Lecture 12 – Public-Key Encryption Schemes

COSC-466 Applied Cryptography

Adam O'Neill

Adapted from

http://cseweb.ucsd.edu/~mihir/cse107/

Today

We will see two types of public-key encryption schemes:

Today

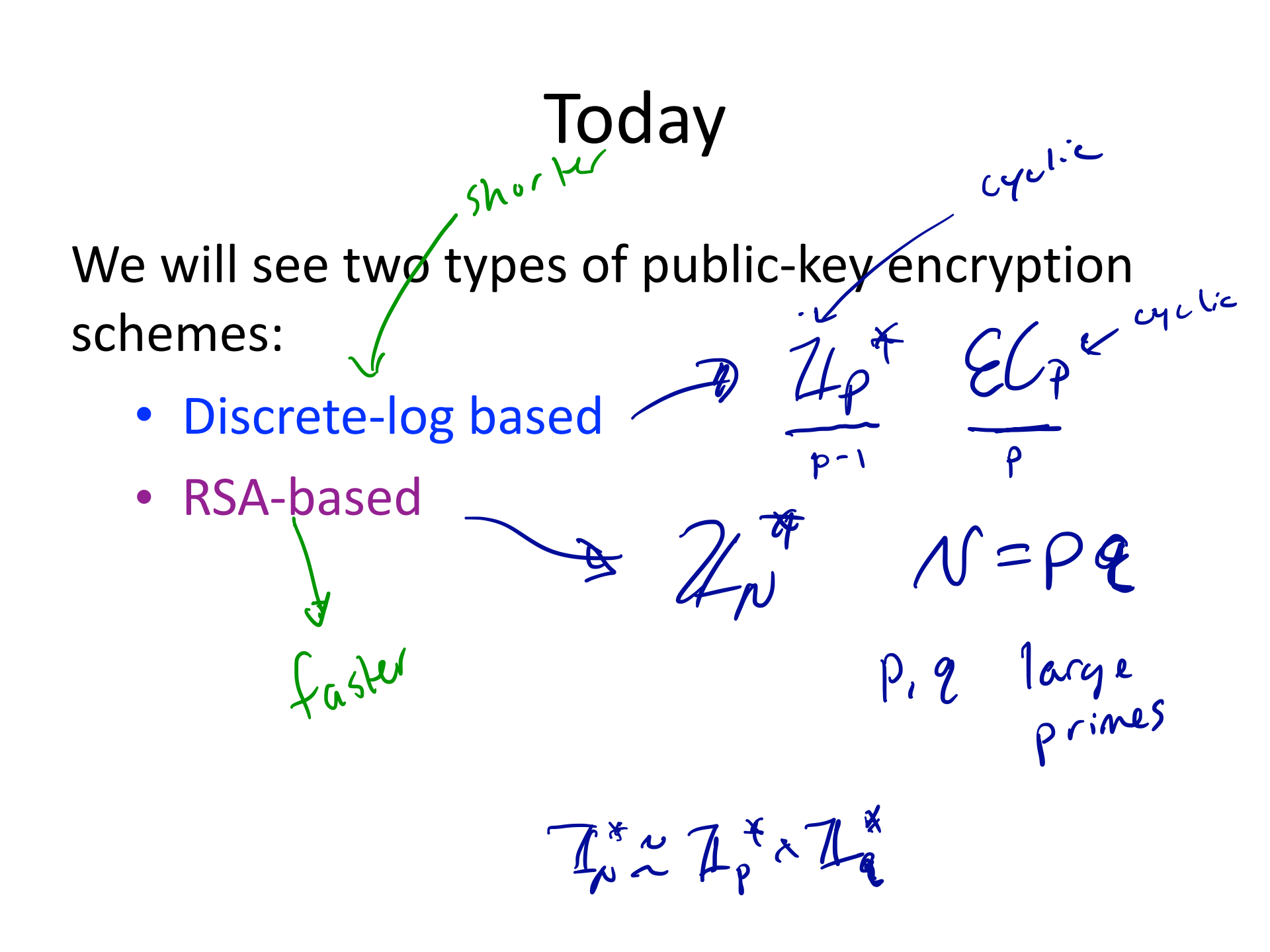
We will see two types of public-key encryption schemes:

