Privacy Homomorphism and Applications through Symmetric Key Encryption Algorithms

4th Febrary 2023

Devesh C Jinwala,
Professor (HAG), SVNIT, Surat and Adjunct Professor, IIT Jammu

Sardar Vallabhbhai National Institute of Technology Surat

1

Homomorphic Encryption [Peter et al]

- A privacy homomorphism (PH) is
  - an encryption transformation that allows direct computation on encrypted data.
  - Let Q and R denote two rings, and + and $\oplus$ denote addition operations on the rings.
  - Let K be the key space.
  - The encryption transformation is E: K X Q $\rightarrow$ R and the corresponding decryption transformation is D: K X R $\rightarrow$ Q.
  - Given a, b $\in$ Q and k_1, $k_2 \in$ K, we term

$$a + b = D_k(E_k(a) \oplus E_k(b))`$$

2

An Illustration

- Additive Privacy homomorphism……

Let $x_1 = 20$ and $x_2 = 22$, to compute $x_1+x_2 = 42$

Use an encryption scheme, for example $E(x) = e^x$

3

An Illustration

- Additive Privacy homomorphism……

Let $x_1 = 20$ and $x_2 = 22$, to compute $x_1+x_2 = 42$

Use an encryption scheme, for example $E(x) = e^x$

Server stores $E(x_1) = e^{20}$ and $E(x_2) = e^{22}$

4

An Illustration

- Additive Privacy homomorphism……

Let $x_1 = 20$ and $x_2 = 22$, to compute $x_1+x_2 = 42$

Use an encryption scheme, for example $E(x) = e^x$

Server stores $E(x_1) = e^{20}$ and $E(x_2) = e^{22}$

Compute using encrypted data
$y = E(x_1)\ E(x_2) = e^{20}.e^{22} = e^{42}$

An Illustration

- Additive Privacy homomorphism……

Let $x_1 = 20$ and $x_2 = 22$, to compute $x_1+x_2 = 42$

Use an encryption scheme, for example $E(x) = e^x$

Server stores $E(x_1) = e^{20}$ and $E(x_2) = e^{22}$

Compute using encrypted data
$y = E(x_1)\ E(x_2) = e^{20}.e^{22} = e^{42}$

Decrypt $z = D(y) = \ln(y)$
$z = D(y)\ \ln(e^{42}) = 42$

Definition

- *Homomorphic encryption is a form of encryption that allows for some computations to be performed on the ciphertext without decrypting the ciphertext.*
- *The result of the operations is returned as an encrypted result, which when decrypted is the same as if some operation was performed on the plaintext.*
  - *Useful in implementation of secure voting systems and in cloud computing.*

7

Homomorphic Encryption Types

- Partially Homomorphic Encryption
- Fully Homomorphic Encryption

Partially homomorphic cryptosystems are those that allow for either addition OR multiplication operation ONLY to be performed on the ciphertext, but not both.

A cryptosystem is considered fully homomorphic if it exhibits both additive and multiplicative homomorphism.

8

Algorithms for Secure Data Aggregation

9

SDA using HE…

- Secure Data Aggregation using Homomorphic Encryption
  - Three broad approaches…..consisting of algorithms
    - based on polynomial rings using SKC
      - Domingo-Ferrer, Castelluccia, Domingo-Ferrer + Castelluccia
    - based on one-time pads
      - CMT 2005 – (in Mobiquitous 2005)
    - based on PKC
      - Okamoto Uchiyama, Goldwasser-Micali, Benaloh, Elgamal, RSA, Paillier
      - the ECC versions – with the focus on improving the overhead

10

SDA using HE…

- Secure Data Aggregation using Homomorphic Encryption
  - Three broad approaches…..consisting of algorithms
    - based on polynomial rings using SKC
      - Domingo-Ferrer, Castelluccia, Domingo-Ferrer + Castelluccia
    - based on one-time pads
      - CMT 2005 – (in Mobiquitous 2005)
    - based on PKC
      - Okamoto Uchiyama, Goldwasser-Micali, Benaloh, Elgamal, RSA, Paillier
      - the ECC versions – with the focus on improving the overhead

