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** | 郭茁宁 | **学 号** | | 1183710109 |
| **班 级** | 1837101 | **指 导 教 师** | | 翟健宏 |
| **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** |  |

实验报告撰写要求

实验操作是教学过程中理论联系实际的重要环节，而实验报告的撰写又是知识系统化的吸收和升华过程，因此，实验报告应该体现完整性、规范性、正确性、有效性。现将实验报告撰写的有关内容说明如下：

1、 实验报告模板为电子版。

2、 下载统一的实验报告模板，学生自行完成撰写和打印。报告的首页包含本次实验的一般信息：

* 组 号：例如：2-5 表示第二班第5组。
* 实验日期：例如：05-10-06 表示本次实验日期。(年-月-日)……
* 实验编号：例如：No.1 表示第一个实验。
* 实验时间：例如：2学时 表示本次实验所用的时间。

实验报告正文部分，从六个方面（目的、内容、步骤等）反映本次实验的要点、要求以及完成过程等情况。模板已为实验报告正文设定统一格式，学生只需在相应项内填充即可。续页不再需要包含首页中的实验一般信息。

3、 实验报告正文部分具体要求如下：

* 实验目的

*本次实验所涉及并要求掌握的知识点。*

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

*实验所使用的设备名称及规格，网络管理工具简介、版本等。*

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(WinXp),  IP地址: 随机

测试主机:test(WinXp),  IP地址: 随机

下载路径：http://tools.hetianlab.com/tools/X-Scan-v3.3-cn.rar

下载路径：http://tools.hetianlab.com/tools/Xshell.rar

三、实验内容与实验要求

*实验内容、原理分析及具体实验要求。*

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。

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。

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TCPflag、ICMPtype、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

      为什么要让Snort与Iptables联动呢？根据前面的介绍我们已经了解了Snort的工作方式与原理，聪明的你一定发现了Snort最致命的密码——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？——Iptable。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。

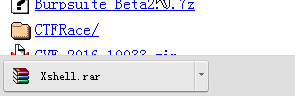
      实现方式总结如下：

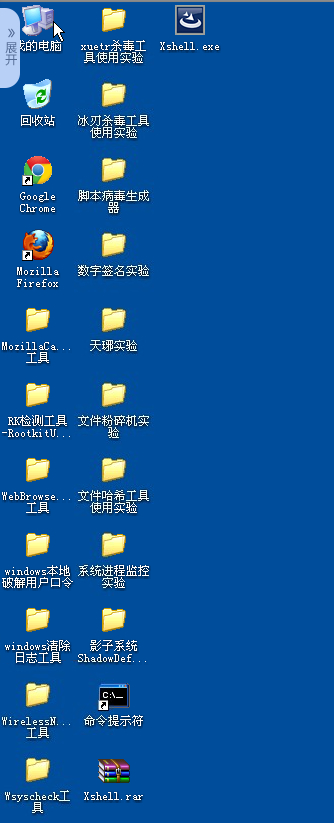
      (1).利用Snort的扩展功能，自定义开发集成插件(目前有snortsam)。

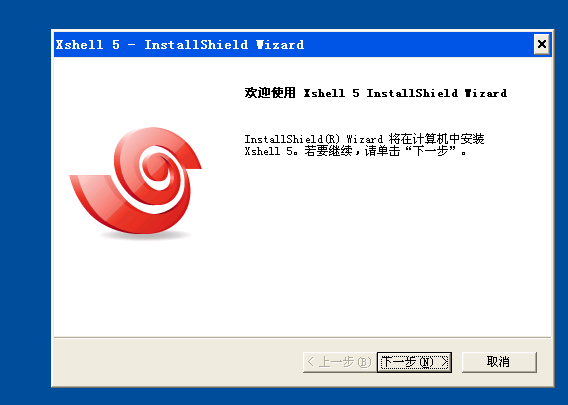
      (2).利用Snort的告警日志，自定义开发脚本。(本次实验所采用，脚本为guardian)

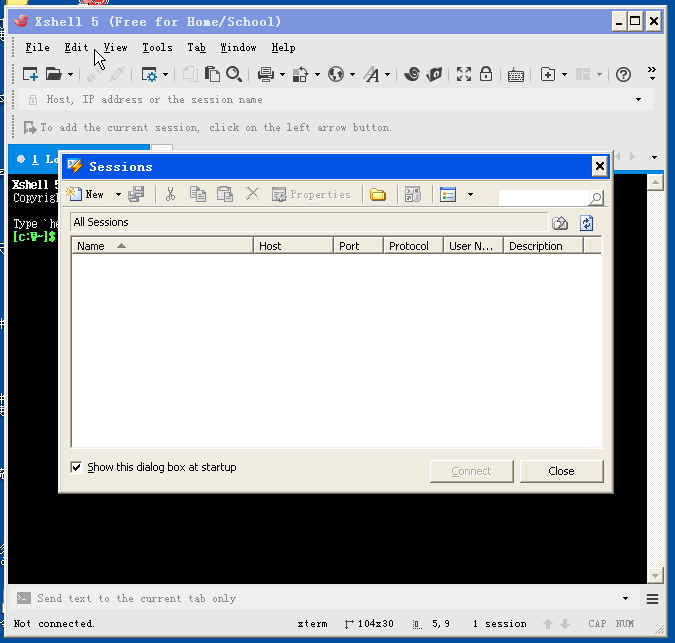
四、实验过程与分析

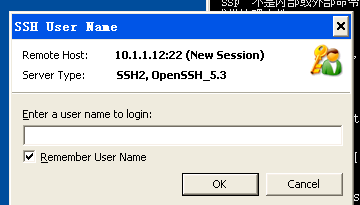
下载 xshell 登陆 linux 下载地址 ：http://tools.hetianlab.com/tools/Xshell.rar

