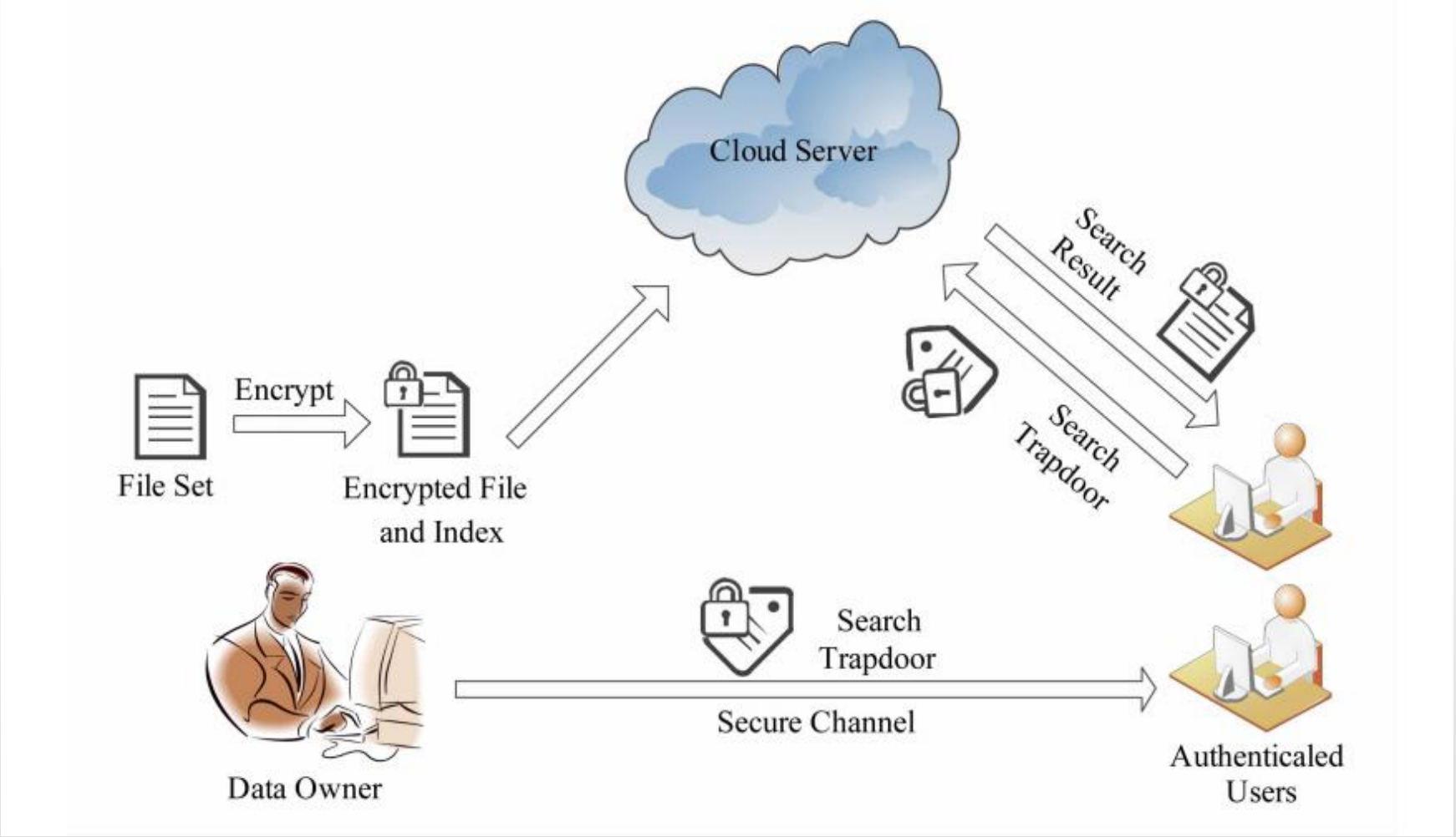
整体方案

- 初始化阶段，数据拥有者首先选定待存储的文件集，然后运行密钥生成算法为系统生成密钥
- 对于已选定的文件集，数据拥有者将采用传统的对称加密算法对其进行加密
- 数据拥有者为待存储的文件设置索引，即运行索引生成算法得到文件安全索引，然后将加密后的文件集与安全索引一同存储到服务器中
- 数据拥有者根据判断选择自己可信赖用户
- 当可信赖用户想要在服务器中搜索待定文件时，将查询关键词发送给数据拥有者
- 数据拥有者在接收到用户发送的关键词集，运行陷门算法为关键词生成搜索陷门
- 可信赖用户将搜索陷门发送到服务器进行搜索请求
- 服务器在接收到搜索陷门后将搜索陷门和加密的安全索引运行搜索算法，得到加密的目标文件集。然后将目标文件集返回给可信赖用户

