CISC859: Topics in Advanced Networks & Distributed Computing: Network & Distributed System Security

Review of Basic Security Concepts & Cryptographic Techniques

Background of Information Security

- What is information security?
  - Keeping information secure against stealing & changing & destroying & forging
  - Traditionally provided by physical (e.g., cabinets with locks) and administrative means (e.g., personal screening procedures)
- Information security requirements have dramatically changed in the last several decades
  - Growing computer use requires automated tools to protect files and other stored information
  - Growing use of networks and communications links requires measures to protect data during transmission

Key Definitions

- Computer security
  - the generic name for the collection of tools designed to protect data and to thwart hackers
- Network security
  - measures to protect data during their transmission
- Internet security
  - measures to protect data during their transmission over a collection of interconnected networks
- Note: boundaries among these definitions are blurred

Aim of Course

- Our focus is on **Network & Distributed Systems Security**
- This consists of measures to deter, prevent, detect, and correct security violations that involve the transmission & storage of information

Information Security Objectives

- Confidentiality (*secrecy*)
  - A service used to keep the content of information from all but those authorized to have it
- Data integrity
  - A service which addresses the unauthorized alteration of data
- Authentication
  - Entity authentication: two communicating parties should identify each other
  - Data origin authentication: information sent over a channel should be authenticated as to origin; implicitly provides data integrity
- Non-repudiation
  - A service which prevents an entity from denying previous commitments

What Is Cryptography?

- Cryptography
  - The study of techniques and applications that depend on the existence of difficult mathematical problems
- Cryptanalysis
  - The study of how to compromise (defeat) cryptographic mechanisms
- Cryptology
  - From the Greek *kryptos logos*, meaning “hidden word”
  - The discipline of cryptography and cryptanalysis combined
  - The study of techniques for ensuring the secrecy and/or authenticity of information
- Our focus is not the study of cryptography itself, but its use in solving practical network security problems

Some Critical Concepts (1)

- Encryption
  - The transformation of a message (called *plaintext*) into a form (called *ciphertext*) that is as close to impossible as possible to read without the appropriate knowledge (a key)
  - To ensure privacy by keeping the plaintext hidden from any non-intended person, even those having access to the ciphertext
- Decryption
  - The reverse of encryption
  - The transformation of ciphertext back into intelligible plaintext
- Key
  - The secret information used in encryption & decryption
  - The same key or different keys may be used

Some Critical Concepts (2)

- Digital signature
  - A piece of information used to prove that a message was generated by a particular individual of a particular key
  - Signature generation and verification use different keys
- Message authentication code (MAC)
  - An authentication tag (also called a *checksum*) derived by applying an authentication scheme, together with a secret key, to a message to be authenticated
  - MAC generation and verification use the same key
- Computationally hard problems
  - Cryptography is fundamentally based on problems that are difficult to solve in terms of computational requirements
  - E.g., Factoring, Discrete Logarithm, Traveling Salesman, Integer Programming, Graph Coloring, Hamiltonian Path

Example: Substitution Ciphers

Substitution Cipher: Map each letter or numeral into another letter or numeral:

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

z y x w v u t s r q p o n m l k j i h g f e d c b a

- Example:
  - hvxfirgb→ security
- Substitution ciphers are easy to break
  - Take histogram of frequency of occurrence of letters in a ciphertext message
  - Match to known frequencies of letters

Example: Transposition Cipher

Transposition Cipher: Rearrange order of letters/numerals in a message using a particular rearrangement:

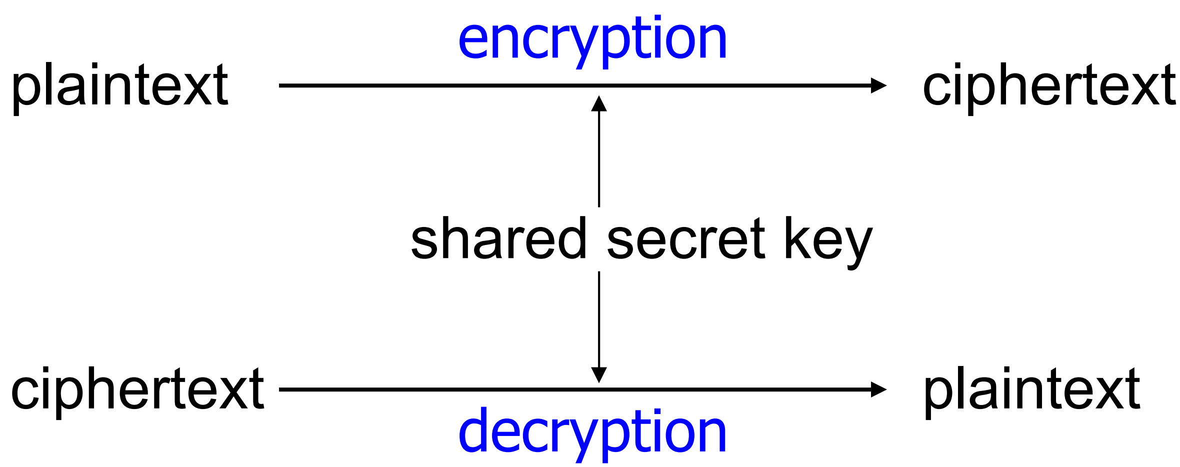
  - interchange character k with character k+1

