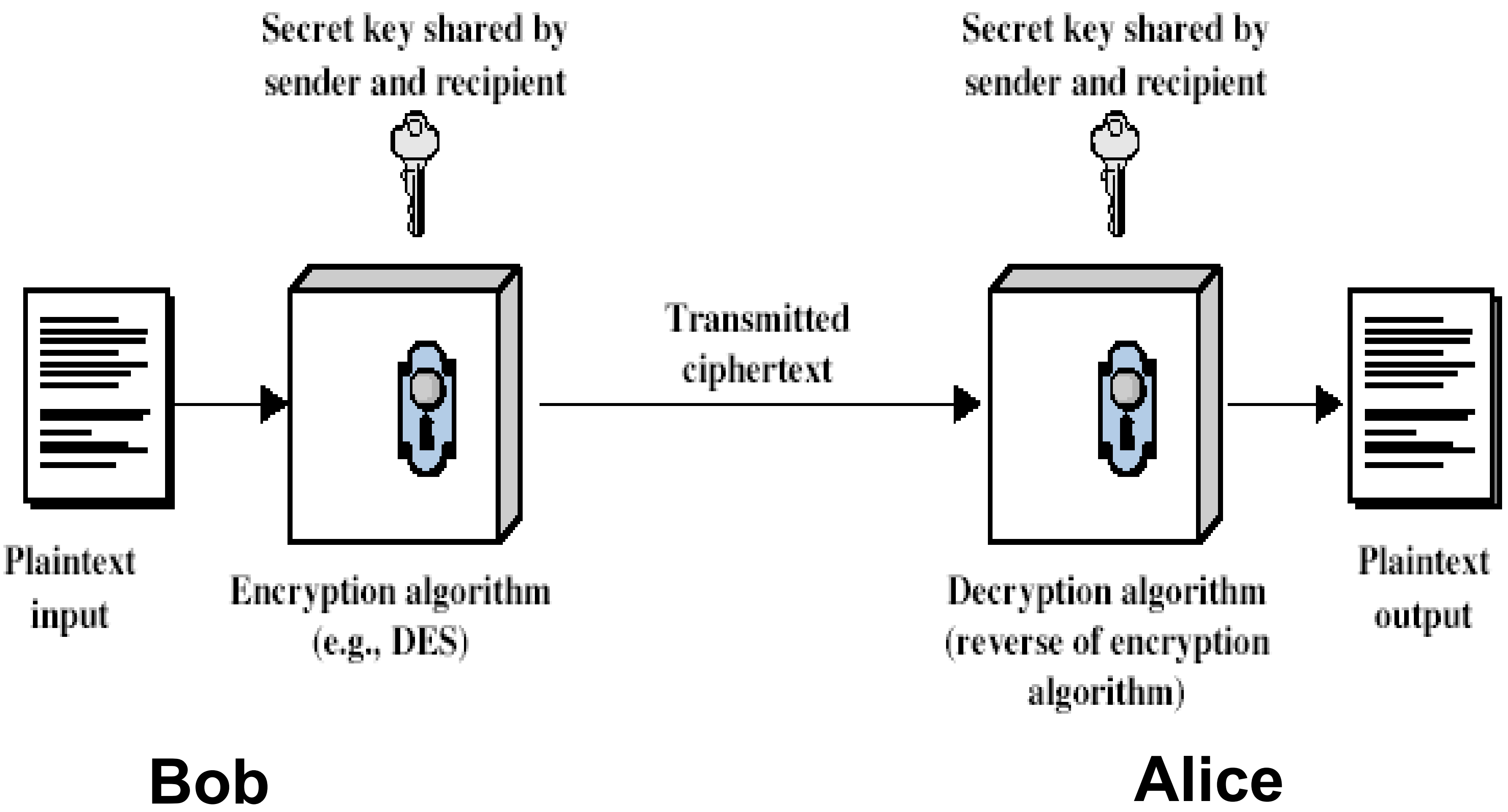
Information Security 05

Message authentication
and Hash function
Chapter 11

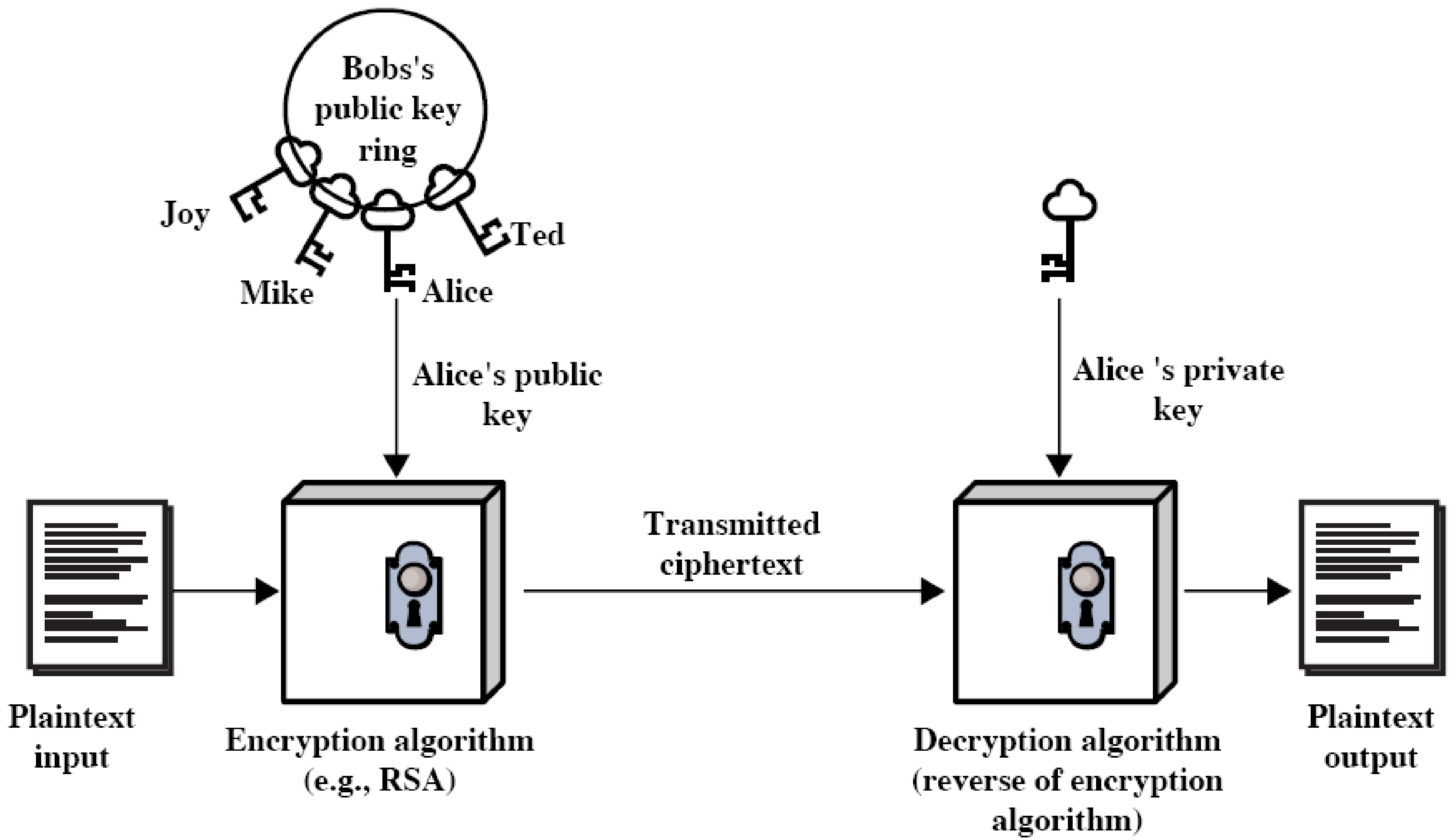
Review

- Symmetric Cryptography
- Asymmetric Cryptography

Review: Symmetric Model



Asymmetric Model



Bob **Alice**

Review

- Confidentiality

Review

- Confidentiality
- enough？

Security Requirements

- disclosure
- traffic analysis
- masquerade
- content modification
- sequence modification
- timing modification
- source repudiation
- destination repudiation

Message Authentication

- message authentication is 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 function

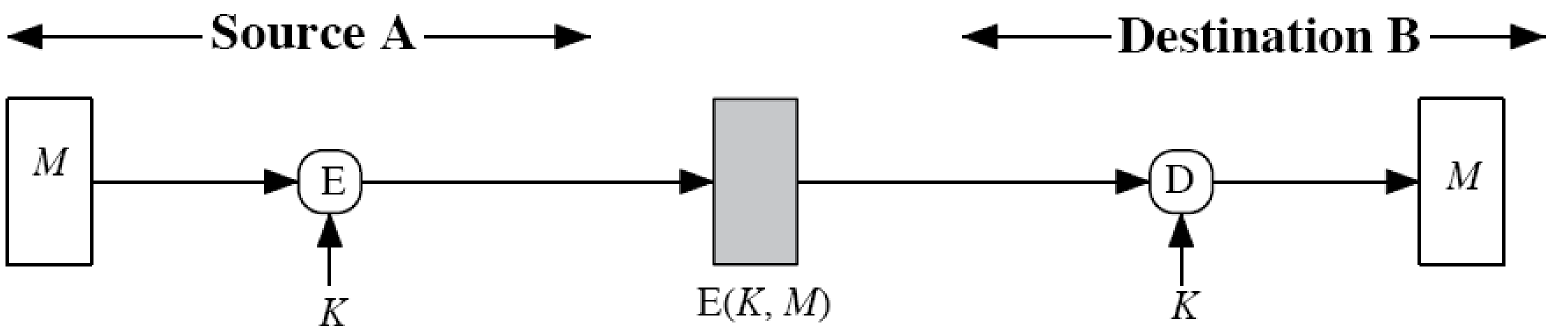
Note !!

