Introduction to Computer Security

Lecture Set 1

What is Computer Security?

(we academics love our definitions)

The **NIST Computer Security Handbook** defines computer security as:

- "The protection afforded to an automated information system in order to attain the applicable objectives of preserving the integrity, availability and confidentiality of information system resources"
    - This includes hardware, software, firmware, information, data, and telecommunications (among others that might not be listed)

Challenges in Computer Security

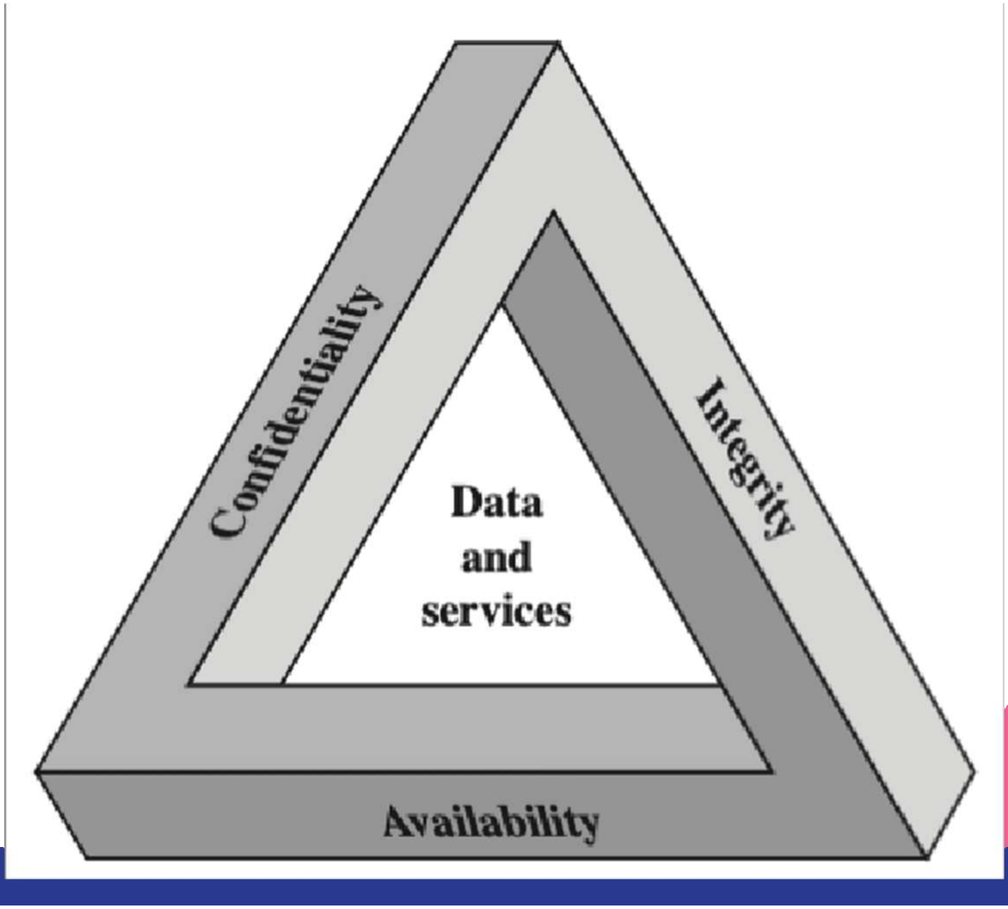
- Computer security is not as simple as it might first appear
- Attackers only need to find a single weakness, the engineer needs to find all weaknesses
- Users and system managers tend to not see the benefits of security until a failure occurs
- Potential attacks on the security features must be considered
- Procedures used to provide particular services are often counterintuitive
- Physical and logical placement needs to be determined
- Security requires regular and constant monitoring (at high cost)
- Often an afterthought to be incorporated into a system after the design is complete

Challenges in Computer Security, ctd.

- Additional algorithms or protocols may be involved (complexity, distribution of "secret" information to users)
- Thought of as an impediment to efficient and user-friendly operation

The CIA Triad

(which has little to do with the Central Intelligence Agency)



The CIA Triad (2)

- Confidentiality
  - Preserving authorized restrictions on information access and disclosure, including means for protecting personal privacy and proprietary information. A loss of confidentiality is the unauthorized disclosure of information.
- Integrity
  - Guarding against improper information modification or destruction, including ensuring information non-repudiation and authenticity. A loss of integrity is the unauthorized modification or destruction of information.
- Availability
  - Ensuring timely and reliable access to and use of information. A loss of availability is the disruption of access to or use of information or an information system

Note: the CIA Triad is a balancing act. As soon as we emphasise one category, we sacrifice elements of the others. Meaning, we can't have high confidentiality and also high availability; those two things are mutually exclusive.

Security Concepts and Relations

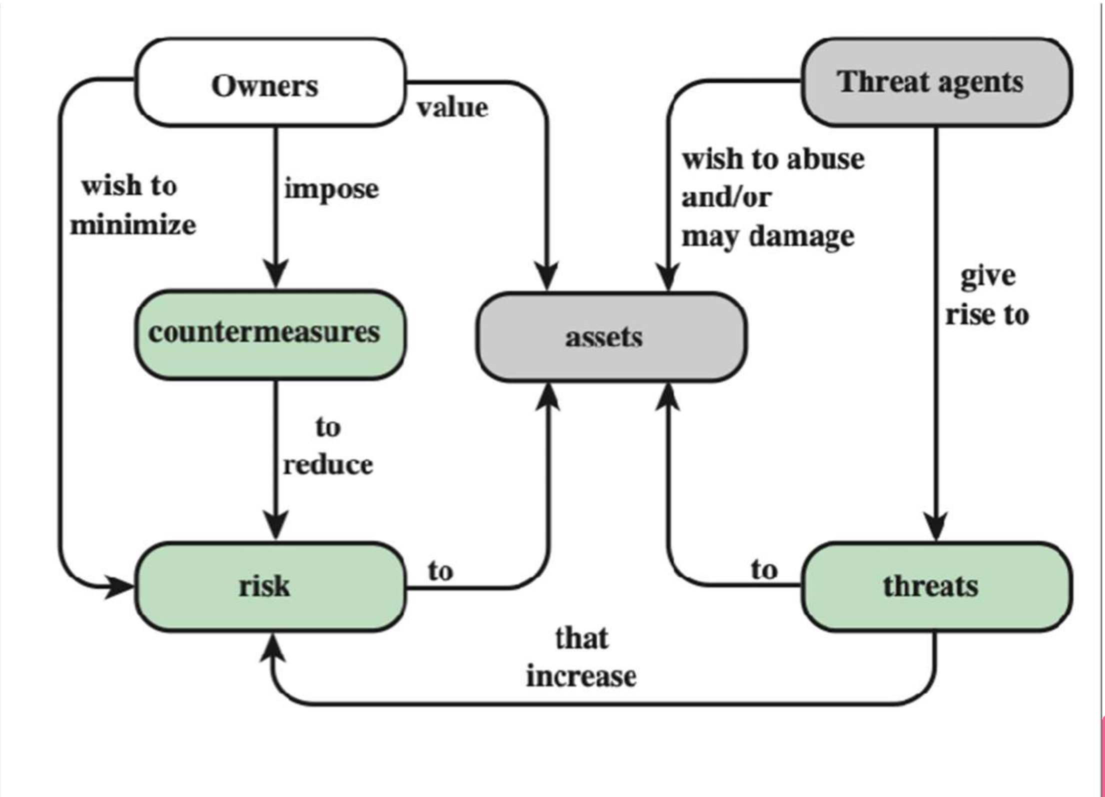


Figure 1.1 Security Concepts and Relationships

Assets of Computing Systems

- Hardware
  - Including computer systems and other data processing, data storage, and data communications devices
- Software
  - Including the operating system, system utilities, and applications
- Data
  - Including files and databases, as well as security-related data, such as password files.
- Communication and Networks
  - Local and wide area network communication links, bridges, routers, and so on.

Vulnerabilities, Threats and Attacks

- Categories
  - Leaks (loss of *confidentiality*)
  - Corruption (loss of *integrity*)
  - Unavailable or slow (loss of *availability*)
- Threats
  - Things that are capable of exploiting a vulnerability
- Attacks (executed threats)
  - Passive
    - An attempt to learn or make use of information from the system that does not affect system resources
  - Active
    - An attempt to alter system resources or affect their operation
  - Insider
    - Initiated by an entity inside the security perimeter or an authorized user
  - Outsider
    - Initiated from outside the perimeter or an illegitimate user

Countermeasures

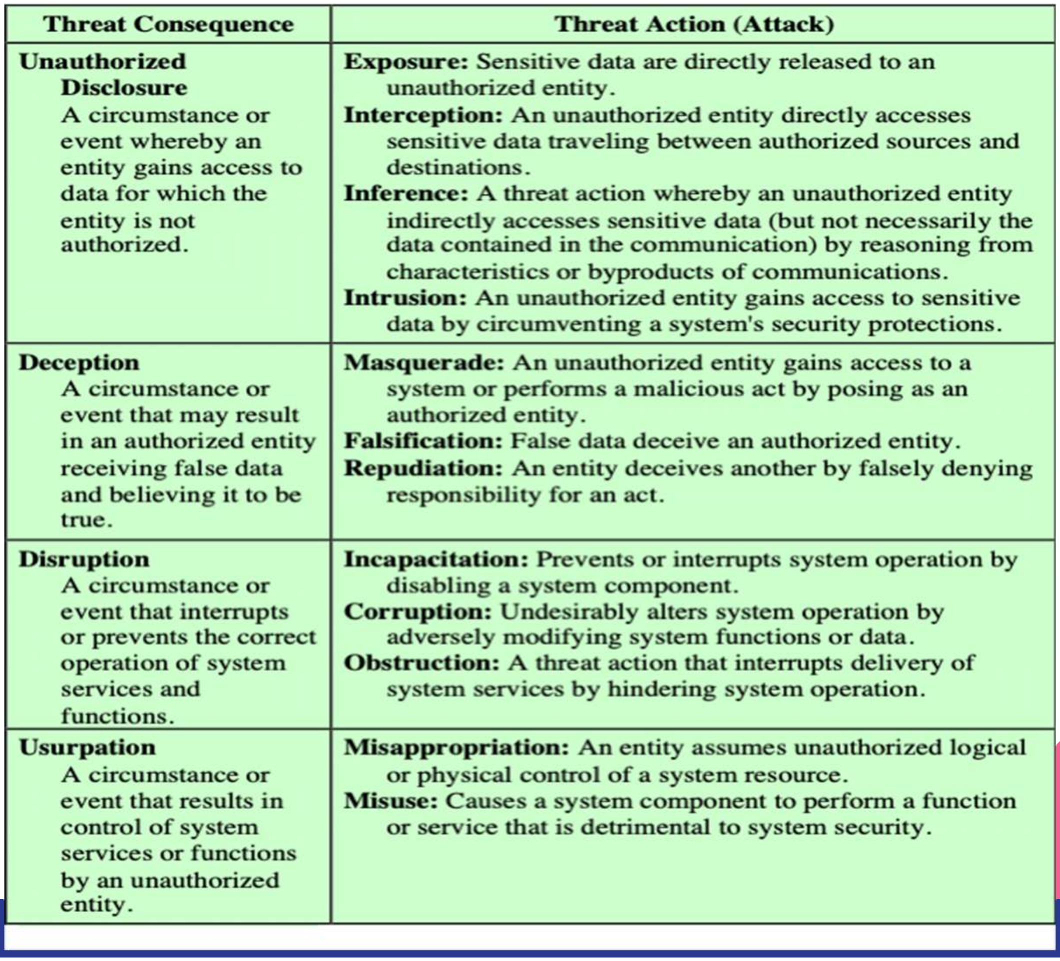
Some means to deal with a security attack:

- Prevent
- Detect
- Recover

Countermeasures are also not without their own issues:

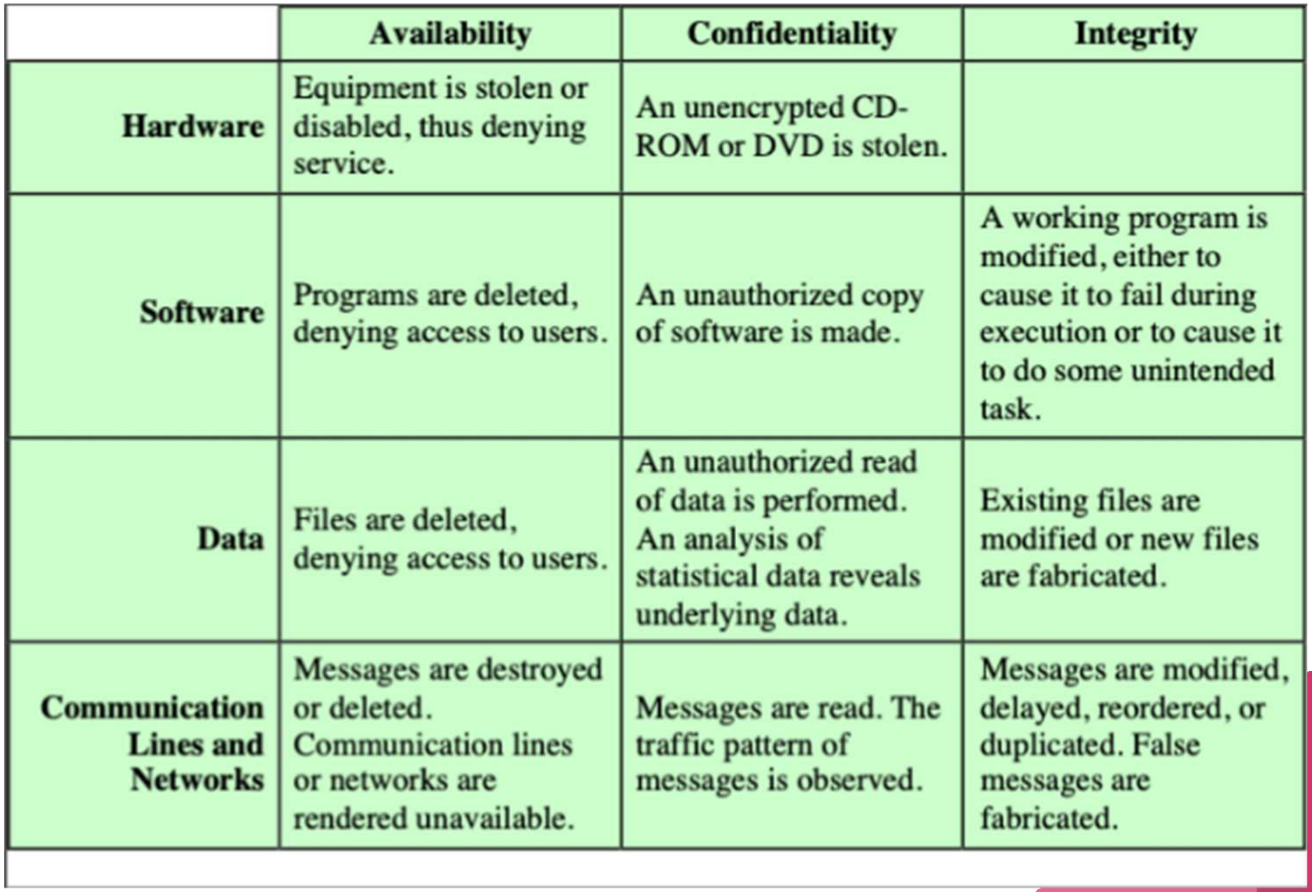
- May not fully neutralize threats
- May introduce their own vulnerabilities
- Ultimately, only used to minimize risk

