



FastNetMonを試してみた

(FastNetMon 1.1.3)

@ishizaghi

自己紹介

■石崎豊

フリービット株式会社 R&D

■JANOGとわたし

#	場所	内容
JANOG27.5	愛宕	Hyper Giants CDNキャッシュサーバーの ISPネットワークへの効果と課題
JANOG28	日本橋	会場ネットワーク提供 (フレッツIPv6 PPPoE, 64フォールバック環境)
JANOG29	和歌山	プログラム委員長

FastNetMonとは

■パケットキャプチャやNetFlowデータをもとにDDoSを高速検知するオープンソースソフトウェア

■RIPE71, NANOG66で紹介された

■NANOG66 (Feb, 2016)

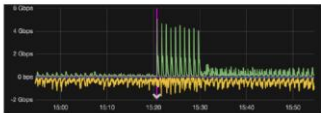
■Research and Education Track

“An open source recipe for detecting DDoS and automate mitigation techniques”

[Vicente De Luca, Zendesk]

DDoS Attacks

An open-source recipe to improve fast detection and automate mitigation techniques



Vicente De Luca
Sr. Network Engineer
vdeluca@zendesk.com
AS21880 / AS61186

FastNetMon: very fast DDoS analyzer

- collects sFlow (v4/v5), NetFlow (v5/v9/v10), IPFIX and SPAN/mirror
- fast detect IPv4 host above certain threshold
- feed Graphite (compatible) time-series DB
- supports BGP daemons (ExaBGP, GoBGP, others)
- supports Lua processing net flows
- CLI client

```
FastNetMon v1.0.0 (2016-02-10)
Copyright (c) 2016 Pavel Odintsov
All rights reserved.

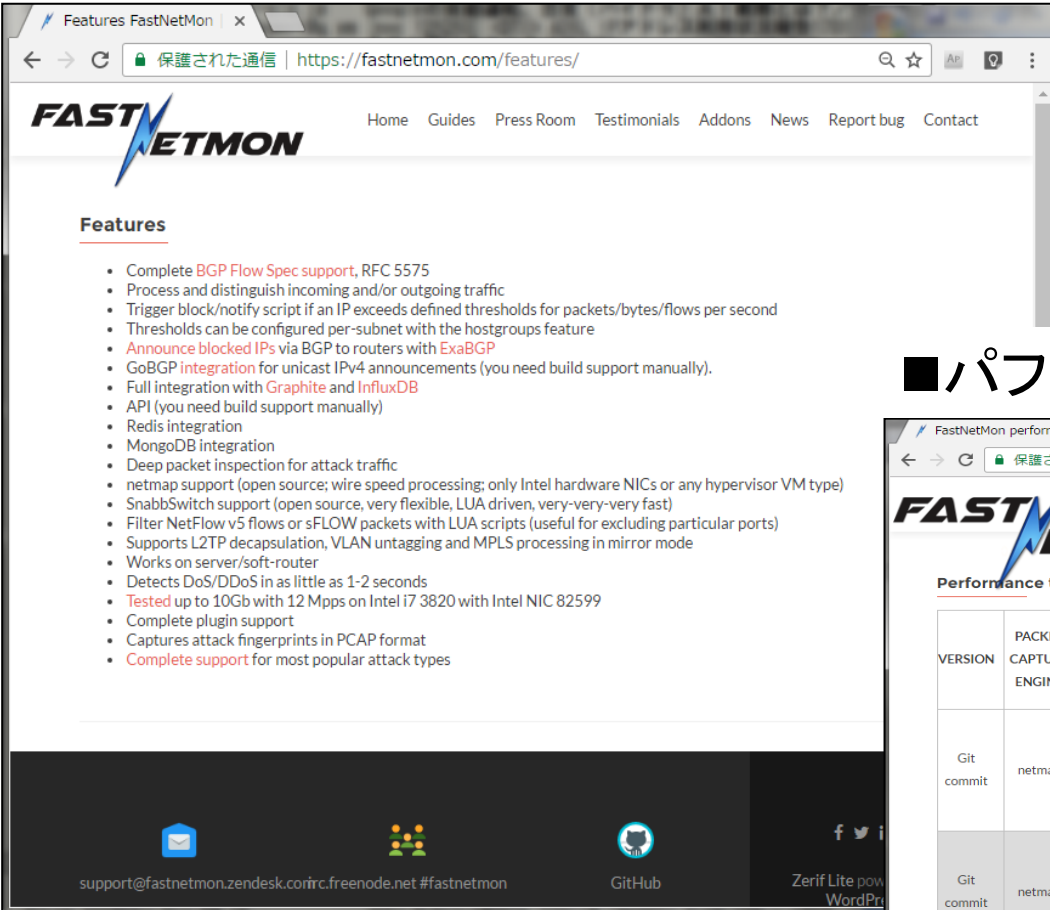
Usage: fastnetmon [options]
Options:
  -h, --help            show this help message and exit
  -c, --config FILENAME configuration file
  -d, --daemon           run as daemon
  -e, --engine ENGINE    engine to use (sflow, netflow, ipfix, span)
