注 1：以下只是最初级的配置，如果有问题讨论~

注 2：安装前请先把 centos 自带的 php,mysql,httpd 全部删除

Yum remove php,mysql,httpd

环境: centos5

硬盘空间配 10G 以上

1. 安装 Zlib,libpcap,libxml2,libpng,gd,jpeg

   ./configure

   Make

   Make install

2. 安装 mysql

   # groupadd mysql

   # useradd –g mysql mysql

   # tar –zxvf mysql

   # cd mysql

   # ./configure –prefix=/usr/local/mysql –with-charset=gb2312

   # make

   # make install

   # cp support-files/my-medium.cnf /etc/my.cnf

   # cd /usr/local/mysql

   # bin/mysql_install_db –user=mysql

   # chown –R root .

   # chown –R mysql var

   # chgrp –R mysql .

   # bin/mysqld_safe –user=mysql &

   #gedit /etc/ld.so.conf

   在文件最后加入 2 行:

   /usr/local/mysql/lib/mysql

   /usr/local/lib

   # ldconfig

3. 安装 DBD-mysql

   # cd DBD-mysql

   # export LANG=C

   # perl Makefile.PL \

   # --libs="-L/usr/local/mysql/lib/mysql -lmysqlclient -lz" \

   # --cflags=-I/usr/local/mysql/include/mysql \

   # --testhost=127.0.0.1 \

   # --mysql_config=/usr/local/mysql/bin/mysql_config

   # make

   # make install

4. 设置 mysql 自启动

```
# cp /usr/local/mysql/share/mysql/mysql.server /etc/init.d/mysql
# chmod 755 /etc/init.d/mysql
# cd /etc/rc3.d
# ln -s /etc/init.d/mysql S85mysql
# ln -s /etc/init.d/mysql K85mysql
# cd /etc/rc5.d
# ln -s /etc/init.d/mysql S85mysql
# ln -s /etc/init.d/mysql K85mysql
```

5. 安装 Apache

```
# mkdir /www
# cd httpd-2.2.14
# ./configure --prefix=/www   --enable-so
# make
# make install
```

6. 安装 php

```
# mkdir /www/php
# cd php
# ./configure \
# --prefix=/www/php \
# --with-apxs2=/www/bin/apxs \
# --with-libxml-dir=/usr/local/lib \
# --with-zlib \
# --with-zlib-dir=/usr/local/lib \
# --with-gd \
# --with-png-dir=某个目录/libpng-1.2.40 \
# --with-jpeg-dir=某个目录/jpeg-7 \
# --with-mysql=/usr/local/mysql \
# --with-mysqli=/usr/local/mysql/bin/mysql_config \
# --enable-mbstring \
# --enable-soap \
# --enable-sockets
# make
# make install
# cp php.ini-dist /www/php/php.ini
# gedit /www/conf/httpd.conf
```

在最后加入 AddType application/x-httpd-php .php

7. 设置 apache 自启动

cp /www/bin/apachectl /etc/init.d/httpd
cd /etc/rc3.d
ln -s /etc/init.d/httpd S85httpd
ln -s /etc/init.d/httpd K85httpd
cd /etc/rc5.d
ln -s /etc/init.d/httpd S85httpd
ln -s /etc/init.d/httpd K85httpd

8. 测试 Apache 和 php

打开浏览器，输入 http://localhost/
如出现 it works，则 Apache 正常
在/www/htdocs 下建立文件 test.php
gedit /www/htdocs/test.php
写入：

```
<?php
phpinfo();
?>
```

打开浏览器，输入 http://localhost/test.php,出现 php 信息(这里显示了 php 的所有配置信息)，则说明 PHP 成功.

9. 安装 pcre

./configure
make
make install

10. 安装 snort

mkdir /etc/snort
mkdir /var/log/snort
cd snort-2.8.3.1
./configure --with-mysql=/usr/local/mysql
make
make install
#mkdir /var/lib/mysql
ln –s /tmp/mysql.sock /var/lib/mysql/mysql.sock (这里要建立一个套接字的软连接)

11. 安装规则

cd 某个目录/ snortrules-snapshot-2[1].8/rules

cp * /etc/snort

cd ../etc

cp snort.conf /etc/snort

cp *.config /etc/snort

cp *.map /etc/snort

上面用到了两个规则，因为在使用 snortrules-snapshot-2[1].8.tar.gz 解压的规则时，

include $RULE_PATH/web-client.rules

include $RULE_PATH/netbios.rules

这两个规则编译有问题，所以解压 snortrules-snapshot-CURRENT[1].tar.gz

在/root/so_rules 文件夹下

cp /root/so_rules/netbios.rules /etc/snort

cp /root/so_rules/web_client.rules /etc/snort

include $RULE_PATH/mysql.rules 也有同样的问题，在/etc/snort/snort.conf 中屏蔽此规则：

#include $RULE_PATH/mysql.rules

(上面就是把源码包里 rules 文件夹下的内容和 etc 文件下的配置文件复制到/etc/snort)