整体方案


Cloud Server
Search Result
Search Trapdoor
Encrypt
File Set
Encrypted File and Index
Search Trapdoor
Secure Channel
Data Owner
Authenticaled Users

服务器中存储数据结构

- 表1 安全索引和对应文件ID集
  - 在原有倒排索引基础上，将索引进行安全加密，并存储
- 表2 文件ID和对应的加密数据

```
[[(array([2.06131966e+11, 2.01480555e+11, 2.04379199e+11, ...,
       2.13377536e+11, 2.07586748e+11, 2.16340343e+11]), array([-2.04598988e+11, -2.07439674e+11, -2.11552977e+11, ...,
       -1.97747858e+11, -2.14265117e+11, -2.09180329e+11])), {'./doc./1.txt', './doc./3.txt'}], [(array([2.06275671e+11, 2.01336043e+11,
       2.13405119e+11, 2.07630112e+11, 2.16253652e+11]), array([-2.04554213e+11, -2.07353034e+11, -2.11655296e+11, ...,
       -1.97695679e+11, -2.14273896e+11, -2.09247891e+11])), {'./doc./4.txt', './doc./1.txt', './doc./3.txt'}], [(array([2.06233477e+11,
       2.13552776e+11, 2.07601348e+11, 2.16308439e+11]), array([-2.04563463e+11, -2.07492288e+11, -2.11631693e+11, ...,
       -1.97671315e+11, -2.14407660e+11, -2.09294002e+11])), {'./doc./1.txt', './doc./2.txt'}], [(array([2.06225736e+11, 2.01271741e+11,
       2.13515518e+11, 2.07788741e+11, 2.16535292e+11]), array([-2.04373000e+11, -2.07370597e+11, -2.11395072e+11, ...,
       -1.97897825e+11, -2.14268725e+11, -2.09101281e+11])), {'./doc./1.txt', './doc./2.txt'}], [(array([2.06551025e+11, 2.01700410e+11,
       2.13800650e+11, 2.07767673e+11, 2.16712570e+11]), array([-2.04354813e+11, -2.07128271e+11, -2.11095928e+11, ...,
       -1.97513073e+11, -2.13950631e+11, -2.08844967e+11])), {'./doc./1.txt'}], [(array([2.06328740e+11, 2.01331997e+11, 2.04391933e+11,
       2.13449108e+11, 2.07625844e+11, 2.16416901e+11]), array([-2.04467141e+11, -2.07523728e+11, -2.11417983e+11, ...,
       -1.97805648e+11, -2.14384914e+11, -2.09173283e+11])), {'./doc./4.txt', './doc./1.txt', './doc./2.txt'}], [(array([2.06329637e+11,
       2.13593579e+11, 2.07657609e+11, 2.16469333e+11]), array([-2.04473366e+11, -2.07407031e+11, -2.11382390e+11, ...,
       -1.97618718e+11, -2.14262268e+11, -2.09156116e+11])), {'./doc./4.txt', './doc./1.txt', './doc./2.txt', './doc./3.txt'}], [(array([
       2.13566613e+11, 2.07755661e+11, 2.16386507e+11]), array([-2.04630990e+11, -2.07215651e+11, -2.11464741e+11, ...,
       -1.97753620e+11, -2.14455314e+11, -2.09374615e+11])), {'./doc./1.txt', './doc./2.txt'}], [(array([2.06169121e+11, 2.01251030e+11,
```

