Modern Cryptography: Lecture 12

Public Key Encryption II/II

Daniel Slamanig

AUSTRIAN INSTITUTE OF TECHNOLOGY

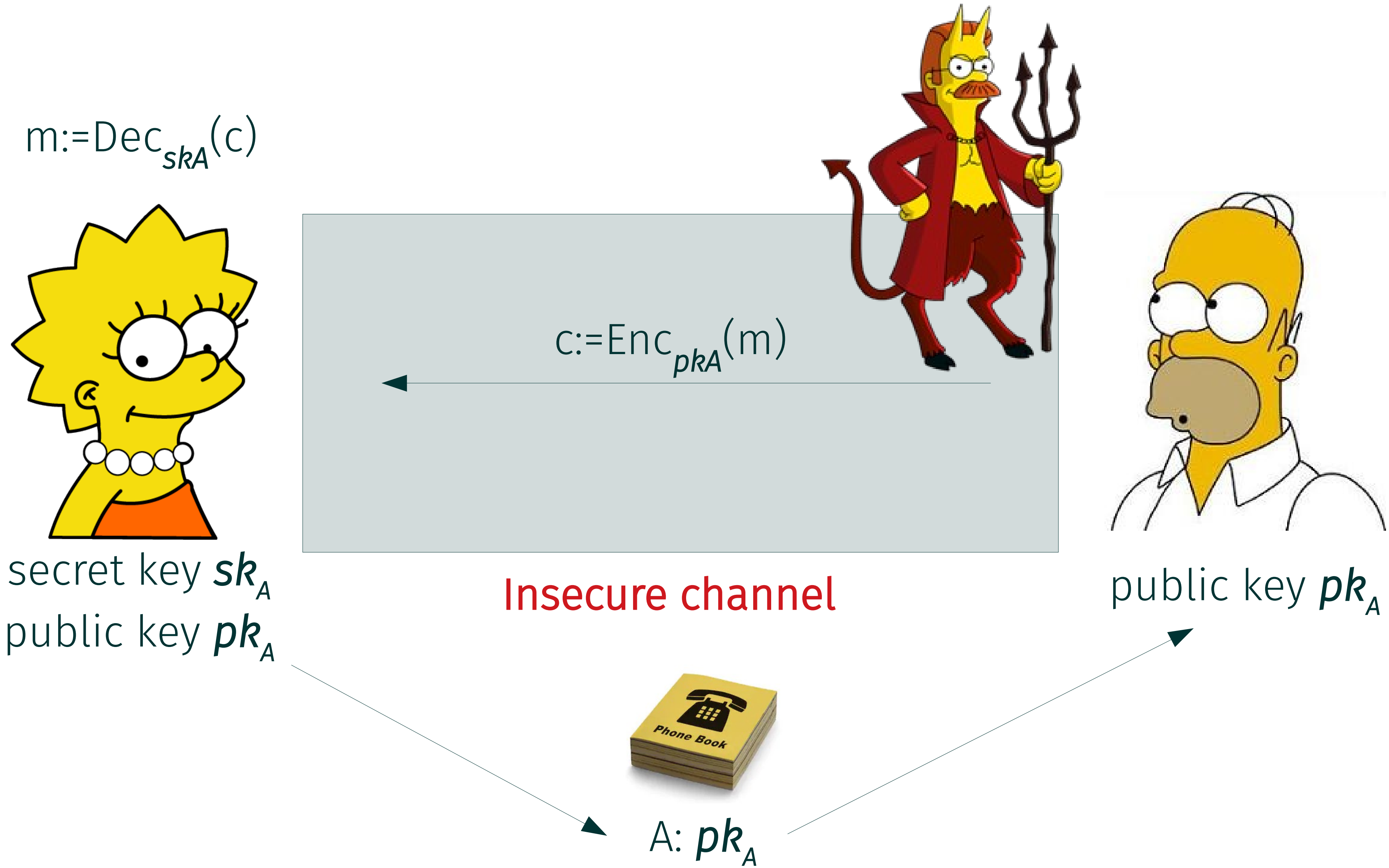
Organizational

- Where to find the slides and homework?
  - https://danielslamanig.info/ModernCrypto18.html
- How to contact me?
  - daniel.slamanig@ait.ac.at
- Tutor: Karen Klein
  - karen.klein@ist.ac.at

- Official page at TU, Location etc.
  - https://tiss.tuwien.ac.at/course/courseDetails.xhtml?dswid=8632&dsrid=679&courseNr=192062&semester=2018W
- Tutorial, TU site
  - https://tiss.tuwien.ac.at/course/courseAnnouncement.xhtml?dswid=5209&dsrid=341&courseNumber=192063&courseSemester=2018W

