Lecture 10 – Public-key encryption, IND-CPA/CCA, ElGamal, RSA

TEK4500

27.10.2021

Håkon Jacobsen

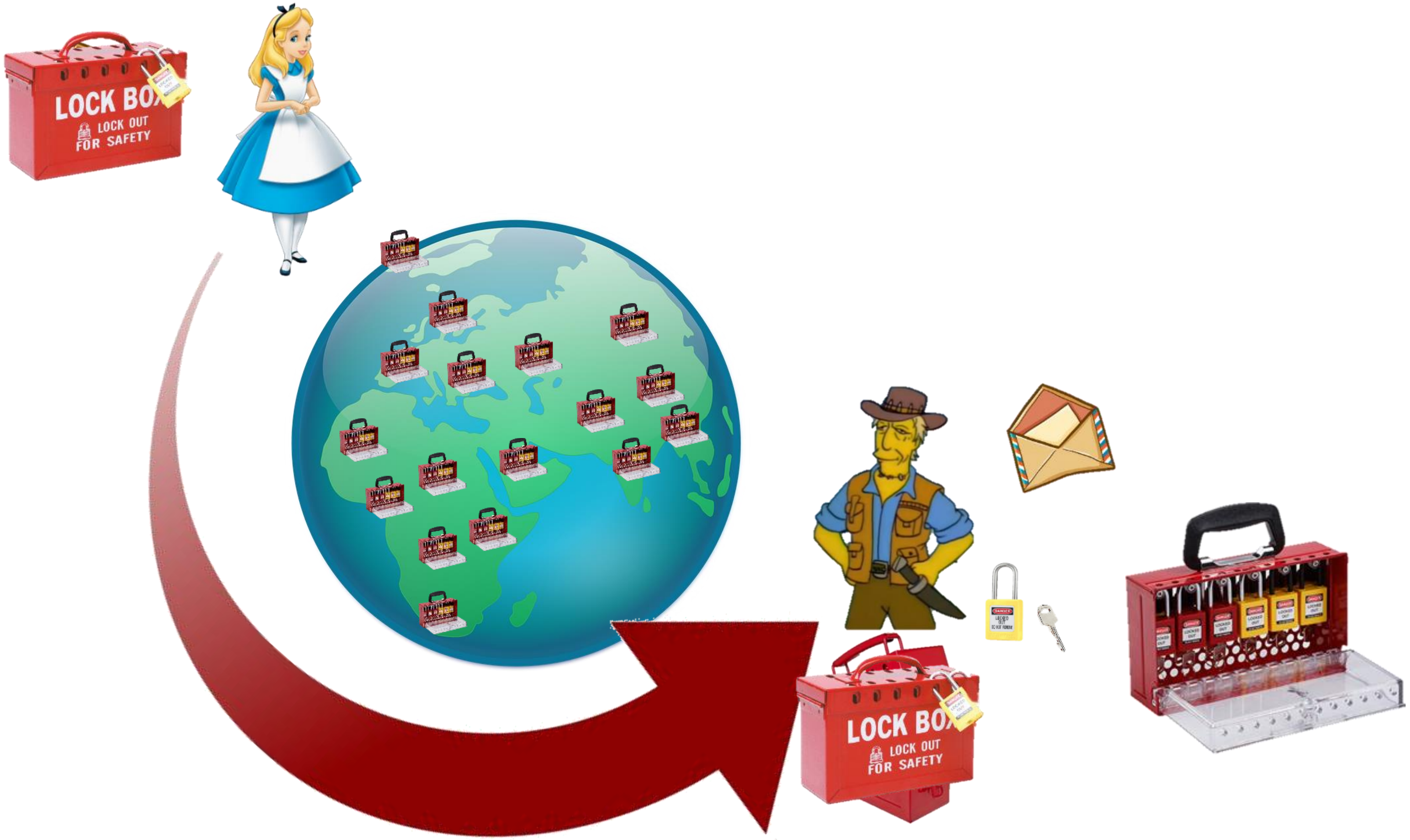
hakon.jacobsen@its.uio.no

Basic goals of cryptography

	Message privacy	**Message integrity / authentication**
Symmetric keys	Symmetric encryption	Message authentication codes (MAC)
Asymmetric keys	Asymmetric encryption (a.k.a. public-key encryption)	Digital signatures

(Key exchange)

Public-key encryption



Public-key encryption – syntax

A **public-key encryption scheme** is a tuple $\Sigma = (\mathrm{KeyGen}, \mathrm{Enc}, \mathrm{Dec})$ of algorithms

$\mathrm{KeyGen} : \{\,\} \to \mathcal{SK} \times \mathcal{PK}$

$\mathrm{Enc} : \mathcal{PK} \times \mathcal{M} \to \mathcal{C}$

$\mathrm{Dec} : \mathcal{SK} \times \mathcal{C} \to \mathcal{M} \cup \{\bot\}$

$(sk, pk) \xleftarrow{\$} \mathrm{KeyGen}$

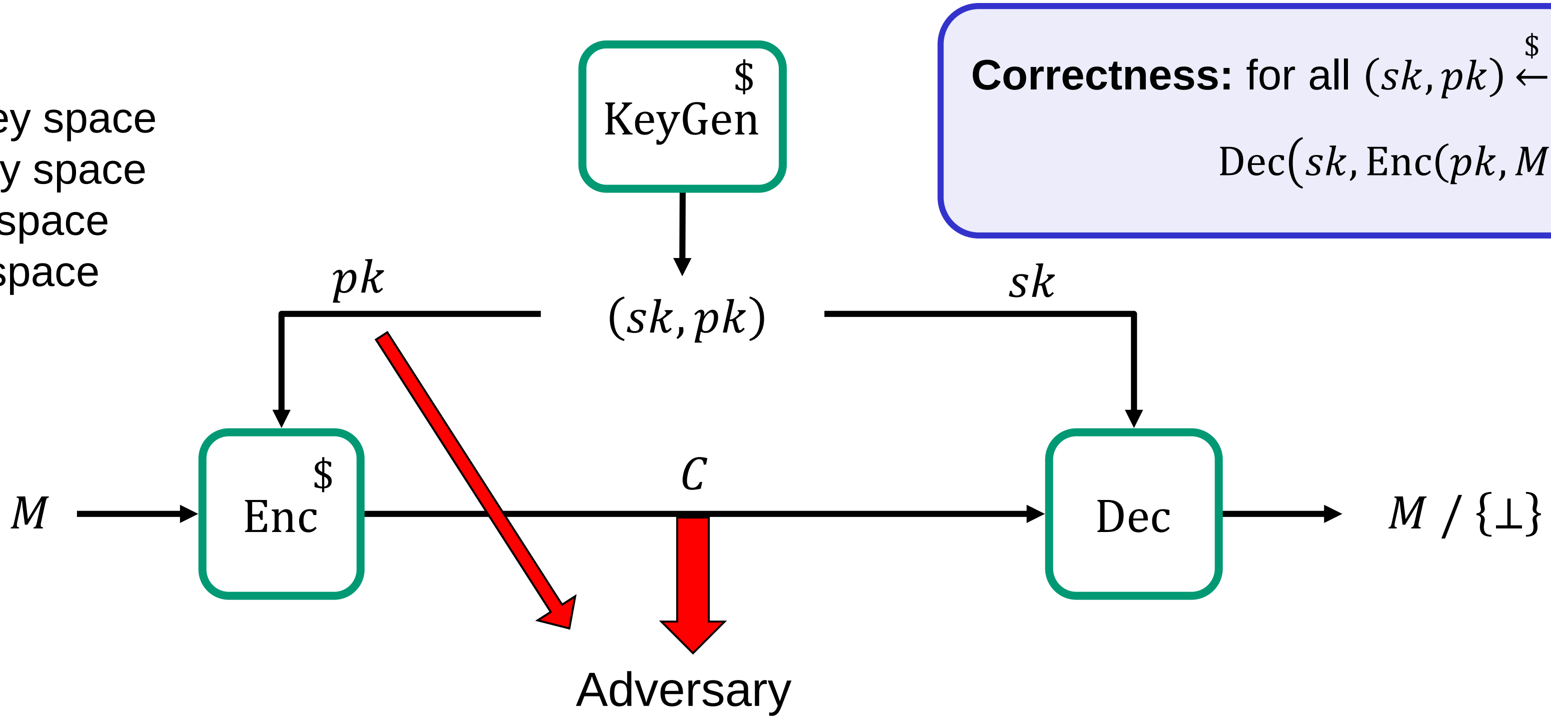
$\mathrm{Enc}(pk, M) = \mathrm{Enc}_{pk}(M) = C$

$\mathrm{Dec}(sk, C) = \mathrm{Dec}_{sk}(C) = M/\bot$

- $\mathcal{SK}$ – private key space
- $\mathcal{PK}$ – public key space
- $\mathcal{M}$ – message space
- $\mathcal{C}$ – ciphertext space

