

# Cryptographic Hash Functions

---

Vitaly Shmatikov

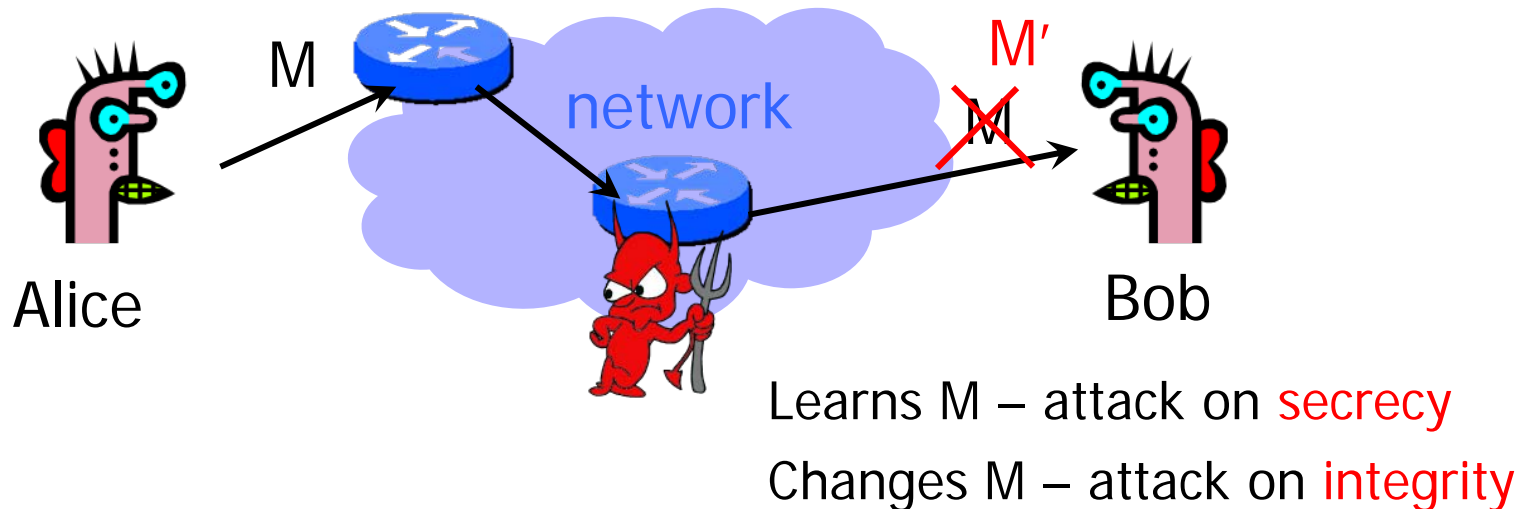
# Reading Assignment

---

◆ Read Kaufman 5.1-2 and 5.6-7

# Communication on the Internet

- ◆ Basic issue:  
sending messages via **untrusted intermediaries**



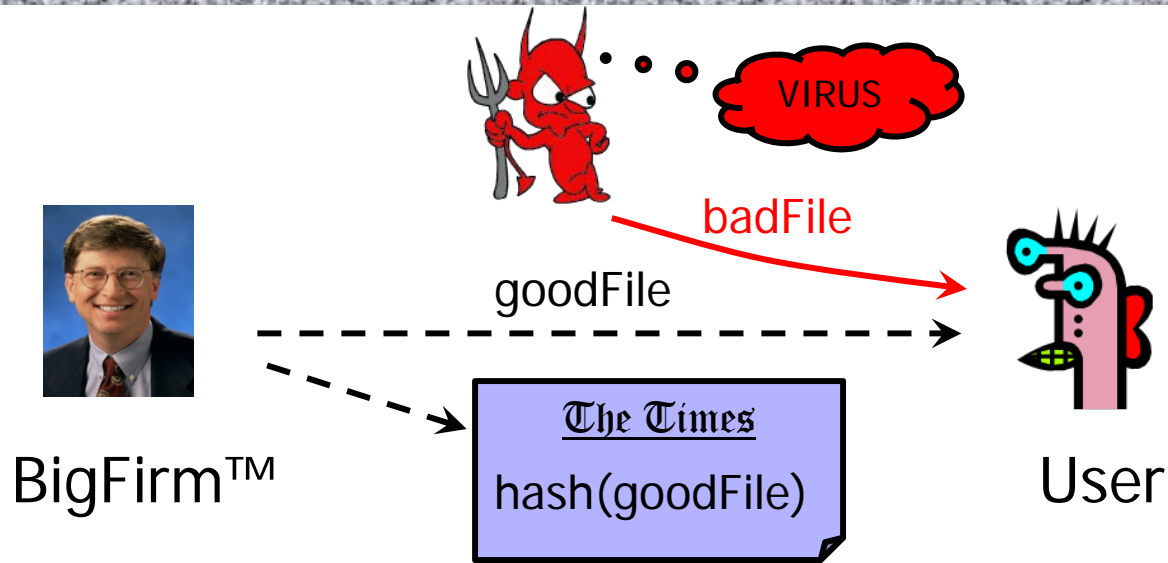
Web access, email, remote login, file transfer...

