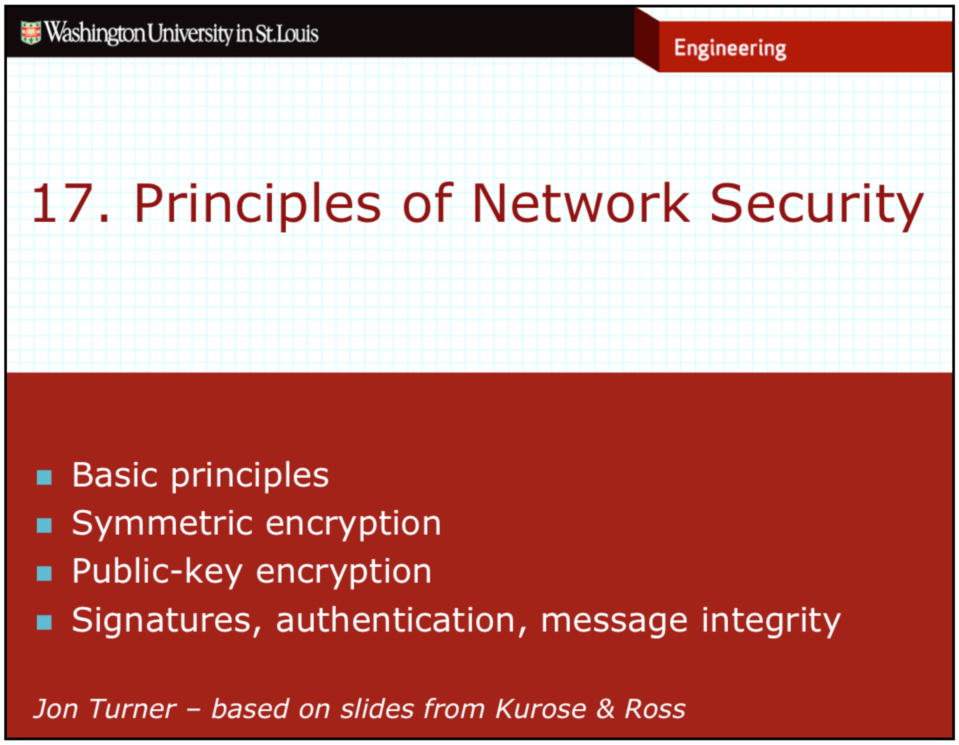


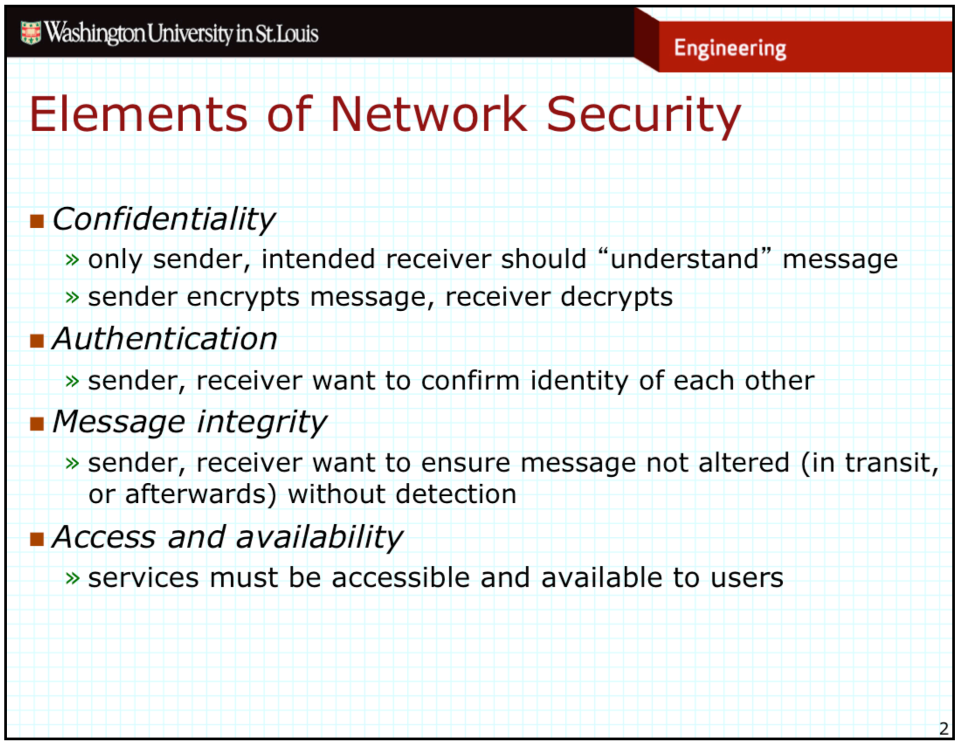
Washington University in St. Louis

Engineering

17. Principles of Network Security

- Basic principles
- Symmetric encryption
- Public-key encryption
- Signatures, authentication, message integrity

Jon Turner – based on slides from Kurose & Ross



Washington University in St. Louis

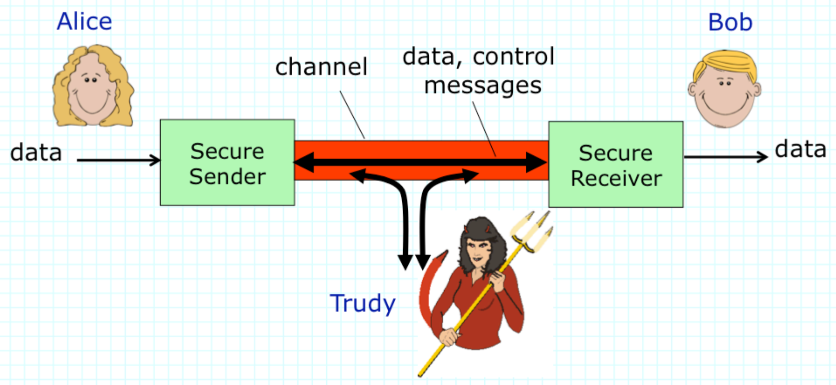
Engineering

Elements of Network Security

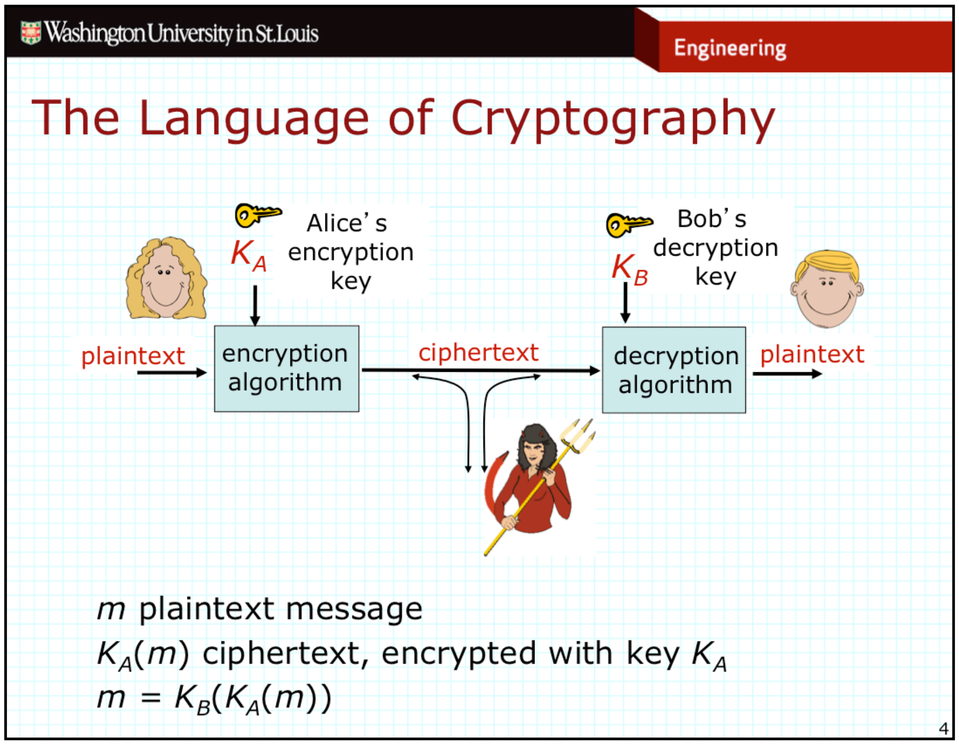
- *Confidentiality*
  - » only sender, intended receiver should "understand" message
  - » sender encrypts message, receiver decrypts
- *Authentication*
  - » sender, receiver want to confirm identity of each other
- *Message integrity*
  - » sender, receiver want to ensure message not altered (in transit, or afterwards) without detection
- *Access and availability*
  - » services must be accessible and available to users