- Exam for the second part: Thursday 31.01.2019 15:00-17:00 (Tutorial slot)

Recap: Public Key Encryption



ElGamal Encryption - Intuition

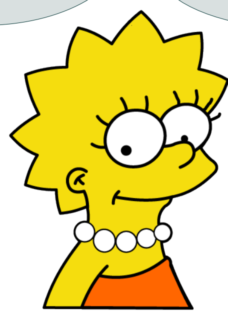
sk

pk

(G, q, g)

$a \xleftarrow{\$} \mathbb{Z}_q; A \leftarrow g^a$

$b \xleftarrow{\$} \mathbb{Z}_q; B \leftarrow g^b$



$\cancel{A}$

$(B, K \cdot M)$

$K \leftarrow B^a$

$K \leftarrow A^b$

$M := K \cdot M \cdot K^{-1}$

- Take the DH KE protocol and fix “first message” A (together with group parameters) of Alice as her public key (secret a is her secret key)
- To encrypt to Alice, Bob chooses ephemeral key B, uses K as one-time pad to encrypt a message M ∈ G and additionally sends B
- Any KE protocol that is secure in the presence of an eavesdropper (Def. 10.1) yields an IND-CPA secure PKE

ElGamal Encryption

- Gen(1^n): Run $(G, q, g) \leftarrow \mathcal{G}(1^n)$, pick $x \xleftarrow{\$} \mathbb{Z}_q$ compute $y := g^x$ and output $(sk, pk) := ((G, q, g, x), (G, q, g, y))$

- Enc (m, pk): On input $m \in G$ and $pk = (G, q, g, y)$, pick $r \xleftarrow{\$} \mathbb{Z}_q$, compute and output

$$C := (g^r, m \cdot y^r)$$

- Dec (C, sk): On input $C = (C_1, C_2)$ and $sk = (G, q, g, x)$, compute and output

$$m := C_2 \cdot (C_1^x)^{-1}$$

Correctness: $C_2 \cdot (C_1^x)^{-1} = m \cdot y^r \cdot ((g^r)^x)^{-1} = m \cdot y^r \cdot (y^r)^{-1} = m \cdot 1 = m$

We can also consider (G,q,g) as system parameters pp which are input to Gen and remove them from the keys.
All algorithms then implicit have access to pp. So, many users can generate keys with respect to the same parameters (as typically the case with elliptic curve cryptography).

ElGamal Encryption: Security Analysis I/III

LEMMA 11.15 Let G be a finite group, and let $m \in G$ be arbitrary. Then choosing uniform $k \in G$ and setting $k' := k \cdot m$ gives the same distribution for k' as choosing uniform $k' \in G$. Put differently, for any $g' \in G$ we have

$$\Pr[k \cdot m = g'] = 1/|G|,$$

where the probability is taken over uniform choice of $k \in G$.

This lemma gives us a perfectly secret private-key encryption scheme with message space G (one-time pad on a different group).

In ElGamal the ciphertext is $C := (g^r, m \cdot y^r)$ where $y = g^x$ is the public key

Using the lemma we construct an alternative ElGamal "encryption method" where we use a random secret key g^z to encrypt

- Here a ciphertext is of the form $C := (g^r, m \cdot g^z)$
- The value g^z will be unknown to the adversary, i.e., m is information-theoretically hidden and the adversary can only guess the challenge bit.

We will show that in the IND-CPA game no adversary can detect that we modify the ElGamal encryption method under the DDH assumption

ElGamal Encryption: Security Analysis II/III

- We recall the DDH assumption

DEFINITION 8.63: We say that the DDH problem is hard relative to $\mathcal{G}$ if for all PPT algorithms $\mathcal{A}$ there is a negligible function negl such that

$$\Pr[\mathcal{A}(G, q, g, g^x, g^r, g^z) = 1] - \Pr[\mathcal{A}(G, q, g, g^x, g^r, g^{xr}) = 1] \leq \text{negl}(n),$$

where in each case the probabilities are taken over the experiment in which $\mathcal{G}(1^n)$ outputs (G, q, g), and then uniform $x, r, z \in \mathbb{Z}_q$ are chosen.