Correctness: for all $(sk, pk) \xleftarrow{\$} \mathrm{KeyGen}$:

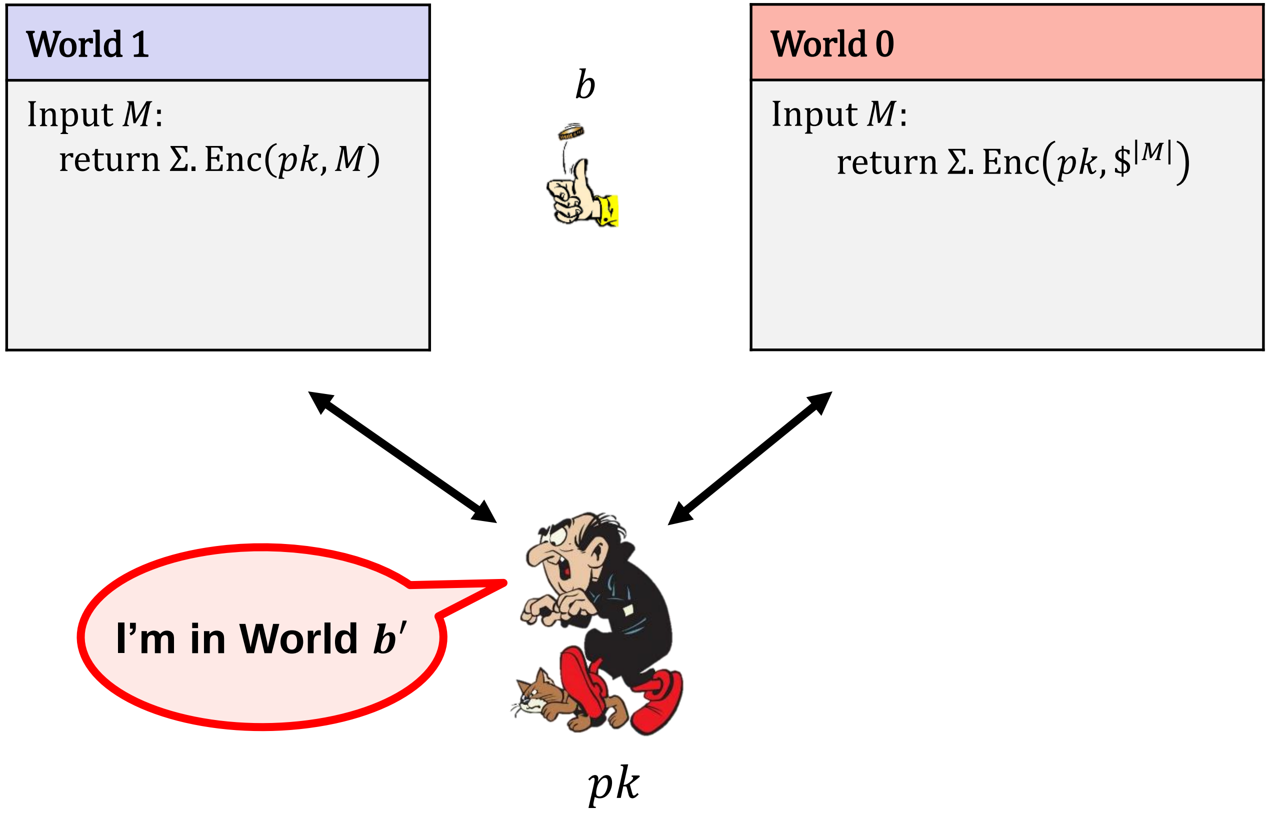
$$\mathrm{Dec}\big(sk, \mathrm{Enc}(pk, M)\big) = M$$



Public-key encryption – security: IND-CPA

```
$\mathbf{Exp}_{\Sigma}^{\text{ind-cpa}}(A)$
    $b \xleftarrow{\$} \{0,1\}$
    $(sk, pk) \xleftarrow{\$} \Sigma.\text{KeyGen}$
    $b' \leftarrow A^{\mathcal{E}(\cdot)}(pk)$
    return $b' \overset{?}{=} b$

$\mathcal{E}(M)$
----------------------------------
    $R \xleftarrow{\$} \{0,1\}^{|M|}$
    $C_0 \leftarrow \Sigma.\text{Enc}(pk, R)$
    $C_1 \leftarrow \Sigma.\text{Enc}(pk\ M)$
    return $C_b$
```



World 1

Input M:
  return $\Sigma.\text{Enc}(pk, M)$

World 0

Input M:
  return $\Sigma.\text{Enc}\big(pk, \$^{|M|}\big)$

Definition: The **IND-CPA advantage** of A is

$$\mathbf{Adv}_{\Sigma}^{\text{ind-cpa}}(A) \overset{\text{def}}{=} |2 \cdot \Pr[b' = b] - 1|$$

Public-key encryption – security: IND-CCA

$\mathbf{Exp}_{\Sigma}^{\text{ind-cca}}(A)$

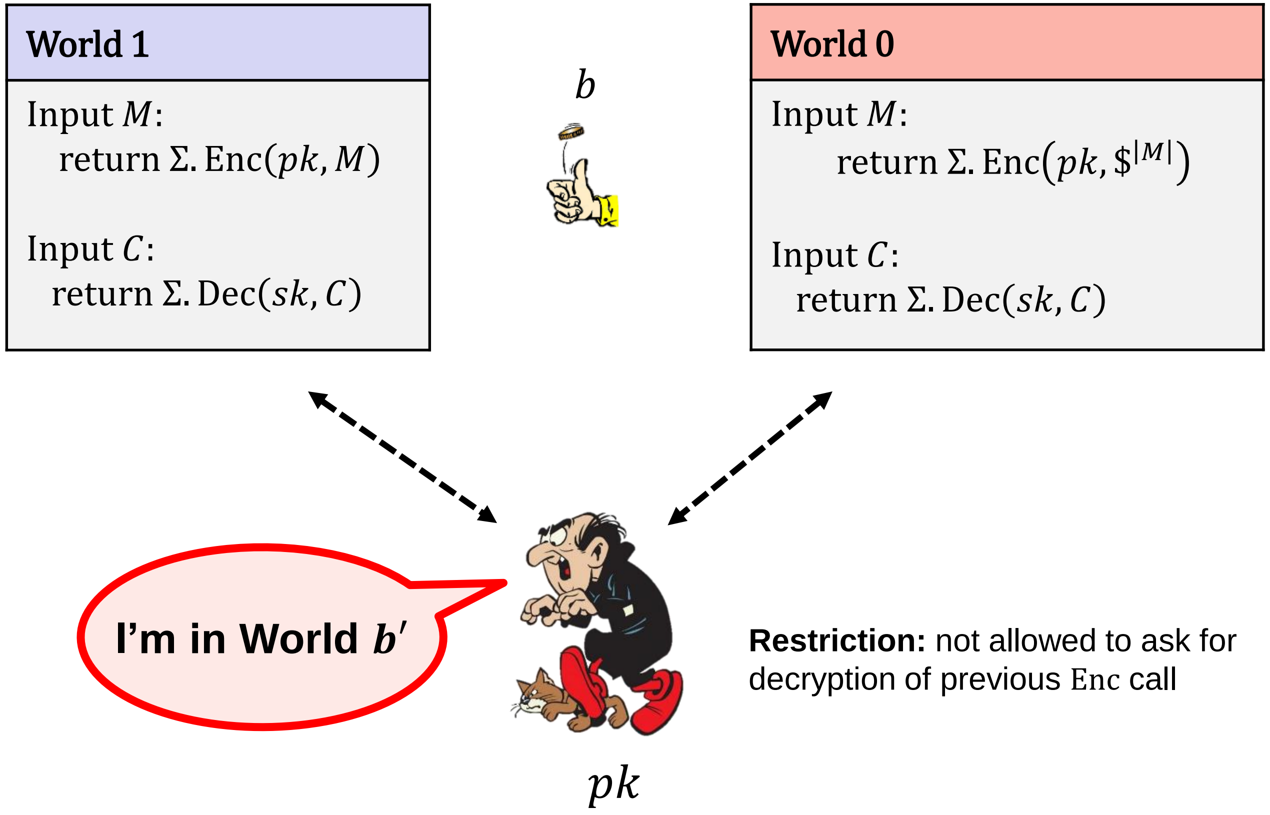
```
$b \overset{\$}{\leftarrow} \{0,1\}$
Ciphertexts $\leftarrow$ [ ]            // bookkeeping
$(sk, pk) \overset{\$}{\leftarrow} \Sigma.\mathrm{KeyGen}$
$b' \leftarrow A^{\mathcal{E}(\cdot),\mathcal{D}(\cdot)}(pk)$
return $b' \overset{?}{=} b$
```

$\mathcal{E}(M)$

```
$R \overset{\$}{\leftarrow} \{0,1\}^{|M|}$
$C_0 \leftarrow \Sigma.\mathrm{Enc}(pk, R)$
$C_1 \leftarrow \Sigma.\mathrm{Enc}(pk\ M)$
Ciphertexts.add($C_b$)
return $C_b$
```