服务器中存储数据结构

- 表1 安全索引和对应文件ID集
  - 在原有倒排索引基础上，将索引进行安全加密，并存储
- 表2 文件ID和对应的加密数据


```
{'./doc./1.txt':
 b'gAAAAABjoZ5PwylcXZ73NfVp2ziknMjdAA_fNu1JhI5DvklMIRK7YdPCiU7MCNjqUmwGLzZZsIdaSf_qC7G8taHLnxnmjd7HVEGj63Iu1RSOZJrbpu9eF_Trp5F1UoZSHZGcu
 L9pA9cDARvcJBQ59dniYmm9XKkumKpEoo1rrbIAm24x_6LbicbNQXr61DmhZ1vzy-PLLevvFLDR7IsD8N_GlA-WGuNevop5JIzM52nnPD
 -5gq7L3No30dHSlhDPNOEZi3EXCAwYTVC1jyxEGPXhIVsMkY
 -aikc4StLGSB1VpMQjzfWPXD0OawrzJCQBy8rYp_Spyu5VEutPiArGmN0SeVdlXYZFzYqtdwENrA4f6siGNnMIdQR4QuZTaE6862Ou9trAE1aKYlfpjZxM-GJSIsm2
 -BLDxw5Dt55-cBEg3WVRh7Ki3hqz7dnQ4TVRN9h_NGEfkZSXH1lIQ9KbFvBa5DMu8PaoNHIvca07_rQjoX4wHWw93WVlxaqR_wqNa34p39lYkxyOtQe7l
 -l8g41qNAG54Fi40Q2xMFAXEj8UVqvh1mPKwJFDt6VP9wzb0cN4wWuRemdIpGCCDvo3R2CcfhrUVKrXK_w0FRVsQ3WbZVyHvFoj85OlOZBz1o8IhE6TrZHV6eJHV1j_XLrBQHOd
 Nl8FEaRl6Vcimf46OALu-1yBe-vhFWtsDbi5nHKatpLoVipy8bko0O4dLe80jXAoFQwaKpV2ny6zOpB7VwVZsq33YGH5t_h3pHCHkLhtd4iQQJnboO6cXOtW2Mcw8AmSNjUlzx
 -sYwK1Gg0NZApDYWXb1Q
 -C6Wh5yicMGPRR67WGnS_9J5QGFzbPewfDIM4fczlULJoY4f9ArlHaOCAdYWkqVmoIzxqWqecTyJtdv2UzNGY0XX_cixbGxH5NJyeRW_1DYtH850KMSvnQXr7JS_ZNQ2byWMyoW
 7FDL-5hHdGt0-b7V7HAJjMFhQ4sZedfGWDZzLEwHOhAmfoLqf-p7qIVJvcBf7Q-9B4_FPun807L0NBXhOH2NIcr4L8GlOcF
 -rYcxEWp0mQ1JduHWRQ1rzjSh0dtmPJUejL8iqRwhZQiNibpXngyEk_91zbFaho35WYPT2BChhANx12ZHQ05e6O4dOEFdvCtD4jE8m2cIkouYDQf708U_fgmRjiNNw28_rN8ff
 -nOF7a_7v4sa2gsGDwBUidQIgbCB4ORnsxXqtE3Qj
 -Zfty0YcNQrvVNg4FsFatLFBBw8IIO4Xc8L5cX59Lbg76DInwL6kfg6Pa6gsfZyxLKrpovjxbVfHy1blt_71i4ogvT79194y5CcUo
 -uyKyfCFfwhBwax_50KX8B6BM92495SOyqk7NPMpR1Sge7UcxNmRThqlscHTsoR796ncBMuYOpvGE8-ulkdUsOAeNsF4SZP_vkmNvQ0JS3A1a
 -s9impzIlJKhiZ7f4CoqTcQpxr91v3myuYiWVVF8wZZ4mHxQTV8maGdraXUcqfcmgmYWJ8RuEYa6d0BOPqJ8l88j2bU3qZ3s2eCvi2WS93njEeSODc3BBM9GEfZpb_zNT5u1IT6
 QpT8WyDD2NqGCZmmesJJSDJTa5GrWzjF5AkIa5cWh3nZeN0uLvKqI-38UqGwMOPNsvHFoHsMiVaq9IPpZp4hzlLU5
```

密钥生成KeyGen(K)

