UNIVERSITY of TECHNOLOGY, MAURITIUS

SCHOOL OF BUSINESS INFORMATICS & SOFTWARE ENGINEERING

Module Information Pack

B.Sc (Hons.) Computer Applications
B.Sc (Hons.) Computer Science & Network Security

Network Security

SECU2102

Academic Year 2008/2009 – Semester 1

Programme Director (BCNS):	Mrs. S. Armoogum
Programme Coordinator (BCNS):	Mrs. S. Armoogum
Programme Director (BCA):	Mr. P. Kanaksabee
Programme Coordinator (BCA):	Mr. P. Kanaksabee
Module Coordinator:	Mrs. S. Armoogum
Module Convenor:	Mr. Rishi Heerasing
Office:	Room G2.14 Level 2 SOBISE BLOCK
Phone:	234 7624 ext. 124
E-mail:	HansHeerasing@utm.intnet.mu
Academic Tutoring:	Mondays: 14:00-16:00
Lecture Day and Time:	Saturdays: 12:30 -14:30 G1. 1 & 14:30-16:30 G1.1
Credits & Level:	3 credits, Level 3
Pre-requisites (If applicable):	None
Co-requisites (If applicable):	None
Method of Delivery & frequency of Class:	15 weeks; 14x 4hrs sessions of lectures & practicals
Method & Criteria of Assessment:	60% 2½ Hours Exam & 40% Coursework

Module Aims:

- Identify, analyse and assess threats pertinent to information systems and introduce the need for data and network security
- To illustrate how cryptography has evolved through the years
- To describe the different techniques that can be used to enforce security in information systems.

Learning Objectives and Outcomes:

- Understand the basic definitions and principles in computer and network security.
- Understand use of cryptography in computer and network security.
- Understand how network threats and attacks work and how to defend against them
- Understand further issues in wireless security.

LECTURE SCHEDULE

Week	Dates	Topics
1	16/08/08	Overview: Security Needs, Security Services; ITU-T X.800
2	23/08/08	Overview (cont.): Security Mechanisms and Protocols
3	30/08/08	Classical cryptography: Monoalphabetic, polyalphabetic, transposition ciphers, product ciphers and rotor machines.
4	06/09/08	Modern cryptography: Block vs. Stream ciphers, Feistel Cipher, DES: ECB, CBC, CFB, OFB, 3-DES
5	13/09/08	Modern cryptography: Rjindael, AES: operation and implementation.
6	20/09/08	Public Key Encryption: Number Theory, Euler Totient Function, Chinese Remainder Theorem, Primality Testing: Miller- Rabin.
7	27/09/08	RSA algorithm, implementation and security
8	04/10/08	**Intensive Tutoring**
9	11/10/08	Class Test
10	18/10/08	Authentication: MAC and Hash functions: SHA-1 implementation and security, Difffie-Hellman Key Algorithm and Exchange, Digital Signatures.
11	25/10/08	Kerberos and X.509 Certificates.
12	All Saint's Day	No Classes
13	08/11/08	Computer Malware, Network defences, SSL and IPSec.
14	15/10/08	Revision+Tutorial
15	22/10/08	Revision+Tutorial

READING LIST

RECOMMENDED TEXTS (as per availability in the UTM Resource Centre)**:**

- **Stallings. W (1994) *Cryptography and Network Security, 2nd Edition*, Prentice-Hall (D4.6STA)**
- Kahate A. (2003) *Network Security and Cryptography,* Mc-Graw-Hill (D4.6KAH)

OTHER READING MATERIALS e.g. TEXTS/JOURNALS/ARTICLES/WEBSITES:

- *X.800 - Security architecture for Open Systems Interconnection for CCITT application:* **http://fag.grm.hia.no/IKT7000/litteratur/paper/x800.pdf**

LECTURE NOTES

The lecture notes are available on my external website: **Nefertum's Shrine** at http://pages.intnet.mu/rhh They are also available on the UTM Academic Staff **INTR@WEB** (campus-access only) at http://intraweb and clicking on my link or to access it directly at http://intraweb/~rh

The notes are in **.pdf** format so you will need **Adobe Acrobat® Reader** to view them. This reader can also be downloaded from the two above-mentioned sites in the **Downloads** Section.

SECU 2102
Network Security

Overview

What is this module about?

This module is to address

- security needs
- security services
- security mechanisms and protocols

for data stored in computers and transmitted across computer networks

What security is about in general?

- Security is about protection of assets
- Prevention
  - take measures to prevent assets from being tampered (or stolen)
- Detection
  - take measures so that you can detect when, how, and by whom an asset has been tampered
- Reaction
  - take measures to recover assets

Real-world example

- Prevention
  - locks at doors, window bars, secure the walls around the property, hire a guard, Dobermans…
- Detection
  - missing items, burglar alarms, CCTV, …
- Reaction
  - attack on burglar, call the police, replace stolen items, make an insurance claim, …

Internet shopping example

- Prevention
  - encrypt your order and card number, enforce merchants to do some extra checks, don't send card number via Internet
- Detection
  - an unauthorized transaction appears on your credit card statement
- Reaction
  - complain, dispute, ask for a new card number, sue
  - Or, pay and forget

Information security: Past & Present

- Traditional Information Security
  - keep the cabinets locked
  - put them in a secure room
  - human guards
  - electronic surveillance systems
  - i.e. Physical and Administrative mechanisms
- Modern World
  - Data is found inside computers in digital format
  - Computers are interconnected
  - **Hence computer and network security required**

Some Terminologies

- Computer Security
  - automated tools and mechanisms to protect data **in** a computer, even if the computers are connected to a network e.g.
    - against hackers (intrusion)
    - against viruses
- Network Security
  - measures to prevent, detect, and correct security violations that involve the **transmission** of information in a network

Services, Mechanisms, Attacks

- 3 aspects of information security:
  - security attacks (and threats)
    - actions that compromise security
  - security services
    - services counter to attacks
  - security mechanisms
    - used by services
    - E.g. secrecy is a service, encryption is the mechanism

Attacks

- Attacks on computer systems
  - break-in to destroy information
  - break-in to steal information
  - blocking to operate properly
  - malicious software (malware)
    - wide spectrum of problems (more later)

Attacks

- Network Security Attacks
  - **Passive** and **Active**
- Passive attacks
  - intercept messages by *sniffing* or *snooping*.
  - What can the attacker do?
    - use information internally ("fetiche")
    - release the content ("palabre")
    - traffic analysis ("veille mouvement")
  - Hard to detect, try to prevent… How?

Attacks

- Active attacks involves interruption, modification, fabrication, deletion of messages.
  - Masquerade/Spoofing (attack on authentication)
    - pretend to be someone else to perform an illegitimate action
  - Insertion/Fabrication (attack on integrity and/or authentication)
    - create a bogus message usually via spoofing
  - Replay (attack on authentication and/or integrity and/or availability)
    - passively capture data and send later

Attacks

- Active attacks
  - Deny (attack on non-repudiation)
    - Refuse to acknowledge sending/receiving a message
  - Modification (attack on integrity)
    - change the content of a message
  - Denial-Of-Service (attack on availability)
    - prevention the normal use of servers, end users, or network itself

Security Services

- to deter or detect attacks
- to enhance security
- replicate functions of physical documents
  - have signatures, dates, seals, watermark
  - protection from disclosure, tampering, or destruction
  - notarize
  - record

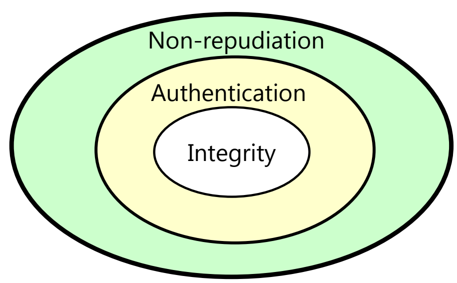
ISO 7498-2 Security Services

- Authentication
  - Assurance of the identity of the communicating entity
    - peer-entity authentication
      - mutual confidence in the identities of the parties involved in a connection
    - data-origin authentication
      - assurance about the source of the received data
- Access Control
  - prevention of the unauthorized use of a resource
- Data Confidentiality
  - protection of data from unauthorized disclosure
  - Network confidentiality is one step ahead

ISO 7498-2 Security Services

- Data Integrity
  - assurance that data received is exactly the same at the time sent by an authorized sender
  - i.e. no modification, insertion, deletion or replay
- Non-Repudiation
  - protection against denial by one of the parties in a communication
  - Origin non-repudiation
    - proof that the message was sent by the specified party
  - Destination non-repudiation
    - proof that the message was received by the specified party

Relationships



Security Mechanisms

- Basically cryptographic techniques/technologies
  - that serve to security services
  - to prevent/detect/recover attacks
- Encipherment (Encryption)
  - use of mathematical algorithms to transform data into a form that is not readily intelligible using ciphers
    - keys are involved

Security Mechanisms

- Message Digest
  - similar to encryption, but one-way (recovery not possible)
  - generally no keys are used
- Digital Signature & Message Authentication Code
  - Addition or Cryptographic transformation of a data unit to prove the source and the integrity of the data
- Authentication Exchange
  - ensure the identity of an entity by exchanging some information

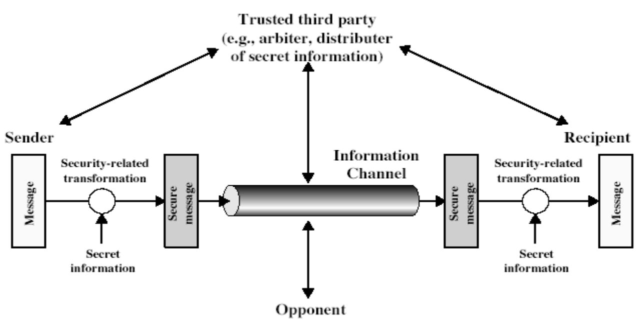
Security Mechanisms

- Notarization
  - use of a trusted third party to assure certain properties of a data exchange
- Time-stamping
  - inclusion of correct date and time within messages
- Non-cryptographic mechanisms
  - traffic padding (for traffic confidentiality)
  - Intrusion Detection Systems (more later)
  - Firewalls, Honeynet, Honeypot (more later)

Two Security references

- ITU-T X.800 Security Architecture for OSI
  - gives a systematic way of defining and providing security requirements
- RFC 2828
  - over 200 pages glossary on Internet Security

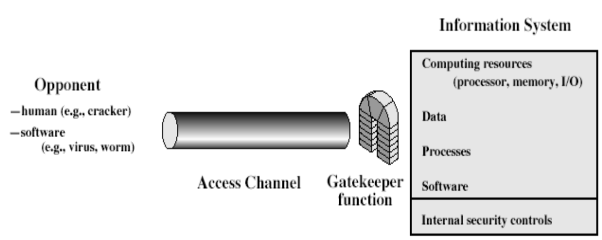
Model for Network Security



Model for Network Security

- This model requires the
  - design a suitable algorithm for the security transformation
  - generation the secret information (keys) used by the algorithm
  - Development of methods to distribute and share the secret information reliably
  - specify a protocol enabling the principals to use the transformation and secret information for a security service.

Model for Network Access Security



Model for Network Access Security

- This model requires the
  - Selection of appropriate gatekeeper functions to identify users and ensure only authorized users access designated information or resources
    - e.g. what you know, what you have, who you are
  - Internal control to monitor the activity and analyze information to detect intrusion.

More on Computer System Security

- Based on security policies
  - Set of rules that specify
    - How resources are managed to satisfy the security requirements
    - Which actions are permitted, which are not
  - Ultimate aim
    - Prevent security violations such as unauthorized access, data loss, service interruptions, etc.
  - Scope
    - Organizational or Individual
  - Implementation
    - Partially automated, but mostly humans are involved

Aspects of Computer Security

- Mostly related to Operating Systems
- Similar to those discussed for Network Security
  - Confidentiality
  - Integrity
  - Availability
  - Authenticity
  - Accountability
  - Dependability

Aspects of Computer Security

- Confidentiality
  - Prevent unauthorised disclosure of information
  - Synonyms: Privacy and Secrecy
    - any differences? Let's discuss
- Integrity
  - In general, "make sure that everything is as it is supposed to be"
  - Specifically, "no unauthorized modification or deletion"
- Availability
  - services should be accessible when needed and without delay

Aspects of Computer Security

- Accountability
  - audit information must be selectively kept and protected so that actions affecting security can be traced to the responsible party
  - How can we do that?
    - Users have to be **identified** and **authenticated** to have a basis for access control decisions and to find out responsible party in case of a violation.
    - The security system keeps an **audit log (audit trail)** of security relevant events to detect and investigate intrusions.
- Dependability
  - Can we trust the system as a whole?

Fundamental Tradeoff

- Between security and ease-of-use
- Security may require clumsy and inconvenient restrictions on users and processes

"If security is an add-on that people have to do something special to get, then most of the time they will not get it"

Martin Hellman,
co-inventor of Public Key Cryptography

Designing a successful product

- User-transparent
- Do not assume potential users to be security experts
  - but provide enough set of options for security experts
- a security feature in a product is a plus, but a security product is a challenge in the market
  - people intend to pay for secure products, but not to pay security products
- Homework: Prove or disprove the last bullet by making a search in the Internet.

SECU 2102 Network Security

Cryptography

Outline (Part 1)

- **Review classical cryptography**
  - Monoalphabetic Ciphers
  - Cryptanalysis using letter frequencies
  - Playfair ciphers
  - Polyalphabetic ciphers
  - Transposition ciphers
  - Product ciphers and rotor machines
- **Background on Block Ciphers**

Classical Cryptography

- Symmetric Encryption
  - sender and recipient share a common key
- **Some Terminology:**
  - **Plaintext** is the information which the sender wishes to transmit to recipient(s).
  - **Ciphertext** is the output of a encryption algorithm using the plaintext as input
  - **Key** is a shared, secret information used in the encryption algorithm
  - **Cryptanalysis** is the study of decrypting ciphertext without knowledge of the key. (codebreaking)

Requirements

- **Two requirements for secure use of symmetric encryption:**
  - a strong encryption algorithm
  - a secret key known only to sender / receiver

$$Y = E_K(X)$$

$$X = D_K(Y)$$

- *Assume encryption/decryption algorithms are known*
- *Assume a secure channel to distribute key*

Security Levels

- Unconditional security
  - no matter how much computer power is available, the cipher cannot be broken since the ciphertext provides insufficient information to uniquely determine the corresponding plaintext
- Computational security
  - given limited computing resources (e.g. time needed for calculations is greater than age of universe), the cipher cannot be broken

Classical Substitution Ciphers

- **Earliest known substitution cipher by Julius Caesar**
- **First attested use in military affairs**
- **Replaces each letter by 3rd letter on**
- **Example:**

```
will bomb the palace tonight
ZLOO ERPE WKH SDODFH WRQLJKW
```

Cryptanalysis of Rotation Cipher

- **only have 26 possible ciphers**
  - A maps to A,B,..Z
- **could simply try each in turn**
- **a** brute force search
- **given ciphertext, just try all shifts of letters**
- **do need to recognize when have plaintext**
- **E.g. break ciphertext "GCUA VQ DTGCM"**

Monoalphabetic Cipher

- **Rather than just shifting the alphabet could shuffle (jumble) the letters arbitrarily.**
- **Each plaintext letter maps to a different random ciphertext letter**
- **Hence key is 26 letters long**

```
Plain:  abcdefghijklmnopqrstuvwxyz
Cipher: DKVQFIBJWPESCXHTMYAUOLRGZN
Plaintext:  ifwewishtoreplaceletters
Ciphertext: WIRFRWAJUHYFTSDVFSFUUFYA
```

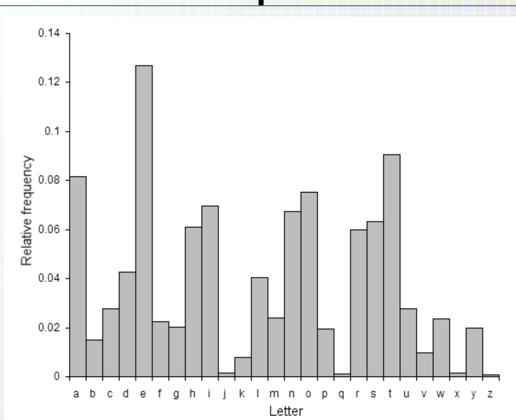

Monoalphabetic Cipher Security

- **now have a total of 26! = 4 x 10^{26} keys**
- **with so many keys, might think is secure**
- **but would be** !!!WRONG!!!
- **problem is language characteristics**

Language Redundancy & Cryptanalysis

- **Human languages are** redundant
- **E.g. "th lrd s m shphrd shll nt wnt"**
- **letters are not equally commonly used**
- **in English** e **is by far the most common letter**
- **then T,R,N,I,O,A,S**
- **other letters are fairly rare**
- **cf. Z,J,K,Q,X**
- **have tables of single, double & triple letter frequencies**

English Letter Frequencies



Use in Cryptanalysis

- **key concept - monoalphabetic substitution ciphers do not change relative letter frequencies**
- **discovered by Arabian scientists in 9th century**
- **calculate letter frequencies for ciphertext**
- **compare counts/plots against known values**
- **if Caesar cipher look for common peaks/troughs**
  - peaks at: TH pair, ON pair, ION triple, etc…
  - troughs at: JK, X-Z
- **for monoalphabetic must identify each letter**
  - tables of common double/triple letters help

Example Cryptanalysis

- **given ciphertext:**

UZQSOVUOHXMOPVGPOZPEVSGZWSZOPFPESXUDBME TXAIZVUEPHZHMDZSHZOWSFPAPPDTSVPQUZWYMXU ZUHSXEPYEPOPDZSZUFPOMBZWPFUPZHMDJUDTMOH MQ

- **count relative letter frequencies**
  - P 13.33, Z 11.67
  - S 8.33, U 8.33, O 7.50 M 6.67
  - …
  - C,K,L,N,R 0.00

Example Cryptanalysis (cont)

UZQSOVUOHXMOPVGPOZPEVSGZWS

ZOPFPESXUDBMETXAIZVUEPHZHM

DZSHZOWSFPAPPDTSVPQUZWYMXU

ZUHSXEPYEPOPDZSZUFPOMBZWPF

UPZHMDJUDTMOHMQ

Homework for next lecture…

Playfair Cipher

- **not even the large number of keys in a monoalphabetic cipher provides security**
- **one approach to improving security was to encrypt multiple letters**
- **the** Playfair Cipher **is an example**
- **invented by Charles Wheatstone in 1854, but named after his friend Baron Playfair**

Playfair Key Matrix

- **5X5 matrix of letters based on a keyword**
- **fill in letters of keyword (sans duplicates)**
- **fill rest of matrix with other letters**
- **example using the keyword MONARCHY**

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


Encrypting and Decrypting

- **plaintext encrypted two letters at a time:**

1. if a pair is a repeated letter, insert a filler like ‘x’,e.g. "balloon" encrypts as "ba lx lo on"
2. if both letters fall in the same row, replace each with letter to right (wrapping back to start from end), e.g. “ar” encrypts as "RM"
3. if both letters fall in the same column, replace each with the letter below it (again wrapping to top from bottom), eg. “mu" encrypts to "CM"
4. otherwise each letter is replaced by the one in its row in the column of the other letter of the pair, eg. “hs" encrypts to "BP", and “ea" to "IM" or "JM" (as desired)

Security of the Playfair Cipher

- **Security much improved over monoalphabetic since have 26 x 26 = 676 digrams**
- **Need a 676 entry frequency table to analyse vs. 26 for monoalphabetic and correspondingly more ciphertext needed**
- **Playfair was widely used for many years by US & British military in WW1**
- **It** can **be broken, given a few hundred letters since still has much of plaintext structure**

Polyalphabetic Ciphers

- **Another approach to improving security is to use multiple cipher alphabets called** polyalphabetic substitution ciphers
- **This makes cryptanalysis harder with more alphabets to guess and flatter frequency distribution**
- **Use a key to select which alphabet is used for each letter of the message**
- **use each alphabet in turn and repeat from start after end of key is reached**

Vigenère Cipher

- **Simplest polyalphabetic substitution cipher is the** Vigenère Cipher **which is effectively multiple Caesar ciphers**
- **Key is multiple letters long K = k1 k2 ... k*d***
- **i^{th} letter specifies the i^{th} alphabet to use**
- **use each alphabet in turn repeat from start after *d* letters in message**
- **decryption simply works in reverse**

Example

- **write the plaintext out**
- **write the keyword repeated above it**
- **use each key letter as a Caesar cipher key**
- **encrypt the corresponding plaintext letter**
- **example using keyword *deceptive***

```
key:       deceptivedeceptivedeceptive
plaintext: wearediscoveredsaveyourself
ciphertext:ZICVTWQNGRZGVTWAVZHCQYGLMGJ
```


Security of Vigenère Ciphers

- **have multiple ciphertext letters for each plaintext letter hence letter frequencies are obscured but not totally lost**
- **start with letter frequencies**
  - see if look monoalphabetic or not
- **if not, then need to determine number of alphabets, since then can attack each.**

Kasiski Attack

- **Method developed by Babbage / Kasiski**
- **Repetitions in ciphertext give clues to period so find same plaintext an exact period apart which results in the same ciphertext of course, could also be random fluke.**
- **Example**

  Repeated "VTW" in previous example suggests size of 3 or 9 then attack each monoalphabetic cipher individually using same techniques as before.

