GIAC
全球互联网架构大会
GLOBAL INTERNET ARCHITECTURE CONFERENCE

使用全同态加密构造数据安全方案的安全风险

彭峙酿

大纲

介绍全同态密码

介绍全同态密码库SEAL

全同态密码库的安全问题(SEAL为例)

- 针对全同态密码的CCA攻击
- 针对PSI协议的数据恢复攻击
- 全同态密码的电路隐私问题
- 编码安全问题
- 解决办法

其他问题

总结

FORTUNE

L.A. Sues IBM's Weather Company over 'Deceptive' Weather Channel App

By DAVID MEYER January 4, 2019

The Weather Channel's app secretly sucks up users' personal data and uses it for things like targeted marketing and hedge fund analysis, the Los Angeles city attorney has claimed in a lawsuit against The Weather Company, the IBM-owned firm that runs the app.

The case was first reported Thursday in the *New York Times*, but City Attorney Mike Feuer will hold a press conference Friday morning. In a tweet, he said he was tal
allege

The New York Times

Your Apps Know Where You Were Last Night, and They're Not Keeping It Secret

Dozens of companies use smartphone locations to help advertisers and even hedge funds. They say it's anonymous, but the data shows how personal it is.

By JENNIFER VALENTINO-DEVRIES, NATASHA SINGER, MICHAEL H. KELLER and AARON KROLIK DEC. 10, 2018

Data reviewed by The Times shows over 235 million locations captured from
1.2 million unique devices during a three-day period in 2017. Image by U.S.D.A

The millions of dots on the map trace highways, side streets and bike trails — each one following the path of an anonymous cellphone user.

One path tracks someone from a home outside Newark to a nearby Planned Parenthood, remaining there for more than an hour. Another represents a person who travels with the mayor of New York during the day and returns to Long Island at night.

Yet another leaves a house in upstate New York at 7 a.m. and travels to a middle school 14 miles away, staying until late afternoon each school day. Only one person makes that trip: Lisa Magrin, a 46-year-old math teacher. Her smartphone goes with her.

Bloomberg

LA County Sues IBM's Weather Channel for User Location Tracking

By Gerrit De Vynck
January 4, 2019, 11:30 AM PST
Updated on January 4, 2019, 12:12 PM PST

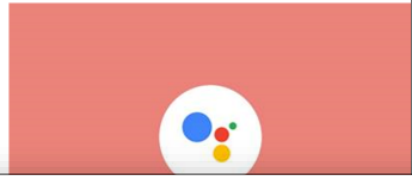
The city of Los Angeles is suing International Business Machines Corp.'s Weather Channel unit, accusing the company of misleading consumers about how their location data was being used.

WIRE

How to stop Google from tracking you and delete your personal data

Take back control of all the data Google stores about you with our easy-to-follow security tips

By MATT BURGESS
Friday 17 August 2018



It's no secret that
Valley giant's busin
data collection. Thi
interactions with t
and a lot more.

The company does
depth knowledge o
it a secret either. W
Google knows abou
purposes. Here's h
control of your da

engadget

Strava begins selling your data points, and no, you can't opt-out [Updated]

Mike Wehner, @MikeWehner
05.23.14

Strava (free) is an extremely popular running and biking app on iPhone, and has long been at or near the top of the fitness app charts. Along with its apps on other platforms including Android and even personal GPS devices, the company has pooled a whole lot of data about your running and cycling habits, which it recently used to create a stunning map of exercise routes around the world. Now the company is using that data as a product of its own, called Strava Metro.

Strava calls Strava Metro a "data service," and it's pitching its massive wealth of user patterns to transportation agencies, city governments, and corporations in a subscription-based format. The data itself, Strava notes, is scrubbed of all identifiable user information, meaning that a company or

news.com.au

How Google is secretly recording you through your mobile, monitoring millions of conversations

The Sun — News Corp Australia Network — AUGUST 23, 2017 7:50AM

If you run Android software on your smartphone, Goog
every day without you knowing. Picture: Getty. Source:

DID you know that Google has been recording you without your knowledge?

The technology giant has effectively turned millions of its users' smartphones into listening devices that can capture intimate conversations — even when they aren't in the room.

If you own an Android phone, it's likely that you've used Google's Assistant, which is similar to Apple's Siri.

Google says it only turns on and begins recording when you utter the words "OK Google".

But a *Sun* investigation has found that the virtual assistant is a little hard of hearing, reports *The Sun*.

In some cases, just saying "OK" in conversation prompted it to switch on your phone