$\mathcal{D}(C)$

```
if $C \in$ Ciphertexts:
    return $\perp$
return $\Sigma.\mathrm{Dec}(sk, C)$
```



World 1

Input M:
return $\Sigma.\mathrm{Enc}(pk, M)$

Input C:
return $\Sigma.\mathrm{Dec}(sk, C)$

World 0

Input M:
return $\Sigma.\mathrm{Enc}\big(pk, \$^{|M|}\big)$

Input C:
return $\Sigma.\mathrm{Dec}(sk, C)$

Restriction: not allowed to ask for decryption of previous Enc call

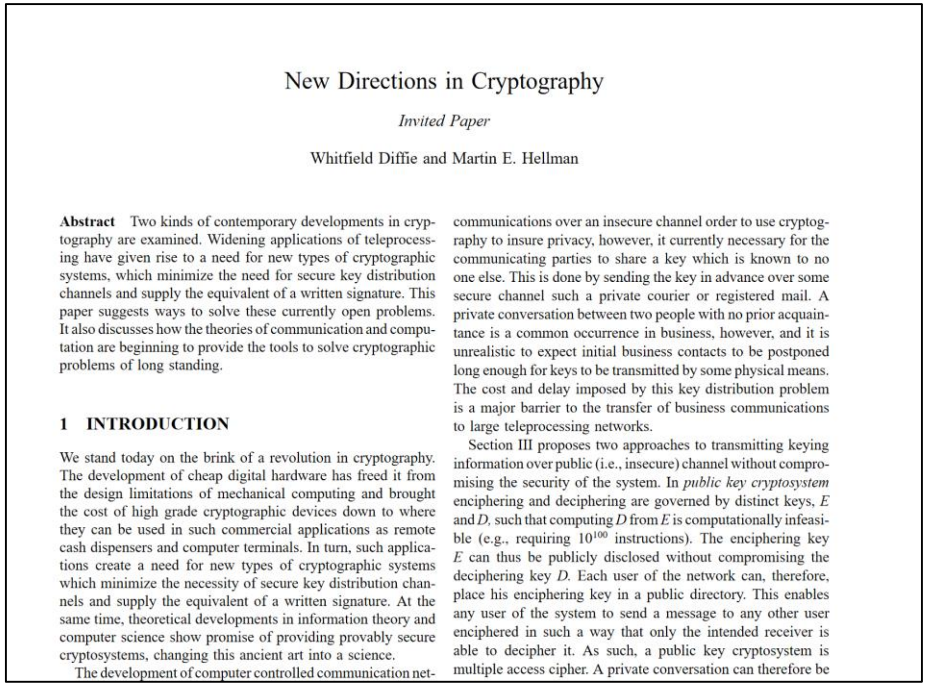
Definition: The **IND-CCA advantage** of A is

$$\mathbf{Adv}_{\Sigma}^{\text{ind-cca}}(A) \overset{\text{def}}{=} |2 \cdot \Pr[b' = b] - 1|$$

Diffie-Hellman key exchange

- Discovered in the 1970's
- Allows two parties to establish a shared secret without ever having met
- Diffie & Hellman paper also introduced the idea of:
  - Public-key encryption
    - But didn't figure out how to do it
    - 1984: ElGamal encryption scheme
  - Digital signatures
    - But didn't figure out how to do it

Ralph Merkle Whitfield Diffie
Martin Hellman



New Directions in Cryptography

Invited Paper

Whitfield Diffie and Martin E. Hellman

Abstract Two kinds of contemporary developments in cryptography are examined. Widening applications of teleprocessing have given rise to a need for new types of cryptographic systems, which minimize the need for secure key distribution channels and supply the equivalent of a written signature. This paper suggests ways to solve these currently open problems. It also discusses how the theories of communication and computation are beginning to provide the tools to solve cryptographic problems of long standing.

1 INTRODUCTION

We stand today on the brink of a revolution in cryptography. The development of cheap digital hardware has freed it from the design limitations of mechanical computing and brought the cost of high grade cryptographic devices down to where they can be used in such commercial applications as remote cash dispensers and computer terminals. In turn, such applications create a need for new types of cryptographic systems which minimize the necessity of secure key distribution channels and supply the equivalent of a written signature. At the same time, theoretical developments in information theory and computer science show promise of providing provably secure cryptosystems, changing this ancient art into a science.

The development of computer controlled communication net-

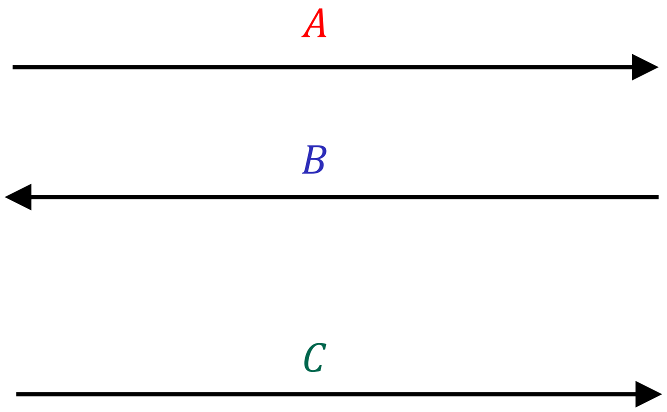
communications over an insecure channel order to use cryptography to insure privacy, however, it currently necessary for the communicating parties to share a key which is known to no one else. This is done by sending the key in advance over some secure channel such a private courier or registered mail. A private conversation between two people with no prior acquaintance is a common occurrence in business, however, and it is unrealistic to expect initial business contacts to be postponed long enough for keys to be transmitted by some physical means. The cost and delay imposed by this key distribution problem is a major barrier to the transfer of business communications to large teleprocessing networks.

Section III proposes two approaches to transmitting keying information over public (i.e., insecure) channel without compromising the security of the system. In *public key cryptosystem* enciphering and deciphering are governed by distinct keys, E and D, such that computing D from E is computationally infeasible (e.g., requiring 10^{100} instructions). The enciphering key E can thus be publicly disclosed without compromising the deciphering key D. Each user of the network can, therefore, place his enciphering key in a public directory. This enables any user of the system to send a message to any other user enciphered in such a way that only the intended receiver is able to decipher it. As such, a public key cryptosystem is multiple access cipher. A private conversation can therefore be

Public-key encryption: ElGamal

$G = \langle g \rangle$

M



A

B

C

$a \xleftarrow{\$} \{1, \dots, |G|\}$

$A \leftarrow g^{a}$

$Z \leftarrow B^{a}$

$C \leftarrow \Sigma.\mathrm{Enc}(Z, M)$

Symmetric encryption scheme

$b \xleftarrow{\$} \{1, \dots, |G|\}$

$B \leftarrow g^{b}$

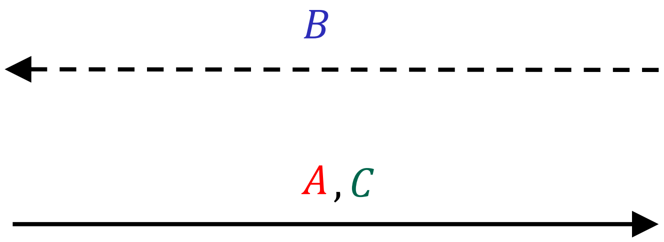
$Z \leftarrow A^{b}$

$M \leftarrow \Sigma.\mathrm{Dec}(Z, C)$

Public-key encryption: ElGamal

$G = \langle g \rangle$

M



KeyGen

```
$sk = b \overset{\$}{\leftarrow} \{1, \dots, |G|\}$
$pk = B \leftarrow g^b$
return $(sk, pk)$
```

Enc(pk, M)

```
$a \overset{\$}{\leftarrow} \{1, \dots, |G|\}$
$A \leftarrow g^a$
$Z \leftarrow B^a$
$C \leftarrow \Sigma.\mathrm{Enc}(Z, M)$
return $(A, C)$
```

$\Sigma.\mathrm{Enc} : G \times \mathcal{M} \to \mathcal{C}$

Actually want $\Sigma.\mathrm{Enc} : \{0,1\}^k \times \mathcal{M} \to \mathcal{C}$

ElGamal IND-CPA security:

- DLOG + DH must be hard
- Σ must be IND-CPA secure
- Is this enough?

