哈尔滨工业大学

计算机科学与技术学院

《信息安全概论》

实验报告



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **姓 名** | cycleke | **学 号** | | xxxxxxxxx |
| **班 级** | xxxxxxx | **指 导 教 师** | | xxx |
| **开 设 学 期** | 2019-2020第二学期 | | | |
| **实验题目** | snort与单台防火墙联动实验 | | | |
| **实验日期** | 2020-06-04 | | | |
| **评 定 成 绩** |  | | **评定人签字** |  |
| **评 定 日 期** |  |

1. 实验目的

通过该实验可以加深理解Snort的系统架构以及工作原理，掌握Snort与Iptables联动的实现方法。

二、实验环境

服务器：snort-host（Centos6.5），IP地址:  10.1.1.12

Snort版本：2.9.7.6（最新）  Guardian版本：1.7（最新）

操作主机：host（Windows XP）,  IP地址: 随机

测试主机：test（Windows XP）,  IP地址: 随机

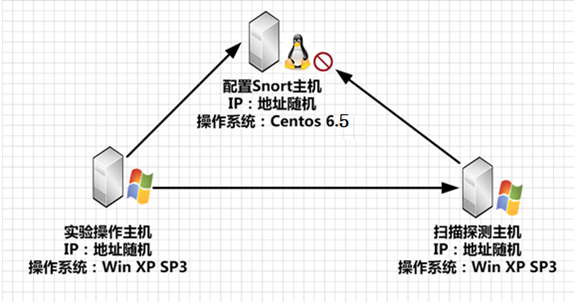


图 1三台主机间关系

三、实验内容与实验要求

**实验内容**

本实验介绍了Snort这种强大的轻量级网络入侵检测系统，让我们深入理解Snort的系统架构以及工作原理，利用Snort的告警日志，自定义开发脚本，掌握Snort和iptables联动的方法。

**基本原理**

**Snort简述**

Snort是一个强大的轻量级网络入侵检测系统，它能够检测到各种不同的攻击方式，对攻击进行实时报警。此外，Snort具有很好的扩展性和可移植性，并且这个软件遵循GPL，这意味着只要遵守GPL的任何组织和个人均可以自由使用这个软件。

Snort具有实时流量分析和日志IP网络数据包的能力，能够快速地检测网络攻击，及时地发出报警。Snort的报警机制很丰富，例如：Syslog、用户指定的文件、一个Unix套接字，还有使用samba协议向Windows客户端发出WinPoup消息。利用XML插件，Snort可以使用SNML把日志存放到一个文件或者实时报警。Snort能够进行协议分析、内容的搜索/匹配。目前Snort能够分析的协议有TCP、UDP、ICMP，将来可能支持ARP、OSPF、IPX、RIP等协议，它能够检测多种方式的攻击和探测，例如：缓冲区溢出、CGI攻击、端口暴力破解、SMB探测以及web应用程序的攻击现在也已经有简单规则的支持。Snort自带的检测攻击的规则数量有限，但Snort支持用户自定义规则的加载，这对有能力的大型企业而言是个不错的IDS选择。

**Snort体系架构**

1、Snort工作原理与应用场景

Snort是一个基于模式匹配的网络入侵检测系统，实际上目前现在市场上的大多商业入侵检测系统都是基于模式匹配的，即将恶意行为和恶意代码预定成入侵规则特征库，然后将实际数据源于规则库的特征码进行匹配，以判断其中是否包含了入侵行为。

