感谢主持人对我的介绍

Thank you for the introduction.

此工作是我与来自贝尔实验室的Vladimir Kalashnikov所共同完成的

This is a joint work with Vladimir Kalashnikov from Bell Labs.

我们工作针对的目标是安全计算 这是密码学上一个非常典型、非常普遍的问题

So, our work is motivated by secure computation, which is the most general problem in cryptography.

当前有很多重要的研究进展 尝试让安全计算从理论走向实际

There's a major research effort currently, which is trying to move secure computation from theory to practice.

这些工作的目标不仅是要提高协议的渐进效率 还要提高协议的真实执行效率

The focus is not only on improving the asymptotic efficiency, but also on the concrete efficiency.

还有一些工作的方向是具体协议的实现 解决系统层面的问题

And work in this area also addresses implementation as well as systems issues.

在过去的5年 无论是理论角度还是实际角度 此领域都诞生了极多的突破性成果

In the last 5 years, they've been some tremendous amount of breakthrough results in theory as well as in practice.

从理论角度 我们得到了很多惊人的结论

On the theory side, we have this amazing result,

我们现在只需要引入常数级额外开销 即可完成安全计算或安全函数求值

which lets you to do secure computation with just constant overhead or in secure evaluation.

另一个突破是全同态加密

Another breakthrough is also fully homomorphic encryption.

对于多种类型的函数 我们甚至可以在最优通信复杂度下构建安全计算协议

We can even obtain secure computation protocols with optimal communication overhead for broad class of functions.

还可以使用基于ORAM的安全计算协议 利用RAM计算模型的优势完成安全计算

We can also use ORAM-based secure computation protocols to take advantage of the RAM model of computation.

这样我们就能在次线性时间复杂度下完成安全计算

And you can also do perform secure computation in sub-linear time.

在实践角度 针对姚氏协议和GMW协议 学者们提出了很多算法或实现层面的优化

On the practical side, there have been a lot of algorithmic as well as implementation level improvements for Yao and GMW-based protocols,

也尝试混合使用姚氏协议和GMW协议

and also a hybrid of Yao as well as GMW protocols.

特别地 姚氏协议的实现结果令人印象深刻

And time results for implementing Yao’s protocols especially is very very impressive.

现在 我们可以在637毫秒内得到AES电路的乱码电路

Now you can garble an AES circuit in 637 microseconds.

虽然在实现层面和理论层面 我们都得到了令人惊讶的成果

So, in spite of the major interest in implementations and as well as the these amazing results on the left side,

但在右侧 效率最高的协议仍然是20世纪80年代提出的姚氏协议和GMW协议

our fastest implementations in the right still make use of Yao and GMW protocols from the 80s, right?

之所以这样 有一个很直观的原因

So, one reason is obvious.

理论成果虽然引入常数复杂度 但这个常数可能非常大

Maybe in these theoretical results, the constants are very high.

实际应用时 可能存在这样一个复杂度与效率的层级关系

But maybe there's also a hierarchy of complexity or efficiency.

FHE的复杂度目前、以后也总会比公钥密码学原语的复杂度高几个量级

Namely the cost of FHE is and will probably always be by orders of magnitude bigger than the cost of public key primitives.

公钥密码学原语的复杂度目前、以后也总会比对称密码学原语的复杂度高几个量级

And the cost which in turn is and will probably always be by orders of magnitude bigger than the cost of symmetric key primitives,

对称密码学原语的复杂度目前、以后也总会比一次一密的复杂度高几个量级

which in turn is and will probably always be by orders of magnitude bigger than the cost of just one-time pad, OK?

本次演讲的主题是OT扩展 其出发点就是减小公钥操作和对称密码操作的效率鸿沟

So, the topic of the talk OT extension is motivated by this difference in efficiency between public key operations and symmetric key operations.

在第一部分演讲中 我会详细介绍OT扩展的研究出发点 并解释OT扩展面临的问题

So, in the first part of the talk, I will give a more detailed motivation about the OT extension, and I will also explain the problem.

诸如密钥协商、不经意传输等公钥密码学原语一般都很难高效实现

So, public key primitives such as key agreement or oblivious transfer are usually hard to implement heuristically.