密文域处理

数据安全问题变得越来越严峻

数据安全、隐私已成为公众关心的重要问题

各国政府出台相应的法律

能否运用加密技术解决问题

将数据加密后再发送给云端

允许云在加密的数据上进行：搜索/修改/处理 数据

数据在云端始终保持加密状态

云服务器无法解密数据内容

密文域处理

加密请求处理

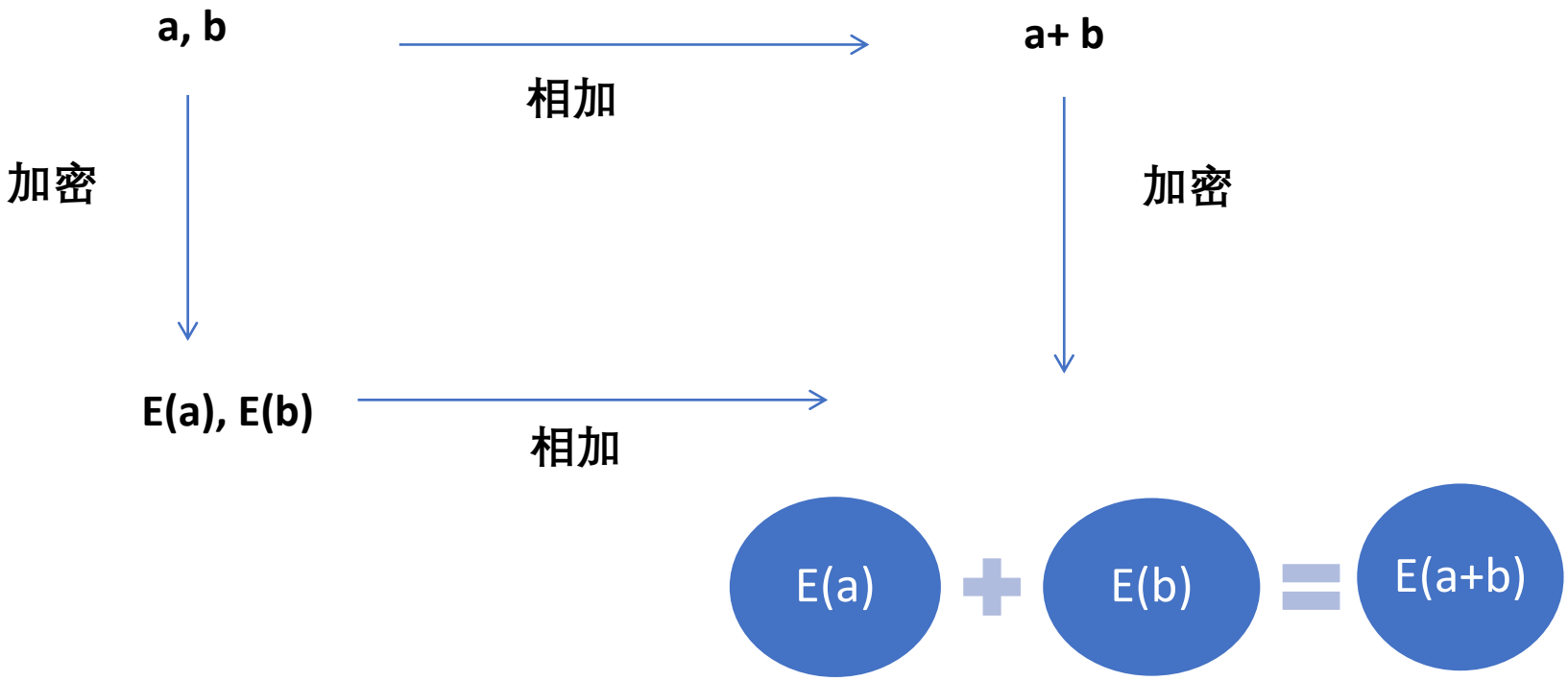
用户加密查询请求给云端

允许云处理用户的请求

云端返回加密的处理结果

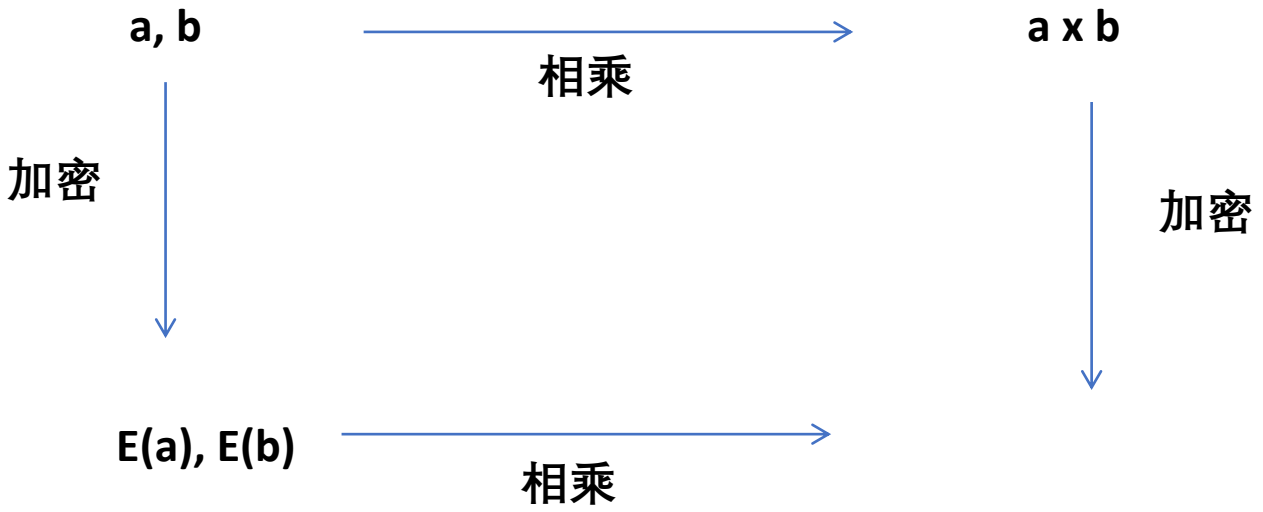
用户解密得到结果

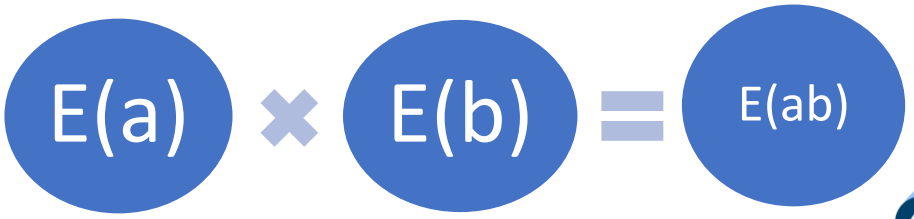
同态加密：加法同态



RSA加密支持同态加法

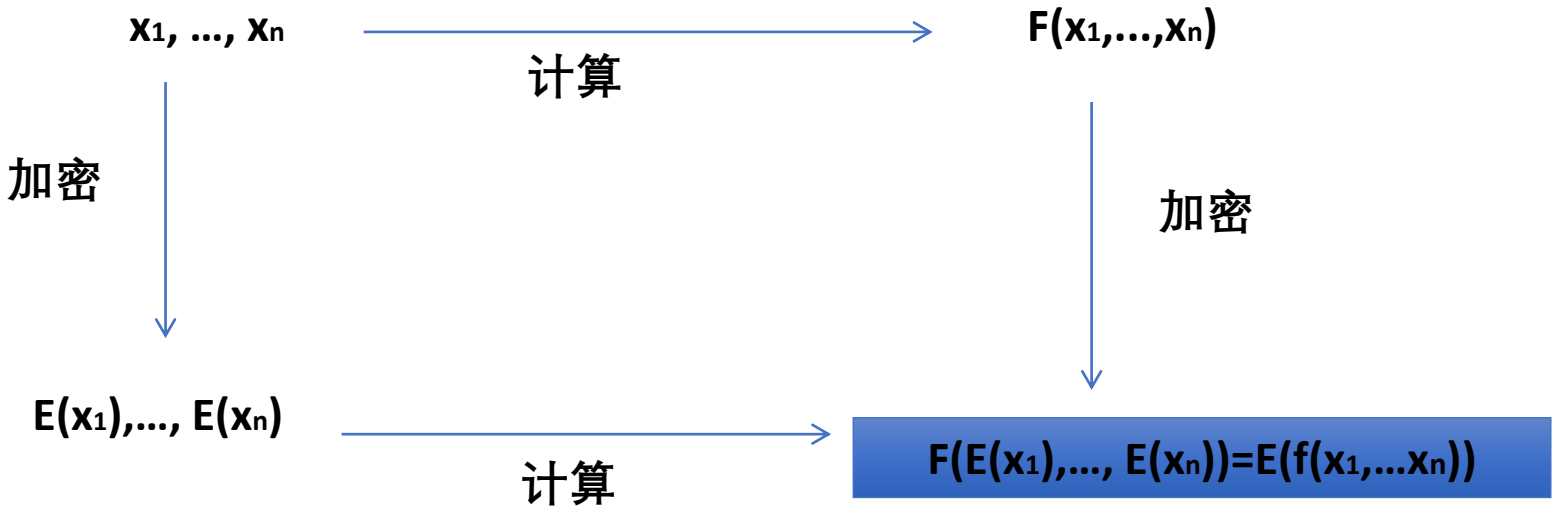
同态加密：乘法同态





Elgamal加密算法支持同态乘法

全同态加密



2009年，gentry设计出第一个全同态加密算法

使用同态加密保护数据：比喻

1. 把金子放进玻璃箱
2. 锁住箱子，藏起钥匙
3. 让工人隔着手套操作
4. 解锁箱子，得到珠宝

同态加密的应用场景

外包计算

对加密数据的机器学习

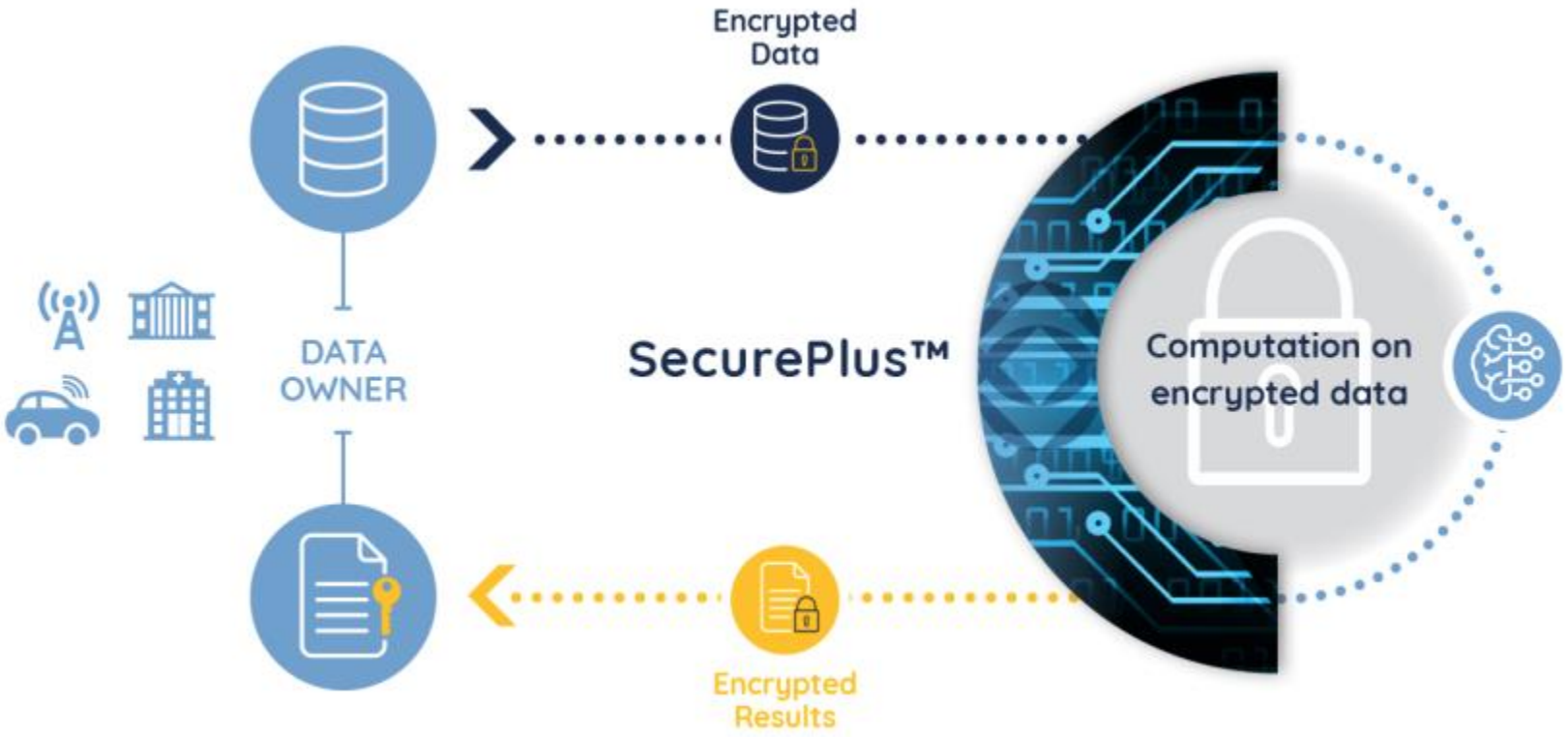
云存储，云计算服务

可以分为两类：

数据隐私，函数公开

数据隐私，函数也隐私

数据隐私，函数公开



数据要求保持隐私
要计算的函数是公开的

疾病预测

基因数据

健康监控数据

病例

实验结果

数据隐私，函数公开

所有数据使用病人的密钥进行加密

云端根据医生的要求进行加密数据计算

云端将加密的预测结果返回给病人

数据隐私，函数隐私

数据和函数都需要保持隐私

电路隐私:

要求同态密码方案能够保持要计算的函数f的隐私性

全同态密码库

SEAL

Simple Encrypted Arithmetic Library