12. 修改/etc/snort/snort.conf 文件（修改的地方红色标注）

修改如下：

```
#--------------------------------------------------
#   http://www.snort.org      Snort 2.8.3.1 Ruleset
#      Contact: snort-sigs@lists.sourceforge.net
#--------------------------------------------------
# $Id$
#
##################################################
# This file contains a sample snort configuration.
# You can take the following steps to create your own custom configuration:
#
#   1) Set the variables for your network
#   2) Configure dynamic loaded libraries
#   3) Configure preprocessors
#   4) Configure output plugins
#   5) Add any runtime config directives
#   6) Customize your rule set
#
##################################################
# Step #1: Set the network variables:
#
# You must change the following variables to reflect your local network. The
# variable is currently setup for an RFC 1918 address space.
```

```
#
# You can specify it explicitly as:
#
# var HOME_NET 10.1.1.0/24
#
# or use global variable $<interfacename>_ADDRESS which will be always
# initialized to IP address and netmask of the network interface which you run
# snort at.  Under Windows, this must be specified as
# $(<interfacename>_ADDRESS), such as:
# $(\Device\Packet_{12345678-90AB-CDEF-1234567890AB}_ADDRESS)
#
# var HOME_NET $eth0_ADDRESS
#
# You can specify lists of IP addresses for HOME_NET
# by separating the IPs with commas like this:
#
# var HOME_NET [10.1.1.0/24,192.168.1.0/24]
#
# MAKE SURE YOU DON'T PLACE ANY SPACES IN YOUR LIST!
#
# or you can specify the variable to be any IP address
# like this:
var HOME_NET 192.168.80.0/24(这个设置要检测的网段)
# Set up the external network addresses as well.  A good start may be "any"
var EXTERNAL_NET any
# Configure your server lists.  This allows snort to only look for attacks to
# systems that have a service up.  Why look for HTTP attacks if you are not
# running a web server?  This allows quick filtering based on IP addresses
# These configurations MUST follow the same configuration scheme as defined
# above for $HOME_NET.
# List of DNS servers on your network
var DNS_SERVERS $HOME_NET
# List of SMTP servers on your network
var SMTP_SERVERS $HOME_NET
# List of web servers on your network
var HTTP_SERVERS $HOME_NET
# List of sql servers on your network
var SQL_SERVERS $HOME_NET
# List of telnet servers on your network
var TELNET_SERVERS $HOME_NET
# List of snmp servers on your network
var SNMP_SERVERS $HOME_NET
# Configure your service ports.  This allows snort to look for attacks destined
# to a specific application only on the ports that application runs on.  For
```

```
# example, if you run a web server on port 8081, set your HTTP_PORTS variable
# like this:
#
# portvar HTTP_PORTS 8081
#
# Ports you run web servers on
portvar HTTP_PORTS 80
# NOTE:  If you wish to define multiple HTTP ports, use the portvar
# syntax to represent lists of ports and port ranges.  Examples:
## portvar HTTP_PORTS [80,8080]
## portvar HTTP_PORTS [80,8000:8080]
# And only include the rule that uses $HTTP_PORTS once.
#
# The pre-2.8.0 approach of redefining the variable to a different port and
# including the rules file twice is obsolete.  See README.variables for more
# details.
# Ports you want to look for SHELLCODE on.
portvar SHELLCODE_PORTS !80
# Ports you might see oracle attacks on
portvar ORACLE_PORTS 1521
# other variables
#
# AIM servers.  AOL has a habit of adding new AIM servers, so instead of
# modifying the signatures when they do, we add them to this list of servers.
var AIM_SERVERS
[64.12.24.0/23,64.12.28.0/23,64.12.161.0/24,64.12.163.0/24,64.12.200.0/24,205.188.3.
0/24,205.188.5.0/24,205.188.7.0/24,205.188.9.0/24,205.188.153.0/24,205.188.179.0/2
4,205.188.248.0/24]
# Path to your rules files (this can be a relative path)
# Note for Windows users:  You are advised to make this an absolute path,
# such as:  c:\snort\rules
var RULE_PATH /etc/snort
var PREPROC_RULE_PATH 某个文件夹/snort-2.8.3.1/preproc_rules
# Configure the snort decoder
# ============================
#
# Snort's decoder will alert on lots of things such as header
# truncation or options of unusual length or infrequently used tcp options
#
#
# Stop generic decode events:
#
# config disable_decode_alerts
#
```

```
# Stop Alerts on experimental TCP options
#
# config disable_tcpopt_experimental_alerts
#
# Stop Alerts on obsolete TCP options
#
# config disable_tcpopt_obsolete_alerts
#
# Stop Alerts on T/TCP alerts
#
# In snort 2.0.1 and above, this only alerts when a TCP option is detected
# that shows T/TCP being actively used on the network.  If this is normal
# behavior for your network, disable the next option.
#
# config disable_tcpopt_ttcp_alerts
#
# Stop Alerts on all other TCPOption type events:
#
# config disable_tcpopt_alerts
#
# Stop Alerts on invalid ip options
#
# config disable_ipopt_alerts
#
# Alert if value in length field (IP, TCP, UDP) is greater than the
# actual length of the captured portion of the packet that the length
# is supposed to represent:
#
# config enable_decode_oversized_alerts
#
# Same as above, but drop packet if in Inline mode -
# enable_decode_oversized_alerts must be enabled for this to work:
#
# config enable_decode_oversized_drops
#
# Configure the detection engine
# ==============================
#
# Use a different pattern matcher in case you have a machine with very limited
# resources:
#
# config detection: search-method lowmem
# Configure Inline Resets
# =======================
```

```
#
# If running an iptables firewall with snort in InlineMode() we can now
# perform resets via a physical device. We grab the indev from iptables
# and use this for the interface on which to send resets. This config
# option takes an argument for the src mac address you want to use in the
# reset packet.  This way the bridge can remain stealthy. If the src mac
# option is not set we use the mac address of the indev device. If we
# don't set this option we will default to sending resets via raw socket,
# which needs an ipaddress to be assigned to the int.
#
# config layer2resets: 00:06:76:DD:5F:E3
##################################################
# Step #2: Configure dynamic loaded libraries
#
# If snort was configured to use dynamically loaded libraries,
# those libraries can be loaded here.
#
# Each of the following configuration options can be done via
# the command line as well.
#
# Load all dynamic preprocessors from the install path
# (same as command line option --dynamic-preprocessor-lib-dir)
#
dynamicpreprocessor directory /usr/local/lib/snort_dynamicpreprocessor/
#
# Load a specific dynamic preprocessor library from the install path
# (same as command line option --dynamic-preprocessor-lib)
#
# dynamicpreprocessor file /usr/local/lib/snort_dynamicpreprocessor/libdynamicexample.so
#
# Load a dynamic engine from the install path
# (same as command line option --dynamic-engine-lib)
#
dynamicengine /usr/local/lib/snort_dynamicengine/libsf_engine.so
#
# Load all dynamic rules libraries from the install path
# (same as command line option --dynamic-detection-lib-dir)
#
# dynamicdetection directory /usr/local/lib/snort_dynamicrule/
#
# Load a specific dynamic rule library from the install path
# (same as command line option --dynamic-detection-lib)
#
```

```
# dynamicdetection file /usr/local/lib/snort_dynamicrule/libdynamicexamplerule.so
#
##################################################
# Step #3: Configure preprocessors
#
# General configuration for preprocessors is of
# the form
# preprocessor <name_of_processor>: <configuration_options>
# Configure Flow tracking module
# ------------------------------
#
# The Flow tracking module is meant to start unifying the state keeping
# mechanisms of snort into a single place. Right now, only a portscan detector
# is implemented but in the long term,  many of the stateful subsystems of
# snort will be migrated over to becoming flow plugins. This must be enabled
# for flow-portscan to work correctly.
#
# See README.flow for additional information
#
#preprocessor flow: stats_interval 0 hash 2