- 1) 随机构建2个k*K矩阵M1，M2,1个k维向量S

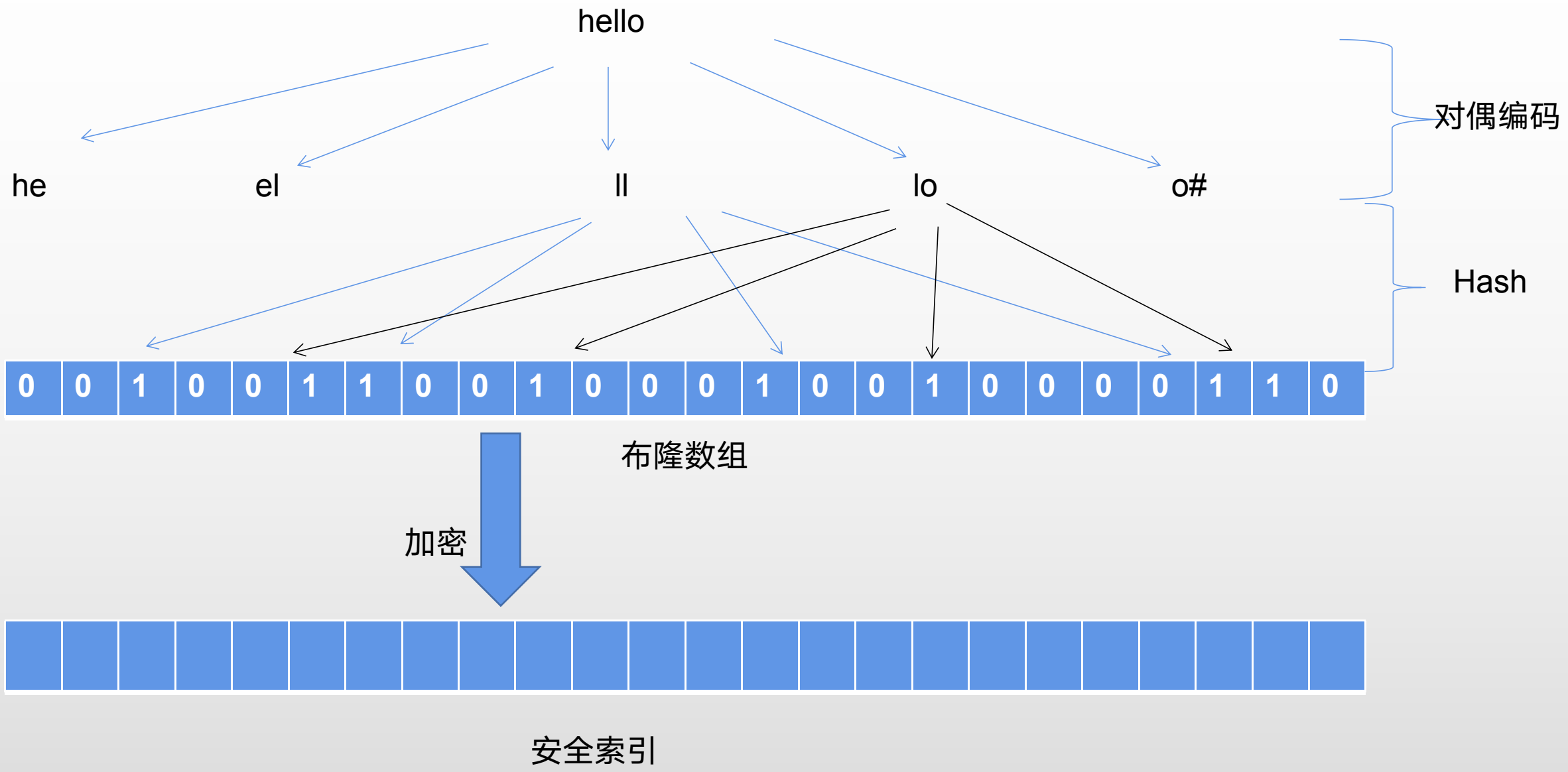
$$M_1, M_2 \in R^{k*k}$$

$$S \in R^k$$

> 其中的K为指定值1470，这是由于后面使用布隆过滤器所调整的值

安全索引

- 将关键词和对应按照倒排索引方式进行存储
- 一个关键词经过三个步骤处理
  - 使用对偶编码，将单词进行分成多个小片段
  - 将这些片段插入到布隆过滤器中
  - 将布隆过滤器值进行安全加密，该加密部分是加密的安全索引、
- 加密方法


hello
he
el
ll
lo
o#
对偶编码
Hash
0 0 1 0 0 1 1 0 0 1 0 0 0 1 0 0 1 0 0 0 0 1 1 0
布隆数组
加密
安全索引

