红队操作中的搭档——Cobalt Strike 让你的渗透测试更快更持久（下）

原创 丝绸之路 嘶吼专业版 昨天

接上文

但是，这种动态编译方法要求在 Cobalt Strike 客户端系统上安装 Mono，因为 Mono 编译器会用于构建程序集。目前，Linux 主机只支持动态编译操作系统，但是 Windows 和 macOS 将来可能会得到支持。每种方法都有它们的优点和缺点，操作人员可以自由选择他们喜欢的方法。

在将有效载荷编译到用于WMI文件移动的程序集时，需要考虑的是文件大小。目前，Cobalt Strike支持文件大小最大为 1MB 的执行程序集。可能有时候生成的程序集会超过这个大小限制，在此之前，Cobalt Strike 会悄无声息地失败，除了在团队服务器上之外，Cobalt Strike 不会给出任何关于为什么没有执行的信息。攻击者脚本内置了一个检查，如果已经达到程序集的大小限制，它将通知用户，并将错误(和潜在的修复提示)写到信标会话中。

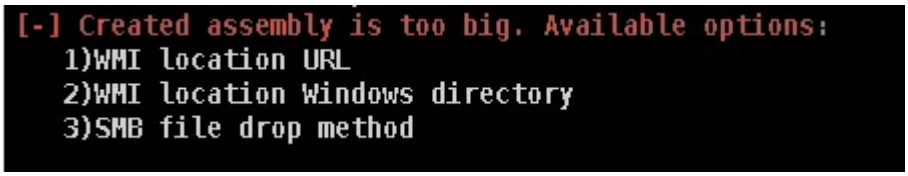

图8—程序集大小检查错误

如果未选择 WMI 文件写入方法，那么下一个可用的选项将是 SMB，它使用 Cobalt Strike 的文件上传功能，但正如前面提到的，它也内置到了程序集中(独立使用)。WMI to Registry 和 Custom 选项的WMI 类仅在文件移动中公开。这两种选项的主要用例是将数据(以 shellcode 为例)写入该位置，以及一个知道如何访问该数据的文件。这很可能需要与自定义预构建文件结合使用。无论如何，如果操作人员选择使用它，选项将保留在那里。

从该工具包创建的有效载荷都来自附带的模板，但这些都来自于其他公开项目。强烈建议更改模板以适应你的需要。在当前状态下，该工具的有效载荷生成方面还不是很成熟。在有效载荷生成组件中没有对 shellcode /源代码进行加密、混淆或密钥控制。这个问题将来可能会得到解决，但是理想情况下，使用外部有效负载生成工具集或CI管道会更好。此外，还有两个可用的自定义选项: 自定义预构建和自定义非预构建。这些选项的目的是让用户在使用有效

载荷时有一点更大的灵活性。如果操作人员创建了一个需要快速部署的自定义有效载荷，那么可以使用自定义预构建选项，并且可以对其应用所有相同的选项，例如动态地将其编译到程序集中，并通过 WMI 事件或 SMB 移动程序集。对于自定义选项，默认情况是执行一个二进制文件，命令为其完整路径和文件名。由于有时情况并非总是如此，自定义选项为用户提供了覆盖该选项并运行自定义命令的选项(如果执行时需要设置特定的参数，则最好使用该选项)。

