Cryptographic Primitives

Building blocks, common terms, alphabetic crypto

Defining Common Terms

Terms used through the course:

- Plaintext – unencrypted “plain” message
- Key – Secret input used to encrypt plaintext
- Encryption Algorithm: Steps to make plaintext un-intelligible using the key; scrambling the plaintext
  - Takes as input: Plaintext, Key
  - Output: Ciphertext

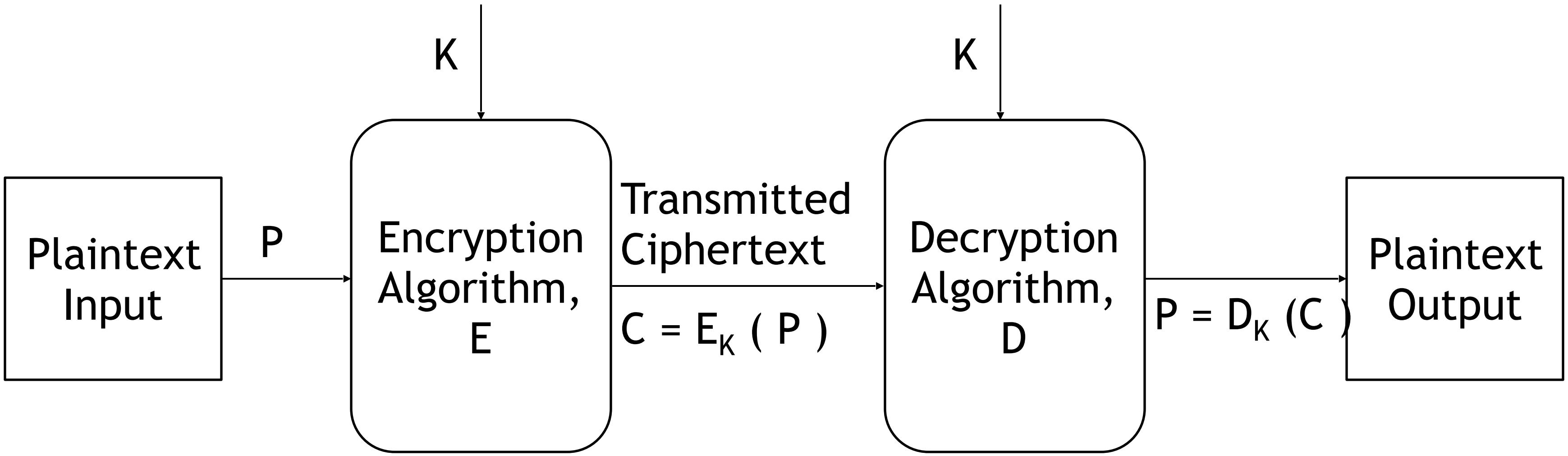
Defining Common Terms

- Ciphertext: Un-intelligible, encrypted message

- Decryption Algorithm: De-scrambling the plaintext; steps to make ciphertext intelligible using the key
  - Takes as input: Ciphertext, Key
  - Output: Plaintext

- Secret: Key
- Public: Encryption/decryption algorithms, ciphertext

Cryptosystem Model



Encryption Building Blocks

- Substitution and Transposition – building blocks of encryption algorithms
- Substitution – mapping elements
- Transposition – re-arranging elements

Early Attempts at Cryptography

- Caesar cipher: Shift letters of alphabet
- E.g., right shift by 3:

Old letter	New “encrypted” letter
A	D
B	E
C, …	F, …
… , Z	… , C

- Vigenère cipher: Multiple rounds of Caesar cipher. Each alphabet separate key
- Others: Playfair, Rotary machines,…

Early Crypto

- Rail Fence Cipher: Zig-zag cipher

a ….. o ….. y ….. t ….. a ….. x

. r . y . u . r . i . g . o . e . d . e

…e …… t …… n…… r …… m

- “are you … x”

- Toy ciphers – can’t be used for serious stuff
- But, can be used for web-page spoofs, story spoilers!