- Message vs. Plaintext
- We will not consider Confidentiality sometimes.
- Authentication ?

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 now key used
  - know content cannot of been altered
  - if message has suitable structure, redundancy or a checksum to detect any changes

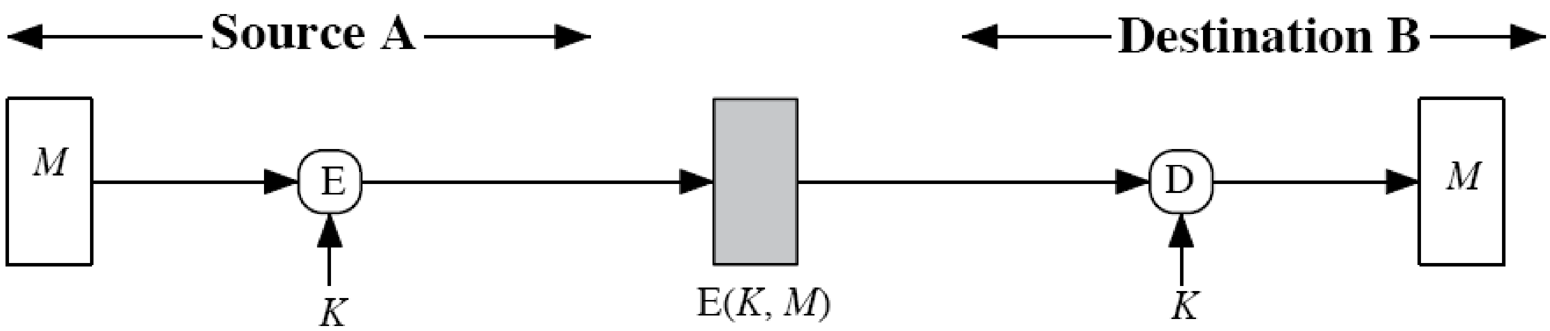
Message Encryption



(a) Symmetric encryption: confidentiality and authentication

- It may be difficult to determine automatically if incoming ciphertext decrypts to intelligible plaintext
- message should have suitable structure, redundancy or a checksum to detect any changes

Message Encryption



(a) Symmetric encryption: confidentiality and authentication

A → B: E(*K*, *M*)