  -f, --flow-id ID       flow id
  -i, --interface IFACE interface to use
  -l, --lua LUA_SCRIPT   lua script to process net flows
  -m, --mirror MIRROR    mirror to use (eth0, eth1, eth2, eth3, eth4, eth5, eth6, eth7, eth8, eth9, eth10, eth11, eth12, eth13, eth14, eth15, eth16, eth17, eth18, eth19, eth20, eth21, eth22, eth23, eth24, eth25, eth26, eth27, eth28, eth29, eth30, eth31, eth32, eth33, eth34, eth35, eth36, eth37, eth38, eth39, eth40, eth41, eth42, eth43, eth44, eth45, eth46, eth47, eth48, eth49, eth50, eth51, eth52, eth53, eth54, eth55, eth56, eth57, eth58, eth59, eth60, eth61, eth62, eth63, eth64, eth65, eth66, eth67, eth68, eth69, eth70, eth71, eth72, eth73, eth74, eth75, eth76, eth77, eth78, eth79, eth80, eth81, eth82, eth83, eth84, eth85, eth86, eth87, eth88, eth89, eth90, eth91, eth92, eth93, eth94, eth95, eth96, eth97, eth98, eth99, eth100)
  -n, --netflow NETFLOW netflow engine to use (v5, v9, v10)
  -o, --output OUTPUT    output directory
  -p, --port PORT        port to listen on
  -s, --sflow SFLOW     sflow engine to use (v4, v5)
  -t, --threshold THRESHOLD threshold to detect DDoS
  -v, --version          show program version
  -w, --worker WORKER   number of workers to run
```

available for CentOS / Ubuntu / Debian / Vyatta / FreeBSD / source / Docker Image
tested with Juniper, Cisco, Extreme, Huawei and Linux (ipt_NETFLOW)

