

事件等待

补充知识：并发是指多个线程在同时执行

单核（是分时分执行，不是真正的同时）

多核（在某一个时刻，会同时有多个线程再执行）

同步则是保证在并发执行的环境中各个线程可以有序的执行

```
DWORD dwVal = 0;    //全局变量

线程中的代码：

    dwVal ++;        //只有一行 安全吗？

对应的汇编代码：

    mov     eax,[0x12345678]
    add     eax,1
    mov     [0x12345678],eax
```

线程中的代码访问全局变量，无论单核多核，线程切换时，可能会产生错误

单核（时间片切换线程）多核（多线程并发）

INC DWORD PTR DS:[0x12345678]

改成

LOCK INC DWORD PTR DS:[0x12345678]

参考：kernel32.InterlockedIncrement

原子操作相关的API：

InterlockedIncrement

InterlockedDecrement

InterlockedExchange

InterlockedCompareExchange

InterlockedExchangeAdd

InterlockedFlushSList

InterlockedPopEntrySList

InterlockedPushEntrySList

//一行汇编代码，安全吗？

单核安全
多核还是无法避免多线程同时执行代码

自旋锁

参考：KeAcquireSpinLockAtDpcLevel（多核环境）

```

8 loc_469A08:                                ; CODE XREF: KeAcquireSpinLockAtDpcLevel(x)+14j
8 lock bts dword ptr [ecx], 0 ; 检测ecx的值 如果值为0 设置 CF=1 否则 CF=0
8                                ; 然后 将ecx指向变量的第0个位置置1
D jb short loc_469A12 ; 如果 [ecx]不为0 跳转
F retn 4
2 ;
2
2 loc_469A12:                                ; CODE XREF: KeAcquireSpinLockAtDpcLevel(x)+9fj
2                                ; KeAcquireSpinLockAtDpcLevel(x)+18j
2 test dword ptr [ecx], 1
8 jz short loc_469A08 ; 当 [ecx]=0 时 跳转
A pause
C jnp short loc_469A12
C _KeAcquireSpinLockAtDpcLevel@4 endp

```

自旋锁指令只有在多核中才有用，loc_469A12函数等待线程切换回来，不停的循环判断

1、自旋锁只对多核有意义。

(查看不同版本的KeAcquireSpinLockAtDpcLevel函数)

2、自旋锁与临界区、事件、互斥体一样，都是一种同步机制，都可以让当前线程处于等待状态，区别在于自旋锁不用切换线程。

可等待对象

在Windbg中查看如下结构体：

dt_KPROCESS	进程
dt_KTHREAD	线程
dt_KTIMER	定时器
dt_KSEMAPHORE	信号量
dt_KEVENT	事件
dt_KMUTANT	互斥体
dt_FILE_OBJECT	文件

这些内核结构体中都包含DispatchHeader



1.创建事件对象：信号

CreateEvent（NULL,TRUE,FALSE,NULL）