Автор:
20.12.10 12:32 - Последнее обновление 30.12.10 08:36

Untangle 8,0 подсчет трафика (ipcad + LightSquid)

Предыдущая статья: **ipcad + sarg**

Дабы народ и я не путались, опишу всё с самого начала.

Данная статья описывает сбор статистики по всем портам

Открываем SSH доступ к Untangle

rm /etc/ssh/sshd_not_to_be_run && /etc/init.d/ssh start

/etc/init.d/ssh restart

ssh 192.168.0.1

Настройка – Сеть - Advanced - Packat Filter - Accept SSH traffic from all interfaces - ставим галочку

После этих манипуляций, на сервер Untangle можно "заходить" по SSH

Раскоментируем все DEB репозитарии в /etc/apt/sources.list

Ставим необходимые пакеты

 # apt-get install libpcap-dev

#apt-get install build-essential

#apt-get install linux-libc-dev

 # apt-get install rsh-client

 # wget http://lionet.info/soft/ipcad-3.7.3.tar.gz
 # tar -xvzf ipcad-3.7.3.tar.gz
 # cd ipcad-3.7.3
 # ./configure

 # make

Возможные ошибки:

Ошибка при установке libpcap-dev

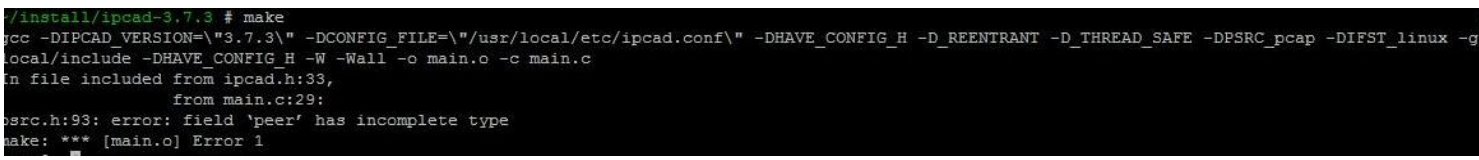
Depends: g++-4.3 (>= 4.3.1-1)

Делаем:
sudo aptitude install build-essential
sudo aptitude update && aptitude install build-essential
читаем тут

Ошибка при сборке ipcad

~/install/ipcad-3.7.3 # make
gcc -DIPCAD_VERSION=\"3.7.3\" -DCONFIG_FILE=\"/usr/local/etc/ipcad.conf\" -DHAVE_CONFIG_H -I
In file included from ipcad.h:33,
from main.c:29:
psrc.h:93: error: field ‘peer’ has incomplete type
make: *** [main.o] Error 1

Картинка



Решение проблемы:

в файле headers.h надо раскоментировать
```
#ifdef HAVE_LINUX_NETLINK_H
#include <linux/netlink.h>
#endif
```

привести к следующему виду
```
//#ifdef HAVE_LINUX_NETLINK_H
#include <linux/netlink.h>
//#endif
```

Подробно об ошибках ТУТ и ТУТ

У меня в процессе установки на Untangle 7.41 выдало ошибку

loop-ipq.c: In function 'process_ipq':
loop-ipq.c:106: error: 'NF_ACCEPT' undeclared (first use in this function)
loop-ipq.c:106: error: (Each undeclared identifier is reported only once
loop-ipq.c:106: error: for each function it appears in.)
*make: *** [loop-ipq.o] Error 1*

непонятно безобразие ... что бы его победить придется кое-что подправить

Редактируем /usr/include/linux/netfilter.h

В файле netfilter.h закоментировать (пример ТУТ)

```
//----------------------------------
//enum nf_inet_hooks {
//    NF_INET_PRE_ROUTING,
//    NF_INET_LOCAL_IN,
//    NF_INET_FORWARD,
//    NF_INET_LOCAL_OUT,
//    NF_INET_POST_ROUTING,
//    NF_INET_NUMHOOKS
```

```
//};

//---------------------------------
//union nf_inet_addr {
//    __u32       all[4];
//    __be32      ip;
//    __be32      ip6[4];
//    struct in_addr    in;
//    struct in6_addr    in6;
//};
```

Теперь

```
# ./configure
# make
# make install
```

```
#whereis ipcad
ipcad: /usr/local/bin/ipcad /usr/local/etc/ipcad.conf
```