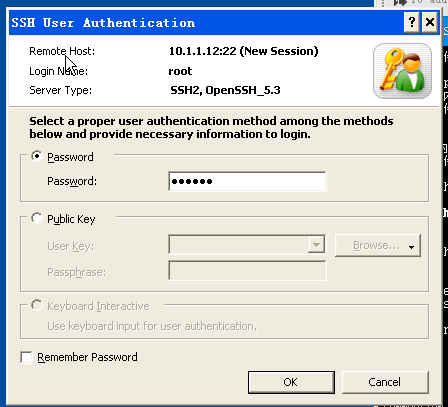


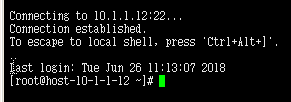












查看网卡

ifconfig（如果网卡没启动）

ifup eth0

更新系统

yum –y update

安装 snort 依赖包

yum -y install pcre pcre-devel gcc gcc-c++ zlib zlib-devel libpcap libpcap-devel make autoconf flex byacc bison libxml2-devel wget tcpdump

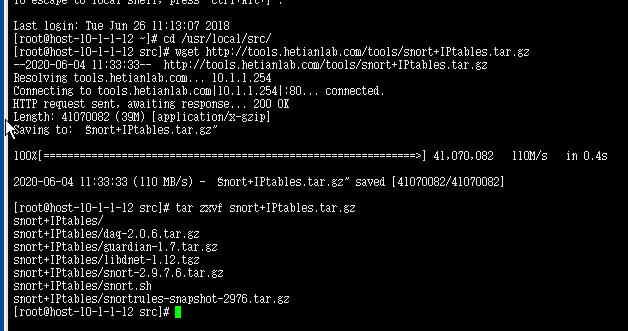
下载 Snort、Snort 规则库、DAQ 、libdnet、guardian

cd /usr/local/src/

wget http://tools.hetianlab.com/tools/snort+IPtables.tar.gz

tar zxvf snort+IPtables.tar.gz





安装 libdnet

cd /usr/local/src/snort+IPtables

tar zxvf libdnet-1.12.tgz

cd libdnet-1.12

./configure --with-pic

make && make install

cd /usr/local/lib/

ldconfig -v /usr/local/lib

安装 DAQ

cd /usr/local/src/snort+IPtables

tar zxvf daq-2.0.6.tar.gz

cd daq-2.0.6

./configure

make && make install

cd /usr/local/lib/

ldconfig -v /usr/local/lib

安装 snort

cd /usr/local/src/snort+IPtables/

tar zxvf snort-2.9.7.6.tar.gz

cd snort-2.9.7.6

./configure –enable-sourcefire

make && make install

cd /usr/local/lib/

ldconfig –v /usr/local/lib

Snort 配置

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

mkdir /etc/snort

mkdir /var/log/snort

cd /etc/snort

cp /usr/local/src/snort+IPtables/snort-2.9.7.6/etc/\* . (注意有个点号跟\*号中间有空 格)

tar zxvf /usr/local/src/snort+IPtables/snortrules-snapshot-2976.tar.gz

cp ./etc/\* .

touch /etc/snort/rules/white\_list.rules /etc/snort/rules/black\_list.rules

为 snort 添加一个用户和组

groupadd -g 40000 snort

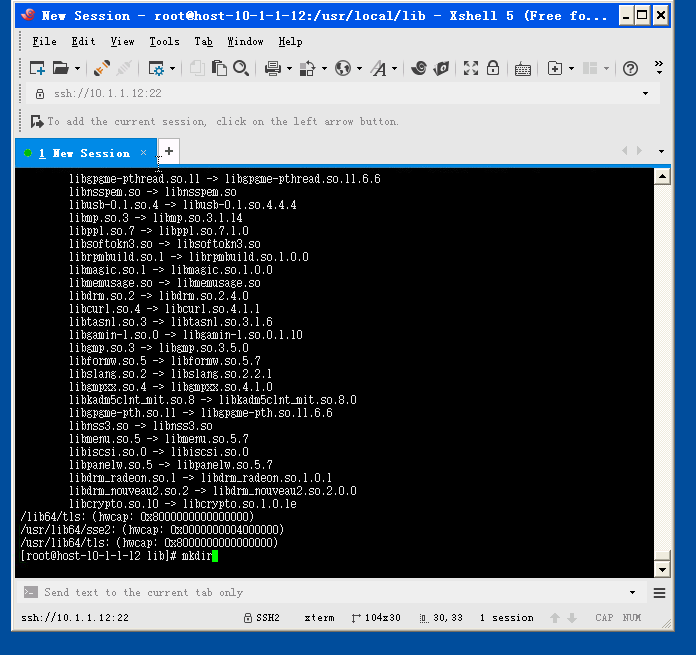
useradd snort -u 40000 -d /var/log/snort -s /sbin/nologin -c SNORT\_IDS -g snort

