UNIVERSITÄT MANNHEIM

Cryptographic Tools For Privacy-Preserving Data Processing

Frederik Armknecht

Group for Theoretical Computer Science and IT-Security

December 16, 2014

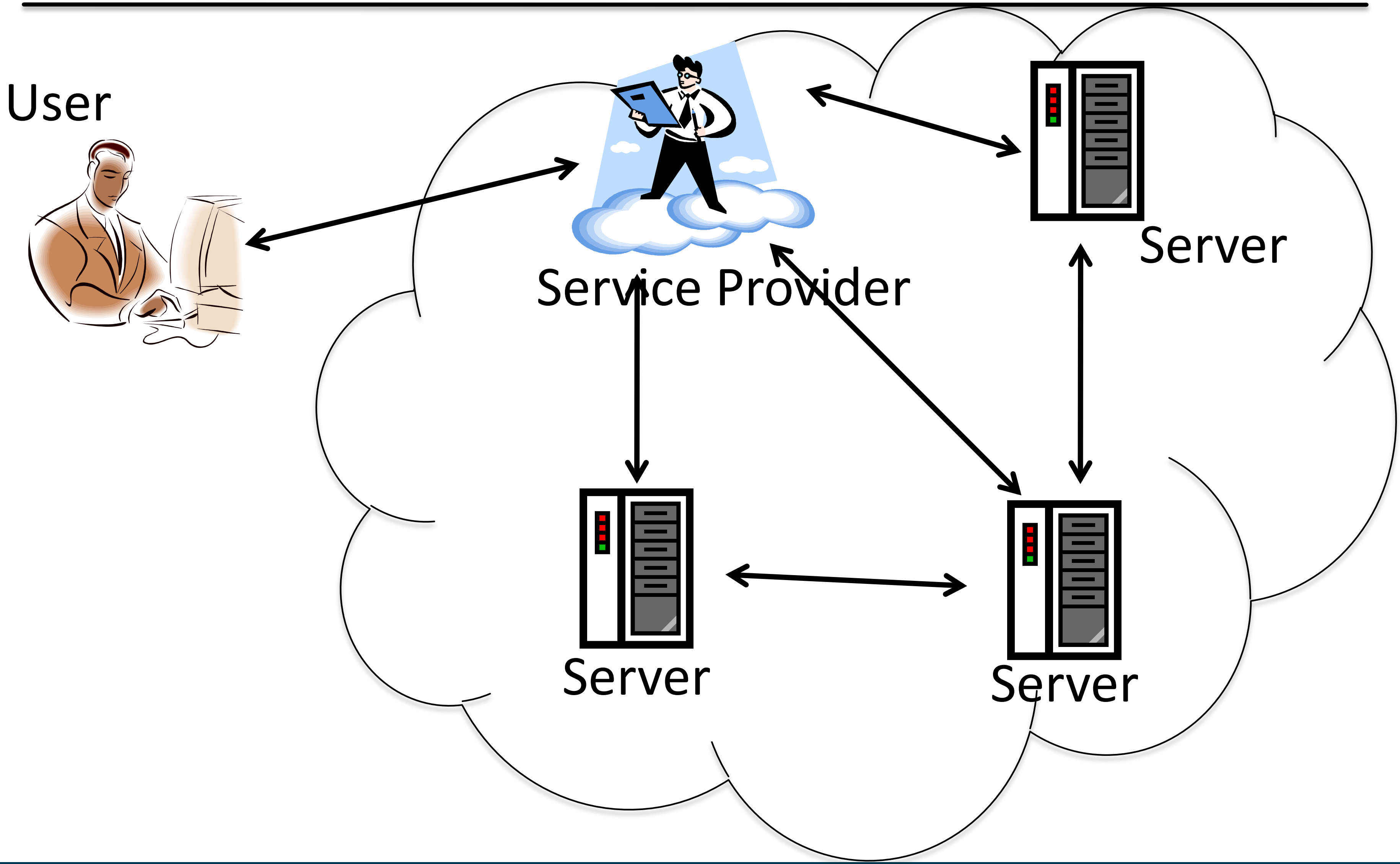
Paris, France

Overview

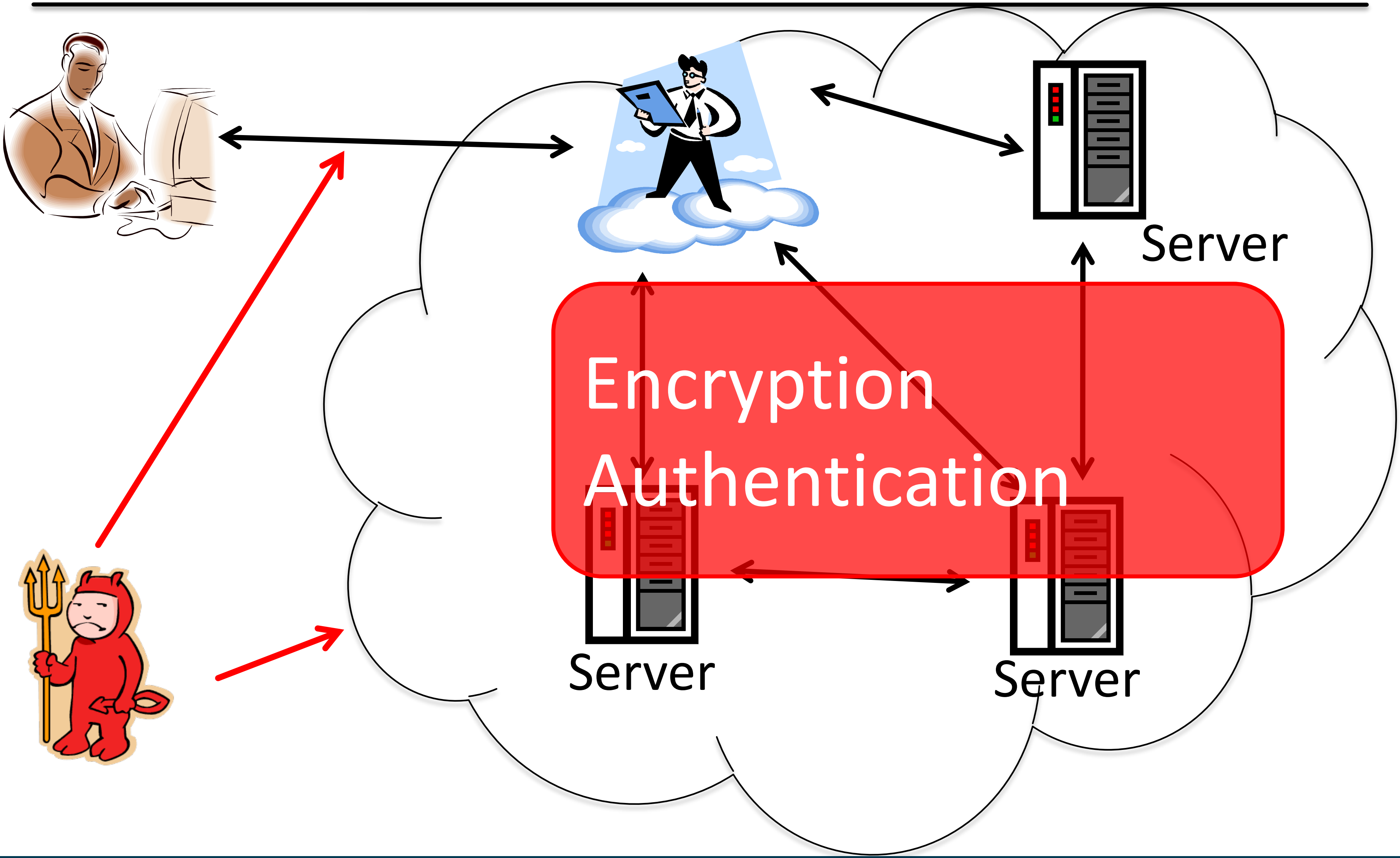
- Introduction
- Group Homomorphic Encryption
- Somewhat Homomorphic Encryption
- Adapted Homomorphic Encryption
- Conclusion

Introduction

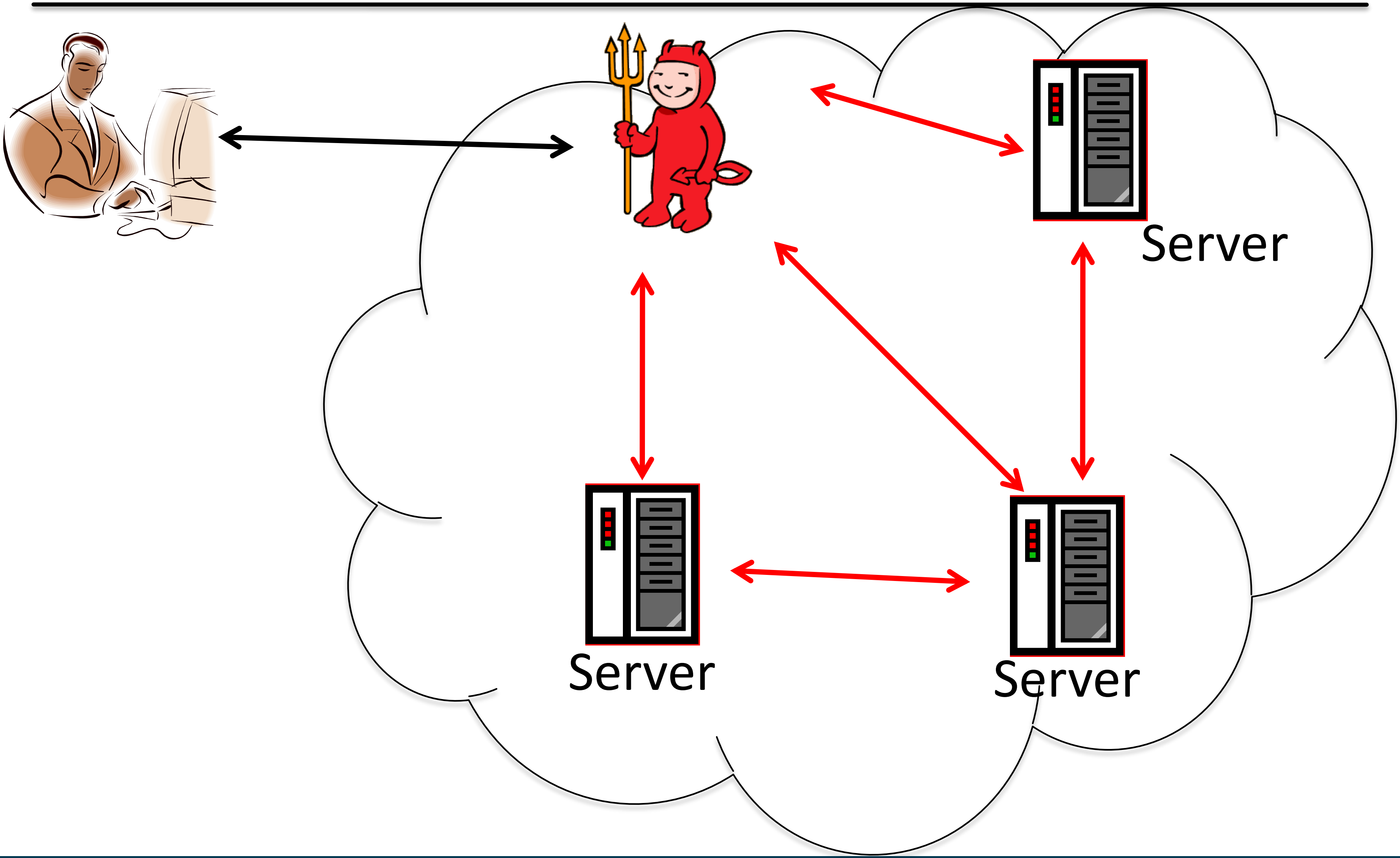
Cloud Computing



Outsider Attacker



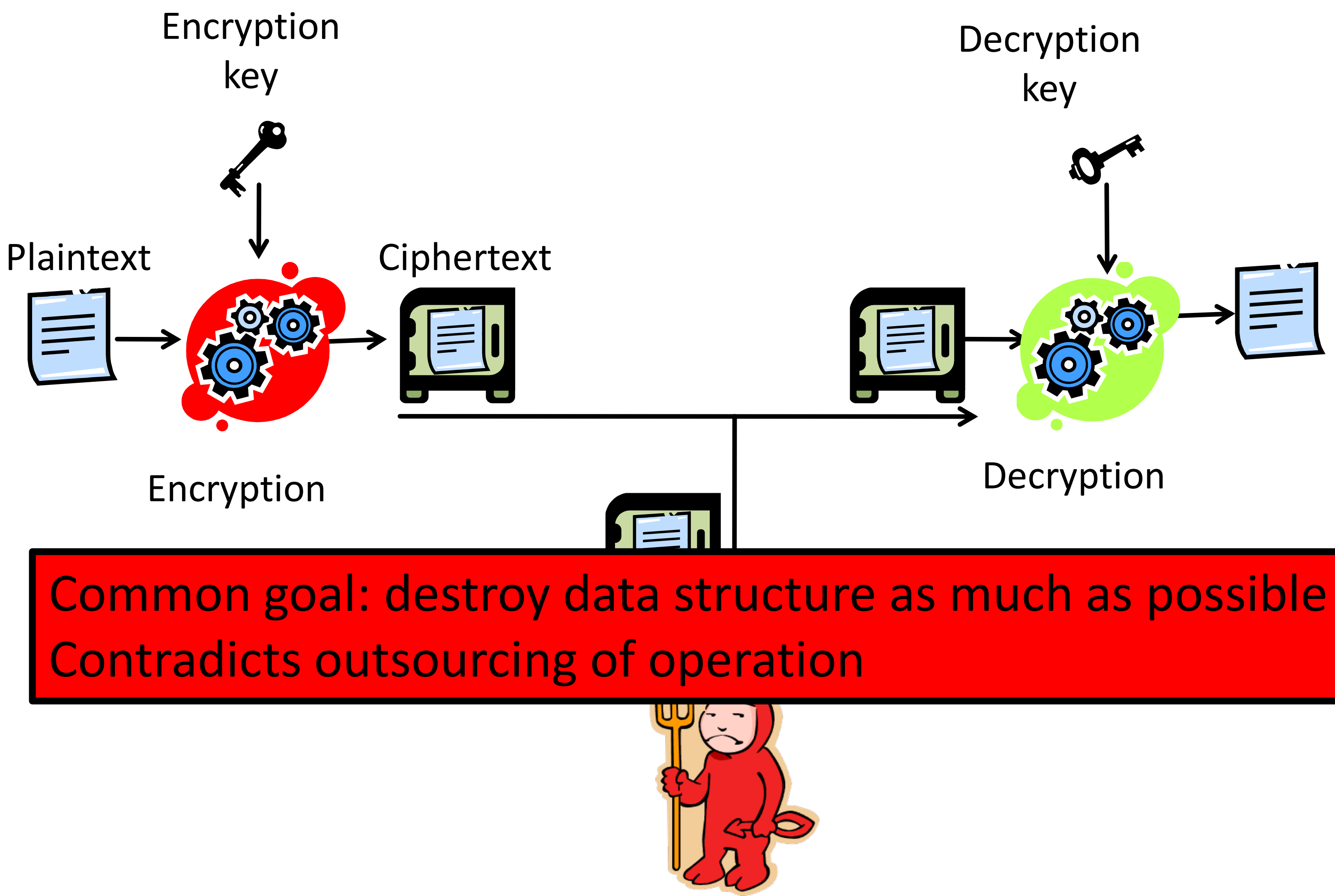
Insider Attacker?