/usr/local/etc/ipcad.conf - конфигурационный файл
/usr/local/etc/ipcad.conf.default - на случай если захотим вернуться к настройкам по умолчанию
/usr/local/etc/ipcad.conf.simple - упрощенный файл конфигурации

ipcad.conf (локальный интерфейс у меня interface eth1) мой конфиг ТУТ

```
        #
# Configuration file for ipcad - Cisco IP accounting simulator daemon.
# Copyright (c) 2001, 2002, 2003, 2004, 2005
#     Lev Walkin <vlm@lionet.info>.
#
# Please see ipcad.conf(5) for additional explanations.
```

```
# Please contact me if you have troubles configuring ipcad. My goal is to make
# initial configuration easier for new users, so your input is valuable.
#

###################
# GLOBAL OPTIONS #
###################

#
# Enable or disable capturing UDP and TCP port numbers, IP protocol and
# ICMP types for RSH output.
#
#     capture-ports {enable|disable} ;
#
# Enabling this will BREAK Cisco RSH output format compatibility,
# increase memory requirements and may slow down traffic processing.
# This option takes effect IMMEDIATELY, that is, it can be specified
# multiple times, even between interfaces configuration.
# This option has NO effect on NetFlow operation (NetFlow always captures
# port information).
#

##capture-ports disable;
capture-ports enable;

#
# Buffers to be used for transferring the data from the kernel,
# if applicable (BPF, ULOG).
# Using larger buffers may increase the performance but will
# affect responsiveness.
#
# buffers = <number>[{k|m}] ;
#
# Reasonable defaults are used if this parameter is not set.
#

## buffers = 64k;


######################
# INTERFACE OPTIONS #
######################

#
# interface <iface> [ promisc ] [ input-only ]
#           [ netflow-disable ] [ filter "<pcap_filter>" ] ;
```

```
# OR
# interface ulog group <group> [, group <group> ...]
#           [ netflow-disabled ];
# OR
# interface ipq [ netflow-disabled ];    # man libipq(3)
# OR
# interface {divert|tee} port <divert-port>           # man divert(4)
#           [ input-only ] [ netflow-disabled ];
# OR
# interface file <tcpdump-output.pcap> [ netflow-disabled ];
#
# Options meaning:
#
# promisc:
#     Put interface into promiscuous mode.
#     This enables listening for the packets which are not destined for
#     this host and thus ipcad will count and display all the traffic within
#     the local network. Note that the interface might be in promiscuous mode
#     for some other reason.
#
# input-only:
#     Use kernel feature of counting only incoming packets.
#
# netflow-sampled:     (DO NOT ENABLE THIS OPTION, unless you have to!)
#     If the NetFlow export mechanism is used, this option instructs
#    the interface to supply only one out of N packets to the NetFlow
#    accounting code, thus lowering the CPU requirements. The value of N
#    is configured explicitly in a NetFlow configuration section.
#    NOTE: This option is NOT used to enable NetFlow on the interface,
#    it just modifies the NetFlow behavior on this interface.
#    DO NOT ENABLE THIS OPTION, UNLESS YOU HAVE TO!
#
# netflow-disable:
#    By default, all interfaces are included into NetFlow accounting.
#    This option is used to disable NetFlow on a particular interface.
#
# filter:
#     Install a custom filter on packets instead of basic
#    IP protocol filter. Requires libpcap (even if BPF is being used).
#    May be employed to eliminate CPU overhead on passing unnecessary
#    data between the kernel and user space (by installing the filter
#    directly into the kernel).
#
# NOTES:
#  * "input-only" directive must be supported by kernel.
#    Probably, you were noticed about it during the compilation process
```

```
#    if it was not supported.
#    FreeBSD 3.x and elder kernels do not support this feature.
#  * ULOG packet source (interface ulog) is supported under
#    Linux >= 2.4.18-pre8.
#    You should configure iptables to dump the packet stream
#    into the appropriate group, i.e.:
#        iptables -A OUTPUT -j ULOG --ulog-nlgroup <group>
#    Given ULOG groups will be OR'ed together.
#  * A wildcard (*) may be specified as part of an interface name.
#
#--------------------------------------------------------------------------------

#interface fxp1 filter "ip and dst net 192.168.0.0/24 and not src net 192.168.0.0/24";
interface eth1 filter "ip and dst net 192.168.0.0/24 and not src net 192.168.0.0/24";
#interface eth1;
##interface ed0;
##interface ed1 promisc filter "ip and not dst net 192.168.0.0/16";
###interface sbni0 input-only netflow-disable;    # Disable NetFlow.
###interface ppp*;                # Dynamically pick up PPP interfaces.
###interface ulog group 3, group 15;    # Use ULOG, do not disable NetFlow.
###interface ipq;                # Use Linux IPQ (libipq(3))
###interface tee port 123;            # Use BSD ipfw(8)'s tee.
###interface divert port 321 netflow-disable;    # Use ipfw(8)'s divert(4).

