CIS 6930/4930 Computer and Network Security

Topic 4. Cryptographic Hash Functions

Hash Function

- Also known as
  - Message digest
  - One-way transformation
  - One-way function
  - Hash
- Length of $H(m)$ much shorter then length of m
- Usually fixed lengths: 128 or 160 bits

Desirable Properties of Hash Functions

- Consider a hash function H
  - Performance: Easy to compute H(*m*)
  - One-way property: Given H(*m*) but not *m*, it's computationally infeasible to find *m*
  - Weak collision resistance (free): Given H(*m*), it's computationally infeasible to find *m'* such that H(*m'*) = H(*m*).
  - Strong collision resistance (free): Computationally infeasible to find m_1, m_2 such that $H(m_1) = H(m_2)$

Length of Hash Image

- Question
  - Why do we have 128 bits or 160 bits in the output of a hash function?
  - If it is too long
    - Unnecessary overhead
  - If it is too short
    - Loss of strong collision free property
    - Birthday paradox

Birthday Paradox

- Question:
  - What is the smallest group size *k* such that
    - The probability that at least two people in the group have the same birthday is greater than 0.5?
    - Assume 365 days a year, and all birthdays are equally likely
  - P(*k* people having *k* different birthdays):

    $Q(365,k) = (1-1/365) \times (1-2/365) \times (1-3/365) \times \ldots \times \{1-(k-1)/365\}$

    $= (364/365) \times (363/365) \times (362/365) \times \ldots \times \{(365-(k-1))/365\}$

    $= 365!/(365-k)!365^k$
  - P(at least two people have the same birthday):

    $P(365,k) = 1-Q(365,k) \geq 0.5$
  - *k* is about 23

Birthday Paradox (Cont'd)

- Generalization of birthday paradox
  - Given
    - a random integer with uniform distribution between 1 and n, and
    - a selection of k instances of the random variables,
  - What is the least value of k such that
    - There will be at least one duplicate
    - with probability $P(n,k) > 0.5$, ?

Birthday Paradox (Cont'd)

- Generalization of birthday paradox
  - $P(n,k) = 1-\{n!/(n-k)!n^k\} \approx 1 - e^{-k*(k-1)/2n}$
  - For large *n* and *k*, to have $P(n,k) > 0.5$ with the smallest *k*, we have

$$k = \sqrt{2(\ln 2)n} = 1.18\sqrt{n} \approx \sqrt{n}$$

  - Example
    - $1.18*(365)^{1/2} = 22.54$

Birthday Paradox (Cont'd)

- Implication for hash function H of length m
    - The hash value of an arbitrary input message is randomly distributed between 1 and 2^m
  - What is the least value of *k* such that
    - If we hash k messages, the probability that at least two of them have the same hash is larger than 0.5?

$$k \approx \sqrt{n} = \sqrt{2^m} = 2^{m/2}$$

  - Birthday attack
    - Choose $m \geq 128$

Hash Function Applications

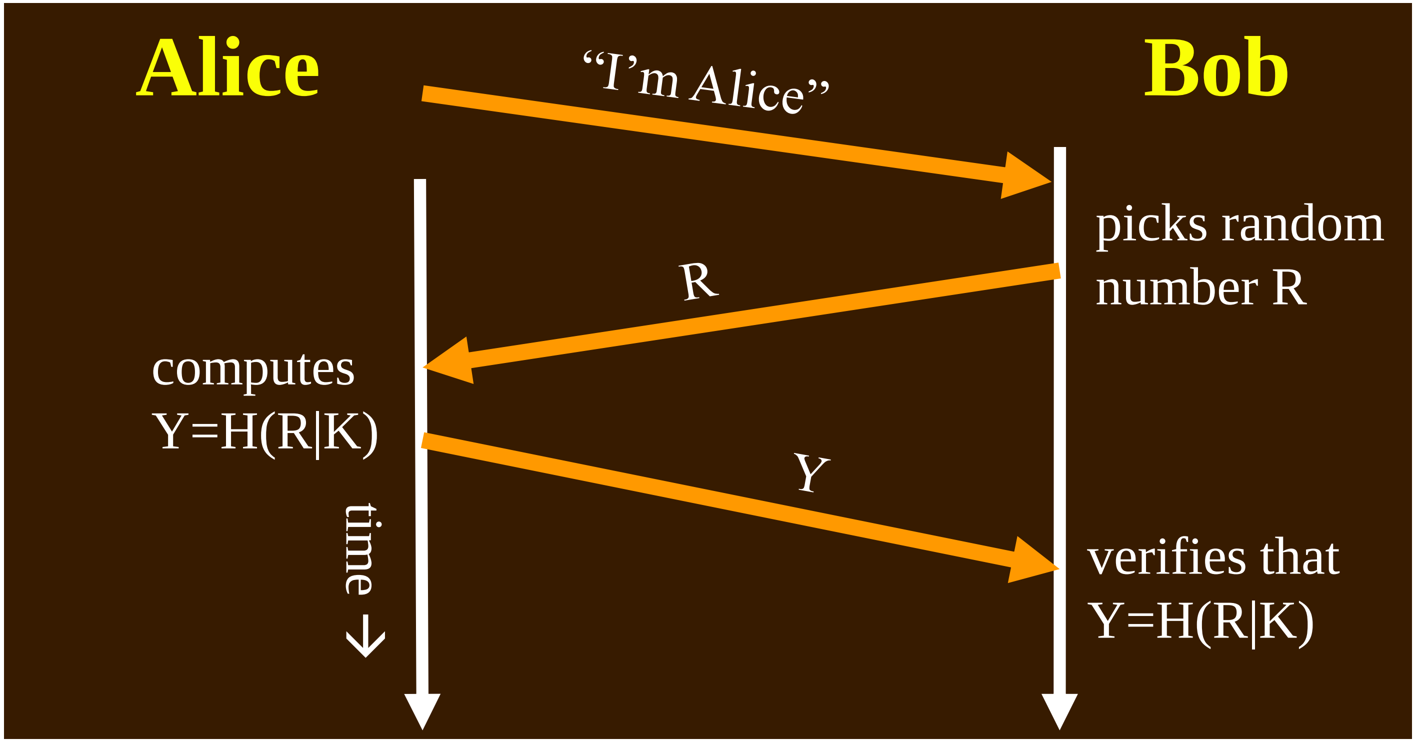
Application: File Authentication

- Want to detect if a file has been changed by someone after it was stored
- Method
  - Compute a hash H(F) of file F
  - Store H(F) separately from F
  - Can tell at any later time if F has been changed by computing H(F’) and comparing to stored H(F)

- Why not just store a duplicate copy of F???