Vernam Cipher (1918)

- **Key as long as and independent of the plaintext**
- **Works on binary data as opposed to letters**

  `Ci = Pi (xor) Ki`
- **where**
  - `Pi – ith binary digit of plaintext`
  - `Ki – ith digit of key`
  - `Ci – ith digit of ciphertext`
- Key needs to be long and random

One-Time Pad

- If a truly random key as long as the message is used, the cipher will be unconditionally secure.
- This key is called a One-Time pad
- It is unbreakable since ciphertext bears no statistical relationship to the plaintext
- Since for any plaintext & any ciphertext there exists a key mapping one to other
- The key can only be used once though
- Problem of safe distribution of key

Transposition Ciphers

- Now consider classical transposition or permutation ciphers
- These hide the message by rearranging the letter order without altering the actual letters used
- can recognise these since have the same frequency distribution as the original text

Rail Fence cipher

- Write message letters out diagonally over a number of rows then read off cipher row by row
- E.g. write message out as:

```
m e m a t r h t g p r y
 e t e f e t e o a a t
```

- giving ciphertext

MEMATRHTGPRYETEFETEOAAT

Product Ciphers

- Ciphers using substitutions or transpositions are not secure because of language characteristics
- Hence consider using several ciphers in succession to make harder, but:
  - two substitutions make a more complex substitution
  - two transpositions make more complex transposition
  - but a substitution followed by a transposition makes a new much harder cipher
- This is bridge from classical to modern ciphers

Rotor Machines

- Before modern ciphers, rotor machines were most common product cipher
- They were widely used in WW2
  - German Enigma, Allied Hagelin, Japanese Purple
- Implemented a very complex, varying substitution cipher using a series of cylinders, each giving one substitution, which rotated and changed after each letter was encrypted
- With 3 cylinders, this could yield $26^3=17576$ alphabets

Block vs Stream Ciphers

- block ciphers process messages in into blocks, each of which is then en/decrypted
- like a substitution on very big characters
  - 64-bits or more
- stream ciphers process messages a bit or byte at a time when en/decrypting
- many current ciphers are block ciphers

Block Cipher Principles

- **most symmetric block ciphers are based on a** Feistel Cipher Structure
- **needed since must be able to** decrypt **ciphertext to recover messages efficiently**
- **block ciphers look like an extremely large substitution**
- **would need table of 2^{64} entries for a 64-bit block**
- **instead create from smaller building blocks**
- **using idea of a product cipher**

Substitution-Permutation Ciphers

- **in 1949 Claude Shannon introduced idea of substitution-permutation (S-P) networks**
  - modern substitution-transposition product cipher
- **these form the basis of modern block ciphers**
- **S-P networks are based on the two primitive cryptographic operations we have seen before:**
  - *substitution* (S-box)
  - *permutation* (P-box)
- **provide *confusion* and *diffusion* of message**

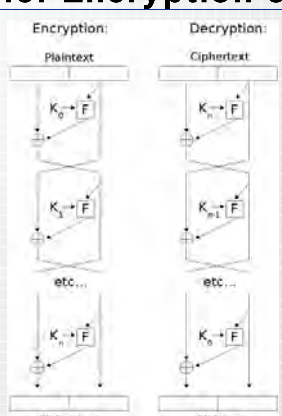
Confusion and Diffusion

- **cipher needs to completely obscure statistical properties of original message**
- **a one-time pad does this**
- **more practically Shannon suggested combining elements to obtain:**
- diffusion **– dissipates statistical structure of plaintext over bulk of ciphertext**
- confusion **– makes relationship between ciphertext and key as complex as possible**

Feistel Cipher Structure

- **Horst Feistel devised the** feistel cipher
  - based on concept of invertible product cipher
- **partitions input block into two halves**
  - process through multiple rounds which
  - perform a substitution on left data half
  - based on round function of right half & subkey
  - then have permutation swapping halves
- **implements Shannon's substitution-permutation network concept**

Feistel Cipher Encryption & Decryption



Feistel Cipher Design Principles

- block size
  - increasing size improves security, but slows cipher
- key size
  - increasing size improves security, makes exhaustive key searching harder, but may slow cipher
- number of rounds
  - increasing number improves security, but slows cipher
- subkey generation
  - greater complexity can make analysis harder, but slows cipher
- round function
  - greater complexity can make analysis harder, but slows cipher
- fast software en/decryption & ease of analysis
  - are more recent concerns for practical use and testing

Data Encryption Standard (DES)

- **most widely used block cipher in world**
- **adopted in 1977 by NBS (now NIST)**
  - as FIPS PUB 46
- **encrypts 64-bit data using 56-bit key**
- **has widespread use**
- **has been considerable controversy over its security**

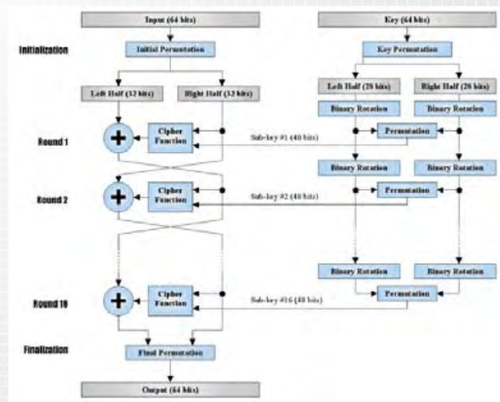
DES History

- **IBM developed Lucifer cipher**
  - by team led by Feistel
  - used 64-bit data blocks with 128-bit key
- **then redeveloped as a commercial cipher with input from NSA and others**
- **in 1973 NBS issued request for proposals for a national cipher standard**
- **IBM submitted their revised Lucifer which was eventually accepted as the DES**

DES Design Controversy

- **although DES standard is public**
- **was considerable controversy over design**
  - in choice of 56-bit key (vs Lucifer 128-bit)
  - and because design criteria were classified
- **subsequent events and public analysis show in fact design was appropriate**
- **DES has become widely used, especially in financial applications**

DES Encryption



Initial Permutation IP

- **first step of the data computation**
- **IP reorders the input data bits**
- **even bits to LH half, odd bits to RH half**
- **quite regular in structure (easy in h/w)**

example:

```
IP(675a6967 5e5a6b5a) = (ffb2194d 004df6fb)
```

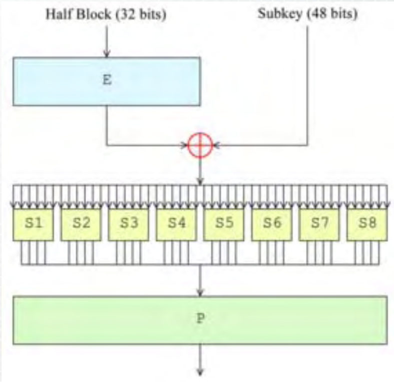

DES Round Structure

- **uses two 32-bit L & R halves**
- **as for any Feistel cipher can describe as:**

  $L_i = R_{i-1}$

  $R_i = L_{i-1} \text{ xor } F(R_{i-1}, K_i)$
- **takes 32-bit R half and 48-bit subkey and:**
  - expands R to 48-bits using perm E
  - adds to subkey
  - passes through 8 S-boxes to get 32-bit result
  - finally permutes this using 32-bit perm P

DES Round Structure



Substitution Boxes S

- **have eight S-boxes which map 6 to 4 bits**
- **each S-box is actually 4 little 4 bit boxes**
  - outer bits 1 & 6 (**row** bits) select one rows
  - inner bits 2-5 (**col** bits) are substituted
  - result is 8 lots of 4 bits, or 32 bits
- **row selection depends on both data & key**
  - feature known as autoclaving (autokeying)
- **example:**

`S(18 09 12 3d 11 17 38 39) = 5fd25e03`

DES Key Schedule

- **forms subkeys used in each round**
- **consists of:**
  - initial permutation of the key (PC1) which selects 56-bits in two 28-bit halves
  - 16 stages consisting of:
    - selecting 24-bits from each half
    - permuting them by PC2 for use in function f,
    - rotating **each half** separately either 1 or 2 places depending on the **key rotation schedule** K

DES Decryption

- **decrypt must unwind steps of data computation with Feistel design, do encryption steps again using subkeys in reverse order (SK16 … SK1)**

- **Note that IP undoes final FP step of encryption: 1st round with SK16 undoes 16th encrypt round 16th round with SK1 undoes 1st encrypt round then final FP undoes initial encryption IP thus recovering original data value**

Avalanche Effect

- **key desirable property of encryption algorithm**
- **where a change of** one **input or key bit results in changing approx** half **output bits**
- **making attempts to "home-in" by guessing keys impossible**
- **DES exhibits strong avalanche**

Strength of DES – Key Size

- **56-bit keys have $2^{56} = 7.2 \times 10^{16}$ values**
- **brute force search looks hard**
- **recent advances have shown is possible**
  - in 1997 on Internet in a few months
  - in 1998 on dedicated h/w (EFF) in a few days
  - in 1999 above combined in 22hrs!
- **still must be able to recognize plaintext**
- **now considering alternatives to DES**

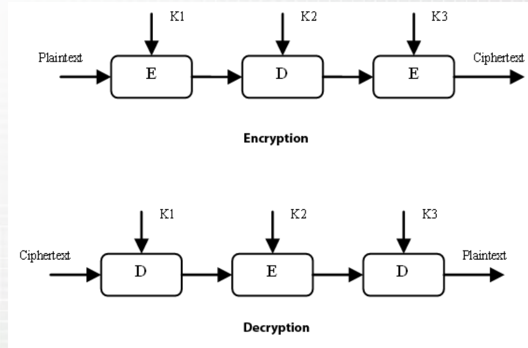
Triple DES (3DES)

- **Use three keys and three executions of the DES algorithm (encrypt-decrypt-encrypt)**

$$C = E_{K3}[D_{K2}[E_{K1}[P]]]$$

  - C = ciphertext
  - P = Plaintext
  - EK[X] = encryption of X using key K
  - DK[Y] = decryption of Y using key K
- **Effective key length of 168 bits**

Triple-DES



Block Cipher Design Principles

- **basic principles still like Feistel in 1970's**
- **number of rounds**
  - more is better, exhaustive search best attack
- **function f:**
  - provides "confusion", is nonlinear, avalanche
- **key schedule**
  - complex subkey creation, key avalanche

Modes of Operation

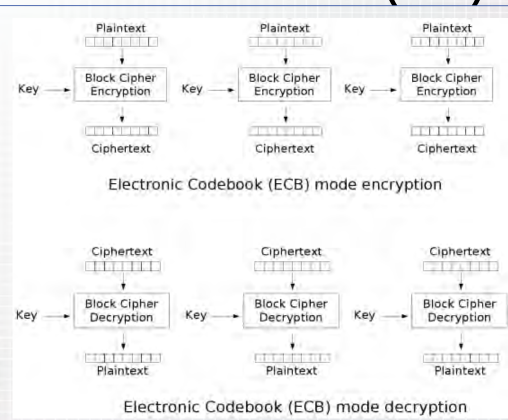
- **block ciphers encrypt fixed size blocks**
- **eg. DES encrypts 64-bit blocks, with 56-bit key**
- **need way to use in practise, given usually have arbitrary amount of information to encrypt**
- **four were defined for DES in ANSI standard** ANSI X3.106-1983 Modes of Use
- **subsequently now have 5 for DES and AES**
- **have** block **and** stream **modes**

Electronic Codebook Book (ECB)

- **message is broken into independent blocks which are encrypted**
- **each block is a value which is substituted, like a codebook, hence name**
- **each block is encoded independently of the other blocks**

  $C_i = DES_{K1}(P_i)$
- **uses: secure transmission of single values**

Electronic Codebook Book (ECB)



Advantages and Limitations of ECB

- **repetitions in message may show in ciphertext**
  - if aligned with message block
  - particularly with data such as graphics
  - or with messages that change very little, which become a code-book analysis problem
- **weakness due to encrypted message blocks being independent**
- **main use is sending a few blocks of data**

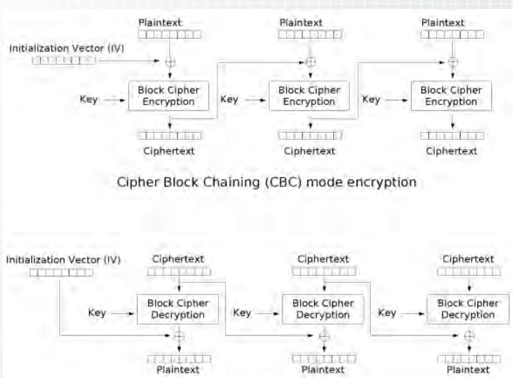
Cipher Block Chaining (CBC)

- **message is broken into blocks**
- **but these are linked together in the encryption operation**
- **each previous cipher blocks is chained with current plaintext block, hence name**
- **use Initial Vector (IV) to start process**

  $C_i = DES_{K1}(P_i \text{ XOR } C_{i-1})$

  $C_{-1} = IV$
- **uses: bulk data encryption, authentication**

Cipher Block Chaining (CBC)



Cipher Block Chaining (CBC) mode encryption

Cipher Block Chaining (CBC) mode decryption

Advantages and Limitations of CBC

- **each ciphertext block depends on** all **message blocks**
- **thus a change in the message affects all ciphertext blocks after the change as well as the original block**
- **need** Initial Value **(IV) known to sender & receiver**
  - however if IV is sent in the clear, an attacker can change bits of the first block, and change IV to compensate
  - hence either IV must be a fixed value (as in EFTPOS) or it must be sent encrypted in ECB mode before rest of message
- **at end of message, handle possible last short block**
  - by padding either with known non-data value (e.g. nulls)
  - or pad last block with count of pad size
    - eg. [b1 b2 b3 0 0 0 0 5] <- 3 data bytes, then 5 bytes pad+count

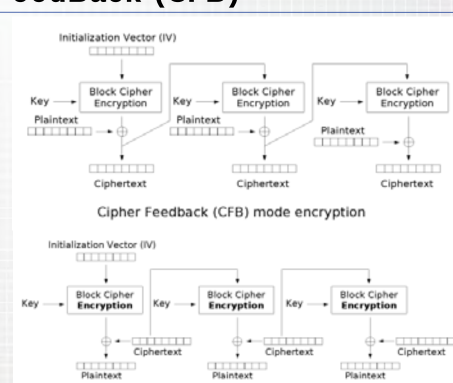
Cipher FeedBack (CFB)

- **message is treated as a stream of bits**
- **added to the output of the block cipher**
- **result is feed back for next stage (hence name)**
- **standard allows any number of bit (1,8 or 64 or whatever) to be feed back**
  - denoted CFB-1, CFB-8, CFB-64 etc
- **is most efficient to use all 64 bits (CFB-64)**

  $C_i = P_i \text{ XOR } DES_{K1}(C_{i-1})$

  $C_{-1} = IV$
- **uses: stream data encryption, authentication**

Cipher FeedBack (CFB)



Cipher Feedback (CFB) mode encryption

Cipher Feedback (CFB) mode decryption

Advantages and Limitations of CFB

- appropriate when data arrives in bits/bytes
- most common stream mode
- limitation is need to stall while do block encryption after every n-bits
- note that the block cipher is used in encryption mode at both ends
- errors propogate for several blocks after the error

Output FeedBack (OFB)

- message is treated as a stream of bits
- output of cipher is added to message
- output is then feed back (hence name)
- feedback is independent of message
- can be computed in advance

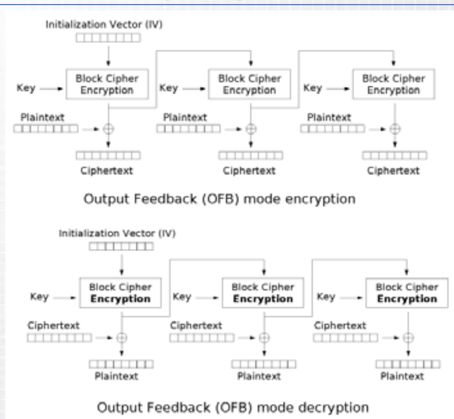
  $C_i = P_i \text{ XOR } O_i$

  $O_i = DES_{K1}(O_{i-1})$

  $O_{-1} = IV$

- uses: stream encryption over noisy channels

Output FeedBack (OFB)



Output Feedback (OFB) mode encryption

Output Feedback (OFB) mode decryption

Advantages and Limitations of OFB

- used when error feedback a problem or where need to encryptions before message is available
- superficially similar to CFB
- but feedback is from the output of cipher and is independent of message
- a variation of a Vernam cipher
  - hence must **never** reuse the same sequence (key+IV)
- sender and receiver must remain in sync, and some recovery method is needed to ensure this occurs
- originally specified with m-bit feedback in the standards
- subsequent research has shown that only OFB-64 should ever be used

Summary

have considered:

- **Block cipher design principles**
- **DES**
  - details
  - strength
- **TRIPLE DES**
- **Modes of Operation**
  - ECB, CBC, CFB, OFB

Network Security
SECU 2102

Cryptography (Cont.)

Origins of AES

- clearly a replacement for DES was needed
  - **have theoretical attacks that can break it**
  - **have demonstrated exhaustive key search attacks**
- can use Triple-DES – but slow, has small blocks
- US NIST issued call for ciphers in 1997
- 15 candidates accepted in Jun 98
- 5 were shortlisted in Aug 99
- Rijndael was selected as the AES in Oct 2000
- Issued as FIPS PUB 197 standard in Nov 2001

Shortlist

- After testing and evaluation, shortlist in Aug 99:
  - **MARS (IBM) - complex, fast, high security margin**
  - **RC6 (USA) - v. simple, v. fast, low security margin**
  - **Rijndael (Belgium) - clean, fast, good security margin**
  - **Serpent (Euro) - slow, clean, v. high security margin**
  - **Twofish (USA) - complex, v. fast, high security margin**
- Then subject to further analysis and comment by contrasting between algorithms with
  - **Few complex rounds vs. many simple rounds**
  - **Existing ciphers vs. new proposals**

AES Evaluation Criteria

- **initial criteria:**
  - security – effort to practically cryptanalyse
  - cost – computational
  - algorithm & implementation characteristics
- **final criteria**
  - general security
  - software & hardware implementation ease
  - implementation attacks
  - flexibility (in en/decrypt, keying, other factors)

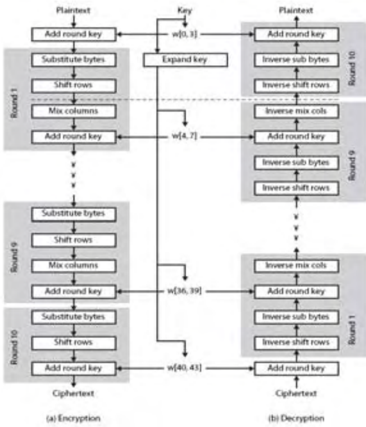
The AES Cipher - Rijndael

- designed by Rijmen-Daemen in Belgium
- has 128/192/256 bit keys, 128 bit data
- private key symmetric block cipher
- stronger & faster than Triple DES
- active life of 20-30 years (+ archival use)
- provide full specification, design & implementations
- an iterative rather than Feistel cipher
  - **processes data as block of 4 columns of 4 bytes**
  - **operates on entire data block in every round**
- designed to be:
  - **resistant against known attacks**
  - **speed and code compactness on many CPUs**
  - **design simplicity**

Rijndael

- data block of 4 columns of 4 bytes is **State**
- key is expanded to array of words
- has 9/11/13 rounds in which **State** undergoes:
  - **byte substitution (1 S-box used on every byte)**
  - **shift rows (permute bytes between groups/columns)**
  - **mix columns (subs using matrix multiply of groups)**
  - **add round key (XOR state with key material)**
  - **view as alternating XOR key & scramble data bytes**
- initial XOR key material & incomplete last round
- with fast XOR & table lookup implementation

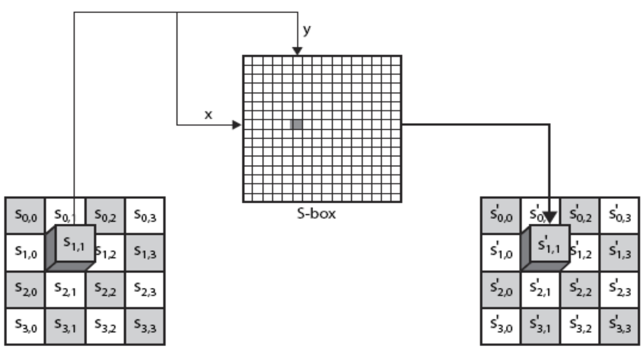
Rijndael



Byte Substitution

- a simple substitution of each byte
- uses one table of 16x16 bytes containing a permutation of all 256 8-bit values
- each byte of state is replaced by byte indexed by row (left 4-bits) & column (right 4-bits)
  - **E.g. byte {95} is replaced by byte in row 9 column 5**
  - **which has value {2A}**
- S-box constructed using defined transformation of values in GF(2^8)
- designed to be resistant to all known attacks

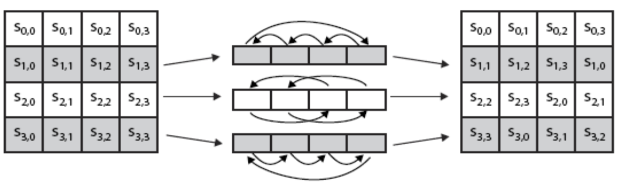
Byte Substitution



Shift Rows

- a circular byte shift in each
  - **1st row is unchanged**
  - **2nd row does 1 byte circular shift to left**
  - **3rd row does 2 byte circular shift to left**
  - **4th row does 3 byte circular shift to left**
- decrypt inverts using shifts to right
- since state is processed by columns, this step permutes bytes between the columns

