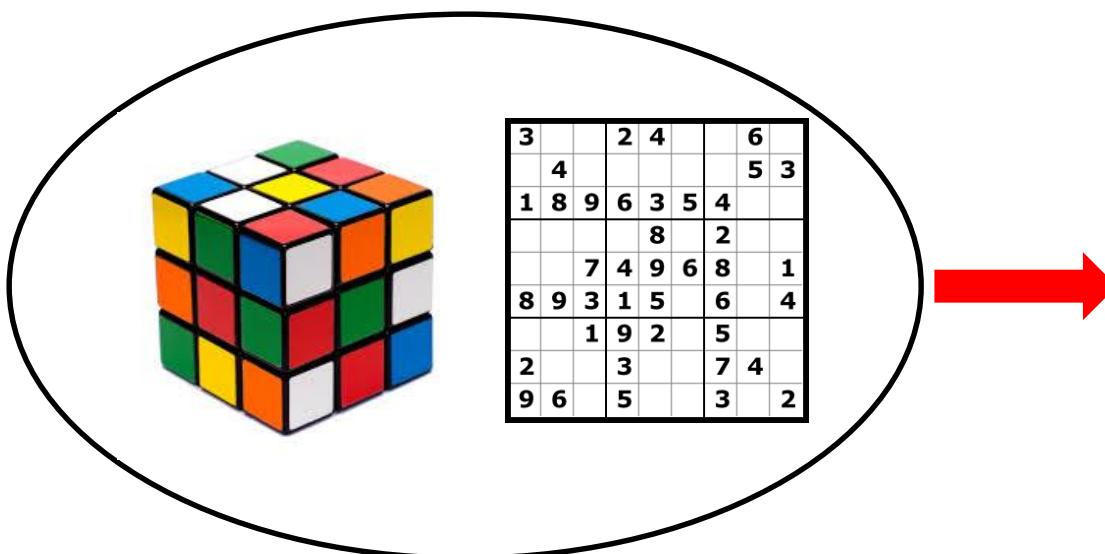


- TheoryCoin (Cryptography)
 - How to *create* coins
 - How to *transfer* coins
 - How to *store* coins
- TheoryCoin → Bitcoin



TheoryCoin: How to create money

1. Everyone **tries to solve** a puzzle
2. The **first one** to solve the puzzle **gets 1 TC**
3. The solution of **puzzle i** defines **puzzle $i+1$**

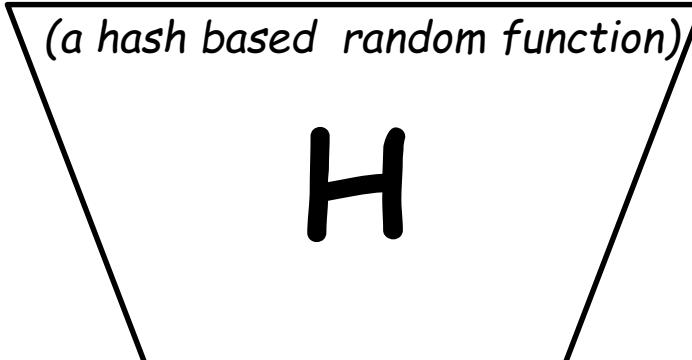


Hash Function

TheoryCoin: How to create money

 $L \in \{0,1\}^*$ $R \in \{0,1\}^*$

(a hash based random function)

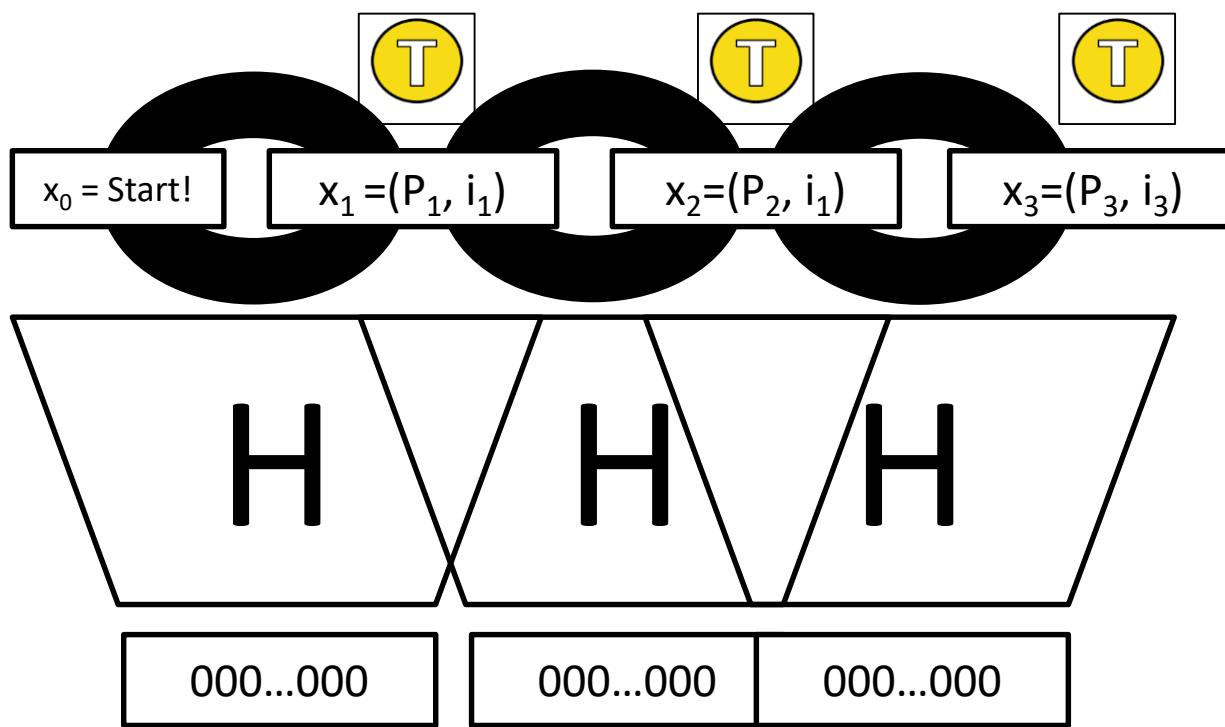
 $T \in \{0,1\}^d$

The puzzle:
given L , find R
such that $T=0^d$

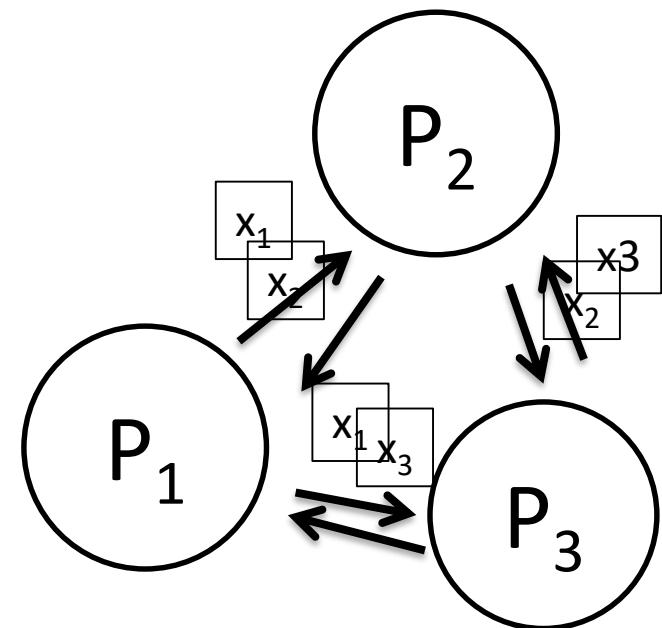
```
SolvePuzzle(L) {  
    repeat {  
        R = my_name || i++  
        T = H(L, R)  
    } while (T != 0d)  
    return R  
}
```

Proof-of-Work

TheoryCoin: How to create money

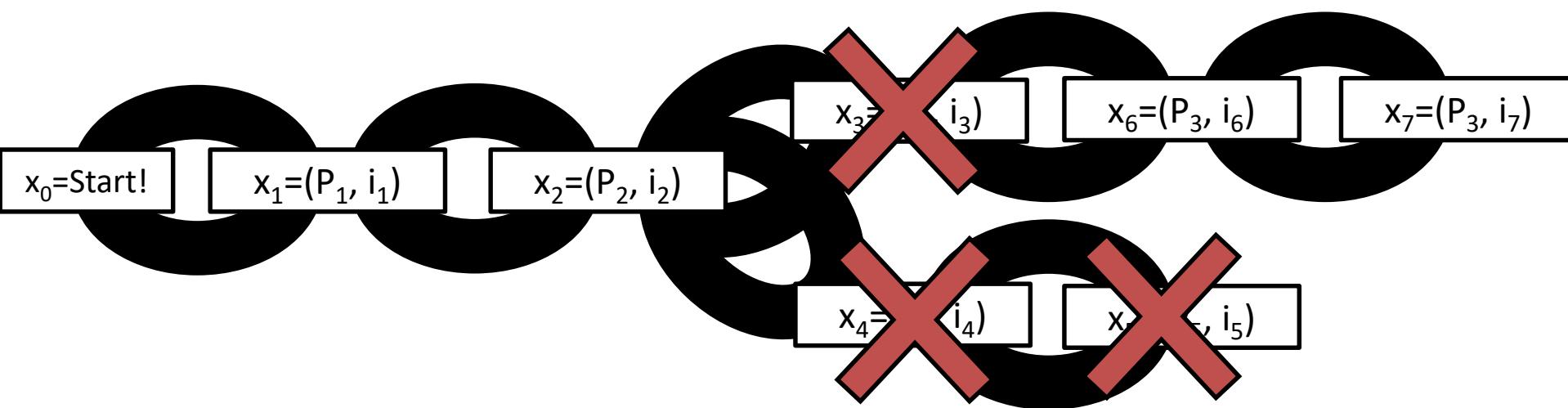


```
SolvePuzzle(L) {  
    repeat{  
        R = my_name || i++  
        T = H(L, R)  
    } while(T ≠ 0d)  
    return R  
}
```



the blockchain

TheoryCoin: How to create money

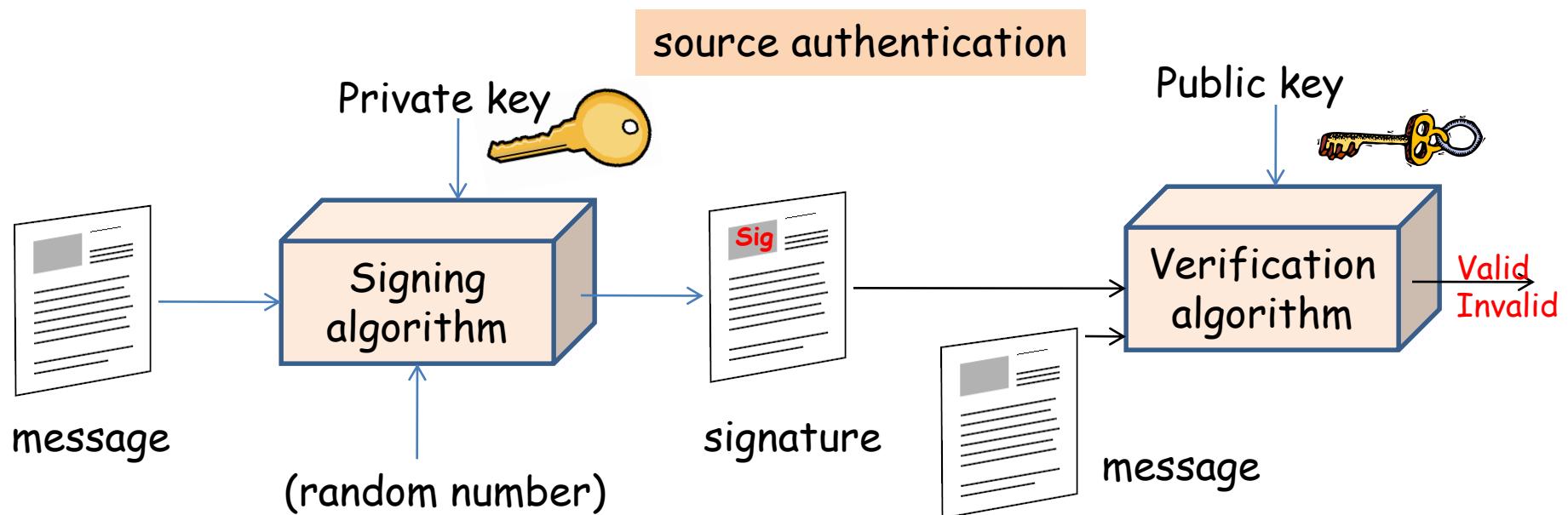


Solve the next puzzle → get a coin
To “solve” puzzle i find x_i s.t $H(x_{i-1}, x_i) = 0^d$
The longest chain defines “next puzzle”
The name in block x_i “gets” coin i .

the 51% attack

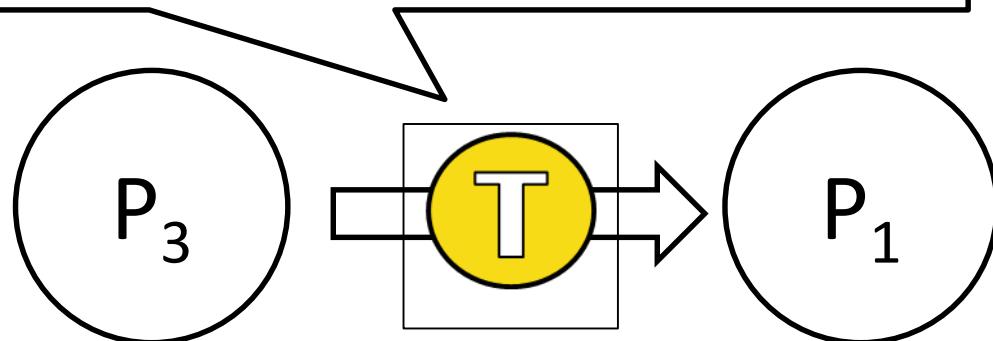
TheoryCoin: How to transfer money

- Use Digital Signature

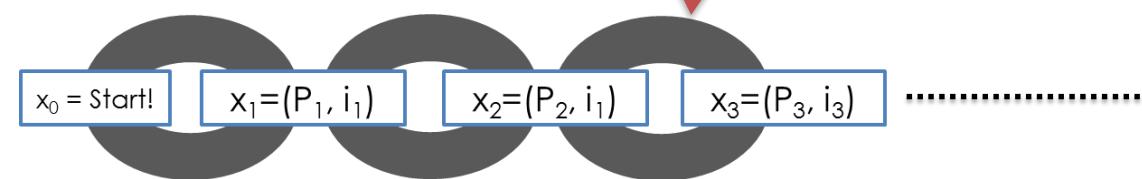


TheoryCoin: How to transfer money

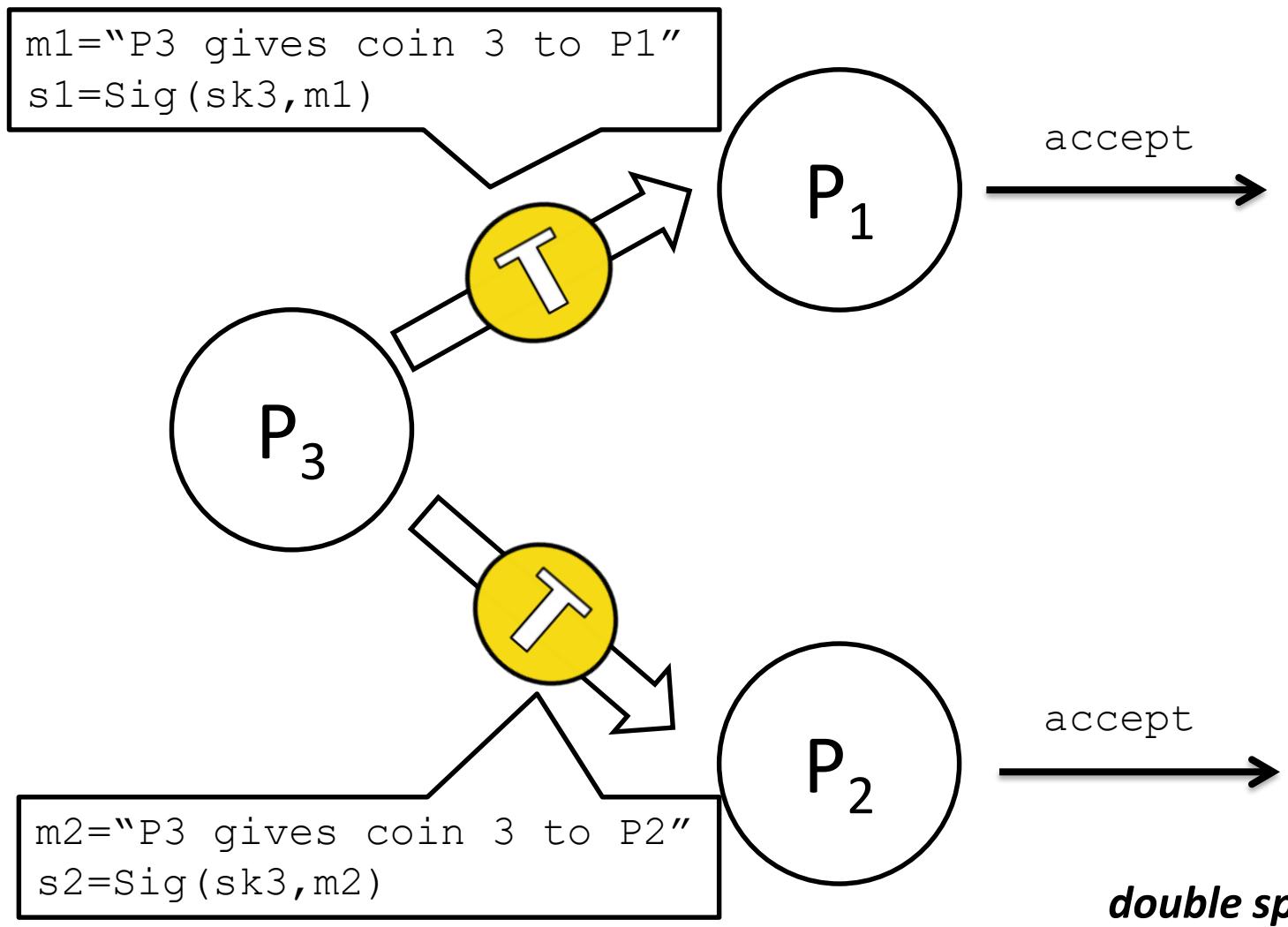
$m = "P_3 \text{ gives coin 3 to } P_1"$
 $s = \text{Sig}(sk_3, m)$



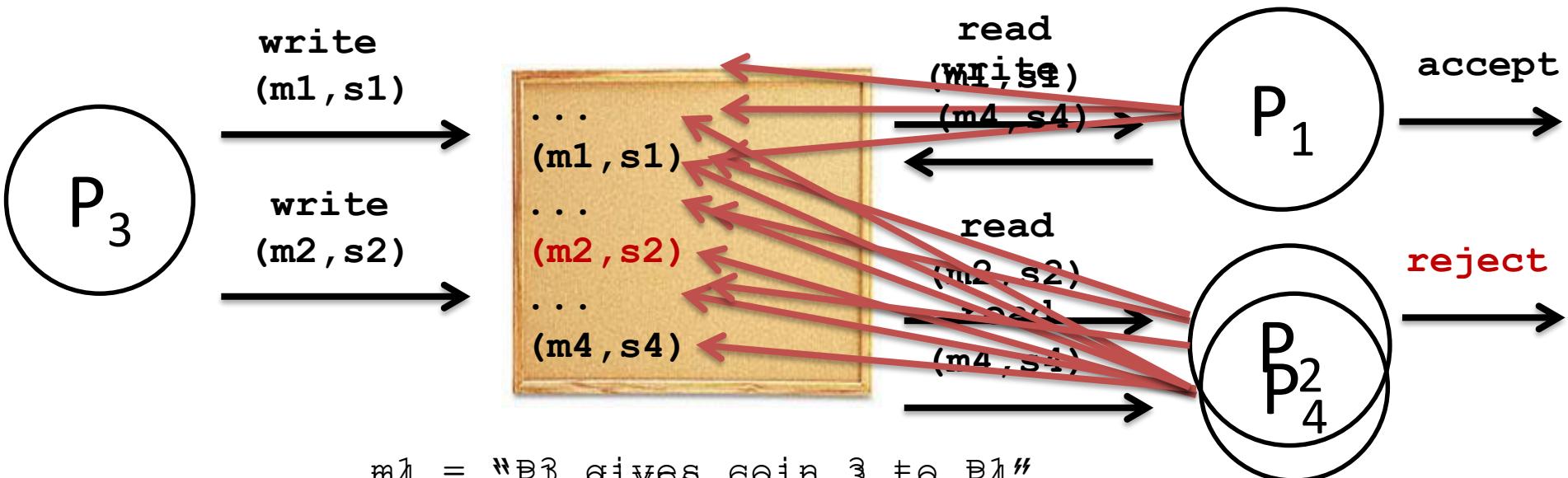
If
 $\text{Ver}(pk_3, m, s) = \text{accept}$
and
 P_3 owns coin 3
then
return accept



TheoryCoin: How to transfer money



TheoryCoin: How to transfer money



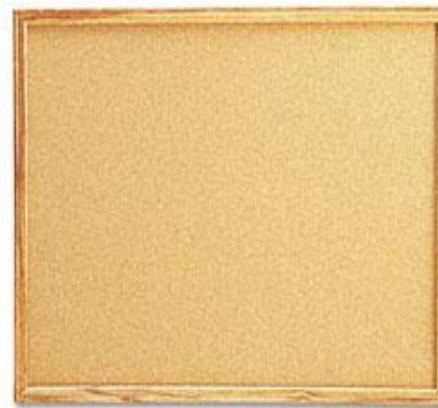
$m_4 \equiv \text{"P}_3 \text{ gives coin 3 to P}_4\text{"}$

$s_4 \equiv \text{Sig}(\text{sk}_3, m_4)$

$m_2 = \text{"P}_3 \text{ gives coin 3 to P}_4\text{"}$

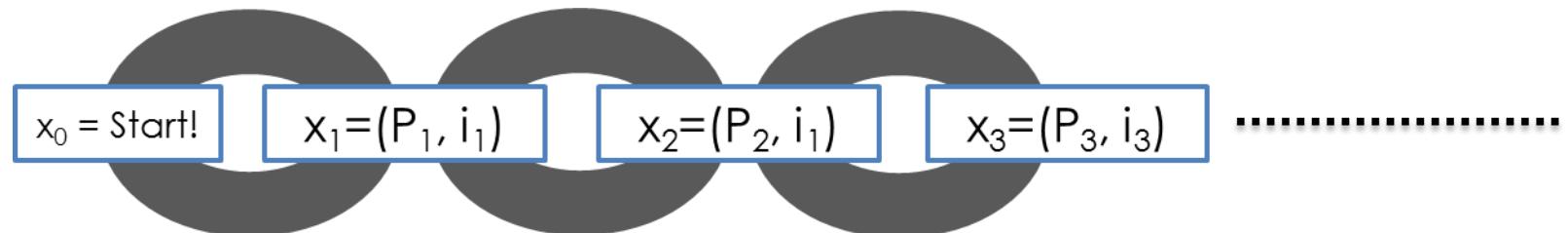
$s_2 = \text{Sig}(\text{sk}_3, m_2)$

TheoryCoin: How to store money



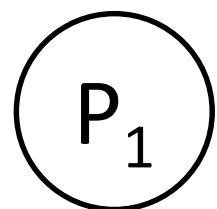
Main Idea:

Record transfers in the blockchain



TheoryCoin: How to store money

```
SolvePuzzle(L) {  
    repeat{  
        R = my_name || i++  
        T = H(L,R)  
    }while(T ≠ 0d)  
    return R  
}
```



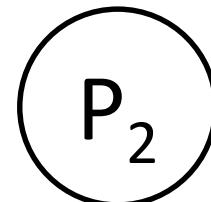
(m, s)

```
SolvePuzzle(L) {  
    repeat{  
        R = my_name || i++  
        T = H(L,R)  
    }while(T ≠ 0d)  
    return R  
}
```

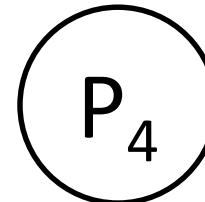
```
SolvePuzzle(L) {  
    repeat{  
        R = my_name  
        T = H(L,R)  
    }while(T ≠ 0d)  
    return R  
}
```



```
SolvePuzzle(L, . . .) {  
  
repeat{  
  
    R = my_name || (m, s) || i++  
    T = H(L, R)  
  
}while(T ≠ 0d)  
  
return R  
}
```



(m, s)



$x_0 = \text{Start!}$

$x_1 = (P_1, i_1)$

$x_2 = (P_2, i_1)$

$x_3 = (P_3, i_3)$

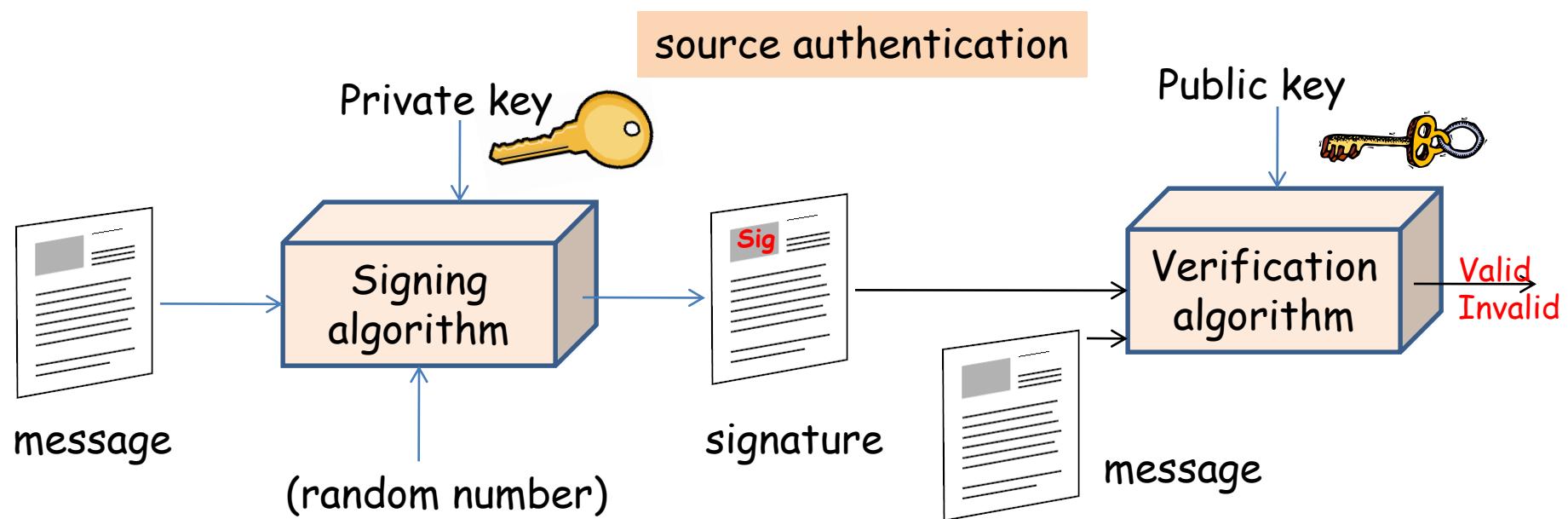
$x_4 = (P_4, (m,s), i_4)$

More Bitcoin Information

- <https://en.bitcoin.it/wiki/Block>



Digital Signature



Digital Signature Properties

- It must verify the author and the date and time of the signature
- It must authenticate the contents at the time of the signature
- It must be verifiable by third parties to resolve disputes

Digital signature
is available
in uERP now



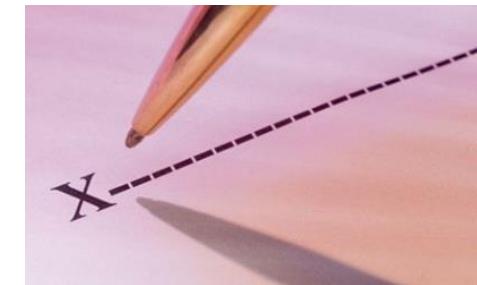
Attack Types on Digital Signatures

- Key-only attack
 - Challenger (C) only knows A 's public key
- Known message attack
 - C is given access to a set of messages and their signatures
- Generic chosen message attack
 - C chooses a list of messages before attempting to break A 's signature scheme, independent of A 's public key; C then obtains from A valid signatures for the chosen messages
- Directed chosen message attack
 - Similar to the generic attack, except that the list of messages to be signed is chosen after C knows A 's public key but before any signatures are seen
- Adaptive chosen message attack
 - C may request from A signatures of messages that depend on previously obtained message-signature pairs



Forgeries on Digital Signatures

- Total break
 - C determines A's private key
- Universal forgery
 - C finds an efficient signing algorithm that provides an equivalent way of constructing signatures on arbitrary messages
- Selective forgery
 - C forges a signature for a particular message chosen by C
- Existential forgery
 - C forges a signature for at least one message; C has no control over the message



Digital Signature Requirements

- The signature must be a bit pattern that depends on the message being signed
- The signature must use some information unique to the sender to prevent both forgery and denial
- It must be relatively easy to produce the digital signature
- It must be relatively easy to recognize and verify the digital signature
- It must be computationally infeasible to forge a digital signature, either by constructing a new message for an existing digital signature or by constructing a fraudulent digital signature for a given message
- It must be practical to retain a copy of the digital signature in storage



RSA Digital Signature



Hash function

$$H(.)$$

$$PK : (n, e)$$

$$SK : d$$



$$m$$

Signing:

$$\sigma = H(m)^d \bmod n$$

$$(m, \sigma)$$

Verification:

$$\sigma^e = ? = H(m) \bmod n$$

$$\sigma^e \bmod n = H(m)^{ed} \bmod n = H(m)^{1+k\varphi(n)} \bmod n$$

$$= H(m) \cdot (H(m)^{\varphi(n)})^k \bmod n = H(m) \bmod n$$

Why is the hash function used?



