Oliver Song
6.857 exam summary sheet

Misc.

Decision Diffie Hellman
Given g, g^a, g^b, g^{ab} ~= g, g^a, g^b, g^r cannot determine that they are different exponents

Diffie Hellman Triple
g, g^a, g^b, g^{ab}
"**triple with respect to some generator**" is....if with the new generator you still satisfy those requirements

"**Perfect secrecy**" : no way to compute the original text from cipher e.g. one time pad

One time pad: Alice and bob share key K, Alice XORs M with K, sends, then Bob XORs again with K.
SUPER MALLEABLE because any bit an adversary flips will flip in Bob's answer.
OTP can be broken if you xor two ciphertexts, you figure out Ms
$C_1 \oplus C_2 = (M_1 \oplus K) \oplus (M_2 \oplus K)$

Malleable = adversary can change the result (not necessarily knowing what it does)

Modular elliptical curves
As per Krishanu, you're looking for how many integer answers there could be within the range [0, Modulus - 1]. If it's a square modulus, it should have two solutions, and if it's anything else it will have none. In the case of the "elliptic curve" it should have N solutions, I think.

Repeated squarings algorithm
To compute a^x mod p, O(log p) multiplations because you calculate something for every binary digit of x.

Fermat's little theorem
6^{46} mod 23 = ?
>>>>>[ ]^{22} = 1 mod 23
so 6^{22} = 1 mod 23, so 6^{44} = 1 mod 23, so $6^{44} * 6^2$ mod 23 = 1 * 36 mod 23 = 13

Hash function properties

One Wayness
Given **y**, it is hard to find any **x** such that **h(x) = y** (cannot find pre-image)

Collision Resistance
Cannot find x != x', h(x) = h(x'). In random oracle model, requires trying $2^{d/2}$ x's before a pair x_i, x_j colliding is found (birthday paradox).

'Weak' Collision resistance (target collision)
Infeasible, given x, to find x' != x s.t. h(x) = h(x'). In ROM, can find x' in $\Theta(2^d)$ time.

Birthday paradox
Reason why hash function outputs are generally twice

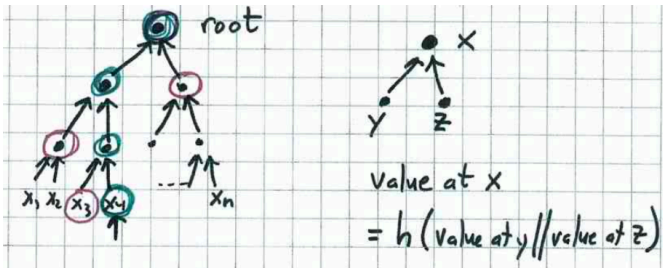
Pseudo-randomness
h is indistinguishable under black-box access from a random oracle. This requires a family of hash functions, one of which is chosen at random.

Theorems (CR / OW)
Collision resistance implies target collision resistance
One-wayness does not imply collision resistance, and vice versa
If h "compresses", then CR => OW.

Hash function applications

1. **Password storage for login**
   - Store h(PW), not PW, on computer
   - When user logs in, check hash of PW against table
   - Disclosure of h(PW) should not reveal PW
   - Need **OW**
2. **File modification detector**
   - For each file F, store h(F) securely
   - Can check if F has been modified by recomputing h(F)
   - Need **WCR** (aka TCR)
3. **Digital Signatures ("hash and sign")**
   - PK_A = Alice's public key (for signature verification)
   - SK_A = Alice's secret key (for signing)
   - Signing: $\sigma = sign(SK_A, M)$
   - Verify: $Verify(M, \sigma, PK_A) = \{True, False\}$
   - Adversary wants to force a signature that verifies.
   - For large M, it's easier to sign h(M):
     - $\sigma = sign(SK_A, h(M))$
   - Verifier recomputes h(M) from M to verify
   - h(M) is the "**proxy**"
   - Needs **CR** (Alice gets Bob to sign x, where h(x) = h(x'), then claims Bob really signed x')
   - Doesn't need OW (e.g. h = identify is actually OK...lol)
4. **Commitments**
   - Alice has value x (e.g. an auction bid)
   - Alice computes C(x) her "commitment to x" and submits C(x) as her "sealed bid"
   - When bidding has closed, Alice should be able to "open" C(x) to reveal x
   - **Binding property**: Alice should not be able to open C(x) in more than one way (committed to just one x)
   - **Secrecy (hiding)**: Auctioneer or anyone else seeing C(x) should not learn anything about x
   - **Non-malleability**: Given C(x) it shouldn't be possible to produce C(x+1) by manipulation
   - **How**: C(x) = x XOR with some random 256 bit number (kinda like one time pad).
   - Must be randomized!
   - Need **OW, CR, NM**
5. **Authenticating a collection**