Possible Approaches

- **Interactive**
  - User and provider run an interactive protocol
  - Cryptographic techniques: multi-party computation, secure function evaluation
  - Advantage: can be quite efficient, good control over who learns what
  - Disadvantage: additional involvement of the user
- **Non-interactive**
  - Data needs to be available to the service provider but at the same time intrinsically protected
  - Solution: encryption

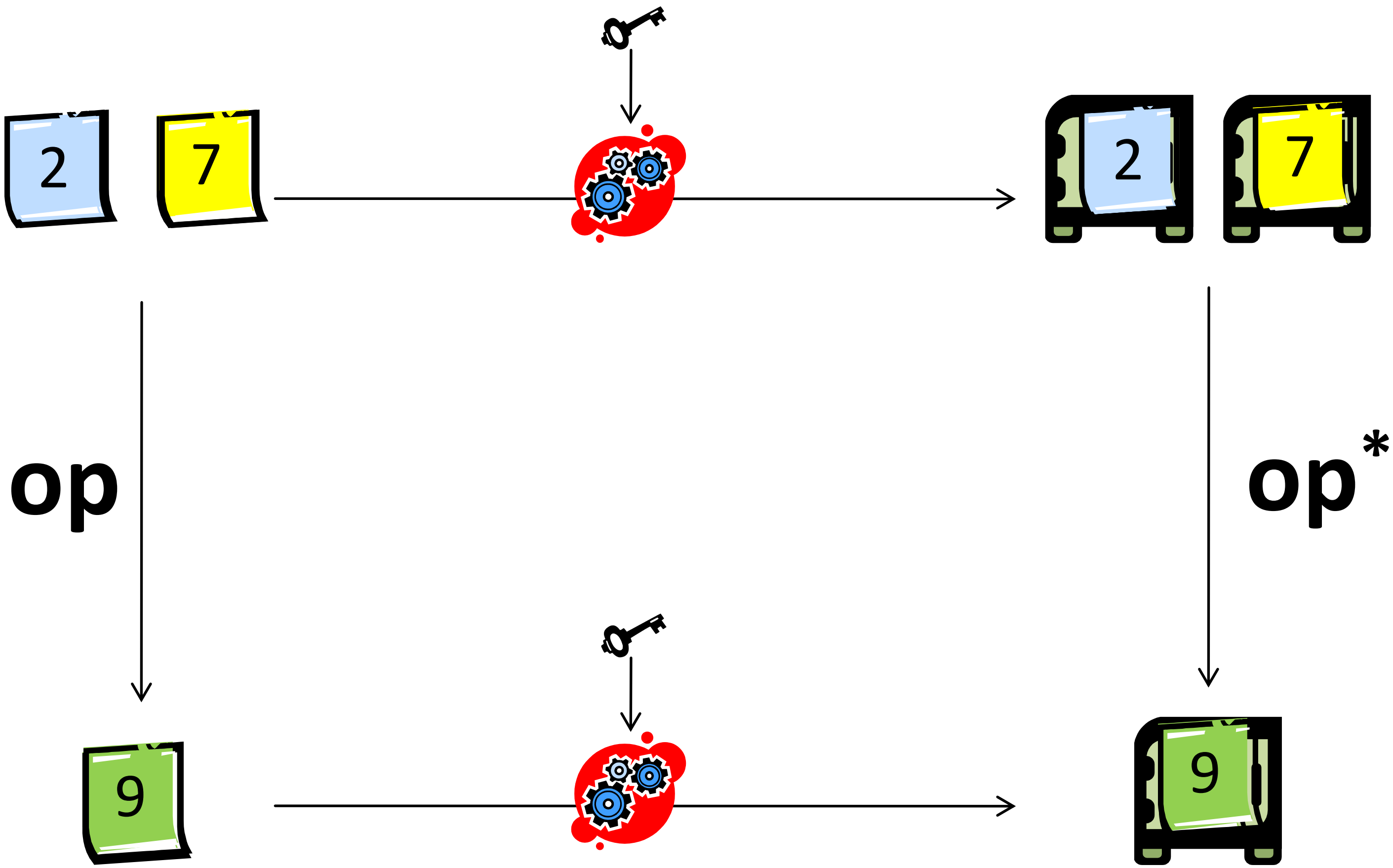
Encryption



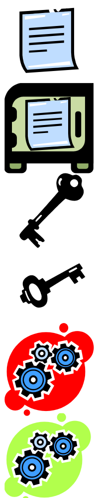
Common goal: destroy data structure as much as possible
Contradicts outsourcing of operation

Homomorphic Encryption

Encryption that allows for meaningful operations on encrypted data



Example: RSA (1978)



Parameters: $N=p \cdot q$ with p,q large primes (approx. 1000 bits)

Plaintext space: $\mathbb{Z}_N$ (={0,…,N-1} modulo N)

Ciphertext: $\mathbb{Z}_N$ (={0,…,N-1} modulo N)

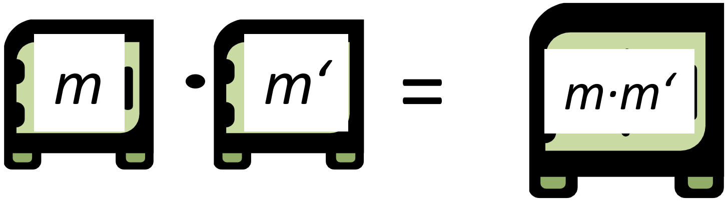
Encryption Key: $e \in \mathbb{Z}_N$ with gcd(e, (p-1)(q-1))=1

Decryption key: $d \in \mathbb{Z}_N$ with $e \cdot d$ mod $\left((p-1)\cdot(q-1)\right) = 1$

Encryption of *m*: $c := m^e \bmod N$

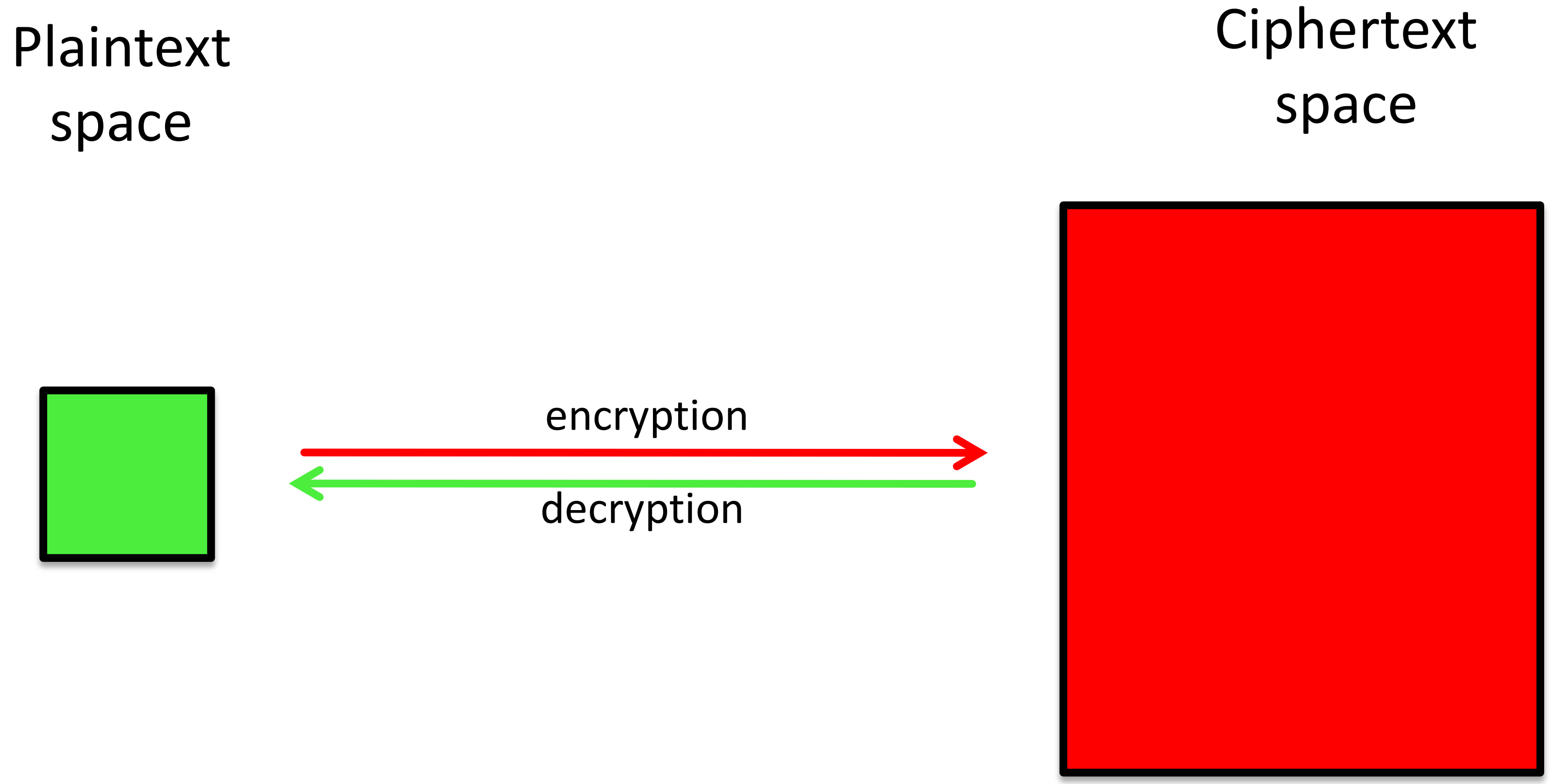
Decryption of *c*: $c^d \bmod N = m$

Homomorphism: $$m^e \cdot m'^e = (m \cdot m')^e$$



Group Homomorphic Encryption

Classical Encryption Scheme



Reminder: Group

- A **group** (in mathematical sense) is a set G together with a binary operation $\circ: G\times G \rightarrow G$ such that

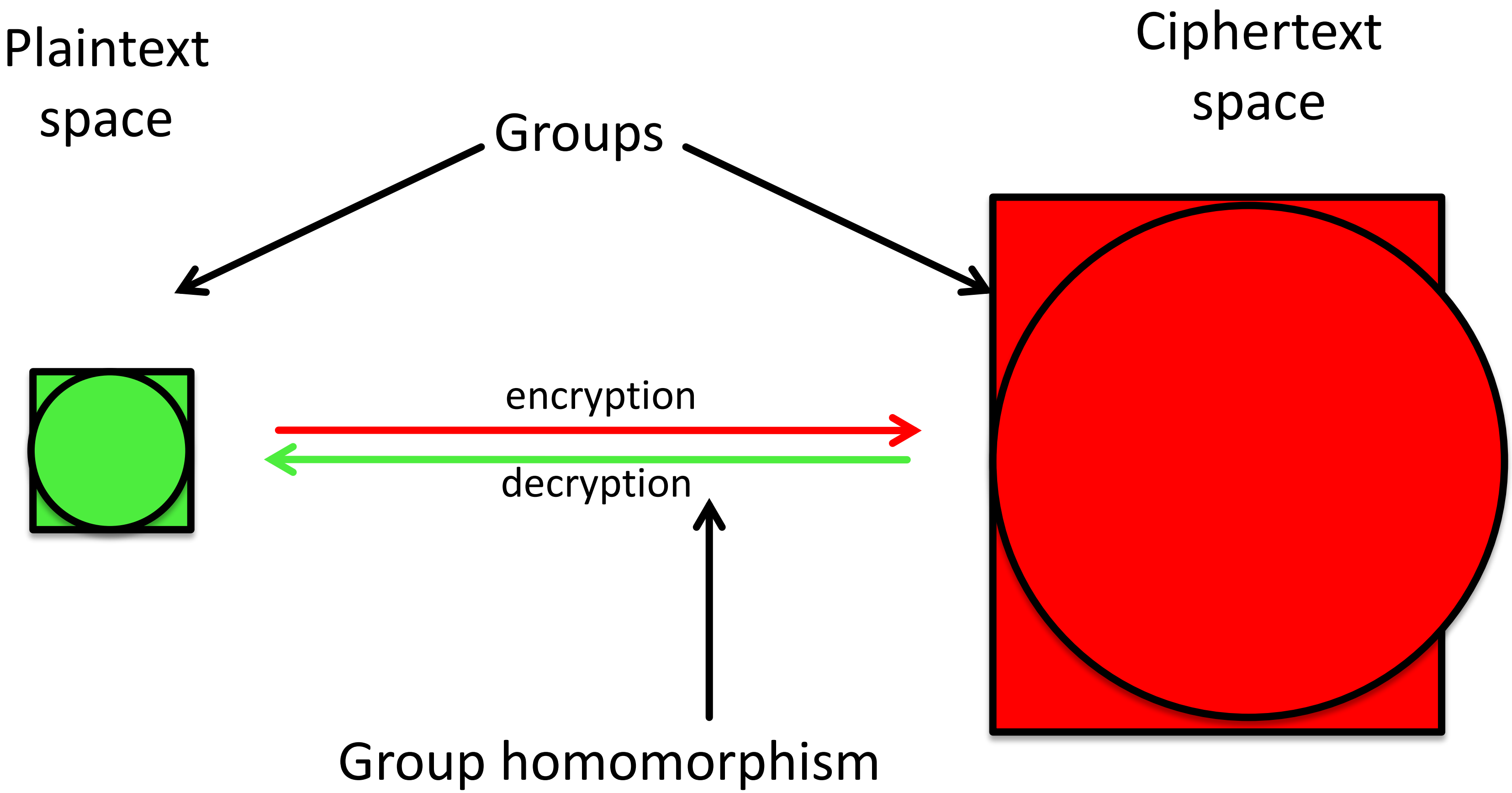
Group Axiom	Property
Closure	For all $g,g' \in G$: $g \circ g' \in G$
Associativity	For all $g,g',g'' \in G$: $(g \circ g') \circ g'' = g \circ (g' \circ g'')$
Neutral element	$e \circ g = g \circ e = g$
Inverse element	For all $g \in G$ exists $g' \in G$ such that $g \circ g' = g' \circ g = e$

