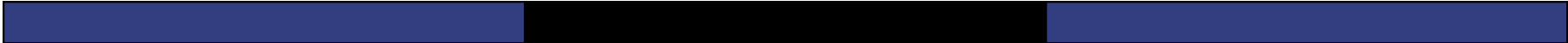


# L4: Basic Cryptography

## III



Hui Chen, Ph.D.  
Dept. of Engineering & Computer Science  
Virginia State University  
Petersburg, VA 23806

# Acknowledgement

---

- Many slides are from or are revised from the slides of the author of the textbook
  - Matt Bishop, Introduction to Computer Security, Addison-Wesley Professional, October, 2004, ISBN-13: 978-0-321-24774-5. [Introduction to Computer Security @ VSU's Safari Book Online subscription](#)
  - <http://nob.cs.ucdavis.edu/book/book-intro/slides/>

# Overview

---

## □ Classical Cryptography

- Caesar cipher
- Vigenere cipher
- DES
- AES

## □ Public Key Cryptography

- Diffie-Hellman
- RSA

## □ Cryptographic Checksums

- HMAC



Previous lectures



This and future lectures

# Public Key Cryptography

---

## □ Two keys

- *Private key* known only to individual
- *Public key* available to anyone

## □ Idea

- Confidentiality
  - Encipher using public key, decipher using private key
- Integrity and authentication
  - Encipher using private key, decipher using public key

# Requirements

---

- ❑ It must be *computationally easy* to encipher or decipher a message given the appropriate key
- ❑ It must be *computationally infeasible* to derive the private key from the public key
- ❑ It must be *computationally infeasible* to determine the private key from a chosen plaintext attack

# RSA

---

R. L. Rivest, A. Shamir, and L. Adleman. 1978. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM* 21, 2 (February 1978), 120-126. DOI=<http://dx.doi.org/10.1145/359340.359342>

- ❑ Exponentiation cipher
- ❑ Relies on the difficulty of determining the number of numbers relatively prime to a large integer  $n$

# Background

---

- Totient function  $\phi(n)$ 
  - Number of positive integers less than  $n$  and relatively prime to  $n$ 
    - *Relatively prime* means with no factors in common with  $n$
- Example:  $\phi(10) = 4$ 
  - 1, 3, 7, 9 are relatively prime to 10
- Example:  $\phi(21) = 12$ 
  - 1, 2, 4, 5, 8, 10, 11, 13, 16, 17, 19, 20 are relatively prime to 21

# Algorithm

---

- Choose two large prime numbers  $p, q$ 
  - Let  $n = pq$ ; then  $\phi(n) = (p-1)(q-1)$
  - Choose  $e < n$  such that  $e$  is relatively prime to  $\phi(n)$ .
  - Compute  $d$  such that  $ed \bmod \phi(n) = 1$
- Public key:  $(e, n)$ ; private key:  $d$
- For confidentiality
  - Encipher:  $c = m^e \bmod n$
  - Decipher:  $m = c^d \bmod n$
- For integrity and authentication
  - Encipher:  $c = m^d \bmod n$
  - Decipher:  $m = c^e \bmod n$



# Example: Confidentiality

---

- Take  $p = 7$ ,  $q = 11$ , so  $n = 77$  and  $\phi(n) = 60$
- Alice chooses  $e = 17$ , making  $d = 53$
- Bob wants to send Alice secret message HELLO (07 04 11 11 14)
  - $07^{17} \bmod 77 = 28$
  - $04^{17} \bmod 77 = 16$
  - $11^{17} \bmod 77 = 44$
  - $11^{17} \bmod 77 = 44$
  - $14^{17} \bmod 77 = 42$
- Bob sends 28 16 44 44 42

# Example

---

- Alice receives 28 16 44 44 42
- Alice uses private key,  $d = 53$ , to decrypt message:
  - $28^{53} \bmod 77 = 07$
  - $16^{53} \bmod 77 = 04$
  - $44^{53} \bmod 77 = 11$
  - $44^{53} \bmod 77 = 11$
  - $42^{53} \bmod 77 = 14$
- Alice translates message to letters to read HELLO
  - No one else could read it, as only Alice knows her private key and that is needed for decryption

