第二章 网络病毒的典型形态和传播行为

在第 4 章到第 5 章，我们将阐述网络反入侵技术，这是当前最重要、也是应用最广泛的网络安全措施之一。为了能深入理解和正确应用反入侵技术，需要对恶意入侵本身有一个足够深入的认识，为此本章和下一章重点解释典型的网络病毒入侵和传播机制。从逻辑上讲，传播行为发生在入侵之后，但相比之下入侵行为更为复杂，因此这一章先着重阐述网络病毒的传播机制，下一章再较详细地讨论典型的病毒入侵技术。

这里需要说明一点：计算机病毒在网络大量应用之前就已经被创造出来，例如 80 年代中期开始流行的各种 PC 病毒，相信许多人都与作者一样至今记忆犹新。那些病毒主要经磁盘存储介质传播，从这个意义上讲，网络只是提供了一条新的病毒传播途径而已。但事实上，今天的网络已经彻底改变了软件系统相互作用的方式，因此网络病毒已经演变成为计算机病毒家族中非常不同而且变体丰富的种群。

目前观测到的绝大部分网络病毒都具有以下行为阶段：选择目标、确定入侵/感染策略、实施入侵、入侵后自我复制/传播及实施破坏，这些构成网络病毒最典型的形态特征。

2.1 网络病毒如何选择入侵目标

设想病毒 V(无论以何种方式和途径)已经入侵一台计算机 A(或者 A 根本就是黑客本人的计算机)，V 常通过应用层或网络层途径自动检索候选攻击对象。

所谓应用层对象，是指常以网络全局命名来标识的对象，全局命名的例子有域名、主机名、邮件地址、ICQ 标识等。

邮件地址实质上指代的是人类用户，在具体应用中所涉及的往往就是该用户的计算机，因此是借助邮件进行传播的网络病毒的最合适的候选攻击对象。收集电子邮件地址的方式可以是通过标准 API 检索 Windows 地址簿、检索 outlook 地址簿、在 Windows 上直接检索.wab 文件并对其解析以提取出电子邮件地址，其他可利用的文件类型还包括.HTM, PHP, ASP, DBX, PL, 等等。病毒也可以直接在网络环境中收集邮件地址，例如通过 NNTP、WEB、ICQ、监视 SMTP 消息和网络新闻组用户消息等途径。

网络层对象指以网络地址标识的对象，例如 IP 地址和 TCP 或 UDP 端口号。这时攻击者常常需要随机生成拟扫描的目标 IP 地址，或按照某种策略生成 IP 地址，然后以某种形式的消息(请求建立 TCP 连接)对目标对象进行试探。

例如，下面是著名的网络病毒 *Linux/Slapper*[1]的 C 程序中关于地址扫描部分的代码：

```
unsigned char classes[]= \
{3,4,6,8,9,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,….,237,238,239};
A=classes[rand()%(sizeof(classes))];
B=rand();
C=0;
D=0;
```

rand()是一个生成随机数的 API。这一病毒程序片段清楚地表明，该病毒随机生成 B 类 IP 地址的网络号部分。

出自病毒的扫描消息可能具有很强烈的异常特征，例如，观察下面这组关于病毒随机扫描的源、宿 IP 地址和目标 TCP 端口号的记录(每项记录时间间隔约 1 毫秒)，你能发现什么特征吗？

```
176.100.10.5    186.63.210.15:1434
176.100.10.5    73.224.212.240:1434
176.100.10.5    156.250.31.226:1434
176.100.10.5    163.183.53.89:1434
176.100.10.5    142.92.63.3:1434
176.100.10.5    205.217.177.104:1434
176.100.10.5    16.30.92.25:1434
176.100.10.5    71.29.72.14:1434
176.100.10.5    162.187.243.220:1434
176.100.10.5    145.12.18.226:1434
176.100.10.5    196.149.3.211:1434
176.100.10.5    43.134.57.196:1434
176.100.10.5    246.16.168.21:1434
176.100.10.5    149.92.155.30:1434
176.100.10.5    236.17.200.51:1434
……
```

很明显，以上所有这些消息出自同一个源 IP 地址、目标端口号完全一样但目标 IP 地址却几乎随机分布，而且消息密度较高，很难想象有哪一种正常的进程会具有这种行为，这实

[1] 这里使用常用的病毒命名记号，例如 *Linux/Slapper* 指 *Linux* 环境中的名字为 *Slapper* 的病毒; *W32/Borm* 指 *Win32* 环境中的名字为 *Borm* 的病毒，等等。*Linux/Slapper* 爆发于 2002 年前后，攻击 Apache 服务器，是目前最复杂和凶狠的网络病毒之一。

际上反映了病毒消息的特征。另外，联系观测记录的上下文来看，源于病毒的扫描消息在整体上常常不事先伴随 DNS 查询消息，这也是与正常进程很不相同的特点。

网络病毒选择入侵目标还有其他途径，这里不一一列举了。要指出的是，当代网络病毒对目标进行扫描的方式越来越隐密，往往不会再有上面这种强烈的可观测特征，例如某些病毒采用极低频扫描，使自己的目标扫描消息淹没在大量的 IP 分组流的“噪声”之中；或利用无辜的“替罪羊”机器代行扫描，使反入侵系统不能直接观测到自己，例如第 5 章中的例子，凡此种种。有效检测这类行为需要开发消息自动分析工具。

2.2 网络病毒如何确定入侵策略

从本质上讲，网络病毒都是针对特定类型软件的特定漏洞进行攻击(*exploit*)，因此病毒在入侵阶段的首要任务是正确识别目标系统的类型。常用的途径就是向目标机器发送某种(可能是不合法的)消息以引发系统响应、根据响应信息来判定系统类型、进而确定具体的入侵策略。

例如，在成功找到开放 TCP 端口 80 的目标机器后，*Linux/Slapper* 向目标端口 80 故意发送一个缺少 HOST-首部的不正常的 HTTP 消息(HTTP1.1 规定该部分必须被包含)，该端口上的 Apache WebServer 将对此响应以下信息：

```
HTTP1.1/ 400 Bad Request
Date: mon 23 Feb 2004 23:43:42 GMT
Server: Apache/1.3.19   (Red-Hat/Linux) mod_ssl/2.8.1
OpenSSL/0.9.6 DAV/1.0.2 PHP/4..4pl1 mod_perl/1.24_01
Connection: Closed
Transfer-Encoding: Chunked
Content-Type: text/html’ charset=iso-8859-1
```

粗体标记出病毒想要的信息，它清楚地表明目标是 Apache Release 1.3.19 及运行环境是 *Red-Hat/Linux*。从被攻击方的角度，你或许认为言多必失！——但请注意，这些信息并非完全多余，实际上它们用以保证与诚实客户方(浏览器)进程之间的互操作性。

