



黑客攻击与防范技术（下）

中科院计算所教育



漏洞扫描和攻击概述

漏洞扫描是一种最常见的攻击模式，用于探测系统和网络漏洞，如获取系统和应用口令，嗅探敏感信息，利用缓存区溢出直接攻击等。

针对某一类型的漏洞，都有专门的攻击工具。另外，也有一些功能强大综合扫描工具，针对系统进行全面探测和漏洞扫描。



口令破解

- 弱口令扫描
- 暴力穷举
 - ❑ 完全取决于机器的运算速度
- 字典破解
 - ❑ 大大减少运算的次数，提高成功率
- 密码监听
 - ❑ 通过嗅探器监听网络中的数据包
- 社会工程学
- 木马和键盘记录程序

常见弱口令

- **!@#\$% !@#\$%^ !@#\$%^& !@#\$%^&* *****
- **@@@ 000000 00000000 0007 007 007007**
- **0246 111 123 1234 12345 123456 1234567**
- **12345678 123456789 1a2b3c 1p2o3i 1q2w3e**
- **1qw23e 1sanjose 2004 2222 369 4444 4runner**
- **54321 654321 777 7777 888888 911 99999999**
- **a12345 a1b2c3 a1b2c3d4 aaa aaaaaa abby
abc**
- **abc123 abcd abcd1234 abcde abcdef abcdefg**
- **access access action active adam adg**
- **adm admin Admin admin123 admin123456**
- **administrator Administrator administrator123**

常见弱口令

- administrator123456 administratorpasswd
adminpasswd adminpwd asdf asdfg asdfgh
- asdfjkl asdfjkl; bill bin daemon doc fgh free
- freedom fuck fuckyou god guest hacker job
- kim love loveyou lp morris mp3 mypass
- mypass123 mypc mypc123 newpass nice
- nobody parol pass passwd Password password
- Password pentium pizza planet playboy ppp
- pw123 pwd qwerty root rose sex sexy shit
- shotgun sos spirit spring sprite ssssss
- storm super superman support sys telecom
- temp test test1 test123 upload warez xxx
- xxxx ytrewq zxcvb zxcvbnm



口令破解

- 弱口令扫描
 - ☐ 流光
 - ☐ NAT
 - ☐ CNIPC NT
- 暴力穷举/字典破解
 - ☐ WMICracker
 - ☐ SMBCrack
- 实例



针对口令破解攻击的防范措施

- 安装入侵检测系统，检测口令破解行为
- 安装安全评估系统，先于入侵者进行模拟口令破解，以便及早发现弱口令并解决
- 提高安全意识，避免弱口令



网络嗅探破解口令

- 以太网采用了**CSMA/CD**技术，由于使用了广播机制，所以，所有与网络连接的工作站都可以看到网络上传递的数据
- 如果两台主机进行通信的信息没有加密，只要使用某些网络监听工具，例如, Sniffer Pro, NetXray, tcpdump , Dsniff等就可以轻而易举地截取包括口令、帐号等敏感信息。



网络嗅探破解口令

- 共享信道
 - 广播型以太网
- 协议不加密
 - 口令明文传输



以太网的工作原理

✧载波侦听/冲突检测(CSMA/CD, carrier sense multiple access with collision detection)技术

- ✓载波侦听：是指在网络中的每个站点都具有同等的权利，在传输自己的数据时，首先监听信道是否空闲
 - 如果空闲，就传输自己的数据
 - 如果信道被占用，就等待信道空闲
- ✓而冲突检测则是为了防止发生两个站点同时监测到网络没有被使用时而产生冲突



以太网卡的工作模式

✧ 网卡的**MAC地址(48位)**

- ✓ 通过**ARP**来解析**MAC**与**IP**地址的转换
- ✓ 用**ipconfig/ifconfig**可以查看**MAC**地址

✧ 正常情况下，网卡应该只接收这样的包

- ✓ **MAC**地址与自己相匹配的数据帧
- ✓ 广播包

✧ 网卡完成收发数据包的工作，两种接收模式

- ✓ 混杂模式：不管数据帧中的目的地址是否与自己的地址匹配，都接收下来
- ✓ 非混杂模式：只接收目的地址相匹配的数据帧，以及广播数据包(和组播数据包)

✧ 为了监听网络上的流量，必须设置为混杂模式



共享网络和交换网络

◇共享式网络

- ✓ 通过网络的所有数据包发往每一个主机
- ✓ 最常见的是通过**HUB**连接起来的子网

◇交换式网络

- ✓ 通过交换机连接网络
- ✓ 由交换机构造一个“**MAC**地址-端口”映射表
- ✓ 发送包的时候，只发到特定的端口上



应用程序抓包的技术

✧ **UNIX**系统提供了标准的**API**支持

- ✓ **Packet socket**

- ✓ **BPF**

✧ **Windows**平台上通过驱动程序来获取数据包

- ✓ 驱动程序

- ✓ **WinPcap**



Packet socket

✧ 设置混杂(**promiscuous**)模式

- ✓ 用**ioctl()**函数可以设置

✧ 不同的**UNIX**或者**Linux**版本可能会有不同的函数调用，本质上

- ✓ 打开一个**socket**(或者通过**open**打开一个设备)

- ✓ 通过**ioctl()**或者**setsockopt()**设置为混杂模式

- ✓ 示例



BPF (Berkeley Packet Filter)

✧ BSD抓包法

- ✓ **BPF**是一个核心态的组件，也是一个过滤器，
- ✓ **Network Tap**，接收所有的数据包
- ✓ **Kernel Buffer**，保存过滤器送过来的数据包
- ✓ **User buffer**，用户态上的数据包缓冲区

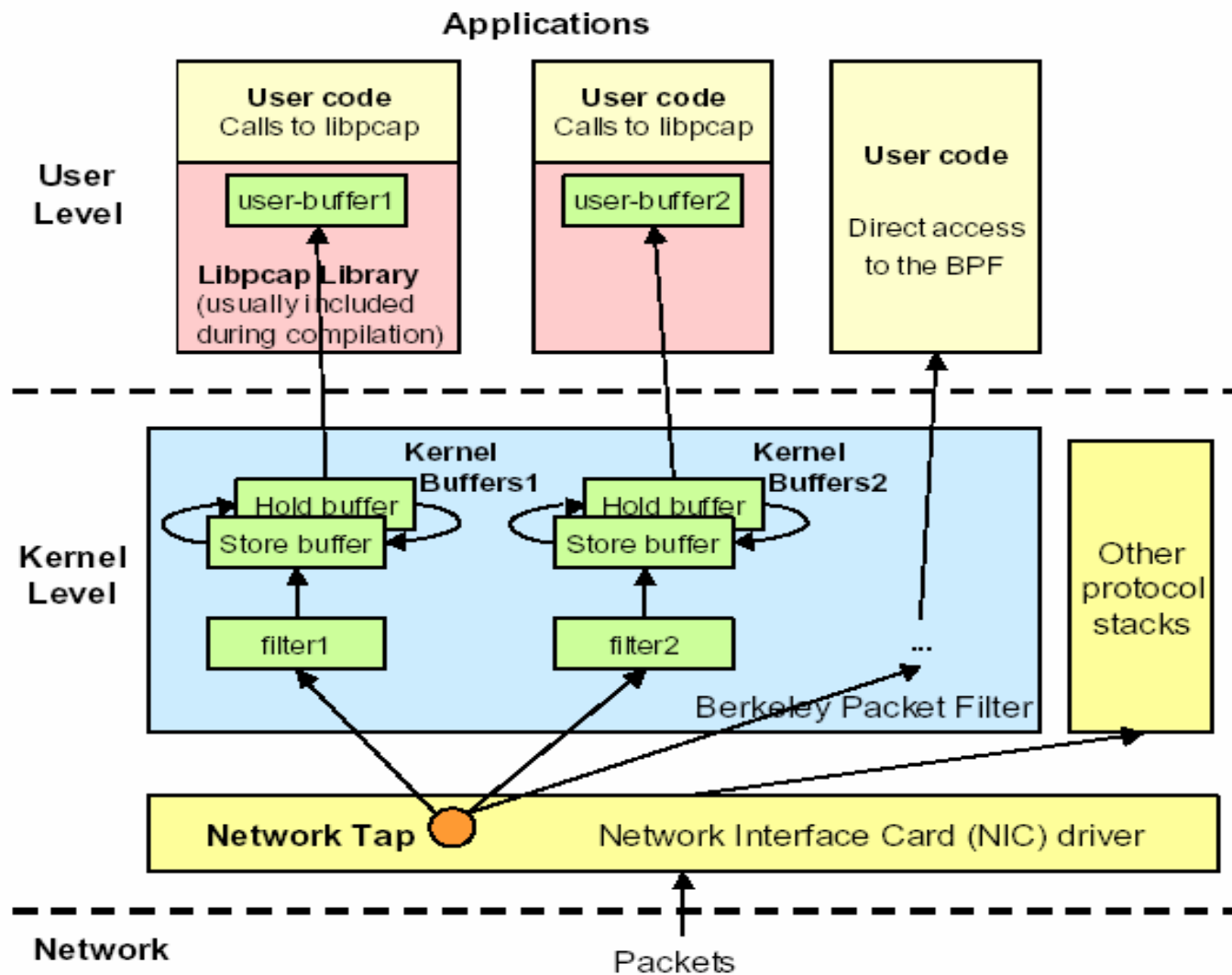
✧ BPF是一种比较理想的抓包方案

- ✓ 在核心态，所以效率比较高，
- ✓ 但是，只有少数**OS**支持(主要是一些**BSD**操作系统)

✧ Libpcap(一个抓包工具库)支持BPF

- ✓ **Libpcap**是用户态的一个抓包工具
- ✓ **Libpcap**几乎是系统无关的

BPF和libpcap





关于libpcap

- ✧ 用户态下的数据包截获
- ✧ 系统独立的**API**接口，**C**语言接口
- ✧ 广泛应用于：
 - ✓ 网络数据收集
 - ✓ 安全监控
 - ✓ 网络调试
- ✧ 支持过滤机制，**BPF**
- ✧ <http://www.tcpdump.org/pcap.htm>



Windows平台下的抓包技术

- ✧ 内核本身没有提供标准的接口
- ✧ 通过增加一个驱动程序或者网络组件来访问内核网卡驱动提供的数据包
 - ✓ 在**Windows**不同操作系统平台下有所不同
- ✧ 不同**sniffer**采用的技术不同
 - ✓ **WinPcap**是一个重要的抓包工具，它是**libpcap**的**Windows**版本

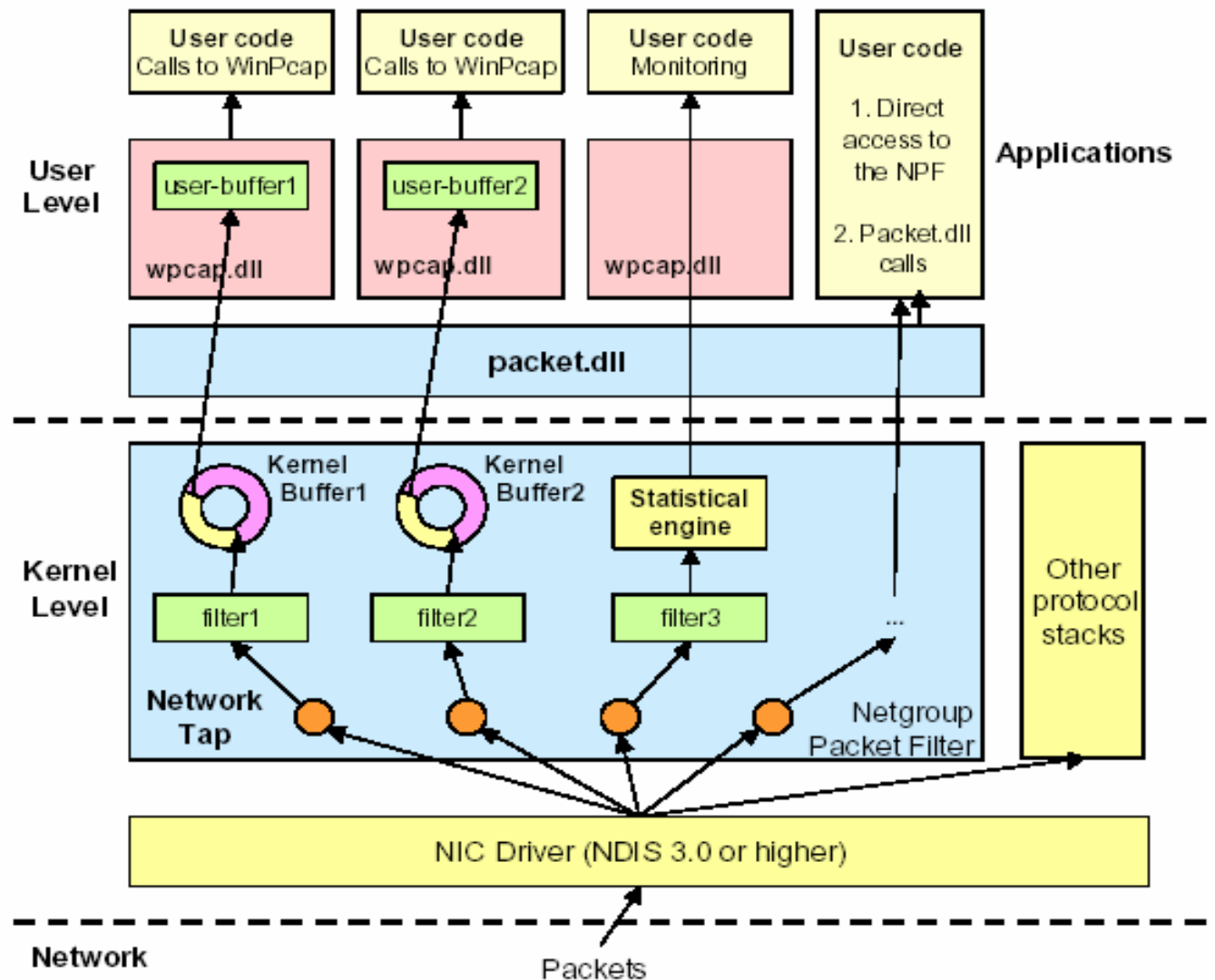


WinPcap

✧ WinPcap包括三个部分

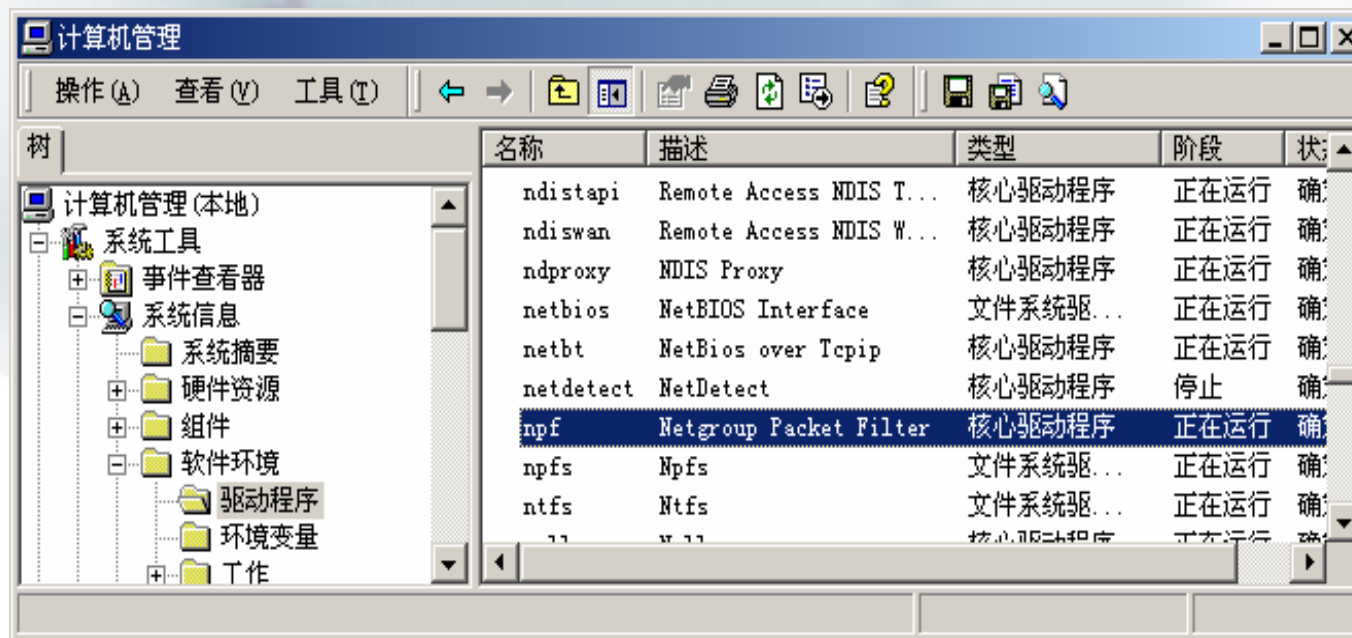
- ✓ 第一个模块**NPF(Netgroup Packet Filter)**，是一个虚拟设备驱动程序文件。它的功能是过滤数据包，并把这些数据包原封不动地传给用户态模块，这个过程中包括了一些操作系统特有的代码
- ✓ 第二个模块**packet.dll**为win32平台提供了一个公共的接口。不同版本的**Windows**系统都有自己的内核模块和用户层模块。**Packet.dll**用于解决这些不同。调用**Packet.dll**的程序可以运行在不同版本的**Windows**平台上，而无需重新编译
- ✓ 第三个模块 **Wpcap.dll**是不依赖于操作系统的。它提供了更加高层、抽象的函数。