B_i为第i个单词生成的布隆数组，与S具有相同的结构，即S的维数和B_i的维数相同,r为随机值 构建新的索引向量B_i'和B_i''

$$当S[j]=0时B_i'[j]=\frac{1}{2}B_i[j]+r, B_i''[j]=\frac{1}{2}B_i[j]-r$$

$$当S[j]=1时B_i'[j]=B_i''[j]=B_i[j]$$

$$I_i'=M_1^T.B_i', I_i''=M_2^T.B_i''$$

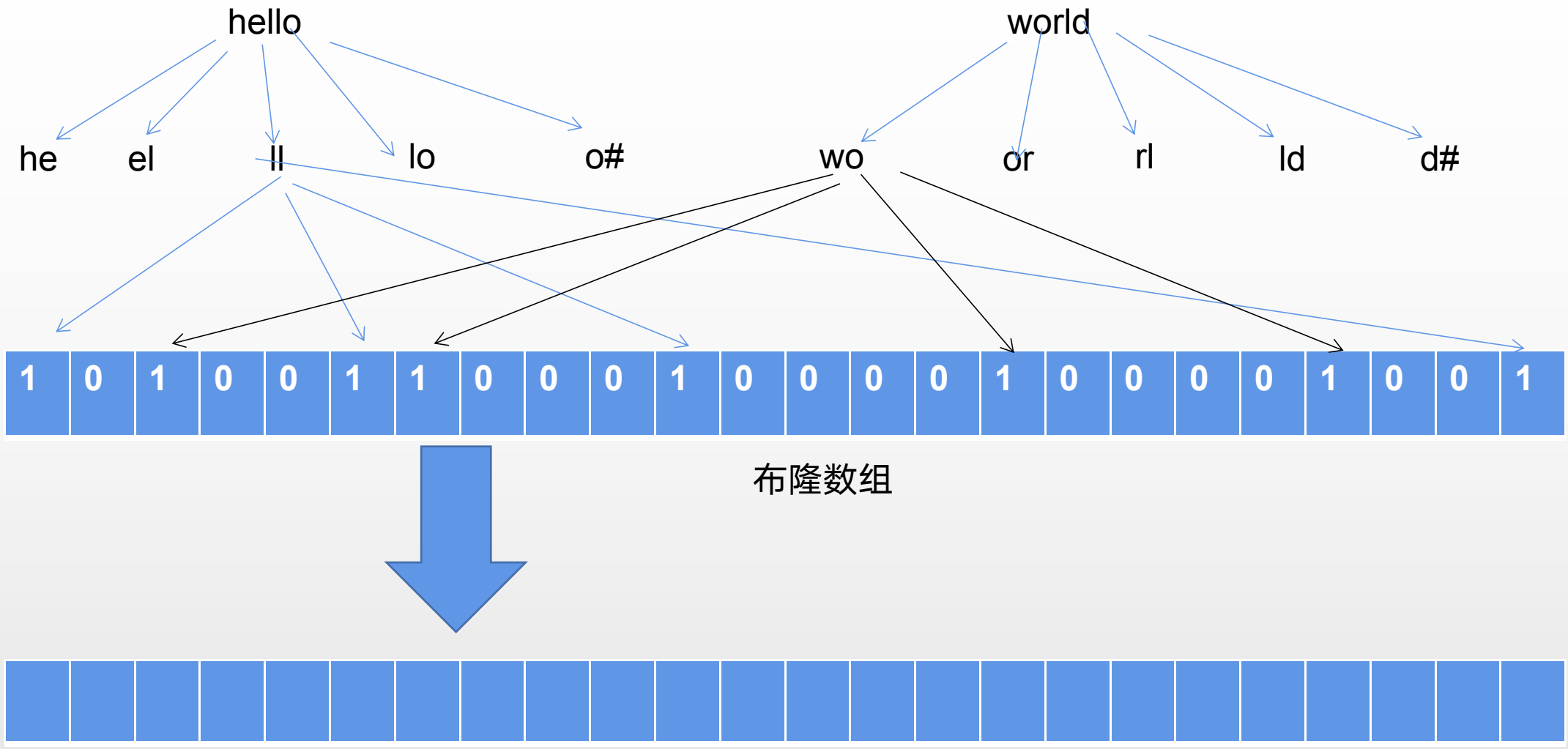
$$I_i=(I_i', I_i'')$$

I_i即为加密的安全索引

```
[[(array([1.19337169e+11, 1.24603969e+11, 1.20129419e+11, ...,
       1.24213184e+11, 1.22377040e+11, 1.17221770e+11]), array([-1.18387197e+11, -1.19839677e+11, -1.21365285e+11, ...,
       -1.23513620e+11, -1.16485207e+11, -1.18741136e+11])), {'./doc./1.txt'}], [(array([1.19284513e+11, 1.24471494e+11, 1.19854456e+11,
       ...,
       1.23911417e+11, 1.21893460e+11, 1.16793126e+11]), array([-1.18828125e+11, -1.20206077e+11, -1.21849353e+11, ...,
       -1.23901351e+11, -1.16992267e+11, -1.19089524e+11])), {'./doc./1.txt', './doc./4.txt', './doc./3.txt'}], [(array([1.19068372e+11,
         1.24239668e+11, 1.19847359e+11, ...,
       1.23804547e+11, 1.21789514e+11, 1.16873726e+11]), array([-1.18743201e+11, -1.20253080e+11, -1.21942847e+11, ...,
       -1.23708926e+11, -1.16829725e+11, -1.19214777e+11])), {'./doc./1.txt', './doc./3.txt'}], [(array([1.19409959e+11, 1.24586504e+11,
         1.19717699e+11, ...,
       1.23752639e+11, 1.21660925e+11, 1.16878073e+11]), array([-1.18639820e+11, -1.20023655e+11, -1.21740652e+11, ...,
       -1.23826783e+11, -1.16842972e+11, -1.18913099e+11])), {'./doc./1.txt', './doc./2.txt'}], [(array([1.19251855e+11, 1.24514294e+11,
         1.19962300e+11, ...,
       1.23855165e+11, 1.21818716e+11, 1.16931569e+11]), array([-1.18844652e+11, -1.20240388e+11, -1.21992009e+11, ...,