接下来 *Linux/Slapper* 根据以上“目标指纹”信息选择具体入侵参数。仔细阅读下面 *Linux/Slapper* 中与入侵策略有关的数据结构，你能猜测出各个域的用途吗？

```
struct archs{
        char* os;
        char* apache;
```

```
        int     func_addr;
} architectures[] = {
        { “Gentoo”, “”, 0x08086c34 },
        {“Debian”, “1.3.26”, 0x080863cc},
        {“Red-Hat”, “1.3.6”, 0x080707ec},
        {“Red-Hat”, “1.3.9”, 0x0808f614},
        {“Red-Hat”, “1.3.12”, 0x0809251c},
        {“Red-Hat”, “1.3.19”, 0x0809af8c},
        {“Red-Hat”, “1.3.26”, 0x08161c14},
        {“SuSE”, “1.3.12”, 0x0809f54c},
        {“SuSE”, “1.3.23”, 0x08061c38},
        ……
};
```

正如你推断的那样，C 结构 archs 中的域 os 用来存储目标平台的名字、域 apache 存储 Apache 服务器的版本号、域 func_addr 存储 Apache 服务器进程空间中某个函数的逻辑入口地址(关于这些神秘地址的涵义和作用，读者读完第 3.3 节的图 3-5 及其说明就能明白了)，粗体对应的正是当前该病毒所发现的目标特征。这意味着病毒接下来将具体针对 *Red-Hat* 平台上的 *Apache* 服务器 *Release* 1.3.19 的软件缺陷实施入侵。

2.3 网络病毒概观

这一节举例说明当前网络病毒的典型传播行为。

首先是利用后门(Trapdoor)的传播机制，著名的 *W32/Borm* 病毒是这方面的典型例子，它利用 *Back Orifice*[2]的远程命令接口实施入侵和传播。图 2-1 完整描述了这一过程，这里入侵的关键在于破译合法用户的口令然后以合法身份运行目标机器上的特权命令(BO_HTTP_ENABLE)，导致在目标机器上开放 HTTP Proxy 服务，进而导致完整的病毒程序上载到目标机器。病毒程序在目标机器上运行起来(响应 BO_PROCESS_SPAWN 命令)之后，以该机器为平台继续如法实施对其他目标的入侵，由此在网络上扩散开去。

这里需要补充的是，在现实之中口令破译不仅可能而且比我们大多数人想象得要容易，甚至在今天(实际上是多年之前)因特网上就有着公开可获取的口令破译工具，一般的口令字可以在几个小时到几天内被彻底破译。抵抗口令破译需要正确的密码方案及协议设计，而不是简单依靠加密，第 11 章将对此详细讨论。

[2] Back Orifice 是一种很流行的 Windows 远程系统管理程序，参见 http://bo2k.sourceforge.net/whatis.html。

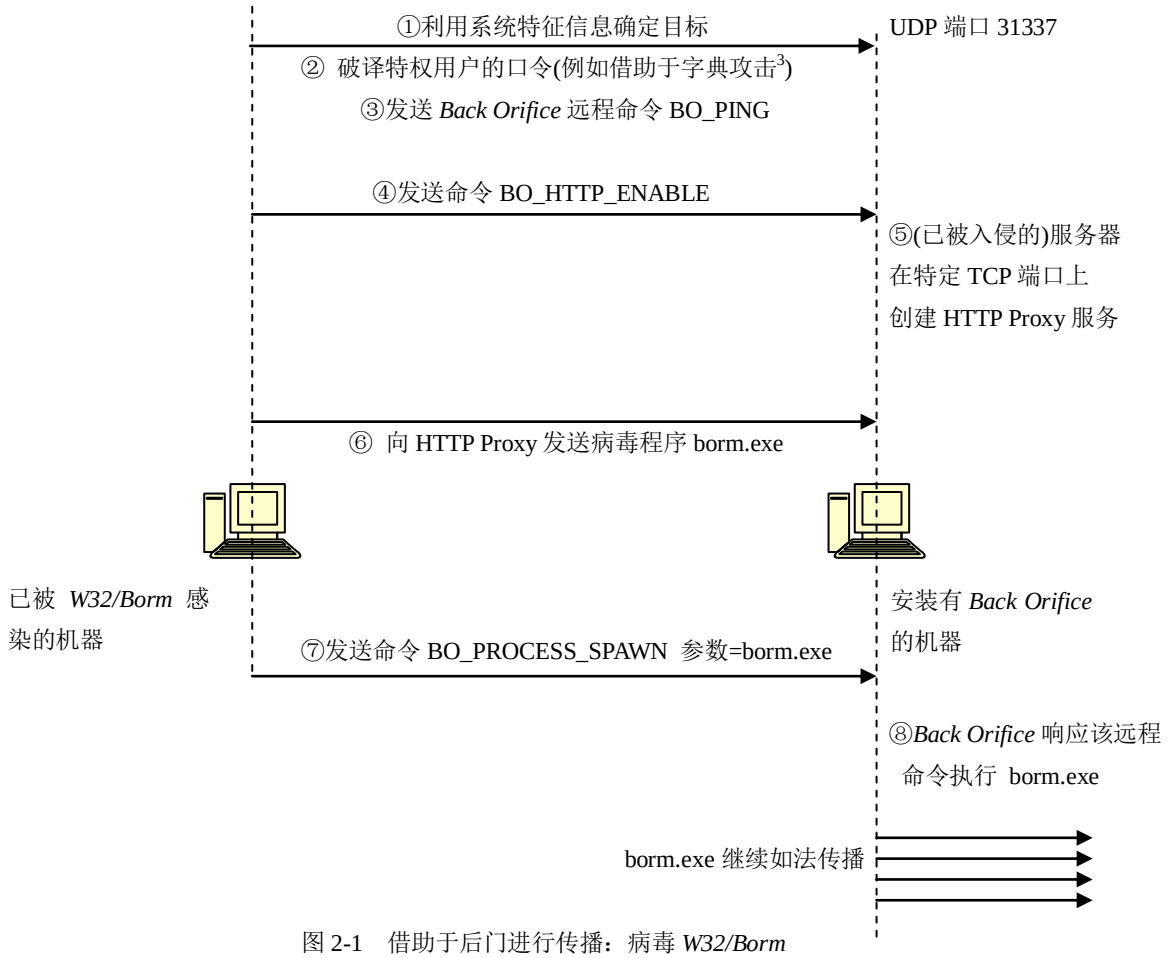


图 2-1 借助于后门进行传播：病毒 *W32/Borm*

病毒传播的第二条典型途径是利用 SMTP 代理，这是许多邮件病毒的传播机制，例如病毒 *W32/Taripox@mm*[4]，图 2-2 描述了其传播过程。该病毒在 *Windows* 环境中获取邮件服务器地址，例如从注册表中的路径

HKCU\Software\Microsoft\Internet Account Manager\Default Mail Account\Accounts\SMTP Server

就可以获得；病毒将该地址更新为本机 IP 地址，并启动自己的 SMTP 代理进程，如此一来，任何从本机发出的邮件都被首先传递到该(恶意的)SMTP 代理进程；该进程将病毒程序作 MIME 编码后作为附件连接到邮件上，然后将处理过的邮件发送(这一次是真正向外发送!)给合法的 SMTP 服务器，以此将病毒自己传播到邮件的接收方机器。

顺便指出，在 *Windows* 注册表中当前用户的邮件地址在以下路径定义

HKCU\Software\Microsoft\Internet Account Manager\Default Mail Account\Accounts\SMTP Email Address