# Exercise L4-1

---

- Take  $p = 3$ ,  $q = 5$  and use RSA to encrypt the following message to achieve confidentiality

H I

# Example: Integrity and Authentication

---

- Take  $p = 7$ ,  $q = 11$ , so  $n = 77$  and  $\phi(n) = 60$
- Alice chooses  $e = 17$ , making  $d = 53$
- Alice wants to send Bob message HELLO (07 04 11 11 14) so Bob knows it is what Alice sent (no changes in transit, and authenticated)
  - $07^{53} \bmod 77 = 35$
  - $04^{53} \bmod 77 = 09$
  - $11^{53} \bmod 77 = 44$
  - $11^{53} \bmod 77 = 44$
  - $14^{53} \bmod 77 = 49$
- Alice sends 35 09 44 44 49

# Analysis on Example: Integrity and Authentication

---

- ❑ Bob receives 35 09 44 44 49
- ❑ Bob uses Alice's public key,  $e = 17$ ,  $n = 77$ , to decrypt message:
  - $35^{17} \bmod 77 = 07$
  - $09^{17} \bmod 77 = 04$
  - $44^{17} \bmod 77 = 11$
  - $44^{17} \bmod 77 = 11$
  - $49^{17} \bmod 77 = 14$
- ❑ Bob translates message to letters to read HELLO
  - Alice sent it as only she knows her private key, so no one else could have enciphered it
  - If (enciphered) message's blocks (letters) altered in transit, would not decrypt properly

# Example: Confidentiality *and* Integrity

---

- ❑ Alice wants to send Bob message HELLO both enciphered and authenticated (integrity-checked)
- ❑ Alice's keys: public (17, 77); private: 53
- ❑ Bob's keys: public: (37, 77); private: 13
- ❑ Alice enciphers HELLO (07 04 11 11 14):
  - $(07^{53} \bmod 77)^{37} \bmod 77 = 07$
  - $(04^{53} \bmod 77)^{37} \bmod 77 = 37$
  - $(11^{53} \bmod 77)^{37} \bmod 77 = 44$
  - $(11^{53} \bmod 77)^{37} \bmod 77 = 44$
  - $(14^{53} \bmod 77)^{37} \bmod 77 = 14$
- ❑ Alice sends 07 37 44 44 14

# Analysis on Example: Security Services (1)

---

## □ Confidentiality

- *Only* the owner of the private key *knows* the private key, so text enciphered with public key cannot be read by anyone except the owner of the private key

## □ Authentication

- *Only* the owner of the private key *knows* the private key, so text enciphered with private key must have been generated by the owner

# Analysis on Example: Security Services (2)

---

## □ Integrity

- Enciphered letters cannot be changed undetectably without knowing private key

## □ Non-Repudiation

- Message enciphered with private key came from someone who knew it



# Analysis on Example: Warnings

---

- ❑ Encipher message in blocks should be considerably larger than the examples above
  - If 1 character per block, RSA can be broken using statistical attacks (as in classical cryptosystems)
  - Attacker cannot alter letters, but can rearrange them and alter message meaning
    - ❑ Example: reverse enciphered message of text ON to get NO

## Exercise L4-2

---

- Take  $p = 3$ ,  $q = 5$  and use RSA to encrypt the following message to achieve Integrity and Authentication.
- To describe your solution, use two personalities, *Alice* and *Bob*  
HI
- Can *Alice* and *bob* achieve both confidentiality and integrity/authentication and how?

# Cryptographic Checksums

---

- ❑ Mathematical function to generate a set of  $k$  bits from a set of  $n$  bits (where  $k \leq n$ ).
  - $k$  is smaller than  $n$  except in unusual circumstances
- ❑ Example
  - ASCII parity bit
    - ❑ ASCII has 7 bits; 8th bit is “parity”
    - ❑ Even parity: even number of 1 bits
    - ❑ Odd parity: odd number of 1 bits

