Введение

FastNetMon - программный пакет для обнаружения DDoS атак и их последующего отражения.

Мой дорогой пользователь, перед Вами продукт, который поможет Вам обеспечить максимальную доступность Вашей сети и исключить любые сбои по вине DoS/DDoS атак.

В данной инструкции будут описаны возможности доступные в версии 1.1.3, которая в данный момент находится в разработке! В стабильной версии 1.1.2 многое недоступно.

Установка

Требования к программному окружению

Проект имеет полностью открытый код и доступен для следующих платформ:

- Linux
- FreeBSD
- Mac OS X.

Но я бы хотел обратить внимание, что рекомендуемая платформа - Linux и только под нее собираются бинарные .rpm и .deb пакеты.

В момент написания данной инструкции стабильной версией является 1.1.2. В то время как версия 1.1.3 находится в стадии бета-тестирования.

Если Вы внедряете продукт в данный момент, я рекомендую использовать версию 1.1.3, так как она уже достаточно стабильна и содержит множество крайне полезных функций (BGP Flow Spec, DPI, Graphite, host groups).

Пакеты, которые необходимы для сборки можно найти вот здесь: https://github.com/FastVPSEestiOu/fastnetmon/blob/master/docs/PACKAGES_INSTALL.md Они собраны из находящейся в разработке версии 1.1.3.

Бинарные пакеты имеются для следующих дистрибутивов Linux только для 64 битных платформ (amd64, x86_64):

- Ubuntu 12.04
- Ubuntu 14.04
- CentOS 6
- CentOS 7
- Debian 6
- Debian 7
- Debian 8
- VyOS 1.1.6

Установка из пакетов - предпочтительный вариант установки, пожалуйста, используйте только его.

Кроме этого, для версии 1.1.2 существует возможность установки с помощью скрипта-инсталлятора https://github.com/FastVPSEestiOu/fastnetmon/blob/master/docs/INSTALL.md Но в этом случае процесс установки займет намного больше времени.

Аппаратные требования

Стоит обратить внимание, что возможна установка продукта как в виртуальные окружение, так и на физические серверы.

Работа в режимах mirror и mirror_netmap требует работы либо на физическом сервере либо на виртуальной машине с аппаратной виртулизацией (KVM, ESXi) внутрь которой посредством методики PCI-E Passthrough пропущена аппаратная сетевая.

Режимы sFLOW и NetFLOW никаких особенных требований не накладывают. Обеспечивается работа внутри Docker окружений, внутри OpenVZ контейнеров и любых системы виртулизации.

Кроме этого, при работе в режимах mirror и mirror_netmap имеются ограничения по сетевым картам. В данный момент поддерживаются только сетевые карты семейства Intel 1/10/40GE. Рекомендуемой сетевой картой, которые обеспечивает максимальное быстродействие и надежность являются любые карты на чипе Intel 82599.

Кроме этого, при работе в указанных режимах mirror, netmap_mirror предъявляют серьезные требования по части скорости центрального процессора и объема необходимой оперативной памяти. В каждом отдельном случае они определяются индивидуально. Для примерно расчета можно использовать формулу - 4 логических ядра процессора Intel на 10GE сети.

Первичная конфигурация

После установки проекта из бинарного rpm или deb пакета Вам необходимо выполнить базовую конфигурацию, она включает два шага:

- Выбор плагина для захвата трафика (для изучения списка доступных плагинов, пожалуйста, ознакомьтесь с разделом «Подсистема захвата трафика»)
- Добавление своих сетей (требуется для выявления нашего трафика из потока и определения его направления)

Конфигурацию подсистемы обнаружения атак мы оставляем на потом, потому что она требует детального описания.

Для выбора плагина захвата трафика откройте файл /etc/fastnetmon.conf текстовым редактором и измените значение с «no» на «on» для приглянувшегося варианта.

Обращаю внимание! Что во всех местах, где подразумевается булонское (да/нет) значение параметра используется два варианта: on (включено) и off (отключено). Варианты «yes» и «no» текущей версией распознаны не будут.

Для добавлений своих подсетей, их нужно добавить в файл /etc/networks_list в формате CIDR, одна сеть на строке.

Кроме этого, обязательно отключите блокировку с помощью параметра enable_ban, установив его значение в no, чтобы исключить любые деструктивные действия.