# Integrity vs. Secrecy

---

- ◆ **Integrity:** attacker cannot tamper with message
- ◆ Encryption may not guarantee integrity!
  - Intuition: attacker may be able to modify message under encryption without learning what it is
    - Given one-time key  $K$ , encrypt  $M$  as  $M \oplus K$ ... Perfect secrecy, but can easily change  $M$  under encryption to  $M \oplus M'$  for any  $M'$
    - Online auction: halve competitor's bid without learning its value
  - This is recognized by industry standards (e.g., PKCS)
    - "RSA encryption is intended primarily to provide confidentiality... It is not intended to provide integrity"
  - Many encryption schemes provide secrecy AND integrity

# More on Integrity



Software manufacturer wants to ensure that the executable file is received by users without modification...

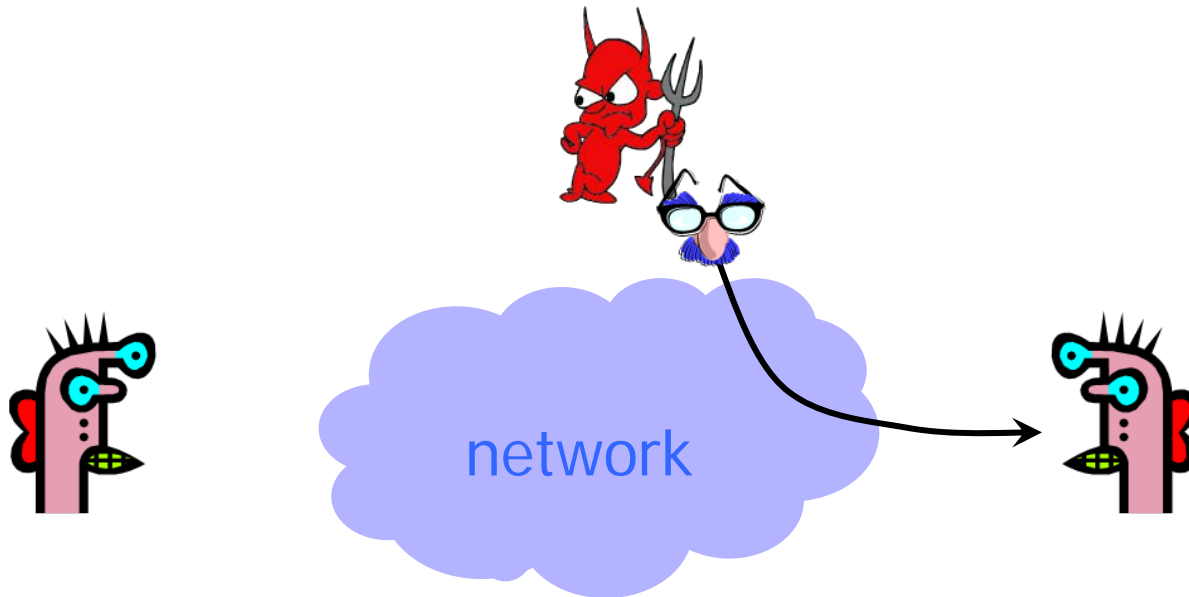
Sends out the file to users and publishes its hash in NY Times

