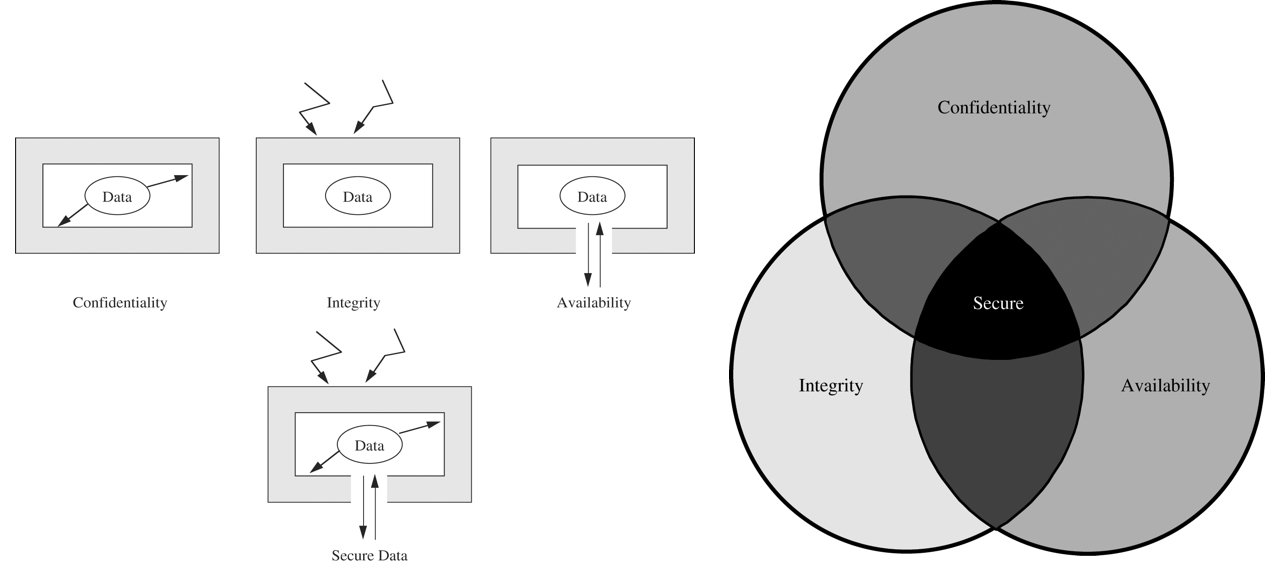
I. Basic Concepts

1.What is computer security?

Answer: 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.

2.Security is a tradeoff:



3. A vulnerability is a weakness in a system that might be exploited to cause loss or harm.

4. A threat is a potential for violation of security which could potentially cause loss or harm to a system.

5. An adversary is a subject who tries to gain unauthorized access by circumventing the security infrastructure in place.

6. An attack occurs when an adversary attempts to exploit a vulnerability.

7. Trust refers to the degree to which an entity is expected to behave.

8. Security Model:

9. Security statement: statement of what is and is not allowed.

10. Security mechanism: method for enforcing a security policy.

11. Method of defense: Prevention; Deterrence; Detection and Response; Recovery…

12. Basic Principle:

(1) Principle of the weakest link;

(2) Principle of effectiveness

II. Crypto

1.Crypto includes:

* Cryptology ⎯ The art and science of making and breaking “secret codes”
* Cryptography ⎯ making “secret codes”
* Cryptanalysis ⎯ breaking “secret codes”

2. Kerckhoffs Principle: Crypto algorithms are not secret.

3. Caesar’s Cipher (simple substitution)

4. Secure System Requirement:

* Large key space.
* No shortcut attacks.

5. Cipher design principle:

* **Confusion**: refers to making the relationship between the ciphertext and the symmetric key as complex and involved as possible;
* **Diffusion**: refers to making the relationship between the plaintext and the ciphertext as complex and involved as possible. This complexity is generally implemented through a well-defined and repeatable series of *substitutions* and *permutations*.