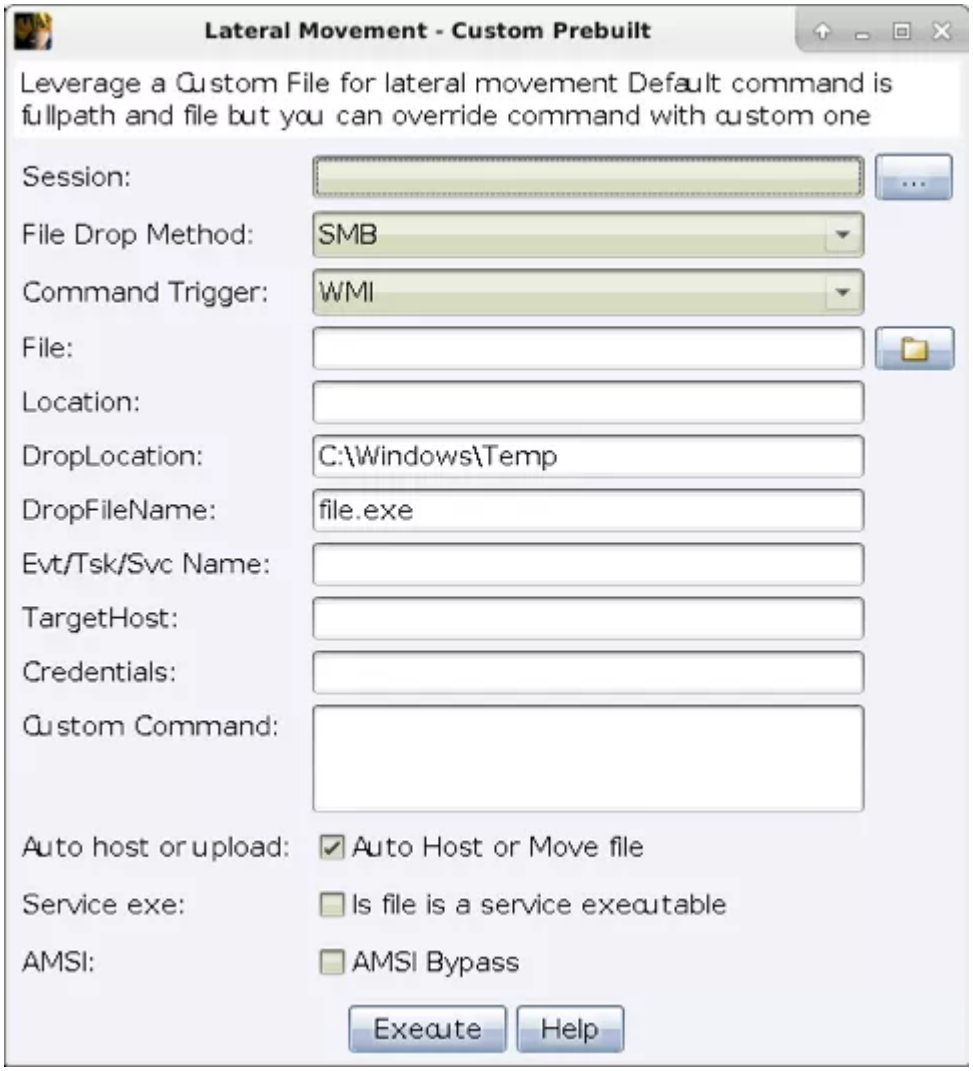

图9—自定义预构建对话框

可以理解的是，有时用户可能会忘记哪些是可选的，哪些不是，因此构建了一个帮助对话框，可以提供所有选项含义的更多细节(图9)。

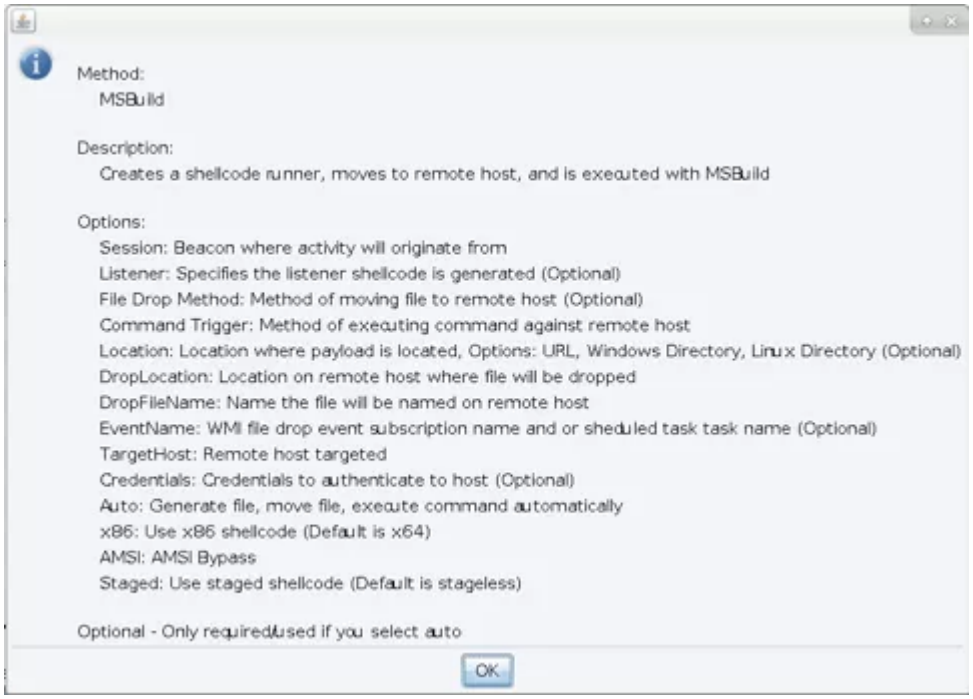

图10—帮助菜单

除了为所有已实现的操作提供对话框外，还为所有“文件”操作构建了信标命令。命令使用默认值，操作人员可以在“设置默认值”对话框中修改这些默认值。信标命令将读取这些缺省值以及一些用于执行横向移动的参数。每个任务参数包括图10)。对于自定义预构建的二进制文件，参数是 。一旦给出这个信息，它将像其他信标命令一样执行。