SEAL介绍

微软研究院发布、维护的全同态密码库

2015年首次发布，目前版本为3.3

地址：https://GitHub.com/Microsoft/SEAL

标准C++开发

两种全同态方案：BFV和CKKS

简单易用

有详细的用例和注释

SEAL性能

CryptoNets (2016)

- MNIST手写数字图片识别
- 每小时6万识别，16个每秒
- 99%+的准确率
- FPGA、GPU可再提速100-1000倍

我们的实验

- 逻辑回归预测
  - 5分钟处理1万数据
  - 相比直接使用sklearn慢300倍
- 一亿个0到100的浮点数相加
  - 860毫秒；8倍明密文扩展

HOW DOES IT WORK

Ring-LWE问题

多项式环 $R=Z_q[x]/(x^n+1)$

给定:

$a_1, b_1=a_1 \cdot s+e_1$

$a_2, b_2=a_2 \cdot s+e_2$

...

$a_k, b_3=a_k \cdot s+e_k$

为了便于理解，可认为所有变量为整数

要求找到: s

s是R上的随机值

e_i 非常小

判定性Ring-LWE问题

多项式环 $R=Z_q[x]/(x^n+1)$

给你:

a_1, b_1

a_2, b_2

...

a_k, b_k

挑战: 是否存在s 和足够小的

$e_1, \dots, e_k$ 满足$b_i=a_i \cdot s+e_i$

BFV方案密钥生成

私钥生成:

随机抽样私钥 $s \in \chi$

公钥生成(s):

随机抽样 $a \in R_q$, $e \in \chi$

$pk_0 = -(a \cdot s + e)$　　$pk_1 = a$

Ring-LWE对
s 无法被解出

BFV加密

encrypt(m): 抽样 $u \in R_q$, e_1 $e_2 \in \chi$