6. One-time pad (XOR):

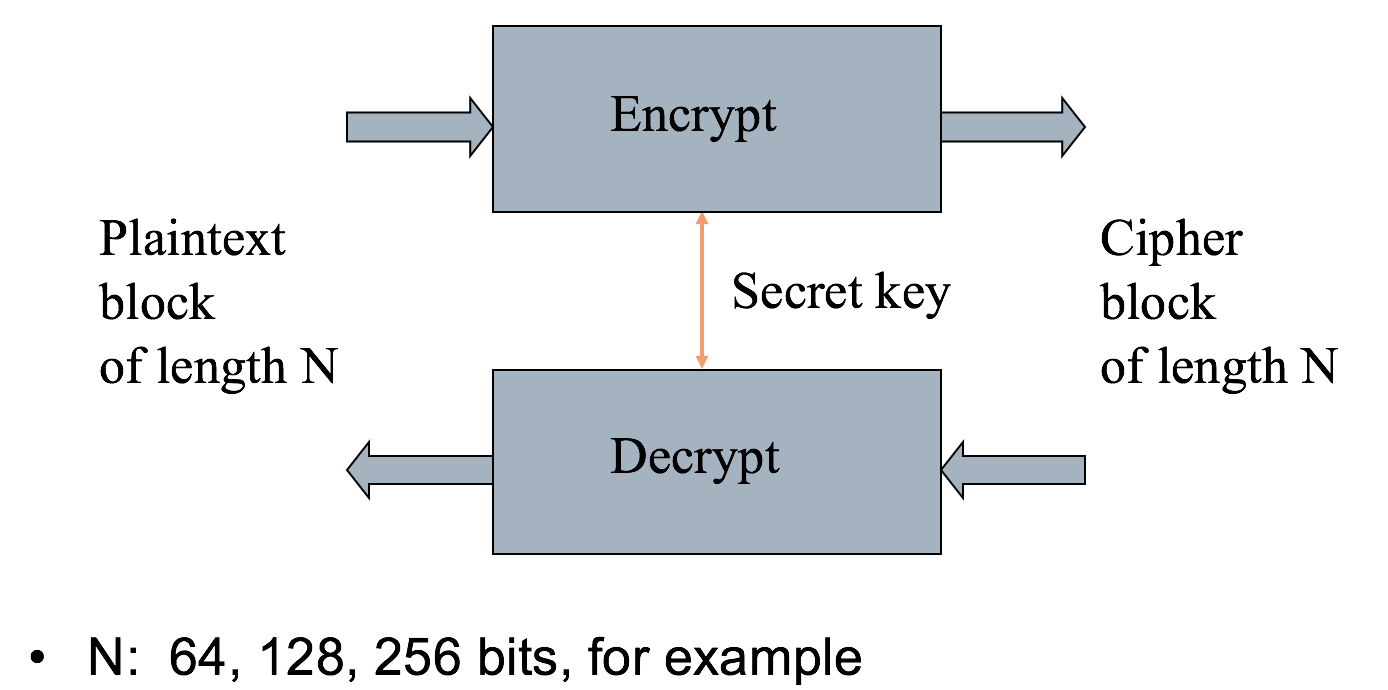
* Plaintext ⊕ Key = Ciphertext
* Ciphertext ⊕ Key = Plaintext
* Plaintext ⊕ Ciphertext = Key

7. Taxonomy of Cryptography

* Symmetric key: stream ciphers and block ciphers
* Public key
* Hash algorithms

III. Symmetric Key

1. Block cipher:



2. Block cipher parameters:

* **Block size**: Larger block sizes mean greater security but reduced encryption/ decryption speed. A block size of 128 bits is currently a reasonable tradeoff.
* **Key size**: Larger key size means greater security but may decrease encryption/ decryption speed. The most common key length in modern algorithms is 128 bits.
* **Number of rounds:** note that a single round offers inadequate security but that multiple rounds offer increasing security. A typical size is 16 rounds.
* **Subkey generation algorithm**: Greater complexity in this algorithm should lead to greater difficulty of cryptanalysis.
* **Round function**: greater complexity means greater resistance to cryptanalysis.

3. Festel cipher refers to a type of block cipher design, not a specific cipher:

For each round i=1,2,...,n, compute

Li= Ri−1

Ri= Li−1 ⊕ F(Ri−1,Ki)

where F is round function and Ki is subkey.