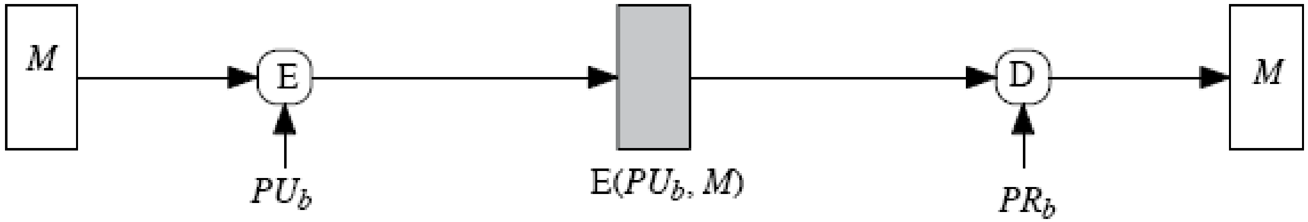
- Provides confidentiality
  - Only A and B share *K*
- Provides a degree of authentication
  - Could come only from A
  - Has not been altered in transit
  - Requires some formatting/redundancy
- Does not provide signature
  - Receiver could forge message
  - Sender could deny message

(a) Symmetric encryption

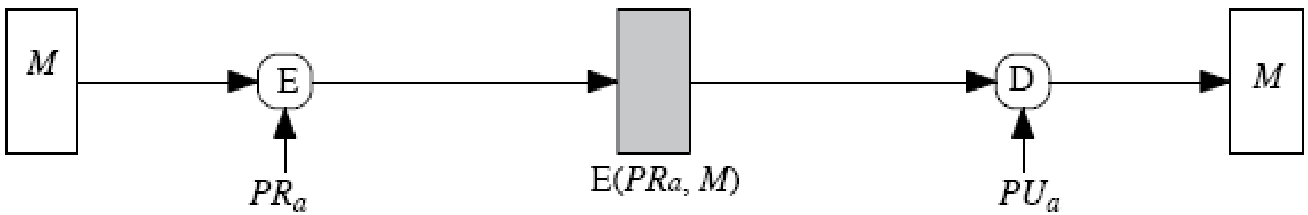
Message Encryption

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

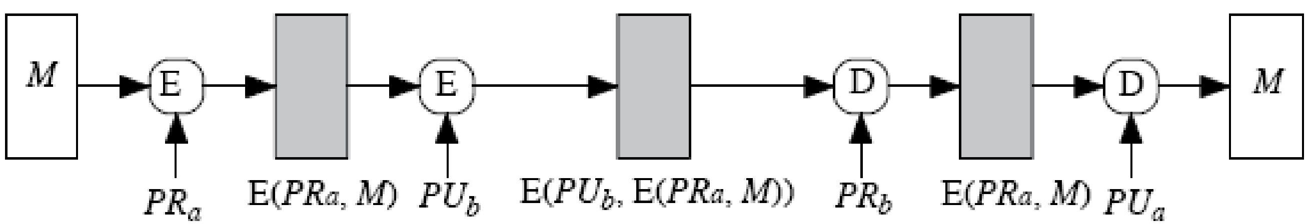
Message Encryption



(b) Public-key encryption: confidentiality



(c) Public-key encryption: authentication and signature



(d) Public-key encryption: confidentiality, authentication, and signature

Message Encryption

A → B: $E(PU_b, M)$

- Provides confidentiality
  - Only B has PR_b to decrypt
- Provides no authentication
  - Any party could use PU_b to encrypt message and claim to be A

(b) Public-key (asymmetric) encryption: confidentiality

A → B: $E(PR_a, M)$

- Provides authentication and signature
  - Only A has PR_a to encrypt
  - Has not been altered in transit
  - Requires some formatting/redundancy
  - Any party can use PU_a to verify signature

(c) Public-key encryption: authentication and signature

A → B: $E(PU_b, E(PR_a, M))$

- Provides confidentiality because of PU_b
- Provides authentication and signature because of PR_a

(d) Public-key encryption: confidentiality, authentication, and signature

Shortcomings of Enc…

- Q: shortcomings of message encryption?

Shortcomings of Enc…

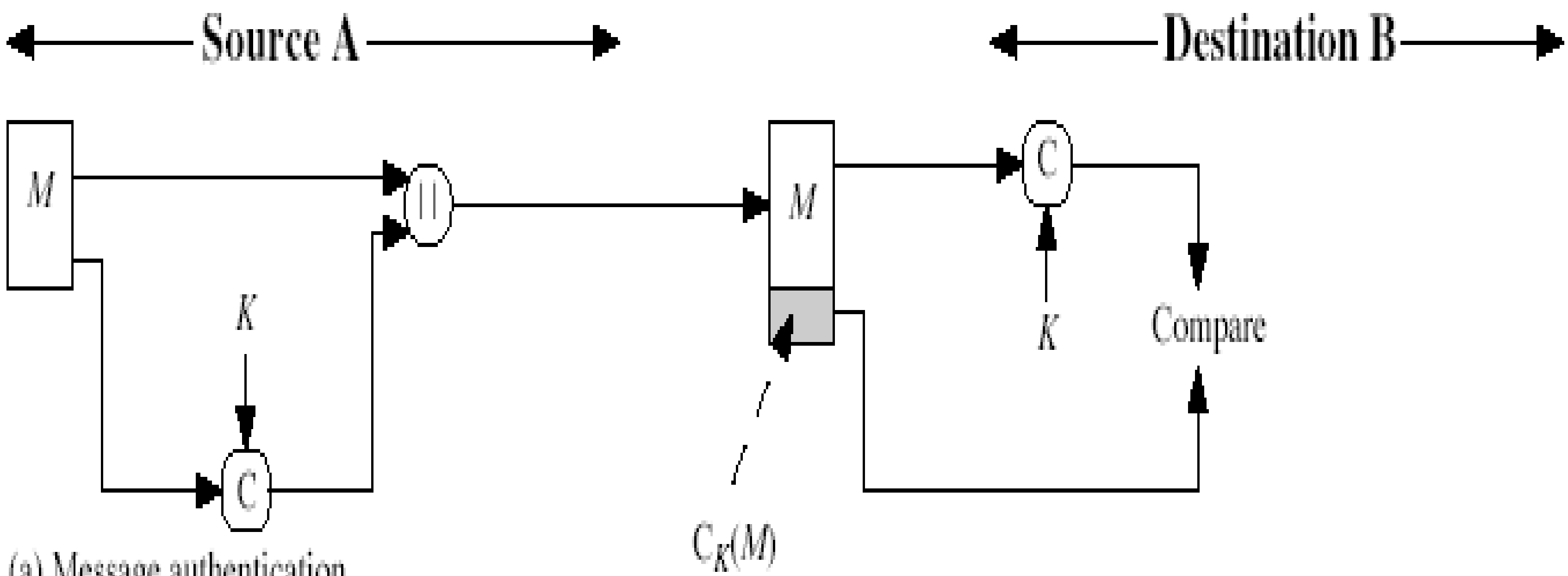
- Q: shortcomings of message encryption?

