【译】Metasploit：Metasploit module 的正确打开方式

王一航 / 2018-06-13 13:41:09 / 浏览数 5196 新手 入门资料 顶(0) 踩(0)

---

---

- 原文地址：https://github.com/rapid7/metasploit-framework/wiki/How-to-use-a-Metasploit-module-appropriately
- 作者：Metasploit Community
- 译者：王一航 2018-06-13
- 校对：王一航 2018-06-13

---

Metasploit 将一些难以理解或者难以设计出来的技术转换成了非常容易使用的东西。不夸张的说，只需要点击几下即可让你看起来像《黑客帝国》中的 Neo)
一样狂拽酷炫吊炸天。

这样 Hacking 变得非常简单。然而，如果你以前没有接触过 Metasploit，那么你需要知道的是：没人能一次成功（译者注：原文为 Nobody makes their first jump，来自《黑客帝国》）。你会犯错误，有时很小，有时可能会是灾难性的.....但愿不会。

在第一次进行利用的时候你很可能会面对失败，就像《黑客帝国》中的 Neo
一样。当然，要想达到你的目标，你必须学会适当地使用这些模块，我们会教你如何做。

在本篇文档中，我们不要求读者有 Exploit 开发的知识。当然如果有一些 Exploit 开发的经验，那当然再好不过了。
总的来说，在漏洞利用之前，实际上有一些"功课"需要做。

## 加载一个 Metasploit 模块

每一个 Metasploit 模块都会携带一些元数据来解释这个模块的用途，首先你需要加载目标模块。例如：

```
msf > use exploit/windows/smb/ms08_067_netapi
```

## 阅读一个模块的描述和参考文献

可能听起来让人惊讶，但是有时候我们仍然会被问道一些已经在文档中解释过的问题。所以，在你决定究竟一个模块是不是应该被使用的时候，你应该先在模块提供的文档和

> 哪个具体产品和哪个具体版本是存在漏洞的：关于一个具体漏洞，这是你应该明确的最基础的问题
>
> 漏洞的类型是什么？漏洞是如何被利用的？基本上来说，你正在学习的是漏洞利用的副作用。例如：如果你正在尝试利用一个内存破坏（译者注：原文 memory corruption）漏洞，如果因为某些原因，利用失败了，你可能会让目标服务器直接崩溃掉。就算利用成功，得到了一个 Shell，当你已经完成了你想要在目标系统上做的事情，并在 Shell 中输入 exit 退出的时候，目标服务器也会崩溃掉。高层（译者注：原文 High level，个人理解相对于底层漏洞，一个不小心系统可能会直接 crash，但是如果高层的漏洞利用，如果只是一个 WEB 漏洞，那么肯定对操作系统造成不了什么影响）的漏洞一般来说比较安全，但是也不是百分之百安全。例如：某一个漏洞利用程序可能需要修改配置文件或安装可能导致
>
> 哪些目标环境是通过漏洞利用测试的？
> 当一个漏洞利用程序开发完成的时候，如果目标操作系统种类太多，通常情况下很难去在每一个可能的目标系统上进行测试。通常开发者仅会在他们手头现有的目标系统
>
> 服务器必须满足什么条件才可以被利用？通常，一个漏洞利用需要很多条件同时满足才可以成功。有些情况下，你可以使用 Metasploit 的 check 命令 来检查目标系统是否可以被利用，因为 Metasploit 将某一个事物标记为存在漏洞的时候，事实上是利用了目标的 BUG。对于使用 BrowserExploitServer mixin 的浏览器攻击程序，会在加载漏洞利用之前检查可利用的条件是否满足。但是自动化的检查并不是每个模块都有的，所以你应该在执行 exploit 命令之前先尝试搜集这些信息，这应该是一个攻击者的常识。例如：一个 WEB 应用的文件上传功能就很容易被滥用，被用来上传一个基于 WEB 的后门程序，但是这个漏洞要成功利用的条件是用户必须有对上传文件夹的访问权限。如果你的目标系统不满足这些必要条件，那就没有必要再进行尝试了。

你可以通过输入如下命令来查看模块的描述信息：

```
msf exploit(ms08_067_netapi) > info
```

## 目标列表（译者注：Target List，具体目标操作系统列表，包括版本等）

每一个 Metasploit 漏洞利用都有一个（可用）目标列表。
开发人员在公开发布漏洞利用程序之前，会进行一系列的测试。
如果目标系统不在这个列表中，最好假设该漏洞利用程序从未在该特定的设置上进行过测试。