#
# aggregate <ip>/<masklen> strip <maskbits> ;
#
# Aggregate addresses from the specified network (<ip>/<masklen>),
# by AND'ing with specified mask (<maskbits>).
#
#
aggregate 192.168.0.0/24 strip 32;
aggregate 0.0.0.0/0 strip 32;

##aggregate 192.168.0.0/16 strip 32; /* Don't aggregate internal range */
##aggregate 0.0.0.0/0 strip 24;    /* Aggregate external networks */

#
# aggregate <port_range_start>[-<port_range_end>] into <port> ;
#
# Aggregate port numbers. Meaningful only if capture-ports is enabled.
#

aggregate 1-19 into 65535;
aggregate 20-21 into 21;
aggregate 22-23 into 22;
```

```
aggregate 25 into 25;
aggregate 24 into 65535;
aggregate 26-79 into 65535;
aggregate 80-81 into 80;
aggregate 82-109 into 65535;
aggregate 110 into 110;
aggregate 111-442 into 65535;
aggregate 443 into 443;
aggregate 444-3127 into 65535;
aggregate 3128 into 3128;
aggregate 3129-65535 into 65535;

##aggregate 1024-65535    into 65535;    /* Aggregate wildly */
##aggregate 3128-3128    into 3128;    /* Protect these ports */
##aggregate 150-1023    into 1023;    /* General low range */


############################
# NetFlow EXPORT OPTIONS #
############################

#
# Enable Cisco NetFlow export method.
# NetFlow uses UDP to feed flow information to the receiver.
# If the destination is not specified, NetFlow is disabled.
#

# netflow export destination 127.0.0.1 9996;
netflow export version 5;    # NetFlow export format version {1|5}
netflow timeout active 30;    # Timeout when flow is active, in minutes
netflow timeout inactive 15;    # Flow inactivity timeout, in seconds
netflow engine-type 73;        # v5 engine_type; 73='I' for "IPCAD"
netflow engine-id 1;        # Useful to differentiate multiple ipcads.

# The following option is enabled by the "netflow-sampled" interface flag.
#netflow sampling-mode packet-interval 10;   # 1 out of 10 packets accounted
# DO NOT ENABLE THIS UNLESS YOU KNOW WHAT ARE YOU DOING.

#
# NetFlow protocol exports an SNMP id instead of the interface name
# (i.e., "eth0", "ppp32"). The following statements options define
# mapping between the interface names and a set of "SNMP identifiers".
#
netflow ifclass eth mapto 0-99;        # i.e., "eth1"->1, "eth3"->3
netflow ifclass fxp mapto 0-99;        # i.e., "fxp4"->4, "fxp0"->0
netflow ifclass ppp mapto 100-199;    # i.e., "ppp32"->532, "ppp7"->507
```

```
netflow ifclass gre mapto 200-299;
netflow ifclass tun mapto 300-399;    # i.e., "tun0"->300

######################
# RSH SERVER OPTIONS #
######################

#
# Enable RSH Server:
#
#   rsh {enable|yes|on|disable|no|off} [at <listen_ip>];
#
# If "at <listen_ip>" omitted, rsh server listens on IP address 0.0.0.0,
# which may be undesirable.
#

rsh enable at 127.0.0.1;

#
# RSH access rules:
#
# rsh [<user>@]<host_addr> {admin|backup|[default]|view-only|deny} ;
#


rsh root@127.0.0.1 admin;    /* Can shutdown ipcad */
rsh staff@127.0.0.1 backup;    /* Can dump/restore/import accounting table */
rsh yourself@127.0.0.1;        /* Can view and modify accounting tables */
/* Note the order! */
###rsh luser@127.0.0.1 deny;    /* Deny this user from even viewing tables */
rsh 127.0.0.1 view-only;    /* Other users can view current tables */



# Keep IP packet time to live reasonably low to avoid remote attacks.
# (The rsh client must reside no more than three hops away from the
# router running ipcad.)
rsh ttl = 3;

# Set rsh timeout for the same purpose.
rsh timeout = 30;

#
# Dump active IP accounting table to this file on exit and read on startup.
# (read about -s and -r options in ipcad(8) manual page)
# NOTE: This setting has no effect on NetFlow operation. The flow cache
```

```
#       contents are flushed to the collector upon ipcad termination.
#

dumpfile = /var/log/ipcad/ipcad.dump;
##dumpfile = ipcad.dump;    # The file is inside chroot(), see below...

#################
# OTHER OPTIONS #
#################

#
# Chroot to this directory before processing.
#
# Of course, you could disable chroot()'ing by commenting it out,
# but it is not recommended, so I left this confusing default
# to encourage you to change it.
#

###chroot = /adm/tmp;
##chroot = /var/log/ipcad;



#
# File to keep getpid() in it. ipcad will also hold a lock.
#
# WARNING: Pidfile is created AFTER chroot()'ing, so if you're using
# chroot statement above, make sure the path to the pidfile exists
# inside chrooted environment.
#

###pidfile = ipcad.pid;
pidfile = /var/log/ipcad/ipcad.pid;

#
# UID/GID privileges dropping
# Please note: RSH service will be UNAVAILABLE when uid is not zero.
# Use it only when you know what are you doing (i.e., NetFlow without RSH).
#
# uid = 65534;
# gid = 65534;

#
# Few useful settings.
#

#
```

```
# Memory usage limit for storing per-stream entries.
#
# memory_limit = <number>[{k|m|e}] ;
# Where k, m and g are for kilobytes, megabytes or table "entries".
#

##memory_limit = 1m;

memory_limit = 10m;
```