- Cost
  - encrypt the whole message
- Not automatically

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 an authenticator
- receiver performs same computation on message and checks it matches the MAC
- provides assurance that message is unaltered and comes from sender

Message Authentication Code



(a) Message authentication

Message Authentication Codes

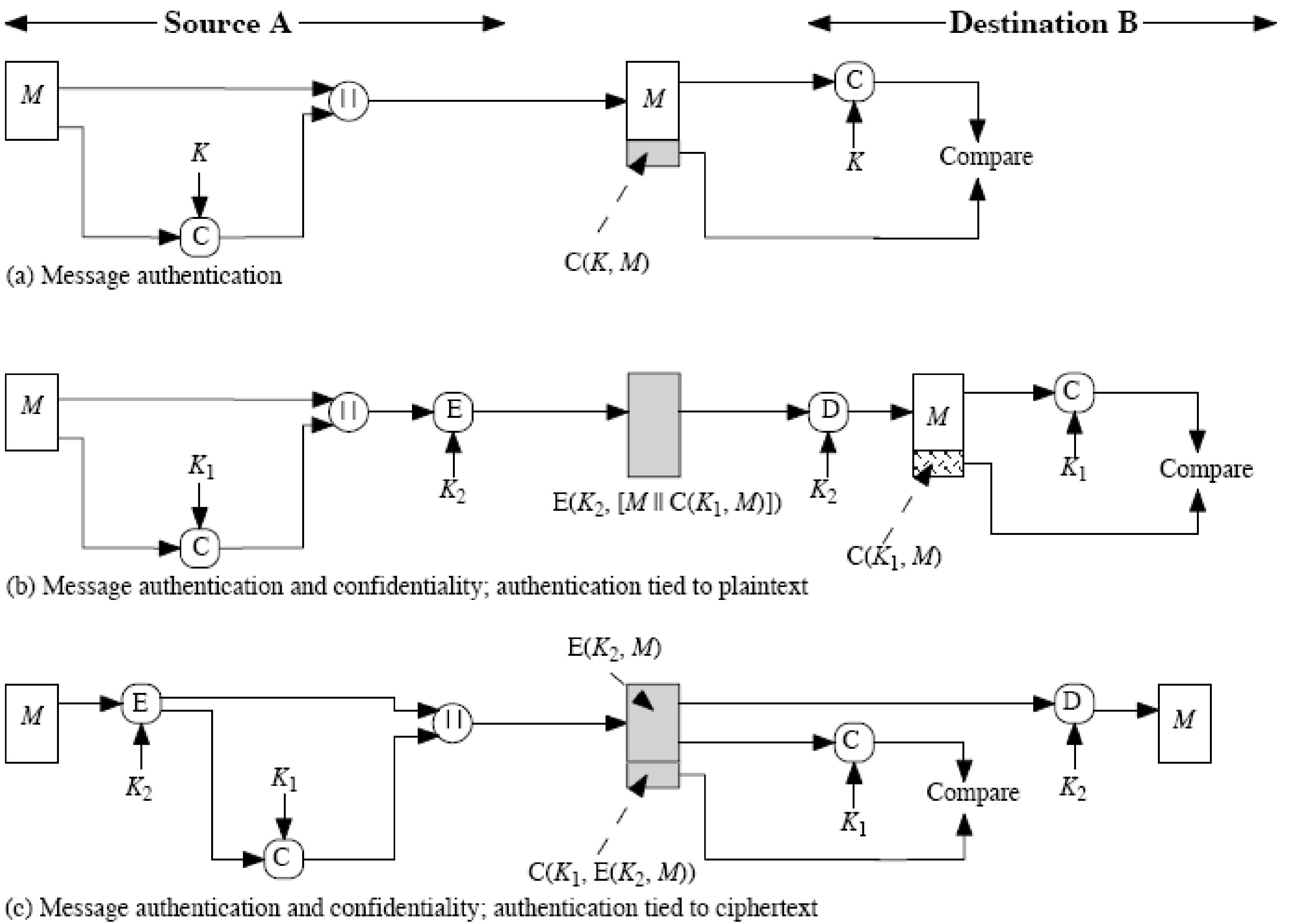
- as shown the MAC provides authentication

- why use a MAC?
  - sometimes only authentication is needed
  - sometimes need authentication to persist longer than the encryption (eg. archival use)
- note that a MAC is not a digital signature

Message Authentication Codes

- can also use encryption for confidentiality
  - generally use separate keys for each
  - can compute MAC either before or after encryption
  - is generally regarded as better done before

Message Authentication Codes



Message Authentication Codes

$A \rightarrow B$: $M \| C(K, M)$
- Provides authentication
  - Only A and B share K

(a) Message authentication

$A \rightarrow B$: $E(K_2, [M \| C(K, M)])$
- Provides authentication
  - Only A and B share K_1
- Provides confidentiality
  - Only A and B share K_2

(b) Message authentication and confidentiality: authentication tied to plaintext

$A \rightarrow B$: $E(K_2, M) \| C(K_1, E(K_2, M))$
- Provides authentication
  - Using K_1
- Provides confidentiality
  - Using K_2

(c) Message authentication and confidentiality: authentication tied to ciphertext

