Автор: lissyara.
Оригинал : http://www.lissyara.su/articles/freebsd/traffic_count/trafd+mysql/

Немного истории. Когда-то у меня на сервере стоял trafd. Стоял и стоял. Считал траффик бегавший в инет и обратно. Раз в день, ночью, запускался скрипт, который подбивал траффик по хостам сети и присылал мне письмо с отчётом: кто и сколько с начала месяца успел натаскать траффика.
Короче, всё хорошо, но душа хотела чего-то большего :). Примерно в это время мне попалась на глаза статья в которой описывалось как трафик полученный от trafd запихать в БД MySQL, были приложены скрипты на perl и образцы WEB-морды на php. Ну я и загорелся.
Скрипты на perl оказались нерабочие. Может быть они и были рабочие, но в перл я ничё не понимал и до сих пор мало понимаю - мне не нравится синтаксис этого языка. Слишком свободный, слишком много можно, в том числе и в плане ошибок - поэтому было переписано на shell - ибо этот язык самый универсальный на UNIX. Также неустраивала функциональность, в нём был самый минимум - кто, куда, сколько. Обработка запускалась раз в сутки, т.е. время "во сколько" автоматически выпадало - потому, что эту статистику trafd не собирает. Также меня интересовали порты и протоколы по которым пользователи ползают - сколько на ICQ, например уходит, сколько на FTP (про HTTP отдельная песня - у меня squid, а под него считалок более чем достаточно).
Логику примерно я понял из описалова исходного скрипта - оставалась только реализация. Вот сам скрипт который получился:

```
#!/bin/sh
#
#
# Вводим данные для подключения к MySQL серверу
# IP адрес MySQL сервера
IP_MySQL_servera="localhost"
# Имя пользователя для доступа к БД в которой храниться траффик
username="trafd"
# Пароль пользователя MySQL
user_passw="trafd"
# Имя базы данных
db_name="trafd"

# поехали

# Сегодяшний день
day="`date +%Y-%m-%d`"
# Текущий год
year="`date +%Y`"
# Текущий месяц
month="`date +%m`"
```

```
# Текущее время (секунды специально сделаны 00 - иногда cron запускает скрипт не
# в 00 секунд а позже (максимум что я видел - в 13), если машина очень загружена -
# как итог в логах начинает фигурировать разное число секунд.
# Мне это непонравилось :)
curr_time="`date +%H:%M:00`"
# Директория в которой будут храниться текстовые файлы с логами trafd
NewDir="/var/traffic/${year}/${month}"
# Пытаемся создать эту самую директорию на случай если это первый запуск
# или произошла смена месяца (года)
mkdir -p ${NewDir}
# Ну и топаем туда
cd ${NewDir}

# Местоположение исполняемого файла клиента MySQL
mysql="/usr/local/bin/mysql"
# Префикс для команд (лень же каждый раз набивать параметры подключения)
sql_preffix="${mysql} --host=${IP_MySQL_servera}
--user=${username} --password=${user_passw} --database=${db_name}"

# Считываем все переменные из файла /etc/rc.conf с целью извлечь оттуда
# строчку с названиями интерфейсов по которым работает trafd
# (У меня три сетевых платы и lo0 - просто интереса ради)
. /etc/rc.conf