[3] 字典攻击将在第 11 章阐述。
[4] *W32/Taripox@mm* 与前面的病毒命名法一致, *@mm* 表示其变体。

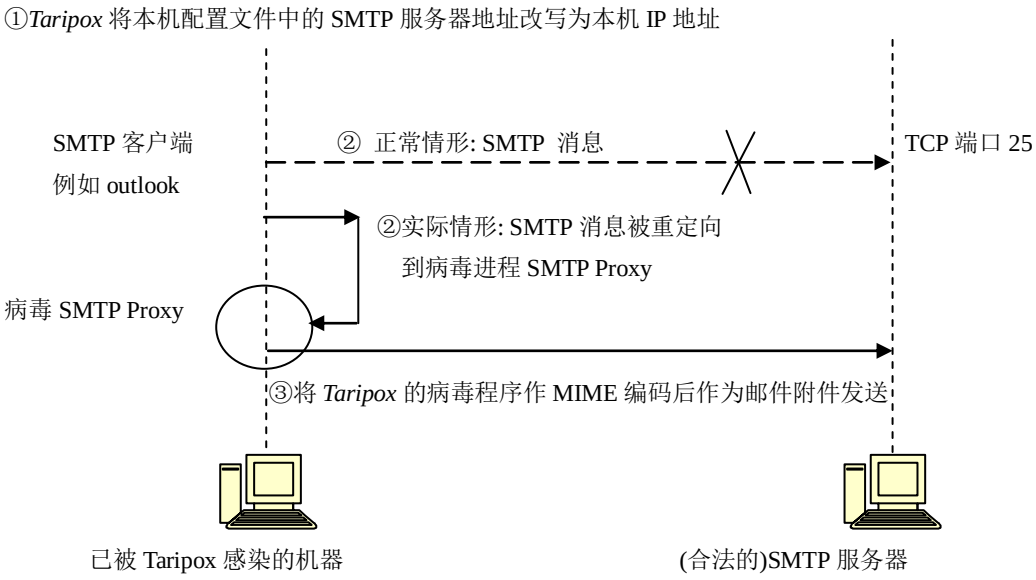


图 2-2 借助于不正当的消息重定向进行传播：病毒 *W32/Taripox@mm*
(图中的“SMTP 消息”即指电子邮件)

SMTP 是基于 TCP 的邮件传输协议，对不熟悉 SMTP 协议的读者，图 2-3 概要描述了其工作过程，其中消息#220 指 SMTP 消息类型代码，它指明该消息的用途。

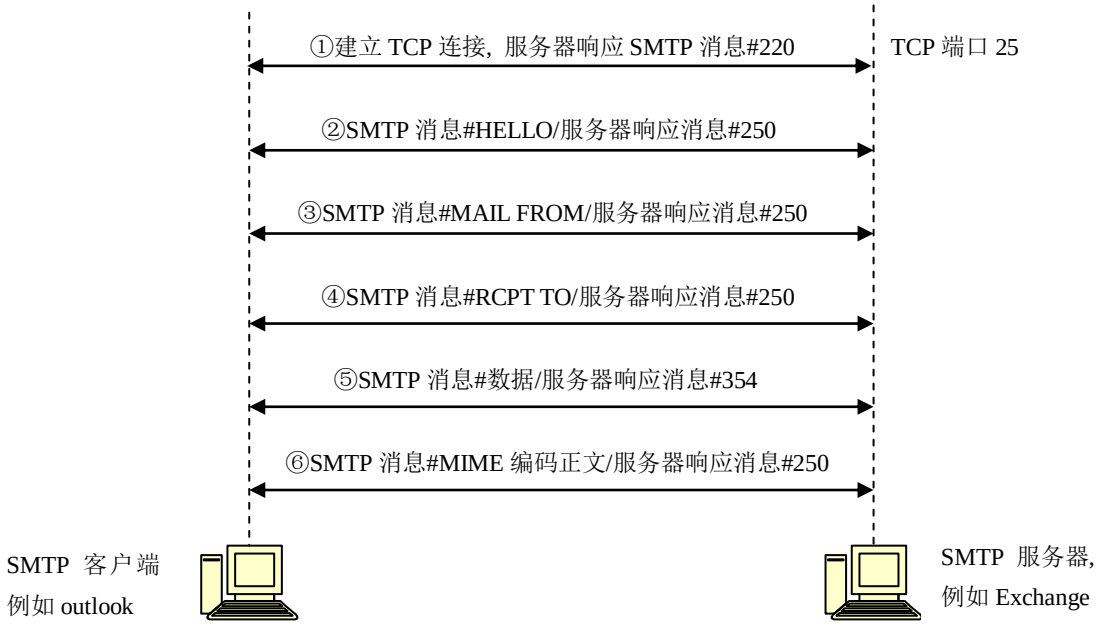


图 2-3 SMTP 协议过程
(其中较复杂的消息往返过程仅以双箭头表示)

病毒 *W32/Taripox@mm* 的传播方式是典型的所谓推式传播，同样利用非法重定向机制，病毒 *W32/Beagle.T* 采用所谓拉式传播方式，见图 2-4。