Hash function

$$H(.)$$

$$PK : (n, e)$$

$$SK : d$$



$$(m, \sigma)$$

$$m$$

Signing:

$$\sigma = H(m)^d \bmod n$$

$$\sigma^e = ? = H(m) \bmod n$$



1. To reduce the size, for a long message $m > n$, we can use hash function to make $H(m) < n$ for RSA signing operation.

Why is the hash function used? (cont.)



Without hash function

$$H(.)$$

$$PK : (n, e)$$

$$SK : d$$



Verification:



$$\sigma_1^e = ? = m_1 \bmod n$$

$$(m_1, \sigma)$$

$$\sigma_1 = m_1^d \bmod n$$

$$\sigma_2^e = ? = m_2 \bmod n$$

$$(m_2, \sigma)$$

$$\sigma_2 = m_2^d \bmod n$$

$$(m_3 = m_1 \cdot m_2, \sigma_3)$$

$$\sigma_1 = m_1^d \bmod n$$

$$\sigma_2 = m_2^d \bmod n$$

$$\sigma_3^e = ? = m_3 \bmod n$$

$$\sigma_3 = \sigma_1 \cdot \sigma_2 = (m_1 \cdot m_2)^d \bmod n$$



2. In order to improve the security

Why is the hash function used? (cont.)



Without hash function

$$H(.)$$

$$\begin{aligned} PK : & (n, e) \\ SK : & d \end{aligned}$$



$$(m_3 = m_1 \cdot m_2, \sigma_3)$$

$$\sigma_1 = m_1^d \bmod n$$

$$\sigma_2 = m_2^d \bmod n$$

$$\sigma_3^e = ? = m_3 \bmod n$$

$$\sigma_3 = \sigma_1 \cdot \sigma_2 = (m_1 \cdot m_2)^d \bmod n$$



- If m_1, m_2 are meaningful, m_3 will be meaningless in most cases, Then the verifier can detect it.
- However, if m_1, m_2 are two session keys (random strings), m_3 is still a random string. In this case, the verifier cannot detect it.
- Because of the one-wayness of the hash function,

Given $H(m_1)^d \bmod n, H(m_2)^d \bmod n$, we can compute $(H(m_1) \cdot H(m_2))^d \bmod n$

However, from $(H(m_1) \cdot H(m_2))^d \bmod n$, we cannot compute $H(m_1 \cdot m_2)^d \bmod n$

Detected!

How to send a source authenticated message with large size over Internet?


$$\begin{aligned}PK_1 : & (n_1, e_1) \\SK_1 : & d_1\end{aligned}$$
 M

$$\sigma_1 = H(M)^{d_1} \bmod n_1$$

choose a short key k

$$C_1 = AES(M \parallel \sigma_1, k)$$

$$C_2 = (k \parallel H(k))^{e_2} \bmod n_2$$

 (C_1, C_2)

$$k \parallel H(k) = C_2^{d_2} \bmod n_2$$

$$M \parallel \sigma_1 \Leftarrow (C_1, k)$$

$$\sigma_1^{e_1} = H(M) \bmod n_1$$

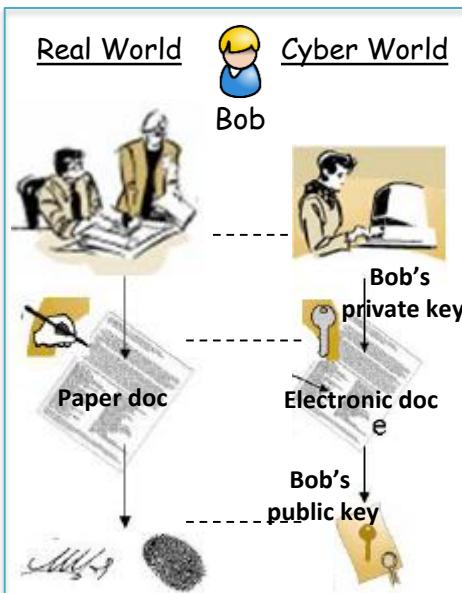
Privacy
(Confidentiality)

Data
Authentication
(Data Integrity)

Non-
Repudiation

User
Authentication

ElGamal Digital Signature

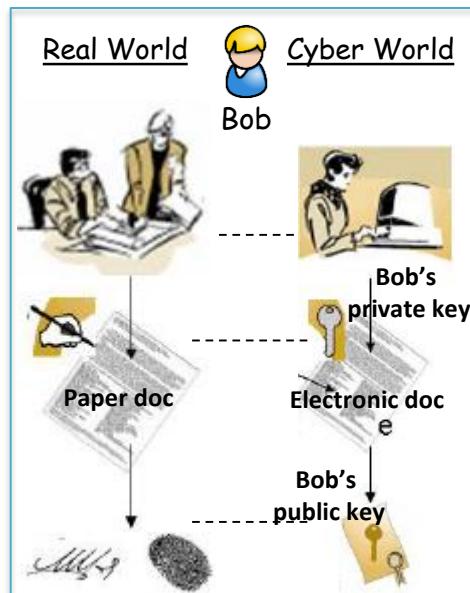


Digital Signature

- + Use private key to sign
- + Use public key to verify
- + Achieve authentication, data integrity, non repudiation

- Description of the original ElGamal Digital Signature
 - Private key: x
 - Public key: $y = g^x \text{ mod } p$
 - Signing: select a random k , compute $r = g^k \text{ mod } p$ and solve the linear equation $m = xr + ks \text{ mod } (p-1)$, finally outputs (r,s) .
 - Verification: check the equation $g^m = y^r r^s \text{ mod } p$.

ElGamal Digital Signature (Cont.)



Digital Signature

- + Use private key to sign
- + Use public key to verify
- + Achieve authentication, data integrity, non repudiation

- The correctness $g^m = y^r r^s \text{ mod } p$.

$$\begin{aligned} & y^r r^s \text{ mod } p \\ &= g^{xr} \cdot g^{ks} \text{ mod } p \\ &= g^{xr+ks} \text{ mod } p \\ &= g^m \text{ mod } p \end{aligned}$$

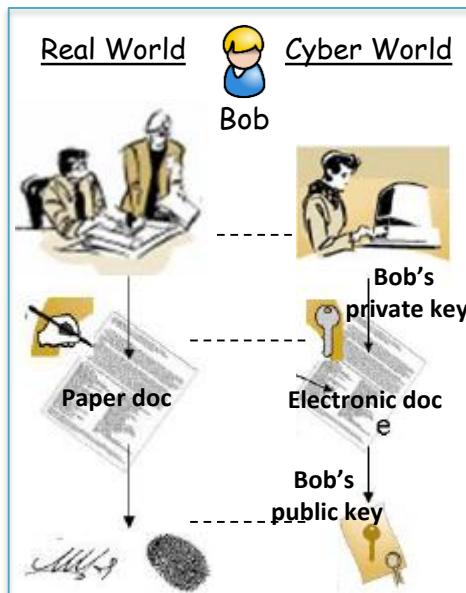
Without knowing the private key x , it is hard to compute s such that $m = xr + ks \text{ mod } (p-1)$. Therefore, The ElGamal digital signature was believed secure! However, the ElGamal digital signature was found insecure later!

Attack 1

– Private key: **X**

– Public key: **$y=g^x \text{ mod } p$**

- Select **e** , Let **$r=g^e y \text{ mod } p$** and **$s=-r \text{ mod } p-1$** , it is easy to see that **(r,s)** is a valid signature for the message **$m=es \text{ mod } p-1$** .



Digital Signature

- + Use private key to sign
- + Use public key to verify
- + Achieve authentication, data integrity, non repudiation

$$\begin{aligned}y^r \cdot r^s &= g^{xr} \cdot g^{es} \cdot y^s \\&= g^{xr} \cdot g^{es} \cdot g^{xs} \\&= g^{es} = g^m \text{ mod } p\end{aligned}$$

If m is meaningful in some applications, **$m=es \text{ mod } p-1$** is obviously meaningless. Then, the receiver can detect it.

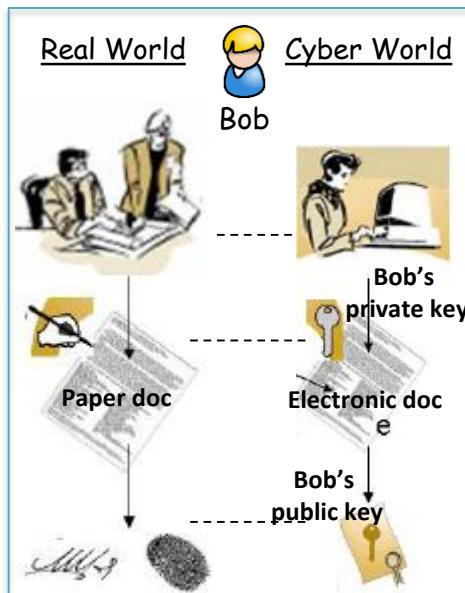
However, if m is a session key (a random string), the receiver cannot identify **$m=es \text{ mod } p-1$** is correct or not.

Attack 2

- Private key: **X**

- Public key: $y=g^x \text{ mod } p$

- Select e, v , let $r=g^{ey^v} \text{ mod } p$ and $s=-rv^{-1} \text{ mod } p-1$, then (r,s) is a signature for the message $m=es \text{ mod } p-1$.



Digital Signature

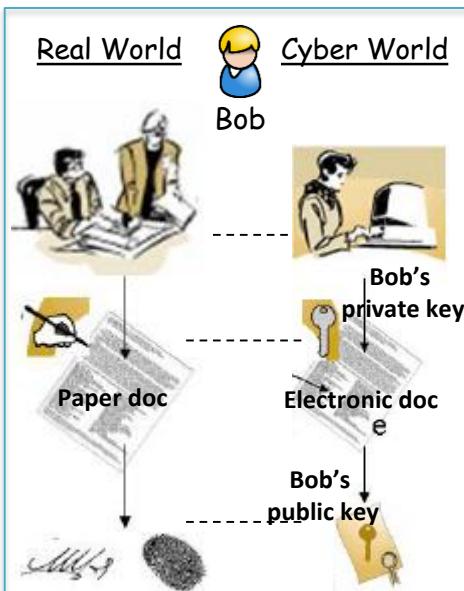
- + Use private key to sign
- + Use public key to verify
- + Achieve authentication, data integrity, non repudiation

$$\begin{aligned}
 y^r \cdot r^s &= y^r \cdot r^{-rv^{-1}} \\
 &= y^r \cdot g^{e \cdot (-rv^{-1})} \cdot y^{v \cdot (-rv^{-1})} \\
 &= y^r \cdot g^{es} \cdot y^{-r} \\
 &= g^{es} = g^m
 \end{aligned}$$

If m is meaningful in some applications, $m=es \text{ mod } p-1$ is obviously meaningless. Then, the receiver can detect it.

However, if m is a session key (a random string), the receiver cannot identify $m=es \text{ mod } p-1$ is correct or not.

Hash function is required for ElGamal Digital Signature



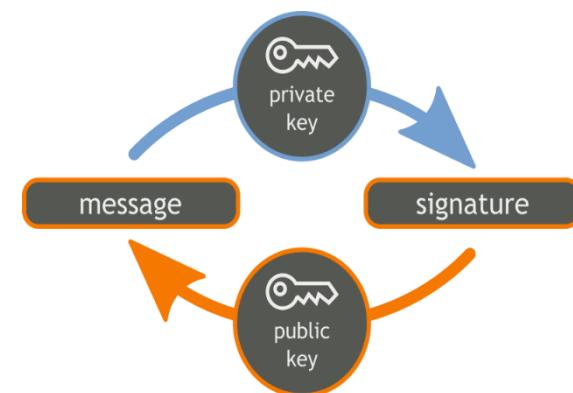
Digital Signature

- + Use private key to sign
- + Use public key to verify
- + Achieve authentication, data integrity, non repudiation

- Description of the original ElGamal Digital Signature
 - Private key: x
 - Public key: $y = g^x \text{ mod } p$
 - Signing: select a random k , compute $r = g^k \text{ mod } p$ and solve the linear equation $H(m) = xr + ks \text{ mod } (p-1)$, finally outputs (r, s) .
 - Verification: check the equation $g^{H(m)} = y^r r^s \text{ mod } p$.

Digital Signature Standard (DSS)

- Digital Signature Standard (DSS) is the digital signature algorithm (DSA) developed by the U.S. National Security Agency (NSA) to generate a digital signature for the authentication of electronic documents.
- DSS was put forth by the National Institute of Standards and Technology (NIST) in 1994, and has become the United States government standard for authentication of electronic documents.



Digital Signature Standard (DSS) (Cont.)

- Key Generation
 - Private key: x
 - Public key: $y = g^x \pmod{p}$
- Description of DSA
 - Signing: to sign a message M
 - generate random k , $k < q$
 - compute
 - $r = (g^k \pmod{p}) \pmod{q}$
 - $s = k^{-1} * (H(M) + x * r) \pmod{q}$
 - the signature is (r, s)
 - Verification: to verify a signature (r, s) :
 - $w = s^{-1} \pmod{q}$
 - $u_1 = (H(M) * w) \pmod{q}$
 - $u_2 = r * w \pmod{q}$
 - $v = (g^{u_1} * y^{u_2} \pmod{p}) \pmod{q}$
 - if $v=r$ then the signature is verified

p of length 2^L is a prime number, where $L= 512$ to 1024 bits and is a multiple of 64
 q is a 160 bit prime factor of $p-1$
 $g = h^{(p-1)/q}$ where h is any number less than $p-1$ with $h^{(p-1)/q} \pmod{p} > 1$
 x is a number less than q (private key) $y = g^x \pmod{p}$

Signature	Length
DSS	320 bits
ElGamal signature	2^L bits

Digital Signature Standard (DSS) (Cont.)

- **Correctness:** DSS is correct in the sense that a verifier will always accept genuine signatures.
- First, if $g = h^{(p-1)/q} \pmod{p}$, it follows $g^q = h^{(p-1)} = 1 \pmod{p}$. That is, g has the order q .
- The signer computes $s = k^{-1} * (H(M) + x * r) \pmod{q}$, thus
$$k = H(M)^* s^{-1} + x^* r^* s^{-1} = H(M)^* w + x^* r^* w \pmod{q}$$
- Since g has the order q , we have
$$g^k = g^{H(m)^* w} g^{x^* r^* w} = g^{H(m)^* w} y^{r^* w} = g^{u_1} y^{u_2} \pmod{p}$$
- In the end, the correctness of DSA follows from
$$r = (g^k \pmod{p}) \pmod{q} = (g^{u_1} y^{u_2} \pmod{p}) \pmod{q} = v$$

Example

Alice chooses $q = 101$ and $p = 8081$. Alice selects $h = 3$ and calculates $g = h^{(p-1)/q} \bmod p = 6968$. Alice chooses $x = 61$ as the private key and calculates $y = g^x \bmod p = 2038$. Now Alice can send a message to Bob. Assume that $h(M) = 5000$ and Alice chooses a random number $k = 61$:

$$h(M) = 5000, k = 61$$

$$r = (g^k \bmod p) \bmod q = 54$$

$$s = k^{-1}(h(M) + x \cdot r) \bmod q = 40$$

Alice sends (M, r, s) to Bob. Bob uses the public keys to calculate V .

$$w = s^{-1} = 48 \bmod 101$$

$$V = (g^{u_1} * y^{u_2} \bmod p) \bmod q$$

$$= [(6968^{5000 \times 48} \times 2038^{54 \times 48}) \bmod 8081] \bmod 101 = 54$$

EC ElGamal Digital Signature



$$y^2 = x^3 + ax + b \quad a, b \in F_p \text{ and } 4a^3 + 27b^2 \neq 0 \pmod p$$

$$SK: d_A \in [1, n-1]$$

$$PK: Q_A = d_A P$$



Sign m :

- $k \in [1, n-1]$

- $R = kP = (x_1, y_1), r = x_1 \pmod n$, if $r = 0$, then goto step 1)

- $e = h(m)$, where $h: \{0,1\}^* \rightarrow F_n$

- $s = \frac{e + d_A \cdot r}{k} \pmod n$, if $s = 0$ then goto step 1.

$\sigma = (R, s)$ is A's signature of message m .

Verify:

- 1) verify that $s \in [1, n-1]$ and $R = (x_1, y_1) \in E(F_q)$

- 2) compute $V_1 = sR$;
- 3) $V_2 = h(m)P + rQ_A$, where $r = x_1$

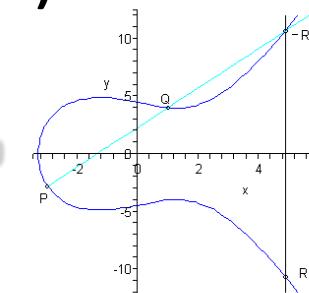
- 4) accept if and only if $V_1 = V_2$

EC Digital Signature Algorithm (ECDSA)

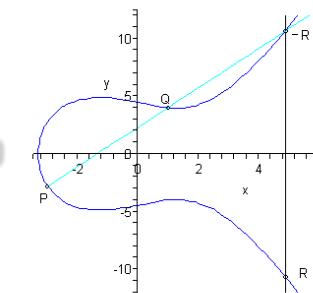


ECDSA key pair generation:

1. Entity A selects an elliptic curve E defined over F_p . The number of points in $E(F_p)$ should be divisible by a large prime n .
2. Select a point P in $E(F_p)$ of order n .
3. Select a statistically unique and unpredictable integer d in the interval $[1, n-1]$.
4. Compute $Q = dP$.
5. A's public key is (E, P, n, Q) . A's private key is d .



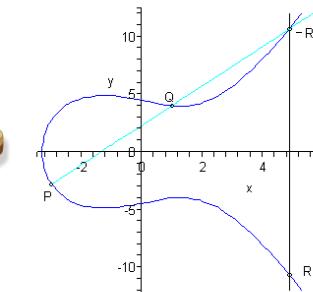
EC Digital Signature Algorithm (ECDSA) cont.



ECDSA signature generation:

1. Entity A selects a statistically unique and unpredictable integer k in the interval $[1, n-1]$.
2. Compute $kP = (x_1, y_1)$ and $r = x_1 \bmod n$. To avoid a security condition, r should not equal 0. If $r = 0$ go to step 1.
3. Compute $k^{-1} \bmod n$.
4. Compute $s = k^{-1} \{h(m) + dr\} \bmod n$. h is the Secure Hash Algorithm (SHA-1).
5. If $s = 0$, then go to Step 1. If s^{-1} mod n does not exist and s^{-1} is required in the signature verification process.
6. The signature for the message m is the pair of integers (r, s) .

EC Digital Signature Algorithm (ECDSA) cont.



ECDSA signature verification:

1. Entity B obtains an authentic copy of Entity A's public key (E, P, n, Q).
2. Verify that r and s are integers in the interval $[1, n-1]$.
3. Compute $w = s^{-1} \bmod n$ and $h(m)$.
4. Compute $u_1 = h(m)w \bmod n$ and $u_2 = rw \bmod n$.
5. Compute $u_1P + u_2Q = (x_0, y_0)$ and $v = x_0 \bmod n$.
6. Entity B accepts the signature if and only if $v = r$.

Thank
you



CS 6355/4355: Cryptanalysis and Database Security

Topic 7: Key Management and Distribution

Lecturer: Rongxing LU

Email: RLU1@unb.ca Office: GE 114

Website: <http://www.cs.unb.ca/~rlu1/>

Faculty of Computer Science, University of New Brunswick

Key Distribution Technique

- Term that refers to the means of delivering a key to two parties who wish to exchange data without allowing others to see the key
- For symmetric encryption to work, the two parties to an exchange must share the same key, and that key must be protected from access by others
- Frequent key changes are desirable to limit the amount of data compromised if an attacker learns the key



Key Distribution Techniques

- Symmetric key distribution using symmetric encryption
- Symmetric key distribution using public-key encryption
- Distribution of public keys
- Announcement, directory, authority, CA
- X.509 authentication and certificates
- Public key infrastructure (PKIX)

Symmetric key distribution using symmetric encryption

- symmetric schemes require both parties to share a common secret key
- issue is how to securely distribute this key
- whilst protecting it from others
- frequent key changes can be desirable
- often secure system failure due to a break in the key distribution scheme

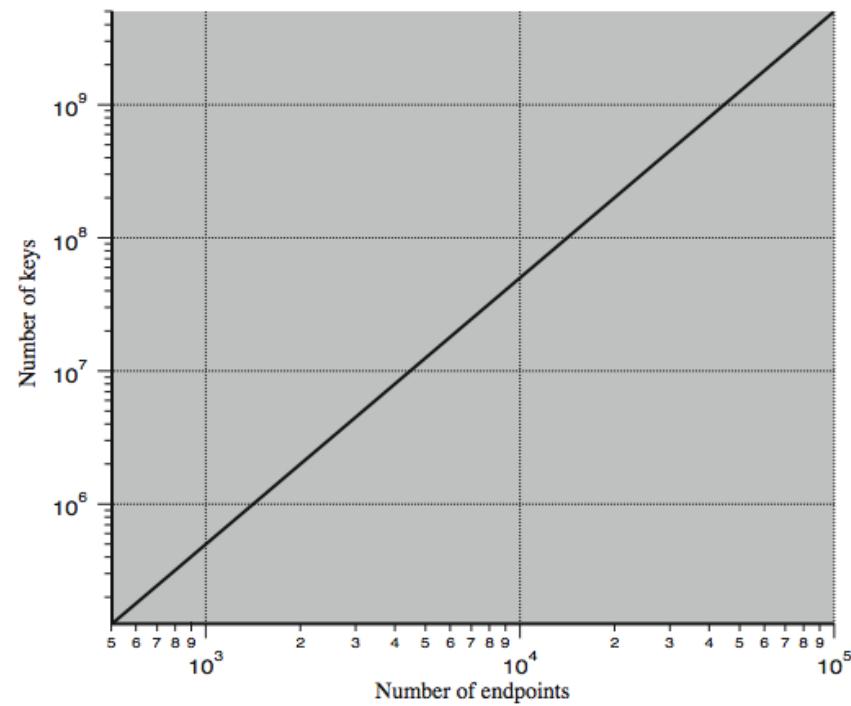
Symmetric Key Distribution



- Given parties A and B, key distribution can be achieved in a number of ways:
 - A can select a key and physically deliver it to B
 - A third party can select the key and physically deliver it to A and B
 - If A and B have previously and recently used a key, one party can transmit the new key to the other, encrypted using the old key
 - If A and B each has an encrypted connection to a third party C, C can deliver a key on the encrypted links to A and B

Key Distribution Is A Big Task

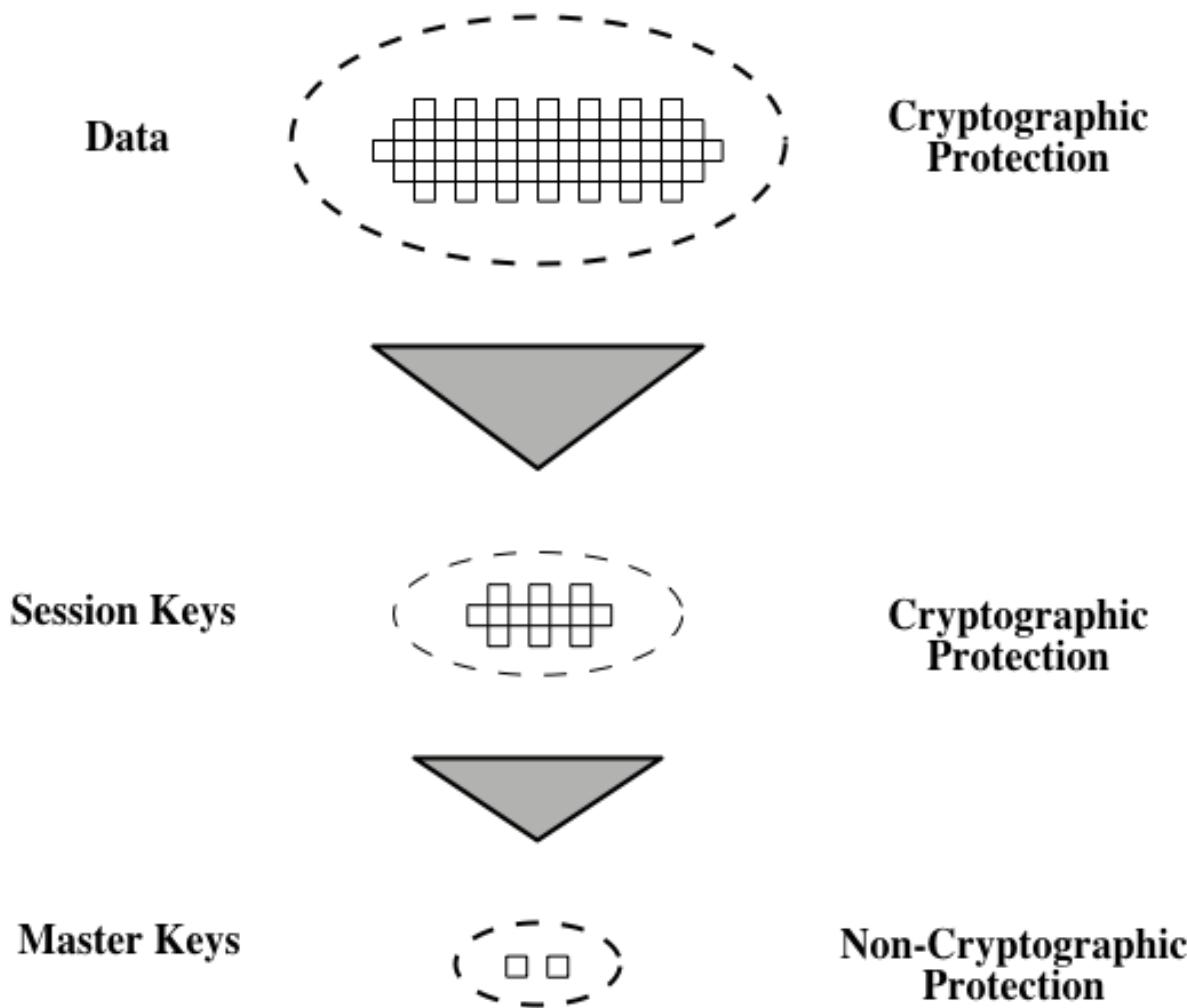
- The scale of the problem depends on the number of communicating pairs that must be supported. If end-to-end encryption is done at a network or IP level, then a key is needed for each pair of hosts on the network that wish to communicate. Thus, if there are N hosts, the number of required keys is $\frac{N(N-1)}{2} \approx \frac{N^2}{2}$



