Textbook: Intro to Modern Cryptography
- J Kate, Y. Lindell.

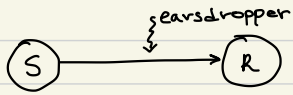
Fantastic -

Q: When is a problem an InfoSec (IS) one?
A: When the problem is impossible to solve perfectly.

If this is indeed true, then naturally the field is fantastic, since we are dealing with problems that are themselves impossible.

Consider the simple problem of a password scheme.
If you want to make a "perfectly secure" password scheme, you require infinite length passwords. ∴ perfectly secure password schemes cannot exist. ⇒ It is an infosec problem.

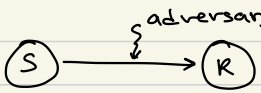
Consider the secure communication problem.



There must be some time t_0, such that the eavsdropper knows everything that the receiver knows.
Now for each time $t > t_0$, the eavsdropper receives everything that the receiver receives.

Notice this is again an IS problem. Due to the inherent impossibility.

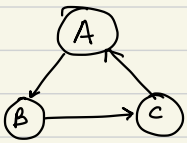
Consider the Data Integrity Problem -



Is the data sent by the sender same as the data received by the receiver?
Assume to the contrary that R knows when a modification is made. If the R can detect this, it can magically distinguish b/w an un-

-modified M' and some M modified into M'.

consider a problem where there is one liar.



How do I find out whose the liar? (who changed the message).
B sends a 0 and claims it sent a 1 OR C claims it received a 1 when it actually received a 0, etc.

Clearly A cannot find out this by simply asking B and C.

Fascinating

Q How do we logically solve a logical impossibility?

A Bring in another impossibility and make it destructively interfere with the other impossibility.

Jerry beating Tom is a logical impossibility.
Similarly Jerry beating Spike is a logical impossibility.
Jerry ensures Tom and Spike destructively interfere.

consider the secure communication problem again.
Now it is impossible for information to travel faster than light.
Add a 2nd impossibility - it is impossible to stop the universe from expanding.
Now its impossible for an adversary sitting at a place where the universe expands faster than light to be able to eavsdrop on this information.

For practical purposes we use the fact that a DTM can't solve

NP-Hard problems. (another impossibility).

Fundamental

Any impossible problem in any field can be solved using IS, since IS itself doesn't care which field that the original impossibility. actually came from. For ex in coding theory, hashing and fingerprinting can beat the singleton-bound.

⇒ clearly IS has applications in every field. This is as fundamental as it can get.

Consider Randomized algorithms. Pseudorandomness is indistinguishable from a random variable. ⇒ pseudorandom algorithms should serve as effectively.

In Mathematics, consider a solution to a problem like winning strategy for chess. The shortest proof would contain expo entries. Soln - Use interactive / zero knowledge proofs.

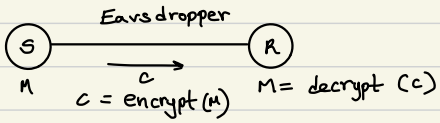
In This Course

1. Computational Hardness. (Resource Complexity)
2. Practical Uncertainties. (Noise, Routing uncertainty etc).
3. Natural Limits (speed of life / quantum)
4. Logical / Philosophical Impossibilities.

Today

1. Kerckhoff's Principle
2. Design / Breaking ciphers.

Secure Communication Problem



S: Sender R: Receiver
E: eavsdropper.
M: Message

Caesar's Cipher

Science = every man's art : no natural gifts required.
Goal of science - convert art to science.

Caesar cipher is back from when the field was still an "art".

$$\text{encrypt}(c) = c + 3 \pmod{26}$$

For ex -

e(a) = d e(hello) = khoor
e(b) = e

Kerckhoff's Principle

In Kannan's words (look at the book for the actual definition).

Security of a system must not depend on the obscurity of the algorithm, rather must only depend on the secrecy of the key.

Intuition -

Reason 1 - Algorithms can be reverse engineered.
2 - Updation/Recovery Complexity - If your security is broken, you will have to change the entire algorithm. However if

you had the latter, changing the key is enough.

3 - Secure memory is costly.

Universal Turing Machine - takes a TM as input and simulates it.
$U_{TM}(T, x) = T(x)$.

4 - For one node to communicate with multiple other nodes I would need a different algo for each pair of nodes. In the latter, I would only need a different key for each pair.
=> Much more scalable.

Kannan's Reasons -
Ethical Hacking: Arose as a consequence of Kerckhoff's principle. Expose the algo to ethical hackers and make them test security.

Standards: If every person on the internet uses the same algorithm (otherwise communication is impossible), naturally we must follow Kerckhoff. ⇒ A communication standard is impossible without following Kerckhoff.

Shift Cipher

$$e(x) = x + k \pmod{26}$$

→ Brute Force Attack - Try each of the possible keys.
Lesson: Use a large key space.

Consider -
$a = 0$, $b = 1$, ..., $z = 25$

Let q_i = prob of ith char in C.

p_i = expected prob of ith char in message M.
(using some sort of prior knowledge).

Now $q_{(i+k) \% 26} \approx p_i$ (because its a shift cipher).

Now based on prior knowledge we know $\sum_{i=0}^{25} p_i^2 = 0.065$

We can compute $\sum_{i=0}^{25} p_i q_{i+k} \approx \frac{1}{26}$ (if our gress of k is wrong)

≈ 0.065 (if our gress of k is right)

Mono-alphabetic Substitution Cipher.

$e(a\ b\ c\ \ldots\ z) = X\ D\ T\ \ldots\ K$ Key is a permutation σ_{26}.

$\Rightarrow$ There are 26! potential keys.

→ Direct Brute force is hard since 26! is large. (Lesson Learnt).
→ Frequency attack -

Notice that for $\forall i$, $\exists j$ s.t. $q_j \approx p_i$ (the one that it replaced)
consider this simple attack -

1. Sort the alphabet based on p_i
2. Sort the alphabet based on q_j
3. Now its likely the key is this permutation.

Vigenere Cipher

1. Choose a keyword. For ex - sea.
2. Add it to the message word.

h	e	l	l	o
s	e	a	s	e
Z	I	L	D	S

Lesson Learnt: The key messes with the frequency of each character.

3. Decode with additive inverse.

This wasn't broken for ~100 years.
Until -

Assume we know the length of the key. for ex - |sea| = 3.
→ Partition the cyphertext into |K| parts. Each bucket contains all characters i s.t. i ≡ |K| are equal.
For ex if |K| = 3, bucket 0 : $C_0\ C_3\ C_6 \cdots$
1 : $C_1\ C_4\ C_7 \cdots$
2 : $C_2\ C_5\ C_8 \cdots$
→ Notice each of these are by themselves outputs of some shift cipher. Break these individually using a modified frequency attack. i.e. instead of using q_i = frequency in the cyphertext, consider q_i = frequency in the bucket.
→ Brute force over the key length |K| and repeat this.

→ Natural Extension: Make a collection of mono-alphabetic buckets.

A general pattern we notice in these is that people are creating-breaking-repeating. There seems to be no goal in sight. This is the indication that literacy about the field is lacking.

Today -

→ Shannon's Perfect Secrecy

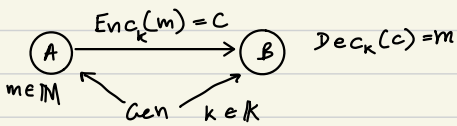
→ Vernam's Cipher is perfect (one-time pad)

→ Limitations of Shannon's approach.

Shannon - Need a mathematical definition for "unbreakable" in order to create an unbreakable scheme.

Shannon's Perfectly Secure Scheme

An encryption scheme is a 4-tuple $\langle Gen, En, Dec, \mathbb{M} \rangle$



An encryption is said to be perfectly secure if for all probability distributions over $\mathbb{M}$, and for all $m \in \mathbb{M}$, for all $c \in \mathbb{C}$ (where $p(\text{ciphertext} = c) > 0$) -

$$p(\text{message} = m \mid \text{ciphertext} = c) = p(\text{message} = m).$$

Intuitively: Looking at the ciphertext doesn't give me any information about the message extra than what I had beforehand (apriori).

Vernam Cipher.

$\mathbb{M} = \mathbb{K} = \mathbb{C} = \{0,1\}^n$

Gen : $K \in_r \{0,1\}^n$ (prob of each is 2^{-n})

Enc : $c = m \oplus k$

Dec : $m = c \oplus k$

correctness here is fairly obvious since XOR is invertible.

we need to show that this conforms to Shannon's definition.

Digression: Proofs by induction are easy for the prover, but offers no intuition / what to do next etc.
contrarily, a proof by construction gives an element of intuition.

In information security - simply defining the problem is a contribution. However in Shannon's case, he managed to show that anything that met his definition ended up being impractical.

∴ Appreciate solutions that are thoroughly defined but at the same time can be practically achieved. (Workable + Comprehensive).