4. DES, 3DES, AES are block ciphers.

5. Stream cipher: one-time pad, RC4, A5/1…

6. Block ciphers process data in blocks, for longer messages must break up. So there are five modes of operations for this:

1. Electronic CodeBook (ECB): it is the simplest mode, split plaintext into blocks and encrypt each block using the same key. But it is not secure for long message.
2. CBC:
3. CFB:
4. OFB:
5. CTR:

IV. Public Key

1.Public key cryptography is Based on trap door, one-way function

* + Easy to compute in one direction
  + Hard to compute in other direction
  + “Trap door” used to create keys
  + Example: Given p and q, product N=pq is easy to compute, but given N, it is hard to find p and q.

2. Public key algorithms:

* + RSA: encryption and digital signature
  + Diffie-Hellman: key exchange
  + DSA: digital signature
  + Elliptic curve: encryption and digital signature

3. Number theory underlies most of public key algorithms

* + Prime numbers
  + The Chinese Remainder Theorem (CRT)
  + Discrete Logarithms

4. RSA base on “factoring large integer is hard.”

5. In RSA, plaintext must “smaller” than the key; ciphertext block size is the same as the key length.

6. Details about RSA.

7. Example of RSA

* + Select “large” primes p = 11, q = 3
  + Then N = pq = 33 and (p−1)(q−1) = 20
  + Choose e = 3 (relatively prime to 20)
  + Find d such that ed = 1 mod 20, we find that d = 7 works
* Public key: (N, e) = (33, 3)
* Private key: d = 7

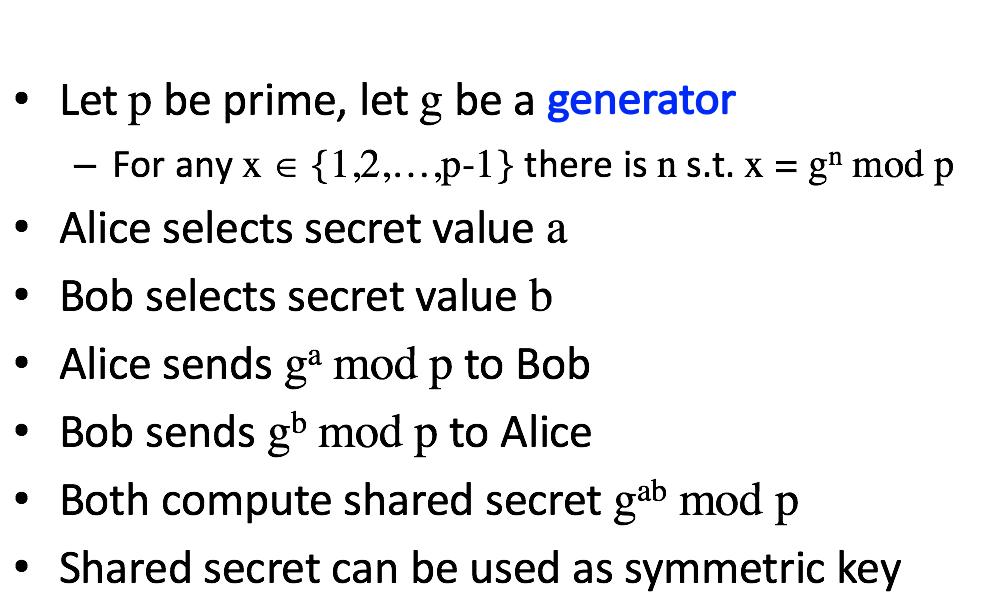
8. Example of encryption/decryption by RSA

