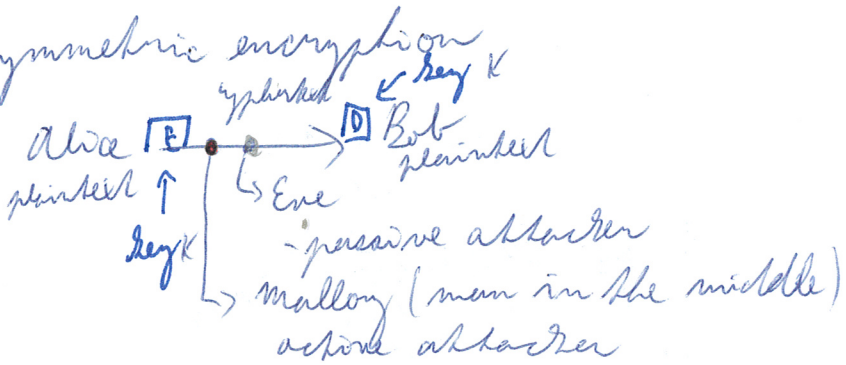
Goal: cryptography — cryptographic primitives
protocols
implementation

understand existing protocols
design of protocols

Symmetric encryption



Kerckhoffs principle
"Secret should be the key not the algorithm"

reason:
① good ciphers are hard to find
② well known ciphers are well analyzed
③ Keys are easier changed when compromised

$E: \{0,1\}^m \times \{0,1\}^n \to \{0,1\}^m$
plaintext key ciphertext

$E(x,K) = y$

$D: \{0,1\}^m \times \{0,1\}^n \to \{0,1\}^m$

$D(y,K) = x$

$\forall K \forall x \quad D(E(x,K),K) = x$

for fixed K: E_K as a permutation on $\{0,1\}^m$
we want "random"

Ceasar cipher
messages: $\{0,\dots,25\} \cong \mathbb{Z}_{26}$
Keys $\mathbb{Z}_{26}$
$E(x,K) = x + K \quad D(y,K) = y - K$
weak because Keys are limited

Asymmetric cipher $D(E(x, k_E), k_D)$

Alice → [E] ——— [D] → Bob

(k_E, k_D) ← Key pair generation

Multi-party comm. network

- n pairs of keys
- k_E is public
- k_D are private

Catch: Key distribution

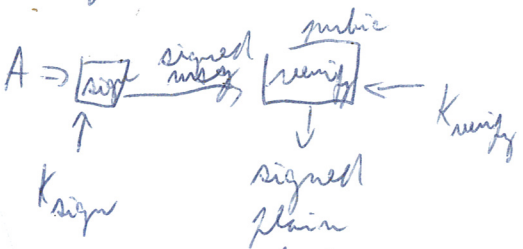
Hash function

$h : \{0;1\}^* \to \{0;1\}^b$

"random"

1. impossible to invert
2. impossible to find collision
   $x \neq x' : h(x) = h(x')$

Signature scheme



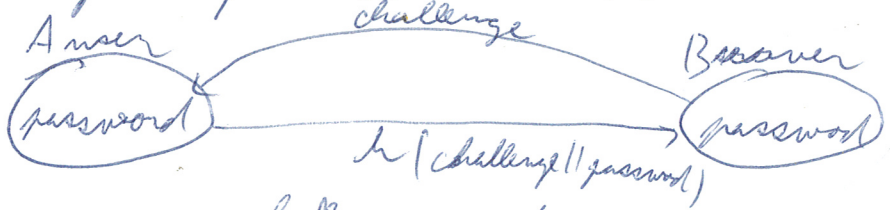
n pairs of keys
- k_E is private
- k_D is public

Task: Authentication protocol
Just a one-way
hash please

Applications

1) signatures: A wants to sign K
   - send x in plaintext
   - $E(h(x), k_{sign}) \to D(-, k_{ver})$

