CS 6846: Quantum Algorithms and Cryptography

Lec 8: Building Cryptography

Shweta Agrawal
IIT Madras

Some Questions

1. How can we build these things?

Some Questions

1. How can we build these things?

2. What guarantees can we have ?

Some Questions

1. How can we build these things?

2. What guarantees can we have ?

3. How do we move from messy real world scenarios to clean mathematical definitions?

Some Questions

1. How can we build these things?

2. What guarantees can we have ?

3. How do we move from messy real world scenarios to clean mathematical definitions?

4. How do theorems in math say anything about real world attacks?

Principles of Crypto Design [Katz-Lindell]

1. Formulate a rigorous and precise definition of security for cryptosystem – security model.

2. Precisely formulate the mathematical assumption (e.g. factoring) on which the security of the cryptosystem relies.

3. Construct cryptosystem (algorithms) and provide proof (reduction) that cryptosystem satisfying security model in (1) is as hard to break as mathematical assumption in (2).

1: Security Model

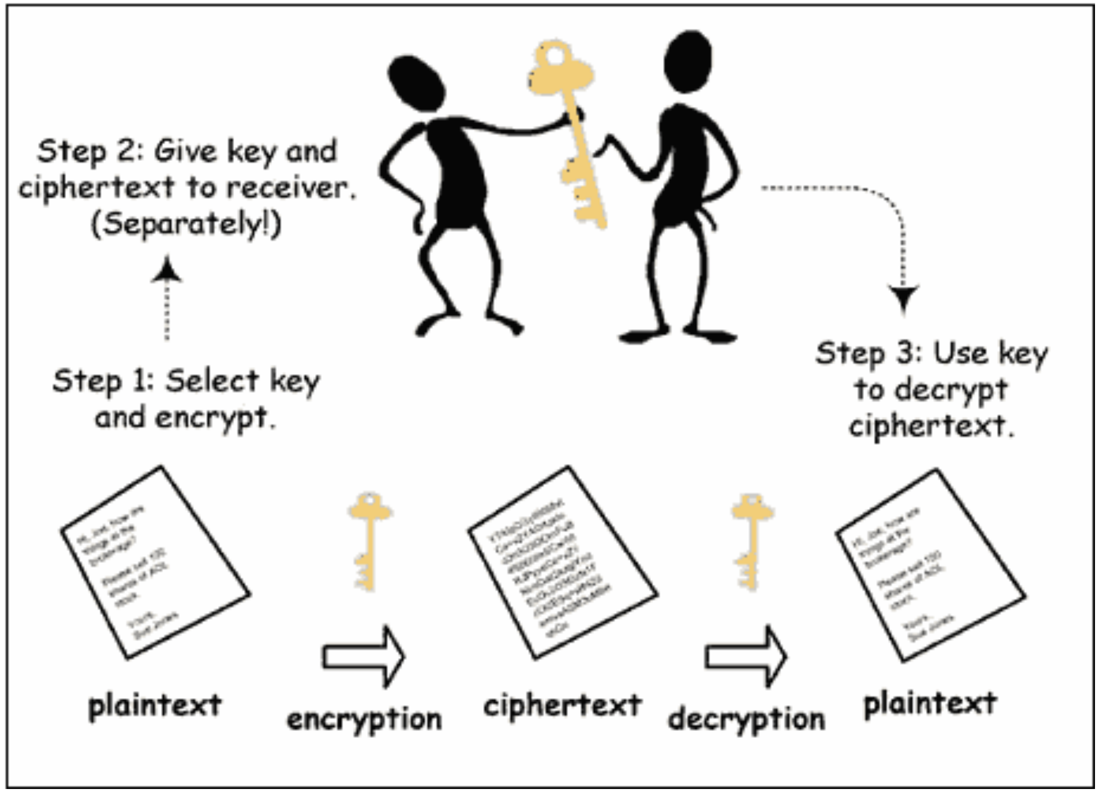
Real world attacks

Crypto
Proofs

Security Model : Mathematical definition that scheme has to satisfy

Scheme achieves security in given model = Scheme secure against attacks captured by that model

Case Study : Secure encryption



- Every pair of users must share a unique secret key
- Need key to encrypt and decrypt
- Intuitively, only holder of secret key should be able to decrypt

Case Study : Secure encryption

Syntax

We must construct the following algorithms:

1. Keygen : Algorithm that generates secret key K

2. Encrypt(K,m) : Algorithm used by Alice to garble message m into "ciphertext" CT

3. Decrypt(K, CT) : Algorithm used by Bob to recover message m from ciphertext CT.

Case Study : Secure encryption

How should security of encryption be defined?