Dec(sk, C)

```
$Z \leftarrow A^b$
$M \leftarrow \Sigma.\mathrm{Dec}(Z, C)$
return $M$
```

- **No:** Σ only guarantees security if the key Z is *independent* and *uniformly* distributed in the group G

  …but $Z = g^{ab}$ *isn't* uniformly distributed in G*!*

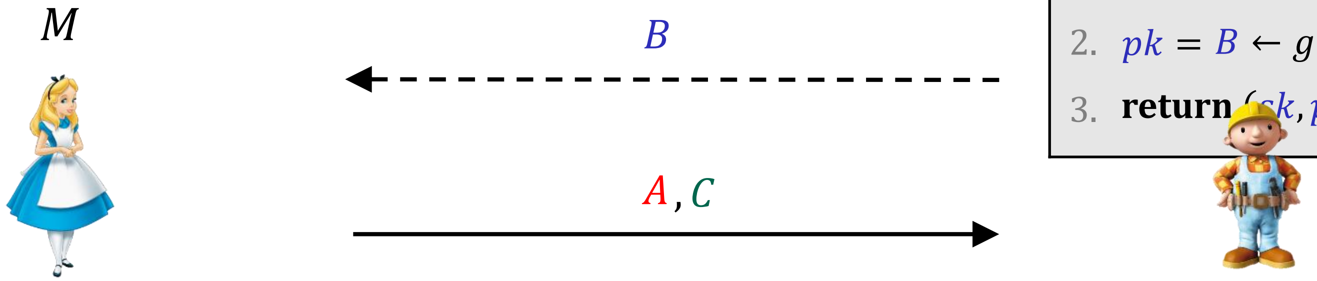
Public-key encryption: Hashed-ElGamal

$H : G \to \{0,1\}^k$

$G = \langle g \rangle$

KeyGen

```
$sk = b \overset{\$}{\leftarrow} \{1, \dots, |G|\}$
$pk = B \leftarrow g^b$
return $(sk, pk)$
```



Enc(pk, M)

```
$a \overset{\$}{\leftarrow} \{1, \dots, |G|\}$
$A \leftarrow g^a$
$Z \leftarrow H(B^a)$
$C \leftarrow \Sigma.\mathrm{Enc}(Z, M)$
return $(A, C)$
```

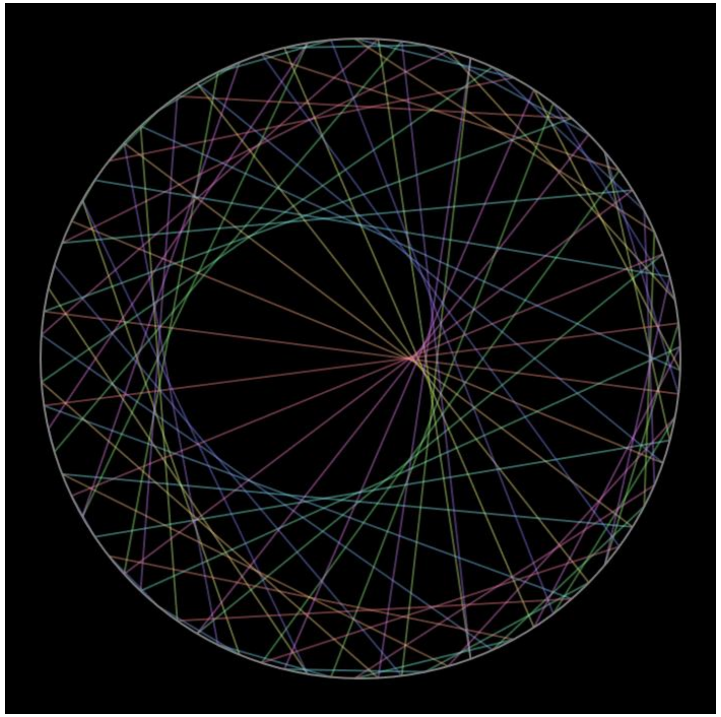
Dec(sk, C)

```
$Z \leftarrow H\left(A^b\right)$
$M \leftarrow \Sigma.\mathrm{Dec}(Z, C)$
```

Theorem: Hashed-ElGamal is IND-CPA secure if:

1. DH-problem is hard in G
2. Σ is IND-CPA secure
3. H is “perfect” (i.e., a **random oracle**)

$$C = M^e \pmod{n}$$

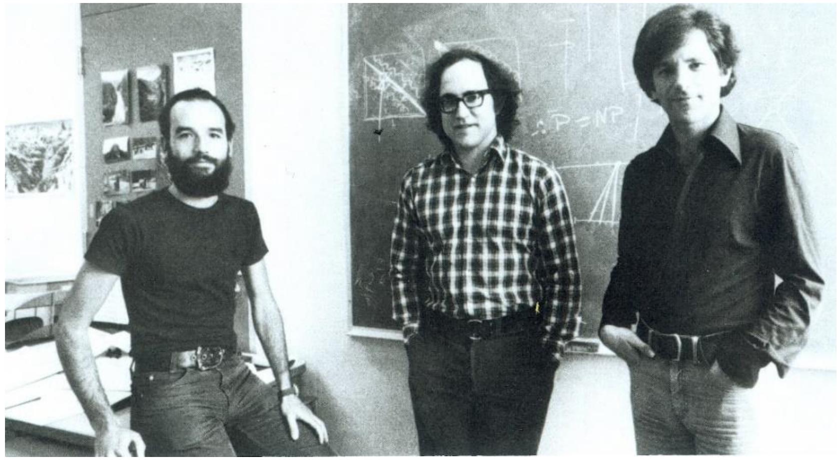


The RSA encryption scheme

RSA

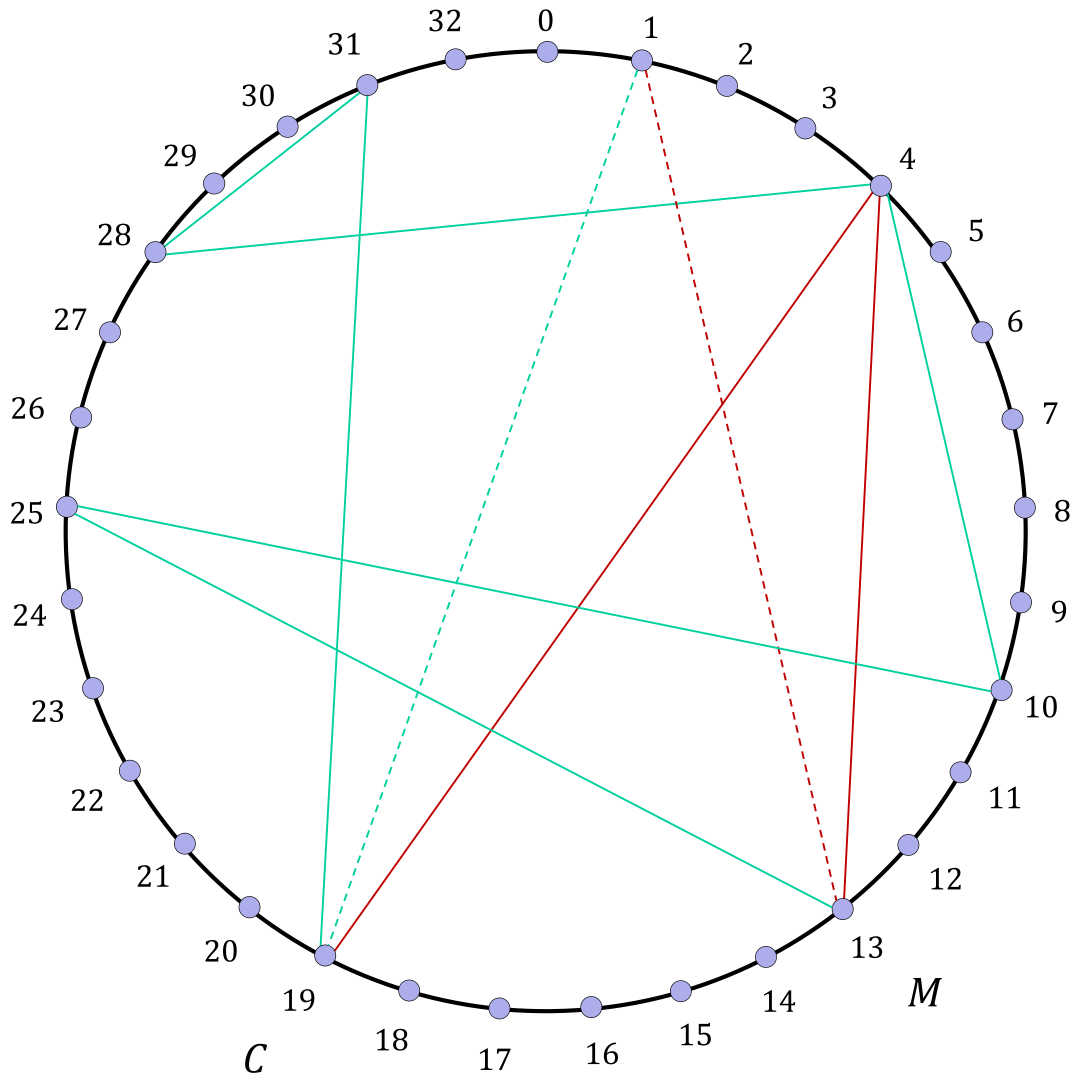
- Designed by Rivest, Shamir and Adleman in 1977
  - One year before ElGamal

- Used both for public-key *encryption* and *digital signatures*



Adi Shamir Ron Rivest Leonard Adleman

RSA encryption



$e = 3$

$d = 7$

The group $(\boldsymbol{Z}_n^*,\cdot)$

$\boldsymbol{Z}_p = \{0,1,\dots,p-1\}$ $(\boldsymbol{Z}_p,\cdot)$ is *not* a group!