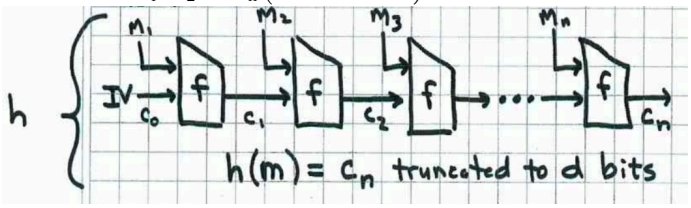
- Build a tree with n leaves x_1, x_2, ..., x_n & compute authenticator node as fn of of values at children. This is a **Merkle tree**.
- Value at node x = h(value at y XOR value at z)



- Root node is the authenticator for all n values! Applies to: time-stamping data, authenticating whole file system
- Requires CR

Merkle-Damgard hash function construction

- Choose output size d (e.g. d=256 bits)
- Choose "chaining variable" size c (e.g. c=512 bits)
  - c must be >= d, better if c >= 2*d
- Choose message block size b (e.g. b = 512 bits)
- Design "compression function" f
  - f: $\{0,1\}^c * \{0,1\}^b \rightarrow \{0,1\}^c$
- Merkle-Damgard 1) essential a "mode of operation" allowing for variable length inputs
  - Choose a c-bit initialization vector IV, c_o Note that c_o is fixed and public
  - **Padding** given message, append
    - 1 0 * bits
    - fixed-length representation of length of input
  - so result is a multiple of b bits in length
  - M = M_1 M_2 ... M_n (n b-bit blocks)



Theorem: If **f** is **CR**, then so is **h**. If **f** is **OW**, then so is **h**.

SHA3 (Keccak)

Uses sponge structure. Inputs get broken in to blocks, XOR'd then block permutation is applied. Then goes through many rounds of this. More info attached.

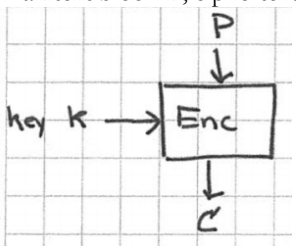
	d	note
MD4	128	broken wrt CR
MD5	128	broken wrt CR
SHA-1	160	? CR ?
SHA-256	256	
SHA-512	512	
SHA-3	2.24256E+11	

File system security with Hashing

1. Confidentiality
   a. How to provide confidentiality? Encryption, but not OTP because it takes up too much space
2. Integrity
   a. How to protect integrity? Merkle trees
   b. Why can't the server modify a file? What property of the hash prevents against? Collision resistance
3. Integrity of history of updates: **chained hashing**
   a. F_0, H_0 = hash(F_0), ch_0 = H_0
   b. F_1, H_1 = hash(F_1), ch_1 = hash(H_0, ch_0)
   c. etc...such that they all depend on each other!

Block ciphers

Plaintext block in, ciphertext block out

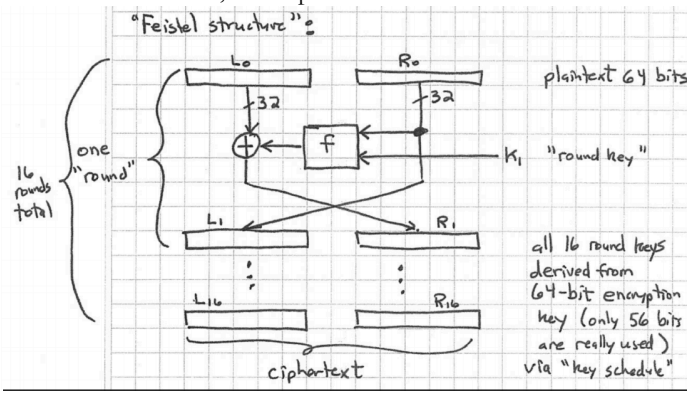


Fixed Length P, C, K. e.g.