После выполнения базовой конфигурации Вы можете запустить продукт посредством привычного для вашего дистрибутива способом.

В случае использования утилиты service Вам потребуется выполнить следующую команду:
service fastnetmon start

В случае использования дистрибутива с systemd (CentOS 7, Debian 8) Вам нужно будет выполнить:
systemctl start fastnetmon

В случае каких-либо проблем обязательно посмотрите в лог файл, /var/log/fastnetmon.log.

После этого Вы можете запустить клиентскую утилиту мониторинга /opt/fastnetmon/fastnetmon_client и убедиться в том, что продукт видит трафик.

Общая архитектура проекта

Продукт представляет собой многопоточное приложение написанное на C++ и работающее в пространстве пользователя.

С точки зрения архитектуры проект состоит из 3х частей:
- Подсистема захвата трафика
- Подсистема анализа
- Подсистема визуализации
- Подсистема действий

Подсистема захвата трафика

Так как основной рабочий материал для программы - это трафик, то нам его нужно каким-то образом захватывать.

Продукт поддерживает захват трафика следующими способами:

- С зеркальных портов (mirror, SPAN)
- sFLOW v4, v5
- NetFlow v5, v9, v10
- IPFIX

Общую сравнительную таблицу для всех режимов захвата трафика можно найти здесь:
https://github.com/FastVPSEestiOu/fastnetmon/blob/master/docs/CAPTURE_BACKENDS.md

В ряде случае может потребовать отключение обработки трафика в том или ином направлении. Это может быть осуществлено с помощь параметров конфигурации process_incoming_traffic и process_outgoing_traffic.

Также поддерживается отключенный стандартно режим flow трекинга, в котором также обрабатывается информация о числе flow в единицу времени. Его можно включить с помощь флага: enable_connection_tracking, но стоит обратить внимание, что данный режим в разы повышает потребление процессоры ресурсов и снижает объем трафика, который может быть обработан решением.

Режим захвата с зеркальных портов

В данном режиме трафик захватывается с зеркальных портов свитча, роутера либо иного устройства.

Происходит захват всей информации о передаваемом трафике (включая заголовки пакетов и payload).

Данный режим является наиболее функциональным (позволяет производить глубокий анализ содержимого пакетов с помощью DPI, что в свою очередь предоставляет возможность работы протокола BGP flow spec).

Кроме этого, он является наиболее ресурсоемким, так как анализируется каждый пакет и требует быстрого CPU с большим количеством логически ядер.

Поддерживается три варианта захвата с зеркальных порто, их имена в конфигурационном файле следующие:
- pcap
- mirror
- mirror_netmap

Их совместное использование запрещено. С помощью параметра *interfaces* можно указать один или несколько сетевых интерфейсов, которые будут мониторится с помощью выбранного режима. Здесь также можно указать несколько интерфейсов через запятую, все модули захвата поддерживают эту возможность (кроме pcap).

Режим захвата pcap крайне медленный и не может быть использован на скоростях более 30-50 мегабит. Он потребляет очень большое количество процессоры ресурсов и теряет пакеты в случае высокой нагрузки. Этот режим не предназначен для промышленного использования в продакшене. Его особенностью является то, что интерфейс не отключается от операционной системы и может быть использован другими приложениями.

Режим mirror обозначет использование библиотеки PF_RING для захвата трафика. Данная библиотека работает в двух режимах - vanilla и zc.

Режим PF_RING vanilla является довольно медленным и потребуется большое количество ресурсов, он может быть использован до скоростей 2 mpps/5GE и не более. Приятной особенностью данного режима является то, что интерфейс не отключается от операционной системы и может быть использован другими приложениями.

Режим PF_RING ZC является улучшенной версией vanilla, работает до линейной скорости 10GE (14 mpps) и более. Но отключается интерфейс от операционной системы и требует лицензии - актуальный цены можно найти по адресу http://www.nmon.net/shop/cart.php. Кроме того, данный режим можно использовать только на сетевых картах Intel.

Стандартно включен режим vanilla, чтобы включить режим zc нужно использовать флаг enable_pf_ring_zc_mode установив его в значение on, а также добавив префикс zc: ко всем именам интерфейсов (zc:eth0).

Режим mirror_netmap реализуется с помощью драйвера Netmap, который во многом аналогичен (как по скорости, так и функционалу) драйверу PF_RING ZC, но поставляется с открытым кодом и полностью бесплатен.