Shift Rows

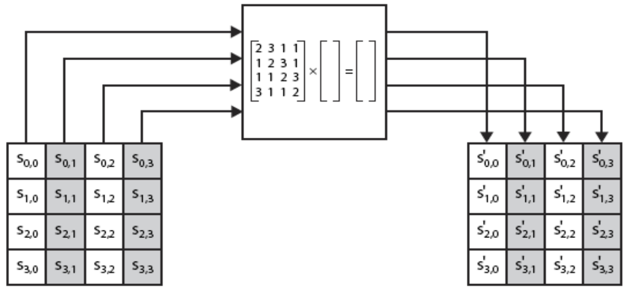


Mix Columns

- each column is processed separately
- each byte is replaced by a value dependent on all 4 bytes in the column
- effectively a matrix multiplication in GF(2^8) using prime polynomial m(x) $= x^8 + x^4 + x^3 + x + 1$

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix} = \begin{bmatrix} s'_{0,0} & s'_{0,1} & s'_{0,2} & s'_{0,3} \\ s'_{1,0} & s'_{1,1} & s'_{1,2} & s'_{1,3} \\ s'_{2,0} & s'_{2,1} & s'_{2,2} & s'_{2,3} \\ s'_{3,0} & s'_{3,1} & s'_{3,2} & s'_{3,3} \end{bmatrix}$$

Mix Columns



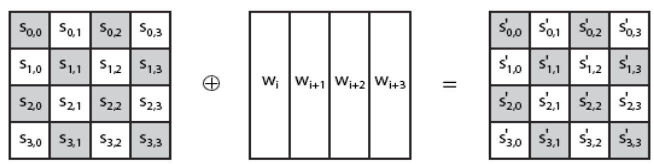
Mix Columns

- can express each column as 4 equations
  - to derive each new byte in column
- decryption requires use of inverse matrix
  - with larger coefficients, hence a little harder
- have an alternate characterization
  - each column a 4-term polynomial
  - with coefficients in $GF(2^8)$
  - and polynomials multiplied modulo (x^4+1)

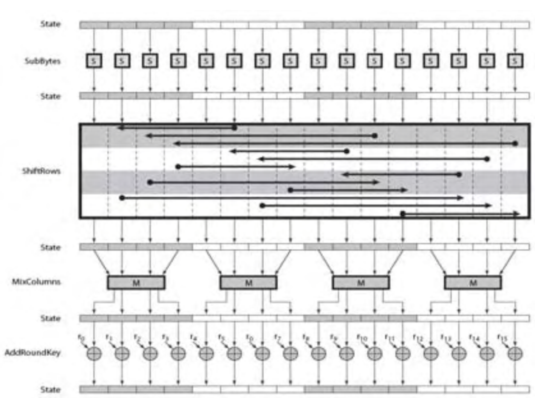
Add Round Key

- XOR state with 128-bits of the round key
- again processed by column (though effectively a series of byte operations)
- inverse for decryption identical
  - since XOR own inverse, with reversed keys
- designed to be as simple as possible
  - a form of Vernam cipher on expanded key
  - requires other stages for complexity / security

Add Round Key



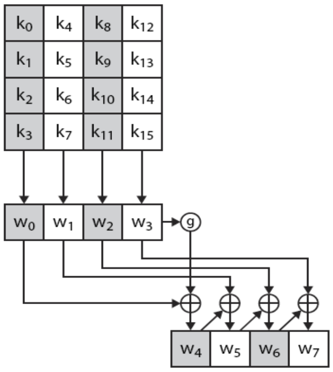
AES Round



AES Key Expansion

- takes 128-bit (16-byte) key and expands into array of 44/52/60 32-bit words
- start by copying key into first 4 words
- then loop creating words that depend on values in previous & 4 places back
  - in 3 of 4 cases just XOR these together
  - 1st word in 4 has rotate + S-box + XOR round constant on previous, before XOR 4th back

AES Key Expansion



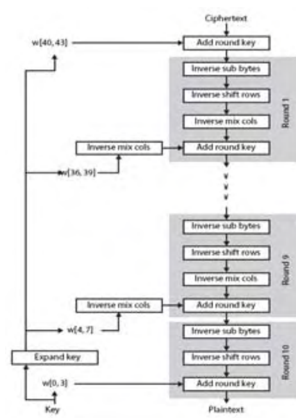
Key Expansion Rationale

- designed to resist known attacks
- design criteria included:
  - **knowing part key insufficient to find many more**
  - **invertible transformation**
  - **fast on wide range of CPU's**
  - **use round constants to break symmetry**
  - **diffuse key bits into round keys**
  - **enough non-linearity to hinder analysis**
  - **simplicity of description**

AES Decryption

- AES decryption is not identical to encryption since steps done in reverse
- but can define an equivalent inverse cipher with steps as for encryption
  - **but using inverses of each step**
  - **with a different key schedule**
- works since result is unchanged when
  - **swap byte substitution & shift rows**
  - **swap mix columns & add (tweaked) round key**

AES Decryption



Implementation Aspects

- can efficiently implement on 32-bit CPU
  - **redefine steps to use 32-bit words**
  - **can pre-compute 4 tables of 256-words**
  - **then each column in each round can be computed using 4 table lookups + 4 XORs**
  - **at a cost of 4Kb to store tables**
- designers believe this very efficient implementation was a key factor in its selection as the AES cipher

Summary

- have considered:
  - **the AES selection process**
  - **the details of Rijndael – the AES cipher**
  - **looked at the steps in each round**
  - **the key expansion**
  - **implementation aspects**

Network Security

SECU2102

Public Key Encryption

Introduction to Number Theory

The Devil said to Daniel Webster: "Set me a task I can't carry out, and I'll give you anything in the world you ask for."

Daniel Webster: "Fair enough. Prove that for n greater than 2, the equation $a^n + b^n = c^n$ has no non-trivial solution in the integers."

They agreed on a three-day period for the labour, and the Devil disappeared.

At the end of three days, the Devil presented himself, haggard, jumpy, biting his lip. Daniel Webster said to him, "Well, how did you do at my task? Did you prove the theorem?'

"Eh? No . . . no, I haven't proved it."

"Then I can have whatever I ask for? Money? The Presidency?'

"What? Oh, that—of course. But listen! If we could just prove the following two lemmas—"

Prime Numbers

- prime numbers only have divisors of 1 and self
  - they cannot be written as a product of other numbers
  - note: 1 is prime, but is generally not of interest
- eg. 2,3,5,7 are prime, 4,6,8,9,10 are not
- prime numbers are central to number theory
- list of prime number less than 200 is:

```
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61
67 71 73 79 83 89 97 101 103 107 109 113 127 131
137 139 149 151 157 163 167 173 179 181 191 193 197
199
```


Prime Factorisation

- to **factor** a number `n` is to write it as a product of other numbers: `n = a x b x c`
- note that factoring a number is relatively hard compared to multiplying the factors together to generate the number
- the **prime factorisation** of a number `n` is when its written as a product of primes
  - e.g. `91=7x13 ; 3600=2`^{4}`x3`^{2}`x5`^{2}

$$a = \prod_{p \in \mathrm{P}} p^{a_p}$$

Relatively Prime Numbers & GCD

- two numbers `a, b` are **relatively prime** if have **no common divisors** apart from 1
  - eg. 8 & 15 are relatively prime since factors of 8 are 1,2,4,8 and of 15 are 1,3,5,15 and 1 is the only common factor
- conversely can determine the greatest common divisor by comparing their prime factorizations and using least powers
  - E.g. $300=2^1 x 3^1 x 5^2$ $18=2^1 x 3^2$ hence $GCD(18,300)=2^1 x 3^1 x 5^0=6$

Fermat's (Little) Theorem

- $a^{p-1} \equiv 1 \pmod{p}$
  - where `p` is prime and `gcd(a,p)=1`
- also known as Fermat's Little Theorem
- also $a^p \equiv a \pmod{p}$
- useful in public key and primality testing

Euler Totient Function ø(n)

- when doing arithmetic modulo n
- **complete set of residues** is: `0..n-1`
- **reduced set of residues** is those numbers (residues) which are relatively prime to n
  - E.g. for n=10,
  - complete set of residues is {0,1,2,3,4,5,6,7,8,9}
  - reduced set of residues is {1,3,7,9}
- number of elements in reduced set of residues is called the **Euler Totient Function ø(n)**

Euler Totient Function ø(n)

- to compute ø(n) need to count number of residues to be excluded
- in general need prime factorization, but
  - for p (p prime) `ø(p) = p-1`
  - for p.q (p,q prime) `ø(pq) =(p-1)x(q-1)`
- E.g.

  `ø(37) = 36`

  `ø(21) = (3–1)x(7–1) = 2x6 = 12`

Euler's Theorem

- a generalisation of Fermat's Theorem
- $a^{ø(n)} = 1 \pmod n$
  - for any `a,n` where `gcd(a,n)=1`
- E.g.

  $a=3;n=10;$ `ø(10)=4;`

  hence `81 ≡ 1 mod 10`

  $a=2;n=11;$ `ø(11)=10;`

  hence `1024 ≡ 1 mod 11`

Primality Testing

- often need to find large prime numbers
- traditionally **sieve** using **trial division**
  - i.e. divide by all numbers (primes) in turn less than the square root of the number
  - only works for small numbers
- alternatively can use **statistical** primality tests based on properties of primes
  - for which all primes numbers satisfy property
  - but some composite numbers, called pseudo-primes, also satisfy the property (nb Carmichael numbers e.g. 561, 1729,...)
- can use a slower deterministic primality test e.g. AKS test

Miller-Rabin Algorithm

- a test based on Fermat's Little Theorem
- Algorithm:

  TEST (any N where $N>2$ and obviously odd) is:

  1. Find integers k, q; $k>0$, q odd, so that $(N-1)=2^k q$
  2. Select random integer a, where $1<a<N$ and $GCD(a,N)=1$
  3. **if** $a^q \bmod N = 1$ **then** return ("maybe prime");
  4. **for** $j=0$ **to** $k-1$ **do**
     5. **if** $(a^{2^j q} \bmod N = N-1)$

        then return (" maybe prime ")
  6. return ("composite")

Probabilistic Considerations

- if Miller-Rabin returns "composite" the number is definitely not prime
- otherwise it is a prime or a pseudo-prime
- chance it detects a pseudo-prime is < $^1/_4$
- hence if repeat test with different random a then chance n is prime after t tests is:
  - P(n prime after t tests) = $1-4^{-t}$
  - E.g. for t=10 this probability is > 0.99999

Prime Distribution

- prime number theorem states that primes occur roughly every (`ln(n)`) integers
- but can immediately ignore evens
- so in practice need only test `ln(n)/2` numbers of size `n` to locate a prime
  - note this is only the "average"
  - sometimes primes are close together
  - other times are quite far apart

Chinese Remainder Theorem

- used to speed up modulo computations
- if working modulo of product of numbers
  - e.g. mod $M = m_1 m_2 .. m_k$
- Chinese Remainder theorem lets us work in each moduli m_i separately
- since computational cost is proportional to size, this is faster than working in the full modulus M

Chinese Remainder Theorem

- can implement CRT in several ways
- to compute A mod M
  - first compute all $a_i = A \bmod m_i$ separately
  - determine constants c_i below, where $M_i = M/m_i$
  - then combine results to get answer using:

$$A \equiv \left(\sum_{i=1}^{k} a_i c_i\right) (\text{mod } M)$$

$$c_i = M_i \times (M_i^{-1} \bmod m_i) \quad \text{for } 1 \le i \le k$$

Primitive Roots

- from Euler's theorem have $a^{ø(n)} \bmod n = 1$
- consider $a^m = 1 \pmod n$, $GCD(a,n)=1$
  - must exist for $m = ø(n)$ but may be smaller
  - once powers reach m, cycle will repeat
- if smallest is $m = ø(n)$ then `a` is called a **primitive root**
- if `p` is prime, then successive powers of `a` "generate" the group mod p
- these are useful but relatively hard to find

Discrete Logarithms

- the inverse problem to exponentiation is to find the **discrete logarithm** of a number modulo p
- that is to find x such that $y = g^x \pmod p$
- this is written as $x = \log_g y \pmod p$
- if g is a primitive root then it always exists, otherwise it may not, e.g.
  - $x = \log_3 4 \bmod 13$ has no answer
  - $x = \log_2 3 \bmod 13 = 4$ by trying successive powers
- whilst exponentiation is relatively easy, finding discrete logarithms is generally a **hard** problem

Summary

- have considered:
  - prime numbers
  - Fermat's and Euler's Theorems
  - Primality Testing
  - Chinese Remainder Theorem
  - Discrete Logarithms

Public Key Cryptography and RSA

Every Egyptian received two names, which were known respectively as the true name and the good name, or the great name and the little name; and while the good or little name was made public, the true or great name appears to have been carefully concealed.

—*The Golden Bough*, Sir James George Frazer

Private-Key Cryptography

- traditional **private/secret/single key** cryptography uses **one** key
- shared by both sender and receiver
- if this key is disclosed communications are compromised
- also is **symmetric**, parties are equal
- hence does not protect sender from receiver forging a message & claiming is sent by sender

Public-Key Cryptography

- probably most significant advance in the 3000 year history of cryptography
- uses **two** keys – a public & a private key
- **asymmetric** since parties are **not** equal
- uses clever application of a number of theoretical concepts to function
- complements **rather than** replaces private key cryptography

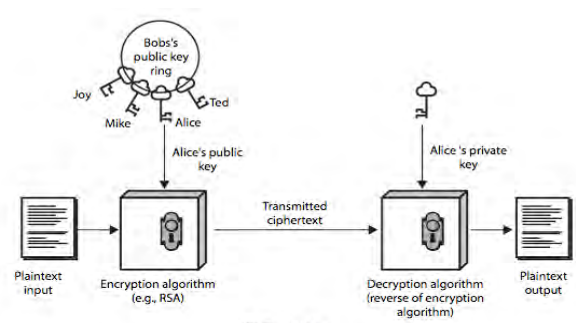
Why Public-Key Cryptography?

- developed to address two key issues:
  - **key distribution** – how to have secure communications in general without having to trust a KDC with your key
  - **digital signatures** – how to verify a message comes intact from the claimed sender
- public invention due to Whitfield Diffie & Martin Hellman at Stanford University in 1976
  - known earlier in classified community

Public-Key Cryptography

- **public-key/two-key/asymmetric** cryptography involves the use of **two** keys:
  - a **public-key**, which may be known by anybody, and can be used to **encrypt messages**, and **verify signatures**
  - a **private-key**, known only to the recipient, used to **decrypt messages**, and **sign** (create) **signatures**
- is **asymmetric** because
  - those who encrypt messages or verify signatures **cannot** decrypt messages or create signatures

Public-Key Cryptography

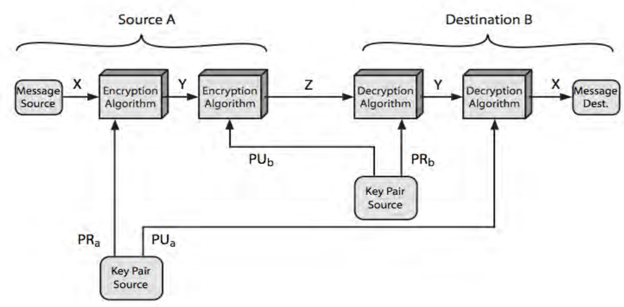


(a) Encryption

Public-Key Characteristics

- Public-Key algorithms rely on two keys where:
  - it is computationally infeasible to find decryption key knowing only algorithm & encryption key
  - it is computationally easy to en/decrypt messages when the relevant (en/decrypt) key is known
  - either of the two related keys can be used for encryption, with the other used for decryption (for some algorithms)

Public-Key Cryptosystems



Public-Key Applications

- can classify uses into 3 categories:
  - **encryption/decryption** (provide secrecy)
  - **digital signatures** (provide authentication)
  - **key exchange** (of session keys)
- some algorithms are suitable for all uses, others are specific to one

Security of Public Key Schemes

- like private key schemes brute force **exhaustive search** attack is always theoretically possible
- but keys used are too large (>512bits)
- security relies on a **large enough** difference in difficulty between **easy** (en/decrypt) and **hard** (cryptanalyse) problems
- more generally the **hard** problem is known, but is made hard enough to be impractical to break
- requires the use of **very large numbers**
- hence is **slow** compared to private key schemes

RSA

- by Rivest, Shamir & Adleman of MIT in 1977
- best known & widely used public-key scheme
- based on exponentiation in a finite (Galois) field over integers modulo a prime
  - nb exponentiation takes $O((\log n)^3)$ operations (easy)
- uses large integers (eg 1024 bits)
- security due to cost of factoring large numbers
  - nb factorization takes $O(e^{\log n \log \log n})$ operations (hard)

RSA Key Setup

- each user generates a public/private key pair by:
- selecting two large primes at random - `p,q`
- computing their system moduli `n=p.q`
  - note `ø(n)=(p-1)(q-1)`
- selecting at random the encryption key `e`
  - where `1<e<ø(n), gcd(e,ø(n))=1`
- solve following equation to find decryption key `d`
  - `e.d=1 mod ø(n) and 0≤d≤n`
- publish their public encryption key: PU={e,n}
- keep secret private decryption key: PR={d,n}

RSA Use

- to encrypt a message M the sender:
  - obtains **public key** of recipient PU={e,n}
  - computes: $C = M^e \bmod n$, where $0 \le M < n$
- to decrypt the ciphertext C the owner:
  - uses their private key PR={d,n}
  - computes: $M = C^d \bmod n$
- note that the message M must be smaller than the modulus n (block if needed)

Why RSA Works

- because of Euler's Theorem:
  - $a^{ø(n)} \bmod n = 1$ where gcd(a,n)=1
- in RSA have:
  - n=p.q
  - ø(n)=(p-1)(q-1)
  - carefully chose e & d to be inverses mod ø(n)
  - hence e.d=1+k.ø(n) for some k
- hence :

$$C^d = M^{e.d} = M^{1+k.ø(n)} = M^1.(M^{ø(n)})^k$$
$$= M^1.(1)^k = M^1 = M \bmod n$$

RSA Example - Key Setup

1. Select primes: $p = $ **17** & $q = $ **11**
2. Compute $n = pq = 17 \times 11 = $ **187**
3. Compute $ø(n) = (p-1)(q-1) = 16 \times 10 = $ **160**
4. Select e: GCD(e,160)= 1; choose $e = $ **7**
5. Determine d: $d.e = 1 \bmod 160$ and $d < 160$ Value is d=**23** since 23x7=161=1x160+1
6. Publish public key **PU={7,187}**
7. Keep secret private key **PR={23,187}**

RSA Example - En/Decryption

- sample RSA encryption/decryption is:
- given plaintext message **M=88** (n.b. 88<187)
- encryption:

  C $= 88^7 \bmod 187 = $ **11**
- decryption:

  M $= 11^{23} \bmod 187 = $ **88**

Exponentiation

- can use the Square and Multiply Algorithm
- a fast, efficient algorithm for exponentiation
- concept is based on repeatedly squaring base
- and multiplying in the ones that are needed to compute the result
- look at binary representation of exponent
- only takes $O(\log_2 n)$ multiples for number n
  - e.g. $7^5 = 7^4.7^1 = 3.7 = 10 \bmod 11$
  - e.g. $3^{129} = 3^{128}.3^1 = 5.3 = 4 \bmod 11$

Exponentiation

```
c = 0; f = 1
for i = k downto 0
    do c = 2 x c
       f = (f x f) mod n
    if b_i == 1 then
       c = c + 1
       f = (f x a) mod n
return f
```

Efficient Encryption

- encryption uses exponentiation to power e
- hence if e small, this will be faster
  - often choose e=65537 (2^{16}-1)
  - also see choices of e=3 or e=17
- but if e too small (e.g. e=3) can attack
  - using Chinese remainder theorem & 3 messages with different moduli
- if e fixed must ensure `gcd(e,ø(n))=1`
  - i.e. reject any p or q not relatively prime to e

Efficient Decryption

- decryption uses exponentiation to power d
  - this is likely large, insecure if not
- can use the Chinese Remainder Theorem (CRT) to compute mod p & q separately. then combine to get desired answer
  - approx 4 times faster than doing directly
- only owner of private key who knows values of p & q can use this technique

RSA Key Generation

- users of RSA must:
  - determine two primes at random - `p, q`
  - select either `e` or `d` and compute the other
- primes `p,q` must not be easily derived from modulus `n=p.q`
  - means must be sufficiently large
  - typically guess and use probabilistic test
- exponents `e, d` are inverses, so use Inverse algorithm to compute the other

RSA Security

- possible approaches to attacking RSA are:
  - brute force key search (infeasible given size of numbers)
  - mathematical attacks (based on difficulty of computing ø(n), by factoring modulus n)
  - timing attacks (on running of decryption)
  - chosen ciphertext attacks (given properties of RSA)

Factoring Problem

- mathematical approach takes 3 forms:
  - factor `n=p.q`, hence compute `ø(n)` and then d
  - determine `ø(n)` directly and compute d
  - find d directly
- currently believe all equivalent to factoring
  - have seen slow improvements over the years
    - as of May-05 best is 200 decimal digits (663) bit with LS
  - biggest improvement comes from improved algorithm
    - cf QS to GHFS to LS
  - currently assume 1024-2048 bit RSA is secure
    - ensure p, q of similar size and matching other constraints

Timing Attacks

- developed by Paul Kocher in mid-1990's
- exploit timing variations in operations
  - eg. multiplying by small vs large number
  - or IF's varying which instructions executed
- infer operand size based on time taken
- RSA exploits time taken in exponentiation
- countermeasures
  - use constant exponentiation time
  - add random delays
  - blind values used in calculations

Chosen Ciphertext Attacks

- RSA is vulnerable to a Chosen Ciphertext Attack (CCA)
- attackers chooses ciphertexts & gets decrypted plaintext back
- choose ciphertext to exploit properties of RSA to provide info to help cryptanalysis
- can counter with random pad of plaintext
- or use Optimal Asymmetric Encryption Padding (OASP)