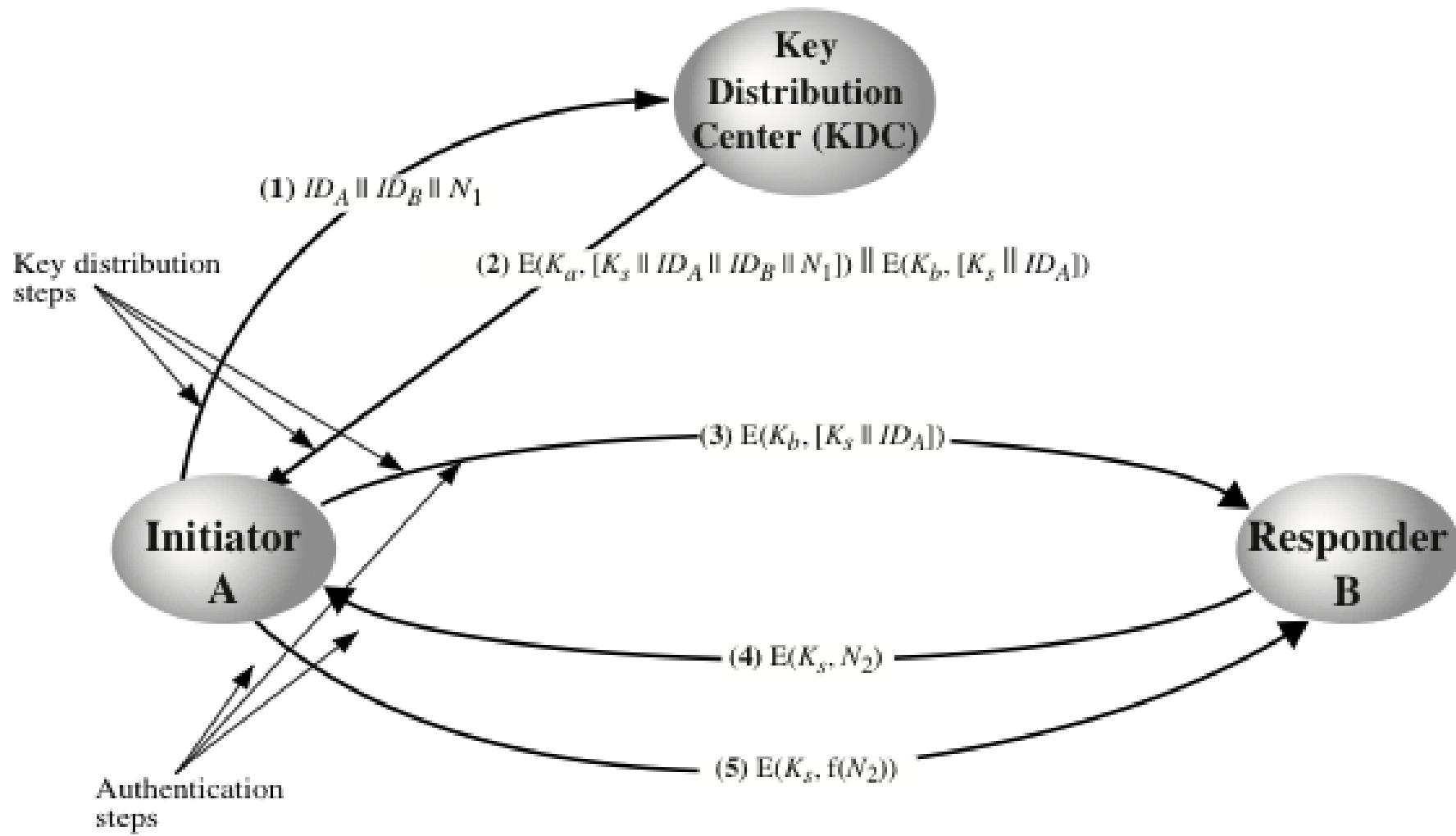
Key Hierarchy

- typically have a hierarchy of keys
- session key
 - temporary key
 - used for encryption of data between users
 - for one logical session then discarded
- master key
 - used to encrypt session keys
 - shared by user & key distribution center

Key Hierarchy (2)



Key Distribution Scenario



Key Distribution Scenario (2)

- In a typical key distribution scenario, a “Key Distribution Center” (**KDC**) which shares a unique key with each party (user).
 1. A requests from the KDC a session key to protect a logical connection to B. The message includes the identity of A and B and a unique nonce N_1 .
 2. The KDC responds with a message encrypted using K_A that includes a one-time session key K_s to be used for the session, the original request message to enable A to match response with appropriate request, and info for B

Key Distribution Scenario (3)

3. A stores the session key for use in the upcoming session and forwards to B the information from the KDC for B, namely, $E(K_b, [K_s \parallel ID_A])$. Because this information is encrypted with K_b , it is protected from eavesdropping.

At this point, a session key has been securely delivered to A and B, and they may begin their protected exchange. Two additional steps are desirable:

4. Using the new session key for encryption B sends a nonce N_2 to A.
5. Also using K_s , A responds with $f(N_2)$, where f is a function that performs some transformation on N_2 (eg. adding one). These steps assure B that the original message it received (step 3) was not a replay. Note that the actual key distribution involves only steps 1 through 3 but that steps 4 and 5, as well as 3, perform an authentication function.

Hierarchical Key Control

- For communication among entities within the same local domain, the local KDC is responsible for key distribution
 - If two entities in different domains desire a shared key, then the corresponding local KDC's can communicate through a global KDC
- The hierarchical concept can be extended to three or more layers
- Scheme minimizes the effort involved in master key distribution because most master keys are those shared by a local KDC with its local entities
 - Limits the range of a faulty or subverted KDC to its local area only

Session Key Lifetime

- For connection-oriented protocols one choice is to use the same session key for the length of time that the connection is open, using a new session key for each new session
- For a connectionless protocol there is no explicit connection initiation or termination, thus it is not obvious how often one needs to change the session key

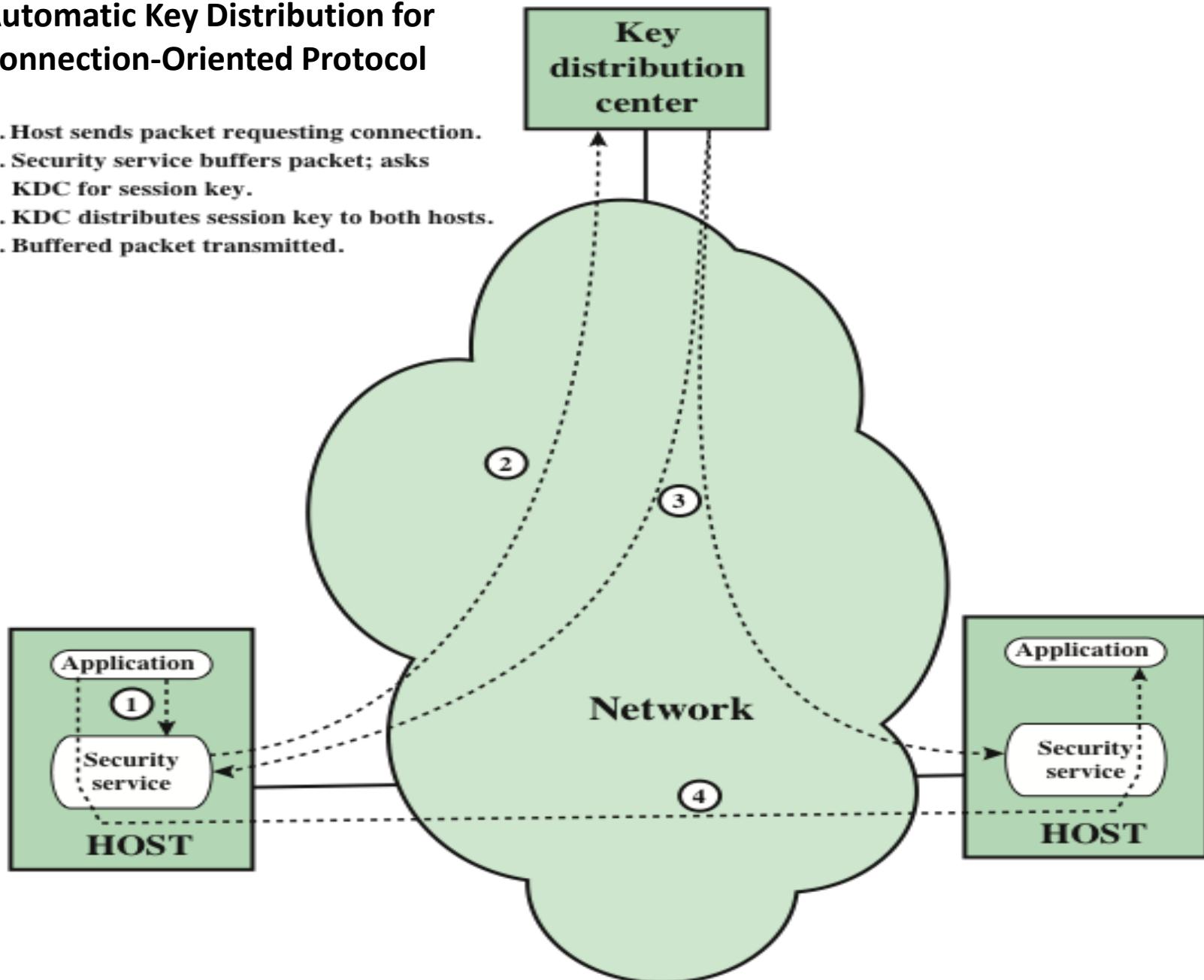
A security manager must balance competing considerations:

The more frequently session keys are exchanged, the more secure they are

The distribution of session keys delays the start of any exchange and places a burden on network capacity

Automatic Key Distribution for Connection-Oriented Protocol

1. Host sends packet requesting connection.
2. Security service buffers packet; asks KDC for session key.
3. KDC distributes session key to both hosts.
4. Buffered packet transmitted.



Controlling Key Usage

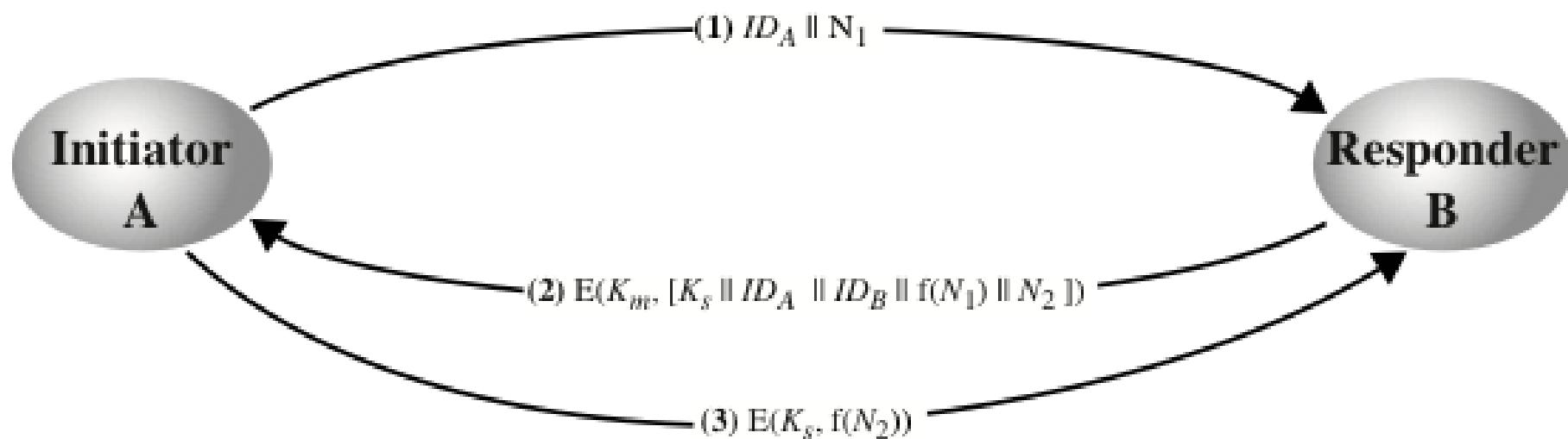
- The concept of a key hierarchy and the use of automated key distribution techniques greatly reduce the number of keys that must be manually managed and distributed
- It also may be desirable to **impose some control** on the way in which automatically distributed keys are used
 - For example, in addition to separating master keys from session keys, we may wish to define different types of session keys on the basis of use



Decentralized Key Distribution

- The use of a key distribution center imposes the requirement that the KDC be trusted and be protected from subversion. This requirement can be avoided if key distribution is fully decentralized. Although full decentralization is not practical for larger networks using symmetric encryption only, it may be useful within a local context.

Why ? $\frac{N(N-1)}{2} \approx \frac{N^2}{2}$

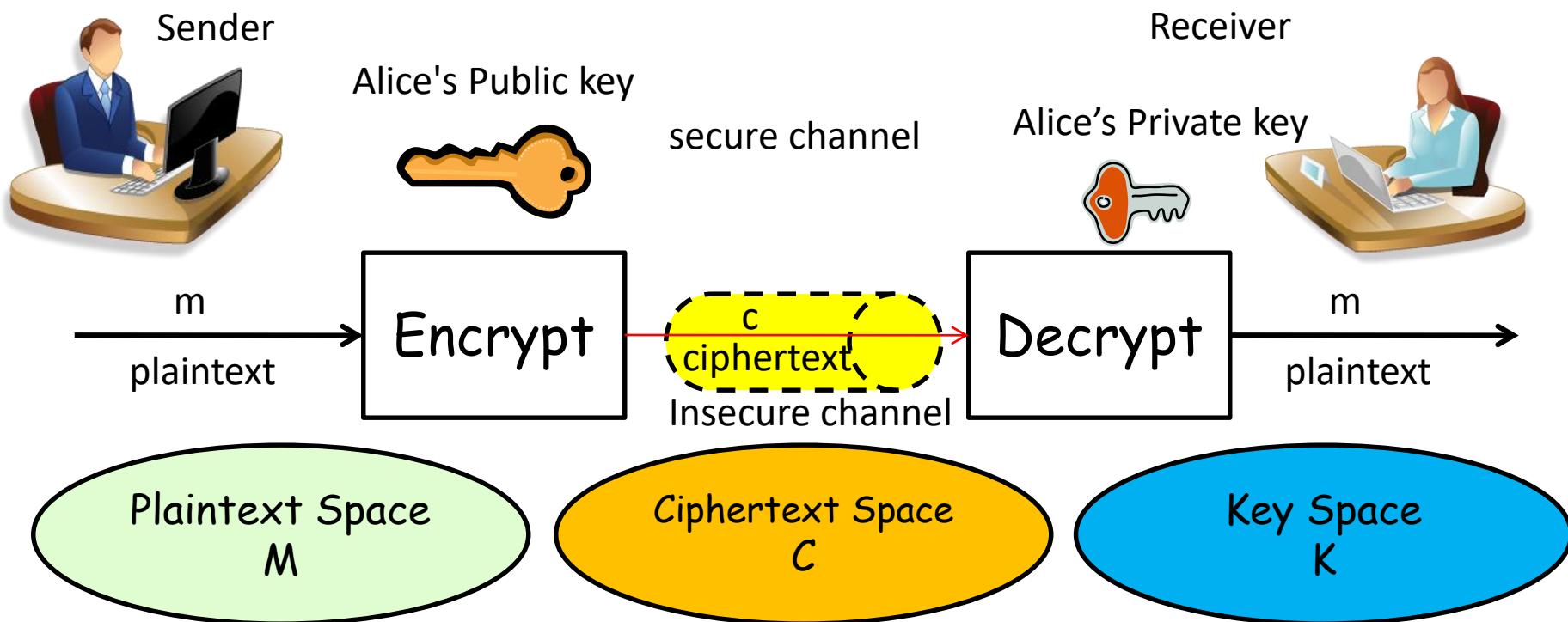


A Summary: Key Distribution Issues

- hierarchies of KDC's required for large networks, but must trust each other
- session key lifetimes should be limited for greater security
- use of automatic key distribution on behalf of users, but must trust system
- controlling key usage
- use of decentralized key distribution, useful within a local context.

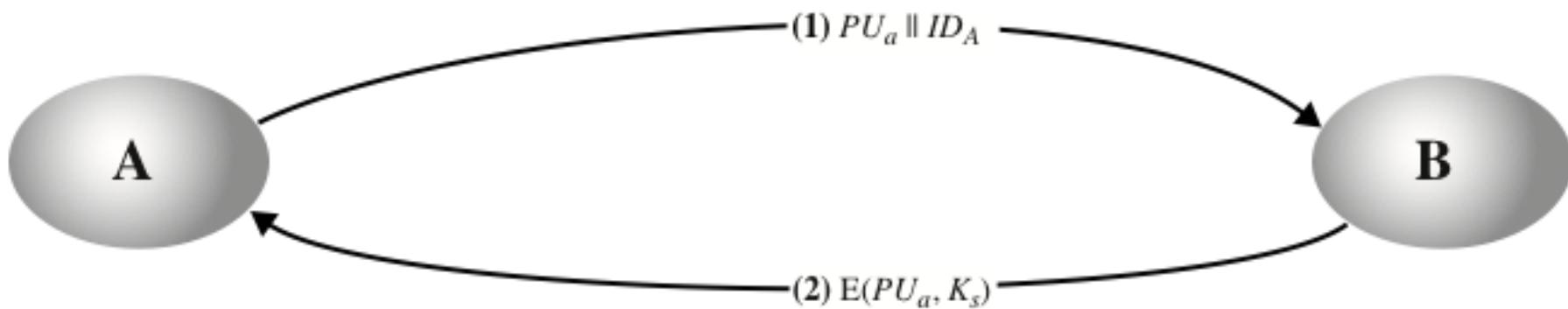
Symmetric key distribution using public-key encryption

- public key cryptosystems are inefficient
 - so almost never use for direct data encryption
 - rather use to encrypt secret keys for distribution



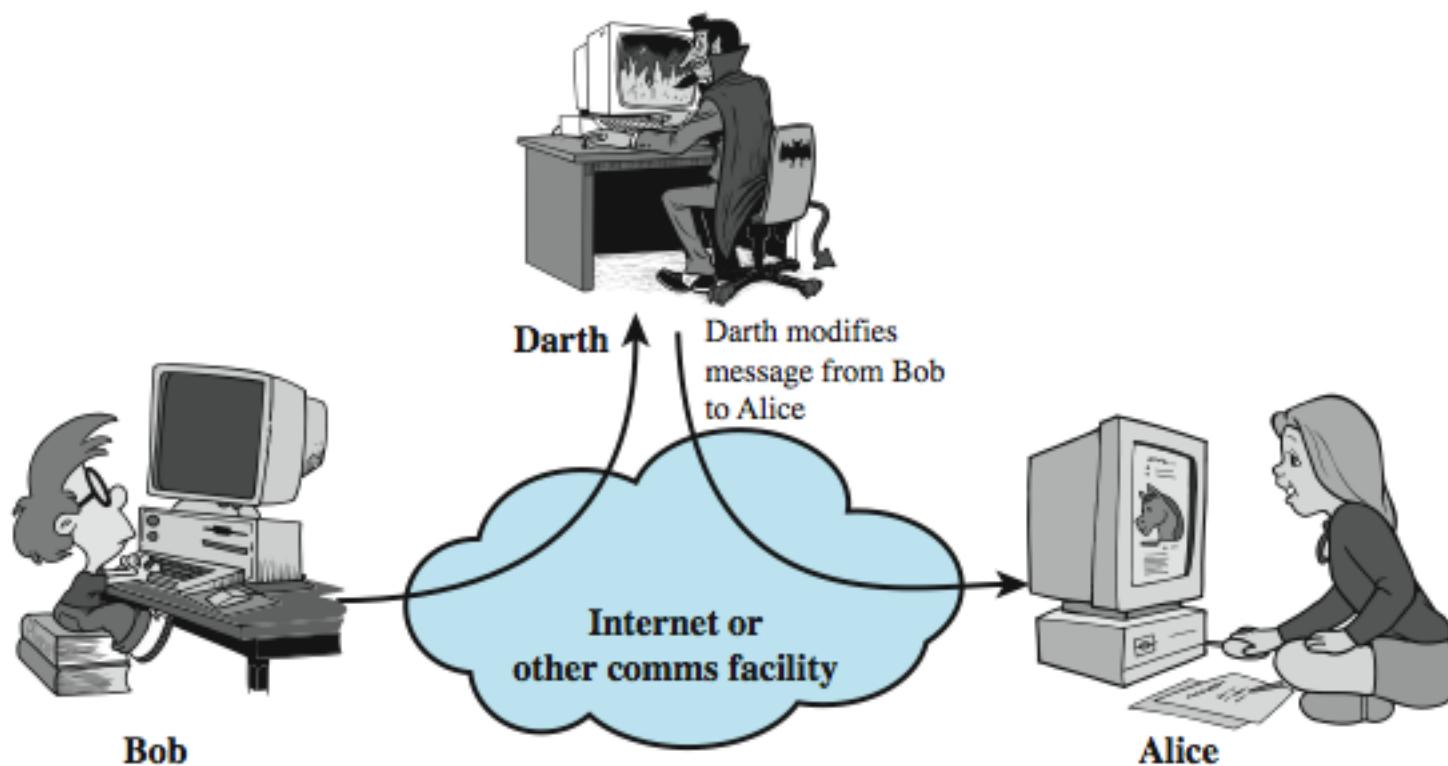
Simple Secret Key Distribution

- An extremely simple scheme was put forward by Merkle in 1979
- Despite its simplicity, this is an attractive protocol.
 - No keys exist before the start of the communication and none exist after the completion of communication. Thus, the risk of compromise of the keys is minimal. At the same time, the communication is secure from eavesdropping.

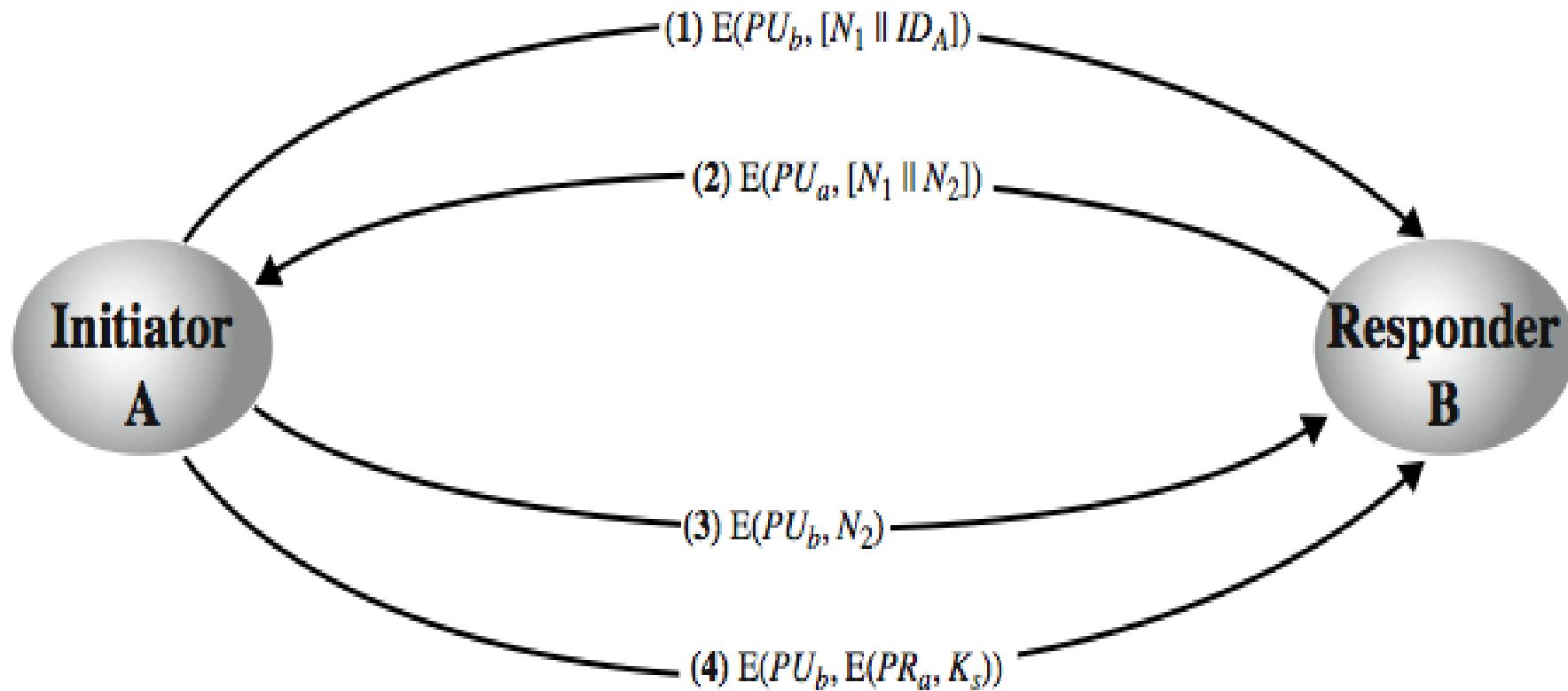


Man-in-the-Middle Attack

- this very simple scheme is vulnerable to an active man-in-the-middle attack?



Secret Key Distribution with Confidentiality and Authentication



Secret Key Distribution with Confidentiality and Authentication (2)

1. A uses B's public key to encrypt a message to B containing an identifier of A (IA) and a nonce (N1), which is used to identify this transaction uniquely.
2. B sends a message to A encrypted with PUa and containing A's nonce (N1) as well as a new nonce generated by B (N2). Because only B could have decrypted message (1), the presence of N1 in message (2) assures A that the correspondent is B.
3. A returns N2, encrypted using B's public key, to assure B that its correspondent is A.
4. A selects a secret key Ks and sends $M = E(PUb, E(PRa, Ks))$ to B. Encryption with B's public key ensures that only B can read it; encryption with A's private key ensures that only A could have sent it.
5. B computes $D(PUa, D(PUb, M))$ to recover the secret key.

The result is that this scheme ensures both confidentiality and authentication in the exchange of a secret key.