- Discrete-log based

Today

We will see two types of public-key encryption schemes:

- Discrete-log based
- RSA-based



Today

We will see two types of public-key encryption schemes:

- Discrete-log based
- RSA-based

Today

We will see two types of public-key encryption schemes:

- Discrete-log based
- RSA-based

Pay careful attention to efficiency, security model and assumptions needed to prove security.

Diffie-Hellman Integrated Encryption scheme

DHIES

Let $G = \langle g \rangle$ be a cyclic group of order m and $H\colon\ G \to \{0,1\}^k$ a (public) hash function. The DHIES PKE scheme $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is defined for messages $M \in \{0,1\}^k$ via

Alg $\mathcal{K}$	**Alg** $\mathcal{E}_X(M)$	**Alg** $\mathcal{D}_x(Y, W)$
$x \xleftarrow{\$} \mathbf{Z}_m$	$y \xleftarrow{\$} \mathbf{Z}_m$; $Y \leftarrow g^y$	$K = Y^x$
$X \leftarrow g^x$	$K \leftarrow X^y$	$M \leftarrow H(K) \oplus W$
return (X, x)	$W \leftarrow H(K) \oplus M$	return M
	return (Y, W)	

Can be generalized as $SE_K(M)$

Correct decryption is assured because $K = X^y = g^{xy} = Y^x$

Note: This is a simplified version of the actual scheme.

generated a shared psuedo-OTP

Which Hash Function to Use?

Our analysis will assume H is "perfect"

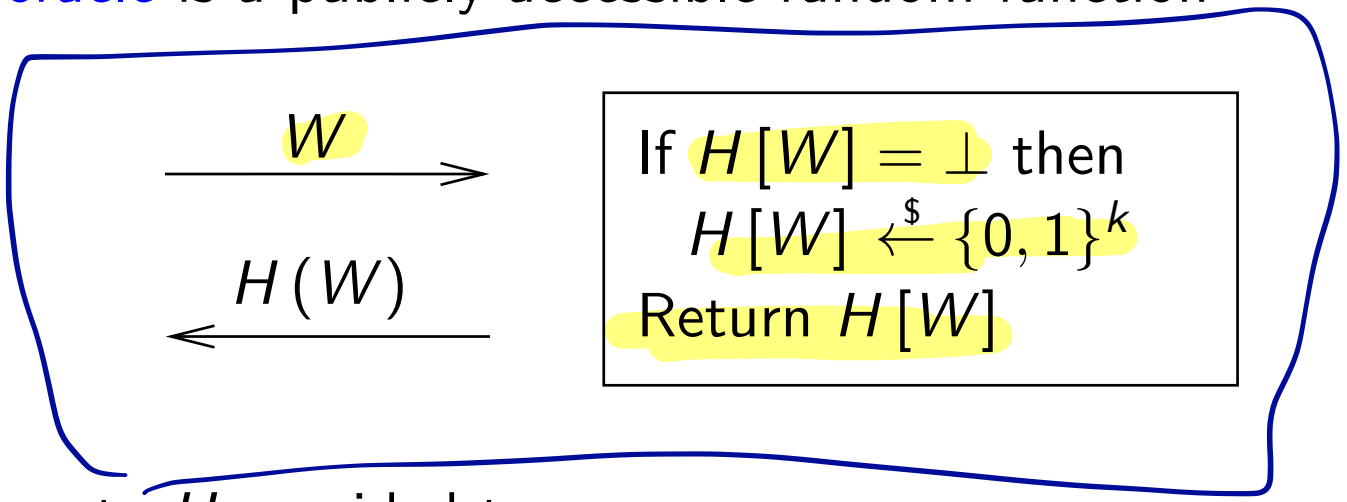
Question: What does this mean?

