IIC3253: Criptografía y Seguridad Computacional

Clase 1: Introducción y Conceptos Básicos

Fernando Krell
fekrell@uc.cl

Pontificia Universidad Católica de Chile

5 de Marzo 2018

Datos del Curso

- Sala de clase y horario: K-303, Lu-Mie. 17:00 - 18:20
- Página web: Piazza & `www.cs.columbia.edu/~fernando/classes/IIC3253-2018-01`
- Referencia: Introduction to Modern Cryptography, Jonathan Katz and Yahuda Lindell, 2nd Ed.
- Profesor: Fernando Krell: `fekrell@uc.cl`
  - Horario consultas: Mie 16:00 - 17:00
- Ayudantes: Geraldine Monsalve
  - e-mail: `gnmonsalve@uc.cl`
  - Horario consultas: TBA
- Evaluación:
  - Participación en clases 20%
  - 4 tareas 25%
    - Soluciones en PDF en LaTeX. 50%
    - Escrito en clases aleatorio. 50%
  - 4 lecturas con control en clases 20%
  - 2 evaluaciones 15%
  - Trabajo de investigación 20%
  - 1 examen. 30 % sobre de la nota final.

Criptografía vs Seguridad Computacional

Seguridad Computacional:

- Seguridad de redes
- Análisis de código
- Seguridad en sistemas operativos
- Sistemas de detección de intrusos
- Seguridad en "la nube"
- Métodos formales
- ...
- y... criptografía

$$\text{CRYPTO} \subset \text{SECURITY}$$

Criptografía vs Seguridad Computacional

Seguridad Computacional:

- Seguridad de redes
- Análisis de código
- Seguridad en sistemas operativos
- Sistemas de detección de intrusos
- Seguridad en "la nube"
- Métodos formales
- ...
- y... criptografía

$$\text{CRYPTO} \subset \text{SECURITY}$$

¿Qué es la criptografía?

Visión clásica de la Criptografía

Privacidad del mensage: Sólo Alice y Bob pueden entender el mensaje.
Autentificación: Bob sabe que el mensage no fue modificado.

Hoy en día, cripto es mucho más

- Cifrados basados en identidad/attributos.
- Cifrado funcional (puedo descifrar $f(m)$ y nada mas).
- Búsqueda privada de datos.
- Computación segura:
  - Protocolos de Nula Divulgación.
  - Encriptación homomórfica.
  - Obfuscación de programas.
- Cadenas de bloque.
- y más

Resumen de Probabilidad Discreta

- Sea S un conjunto *discreto*.
- Sea $p(\cdot) : S \to [0, 1]$ una función, p define un espacio de probabilidades sobre S si $\sum_{a \in S} p(a) = 1$.
- Un evento E es un subconjunto de S

$$\Pr[E] \stackrel{def}{=} \sum_{e \in E} p(e)$$

Propiedades

- $\Pr[E_1 \vee E_2] = \Pr[E_1] + \Pr[E_2] - \Pr[E_1 \wedge E_2]$
- $\Pr[E_1 \vee E_2] \leq \Pr[E_1] + \Pr[E_2]$ (cota de unión, $=$ sólo si son ind.)

Propiedades

- $\Pr[E_1 \vee E_2] = \Pr[E_1] + \Pr[E_2] - \Pr[E_1 \wedge E_2]$
- $\Pr[E_1 \vee E_2] \leq \Pr[E_1] + \Pr[E_2]$ (cota de unión, $=$ sólo si son ind.)
- $\Pr[\bigvee_{i=1}^{k} E_i] \leq \sum_{i=1}^{k} \Pr[E_i]$ (cota de unión generalizada)

Propiedades

- $\Pr[E_1 \vee E_2] = \Pr[E_1] + \Pr[E_2] - \Pr[E_1 \wedge E_2]$
- $\Pr[E_1 \vee E_2] \leq \Pr[E_1] + \Pr[E_2]$ (cota de unión, $=$ sólo si son ind.)
- $\Pr[\bigvee_{i=1}^{k} E_i] \leq \sum_{i=1}^{k} \Pr[E_i]$ (cota de unión generalizada)