WinPcap and NPF



Windows的网络结构

- ✧ **NDIS(Network Driver Interface Specification, 网络驱动接口规范)**描述了网络驱动与底层网卡之间的接口规范, 以及它与上层协议之间的规范
- ✧ **NPF**作为一个核心驱动程序而提供的





WinPcap的优势

- ✧ 提供了一套标准的抓包接口
 - ✓ 与**libpcap**兼容，可使得原来许多**UNIX**平台下的网络分析工具快速移植过来
 - ✓ 便于开发各种网络分析工具
- ✧ 除了与**libpcap**兼容的功能之外，还有
 - ✓ 充分考虑了各种性能和效率的优化，包括对于**NPF**内核层次上的过滤器支持
 - ✓ 支持内核态的统计模式
 - ✓ 提供了发送数据包的能力
- ✧ **<http://winpcap.polito.it/>**



发送数据包——Libnet

✧ 利用**Libnet**构造数据包并发送出去

✧ 关于**Libnet**

- ✓ 支持多种操作系统平台
- ✓ 提供了**50**多个**API**函数，功能涵盖
 - 内存管理(分配和释放)函数
 - 地址解析函数
 - 各种协议类型的数据包构造函数
 - 数据包发送函数(**IP**层和链路层)
 - 一些辅助函数，如产生随机数、错误报告等



使用Libnet的基本过程

- ✧数据包内存初始化
- ✧网络接口初始化
- ✧构造所需的数据包
- ✧计算数据包的校验和
- ✧发送数据包
- ✧关闭网络接口
- ✧释放数据包内存

```
libnet_init_packet(...);  
libnet_open_raw_sock(...);  
libnet_build_ip(...);  
libnet_build_tcp(...);  
libnet_do_checksum(...);  
libnet_write_ip(...);  
libnet_close_raw_sock(...);  
libnet_destroy_packet(...);
```




Sniffer的反措施

- ✧合理的网络分段，在网络中使用网桥和交换机；相互信任的主机处于同一网段
- ✧使用加密技术传送敏感数据，如**SSH**
- ✧为了防止**ARP**欺骗，使用永久的**ARP**缓存条目
- ✧如何检测处于混杂模式的节点？

检测处于混杂模式的节点

- ✧ 网卡和操作系统对于是否处于混杂模式会有一些不同的行为，利用这些特征可以判断机器是否运行在混杂模式下。
- ✧ 检测手段
 - **Linux**内核的特性：正常情况下，只处理本机**MAC**地址或者以太广播地址的包。在混杂模式下，许多版本的**Linux**内核只检查数据包中的**IP**地址以确定是否送到**IP**堆栈。因此，可以构造无效以太地址而**IP**地址有效的**ICMP ECHO**请求，看机器是否返回应答包(混杂模式)，或忽略(非混杂模式)。
 - **Windows 9x/NT**：在混杂模式下，检查一个包是否为以太广播包时，只看**MAC**地址前八位是否为**0xff**。



检测处于混杂模式的节点

✓ 根据网络和主机的性能

- 根据响应时间：向本地网络发送大量的伪造数据包，然后，看目标主机的响应时间，首先要测得一个响应时间基准和平均值

✧ **L0pht**的**AntiSniff**产品，参考它的技术文档

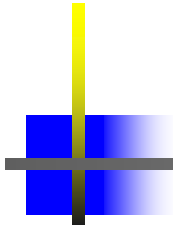


一些sniffer工具

- ✧ buttsniffer, netmon, netxray, sniffer pro, ethereal, analyzer, windump(tcpdump的windows版本)。
- ✧ dsniff, linux_sniffer, snort, tcpdump, sniffit, ethereal

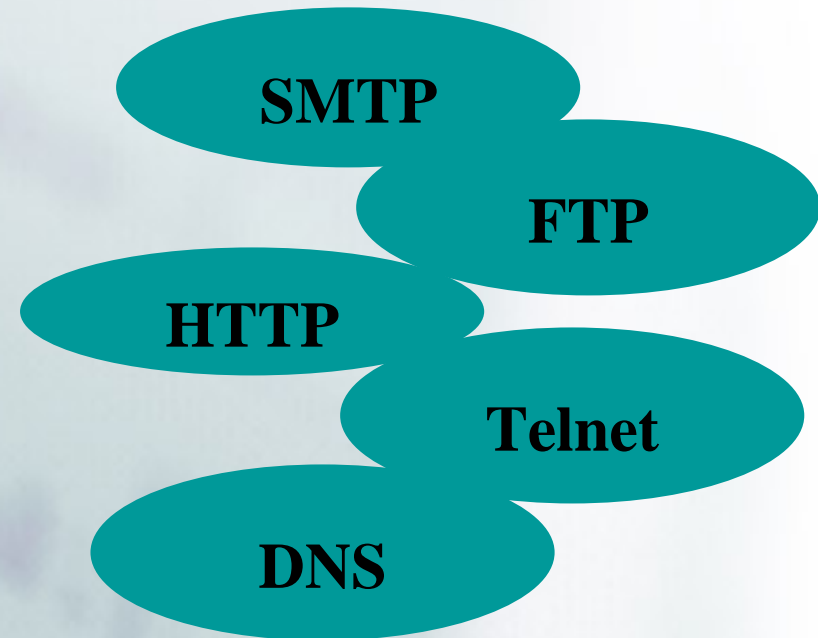
Windump

```
c:\D:\Tools\Network security\windump.exe
D:\Tools\Network security\windump.exe: listening on \Device\NPF_{A4B1317A-2CBC-4
751-AB1B-0E54FA8E6D33}
23:21:58.140272 IP YINGL.2425 > 202.102.156.130.80: P 2413971740:2413972080(340)
  ack 216974151 win 16161 <DF>
23:21:58.214676 IP CAROL.3644 > infosec.pku.edu.cn.8080: F 925252097:925252097<0
> ack 1745879597 win 64240 <DF>
23:21:58.215060 IP infosec.pku.edu.cn.8080 > CAROL.3644: . ack 1 win 33580 <DF>
23:21:58.215106 IP infosec.pku.edu.cn.8080 > CAROL.3644: F 1:1<0> ack 1 win 3358
0 <DF>
23:21:58.215232 IP CAROL.3644 > infosec.pku.edu.cn.8080: . ack 2 win 64240 <DF>
23:21:58.254776 IP CAROL.3645 > infosec.pku.edu.cn.8080: S 925340782:925340782<0
> win 64240 <mss 1460,nop,nop,sackOK> <DF>
23:21:58.255299 IP infosec.pku.edu.cn.8080 > CAROL.3645: S 4051635685:4051635685
<0> ack 925340783 win 32768 <mss 1460> <DF>
23:21:58.255422 IP CAROL.3645 > infosec.pku.edu.cn.8080: . ack 1 win 64240 <DF>
23:21:58.286402 IP CAROL.3645 > infosec.pku.edu.cn.8080: P 1:248<247> ack 1 win
64240 <DF>
23:21:58.288350 IP infosec.pku.edu.cn.8080 > CAROL.3645: . 1:1461<1460> ack 248
win 33580 <DF>
23:21:58.288439 IP infosec.pku.edu.cn.8080 > CAROL.3645: P 1461:2314<853> ack 24
8 win 33580 <DF>
23:21:58.288498 IP CAROL.3645 > infosec.pku.edu.cn.8080: . ack 2314 win 64240 <D
F>
23:21:58.444783 IP CAROL.3645 > infosec.pku.edu.cn.8080: F 248:248<0> ack 2314 w
in 64240 <DF>
23:21:58.445306 IP infosec.pku.edu.cn.8080 > CAROL.3645: . ack 249 win 33580 <DF
>
23:21:58.445374 IP infosec.pku.edu.cn.8080 > CAROL.3645: F 2314:2314<0> ack 249
win 33580 <DF>
23:21:58.445473 IP CAROL.3645 > infosec.pku.edu.cn.8080: . ack 2315 win 64240 <D
```

- 
- 一个使用**sniffer**分析网页监控的[例子](#)

应用层介绍

- 简单邮件传输协议 (SMTP)
- 文件传输协议 (FTP)
- 超文本传输协议 (HTTP)
- 远程连接服务标准协议 (Telnet)
- 域名系统 (DNS)



DNS报文



应用层的安全防护

- 实施强大的基于用户的身份认证
- 实施数据加密，访问控制的理想位置
- 加强数据的备份和恢复措施
- 对资源的有效性进行控制，资源包括各种数据和服务



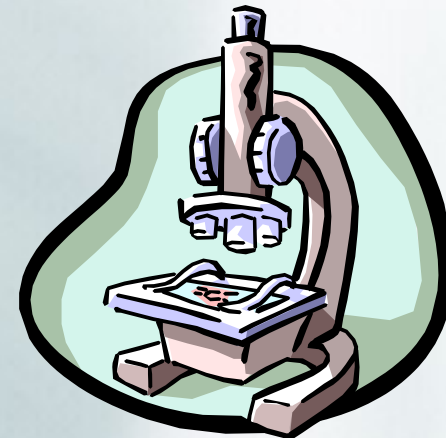


文件传输协议 (FTP)

- 主要的威胁
 - ☐ 塞满整个硬盘空间
 - ☐ 用户名及密码明文传输
- 防范措施
 - ☐ 仅允许匿名登录
 - ☐ **FTP**服务放在其他分区中

Telnet

- Telnet本身的缺陷是
 - 没有口令保护
 - 传输数据没有认证过程
 - 没有完整性检查
 - 传送的数据都没有加密



SMTP

•Simple Mail Transfer Protocol(SMTP)

- 用TCP进行的邮件交换是由报文传送代理MTA(Message Transfer Agent)完成的。两个MTA之间用NVT ASCII进行通信，客户向服务器发出命令，服务器用数字应答码和可选的字符串进行响应





简单邮件传输协议

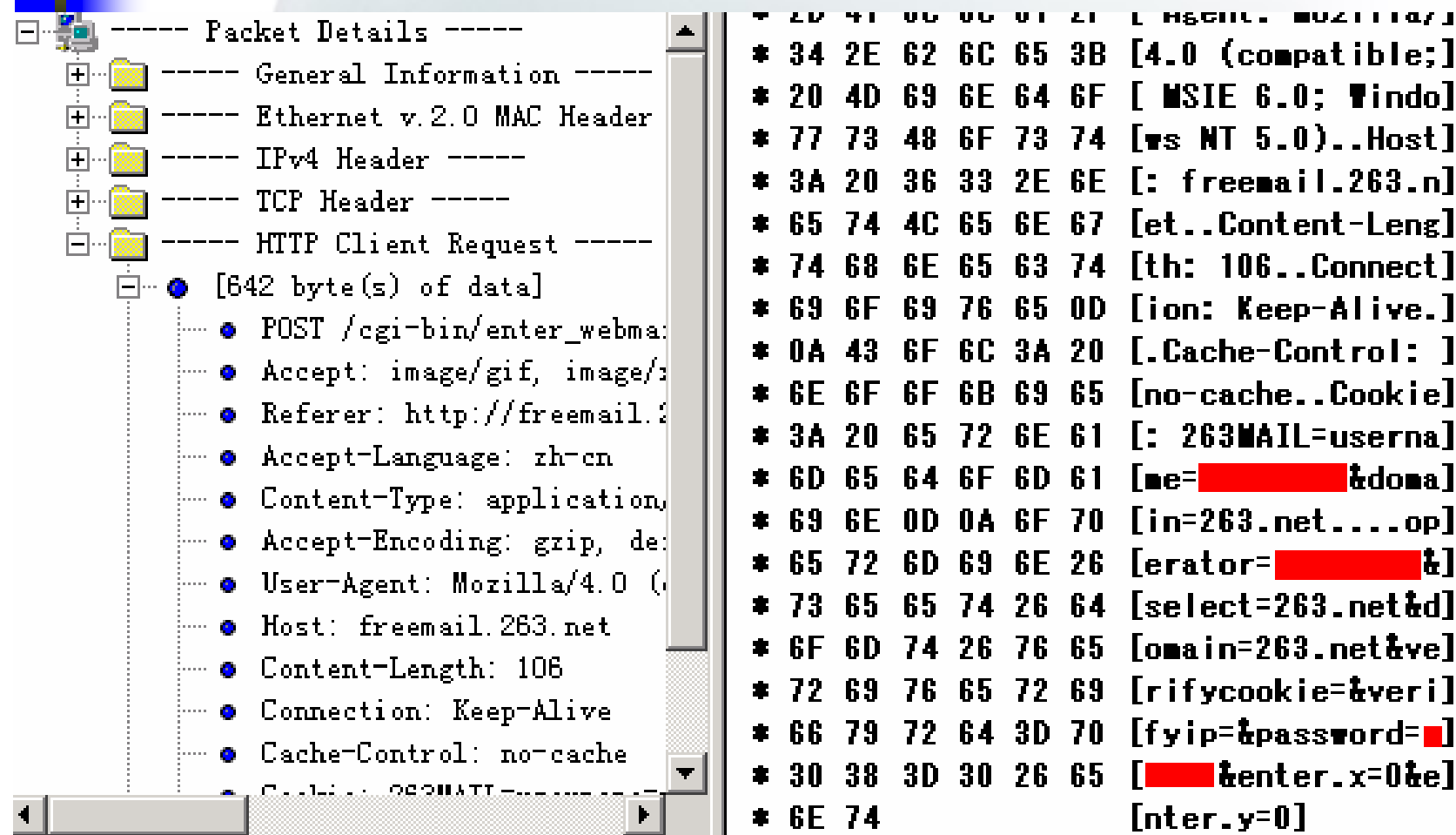
❑ 主要威胁:

- mail flood
- sending viruses and Trojan
- username leak
- spam

❑ 防护措施

- Anti-viruses gateway
- 使用安全的SMTP 软件
- 培训 email 用户
- anti-spam

认证例子：263的邮件登录



The image shows a Wireshark packet capture of an HTTP POST request. The left pane displays the packet details tree, and the right pane shows the raw hex and ASCII data.