Answer: H will be modeled as a random oracle [BR93]

hash function viewed as a truly random function accessible only via oracle access (to all parties)

Random Oracle Model

A random oracle is a publicly-accessible random function



Oracle access to H provided to

- all scheme algorithms
- the adversary

The only access to H is oracle access.

Security of DHIES

The DHIES scheme $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ associated to cyclic group $G = \langle g \rangle$ and (public) hash function H can be proven IND-CPA assuming

- CDH is hard in G, and
- H is a "random oracle," meaning a "perfect" hash function.

In practice, $H(K)$ could be the first k bits of the sequence

$$\text{SHA256}(0^8 \| K) \| \text{SHA256}(0^7 1 \| K) \| \cdots$$

ECIES

- DHIES scheme where *G* is an appropriate elliptic curve group.

ECIES

- DHIES scheme where *G* is an appropriate elliptic curve group.
- Attractive performance: ciphertext size 160 bits, encryption is 2 160-bit exponentiations.

a^n

$|n|$ group operations w/ a

ECIES

- DHIES scheme where *G* is an appropriate elliptic curve group.
- Attractive performance: ciphertext size 160 bits, encryption is 2 160-bit exponentiations.
- Widely standardized and used, prevailing scheme in next-generation Internet protocols.

ECIES

- DHIES scheme where *G* is an appropriate elliptic curve group.
- Attractive performance: ciphertext size 160 bits, encryption is 2 160-bit exponentiations.
- Widely standardized and used, prevailing scheme in next-generation Internet protocols.

$\mathbb{Z}_N^*$ for $N = pq$, p & q are large primes (eg. 1024 bits)

$$\mathbb{Z}_N^* \cong \underbrace{\mathbb{Z}_p^*}_{p-1} \times \underbrace{\mathbb{Z}_q^*}_{q-1}$$

$$|\mathbb{Z}_N^*| = (p-1)(q-1) = \varphi(N).$$

$\Rightarrow$ if $a \in \mathbb{Z}_N^*$ then

$$a^{\varphi(N)} = 1 \bmod N$$

$$a \cdot b \bmod N$$

$$e \cdot d \bmod \varphi(N)$$

$a \in \mathbb{Z}_N^*$ $\quad a'a + NN' = 1$

RSA Function

A modulus N and encryption exponent e define the RSA function $f : \mathbf{Z}_N^* \to \mathbf{Z}_N^*$ defined by

$$RSA_{N,e}(x) = f(x) = x^e \mod N$$

e should be relatively prime to $\ell(N)$

for all $x \in \mathbf{Z}_N^*$.

A value $d \in Z_{\varphi(N)}^*$ satisfying $ed \equiv 1 \pmod{\varphi(N)}$ is called a decryption exponent.

Claim: The RSA function $f : \mathbf{Z}_N^* \to \mathbf{Z}_N^*$ is a permutation with inverse $f^{-1} : \mathbf{Z}_N^* \to \mathbf{Z}_N^*$ given by

$$f^{-1}(y) = y^d \mod N$$

$(x^e)^d \mod N = x^{ed \mod \ell(N)} \mod N$

$ed \mod \ell(N) = 1$

$= x \mod N$

Example

$N = 15 = 5 \cdot 3$

Let $N = 15$. So

$$\begin{aligned} \mathbf{Z}_N^* &= \{1, 2, 4, 7, 8, 11, 13, 14\} \\ \varphi(N) &= 8 \\ \mathbf{Z}_{\varphi(N)}^* &= \{1, 3, 5, 7\} \end{aligned}$$

