ShaftStop：一种高性能完全同态加密算法

郑珂威，范宏达，李立中

shaftstop.fhe@gmail.com

www.shaftstop.com

摘要

完全同态加密被称为“密码学圣杯”。这项技术可以直接计算密文，实现对数据隐私的全方位保护。已有的几种完全同态加密构造方法无法商用，主要原因在于较低的密文运算效率。ShaftStop 是一种全新的高性能完全同态加密算法，其安全性在最差情况下可以归结为二次多元方程组的求解难题（Multivariate Quadratic Problem），结合混沌密码（Chaotic Cryptography）设计思想强化了安全性，同时引入运算字典的概念，大大提高了密文运算的安全性和运算效率。本文详细介绍了 ShaftStop 算法的构造原理，分析了其安全性，并就该算法的典型应用场景进行了简要说明。

一、问题的提出

随着数字经济时代的到来，数据的价值越来越高，数据隐私遭遇的挑战也越来越严峻。

加密技术是最佳的数据隐私保护方法。现代密码学使用严谨的数学原理保证了数据一旦被加密成密文形式，除非使用密钥进行解密还原，否则任何人使用任何方法都很难窃取被保护数据的隐私。

然而，密文无法直接进行运算，这成了密码学的“阿喀琉斯之踵”。数据在传输和存储的过程中可以使用密文形式保障安全性，而使用时必需在明文状态下进行。这就给了攻击者

可乘之机。

设想有这样一种特殊的加密技术，其生成的密文支持任意的运算，那么受保护的数据就可以始终以密文形式存在，数据处理方无需密钥就能接受密文数据托管，实现数据存储、数据传输和数据运算整个过程安全可靠。

这种被称为“密码学圣杯”的技术就是同态加密（Homomorphic Encryption）。

同态加密的概念由 Rivest, Adleman and Dertouzos [Rivest78]提出，当时他们使用的术语是“privacy homomorphism”，即隐私同态。使用该技术可以在无需解密的情况下对密文直接进行运算，其结果等同于先对明文进行运算，再对运算结果进行加密。同态加密自提出后一直未能找到既满足安全性，又能同时支持灵活密文运算的算法。

直到 2009 年，Craig Gentry [STOC2009] 首度公开了一种基于理想格的同态加密算法，并正式使用了完全同态加密（Fully Homomorphic Encryption）的术语，该方法允许对密文进行任意操作。

至今，有三类主要的完全同态加密构造。

第一类构造是 Gentry 方案 [Gentry2009]。基于理想格，引入随机噪声用于保护明文信息，并在每一步运算之后执行 Bootstrappling 操作，控制噪声大小始终保持在不影响计算结果的范围内。针对明文的每一位(bit)进行加密，导致整体运算量巨大，同时其密文所需的存储空间也相当庞大。

第二类构造是 Brakerski-Vaikuntanathan 方案 [BV11]。基于 LWE（Learning With Errors）或环 LWE 假设，其密文和密钥均用向量来表示，用密钥交换技术将维数膨胀的密文约简回原来的维数，用模交换技术控制密文噪声的增长，无需 Bootstrappling 技术，但是密钥交换技术需要在公钥里增加大量用于密钥交换的矩阵，大大增加公钥长度。

第三类构造是 Gentry-Sahai-Waters 方案 [Crypto2013]。基于 LWE (Learning With

Errors）或环 LWE 假设，使用近似特征向量方案，密文的同态计算就是矩阵的加法和乘法。由于使用的是矩阵，所以密文的乘积没有膨胀问题，从而不需要密钥交换技术和模交换技术。

但无论采用上述哪种构造，其运算效率始终无法满足实际应用需求。这也是为什么直到现在都没有完全同态加密技术的成功商业实践。ShaftStop 采用了全新的构造思想，直接对明文数值进行加密，并引入函数密钥和运算字典，大大提高了密文运算的安全性和运算效率，初步具备了商业化应用的可能。

二、ShaftStop 密码体制

2.1. 完全同态加密的定义

传统的加密体制由三元组 $\{E, D, KGen\}$ 构成。这里使用 P 代表明文，C 代表密文，K 代表密钥，E 代表加密操作 $C = E(P, K)$，D 代表解密操作 $P = D(C, K)$，$KGen$ 代表密钥生成操作 $KGen(rand) = K$。

