第 12 章 静态分析

执行或者不执行恶意软件的情况下，都可以对恶意软件进行分析。静态分析技术即是在不执行样本的情况下对恶意软件进行分析，而执行样本查看行为的分析方法被称为动态分析。看起来针对样本的静态分析似乎与其他分析阶段是独立的，但其实并非如此，分析恶意软件时往往需要在静态分析与动态分析之间来回切换。本章中将会介绍静态分析的基本方法，与分析时常用的工具与技巧。

在前序章节中，已经涉及到了部分静态分析技术并且进行了实际练习。本章中仍然也会提及之前已经介绍过的各种技术。在阅读本章内容时，建议读者与前序章节中涉及静态分析技术的部分进行对照阅读，以此巩固对这些知识的掌握。必须记住，只有实际练习的更多，才能成为一个优秀的恶意软件分析人员。

为什么要进行静态分析?

静态分析通常在分析的起始阶段。在不运行样本的情况下，通过静态分析确定样本是良性的还是恶意的。通过静态分析，有时候甚至可以在无需进行动态分析的情况下确定恶意软件的类型、所属家族与攻击意图。

静态分析完成后，通常下一步就是动态分析。分析人员首先需要通过静态分析对样本文件的各种静态属性、环境要求进行分析，之后才能开始动态分析。如图 12- 1 所示。

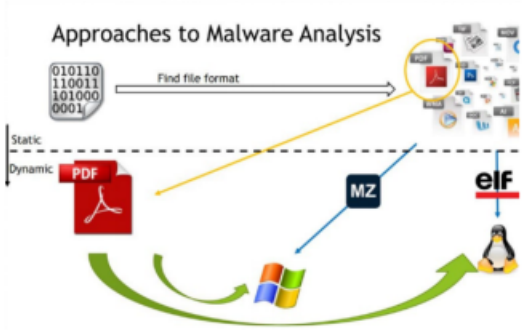

图 12- 1 通过静态分析帮助确定动态分析的环境要求

用于信息交换的样本哈希

无论是要进行动态分析还是静态分析，首先都需要检查其他人是否对该样本进行过分析并给出了结论。通常来说，很多情况下其他人都分析过相同的样本文件或者是属于同一位软件家族的样本文件，并且对外发布了分析报告。很多时候，相同的样本文件可能已经被上传到VirusTotal等恶意软件分析平台。

如果样本来自工作环境，通常是禁止将样本上传到公开平台或者与他人进行共享的，因为这些样本很可能包含敏感信息。特别是在恶意代码作为敏感文件的一部分嵌入，或者相关样本文件并不是恶意软件而是用户的良性文件时，尤其要注意敏感信息的数据安全问题。

为了解决该问题，分析人员通常会使用文件哈希代表样本文件来进行数据交换。几乎所有位置，包括互联网上公开的恶意软件分析平台、分析报告与博客在内，都会使用哈希值来标识样本文件。这样就可以在不需要上传样本文件本身的情况下，实现对恶意软件相关信息的共享。

哈希计算

在对样本进行分析时，通常都需要获取其哈希值。常用的哈希有 MD5、SHA1 与 SHA256，通常都要为样本文件生成这三个哈希。正如第 3 章中介绍的，可以使用多种工具来获取文件的哈希值。例如使用 HashMyFiles 针对样本文件 Sample-12-1 生成三个哈希，如图 12- 2 所示。

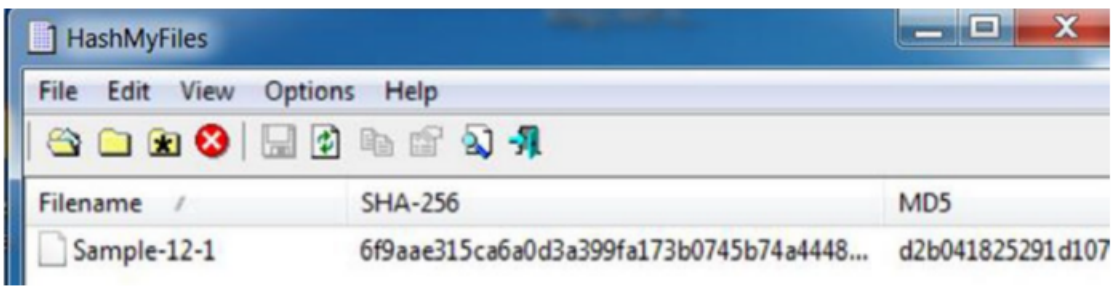

图 12- 2 使用 HashMyFiles 为样本文件 Sample-12-1 生成三个哈希

样本文件 Sample-12-1 的三个哈希如下所示：

SHA256 为 6f9aae315ca6a0d3a399fa173b0745b74a444836b5efece5c8590589e228dbca；SHA1 为 5bea9f59d5f2bdc67ec886a4025cdcc59a2d9c3；MD5 为 d2b041825291d1075242bd4f76c4c526。

互联网、博客与分析报告

从事恶意软件分析领域的分析人员经常会对外分享新发现的恶意软件的相关信息，以及其他各种与网络安全相关的内容。这些内容大部分会通过各安全公司发布的研究报告与博客、个人博客、年度安全报告等形式进入互联网。许多安全研究人员也在公共论坛、私密论坛和邮件列表中十分活跃，大家可以互相交流样本文件、样本信息与其他安全相关信息。

以上这些信息的来源，再加上 Google 等搜索引擎的助力，在遇到新样本时就可以通过这些途径来查看是否存在样本相关信息。

以样本文件 Sample-12-1 为了，在 Google 上搜索其 SHA256 哈希值如图 12- 3 所示。Google 返回了各种有关该样本的分析报告的链接地址，其中都包含该 SHA256 哈希值。如图中所示，文章显示该样本文件与 Petya 勒索软件有关。

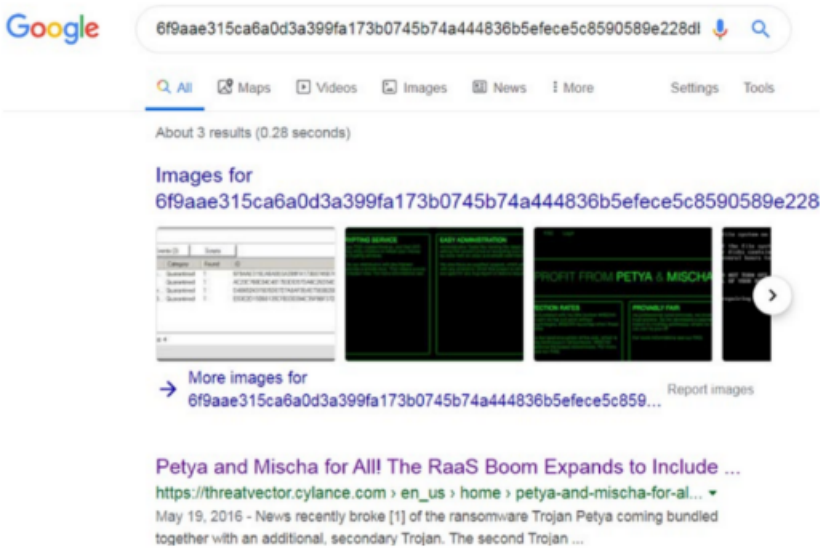

图 12- 3 使用 SHA256 哈希值在互联网上检索样本文件 Sample-12-1 的分析报告

读者可以自行尝试检索，查看 Google 与其他搜索引擎的返回结果。值得注意的是，如果在 Google 中检索该样本的 MD5 与 SHA1 哈希值则没有返回结果，只能检索样本文件的 SHA256 哈希值。这是因为文章中只提到了样本文件的 SHA256 哈希，而没有提及 MD5 与 SHA1 哈希值。所以读者在检索样本相关信息时，一定要尝试检索三种哈希 (MD5、SHA1 与 SHA256)。