# Для всех интерфейсов выковырнутых из rc.conf (висят в ${trafd_ifaces})
# выполняем один и тот же набор действий по разбору логов и запихиванию
# их в базу данных
for iface in ${trafd_ifaces}
do
# Сохраняем статистику по текущему интерфейсу
/usr/local/bin/trafsave ${iface}
# Преобразуем логи из двоичного в текстовый формат. Сохраняются они в
# папке /tmp в виде файлов summary.* с расширением по имени интерфейса
/usr/local/bin/traflog -i ${iface} -a -n -s > /tmp/summary.${iface} 2>/dev/null
# Очищаем файл с логами в двоичном формате
cat /dev/null > /usr/local/var/trafd/trafd.${iface}
# Дозаписываем логи в текстовый файл (пусть лежат на всякий случай...)
cat /tmp/summary.${iface} >> ${NewDir}/summary.${iface}
# Далее - загоняем траффик в БД
#
${sql_preffix} --execute="CREATE TABLE `traffic_tmp`
(`date` DATE NOT NULL, `time` TIME NOT NULL,
`from_IP` CHAR(16) NOT NULL, `port_from_IP` CHAR(8) NOT NULL,
`to_IP` CHAR(16) NOT NULL, `port_to_IP` CHAR(8) NOT NULL,
`protocol` ENUM('icmp','tcp','udp') NOT NULL, `bytes` int(16) NOT NULL,
`all_bytes` int(16) NOT NULL) TYPE=MyISAM COMMENT='tmp_table'" 2>/dev/null
```

```
# Лопатим данные для интерфейса ${iface}
# Очищаем временную таблицу
${sql_preffix} --execute="DELETE FROM `traffic_tmp`"
# Построчно превращаем файл со статистикой в набор переменных
grep -v "^ " /tmp/summary.${iface} |
{
while read stroka
do
from_IP=`echo "${stroka}" | awk '{print $1}'`
port_from_IP=`echo "${stroka}" | awk '{print $2}'`
to_IP=`echo "${stroka}" | awk '{print $3}'`
port_to_IP=`echo "${stroka}" | awk '{print $4}'`
protocol=`echo "${stroka}" | awk '{print $5}'`
bytes=`echo "${stroka}" | awk '{print $6}'`
all_bytes=`echo "${stroka}" | awk '{print $7}'`
# Загоняем полученный набор во временную таблицу
${sql_preffix} --execute="INSERT INTO `traffic_tmp` (`date`,
`time`, `from_IP`, `port_from_IP`, `to_IP`, `port_to_IP`,
`protocol`, `bytes`, `all_bytes`)
values ('${day}', '${curr_time}', '${from_IP}',
'${port_from_IP}', '${to_IP}', '${port_to_IP}',
'${protocol}', '${bytes}', '${all_bytes}')"
done
}
# Стираем пустые строки (а вот откуда они вылазиют я так и непонял....)
${sql_preffix} --execute="DELETE FROM `traffic_tmp` WHERE from_IP='' AND
port_from_IP='' AND to_IP='' AND port_to_IP='' AND protocol=""
# Стираем строки в которых полное число байт (вместе с технической инфой)
# равно нулю (тоже непойми откуда берутся - раз в статистику trafd попали -
# значит соединение было и байты должны были б быть...)
${sql_preffix} --execute="DELETE FROM `traffic_tmp` WHERE all_bytes='0'"
# Создаём таблицу для окончательного хранения траффика
# (на тот случай если она не создана - хотя конечно тоже дурацкий вариант -
# пытаться создать таблицу при каждом запуске скрипта, но другой вариант -
# проверять существование и если нету её - то создавать. А какая разница? Так
# как сделано сейчас - проще и менее ресурсоёмко)
${sql_preffix} --execute="CREATE TABLE `${iface}_${year}`
(`unic_id` INT(16) NOT NULL AUTO_INCREMENT,
`date` DATE NOT NULL, `time` TIME NOT NULL,
`from_IP` CHAR(16) NOT NULL, `port_from_IP` CHAR(8) NOT NULL,
`to_IP` CHAR(16) NOT NULL, `port_to_IP` CHAR(8) NOT NULL,
`protocol` ENUM('icmp','tcp','udp') NOT NULL, `bytes` int(16) NOT NULL,
`all_bytes` int(16) NOT NULL,
PRIMARY KEY (`unic_id`),
KEY `date`(`date`)
) TYPE=MyISAM COMMENT='База
```