2

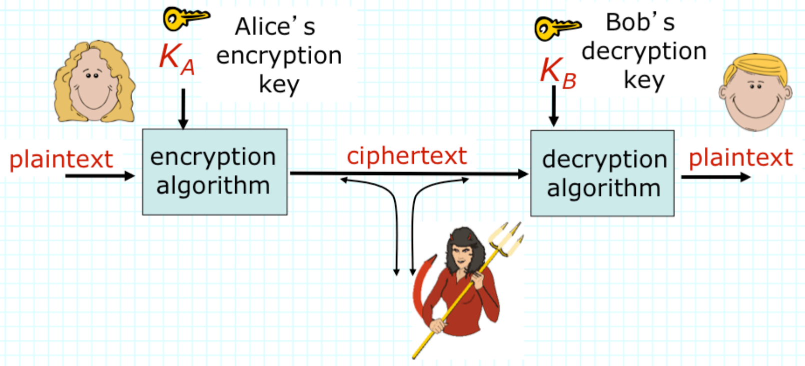
Friends, Enemies: Alice, Bob, Trudy



- Alice, Bob want to communicate "securely"
- Trudy (intruder) may intercept, delete, add messages



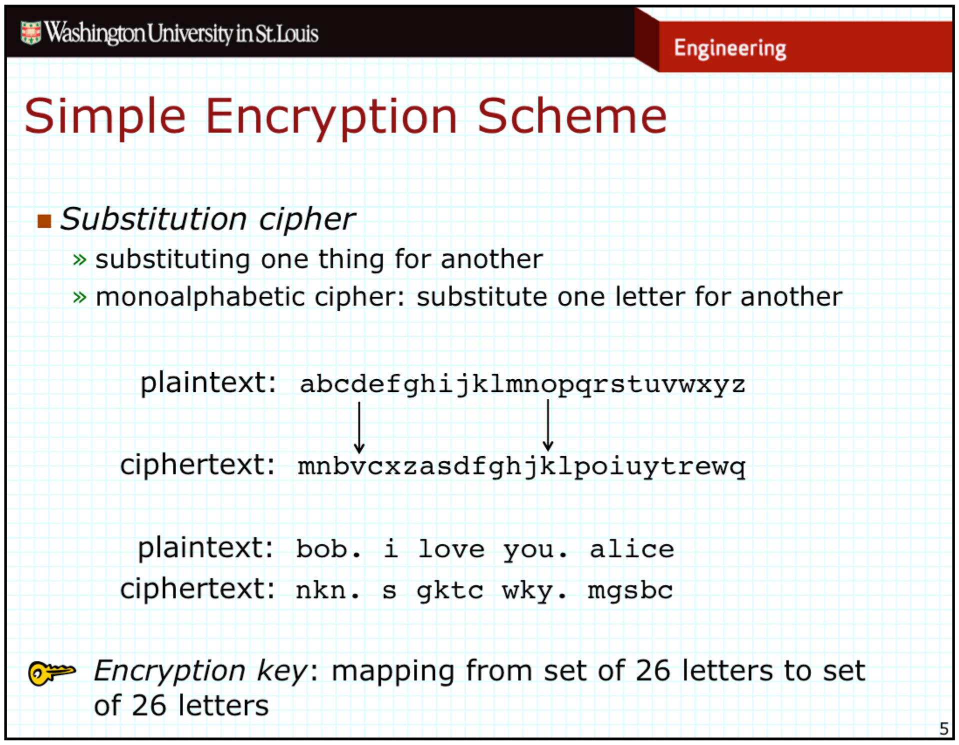
The Language of Cryptography



m plaintext message

$K_A(m)$ ciphertext, encrypted with key K_A

$m = K_B(K_A(m))$

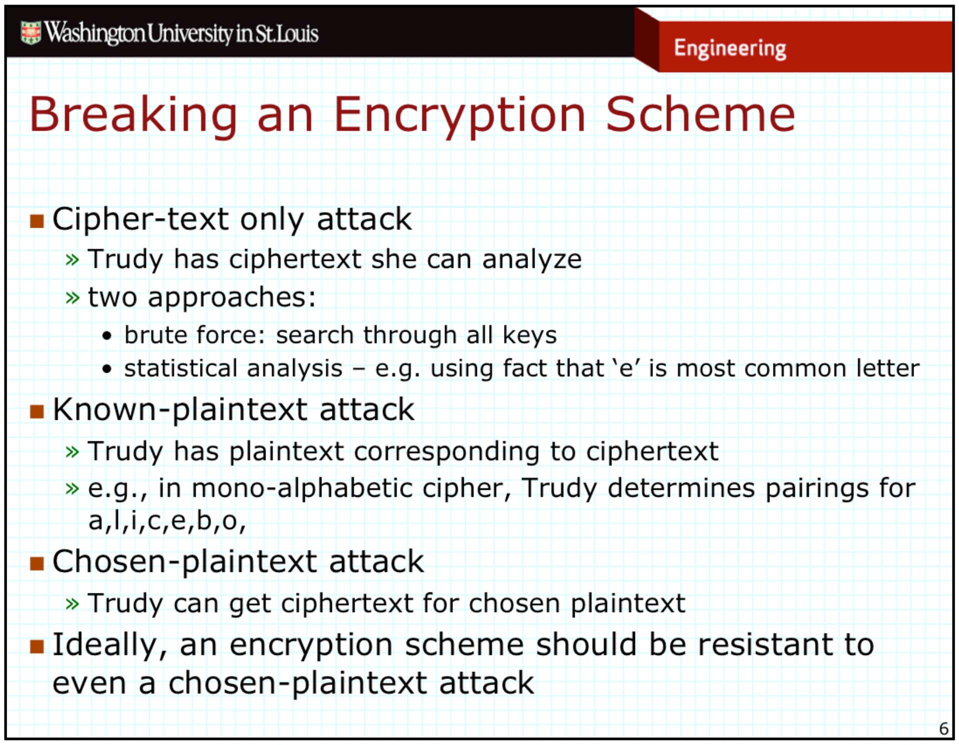


Simple Encryption Scheme

- *Substitution cipher*
  - substituting one thing for another
  - monoalphabetic cipher: substitute one letter for another

