说明：仅包含程序分析部分。

由于题目要求，在原始样本增加了难度，思路如下：

和之前一样关键代码不在main函数，但是main函数由两个函数构成，在while无穷循环里面的是fake函数，完全没有用。

在while循环前的函数，会读取一个running\_pass的文件，只取25个字符，然后进行一个5轮的判断，如果判断都通过了，且前7个字符是 Red{EV3，那么就把一个全局变量设置上。该变量影响后续的加解密。

该判断实际是一个5皇后问题，25个字符对应了摆放的位置。该问题多解，一共有10个解，确定前7个字符用于限定解。

加解密流程替换为一个随机的算法，从ecb，cbc，cfb，ofb算法中，按照设置的全局变量进行排序，然后根据随机生成的key的第3位来选择3个要用的算法，依次进行3轮加密。

IV的生成是从key用有限域乘法乘上一个magic number计算的，解密时获得iv之后需要乘上“magic number在有限域的逆”来逆推计算。

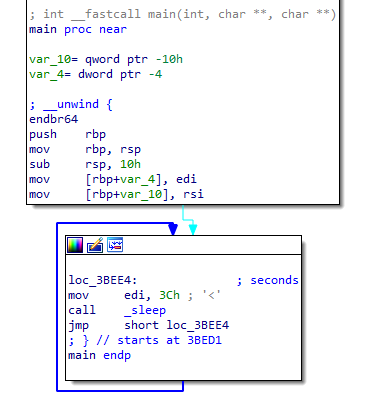
所以选手拿到题目，需要先破解5皇后问题，拿到解，从而知道ecb，cbc，cfb，ofb算法的排布。

然后计算magic number在有限域上的逆，从而知道每个文件的解密key。

最后逆写加密流程编写脚本自动3轮解密，得出明文。

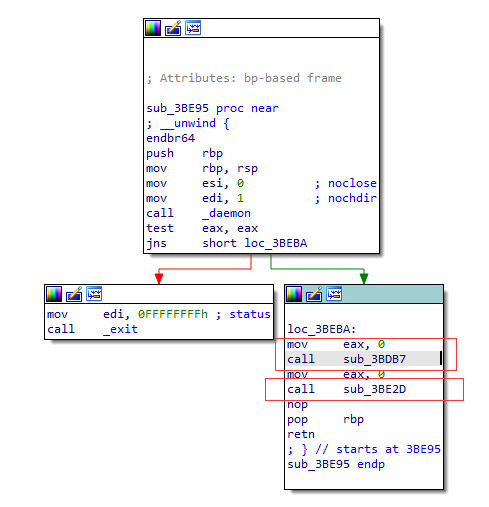
原样本思路：

**【首先寻找程序的正确执行流程】**，这个程序一打开的时候，会看见main函数非常的简单，就只是一个死循环， 都不具备正常的程序功能：

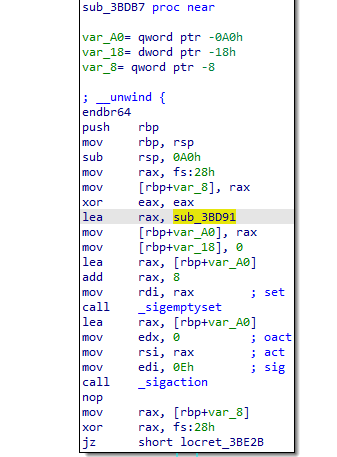
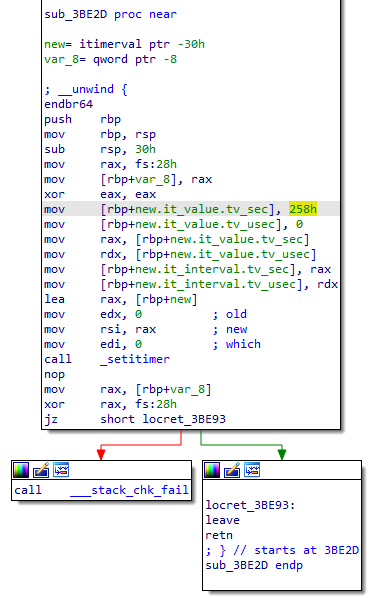


这个显然不正常，说明该恶意程序采用了一定的控制流隐藏手段。该程序将真正的行为执行流隐藏在了init流程中，该执行流会先于main函数执行，因此不容易第一时间察觉。

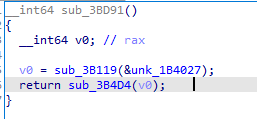
恶意程序真正的入口点是sub\_3BE95，该程序入口并不复杂。如下图所示，只有红框内两个函数需要具体分析。



【**分析程序真正的执行流**】其中第一个函数sub\_3BDB7是信号量相关的处理，其中sigaction操作将函数sub\_3BD91注册为了处理第14号信号量（0Eh）。而在第二个函数sub\_3BE2D中，设置了每10分钟（258h秒）发送一次信号量，该信号量正好就是第14号信号量。因此，sub\_3BDB7和sub\_3BE2D这两个函数相当于设置了一个定时器，每10分钟执行一次sub\_3BD91函数。因此后续的分析重点是sub\_3BD91函数。