<https://github.com/pavel-odintsov/fastnetmon>

FastNetMonとは

■主な機能

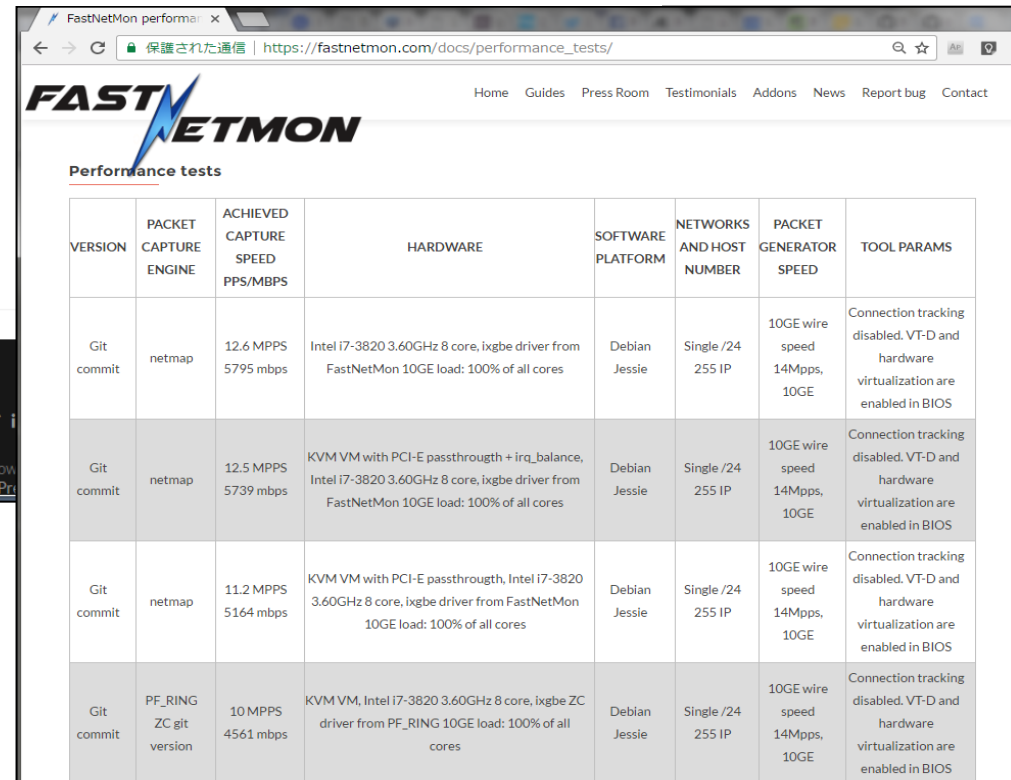


The screenshot shows the 'Features' page of the FastNetMon website. The page lists various capabilities of the tool, including support for BGP Flow Spec, traffic processing, threshold-based blocking, and integration with various databases and protocols. The footer contains contact information and social media links.

- Complete [BGP Flow Spec support](#), RFC 5575
- Process and distinguish incoming and/or outgoing traffic
- Trigger block/notify script if an IP exceeds defined thresholds for packets/bytes/flows per second
- Thresholds can be configured per-subnet with the hostgroups feature
- [Announce blocked IPs](#) via BGP to routers with [ExaBGP](#)
- [GoBGP integration](#) for unicast IPv4 announcements (you need build support manually).
- Full integration with [Graphite](#) and [InfluxDB](#)
- API (you need build support manually)
- Redis integration
- MongoDB integration
- Deep packet inspection for attack traffic
- netmap support (open source; wire speed processing; only Intel hardware NICs or any hypervisor VM type)
- SnabbSwitch support (open source, very flexible, LUA driven, very-very-fast)
- Filter NetFlow v5 flows or sFLOW packets with LUA scripts (useful for excluding particular ports)
- Supports L2TP decapsulation, VLAN untagging and MPLS processing in mirror mode
- Works on server/soft-router
- Detects DoS/DDoS in as little as 1-2 seconds
- [Tested](#) up to 10Gb with 12 Mpps on Intel i7 3820 with Intel NIC 82599
- Complete plugin support
- Captures attack fingerprints in PCAP format
- [Complete support](#) for most popular attack types

support@fastnetmon.zendesk.com | irc.freenode.net #fastnetmon | GitHub | Zerif Lite powered by WordPress

