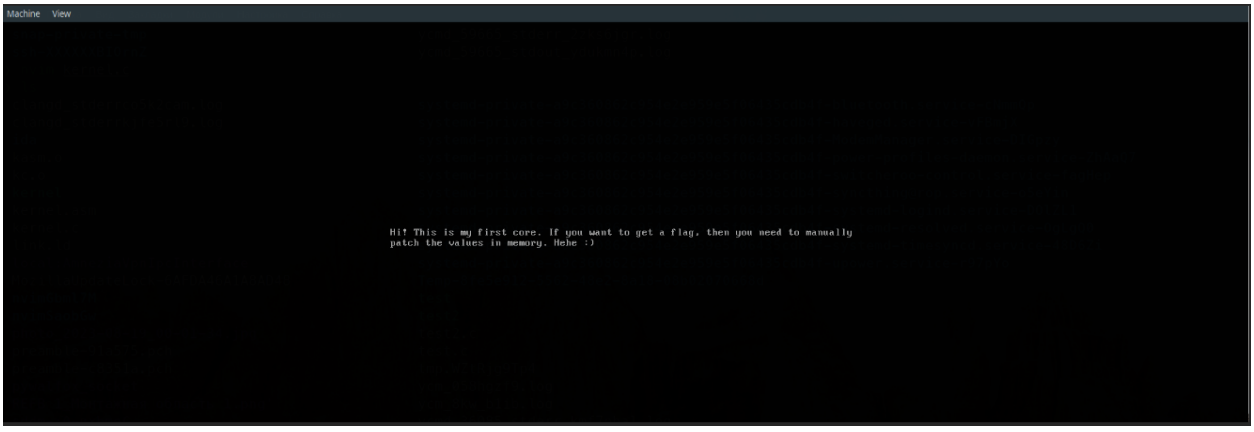
Название:	Моё первое ядро
Категория:	Реверс-инжиниринг
Уровень:	Средний
Очки:	400
Описание:	Попробуй запустить моё первое ядро. Там есть вывод текста :)
Теги:	Ядро, QEMU, патчинг
Автор:	ROP

Прохождение:

Открываем архив, видим файл `kernel`. Его можно запустить через `qemu-system-i386 -kernel kernel`.



Идём тогда в IDA. Изучаем функцию `kmain`. Можем восстановить исходник.

```
void kmain(void)
{
   const char *str = "Hi! This is my first core. If you want
to get a flag, then you need to manually patch the values in
memory. Hehe :)";
   const char *good = "Great! Look in memory before the loop
how they will look as a string, and then add them between cur
ly brackets in CODEBY{
}.";
   char *vidptr = (char*)0xb8000;
   unsigned int i = 0;
   unsigned int j = 0;

   while(j < 80 * 50 * 2) {
       vidptr[j] = ' ';
       vidptr[j+1] = 0x07;
       j = j + 2;
   }

   j = 0;

   while(str[j] != '\\0') {
       vidptr[i] = str[j];
       vidptr[i+1] = 0x07;
       ++j;
       i = i + 2;
   }

       unsigned int dec1 = 0x1;
       unsigned int dec2 = 0x1;
       unsigned int dec3 = 0x1;
       unsigned int dec4 = 0x1;
       unsigned int dec5 = 0;

       for (i = 0; i < dec1; i++) {
```

```
                        dec2 ^= dec3;
                        dec3 += dec4^1337;
                        dec4 -= dec3^dec2;
                        dec2 ^= dec4+dec3;
                        dec3 ^= dec2-dec4;
                        dec4 ^= dec2+dec3;
        }

        if (
                        dec1 == 0x1337 &&
                        dec2 == 0x82b9f885 &&
                        dec3 == 0x5afcb5c &&
                        dec4 == 0x77e64835
                                        ) {
                while(j < 80 * 50 * 2) {
                        vidptr[j] = ' ';
                        vidptr[j+1] = 0x07;
                        j = j + 2;
                }

                j = 0;
                i = 0;

                while(good[j] != '\\0') {
                        vidptr[i] = good[j];
                        vidptr[i+1] = 0x02;
                        ++j;
                        i = i + 2;
                }

        }

    return;
}
```

Нам нужно менять от `dec1` до `dec4`. Можем написать программу, что выдаст нам нужные значения.

