Roadmap

- ❑ Birth of modern cryptography
- ❑ Computational security
  - ❖ Necessary evils associated with computational security
- ❑ Defining efficient algorithms and negligible probability asymptotically

Perfect Security : The Impractical Goal

- Perfect secrecy is always desirable. But comes with a heavy price
  - Key as long as the message
  - Fresh key for every instance of encryption
- Practical perfectly-secure encryption --- cheating

- Modern cryptography follows a different "approach"
  - Attempt to get "closer" to perfect secrecy
  - Getting rid of the practical limitations imposed by perfect secrecy --- shorter, re-usable key

Birth of Modern Cryptography

Implications :

- ❑ Relatively much shorter key
- ❑ Key can be re-used

Computationally bounded

Learns additional info. about the plain-text with a negligible prob.

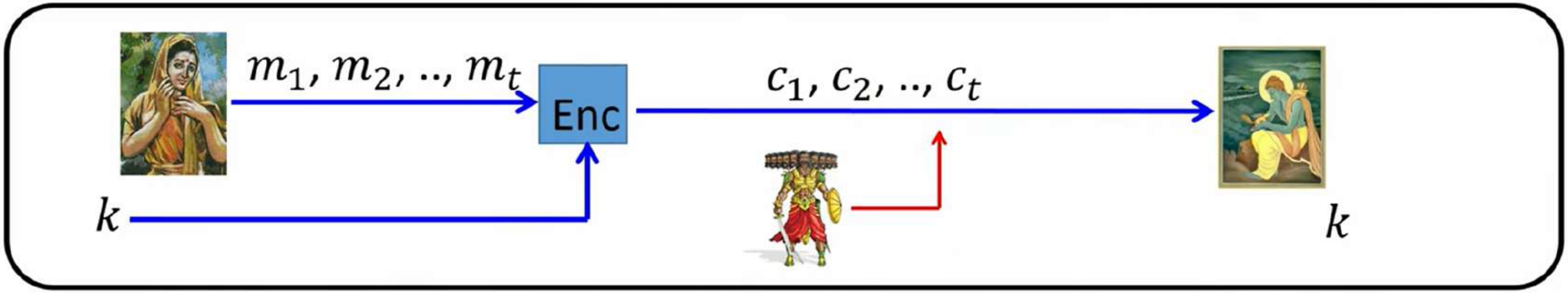
Computational security (Modern Cryptography)

Computational Security : The Basic Idea

- Two relaxations to the model of perfect-security to achieve key reusability
  - Security preserved only against efficient adversaries running in a feasible/practical amount of time
  - Adversaries are allowed to break the scheme with some probability, which is so small that we do not bother
    - Under certain assumptions, the amount of time required to break the scheme will be of order of few lifetimes
    - Acceptable, as most applications do not require ever-lasting security
- The above relaxations are necessary if key reusability is the goal

Relaxation I : Security Only Against Efficient Adversaries

- Consider an encryption scheme where same key is used to encrypt multiple messages

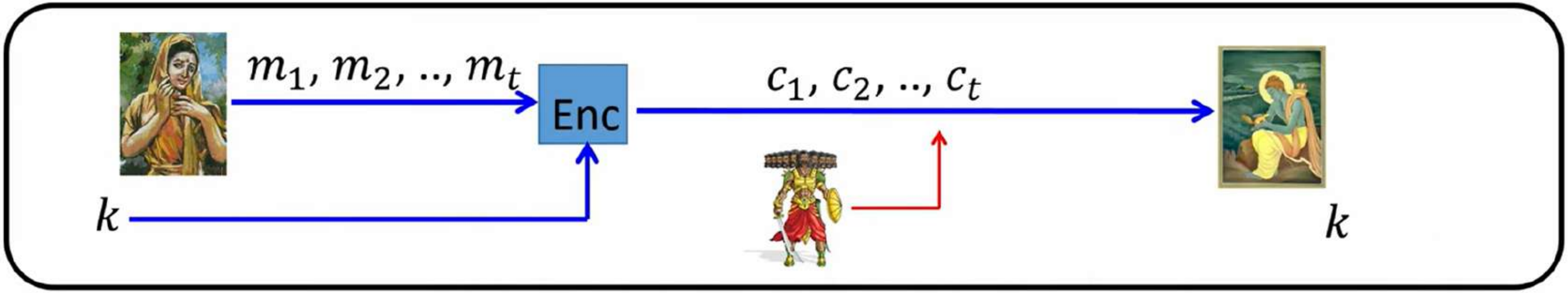


- Consider an adversary, launching a brute-force key-recovery attack in the KPA model
  - Adversary gets access to $(m_1, c_1), \dots, (m_t, c_t)$, where each $c_i \longleftarrow \mathsf{Enc}_k(m_i)$
  - Checks if there is some $\mathrm{k} \in \mathcal{K}$, such that $\mathrm{Dec}_{\mathrm{k}}(c_i) := m_i$, for each (m_i, c_i)
  - Running time: $\mathcal{O}(|\mathcal{K}|)$, success probability: 1