病毒 *W32/Beagle.T* 利用了 SMTP 和 Web 代理进程，仿照图 2-2 的机理，请读者自行解释这一传播过程。

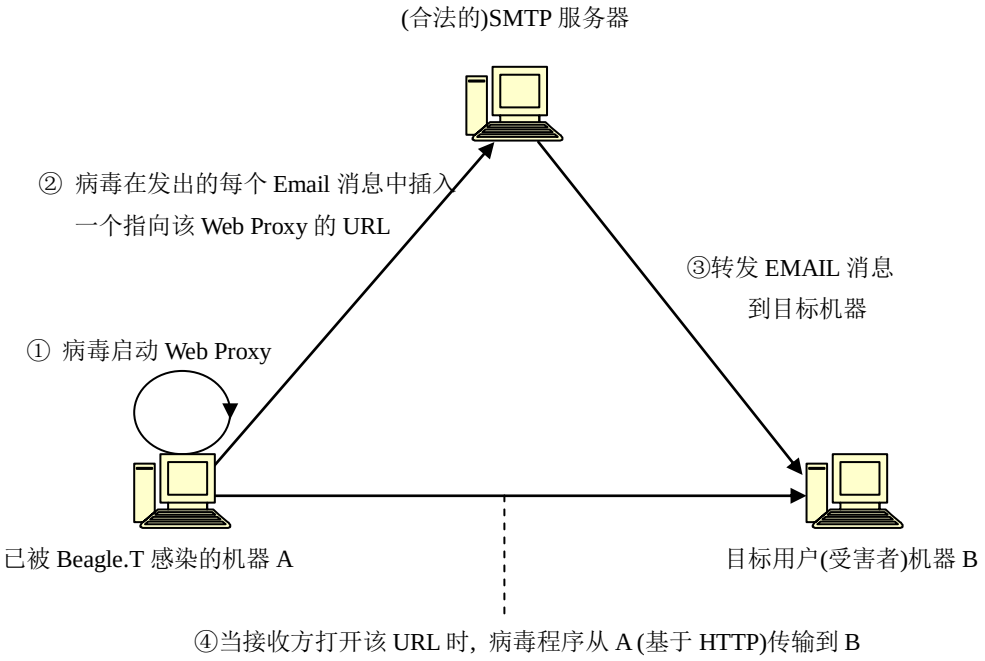
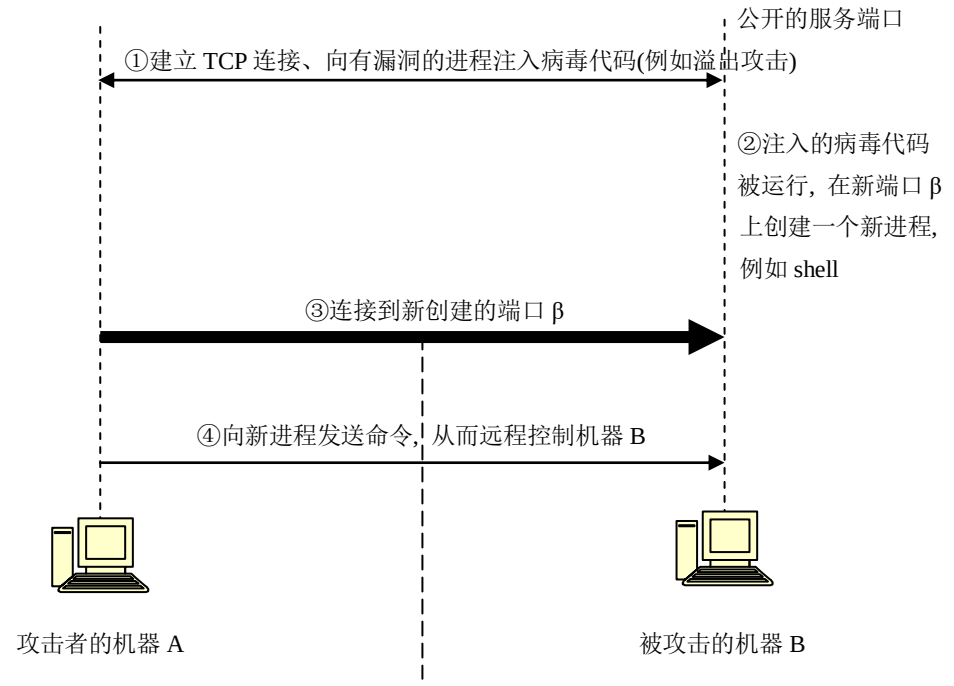


图 2-4 借助于非法重定向的拉式传播：病毒 *W32/Beagle.T*

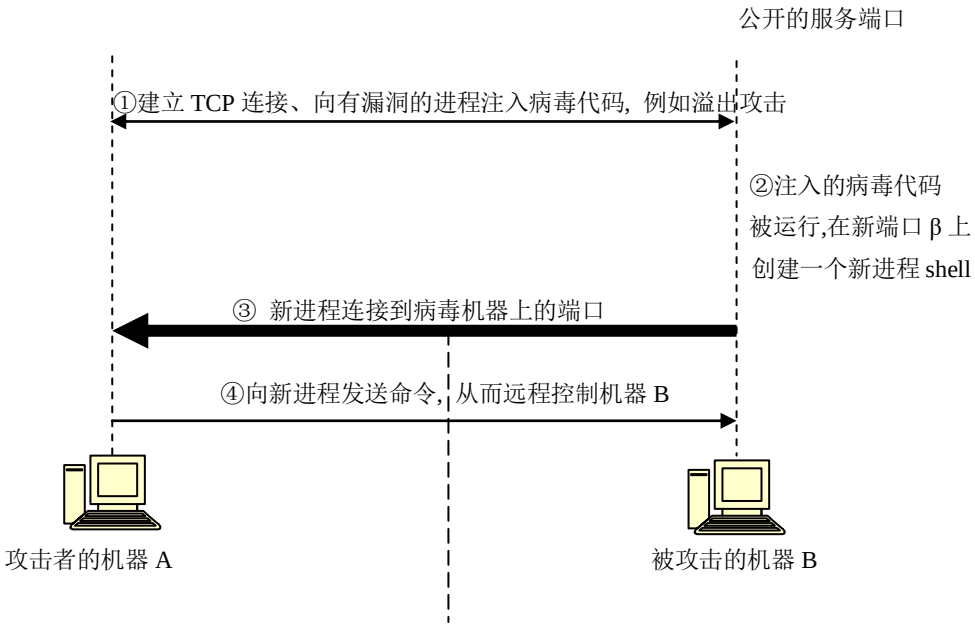
拉式传播较之推式传播的“优点”在哪里？对病毒而言，最大的优点在于更加隐秘，因为图 2-4 (第 2 步和 3 步)中的邮件不再需要携带附件，从而不至引起接收方警觉)，而下载一个 URL 所指代的对象往往是接收进程(无论 outlook 还是浏览器 IE)的隐式动作，接收方可能事先不被给予任何警告。

有一大类网络病毒不是象以上病毒这样利用某种消息完整地传播自己，而是先通过所谓代码注入使目标进程完全置于攻击者的控制之下，然后再实施后续入侵和破坏，包括基于被入侵的进程继续对其它目标进程进行入侵。下一章将详细分析几类典型的代码注入式入侵过程，这里仅用图 2-5 给出该病毒的整体工作过程。因为攻击者第一次能够向目标进程注入的代码量通常很小(典型为 100 字节左右的机器指令编码)，能够完成的任务很有限，因此这些被注入的指令往往用来创建一个新的 TCP/UDP 端口和一个在该端口上处于侦听态的进程，例如目标平台上的命令解释程序(如 *Unix/Linux* 环境中的各种 shell：bsh、csh、ksh 等，*Windows* 环境中的 cmd.exe)，一旦完成这一工作，后续入侵便可以通过攻击者进程与

已经运行起来的目标端进程之间的通讯来继续完成。这类攻击因此也称为 shellcode 攻击。



a）端口 β 可能不在 B 方的防火墙所允许的合法目标范围之内，
从而该连接请求被禁止。



b)改进的攻击行为：B 方的防火墙常不禁止源于内部非特权端口
的 TCP 连接，从而该连接成功传输。

图 2-5 代码注入攻击: 病毒 W32/Blaster

有趣的时，当今许多防火墙(第 5 章详细讨论)往往禁止被保护的网络内部的非特权 TCP 端口(即端口号在 1024 以上的端口，病毒代码在这一阶段常常只具有创建这类端口的权限)

接受外部发起的连接请求，这意味着当 TCP 连接请求是从攻击者的进程向目标进程主动发起时，防火墙会屏蔽该消息，攻击者所需要的 TCP 连接不能如愿建立。但这不足以抵挡该攻击顺利完成。事实上，大多数防火墙被配置成不禁止发自内部非特权端口的 TCP 连接请求（请思考为什么？），攻击者只要利用这一特点，即可使入侵过程继续进行，见图 2-5(b)。

病毒传播机制不仅用以单纯的自我表现复制/扩散，也用以病毒的自我更新/升级。图 2-6 描述了经典的病毒 *W95/BabyLonia* 利用公共服务器进行更新/升级传播的机理。