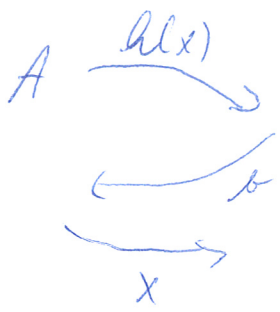
2) challenge - response communication



challenge: random X

Random generation - unpredictable, cannot be influenced

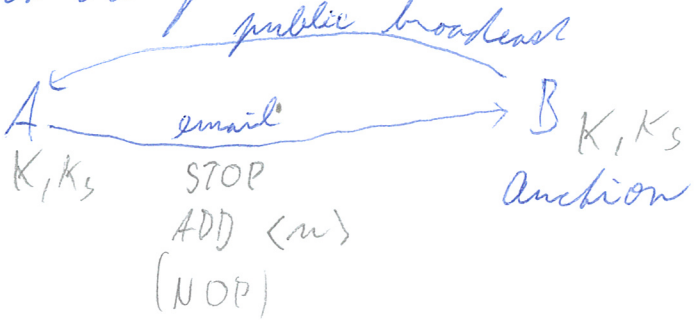
Tossing a coin over a phone call



$A: Y \in \{0,1\}^{128}$

bit commitment protocol

Auction protocol



secrecy?
- against whom
- for how long

message authentication code (MAC)

signature ← $h(K_S \,||$ encrypted part $||$ session ID)

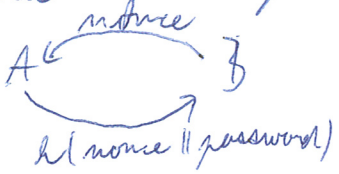
fixed size { sequential #, nonce, comand, padding } E_K

Kinds of attacks
- known ciphertext → recover plaintext
- known plaintext → recover Key
- chosen plaintext
- distinguishing attack

How to measure strength of primitives
b: security level (bits) ... attacker requires $\geq 2^b$ operations to break the protocol

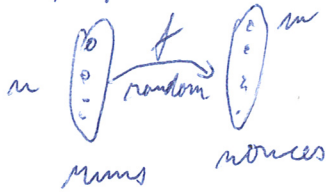
Birthday attacks

challenge-response scheme



$2^{b/2}$ → likely to repeat nonces

23 people 366 days → $p(2 \text{ have same bd.}) \geq \frac{1}{2}$



$$p(f \text{ is injective}) = \frac{\#\text{ injective functions}}{\#\text{ all functions}} = \frac{(m)\cdot(m-1)\cdots(m-n+1)}{m^n} =$$

$$= \frac{m}{m}\cdot\frac{m-1}{m}\cdot\frac{m-2}{m}\cdot\frac{m-n+1}{m} \qquad 1-x \approx e^{-x}$$

$$1\cdot\left(1-\frac{1}{m}\right)\cdot\left(1-\frac{2}{m}\right)\cdots \approx 1\cdot e^{-\frac{1}{m}}\cdot e^{-\frac{2}{m}}\cdot e^{-\frac{n-1}{m}} =$$

$$= e^{-\frac{1+2+3\cdots n-1}{m}} \approx e^{-\frac{n^2}{2m}}$$

$$e^{-\frac{n^2}{2m}} = \frac{1}{2}$$

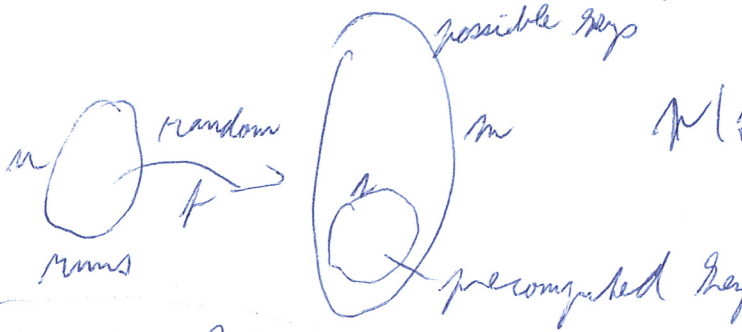
$$-\frac{n^2}{2m} = \ln\frac{1}{2}$$

$$\frac{n^2}{m} = -2\ln\frac{1}{2} \doteq 1{,}38 \Rightarrow n^2 \approx m$$

A → B $E_{asym}(K, K^B_{pub})$

$E_{sym}\ K_{guess}$

$E_{asym}(K_{guess}, K^B_{pub})$ → precompute or table



$$p(f \text{ avoids subset}) = \left(1-\frac{1}{m}\right)^n \approx e^{-\frac{kn}{m}}$$

$$kn \sim m$$

$$k := \sqrt{m}$$

$$n := \sqrt{m}$$

security level $\leq \frac{\#\text{ key bits}}{2}$

One time Pad (Vernam cipher)