# frag3: Target-based IP defragmentation
# --------------------------------------
#
# Frag3 is a brand new IP defragmentation preprocessor that is capable of
# performing "target-based" processing of IP fragments.  Check out the
# README.frag3 file in the doc directory for more background and configuration
# information.
#
# Frag3 configuration is a two step process, a global initialization phase
# followed by the definition of a set of defragmentation engines.
#
# Global configuration defines the number of fragmented packets that Snort can
# track at the same time and gives you options regarding the memory cap for the
# subsystem or, optionally, allows you to preallocate all the memory for the
# entire frag3 system.
#
# frag3_global options:
#   max_frags: Maximum number of frag trackers that may be active at once.
#            Default value is 8192.
#   memcap: Maximum amount of memory that frag3 may access at any given time.
#          Default value is 4MB.
#   prealloc_frags: Maximum number of individual fragments that may be processed
#                at once.  This is instead of the memcap system, uses static
#                allocation to increase performance.  No default value.  Each
```

```
#                   preallocated fragment typically eats ~1550 bytes.  However,
#                   the exact amount is determined by the snaplen, and this can
#                   go as high as 64K so beware!
#
# Target-based behavior is attached to an engine as a "policy" for handling
# overlaps and retransmissions as enumerated in the Paxson paper.  There are
# currently five policy types available: "BSD", "BSD-right", "First", "Linux"
# and "Last".  Engines can be bound to standard Snort CIDR blocks or
# IP lists.
#
# frag3_engine options:
#   timeout: Amount of time a fragmented packet may be active before expiring.
#            Default value is 60 seconds.
#   ttl_limit: Limit of delta allowable for TTLs of packets in the fragments.
#              Based on the initial received fragment TTL.
#   min_ttl: Minimum acceptable TTL for a fragment, frags with TTLs below this
#            value will be discarded.  Default value is 0.
#   detect_anomalies: Activates frag3's anomaly detection mechanisms.
#   policy: Target-based policy to assign to this engine.  Default is BSD.
#   bind_to: IP address set to bind this engine to.  Default is all hosts.
#
# Frag3 configuration example:
#preprocessor frag3_global: max_frags 65536, prealloc_frags 65536
#preprocessor frag3_engine: policy linux \
#                     bind_to [10.1.1.12/32,10.1.1.13/32] \
#                     detect_anomalies
#preprocessor frag3_engine: policy first \
#                     bind_to 10.2.1.0/24 \
#                     detect_anomalies
#preprocessor frag3_engine: policy last \
#                     bind_to 10.3.1.0/24
#preprocessor frag3_engine: policy bsd
preprocessor frag3_global: max_frags 65536
preprocessor frag3_engine: policy first detect_anomalies

# stream4: stateful inspection/stream reassembly for Snort
#---------------------------------------------------------------------
# Use in concert with the -z [all|est] command line switch to defeat stick/snot
# against TCP rules.  Also performs full TCP stream reassembly, stateful
# inspection of TCP streams, etc.  Can statefully detect various portscan
# types, fingerprinting, ECN, etc.
# stateful inspection directive
# no arguments loads the defaults (timeout 30, memcap 8388608)
# options (options are comma delimited):
```

```
#   detect_scans - stream4 will detect stealth portscans and generate alerts
#                  when it sees them when this option is set
#   detect_state_problems - detect TCP state problems, this tends to be very
#                           noisy because there are a lot of crappy ip stack
#                           implementations out there
#
#   disable_evasion_alerts - turn off the possibly noisy mitigation of
#                            overlapping sequences.
#
#   ttl_limit [number]      - differential of the initial ttl on a session versus
#                             the normal that someone may be playing games.
#                             Routing flap may cause lots of false positives.
#
#   keepstats [machine|binary] - keep session statistics, add "machine" to
#                        get them in a flat format for machine reading, add
#                        "binary" to get them in a unified binary output
#                        format
#   noinspect - turn off stateful inspection only
#   timeout [number] - set the session timeout counter to [number] seconds,
#                      default is 30 seconds
#   max_sessions [number] - limit the number of sessions stream4 keeps
#                        track of
#   memcap [number] - limit stream4 memory usage to [number] bytes (does
#                     not include session tracking, which is set by the
#                     max_sessions option)
#   log_flushed_streams - if an event is detected on a stream this option will
#                        cause all packets that are stored in the stream4
#                        packet buffers to be flushed to disk.   This only
#                        works when logging in pcap mode!
#   server_inspect_limit [bytes] - Byte limit on server side inspection.
#   enable_udp_sessions - turn on tracking of "sessions" over UDP.   Requires
#                        configure --enable-stream4udp.   UDP sessions are
#                        only created when there is a rule for the sender or
#                        responder that has a flow or flowbits keyword.
#   max_udp_sessions [number] - limit the number of simultaneous UDP sessions
#                               to track
#   udp_ignore_any - Do not inspect UDP packets unless there is a port specific
#                    rule for a given port.   This is a performance improvement
#                    and turns off inspection for udp xxx any -> xxx any rules
#   cache_clean_sessions [number] - Cleanup the session cache by number sessions
#                                   at a time.   The larger the value, the
#                                   more sessions are purged from the cache when
#                                   the session limit or memcap is reached.
#                                   Defaults to 5.
```

```
#
#
#
# Stream4 uses Generator ID 111 and uses the following SIDS
# for that GID:
#  SID     Event description
# -----   -------------------
#   1       Stealth activity
#   2       Evasive RST packet
#   3       Evasive TCP packet retransmission
#   4       TCP Window violation
#   5       Data on SYN packet
#   6       Stealth scan: full XMAS
#   7       Stealth scan: SYN-ACK-PSH-URG
#   8       Stealth scan: FIN scan
#   9       Stealth scan: NULL scan
#   10      Stealth scan: NMAP XMAS scan
#   11      Stealth scan: Vecna scan
#   12      Stealth scan: NMAP fingerprint scan stateful detect
#   13      Stealth scan: SYN-FIN scan
#   14      TCP forward overlap
#preprocessor stream4: disable_evasion_alerts