Early Crypto

- Substitution ciphers, e.g., Caesar cipher were used until recently
- WW II German Enigma machine, broken by the Allies

- Disadvantages
  - By and large, uses English alphabet (only 26 letters)
  - Bigger, fundamental problem is each letter always encoded same way – we'll see more on this later

Breaking Crypto

- Goal of Breaking
  - *Try to deduce key*
  - *Deduce plaintext(s)*

- Cryptanalysis
  - Try out sample plaintexts, or ciphertexts or both
  - Look for patterns or structure in the ciphertext

- Exhaustive search or Brute-force attacks
  - Try all possible combinations of key
  - On an average, for X keys, need to try $X/2$ keys

Block Ciphers

Formal definition:

$$E : \{0,1\}^k \times \{0,1\}^l \rightarrow \{0,1\}^l$$

Where:

- $\{0,1\}^x$ denote binary bitstrings of length x, e.g., $\{0,1\}^5 = 001010$
- k denotes key length (number of bits in key)
- l denotes block length (number of bits in a block)

Block Ciphers

Parametrizing E: Given key *K* and plaintext *M*

$$\underbrace{E_K}_{\text{k bits}} \quad \underbrace{(M)}_{\text{l bits}} \qquad = \underbrace{(K, M)}_{\text{l bits}}$$

We say that *E* is a block cipher if:

- For every *E*, there exists an E^{-1}
- *E* and E^{-1} are efficiently computable (polynomial time)
- $E(K,M) = E^{-1}(K,M)$

Block Ciphers

- Block cipher: Message divided into groups of bits called "blocks". Each block encrypted by same key

E, D, K

$c_1, \ldots, c_n$

$M = p_1 || p_2 || \ldots || p_n$

$E_K(p_1) = c_1$

…

$E_K(p_n) = c_n$

$D_K(c_1) = p_1$

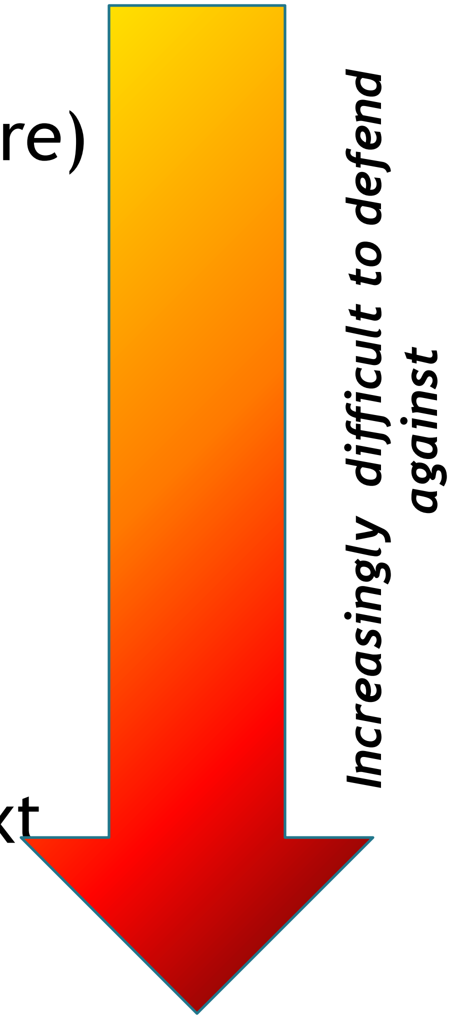
…

$D_K(c_n) = p_n$

$D_K(x) = E_K^{-1}(x)!$

Attacks

- Known Ciphertext attack (least severe)
- Known Plaintext attack
- Chosen Plaintext attack
- Chosen Ciphertext attack
- Chosen Plaintext + Chosen Ciphertext attack (most severe)



Principles of Modern Crypto

- Why need formalisms and models?
- Historical approach: Design-find attack-then-patch
- No agreed-upon design principles, security requirements, no evidence of security
- No systematic design

Principles of Modern Crypto

- Most crypto schemes are based on some assumptions
- Precisely state assumptions
- Prove, w.r.t., assumptions that a given scheme is "secure"
- Breaking scheme at least as hard as breaking assumption, in polynomial time

Principles of Modern Crypto

- Proof of security has to be in a well-defined model
- Model guarantees that any scheme proven secure within its parameters is secure in some sense (e.g., IND-CPA secure, IND-CCA secure, …)

Computational Security

- In modern crypto, security is defined in a computational sense
  - E.g., “This algorithm leaks info with Pr[2^{-60}] over 1000 years on the fastest supercomputer of today” --- Mostly good enough for the real world
  - Success probability of adversary negligible, but not 0
  - ... As opposed to *perfect* or *information-theoretic* security: “This algorithm leaks info with Pr[0] over infinite years on machines of infinite speed/ memory”

Polynomial-time Adversaries

- Modern crypto only considers polynomial-time adversaries
  - Polynomial = bounded time and bounded space
  - Limited speed and memory

- Budget for fastest (adversary) supercomputers of the day, but not infinitely powerful adversaries