The goal is integrity, not secrecy

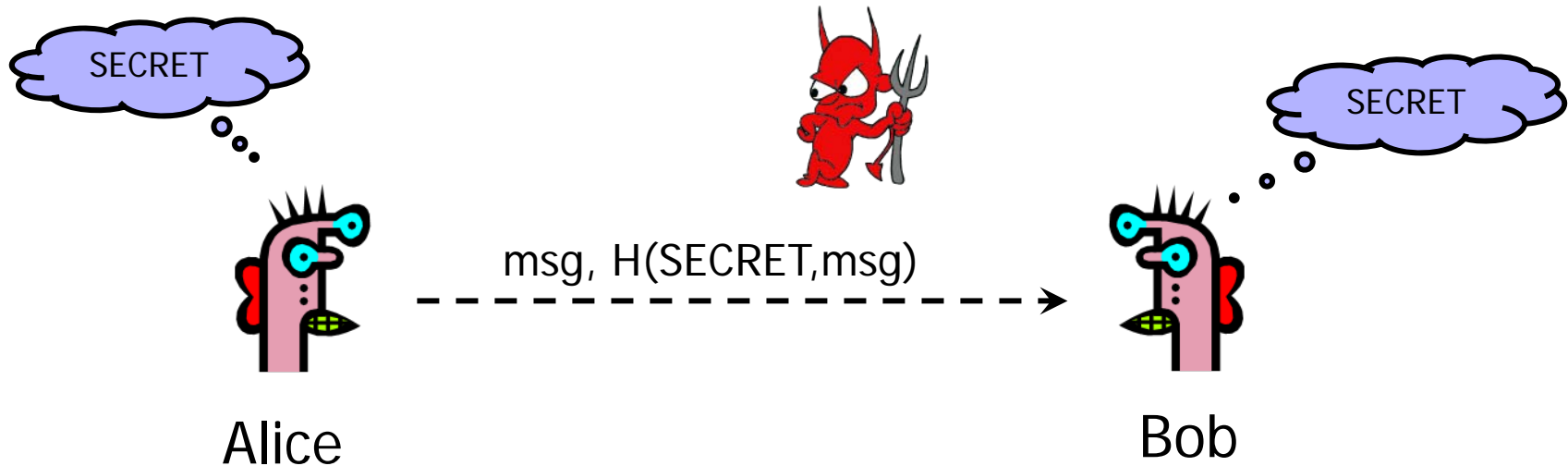
Idea: given goodFile and hash(goodFile),  
very hard to find badFile such that  $\text{hash}(\text{goodFile}) = \text{hash}(\text{badFile})$

# Authentication

- ◆ Authenticity is **identification and assurance of origin of information**
  - We'll see many specific examples in different scenarios



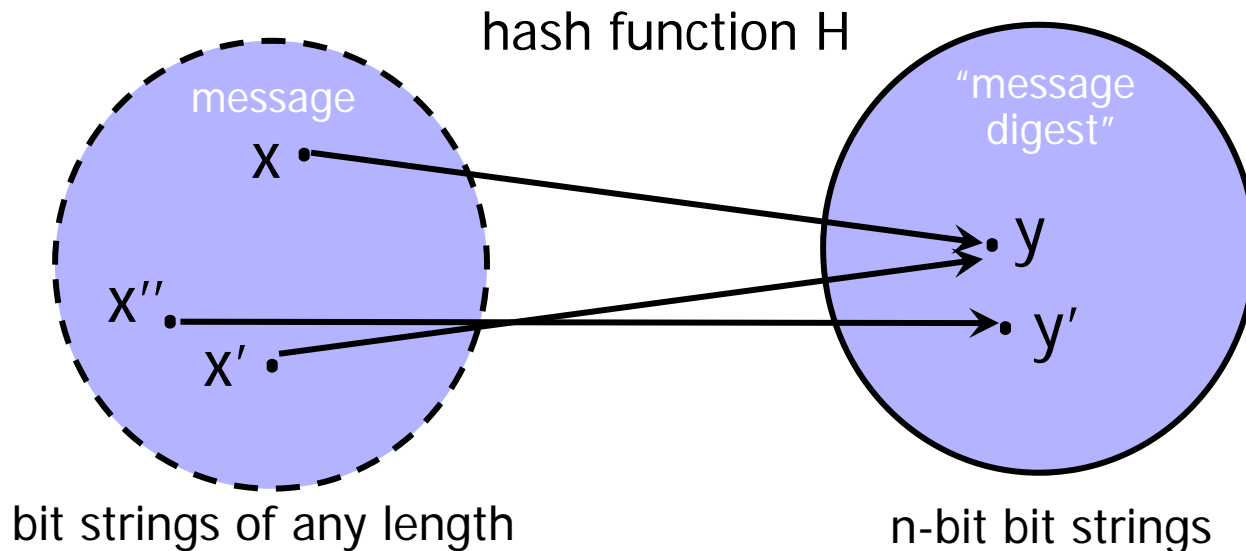
# Authentication with Shared Secrets



Alice wants to ensure that nobody modifies message in transit (both integrity and authentication)