# tcp stream reassembly directive
# no arguments loads the default configuration
#   Only reassemble the client,
#   Only reassemble the default list of ports (See below),
#   Give alerts for "bad" streams
#
# Available options (comma delimited):
#   clientonly - reassemble traffic for the client side of a connection only
#   serveronly - reassemble traffic for the server side of a connection only
#   both - reassemble both sides of a session
#   noalerts - turn off alerts from the stream reassembly stage of stream4
#   ports [list] - use the space separated list of ports in [list], "all"
#                  will turn on reassembly for all ports, "default" will turn
#                  on reassembly for ports 21, 23, 25, 42, 53, 80, 110,
#                  111, 135, 136, 137, 139, 143, 445, 513, 514, 1433, 1521,
#                  2401, and 3306
#   favor_old - favor an old segment (based on sequence number) over a new one.
#               This is the default.
#   favor_new - favor an new segment (based on sequence number) over an old one.
#   overlap_limit [number] - limit on overlaping segments for a session.
#   flush_on_alert - flushes stream when an alert is generated for a session.
#   flush_behavior [mode] -
```

```
#           default       - use old static flushpoints (default)
#           large_window - use new larger static flushpoints
#           random        - use random flushpoints defined by flush_base,
#                           flush_seed and flush_range
#    flush_base [number] - lowest allowed random flushpoint (512 by default)
#    flush_range [number] - number is the space within which random flushpoints
#                           are generated (default 1213)
#    flush_seed [number] - seed for the random number generator, defaults to
#                          Snort PID + time
#
# Using the default random flushpoints, the smallest flushpoint is 512,
# and the largest is 1725 bytes.
#preprocessor stream4_reassemble
# stream5: Target Based stateful inspection/stream reassembly for Snort
# ---------------------------------------------------------------------
# Stream5 is a target-based stream engine for Snort.   Its functionality
# replaces that of Stream4.   Consequently, BOTH Stream4 and Stream5
# cannot be used simultaneously.   Comment out the stream4 configurations
# above to use Stream5.
#
# See README.stream5 for details on the configuration options.
#
# Example config (that emulates Stream4 with UDP support compiled in)
preprocessor stream5_global: max_tcp 8192, track_tcp yes, \
                             track_udp no
preprocessor stream5_tcp: policy first, use_static_footprint_sizes
# preprocessor stream5_udp: ignore_any_rules

# Performance Statistics
# ----------------------
# Documentation for this is provided in the Snort Manual.   You should read it.
# It is included in the release distribution as doc/snort_manual.pdf
#
# preprocessor perfmonitor: time 300 file /var/snort/snort.stats pktcnt 10000
# http_inspect: normalize and detect HTTP traffic and protocol anomalies
#
# lots of options available here. See doc/README.http_inspect.
# unicode.map should be wherever your snort.conf lives, or given
# a full path to where snort can find it.
preprocessor http_inspect: global \
    iis_unicode_map unicode.map 1252
preprocessor http_inspect_server: server default \
    profile all ports { 80 8080 8180 } oversize_dir_length 500
```

```
#
#  Example unique server configuration
#
#preprocessor http_inspect_server: server 1.1.1.1 \
#    ports { 80 3128 8080 } \
#    server_flow_depth 0 \
#    ascii no \
#    double_decode yes \
#    non_rfc_char { 0x00 } \
#    chunk_length 500000 \
#    non_strict \
#    oversize_dir_length 300 \
#    no_alerts

# rpc_decode: normalize RPC traffic
# --------------------------------
# RPC may be sent in alternate encodings besides the usual 4-byte encoding
# that is used by default. This plugin takes the port numbers that RPC
# services are running on as arguments - it is assumed that the given ports
# are actually running this type of service. If not, change the ports or turn
# it off.
# The RPC decode preprocessor uses generator ID 106
#
# arguments: space separated list
# alert_fragments - alert on any rpc fragmented TCP data
# no_alert_multiple_requests - don't alert when >1 rpc query is in a packet
# no_alert_large_fragments - don't alert when the fragmented
#                            sizes exceed the current packet size
# no_alert_incomplete - don't alert when a single segment
#                       exceeds the current packet size
preprocessor rpc_decode: 111 32771
# bo: Back Orifice detector
# -------------------------
# Detects Back Orifice traffic on the network.
#
# arguments:
#   syntax:
#     preprocessor bo: noalert { client | server | general | snort_attack } \
#                      drop    { client | server | general | snort_attack }
#   example:
#     preprocessor bo: noalert { general server } drop { snort_attack }
#
#
# The Back Orifice detector uses Generator ID 105 and uses the
```

```
# following SIDS for that GID:
#   SID     Event description
# -----   -------------------
#    1       Back Orifice traffic detected
#    2       Back Orifice Client Traffic Detected
#    3       Back Orifice Server Traffic Detected
#    4       Back Orifice Snort Buffer Attack
preprocessor bo
# ftp_telnet: FTP & Telnet normalizer, protocol enforcement and buff overflow
# ----------------------------------------------------------------------------
# This preprocessor normalizes telnet negotiation strings from telnet and
# ftp traffic.  It looks for traffic that breaks the normal data stream
# of the protocol, replacing it with a normalized representation of that
# traffic so that the "content" pattern matching keyword can work without
# requiring modifications.
#
# It also performs protocol correctness checks for the FTP command channel,
# and identifies open FTP data transfers.
#
# FTPTelnet has numerous options available, please read
# README.ftptelnet for help configuring the options for the global
# telnet, ftp server, and ftp client sections for the protocol.
#####
# Per Step #2, set the following to load the ftptelnet preprocessor
# dynamicpreprocessor file <full path to libsf_ftptelnet_preproc.so>
# or use commandline option
# --dynamic-preprocessor-lib <full path to libsf_ftptelnet_preproc.so>
preprocessor ftp_telnet: global \
  encrypted_traffic yes \
  inspection_type stateful
preprocessor ftp_telnet_protocol: telnet \
  normalize \
  ayt_attack_thresh 200
# This is consistent with the FTP rules as of 18 Sept 2004.
# CWD can have param length of 200
# MODE has an additional mode of Z (compressed)
# Check for string formats in USER & PASS commands
# Check nDTM commands that set modification time on the file.
preprocessor ftp_telnet_protocol: ftp server default \
  def_max_param_len 100 \
  alt_max_param_len 200 { CWD } \
  cmd_validity MODE < char ASBCZ > \
  cmd_validity MDTM < [ date nnnnnnnnnnnnnn[.n[n[n]]] ] string > \
  chk_str_fmt { USER PASS RNFR RNTO SITE MKD } \
```

```
   telnet_cmds yes \
   data_chan
preprocessor ftp_telnet_protocol: ftp client default \
   max_resp_len 256 \
   bounce yes \
   telnet_cmds yes
# smtp: SMTP normalizer, protocol enforcement and buffer overflow
# ------------------------------------------------------------------------
# This preprocessor normalizes SMTP commands by removing extraneous spaces.
# It looks for overly long command lines, response lines, and data header lines.
# It can alert on invalid commands, or specific valid commands.  It can optionally
# ignore mail data, and can ignore TLS encrypted data.
#
# SMTP has numerous options available, please read README.SMTP for help
# configuring options.
#####
# Per Step #2, set the following to load the smtp preprocessor
# dynamicpreprocessor file <full path to libsf_smtp_preproc.so>
# or use commandline option
# --dynamic-preprocessor-lib <full path to libsf_smtp_preproc.so>
preprocessor smtp: \
  ports { 25 587 691 } \
  inspection_type stateful \
  normalize cmds \
  normalize_cmds { EXPN VRFY RCPT } \