公钥密码学原语天生要依赖于某类代数结构 因此也会遭受很多密码学攻击

They're typically tend to be algebraic in nature, so there are more and more cryptanalytic attacks.

因此 相应的参数需要设置得比较大

So, the parameter has to be bigger.

这导致公钥密码学的计算复杂度也相对较高

And they usually end up more expensive.

另一方面 我们更容易实现伪随机数生成器或哈希函数等对称密码学运算

On the other hand, for symmetric key operations, like pseudo-random generator or hash functions, is typically easier to implement heuristically.

学者们设计出了很多算法

There are a lot of candidates.

相应的参数要比公钥密码学小得多

And the parameter is typically smaller.

实际中也可以更容易、更轻量级地实现对称密码学运算

And therefore, they are also cheaper in practice to implement.

这个结论背后还有相应的理论支撑

This is also backed by theory,

理论上 无法通过黑盒方式应用对称密码学原语构造大多数公钥密码学原语

which says the broad class of public key primitives cannot be black box reduced to symmetric key parameters in general.

这意味着公钥密码学原语和对称密码学原语的性能差约为3至4个数量级

In concrete terms, what this means is that we have a factor of 3 to 4 orders of magnitude difference in efficiency between public key primitives and symmetric key primitives.

对于诸如AES等被广泛使用的特定对称密码学操作

And in the particular case of symmetric key operation such as AES, which are widely used,

Intel专门提供了对应的指令集 进一步提高了这类运算的执行效率

Intel provides an instruction set, which makes them even faster in practice.

因此 我们无法通过对称密码学原语实现公钥密码学原语 那接下来该怎么办？

So, we cannot do public key primitives from symmetric key primitives, but what is the next best thing?

或许可以用少量公钥原语实例和大量对称密码学操作 生成大量公钥实例

Maybe just with a few instances of public key primitives and a lot of symmetric key operations, maybe we can generate many many instances of public key operations, right?

这一技术称为“扩展原语”

So, this is known as extension of a primitive.

我们已经知道 很容易扩展公钥加密体制

And we know that extending public key encryption is easy.

具体过程是用对称加密算法加密具体的明文 用公钥加密算法加密对称加密的密钥

And this process is just by encrypting the payload with the symmetric key, and then you encrypt the symmetric key with the public key.

这样一来 我们只需要执行一次公钥操作

So, you only do one public key operation.

这一技术在我们每天使用的加密过程中起到了非常重要的作用

So, this has had a huge practical impact on our everyday use of encryption.

我们可以很自然地把这个问题展开 是否可以扩展其它公钥密码学原语 如OT？

So, this naturally raises the question, what about extending other public key primitives, such as OT, or oblivious transfer.

回忆一下不经意传输要解决的问题 发送方有两个输入x\_0和x\_1 接收方有一个输入r

So, recall in the problem of oblivious transfer, there's the sender with 2 inputs x\_0 and x\_1, and there is a receiver with input r.

协议执行完毕后 发送方无法得到任何信息

At the end of the protocol, the sender does not learn anything,

接收方可以得到与其选择比特关联的发送方输入

but then the receiver learns the sender’s input which corresponds to the selection bit.

OT是SFE的基础构建模块

So, OT is a very fundamental building block in SFE.

姚氏电路中应用不经意传输实现了乱码密钥的2选1过程

In Yao, it is used to select 1-out-of-2 garbled keys in an oblivious fashion.

在GMW协议中 OT的用途更加广泛 每个AND门的求值过程都要使用一次OT协议

In GMW, it's more intensively used, where you evaluate each AND gate in the circuit using an OT protocol.

我们知道OT的执行开销非常大

So, we know that the cost of OT is significantly higher.

我们无法通过对称密码学操作实现OT

You cannot get it from symmetric key operations.

但假设我们有一个可以传输短字符串的OT

But suppose we have OT on short strings,

通过使用标准的伪随机数生成器 我们能得到可以传输长字符串的OT

then by a standard application of pseudo-random generators, we can get OT on long strings.

这一技术称为“OT长度扩展”

So, this is called the OT length extension.

还有一个更难解决的问题 称为“OT实例扩展” 简称“OT扩展”

The more non-trivial problem which is OT instance extension, I'll just call OT extension,

幸运的是 我们知道OT扩展是可行的

which we know is fortunately possible.