       -1.23595539e+11, -1.16915970e+11, -1.19084249e+11])), {'./doc./1.txt', './doc./4.txt', './doc./2.txt', './doc./3.txt'}],
        [(array([1.19283220e+11, 1.24588624e+11, 1.20010548e+11, ...,
       1.23931406e+11, 1.21623326e+11, 1.16871616e+11]), array([-1.18680893e+11, -1.20435685e+11, -1.21880754e+11, ...,
```

陷门算法

- 将查询的关键词集发送给数据拥有者，数据拥有者执行陷门算法生成陷门回传给可信赖用户


hello
world
he
el
ll
lo
o#
wo
or
rl
ld
d#
1 0 1 0 0 1 1 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 1
布隆数组

r为随机值 构建新的向量B'和B''

$$当S[j]=1时B'[j]=\frac{1}{2}B[j]+r,B''[j]=\frac{1}{2}B[j]-r$$

$$当S[j]=0时B'_i[j]=B''[j]=B[j]$$

$$t'=M_1^{-1}.B',t''=M_2^{-1}.B''$$

$$t=(t',t'')$$

t即为输出的陷门

查询

- 通过安全索引和陷门索引得到值v（匹配值），该值表征查询关键词和文档关键词匹配程度
- 对每个文档，如果文档存在有该关键词，将该关键词匹配值加到文档总匹配值中
- 可以得到每个文档关键词的总体匹配值，并根据该值进行排序，返回值最大的文档

$I_i = (I_i', I_i'')$表示第i个关键词生成的安全索引,t表示查询关键词集的陷门

$$R_i = I_i'.t' + I_i''.t''$$

R_i表示第i个关键词的匹配值

D_i表示第i个文档所有关键词的总匹配值

$$D_i = \sum_{w \subset \text{第}i\text{个文档关键词集}} v_w$$

> 根据D_i对文档进行排序

```
./doc./1.txt:[b'on' b'it' b'in' b'no' b'at' b'you' b'am' b'as' b'of' b'he']
./doc./2.txt:[b'on' b'to' b'it' b'in' b'at' b'so' b'am' b'she' b'as' b'he']
./doc./3.txt:[b'or' b'to' b'xh' b'in' b'no' b'do' b'so' b'as' b'and' b'of']
./doc./4.txt:[b'on' b'in' b'pb' b'at' b'him' b'one' b'she' b'as' b'of' b'my']
查询关键词['zy', 'in', 'on']
查询结果[('./doc./4.txt', 64.99999999590585), ('./doc./1.txt', 46.999999996296765), ('./doc./2.txt', 46.99999999563323), ('./doc./3.txt',
  29.99999999686389)]
```

思考

- 该算法可以实现模糊查找，主要是利用对偶编码和布隆过滤器
- 可以修改对偶编码实现不同的模糊查找方式，如使用LSH
- 布隆过滤器主要是为了提升查询的效率

缺点

- 该算法时间消耗主要在于安全索引和陷门的生成，但M1,M2和S的维数由布隆过滤器的维数决定，而布隆过滤器的维数根据其错误率和单词分解的片段个数决定
- 因此，设计一个好的单词分解方式，将可以有效提高该算法的效率