Let $e = 3$ and $d = 3$. Then

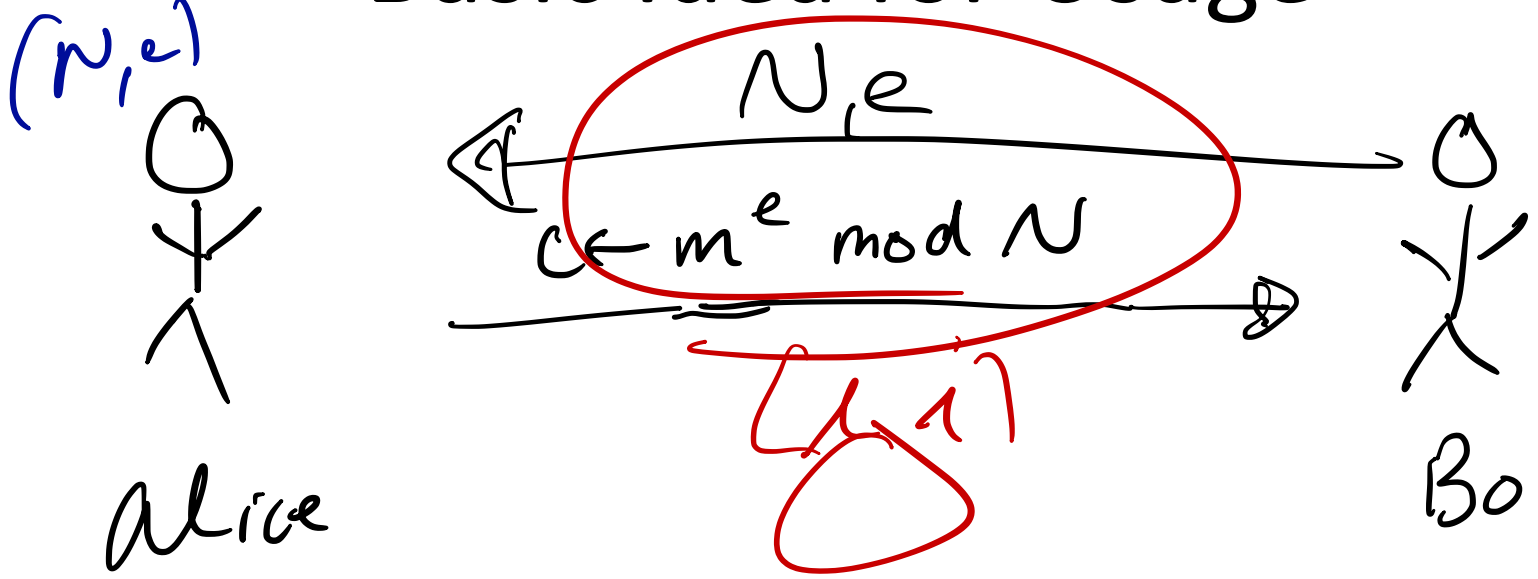
$$ed \equiv 9 \equiv 1 \pmod 8$$

Let

$$\begin{aligned} f(x) &= x^3 \bmod 15 \\ g(y) &= y^3 \bmod 15 \end{aligned}$$

x	$f(x)$	$g(f(x))$
1	1	1
2	8	2
4	4	4
7	13	7
8	2	8
11	11	11
13	7	13
14	14	14

Basic Idea for Usage



Chooses message $m \in \mathbb{Z}_N^*$

Eve observes

Should be hard for Eve to compute m

Chooses (N, e, d)

$m' \leftarrow c^d \bmod N$

m' is received message

RSA Generators

$(N, p, q, e, d) \xleftarrow{R} RSAgen(1^k)$

An RSA generator with security parameter k is an algorithm $\mathcal{K}_{rsa}$ that returns N, p, q, e, d satisfying

- p, q are distinct odd primes
- $N = pq$ and is called the (RSA) modulus
- $|N| = k$, meaning $2^{k-1} \leq N \leq 2^k$
- $e \in \mathbf{Z}^*_{\varphi(N)}$ is called the encryption exponent
- $d \in \mathbf{Z}^*_{\varphi(N)}$ is called the decryption exponent
- $ed \equiv 1 \pmod{\varphi(N)}$