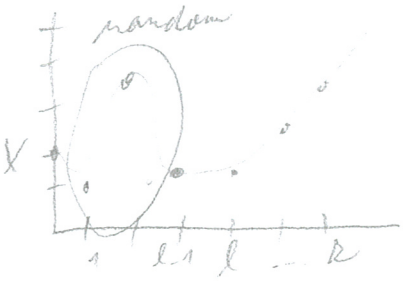
message: $x \in \{0,1\}^n$

key: $K \in_R \{0,1\}^n$ - must be secret (only one) (perfect security)

ciphertext: $y = x \oplus K$ → independent uniform random bits

decrypt: $x = y \oplus K$

continuation (4)



Polynomials of degree < l
unique poly p shares: $s_i := p(i)$

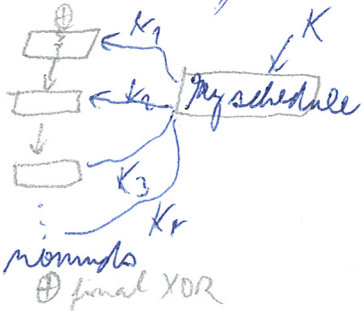
Symmetric ciphers → stream
↳ block
fixed sized blocks
$E: \{0,1\}^b \times \{0,1\}^k \to \{0,1\}^b$
$E_K: \{0,1\}^b \to \{0,1\}^b$ bijection/permutation on set of block values

$K \to$ gen $\to$ IV
keystream
$X \to \oplus \to Y$
$D_K = E_K^{-1}$
$E_K(X) = E_K(X)$
$E_{K_1}(E_{K_2}(X)) = E_{K_1}(E_{K_2}(X))$

Security of block cipher

Game ⇔ ∄ a distinguisher with $Pr[success] \geq \frac{2}{3}$ and ~~num~~ time security level $< 2^{\text{security level}}$
In real constructions: almost always even permutations

Iterated cipher

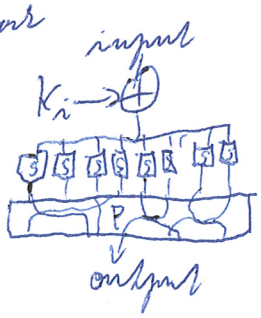


rounds
⊕ final XOR

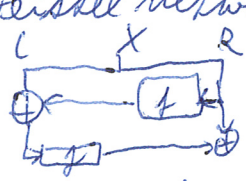
Substitution permutation network

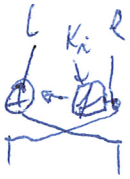
Round: - S-boxes confusion
- P-box diffusion
- mixing round key

round is invertible
→ again SPN
- reverse schedule of keys
- permute round keys
inverse of P, inverse S_i



Feistel network

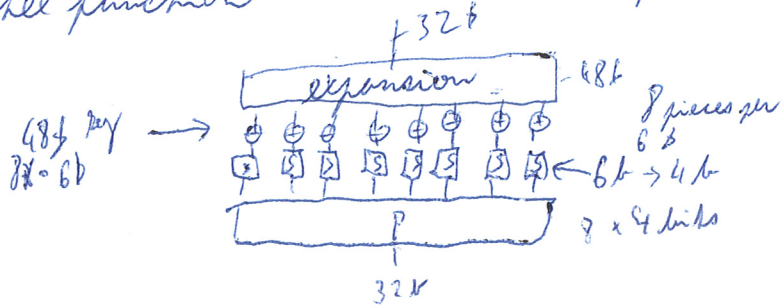




1 round of FN

f needn't be injective
inverse of FN is FN

Feistel function



DES (data encryption standard)
- 1970s IBM & US government NSA
- 64 bit blocks, 56 bit keys
- S boxes were replaced by NSA
- Feistel network
- all S boxes are different
- expansion - each block takes 1b from each side