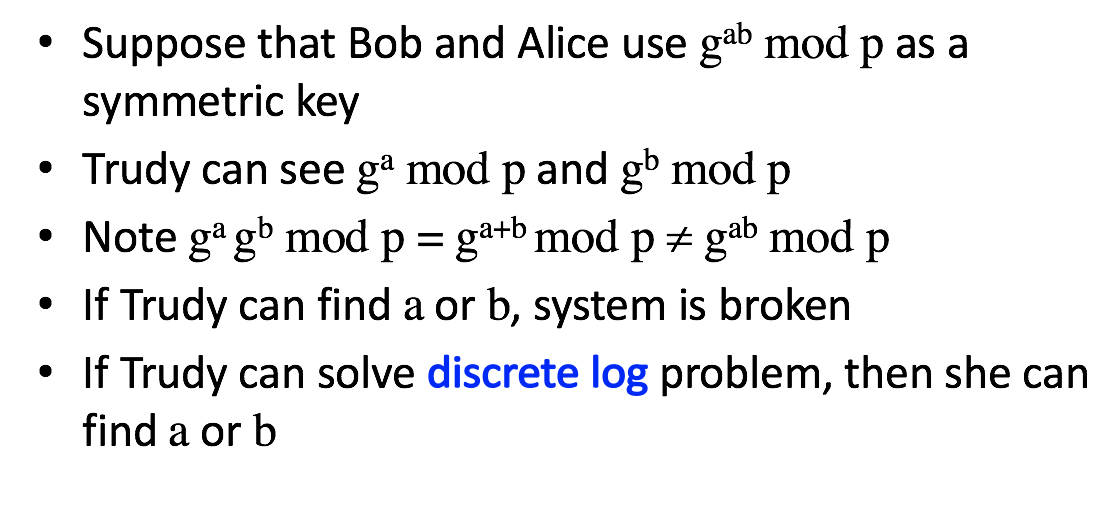
* Public key: (N, e) = (33, 3)
* Private key: d = 7
* Suppose message M = 8
* Ciphertext C is computed as C = Me mod N = 83 = 512 = 17 mod 33
* Decrypt C to recover the message M by M = Cd mod N = 177 = 410,338,673 = 12,434,505 \* 33 + 8 = 8 mod 33

9. Diffie-Hellman is a key-exchange algorithm, not for encrypting or signing.

10. The DH algorithm security rests on difficulty of discrete log problem: given g, p, and gk mod p find k.

11. Details of DH:





12. **MUST** be aware of MiM attack on Diffie-Hellman.

13. Digital signature provides integrity and **non-repudiation.**

14. Sign message M with Alice’s private key: [M]Alice

Encrypt message M with Alice’s public key: {M}Alice

15. Certificate content:

* + A unique sequence number
  + Identity of principal (who uses it)
  + Corresponding public key
  + Timestamp
    - when issued and when to expire
  + Other information
    - Can this principle issue certificates to others
    - What is the purpose of the public key
      * Encryption or signature

V. Hash Functions

1. Crypto hash function h(x) must provide:

* + Compression ⎯ output length is small
  + Efficiency ⎯ h(x) easy to computer for any x
  + One-way ⎯ given a value y it is infeasible to find an x such that h(x) = y
  + Weak collision resistance ⎯ given x and h(x), infeasible to find y ≠ x such that h(y) = h(x)
  + Strong collision resistance ⎯ infeasible to find any x and y, with x ≠ y such that h(x) = h(y)
  + Lots of collisions exist, but hard to find one

2. Hash function and Birthday problem:

* If h(x) is N bits, then 2N different hash values are possible
* sqrt(2N) = 2N/2
* Therefore, hash about 2N/2 random values and you expect to find a collision
* Implication: secure N bit symmetric key requires 2N−1 work to “break” while secure N bit hash requires 2N/2 work to “break”

3. Popular crypto hashes:

* MD5: 128 bit output; its collision recently found.
* SHA1: 160 bit output; SHA-256, SHA-384, SHA-512;