Building an RSA Generator

Typically e is fixed $e = 2^{16}+1$ (prime)

$e = 3$

Then choose $p, q \xleftarrow{\$} \{0,1\}^{k/2}$ until

- p, q are primes
- e is relatively prime to $\varphi(N)$

primes are sufficiently dense that you don't need to try very many times

One-Wayness of RSA

relative to an RSA generator

The following should be hard:

Given: N, e, y where $y = f(x) = x^e \mod N$

Find: x

Formalism picks x at random and generates N, e via an RSA generator.

One-Wayness Game

Let $\mathcal{K}_{\mathrm{rsa}}$ be a RSA generator and I an adversary.

Game $\mathrm{OW}_{\mathcal{K}_{\mathrm{rsa}}}$

procedure Initialize
$(N, p, q, e, d) \xleftarrow{\$} \mathcal{K}_{\mathrm{rsa}}$
$x \xleftarrow{\$} \mathbf{Z}_N^*$; $y \leftarrow x^e \bmod N$
return N, e, y

procedure Finalize(x')
return $(x = x')$

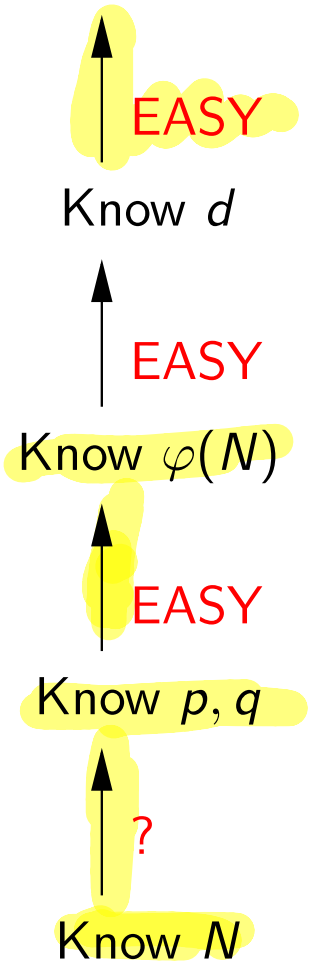
The ow-advantage of I is

$$\mathbf{Adv}^{\mathrm{ow}}_{\mathcal{K}_{\mathrm{rsa}}}(I) = \Pr\left[\mathrm{OW}^{I}_{\mathcal{K}_{\mathrm{rsa}}} \Rightarrow \mathrm{true}\right]$$

Inverting RSA

$\varphi(N) = (p-1)(q-1)$

Inverting RSA : given N, e, y find x such that $x^e \equiv y \pmod{N}$



because $f^{-1}(y) = y^d \bmod N$

because $d = e^{-1} \bmod \varphi(N)$

because $\varphi(N) = (p-1)(q-1)$

Factoring and RSA

- We know if we can break factoring we can break RSA
- the converse is open; factoring is potentially harder.

```
Factor(N) // N = pq
  For i = 1 to √N
    if i | N then ret i, N/i
```

exponential in $|N|$ ← bit length of N.

Best Algorithms and Implication

Algorithm	**Time taken to factor N**
Naive	$O(e^{0.5 \ln N})$
Quadratic Sieve (QS)	$O(e^{c(\ln N)^{1/2}(\ln \ln N)^{1/2}})$
Number Field Sieve (NFS)	$O(e^{1.92(\ln N)^{1/3}(\ln \ln N)^{2/3}})$

Sub-exponential

NFS ⇒ for 80 bit factoring sec you need

$|N| = 1024$

"Plain RSA" Encryption

The plain RSA PKE scheme $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ associated to RSA generator $\mathcal{K}_{\text{rsa}}$ is