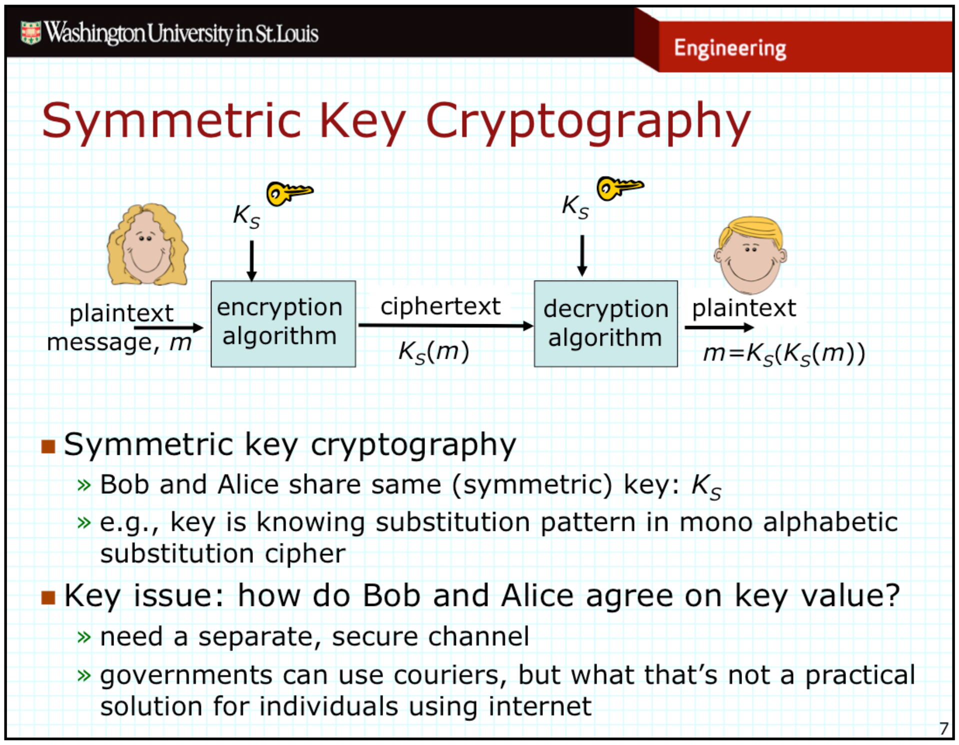
```
 plaintext:  abcdefghijklmnopqrstuvwxyz
                ↓          ↓
ciphertext:  mnbvcxzasdfghjklpoiuytrewq

 plaintext:  bob. i love you. alice
ciphertext:  nkn. s gktc wky. mgsbc
```

Encryption key: mapping from set of 26 letters to set of 26 letters



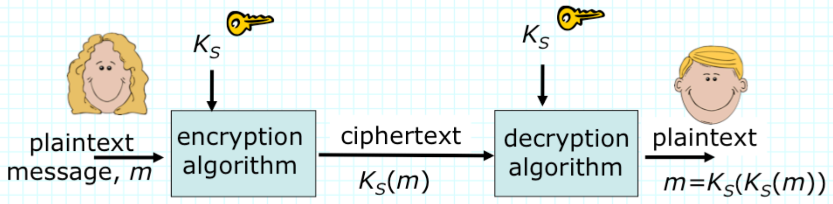
Breaking an Encryption Scheme

- Cipher-text only attack
  - » Trudy has ciphertext she can analyze
  - » two approaches:
    - brute force: search through all keys
    - statistical analysis – e.g. using fact that 'e' is most common letter
- Known-plaintext attack
  - » Trudy has plaintext corresponding to ciphertext
  - » e.g., in mono-alphabetic cipher, Trudy determines pairings for a,l,i,c,e,b,o,
- Chosen-plaintext attack
  - » Trudy can get ciphertext for chosen plaintext
- Ideally, an encryption scheme should be resistant to even a chosen-plaintext attack



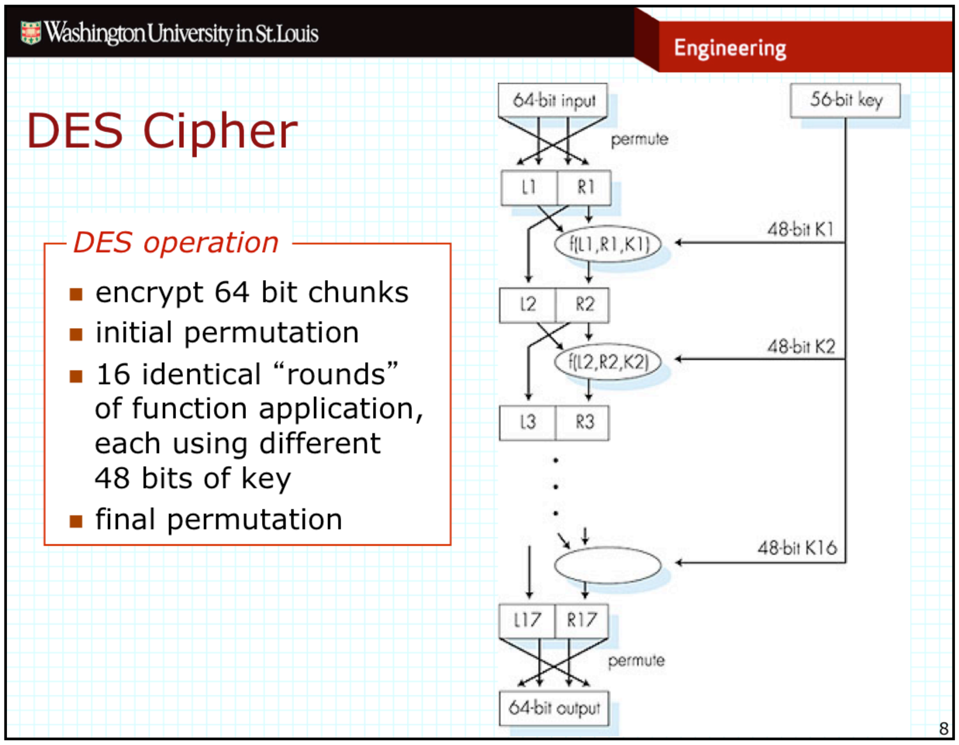
Washington University in St.Louis Engineering

Symmetric Key Cryptography



- Symmetric key cryptography
  - » Bob and Alice share same (symmetric) key: K_S
  - » e.g., key is knowing substitution pattern in mono alphabetic substitution cipher
- Key issue: how do Bob and Alice agree on key value?
  - » need a separate, secure channel
  - » governments can use couriers, but what that’s not a practical solution for individuals using internet

7



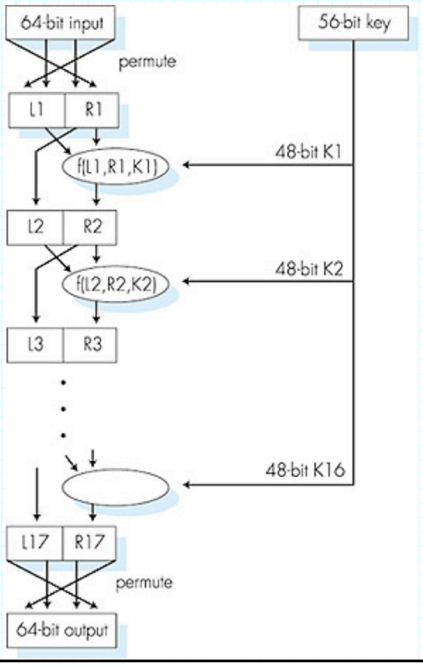
Washington University in St.Louis

Engineering

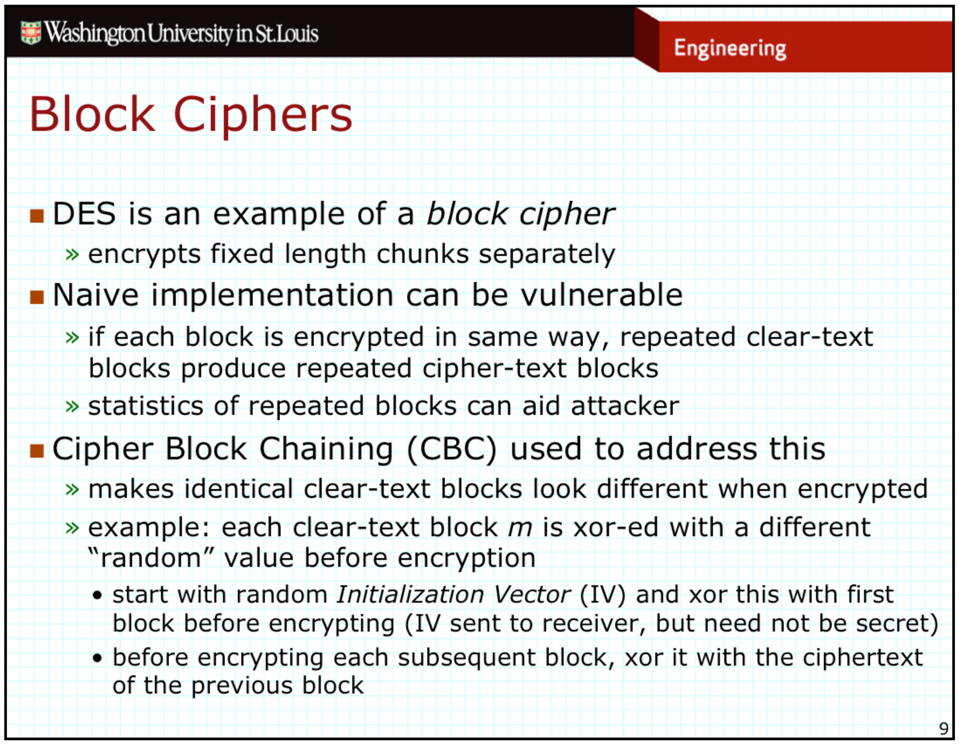
DES Cipher

DES operation

- encrypt 64 bit chunks
- initial permutation
- 16 identical “rounds” of function application, each using different 48 bits of key
- final permutation



