1

Network Security

- Introduction
- Symmetric-Key Cryptography
- Asymmetric-Key Cryptography
- Digital Signatures, X.509 Certs & PKI
- Authentication Protocols
- Secure Socket Layer (SSL)
- IPsec
- DNSSEC

1

2

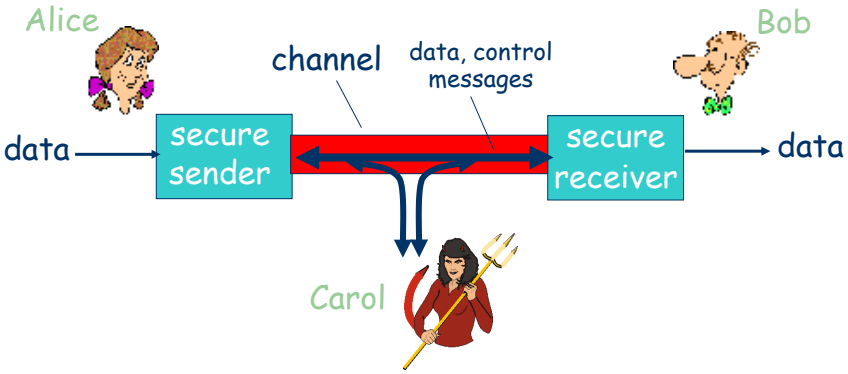
What is Network Security?

- Confidentiality
  - Only sender and intended receiver should “understand” the message contents
    - Sender encrypts msg and receiver decrypts msg
- Authentication
- Message Integrity

2

3

Friends and Enemies



- Well-known in network security world
- Bob, Alice want to communicate “securely”

3

4

Who might Bob and Alice be?

- Well, real-life Bobs and Alices!
- Web browser/server for electronic transactions
- DNS servers
- Routers exchanging routing table updates

4

5

What can the bad guys do?

- Passive Attack
- Active Attack
  - Actively insert messages into connection
- Impersonation
  - Fake (spoof) source address in packet (or any field in packet)
- Hijacking
  - "Take over" ongoing connection by removing sender or receiver, inserting himself in place

5

6

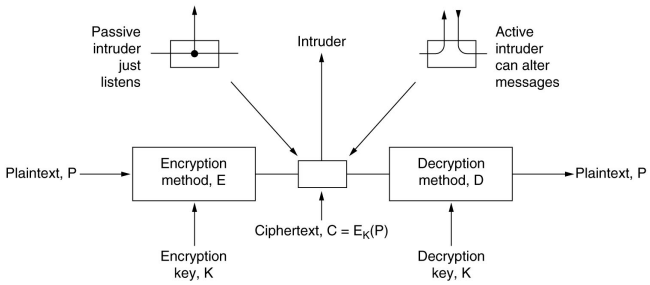
Cryptography

- Original data to be transferred is called Plaintext or Cleartext
  - Encrypted version is called Ciphertext
- Plaintext is denoted P, whereas ciphertext is denoted C
  - Encryption function E operates on P to produce C
- In the reverse process
  - Decryption function D operates on C to produce P
- Following identity must also hold true for the cryptosystem to function correctly

6

Cryptographic Keys

- All modern encryption algorithms use a key denoted by K
- The key can take on many possible values



- The encryption and decryption functions now become

7

Substitution Ciphers

- Each letter or a group of letters is replaced by another letter or group of letters to disguise it
- Caesar Cipher - Mono-alphabetical Substitution

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

- Q: What is the key?
- To send a secret message
  - Letters of the message are taken one by one and the letters appearing below are written instead
- The message “send spears” would be enciphered as

8

9

Substitution Ciphers II

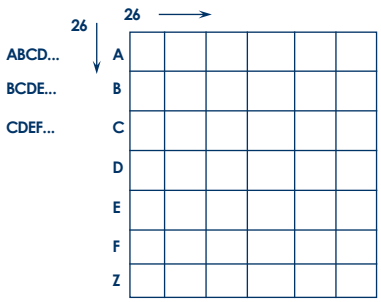
- Attacks
  - Identify commonly occurring characters
  - Commonly occurring bigrams/digrams
  - Domain specific buzz words
- Substitution ciphers **preserve** the order of the text symbols but disguise them

9

10

The Vigenère Cipher

- Some protection from the above can be gained by using a poly-alphabetical cipher



- Now pick a key e.g. AFGHANISTANBANANISTAN
  - Use row A to encrypt first letter of plaintext, row F the second time around etc...

Q: How can we break this cipher?

10

Transposition Ciphers

- Transposition ciphers reorder the symbols

M	E	G	A	B	U	C	K
7	4	5	1	2	8	3	6
p	l	e	a	s	e	t	r
a	n	s	f	e	r	o	n
e	m	i	l	l	i	o	n
d	o	l	l	a	r	s	t
o	m	y	s	w	i	s	s
b	a	n	k	a	c	c	o
u	n	t	s	i	x	t	w
o	t	w	o	a	b	c	d

Plaintext

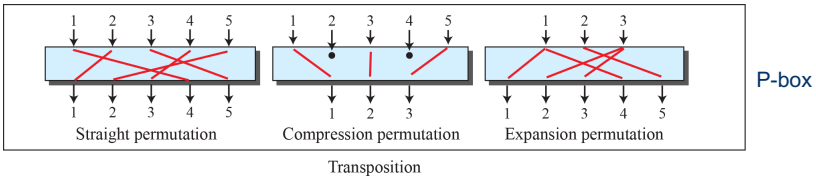
pleasetransferonemilliondollarsto
myswissbankaccountsixtwotwo

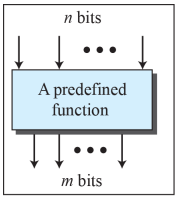
Ciphertext

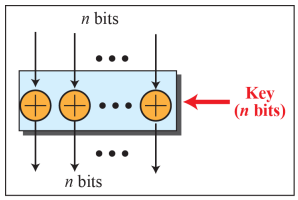
AFLLSKSOSELAWAIATOOSSCTCLNMOMANT
ESILYNTWRNNTSOWDPAEDOBUOERIRICXB

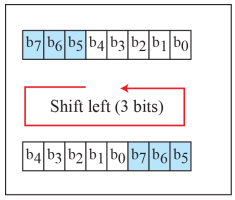
- The plaintext is written horizontally in rows
- Ciphertext is read out in columns
  - Starting with the column whose weight is the lowest

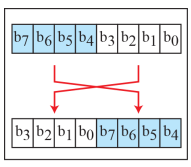
Components of a Modern Block Cipher

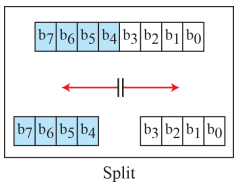


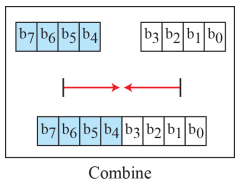












13

Symmetric-Key Encryption

- Based on the sender and the receiver of a message knowing and using

- Sender uses the secret key to encrypt the message
  - Receiver uses the same secret key to decrypt the message
- Main problem?
- Examples of symmetric key algorithms are
  - DES, Triple DES, IDEA, AES

13

14

Data Encryption Standard (DES)

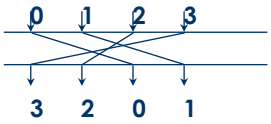
- In January 1977 a standard encryption method was adopted by the U.S. gov
  - Origins lie in an internal IBM project codenamed Lucifer
- Though the algorithm used is complex
  - It is easily implemented in hardware
  - Software implementations are also widely available
- DES is a Block Cipher
  - Operates on a single chunk of data at a time
    - 64 bits (8 bytes)
- The key length is 56 bits
  - Often expressed as a 8 character string with the extra bits used as a parity check

14

15

DES

- Algorithm has 19 distinct stages
- First stage re-orders the bits of the 64-bit input block by applying a fixed permutation (P-box)



Transposition of 4 bits (example only)

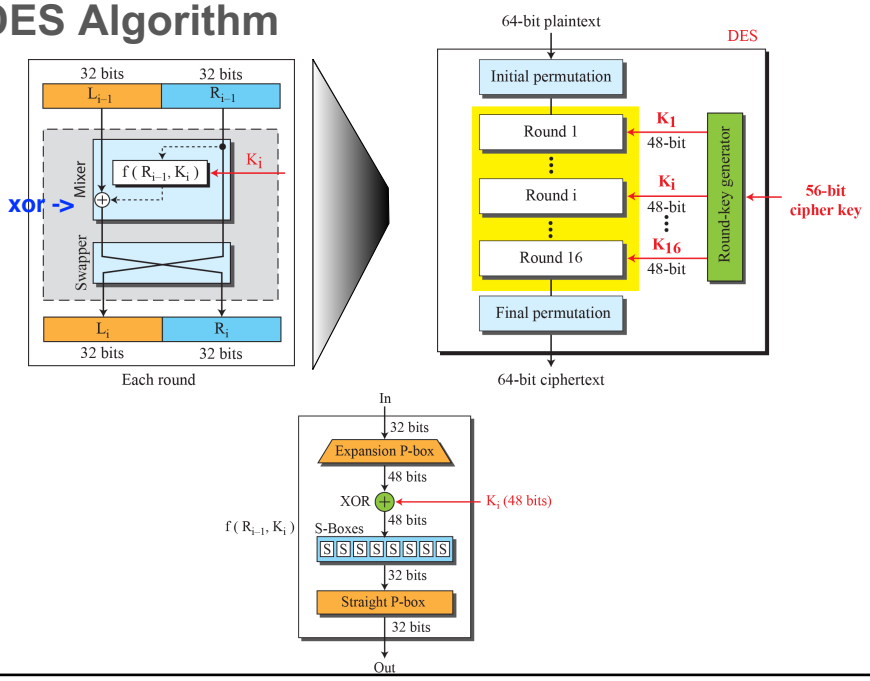
- Last stage is the exact inverse of this permutation
- Stage penultimate to the last one
  - Exchanges the leftmost 32 bits with the rightmost 32 bits
- Remaining 16 stages are called **Rounds**
  - Functionally identical but take as an input a quantity computed from the key and the round number

-Round key is 48 bits

15

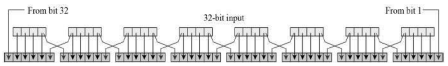
16

DES Algorithm

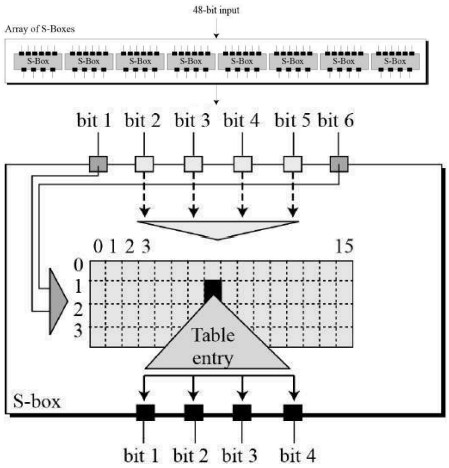


16

Permutations and Substitutions



32	01	02	03	04	05
04	05	06	07	08	09
08	09	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	31	31	32	01



17

Cracking DES

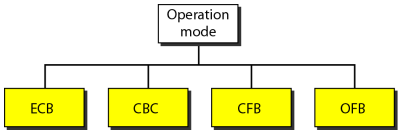
- 56 bits is a short key
- Brute force attack (try every key)
  - **2^56 encryptions to try all keys**
  - Special chips can check 4 million keys/sec
  - $1 million DES cracking machine could break it in a few hours
- Jan'99 Challenge III won in 22 hrs and 15 mins using a supercomputer and 100,000 Internet nodes
  - Tested 245 billion keys per second
- Improvement - Triple DES or 3DES
  - **Use two or three keys (112 or 168 bits)**
  - **uses EDE or DED mode - why?** **Encryption-Decryption-Encryption** **Decryption-Encryption-Decription**
    Use a different key for each cycle
    ie. encrypt with one, decrypt with the other - still compatible with one key

Side note: it took only a few days for the ~100 machines in the ICT huts to crack DES Hitesh and the boys were bored one summer

18

19

Modes of Operation for Block Ciphers



- The previous discussion of DES is known as Electronic Code Book (ECB) mode
- Each 64-bit block is encoded *independently* of all other blocks
  - Q: What is the problem with this approach?