Generalise one time pad

Group $(G, +, 0, -)$

$x, y, K \in G$ $\quad K \in_R G$ $\quad \mathbb{Z}_n = \{0, \dots, n-1\}$ mod n

$E(x, K) = x + K$

$D(y, K) = y - K = y + (-K)$

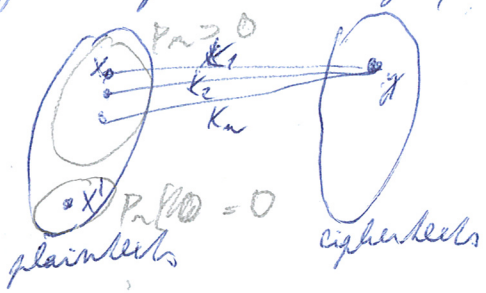
D_f: a cipher is perfectly secure $\equiv \forall x \forall y \; Pr(E(x,K) = y)$ is constant

$Pr[OTP] = \frac{1}{|G|}$ ... it's perfectly secure

usefulness of OTP
- never repeat keys! → if same keys $y_1 \oplus y_2 = x_1 \oplus x_2$ - bit of info
- code book
- replace randomness by pseudorandomness
- attacker toggles $y[i]$ → toggles $x[i]$

Theorem: if #keys < #messages, then the cipher is not perfectly secure

Proof:



x' can't be obtained by a key
⇒ probability is not uniform y

Information theoretic security (Shannon security)

Secret sharing (splitting)

① S_1 = random
$S_2 = X \oplus S_1$
$S_1 \oplus S_2 \to X$ **(2,2)**

② $S_1 \dots S_{K-1}$ = random **(K,K)** $\bigoplus S_i = X$
$S_K = S_1 \oplus \dots S_{K-1} \oplus X$

D_f: (k,l) threshold scheme
split X to shares $S_1 \dots S_K$
s.t. any l shares give X
with $< l$ shares no info on X

③ **(K,2)** finite field F
hooking for $f(x) = ax + b$
$f(0) = X$
$f(1) \in_R F$ $\quad S_1 \dots S_K$ $\quad S_i := f(i)$

④ **(k,l)** ... Polynomials

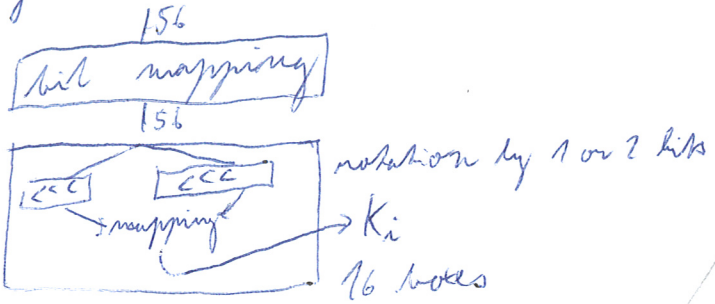
Ⓐ if $x_1 \dots x_d$ are all the roots of p then
$p(x) = (x - x_1)(x - x_2) \dots (x - x_d) \cdot q(x) \in$ no roots
⇒ nonzero p of deg d has at most d roots

Ⓑ Let p, q polys of deg $< d$ with the same graph ⇒ $p = q$
Proof: $r := p - q$ deg $< d$ ⇒ $x_1 \dots x_d$ are roots of r ⇒ $r = 0$ ⇒ $p = q$

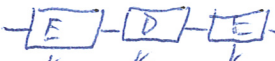
Ⓒ $\forall x_1 \dots x_d$ distinct and $y_1 \dots y_d$ $\exists!$ p poly of deg $< d$ s.t. $\forall i \; p(x_i) = y_i$
Lagrange theorem

DES - Feistel network with 16 rounds
64-bit blocks 56 bit keys

Key schedule



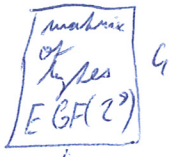
Critique of DES
- weak keys: if $K = 0^{56} \Rightarrow \forall i\ K_i = \ldots = 0^{48}$
$\Rightarrow$ all rounds are identical
$E_K = D_K$
- $E_K(\bar{x}) = \overline{E_K(x)}$ proof?
- too short key: brute force attack
2012 - crack in 26 hours (FPGA-based machine)
- too short blocks $\approx 2^{32}$ blocks have collision
workaround: double DES key size: 112 bits
but security level ≤ 57 b (meet-in-the-middle)

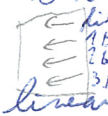
3DES 

168 bit keys
sec level ≤ 113
- too much secrets (unexplainable constants)
- attacks on structure ... 2^{47} chosen plaintexts