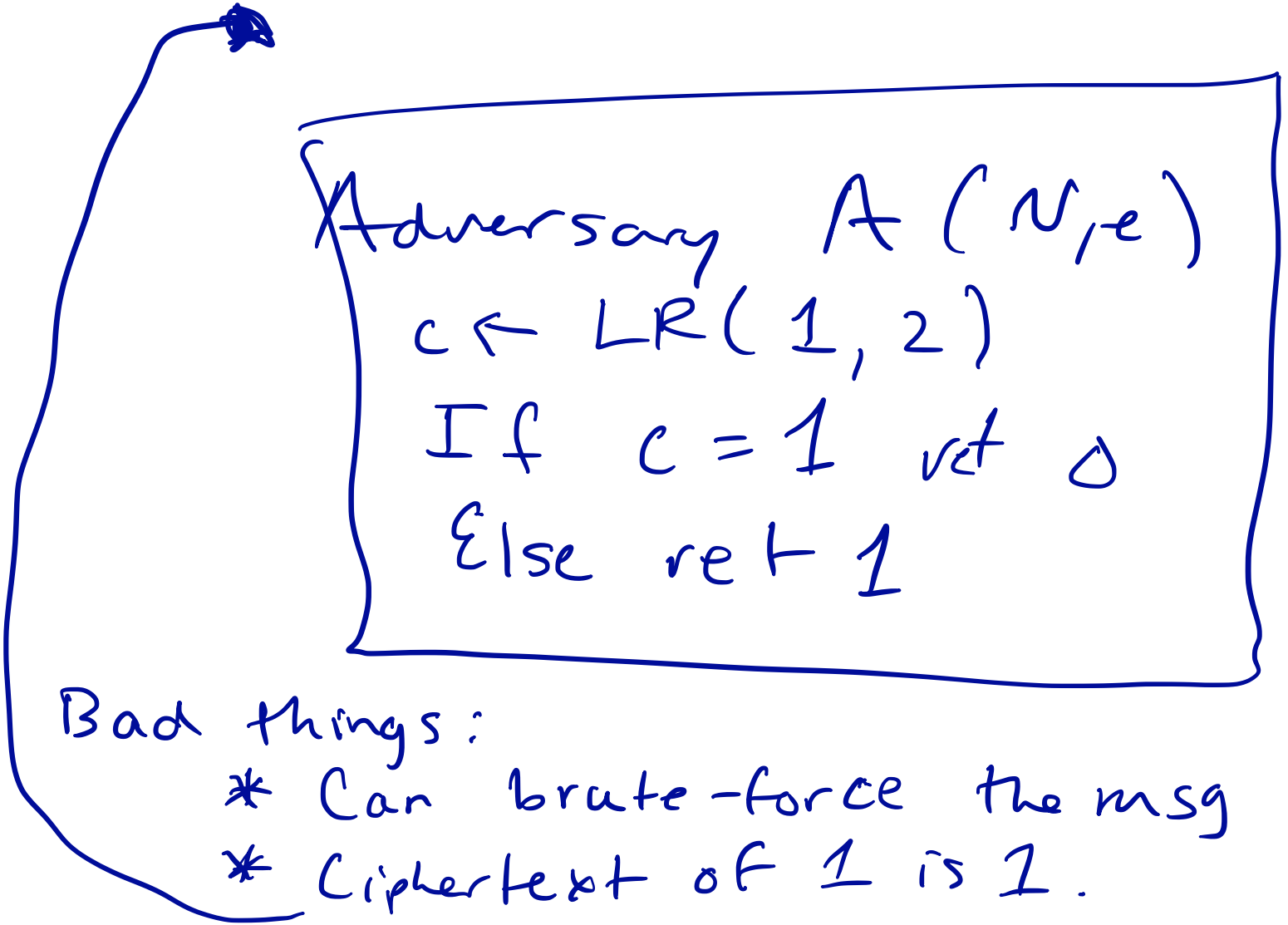
Alg $\mathcal{K}$	**Alg** $\mathcal{E}_{pk}(M)$	**Alg** $\mathcal{D}_{sk}(C)$
$(N, p, q, e, d) \xleftarrow{\$} \mathcal{K}_{\text{rsa}}$ $pk \leftarrow (N, e)$ $sk \leftarrow (N, d)$ return (pk, sk)	$C \leftarrow M^e \mod N$ return C	$M \leftarrow C^d \mod N$ return M

