CISC 7310X

C13c System Security: Cryptography and Authentication

Hui Chen

Department of Computer & Information Science

CUNY Brooklyn College

Acknowledgement

- These slides are a revision of the slides provided by the authors of the textbook via the publisher of the textbook

Outline

- Cryptography as a Security Tool
- User Authentication
- Implementing Security Defenses

Encryption

- Constrains the set of possible receivers of a message
- **Encryption** algorithm consists of
  - Set K of keys
  - Set M of Messages
  - Set C of ciphertexts (encrypted messages)
  - A function $E : K \rightarrow (M \rightarrow C)$. That is, for each $k \in K$, E_k is a function for generating ciphertexts from messages
    - Both E and E_k for any k should be efficiently computable functions
  - A function $D : K \rightarrow (C \rightarrow M)$. That is, for each $k \in K$, D_k is a function for generating messages from ciphertexts
    - Both D and D_k for any k should be efficiently computable functions

Essential Property of Encryption Algorithms

- Given a ciphertext $c \in C$, a computer can compute m such that $E_k(m) = c$ only if it possesses *k*
  - Thus, a computer holding *k* can decrypt ciphertexts to the plaintexts used to produce them, but a computer not holding *k* cannot decrypt ciphertexts
  - Since ciphertexts are generally exposed (for example, sent on the network), it is important that it be infeasible to derive *k* from the ciphertexts

Types of Encryption Algorithms

- Symmetric Encryption (or secret key encryption)
- Asymmetric encryption (or public-key encryption)

Symmetric Encryption

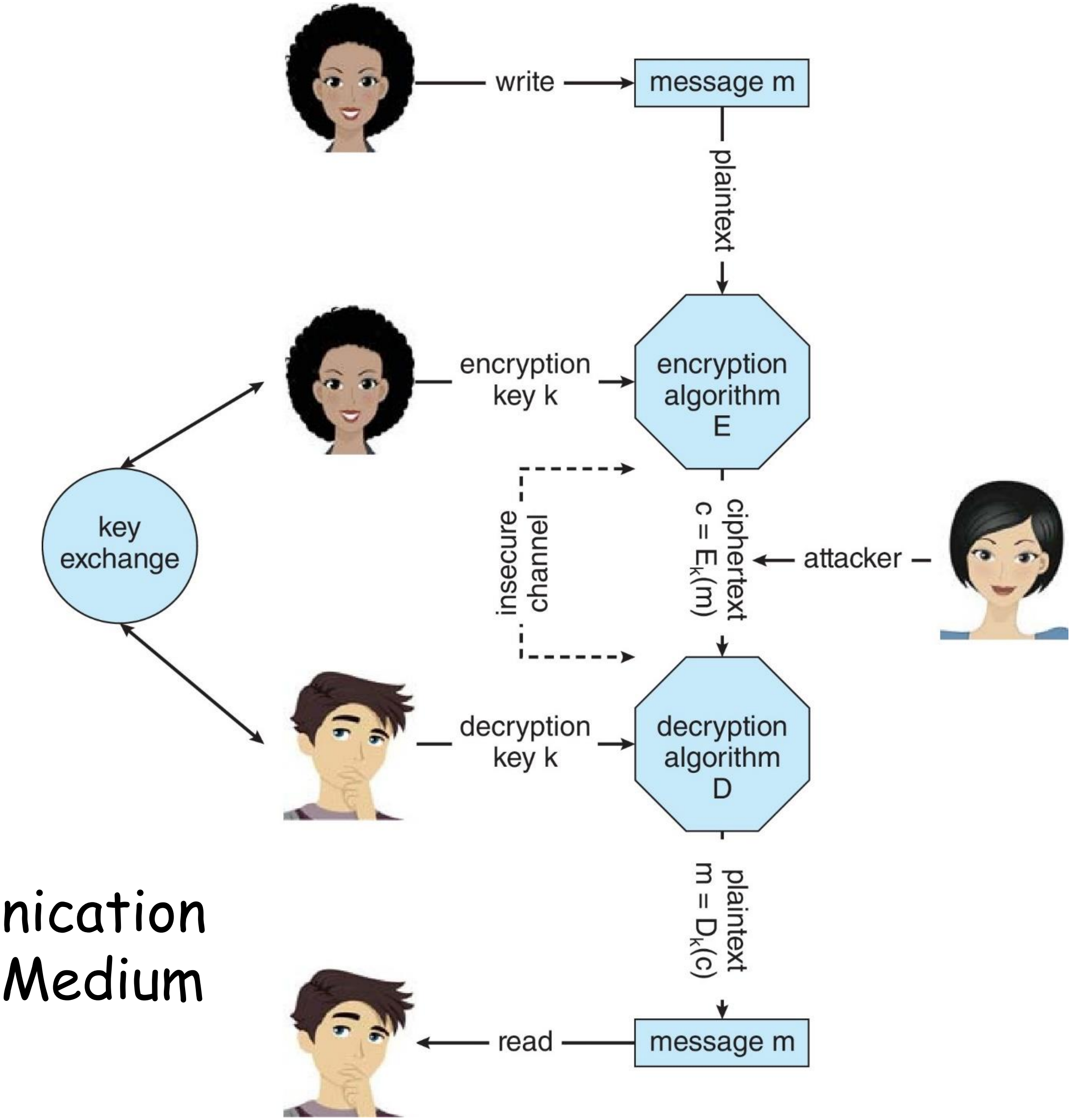
- Same key used to encrypt and decrypt
  - Therefore the key k must be kept secret
  - Also called secret key encryption