AES advanced encryption standard
- 1997 NIST public competition
- Rijndael $\rightarrow$ became AES in 2001
128b-blocks 128/192/256 b keys
rounds 10 12 14

Structure: SPN with linear step
internal state and round key



Round: S Bytesub - 16 identical S-boxes
P Shiftrows - 

L Mixcolumns - linear transform on every column
$\oplus$ Add round key ... XOR with K_i

Round of decryption:
Add round key
Inv Mixcolumns
Inv Shiftrows
Inv Bytesub

$\Downarrow$

Inv mix
add RK (mixed)
Inv Bytesub
Inv Shiftrows

Decryption is very similar to E^{-1} = reverse

Implementation technique: see lecture, usage on real CPU

Critique of AES
- single algebraic structure - advance in Lin Alg might break it
- small margin in # rounds
- byte-aligned
- 128b key: Quantum computer attack $\Rightarrow$ Grover algorithm $\sqrt{n}$ operations
- 128 b blocks: block collisions in $\approx 2^{64}$ blocks
workaround: change keys after 2^{32} blocks

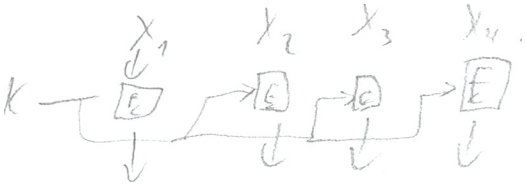
Other finalists in AES comp
- Serpent 128b blocks, 128-256b keys / 32 rounds SPN+linear
16 rounds Feistel net key dependent S-boxes
- Twofish - " -

Use of block ciphers:

block ciphers modes of operation

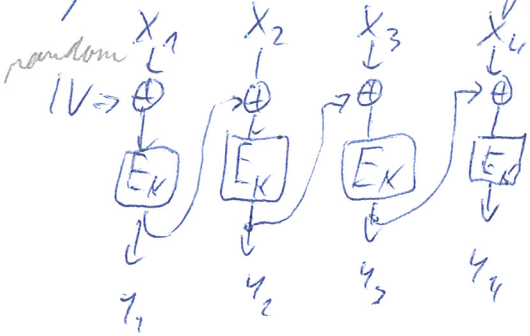
Padding - fill the rest of the last block

ECB: Electronic Code Book



AVOID
- no initial value (IV)
- reveals $X_i = X_j \Leftrightarrow Y_i = Y_j$
- flip bit in $Y_i \Rightarrow$ destroy X_i
  swap ciphertext $\Leftrightarrow$ swap plaintext

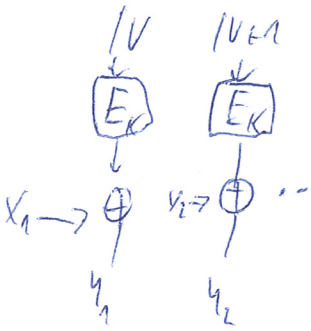
CBC: Cipher block chaining



decryption also left → right

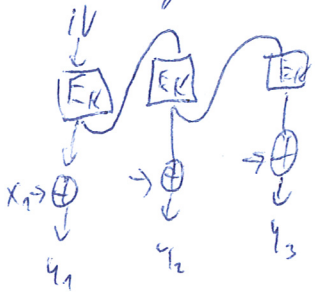
- requires random IV
- Bit flip in Y_i, destroys X_i
  flip in X_{i+1}
- $Y_i \Leftrightarrow Y_j$ flips in predictable way
- proof of security w.r.t. (Chosen plaintext attack) for not too long messages

CTR: Counter



- stream cipher
  - no padding needed
- MUST not repeat IV
- bit flip in $Y_i \Rightarrow$ bit flip in X_i
- random access
- can be parallelized