Propiedades

- Sea $\{E_i\}_{i=1}^k$ tales que $E_i \cap E_j = \emptyset \; \forall i \neq j$ y $\sum_{i=1}^k \Pr[E_i] = 1$ (una partición del espacio de probabilidades). Entonces $\forall$ evento F:

$$\Pr[F] = \sum_{i=1}^{k} \Pr[F \wedge E_i]$$

- $\Pr[E_1|E_2] \stackrel{\text{def}}{=} \frac{\Pr[E_1 \wedge E_2]}{\Pr[E_2]}$ (probabilidad condicional)

$$\Rightarrow$$

Si E_1 y E_2 son independientes, $\Pr[E_1 \wedge E_2] = \Pr[E_1] \cdot \Pr[E_2]$

Propiedades

- Sea $\{E_i\}_{i=1}^k$ tales que $E_i \cap E_j = \emptyset \; \forall i \neq j$ y $\sum_{i=1}^k \Pr[E_i] = 1$ (una partición del espacio de probabilidades). Entonces $\forall$ evento F:

$$\Pr[F] = \sum_{i=1}^{k} \Pr[F \wedge E_i]$$

- $\Pr[E_1|E_2] \stackrel{\text{def}}{=} \frac{\Pr[E_1 \wedge E_2]}{\Pr[E_2]}$ (probabilidad condicional)

$$\Rightarrow$$

Si E_1 y E_2 son independientes, $\Pr[E_1 \wedge E_2] = \Pr[E_1] \cdot \Pr[E_2]$

Theorem (Bayes)

Si $\Pr[E_2] > 0$ *entonces* $\Pr[E_1|E_2] = \frac{\Pr[E_1] \cdot \Pr[E_2|E_1]}{\Pr[E_2]}$

Theorem (Bayes)

Si $\Pr[E_2] > 0$ *entonces* $\Pr[E_1|E_2] = \frac{\Pr[E_1] \cdot \Pr[E_2|E_1]}{\Pr[E_2]}$

Proof.

$$\begin{aligned}\Pr[E_1|E_2] &= \frac{\Pr[E_1 \wedge E_2]}{\Pr[E_2]} \\ &= \frac{\Pr[E_1] \cdot \Pr[E_2|E_1]}{\Pr[E_2]}\end{aligned}$$

□

Hemos usado probabilidad condicional 2 veces.

Notación asintótica

- $f(n) = O(g(n))$ si $\exists c, n'$ tales que $f(n) \leq c \cdot g(n)\ \forall n \geq n'$.
- $f(n) = \Omega(g(n))$ si $\exists c, n'$ tales que $f(n) \geq c \cdot g(n)\ \forall n \geq n'$.
- $f(n) = \Theta(g(n))$ si $f(n) = O(g(n))$ y $f(n) = \Omega(g(n))$
- $f(n) = o(g(n))$ si $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$
- $f(n) = \omega(g(n))$ si $\lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$
- $f(n) = \tilde{O}(g(n))$ si $f(n) = O(g(n) \cdot \log^c g(n))$ para alguna constante c.

Funciones negligibles

Definition

Una function $\mu : \mathbb{N} \to \mathbb{R}$ es negligible si $\forall$ polinomio $p(\cdot)$

$$\mu(n) < 1/p(n)$$

para todo n suficientemente grande. (En otras palabras $\mu(n) = o(1/p(n))$ para todo polinomio $p(\cdot)$)

Uso: Construiremos esuquemas criptográficos que aseguran que la probabilidad de que el adversario "lo quiebre" es negligible.
Propiedades:

- negligible + negligible = negligible.
  Podremos componer algunos esquemas
- **negligible x polinomio = negligible**
  Si repetimos el adversario $p(n)$ veces, entonces su ventaja es a lo más $p(n) \cdot \mu(n) = \text{negl}(n)$