Threat Consequences



Threat Consequence	Threat Action (Attack)
Unauthorized Disclosure A circumstance or event whereby an entity gains access to data for which the entity is not authorized.	**Exposure:** Sensitive data are directly released to an unauthorized entity. **Interception:** An unauthorized entity directly accesses sensitive data traveling between authorized sources and destinations. **Inference:** A threat action whereby an unauthorized entity indirectly accesses sensitive data (but not necessarily the data contained in the communication) by reasoning from characteristics or byproducts of communications. **Intrusion:** An unauthorized entity gains access to sensitive data by circumventing a system's security protections.
Deception A circumstance or event that may result in an authorized entity receiving false data and believing it to be true.	**Masquerade:** An unauthorized entity gains access to a system or performs a malicious act by posing as an authorized entity. **Falsification:** False data deceive an authorized entity. **Repudiation:** An entity deceives another by falsely denying responsibility for an act.
Disruption A circumstance or event that interrupts or prevents the correct operation of system services and functions.	**Incapacitation:** Prevents or interrupts system operation by disabling a system component. **Corruption:** Undesirably alters system operation by adversely modifying system functions or data. **Obstruction:** A threat action that interrupts delivery of system services by hindering system operation.
Usurpation A circumstance or event that results in control of system services or functions by an unauthorized entity.	**Misappropriation:** An entity assumes unauthorized logical or physical control of a system resource. **Misuse:** Causes a system component to perform a function or service that is detrimental to system security.

Assets and Examples of Threats



	Availability	Confidentiality	Integrity
Hardware	Equipment is stolen or disabled, thus denying service.	An unencrypted CD-ROM or DVD is stolen.	
Software	Programs are deleted, denying access to users.	An unauthorized copy of software is made.	A working program is modified, either to cause it to fail during execution or to cause it to do some unintended task.
Data	Files are deleted, denying access to users.	An unauthorized read of data is performed. An analysis of statistical data reveals underlying data.	Existing files are modified or new files are fabricated.
Communication Lines and Networks	Messages are destroyed or deleted. Communication lines or networks are rendered unavailable.	Messages are read. The traffic pattern of messages is observed.	Messages are modified, delayed, reordered, or duplicated. False messages are fabricated.

Passive and Active Threats

- Passive
  - Attempts to learn about system without affecting resources
  - Can consist of eavesdropping or monitoring
- Active
  - Attempts to affect system resources or operations
  - Typically involve some modification of data or falsifying data
  - Four categories:
    - Replay
    - Masquerade
    - Modification of Messages
    - Denial of Service (DoS)

Design Principles

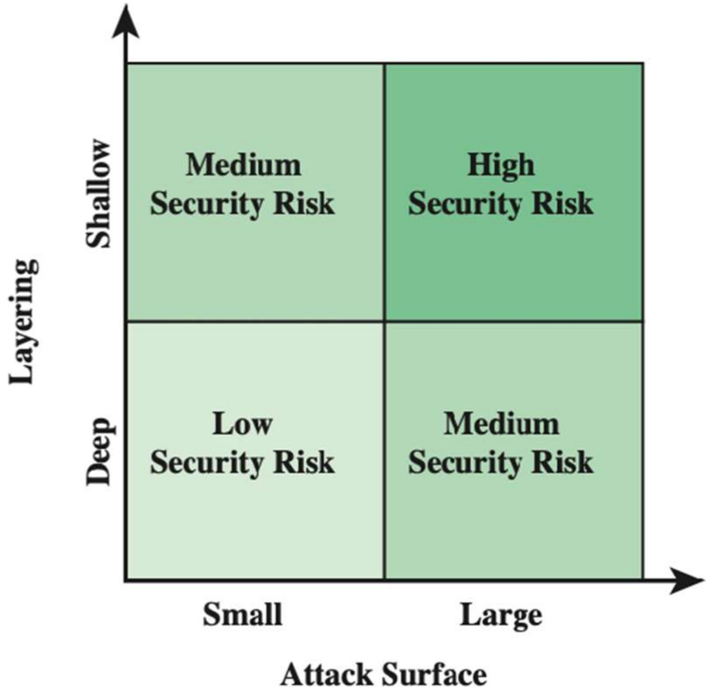
- Economy of Mechanism
- Fail-Safe Defaults
- Complete Mediation
- Open Design
- Separation of Privilege
- Least Privilege
- Least Common Mechanism
- Psychological Acceptability
- Isolation
- Encapsulation
- Modularity
- Layering
- Least Astonishment

Attack Surfaces

Consist of the reachable and exploitable vulnerabilities in a system

- Examples:
  - Open ports
  - Services within a firewall
  - Interpretive code (eg. XML, PHP, Office Docs)
  - Interfaces (eg. SQL, web forms)
  - People! (eg. social engineering)

Attack Surfaces, ctd.



	Small	Large
Shallow	Medium Security Risk	High Security Risk
Deep	Low Security Risk	Medium Security Risk

Figure 1.3 Defense in Depth and Attack Surface

Attack Surfaces, ctd.

Consider an automated teller machine (ATM) at a typical bank:

- In order of priority (high, medium, low) where each priority level has one item, how would you design an ATM machine using the CIA triad?
  - Which item should have a **high** priority?
  - Which item should have a **medium** priority?
  - Which item should have a **low** priority?
  - **Why did you choose** these priority levels?

  Remember, you cannot make everything a high priority. You must make choices based on your design. What are the problems with your design? Would changing priorities fix your problems? What problems might be introduced by changing priorities?

Attack Trees

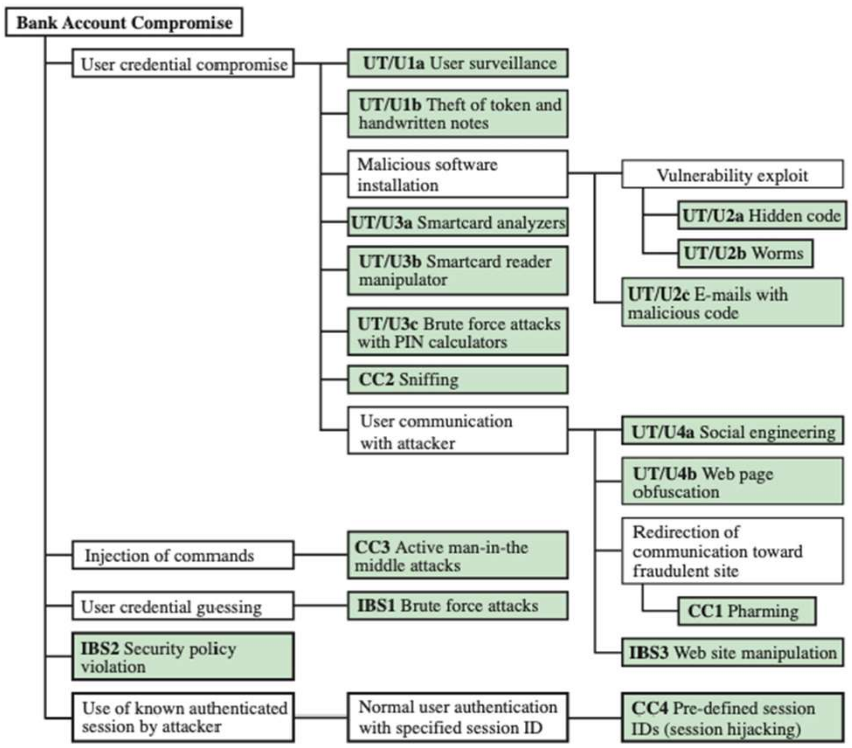

Bank Account Compromise
User credential compromise
UT/U1a User surveillance
UT/U1b Theft of token and handwritten notes
Malicious software installation
Vulnerability exploit
UT/U2a Hidden code
UT/U2b Worms
UT/U2c E-mails with malicious code
UT/U3a Smartcard analyzers
UT/U3b Smartcard reader manipulator
UT/U3c Brute force attacks with PIN calculators
CC2 Sniffing
User communication with attacker
UT/U4a Social engineering
UT/U4b Web page obfuscation
Redirection of communication toward fraudulent site
CC1 Pharming
IBS3 Web site manipulation
Injection of commands
CC3 Active man-in-the middle attacks
User credential guessing
IBS1 Brute force attacks
IBS2 Security policy violation
Use of known authenticated session by attacker
Normal user authentication with specified session ID
CC4 Pre-defined session IDs (session hijacking)

Figure 1.4 An Attack Tree for Internet Banking Authentication

Cryptography

Lecture Set 2

What is Cryptography?

Cryptography is the study of techniques for secure communication in the presence of third parties, which are referred to as adversaries

Modern cryptography is not necessary synonymous with *encryption*, which is the technique of securing a message against adversaries

A *cipher* (or cypher) is a pair of algorithms that can create the encryption and the reversing decryption

Encryption Techniques

Regardless of how advanced many of the encryption techniques seem, all of them ultimately boil down to the use of two different methods:

- Substitution
    - Swapping one letter for another
- Transposition
    - Rearranging the relative position of one letter for another

Encryption Techniques (2)

We can also categorize encryption techniques by several different means:

- The number of keys used
  - Symmetric
    - Sender and receiver use the same key
  - Asymmetric
    - Sender and receiver use different keys
- The way plaintext is processed
    - Block Ciphers
      - Processes input one block of elements at a time
    - Stream Ciphers
      - Continuously process elements as they arrive

Cryptographical Eras

In the computer age, we typically divide up cryptographic techniques into two main eras:

- Classical cryptographical era
  - All methods of analog encryption, and mostly used in the pre-WWII historical eras
- Modern cryptographical era
  - Virtually all forms of digital encryption since WWII, and encompasses the majority of methods we will discuss in this course

History of Cryptography

Interestingly, cryptography has a long and rich history:

- The Caesar Cipher

  One of the oldest forms of encryption, rumored to be used by Julius Caesar himself; he would encrypt messages to his legion commanders by employing a simple letter shift of three letters across the alphabet set. Internet forums also commonly use a form of Caesar cypher to hide "spoilers" in the form of ROT13 (meaning, 13 rotations instead of 3)*.

  Cryptii: https://cryptii.com/pipes/caesar-cipher

History of Cryptography, ctd.

- Substitution Cipher

  While technically the Caesar cipher is a form of substitution cipher, cryptographers typically reserve the phrase for slightly more advanced forms of crypto. Substitution ciphers involve replacing letters with differing letters.

  Cryptii: https://cryptii.com/pipes/alphabetical-substitution

History of Cryptography, ctd.

- Book Cipher

  Each word in the secret message would be replaced with a number which represents the same word in the book. For example, if the word "`attack`" appeared in the book as word number 713, then "`attack`" would be replaced with this number. The result would be an encoded message that looked something like this:

  `713-23-245-45-124-1269-586-443-8-234`

  To decipher the message you simply count the number of words in the book and write down each one.

History of Cryptography, ctd.

Symmetric Encryption

- The universal technique for providing confidentiality for transmitted or stored data.
- Also referred to as *conventional encryption* or *single-key encryption*
- Requirements for secure use:
  - Need a strong encryption algorithm
  - *Sender* and *receiver* must have copies of the *same key*
  - The key must remain private and secure from other parties

Symmetric Encryption, ctd.

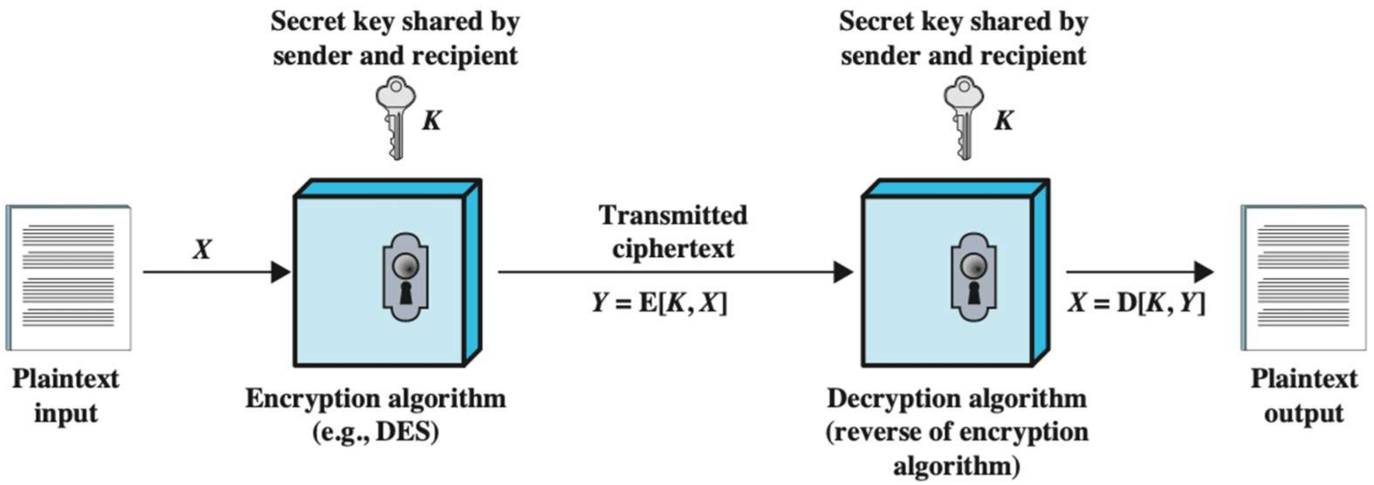

Secret key shared by sender and recipient
K
Secret key shared by sender and recipient
K
X
Transmitted ciphertext
$Y = E[K, X]$
$X = D[K, Y]$
Plaintext input
Encryption algorithm (e.g., DES)
Decryption algorithm (reverse of encryption algorithm)
Plaintext output

Figure 2.1 Simplified Model of Symmetric Encryption

Symmetric Encryption, ctd.