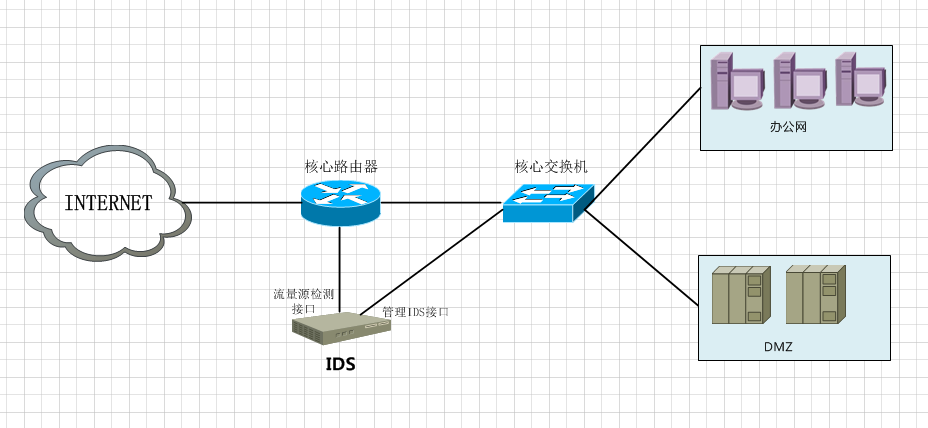


图 2IDS的一般应用场景图示

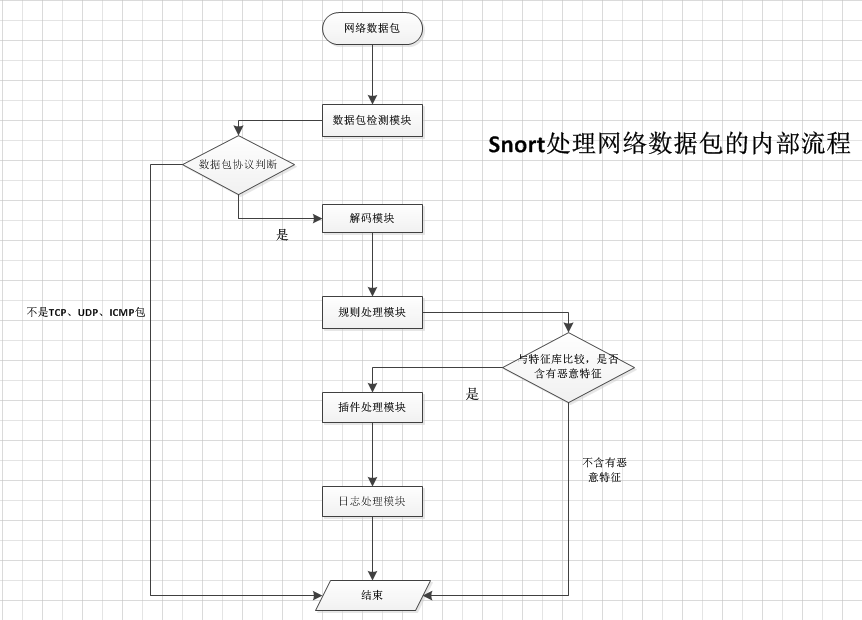


图 3Snort的大致处理流程

2、Snort的主体架构

Snort系统总体上是由规则集及Snort可执行程序两大部分组成。

1）Snort规则集

Snort规则集是Snort的攻击特征库，每条规则是一条攻击标识，Snort通过它来识别攻击行为。

2）Snort可执行程序

可执行程序由4个重要的子系统构成：数据包捕获和解码子系统、检测引擎、日志/报警子系统、预处理器。

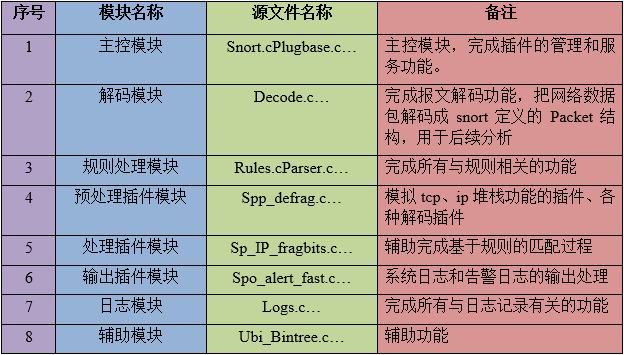


图 4Snort的总体结构

3. Snort的插件机制

1）预处理插件

预处理插件在规则匹配之前运行，完成的功能主要为：

（1）模拟TCP、IP堆栈功能的插件，如IP碎片重组、TCP流重组插件；

（2）各种解码插件：http解码插件、Unicode解码插件、RPC解码插件、Telnet解码插件等；

（3）规则匹配无法进行攻击检测时所用的插件：端口扫描插件、spade异常入侵检测插件、bo检测插件等

2）处理插件

处理插件在规则匹配阶段的parse rule options中被调用，辅助完成基于规则的匹配库。每个规则处理函数通常对应规则选项中的一个关键字，实现对这个关键字的解释。其主要功能为：

（1）检查协议各字段，如TCP Flag、ICMP Type、fragbits、RPC、dsize等；

（2）辅助功能，例如关闭连接、会话记录、攻击响应等

3）输出插件

输出插件在规则匹配过程中和匹配过程结束后调用，以便记录日志和告警。

4.总体流程

Snort的入侵检测流程分成两大步：

第一步是规则的解析流程，包括从规则文件中读取规则和在内存中组织规则。其过程为：

（1）读取规则文件；

（2）依次读取每条规则；

（3）解析规则；

（4）在内存中对规则进行组织，建立规则语法树；

第二步是使用这些规则进行匹配的入侵流程。其过程为对从网络上捕获的每一条数据报文和在第一步建立的规则树进行匹配，若发现存在一条规则匹配该报文，就表示检测到一个攻击，然后安装规则规定的行为进行处理；若搜索完所有的规则都没有找到匹配的规则，则视此报文正常。