完全同态加密体制由四元组 $\{E, D, KGen, Eval\}$ 构成，与普通的加密体制相比，新增了一项 $Eval$ 操作，即密文运算操作，且满足：

1）针对一组明文 P_i，进行加密操作得到密文组 $C_i = E(P_i, K), i \in [1, n]$；

2）对密文组进行任意函数 F 的密文运算，得到 $C_{res} = Eval(F, \{C_1, C_2, ..., C_n\})$；

3）解密结果密文，得到结果明文 $P_{res} = D(C_{res}, K)$；

4）此结果等于针对明文组 P_i 进行相同的函数运算所得，即 $P_{res} = F(P_1, P_2, ..., P_n)$；

ShaftStop 密码体制在此基础上又引入了一项运算字典生成操作 $Dict$，用以辅助实现密文运算。运算字典类似于文献 [Lai2016] 中提出的运算操作密钥，可以让密文运算操作在受保护的前提下进行。因此，ShaftStop 由五元组 $\{E, D, KGen, Eval, Dict\}$ 构成。

其中，运算字典生成操作 $Dict(K) = G$ 由密钥作为输入，输出为运算字典 G。针对密文的任意运算需要在运算字典 G 的参与下进行，即 $C_{res} = Eval(F, \{C_1, C_2, ..., C_n\}, G)$。

2.2. ShaftStop 密文构造

ShaftStop 体制的密文基于带有未知函数的多元高次多项式来进行构造，在最差情况下基本困难问题为二次多元方程组的求解难题（MQ 问题），此时密文表达式的最高次为 2，表达式为 $P=\sum_{i=1}^{n} a_i \cdot f(x_i) \cdot y_i$。

其中，密文为 $C=\{A,X\}$，$A=\{a_i \mid i \in I\}$，$X=\{x_i \mid i \in I\}$，$I=[1,n]$。

自变量 y_i 保密，作为密钥一部分。解析函数 f 的表达式同样保密，与 y_i 一起构成密钥，即 $K=\{f,Y\}$，$Y=\{y_i \mid i \in I\}$。

使用函数 f 的表达式作为密钥的一部分，是 ShaftStop 最大的创新点之一。在具体实现时，可以列举若干种函数的模式，并将模式代码和函数解析表达式的参数作为密钥。

由于 ShaftStop 体制的密文表达式是一种多项式，而密文为两个向量的组合，因此在两个密文相乘之后，结果密文的最高次会上升，所以需要对密文的表达式进行扩展。

最高次为 3 的情况下，密文表达式为：

$$P=\sum_{i=1}^{n}\sum_{j=i+1}^{n}\sum_{k=0}^{2} a_{ij} \cdot f(x_{ij}) \cdot y_i^{2-k} \cdot y_j^{k}$$

最高次为 4 的情况下，密文表达式为：

$$P=\sum_{i_1=1}^{n}\sum_{i_2=i_1+1}^{n}\sum_{i_3=i_2+1}^{n}\sum_{k_1=0}^{3}\sum_{k_2=0}^{3-k_1} a_{i_1 i_2 i_3} \cdot f(x_{i_1 i_2 i_3}) \cdot y_{i_1}^{3-k_1-k_2} \cdot y_{i_2}^{k_1} \cdot y_{i_3}^{k_2}$$

综上，最高次为 m 时的密文表达式为：

$$P=\sum_{i_1}^{n}\sum_{i_2}^{n}\cdots\sum_{i_j}^{n}\left(\sum_{k_1=0}^{m-1}\sum_{k_2=0}^{m-1-k_1}\cdots\sum_{k_{j-1}=0}^{m-1-\sum_{p=1}^{j-2}k_p} a_{i_1 i_2 \ldots i_j} \cdot f(x_{i_1 i_2 \ldots i_j}) \cdot y_{i_1}^{m-1-\sum_{p=1}^{j-1}k_p} \cdot \prod_{q=2}^{n} y_{i_q}^{k_{q-1}}\right), j \in [1,n]$$