Vernan's Cipher meets Shannon's Definition -
consider an equivalent definition for Shannon's definition -

Defn (2) $\forall$ prob distr over $\mathbb{M}$, $\forall\, m \in \mathbb{M}$, $\forall\, c \in \mathbb{C}$,
$p(C=c \mid M=m) = p(C=c)$.

Proof: We need to show $p(C=c \mid M=m) = p(C=c)$
$\Leftrightarrow$ $p(M=m \mid C=c) = p(M=m)$

Multiply both sides by $\frac{p(M=m)}{p(C=c)}$ we get,

$$\frac{p(M=m)}{p(C=c)} \cdot p(C=c \mid M=m) = \frac{p(M=m)}{p(C=c)} \cdot p(C=c)$$

$$\Rightarrow \frac{p(C=c, M=m)}{p(C=c)} = p(M=m) \quad \text{(Bayes Theorem)}.$$

$$\Rightarrow p(M=m \mid C=c) = p(M=m)$$

The reverse direction can be shown the same way.

Lets consider a further equivalent definition -

Defn (3) $\forall c \in \mathbb{C}, \forall m_0, m_1 \in \mathbb{M}$,

$$p[C=c \mid M=m_0] = p[C=c \mid M=m_1]$$

Proof: Our second definition holds for all m.

$\therefore p(C=c \mid M=m_0) = p(C=c)$

similarly $p(C=c \mid M=m_1) = p(C=c)$.

$\Rightarrow$ Defn (2) $\Rightarrow$ Defn (3).

We now need to show (3) $\Rightarrow$ (2),

Now $p(C=c) = \sum_{m \in M} p(C=c \mid M=m) \cdot p(M=m)$

$=$

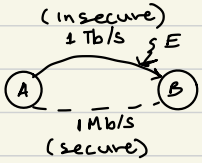
Now lets show Vernam meets this Defn (3),

$$p(C=c \mid M=m_0) = p[C = m \oplus k] = p[k = c \oplus m_0]$$

$$= \frac{1}{2^n} = \text{RHS}.$$

So why is this impractical?

Need a secure channel to share the secure key.



Notice that for each bit of message I need 1 bit of key. This is as bad as sending the entire message on the secure channel.

This makes this 1 time pad really impractical.

So why is it popular? -

→ Consider a secure channel that is available when the message is not available. This sort of "defers" the messages.

→ Sending messages to the same node. For ex - cases where some thing has to be removed and re covered.

Shannon also proved that any algo reaching his bound doesn't work practically.

This caused a sort of bifurcation in the field -

1. Slow Secure channel. + Fast Insecure Channel. } Fast Secure Channel. (Symmetric Key Crypto)

2. Fast only insecure channel. ⇒ slow insecure channel. } Public Key Cryptography.

Intuition of Shannon's Theorem -
For any perfectly secret encryption scheme $|\mathbb{K}| \geq |\mathbb{M}|$.
(For Shannon, it was $H(\mathbb{K}) \geq H(\mathbb{M})$.).
But we can be satisfied by the above statement since $H(|\mathbb{K}|) = \log_2 |\mathbb{K}|$, and any compression on $\mathbb{M}$ only makes $|\mathbb{M}|$ smaller.

Consider some ciphertext c. construct $D = \{ m \mid \exists K \in \mathbb{K}, Dec_K(c) = m \}$

Now, $|D| \leq |\mathbb{K}|$. Assume to the contrary that $|\mathbb{K}| < |\mathbb{M}|$
$\Rightarrow |D| < |\mathbb{M}|$
$\exists m^* \in \mathbb{M}$ s.t. $m^* \notin D$
Now $p(M = m^* \mid C = c) = 0 \neq p(M = m^*)$
This contradicts the fact that the scheme is perfectly secret.

Finite Fields (Chiranjeevi)

Groups

→ Non empty set G and an operator $\cdot$ is a group if

a. $\cdot$ is associative. i.e. $(a \cdot b) \cdot c = a \cdot (b \cdot c)$

b. $a \cdot b \in G$ for $a, b \in G$

c. $\exists\ 1 \in G$, $a \cdot 1 = a$ (Identity)

d. $\forall a\ \exists\ a^{-1} \in G$, $a \cdot a^{-1} = a^{-1} \cdot a = 1$ (Inverse)

Subgroups

$H \subseteq G$ s.t.

a. for $a, b \in H$, $a \cdot b \in H$

b. $1 \in H$

c. $\forall\ a \in H$, $\exists\ a^{-1} \in H$, $a \cdot a^{-1} = a^{-1} a = 1$.

Cyclic Groups

$\langle a \rangle = \{ a^0, a^1, \ldots, a^i \}$

For ex - $(\mathbb{Z}_p, +)$

Ring

$(R, +, \cdot)$ is a Ring if -

i) $(R, +)$ is an Abelian group

ii) $a, b \in R \Rightarrow a \cdot b \in R$ (closure)

iii) $(a \cdot b) \cdot c = a \cdot (b \cdot c)$ for $a, b, c \in R$ (Associative)

iv) $(a + b) \cdot c = a \cdot c + b \cdot c$ } (Distributive)
and $a \cdot (b + c) = a \cdot b + a \cdot c$ }

Zero Divisors

$a\ (\neq 0) \in R$ if $\exists\ b \in R\ (b \neq 0)$, $a \cdot b = 0$ OR $b \cdot a = 0$. (0 is identity wrt $\cdot$)

For ex - $(\mathbb{Z}_4, +\bmod 4, \cdot \bmod 4)$.

since $2 \cdot 2 \equiv 0 \bmod 4$

Integral Domain

A commutative ring (wrt $\cdot$) with no zero divisors.

Division Ring

An integral domain such that $(R - \{0\}, \cdot)$ is a group.

Field

$(F, +, \cdot)$ is a Field if -

i) $(F, +)$ is an abelian group.

ii) $(F, \cdot)$ is an abelian group.

iii) $(a+b) \cdot c = a \cdot c + b \cdot c$ and $a \cdot (b+c) = a \cdot b + a \cdot c$ } (Distributive)

Characteristic of an integral domain:

The least positive integer such that

$$a + a + \cdots + a \ (m \text{ times}) = 0 \qquad \forall\ a \in I$$

If no such m exists, then characteristic is 0.

Claim: If F is finite, characteristic $= p$ for some prime p.

Proof: consider $1 \in F$.

consider $1,\ 1+1,\ 1+1+1,\ \ldots$

Now at some point, since F is finite, no.s begin to repeat. Let the first no. after which it reaches 0 be n. Assume to the contrary that n is not prime.

Then n can be written as a product of 2 no.s a, b. s.t. $a \neq 0 \quad b \neq 0$. $\Rightarrow (1+1+1\ldots \text{ a times}) * (1+1+1\cdots \text{ b times})$

But $\Rightarrow$ either $(1+1\cdots \text{ a times}) = 0$ or $(1+1\cdots \text{ b times}) = 0$.

This violates that n was the smallest no. that $(1+1\cdots \text{ n times}) = 0$.

Let $(F, +, \cdot)$ be a finite field. Let $F \subset K$ where K is also a finite field. Then K has q^n elements where n is dimension K over F.

$\Rightarrow$ every finite field has a prime no. of elements.

Today
- Two famous relaxations
- One universal assumption.

Shannon's Eavsdropper-
The adversary in Shannon's case is unbounded.
Lets relax these conditions-
i) The adversary is computationally bounded.
ii) There is some finite probability of error.

Negligible Function: A function $\mu(n)$ is said to be negligible in n if for all +ve polynomials $p()$, $\exists\, n_0$ s.t. $\forall\ n > n_0$ $\mu(n) \leq \frac{1}{p(n)}$

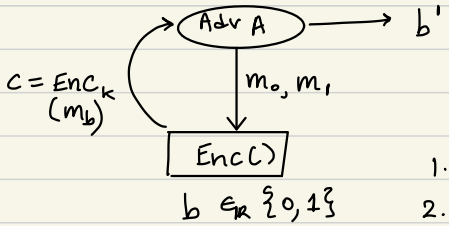
We model the adversary as a probabilistic polynomial turing machine (PPTM). $\Rightarrow$ Adv tries to break in a polynomial no. of times.

Now we know (by asymptotics)
$\forall p(), \exists n_0, \forall n \geq n_0$
$$\frac{1}{2^n} \leq \frac{1}{p(n)}$$

Notice that PPTM Adv makes the two above assumptions-
polynomial $\rightarrow$ comp bounded
probabilistic $\Rightarrow$ non-zero prob(error)

Now if the adversary uses a larger polynomial, it is sufficient to change only the key length (or other security parameter).

Equivalent Defn for Perfect Security -



$\forall$ PPTM A, $p[b' = b] \leq \frac{1}{2} + negl(n)$.

1. Adv A chooses 2 messages m_0 and m_1
2. Enc() chooses one of them at random and returns $Enc_k()$ of it.
3. A must know whether Enc chose m_0 or m_1.
4. Scheme is perfectly secure if his guess is as bad as guessing at random.