  alt_max_command_line_len 260 { MAIL } \
  alt_max_command_line_len 300 { RCPT } \
  alt_max_command_line_len 500 { HELP HELO ETRN } \
  alt_max_command_line_len 255 { EXPN VRFY }
# sfPortscan
# ----------
# Portscan detection module.  Detects various types of portscans and
# portsweeps.  For more information on detection philosophy, alert types,
# and detailed portscan information, please refer to the README.sfportscan.
#
# -configuration options-
#     proto { tcp udp icmp ip all }
#       The arguments to the proto option are the types of protocol scans that
#       the user wants to detect.  Arguments should be separated by spaces and
#       not commas.
#     scan_type { portscan portsweep decoy_portscan distributed_portscan all }
#       The arguments to the scan_type option are the scan types that the
#       user wants to detect.  Arguments should be separated by spaces and not
#       commas.
```

```
#     sense_level { low|medium|high }
#       There is only one argument to this option and it is the level of
#       sensitivity in which to detect portscans.  The 'low' sensitivity
#       detects scans by the common method of looking for response errors, such
#       as TCP RSTs or ICMP unreachables.  This level requires the least
#       tuning.  The 'medium' sensitivity level detects portscans and
#       filtered portscans (portscans that receive no response).  This
#       sensitivity level usually requires tuning out scan events from NATed
#       IPs, DNS cache servers, etc.  The 'high' sensitivity level has
#       lower thresholds for portscan detection and a longer time window than
#       the 'medium' sensitivity level.  Requires more tuning and may be noisy
#       on very active networks.  However, this sensitivity levels catches the
#       most scans.
#     memcap { positive integer }
#       The maximum number of bytes to allocate for portscan detection.  The
#       higher this number the more nodes that can be tracked.
#     logfile { filename }
#       This option specifies the file to log portscan and detailed portscan
#       values to.  If there is not a leading /, then snort logs to the
#       configured log directory.  Refer to README.sfportscan for details on
#       the logged values in the logfile.
#     watch_ip { Snort IP List }
#     ignore_scanners { Snort IP List }
#     ignore_scanned { Snort IP List }
#       These options take a snort IP list as the argument.  The 'watch_ip'
#       option specifies the IP(s) to watch for portscan.  The
#       'ignore_scanners' option specifies the IP(s) to ignore as scanners.
#       Note that these hosts are still watched as scanned hosts.  The
#       'ignore_scanners' option is used to tune alerts from very active
#       hosts such as NAT, nessus hosts, etc.  The 'ignore_scanned' option
#       specifies the IP(s) to ignore as scanned hosts.  Note that these hosts
#       are still watched as scanner hosts.  The 'ignore_scanned' option is
#       used to tune alerts from very active hosts such as syslog servers, etc.
#     detect_ack_scans
#       This option will include sessions picked up in midstream by the stream
#       module, which is necessary to detect ACK scans.  However, this can lead to
#       false alerts, especially under heavy load with dropped packets; which is why
#       the option is off by default.
#
preprocessor sfportscan: proto  { all } \
                    memcap { 10000000 } \
                    sense_level { low }
# arpspoof
#--------------------------------------
```

```
# Experimental ARP detection code from Jeff Nathan, detects ARP attacks,
# unicast ARP requests, and specific ARP mapping monitoring.  To make use of
# this preprocessor you must specify the IP and hardware address of hosts on
# the same layer 2 segment as you.  Specify one host IP MAC combo per line.
# Also takes a "-unicast" option to turn on unicast ARP request detection.
# Arpspoof uses Generator ID 112 and uses the following SIDS for that GID:
#  SID     Event description
# -----   -------------------
#   1       Unicast ARP request
#   2       Etherframe ARP mismatch (src)
#   3       Etherframe ARP mismatch (dst)
#   4       ARP cache overwrite attack
#preprocessor arpspoof
#preprocessor arpspoof_detect_host: 192.168.40.1 f0:0f:00:f0:0f:00
# ssh
#--------------------------------------
# EXPERIMENTAL CODE!!!
#
# THIS CODE IS STILL EXPERIMENTAL AND MAY OR MAY NOT BE STABLE!
# USE AT YOUR OWN RISK!  DO NOT USE IN PRODUCTION ENVIRONMENTS.
# YOU HAVE BEEN WARNED.
#
# The SSH preprocessor detects the following exploits: Gobbles, CRC 32,
# Secure CRT, and the Protocol Mismatch exploit.
#
# Both Gobbles and CRC 32 attacks occur after the key exchange, and are
# therefore encrypted.  Both attacks involve sending a large payload
# (20kb+) to the server immediately after the authentication challenge.
# To detect the attacks, the SSH preprocessor counts the number of bytes
# transmitted to the server.  If those bytes exceed a pre-defined limit
# within a pre-define number of packets, an alert is generated.  Since
# Gobbles only effects SSHv2 and CRC 32 only effects SSHv1, the SSH
# version string exchange is used to distinguish the attacks.
#
# The Secure CRT and protocol mismatch exploits are observable before
# the key exchange.
#
# SSH has numerous options available, please read README.ssh for help
# configuring options.
#####
# Per Step #2, set the following to load the ssh preprocessor
# dynamicpreprocessor file <full path to libsf_ssh_preproc.so>
# or use commandline option
# --dynamic-preprocessor-lib <full path to libsf_ssh_preproc.so>
```

```
#
#preprocessor ssh: server_ports { 22 } \
#                max_client_bytes 19600 \
#                max_encrypted_packets 20
# DCE/RPC
#--------------------------------------
#
# The dcerpc preprocessor detects and decodes SMB and DCE/RPC traffic.
# It is primarily interested in DCE/RPC data, and only decodes SMB
# to get at the DCE/RPC data carried by the SMB layer.
#
# Currently, the preprocessor only handles reassembly of fragmentation
# at both the SMB and DCE/RPC layer.  Snort rules can be evaded by
# using both types of fragmentation; with the preprocessor enabled
# the rules are given a buffer with a reassembled SMB or DCE/RPC
# packet to examine.
#
# At the SMB layer, only fragmentation using WriteAndX is currently
# reassembled.  Other methods will be handled in future versions of
# the preprocessor.
#
# Autodetection of SMB is done by looking for "\xFFSMB" at the start of
# the SMB data, as well as checking the NetBIOS header (which is always
# present for SMB) for the type "SMB Session".
#
# Autodetection of DCE/RPC is not as reliable.  Currently, two bytes are
# checked in the packet.  Assuming that the data is a DCE/RPC header,
# one byte is checked for DCE/RPC version (5) and another for the type
# "DCE/RPC Request".  If both match, the preprocessor proceeds with that
# assumption that it is looking at DCE/RPC data.  If subsequent checks
# are nonsensical, it ends processing.
#
# DCERPC has numerous options available, please read README.dcerpc for help
# configuring options.
#####
# Per Step #2, set the following to load the dcerpc preprocessor
# dynamicpreprocessor file <full path to libsf_dcerpc_preproc.so>
# or use commandline option
# --dynamic-preprocessor-lib <full path to libsf_dcerpc_preproc.so>
preprocessor dcerpc: \
    autodetect \
    max_frag_size 3000 \
    memcap 100000
```

```
# DNS
#---------------------------------------
# The dns preprocessor (currently) decodes DNS Response traffic
# and detects a few vulnerabilities.
#
# DNS has a few options available, please read README.dns for
# help configuring options.
#####
# Per Step #2, set the following to load the dns preprocessor
# dynamicpreprocessor file <full path to libsf_dns_preproc.so>
# or use commandline option
# --dynamic-preprocessor-lib <full path to libsf_dns_preproc.so>
preprocessor dns: \