ShaftStop 体制通过引入未知表达式的函数密钥 f，实现了较强安全性，因此为了尽量提高密文运算性能，在实际使用时可以采用较低的参数配置，即最高次 m=2，多项式自变量 y 的数量 n=2，此时整体难度相当于拥有较大规模自变量数的 MQ 问题。

2.3. 运算字典

ShaftStop 的运算字典定义如下：

$$G=\{g_1,g_2,h_1,h_2\}$$

$$\begin{cases} g_1(x_1,x_2,c)=\dfrac{f(x_1)+c\cdot f(x_2)}{f[h_1(x_1,x_2)]} \\ g_2(x_1,x_2)=\dfrac{f(x_1)\cdot f(x_2)}{f[h_2(x_1,x_2)]} \\ h_1(x_1,x_2),h_2(x_1,x_2):R^2\to R \\ h_1(x_1,x_2)\neq h_2(x_1,x_2)\neq x_1\neq x_2 \end{cases}$$

其中，g_1,g_2 是两个多维键值映射，其解析表达式未知，采用离散化的方法使用取值点集合代替解析式，将自变量作为 key，函数运算结果作为 value，构成多维键值映射结构。这种数据结构在一些高级语言中被称为字典，运算字典因此得名。为了使离散化策略可以有效地覆盖整个自变量定义域，可以在构造 $f(x)$ 时控制其定义域的大小，或者让 $f(x)$ 具有周期性。

$h_1(x_1,x_2),h_2(x_1,x_2)$ 为自变量映射函数，其解析表达式已知，且值域覆盖了 $f(x)$ 的整个定义域，因此该映射具有定义域遍历性，或者说 $h_1(x_1,x_2),h_2(x_1,x_2)$ 是将二维实数空间的点映射到一维实数空间上特定区间的满射。

运算字典的作用十分类似自然对数表，通过预先计算指定连续函数的离散取值，在具体使用时，配合插值算法可以简单快速的得到任意自变量的对应函数值。对于无法提供解析表达式的函数，采用查表法配合插值公式是十分有效的方法。

ShaftStop 引入运算字典，解决函数密钥 f 的运算问题，无需掌握 f 的解析式，就可以对函数 f 进行运算，即计算：

$$\begin{cases} f(x)+f(y)=b_1\cdot f(z_1) \\ f(x)\cdot f(y)=b_2\cdot f(z_2) \end{cases}$$

通过运算字典 G 可以从 x 和 y 直接得到：

$$\begin{cases} b_1 = g_1(x, y, 1) \\ z_1 = h_1(x, y) \\ b_2 = g_2(x, y) \\ z_2 = h_2(x, y) \end{cases}$$

这里要求 g_1, g_2 连续且在定义域内处处可微，其定义域内任意位置的偏导数 $\left.\frac{\partial g_1}{\partial x_i}\right|_{x_1, x_2} < d_1, i \in \{1,2\}$，$\max\left(\left.\frac{\partial g_2}{\partial x_i}\right|_{x_1, x_2, c}, \left.\frac{\partial g_2}{\partial c}\right|_{x_1, x_2, c}\right) < d_2, i \in \{1,2\}$，其中 d_1, d_2 为取值较小的实数。这时 g_1, g_2 具有很好的拟合特性，使用插值算法可以很高精度的获得任意自变量取值下的 g_1, g_2 的函数值。

同时，运算字典 G 也可以看作是运算操作密钥，只有掌握 G 才可以进行密文运算。通过引入运算操作密钥，让密文运算必需在取得授权的前提下才能进行，提升了系统的安全性。

2.4. 密钥生成

ShaftStop 的密钥由两部分组成 $K = \{f, Y\}$，其中 f 称为函数密钥，由解析函数的模式代码和解析式参数构成；Y 称为多项式密钥，由一个实数向量构成。函数密钥可以与多项式密钥分开生成。

函数密钥部分是 ShaftStop 体制安全性的关键，为了增加攻击者通过数值方式拟合函数密钥的难度，这里要求函数密钥 f 具有混沌特性。根据混沌系统的初始条件敏感性，任意初始条件的微小差异，经过混沌系统的不断放大，都可以对未来的状态造成巨大的影响。这就是混沌系统的“蝴蝶效应”。