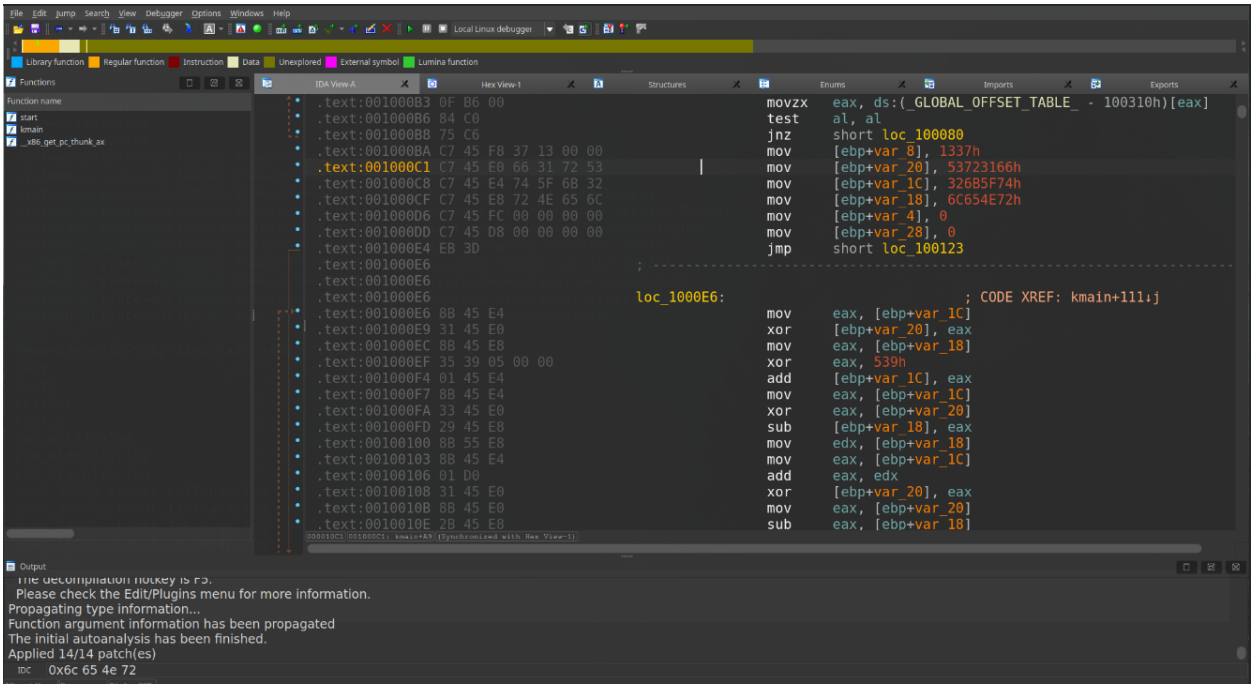
```
#include <stdio.h>

int main() {

	unsigned int dec1 = 0x1337;
	unsigned int dec2 = 0x82b9f885;
	unsigned int dec3 = 0x5afcb5c;
	unsigned int dec4 = 0x77e64835;
	unsigned int dec5 = 0;

	unsigned int i;

	for (i = 0; i < dec1; i++) {
		dec4 ^= dec2+dec3;
		dec3 ^= dec2-dec4;
		dec2 ^= dec4+dec3;
		dec4 += dec3^dec2;
		dec3 -= dec4^1337;
		dec2 ^= dec3;
	}

	printf("%#x\\n", dec1);
	printf("%#x\\n", dec2);
	printf("%#x\\n", dec3);
	printf("%#x\\n", dec4);

}
```

Запускаем.