- DES: |P| = |C| = 64 bits, |K| = 56 bits
- AES: |P| = |C| = 128 bits, |K| = 128, 192, 256 bits

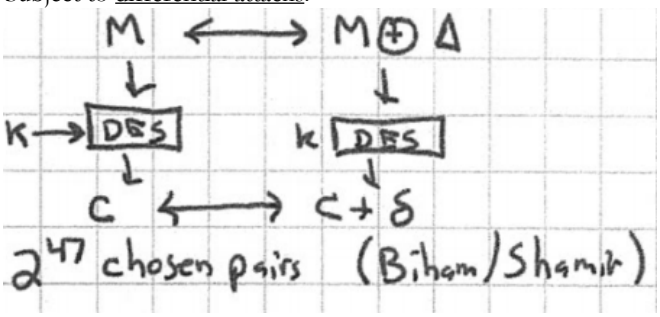
DES

Standardized in 1976, not deprecated. Uses "Feistel structure"



Invertible for any **f** and any key schedule. **f** uses 8 "S-boxes" mapping 6-bits => 4bits nonlinearly. Key is too short! Breakable now quite easily but brute force.

Subject to differential attacks:



Subject to linear attacks:
e.g. if $M_3 \oplus M_{15} \oplus C_2 \oplus K_{14} = 0$
with prob p = 1/2 + e
then need $1/e^2$ samples to break.

AES

View input as 4x4 byte array. 4x4x8 = 128, this is 128 bit version.

- Derive 11 "round keys" each 128 bits (4x4xbyte)
- In each round:
  - XOR round key

- Substitute bytes (lookup table)
- Rotate rows (by different amts)
- Mix each column (by linear operation)

- Output final state

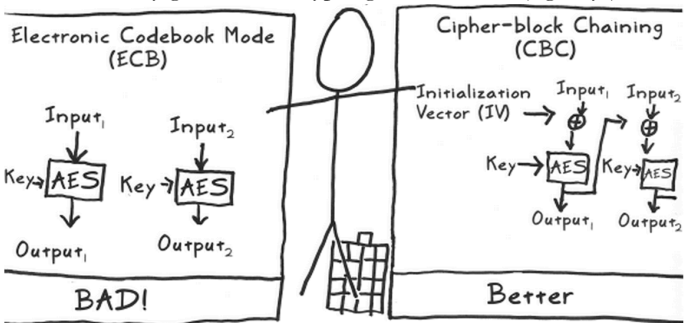
AES can be pretty much treated as an **ideal block cipher**:

- For each eky, mapping Enc(K, •) is a random independent permutation of $\{0, 1\}^{128}$ to itself.

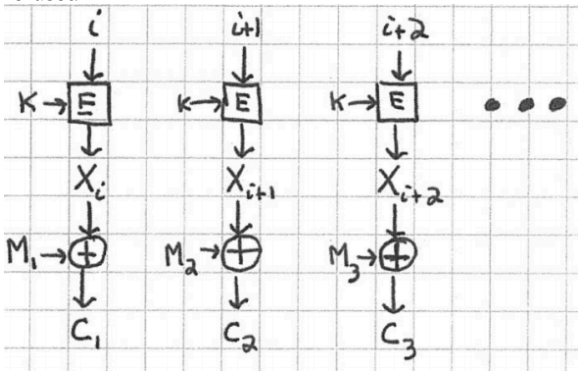
Block Cipher Modes of Operation

How to encrypt variable length messages?

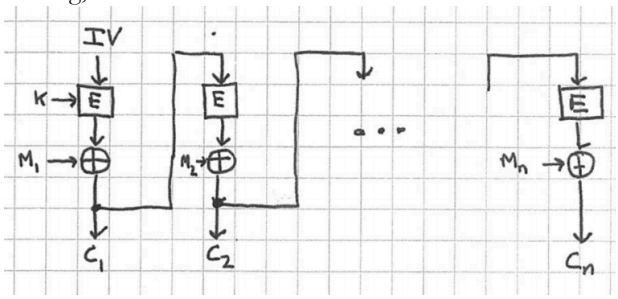
- **ECB = "Electronic code book"**
  - See below. Pad then run parallel. Bad. Preserves patterns, and only good for encrypting random data (e.g. keys)



- **CTR = "Counter mode"**
  - Cenerate Pseudorandom sequence by encryption i, i+1, ... XOR with message to obtain ciphertext. Initial counter value can be transmitted first. No counter value should be re-used.



- **CBC = "Cipher block chaining"**
  - See above. Chain together, outputs affect each other. Good. Choose IV randomly, then use each C_i. Transmit IV with ciphertext: IV, C1, C2, C3...Cn. Decryption easy and parallelizable (! so little error propagation).
- **CFB = "Cipher Feedback"**
  - Similar to CBC mode. Uses random IV transmitted with ciphertext. If M is not multiple of b bits in length, can just transmit shortened ciphertext. No need for ciphertext stealing, unlike CBC.