$c_0 = pk_0 \cdot u + e_1 + \Delta \cdot m$, $c_1 = pk_1 \cdot u + e_2$

替换 pk为 $(-(a \cdot s + e), a)$

$c_0 = -(a \cdot s + e) \cdot u + e_1 + \Delta \cdot m$, $c_1 = a \cdot u + e_2$

$c_0 = -w \cdot s + e_1 + e \cdot u + \Delta \cdot m$, $c_1 = w + e_2$

判定性Ring-LWE 对 (无法与随机值区分)
可认为消息被随机值混淆

换一种形式看密文:

$f(x) = c_0 + c_1 * x$

BFV解密

Decrypt(c):

$f(x)=c_0+c_1 \cdot x$

$f(s)=c_0+c_1 \cdot s$

替换 x 为 s

替换 c 为 $=([-w \cdot s +e_1+e \cdot u+ \Delta \cdot m]_q, [w + e_2]_q)$

$f(s)=-w \cdot s +e_1+e \cdot u+ \Delta \cdot m + (w + e_2) \cdot s$

$=e_1+e \cdot u+e_2 \cdot s+ \Delta \cdot m$

远小于Δ
可恢复出m

简单结论:

$f(s)= v + \Delta \cdot m$

其中 v 远小于 Δ

同态加法

同态加法:

密文1: $f_1(x)$

密文2: $f_2(x)$

相加: $f_3(x)=f_1(x)+f_2(x)$

因为:

$f_1(s)=v_1+\Delta \cdot m_1$

$f_2(s)=v_2+\Delta \cdot m_2$

那么$f_3(x)$:

$$f_3(s)=f_1(s)+f_2(s)=v_1+v_2+\Delta \cdot (m_1+m_2)$$
$$=v_3+\Delta \cdot (m_1+m_2)$$

