在Libev中，如果某种结构的数组需要扩容，它使用array\_needsize宏进行处理，比如：

array\_needsize **(**int**,** fdchanges**,** fdchangemax**,** fdchangecnt**,** EMPTY2**);**

这就表示要将整型（int）数组fdchanges，由原来的fdchangemax个元素扩容为fdchangecnt，新扩容的内存空间使用EMPTR2进行初始化。

array\_needsize宏定义如下：

**#**define array\_needsize**(type,**base**,**cur**,**cnt**,**init**)** \

**if** **(**expect\_false **((**cnt**)** **>** **(**cur**)))** \

**{** \

int ecb\_unused ocur\_ **=** **(**cur**);** \

**(**base**)** **=** **(type** **\*)**array\_realloc**(**sizeof**(type),** **(**base**),** **&(**cur**),** **(**cnt**));** \

init **((**base**)** **+** **(**ocur\_**),** **(**cur**)** **-** ocur\_**);** \

**}**

base是type类型的数组，当前有cur个元素，需要调整到cnt个元素，新扩充的内存空间使用init函数初始化。

该宏的关键在于array\_realloc函数，它的实现如下：

static void **\*** array\_realloc **(**int elem**,** void **\***base**,** int **\***cur**,** int cnt**)**

**{**

**\***cur **=** array\_nextsize **(**elem**,** **\***cur**,** cnt**);**

**return** ev\_realloc **(**base**,** elem **\*** **\***cur**);**

**}**

该函数中，首先使用array\_nextsize计算最终的元素个数，然后调用ev\_realloc申请空间。array\_nextsize计算新元素个数的方法如下：

/\* find a suitable new size for the given array, \*/

/\* hopefully by rounding to a nice-to-malloc size \*/

int array\_nextsize **(**int elem**,** int cur**,** int cnt**)**

**{**

int ncur **=** cur **+** 1**;**

**do**

ncur **<<=** 1**;**

**while** **(**cnt **>** ncur**);**

*/\* if size is large, round to MALLOC\_ROUND - 4 \* longs to accommodate malloc overhead \*/*

**if** **(**elem **\*** ncur **>** MALLOC\_ROUND **-** **sizeof** **(**void **\*)** **\*** 4**)**

**{**

ncur **\*=** elem**;**

ncur **=** **(**ncur **+** elem **+** **(**MALLOC\_ROUND **-** 1**)** **+** **sizeof** **(**void **\*)** **\*** 4**)** **&** **~(**MALLOC\_ROUND **-** 1**);**

ncur **=** ncur **-** **sizeof** **(**void **\*)** **\*** 4**;**

ncur **/=** elem**;**

**}**

**return** ncur**;**

**}**

该函数中，首先得到一个比cnt大的偶数ncur，如果ncur个元素占用的空间（elem \* ncur + sizeof (void \*) \* 4）大于MALLOC\_ROUND(4096)个字节，则需要调整ncur。

这里之所以要加上sizeof (void \*) \* 4，是因为malloc在申请空间时，除了申请的字节数之外，它还会在内存块之外加上额外的空间，记录当前内存块的信息，也就是sizeof (void \*) \* 4个字节。

调整ncur的方法，主要是下面的语句：

ncur **\*=** elem**;**

ncur **=** **(**ncur **+** elem **+** **(**MALLOC\_ROUND **-** 1**)** **+** **sizeof** **(**void **\*)** **\*** 4**)** **&** **~(**MALLOC\_ROUND **-** 1**);**

它的主要作用，就是使得ncur向上调整成MALLOC\_ROUND的倍数。这里的ncur代表的是最终申请空间的总字节数。因此，还需要将其调整为元素个数：

ncur **=** ncur **-** **sizeof** **(**void **\*)** **\*** 4**;**

ncur **/=** elem**;**

得到最终的元素个数之后，接下来就是调用ev\_realloc申请空间了，它的实现如下：

static void **\***ev\_realloc\_emul **(**void **\***ptr**,** long size**)** EV\_THROW

**{**

*/\* some systems, notably openbsd and darwin, fail to properly*

*\* implement realloc (x, 0) (as required by both ansi c-89 and*

*\* the single unix specification, so work around them here.*

*\* recently, also (at least) fedora and debian started breaking it,*

*\* despite documenting it otherwise.*

*\*/*

**if** **(**size**)**

**return** realloc **(**ptr**,** size**);**

free **(**ptr**);**

**return** 0**;**

**}**

static void **\*(\***alloc**)(**void **\***ptr**,** long size**)** **=** ev\_realloc\_emul**;**

void **\***ev\_realloc **(**void **\***ptr**,** long size**)**

**{**

ptr **=** alloc **(**ptr**,** size**);**

**if** **(!**ptr **&&** size**)**

**{**

fprintf **(**stderr**,** "(libev) cannot allocate %ld bytes, aborting."**,** size**);**

abort **();**

**}**

**return** ptr**;**

**}**

PS：Libev的代码就分析到这了！