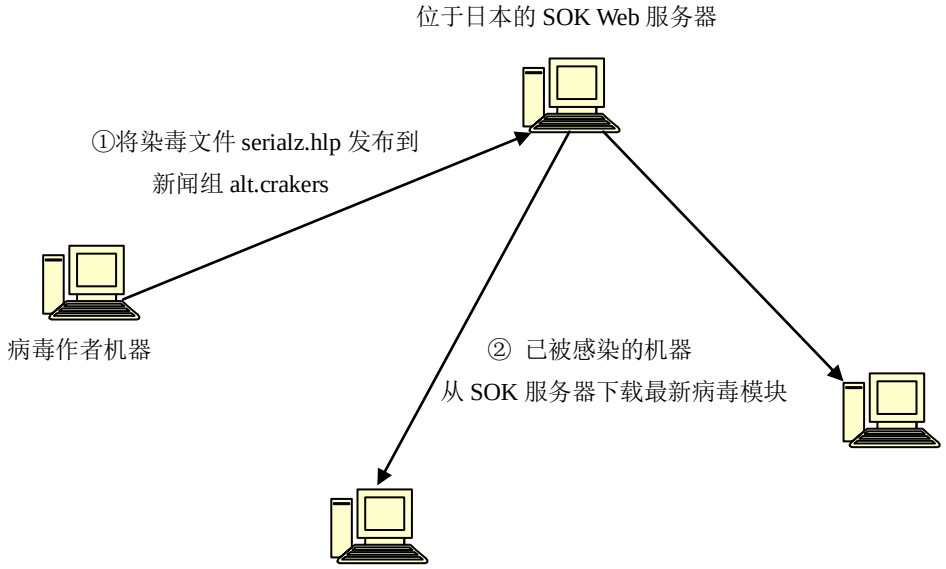


图 2-6 病毒的更新/升级过程：*W95/BabyLonia*

从病毒作者本身的利益看，以上机制有明显的不足：仅利用公开的单点服务器提供更新服务；病毒模块没有完整性保护(即接收方不能验证所下载的是否为真正的病毒程序)。实际上，后期的病毒如 *W95/Hybris* 已经有能力实现更可靠、更隐蔽的自我更新/升级机制，包括利用加密、散列和 1024 位 RSA 数字签名来保护病毒程序本身不被篡改[5]。

2.4 小结与进一步学习的指南

病毒形态丰富，本章仅给出了一个概要，关于计算机病毒内容丰富的参考书是

P.Szor *The Art of Computer Virus Research and Defense*, Symantec Press, 2005

这本书的内容不限于网络病毒，书后还附有许多病毒方面的文献和专业研究团体的网站地

[5] 这些密码机制原本用以保护合法用户，但具讽刺意味的是，目前越来越多的病毒也利用这些技术来保护自己，而且某些病毒在这方面的水平毫不逊色。第 III 部分特别是第 8 章详细阐述这些密码技术。

址。下面这本名著从网络入侵检测(本书在第 5 章讨论)的角度汇集了许多网络病毒实例及分析：S.Northcutt et al, *Network Intrusion Detection: An Analyst's Handbook*, Addison-Wesley Inc. 2003

无论是网络安全实验还是实际从事安全分析工作，掌握基本工具都是必须具备的技能之一。这里给出两个附录，分别介绍两组常用的软件工具，读者可以利用它们开始自己的实践。

附录 2.1 Windows 环境常用的网络命令

Microsoft Windows 是使用最多的操作系统，使用 Windows 环境下的网络命令来获取网络参数或检测网络状态将是最常用的方法。Windows 提供的网络命令很多，这里介绍最常用的 6 个命令：ping, ipconfig, arp, netstat, tracert, 以及 nslookup。获得这些命令的帮助的一个通用方法是在命令行窗口中键入这些命令的名字，并且后面跟“/?”作为命令行参数（注意在 Unix/Linux 上，类似的命令也存在而且用法相似）。

1、 ping 命令

ping 主要用于确定本地主机是否与网络上另外一台主机连通。按照缺省设置，Windows 上运行的 Ping 命令共发送 4 个 ICMP Echo Request 消息，每个消息含有 32 个字节的数据，如果网络连通，我们应能收到 4 个 ICMP Echo Reply 消息。Ping 命令的可选参数有很多，以下仅介绍两个较为常用的参数使用方法：

ping [-t] [-n count] 主机地址

其中：

• ping –t 主机地址

连续向主机地址发送 Echo Request 消息，直到被用户以 Ctrl+C 中断。

• ping –n count 主机地址

向主机地址发送“count”个 Echo Request 消息，然后停止。

2、 ipconfig 命令

ipconfig 用于显示当前 TCP/IP 配置的许多网络参数，例如：IP 地址，子网掩码，缺省路由器的 IP 地址，以及网卡的 MAC 地址等(Windows 95 和 Windows 98 使用 winipcfg 命令而不是 ipconfig 命令)。ipconfig 的一般用法是：

ipconfig [/all | /renew | /release | /displaydns | /flushdns]

其中各参数的含义如下：

• ipconfig

当使用 ipconfig 时不带任何参数选项，那么显示 IP 地址、子网掩码和缺省路由器的 IP 地址。

• ipconfig /all

当使用 /all 选项时，ipconfig 能显示 DNS 服务器的 IP 地址，以及本机网卡的物理地址（即 MAC 地址）等。如果 IP 地址是从 DHCP 服务器动态获取的，”ipconfig /all”将显示 DHCP 服务器的 IP 地址和所获取的 IP 地址失效的时间。请重点掌握的是，”ipconfig /all”是获得本机网卡的物理地址的最常用方法。

• ipconfig /renew 和 ipconfig /release

这两个选项只能在从 DHCP 服务器动态的获得 IP 地址的计算机上使用。如果我们输入 ipconfig /release，那么网卡所动态获得的 IP 地址便会归还给 DHCP 服务器。如果我们输入 ipconfig /renew，那么计算机便会与 DHCP 服务器进行联系，从而获得一个 IP 地址。

• ipconfig /displaydns 和 ipconfig /flushdns

Ipconfig /displaydns 用于显示 DNS 缓存中的所有内容; ipconfig /flushdns 用于删除 DNS 缓存中的所有内容。DNS 中存放的是最近的域名解析操作所获得的域名和 IP 地址的对应关系。

3、arp 命令

ARP 是用于由 IP 地址解析 MAC 地址的协议。ARP 协议把解析到的 MAC 地址保存在内存的一块空间中，这块内存空间被称为 ARP 缓存。使用 arp 命令，能够查看本机 ARP 缓存中的当前内容。arp 命令的一般用法是：

arp [-a | -d IP 地址]

其中：

• arp -a

用于查看 ARP 缓存中的所有项目。（-a 可被视为 all，即全部的意思）

• arp -d IP 地址

使用本命令能够人工删除对应于该 IP 地址的 MAC 地址项目。如果使用”arp –d *”则可以删除 ARP 缓存中的所有项目。

4、netstat 命令

netstat 用于显示当前存在的 TCP 连接，路由表，与 IP、TCP、UDP 和 ICMP 协议相关的统计数据等，一般用法是：

netstat [-s] [-e] [-r] [-a] [-n]