    Modified content cannot be detected
- Block ciphers operating in ECB mode can be parallelized

  Very parallel - High speed

19

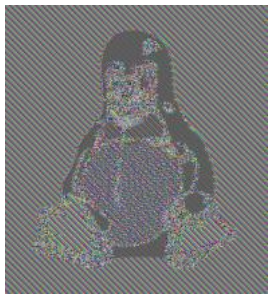
20

ECB Problems

- ECB mode encrypts in a highly **deterministic** manner
- Identical plain text blocks result in identical ciphertext blocks

  -As long as key does not change

Original Image



Encrypted with ECB Mode

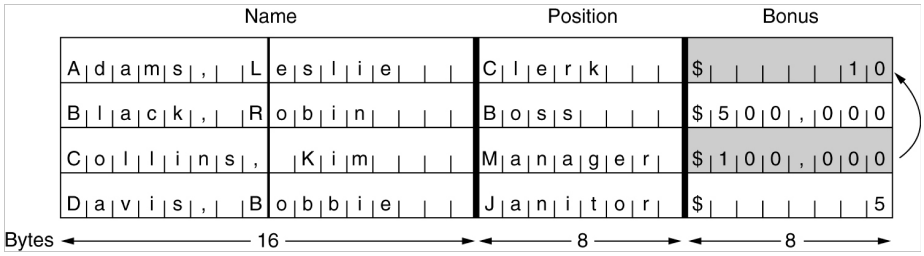
20

21

ECB Substitution Attacks

- Each block is encrypted independently of all other blocks

- allows for a passive intruder to manipulate the ciphertext blocks without the receiver noticing



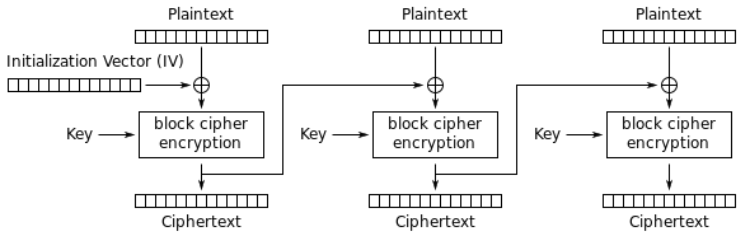
Swapping the two blocks gives leslie a huge bonus and is undetectable

21

22

Cipher Block Chaining Mode (CBC)

- In CBC mode each block of plaintext is XORed with previous ciphertext block before being encrypted
- Each ciphertext block depends on all plaintext blocks processed up to that point



Cipher Block Chaining (CBC) mode encryption

-To make each message unique an initialisation vector must be used in the first block

22

CBC Mode

Definition 5.1.2 Cipher block chaining mode (CBC)
Let $e()$ be a block cipher of block size b; let x_i and y_i be bit strings of length b; and IV be a nonce of length b.
Encryption (first block): $y_1 = e_k(x_1 \oplus IV)$
Encryption (general block): $y_i = e_k(x_i \oplus y_{i-1}), \quad i \geq 2$
Decryption (first block): $x_1 = e_k^{-1}(y_1) \oplus IV$
Decryption (general block): $x_i = e_k^{-1}(y_i) \oplus y_{i-1}, \quad i \geq 2$

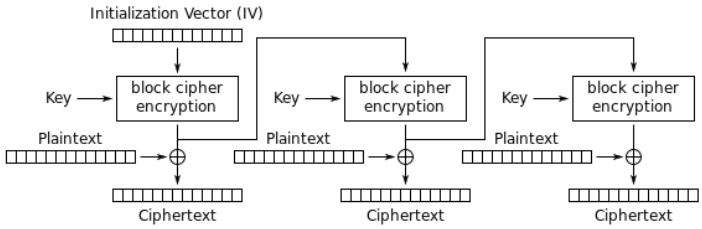
- If we encrypt a string of blocks $x_1, \ldots, xt$ once with a first IV and a second time with a different IV
  - Two resulting ciphertext sequences look completely unrelated to each other i.e. **probabilistic** encryption
- Note that we do not have to keep the IV secret

- Make use of a 'nonce' ie. use it only once

Output Feedback Mode (OFB)

- In OFB mode a block cipher is used to build a **stream cipher** encryption scheme
  - Keystream is not generated bitwise but in a blockwise fashion
- The output of the cipher produces b keystream bits, where b is the width of the block cipher used

— we can then encrypt b plaintext bits using the XOR operation



Output Feedback (OFB) mode encryption

XOR is super super fast in hardware

Very practical

Yes

Much computer science

25

OFB Mode

- Encryption and decryption are exactly the **same** operation

Definition 5.1.3 Output feedback mode (OFB)
Let $e()$ be a block cipher of block size b; let x_i, y_i and s_i be bit strings of length b; and IV be a nonce of length b.
Encryption (first block): $s_1 = e_k(IV)$ *and* $y_1 = s_1 \oplus x_1$
Encryption (general block): $s_i = e_k(s_{i-1})$ *and* $y_i = s_i \oplus x_i, \quad i \geq 2$
Decryption (first block): $s_1 = e_k(IV)$ *and* $x_1 = s_1 \oplus y_1$
Decryption (general block): $s_i = e_k(s_{i-1})$ *and* $x_i = s_i \oplus y_i, \quad i \geq 2$

- The OFB mode forms a *synchronous stream cipher* as the computations are **independent** of the plaintext
  - One can precompute several blocks 'Si' of keystream material

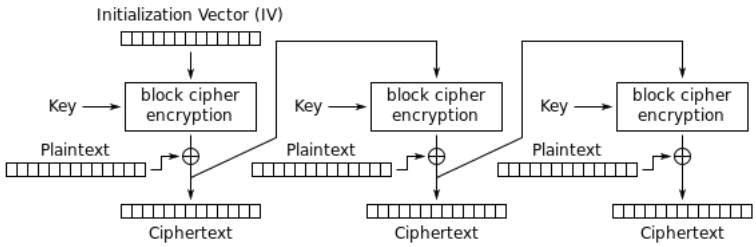
Left side is all the same, ie encryption and decryption are the same
Really good for code maintenance

25

26

Cipher Feedback Mode (CFB)

- It is similar to the OFB mode but instead of feeding back the output of the block cipher, the ciphertext is fed back



Cipher Feedback (CFB) mode encryption

- CFB mode is an example of an *asynchronous stream cipher*
  - since the stream cipher output is also a function of the ciphertext

Cannot precompute

26

CFB Mode

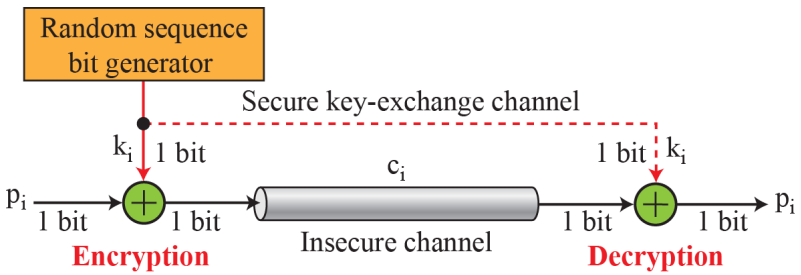
- A variant of the CFB mode can be used in situations where *short* plaintext blocks are to be encrypted
  - e.g. encryption of the link between a (remote) keyboard and a computer

Definition 5.1.4 Cipher feedback mode (CFB)
Let $e()$ be a block cipher of block size b; let x_i and y_i be bit strings of length b; and IV be a nonce of length b.
Encryption (first block): $y_1 = e_k(IV) \oplus x_1$
Encryption (general block): $y_i = e_k(y_{i-1}) \oplus x_i, \quad i \geq 2$
Decryption (first block): $x_1 = e_k(IV) \oplus y_1$
Decryption (general block): $x_i = e_k(y_{i-1}) \oplus y_i, \quad i \geq 2$

- Plaintexts generated by the keyboard are typically only 1 byte long, e.g., an ASCII character
  - **In this case, only 8 bits of the keystream are used for encryption and the ciphertext also only consists of 1 byte**

The Vernam Cipher

- Simplest and most secure stream cipher is called the One-time Pad
- Chooses a key stream (k) that is randomly chosen for each encipherment – makes use of XOR operator
  - **k >= p, ie. key longer than data**



Hand someone the encrypted material (Manually)

29

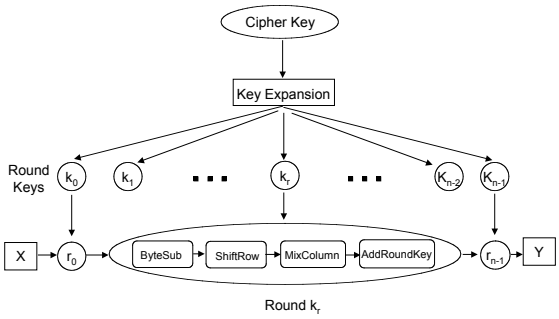
Advanced Encryption Standard (AES)

- In 1997 NIST announced a call for proposals to develop a new Advanced Encryption Standard
  - After a long vetting process five algorithms were short listed
    - Rijndael (Rhine-doll) was eventually chosen as the new AES
- Symmetric cipher with variable key and block sizes of 128, 192 and 256 bits **<- Can't precompute that many keys**
  - **— Most common mode is 128 bit key and block size**
  - Support for fast encryption and decryption in s/w - 700 Mbps **Minimum speed in software**
    - **— Can be implemented efficiently in smartcards (8-bit microcontroller at the time)**
- A device that could check a 10^{18} AES keys/s would in theory require about 3×10^{51} years to exhaust the 256-bit key space
  - Brute force decryption taking 1 sec on DES, takes 149 trillion years for AES

29

30

AES



- The cipher consists of between 10 or 14 rounds (N_r)
  - **— Depending on key length (Nk) and block length (Nb)**
- A plaintext block X undergoes n rounds of operations to produce an output block Y
  - Each operation is based on the value of the n^{th} round key
- Round keys are derived from the Cipher Key by first expanding the key
  - Then selecting parts of the expanded key for each round

30

31

Asymmetric-Key Cryptosystems

- Public key cryptography was invented by Diffie and Hellman in 1976
  - Solves the key management problem associated with symmetric key cryptosystems
- In public key cryptography each person generates a pair of keys
  - The public key and the private key
- Public key is published and widely distributed
  - While the private key is kept secret
- Examples of public-key cryptographic algorithms are
  - RSA, Diffie-Hellman, ElGamal, ECC

31

32

Properties of Asymmetric-Key Cryptosystems

- Must be computationally easy to encipher or decipher a message given the appropriate key

- Must be computationally infeasible to derive the private key from the public key

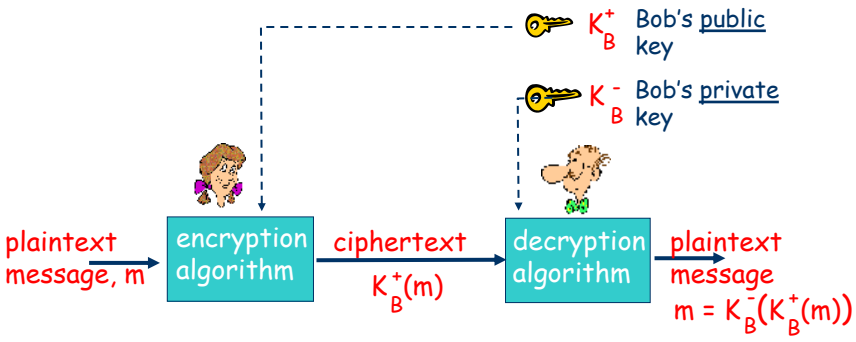
- Need for exchanging secret keys is eliminated
  - All secure communications now only involves public keys

32

33

Public-Key Cryptography

- Each user in a public-key system selects his own private key (k^-) and his own public key (k^+)



- When user Alice wants to send an encrypted message to Bob
  - she looks up his public key (kB+) in a public directory
    - Or obtains it by some other means

<— this mechanism stands no matter how keys are generated

33

34

RSA

- De-facto standard algorithm for implementing asymmetric-key cryptography
  - Named after Rivest, Shamir and Adleman who developed it in 1978 at MIT
- It's security is based on the difficulty of factoring very large numbers
  - one-way "Trapdoor Function"
- Example: Prime Factoring
  - Easy to calculate product of 2 large prime numbers
  - difficult to calculate the prime factors from product

34

35

Modular Arithmetic I

- Most number sets we are used to are infinite e.g. set of real numbers
  — However most cryptographic algorithms are based on arithmetic with a finite set of numbers
- "Clock Arithmetic"
  - Converting between 24-hr and 12-hr systems is easy
    - 13:00 in the 24-hr system is one o'clock in the 12-hr system
      — 13 mod 12 = 1
- Let $a, b, n \in \mathbb{Z}$ (where $\mathbb{Z}$ is the set of all integers) and $n > 0$
  - $a \equiv b \pmod{n}$

congruent operator ——————>

- a is said to be **congruent** to b mod n, if n divides $a - b$
  — 13 == 1 (mod 12) or 12 | (13-1)

35

36

Modular Arithmetic II

- It is always possible to write $a \in \mathbb{Z}$ such that
  - $a = q \cdot n + r \quad 0 \leq r < n$
- Since $a - r = q \cdot n$ (n divides $a - r$) we can now write
  - $a \equiv r \pmod{n}$
- Example : Let a = 42, n = 9
  — 42 = 4 * 9 + 6
  — 42 == 6(mod 9)
- Q: -11 ≡ x (mod 7)
  — -11 == 3(mod 7)
  — -11-3 = -2*7

36

37

Euler's Phi Function

- Important to know if there is a solution to the equation
  - $a \cdot x \equiv b \pmod{n}$
- If $\gcd(a, n) = 1$
  - Exactly one solution
    - $7 \cdot x \equiv 1 \pmod{143}$
      - x = ?

<— First interested to know that there is a solution

- If $g = \gcd(a, n) \neq 1$ and $\gcd(a, n)$ divides b
  - There are g solutions
    - $11 \cdot x \equiv 22 \pmod{143}$
      - g = ?
- Otherwise there are no solutions
  - $11 \cdot x \equiv 3 \pmod{143}$

37

38

Euler's ϕ Function

- In general let m have the following canonical factorization

$$m = {p_1}^{e_1} \cdot {p_2}^{e_2} \cdot \ldots . {p_n}^{e_n}$$

  - where p_i are distinct primes and e_i are positive integers

$$\Phi(m) = \prod_{i=1}^{n} (p_i^{e_i} - p_i^{e_i - 1})$$

- Exercise: Calculate $\phi(240)$?
  - Hint m = 240 = 16 * 15
    - = 2^4 * 3 * 5

(2^4 - 2^3) * (3^1 - 3^0) * (5^1 - 5^0) = 64

38

39

Euler's Totient Function $\phi(n)$

- No. of positive integers less than n and **relatively prime** to n
  - Relatively prime means $\gcd(a, n) = 1$
- Example: $\phi(10) = 4$
  — 1, 3, 7, 9 are relatively prime to 10
- Example: $\phi(21) = 12$
  — 1, 2, 3, 4, 5, 8, 10, 11, 13, 16, 17, 19, 20 are relatively prime to 21
- Q: What is $\phi(7)$ and $\phi(11)$?