同态乘法

同态乘法:

密文1: $f_1(x)$

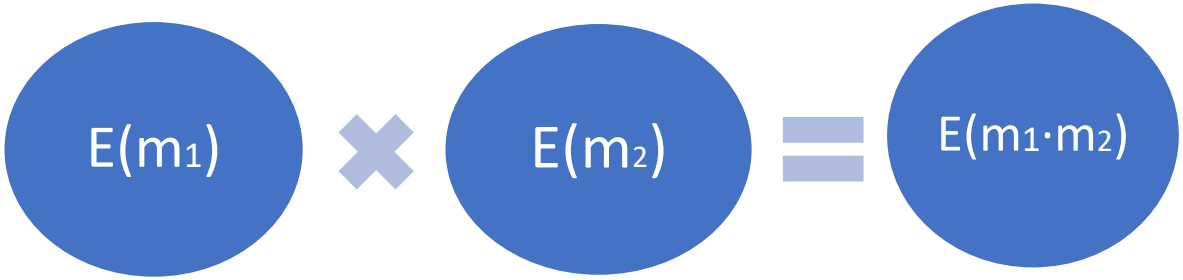
密文2: $f_2(x)$

相乘: $f_3(x)=f_1(x)*f_2(x)$

因为:

$f_1(s)=v_1+\Delta \cdot m_1$

$f_2(s)=v_2+\Delta \cdot m_2$



那么f3(x):

$f_3(s)=f_1(s)*f_2(s)=v_1 \cdot v_2+\Delta \cdot (v_1 \cdot m_2+ v_2 \cdot m_1)+\Delta_2 \cdot m_1 \cdot m_2$

除以 Δ :

$f_3(s)/\Delta= v_3+\Delta_2 \cdot (m_1 \cdot m_2)$

以上是简化版本的BFV算法

但足够让我们理解全同态的问题

更多BFV算法的细节

Brakerski, Z.: Fully homomorphic encryption without modulus switching fro classical gapsvp. In: CRYPTO 2012 - Volume 7417. pp. 868–886 (2012)

Fan, J., Vercauteren, F.: Somewhat practical fully homomorphic encryption. Cryptology ePrint Archive, Report 2012/144 (2012)

简化版的BFV:

https://github.com/edwardz246003/danger-of-using-homomorphic-encryption/blob/master/BFV.py

BFV方案安全问题

将消息m加密为多项式f(x)

将s带入来进行解密

$f(s)=v+\Delta \cdot m$

消息被Ring-LWE对混淆

能够区分密文 → 能够解决 Ring-LWE问题

可证明安全: IND-CPA → Ring-LWE

选择明文攻击

如果能再IND-CPA模型下攻破BFV，则能攻破Ring-LWE

IND-CCA?

选择密文攻击

攻击者能够访问解密机

BFV 并不能抵抗IND-CCA攻击

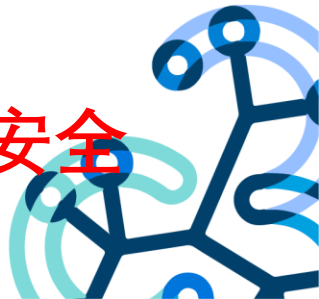
所有的实用全同态方案都不能抵抗IND-CCA攻击

全同态的性质看上去与IND-CCA冲突

关于IND-CCA的全同态方案的理论研究

Chosen-Ciphertext Secure Fully Homomorphic Encryption

没有全同态密码方案能够在IND-CCA模型下保证安全



IND-CCA

为什么需要IND-CCA

在很多场景中，攻击者能访问解密机

IND-CCA目前已经是加密算法的标准需求

全同态密码的应用场景通常更需要IND-CCA安全

外包计算看似处于CPA模型

用户和云端之间有丰富的数据流动

多方参与和数据交换

任何解密后的数据流向云端：

将打破CPA模型，需要 CCA 安全！

单次查询攻击

假设攻击者可以查询解密机一次

这在很多场景中合理

攻击者要求解密恶意密文f(x)

$f(x)=c_0+c_1x$ 其中 $c_0=0$, $c_1=\Delta$

解密机带入s

得到: $f(s)=\Delta s$

得到密文： s (密钥)

攻击者可通过单次解密查询，恢复密钥

非常危险

其他的全同态方案面临相同的问题

https://eprint.iacr.org/2014/535

攻击示例

```
def recover_key():
    cc0=0
    cc1=delta
    print "Recover private key successfully:", (s).list()==decrypt([cc0,cc1]).list()
    print "Secret key:",Roundt(s).list()
    print "recovered key",decrypt([cc0,cc1]).list()

s,pk=gen()
rlk,E=gen_rlk(T)
recover_key()
```

```
Recover private key successfully: True
Secret key: [1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 0
0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1,
0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1,
0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1,
0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1,
0, 1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0, 1,
0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0,
1, 1, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0,
0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0,
1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0,
0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0,
1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 1,
0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 1, 0,
1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 0,
1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 1, 1,
```

https://github.com/edwardz246003/danger-of-using-homomorphic-encryption/blob/master/CCA_attack.py