Answer 1 : Upon seeing ciphertext, Eve should not be able to find the secret key.

But our goal is to protect the message!

Consider encrypt algorithm that ignores the secret key and just outputs the message. An attacker cannot learn the key from the ciphertext but learns the entire message!

Case Study : Secure encryption

Answer 2 : Upon seeing ciphertext, Eve should not be able to find the message.

Is it secure intuitively to find 99% of the mesg?

Answer 3 : Upon seeing ciphertext, Eve should not be able to find a single character of the message.

Is it ok to leak some property of the mesg, such as whether $m > k$?

Case Study : Secure encryption

Answer 4 : Any function that Eve can compute given the ciphertext, she can compute without the ciphertext.

Still need to specify :

- Can Eve see ciphertexts of messages of her choice?
- Can Eve see decryptions of some ciphertexts?
- How much power does she have?

Case Study : Secure encryption

Answer 4 : Any function that Eve can compute given the ciphertext, she can compute without the ciphertext.

Still need to specify :

- Can Eve see ciphertexts of messages of her choice?
- Can Eve see decryptions of some ciphertexts?
- How much power does she have?

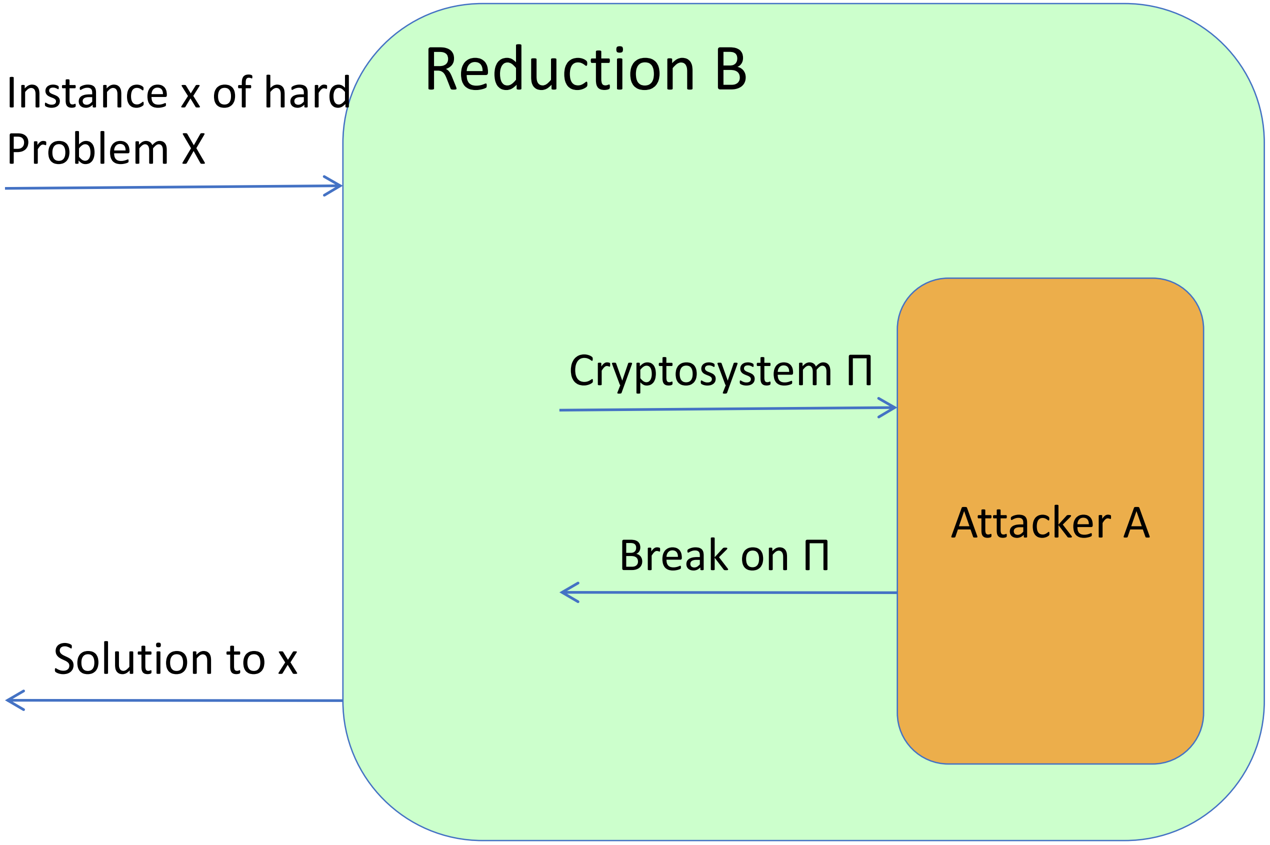
2: Mathematical Assumption

- Trivial assumption : my scheme is secure

- Use minimal assumptions
  - Existence of one way functions

- Use well studied assumptions
  - Examples: factoring, discrete log, shortest vector problem etc…

3: Reduction



Instance x of hard
Problem X
Reduction B
Cryptosystem Π
Attacker A
Break on Π
Solution to x

3: Reduction

Show how to use an adversary for breaking primitive 1 in order to break primitive 2

Important :

- Run time: how does T_1 relate to T_2
- Probability of success: how does $Succ_1$ relate to $Succ_2$
- Access to the system 1 vs. 2

SKE: The Simplest Construction

One Time Pad.

Let message is ℓ bits

Let SK be ℓ bits also.

$CT = m \oplus SK$. ← one time use

$Dec(CT, SK) = CT \oplus SK$

$= m$.

Security: Perfect

Shannon Lower Bound: Key length is inherent..

SK: 100 bits.

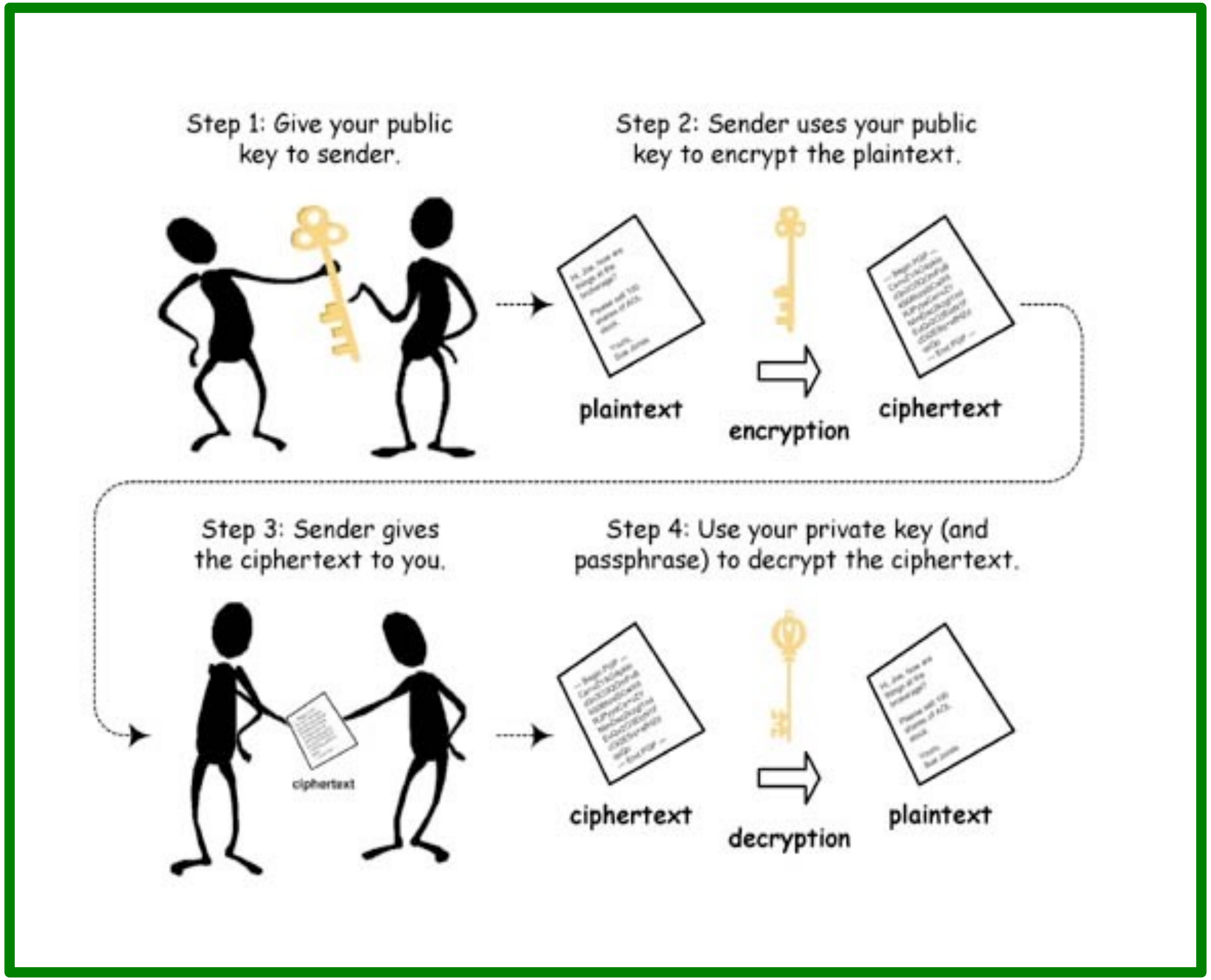
Enc(SK, m) : (1) "Expand" SK to 1000 bits. → Pseudo random generators.
(SK: 100 bits)