Example: Rational numbers without zero

Neutral element: 1

Inverse element: x^{-1}

Considered Hom. Encr. Schemes



Overview of some homomorphic encryption schemes

Scheme	Plaintext Space	Security related to
RSA; 1978	Integers modulo N=p*q	Factorization
Goldwasser, Micali; 1984	1 Bit	Quadratic residues mod N
Benaloh; 1985	Integers modulo R s.t. …	R^{th} residues mod N
ElGamal; 1985	Cyclic group *G*	Decision Diffie-Hellman in *G*
Paillier; 1999	Integers modulo N	N^{th} residues mod N^2
Daamgard, Jurik; 2001	Integers modulo N^s	N^{th} residues mod N^{s+1}

- Different approaches
- For some proofs of security are known, for other not
- Some are much better understood than others
- Question: Unified view on security and design of homomorphic schemes

Security of Some Existing Schemes

Scheme	IND-CPA secure if the following problem is hard	IND-CCA1 secure if the following problem is hard
ElGamal; 1985	Decision Diffie-Hellman; 1998	[Lipmaa; 2010]
Paillier; 1999	N^{th} residues mod N^2; 1999	??
Daamgard, Jurik; 2001	N^{th} residues mod N^{s+1}; 2001	??
Boneh et al.; 2005	Decision Diffie-Hellman; 2005	??

Our Result: Abstraction

A., Katzenbeisser, Peters. Designs, Codes and Cryptography 2013.

Scheme	IND-CPA secure if the following problem is hard	IND-CCA1 secure if the following problem is hard
Abstract scheme	Abstract problem: SMP (subgroup membership problem)	Abstract problem: SOAP (splitting oracle assisted SMP)

Application: Easy Confirmation of Known Results

A., Katzenbeisser, Peters. Designs, Codes and Cryptography 2013.

Scheme	IND-CPA secure if the following problem is hard	IND-CCA1 secure if the following problem is hard
ElGamal; 1985	Decision Diffie-Hellman; 1998	[Lipmaa; 2010]
Paillier; 1999	N^{th} residues mod N^2; 1999	??
Daamgard, Jurik; 2001	N^{th} residues mod N^{s+1}; 2001	??
Boneh et al.; 2005	Decision Diffie-Hellman; 2005	??

Application: Missing Characterizations

A., Katzenbeisser, Peters. Designs, Codes and Cryptography 2013.

Scheme	IND-CPA secure if the following problem is hard	IND-CCA1 secure if the following problem is hard
ElGamal; 1985	Decision Diffie-Hellman; 1998	[Lipmaa; 2010]
Paillier; 1999	N^{th} residues mod N^2; 1999	✓
Daamgard, Jurik; 2001	N^{th} residues mod N^{s+1}; 2001	✓
Boneh et al.; 2005	Decision Diffie-Hellman; 2005	✓

Application: New Schemes

A., Katzenbeisser, Peters. Designs, Codes and Cryptography 2013.

Scheme	IND-CPA secure if the following problem is hard	IND-CCA1 secure if the following problem is hard
ElGamal; 1985	Decision Diffie-Hellman; 1998	[Lipmaa; 2010]
Paillier; 1999	N^{th} residues mod N^2; 1999	✓
Daamgard, Jurik; 2001	N^{th} residues mod N^{s+1}; 2001	✓
Boneh et al.; 2005	Decision Diffie-Hellman; 2005	✓
Scheme 1	K-linear Problem	New Problem
Scheme 2	Gonzales Nieto et al.; 2005	New Problem

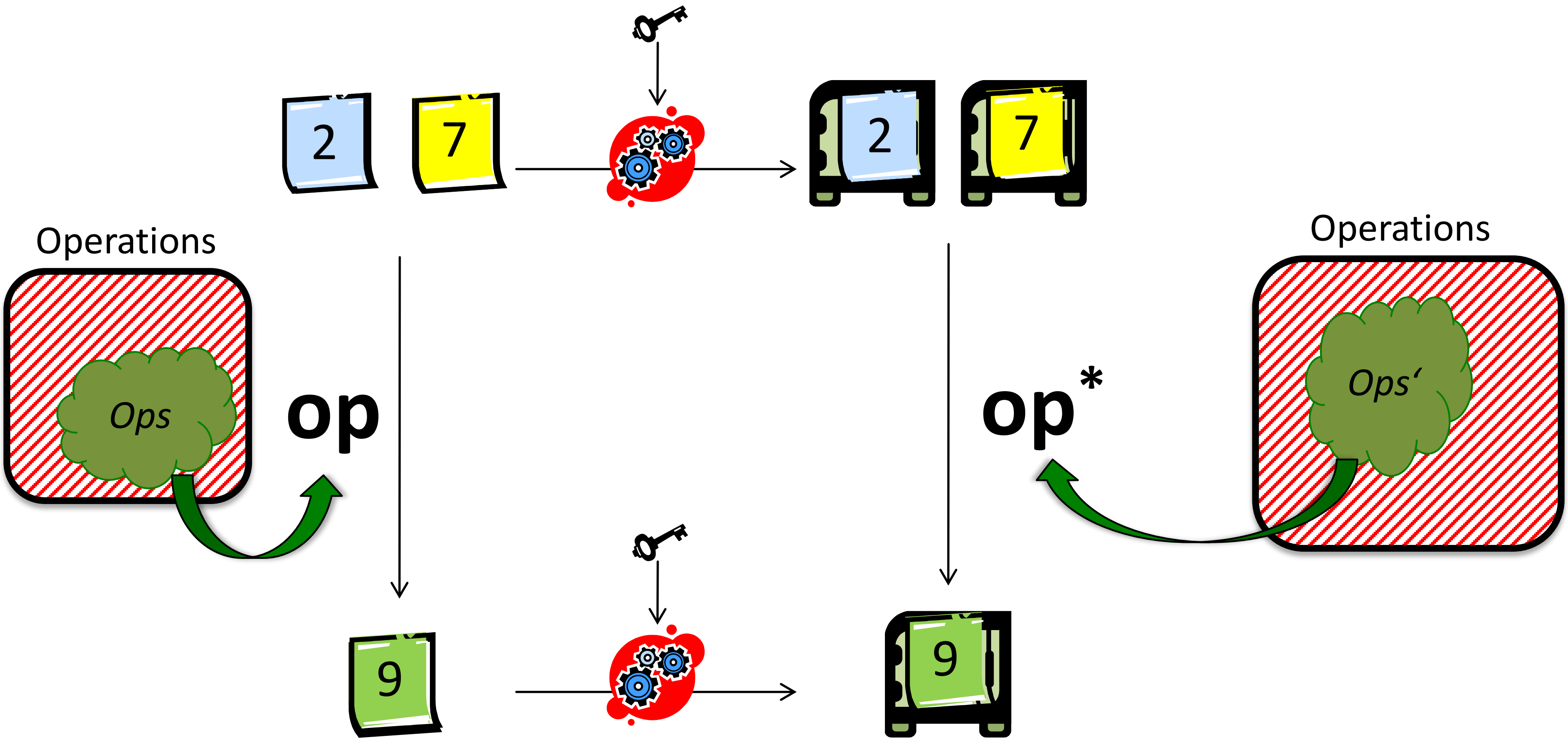
Summary

- **Situation for group homomorphic encryption schemes very well understood**
- **Open questions:**
  - What about symmetric key schemes?
  - What about schemes that support more operations?

Somewhat Homomorphic Encryption

Somewhat Homomorphic Encryption

Generalization: An encryption scheme is homomorphic wrt a set of operations *Ops* if there exists a set Ops* such that …



Example

A., Augot, Perret, Sadeghi. Cryptography and Coding 2011.

- **Generic construction for homomorphic schemes based on certain error-correcting codes**
- **Advantages**
  - Allows for unlimited additions and fixed (but arbitrary) number of multiplications
  - Many instantiations possible, e.g., Reed-Solomon codes, Reed-Muller codes
  - Simple operations
  - Decryption is very efficient
- **Disadvantages**
  - Number of encryptions needs to be limited
  - Length of ciphertexts

Concrete Implementation

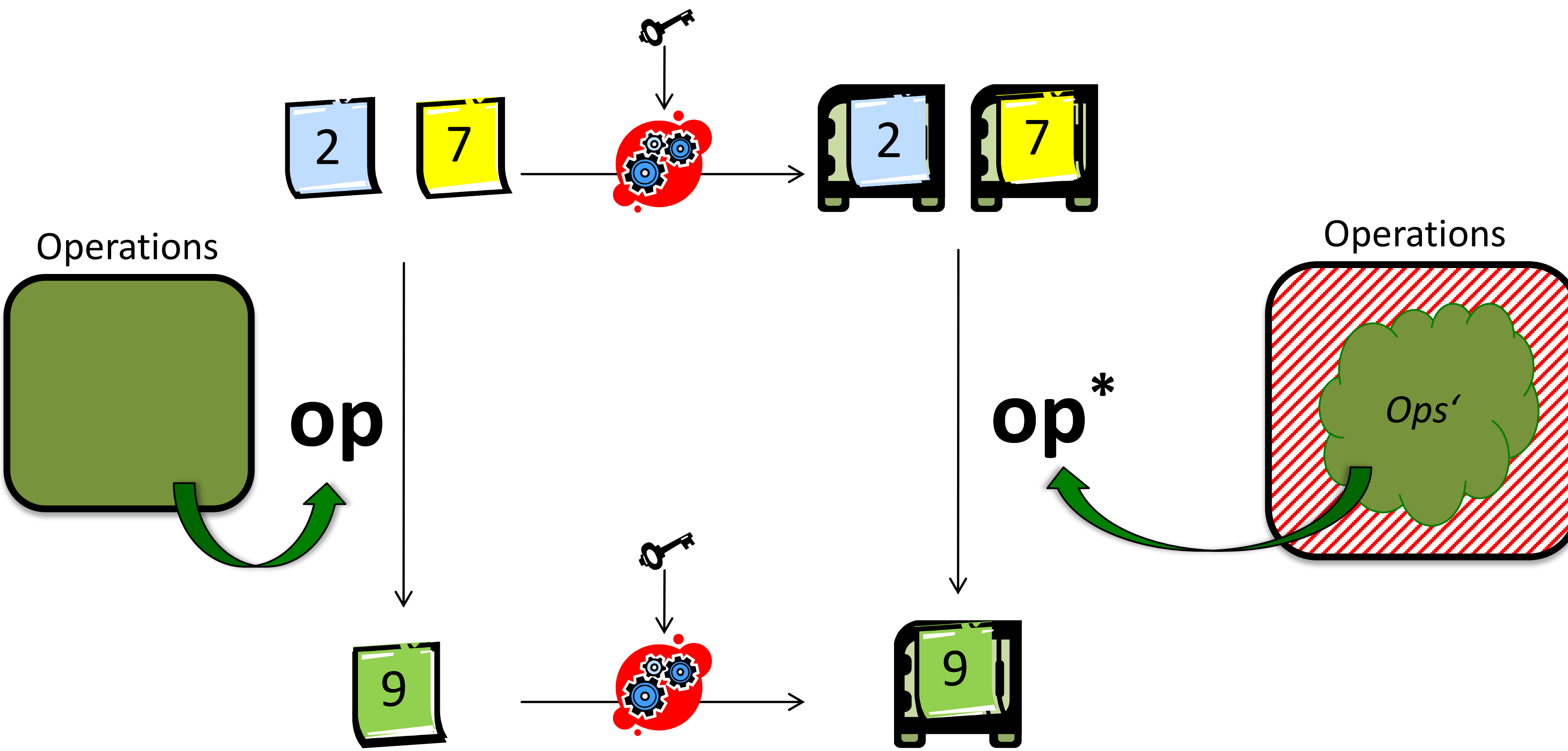
- $\mu - 1$ = #multiplications, #fresh encryptrions $\approx n/2$
- **Observe: we can use any finite field that is big enough, e.g., GF(2^r) (efficiency)**

Security Parameter	$s = 80$	$s = 128$	$s = 256$	$s = 80$	$s = 128$	$s = 256$
μ	$\mu = 2$			$\mu = 3$		
$n_{\min}$	4,725	8,411	19,186	14,236	26,280	61,044
$\log_2(q_{\min})$	17	18	23	18	19	24

Parameters	Effort Setup	Effort Encryption	Effort Decryption	Effort Addition	Effort Multiplication
$\mu = 2$ $s = 80$	Min: 1m 57.781 s Max: 1m 58.998s Av: 1m 58.33s	Min: 0.031s Max: 0.11s Av: 0.072s	Min: $< 10^{-28}$s Max: 0.032s Av: 0.001	Min: $< 10^{-28}$s Max: 0.016s Av: 0.000573s	Min: $< 10^{-28}$s Max: 0.032s Av: 0.005238s
$\mu = 2$ $s = 128$	Min: 1h 18m 22.089 s Max: 1h 20m 21.024s Av: 1h 19m 12.149s	Min: 0.686s Max: 1.014s Av: 0.817s	Min: $< 10^{-28}$s Max: 0.016s Av: 0.004s	Min: $< 10^{-28}$s Max: 0.031s Av: 0.0017s	Min: $< 10^{-28}$s Max: 0.032s Av: 0.01044s
$\mu = 3$ $s = 80$	Min: 46m 3.089 s Max: 47m 4.024s Av: 46m 40.149s	Min: 0.171s Max: 0.312s Av: 0.234s	Min: $< 10^{-28}$s Max: 0.016s Av: 0.002s	Min: $< 10^{-28}$s Max: 0.016s Av: 0.0015s	Min: $< 10^{-28}$s Max: 0.047s Av: 0.014s