Application: User Authentication

- Alice wants to authenticate herself to Bob
  - assuming they already share a secret key K
- Protocol:



User Authentication… (cont'd)

- Why not just send…
  - …K, in plaintext?
  - …H(K)? , i.e., what's the purpose of R?

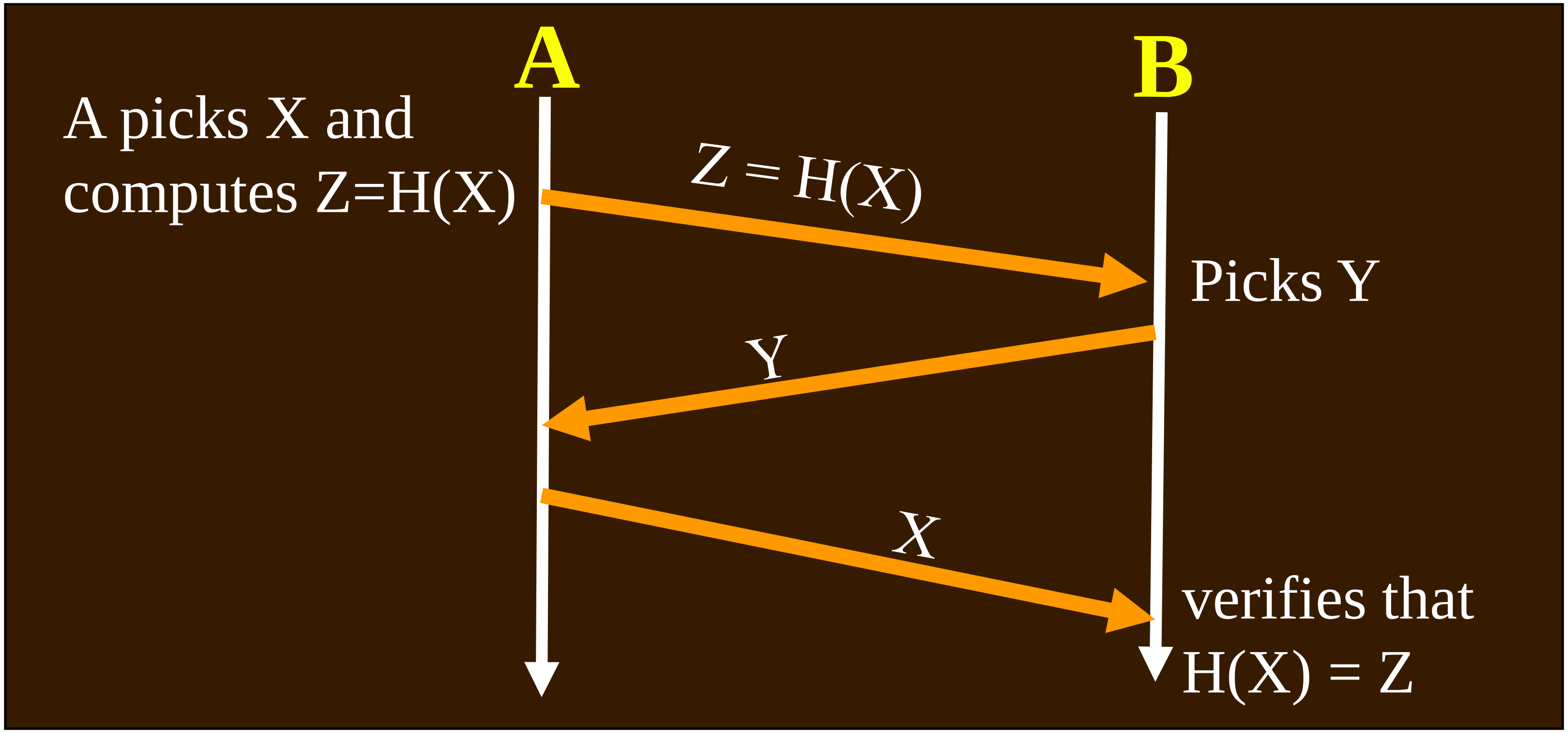
Application: Commitment Protocols

- Ex.: A and B wish to play the game of "odd or even" over the network
  1. A picks a number X
  2. B picks another number Y
  3. A and B "simultaneously" exchange X and Y
  4. A wins if X+Y is odd, otherwise B wins

- If A gets Y before deciding X, A can easily cheat (and vice versa for B)
  - How to prevent this?

Commitment… (Cont'd)

- Proposal: A must commit to X before B will send Y
- Protocol:



- Can either A or B successfully cheat now?