A bunch of papers in this period achieved this definition by making assumptions of the existence of various mathematical objects like - one way functions, collision resistant hash functions etc.

Further papers showed that all these are equivalent. Thus we assume - One Way Functions exist.

This is still an open problem, but if it is assumed, we can still achieve this definition.

This assumption lead to -

i) Heuristic Security
   → Algorithms that are unproven / security based on heuristics
   → Efficient for terabytes / petabytes of data.

ii) Provable security
   → Assuming one-way func exist

iii) Proven Security -
→ Empty set. :O

In this course we will focus on one-way functions and provable security

One-Way Function (Kannan's Defn)
A function $f: \{0,1\}^* \rightarrow \{0,1\}$ is said to be one-way if -
a) Easy to compute (polynomial in length of x)
b) Hard to invert -
$\forall$ PPTM A, $P[A(f(x)) = y \mid f(x) = f(y)] \leq negl(|x|)$
↳ $x = y$. We write $f(x) = f(y)$ to handle non-invertible f.

Existence of one-way functions $\Rightarrow P \neq NP$
Consider a one-way function f.

$x \longrightarrow f(x)$ is in P.

similarly,
$f(x) \longrightarrow x$ is not in P (not even in BPP)

If $f(x) \rightarrow x$ is in NP, then it $\in NP - P$ thereby separating P and NP.

Clearly this is in NP since $\exists$ a verifier which can verify in poly time - given witness y, we can compute f(y) and check if its equal to f(x).

Verifier - For a language L, V verifies L if -
$L = \{x \mid \exists w, V(x,w) \text{ accepts}\}$.

Today

- Pseudorandomness
- An encryption scheme
- Discrete Log Problem

Discrete Log Problem

Given the group $\mathbb{Z}_p^*$ and its generator g. and $y = g^x$. Find x.

↳ consider a brute force – if g is a generator, we'll have to try all p in the worst case.

Pseudorandom numbers

An efficient deterministic code G that inputs n-bit string u and outputs an $l(n)$ bit string is a pseudorandom generator (PRG) if -

a) $l(n) \geq n$ (Expansion)

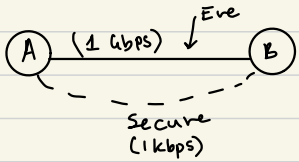
b) $\forall$ PPTM D, $|p(D(u_{l(n)} \text{ accepts}) - p(D(G(u_{(n)})) \text{ accepts})| \leq \text{negl}(n)$

↳ uniformly distr. string of length $l(n)$

↳ output of G on a uniformly distr. string of length n.

Intuition: No PPTM can distinguish bw an actually random string and one generated by the G.

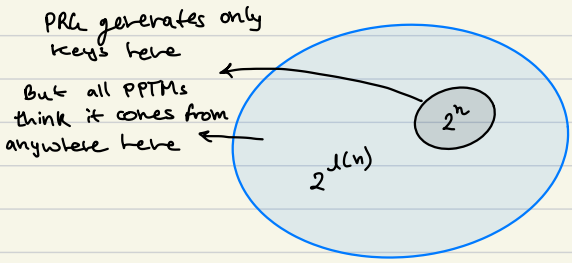
What does this give us?



I only need to send my 'seed' on the secure channel. Now using this I can generate a key of a length $l(n)$ and use that on the main channel. Eve cannot figure out if we used an actually random key from $l(n)$ ($2^{l(n)}$ possibilities)

Notice that G is deterministic. $\Rightarrow$ there will be only 2^n

keys that are actually used.



How will a distinguisher work?

Ask for samples. Notice that with $2^n + 1$ unique samples, by Pigeonhole principle, if it was pseudorandom, the no. will repeat. However, notice that no PPTM can query $2^n + 1$ times.

PRGs exist $\Rightarrow$ One-way functions exist.

consider an input s to a PRG G.

$G(s) = y$.

since G is a deterministic program. Given G and s, computing y is easy.

However since y is pseudorandom, finding s given G and y is hard. (need to try at least $2^n + 1$ where n is no. of bits in s.)

$\Rightarrow$ G is a one-way function.

If Discrete Log is a one-way func we can construct PRGs.

consider a length preserving one-way func.

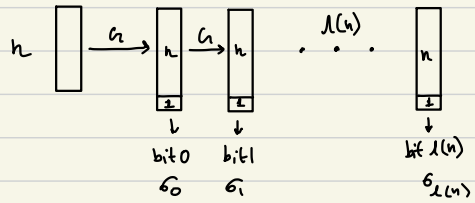
$f: \{0,1\}^n \longrightarrow \{0,1\}^n$ (i) This $\Rightarrow$ a PRG.

If I have a generator that expands by 1. i.e. $G: \{0,1\}^n \to \{0,1\}^{n+1}$

(ii) This gives us a generator $G': \{0,1\}^n \to \{0,1\}^{l(n)}$, $l(n)$ is poly in n.

(iii) DLP is a specific OWF $\Rightarrow G$.

Construction for (ii):



Assume to the contrary that $\sigma_0 \cdots \sigma_{l(n)}$ is not pseudorandom

Consider an expt where row 0 is $\sigma_0 \cdots \sigma_{l(n)}$

row 1 is $r_0 \cdots \sigma_{l(n)}$ (where r is randomly sampled)

$\vdots$

$l(n)$ is $r_0 \cdots r_{l(n)}$

σ not being pseudorandom $\Rightarrow \exists$ a distinguisher can distinguish row 0 and row $l(n)$.

Now $\exists$ i such that this distinguisher can distinguish the rows i and row i+1.

$$r_0\ r_1 \cdots r_{i+1}\ \boxed{\sigma_i}\ \cdots \sigma_{l(n)}$$
$$r_0\ r_1 \cdots r_{i-1}\ \boxed{r_i}\ \sigma_{i+1} \cdots \sigma_{l(n)}$$

$\Rightarrow$ The distinguisher can diff b/w σ_i and r_i. $\Rightarrow$ it can distinguish the ith use of G. Contradicts that G is a PRG.

Now consider an OWFs f(s) and h(s) s.t. $f: \{0,1\}^n \to \{0,1\}^n$ and $h: \{0,1\}^n \to \{0,1\}$. h(s) satisfies the prop -

h(s) $\longrightarrow$ given s, easy to find h(s).

$\longrightarrow$ given f(s), $p[A(f(s)) = h(s)] \leq \text{negl}(|s|) + \frac{1}{2}$

Consider a $G: \{0,1\}^n \to \{0,1\}^{n+1}$.

$G(s) = \text{Concatenate}(f(s), h(s))$.

Such $h(s)$ are called **Hardcore predicates of f.**

consider a hardcore predicate of DLP.

Discrete Log Most significant Bit.

Given $g^x \bmod p$ what is the most significant bit?

Thm: $DLP \leq_p DLPM$

Pf Outline

i) DLPL: Given $g^x \bmod p$ what is the LSB (x)?
Prove that $DLPL \in P$.

ii) $DLPL + DLPM \Rightarrow DLP$.

Proof

i) $y = g^x \bmod p$.

Now by Fermat's Little Thm states -

$$y^{p-1} \equiv 1 \pmod p.$$

$$\Rightarrow g^{x(p-1)} \equiv 1 \pmod p.$$

Now consider,

$y^{\frac{p-1}{2}} \bmod p$. ($p-1$ is even since p is prime)

$= g^{x\left(\frac{p-1}{2}\right)} \bmod p$

Notice if x is even, $g^{c(p-1)} \bmod p \equiv 1$ where $c = \frac{x}{2}$.

x is odd, $g^{\left(\frac{p-1}{2}\right)} \bmod p$

$= \sqrt{g^{p-1}} = \sqrt{1} = \pm 1$.

but 1 occurs at $p-1$ and its a permutation. $\Rightarrow$ must be $p-1$.

Now we can use this to solve DLPL.

Given a g^x, take $g^{x(\frac{p-1}{2})}$. If 1 LSB = 0

-1 LSB = 1.

Now an algo to find the rest of the DLP,

If x is even → take square root to get $g^{x/2}$ and repeat.

x is odd → find g^{x-1} (find multiplicative inverse g^{-1} $g^{-1} \cdot g^x = g^{x-1}$). Now repeat the even case.

We know we can find inverse in $\log n$ time.

We just need to show sqrt can be done in P. ie. we need an algo that runs in $\log n$.

The ones that do run in $\log n$, we get 2 answers, $\pm$ root.

This causes expo blow-up ⇒ beats the purpose.

If I had an algo to distinguish, i.e. is $x > \frac{p-1}{2}$

This can be found by checking if MSB of x is 1 or not.

⇒ DLPM.

⇒ These 2 together ⇒ DLP.

Today

1. CPA - Security (chosen plaintext attacks)

Pseudorandom Functions

Why? ← PRGs are not sufficient since they do not scale. Since PRGs gives a single use key.

Chosen Plaintext Attacks (CPA)

Recall that if the adversary sends two messages m_0 and m_1. The $enc(m_b)$ is given to Adv where $b \in_R \{0, 1\}$. Adv should not be able to recognize if m_0 or m_1 was chosen.