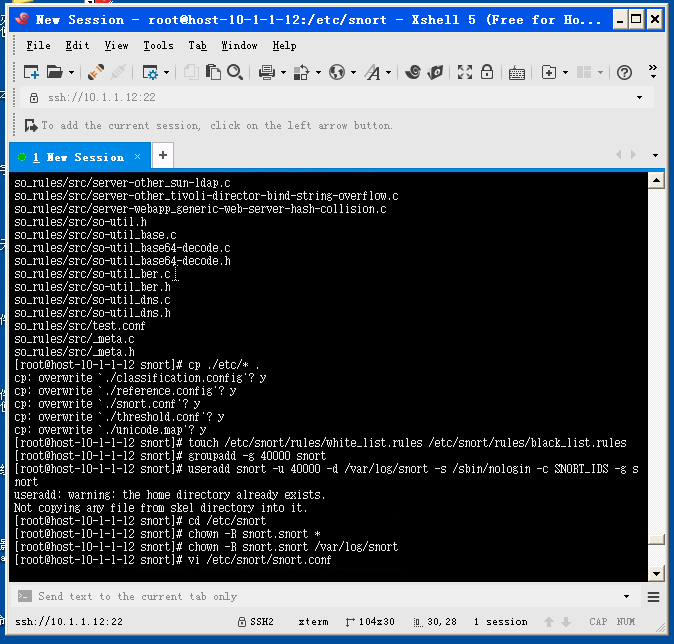
cd /etc/snort

chown -R snort.snort \*

chown -R snort.snort /var/log/snort

vi /etc/snort/snort.conf





ipvar HOME\_NET 10.1.1.0/24（本次实验为此 ip 段地址）

ipvar EXTERNAL\_NET any

var RULE\_PATH /etc/snort/rules

var SO\_RULE\_PATH /etc/snort/so\_rules

var PREPROC\_RULE\_PATH /etc/snort/preproc\_rules

var WHITE\_LIST\_PATH /etc/snort/rules

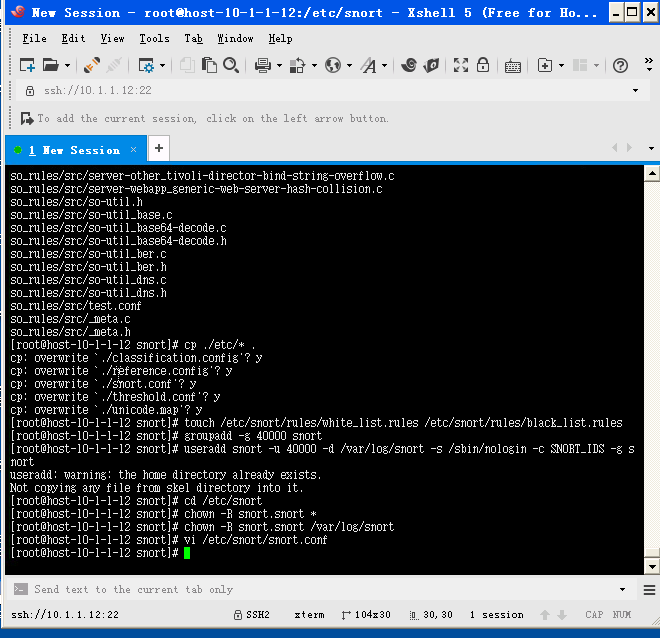
var BLACK\_LIST\_PATH /etc/snort/rules

preprocessor sfportscan: proto { all } memcap { 10000000 } sense\_level { low }(检 测端口扫描，不去注释也可以，去掉注释用 nmap 扫描即可看到扫描日志)

include $PREPROC\_RULE\_PATH/preprocessor.rules(注释去掉)

include $PREPROC\_RULE\_PATH/decoder.rules(注释去掉)

include $PREPROC\_RULE\_PATH/sensitive-data.rules(注释去掉)



为 snort 的文件在另一个位置创建一个同步链接

ln -s /usr/local/bin/snort /usr/sbin/snort

修改 snort 及 daq 等相关目录和文件权限

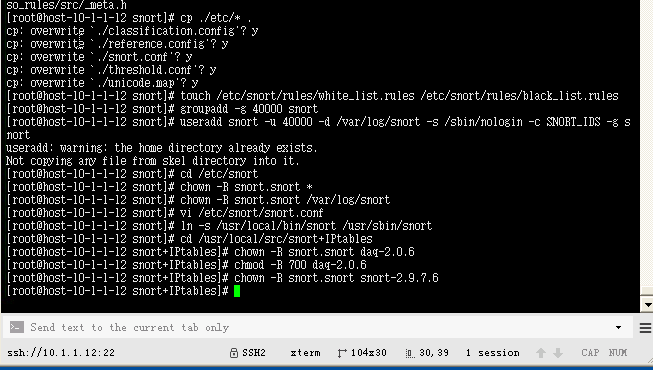
cd /usr/local/src/snort+IPtables

chown -R snort.snort daq-2.0.6

chmod -R 700 daq-2.0.6

chown -R snort.snort snort-2.9.7.6

chmod -R 700 snort-2.9.7.6

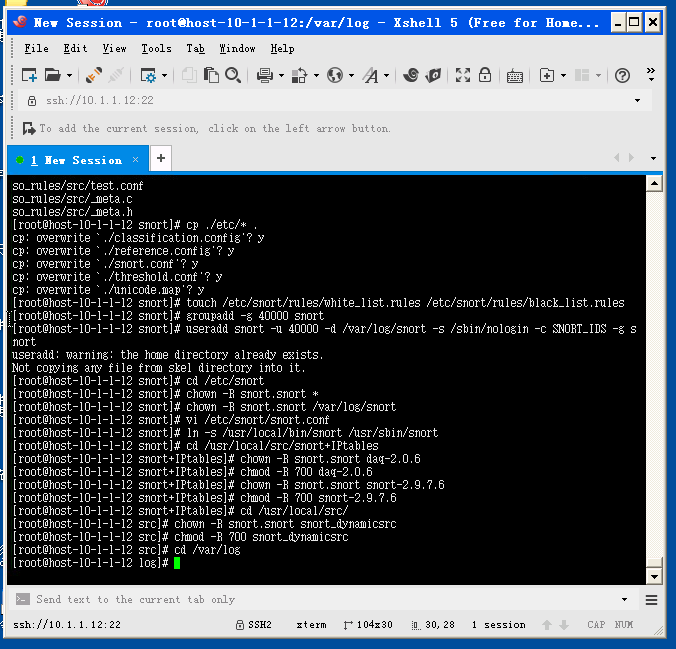


cd /usr/local/src/

chown -R snort.snort snort\_dynamicsrc

chmod -R 700 snort\_dynamicsrc

cd /var/log



chown -R snort.snort snort

chmod -R 700 snort

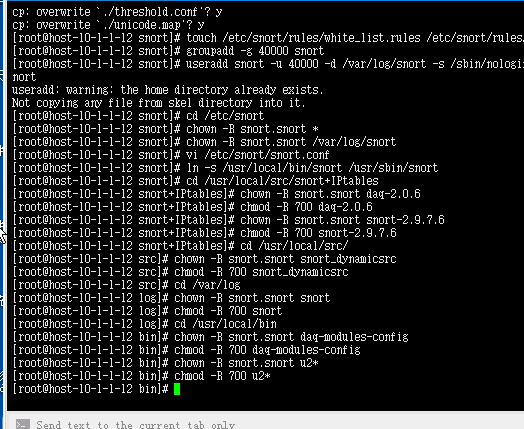
cd /usr/local/bin

chown -R snort.snort daq-modules-config

chmod -R 700 daq-modules-config

chown -R snort.snort u2\*

chmod -R 700 u2\*



cd /etc

chown -R snort.snort snort

chmod -R 700 snort

配置动态规则

mkdir -p /usr/local/lib/snort\_dynamicrules

cp /etc/snort/so\_rules/precompiled/RHEL-6-0/x86-64/2.9\*/\*.so /usr/local/lib/snort\_dynamicrules (实验环境是 64 位系统)