    ports { 53 } \
    enable_rdata_overflow
# SSL
#---------------------------------------
# Encrypted traffic should be ignored by Snort for both performance reasons
# and to reduce false positives.  The SSL Dynamic Preprocessor (SSLPP)
# inspects SSL traffic and optionally determines if and when to stop
# inspection of it.
#
# Typically, SSL is used over port 443 as HTTPS.  By enabling the SSLPP to
# inspect port 443, only the SSL handshake of each connection will be
# inspected.  Once the traffic is determined to be encrypted, no further
# inspection of the data on the connection is made.
#
#   Important note: Stream4 or Stream5 should be explicitly told to reassemble
#                   traffic on the ports that you intend to inspect SSL
#                   encrypted traffic on.
#
#   To add reassembly on port 443 to Stream5, use 'port both 443' in the
#   Stream5 configuration.
preprocessor ssl: noinspect_encrypted

##################################################################
############
# Step #4: Configure output plugins
#
# Uncomment and configure the output plugins you decide to use.  General
# configuration for output plugins is of the form:
#
# output <name_of_plugin>: <configuration_options>
#
```

```
# alert_syslog: log alerts to syslog
# ----------------------------------
# Use one or more syslog facilities as arguments.  Win32 can also optionally
# specify a particular hostname/port.  Under Win32, the default hostname is
# '127.0.0.1', and the default port is 514.
#
# [Unix flavours should use this format...]
# output alert_syslog: LOG_AUTH LOG_ALERT
#
# [Win32 can use any of these formats...]
# output alert_syslog: LOG_AUTH LOG_ALERT
# output alert_syslog: host=hostname, LOG_AUTH LOG_ALERT
# output alert_syslog: host=hostname:port, LOG_AUTH LOG_ALERT
# log_tcpdump: log packets in binary tcpdump format
# -------------------------------------------------
# The only argument is the output file name.
#
# output log_tcpdump: tcpdump.log
# database: log to a variety of databases
# ---------------------------------------
# See the README.database file for more information about configuring
# and using this plugin.
#
output database: log, mysql, user=root password=123123 dbname=snort host=localhost
# output database: alert, postgresql, user=snort dbname=snort
# output database: log, odbc, user=snort dbname=snort
# output database: log, mssql, dbname=snort user=snort password=test
# output database: log, oracle, dbname=snort user=snort password=test
# unified: Snort unified binary format alerting and logging
# -------------------------------------------------------------
# The unified output plugin provides two new formats for logging and generating
# alerts from Snort, the "unified" format.  The unified format is a straight
# binary format for logging data out of Snort that is designed to be fast and
# efficient.  Used with barnyard (the new alert/log processor), most of the
# overhead for logging and alerting to various slow storage mechanisms such as
# databases or the network can now be avoided.
#
# Check out the spo_unified.h file for the data formats.
#
# Two arguments are supported.
#    filename - base filename to write to (current time_t is appended)
#    limit    - maximum size of spool file in MB (default: 128)
#
```

```
# output alert_unified: filename snort.alert, limit 128
# output log_unified: filename snort.log, limit 128

# prelude: log to the Prelude Hybrid IDS system
# ----------------------------------------------
#
# profile = Name of the Prelude profile to use (default is snort).
#
# Snort priority to IDMEF severity mappings:
# high < medium < low < info
#
# These are the default mapped from classification.config:
# info   = 4
# low    = 3
# medium = 2
# high   = anything below medium
#
# output alert_prelude
# output alert_prelude: profile=snort-profile-name

# You can optionally define new rule types and associate one or more output
# plugins specifically to that type.
#
# This example will create a type that will log to just tcpdump.
# ruletype suspicious
# {
#   type log
#   output log_tcpdump: suspicious.log
# }
#
# EXAMPLE RULE FOR SUSPICIOUS RULETYPE:
# suspicious tcp $HOME_NET any -> $HOME_NET 6667 (msg:"Internal IRC Server";)
#
# This example will create a rule type that will log to syslog and a mysql
# database:
# ruletype redalert
# {
#   type alert
#   output alert_syslog: LOG_AUTH LOG_ALERT
#   output database: log, mysql, user=snort dbname=snort host=localhost
# }
#
# EXAMPLE RULE FOR REDALERT RULETYPE:
```

```
# redalert tcp $HOME_NET any -> $EXTERNAL_NET 31337 \
#   (msg:"Someone is being LEET"; flags:A+;)
#
# Include classification & priority settings
# Note for Windows users:  You are advised to make this an absolute path,
# such as:  c:\snort\etc\classification.config
#
include classification.config
#
# Include reference systems
# Note for Windows users:  You are advised to make this an absolute path,
# such as:  c:\snort\etc\reference.config
#
include reference.config
##################################################################
############
# Step #5: Configure snort with config statements
#
# See the snort manual for a full set of configuration references
#
# config flowbits_size: 64
#
# New global ignore_ports config option from Andy Mullican
#
# config ignore_ports: <tcp|udp> <list of ports separated by whitespace>
# config ignore_ports: tcp 21 6667:6671 1356
# config ignore_ports: udp 1:17 53

##################################################################
############
# Step #6: Customize your rule set
#
# Up to date snort rules are available at http://www.snort.org
#
# The snort web site has documentation about how to write your own custom snort
# rules.
#==========================================
# Include all relevant rulesets here
#
# The following rulesets are disabled by default:
#
#   web-attacks, backdoor, shellcode, policy, porn, info, icmp-info, virus,
#   chat, multimedia, and p2p
#
```

```
# These rules are either site policy specific or require tuning in order to not
# generate false positive alerts in most enviornments.
#
# Please read the specific include file for more information and
# README.alert_order for how rule ordering affects how alerts are triggered.
#========================================
include $RULE_PATH/local.rules
include $RULE_PATH/bad-traffic.rules
include $RULE_PATH/exploit.rules
include $RULE_PATH/scan.rules
include $RULE_PATH/finger.rules
include $RULE_PATH/ftp.rules
include $RULE_PATH/telnet.rules
include $RULE_PATH/rpc.rules
include $RULE_PATH/rservices.rules
include $RULE_PATH/dos.rules
include $RULE_PATH/ddos.rules
include $RULE_PATH/dns.rules
include $RULE_PATH/tftp.rules
include $RULE_PATH/web-cgi.rules
include $RULE_PATH/web-coldfusion.rules
include $RULE_PATH/web-iis.rules
include $RULE_PATH/web-frontpage.rules
include $RULE_PATH/web-misc.rules
include $RULE_PATH/web-client.rules
include $RULE_PATH/web-php.rules
include $RULE_PATH/sql.rules
include $RULE_PATH/x11.rules
include $RULE_PATH/icmp.rules
include $RULE_PATH/netbios.rules
include $RULE_PATH/misc.rules
include $RULE_PATH/attack-responses.rules
include $RULE_PATH/oracle.rules
#include $RULE_PATH/mysql.rules
include $RULE_PATH/snmp.rules
include $RULE_PATH/smtp.rules
include $RULE_PATH/imap.rules
include $RULE_PATH/pop2.rules
include $RULE_PATH/pop3.rules
include $RULE_PATH/nntp.rules
include $RULE_PATH/other-ids.rules
include $RULE_PATH/web-attacks.rules
include $RULE_PATH/backdoor.rules
include $RULE_PATH/shellcode.rules
```