其中：

• netstat –s

本选项能够按照各个协议分别显示其统计数据，例如每个协议所收到的和发出的包的个数等。

• netstat -e

本选项用于显示关于以太网协议的统计数据。它列出的项目包括收到的或发出的以太网帧的总字节数、收到的或发出的以太网帧的个数等。

• netstat –r

本选项可以显示本机路由表的信息。

• netstat –a

本选项显示当前所有被使用的 TCP 或 UDP 端口号的信息，包括正处于监听（LISTENING）状态的 UDP 或 TCP 端口号，以及已建立起连接的（ESTABLISHED）TCP 端口号等。

• netstat –n

显示所有已建立的 TCP 连接，并且主机地址和端口号用数字表示，不做域名解析。

5、tracert 命令

tracert 用于依次列出到达某目的主机路径上的每个路由器的 IP 地址。tracert 的工作原理是利用如下事实：IP 分组中 TTL 域的作用是防止该 IP 分组在网络上被无穷的转发下去；每个路由器在收到一个 IP 分组的时候，都会把该 IP 分组中的 TTL 域减 1，如果 TTL 的值等于 0，路由器就会抛弃该 IP 分组，并向该 IP 分组的源主机发送一个 ICMP Time Exceeded 消息。Tracert 命令正是利用 Time Exceeded 消息来确定路径上每个路由器的 IP 地址的。Tracert 第一轮发送 TTL=1 的 ICMP Echo Request 消息给目的主机，当该 Echo Request 消息到达路径上的第一个路由器的时候，该路由器将 TTL 减 1，得到 TTL=0，故该路由器会返回 Time Exceeded 消息，这样 tracert 就从该消息的源 IP 地址知道了第一跳路由器的 IP 地址；在第二轮，tracert 会发送 TTL=2 的 ICMP Echo Request 消息给目的主机，该 Echo Request 消息中的 TTL 值会在第二跳路由器被减为 0，故第二跳路由器会返回 Time Exceeded 消息，这样 tracert

就从该消息的源 IP 地址知道了第二跳路由器的 IP 地址；以此类推，tracert 会依次探测第三跳，第四跳等后面路由器的 IP 地址，直到 Echo Request 消息最终到达目的主机，这时该目的主机会返回 Echo Reply 消息。tracert 的一般用法是：

tracert [-d] IP 地址 | 主机域名

如果使用 -d 选项，那么 tracert 不会去尝试解析路径中路由器的 IP 地址所对应的域名，因此将更快地显示路径上每个路由器的 IP 地址。作者在这里建议读者在执行 tracert 时使用 –d 选项。

6、nslookup 命令

nslookup 命令主要用于解析一个域名所对应的 IP 地址。它的一般用法是：

nslookup 主机域名

例如在本机上执行：nslookup ssdut.dlut.edu.cn

将返回以下结果：

C:\Documents and Settings\ibm>nslookup ssdut.dlut.edu.cn

Server: cedrus.dlut.edu.cn

Address: 202.118.66.6

Name: ssdut.dlut.edu.cn

Address: 210.30.96.93

其中，202.118.66.6 是被查询的域名服务器的地址，210.30.96.93 则是 ssdut.dlut.edu.cn 所对应的 IP 地址。

附录 2.2 网络协议分析工具 Ethereal

Ethereal 是目前使用最广泛的网络协议分析工具，能够实时地捕获网络上的 IP 分组，并且把分组中每个域的细节显示出来。Ethereal 的广为流行主要有以下三个原因：功能非常强大，这一点随着大家对 Ethereal 的熟悉将会非常深切的感受到；开放源代码；具有非常详细的文档。Ethereal 的安装文件和文档都可以在 ”http://www.ethereal.com” 下载。为了帮助读者迅速的学会使用 Ethereal，下面分三个方面对 Ethereal 的用法做出介绍。

1、图形界面

Ethereal 的图形界面主要分为四个部分，如图 2-7 所示。第一部分是菜单项和工具栏，Ethereal 提供的所有功能都可以从这一部分中找到；第二部分是被捕获 IP 分组的列表，给出了被捕获 IP 分组的一般信息，例如被捕获的时间，源和目标 IP 地址，所属的协议类型，以及 IP 分组的类型等信息;第三部分显示第二部分中被选中的 IP 分组的每一个域的具体信息，从以太网帧的首部开始，一直到该 IP 分组中负载的内容，都会被显示得清清楚楚；第四部分显示第二部分中被选中的 IP 分组的 16 进制和 ASCII 表示，以帮助用户了解一个 IP 分组的本来的样子。

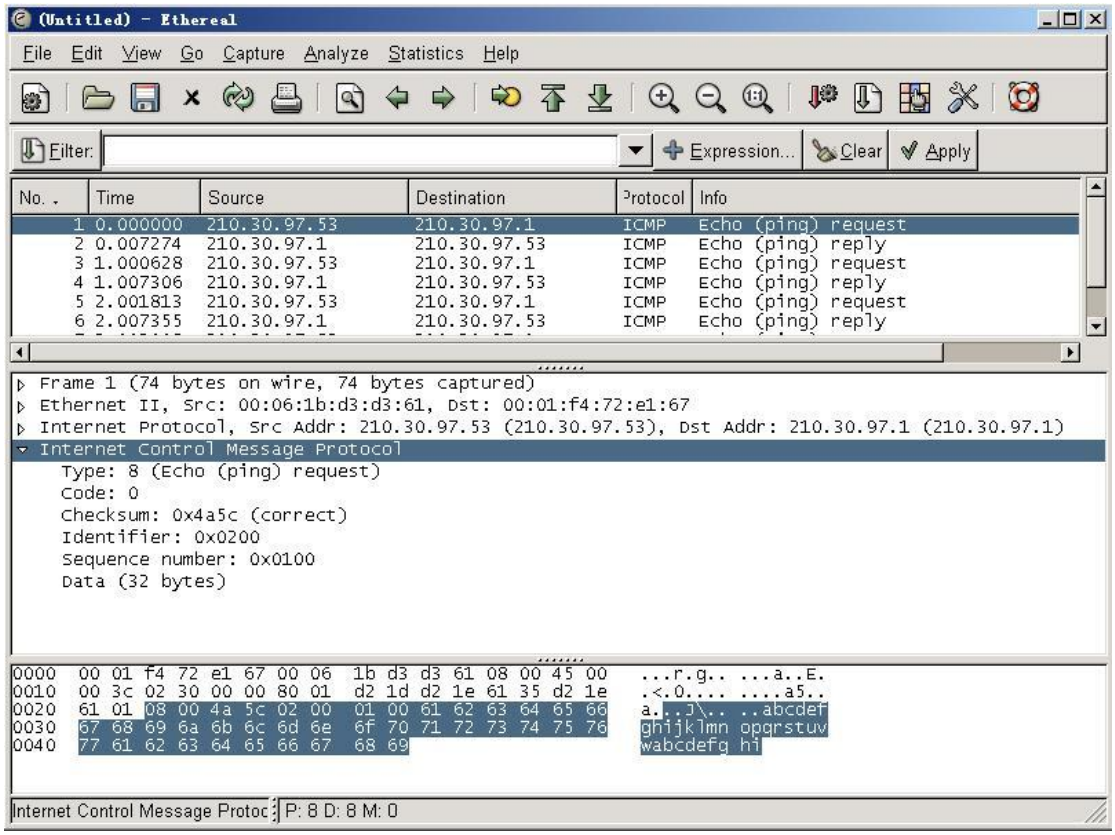


图 2-7 Ethereal 图形界面

2、 Ethereal 的基本用法

Ethereal 最基本的功能是被用来捕获网络 IP 分组，使用这个功能的方法很简单，按如下步骤操作即可：

(1) 选择”Capture”菜单项中的”Start”，这时会弹出一个对话框，如图 2-8 所示。这个对话框中的栏目虽然很多，但一般只需要配置其中的两项即可。一是“Capture Filter”栏，