- Example:
  - security→ esuciryt
- Transposition Ciphers are easy to break
  - Suppose plaintext and ciphertext are known; matching of letters in plaintext and ciphertext will reveal transposition mapping
  - Using anagram analysis: sliding pieces of ciphertext around, then looking for sections that look like anagrams of English words, and solving the anagrams

Essential Crypto Techniques

- Secret-key cryptography
- Public-key cryptography
- Hash functions
- Merkel Hash Tree
- Secret Sharing
- Information Dispersal
- Identity-based encryption
- Attribute-based encryption
- Homomorphic encryption
- Blind signature
- Private set intersection

Secret-Key Cryptography



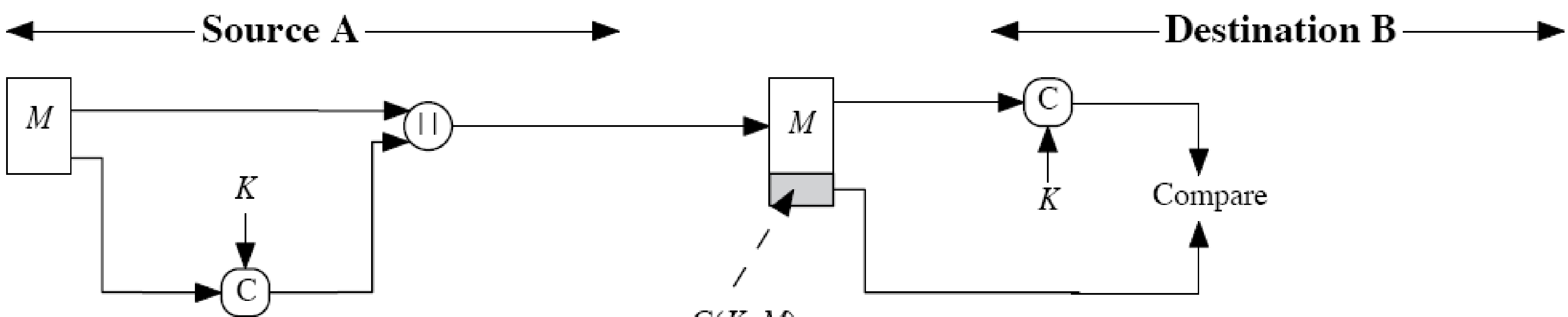
- The sender and receiver share a key before communicating
- The shared key is used in both encryption and decryption
- Also known as symmetric cryptography
- E.g., RC4, RC5, RC6, DES, 3DES, AES

Security Uses of Secret-Key Crypto (1)

- Transmitting over an insecure channel
  - Guaranteeing message confidentiality
- Secure storage on insecure media
  - Guaranteeing information confidentiality
- Authentication
  - Alice and Bob share a secret key K_{AB}
  - Challenge-response authentication with the shared secret

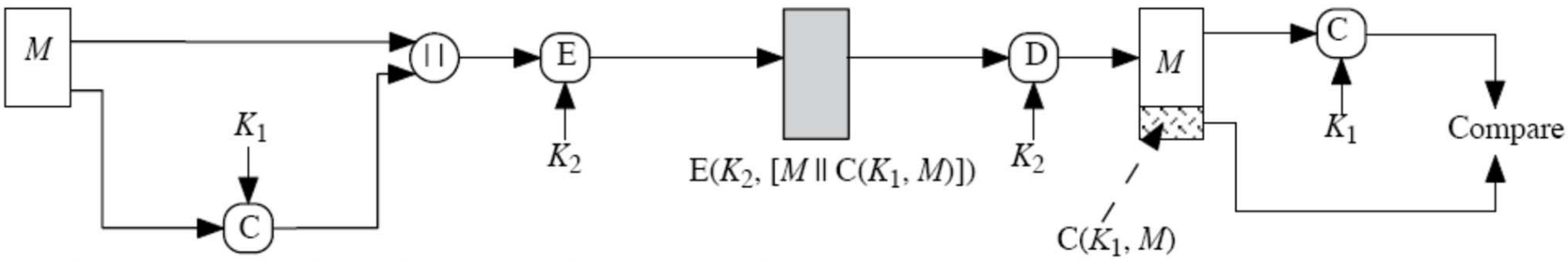
Security Uses of Secret-Key Crypto (2)

- Message Authentication Codes (MACs)
  - Source A and destination B shares a secret key *K*
  - *C* denotes a suitable MAC function (examples given later)
  - *E*/*D* denotes a suitable symmetric encryption/decryption algorithm