Создаем dump файл

```
mkdir /var/log/ipcad/
touch /var/log/ipcad/ipcad.dump
chmod 600 /var/log/ipcad/ipcad.dump
```

Запускаем ipcad командой /usr/local/bin/ipcad -rds

```
        # /usr/local/bin/ipcad -rds
Opening fxp1... [LCap] [ERSH] [4096] Initialized as 1
Aggregate network 192.168.0.0/255.255.255.0 -> 255.255.255.255
Aggregate network 0.0.0.0/0.0.0.0 -> 255.255.255.255
Aggregate ports 1..19 into 65535
Aggregate ports 20..21 into 21
Aggregate ports 22..23 into 22
Aggregate ports 25..25 into 25
Aggregate ports 24..24 into 65535
Aggregate ports 26..79 into 65535
Aggregate ports 80..81 into 0
Aggregate ports 82..109 into 65535
Aggregate ports 110..110 into 110
Aggregate ports 111..442 into 65535
```

Aggregate ports 443..443 into 443
Aggregate ports 444..3127 into 65535
Aggregate ports 3128..3128 into 0
Aggregate ports 3129..65535 into 65535
Configured RSH Server listening at 127.0.0.1
No valid entries found in ipcad.dump.
Daemonized.

Означает, что всё работает!

Добавим ipcad в автозагрузку

идем в /etc/init.d создаем файл ipcad делаем файл исполняемым.