Example Symmetric Encryption

- DES was most commonly used symmetric block-encryption algorithm (created by US Govt)
  - Encrypts a block of data at a time
  - Keys too short so now considered insecure
- Triple-DES considered more secure
  - Algorithm used 3 times using 2 or 3 keys

$$c = E_{k3}(D_{k2}(E_{k1}(m)))$$

- 2001 NIST adopted new block cipher - Advanced Encryption Standard (**AES**)
  - Keys of 128, 192, or 256 bits, works on 128 bit blocks
- RC4 is most common symmetric stream cipher, but known to have vulnerabilities
  - Encrypts/decrypts a stream of bytes (i.e., wireless transmission)
  - Key is a input to pseudo-random-bit generator
    - Generates an infinite **keystream**



Secure Communication over Insecure Medium

Asymmetric Encryption

- **Public-key encryption** based on each user having two keys:
  - **public key** – published key used to encrypt data
  - **private key** – key known only to individual user used to decrypt data
- Must be an encryption scheme that can be made public without making it easy to figure out the decryption scheme

Example Asymmetric Encryption: RSA

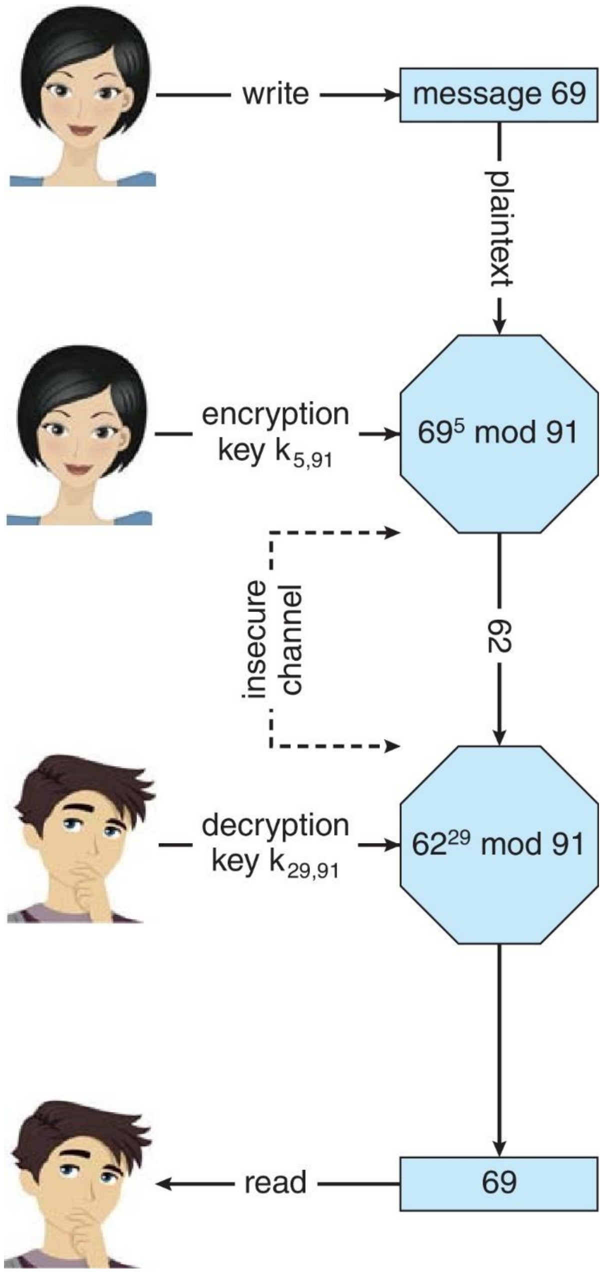
- Most common asymmetric encryption algorithm is RSA
- Based on two factors
    - Efficient algorithms exists for testing whether or not a number is prime
    - No efficient algorithm is know for finding the prime factors of a number
- RSA block cipher