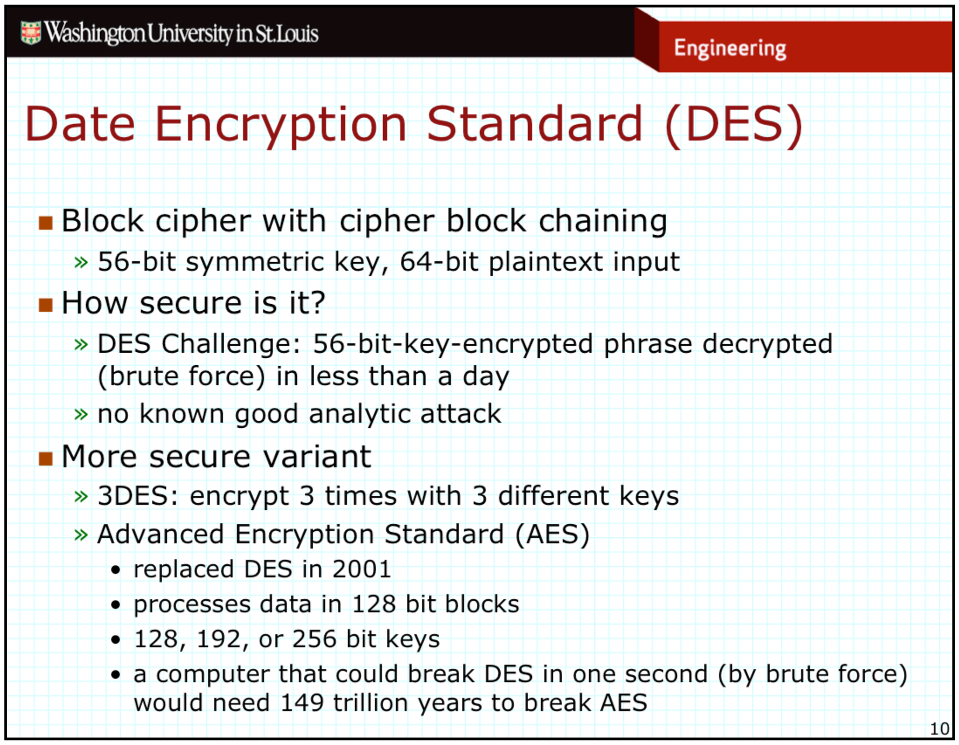
8



Block Ciphers

- DES is an example of a *block cipher*
  - » encrypts fixed length chunks separately
- Naive implementation can be vulnerable
  - » if each block is encrypted in same way, repeated clear-text blocks produce repeated cipher-text blocks
  - » statistics of repeated blocks can aid attacker
- Cipher Block Chaining (CBC) used to address this
  - » makes identical clear-text blocks look different when encrypted
  - » example: each clear-text block *m* is xor-ed with a different "random" value before encryption
    - start with random *Initialization Vector* (IV) and xor this with first block before encrypting (IV sent to receiver, but need not be secret)
    - before encrypting each subsequent block, xor it with the ciphertext of the previous block

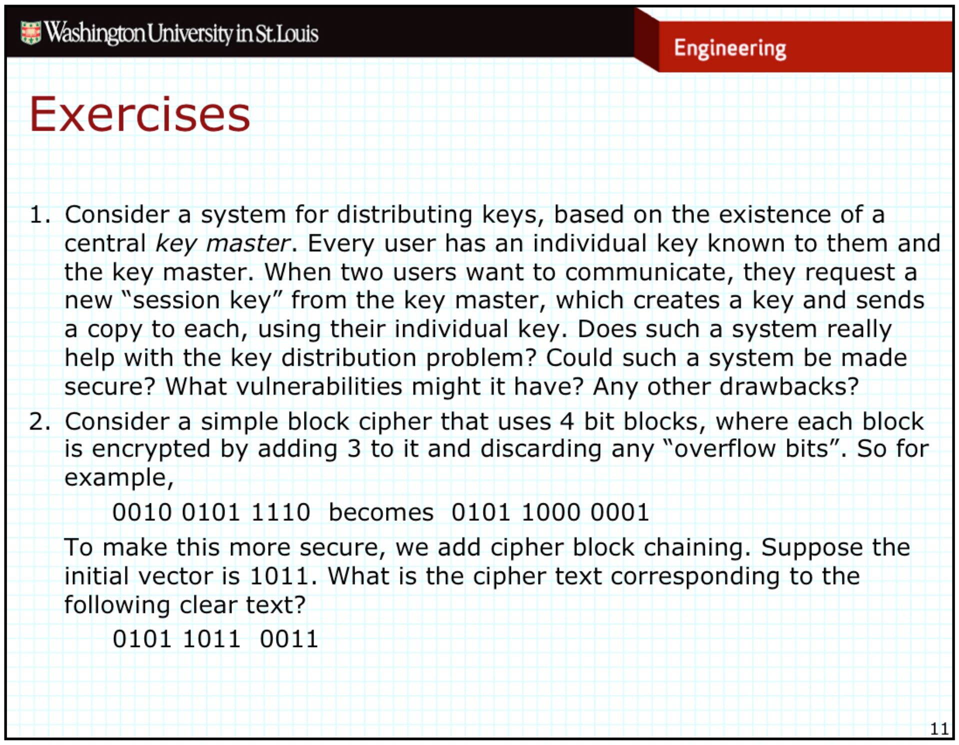
9


Washington University in St. Louis
Engineering

Date Encryption Standard (DES)

- Block cipher with cipher block chaining
  - » 56-bit symmetric key, 64-bit plaintext input
- How secure is it?
  - » DES Challenge: 56-bit-key-encrypted phrase decrypted (brute force) in less than a day
  - » no known good analytic attack
- More secure variant
  - » 3DES: encrypt 3 times with 3 different keys
  - » Advanced Encryption Standard (AES)
    - replaced DES in 2001
    - processes data in 128 bit blocks
    - 128, 192, or 256 bit keys
    - a computer that could break DES in one second (by brute force) would need 149 trillion years to break AES