11

Symmetric Key Homomorphic Algorithms

- Castelluccia's Algorithm
- Domingo-Ferrer's Algorithm
- Stefeen Peter's Algorithm

12

Symmetric Key Homomorphic Algorithms

- Castelluccia's Algorithm
- Domingo-Ferrer's Algorithm
- Stefeen Peter's Algorithm

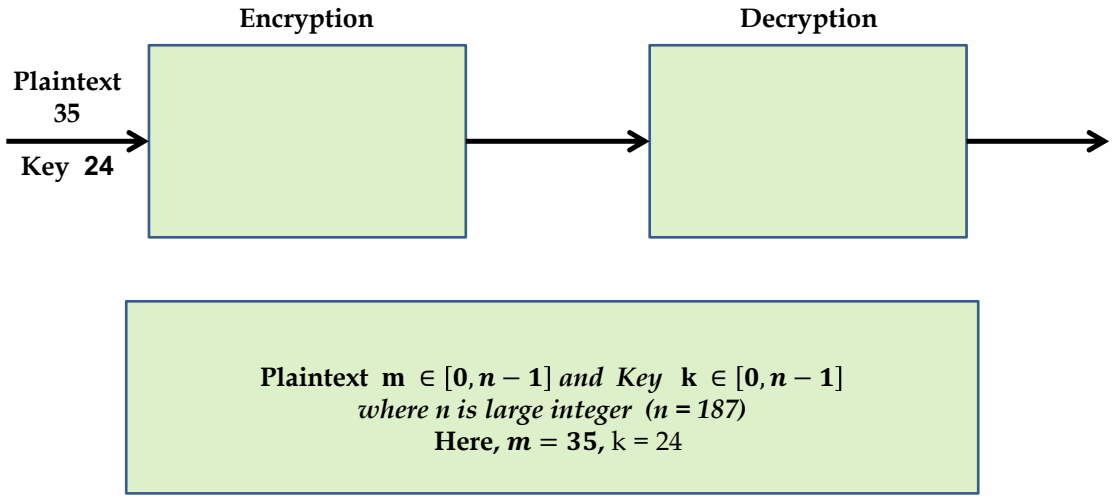
13

Castelluccia's Algorithm

Algorithm Casstelluccia ()
Parameters: Select large integer M
Encryption: Message $m \in [0, M-1]$,
Randomly generated key stream $k \in [0, M-1]$
$c = (m + k) \bmod M$
Decryption: $m = (c\text{-}k) \bmod M$
Aggregation: $c_{12} = (c_1 + c_2) mod M$