Example Asymmetric Encryption: RSA

- k_e is the **public key,** k_d is the **private key**
- $N = p\ q$ *where* p and q are wo large, randomly chosen prime numbers
  - e.g., p and q are 2048 bits each
- It must be computationally infeasible to derive k_d from k_e, so that k_e need not be kept secret and can be widely disseminated
- Encryption algorithm is $E_{ke,N}(m) = m^{k_e}$ mod N, where k_e satisfies $k_e k_d$ mod $(p-1)(q-1) = 1$
- The decryption algorithm is then $D_{kd,N}(c) = c^{k_d}$ mod N

RSA: Working Example

- For example. make p = 7 and q = 13
- We then calculate $N = 7 * 13 = 91$ and $(p-1)(q-1) = 72$
- We next select k_e relatively prime to 72 and < 72, yielding 5
- Finally, we calculate k_d such that $k_e k_d \bmod 72 = 1$, yielding 29
- We how have our keys
  - Public key, $k_{e,N}$ = 5, 91
  - Private key, $k_{d,N}$ = 29, 91
- Encrypting the message 69 with the public key results in the cyphertext 62
- Cyphertext can be decoded with the private key
  - Public key can be distributed in cleartext to anyone who wants to communicate with holder of public key


write
message 69
plaintext
encryption key $k_{5,91}$
$69^5 \bmod 91$
insecure channel
62
decryption key $k_{29,91}$
$62^{29} \bmod 91$
read
69

Comparison: Symmetric and Asymmetric Encryption

- In addition to the properties of the keys,
  - Note symmetric cryptography based on transformations, asymmetric based on mathematical functions
  - Asymmetric much more compute intensive, and typically not used for bulk data encryption

Questions?

- Cryptography, encryption & decryption
- Concept of symmetric cryptography
- Concept of asymmetric cryptography
- Comparison between symmetric and asymmetric cryptography

Applications of Cryptography

- Authentication
    - Hash function
    - Message authentication code
    - Digital signature
- Digital certificates

Authentication

- Constraining set of potential senders of a message
  - Complementary to encryption
  - Also can prove message unmodified

Authentication Algorithm: Setup

- A set K of keys
- A set M of messages
- A set A of authenticators
- A function $S : K \rightarrow (M \rightarrow A)$
  - That is, for each $k \in K$, S_k is a function for generating authenticators from messages
  - Both S and S_k for any k should be efficiently computable functions
- A function $V : K \rightarrow (M \times A \rightarrow \{\text{true, false}\})$. That is, for each $k \in K$, V_k is a function for verifying authenticators on messages
  - Both V and V_k for any k should be efficiently computable functions

Authentication Algorithm

- For a message *m*, a computer can generate an authenticator *a* $\in$ *A* such that $V_k(m, a)$ = `true` only if it possesses *k*
- Thus, computer holding *k* can generate authenticators on messages so that any other computer possessing *k* can verify them
- Computer not holding *k* cannot generate authenticators on messages that can be verified using V_k
- Since authenticators are generally exposed (for example, they are sent on the network with the messages themselves), it must not be feasible to derive *k* from the authenticators
- Practically, if $V_k(m,a)$ = **`true`** then we know *m* has not been modified and that send of message has *k*
  - If we share *k* with only one entity, know where the message originated