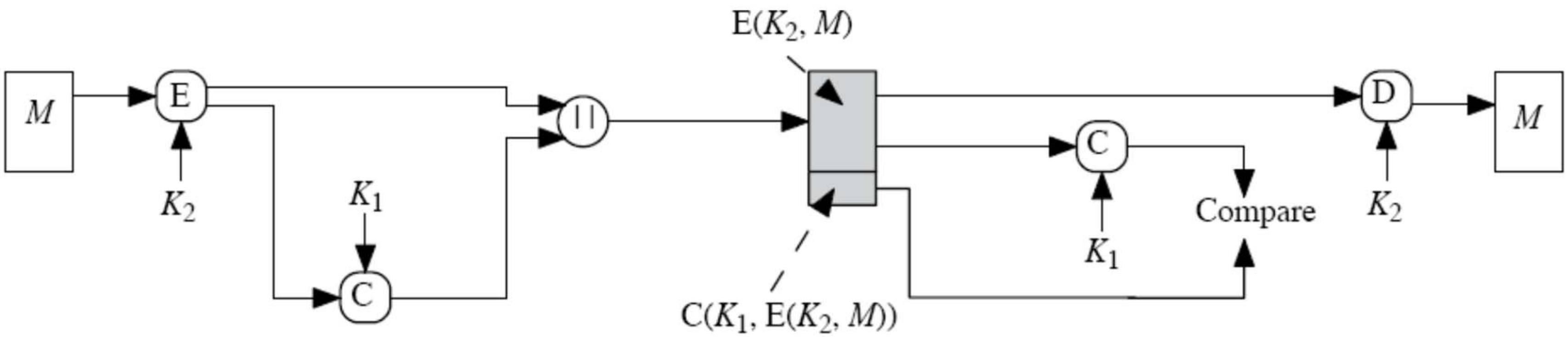


(a) Message authentication

Security Uses of Secret-Key Crypto (3)



(b) Message authentication and confidentiality; authentication tied to plaintext



(c) Message authentication and confidentiality; authentication tied to ciphertext

Public-Key Cryptography (1)

- Each user generates a unique pair of keys
  - A private key (K^{-1}), kept confidential to himself/herself
  - A public key (K), preferably known to the entire world
  - There is a one-to-one correspondence between K & K^{-1}
  - It is computationally infeasible to determine K^{-1} given K
- Each user places its public key in a public register or accessible file, while keeping its private key confidential
- Each user maintains a collection of public keys obtained from others
- If Bob wishes to send a confidential message to Alice, Bob encrypts the message using Alice's public key
- When Alice receives the message, she decrypts it using her private key. No other recipient can decrypt the message because only Alice knows Alice's private key
- Also known as asymmetric cryptography
  - E.g., RSA, DSA, Elliptic Curve Cryptography, Diffie-Hellman

RSA Public Key Algorithm

- Named after Rivest, Shamir, and Adleman
- Modular arithmetic & factorization of large numbers
  - Let $n = pq$, where p & q are two large numbers
    - ➔ n typically several hundred bits long, i.e. 512 bits
    - ➔ Plaintext must be shorter than n
  - Find e relatively prime to $(p – 1)(q – 1)$
    - ➔ i.e. e has no common factors with $(p – 1)(q – 1)$
    - ➔ **Public key** is $\{e,n\}$
  - Let d be multiplicative inverse of e
    - ➔ $de = 1$ modulo $(p – 1)(q – 1)$
    - ➔ **Private key** is $\{d,n\}$

Encryption & Decryption

- Fact: For $P<n$ and n, p, q, d as above:

$$P^{de} \bmod n = P \bmod n$$

- Encryption:

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

  - Result is number less than n and is represented by same number of bits as key

- Decryption:

$$C^d \bmod n = P^{ed} \bmod n = P \bmod n = P$$

- Security stems from fact that it is very difficult to factor large numbers n, and with e to then determine d

RSA Example

- Let $p = 5$, $q = 11$
  - $n = pq = 55$ and $(p - 1)(q - 1) = 40$
- Let $e = 7$, which is relatively prime to 40
  - $7d \bmod 40 = 1$, gives $d = 23$
- Public key is {7, 55}
- Private key is {23, 55}

