Computational Security

It focuses on efficient adversaries (attacker is a PPT Turing machine)
We also admit a small chance of success, where "small" = negligible.
We would like to parametrize everything by a security parameter.

<u>DEF</u> A function $\varepsilon : \mathbb{N} \to [0,1]$ is negligible if

$$\forall p(\lambda) = poly(\lambda),\ \exists \lambda_0 \in \mathbb{N} \mid \forall \lambda \geq \lambda_0\ \varepsilon(\lambda) \leq \frac{1}{p(\lambda)}$$

Equivalent: $\varepsilon(\lambda) = O(1/p(\lambda))$
$\forall p(\lambda) = poly(\lambda)$

for each polynomial, ε is smaller than its own inverse.
It is an arbitrary way to say that something is small

<u>Intuition</u>: Think of some algorithm for solving some problem.
Assume algorithm succesful w.p. p (i.e. fails w/ prob. $1-p$).

Say $p = 1/2$.

$\Pr[\text{FAIL after } k \text{ times}] \leq \frac{1}{2^k}$

Assume instead that $p = 1/\lambda$ but we don't know if the algorithm is succesful

EX. Let $p(\lambda), p'(\lambda) = poly(\lambda)$

$\varepsilon(\lambda), \varepsilon'(\lambda) = negl(\lambda)$

$c \in \mathbb{N}$

Then:

i) $p(\lambda) \cdot p'(\lambda) = poly(\lambda)$

ii) $p'(p(\lambda)) = poly(\lambda)$

iii) $\varepsilon(\lambda) + \varepsilon'(\lambda) = negl(\lambda)$

iv) $p(\lambda) \cdot \varepsilon(\lambda) = negl(\lambda)$

v) $\varepsilon(\sqrt[c]{\lambda}) = negl(\lambda)$

Idea: Exploit computational hardness.
(There are some tasks that don't run in poly times, and are really hard for Turing machines)
In particular, the fact that $P \neq NP$

For example, factoring $n = p \cdot q$ where p, q are primes with λ bits.
(it's impossible to solve polynomially)

<u>DEF</u>: ONE-WAY FUNCTION. A deterministic function $f : \{0,1\}^* \to \{0,1\}^*$ is a OWF if f can be computed in poly-time, and

$\forall$ PPT attackers $\mathcal{A}$ $\exists\ \varepsilon(\lambda) = negl(\lambda)$ s.t.

$$\Pr\left[f(x') = f(x) : \begin{array}{l} x \leftarrow \$ \{0,1\}^* \\ x' \leftarrow \$ \mathcal{A}(1^\lambda, f(x)) \end{array}\right] \leq \varepsilon(\lambda)$$

$\Rightarrow$ \$ = random uniform

Equivalent: Consider

$$\mathrm{GAME}^{owf}_{\mathcal{A}, f(\lambda)}(\lambda)$$

challenger

$$\mathcal{A}(1^\lambda) \xleftarrow{\ y\ } C(1^\lambda)$$

$$x \leftarrow \$ \{0,1\}^* ;\ y = f(x)$$

$\xrightarrow{\ x'\ }$ OUTPUT 1 if and only if $f(x') = y$

$\forall$ PPT $\mathcal{A}$ $\exists\ \varepsilon(\lambda) = negl(\lambda)$ s.t.

$$\Pr\left[\mathrm{GAME}^{owf}_{\mathcal{A}, f}(\lambda) = 1\right] \leq \varepsilon(\lambda)$$

But why 1^λ? Take $f(x) = |x|$

$$\forall x \in \{0,1\}^\lambda$$

So $|x| = \lambda$, but $|f(x)| = \log \lambda$

This function is not a OWF.

If $\mathcal{A}$ takes λ as input, it runs in time poly$(\log \lambda)$

EX. Let $f : \{0,1\}^{n(\lambda)} \to \{0,1\}^{n(\lambda)}$

Show that there exists

1) INEFFICIENT $\mathcal{A}$ breaking f w.p. $\varepsilon = 1$

2) POLY-TIME $\mathcal{A}$ breaking f w.p. $2^{-n(\lambda)} = \text{negl}(\lambda)$

$\Rightarrow$ we need $n(\lambda) = \omega(\log \lambda)$
(SUPER LOGARITHMIC)
otherwise $n(\lambda) = O(\log \lambda)$ and 2^{-n} would not be negligible

Q: Is $P \neq NP$ equivalent to assuming OWFs?

A: We don't know, but for sure if OWFs exist, then $P \neq NP$

We know that OWFs are equivalent to ONE-WAY PUZZLES (PGen, PVer)

PGen: Outputs a solved instance of PUZZLE
y = PUZZLE, X = SOLUTION

PVer: Verifies if x is solution for y.

OWFs $\Leftrightarrow$ ONE-WAY PUZZLE

The computational worlds (Rusel Impagliazzo):

1) ALGORITHMICA: P = NP
2) HEURISTICA: P $\neq$ NP
   but no average-hard puzzles (PGen outputs y but not x)

3) PESSILAND: P $\neq$ NP
   average-hard puzzles but no OWFs.