Hybrid Key Distribution

- retain use of private-key KDC
- shares secret master key with each user
- distributes session key using master key
- **public-key used to distribute master keys**
 - especially useful with widely distributed users
- rationale
 - performance
 - backward compatibility

Distribution of Public Keys

- can be considered as using one of:
 - ✓ public announcement
 - ✓ publicly available directory
 - ✓ public-key authority
 - ✓ public-key certificates

Public Announcement

- users distribute public keys to recipients or broadcast to community at large
 - eg. append PGP keys to email messages or post to news groups or email list
- major weakness is forgery
 - anyone can create a key claiming to be someone else and broadcast it
 - until forgery is discovered can masquerade as claimed user

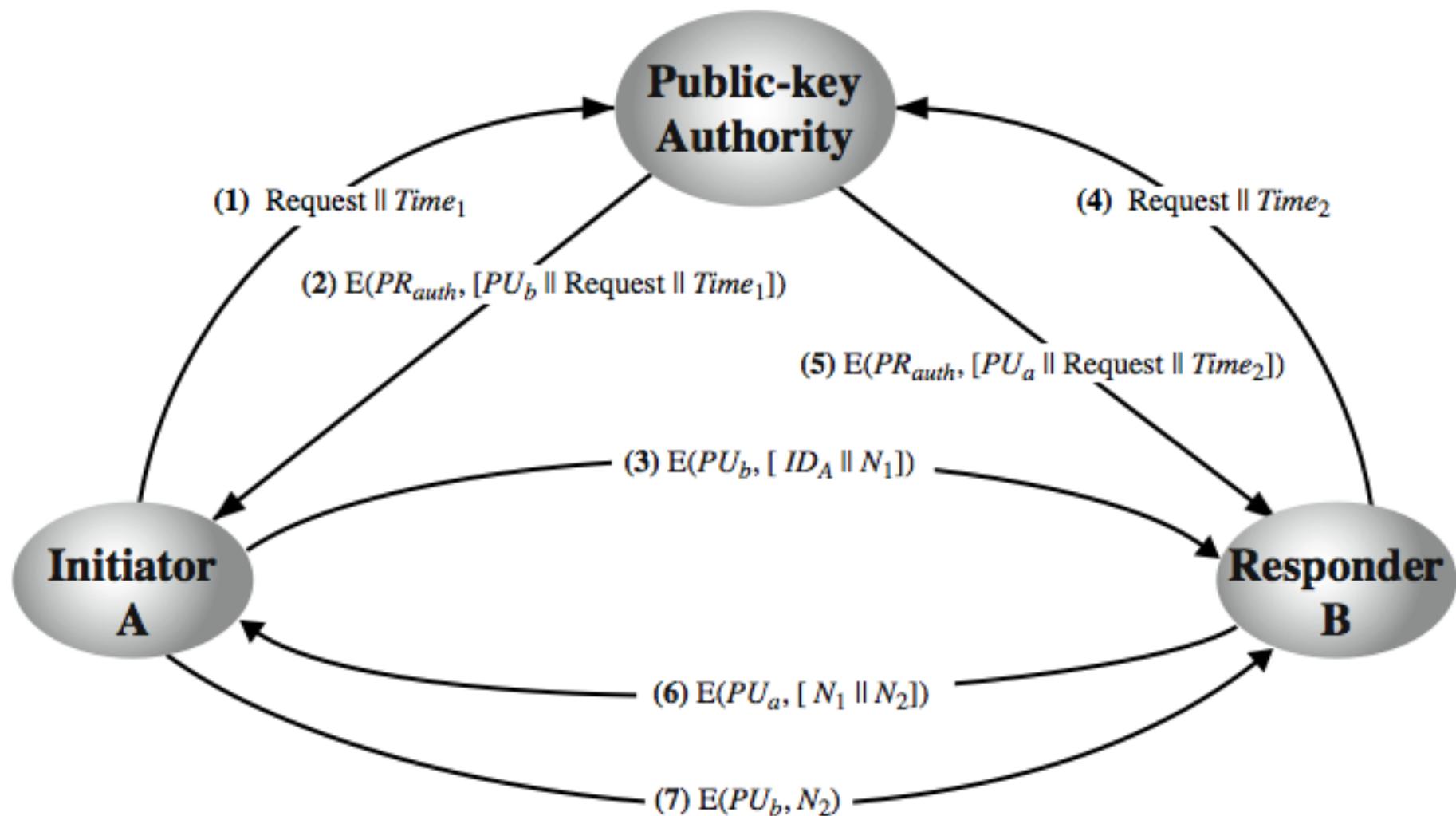
Publicly Available Directory

- can obtain greater security by registering keys with a public directory
- directory must be trusted with properties:
 - contains {name,public-key} entries
 - participants register securely with directory
 - participants can replace key at any time
 - directory is periodically published
 - directory can be accessed electronically
- still vulnerable to tampering or forgery

Public-Key Authority

- improve security by tightening control over distribution of keys from directory
- has properties of directory
- and requires users to know public key for the directory
- then users interact with directory to obtain any desired public key securely
 - does require real-time access to directory when keys are needed
 - may be vulnerable to tampering

Public-Key Authority (2)



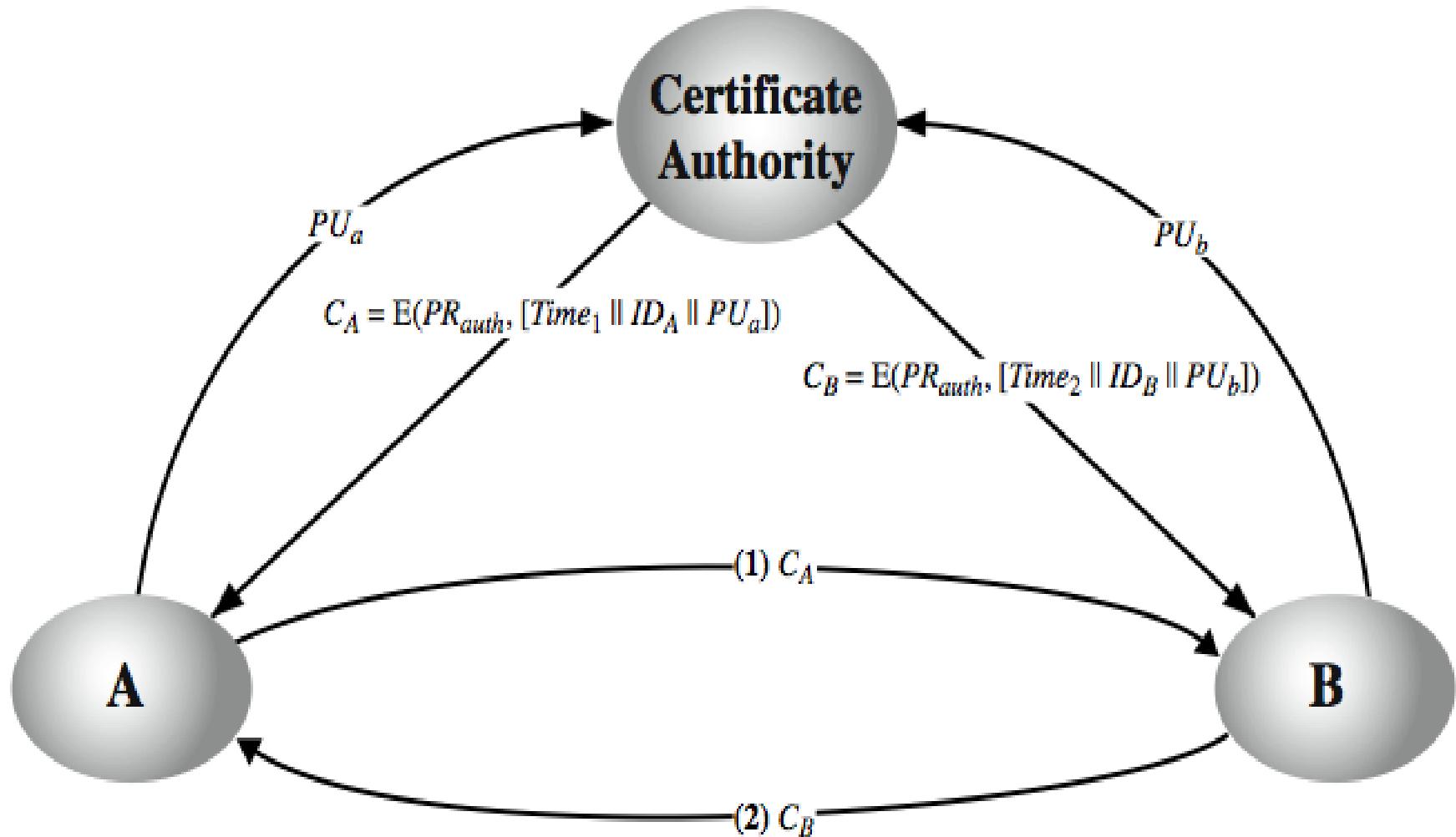
Public-Key Authority (3)

- The scenario assumes that a central authority maintains a dynamic directory of public keys of all participants.
- In addition, each participant reliably knows a public key for the authority, with only the authority knowing the corresponding private key
- A total of seven messages are required. However, the initial four messages need be used only infrequently because both A and B can save the other's public key for future use, a technique known as caching.
- Periodically, a user should request fresh copies of the public keys of its correspondents to ensure currency.

X.509 authentication and certificates

- Public-Key Certificates
 - certificates allow key exchange without real-time access to public-key authority
 - a certificate binds identity to public key
 - usually with other info such as period of validity, rights of use etc
 - with all contents signed by a trusted Public-Key or Certificate Authority (CA)
 - can be verified by anyone who knows the public-key authorities public-key

Public-Key Certificates



Public-Key Certificates (2)

- Each participant applies to the certificate authority, supplying a public key and requesting a certificate.
- Application must be in person or by some form of secure authenticated communication.
- For participant A, the authority provides a certificate CA. A may then pass this certificate on to any other participant, who can read and verify the certificate by verifying the signature from the certificate authority. Because the certificate is readable only using the authority's public key, this verifies that the certificate came from the certificate authority.

Public-Key Certificates (3)

- The timestamp counters the following scenario.
- A's private key is learned by an adversary. A generates a new private/public key pair and applies to the certificate authority for a new certificate. Meanwhile, the adversary replays the old certificate to B. If B then encrypts messages using the compromised old public key, the adversary can read those messages. In this context, the compromise of a private key is comparable to the loss of a credit card. The owner cancels the credit card number but is at risk until all possible communicants are aware that the old credit card is obsolete. Thus, the timestamp serves as something like an expiration date. If a certificate is sufficiently old, it is assumed to be expired.

X.509 Authentication Service

- part of CCITT X.500 directory service standards
 - distributed servers maintaining user info database
- defines framework for authentication services
 - directory may store public-key certificates
 - with public key of user signed by certification authority
- also defines authentication protocols
- uses public-key crypto & digital signatures
 - algorithms not standardised, but RSA recommended
- X.509 certificates are widely used
 - have 3 versions (1988, 1993, 1995-2000)

X.509 Certificate Use

Unsigned certificate:

contains user ID,
user's public key



Generate hash
code of unsigned
certificate

H



Sign hash code
with CA's private key
to form signature



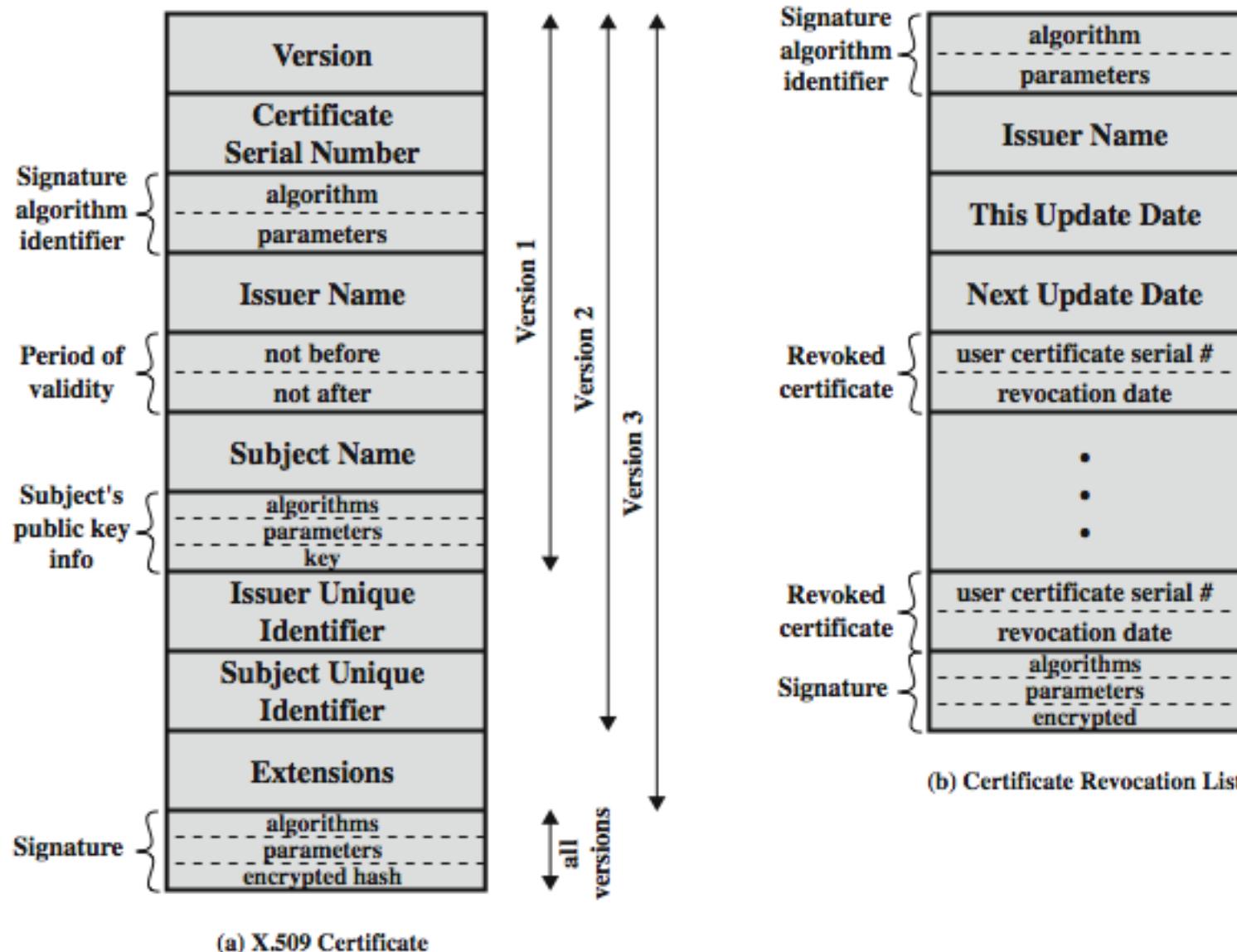
Signed certificate:

Recipient can verify
signature using CA's
public key.

X.509 Certificates

- issued by a Certification Authority (CA), containing:
 - version V (1, 2, or 3)
 - serial number SN (unique within CA) identifying certificate
 - signature algorithm identifier AI
 - issuer X.500 name CA)
 - period of validity TA (from - to dates)
 - subject X.500 name A (name of owner)
 - subject public-key info Ap (algorithm, parameters, key)
 - issuer unique identifier (v2+)
 - subject unique identifier (v2+)
 - extension fields (v3)
 - signature (of hash of all fields in certificate)
- notation $CA\ll A \gg$ denotes certificate for A signed by CA

X.509 Certificates



Obtaining a Certificate

- any user with access to CA can get any certificate from it
- only the CA can modify a certificate
- because cannot be forged, certificates can be placed in a public directory

CA Hierarchy

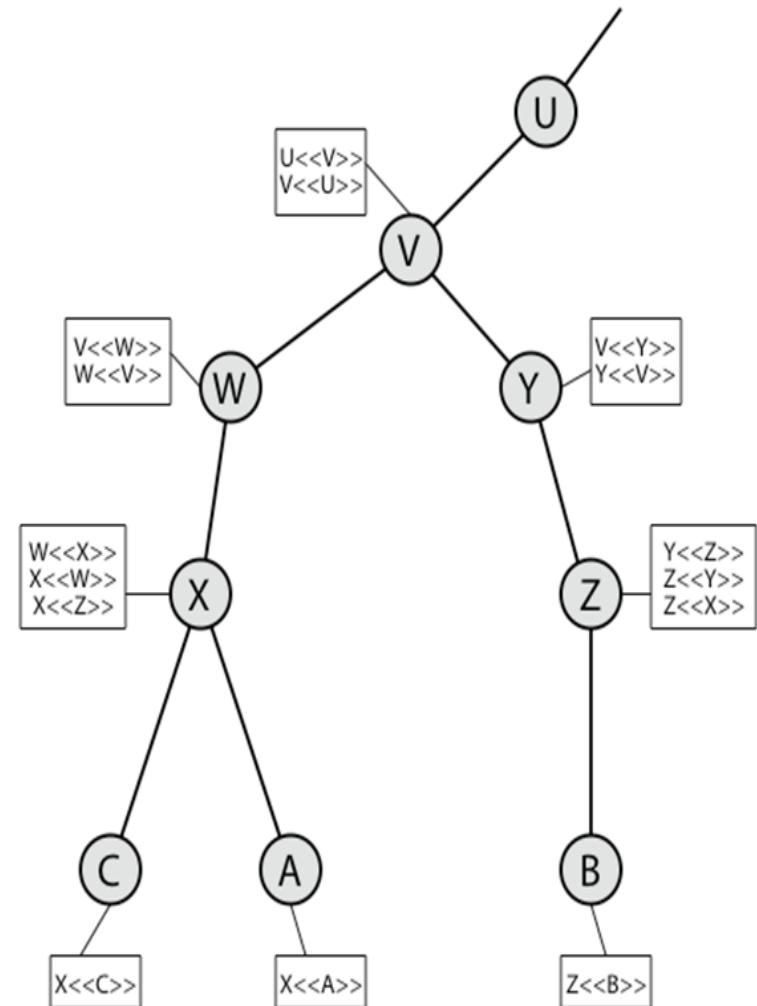
- if both users share a common CA then they are assumed to know its public key
- otherwise CA's must form a hierarchy
- use certificates linking members of hierarchy to validate other CA's
- each CA has certificates for clients (forward) and parent (backward)
- each client trusts parents certificates
- enable verification of any certificate from one CA by users of all other CAs in hierarchy

CA Hierarchy Use

- A can verify B's certificate using the following certificate chain:
 $X<<W>>W<<V>>V<<Y>>Y<<Z>>Z<>$
- Likewise, B can verify A's public key using the following certificate chain:
 $Z<<Y>>Y<<V>>V<<W>>W<<X>>X<<A>>$

All users can obtain these certificates from the directory.

notation $X<<W>>$ denotes certificate for W signed by X



Certificate Revocation

- certificates have a period of validity
- may need to revoke before expiry, e.g.:
 - user's private key is compromised
 - user is no longer certified by this CA
 - CA's certificate is compromised
- CA's maintain list of revoked certificates
 - the Certificate Revocation List (CRL)
- users should check certificates with CA's CRL

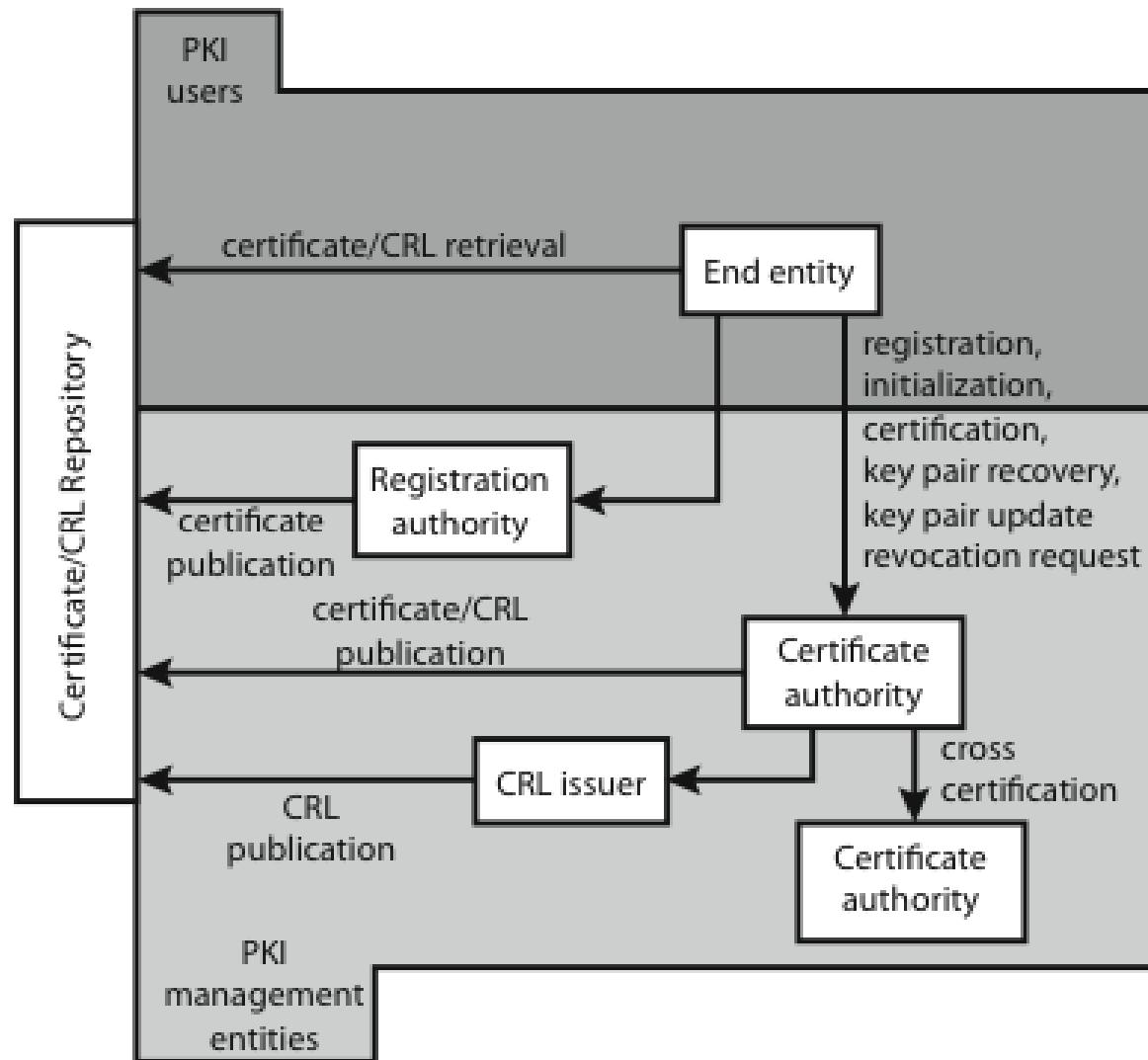
X.509 Version 3

- has been recognised that additional information is needed in a certificate
 - email/URL, policy details, usage constraints
- rather than explicitly naming new fields defined a general extension method
- extensions consist of:
 - extension identifier
 - criticality indicator
 - indicates whether an extension can be safely ignored or not (in which case if unknown the certificate is invalid)
 - extension value

Certificate Extensions

- key and policy information
 - convey info about subject & issuer keys, plus indicators of certificate policy
- certificate subject and issuer attributes
 - support alternative names, in alternative formats for certificate subject and/or issuer
- certificate path constraints
 - allow constraints on use of certificates by other CA's

Public Key Infrastructure



Public Key Infrastructure (2)

RFC 2822 (Internet Security Glossary) defines public-key infrastructure (PKI) as the set of hardware, software, people, policies, and procedures needed to create, manage, store, distribute, and revoke digital certificates based on asymmetric cryptography. Its principal is to enable secure, convenient, and efficient acquisition of public keys. The IETF Public Key Infrastructure X.509 (PKIX) working group has setup a formal (and generic) model based on X.509 that is suitable for deploying a certificate-based architecture on the Internet.

- End entity: A generic term used to denote end users, devices (e.g., servers, routers), or any other entity that can be identified in the subject field of a public key certificate. End entities can consume and/or support PKI-related services.

Public Key Infrastructure (3)

- Certification authority (CA): The issuer of certificates and (usually) certificate revocation lists (CRLs). It may also support a variety of administrative functions, although these are often delegated to Registration Authorities.
- Registration authority (RA): An optional component that can assume a number of administrative functions from the CA. The RA is often associated with the End Entity registration process, but can assist in a number of other areas as well.
- CRL issuer: An optional component that a CA can delegate to publish CRLs.
- Repository: A generic term used to denote any method for storing certificates and CRLs so that they can be retrieved by End Entities.

Thank
you



CS 6355/4355: Cryptanalysis and Database Security

Topic 8: User Authentication

Lecturer: Rongxing LU

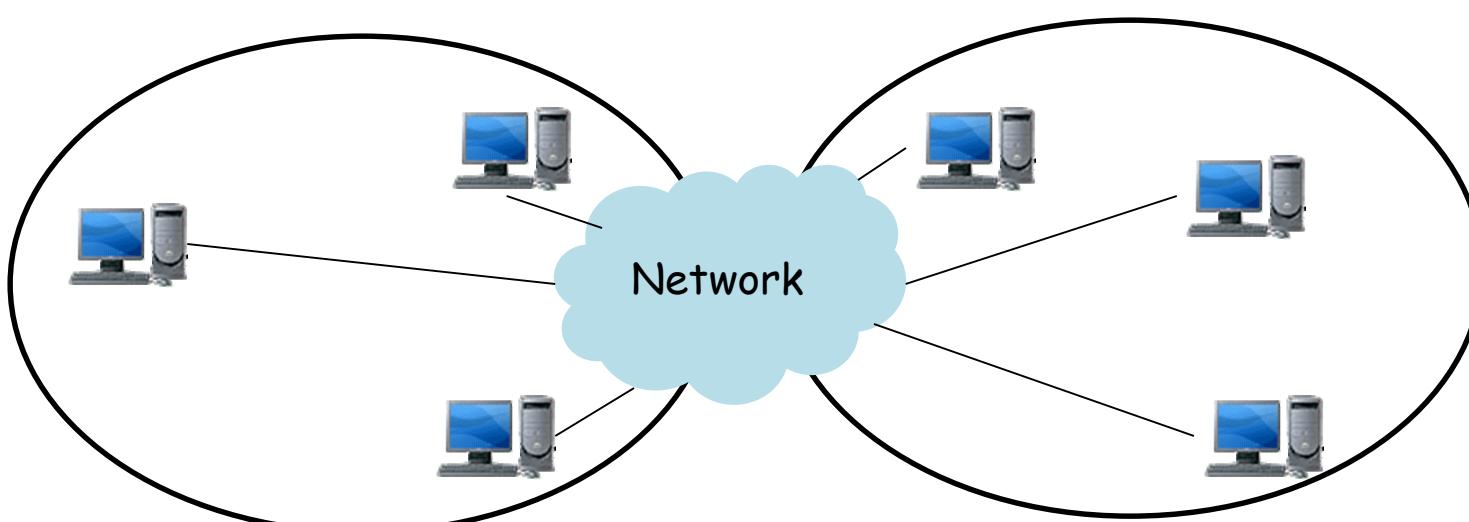
Email: RLU1@unb.ca Office: GE 114

Website: <http://www.cs.unb.ca/~rlu1/>

Faculty of Computer Science, University of New Brunswick

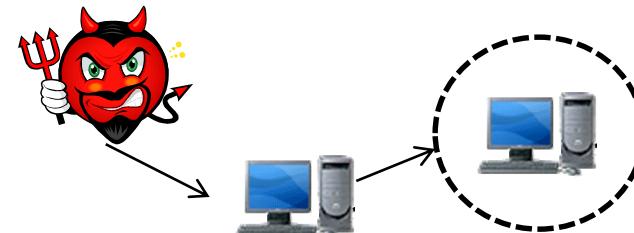
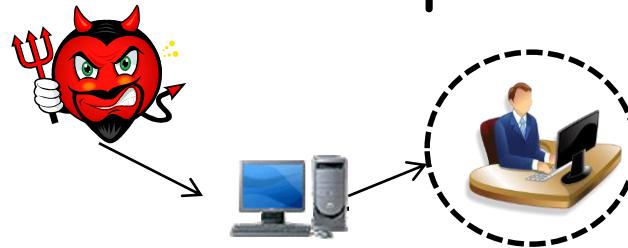
What are Distributed Systems

- A **distributed system** is a collection of computers linked via some network.
- **Characteristics:** The components of the distributed system may be under the authority of different organizations, and may be governed by different security policies.