(2) Use OTP construction on part of SK.

Dec(SK, m) (1) Mimic step 2
(2) As in OTP.

X OWF ⇒ PRGs. ⇒ PRFs.

Public Key Encryption



What we need…

1. Invertible: It must be possible for Alice to decrypt encrypted messages.
2. Efficient to compute: It must be reasonable for people to encrypt messages for Alice.
3. Difficult to invert: Eve should not be able to compute m from the "encryption" f(m).
4. Easy to invert given some auxiliary information: Alice should restore m using SK.

What we need…

1. Invertible

1. Efficient to compute
2. Difficult to invert

} One way functions!

3. Easy to invert given some auxiliary information

What we need…

1. Invertible

1. Efficient to compute

2. Difficult to invert

3. Easy to invert given some auxiliary information

(Items 1–2 above, bracketed:) One way permutations!

What we need…

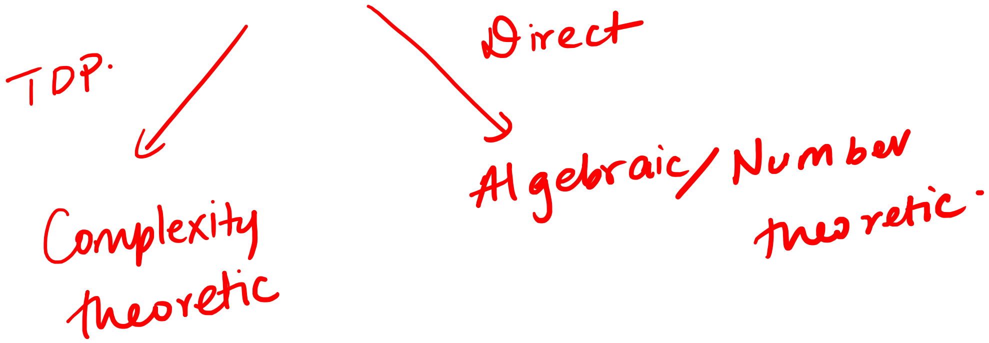
0. Invertible
1. Efficient to compute
2. Difficult to invert
3. Easy to invert given some auxiliary information