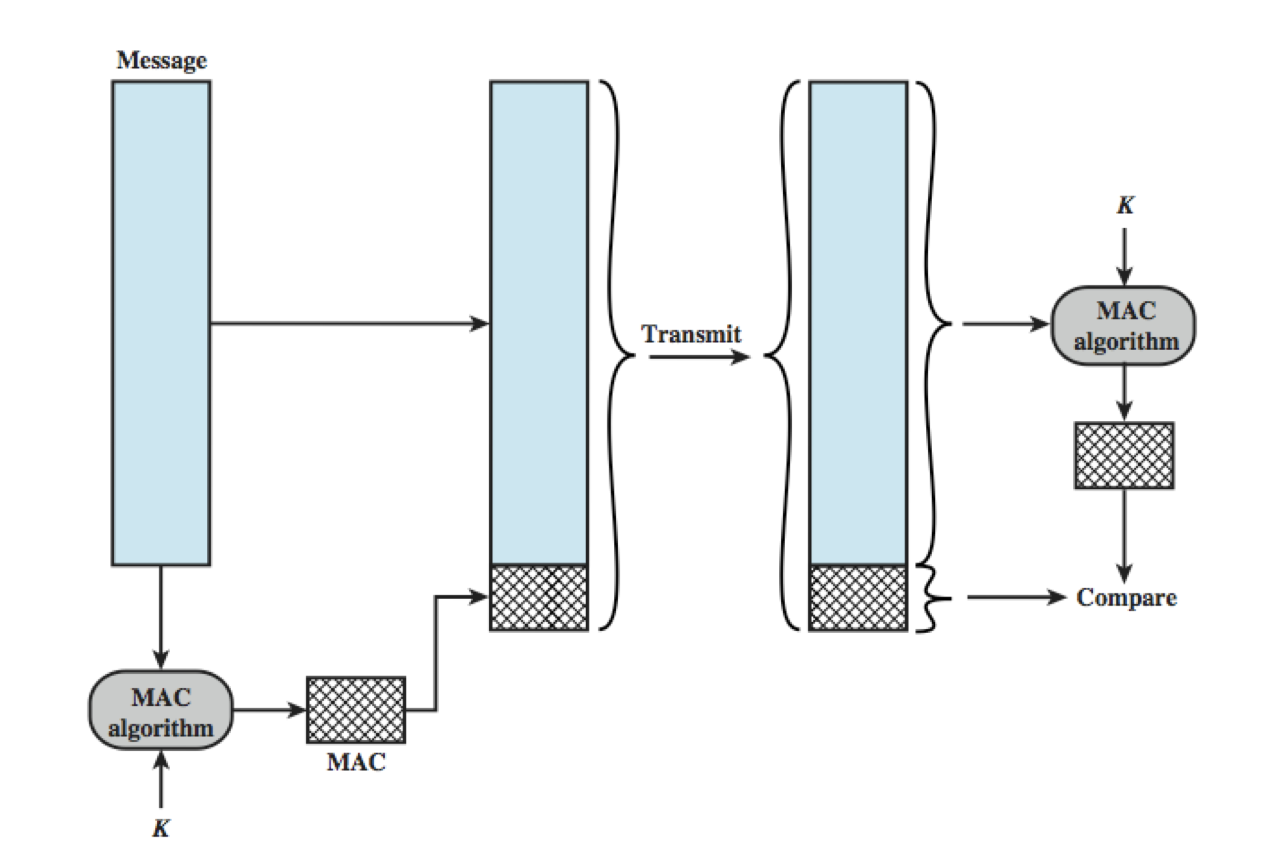
4. Crypto Hash Design:

* Desired property: avalanche effect
  + Change to 1 bit of input should affect about half of output bits
* Crypto hash functions consist of some number of rounds
* Want security and speed
  + Avalanche effect after few rounds
  + But simple rounds
* Analogous to design of block ciphers

5. Usage of Hashes:

* Authentication (HMAC)
* Message integrity (HMAC)
* Message fingerprint
* Data corruption detection
* Digital signature efficiency
* Anything you can do with symmetric crypto
* Spam reduction

6. MAC (Message Authentication Code):



7. HMAC (Hashed MAC): We can compute a MAC of the message M with key K using a “hashed MAC” or HMAC.

8. Details of HMAC: HMAC(M,K) = H(K ⊕ opad, H(K ⊕ ipad, M))

9. Secret Sharing:

* Key escrow ⎯ suppose it’s required that your key be stored somewhere
* Key can be “recovered” with court order
* But you don’t trust FBI to store your keys
* We can use secret sharing
  + Say, three different government agencies
  + Two must cooperate to recover the key

10. Random numbers of cryptography

* Random numbers used to generate keys
  + Symmetric keys
  + RSA: Prime numbers
  + Diffie Hellman: secret values
* Random numbers used for nonces
  + Sometimes a sequence is OK
  + But sometimes nonces must be random
* Random numbers also used in simulations, statistics, etc.
  + Such numbers need to be “statistically” random

11. Cryptographic random numbers must be statistically random and unpredictable.

12. Information hiding

* Digital Watermarks: add a mark to data;
* Steganography (Stego): Easy to hide info in unimportant bits; to be robust, must use important bits

Authentication Protocols

1.To prevent replay attack, challenge-response used.