■パフォーマンス

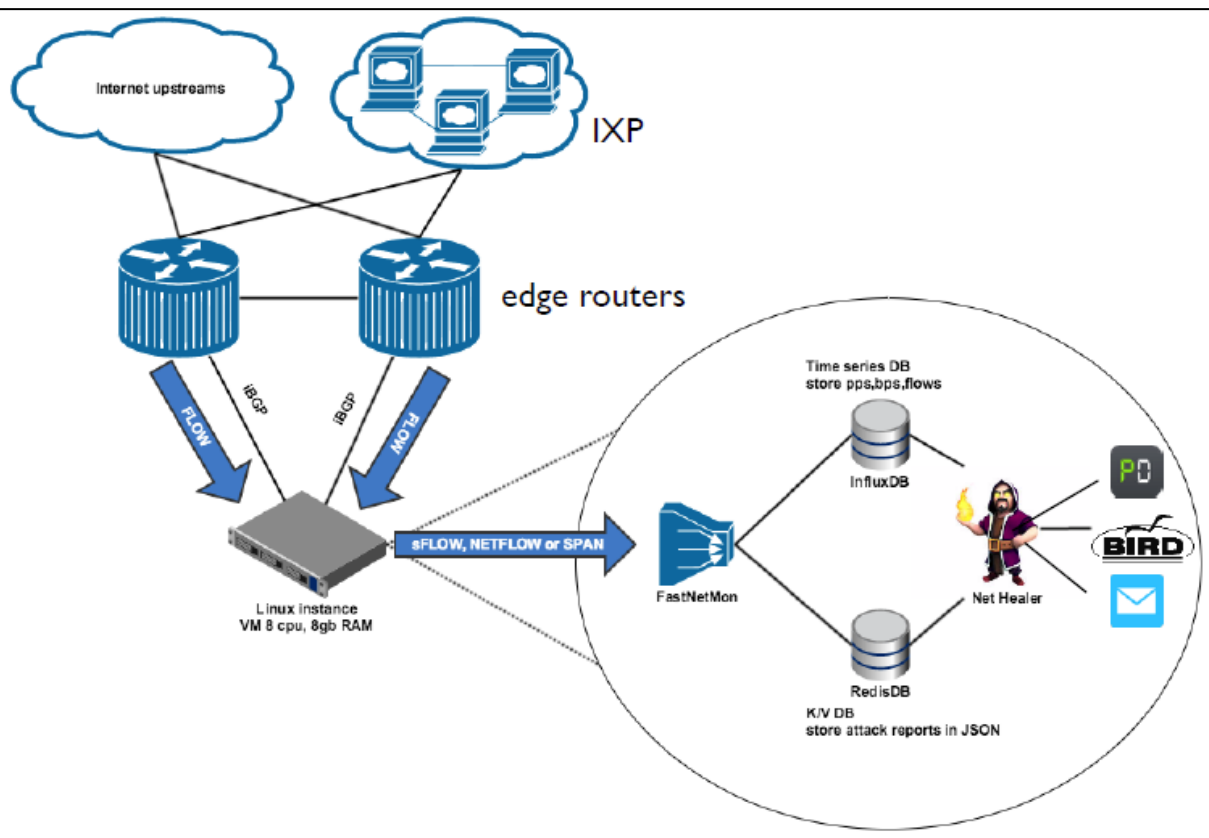


The screenshot shows the 'Performance tests' page of the FastNetMon website. It features a table with performance metrics for different hardware and software configurations. The table includes columns for Version, Packet Capture Engine, Achieved Capture Speed, Hardware, Software Platform, Networks and Host Number, Packet Generator Speed, and Tool Params.