10



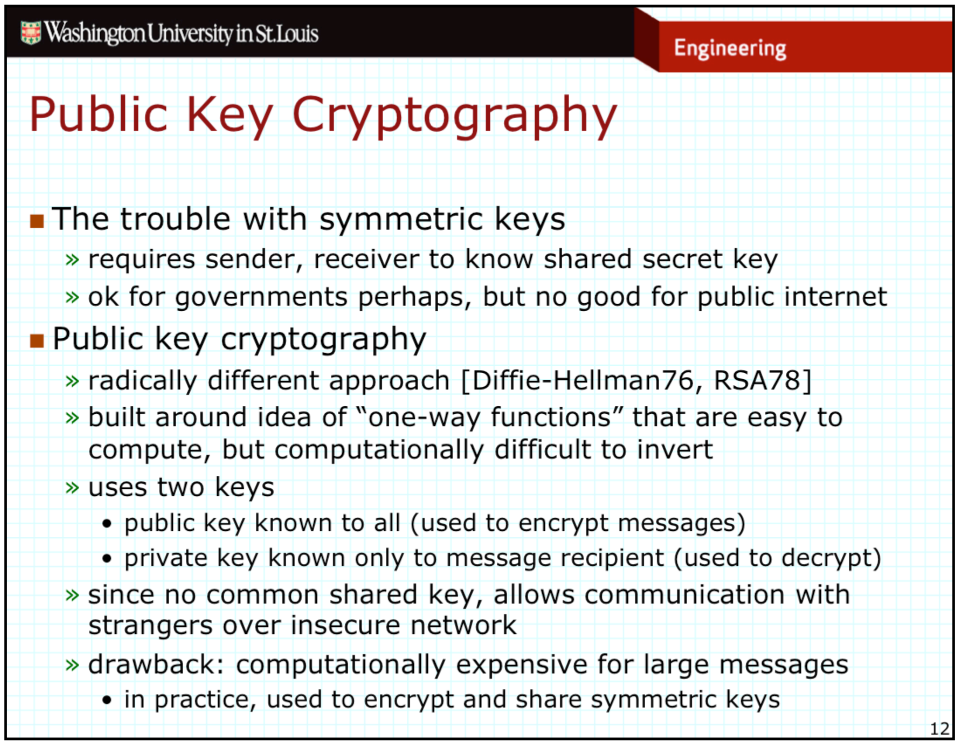
Exercises

1. Consider a system for distributing keys, based on the existence of a central *key master*. Every user has an individual key known to them and the key master. When two users want to communicate, they request a new "session key" from the key master, which creates a key and sends a copy to each, using their individual key. Does such a system really help with the key distribution problem? Could such a system be made secure? What vulnerabilities might it have? Any other drawbacks?
2. Consider a simple block cipher that uses 4 bit blocks, where each block is encrypted by adding 3 to it and discarding any "overflow bits". So for example,

   0010 0101 1110 becomes 0101 1000 0001

   To make this more secure, we add cipher block chaining. Suppose the initial vector is 1011. What is the cipher text corresponding to the following clear text?

   0101 1011 0011



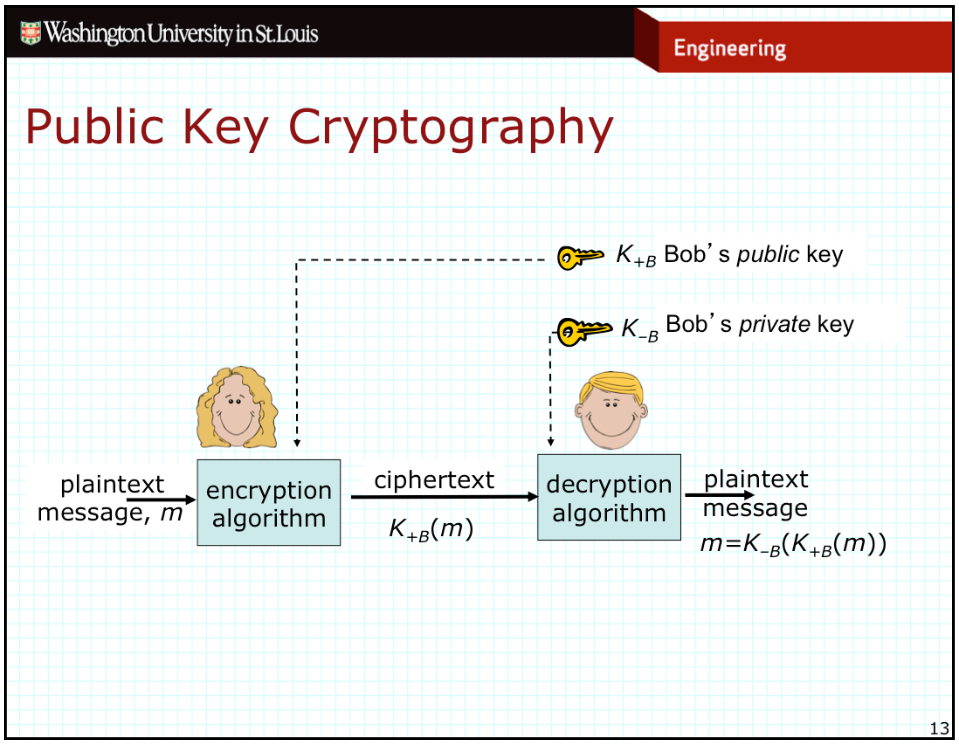
Washington University in St. Louis

Engineering

Public Key Cryptography

- The trouble with symmetric keys
  - » requires sender, receiver to know shared secret key
  - » ok for governments perhaps, but no good for public internet
- Public key cryptography
  - » radically different approach [Diffie-Hellman76, RSA78]
  - » built around idea of "one-way functions" that are easy to compute, but computationally difficult to invert
  - » uses two keys
    - public key known to all (used to encrypt messages)
    - private key known only to message recipient (used to decrypt)
  - » since no common shared key, allows communication with strangers over insecure network
  - » drawback: computationally expensive for large messages
    - in practice, used to encrypt and share symmetric keys