Fully Homomorphic Encryption

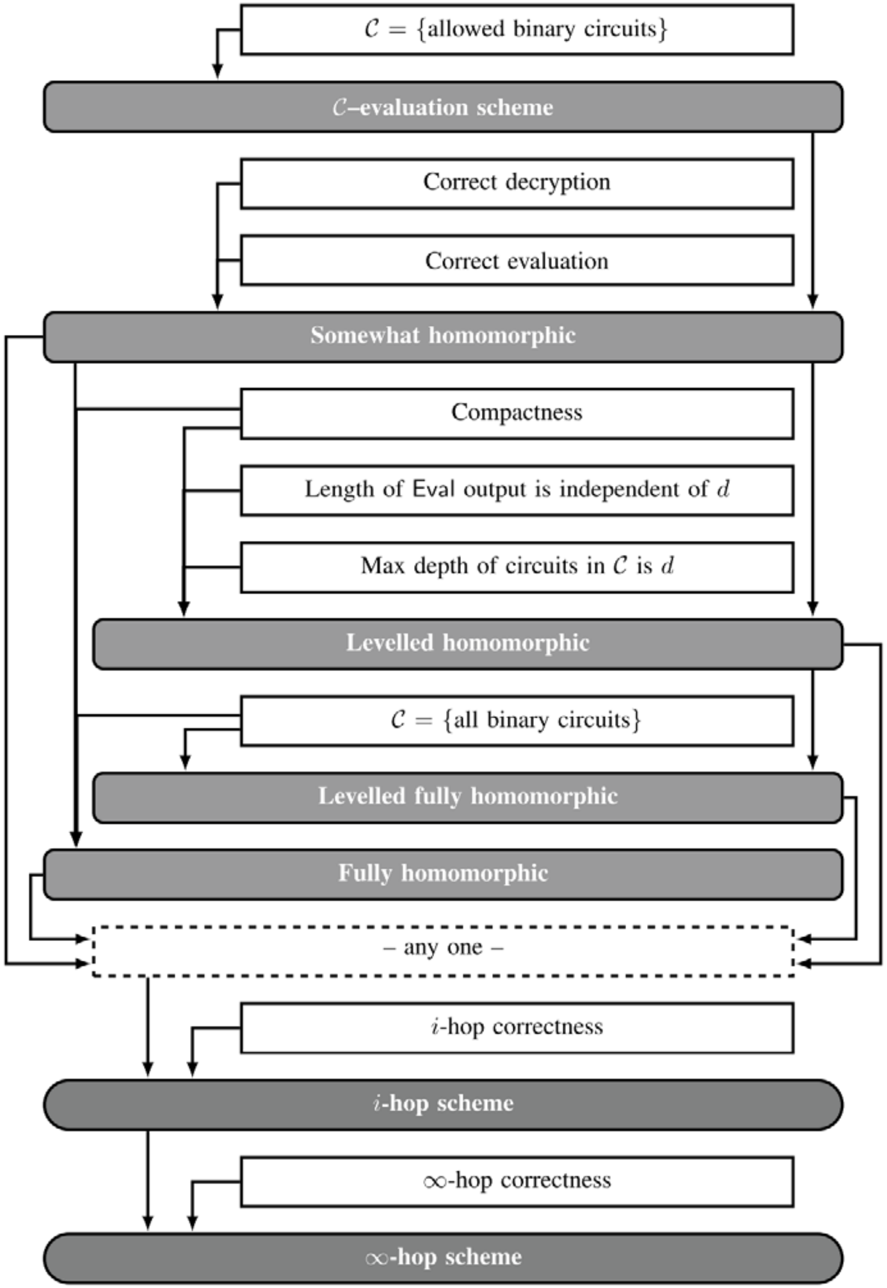
A fully homomorphic encryption scheme is homomorphic wrt all possible operations



UNIVERSITÄT MANNHEIM

Gentry's Breakthrough Result (2009)

Theory?


$\mathcal{C}$ = {allowed binary circuits}
$\mathcal{C}$–evaluation scheme
Correct decryption
Correct evaluation
Somewhat homomorphic
Compactness
Length of Eval output is independent of d
Max depth of circuits in $\mathcal{C}$ is d
Levelled homomorphic
$\mathcal{C}$ = {all binary circuits}
Levelled fully homomorphic
Fully homomorphic
– any one –
i-hop correctness
i-hop scheme
∞-hop correctness
∞-hop scheme

Practice?

Homomorphic Evaluation of the AES Circuit

Craig Gentry
IBM Research

Shai Halevi
IBM Research

Nigel P. Smart
University of Bristol

June 15, 2012

Abstract

We describe a working implementation of leveled homomorphic encryption (without bootstrapping) that can evaluate the AES-128 circuit in three different ways. One variant takes under over 36 hours to evaluate an entire AES encryption operation, using NTL (over GMP) as our underlying software platform, and running on a large-memory machine. Using SIMD techniques, we can process over 54 blocks in each evaluation, yielding an amortized rate of just under 40 minutes per block. Another implementation takes just over two and a half days to evaluate the AES operation, but can process 720 blocks in each evaluation, yielding an amortized rate of just over five minutes per block. We also detail a third implementation, which theoretically could yield even better amortized complexity, but in practice turns out to be less competitive.

Our Implementation. Our implementation was based on the NTL C++ library running over GMP, we utilized a machine which consisted of a processing unit of Intel Xeon CPUs running at 2.0 GHz with 18MB cache, and most importantly with 256GB of RAM.[2]

State of the Art?

Scheme	Underlying Problems	Asymptotic Runtime	Concrete Instantiation Runtime
Gentry: A Fully Homomorphic Encryption Scheme [18]	BDDP & SSSP	$\mathcal{O}(\lambda^6 \log(\lambda))$ per gate	-
van Dijk, Gentry, Halevi, Vaikuntanathan: FHE over the Integers [35]	AGCD & SSSP	$\mathcal{O}(\lambda^{10})$	-
Coron, Naccache, Tibouchi: Public Key Compression and Mudulus Switsching for FHE over the Integers [13]	DAGCD & SSSP	-	Recryption (a step that takes place after every addition/multiplication) takes about 11 minutes.
Brakerski, Vaikuntanathan: Efficient FHE from (standard) LWE [9]	DLWE	$\tilde{\mathcal{O}}(\lambda^{2C})$ where C is a very large parameter that ensures bootstrappability.	-
Brakerski, Vaikuntanathan: FHE from Ring-LWE and Security for Key Dependent Messages [10]	PLWE	-	-
Brakerski, Gentry, Vaikuntanathan: FHE without Bootstrapping [8]	RLWE	Per-gate computation overhead $\tilde{\mathcal{O}}(\lambda \cdot d^3)$ (where d is the depth of the circuit) without bootstrapping, $\tilde{\mathcal{O}}(\lambda^2)$ with bootstrapping.	In [21]: 36 hours for an AES encryption on a supercomputer
Smart, Vercauteren: FHE with Relatively Small Key and Ciphertext Sizes [34]	PCP & SSSP	-	Key generation took several hours even for small parameters which do not deliver a fully homomorphic scheme, for larger parameters the keys could not be generated
Rohloff, Cousins: A Scalable Implementation of Fully Homomorphic Encryption Built on NTRU [32]	SVP & RLWE	-	Recryption at 275 seconds on 20 cores with 64-bit security
Halevi, Shoup: Bootstrapping for `HElib` [27]	RLWE	-	Vectors of 1024 elements from GF(2^{16}) was recrypted in 5.5 minutes at security level $\approx$ 76, single CPU core.

Observations

- Somewhat-homomorphic ⇒ fully-homomorphic seems to induce high costs
- Rothblum‘s result on fully-homomorphic encryption schemes: symmetric key ⇔ public key
- Question: are <u>efficient</u> fully-homomorphic encryption schemes possible at all?

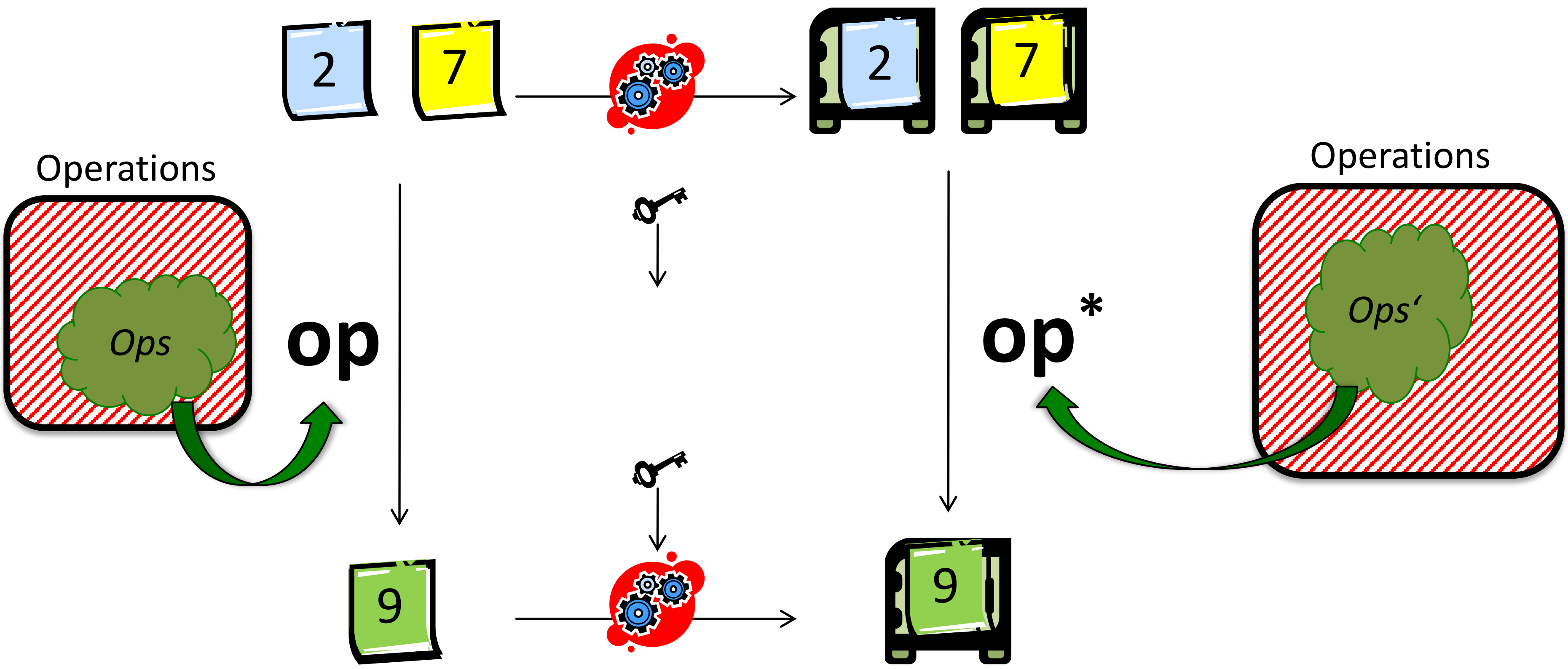
Counter-question: do we need fully-homomorphism in practice?

- Examples exist where a scheme with less functionalities would be sufficient
- Adapted homomorphic encryption schemes

Adapted Homomorphic Encryption

Adapted Homomorphic Encryption

1. Given: a concrete use case
2. Identify the necessary operations
3. Develop appropiate encryption scheme



Example: Recommender System

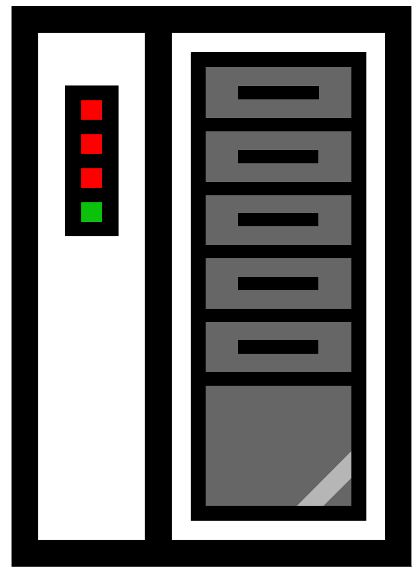
- **Recommender systems are a way of suggesting like or similar items and ideas to a user.**
- **Automates quotes like:**
  - "I like this book; you might be interested in it"
  - "I saw this movie, you' ll like it“
  - "Don' t go see that movie!"
- **Examples**
  - Amazon
  - Ebay

Considered General Scenario

Service provider

User

$(\overrightarrow{P}, obj)$



Recommendation *r*

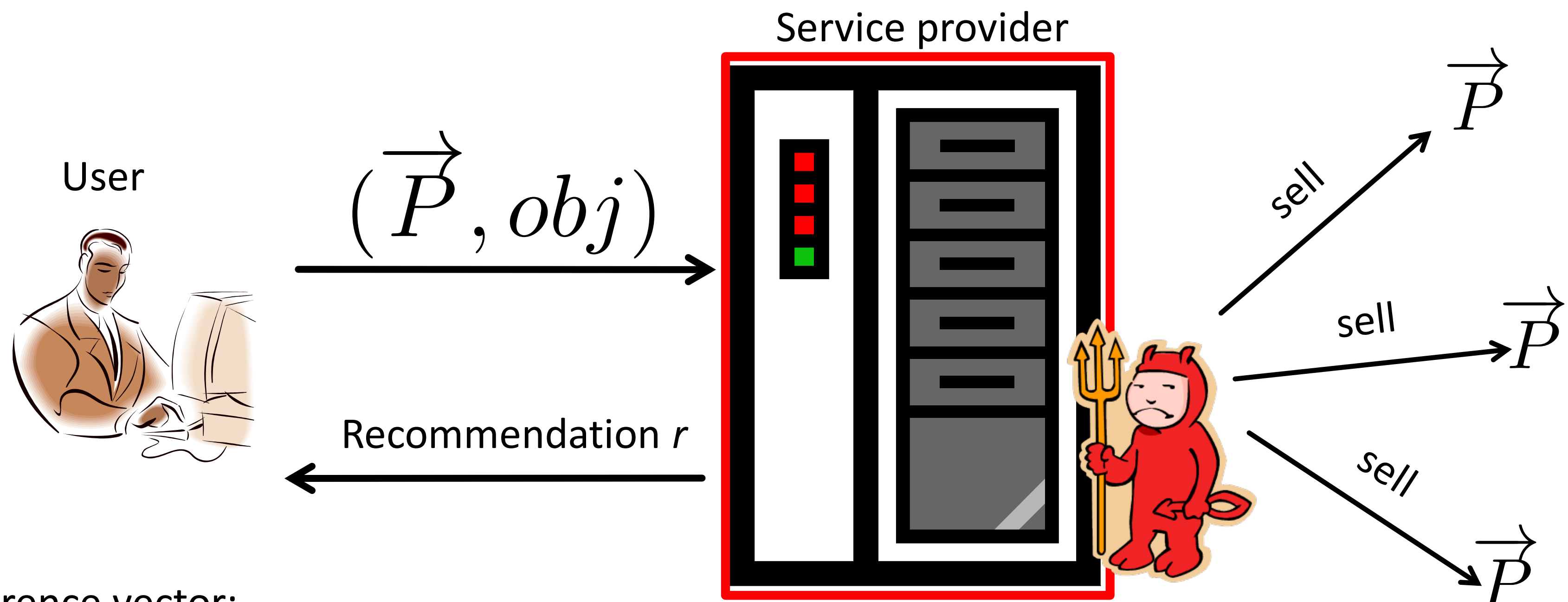
Preference vector:

$\overrightarrow{P} = (p_1, \ldots, p_n) \in R^n$

$r = \sum_{i=1}^{n} f_i(obj) \cdot p_i$

Example: Regularized Matrix Factorization (RMF) Recommender

Threat: data misuse



Preference vector:
$\overrightarrow{P} = (p_1, \ldots, p_n) \in R^n$

$r = \sum_{i=1}^{n} f_i(obj) \cdot p_i$

Question: Is it possible to ask for recommendations **without** revealing the preferences?

