Encryption 101系列：破解之道

Encryption 101系列：破解之道

原文：https://blog.malwarebytes.com/threat-analysis/2018/03/encryption-101-how-to-break-encryption/

在之前的系列文章中，我们为恶意软件分析人员介绍了加密技术的入门知识，并以ShiOne勒索软件为例，演示了其使用的加密技术。现在，让我们开始学习加密的破解之道

对于加密系统的实现人员来说，有可能在许多方面出现纰漏。所以，如何识别和分析程序员所用的加密算法，并找出可资利用的弱点，就是破解加密技术的难点所在。

当然，这些弱点也可能是弱加密算法、弱密钥生成器、服务器端漏洞和泄漏密钥，等等。

0x00 定位加密算法

要想找出加密技术的弱点，必须首先知道其所用的加密算法是什么。不过读者不要担心，因为在很多时候，只要查看API调用就行了——在这种情况下，识别算法就会非常简

然而，有时候，加密算法会静态编译到恶意软件中，甚至会使用自定义加密算法。在这种情况下，只有理解加密算法的内部工作方式后，才能识别出相应的实现代码。

一般情况下，恶意软件首先会对文件的内容进行加密，然后再写回文件中，因此，为了缩小加密代码的搜索范围，一种便捷的方法是直接对ReadFile和WriteFile API调用进行xref处理。这是因为，加密算法的实现代码通常会位于这两个API调用之间。

0x01 识别加密代码

正如我们所提到的那样，当查找静态编译的加密代码时，就别指望能找到任何API调用了。这时，就需要设法了解这些加密算法工作机制的低层细节了。

在下图中，我们给出了AES算法的抽象（而非低层细节）流程。一般来说，大多数同步加密算法的流程与此类似；不同之处，主要在于所执行的数学运算的类型，但核心概念

AES是一种对称加密算法，它会针对下列三种数据对象执行一系列的数学和逻辑运算：

1. □ 待加密的明文数据
2. □ 作为算法一部分的静态字节（查找表）

   □ 用于加密的密钥

   根据AES和密钥长度的不同，处理流程会稍有不同。在上图中，有一个包含多个构造块的循环结构：

   □ 轮密钥加

5. □ 行移位
6. □ 字节代换
7. □ 列混合

在上述流程中，文件数据被读入一个字节数目固定的矩阵中。就本例来说，它是16个字节，但是，这个数字的大小具体取决于算法，也就是说，可以是任意数量。下面我们对

- □ 轮密钥加操作将输入数据矩阵中的数据与密钥数据进行异或运算。
- □ 行移位操作将矩阵中的每个横列进行循环式移位。举例来说，如果原来某一行中的数据为4 5 2 1，那么如果左移1位的话，它将变成5 2 1 4；如果再左移1位的话，它将变成2 1 4 5。
- □ 字节代换操作涉及到算法中内置的静态字节数组。（译者注：通过非线性的替换函数，用查找表的方式把每个字节替换成对应的字节。）前面步骤中的每个数据字节都用
- □ 在列混合操作中，会对矩阵中的字节进行一些数学运算和线性变换，使矩阵的每个字节不同于原来状态。

这四种操作，每执行一遍，称为进行了一轮。AES可能会执行10到14轮。这意味着，在二进制文件中寻找加密代码时，它们可能是一个非常冗长的函数，因为其中存在很多重

下面给出另一个进行一轮加密的例子，它可能源自不同类型的AES或类似的同步加密算法：

正如你所看到的，这里的操作顺序稍有不同。这些细节对我们来说不是太重要，因为我们不是密码学家。一般来说，我们并不奢望能够发现AES算法本身的弱点，相反，我们

之前，我们曾经对Scarab勒索软件进行过安全分析。实际上，上面的代码就是来自这个恶意软件。该软件就是使用静态编译的AES-no API调用来加密文件的。我们必须对各种加密方法的内部工作都进行深入的了解，只有这样，才能够弄清楚算法到底要做什么。

这个函数中操作集的数量，正是判断这段代码属于哪种算法的一个主要指标。

在这里，我们依旧使用上一篇文章中的这个图像，之所以这样做，仅仅是为了提醒大家单个勒索软件中常常会使用多种加密方法。遇到这种情况时，一定要多加留意，不要被

问题在于，从技术上讲，只要恶意软件作者愿意的话，可以使用任意数量的加密组合。所以，我们必须能够理解和识别每种加密技术，以及它们在整个程序中扮演的角色。这

0x02 随机数发生器

寻找加密技术弱点的时候，最好从检查加密密钥生成器开始着手，因为在大多数情况下，这些密钥生成器只是某种形式的随机数生成器而已。

如果你曾经阅读过关于加密技术方面的介绍的话，很可能已经听说过随机数生成器的重要性了。原因在于，如果可以强制随机数生成器的输出重现先前加密过程中生成的相同

下面，我们将为读者介绍一个具体的例子。在这个例子中，系统时间被用作弱随机数发生器的种子。

在大多数情况下，任何计算机算法都只能执行有限的一系列操作，并且
，如果函数的输入相同，那么输出也必须相同——这很合乎逻辑。但是，在随机数生成器的情况下，其新颖之处在于，如果为其提供适当的输入（即这些输入值足够丰富多样