Packet Details:

- General Information
- Ethernet v.2.0 MAC Header
- IPv4 Header
- TCP Header
- HTTP Client Request
- [642 byte(s) of data]
 - POST /cgi-bin/enter_webmail.cgi
 - Accept: image/gif, image/jpeg
 - Referer: http://freemail.263.net
 - Accept-Language: zh-cn
 - Content-Type: application/x-www-form-urlencoded
 - Accept-Encoding: gzip, deflate
 - User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.0)..Host
 - Host: freemail.263.net
 - Content-Length: 106
 - Connection: Keep-Alive
 - Cache-Control: no-cache

Raw Data (Hex and ASCII):

```
* 20 4D 69 6E 64 6F [ MSIE 6.0; Windo
* 77 73 48 6F 73 74 [ws NT 5.0)..Host]
* 3A 20 36 33 2E 6E [: freemail.263.n
* 65 74 4C 65 6E 67 [et..Content-Leng
* 74 68 6E 65 63 74 [th: 106..Connect
* 69 6F 69 76 65 0D [ion: Keep-Alive.]
* 0A 43 6F 6C 3A 20 [..Cache-Control: ]
* 6E 6F 6F 6B 69 65 [no-cache..Cookie]
* 3A 20 65 72 6E 61 [: 263MAIL=userna
* 6D 65 64 6F 6D 61 [me= [REDACTED]&doma
* 69 6E 0D 0A 6F 70 [in=263.net....op
* 65 72 6D 69 6E 26 [erator= [REDACTED]&
* 73 65 65 74 26 64 [select=263.net&d
* 6F 6D 74 26 76 65 [omain=263.net&ve
* 72 69 76 65 72 69 [rifycookie=&veri
* 66 79 72 64 3D 70 [fyip=&password=[REDACTED]
* 30 38 3D 30 26 65 [ [REDACTED]&enter.x=0&e
* 6E 74 [nter.y=0]
```

认证例子：sina的邮件登录

Packet Details

General Information

Ethernet v.2.0 MAC Head

IPv4 Header

TCP Header

HTTP Client Request

[849 byte(s) of data]

107

mynum=1&user=&pass=&u=s

```

* 33 64 72 3A 30 33 [3d6310e7c,curr:03]
* 2C 70 30 35 2C 65 [,pos:08,sal:05,e]
* 64 75 2C 6D 61 72 [du:03,sta:02,mar]
* 3A 30 3A 33 32 3B [:0,gen:M,age:32;]
* 20 53 20 53 49 44 [ SINA_USER=; SID]
* 3D 3B 6C 6F 67 69 [=; userinfo_logi]
* 6E 74 34 32 39 37 [ntime=1016174297]
* 3B 20 68 61 6E 6E [; userinfo_chann]
* 65 6C 72 69 6E 66 [el=mail; userinf]
* 6F 5F 3D 31 36 32 [o_remoteaddr=162]
* 2E 31 20 53 4D 3D [.105. ; SM=]
* 53 69 6D 79 6E 75 [SinaMail....mynu]
* 6D 3D 73 73 3D 26 [m=1&user=&pass=&]
* 75 3D 70 73 77 3D [u= &psw=]
* 25 33 41 25 [ &l=http%3A%]
* 32 46 6E 61 2E 63 [2F%2Fmail.sina.c]
* 6F 6D 62 69 6E 25 [om.cn%2Fcgi-bin%]
* 32 46 72 6F 64 75 [2Fmail.cgi&produ]
* 63 74 [ct=mail]

```



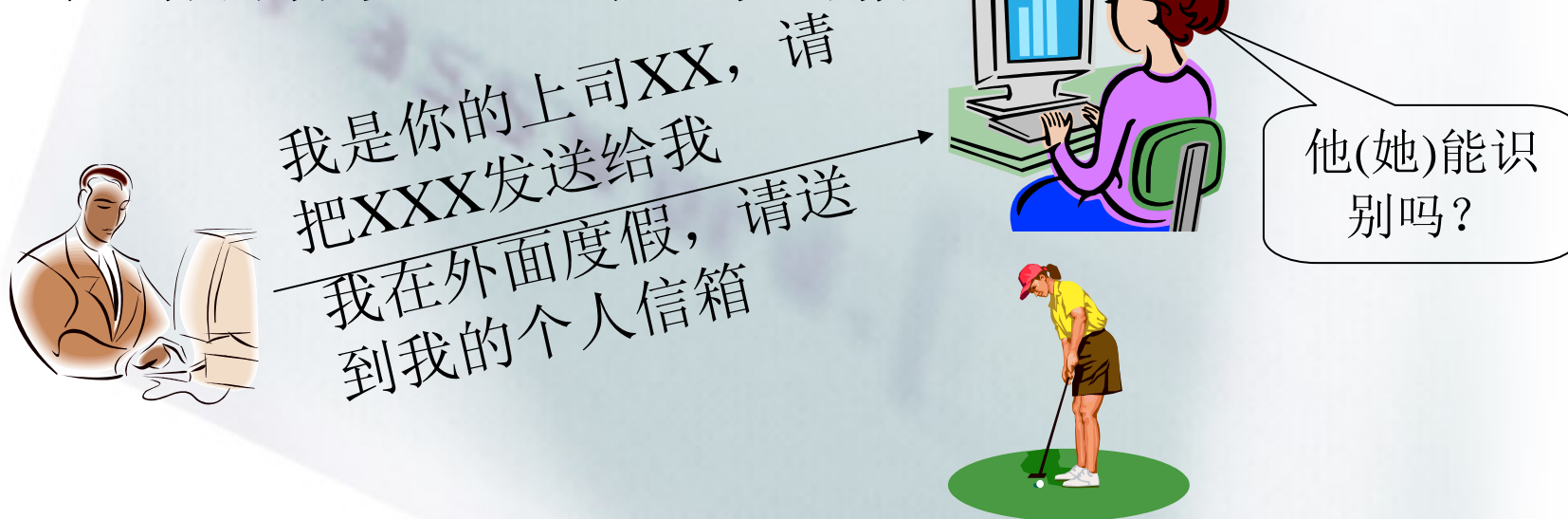
电子邮件欺骗

- 为什么要进行电子邮件欺骗
 - ☐ 隐藏发件人的身份，例如匿名信
 - ☐ 骗取敏感信息
- 电子邮件欺骗的方法
 - ☐ 与现实邮局进行比较
 - ☐ 基本邮件协议没有认证机制
 - ☐ 发信可以要求认证

电子邮件欺骗

- 使用类似的邮件地址

□ 发信人使用被假冒者的名字注册一个账号，然后给目标发送一封正常的信



解决：使用数字签名

电子邮件欺骗

- 直接连到**SMTP**服务器上发信
 - **telnet**连到**SMTP**服务器的**25**端口，然后发送命令，常用的命令为：
 - **Helo**、**Mail from**、**Rcpt to**、**Data**、**Quit**
 - 发件人、电子邮件地址等信息，都是一些字段的值：
From: MAIL FROM: RCPT TO:

电子邮件保护措施

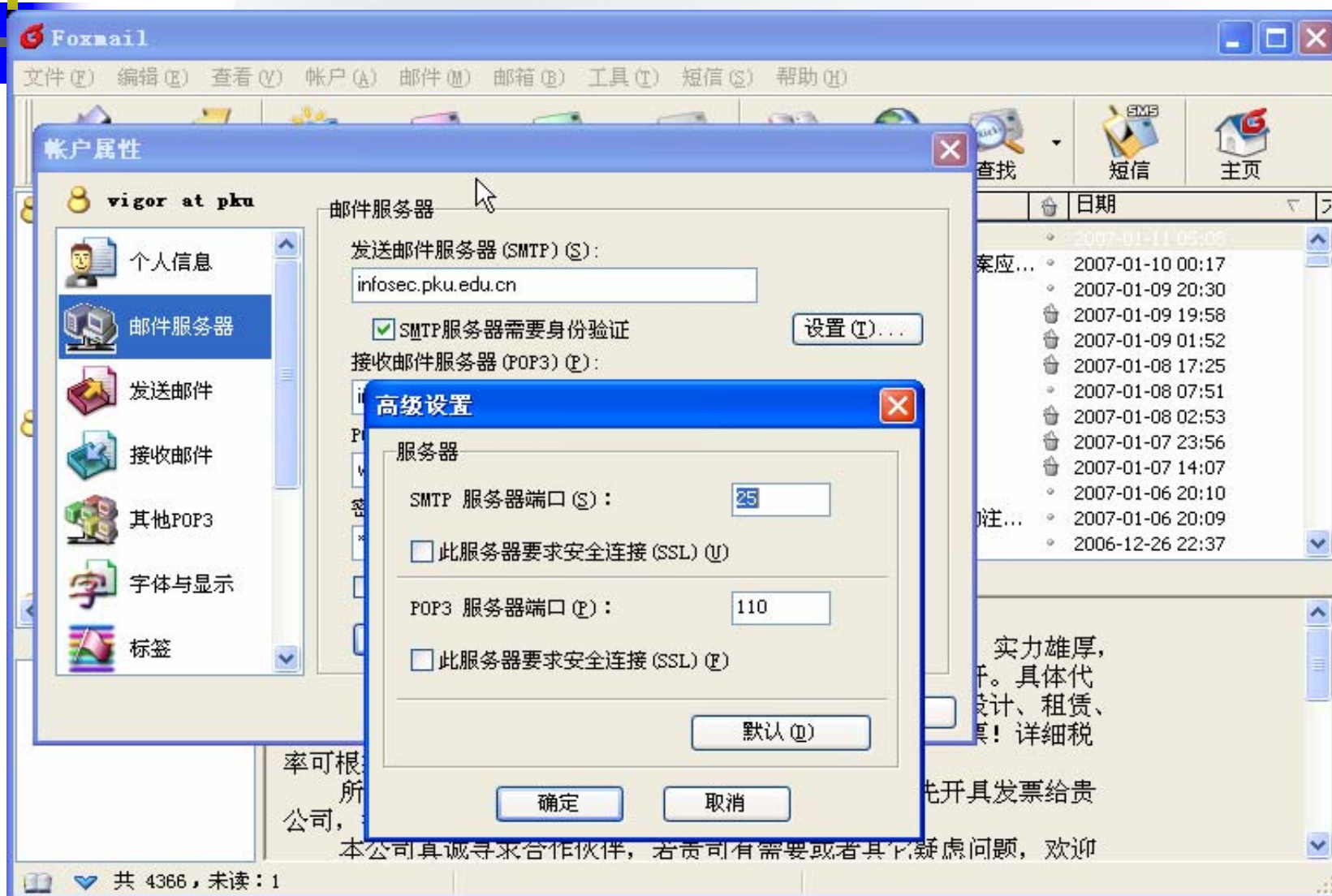
- 查看完整的电子邮件头部信息
 - 多数邮件系统允许用户查看邮件从源地址到目的地址经过的所有主机
- 邮件服务器的验证
 - ❖ **Smtp**服务器验证发送者的身份。
 - ❖ 验证接收方的域名与邮件服务器的域名是否相同（如果是自己的用户，可以发往别处，如果不是，则不**relay**）。
 - ❖ 有的也验证发送者的域名是否有效，通过反向**DNS**解析。
- 注意一个内部用户冒充另一个内部用户发送邮件