Funciones negligibles

Definition

Una function $\mu : \mathbb{N} \to \mathbb{R}$ es negligible si $\forall$ polinomio $p(\cdot)$

$$\mu(n) < 1/p(n)$$

para todo n suficientemente grande. (En otras palabras $\mu(n) = o(1/p(n))$ para todo polinomio $p(\cdot)$)

Uso: Construiremos esuquemas criptográficos que aseguran que la probabilidad de que el adversario "lo quiebre" es negligible.
Propiedades:

- negligible + negligible = negligible.
  Podremos componer algunos esquemas
- **negligible x polinomio = negligible**
  Si repetimos el adversario $p(n)$ veces, entonces su ventaja es a lo más $p(n) \cdot \mu(n) = \mathsf{negl}(n)$

Aleatoriedad

- Algoritmo determinista: Siempre arroja la misma salida para la misma entrada.
- Algoritmo aleatorizado: Toma decisiones aleatorias, y puede arrojar distintas salidas para una misma entrada.
  Ejemplo:

```
def CS():
  bit = rand(0,1)
  if bit == 0:
   return "cara"
  else
    return "sello"
```

- $\Pr[CS() = \text{cara}] = 1/2$. probabilidad tomada sobre las decisiones aleatorias de CS()
- Def. equivalente: Alg. aleatorizado es un algoritmo determinista con entrada extra de cierto largo uniformemente aleatoria. Ej, $CS(r)$: si r es 0 return “cara”, si no return “sello”

Ejemplo: Igualdad de polinomios

- Dado polinomios $P(x)$, $Q(x)$ sobre un conjunto S, determinar si son equivalentes.
- Algoritmo aleatorizado:

```
son_equiv(P,Q):
    x = rand(S)
    if P(x)!=Q(x)
     return False
    return True
```

- Si $P \equiv Q$, entonces $\Pr[\texttt{son\_equiv}(P,Q) = \texttt{True}] = 1$
- Si $P \not\equiv Q$, entonces $\Pr[\texttt{son\_equiv}(P,Q) = \texttt{True}] \leq d/|S|$ (Schwartz-Zippel Theorem)

Cifradores antiguos

- Cifrador del César: Rotar alfabeto en 3 posiciones:

a	b	c	d	...	w	x	y	z
D	E	F	G	...	Z	A	B	C

Ejemplo:

Enc(`criptografia`) = `GVMTXSKVEJME`

- Cifrador de desplazamiento: Usar una *llave* en vez de cosntante.

$$\text{Enc}(k, X) = (X + k)$$

Ejercicio: Cuál es la llave que descifra YNYYNIRRFGERPR

Cifradores antiguos

- Cifrador del César: Rotar alfabeto en 3 posiciones:

a	b	c	d	...	w	x	y	z
D	E	F	G	...	Z	A	B	C

Ejemplo:

Enc(`criptografia`) = `GVMTXSKVEJME`

- Cifrador de desplazamiento: Usar una *llave* en vez de cosntante.

$$\text{Enc}(k, X) = (X + k)$$

Ejercicio: Cuál es la llave que descifra YNYYNIRRFGERPR

- Cifrado mono-alfabético: Cada letra es reemplazada por otra.

a	b	c	d	...	w	x	y	z
T	X	A	F	...	G	Y	S	R

¿Cuál es la llave en este esquema?

Cifradores antiguos

- Cifrador del César: Rotar alfabeto en 3 posiciones:

a	b	c	d	...	w	x	y	z
D	E	F	G	...	Z	A	B	C

Ejemplo:
Enc(`criptografia`) = `GVMTXSKVEJME`

- Cifrador de desplazamiento: Usar una *llave* en vez de cosntante.

$$\text{Enc}(k, X) = (X + k)$$

Ejercicio: Cuál es la llave que descifra YNYYNIRRFGERPR

- Cifrado mono-alfabético: Cada letra es reemplazada por otra.

a	b	c	d	...	w	x	y	z
T	X	A	F	...	G	Y	S	R

¿Cuál es la llave en este esquema?