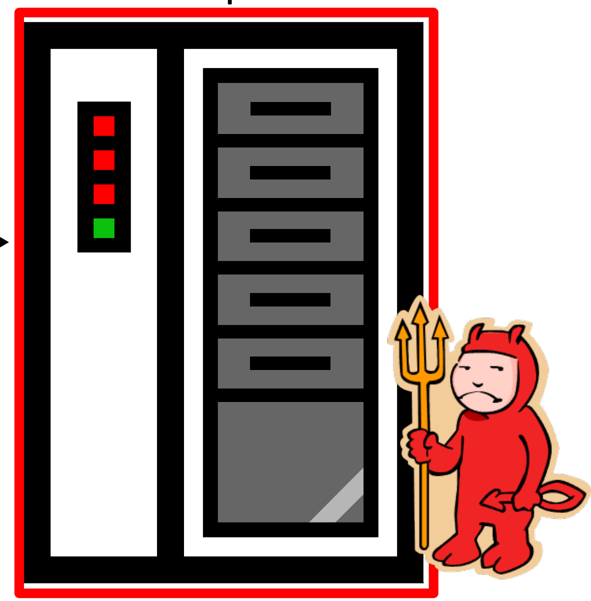
Solution

A., Strufe. Med-Hoc 2011.

User

$(\{Enc_k(p_i)\}_{i=1}^n, obj)$

Service provider



Encrypted Recommendation $Enc_k(r)$

Preference vector:

$\overrightarrow{P} = (p_1, \ldots, p_n) \in R^n$

$$\sum_{i=1}^{n} f_i(obj) \cdot Enc_k(p_i) = Enc_k(r)$$

Challenge: Develop an appropriate encryption scheme!

Our Solution

- **Encrypt preference vector such that**
  - Service provider cannot read the encrypted preferences
  - Computation on encrypted data possible
- **More formal:**
  - Encryption scheme $Enc_k(.)$ encrypts real-valued data
  - Additively homomorphic:

$$Enc_k(m) \circ Enc_k(m') = Enc_k(m + m') \quad \forall m, m' \in R$$

  - „External homomorphism“:

$$\lambda \cdot Enc_k(m) = Enc_k(\lambda \cdot m) \quad \forall \lambda, m \in R$$

Concrete Scheme

- **Adaptation of the 2011 code-based scheme**
- **Key generation**
  - Sample vector $\vec{K} \in R^n \setminus \{\vec{0}\}$
- **Encryption of a real value $\boldsymbol{m}$**
  - Generate a vector $\vec{C} \in R^n$ such that
$$\langle \vec{C}, \vec{K} \rangle = m$$
- **Decryption of a ciphertext**
  - Compute $\langle \vec{C}, \vec{K} \rangle = m$

Properties

- **Efficient (pre-computation)**
- **Additive homomorphism:** Let $\overrightarrow{C}$ and $\overrightarrow{C'}$ be an encryption of *m* and *m'*, respectively. Consider the decrpytion of $\overrightarrow{C} + \overrightarrow{C'}$:

$$\boxed{(\overrightarrow{C} + \overrightarrow{C'})^T \cdot \overrightarrow{K} = \overrightarrow{C}^T \cdot \overrightarrow{K} + \overrightarrow{C'}^T \cdot \overrightarrow{K} = m + m'}$$

- **External homomorphism:** Let $\overrightarrow{C}$ be an encryption of *m* and let λ be an arbitrary real value. Consider the decrpytion of $\lambda \cdot \overrightarrow{C}$:

$$\boxed{(\lambda \cdot \overrightarrow{C})^T \cdot \overrightarrow{K} = \lambda \cdot \left(\overrightarrow{C}^T \cdot \overrightarrow{K}\right) = \lambda \cdot m}$$

Conclusion

Summary

- **Homomorphic encryption allow for processing encrypted data without the need of decryption**
- **Many applications**
- **Problem: efficiency (in the case of huge data amount)**
- **Alternative approach: adapted homomorphic encryption schemes**

Open Questions

- **Identify further (more realistic) use cases**
- **Improve understanding between conditions and design possibilities**
- **Develop appropriate adapted cryptographic schemes**

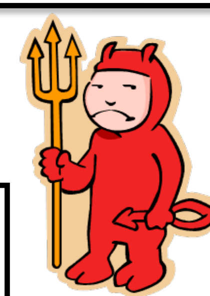
Backup Slides

Security Characterizations

Defining security: IND-CPA

Oracle

Attacker



Time

Setup

Public param.

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

C:=Encrypt(M_b)

M_0, M_1

C

Challenge

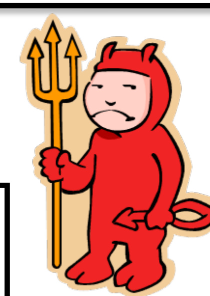
Guess for b

Attacker wins if he correctly guesses b

Defining security: IND-CCA1

Oracle

Attacker



Time

Setup — *Public param.* → Attacker

Decrypt ← c_j — Choose Ciphertext

Decrypt — m_j → Attacker

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

C:=Encrypt(M_b)

← M_0, M_1 — Challenge

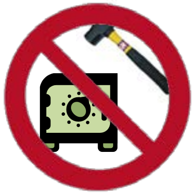
— C → Attacker

Guess for b

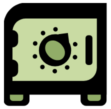
Attacker wins if he correctly guesses b

Proof of Security

Goal: Prove security of scheme



Approach:
Reduce security



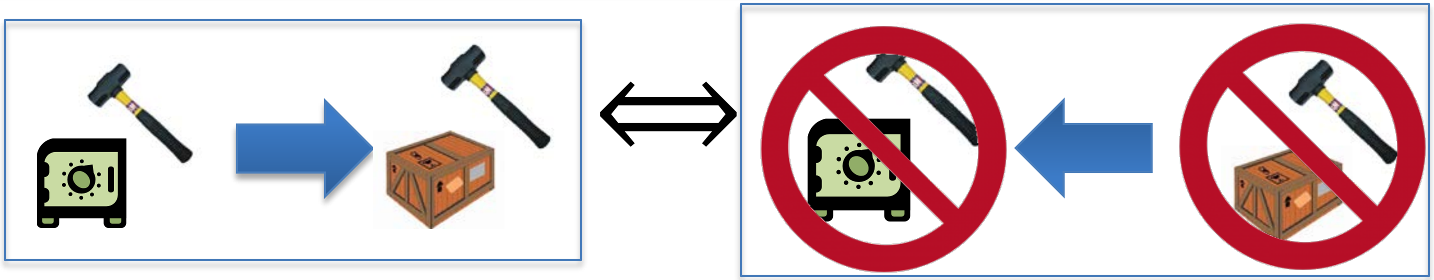
Crypto
scheme

Mathematical
Problem

Assumption:

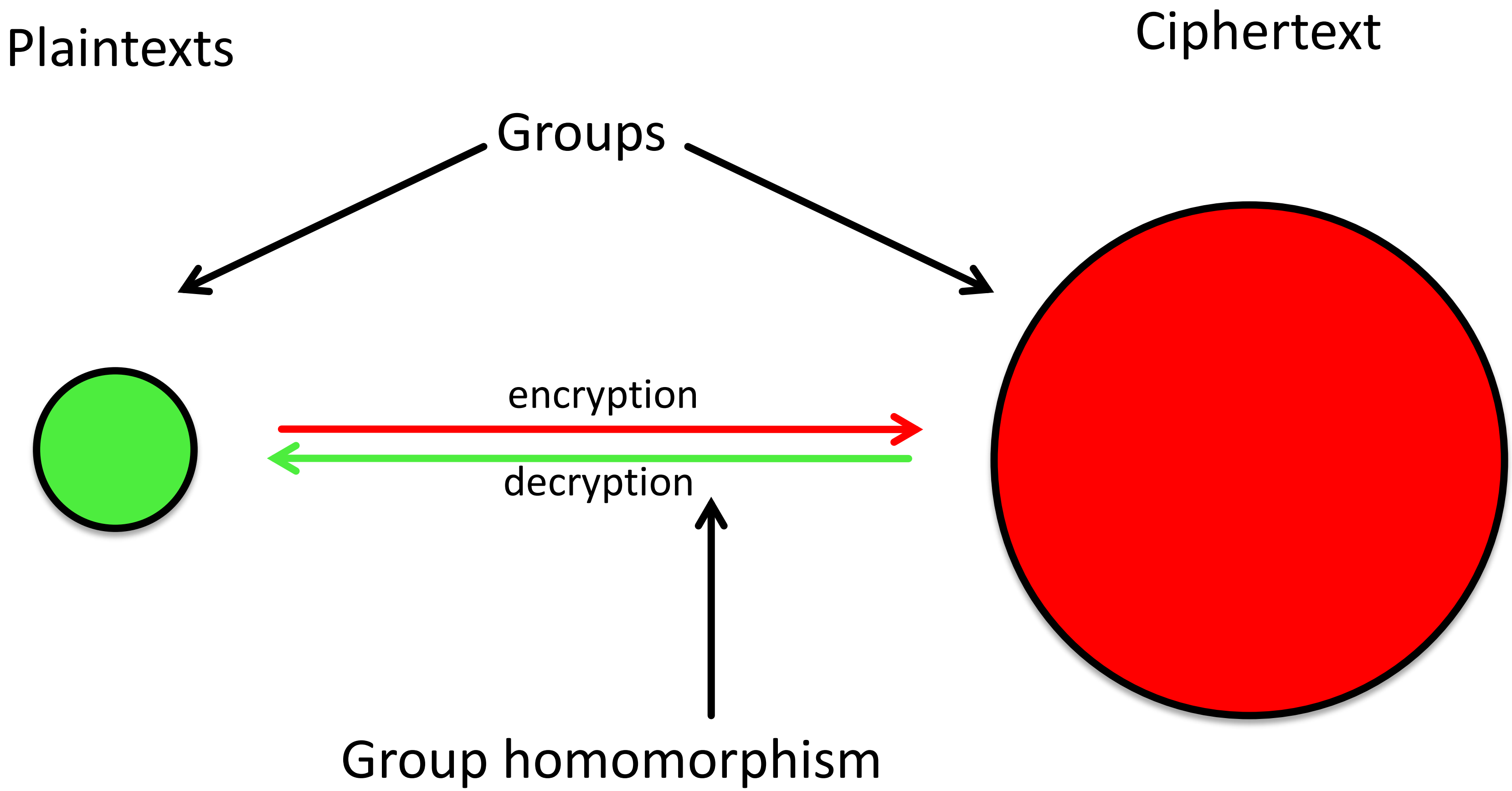
Mathematical
problem is is
hard to solve

Reduction:

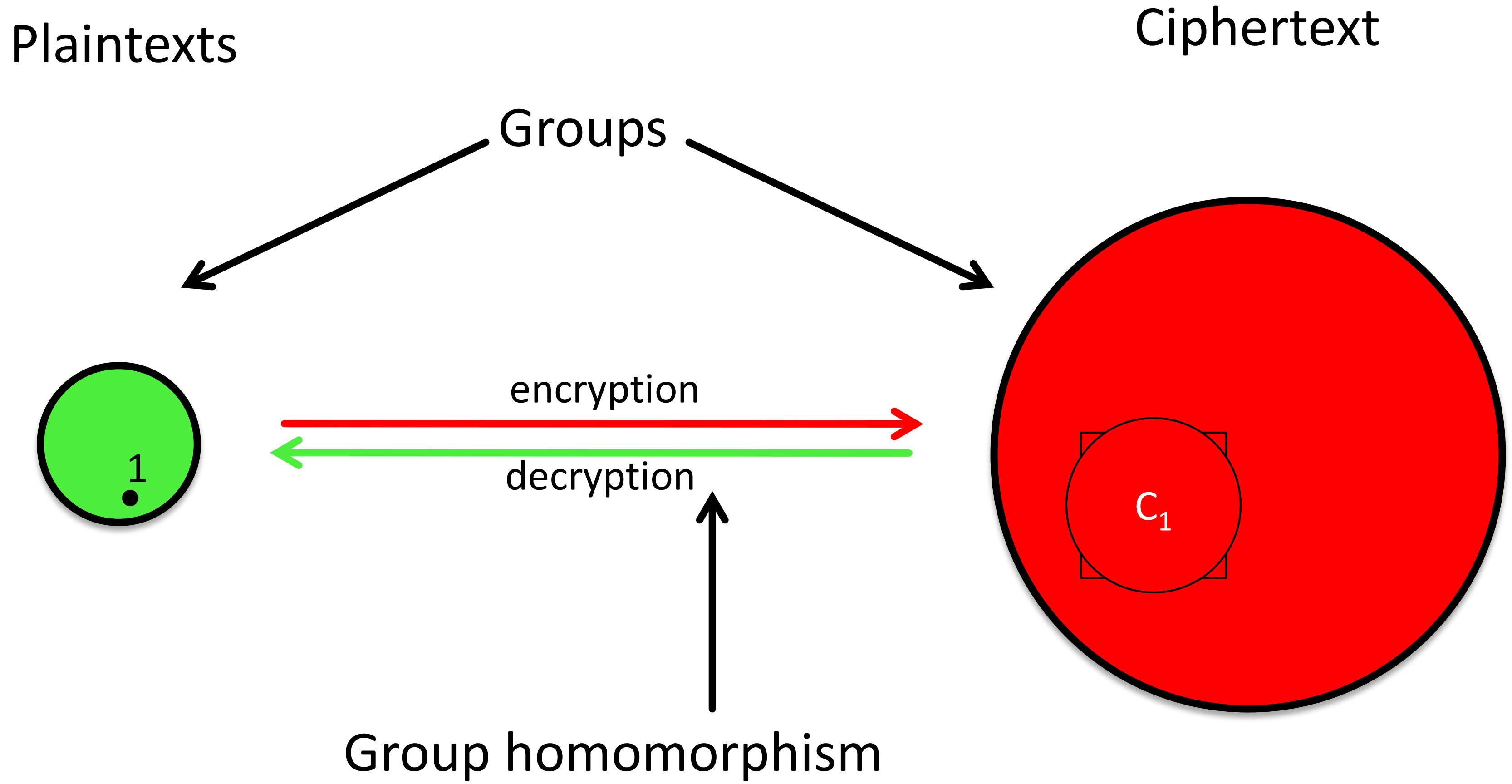


Characterization of Group Homomorphic Encryption Schemes

Recall: Considered Hom. Encr. Schemes

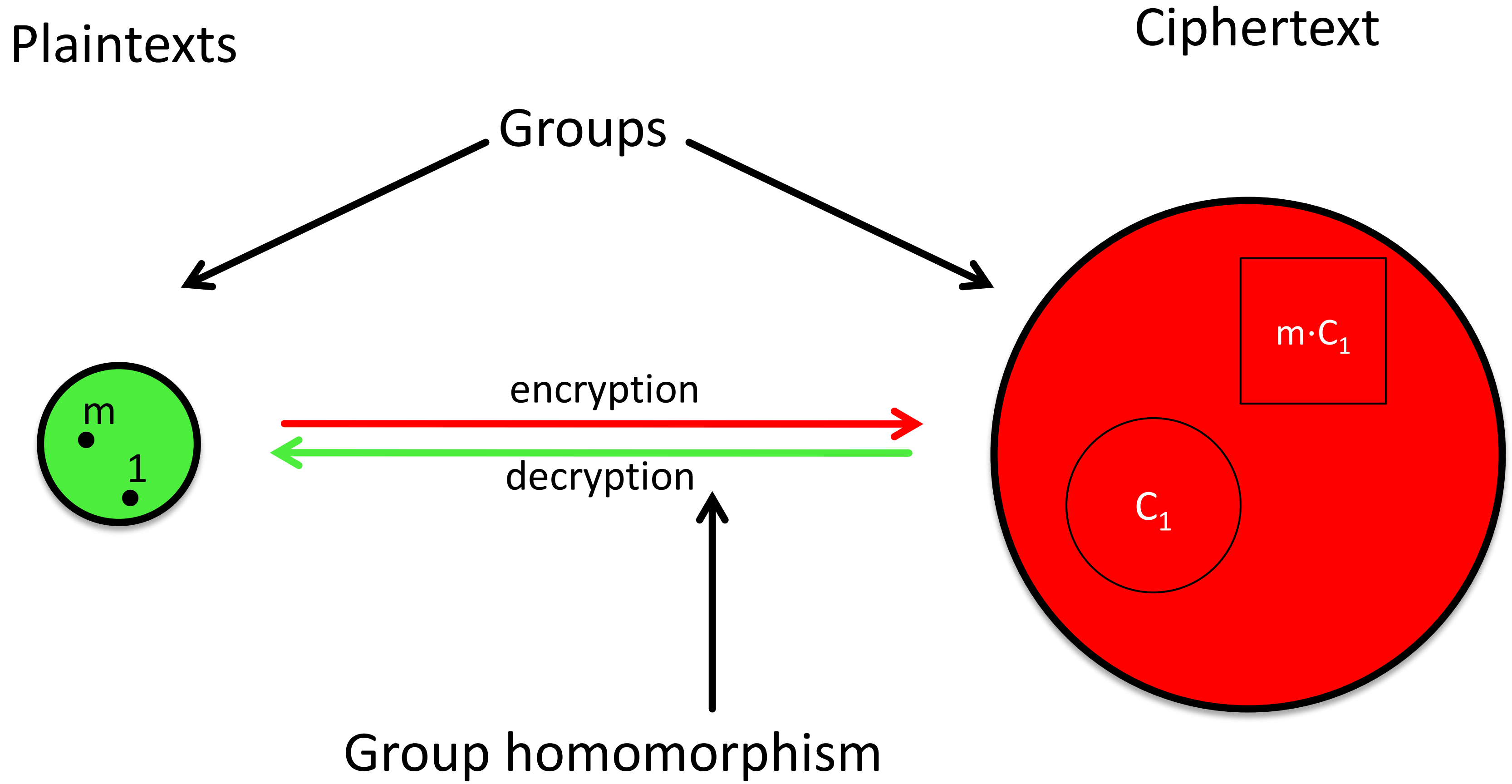


1st Observation: Encryption of "1"



Encryptions of „1" form a subgroup of the ciphertext space!

2nd Observation: Encryption of m≠1



Set of encryptions of „m“ is equal to $m \cdot C_1$

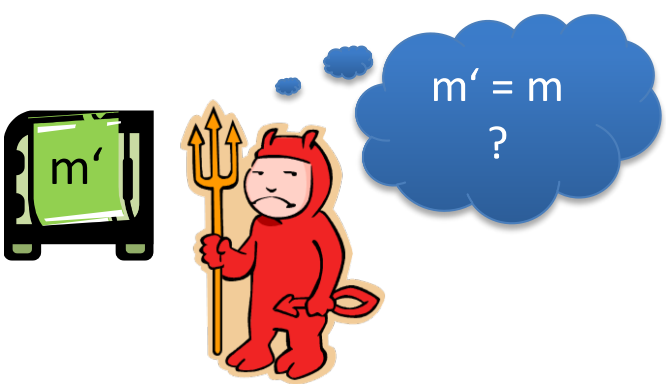
Consequence

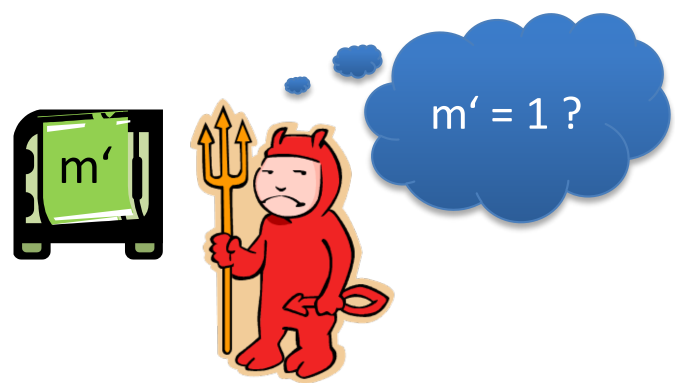
Simple observation:

c = encryp-
tion of m $\Leftrightarrow$ $c \in m \cdot C_1$ $\Leftrightarrow$ $c \cdot m^{-1} \in C_1$

Consequence:

Recognizing encryptions of **m** $\Leftrightarrow$ Recognizing encryptions of **1**

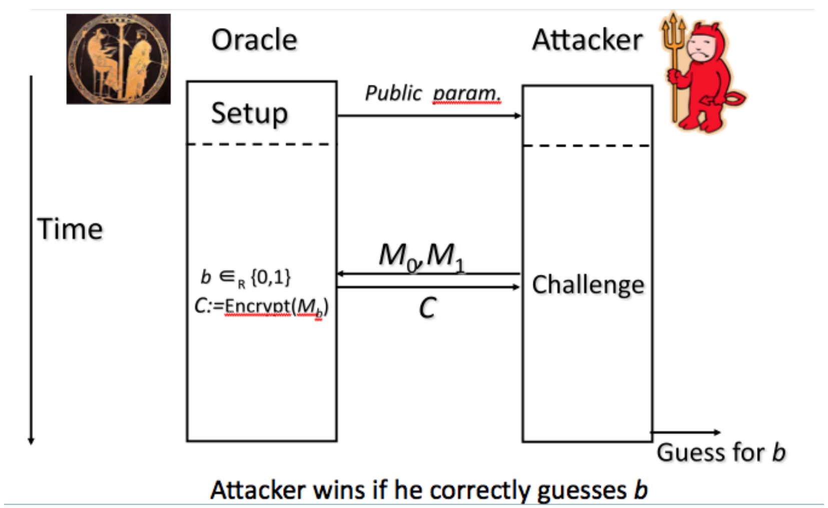




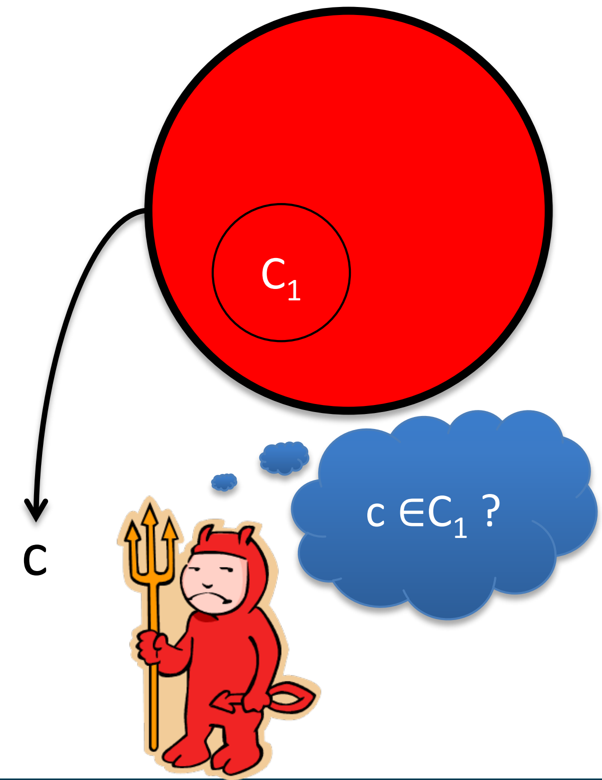
Security Characterization

Scheme is IND-CPA SECURE

Subgroup membership problem (SMP) is hard w.r.t. C_1



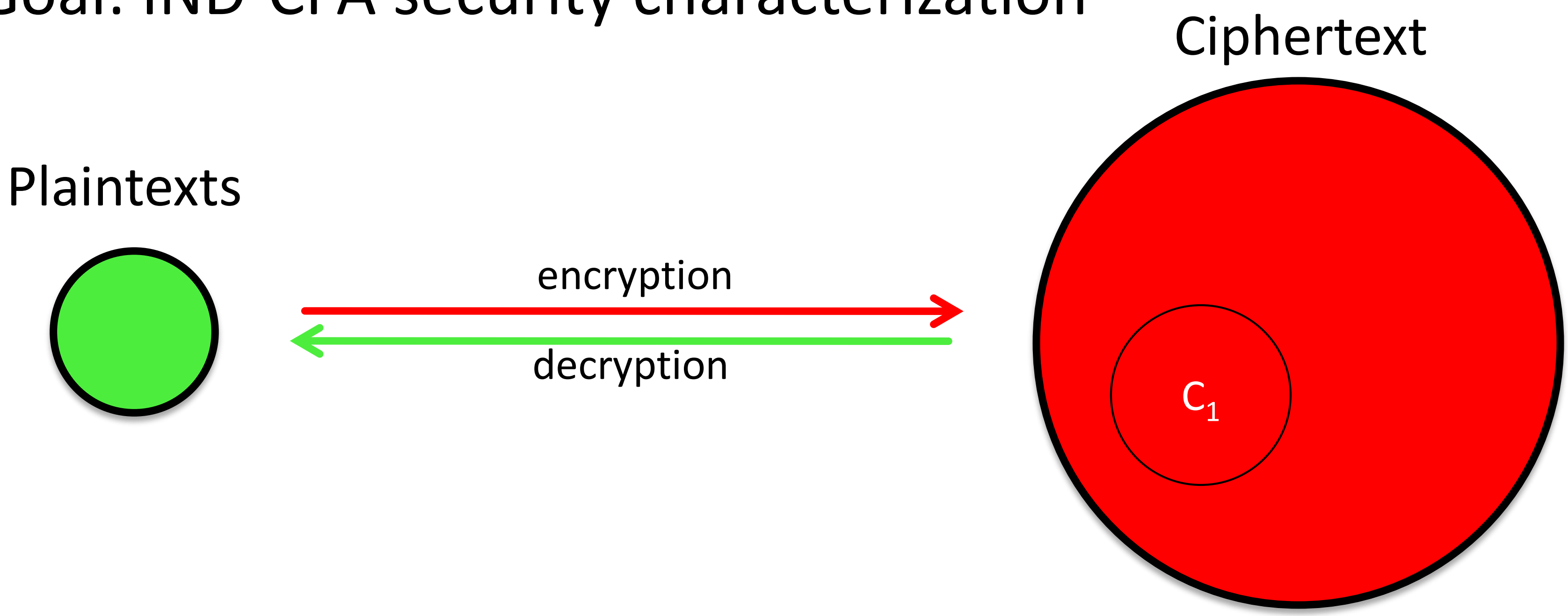
$\Leftrightarrow$



Application

Let a homomorphic scheme be given

Goal: IND-CPA security characterization



1. Identify subgroup C_1 (= encryptions of 1)
2. Formulate SMP wrt. to C_1

Application: Easy IND-CPA characterization of existing schemes

Scheme	IND-CPA secure <u>if and only if</u> the following problem is hard	IND-CCA1 secure if the following problem is hard
ElGamal; 1985	Decision Diffie-Hellman; 1998	[Lipmaa; 2010]
Paillier; 1999	N^{th} residues mod N^2; 1999	??
Daamgard, Jurik; 2001	N^{th} residues mod N^{s+1}; 2001	??
Boneh et al.; 2005	Decision Diffie-Hellman; 2005	??