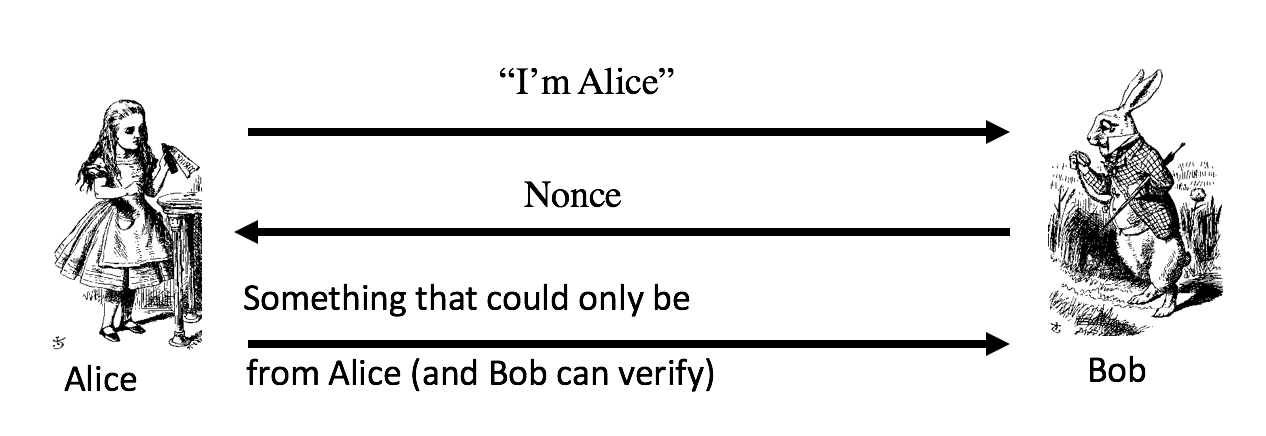
2. Challenge-Response: Suppose Bob wants to authenticate Alice

* 1. Challenge sent from Bob to Alice
  2. Only Alice can provide the correct response
  3. Challenge chosen so that replay is not possible

3. How to accomplish it?

* 1. Password is something only Alice should know…
  2. For freshness, a “number used once” or nonce

4. Details of challenge-response:



5. Symmetric key authentication:

1. Alice and Bob share symmetric key KAB
2. Key KAB known only to Alice and Bob
3. Authenticate by proving knowledge of shared symmetric key

6. Mutual authentication:

7. Public Key Authentication and Session Key:

1. Sign and encrypt with nonce…
   1. Secure
2. Encrypt and sign with nonce…
   1. Secure
3. Sign and encrypt with timestamp…
   1. Secure
4. Encrypt and sign with timestamp…
   1. Insecure
5. Protocols can be subtle!