如果要尝试拟合一个混沌函数，任何当前函数取值的微小误差，都会对函数未来取值的推导产生严重的偏差，这就使得针对同一个混沌函数的不同次拟合尝试会得到完全迥异的结果。除非可以精确地知道混沌函数的解析式，否则很难通过拟合方法预测其行为。混沌函数的这种特性十分适合被用来构造加密体制，这也正是混沌密码的设计思路。

根据李雅普诺夫（Lyapunov）指数判据，若指定函数 f 的 Lyapunov 指数 $\lambda > 0$ 大于

零，则 f 具有混沌特性，其中 Lyapunov 指数定义如下：

$$\lambda = \lim_{n\to\infty} \frac{1}{n} \sum_{i=0}^{n-1} \ln \left| \frac{df(x_n)}{dx} \right|$$

通过混沌函数生成工具可以随机获得符合要求的混沌函数用来作为函数密钥 f，然后根据 f 可以生成运算字典 G，再随机生成作为多项式密钥的实数向量 Y，即完成了密钥的生成操作。

2.5. 加密解密

由于引入具有混沌特性的函数密钥 f，整个体系实现了较强的安全性，在 $m=2, n=2$ 时，即可提供足够的安全保障。因此默认情况下，ShaftStop 采用此参数配置来进行加密，密钥为 $K=\{f, y_1, y_2\}$，密文表达式为 $P = a_1 \cdot f(x_1) \cdot y_1 + a_2 \cdot f(x_2) \cdot y_2$。

加密操作步骤如下：

1）在 f 的定义域 $Def \subset R$ 内随机选择两个实数 $x_1, x_2 \in Def$，计算 $f(x_1)$ 和 $f(x_2)$；

2）随机生成实数 $\alpha \in (0,1)$，计算 $P_1 = P \cdot \alpha$；

3）计算 $a_1 = \frac{P_1}{f(x_1) \cdot y_1}$，$a_2 = \frac{P - P_1}{f(x_2) \cdot y_2}$。

经过上述三步操作即可得到密文 $C = \{a_1, x_1, a_2, x_2\}$。

这里的加密操作具有三个随机变量作为输入，使得密文向量的四个元素中拥有三个独立的随机因子，密文的整体随机特征非常显著。针对相同明文，每次加密操作的结果差别巨大，可有效对抗统计分析型攻击。

解密操作则简单直接，只需将密文 $C = \{a_1, x_1, a_2, x_2\}$ 和密钥 $K = \{f, y_1, y_2\}$ 代入密文表达式 $P = a_1 \cdot f(x_1) \cdot y_1 + a_2 \cdot f(x_2) \cdot y_2$，即可计算出明文 P。

综上所述，ShaftStop 的加密解密操作相对简单高效，在掌握密钥的前提下，可以非常快速的进行，同时也容易通过硬件来实现。

2.6. 全同态性推导

在 m=2，n=2 的情况下，可以推导出加法和乘法的同态性。

1）加法同态

$$
\begin{aligned}
&C_1 + C_2 \\
&= [a_{11} \cdot f(x_{11}) \cdot y_1 + a_{12} \cdot f(x_{12}) \cdot y_2] + [a_{21} \cdot f(x_{21}) \cdot y_1 + a_{22} \cdot f(x_{22}) \cdot y_2] \\
&= a_{11} \cdot [f(x_{11}) + \frac{a_{21}}{a_{11}} \cdot f(x_{21})] \cdot y_1 + a_{12} \cdot [f(x_{12}) + \frac{a_{22}}{a_{12}} \cdot f(x_{22})] \cdot y_2 \\
&= a_{11} \cdot g_1(x_{11}, x_{21}, \frac{a_{21}}{a_{11}}) \cdot f[h_1(x_{11}, x_{21})] \cdot y_1 + a_{12} \cdot g_1(x_{12}, x_{22}, \frac{a_{22}}{a_{12}}) \cdot f[h_1(x_{12}, x_{22})] \cdot y_2 \\
&= a_{31} \cdot f(x_{31}) \cdot y_1 + a_{32} \cdot f(x_{32}) \cdot y_2 \\
&= C_3
\end{aligned}
$$