Idea: given msg,  
very hard to compute  $H(\text{SECRET}, \text{msg})$  without SECRET;  
easy with SECRET

# Hash Functions: Main Idea



## ◆ H is a lossy compression function

- **Collisions:**  $h(x)=h(x')$  for some inputs  $x, x'$
- Result of hashing should "look random" (make this precise later)
  - Intuition: half of digest bits are "1"; any bit in digest is "1" half the time

## ◆ **Cryptographic hash function** needs a few properties...

# One-Way

---

## ◆ Intuition: hash should be hard to invert

- “Preimage resistance”
- Let  $h(x') = y \in \{0,1\}^n$  for a random  $x'$
- Given  $y$ , it should be hard to find any  $x$  such that  $h(x) = y$

## ◆ How hard?

- Brute-force: try every possible  $x$ , see if  $h(x) = y$
- SHA-1 (common hash function) has 160-bit output
  - Suppose have hardware that'll do  $2^{30}$  trials a pop
  - Assuming  $2^{34}$  trials per second, can do  $2^{89}$  trials per year
  - Will take  $2^{71}$  years to invert SHA-1 on a random image

# "Birthday Paradox"

---

- ◆ T people
- ◆ Suppose each birthday is a random number taken from K days ( $K=365$ ) – how many possibilities?
  - $K^T$  (samples with replacement)
- ◆ How many possibilities that are all different?
  - $(K)_T = K(K-1)\dots(K-T+1)$  samples without replacement
- ◆ Probability of no repetition?
  - $(K)_T / K^T \approx 1 - T(T-1)/2K$
- ◆ Probability of repetition?
  - $O(T^2)$

# Collision Resistance

- ◆ Should be hard to find  $x, x'$  such that  $h(x)=h(x')$
- ◆ Brute-force collision search is  $O(2^{n/2})$ , not  $O(2^n)$ 
  - $n$  = number of bits in the output of hash function
  - For SHA-1, this means  $O(2^{80})$  vs.  $O(2^{160})$
- ◆ Reason: birthday paradox
  - Let  $T$  be the number of values  $x, x', x'' \dots$  we need to look at before finding the first pair  $x, x'$  s.t.  $h(x)=h(x')$
  - Assuming  $h$  is random, what is the probability that we find a repetition after looking at  $T$  values?  $O(T^2)$
  - Total number of pairs?  $O(2^n)$
  - Conclusion:  $T \approx O(2^{n/2})$

# One-Way vs. Collision Resistance

---

- ◆ One-wayness does not imply collision resistance
  - Suppose  $g$  is one-way
  - Define  $h(x)$  as  $g(x')$  where  $x'$  is  $x$  except the last bit
    - $h$  is one-way (to invert  $h$ , must invert  $g$ )
    - Collisions for  $h$  are easy to find: for any  $x$ ,  $h(x0)=h(x1)$
- ◆ Collision resistance does not imply one-wayness
  - Suppose  $g$  is collision-resistant
  - Define  $h(x)$  to be  $0x$  if  $x$  is  $n$ -bit long,  $1g(x)$  otherwise
    - Collisions for  $h$  are hard to find: if  $y$  starts with  $0$ , then there are no collisions, if  $y$  starts with  $1$ , then must find collisions in  $g$
    - $h$  is not one way: half of all  $y$ 's (those whose first bit is  $0$ ) are easy to invert (how?); random  $y$  is invertible with probab.  $1/2$

# Weak Collision Resistance

---

- ◆ Given randomly chosen  $x$ , hard to find  $x'$  such that  $h(x) = h(x')$ 
  - Attacker must find collision for a specific  $x$ . By contrast, to break collision resistance, enough to find any collision.
  - Brute-force attack requires  $O(2^n)$  time
- ◆ Weak collision resistance does not imply collision resistance (why?)