在这个栏中，可以输入过滤规则，用于规定 Ethereal 捕获 IP 分组的种类（注：关于过滤规则的写法，将在下一小节作专门介绍）；如果不输入任何过滤规则，Ethereal 将捕获所有从网卡发送或收到的 IP 分组。另外一个是“Update list of packets in real time”选项，请大家一定要选择这一项，这样可以使 Ethereal 在捕获 IP 分组的同时，实时地把捕获到的 IP 分组显示出来。

(2) 在做完如上配置后，点击 OK 按钮，Ethereal 就会开始捕获 IP 分组了。

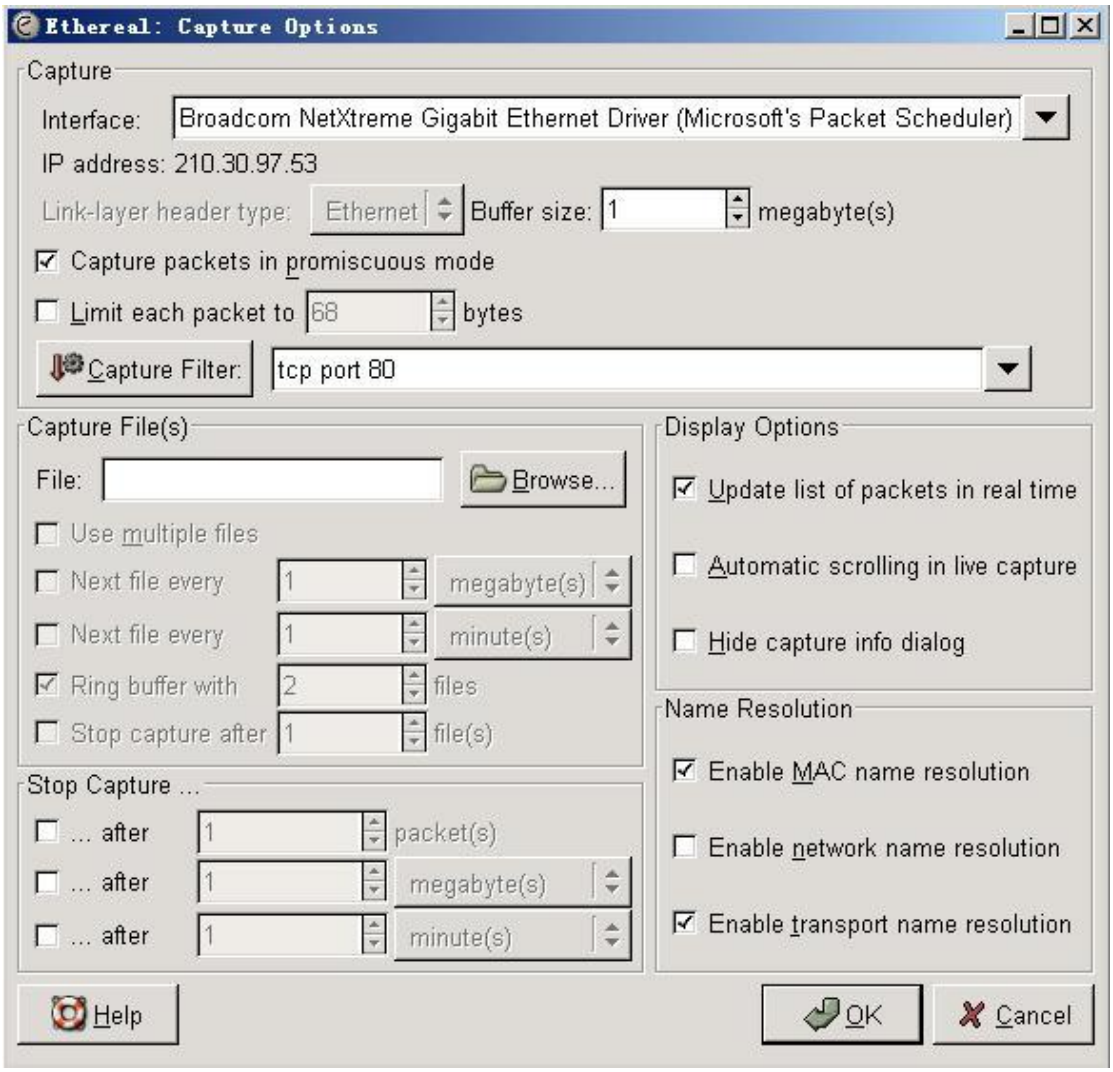


图 2-8 Ethereal 捕获 IP 分组对话框

3、 Ethereal 的过滤规则

分析者并非总需要观察所有 IP 分组，而常常仅需要观察某些特定类型的 IP 分组，例如来往于特定 IP 地址之间、承载特定协议的 IP 分组，因此需要设置过滤规则以简化分析。

Ethereal 过滤规则的写法与另外一个早期但很著名的字符界面的 IP 分组捕获工具“tcpdump”一致(获得“tcpdump”过滤规则帮助的方法是在 Linux 操作系统下执行“man tcpdump”命令)。www.ethereal.com 的文档也详细描述了 Ethereal 过滤规则的写法。

总的来说，Ethereal 的过滤规则可以有两种形式：

(1) 一个原语（一个原语即一条最基本的过滤规则）；

(2）用 ”and”, “or”, “not” 关系运算符，以及括号”()”组合起来的原语；其中，”and”的含义是它所连接的两个原语必须都成立，“or”的含义是它所连接的两个原语只要有一个成立即可，“not”的含义是它后面跟的原语不成立；括号”()”的作用是可以对关系运算符的运算顺序作出规定。

从上面的描述可以看出，只要掌握原语的写法，再用关系运算符把它们组合起来，就可以写出满足各种不同要求的过滤规则了。Ethereal 提供的原语非常多，这里只介绍最常用的 4 种（在下面的叙述中， “[ ]”表示可选项，”< >”表示必须存在的项, “ | ”表示两者选择其中之一，其它的字符串则是关键字了，必须照写不误。）：

(1) ether [src|dst] host <mac_addr>

若无可选项“src|dst”，这条原语用于捕获源和目的 MAC 地址其中之一是“mac_addr”的以太网帧；如果加上“src”或“dst”的限制，则分别仅捕获源或目的 MAC 地址是“mac_addr”的以太网帧。

例如原语 ”ether host 08:00:1B:D3:D3:61” 的含义是捕获所有源或目的 MAC 地址是 ”08:00:1B:D3:D3:61”的以太网帧；原语 ”ether src host 08:00:1B:D3:D3:61” 的含义是捕获所有源 MAC 地址是 ”08:00:1B:D3:D3:61”的以太网帧。