Summary

- have considered:
  - principles of public-key cryptography
  - RSA algorithm, implementation, security

Network Security

SECU2102

Authentication

Authentication Protocols

- ***We cannot enter into alliance with neighbouring princes until we are acquainted with their designs.***

  —*The Art of War*, Sun Tzu

- **used to convince parties of each others identity and to exchange session keys**
- **may be one-way or mutual**
- **key issues are**
  - confidentiality – to protect session keys
  - timeliness – to prevent replay attacks

Message Authentication

- **message authentication is usually concerned with:**
  - protecting the integrity of a message
  - validating identity of originator
  - non-repudiation of origin (dispute resolution)
- **will consider the security requirements then three alternative functions used:**
  - message Encryption
  - Message Authentication Code (MAC)
  - Hash functions

Message Encryption

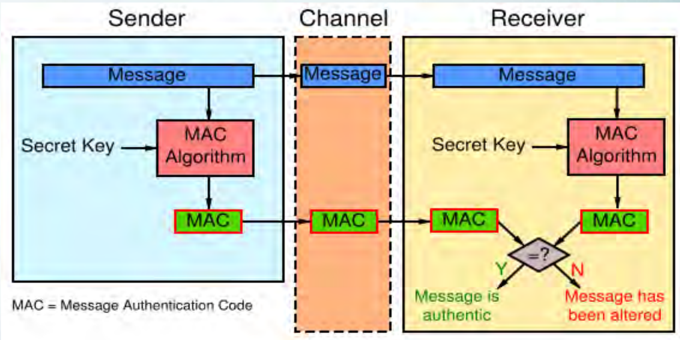
- **message** encryption **by itself also provides a measure of** authentication.
- **if symmetric** encryption **is used then:**
  - Receiver know sender must have created it
  - since only sender and receiver know key used
  - know content cannot of been altered
  - if message has suitable structure, redundancy or a checksum to detect any changes

Message Encryption

- **If public-key** encryption **is used:**
  - **encryption** provides no confidence of sender
  - since anyone potentially knows public-key
  - However if
    - sender **signs** message using their private-key
    - then encrypts with recipients public key
    - have both secrecy and **authentication**
  - Again **need** to recognize corrupted messages
  - but at cost of two public-key uses on message

Message Authentication Code (MAC)

- **generated by an algorithm that creates a small fixed-sized block**
  - depending on both message and some key
  - like encryption though need not be reversible
- **appended to message as a** signature
- **receiver performs same computation on message and checks it matches the MAC**
- **provides assurance that message is** unaltered **and** comes **from sender**



Message Authentication Codes

- ❖ **as shown the MAC provides authentication**
- ❖ **can also use encryption for secrecy**
  - generally use separate keys for each
  - can compute MAC either before or after encryption
  - is generally regarded as better done before
- ❖ **why use a MAC?**
  - sometimes only authentication is needed
  - sometimes need authentication to persist longer than the encryption (e.g. archival use)

MAC Properties

- ❖ **a MAC is a cryptographic checksum**

  `MAC = C`$_K$`(M)`
  - condenses a variable-length message M
  - using a secret key K
  - to a **fixed-sized** authenticator
- ❖ **is a many-to-one function**
  - potentially many messages have same MAC
  - but finding these needs to be very difficult

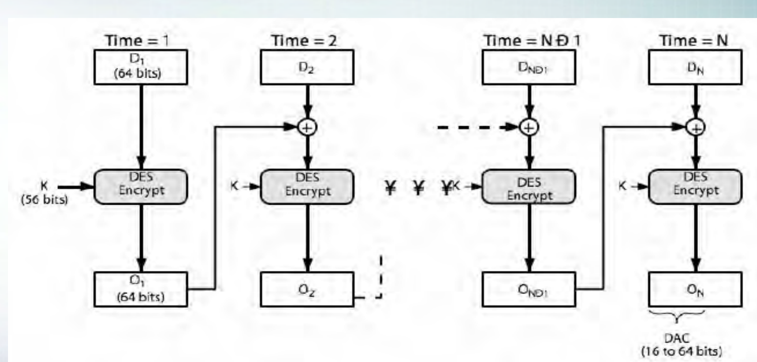
Requirements for MACs

- ❖ **Take into account the types of attacks**
- ❖ **Need the MAC to satisfy the following:**
  1. knowing a message and MAC, it is infeasible to find another message with same MAC
  2. MACs should be uniformly distributed
  3. MAC should depend equally on all bits of the message (i.e. Avalanche effect)

Using Symmetric Ciphers for MACs

- ❖ **can use any block cipher chaining mode and use final block as a MAC**
- ❖ Data Authentication Algorithm (DAA) **is a widely used MAC based on DES-CBC**
  - using IV=0 and zero-pad of final block
  - encrypt message using DES in CBC mode
  - and send just the final block as the MAC
    - or the leftmost M bits (16≤M≤64) of final block
- ❖ **but final MAC is now too small for security**

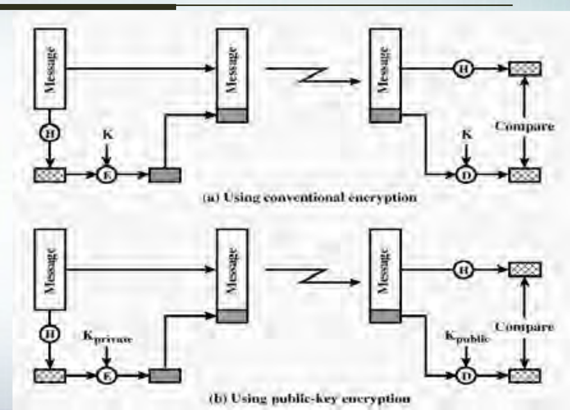
Data Authentication Algorithm



One Way Hash Functions

- ❖ Alternative to MAC
- ❖ As with MAC condenses arbitrary message to fixed size: h = H(M)
- ❖ usually assume that the hash function is public and not keyed
  - cf. MAC which is keyed
- ❖ hash used to detect changes to message
- ❖ can use in various ways with message
- ❖ most often to create a digital signature

One-way HASH function

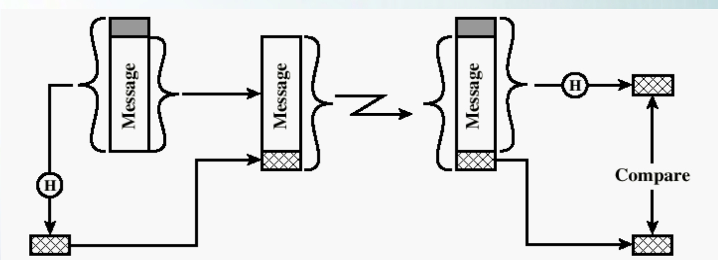


(a) Using conventional encryption

(b) Using public-key encryption

One-way HASH function

- ❖ Secret value is added before the hash and removed before transmission.



(c) Using secret value

Simple Hash Functions

- ❖ There are several proposals for simple functions
- ❖ Based on XOR of message blocks
- ❖ But predictability in data causes problems
- ❖ e.g. text which is in ASCII has leading 0
- ❖ not secure since can manipulate any message and either not change hash or change hash also
- ❖ need a stronger cryptographic function

Simple Hash Function

	bit 1	bit 2	• • •	bit n
block 1	b_{11}	b_{21}		b_{n1}
block 2	b_{12}	b_{22}		b_{n2}
	• • •	• • •	• • •	• • •
block m	b_{1m}	b_{2m}		b_{nm}
hash code	C_1	C_2		C_n

Figure 3.3 Simple Hash Function Using Bitwise XOR

- ❖ One-bit circular shift on the hash value after each block is processed would improve security

Secure Hash Functions

Purpose of the HASH function is to produce a “fingerprint.”
Properties of a HASH function *H* :

H can be applied to a block of data at any size and produces a fixed length output

H(x) is easy to compute for any given x.

One way property - For a given output h, it is computationally infeasible to find input x such that H(x) = h

Weak Collision Resistance Property - For any input x and respective output h, it is computationally infeasible to find another input y that yields the same output h i.e. H(y) = H(x).

Strong Collision Resistance Property - It is computationally infeasible to find two arbitrary inputs x and y that produces the same output h i.e H(x) = H(y) = h where x ≠ y

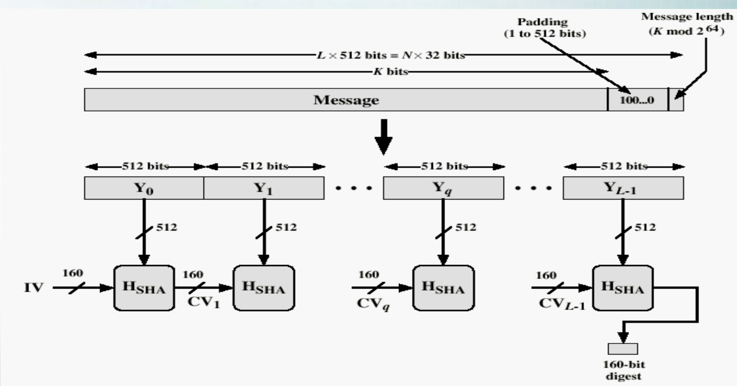
Secure Hash Algorithm (SHA-1)

- SHA was designed by NIST & NSA in 1993, revised 1995 as SHA-1
- US standard for use with DSA signature scheme
  - standard is FIPS 180-1 1995, also Internet RFC3174
  - Note: the algorithm is SHA, the standard is SHS
- produces 160-bit hash values
- now the generally preferred hash algorithm
- based on design of MD4 with key differences

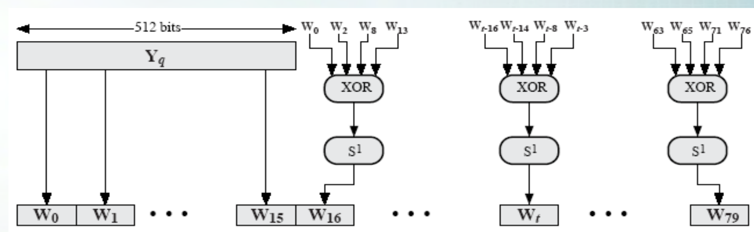
SHA-1 Overview

- pad message so its length is 448 mod 512
- append a 64-bit length value to message
- initialize 5-word (160-bit) buffer (A,B,C,D,E) to (67452301,efcdab89,98badcfe,10325476,c3d2e1f0)
- process message in 16-word (512-bit) chunks:
  - expand 16 words into 80 words by mixing & shifting
  - use 4 rounds of 20 bit operations on message block & buffer
  - add output to input to form new buffer value
- output hash value is the final buffer value

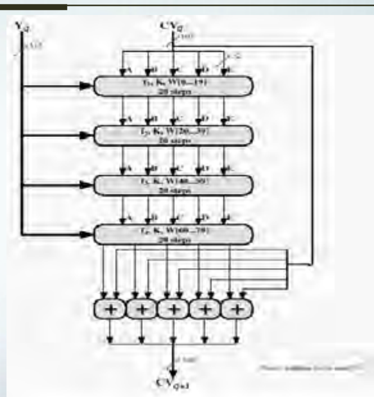
Message Digest creation Using SHA-1



Creation of 80-word Input Sequence for SHA-1 Processing of Single Block



SHA-1 Processing of single 512-Bit Block



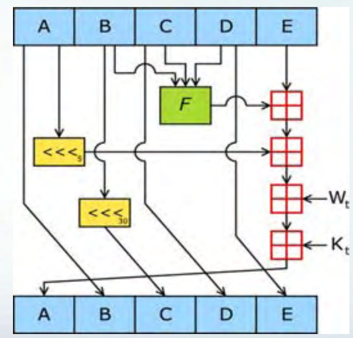
Output Stage of SHA-1

- After all 512 bit blocks have been processed, where
  - IV = initial vector, initial value of five words
  - L = number of 512 bit blocks in padded message
  - MD = final Message Digest
  - $ABCDE_q$ = output of last round of processing of the q^{th} block
- Algorithm
  - $CV_0 = IV$
  - for q = 0 to L
    - $CV_{q+1} = CV_q + ABCDE_q$
  - $MD = CV_L$

SHA-1 Compression Function

- each round has 20 steps which replaces the 5 buffer words thus:
  (A,B,C,D,E) <-
  (E+f(t,B,C,D)+(A<<5)+W_t+K_t),A,(B<<30),C,D)
- a,b,c,d refer to the 4 words of the buffer
- t is the step number
- f(t,B,C,D) is nonlinear function for round
- W_t is derived from the message block
- K_t is a constant value derived from step

SHA-1 Compression Function



One iteration within the SHA-1 compression function

A, B, C, D and E are 32-bit words of the state;

F is a nonlinear function that varies with the step;

<<<*n* denotes a left bit rotation by *n* places;

n varies for each operation.

⊞ denotes addition modulo 2^{32}

K_t is a constant which depends on the step

W_t is an XOR function that depends on the step

K_t - Constants for SHA-1 steps

Step Number	K_t (in hex)	Integer portion of
$0 \le t \le 19$	5A827999	2^{30} x sqrt(2)
$20 \le t \le 39$	6ED9EBA1	2^{30} x sqrt(3)
$40 \le t \le 59$	8F1BBCDC	2^{30} x sqrt(5)
$60 \le t \le 79$	CA62C1D6	2^{30} x sqrt(10)

f_t – Round Functions for steps (t)

Step Number	f_t(t,B,C,D)
$0 \le t \le 19$	(B AND C) OR (B’ AND D)
$20 \le t \le 39$	B XOR C XOR D
$40 \le t \le 59$	(B AND C) OR (B AND D) OR (C AND D)
$60 \le t \le 79$	B XOR C XOR D

Note: B’ means NOT B

W_t words

- 32 bit W_t words
- For the first 16 words W_t = 16 words of current block

Henceforth

$$W_t = S^1(W_{t-16} + W_{t-14} + W_{t-8} + W_{t-3})$$

Note: + means XOR

Other Secure HASH functions

	SHA-1	MD5	RIPEMD-160
Digest length	160 bits	128 bits	160 bits
Basic unit of processing	512 bits	512 bits	512 bits
Number of steps	80 (4 rounds of 20)	64 (4 rounds of 16)	160 (5 paired rounds of 16)
Maximum message size	2^{64}-1 bits	∞	∞

Keyed Hash Functions as MACs

- have desire to create a MAC using a hash function rather than a block cipher
  - because hash functions are generally faster
  - not limited by export controls unlike block ciphers
- hash includes a key along with the message
- original proposal:

  `KeyedHash = Hash(Key|Message)`
  - some weaknesses were found with this
- eventually led to development of HMAC

HMAC

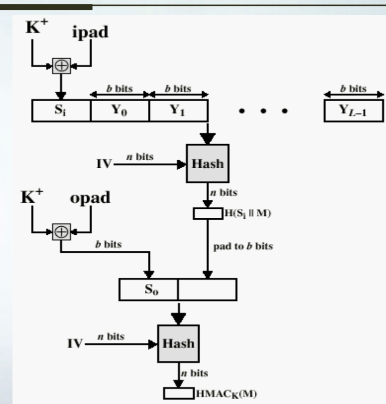
- Use a MAC derived from a cryptographic hash code, such as SHA-1.
- Motivations:
  - Cryptographic hash functions executes faster in software than encryption algorithms such as DES
  - Library code for cryptographic hash functions is widely available
  - No export restrictions from the US

HMAC Design Objectives

- Proposal to include secret key in hash function
- RFC 2104 lists design objectives for HMAC

1. To use available hash functions
2. Allow easy replaceability of hash function
3. Maintain performance of original hash
4. Use and handle keys simply
5. Have well understood cryptographic analysis of strength of the authentication method

HMAC Structure



HMAC Details

- Hash = embedded hash function (e.g., SHA-1)
- M – message
- L – number of blocks in M
- Y_i – the ith block of M 0 < i < L
- b = number of bits in a block
- n = length of hash code produced by embedded hash
- K = secret Key
- K+ = K padded on left with zeroes so length is b
- Ipad = 00110110 repeated b/8 times
- Opad = 01011100 repeated b/8 times

Other Public-Key Cryptographic Algorithms

- Digital Signature Standard (DSS)
  - Makes use of the SHA-1
  - Not for encryption or key echange
- Elliptic-Curve Cryptography (ECC)
  - Good for smaller bit size
  - Low confidence level, compared with RSA
  - Very complex

KERBEROS

In Greek mythology, a three headed dog, the guardian of the entrance of Hades

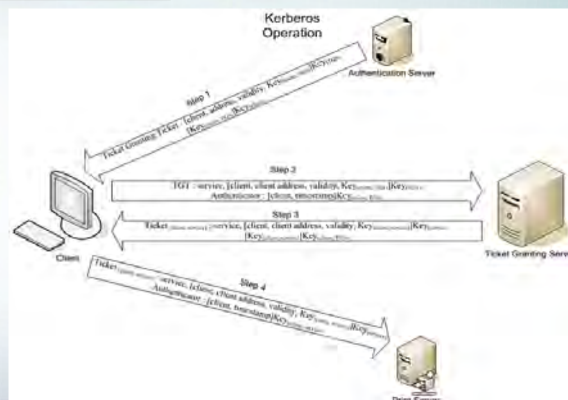
Approaches to Security in a Network

1. Rely on clients to assure the identity of users. Rely on server to enforce security policy on users.
2. Require clients to authenticate themselves to servers, trust clients to authenticate users.
3. Require the user to prove identity for each service requested. Also servers prove identity to clients.

Kerberos

- Trusted key server system from MIT
- provides centralised private-key third-party authentication in a distributed network
  - allows users access to services distributed through network
  - without needing to trust all workstations
  - rather all trust a central authentication server
- Two versions in use: 4 & 5
- Windows 2K, XP, Server 2003 & 2008 uses Kerberos as their default authentication method

Kerberos V5 overview



Detailed Interactions (1-6)

1. A user enters a username and password on the client.
2. The client performs a one-way hash on the entered password, and this becomes the secret key of the client.
3. The client sends a clear-text message to the AS requesting services on behalf of the user. Sample Message: "User XYZ would like to request services". Note: Neither the secret key nor the password is sent to the AS.
4. The AS checks to see if the client is in its database. If it is, the AS sends back the following two messages to the client:
   * Message A: Client/TGS session key encrypted using the secret key of the user.
   * Message B: Ticket-Granting Ticket (which includes the client ID, client network address, ticket validity period, and the client/TGS session key) encrypted using the secret key of the TGS.
5. Once the client receives messages A and B, it decrypts message A to obtain the client/TGS session key. This session key is used for further communications with TGS. (Note: The client cannot decrypt the Message B, as it is encrypted using TGS's secret key.) At this point, the client has enough information to authenticate itself to the TGS.
6. When requesting services, the client sends the following two messages to the TGS:
   * Message C: Composed of the Ticket-Granting Ticket from message B and the ID of the requested service.
   * Message D: Authenticator (which is composed of the client ID and the timestamp), encrypted using the client/TGS session key.

Detailed Interactions (7-11)

7. Upon receiving messages C and D, the TGS decrypts message D (Authenticator) using the client/TGS session key and sends the following two messages to the client:
   * Message E: Client-to-server ticket (which includes the client ID, client network address, validity period) encrypted using the service's secret key.
   * Message F: Client/server session key encrypted with the client/TGS session key.
8. Upon receiving messages E and F from TGS, the client has enough information to authenticate itself to the SS. The client connects to the SS and sends the following two messages:
   * Message G: the client-to-server ticket, encrypted using service's secret key.
   * Message H: a new Authenticator, which includes the client ID, timestamp and is encrypted using client/server session key.
9. The server decrypts the ticket using its own secret key and sends the following message to the client to confirm its true identity and willingness to serve the client:
   * Message I: the timestamp found in client's recent Authenticator plus 1, encrypted using the client/server session key.
10. The client decrypts the confirmation using its shared key with the server and checks whether the timestamp is correctly updated. If so, then the client can trust the server and can start issuing service requests to the server.
11. The server provides the requested services to the client.