客户端安全

- 使用**ssl**
- 是用**PGP**
- 追查**IP**

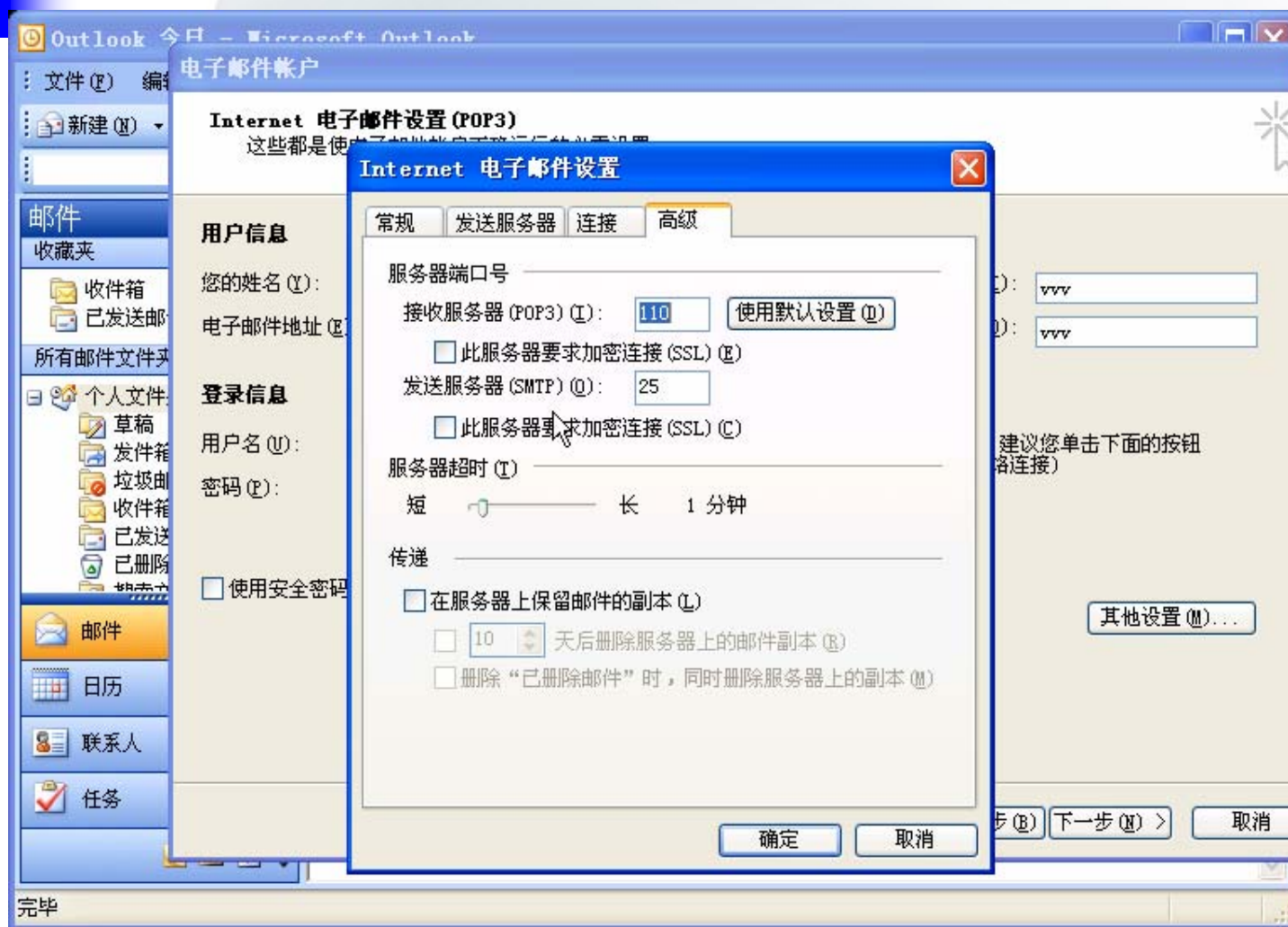
foxmail



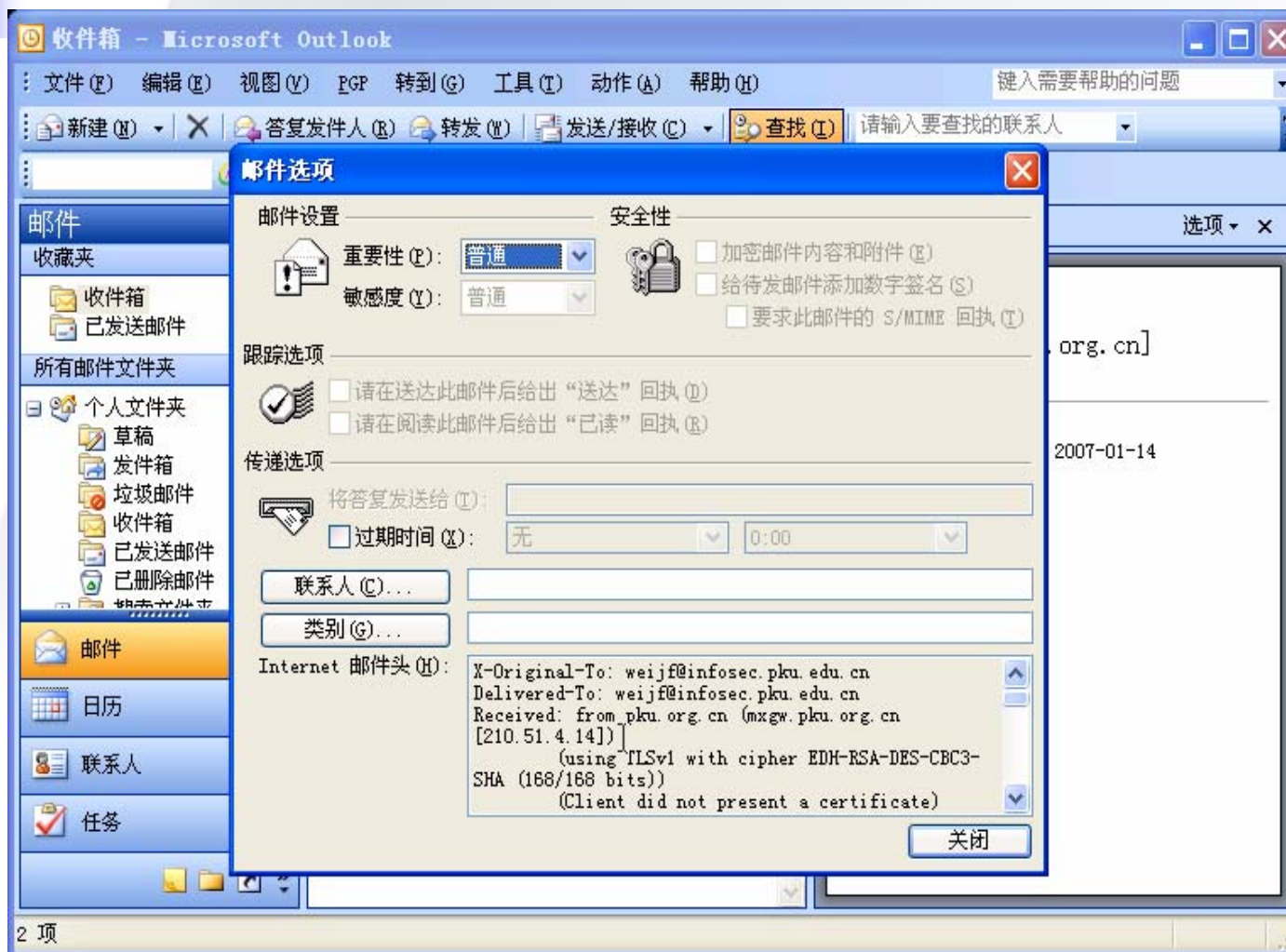
ctrl-l



outlook



邮件选项

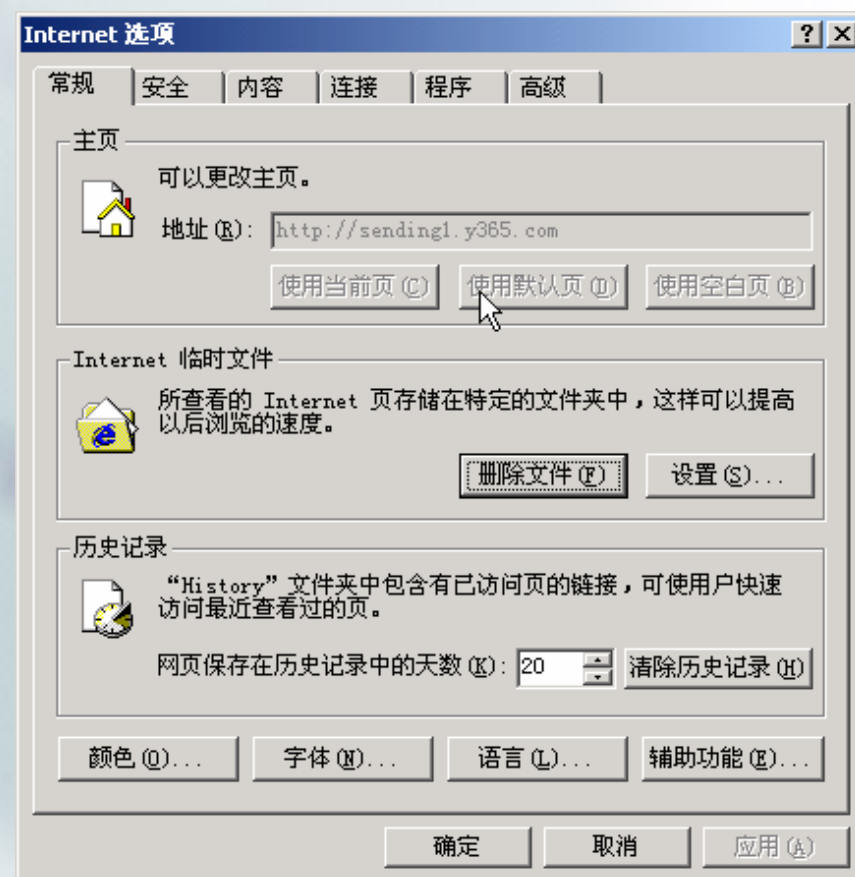
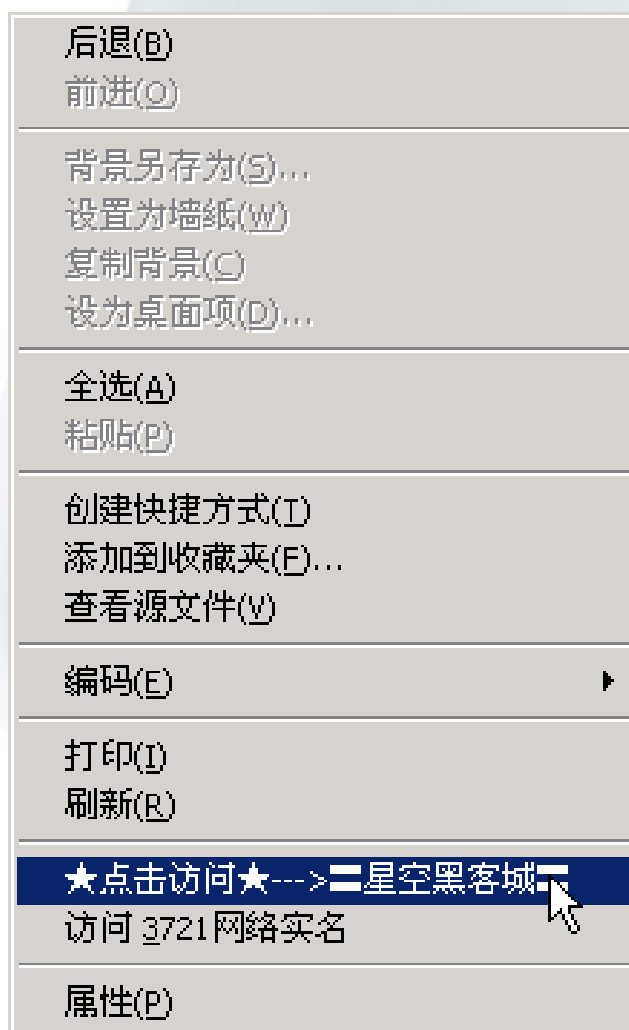




超文本传输协议HTTP

- 主要威胁
 - ☐ 恶意的**active X or Java applet**
 - ☐ **Extended applications(Java,CGI,ASP..)**
- 防范措施
 - ☐ **Secure HTTP 和SSL**
 - ☐ 安全级别设置
 - ☐ **ASP的安全**

浏览器骚扰





针对浏览器骚扰的防范措施

- 使用IE Security等工具保护浏览器。
- 因为修改注册表设置都是用的JavaScript脚本语言，所以只需禁用它即可。但这种脚本语言应用广泛，所以建议在IE的设置中将脚本设为“提示”。
- 对使用Win2000的用户，只需在“控制面板”→“管理工具”→“服务”中禁用Remote Registry Service服务，这样，网页也无法来修改用户的注册表了，但可能会影响其他应用。
- 很多杀毒软件具备Script Blocking功能，将通过IE修改注册表的代码定义为Trojan. Offensive予以拦截。
- 使用**IE6.0**，可以防止网页对注册表的恶意修改。



Web欺骗

- 何谓Web欺骗？
 - ☐ 创建一个Web站点的映像，使得用户的连接被接入到Hacker的服务器，达到欺骗网络用户的目的。
- 为什么会受到Web欺骗？
 - ☐ 监控网络用户；
 - ☐ 窃取帐户信息；
 - ☐ 损坏网站形象；
 - ☐ 失效SSL连接（使用过期的证书）。



Web欺骗

- **Web**是建立在应用层上的服务，直接面向Internet用户
- **Web欺骗的根源**
 - 由于**Internet**的开放性，任何人都可以建立自己的**Web**站点
 - **Web**站点名字(**DNS**域名)可以自由注册，按先后顺序
 - 并不是每个用户都清楚**Web**的运行规则
- **Web欺骗的形式**
 - 使用相似的域名
 - **Cross-Site Scripting**



Web欺骗

- 使用相似的域名
 - ❑ 注册一个与目标公司或者组织相似的域名，然后建立一个欺骗网站，骗取该公司用户的信任，以得到这些用户的信息
 - ❑ 例如：使用**www.whitehouse.com**来混淆**www.whitehouse.gov**

Web欺骗

- **Cross-Site Scripting**（跨站脚本攻击）

- ❑ 交互网站或者服务提供网站可能存在这种漏洞

- ❑ 攻击产生的根源

- 网站提供交互时可使用**HTML**代码，用户可上传恶意脚本代码

- ❑ 攻击方式（以一个论坛网站为例）

- 攻击者在签名档里添加一个链接，内容是把用户的**cookie**发送到攻击者的邮箱中去

- 攻击者发送一个消息，吸引其他用户来看

- 用户浏览攻击者留下的信息，浏览器自动执行攻击者的脚本

- 攻击者接收到用户的**cookie**，然后可以伪装成用户进行其他恶意行为

Web欺骗

- 防止**Web**欺骗—使用相似的域名
 - ☐ 注意观察**URL**地址栏的变化
 - ☐ 不要信任不可靠的**URL**信息
 - ☐ 你是不是相信自己?
 - 许多人根据自己的猜测去在浏览器地址栏中写下要访问的网址
- 防止**Web**欺骗—**Cross-Site Scripting**
 - ☐ 网站不允许使用**HTML**语言，不允许使用脚本程序
 - 过于严格没有必要
 - 对特殊字符的过滤，注意黑客的对策:采用另一种字符表达方式来逃避过滤，例如**%20**

IIS的unicode漏洞

- 如果存在该漏洞，使用浏览器，就可进行攻击，输入：
http://x.x.x.x/scripts/..%c0%af../winnt/system32/cmd.exe?/c+dir
就可列出服务器硬盘目录。
相当于执行了**cmd /c dir**
当然也可运行**net**命令。
- 不同语言版本中，不同的代码，如上面是英文**win2000**,对于中文**win2000**，用**%c1%1c**，起到/的作用。



IIS+ASP: 问题出奇的多

- **idc&ida**漏洞
- **.httr**漏洞
- **NT Site Server Adsamples**漏洞
- **.printer**漏洞
- **Unicode**漏洞
- **Webdav**漏洞
- **Null.htw**漏洞
- **IIS HACK**漏洞
- **webhits.dll & .htw**漏洞
- **ISM.DLL**漏洞
- ...



对IIS服务进行安全配置

- **(1)**将**IIS**安装在和系统不同的分区，使得**IIS**漏洞引起的安全问题尽可能不要影响破坏，如果已安装**IIS**，建议卸载重装。
- **(2)**更改**Web**服务主目录。右键单击“默认**Web**站点→属性→主目录→本地路径”，将“本地路径”指向其他目录。默认的根本目录在**inetpub\wwwroot**。将这个默认目录删掉。

对IIS服务进行安全配置

- **(3)**将默认生成的众所周知的目录删掉，如 **IIShelp**, **IISAdmin**, **IISSamples**, **_vti_bin**, **Scripts**, **MSADC**等等，以免给攻击者留下机会。
- **(4)** 删除不必要的**IIS**扩展名映射。右键单击“默认**Web**站点”，选→属性→主目录→应用程序设置-→配置”，打开应用程序窗口，去掉不必要的应用程序映射。如不用其他映射，只保留**.asp**即可。或依情况保留**shtm,shtml,stm**这几种格式。



对IIS服务进行安全配置

- **(5)** 站点→属性→**web**站点→启用日志记录，建议日志路径设在和**web**主目录不同的分区，以增加攻击者读取日志的难度。日志文件的权限：**administrator**和**system**为完全控制，**everyone**为只读。



对IIS服务进行安全配置

- **(6) web**主目录中的文件和目录，通过右键点“属性”设置权限，静态文件允许读，不许写，**ASP**脚本文件、**exe**程序允许执行，但拒绝读、写；所有文件和目录要将**everyone**用户组的权限设为只读。



对IIS服务进行安全配置

- **(7)**可以考虑在站点→属性→**web**站点→**TCP**端口中，修改端口，不使用众所周知的端口**80**。
- **(8)**可以考虑在站点→属性→**web**站点→目录安全性中，在**IP**地址和域名限制中，允许或拒绝特定**IP**发来的**web**请求。



对IIS服务进行安全配置

- 如果有机密性需求，可以考虑使用**SSL**加密。
- 图省事的话，可以使用微软提供的**IIS lockdown**工具。



域名系统（DNS）

- **Domain Name System(DNS)**

- ❑ **主要威胁**

- Obtain zone files

- DNS spoof/poison

- ❑ **防范措施**

- Firewall blocks out zone transfer

- Accept zone transfer only from specific hosts

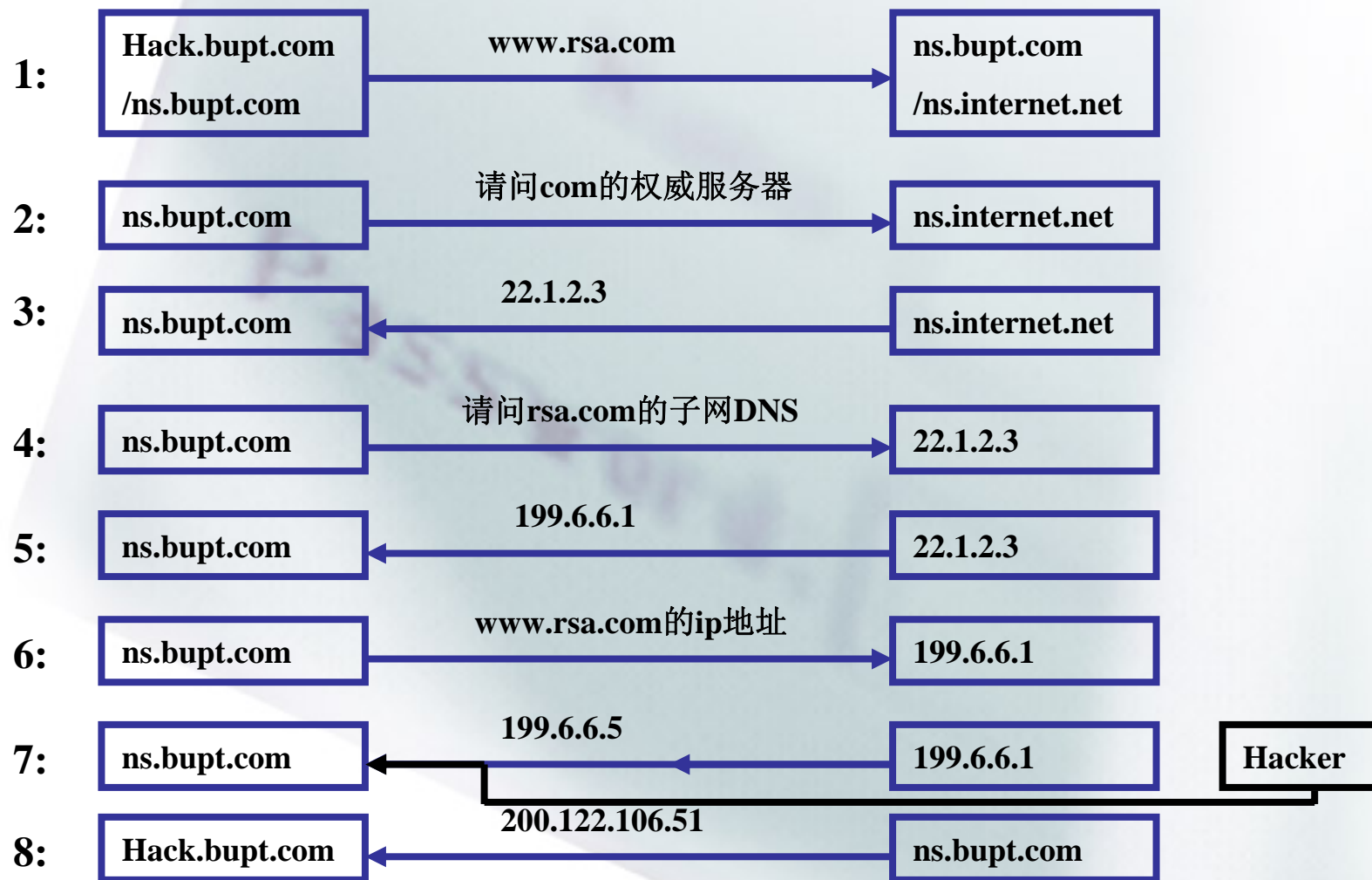
- ❑ 背景知识：DNS的zone file 有变动后, name server 管理者就增加一个序列号, slave会对比序列号以决定是否要再 copy 一次, 即 zone transfer。