我们只需要k个公钥密码学操作 即k个种子OT 再加上n次对称密码学操作

We only need k public key operations, like k seed OTs, and additional n symmetric key operations,

应用这k次公钥密码学操作 我们就可以执行任意多项式大小次的OT操作

and this is to perform an arbitrary polynomial number of OT operations just using this k public key operations.

这一技术大幅降低了公钥密码学运算次数 对SFE的实际应用起到了重要的作用

Because we cut down on the number of public key operations, this has naturally had a huge practical impact on the on feasibility of SFE.

近期大多数SFE的实现都应用了OT扩展协议 从而提高协议的执行效率

For the extent, all recent implementations rely on OT extension for efficiency.

Beaver在1996年提出了第一个OT扩展协议

So, Beaver in 1996 gave the first OT extension protocol.

第一个高效的OT扩展协议由Ishai、Kilian、Nissim和Petrank给出

The first practical OT extension protocol was given by Ishai, Kilian, Nissim and Petrank,

此协议也称为IKNP协议

which will call IKNP construction.

对应的论文发表在CRYPTO 2003上

This was given in CRYPTO 2003.

后续 学者们期望在恶意攻击场景下提高OT扩展协议的执行效率

Later, ones have looked at improving the efficiency of OT extension in the malicious case,

学者们同时也在加深对OT扩展协议的理解 从而知道OT扩展协议的上限是什么

and also on improving our understanding of OT extension and when it's possible.

在本工作中 我们将在半可信场景下提高IKNP协议的效率

So, in this work, we will improve the IKNP construction for the semi-honest case.

我们给出了协议的渐进优化方法和实际优化方法

We will provide both asymptotic as well as concrete improvements.

在接下来的讲座中 我们会描述Ishai等人的OT扩展协议构造

So, in the next part of the talk, we will describe Ishai et al. construction of OT extension.

实际上 我们将直接使用Ishai等人CRYPTO 2003的演讲幻灯片

In fact, we will use Ishai et al.’s slides from CRYPTO 2003 to do this.

IKNP第一个、也是最重要的步骤是将n个OT归约为传输n比特字符串的k个OT

The first and the main step of IKNP is to do a reduction from n OTs to k OTs, but on n bit strings,

这一步骤将引入额外的、线性数量级的对称密码学操作

which will also incur an additional linear number of symmetric key operations.

下一步是长度扩展步骤 我们之前已经讲解过这一步骤了

And then, the next step is the length extension case, which we already saw before.

这可以让我们把长字符串OT协议归约为短字符串OT协议

So, this allows us to reduce OTs on long strings to OTs on short strings.

这一步骤进一步引入了线性数量级的对称密码学操作

And this would also incur an additional cost of linear number of symmetric key operations.

在核心归约步骤中 我们让接收方选择一个随机的n×k矩阵T

Right, so in the main reduction, we will let the receiver pick a random n by k matrix T.

随后 发送方选择一个随机的行向量s

And then, the sender picks a random row vector s.

接下来 接收方和发送方执行k个OT协议 但此OT协议中两个参与方的角色互换

And then, the receiver and the sender will participate in k instances of OTs, except their roles will be reversed.

在每个OT协议中 接收方要选择长度为n的两列比特值

So, in each instance, the receiver is going to pick 2 columns of length n,

每对比特值中 第一列为矩阵T中的某一列

such that the first column of each pair is exactly the pair corresponding to the matrix T.

第二列为第一列比特值与选择向量r的异或结果

And the second column will be the first column xored with the selection vector r.

发送方实际上应用它随机选择的行向量在接收方的两列比特值中选择一列

And then, the sender will actually choose 1-out-of-these-2 columns using his random row selection vector.

这样 发送方通过OT协议得到了一个矩阵Q

Then, the sender obtains a matrix Q as the output of this OT protocol.

我们来看看矩阵Q满足何种性质

And let's look at how the row of this matrix Q looks like.

如果r\_i=0 则q\_i=t\_i

When r\_i=0, q\_i will equal t\_i.

如果r\_i=1 收到的每对比特值就不太一样了

And when r\_i=1, in each pair, the the bits will be different.