  7 -> 6, 11 -> 10 ———they are prime

39

40

Multiplicative Inverse

- The integers modulo n, denoted $\mathbb{Z}_n$ is the set of integers {0, 1, 2, … $n-1$}
  - Addition, subtraction and multiplication are performed modulo *n*
    — Zn is referred to as an Integer Ring
- $\mathbb{Z}_{25}$ = {0, 1, 2, …, 24}
  - 13 + 16 = 4
    29 == 4(mod 25)
- The multiplicative inverse (a^{-1}) of a modulo n is an integer $x \in \mathbb{Z}_n$ such that
  - $a \cdot x \equiv 1 \pmod{n}$
- The multiplicative inverse only exits for an element $a \in \mathbb{Z}_n$ iff
  gcd(a,n) = 1

Q: Does the multiplicative inverse of 15 exist in $\mathbb{Z}_{26}$?

40

41

Extended Euclidean Algorithm (EEA)

- Division of a by b modulo n is a product of a and b^{-1} modulo n
  - bla is equivalent to $a \cdot b^{-1} \pmod{n}$
- Q: What is 4^{-1} modulo 11?
  - $x \equiv 4^{-1} \pmod{11}$

    4 * x == 1 (mod 11)
- The modular multiplicative inverse of a modulo n can be found with the Extended Euclidean Algorithm
  - $s \cdot r_0 + t \cdot r_1 = \gcd(r_0, r_1)$
  - $s \cdot r_0 + t \cdot r_1 = 1$
  - $s \cdot 0 + t \cdot r_1 \equiv 1 \bmod r_0$
  - $t \equiv r_1^{-1} \bmod r_0$

Exercise: Compute 15^{-1} mod 26?

- $\gcd(11, 4) = 1$
  - $11 = 2 \cdot 4 + 3$
  - $4 = 1 \cdot 3 + 1$
  - $3 = 3 \cdot 1 + 0$
- Back substitution
  - $1 = 4 - 1 \cdot 3$
  - =4 -1 * (11 - 2*4)
  - =3 * 4 - 1 * 11
  - =(-1) * 11 + (3) * 4
  - t = 3

41

42

Fermat's Little Theorem

- Let a be an integer and p be a prime, then
  - $a^p \equiv a \pmod{p}$

    a^(p-1) == 1 (mod p)
  - $a \cdot a^{p-2} \equiv 1 \pmod{p}$
  - Thus we have a way for inverting an integer a modulo a prime
- Compute 4^{-1} mod 11

42

43

RSA Algorithm

- Choose two large distinct primes p and q
- Compute the product (modulus)
  - $n = p \cdot q$

    phi(n) = (p-1) * (q-1)
- Randomly choose an encryption key e, less than n that has no common factors with $\phi(n)$
  - e and $\phi(n)$ are relatively prime
    - e is invertible iff gcd(e, phi(n))= 1
- Finally compute the decryption key, d such that
  - $e \cdot d \equiv 1 \pmod{\phi(n)}$
  - $d \equiv e^{-1} \bmod \phi(n)$

    d == e^-1 mod(p-1)(q-1)

43

44

RSA Usage

- The numbers e and n are the public key
  - the number d is the private key
- Break the plaintext message into a number of blocks
  - Represent each block as an integer
- Encryption
  - CiphertextBlock = $(\text{PlaintextBlock})^e \bmod n$
- Decryption
  - PlaintextBlock = $(\text{CiphertextBlock})^d \bmod n$
- Key Sizes
  - 1024, 2048, 3072, 7680 bits
    - Recommended keysize from 2015 is 3072 bits

44

45

RSA with Workable Numbers

- Let p = 3 and q = 11
- Using $n = p \cdot q = 33$

  phi(n) = (p-1)*(q-1) = 20
- Choose e = 3 (e and $\phi(n)$ have no common factors)
- Solving $e \cdot d \equiv 1 \bmod 20$ and $d < 20$
  - $d \equiv e^{-1} \bmod 20$
  - d = 7

    7 * 3 = 21 = 20 * 1 + 1 (EEA)

45

46

RSA Example

Plaintext (P)			Ciphertext (C)		After decryption	
Symbolic	Numeric	P^3	$P^3 \pmod{33}$	C^7	$C^7 \pmod{33}$	Symbolic
S	19	6859	28	13492928512	19	S
U	21	9261	21	1801088541	21	U
Z	26	17576	20	1280000000	26	Z
A	01	1	1	1	01	A
N	14	2744	5	78125	14	N
N	14	2744	5	78125	14	N
E	05	125	26	8031810176	05	E
		Sender's computation		Receiver's computation		

- Since the primes chosen in this example are small, P must be less than 33
  - Each block can only contain a single character
- If p and $q \approx 2^{512}$, we would have $n \approx 2^{1024}$
  - each block would be up to 1024 bits or 128 bytes

46

47

Practical RSA

$$
\begin{aligned}
p &= E0DFD2C2A288ACEBC705EFAB30E4447541A8C5A47A37185C5A9 \\
&\quad CB98389CE4DE19199AA3069B404FD98C801568CB9170EB712BF \\
&\quad 10B4955CE9C9DC8CE6855C6123_h \\
q &= EBE0FCF21866FD9A9F0D72F7994875A8D92E67AEE4B515136B2 \\
&\quad A778A8048B149828AEA30BD0BA34B977982A3D42168F594CA99 \\
&\quad F3981DDABFAB2369F229640115_h \\
n &= CF33188211FDF6052BDBB1A37235E0ABB5978A45C71FD381A91 \\
&\quad AD12FC76DA0544C47568AC83D855D47CA8D8A779579AB72E635 \\
&\quad D0B0AAAC22D28341E998E90F82122A2C06090F43A37E0203C2B \\
&\quad 72E401FD06890EC8EAD4F07E686E906F01B2468AE7B30CBD670 \\
&\quad 255C1FEDE1A2762CF4392C0759499CC0ABECFF008728D9A11ADF_h \\
e &= 40B028E1E4CCF07537643101FF72444A0BE1D7682F1EDB553E3 \\
&\quad AB4F6DD8293CA1945DB12D796AE9244D60565C2EB692A89B888 \\
&\quad 1D58D278562ED60066DD8211E67315CF8985716720612040 5B0 \\
&\quad 8B54D10D4EC4ED4253C75FA74098FE3F7FB751FF5121353C554 \\
&\quad 391E114C85B56A9725E9BD5685D6C9C7EED8EE442366353DC39_h \\
d &= C21A93EE751A8D4FBFD77285D79D6768C58EBF283743D2889A3 \\
&\quad 95F266C78F4A28E86F545960C2CE01EB8AD5246905163B28D0B \\
&\quad 8BAABB959CC03F4EC499186168AE9ED6D88058898907E61C7CC \\
&\quad CC584D65D801CFE32DFC983707F87F5AA6AE4B9E77B9CE630E2 \\
&\quad C0DF05841B5E4984D059A35D7270D500514891F7B77B804BED81_h
\end{aligned}
$$

47

48

How long is a 2048-bit RSA key?

- One bit can be 0 (zero) or 1 (one)
  - So 2048 bits gives 2^{2048} distinct numbers
- A decimal digit has ten possible values 0, 1, 2, ... , 9
  - To find the number of decimal digits to make 2^{2048} distinct number we need to solve

    2^2048 = 10^n
- Take a logarithm (base 10) on both sides to get
  - $n \log_{10}(10) = 2048 \log_{10}(2)$
  - n = 2048 * 0.30102999566
  - n = 616.5 or 617 digits

48

Fast Exponentiation

- A straightforward way of exponentiation is like this:

$$x \xrightarrow{SQ} x^2 \xrightarrow{MUL} x^3 \xrightarrow{MUL} x^4 \xrightarrow{MUL} x^5 \cdots$$

  – Where *SQ* denotes squaring and *MUL* multiplication

- How many multiplications are required to compute x^8?

  7

- Alternatively we can compute

  x -> x^2 -> x^4 -> x^8

- What about x^{26}?

  x -> x^2 -> x^3 -> x^6 -> x^12 -> x^13 -> x^26

49

Square-and-Multiply Algorithm

- Based on scanning the bit of the exponent from the left (the most significant bit) to the right (the least significant bit)
- In every iteration
  – i.e., for every exponent bit the current result is squared
- Iff the currently scanned exponent bit has the value 1

  A multiplication of the current result by x is executed following the squaring
- Example: We consider the exponentiation x^{26}. For the square-and-multiply algorithm, the binary representation of the exponent is:

  x^26 = x^(11010_2) = x^(h_4 h_3 h_2 h_1 h)

50

Square-and-Multiply Algorithm II

Step		
#0	$x = x^{1_2}$	inital setting, bit processed: $h_4 = 1$
#1*a*	$(x^1)^2 = x^2 = x^{10_2}$	SQ, bit processed: h_3
#1*b*	$x^2 \cdot x = x^3 = x^{10_2} x^{1_2} = x^{11_2}$	MUL, since $h_3 = 1$
#2*a*	$(x^3)^2 = x^6 = (x^{11_2})^2 = x^{110_2}$	SQ, bit processed: h_2
#2*b*		no MUL, since $h_2 = 0$
#3*a*	$(x^6)^2 = x^{12} = (x^{110_2})^2 = x^{1100_2}$	SQ, bit processed: h_1
#3*b*	$x^{12} \cdot x = x^{13} = x^{1100_2} x^{1_2} = x^{1101_2}$	MUL, since $h_1 = 1$
#4*a*	$(x^{13})^2 = x^{26} = (x^{1101_2})^2 = x^{11010_2}$	SQ, bit processed: h_0
#4*b*		no MUL, since $h_0 = 0$

- Modulo reduction is applied after each multiplication and squaring operation
  - In order to keep intermediate results small

51

Short Public Exponents

- In practice the public key e can be chosen to be a very small value
  - e=3, e=17, e=2^16 + 1
- The resulting complexities when using these public keys are given below

Public key e	e as binary string	#MUL + #SQ
3	11_2	3
17	10001_2	5
$2^{16}+1$	10000000000000001_2	17

- On average$1.5t$ multiplications and squarings are required for exponents of full length
  - $t + 1$ is the bit length of the RSA modulus n

52

53

Short Public Exponents II

- We note that the short public exponents listed above have a low Hamming weight
  - i.e. the number of ones in the binary representation

— Results in a low number of operations when performing an exponentiation

- An important consequence of the use of short public exponents is

— encryption of a msg or verification of a RSA signature is a very fast operation

- Unfortunately there is no such easy way to accelerate RSA when the private key d is involved

53

54

RSA Decryption

- One limitation of RSA is that it requires doing a lot of computations
- Suppose Alice sets up a RSA system with n = 3293 & e = 35
- Bob wants to send the value m = 153, so he computes

  153^35 == 2494 mod 3293
- Alice's secret decryption exponent d = 2987
  - So she must evaluate $2494^{2987} \bmod 3293$

    That is a lot of work — think 200+ digit numbers
- Is there an easier way?
  - An old piece of mathematics known as the Chinese Remainder Theorem (CRT) will help us here

54

55

Chinese Remainder Theorem

- CRT states that if we have a number congruent to a product
$$x \equiv a \bmod (p \cdot q)$$
  - It is also congruent to
$$x \equiv a \pmod{p} \text{ and } x \equiv a \pmod{q}$$
- Our goal is to perform the exponentiation $x^d \bmod n$ efficiently
- We note that the party that posses the private key
  - Also knows that primes p and q (which are co-prime)
- Basic idea of CRT is that rather than doing arithmetic with one "long" modulus n

55

56

CRT

- Reduce the base element x modulo the two factors p and q of the modulus n
$$x_p \equiv x \bmod p$$
$$x_q \equiv x \bmod q$$
- With the reduced version of x we perform the following two exponentiations
$$y_p = x_p^{d_p} \bmod p$$
$$y_q = x_q^{d_q} \bmod q$$
- where the two new exponents are given by
$$d_p \equiv d \bmod (p-1)$$
$$d_q \equiv d \bmod (q-1)$$
- Finally we calculate
$$y \equiv [qc_p]y_p + [pc_q]y_q \bmod n$$
- where the coefficients c_p and c_q are computed as
$$c_p \equiv q^{-1} \bmod p \text{ and } c_q \equiv p^{-1} \bmod q$$

56

Example - RSA Decryption with CRT

- Let the RSA parameters be given by:

$$p = 11, \quad q = 13, \quad n = p \cdot q = 143, \quad e = 7$$
$$\text{and } d \equiv e^{-1} \equiv 103 \bmod 120$$

- We now compute the RSA decryption for x = 15 i.e. $15^{103} \bmod 143$

$$x_p \equiv 15 \equiv 4 \bmod 11$$

$$d_p \equiv 103 \equiv 3 \bmod 10$$

$$y_p = 4^3 = 64 \equiv 9 \bmod 11$$

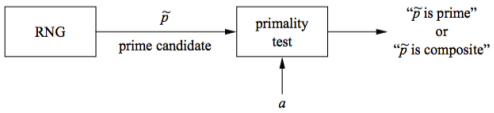
$$c_p = 13^{-1} \equiv 2^{-1} \equiv 6 \bmod 11 \text{ and } c_q = 11^{-1} \equiv 6 \bmod 13$$

$$x \equiv [13 \cdot 6]9 + [11 \cdot 6]11 \bmod 143$$

57

Finding Large Primes

- General approach is to generate integers at random which are then checked for primality



- Chance that a randomly picked integer $\tilde{p}$ is a prime is $\frac{1}{\ln(\tilde{p})}$

e.g. For RSA with a 1024-bit modulus n, the primes p and q each should have length 512 bits i.e. $p, q \approx 2^{512}$

$$P(\tilde{p} \text{ is a prime}) \approx \frac{2}{\ln(2^{512})} = \frac{2}{512 \ln(2)} \approx \frac{1}{177}$$

58

59

Primality Tests

- A simple primality test can be based on Fermat's Little Theorem

- However there are certain composite numbers (aka Charmichael numbers) which may fulfill the above condition
  - In order to detect them the algorithm runs s times with different values of $a \in \{2, \ldots, p - 2\}$

Exercise: Is 221 a prime number?