VirusTotal 与其他分析平台

VirusTotal (www.virustotal.com) 是一个汇聚了众多反恶意软件引擎的在线分析平台。用户可以上传恶意软件样本文件，平台使用各种反恶意软件引擎进行扫描并生成分析报告。分析报告中会包含反恶意软件引擎的检出结果，如果检测引擎认为样本为恶意软件会给出所属类型、类别与家族。当然，也可以使用文件哈希值来进行直接检索，如果 VirusTotal 已经分析过该样本，则会直接给出分析报告。

从恶意软件分析人员的角度来看，在 VirusTotal 与其他分析平台上进行检索是非常必要的。将这些分析平台作为检测来源，可以使用样本文件的哈希值进行查询。如图 12- 4 所示，可以使用样本文件 Sample-12-1 的 SHA256 哈希值在 VirusTotal 上进行查询。

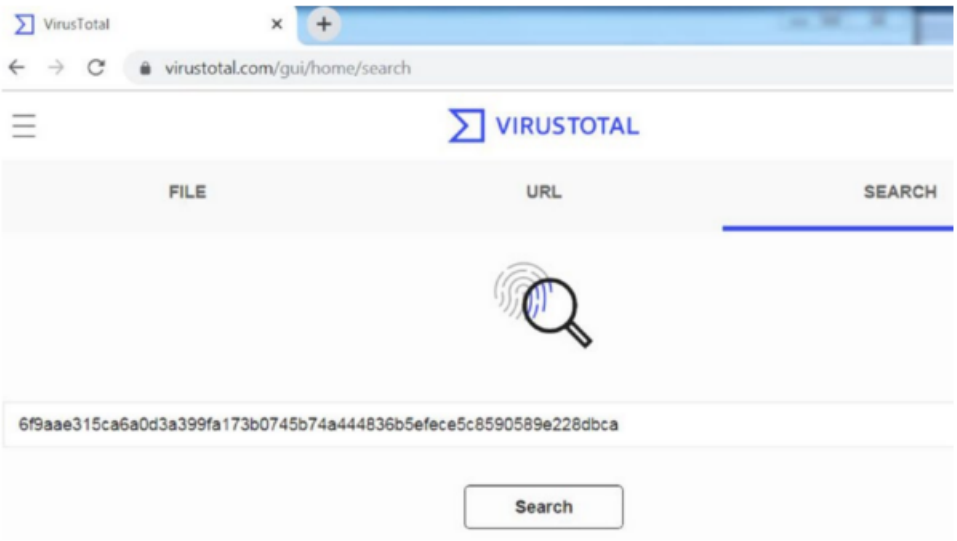

图 12- 4 在 VirusTotal 上查询样本文件 Sample-12-1 的哈希值

该样本的分析报告如图　12- 5 所示，在 VirusTotal 使用的 70 个反恶意软件引擎中共有 58 个引擎将文件检出为恶意软件。

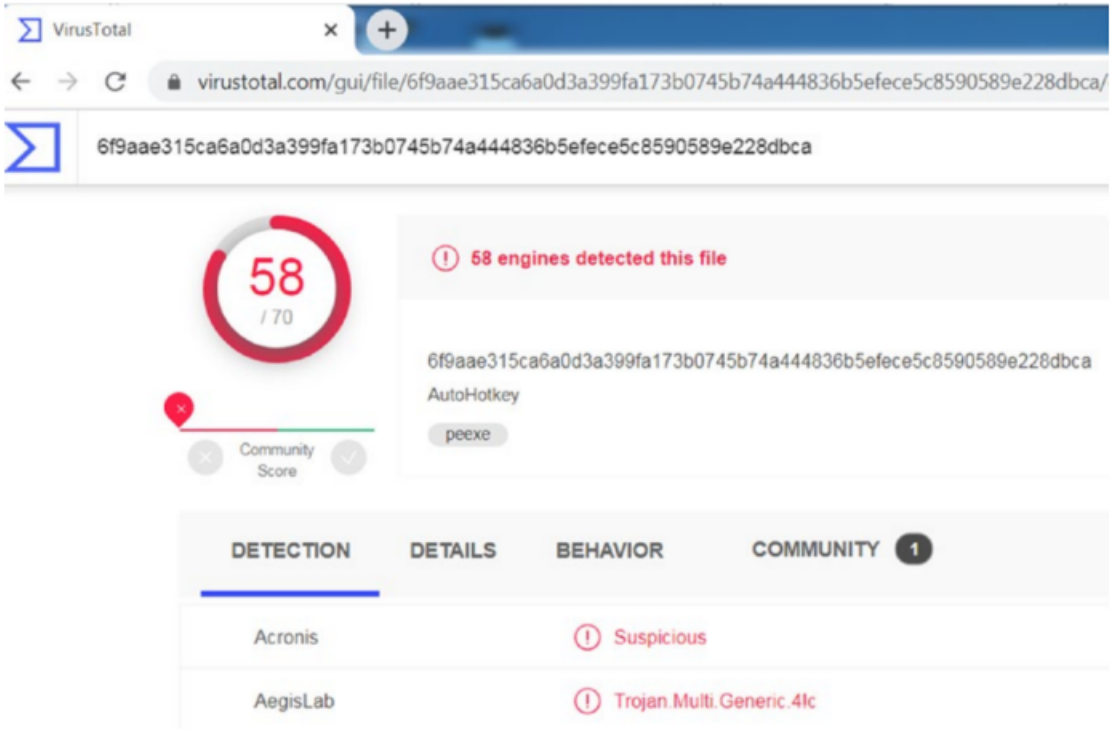

图　12- 5 样本文件 Sample-12-1 的分析结果

分析报告中通过多个选项卡（DETECTION、DETAILS、BEHAVIOR 与 COMMUNITY）提供了与查询样本相关的多类信息。DETECTION 选项卡展示各个反恶意软件引擎对样本的检出情况、DETAILS 选项卡展示从样本中提取的各种静态属性、BEHAVIOR 选项卡展示执行样本时观察到的各种动态行为，这些信息可以帮助分析人员快速了解该样本的完整情况。通过哈希值在 VirusTotal 等各种其他在线恶意软件分析平台中进行检索，可以获得样本的分析报告。通过图　12- 5 可以查看 VirusTotal 的分析报告，读者可以通过分析报告获取有关样本文件的信息。典型的在线恶意软件分析平台如下所示：

- VirusTotal
- Hybrid Analysis
- SNDBOX
- any.run

确定是良性文件吗?

在使用在线恶意软件分析平台时,经常会遇到平台认为是良性的样本或者说没有检测引擎检出的样本文件，这是否能表明该样本文件是良性文件？这些文件真的是良性文件吗?