解决办法

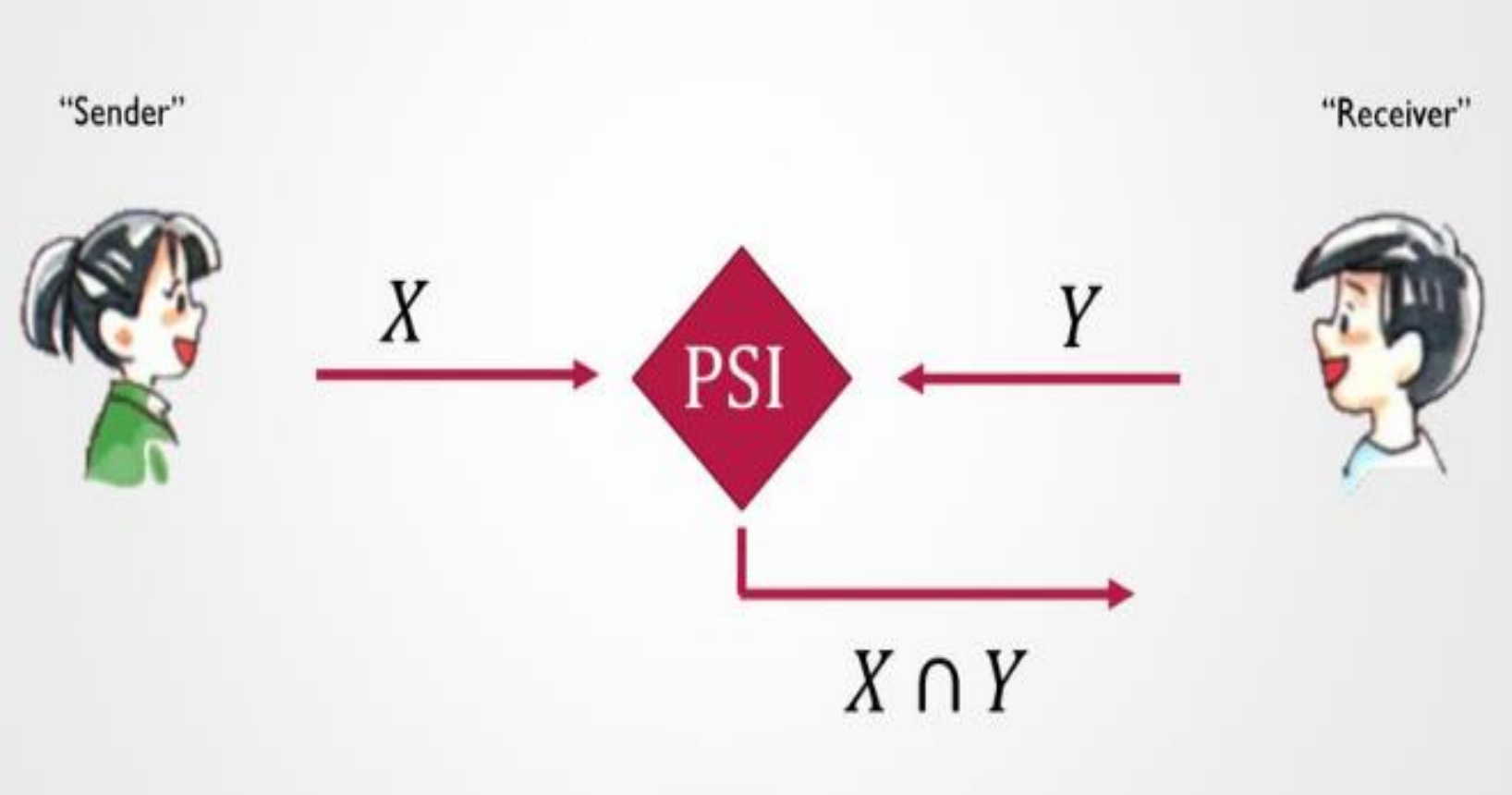
不要在任何明文数据会泄露给操作者的场景中使用全同态密码

否则，相当于没有加密。

如何才能确保没有数据泄露?