Symmetric encryption is not invulnerable by a long shot. There are two main methods of attacking any form of symmetric encryption:

- Cryptanalytic attacks
  - Rely on some knowledge of the underlying algorithm
  - Can also use the encrypted text against itself in an attack by looking for patterns
- Brute-force attacks
  - Attacker attempts to try all possible keys on the ciphertext until a pattern emerges
    - On average, half of all keys must be tried (referred to as the *half life* of the algorithm)

Symmetric Encryption, ctd.

Symmetric encryption can further be attacked by making use of the ciphertext itself. *Every* message has inherent patterns in it, no matter how complex the encryption, and we can derive some information about the message itself without ever having to crack it.

- Ciphertext only
  - If a ball fell from space with a message in an alien language on it, we would have no idea how to translate the contents of that message. Rarely will we ever encounter such a thing on Earth. This is an example of only knowing the ciphertext.
- Known plaintext
  - More often, we will have some idea about the message we are attempting to decrypt, even if it is only the language the plaintext message will be in. We also might know the rough type of encryption that was used to secure that message.

Symmetric Encryption, ctd.

- Chosen plaintext
  - Maybe we are fortunate to have a greater idea about what the message plaintext should say. Is it a secret message to a foreign agent? Is it a shopping list? We can use that knowledge to further analyze the message.
- Chosen ciphertext
  - We might also have a bit of insight as to how the message was encrypted. Was the message encrypted using DES? AES? What kinds of computing machines were used? We can also use that knowledge for our cracking benefit.
- Chosen text
  - Finally, we may have some understanding about what things should be contained within the message. If it's a message from a secret agent, it's highly unlikely that it is his Amazon wish list, and more likely that the message contains something about international secrets. Even better if we know the general gist of the message.

What makes Computationally-Secure Encryption?

We can never have truly secure encryption. Any encryption that we deploy will always have faults and insecurities. So, how do we know if the encryption we are using is secure *enough*?

- Encryption is computationally secure if:
  - Cost of breaking cipher exceeds value of information
  - Time required to break cipher exceeds the useful lifetime of the information
- Usually very difficult to estimate the amount of effort required to break
- Helps to estimate time and cost of a brute-force attack

Symmetric Encryption Compared

	DES	3-DES	AES
Plaintext block size	64	64	128
Ciphertext block size	64	64	128
Key size	56	112 or 168	128-512 and up

DES = Data Encryption Standard

AES = Advanced Encryption Standard

* All sizes are in bits

Symmetric Encryption Compared, ctd.

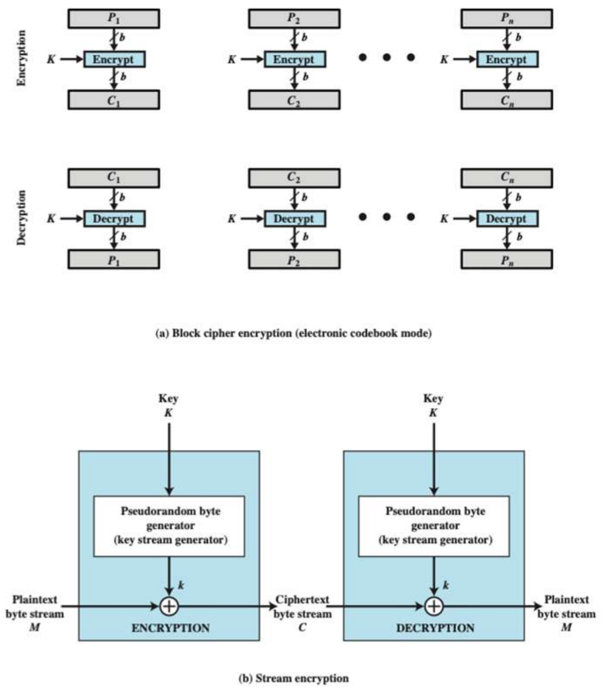
Key Size	Cipher	Number of Keys	T(10^9) dec/sec	T(10^13) dec/sec
56 bits	DES	2^56	2^55 ns = 1.125 yrs	1 hour
128 bits	AES	2^128	2^127 ns = 5.3*10^21 yrs	5.3*10^17 yrs
168 bits	3-DES	2^168	2^167 ns = 5.8*10^33 yrs	5.8*10^29 yrs
192 bits	AES	2^192	2^191 ns = 9.8*10^40 yrs	9.8*10^36 yrs
256 bits	AES	2^256	2^255 ns = 1.8*10^60 yrs	1.8*10^56 yrs

Thought Experiment: Life Expectancy of the Sun

WolframAlpha: Life Expectancy of the Sun

Consider: Our weakest form of AES encryption has a half life of `5.3*10^17` years with a modern supercomputer. If the sun will only last roughly `1*10^10` years, why do we need encryption that lasts longer than that? **Discuss**.

Types of Symmetric Encryption



(a) Block cipher encryption (electronic codebook mode)

(b) Stream encryption

Figure 2.2 Types of Symmetric Encryption

Data Encryption Standard (DES)

The original encryption standard. Today, it is severely outdated.

- Adopted in 1977 by National Bureau of Standards (Now NIST--the same people who bring you the atomic clock time every day!)
- DES is a (very) minor variation of the cipher created by Horst Feistel in the early 1970s.

Data Encryption Standard, ctd.

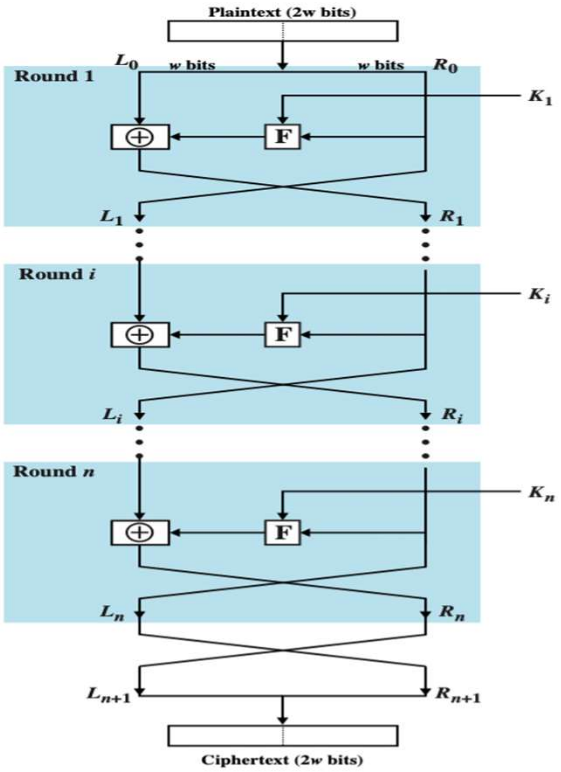


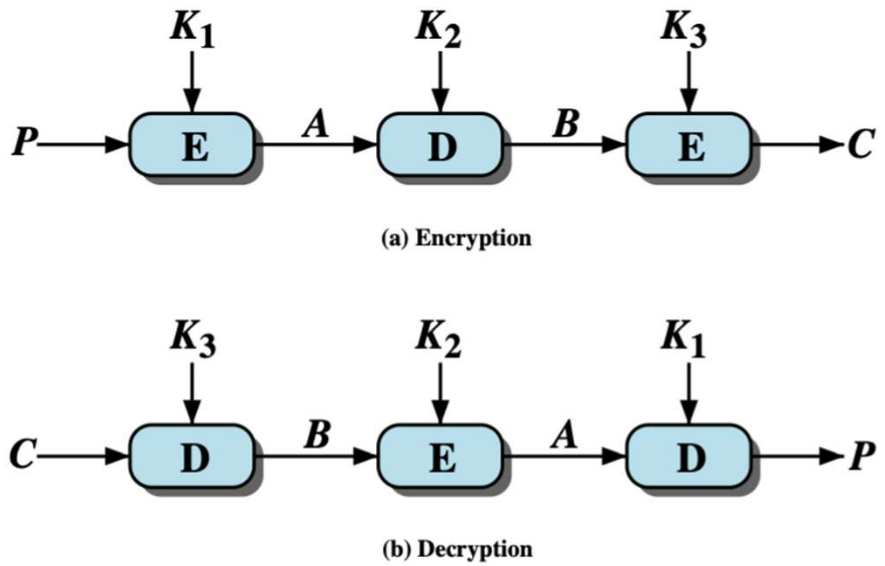
Figure 20.1 Classical Feistel Network

Triple DES (3-DES)

Since the late 1980s, the faults and insecurity of DES have been well-known. Unfortunately, at that time, a suitable replacement for DES was years away. A stop-gap measure was created to marginally secure DES so that it could continue to be used without significant cost to the implementers. That stop-gap was 3DES.

- Properties of 3-DES:
    - Exactly the same underlying algorithm as DES
    - Performs the algorithm three separate times
    - Is nearly three times as a slow as the original DES algorithm with only a slight improvement of encryption strength.

Triple DES, ctd.



(a) Encryption

(b) Decryption

Figure 20.2 Triple DES

Advanced Encryption Standard (AES)

- Any board game or tabletop RPG players out there?
- Internet privacy was spearheaded by an organization called the EFF, in 1990
- How does this relate to board games and RPGs?

Advanced Encryption Standard, ctd.

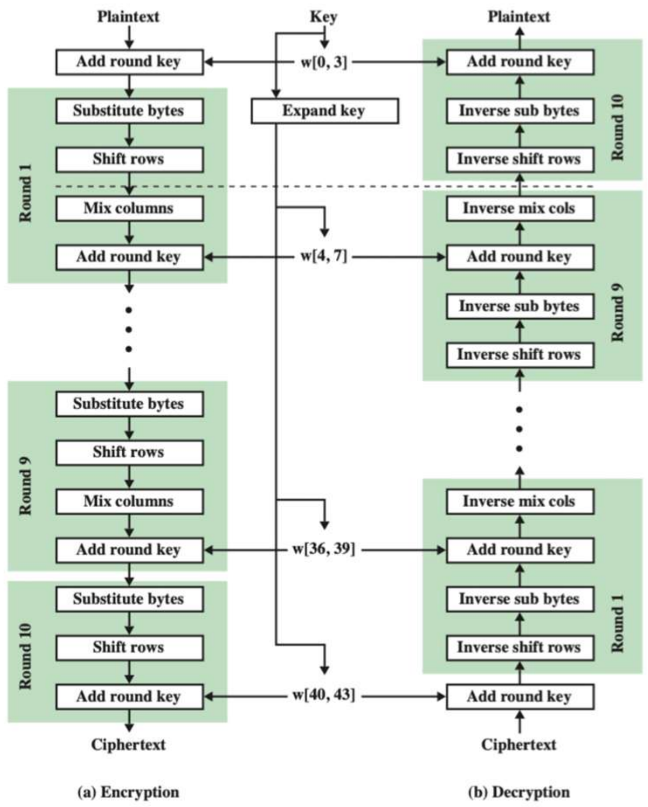


Figure 20.3 AES Encryption and Decryption

Advanced Encryption Standard, ctd.

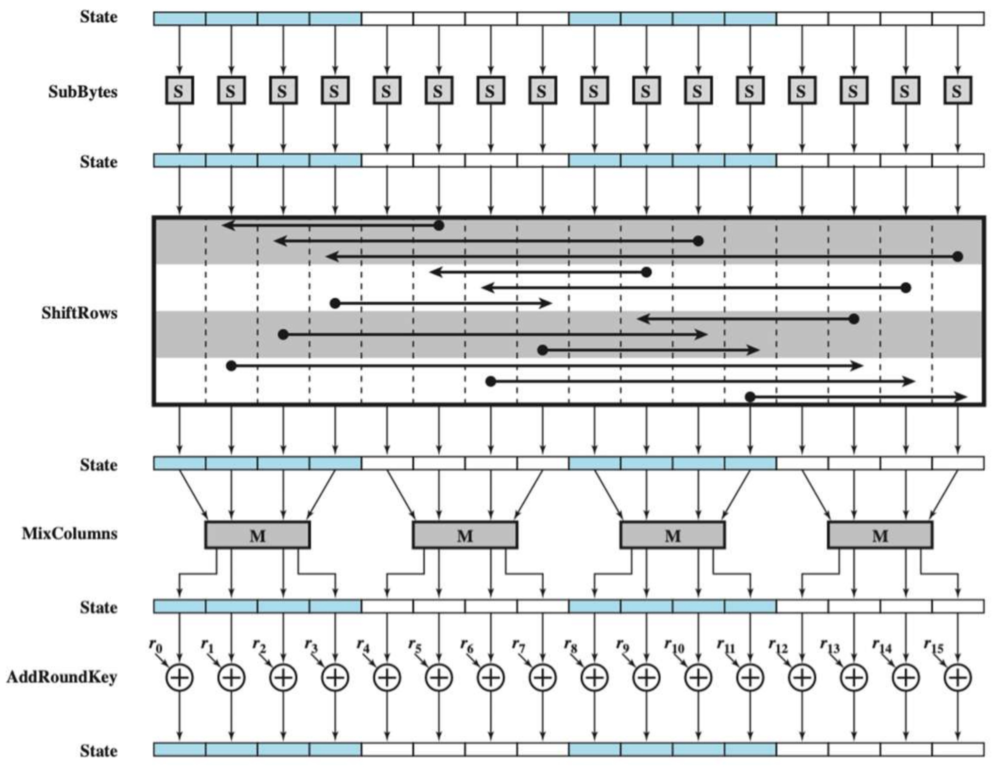


Figure 20.4 AES Encryption Round

Advanced Encryption Standard, ctd.