59

60

Miller-Rabin Primality Test

- Given the decomposition of an odd prime candidate $\tilde{p}$

$$\tilde{p} - 1 = 2^u r$$

  - where r is odd

- If we can find an integer a such that

$$a^r \not\equiv 1 \bmod \tilde{p} \text{ and } a^{r2^j} \not\equiv -1 \bmod \tilde{p}$$

  - for all $j = \{0, 1, \ldots, u - 1\}$, then $\tilde{p}$ is probably a prime

60

61

Miller-Rabin Primality Test II

Example:

- If $\tilde{p}$ = 13 then $\tilde{p}$ - 1 = 4 · 3
  - $a^3 \not\equiv 1 \bmod \tilde{p}$ or $a^3 \not\equiv -1 \bmod \tilde{p}$ or $a^6 \not\equiv -1 \bmod \tilde{p}$ for each a from 1 to 12

Exercise: Is 561 a prime?

- If $\tilde{p}$ = 561 then $\tilde{p}$ - 1 = 16 · 35

61

62

RSA Malleability

- A crypto scheme is said to be *malleable*
  - If an attacker is capable of transforming the ciphertext into another ciphertext
    - Which leads to a known transformation of the plaintext
- Easily achieved in the case of RSA if the attacker replaces the ciphertext y by $s^e y$, where s is some integer
  - $y \equiv x^e \bmod n$
- Receiver computes

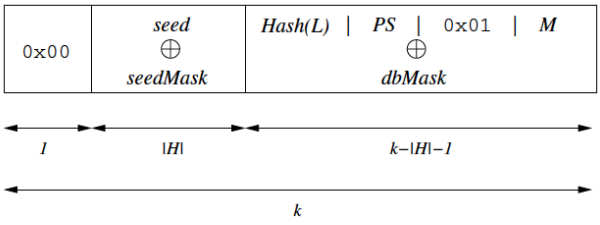
  - (s^e y)^d == s^(ed) x^(ed) == sx mod n
- If x were an amount of money which is to be transferred

  — by choosing s = 2 Trudy could exactly double the amount in a way that goes undetected by the receiver

62

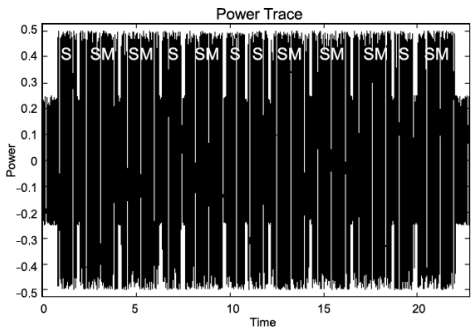
RSA in Practice

- RSA encryption is *deterministic*
  - for a specific key, a particular plaintext is always mapped to a particular ciphertext
- In practice RSA has to be used with a padding scheme
  - Optimal Asymmetric Encryption Padding (OAEP) for padding RSA messages are specified
    - Standardized in Public Key Cryptography Standard #1 (PKCS #1)



RSA Side Channel Attacks

- Exploit information about the private key which is leaked through physical channels such as the
  - power consumption or
  - Timing behaviour



- The figure shows the electric current drawn by the processor over time
- Our goal is to extract the private key d

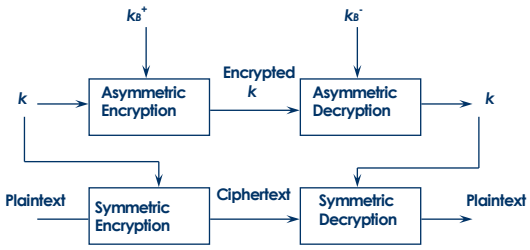
RSA Side Channel Attacks II

- We see that there are high activity intervals which are short and others which are longer
- Explained by the square-and-multiply algorithm
  - If an exponent bit has the value 0, only a *SQ* is performed
  - if an exponent bit has the value 1, a SQ + Mul is performed
- A long period of activity corresponds to the bit value 1 of the secret key
  - Short period to a key bit with value 0

operations:	*S*	*SM*	*SM*	*S*	*SM*	*S*	*S*	*SM*	*SM*	*SM*	*S*	*SM*
private key:	0	1	1	0	1	0	0	1	1	1	0	1

Hybrid Schemes

- Asymmetric-key algorithms are not a replacement for symmetric-key algorithms such as DES or AES
  - Rather they supplement AES or any other fast bulk encryption cipher



- The above example shows
  - How we can use a public key algorithm to securely transfer a session key (k), and
  - Use session key for bulk encryption/decryption

An Important Property of RSA

The following *commutative* property will be very useful later

$$K_B^-(K_B^+(m)) = m = K_B^+(K_B^-(m))$$

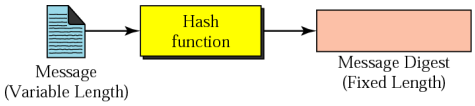
use public key first, followed by private key

use private key first, followed by public key

result is the same!

67

Msg Auth & Integrity



- Bob receives a msg from Alice, wants to ensure
  - Msg originally came from Alice - Authentication
  - Msg not changed since sent by Alice - Integrity

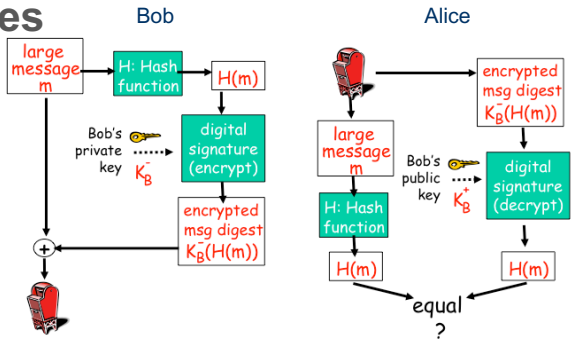
Message Digest/Cryptographic Hash

- A message digest is a strong digital fingerprint of a message
- Takes input m, produces fixed length value $H(m)$
  - 128/160/256 bits
- Computationally infeasible to find two different messages x and y such that
  - H(x) == H(y)
- Examples of message digest algorithms are
  - MD2, MD4, MD5, SHA-1 and SHA-2

68

69

Digital Signatures

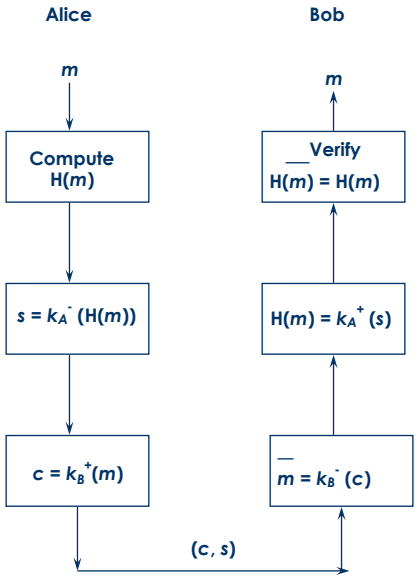


- Cryptographic technique analogous to hand-written signatures
- Sender (Bob) digitally signs the document establishing he is the document owner/creator
- Recipient (Alice) can prove to someone that Bob, and no one else (including Alice) must have signed document
  - verifiable and non-forgeable (non-repudiation)

69

70

Enveloped and Signed Data



70

Key Management

- When Alice obtains Bob's public-key (from a website, e-mail etc.), how does she know it is Bob's public key, not Trudy's?
- Public key cryptography is based on the idea that
  - Keep one component secret and publish the other component (public key)
- Other users on the network
  - Must be able to retrieve this public key and associate the user's identity with it
- One way to form this association is to

71

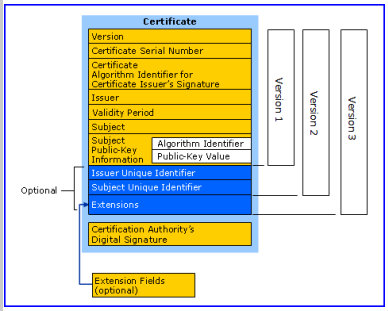
X.509 Certificates

- The TTP will construct a message referred to as a Certificate

- The cert contains a number of fields
- Assumes that every user in the system is equipped with the public-key of the TTP
  - Allows one to verify the digital signature on the certificate
    - Guaranteeing that the public key is associated with the named user

72

Example X.509 Certificate

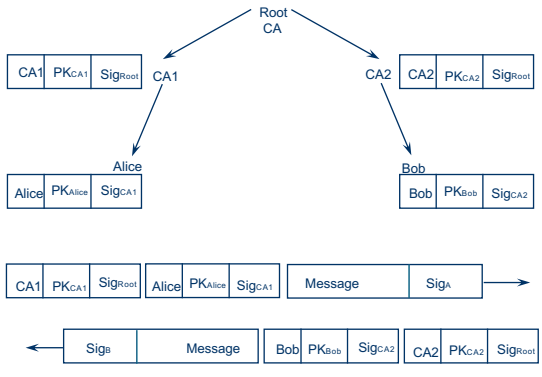



```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            ab:87:45:a4:28:0b:d5:c8
        Signature Algorithm: sha1WithRSAEncryption
        Issuer:
            countryName               = DA
            stateOrProvinceName       = Kofisa
            localityName              = Bersoo
            organizationName          = Goaalsd
            commonName                = jfollsd.com
            emailAddress              = sdf@sdf.com
        Validity
            Not Before: Jul 20 10:40:44 2009 GMT
            Not After : Jul 18 10:40:44 2019 GMT
        Subject:
            countryName               = DA
            stateOrProvinceName       = Kofisa
            localityName              = Bersoo
            organizationName          = Goaalsd
            commonName                = jfollsd.com
            emailAddress              = sdf@sdf.com
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (1024 bit)
                Modulus (1024 bit):
                    00:c6:45:a9:ad:91:11:13:17:8a:60:5d:e2:38:87:
                    9c:37:0d:ab:eb:fc:3b:45:64:db:18:0e:dc:ae:94:
                    34:77:21:c5:49:8c:00:f6:51:18:4b:9a:d0:a9:ef:
                    4d:8c:a3:3b:cc:f8:0c:8a:a0:90:9f:97:74:ee:9f:
                    cb:26:b9:ce:4f:00:e4:09:3f:79:67:77:f9:43:a9:
                    af:00:2e:54:6e:12:dc:89:4c:72:ec:66:6d:40:28:
                    85:77:a0:1b:3d:bb:8a:d3:c3:10:ad:6a:dd:42:d9:
                    86:42:bd:a7:29:7b:e0:06:0f:7b:2d:97:5c:e5:9f:
                    86:6c:69:c4:c5:03:c7:9e:31
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Subject Key Identifier:
                FA:55:53:A3:3D:4C:70:DE:F9:D6:D9:93:33:85:F0:4C:0F:6B:A2:91
            X509v3 Authority Key Identifier:
                keyid:FA:55:53:A3:3D:4C:70:DE:F9:D6:D9:93:33:85:F0:4C:0F:6B:A2:91
                DirName:/C=DA/ST=Kofisa/L=Bersoo/O=Goaalsd/CN=jfollsd.com/emailAddress=sdf@sdf.com
                serial:AB:87:45:A4:28:0B:D5:C8

            X509v3 Basic Constraints:
                CA:TRUE
    Signature Algorithm: sha1WithRSAEncryption
        6c:95:14:65:18:e3:24:5f:12:c3:8f:16:dc:0d:18:fb:3a:5e:
        1b:e2:eb:ed:63:ef:2d:d0:d2:9d:43:25:e8:1f:5e:d2:7f:96:
        34:27:f2:ad:5b:21:fc:88:bd:43:ab:1f:7d:36:17:f4:75:d5:
        9a:41:74:36:a8:71:21:1f:2c:04:a0:e3:6d:24:42:c9:80:e6:
        88:e2:3c:ec:d2:3e:ee:a2:c0:4e:4b:13:f8:11:e5:29:ad:79:
        f7:79:25:c1:a4:2f:3e:46:b7:85:fc:e9:97:3e:8b:9a:f5:0e:
        cf:dd:d7:1f:e2:c5:12:f8:0e:d3:80:4f:d4:e5:31:5e:ec:59:
        28:79
```