RSA Example continued

- Encrypt "RSA": R=18, S=19, A=1

  $C_1 = 18^7 \bmod 55 = 18^{4+2+1} \bmod 55$

  $= (18 \bmod 55)\ (18^2 \bmod 55)\ (18^4 \bmod 55) \bmod 55$

  $= (18)\ (324 \bmod 55)\ (18^4 \bmod 55) \bmod 55$

  $= (18)\ (49)\ (49^2 \bmod 55) \bmod 55 = (18)(49)(36) \bmod 55$

  $= 31752 \bmod 55 = 17$

  $C_2 = 19^7 \bmod 55 = 24$

  $C_3 = 1^7 \bmod 55 = 1$

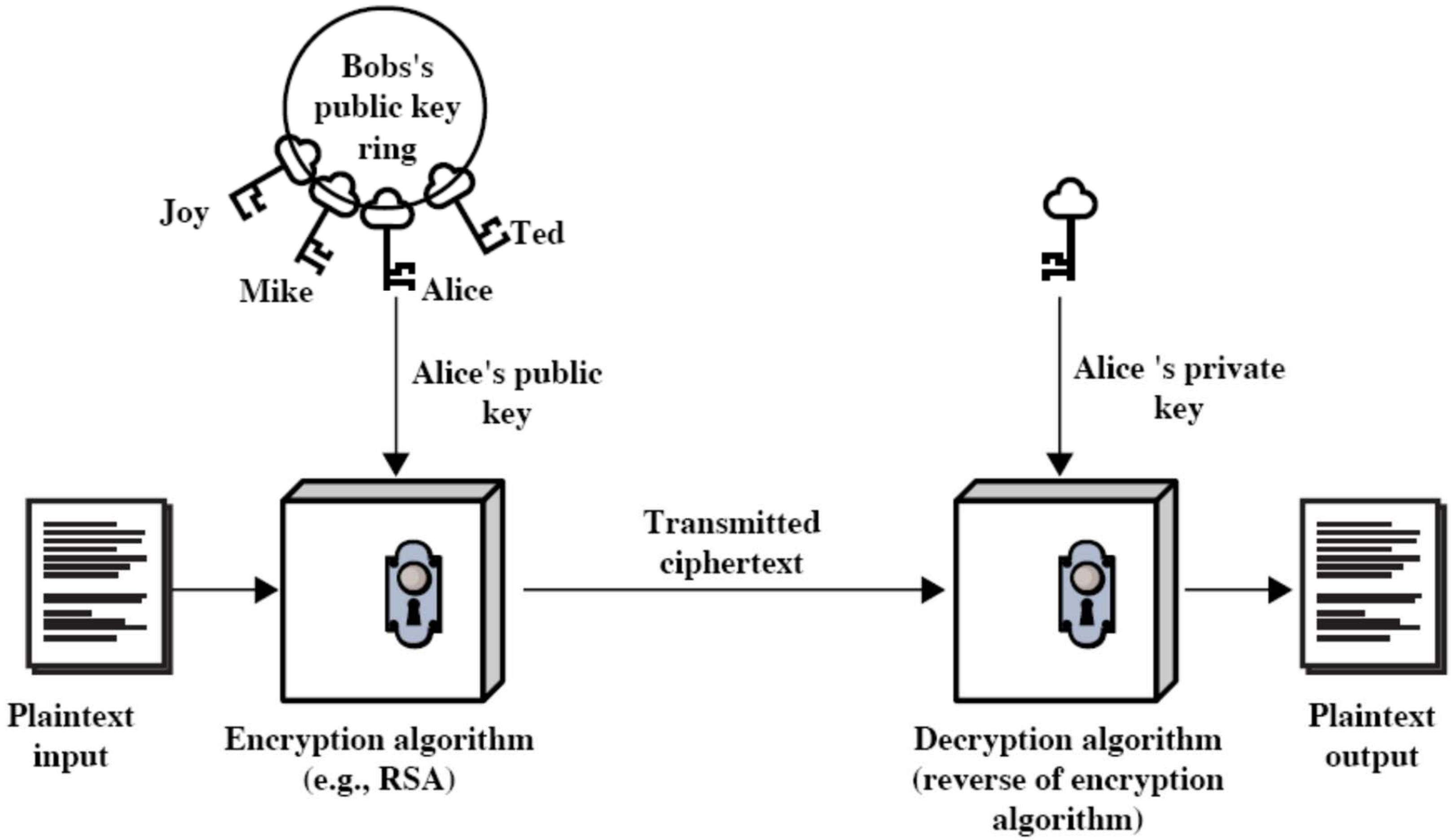
- Decrypt

  $17^{23} \bmod 55 = 17^{16+4+2+1} \bmod 55 = 18$

  $24^{23} \bmod 55 = 19$

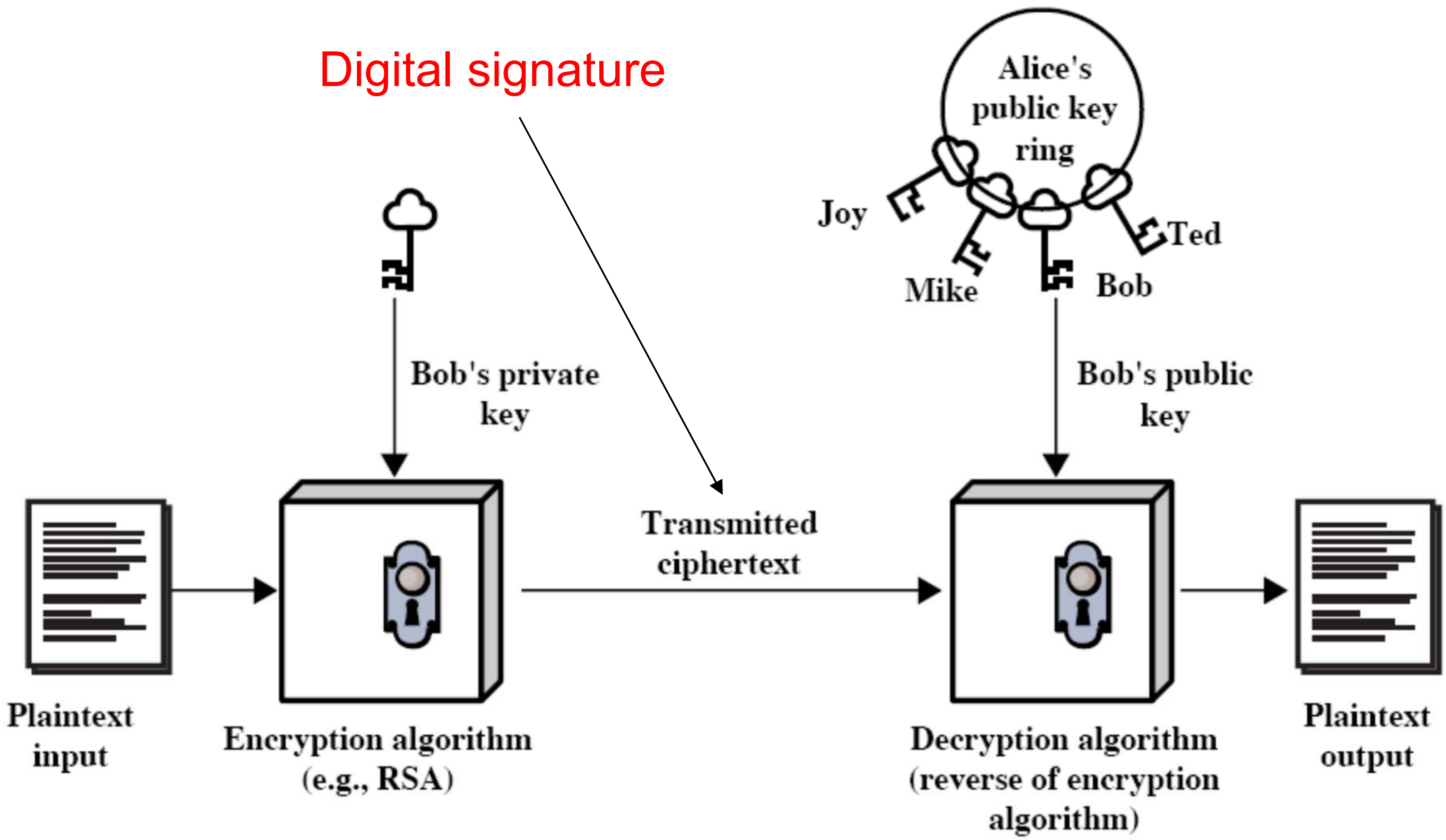
  $1^{23} \bmod 55 = 1$

Security Uses of Public-Key Crypto (1)



(a) Encryption

Security Uses of Public-Key Crypto (2)



(b) Authentication

Need for Authentication of Public keys

- Suppose Alice wants to find Bob's public key. How?
  - Call him up and ask him to send his public key via email
  - Request it via email
  - Retrieve it from some public-key repository
  - …
- An attacker could intercept the transmission and replace Bob's key with his or her own
  - Able to intercept and decrypt messages that are sent from Alice to Bob and encrypted using the fake public key
- Alice needs a measure to authenticate Bob's public key

Public-Key Certificates

- What are they?
  - Digital documents attesting to the binding of a public key to an individual or other entity
  - Allow verification of the claim that a specific public key does in fact belong to a specific individual
  - Help prevent someone from using a phony public key to impersonate someone else
- What are in a public-key certificate?
  - A public key and a name
  - An expiration date
  - The name of the Certificate Authority (CA) issuing the certificate
  - The digital signature of the CA on all the other fields, which can be verified by anyone who trusts the CA and knows its public key

Secret-Key vs. Public-key (1)

- Pros of secret-key cryptography
  - Very fast computation speed
  - Shorter key sizes
  - An extensive history against cryptanalysis
- Cons of secret-key cryptography
  - An efficient and secure method is required to establish a shared secret key between two parties intending to communicate
  - The secret key must be kept secret at both parties
  - How to establish and update pairwise secret keys in a large network is challenging, e.g., $N(N-1)/2$ in a network with N users
  - No support for digital signatures because the secret key is known to both parties

Secret-Key vs. Public-key (2)