这个问题很难回答，因为一个文件是否为良性文件需要通过多方面因素进行判断。安全厂商每天都会看到数百万个样本文件，其中有良性文件也有恶意样本文件。如此数量的文件决定了，想要通过哈希值来进行检测是不可行的，这也是推动基于行为的恶意软件检测技术发展的主要原因。与此同时，随着各种新兴复杂恶意软件的不断涌现，反恶意软件产品现有的签名与检测机制可能无法将样本检出。这也是这些在线恶意软件分析平台无法将样本文件检出为恶意软件的原因。

为了应对这种困境，每当发现新的恶意软件并且反恶意软件产品无法将其检出时，检测团队与工程团队必须新增/更新签名。在某些情况下，团队会在产品中增加新签名和新功能实现对该类恶意软件的检出。这些新签名与新功能都会随着产品的软件升级，更新到每个安装终端。通常来说，检测团队可能需要数天时间来发布更新。随着更新的下发与部署，检测产品再次遇到来自同一位软件家族的相同或者相似的恶意软件，就能够成功检出。

值得注意的是，检测团队需要数天时间才能给出新的签名与功能更新。如果恶意软件的样本文件在 VirusTotal 等在线恶意软件分析平台中被判定为良性，可能需要在几天后重新进行检测，希望那时检测引擎已经收到了包含新签名与新功能的更新。另外，这些样本首次提交到这些在线分析平台的日期也十分重要。通常来说，可以在样本首次提交到平台后的一周到两周内重新进行检测，要给安全产品的检测团队足够的时间发布能够识别/检测这些样本的更新。如果在样本文件首次提交给 VirusTotal 两到三周后，所有的反恶意软件产品仍然都认为该文件是良性的，那么很可能确实是良性文件。

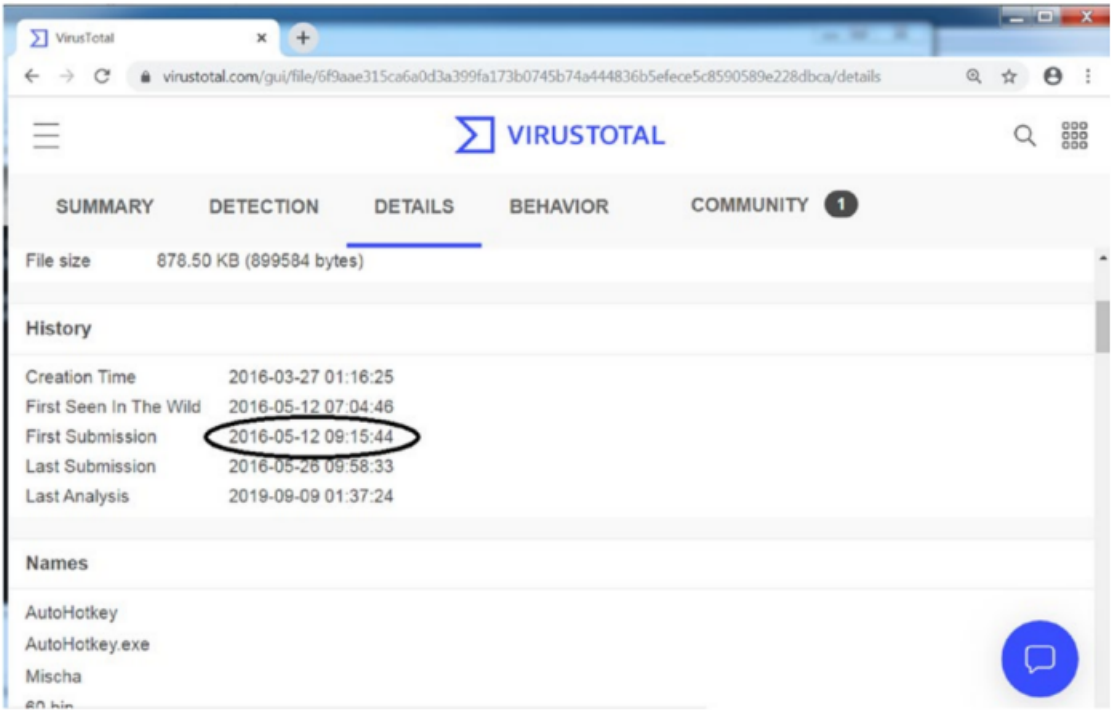

图 12- 6 表明样本文件首次提交到 VirusTotal 的时间字段

如图 12- 6 所示，VirusTotal 的 DETAILS 选项卡中的 First Submission 字段即为样本文件 Sample-12-1 的首次提交时间。

确定文件格式

恶意软件可能是各种文件格式的，例如 PE 可执行文件、.NET 可执行文件、Java 文件、脚本文件、JavaScript 文件、WMI 文件等。恶意软件也可能可以在各种操作系统与处理器架构上运行，例如 Linux、Windows、macOS 或者 Android 等操作系统；x86、x64、PowerPC、ARM 等处理器架构。

根据样本文件的文件类型与目标环境不同分析人员需要使用不同的工具或者分析方式来对样本文件进行分析。首先需要确定样本文件的文件格式，这与样本预期生效的目标环境息息相关。

如 **12-7** 所示，使用 trid.exe 查看样本文件 Sample-12-2 的文件格式。表明该样本文件是一个 PE 可执行文件，分析人员需要在 Windows 操作系统的环境下来对其进行分析。

使用 trid.exe 查看样本文件 Sample-12-4 的文件格式，根据结果 81.0%(.exe) generic CIL Executable (.NET, Mono, etc.) (73294/58/13)推断，该文件最可能为.NET 文件。在 Windows 操作系统上分析.NET 文件需要特定的.NET 框架与对应的分析工具，如果未安装.NET 框架或者安装的版本不匹配，都可能导致分析失败。在了解.NET 文件的前提下，可以在正确的分析环境下使用对应的.NET 框架与分析工具来对.NET 样本文件进行分析。

获取完整的失陷上下文

攻击者的入侵也存在完整的生命周期，首先可能会通过各种技术手段（包括鱼叉邮件、漏洞利用或其他方式）来投递恶意软件。在攻击利用成功后，攻击者可能会在网络中进行横向平移。作为分析人员，获取有关正在分析的恶意软件样本尽可能多的信息是非常重要的。特别是安全运营中心（SOC）的分析人员，或者是通过 SOC 获取样本文件进行分析的情况。以下为一些攻击与失陷的示例：

- 有攻击者将恶意软件作为附件通过鱼叉邮件发送到公司的财务部门、高层管理人员、人力资源部门等。
- 有攻击者将恶意软件作为附件通过普通垃圾邮件发送到公司的 IT 团队。
- 安全团队发现该恶意软件已通过网络传播到另一台机器中。
- 该来自垃圾邮件附件的恶意软件，其名为 Invoice.pdf.exe。

第一点可能表明这是一次有针对性的攻击尝试，通过分析恶意软件可以帮助确定攻击目标，判断是否为针对性网络攻击。如果财务部门是攻击的目标，恶意软件很可能为金融领域相关的恶意软件，例如银行木马。这可以为分析人员后续的分析工作提供指导与线索，收集更多信息来确认或证伪对应的判断。

第三点可能表明该恶意软件具有蠕虫传播或者横向平移的能力，其中可能嵌入了能够进行横

向网络扫描的工具。了解这些信息后，分析人员可以将精力集中在恶意软件中那些与网络扫描或调用网络 API 有关的部分。

第四点可能表明该恶意软件正在使用文件名与扩展名伪造的攻击方式，结合它作为垃圾邮件附件的背景，该文件很有可能是恶意的。

收集与正在分析的恶意软件有关的、尽可能多的上下文信息，可以帮助分析人员更好地作出判断。

文件名与扩展名伪造

本书的第 11 章将会更加详细地探讨文件名与扩展名伪造技术，攻击者通过这种方式利用能够引起受害者关注的文件名来引诱用户点击恶意文件，从而触发攻击与感染。部分常见的命名如 Invoice.exe、Invoice.pdf.exe、January_salary.exe、Resume.exe 等。