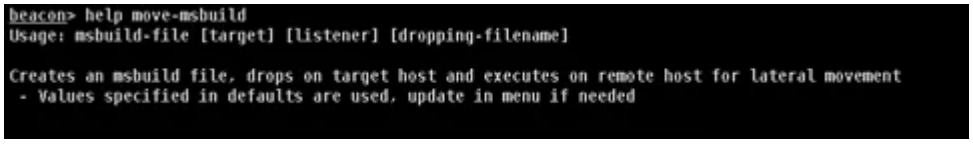

图11— 信标（Beacon）命令

这个工具包中还缺少一些项目，这些项目将在未来实现，可以使它更加强大。这样做的目的是为了在一定程度上可扩展，因为如果要在其中实现任何新技术，则应该只需要增加一些内容。所有当前的模板都可以替换(并且应该替换，因为它们主要是位置设置)，以满足操作人员的需要，并且可以将它们自己的模板放在原来的位置，但是文件名必须保持不变。除了命名之外，还需要替换模板。模板必须有一个find和replace字符串(默认为$$PAYLOAD$$，但可以更改)，以便找到以适当格式放置替换代码的位置。下面是 C# 中的一个例子。

```
string strSC = "$$PAYLOAD$$";byte[] sc = Convert.FromBase64String(strSC);
```

StayKit

Staykit 的操作与 MoveKit 非常相似，但只能在已经建立了 C2通信的主机上安装持久性，而不是通过远程任务。该套件与 SharpStay 一起包装在同一个 .Net 程序集，其中的一个攻击者脚本用于处理持久有效载荷的用户功能和模板。持久性并不是专门为 CobaltStrike 设计的，大多数公开可用的持久性脚本都有两种可靠性: Windows 二进制文件和 PowerShell。这并不是说它们不好，它们是否可以使用非常依赖于环境。StayKit为操作人员提供了现有产品的替代品。

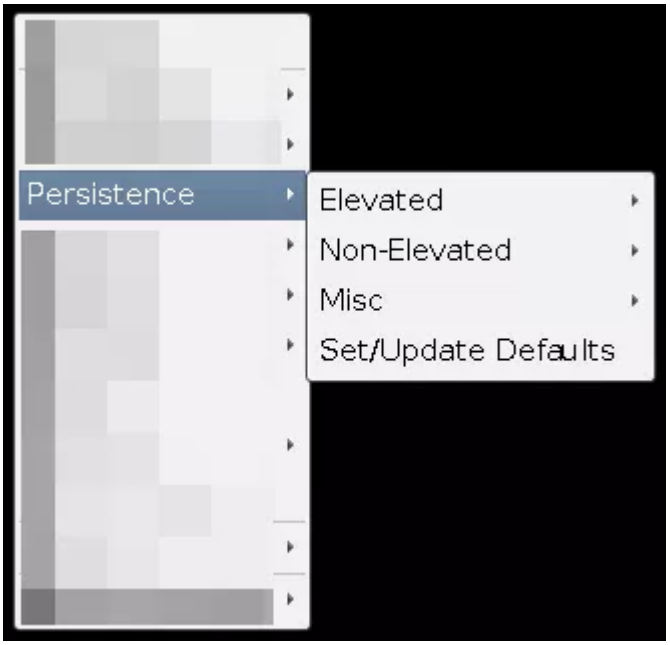

图12— StayKit 菜单

Staykit 的持久性选项分为三类，分别是“权限提升（Elevated）”、“非权限提升（Non-Elevated）”和“杂项（Misc）”。与 MoveKit 类似，StayKit 也有配置默认设置的选项，可以使持久性安装更快。可用的持久性选项有：

- Elevated （权限提升）
- 注册表项(多个)
- UserInit
- 计划任务

- 服务
- WMI 事件订阅
- Non-Elevated （非权限提升）
- UserInitMprLogonScript
- 计划任务 COM 处理程序劫持
- 交接文件夹
- 杂项
- 添加计划任务操作
- 替换二进制文件/其他文件
- 启动文件夹
- 创建新的快捷方式
- 后门快捷方式
- 获得运行中的服务
- 获取计划任务
- 获取计划任务 COM 处理程序