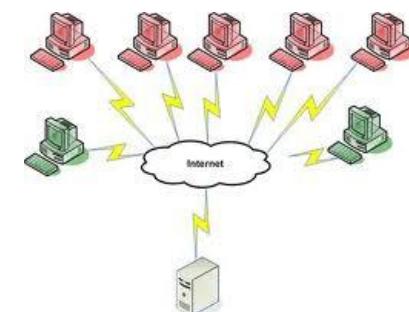
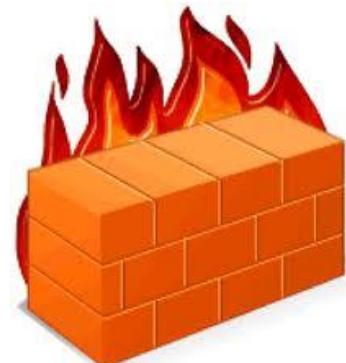
Security Issues in Distributed Systems

- Impersonation of user:
 - A user may gain access to a particular workstation and pretend to be another user operating from that workstation.
- Impersonation of workstation:
 - A user may alter the network address of a workstation so that the requests sent from the altered workstation appear to come from the impersonated workstation.



Security Services in Distributed Systems

- Authentication
- Guarding the boundaries of internal networks
 - Firewalls
- Access control to distributed objects
 - Access control techniques
- Availability
 - Counter DoS techniques



Remote User-Authentication Principles

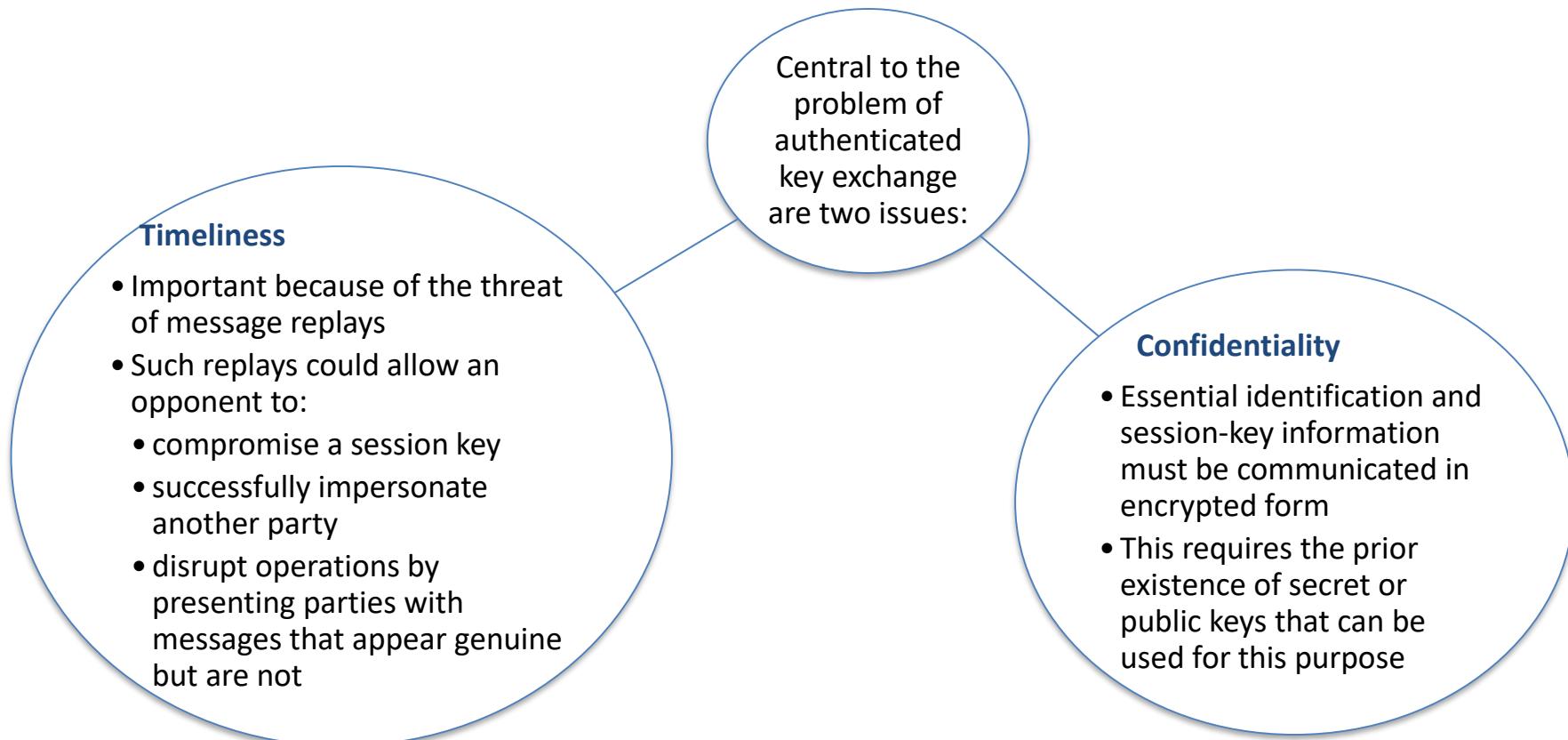
- The process of verifying an identity claimed by or for a system entity
- An authentication process consists of two steps:
 - Identification step
 - Presenting an identifier to the security system
 - Verification step
 - Presenting or generating authentication information that corroborates the binding between the entity and the identifier

Means of User Authentication

- There are four general means of authenticating a user's identity, which can be used alone or in combination
 - 1. Something the individual knows
 - Examples include a password, a personal identification number (PIN), or answers to a prearranged set of questions
 - 2. Something the individual possesses
 - Examples include cryptographic keys, electronic keycards, smart cards, and physical keys
 - This is referred to as a token
 - 3. Something the individual is (static biometrics)
 - Examples include recognition by fingerprint, retina, and face
 - 4. Something the individual does (dynamic biometrics)
 - Examples include recognition by voice pattern, handwriting characteristics, and typing rhythm
 - For network-based user authentication, the most important methods involve cryptographic keys and something the individual knows, such as a password

Mutual Authentication

- Protocols which enable communicating parties to satisfy themselves mutually about each other's identity and to exchange session keys



One-Way Authentication

One application for which encryption is growing in popularity is electronic mail (e-mail)

- Header of the e-mail message must be in the clear so that the message can be handled by the store-and-forward e-mail protocol, such as SMTP or X.400
- The e-mail message should be encrypted such that the mail-handling system is not in possession of the decryption key

A second requirement is that of authentication

- The recipient wants some assurance that the message is from the alleged sender

Remote User-Authentication Using Symmetric Encryption

A two-level hierarchy of symmetric keys can be used to provide confidentiality for communication in a distributed environment

- Strategy involves the use of a trusted key distribution center (KDC)
- Each party shares a secret key, known as a master key, with the KDC
- KDC is responsible for generating keys to be used for a short time over a connection between two parties and for distributing those keys using the master keys to protect the distribution

Kerberos – Authentication Protocol

- Centralized network authentication service
- Developed in the Project Athena in MIT
- In Greek Mythology, the three headed guard dog of Hades
- Provides a centralized authentication server to authenticate users to servers and servers to users.
- Relies on conventional encryption, making no use of public-key encryption
- Two versions: version 4 and 5, Version 4 makes use of DES



Requirements for Kerberos

- **Secure**: Opponent cannot impersonate a user and the Kerberos service should not be a weak link.
- **Reliable**: Highly reliable Kerberos service to ensure availability of supported service of application services.
- **Transparent**: Users are only required to enter a password once and don't know the authentication.
- **Scalable**: System can support large number of clients and servers.

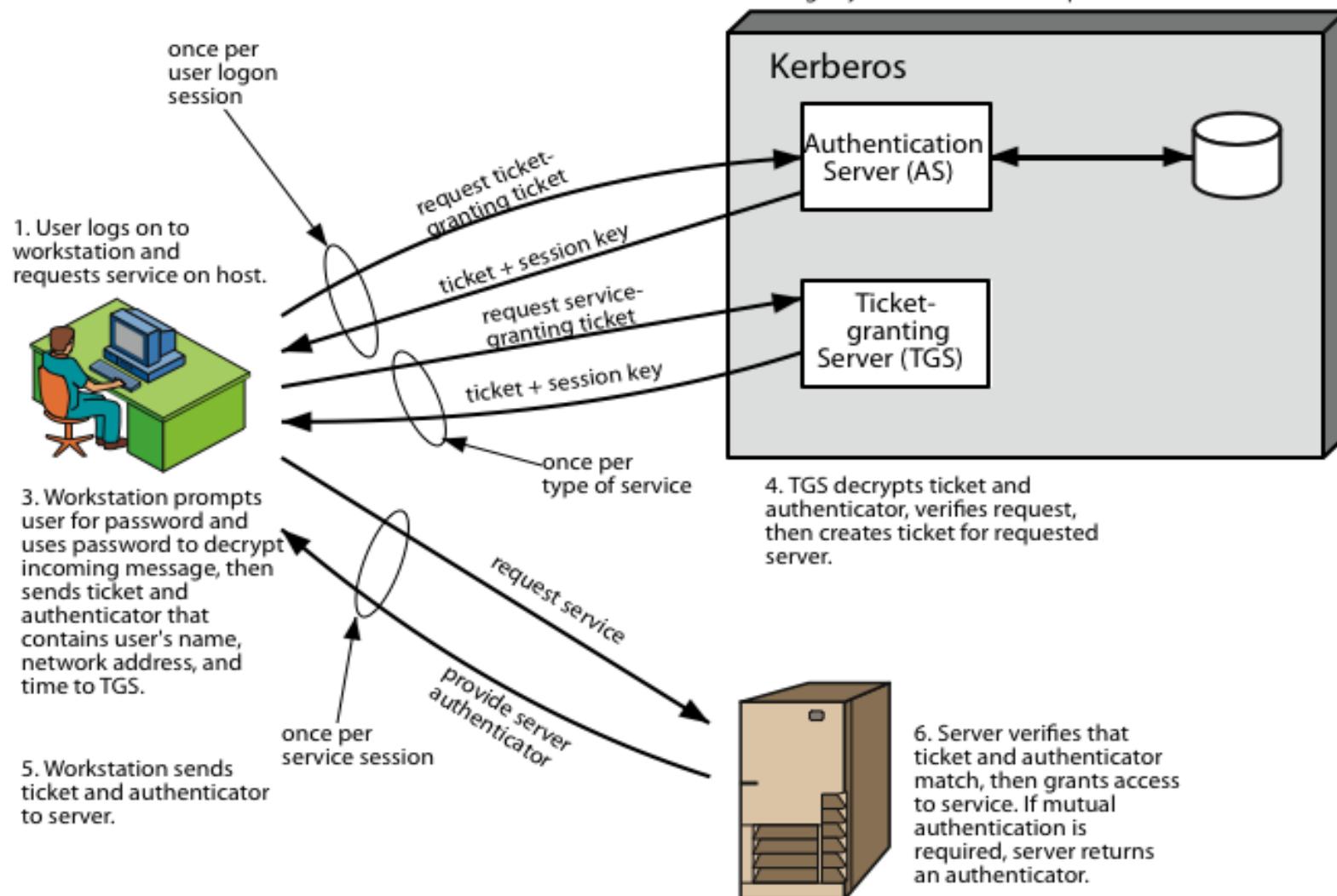


Overview of Kerberos 4

- A basic third-party authentication scheme
- Have an Authentication Server (AS)
 - Users initially negotiate with AS to identify self
 - AS provides a non-corruptible authentication credential (ticket granting ticket TGT)
- Have a Ticket Granting Server (TGS)
 - Users subsequently request access to other services from TGS on basis of users TGT.

Overview of Kerberos 4

2. AS verifies user's access right in database, creates ticket-granting ticket and session key. Results are encrypted using key derived from user's password.



Terms in Kerberos 4

- Terms:
 - C = Client
 - AS = authentication server
 - V = server
 - ID_c = identifier of user on C
 - ID_v = identifier of V
 - P_c = password of user on C
 - AD_c = network address of C
 - K_v = secret encryption key shared by AS and V
 - TS = timestamp
 - \parallel = concatentat
 - $K_{c,tgs}$ = the session key for C and TGS, generated by AS
 - ID_{tgs} = TGS's ID
 - K_c = the secret key shared between C and the AS.
 - K_{tgs} = the secret key shared between the TGS and the AS.
 - K_v = the secret key shared between V and the TGS
 - $K_{c,v}$ = the session key for C and V

A Simple Authentication Dialogue

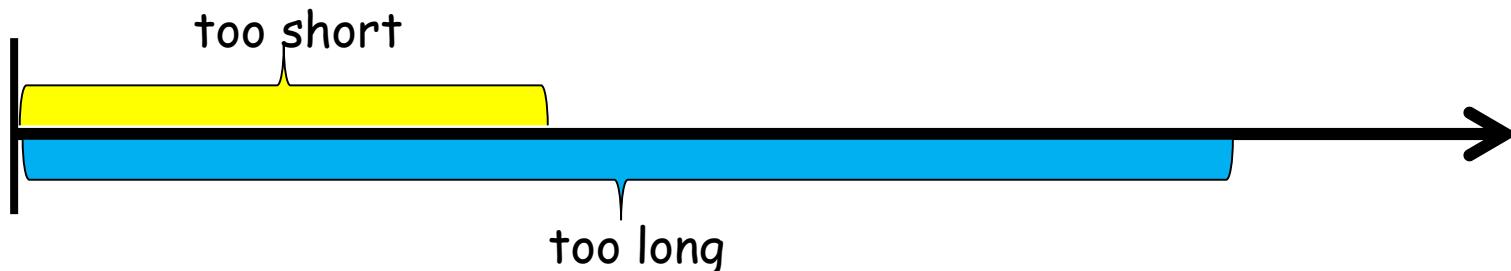
(1) $C \rightarrow AS:$	$ID_c \parallel P_c \parallel ID_v$
(2) $AS \rightarrow C:$	Ticket
(3) $C \rightarrow V:$	$ID_c \parallel \text{Ticket}$

$$\text{Ticket} = E_{K_v}[ID_c \parallel P_c \parallel ID_v]$$

- Problems with this simple scheme?
Passwords
 - a. Frequently transmitted
 - b. As plaintext

Kerberos Version 4 Authentication Dialogue

- Problems:
 - Lifetime associated with the ticket-granting ticket
 - If too short → repeatedly asked for password
 - If too long → greater opportunity to replay
 - ❖ The threat is that an adversary will steal the ticket and use it before it expires



Kerberos Version 4 Authentication Dialogue

Authentication Service Exchange: To obtain Ticket-Granting Ticket

(1) $C \rightarrow AS:$

$$ID_c || ID_{tgs} || TS_1$$

(2) $AS \rightarrow C:$

$$E_{K_c}[K_{c,tgs}] || ID_{tgs} || TS_2 || Lifetime_2 || Ticket_{tgs}$$
$$Ticket_{tgs} = E_{K_{tgs}}[K_{c,tgs} || ID_c || AD_c || ID_{tgs} || TS_2 || Lifetime_2]$$

Ticket-Granting Service Exchange: To obtain Service-Granting Ticket

(3) $C \rightarrow TGS:$

$$ID_v || Ticket_{tgs} || Authenticator_c$$

(4) $TGS \rightarrow C:$

$$E_{K_{c,tgs}}[K_{c,v}] || ID_v || TS_4 || Ticket_v$$
$$Ticket_{tgs} = E_{K_{tgs}}[K_{c,tgs} || ID_c || AD_c || ID_{tgs} || TS_2 || Lifetime_2]$$
$$Ticket_v = E_{K_V}[K_{c,v} || ID_c || AD_c || ID_v || TS_4 || Lifetime_4]$$
$$Authenticator_c = E_{K_{c,tgs}}[ID_c || AD_c || TS_3]$$

Client/Server Authentication Exchange: To Obtain Service

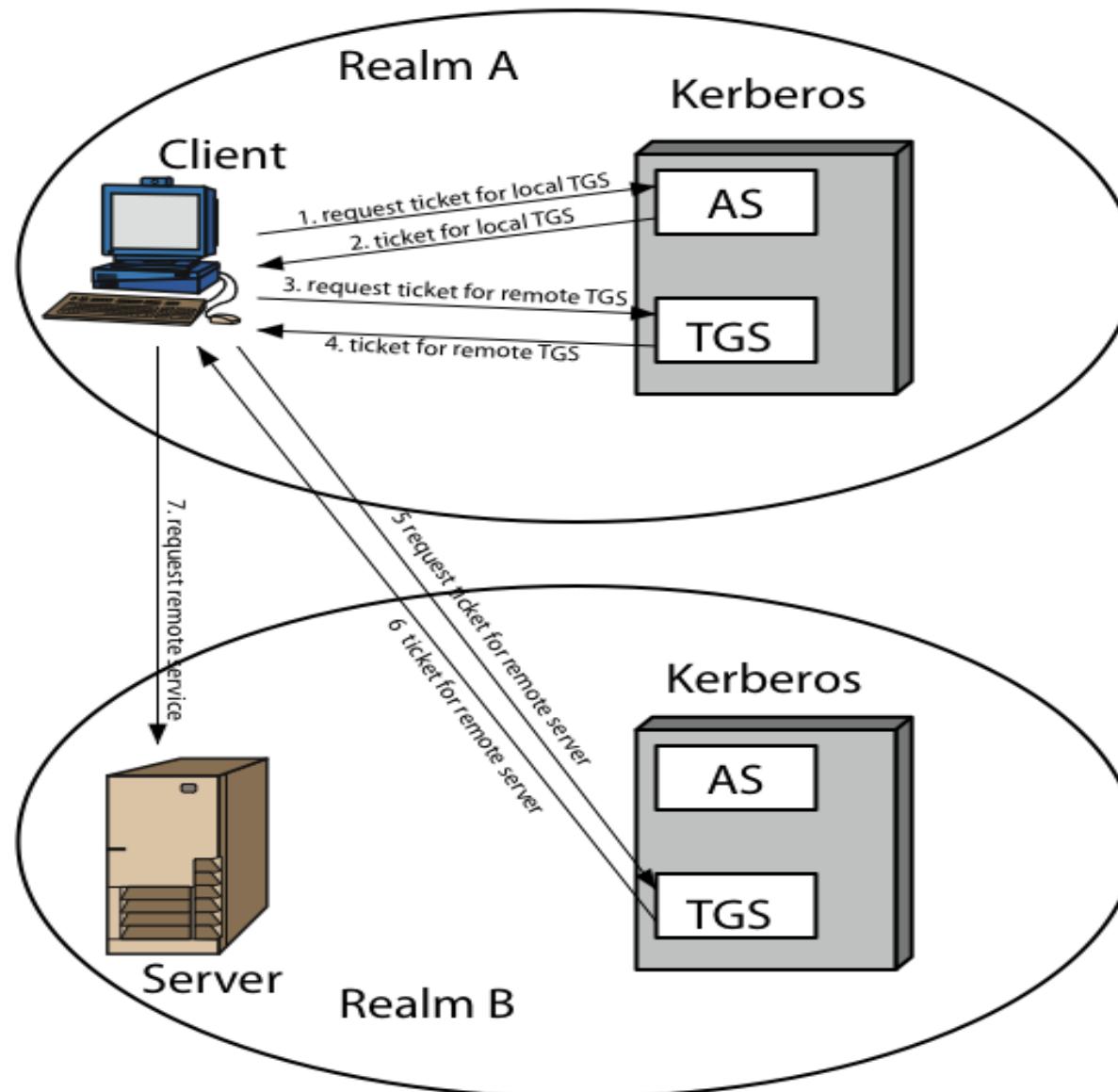
(5) $C \rightarrow V:$

$$Ticket_v || Authenticator_c$$

(6) $V \rightarrow C:$

$$E_{K_{c,v}}[TS_5 + 1] \quad (\text{optional: for mutual authentication})$$
$$Ticket_v = E_{K_V}[K_{c,v} || ID_c || AD_c || ID_v || TS_4 || Lifetime_4]$$
$$Authenticator_c = E_{K_{c,v}}[ID_c || AD_c || TS_5]$$

Request for Service in Another Realm

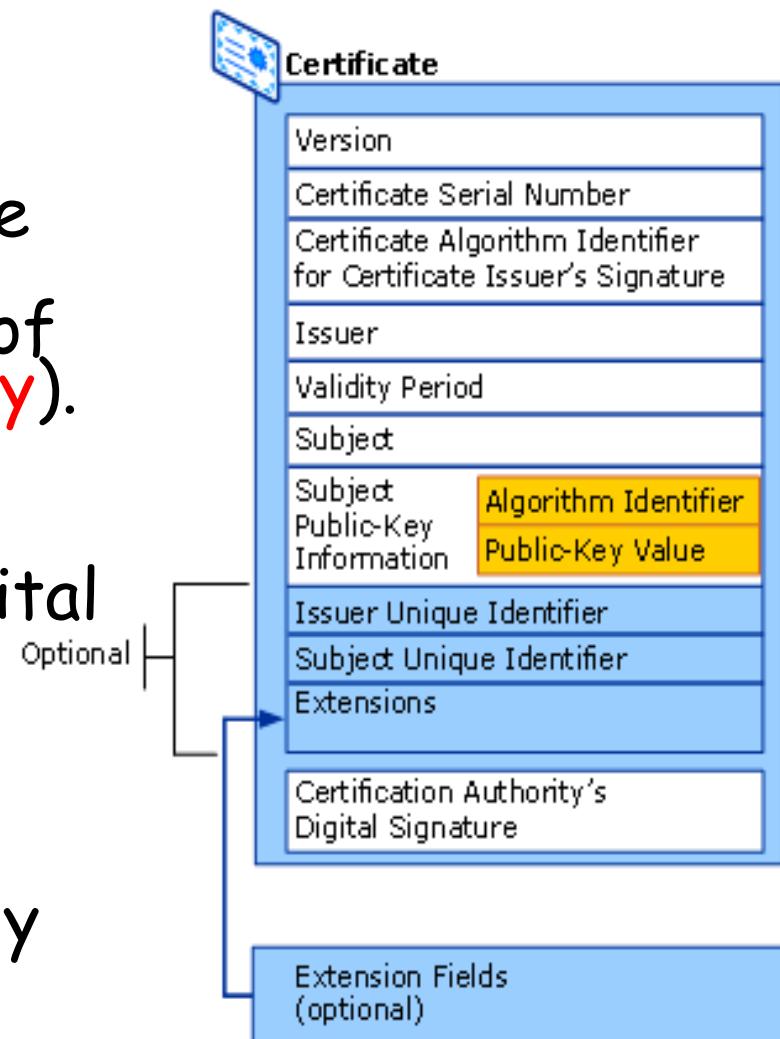


Kerberos Version 5

- Developed in mid 1990's to address the deficiencies of v4
- Provides improvements over v4
 - encryption algorithm: DES is weak and vulnerable to attacks. V5 allows a suit of encryption algorithms.
 - V5 breaks away from IP only networks, to use any internet protocol
 - V4 uses 8bit ticket lifetime. V5 uses start time and end time, 1280 minutes (maximum time) → any length of time
- Technical deficiencies
 - Double encryption in V4
 - PCBC encryption (a new mode of operation, non-standard)
 - In V5, Standard CBC is used.

X.509 Authentication Service

- distributed set of servers that maintains a database about users.
- each certificate contains the public key of a user and is signed with the private key of a CA (**Certification Authority**).
- also defines authentication protocols
- uses public-key crypto & digital signatures
 - Algorithms not standardised, but RSA is recommended to use.
- X.509 certificates are widely used

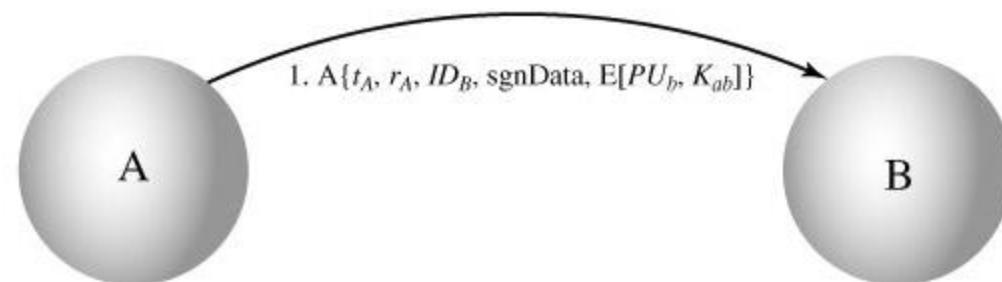


Authentication Procedures

- X.509 includes three alternative authentication procedures:
- One-Way Authentication
- Two-Way Authentication
- Three-Way Authentication
- All authentication use public-key signatures

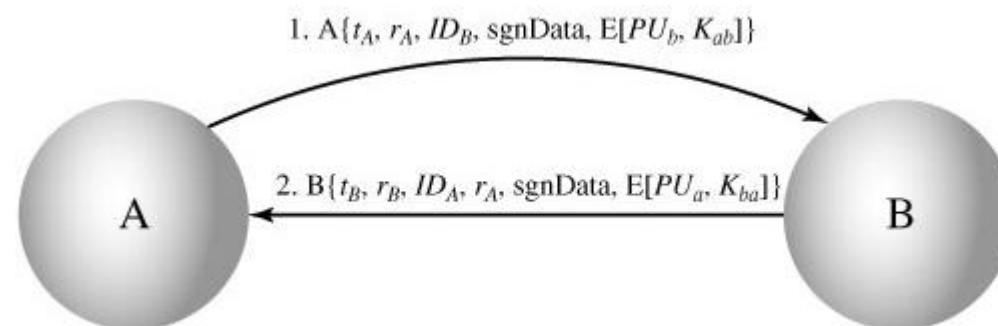
One-Way Authentication

- 1 message ($A \rightarrow B$) used to establish
 - the identity of A and that message is from A
 - message was intended for B
 - integrity & originality of message
- message must include timestamp, nonce, B's identity and is signed by A
- may include additional info for B
 - eg session key



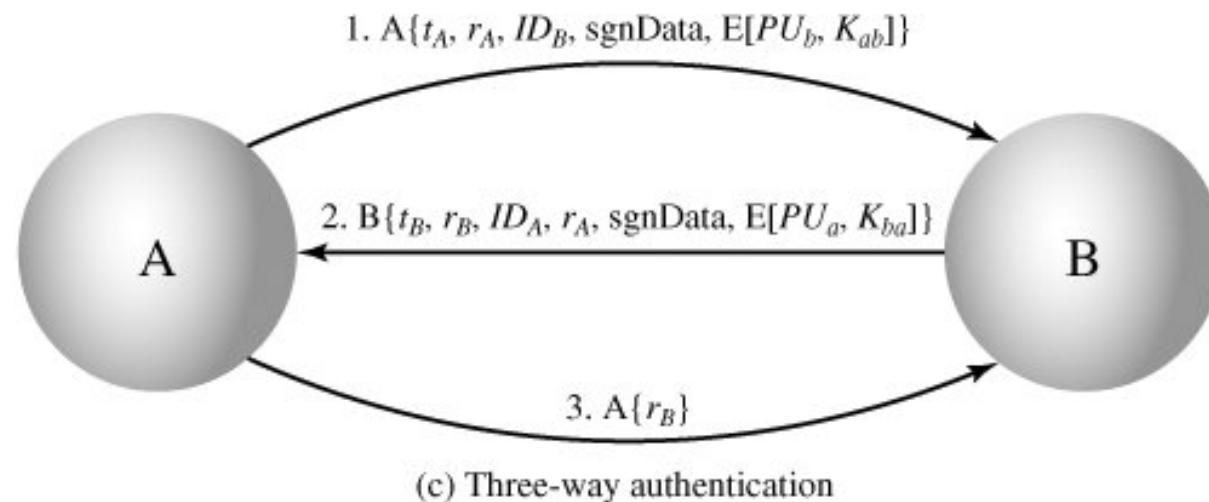
Two-Way Authentication

- 2 messages ($A \rightarrow B$, $B \rightarrow A$) which also establishes in addition:
 - the identity of B and that reply is from B
 - that reply is intended for A
 - integrity & originality of reply
- reply includes original nonce from A, also timestamp and nonce from B
- may include additional info for A