- We fix some notation
  - $\text{PubK}^{\text{cpa}}_{\mathcal{A},\mathcal{EG}}(n)$ represents the IND-CPA experiment for ElGamal $\mathcal{EG}$
  - $\text{PubK}^{\text{cpa}}_{\mathcal{A},\mathcal{EG}'}(n)$ the experiment for modified ElGamal $\mathcal{EG}'$
  - Recall that $\Pr[\text{PubK}^{\text{cpa}}_{\mathcal{A},\mathcal{EG}}(n)=1]$ denotes the probability that adversary A wins the game (for ElGamal in this case)

ElGamal Encryption: Security Analysis III/III

IND-CPA Game with original ElGamal $\mathcal{EG}$

$(G, q, g) \xleftarrow{\$} \mathcal{G}(1^n),\ x \xleftarrow{\$} \mathbb{Z}_q,\ y := g^x$
$((m_0, m_1), \text{state}) \xleftarrow{\$} \mathcal{A}(G, q, g, y)$
$b \xleftarrow{\$} \{0, 1\}$
$r \xleftarrow{\$} \mathbb{Z}_q\ C_1 := g^r,\ Z := y^r,\ C_2 := m_b \cdot Z$
$b^* \xleftarrow{\$} \mathcal{A}(\text{state}, (C_1, C_2))$
if $b = b^*$ then
  return 1
else
  return 0

$\Pr[\mathrm{PubK}^{\mathrm{cpa}}_{\mathcal{A},\mathcal{EG}}(n)=1]$

IND-CPA Game with modified ElGamal $\mathcal{EG}'$

$(G, q, g) \xleftarrow{\$} \mathcal{G}(1^n),\ x \xleftarrow{\$} \mathbb{Z}_q,\ y := g^x$
$((m_0, m_1), \text{state}) \xleftarrow{\$} \mathcal{A}(G, q, g, y)$
$b \xleftarrow{\$} \{0, 1\}$
$r \xleftarrow{\$} \mathbb{Z}_q\ C_1 := g^r,\ z \xleftarrow{\$} \mathbb{Z}_q,\ Z := g^z,\ C_2 := m_b \cdot Z$
$b^* \xleftarrow{\$} \mathcal{A}(\text{state}, (C_1, C_2))$
if $b = b^*$ then
  return 1
else
  return 0

$\Pr[\mathrm{PubK}^{\mathrm{cpa}}_{\mathcal{A},\mathcal{EG}'}(n)=1] = 1/2$

Reduction $\mathcal{B}$ to DDH

$\underline{\mathcal{B}^{\mathcal{A}}(G, q, g, U, V, W)}$
$((m_0, m_1), \text{state}) \xleftarrow{\$} \mathcal{A}(G, q, g, U)$
$b \xleftarrow{\$} \{0, 1\}$
$C_2 := m_b W$
$b^* \xleftarrow{\$} \mathcal{A}(\text{state}, (V, C))$
if $b = b^*$ then
  return 1
else
  return 0

$\Pr[\mathcal{B}^{\mathcal{A}}(G,q,g,g^x,g^r,g^{xr})=1] = \Pr[\mathrm{PubK}^{\mathrm{cpa}}_{\mathcal{A},\mathcal{EG}}(n)=1]$

$\Pr[\mathcal{B}^{\mathcal{A}}(G,q,g,g^x,g^r,g^z)=1] = \Pr[\mathrm{PubK}^{\mathrm{cpa}}_{\mathcal{A},\mathcal{EG}'}(n)=1]$

$\mathrm{negl}(n) \geq |\Pr[\mathcal{B}^{\mathcal{A}}(G,q,g,g^x,g^r,g^z)=1] - \Pr[\mathcal{B}^{\mathcal{A}}(G,q,g,g^x,g^r,g^{xr})=1]|$

$= |1/2 - \Pr[\mathrm{PubK}^{\mathrm{cpa}}_{\mathcal{A},\mathcal{EG}}(n)=1]|$

This gives us: $\Pr[\mathrm{PubK}^{\mathrm{cpa}}_{\mathcal{A},\mathcal{EG}}(n)=1] \leq 1/2 + \mathrm{negl}(n)$ ∎

ElGamal Encryption: Properties

- Homomorphic
  - Given two ciphertexts $(C_1, C_2) = (g^r, m \cdot y^r)$ and $(C'_1, C'_2) = (g^{r'}, m' \cdot y^{r'})$ we can compute $(C_1, C_2) \boxdot (C'_1, C'_2)$ as componentwise multiplication
    - $(C_1, C_2) \boxdot (C'_1, C'_2) = (g^r \cdot g^{r'}, (m \cdot y^r) \cdot (m' \cdot y^{r'})) = (g^{r+r'}, (m \cdot m') \cdot y^{r+r'})$
  - Encode message $m \in \mathbb{Z}_q$ in the exponent as g^m ("Exponential ElGamal")
    - $(C_1, C_2) \boxdot (C'_1, C'_2) = (g^r \cdot g^{r'}, (g^m \cdot y^r) \cdot (g^{m'} \cdot y^{r'})) = (g^{r+r'}, (g^{m+m'}) \cdot y^{r+r'})$
    - Decryption requires computing DLOG! Messages from restricted space
- Perfectly Re-randomizable
  - Homomorphic property with encryption of 1: $(C'_1, C'_2) = (g^{r'}, y^{r'})$
    - $(C_1, C_2) \boxdot (C'_1, C'_2) = (g^{r+r'}, m \cdot y^{r+r'})$
    - Identically distributed to "fresh" ciphertexts
- Key-private
  - Ciphertext does not leak public key (if keys are with respect to same pps)
  - Adversary can not tell apart ciphertexts for chosen messages encrypted either under public key y or y'