VERSION	PACKET CAPTURE ENGINE	ACHIEVED CAPTURE SPEED PPS/MBPS	HARDWARE	SOFTWARE PLATFORM	NETWORKS AND HOST NUMBER	PACKET GENERATOR SPEED	TOOL PARAMS
Git commit	netmap	12.6 MPPS 5795 mbps	Intel i7-3820 3.60GHz 8 core, ixgbe driver from FastNetMon 10GE load: 100% of all cores	Debian Jessie	Single /24 255 IP	10GE wire speed 14Mpps, 10GE	Connection tracking disabled. VT-D and hardware virtualization are enabled in BIOS
Git commit	netmap	12.5 MPPS 5739 mbps	KVM VM with PCI-E passthrough + irq_balance, Intel i7-3820 3.60GHz 8 core, ixgbe driver from FastNetMon 10GE load: 100% of all cores	Debian Jessie	Single /24 255 IP	10GE wire speed 14Mpps, 10GE	Connection tracking disabled. VT-D and hardware virtualization are enabled in BIOS
Git commit	netmap	11.2 MPPS 5164 mbps	KVM VM with PCI-E passthrough, Intel i7-3820 3.60GHz 8 core, ixgbe driver from FastNetMon 10GE load: 100% of all cores	Debian Jessie	Single /24 255 IP	10GE wire speed 14Mpps, 10GE	Connection tracking disabled. VT-D and hardware virtualization are enabled in BIOS
Git commit	PF_RING ZC git version	10 MPPS 4561 mbps	KVM VM, Intel i7-3820 3.60GHz 8 core, ixgbe ZC driver from PF_RING 10GE load: 100% of all cores	Debian Jessie	Single /24 255 IP	10GE wire speed 14Mpps, 10GE	Connection tracking disabled. VT-D and hardware virtualization are enabled in BIOS

FastNetMonとは

■アーキテクチャ



Opensource recipe

- **FastNetMon**: main core of our solution. DDoS analyzer with sflow/netflow/mirror support
- **InfluxDB**: Scalable data store for metrics, events, and real-time analytics
- **Grafana**: Gorgeous metric viz, dashboards & editors
- **Redis**: An in-memory database that persists on disk
- **Morgoth**: Metric anomaly detection for Influx databases
- **BIRD**: a fully functional dynamic IP routing daemon
- **Net Healer**: experimental code to "glue" all moving parts, trigger actions and provide API queries



FastNetMonとは

■アタック検知ロジック

FastNetMon

Detection Logic:

- number of **pps, mbps and flows** to/from a /32
- number of **fragmented packets** to/from a /32
- number of **tcp syn / udp** to/from a /32
- global / per protocol (udp/tcp/icmp) / per host group (CIDR)
- nDPI support (SPAN/mirror)

Complete support most popular attacks for channel overflow:

- **SYN Flood**
- **UDP Flood** (amplified SSDP, Chargen, DNS, SNMP, NTP, etc)
- **IP Fragmentation**

■アタック検知時の反応

FastNetMon

How it can react during an attack ?

- Custom script (send email, apply an ACL, shutdown a VM, etc etc etc...)
- BGP Announce (community, blackhole, selective blackhole, cloud mitigation)
- BGP Flow Spec (**RFC 5575**) for selective traffic blocking
- Populate Redis DB (target, type, attack peak, tcpdump during attack, etc)

インストール、基本設定、起動 (Ubuntu16)

■ インストール

```
$ wget https://raw.githubusercontent.com/pavel-odintsov/fastnetmon/master/src/fastnetmon_install.pl -fastnetmon_install.pl  
$ sudo perl fastnetmon_install.pl
```

■ 設定

```
# vi /etc/networks_list  
203.0.113.0/24  
198.51.100.0/24
```

```
# vi /etc/fastnetmon.conf  
後述....
```

■ 起動

```
# systemctl start fastnetmon
```

設定

■ /etc/fastnetmon.conf (抜粋)

```
# Different approaches to attack detection
ban_for_pps = on
ban_for_bandwidth = on
ban_for_flows = off

# Limits for Dos/DDoS attacks
threshold_pps = 20000
threshold_mbps = 1000
threshold_flows = 3500

# Per protocol attack thresholds
# We don't implement per protocol flow limits, sorry :(
# These limits should be smaller than global pps/mbps limits
```

総トラフィックのしきい値
設定 (pps, mbps,
flows)

```
ban_for_tcp_bandwidth = off
ban_for_udp_bandwidth = off
ban_for_icmp_bandwidth = off

threshold_tcp_mbps = 100000
threshold_udp_mbps = 100000
threshold_icmp_mbps = 100000
```

