Computer Security

08r. Assignment 7 review

Pre-exam 2 review – the major concepts

Paul Krzyzanowski • David Domingo • Ananya Jana

Rutgers University

Spring 2019

Assignment 7 Review

Question 1

What is a necessary condition for perfect secrecy?

Claude Shannon proved that a cipher has perfect secrecy if and only if there are as many possible keys as possible plaintexts, so every key is equally likely.

This means the key has to be random and as long as the message … which means that this is not practical for most real-word use cases

See page 133

Question 2

How did Robert Hooke use a one-way function in 1678?

He published an anagram of a message and revealed it two years later. This allowed him to establish priority for his idea (Hooke's Law for a spring) without disclosing it at the time.

Discussion:

This is a precursor to the idea of using a hash.

If I publish a hash of a message, H(M)

And later show you the message, M, you know that I *must have had the message in order to generate the hash* – a good cryptographic hash function will make it difficult to generate a message that hashes to a specific, desired value

Note that "difficult" = "not feasible" = "impossible for all practical purposes"

See page 137

Question 3

What are the three properties of hash functions listed in the text?

1. They are **one-way functions**
   – Given *x*, it is easy to compute *h(x)* but difficult to find *x* when given *h(x)*
2. The function **does not give any information** about any part of the input.
3. It is hard to find collisions
   – A collision is when you can find two messages M_1, M_2 where $M_1 \neq M_2$ but $h(M_1) = h(M_2)$

See section 5.3.1 (Random Functions – Hash Functions), 5.3.1.1 (Properties) Page 141

Question 4

What does an s-box do in a symmetric block cipher?

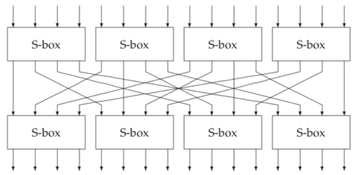
- It is a substitution box – it substitutes one bit pattern with another
- Think of it as a lookup table

See section 5.4.1, SP Networks, p. 149

Question 4 – discussion

From the text:

"The earliest block ciphers were simple networks which combined substitution and permutation circuits, and so were called SP-networks. Figure 5.10 shows an SP-network with sixteen inputs, which we can imagine as the bits of a sixteen-bit number, and two layers of four-bit invertible substitution boxes (or *S-boxes*), each of which can be visualized as a lookup table containing some permutation of the numbers 0 to 15."



Question 4 – discussion

Popular symmetric ciphers (e.g., AES and DES) are

- **Block ciphers**
  - They encrypt a chunk of data, then do the next chunk, …
- **Use an SP network: a combination of substitutions & permutations**
  - This adds confusion & diffusion
    - **Confusion** = every bit of ciphertext depends on various bits of the key. You cannot find a connection between a bit of the key and a bit of the ciphertext.
    - **Diffusion** = if you change a bit in the plaintext, approximately half of the bits in the ciphertext will change.
- **Iterate over multiple rounds**
  - One or a few iterations through s-boxes will not add enough confusion & diffusion to the output.
  - Modern symmetric ciphers use many more rounds (iterations)

Topics you should know for the exam

This is not a review of everything we covered in the past 4 lectures but a listing of some of the major concepts you should know

You should be familiar with these topics

If you are not, this this an indication of areas you need to focus on in preparing for the exam

We will not review everything from four lectures in 50 minutes!

Application Sandboxing

Application sandboxing vs. full sandboxing

- **Full sandboxing** (e.g., containers)
  - Create an isolated environment for an application or group of related programs
  - Almost always: isolated file system namespace
  - Try to simulate a virtual machine – isolate a service (program)
- **Application sandboxing**
  - Restrict operations that an application can perform
  - Not just if it's root
  - Example: deny access to the network or to specific files

Forms of application sandboxing

- **System call interposition with user-level validation**
  - System call hooks in the kernel but decisions made by a user process (example: Janus sandbox)
- **Full OS sandboxing**
  - The OS has native support for sandboxing
  - Policies are compiled & pushed into the kernel
  - Examples: Linux Seccomp-BPF, Apple sandbox
- **Brower-based native applications**
  - Example: Chrome NaCL (Native Client)
  - Compile code with special libraries that perform validation of system requests
- **Java sandbox** (process virtual machine)
  - Runtime environment gets all requests & validates them

Malware

Worms & Viruses

- Worm
  - Standalone software
- Virus
  - Requires a host program: a virus attaches itself to another piece of software
- Components
  - Infection mechanism: how does it spread?
  - Payload: what does the malware do?
  - Trigger (logic bomb): when will the payload run?

