<https://blog.csdn.net/luotuo44/article/details/38300965>

**转载请注明出处:**[**http://blog.csdn.net/luotuo44/article/details/38300965**](http://blog.csdn.net/luotuo44/article/details/38300965)

        如果你打开Libevent的一些文件，比如util.h文件。就会发现使用了很多宏定义，并根据一些宏定义而进行条件编译。这些宏定义往往来自event-config.h文件中。

        如util.h文件的代码开始处：

**[cpp]** [view plain](https://blog.csdn.net/luotuo44/article/details/38300965) [copy](https://blog.csdn.net/luotuo44/article/details/38300965)

1. #ifdef \_EVENT\_HAVE\_SYS\_TIME\_H
2. #include <sys/time.h>
3. #endif
4. #ifdef \_EVENT\_HAVE\_STDINT\_H
5. #include <stdint.h>
6. #elif defined(\_EVENT\_HAVE\_INTTYPES\_H)
7. #include <inttypes.h>
8. #endif

        其会根据是否定义了某个宏，而决定是否包含某个头文件。从宏的名字来看，其指明了是否有这个头文件。有时还会指明是否有某个函数。这样做的原因很简单，因为Libevent是跨平台的，必须得考虑到某些系统可能没有一些头文件或者函数。

        event-config.h文件是一个很基础和重要的文件。在文件的一开始有这样一句"This file was generated by autoconf when libevent was built"。这说明这个文件是在Libevent配置的时候生成的，即在编译Libevent之前就应该要生成该文件了。当然也早于我们在Libevent基础上编写应用程序。

        其在编译之前就检查所在的系统的一些情况。比如是否含有某个文件或者函数。其对这些进行检测，然后把结果写入到event-config.h文件中。等到编译Libevent和编译我们的APP时，会include该头文件。

        PS：上面两段的说法有点错误。待修改。

        该文件大部分内容是根据config.h.in文件生成的。比如，config.h.in文件里面有下面的代码：

**[cpp]** [view plain](https://blog.csdn.net/luotuo44/article/details/38300965) [copy](https://blog.csdn.net/luotuo44/article/details/38300965)

1. /\* Define to 1if you have the <arpa/inet.h> header file. \*/
2. #undefHAVE\_ARPA\_INET\_H
3. .....
4. /\* Define ifyour system supports the epoll system calls \*/
5. #undefHAVE\_EPOLL

        对应地，Linux内核版本在2.6以上的Linux对应生成的event-config.h文件会定义这两个宏，如下：

**[cpp]** [view plain](https://blog.csdn.net/luotuo44/article/details/38300965) [copy](https://blog.csdn.net/luotuo44/article/details/38300965)

1. /\* Define to 1if you have the <arpa/inet.h> header file. \*/
2. #define\_EVENT\_HAVE\_ARPA\_INET\_H 1
3. ......
4. /\* Define to 1if you have the <sys/epoll.h> header file. \*/
5. #define\_EVENT\_HAVE\_SYS\_EPOLL\_H 1

        而在Windows系统下生成的event-config.h文件就没有定义这两个宏。

        可以说，event-config.h这个文件定义的宏指明了所在的系统有哪些可用的头文件、函数和一些配置。

        又比如对于gcc来说，是支持\_\_func\_\_这个宏的，但对于VS编译器就不支持，VS对应功能的宏为\_\_FUNCTION\_\_。此时在Windows系统的event-config.h文件中，就会定义：

**[cpp]** [view plain](https://blog.csdn.net/luotuo44/article/details/38300965) [copy](https://blog.csdn.net/luotuo44/article/details/38300965)

1. /\* Define toappropriate substitue if compiler doesnt have \_\_func\_\_ \*/
2. #define\_EVENT\_\_\_func\_\_ \_\_FUNCTION\_\_

        而在util-internal.h文件中，有这样的定义：

**[cpp]** [view plain](https://blog.csdn.net/luotuo44/article/details/38300965) [copy](https://blog.csdn.net/luotuo44/article/details/38300965)

1. #ifdef\_EVENT\_\_\_func\_\_
2. #define \_\_func\_\_\_EVENT\_\_\_func\_\_
3. #endif

        这样就可以在其他文件中通用\_\_func\_\_宏了，无需关注是什么系统了。

        event-config.h文件的有些内容是根据编译Libevent时的配置选项生成的。比如是否支持多线程这个选项。如果配置Libevent的时，加入了这样一句

**[cpp]** [view plain](https://blog.csdn.net/luotuo44/article/details/38300965) [copy](https://blog.csdn.net/luotuo44/article/details/38300965)

1. ./configure--disable-**thread**-support

        那么，在event-config.h文件将定义DISABLE\_THREAD\_SUPPORT这个宏，此时得到的Libevent是不支持多线程的。

版权声明：本文为博主原创文章，未经博主允许不得转载。 https://blog.csdn.net/luotuo44/article/details/38300965