CCA Security

- Like in the private-key (symmetric) setting we will also consider a notion strictly stronger than IND-CPA security
- IND-CCA security: we give the adversary access to a decryption oracle
  - Even more of a concern in the PKE setting: parties may receive ciphertext from multiple potentially unknown senders
  - Adversary intercepts ciphertext from client to server. Say encryption of a symmetric key (e.g., pre-master secret in TLS) used to secure a connection
    - Adversary may send modified versions of the ciphertext to the server to check how the server reacts when trying to decrypt
    - Like with padding-oracles the behaviour of the server may help to recover the original message (without seeing the decrypted message)
    - Huge problem in practice! (will discuss later)
  - Auction application. Assume an application where bidders send encrypted bids
    - Non-CCA secure schemes are typically malleable (given c change underlying message from m to e.g., 2m)
    - Intercepting an encrypted bid from competitor, can always double the unknown value

Oracle $\mathsf{Dec}'_{\mathsf{sk}}(\cdot)$

if $c_i = c^*$

return $\perp$

else return $\mathsf{Dec}_{\mathsf{sk}}(c_i)$

c_i m_i

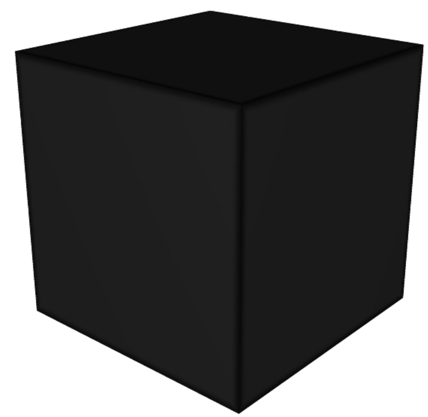
$\mathcal{A}$

pk

(m_0, m_1)

c^*

b^*



$(\mathsf{pk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{Gen}(1^n)$

$b \xleftarrow{\$} \{0, 1\}$

$c^* \xleftarrow{\$} \mathsf{Enc}_{\mathsf{pk}}(m_b)$

if $b^* = b$

return 1

else return 0

A public-key encryption scheme Π = (Gen, Enc, Dec) has indistinguishable encryptions under chosen-ciphertext attacks if for all probabilistic polynomial-time adversaries $\mathcal{A}$ there is a negligible function negl s.t.

$$\Pr[\mathrm{PubK}^{\mathrm{cca}}_{\mathcal{A},\Pi}(n)=1] \leq \tfrac{1}{2} + \mathrm{negl}(n) .$$

IND-CCA Security – Discussion

- Access to oracle Dec' can be limited
  - Non-adaptive access: only before seeing c* (i.e., queries to decryption do not depend on challenge ciphertext).
    - IND-CCA1 security (aka security against "lunch-time attacks")
  - Adaptive access: queries to Dec' can depend on c*
    - IND-CCA2 security (aka "adaptive" security)
- IND-CCA1 is stricly weaker than IND-CCA2
  - Any IND-CCA2 scheme is obviously IND-CCA1 secure
  - There are schemes that are IND-CCA1 but not IND-CCA2 (e.g., Cramer-Shoup "lite")
- When we speak of IND-CCA security, we alsways mean <u>adaptive</u> access to the decryption oracle (i.e., CCA2 security)

Hierarchy of Security Notions

Cryptographic applications that require homomorphic properties/re-randomizability

OW-CPA $\Leftarrow$ IND-CPA $\Leftarrow$ IND-CCA1 $\Leftarrow$ IND-CCA2

Building block in cryptographic protocols

Practical encryption applications (and only schemes satisfying this notion!)

increasing strength $\longrightarrow$

Exercise: show reductions between these notions

CCA Security – Some Facts

- IND-CCA notion under mutliple encryptions can be shown analogously to as we have done for IND-CPA security
- Claim 11.7 (PKE for larger messages by blockwise encryption) does not hold for CCA security. Why?
  - Let us wlog assume 2 blocks: $\mathrm{Enc}'_{pk}(m) := (\mathrm{Enc}_{pk}(m_1), \mathrm{Enc}_{pk}(m_2))$
  - Receive challenge ciphertext $c^*=(c_1^*,c_2^*)$ and send $c^{*\prime}= (c_2^*,c_1^*)$ to Dec oracle?
- As with CPA secure encryption, for longer messages a “hybrid encryption” approach will be used
  - Will formalize “hybrid encryption”
  - Generic composition of a key-encapsulation mechanisms (KEM) and a data-encapsulation mechanism (DEM)
  - KEM will be a “lightweight” PKE, DEM a symmetric encryption scheme