Certification Hierarchy

- TTPs that issue certificates are referred to as certification authorities (CAs)
- The root CA issues certificates only to other CAs

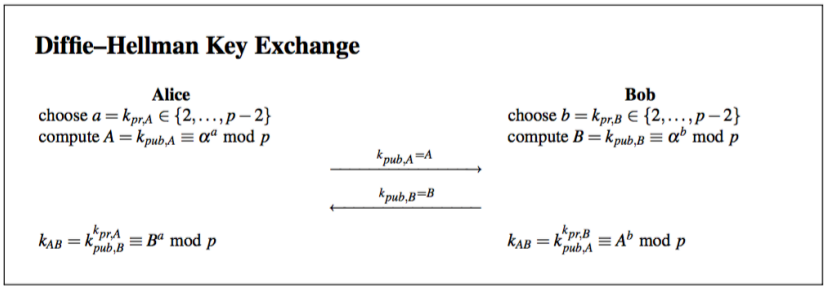


Diffie-Hellman Key Exchange (DHKE)

- A protocol that allows strangers to establish a shared symmetric key without them having to meet
- The basic idea behind DHKE is
  - That exponentiation in the multiplicative group $\mathbb{Z}_p^*$ (p is a prime) is a **one-way function**, and
  - That exponentiation is **commutative**
- The value $k \equiv (\alpha^x)^y \equiv (\alpha^y)^x \bmod p$ is a "joint secret"
  - Can be used as a *session key* between the two parties

DHKE

- Securely choose the *domain parameters*
  - Choose a large prime p
  - Choose an integer $\alpha \in \{2, 3, \ldots, p - 2\}$
  - Publish p and α



- Exercise: Generate a DH key using the domain parameters p = 29 and α = 2. Assume that a = 5 and b = 12

77

Groups and Fields

- A group is set with **one** operation and the corresponding inverse operation
  - If the operation is *addition*, the inverse operation is subtraction
  - If the operation is *multiplication*, the inverse operation is division (or multiplication with the inverse element)
- A *finite* field *(*aka *Galois* field) is a set with a finite number of elements in which we can

77

78

Group

- *A* group is a *set of elements* G together with an *operation* $\circ$ which combines two elements of G and has the following properties
  - The group operation $\circ$ is *closed*
  - The group operation is *associative*
    - $a \circ (b \circ c) = (a \circ b) \circ c \quad \forall\, a, b, c \in G$
  - There is an element $e \in G$, called the *identity element*
    - $a \circ e = e \circ a = a \quad \forall\, a, e \in G$
      - 0 is the additive identity while 1 is the multiplicative identity
  - $\forall\, a \in G$ there exists an element $a^{-1} \in G$, called the *inverse* of a
  - A group G is *commutative (*abelian*)*
    - If $a \circ b = b \circ a \quad \forall\, a, b \in G$

78

79

Group II

- Example: The set of integers $\mathbb{Z}_n$ = {0, 1, . . ., n - 1} and the operation addition modulo n form a group with the neutral element 0
  - Every element *a* has an inverse $-a$ such that $a + (-a) \equiv 0 \bmod n$
  - Note that this set does not form a group with the operation multiplication because most elements *a* do not have an inverse
- In order to have all **four** basic arithmetic operations (i.e., addition, subtraction, multiplication, division) in one structure
  - Need a set which contains an *additive* and a *multiplicative* group

79

80

Field

- A field F is a set of elements with the following properties
  - All elements of F form an *additive group* with the group operation "+" and the identity element *0*
  - All elements of F except 0 form a *multiplicative group* with the group operation "$\cdot$" and the identity element *1*
  - When the two group operations are mixed, the *distributivity* law holds
- The set $\mathbb{R}$ of real numbers is a field with
  - Neutral element 0 for the additive group
  - Every real number a has an additive inverse $-a$

80

81

Finite Fields

Theorem 4.3.1 *A field with order m only exists if m is a prime power, i.e., $m = p^n$, for some positive integer n and prime integer p. p is called the* characteristic *of the finite field.*

- The theorem implies that there are finite fields with 11 elements or 81 elements
- Is there a finite field with 256 elements?
- Is there a finite field with 12 elements?

81

82

Prime Fields

- The most intuitive examples of finite fields are fields of prime order
- The set $\mathbb{Z}_p$ (where p is a prime) is denoted as *GF*(p) and is referred to as a prime field
  - aka *Galois* field with a prime number of elements
- Elements of the field *GF*(p) can be represented by integers 0,1, . . . , p - 1
  - All nonzero elements of *GF*(p) have an inverse
  - Arithmetic in *GF*(p) is done modulo p
- Two operations of the field are *modular integer addition* and *integer multiplication* modulo p

82

83

Finite Groups

- A group $(G, \circ)$ is finite if it has a finite number of elements
  - e.g. $(\mathbb{Z}_n, +)$, $(\mathbb{Z}_n, \cdot)$
- We denote the *cardinality* of the group G by $|G|$
  - $\mathbb{Z}_n = \{0, 1, 2, \ldots, n - 1\}$
  - $\mathbb{Z}_n^*$ consists of integers $i = \{1, . , n - 1\}$ for which gcd $(i, n) = 1$
- The order $ord(a)$ of an element a of a group $(G, \circ)$ is the smallest positive integer k such that

83

84

Finite Groups II

- Example: We try to determine the order of $a = 3$ in $\mathbb{Z}_{11}^*$
  - We keep computing powers of a until we obtain the identity element 1
- $a^1 = 3$
- Q. What is a^6 and a^7 ?

84

Cyclic Groups

- A group G which contains an element α with maximum order i.e. $ord(\alpha) = |G|$ is said to be *cyclic*
- Elements with maximum order are called
- Exercise: Check if $a = 2$ is primitive root of $\mathbb{Z}_5^*$
- $ord(a) = 4 = |\mathbb{Z}_5^*|$
- For every prime p, $(\mathbb{Z}_p^*, \cdot)$ is a *commutative finite cyclic* group
  - i.e. the multiplicative group of every prime field is cyclic

85

Cyclic Groups II

Theorem 8.2.3
Let G be a finite group. Then for every a ∈ G it holds that:

1. $a^{|G|} = 1$
2. *ord(a) divides* $|G|$

- The second property says that in a cyclic group only element orders that divide the group cardinality exist
- Q. What are the element orders for the group $\mathbb{Z}_{11}^*$?
  - The cardinality $|\mathbb{Z}_{11}^*| = 10$

$$\begin{aligned} \text{ord}(1) &= 1 & \text{ord}(6) &= 10 \\ \text{ord}(2) &= 10 & \text{ord}(7) &= 10 \\ \text{ord}(3) &= 5 & \text{ord}(8) &= 10 \\ \text{ord}(4) &= 5 & \text{ord}(9) &= 5 \\ \text{ord}(5) &= 5 & \text{ord}(10) &= 2 \end{aligned}$$

86

Cyclic Groups III

Theorem 8.2.4 *Let G be a finite cyclic group. Then it holds that*
1. *The number of primitive elements of G is* $\Phi(|G|)$.
2. *If* $|G|$ *is prime, then all elements* $a \neq 1 \in G$ *are primitive.*

Exercise: Find the number of primitive roots in $\mathbb{Z}_{11}^*$

Q. What are the primitive roots of $\mathbb{Z}_{11}^*$?

87

Finding a Primitive Root

- To test that *a* is a primitive root of $\mathbb{Z}_p^*$
- Let $s = \phi(p)$
  - If p is prime
- Determine all the prime factors of s
  - $s = p_1, \ldots, p_k$
- Finally, calculate a^{s/p_i} mod p for all $i = 1 \ldots k$
  - If you find 1 among residuals then it is NOT a primitive root
    - Otherwise it is a primitive root

88

89

Finding a Primitive Root - Example

- Let us find the lowest primitive root of 761

- The powers to test are
  - 760/2=380, 760/5=152 and 760/19=40 (just 3 instead of testing all of them)
- test 2:
  - $2^{380} \equiv 1 \bmod 761$ oops
- test 3:
  - $3^{380} \equiv -1 \bmod 761$ OK
  - $3^{152} \equiv 1 \bmod 761$ oops
- test 5 (skip 4 because it is 2^2):
  - $5^{380} \equiv 1 \bmod 761$ oops
- test 6:
  - $6^{380} \equiv -1 \bmod 761$ OK
  - $6^{152} \equiv 67 \bmod 761$ OK
  - $6^{40} \equiv -263 \bmod 761$ hooray!
- So the least primitive root of 761 is 6

89

90

How to Find All Other Primitive Roots

- Once you have found one primitive root, you can easily find all the others!!

- If *a* is a primitive root mod p, and p is prime
  - Then a can generate all other remainders 1, …, $(p-1)$ as powers
    - $a^1 \equiv a, a^2, \ldots, a^{p-1} \equiv 1 \bmod p$
- $a^m \bmod p$ is another primitive root iff m and $p-1$ are coprime

For example:

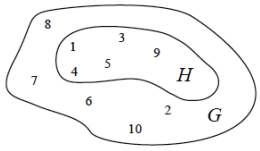
- $6^2 \equiv 36$ or $6^{15} \equiv 686$ are not primitive roots of 761 because
  - $\gcd(2, 760) \neq 1$ and $\gcd(15, 760) \neq 1$
- But, $6^3 = 216$ is another primitive root of 761

Exercise: Find the primitive roots in $\mathbb{Z}_{11}^*$

90

Subgroups

- Let $(G, \circ)$ be a cyclic subgroup. Every element $a \in G$ with $ord(a) = s$ is the primitive root of a cyclic subgroup with s elements
  - i.e. any element of a cyclic group is the generator of a subgroup which in turn is cyclic
- Example: Consider a subgroup of $G = \mathbb{Z}_{11}^*$
  - ord(3) = 5 which generates the subgroup H = {1, 3, 4, 5, 9}



- More precisely it is a subgroup of **prime order** 5 (Theorem 8.2.4)

91

The Discrete Logarithm Problem (DLP) in $\mathbb{Z}_p^*$

- Given a finite cyclic group $\mathbb{Z}_p^*$ of order p - 1 and a primitive root $\alpha \in \mathbb{Z}_p^*$ and another element $\beta \in \mathbb{Z}_p^*$
- The DLP is determining an integer $1 \leq x \leq p$ - 1 such that
  - $\alpha^x \equiv \beta \bmod p$
- x is called the discrete logarithm of β to the base α
  — x = log_alpha B mod p
- Computing discrete logarithms modulo a prime is a **very hard problem**
  - If the parameters are sufficiently large

92

93

DLP and Subgroups

- In practice it is desirable to have DLP in groups with prime cardinality to prevent the Pohlig-Hellman attack
- Since groups $\mathbb{Z}_p^*$ have cardinality p-1 which is not a prime
  — make subgroups of Z*_p with prime order
- Consider the group $\mathbb{Z}_{47}^*$ which has cardinality 46
  — the subgroups have cardinality 1, 2 and 23 (theorem 8.2.3)
- α = 2 is an element in the subgroup with 23 elements
  - Since 23 is a prime α is a primitive element of the subgroup
- A possible discrete logarithm problem is given for β = 36 (which is also in the subgroup)
  - $2^x \equiv 36 \bmod 47$

93

94

Attacks Against DLP

- The security of many asymmetric primitives is based on the difficulty of computing the DLP in cyclic groups, i.e.
  — To compute x for a given alpha and Beta in G such that
- Brute-Force Search
  - Most naive and computationally costly way for computing the discrete logarithm $\log_\alpha \beta$
    - Simply compute powers of the generator α successively until the result equals β

$$\alpha^1 \stackrel{?}{=} \beta$$
$$\alpha^2 \stackrel{?}{=} \beta$$
$$\vdots$$
$$\alpha^x \stackrel{?}{=} \beta$$

94

95

Brute-Force Search

- For a random logarithm x, we do expect to find the correct solution after checking half of all possible x
  - This gives us a complexity of $\mathcal{O}(|G|)$ steps, where $|G|$ is the cardinality of the group
- To avoid brute-force attacks on DLP based cryptosystems in practice
  - The cardinality $|G|$ of the underlying group must thus be sufficiently large
- In the case of the group $\mathbb{Z}_p^*$, p prime, $(p-1)/2$ tests are required on average to compute a discrete logarithm
  - IGI = p -1 should be at least in the order of 2^80 bits

95

96

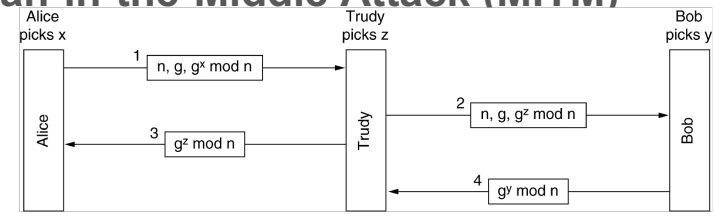
Security of DHKE

- The domain parameter p should have a length of 1024 bits (308 digits) or longer
- α should be a *primitive root* or *generator* of the group (G), whose powers modulo p generate all integers from 1 to p -1
  - Every element a of G can be written as
    - alpha^i = a
- Q: Is 3 is a primitive element modulo $\mathbb{Z}_7^*$?