Snort与iptables的联动

1. 概述

Snort最致命的弱点是不能阻断攻击。Snort的主要作用是对整个网络起到预警作用，从它的旁路部署模式也可以看出，它并不能阻断网络里的攻击行为。谁能阻断攻击呢？iptables。可为什么不采用iptables呢？因为iptables的规则过于固定，并且iptables并不能识别网络里的攻击行为。那能不能综合一下它们二者的优点互补对方的缺点，而达到检测到攻击即切断攻击连接这样的效果呢。答案是可以！

2. 实现方式和实现原理

通过前面的知识我们了解到，Snort有个插件机制提供了预处理插件和处理插件等方式。而这种插件在Snort里是支持自定义开发并加载的。因此第一种实现方式就是自定义开发插件，当检测到规则匹配时则调用远程或对应主机的防火墙，将有入侵行为的IP和端口，建立对应的一条iptables规则丢弃这个连接、端口的数据包或将此IP的所有包都丢弃。

相对于Snort的插件方式，第二种的实现方式非常简单且易于实现，很适合本次实验。这种方式就是利用一个简单的脚本实时读取告警日志，将记录到的ip和端口，创建对应的一条iptables规则，加入到远程或对应主机的防火墙规则中，也就是实现了同第一种方式相同的功能，虽然后者在处理速度上没有第一种方式及时，但整体防护能力上并未有太大什么区别。

实现方式总结如下：

（1）利用Snort的扩展功能，自定义开发集成插件（目前有snortsam）。

（2）利用Snort的告警日志，自定义开发脚本。（本次实验所采用，脚本为guardian）

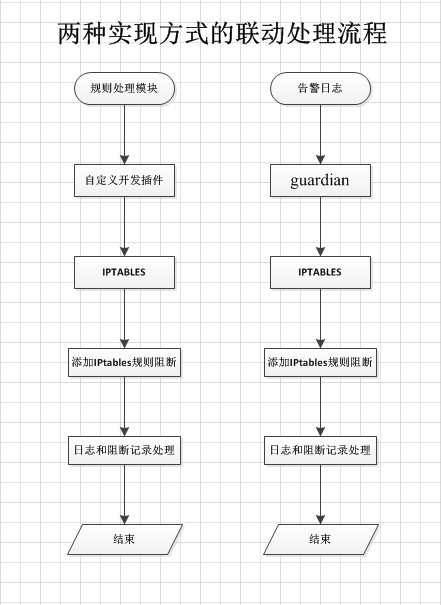


图 5两种实现方式的联动处理流程

四、实验过程与分析

**实验步骤一 Snort的安装和配置**

首先在操作主机上安装Xshell，之后在Xshell中使用ssh连接到Snort主机的Linux操作系统上。输入账户root和密码123456后即可连接到Linux主机。