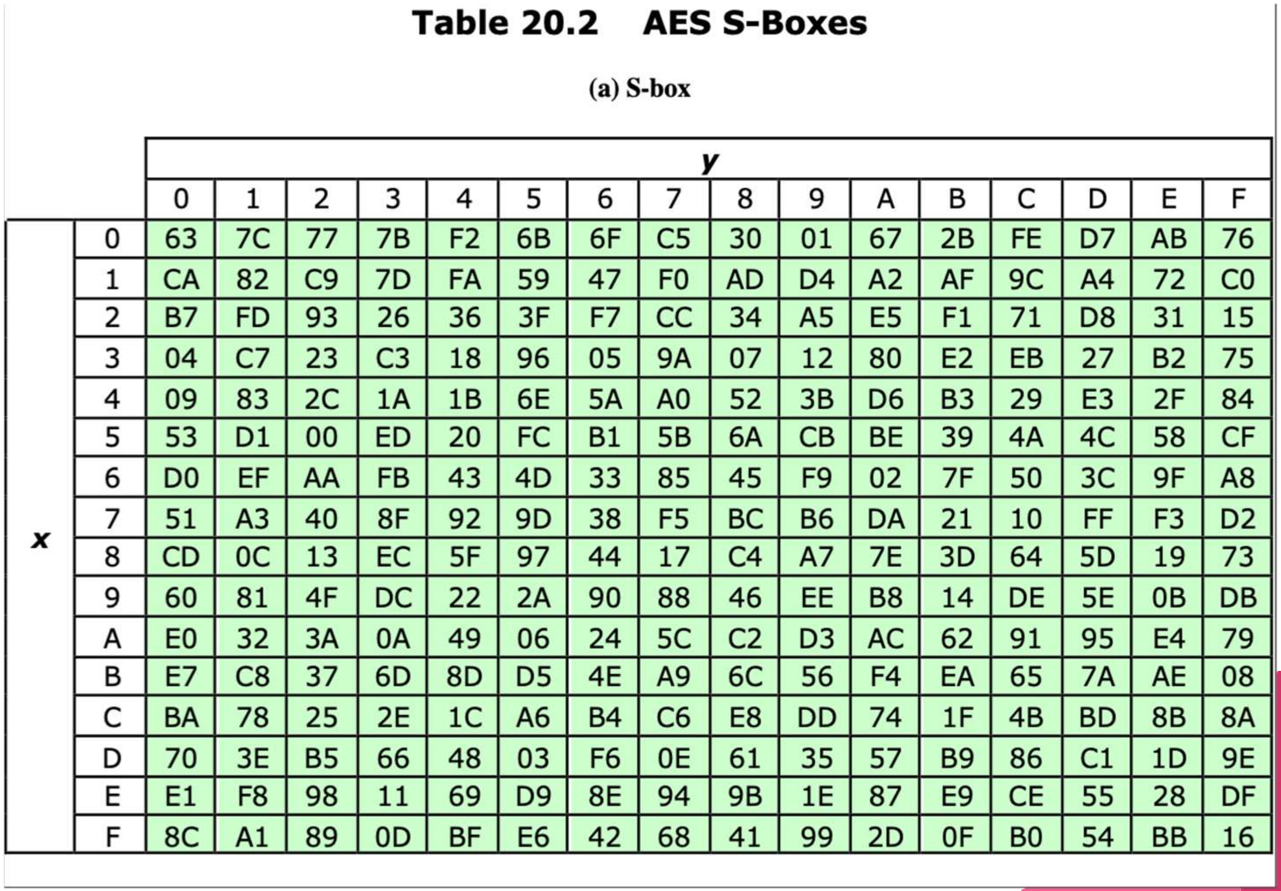
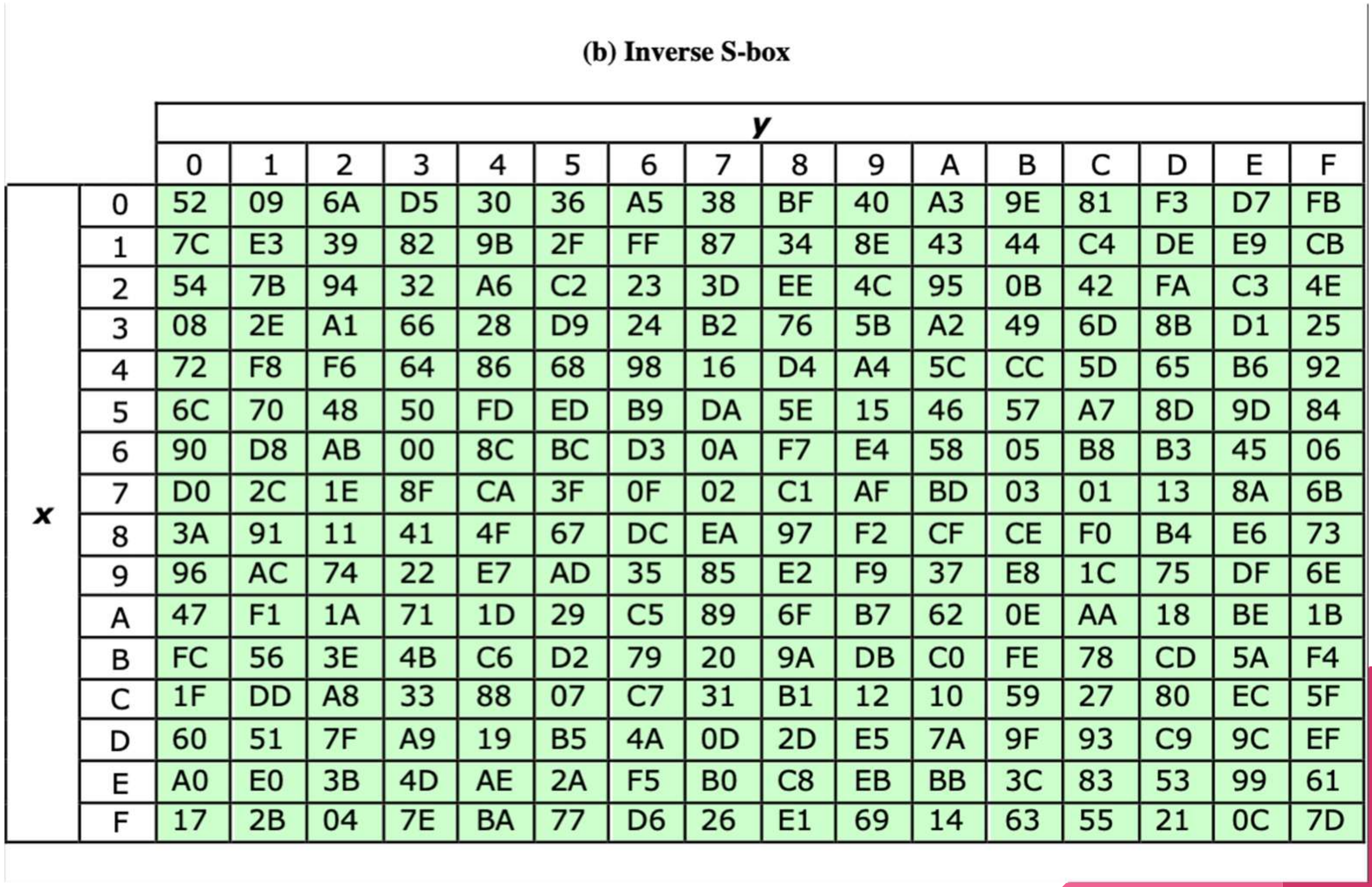


Table 20.2 AES S-Boxes

(a) S-box

		y															
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
x	0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
	1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
	2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
	3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
	4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
	5	53	D1	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
	6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8
	7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
	8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
	9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
	A	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79
	B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	08
	C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
	D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
	E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
	F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

Advanced Encryption Standard, ctd.



(b) Inverse S-box

		y															
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
x	0	52	09	6A	D5	30	36	A5	38	BF	40	A3	9E	81	F3	D7	FB
	1	7C	E3	39	82	9B	2F	FF	87	34	8E	43	44	C4	DE	E9	CB
	2	54	7B	94	32	A6	C2	23	3D	EE	4C	95	0B	42	FA	C3	4E
	3	08	2E	A1	66	28	D9	24	B2	76	5B	A2	49	6D	8B	D1	25
	4	72	F8	F6	64	86	68	98	16	D4	A4	5C	CC	5D	65	B6	92
	5	6C	70	48	50	FD	ED	B9	DA	5E	15	46	57	A7	8D	9D	84
	6	90	D8	AB	00	8C	BC	D3	0A	F7	E4	58	05	B8	B3	45	06
	7	D0	2C	1E	8F	CA	3F	0F	02	C1	AF	BD	03	01	13	8A	6B
	8	3A	91	11	41	4F	67	DC	EA	97	F2	CF	CE	F0	B4	E6	73
	9	96	AC	74	22	E7	AD	35	85	E2	F9	37	E8	1C	75	DF	6E
	A	47	F1	1A	71	1D	29	C5	89	6F	B7	62	0E	AA	18	BE	1B
	B	FC	56	3E	4B	C6	D2	79	20	9A	DB	C0	FE	78	CD	5A	F4
	C	1F	DD	A8	33	88	07	C7	31	B1	12	10	59	27	80	EC	5F
	D	60	51	7F	A9	19	B5	4A	0D	2D	E5	7A	9F	93	C9	9C	EF
	E	A0	E0	3B	4D	AE	2A	F5	B0	C8	EB	BB	3C	83	53	99	61
	F	17	2B	04	7E	BA	77	D6	26	E1	69	14	63	55	21	0C	7D

Practical Security Issues

- Typically symmetric encryption is applied to a unit of data larger than a single 64-bit or 128-bit block
- Electronic codebook (ECB) mode is the simplest approach to multiple-block encryption
  - Each block of plaintext is encrypted using the same key
  - Cryptanalysts may be able to exploit regularities in the plaintext
- Modes of operation
  - Alternative techniques developed to increase the security of symmetric block encryption for large sequences
  - Overcomes the weaknesses of ECB

Block and Stream Ciphers

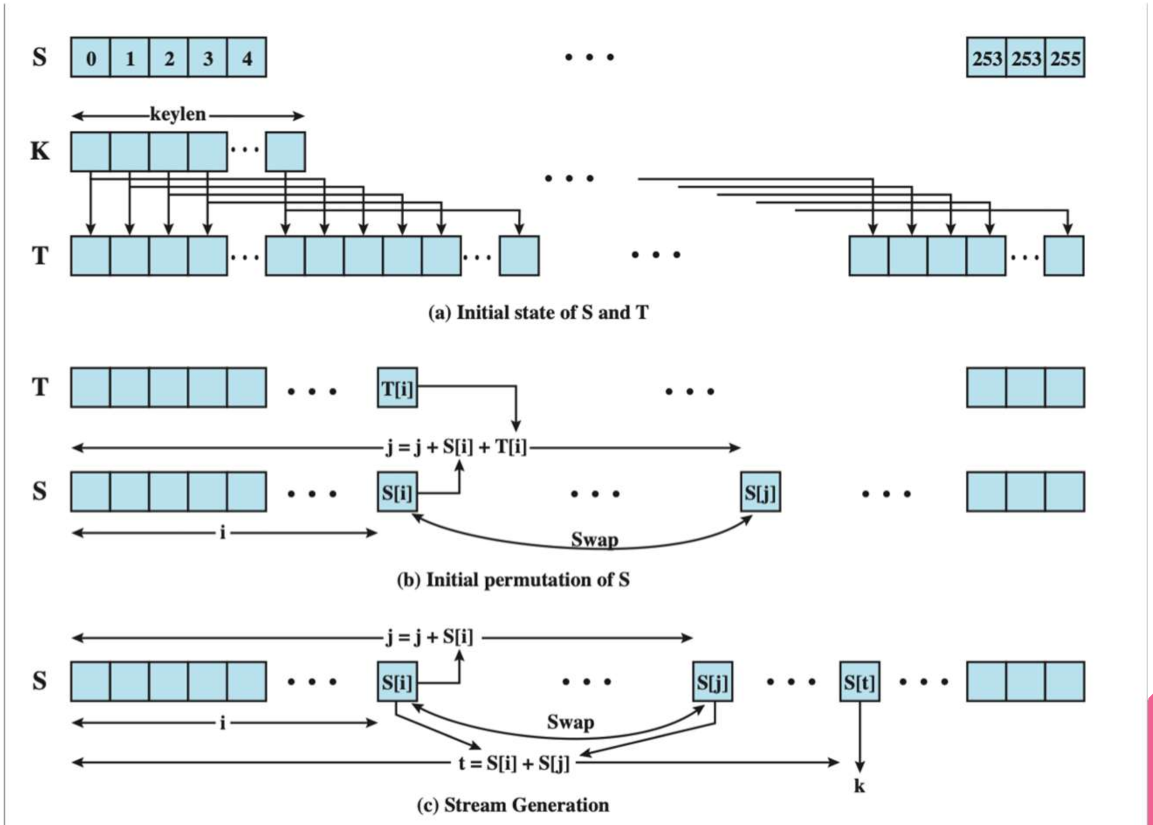
- Block Ciphers
  - Processes the input one block of elements at a time
  - Produces an output block for each input block
  - Can reuse keys
  - More common
- Stream Ciphers
  - Processes the input elements continuously
  - Produces output one element at a time
  - Primary advantage is that they are almost always faster and use far less code
  - Encrypts plaintext one byte at a time
  - Pseudorandom stream is one that is unpredictable without knowledge of the input key

Block and Stream Ciphers, ctd.

Cipher	Key Length	Speed (Mbps)*
DES	56	21
3-DES	168	10
AES	128	61
RC4	Variable	113

*Speed compared on an Intel Pentium 4

Block and Stream Ciphers, ctd.