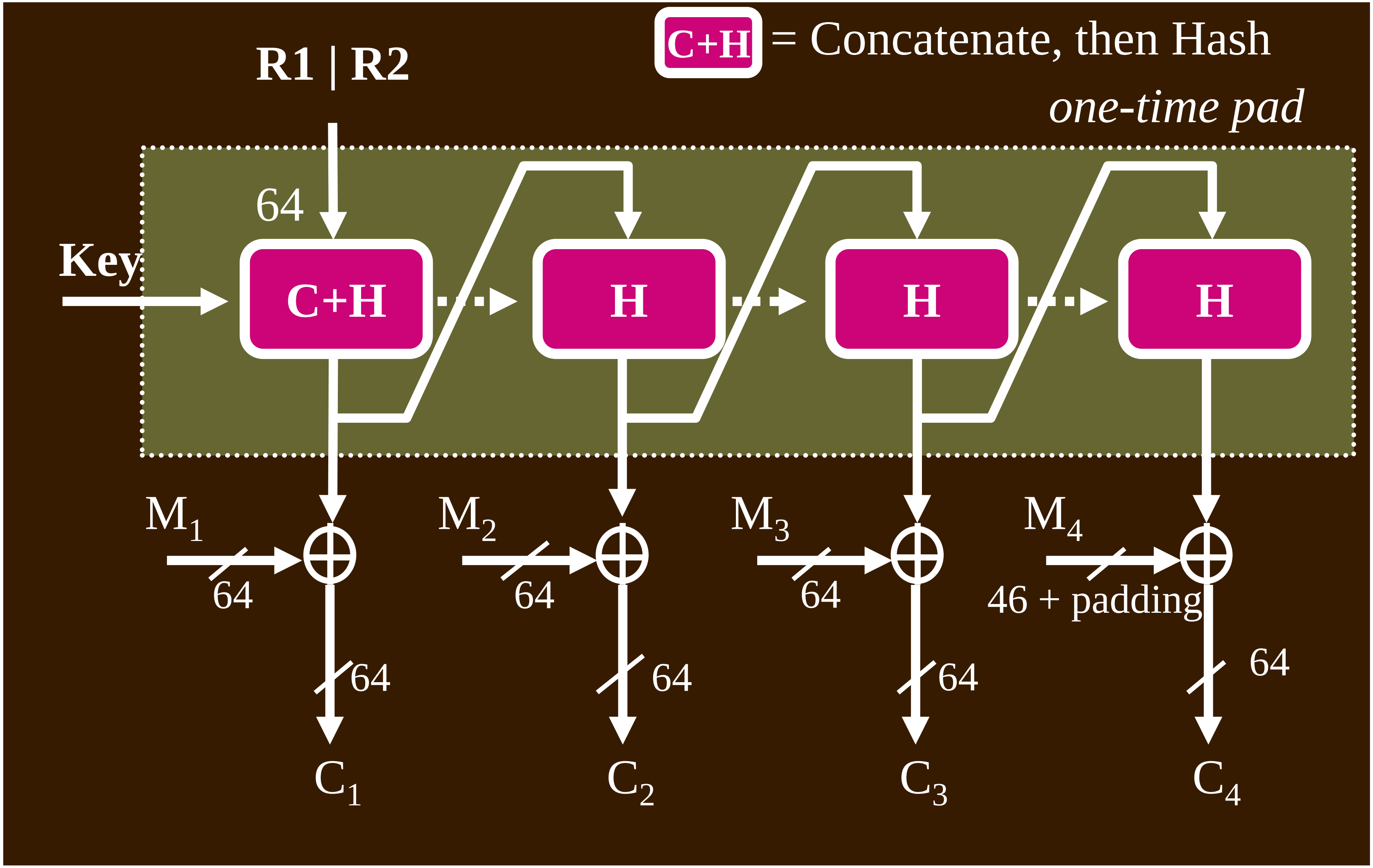
Commitment… (Cont'd)

- Why is sending H(X) better than sending X?
- Why is sending H(X) good enough to prevent A from cheating?
- Why is it not necessary for B to send H(Y) (instead of Y)?
- What problems are there if:

  The set of possible values for X is small?

Application: Message Encryption

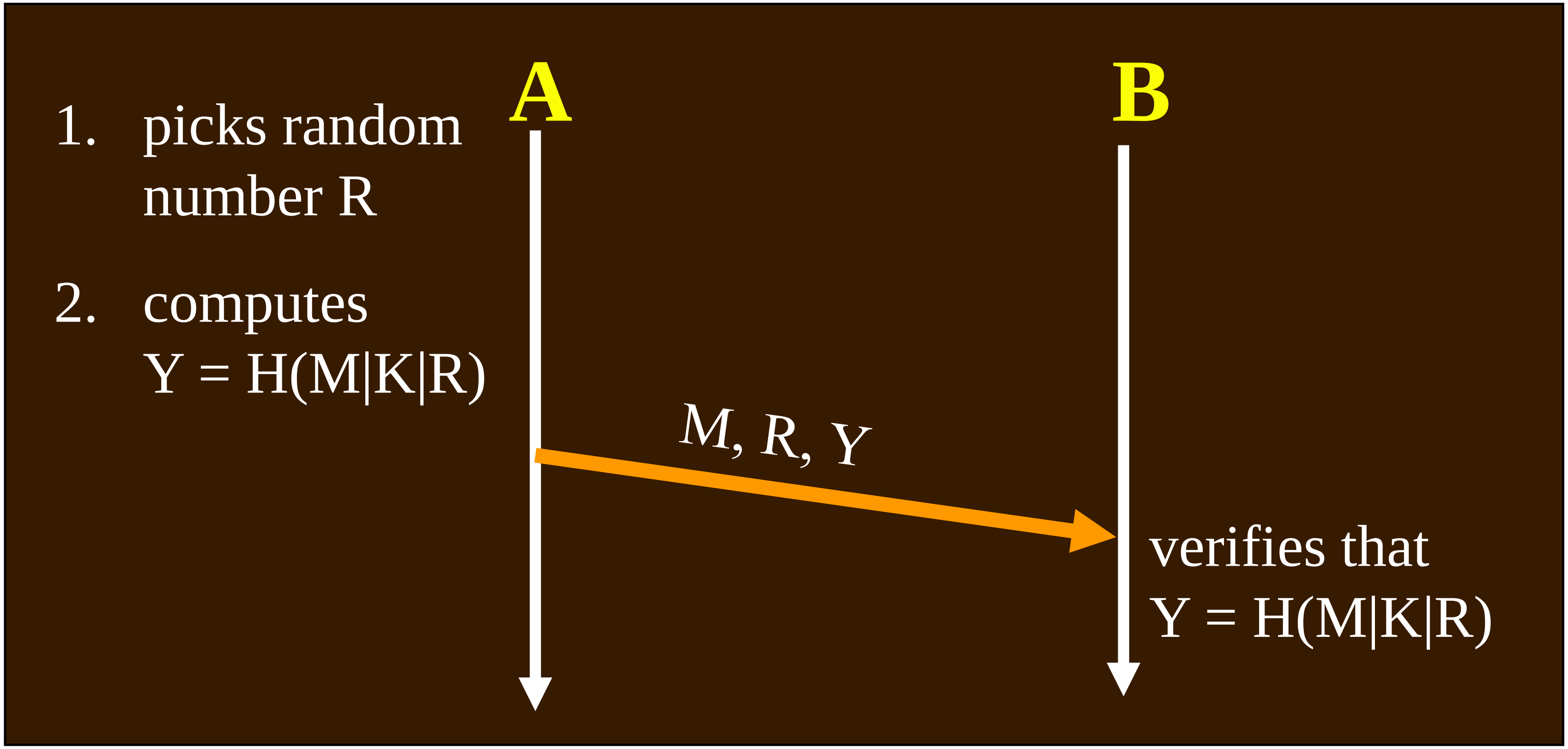
- Assume A and B share a secret key K
  - but don't want to just use encryption of the message with K
- A sends B the (encrypted) random number R1,
  B sends A the (encrypted) random number R2
- And then…