cd /usr/local/lib

chown -R snort.snort snort\*

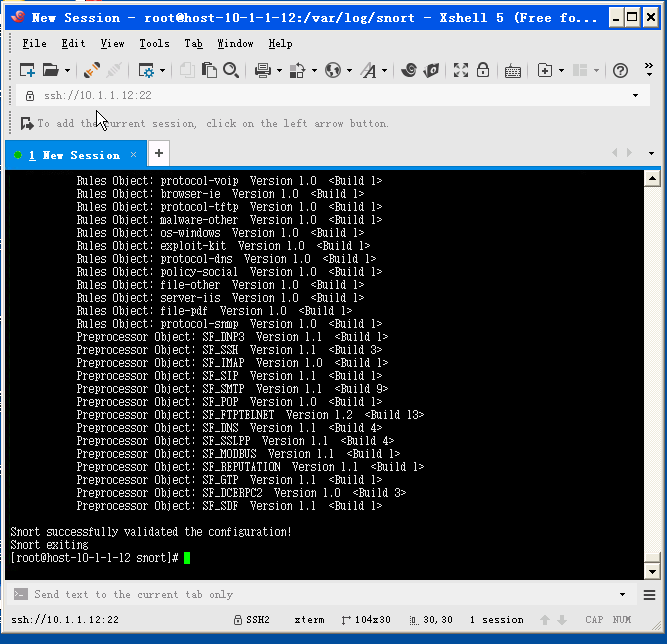
chmod -R 700 snort\*

chown -R snort.snort pkgconfig

chmod -R 700 pkgconfig

导出动态规则文件

snort -c /etc/snort/snort.conf --dump-dynamic-rules=/etc/snort/so\_rules



设定告警文件权限

touch /var/log/snort/alert

cd /var/log/snort

chown snort.snort alert

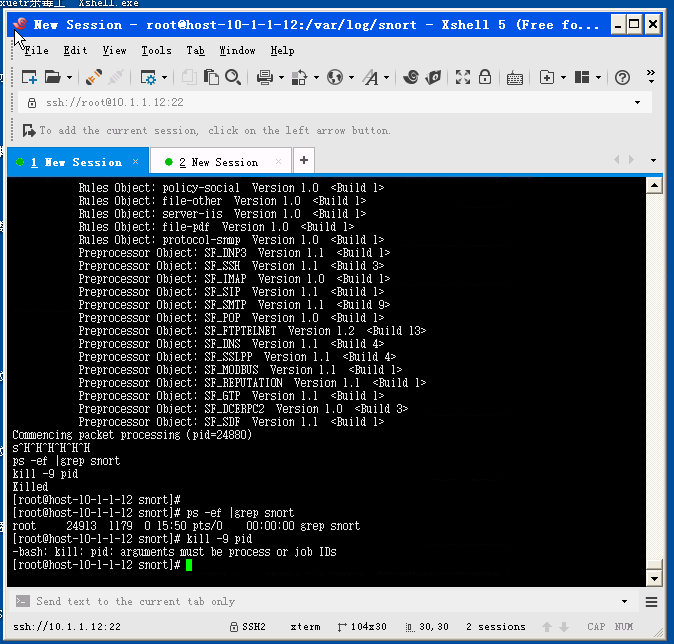
chmod 700 alert

snort -T -c /etc/snort/snort.conf -i eth0 检测配置文件是否有错误

启动 snort

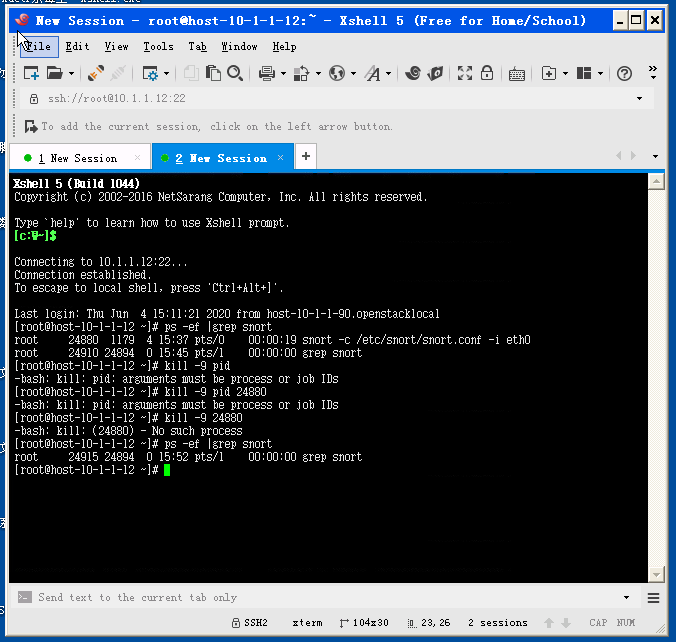
snort -c /etc/snort/snort.conf -i eth0

停止 snort



ps -ef |grep snort

kill -9 pid 号 即可结束进程



实验步骤二

安装配置 guardian

安装配置

cd /usr/local/src/snort+IPtables

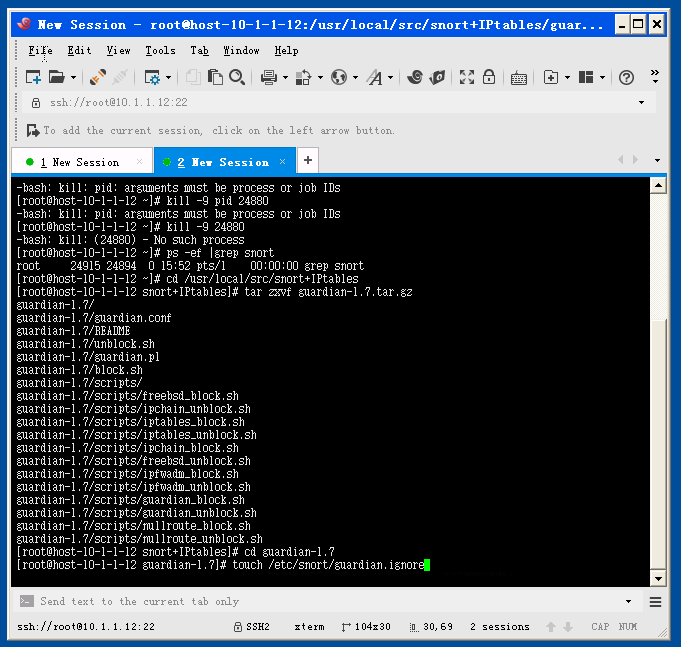
