Network Security: Hashes

Henning Schulzrinne
Columbia University, New York
schulzrinne@cs.columbia.edu

Columbia University, Fall 2000

Last modified October 5, 2000

Hashes

- hash $\equiv$ *message digest*
- one-way function: $d = h(m)$, but no $h'(d) = m$
- cannot find message with given digest
- cannot find m_1, m_2 where $d_1 = d_2$
- cannot find different m_2 with same digest (given some m_1)
- arbitrary-length message $\longrightarrow$ fixed-size hash
- different: checksums (CRC):
  - fast
  - easy to compute two messages
  - with same 'hash', protect against noisy channels

Randomness

- for 1000 inputs, any bit in the outputs ’1’ half the time
- each output: 50% ’1’ bits
- similar inputs ➠ uncorrelated outputs (half the bits should differ)

note: efficiency **not** important

Random vs. Random

- random numbers for simulation ($\rightarrow$ Knuth, Bratley/Fox/Schrage):
  - sequence
  - no short cycles
  - average = 0.5, variance, ...
  - completely predictable, repeatable
  - "white noise": flat frequency spectrum
  - must be very efficient (multiply, add)
- random numbers for picking keys:
  - unpredictable by outside observer
  - two processes at the same time ➠ different number
  - use audio, video
  - efficiency not very important

The Birthday Problem (Approximate)

two people with same birthday in room

- n inputs (humans)
- $k = 365$ possible outputs (birthdays)
- $n \cdot (n-1)/2$ pairs of inputs
- check for each pair: $1/k$ of both humans having same birthday
- ➠ $k/2$ pairs for 50% matching probability
- ➠ $n \approx \sqrt{k}$ for 50% chance
- warning: simplified, does not hold for large n!
- or:

$$p_{\text{same}} = \frac{n(n-1)}{2} \cdot \frac{1}{k}$$

Example: $n = 23$ ➠ $p_{\text{alldifferent}} = 0.31$

Birthday Problem, Correct

compute probability of *different* birthdays ➠ no sample twice

$$\frac{\text{sampling without replacement}}{\text{sampling with replacement}}$$

- random sample of n people (birthdays) taken from k (365 days)
- k^n samples with replacement
- $(k)_n = k(k-1)\cdots(k-n+1)$ samples without replacement

Birthday Problem, Correct

probability of no repetition:

$$p = \frac{(k)_n}{k^n} = \frac{k(k-1)\cdots(k-n+1)}{k^n} \approx 1 - \frac{n(n-1)}{2k}$$

$$\begin{aligned} p &= \frac{(365)_n}{365^n} = \frac{365 \cdot 364 \cdots (365-n+1)}{365 \cdot 365 \cdots 365} \\ &= 1 \cdot \frac{365-1}{365} \cdot \frac{365-2}{365} \cdots \frac{365-n+1}{365} \\ &= \left(1 - \frac{1}{365}\right)\left(1 - \frac{2}{365}\right) \cdots \left(1 - \frac{n+1}{365}\right) \end{aligned}$$

Example: $n = 23 \Rightarrow p = 0.43$

How Many Bits for Hash?

- m bits, takes $2^{m/2}$ to find two with the same hash
- 64 bits ➠ 2^{32} messages to search
- note: different from picking h and finding m

Example:

- Alice (secretary: Bob) wants to fire Fred
- Bob is Fred's friend
- Bob composes two messages (same hash), one to be read, the other sent
- generate lots of similar messages: 2 choices of wording in 32 places ➠ 2^{32} messages

Hash Functions

RFC: `http://www.normos.org`

name	inventor	docs	entity	hash	sub-stages
MD2	Ron Rivest	RFC 1319	bytes	128	1
MD4	Ron Rivest	RFC 1320	32-bit	128	3
MD5	Ron Rivest	RFC 1321	32-bit	128	4
SHS	NIST		32-bit	160	

Authentication

Alice		Bob
r_A	$\longrightarrow$	
	$\longleftarrow$	$\mathrm{MD}(K_{AB} \vert r_A)$
	$\longleftarrow$	r_B
$\mathrm{MD}(K_{AB} \vert r_B)$	$\rightarrow$	

only need to compare result, not decrypt

Computing a MIC with a Hash

- can't just compute MD(m)
- MIC: MD($K_{AB}|m_1$)
- but: Trudy can get MIC of m_2, given m_1
  - 512-bit blocks, append (message length, pad)
  - allow to concatenate padding, plus additional message
  - ➠ put secret at end of message
  - or: use only half the bits of MD ➠ can't continue message