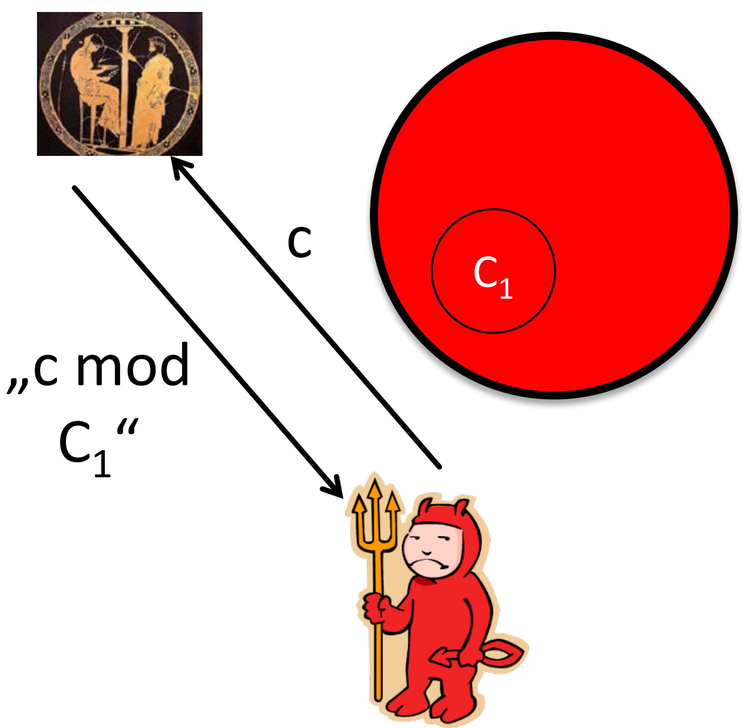
What about IND-CCA1 ?

SOAP

SOAP = **S**plitting **o**racle **a**ssisted SM**P**

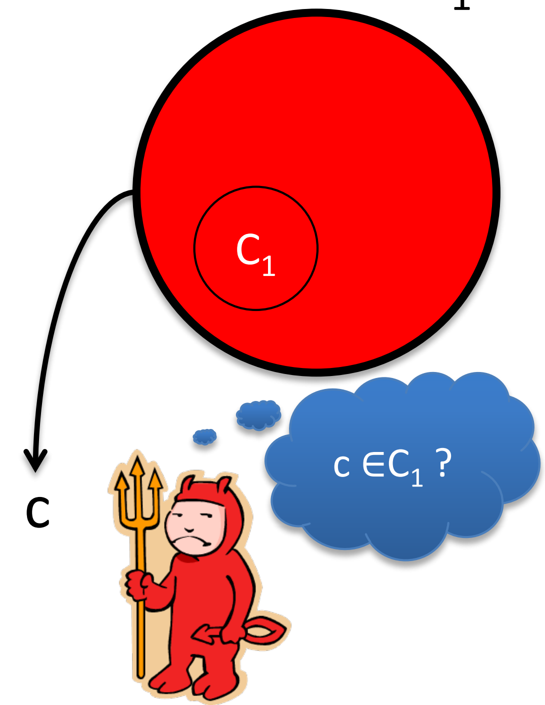
Phase 1: Learning

Splitting Oracle



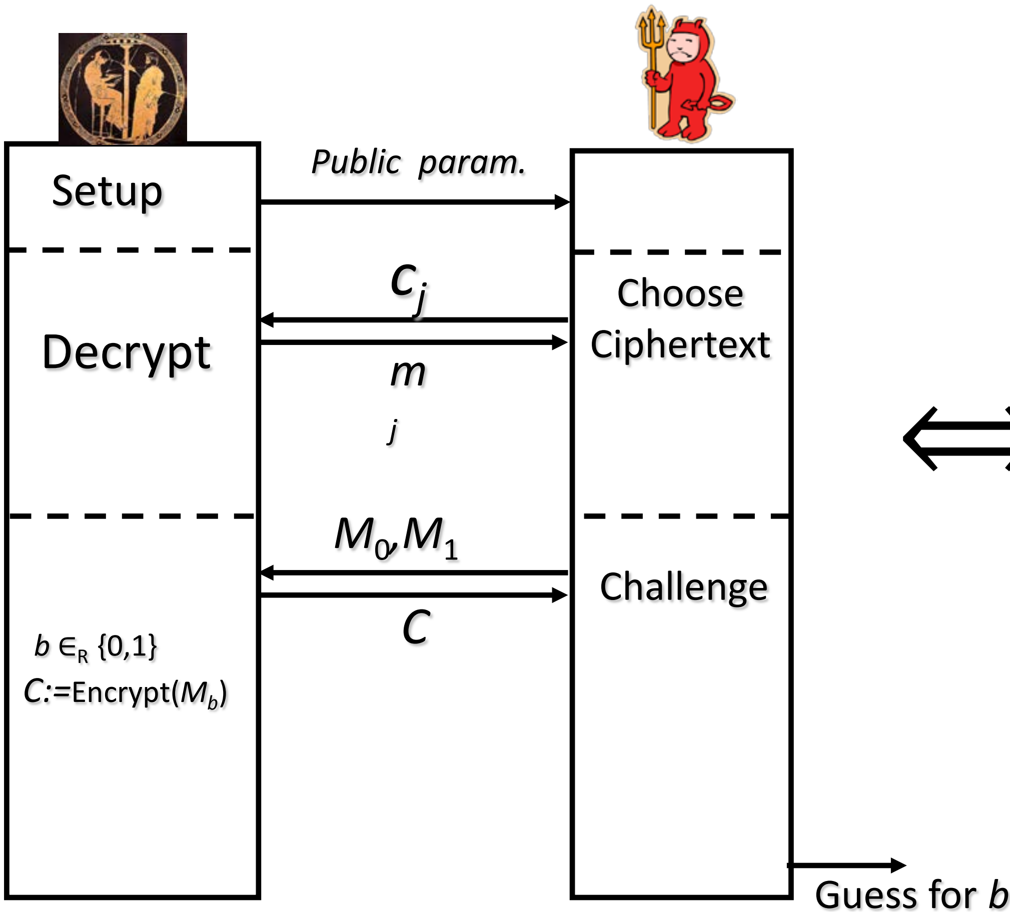
Phase 2: Challenge

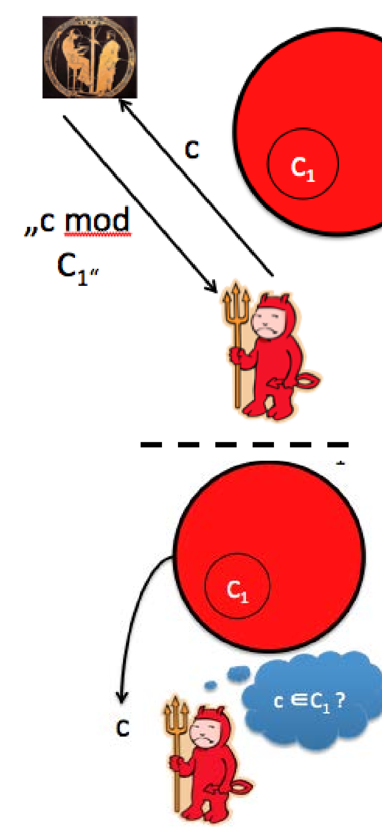
SMP w.r.t. C_1



Security Characterization

Scheme is IND-CCA1 SECURE $\Longleftrightarrow$ SOAP is hard w.r.t. C_1

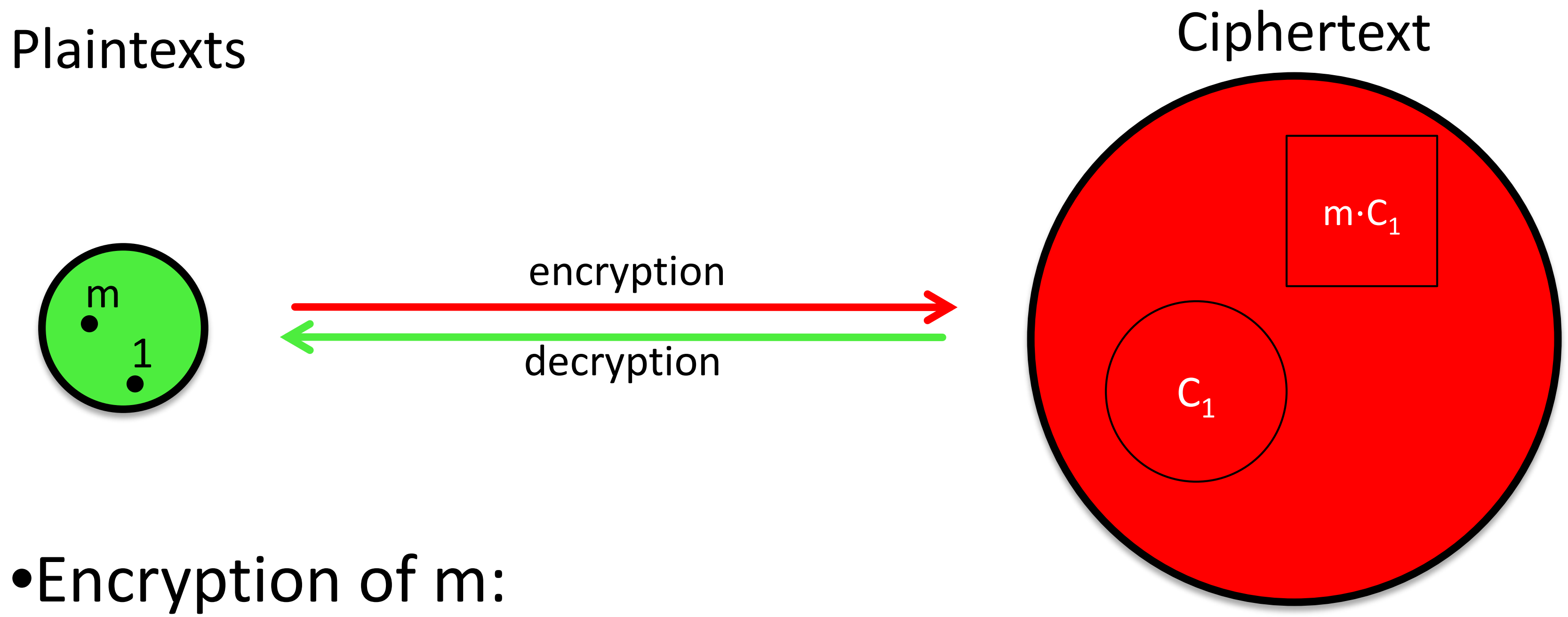




Application: IND-CCA1 Characterization of Existing Schemes

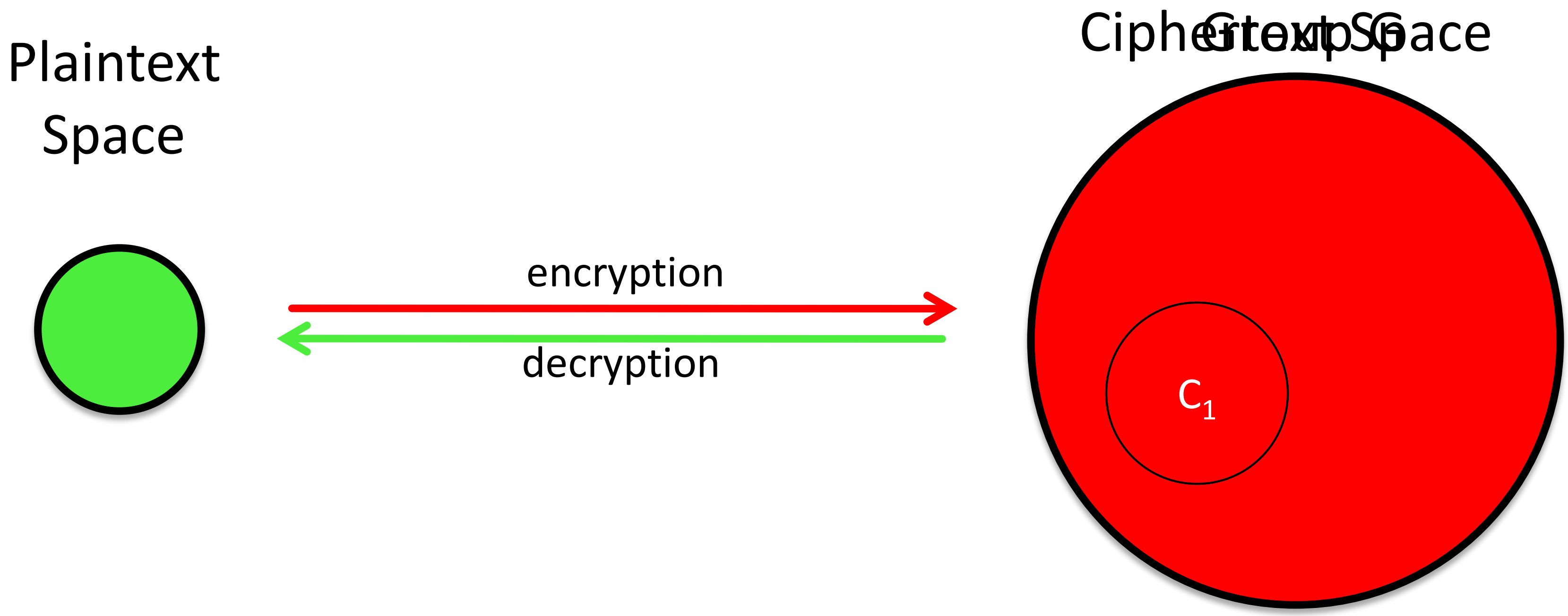
Scheme	IND-CPA secure if and only if the following problem is hard	IND-CCA1 secure if and only if the following problem is hard
ElGamal; 1985	Decision Diffie-Hellman; 1998	[Lipmaa; 2010]
Paillier; 1999	N^{th} residues mod N^2; 1999	✓
Daamgard, Jurik; 2001	N^{th} residues mod N^{s+1}; 2001	✓
Boneh et al.; 2005	Decision Diffie-Hellman; 2005	✓