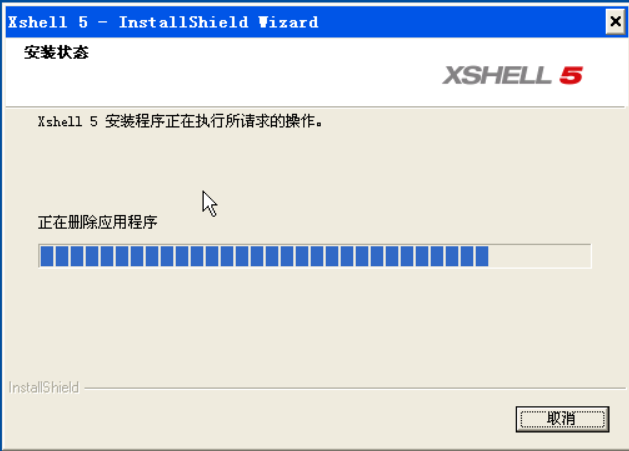


图 6安装Xshell

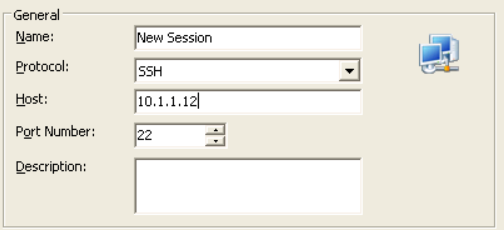


图 7使用ssh连接到Linux主机

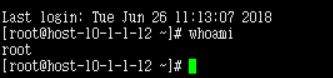


图 8连接成功

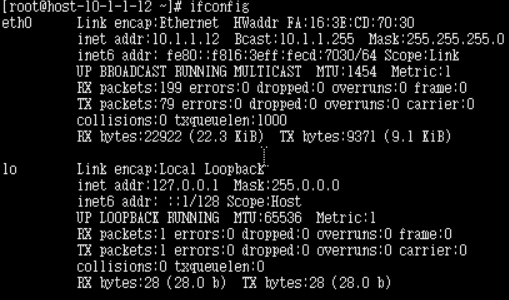


图 9查看网卡信息

在连接到Linux系统后，我们需要下载安装Snort、Snort规则库、DAQ 、libdnet、guardian。由于外网下载的速度缓慢，所以我们可以在<http://tools.hetianlab.com/tools/>下载压缩包。将下载的压缩包使用tar指令解压，获得源代码，之后就可以编译安装各个软件。

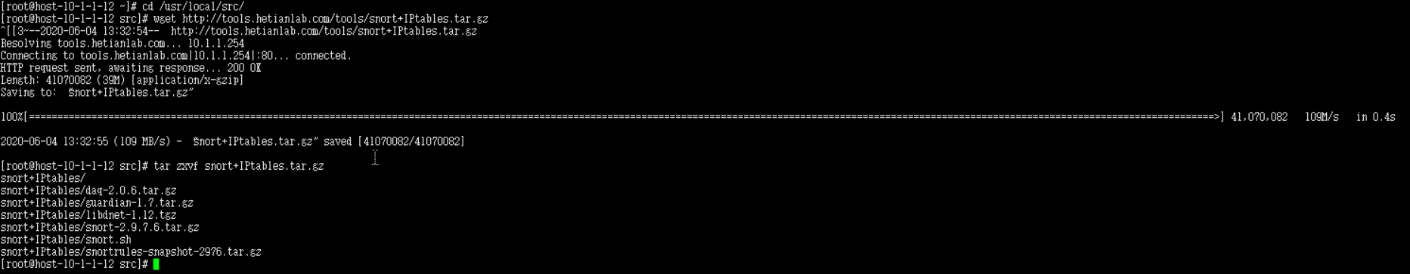


图 10下载压缩包



图 11安装指令

之后我们依次输入对应的指令来创建snort系统必要的文件夹，导入规则库，建立黑名单和白名单文件。

1. mkdir /etc/snort
2. mkdir /**var**/log/snort
3. cd /etc/snort
4. cp /usr/local/src/snort+IPtables/snort-2.9.7.6/etc/\* .
5. tar zxvf /usr/local/src/snort+IPtables/snortrules-snapshot-2976.tar.gz
6. cp ./etc/\* .
7. touch /etc/snort/rules/white\_list.rules /etc/snort/rules/black\_list.rules

之后我们需要为Snort添加一个用户和组，更改相关文件的权限，并修改Snort的配置文件。

我们需要将配置文件的对应内容修改为如下：

1. ipvar HOME\_NET 10.1.1.0/24
2. ipvar EXTERNAL\_NET any
3. **var** RULE\_PATH /etc/snort/rules
4. **var** SO\_RULE\_PATH /etc/snort/so\_rules
5. **var** PREPROC\_RULE\_PATH /etc/snort/preproc\_rules
6. **var** WHITE\_LIST\_PATH /etc/snort/rules
7. **var** BLACK\_LIST\_PATH /etc/snort/rules

取消下面四行的注释

1. preprocessor sfportscan: proto  { all } memcap { 10000000 } sense\_level { low }
2. include $PREPROC\_RULE\_PATH/preprocessor.rules
3. include $PREPROC\_RULE\_PATH/decoder.rules
4. include $PREPROC\_RULE\_PATH/sensitive-data.rules