Key-Encapsulation Mechanism: Definition

DEFINITION 11.9 A **key-encapsulation mechanism (KEM)** is a triple of PPT algorithms (Gen, Encaps, Decaps) such that:

1. The **key-generation** algorithm **Gen** takes as input the security parameter 1^n and outputs a pair of keys (pk, sk) (keys have length at least n and n can be determined from pk).
2. The **encapsulation** algorithm **Encaps** takes as input a public key pk and the secuirty parameter 1^n. It outputs a ciphertext c and a key $k \in \{0,1\}^{p(n)}$, where p is the key length. We write this as $(c,k) \leftarrow \mathrm{Encaps}_{pk}(1^n)$ (or $(c,k) \leftarrow \mathrm{Encaps}(1^n,pk)$).
3. The deterministic **decapsulation** algorithm **Decaps** takes as input a private key sk and a ciphertext c, and outputs a key k or a special symbol $\perp$ denoting failure. We write this as $k := \mathrm{Decaps}_{sk}(c)$ (or as $k := \mathrm{Decaps}(sk, c)$).

It is required that, except possibly with negligible probability over $(pk, sk) \leftarrow \mathrm{Gen}(1^n)$, we have that if $(c,k) \leftarrow \mathrm{Encaps}_{pk}(1^n)$, then $k := \mathrm{Decaps}_{sk}(c)$.

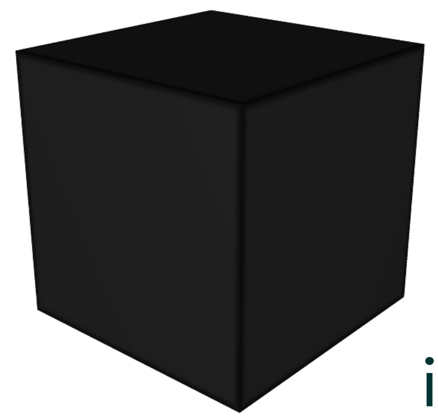
Security Definitions for KEMs (IND-CPA)

Define them for IND-CPA and IND-CCA analogoulsy to PKE (CCA game provides a Decaps* oracle)

$\mathcal{A}$

$(\mathrm{pk}, (c^*, k^*))$

b^*



$(\mathrm{pk}, \mathrm{sk}) \xleftarrow{\$} \mathrm{Gen}(1^n)$

$(c^*, k) \xleftarrow{\$} \mathrm{Encaps}_{\mathrm{pk}}(1^n)$

$b \xleftarrow{\$} \{0, 1\}$

if $b = 0$

$k^* := k$

else $k^* \xleftarrow{\$} \{0, 1\}^n$

if $b^* = b$

return 1

else return 0

A key-encapsulation mechansims Π = (Gen, Enc, Dec) is IND-CPA-secure if for all PPT adversaries $\mathcal{A}$ there is a negligible function negl s.t.

$$\Pr[\mathrm{KEM}^{\mathrm{cpa}}_{\mathcal{A},\Pi}(n)=1] \leq \tfrac{1}{2} + \mathrm{negl}(n)\ .$$

IND-CPA/IND-CCA PKE implies IND-CPA/IND-CCA KEM

Given any PKE scheme* Π = (Gen, Enc, Dec) with message space $\mathcal{M}$ construct a KEM Π' = (Gen', Encaps, Decaps) as follows:

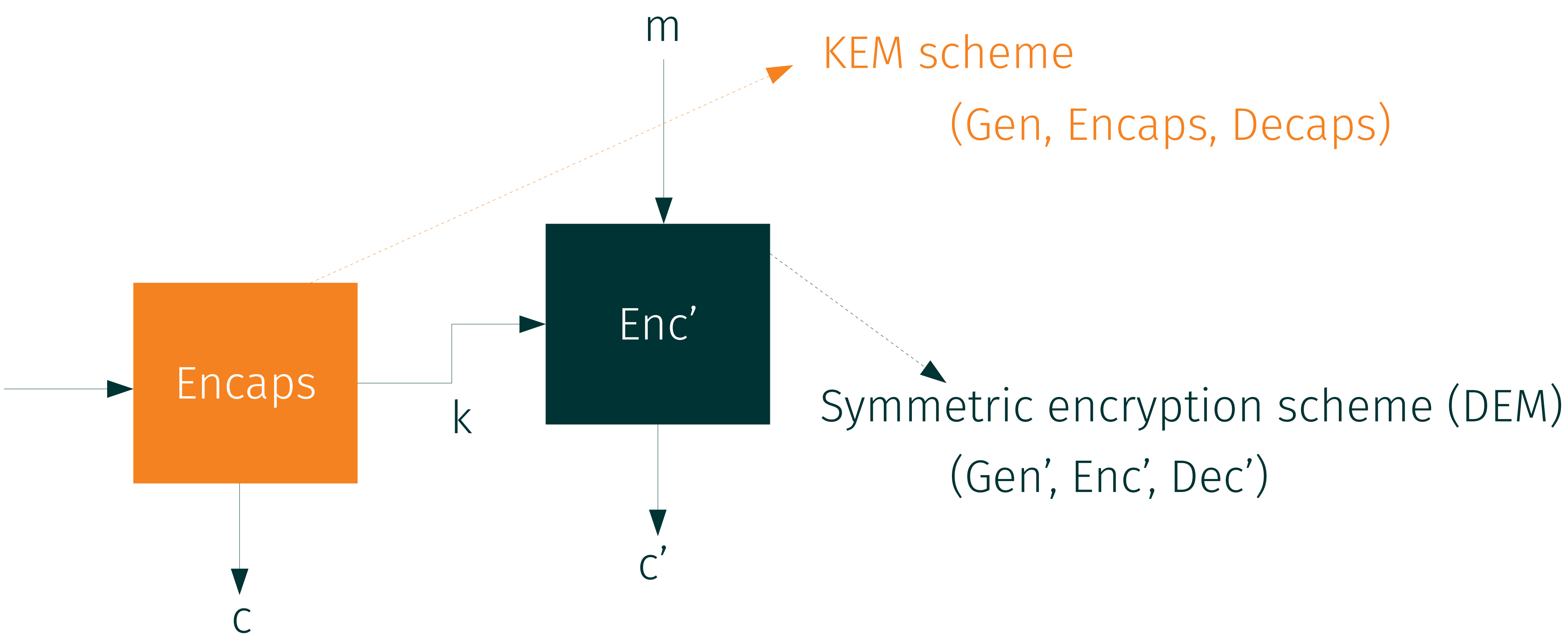
- **Π'.Gen'(1^n):** Run (pk,sk) ← Π.Gen(1^n)
- **Π'.Encaps$_{pk}$(1^n):** Choose $k \leftarrow_{\$} \mathcal{M}$, compute $c \leftarrow \Pi.\mathrm{Enc}_{pk}(k)$ and output (c,k)
- **Π'.Decaps$_{sk}$(c):** Output $\Pi.\mathrm{Dec}_{sk}(c)$

– * we need to assume that the message space of Π has sufficient min-entropy (so that keys k are "random enough")

- Dedicated KEM constructions typically far more efficient!

Hybrid Encryption from KEM/DEM

- Given a KEM and a DEM (symmetric encryption scheme) we can easily construct a hybrid encryption scheme
- In practice such schemes are significantly more efficient than pure PKE schemes
  - As soon as the message would require more than one PKE invocation



Hybrid Encryption from KEM/DEM: Generic Construction

Let Π = (Gen, Encaps, Decaps) be a KEM with key length n, and let Π' = (Gen' , Enc' , Dec') be a private-key encryption (DEM) scheme. We construct a public-key encryption scheme Π^{hy} = (Gen^{hy} , Enc^{hy} , Dec^{hy}) as follows:

- Gen^{hy}: On input 1^n output (pk, sk) ← Gen(1^n)
- Enc^{hy}: On input a public key pk and a message m ∈ {0, 1}* do:
  - Compute (c, k) ← $\text{Encaps}_{pk}(1^n)$
  - Compute c' ← $\text{Enc}'_k(m)$
  - Output the ciphertext (c, c')
- Dec^{hy}: on input a private key sk and a ciphertext (c, c') do:
  - Compute k := $\text{Decaps}_{sk}(c)$
  - Output the message m := $\text{Dec}'_k(c')$

THEOREM 11.12: If Π is a CPA-secure KEM and Π’ is a private-key encryption scheme that has indistinguishable encryptions in the presence of an eaves dropper, then Π^{hy} is a CPA-secure public-key encryption scheme.