文件名伪造主要用于恶意软件投递阶段，如将恶意软件作为垃圾邮件和鱼叉邮件的附件，提高引诱受害者下载并点击的概率。电子邮件与附件可能会根据攻击目标的情况使用各种语言，而不局限于英语。在分析时，分析人员可以将这些外文翻译成母语来便于理解。如图 12- 7 所示，为意大利语编写的恶意电子邮件，其携带一个名为 Fatture_582_2018.xls（fatture 意为发票）的恶意附件。

图 12- 7 携带恶意附件的恶意电子邮件，文件名引诱用户点击

类似的，扩展名伪造通过构造特定文件名诱骗用户将其认为是特定扩展名文件。这种技术利用了大多数用户的无知，他们将.pdf、.xlsx 与.doc 等扩展名认为是不可执行文件的扩展名，因此将其认为是安全的。攻击者将这些扩展名添加到恶意样本的文件名中，攻诱骗用户将文件认为不是.exe 文件，从而下载并点击执行这些恶意文件。例如 January_salary.pdf.exe 与 Invoice.doc.exe。

将恶意软件作为电子邮件附件进行投递，用户匆忙之下可能会忽略文件名中的.exe 扩展名，而下意识地觉得文件名为 January_salary.pdf 与 Invoice.doc。

更糟糕的是，Windows 操作系统默认隐藏了扩展名。恶意软件下载到本地也会将.exe 扩展名隐藏起来，在文件浏览器中查看这些文件就会显示文件名为 January_salary.pdf 与 Invoice.doc。

分析人员应注意以下事项：

- 在电子邮件或者其他投递机制中，获取完整的失陷上下文确定附件的实际文件名是非常重要的。

- 留意那些引人注目的文件名，尤其是电子邮件的附件文件，有助于发现那些使用文件名伪造的恶意软件。
- 如果文件名与电子邮件是外语的情况，可以将其翻译成母语。
- 在分析环境中需要禁用扩展名隐藏，在个人主机上也应该这样做，这样就可以直观地查看文件的实际扩展名。

文件缩略图伪造

本书在第 11 章的“缩略图伪造“章节对该技术进行了详细介绍，故而此处只进行简要回顾。攻击者使用其他良性应用程序的缩略图、图标作为恶意软件的缩略图、图标，从而诱使用户认为这些样本是良性应用程序并点击执行。

如图 12- 8 所示，可见为 Microsoft Office Word 与 Excel 的缩略图。

图 12- 8 使用 Microsoft Office Word 与 Excel 文件图标进行缩略图伪造

在第 11 章可以看到，.doc 与.xls 文件会使用这些缩略图（如 **11-9** 所示）。攻击者将恶意软件的缩略图更改为 Microsoft Office 文档文件或者其他良性文件的缩略图——Adobe、VLC 视频文件等（如 **3-11** 所示）。

读者可以自行为样本文件 Sample-12-2 的文件名增加.exe 扩展名，如图 12- 9 所示。能够发现该 PE 可执行文件使用了 Microsoft Word 的缩略图。

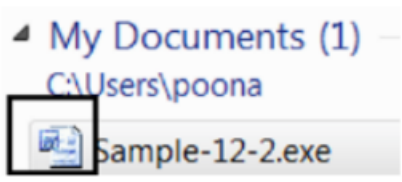

图 12- 9 样本文件 Sample-12-2 使用 Microsoft Word 的缩略图进行伪装

读者可以在 CFF Explorer 中打开样本文件，检查资源段查看文件缩略图。如图 12- 10 所示，样本文件 Sample-12-2.exe 的缩略图是 Microsoft Word 文档文件的缩略图。

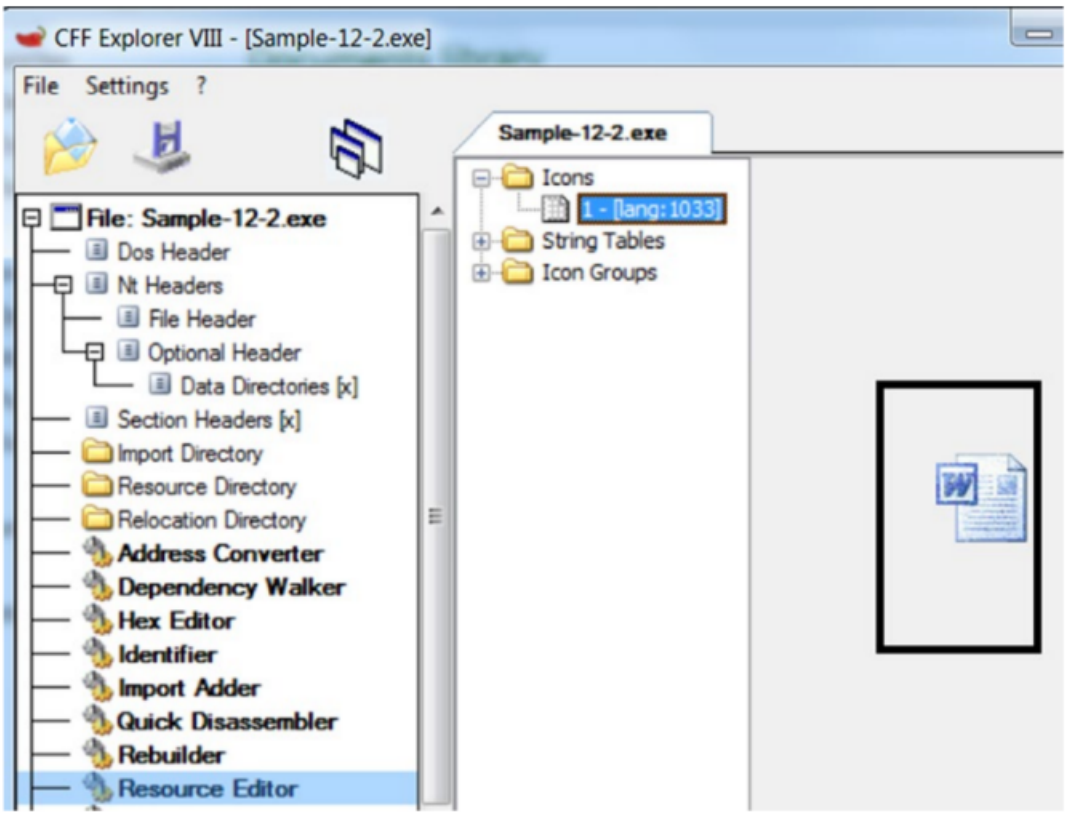

图 12- 10 样本文件 Sample-12-2 使用的 Microsoft Word 缩略图

使用前文所述方法确定文件的实际格式，如图 12- 11 所示，该文件为 PE 可执行文件。一个显示为 Microsoft Word 缩略图的 PE 可执行文件很可能是恶意的，最好对其进行进一步调查。