Generic scheme



- Encryption of m:
  - Sample $c' \in C_1$
  - Output $c := m \cdot c'$
- Decryption of c:
  - Determine $c \bmod C_1$

Application: Design of New Schemes



- Given: SMP with group G and subgroup S
- Interpret G as ciphertext space and S as encryption of 1
- Construct encryption/decryption as described before
- Scheme is IND-CPA secure iff initial SMP is hard

Application: New Schemes

Scheme	IND-CPA secure if the following problem is hard	IND-CCA1 secure if the following problem is hard
ElGamal; 1985	Decision Diffie-Hellman; 1998	[Lipmaa; 2010]
Paillier; 1999	N^{th} residues mod N^2; 1999	✓
Daamgard, Jurik; 2001	N^{th} residues mod N^{s+1}; 2001	✓
Boneh et al.; 2005	Decision Diffie-Hellman; 2005	✓
Scheme 1	K-linear Problem	New Problem
Scheme 2	Gonzales Nieto et al.; 2005	New Problem

Scheme 1

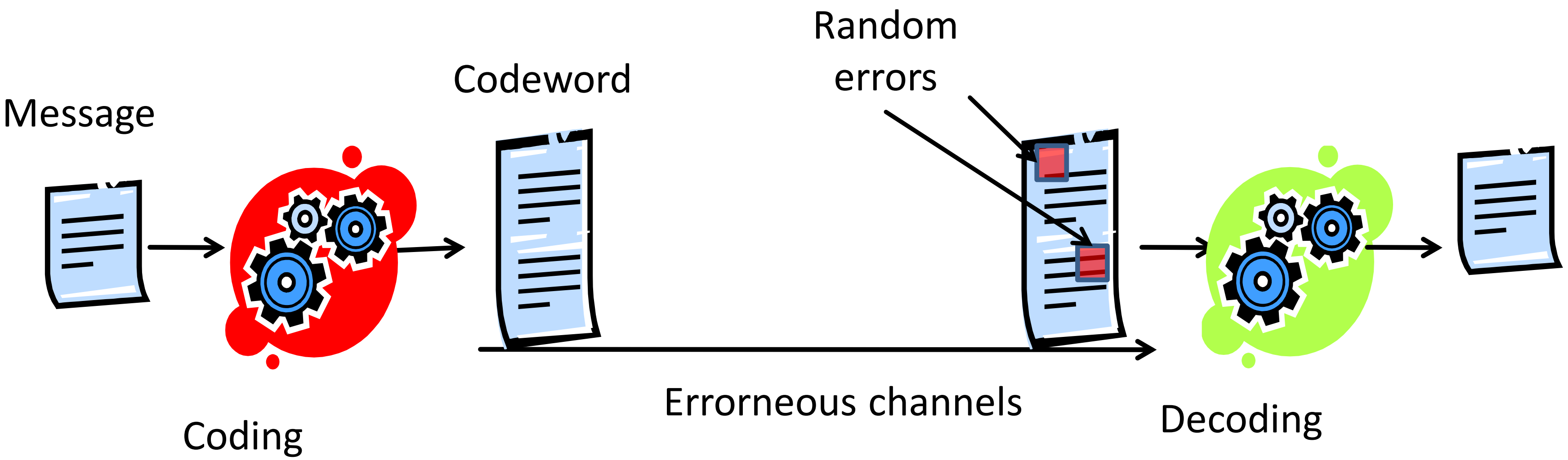
- **IND-CPA secure if and only if k-linear problem is hard**
- **K-linear problem:**
  - Extension of Diffie-Hellman problem
  - Can be instantiated for any positive integer k
  - In generic group model: is hard for k+1 even if weak for k

Scheme 2

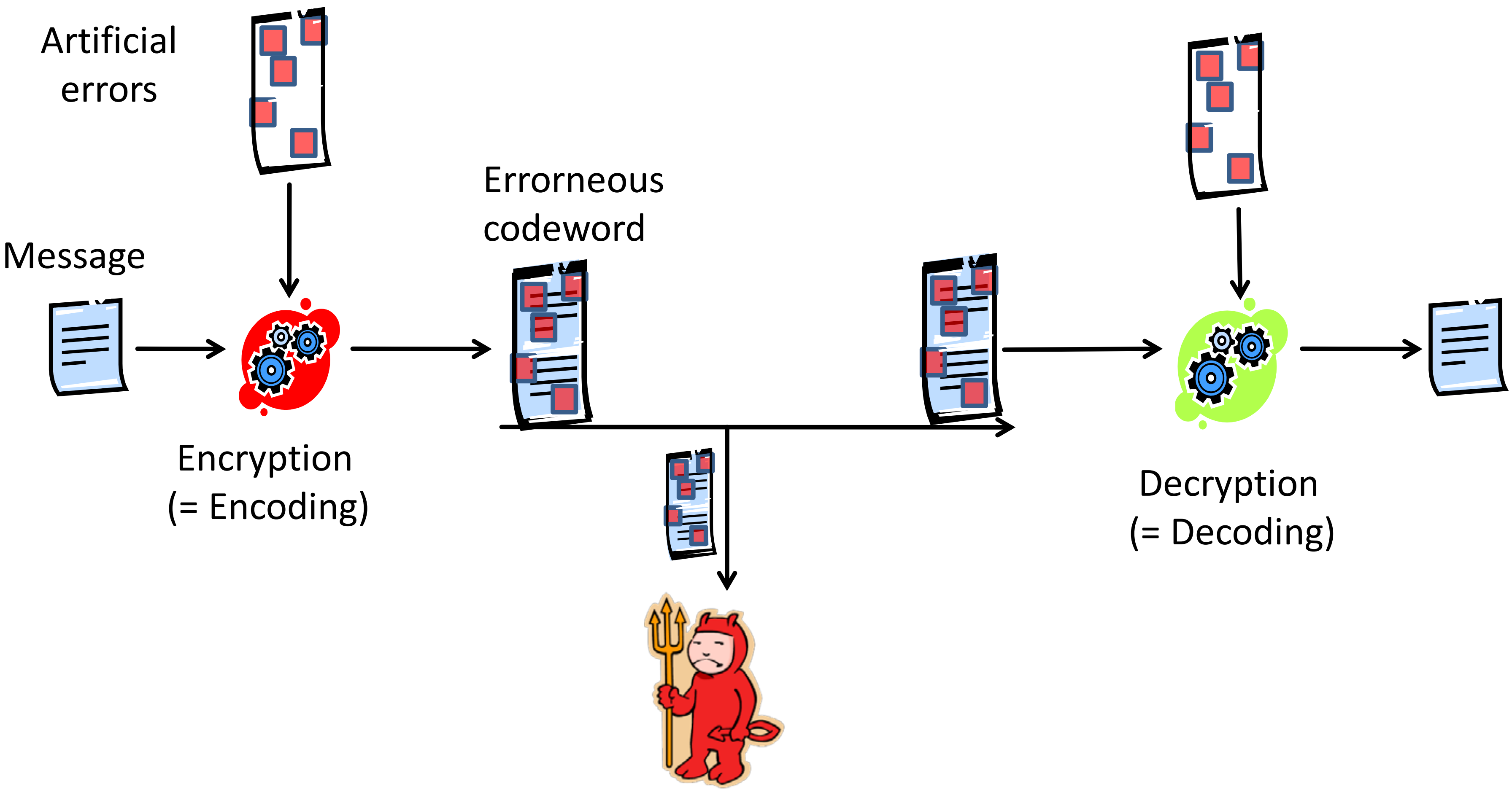
- IND-CPA secure if and only if a problem introduced by Manuel Gonzáles, Boyd, and Dawson is hard
- Distinctive feature: First homomorphic scheme with a cyclic ciphertext group
- Can be directly combined with a work by Hemenway and Ostrovsky for efficiently constructing IND-CCA2 secure schemes

The Code-Based Encryption Scheme

Coding Theory



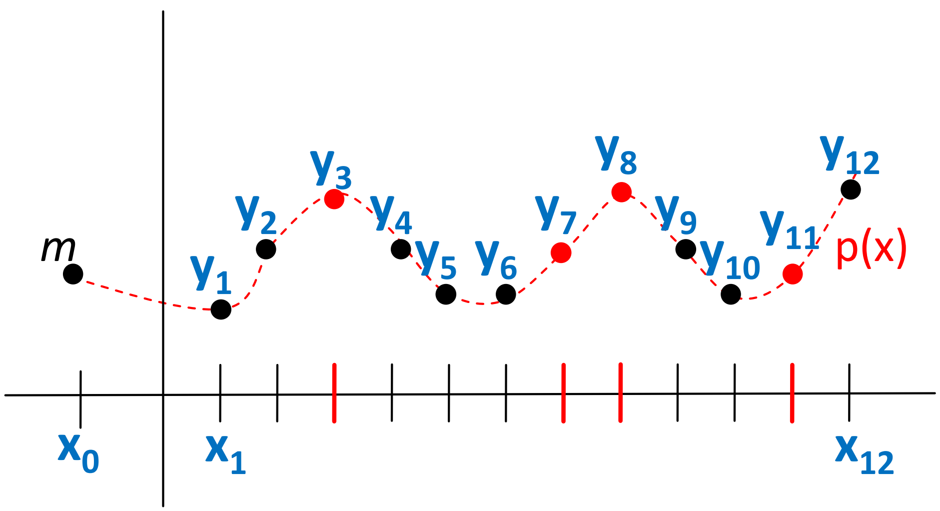
Encryption based on Coding Theory



Example: Reed-Solomon Codes

Encryption of a plaintext *m*

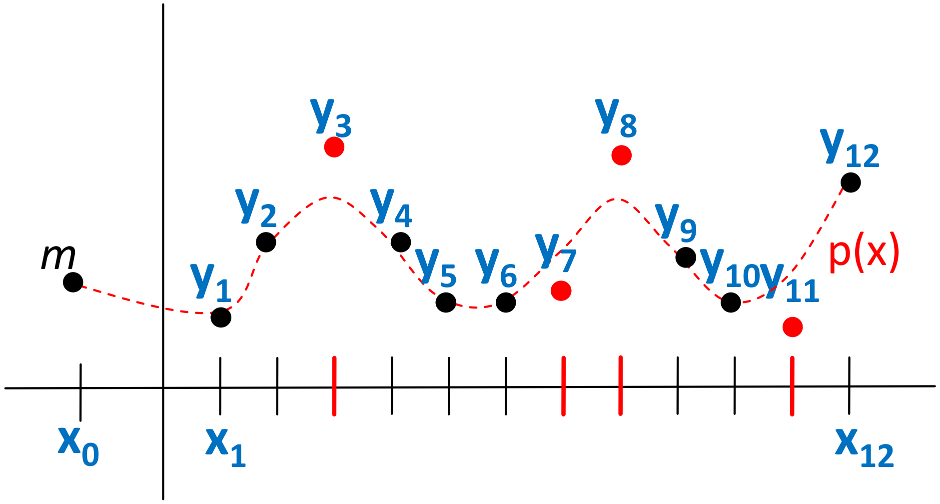
- **Parameters:**
  - Finite field F; support points $x_0,x_1,\ldots,x_n$; degree d
  - Encryption key: I = error positions
- **Encryption of a message *m*:**
  - Choose random polynomial $p(x)$ of degree d *with* $p(x_0)=m$
  - Compute $Y:=(y_1,\ldots,y_n):=(p(x_1),\ldots,p(x_n))$
  - Randomize y_i at error positions
  - Ciphertext $C=(y_1,\ldots,y_n)$ (= erroneous Reed-Solomon codeword)



Example: Reed-Solomon Codes

Decryption of a ciphertext $\vec{c}=(y_1,\ldots,y_n)$:

- Ignore errorneous y_i - values
- Interpolate $p(x)$ through the remaining, correct y_i -values
- Compute $p(x_0)=m$



Additive Homomorphism

$\vec{c}$ =($p(x_1)$, c_2, $p(x_3)$,c_4, c_5,$p(x_6)$) = encryption of $p(x_0)=m$

+

$\overrightarrow{c'}$ =($p'(x_1)$, c'_2, $p'(x_3)$,c'_4, c'_5,$p'(x_6)$) = encryption of $p'(x_0)=m'$

=

$\overrightarrow{c''}$ =($(p+p')(x_1)$, c''_2, $(p+p')(x_3)$,c''_4, c''_5,$(p+p')(x_6)$) = encryption of $(p+p')(x_0)=m+m'$

Multiplicative Homomorphism

$\vec{c}$=(p(x_1), c_2, p(x_3),c_4, c_5,p(x_6)) = encryption of p(x_0)=m

•

$\vec{c'}$=(p'(x_1), c'$_2$, p'(x_3),c'$_4$, c'$_5$,p'(x_6)) = encryption of p'(x_0)=m'

=

$\vec{c''}$ =((p·p')(x_1), c''$_2$, (p·p')(x_3),c''$_4$, c''$_5$,(p·p')(x_6)) = encryption of (p·p')(x_0)=m·m'

if degree is not too high

Generic Scheme

- **Key generation:**
  - Sample vector $\overrightarrow{K} \in \mathbb{F}^n \setminus \{\overrightarrow{0}\}$ with certain properties
- **Encryption of a real value *m***
  - Output a vector $\overrightarrow{C} \in \mathbb{F}^n$ such that

$$\overrightarrow{C}^T \cdot \overrightarrow{K} = m$$

- **Decryption of a ciphertext** $\overrightarrow{C} \in \mathbb{F}^n$
  - Compute

$$\overrightarrow{C}^T \cdot \overrightarrow{K} = m$$

Restrictions

1. **Number of encryptions needs to be limited**
   - Otherwise, key can be recovered by solving a system of linear equations
2. **Cannot be public-key**
   - All encryptions of 0 form a sub-space C_0
   - If public-key, an attacker can derive a basis for C_0
   - Once such a basis is known, one can easily decide if ciphertext is encryption of 0
   - This is equivalent to win the IND-CPA game

Security

- **Proof of security**
  - Scheme is secure if Decisional Synchronized Codeword Problem (DSCP) is hard
- **Hardness of DSCP?**
  - Depends on the deployed code
  - For Reed-Muller codes, extensive analysis conducted
  - Identified parameter ranges that seem to provide certain levels of security