IND-CCA

Game with adversary to test if these modes for block ciphers are good. Mode is IND-CCA secure **if adversary can win with probability at most 1/2 + e for "negligable" e.**

Let **K** be randomly chosen key
Let $\mathbf{E_K}$ denote encryption (using mode) with key **K**
Let $\mathbf{D_K}$ denote decryption

Phase I ("Find")

- Adversary given black-box access to $\mathbf{E_K}$, $\mathbf{D_K}$ (can encrypt/ decrypt whatever it likes)
- Adversary outputs two messages $\mathbf{m_0}$, $\mathbf{m_1}$ of same length, plus state information **s**.

Phase II ("Guess")

- Examiner secretly picks $d \leftarrow^R \{0, 1\}$
- Examiner computes $y = E_k(m_d)$
- Adversary given y, s, access to E_k, and access to D_k (except on y)
- Adversary computes for a while, then must produce bit **d'** as its guess for **d**.
- Adversary's advantage is $|P(d' = d) - 1/2|$

Encryption secure against CCA attack if advantage is negligible.

Fact: **To be IND-CCA secure, encryption method must be randomized!** (else adversary can encrypt $\mathbf{m_0}$, $\mathbf{m_1}$ & compare to **y**). Also, random values used should not be evident / available to the adversary, so he can use them in decryption.

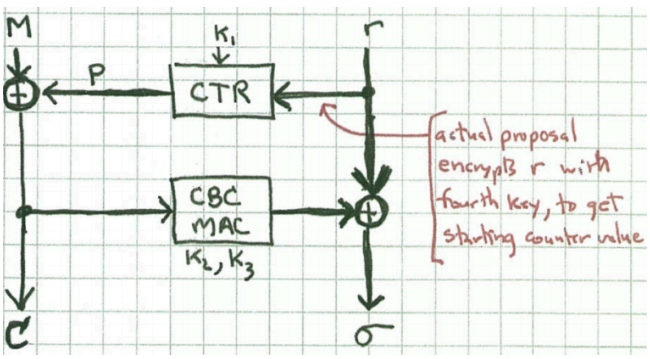
Theorem: Modes ECB, CTR, CBC, CFB are not IND-CCA secure.

UFE mode (to the rescue!)

M = long message, sequence $M_1, M_2, \ldots, M_n$ of b-bit blocks
K = (K_1, K_2, K_3) ← Three independent keys for block ciphers
$r \leftarrow^R \{0, 1\}^b$ Starting counter value

pad $P = P_1, P_2, \ldots, P_n$ where $P_i = E_{K1}(r+i)$ ← (CTR mode)
ciphertext $C = C_1, C_2, \ldots, C_n$ where $C_i = M_i \oplus P_i$

CBC-MAC: $X_0 = 0^b$
$X_i = E_{K2}(X_{i-1} \oplus C_i)$ $1 <= i < n$
$X_n = E_{K3}(X_{n-1} \oplus C_n)$ (MAC)
$\sigma = r \oplus X_n$ use MAC to mask r (no msg authentication)
Output: $C = C_1, C_2, \ldots, C_n, \sigma$



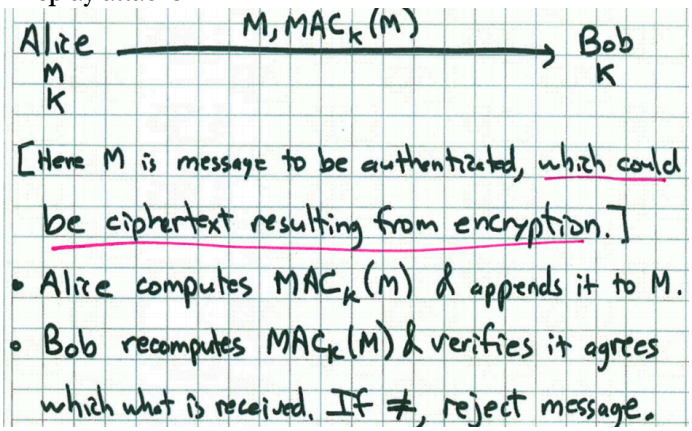
- Encryption with UFE can be done in single pass over data, but decryption requires two passes:
  - First to compute MAC X_n then to get r
  - Second to decrypt C to get M
- Only design for confidentiality (there is no way provided for receiver to tell it ciphertext has been tampered with). Need to use MAC on top of all of this, or some "combined mode" providing both confidentiality and integrity
- Note "unbalanced Feistel structure"
- Length of ciphertext (C, σ) = |M| + |r|; expansion only as needed for randomization, no need for ciphertext stealing since we use CTR mode.

