In [1]:

```
file=open('equinix-chicago.dirA.20140320-130000.UTC.anon.pcap.flow.txt','r')
Data=file.readlines()
k=0
for i in Data:
    k=k+1
print(k)
#总的键值对个数为18866027，但单机序号范围740766605远大于总键值对个数
```

18866027

In [2]:

```
def hash1(a,b,c):
    z=(a*b)%c
    return z
def hash2(a,b,c):
    z=(a+b)%c
    return z
def hash3(a,b,c):
    k=int(a&b)%c
    return k
def hash4(a,b,c):
    k=int(a|b)%c
    return k
def hash5(a,b,c):
    k=int(a^b)%c
    return k
hash5(1059533709,740766605,k)
print(type(Data))
#构造五种哈希函数
```

<class 'list'>

In [6]:

```
#哈希冲突测试
#A为hash表
A=[]
#B=[]
for j in range(0,5):
    temp=[]
    for i in range(0,k):
        temp.append(0)
    A.append(temp)
    print('precreate:',20*(j+1),'%')
    #B.append(temp)

z=0
#第一次扫描创建哈希表
for i in range(0,len(Data)):
    if(i%377320==0):
        z=z+1
        print('hash:',(z-1)*2,'%')
    j=Data[i].split()
    t1=hash1(int(j[0]),int(j[1]),k)
    t2=hash1(int(j[0]),int(j[1]),k)
    t3=hash1(int(j[0]),int(j[1]),k)
    t4=hash1(int(j[0]),int(j[1]),k)
    t5=hash1(int(j[0]),int(j[1]),k)
    A[0][t1]=A[0][t1]+1
    A[1][t2]=A[1][t2]+1
    A[2][t3]=A[2][t3]+1
    A[3][t4]=A[3][t4]+1
    A[4][t5]=A[4][t5]+1

#第二次扫描找最值89002
z=0
maxh=0
a=0
b=0
for i in range(0,len(Data)):
    if(i%377320==0):
        z=z+1
        print('findmax:',(z-1)*2,'%')
    j=Data[i].split()
    t1=hash1(int(j[0]),int(j[1]),k)
    t2=hash1(int(j[0]),int(j[1]),k)
    t3=hash1(int(j[0]),int(j[1]),k)
    t4=hash1(int(j[0]),int(j[1]),k)
    t5=hash1(int(j[0]),int(j[1]),k)
    minmax=min(A[0][t1],A[1][t2],A[2][t3],A[3][t4],A[4][t5])
    if(minmax>maxh):#记录当前最大值
        maxh=minmax
        a=j[0]
        b=j[1]
print(a,b)
print(maxh)
```

```
precreate: 20 %
precreate: 40 %
precreate: 60 %
precreate: 80 %
precreate: 100 %
hash: 0 %
hash: 2 %
hash: 4 %
hash: 6 %
hash: 8 %
hash: 10 %
hash: 12 %
hash: 14 %
hash: 16 %
hash: 18 %
hash: 20 %
hash: 22 %
hash: 24 %
hash: 26 %
hash: 28 %
hash: 30 %
hash: 32 %
hash: 34 %
hash: 36 %
hash: 38 %
hash: 40 %
hash: 42 %
hash: 44 %
hash: 46 %
hash: 48 %
hash: 50 %
hash: 52 %
hash: 54 %
hash: 56 %
hash: 58 %
hash: 60 %
hash: 62 %
hash: 64 %
hash: 66 %
hash: 68 %
hash: 70 %
hash: 72 %
hash: 74 %
hash: 76 %
hash: 78 %
hash: 80 %
hash: 82 %
hash: 84 %
hash: 86 %
hash: 88 %
hash: 90 %
hash: 92 %
hash: 94 %
hash: 96 %
hash: 98 %
hash: 100 %
findmax: 0 %
findmax: 2 %
findmax: 4 %
findmax: 6 %
findmax: 8 %
```

```
findmax: 10 %
findmax: 12 %
findmax: 14 %
findmax: 16 %
findmax: 18 %
findmax: 20 %
findmax: 22 %
findmax: 24 %
findmax: 26 %
findmax: 28 %
findmax: 30 %
findmax: 32 %
findmax: 34 %
findmax: 36 %
findmax: 38 %
findmax: 40 %
findmax: 42 %
findmax: 44 %
findmax: 46 %
findmax: 48 %
findmax: 50 %
findmax: 52 %
findmax: 54 %
findmax: 56 %
findmax: 58 %
findmax: 60 %
findmax: 62 %
findmax: 64 %
findmax: 66 %
findmax: 68 %
findmax: 70 %
findmax: 72 %
findmax: 74 %
findmax: 76 %
findmax: 78 %
findmax: 80 %
findmax: 82 %
findmax: 84 %
findmax: 86 %
findmax: 88 %
findmax: 90 %
findmax: 92 %
findmax: 94 %
findmax: 96 %
findmax: 98 %
findmax: 100 %
647067801 1281048150
89002
```

In [8]:

```
#测试
count=0
for i in Data:
    j=i.split()
    if((j[0]==a)and(j[1]==b)):
        count=count+1
    if((j[0]==b)and(j[1]==a)):
        count=count+1
print(count)
```

```
89002
```