- R1 | R2 is used like the IV of OFB mode, but C+H replaces encryption; Why do we use the key at all, if R1 | R2 is secure?

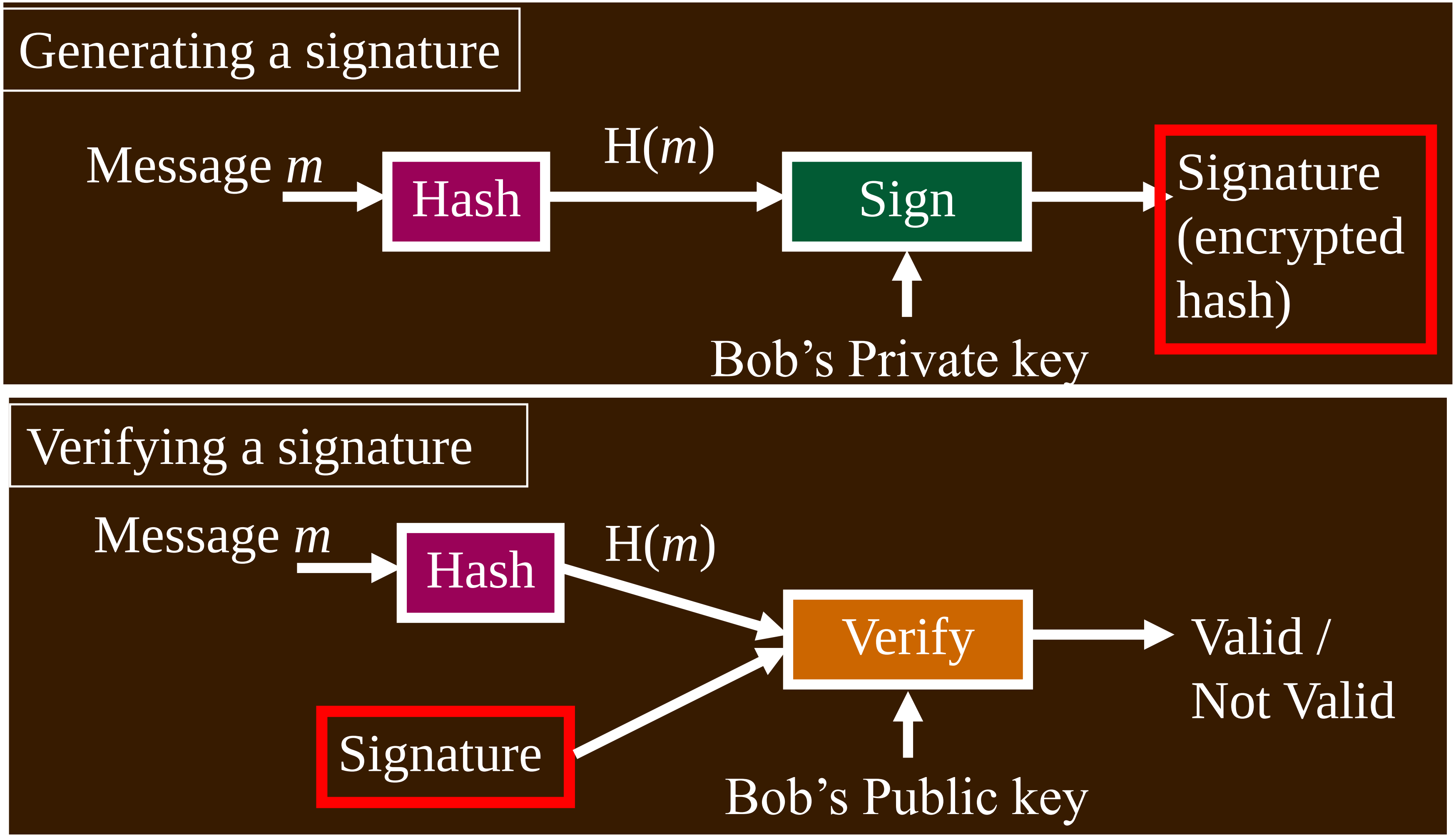
Application: Message Authentication

- A wishes to authenticate (but not encrypt) a message M (and A, B share secret key K)



