哈尔滨工业大学计算机学院

《网络攻击与防御》

实验报告

计算机科学与技术学院

计算机系网络教研室制

|  |  |
| --- | --- |
| 课程名称： | **信息安全概论** |
| 实验名称： | **snort与单台防火墙联动实验** |
| 指导教师： | 翟建宏 |
| 学生姓名： | 卢兑玧 |
| 组 号： | 9班2组 |
| 实验日期： | 2019-6-20 |
| 实验地点： |  |
| 实验成绩： |  |

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：

一、实验目的

本次实验所涉及并要求掌握的知识点。

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。

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。

四、实验过程与分析

根据具体实验，记录、整理相应命令、运行结果等，包括截图和文字说明。

详细记录在实验过程中发生的故障和问题，并进行故障分析，说明故障排除的过程及方法。

五、实验结果总结

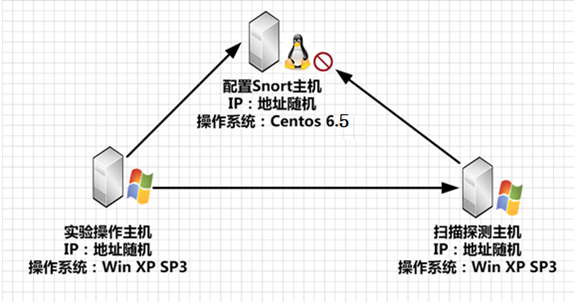
对实验结果进行分析，完成思考题目，总结实验的心得体会，并提出实验的改进意见。

六、附录

一、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(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

# Snort与guardian的安装和配置

# 1.  操作指导（实验过程中操作命令综述）

      Vim编辑器是linux下的一款相当于记事本的编辑器。

      命令：vi 某文件名即进入编辑状态，输入:i即可插入或删除文字

      按下ESC可退出编辑。

      shift键+冒号(:) 输入q! 即不保存退出输入x即可保存退出

# 2.  Snort安装与配置

**/\*\*\*\*\*\*\*\*\*\*更新系统、安装snort依赖包、下载snort、规则库\*\*\*\*\*\*\*\*\*/**

SSH登录centos6.5主机

      用户名：root

      password:123456

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

(外网下载十分缓慢，因此安装包已经提前下载到实验环境服务器（http://tools.hetianlab.com/tools/）中)

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

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)

如果在32位系统上，这样导入动态规则:

cp /etc/snort/so\_rules/precompiled/RHEL-6-0/i386/2.9\*/\*.so /usr/local /lib/snort\_dynamicrules/

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

如报错，很有可能是导入动态规则时未能导入对应系统架构的规则文件。

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

**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

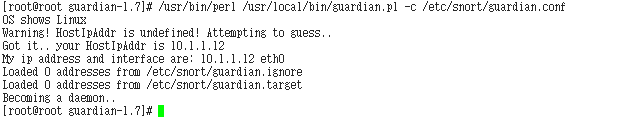
      targetFile   /etc/snort/guardian.target        //黑名单