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

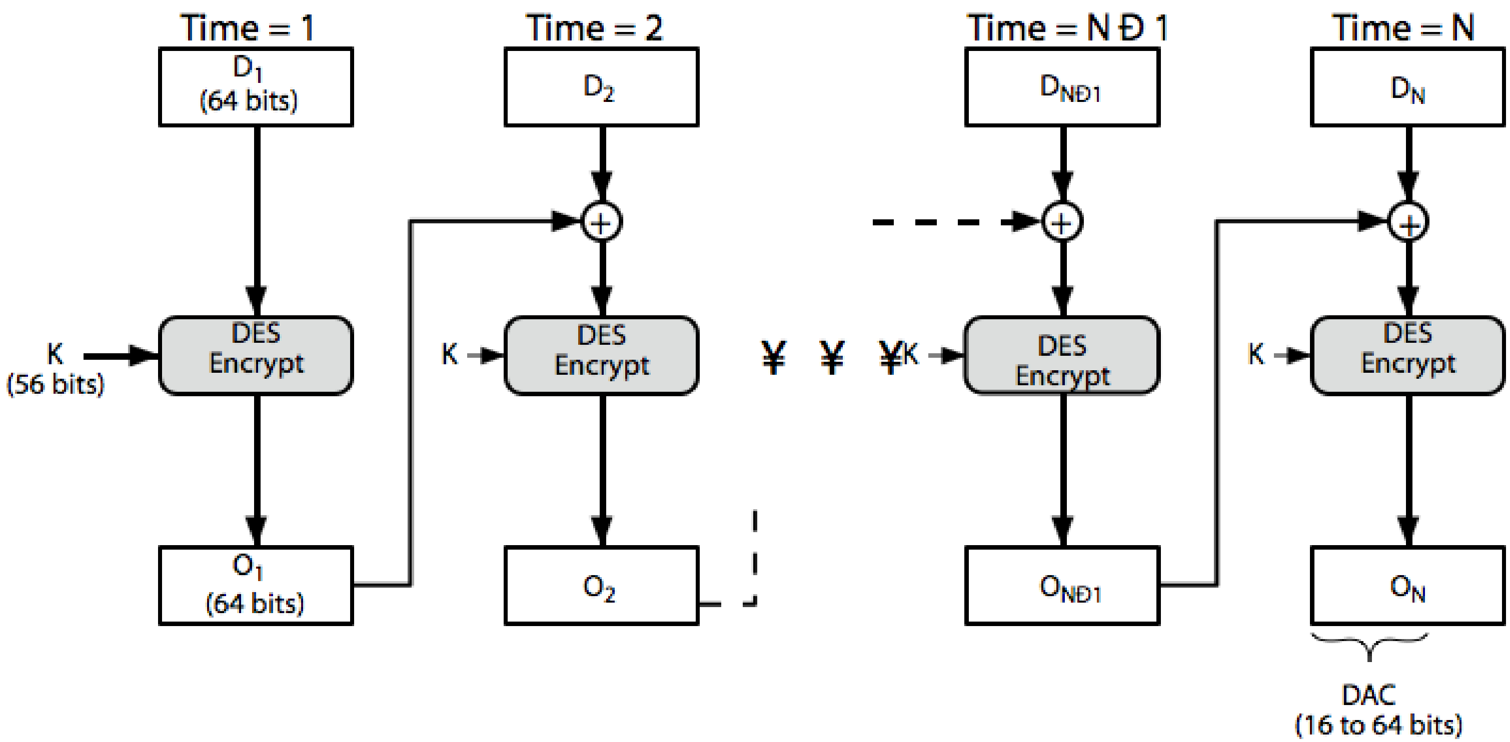
- taking into account the types of attacks
- need the MAC to satisfy the following:

1. knowing a message and MAC, 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

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 ($16 \leq M \leq 64$) of final block
- but final MAC is now too small for security