tar zxvf guardian-1.7.tar.gz

cd guardian-1.7

touch /etc/snort/guardian.ignore

touch /etc/snort/guardian.target

touch /var/log/snort/guardian.log

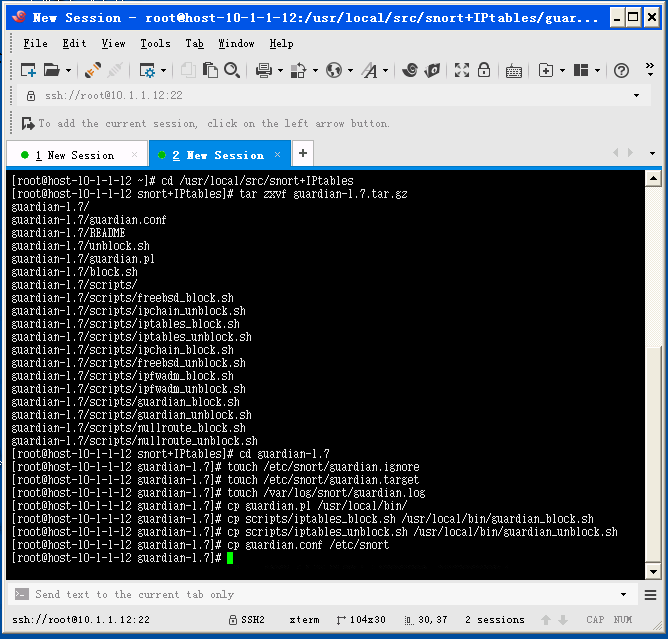


cp guardian.pl /usr/local/bin/

cp scripts/iptables\_block.sh /usr/local/bin/guardian\_block.sh

cp scripts/iptables\_unblock.sh /usr/local/bin/guardian\_unblock.sh

cp guardian.conf /etc/snort



编辑 guardian 配置文件

vi /etc/snort/guardian.conf

Interface eth0

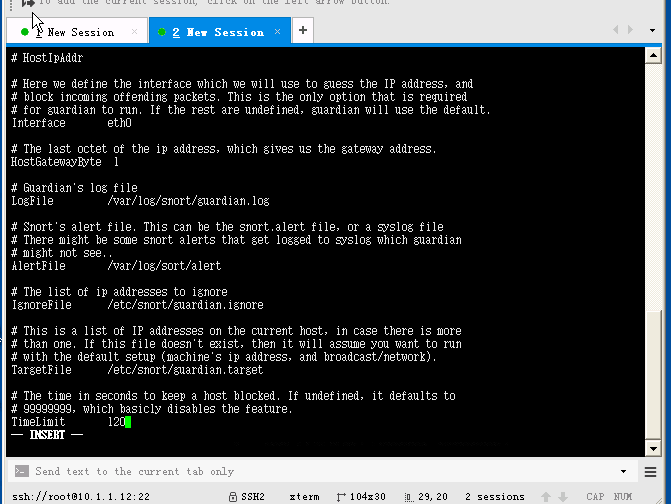
LogFile /var/log/snort/guardian.log

AlertFile /var/log/snort/alert //alert 文件的位置

IgnoreFile /etc/snort/guardian.ignore //白名单

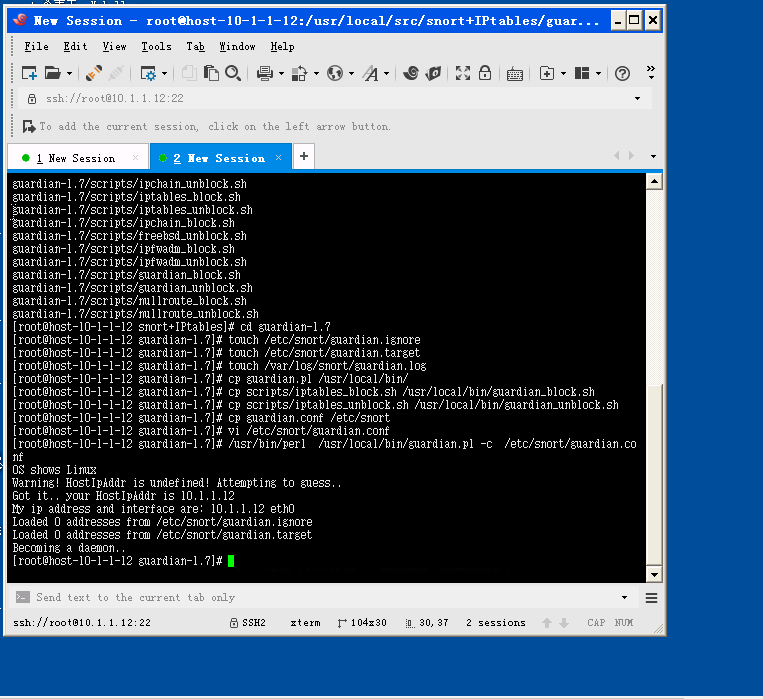
targetFile /etc/snort/guardian.target //黑名单