- Cifrador de Vignère: Usar diferentes desplazamientos, luego repetir.
Ejemplo: Llave es 3 6 7 8. Enc(`seguridad`) =`VKNCUÑKIL`
Tarea: describir como quebrar este cifrador.
Caso1:El largo de la llave es conocido.
Caso2:El largo de la llave es desconocido.

Cifradores antiguos

- Cifrador del César: Rotar alfabeto en 3 posiciones:

a	b	c	d	...	w	x	y	z
D	E	F	G	...	Z	A	B	C

Ejemplo:
Enc(`criptografia`) = `GVMTXSKVEJME`

- Cifrador de desplazamiento: Usar una *llave* en vez de cosntante.

$$\text{Enc}(k, X) = (X + k)$$

Ejercicio: Cuál es la llave que descifra YNYYNIRRFGERPR

- Cifrado mono-alfabético: Cada letra es reemplazada por otra.

a	b	c	d	...	w	x	y	z
T	X	A	F	...	G	Y	S	R

¿Cuál es la llave en este esquema?

- Cifrador de Vignère: Usar diferentes desplazamientos, luego repetir.
Ejemplo: Llave es 3 6 7 8. Enc(`seguridad`) =`VKNCUÑKIL`
Tarea: describir como quebrar este cifrador.
Caso1:El largo de la llave es conocido.
Caso2:El largo de la llave es desconocido.

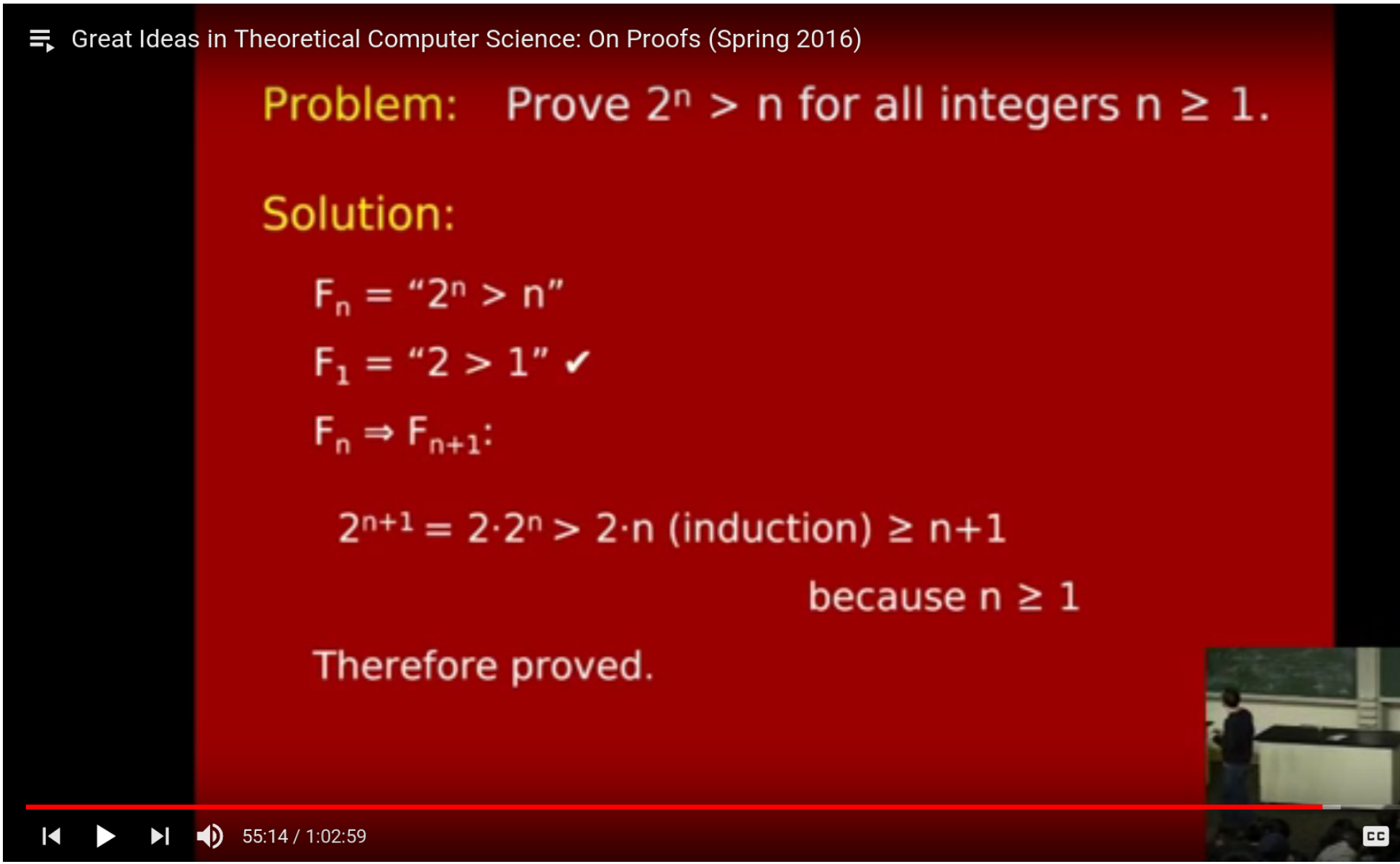
Ahora criptografía *en serio*!