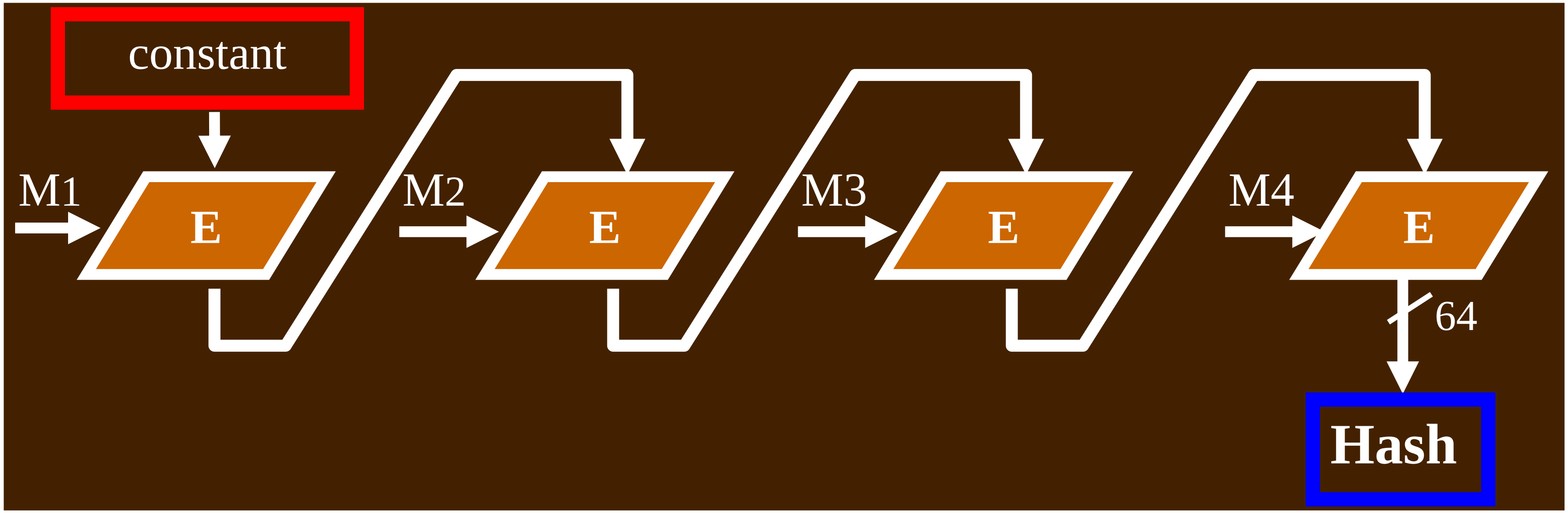
- Why is R needed? Why is K needed?

Application: Digital Signatures



- Only one party (Bob) knows the private key

Is Encryption a Good Hash Function?



- Building hash using block chaining techniques
  - Encryption block size may be too short (DES=64)
    - Birthday attack
  - Expensive in terms of computation time

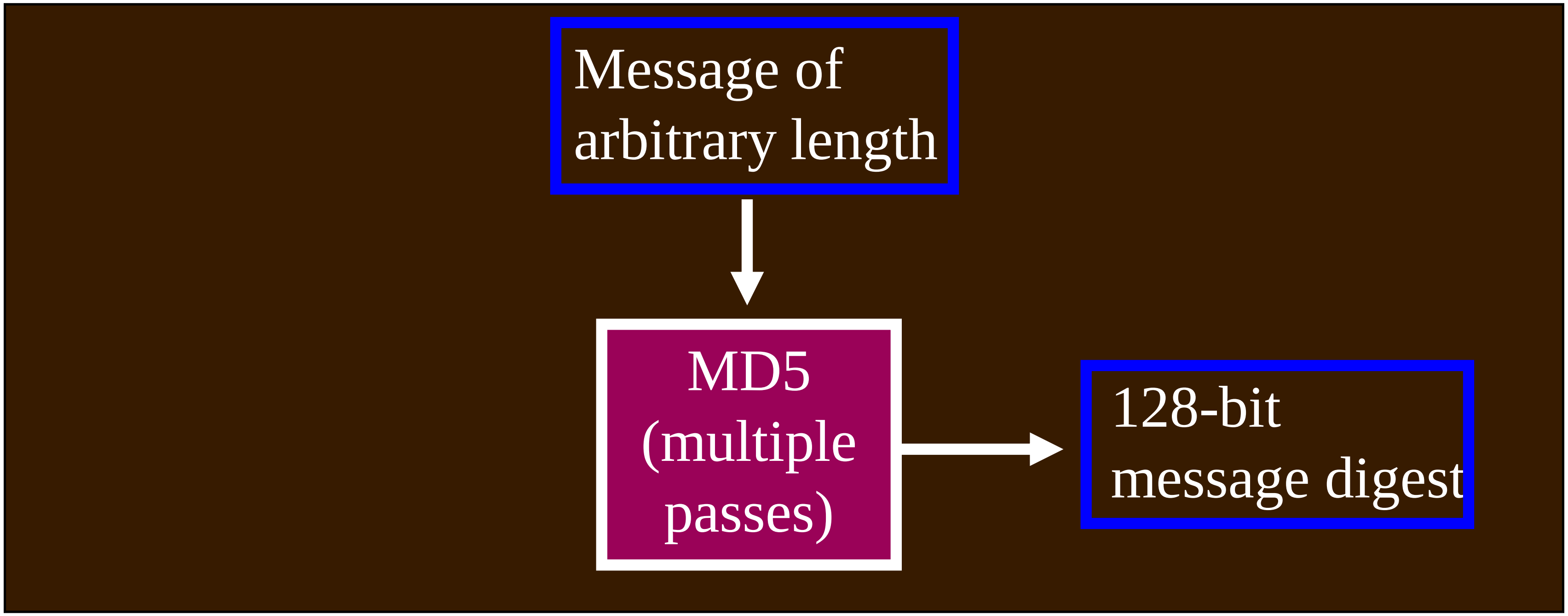
Modern Hash Functions

- MD5
  - Previous versions (i.e., MD2, MD4) have weaknesses.
  - Broken; collisions published in August 2004
  - Previous versions are too weak to be used for serious applications
- SHA (Secure Hash Algorithm)
  - Weaknesses were found
- SHA-1
  - Broken, but not yet cracked
  - Collisions in 2^{69} hash operations, much less than the birthday attack of 2^{80} operations
  - Results were circulated in February 2005, and published in CRYPTO ’05 in August 2005
- SHA-256, SHA-384, …