之后为snort的文件在另一个位置创建一个同步链接



图 12添加用户和组



图 13更改对应文件的权限

在配置动态规则之前，我们需要使用uname指令来判断系统是32位还是64位。在导入动态规则后也需要对于相关文件修改权限，然后对出动态规则文件，设置权限。

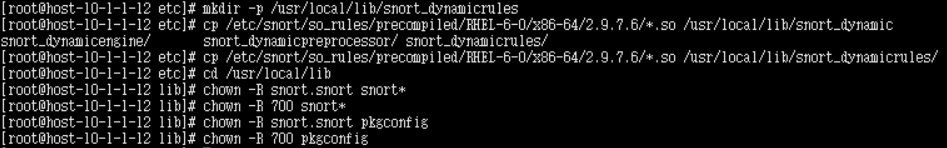


图 14配置动态规则

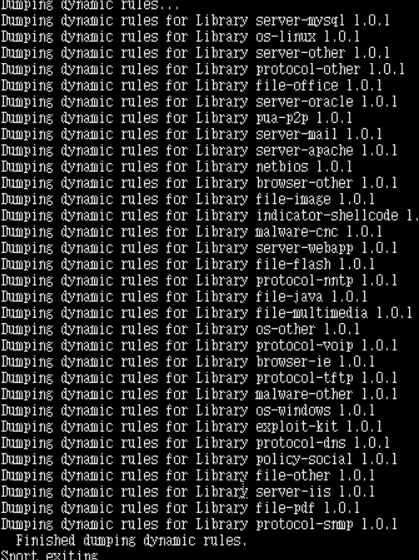


图 15导出动态规则文件

在启动Snort之前可以使用snort -T -c /etc/snort/snort.conf -i eth0指令来检测配置文件是否有错误。删除-T参数即可启动Snort。



图 16启动Snort

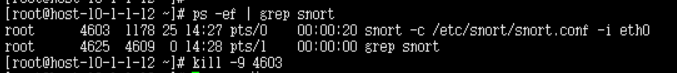


图 17停止Snort

**实验步骤二 guardian的安装和配置**

首先我们需要安装guardian。与Snort不同，guardian需要编译出二进制文件，而是复制脚本文件。

1. cd /usr/local/src/snort+IPtables
2. tar zxvf guardian-1.7.tar.gz
3. cd guardian-1.7
4. touch /etc/snort/guardian.ignore
5. touch /etc/snort/guardian.target
6. touch /**var**/log/snort/guardian.log
7. cp guardian.pl /usr/local/bin/
8. cp scripts/iptables\_block.sh /usr/local/bin/guardian\_block.sh
9. cp scripts/iptables\_unblock.sh /usr/local/bin/guardian\_unblock.sh
10. cp guardian.conf /etc/snort

之后我们来配置guardian。使用Vim打开guardian配置文件/etc/snort/guardian.conf。将配置文件的参数修改如下：

1. Interface   eth0
2. LogFile    /**var**/log/snort/guardian.log
3. AlertFile   /**var**/log/snort/alert
4. IgnoreFile  /etc/snort/guardian.ignore
5. targetFile   /etc/snort/guardian.target
6. TimeLimit 120

之后使用perl来运行脚本以启动guardian。

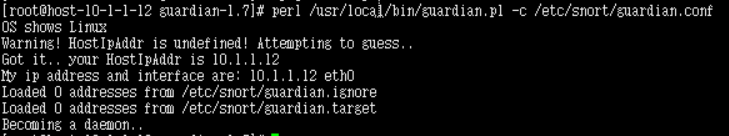


图 18guardian启动成功

**实验步骤三 联动测试**

先测试规则是否能够加载生效。在/etc/snort/rules/local.rules中添加两条规则来告警外网和内网之间的所有tcp流量。



图 19添加两条规则

之后重启Snort，可以在/var/log/snort/alert中发现有关连接。

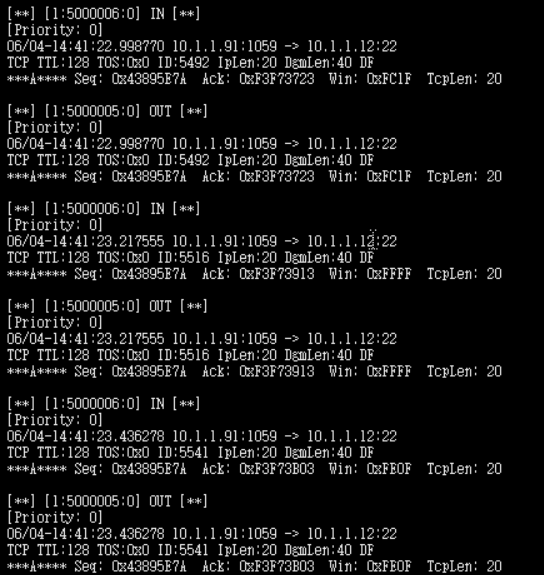


图 20Snort日志

确认无误后删除添加的规则，之后在另外一台Windows测试主机上进行测试。打开浏览器，登录http://tools.hetianlab.com/tools/X-Scan-v3.3-cn.rar下载X-Scan并解压，双击文件夹，打开X-Scan程序。