Data Authentication Algorithm



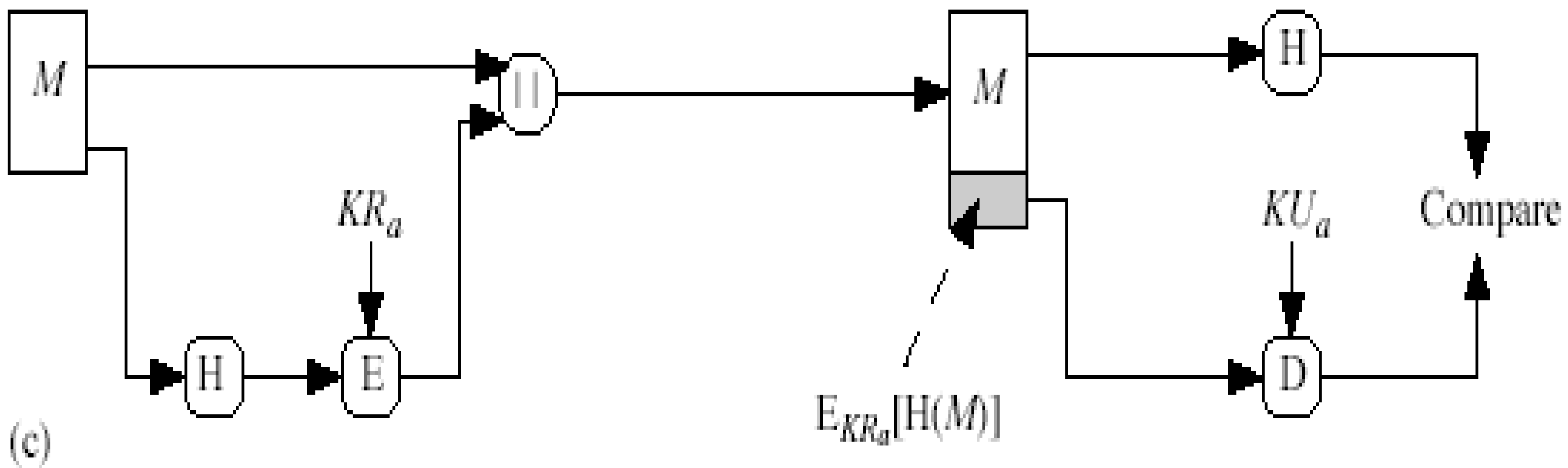
Hash Functions

- 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

Hash Functions & Digital Signatures



Hash Function Properties

- a Hash Function produces a fingerprint of some file/message/data

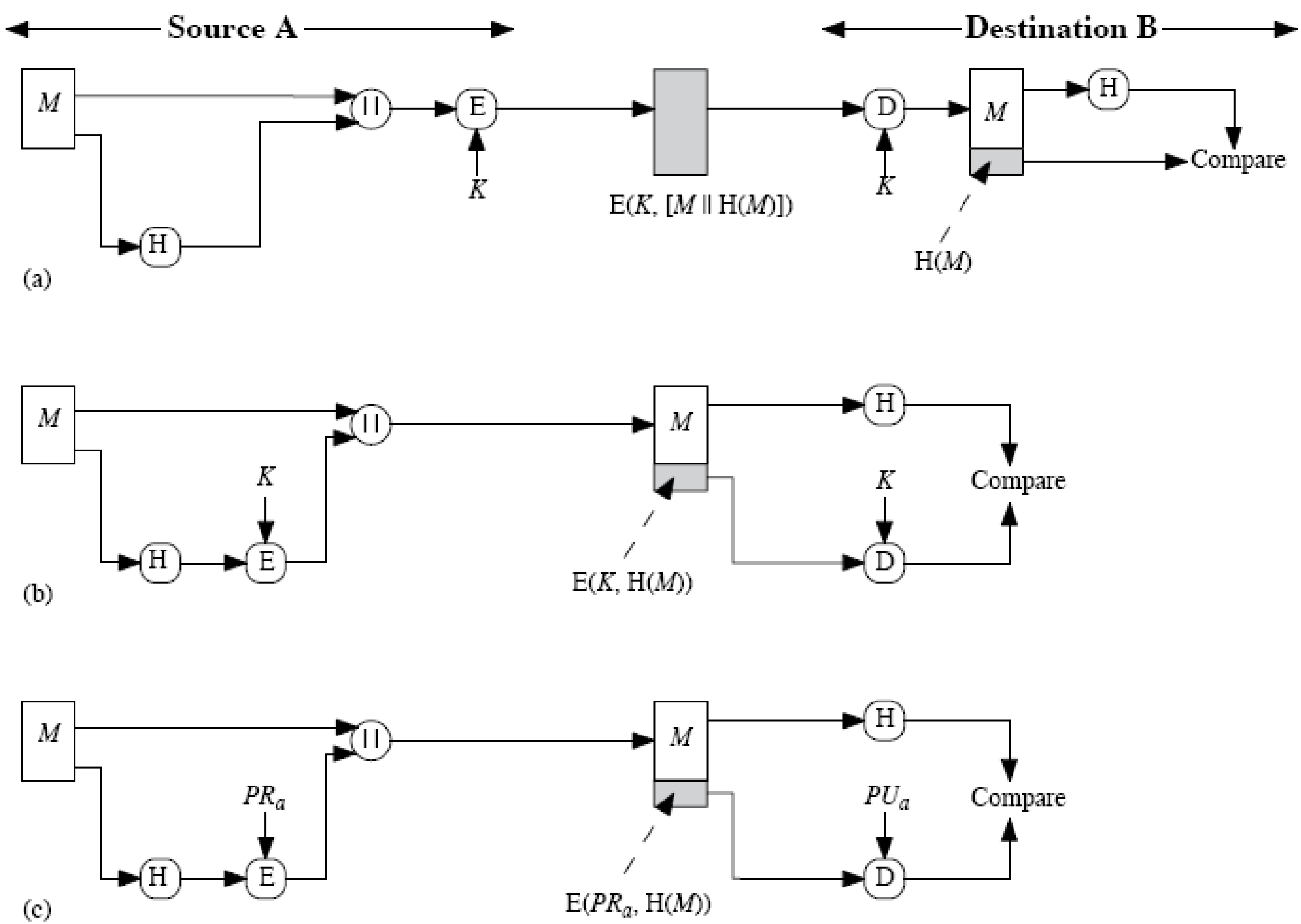
  h = H(M)

  - condenses a variable-length message M
  - to a fixed-sized fingerprint
- assumed to be public