include $RULE_PATH/policy.rules
#include $RULE_PATH/porn.rules
include $RULE_PATH/info.rules
include $RULE_PATH/icmp-info.rules
include $RULE_PATH/virus.rules
include $RULE_PATH/chat.rules
include $RULE_PATH/multimedia.rules
include $RULE_PATH/p2p.rules
include $RULE_PATH/spyware-put.rules
include $RULE_PATH/specific-threats.rules
include $RULE_PATH/experimental.rules
include $PREPROC_RULE_PATH/preprocessor.rules
include $PREPROC_RULE_PATH/decoder.rules
Include any thresholding or suppression commands. See threshold.conf in the
<snort src>/etc directory for details. Commands don't necessarily need to be
contained in this conf, but a separate conf makes it easier to maintain them.
Note for Windows users: You are advised to make this an absolute path,
such as: c:\snort\etc\threshold.conf
Uncomment if needed.
include threshold.conf

13. 在 mysql 中建立数据库
# /usr/local/mysql/bin/mysql -p
mysql>SET PASSWORD FOR root@localhost=PASSWORD('123123');
mysql>create database snort;
mysql>grant INSERT,SELECT on root.* to snort@localhost;
mysql>quit;
cd /usr/local/tarballs/snort-2.8.3.1/schemas
/usr/local/mysql/bin/mysql -p <create_mysql snort
>Enter password:123123
进入 mysql 数据库，看看 snort 数据库中的表：
/usr/local/mysql/bin/mysql -p
>Enter password:123123
mysql>show databases;
mysql>use snort;
mysql>show tables;
mysql>exit;