The "easy-backwards with trapdoor" property implies

$$\mathcal{D}_{sk}(\mathcal{E}_{pk}(M)) = M$$

for all $M \in \mathbf{Z}_N^*$.

Security Analysis



Fact. RSA is multiplicatively homomorphic.

What this means is that given $x_1^e \bmod N$ and $x_2^e \bmod N$ we can efficiently compute $(x_1 \cdot x_2)^e \bmod N$ for any (unknown) $x_1, x_2 \in \mathbb{Z}_N^*$

From $y_1 = RSA_{N,e}(x_1)$ & $y_2 = RSA_{N,e}(x_2)$ can compute $RSA_{N,e}(x_1 \cdot x_2)$

this is bad for "active" attacks (not covered in this course)

proof: just multiply

$= x_1^e \cdot x_2^e \bmod N = y_1 \cdot y_2 \bmod N = (x_1 x_2)^e \bmod N$

"Simple RSA" Encryption Scheme

The SRSA PKE scheme $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ associated to RSA generator $\mathcal{K}_{\text{rsa}}$ and (public) hash function H: $\{0,1\}^* \to \{0,1\}^k$ encrypts k-bit messages via:

Alg $\mathcal{K}$

$(N, p, q, e, d) \xleftarrow{\$} \mathcal{K}_{\text{rsa}}$
$pk \leftarrow (N, e)$
$sk \leftarrow (N, d)$
return (pk, sk)

Alg $\mathcal{E}_{N,e}(M)$

$x \xleftarrow{\$} \mathbf{Z}_N^*$
$K \leftarrow H(x)$
$C_a \leftarrow x^e \bmod N$
$C_s \leftarrow K \oplus M$
return (C_a, C_s)

Alg $\mathcal{E}_{N,d}(C_a, C_s)$

$x \leftarrow C_a^d \bmod N$
$K \leftarrow H(x)$
$M \leftarrow C_s \oplus K$
return M

Can generalize to $SE_K(m)$ for symmetric encryption scheme SE.

Security Analysis

The SRSA PKE scheme $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ associated to RSA generator $\mathcal{K}_{\mathrm{rsa}}$ and (public) hash function H: $\{0,1\}^* \to \{0,1\}^k$ can be proven IND-CPA assuming

- $\mathcal{K}_{\mathrm{rsa}}$ is one-way
- H is a "random oracle," meaning a "perfect" hash function.

In practice, $H(K)$ could be the first k bits of the sequence

$$\mathsf{SHA256}(0^8 \| K) \| \mathsf{SHA256}(0^7 1 \| K) \| \cdots$$

PKCS #1 v1.5

public-key cryptography standards

Algorithm $\mathcal{E}_{N,e}(m)$ // $|m| << |N|$

Choose r at random of suitable length

$c \leftarrow \mathcal{E}_{N,e}(r \ldots 00 \ldots m)^e \bmod N$

return c

decryption strips this off

Security Analysis

Is this secure? More specifically:
For what length r is this secure?

Minimum length of r: 80

Algebraic attacks imply r has to be much larger.