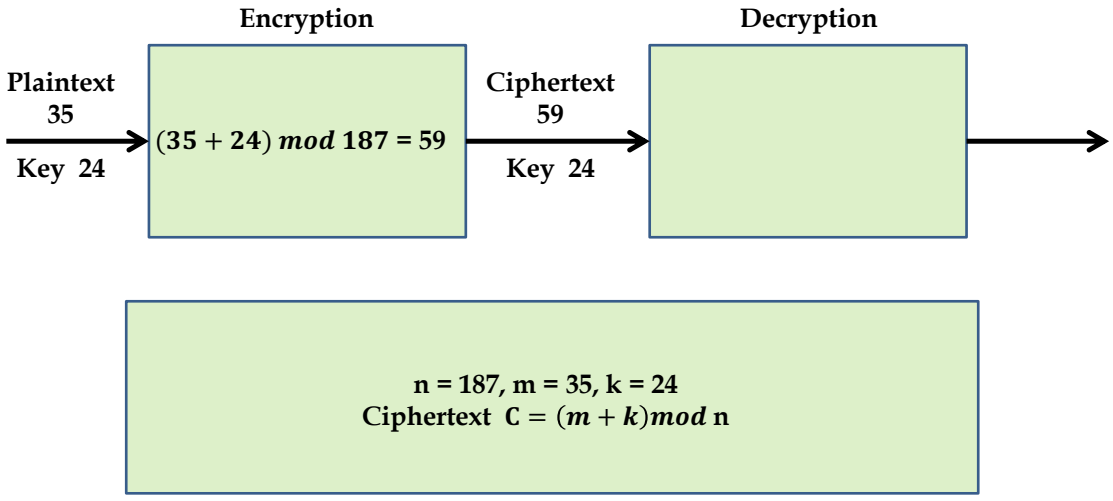
14

Castelluccia's Algorithm



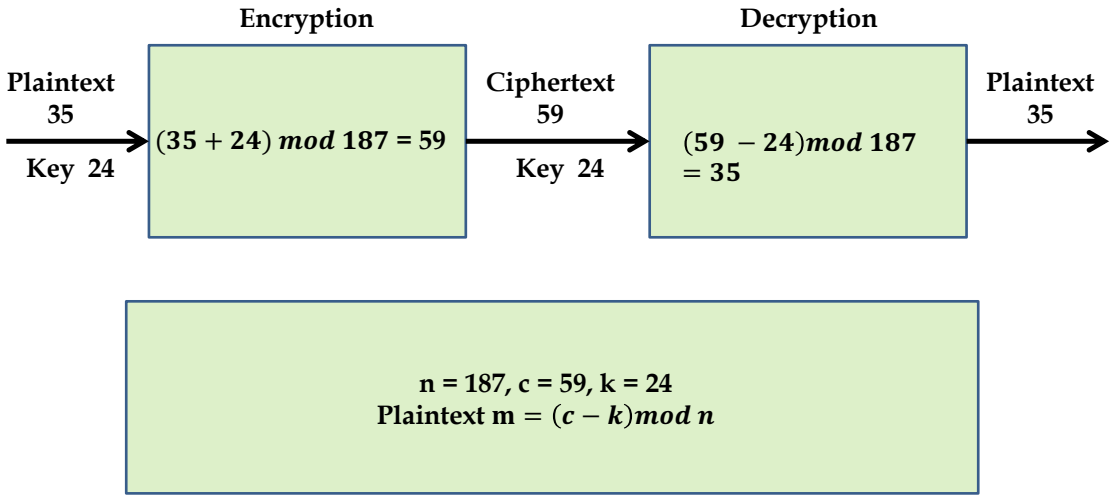
15

Castelluccia's Algorithm



16

Castelluccia's Algorithm



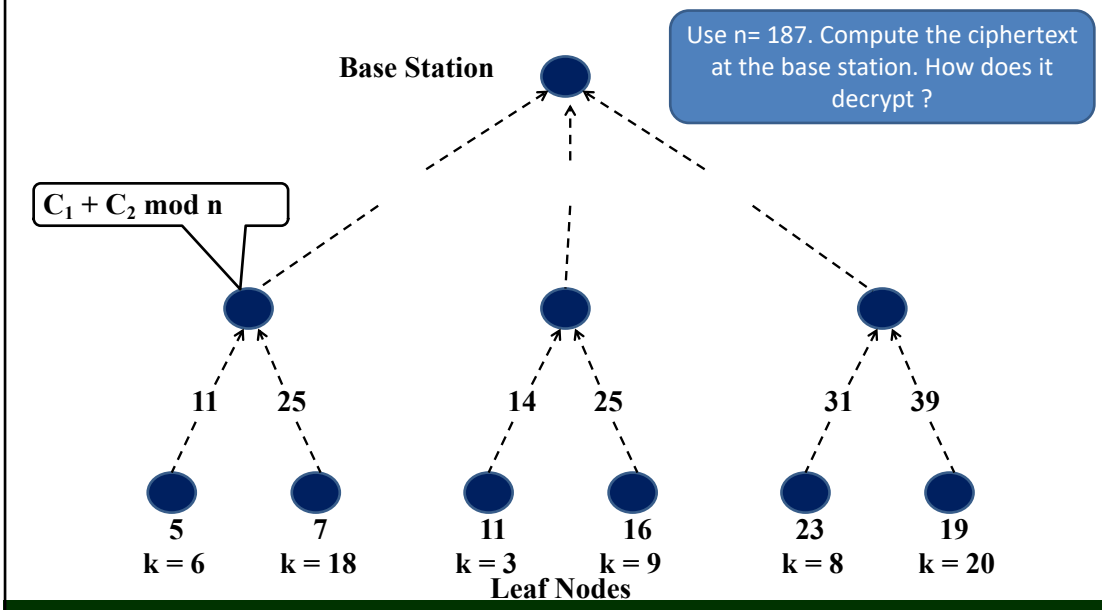
17

Castelluccia's Algorithm – Homomorphic Property

$$(C_1 + C_2) \bmod n = E(m_1 + m_2) \bmod n$$

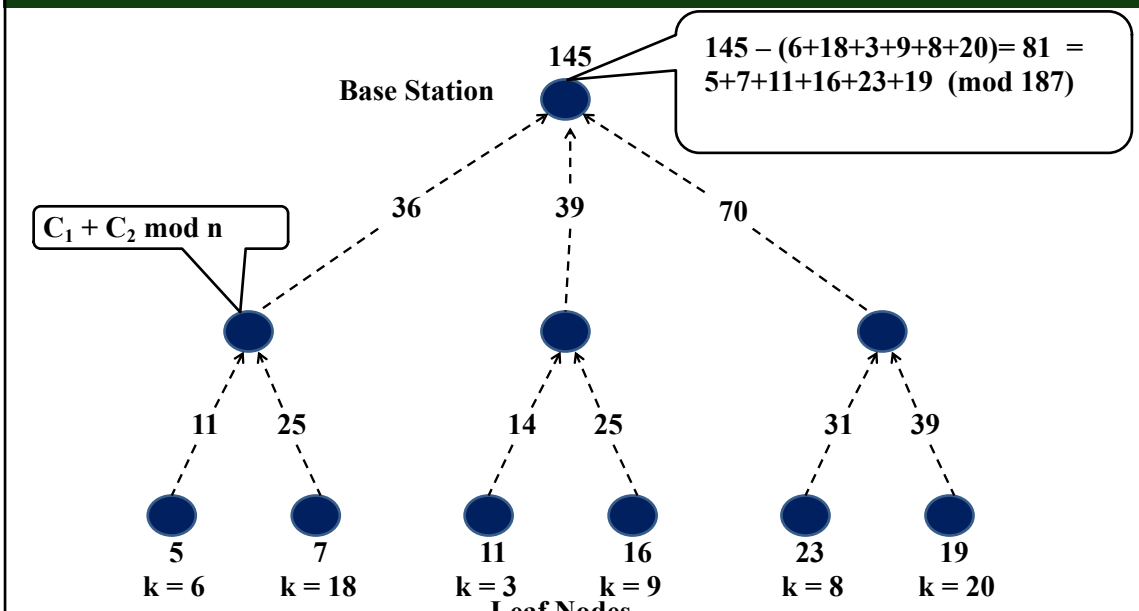
18

Class Tutorial #1 - Castelluccia's Algorithm



19

Class Tutorial #1 - Castelluccia's Algorithm



20

Symmetric Key Homomorphic Algorithms