其中：

$$
\begin{cases}
a_{31} = a_{11} \cdot g_1(x_{11}, x_{21}, \frac{a_{21}}{a_{11}}) \\
a_{32} = a_{12} \cdot g_1(x_{12}, x_{22}, \frac{a_{22}}{a_{12}}) \\
x_{31} = h_1(x_{11}, x_{21}) \\
x_{32} = h_1(x_{12}, x_{22})
\end{cases}
$$

通过上述推导可以清楚的看到，针对两个二次密文的加法运算，其结果也是一个二次密文。依托运算字典可以非常快速的直接得到加法运算的结果。

2）乘法同态

$$
\begin{aligned}
&C_1 \cdot C_2 \\
&= [a_{11} \cdot f(x_{11}) \cdot y_1 + a_{12} \cdot f(x_{12}) \cdot y_2] \cdot [a_{21} \cdot f(x_{21}) \cdot y_1 + a_{22} \cdot f(x_{22}) \cdot y_2] \\
&= a_{11} \cdot a_{21} \cdot f(x_{11}) \cdot f(x_{21}) \cdot y_1^2 + [a_{11} \cdot a_{22} \cdot f(x_{11}) \cdot f(x_{22}) + a_{12} \cdot a_{21} \cdot f(x_{12}) \cdot f(x_{21})] \cdot y_1 \cdot y_2 \\
&\quad + a_{12} \cdot a_{22} \cdot f(x_{12}) \cdot f(x_{22}) \cdot y_2^2 \\
&= a_{11} \cdot a_{21} \cdot g_2(x_{11}, x_{21}) \cdot f[h_2(x_{11}, x_{21})] \cdot y_1^2 + [a_{11} \cdot a_{22} \cdot g_2(x_{11}, x_{22}) \cdot f[h_2(x_{11}, x_{22})] \\
&\quad + a_{12} \cdot a_{21} \cdot g_2(x_{12}, x_{21}) \cdot f[h_2(x_{12}, x_{21})]] \cdot y_1 \cdot y_2 + a_{12} \cdot a_{22} \cdot g_2(x_{12}, x_{22}) \cdot f[h_2(x_{12}, x_{22})] \cdot y_2^2 \\
&= a_{31} \cdot f(x_{31}) \cdot y_1^2 + [a'_{32} \cdot f(x'_{32}) + a''_{32} \cdot f(x''_{32})] \cdot y_1 \cdot y_2 + a_{33} \cdot f(x_{33}) \cdot y_2^2 \\
&= a_{31} \cdot f(x_{31}) \cdot y_1^2 + a'_{32} \cdot g_1(x'_{32}, x''_{32}, \frac{a'_{32}}{a''_{32}}) \cdot f[h_1(x'_{32}, x''_{32})] \cdot y_1 \cdot y_2 + a_{33} \cdot f(x_{33}) \cdot y_2^2 \\
&= a_{31} \cdot f(x_{31}) \cdot y_1^2 + a_{32} \cdot f(x_{32}) \cdot y_1 \cdot y_2 + a_{33} \cdot f(x_{33}) \cdot y_2^2 \\
&= C_4
\end{aligned}
$$

其中：

$$\begin{cases} a_{31} = a_{11} \cdot a_{21} \cdot g_2(x_{11}, x_{21}) \\ a'_{32} = a_{11} \cdot a_{22} \cdot g_2(x_{11}, x_{22}) \\ a''_{32} = a_{12} \cdot a_{21} \cdot g_2(x_{12}, x_{21}) \\ a_{33} = a_{12} \cdot a_{22} \cdot g_2(x_{12}, x_{22}) \\ a_{32} = a'_{32} \cdot g_1(x'_{32}, x''_{32}, \frac{a'_{32}}{a''_{32}}) \\ x_{31} = h_2(x_{11}, x_{21}) \\ x'_{32} = h_2(x_{11}, x_{22}) \\ x''_{32} = h_2(x_{12}, x_{21}) \\ x_{33} = h_2(x_{12}, x_{22}) \\ x_{32} = h_1(x'_{32}, x''_{32}) \end{cases}$$

乘法同态的运算过程相对复杂，两个二次密文的乘法运算，其结果是一个三次密文。同时，乘法运算中包含加法运算，其运算字典查询次数多于加法运算。

2.7. 问题与改进

ShaftStop 密码体制通过引入函数密钥和运算字典，实现了较强的安全性和较高的密文运算效率，但是同时也带来了新的问题：

1）函数密钥 f 的定义决定了 ShaftStop 密码体制的安全性和整体效率。如何生成简单有效，同时符合混沌判据，又容易运算的函数密钥，是 ShaftStop 密码体制的基本问题。