如果r\_i=1 在IKNP协议中q\_i=t\_i⊕s

So, in this case, q\_i will equal t\_i⊕s, right?

注意到在第一种情况下 接收方知道t\_i 但无法知道t\_i⊕s

So, in this case, note that the receiver knows t\_i, but it will not know t\_i⊕s.

因此 在第一种情况下 接收方能得到q\_i 但无法得到q\_i⊕s

So, in the first case, you will know q\_i, but not q\_i⊕s.

在第二种情况下 接收方能得到q\_i⊕s 也就是t\_i 但无法得到q\_i

And in the second case, he will know q\_i⊕s, which is t\_i, but not q\_i.

这意味着我们或许可以使用q\_i和q\_i⊕s作为OT协议中的数据加密密钥

So, this turns out that maybe we can use q\_i as well as q\_i⊕s as masks for the OT.

但需要注意的是 我们必须要破坏矩阵中q\_i和q\_i⊕s的相互关系

But then, we have to destroy correlations between different rows of this matrix.

我们应用随机预言机H来破坏q\_i和q\_i⊕s的相互关系

So, we use the random oracle H to destroy these correlations, right?

最后 接收方根据t\_i选择得到它的输出 也就是应用t\_i进行解密

And then, the receiver will pick his output depending on each of t\_i. This is how we unmask things.

IKNP协议非常简单、非常优雅、效率极高

So, the IKNP protocol is very simple and elegant. And so, it is very efficient.

我们考虑n个OT协议所需要的通信开销 其中发送方输入的长度为L

So, if you look at the communication cost of n instances of OT, where the sender inputs are of length L,

大家已经了解到 在核心归约步骤就是对x\_(i,0)和x\_(i,1)加密

then in the main reduction, which you just saw before, is just hashing the x\_(i,0) and x\_(i,1).

这需要发送2nL比特的数据

So, this will be 2nL bits.

在长度扩展步骤中 我们要应用一个PRG 这需要发送2nk比特的数据

Well, as the length extension, you choose the PRG. It would be 2nk bits.

在姚氏电路中 我们需要传输长度为L=k的密钥

So, in Yao, we need to transfer keys of length L=k.

因此核心归约步骤和长度扩展步骤中的通信开销相同

So, the main reduction and the length extension are exactly equal.

在GMW中 我们只需要传输L=1的信息

So, in GMW, we only need to transfer L=1.

令人惊讶的是 这一场景下长度扩展步骤的通信开销远高于核心归约步骤的通信开销

But in this case, somewhat surprisingly, the cost of the length extension is far more dominant than the cost of the main reduction.

这就是一个问题了 我们可能可以在这一场景下对通信开销进行优化

So, this is the question. Maybe there's some hope for improvement here, right?

在讲座的后半部分 我们会提出IKNP的通用框架

So, in the next part of the talk, we will propose a more general framework for IKNP.

我们也会向大家展示如何提高IKNP的效率

And we will also show how to improve the efficiency of IKNP.

我们先来详细分析一下IKNP协议

So, as a starting point, we take a closer look at IKNP.

可以看到 接收方要选择这个n×k的随机矩阵

And we see that the receiver is going to select this n by k matrix, which is a random matrix.

随后 接收方要生成另一个矩阵

And then, it is going to generate this other matrix, which is just…

这个矩阵的第i列为第一个矩阵的第i列异或选择向量r

I mean the i-th column here will be the i-th column here xor the selection vector r, right?

换句话说 U= T⊕R 其中R是所有列均相等的矩阵 每个列都为接收方的选择向量

So, in other words, U is exactly equal to T⊕R, where R is a matrix in which all those are exactly identical, which is equal to the selection vector of the receiver.

如果我们从行的视角看 就会发现R的第i行为k个r\_i

Right, if we switch to the dual view, then we see that in the i-th row of R, we see that it contains k copies of r\_i.

这意味着R的每一行都是r\_i的一种编码

So, this means as the row-wise encoding going on.

在IKNP协议中 0被映射为0^k 而1被映射为1^k

In particular, 0 maps 0^k, and 1 maps 1^k.

因此 我们可以看到IKNP这一高效协议在底层执行了一次逐行重复编码

So, we see that the efficient protocol IKNP at some basic level is using row-wise repetition encoding, right?