TimeLimit 120 //阻断时间,以秒为单位



guardian 启动

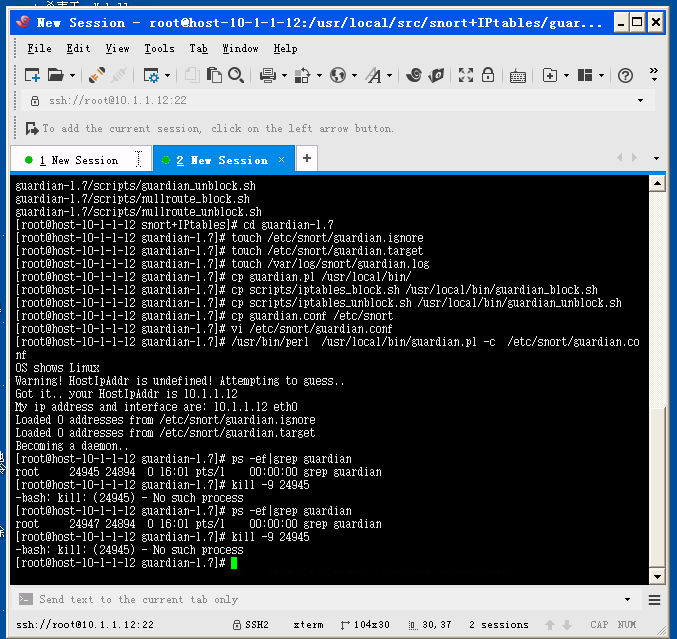
/usr/bin/perl /usr/local/bin/guardian.pl -c /etc/snort/guardian.conf



Guardian 停止

ps -ef|grep guardian

kill -9 pid 号即可杀死该进程



实验步骤三

联动测试（在执行以下操作之前，请先停止 guradian，否则会导致无法登陆试验机。）

Snort 与本地 iptables 联动

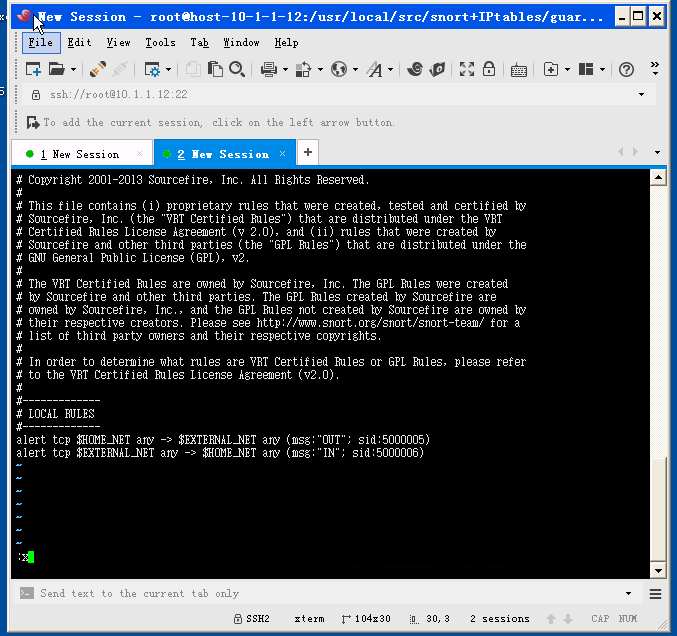
测试规则是否加载生效

vi /etc/snort/rules/local.rules

添加下面两条规则

alert tcp $HOME\_NET any -> $EXTERNAL\_NET any (msg:"OUT"; sid:5000005)

alert tcp $EXTERNAL\_NET any -> $HOME\_NET any (msg:"IN"; sid:5000006)



规则说明：

告警外网和内网之间的所有 tcp 流量，用来测试你的 snort.conf 配置是否有 问题

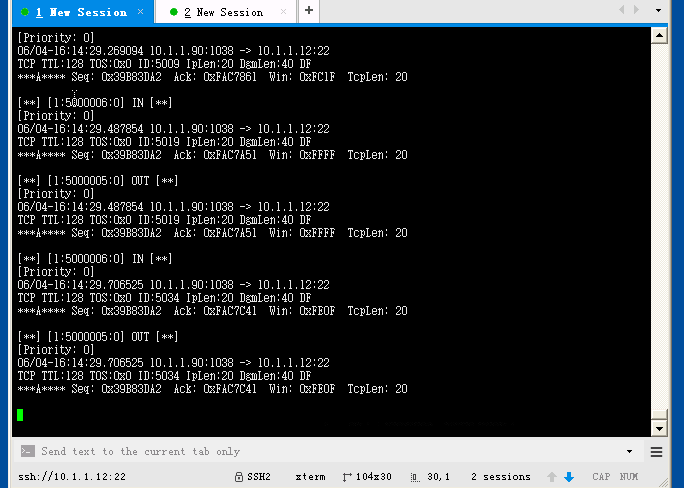
启动 snort，查看 alert 是否有日志

snort -c /etc/snort/snort.conf -i eth0

cd /var/log/snort

tail -f alert

如果可以看到日志，则表明 snort.conf 配置没有问题，可以继续下面的测 试，否则请检查 snort.conf 的配置，以及 snort 目录和文件权限设置。



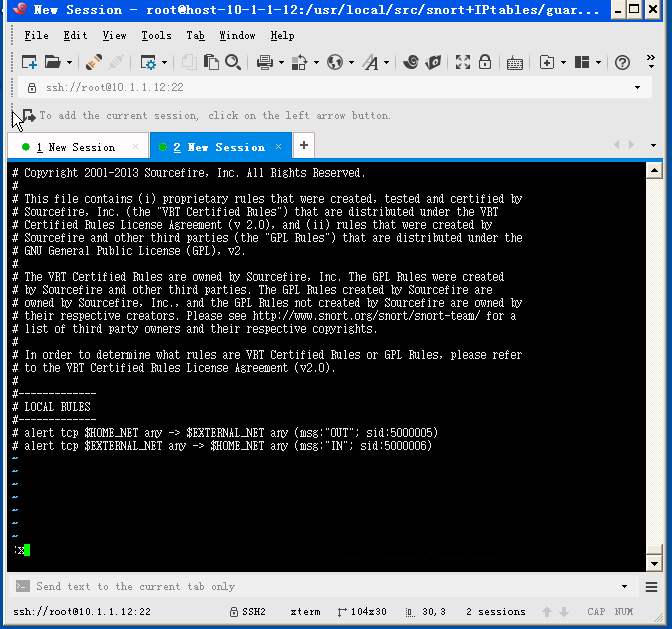
联动测试

如果之前的测试没有问题，请将/etc/snort/rules/local.rules 你所添加的两条规则 删除或注释掉。

vi /etc/snort/rules/local.rules

#alert tcp $HOME\_NET any -> $EXTERNAL\_NET any (msg:"OUT"; sid:5000005)