Three-Way Authentication

- 3 messages ($A \rightarrow B$, $B \rightarrow A$, $A \rightarrow B$) which enables above authentication without synchronized clocks
- has reply from A back to B containing signed copy of nonce from B
- means that timestamps need not be checked or relied upon



Thank
you



CS 6355/4355: Cryptanalysis and Database Security

Topic 9: Network Access Control and Cloud Security

Lecturer: Rongxing LU

Email: RLU1@unb.ca Office: GE 114

Website: <http://www.cs.unb.ca/~rlu1/>

Faculty of Computer Science, University of New Brunswick

Network Access Control (NAC)

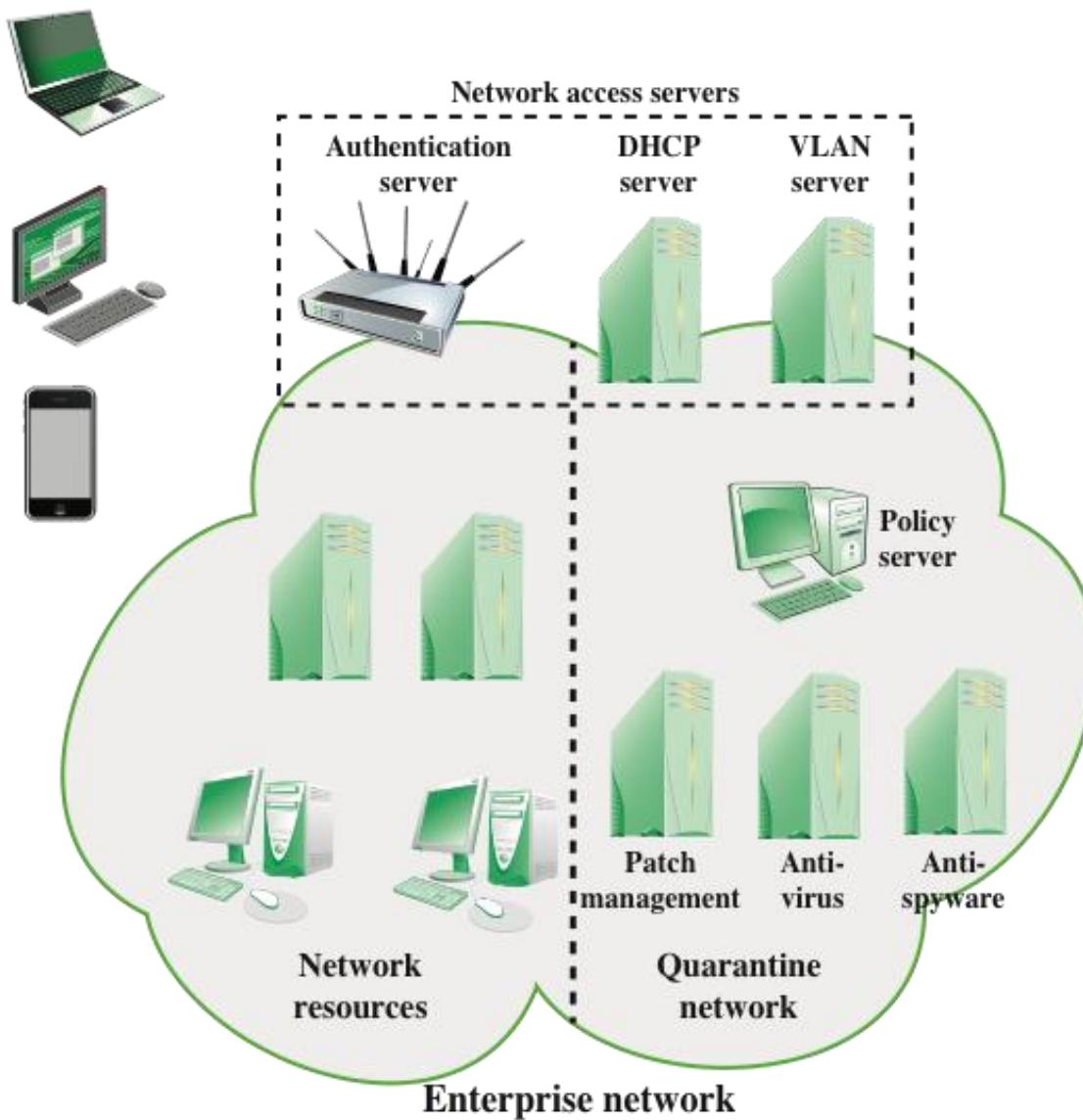
- An umbrella term for managing access to a network
- Authenticates users logging into the network and determines what data they can access and actions they can perform
- Also examines the health of the user's computer or mobile device

Elements of a Network Access Control System

- **Access requestor (AR)**: referred to as supplicants, or clients. The AR is the node that is attempting to access the network and may be any device that is managed by the NAC system.
- **Policy server**: Based on the AR's posture and an enterprise's defined policy, the **policy server** determines what access should be granted.
- **Network access server (NAS)**: Also called a media gateway, a remote access server (RAS), or a policy server. The **NAS** functions as an access control point for users in remote locations connecting to an enterprise's internal network.

Network Access Control (NAC)

Suplicants



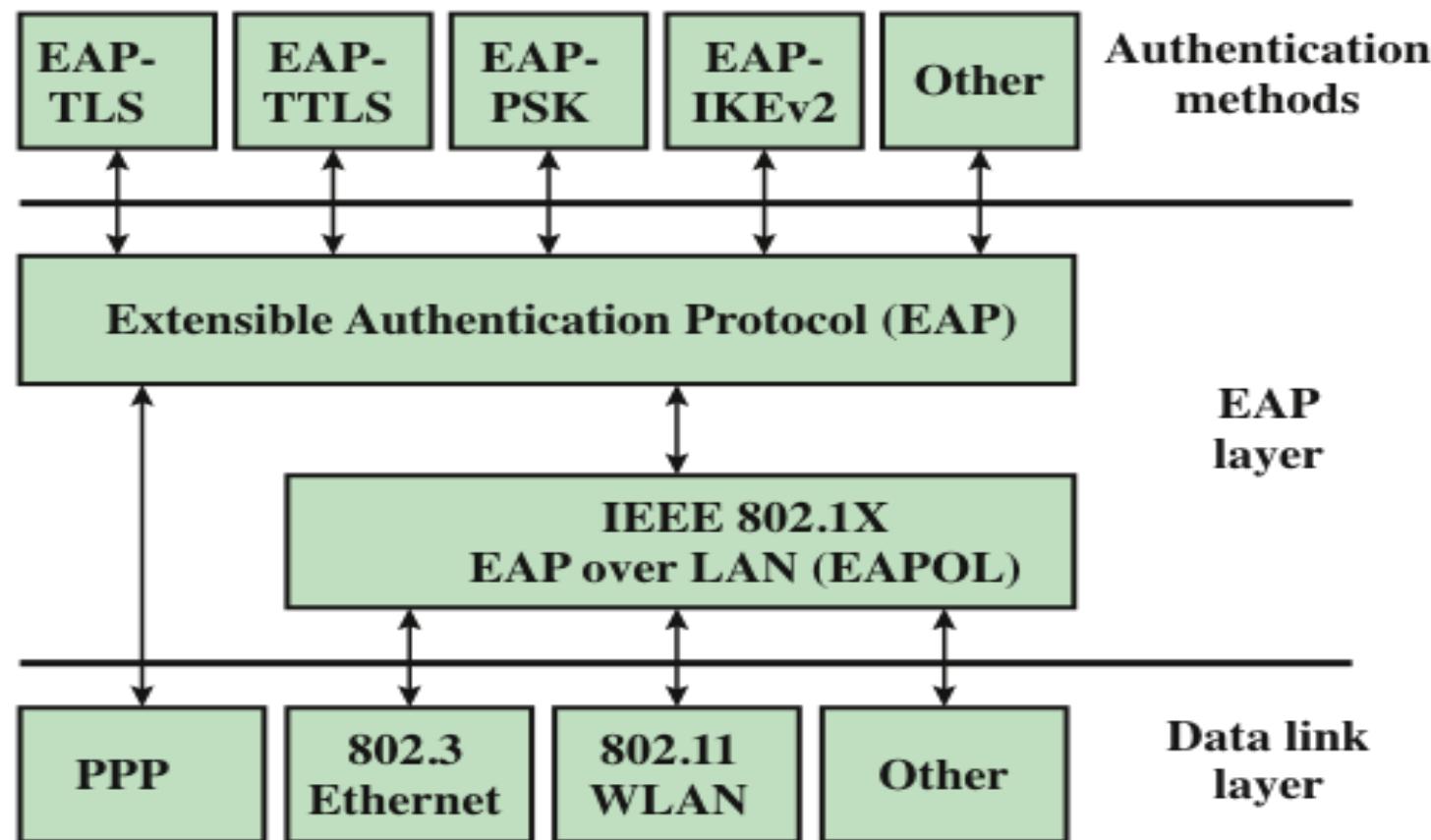
Network Access Enforcement Methods

- Enforcement methods are the actions that are applied to ARs to regulate access to the enterprise network.
 - IEEE 802.1X: enforces authorization before a port is assigned an IP address. IEEE 802.1X makes use of the Extensible Authentication Protocol for the authentication process.
 - Virtual local area networks (VLANs): the enterprise network, consisting of an interconnected set of LANs, is segmented logically into a number of virtual LANs. The NAC system decides to which of the network's VLANs it will direct an AR.
 - Firewall: allow or deny network traffic between an enterprise host and an external user.
 - DHCP management: DHCP enables dynamic allocation of IP addresses to hosts. A DHCP server intercepts DHCP requests and assigns IP addresses. Thus, NAC enforcement occurs at the IP layer based on subnet and IP assignment.

Extensible Authentication Protocol

- The Extensible Authentication Protocol (EAP) acts as a framework for network access and authentication protocols.
- EAP provides a set of protocol messages, encapsulate various authentication methods to be used between a client and an authentication server.
- EAP can operate over a variety of network and link level facilities, including point-to-point links, LANs, and other networks, and can accommodate the authentication needs of the various links and networks.

EAP Layered Context

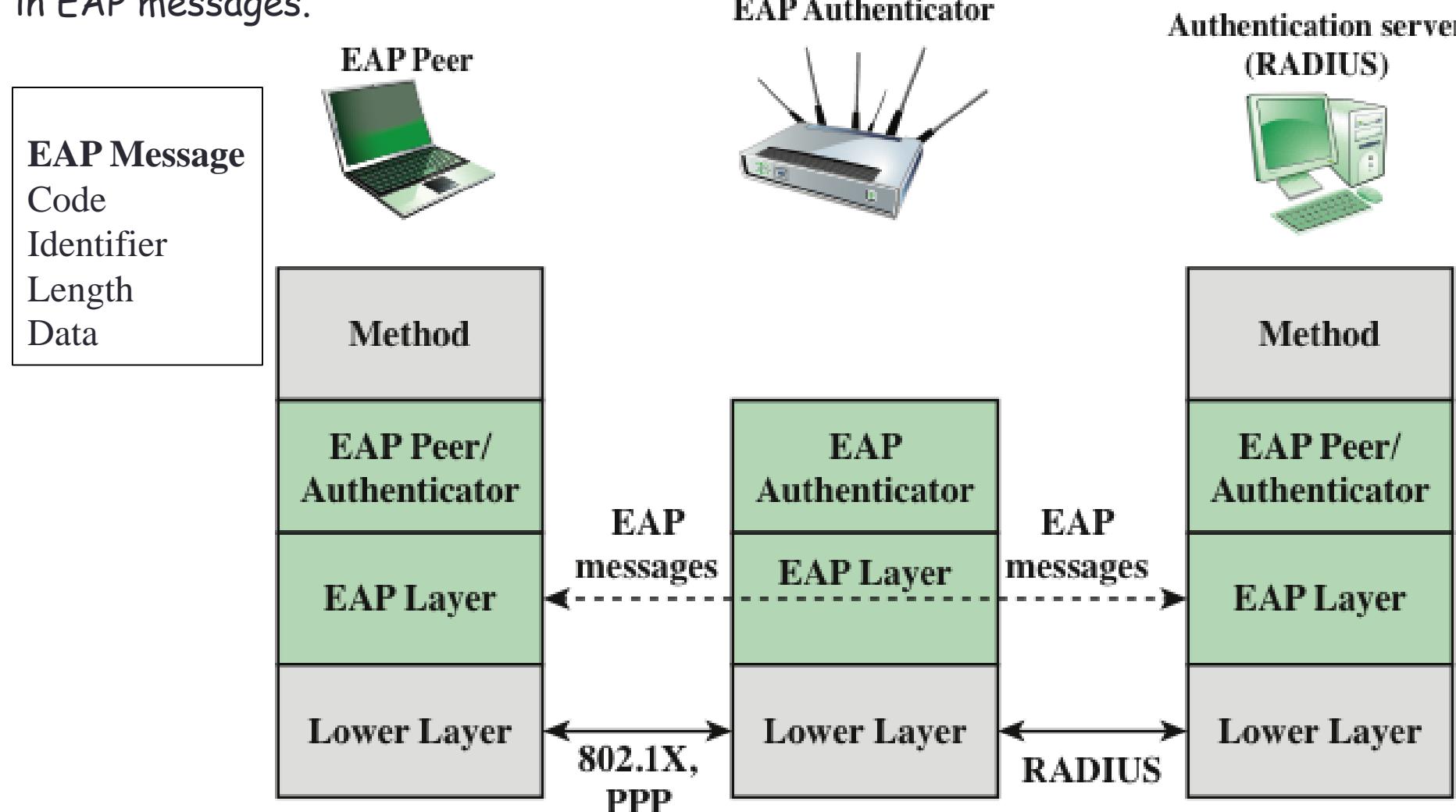


EAP Authentication

- **EAP-TLS (EAP Transport Layer Security)**: defines how the TLS protocol can be encapsulated in EAP messages. It uses the handshake protocol in TLS.
- **EAP-TTLS (EAP Tunneled TLS)**: like EAP-TLS, except only the server has a certificate to authenticate itself to the client first. In EAP-TLS, a secure connection (the "tunnel") is established with secret keys.
- **EAP-GPSK (EAP Generalized Pre-Shared Key)**: is an EAP method for mutual authentication and session key derivation using a pre-shared key (PSK). It specifies an EAP method based on PSKs and employs secret key-based cryptographic algorithms.
- **EAP-IKEv2**: based on the Internet Key Exchange protocol ver.2 (IKEv2). It supports mutual authentication and session key establishment using a variety of methods.

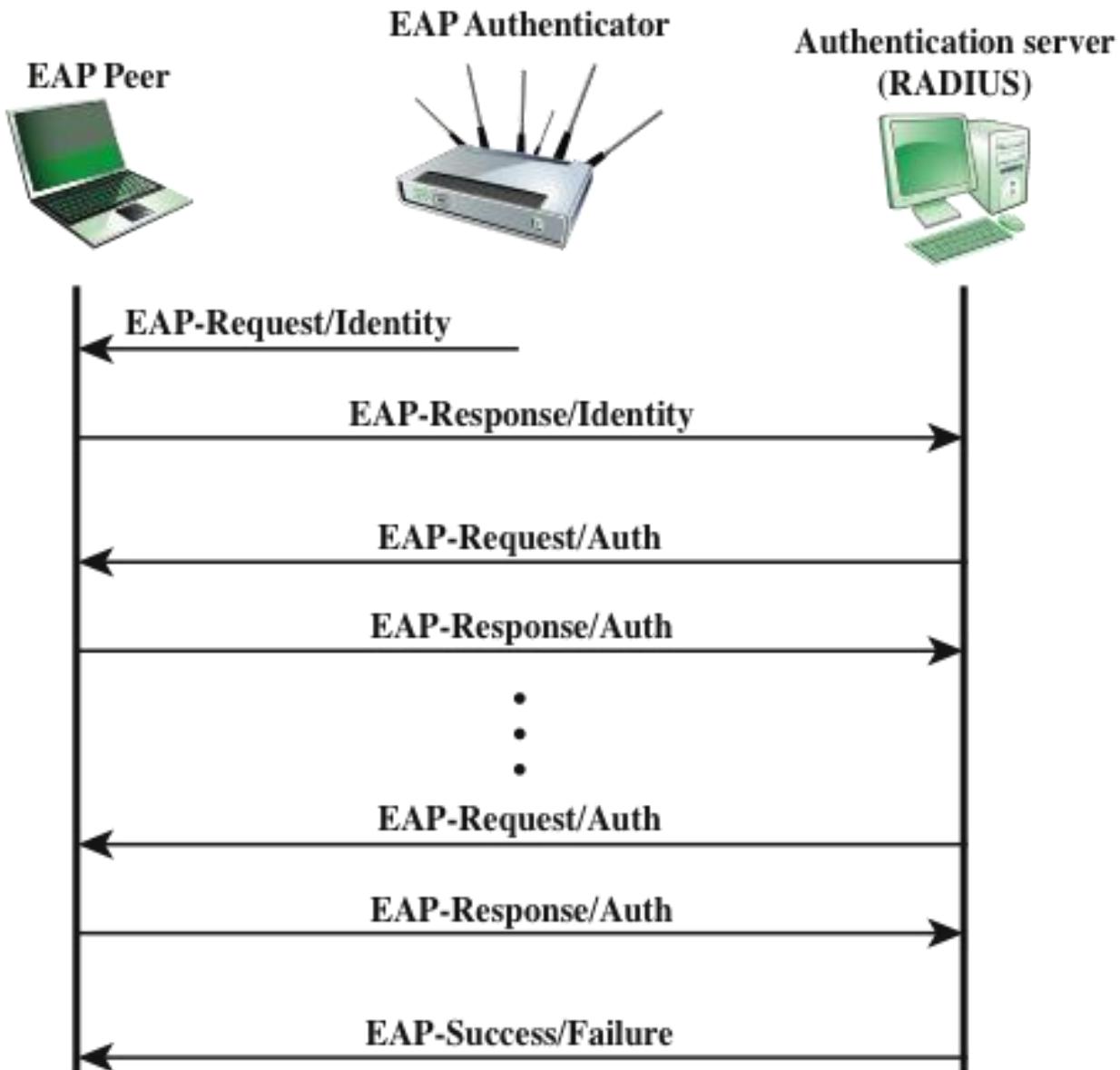
The authentication information and authentication protocol information are carried in EAP messages.

EAP Exchanges



EAP Exchanges

EAP Message Flow in Pass-Through Mode



EAP Exchanges

- The first pair of EAP Request and Response messages is of Type identity, in which the authenticator requests the peer's identity, and the peer returns its claimed identity in the Response message. This Response is passed through the authenticator to the authentication server. Subsequent EAP messages are exchanged between the peer and the authentication server.
- Upon receiving the identity Response message from the peer, the server selects an EAP method and sends the first EAP message with a Type field related to an authentication method.
- If the peer supports and accepts the selected EAP method, it replies with the corresponding Response message of the same type.
- Otherwise, the peer sends a NAK, and the EAP server either selects another EAP method or aborts the EAP execution with a failure message.
- The selected EAP method determines the number of Request/Response pairs. During the exchange the appropriate authentication information, including key material, is exchanged.
- The exchange ends when the server determines that authentication has succeeded or that no further attempt can be made and authentication has failed.

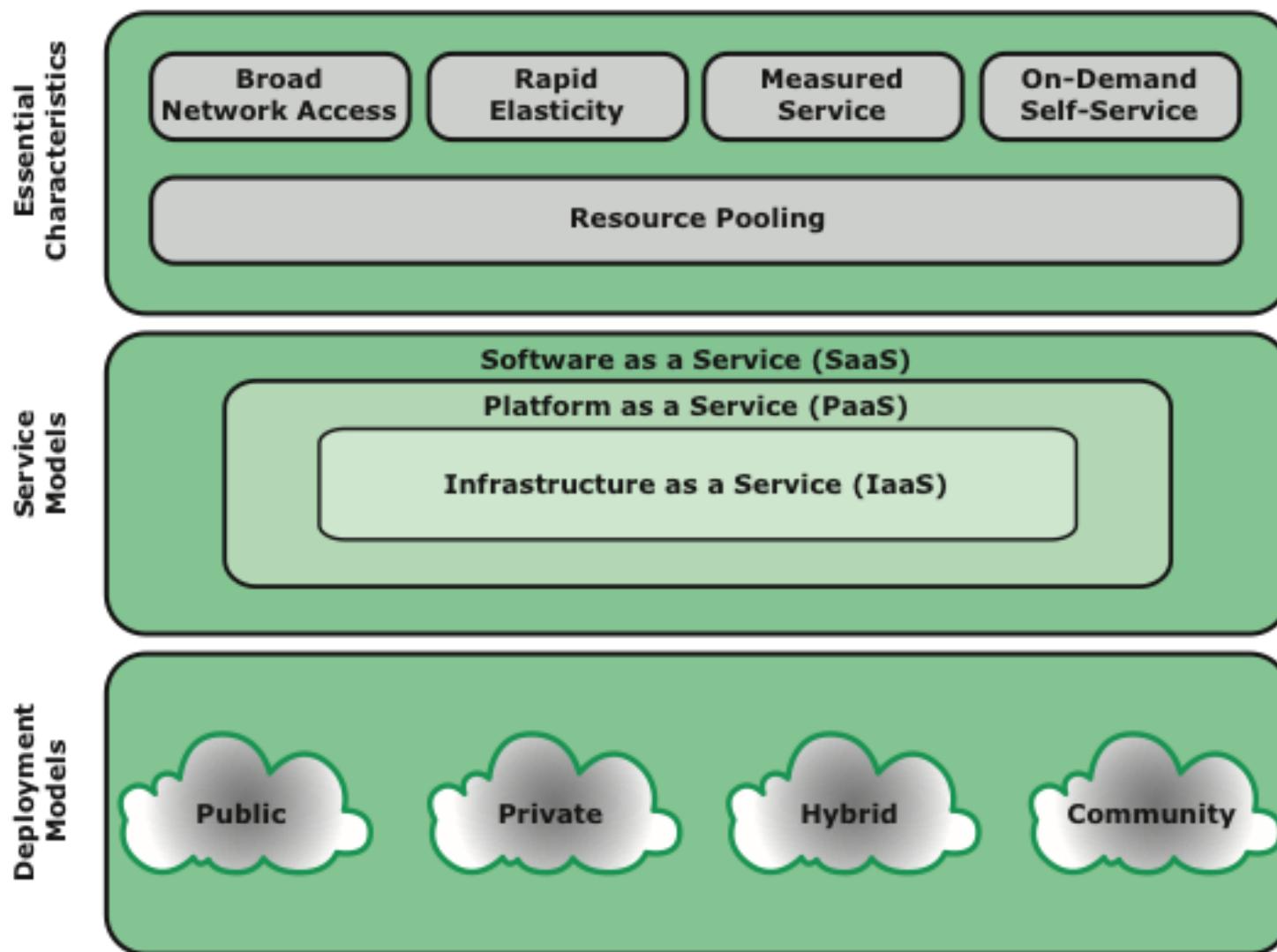
Cloud Computing

- NIST defines cloud computing, in NIST SP-800-145 (*The NIST Definition of Cloud Computing*), as follows:

"A model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model promotes availability and is composed of five essential characteristics, three service models, and four deployment models."



Cloud Computing Elements



Cloud Computing Characteristics

- **Resources** related to some aspects, such as storage, processing, memory, network bandwidth, and virtual machine.
- **Broad network access** - available over the network and accessed through standard mechanisms, use by client platforms or other cloud-based services.
- **Rapid elasticity** - ability to expand and reduce resources according to specific requirements.
- **Measured service** - control and optimize resource suitable to the appropriate type of service. Resource usage can be monitored, controlled, reported, provide clearly utilized service.
- **On-demand self-service** - ability to provision resource capabilities automatically, no need human interaction. The resources is temporary in IT infrastructure.
- **Resource pooling** - ability to serve multiple consumers using a multi-tenant model, with different physical and virtual resources, dynamically assigned and reassigned base on consumer demand.

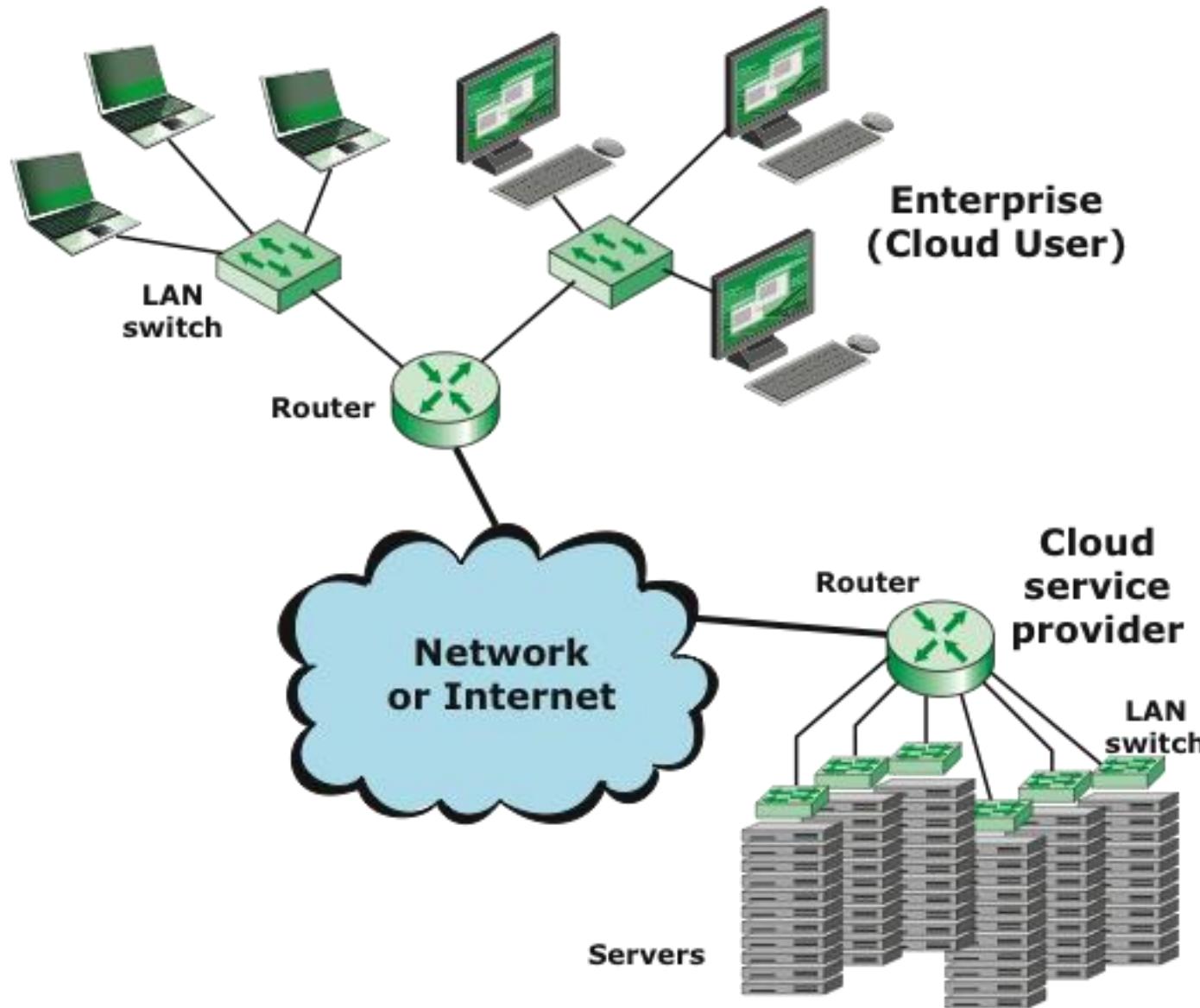
Cloud Computing Service Models

- **Software as a Service (SaaS)** - the capability allows consumer to use the provider's application running on a cloud infrastructure. The applications are accessible from various client devices by just a thin client interface (Web browser). **SaaS** saves the complexity of software installation, maintenance, upgrades, patches.
- **Platform as a Service (PaaS)** - the capability allows consumer to deploy onto the cloud infrastructure consumer or acquired applications - created. Also, **PaaS** provides middleware-style services , such as database and component services use by apps. **PaaS** is such like an operating system in the cloud.
- **Infrastructure as a Service (IaaS)** - the capability allows consumer to provision processing, storage, networks, and other computing resources, that is used to deploy and run various software. **IaaS** enables customers to combine basic computing services to build highly adaptable computer systems.