2) 这里定义的运算字典 $g_1(x_1, x_2, c)$ 是一个三元函数，需要有效遍历三个自变量的定义域方可实现离散化，这会导致庞大的数据量，对实际使用时的存储和传输都会带来很大的麻烦。如何既能保证安全性和运算精度要求，同时又能减少运算字典的数据量是 ShaftStop 密码体制走向实用的关键问题。

3）由于运算字典的设计思路是首先离散化函数运算的中间结果，再结合插值算法获得最终结果，这种数值处理方式带来了精度问题。要提高精度，一方面可以通过增加运算字典的采样密度来实现，但是这样会显著增加运算字典的数据量；另一方面，也可通过采用更加平缓的函数密钥 f 来实现，但是这样又会降低整体安全性。如何有效控制密文运算的精度，

同时保证安全性和合理的运算字典数据量是 ShaftStop 密码体制重点需要解决的问题。

4）完全同态加密在实际使用时，不是光解决加法和乘法运算就能满足所有应用场景的处理需求，还需要一系列的配套技术，诸如：同态比较运算，密钥交换操作，密文交割操作，同态除法运算，同态取整运算，同态三角函数运算等等。这些都是 ShaftStop 加密算法要商业化使用所必需克服的问题。

针对上述问题，我们基本都已经有了很好的解决方案。这里暂时不做展开，之后我们会陆续通过论文或其他形式给出进一步的说明。

三、安全性分析

3.1. 基本困难问题

这里选择默认安全参数设置（m=2，n=2）下的密文，作为安全性分析的主要研究对象，其密文表达式为： $P = a_1 \cdot f(x_1) \cdot y_1 + a_2 \cdot f(x_2) \cdot y_2$。

由于函数密钥 f 的函数解析式模式和参数均未知，直接通过大量明文和密文的对应关系尝试求解密钥 $K = \{f, y_1, y_2\}$，相当于求解一般类型的函数方程组。而一般类型的函数方程组在数学上至今尚无有效的求解方法。因此无法直接反推函数密钥的解析式。

试图通过数值方法拟合 f 时，一方面由于多项式密钥 Y 的存在，很难准确分离出 f 的取值，同时会因为 f 具有混沌特性，让逐步逼近的数值分析策略遇到严重干扰。所以很难有效的利用数值方法反推出密钥。

如果放宽限制，假设 f 的解析式模式已知，仅参数未知。可以通过大量已知的明文和密文对应关系列出一系列方程组进行求解，则理论上存在破解可能。但是实际上，由于密文生成机制的随机性，加上密文运算字典的自变量映射函数 h 具有定义域遍历性，只要 f 的定义域或者周期足够大，能够反推求解出密钥的方程组数量是巨大的。多项式时间的攻击者无法在可接受的运算时间内完成破解工作。

为了简化论述，假设 f 的定义域可以离散化为 N 个取值点，采用换元法引入 N 个变量替换 $f(x)$，则密文表达式退化为 $P = a_1 \cdot z_{i_1} \cdot y_1 + a_2 \cdot z_{i_2} \cdot y_2 \mid i_1, i_2 \in [1, N]$，这时的密钥为 $K = \{Z, Y\}, Z = \{z_{x_i} \mid x_i \in C\}$。

这是典型的二元二次方程组（MQ）求解问题，被证明为 NP-Complete 问题。通过控制函数密钥定义域的大小可以保证 MQ 问题具有足够安全的规模。降低函数密钥的复杂度，可以在维持运算字典规模基本不变的前提下，有效增加定义域的大小。在函数密钥定义域足够大时，由于新引入的未知变量 Z 针对每一个不同的密文 C 均不同，相当于实现了“一次一密”的加密效果，理论上很难破解。

综上所述，ShaftStop 密码体制的基本困难问题在最差情况下可以归结为二元二次方程组的求解问题，其不但是多项式时间安全的，同时也是公认的能够抵御量子计算的困难问题。