正如你在上面看到的，除了使用时间之外，更可靠的随机数发生器还可以对音频数据采样，并且使用鼠标输入和许多其他元素作为其输入值，从而提高输入的随机性。这时候

0x03 破解弱RNG的理论过程

接下来，我们给出一个破解勒索软件的加密算法的理论示例，在这个例子中，该软件使用了一种称为RNG的弱生成器。假设该勒索软件使用以微秒为单位的当前时间作为RN

- □ 网络管理员对该勒索软件进行了相应的分析，并发现用于加密的公钥被用作勒索软件的受害者ID。
- □
  网络管理员能够确定该软件感染网络的大致时间，这个可以通过查看网络日志获悉。假设感染发生在上午10:00:00到上午10:00:10之间的某个时刻，那么这里的时间窗口
- □ 由于RNG使用微秒计时，因此大概有10,000,000个可能的种子。
- □ 管理员然后就可以判断："如果该勒索软件使用时间作为种子值x，那么加密代码就会产生密钥对值KEY~x~。"
- □ 他从10:00:00这一刻开始，逐个尝试以微秒计时的所有时间，并通过一些标准软件来创建密钥对。
- □ 现在，检查它是否与获得的公钥（受害者ID）相匹配。
- □ 很明显，这里并不匹配。这意味着RNG并非使用x（10:00:00 AM）作为种子。
- □ 继续以x + 1方式进行尝试，直到10:00:10之前的最后一个微秒时刻为止。
- □ 最终，将找到一个匹配的时间——生成的公钥与受害者ID匹配的那个时刻。
- □ 现在，会发现生成的私钥与加密硬盘驱动器时生成的私钥是相同的。
- □ 这样，就可以利用这个私钥，通过现成的解密软件来恢复原始文件了。

在这种情况下，暴力破解是完全可行的。不过，如果RNG使用的是毫秒，再结合在给定时间运行的进程数量的话，则会增加一点复杂性。因为这就需要将最初的10,000,000

如果添加了足够多的参数，或参数具有更多的可能结果的话，那么这个数字最终会变得非常之大，从而导致破解者在有生之年都无法完成暴力破解工作，具体如下所示。

0x04 破解实例

以下是一些已经被成功破解勒索软件以及所使用的破解方法。

□ 7ev3n、XORist、Bart：弱加密算法

□ Petya：加密实现中的漏洞

□ DMA Locker、CryptXXX：弱密钥生成器

□ Cerber：服务器端漏洞

□ Chimera：泄漏密钥

0x05 弱加密算法

DES算法是在20世纪70年代问世的，目前已经广泛用于各种加密环境。根据其密钥长度来看，现在只能归类为弱加密算法了——对于加密算法来说，生成的密钥的比特数是

当然，对于普通的恶意软件分析人员来说，不一定能够获得如此强大的计算资源。之所以举这个例子，只是想让大家更好地理解为什么某些加密算法会被认为是弱的。

通常，只需查看文件的可视化图像，就能对加密方法的强度有一个初步的判断。

如您所见，这里的熵值较低，并且加密文件中的数据与原始明文具有明显的相似性。这可能是由于异或类型加密算法本身，或者其他原因所导致的。

为了进行比较，让我们来看看使用另一种算法加密的文件的效果。我们不难看出，这里的数据具有更高的熵：

当判断给定的勒索软件是否能够进行破解时，也可以首先从文件可视化开始着手，因为它可以帮助我们判断出该软件使用的算法及其加密的强度。同时，在破解加密的过程中

0x06 小结

在本文中，我们首先阐释了加密算法的识别和分类在寻找加密弱点过程中的必要性。接着，我们通过实例介绍了如何识别加密代码所属加密算法的类别。然后，我们介绍了在

在本系列的下一篇文章中，我们将通过一个加密强度不大的勒索软件的代码为例，逐行介绍创建一个解密程序的具体过程。

上一篇：WannaMine?警惕"永恒之蓝... 下一篇：利用DSPL对Google发动存储...

- 动动手指，沙发就是你的了！

登录 后跟帖

先知社区

现在登录

热门节点

技术文章

社区小黑板

目录

RSS 关于社区 友情链接 社区小黑板