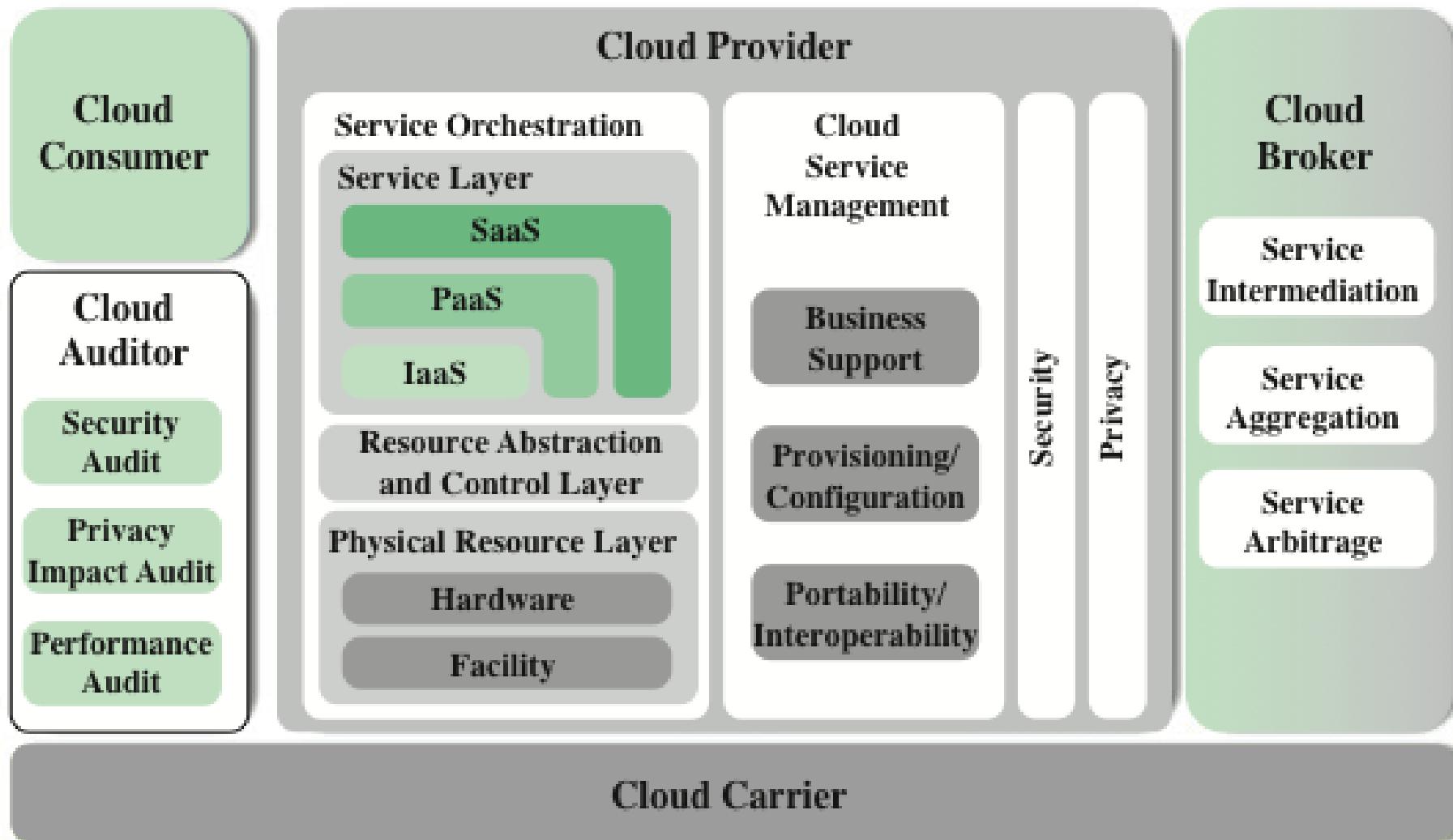
Cloud Computing Deployment Models

- **Public cloud** - available to the general public or a large industry group, is owned by an organization selling cloud services. The cloud provider (**CP**) is responsible for cloud infrastructure and for control data and operations within cloud.
- **Private cloud** - operated solely for an organization, managed by organization or a third party. The **CP** is responsible only for the infrastructure.
- **Community cloud** - shared by several organizations and supports a specific community shared specific concerns (mission, policy, security ...), managed by the organization or a third party.
- **Hybrid cloud** - is a composition of two or more clouds, remain unique entities but are bound together by standardized or proprietary technology that enables data and application portability (e.g., cloud bursting for load balancing between clouds).

Cloud Computing Context



Cloud Computing Reference Architecture



Cloud Computing Reference Architecture

- **Cloud consumer** - a person or organization maintains a business relationship with, and uses service from, cloud providers.
- **Cloud provider** - a person, organization, or entity responsible for making a service available to interested parties.
- **Cloud auditor** - a party conducts independent assessment of cloud services, info. system operations, performance, and security of cloud implementation.
- **Cloud broker** - an entity manages the use, performance, and delivery of cloud services, and negotiates relationships between CP and consumers.
- **Cloud carrier** - an intermediary provides connectivity and transport of cloud services from CPs to consumers.



Cloud Security Risks & Countermeasures

- **Abuse and nefarious use of cloud computing** - The easy of register and use cloud service leads to high risks from attackers inside the cloud, such as spamming, malicious code attacks, or DOS attack.

Countermeasures: (1) stricter initial registration and validation processes, (2) enhance credit card fraud monitoring and coordination, (3) comprehensive introspection of customer network traffic, (4) monitoring public blacklists for one's network blocks.

- **Insecure interfaces and APIs** - CPs expose a set of software interfaces or APIs customers use to manage and interact with cloud services. From authentication and access control, these interfaces need to be resisted against accidental and malicious attempts.

Countermeasure: (1) analyzing the security model of CP interfaces, (2) ensuring that strong authentication and access control are implemented with encrypted transmission, (3) understanding the dependency chain associated with the API.

Cloud Security Risks & Countermeasures

- **Malicious insiders** - risk of malicious insider activity. Cloud architectures necessitate roles that extremely high risk.
Countermeasures: (1) enforce strict supply chain management and conduct a comprehensive supplier assessment, (2) specify human resource requirements as part of legal contract, (3) require transparency into overall infor. security and management practices, and compliance reporting, (4) determine security breach notification processes.
- **Shared technology issues:** IaaS vendors deliver services by sharing infrastructure which is not strong enough in isolation properties for a multi-tenant architecture.
Countermeasures: implement security best practices for installation/ configuration, (2) monitor environment for unauthorized changes/ activity, (3) promote strong authentication and access control for administrative access and operation.

Cloud Security Risks & Countermeasures

- **Data loss and leakage** - for clients. The most devastating from security breach is the loss or leakage of data.

Countermeasures: (1) implement strong API access control, (2) encrypt, protect integrity of data in transit, (3) analyze data protection at design and run-time, (4) implement strong keys generation, storage and management, destruction practices.

- **Account or service hijacking** - usually with stolen credentials, attackers can access critical areas of cloud services, allowing to compromise the confidentiality, integrity, and availability (CIA).

Countermeasures: (1) prohibit the sharing of account credentials between users and services, (2) leverage strong two-factor authentication techniques, (3) employ proactive monitoring to detect unauthorized activity, (4) understand CP security policies and SLAs.

- **Unknown risk profile** - in using cloud infrastructure, client should cedes control to the CP on a number of issues that may affect security, and pay attention, clearly define the roles and responsibilities involved for managing risks.

Countermeasures: (1) disclosure of applicable logs and data, (2) partial/full disclosure of infrastructure details (patch levels and firewalls), (3) monitoring and alerting on necessary infor

Data Protection in the Cloud

NIST Guidelines on Security and Privacy Issues and Recommendations

Governance

Extend organization practices pertaining to the policies, procedures, and standards used for application development and service provisioning in the cloud, as well as the design, implementation, testing, use, and monitoring of deployed or engaged services.

Put in place audit mechanisms and tools to ensure organizational practices are followed throughout the system life cycle.

Compliance

Understand the various types of laws and regulations that impose security and privacy obligations on the organization and potentially impact cloud computing initiatives, particularly those involving data location, privacy and security controls, records management, and electronic discovery requirements.

Review and assess the cloud provider's offerings with respect to the organizational requirements to be met and ensure that the contract terms adequately meet the requirements.

Ensure that the cloud provider's electronic discovery capabilities and processes do not compromise the privacy or security of data and applications.

Data Protection in the Cloud

Trust

Ensure that service arrangements have sufficient means to allow visibility into the security and privacy controls and processes employed by the cloud provider, and their performance over time.

Establish clear, exclusive ownership rights over data.

Institute a risk management program that is flexible enough to adapt to the constantly evolving and shifting risk landscape for the life cycle of the system.

Continuously monitor the security state of the information system to support ongoing risk management decisions.

Architecture

Understand the underlying technologies that the cloud provider uses to provision services, including the implications that the technical controls involved have on the security and privacy of the system, over the full system life cycle and across all system components.

Identity and access management

Ensure that adequate safeguards are in place to secure authentication, authorization, and other identity and access management functions, and are suitable for the organization.

Data Protection in the Cloud

Software isolation

Understand virtualization and other logical isolation techniques that the cloud provider employs in its multitenant software architecture, and assess the risks involved for the organization.

Data protection

Evaluate the suitability of the cloud provider's data management solutions for the organizational data concerned and the ability to control access to data, to secure data while at rest, in transit, and in use, and to sanitize data.

Take into consideration the risk of collating organizational data with those of other organizations whose threat profiles are high or whose data collectively represent significant concentrated value.

Fully understand and weigh the risks involved in cryptographic key management with the facilities available in the cloud environment and the processes established by the cloud provider.

Data Protection in the Cloud

Availability

Understand the contract provisions and procedures for availability, data backup and recovery, and disaster recovery, and ensure that they meet the organization's continuity and contingency planning requirements.

Ensure that during an intermediate or prolonged disruption or a serious disaster, critical operations can be immediately resumed, and that all operations can be eventually reinstated in a timely and organized manner.

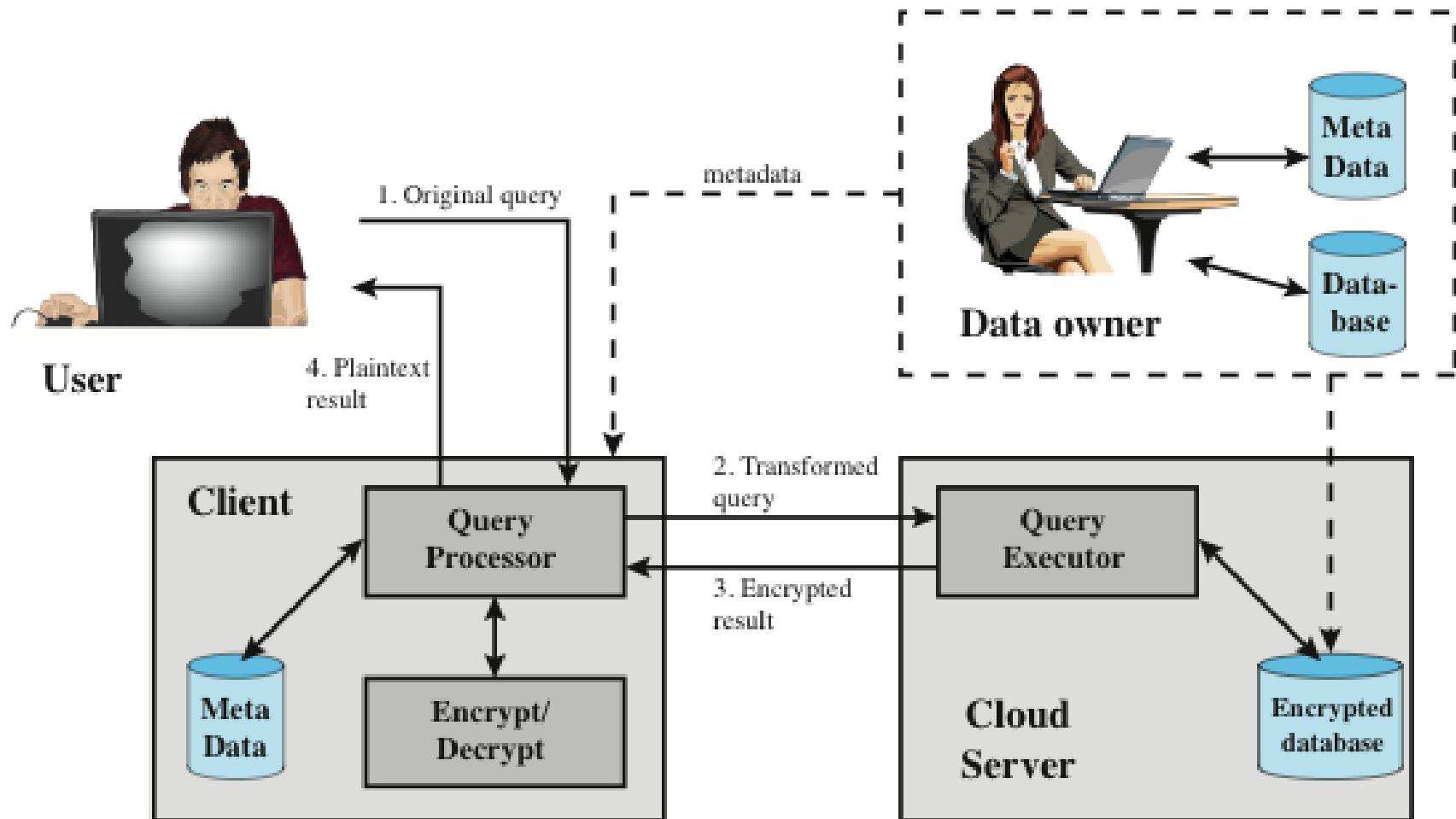
Incident response

Understand the contract provisions and procedures for incident response and ensure that they meet the requirements of the organization.

Ensure that the cloud provider has a transparent response process in place and sufficient mechanisms to share information during and after an incident.

Ensure that the organization can respond to incidents in a coordinated fashion with the cloud provider in accordance with their respective roles and responsibilities for the computing environment.

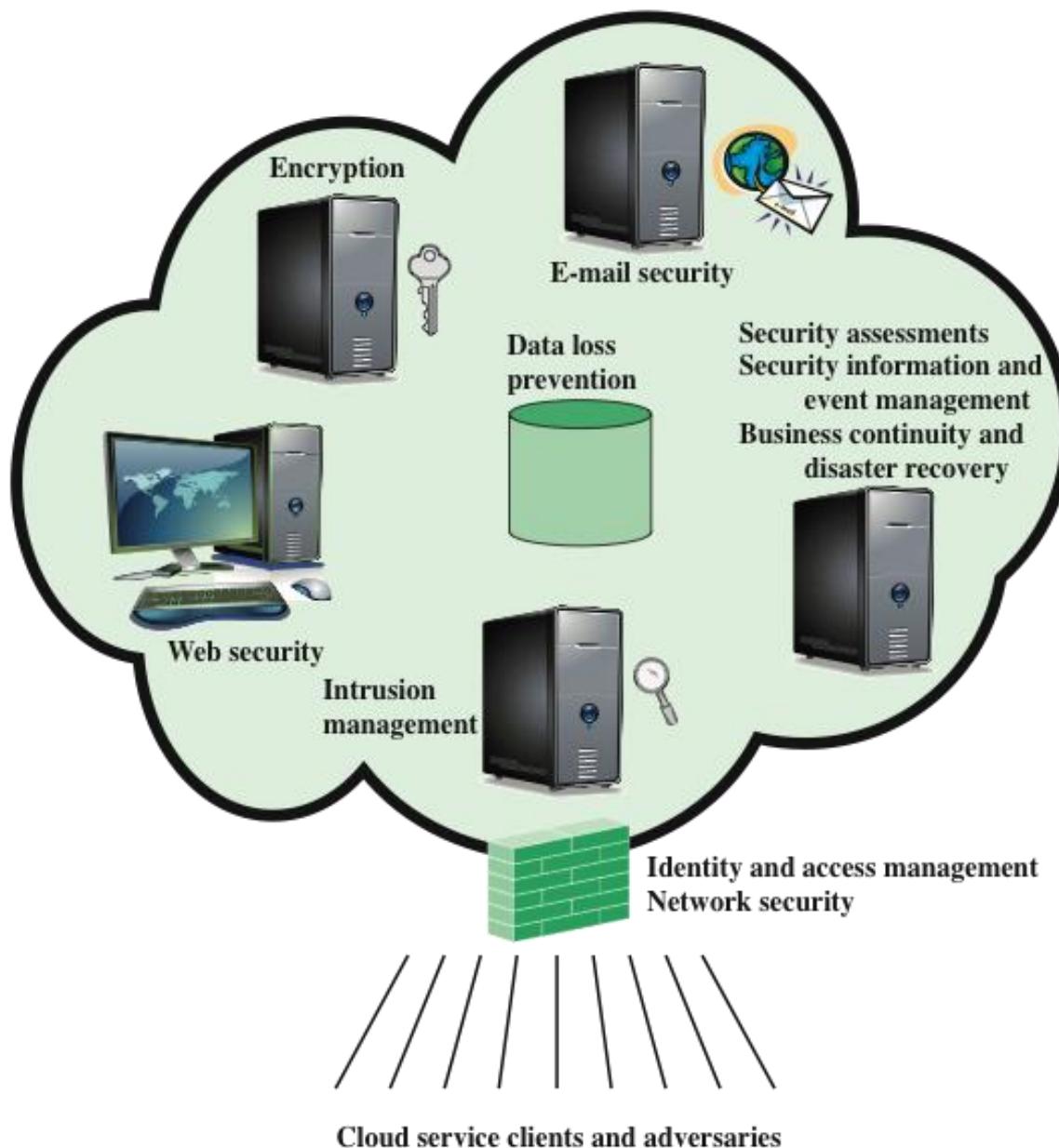
An Encryption Scheme for a Cloud-Based Database



Cloud Security as a Service (SecaaS)

- **SecaaS** is a segment of the **SaaS**, meant a package of security services offered by a service provider that offloads much of the security responsibility from an enterprise to the security service provider.
- The services: authentication, antivirus, antimalware-spyware, intrusion detection, security event management.
- **SecaaS** categories:
 - ✓ Identity and access management
 - ✓ Data loss prevention
 - ✓ Web security
 - ✓ Email security
 - ✓ Security assessments
 - ✓ Intrusion management
 - ✓ Security info. and event management
 - ✓ Encryption
 - ✓ Business continuity and disaster recovery
 - ✓ Network security

Elements of Cloud Security as a Service (SecaaS)



Cloud Security as a Service (SecaaS)

- **Identify and access management** - people, processes, and systems. Used to manage access to enterprise resources, assure the identity is verified, and grants correct level to access. It involves authentication and access control services.
- **Data loss prevention** - monitoring, protecting, and verifying the data, implemented by cloud client, make rules about what functions can be performed on data.
- **Web security** - real-time protection offered through software/appliance installation, or the cloud by proxying or redirecting web traffic to the CP. Antivirus, antimalware, usage policy enforcement, data backup, traffic control, web access control within it.
- **Email security** - provides control over inbound and outbound email, protects from phishing, malicious attachments, offers corporate policies, spam prevention, digital signatures and email encryption.
- **Security assessments** - third part audits of cloud services, provides tools and access points to facilitate assessment activities.

Cloud Security as a Service (SecaaS)

- **Intrusion management** - intrusion detection, prevention, and response, the core is intrusion detection systems (**IDSs**) and intrusion prevention systems (**IPSs**). **IDS** detects unauthorized accesses to host system, while **IPS** block traffic from intruders.
- **Security info. and event management** - aggregates log and event data from virtual and real networks, applications, and systems, provides real-time reporting and info./event alarming.
- **Encryption** - provides for data, as email traffic, client-specific network management info, and identifies info. Involves key management, application encryption, and data content access.
- **Business continuity and disaster recovery** - measures and mechanisms to ensure operational resiliency in the events or service interruptions. Includes flexible infrastructure, redundancy of functions and hardware, monitored operations, geographically distributed data centers, and network survivability.
- **Network security** - security services that allocate access, distribute, monitor, and protect resource services. Includes perimeter, server firewalls, DOS protection, in the network security service.

Thank
you



CS 6355/4355: Cryptanalysis and Database Security

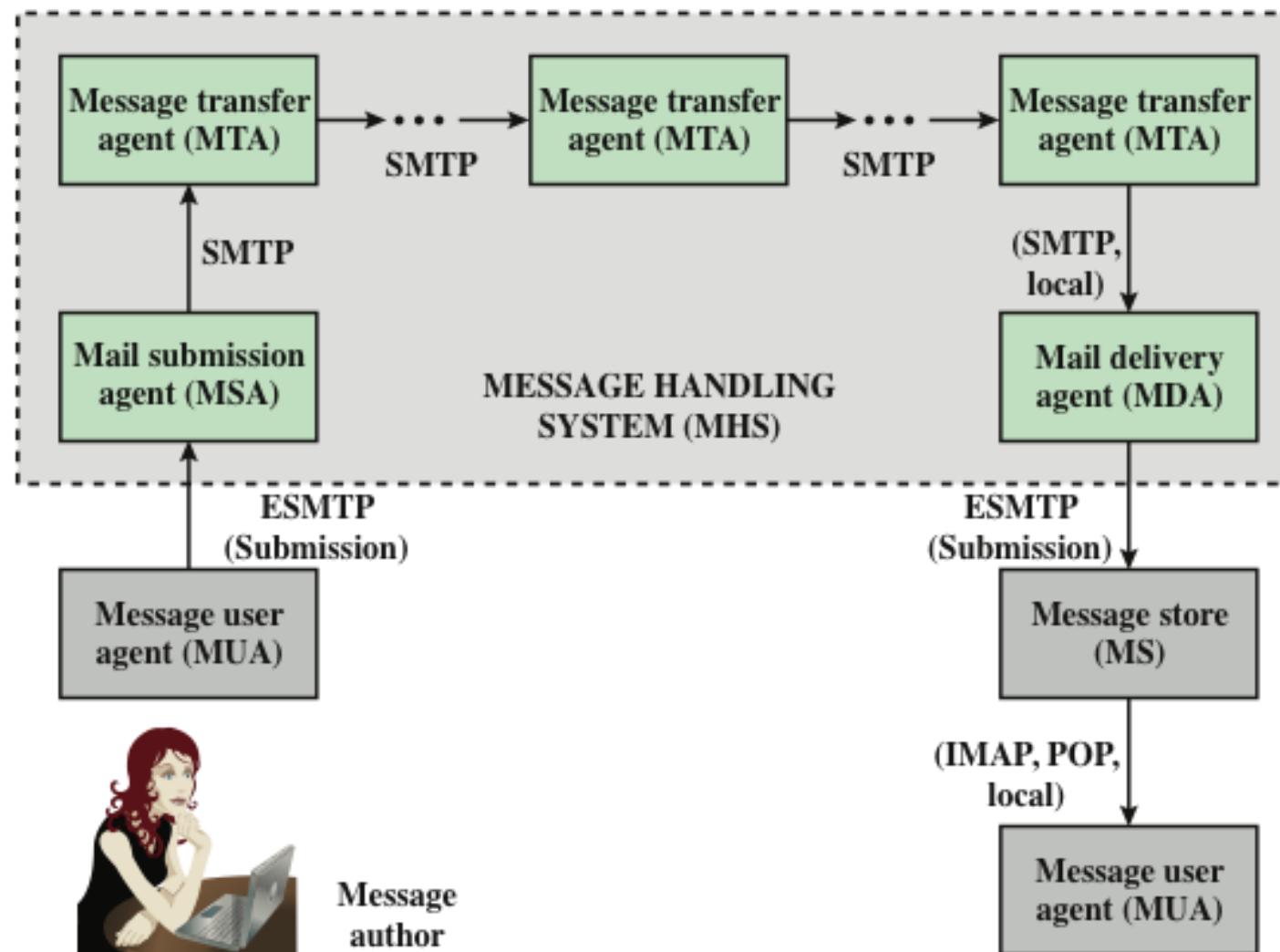
Topic 10: Electronic Mail Security

Lecturer: Rongxing LU

Email: RLU1@unb.ca Office: GE 114

Website: <http://www.cs.unb.ca/~rlu1/>

Faculty of Computer Science, University of New Brunswick



**Function Modules and Standardized Protocols
Used in the Internet Mail Architecture**

Message
recipient



Email Protocols

- Two types of protocols are used for transferring email:
 - Used to move messages through the Internet from source to destination
 - Simple Mail Transfer Protocol (SMTP)
 - Used to transfer messages between mail servers
 - IMAP and POP are the most commonly used

SMTP (Simple Mail Transfer Protocol)

- SMTP encapsulates an email message in an envelope and is used to relay the encapsulated messages from source to destination through multiple MTAs.
- SMTP was originally specified in 1982 as RFC 821 and has undergone several revisions, the most current being RFC 5321 (October 2008).
- These revisions have added additional commands and introduced extensions. The term Extended SMTP (ESMTP) is often used to refer to these later versions of SMTP.

Example SMTP Transaction Scenario

```
S: 220 foo.com Simple Mail Transfer Service Ready
C: HELO bar.com
S: 250 OK
C: MAIL FROM:<Smith@bar.com>
S: 250 OK
C: RCPT TO:<Jones@foo.com>
S: 250 OK
C: RCPT TO:<Green@foo.com>
S: 550 No such user here
C: RCPT TO:<Brown@foo.com>
S: 250 OK
C: DATA
S: 354 Start mail input; end with <CRLF>.<CRLF>
C: Blah blah blah...
C: ...etc. etc. etc.
C: <CRLF><CRLF>
S: 250 OK
C: QUIT
S: 221 foo.com Service closing transmission channel
```

Example SMTP Transaction Scenario (2)

- illustrates the SMTP exchange between a client (*C*) and server (*S*).
- The interchange begins with the client establishing a TCP connection to TCP port **25** on the server. This causes the server to activate SMTP and send a **220** reply to the client.
- The HELO command identifies the sending domain, which the server acknowledges and accepts with a **250** reply.
- The SMTP sender is transmitting mail that originates with the user `Smith@bar.com`.
- The MAIL command identifies the originator of the message.
- The message is addressed to three users on machine `foo.com`, namely, Jones, Green, and Brown.
- The client identifies each of these in a separate RCPT command. The SMTP receiver indicates that it has mailboxes for Jones and Brown but does not have information on Green.
- Because at least one of the intended recipients has been verified, the client proceeds to send the text message, by first sending a DATA command to ensure the server is ready for the data.
- After the server acknowledges receipt of all the data, it issues a **250 OK** message.
- Then the client issues a QUIT command and the server closes the connection.