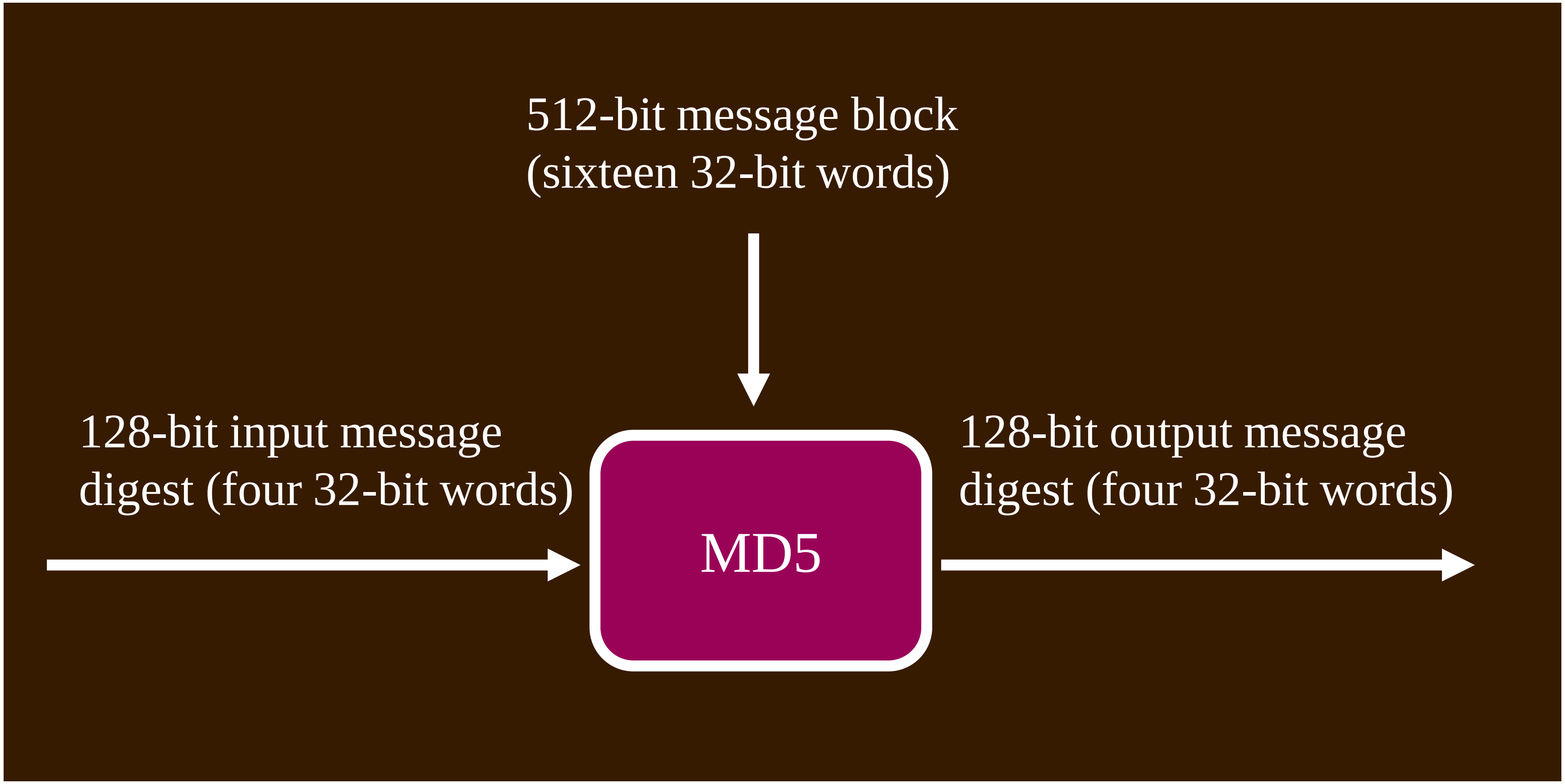
The MD5 Hash Function

MD5: Message Digest Version 5

- MD5 at a glance

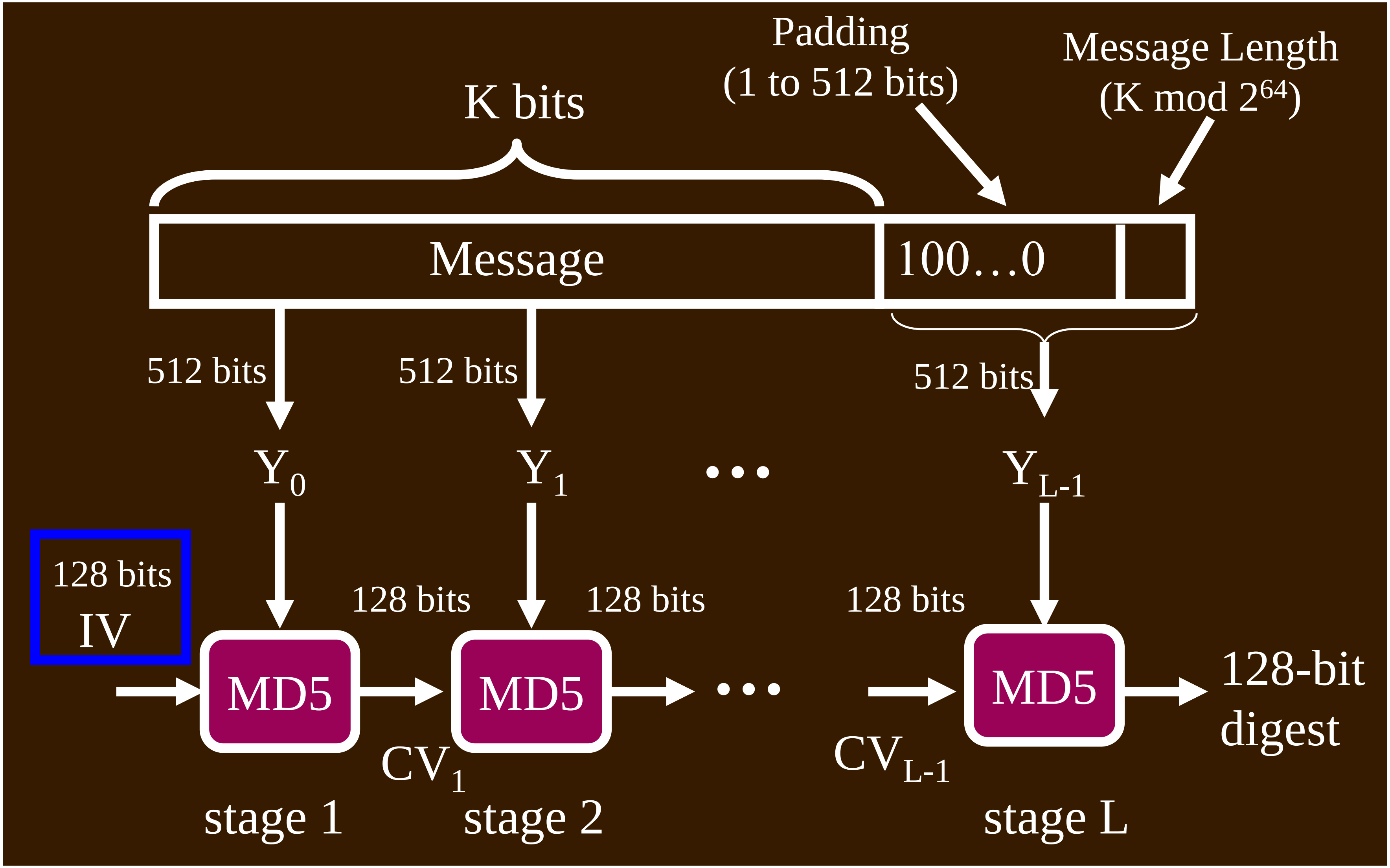


Processing of A Single Block



Called a compression function

MD5: A High-Level View



Padding

- There is always padding for MD5, and padded messages must be multiples of 512 bits
- To original message M, add padding bits “10…0”
  - enough 0’s so that resulting total length is 64 bits less than a multiple of 512 bits
- Append L (original length of M), represented in 64 bits, to the padded message

- Footnote: the bytes of each 32-bit word are stored in little-endian order (LSB to MSB)

Padding… (cont'd)

- How many 0's if length of M =
- n * 512?
- n * 512 – 64?