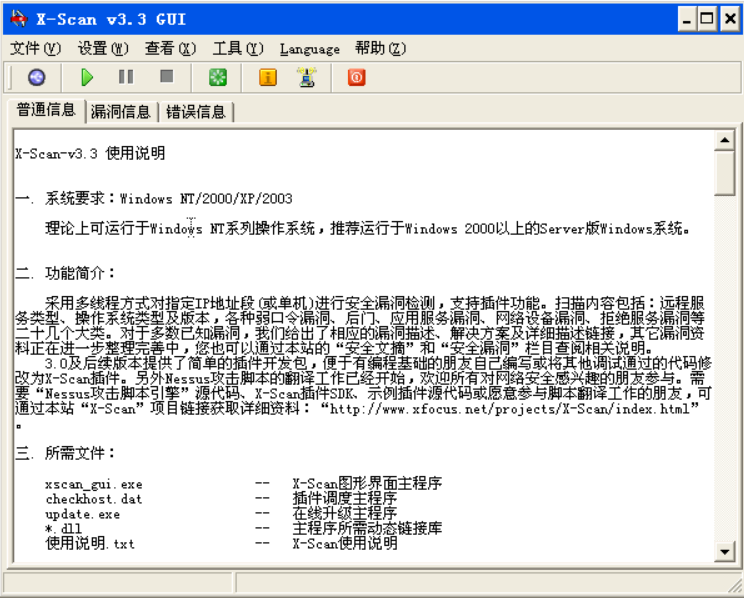


图 21X-Scan程序界面

点击“扫描参数”按钮，将指定IP范围设置成你的Snort主机IP地址，在“全局设置”的“扫描模块”中选中“全选”，“插件设置”的“SNMP相关设置”、“NETBIOS相关设置”、“漏洞检测脚本设置”均选择全部选中。