Proof idea:

$$
\begin{tikzcd}
\left(pk, \mathsf{Encaps}^{(1)}_{pk}(1^n),\ \mathsf{Enc}'_k(m_0)\right) \arrow[r, dashed, leftrightarrow, "\text{(by ``transitivity'')}"] \arrow[d, leftrightarrow, "\text{(by security of } \Pi)"'] & \left(pk, \mathsf{Encaps}^{(1)}_{pk}(1^n),\ \mathsf{Enc}'_k(m_1)\right) \arrow[d, leftrightarrow, "\text{(by security of } \Pi)"] \\
\left(pk, \mathsf{Encaps}^{(1)}_{pk}(1^n),\ \mathsf{Enc}'_{k'}(m_0)\right) \arrow[r, leftrightarrow, "\text{(by security of } \Pi')"'] & \left(pk, \mathsf{Encaps}^{(1)}_{pk}(1^n),\ \mathsf{Enc}'_{k'}(m_1)\right)
\end{tikzcd}
$$

THEOREM 11.14: If Π is a CCA-secure KEM and Π’ is a CCA-secure private-key encryption scheme, then Π^{hy} is a CCA-secure public-key encryption scheme.

Proof works analogous

RSA KEM

- Recall construction for which we have shown IND-CPA security as PKE in the ROM (i.e., H is modeled as a random oracle)
  - $\mathrm{Enc}(m, pk) := (H(x) \oplus m, x^e \bmod N)$ for $m \in \{0,1\}^k$ and $x \leftarrow^{\$} \mathbb{Z}_N^*$
  - $\mathrm{Dec}((c_1,c_2), sk) := H(c_2^d \bmod N) \oplus c_1$
- Clearly, this PKE is not IND-CCA secure
  - Take challenge ciphertext (c^*_1,c^*_2) and send $(c^*_1 \oplus 0\ldots1, c^*_2)$ to Dec* oracle
- View the above scheme as a KEM combined with a eav secure private-key encryption scheme
- We just consider the KEM part of this scheme
  - $\mathrm{Encaps}_{pk}(1^n)$: Return (c, k) as $(x^e \bmod N, H(x))$ for $x \leftarrow^{\$} \mathbb{Z}_N^*$
  - $\mathrm{Decaps}_{sk}(c)$: Return $H(c^d \bmod N)$

THEOREM 11.38: If the RSA problem is hard relative to GenRSA and H is modeled as a random oracle, then the above is a CCA-secure KEM.

RSA KEM – Sketch of Security Proof

Proof idea (similar to last time):

- If the adversary does not query H(x), for the key k*=H(x) in the challenge ciphertext (c*,k*), the key k* is uniformly random
- To learn information about k*=H(x), adversary has to query H(x). We can embed an RSA challenge y as c* = y
- Challenge key is hidden information theoretically unless random oracle queried H on x s.t. $y = x^e \bmod N$
- If this happens, we have an adversary against the RSA assumption (thus we can rule out the adversary will query x to H and can only guess).

For CCA security we have to consitently simulate the Decaps* oracle for the adversary (and we do not know the secret key!)

RSA KEM – Sketch of Security Proof

Simulate the Decaps* oracle (without the secret key):

- We keep two lists L_H and L_{Dec}
  - We initally put (c*, k) for uniform k into L_{Dec}
  - We consitently simulate the random oracle and the Decaps* oracle

H(x'):
If $(x', k) \in L_H$ for k return k
Else $c' \leftarrow x'^e \bmod N$
 If $(c', k) \in L_{Dec}$ for k return k and $L_H := L_H \cup (x', k)$
 Else $k \leftarrow^{\$} \{0,1\}^n$, return k and $L_H := L_H \cup (x', k)$

For every hash query proactively compute the corresponding ciphertext in the forward direction

Decaps(c'):
If $(c', k) \in L_{Dec}$ for k return k
Else for each $(r', k) \in L_H$ check $r'^e = c' \bmod N$
 If so return k
 Else $k \leftarrow^{\$} \{0,1\}^n$, return k and $L_{Dec} := L_{Dec} \cup (c', k)$

For every decapsulation we check if we have computed the corresponding key

Otherwise we sample a new random key and update our key list

The simulation is perfect from the view of the adversary

KEM under DDH/CDH/gapDH

- Analogously to the RSA KEM we can define KEMs in the DL setting
- IND-CPA secure KEM
  - **Encaps:** on input a public key pk = (G, q, g, $y=g^x$, H) choose a uniform $r \in \mathbb{Z}_q$ and output the ciphertext g^r and the key $H(y^r)$.
  - **Decaps:** on input a private key sk = (G, q, g, x) and a ciphertext c, output the key $H(c^x)$.
  - Depending on the choice of H we can show
    - IND-CPA security under the DDH assumption if H is a "good" key-derivation function
    - IND-CPA security under the CDH assumption if H is modeled as a random oracle
- IND-CCA secure KEM
  - If we model H as a random oracle under the gapDH assumption (CDH holds even if we allow a DDH oracle)
  - Standardized and often used in practice as DHIES/ECIES