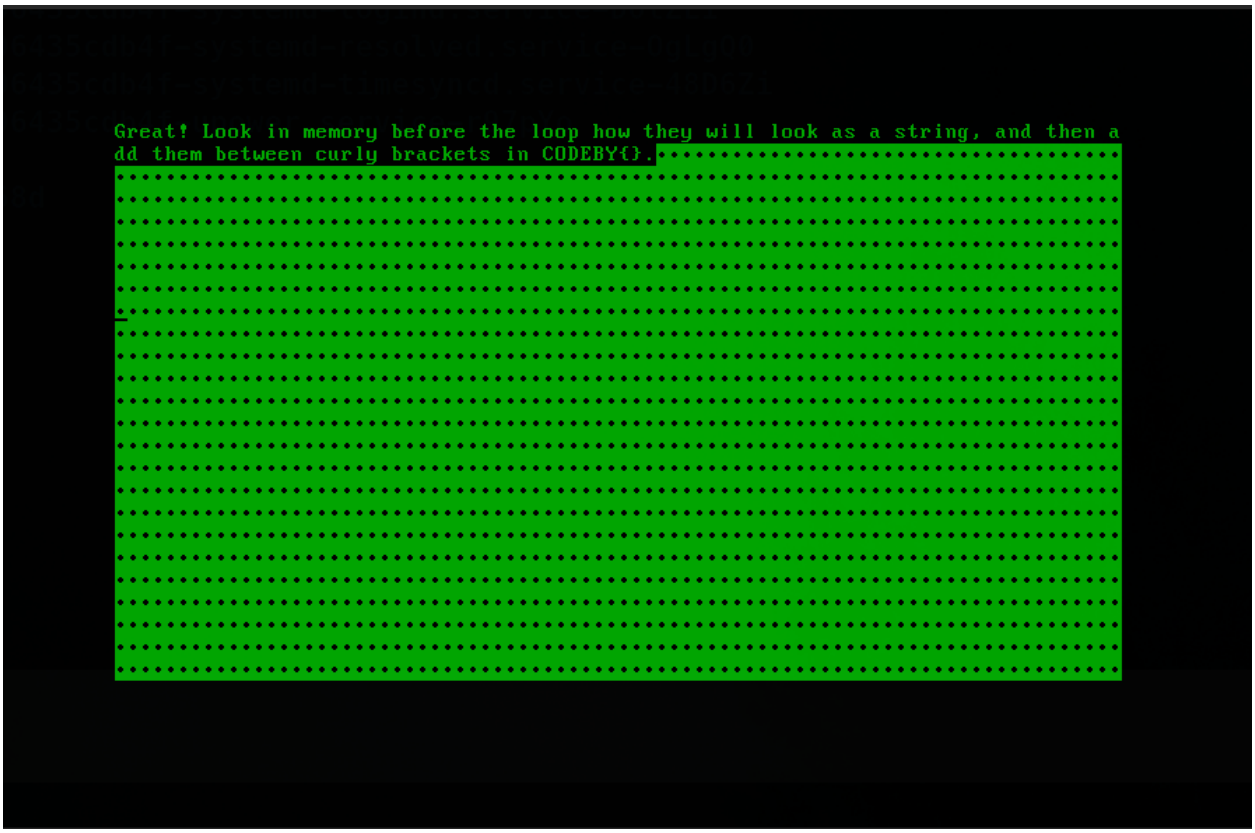
```
0x1337
0x53723166
```

```
0x326b5f74
0x6c654e72
```

Пропатчим ядро и проверим это.

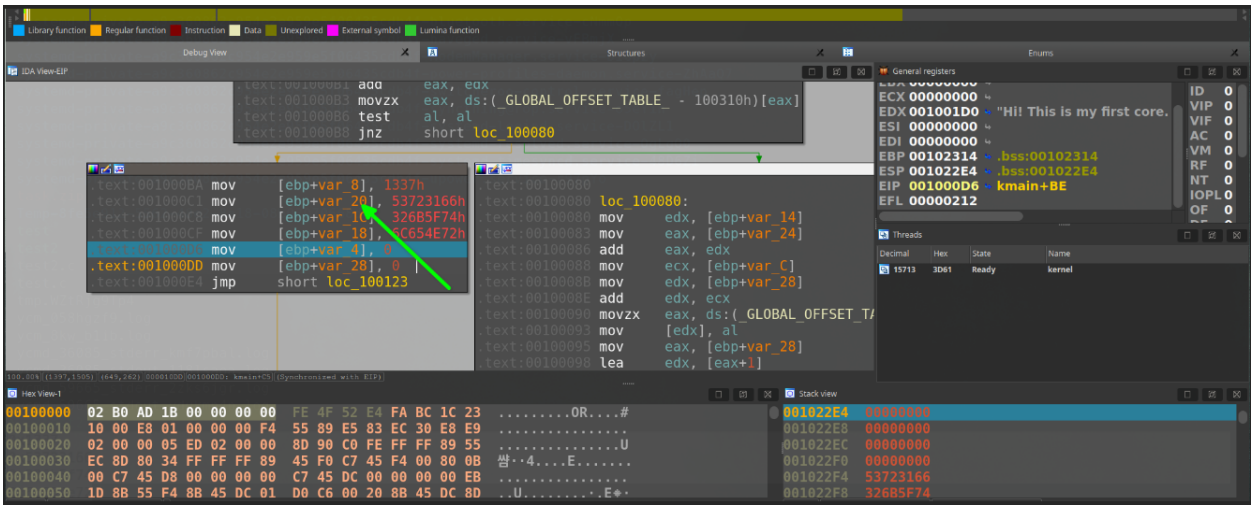


Запуск.



Успех!

Поменяв немного EIP в IDA мы можем посмотреть содержимое памяти и без QEMU.



Нажимаем 2 раза ЛКМ на `var_20`. Там будет флаг.