Encryption: One-Time Pad

- $b_1 = \text{MD}(K_{AB}|\text{IV})$
- $b_2 = \text{MD}(K_{AB}|b_1)$
- $b_i = \text{MD}(K_{AB}|b_{i-1})$
- then: b used as one-time pad
- can't use same OTP twice ➠ use IV
- any OTP has privacy only: if plaintext known, can encrypt anything
- separate integrity function

Encryption: Mixing in the Plaintext

- similar to cipher feedback mode (CFB)
- break message into MD-length chunks p_i

$$
\begin{array}{rclrcl}
b_1 & = & \mathrm{MD}(K_{AB}|IV) & c_1 & = & p_1 \oplus b_1 \\
b_2 & = & \mathrm{MD}(K_{AB}|c_1) & c_2 & = & p_2 \oplus b_2 \\
\vdots & & & & & \\
b_i & = & \mathrm{MD}(K_{AB}|c_{i-1}) & c_i & = & p_i \oplus b_i
\end{array}
$$

Using Secret Key for a Hash

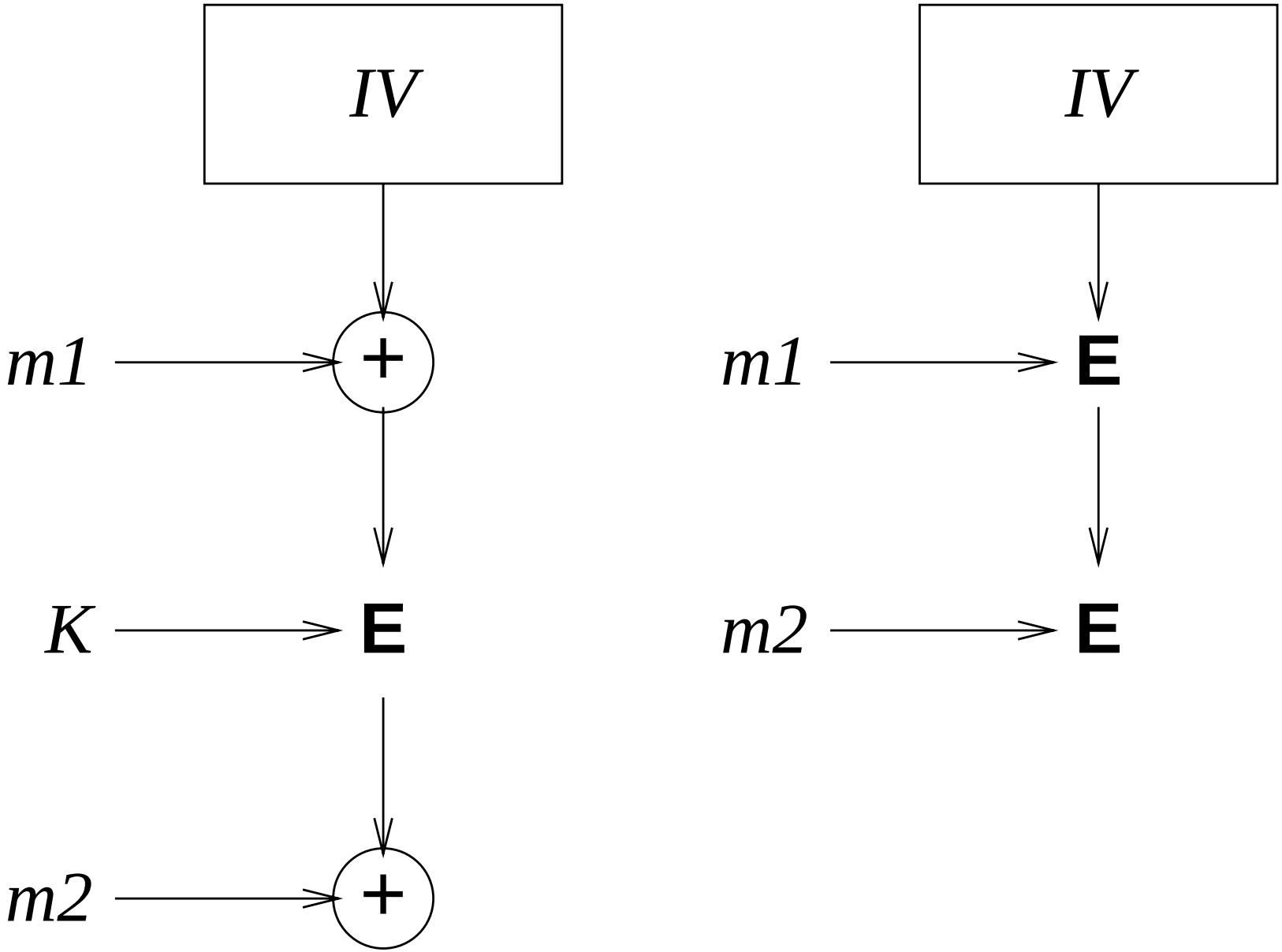
Unix password algorithm:

- compute hash of user password, compare to hash of password typed
- first 8 bytes of password $\rightarrow$ secret key
- encrypt 0 with DES-like algorithm
- *salt:* 12-bit random number ➠ determines bits to duplicate in mangler when expanding 32 to 48 bits
- salt stored with hashed result

Hashing Large Messages Using Encryption

- key length k, block length b
- divide message into k-bit chunks $m_1, m_2, \ldots$
- first, encrypt a constant with m_1 as key
- encrypt output with m_2 as key
- ➠ output $b(= 64)$ too short
- generate 2nd 64-bit quantity by reversing chunk order
- or: use different initial constants

Hashing Large Messages


IV
m1
+
K
E
m2
+
IV
m1
E
m2
E

MD2

128-bit message digest

- arbitrary number of bytes
- pad to multiple of 16 bytes
- append MD2 checksum to end
- process whole message

MD2 Checksum

- m_{nk}: byte nk of message
- $c_n := \pi(m_{nk} \oplus c_{n-1}) \oplus c_n$
- π: $0 \to 41, 1 \to 46, \ldots$
- *supposedly* based on π

MD2 Final Pass

- operate on 16-byte chunks
- 48-byte quantity q: (current digest $\mid$ chunk $\mid$ digest $\oplus$ chunk)
- 18 passes over q
- $c_n = c_n \oplus \pi(c_{n-1})$ for $n = 0, \ldots, 47$; $c_{-1} = 0$
- $c_{-1} = (c_{47} + \text{pass \#}) \bmod 256$
- after pass 17, use first 16 bytes as new digest
- minimal storage: checksum, 48 bytes

MD4

- any number of bits
- operates on 32-bit quantities
- pad to multiple of 512 bits (16 32-bit words):
  100…, message length in bits

- digest = f(512-bit blocks, previous digest or initial constant)
- 1 stage = three mangling passes
- 16 message words $m_0, m_1, \ldots, m_{15}$
- 4 message digest words d_0, d_1, d_2, d_3
- $d_0 = 01234567_{16}, d_1 = 89abcdef_{16}, d_2 = fedcba98_{16}, \ldots$

MD4 Operations

- $\lfloor x \rfloor$
- $\sim$: bitwise complement
- $x \wedge y$: bitwise and
- $x \vee y$: bitwise or
- $x + y$: addition mod 2^{32}
- $x \hookleftarrow y$: rotate left y bits

MD4 Pass 1: Selection

- selection function: $F(x, y, z) = (x \wedge y) \vee (\sim x \wedge z)$
- if bit x_n is 1: pick $F_n = y_n$, otherwise z_n

$$d_{(-i)\wedge 3} = (d_{(-i)\wedge 3} + F(d_{(1-i)\wedge 3}, d_{(2-i)\wedge 3}, d_{(3-i)\wedge 3}) + m_i) \hookleftarrow S_1(i \wedge 3)$$

where $S_1(i) = 3 + 4i$, $-1_{10} = 1111_2, -2_{10} = 1110_2, \ldots$

$$\begin{aligned}
d_0 &= (d_0 + F(d_1, d_2, d_3) + m_0) \hookleftarrow 3 \\
d_1 &= (d_3 + F(d_0, d_1, d_2) + m_1) \hookleftarrow 7 \\
d_2 &= (d_2 + F(d_3, d_0, d_1) + m_2) \hookleftarrow 11 \\
d_3 &= (d_1 + F(d_2, d_3, d_0) + m_3) \hookleftarrow 15 \\
d_0 &= (d_0 + F(d_1, d_2, d_3) + m_4) \hookleftarrow 3
\end{aligned}$$