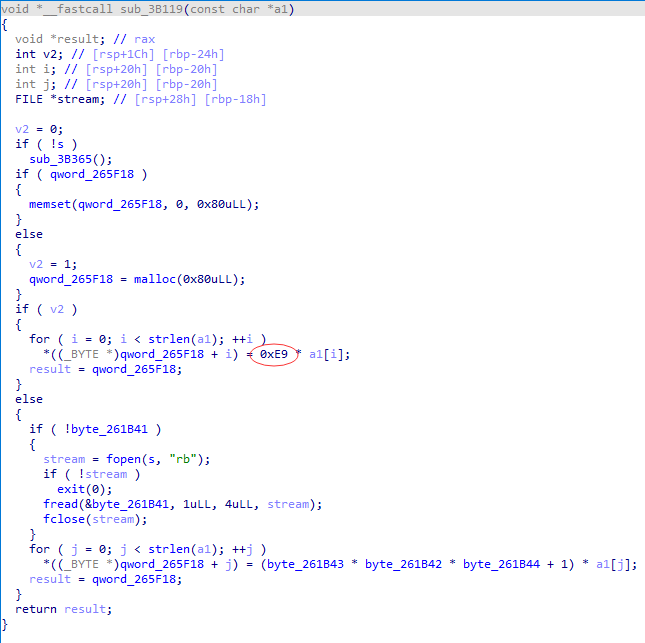
 

sub\_3BD91看起来也很简单，只有两个函数，我们一个一个分析：



【**分析混淆算法**】第一个函数sub\_3b119需要仔细分析，伪代码见下图。具体分析功能，该函数是一个字符串解混淆函数，参数a1就是待解密的字符串。比较复杂的是解混淆使用的秘钥：首先它会打开s中存储的文件名，然后读取该文件的前4个字节，其中第2、3、4个字节的乘积加1会作为秘钥，与a1中的每一个字符进行乘积，然后将解密的秘钥放在qword\_265F18处，返回供使用。实际上，这个秘钥也提示了，就是

s中存储的信息来自sub\_3B365，而且也需要解混淆字符串。该流程使用的秘钥固定为0xE9。解混淆之后发现s中存的信息是读取”/proc/self/exe”的结果，也就是该可执行程序自己。而Linux可执行程序前四个字节是固定的，第2~4字节就是字符“ELF”，所以该秘钥（byte\_261B43 \* byte\_261B42 \* byte\_261B44 + 1）就是一个固定值，在保留最低8比特（bit）之后也是0xE9。



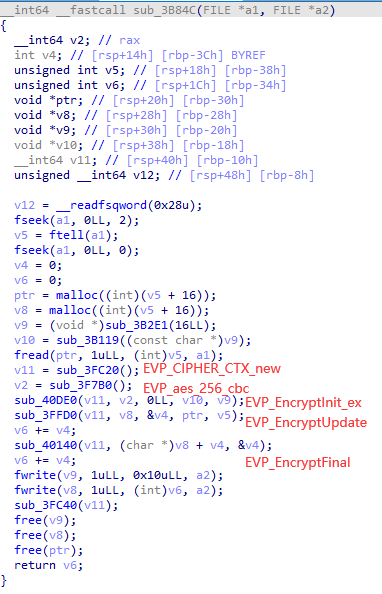
【**分析扫描器**】得到秘钥之后就可以进行解密，sub\_3BD91中使用的unk\_1B4027解密结果是字符串”.”代表当前目录。该字符串作为参数输入了sub\_3B4D4。所以接下来需要分析sub\_3B4D4。

sub\_3B4D4将输入的参数作为目录，遍历该目录下所有文件。如果是普通文件且不是以“.dp”结尾，那么就会调用sub\_3BBF1进行处理。

sub\_3BBF1恶意代码中负责文件加解密的函数。会打开该文件，调用sub\_3B84C进行加密。然后将加密数据存储在以“.dp”结尾的新文件中，并且删除原文件。

【**分析加密算法**】因此要想解密文件，就要具体分析sub\_3B84C是如何进行加密的。

这里面明显用到了一些openssl的函数，需要对应一下库函数，分析之后，具体对应如下：



所以该函数使用的是AES-CBC-256位算法进行的加密。该算法需要一个秘钥key和一个IV。IV是变量v9，是16字节的全随机数（sub\_3B2E1中生成）。而key也是动态生成的，是由sub\_3B119处理IV得到。sub\_3B119我们之前已经分析过，作用是字符串解混淆。

【**总结**】至此，该恶意程序的功能就分析完成，总结如下：

该恶意程序每10分钟扫描一遍当前目录，将目录中未加密的文件用AES -256算法的CBC模式加密转储，之后删除原文件。加密文件的首16字节是CBC模式的IV，IV经过特殊处理是加密使用的Key。基于此我们就可以编写相应的解密程序。