STARTTLS

- A significant security-related extension for SMTP
- Defined in RFC 3207 (*SMTP Service Extension for Secure SMTP over Transport Layer Security*, February 2002)
- Enables the addition of confidentiality and authentication in the exchange between SMTP agents
- This gives SMTP agents the ability to protect some or all of their communication from eavesdroppers and attackers
 - Advantage of using STARTTLS is that the server can offer SMTP service on a single port, rather than requiring separate port numbers for secure and cleartext operations

Mail Access Protocols

- **POP3**
 - Post Office Protocol
 - Allows an email client to download an email from an email server (MTA)
 - POP3 user agents connect via TCP to the server
 - After authorization, the UA can issue POP3 commands to retrieve and delete mail
- **IMAP**
 - Internet Mail Access Protocol
 - Enables an email client to access mail on an email server
 - Also uses TCP, with server TCP port 143
 - Is more complex than POP3
 - Provides stronger authentication and provides other functions not supported by POP3

RFC 5322

- Defines a format for text messages that are sent using electronic mail
- Messages are viewed as having an envelope and contents
 - The envelope contains whatever information is needed to accomplish transmission and delivery
 - The contents compose the object to be delivered to the recipient
 - RFC 5322 standard applies only to the contents
- The content standard includes a set of header fields that may be used by the mail system to create the envelope

Example Message

Date: October 8, 2009 2:15:49 PM EDT
From: "William Stallings" <ws@shore.net>
Subject: The Syntax in RFC 5322
To: Smith@Other-host.com
Cc: Jones@Yet-Another-Host.com

Hello. This section begins the actual message body, which is delimited from the message heading by a blank line.

Multipurpose Internet Mail Extensions (MIME)

- An extension to the RFC 5322 framework that is intended to address some of the problems and limitations of the use of Simple Mail Transfer Protocol (SMTP)
 - Is intended to resolve these problems in a manner that is compatible with existing RFC 5322 implementations
 - The specification is provided in RFCs 2045 through 2049

Limitations of the SMTP/5322 Scheme

- SMTP cannot transmit executable files or other binary objects
- SMTP cannot transmit text data that includes national language characters
- SMTP servers may reject mail message over a certain size
- SMTP gateways that translate between ASCII and the character code EBCDIC do not use a consistent set of mappings, resulting in translation problems
- SMTP gateways to X.400 electronic mail networks cannot handle nontextual data included in X.400 messages
- Some SMTP implementations do not adhere completely to the SMTP standards defined in RFC 821

Email Security Threats

- Authenticity-related threats
 - Could result in unauthorized access to an enterprise's email system
- Integrity-related threats
 - Could result in unauthorized modification of email content
- Confidentiality-related threats
 - Could result in unauthorized disclosure of sensitive information
- Availability-related threats
 - Could prevent end users from being able to send or receive mail

How to develop a system for e-mail security

- Having learned the basics of encryption/decryption, digital signature, and authentication, you are asked to design a system to support the followings for secure email system:
 - Confidentiality of message;
 - Non repudiation of the sender; and
 - Authentication of message.
- Question:** How do you design such a secure system?



How to develop a system for e-mail security (Cont.)

- **Answer:** You need to carry out the follows:
 - Select the best available cryptographic algorithms as building blocks;
 - Integrate these algorithms into a general-purpose application that is independent of operating system and processor and that is based on a small set of easy-to-use commands.
- This is how PGP and S/MIME were developed.
 - **PGP: Pretty Good Privacy**
 - **S/MIME: Secure/Multipurpose Internet Mail Extension**



PGP: Pretty Good Privacy

- It is a program for email communication security.
- Phil Zimmermann started writing PGP in the mid 1980s and finished the first version in 1991.
- It is available free worldwide in versions, and runs on a variety of platforms, including DOS/Windows, Unix, Macintosh, and many more.
- It is based on cryptography algorithms that have survived extensive public reviews.
- It has a wide range of applicability: within corporations and for individuals within themselves.



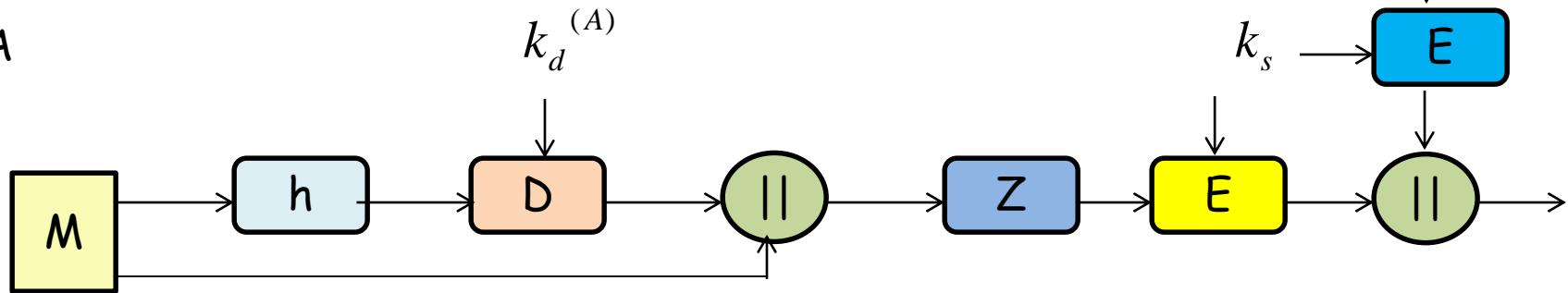
Services of PGP

- Nonrepudiation and authentication (Digital signature using DSS/SHA or RSA/SHA)
- Message confidentiality (symmetric encryption using CAST, IDEA, or 3DES, and session key encryption with ElGamal or RSA)
- Compression (using ZIP) - A message may be compressed, for storage or transmission.
- Email Compatibility (using RADIX-64 conversion) to provide transparency for email applications, an encrypted message may be converted to an ASCII string using RADIX-64 conversion.
- Segmentation - to accommodate maximum message size limitations, PGP performs segmentation and reassembly.

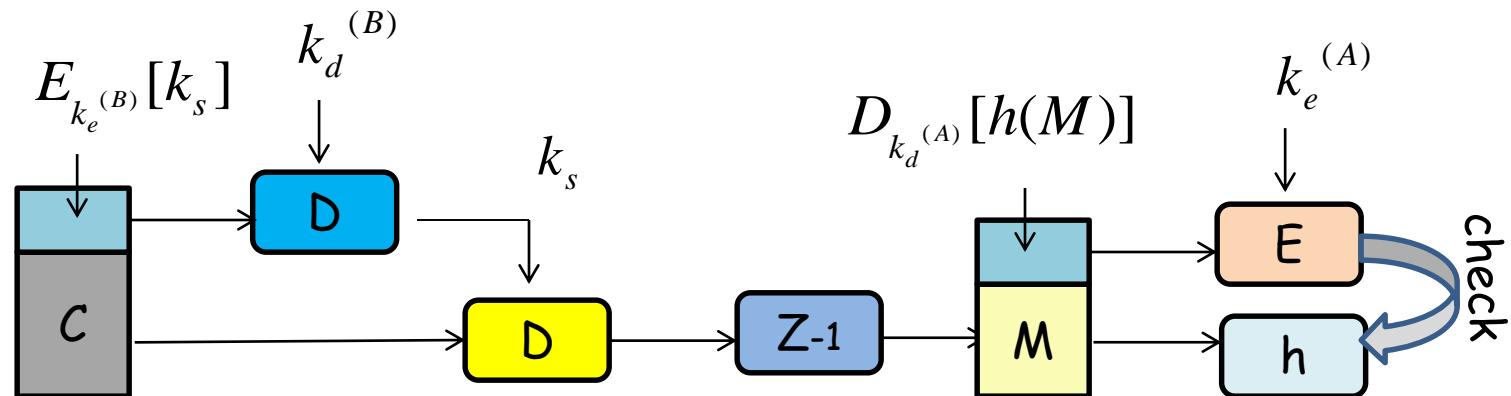


Authentication, Confidentiality, Nonrepudiation in PGP

Source A



Destination B



- DSS/SHA-1 or RSA/SHA-1, Z=ZIP algorithm,
- RSA or ElGamal, CAST or IDEA or 3DES, ks the session key
- In new versions of PGP, SHA-256 and SHA-512 will be used.

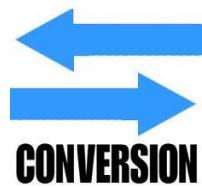
Compression in PGP

- **Why compression?**
 - Save space both for email transmission and for file storage.
- Placement of compression: After applying the signature, but before encryption. **Why should compression before encryption?**
 - Compression reduces the redundancy of messages and makes cryptanalysis more difficult!



Email Compatibility

- **Problem:** When PGP is used, at least part of the block to be transmitted is encrypted, consists of a stream of arbitrary 8-bit octets. However, many electronic mail systems only permit the use of blocks consisting of ASCII text.
- **Solution:** To accommodate this restriction, PGP provides the service of converting the raw 8-bit binary stream to a stream of printable ASCII characters. The scheme used is the "RADIX-64 conversion"
- **Comment:** The use of RADIX-64 conversion expands a message by 33%. Fortunately, the compression should be more than enough to compensate for the RADIX-64 conversion.



RADIX-64

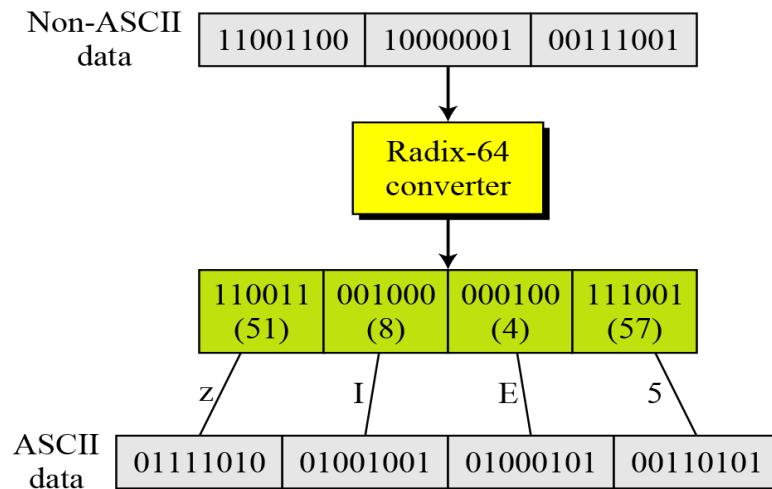
Value	Char	Value	Char	Value	Char	Value	Char
0	A	16	Q	32	g	48	w
1	B	17	R	33	h	49	x
2	C	18	S	34	i	50	y
3	D	19	T	35	j	51	z
4	E	20	U	36	k	52	0
5	F	21	V	37	l	53	1
6	G	22	W	38	m	54	2
7	H	23	X	39	n	55	3
8	I	24	Y	40	o	56	4
9	J	25	Z	41	p	57	5
10	K	26	a	42	q	58	6
11	L	27	b	43	r	59	7
12	M	28	c	44	s	60	8
13	N	29	d	45	t	61	9
14	O	30	e	46	u	62	+
15	P	31	f	47	v	63	/

Text content	M	a	n	
ASCII	77	97	110	
Bit pattern	0 1 0 0 1 1 0 1 0 1 1 0 0 0 0 1 0 1 1 0 1 1 1 1 0			
Index	19	22	5	46
Base64-encoded	T	W	F	u

Content-Transfer-Encoding: base64

```
JVBERi0xLjUNCiW1tbW1DQoxlDAgb2JqDQo8PC9UeXBIL0NhdGFsb2cvUGFnZXMGMiAwIFlvTGFu
Zyhlbi1VUykgL1N0cnVjdFRyZWVSb290IDMxIDAgUi9NYXJrSW5mbzw8L01hcmtlZCB0cnVIPj4+
Pg0KZW5kb2JqDQoylDAgb2JqDQo8PC9UeXBIL1BhZ2VzL0NvdW50IDEvS2IkctsgMyAwIFlJdID4+
DQplbmRvYmoNCjMgMCBvYmoNCjw8L1R5cGUvUGFnZ39QYXJlbmQgMiAwIFlvUmVzb3VyY2VzPDw
Rm9udDw8L0YxIDUgMCBSL0YyIDEyIDAgUi9GMyAxNCAwIFlvRjQgMTYgMCBSL0Y1IDxIDAgUi9G
NiAyMyAwIFlvRjcgMjggMCBSPj4vRXh0R1N0YXRIPDwR1MxMCAwIFlvR1MxMSAxMSAwIFl+
```

The use of RADIX-64 conversion expands a message by 33%.



ASCII-Code

ASCII Hex	Symbol							
0	0	NUL	16	10	DLE	32	20	(space)
1	1	SOH	17	11	DC1	33	21	!
2	2	STX	18	12	DC2	34	22	"
3	3	ETX	19	13	DC3	35	23	#
4	4	EOT	20	14	DC4	36	24	\$
5	5	ENQ	21	15	NAK	37	25	%
6	6	ACK	22	16	SYN	38	26	&
7	7	BEL	23	17	ETB	39	27	'
8	8	BS	24	18	CAN	40	28	(
9	9	TAB	25	19	EM	41	29)
10	A	LF	26	1A	SUB	42	2A	*
11	B	VT	27	1B	ESC	43	2B	+
12	C	FF	28	1C	FS	44	2C	,
13	D	CR	29	1D	GS	45	2D	-
14	E	SO	30	1E	RS	46	2E	.
15	F	SI	31	1F	US	47	2F	/
						63	3F	?

ASCII-Code (Cont.)

ASCII Hex	Symbol										
64	40	@	80	50	P	96	60	'	112	70	p
65	41	A	81	51	Q	97	61	a	113	71	q
66	42	B	82	52	R	98	62	b	114	72	r
67	43	C	83	53	S	99	63	c	115	73	s
68	44	D	84	54	T	100	64	d	116	74	t
69	45	E	85	55	U	101	65	e	117	75	u
70	46	F	86	56	V	102	66	f	118	76	v
71	47	G	87	57	W	103	67	g	119	77	w
72	48	H	88	58	X	104	68	h	120	78	x
73	49	I	89	59	Y	105	69	i	121	79	y
74	4A	J	90	5A	Z	106	6A	j	122	7A	z
75	4B	K	91	5B	[107	6B	k	123	7B	{
76	4C	L	92	5C	\	108	6C	l	124	7C	
77	4D	M	93	5D]	109	6D	m	125	7D	}
78	4E	N	94	5E	^	110	6E	n	126	7E	~
79	4F	O	95	5F	_	111	6F	o	127	7F	ÿ

Segmentation and Reassembly

- **Problem:** Email facilities often are restricted to a maximum message length (e.g., 50,000 octets). Any message longer than that must be broken into smaller segments, each of which is mailed separately.
- **Solution:** To accommodate this restriction, PGP automatically subdivides a message that is too large into segments that are small enough to send via email.
- **When is segmentation done?** After all of the other processing, including the RADIX-64 conversion.
- **Reassembly:** The session key component and signature component appear only once, at the beginning of the first segment. At the receiving end, PGP must strip off all email headers and reassemble the entire original block before performing the steps.

Key Requirements in PGP

- A means of generating unpredictable session keys is needed.
- A user is allowed to have multiple public/private key pairs. (A user may wish to have multiple key pairs at a given time to interact with different groups of correspondents or simply to enhance security by limiting the amount of material encrypted with any one key.) Hence there is not a one-to-one correspondence between users and their public keys.
- Each PGP entity must maintain a file of its own public/private key pairs as well as a file of public keys of correspondents.



Key Identifiers

- **Problem:** Recall that A sends $E_{k_e^{(B)}}[k_s] || E_{k_s}[x]$ to B if encryption is needed. But in the system B could have more than one private/public key pairs. How could B know which of his public key was used by A?
- **Solution 1:** Transmit the public key $k_e^{(B)}$ together with that message. Then B could check that it is indeed one of his public keys.
- **Disadvantages:** But it is a waste of resource, as a public key could have hundreds of digits in length.



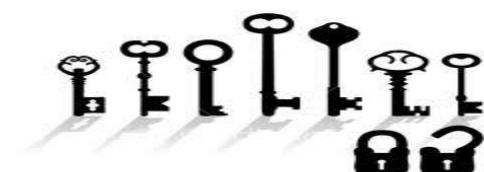
Key Identifiers (Cont.)

- **Problem:** Recall that A sends $E_{k_e^{(B)}}[k_s] || E_{k_s}[x]$ to B if encryption is needed. But in the system B could have more than one private/public key pairs. How could B know which of his public key was used by A?
- **Solution 2:** Associate an **identifier** with each public key that is unique at least within each one user. That is, user ID plus key ID would be sufficient to identify a key uniquely.
- **Disadvantages:** It leads to a management and overhead problem: Key IDs must be assigned and stored so that both sender and recipient could map from key ID to public key.



Key Identifiers (Cont.)

- **Problem:** Recall that A sends $E_{k_e^{(B)}}[k_s] || E_{k_s}[x]$ to B if encryption is needed. But in the system B could have more than one private/public key pairs. How could B know which of his public key was used by A?
- **Solution adopted in PGP:** ID of a public key $k_e^{(B)}$ is defined to be $k_e^{(B)} \bmod 2^{64}$
- **Comments:** Hence with very high probability that the IDs of a user's public keys are unique.
- **Is key ID needed for PGP signature?** Yes. Key ID is also included in the component of PGP signature.



Key Ring

- **Observation:** Two key IDs $\text{ID}(k_{e(A)})$ and $\text{ID}(k_{e(B)})$ are included in any PGP message that provides both confidentiality and authentication.
- **Question:** How to store and organize them in a systematic way for efficient and effective use by all parties?
- **Scheme used in PGP:** It provides a pair of data structure at each node, one to store the public/private key pairs owned by that node and one to store the public keys of other users known at this node. The data structures are referred to, respectively, as the private-key ring and public-key ring.

Privacy Key Ring: 1-Row 1-Key Pair



User ID	Key ID	Public key	Encrypted private key	Timestamp
⋮	⋮	⋮	⋮	⋮

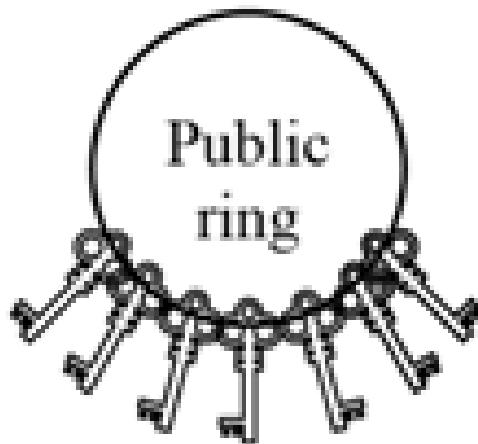
T	key ID*	public key	encrypted private key	user ID*
⋮	⋮	⋮	⋮	⋮
T_i	$k_e^{(i)} \text{ mod } 2^{64}$	$k_e^{(i)}$	$E_{h(P_i)}[k_d^{(i)}]$	user i
⋮	⋮	⋮	⋮	⋮

- * indicates the fields to index the table. The private-key ring can be indexed by either User ID or Key ID.
- The private-key ring is stored only on the machine of the user that created and owns the key pair, and is accessible only to that user.
- The user selects a password P_i , computes $h(P_i) = \text{SHA-1}(P_i)$. The private key is encrypted with Symmetric encryption, using part of $h[P_i]$ as the key. The encrypted private key is stored.

When a user accesses his/her private key, he/she must supply the password. When generating a new public/private key pair, the password is also required.

The security of the system depends on the security of the password!

Public Key Ring: 1-Column 1-Key Pair



timestamp	...	T_i	...
Key ID*	...	$k_e^{(i)} \bmod 2^{64}$...
public key	...	$k_e^{(i)}$...
user ID*	...	user_i	...

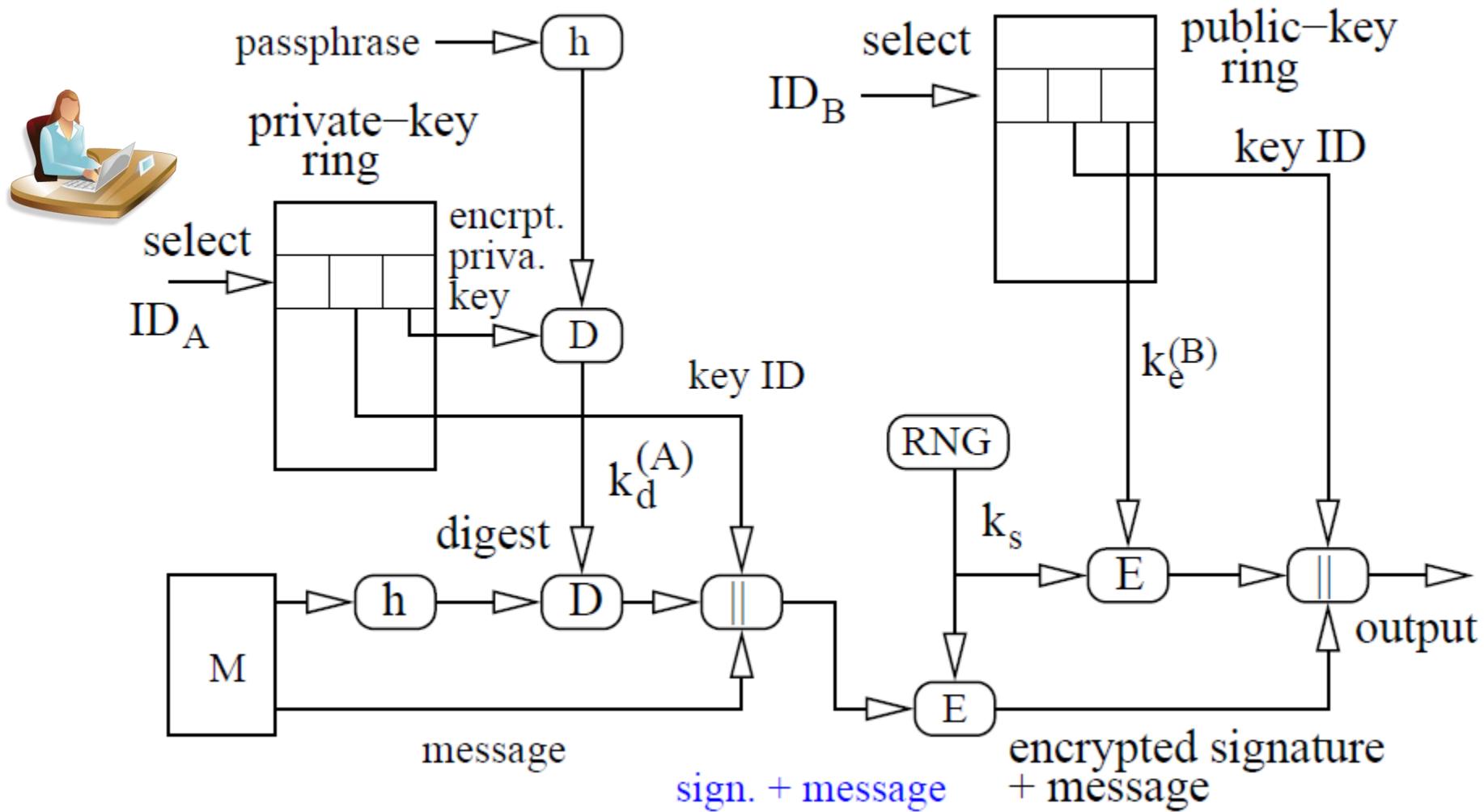
- **Definition:** It is used to store public keys of other users that are known to this user.
- **Remark:** The public-key ring can be indexed by either User ID or Key ID. We will see the need for both means of indexing later.
- **Comment:** PGP is intended for use in a variety of formal and informal environments, no rigid public-key management scheme is set up! One should update and verify the correctness of the information in his/her public key rings.

Key Revocation

- **Why key revocation?** Because compromise is suspected or a user wants to avoid the use of the same key for an extended period.
- **How to do this?** The owner issues a key revocation certificate, signed by the owner. This certificate has the same form as a normal signature certificate, but includes an indicator that purpose of this certificate is to revoke the use of this public key.
- **Remark:** The corresponding private key must be used to sign a certificate that revokes a public key.
- **Comments:** An opponent who has compromised the private key of an owner can also issue such a certificate. However, this would deny the opponent as well as the legitimate owner the use of the public key.

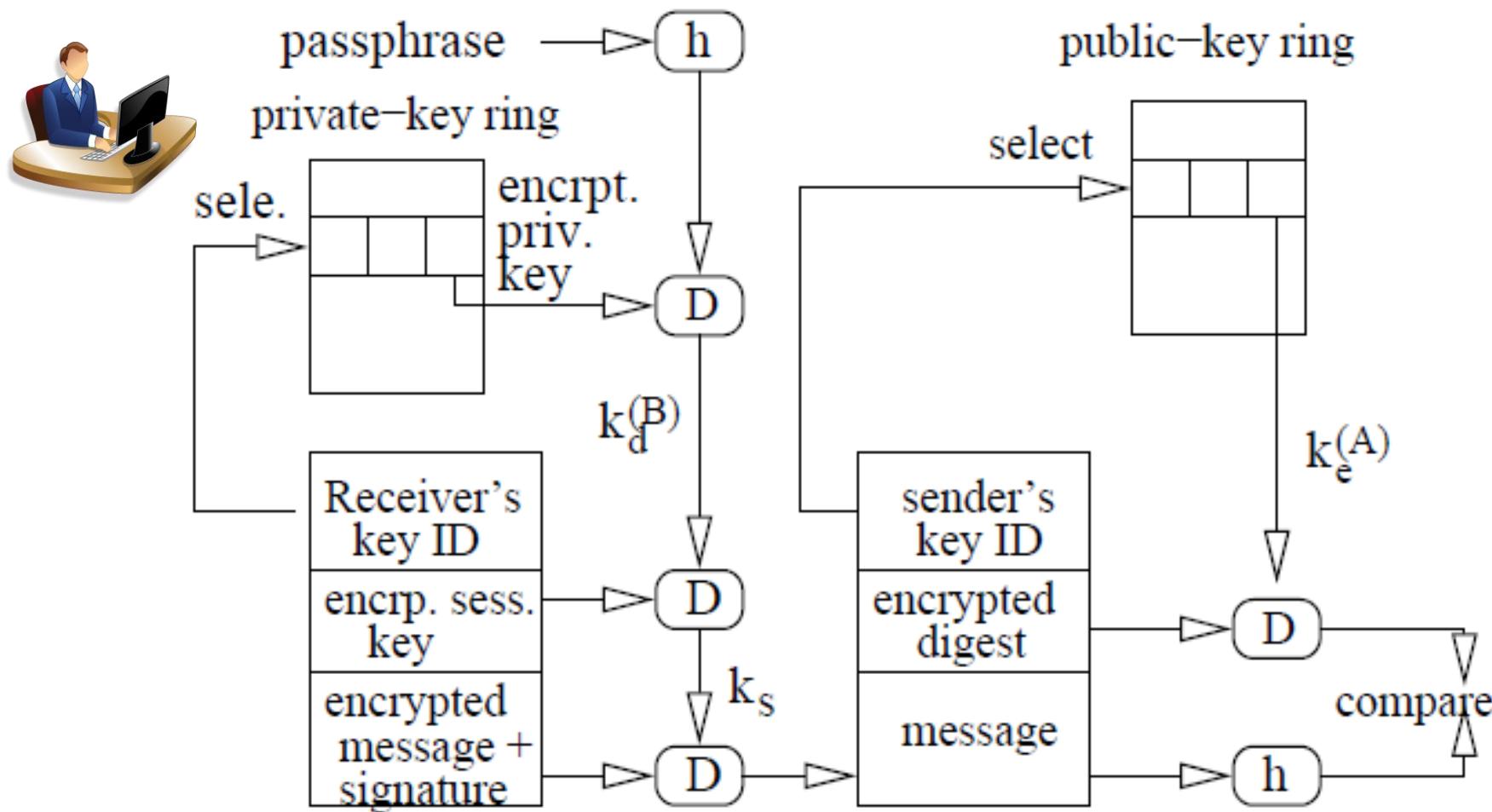
PGP Message Generation

At sender A: ZIP, R64 are omitted



PGP Message Reception

At receiver *B*: ZIP, R64 are omitted



Thank
you