$\boldsymbol{Z}_p^* = \{1,\dots,p-1\}$ $(\boldsymbol{Z}_p^*,\cdot)$ *is* a group!

$\boldsymbol{Z}_n = \{0,1,\dots,n-1\}$ $(\boldsymbol{Z}_n,\cdot)$ is *not* a group!

$\boldsymbol{Z}_n^* \neq \underbrace{\{1,\dots,n-1\}}_{\boldsymbol{Z}_n^+}$ $(\boldsymbol{Z}_n^+,\cdot)$ is *also not* a group!

$$\boldsymbol{Z}_n^* = \underbrace{\text{invertible elements in } \boldsymbol{Z}_n}_{(\boldsymbol{Z}_n^*,\cdot)\ \textit{is}\text{ a group!}} \overset{\textbf{theorem}}{=} \{\, a \in \boldsymbol{Z}_n \mid \gcd(a,n) = 1 \,\}$$

Not invertible	Invertible
$2,4,5,6,8$	$1,3,7,9$

$\boldsymbol{Z}_{10}^+ = \{1,2,3,4,5,6,7,8,9\}$

$2\cdot 1 = 2 \mod 10$

$2\cdot 2 = 4 \mod 10$

$2\cdot 3 = 6 \mod 10$

$2\cdot 4 = 8 \mod 10$

$2\cdot 5 = 0 \mod 10$

$2\cdot 6 = 2 \mod 10$

$2\cdot 7 = 4 \mod 10$

$2\cdot 8 = 6 \mod 10$

$2\cdot 9 = 8 \mod 10$

$1\cdot 1 = 1 \mod 10$

$3\cdot 7 = 21 = 1 \mod 10$

$9\cdot 9 = 81 = 1 \mod 10$

$10 = 2\cdot 5$

$2 = 2$

$4 = 2\cdot 2$

$5 = 5$

$6 = 2\cdot 3$

$8 = 2\cdot 2\cdot 2$

Proof: let $d = \gcd(a,n)$

- a invertible $\Longrightarrow \exists b \in \boldsymbol{Z}_n$ such that $ab = 1 \bmod n \Longrightarrow \exists k\colon ab = 1 + kn \Longrightarrow ab - kn = 1 \Longrightarrow d(a'b - kn') = 1 \Longrightarrow d = 1$
- $d = 1 \Longrightarrow$ Claim: $\exists s,t \in \boldsymbol{Z}$ such that $sa + tn = d = 1 \Longrightarrow sa = 1 - tn \Longrightarrow sa = 1 \bmod n \Longrightarrow a$ is invertible

Euler's $\phi(n)$ function

- $\phi(n) \overset{\text{def}}{=} |\boldsymbol{Z}_n^*| \;=\; |\{\, a \in \boldsymbol{Z}_n \mid \gcd(a,n) = 1 \,\}|$

- $\phi(p) = p - 1$

- $\phi(p \cdot q) = \;(p-1)\cdot(q-1)$

$$\overbrace{1\cdot p \quad 2\cdot p \quad 3\cdot p \quad \cdots \quad q\cdot p}^{q}$$

$$\underbrace{1\cdot q \quad 2\cdot q \quad 3\cdot q \quad \cdots \quad p\cdot q}_{p}$$

$$\begin{aligned}\phi(pq) &= pq \;-\; \#\text{numbers with } \gcd(x, pq) \neq 1 \\ &= pq \;-\; q \;-\; p \;+\; 1 \\ &= (p-1)\cdot(q-1)\end{aligned}$$

- **Note:** $\phi(\overset{p\cdot q}{n}) \approx n - 2\sqrt{n} \approx n$
  - *almost all* integers are invertible for large p, q

n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
$\phi(n)$	1	**1**	**2**	2	**4**	2	**6**	4	6	4	**10**	4	**12**	6	8	8	**16**	6	**18**	8

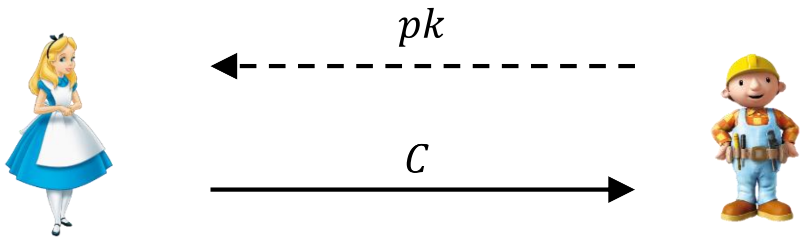
Textbook RSA

$$\mathrm{RSA.Enc}: \quad \overbrace{\boldsymbol{Z}^+ \times \boldsymbol{Z}^*_{\phi(n)}}^{\mathcal{PK}} \times \overset{\mathcal{M}}{\boldsymbol{Z}^*_n} \to \overset{\mathcal{C}}{\boldsymbol{Z}^*_n}$$

$$\mathrm{RSA.Dec}: \quad \underbrace{\boldsymbol{Z}^+ \times \boldsymbol{Z}^*_{\phi(n)}}_{\mathcal{SK}} \times \underset{\mathcal{C}}{\boldsymbol{Z}^*_n} \to \underset{\mathcal{M}}{\boldsymbol{Z}^*_n}$$

```
KeyGen
    $p, q \xleftarrow{\$}$ two random prime numbers
    $n \leftarrow p \cdot q$
    $\phi(n) = (p-1)(q-1)$
    **choose** $e$ such that $\gcd(e, \phi(n)) = 1$
    $d \leftarrow e^{-1} \bmod \phi(n)$
    $sk \leftarrow (n, d)$      $pk \leftarrow (n, e)$
    **return** $(sk, pk)$
```

```
Enc($pk = (n, e), M \in \boldsymbol{Z}^*_n$)
    $C \leftarrow M^e \bmod n$
    **return** $C$
```



```
Dec($sk = (n, d), C \in \boldsymbol{Z}^*_n$)
    $M \leftarrow C^d \bmod n$
    **return** $M$
```