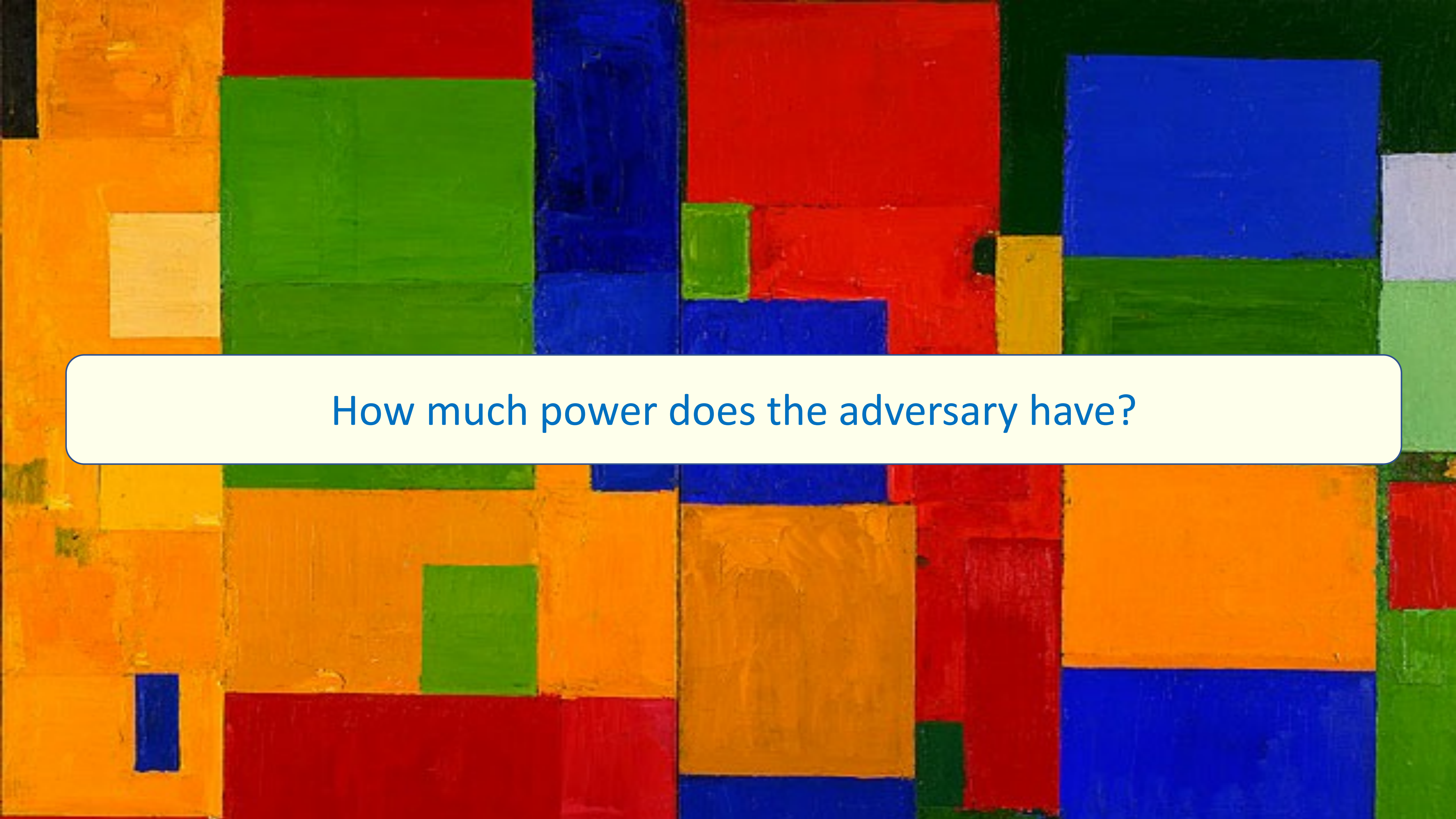
Trapdoor permutations! → Enc.

Public Key

Enter RSA.

How to build PKE?





How much power does the adversary have?

Classical Setting.

Polynomial time : If an algorithm A gets an input of size k, it is considered poly time if it runs in $O(k^c)$ steps, where c is a constant.

Prob. poly time : As above but algo. can be randomized.

$$Y \leftarrow A(\text{input}, \text{randomness}) \qquad |\text{input}| = k.$$

Y is a Random Variable.

Negligible functions:

A function $\nu(k)$ is negligible, denoted by $negl(k)$, if

$\forall c > 0, \exists k'$ s.t. $\forall k \geq k'$

$$\nu(k) \leq \frac{1}{k^c}.$$

Security Parameter:

"size of the problem" "k"

Honest Algorithms are PPT in k

Tradeoff bet^n efficiency & security.

Prob. of Attacker winning is negl(k).

eg: $\frac{1}{2^k}$.

How small is $1/2^k \Leftrightarrow$ How big is 2^k?

freemars.org: If I started with 0.1 mm thicknes then after 100 iterations, I get

2^{100}: 13.4 billion light years height.

2^{150}: Billions of supercomputers running for the age of universe "maybe" do these many steps.

2^{240} : # elementary particles in the observable universe.

→ If I multiply 2 numbers of 200 bits each. VERY FAST.

→ If I want to factor a 200 bit number best algo takes time 2^{100}

ONE WAYNESS.