1. Principio de Kerckhoffs: Algoritmo *público*, llave secreta.
2. Seguridad demostrable: Definición, teoremas y demostraciones.
3. $P = NP \Rightarrow$ no hay encriptación*. Adversario podría "*adivinar*" la llave y chequear que es la correcta.
4. Supuesto básico: $P \neq NP$. Es una condición necesaria, es suficiente? Problema abierto.
   - Necesitamos problema NP-completo difícil en la mayoria de los casos. NP-completitud es sólo análisis de peor caso.
5. ¿Cuál condición es suficiente?
   - Depende para qué.
   - Criptografía simétrica: Funciones unidireccionales.
   - Criptografía asimétrica: Permutaciones con "puerta trasera".
   - Hay más ....

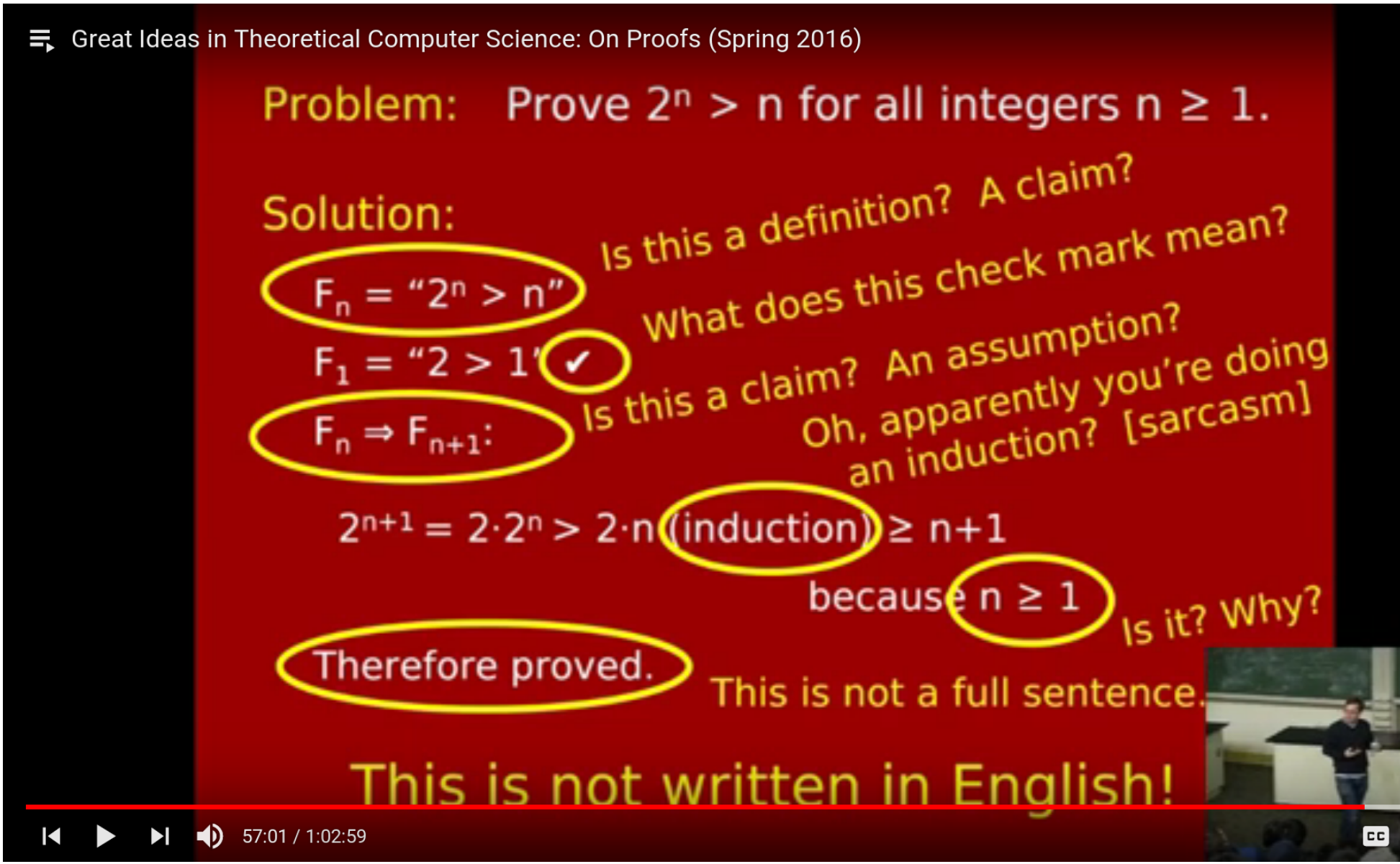
Temario

1. Seguridad Perfecta.
2. Criptografía simétrica.
   - 2.1 Definiciones.
   - 2.2 Generadores pseudo aleatoreos y "stream ciphers".
   - 2.3 Cifradores de Bloque. construcción prácticas (AES) y teórica.
   - 2.4 Modos de operacion
   - 2.5 Autenticación. MACs, HMAC, etc.
   - 2.6 Cifrados autenticados.
3. Criptografía asimétrica.
   - 3.1 Teoría de números.
   - 3.2 Cifradores asimétricos: RSA, ElGamal, Rabin, Paillier.