Common choices of e:	3	17	65 537
	11_2	10001_2	$1\ 0000\ 0000\ 0000\ 0001_2$

RSA example

$\boldsymbol{Z}_{33}^{*}$

$n = 33 = 3 \cdot 11$

$e = 3$ // need to find $d = e^{-1} \bmod \phi(n)$

$\phi(n) = (3-1)(11-1) = 20$

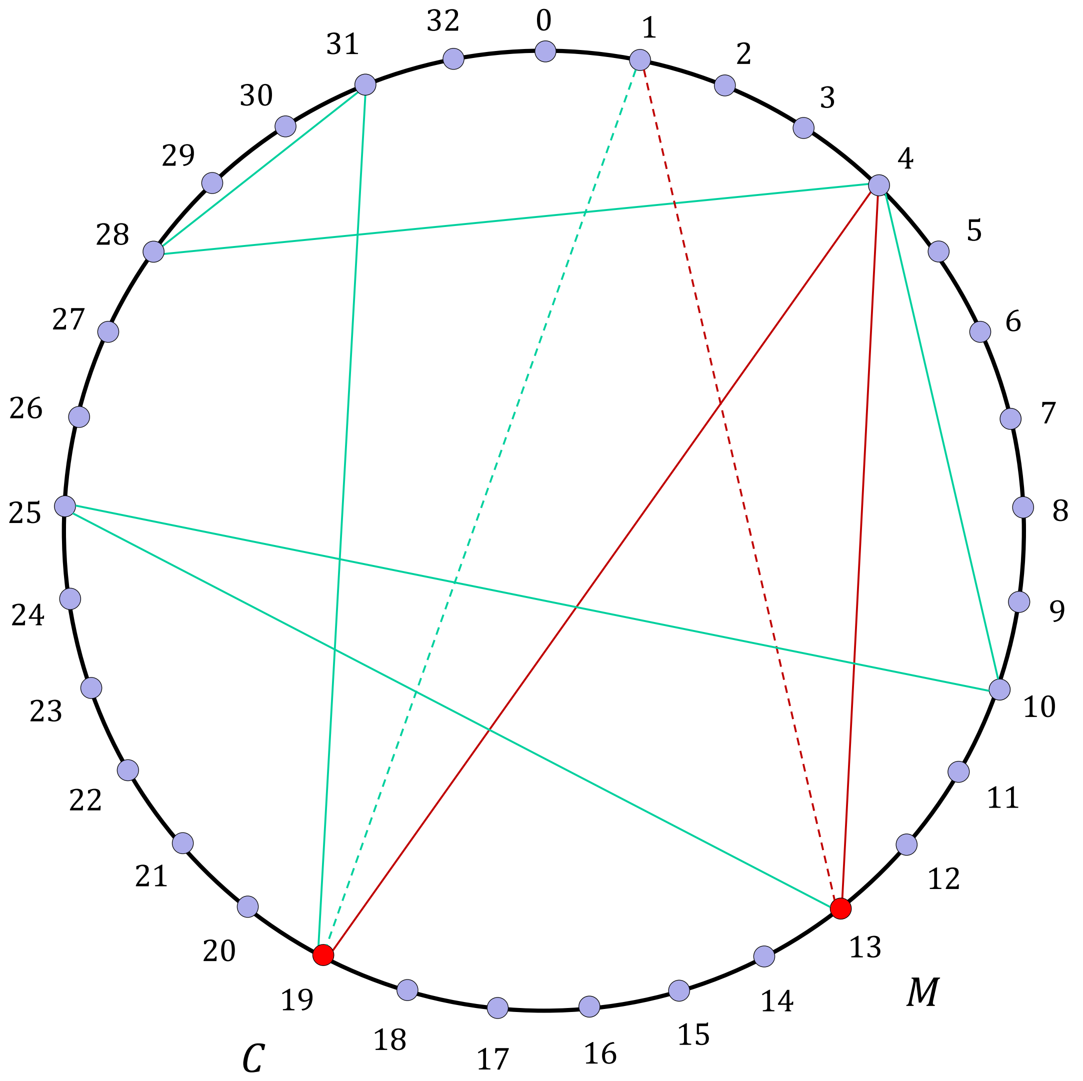
$3 \cdot 7 = 21 = 1 \bmod 20$

$d = 7$

$M = 13$

$C \leftarrow 13^3 = 2197 = 19 \bmod 33$

$C^7 = 19^7 = 893871739 = 13 \bmod 33$



Euler’s Theorem

Theorem: if $(G,\circ)$ is a finite group, then for all $g \in G$:

$$g^{|G|} = e$$

- $(\boldsymbol{Z}_p^*,\cdot)$: $\left|\boldsymbol{Z}_p^*\right| = p - 1$

Fermat’s theorem: if p is prime, then for all $a \neq 0 \pmod{p}$:

$$a^{p-1} \equiv 1 \pmod{p}$$

- $(\boldsymbol{Z}_n^*,\cdot)$: $|\boldsymbol{Z}_n^*| = \phi(n)$

Euler’s theorem: for all positive integers n, if $\gcd(a,n) = 1$ then

$$a^{\phi(n)} \equiv 1 \pmod{n}$$

Textbook RSA – correctness

$$\mathrm{RSA.Dec}\big(sk, \mathrm{RSA.Enc}(pk, M)\big) = M$$

Euler's theorem: for all $a \in \boldsymbol{Z}_n^*$

$$a^{\phi(n)} \equiv 1 \pmod{n}$$

$$C^d = M^{ed} = M^{1+\phi(n)k} = M^1 \cdot M^{\phi(n)k} = M \bmod n$$

$$d = e^{-1} \bmod \phi(n) \iff ed = 1 \bmod \phi(n) \iff ed = 1 + \phi(n)k$$

Fact: RSA also works for $M \in \boldsymbol{Z}_n$

```
KeyGen
    $p, q \overset{\$}{\leftarrow}$ two random prime numbers
    $n \leftarrow p \cdot q$
    $\phi(n) = (p-1)(q-1)$
    choose $e$ such that $\gcd\big(e, \phi(n)\big) = 1$
    $d \leftarrow e^{-1} \bmod \phi(n)$
    $sk \leftarrow (n, d)$        $pk \leftarrow (n, e)$
    return $(sk, pk)$
```

```
Enc($pk = (n, e), M \in \boldsymbol{Z}_n^*$)
    $C \leftarrow M^e \bmod n$
    return $C$
```

```
Dec($sk = (n, d), C \in \boldsymbol{Z}_n^*$)
    $M \leftarrow C^d \bmod n$
    return $M$
```

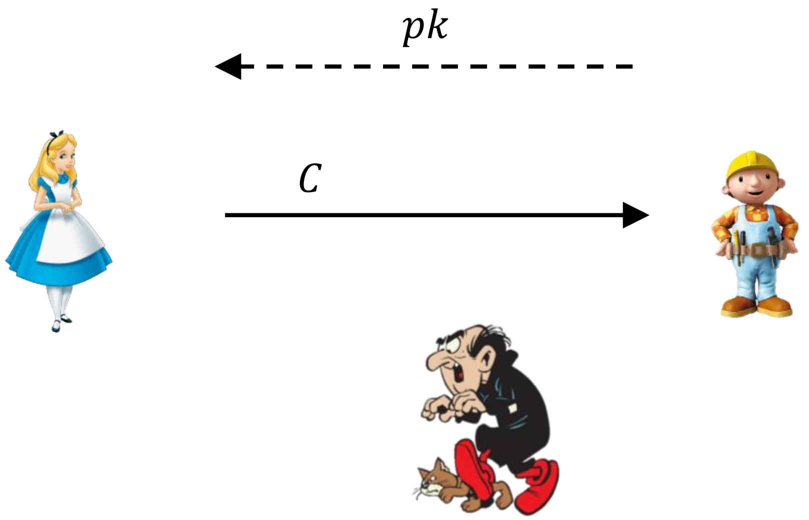
Textbook RSA – security

$$\mathrm{RSA.Enc}: \quad \boldsymbol{Z}^{+} \times \boldsymbol{Z}^{*}_{\phi(n)} \times \boldsymbol{Z}^{*}_{n} \to \boldsymbol{Z}^{*}_{n}$$

$$\mathrm{RSA.Enc}: \quad n, e, M \mapsto M^{e} \bmod n$$

Enc$(pk = (n, e), M \in \boldsymbol{Z}^{*}_{n})$

```
$C \leftarrow M^{e} \bmod n$
return $C$
```



KeyGen

```
$p, q \xleftarrow{\$}$ two random prime numbers
$n \leftarrow p \cdot q$
$\phi(n) = (p-1)(q-1)$
choose $e$ such that $\gcd(e, \phi(n)) = 1$
$d \leftarrow e^{-1} \bmod \phi(n)$
$sk \leftarrow (n, d)$      $pk \leftarrow (n, e)$
return $(sk, pk)$
```

Dec$(sk = (n, d), C \in \boldsymbol{Z}^{*}_{n})$

```
$M \leftarrow C^{d} \bmod n$
return $M$
```

Textbook RSA – security

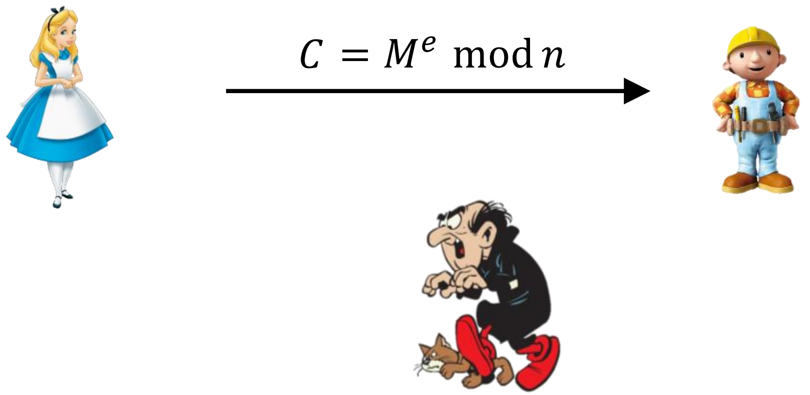
$\mathrm{RSA.Enc}_{pk} : \ \boldsymbol{Z}_n^* \to \boldsymbol{Z}_n^*$

$\mathrm{RSA.Enc}_{pk} : \ M \mapsto M^e \bmod n$

$pk = (\overset{pq}{n}, e)$

$sk = d = e^{-1} \bmod \underset{(p-1)(q-1)}{\phi(n)}$

```
Enc(pk = (n, e), M ∈ Z_n^*)
1.    $C \leftarrow M^e \bmod n$
2.    return $C$
```



$C = M^e \bmod n$

Security:

- Must be hard to compute M given pk, $M^e \bmod n$ (RSA assumption)
- Must be hard to find d given n, e
- Must be hard to find $\phi(n)$ given n, e
- Must be hard to find p, q given n (Factoring assumption)

equivalent!

How hard is factoring?