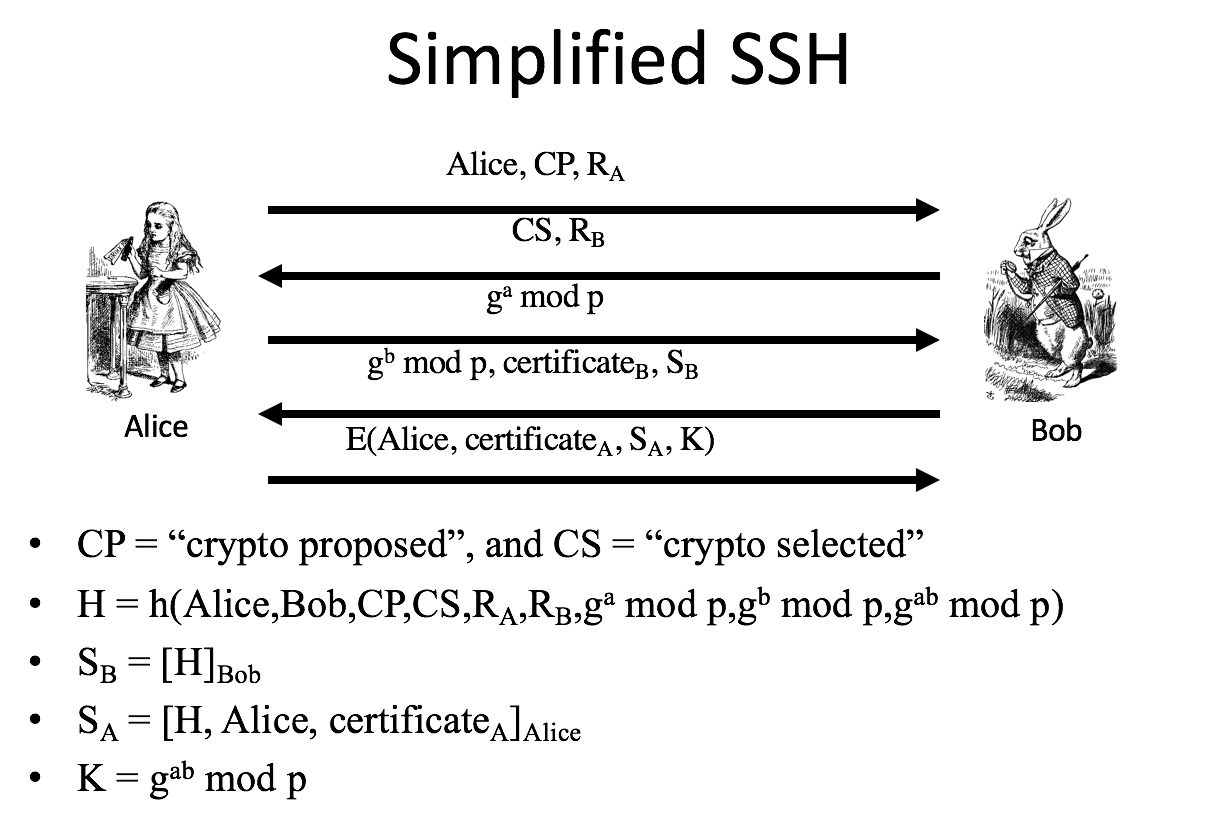
Real-world protocols

1.SSH, SSL, IPSec, Kerberos, GSM

2. Secure shell (SSL):

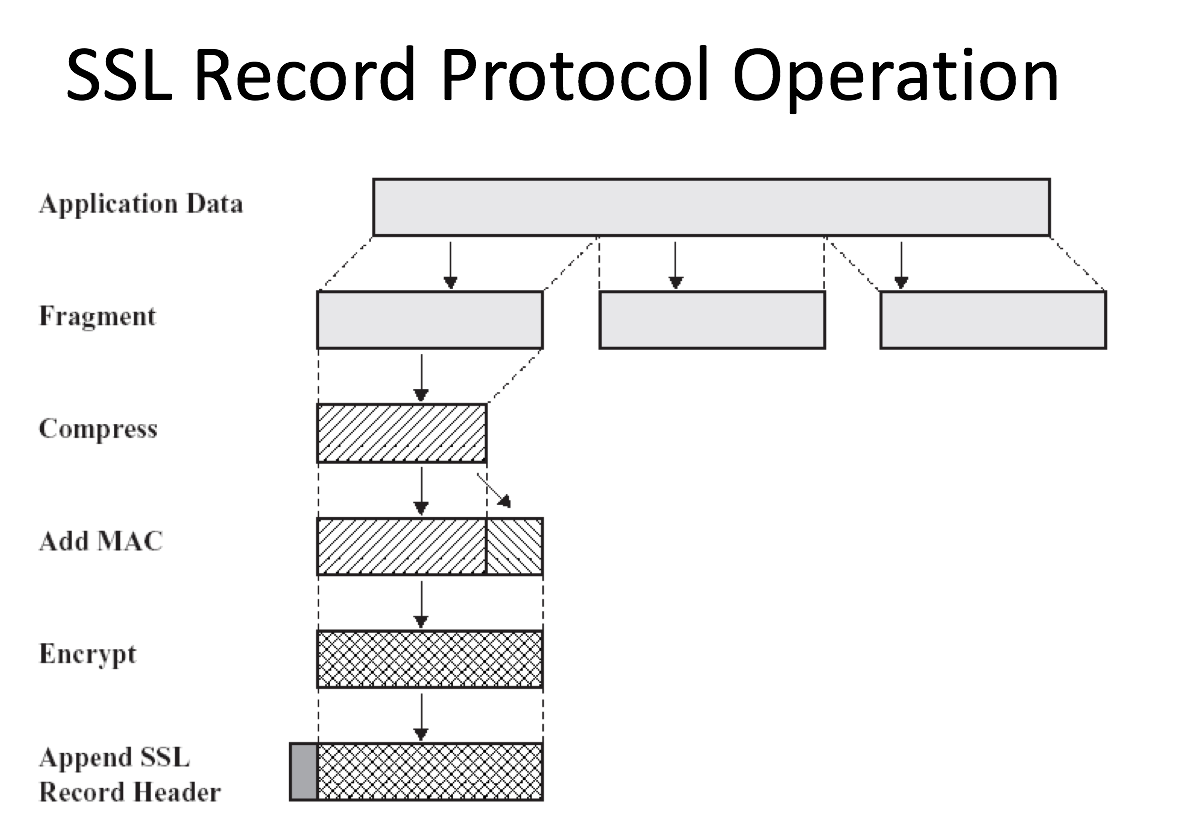
1. Creates a “secure tunnel”
2. Insecure command sent thru SSH tunnel are then secure
3. SSH used with things like rlogin
   1. Why is rlogin insecure without SSH?
   2. Why is rlogin secure with SSH?
4. SSH is a relatively simple protocol

3. SSH authentication can be based on: Public keys, or Digital certificates, or Passwords. Here, we consider certificate mode.



4. Secure Socket Layer(SSL) is the protocol used for most secure transactions over the Internet.

5. SSL record protocol operation:



6. IPSec lives at the network layer; SSL lives between application layer and transport layer.

7. IPSec architecture:

1. Two Protocols (Mechanisms)
   1. Authentication Header (AH)
   2. Encapsulating Security Payload (ESP)
2. Can be implemented in
   1. Host or gateway
3. Can work in two modes
   1. Tunnel mode
   2. Transport mode