- Pros of public-key cryptography
  - Key management is very simple because each user just need maintain his or her public/private key pair
  - Efficient support for digital signatures
- Cons of public-key cryptography
  - Relatively slow computation speed, normally several orders of magnitude than secret-key techniques
  - Larger private-key sizes (a factor of 10 or more than secret keys)
  - No public-key scheme has proven to be secure
  - Doesn't have as extensive a history as secret-key crypto, being discovered only in the mid 1970s
- Common practice
  - Using public-key techniques to establish a shared secret key for subsequent use by secret-key techniques

Hash Functions

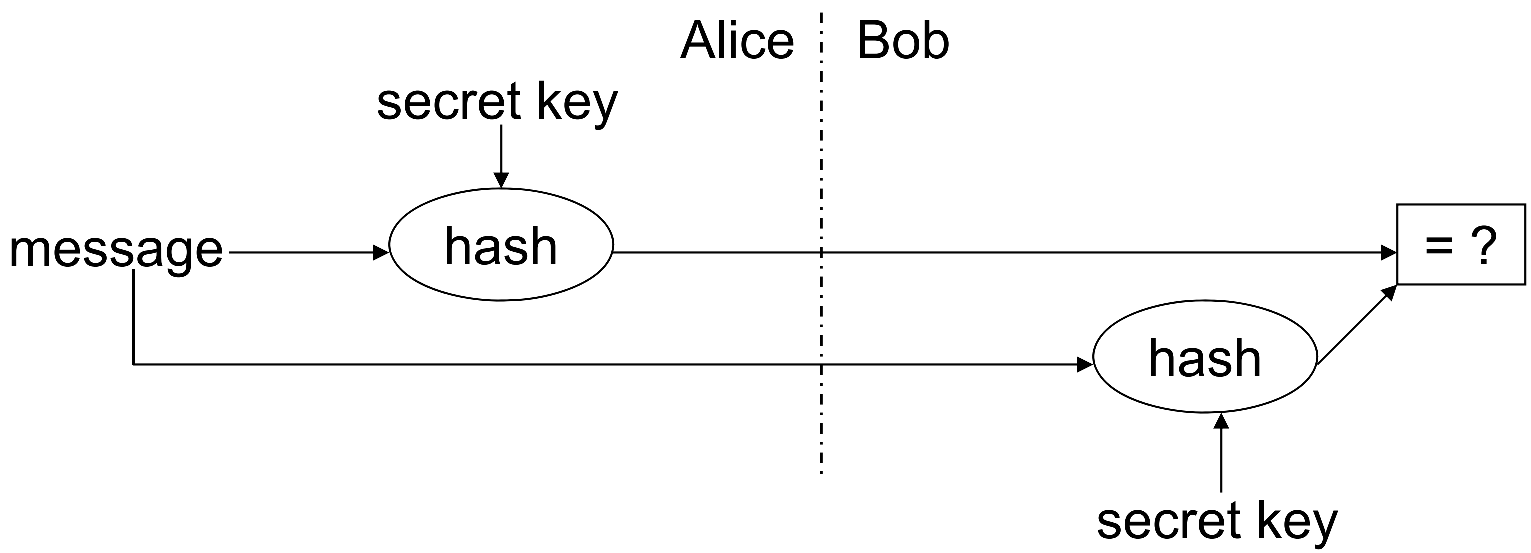
- A *hash function* **H** is a transformation that takes an input *x* and returns a fixed-size string ***h***, which is called a hash value or message digest, i.e., ***h*** = **H**(*x*)
- Basic requirements for a cryptographic hash function
  - The input can be of any length
  - The output has a fixed length
  - **H**(*x*) is relatively easy to compute for any given *m*
  - **H** is one-way (pre-image resistance): for any given ***h*,** it is computationally infeasible to find *x* such that **H**(*x*) = ***h***
  - **H** has weak collision resistance (second pre-image resistance): for any given *x*, it is computationally infeasible to find $y \neq x$ such that **H**(*y*) = **H**(*x*)
  - **H** has strong collision resistance: it is computationally infeasible to find any pair (*x*, *y*) such that **H**(*x*) = **H**(*y*)

Security Uses of Hash Functions (1)

- Password hashing
  - A server stores hashes of user passwords so that anyone with access to the system storage cannot steal the passwords
  - On input of your password, the server computes the hash and compares it with the stored one
- Message fingerprint
  - You may want to know whether some large data structure (e.g., a program) has been modified from one day to the next
  - You can keep a copy of the data on some tamper-proof backing store and periodically compare it to the active version
  - You can save storage with a hash function: simply saving the hash value of the data on the tamper-proof backing store

Security Uses of Hash Functions (2)

- Digital signature efficiency
  - Digital signature operations are expensive, closely related to the message size
  - Generates a hash value of the long message to be digitally signed
  - Produces a digital signature of the shorter hash value
- Message Authentication Codes (MACs)
  - Alice and Bob shares a secret key



Cryptanalysis of Hash Functions (1)