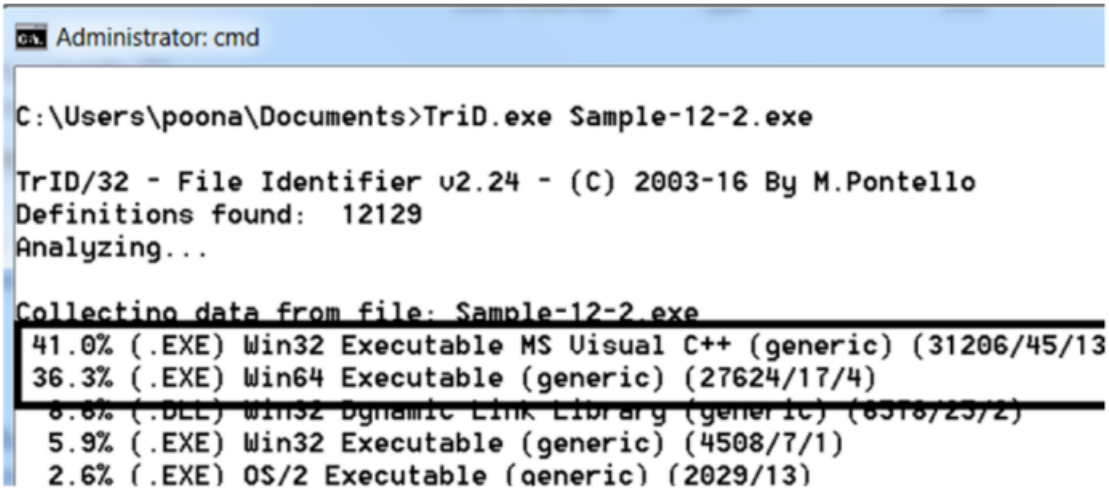

图 12- 11 样本文件 Sample-12-2 的文件格式为 PE 文件

许多恶意软件都会使用自定义缩略图，也有一些使用虚假缩略图。作为分析人员，必须要注意那些缩略图与文件格式类型不对应的文件，这种不匹配即是指向恶意文件的线索。

文件类型与文件扩展名不匹配

仍然以样本文件 Sample-12-2 为例，为其添加.txt 或.dat 扩展名，如将文件重命名为 Sample-12-2.dat。修改扩展名后能表明该文件是文本文件或者数据文件吗？当然不行，如图 12- 11 所示，该文件仍然是一个 PE 可执行文件。

在分析恶意软件时，尤其是在进行动态分析时，样本文件可能会创建、释放其他 Payload 恶意软件。这些文件可能是具有不正确文件扩展名的可执行文件或者配置文件，诱使用户将其认为是其他文件类型。

无论恶意软件样本的文件扩展名是什么，作为分析人员都要对文件的文件格式进行确认，包括恶意软件在动态分析时创建、释放的新文件。文件扩展名与实际文件格式之间的不匹配是非常可疑的，通常都需要进行进一步调查。

版本信息/详细信息

系统上大多数良性文件在“属性“窗口下有一个”详细信息“选项卡，可以通过右键单击文件并选择”属性“来查看。选项卡中将会显示有关文件的各种详细信息，包括文件版本、产品名称、产品版本与版权信息等。

读者可以自行查看 notepad.exe 的属性信息，如图 12- 12 所示，能够看到该应用程序的各种属性字段。同样的，也可以查看样本文件 Sample-12-2 的属性信息。图中可以明显看出，良性软件的这些属性信息在恶意软件中基本丢失。

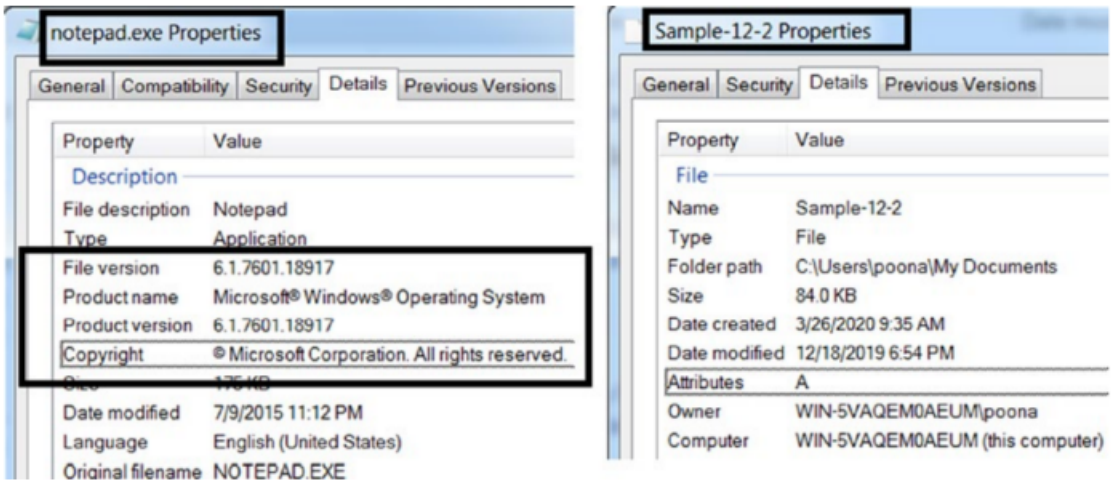

图 12- 12 文件提供的属性信息

在对样本文件进行分析时：

- 如果没有应用程序的属性字段，可认为样本是可疑的，需要进一步调查。
- 如果应用程序的属性字段看起来毫无意义，可以认为样本是恶意的，毕竟良性应用程序不会用毫无意义的字符串来描述应用程序的属性。

数字签名

在上一节中提到了使用应用程序的属性信息来发现可疑的恶意样本文件的方法。但是，如果攻击者将一个良性应用程序的属性信息完全复制到一个恶意软件中，该方式就失效了。为了解决该困境，并确定应用程序的开发者，数字签名应运而生。

读者可以在互联网上检索有关数字签名的更多信息。简单来说，被称为代码签名证书的数字密钥与传统签字署名文档类似，也可以对程序文件进行签名。代码签名证书是通过加密算法生成的，使用其为文件生成的唯一数字签名可追溯到文件的原始开发者。

如果你是一个软件开发者，就可以向颁发证书的机构申请代码签名证书，这些机构在审核申请者的身份后就会颁发对应的证书。软件开发者获得证书后，就可以使用该证书对应用程序进行签名，签名会随着应用程序一同分发以标识来源。使用该应用程序的用户可以验证该数字签名，并追溯到对应的厂商与软件开发者，从而确定应用程序的来源。

大多数软件厂商都会对应用程序进行签名。例如 firefox.exe 与 chrome.exe 分别为 Firefox 与 Chrome 浏览器的应用程序文件，可以右键查看程序的数字签名，如图 12- 13 所示。