如果漏洞利用程序支持自动选择目标系统，那么列表中的第一项就会是 "自动选择目标系统"（Automatic Targeting）（也就是索引为 0 的一项）。第一项总是会被默认选中，这意味着如果你以前并没有使用过这个漏洞利用程序那么你就永远不应该假设漏洞利用程序会为你自动选择目标系统，并且这个默认值

"show options" 命令可以告诉我们哪些目标系统已经被选择，例如：

```
msf exploit(ms08_067_netapi) > show options
```

"show targets" 命令会列出所有该漏洞利用程序支持的目标操作系统列表

```
msf > use exploit/windows/smb/ms08_067_netapi
msf exploit(windows/smb/ms08_067_netapi) > show targets

Exploit targets:

  Id  Name
  --  ----
  0   Automatic Targeting
  1   Windows 2000 Universal
  2   Windows XP SP0/SP1 Universal
  3   Windows 2003 SP0 Universal
  4   Windows XP SP2 English (AlwaysOn NX)
  5   Windows XP SP2 English (NX)
  6   Windows XP SP3 English (AlwaysOn NX)
  7   Windows XP SP3 English (NX)
  8   Windows XP SP2 Arabic (NX)
  9   Windows XP SP2 Chinese - Traditional / Taiwan (NX)
  ...
```

## 检查所有选项

所有的 Metasploit 模块都预先配置了大部分的数据存储的选项。然而，它们可能并不适合于你正在测试的目标系统。你可以使用 “show options” 命令来进行一次快速检查：

```
msf exploit(ms08_067_netapi) > show options
```

然而，“show options” 仅为你显示所有的基础选项，一些逃避（译者注：原文 evasion，该选项主要用于逃避反病毒软件或者攻击检测系统的审查，例如在 WEB 漏洞的利用中可以通过随机 HTTP User-Agent 字段来逃避某些审查，例如 Pull Request 5872）和高级选项并不会被显示出来（使用 "show evasive" 和 “show advanced”），可以使用 set 命令来对数据存储选项进行配置：

```
msf exploit(ms08_067_netapi) > set
```

## 寻找模块对应的 Pull Request

Metasploit 仓库是托管在 GitHub 上的（嗯，其实就是你正在浏览的网站），开发者和贡献者在开发中严重依赖 GitHub。在一个模块被公开之前，它将会被作为一个 Pull Request 提交到 Metasploit 仓库，之后该 Pull Request 将会被测试以及被 Review。在一个模块的 Pull Request 页面，你将会找到你需要知道的关于该模块的 所有信息，你或许还能得到一些从模块描述或者一些别的博客文章中没有的信息。这些信息是非常有价值的！

你可以从 Pull Request 中学到的有：

- 搭建漏洞环境的步骤
- 哪些目标环境已经被测试过
- 该模块应该如何使用
- 模块如何被检验并通过审核
- 哪些问题是已经被确定的，其中可能就有你想要知道的
- Demo
- 其他的惊喜

有两种主要的方法可以找到一个指定模块的 Pull Request

通过 Pull Request 的序号：如果你已经知道了具体的 Pull Request 序号，那么就很简单了，只需要访问如下连接

```
https://github.com/rapid7/metasploit-framework/pull/[PULL REQUEST NUMBER HERE]
```

通过搜索（过滤器）
这个方法更加通用。首先你应该先访问：https://github.com/rapid7/metasploit-framework/pulls。在页面顶部，你将看到一个搜索输入框和一个默认的过滤器：`is:open`，这些默认值意味着你正在查看 Pull Request，并且你正在查看那些待定的 Pull Request（等待被合并到 Metasploit 中的 Pull Request）。因为你正在寻找已经被合并到 Metasploit 的模块，所以你需要按照如下操作：

选择已经关闭的 Pull Request

- 选择 “模块” 标签
- 在搜索框中，输入其他和你要搜索的模块相关的关键字。一般来说，模块的标题即可。

注意：如果你想要找的模块是 2011 年写的，那么将不会有关于这个模块 Pull Request（译者注：可能是因为 2011 年才开始在 GitHub 上托管？或者 GitHub 2011 年才上线 Pull Request 功能？）

上一篇：python异步编程之asynci... 下一篇：【译】Metasploit：漏洞利...

1. 0 条回复

- 动动手指，沙发就是你的了！

登录 后跟帖

先知社区

现在登录

热门节点

技术文章

社区小黑板

目录

RSS 关于社区 友情链接 社区小黑板