目录[+]

ecryptfs.h

- 作者：邢万里
- 时间：6/10/2016 1:07:39 PM

一、 宏定义

```
#define ECRYPTFS_DEFAULT_IV_BYTES 16
```

解释：eCryptfs默认IV(Initialization Vector)字节大小为16 Octets。
作用：用于初始化数据结构ecryptfs_crypt_stat的成员iv_bytes。
意义：初始化向量用于文件加密部分。根初始化向量(Root IV)随着每次文件打开由系统随机生成，每次均重新生成Root IV。而初始化向量(IV)不同于根初始化向量，顾名思义，即IV是由Root IV得到的；eCryptfs文件是以extent为单位进行划分的（不同文件系统的文件划分方式不同，有的以block块方式，有的以page页方式等），不同的extent有不同的offset偏移量（也可称index索引），因此，eCryptfs将offset+Root IV通过MD5加密方式生成对于extent的IV值（很显然，每个extent的offset和IV均不同）。

```
#define ECRYPTFS_DEFAULT_EXTENT_SIZE 4096
```

解释：eCryptfs默认文件中（不包括头部）每一个extent大小为4096 Octets（即4KB）。
作用：用于初始化数据结构ecryptfs_crypt_stat的成员extent_size，但是该成员可能在网络通信中传递包时过程中修改此值。
意义：上面已经提到，eCryptfs文件是以extent进行划分，从而eCryptfs需要对文件进行处理时（如：加密、解密等），以逐extent方式进行一一处理。

```
#define ECRYPTFS_MINIMUM_HEADER_EXTENT_SIZE 8192
```

解释：eCryptfs定义头部(header)的extent大小为8192 Octets（8KB）。
作用：用于初始化数据结构ecryptfs_crypt_stat的成员metadata_size。
意义：在版本0(Version 0)的eCryptfs中，通过set_default_header_data()函数默认定义metadata为该宏定义值，即8KB。而当eCryptfs升级后，该函数只是用于兼容(backwards compatibility)版本0。升级后的版本中metadata_size默认大小是根据元数据存在的具体位置而定，如果metadata存在头部且页大小小于该宏定义的值，则metadata_size为该宏定义；如果metadata存在扩展属性中，同上。
注：metadata即为元数据，是存放在文件的头部或者文件尾部(即，xattr：扩展属性)中。

```
#define ECRYPTFS_DEFAULT_MSG_CTX_ELEMS 32
```

解释：eCryptfs默认消息(MSG:message)内容(CTX:context)中的成员(ELEMS:elements)个数为32个。更深一层的说，就是消息由很多个buffer缓冲块组成，此处定义的是默认的缓冲块数量。
作用：用于初始化ecryptfs_message_buf_len变量。代码如下：

```
    " unsigned int ecryptfs_message_buf_len = ECRYPTFS_DEFAULT_MSG_CTX_ELEMS; "
```

意义：编写一个内核模块则通过module_param()进行传递参数。该变量用于代码
” module_param(ecryptfs_message_buf_len, uint, 0); “中，ecryptfs_message_buf_len为参数名，uint为其类型，0为访问参数的权限。

```
#define ECRYPTFS_DEFAULT_SEND_TIMEOUT HZ
```

解释：eCryptfs默认发送消息（或者叫做包）的超时时间为HZ。
注：内核源码中没有该值的使用。

```
#define ECRYPTFS_MAX_MSG_CTX_TTL (HZ*3)
```

解释：eCryptfs定义最大的消息内容存在时间为3HZ。TTL全称Time To Live，为IP协议包中的一个值，代表数据包在网络中的时间是否太长而应被丢弃（因为存在因为一些原因
作用：用于初始化ecryptfs_message_buf_len变量。代码如下：

```
    " signed long ecryptfs_message_wait_timeout = ECRYPTFS_MAX_MSG_CTX_TTL / HZ; "
```

意义：编写一个内核模块则通过module_param()进行传递参数。该变量用于代码
” module_param(ecryptfs_message_wait_timeout, long, 0); “中（解释同上）。该参数定义了内核等待应用层的ecryptfsd守护进程回应（或称消息、包等）的最大等待时间。

```
#define ECRYPTFS_DEFAULT_NUM_USERS 4
```

解释：eCryptfs默认同时并行使用eCryptfs的用户数量为4个。
作用：用于初始化ecryptfs_message_buf_len变量。代码如下：

```
    " unsigned int ecryptfs_number_of_users = ECRYPTFS_DEFAULT_NUM_USERS; "
```

意义：编写一个内核模块则通过module_param()进行传递参数。该变量用于代码” module_param(ecryptfs_number_of_users, uint, 0); “中（解释同上）。

```
#define ECRYPTFS_MAX_NUM_USERS 32768
```

解释：eCryptfs定义最大用户数量为32768个。

```
#define ECRYPTFS_XATTR_NAME "user.ecryptfs"
```

解释：eCryptfs定义扩展属性名字为user.ecryptfs。扩展属性名是以null结尾的字符串，这个名字通常包括一个命名空间前缀。 扩展属性的值是一个任意指定长度的文本或二进制数据。
意义：和setxattr()、getxattr()有关。getxattr()：列出扩展属性所对应的值，该函数的返回值是扩展属性值的长度。
注：这里简单介绍下getxattr()和setxattr()函数。

1. getxattr()函数
   函数原型：

```
ssize_t getxattr(const char *path,          //路径
                       const char *name,    //扩展属性名字
                      void *value,          //扩展属性所对应的值
                       size_t size);        //扩展属性的长度
```

   Used by the VFS to copy into @value(dst) the value of the extended attribute @name for the specified file.

2. setxattr()函数
   函数原型：

```
setxattr(path,      //路径
         key,       //扩展属性的名字
        value,      // 扩展属性的值
        size,       //扩展属性的长度
      flags);       //标识
```

   Used by the VFS to set the extended attribute @name(dst) to the @value on the file referenced by dentry.

```
#define ECRYPTFS_PREPARE_COMMIT_MODE 0"
```

解释：eCryptfs准备提交模式，用0代表。
注：内核源码中没有该值的使用。

```
#define ECRYPTFS_WRITEPAGE_MODE 1"
```

解释：eCryptfs写页模式，用1代表。
注：内核源码中没有该值的使用。