- n * 512 – 65?

Preliminaries

- The four 32-bit words of the output (the *digest*) are referred to as **d0, d1, d2, d3**
- Initial values (in little-endian order)
  - **d0** = 0x67452301
  - **d1** = 0xEFCDAB89
  - **d2** = 0x98BADCFE
  - **d3** = 0x10325476
- The sixteen 32-bit words of each message block are referred to as **m0, …, m15**
  - (16*32 = 512 bits in each block)

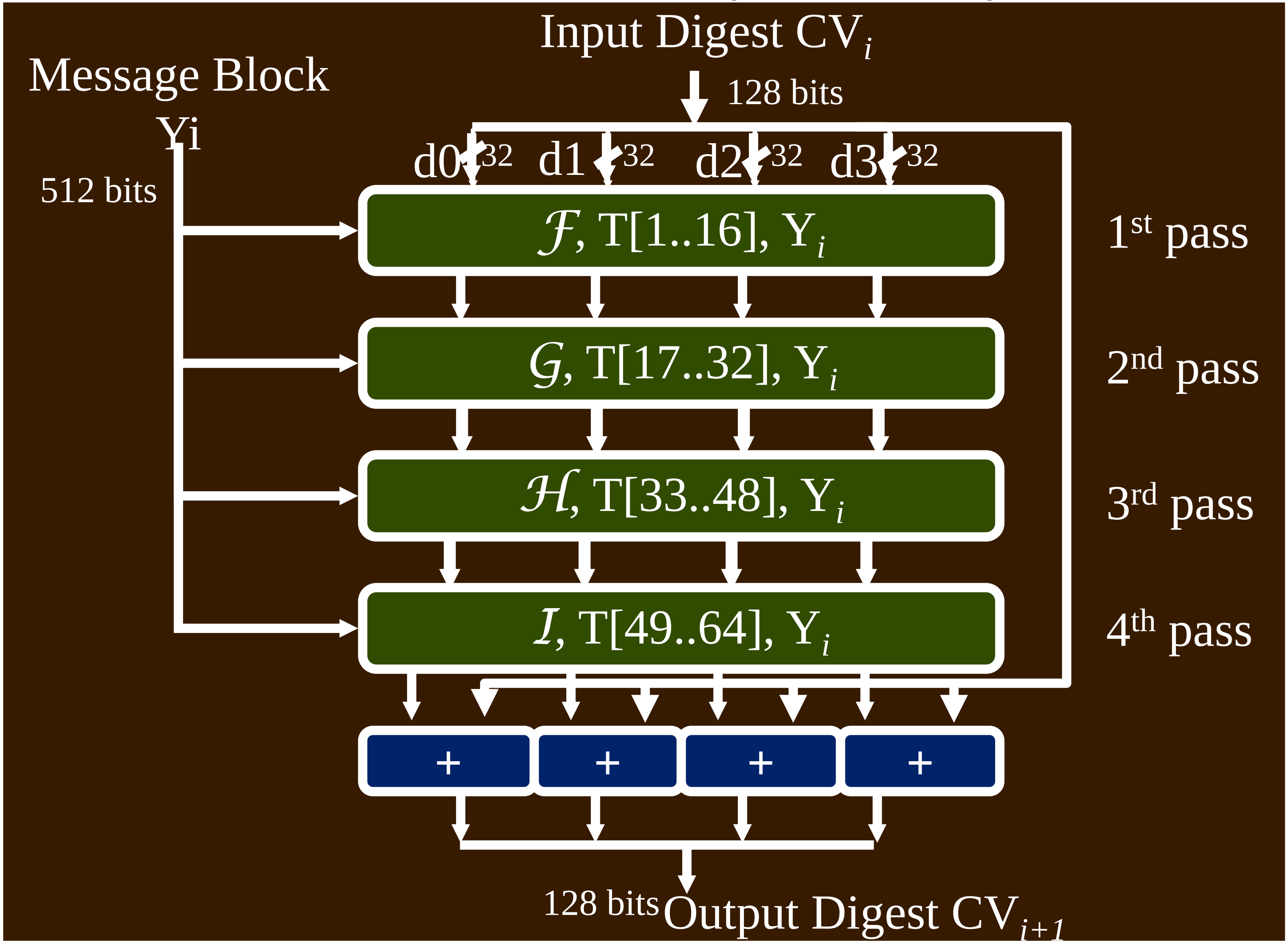
Notation

- $\sim x$ = bit-wise complement of x
- $x \wedge y$, $x \vee y$, $x \oplus y$ = bit-wise AND, OR, XOR of x and y
- $x \lll y$ = left circular shift of x by y bits
- $x+y$ = arithmetic sum of x and y (discarding carry-out from the msb)
- $\lfloor x \rfloor$ = largest integer less than or equal to x