3.2. 语义安全性分析

针对 ShaftStop 的语义安全性分析可以在最差情况下进行，这里讨论最常见的两种攻击模式下的安全性。

1）选择明文攻击

在选择明文攻击情况下，假设敌手可以掌握加密操作，从而可以加密任意指定的明文，获得其对应的密文。

此时，敌手输出两个明文 $\{p_0, p_1\}$，挑战者随机选择其中任意一个进行加密，并将目标密文 C_1 给敌手。如果敌手可以正确指出挑战者输出的密文是对应哪一个明文，则敌手获胜。由于 ShaftStop 的加密操作具有高度随机特性，因此敌手很难通过对比密文的方式来识别挑战者选择的是哪一个明文。同时，依靠列举大量明文-密文对来破解密钥也不可行。因此，敌手除了猜测，没有多项式时间算法可以有效提升给出答案的正确率。

反复进行上述实验，如果经过足够多次后，敌手获胜的概率不能显著地大于 50%，则

认为加密算法在选择明文攻击下具有不可区分性，或者称为 IND-CPA 安全。由于针对 ShaftStop 的密文区分只有随机猜测的方法，因此 ShaftStop 加密体制是 IND-CPA 安全的。

2）选择密文攻击

在选择密文攻击情况下，假设敌手可以在挑战开始前访问解密操作，从而对任意构造的密文进行解密，获得其对应的明文。

如果不进行密文有效性检验，则敌手可以构造一些特殊的密文，从而快速破解密钥。例如，敌手可以构造密文 $C_1 = \{a_{11}, x_{11}, a_{21}, x_{21}\}$，其中 $a_{21} = 0$，则密文表达式的第二项就失效了，简化为 $P_1 = a_{11} \cdot f(x_{11}) \cdot y_1$。由于可以调用解密操作得到 P_1 的值，因此敌手可以构造函数 $\varphi_1(x_{11}) = f(x_{11}) \cdot y_1 = \dfrac{P_1}{a_{11}}$。虽然 f 和 y 尚未被破解，但是敌手可以遍历定义域获得 $\varphi_1(x)$ 的所有取值。使用同样的方法，敌手可以构造 $a_{11} = 0$ 的密文，并遍历 x_2 获得 $\varphi_2(x)$ 的所有取值。这时，敌手可以使用新的密文表达式 $P = a_1 \cdot \varphi_1(x_1) + a_2 \cdot \varphi_2(x_2)$ 来破解密文。

挑战开始后，敌手输出两个明文 $\{p_0, p_1\}$，挑战者随机选择其中任意一个进行加密，并将目标密文 $C_1 = \{a_{11}, x_{11}, a_{21}, x_{21}\}$ 给敌手。敌手将密文直接代入新的密文表达式，即可得到明文 P_1'。即使存在误差，敌手仍可选择两个明文 $\{p_0, p_1\}$ 中更加接近 P_1'的一个作为结果返回给挑战者。显然，敌手获胜的概率是非常高的。这种情况下，ShaftStop 不是 IND-CCA1 安全的。

解决的办法是解密操作执行前必需对密文进行有效性校验，要求密文的每个组分均不能为 0。同时可以借鉴其他加密体制的做法，在密文最后附加校验码，来阻止敌手随意构造特殊密文。经过这些强化后，由于无法任意构造密文，敌手掌握解密操作的优势就被削弱了。这时 ShaftStop 体制可以做到非适应性选择密文攻击下的不可区分性，即 IND-CCA1 安全。

更进一步，如果敌手在挑战者给出目标密文后，还有机会再次访问解密操作，并且允许提交除目标密文以外的其他密文进行解密，然后再给出最终结果，这就是适应性选择密文攻

击。满足适应性选择密文攻击下的不可区分性条件的加密体制称为 IND-CCA2 安全。

对于任何允许公开操作密文的完全同态加密体制，敌手都可以在接收到挑战者给出的目标密文后，进行一次密文运算，例如将目标密文加上 0，获得新的密文，再调用解密操作，即可准确指出目标密文所对应的明文，赢得挑战。所以，允许公开操作密文的完全同态加密体制一定不是 IND-CCA2 安全的。