$|\mathcal{K}| = 2^{256}$

Relaxation II : Allowing the Scheme to be Broken with a Small Probability

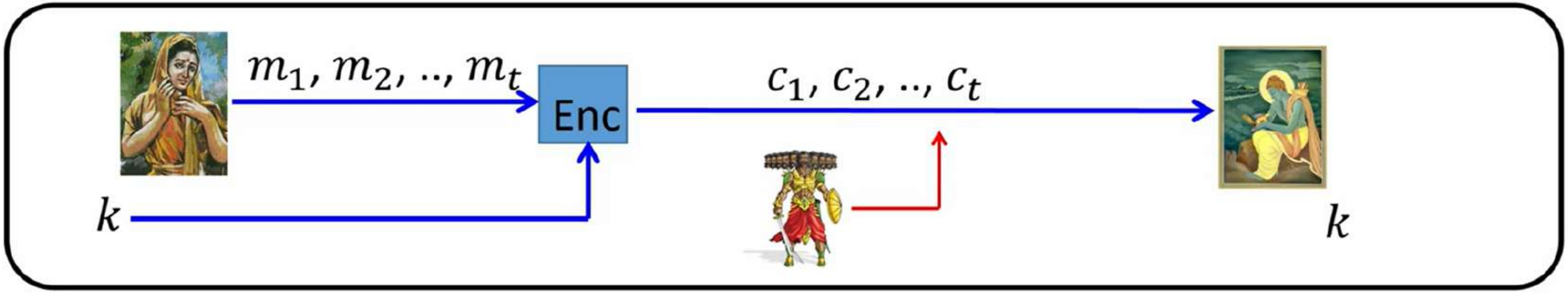
- Consider an encryption scheme where same key is used to encrypt multiple messages



- Consider an adversary, launching a guessing key-recovery attack in the KPA model
  - Adversary gets access to $(m_1, c_1), \ldots, (m_t, c_t)$, where each $c_i \longleftarrow \mathrm{Enc}_k(m_i)$
  - Randomly guess a $\mathrm{k} \in \mathcal{K}$, and check if $\mathrm{Dec}_{\mathrm{k}}(c_i) := m_i$, for each (m_i, c_i)
  - Running time: $\mathcal{O}(|1|)$, success probability: $1 \,/\, |\, \mathcal{K}\, |$

Key-Reusability : Necessary Evils

- Goal: key-reusability



- Unavoidable to prevent two extreme attacks on such a system in the KPA model
  - ~~Brute-force key-recovery attack: Running time: $\mathcal{O}(|\mathcal{K}|)$, success probability: 1~~
  - Guessing key-recovery attack: Running time: $\mathcal{O}(|1|)$, success probability: $1 \,/\, |\,\mathcal{K}\,|$

- Relaxation I : Goal is to achieve security only against efficient adversaries
- Relaxation II : Small probability of a break in the scheme

Key-Reusability : Necessary Evils

- ❑ Relaxation I : Security targeted only against efficient adversaries
- ❑ Relaxation II : Small probability of a break in the scheme

- ❑ How to mathematically define efficient adversaries ?
- ❑ How to mathematically define small (negligible) probability ?

Defining Efficient Algorithms and Negligible Probability Asymptotically

- Security parameter n --- publicly known (part of the scheme)
  - Typically size of secret-key (ex: n = 128, 256, etc)

$$\underbrace{\text{Running time of the users} \qquad \text{Running time of the adversary} \qquad \text{Success probability of the attacker}}_{\text{Functions of the security parameter } n}$$

Defining Efficient Algorithms Asymptotically

- Efficient algorithms --- algorithms with a polynomial running time
  - Algorithm A has a polynomial running time, if there exists a polynomial $p(.)$, such for every input $x \in \{0,1\}^*$, the computation of $A(x)$ terminates within $p(|x|)$ steps, where $|x|$ denotes the length of the string x

- Requirement from any cipher (Gen, Enc, Dec)
  - Gen, Enc and Dec should be efficient algorithms
  - Running time of Gen, Enc and Dec should be a polynomial function of the security parameter n

Defining Negligible Probability Asymptotically

- Negligible functions --- functions which are asymptotically smaller than the inverse of every polynomial function
  - Function $f(n)$ is a negligible function in n, if for every polynomial $p(n)$, there exists some N, such that $f(n) < \frac{1}{p(n)}$, for all $n > N$

$$\approx$$

  - For every constant c, there exists some N, such that $f(n) < n^{-c}$, for all $n > N$