8. SSL vs IPSec:

1. IPSec
   1. Lives at the network layer (part of the OS)
   2. Has encryption, integrity, authentication, etc.
   3. Is overly complex (including serious flaws)
2. SSL (and IEEE standard known as TLS)
   1. Lives at socket layer (part of user space)
   2. Has encryption, integrity, authentication, etc.
   3. Has a simpler specification

Appendix

1.Confidentially- read; Integrity- write;

2.HMAC, MAC are for integrity.

3.Kerckhoffs principle: 所谓密码系统是安全的，表明可以公开除了密钥以外的整个密码系统的一切内容。

4.Cipher design principle:

1. Confusion: 混淆明文和密文之间的关系；
2. Diffusion: 将明文中的统计特性扩散，并使其湮没于整个密文之中；

5.Keep cipher secure:

1. Key length;
2. Prevent shortcut attack.

6.对称密码分为两种：分组密码和流密码。分组密码（block cipher）是每次只能处理特定长度的一块数据的一类密码算法，这里的“一块”就称为分组（block）。 一个分组的比特数就称为分组长度（block lenght）。 流密码（stream cipher）是对数据流进行连续处理的一类密码算法。

7.HMAC是将密钥混入消息M之中，再进行Hash运算。他的引入是为了防止篡改Hash值。

8.DES加密算法：

1. 16轮计算
2. 64位block size
3. 56 位密钥长度
4. 48 位字密钥长度

9.AES加密算法：

1. 128位block size
2. 128，192，256位密钥长度
3. 10-14轮计算