- Castelluccia's Algorithm
- Domingo-Ferrer's Algorithm
- Stefeen Peter's Algorithm

Domingo-Ferrer's Algorithm

Algorithm Domingo-Ferrer ()

Parameters:

Public Key: integer d ≥ 2, *large integer* M

Secret Key: g that divides M; r so that r^{-1} *exists in* Z_M

Encryption: Split m into d parts $m_1..m_d$ *such that*

$\sum_{i=1}^{d}(m_i) \bmod g = m$

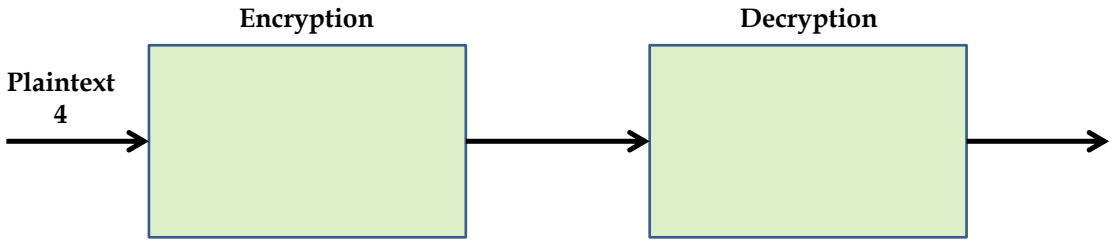
$C = [c_1, ..., c_d] = [m_1 r^1 mod M, m_2 r^2 mod M, .., m_d r^d mod M]$