Для работы режимов mirror и mirror_netmap Вам необходимо произвести установку соответствующиех драйверов и модулей ядра.

Для netmap стоит использовать инструкцию: http://www.stableit.ru/2014/10/netmap-debian-7-wheezy-intel-82599.html

Для PF_RING стоит использовать инструкцию: http://www.stableit.ru/2014/06/pfring-debian-7-wheezy.html (нужен только модуль ядра)

В случае использования режимов mirror и mirror_netmap возможно использование сэмплированных режимов захвата с порта (когда дублируется не весь трафик, а его определенная часть, например, 1/256). В этом случае нужно программе сообщить выбранную на стороне сетевого устройства частоту семплирования с помощью параметров: netmap_sampling_ratio (для mirror_netmap) и pfring_sampling_ratio для режима mirror.

Режим захвата с помощью NetFlow

В данном режиме мы принимаем телеметрическую информацию о трафике. Стандартно она принимается на 2055й порт протокола UDP по всем интерфейсам. Режим включается параметром netflow = on.

Стоит заметить, что данный режим не обеспечивает столь высокой скорости обнаружения атака как sflow и mirror, так как по своей сущности данный протокол содержит таймаут после которого данные о трафике передаются на коллектор.

Со второго устройства генерирующего NetFlow необходимо установить минимально возможные значения для active flow timeout и inactive flow timeout не вызывающее деградацияи производительности (проконсультируйтесь с документацией от Вашего вендора).

Задержка в определении атаке в случае NetFlow/IPFIX увеличивается на большее из этих двух значений (active flow timeout и inactive flow timeout)

Для ОС JunOS Вы можете использовать следующую инструкцию по настройке: https://github.com/FastVPSEestiOu/fastnetmon/blob/master/docs/JUNOS_INTEGRATION.md

Стоит обратить внимание, что NetFlow / ipfix также может быть сэмплированный. Для NetFlow 5 мы поддерживаем извлечение информации о частоте семплирования прямо из NetFlow, но для NetFlow 9 и NetFlow IPFIX это не поддерживается и требуется указывать частоту семплирования в файле конфигурации используя параметр netflow_sampling_ratio.

В ряде случаев у Вас может возникнуть необходимость изменить порт, куда принимается NetFlow, это можно осуществить посредством правки переменной netflow_port. Также можно указать несколько портов, разделенных запятой, чтобы запустить несколько коллекторов (удобно в случае сбора NetFlow информации с разных устройств на разные порты).

Чтобы задать конкретный хост, на котором FastNetMon должен принимать трафик можно использовать переменную netflow_host, в которой явно указать IP на котором должно осуществляться прослушивание трафика либо использовать 0.0.0.0, чтобы прослушивать все интерфейсы.

При наличии такой возможности, рекомендуется отключать агрегацию host+port, так как в противном случае анализ атак будет усложнен.

Данный режим не предоставляет возможностей для работы DPI и исключает использование BGP Flow Spec в данном режиме.

Для особо сложных программных случаев существует возможность обработки всех входящих NetFlow пакетов с помощью lua скрипта, указывая его через параметр конфигурации netflow_lua_hooks_path. В данный момент поддерживается только исключение из рассмотрения определенных пакетов на основании их содержимого. Особенно актуален данный режим в сетях с MPLS, чтобы исключить дублирование трафика. В данный момент поддерживается обработка только NetFlow v5 пакетов.

Режим захвата с помощью sFLOW

Данный режим включается посредством параметра sflow = on.

В данном режиме со стороны сетевого устройства осуществляется сэмплирование данных, без аггрегации.

При выборе частоты сэмплирования трафика нужно всегда обращать внимание на нагрузку устройства, которую создает генерацию sFLOW с него, а также на минимальные значения частоты семплирования для каждого заданного типа интерфейсов http://blog.sflow.com/2009/06/sampling-rates.html Задавая слишком большое значение Вы можете очень сильно понизить точность распознавания атак, что приведет к ложно положительным либо наоборот ложно отрицательным срабатываниям.

Данный режим является очень точным (при разумной частоте сэмплирования) наряду с mirror, обеспечивает высокую скорость обнаружения атаки. Но не предоставляет возможностей для работы DPI и исключает использование BGP Flow Spec в данном режиме.