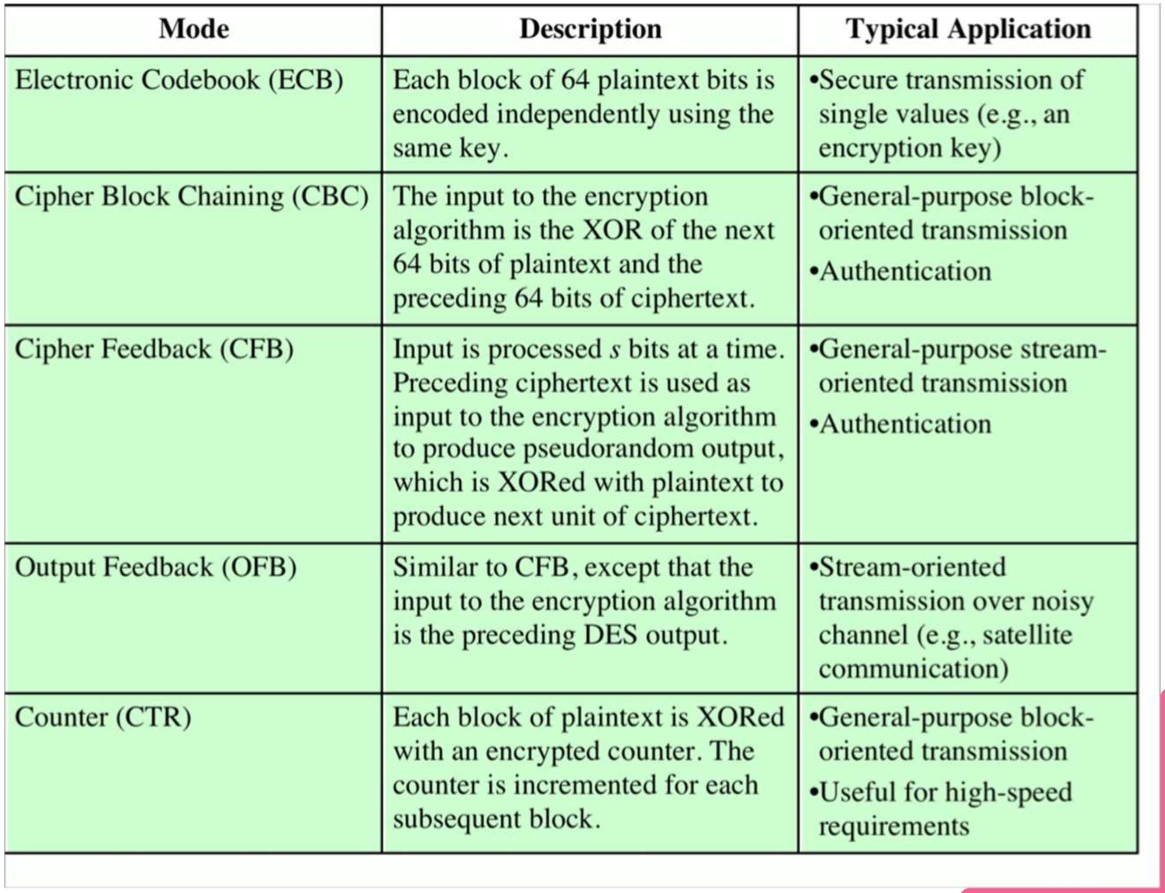
(a) Initial state of S and T

(b) Initial permutation of S

(c) Stream Generation

Figure 20.5 RC4

Block Cipher Modes of Operation



Mode	Description	Typical Application
Electronic Codebook (ECB)	Each block of 64 plaintext bits is encoded independently using the same key.	•Secure transmission of single values (e.g., an encryption key)
Cipher Block Chaining (CBC)	The input to the encryption algorithm is the XOR of the next 64 bits of plaintext and the preceding 64 bits of ciphertext.	•General-purpose block-oriented transmission •Authentication
Cipher Feedback (CFB)	Input is processed *s* bits at a time. Preceding ciphertext is used as input to the encryption algorithm to produce pseudorandom output, which is XORed with plaintext to produce next unit of ciphertext.	•General-purpose stream-oriented transmission •Authentication
Output Feedback (OFB)	Similar to CFB, except that the input to the encryption algorithm is the preceding DES output.	•Stream-oriented transmission over noisy channel (e.g., satellite communication)
Counter (CTR)	Each block of plaintext is XORed with an encrypted counter. The counter is incremented for each subsequent block.	•General-purpose block-oriented transmission •Useful for high-speed requirements

Symmetric Cipher’s Biggest Issue

We still haven't discussed the biggest issue regarding symmetric ciphers:

- **How do we exchange the key**?

Discuss.

Message Authentication

- Protects against active attacks
- Verifies that messages are authentic and not altered
- Can be used with conventional encryption

Message Authentication, ctd.

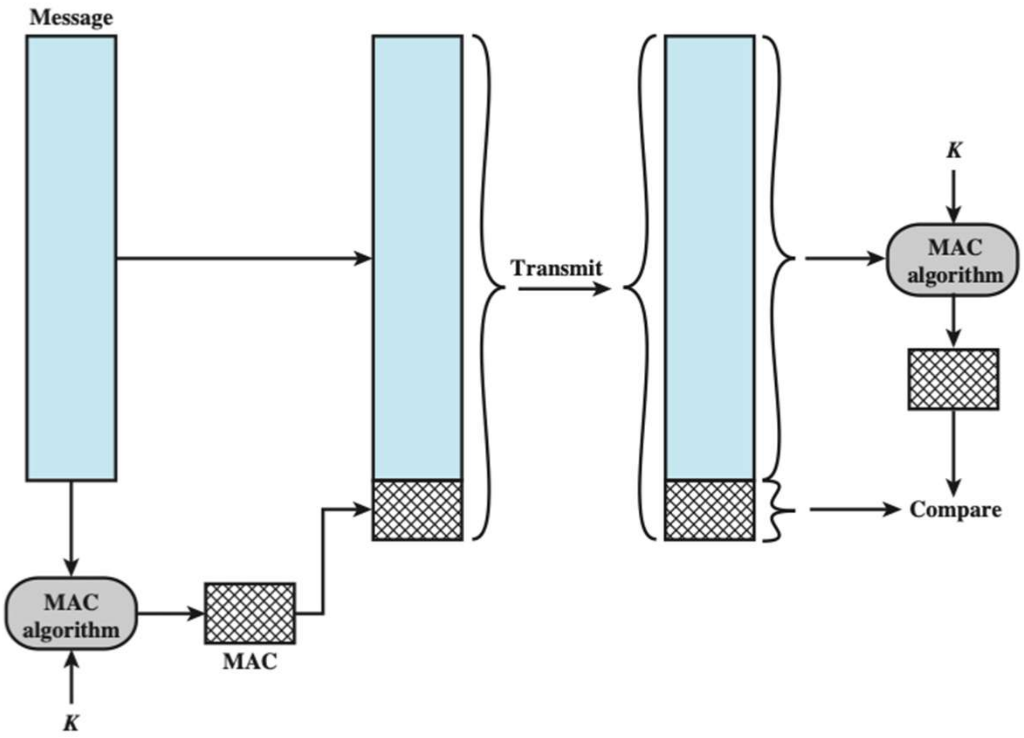
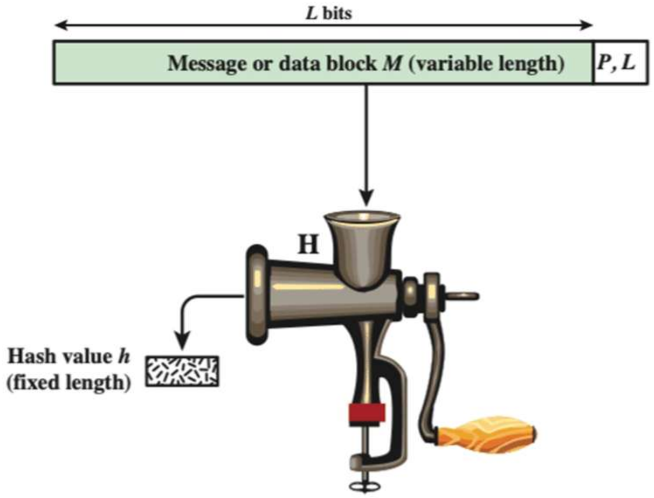


Figure 2.3 Message Authentication Using a Message Authentication Code (MAC).

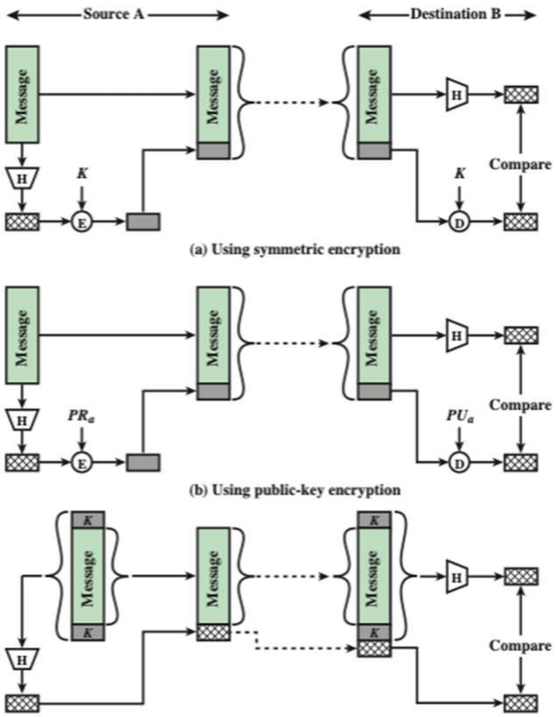
Cryptographic Hash Functions



P, *L* = padding plus length field

Figure 2.4 Cryptographic Hash Function; *h* = H(*M*)

Cryptographic Hash Functions, ctd.



(a) Using symmetric encryption

(b) Using public-key encryption

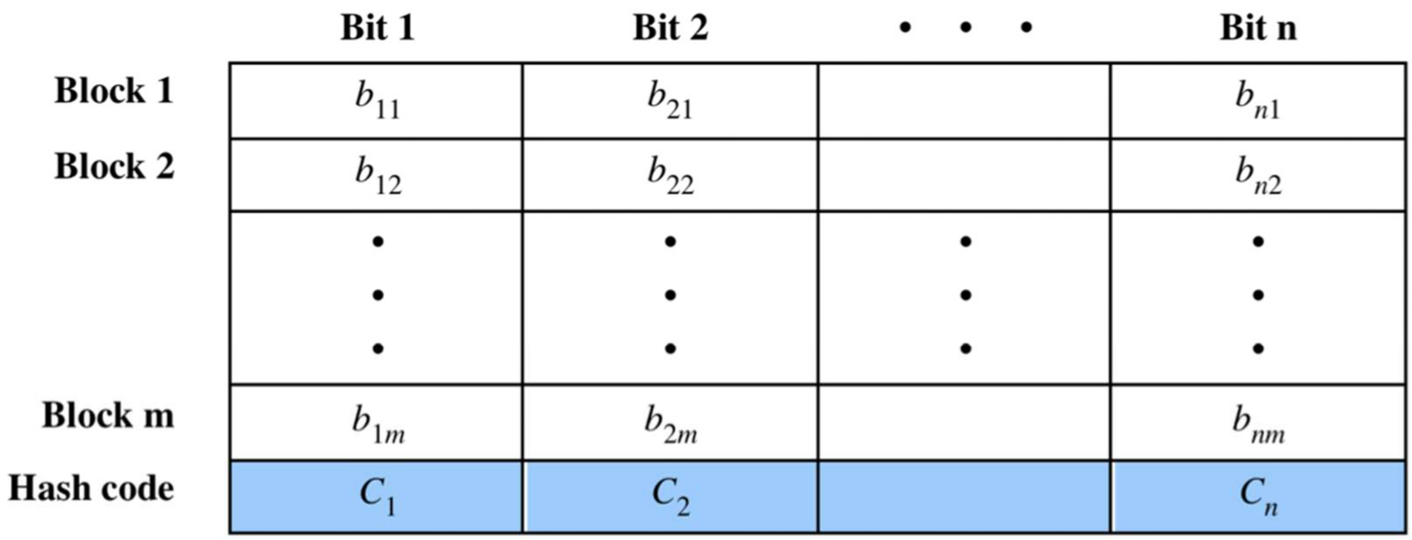
(c) Using secret value

Figure 2.5 Message Authentication Using a One-Way Hash Function.

Hash Function Requirements

- Can be applied to block data of any size
- Produces a fixed-length output
- H(x) is relatively easy to compute for any given x
- One way or pre-image resistant
    - Computationally infeasible to find x such that H(x) = h
- Computationally infeasible to find y!=x such that H(y)=H(x)
- Collision resistant or strong collision resistance
    - Computationally infeasible to find any pair (x,y) such that H(x)=H(y)

Simple One-Way 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 21.1 Simple Hash Function Using Bitwise XOR

Secure Hash Algorithm (SHA)

- Originally developed by NIST (the time people!) and published in 1993
- Quickly revised in 1995 as SHA-1 to introduce stronger hashing values (160-bit)
- Even further revised in 2002:
    - Adds 3 additional versions of SHA
    - SHA-256, SHA-384, SHA-512
    - With 256/384/512-bit hash values
    - Same basic structure as SHA-1 but greater security
- Older versions phased out by 2010 (but are still seen in use even today)

Secure Hash Algorithm, ctd.