   - 3.3 Firmas digitales.
4. Seguridad de redes
   - 4.1 Infraestructura de clave pública.
   - 4.2 TLS, IPSEC, DNSSEC, ...
5. Tópicos avanzados
   - 5.1 Computación segura multipartita.
   - 5.2 Protocolos de nula divulgación.
   - 5.3 Cifradores homomórficos.
   - 5.4 Cifradores funcionales: IBE, ABE, etc.

Una mala solución



Razón


Great Ideas in Theoretical Computer Science: On Proofs (Spring 2016)
Problem: Prove $2^n > n$ for all integers $n \geq 1$.
Solution:
F_n = "$2^n > n$"
F_1 = "$2 > 1$" ✔
$F_n \Rightarrow F_{n+1}$:
$2^{n+1} = 2 \cdot 2^n > 2 \cdot n$ (induction) $\geq n+1$
because $n \geq 1$
Therefore proved.
Is this a definition? A claim?
What does this check mark mean?
Is this a claim? An assumption?
Oh, apparently you're doing an induction? [sarcasm]
Is it? Why?
This is not a full sentence.
This is not written in English!
57:01 / 1:02:59

Tarea0

- Crear llave GPG para su correo electrónico: Utilizar thunderbird y enigmail plugin. Enviar correo al profesor y ayudante con la llave pública.
- Ver clase "On Proofs" de Prof. Ryan O'Donnell en curso "Great Theorical Ideas in Computer Science" (youtube).