- What is the implication of arbitrary-length inputs and fixed-length outputs?
  - Lots of messages will yield the same hash value
  - For 1000-bit messages and a 128-bit hash value, there on the average 2^{872} messages that hash to any particular hash value
  - But “lots” is so many that it is essentially impossible
- How long should a hash value be?
  - Assume a good *m*-bit hash function
  - It would take trying approximately 2^m possible messages before one would find a message that hashed to a particular hash value
  - It would take trying approximately $2^{m/2}$ messages before finding two messages that have the same hash value (google *The Birthday Problem*)

Example

- M = 1000, m = 128
- Number of possible messages: 2^{1000}
- Number of possible hashes: 2^{128}
- For each hash value there are $2^{1000}/2^{128} = 2^{872}$ messages that generate the hash
- A randomly selected message produces a desired hash value with probability 2^{-128}
- If each attempt requires 1 microsecond, time to find matching message to a hash is:

  2^{128}x1 microsecond = 2^{25} years

Cryptanalysis of Hash Functions (2)

- SHA-0 (Secure Hash Algorithm): 160-bit outputs
  - Ideally it takes 2^{80} attempts to find a collision
  - 1998, 2^{61} attempts by Chabaud and Joux
  - 2004, 2^{51} attempts by Joux, et al.
  - 2004, 2^{40} attempts by Xiaoyun Wang, et al.
  - 2005, 2^{39} attempts by Xiaoyun Wang, et al.
- SHA-1: 160-bits outputs
  - Feb. 2005, 2^{69} attempts by Xiaoyun Wang, et al.
  - Aug. 2005, 2^{63} attempts by Xiaoyun Wang, et al.
- Implications
  - These attacks on SHA-1 don't necessarily mean that they can be practically exploited, but might pave the way to more efficient ones
  - NIST has planned to phase out the use of SHA-1 by 2010

Efficient Authenticators

- One-way chains
- Chained hashes
- Merkle hash trees

Recall One-Way Hash Chains?

- Versatile cryptographic primitive
- Construction
  - Pick random r_N and public one-way function F
  - $r_i = F(r_{i+1})$
  - Secret value: r_N , public value r_0

$$r_3 \xleftarrow{F} r_4 \xleftarrow{F} r_5 \xleftarrow{F} r_6 \xleftarrow{F} r_7$$

- Properties
  - Use in reverse order of construction: r_1 , r_2 … r_N
  - Infeasible to derive r_i from r_j $(j<i)$
  - Efficiently authenticate r_i knowing r_j $(j<i)$: verify $r_j = F^{i-j}(r_i)$
  - Robust to missing values

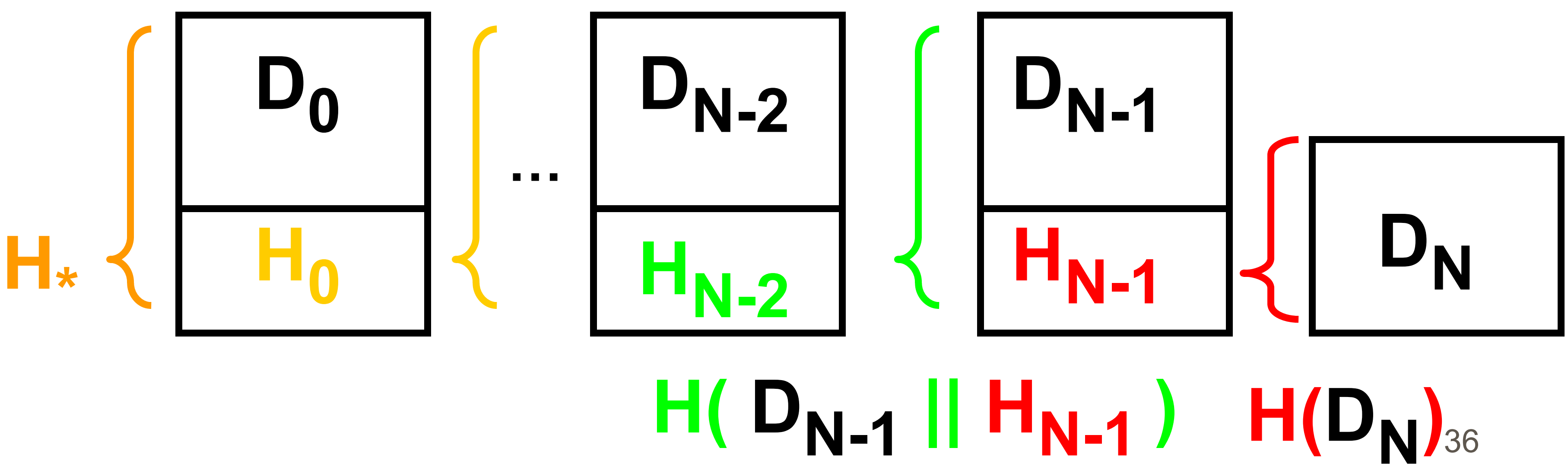
One-Way Chain Application

- S/Key one-time password system
- Goal
  - Use a different password at every login
  - Server cannot derive password for next login
- Solution: one-way chain
  - Pick random password P_L
  - Prepare sequence of passwords $P_i = F(P_{i+1})$
  - Use passwords $P_0, P_1, \ldots, P_{L-1}, P_L$
  - Server can easily authenticate user

$$p_3 \xleftarrow{F} p_4 \xleftarrow{F} p_5 \xleftarrow{F} p_6 \xleftarrow{F} p_7$$