Kerberos Version 5

- developed in mid 1990's
- provides improvements over v4
  - addresses environmental shortcomings
    - encryption algorithm (DES vs AES), network protocol, byte order, ticket lifetime, inter-realm authorization
  - and technical deficiencies
    - double encryption, non-standard mode of use, session keys, password attacks
- specified as Internet standard RFC 1510

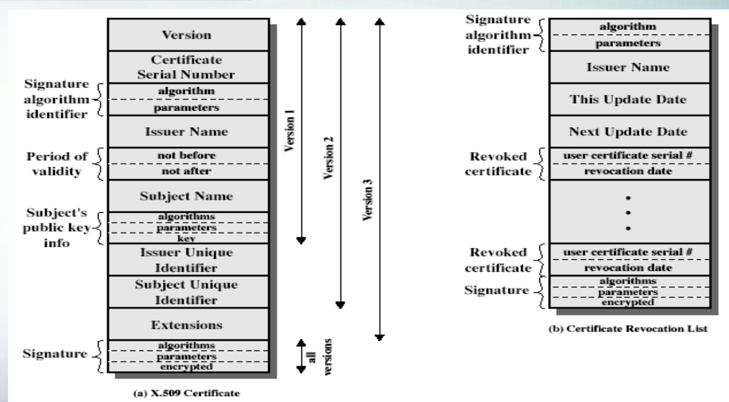
X.509 Authentication Service

- part of CCITT X.500 directory service standards
  - distributed servers maintaining some info database
- defines framework for authentication services
  - directory may store public-key certificates
  - with public key of user
  - signed by certification authority
- also defines authentication protocols
- uses public-key crypto & digital signatures
  - algorithms not standardised, but RSA recommended

X.509 Certificates

- issued by a Certification Authority (CA), containing:
  - version (1, 2, or 3)
  - serial number (unique within CA) identifying certificate
  - signature algorithm identifier
  - issuer X.500 name (CA)
  - period of validity (from - to dates)
  - subject X.500 name (name of owner)
  - subject public-key info (algorithm, parameters, key)
  - issuer unique identifier (v2+)
  - subject unique identifier (v2+)
  - extension fields (v3)
  - signature (of hash of all fields in certificate)
- CA<<A>> denotes certificate for A signed by CA

X.509 Certificates



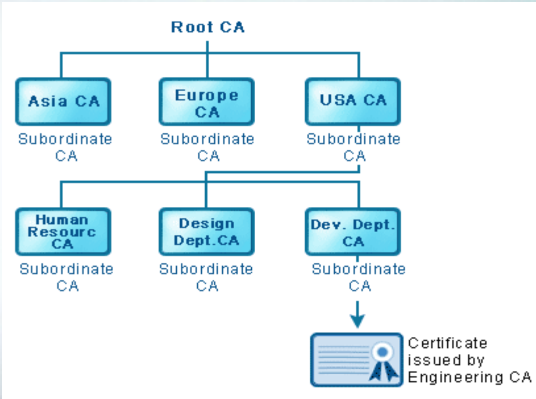
Obtaining a Certificate

- any user with access to CA can get any certificate from it
- only the CA can modify a certificate
- because cannot be forged, certificates can be placed in a public directory

CA Hierarchy

- If both users share a common CA then they are assumed to know its public key otherwise CA's must form a hierarchy.
- Use certificates linking members of hierarchy to validate other CA's
  - each CA has certificates for clients (forward) and parent (backward)
- Each client trusts parents certificates thus enable verification of any certificate from one CA by users of all other CA's in hierarchy

CA Hierarchy



Certificate Revocation

- **Certificates have a period of validity but may need to revoke before expiry, e.g.**
  1. user's private key is compromised
  2. user is no longer certified by this CA
  3. CA's certificate is compromised
- **CA's maintain list of revoked certificates**
  - Certificate Revocation List (CRL)
- **Users should check certificates with CA's CRL**

X.509 Version 3

- **has been recognised that additional information is needed in a certificate**
  - email/URL, policy details, usage constraints
- **rather than explicitly naming new fields defined a general extension method**
- **extensions consist of:**
  - extension identifier
  - criticality indicator
  - extension value

Certificate Extensions

- **key and policy information**
  - convey info about subject & issuer keys, plus indicators of certificate policy
- **certificate subject and issuer attributes**
  - support alternative names, in alternative formats for certificate subject and/or issuer
- **certificate path constraints**
  - allow constraints on use of certificates by other CA's

Network Security

SECU2102

Key Exchange and Digital Signatures

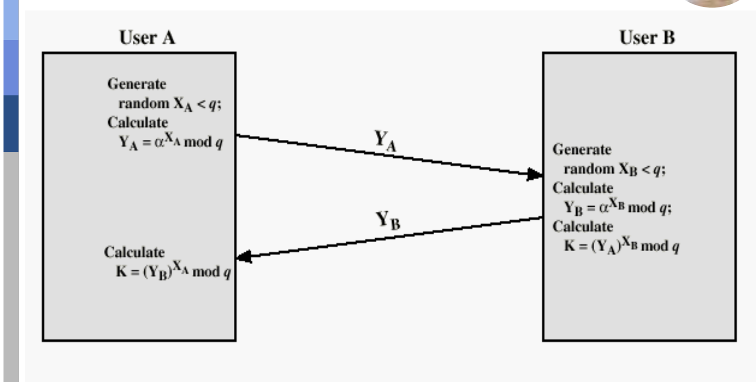
Diffie-Hellman Key Exchange

- First published public-key algorithm (1976)
- Purpose is to allow two users to exchange a private key
- Diffie-Hellman depends on the difficulty in computing discrete logarithms (inverse exponentials)
- Choose a prime p, consider the sequence
  - a mod p, a^2 mod p, a^3 mod p, … a^{p-1} mod p
- If these are distinct and a permutation of 1 … p-1, then
  1. b = a^i mod p then 'i' is the discrete logarithm of b
  2. a is called a primitive root of p

Diffie-Hellman Algorithm

- Global public elements
  - q – prime and 'a' is a primitive root of q
- User A key generation
  - Select private X_A, calculate public $Y_A = a^{X_A}$ mod q
- User B key generation
  - Select private X_B, calculate public $Y_B = a^{X_B}$ mod q
- Generation of Secret Key by User A
  - $K = (Y_B)^{X_A}$ mod q
- Generation of Secret Key by User B
  - $K = (Y_A)^{X_B}$ mod q

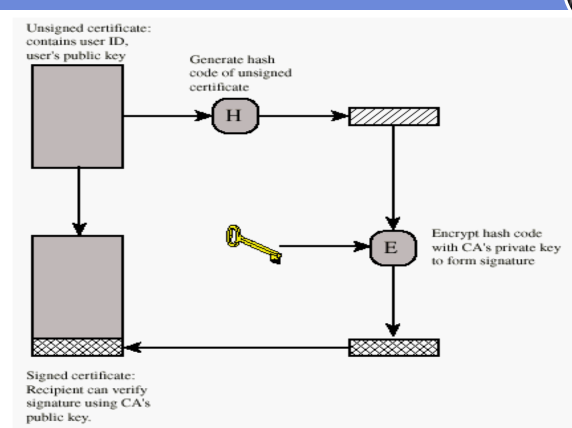
Diffie-Hellman Key Exchange



Key Management

- Major contribution of public-key encryption is to address the problem of key distribution
  - Private keys as in RSA to distribute key for AES
  - Public keys
- Public Key Certificates
  - Public keys are public. Why not just broadcast?
  - Forgery of public announcement
    - Darth sends out "I'm Bob and my public key is XXX"
    - Then Darth can read secret messages for Bob and Bob can't
- Public Key certificate: public key + User Id signed by trusted third party
- X.509 protocol for certificates (see before)

Public-Key Certificate



Public-Key Distribution of Secret Keys

- How to share private key between Bob and Alice?
- Diffie-Hellman
  - Works but no user authentication
- Alternative

1. Prepare message.
2. Encrypt message using conventional encryption using one-time session key.
3. Encrypt session key using Alice's public key.
4. Attach the encrypted session key to the message and send to Alice.
5. Only Alice is capable of decrypting the session key.
6. Bob get's public key from Alice's public-key certificate.

Digital Signatures

- have looked at message authentication
  - but does not address issues of lack of trust
- digital signatures provide the ability to:
  - verify author, date & time of signature
  - authenticate message contents
  - be verified by third parties to resolve disputes
- hence include authentication function with additional capabilities

Digital Signature Properties

- must depend on the message signed
- must use information unique to sender
  - to prevent both forgery and denial
- must be relatively easy to produce
- must be relatively easy to recognize & verify
- be computationally infeasible to forge
  - with new message for existing digital signature
  - with fraudulent digital signature for given message
- be practical to save digital signature in storage

Direct Digital Signatures

- involve only sender & receiver
- assumed receiver has sender's public-key
- digital signature made by sender signing entire message or hash with private-key
- can encrypt using receiver's public-key
- important that sign first then encrypt message & signature
- security depends on sender's private-key

Replay Attacks

- where a valid signed message is copied and later resent
  - simple replay
  - repetition that can be logged
  - repetition that cannot be detected
  - backward replay without modification
- countermeasures include
  - use of sequence numbers (generally impractical)
  - timestamps (needs synchronized clocks)
  - challenge/response (using unique nonce)

Digital Signature Standard (DSS)

- US Govt approved signature scheme FIPS 186
- uses the SHA hash algorithm
- designed by NIST & NSA in early 90's
- DSS is the standard, DSA is the algorithm
- a variant on ElGamal and Schnorr schemes
- creates a 320 bit signature, but with 512-1024 bit security
- security depends on difficulty of computing discrete logarithms

DSA Key Generation

- have shared global public key values (p, q, g):
  - a large prime **p**, with $2^{L-1} < p < 2^L$
    - where L= 512 to 1024 bits and is a multiple of 64
  - choose **q**, a 160-bit prime factor of p-1
  - choose **g** $= h^{(p-1)/q}$
    - where **h** is any integer 1<h<p-1, such that $h^{(p-1)/q} \pmod p > 1$
- users choose private & compute public key:
  - choose a random number **x** with x<q
  - Compute public key **y** $= g^x \pmod p$

DSA Signature Creation

- to **sign** a message M the sender:
  - generates a random signature key k, k<q
  - N.b. k must be random, be destroyed after use, and never be reused
- then computes signature pair:

  $r = (g^k \pmod p) \pmod q$

  $s = (k^{-1}.(SHA(M)+ x.r)) \pmod q$
- sends signature (r,s) with message M

DSA Signature Verification

- having received M & signature (r,s)
- to **verify** a signature, recipient computes:

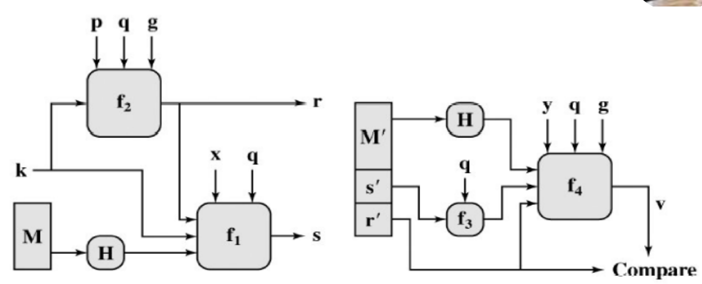
  $w = s^{-1} \pmod q$

  $u1= (SHA(M).w) \pmod q$

  $u2= (r.w) \pmod q$

  $v = (g^{u1}.y^{u2} \pmod p) \pmod q$
- if v=r then signature is verified

DSS Signing And Verifying



$s = f_1(H(M), k, x, r, q) = (k^{-1} (H(M) + xr)) \bmod q$

$r = f_2(k, p, q, g) = (g^k \bmod p) \bmod q$

(a) Signing

$w = f_3(s', q) = (s')^{-1} \bmod q$

$v = f_4(y, q, g, H(M'), w, r')$

$= ((g^{(H(M')w) \bmod q}\, y^{r'w \bmod q}) \bmod p) \bmod q$

(b) Verifying

Network Security

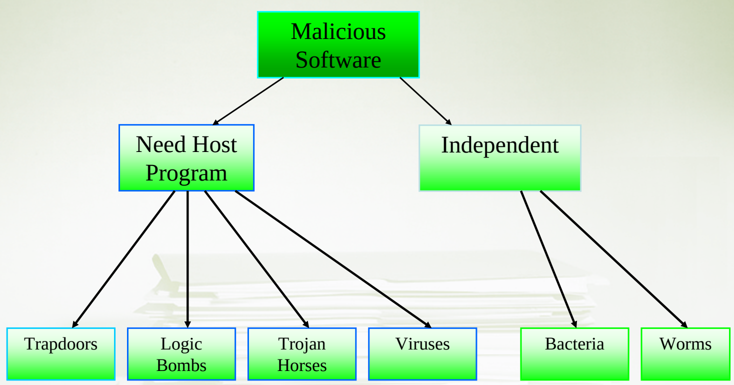
SECU2102

Malware

Viruses and "Malicious Software"

- **Computer Viruses** and related programs have the ability to replicate themselves on an ever increasing number of computers. They originally spread by people sharing floppy disks. Now they spread primarily over the Internet as *worms*.
- Other **Malicious software** may be installed by hand on a single machine. They may also be built into widely distributed commercial software packages. These are very hard to detect before the *payload* activates (Trojan Horses, Trap Doors, and Logic Bombs).

Taxanomy of Malicious Software



Definitions

- **Virus -** code that copies itself into other programs.
- **Bacteria** replicate until it fills all disk space, or CPU cycles.
- **Payload -** harmful things the malicious program does, after it has had time to spread.
- **Worm -** a program that replicates itself across the network.

Definitions

- **Trojan Horse** - instructions in an otherwise good program that cause bad things to happen (sending your data or password to an attacker over the Net).
- **Logic Bomb** - malicious code that activates on a specific event (e.g., specific Date).
- **Trap Door** (or Back Door) - undocumented entry point written into code for debugging that can allow unwanted users.
- **"Easter Egg"** - extraneous code that does something "cool." A way for programmers to show that they control the product.

Virus Phases

- **Dormant phase** - the virus is idle
- **Propagation phase** - the virus places an identical copy of itself into other programs
- **Triggering phase –** the virus is activated to perform the function for which it was intended
- **Execution phase –** the function is performed

Types of Viruses

- **Parasitic Virus** - attaches itself to executable files as part of their code. Runs whenever the host program runs.
- **Memory-resident Virus** - Lodges in main memory as part of the residual operating system.
- **Boot Sector Virus** - infects the boot sector of a disk, and spreads when the operating system boots up (original DOS viruses).
- **Stealth Virus** - explicitly designed to hide from Virus Scanning programs.
- **Polymorphic Virus** - mutates with every new host to prevent signature detection.

Macro Viruses

- Microsoft Office applications allow “macros” to be part of the document. The macro could run whenever the document is opened, or when a certain command is selected (Save File).
- Platform independent.
- Infect documents, delete files, generate email and edit letters.

Antivirus Approaches

- 1st Generation, Scanners: searched files for any of a library of known virus “signatures.” Checked executable files for length changes.
- 2nd Generation, Heuristic Scanners: looks for more general signs than specific signatures (code segments common to many viruses). Checked files for checksum or hash changes.
- 3rd Generation, Activity Traps: stay resident in memory and look for certain patterns of software behavior (e.g., scanning files).
- 4th Generation, Full Featured: combine the best of the techniques above.

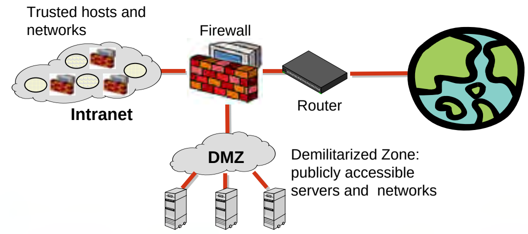
Network Security

SECU2102

Firewall & Network Defences

Firewalls

- Idea: separate local network from the Internet



Castle and Moat Analogy

- More like the moat around a castle than a firewall
  - Restricts access from the outside
  - Restricts outbound connections, too

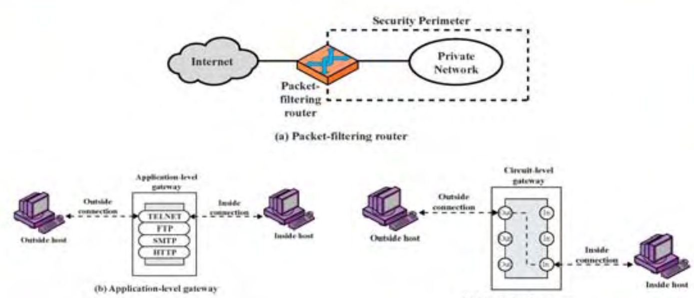
Firewall Locations in the Network

- Between internal LAN and external network
- At the gateways of sensitive subnets within the organizational LAN
  - Payroll's network must be protected separately within the corporate network
- On end-user machines
  - "Personal firewall"
  - Microsoft's Internet Connection Firewall (ICF) comes standard with Windows XP

Firewall Types

- Packet-filtering or session-filtering
- Proxy gateway (Server)
  - All incoming traffic is directed to firewall, all outgoing traffic appears to come from firewall (NAT)
  - Application-level: separate proxy for each application
    - Different proxies for SMTP (email), HTTP, FTP, etc.
    - Filtering rules are application-specific
  - Circuit-level: application-independent, "transparent"
    - Only generic IP traffic filtering (example: SOCKS)
- Personal firewall with application-specific rules

Firewall Types: Illustration

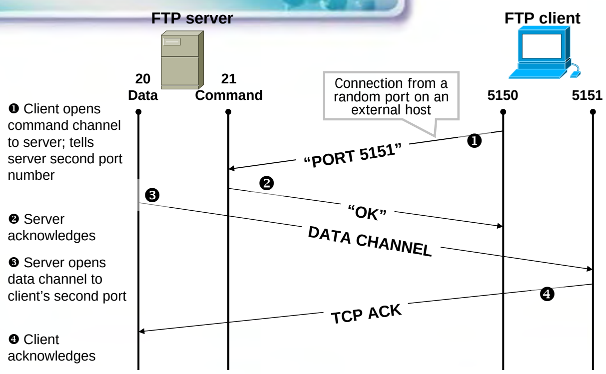


Packet Filtering

- For each packet, firewall decides whether to allow it to proceed
  - Decision must be made on per-packet basis
    - Stateless; cannot examine packet's context (TCP connection, application to which it belongs, etc...)
- To decide, use information available in the packet
  - IP source and destination addresses, ports
  - Protocol identifier (TCP, UDP, ICMP, etc.)
  - TCP flags (SYN, ACK, RST, FIN)
  - ICMP message type
- Filtering rules are based on pattern-matching

Packet Filtering Examples

Example: FTP



FTP Packet Filter

The following filtering rules allow a user to FTP from any IP address to the FTP server at 172.168.10.12

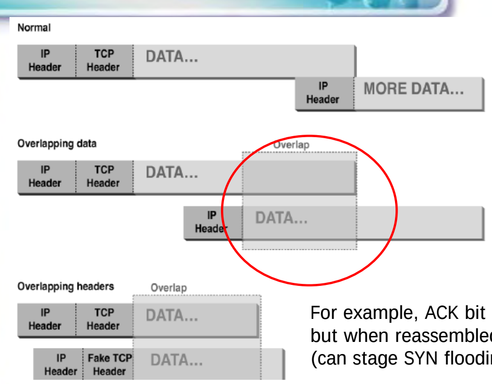
```
access-list 100 permit tcp any gt 1023 host 172.168.10.12 eq 21
access-list 100 permit tcp any gt 1023 host 172.168.10.12 eq 20
 ! Allows packets from any client to the FTP control and data ports
access-list 101 permit tcp host 172.168.10.12 eq 21 any gt 1023
access-list 101 permit tcp host 172.168.10.12 eq 20 any gt 1023
 ! Allows the FTP server to send packets back to any IP address with TCP ports > 1023

interface Ethernet 0
 access-list 100 in    ! Apply the first rule to inbound traffic
 access-list 101 out   ! Apply the second rule to outbound traffic
!
```

Anything not explicitly permitted by the access list is denied!