Для особо сложных программных случаев существует возможность обработки всех входящих sFLOW пакетов с помощью lua скрипта, указывая его через параметр конфигурации sflow_lua_hooks_path. В данный момент поддерживается только исключение из рассмотрения определенных пакетов на основании их содержимого. Особенно актуален данный режим в сетях с MPLS, чтобы исключить дублирование трафика.

Подсистема анализа

Конфигурация детектора

Для глобального включения/выключения любой реакции продукта на атаки служит параметр enable_ban.

Для каждого узла есть возможность указать какие именно параметры трафика могут быть использованы для обнаружения атаки:

- ban_for_pps - блокировка в случае превышения заданного числа pps на/с хоста
- ban_for_bandwidth - блокировка в случае превышения заданного числа мегабит/секунду на/с заданного хоста
- ban_for_flows - блокировка в случае превышения заданного числа потоков/секунду на/с хоста (требует активированной опции enable_connection_tracking, про недостатки которой читайте в соответствующем разделе инструкции)

Каждый из режимов может быть включен или выключен в зависимости от Ваших потребностей.

Для каждого из этих параметров трафика можно указать конкретное значение, после превышения которого будет фиксироваться атака:

- threshold_pps
- threshold_mbps
- threshold_flows

Кроме этого, если Вам требуется полностью отключить обнаружение атак для заданного узла, то для этого может быть использован белый список сетей. Он находится в файле /etc/networks_whitelist, данные в нем размещаются в формате CIDR, одна сеть на строку.

По списку этих сетей в случае обнаружения атаки осуществляется long prefix match и если узел, по которому зафиксирована атака в белом списке - никаких действий не производится.

Но будьте аккуратны с этой опцией! Лучше использовать повышенный порог для заданного узла.

При обнаружении атаки каждый узел блокируется на заданное время, его можно задать через параметр ban_time. По истечении этого времени будет осуществлена его блокировка, если атака прекратилась. Можно добиться разблокировки даже в случае, если атака все еще идет, этого можно добиться с помощью параметра конфигурации unban_only_if_attack_finished. При проверке атаки на активность, скорость сверяется с пороховыми значениями и если они все еще превышены - разблокировка не осуществляется и будет при следующей попытке разбана.

Поток разблокировки запускается стандартно каждые 60 секунд. Но в случае, если Вы задаете время блокировки меньшее 60 секунд, то поток разблокировки запускается каждые время_блокировки/2 секунд.

Механизм групп узлов

Если требуется задать различные значения порогов по трафику для разных сетей, то можно использовать функционал групп хостов.

С помощью следующего синтаксиса можно объявить новую группу узлов с именем my_hosts:
hostgroup = my_hosts:10.10.10.221/32,192.168.1.0/24

Обращаю внимание, что узлы которые Вы здесь перечисляете должны быть явно перечислены в файле /etc/networks_list, иначе ничего работать не будет!

После создания группы, Вы можете указать для нее собственные параметры порогов отличные от заданных глобально с тем лишь отличием, что перед именем переменной добавляется имя группы узлов:

```
 my_hosts_enable_ban = no

my_hosts_ban_for_pps = no
my_hosts_ban_for_bandwidth = no
my_hosts_ban_for_flows = no

my_hosts_threshold_pps = 20000
my_hosts_threshold_mbps = 1000
my_hosts_threshold_flows = 3500
```

Логика расчета скоростей узлов

Для каждого хоста для каждой сети из /etc/networks_list выделяется счетчик заданных типов трафика (pps, bps, flows) в направлениях входящий и исходящий

Скорость обсчитывается каждую секунду для всех хостов в сети для всех типов трафика.

После этого применяется алгоритм усреднения за последние average_calculation_time секунд.

После усреднения осуществляется поиск порогов для заданного узла и если они превышены, активируются действия при обнаружении атаки.

Подсистема визуализации

К данной подсистеме относятся все плагины, которые обеспечивают визуализацию трафика в том или ином виде.

В данный момент поддерживаются:
- Command Line Interface fastnetmon_client (/opt/fastnetmon/fastnetmon_client)
- Система хранения статистики Graphite
- Система хранения статистики infIuDB

FastNetMon client

FastNetMon client отображает 7 (конкретное значение задается с помощью параметра max_ips_in_list) потребителей трафика во входящем и исходящем направлении.

Сортировка осуществляется стандартно по числу пакетов, изменить это можно с помощью параметра конфигурации sort_parameter (он может принимать значения packets, bytes, flows).