这自然引出了一个问题：我们是否可以使用更复杂的编码？

So, it naturally raises the question, can we use more sophisticated encodings?

毕竟 重复编码是一种最简单的编码

After all, repetition encoding is as low as it could.

假设我们使用编码C 并且我们假定r\_i属于一个很大的域 域为从1到m

So, suppose we use the code C. And we assume that r\_i comes from a larger domain, say 1 through m.

我们用编码C将r\_i映射成C(r\_i) 这是一个k比特长字符串

Then we use the code C to map r\_i to C(r\_i), which is now a k-bit strings.

现在 接收方需要用选择比特r\_i构建矩阵C(R)

So now, let's say that the receiver, using his selection r\_i, is going to construct this matrix C(R). Let’s call it C(R).

我们来看看 在这个理论框架下IKNP协议的执行过程

And then, let's see how the IKNP protocol works in this case.

第一步 接收方获得了矩阵C(R)

So, in the first step, he's going to take the matrix C(R).

随后 接收方用加法秘密分享方案将C(R)分享为T和U 即C(R)= T⊕U

And then, he's going to additively share it as T and U. So, it is T⊕U.

接下来 接收方和发送方角色互换 执行k个OT协议

And then again, he's going to participate acting as the sender in k instances of OT.

在第i个OT中 接收方的输入是t^i和u^i 即T和U的第i列

Now, in the i-th OT, he will use t^i and u^i. These were the columns of T and U respectively.

执行完OT协议后 发送方得到矩阵Q

So, once again, the sender will obtain the result of the OT as a matrix Q.

在IKNP协议中 我们知道q\_i或者等于t\_i 或者等于t\_i⊕s

And in the earlier case, in the IKNP case, we saw that q\_i is either t\_i or t\_i⊕s.

在这一理论框架中 我们可以知道q\_i=t\_i⊕(C(r\_i)⊙s)

So, in this case, what happens is that we will have q\_i=t\_i⊕(C(r\_i)⊙s).

结果不算太复杂

It's not terribly complicated.

我们可以验证一下 当C是重复编码时 此框架对应的协议就是IKNP协议

So, when C is the repetition code, as a sanity check, we can see that it is exactly the same as IKNP.

特别地 当r\_i=0时 C(r\_i)是一个全0向量 因此我们得到q\_i=t\_i

In particular, when r\_i=0, C(r\_i) will be the all 0’s vector. So, you will have q\_i=t\_i.

当r\_i=1时 C(r\_i)是一个全1向量 此时我们可以得到q\_i=t\_i⊕s

And when r\_i=1, then C(r\_i) will be the all 1’s vectors, in which case you will have q\_i=t\_i⊕s, right?

这样一来 我们得到了m个密钥 后面的执行过程就完全一样了

So now, you will have m masks, which are generated in the same case as before.

密钥生成算法为q\_i⊕(C(r)⊙s)

The mask generator is q\_i⊕(C(r)⊙s).

我们仍然可以证明接收方只能知道t\_i

And once again, we can show that the receiver will know only t\_i.

因此 接收方只能解密对应的密文 从而得到对应的输入

And therefore he will be able to unmask the right value and get the correct standard input.

核心归约步骤在恶意发送方的攻击下是完美安全的

So, again, the main reduction is perfectly secure against the malicious sender.

特别地 恶意发送方只能得到矩阵Q 这是编码的随机独立分享结果

In particular, the malicious sender gets only the matrix Q, which is just a random independent share of the encoding.

核心归约步骤在半诚实接收方的攻击下是统计安全的

Against a semi-honest receiver, we can get statistical security.

这是因为除了在加密过程中使用了随机预言机之外 核心归约步骤没有安全性损失

This is because there is no loss in security unless the random oracle is queried on one of these other pairs, which are used for masking, right?

因此 整个协议的安全性损失为m 即r\_i的取值范围

So, the loss in security will be m, which is the number of codes,

以及2^(-d) 其中d是线性编码C的最小距离

and then times 2^(-d), where d is the minimum distance of the linear code C.

注意到在此理论框架下 我们可以从1到m选取消息

So, note that in the case, we had our selection coming from 1 through m.