Decryption: $m = (c_1 r^{-1} + c_2 r^{-2} + ... + c_d r^{-d}) mod g$

Aggregation: Scalar addition modulo M: $C_{12} = C_1 + C_2 =$

$[(c_1 1 + c_2 1) mod M, ..., (c_1 d + c_2 d) mod M]$

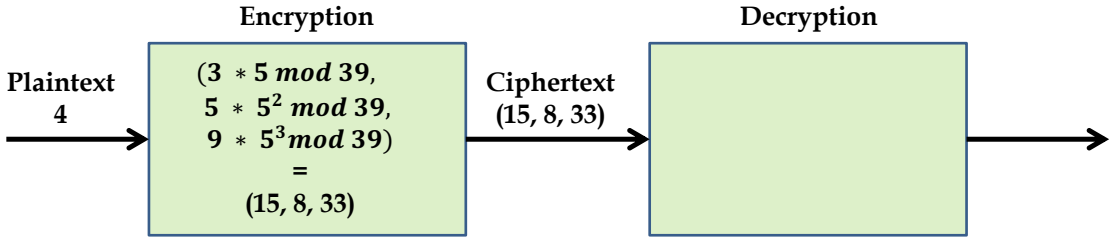
Domingo-Ferrer's Algorithm



Public parameters: d > 2 and large integer n
Here, d = 3, n = 39
Secret parameters: $n' > 1 \mid n, r \in \mathbb{Z}_n^*$
$r = 5, r^{-1} = 8, n' = 13$
Plaintext m = 4

23

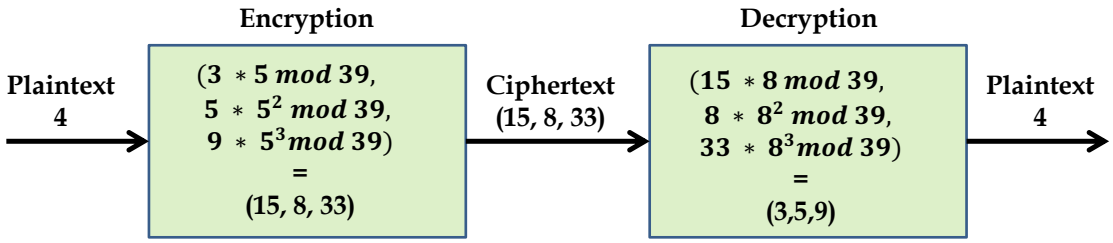
Domingo-Ferrer's Algorithm



Split $m \in \mathbb{Z}_n$, into $m_1, m_2, \ldots, m_d$
Where $m = \Sigma_{j=1}^{d} mj \bmod n'$, $m_j \epsilon \mathbb{Z}_n$
Here m = (3, 5, 9) mod 13
Ciphertext $C = (m_1 r \bmod n, m_2 r^2 \bmod n, \ldots, mdrd \bmod n)$

24

Domingo-Ferrer's Algorithm



Here C = (c_1, c_2, c_3) = (15, 8, 33)

Plaintext $m = (c_1 r^{-1} \ mod\ n, c_2 r^{-2} mod\ n, \ldots, cdr^{-d} mod\ n)$