CIA triad

- Confidentiality
  - It can't be seen by adversaries
- Integrity
  - The message the receiver gets is the same the sender sent
- Availability
  - Are you even available to receive? (DDoS)

MAC (Message Authentication Code)

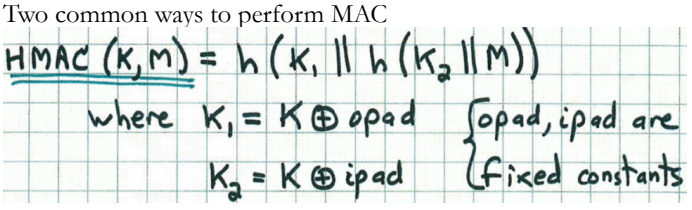
- Not **confidentiality**, but **integrity**
- Alice wants to send messages to Bob, such that Bob can verify that messages originated with Alice and arrive unmodified.
- Alice and Bob share a secret key K
- Orthogonal to confidentiality; typically do both (e.g. encrypt then append MAC for integrity)
- Need additional methods (e.g. counters) to protect against **replay attacks**

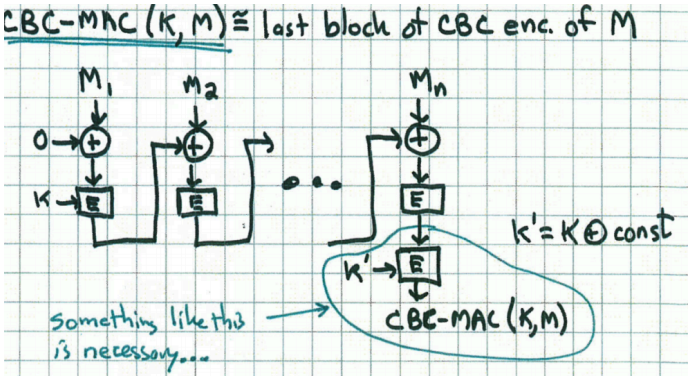


Adversary Eve wants to forge M', $MAC_K(M')$ pair that Bob accepts, without Eve knowing K.

- She may hear a number of valid (M, $MAC_K(M)$) pairs first, possibly even with M's of her choice (chosen msg attacks).
- She wants to forge for M' for which she hasn't seen (M', $MAC_k(M')$) valid pair.

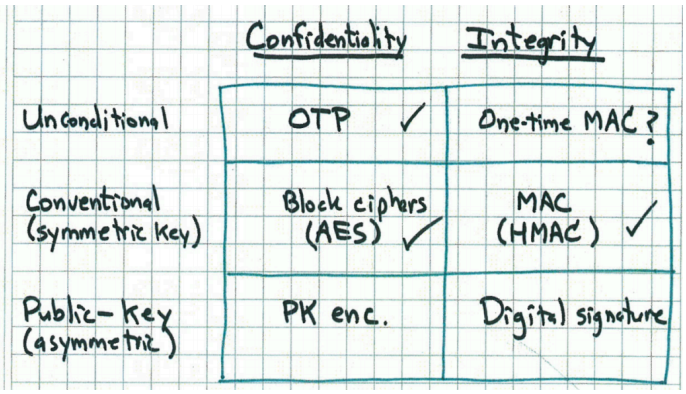
Two common ways to perform MAC





There's also MAC using random oracle, which is pretty straightforward.

$MAC_K(M) = h(K||M)$



Finite Fields

Field means group of numbers with some property, aka R, set of reals, and C, sex of complex numbers. These are infinite fields. Regular rules of mathematics / algs apply to fields.

- **Abelian = commutative**
- In crypto we like finite fields, like Z_p, set of integers mod prime p.
- **GF(q)** is the finite field ("Galois field") with q elements.
- **Theorem**: GF(q) exists whenever $q = p^k$, p prime, k > 0

(I printed the rest)

Diffie Helman

Remember the color mixing example

Alice sends g^x, Bob sends g^y, they both compute g^{xy}. Eve cannot discern x, y or xy.

RSA

OAEP (padding)

IND-CCA2