# Which Property Do We Need?

---

- ◆ UNIX passwords stored as  $\text{hash}(\text{password})$ 
  - One-wayness: hard to recover password
- ◆ Integrity of software distribution
  - Weak collision resistance
  - But software images are not really random... maybe need full collision resistance
- ◆ Auction bidding
  - Alice wants to bid  $B$ , sends  $H(B)$ , later reveals  $B$
  - One-wayness: rival bidders should not recover  $B$
  - Collision resistance: Alice should not be able to change her mind to bid  $B'$  such that  $H(B) = H(B')$

# Common Hash Functions

---

## ◆ MD5

- 128-bit output
- Still used very widely
- Completely broken by now

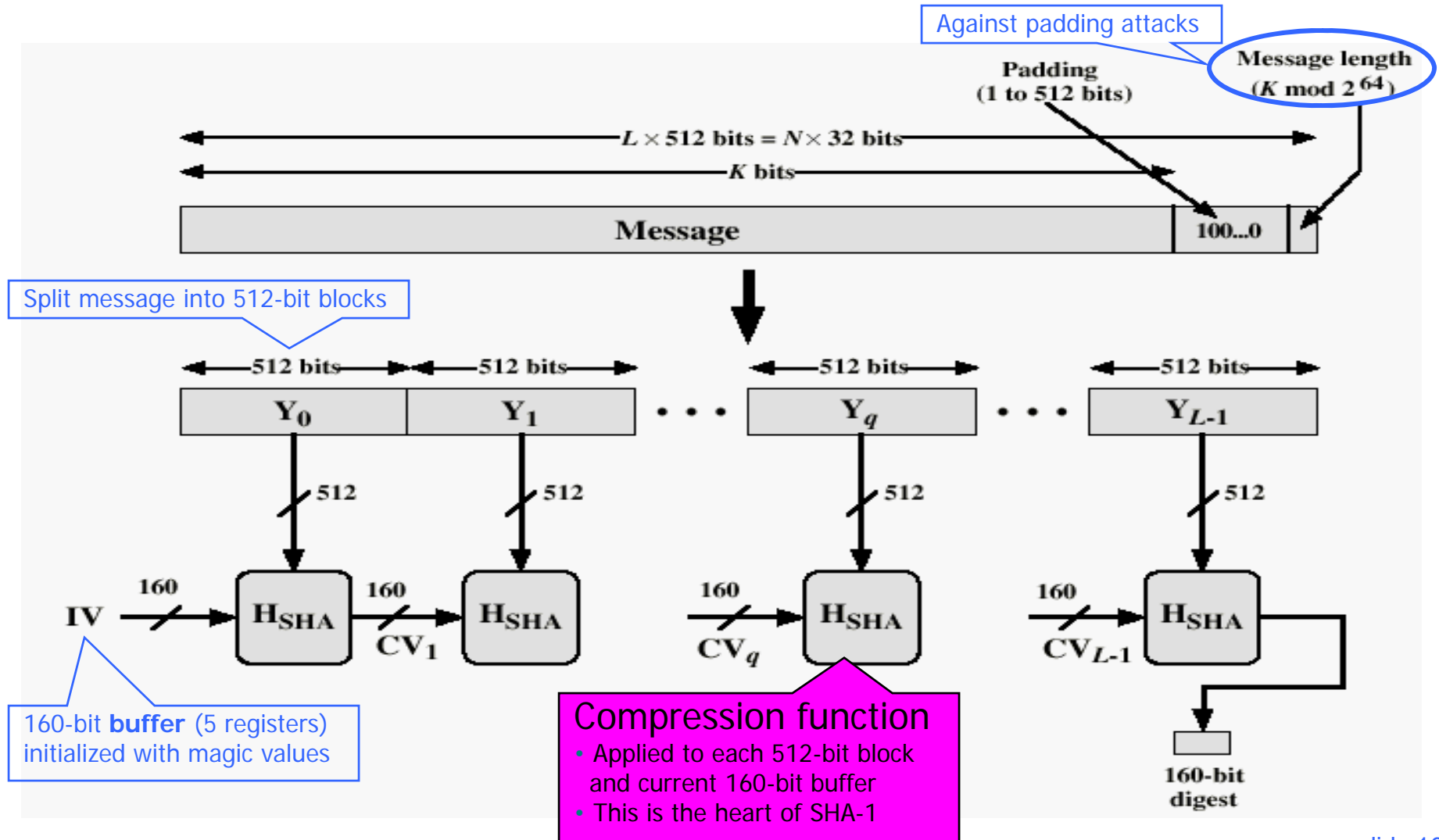
## ◆ RIPEMD-160

- 160-bit variant of MD-5

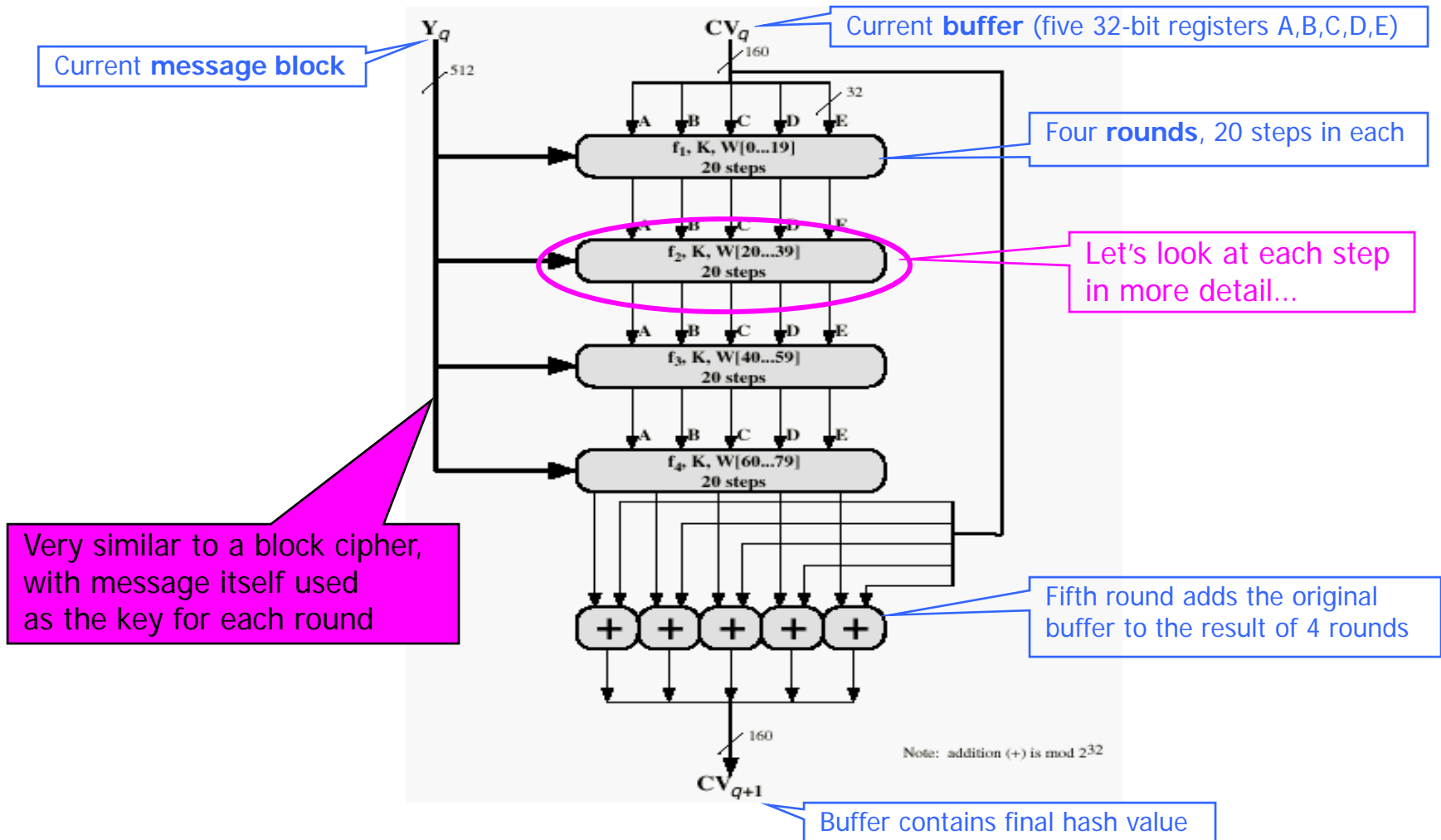
## ◆ SHA-1 (Secure Hash Algorithm)