```
данных траффика по (${iface}) интерфейсу за ${year} год'" 2>/dev/null
# Перекидываем траффик из временной таблицы в окончательную, при этом
# объединяем строки в которых совпадает ВСЁ кроме числа байт.
${sql_preffix} --execute="INSERT INTO `${iface}_${year}`
(`date`, `time`, `from_IP`, `port_from_IP`, `to_IP`,
`port_to_IP`, `protocol`, `bytes`, `all_bytes`)
SELECT `date`, `time`, `from_IP`, `port_from_IP`,
`to_IP`, `port_to_IP`, `protocol`, sum(`bytes`) as `bytes`,
sum(`all_bytes`) as `all_bytes` FROM
`traffic_tmp` GROUP BY `date`, `time`, `from_IP`, `port_from_IP`,
 `to_IP`, `port_to_IP`, `protocol`"

done

# Очищаем файл с логами о том когда и по какому интерфейсу сохранялась статистика
cat /dev/null > /var/log/traffic.log
```

Теперь вкратце описалово. С той частью где ввод переменных кажется проблем быть не должно
— NewDir="/var/traffic/${year}/${month}" - директория для сохранения статистики trafd в его "родном", текстовом формате. Нужно это или нет - сами решайте. Я на всякий случай сохранял - мало ли что (БД случайно грохну, например :)).
— IP_MySQLservera="localhost" - IP адрес MySQL сервера на котором хранится БД. В данном варианте это наследие времён, когда сервер БД стоял на одной машине, а trafd крутился на другой. Если они у Вас стоят на одной машине то можно смело повыкидывать все упоминания о хосте сервера.
— username="trafd" - имя пользователя для доступа к БД
— user_passw="trafd" - парольпользователя для доступа к БД
— db_name="trafd" - имя базы данных
Далее статистика интерфейсов сохраняется в /tmp и дописывается в /var/traffic/${year}/${month}. Временный файл читается построчно, строка разбирается по полям в переменные - кто, куда, откуда, по каким портам, по каким протоколам, сколько байт, сколько байт вместе с технической информацией (не знаю, пригодится ли кому-то последнее поле, но при написании скрипта я постарался чтобы никакие из данных не пропали - пусть лежат, пригодятся). Получается куча переменных содержащих все эти данные, которые и пхаются в MySQL - во временную таблицу `traffic_first`.
После того, как весь текстовый файл переехал в БД начинается разбор того, что попало в БД - удаляются пустые строки, строки в которых число байт равно нулю (вот откуда они появляются я не вполне понимаю - если есть адреса IP, порты - то запрос был, были и данные, но данных нет....). Затем данные перелохмачиваются в таблицу `traffic_tmp` с одновременной суммацией одинаковых строк - бывает такое, что все данные одинаковые кроме числа байт. Чтоб не хранить лишние строки - такие суммируются в одну.
После попытки создать таблицу для хранения данных за текущий месяц (чтобы каждый раз не заморачиваться скрипт при каждом запуске пытается создать свои таблицы, на

случай если это первый запуск, или сменился год) траффик запихивается на постоянное хранение в таблицу с именем типа `sis0_2005_06`. Затем цикл повторяется для другого интерфейса.
Запуск в "кроне" ставится на каждую минуту. Сам по себе скрипт конечно же небыстрый (это связяано с тем, что на каждый запрос соединение с MySQL устанавливается заново), но тем не менее - на моём P-166MMX он успевал отработать за 5-7 секунд даже если процессор был занят чем-то ресурсоёмким (компиляция и прочее).Если нет реальной необходимости иметь поминутную статистику то можно запускать реже - раз в 10 минут или раз в час. WEB-морду к этому хозяйству прикручивать изначально планировал - и даже приделал. Но - та, что была мне не понравилась - интерфейсом и убогостью метода подсчёта - хосты валились в кучу без разбора локальная или нет сетка, и траффик считался и входящий и исходящий без учёта того, что могут быть обращения на сам сервер (например я постоянно работаю с серваком и к концу месяца при таких методиках подсчёта лишнего траффика, неимеющего отношения к инету набирается бывает несколько гигабайт(!) информации.) Потому считать начал в командной строке запросом типа:
mysql --user=trafd --password=trafd --database=trafd --execute="SELECT to_IP,SUM(bytes) FROM sis0_2005 WHERE to_IP LIKE '192.168.%' AND from_IP NOT LIKE '192.168.8.254' GROUP BY to_IP" | mail -s traffic admin@lissyara.su

А текст из полученого письма просто в ёксель скопировать и построить графики для руководства, если вдруг выяснится, что кто-то по http налазил 50 мег и вот по ftp затащил пару порнофильмов :) Будет время, или реальная необходимость в WEB-интерфейсе, тогда и сделаю. Сложности никакой вроде не видно в этом....