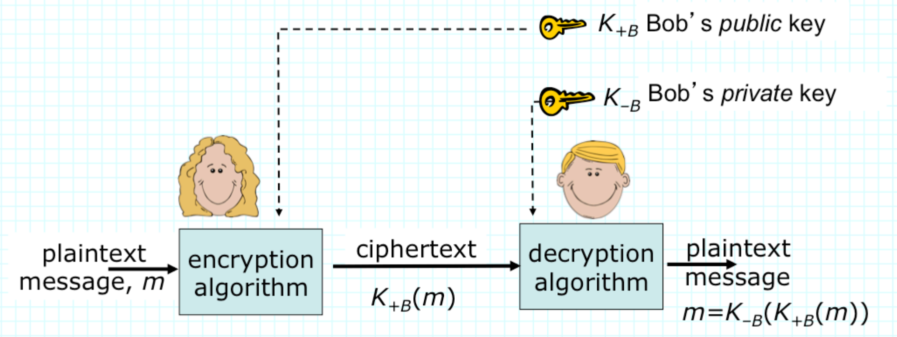
12



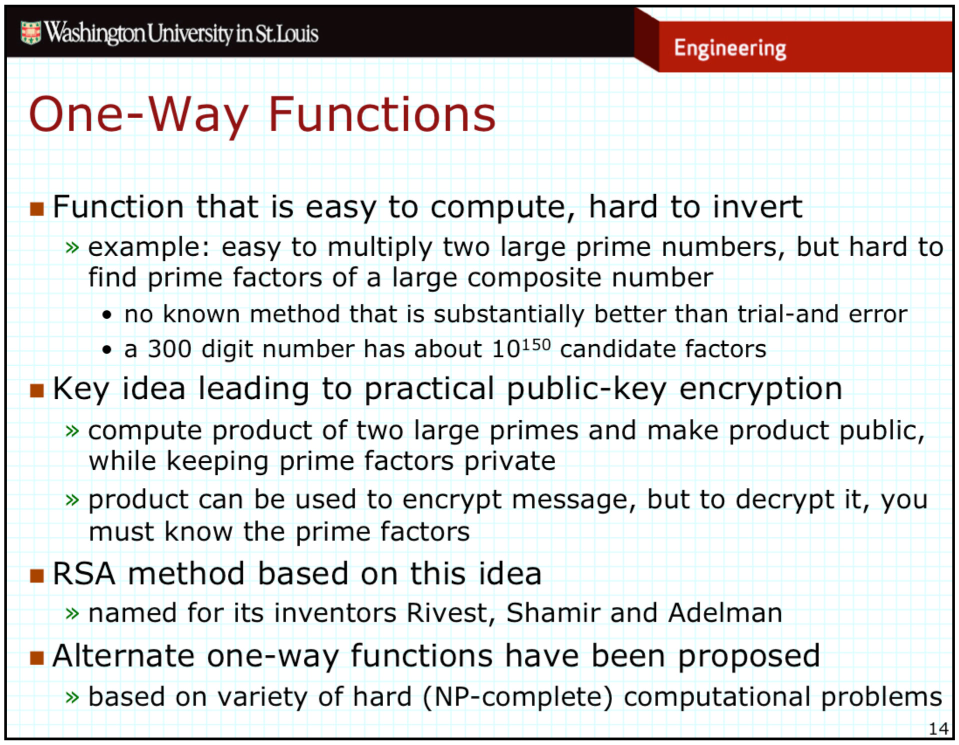
Washington University in St. Louis

Engineering

Public Key Cryptography



13



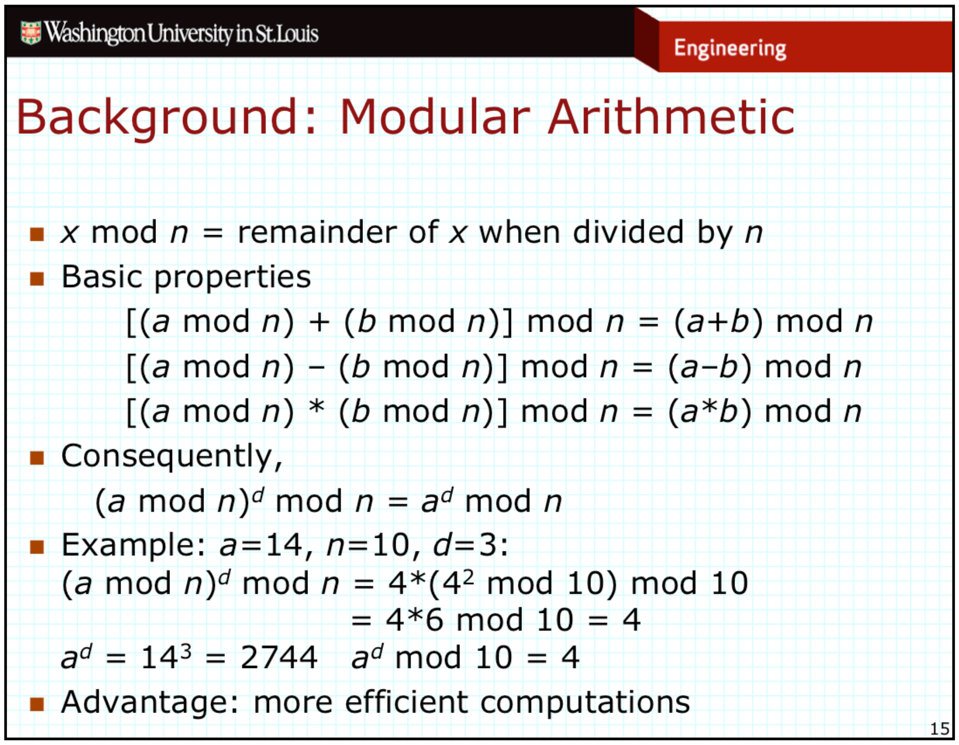
Washington University in St. Louis

Engineering

One-Way Functions

- Function that is easy to compute, hard to invert
  - » example: easy to multiply two large prime numbers, but hard to find prime factors of a large composite number
    - no known method that is substantially better than trial-and error
    - a 300 digit number has about 10^{150} candidate factors
- Key idea leading to practical public-key encryption
  - » compute product of two large primes and make product public, while keeping prime factors private
  - » product can be used to encrypt message, but to decrypt it, you must know the prime factors
- RSA method based on this idea
  - » named for its inventors Rivest, Shamir and Adelman
- Alternate one-way functions have been proposed
  - » based on variety of hard (NP-complete) computational problems

14



Background: Modular Arithmetic

- x mod n = remainder of x when divided by n
- Basic properties

  $[(a \bmod n) + (b \bmod n)] \bmod n = (a+b) \bmod n$

  $[(a \bmod n) - (b \bmod n)] \bmod n = (a-b) \bmod n$

  $[(a \bmod n) * (b \bmod n)] \bmod n = (a*b) \bmod n$
- Consequently,

  $(a \bmod n)^d \bmod n = a^d \bmod n$
- Example: a=14, n=10, d=3:

  $(a \bmod n)^d \bmod n = 4*(4^2 \bmod 10) \bmod 10$

  $= 4*6 \bmod 10 = 4$

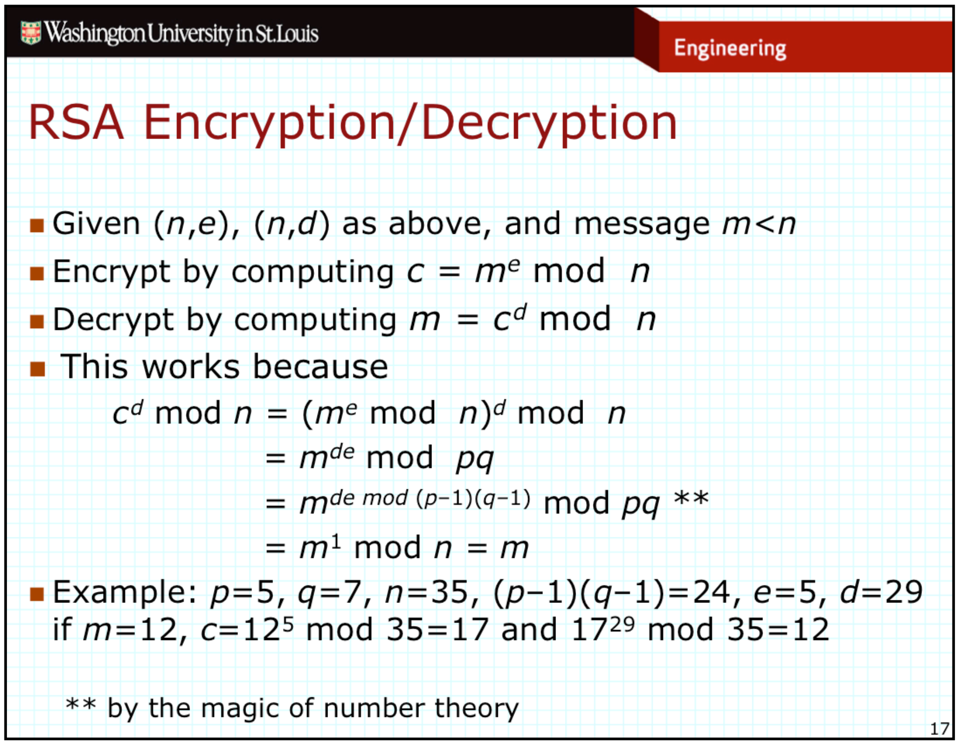
  $a^d = 14^3 = 2744$ $a^d \bmod 10 = 4$
- Advantage: more efficient computations

Creating an RSA Key Pair

1. Choose two large prime numbers p, q (say, 1024 bits long) and compute $n=pq$
2. Choose a number e that has no common factors with $(p-1)(q-1)$ – that is, e, $(p-1)(q-1)$ are *relatively prime*
3. Choose a number d such that $ed-1$ is a multiple of $(p-1)(q-1)$ (or equivalently, $d = (k(p-1)(q-1)+1)/e)$) for some positive integer k
4. Public key $K_+=(n,e)$, private key $K_-=(n,d)$

Example: $p=5$, $q=7$, $n=35$, $(p-1)(q-1)=24$, $e=5$, $d=29$

Depends on having an efficient way to generate large prime numbers and efficient ways to select e and d