Кроме информации о входящем и исходящем трафике отображается еще ряд полей:
- Internal traffic - трафик между нашими узлами
- Other traffic - трафик, который мы не смогли идентифицировать на факт принадлежности нашим сетям
- Total amount of IPv6 packets - общее число заификсированных IPv6 пакетов
- Total amount of not processed packets - общее число пакетов, которые по тем или иным причинам не были распознаны корректно (arp и прочий служебный трафик не использующий протокол IP).

Graphite

В случае активации данного плагина параметром graphite = on при каждом вызове функции пересчета скорости скорости трафика для всех узлов имеющих непутевую скорость трафика будут выгружены в Graphite.

Кроме этого, если в конфигурации был активирован параметр enable_subnet_counters, то в систему Graphite будет также отправлять информация о трафике всех подсеете, указанных в файле /etc/networks_list.

Вы можете добавить ко всем именам полей, которые отправляются в Graphite свой собственный prefix используя параметр конфигурации graphite_prefix.

Задать порт и хост (только в виде IP адреса) можно с помощью соответствующих параметров graphite_host и graphite_port.

Для настройки системы Graphite Вы можете использовать данную инструкцию: https://github.com/FastVPSEestiOu/fastnetmon/blob/master/docs/GRAPHITE_INTEGRATION.md

InfluxDB

Данная база данных имеет поддержку протокола передачи телеметрической информации Graphite, поэтому поддержка InfluxDB настраиваться идентично поддержке протокол Graphite.

Полная инструкция по конфигурации системы InfluxDB может быть найдена здесь: https://github.com/FastVPSEestiOu/fastnetmon/blob/master/docs/INFLUXDB_INTEGRATION.md

Подсистема действий

Она начинает свою работу после того, как будет обнаружена атака.

Она включается в себя несколько компонент:
- Command Line Notify script
- Сбор отпечатков атаки
- Подсистема логирования
- Redis
- BGP анонс
- BGP Flow Spec анонс

Command Line Interface

С помощью параметра конфигурации notify_script_path Вы можете задать абсолютный путь до скрипта, которы будет вызван в следующих случаях:

- Блокировка узла после фиксации атаки «ban»
- Сбор отпечатка атаки «details»
- Разблокировка узла после окончания атаки «unban»

Рядом с каждым случаем дано его имя, которое передается как один из параметров заданного скрипта.

В случае вызова действий ban и details мы передаем множество полезной информации об атаке посредством передачи на stdin скрипту. Обращаю внимание, в связи с особенностями ОС семейства Linux, Вы обязаны принять эту информацию в скрипте, в противном случае вся программа завершится с ошибкой.

Впрочем, если эта информация не требуется, Вы можете отключить передачу дополнительных параметров атаки на stdin для «ban» действия с помощью параметра конфигурации notify_script_pass_details отключить это поведение.

Данному скрипту передаются следующие параметры командной строки:
- IP

- Направление атаки (incoming, outgoing)
- Мощность атаки в пакетах/секунду
- Действие (ban, unban, details)

Пример notify скрипта можно взять вот здесь: https://raw.githubusercontent.com/FastVPSEestiOu/fastnetmon/master/src/notify_about_attack.sh

Подсистема логирования

В ряде случае удобно отправлять все сообщения, добавляемые продуктом в файлы логов, на удаленный либо локальный syslog сервер.

Активации логгирования в локальный syslog можно добиться параметром logging:local_syslog_logging.

Если требуется отправка лог файлов на удаленный сервер, нужно активировать параметр logging:remote_syslog_logging и указать адрес и порт лог сервера через параметры logging:remote_syslog_server, logging:remote_syslog_port.

Сбор отпечатков атаки

В случае фиксации атаки в директории /var/log/fastnetmon_attacks будет создан файл с именем в формате «IP_28_07_15_10:08:17», в котором будет сохранена информация о силе атаки, о протоколах, её название, а также будет сохранен дамп пакетов и дамп flow (если включен flow tracking).

В дополнение к этому режиму есть возможность полного захвата трафика атаки и его последующего сохранения в формате pcap, этого можно добиться параметром collect_attack_pcap_dumps. В этом случае рядом с файлом из предыдущего абзаца будет создан файл с тем же именем, но с расширением .pcap, который может быть прочитан утилизации Wireshark и Tcpdump с целью детального анализа атаки.