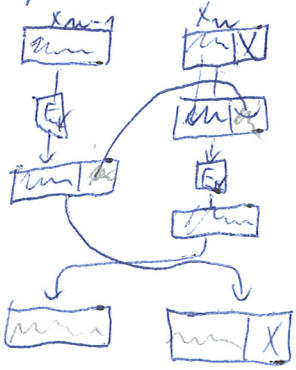
OFB: Output feedback



counter mode is just better

Ciphertext stealing - trick for avoiding padding for CBC - see lecture

for ECB



CBC $y_i = E_K(x_i \oplus y_{i-1})$

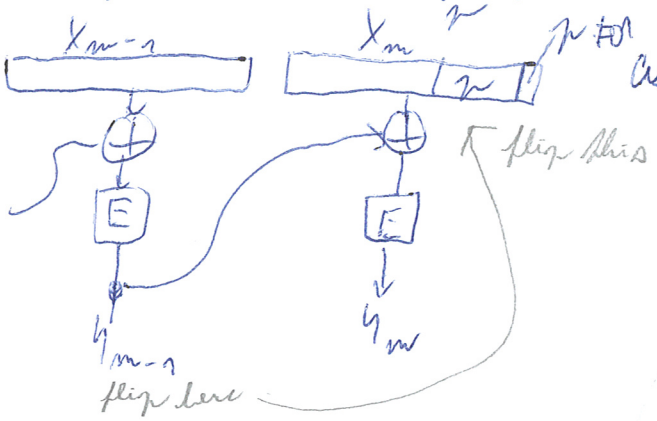
Leakage
ECB: $y_i = y_j \Leftrightarrow x_i = x_j$

$y_i = y_j$
likely to happen after $2^{b/2}$ blocks
$x_i \oplus y_{i-1} = x_j \oplus y_{j-1}$
$x_i \oplus x_j = y_{i-1} \oplus y_{j-1}$ } known to attacker
leaks b bits of data

on CBC
padding oracle attack
side channel attack
attacker can distinguish between wrong padding and wrong signature

MORAL
don't use 1 key for $2^{b/2}$ blocks

Padding scheme: data | x - - y | x bytes of value y



Sometimes in TLS

Assume $y = 01$,
exactly one solution with correct padding
$P \oplus F = 01$
↳ recovered y bytes of P

P bytes under control by increasing padding size by y+1
original bytes $\oplus F = P+1$
↖ recover last block easily
Complexity to recover B bytes $\Leftarrow B \cdot 2^8$

then you can throw out last block and recover every except first (if you can change IV)

CTR $y_i = x_i \oplus E_K(IV + i - 1)$

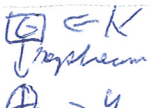
$C_1 \dots C_m$ $C_i = E_K(IV + i - 1)$
all C_i s are different

$y_i \oplus y_j = (x_i \oplus C_i) \oplus (x_j \oplus C_j) =$
$= (x_i \oplus x_j) \oplus (C_i \oplus C_j)$
$C_i \oplus C_j \neq 0$

$y_i \oplus y_j \neq x_i \oplus x_j$
$y_i \oplus y_j$ known
rules 1 out of 2^b

leak: $b - \log(2^b - 1) \approx c \cdot 2^{-b}$
bits leaked $\leq \binom{m}{2} \cdot c \cdot 2^{-b}$ ... costs for $m \sim 2^{b/2}$
$\binom{m}{2}$ # pairs
$c \cdot 2^{-b}$ leak per pair

Stream ciphers



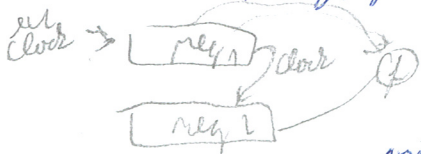
eSTREAM project
- goal: find new stream ciphers
2004 --- finals 2008
profile 1: SW → 4 ciphers
profile 2: HW → 3 ciphers
↳ TRIVIUM 288 bits of state
- key - 80 b
- i.v. - 80 b
- constants ...
1152 idle steps
see level 80

LSFR - Linear feedback shift register

trivial in hardware
↳ trivial to crack in KPA - known plaintext attack

Attempts to save
- non-linear feedback (&)
- non-linear output
- Combine multiple registers

control clock of reg1 by out of reg2



Ex A5/1 (GSM)
privately developed ⇒ weak

RC4 (Rivest 1987) permutation based working on bytes

state: 256 bytes

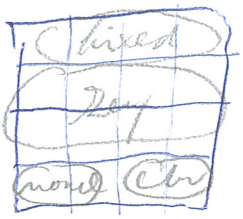
BROKEN KPA

Chacha 20 (Bernstein 2008) derived from SALSA20 eSTREAM SW profile

32 B key
8 B nonce
8 B block counter

} chacha20 → not bijective → 1 block of keystream

state: matrix 32 bit values



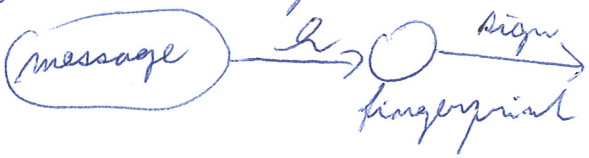
20 rounds
ARX ← XOR
↑ addition
↖ rotation