      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

启动成功会出现如下示例：



Guardian 停止

      ps -ef|grep guardian

      kill -9 pid号即可杀死该进程

**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)

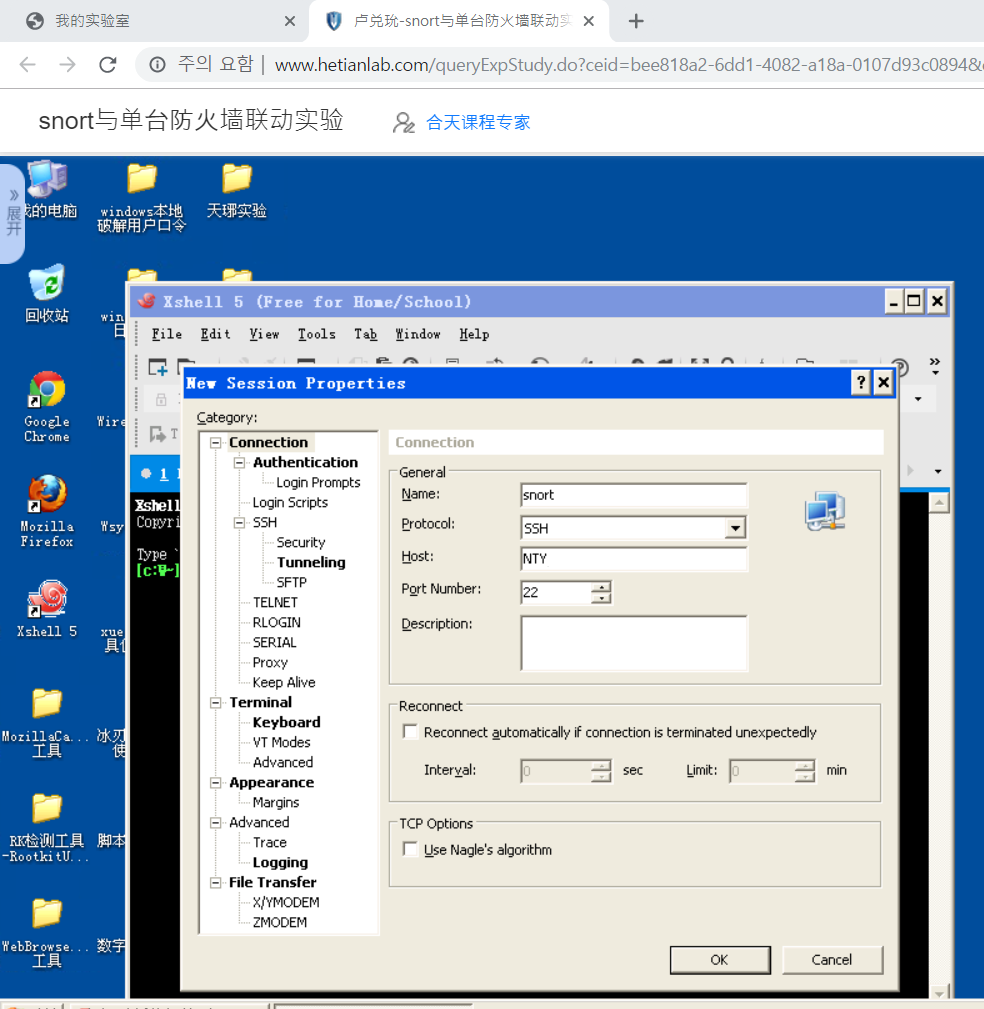
      #alert tcp $EXTERNAL\_NET any -> $HOME\_NET any (msg:"IN"; sid:5000006)

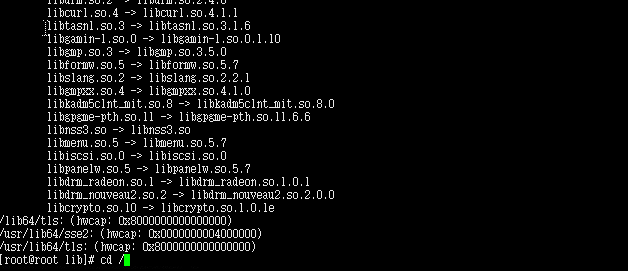
      登录另一台windows 测试主机(注意:如果用本机扫描很有可能导致snort主机无法登陆)，打开浏览器，登录http://tools.hetianlab.com/tools/X-Scan-v3.3-cn.rar下载x-scan并解压，双击文件夹