DNS欺骗

- 攻击复杂，使用简单
- RSA Security网站曾被成功攻击
- DNS欺骗原理
 - ❑ IP与域名的关系
 - ❑ DNS的域名解析
 - ❑ Rand Port to DNS' s 53

DNS欺骗原理





应用层协议安全问题小结

- 应用层协议本身存在的主要问题是信息明文传输，包括如**FTP**、**Telnet**登录验证帐号和密码都是明文，攻击者可以通过网络嗅探获取有价值信息，以备下一步攻击之用，此外应用层协议的许多威胁来自于低层协议的安全性问题。



应用层安全措施

- 安全协议：SSH，S-HTTP, SET
- 应用层的安全服务实际上是最灵活的处理单个文件安全性的手段，可以实施细粒度的安全控制
- 可能的方法：对每个应用(及应用协议)分别进行修改；实施强大的基于用户的身份认证；实施数据加密；访问控制；数据的备份和恢复措施；对资源的有效性进行控制。
- 应用层安全的解决目前往往依赖于网络层、操作系统、数据库的安全

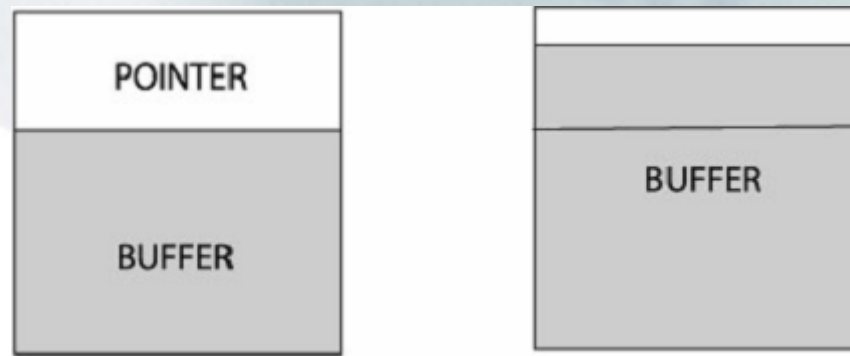
TCP/IP各层次与安全服务

安全服务	TCP/IP 协议层			
	网络接口	互联网层	传输层	应用层
对等实体鉴别	-	Y	Y	Y
数据源鉴别	-	Y	Y	Y
访问控制服务	-	Y	Y	Y
连接保密性	Y	Y	Y	Y
无连接保密性	Y	Y	Y	Y
选择域保密性	-	-	-	Y
流量保密性	Y	Y	-	Y
有恢复功能的连接完整性	-	-	Y	Y
无恢复功能的连接完整性	-	Y	Y	Y
选择域连接完整性	-	-	-	Y
无连接完整性	-	Y	Y	Y
选择域非连接完整性	-	-	-	Y
源发方不可否认	-	-	-	Y
接收方不可否认	-	-	-	Y

缓冲区溢出攻击

十年来最大的安全问题

通过向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。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。





缓冲区溢出历史

- ❑ 1988年的Morris蠕虫病毒，放倒了6000多台机器：利用UNIX服务finger中的缓冲区溢出漏洞来获得访问权限，得到一个shell
- ❑ 1996年前后，开始出现大量的Buffer Overflow攻击，因此引起人们的广泛关注
- ❑ 源码开放的操作系统首当其冲
- ❑ 随后，Windows系统下的Buffer Overflows也相继被发掘出来

缓冲区溢出技术基础

- 进程的内存空间示意图

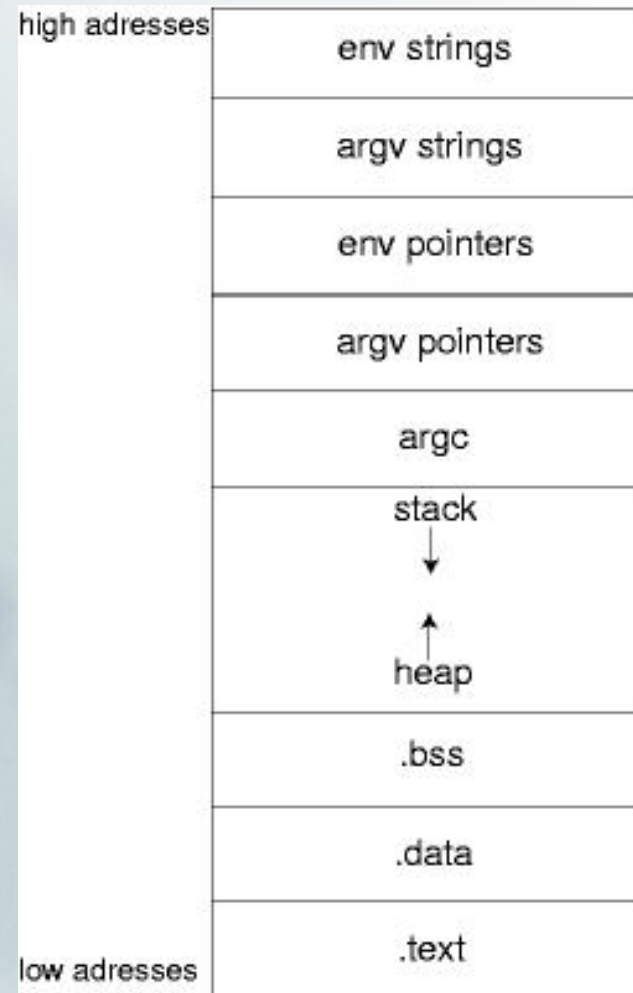
☐ Stack

☐ Heap

☐ Bss

☐ Data

☐ Text



缓冲区溢出技术基础

```
int func(int a, int b){
    int ret= a + b;
    return ret;
}
int main(int argc, char* argv[ ])
{
    int result = func(1, 2);
    printf("Hello World!\n");
    return 0;
}
```

Stack frame

...
2
1
Ret-add
pre-ebp
ret
...

缓冲区溢出技术基础

- EIP、EBP、ESP指针

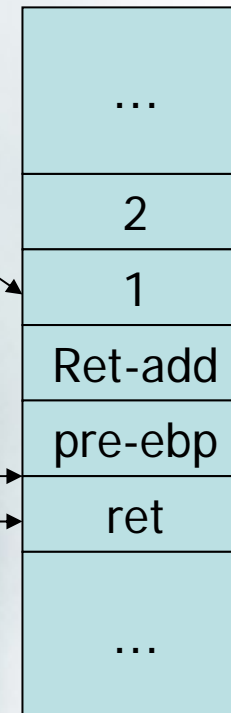
```
00401000 push
00401001 mov
00401003 push
00401004 mov
00401007 add
0040100A mov
0040100D mov
00401010 mov
00401012 pop
00401013 ret
```

```
ebp          保存pre-ebp
ebp,esp
ecx
eax,dword ptr [ebp+8]
eax,dword ptr [ebp+0Ch]
dword ptr [ebp-4],eax
eax,dword ptr [ebp-4]
esp,ebp
ebp
```

```
0040101C push
0040101E push
00401020 call
00401025 add
```

```
2
1           参数压栈
00401000
esp,8
```

Stack frame



恢复到调用func前的栈



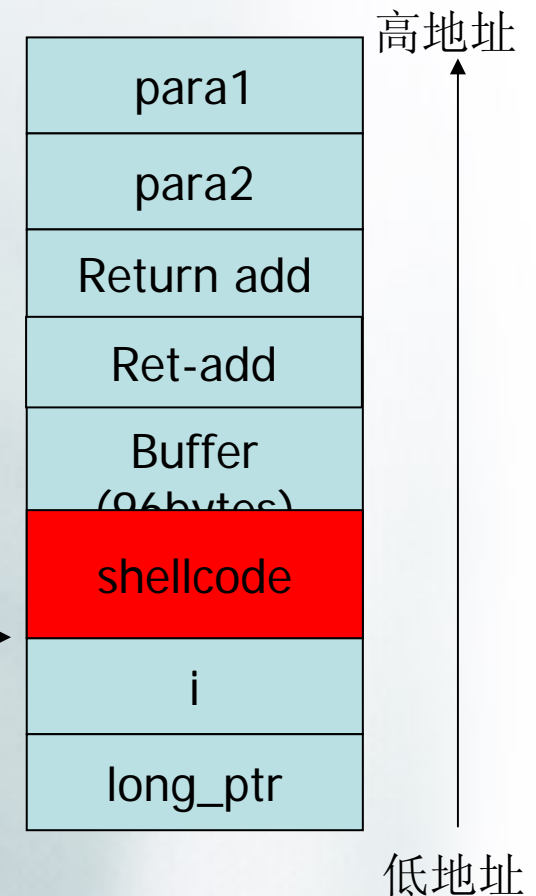
缓冲区溢出技术基础

- 产生缓冲区溢出的原因
 - ❑ 在C语言中，指针和数组越界不保护是**Buffer overflow**的根源，而且，在C语言标准库中就有许多能提供溢出的函数，如**strcat()**, **strcpy()**, **sprintf()**, **vsprintf()**, **bcopy()**, **gets()** 和 **scanf()**
 - ❑ 通过指针填充数据
 - ❑ 不好的编程习惯

栈溢出

```
#include <stdio.h>
#include <string.h>
char shellcode[] = "\xeb\x1f\x.....";
char large_string[128];
int main(int argc, char **argv){
    char buffer[96]; int i;
    long *long_ptr = (long *) large_string;
    for (i = 0; i < 32; i++){
        *(long_ptr + i) = (int) buffer;
    }
    for (i = 0; i < (int) strlen(shellcode); i++){
        large_string[i] = shellcode[i];
    }
    strcpy(buffer, large_string);
    return 0;
}
```

Stack frame



堆溢出

- 在**.data**、**.bss**和**heap**中溢出的情形，都称为**heap overflow**，这些数据区的特点是：
数据的增长由低地址向高地址
- 比较少引起人们的关注，原因在于
 - ❑ 比栈溢出难度更大
 - ❑ 需要结合其他的技术，比如
 - 函数指针改写
 - **Vtable**改写
 - **Malloc**库本身的漏洞



缓冲区溢出目的

- 溢出之后，让程序执行我们指定的代码
- 由于这段代码往往不能太长，所以需要精心设计，并且充分利用系统中现有的函数和指令
- 对于不同的操作系统
 - ❑ **Linux/Unix**，尽可能地得到一个**shell**(最好是 **root shell**)
 - ❑ **Windows**，一个可以远程建立连接的**telnet**会话

缓冲区溢出过程

- 过程

- 找到有漏洞的程序，如在输入非正常字符串的时候，出现异常。

- 或者从程序中找漏洞，用好的反汇编工具，加上耐心。

- ❖以一个特定的字符串作为线索，跟踪到 **strcpy** 这样的函数，看是否有边界检查

- 编写 **shellcode**

- 编写试验程序，直到成功

- 示例

Shell Code代码的特点

- 本地**shellcode**:

- ❑ 最简单的做法是调用**CreateProcess**创建一个进程，执行**cmd.exe**

- ❑ **Unix下: shellcode.c**

```
#include <unistd.h>
int main() {
    char *name[2];
    name[0] = "/bin/sh";
    name[1] = NULL;
    execve(name[0], name, NULL);
    _exit(0); }
```



Shell Code代码的特点

- 远程**shellcode**:
 - ❑ 在远程机器上执行一个网络服务程序，打开一个**socket**端口，等待客户程序来连接。
 - ❑ 当客户程序连接上之后，为客户建立一个**cmd.exe**进程，并且把客户的输入输出与**cmd.exe**的输入输出联系起来，于是客户就有了一个远程**shell**



避免缓冲区溢出

- 编程的问题都可以在开发阶段防止，事实上，并没有这么简单
 - ❑ 有些开发人员没有意识到问题的存在
 - ❑ 有些开发人员不愿意使用边界检查，因为会影响效率和性能
 - ❑ 另一方面，许多遗留下来的代码还很多
- 在开发过程中，尽量使用带有边界检查的函数版本，或者自己进行越界检查



避免缓冲区溢出

- 作为用户，应该怎么做来避免缓冲区溢出？
- 关闭端口或服务。管理员应该知道自己的系统上安装了什么，并且哪些服务正在运行
- 漏洞一公布，大的厂商就会及时提供补丁
- 在防火墙上过滤特殊的攻击包
- 以所需要的最小权限运行软件



著名的缓冲区溢出介绍

- **Outlook缓冲区溢出**
 - ❑ Outlook处理电子邮件的方法有一个漏洞
 - ❑ 攻击者发送一个带有畸形头信息的电子邮件
 - ❑ 使受害计算机瘫痪或者在上面运行任意代码，建立后门
 - ❑ 攻击使在邮件头信息里放入过量信息
 - ❑ 很难被检测



著名的缓冲区溢出介绍

- **Linuxconf**缓冲区溢出
 - ❑ 带有**GUI**界面的管理员工具
 - ❑ 运行在**98**端口，允许通过**Web**远程访问
 - ❑ 意味着程序必须处理头信息来获得需要的信息
 - ❑ 在**HTTP**头信息中插入过量的信息，导致计算机缓冲区溢出
 - ❑ 不允许远程访问**98**端口



著名的缓冲区溢出介绍

- **IIS缓冲区溢出**
 - ❑ **IIS: windows**系统上最不安全的服务
 - ❑ **ISAPI**提供对管理脚本(.ida文件)和**Inernet**数据(.idq)查询的支持
 - ❑ 向**idq.dll**发送一个过量的变量
 - **GET /null.ida? [buffer]=X HTTP/1.1**
 - ❑ 一些具体的攻击方法没有公开
 - ❑ **Code Red**蠕虫利用了这一**IIS**漏洞
 - ❑ **IIS安全**