因此从效果上看 我们可以实现m选1-OT 而不是2选1-OT

So, in effect, we are actually doing 1-out-of-m OT instead of 1-out-of-2 OT。

但在这种情况下 核心归约步骤的通信开销会从2nL提高到nmL

But then, we increased the the cost of the main reduction, which was 2nL, to nmL.

随后 我们将标准的2选1-OT转换为n/log(m)个字符串长度稍长的m选1-OT实例

And then, we use a standard transformation of 1-out-of-2 OT to n/log(m) instances of 1-out-of-m OT, on slightly longer strings.

这也允许我们将通信开销表示为与m相关的函数

And this allows us to express the communication costs as a function of m.

现在 我们有了一个自由变量m 用它来平衡核心归约步骤和长度扩展步骤的开销

So now, we have a free variable m, which we can use to balance the cost of the main reduction and the length extension, right?

具体来说 如果我们使用的是最小距离为k/2的Hadamard编码

So, in concrete terms, if we use the Hadamard codes for encoding, it still has minimum distance k/2.

在这种情况下 2选1-OT的通信开销可以降低2倍

And therefore, for this case, we can show that we can get a factor of 2 improvements for 1-out-of-2 OT,

在多方GMW协议中 如果k=256 则通信开销也可以降低2倍

and also for the resulting MPC protocols like GMW for the multi-party case for k=256.

也可以进一步优化长度扩展步骤的通信开销 优化程度为…

There are additional optimizations especially in the length extension case, which are…

通信开销优化程度是算法层面的 而不是渐进层面的 相应的优化程度为…

This is algorithmic but not asymptotic, but this can lead to a factor…

结合Hadamard编码后 与未优化的IKNP协议相比 新协议的通信开销要降低3.5倍

I mean, combined with our Hadamard codes, this will lead to a factor of 3.5 improvement over the unoptimized IKNP protocol.

Asharov等人也独立发现了这一优化点 他们的论文将发表在CCS 2013上

I think this was also independently discovered by Asharov et al., which is going to appear in CCS later this year.

与IKNP相比 我们从渐进层面降低了每一个OT的通信开销

We also improve the asymptotic costs over IKNP per OT.

当L=1时 IKNP需要通信O(k)比特 而我们需要通信O(k/log(k))比特

So, IKNP uses O(k) bits for L=1. And we use O(k/log(k)) bits.

总结一下

So, wrap up.

为了平衡公钥密码学原语和对称密码学原语的性能鸿沟 学者们提出了OT扩展协议

We saw that OT extension is motivated by this difference in efficiency between public key primitives and symmetric key primitives.

这一协议在安全函数求值的实例落地中产生了巨大的影响

And this has had a huge impact on the practicality of secure function evaluation.

在本次讲座中 我们提出了IKNP的编码理论框架

So, in this talk, we propose the coding theory framework for IKNP.

可以在随机预言模型下证明此框架的安全性

It can be proven secure neither the random oracle model,

随机预言模型也可以换为特定类型哈希函数假设 即相关性健壮哈希函数

or which we use this special type of hash functions known as code correlation robust hash functions,

这沿用了IKNP中安全性所依赖的相关性健壮哈希函数假设

which are a generalization of the IKNP correlation robust hash functions.

当使用复杂编码时 此框架提高了多方GMW中2选1-OT和m选1-OT的性能

As a result, we can get concrete improvements for multi-party GMW, for 1-out-of-2 OT, as well as for 1-out-of-m OT if we use more sophisticated codes.

我想用GMW和姚氏电路的性能对比问题作为讲座的结尾

So, I would like to end my talk by raising this issue of the efficiency of GMW vs. Yao.

近期的安全多方计算研究主要关注恶意模型下姚氏电路的性能优化问题

So, in many recent implementations, they implemented Yao, this is more so prevalent in a malicious case.

在半可信安全模型下 学者们也提出了很多姚氏电路的优化方法

And also in semi-honest case, there are more implementations of Yao.

但近期的一些工作也表明 GMW协议也有很多算法层面的优化点

But a recent line of work has shown that a lot of algorithmic improvements possible to GMW as well.

我们的工作适用于GMW协议

And our work also fits in this line of work.

谢谢大家

Thank you.