Authentication – Hash Functions

- Basis of authentication
- Creates small, fixed-size block of data **message digest** (**hash value)** from *m*
- Hash Function *H* must be collision resistant on *m*
  - Must be infeasible to find an *m'* ≠ *m* such that *H*(*m*) = *H*(*m'*)
- If *H*(*m*) = *H*(*m'*), then *m* = *m'*
  - The message has not been modified
- Common message-digest functions include **MD5**, which produces a 128-bit hash, and **SHA-1**, which outputs a 160-bit hash
- Not useful as authenticators
  - For example *H(m*) can be sent with a message
    - But if *H* is known someone could modify *m* to *m'* and recompute *H(m')* and modification not detected
    - So must authenticate *H(m)*

Authentication - MAC

- Symmetric encryption used in **message-authentication code** (**MAC**) authentication algorithm
- Cryptographic checksum generated from message using secret key
  - Can securely authenticate short values
- If used to authenticate *H(m)* for an *H* that is collision resistant, then obtain a way to securely authenticate long message by hashing them first
- Note that *k* is needed to compute both S_k and V_k, so anyone able to compute one can compute the other

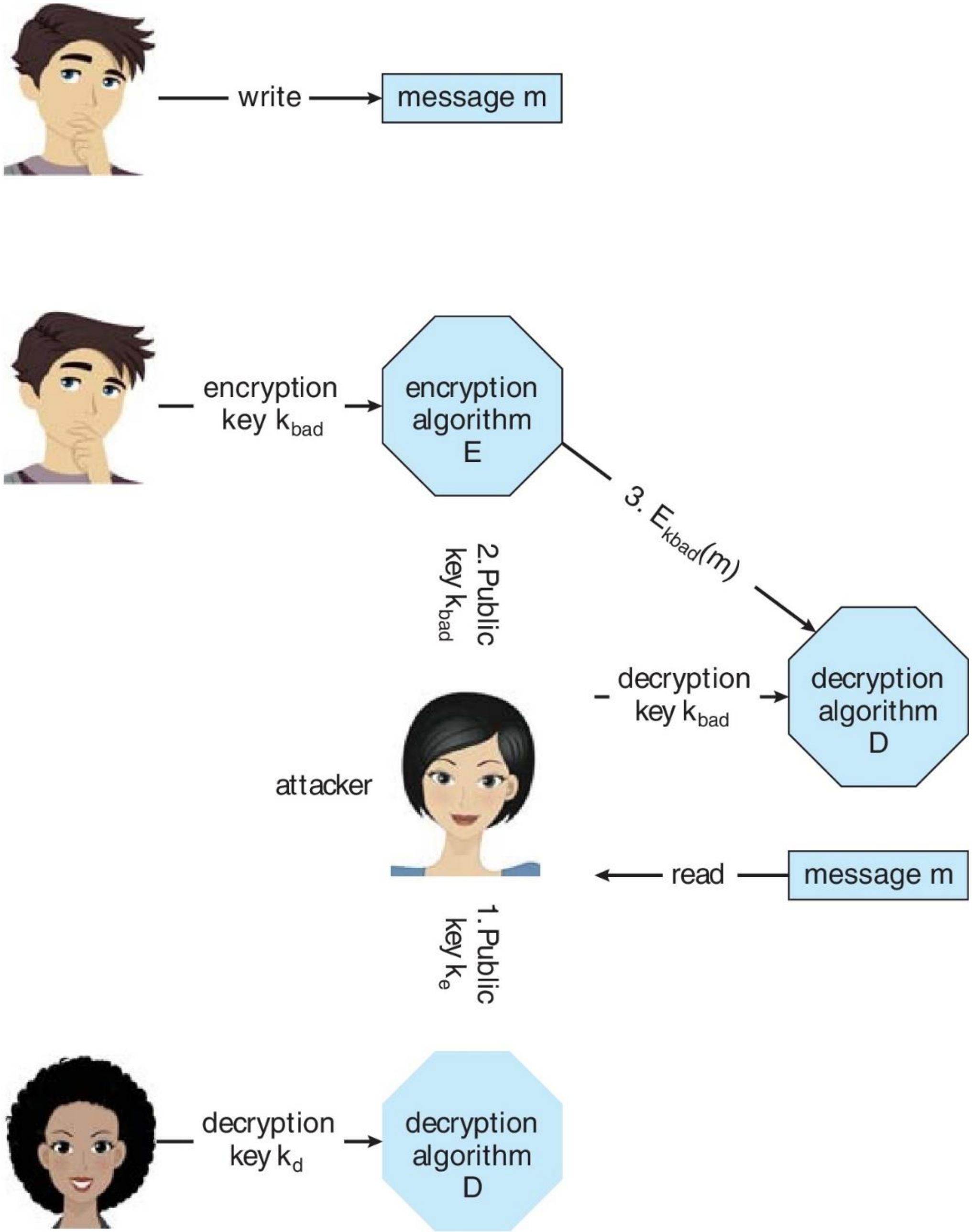
Authentication – Digital Signature

- Based on asymmetric keys and digital signature algorithm
- Authenticators produced are digital signatures