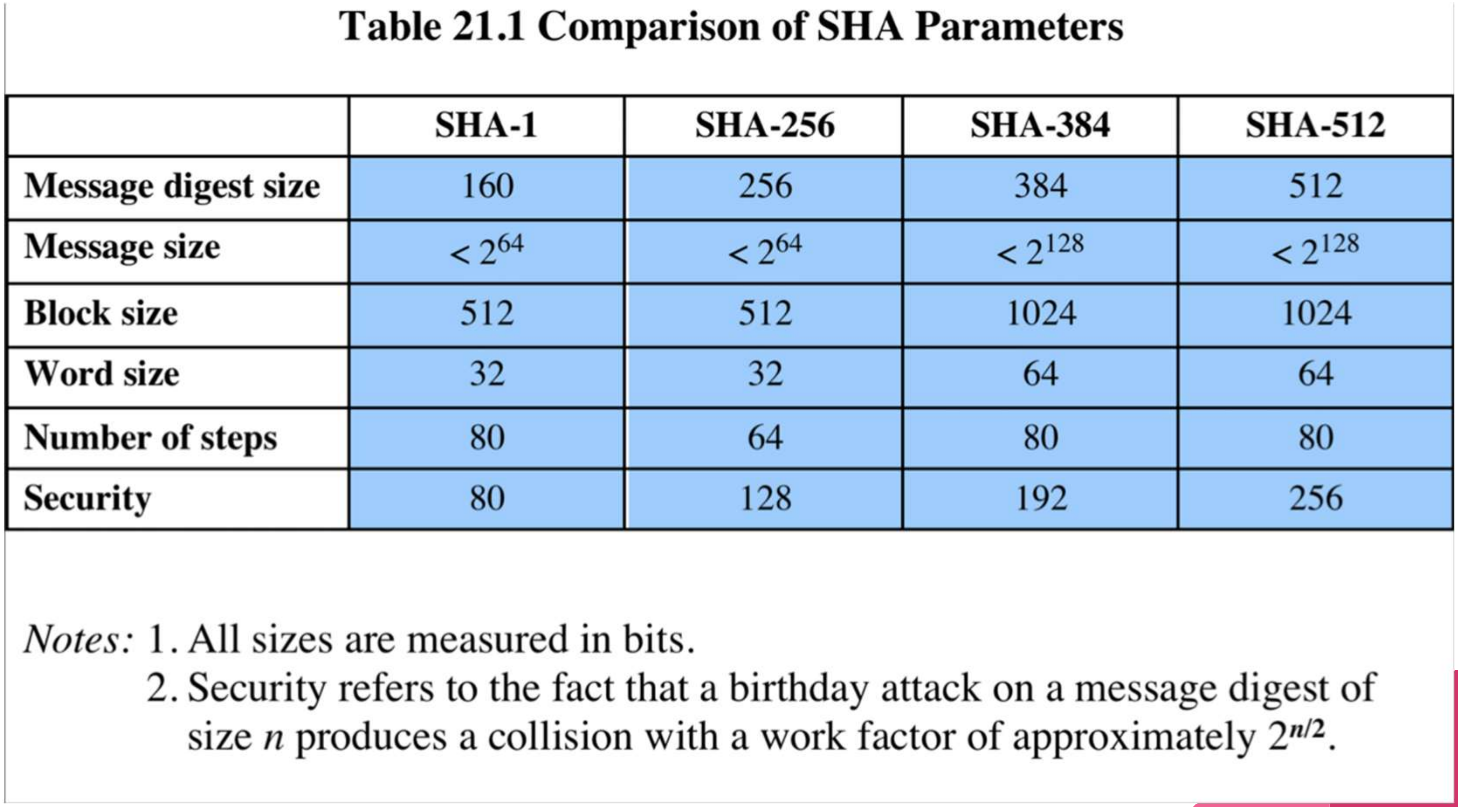


Table 21.1 Comparison of SHA Parameters

	SHA-1	SHA-256	SHA-384	SHA-512
Message digest size	160	256	384	512
Message size	$< 2^{64}$	$< 2^{64}$	$< 2^{128}$	$< 2^{128}$
Block size	512	512	1024	1024
Word size	32	32	64	64
Number of steps	80	64	80	80
Security	80	128	192	256

Notes: 1. All sizes are measured in bits.

2. Security refers to the fact that a birthday attack on a message digest of size n produces a collision with a work factor of approximately $2^{n/2}$.

Secure Hash Algorithm, ctd.

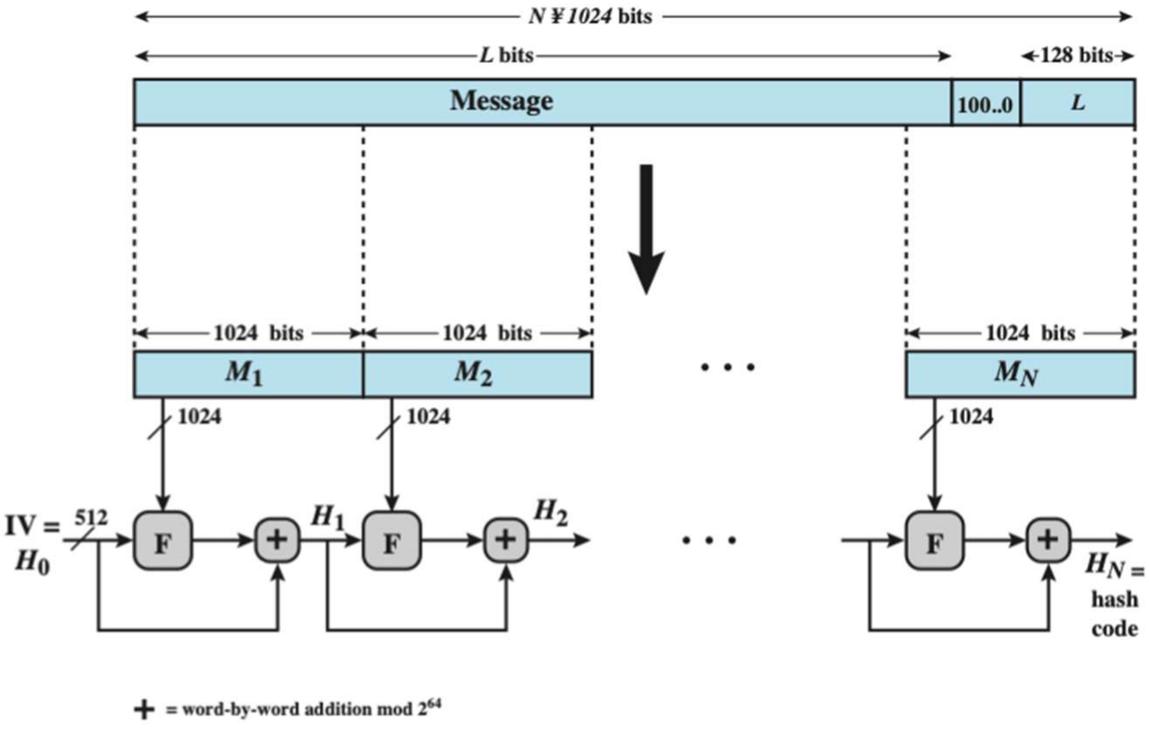


Figure 21.2 Message Digest Generation Using SHA-512

Secure Hash Algorithm, ctd.

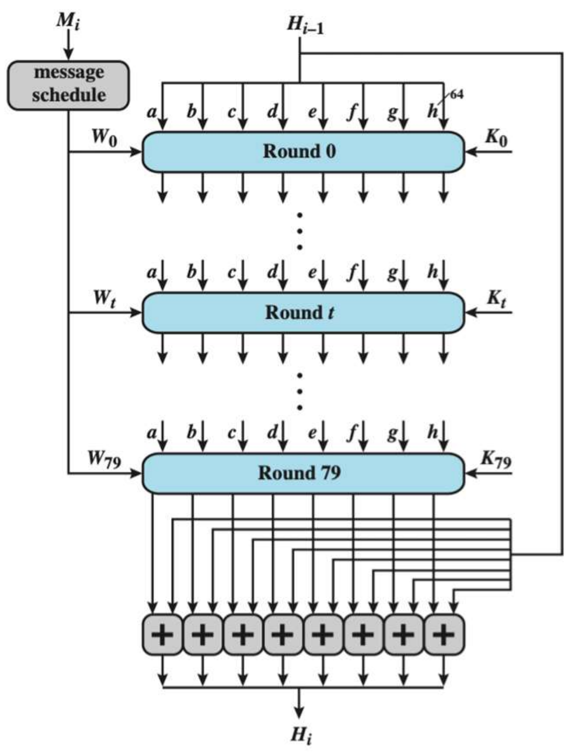


Figure 21.3 SHA-512 Processing of a Single 1024-Bit Block

Hash-Based Message Authentication Code (HMAC)

- Interest in developing a MAC derived from a cryptographic hash code
    - Cryptographic hash functions generally execute faster
    - Library code is widely available
    - SHA-1 was not designed for use as a MAC because it does not rely on a secret key
- Has been chosen as the mandatory-to-implement MAC for IP security
    - Used in other Internet protocols such as Transport Layer Security (TLS) and Secure Electronic Transaction (SET)

HMAC, ctd.

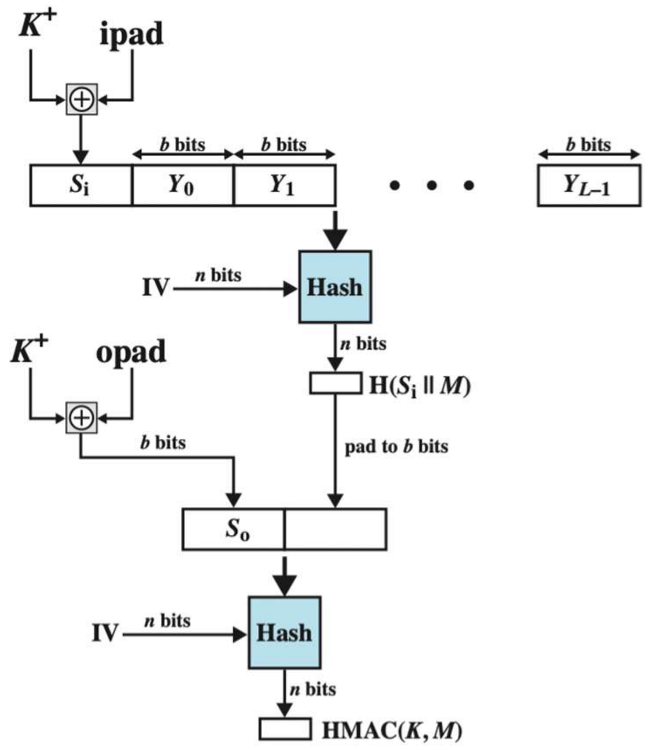


Figure 21.4 HMAC Structure

HMAC, ctd.

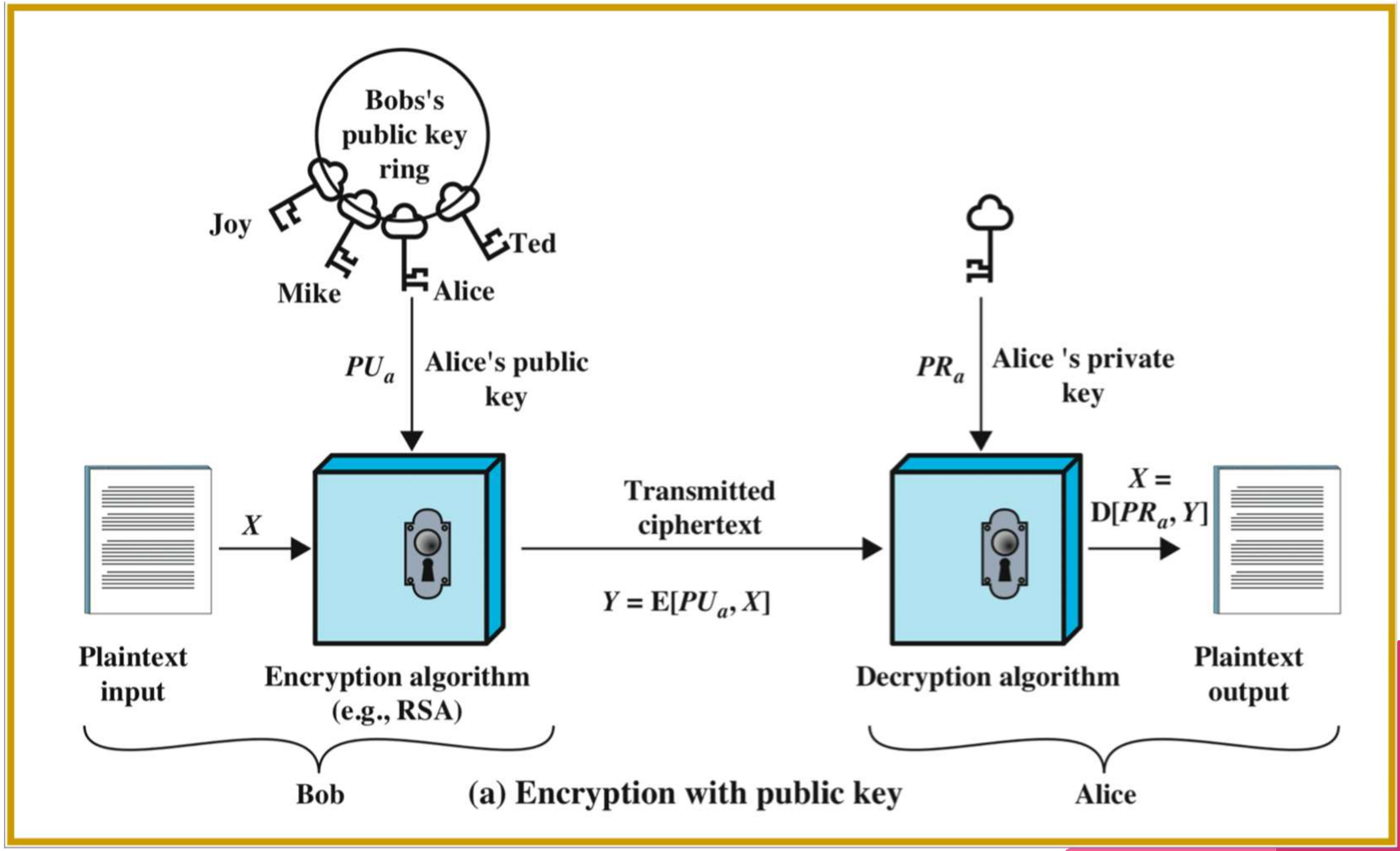
- Security depends on the cryptographic strength of the underlying hash function
- For a given level of effort on messages generated by a legitimate user and seen by the attacker, the probability of successful attack on HMAC is equivalent to one of the following attacks on the embedded hash function:
  - Either attacker computes output even with random secret IV
    - Brute force key O((2^n)) or use birthday attack
  - Or attacker finds collisions in hash function even when IV is random and secret
    - ie. find M and M' such that H(M)=H(M')
    - Birthday attack O((2^n)/2)
    - MD5 secure in HMAC since only observe

Public-Key Encryption

Sometimes called **Public/Private-Key Encryption**, but ***never*** Private-Key Encryption

- Introduced by Whitfield Diffie and Martin Hellman in 1976
- Based on purely mathematical functions
- Distinguishing Property: Asymmetry
  - Uses two separate keys: public and private key pair
  - Public key is made public for anyone to see

Public-Key Encryption, ctd.