Weaknesses of Packet Filters

- Do not prevent application-specific attacks
  - For example, if there is a buffer overflow in URL decoding routine, firewall will not block an attack string. More later…
- No user authentication mechanisms
  - … except (spoofable) address-based authentication
  - Firewalls don't have any upper-level functionality
- Vulnerable to TCP/IP attacks such as spoofing
  - Solution: list of addresses for each interface (packets with internal addresses shouldn't come from outside)
- Security breaches due to misconfiguration

Abnormal Fragmentation

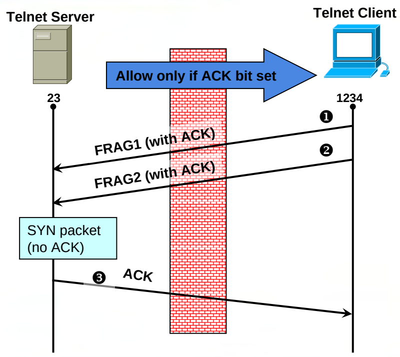


For example, ACK bit is set in both fragments, but when reassembled, SYN bit is set (can stage SYN flooding through firewall)

Fragmentation Attack

❶,❷ Send 2 fragments with the ACK bit set; fragment offsets are chosen so that the full datagram re-assembled by server forms a packet with the SYN bit set (the fragment offset of the second packet overlaps into the space of the first packet)

❸ All following packets will have the ACK bit set



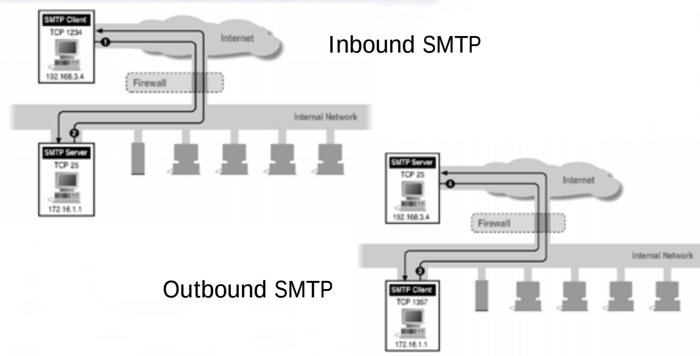
More Fragmentation Attacks

- Split ICMP message into two fragments, the assembled message is too large
  - Buffer overflow, OS crash
- Fragment a URL or FTP “put” command
  - Firewall needs to understand application-specific commands to catch this
- chargen attacks
  - “Character generation” debugging tool: connect to a certain port and receive a stream of data
  - If attacker fools it into connecting to itself, CPU locks

Stateless Filtering Is Not Enough

- In TCP connections, ports with numbers less than 1024 are permanently assigned to servers
  - 20,21 for FTP, 23 for telnet, 25 for SMTP, 80 for HTTP…
- Clients use ports numbered from 1024 to 16383
  - They must be available for clients to receive responses
- What should a firewall do if it sees, say, an incoming request to some client’s port 5612?
  - It must allow it: this could be a server’s response in a previously established connection…
  - …OR it could be malicious traffic
  - Can’t tell without keeping state for each connection

Example: Variable Port Use



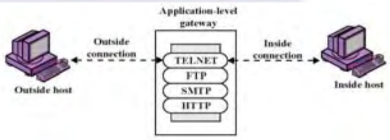
Session Filtering

- Decision is still made separately for each packet, but in the context of a connection
  - If new connection, then check against security policy
  - If existing connection, then look it up in the table and update the table, if necessary
    - Only allow incoming traffic to a high-numbered port if there is an established connection to that port
- Hard to filter stateless protocols (UDP) and ICMP
- Typical filter: deny everything that’s not allowed
  - Must be careful filtering out service traffic such as ICMP
- Filters can be bypassed with IP tunneling

Example: Connection State Table

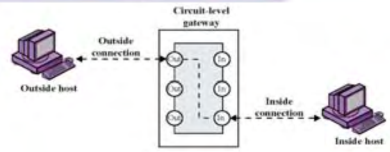
Source Address	Source Port	Destination Address	Destination Port	Connection State
192.168.1.100	1030	210.9.88.29	80	Established
192.168.1.102	1031	216.32.42.123	80	Established
192.168.1.101	1033	173.66.32.122	25	Established
192.168.1.106	1035	177.231.32.12	79	Established
223.43.21.231	1990	192.168.1.6	80	Established
219.22.123.32	2112	192.168.1.6	80	Established
210.99.212.18	3321	192.168.1.6	80	Established
24.102.32.23	1025	192.168.1.6	80	Established
223.212.212	1046	192.168.1.6	80	Established

Application-Level Gateway



- Splices and relays two application-specific connections
  - Example: Web browser proxy
  - Daemon spawns proxy process when communication is detected
  - Big processing overhead, but can log and audit all activity
- Can support high-level user-to-gateway authentication
  - Log into the proxy server with your name and password
- Simpler filtering rules than for arbitrary TCP/IP traffic
- Each application requires implementing its own proxy

Circuit-Level Gateway



- Splices two TCP connections, relays TCP segments
- Less control over data than application-level gateway
  - Does not examine the contents of TCP segment
- Client's TCP stack must be aware of the gateway
  - Client applications are often adapted to support SOCKS
- Often used when internal users are trusted
  - Application-level proxy on inbound connections, circuit-level proxy on outbound connections (lower overhead)

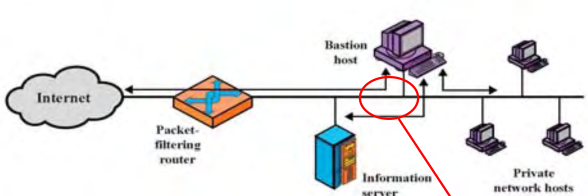
Comparison

	Performance	Modify client application	Defends against fragm. attacks
Packet filter	Best	No	No
Session filter		No	Maybe
Circuit-level gateway		Yes	Yes (SOCKS)
Application-level gateway	Worst	Yes	Yes

Bastion Host

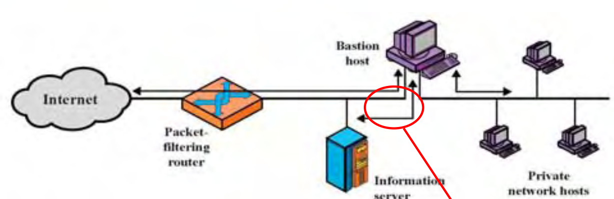
- Bastion host is a hardened system implementing application-level gateway behind packet filter
  - All non-essential services are turned off
  - Application-specific proxies for supported services
    - Each proxy supports only a subset of application's commands, is logged and audited, disk access restricted, runs as a non-privileged user in a separate directory (independent of others)
  - Support for user authentication
- All traffic flows through bastion host
  - Packet router allows external packets to enter only if their destination is bastion host, and internal packets to leave only if their origin is bastion host

Single-Homed Bastion Host



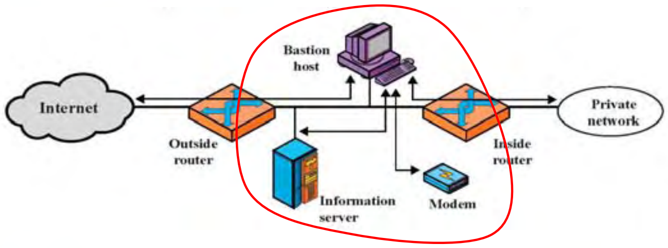
If packet filter is compromised, traffic can flow to internal network

Dual-Homed Bastion Host



No physical connection between internal and external networks

Screened Subnet



Only the screened subnet is visible
to the external network;
internal network is invisible

General Problems with Firewalls

- Interfere with networked applications
- Don't solve the real problems
  - Buggy software (think buffer overflow exploits)
  - Bad protocol design (think WEP in 802.11b)
- Generally don't prevent denial of service
- Don't prevent insider attacks
- Increasing complexity and potential for misconfiguration

Network Telescopes and Honeypots

- Monitor a cross-section of Internet address space
  - Especially useful if includes unused "dark space"
- Attacks in far corners of the Internet may produce traffic directed at your addresses
  - "Backscatter": responses of DoS victims to randomly spoofed IP addresses
  - Random scanning by worms
- Can combine with "honeypots"
  - Any outbound connection from a "honeypot" behind an otherwise unused IP address means infection
  - Can use this to extract worm signatures

Scanning Detection and Defence

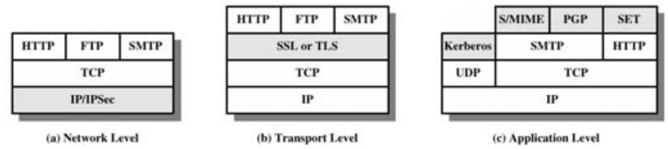
- Port scan is often a prelude to an attack
  - Someone is investigating which network services are available on your machine
    - Looking for an old version of some daemon with unpatched buffer overflow?
- Scan suppression: block traffic from addresses that previously produced too many failed connection attempts
  - Goal: detect port scans from attacker-controlled hosts
  - Requires network filtering and maintaining state
  - Can be subverted by slow scanning; does not work very well if the origin of scan is far away (why?)

Network Security

SECU2102

SSL & IPSec

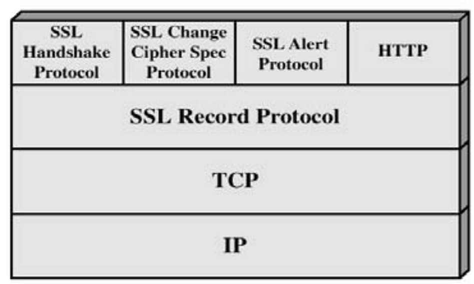
Security facilities in the TCP/IP protocol stack



SSL and TLS

- SSL was originated by Netscape
- TLS working group was formed within IETF
- First version of TLS can be viewed as an SSLv3.1
- TLS provides endpoint authentication and communications privacy over the Internet using cryptography.
- Most browsers nowadays have support for both SSL/TLS

SSL Architecture

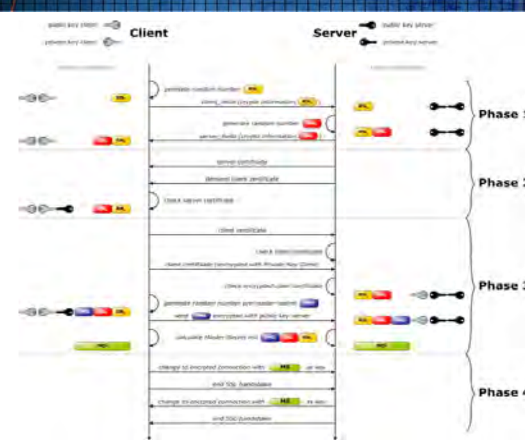


SSL Protocol Stack

SSL/TLS Handshake Protocol

- The most complex part of SSL.
- Allows the server and client to authenticate each other.
- Negotiate encryption, MAC algorithm and cryptographic keys.
- Used before any application data are transmitted.
- Message Fields
  - Message Type (8 bits)
  - Message Length (24 bits)
  - Content Type = 22 (handshake)

Handshake Protocol Action



Handshake Protocol Phase 1

- Establish security capabilities
- Client_hello →
  - Version = highest SSL understood by client
  - Random 32 bit time stamp + 28 random bytes (secure random number generator)
  - sessionID: 0→ establish new connection, non-zero means update parameters of an existing session
  - Ciphersuite: sequence of cryptographic algorithms in decreasing order of preference (key exchange + CipherSpec)
  - Compression methods: sequence of compression methods
- Server_hello ← is sent back
  - same as above but confirmation
  - Highest common version, new random field, same sessionID if nonzero, new sessionID otherwise, the selected ciphersuite and the selected compression technique

Handshake Protocol Phase 1

- Key Exchange methods
  1. RSA – secrt key is encrypted with receiver's RSA public key
  2. Fixed Diffie-Hellman
  3. Ephemeral Diffie Hellman
  4. Anonymous Diffie-Hellman
  5. Fortezza

  CipherSpec follows containing the fields
  1. Cipher algorithm
  2. MAC algorithm
  3. CipherType: block or stream
  4. Hash size: 0, 16 for MD5 or 20 for SHA-1 bytes
  5. Key material – sequence of bytes used to generate keys
  6. IV size of Initial Value for Cipher Block Chaining (CBC)

Handshake Protocol Phase 2

- Client Authentication and Key Exchange

- Server sends
  1. Certificate: X.509 certificate chain (not required for anonymous Diffie-Hellman)
  2. Server_key_exchange (not always need e.g. fixed Diffie-Hellman)
     - Hash(Client_hello.random||ServerHello.random||Server Parms)
  3. Certificate_request: certificate type and certificate authorities
  4. Server_hello_done: I'm done and I'll wait on response

Handshake Protocol Phase 3

- Client Authentication and Key Exchange
- Client verifies server certificate a checks the server hello parameters
- Client sends
  1. Certificate: if requested
  2. Client_key_exchange message must be sent
  3. Certificate_verify message to provide explicit verification of client certificate

Handshake Protocol Phase 4

- Finish up: switch to next cipher_spec
- Client sends
  1. Change_cipher_spec message
  2. Finished message under new algorithms, keys (new cipher_spec)

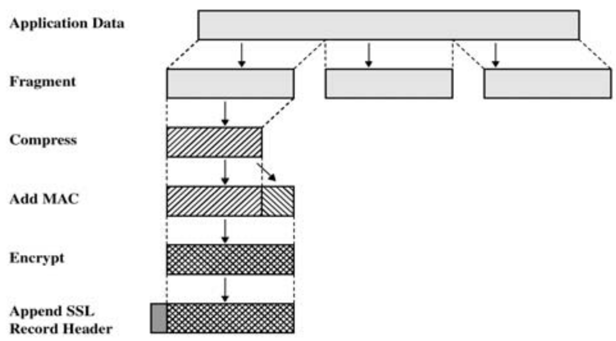
1. Server sends back
   1. Change_cipher_spec message
   2. Finished message under new algorithms, keys (new cipher_spec)

SSL Record Protocol Services

- Confidentiality – the handshake protocol defines a shared key for encryptions of SSL payloads

- Message Integrity – the handshake protocol defines a shared key used to form message authentication code (MAC)

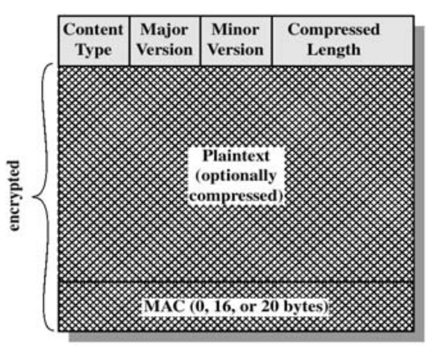
SSL Record Protocol Operation



Encryption Methods for SSL

- Fragments 2^{14} = 16384 bytes
- Compression must be lossless and must not increase length more than 1024
- No compression algorithm specified in SSLv3 – default no compression
- Block Cipher Encryption Methods
  - IDEA (128) RC2-40, DES-40, DES (56), 3DES (168), Fortezza(80)
- Stream Cipher Encryption choices
  - RC4-40, RC4-128
- Fortezza used in smart cards

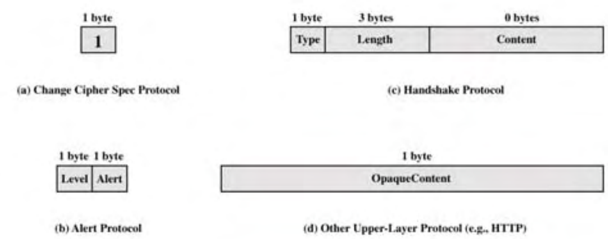
SSL Record Format



Content Types	
20	ChangeCipherSpec
21	Alert
22	Handshake
23	Application

Versions		
Major Version	Minor Version	Version Type
3	0	SSLv3
3	1	TLS 1.0
3	2	TLS 1.1
3	3	TLS 1.2

SSL Record Protocol Payload



Change Cipher Spec and Alert Protocols

- Change Cipher Spec Protocol

+	Bits 0–7	8.15	16.23	24–31
0	20	Version (MSB)	Version (LSB)	0
32	1	1 (CCS protocol type)		

- Alert Protocol – use to convey SSL-related alerts.

  Two Bytes:

  First byte: Level – i.e. either value 1 or 2

  1=warning: Connection or security may be unstable

  2=fatal: connection or security may be compromised, or an unrecoverable error has occurred.

  Second Byte: Description – which type of alerts

Some Alert Descriptions

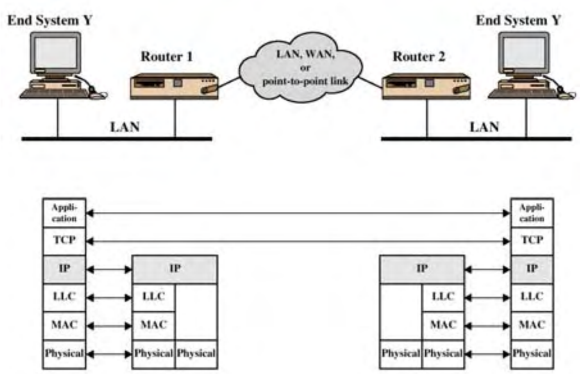
Description Types	
Code	Description
0	Close notify (warning or fatal)
10	Unexpected message (fatal)
20	Bad record MAC (fatal)
21	Decryption failed (fatal, TLS only)
22	Record overflow (fatal, TLS only)
30	Decompression failure (fatal)
40	Handshake failure (fatal)
41	No certificate (SSL v3 only) (warning or fatal)
42	Bad certificate (warning or fatal)
43	Unsupported certificate (warning or fatal)
44	Certificate revoked (warning or fatal)
45	Certificate expired (warning or fatal)
46	Certificate unknown (warning or fatal)
47	Illegal parameter (fatal)
48	Unknown CA (fatal, TLS only)
49	Access denied (fatal, TLS only)
50	Decode error (fatal, TLS only)
51	Decrypt error (TLS only) (warning or fatal)
60	Export restriction (fatal, TLS only)
70	Protocol version (fatal, TLS only)

IP Security (IPSec)

Outline

- Internetworking and Internet Protocols
- IP Security Overview
- IP Security Architecture
- Authentication Header
- Encapsulating Security Payload
- Combinations of Security Associations

TCP / IP Example



IP Security

- have considered some application specific security mechanisms
  - E.g. SSL/HTTPS
- however there are security concerns that cut across protocol layers
- would like security implemented by the network for all applications

IPSec

- general IP Security mechanisms
- provides
  - authentication
  - confidentiality
  - key management (not discussed)
- applicable to use over LANs, across public & private WANs, & for the Internet

- Internet Engineering Task Force (IETF) develops protocol standards for the internet

IP version 4 Header

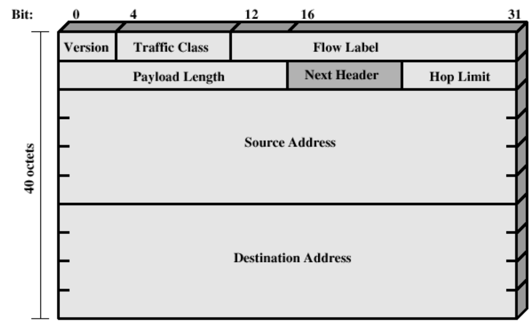
Bit: 0, 4, 8, 16, 19, 31 — 20 octets

Version	IHL	Type of Service	Total Length	
Identification			Flags	Fragment Offset
Time to Live		Protocol	Header Checksum	
Source Address				
Destination Address				
Options + padding				

IP version 4 Fields

- ❖ Version (4 bits) the value is 0100 = 4
- ❖ Internet Header Length (IHL)(4) length of header in 32bit words. The minimum value is 5.
- ❖ Type of Service(8)
- ❖ Total Length (16) Total IP packet length in octets
- ❖ Identification (16) sequence number
- ❖ Flags(3) "more bit", and "don't fragment bit"
- ❖ Fragment offset (13) where is belongs in 64bit units
- ❖ Time to Live (TTL) (8) number of "seconds" for packet to live
- ❖ Checksum
- ❖ Addresses 32-bit source and destination addresses
- ❖ Options

IP version 6 Header



IP version 6 Fields

- ❖ Version (4 bits) the value is 0110 (6)
- ❖ Traffic class (8) priority of this packet for routers
- ❖ Flow Label (20) label packets for special processing by routers
- ❖ Payload Length(16)
- ❖ Next Header(8) – usually TCP or UDP or an IPv6 extension
- ❖ Hop limit (8)
- ❖ Source Address(128=16 octets=4 words)
- ❖ Destination address (128=16octets=4 words)

IP Security Overview

IPSec is not a single protocol. Instead, IPSec provides a set of security algorithms plus a general framework that allows a pair of communicating entities to use whichever algorithms provide security appropriate for the communication.

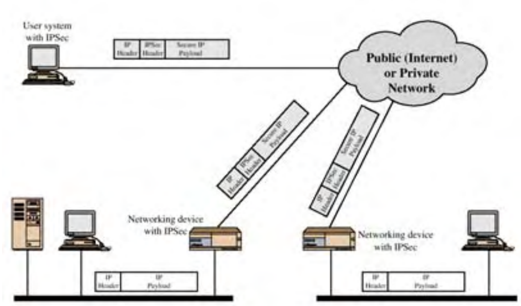
IP Security Overview

- ❖ Applications of IPSec
  - Secure branch office connectivity over the Internet
  - Secure remote access over the Internet
  - Establishing extranet and intranet connectivity with partners
  - Enhancing electronic commerce security
- ❖ Virtual Private Networks

  Check out http://www.howstuffworks.com/vpn.htm
- ❖ Two protocols

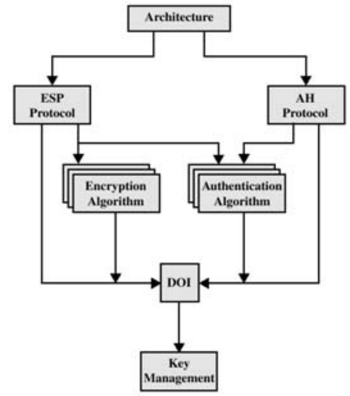
1. Authentication Header (AH) authentication protocol
2. Encapsulating Security Protocol (ESP) combined encryption/authentication protocol

IP Security Scenario



IP Security Architecture

- specification is quite complex
- defined in numerous RFC's
  - RFC 2401 – overview of security architecture
  - RFC 2402 – packet authentication extension
  - RFC 2406 – packet encryption
  - RFC 2408 – key management
  - many others, grouped by category
- mandatory in IPv6, optional in IPv4



Benefits of IPSec

- in a firewall/router provides strong security to all traffic crossing the perimeter
- is resistant to bypass
- is below transport layer, hence transparent to applications
- can be transparent to end users
- can provide security for individual users if desired

IPSec Services

- Access control
- Connectionless integrity
- Data-origin authentication
- Rejection of replayed packets
  - a form of partial sequence integrity
- Confidentiality (encryption)
- Limited traffic flow confidentiality

Security Associations

- a one-way relationship between sender & receiver that affords security for traffic flow
- For two-way it requires two separate SAs
- Uniquely defined by 3 parameters:
  - Security Parameters Index (SPI) this is carried in AH and ESP headers
  - IP Destination Address
  - Security Protocol Identifier
- has a number of other parameters
  - Sequence number, AH & EH info, lifetime etc
- have a database of Security Associations

SA Parameters

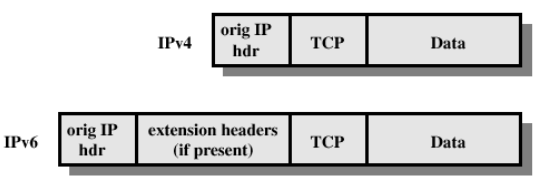
- Sequence number counter
- Sequence counter overflow flag
- Anti-replay window
- AH info: authentication algorithm, keys, key lifetimes
- ESP info: encryption and authentication algorithm, keys, key lifetimes
- Lifetime of this Security Association (SA)
- IPSec protocol mode: tunnel or transport
- Path MTU maximum transmission unit

Authentication Header (AH)

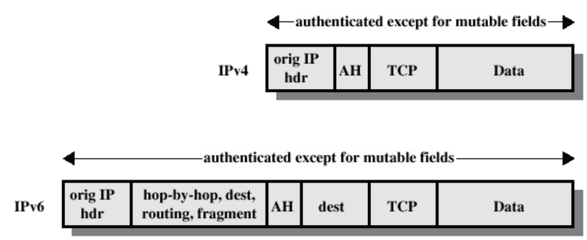
- **provides support for data integrity & authentication of IP packets**
  - end system/router can authenticate user/app
  - prevents address spoofing attacks by tracking sequence numbers
- **based on use of a MAC**
  - HMAC-MD5-96 or HMAC-SHA-1-96
- **parties must share a secret key**

	Transport Mode SA	Tunnel Mode SA
AH	Authenticates IP payload and selected portions of IP header and IPv6 extension headers	Authenticates entire inner IP packet plus selected portions of outer IP header
ESP	Encrypts IP payload and any IPv6 extesion header	Encrypts inner IP packet
ESP with authentication	Encrypts IP payload and any IPv6 extesion header. Authenticates IP payload but no IP header	Encrypts inner IP packet. Authenticates inner IP packet.