For the CPA scenario, the adversary is given oracle access to the encryption function. It is CPA secure if the adv given oracle access to enc() cannot recognize if m_0 or m_1 was chosen.

Thm: No deterministic enc() algo is CPA secure.

Pf: Let $c_0 = enc(m_0)$.

$c_1 = enc(m_1)$.

Now simply check if $enc(m_b) = c_0$, if so return $b=0$ else return $b=1$.

Notice PRG is a deterministic enc(). It is not CPA secure.

But using a non-deterministic enc() is unintuitive since decrypting it must be deterministic.

An award winning trick – Probabilistic Encryption

consider an $enc_K : \{0,1\}^n \longrightarrow \{0,1\}^n$

choose $r \in_R \{0,1\}^n$ (called a Nonce).

Set $c = \langle r, m \oplus enc_K(r) \rangle$

Now $dec_K(r, v) = v \oplus enc_K(r)$

Notice that CPA attacks won't work since each time I query the oracle, the oracle chooses some random r, and returns the pair. If this r is different the Adv can't use this info.

Modes of operation (Block Ciphers)

Notice the above method is length doubling. This makes it seriously problematic in a practical setting. We now provide better modes of operation that use the trick while being almost length preserving.

1) Cipher Block Chaining
2) Output Feedback Mode (OFB)
3) Randomized counter mode.

Output Feedback Mode.

Lets define $r_i = enc_K(r_{i-1})$.

Now all I need to provide is r_0.

i.e. $c = \langle r_0, c_1, c_2, \ldots, c_t \rangle$

Intuition – Assume r_i is generated using r_{i-1} as a seed. Thus the generated r_is are pseudorandomly generated. Even though each r_i is not actually random, no PPTM can differentiate.

Randomized Counter Mode

Notice output feed back needs sequential decoding. To decode the ith message I need to decode ones before it.
i.e. performance is stateful.

Randomized Counter uses the following -

$r_i = r_0 + i$.

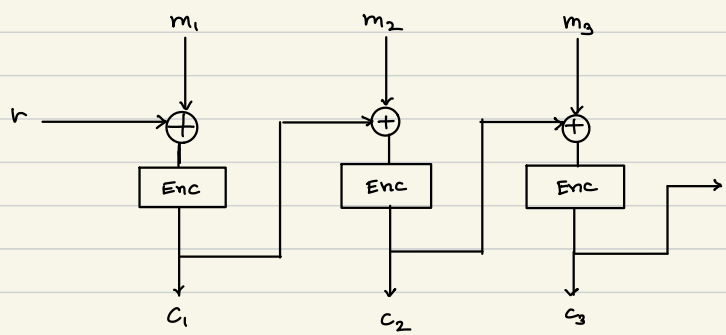
Notice that this is stateless in performance. However it is not obvious that this is secure.
proof to be done later.

Cipher Block Chaining (CBC).

For t messages $m_1, m_2, \cdots, m_t$

$< r = c_0, c_1, c_2, \cdots, c_t >$

where $c_i = enc_k(m_i \oplus c_{i-1})$



Quiz Questions (2 revealed, 2 secret)

1) a) Show that shift/vigenere cipher is perfectly secret if one alphabet / message length $\leq$ key word length is encrypted.

a) secret from class notes.

2) a) Secret from class notes.

b) If $p-1$ can be written as $p-1 = s \cdot 2^r$ where s is odd, then the $(r+1)$th LSB is a hardcore predicate for DLP.

Design a pseudorandom gererator using this.

Today

→ what are pseudorandom functions

→ PRF iff PRGs.

Consider the prob encryption $c = \langle r, R_K(r) \oplus m \rangle$

Consider a function $R_K: \{0,1\}^n \to \{0,1\}^n$

Now there are a total of 2^{n2^n} possible funcs.

Choose one of these uniformly at random.

However this needs atleast $n \cdot 2^n$ length key.

If we have a pseudorandom function, that no PPTM can differentiate b/w a truly random function and pseudorandom function.

Unlike PRGs, notice we can't even "give the PPTM" a random function since specifying it would need $2^{n \cdot 2^n}$ which is non polynomial. We'll have to argue differently.

A function $F_K: \{0,1\}^n \to \{0,1\}^n$ is said to be a PRF if

a) $\exists$ an efficient DTM that computes $F_K(x)$ (efficiency)

b) $\forall$ PPTM D, for sufficiently large n

$$p\left[D^{F_K(\cdot)}(1^n) = 1\right] - p\left[D^{R(\cdot)}(1^n) = 1\right] \leq negl(n)$$

↳ Input size is still n, but I can query the PRF and random function

PRF iff PRG

Thm: PRFs exist if and only if PRGs exist.

i) PRFs $\Rightarrow$ PRGs

$s_i = F_K(s+i)$

PRG $G(s) = F_K(s_0) \,||\, F_K(s_1) \,||\, F_K(s_2) \cdots$

↳ concatenate.

ii) PRGs $\Rightarrow$ PRFs

$G : \{0,1\}^n \longrightarrow \{0,1\}^{2n}$

Let $G_0(s)$ = Left half of $G(s)$

$G_1(s)$ = right half of $G(s)$

Here $G(s) = G_0(s) \,||\, G_1(s)$

$k \in \{0,1\}^n$, $r \in \{0,1\}^n$

$$F_k(r) = G_{r_{n-1}}\left[G_{r_{n-2}}\left[\cdots G_{r_2}\left[G_{r_1}\left[G_{r_0}(k)\right]\right]\cdots\right]\right]$$

TP: $\forall$ PPTM $\left| P\left(D^{F_K(\cdot)}(1^n)=1\right) - P\left(D^{RC}(1^n)=1\right) \right| \leq \text{negl}(n)$

Assume to the contrary that $\exists$ a PPTM D.

Base Case: For $|r| = 1$, $F_K(r) = G_{r_0}(k)$

clearly if D can distinguish $F_K(r)$ then it can distinguish G.

Inductive Hypo: We cannot distinguish upto r of length n.

Consider r of length $n+1$.

If we can distinguish $F_K(r)$

Consider an OWF $f(x) = g^x \bmod p$
(DLP).

PRG $\quad s_0 = s$

$s_i = g^{s_{i-1}} \bmod p$

$G(s) = MSB(s_1) \| MSB(s_2) \| \cdots \| MSB(s_{2n})$

$F_k(r) = G_{r_{n-1}}[\cdots G_{r_0}(k)]\cdots]$

$Enc_K(m) = \langle r, F_K(r) \oplus m \rangle$

$Enc_K(m_0 \cdots m_t) = \langle r_0, F_K(r_0 + i) \oplus m \rangle$

Notice this entire scheme depends only on a single assumption i.e. OWF exist.

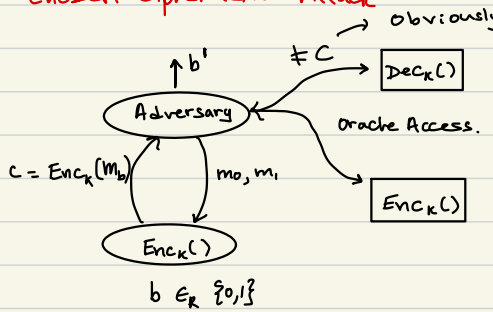
A specific OWF called Trapdoor OWFs. TOWFs gives rise to concepts like public key crypto and a bunch of other problems.

The assumption that TOWF exist gives rise to a world called "Crypto Mania".
If TOWF don't exist and OWFs exist this is "Mini crypt".
If both don't exist, but on average case hard this is "Pessiland". (On avg, problem instances are hard).
If only worst-case hard then in "Heuristica". (Hard to find a hard input)
If $P = NP$, then in "Algorithmica". This scenario is an apocalypse. Verifiers can produce proofs.

Today

- CCA security
- MAC

CCA - Chosen Ciphertext Attack



$\forall$ PPTM A , $p(b' = b) \leq \frac{1}{2} + negl(|m_0|)$

Consider our CPA secure modes of operation.

Choose m_0 as 0^n and m_1 as $1\,0^{n-1}$

$c_0 = \langle r, m_0 \oplus F_k(r) \rangle$

$c_1 = \langle r, m_1 \oplus F_k(r) \rangle$

$c' = \langle r, \text{ToggleMSB}(m_b \oplus F_k(r)) \rangle$

Now the Adv can query the Decrypt Oracle

$Dec_K(c') \xrightarrow{\text{if } b=0} M_1$

$\xrightarrow{b=1} M_0$

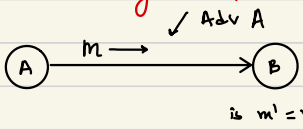
Then I can choose b' as

$$b' = \begin{cases} 0, & \text{if } Dec_K(c') = M_1 \\ 1, & \text{" } = M_0 \end{cases}$$

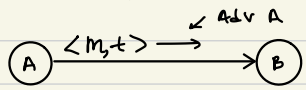
$\Rightarrow p(b' = b) = 1.$

In the real world: Adv has access to only an enc server. However this can be tweaked - flipping some bits. This can have consequences in areas such as money transfer.