Washington University in St. Louis — Engineering

RSA Encryption/Decryption

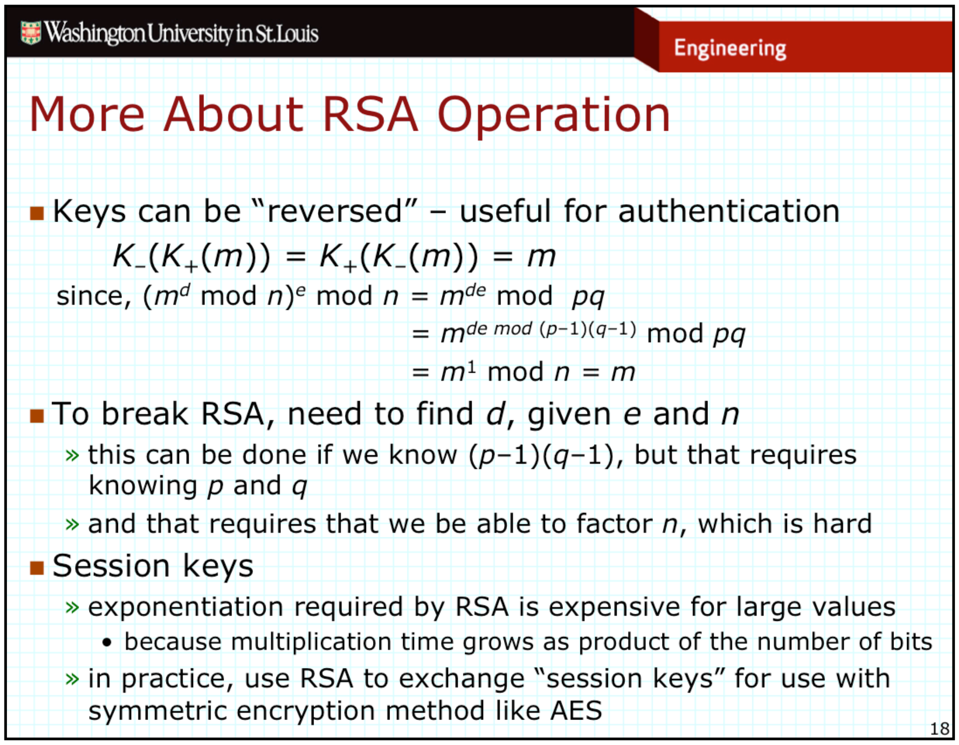
- Given (n,e), (n,d) as above, and message $m<n$
- Encrypt by computing $c = m^e \bmod n$
- Decrypt by computing $m = c^d \bmod n$
- This works because

$$\begin{aligned} c^d \bmod n &= (m^e \bmod n)^d \bmod n \\ &= m^{de} \bmod pq \\ &= m^{de \bmod (p-1)(q-1)} \bmod pq \; ** \\ &= m^1 \bmod n = m \end{aligned}$$

- Example: $p=5$, $q=7$, $n=35$, $(p-1)(q-1)=24$, $e=5$, $d=29$ if $m=12$, $c=12^5 \bmod 35=17$ and $17^{29} \bmod 35=12$

** by the magic of number theory

17



Washington University in St. Louis

Engineering

More About RSA Operation

- Keys can be "reversed" – useful for authentication

  $K_-(K_+(m)) = K_+(K_-(m)) = m$

  since, $(m^d \bmod n)^e \bmod n = m^{de} \bmod pq$

  $= m^{de \bmod (p-1)(q-1)} \bmod pq$

  $= m^1 \bmod n = m$
- To break RSA, need to find d, given e and n
  - » this can be done if we know $(p-1)(q-1)$, but that requires knowing p and q
  - » and that requires that we be able to factor n, which is hard
- Session keys
  - » exponentiation required by RSA is expensive for large values
    - because multiplication time grows as product of the number of bits
  - » in practice, use RSA to exchange "session keys" for use with symmetric encryption method like AES

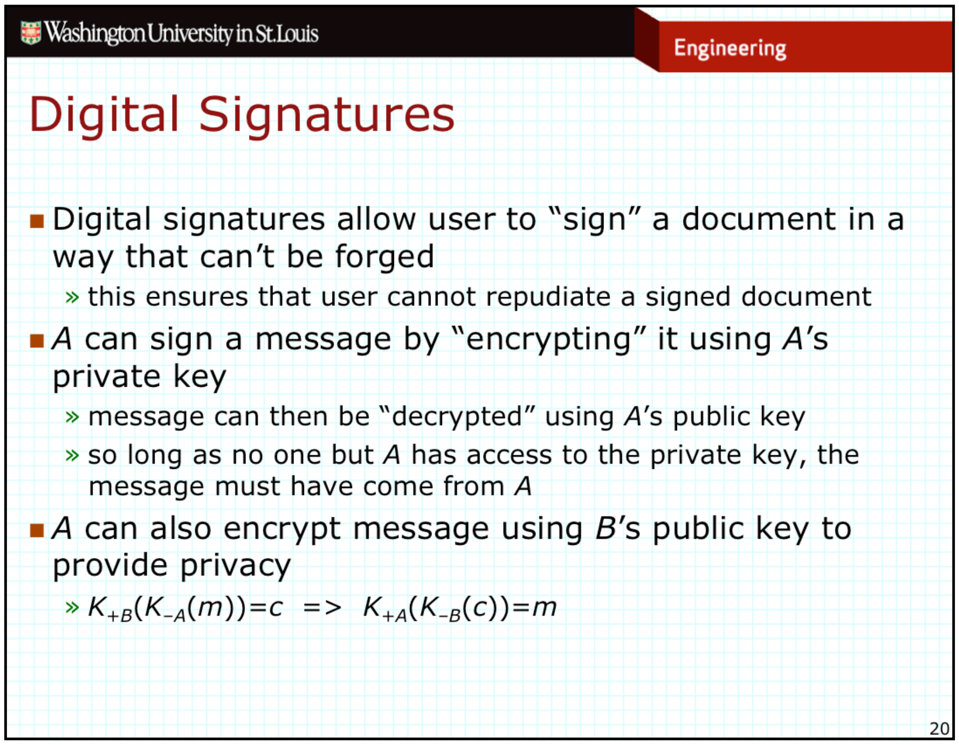
18

Washington University in St.Louis | Engineering

Exercises

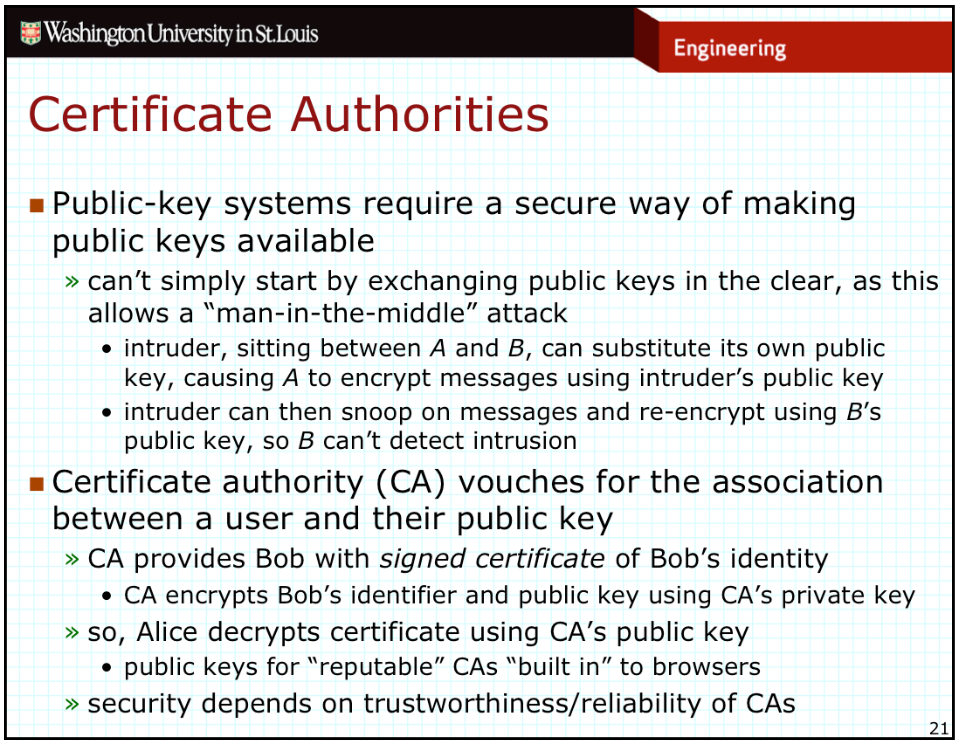
1. Consider the RSA key pair (91,5), (91,29). Assume the first is the encryption key and the second is the decryption key. What is the encrypted value of the number 10? (*Hint*: 10^2 mod 91=9)
2. Consider (31,5), (31,11) as a possible RSA key pair. Does it satisfy the requirements for such a key pair (ignore the fact that the values are too small)? Why or why not? What about (77,7), (77,43)?
3. Construct a valid RSA key pair using p=11, q=13.