著名的缓冲区溢出介绍

- **Windows XP UPNP缓冲区溢出**
 - ❑ 通用即插即用功能，打开**5000**端口，**HTTP**格式
 - ❑ 缓冲区溢出，取得一个系统级**Shell**
- **SSL-Too-Open**
 - ❑ 当系统打开**HTTP**服务和**SSL**服务时
 - 通过**SSL**溢出一个和启动**HTTP**服务器帐号权限相等的**Shell**
 - ❑ 如果使用**root**帐号启动**WEB**服务，攻击者就取得**root**权限



综合扫描

- 安全扫描审计

- 分类

- 网络安全扫描
 - 系统安全扫描

- 优点

- 较全面检测流行漏洞
 - 降低安全审计人员的劳动强度
 - 防止最严重的安全问题

- 缺点

- 无法跟上安全技术的发展速度
 - 只能提供报告，无法实际解决
 - 可能出现漏报和误报

综合扫描器的使用





几款著名的扫描器

- **ISS Internet Scanner**

优点：报告功能强大，漏洞检查集完备，可用性很好。

平台：**Windows NT**

URL: [Http://www.iss.net](http://www.iss.net)

- **SAINT**

以**SATAN**为基础的网络安全扫描工具。

平台：**UNIX**

URL: [Http://www.wwwdsi.com](http://www.wwwdsi.com)



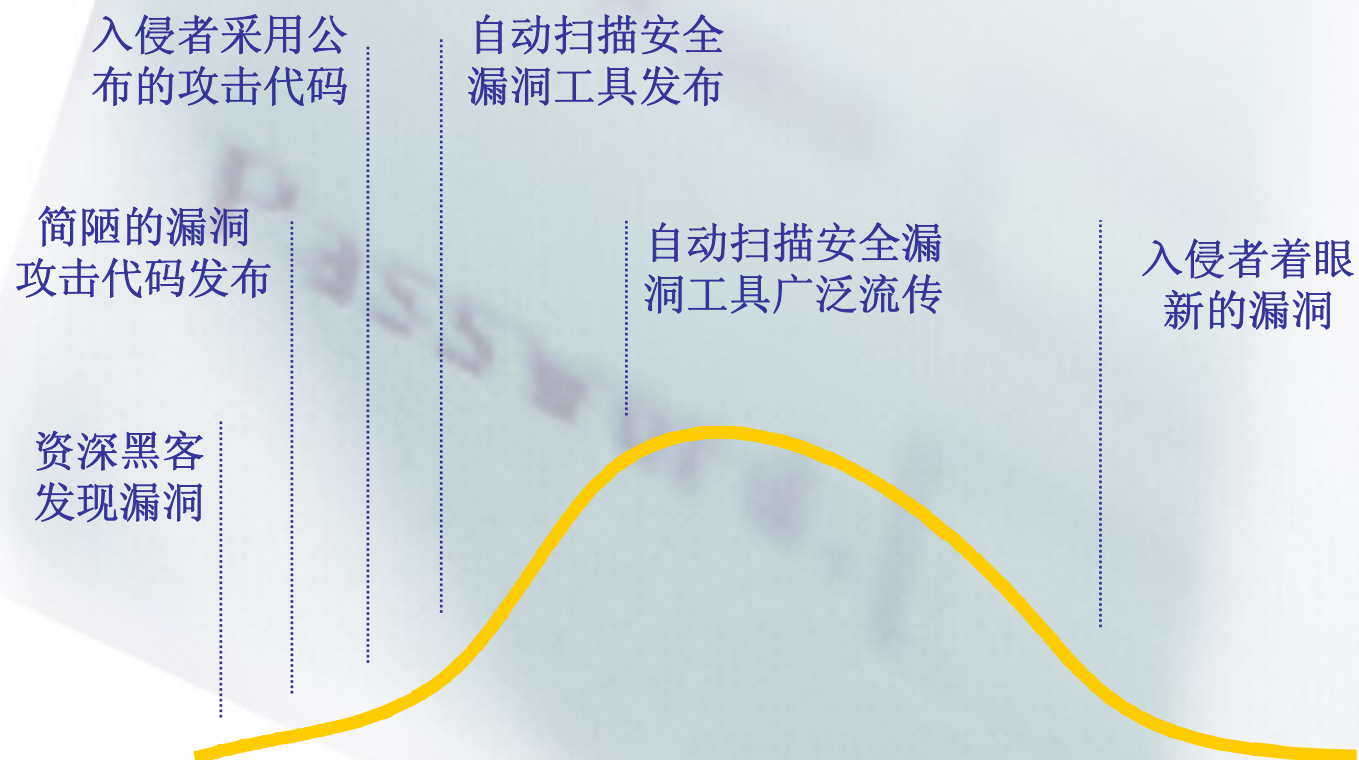
扫描器产品介绍

- **Nessus**

由**Renaud**编写的开放源码项目。

- 优点：采用分布式结构引擎具有极大弹性，可扩展性强，漏洞库较全面。
- 平台： **UNIX**
- URL: [Http://www.nessus.org](http://www.nessus.org)
- [演示](#)

漏洞利用周期图表





漏洞攻击的防范措施

- 安装防火墙，禁止访问不该访问的服务端口，使用NAT隐藏内部网络结构
- 安装入侵检测系统，检测漏洞攻击行为
- 安装安全评估系统，先于入侵者进行模拟漏洞攻击，以便及早发现漏洞并解决
- 提高安全意识，经常给操作系统和应用软件打补丁

木马概述

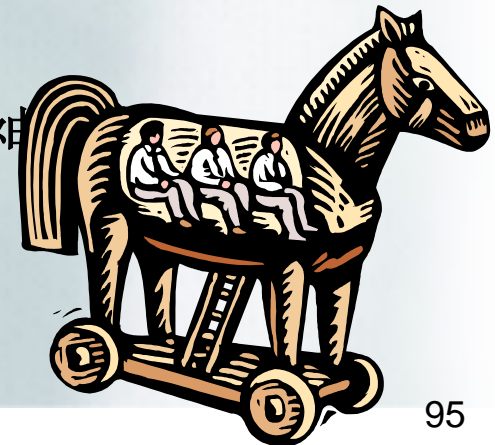
特洛伊木马程序可以直接侵入用户的电脑并进行破坏，它常被伪装成工具程序或者游戏等诱使用户打开带有木马程序的邮件附件或从网上直接下载，一旦用户打开了这些邮件的附件或者执行了这些程序之后，它们就会在计算机系统中隐藏一个可以在启动时悄悄执行的程序。这种远程控制工具可以完全控制受害主机，危害极大。

Windows下：

Netbus、subseven、BO、冰河、网络神

UNIX下：

Rhost ++、Login后门、rootkit等





木马组成

- 对木马程序而言，它一般包括两个部分：客户端和服务端。
- 客户端程序是控制者所使用的，用于对受控的计算机进行控制。服务器端程序和客户端程序建立起连接就可以实现对远程计算机的控制了。
- 木马运行时，首先服务器端程序获得本地计算机的最高操作权限，当本地计算机连入网络后，客户端程序可以与服务器端程序直接建立起连接，并可以向服务器端程序发送各种基本的操作请求，并由服务器端程序完成这些请求，也就实现对本地计算机的控制了。
- 木马本身不具备繁殖性和自动感染的功能。



木马分类

- **远程访问型木马**是现在最广泛的特洛伊木马，它可以访问受害人的硬盘，并对其进行控制。这种木马用起来非常简单，只要某用户运行一下服务端程序，并获取该用户的IP地址，就可以访问该用户的计算机。这种木马可以使远程控制者在本地机器上做任意的事情，比如键盘记录、上传和下载功能、截取屏幕等等。
- **密码发送型木马**的目的是找到所有的隐藏密码，并且在受害者不知道的情况下把它们发送到指定的信箱。
- **键盘记录型木马**非常简单的，它们只做一种事情，就是记录受害者的键盘敲击，并且在LOG文件里做完整的记录。这种木马随着系统的启动而启动，知道受害者在线并且记录每一件事。

木马常用欺骗法

- 捆绑欺骗：如把木马服务端和某个游戏捆绑成一个文件在邮件中发给别人；
- 危险下载点：攻破一些下载站点后，下载几个下载量大的软件，捆绑上木马，再悄悄放回去让别人下载；或直接将木马改名上载到**FTP**网站上，等待别人下载；
- 文件夹惯性点击：把木马文件伪装成文件夹图标后，放在一个文件夹中，然后在外面再套三四个空文件夹；
- **zip**伪装：将一个木马和一个损坏的**zip**包捆绑在一起，然后指定捆绑后的文件为**zip**图标；



木马常用欺骗法

- 木马隐藏方式：
 - ☐ 文件隐藏，伪装，如如Kernl32.dll, sysexlpr.exe 等。
 - ☐ 在任务管理器中隐形。
 - ☐ 悄无声息地启动，如在win.ini、system.ini、注册表等处设置；
 - ☐ 怎样察看启动时加载的进程？ [工具](#)

一种获取口令的攻击

- 改变登录程序到一个后门程序，它捕获用户口令，由于后门象正常的登录程序一样的工作，因此用户不能辨别出来。
- 在UNIX环境中想查看当前用户的PATH，可以用set或env命令来查看，普通用户的PATH会像这样
PATH=/bin:/usr/bin:/usr/local/bin:/usr/bin/X11，对于想执行不在这些目录下的命令时，用户需要打./，如#. /configure;
- 如果有些系统管理员为了省事，在自己的路径中加了一个“.”，可能会造成严重的问题。

实例

- 一个黑客取得了一个普通用户的权限，编写一个类似su这样的程序来骗得管理员的超级用户密码。源代码[su.c](#)。

```
$gcc -o su su.c
```

```
$cp su /tmp
```

```
$su
```

```
Password: [不可见的密码]
```

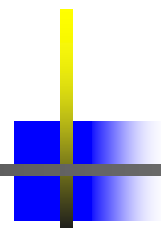
```
su: incorrect Password
```

- 这时我们可以到/tmp/下看到刚才输入的密码已被存到/tmp/catchpass这个文件中了。



针对木马攻击的防范措施

- 安装防火墙，禁止访问不该访问的服务端口，使用**NAT**隐藏内部网络结构
- 安装防病毒软件
- 提高安全意识，尽量避免使用来历不明的软件



Denial of Service (DoS)

此类攻击耗尽服务器的计算资源或存储资源或带宽资源，使得系统没有剩余的资源给其他用户可用。

Flooding攻击：发送垃圾数据来阻塞服务

如**SYN (Synchronize) Flooding**攻击，利用 TCP 实现中的漏洞（有限的缓存）来阻塞外来的连接请求。是当前最流行的DoS与DDoS（分布式拒绝服务攻击）的方式之一。



SYN-Flooding攻击

- 建立TCP连接的标准过程为三次握手。
- 假设一个用户向服务器发送了SYN报文后突然死机或掉线，那么服务器在发出SYN+ACK应答报文后是无法收到客户端的ACK报文的（第三次握手无法完成）。
- 这种情况下服务器端一般会重试（再次发送SYN+ACK给客户端）并等待一段时间后丢弃这个未完成的连接，这段时间的长度称为SYN 超时，一般来说这个时间是分钟的数量级（大约为30秒到2分钟）。
- 如果攻击者大量模拟这种情况，服务器端将为了维护一个非常大的半连接列表而消耗非常多的资源：数以万计的半连接，简单的保存并遍历也会消耗非常多的CPU时间和内存。
- 服务器忙于伪造的TCP连接请求而无暇理睬客户的正常请求。

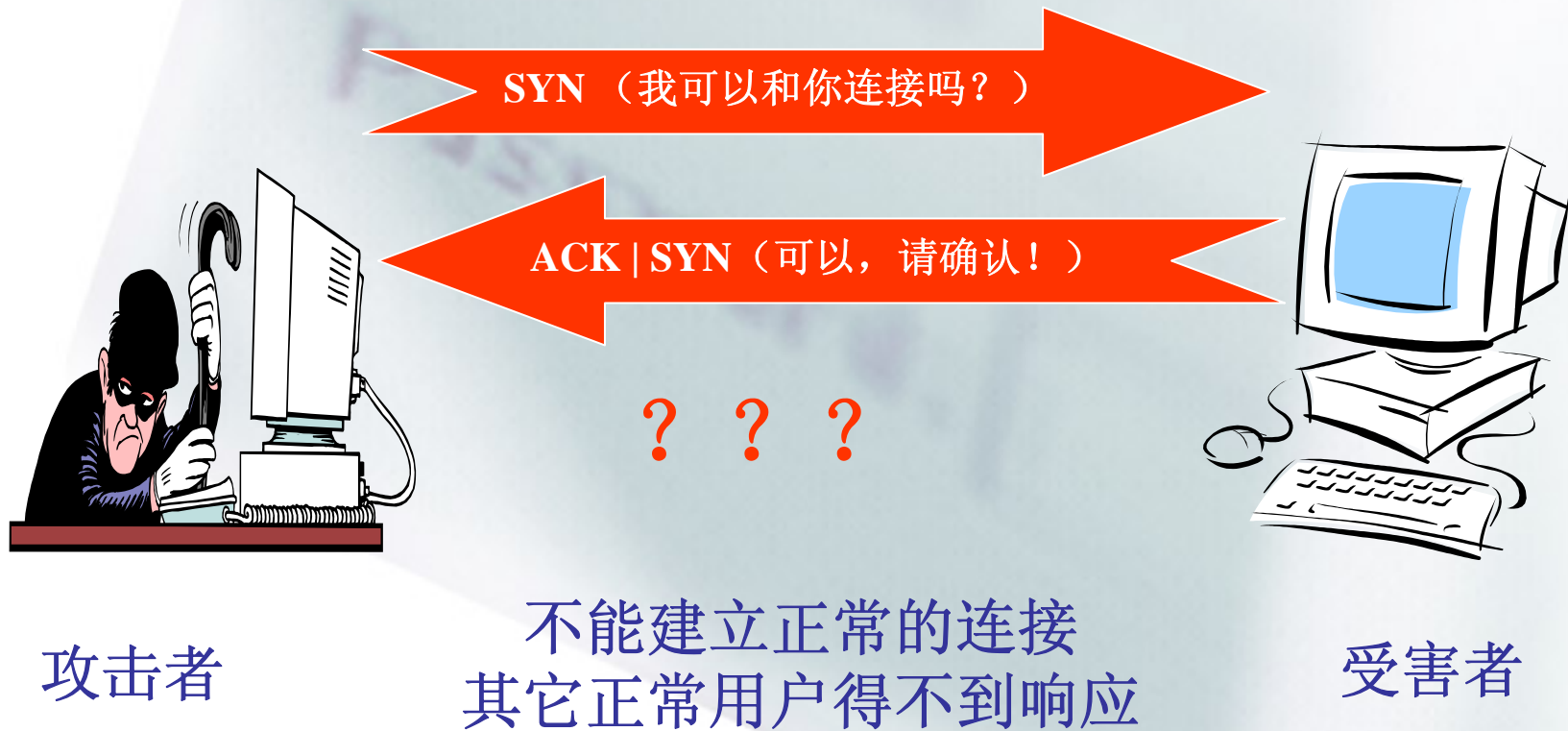
SYN-Flooding攻击

正常的三次握手建立通讯的过程



SYN-Flooding攻击

攻击者伪造源地址进行SYN请求



SYN-Flooding攻击



攻击者



正常用户

大量的tcp connect



不能建立正常的连接



受害者

SYN Defender



SYN proxy





一种DoS攻击

- **chargen**服务：UDP/TCP端口19。UDP **chargen server**若收到一个包,就会回一个包回去;而**TCP chargen server**若发现与**client**的连接存在,就会不断的发送封包给**client**, 直到连接中断。RFC864。

一种DoS攻击

\$telnet 140.0.0.0 19

Trying 140.0.0.0...

Connected to haydn.ntu.edu.tw.