$m = \Sigma_{j=1}^{d} mj \bmod n'$

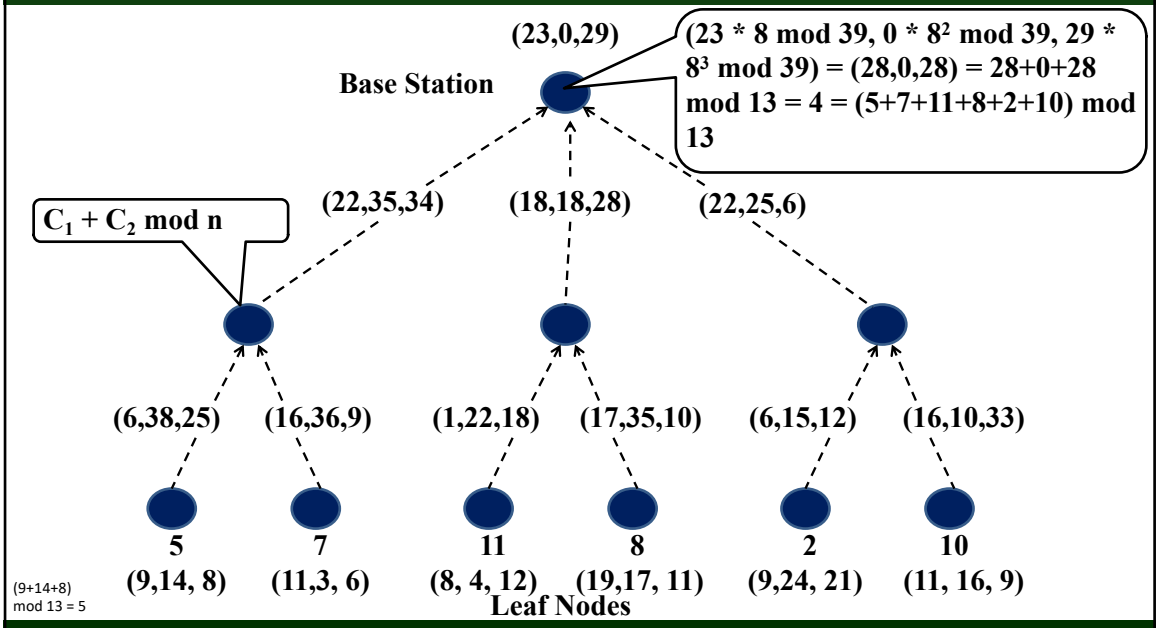
25

Domingo-Ferrer's Algorithm – Homomorphic Property

$$(C_1 + C_2) \bmod n = E(m_1 + m_2) \bmod n$$

26

Domingo-Ferrer's Algorithm – Example



27

Symmetric Key Homomorphic Algorithms

- Castelluccia's Algorithm
- Domingo-Ferrer's Algorithm
- Stefeen Peter's Algorithm

28

Algorithm Casstelluccia+Domingo-Ferrer ()

Parameters:

Public Key: integer d ≥ 2, *large integer* M

Secret Key: g that divides M; r so that r^{-1} *exists in* Z_M

Encryption: Randomly generated key stream $k \in [0, M-1]$

$e1 = (k + m) \bmod M$

Split e1 into d parts $m_1..m_d$ *such that*

$\sum_{i=1}^{d} (m_i) \bmod g = m$

$C = [c_1, ..., c_d] = [m_1 r^1 mod M, m_2 r^2 mod M, .., m_d r^d mod M]$

Aggregation: Scalar addition modulo M: $C_{12} = C_1 + C_2 =$

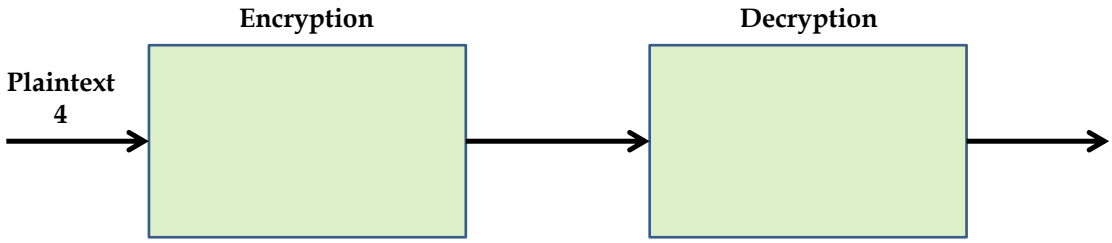
$[(c_1 1 + c_2 1) mod M, ..., (c_1 d + c_2 d) mod M]$