19



Digital Signatures

- Digital signatures allow user to "sign" a document in a way that can't be forged
  - » this ensures that user cannot repudiate a signed document
- *A* can sign a message by "encrypting" it using *A*'s private key
  - » message can then be "decrypted" using *A*'s public key
  - » so long as no one but *A* has access to the private key, the message must have come from *A*
- *A* can also encrypt message using *B*'s public key to provide privacy
  - » $K_{+B}(K_{-A}(m))=c \;\; => \;\; K_{+A}(K_{-B}(c))=m$



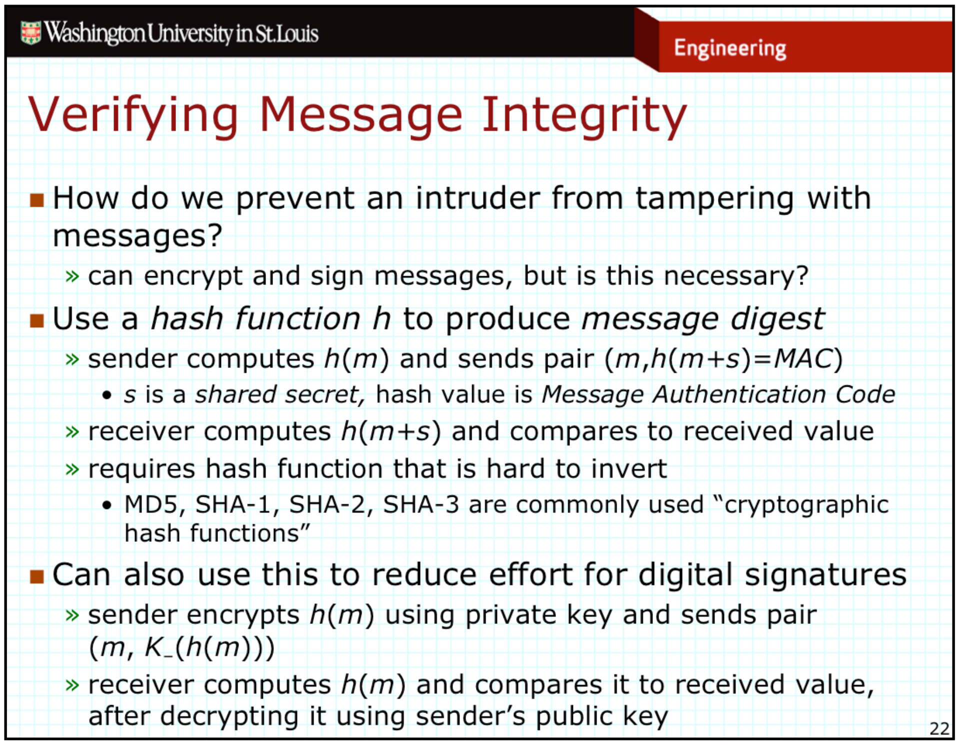
Washington University in St.Louis

Engineering

Certificate Authorities

- Public-key systems require a secure way of making public keys available
  - » can't simply start by exchanging public keys in the clear, as this allows a "man-in-the-middle" attack
    - intruder, sitting between *A* and *B*, can substitute its own public key, causing *A* to encrypt messages using intruder's public key
    - intruder can then snoop on messages and re-encrypt using *B*'s public key, so *B* can't detect intrusion
- Certificate authority (CA) vouches for the association between a user and their public key
  - » CA provides Bob with *signed certificate* of Bob's identity
    - CA encrypts Bob's identifier and public key using CA's private key
  - » so, Alice decrypts certificate using CA's public key
    - public keys for "reputable" CAs "built in" to browsers
  - » security depends on trustworthiness/reliability of CAs

21

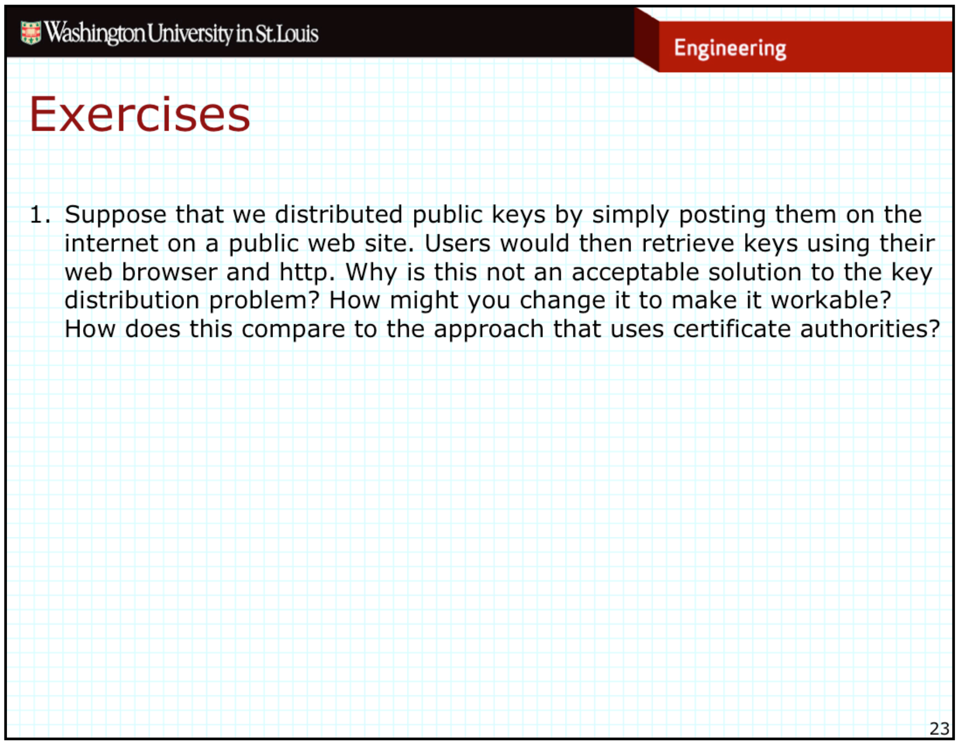


Washington University in St. Louis — Engineering

Verifying Message Integrity

- How do we prevent an intruder from tampering with messages?
  - » can encrypt and sign messages, but is this necessary?
- Use a *hash function h* to produce *message digest*
  - » sender computes $h(m)$ and sends pair $(m, h(m+s)=MAC)$
    - *s* is a *shared secret,* hash value is *Message Authentication Code*
  - » receiver computes $h(m+s)$ and compares to received value
  - » requires hash function that is hard to invert
    - MD5, SHA-1, SHA-2, SHA-3 are commonly used "cryptographic hash functions"
- Can also use this to reduce effort for digital signatures
  - » sender encrypts $h(m)$ using private key and sends pair $(m, K_-(h(m)))$
  - » receiver computes $h(m)$ and compares it to received value, after decrypting it using sender's public key

22



Exercises

1. Suppose that we distributed public keys by simply posting them on the internet on a public web site. Users would then retrieve keys using their web browser and http. Why is this not an acceptable solution to the key distribution problem? How might you change it to make it workable? How does this compare to the approach that uses certificate authorities?