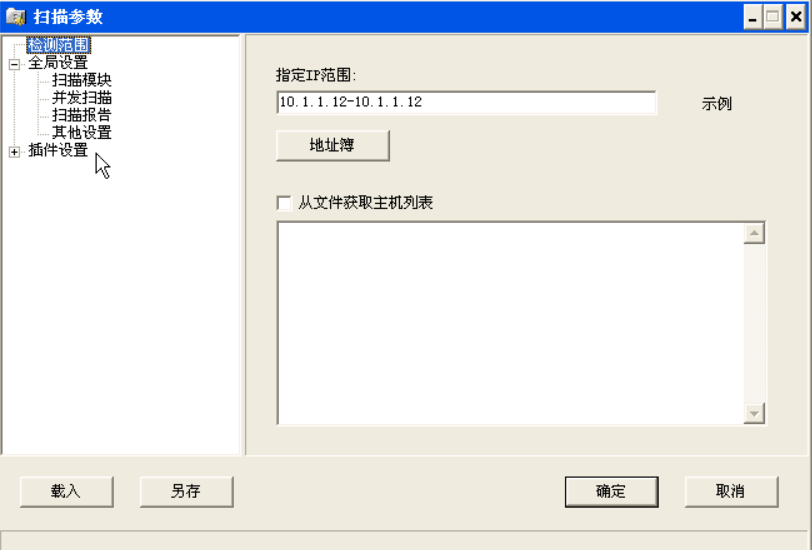


图 22设置检测范围

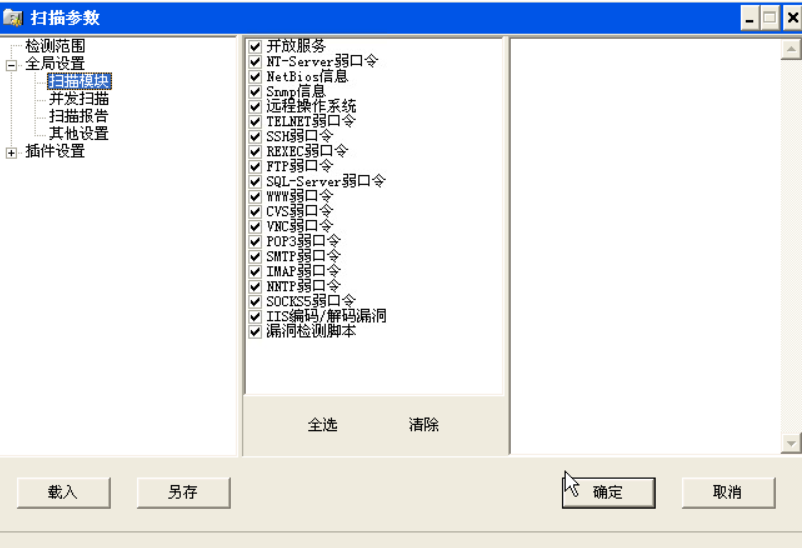


图 23全选扫描模块

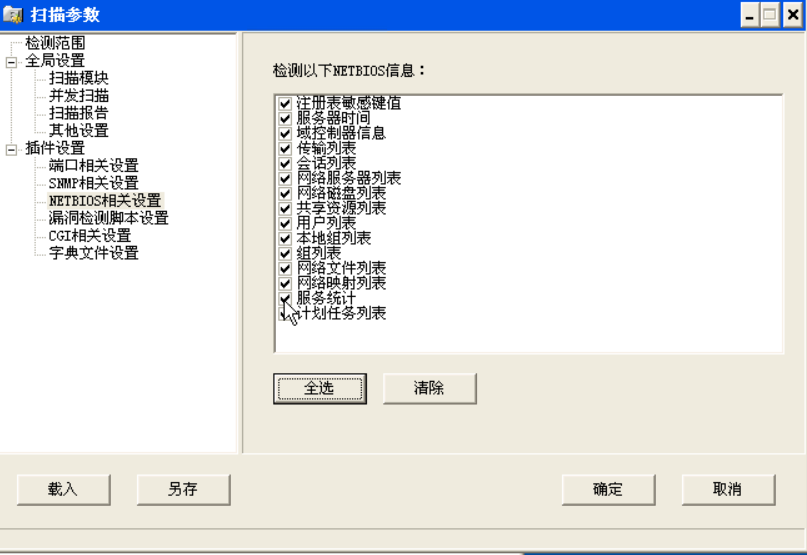


图 24全选NETBIOS相关设置

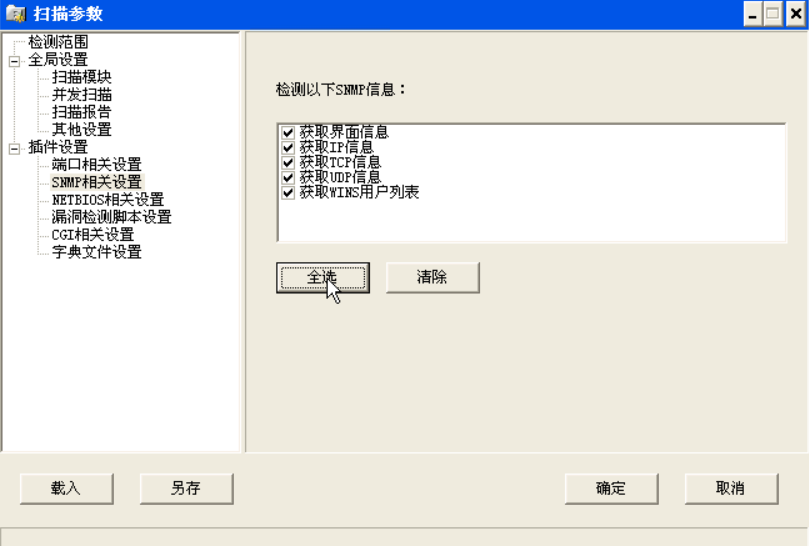


图 25全选SNMP相关设置

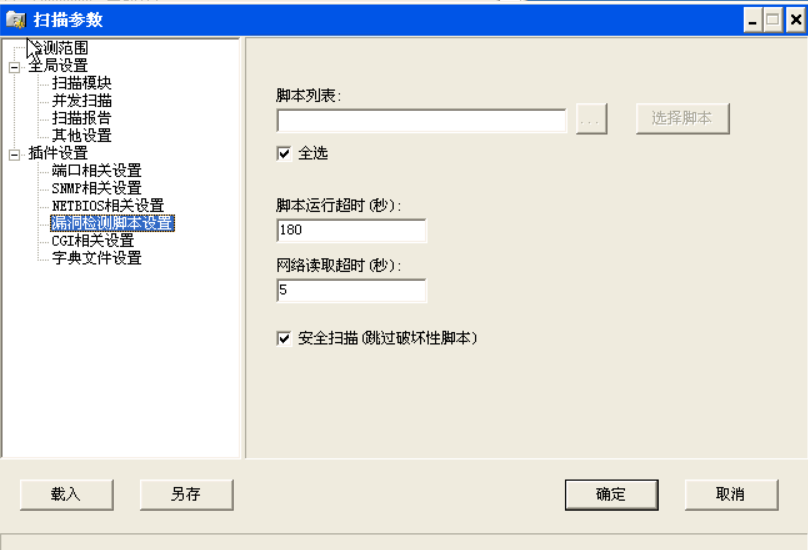


图 26全选漏洞检测脚本设置

在设置完成后点击开始按钮来扫描。

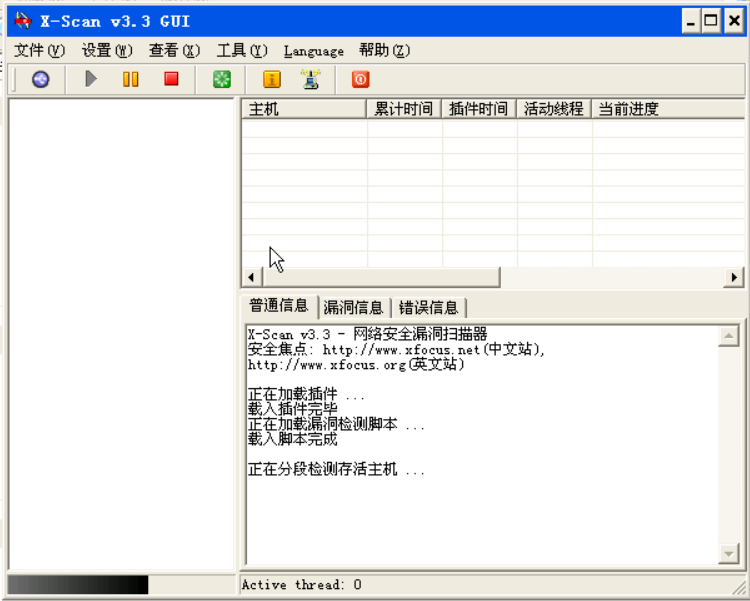


图 27X-Scan开始扫描

之后切换到操作主机，使用XShell来控制Snort主机。观测alert出现告警日志，于是启动guardian与iptables联动。在Snort主机上执行iptables -L即可看到新加入的规则。至此，单台防火墙联动已经成功实现。

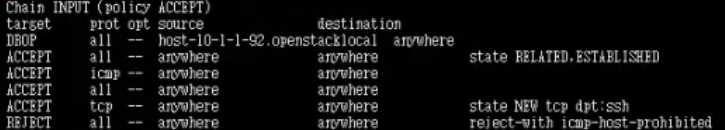


图 28iptables查询结果

五、实验结果总结

在此次实验中，我学习了如何安装配置Snort，利用Snort来检测各种连接，对攻击进行实时报警。同时由于Snort良好的拓展性，我们可以使用guardian脚本来拓展Snort，让它与iptables联动，进而实现将检测到的攻击阻断的功能。