  3,2,

96

Man-in-the-Middle Attack (MITM)



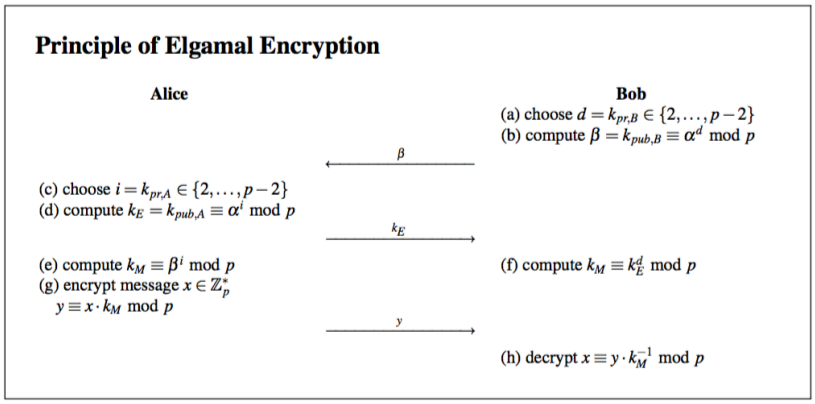
- Alice computes the key g^{xz} mod n
  - So does Trudy (for messages to Alice)
- Bob computes g^{yz} mod n
  - So does Trudy (for messages for Bob)
- Alice thinks she is talking to Bob and so establishes a session key (with Trudy) and so does Bob
  - Also known as the bucket brigade attack
- Exercise: What is the Diffie-Hellman Station-to-Station (STS) protocol?

The ElGamal Encryption Scheme

- The ElGamal encryption scheme can be viewed as an extension of the DHKE protocol
  - Its security is also based on the intractability of the discrete logarithm problem
- We consider the ElGamal encryption scheme over the group $\mathbb{Z}_p^*$, where p is a prime
- If Alice wants to send an encrypted message x to Bob, both parties first perform a DHKE to derive a shared key k_M
  - Assume p and α have been generated
- The new idea here is that Alice uses k_M as a *multiplicative mask* to encrypt x
  - y == x * k_m mod p

The ElGamal Encryption Scheme II

Principle of Elgamal Encryption

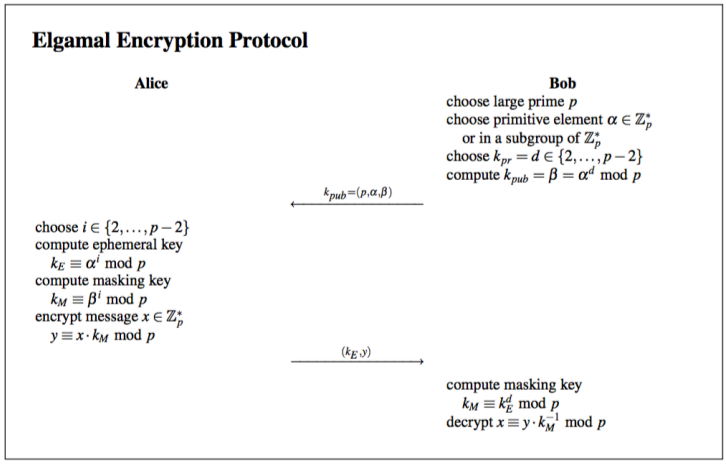


- Bob computes his private key d and public key β
  - This key pair does not change, ie can be used for many messages
- Alice, however, has to generate a new public–private key pair for the encryption of every message
  - Her private key is denoted by i and her public key by k_E
    - the latter is an ephemeral key ie existing only temporarily
- The joint key is denoted by k_M and is used for masking the plaintext

99

The ElGamal Protocol

Elgamal Encryption Protocol



- The set-up phase is executed once by the party who issues the public key and who will receive the message
- The encryption phase and the decryption phase are executed every time a message is being sent

100 — exercise: compute the encryption and decryption of x=26, given p=29, alpha=2, d=12 and i=5

101

ElGamal Miscellaneous Issues

- ElGamal is a *probabilistic encryption scheme.* Encrypting two identical messages x_1 and x_2, where $x_1, x_2 \in \mathbb{Z}_p^*$ using the same public key
  - Results (with extremely high likelihood) in two different ciphertexts $y_1 \neq y_2$

— the session key k_m = beta^i used for encryption is chosen at random for each encryption

- The ciphertext consists of two parts, the ephemeral key k_E and the masked plaintext y
- Since in general all parameters have a bit length of $\lceil \log_2 p \rceil$
  - The ciphertext (k_E, y) is twice as long as the message

— message expansion factor of ElGamal is two!!

101

102

Elliptic Curve Cryptography (ECC)

- Elliptic Curve Cryptography is based on the generalized discrete logarithm problem of finding the integer x such that:

$$\beta = \underbrace{\alpha \circ \alpha \circ \ldots \circ \alpha}_{x \text{ times}} = \alpha^x$$

- ECC is more efficient
- An elliptic "curve" over $\mathbb{Z}_p$, $p > 3$ is the set of all pairs $(x, y) \in \mathbb{Z}_p$ which fulfill

$$y^2 = x^3 + ax + b \bmod p$$

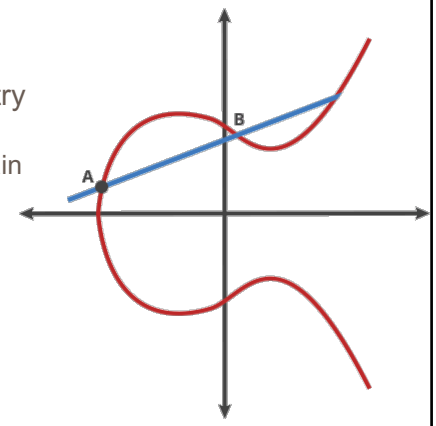
- Together with an imaginary point $\mathcal{O}$, and the condition $4 * a^3 + 27 \cdot b^2 \neq 0 \bmod p$ where $a, b \in \mathbb{Z}_p$
  - The curve is non-singular i.e. has no self-intersections or vertices

102

103

Elliptic Curve Properties II

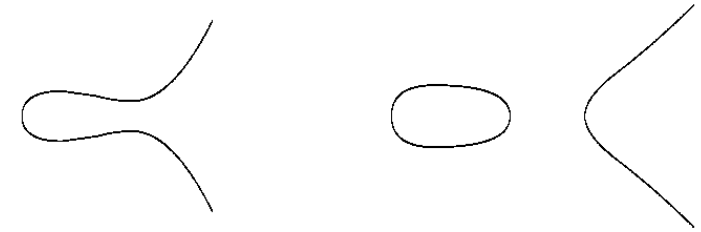
- An elliptic curve has several interesting properties
- One of these is horizontal symmetry
  - Any point on the curve can be reflected over the x-axis and remain the same curve



103

104

Elliptic Curve Properties



$E_1 : Y^2 = X^3 - 3X + 3$ $\quad$ $E_2 : Y^2 = X^3 - 6X + 5$

- A feature of elliptic curves is that there is a natural way to take two points on an elliptic curve and "add" them

104

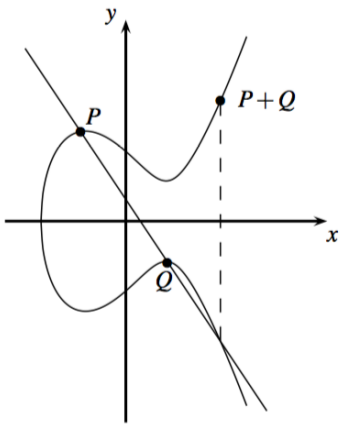
105

Group Operations on Elliptic Curves

- Given two points $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ we compute the coordinates of the third point R such that

$$P + Q = R$$

$$(x_1, y_1) + (x_2, y_2) = (x_3, y_3)$$



Point Addition $P + Q$

- Compute $R = P + Q$ where $P \neq Q$
- Draw a line through P and Q and obtain a third point of intersection between the elliptic curve and the line

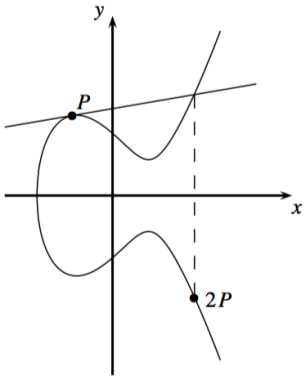
105

106

Group Operations on Elliptic Curves II

Point Doubling $P + P$

- Compute $R = P + Q$
  - where $P = Q$
- Draw draw the tangent line through P
  - Obtain a second point of intersection between this line and the elliptic curve
- Mirror the point of the second intersection along the x-axis to obtain R



106

Computations on Elliptic Curves

Elliptic Curve Point Addition and Doubling Formulas

$$x_3 = s^2 - x_1 - x_2 \bmod p$$
$$y_3 = s(x_1 - x_3) - y_1 \bmod p$$

where

$$s = \begin{cases} \frac{y_2 - y_1}{x_2 - x_1} \bmod p \text{ ; if } P \neq Q \text{ (point addition)} \\ \frac{3x_1^2 + a}{2y_1} \bmod p \text{ ; if } P = Q \text{ (point doubling)} \end{cases}$$

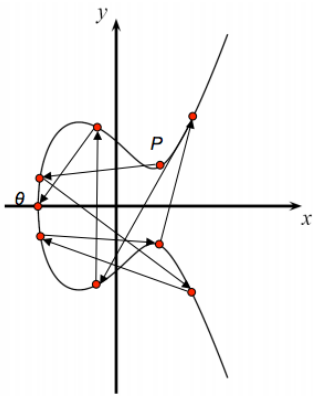
Example: Given E: $y^2 = x^3 + 2x + 2$ mod 17 and point $P = (5,1)$ compute $2P$?

- $2P = P + P = (5,1) + (5,1) = (x_3, y_3)$
- $s = \frac{3x_1^2 + a}{2y_1}$
- $x_3 = s^2 - x_1 - x_2 = 13^2 - 5 - 5 = 159 \equiv 6 \bmod 17$
- $y_3 = s(x_1 - x_3) - y_1 = 13(5 - 6) - 1 = -14 \equiv 3 \bmod 17$
- $2P = (5, 1) + (5, 1) = (6, 3)$

Computations on Elliptic Curves II

- The points on an elliptic curve and the point at infinity $\mathcal{O}$ form cyclic subgroups

$2P = (5,1) + (5,1) = (6,3)$
$3P = 2P+P = (10,6)$
$4P = (3,1)$
$5P = (9,16)$
$6P = (16,13)$
$7P = (0,6)$
$8P = (13,7)$
$9P = (7,6)$
$10P = (7,11)$
$11P = (13,10)$
$12P = (0,11)$
$13P = (16,4)$
$14P = (9,1)$
$15P = (3,16)$
$16P = (10,11)$
$17P = (6,14)$
$18P = (5,16)$
$19P = \mathcal{O}$



- This elliptic curve has order $\#E = |E| = 19$ since it contains 19 points in its cyclic group

Number of Points on an Elliptic Curve

- Determining the point count on elliptic curves in general is hard
- Hasse's theorem bounds the number of points to a restricted interval

Theorem 9.2.2 Hasse's theorem
Given an elliptic curve E modulo p, the number of points on the curve is denoted by #E and is bounded by:

$$p+1-2\sqrt{p} \leq \#E \leq p+1+2\sqrt{p}.$$

- e.g. To generate a curve with about 2^{160} points

Elliptic Curve Discrete Logarithm Problem

- ECC cryptosystems are based on the idea that *d* is large, kept secret, and attackers cannot compute it easily

Definition 9.2.1 Elliptic Curved Discrete Logarithm Problem (ECDLP)
Given is an elliptic curve E. We consider a primitive element P and another element T. The DL problem is finding the integer d, where $1 \leq d \leq \#E$, *such that:*

$$\underbrace{P+P+\cdots+P}_{d \text{ times}} = dP = T. \qquad (9.2)$$

- If d is known, an efficient method to compute the point multiplication dP is required to create a reasonable cryptosystem

111

Double-and-Add Algorithm for Point Multiplication

- Point multiplication is analog to exponentiation in multiplicative groups
- In order to do it efficiently, we can directly adopt the square-and-multiply algorithm (RSA)

Algorithm

Input: Elliptic curve E, an elliptic curve point P and a scalar d with bits d_i

Output: $T = dP$

Initialization: $T = P$

FOR $i = t - 1$ DOWNTO 0
    $T = T+T \bmod n$
    IF $d_i = 1$
        $T = T+P \bmod n$
    RETURN (T)

111

112

Example of Double-and-Add Algorithm

- We consider the scalar multiplication $26P$, which has the following binary representation:

  $26P = (11010_2)P = (d_4d_3d_2d_1d_0)_2P$

- The algorithm scans the scalar bits starting on the left with d_4 and ending with the rightmost bit d_0

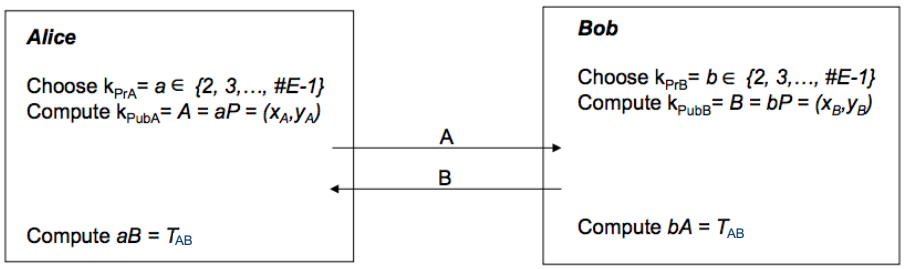
Step		
#0	$P = \mathbf{1}_2 P$	inital setting, bit processed: $d_4 = 1$
#1a	$P+P = 2P = \mathbf{10}_2 P$	DOUBLE, bit processed: d_3
#1b	$2P+P = 3P = 10_2 P + 1_2 P = \mathbf{11}_2 P$	ADD, since $d_3 = 1$
#2a	$3P+3P = 6P = 2(11_2 P) = \mathbf{110}_2 P$	DOUBLE, bit processed: d_2
#2b		no ADD, since $d_2 = 0$
#3a	$6P+6P = 12P = 2(110_2 P) = \mathbf{1100}_2 P$	DOUBLE, bit processed: d_1
#3b	$12P+P = 13P = 1100_2 P + 1_2 P = \mathbf{1101}_2 P$	ADD, since $d_1 = 1$
#4a	$13P+13P = 26P = 2(1101_2 P) = \mathbf{11010}_2 P$	DOUBLE, bit processed: d_0
#4b		no ADD, since $d_0 = 0$