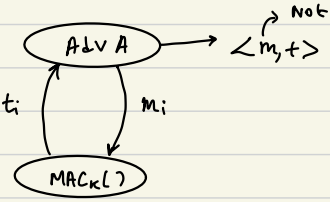
MAC - Message Authentication Codes.



B can't verify all the time. However we can define a test, s.t. not PPTM can fool the verifier.



security of MAC



$\forall$ PPTM A,

a) $m \notin Q$

b) $P[\text{Verifier}_K(m,t) = \text{Yes}] \leq \text{negl}(n)$

Encrypt Then Authenticate

if I have such MAC, I can get CCA security.

consider

$$Enc_K(m) = \langle c, t \rangle$$

c: Any CPA-secure $Enc_{K_1}(m)$

t: $MAC_{K_2}(c)$

$Dec_K(m)$:

if $verify_{K_2}(c,t) = Yes$

$Dec_{K_1}(c)$

Else

Reject.

Notice that this is CCA secure since the decryption server is useless.

Designing MAC from PRFs

$F_k : \{0,1\}^n \rightarrow \{0,1\}^n$

If message size is = key length, i.e. $\{0,1\}^n$.
Then we can use $t = F_k(m)$.

But clearly this is impractical. Consider the following failed attempts.

1. $t = F_k(\oplus_i m_i)$
   There are multiple way of getting the same XOR.

2. $t = \langle t_1 \ldots t_\ell \rangle$
   $t_i = F_k(m_i)$
   Any permutation / subset of the message blocks will pass.

3. $t_i = F_k(\text{concat}(i, m_i))$
   First notice, I can drop packets from the end and it will still pass.

4. $t_i = F_k(\text{concat}(\ell, i, m_i))$

Consider two messages -

$P_1 P_2 \ldots P_\ell$ with tag $t_1 \ldots t_\ell$

$Q_1 Q_2 \ldots Q_\ell$ $s_1 \ldots s_\ell$

$P_1 Q_2 P_3 P_4 Q_5 \ldots Q_\ell$ with tag $t_1 s_2 t_3 t_4 s_5 \ldots s_\ell$
Will pass.

There are actually 2^ℓ possible interleavings. I can therefore get a lot of information.

A correct scheme

Let $t_i = F_k(\text{concat}(r, l, i, m_i))$

r: random no. chosen for the entire message.

called the Message Identifier.

Claim: This scheme is a MAC for length n.

Proof Idea: If $\exists$ PPTM that breaks this, it breaks F_K and hence pseudorandomness of F_K is violated.

Proof: Suppose a PPTM A can produce $m_1, m_2 \dots m_l \quad t_1 t_2 \dots t_l$ ↳ not in the query set.

Notice that if I query two messages that have the same r, I can interleave the messages I can produce a valid m that is not in the query set. However prob of getting the same r is negl.

Suppose now that any one of l, m_i were changed. Then being able to find $F_K(\text{concat}(r, l, i, m_i))$ without knowing K ends up contradicting the pseudorandomness of F_K.

CBCMAC

$t_i = F_K(t_{i-1} \oplus m_i)$

let $t = t_q$ where $m = m_1 \cdots m_q$

MID Sem Qn: Show that this is not CCA-secure for variable length queries.
i.e. give two var len querys that an adv A can produce another message and its corresponding tag.

Possible Fixes – (all of these are state of art - however not used).

i) $t = F_{K_2}(t_q)$ Works even when length l is not known.
$t_q = CBCMAC_{K_1}(m)$ However key size doubles.

ii) $K' = F_K(l)$
$t = CBCMAC_{K'}(m)$

iii) Prepend l to message
$t = CBCMAC_K(m')$
$m' = concat(l, m)$

Today

Today

Today

Mid Sem Review

A) Breaking Historical Ciphers
- A.1) Shift Cipher
- A.2) Mono-alphabetic substitution cipher
- A.3) Vigenere Cipher
- A.4) Kerckhoff Principle

B) Perfect Secrecy
- B.1) Definition
- B.2) Proof that vernam cipher is P.S.
- B.3) Limitation of P.S.
  $(|K| \geq |M|)$

C) Relaxations to P.S.
- C.1) PPTM Adversary
- C.2) Negligible, non-zero error prob.
- C.3) Precise Assumptions.

D) Security against ciphertext only attacks.
- D.1) Defining Pseudorandom generators (PRGs)
- D.2) PRG $\Leftrightarrow$ OWF (designing PRG using DLP)
- D.3) Encoder $C = G(k) \oplus m$ works.

E) CPA-Security
- E.1) Defining CPA secure encryption
- E.2) No deterministic scheme is CPA secure
- F.3) Probabilistic Enc.
- E.4) Pseudorandom Functions (PRF)
- E.5) PRF $\Leftrightarrow$ PRG

E6) $C = \langle r, m \oplus F_k(r) \rangle$ works.

E7) Modes of operation.

E8) Feistel structure.

F) CCA-Secure and MACs

F1) Defining CCA security and MACs.

F2) MAC from PRF.

F3) CBCMAC

F4) CCA-Sec Enc from CPA-Sec Enc and secure MAC.

G) Hashing

G1) Define collision resistance.

G2) Birthday Attack

G3) Merkle-Damgard transform.

G4) Designing Hash functions from DLP.

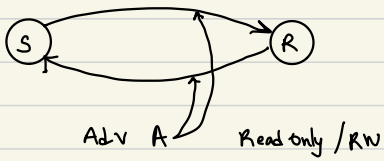
H) Key Exchange.

H1) Diffie Hellman

H2) DDH Assumption.

Mid Questions

H) Given a sender-receiver with one read only channel and one with read-write access (for the adversary). However we don't know which one is which.
Design a protocol to exchange a key regardless.



G) Design a modified Merkle-Damgard Transform which works when

$h: \{0,1\}^{3n} \longrightarrow \{0,1\}^{2n}$ ⟩ MMDT

$H: \{0,1\}^{*} \longrightarrow \{0,1\}^{2n}$. ↙

F) Break the basic CBCMAC. scheme.

i.e. show the list of messages the adversary will query to create a message that will pass MAC.

In a normal CBCMAC, replay attacks are allowed. Why are they allowed in the textbook definition. (For ex reusing the tag is not considered a valid break, since a new message must be made by the adversary.)

E) OWF $\Leftrightarrow$ PRG $\Leftrightarrow$ PRF $\Leftrightarrow$ Block Cipher.

Start from anywhere other than OWFs and build the rest.

Midsem Exc Soln

1. Shift

Gen: $K \in_R \mathbb{Z}_{26}$

M: $\mathbb{Z}_{26}^*$.

Enc: $Enc_K(m) = \text{concat}\left(m_i + k \mod 26 \right)$
$0 \le i \le |m|-1$

Dec: $Dec_K(c) = \text{concat}\left(c_i - k \mod 26 \right)$
$0 \le i \le |c|-1$

Substitution

Gen: $K \in \sigma_{26}$ (where σ_i is set of i length permutations).

M: $\mathbb{Z}_{26}^*$

Enc: $Enc_K(m) = \text{concat}\left(K(m_i) \right)$
$\forall\ 0 \le i \le |m|-1$

Dec: $Dec_K(c) = \text{concat}\left(k^{-1}(c_i) \right)$
$\forall\ 0 \le i \le |c|-1$

Vigerere

Gen: $K \in \mathbb{Z}_{26}^*$

M: $\mathbb{Z}_{26}^*$

Enc: $Enc_K(m) = \text{concat}\left(m_i + K_{i \bmod |k|} \mod 26 \right)$
$\forall\ 0 \le i \le |m|-1$.

Dec: $Dec_K(c) = \text{concat}\left(m_i - k_{i \bmod |k|} \mod 26 \right)$
$\forall\ 0 \le i \le |c|-1$.

Q2 Consider a ciphertext $C = c_0 c_1 c_2 \cdots c_{n-1}$

Split the cipher text into buckets as -

Bucket 0 $c_0\ c_t\ c_{2t} \cdots$

1 $c_1\ c_{t+1}\ c_{2t+1} \cdots$

$\vdots$

i $c_i\ c_{t+i}\ c_{2t+i} \cdots$

Now for each bucket we can do a frequency attack.
i.e. for each bucket, compute q_i as the frequency of occurance of character i in that bucket.

Let p_i be the frequency of character i in the real world.
We match the most frequent character i in the bucket with the corresponding character j in the real world. (i.e. $p_j \approx q_i$)

Q3 Definition of perfect secrecy implies -

$$p(M=m \mid C=c) = p(M=m).$$
$$p(M=m' \mid C=c) = p(M=m').$$

Consider any distribution over the message space where $p(M=m) \neq p(M=m')$. Clearly this is a distribution that violates. $p(M=m \mid C=c) = p(M=m' \mid C=c)$.