Staykit 中的技术都是已知的持久性方法，但在将来还会添加更多。由于每种持久化机制具有非常不同的选项，因此没有内置信标命令，只有对话框。此外，与 MoveKit 不同的是，它没有附带的模板，这取决于操作人员提供自己的模板或预先创建的文件。当考虑安装持久性时，需要记住一个工作流。如果给定了“自定义文件”或“模板”选项，那么文件只写入主机。如果选择‘Template’（模板），则使用指定的模板，并为选定的监听器生成 shellcode，然后将文件放到磁盘上。如果使用“自定义文件“，那么该文件将上传到主机。当不再需要持久性时，每种技术都有一个清理选项。至少有几个选项需要填写，以便知道到底需要清理或删除什么。

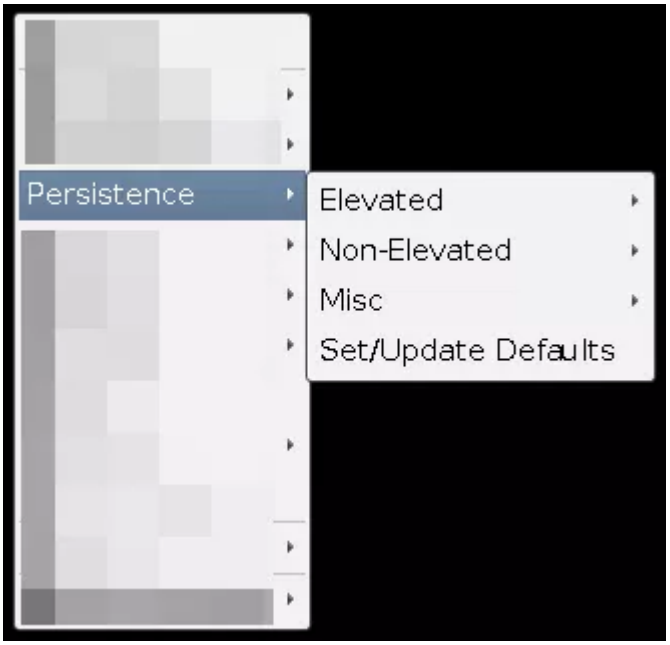

图12—注册表项的持久性选项

使用每种技术的所有可用选项，很容易在安装或删除持久性时混淆所需的选项。每个可用选项都添加了相应的帮助菜单，以帮助确保操作人员知道每个选项的含义(图13)。

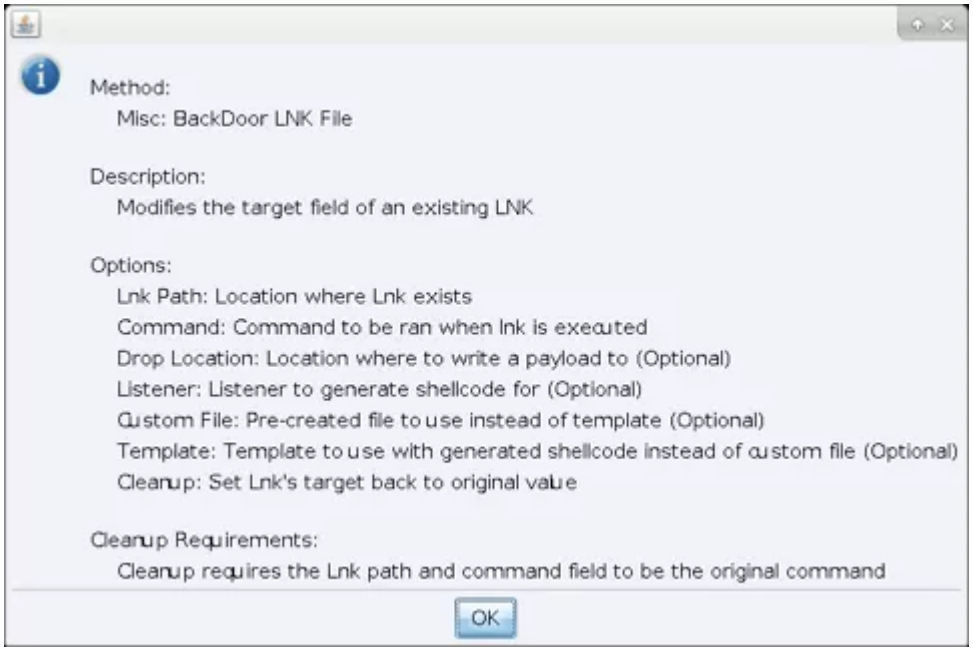

图13—后门快捷方式文件持久性的帮助菜单

赞誉

通过模板，MoveKit 提供了快速、高效地修改和定制横向移动选项的能力。所有包含在 MoveKit 中的模板都是根据业内其他几位研究人员的研究成果创建的。凯西•史密斯(Casey Smith)和奥德瓦尔•莫(Oddvar Moe)等研究人员进行了大量分析，他们发现了通过利用对二进制文件、脚本和dll的错误信任来获得命令执行的技术，这些技术也被称为“活在大地上”

(Living off the Land)。奥德瓦尔 · 莫创建了一个名为 LOLBAS 的项目，该项目记录了许多业内人士发现的一些技术。这些都是 MoveKit 执行选项的一部分。

就像模板一样，在工具包中还有很多我没有提到或创建的想法、代码和示例。保为这些项目付出辛勤劳动的人得到他们应得的荣誉是很重要的。这些想法和代码归功于：

MoveKit：

• WMI —想法来自于 harmj0y 的 SharpWMI 项目的代码

• SCM —想法来自于 Tim Malcomvetter 的 CSExec 项目

• DCOM —代码来自于 rvrsh3ll 的 SharpCOM 项目和 cobbr 的 SharpSploit 项目

• Excel 4 DCOM ——代码来自于rvrsh3ll 的 Excel4-DCOM 项目和 Outflank 的 Excel4-DCOM 项目

• 修改服务的 binpath ——想法来自于 Mr-Un1k0d3r的 SCShell 项目