# Example Use

---

- Bob receives “10111101” as bits.
- Sender is using *even* parity
  - Bob counts 6 1-bits and 6 is *even*; no error detected in the character received, assume the received character is correct
  - Note: could still be garbled, but 2 or more bits would need to have been changed to preserve parity
- Sender is using *odd* parity
  - Bob counts 6 of 1-bits. Since 6 is *even*, so character was not received correctly

# Cryptographic Checksum

---

- Strong hash function or strong one-way function
- Cryptographic checksum  $h: A \rightarrow B$ :
  1. For any  $x \in A$ ,  $h(x)$  is easy to compute
  2. For any  $y \in B$ , it is computationally infeasible to find  $x \in A$  such that  $h(x) = y$
  3. It is computationally infeasible to find two inputs  $x, x' \in A$  such that  $x \neq x'$  and  $h(x) = h(x')$

# Collisions

---

- If  $x \neq x'$  and  $h(x) = h(x')$ ,  $x$  and  $x'$  are a *collision*
- Pigeonhole principle: if there are  $n$  containers for  $n+1$  objects, then at least one container will have 2 objects in it.
- Application: if there are 32 files and 8 possible cryptographic checksum values, at least one value corresponds to at least 4 files

# Collision Requirement

---

3. It is computationally infeasible to find two inputs  $x, x' \in A$  such that  $x \neq x'$  and  $h(x) = h(x')$

Or

4. Given any  $x \in A$ , it is computationally infeasible to find another  $x' \in A$  such that  $x \neq x'$  and  $h(x) = h(x')$
- Subtle difference between the one: it is considerably harder to find  $x'$  meeting the conditions in property 4 than it is to find a pair  $x$  and  $x'$  meeting the conditions in property 3

# Keyless or Keyed Checksum

---

- Keyless cryptographic checksum
  - Requires no cryptographic key
  - Example
    - MD5 (and MD4), SHA-1 (and SHA-2, SHA-3), HAVAL, and Snefru
- Keyed cryptographic checksum
  - Requires cryptographic key
  - Example
    - DES in chaining mode: encipher message, use last  $n$  bits. Requires a key to encipher, so it is a keyed cryptographic checksum.



# Known Attacks

---

## □ MD4 and MD5

- Dobbertin's attack (1996)

## □ Snefru

- Differential cryptanalysis if 4 rounds or less are used (Biham and Shamir, 1993)

## □ SHA-0, SHA-1, and SHA-2

- Chabaud and Joux 's attack on SHA-0 (1998)
- Wang, Yin, and Yu's attack on SHA-1 (2005)
- Khovratovich, Rechberger and Savelieva's attack on SHA-2 (2011)

# HMAC

---

- ❑ Make keyed cryptographic checksums from keyless cryptographic checksums
- ❑  $h$ : keyless cryptographic checksum function
  - Input: blocks of  $b$  bytes
  - Output: blocks of  $l$  bytes
  - $k'$ : cryptographic key of length  $b$  bytes
  - If short, pad with 0 bytes; if long, hash to length  $b$
  - $ipad$  is 00110110 repeated  $b$  times
  - $opad$  is 01011100 repeated  $b$  times
- ❑  $HMAC-h(k, m) = h(k' \oplus opad || h(k' \oplus ipad || m))$ 
  - $\oplus$  exclusive or,  $||$  concatenation

# Strength of HMAC

---

- Strength of HMAC depends on the strength of the hash function  $h$  (Bellare, Canetti, and Krawczyk, 1996)

## Exercise L4-3

---

- ❑ In a Linux system, with your own research, create a checksum for a file and use it to check whether the file is modified after the checksum is created.
- ❑ Examine how it may be used by surveying a few file downloading sites,
  - e.g., the Fedora Linux project  
[http://mirror.pnl.gov/fedora/linux/releases/24/Workstation/x86\\_64/iso/](http://mirror.pnl.gov/fedora/linux/releases/24/Workstation/x86_64/iso/)

# Summary

---

- ❑ Two main types of cryptosystems: classical and public key
- ❑ Classical cryptosystems encipher and decipher using the same key
  - Or one key is easily derived from the other
- ❑ Public key cryptosystems encipher and decipher using different keys
  - Computationally infeasible to derive one from the other
- ❑ Cryptographic checksums provide a check on integrity