# 零知识证明中的同态加密

在很多密码学问题中，我们有时候不能暴露一些值，所以，有时候需要对原始加密以后让验证者去验证，这样秘密就不会暴露。 当然，这就需要加密后的值 和 原始数据之间有一种映射关系。

## 定义

假设 a b 是原始值， A B 是加密后的值，也就是说

```
A = E(a), B = E(b), C = E(c)
a * b = c  <=>  A ∘ B = C
```

也就是要验证 a * b = c 只要验证 A ∘ B = C， 注意 * ∘ 都只是一个二元运算，而且这两个运算符号不一定相同。

## 例子1：RSA 同态乘法

我们有两个原始的数据，M1， M2

```
M1 * M2 = M
```

现在我们要证明 RSA 加密后也有

```
R(M1) * R(M2) = R(M1 * M2) = R(M)
```

我们看看加密:

```
M1 ^ e = R(M1)
M2 ^ e = R(M2)
R(M1) * R(M2) = M1 ^ e * M2 ^e =  (M1 * M2) ^ e = M ^ e = R(M)
```

有了这个同态，我们看看有什么应用。

## 例子2: ECC 同态加法

在ECC中，加密的方法不再是用指数，而是乘法。比如我们有两个消息m1 和 m2，他们都是整数。

```
E(m1) = m1 * G

E(m2) = m2 * G
```

```
E(m1) + E(m2) = m1 * G + m2 * G = (m1 + m2) * G = E(m1 + m2)
```

## 零知识证明中如何加密多项式

零知识证明中，我们需要对多项式进行加密，不能让人看到多项式以及多项式的系数。

比如这样一个多项式

```
f(x) = 2x^3 –3x^2 + x + 1
```

因为ECC不支持同态乘法，所以只支持两种运算：

```
常数 * 加密值
加密值 + 加密值
```

因为不支持两个加密值的乘法，在做同态加密的时候需要提供 E(x^3), E(x^2), E(x) 的值。 多项式可以看作是 [E(x^3), E(x^2), E(x), 1] 和 向量 [2, -3, 1, 1] 的内积。 当然，细节要比这个复杂.