Hash functions $\{0,1\}^* \xrightarrow{h} \{0,1\}^b$

basic properties

1. no collision: $h(x) = h(x'), x \neq x'$
2. no second: for given x find x': $h(x) = h(x')$
3. no inversion: for given y find x: $h(x) = y$

use case: signature



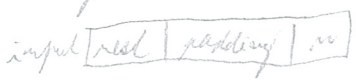
Hash functions: want collision resistance

Merkle-Damgard construction

Given a compression function $f: \{0,1\}^b \times \{0,1\}^b \to \{0,1\}^b$

construct h:

typical implementation



Theorem: If f is collision resistent, then h is collision resistent

Proof: If we have $h(x_1 \dots x_n) = h(x'_1 \dots x'_{n'})$ different

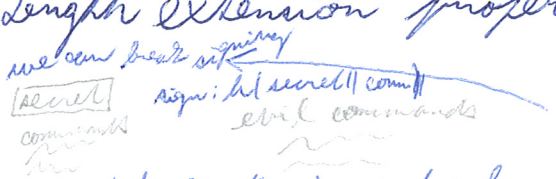
for contradiction either $n \neq n'$ ... collision in f $f(-, n) = f(-, n')$

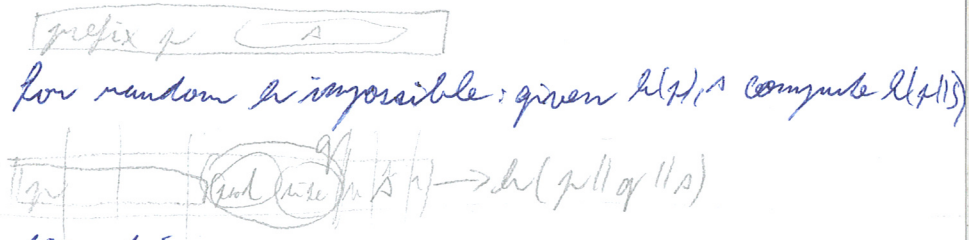
or $n = n'$ ... we go backwards and we have to find different blocks with collision

□

Length extension property

we can break signing: $h(\text{secret} \| \text{com})$





for random h impossible: given $h(p)$, s compute $h(p \| q \| s)$

How to obtain compression function f:

Davies-Meyer construction from block cipher

$f(u, v) := E_u(v) \oplus v$

random permutation

Theorem: With an ideal block cipher, f is collision-resistant

For attack evaluating E/D q times

$\Pr[\text{collision found}] \leq q^2 / 2^b$

Proof: WLOG we ask no redundant questions

we can ask $E_u(v) \to f(u, v) = E_u(v) \oplus v$

$D_w(w') \to f(u, D_w(w') = v \oplus D_w(w')$

if we find collision in step i ($i \leq q$) we found pair in $i-1$ previous

for every pair: $\Pr[\text{collision}] = \frac{1}{\#\text{possible}} \leq \frac{1}{2^{b-i-1}} \leq \frac{1}{2^{b-1}}$

$\Pr[\text{collision}] \leq \frac{1}{2^{b-1}} \cdot \#\text{pairs} \overset{\binom{q}{2}}{=} \frac{q^2}{2 \cdot 2^{b-1}} = \frac{q^2}{2^b}$

① why $\oplus v$, if not easy collision

$f'(u, v) := E_u(v)$

$E_w(v) = y$

$D_{w'}(y) = v'$

then $f(u, v) = f'(w', v')$

$y = y$

② with DES

$E_{\bar{k}}(\bar{x}) = \overline{E_k(x)} \Rightarrow f(\bar{a}, \bar{b}) = \overline{E_a(b)} \oplus \bar{b} =$