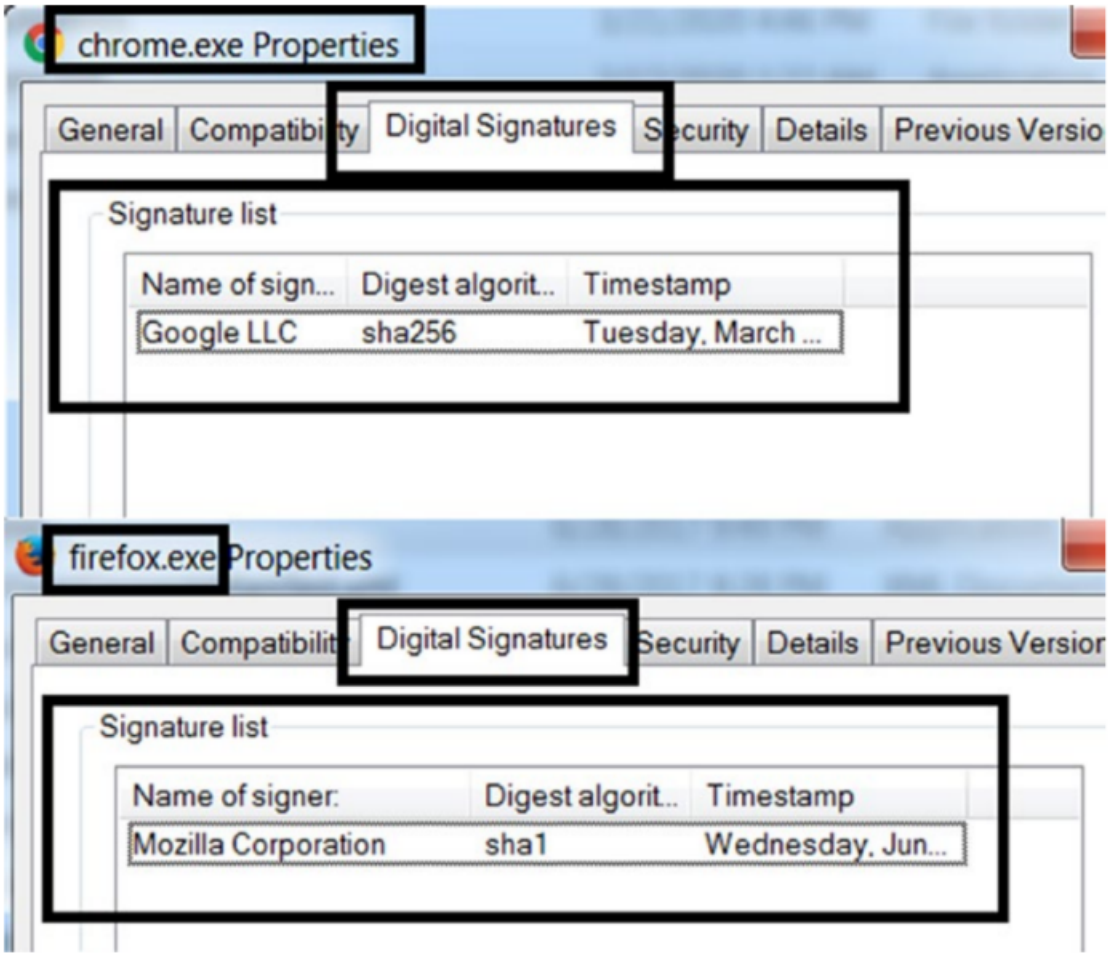

图 12- 13 firefox.exe 与 chrome.exe 的数字签名

如果分析的样本文件带有知名厂商签发的数字签名，则该样本为恶意样本的可能性会相应降低。对于恶意软件来说，大多数都不具备数字签名。如果文件不带有数字签名，可以将样本列入可疑列表等待进一步分析。

众所周知，部分攻击者会利用各种自持公司来购买数字证书，甚至是入侵其他公司窃取证书。攻击者会使用这些证书来签署恶意软件，期望这些恶意软件看起来更可靠且不会引起他人注意。

作为恶意软件分析人员，必须要记住的是，并不能认为带有数字签名的程序就是良性的。恶意软件攻击者可以购买证书来签署自己的恶意软件，带有数字签名只能表明应用程序的提供

方是已知的。

分析人员可以根据签署恶意样本文件的签发者、开发者、公司名称，来构建恶意软件签名数据库。当发现一个新的、带有数字签名在恶意样本时，可提取签发者名称（如图 12- 13 所示）增加到恶意软件签名的数据库中。后续发现任何能够与恶意软件签名数据库匹配命中的新样本，就可以将其标记为可疑样本进行深入分析。

静态字符串分析

恶意软件也是软件，程序文件中会包含许多字符串，这些字符串通常能够标识恶意软件的类型、功能与意图。恶意软件中的字符串也是非常有用的指标，可以用于识别恶意软件，还可以通过其来了解恶意软件的组件、功能、意图与分类。

正如第 7 章中所述，大多数恶意软件都是加壳的。大多数情况下，当恶意软件被加壳时，原始样本文件中的数据与字符串会被混淆并且不再可见。在某些情况下，有些数据与字符串能够躲开加壳，在最终加壳后的样本文件中仍然可见。甚至在有些时候，攻击者不会对恶意软件进行加壳，或者有其他分析人员脱壳并提取了原始的恶意样本文件。这时就可以对未加壳的恶意软件进行分析，也可以直接查看原始的字符串与恶意软件内部结构。

可以使用第 2 章中介绍过的 BinText 分析工具来查看样本文件中的字符串，第 7 章中也使用该工具进行了对应的分析练习。

现在，使用 BinText 查看样本文件 Sample-12-3，并查看是否存在任何可疑字符串。如图 12- 14 与图 12- 15 所示，这些可疑的字符串表明该样本文件可能是恶意软件。