• 服务 DLL 劫持 ——代码和想法来源于 djhohnstein 的研究成果和 SharpSC 项目

• C# 模板 —— 代码来自于 subTee 的研究成果

• 脚本模板 —— 代码来自于 vysecurity 的 CACTUSTORCH 项目

StayKit：

• WMI Event subscription ——参考了 Empire项目(实际的 WQL 查询)和 Matt Graeber 的原创研究

• 计划任务 COM 处理程序劫持 ——思路来自于 enigma0x3 的研究成果

• Junction 文件夹 ——代码和思路来自于 matterpreter 的 OffensiveCSharp 项目

• 后面快捷方式（Backdoor LNK）—— 参考了 harmj0y 的 LNKBackdoor 脚本

写在最后

总的来说，我认为这展示了 Cobalt Strike 的一些惊人之处，它让用户能够做出他们认为必要的任何改变。如前所述，这两个工具包的的攻击者脚本方面为操作人员生成有效载荷或移动文件提供了一部分协助。然而，底层的功能都是解耦的，不需要Cobalt Strike。但这些项目绝不是解决横向移动或者是在所有环境中解决持久性问题的银弹，而是提供了进一步的定制选项和另一种选择。这两个工具包都将得到支持，并且随着时间的推移，它们内置了新的功能，我收到了更多的反馈。此外，还有一个待办事项列表，我想添加的功能也将很快实现。读者提出的可行的想法将被考虑添加到工具包中。

关于横向移动、持久性和使用的有效载荷的检测方法都有很好的记录。随着 C# 和.Net 恶意软件的广泛流行，以及随着检测能力的不可避免的提高，C#在攻击方面的使用也在不断加强。在 .Net 4.8版本中，我们已经看到了在公共语言运行时为Windows使用程序集和事件跟踪时用到的反恶意软件扫描接口(AMSI)的集成。利用这些能力和强有力的方法，将为 .Net 攻击工具的未来检测铺平道路。关于两个工具包中，我们讨论过的选项的检测能力的想法我将在稍后的文章中发表。

从众多被引用的作品中可以看出，这些项目依赖于业内其他人的大量努力。除了一些想法和方法之外，我主要将其他工具的不同技术和想法整合到一个工具包中，以便快速、自定义地使用。

在处理横向移动和持久性时，操作人员在对主机执行操作之前必须考虑到所有操作的影响，这一点非常重要。有许多机会可以扩展每个项目的功能以满足操作人员的需要。

免责声明:这些工具包并不是由 Cobalt Strike 或其创始人Raphael Mudge 正式授权/认可的。

本文参考自：https://posts.specterops.io/move-faster-stay-longer-6b4efab9c644

END

阅读原文