プロトコル毎のしきい値
設定 (mbps)

```
ban_for_tcp_pps = off
ban_for_udp_pps = off
ban_for_icmp_pps = off

threshold_tcp_pps = 100000
threshold_udp_pps = 100000
threshold_icmp_pps = 100000
```

プロトコル毎のしきい値
設定 (pps)

つづき..

```
##
### Actions when attack detected
###

# This script executed for ban, unban and attack detail collection
notify_script_path = /usr/local/bin/notify_about_attack.sh

# announce blocked IPs with BGP protocol with ExaBGP
exabgp = off
exabgp_command_pipe = /var/run/exabgp.cmd
exabgp_community = 65001:666
```


アタック検知時に起動するscript

■ /usr/local/bin/notify_about_attack.sh

```
#!/usr/bin/env bash

# This script will get following params:
# $1 client_ip_as_string
# $2 data_direction
# $3 pps_as_string
# $4 action (ban or unban)

email_notify="root, please_fix_this_email@domain "

#
# Please be careful! You should not remove cat >
#

if [ "$4" = "unban" ]; then
    # No details arrived to stdin here

    # Unban actions if used
    exit 0
fi

if [ "$4" = "ban" ]; then
    cat | mail -s "FastNetMon Guard: IP $1 blocked because $2 attack with power $3 pps" $email_notify;
    # You can add ban code here!
    exit 0
fi

if [ "$4" == "attack_details" ]; then
    cat | mail -s "FastNetMon Guard: IP $1 blocked because $2 attack with power $3 pps" $email_notify;

    exit 0
fi
```

オプションパラメーター例

\$1 = 203.0.113.123

\$2 = incoming / outgoing

\$3 = 1000521 (string)

\$4 = ban / unban / attack_details

CLIモニタリング

■ /opt/fastnetmon_client

```
# /opt/fastnetmon/fastnetmon_client
```

```
FastNetMon 1.1.3 master git-94f4947e87753b8be193ca54d17dac24cac599fb Pavel Odintsov: stableit.ru
```

```
IPs ordered by: packets
```

```
Incoming traffic      3167 pps   36 mbps   0 flows
203.0.113.164        3166 pps   36 mbps   0 flows *banned*
203.0.113.165         6 pps     0 mbps    0 flows
```

```
Outgoing traffic     1152 pps   0 mbps    0 flows
203.0.113.164       1151 pps   0 mbps    0 flows *banned*
203.0.113.165        4 pps     0 mbps    0 flows
```

```
Internal traffic     0 pps     0 mbps
```

```
Other traffic        0 pps     0 mbps
```

```
Screen updated in:   0 sec 340 microseconds
```

```
Traffic calculated in: 0 sec 180 microseconds
```

```
Total amount of IPv6 packets related to our own network: 0
```

```
Not processed packets: 0 pps
```

```
Ban list:
```

```
203.0.113.164/4725 pps incoming at 13_01_17_15:28:57
```

```
Subnet load:
```

```
203.0.113.160/29 pps in: 0      out: 0      mbps in: 0      out: 0
203.0.113.168/29 pps in: 0      out: 0      mbps in: 0      out: 0
203.0.113.174/32 pps in: 0      out: 0      mbps in: 0      out: 0
```