Some places where malware resides (1)

- File infector
  - Virus is part of an executable program
- Bootloader
  - Boot process invokes the malware
- Flash drive
  - Malicious software on the drive
  - or modified malicious firmware makes the drive send commands
  - or lost drive causes allows data to be stolen by someone else
- Macros
  - Office documents, editor files, PDF files
- Trojan horse
  - Useful program but also has a covert purpose

Some places where malware resides (2)

- Backdoors
  - Program with some undocumented mechanism to allow a user to log in or execute commands
- Rootkit
  - Modify the operating system, libraries, and/or commands to hide the presence of malware

Social engineering

- #1 way of getting malware onto a system
- Get users to do something that isn't in their best interest
- Examples:
  - Phishing: email masqueraded to come from someone you trust with links or attachments you're asked to click on
  - Spear phishing: personalized phishing
  - Deceptive web sites: web sites that masquerade as legitimate companies or services
    - Phishing requests often take you there
    - Goal: steal login credentials
  - Deceptive content on web sites
    - Ads on file sharing sites often look like download links

Defenses

1. Signature-based scanning
   – Accurate but requires knowledge of sample bytes of the malware
2. Behavior-based monitoring
   – Difficult to detect what is an anomaly but works on new malware

Countermeasures
– Use a packer to obscure the payload when it's in the file system
– Polymorphic viruses
  • Modify the code each time the virus propagates
– Social engineering: ask a user to override permissions

Cryptographic Systems

Ciphers

- Symmetric
  – Same key for encryption & decryption
- Asymmetric (public key)
  – Two related keys: encrypt with one, decrypt with the other
  – Based on trapdoor functions
- Kerckhoffs's Principle – use publicly known algorithms

Classic cipher types

- Monoalphabetic substitution cipher
  – Replace one character (bit pattern) with another
  – Caesar cipher: the substitution alphabet is the alphabet shifted by *n* positions
  – Vulnerable to frequency analysis: certain letters are more likely than others
- Polyalphabetic substitution cipher
  – Change the substitution alphabet based on the position of the character
  – Still vulnerable to frequency analysis but more difficult
- One-time pad
  – Key = long set of random characters
  – Each key character encrypts one plaintext character
  – Problem
    • Key must be (1) truly random, (2) as long as the message, (3) never reused
- Transposition cipher
  – Scramble the data instead of substituting it

Ciphers

- What is perfect secrecy?
  – Ciphertext contains no information about the plaintext
- Stream cipher
  – Approximation of a one time pad
  – Instead of a random stream of bits for the long key, create the stream using a pseudorandom number generator
- Block ciphers
  – Encrypt a chunk of data at a time
  – Most ciphers we use are block ciphers – usually based on key size
    • Symmetric ciphers: AES (32- or 64-byte blocks), DES (8-byte blocks)
    • Public key ciphers:
      RSA (typically 8- or 16-byte blocks)
      ECC (typically 28-, 32-, or 64-byte blocks)

Using block ciphers

- Electronic code book (ECB)
  – Each block of plaintext is encrypted individually
- Problem
  – Common parts of plaintext will produce identical ciphertext
- Solution
  – **Counter mode**
    • An incrementing count is encrypted with the key for each block
    • Result is XORed with the block of plaintext to create ciphertext
  – **Cipher block chaining (CBC) mode**
    • Encryption of each block is a function of the previous blocks

Secure communication

Secure communication

- Symmetric cryptography
  - Encrypt and decrypt with a shared secret key
- Public key cryptography
  - Encrypt with the destination's public key
  - They decrypt with their private key
- Hybrid cryptography
  - Public key cryptography is *really slow* … and generating keys takes time
    - Use public key cryptography to send a random *session key*
    - Then communicate with symmetric cryptography

Key Exchange

Key exchange

- **Diffie-Hellman** key exchange
  - Not an encryption algorithm: only key exchange
  - Forward secrecy
    - Means that even if you steal someone's private key, you cannot decipher their past communications
    - Requires single-use (ephemeral) keys
- **Needham-Schroeder** algorithm – use a trusted 3rd party to send a session key
  - Denning-Sacco protocol: add timestamps
  - Otway-Rees protocol: Use a random session ID for each message
- **Kerberos**
  - Authentication, authorization, & key exchange
  - Essentially the Denning-Sacco protocol

Integrity

Integrity

- Hash function = strong checksum
- Hash pointer = pointer and hash(data pointed to)
  - Blockchain = linked list
  - Merkle tree = binary tree
- MAC = Message Authentication Code
  - Encrypted hash – shared key
  - Forms: HMAC, CBC-MAC
- Digital signature
  - Encrypted hash – using public key cryptography
- Digital certificates
  - { name, public key } signed by the issuer (Certification Authority)

The end