#alert tcp $EXTERNAL\_NET any -> $HOME\_NET any (msg:"IN"; sid:5000006)



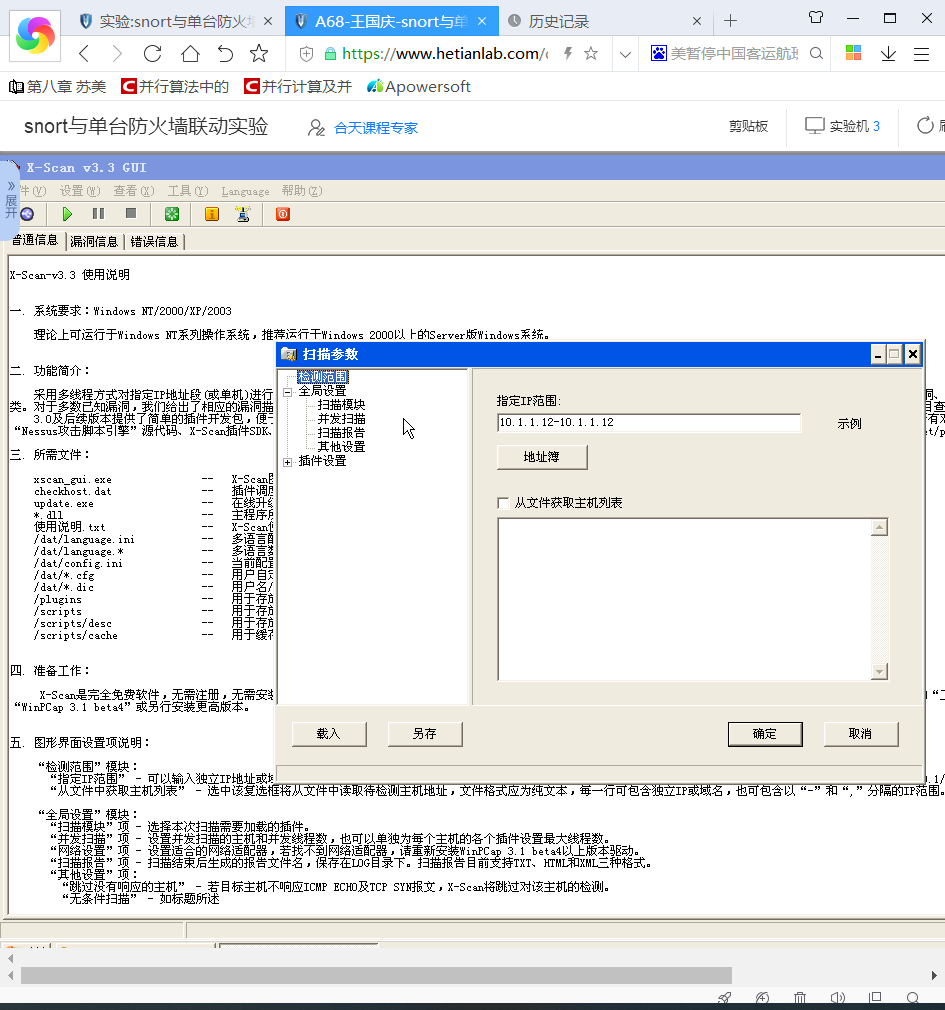
登录另一台 windows 测试主机(注意:如果用本机扫描很有可能导致 snort 主 机无法登陆)，打开浏览器，登录 http://tools.hetianlab.com/tools/X-Scan-v3.3cn.rar 下载 x-scan 并解压，双击文件夹