112

113

Elliptic Curve Diffie-Hellman Key Exchange (ECDH)

- Given a prime p, a suitable elliptic curve E and a point $P = (x_P, y_P)$
  - The ECDH Key Exchange is defined by the following protocol



- Joint secret between Alice and Bob: $T_{AB} = (x_{AB}, y_{AB})$
  - One of the coordinates of the point T_{AB} (usually the x-coordinate) can be used as session key

113

114

ElGamal with Elliptic Curves

- As before choose a suitable elliptic curve E and a point $P = (x_P, y_P)$

Bob chooses $k_{PrB} = b$
Computes $K_{PubB} = B = bP$

B ←

Alice chooses $k_{PrA} = a$
Computes $C_1 = kP$
Encrypts the msg x: $C_2 = x + Bk$

C_1, C_2 →

Bob computes $C_2 - bC_1$
Decrypts to get $x = x + bPk - bkP$

114

115

Key Lengths and Efficiency

Algorithm Family	Cryptosystems	Security Level (bit)			
		80	128	192	256
Integer factorization	RSA	1024 bit	3072 bit	7680 bit	15360 bit
Discrete logarithm	DH, DSA, Elgamal	1024 bit	3072 bit	7680 bit	15360 bit
Elliptic curves	ECDH, ECDSA	160 bit	256 bit	384 bit	512 bit
Symmetric-key	AES, 3DES	80 bit	128 bit	192 bit	256 bit

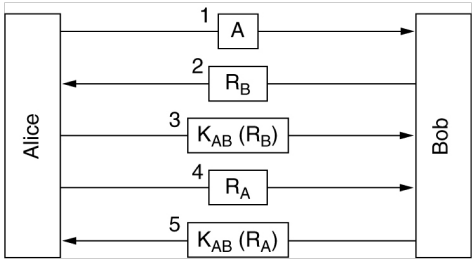
- 256-bit ECC key provides the same security as a 3072-bit RSA key

115

116

Authentication Protocols

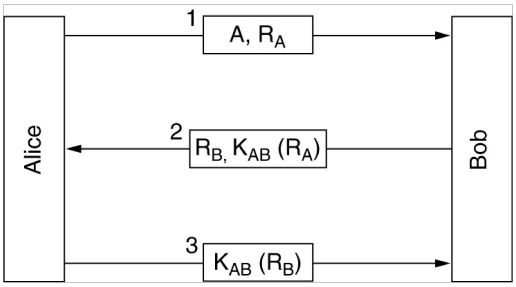
- Authentication is the technique by which a process verifies that a communicating partner is who it is supposed to be and not an impostor
- Authentication based on a *shared secret key*



- The above is an example of a *challenge-response* protocol

116

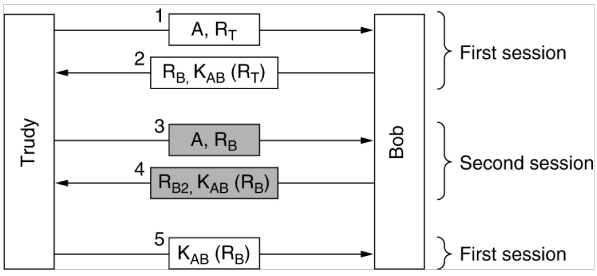
A Shortened Two-Way Protocol



Q: Can you see a problem?

The Reflection Attack

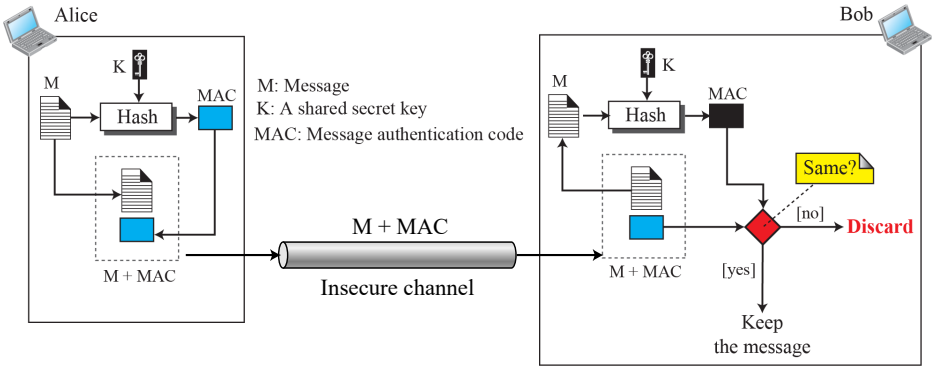
- Trudy can break this if it is possible to open multiple sessions with Bob
  - e.g. if Bob is a bank and is prepared to accept many connections from teller machines at once



Message Authentication Code from Hash Functions (HMAC)

119

- Calculated using a cryptographic hash function in combination with a *secret key*
- Like digital signatures they can be used to quickly verify *data integrity* and *authenticity* of a msg

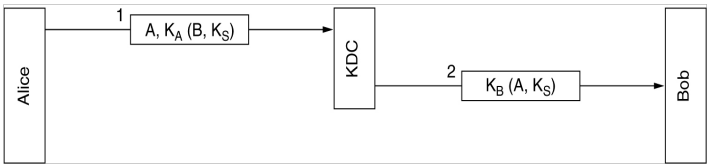


119

Authentication Using a KDC

120

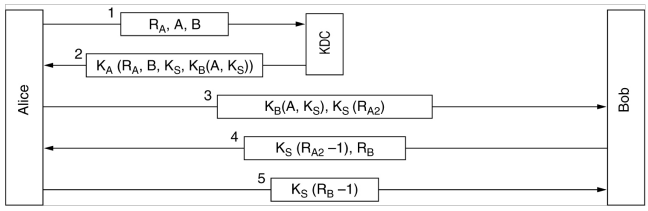
- To talk to n people using a shared secret would require setting up how many keys?
- Alternative is to introduce a Key Distribution Center (KDC)
- Each user has a *single shared key* with the KDC



120

Needham-Schroder Authentication Protocol

- Alice tells the KDC that she wants to talk to Bob
  - Sends a msg containing a random number R_A as a nonce
- The KDC sends back msg 2 containing R_A, a session key and a ticket that she can send to Bob

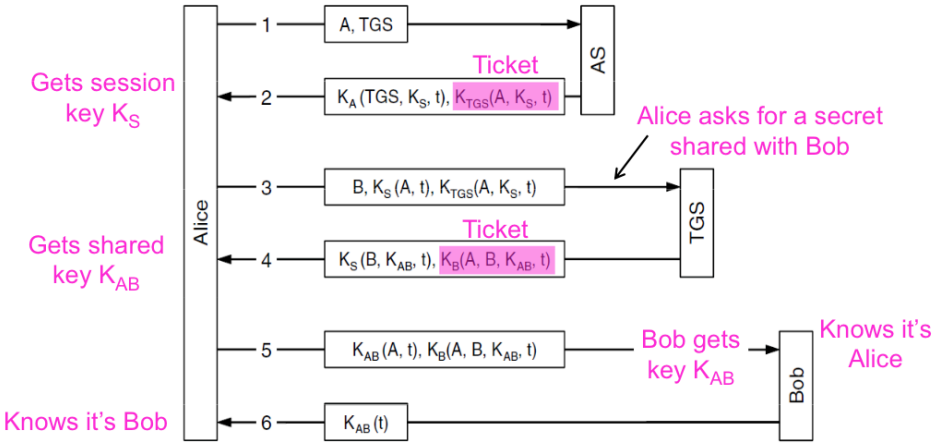


- Alice now sends the ticket to Bob along with a new random number R_{A2} encrypted with the session key K_S

121

Kerberos

- Kerberos V5 is a widely used protocol (e.g. Windows)
  - Authentication includes a Ticket Granting Server (TGS)

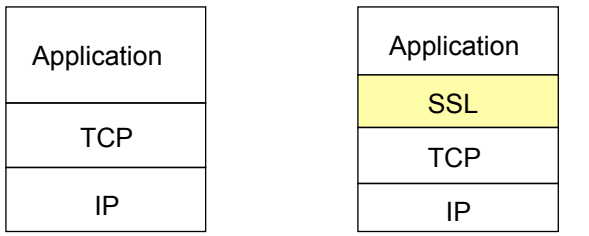


122

123

Secure Sockets Layer (SSL)

- Provides transport layer security to any TCP-based application using SSL services
  - e.g., Between Web browsers and servers for E-commerce
    - In practice Transport Layer Security (TLS) v1.2 should be used
- Security Services



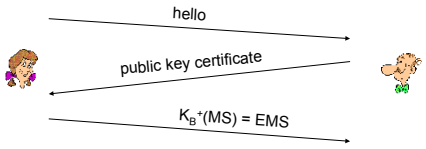
normal application *application with SSL*

123

124

Toy SSL: A Simple Secure Channel

- Handshake
  - Alice and Bob use their certificates, private keys to authenticate each other and exchange shared secret
- Key Derivation
- Data Transfer
  - Data to be transferred is broken up into series of records
- Connection Closure
  - Special messages to securely close connection



MS: master secret
EMS: encrypted master secret

124

125

Toy SSL: Key Derivation

- Considered bad to use same key for more than one cryptographic operation

- Four Keys
  - K_c = Encryption key for data sent from client to server
  - M_c = MAC key for data sent from client to server
  - K_s = Encryption key for data sent from server to client
  - M_s = MAC key for data sent from server to client

- Keys derived from Key Derivation Function (KDF)
  - Takes master secret and (possibly) some additional random data and creates the keys

125

126

Toy SSL: Data Records

- Why not encrypt data in constant stream as we write it to TCP?

- Instead, break stream in series of records
  - Each record carries a MAC

- Receiver can act on each record as it arrives
  - Need to distinguish MAC from data
    - Want to use variable-length records

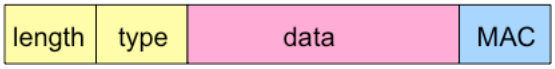
126

127

Toy SSL: Control Information

- Problem: Attacker can capture and replay, record or re-order records
- Sol: Put sequence number into MAC
  - MAC = MAC(M_x, sequence||data)
    - Note: no sequence number field
- Problem: Attacker could replay all records

Sol: Use nonce

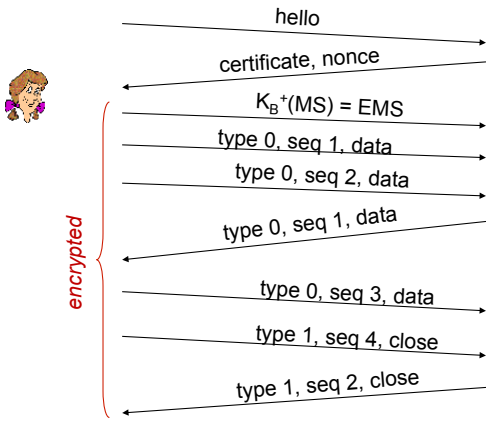


- Problem: Truncation Attack
  - Attacker forges TCP connection close segment
    - One or both sides thinks there is less data than there actually is
- Sol: Record Types, with one type for closure
  - Type 0 for data; Type 1 for closure
  - MAC = MAC(M_x, sequence||type||data)

127

128

Toy SSL: Summary & Outstanding Issues



- How long are fields?
- Which encryption protocols?
- Want negotiation?
  - Allow client and server to support different encryption algorithms
  - Allow client and server to choose together specific algorithm before data transfer

128

SSL Cipher Suite

- Cipher Suite
  - Public-key algorithm
  - Symmetric encryption algorithm
  - MAC algorithm
- SSL supports several cipher suites
- Negotiation
  - Client, server agree on cipher suite
  - Client offers choice

Common SSL symmetric ciphers

- DES – Data Encryption Standard: block
- 3DES – Triple strength: block
- RC2 – Rivest Cipher 2: block
- RC4 – Rivest Cipher 4: stream

SSL Public key encryption

- RSA

129

Real SSL: Handshake

- Server Authentication
- Negotiation: Agree on crypto algorithms
- Establish Keys
- Client Authentication (optional)