- In practice factoring is only known way to break RSA

- Naïve $\mathrm{Factor}(n)$:
  - 3 divides n? return 3
  - 5 divides n? return 5
  - 7 divides n? return 7

    $\vdots$
  - $\lfloor\sqrt{n}\rfloor$ divides n? return $\lfloor\sqrt{n}\rfloor$
  - return n

Very inefficient: $n \approx 2^k \implies \pi(n) \approx \frac{2^k}{\ln 2^k} \approx \frac{2^k}{k}$

Time to factor n for $p \approx q \approx 2^{k/2}$

$$\approx \mathcal{O}\left(e^{1.93k^{1/3}(\log k)^{2/3}+c}\right)$$

- *Much* faster algorithms known
  - Quadratic sieve
  - Rational sieve
  - General number field sieve

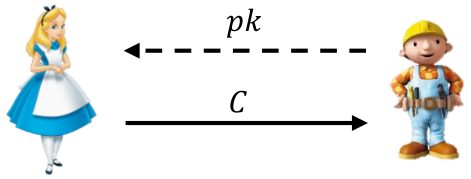
  - Current record: 829 bits

Textbook RSA – security

- Textbook RSA is **not** IND-CPA secure!
  - Deterministic
  - Malleable
  - Many other attacks as well*

- Textbook RSA is *not* an encryption scheme!

```
Enc($pk = (n, e), M \in \mathbb{Z}_n^*$)
    $C \leftarrow M^e \bmod n$
    return $C$
```



```
KeyGen
    $p, q \xleftarrow{\$}$ two random prime numbers
    $n \leftarrow p \cdot q$
    $\phi(n) = (p - 1)(q - 1)$
    choose $e$ such that $\gcd(e, \phi(n)) = 1$
    $d \leftarrow e^{-1} \bmod \phi(n)$
    $sk \leftarrow (n, d)$      $pk \leftarrow (n, e)$
    return $(sk, pk)$
```

```
Dec($sk = (n, d), C \in \mathbb{Z}_n^*$)
    $M \leftarrow C^d \bmod n$
    return $M$
```

- So what is it? Answer: a *one-way trapdoor permutation*

* https://crypto.stackexchange.com/questions/20085/which-attacks-are-possible-against-raw-textbook-rsa

Shoup's RSA variant

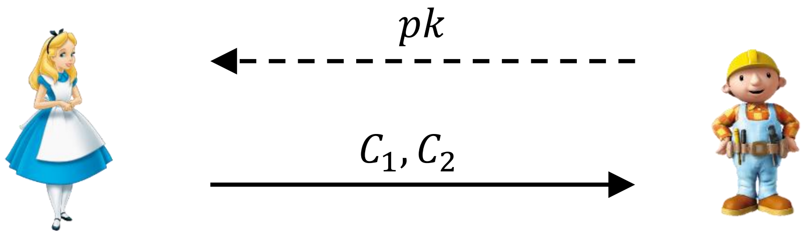
Enc$(pk = (n, e), M)$

```
$K \leftarrow \mathbb{Z}_n^*$
$C_1 \leftarrow K^e \bmod n$
$C_2 \leftarrow \Sigma.\mathrm{Enc}(K, M)$
return $C_1, C_2$
```

Symmetric encryption scheme

$$\Sigma.\mathrm{Enc} : \mathbb{Z}_n^* \times \mathcal{M} \rightarrow \mathcal{C}$$

Actually want: $\Sigma.\mathrm{Enc} : \{0,1\}^k \times \mathcal{M} \rightarrow \mathcal{C}$



$$H : \mathbb{Z}_n^* \rightarrow \{0,1\}^k$$

KeyGen

```
$(sk, pk) \leftarrow \mathrm{RSA.KeyGen}$
return $(sk, pk)$
```

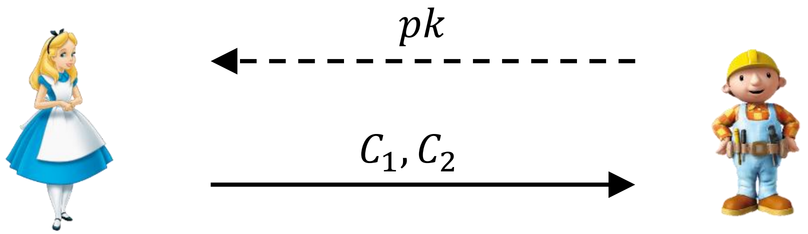
Dec$(sk = d, C = (C_1, C_2))$

```
$K \leftarrow C_1^d \bmod n$
$M \leftarrow \Sigma.\mathrm{Dec}(K, C_2)$
return $M$
```

Shoup's RSA variant

Enc$(pk = (n, e), M)$

```
$R \leftarrow \boldsymbol{Z}_n^*$
$C_1 \leftarrow R^e \bmod n$
$K \leftarrow H(R)$
$C_2 \leftarrow \Sigma.\mathrm{Enc}(K, M)$
**return** $C_1, C_2$
```



KeyGen

```
$(sk, pk) \leftarrow \mathrm{RSA.KeyGen}$
**return** $(sk, pk)$
```

Dec$(sk = d, C = (C_1, C_2))$

```
$R \leftarrow C_1^d \bmod n$
$K \leftarrow H(R)$
$M \leftarrow \Sigma.\mathrm{Dec}(K, C_2)$
**return** $M$
```

$$H : \boldsymbol{Z}_n^* \rightarrow \{0,1\}^k$$

Theorem: Shoup-RSA is IND-CPA (IND-CCA) secure if:

1. RSA-problem is hard in $\boldsymbol{Z}_n^*$
2. Σ is IND-CPA (IND-CCA) secure
3. H is “perfect” (i.e., a random oracle)

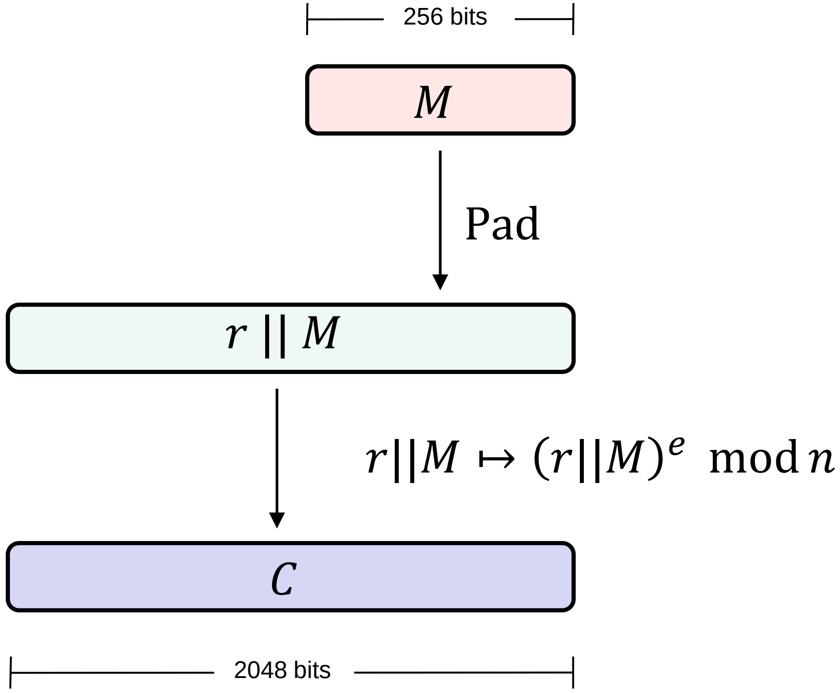
RSA in practice I

- Textbook RSA is deterministic ⟹ cannot be IND-CPA/IND-CCA secure

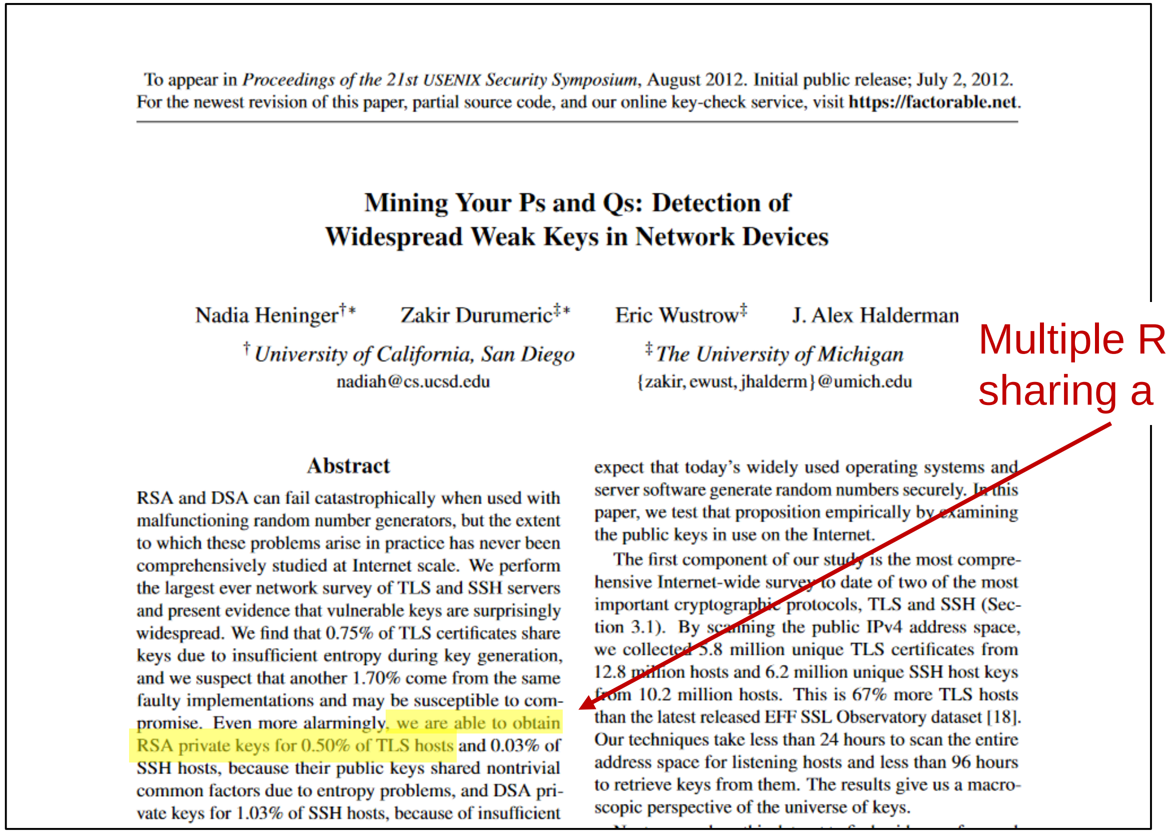
- How to achieve IND-CPA/IND-CCA?
  - Pad message with random data before applying RSA function
  - PKCS#1v1.5 (no existing security proof; many impl. attacks)
  - RSA-OAEP (complicated security proof; first proof buggy)