InfluxDBにストアされるテーブル、タグキー



Write Data Documentation

Database: graphite

Query: show measurements

measurements

name

hosts

networks

total



Write Data Documentation

Database: graphite

Query: show tag keys

Generate Query URL

Query Templates

hosts

tagKey

app

cidr

direction

function

resource

networks

tagKey

app

cidr

direction

resource

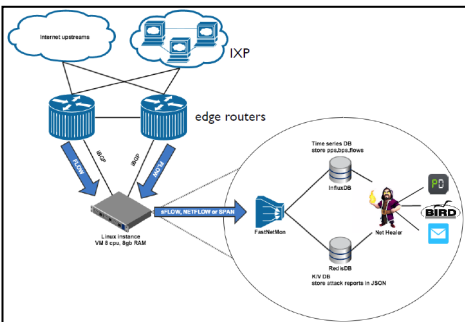
total

tagKey

app

direction

resource



InfluxDBにストアされるレコード

Query: select * from total limit 5

Generate Query URL Query Templates

total

time	app	direction	resource	value
2017-01-12T06:11:25Z	"fastnetmon"	"incoming"	"bps"	1744
2017-01-12T06:11:25Z	"fastnetmon"	"outgoing"	"pps"	0
2017-01-12T06:11:25Z	"fastnetmon"	"incoming"	"pps"	0
2017-01-12T06:11:25Z	"fastnetmon"	"incoming"	"flows"	0
2017-01-12T06:11:25Z	"fastnetmon"	"outgoing"	"flows"	0

Generate Query URL Query Templates

networks

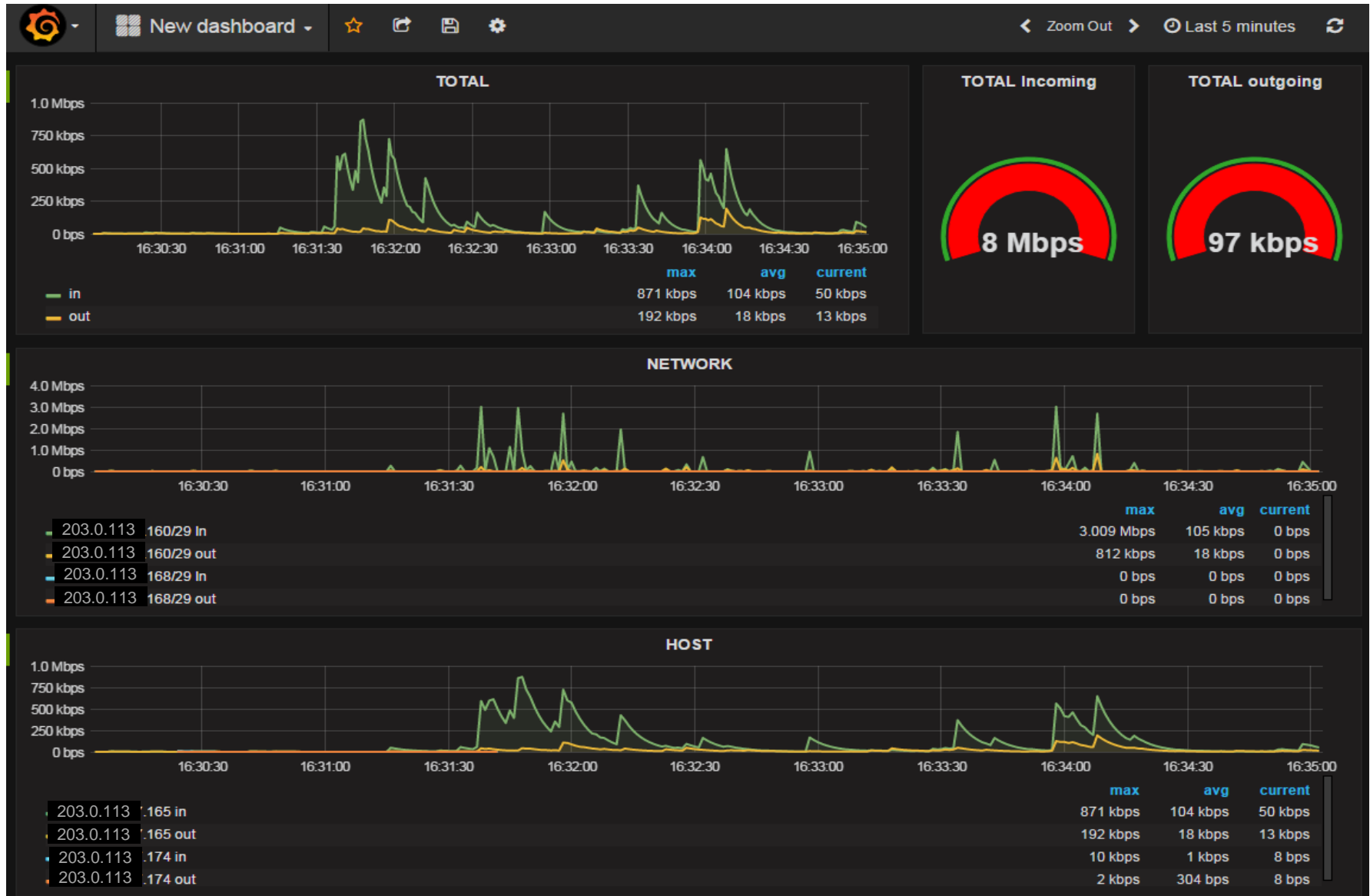
time	app	cidr	direction	resource	value
2017-01-12T06:11:25Z	"fastnetmon"	203.0.113 _160_29"	"incoming"	"bps"	1720
2017-01-12T06:11:25Z	"fastnetmon"	203.0.113 _174_32"	"outgoing"	"pps"	0
2017-01-12T06:11:25Z	"fastnetmon"	203.0.113 _160_29"	"outgoing"	"bps"	1256
2017-01-12T06:11:25Z	"fastnetmon"	203.0.113 _168_29"	"outgoing"	"bps"	0
2017-01-12T06:11:25Z	"fastnetmon"	203.0.113 _168_29"	"incoming"	"pps"	0