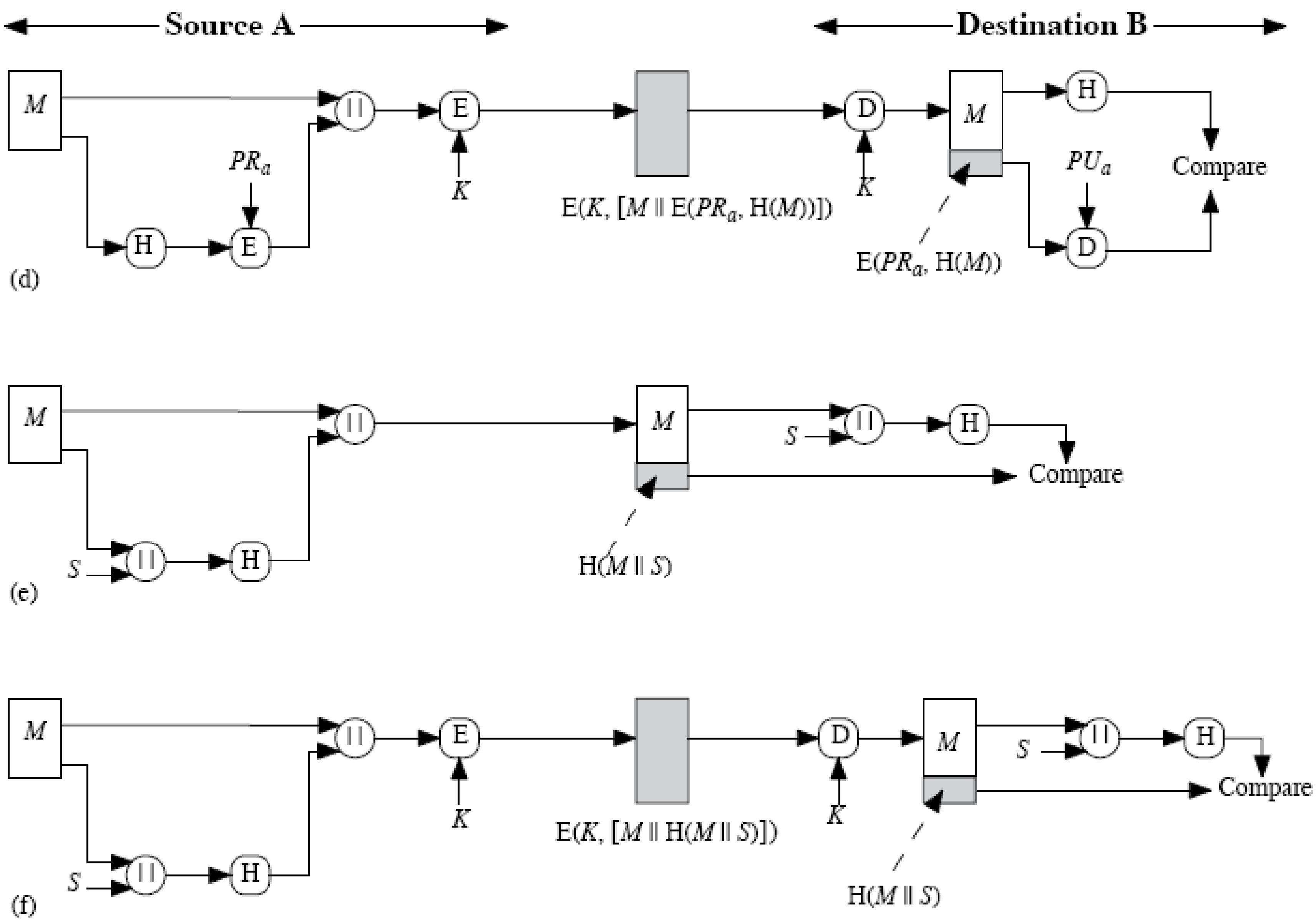
Requirements for Hash Functions

1. can be applied to any sized message `M`
2. produces fixed-length output `h`
3. is easy to compute `h=H(M)` for any message `M`
4. given `h` is infeasible to find `x` s.t. `H(x)=h`
   - one-way property
5. given `x` is infeasible to find `y` s.t. `H(y)=H(x)`
   - weak collision resistance
6. is infeasible to find any `x,y` s.t. `H(y)=H(x)`
   - strong collision resistance

Apps of Hash Functions



Apps of Hash Functions



Apps of Hash Functions

A → B: E(*K*, [*M* || H(*M*)])

- Provides confidentiality
  - Only A and B share *K*
- Provides authentication
  - H(*M*) is cryptographically protected

(a) Encrypt message plus hash code

A → B: *M* || E(*K*, H(*M*))

- Provides authentication
  - H(*M*) is cryptographically protected

(b) Encrypt hash code - shared secret key

A → B: *M* || E(PR_a, H(*M*))

- Provides authentication and digital signature
  - H(*M*) is cryptographically protected
  - Only A could create E(PR_a, H(*M*))

(c) Encrypt hash code - sender's private *key*

A → B: E(*K*, [*M* || E(PR_a, H(*M*))])

- Provides authentication and digital signature
- Provides confidentiality
  - Only A and B share *K*

(d) Encrypt result of (c) - shared secret key

A → B: *M* || H(*M* || *S*)

- Provides authentication
  - Only A and B share *S*

(e) Compute hash code of message plus secret value

A → B: E(*K*, [*M* || H(*M* || *S*)])

- Provides authentication
  - Only A and B share *S*
- Provides confidentiality
  - Only A and B share *K*

(f) Encrypt result of (e)

Simple Hash Functions

- are several proposals for simple functions
- based on XOR of message blocks
  - not secure since can manipulate
- need a stronger cryptographic function

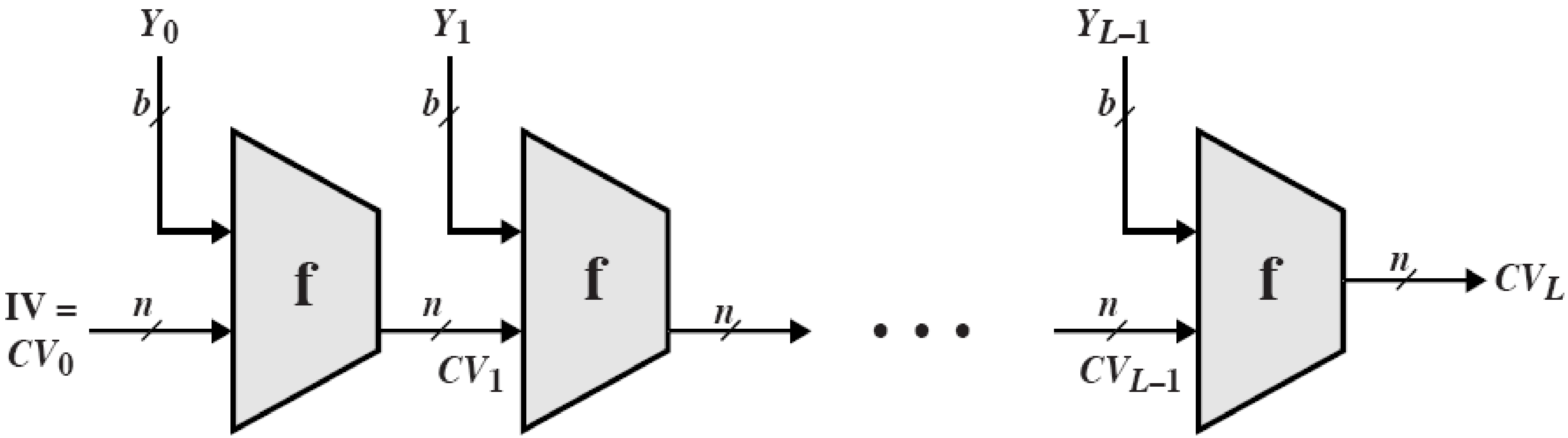
Block Ciphers as Hash Functions

- can use block ciphers as hash functions
  - using $H_0=0$ and zero-pad of final block
  - compute: $H_i = E_{M_i} [H_{i-1}]$
  - and use final block as the hash value
  - similar to CBC but without a key
- resulting hash is too small (64-bit)
- other variants also susceptible to attack

Hash Algorithms

- MD5
- SHA-1
- RIPEMD-160

Hash Algorithm Structure



IV	=	Initial value	L	=	number of input blocks
CV_i	=	chaining variable	n	=	length of hash code
Y_i	=	ith input block	b	=	length of input block
f	=	compression algorithm			