Computational Indistinguishability

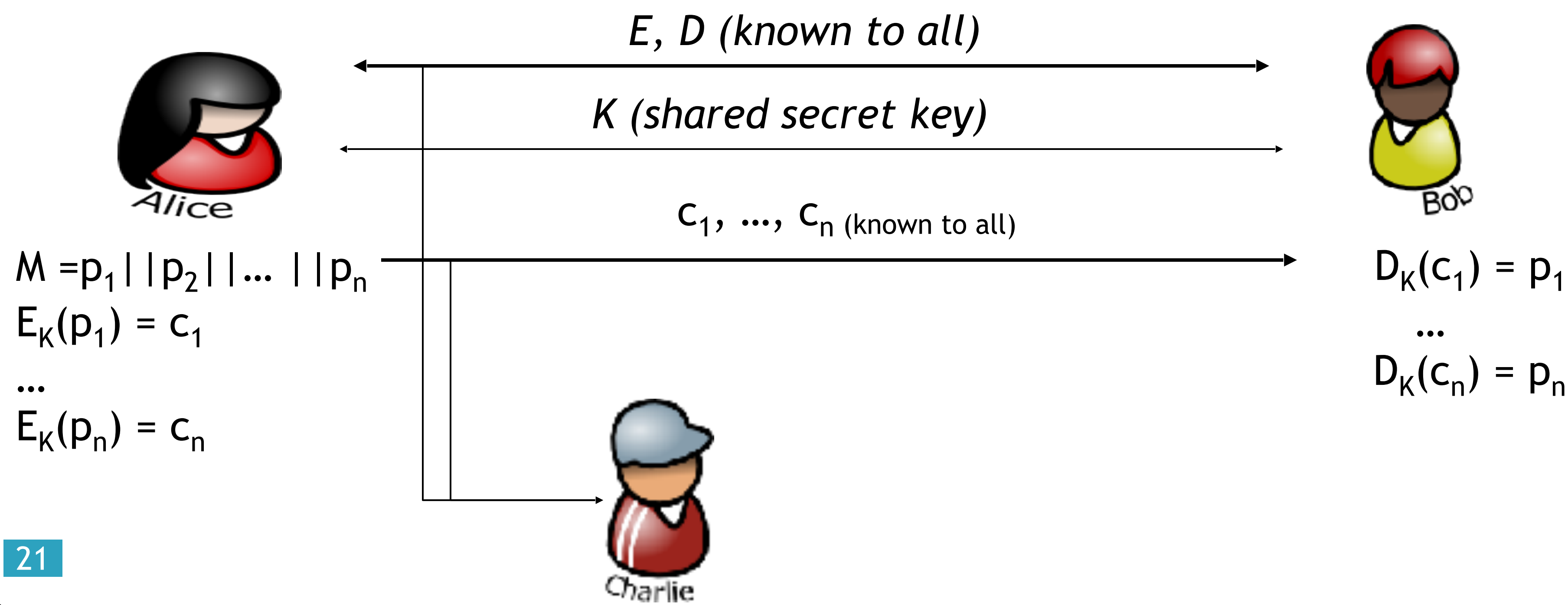
- Stems from concepts in complexity theory
- Definition: Two probability distributions X and Y are *computationally indistinguishable* if no polynomial- time adversary A can tell them apart:
- $Pr[A_{str \in X}(str) = 1] – Pr[A_{str \in Y}(str) = 1] \approx 0$
- *Tends* to 0, e.g., $\approx 2^{-60}$
- Any computationally indistinguishable crypto scheme will be *computationally secure*

Computational Indistinguishability

- For our purposes, X = real cipher text, Y = random string
- Or, X = real ciphertext of a known plaintext, Y = random string
- Point is adversary shouldn't be able to distinguish between real and random

Indistinguishability against Known Ciphertext Attack (IND-KCA)

- Charlie's given: ciphertext, encryption, decryption algorithm
- Charlie's goal: deduce $p_1, \ldots p_n$, or better, K
- Simplest possible attack to mount



Indistinguishability against Known Ciphertext Attack (IND-KCA)

- Make assumptions only about Charlie's (adversary) capabilities, not strategies
- Charlie is a PPT algorithm (probabilistic polynomial time)
- Accounts for any polynomially-bounded adversary
- IND-KCA game (experiment):
  - Charlie generates m_0, m_1, gives to Alice
  - Alice chooses m_b, does $C_b \leftarrow E_K(m_b)$
  - C_b given to Charlie

Indistinguishability against Known Ciphertext Attack (IND-KCA)

- Charlie outputs a guess, b'
- If Pr[b == b'] >> 1/2, Charlie wins the game

- We assumed single-block messages, can be generalized to messages with n encrypted blocks (e.g., $c_1,\ldots c_n$)
- C_b is called the "challenge" ciphertext
- $|m_1| = |m_2|$
- Charlie can use whatever strategy he wants

Indistinguishability against Known Ciphertext Attack (IND-KCA)