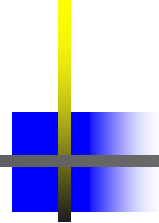
Escape character is '^['.

```
!"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefg
!"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefgh
!"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghi
!"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghij
!"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijk
!"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijkl
!"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklm
!"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmn
!"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmno
^C^]
```

Connection closed.

用tcp chargen进行DoS

- 使得**Client**端连接一个**URL**,在**html**里置放自动会**Load**(使用**Java/Script**)之http://CHARGEN_HOST:19/之**URL**,则此主机就会疯狂的向**CHARGEN_HOST**主机抓取大量的资料.而**CHARGEN_HOST**可以使用扫描技术来获得。
- 最终可以让**browser**收到大量的封包而消耗大量的内存。
- 攻击强度:
 - 1)**win2000**的**client**端**IE**,每秒钟会被消耗超过**1MB**的虚拟内存,数百秒之后, **client**反应迟钝,**VM**都被用光,然後**Win2000**会增大**VM**的大小,就会将**C**盘整个占满。
 - 2)**client**如果使用**http proxy**,则**proxy server**也会被拖垮。

- 
- 对策：关掉不必要的**TCP/IP**服务（如**chargen**），或者对防火墙进行配置阻断来自**Internet**的对这些服务的请求都可以防范**chargen flood**攻击。



Teardrop攻击

- 泪滴也是一种常见的拒绝攻击方式，它利用的是系统在实现时的一个错误，即攻击特定的IP协议栈实现片段重组代码存在的缺陷，会造成系统的死机或重新启动。
- 网络分组穿越不同的网络时，有时候需要根据网络最大传输单元MTU来把它们分割成较小的片。
- 早期的Linux系统在处理IP分片重组问题时，尽管对片段是否过长进行检查，但对过短的片段却没有进行验证，所以导致了泪滴形式的攻击。

Teardrop攻击



- ❖ 在linux(2.0内核)中有以下处理：当发现有位置重合时（ $\text{offset2} < \text{end1}$ ），将 offset2 向后调到 end1 （ $\text{offset2} = \text{end1}$ ），然后修改 len2 的值： $\text{len2} = \text{end2} - \text{offset2}$ ；注意此时 len2 可能会变成一个小于零的值，在以后处理时若不加注意便会出现溢出。
- ❖ 检测：组装时检查 len2 的值便可，这样可以发现所有的变种（如newtear）。



Land攻击

- 一个特别打造的SYN包中的源地址和目标地址都被设置成同一个服务器地址，这时将导致目标服务器向它自己的地址发送SYN-ACK消息，结果这个地址又发回ACK消息并创建一个空连接，每一个这样的连接都将保留直到超时掉。
- 对Land攻击的反应不尽相同，许多种类的UNIX将崩溃，而Windows NT会变得极其缓慢（大约持续五分钟）。目前流行的操作系统都已解决了此问题。
- 预防**LAND**攻击最好的办法是配置防火墙，对那些在外部接口入站的含有内部源地址的数据包过滤。

Land攻击

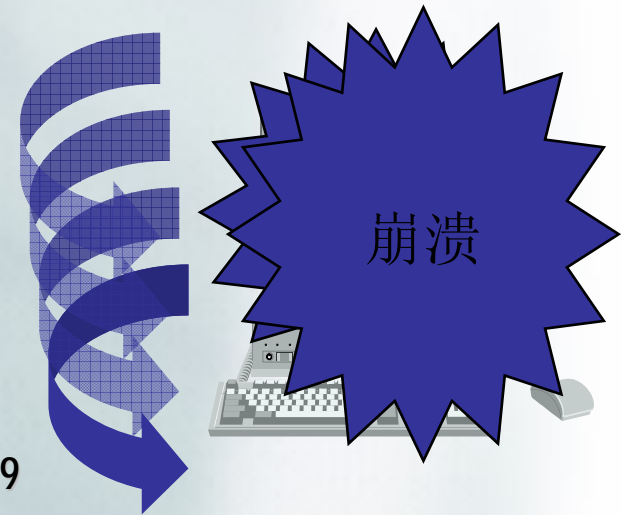


Land攻击

攻击者
172.18.1.1



目标
204.241.161.12



IP包欺骗

源地址 204.241.161.12 Port 139
目的地址 204.241.161.12 Port 139
包被送回它自己

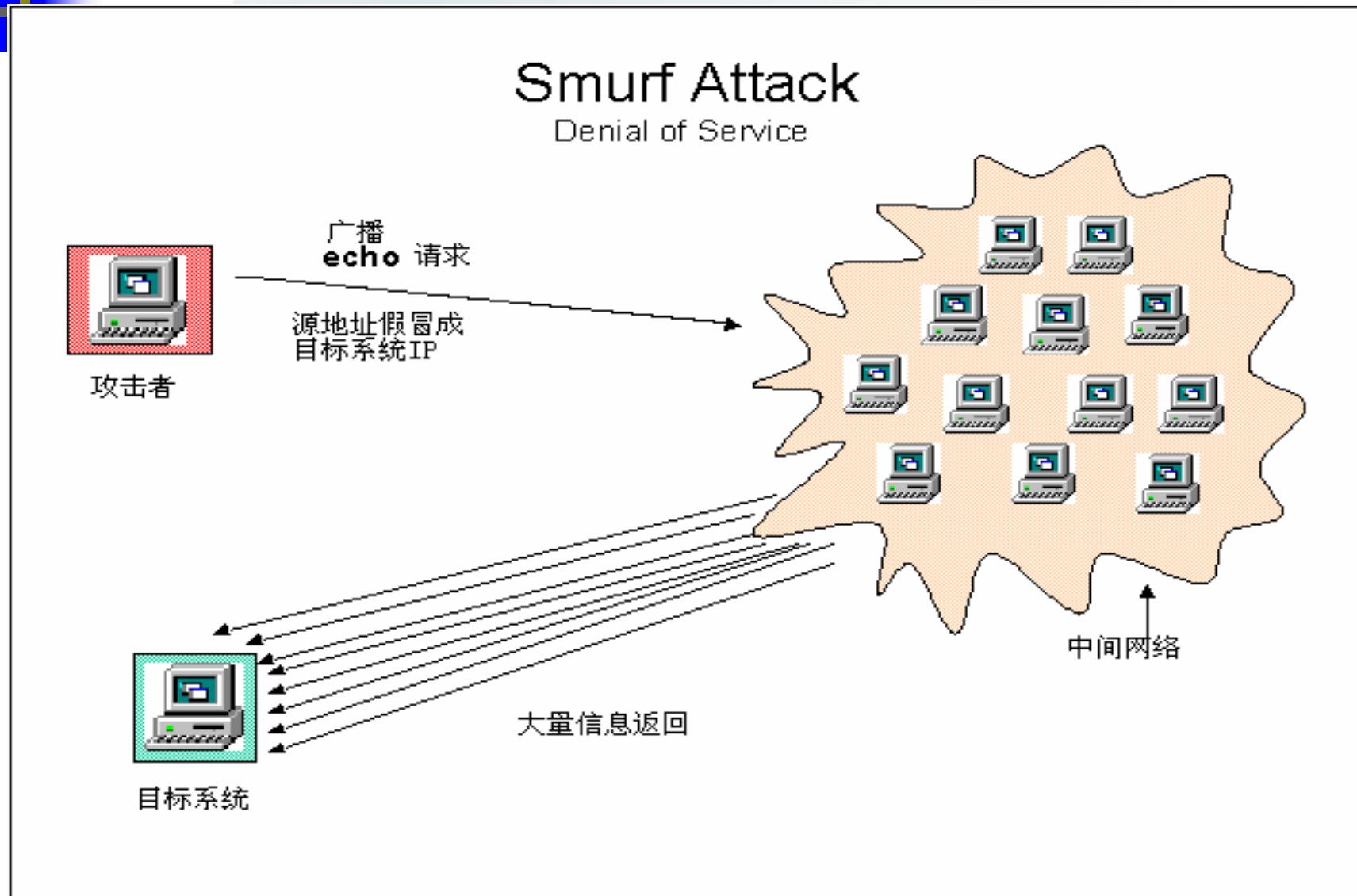


Smurf攻击

方法:

- 向大量的远程主机发送一系列的**ping** 请求命令。黑客把源**IP** 地址换成想要攻击目标主机的**IP** 地址。所有的远程计算机都响应这些**ping** 请求，然后对目标地址进行回复而不是回复给攻击者的**IP** 地址用。目标**IP** 地址将被大量的**ICMP** 包淹没而不能有效的工作。

Smurf攻击





Smurf攻击检测与预防

- 如果在网络内检测到目标地址为广播地址的icmp包。证明内部有人发起了这种攻击（或者是被用作攻击，或者是内部人员所为）。
- 如果icmp包的数量在短时间内上升许多(正常的ping程序每隔一秒发一个ICMP echo请求)，证明有人在利用这种方法攻击系统。
- 最好关闭外部路由器或防火墙的广播地址特性；为了防止被攻击，在防火墙的设置规则中可丢弃ICMP包。
- [Smurf](#)还有一个变种为Fraggle攻击，它利用UDP来代替ICMP包。

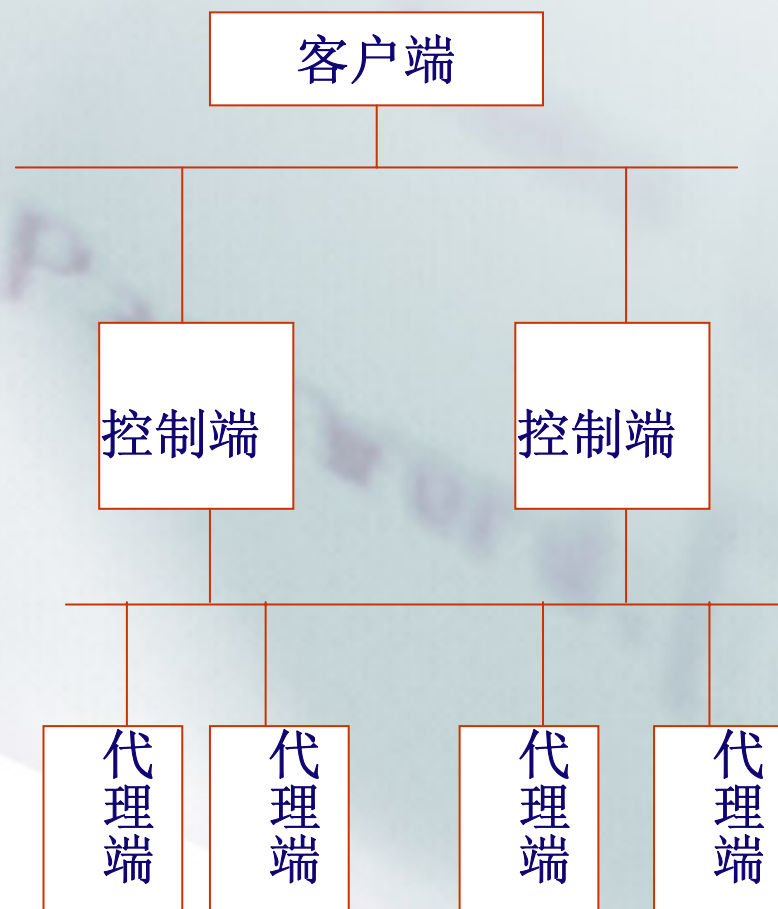


分布式拒绝服务攻击（DDoS）

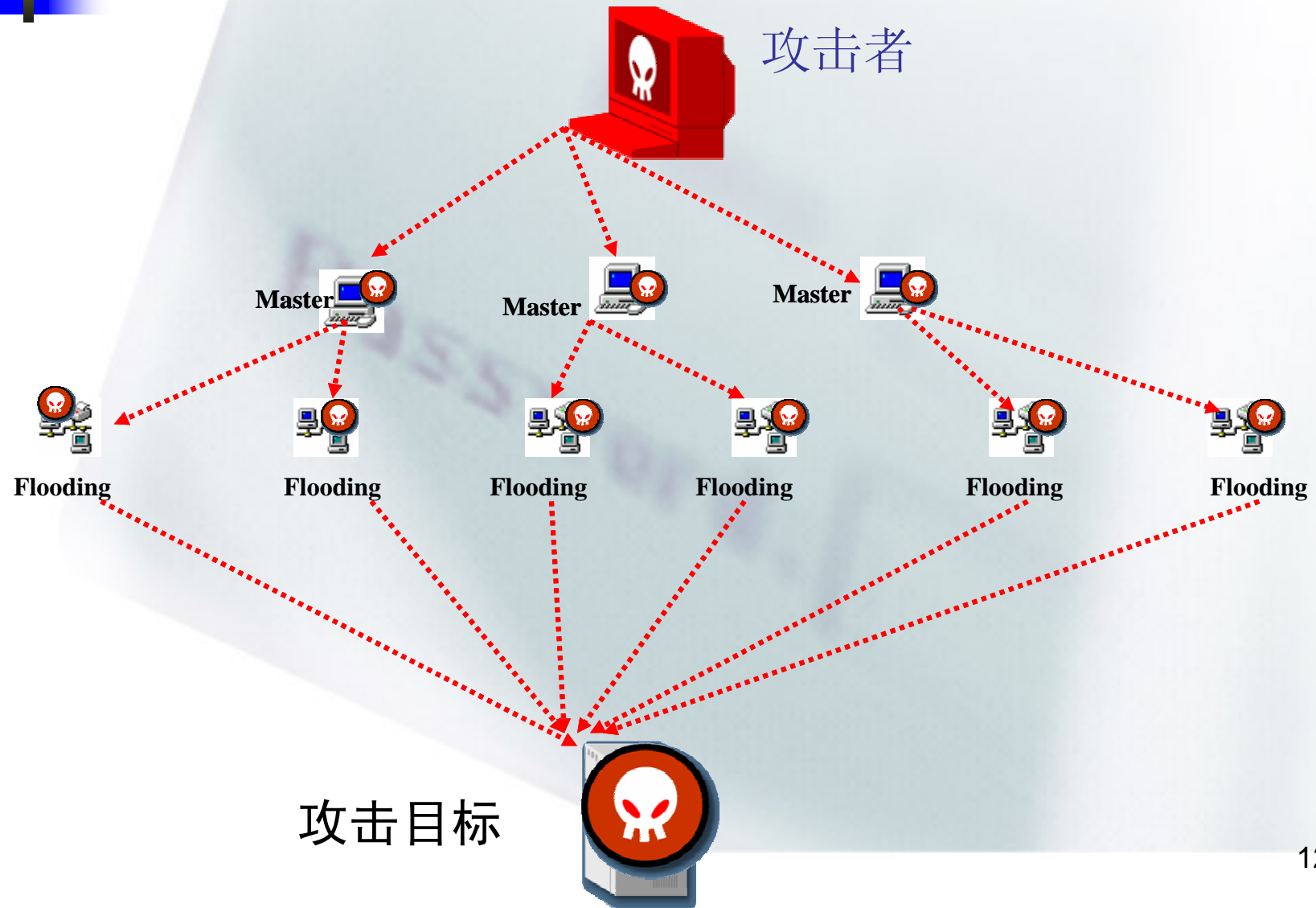
☞ **DDoS攻击由三部分组成**

- 客户端程序（黑客主机）
- 控制端（**Master**）
- 代理端（**Zombie**）

分布式拒绝服务攻击 (DDoS)



Yahoo受攻击过程





针对拒绝服务攻击的防范措施

- 安装防火墙，禁止访问不该访问的服务端口，过滤不正常的畸形数据包，使用NAT隐藏内部网络结构
- 安装入侵检测系统，检测拒绝服务攻击行为
- 安装安全评估系统，先于入侵者进行模拟攻击，以便及早发现问题并解决
- 提高安全意识，经常给操作系统和应用软件打补丁