4) MINICRYPT: P $\neq$ NP, OWFs. ← we assume we're at least here

5) CRYPTOMANIA: P $\neq$ NP, OWFs but no public-key CRYPTOGRAPHY

GOAL: OWFs imply SECURE SKE beating Shannon
GOAL: OWFs imply COMP. SECURE MACs beating INF. THEORETIC lower bounds
GOAL: PUBLIC-KEY CRYPTOGRAPHY (would require more than OWFs)

PSEUDORANDOM GENERATOR (PRG)

A PRG $G: \{0,1\}^{\lambda} \to \{0,1\}^{\lambda+\ell}$ with $\ell \geq 1$
↳ STRETCH

→ Efficiently computable

→ Secure: no PPT attacker can distinguish $G(s)$ with $s \leftarrow \$ \{0,1\}^\lambda$ from $z \leftarrow \$ \{0,1\}^{\lambda+\ell}$

To formalize security:

COMPUTATIONAL INDISTINGUISHABILITY

DEF: Let $X = \{X_\lambda\}_{\lambda \in \mathbb{N}}$, $Y = \{Y_\lambda\}_{\lambda \in \mathbb{N}}$

↳ a set of random variables

We say that $X \approx_c Y$ if

distinguisher ↳ computationally close

$\forall$ PPT $\mathcal{D}$, $\exists\, \varepsilon(\lambda) = negl(\lambda)$ s.t.

$$\left| \Pr[\mathcal{D}(X_\lambda) = 1] - \Pr[\mathcal{D}(Y_\lambda) = 1] \right| \leq \varepsilon(\lambda)$$

For sufficiently large λ, the probability that the distinguisher can distinguish X from Y is negligible

The distinguisher gets a variable z picked from X or Y, and says if he thinks that $z \in X$ or $z \in Y$

For the PRG: $X_\lambda \equiv \text{REAL}_{\mathcal{D},G}(\lambda)$

$$Y_\lambda \equiv \text{RAND}_{\mathcal{D},G}(\lambda)$$

$\text{REAL}_{\mathcal{D},G}(\lambda)$	$\text{RAND}_{\mathcal{D},G}(\lambda)$
$\mathcal{D}(1^\lambda) \xleftarrow{z \in \{0,1\}^{\lambda+\ell}} \mathcal{C}(1^\lambda)$	$\mathcal{D}(1^\lambda) \xleftarrow{z \in \{0,1\}^{\lambda+\ell}} \mathcal{C}(1^\lambda)$

$$\xrightarrow{0/1} \quad \begin{array}{l} s \leftarrow \$ \{0,1\}^{\lambda} \\ z = G(s) \end{array} \qquad\qquad \xrightarrow{0/1} \quad z \leftarrow \$ \{0,1\}^{\lambda+\ell}$$

<u>DEF</u>: A PRG G is secure if $\forall$ PPT $\mathcal{D}$

$$\{\mathrm{REAL}_{\mathcal{D},G}(\lambda)\}_{\lambda \in \mathbb{N}} \approx_c \{\mathrm{RAND}_{\mathcal{D}}(\lambda)\}_{\lambda \in \mathbb{N}}$$

if you can't distinguish G from real random source, it's secure

<u>PLAN</u>: 1) OWFs $\Rightarrow$ PRGs

2) PRGs $\Rightarrow$ ONE-TIME COMP. SECURE SKE

RECALL: Perfect secrecy has two limitations:

1) $|\mathcal{K}| \geq |\mathcal{M}|$

2) ONE-TIME SECURITY

(such as the one time pad $c = m \oplus k$)

How to REMOVE 1) with a PRG?

Short random secret key $k \in \{0,1\}^{\lambda}$

Let $G : \{0,1\}^{\lambda} \to \{0,1\}^{n(\lambda)}$ $\quad n(\lambda) \gg \lambda$

$\mathrm{Enc}(k, m \in \{0,1\}^{n}) = G(k) \oplus m = c$

$$Dec(k, c) = G(k) \oplus c$$

What security? This k satisfies a meaningful notion of security

This security is ONE-TIME COMPUTATIONAL SECURITY

$$\text{GAME}^{1\text{-time}}_{\mathcal{A},\Pi}(\lambda, b \in \{0,1\})$$

$\Pi = (Enc, Dec)$

$\mathcal{M} = \{0,1\}^{n(\lambda)}$

$$\mathcal{A}(1^\lambda) \xrightarrow{m_0, m_1 \in \{0,1\}^{n(\lambda)}} \mathcal{C}(1^\lambda, b)$$

$c = Enc(k, m_b)$

$k \leftarrow \$ \{0,1\}^\lambda$

$$\xleftarrow{\quad c \quad}$$

$$\xrightarrow{\quad 0/1 \quad}$$

<u>DEF</u>: A SKE Π is ONE-TIME COMP. SECURE

if $\forall$ PPT $\mathcal{A}$, $\exists$ $\varepsilon(\lambda) = negl(\lambda)$ s.t.

$$\left\{ \text{GAME}^{1\text{-time}}_{\mathcal{A},\Pi}(\lambda, 0) \right\} \approx_c \left\{ \text{GAME}^{1\text{-time}}_{\mathcal{A},\Pi}(\lambda, 1) \right\}$$

The adversary can't know if m_0 or m_1 is encrypted!