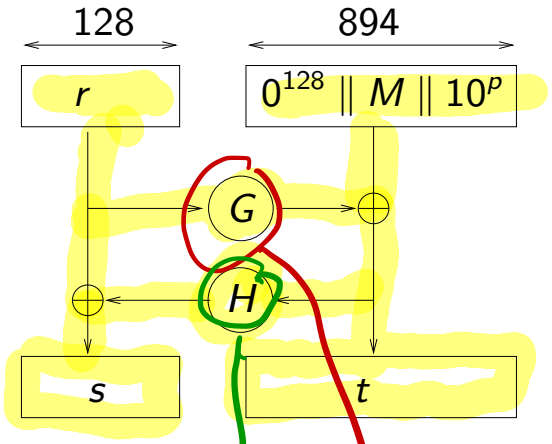
Even then it seems hard to reduce IND-CPA security of PKCS #1 v1.5 to one-wayness of RSA

RSA-OAEP (PKCS #1 v2.1) [BR'94]

Receiver keys: $pk = (N, e)$ and $sk = (N, d)$ where $|N| = 1024$
Hash functions: G: $\{0,1\}^{128} \to \{0,1\}^{894}$ and H: $\{0,1\}^{894} \to \{0,1\}^{128}$

Algorithm $\mathcal{E}_{N,e}(M)$ // $|M| \le 765$

$r \xleftarrow{\$} \{0,1\}^{128}$; $p \leftarrow 765 - |M|$

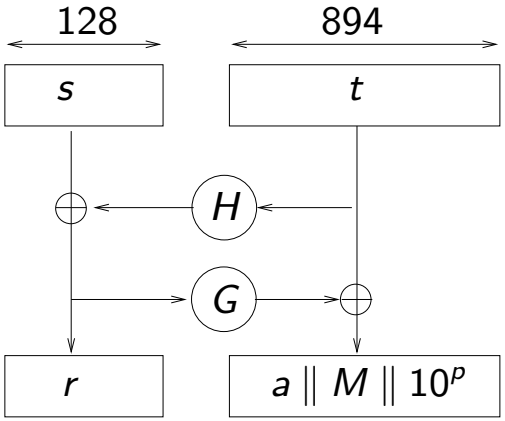


$x \leftarrow s||t$
$C \leftarrow x^e \bmod N$
return C

expanding
Compressing

Algorithm $\mathcal{D}_{N,d}(C)$ // $C \in \mathbb{Z}_N^*$

$x \leftarrow C^d \bmod N$
$s||t \leftarrow x$



if $a = 0^{128}$ **then return** M
else return $\perp$

Attractive Features

- Same ciphertext length as RSA PKCS v1.5.

Attractive Features

- Same ciphertext length as RSA PKCS v1.5.
- But has provable security guarantees:

Attractive Features

- Same ciphertext length as RSA PKCS v1.5.
- But has provable security guarantees:
  - IND-CPA in the RO model assuming RSA is one-way [BR'94].

Attractive Features

- Same ciphertext length as RSA PKCS v1.5.
- But has provable security guarantees:
  - IND-CPA in the RO model assuming RSA is one-way [BR'94].
  - IND-CCA in the RO model assuming RSA is one-way [FOPS'00].

→ "active" attacks

Attractive Features

- Same ciphertext length as RSA PKCS v1.5.
- But has provable security guarantees:
  - IND-CPA in the RO model assuming RSA is one-way [BR'94].
  - IND-CCA in the RO model assuming RSA is one-way [FOPS'00].
  - IND-CPA in the standard model assuming RSA is "lossy" [KOS'10].

Careful in Practice

- Attacks are possible if *d* is too small, timing information leaks, *etc.* (cf. “Twenty Years of Attacks on RSA” by Dan Boneh).

Careful in Practice

- Attacks are possible if *d* is too small, timing information leaks, *etc.* (cf. "Twenty Years of Attacks on RSA" by Dan Boneh).
- Lenstra *et al.* recently found many keys share a common divisor due to buggy randomness!!

Careful in Practice

- Attacks are possible if *d* is too small, timing information leaks, *etc.* (cf. "Twenty Years of Attacks on RSA" by Dan Boneh).
- Lenstra *et al.* recently found many keys share a common divisor due to buggy randomness!!
- Use open-source, publicly scrutinized implementations!

How does ECIES compare to RSA-OEAP??