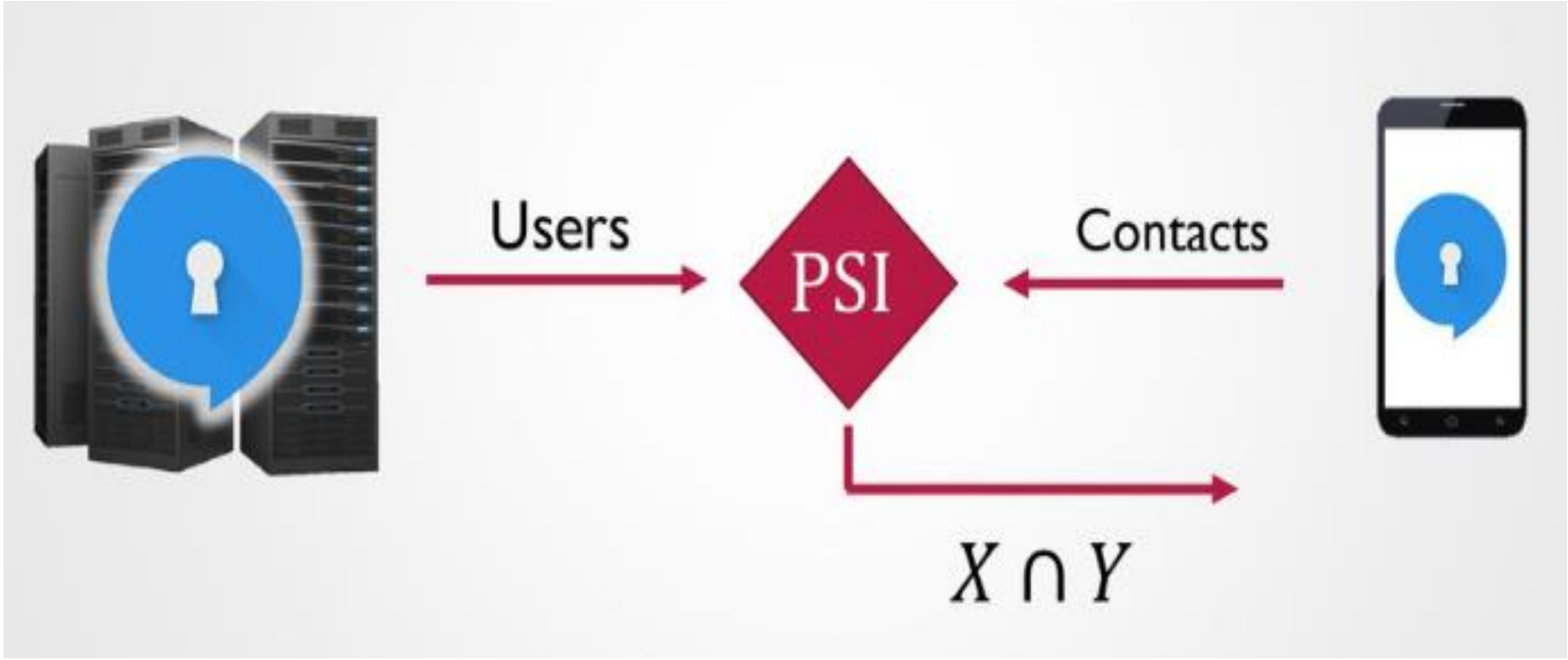
SEAL中将加入新的缓解措施。

私有集合交集计算(PSI)



不泄露任何其他信息

应用：私有通讯录匹配



私有通讯录匹配在端到端加密的通讯软件上

如signal

使用同态密码构造PSI（CCS17）

本地数据库Y

加密通讯录X

发送加密后的X给服务器

与本地数据库Y，进行同态计算
得到加密后的X∩Y

发送加密后的X∩Y 给用户

解密得到X∩Y

该场景下的CCA攻击

用户得到X∩Y后
发现X∩Y 在使用signal
添加他们为好友

将X∩Y 明文发送给服务器

信息泄露
服务器可发起CCA攻击!

客户与服务器之间总是有很多不经意的数据流
使用全同态密码要十分小心

1比特信息泄露的后果

与之前的CCA attack

用户会检查解密结果是否为0

只会泄露1比特信息给服务器

可以使用1比特信息泄露恢复1比特密钥

攻击者无需查询解密机

任何1比特信息泄露，将转换成1比特密钥泄露

全同态密码方案的安全性将指数下降

1比特信息泄露在显示生活中不可避免

攻击实例（1比特信息泄露）

```
def recover_key(i):
    t1=[0 for _ in range(d)]
    t1[i]=M
    t2=M
    cc0=pk[0]+R(t1)
    cc1=pk[1]+R(t2)
    return (decrypt([cc0,cc1]).list())[i]

s,pk=gen()
rlk,E=gen_rlk(T)
M=delta//4+50
Recoverd_key=[]
for i in range(d):
    Recoverd_key.append(recover_key(i))
print "Recover private key successfully:",Recoverd_key==s.list()
```

```
Recover private key successfully: True
```

https://github.com/edwardz246003/danger-of-using-homomorphic-encryption/blob/master/CCA_attack.py

其他攻击

本地数据库Y

使用全同态加密X

发送加密的X给服务器

计算得到加密的X∩Y .
大多数HE没有

发送加密的X∩Y给客户

客户解密得到X∩Y
也会得到Y的信息

```
ma=randint(0,t-1) # Alice's input
mb=randint(0,t-1) # Bob's input
print "Alice has input",ma
print "Bob has input",mb
#Alice encrypt her input
#Alice keep u, e1 , e2
u=sample_2()
e1=sample_e()
e2=sample_e()
ca=(Roundq(pk[0]*u+e1+delta*ma),Roundq(pk[1]*u+e2))
#Then Alice send the ciphertext ca to bob

#Bob receive ca, and do some homomorphic computation.
cab=(Roundq(ca[0]-delta*mb),Roundq(ca[1])) # plus const mb
r=randint(0,t)
print "Bob choose a random factor,r:",r
cab=(Roundq(r*cab[0]),Roundq(r*cab[1])) # mul a random number r
#Bob respond cab back to Alice

result=decrypt(cab)
print "is mb==mb?",result==0

# A semi-honest Alice can recover the r by using
for i in range(t):
    if Roundq(i*ca[1])==Roundq(cab[1]):
        break
r_prime=i
print "Alice recover r':",r_prime
print "Is r' equals to r:",r_prime==r

#Alice can use r to recover Bob's input mb
mb_prime=(ma-(int(result)*inverse_mod(r_prime,t)))%t
print "Alice recover mb'",mb_prime
print "Is mb' equals to mb:",mb_prime==mb
```

SEAL的电路隐私

SEAL 不提供电路隐私

SEAL手册中有提及

最佳实践：噪音混淆

给计算结果添加足够的噪音

SEAL中没有噪音混淆的标准接口

普通程序员无法实现噪音混淆

提供噪音混淆的困难性

需要知道我们到底需要多少噪音，这实际上也是需要保护的信息:(

电路隐私问题目前是全同态密码方案的通用性问题

没有通用性的解决方案

解决办法

修正版的PSI 协议(CCS2018)

https://eprint.iacr.org/2018/787

解决PSI问题（非通用性解决方案）

对于全同态密码的电路隐私问题

目前需要密码专家来审计具体实现

格密码方面的专业知识

SEAL 团队考虑提供一个通用接口

SEAL中的编码信息泄露

全同态方案在多项式环上工作

要处理的明文通常是：数字、字符串

需要将他们转换到多项式环上

SEAL的编码方案(IntegerEncoder)

将整数编码到多项式环上

多对一映射

信息泄露!