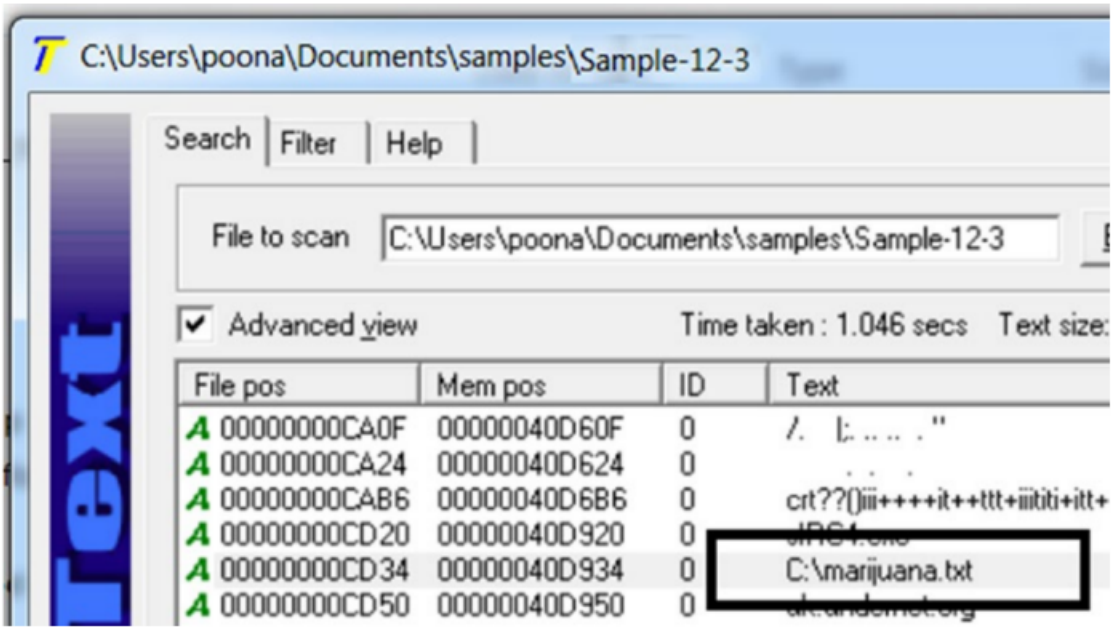

图 12- 14 BinText 工具查看样本文件 Sample-12-3 的可疑字符串

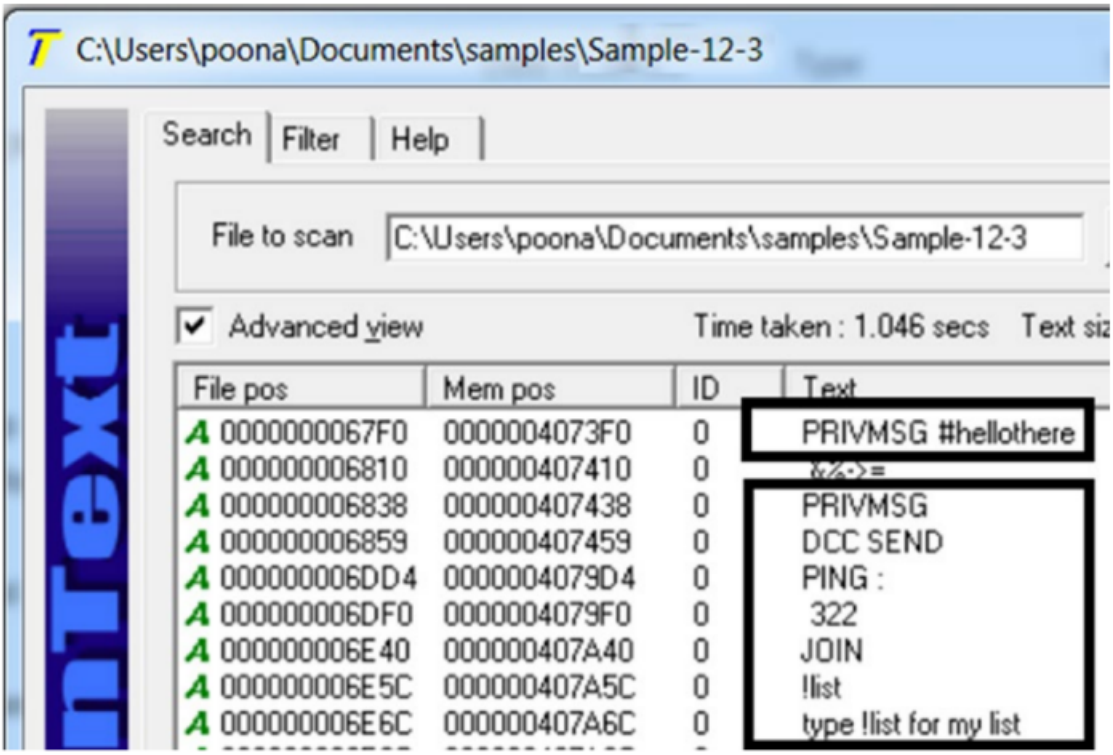

图 12- 15 BinText 工具查看样本文件中存在可能使用 IRC 进行 C&C 通信的可疑字符串

图 12- 15 中为与 IRC 协议相关的字符串，恶意软件可能使用 IRC 协议进行 C&C 通信。如何确定这些字符串是恶意的呢？后续将会详细进行介绍。总体来说，就是要寻找那些奇怪的字符串，那些通常在良性软件中不存在，而只会在恶意软件中发现的字符串。例如字符串 C:\ marijuana.txt 就是一个奇怪的字符串，在任何良性软件中可能都找不到该字符串。同

样的，图 12- 16 中的 IRC 字符串表示样本使用了 IRC 协议，这也是恶意软件常用的通信协议。这些线索都表明该样本非常可疑，可以进一步分析了解更多信息。

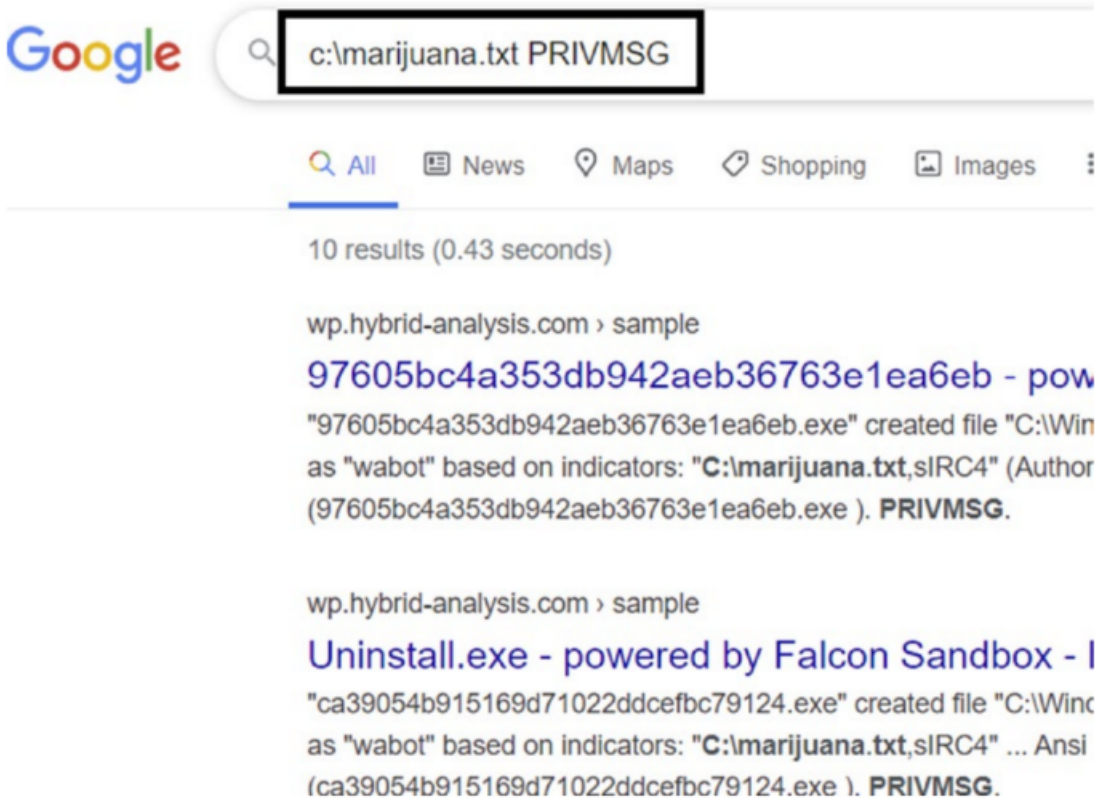

图 12- 16 在 Google 中检索样本文件 Sample-12-3 中的可疑字符串

可疑字符串

使用这些可疑的字符串，可以在互联网中检索其他分析人员发布的分析报告，寻找那些出现相同字符串的样本文件。其他分析人员可能并不拥有相同的样本文件（哈希值完全相同），但其他人可能分析过同一恶意软件家族的其他样本文件。正如图 12- 16 所示，检索这些可疑的字符串就能够找到能够确定样本文件为恶意软件的分析报告。

表示恶意的模式与字符串并不是一成不变的。分析人员可以不断收集那些能够标识样本为恶意软件的恶意字符串构建相关数据库，并且随着不断发现新样本而持续更新。以下是部分要点：