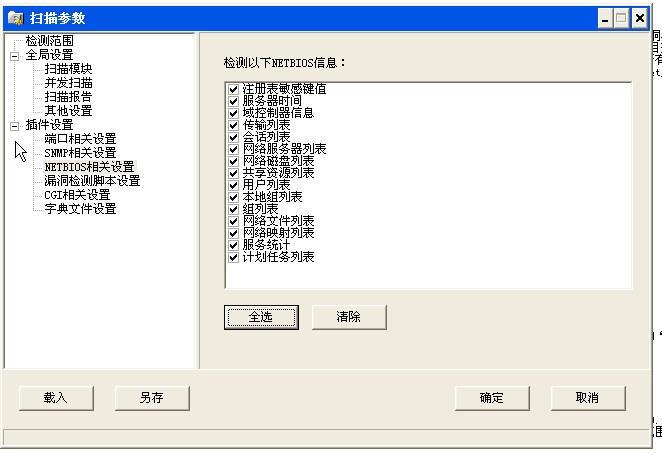


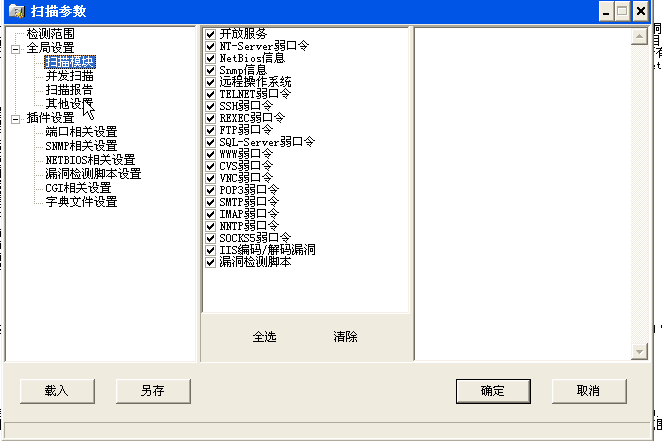
打开 x-scan

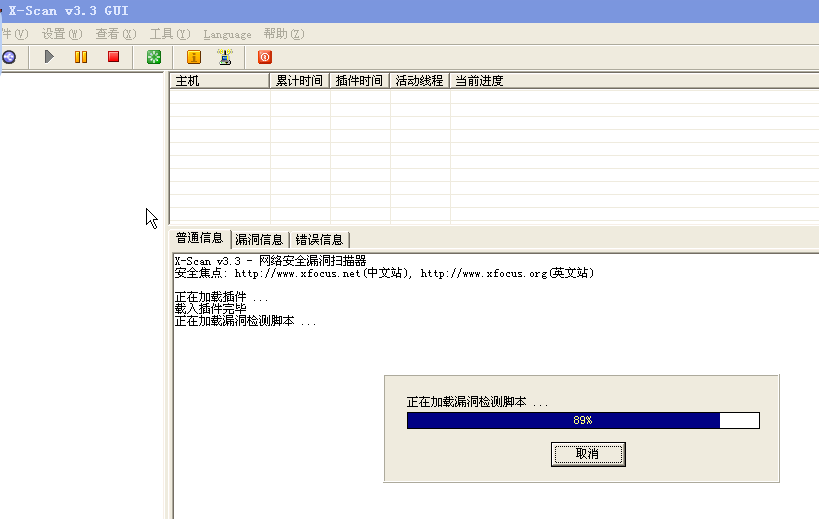
设置扫描参数

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。





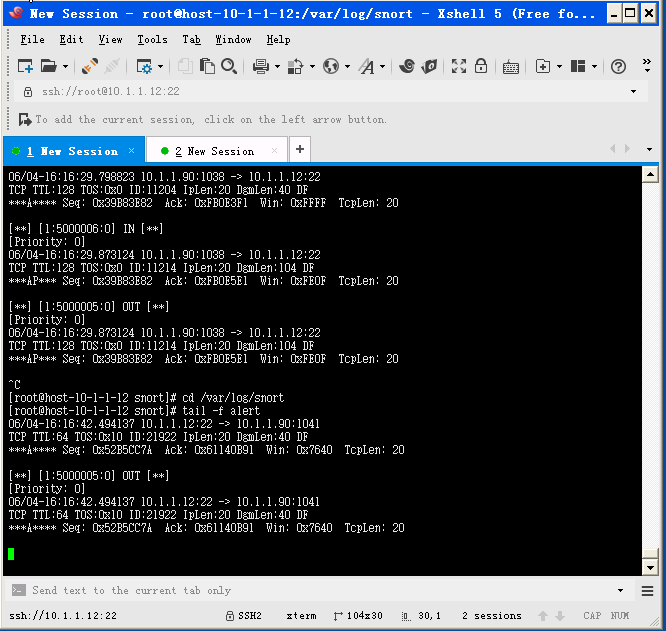




观察 alert 是否有告警日志

cd /var/log/snort

tail -f alert



检测到的扫描日志

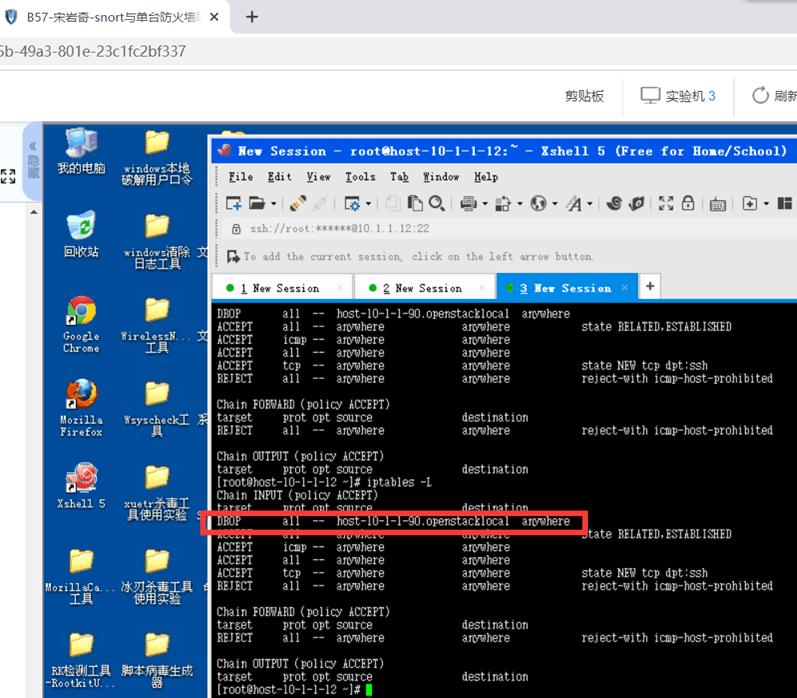
发现告警日志，启动 guardian 与 iptables 联动

/usr/bin/perl /usr/local/bin/guardian.pl -c /etc/snort/guardian.conf

在 snort 主机上执行 iptables -L 观察是否有规则加入，有规则即实验成功。

可以看到，执行了 block 脚本

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。

问题1：安装snort时 ./configure –enable-sourcefire命令错误

解决方法：根据群里同学的提醒，去除后面的参数，只保留./configure成功通过。

问题2：开始的时候修改var PREPROC\_RULE\_PATH /etc/snort/preproc\_rules

var WHITE\_LIST\_PATH /etc/snort/rules

var BLACK\_LIST\_PATH /etc/snort/rules找不到在哪里

解决方法：使用/字符串 来进行搜索，极大的提高搜索效率。

五、实验结果总结

*对实验结果进行分析，完成思考题目，总结实验的心得体会，并提出实验的改进意见。*

查阅相关资料了解 snortsam 插件，分析并简述 snortsam 与 guardian 的优缺点。

Snort：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。

Iptables：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。

你认为 snortsam 和 guardian 的这种联动方式有哪些弊端？

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，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

这次实验又有了之前做计算机系统实验的感觉，还顺便回顾了如何使用命令行编译，kill进程等等，又捡起了很多知识，同时对于信息安全收获了更多的知识。这次实验做的很有意义。