содержимое файла ipcad

```
#! /bin/sh
/usr/local/bin/ipcad -rds
```

далее делаем

update-rc.d ipcad defaults

(Удаление скрипта из автозагрузки: # update-rc.d -f имя_скрипта_в_initd remove)

Перегружаем Untangle и проверяем, загрузился ли наш ipcad

ps ax | grep ipcad

```
        ~ # ps ax | grep ipcad
 3978 ?        S<sl   0:00 /usr/local/bin/ipcad -rds
```

Создадим директорию squid (/var/log/squid) и файл в ней access.log

Теперь нам нужен **скрипт**, который будет переносить статистику из ipcad в лог /var/log/squid/access.log

Забираем скрипт ТУТ или из winscp создаем в /root файл tolog.sh и заносим в него следующее:

```
        #!/bin/sh
 net="192.168"
 ttime=`/usr/bin/rsh localhost sh ip acco | grep 'Accounting data saved' | awk '{print ($4)}'`
 rsh localhost clear ip accounting
 rsh localhost show ip accounting checkpoint | grep $net | awk -v vtime=$ttime '{if ($5 != 0) print (vtime".0
```

сохраняем и в свойствах файла ставим атрибуты в 0555

Правим /etc/crontab

Дописываем данную строку (*/1 * * * * root cd / && /root/tolog.sh) в /etc/crontab

и перегрузить crond

Установка LightSquid

Ставим perl

aptitude install perl

1 Установка LightSquid

Создаем папку в котору юбудем устанавливать LightSquid:

```
#mkdir /var/www/lightsquid
#cd lightsquid
#wget http://downloads.sourceforge.net/project/lightsquid/lightsquid/1.8/lightsquid-1.8.tgz
```

/посмотреть ссылку для скачивания можно тут -
http://sourceforge.net/project/showfiles.php?group_id=135328
#tar -xzf lightsquid-1.8.tgz /распаковываем архив

Устанавливаем права на запуск скриптов:

#chmod +x *.cgi
#chmod +x *.pl

2 Настройка Apache

.cgi файлы должны обрабатываться как CGI скрипты

правим /etc/apache2/apache2.conf
Нужно дописать в конф файл следущие строчки

<Directory "/var/www/lightsquid">
AddHandler cgi-script .cgi
AllowOverride All
</Directory>

#/etc/init.d/apache restart /перезагружаем apache

3 Редактируем lightsquid.cfg

Нужно ввести свои значения в этих полях, если все делали как выше написано, то изменить тогда надо только путь к логам squid

#path to additional `cfg` files
$cfgpath ="/var/www/lightsquid";
#path to `tpl` folder
$tplpath ="/var/www/lightsquid/tpl";
#path to `lang` folder
$langpath ="/var/www/lightsquid/lang";
#path to `report` folder
$reportpath ="/var/www/lightsquid/report";
#path to access.log
$logpath ="/var/log/squid";
#path to `ip2name` folder
$ip2namepath ="/var/www/lightsquid/ip2name";

4 Графические отчеты.

Для работы графический отчетов в LightSquid вам нужно установить GD библиотеку.

#aptitude install libgd-gd2-perl

5 Проверка check-setup.pl

Чтоб проверить правильно ли все настроено запускаем check-setup.pl

#./check-setup.pl

Если все нормально, идем дальше, если нет - читаем в чем ошыбка и исправляем.

6 Пробный запуск lightparser.pl

#./lightparser.pl

Так же возможно пропарсить старые log файлы если у вас уже работал SQUID

#./lightparser.pl access.log.1.{gz|bz2}
#./lightparser.pl access.log.2.{gz|bz2}
#./lightparser.pl access.log.3.{gz|bz2}
...

7 Пробуем зайти на lightsquid

http://<ваш хост>/lightsquid/

Вероятнее всего вы увидите таблицу с ошибками, так как лог файл еще пуст!

8 Установка задания в cron

Если вы не хотите каждый раз перед тем как посмотреть статистику, заходить на сервер, и запускать вручную скрипт, то переложим эту обязанность на cron.

#crontab -e
В открывшемся списке нужно дописать строчку
*/30 * * * * /var/www/lightsquid/lightparser.pl today
Будьте внимательны с адресом если он у вас отличается.
*/30 означает что скрипт будет запущен каждых полчаса.

Либо дописать данную строку (*/2 * * * * root cd / && /var/www/lightsquid/lightparser.pl today) в /etc/crontab

и перегрузить cron /etc/init.d/cron restart

В /etc/crontab у вас должно быть две дополнительные строчк:

```
*/1 * * * * root   cd / && /root/tolog.sh
 */2 * * * * root  cd / && /var/www/lightsquid/lightparser.pl today
```

Смотрим статистику http://192.168.0.15/lightsquid

P.S. Дополнения, исправления, мыльте через форму обратной связи.

{jcomments on}