Q4 a) By the definition of negligible functions,

$\forall$ +ve polynomials p,

$\exists\ n_0$ s.t $\forall x > n_0$, $f(x) \leq \frac{1}{p(x)}$

Similarly

$\exists\ n_1$ s.t $\forall x > n_1$, $g(x) \leq \frac{1}{p(x)}$

$\Rightarrow \forall x > \max(n_0, n_1) \quad p(x) \leq \frac{1}{f(x)g(x)}$

$\Rightarrow f(x) \cdot g(x)$ is negligible.

b) Consider $f(n) = \frac{1}{2^n}$ $\quad g(n) = \frac{1}{n2^n}$ $\quad \frac{f(n)}{g(n)} = n$ which is clearly not negligible.

Q5 b) Suppose $h(x)$ is invertible.

$\Rightarrow$ I have a function computing $f^{-1} \circ f^{-1}$.

Now if I want to invert $f(x)$, I'll apply f on it and use my inverter for h.

$\Rightarrow$ f is not one way. (contradiction)

$\Rightarrow$ h must be one way.

c) $h(x_1 \| x_2) = f(x_1) \| g(x_2)$

Suppose h was invertible. Then to invert $f(x_1)$ and $g(x_2)$, I concat them and use h^{-1}. The string I receive $x = x_1 \| x_2$. I can find the split between x_1 and x_2 even if length is not known by computing $f(x_{pre})$ for every prefix of x and matching it with $f(x_1)$.

$\Rightarrow$ h must be one-way.

d) $h(x_1, x_2) = (f(x_1), x_2)$

Suppose h was invertible, then to invert $f(x_1)$, I can append some random x_2 to it and apply h^{-1}. I can then match and remove x_2 to recover x_1.

$\Rightarrow$ h must be one way.

a)

Q6. Definition of a pseudorandom generator –

$G : \{0,1\}^n \longrightarrow \{0,1\}^{l(n)}$ is a PRG if

i) $l(n) > n$

ii) $\forall$ PPTM A, $\left| P\left[A(G(U_n)) = 1\right] - P\left[A(U_{l(n)}) = 1\right] \right| \leq \varepsilon(n)$

$$P[b' = b] = P[A(y_0) = 0 \text{ and } A(y_1) = 1] \leq \frac{1}{2} + \varepsilon(n)$$

$$\Rightarrow P[A(y_0) = 0] \leq \frac{1}{2} + \varepsilon(n)$$

$$\Rightarrow 1 - P[A(y_0) = 1] \leq \frac{1}{2} + \varepsilon(n)$$

$$\Rightarrow P[A(y_1) = 1] + 1 - P[A(y_0) = 1] \leq 1 + 2\varepsilon(n)$$

$$\Rightarrow P[A(y_1) = 1] - P[A(y_0) = 1] \leq 2\varepsilon(n).$$

Q8 Assume to the contrary that $\gamma(n) > \frac{1}{2} + \text{non negl}(n)$.
Now consider a PPTM A that does -

$$A(x) = (n+1)\text{th bit of } x.$$

Now $\Pr(A(U_{2n}) = 1) = \frac{1}{2}$

But $\Pr(A(G(U_n)) = 1) = \gamma(n) > \frac{1}{2} + \text{non negl}(n)$

$\exists$ a PPTM A s.t.

$$\Rightarrow \left| \Pr(A(U_{2n}) = 1) - \Pr(A(G(U_n)) = 1) \right|$$

$$= \frac{1}{2} + \text{non-negl}(n) - \frac{1}{2}$$

$$= \text{non-negl}(n).$$

This contradicts that G is a PRG.

Q9. a) $Dec_K(C) =$

i) $m_1 = G(k) \oplus C_1$

ii) $m_2 = G(k) \oplus m_1 \oplus C_2$

b) Let the adversary behave as -

$m_0 = a \| a$

$m_1 = b \| b$

The Encryption function returns

$G(k) \oplus a \| G(k) \oplus a \oplus a$

or $G(k) \oplus b \| G(k) \oplus b \oplus b$

Notice that $C_2 = G(k)$. in both cases.

Since the adversary has the key, it can use the Decrypt function described in a).

Q10 a) Let $f: \{0,1\}^n \to \{0,1\}^n$ be a OWF.
Let h be a hard core predicate of f.
Now $G(x) = \langle f(x) \| h(x) \rangle$ is a PRG.

b) Consider PRG G. Since G can be computed in polytime by a DTM, efficiency holds.
If G was invertible. Consider a function A that given $G(s)$ outputs s. Now consider a pPTM A s.t. given a string y, uses A as a subroutine to compute s if it exists.

$$p(A(G(x))=1) = 1$$
$$p(A(U_{\ell(n)})=1) = \text{negl}(n)$$

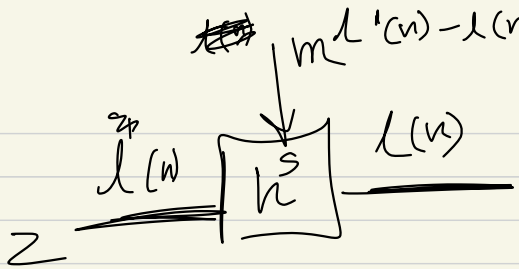
$\Rightarrow$ G is not pseudorandom (contradiction).

c) $F_k(r) = \cdots G_{r_2}(G_{r_1}(G_{r_0}(k)))$

where $G(x) = G_0(x) \| G_1(x)$
$G: \{0,1\}^n \to \{0,1\}^{2n}$

d) $L_{i+1} = R_i$
$R_{i+1} = F_k(R_i) \oplus L_i$

Q1.



a) Design a modified Merkle-Damgard Transform which works when

$h: \{0,1\}^{3n} \longrightarrow \{0,1\}^{2n}$ ⟩ MMDT

$H: \{0,1\}^{*} \longrightarrow \{0,1\}^{2n}$. ↙

F) Break the basic CBCMAC. scheme.

i.e. show the list of messages the adversary will query to create a message that will pass MAC.

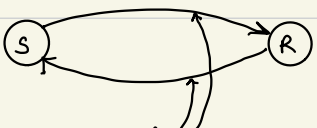
In a normal CBCMAC, replay attacks are allowed. Why are they allowed in the textbook definition. (For ex reusing the tag is not considered a valid break, since a new message must be made by the adversary.)

E) OWF ⇔ PRG ⇔ PRF ⇔ Block Cipher.

Start from anywhere other than OWFs and build the rest.

H) Given a sender-receiver with one read only channel and one with read-write access (for the adversary). However we don't know which one is which.

Design a protocol to exchange a key regardless.



Today

→ Textbook RSA
→ Some attacks
→ PKCS V1.5
→ RSA signatures — Text, Attack, Remedies.

RSA

Gen: Choose 2 large distinct primes p, q and choose (e, d) s.t. $ed \equiv 1 \pmod{(p-1)(q-1)}$, $N = pq$.

Public Key = $\langle N, e \rangle$
Private Key = $\langle d, p, q \rangle$

Enc

$m \in \{0, 1, 2, \ldots, N-1\}$

$c = m^e \mod N$. → receiver's public key.

Dec

$c \in \{0, 1, \ldots, N-1\}$

$m = c^d \mod N$. → my private key.

Correctness: $(m^e)^d \mod N$

Now Euler's extension to the FLT states

if $\gcd(a, N) = 1$

then $a^{\phi(n)} \mod N = 1$.

where $\phi(N)$ is the Euler totient function.

we know that $m^{\phi(N)} \equiv 1 \mod N$.

$$\phi(pq) = pq - q - p + 1$$

q: all mults of p; p: of q; pq: itself.

$$= q(p-1) - (p-1)$$
$$= (p-1)(q-1)$$

$$\Rightarrow m^{(p-1)(q-1)} \equiv 1 \mod N.$$
$$\Rightarrow m^{-(p-1)(q-1)} \equiv 1$$
$$\Rightarrow m^{ed} \cdot m^{-(p-1)(q-1)} \mod N \equiv m^{ed} \mod N.$$
$$\Rightarrow m^{ed - k(p-1)(q-1)} \mod N \quad \forall k.$$

but $ed \equiv 1 \mod (p-1)(q-1)$
$$\Rightarrow ed = k(p-1)(q-1) + 1.$$
$$\Rightarrow m^{k(p-1)(q-1)+1-k(p-1)(q-1)}$$
$$\equiv m \mod N.$$

Attacks. Notice that this is already not CPA secure since it is deterministic.
The smallest possible $e = 3$ since $ed \equiv 1 \mod (p-1)(q-1)$
$$\Rightarrow \gcd(e, \underbrace{(p-1)(q-1)}_{\text{even}}) = 1.$$
$\Rightarrow e$ cannot be 2.