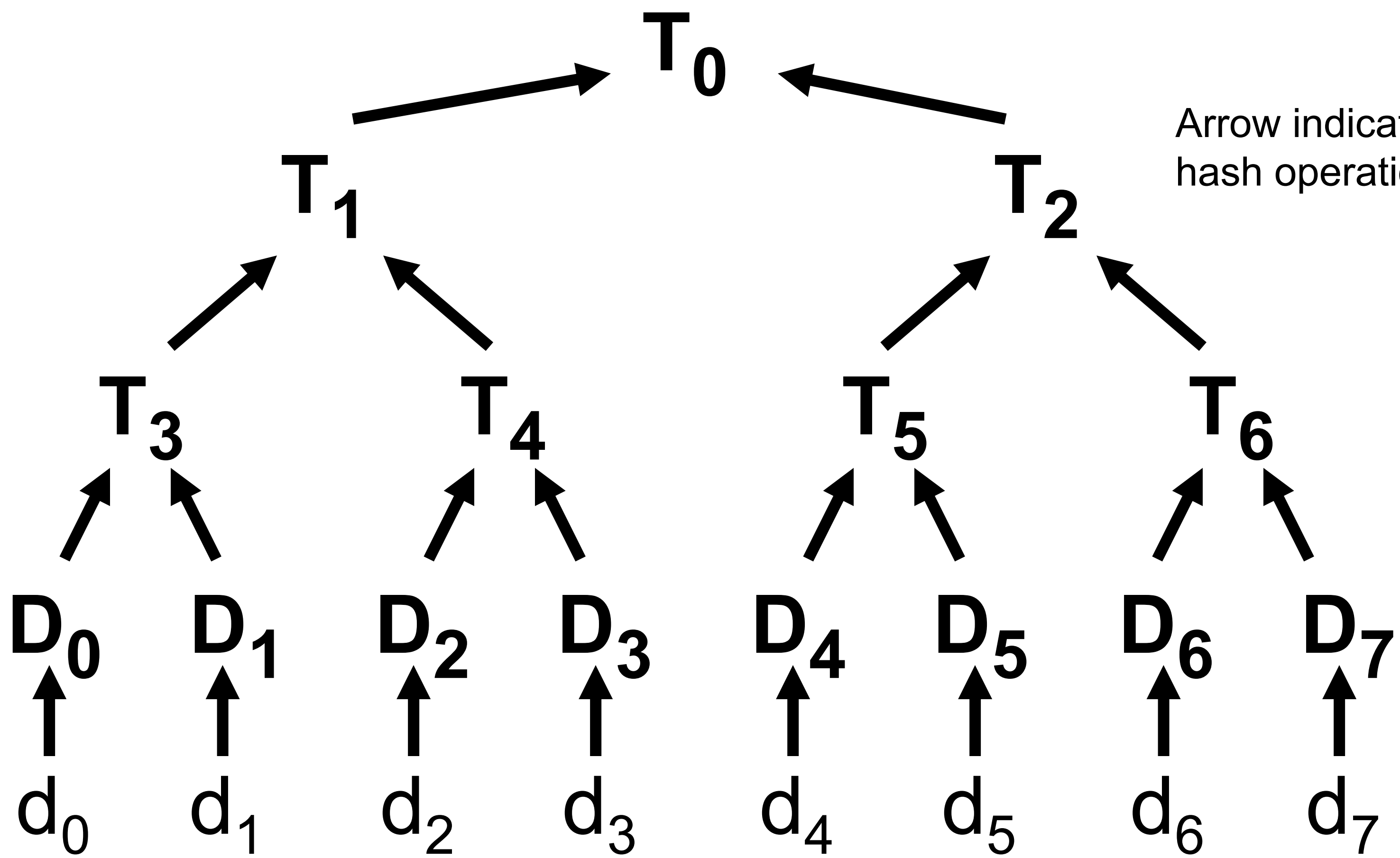
Chained Hashes

- Useful for authenticating a sequence of data values D_0 , D_1 , …, D_N
- H_* authenticates entire chain



Merkle Hash Trees

- Authenticate a sequence of data values d_0 , d_1 , …, d_N
- Construct binary tree over data values



Arrow indicate a hash operation H( )

Merkle Hash Trees II

- Verifier knows T_0
- How can verifier authenticate leaf d_i ?
- Solution: recompute T_0 using d_i
- Example authenticate d_2 , send D_3 T_3 T_2
- Verify $T_0 = H(H(T_3 || H(H(d_2) || D_3)) || T_2)$