- 160-bit output
- US government (NIST) standard as of 1993-95
  - Also the hash algorithm for Digital Signature Standard (DSS)

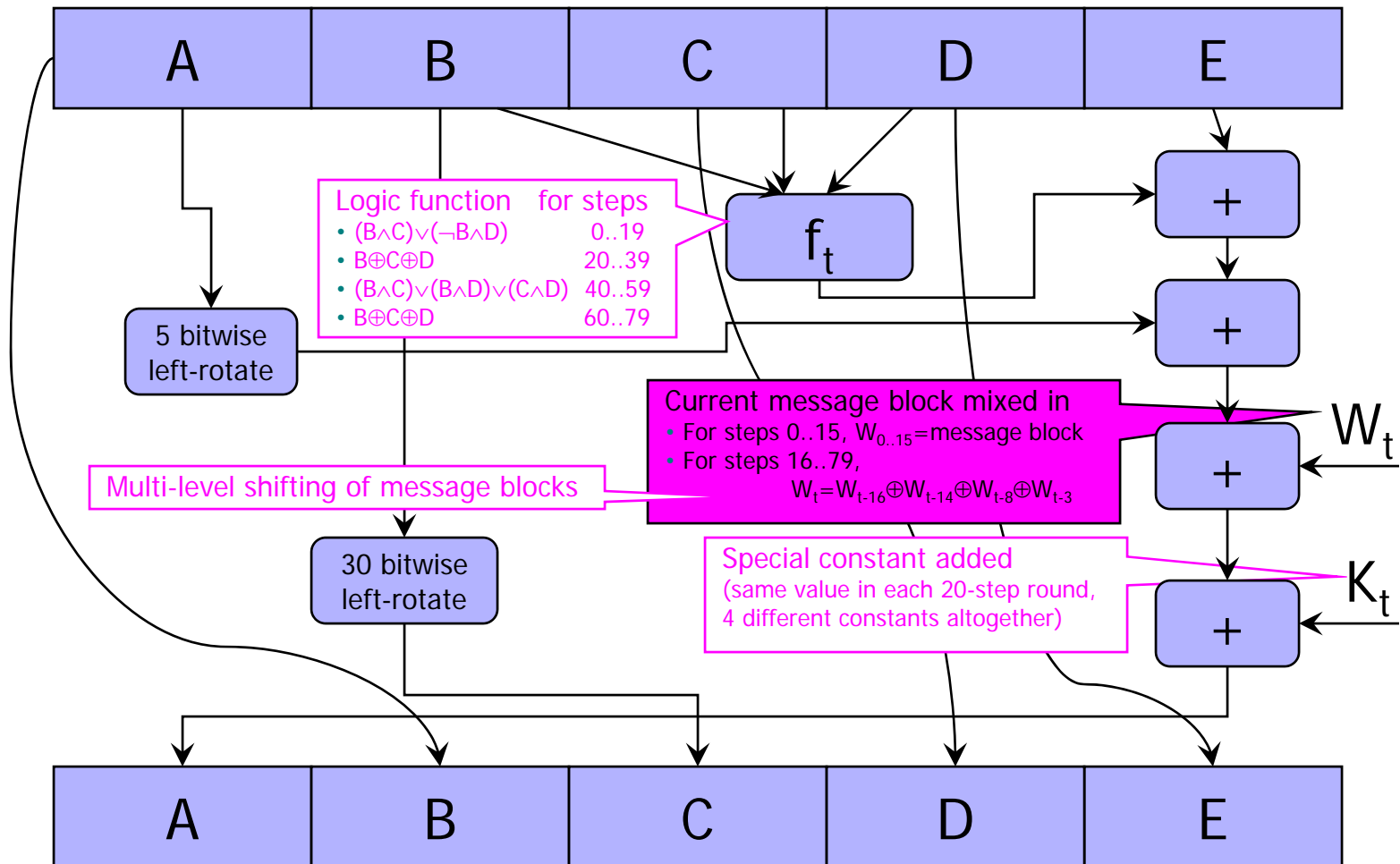
# Basic Structure of SHA-1



# SHA-1 Compression Function



# One Step of SHA-1 (80 steps total)

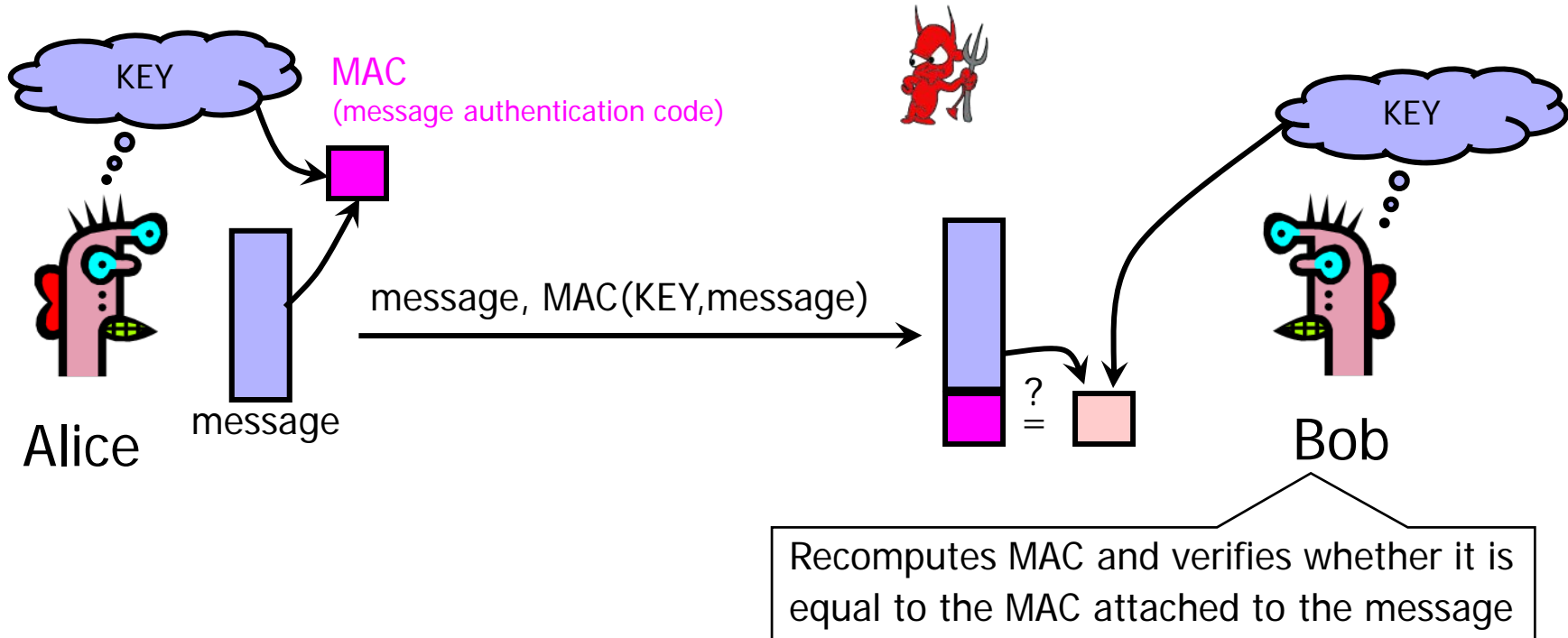


# How Strong Is SHA-1?

---

- ◆ Every bit of output depends on every bit of input
  - Very important property for collision-resistance
- ◆ Brute-force inversion requires  $2^{160}$  ops, birthday attack on collision resistance requires  $2^{80}$  ops
- ◆ Some recent weaknesses (2005)
  - Collisions can be found in  $2^{63}$  ops

# Authentication Without Encryption



Integrity and authentication: only someone who knows KEY can compute MAC for a given message

# HMAC

---

- ◆ Construct MAC by applying a cryptographic hash function to message and key
  - Could also use encryption instead of hashing, but...
  - Hashing is faster than encryption in software
  - Library code for hash functions widely available
  - Can easily replace one hash function with another
  - There used to be US export restrictions on encryption
- ◆ Invented by Bellare, Canetti, and Krawczyk (1996)
- ◆ Mandatory for IP security, also used in SSL/TLS

# Structure of HMAC