但是，假设敌手无法掌握运算操作密钥，在 ShaftStop 加密体制中就是运算字典，则敌手就无法对密文进行运算。加上敌手无法任意构造合法的密文，所以很难找到有效的方法来识别目标密文。因此，在敌手无法掌握运算字典的前提下，ShaftStop 体制是 IND-CCA2 安全的。

四、典型应用场景

4.1. 区块链数据保护

区块链技术是一种分布式账本技术，是由密码学数字货币技术发展而来的一项通用技术。由于其具有去中心化、高可用、强共识和点对点可信交易等特点，区块链技术可以使用在数字经济，数字金融，数字供应链等领域的各类场景中，应用前景广泛。但是区块链技术本身不具备数据保护能力，全网同步共识和透明可追溯的特点对运行于其上的各种应用带来了数据隐私的担忧。

完全同态加密技术可以很好地与区块链技术相结合，保护应用数据。在数字货币交易数据保护方面，应用完全同态加密技术可以做到保留交易账户信息可追溯的同时实现交易金额的加密，这种保护模式更加接近真实金融系统的隐私模型。在智能合约及链上应用层面，完全同态加密为 DApp 提供了隐私数据处理能力，可以实现更加强大灵活的业务逻辑。

4.2. 加密数据集上的 AI

人工智能的蓬勃发展带来了巨大的数据需求，更多维度高质量的训练数据可以得到更加

有效的 AI 模型。如何在保护数据隐私的前提下应用 AI 技术，已经是业界普遍关注的问题。差分隐私技术可以部分解决这个问题，完全同态加密则提供了更加完整的解决方案。不但在模型训练时可以在密文数据集上进行，在模型应用时也可以基于密文数据得到运行结果，从而保护数据隐私。例如，在生物识别领域，如果在线模型可以处理使用完全同态加密技术保护的图像，进而完成识别工作，就可以最大程度的保护被识别人的隐私，同时避免被黑客截获图像信息用于伪造身份。

4.3. 保密数据银行

保密数据银行（或称隐私数据银行 Private Data Bank）的概念在同态加密被提出时就由 Rivest 等人做了描述。个人或企业通过同态加密算法将其数据资产加密成密文后，可以托管在保密数据银行的云端安全存储空间中。保密数据银行负责存储及保管数据资产，并且基于完全同态加密的密文操作特点，为客户提供数据资产的运算服务，交易服务，以及租赁服务。数据资产所有方自己保管密钥，保密数据银行也可以提供额外的密钥保护方案，及密钥恢复能力。结合区块链技术和加密数据集上的人工智能应用，相信保密数据银行会是一项非常有趣的全新服务。

五、总结展望

完全同态加密是数据隐私保护方面的强大工具，应用前景十分广阔。ShaftStop 提出了一种全新的构造完全同态加密体制的思路和方法。通过初步分析和原型实验，其安全性和运算性能均能满足商业化的基本需求，潜力巨大。作为一项以商用为目标的新技术，ShaftStop 已经发展出十余项配套技术，构成了一整套技术体系，其中一些技术已经实现了原型，并且通过实验进行了验证，大部分技术申请了专利保护。但是，整个体系尚有很多问题需要解决和改进，诸如二进制对象的加密和处理，密文合法性校验，同态 Hash 技术，同态签名技术等等。这些方向都是未来工作的重要内容。

参考文献

[Rivest78] Rivest, R., Adleman, L., & Dertouzos, M. (1978a). On Data Banks and Privacy Homomorphisms. Foundations of Secure Communication, pp. 169-177, Academic Press.
[STOC2009] Craig Gentry. Fully homomorphic encryption using ideal lattices. STOC 2009.
[Gentry2009] Craig Gentry. A Fully Homomorphic Encryption Scheme, Ph.D. thesis. 2009.
[BV2011] Brakerski and V. Vaikuntanathan. Efficient Fully Homomorphic Encryption from (Standard) LWE. In: FOCS, pp.97-106 2011.
[Crypto2013] Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In Advances in Cryptology–CRYPTO，2013, pp. 75–92. Springer, 2013.
[Lai2016] Lai J., Deng R.H., Ma C., Sakurai K., Weng J. (2016) CCA-Secure Keyed-Fully Homomorphic Encryption. In: Cheng CM., Chung KM., Persiano G., Yang BY. (eds) Public-Key Cryptography – PKC 2016.