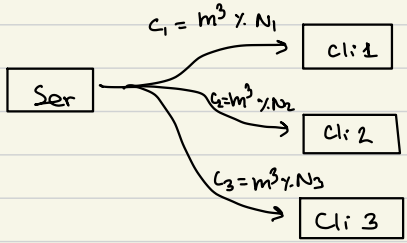
if $e = 3$, consider an attack -
$C, N, e = 3$

If $m < \sqrt[3]{N}$

$$\Rightarrow c = m^e \bmod N$$
$$= m^3 \bmod N$$
$$= m^3$$
$$\Rightarrow m = \sqrt[3]{c}$$

This is a very niche case.

Consider another case.



$$x \equiv c_1 \bmod N_1$$
$$x \equiv c_2 \bmod N_2$$
$$x \equiv c_3 \bmod N_3$$

If $\gcd(N_i, N_j) \neq 1$ we can trivially solve
why?

We know CRT says $\exists$ unique $x \in [0, \dots, N_1 N_2 N_3 - 1]$

then $m^3 = x$

$$\Rightarrow m = \sqrt[3]{x}$$

The RSA standard

Public Key Cryptography Standard (PKCS) V 1.5.

Gen: Same as RSA.

There are guidlines for choosing p and q, but we are not considering them.

Enc

$$c = \left[0100\cdots0 \;\middle|\; 000\cdots00 \;\middle|\; \longleftrightarrow \;\middle|\; 0\cdots00 \;\middle|\; m \right]^e \bmod N.$$

$\frac{\log(N)}{8}$ bytes

First Byte | Second Byte | $r \geq 8$ bytes | 1 byte.

↳ r cannot be all 0s.

i.e. r is chose UAR from sample space without all 0s.

Finding MSBs are easy → why?

$r \geq 8$ bytes so you can't really guess.

Whats the proof that this is secure — there is none!

Provable RSA — (not in practice however)

RSA – OAEP

Optimal Asymmetric Encryption Padding.

Kannan — "IMO both will be obsolete, so no point learning"

Signatures

RSA seemlessly gives a signature scheme.

$$A \xrightarrow{m, \sigma_A} B$$

can B check authenticity of not only the data (MAC) but also the sender?

i.e. we want a pair of algos –

$\sigma_A = \text{Sign}(SK_A, m)$ $\quad$ $\text{Verify}(M, \sigma_A, PK_A)$

Why do we need p, q in prv key?

RSA suggested -

$$\text{Sign}(d, m) = m^d \bmod N = \sigma$$

$$\text{Verify}(m, \sigma, \langle N, e \rangle) = \text{Yes if } \sigma^e \bmod N = m$$

Attacks -

Choose a random signature σ, compute $\sigma^e \bmod N$

Send $\langle \sigma^e \bmod N, \sigma \rangle$

This will always work out to be Yes.

Seems harmless since the signature is chosen and it may be some garbage.

Improved Attack -

Say the sender sent 2 msgs $\langle m_1, \sigma_1 \rangle$ and $\langle m_2, \sigma_2 \rangle$

Now the Adv can send $\langle m_1 m_2, \sigma_1 \sigma_2 \rangle$.

Since $\sigma_1^e = m_1 \bmod N$

$\sigma_2^e = m_2 \bmod N$

Still looks random.

Even better attack.

Consider the CPA - query m_1 and get σ_1

query $\frac{m}{m_1}$ and get σ_2

Now $\sigma_1 \sigma_2$ is the sign for m.

Hash and Sign Paradigm

$$\sigma = [H(m)]^d \bmod N$$

$$\text{Verify}(m, \sigma, \langle N, e \rangle) = \text{Yes only if } \sigma^e \bmod N \equiv H(m)$$

This looks secure.
However even this scheme is not provably secure.

Middle Ground - Tweak the model of what a secure signature means. This is called the **Random Oracle Model.**

In the Random oracle model, Hash and Sign is provably secure

Rest of the Course -

a) More PKCs
b) Proof of CPA security for El-Gamal PKC
c) El-Gamal is not CCA-secure
d) CCA-Secure PKCs.

} Book stuff

e) Secure protocols for 'everything' using
   i) Complexity Theory
   ii) Quantum Mechanics
   iii) Noisy channels
   iv) Network/Distributed systems

} Block chains using these?

Today

→ Digital Signatures
→ Zero knowledge proofs.

Authentication

Ask the user to prove that he/she knows the secret key.
Obviously revealing the key is pointless.

Aim of Digital signatures - prove that you know the secret key without actually revealing it.

Interactive Proofs -

The chalk experiment - If an Alien claims to be able to see colors that humans cannot. Consider two chalks colored white' and white". To verify if he can actually do this, consider the interaction -

i) Verifier takes 2 chalks and decides to either swap them or not.
ii) Prover answers whether the verifier actually swapped or not.
iii) If the prover could actually can differentiate, he can answer this with prob = 1.
iv) Repeat this same polynomial times. Prob that the prover answers correctly all p(n) times is $\frac{1}{2^{p(n)}}$ which is negligible.

Bit Commitment Schemes.

These schemes need two properties -

Binding - b can't be changed once chosen.

Blinding - Nobody should know which b was chosen.

Claim: OWF $\Rightarrow$ Bit Commitment Schemes

Consider an OWF $f: n \to n$. (For ex - DLP $f(x) = g^x \bmod p$)

↳ f is bijective

i) Commit Phase -

Pick a random input s.

Publish $\langle f(s), h(s) \oplus b \rangle$ ($h(s)$ is hard-core predicate)

For ex - $\langle g^s \bmod p, MSB(s) \oplus b \rangle$

ii) Reveal Phase -

Give b and s.

Check if $f(s)$ is unchanged.

If not reject.

else check if $h(s) \oplus (h(s) \oplus b) == b$

This satisfies both binding and blinding. Changes are detected in the reveal phase. But since $h(s)$ is hard to find better than $negl() + \frac{1}{2}$ when s is not known.

Graph 3-coloring

Problem is NPC. There are however exponential algorithms -

Try all 3-colorings. $O(3^{|V|})$.

Consider an interactive proof -

i) Prover permutes the 3 colors.

ii) Prover creates n locked boxes each with the color of the corresponding vertex; using some bit commitment scheme. Send these to Verifier.

iii) Verifier picks an edge u,v at random and asks to reveal c_u and c_v.

iv) If $c_u = c_v$, reject
else repeat till confident.

Completeness: If 3-colorable, verifier always accepts
Soundness: If the graph is not 3-colorable, verifier rejects with high prob.

Not 3-colorable $\Rightarrow$ $\exists (u,v) \in E$ s.t. $\chi(u) = \chi(v)$

Prob that verifier chooses it $= \frac{1}{|E|}$

$\Rightarrow$ pr of rejecting $= \left(1 - \frac{1}{|E|}\right)^k$

when experiment is done k times.

Zero knowledge – I don't get the actual coloring.

ZKP for DLP

i) Prover chooses $r \in_R \mathbb{Z}_p^*$ and sends $t = g^r \bmod p$
ii) Verifier sends the prover a challenge $c \in_R \mathbb{Z}_p^*$
iii) Prover sends $z = (cx + r)$
iv) Verifier checks if $g^z = y^c \cdot t$, repeat
else reject.

Completeness – $g^z = g^{(cx+r)}$

$= g^{cx} \cdot g^r = g^{cx} \cdot t = y^c \cdot t$

where $y = g^x \bmod p$
and y, g, p are public info.

Soundness – A prover who cannot find x has to guess it s.t. $(cx+r)$ passes the test.

Random Oracle version – (Schnorr's signature)

i) P chooses $r \in \mathbb{Z}_p^*$ and computes $H(g, p, y) = c$.
and sends $t = g^r \bmod p$. and $z = (cx+r)$.

ii) V accepts if $z = H(g,p,y)x + r$ by checking if –

$$g^z = y^{H(g,p,y)} \cdot t.$$

Digital Signature version –

prover chooses x and signs with (z, t). A verifier can verify by checking if $g^z = y^{H(g,p,y)} \cdot t$.

Today

- → secret sharing.
- → Shamir's solution
- → A generalization
- → Beautiful Relation to computing (some open problems)

Single Point of Failure vs Multiple access points -

Storing a key in a single point is dangerous. However distributing the key allows multiple access points. How do we achieve a tradeoff?

Perfect Version -

→ Split S into $S_1, S_2, \ldots, S_n$ s.t. any $\geq (t+1)$ S_is can reconstruct S.

→ Any set of $\leq t$ S_is has NO information about S.

Shamir's Secret sharing.

Define $s \in \mathbb{F}$ → Finite field. (for ex - $\mathbb{Z}_p$)

$$Q(x) = \sum_{j=0}^{t} r_j x^i \quad \text{(is a polynomial in } \mathbb{F}\text{)}.$$

s.t. $Q(0) = r_0 = S$

$r_{j>0} \in_R \mathbb{F}$. (random)

Then $\alpha_1, \alpha_2, \ldots, \alpha_n$ are distinct public elements of $\mathbb{F}$.

$\Rightarrow S_i = Q(\alpha_i)$.

You need $t+1$ points to reconstruct a t degree polynomial

→ if we have $\geq (t+1)$ S_i we can find Q.