**思考题目**

1、请查阅相关资料了解SnortSAM插件，分析并简述SnortSAM与guardian的优缺点。

SnortSAM是Snort的一个插件，利用了Snort的插件机制让Snort既可以实现对于攻击的检测，也具有改变边界设备访问控制的功能进而实现对于攻击的阻断。当检测到规则匹配时则调用远程或对应主机的防火墙，将有入侵行为的ip和端口，建立对应的一条Iptables规则丢弃这个连接、端口的数据包或将此ip的所有包都丢弃。SnortSAM的反应速度更快，可以更及时地阻断攻击。但是它地安装和配置比较复杂，不仅需要安装SnortSAM，还需要对Snort安装补丁。

guardian是利用Snort的告警日志，实时读取日志的内容，将记录到的IP和端口，创建对应的一条iptables规则，加入到远程或对应主机的防火墙规则中来阻断攻击。由于它本身是一个脚本，所以安装过程更为容易，配置也较方便。不过因为是根据日志信息来进行阻断，而不是在Snort内部执行，所以在处理速度上没有SnortSAM及时。

2、你认为SnortSAM和guardian的这种联动方式有哪些弊端？

SnortSAM与Snort联动是，是一种自定义开发插件，它的安装较麻烦，但是过程是在Snort内部执行，不方便扩展。

guardian是一个简单的脚本，是通过实时读取日志的内容来添加规则。由于这种机制是Snort在外部的，所以第三方利用向log中输入指定的信息来提供错误信息，让iptables将正常的访问阻塞。