MD4 Pass 2: Majority

- $G(x, y, z) = (x \wedge y) \vee (x \wedge z) \vee (y \wedge z)$
- $G_n = 1$ iff 2 out of 3 bits x_n, y_n, z_n are 1
- constant $\lfloor 2^{30}\sqrt{2} \rfloor = 5a827999_{16}$

$$d_{(-i)\wedge 3} = (d_{(-i)\wedge 3} + G(d_{(1-i)\wedge 3}, d_{(2-i)\wedge 3}, d_{(3-i)\wedge 3}) + m_{X(i)} + 5a827999_{16}) \hookleftarrow S_2(i \wedge 3)$$

where $X(i)$ eXchange bits (2,3) with (0,1); $S_2 = 3, 5, 9, 13, 3, \ldots$

$$\begin{aligned}
d_0 &= (d_0 + G(d_1, d_2, d_3) + m_0 + 5a827999_{16}) \hookleftarrow 3 \\
d_1 &= (d_3 + G(d_0, d_1, d_2) + m_4 + 5a827999_{16}) \hookleftarrow 5 \\
d_2 &= (d_2 + G(d_3, d_0, d_1) + m_8 + 5a827999_{16}) \hookleftarrow 9 \\
d_3 &= (d_1 + G(d_2, d_3, d_0) + m_{12} + 5a827999_{16}) \hookleftarrow 13 \\
d_0 &= (d_0 + G(d_1, d_2, d_3) + m_1 + 5a827999_{16}) \hookleftarrow 3
\end{aligned}$$

MD4 Pass 3: Ex-Or

- $H(x, y, z) = x \oplus y \oplus z$
- constant $\lfloor 2^{30}\sqrt{3} \rfloor = 6ed9eba1_{16}$

$$d_{(-i)\wedge 3} = (d_{(-i)\wedge 3} + H(d_{(1-i)\wedge 3}, d_{(2-i)\wedge 3}, d_{(3-i)\wedge 3}) + m_{R(i)} + 6ed9eba1_{16}) \hookleftarrow S_3(i \wedge 3)$$

where $R(i)$ Reverses bits in i; $S_3 = 3, 9, 11, 15$

$$\begin{aligned}
d_0 &= (d_0 + H(d_1, d_2, d_3) + m_0 + 6ed9eba1_{16}) \hookleftarrow 3 \\
d_1 &= (d_3 + H(d_0, d_1, d_2) + m_4 + 6ed9eba1_{16}) \hookleftarrow 9 \\
d_2 &= (d_2 + H(d_3, d_0, d_1) + m_8 + 6ed9eba1_{16}) \hookleftarrow 11 \\
d_3 &= (d_1 + H(d_2, d_3, d_0) + m_{12} + 6ed9eba1_{16}) \hookleftarrow 15 \\
d_0 &= (d_0 + H(d_1, d_2, d_3) + m_2 + 6ed9eba1_{16}) \hookleftarrow 3
\end{aligned}$$

MD5

- MD4, MD5: same message padding, blocking, initial d_i
- MD4: 3 passes ➠ MD5: 4 passes
- different functions, different shifts
- MD4: two constants for all m_i ➠ MD5: different

$$T_i = \lfloor 2^{32} |\sin i| \rfloor$$

MD5 Pass 1: Selection

$$\begin{aligned} d_{(-i)\wedge 3} &= (d_{(-i)\wedge 3} + F(d_{(1-i)\wedge 3}, d_{(2-i)\wedge 3}, d_{(3-i)\wedge 3}) + m_i + T_{i+1}) \\ &\quad \hookleftarrow S_1(i \wedge 3) \end{aligned}$$

where $S_1 = 7 + 5i$:

$$\begin{aligned} d_0 &= (d_0 + F(d_1, d_2, d_3) + m_0 + T_1) \hookleftarrow 7 \\ d_1 &= (d_3 + F(d_0, d_1, d_2) + m_1 + T_2) \hookleftarrow 12 \\ d_2 &= (d_2 + F(d_3, d_0, d_1) + m_2 + T_3) \hookleftarrow 17 \\ d_3 &= (d_1 + F(d_2, d_3, d_0) + m_3 + T_4) \hookleftarrow 22 \\ d_0 &= (d_0 + F(d_1, d_2, d_3) + m_4 + T_5) \hookleftarrow 7 \end{aligned}$$

MD5 Pass 2: Selection'

Selection function:

$$G(x, y, z) = (x \wedge z) \vee (y \wedge \sim z)$$

$$\begin{aligned} d_{(-i)\wedge 3} &= (d_{(-i)\wedge 3} + F(d_{(1-i)\wedge 3}, d_{(2-i)\wedge 3}, d_{(3-i)\wedge 3}) + m_{(5i+1)\wedge 15} + T_{i+17}) \\ &\quad \hookleftarrow S_2(i \wedge 3) \end{aligned}$$

where $S_2(i) = i(i+7)/2 + 5$:

$$\begin{aligned} d_0 &= (d_0 + G(d_1, d_2, d_3) + m_1 + T_{17}) \hookleftarrow 5 \\ d_1 &= (d_3 + G(d_0, d_1, d_2) + m_6 + T_{18}) \hookleftarrow 9 \\ d_2 &= (d_2 + G(d_3, d_0, d_1) + m_{11} + T_{19}) \hookleftarrow 14 \\ d_3 &= (d_1 + G(d_2, d_3, d_0) + m_0 + T_{20}) \hookleftarrow 20 \\ d_0 &= (d_0 + G(d_1, d_2, d_3) + m_5 + T_{21}) \hookleftarrow 5 \end{aligned}$$