130

131

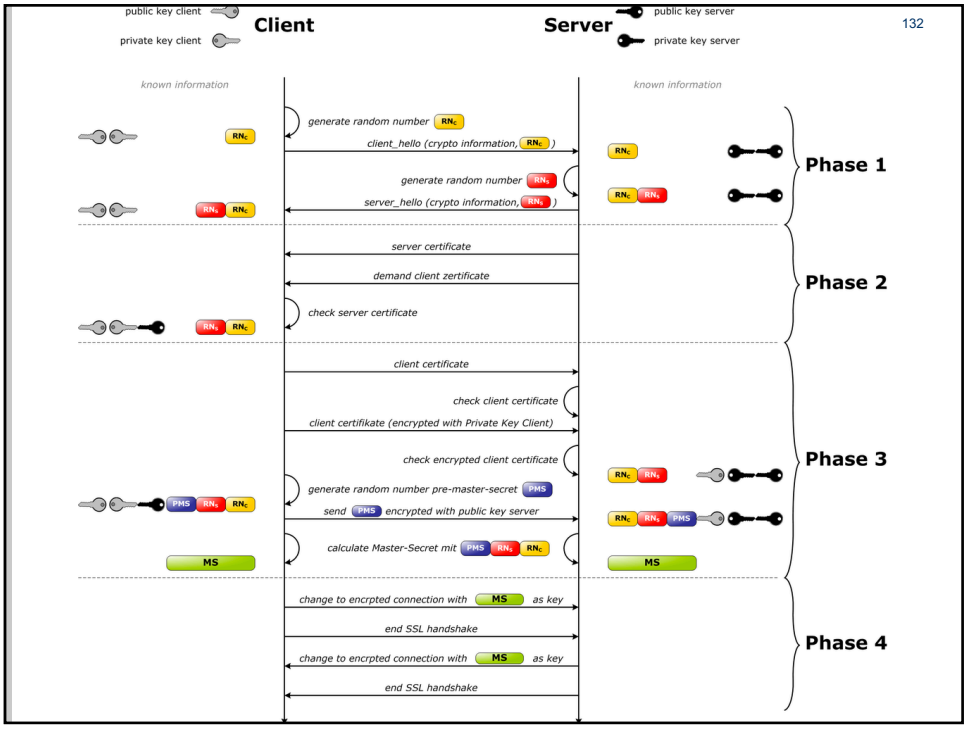
Real SSL: Handshake II

- Client sends list of algorithms it supports, along with client nonce
- Server chooses algorithms from list; sends back: choice + certificate + server nonce
- Client verifies certificate, extracts server's public key, generates pre_master_secret, encrypts with server's public key, sends to server

  Client and server independently compute encryption and MAC keys from pre_master_secret nad nonces

- Client sends a MAC of all the handshake messages
- Server sends a MAC of all the handshake messages
  - Q: Why do we this?

131



132

133

Real SSL: Handshake III

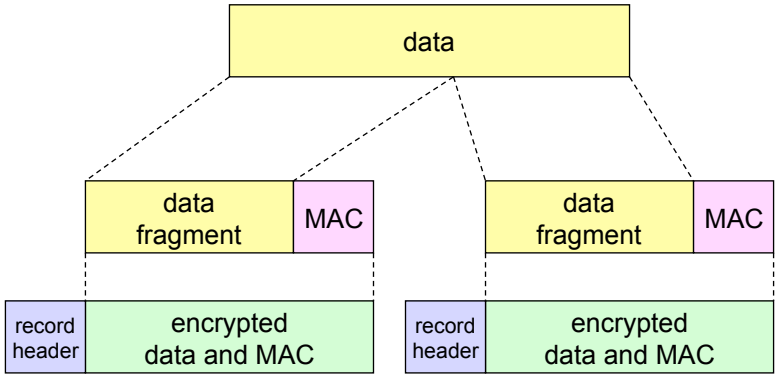
- Last 2 steps protect handshake from tampering
  - Client typically offers range of algorithms, some strong, some weak
- MITM could delete stronger algorithms from list
  - Last 2 steps prevent this
- Why two random nonces?
  - Suppose Trudy sniffs all messages between Alice & Bob
- Next day, Trudy sets up TCP connection with Bob (Amazon), sends exact same sequence of records
  - Bob thinks Alice made two separate orders for the same thing
- Sol: Bob sends different random nonce for each connection. This causes encryption keys to be different on the two days

  Trudy's messages will fail bob's integrity check

133

134

SSL Record Protocol



Record Header: content type; version; length

MAC: includes sequence number, MAC key M_x

Fragment: each SSL fragment 2^{14} bytes (~16 Kbytes)

134

SSL Key Derivation

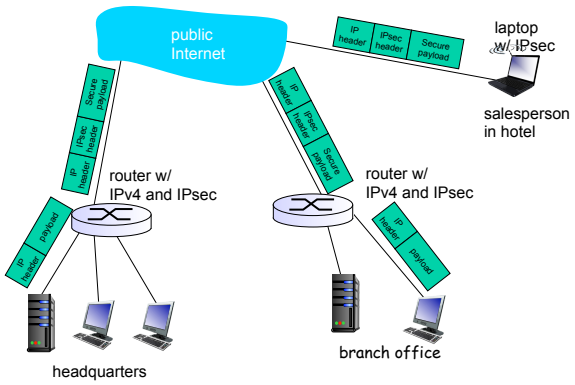
- Client nonce, server nonce, and pre-master secret input into pseudo random-number generator

  Produces Master Secret

- Master Secret and nonces input into another random-number generator
  - To produce the "key block"
- Key block sliced and diced
  - Client MAC key
  - Server MAC key
  - Client encryption key
  - Server encryption key
  - Client initialization vector (IV)
  - Server initialization vector (IV)

Virtual Private Networks (VPNs)

- Institutions often want private networks for security
  - Costly! Separate routers, links, DNS infrastructure

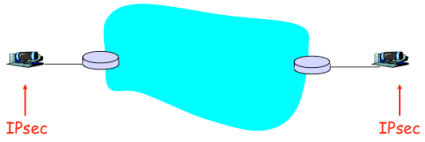


- With a VPN, inter-office traffic is sent over the public Internet

137

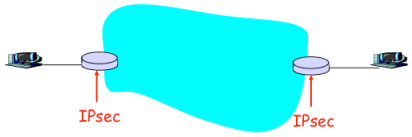
IPsec - Transport vs Tunnel Mode

- Transport Mode



  - IPsec datagram emitted and received by end-system
    - Protects upper level protocols

- Tunnel Mode



  - End routers are IPsec aware
    - Hosts need not be

137

138

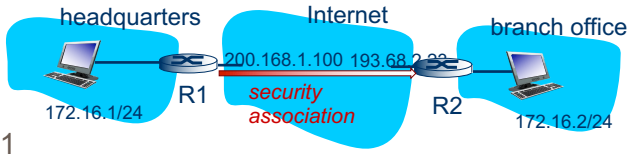
IPsec – Network Layer Security

- Network-layer authentication
  - Destination host can authenticate source IP address
- Network-layer secrecy
  - Sending host encrypts the data in IP datagram
    - TCP and UDP segments; ICMP and SNMP messages
- Two principal protocols
  - Authentication header (AH) protocol
  - Encapsulation security payload (ESP) protocol
- Most common
  - Tunnel mode with ESP

138

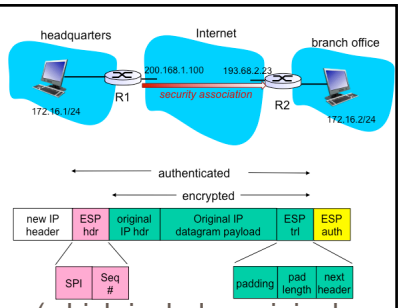
IPsec – Network Layer Security II

- Source and destination perform a handshake
  - Create network-layer logical channel called a security association (SA)
    - each SA is unidirectional



- SA at R1
  - 32-bit identifier for SA: Security Parameter Index (SPI)
  - Origin interface of the SA (200.168.1.100)
  - Destination interface of the SA (193.68.2.23)
  - Type of encryption to be used (e.g. 3DES with CBC)
  - Encryption key
  - Type of integrity check (e.g. HMAC with with MD5)
  - Authentication key

ESP with Tunnel Mode



- Appends to back of original datagram (which includes original header fields!) an "ESP trailer" field
  - encrypts result using algorithm and key specification by SA
- Appends to front of this encrypted quantity the "ESP header
- Creates authentication MAC over the four fields, using algorithm and key specified in SA
  - appends MAC to the end forming payload
- Creates brand new IP header, with all the classic IPv4 header fields, which it appends before payload

END END END

141

DNS Security Extension (DNSSEC)

- Plain old DNS does not allow you to check the *authenticity* or *integrity* of a message
- MITM Attack
  - A resolver has no way to verify the authenticity and integrity of the data sent by name servers
- Packet Sniffing
  - DNS sends an entire query or response in a single unsigned and unencrypted UDP pkt
- Transaction ID Guessing
  - An attacker can respond with false answers to a predicted query
    - On the client there are 2^{32} possible combinations of ID (2^{16}) and UDP ports (2^{16})

141

I2

DNSSEC

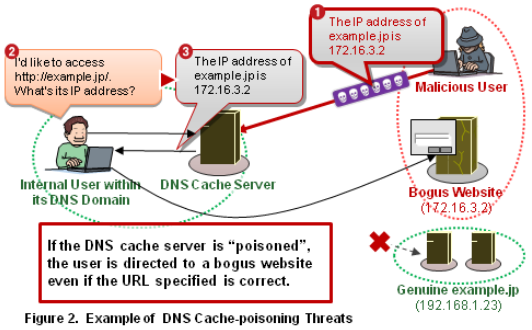


Figure 2. Example of DNS Cache-poisoning Threats

- DNSSEC provides origin authentication and integrity assurance services for DNS data
- DNSSEC has two perspectives
  - Domain owners sign their zone and publish the signed zone on their authoritative name servers
  - Querying hosts validate the digital signatures they receive in answers, along a chain of trust

142

143

DNSSEC RRs

DNSSEC adds a number of new resource record types:

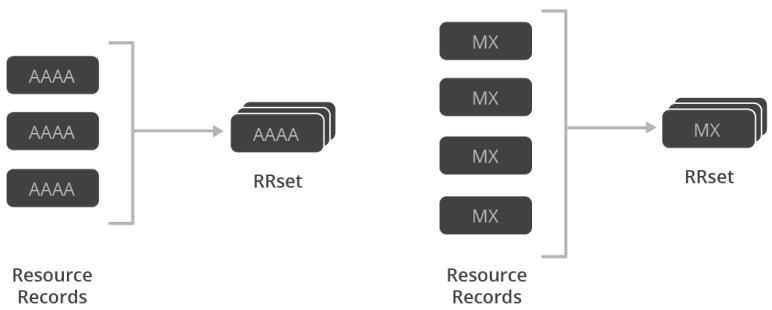
- RRSIG – Contains a cryptographic signature
- DNSKEY – Contains a public signing key
- DS – Contains a hash of a DNSKEY record
- Others ….

143

144

RRsets

- First step towards securing a zone with DNSSEC is to group all the records with the same type into a resource record set (RRset)
  - E.g., if you have three AAAA records in your zone, they would all be bundled into a single AAAA RRset



144

145

Zone-Signing Keys

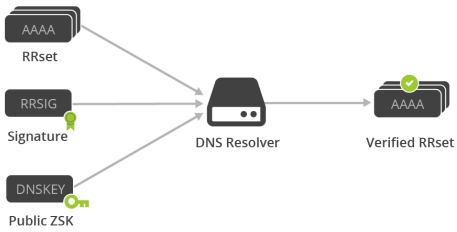
- Each zone in DNSSEC has a zone-signing key pair (ZSK)
- A zone operator creates digital signatures for each RRset using the private ZSK

145

146

Zone-Signing Keys II

- Zone operators also needs to make their public ZSK available by adding it to their name server in a DNSKEY record
- When a DNSSEC resolver requests a particular record type (e.g., AAAA), the name server also returns the corresponding RRSIG

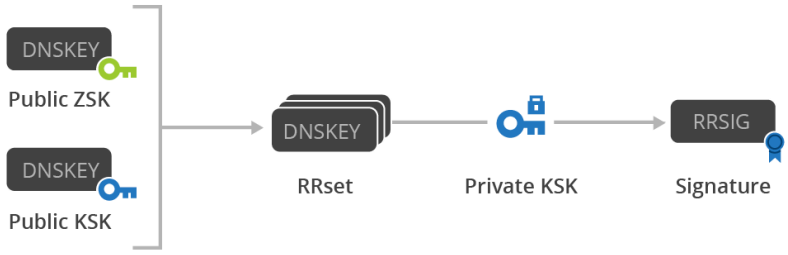


146

147

Key-Signing Keys

- The KSK validates the DNSKEY record
- It signs the public ZSK (which is stored in a DNSKEY record)

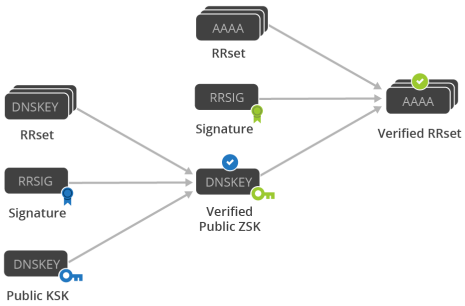


147

148

DNSSEC Validation

- Request the desired RRset
  - Returns the corresponding RRSIG record
- Request the DNSKEY records containing the public ZSK and public KSK
  - Returns the RRSIG for the DNSKEY RRset
- Verify the RRSIG of the DNSKEY RRset with the public KSK
- Verify the RRSIG of the requested RRset with the public ZSK



148

Delegation Signer Records

- DNSSEC introduces a delegation signer (DS) record to allow the transfer of trust from a parent zone to a child zone
- A zone operator hashes the DNSKEY record containing the public KSK
- Every time a resolver is referred to a child zone, the parent zone also provides a DS record
  - To check the validity of the child zone's public KSK