- 在遇到可疑字符串时，可在互联网上进行检索，查看是否在他人的分析与威胁报告中提

及这些字符串，来对其进行验证。

- 要注意奇怪的字符串，例如样本文件 Sample-12-3 中的字符串如图 12- 14 所示。另一个同类的例子是样本文件 Sample-12-4 中的 YUIPWDFILE0YUIPKDFILE0YUICRYPTED0YUI1.0。乍看之下该字符串像是垃圾字符串，但其中实际上包含 FILE0 与 CRYPTED1.0 之类有意义的单词。在搜索引擎中检索该字符串可以发现其直接指向恶意软件家族 Pony Loader 或者 Fareit。后续在第 13 章中也会对该字符串进行深入讨论。
- 要注意异常的字符串，尤其是那些不会在良性软件中出现的字符串。如图 12- 15 所示，样本文件 Sample-12-3 中出现了 IRC 协议的字符串。良性软件很少会使用 IRC 协议，这表示该样本是可疑的，可能需要进一步调查。
- 要注意带有域名的字符串，这可能表明攻击者使用域名进行 C&C 通信。
- 要注意带有反恶意软件和其他安全工具名称的字符串。众所周知，恶意软件会通过检查安全工具的存在来保护自身。例如反病毒引擎的名称、ProcMon、Process Hacker、Process Explorer、Wireshark、OllyDbg 等。
- 要注意带有 IP 地址的字符串，这可能是攻击者的 C&C 服务器或者用于与 C&C 服务器通信的中继服务器。
- 要注意带有大量文件扩展名的字符串，这可能表明该样本是勒索软件。通常勒索软件会检查系统上的所有文件并加密某些特定文件扩展名的文件。在第 15 章中将会更详细地探讨勒索软件的分类与识别。

在第 13 章与第 15 章中将会继续讨论基于字符串的分析方法样本间存在相同的模式不仅可以用于识别恶意软件，还可以对样本进行分类。

Yara

Yara 被称为恶意软件研究人员的瑞士军刀，其为一个针对文件的规则匹配引擎。使用 Yara 就可以使用人类可读的字符串甚至二进制串创建匹配规则，再使用布尔表达式组合这些模式来进行匹配。

使用样本文件 Sample-12-3 进行分析，其中包含字符串。如代码 12- 1 所示，这个简单的 Yara 规则能够检出所有包含该字符串的样本文件。打开一个名为 YARA-example.txt 的文本文件，并将代码 12- 1 中的内容都写入该文件，这就成功创建了一个 Yara 规则文件。

代码 12- 1 通过特定模式匹配文件的示例 Yara 规则

```
rule YARA_example
{
 meta:
 description = "This is just an example"
 strings:
 $a = "marijuana.txt"
 condition:
 $a
}
```

如图 12- 17 所示，使用规则针对样本文件进行匹配，可以发现匹配成功。也可以针对文件 C:\Windows\notepad.exe 使用相同的 Yara 规则进行匹配，匹配不成功说明该文件中不包含字符串 marijuana.txt。

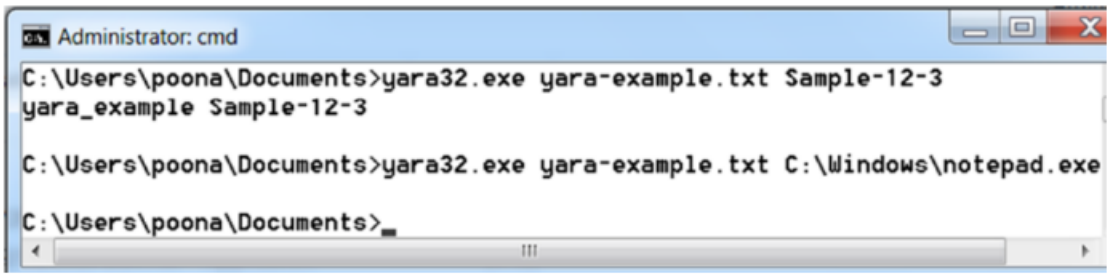

图 12- 17 使用代码 12-1 匹配样本文件 Sample-12-3

如代码 12- 2 所示，也可以使用多个模式并且结合布尔表达式来创建更加复杂的规则，针对样本文件 Sample-12-3 进行匹配。能够看到是可以匹配成功的，该样本中包含规则中规定的所有三种模式：marijuana.txt、PRIVMSG 与 hellothere。

代码 12- 2 多模式与布尔表达式结合的复杂 Yara 规则

```
rule YARA_example
{
 meta:
 description = "This is just an example"
 strings:
 $a = "marijuana.txt"
 $b = "PRIVMSG"
 $c = "hellothere"
 condition:
 $a and $b and $c
}
```

Yara 对恶意软件分析人员来说十分有用，可以帮助分析人员快速创建自定义规则并将其针对恶意软件样本文件进行匹配，以查看其能否匹配成功。

Yara 的典型应用场景是：根据时间的推移构建自定义 Yara 规则库，在日常分析时遇到新恶意软件就新增对应的规则。在遇到恶意软件样本需要进行分析时，可以针对该样本使用 Yara 规则库进行匹配，查看是否已有规则能够匹配成功，从而加快分析效率。

许多安全分析人员会在 GitHub 与社区中免费共享他们个人编写的 Yara 规则，在下载与使用其他人的 Yara 规则前请务必注意：部分规则会导致漏报，甚至误报。

完整介绍 Yara 的所有特性不在本书的讨论范围，强烈建议读者仔细阅读 Yara 的相关功能并尝试编写更多规则进行练习，这样才能快速掌握 Yara 规则编写技能。

Yara 的局限

尽管 Yara 是恶意软件分析人员的绝佳分析工具，但大多数分析人员都在滥用该工具。在第 7 章中提到过，大多数恶意软件都是加壳的，这意味着原始的字符串和数据已经被壳混淆了，直接查看样本会发现类似如 **7-11** 所示的垃圾字符串。

许多分析人员倾向于从加壳的恶意样本中提取这些混淆后的字符串，并利用这些字符串编写 Yara 规则。这其实收效甚微，而且利用这些混淆后的字符串编写的 Yara 规则可能与其他加

壳的良性文件也能够匹配，这会带来负面影响。

实际上，应该是用脱壳后的代码模式进行编写 Yara 规则。但在大多数恶意软件已经加壳的情况下，哪里找未加壳的恶意软件呢？这也就是动态分析技术发挥作用的地方，恶意软件在执行时需要自动脱壳并将恶意代码解压缩到内存中，这就可以针对正在运行的进程的内存进行 Yara 规则匹配。没错，Yara 规则也可以针对正在运行的进程进行匹配。在下一章中将更加详细地介绍该技术。

静态分析是动态分析的序章

静态分析通常是分析的开始，但很多时候只靠静态分析可能无法得出任何有用的结论。这时需要进入分析的下一阶段——动态分析，执行样本文件观察其动态行为。

在开始动态分析之前，仍然需要通过静态分析确定样本执行所需的操作系统等环境信息，以及需要使用的动态分析工具。本章前文也提到过，在样本分析样本时可能需要安装某些版本的.NET 框架。同样的，如果通过文件格式识别工具 trid.exe 确定恶意样本是一个 Java 程序。想要运行并分析 Java 程序，就需要分析环境中安装 Java 运行时引擎（JRE）。这些所依赖的环境信息，很大程度上需要依赖静态分析获得。因此，在开始动态分析样本前，通过静态分析尽可能多的收集与样本有关的信息非常重要。

总结

本章着重介绍了分析的初始阶段——静态分析。静态分析通常是动态分析的前序阶段，依赖静态分析的结果才能配置动态分析的环境。本章中也涉及到了大量静态分析工具技术，这些都可以帮助分析人员识别恶意样本，也可以帮助判断良性文件以避免浪费宝贵的分析时间。本章也为分析进入分析的下一阶段——动态分析，做出了铺垫。下一章中将对动态分析进

行详细介绍。