- Example : 2^{-n}, $2^{-\sqrt{n}}$, $n^{-n \log n}$ are all negligible functions

Negligible and Polynomial Functions : Closure Properties

- Let $P_1(n)$ and $P_2(n)$ be two arbitrary polynomial functions. Then
  - $P_1(n) + P_2(n)$, as well as $P_1(n) \times P_2(n)$ are polynomial functions

- Let $\mathsf{negl}_1(n)$ and $\mathsf{negl}_2(n)$ be two arbitrary negligible functions. Then
  - $\mathsf{negl}_1(n) + \mathsf{negl}_2(n)$, as well as $\underline{P(n) \times \mathsf{negl}_1(n)}$ are negligible functions
  - No amplification of a negligible advantage
    - Ex: Prob. that n fair coin−flips turn out to be $(0, \ldots, 0)$: 2^{-n} (negligible)
    - Even if the experiment repeated polynomial number of times, $(0, \ldots, 0)$ will occur in any of these experiments with a negligible prob.

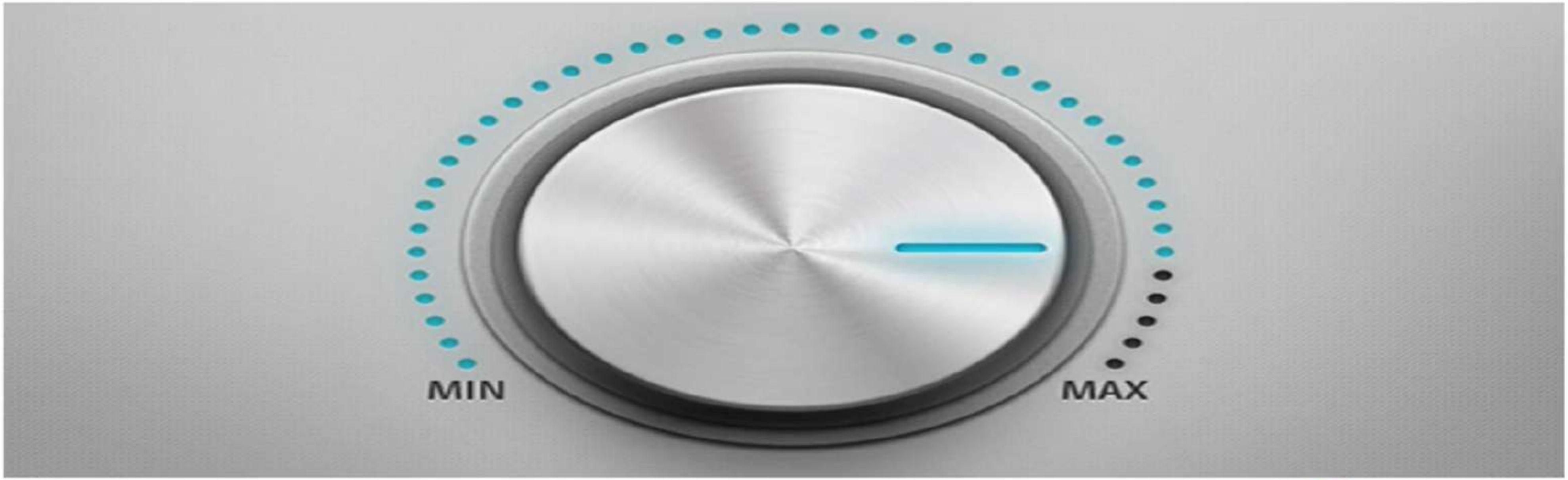
Asymptotic Security in Practice

- Need to carefully select n while deploying a scheme, for meaningful security
  - Consider an encryption scheme, for which an adversary can break the scheme with prob. $2^{40}.2^{-n}$, by doing computations for n^3 minutes
  - the scheme is asymptotically secure, as $2^{40}.2^{-n}$ is negligible

- What value of n should be used while deploying the scheme in practice ?
  - $n = 40 \Rightarrow$ attacker's success probability will be 1, after doing computation for 40^3 minutes (6 weeks)
  - $n = 50 \Rightarrow$ attacker's success probability will be 1/1000, after doing computation for 50^3 minutes (3 months)
  - $n = 500 \Rightarrow$ attacker's success probability will be 2^{-460} , after doing computation for 200 years

Asymptotic Security in Practice

(Slide courtesy : Arpita Patra)



User's running also increases 

Adversary's job becomes harder 

min $\xrightarrow[n]{}$ max