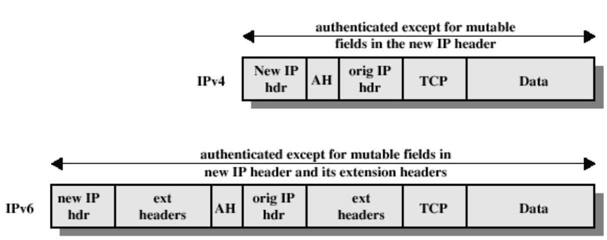
Before applying AH



Transport Mode (AH Authentication)

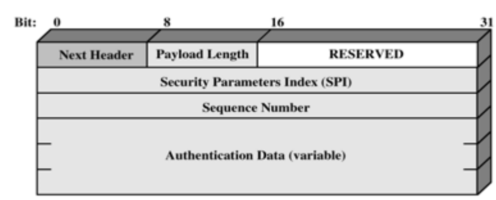


Tunnel Mode (AH Authentication)



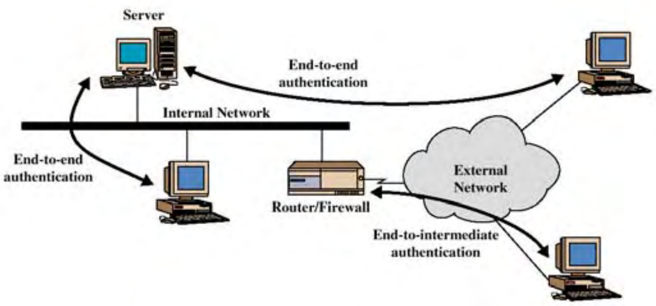
Authentication Header

- **Provides support for data integrity and authentication (MAC code) of IP packets.**
- **Guards against replay attacks.**



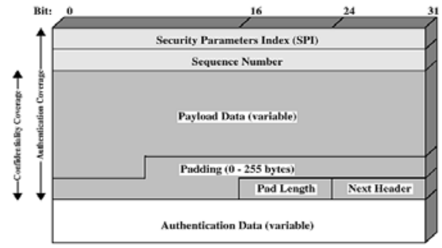
IPSec Authentication Header

End-to-end versus End-to-Intermediate Authentication



Encapsulating Security Payload

- ESP provides confidentiality services

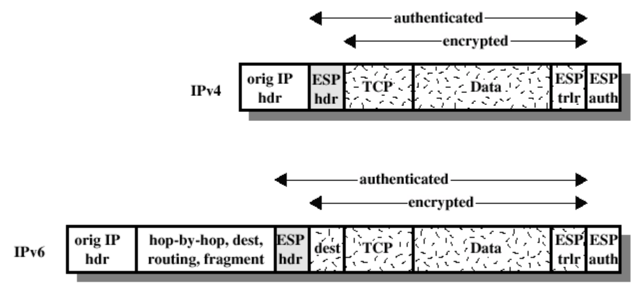


IPSec ESP Format

Encryption and Authentication Algorithms

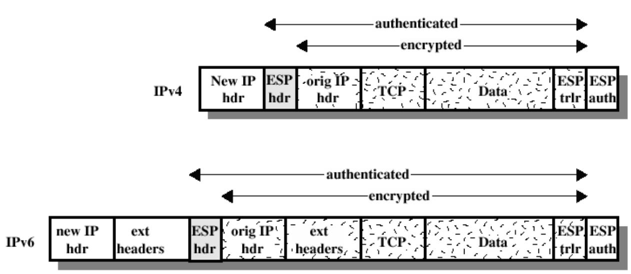
- **Encryption:**
  - Three-key triple DES
  - RC5
  - IDEA
  - Three-key triple IDEA
  - CAST
  - Blowfish
- **Authentication:**
  - HMAC-MD5-96
  - HMAC-SHA-1-96

ESP Encryption and Authentication



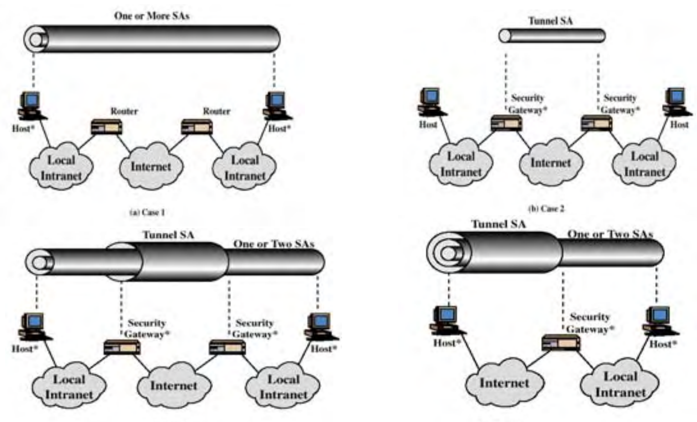
(a) Transport Mode

ESP Encryption and Authentication



(b) Tunnel Mode

Combinations of Security Associations



Cryptographic Assignment

AIM

You will be required to do create an application to encrypt and decrypt a line of plaintext using the following algorithm:

ENCRYPTION ALGORITHM

1. Remove any leading or trailing white space from the line.
2. Convert all letters in the string to UPPERCASE.

```
 Step 2 sample Input Line:   "abcdefghijklmnopqrstuvwxyz"
Step 2 sample Output Line:   "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
```

3. Perform the following character substitutions:

Original	Substitution	Original	Substitution
A	@	R	&
E	=	S	$
I	!	T	+
J	?	V	^
O	*	X	%
P	#	(space)	_

4. Move the first half of the string to be the last half. (Note: for lines of odd length the line must be divided such that the first half being moved contains one more character than the last half.)

```
 Step 4 sample Input Line:  "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
Step 4 sample Output Line:  "NOPQRSTUVWXYZABCDEFGHIJKLM"

 Step 4 sample Input Line:  "ABCDEFGHIJKLMNOPQRSTUVWXY"
Step 4 sample Output Line:  "NOPQRSTUVWXYABCDEFGHIJKLM"
```

5. Swap the first 2 characters of the line with the last two characters.

```
 Step 5 sample Input Line:  "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
Step 5 sample Output Line:  "YZCDEFGHIJKLMNOPQRSTUVWXAB"
```

6. Swap the two characters immediately to the left of the middle of the string with the two characters that immediately follow them.

```
 Step 6 sample Input Line:  "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
Step 6 sample Output Line:  "ABCDEFGHIJKNOLMPQRSTUVWXYZ"

 Step 6 sample Input Line:  "ABCDEFGHIJKLMNOPQRSTUVWXY"
Step 6 sample Output Line:  "ABCDEFGHIJKNOLMPQRSTUVWXY"
```

Below gives an example of applying all six steps sequentially to the alphabet string:

```
         Sample plaintext:  "abcdefghijklmnopqrstuvwxyz"
  Corresponding ciphertext:  "LM#Q&$+U^W%@BYZCD=FGH!?KN*"
```

THE FINAL OUTPUT

The final output i.e. the ciphertext should be written to a text file or displayed in a text box.

DECRYPTION ALGORITHM

Simply perform the reverse process.

SPECIFICATIONS

1. Your application can be GUI or non-GUI based.
2. The choice of programming language is open (C/C++, Java, VB 6.0, VB.Net, Assembly Language but no JavaScript)
3. Your application should be able be able to read a plaintext file e.g. plain.txt and output the ciphertext version e.g. crypt.txt. In decryption mode, you should be able to perform the reverse.
4. Your application should also allow the user to enter the plaintext in a text box and print the ciphertext in another text box and vice versa.
5. You should ensure that the user enters a plaintext with at least 4 characters and that at least two characters are substituted (see Encryption Step 3).
6. Additional functionalities will yield bonus marks.

DELIVERABLES

Please upload a copy of your zipped executable file + any .dll or .h files as appropriate with filename **your_index_no.zip** in the ftp://intraweb/assignments/rh/SECU2102/BCA07BPT/ass1 and ftp://intraweb/assignments/rh/SECU2102/BCNS06FT/ass1 folder.

Deadline: 20th September 2008

This assignment will count as 10% of the entire module.

ASSIGNMENT 2

Objectives

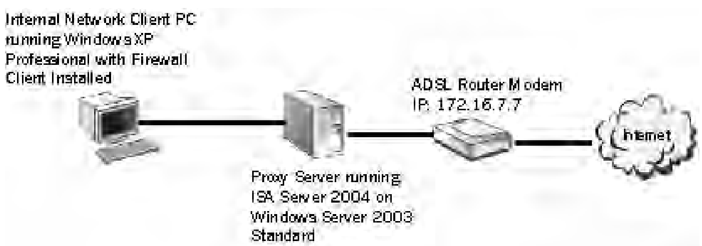
At the end of this assignment, you will be able to:

- Implement secure access to Internet resources
- Implement secure Internet client access to an organization's internal servers
- Monitor Microsoft® Internet Security and Acceleration (ISA) Server 2004

Scenario

You are all network administrators for UTM. You are deploying ISA Server 2004 in order to enhance the security of the network perimeter. You will be using ISA Server 2004 to provide internal users with secure access to Internet resources. You will also be using ISA Server to publish internal servers so that they can be accessed from the Internet. Finally, you will be monitoring ISA Server 2004.

A portion of the UTM network infrastructure is illustrated below:



Computers

Each group will have two computers:

one computer acting as Proxy Server
one client PC representing the internal network.

The computers are:
ALPHA (proxy server) and **ALPHACLIENT**
BETA (proxy server) and **BETACLIENT**
GAMMA (proxy Server) and **GAMMACLIENT**
PI (proxy server) and **PICLIENT**
SIGMA (proxy server) and **SIGMACLIENT**
OMEGA (proxy server) and **OMEGACLIENT**

Lab Setup

You will need to configure the Network Interface Cards and install ISA Server 2004 on the machine acting as proxy server. First log on the computer running Microsoft Windows Server 2003 as follows:

Username: **Administrator**
Password: **secu2102**

Select *Start->Settings->Network Connections.* You should see two connections labeled as **Internal** and **External.** Right-Click on each connection and select *Properties,* then double-click on the *Internet Protocol (TCP/IP)* settings. Enter the following configurations depending on your proxy server:

ALPHA: Internal
IP Address: 172.16.29.10
Subnet Mask: 255.255.0.0
Gateway: (leave blank)
DNS Servers: (leave blank)

ALPHA: External
IP Address: 172.16.29.25
Subnet Mask: 255.255.0.0
Gateway: 172.16.7.7
Primary DNS Server: 202.123.2.6

BETA: Internal
IP Address: 172.16.29.30
Subnet Mask: 255.255.0.0
Gateway: (leave blank)
DNS Servers: (leave blank)

BETA: External
IP Address: 172.16.29.45
Subnet Mask: 255.255.0.0
Gateway: 172.16.7.7
Primary DNS Server: 202.123.2.6

GAMMA: Internal
IP Address: 172.16.29.50
Subnet Mask: 255.255.0.0
Gateway: (leave blank)
DNS Servers: (leave blank)

GAMMA: External
IP Address: 172.16.29.65
Subnet Mask: 255.255.0.0
Gateway: 172.16.7.7
Primary DNS Server: 202.123.2.6

PI: Internal
IP Address: 172.16.29.70
Subnet Mask: 255.255.0.0
Gateway: (leave blank)
DNS Servers: (leave blank)

PI: External
IP Address: 172.16.29.85
Subnet Mask: 255.255.0.0
Gateway: 172.16.7.7
Primary DNS Server: 202.123.2.6

SIGMA: Internal
IP Address: 172.16.29.90
Subnet Mask: 255.255.0.0
Gateway: (leave blank)
DNS Servers: (leave blank)

SIGMA: External
IP Address: 172.16.29.105
Subnet Mask: 255.255.0.0
Gateway: 172.16.7.7
Primary DNS Server: 202.123.2.6

OMEGA: Internal
IP Address: 172.16.29.110
Subnet Mask: 255.255.0.0
Gateway: (leave blank)
DNS Servers: (leave blank)

OMEGA: External
IP Address: 172.16.29.125
Subnet Mask: 255.255.0.0
Gateway: 172.16.7.7
Primary DNS Server: 202.123.2.6

Next, we will need to Install ISA Server 2004…

Insert the CD-ROM provided, which should auto-run, and click on *Install ISA server 2004* from the menu. Else download it from the Intr@web ftp site at ftp://172.16.2.2/ and copy the folder ISA2004.

Follow the instructions, insert the Product-Key found on the CD-ROM when prompted and choose *Custom Installation.* Then make sure you additionally select *ISA Client Setup Files -> Run on this computer*. This will install the Client Firewall Setup files in a shared folder on the proxy server. You will need to access this folder later when you will install the Firewall Client on the Client PC.

During the setup of ISA Server you will need to specify a range of IP addresses representing your internal network. Please add the following depending on your proxy server:

ALPHA:	172.16.29.10 to 172.16.29.20
BETA:	172.16.29.30 to 172.16.29.40
GAMMA:	172.16.29.50 to 172.16.29.60
PI:	172.16.29.70 to 172.16.29.80
SIGMA:	172.16.29.90 to 172.16.29.100
OMEGA:	172.16.29.110 to 172.16.29.120

For the sake of this lab, you will assume that you have 10 hosts on your internal network.

Note: The Internal IP address of your proxy server **must** be within the Internal IP address range.

After ISA Server 2004 is installed, you will be asked to reboot the PC. Please do so. In the meantime you can now configure the NIC on your CLIENT PC by inserting the following respective configurations:

ALPHACLIENT	IP Address: 172.16.29.15 Subnet Mask: 255.255.0.0 Gateway: 172.16.29.10 DNS Servers: (leave blank)
BETACLIENT	IP Address: 172.16.29.35 Subnet Mask: 255.255.0.0 Gateway: 172.16.29.30 DNS Servers: (leave blank)
GAMMACLIENT:	IP Address: 172.16.29.55 Subnet Mask: 255.255.0.0 Gateway: 172.16.29.50 DNS Servers: (leave blank)
PICLIENT:	IP Address: 172.16.29.75 Subnet Mask: 255.255.0.0 Gateway: 172.16.29.70 DNS Servers: (leave blank)
SIGMACLIENT:	IP Address: 172.16.29.15 Subnet Mask: 255.255.0.0 Gateway: 172.16.29.10 DNS Servers: (leave blank)
OMEGACLIENT:	IP Address: 172.16.29.15 Subnet Mask: 255.255.0.0 Gateway: 172.16.29.10 DNS Servers: (leave blank)

The next step is to install the Firewall Client. Choose *Start->Run* then enter **\\your_proxy_ server_ name** to connect to your proxy server. Supply the credentials given previously. You should see a shared folder called **mspclnt**.
Go inside this folder and run the **setup.exe** file. This will install the Firewall client and will ask you to reboot when you've done.

You can now start the first exercise:

Exercise 1
Implementing Internet Access with ISA Server 2004

In this exercise, you will configure ISA server to allow outbound Web Access for client computers on the internal network.

Scenario

UTM is deploying ISA Server 2004. One of the reasons for deploying ISA Server is to provide a proxy server that can be used by internal clients when accessing resources on the Internet.

In this exercise, you will configure ISA server to support outbound Internet Access

Tasks	Detailed Steps
Note: Perform the following step on your client machine (e.g. PICLIENT)	
1. Test your connectivity by opening Microsoft Internet Explorer and attempting to connect to my website at ***http://pages.intnet.mu/rhh***	**a.** Open Internet Explorer. In the **Address** text box, type **http://pages.intnet.mu/rhh,** and then press ENTER. *Internet Explorer is unable to connect to the Web site.* **b.** Look at the bottom of the Web page and view the reason why the Web page cannot be displayed. *ISA Server denies the request. (502 Proxy Error - ISA Server denied the specified URL.) This is because you have not created any access rules yet.* **c.** Close Internet Explorer.
Note: Perform the following steps on your proxy server (e.g. PI)	
2. Create a new access rule. Name: **Allow outbound Web traffic** Applies to: **Hypertext Transfer Protocol (HTTP), Hypertext Transfer Protocol Secure (HTTPS), File Transfer Protocol (FTP)** From network: **Internal** To network: **External**	**a.** On the Start menu, point to **All Programs,** point to **Microsoft ISA Server,** and then click **ISA Server Management**. *The ISA Server console opens.* **b.** In the ISA Server console, expand **PI,** and then select **Firewall Policy**. **c.** In the right pane, on the **Firewall Policy** tab, select **Last Default rule.** **d.** In the task pane, on the **Tasks** tab, click **Create New Access Rule.** *Instead of using the task pane, you can also right-click **Firewall Policy**, click **New**, and then click **Access Rule**.* **e.** In the New Access Rule Wizard dialog box, in the Access rule name text box, type **Allow outbound Web traffic,** and then click **Next.** **f.** On the **Rule Action** page, select **Allow**, and then click **Next.** **g.** On the **Protocols** page, in the **This rule applies to** list box, select **Selected protocols,** and then click **Add.** *The **Add Protocols** dialog box appears.* **h.** In the **Add Protocols** dialog box, click **Common Protocols**, click **HTTP**, and click **Add.** Click **HTTPS**, and click **Add**. Click **Web**, click **FTP**, and click **Add**. Finally, click **Close** to close the **Add Protocols** dialog box.

	Note: the same protocols can be listed under multiple headings in the ***Add Protocols*** *dialog box.* **i.** On the **Protocols** page, click **Next**. **j.** On the **Access Rule Sources** page, click **Add**. *The* ***Add Network Entities*** *dialog box appears.* **k.** In the **Add Network Entities** dialog box, click **Networks**, click **Internal**, and click **Add**. Click **Close** to close the **Add Network Entities** dialog box. **l.** On the **Access Rule Sources** page, click **Next**. **m.** On the **Access Rule Destinations** page, click **Add**. **n.** In the **Add Network Entities** dialog box, click **Networks**, click **External**, and click **Add**. Click **Close** to close the **Add Network Entities** dialog box. **o.** On the **Access Rule Destinations** page, click **Next**. **p.** On the **User Sets** page, click **Next**. **q.** On the **Completing the New Access Rule Wizard** page, click **Finish**. *A new firewall policy rule is created that allows the FTP, HTTP & HTTPS protocols to operate from the Internal to the External network for all users. The new rule has not been applied yet.* **r.** Click **Apply** to apply the new rule, and then click **OK**.
3. Examine the network rule for connectivity between the **Internal** network and the **External** network.	**a.** In the left pane, expand **Configuration**, and then select **Networks**. **b.** In the right pane, on the **Network Rules** tab, select the rule that defines the connectivity between the **Internal** network and the **External** network. *In the default configuration, the network rule named Internet Access (rule 3) indicates that network traffic between the Internal network and the External network will use NAT.*
4. Examine the Web Proxy settings of the **Internal** network.	**a.** On the **Networks** tab, right-click **Internal**, and then click **Properties**. **b.** In **the Internal Properties** dialog box, select the **Web Proxy** tab. *The Enable Web Proxy Clients check box shows that ISA Server listens (on port 8080) for requests from Web Proxy clients on the Internal network.* **c.** Click **CANCEL** to close the **Internal Properties** dialog box.
Note: Perform the following steps on your client (e.g. PICLIENT)	
5. Test your connectivity again by opening Internet Explorer and connecting to **http://pages.intnet.mu/rhh**	**a.** Open Internet Explorer. In the **Address** box, type **http://pages.intnet.mu/rhh**, and then press ENTER. *Internet Explorer displays Nefertum's Shrine. The access rule that you created in step 2 grants the PICLIENT access to network traffic.* **b.** In Internet Explorer, on the **Tools** menu, click