Decryption: $d_1 = (c_1 r^{-1} + c_2 r^{-2} + ... + c_d r^{-d}) mod g$

$m = (d_1 - k) \bmod M$

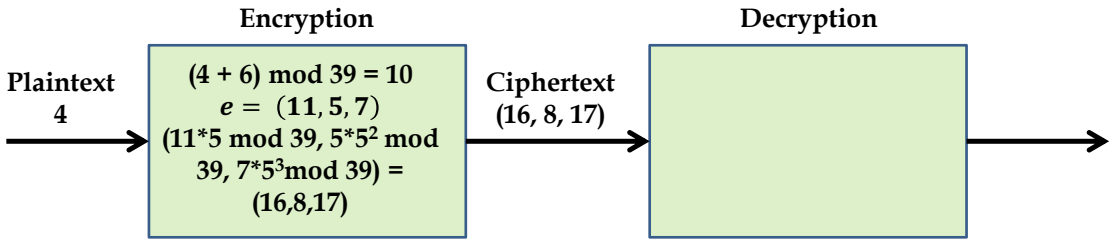
where k is the sum of aggregated key streams

Stefeen Peter's Algorithm



Public parameters: d > 2 and large integer n,
k ∈ [0, $n-1$] Here, d = 3, n = 39, key k = 6
Secret parameters: n′ > 1 | n′ must divide n; n, r ∈ $\mathbb{Z}_n^*$
r = 5, r^{-1} = 8, n′ = 13
Plaintext m = 4

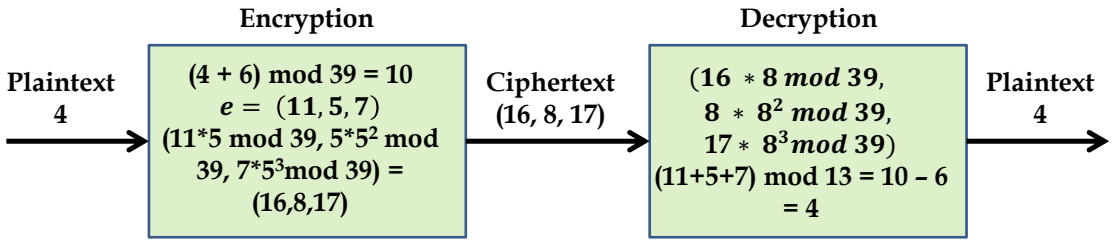
Stephen Peter's Algorithm



r = 5, r-1 = 8, n′ = 13 ,
e = (m + k) mod n
Split e ∈ $\mathbb{Z}_n$, into $e_1, e_2, \ldots, e_d$ Where e = $\Sigma_{j=1}^{d}$ (ej mod n′)
i.e. (11+5+7) mod 13 = 23 mod 13 = 10
....So e = (11,5,7) mod 13
Ciphertext C = $(m_1 r \bmod n, m_2 r^2 \bmod n, \ldots, mdrd \bmod n)$

31

Stefeen Peter's Algorithm



Here C = (c_1, c_2, c_3) = (16,8,17)
$d_1 = (c_1 r^{-1} \bmod n + c_2 r^{-2} \bmod n + \cdots + cdr^{-d} \bmod n)$
m = $\Sigma_{j=1}^{d}$ d_j (mod n′) – k