- Very useful – ***anyone*** can verify authenticity of a message
- In a digital-signature algorithm, computationally infeasible to derive k_s from k_v
  - V is a one-way function
  - Thus, k_v is the public key and k_s is the private key
- Consider the RSA digital-signature algorithm
  - Similar to the RSA encryption algorithm, but the key use is reversed
  - Digital signature of message $S_{ks}(m) = H(m)^{k_s} \bmod N$
  - The key k_s again is a pair (d, N), where N is the product of two large, randomly chosen prime numbers p and q
  - Verification algorithm is $V_{kv}(m, a)$ $(a^{k_v} \bmod N = H(m))$
    - Where k_v satisfies $k_v k_s \bmod (p - 1)(q - 1) = 1$

Digital Certificates

- Proof of who or what owns a public key
- Public key digitally signed a trusted party
- Trusted party receives proof of identification from entity and certifies that public key belongs to entity
- **Certificate authority** are trusted party – their public keys included with web browser distributions
  - They vouch for other authorities via digitally signing their keys, and so on

Man-in-the-middle Attack on Asymmetric Cryptography



Questions?

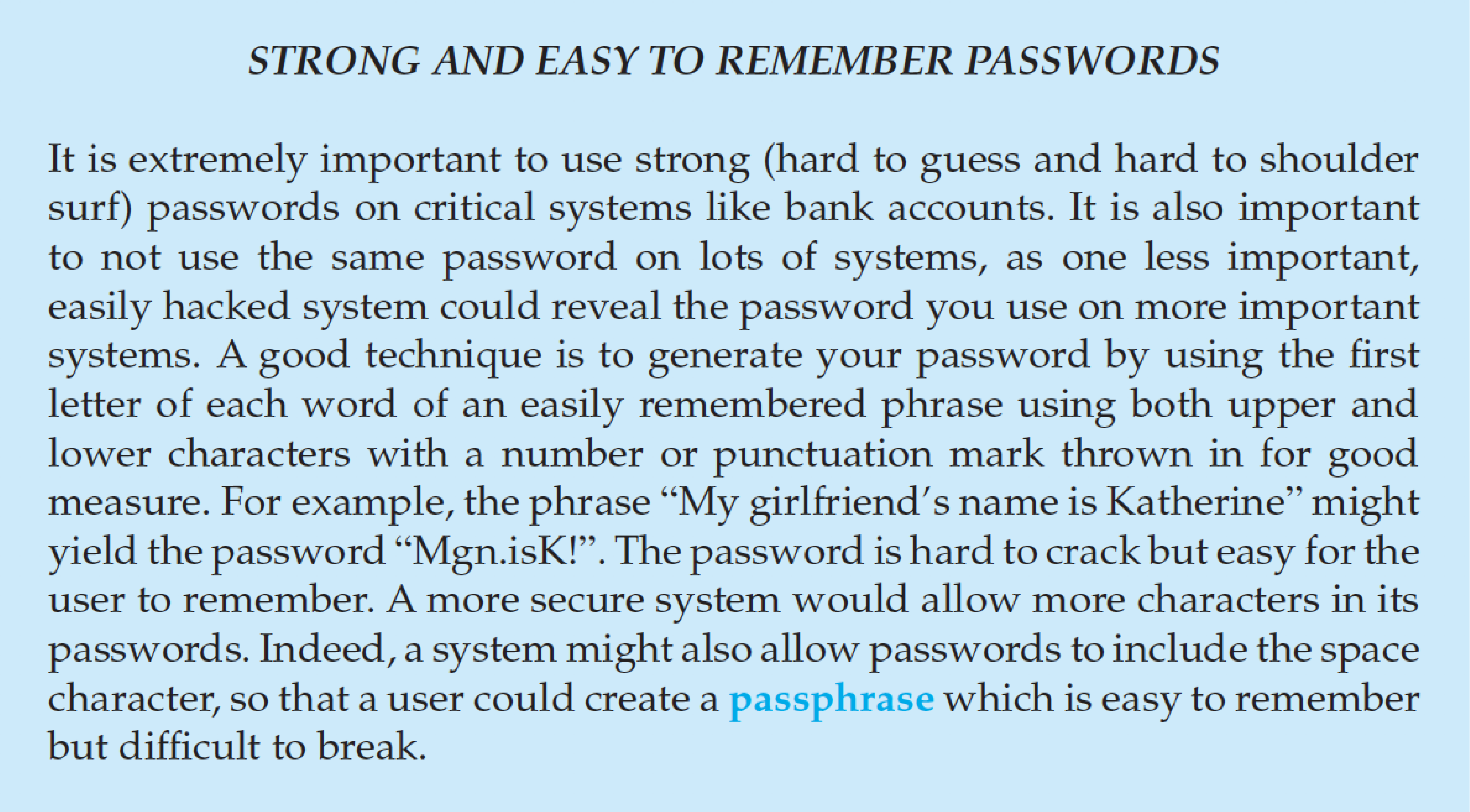
- Applications of cryptography
  - Authentication
    - Hash function
    - Message authentication code
    - Digital signature
  - Digital certificates