MD5

- designed by Ronald Rivest (the R in RSA)
- latest in a series of MD2, MD4
- produces a 128-bit hash value
- until recently was the most widely used hash algorithm
  - in recent times have both brute-force & cryptanalytic concerns
  - At the rump session of CRYPTO 2004, **she and co-authors** demonstrated collisions in MD5 and other related hash functions. A collision occurs when two distinct messages result in the same hash function output. They received a standing ovation for the work.
- specified as Internet standard RFC1321

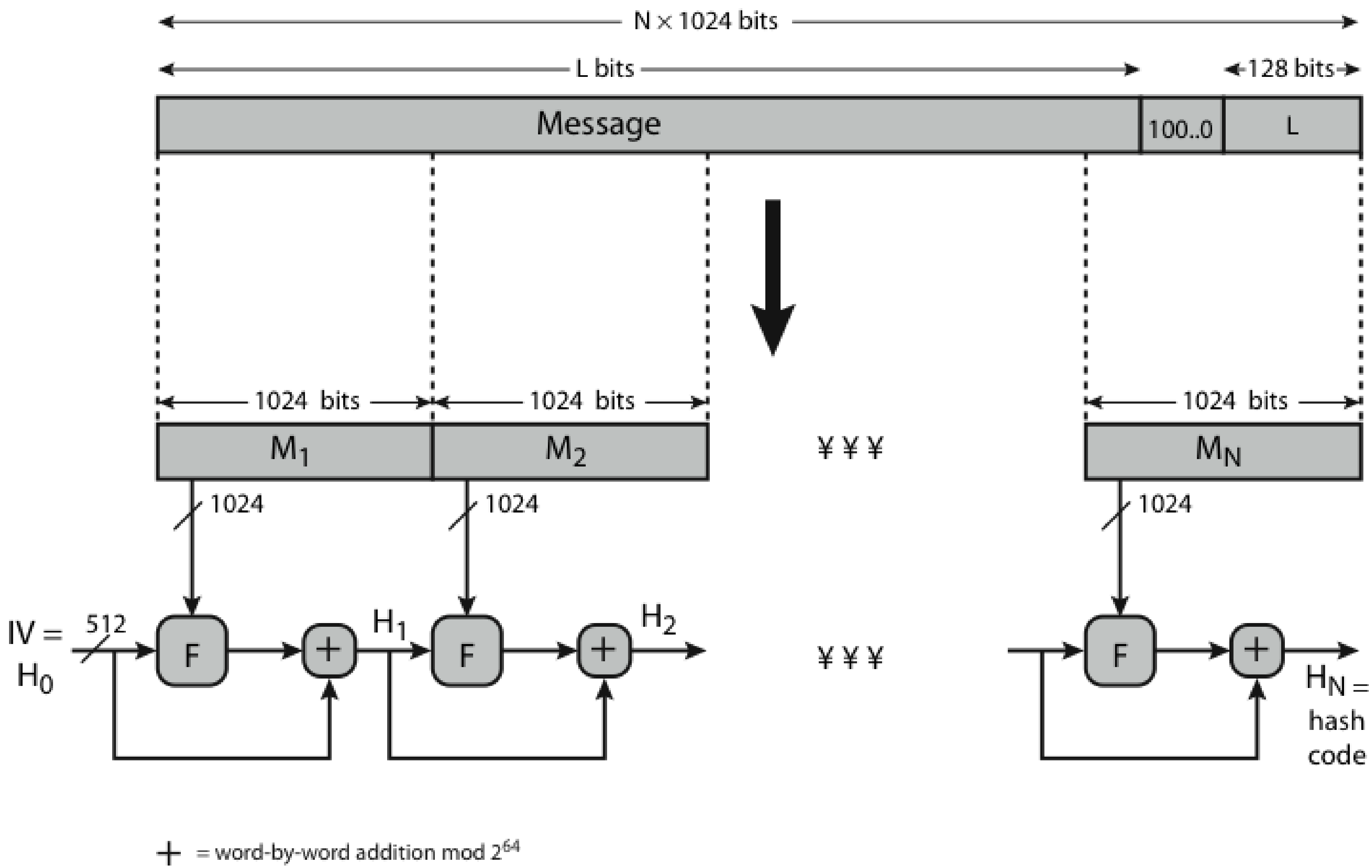
Secure Hash Algorithm

- SHA originally designed by NIST & NSA in 1993
- was revised in 1995 as SHA-1
- US standard for use with DSA signature scheme
  - standard is FIPS 180-1 1995, also Internet RFC3174
  - nb. the algorithm is SHA, the standard is SHS
- based on design of MD4 with key differences
- produces 160-bit hash values
- recent 2005 results on security of SHA-1 have raised concerns on its use in future applications

Revised Secure Hash Standard

- NIST issued revision FIPS 180-2 in 2002
- adds 3 additional versions of SHA
  - SHA-256, SHA-384, SHA-512
- designed for compatibility with increased security provided by the AES cipher
- structure & detail is similar to SHA-1
- hence analysis should be similar
- but security levels are rather higher

SHA-512 Overview



RIPEMD-160

- RIPEMD-160 was developed in Europe as part of RIPE project in 96
- by researchers involved in attacks on MD4/5
- initial proposal strengthen following analysis to become RIPEMD-160
- somewhat similar to MD5/SHA
- uses 2 parallel lines of 5 rounds of 16 steps
- creates a 160-bit hash value
- slower, but probably more secure, than SHA

Hash Functions & MAC Security

- like block ciphers have:
- **brute-force** attacks exploiting
  - strong collision resistance hash have cost $2^{m/2}$
    - have proposal for h/w MD5 cracker
    - 128-bit hash looks vulnerable, 160-bits better
  - MACs with known message-MAC pairs
    - can either attack keyspace (cf key search) or MAC
    - at least 128-bit MAC is needed for security

- **cryptanalytic attacks** exploit structure
  - like block ciphers want brute-force attacks to be the best alternative
- have a number of analytic attacks on iterated hash functions
  - $CV_i = f[CV_{i-1}, M_i]$; $H(M)=CV_N$
  - typically focus on collisions in function f
  - like block ciphers is often composed of rounds
  - attacks exploit properties of round functions

Summary

- have considered:
  - message authentication using
    - message encryption
    - MACs
    - hash functions
  - general approach & security
  - Some MACs & hash functions's examples