32

Stefeen Peter's Algorithm – Homomorphic Property

$$(C_1 + C_2) \bmod n = E(m_1 + m_2) \bmod n$$

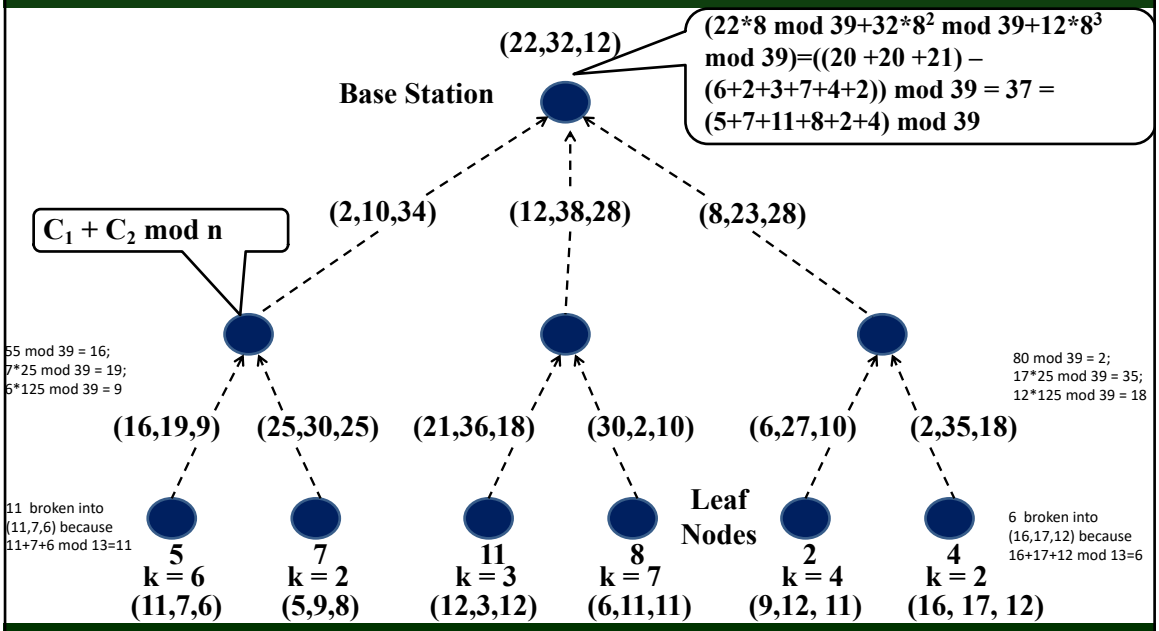
Another Example

m = 2, n =28, k=1, r=3, n'=7, r^{-1}=19 mod 28
C=(2+1) mod 28 = 3
Now, Partition
(4,6) i.e. 10 mod n' = 10 mod 7 =3
Encryption:
=[(4*3) mod 28 , (6*3^2) mod 28]
=(12, 26)

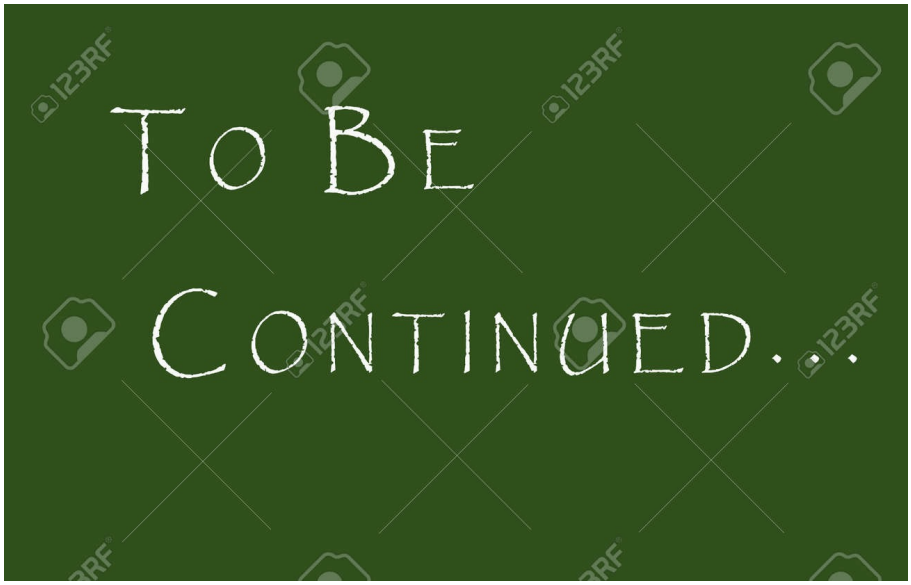
m = 1, n=28, k=1, r=3, n'=7, r^{-1}=19
C=(1+1) mod 28 = 2
Now, Partition
(4,5) i.e. 9 mod n' =9 mod 7 =2
Encryption:
=[(4*3) mod 28 , (5*3*3) mod 28]
=(12,17)

Aggregation:
(12,26) + (12,17) = (24,43)
Decryption:
(24,43)
= [(24*19)mod28 + (43*19*19)mod28]mod7 =
=5
(5-2)mod28 = 3

Stefeen Peter's Algorithm – Example



35



36