Processing a Block -- Overview

- Every message block Yi contains 16 *32-bit words*:
  - $m_0\ m_1\ m_2\ \ldots\ m_{15}$
- A block is processed in 4 consecutive passes, each modifying the MD5 buffer $d_0, \ldots, d_3$.
  - Called $\mathcal{F}, \mathcal{G}, \mathcal{H}, \mathcal{I}$
- Each pass uses one-fourth of a 64-element table of constants, T[1…64]
  - $T[i] = \lfloor 2^{32} * abs(\sin(i)) \rfloor$, represented in 32 bits
  - Page 137
- Output digest = input digest + output of 4th pass

Overview (Cont'd)



1^{st} Pass of MD5

- $\mathcal{F}(x,y,z) \stackrel{\text{def}}{=} (x \wedge y) \vee (\sim x \wedge z)$
- 16 processing steps, producing $\mathbf{d_0..d_3}$ output:
  $\mathbf{d}_i = \mathbf{d}_j + (\mathbf{d}_k + \mathcal{F}(\mathbf{d}_l\ ,\ \mathbf{d}_m\ ,\ \mathbf{d}_n) + \mathbf{m}_o + \mathrm{T}_p) << s$
  – values of subscripts, in this order

i	*j*	*k*	*l*	*m*	*n*	*o*	*p*	*s*
0	1	0	1	2	3	0	1	7
3	0	3	0	1	2	1	2	12
2	3	2	3	0	1	2	3	17
1	2	1	2	3	0	3	4	22
0	1	0	1	2	3	4	5	7

2^{nd} Pass of MD5

- $\mathcal{G}(x,y,z) \stackrel{\text{def}}{=} (x \wedge z) \vee (y \wedge \sim z)$
- Form of processing (16 steps):
  $\mathbf{d}_i = \mathbf{d}_j + (\mathbf{d}_k + \mathcal{G}(\mathbf{d}_l, \mathbf{d}_m, \mathbf{d}_n) + \mathbf{m}_o + \mathrm{T}_p) << s$

i	*j*	*k*	*l*	*m*	*n*	*o*	*p*	*s*
0	1	0	1	2	3	1	17	5
3	0	3	0	1	2	6	18	9
2	3	2	3	0	1	11	19	14
1	2	1	2	3	0	0	20	20
0	1	0	1	2	3	5	21	5

3rd Pass of MD5

- $\mathcal{H}(x,y,z) \stackrel{\text{def}}{=} (x \oplus y \oplus z)$
- Form of processing (16 steps):
  $\mathbf{d}_i = \mathbf{d}_j + (\mathbf{d}_k + \mathcal{H}(\mathbf{d}_l, \mathbf{d}_m, \mathbf{d}_n) + \mathbf{m}_o + T_p) \ll s$

i	*j*	*k*	*l*	*m*	*n*	*o*	*p*	*s*
0	1	0	1	2	3	5	33	4
3	0	3	0	1	2	8	34	11
2	3	2	3	0	1	11	35	16
1	2	1	2	3	0	14	36	23
0	1	0	1	2	3	1	37	4

4^{th} Pass of MD5

- $\mathcal{I}(x,y,z) \stackrel{\text{def}}{=} y \oplus (x \vee \sim z)$
- Form of processing (16 steps):
  $\mathbf{d}_i = \mathbf{d}_j + (\mathbf{d}_k + \mathcal{I}(\mathbf{d}_l, \mathbf{d}_m, \mathbf{d}_n) + \mathbf{m}_o + T_p) \ll s$

i	*j*	*k*	*l*	*m*	*n*	*o*	*p*	*s*
0	1	0	1	2	3	0	49	6
3	0	3	0	1	2	7	50	10
2	3	2	3	0	1	14	51	15
1	2	1	2	3	0	5	52	21
0	1	0	1	2	3	12	53	6

- Output of this pass added to input CV

Logic of Each Step

- Within each pass, each of the 16 words of the message block is used exactly once
  - Pass 1, m_i are used in the order of i
  - Pass 2, in the order of $\rho 2(i)$, where $\rho 2(i)=(1+5i) \wedge 15$
  - Pass 3, in the order or $\rho 3(i)$, where $\rho 3(i)=(5+3i) \wedge 15$
  - Pass 4, in the order or $\rho 4(i)$, where $\rho 4(i)=7i \wedge 15$
- Each word of T[i] is used exactly once throughout all passes
- Number of bits s to rotate to get d_i
  - Pass 1, $s(d_0)=7$, $s(d_1)=22$, $s(d_2)=17$, $s(d_3)=12$
  - Pass 2, $s(d_0)=5$, $s(d_1)=20$, $s(d_2)=14$, $s(d_3)=9$
  - Pass 3, $s(d_0)=4$, $s(d_1)=23$, $s(d_2)=16$, $s(d_3)=11$
  - Pass 4, $s(d_0)=6$, $s(d_1)=21$, $s(d_2)=15$, $s(d_3)=10$

(In)security of MD5

- A few recently discovered methods can find collisions in a few hours
  - A few collisions were published in 2004
  - Can find many collisions for 1024-bit messages
  - In 2005, two X.509 certificates with different public keys and the same MD5 hash were constructed
    - This method is based on differential analysis
    - 8 hours on a 1.6GHz computer
    - Much faster than birthday attack