- RSA *encryption* not used much in practice anymore
  - Mostly **key transport** of (short) symmetric key

- RSA *digital signatures* still very common
  - Covered next week



RSA in practice II



To appear in *Proceedings of the 21st USENIX Security Symposium*, August 2012. Initial public release; July 2, 2012.
For the newest revision of this paper, partial source code, and our online key-check service, visit **https://factorable.net**.

Mining Your Ps and Qs: Detection of Widespread Weak Keys in Network Devices

Nadia Heninger[†*] Zakir Durumeric[‡*] Eric Wustrow[‡] J. Alex Halderman

[†] *University of California, San Diego*
nadiah@cs.ucsd.edu

[‡] *The University of Michigan*
{zakir, ewust, jhalderm}@umich.edu

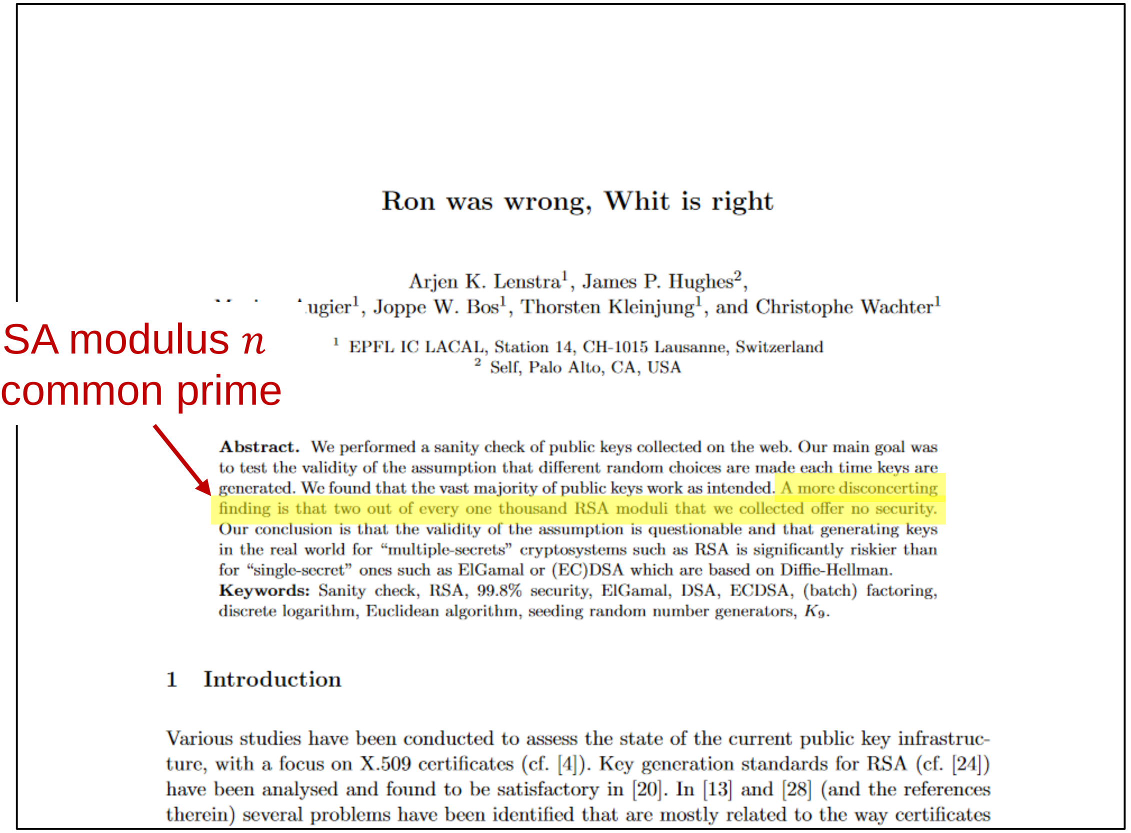
Abstract

RSA and DSA can fail catastrophically when used with malfunctioning random number generators, but the extent to which these problems arise in practice has never been comprehensively studied at Internet scale. We perform the largest ever network survey of TLS and SSH servers and present evidence that vulnerable keys are surprisingly widespread. We find that 0.75% of TLS certificates share keys due to insufficient entropy during key generation, and we suspect that another 1.70% come from the same faulty implementations and may be susceptible to compromise. Even more alarmingly, we are able to obtain RSA private keys for 0.50% of TLS hosts and 0.03% of SSH hosts, because their public keys shared nontrivial common factors due to entropy problems, and DSA private keys for 1.03% of SSH hosts, because of insufficient

expect that today's widely used operating systems and server software generate random numbers securely. In this paper, we test that proposition empirically by examining the public keys in use on the Internet.

The first component of our study is the most comprehensive Internet-wide survey to date of two of the most important cryptographic protocols, TLS and SSH (Section 3.1). By scanning the public IPv4 address space, we collected 5.8 million unique TLS certificates from 12.8 million hosts and 6.2 million unique SSH host keys from 10.2 million hosts. This is 67% more TLS hosts than the latest released EFF SSL Observatory dataset [18]. Our techniques take less than 24 hours to scan the entire address space for listening hosts and less than 96 hours to retrieve keys from them. The results give us a macroscopic perspective of the universe of keys.

Multiple RSA modulus n
sharing a common prime



Ron was wrong, Whit is right

Arjen K. Lenstra[1], James P. Hughes[2],
ugier[1], Joppe W. Bos[1], Thorsten Kleinjung[1], and Christophe Wachter[1]

[1] EPFL IC LACAL, Station 14, CH-1015 Lausanne, Switzerland
[2] Self, Palo Alto, CA, USA

Abstract. We performed a sanity check of public keys collected on the web. Our main goal was to test the validity of the assumption that different random choices are made each time keys are generated. We found that the vast majority of public keys work as intended. A more disconcerting finding is that two out of every one thousand RSA moduli that we collected offer no security. Our conclusion is that the validity of the assumption is questionable and that generating keys in the real world for "multiple-secrets" cryptosystems such as RSA is significantly riskier than for "single-secret" ones such as ElGamal or (EC)DSA which are based on Diffie-Hellman.
Keywords: Sanity check, RSA, 99.8% security, ElGamal, DSA, ECDSA, (batch) factoring, discrete logarithm, Euclidean algorithm, seeding random number generators, K_9.

1 Introduction

Various studies have been conducted to assess the state of the current public key infrastructure, with a focus on X.509 certificates (cf. [4]). Key generation standards for RSA (cf. [24]) have been analysed and found to be satisfactory in [20]. In [13] and [28] (and the references therein) several problems have been identified that are mostly related to the way certificates

https://factorable.net/weakkeys12.extended.pdf

https://eprint.iacr.org/2012/064

Summary

- Public-key encryption security goals: IND-CPA and IND-CCA

- ElGamal
  - Public-key encryption from Diffie-Hellman key exchange + symmetric encryption scheme
  - Hashed-ElGamal: hash DH key to obtain a symmetric key K

- RSA
  - Textbook-RSA: *not* a public-key encryption scheme directly (not IND-CPA secure!)
  - Shoup's RSA: encrypt random number with Textbook-RSA and derive symmetric key from hash function
    - IND-CPA / IND-CCA secure (depending on symmetric scheme)
    - Not much used in practice
    - In practice: pad message before encrypting with Textbook-RSA (PKCS#1v1.5, RSA-OAEP)
  - Must be hard: RSA-problem and factoring problem