TOPIC: 6.857

DATE: 2/18/14

FILE UNDER:

PAGE: L4.1

Admin: Pset #1 due 2/24. Pset #2 out 2/24. (new groups for pset #2)

Project idea: AEG - Automatic Exploit Generation CACM 2/14 p. 74-84

Discuss: (The Tech) Tidbit students / letter / MIT legal aid.
2/18/14

Today: Cryptographic hash functions

- definitions
- random oracle model
- desirable properties
- applications
- Keccak (SHA-3) overview

(Cryptographic) Hash functions

A cryptographic hash function h maps bit-strings of arbitrary length to a fixed-length output in an efficient, deterministic, public, "random" manner:

$$h : \underbrace{\{0,1\}^*}_{\text{all strings (of } \underline{\text{any}} \text{ length} \geq 0)} \longrightarrow \underbrace{\{0,1\}^d}_{\text{all strings of length } d}$$

Sometimes called a "message digest" function.

Typical output lengths are $d = 128, 160, 256, 512$ bits.

No secret key. Anyone can compute h from its public description. Computation is efficient (poly-time).

Examples:

	d	note
MD4	128	"broken" wrt CR
MD5	128	"broken" wrt CR
SHA-1	160	? CR?
SHA-256	256	
SHA-512	512	
SHA-3 (coming!)	224, 256, 384, 512	

"Ideal" Hash Function: Random Oracle (RO)

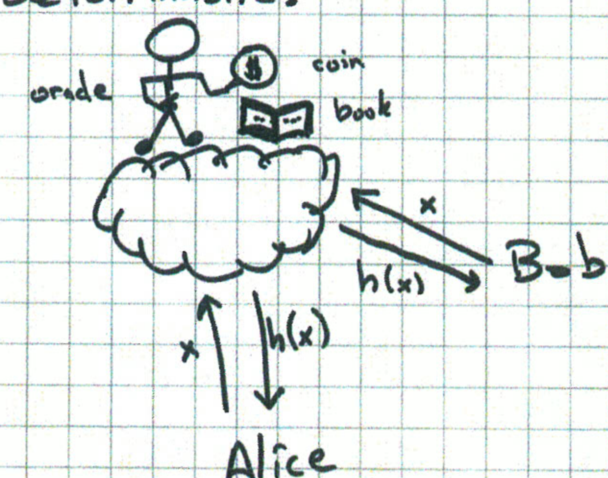
- Theoretical model – not achievable in practice

Oracle ("in the sky")

- receives inputs x & returns output $h(x)$, for any $x \in \{0,1\}^*$. $|h(x)| = d$ bits.
- On input $x \in \{0,1\}^*$:

  if x not in book:
  - flip coin d times to determine $h(x)$
  - record $(x, h(x))$ in book

  else: return y where (x,y) in book.
- Gives random answer every time, but uses book to record previous answers, so h is deterministic.



TOPIC:

DATE:

FILE UNDER:

PAGE L8.9

Many cryptographic schemes are proved secure in ROM ("Random Oracle Model"), which assumes existence of RO. Then RO is replaced by conventional hash function (e.g. SHA-256) in practice, which is hopefully "pseudorandom enough" (!?).

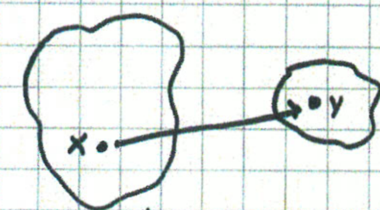
Hash function desirable properties:

OW

① "One-way" (pre-image resistance)

"Infeasible", given $y \in_R \{0,1\}^d$ to find any x s.t. $h(x) = y$ (x is a "pre-image" of y)

Actually, the correct definition is that is hard for an adversary, given y=h(x) (where x was picked uniformly at random from {0,1}^n) to find any x' such that h(x')=y.



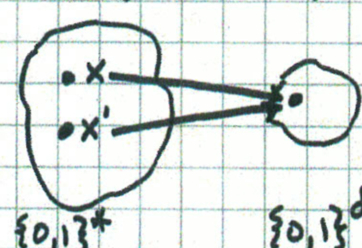
$h: \{0,1\}^* \longrightarrow \{0,1\}^d$

(Note that a "brute-force" approach of trying x's at random requires $\Theta(2^d)$ trials (in ROM).)

CR

② "Collision-resistance" (strong collision resistance)

"Infeasible" to find x, x' s.t. $x \neq x'$ and $h(x) = h(x')$ (a "collision")



$h(x) = h(x')$

(In ROM, requires trying about $2^{d/2}$ x's $(x_1, x_2, \ldots)$ before a pair x_i, x_j colliding is found. (This is the "birthday paradox".))

TOPIC:

DATE:

FILE UNDER:

Note that collisions are unavoidable since

$$|\{0,1\}^*| = \infty$$
$$|\{0,1\}^d| = 2^d$$

Birthday paradox detail:

If we hash $x_1, x_2, \ldots, x_n$ (distinct strings)

then

$$E(\#\text{collisions}) = \sum_{i \neq j} \Pr(h(x_i) = h(x_j))$$
$$= \binom{n}{2} \cdot 2^{-d} \qquad [\text{if } h \text{ "uniform"}]$$
$$\approx \frac{n^2 \cdot 2^{-d}}{2}$$

This is ≥ 1 when $n \geq 2^{(d+1)/2} \approx 2^{d/2}$

The birthday paradox is the reason why hash function outputs are generally twice as big as you might naively expect; you only get 80 bits of security (w.r.t. CR) for a 160-bit output.

With some tricks, memory requirements can be dramatically reduced.

TOPIC:

DATE:

FILE UNDER:

TCR

③ "Weak collision resistance" (target collision resistance, 2^{nd} pre-image resistance)

"Infeasible", given $x \in \{0,1\}^*$, to find $x' \neq x$ s.t. $h(x) = h(x')$.

Like CR, but one pre-image given & fixed.

(In ROM, can find x' in time $\Theta(2^d)$ (as for OW, since knowing x doesn't help in ROM to find x').)

PRF

④ Pseudo-randomness

"h is indistinguishable under black-box access from a random oracle"

(To make this notion workable, really need a family of hash functions, one of which is chosen at random. A single, fixed, public hash function is easy to identify...)

NM

⑤ Non-malleability

"Infeasible", given $h(x)$, to produce $h(x')$ where x and x' are "related" (e.g. $x' = x+1$).

These are informal definitions...

TOPIC:
DATE:
FILE UNDER:

Theorem: If h is CR, then h is TCR.
(But converse doesn't hold.)

Theorem: h is OW $\not\Leftrightarrow$ h is CR
(neither implication holds)
But if h "compresses", then CR $\Rightarrow$ OW.

Hash function applications

1. Password storage (for login)
   - Store h(PW), not PW, on computer
   - when user logs in, check hash of his PW agaist table.
   - Disclosure of h(PW) should not reveal PW (or any equivalent pre-image)
   - Need OW

2. File modification detector
   - For each file F, store h(F) securely (e.g. on off-line DVD)
   - Can check if F has been modified by recomputing h(F)
   - need WCR (aka TCR)
     (Adversary wants to change F but not h(F).)

   - Hashes of downloadable software = equivalent problem.