$= \overline{E_a(b)} \oplus \bar{b} = E_a(b) \oplus b = f(a, b)$

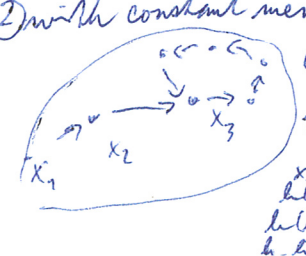
③ for $v := D_w(0)$

$f(w, v) = E_v(v) \oplus v = 0 \oplus v = v$

Finding collision

1. brute force ... by birthday paradox ... matching $2^{b/2}$ steps (with lost of memory)

2. with constant memory: imagine a graph



$G = (V, E)$, $E = \{(x, h(x))\}$

possible blocks $\Rightarrow$ it's lollipop graph

x_1
$h(x_1)$
$h(h(x_1))$
$h = h(x_1)$

tortoise: 1 step

hare: 2 steps

wait until they meet - that's the collision with tortoises previous step

$\Rightarrow$ after l steps they are on the cycle and can't escape $\Rightarrow$ they meet (only at that point)

③ need meaningful messages
parametrised message → previous construction with $E := (x, h(\text{parametrised}))$

④ $h(\text{evil}) = h(\text{innocent})$: A, B random subsets of X ($|X| = n$)
$|A| |B| \sim \sqrt{n} \Rightarrow$ likely $|A \cap B| \geq 1$
generate $2^{b/2}$ innocent → hash → A ...
$2^{b/2}$ evil → hash → B

⑤ if h is M.-D. lots of collisions are as hard as 1 collision
in $\sim 2^{b/2}$ steps: $x_1 \neq x_1'$ $f(v, x_1) = f(v, x_1') = y_1$
— " — $x_2 \neq x_2'$ $f(y_1, x_2) = f(y_1, x_2') = y_2$ } r times

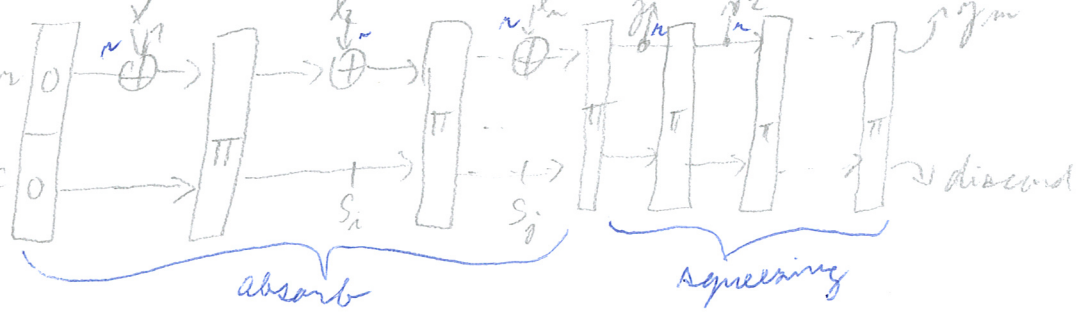
x_1	x_1'
x_2	x_2'
...	...

2^r combinations which hash to the same result

in time $2^{b/2} \cdot r$ we produce 2^r-fold collision

note: Concatenation of 2 hashes $h(x) = h_1(x) \| h_2(x)$, how strong?
if one of $h_{1,2}$ is M-D — by ⑤ we find $2^{b/2}$ colliding msgs in $b/2 \cdot 2^{b/2}$
↳ the other will likely collide for 2 of these — collision in time $b/2 \cdot 2^{b/2}$ with $r = b/2$

Real world — MD5 (Rivest 1995) 128b result (small) Broken! - can't find collision
SHA-1 (NSA) 160b result - Broken! (2017)
SHA-2 (NSA) 224-512b result not broken yet
↳ Competition by NIST → SHA-3 - published in 2015
Sponge construction: phase 1 - absorbing input
... 2 - squeezing out output
- permutation π on blocks of size $w = r + c$ width, rate, capacity



Security level against brute-force = ① $2^{r/2}$ by birthday paradox (attacking output)
② internal collisions: in $2^{c/2}$ blocks we find $i < j$: $S_i = S_j$
first: $\pi(i)$
and: $\pi^{-1}(h_i \oplus h_j)$ } output of π is the same ⇒ we get same output
for π random
security level of sponge $\geq \min(r/2, c/2)$