User Authentication

- Crucial to identify user correctly, as protection systems depend on user ID
- Based on one or combinations of 3 factors
  - What you are, what you know, and what you have
  - Password authentication
    - User identity most often established through **passwords**, can be considered a special case of either keys or capabilities
    - Assumption: you are the only one who knows your password

Password must be kept secret

- Passwords must be kept secret
  - Frequent change of passwords
  - History to avoid repeats
  - Use of “non-guessable” passwords
  - Log all invalid access attempts (but not the passwords themselves)
  - Unauthorized transfer



STRONG AND EASY TO REMEMBER PASSWORDS

It is extremely important to use strong (hard to guess and hard to shoulder surf) passwords on critical systems like bank accounts. It is also important to not use the same password on lots of systems, as one less important, easily hacked system could reveal the password you use on more important systems. A good technique is to generate your password by using the first letter of each word of an easily remembered phrase using both upper and lower characters with a number or punctuation mark thrown in for good measure. For example, the phrase “My girlfriend’s name is Katherine” might yield the password “Mgn.isK!”. The password is hard to crack but easy for the user to remember. A more secure system would allow more characters in its passwords. Indeed, a system might also allow passwords to include the space character, so that a user could create a **passphrase** which is easy to remember but difficult to break.

Encrypted or One-time Use Passwords

- Passwords may also either be encrypted or allowed to be used only once
  - Does encrypting passwords solve the exposure problem?
    - Might solve **sniffing**
    - Consider **shoulder surfing**
    - Consider Trojan horse keystroke logger
    - How are passwords stored at authenticating site?

Encrypted Passwords

- Encrypt to avoid having to keep secret
  - But keep secret anyway (i.e. Unix uses superuser-only readably file `/etc/shadow`)
  - Use algorithm easy to compute but difficult to invert
  - Only encrypted password stored, never decrypted
  - Add "salt" to avoid the same password being encrypted to the same value

One-time Passwords

- One-time passwords
  - Use a function based on a seed to compute a password, both user and computer
  - Hardware device / calculator / key fob to generate the password
    - Changes very frequently

Other User Authentication Methods

- Biometrics (what you are)
  - Some physical attribute (fingerprint, hand scan)
- Multi-factor authentication
  - Need two or more factors for authentication
    - i.e. USB “dongle”, biometric measure, and password

Questions

- User authentications
- Factors used in user authentication
- Password authentication