整数编码问题示例


```
m1=IntegerEncoder(2)
m2=IntegerEncoder(2)
c1=encrypt(m1)
c2=encrypt(m2)
result=add(c1,c2)
print decrypt(result)
print IntegerDecoder(decrypt(result))
```

```
2*X
4
```

```
m1=IntegerEncoder(1)
m2=IntegerEncoder(3)
c1=encrypt(m1)
c2=encrypt(m2)
result=add(c1,c2)
print decrypt(result)
print IntegerDecoder(decrypt(result))
```

```
X + 2
4
```

百万富翁问题：
https://github.com/edwardz246003/danger-of-using-homomorphic-encryption/blob/master/millionaire.py

解决办法

编码问题也影响其他全同态方案

密码方案可能有安全证明, 但是编码方案没有

不要使用SEAL的整数编码器(不理解安全模型时)

IntegerEncoder 只能认为是一个实例工具

可以使用BatchEncoder 和 CKKSEncoder

其他安全问题

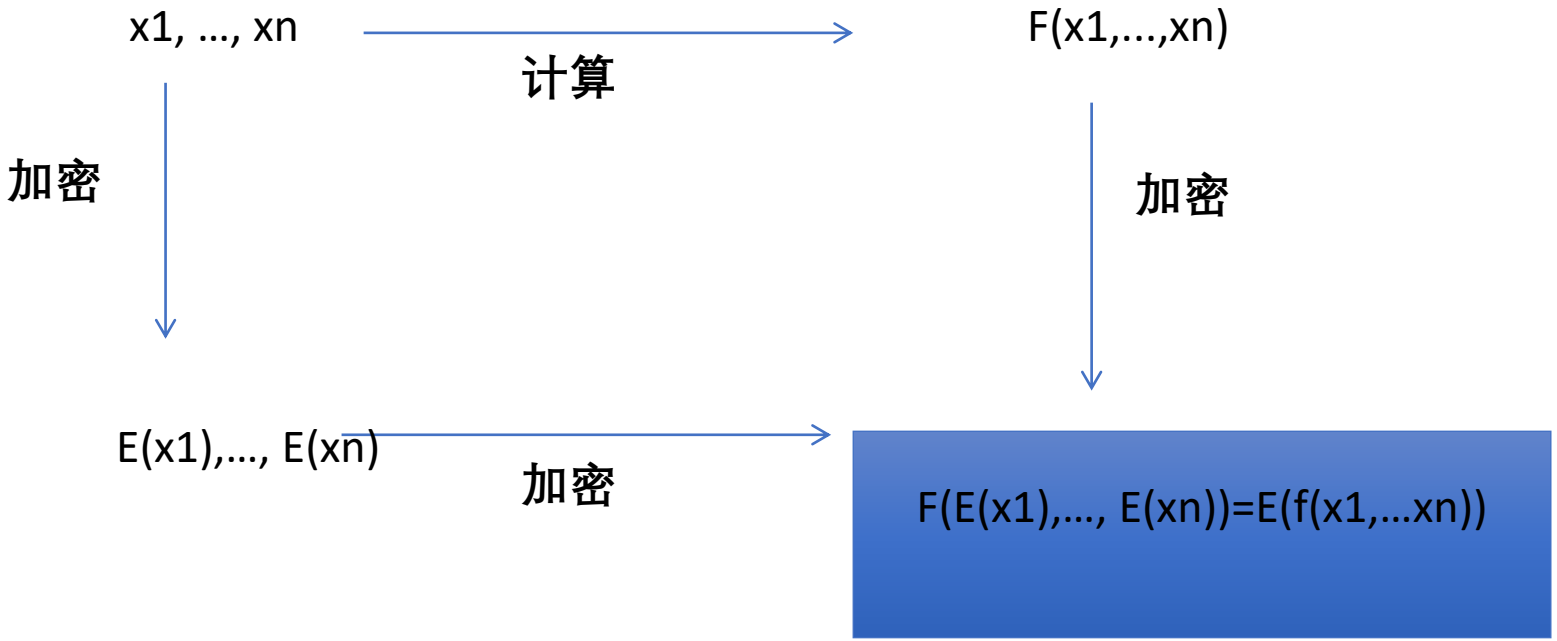
全同态加密方案不能提供常见的加密安全属性

- 全同态加密方案不是可认证加密方案
- 不能保证数据的完整性
- 攻击者可以利用全同态性质修改数据内容
- 不要使用全同态加密直接进行数据传输、数据存储

全同态密码需要标准

- 微软目前在做相关标准
- 建议将安全协议相关内容纳入

全同态可以计算任意函数？



任意函数在全同态中意味着：任意的加法和乘法

任意的加法和乘法，并不表示你可以在数据上运行任意程序

你不能直接进行数据对比

（不支持if语句）

更新比喻

1. 把金子放进不透明的箱子
2. 锁住箱子，藏起钥匙
3. 让工人隔着手套操作(蒙上眼罩)
4. 解锁箱子，得到珠宝

总结

全同态密码具有很好的应用场景

近年来性能提高了很多

全同态密码并非万能的

它不能计算任意的函数

使用全同态密码会有很多安全隐患

现阶段没有安全易用的基础库

实现需要被密码专家审计

实际应用比学术论文的模型更复杂，安全问题更严峻

密码社区需要更多关注具体场景中的安全问题

致谢

感谢

Kim Laine of Microsoft Research

Chen Hong of Alibaba Gemini Lab

对本议题的帮助与建议