14. 安装配置 web 接口
cp 某个目录/acid-0.9.6b23.tar.gz /www/htdocs
cp 某个目录/adodb498.tgz /www/htdocs

```
# cp 某个目录/jpgragh-3.0.6.tar.bz2   /www/htdocs
# cd   /www/htdocs
# tar -jxvf jpgragh-3.0.6.tar.bz2
# tar -zxvf adodb498.tgz
# tar -zxvf acid-0.9.6b23.tar.gz
# cd acid
# gedit acid_conf.php
```

配置如下（修改有红色标注）：

```
<?php
$ACID_VERSION = "0.9.6b23";
/* Path to the DB abstraction library
 *  (Note: DO NOT include a trailing backslash after the directory)
 *   e.g. $foo = "/tmp"      [OK]
 *        $foo = "/tmp/"     [OK]
 *        $foo = "c:\tmp"    [OK]
 *        $foo = "c:\tmp\"   [WRONG]
 */
$DBlib_path = "/www/htdocs/adodb";
/* The type of underlying alert database
 *
 *  MySQL       : "mysql"
 *  PostgresSQL : "postgres"
 *  MS SQL Server : "mssql"
 */
$DBtype = "mysql";
/* Alert DB connection parameters
 *   - $alert_dbname   : MySQL database name of Snort alert DB
 *   - $alert_host     : host on which the DB is stored
 *   - $alert_port     : port on which to access the DB
 *   - $alert_user     : login to the database with this user
 *   - $alert_password : password of the DB user
 *
 *  This information can be gleaned from the Snort database
 *  output plugin configuration.
 */
$alert_dbname   = "snort";
$alert_host     = "localhost";
$alert_port     = "";
$alert_user     = "root";
$alert_password = "123123";
/* Archive DB connection parameters */
$archive_dbname   = "snort";
$archive_host     = "localhost";
$archive_port     = "";
```

```
$archive_user     = "root";
$archive_password = "123123";
/* Type of DB connection to use
 *   1  : use a persistant connection (pconnect)
 *   2  : use a normal connection (connect)
 */
$db_connect_method = 1;
/* Use referential integrity
 *   1  : use
 *   0  : ignore (not installed)
 *
 * Note: Only PostgreSQL and MS-SQL Server databases support
 *       referential integrity.  Use the associated
 *       create_acid_tbls_?_extra.sql script to add this
 *       functionality to the database.
 *
 *       Referential integrity will greatly improve the
 *       speed of record deletion, but also slow record
 *       insertion.
 */
$use_referential_integrity = 0;
/* Path to the graphing library
 *  (Note: DO NOT include a trailing backslash after the directory)
 */
$ChartLib_path = "/www/htdocs/jpgraph/src";
/* File format of charts ('png', 'jpeg', 'gif') */
$chart_file_format = "png";
/* Chart default colors - (red, green, blue)
 *    - $chart_bg_color_default    : background color of chart
 *    - $chart_lgrid_color_default : gridline color of chart
 *    - $chart_bar_color_default   : bar/line color of chart
 */
$chart_bg_color_default     = array(255,255,255);
$chart_lgrid_color_default  = array(205,205,205);
$chart_bar_color_default    = array(190, 5, 5);
/* Maximum number of rows per criteria element */
$MAX_ROWS = 10;
/* Number of rows to display for any query results */
$show_rows = 50;
/* Number of items to return during a snapshot
 *  Last _X_ # of alerts/unique alerts/ports/IP
 */
$last_num_alerts = 15;
$last_num_ualerts = 15;
```

```
$last_num_uports = 15;
$last_num_uaddr = 15;
/* Number of items to return during a snapshot
 *  Most Frequent unique alerts/IPs/ports
 */
$freq_num_alerts = 5;
$freq_num_uaddr = 15;
$freq_num_uports = 15;
/* Number of scroll buttons to use when displaying query results */
$max_scroll_buttons = 12;
/* Debug mode     - how much debugging information should be shown
 * Timing mode    - display timing information
 * SQL trace mode - log SQL statements
 *   0 : no extra information
 *   1 : debugging information
 *   2 : extended debugging information
 *
 * HTML no cache - whether a no-cache directive should be sent
 *               to the browser (should be = 1 for IE)
 *
 * SQL trace file - file to log SQL traces
 */
$debug_mode = 0;
$debug_time_mode = 1;
$html_no_cache = 1;
$sql_trace_mode = 0;
$sql_trace_file = "";
/* Auto-Screen refresh
 * - Refresh_Stat_Page - Should certain statistics pages refresh?
 * - Stat_Page_Refresh_Time - refresh interval (in seconds)
 */
$refresh_stat_page = 1;
$stat_page_refresh_time = 180;
/* Display First/Previous/Last timestamps for alerts or
 * just First/Last on the Unique Alert listing.
 *    1: yes
 *    0: no
 */
$show_previous_alert = 0;
/* Sets maximum execution time (in seconds) of any particular page.
 * Note: this overrides the PHP configuration file variable
 *       max_execution_time.  Thus script can run for a total of
 *       ($max_script_runtime + max_execution_time) seconds
```

```
 */
$max_script_runtime = 180;
/* How should the IP address criteria be entered in the Search screen?
 *   1 : each octet is a separate field
 *   2 : entire address is as a single field
 */
$ip_address_input = 2;
/* Should a combo box with possible signatures be displayed on the
 * search form. (Requires Javascript)
 *   0 : disabled
 *   1 : show only non pre-processor signatures (e.g., ignore portscans)
 *   2 : show all signatures
 */
$use_sig_list = 0;
/* Resolve IP to FQDN (on certain queries?)
 *    1 : yes
 *    0 : no
 */
$resolve_IP = 1;
/* Should summary stats be calculated on every Query Results page
 * (Enabling this option will slow page loading time)
 */
$show_summary_stats = 0;
/* DNS cache lifetime (in minutes) */
$dns_cache_lifetime = 20160;
/* Whois information cache lifetime (in minutes) */
$whois_cache_lifetime = 40320;
/* Snort spp_portscan log file */
$portscan_file = "";
/* Event cache Auto-update
 *
 *  Should the event cache be verified and updated on every
 *  page log?  Otherwise, the cache will have to be explicitly
 *  updated from the 'cache and status' page.
 *
 *  Note: enabling this option could substantially slow down
 *  the page loading time when there are many uncached alerts.
 *  However, this is only a one-time penalty.
 *
 *   1 : yes
 *   0 : no
 */
$event_cache_auto_update = 1;
```

```
/* Maintain a history of the visited pages so that the "Back"
 * button can be used.
 *
 * Note: Enabling this option will cause the PHP-session to
 * grow substantially after many pages have been viewed causing
 * a slow down in page loading time. Periodically return to the
 * main page to clear the history.
 *
 *   1 : yes
 *   0 : no
 */
$maintain_history = 1;
/* Level of detail to display on the main page.
 *
 * Note: The presence of summary statistics will slow page loading time
 *
 *   1 : show both the links and summary statistics
 *   0 : show only the links and a count of the number of alerts
 */
$main_page_detail = 1;
/*
 * External URLs
 */
/* Whois query */
$external_whois_link = "http://www.samspade.org/t/ipwhois?a=";
//$external_whois_link =
"http://www.geektools.com/cgi-bin/proxy.cgi?targetnic=auto&query="
 /* DNS query */
$external_dns_link = "http://www.samspade.org/t/dns?a=";
/* SamSpade "all" query */
$external_all_link = "http://www.samspade.org/t/lookat?a=";
/* TCP/UDP port database */
$external_port_link = "http://www.snort.org/ports.html?port=";
//$external_port_link = "http://www.portsdb.org/bin/portsdb.cgi?portnumber=";
/* Signature references */
$external_sig_link = array("bugtraq"    => array("http://www.securityfocus.com/bid/", ""),
                     "snort"     => array("http://www.snort.org/snort-db/sid.html?sid=", ""),
                     "cve"       => array("http://cve.mitre.org/cgi-bin/cvename.cgi?name=",
""),
                     "arachnids" => array("http://www.whitehats.com/info/ids", ""),
                     "mcafee"    => array("http://vil.nai.com/vil/content/v_", ".htm"),
                     "icat"      => array("http://icat.nist.gov/icat.cfm?cvename=", ""));
/* Email Alert action
 *
```

```
 * - action_email_from : email address to use in the FROM field of the mail message
 * - action_email_subject : subject to use for the mail message
 * - action_email_msg : additional text to include in the body of the mail message
 * - action_email_mode : specifies how the alert information should be enclosed
 *     0 : alerts should be in the body of the message
 *     1 : alerts should be enclosed in an attachment
 */
$action_email_from = "ACID Alert <acid>";
$action_email_subject = "ACID Incident Report";
$action_email_msg = "";
$action_email_mode = 0;
/* Custom (user) PHP session handlers
 *
 * - use_user_session : sets whether user PHP session can be used (configured
 *                      with the session.save_handler variable in php.ini)
 *     0 : no
 *     1 : yes (assuming that 'user_session_path' and 'user_session_function'
 *             are configured correctly)
 * - user_session_path : file to include that implements the custom PHP session
 *                       handler
 * - user_session_function : function to invoke in the custom session
 *                           implementation that will register the session handler
 *                           functions
 */
$use_user_session = 0;
$user_session_path = "";
$user_session_function = "";
?>
```

15. 进入 WEB 界面：

http://localhost/acid/acid_main.php

点"Setup Page"链接 ->Create Acid AG

访问 http://localhost/acid 将会看到 ACID 界面。

16. 测试运行

```
# /etc/init.d/mysql restart
# /etc/init.d/httpd start
# snort -c /etc/snort/snort.conf
```

打开 http://yourhost/acid 查看记录

(这里可以 ping 一下 snort 所在的虚拟机,再查看一下网页,会有 icmp 包的显示)