MD5 Pass 3: Ex-Or

$H(x, y, z) = x \oplus y \oplus z$ as in MD4

$$d_{(-i)\wedge 3} = (d_{(-i)\wedge 3} + H(d_{(1-i)\wedge 3}, d_{(2-i)\wedge 3}, d_{(3-i)\wedge 3}) + m_{(3i+5)\wedge 15}) + T_{i+33}) \hookleftarrow S_3(i \wedge 3)$$

where $S_3(i) = 4, 11, 16, 23, 4, \ldots$

$$\begin{aligned}
d_0 &= (d_0 + H(d_1, d_2, d_3) + m_5 + T_{33}) \hookleftarrow 4 \\
d_1 &= (d_3 + H(d_0, d_1, d_2) + m_8 + T_{34}) \hookleftarrow 11 \\
d_2 &= (d_2 + H(d_3, d_0, d_1) + m_{11} + T_{35}) \hookleftarrow 16 \\
d_3 &= (d_1 + H(d_2, d_3, d_0) + m_{14} + T_{36}) \hookleftarrow 23 \\
d_0 &= (d_0 + H(d_1, d_2, d_3) + m_1 + T_{37}) \hookleftarrow 4
\end{aligned}$$

MD5 Pass 4: Ex-or'

$$I(x, y, z) = y \oplus (x \vee \sim z)$$

$$\begin{aligned} d_{(-i)\wedge 3} &= (d_{(-i)\wedge 3} + I(d_{(1-i)\wedge 3}, d_{(2-i)\wedge 3}, d_{(3-i)\wedge 3}) + m_{(7i)\wedge 15} + T_{i+49}) \\ &\quad \hookleftarrow S_4(i \wedge 3) \end{aligned}$$

where $S_4(i) = 6, 10, 15, 21, 6, \ldots$

$$\begin{aligned} d_0 &= (d_0 + I(d_1, d_2, d_3) + m_0 + T_{49}) \hookleftarrow 6 \\ d_1 &= (d_3 + I(d_0, d_1, d_2) + m_7 + T_{50}) \hookleftarrow 10 \\ d_2 &= (d_2 + I(d_3, d_0, d_1) + m_{14} + T_{51}) \hookleftarrow 15 \\ d_3 &= (d_1 + I(d_2, d_3, d_0) + m_5 + T_{52}) \hookleftarrow 21 \\ d_0 &= (d_0 + I(d_1, d_2, d_3) + m_{12} + T_{53}) \hookleftarrow 6 \end{aligned}$$

Secure Hash Standard (SHS)

- 64 bits $\longrightarrow$ 160 bits
- similar to MD5
- 5 passes
- 5 32-bit digest words ➠ digest $A|B|C|D|E$:

$$\begin{aligned} A &= 67452301_{16} \\ B &= efcdab89_{16} \\ C &= 98badcfe_{16} \\ D &= 10325376_{16} \\ E &= c3d2e1f0_{16} \end{aligned}$$

SHS

- start with 512-bit block
- fill 4 more blocks with recursion ➠ 80 32-bit words:
- word $W_n = W_{n-3} \oplus W_{n-8} \oplus W_{n-14} \oplus W_{n-16} \hookleftarrow 1$
- modify digest:

$$\begin{aligned} A' &= E + (A \hookleftarrow 5) + W_t + K_t + f(t, B, C, D) \\ B' &= A \\ C' &= B \hookleftarrow 30 \\ D' &= C \\ E' &= D \end{aligned}$$

- $K_t = \lfloor 2^{30}\sqrt{x} \rfloor$, where $x = \{2, 3, 5, 10\}[t/20]$

SHS

Function $f()$:

$$
\begin{array}{lcll}
f(t,B,C,D) & = & (B \wedge C) \vee (\sim B \wedge D) & \text{for } 0 \leq t \leq 19 \\
f(t,B,C,D) & = & B \oplus C \oplus D & \text{for } 20 \leq t \leq 39 \\
f(t,B,C,D) & = & (B \wedge C) \vee (B \wedge D) \vee (C \wedge D) & \text{for } 40 \leq t \leq 59 \\
f(t,B,C,D) & = & B \oplus C \oplus D & \text{for } 60 \leq t \leq 79
\end{array}
$$