CCA (In-)Security of ElGamal

- We have shown IND-CPA security of ElGamal under DDH (in prime order groups)
- What about IND-CCA1 (i.e., non-adpative chosen ciphertext) security?
  - Can be shown under a tailored interactive complexity assumption
  - There are ElGamal variants, e.g., Cramer-Shoup “lite”, where IND-CCA1 security can be shown under DDH
- What about IND-CCA2 (i.e., adaptive chosen ciphertext) security?
  - Can not hold due to the homomorphic property!
  - How to construct an attacker?
    - Re-randomize the challenge ciphertext c* and send it to Dec* oracle

CCA Insecurity in Practice

2018

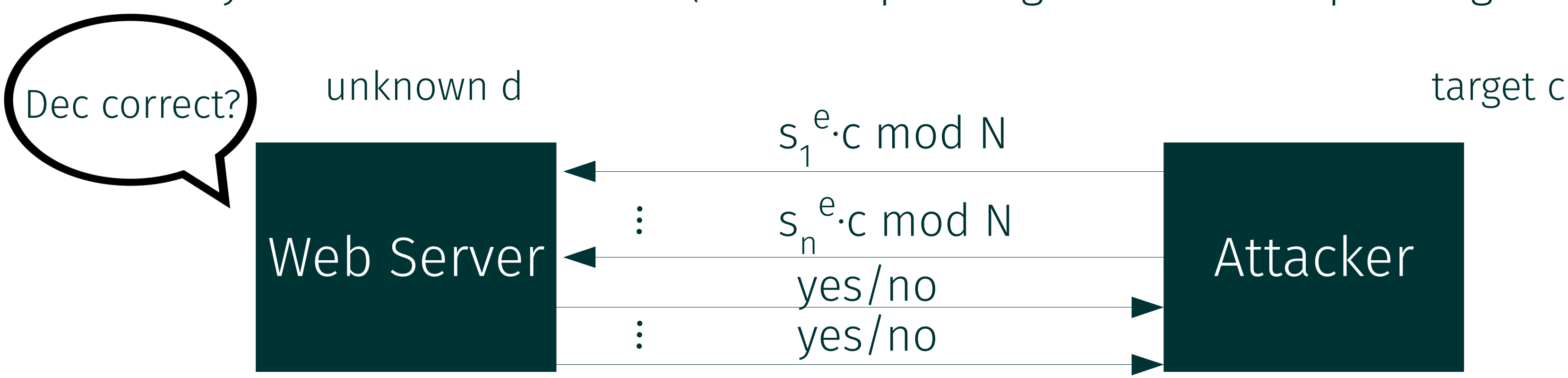
2017

2016

Bleichenbacher attack
on RSA PKCS#1 v1.5 encryption

Using some facts of RSA

- Having $c = m^e \bmod N$; sending $c' := s^e \cdot c \bmod N$ for random $s \leftarrow^{\$} \mathbb{Z}_N^*$ to Dec
- Recovering m: $m' = (s^e \cdot c)^d \bmod N = s \cdot m \bmod N$; $m := s^{-1} \cdot m' \bmod N$
- m is specifically padded (PKCS#1 v1.5 - 0x00||0x02||R||0x00||m) and the Dec oracle only needs to return a bit (whether padding is correct – a padding oracle)



Attacker can test if 16 MSBs of plaintext are 02

If i'th answer is yes, we know that $s_i \cdot m \bmod N$ is in a specific range (we know it starts with 0x0002)

Repeat this blinding step to further narrow down the range – until one candidate, i.e., m, is left

For N of 1024 bits the original attack requires ~ 10^6 tries (can be significantly reduced)

IND-CCA Secure Schemes

- We have seen several variants of “hybrid” PKE schemes that are CCA2 secure
  - RSA-KEM used in combination with a CCA-secure DEM (ROM)
  - DHIES/ECIES (ROM)
- We know that a CCA-secure KEM is sufficient to obtain a CCA-secure PKE
- There are various generic conversions for weakly secure PKEs
  - Fujisaki-Okamoto (FO): starts from OW-CPA secure PKE (ROM)
  - REACT, GEM: starts from one-way against plaintext checking attacks (OW-PCA) secure PKE (ROM)
  - Naor-Yung (“twin encryption”): IND-CPA secure PKE + non-interactive zero-knowledge proofs (w/o ROM)
- Any CPA secure identity-based encryption scheme + strongly secure one-time signatures (w/o ROM)
- Cramer-Shoup: From hash-proof systems (w/o ROM)