Proof skipped for proving that $\leq (t+1)$ S_is give no information. (Similar to Reed Solomon codes).

General Access Structure -

$$\mathbb{A} \subseteq 2^{[n]}$$

- All subsets $B \in \mathbb{A}$ can access S
- Others should have no information about S.

This is clearly a generalization. If we define $\mathbb{A}$ as-

$$\mathbb{A} = \{ B \mid |B| \geq t+1 \}$$ we get the same access structure as in the previous version.

For ex- let $n = 5$.

Let $\mathbb{A} = \{ \{1,2\}, \{3,4,5\}, \{2,3\} \}$

↳ i.e. Monotone.

Closure: As long as we exclude Quantum, if $b \in \mathbb{A}$, then $\forall\ X \subseteq [n]$ s.t. $b \in X$ then $X \in \mathbb{A}$

A solution -

"An access structure can be viewed as a computation problem"- Kannan.

Consider a function $f: \{0,1\}^n \longrightarrow \{0,1\}$.

Now we know that there is a 1-1 correspondance b/w subsets and bit strings- (bit mask).

$\Rightarrow$ f can be written as the set $\mathbb{A}_f$ -

$$\mathbb{A}_f = \{ B \mid f(B) = 1 \}$$

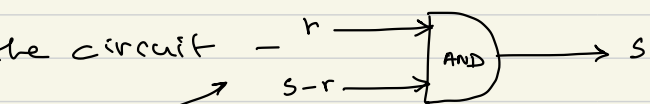
Similarly an access structure $\mathbb{A}$ defines a function f where $f(x) = 1$, if the set corresponding to x is in $\mathbb{A}$.

Notice if closure holds in $\mathbb{A}$, then $f_{\mathbb{A}}$ is a monotone function. i.e. remains 1 even if some zero bits are flipped.

Monotone function theory — can be implemented using just AND and OR gates.

AND-OR circuits for $\mathbb{A}$ to secret sharing schemes.

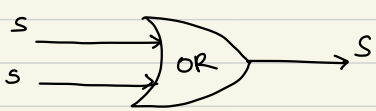
Consider the circuit —



$\mathbb{A} = \{\{1, 2\}\}$

$\mathbb{A} = \{\{1\}, \{2\}, \{3\}\}$

Consider the circuit



Given $\mathbb{A}$, we can construct a secret sharing scheme.

i) Model $\mathbb{A}$ as a boolean function $f_{\mathbb{A}} : \{0,1\}^n \longrightarrow \{0,1\}$.

ii) Design an AND-OR Boolean circuit for $f_{\mathbb{A}}$.

iii) Build the secret sharing scheme level by level.
i.e. If the highest gate is OR, set both inputs to s.
Else set inputs to s-r and r.

What if we added NOT gates?

Let Class P – set of all functions that can be implemented using AND OR – NOT circuits of poly size.

Let Class MP - Set of all functions that can be implemented using AND - OR circuits of polynomial size.

However $MP \neq P \cap$ Monotone

Consider $A \in mP$.

-> We can repeat the secret sharing shown above

OPEN: Schemes for $A \in P \cap$ Monotone
$A \notin mP$

Computational Secret Sharing -

$\forall B \in A$, there is a poly-time reconstruction algorithm and $\forall B \notin A$, $\forall$ PPTM D

$$Pr[D(\text{share}_B) = s] \leq \text{negl}(s)$$

$|s| = \log |F|$ bits

$|s_i| = |s|$

$= n \log |F|$ bits.

Consider a secret S.

$c = Enc_K(s) = C$.

Information Dispersal Algorithm (Reed-Solom)

$$n|k| + \frac{n}{t+1} \log |F|$$

OPEN: Smallest Computational Secret Scheme.

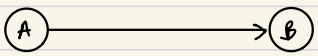
↳ No known poly-sized solution is known.

Today

→ Oblivious Transfer (OT)

→ Secure 2 party computation.

Oblivious Transfer.



index i — Array $B = [b_0 \dots b_n]$

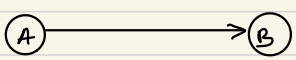
send b_i to A s.t. -

i) A doesn't know $b_{j \neq i}$

ii) B doesn't know i

Seemingly impossible - DB has to answer query correctly without knowing the query, and has to do so without giving away any extra info.

Secure 2 party communication (generalized)



x — y

Output $f(x,y)$ s.t.

i) A doesn't know y

ii) B doesn't know x

Real World: Millionaire Problem - how can 2 millionaires find out who's richer without revealing their wealth.

i.e. $f(x,y) = \begin{cases} 1, & x > y \\ 0, & o/w \end{cases}$

Notice OT is a special case of S2PC since $f(i, B) = B[i]$

Claim: A solution for OT $\Rightarrow$ A solution for any finite domain $f(x,y)$.

Let $f: D_A \times D_B \longrightarrow C$ i.e. $x \in D_A$, $y \in D_B$

Let $C = \{0,1\}$ WLOG (since we can always make $\log n$ queries for each bit).

B builds an array as follows—

Let $\phi : D_A \longrightarrow [n]$ where $n = |D_A|$.

Now the array is —

$$B_i = f(\phi^{-1}(i), y)$$ (lookup table for every possible input)

Now to compute the output of $f(x,y)$ A does an OT of $\phi(x)$ to B

$\Rightarrow$ If I have a solution for OT I can solve S2PC without revealing x or the array B (and hence y).

Inefficient — Even if x is only 100 bits, a 2^{100} array is out of reach and impractical.

Efficiently Computable Functions-

We know that the AND and XOR gates together are universal

A function is efficiently computable if there is a circuit of depth at most $\log n$. that computes f.

An efficient S2PC based on OT

Consider $x \in \{0,1\}$

$y \in \{0,1\}$

Compute Model — Find z_A on A
z_B on B
s.t. $z_A \oplus z_B = f(x, y)$.

Note it is sufficient to give a protocol to find XOR(x,y) and AND(x, y).

First lets find x_A and x_B s.t. $x_A \oplus x_B = x$.
One way to acheive this is -
A sends B $r_A \oplus x$
$\Rightarrow x_A = r_A$
$x_B = r_A \oplus x$, $r_A \in_R \{0,1\}$

Similarly for y,
$y_A = r_B \oplus y$, $r_B \in_R \{0,1\}$
$y_B = r_B$

Protocol for XOR(x, y)
$z_A = x_A \oplus y_A$
$z_B = x_B \oplus y_B$

$z_A \oplus z_B = (x_A \oplus y_A) \oplus (x_B \oplus y_B)$

$= (x_A \oplus x_B) \oplus (y_A \oplus y_B)$
$= x \oplus y$

Now if we find a protocol for AND, we can get a log n round way to solve ANY function.

Aside: Finding AND with perfect secrecy is impossible.
(proof not done)

Goal: Find z_A, z_B s.t.

$$z_A \oplus z_B = x \wedge y.$$
$$= (x_A \oplus x_B) \wedge (y_A \oplus y_B)$$

Protocol for AND (x, y)

1. B defines $z_B \in_R \{0,1\}$
2. B creates an array containing the 4 possibilities for $x_A y_A$.

x_A	y_A	z_A	
0	0	$(0 \oplus x_B) \wedge (0 \oplus y_B)$	$= x_B \wedge y_B$
0	1	$(0 \oplus x_B) \wedge (1 \oplus y_B)$	$= x_B \wedge \overline{y_B}$
1	0	$(1 \oplus x_B) \wedge (0 \oplus y_B)$	$= \overline{x_B} \wedge y_B$
1	1	$(1 \oplus x_B) \wedge (1 \oplus y_B)$	$= \overline{x_B} \wedge \overline{y_B}$

3. A determines z_A using an OT sending $x_A y_A$.

Efficient: this is as fast as computing the circuit with a constant factor due to communication.

Digression: Communication will never be faster than computation since any advances in communication would result in improvements in the short length communication in the computations datapath.

A Protocol for OT

If public key systems exist, OT can be done.

1. A creates a random array $[r_1, r_2, \ldots, r_n]$
2. A replaces the ith bit with $Enc_B(r \in_R \{0,1\})$
3. A sends the array to B.

4. B decrypts the entire array. For B everything appears random.

$$D = [\text{Dec}_B(r_1), \text{Dec}_B(r_2), \ldots, r_i, \ldots, \text{Dec}_B(r_n)]$$

5. B computes DB as $DB_i = b_i \oplus \text{Dec}_B(r_i)$

In practice $\text{Dec}_B(r_i)$ is a string whereas b_i is a bit. We have to use a hard-core predicate of the OWF used in the Enc/Dec function.

6. A obtains $b_i = (DB[i] \oplus r_i)$.

Digression: XOR of random subsets of bits of x is a hard core predicate of OWF f. (proof not done).

Note: A could have scammed b by replacing each random in the array by $\text{Enc}_B(r_i)$. Note that this means A doesn't follow the algorithm.

Solution? Make A do a zero-knowledge proof each time it generates its array. If B is not convinced, simply reject.