	Internet Options. **c.** In the Internet Options dialog box, on the Connections tab, click **LAN Settings**. *Notice that PICLIENT is indeed configured as the Web Proxy client.* **d.** Click **Cancel** to close the **Local Area Network (LAN) Settings** dialog box. **e.** Click **Cancel** to close **the Internet Options** dialog box. **f.** Close **Internet Explorer**.
Note: Perform the following steps on your proxy server (e.g. PI)	
6. Create a new Computer Set rule element. Name: **Restricted Internal Computers** Included in the set: ALPHA: 172.16.29.11-172.16.29.15 Include the respective ranges below for your proxy: BETA: 172.16.29.31-172.16.29.35 GAMMA: 172.16.29.51-172.16.29.55 PI: 172.16.29.71-172.16.29.75 SIGMA: 172.16.29.91-172.16.29.95 OMEGA: 172.16.29.111-172.16.29.115	**a.** In the **ISA Server** console, in the left pane, select **Firewall Policy**. **b.** In the task pane, on the **Toolbox** tab, in the **Network Objects** section, right-click **Computer Sets**, and then click **New Computer Set**. **c.** In the **New Computer Set Rule Element** dialog box, in the **Name** text box, type **Restricted Internal Computers**. **d.** Click **Add**, and then click **Address Range**. **e.** In the **New Address Range Rule Element** dialog box, complete the following information: • Name: **PI Restrictions** • Start Address: **172.16.29.71** • End Address: **172.16.29.75** • Description: **PI internal servers** **f.** Click **OK**. **g.** Click **OK** to close the **New Computer Set Rule Element** dialog box. *A new Computer Set rule element is created.*
7. Create a new access rule. Name: **Deny restricted computers** Action: **Deny** Applies to: **All outbound traffic** From: **Restricted Internal Computers** To network: **External**	**a.** In the Firewall Policy list, select the Allow outbound Web traffic rule. *The new rule will be added before the selected rule.* **b.** In the task pane, on the **Tasks** tab, click **Create New Access Rule**. **c.** In the **New Access Rule Wizard** dialog box, in the Access rule name text box, type **Deny restricted computers**, and then click **Next**. **d.** On the **Rule Action** page, select **Deny**, and then click **Next**. **e.** On the **Protocols** page, in the **This rule applies to** list box, select **All outbound traffic**, and then click **Next**. **f.** On the **Access Rule Sources** page, click **Add**. **g.** In the **Add Network Entities** dialog box, click **Computer Sets**, click **Restricted Internal Computers**, and click **Add**. Click **Close** to close the **Add Network Entities** dialog box. **h.** On the **Access Rule Sources** page, click **Next**. **i.** On the **Access Rule Destinations** page, click **Add**. **j.** In the **Add Network Entities** dialog box, click **Networks**, click **External**, and click **Add**. Click **Close** to

	close the **Add Network Entities** dialog box. **k.** On the **Access Rule Destinations** page, click **Next. l.** On the **User Sets** page, click **Next**. **m.** On the **Completing the New Access Rule Wizard** page, click **Finish.** *A new firewall policy rule is created that denies all network traffic going from the computers in the* ***Restricted Internal Computers*** *set to the External network.* *The new rule is listed first in the firewall policy rule list.* **n.** Click **Apply** to apply the new rule, then click **OK**.
Note: Perform the following steps on your client (e.g. PICLIENT)	
8. Test your connectivity again by opening Internet Explorer and attempting to connect to http://pages.intnet.mu/rhh	**a.** Open Internet Explorer. In the **Address** box, type **http://pages.intnet.mu/rhh**, and then press ENTER. *Internet Explorer is unable to connect to the Web site (502 Proxy Error). ISA Server denies access to the Nefertum's Shrine because PICLIENT (172.16.29.75) is in the* ***Restricted Internal Computers*** *set and is denied access by the new access rule.* **b.** Close Internet Explorer.
Note: Perform the following steps on your proxy server (e.g. PI)	
9. Move the **Allow outbound Web traffic** rule before the **Deny restricted computers** rule.	**a.** In the **ISA Server** console, in the left pane, select **Firewall Policy**. **b.** In the right pane, right-click the **Allow outbound Web traffic** rule (order 2), and then click **Move Up**. *The Allow outbound Web traffic rule (order 1) is now listed before the Deny restricted computers rule (order 2).* **c.** Click **Apply** to save the changes, and then click **OK**.
Note: Perform the following steps on your client (e.g. PICLIENT)	
10. Test your connectivity again by opening Internet Explorer and connecting to **http://pages.intnet.mu/rhh**	**a.** Open Internet Explorer. In the **Address** box, type **http://pages.intnet.mu/rhh**, and then press ENTER. *Internet Explorer displays Nefertum's Shrine, even though the* ***Firewall Policy*** *list contains a rule that denies access to the site from the PICLIENT (172.16.29.75) computer.* **b.** Close Internet Explorer.
Note: Perform the following steps on your proxy server (e.g. PI)	
11. Delete the **Deny restricted computers** access rule.	**a.** In the **ISA Server** console, in the left pane, select **Firewall Policy**. **b.** In the right pane, right-click the **Deny restricted computers** rule, and then click **Delete**. **c.** Click **Yes** to confirm that you want to delete the rule. *The access rule is deleted.* **d.** Click **Apply** to save the changes, and then click **OK**. **e.** Close all open windows.

Exercise 2
Monitoring ISA Server 2004

In this exercise, you will explore the monitoring functions of ISA Server 2004.

Scenario

As part of your role as the ISA Server administrator at UTM, you are responsible for ongoing monitoring of the ISA Server computers.

In this exercise, you will configure alerts as well as log client access to Internet resources through the ISA Server computer.

Tasks	Detailed Steps
Note: Perform the following step on your proxy server (e.g. PI)	
1. Examine the alert definition for the **Service shutdown** event.	**a.** On the **Start** menu, point to **All Programs**, point to **Microsoft ISA Server**, and then click **ISA Server Management**. **b.** In the **ISA Server** console, in the left pane, select **Monitoring**. **c.** In the right pane, select the **Dashboard** tab. *The* ***Monitoring*** *node has seven tabs that allow you to monitor, control, investigate, troubleshoot, and plan firewall operations. The first tab (****Dashboard****) contains six boxes that provide a quick summary of the detailed information on the other tabs. Whenever you need to investigate a particular event or reported issue in more detail, you switch from the* ***Dashboard*** *tab to the other tabs.* **d.** Select the **Alerts** tab. *The* ***Alerts*** *tab lists events at the ISA Server computer that are significant enough to merit an alert.* **e.** In the task pane, on the **Tasks** tab, click **Configure Alert Definitions.** **f.** In the **Alert Properties** dialog box, select the **Service shutdown** line (do *not* clear the check box for **Service shutdown**), and then click **Edit**. *On the* ***General*** *tab, in the* ***Severity*** *drop-down list, notice that ISA Server considers a service shutdown an Information alert.* **g.** In the **Service shutdown Properties** dialog box, select the **Events** tab. *On the* ***Events*** *tab, you specify the threshold to trigger an alert when the event occurs. In this example, the event is*

	a shutdown of any ISA Server service. **h.** Select the **Actions** tab. *On the **Actions** tab, you specify the action that should happen when an alert for this event is triggered (besides listing it on the **Alerts** tab).In this example, the only action is to report the alert in the Windows event log (Application log).* **i.** Click **Cancel** to close the Service shutdown Properties dialog box. **j.** Click **Cancel** to close the **Alerts Properties** dialog box. **k.** Notice that the current status of the ISA Server services is considered so significant that there is also a special tab (**Services**) that will specifically display the status of the services.
2. Simulate an unexpected shutdown of the service by using the **Services** console to stop the **ISA Server Job Scheduler**.	**a.** On the **Start** menu, click **Administrative Tools**, and then click **Services**. **b.** In the **Services** console, in the right pane, right-click **Microsoft ISA Server Job Scheduler**, and then click **Stop**. *The **ISA Server Job Scheduler** service is stopped. This simulates an unexpected shutdown of one of the ISA Server services.* **c.** Close the **Services** console.
3. Examine how an alert shows up on the **Alerts** tab and the **Dashboard** tab.	**a.** In the **ISA Server** console, on the **Alerts** tab, wait 30 seconds for the new alert (**Service shutdown**) to show up, or in the task pane, on the **Tasks** tab, click **Refresh Now**. *A new Information alert (**Service shutdown**) appears.* **b.** Select the **Dashboard** tab. Wait 30 seconds, or in the task pane, on the **Tasks** tab, click **Refresh Now**. *In the **Alerts** summary box, the **Service shutdown** Information alert is displayed as well. Notice the column that list the number of **New (not acknowledged yet)** alerts. The icon in the top left corner of each summary box indicates the highest severity or status of the information in that summary box. You may click the circle with the two up arrows to roll up the summary box.*
4. Investigate the **Service shutdown** alert and resolve the issue by starting the **ISA Server Job Scheduler** service on the **Services** tab	**a.** On the **Dashboard** tab, click the heading of the **Alerts** summary box to return to the **Alerts** tab. **b.** On the **Alerts** tab, select the **Service shutdown** alert, and then expand the **Service shutdown** alert. *The **Messages** area shows a general description of the event. (The service was stopped gracefully.)* **c.** Select the second **Service shutdown** alert line. *The **Messages** area shows a more specific description of*

	*the event. (The **ISA Server Job Scheduled** service was stopped gracefully.) When multiple similar alerts occur, they are grouped with a common general description.* **d.** In the task pane, on the **Tasks** tab, click **Acknowledge Selected Alerts**. *The **Status** of the **Service shutdown** alert changes from **New** to **Acknowledged** to indicate that you have seen this alert. **Acknowledged** alerts are removed from the **Alerts** summary box on the **Dashboard** tab as well.* **e.** Select the **Services** tab, and then in the task pane, on the **Tasks** tab, click **Refresh Now**. **f.** In the right pane, select **Microsoft ISA Server Job Scheduler**, and then in the task pane, on the **Tasks** tab, click **Start Selected Service**. *The **ISA Server Job Scheduler** service is started again.* **g.** On the **Alerts** tab, select the second acknowledged **Service shutdown** alert line. **h.** In the task pane, on the **Tasks** tab, click **Reset Selected Alerts**. **i.** Click **Yes** to confirm that you want to reset **Service shutdown**. *The **Service shutdown** alert is removed from the **Alerts** tab to indicate that you have resolved this alert. The alert will still be in the Windows event log (Application log).* ***Note:*** *This particular event (Service shutdown) is used as an example in this exercise. You would normally investigate a Service shutdown alert on the ISA Server computer more extensively, rather than just starting up the service again.*
5. Start a new online mode log query	**a.** In the **ISA Server** console, in the left pane, select **Monitoring**. **b.** In the right pane, on the **Logging** tab, click **Start Query**. (You may have to close the task pane to see the logging tab.) ***Start Query*** *starts a new online mode log query of the ISA Server log files. When a successful or failed connection is made through ISA Server, the records of the log file are displayed on the screen.*
Note: Perform the following steps on your client (e.g. PICLIENT)	
6. Use Internet Explorer to connect to **http://pages.intnet.mu/rhh**	**a.** Use Internet Explorer to connect to **http://pages.intnet.mu/rhh**
Note: Perform the following steps on your proxy server (e.g. PI)	
7. Create a filter definition for online mode logging. Filter by: **Destination IP** Condition: **Equals** Value: **202.123.2.119**	**a.** In the ISA Server console, in the left pane, select **Monitoring**, and then select the **Logging** tab. *ISA Server lists all Firewall service log file and Web Proxy log file records that have occurred since the since the Start Query command. This may include several of the*

	same denied NetBIOS Name Service and NetBIOS Datagram requests. The HTTP request to pages.intnet.mu/rhh (202.123.2.119) is also in this list. You can filter the on-screen display by creating a filter definition. **b.** In the task pane, on the **Tasks** tab, click **Edit Filter**. **c.** In the **Edit Filter** dialog box, complete the following information: • Filter by: **Destination IP** • Condition: **Equals** • Value: **202.123.2.119** **d.** Click **Add To List** to add the filter definition. **e.** Click **Start Query** to close the **Edit Filter** dialog box. **f.** The on-screen display is cleared, and the new filter definition (Destination IP equals 202.123.2.119) is in effect.
Note: Perform the following steps on your client (e.g. PICLIENT)	
8. Refresh the content of the Web page at **http://pages.intnet.mu/rhh** twice. • First press **CTRL+F5** (**CTRL+Refresh**) • then press **F5** (**Refresh**)	**a.** In Internet Explorer, ensure that the http://pages.intnet.mu/rhh page is opened. **b.** Hold the **CTRL** key, and click the **Refresh** button on the toolbar to refresh the content of the Web page, regardless of any changes **c.** Wait a few seconds, and then click the **Refresh** button on the toolbar (without the **CTRL**-key) to refresh the content of the Web page when it has changed *Internet Explorer displays the same Web page after each refresh.*
9. Attempt to open the non-existing Web page at http://pages.intnet.mu/rhh/noexist.htm	**a.** In Internet Explorer, in the **Address** box, type **http://pages.intnet.mu/rhh/noexist.htm**, and then press ENTER. *Internet Explorer cannot find the noexist.htm page (HTTP Error 404).* **b.** Close Internet Explorer.
Note: Perform the following steps on your proxy server (e.g. PI)	
10. View the online mode logging records for destination IP **202.123.2.119.** Add column: **HTTP Status Code**	**a.** On the **Logging** tab, wait 1 minute for the log file entries for destination IP 202.123.2.119 to appear on the screen. *A total of three or more log file records will appear for Destination IP 202.123.2.119 (Telecom Plus Personal Pages).* **b.** Right-click the **Log Time** heading, and then click **Add/ Remove Columns**. **c.** In the **Add/Remove Columns** dialog box, in the **Available columns** list box, select **HTTP Status Code**, and then click **Add ->**

	***HTTP Status Code** is moved into the **Displayed columns** list.* **d.** In the **Displayed columns** list, select **HTTP Status Code**, and then click **Move Up** eight times, until **HTTP Status Code** is just after **Protocol**. **e.** Click **OK** to close the **Add/Remove Columns** dialog box. *The following log file records are on the screen:* • *Protocol http - HTTP Method GET - HTTP Status Code 200* • *Protocol http - HTTP Method GET - HTTP Status Code 304* • *Protocol http - HTTP Method GET - HTTP Status Code 404* *Result code 200 means Success (is after CTRL+F5), 304 means Content not changed (is after F5), and 404 means File not found (is after attempt to get noexist.htm)*

Exercise 3

In this exercise, you will need to perform the following tasks using of ISA Server 2004.

1. You will need to prevent access to the following site: http://www.uom.ac.mu at all times
2. You will need to prevent any client from viewing any pdf document in their browser.
3. You should prevent access to the site http://www.bootdisk.com after 7.00 p.m.
4. You will need to disable MSN messenger form running on your internal network
5. You will need to show improvement in response time when implementing a cache of 200 MB on your ISA server when accessing random sites.
6. Export a script for all your settings of your respective ISA server once you are done with the above tasks.

You should create a report explaining briefly how you went about performing the above-mentioned tasks with screen captures as appropriate along with the relevant reports generated and your script file. All the above should be zipped into a single file with filename ***your_ISA_server_name*.zip** and uploaded on **ftp://intraweb/assignments/rh/secu2102/BCA07PT/ass2** and **ftp://intraweb/assignments/rh/secu2102/BCNS06FT/ass2**

Deadline: Friday 31st October 2008.

This assignment count as 15% of module weight.

PRACTICAL / TUTORIAL EXERCISES

This will be handed in class.

CASE STUDY (If Applicable)

This will be handed in class and can be found in past exam papers at the end of this document

SAMPLE OF PAST EXAMINATION PAPERS

(With the seal of the Resource Centre)

UNIVERSITY of TECHNOLOGY, MAURITIUS

B.Sc. (Hons.) in Computer Applications

BCA/05B/PT

Examinations for 2006 – 2007 / Semester 1

MODULE: NETWORK SECURITY

MODULE CODE: SECU2102

Duration: 2½ Hours

Reading time: 0 Minutes

Instructions to Candidates:

1. Attempt **all 3 questions**.
2. Start your answer to each question on a fresh page.
3. Questions carry **unequal** marks.
4. Total Marks = **100**.
5. Silent calculators are allowed in the Examination Room.

This question paper contains 3 questions and 4 pages.

ATTEMPT ALL 3 QUESTIONS

QUESTION 1: (32 Marks)

a) Describe the **six** main security services as in the ITU-T X.800 standard.

[6 marks]

b)

i) Using Fermat Theorem, calculate $3^{201} \bmod 11$

ii) Using Euclid algorithm, calculate $\mathrm{GCD}(4655, 12075)$

iii) Using Chinese Remainder Theorem, find the value of x given that:

$$x = 4 \bmod 5$$
$$x = 7 \bmod 8$$
$$x = 3 \bmod 9$$

[3+4+4 marks]

c) Assume the following mappings: $A \to 1, B \to 2, C \to 3, D \to 4$

Using the Rivest-Shamir-Adleman (RSA) cryptosystem with $p = 5$ and $q = 11$, calculate the corresponding cipher text which is transmitted for the following plaintext "BAD"

[11 marks]

d) Consider the following statement:

"It is often said that the length of a key k of a public-key cryptosystem is as strong as the same key length on a private-key cryptosystem."

How far do you agree with the following statement? Justify your answer.

[4 marks]

QUESTION 2: (34 Marks)

a) The Diffie-Hellman protocol is used to establish a shared secret key between two parties that do not share a key.
   The protocol runs as follows:

   p, g are public identifiers (p is prime and g is primitive root of p and g<p)
   $A \rightarrow B: g^x \bmod p$
   $B \rightarrow A: g^y \bmod p$

   i) What is the private key that is established in terms of g, x, y and p?
   ii) How do A and B generate this key?
   iii) What conditions are required to stop an outside observer that can read these two messages, from also generating the private key?
   iv) Show how a *"man in the middle attack"* can trick A and B into using a private key which he also knows

[1+2+2+4 marks]

b)
   i) Describe how a Message Authentication Code (MAC) is generated.
   ii) Explain the desirable properties that a secure hash function should possess.
   iii) Explain how a digital signature is constructed and explain why a digital signature is favoured to a MAC nowadays.

[3+4+6 marks]

c)
   i) Outline the SSL Record Protocol operations.
   ii) What are the main differences between IPSec and SSL.
   iii) Describe briefly the IPSec suite of protocols and how it is related to virtual private networks (VPN).

[2+4+6 marks]

QUESTION 3: (34 Marks)

a) A firewall can be classified according to which layer of the OSI model the protection is at. Using this classification, describe the different types of firewalls.

[6 marks]

b) Why is a demilitarized zone (DMZ) used in a firewall configuration?
   Describe the most important characteristics of a DMZ.

[2+4 marks]

c) Describe five different types of computer viruses. Identify the most dangerous type and justify your choice.

[5+1 marks]

d) Briefly describe the essential features Microsoft ISA Server. How does it works and how is it configured.

[8 marks]

e) Explain why WEP is no longer secure for wireless networks nowadays.
   Give two techniques that can be used to overcome this problem.

[2+2 marks]

f) Describe the following form of attacks and rate them in order of lowest to highest advantage gained by a cryptanalyst:
   i. Chosen ciphertext
   ii. Known plaintext
   iii. Chosen plaintext

[3+1 marks]

END OF PAPER

UNIVERSITY of TECHNOLOGY, MAURITIUS

B.Sc. (Hons.) in Computer Applications

BCA/07A/PT

Examinations for 2007 – 2008 / Semester 2

MODULE: NETWORK SECURITY

MODULE CODE: SECU2102

Duration: 2 Hours and 15 minutes

Reading time: 0 Minutes

Instructions to Candidates:

1. Attempt **Section A** and **any 2 questions from Section B**.
2. Start your answer to each question on a fresh page.
3. Questions carry **unequal** marks.
4. Total Marks = **100**
5. Silent calculators are allowed in the Examination Room.

This question paper contains 4 questions and 5 pages.

SECTION A

QUESTION 1: (36 Marks)

a) Contrast the following security services and provide one mechanism to enforce each type of service mentioned below:

i. Authorization vs. Authentication
ii. Data Confidentiality vs. Data Integrity

[4+4 marks]

b) Using CRT or otherwise, solve the three following congruencies to find the smallest, positive value of x:

$$x \equiv 3 \bmod 5$$
$$x \equiv 1 \bmod 3$$
$$x \equiv 6 \bmod 7$$

[8 marks]

c) Assume the following mappings: $A \rightarrow 1$, $B \rightarrow 2$, $C \rightarrow 3$, $D \rightarrow 4$, $E \rightarrow 5$

Using the RSA cryptosystem with $p = 5$ and $q = 7$, calculate the corresponding cipher text which is transmitted for the following plaintext "ACED"

[12 marks]

d) Describe the following form of attacks and rate them in order of lowest to highest advantage gained by a cryptanalyst:

i. Chosen ciphertext
ii. Known plaintext
iii. Chosen plaintext

[6+2 marks]

SECTION B

ATTEMPT ANY TWO QUESTIONS

QUESTION 2: (32 Marks)

a) Describe briefly how the Diffie-Hellman key exchange works?
Show how it is vulnerable to the *"man in the middle"* attack.

[10+4 marks]

b) Suppose that someone suggest the following way to confirm that the two of you are both in possession of the same secret key. You create a random bit string the length of the key, XOR it with the key, and send the result over the channel. Your partner XORs the incoming block with the key (which should be the same as your key) and sends it back. You check, and if what you receive is your original random string, you have verified that your partner has the same secret key, yet neither of you has ever transmitted the key.
Is there a flaw in this scheme? Explain.

[6 marks]

c)

i. Explain how a Simple Hash Function operates and what is its main weakness?
ii. What do you understand by a hash function having "strong collision resistance" properties?
iii. Explain how a digital signature is constructed and explain why a digital signature is favoured to a Message Authentication Code nowadays.

[4+2+6 marks]

QUESTION 3: (32 Marks)

a) A firewall can be classified according to which layer of the OSI model the protection is at. Using this classification, describe the different types of firewalls.

[6 marks]

b) Give one advantage and one disadvantage of IPSec over SSL.
How is IPSec operating in Tunnel mode different from Transport mode?

[4+2 marks]

c) Describe four different types of computer malware and justify which type poses most threat to:

i. availability of network resources
ii. availability of data.

[8+4 marks]

d) Explain why WEP is no longer secure for wireless networks nowadays.
Give two techniques that can be used to overcome this problem.

[4+4 marks]

QUESTION 4: (32 Marks)

a) Describe the steps needed to configure the following rules using a proxy solution like Microsoft ISA Server in a networked environment.

   i. You wish to prevent some specific hosts from downloading audio content in a new format whose file extension is not yet catered for on your proxy.
   ii. You wish to deny a newly developed P2P application from running inside your internal network?
   iii. You want to prevent users from accessing a certain external web server who regularly changes its IP address.

[4+4+3 marks]

b) Describe briefly the following:

   i. Demilitarized Zone
   ii. Advanced Encryption Standard
   iii. Kerberos
   iv. Avalanche Effect
   v. One-time Pad
   vi. 3-DES

[2+6+4+2+2+5 marks]

*****END OF PAPER*****