(2) [src|dst] host <ip_addr>

若无可选项“src|dst”，这条原语用于捕获源和目的 IP 地址其中之一是“ip_addr”的 IP 分组；如果加上“src”或“dst”的限制，则分别仅捕获源或目的 IP 地址是“ip_addr”的 IP 分组。

例如原语 ”host 210.30.97.53” 的含义是捕获所有源或目的 IP 地址是 ”210.30.97.53” 的 IP 分组；原语 ”dst host 210.30.97.53” 的含义是捕获所有目的 IP 地址是 ”210.30.97.53”的 IP 分组。

(3) [tcp|udp] [src|dst] port <number>

这条原语用于捕获承载 TCP 或 UDP 协议的 IP 分组, 源或目的端口号是”number”的 IP 分组；可以用可选项”tcp”和 “udp”来对传输层协议进行限制，用可选项”src”和”dst”来对端

口号是源还是目的进行限制。

例如原语 “tcp port 80” 的含义是捕获所有源或目的端口号是 80 的 TCP 协议的 IP 分组；原语 “udp dst port 53” 的含义是捕获所有目的端口号是 53 的 UDP 协议的 IP 分组。

(4) arp, ip, icmp, udp, tcp 等

这条原语用于捕获属于某种协议类型的 IP 分组，协议的类型可以是”arp”, “ip”, “icmp”, “udp”, 以及”tcp”等。

例如只含一个关键字的原语 “icmp”的含义是捕获所有 ICMP 协议的 IP 分组。

以上介绍了四种最常用的原语的写法，下面给出两个例子来说明如何把这些原语组合起来以构造比较复杂的过滤规则。

例 1：捕获从主机 192.168.0.10 发出或收到的，除 HTTP 协议之外的网络 IP 分组，过滤规则为 host 192.168.0.10 and not tcp port 80

例 2：记主机 192.168.0.10 为 A，捕获 A 与主机 192.168.0.20 或 A 与主机 192.168.0.30 之间的网络 IP 分组，过滤规则为 host 192.168.0.10 and (host 192.168.0.20 or host 192.168.0.30)

习 题

2-1 你能编写出以上图 2-2 和图 2-4 类型的传播程序吗？选择 *Win32* 或 *Unix/Linux* 平台，假设进程具有合法身份(暂不考虑入侵)，考虑一个软件设计方案，尽可能地明确所有技术细节、尽可能详细地查阅你所需要的技术说明。

2-2 即使不借助于复杂的密码机制，你能设计出一些尽可能可靠、隐蔽、抗篡改的更新机制吗？

2-3 以下是关于运用附录 2.1 中所介绍的网络命令的一些实验，需要的实验设备为一台可访问因特网的 PC 机，安装有 TCP/IP 的 Windows 2000/XP/2003 Server 均可(注：在实验时可更换 下面的域名 www.dlut.edu.cn 为因特网上的任何一台其它主机的域名)。

(1) 使用 ipconfig 命令获得本机的网卡物理地址，IP 地址，缺省路由器的 IP 地址，以及 DNS 服务器的 IP 地址。

(2) 执行命令“ping –n 6 www.dlut.edu.cn”，请解释此条命令的含义；根据此条命令的执行结果说出由你所在的主机到 www.dlut.edu.cn 的平均往返时间是多少？丢包率有多大？

(3) 执行命令“tracert www.dlut.edu.cn”，简要解释 tracert 的工作原理；列出通往 www.dlut.edu.cn 路径上的路由器的 IP 地址。

(4) 使用 arp 命令列出 ARP Cache 中的所有表项；使用 arp 命令删除 ARP Cache 中的所有表项。

(5) ping 缺省路由器或本局域网内的另外一台主机。

(6) 使用 ARP 命令列出 ARP Cache 中的所有表项。

请把以上所执行的命令（包括命令行参数）和所获得的结果写到实验报告中，并解释所观察到的现象。

(7) 执行命令“netstat –a”，根据命令输出指出本机处于 LISTENING 状态的 TCP 端口号和本机上处于 ESTABLISHED 状态的 TCP 端口号。

(8) 使用 netstat 命令列出本机路由表。

(9) 执行命令“nslookup mail.tom.com”，请解释该命令所返回的结果。

2-4 以下是关于运用附录 2.2 中所介绍的网络工具的一些实验，需要一台连接到因特的 PC 机，该 PC 机使用 DHCP 动态获得 IP 地址(此条件不满足则免做(4))，并且安装有 Ethereal 软件。

(1) 用 Ethereal 观察 ARP 协议以及 ping 命令的工作过程：

用“ipconfig”命令获得本机的 MAC 地址和缺省路由器的 IP 地址；用“arp”命令清空本机的缓存；运行 Ethereal，开始捕获所有源或目的是本机的 Ethernet 帧(提示：在设置过滤规则时需要使用本机的 MAC 地址)；执行“ping 缺省路由器的 IP 地址”；

请给出所执行的完整命令和需要设置的 Ethereal 的过滤规则，并解释用 Ethereal 所观察到的网络现象。

(2) 用 Ethereal 观察 tracert 命令的工作过程：

运行 Ethereal, 开始捕获 tracert 命令中用到的消息(提示：tracert 使用 ICMP 消息，具体地讲，tracert 发出的是 ICMP/Echo Request 消息，触发返回的是 ICMP/Time Exceeded 消息和 ICMP/Echo Reply 消息)；执行“tracert -d www.dlut.edu.cn”

请根据 Ethereal 所观察到的现象解释 tracert 的工作原理。

(3) 用 Ethereal 观察 TCP 连接的建立过程：

启动 Ethereal, 配置过滤规则为捕获所有源或目的是本机的 Telnet 协议中的 IP 分组（提示：Telnet 使用的传输层协议 TCP 和端口号 23)；在 Windows 命令行窗口中执行命令 “telnet bbs.dlut.edu.cn”，登录后再退出。

根据 Ethereal 捕获到的 IP 分组解释 TCP 三次握手的连接建立过程；

(4) 用 Ethereal 观察 DHCP 协议的运行过程：

执行命令 “ipconfig /release”，释放本机正在使用的 IP 地址；启动 Ethereal，开始捕获 DHCP 消息(提示：DHCP 使用 UDP 协议，服务器端使用端口号 67，客户机端使用端口号 68)；执行命令“ipconfig /renew”，重新获得一个 IP 地址。

假设所获得的 IP 地址的租用期（lease time）即将到期，执行命令“ipconfig /renew”来进行延续；

执行命令“ipconfig /release”，释放目前获得的 IP 地址；

执行命令“ipconfig /renew”，重新获得一个 IP 地址；

停止 Ethereal 对 DHCP 消息的捕获。

请根据 Ethereal 观察到的现象回答下列问题：

DHCP 的消息类型是由消息中的那个域标识出来的？

使用 DHCP 来获得 IP 地址共需要几步消息交换？这些消息分别是什么消息？

在 DHCP 中，延续租用期共需要几步消息交换？这些消息分别是什么消息？你的 PC 机所获得的租用期的时间是多长？

在 DHCP 中，释放当前使用的 IP 地址需要发送什么消息？