(a) Encryption with public key

Public-Key Encryption Compared

Algorithm	Digital Signature	Symm Key	Key Encrypt
RSA	Yes	Yes	Yes
Diffie-Hellman	No	Yes	Yes
DSS	Yes	No	No
Elliptic Curve	Yes	Yes	Yes

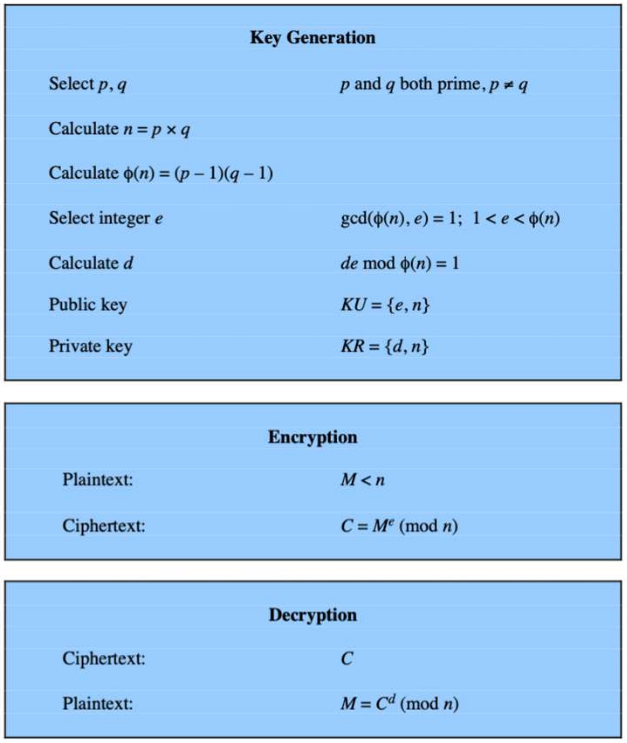
Requirements for Public-Key Cryptosystems

- Computationally easy to make keys
- Computationally easy for sender knowing public key to encrypt messages
- Computationally easy for receiver knowing private key to decrypt ciphertext
- Computationally infeasible for opponent to determine private key from public key
- Computationally infeasible for opponent to otherwise recover original message
- Useful if either key can be used for each role

RSA Public-Key Encryption

- By Rivest, Shamir & Adleman of MIT in 1977
- Best known and widely used public-key algorithm
- Uses exponentiation of integers modulo a prime
    - **Encrypt**: $C = M^e \bmod n$
    - **Decrypt**: $M = C^d \bmod n = (M^e)^d \bmod n = M$
- Both sender and receiver know values of n and e
- Only receiver knows value of d
- Public-key encryption algorithm with public key PU = $\{e,n\}$ and private key PR = $\{d,n\}$

RSA Public-Key Encryption, ctd.



Key Generation

Select p, q	p and q both prime, $p \neq q$
Calculate $n = p \times q$	
Calculate $\phi(n) = (p - 1)(q - 1)$	
Select integer e	$\gcd(\phi(n), e) = 1;\ 1 < e < \phi(n)$
Calculate d	$de \bmod \phi(n) = 1$
Public key	$KU = \{e, n\}$
Private key	$KR = \{d, n\}$

Encryption

Plaintext:	$M < n$
Ciphertext:	$C = M^e \pmod{n}$

Decryption

Ciphertext:	C
Plaintext:	$M = C^d \pmod{n}$

Figure 21.5 The RSA Algorithm

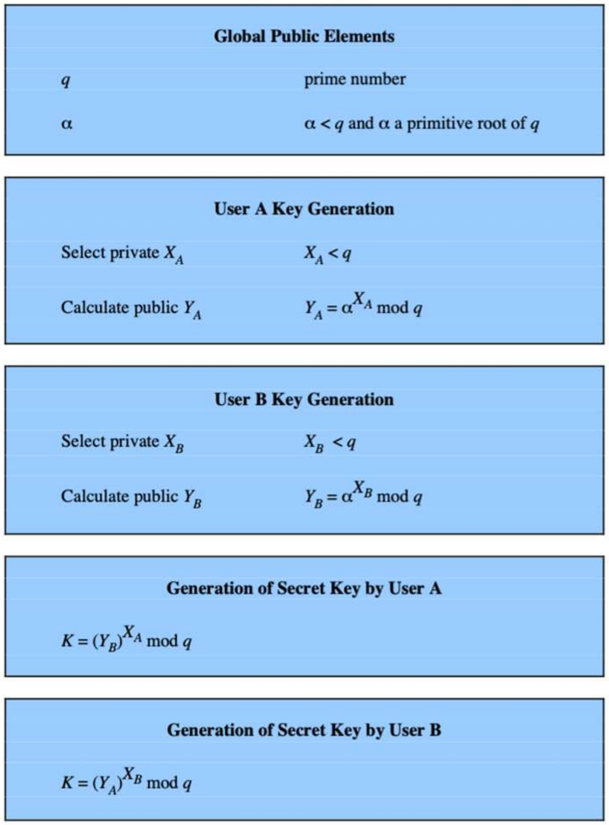
RSA Public-Key Encryption, ctd.

RSA Demo…

Diffie-Hellman Key Exchange

- First published public-key algorithm
- By Diffie and Hellman in 1976 along with the exposition of public key concepts
- Used in a number of commercial products
- Practical method to exchange a secret key securely that can then be used for subsequent encryption of messages
- Security relies on difficulty of computing discrete logarithms

Diffie-Hellman Key Exchange, ctd.



Global Public Elements

q	prime number
α	$\alpha < q$ and α a primitive root of q

User A Key Generation

Select private X_A	$X_A < q$
Calculate public Y_A	$Y_A = \alpha^{X_A} \bmod q$

User B Key Generation

Select private X_B	$X_B < q$
Calculate public Y_B	$Y_B = \alpha^{X_B} \bmod q$

Generation of Secret Key by User A

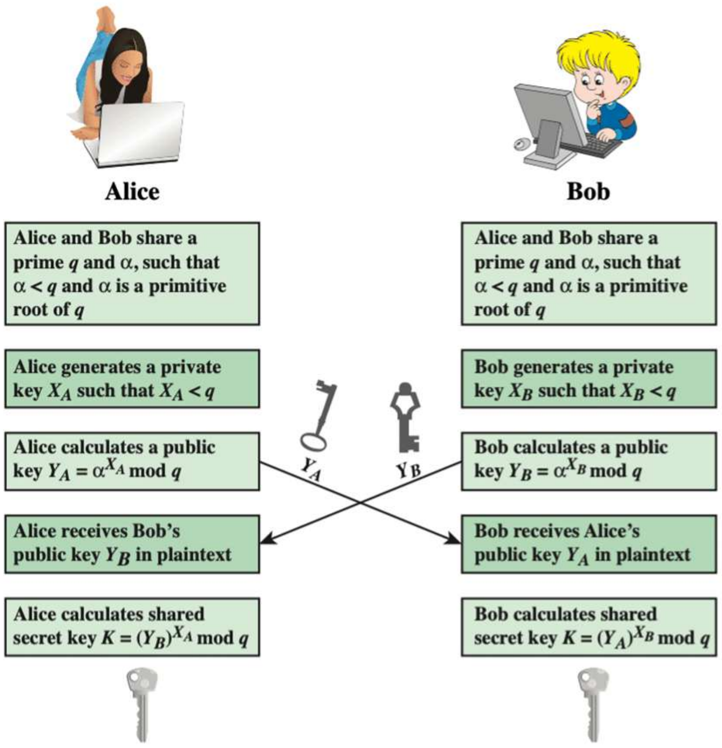
$K = (Y_B)^{X_A} \bmod q$

Generation of Secret Key by User B

$K = (Y_A)^{X_B} \bmod q$

Figure 21.7 The Diffie-Hellman Key Exchange Algorithm

Diffie-Hellman Key Exchange, ctd.



Alice and Bob share a prime q and α, such that $\alpha < q$ and α is a primitive root of q

Alice generates a private key X_A such that $X_A < q$

Alice calculates a public key $Y_A = \alpha^{X_A} \bmod q$

Alice receives Bob's public key Y_B in plaintext

Alice calculates shared secret key $K = (Y_B)^{X_A} \bmod q$

Alice and Bob share a prime q and α, such that $\alpha < q$ and α is a primitive root of q

Bob generates a private key X_B such that $X_B < q$

Bob calculates a public key $Y_B = \alpha^{X_B} \bmod q$

Bob receives Alice's public key Y_A in plaintext

Bob calculates shared secret key $K = (Y_A)^{X_B} \bmod q$

Figure 21.8 Diffie-Hellman Key Exchange

Diffie-Hellman Key Exchange, ctd.

Diffie-Hellman Demo…

Man-in-the-Middle Attack

There is one type of attack that we have not discussed yet, but virtually all types of encryption are susceptible to this type of attack. That attack is the man-in-the-middle attack.

- How the attack is carried out:
  - Darth (Vader) generates private keys Subscript[X, D1] and Subscript[X, D2], and their public keys Subscript[Y, D1] and Subscript[Y, D2]
  - Alice transmits Subscript[Y, A] to Bob
  - Darth intercepts Subscript[Y, A] and transmits Subscript[Y, D1] to Bob. Darth also calculates Subscript[K, 2]
  - Bob receives Subscript[Y, D1] and calculates Subscript[K, 1]
  - Bob transmits Subscript[X, A] to Alice
  - Darth intercepts Subscript[X, A] and transmits Subscript[Y, D2] to Alice. Darth calculates Subscript[K, 1]
  - Alice receives Subscript[Y, D2] and calculates Subscript[K, 2]
- All subsequent communications are now compromised

Digital Signatures

- Used for authenticating both source and data integrity
- Created by encrypting hash code with private key
- Does not provide confidentiality
  - Even in the case of complete encryption
  - Message is safe from alteration but not eavesdropping

Digital Envelopes

- Protects a message without needing to first arrange for sender and receiver to have the same secret key
- Equates to the same thing as a sealed envelope containing an unsigned letter

Digital Envelopes, ctd.

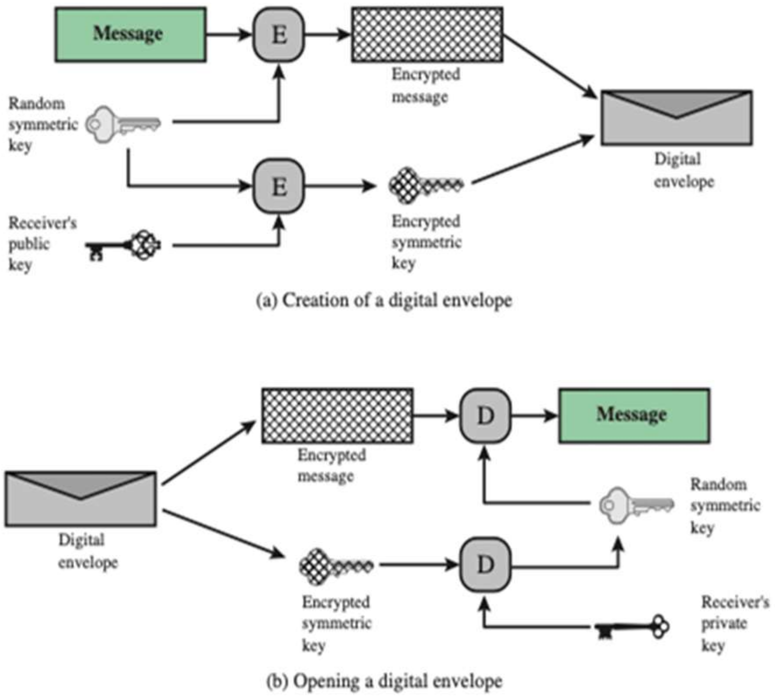


Figure 2.8 Digital Envelopes

Random Numbers

Having strongly random numbers is extremely important for having good encryption, but this is not something that computers can do well. Instead, we are forced to have *pseudorandom* numbers

- Criteria:
    - Uniform Distribution
        - Frequency of individual number's occurrence should be approximately the same
    - Independence
        - No one value from the sequence can be inferred from another

Random Numbers

- Pseudorandom numbers should be unpredictable
    - Each number is statistically independent of other numbers in the sequence
    - Opponent should not be able to predict future elements of the sequence on the basis of earlier elements
- Predictability is the enemy of good encryption

Random Numbers, ctd.

Why can computers only create pseudorandom numbers?

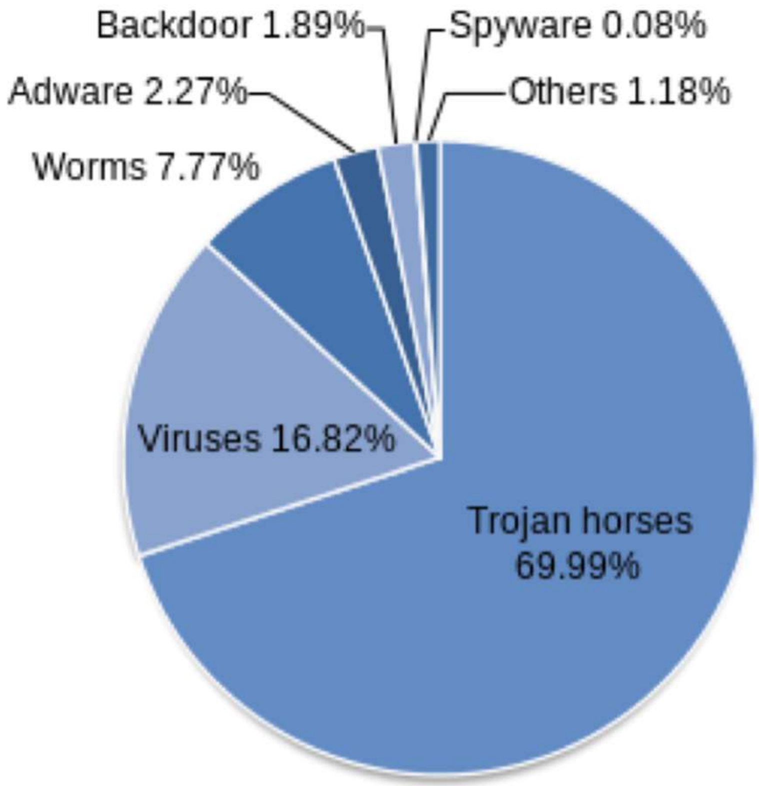
Malicious Software

Lecture Set 3

What is Malware?