四、实验过程与分析

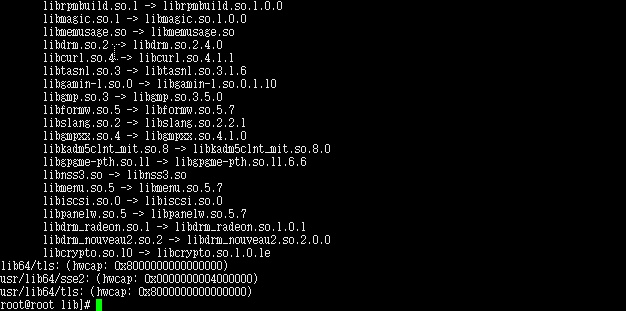
*根据具体实验，记录、整理相应命令、运行结果等，包括截图和文字说明。*

*详细记录在实验过程中发生的故障和问题，并进行故障分析，说明故障排除的过程及方法。*

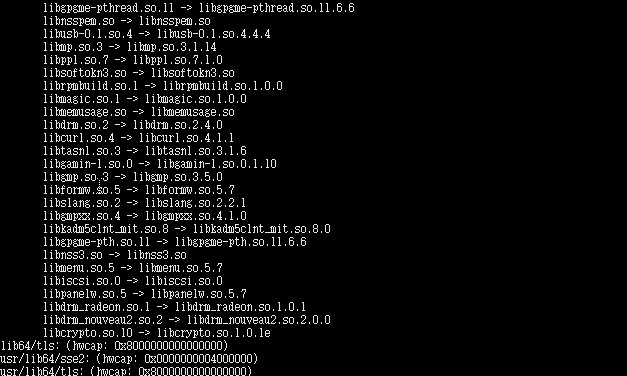




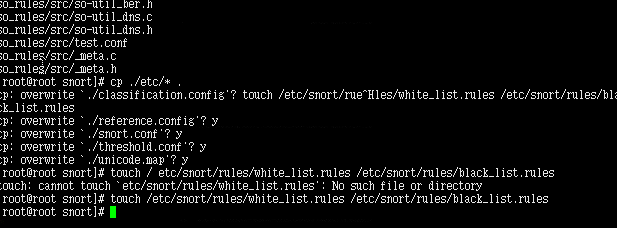
安装libdnet



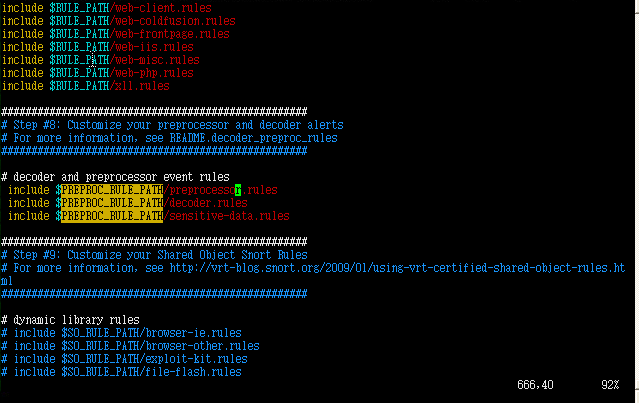
安装DAQ

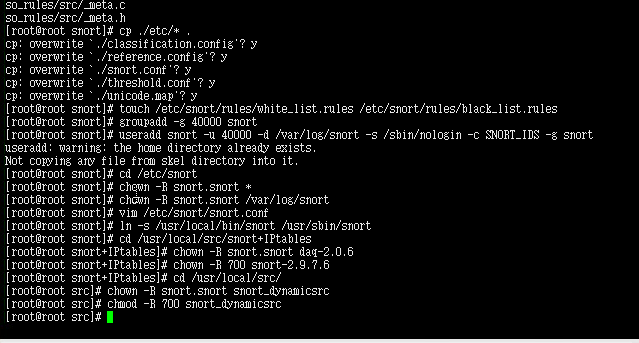


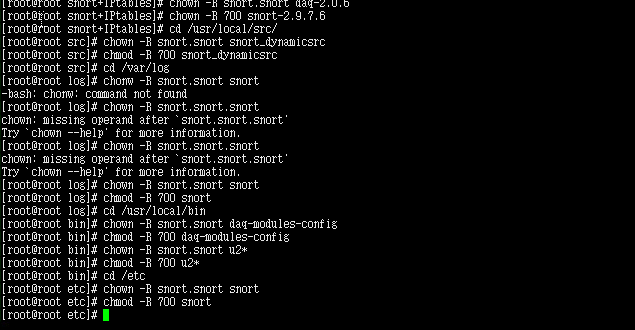
安装snort



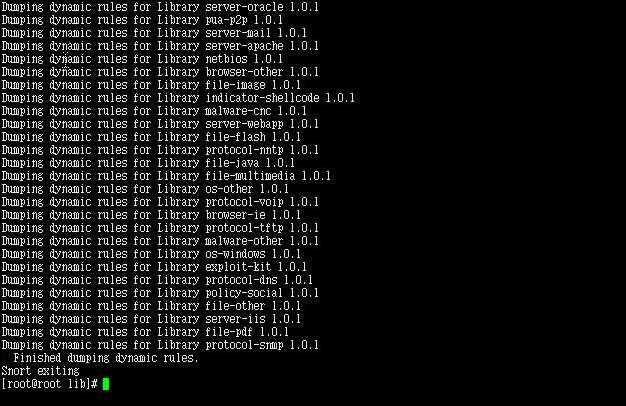
Snort配置



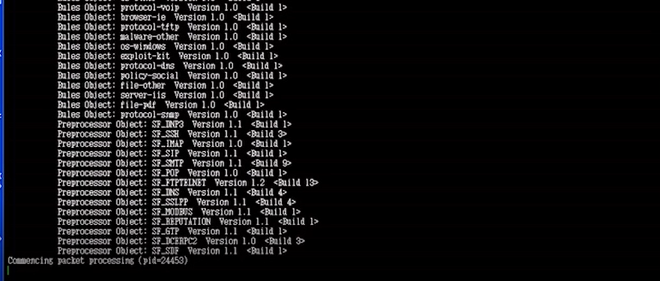




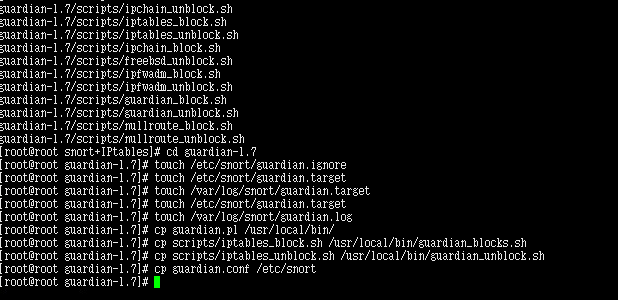
Snort配置



导出动态规则文件



启动snort

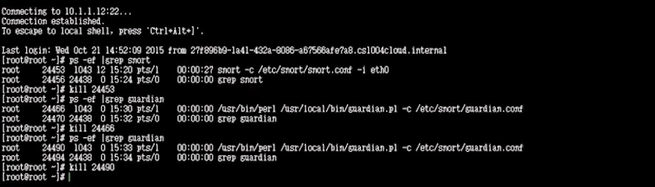


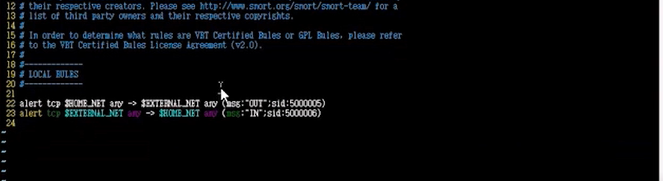