计算机病毒

- 与生物医学上“病毒”的异同
 - 同:都具有传染性、流行性、以及针对性.....
 - 异:不时天生的,是人为编制的具有特殊功能的程序
- 病毒传播的主要途径: 网络下载或浏览、电子邮件、局域网、光盘或磁盘。
- 国内学者给出的一种较为广泛的定义:计算机病毒就是能够通过某种途径潜伏在计算机存储介质(或程序)里,当达到某种条件时即被激活的具有对计算机资源进行破坏作用的一组程序或指令集合。

计算机病毒-定义

- 直至**1994年2月18日**，我国正式颁布实施了《中华人民共和国计算机信息系统安全保护条例》，在《条例》第二十八条中明确指出：“计算机病毒，是指编制或者在计算机程序中插入的破坏计算机功能或者毁坏数据，影响计算机使用，**并能自我复制**的一组计算机指令或者程序代码。”此定义具有法律性、权威性。



计算机病毒-分类

- (1) 攻击**DOS**系统的病毒。
- (2) 攻击**Windows** 系统的病毒。
- (3) 攻击**UNIX**系统的病毒。攻击**UNIX**操作系统的计算机病毒相对来讲，为数不多，传播效率低，不易流行。
- (4) 攻击其他操作系统的病毒。
- (5) 攻击嵌入式操作系统的病毒。随着掌上电脑、手机各种便携电子设备的出现，**Palm**、**EPOC**等嵌入式操作系统也未能幸免于难。



最早的病毒

- 一九四九年,电脑的先驱者约翰.范纽曼(**John Von Neumann**)在他所提出的一篇论文《复杂自动装置的理论及组织的进行》里,即已把病毒程式的蓝图勾勒出来,当时,绝大部份的电脑专家都无法想像这种会自我繁殖的程式是可能的。
- 十年之后,在美国电话电报公司(**AT&T**)的贝尔(**Bell**)实验室中,这些概念在一种很奇怪的电子游戏中成形了,这种电子游戏叫做“磁蕊大战”(**core war**)。

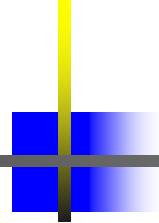


Morris父子

- 磁蕊大战是当时贝尔实验室中三个年轻程式人员在工余想出来的,他们是道格拉斯麦耀莱(**H, Douglas McIlroy**),维特.维索斯基(**Victor Vysotsky**)以及罗伯.莫里斯(**Robert T. Morris**),当时三人年纪都只有二十多岁。
- 附注: **Robert T. Morris** 就是后来把 **Internet** 搞得天翻地覆的那个**Robert T. Morris Jr.** 的爸爸,当时大 **Morris** 刚好是负责 **Arpanet**网路安全。

磁蕊大战

- 两方各写一套程式，输入同一部电脑中，这两套程式在电脑的内存互相追杀，并会停下来修理被对方破坏的几行指令；当它被困时，也可以把自己复制一次，逃离险境。
- 玩游戏的人只能看著萤幕上显示的战况，而不能做任何更改，一直到某一方的程式被另一方的程式完全“吃掉”为止。
- 磁蕊大战是个统称，事实上还可细分成好几种，麦耀莱所制定的游戏规则叫“达尔文”，以汇编语言遍写程序，叫有机体(**organism**)，两个有机体在电脑里争斗不休。

- 
- 爬行者(**Creeper**), 每一次把它读出时, 它便自己复制一个副本。另外, 它也会从一部电脑“爬”到另一部与之相连的电脑。很快地电脑中原有资料便被这些爬行者挤掉了。
 - 爬行者的唯一生存目的就是繁殖。为了对付“爬行者”, 有“收割者”(**Reaper**)。它的唯一生存目的便是找到爬行者, 把它们毁灭掉。当所有爬行者都被收割掉之后, 收割者便执行程式中最后一项指令: 毁灭自己, 从电脑中消失。



计算机病毒-分类

- (1) 源码型病毒:该病毒攻击高级语言编写的程序，在高级语言所编写的程序编译前插入到原程序中，经编译成为合法程序的一部分。
- (2) 入侵型病毒:这种病毒是将自身嵌入到攻击目标中，代替宿主程序中不常用到的堆栈区或功能模块，而不是链接在它的首部或尾部
- (3) 外壳型病毒:寄生在宿主程序的前面或后面，并修改程序的第一个执行指令，使病毒先于宿主程序执行，这样随着宿主程序的使用而传染扩散
- (4) 操作系统型病毒:这种病毒在运行时，用自己的逻辑模块取代操作系统的部分合法程序模块。



引导型病毒

- 引导型病毒是一种在系统引导时出现的病毒，引导型病毒先于操作系统运行，引导型病毒依托的环境是**BIOS**中断服务程序，引导型病毒感染硬盘时，驻留硬盘的主引导扇区或引导扇区。
- 事实上并没有什么**COMS**病毒。"**CMOS**设置破坏者"病毒，不等于说躲藏在**CMOS**里面的病毒。
- 另：**CMOS**是系统重要数据区，它记录的参数是系统时钟、硬件配置、用户口令。



相关概念

- (1) 逻辑炸弹是指修改计算机程序，使它在某种特殊条件下按某种不同的方式运行。逻辑炸弹也是由程序员插入其它程序代码中间的，但并不进行自我复制。
- (2) 特洛伊木马泛指那些内部包含有为完成特殊任务而编制的代码的程序，一种潜伏执行非授权功能的技术。木马本身不进行自我复制。
- (3) 计算机蠕虫程序是一种通过某种网络媒介——电子邮件、**TCP/IP**等自身从一台计算机复制到其他计算机的程序。与病毒在文件之间进行传播不同，它们是从一台计算机传播到另一台计算机，从而感染整个系统。



红色代码

- “红色代码”病毒造成的破坏主要是涂改网页，被攻击的服务器又可以继续攻击其他服务器。
- 病毒最初于**2001年7月19日**首次爆发，**7月31日**该病毒再度爆发，但由于大多数计算机用户都提前安装了修补软件，所以该病毒第二次爆发的破坏程度明显减弱。
- **Code Red**采用了一种叫做“**缓存区溢出**”的黑客技术，利用网络上使用微软**IIS**系统的服务器来进行病毒的传播。这个病毒使用服务器的**端口80**进行传播，而这个端口正是**Web**服务器与浏览器进行信息交流的渠道。
- **Code Red**主要有如下特征：**入侵IIS服务器**，code red会将**WWW**英文站点改写为“**Hello! Welcome to www.Worm.com! Hacked by Chinese!**”



白雪公主 (Hybris) 病毒

- 通过**EMAIL**接收的网络蠕虫。
- 蠕虫的代码中包含插件，这些插件可以。
- 当**Hybris**被启动后，从**Internet**网站上下载所需的插件，该站点被卸载后，病毒继续从新闻组**alt.comp.virus**上下载插件。
- 网络蠕虫**alt.comp.virus**的新闻组发送病毒升级的细节。这个消息是一个通讯信息，能够使主机上运行的蠕虫自动升级。

随机文件名

- 蠕虫在**Windows system**目录下以随机文件名创建其自身的副本。
- 在注册表中添加下列注册键：
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunOnce
{Default} = %WinSystem%\WormName
或
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunOnce
{Default} = %WinSystem%\WormName
其中"**WormName**" 是随机文件名, 例如:
CCMBOIFM.EXE
LPHBNGAE.EXE
LFPCMOIF.EXE



通过邮件发送

- 当蠕虫处于激活状态时，它将截获 **Windows** 用于建立网络连接的函数。蠕虫截获发送和接收的数据并扫描其**email**地址，一旦地址被检查到，蠕虫将等待一段时间然后将一个染毒的邮件（扩展名为**.EXE**或**.SCR**）发送给这些地址。
- 无法追踪发信人



计算机病毒-防治技术

- 病毒技术与反病毒技术存在着相互对立、相互依存的关系，他们都在彼此的较量中不断发展，当然，从总体上，反病毒技术要滞后于病毒技术。计算机病毒的防治技术可以分成四个方面，即检测、清除、免疫和预防。

计算机病毒-检测技术

(1) 搜索法.

搜索法是用每一种计算机病毒体含有的特定字符串对被检测的对象进行扫描。特征串选择的好坏，对于病毒的发现具有决定作用。缺点：如被扫描的文件很长时，扫描所花时间也越多；不容易选出合适的特征串；计算机病毒代码库未及时更新时，无法识别出新的计算机病毒；不易识别变形计算机病毒等。

(2) 校验和法

计算出正常文件的程序代码的校验和，并保存起来，可供被检测对象对照比较，以判断是否感染了计算机病毒。这种技术可侦测到各式的计算机病毒，包括未知病毒，但误判断高，无法确认病毒种类。



计算机病毒-检测技术

(3) 行为监测法

由于病毒在感染及破坏时都表现出一些共同行为，而且比较特殊，这些行为在正常程序中比较罕见，因此可通过监测这些行为来检测病毒的存在与否。该方法不仅可检测已知病毒，而且可预报未知病毒，但是有可能误报。

(4) 病毒行为软件模拟法

软件模拟法专门用来对付多态病毒，多态病毒在每次传染时都通过加密变化其特征码，使得搜索法失效。该方法监视病毒运行，待病毒自身密码破译后，再进行代码的分析。



计算机病毒-清除

- 病毒的清除是指将染毒文件的病毒代码摘除，使之恢复为可正常运行的健全文件。病毒消除可手工进行，也可用专用软件杀毒。
- 一个有趣的统计现象：计算机感染病毒超过三次的用户比仅感染一次的用户要多。

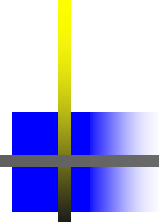
计算机病毒-预防

- 经常进行数据备份，特别是一些非常重要的数据及文件，以免被病毒侵入后无法恢复。特别是对系统引导区、**Boot**区及**FAT**表等要做好备份。
- 采用一套公认最好的驻留式防病毒软件，以便在对文件和磁盘操作时，进行实时监控，及时控制病毒的入侵,并及时、可靠地升级反病毒产品。

☐ 两个判断题：

杀毒种数越多，并不意味着该杀毒软件越好。

反病毒软件能杜绝自身被病毒传染。

- 
- 对于从因特网上下载的可执行文件和WORD/EXECEL文件一定要非常小心，在打开这些东西之前一定要进行非常仔细的检查。
 - 宏病毒是一种的文件型病毒，它寄存于 **Word、Excel** 文档中
 - 打开邮件的附件之前一定要三思而后行，即使是来自你信任的人，因为他很可能已经被病毒感染。



计算机病毒-预防

- 在资源管理器中，选择“工具”|“文件夹选项”|“查看”，去掉“隐藏已知文件类型的扩展名”前的对号，这样就可以使那些想伪装成正常文件的病毒文件原形毕露。
- 大多数的蠕虫类病毒是通过Microsoft Outlook或Outlook Express进行传播的，注意安装最新的安全补丁。可以使用第三方邮件程序取代Outlook Express，如Foxmail、The Bat!等，由于它们的地址簿与Outlook Express不同，所以被病毒利用的可能性比较小。

计算机病毒-预防

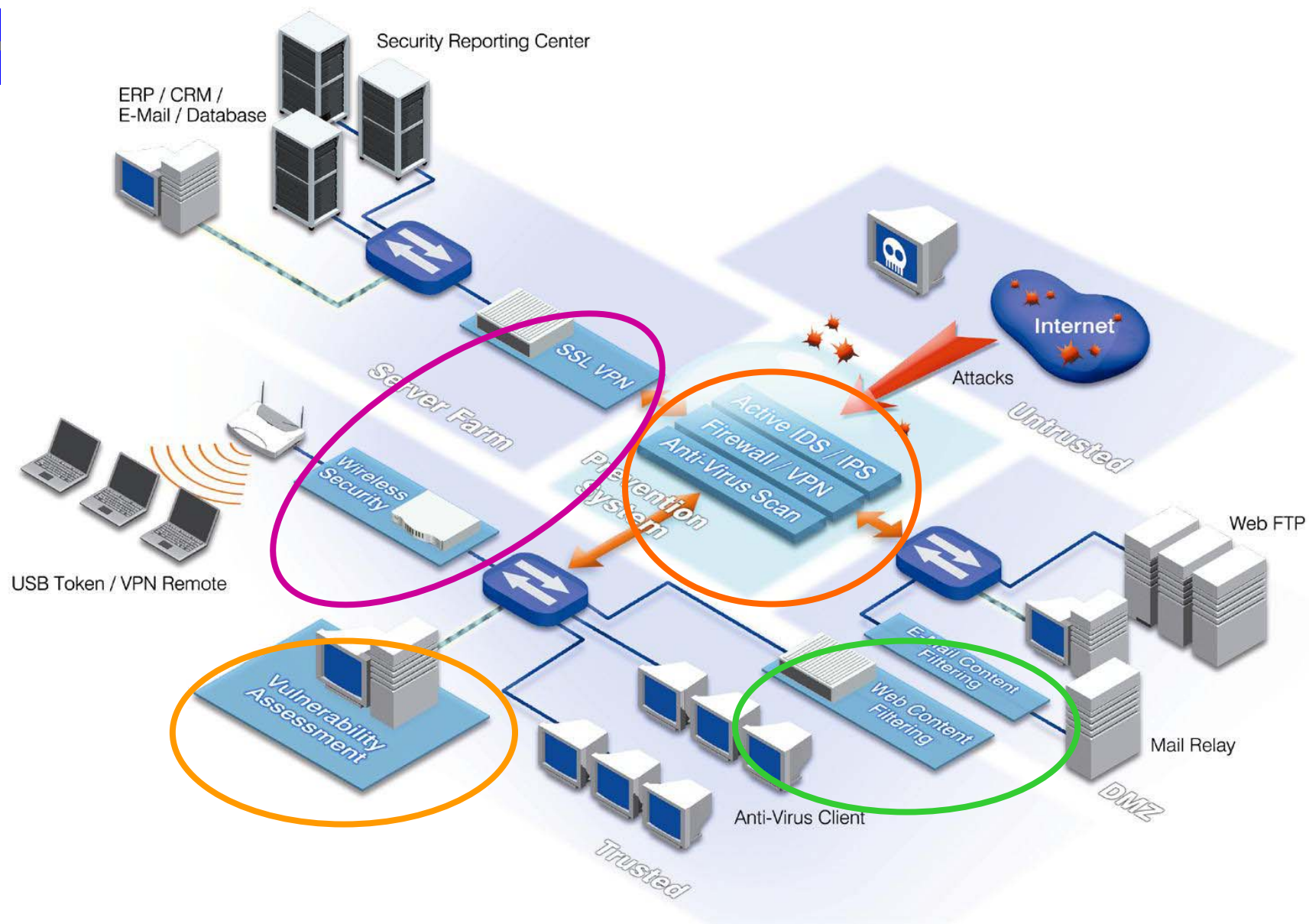
- 局域网的管理员应该特别注意的是，将所有共享目录的可执行文件设置成只读文件，限制普通权限的用户对这些目录的文件拥有写权限。
- 在控制面板中的“**Internet**选项”中，进行合理的“安全”设置，不要随意降低安全级别，以减少来自恶意代码和**ActiveX**控件的威胁，在系统推荐的默认设置级别“中”的基础上，点击“自定义级别”按钮，可以进一步进行更严格的设置，建议尝试着每次只更改一两个项目，如果导致不能正常上网，或者上网不方便了，则适当地降低安全设置，多试几次直到找到适合自己的最佳安全设置组合

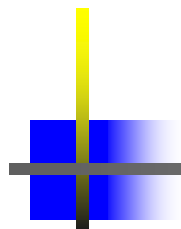


社交工程 social engineering

- 这种攻击属于非技术类攻击。入侵者给用户打电话或者发送E-mail告诉用户自己是系统管理员，这就是所谓的社交工程攻击的一个典型例子。通常，入侵者会要求用户告诉他口令或其他重要信息。
- 制定完善的安全管理制度
- 提高全员安全意识，尤其是管理者安全意识

网络安全基础架构





Q&A

谢谢！