Simply put, malware (short for malicious software) is any software used to disrupt computer or mobile operations, gather sensitive information, gain access to private computer systems, or display unwanted advertising.

Malware Statistics



Malware by categories March 16, 2011

Malware History

- Conceptualized by John von Neumann
  - published paper in 1949 titled "Theory of self-reproducing automata"
  - provided sample pseudo-code to demonstrate theory
- Recognized as the first computer virus
- Malware has not fundamentally changed since his theory and von Neumann's work has heavily influenced malware authors to this day

Classification of Malware

- Two broad categories:
  - how it spreads
  - actions or payloads
- Additionally, we should consider:
  - host-based (parasitic code, like viruses)
  - independent/self-contained (worms, trojans, bots)
  - replicating (viruses and worms)
  - non-replicating (trojans, spam, backdoors)

Advanced Persistent Threat Attacks (APT)

Industry term to describe malware that is targeted against a specific organization

- Many different techniques can be used to deploy an APT:
    - social engineering (very common)
    - spear-phishing
    - drive-by-downloads

Many APTs include payloads that can infect a system with backdoors, rootkits, destructive payloads, and much more. It's completely up to the imagination of the attacker.

Viruses

Viruses are software programs that infect other programs.

- modifies target programs to include a copy of the virus
- replicate themselves and continue on to infect other programs
- typically can spread easily through LAN networks and connected devices

When the virus attaches itself to a program, it gets all of the same permissions as the original program with the user executor's permissions.

On the plus side, viruses tend to affect specific operating systems and specific architectures only.

Virus Components

Viruses have three main components:

- Infection mechanism (or infection vector)
  - how the virus spreads
- Trigger
  - some event that causes the virus to activate
- Payload
  - what the virus does (besides spreading)
  - can be harmful or benign

Case Study: Creeper

- Appeared in 1971
    - first reported virus in the wild and spread over ARPANET
- Affected the DEC PDP-10 computer which ran the TENEX (TOPS-20) operating system
- Virus displayed "*I'm the creeper, catch me if you can!*" on infected systems
- Program called *Reaper* created to eliminate *Creeper*

Case Study: Elk Cloner

- Designed in 1982 by Richard Skrenta as a prank for his friends
- First example of a boot-sector virus
    - virus infected floppy disks and would execute on the 50th run of a disk and would show a poem
    - overwrote boot sector of floppy disks, but otherwise harmless to computer and disks could be repaired

Case Study: Brain Virus

- Created in 1986 for IBM PC compatibles
    - first PC virus and first boot sector virus that was released by a commercial distributor
- Intended to protect the authors' software from piracy, but ended up infecting computers which did not copy software
- Authors' gave contact info in virus message for "*vaccination*"
- Notable for creating the first effective boot sector virus that is easily modifiable

Case Study: Stoned Virus

- Shortly followed Brain Virus as a harmful variant
- Possibly one of the most copied viruses of all time
- Simple premise:
  - while *Brain virus* intentionally did not infect hard disks by first checking the boot sector bit, *Stoned* (and variants) eliminated this check
  - all disks had potential to be infected–even MBR
- Only way to eliminate Stoned was to overwrite modified sectors
- Later variants of Stoned had additional harmful payloads

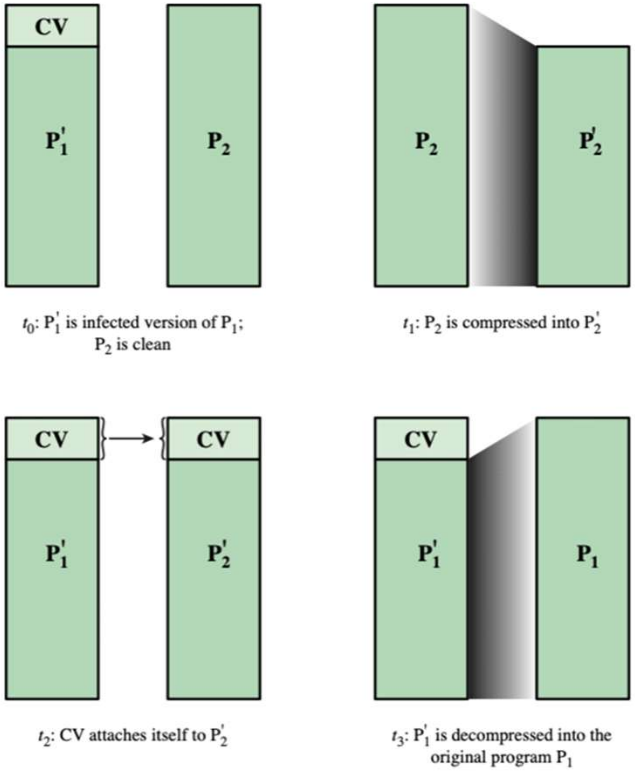
Case Study: Michelangelo Virus

- Technically a variant of *Stoned*
  - Notable for the hysteria that it caused in mass media
  - Found in the wild in 1991
- First malware that made general public aware of threat ("*Michelangelo Madness*")
- Early example of a time-bomb virus
  - Payload would wipe disks attached to the computer on Michelangelo's birthday every calendar year (March 6)

Virus Phases

- Dormant Phase
  - virus is idle and waiting for target
  - not all viruses have a dormant phase
- Triggering Phase
  - virus is activated and can be caused by various number of events
- Propagation Phase
  - virus spreads itself before carrying out payload
- Execution Phase
  - the payload is delivered

Compression Viruses



t_0: P'_1 is infected version of P_1; P_2 is clean

t_1: P_2 is compressed into P'_2

t_2: CV attaches itself to P'_2

t_3: P'_1 is decompressed into the original program P_1

Figure 6.2 A Compression Virus

Macro and Scripting Viruses

- Very common in mid-1990s
    - platform independent
    - infect documents (not executable portions of code)
    - spread easily
- Exploit macro capability of MS Office applications
    - modern anti-virus can typically handle these easily

Case Study: Love Letter Virus

- Probably closer to a worm than a virus
- Released into the wild on May 5, 2000
- Exploited auto-run vulnerability in .vbs scripting language
- Virus/worm propagated through email and relied on the recipient to open the email in MS Outlook
- Once the email was viewed, the *.vbs* file would execute without prompting the user and infect their machine along with replicating itself and emailing a copy of the malware to the user's contact list

Love Letter Source Code

The Love Letter Malware

Worms

- Actively seek out more machines to infect and each infected machine serves as an automated launching pad for additional attacks
- Exploits software vulnerabilities in client or server programs
- Can use network connections to spread from system to system
- Spreads through shared media (USB drives, CD, DVD data disks)
- Email worms spread in macro or script code included in attachments and instant messenger file transfers
- Upon activation the worm may replicate and propagate again
- First known implementation was done in Xerox Palo Alto Labs in the early 1980s

Worm Replication

- Email or instant messaging
- File sharing
- Remote execution
- Remote file transfer (ftp, sftp)
- Remote login (vnc, ssh)

Target Discovery

Called *scanning* or *fingerprinting* where a worm attempts to find other systems to infect

- Random
- Hit-list
- Topological
- Local subnet

Worm Technology

- Multiplatform
- Metamorphic
- Polymorphic
- Multiple exploits

Worm Countermeasures

- Considerable overlap in techniques for dealing with viruses and worms
- Once a worm is resident on a machine anti-virus software can be used to detect and possibly remove it
- Perimeter network activity and usage monitoring can form the basis of a worm defense
- Worm defense approaches include:
  - signature-based worm scan filtering
  - filter-based worm containment
  - payload-classification-based worm containment
  - threshold random walk (TRW) scan detection
  - rate limiting/halting

Case Study: Morris Worm

- First computer worm, created by Robert Morris in 1988. Also known as the "Great Internet Worm"
- Intended to determine the size of the Internet of the day by spreading across machines and probing their networks
- Designed to spread on UNIX systems
  - attempted to crack local password file to use login/password to log in to other systems
  - exploited a bug in the finger protocol which reports the whereabouts of a remote user
  - exploited a trapdoor in the debug option of the remote process that receives and sends mail

Case Study: Morris Worm, ctd.

- Coding error caused the worm to self-replicate multiple times on a single computer and crash the machine
- Caused an estimated $1 million in damages and lost computing time

Mobile Phone Worms

- First discovery was *Cabir* worm in 2004
- Then *Lasco* and CommWarrior in 2005
- Communicate through Bluetooth wireless connections or MMS
- Can completely disable the phone, delete data on the phone, or force the device to send costly messages
- *CommWarrior* replicates by means of Bluetooth to other phones, sends itself as an MMS file to contacts and as an auto reply to incoming text messages

Case Study: Cabir Worm

- Released to the wild in 2004
- First malware to specifically target mobile devices (*Symbian OS*)
- Spread by use of the Bluetooth protocol and discovered other devices on startup
- Users were prompted to accept a file
    - if file was rejected, phone would not become infected
- Only able to be removed by re-imaging the device, which was not possible by average users

Mobile Code

- Not to be confused with *mobile phone* code
- Programs that can be shipped unchanged to a variety of platforms
- Transmitted from a remote system to a local system and then executed on the local system
- Often acts as a mechanism for a virus, worm, or Trojan horse
- Takes advantage of vulnerabilities to perform its own exploits
- Popular vehicles include *Java applets*, *ActiveX*, *JavaScript*, *Python*, *Ruby*, and *VBScript*

Drive-By-Downloads

- Exploits browser vulnerabilities to download and installs malware on the system when the user views a Web page controlled by the attacker
- In most cases does not actively propagate
- Spreads when users visit the malicious Web page

Clickjacking

Also known as a user-interface (UI) redress attack. Using a similar technique, keystrokes can also be hijacked

- Users can be tricked into entering text/passwords into an invisible frame that is overlaying standard GUI input boxes
- Attackers can also use remapped UI buttons or controls in order to trick a user into clicking the wrong dialog box

Social Engineering

Using social norms and mores in order to convince a user to compromise their own systems or data

- Spam
  - yes, this is considered a form of social engineering
- Trojan horses
  - possibly the most common form of social engineering with malware

Stealthing

Backdoors

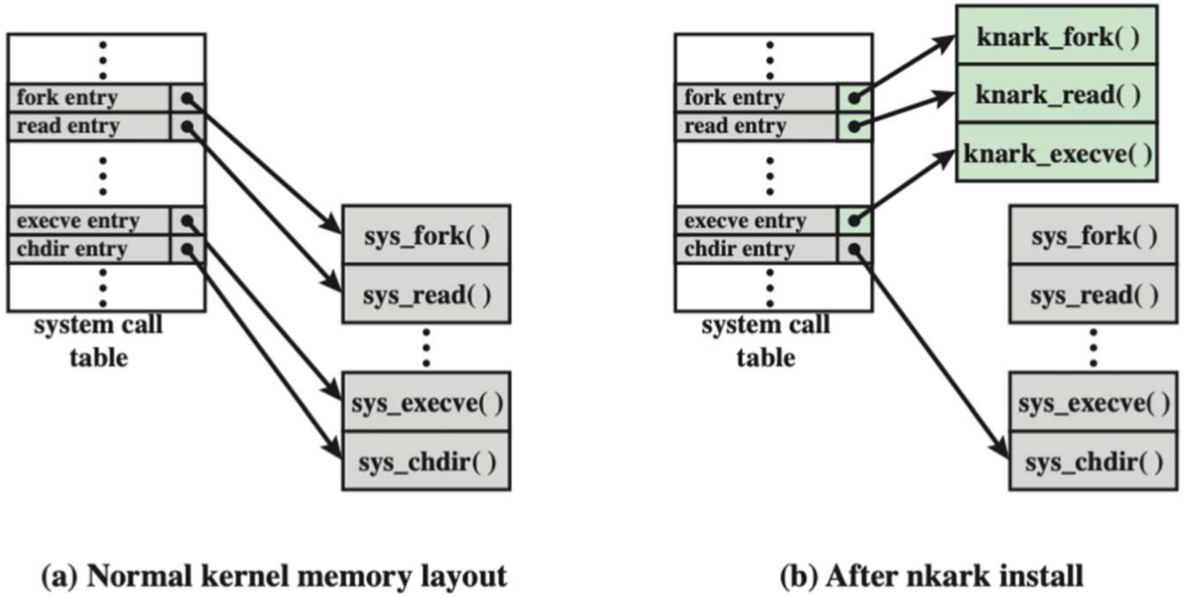
- Secret entry point into a program allowing the attacker to gain access and bypass the security access procedures
- Difficult to implement operating system controls for backdoors in applications
- Most "infamous" was Ken Thompson's login backdoor to UNIX
    - Reflections on Trusting Trust

Stealthing, ctd.

Rootkits

- A set of hidden programs installed on a system to maintain covert access to that system
  - not the same as a backdoor, but often used together
  - hides by subverting the mechanisms that monitor and report on the processes, files, and registries on a computer
  - gives administrator (or root) privileges to attacker

Stealting, ctd.



(a) Normal kernel memory layout

(b) After nkark install

Figure 6.4 System Call Table Modification by Rootkit

Malware Countermeasures

- Ideal solution is total prevention
  - How likely is this?
- Four main ways to prevent malware:
  - Policy
  - Awareness/user education
  - Vulnerability mitigation
  - Threat mitigation

Malware Countermeasures, ctd.

Inevitably, malware will eventually affect computing systems, and when that happens, there are only a few options left to consider:

- Detection
- Identification
- Removal

All three methods are typically employed to counteract malware

Host-Based Behavior Scanning

- Integrates with the operating system of a host computer and monitors program behavior in real time for malicious action
- Blocks potentially malicious actions before they have a chance to affect the system
- Blocks software in real time so it has an advantage over antivirus detection techniques such as fingerprinting or heuristics
- Because malicious code must run on the target machine before all its behaviors can be identified, it can cause harm before it has been detected and blocked

Perimeter-Based Scanning

Approach is to block malware from infecting computers on a network at scale

- Ingress monitors
  - protect systems from malware spreading *within* a LAN
- Egress monitors
  - protect systems from malware *entering* into a network from a WAN

Both types of monitors will do *nothing* to protect any individual machine