- Advantage of Charlie:

$$\Pr\,[A(n);\ b = b'] \le 1/2 + \varepsilon$$

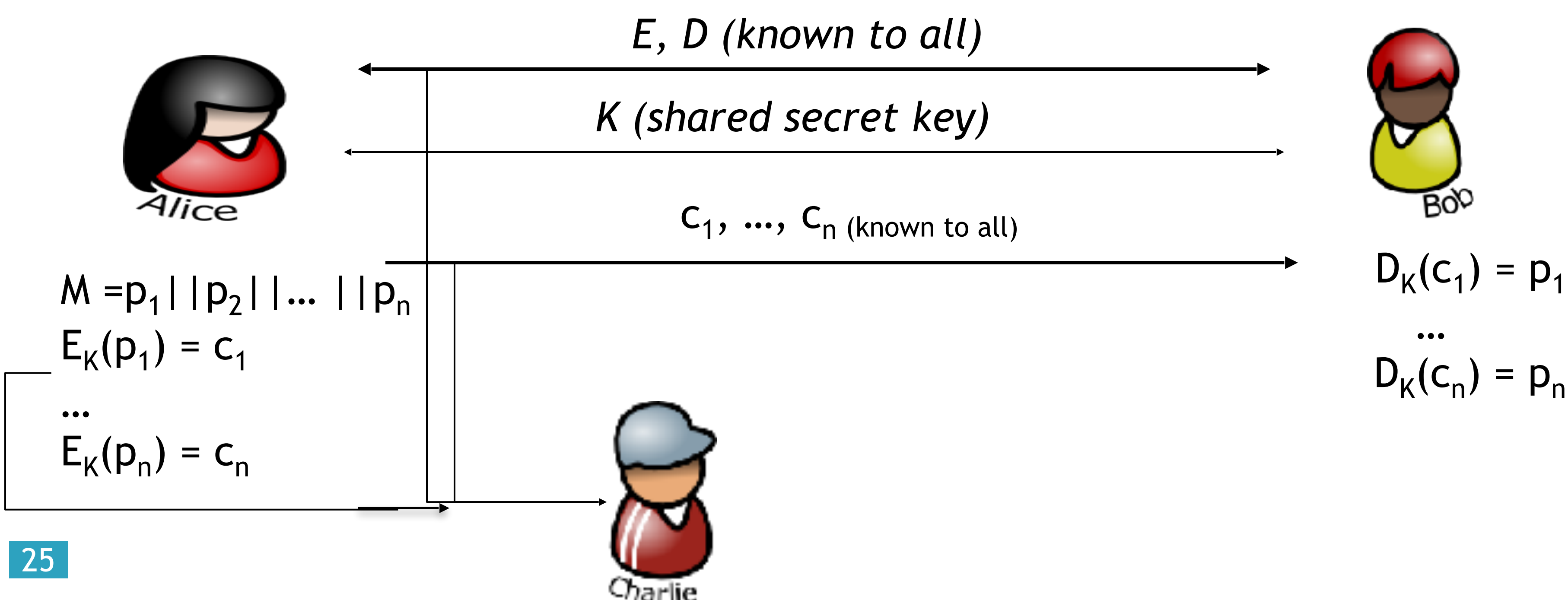
- ε is a negligible quantity, n is a security parameter, A is the adversary

- Alternative advantage definition:

$$\Pr[A(m_0);\ b{=}b'] - \Pr[A(m_1);\ b{=}b'] \le \varepsilon$$

Indistinguishability against Known Plaintext Attack (IND-KPA)

- Charlie's given: ciphertexts, E,D, plaintext-ciphertext pairs
- Charlie's goal: deduce encryptions of plaintexts not in $(p_1,\ldots p_n)$



Indistinguishability against Known Plaintext Attack (IND-KPA)

- Accounts for any PPT adversary

- IND-KPA game (same as IND-KCA)
  - Charlie generates m_0, m_1, gives to Alice
  - Alice chooses m_b, does C_b <— $E_K(m_b)$
  - C_b given to Charlie
  - Charlie outputs a guess, b'
  - If Pr[b == b'] >> 1/2, Charlie wins the game
  - Advantage definitions same too

Indistinguishability against Known Plaintext Attack (IND-KPA) Example

- Alice and Bob, are employees of Cryptologic Inc. setup (E,D,K); nosy employee Charlie
- Alice encrypts and sends Bob a piece of source code, C_{file} (for a top-secret project they are collaborating on)
- Charlie knows Alice and Bob are developing a joint project
- Charlie also knows that all company source code begins with an English-text preamble (copyright, authors, description, etc.)
- Charlie intercepts the encrypted file in transit
- Charlie now has a plaintext-ciphertext sample (P_{File}, C_{File}) without either Alice or Bob having explicitly given him anything