**安装配置guardian**

**安装配置**



**guardian启动**

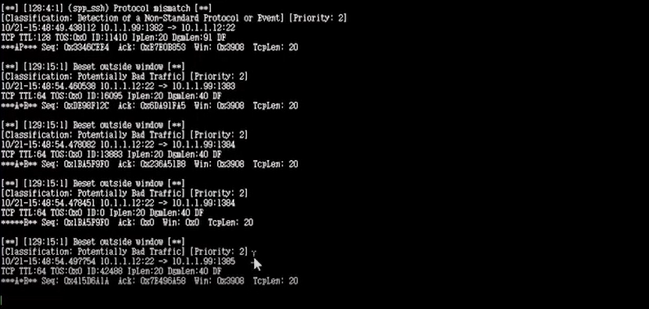




**测试规则是否加载生效**



**启动snort，查看alert是否有日志**



五、实验结果总结

利用 Guardian 实现 snort 与 iptables 联动 Guardian 是基于 snort 和 iptables 的一个主动防火墙，运行于后台。Guardian 分析 snort 的 报警日志 alert 文件（默认路径/var/log/snort/） ，根据一定的判断自动将某些恶意的 IP 加入 iptables 的输入链，将其数据报丢弃。当 guardian 退出时，它会将先前插入到 iptables 输入链 中的规则删除掉

首先我在计划在我的windows xp物理机上进行本次实验，但是装到snort时，我发现有一个小插件是不支持这个版本的，又加上当时物理机的磁盘存储逼近饱和，决定放弃物理机。

利用 snortsam 插件实现 snort 与 iptables 联动 SnortSam 是 snort 的入侵防范插件。它通过向 snort 规则添加新响应来工作，规则一旦触 发将会使防火墙或路由器发生变化。 这个变化通常在一段时间内阻塞或禁止来自或去往某个 特定的 IP 地址的流量。SnortSam 协同 Checkpoint Firewall-1 防火墙、Cisco PIX 防火墙和 iptables 防火墙等一起工作.