Кроме этого, с помощью DPI может быть осуществлена попытка проведения анализа собранного pcap файла, этого можно добиться с помощью активации функции: process_pcap_attack_dumps_with_dpi.

Redis

Данное действие включается параметром redis_enabled, а через параметры redis_host и redis_port задается IP адрес узла, на котором запущен Redis. Так как все ключи хранятся в одном месте, в случае запуска нескольких инстансов приложения может потребоваться разделение их данных в пределах БД Redis, для этого можно использовать ключ redix_prefix.

В Redis сохраняется информация об атаке как только она зафиксирована, сохраняется она в ключ под именем IP_information. Если включен режим трекинга flow, то в ключ IP_flow_dump будет помещена вся информация о всех flow относительно атакованного хоста.

После сбора отпечатка атаки, по адресу IP_packets_dump будет сохранен этот отпечаток.

Обращаю внимание, что значение ключа _information сохраняется в формате JSON, который удобен для обработки из скриптов или внешних приложений. Все остальные данные сохраняются в формате простого текста.

BGP

В этом варианте мы можем поднимать заданный анонс при фиксации атаки на наш хост и опускать его, когда атака прекратилась. Обычно этот вариант называется RTBH, BGP Blackhole.

Для использования этого варианта нам понадобится внешний BGP демон, который можно настроить по инструкции: https://github.com/FastVPSEestiOu/fastnetmon/blob/master/docs/EXABGP_INTEGRATION.md, после установки Вы должны установить BGP сессию между ним и своим сетевым оборудованием.

После того, как демон настроен и запущен, нам нужно активировать поддержку BGP непосредственно в программе.
Это осуществляется с помощью активации флага exabgp = on, также нужно указать путь до API сокета через параметр exabgp_command_pipe.

Кроме этого, нам необходимо указать next hop (эту информацию должен дать Вам сетевой инженер), который будет прописываться в анонсах через параметр: exabgp_next_hop.

Кроме этого, мы можем BGP Community, которое будет добавлено в анонс, это можно сделать с помощью параметра exabgp_community, он имеет вид «65001:666», если требуется установить несколько community, то используйте вот такой синтаксис: «[65001:666 65001:777]». Квадратные скобки обязательны!

Анонсировать мы можем как сам хост, на который зафиксирована атака (exabgp_announce_host = on), так и сеть, которой он принадлежит (exabgp_announce_whole_subnet = on).

BGP Flow Spec анонс

Данный режим во многом аналогичен режиму BGP, но в данном случае мы блокируем не хост целиком, а лишь блокируем паразитный трафик, который с/на него идет.

Как было сказано ранее, этот режим будет доступен только в случае, когда трафик собирается по mirror-интерфейсу, для sFLOW, NetFlow он недоступен.

Данный режим требует несколько иной конфигурации https://github.com/FastVPSEestiOu/fastnetmon/blob/master/docs/BGP_FLOW_SPEC.md и должен использовать отдельно от режима BGP.

Также для работы данного режима нужно активировать сбор всех данных об атаке в формате pcap: collect_attack_pcap_dumps = on, а также включить их обработку с помощью DPI: process_pcap_attack_dumps_with_dpi = on.

В данном случае при обнаружении атаки, если она будет идентифицирована будет поднят BGP Flow Spec анонс против данной атаки.

В данный момент поддерживаются только следующие типы атак и действия по их отражению:
- DNS amplification (блокируем весь udp traffic с порта 53)
- NTP amplification (блокируем весь udp traffic с порта 123)
- SSDP amplification (блокируем весь udp traffic с порта 1900)
- SNMP amplification (блокируем весь udp traffic с порта 161)

Стоит обратить внимание, что в данный момент не поддерживается разблокировка правил Flow Spec и она должна осуществляться вручную путем передачи команд BGP демону. Это планируется исправить в следующем релизе.

Мне нужна помощь!

Есть несколько вариантов получения помощи по проекту:

- Рассылка пользователей https://groups.google.com/forum/#!forum/fastnetmon
- Официальный бактерией проекта на GitHub: http://bit.ly/fastnetmon
- Twitter автора: https://twitter.com/odintsov_pavel
- Прочтение исходного кода :)
- IRC канал: #fastnetmon на сервере irc.freenode.net
- В случае, если предыдущие шаги не помогли email автора: pavel.odintsov@gmail.com