Generate Query URL Query Templates

hosts

time	app	cidr	direction	function	resource	value
2017-01-12T06:11:25Z	"fastnetmon"	203.0.113 _174"	"outgoing"	"average"	"bps"	56
2017-01-12T06:11:25Z	"fastnetmon"	203.0.113 _174"	"incoming"	"average"	"bps"	56
2017-01-12T06:11:25Z	"fastnetmon"	203.0.113 _165"	"outgoing"	"average"	"bps"	1144
2017-01-12T06:11:25Z	"fastnetmon"	203.0.113 _165"	"incoming"	"average"	"bps"	1680
2017-01-12T06:11:26Z	"fastnetmon"	203.0.113 _165"	"outgoing"	"average"	"bps"	3928

Grafanaで可視化すると



情報量少ない..

redisにストアされるレコード (アタック検知時の情報)

```
$ redis-cli keys fastnetmon *
```

```
fastnetmon_203.0.113.162_packets_dump  
fastnetmon_203.0.113.162_information  
fastnetmon_203.0.113.174_packets_dump  
fastnetmon_203.0.113.174_information  
fastnetmon_203.0.113.165_packets_dump  
fastnetmon_203.0.113.165_information
```

```
$ redis-cli get fastnetmon_203.0.113.165_information
```

```
{ "ip": "203.0.113.165", "attack_details": { "attack_type": "udp_flood",  
"initial_attack_power": 515, "peak_attack_power": 515, "attack_direction": "incoming",  
"attack_protocol": "udp", "total_incoming_traffic": 670292, "total_outgoing_traffic":  
26985, "total_incoming_pps": 515, "total_outgoing_pps": 276,  
"total_incoming_flows": 0, "total_outgoing_flows": 0, "average_incoming_traffic":  
670292, "average_outgoing_traffic": 26985, "average_incoming_pps": 515,  
"average_outgoing_pps": 276, "average_incoming_flows": 0,  
"average_outgoing_flows": 0, "incoming_ip_fragmented_traffic": 0,  
"outgoing_ip_fragmented_traffic": 0, "incoming_ip_fragmented_pps": 0,  
"outgoing_ip_fragmented_pps": 0, "incoming_tcp_traffic": 6340,  
"outgoing_tcp_traffic": 1277, "incoming_tcp_pps": 8, "outgoing_tcp_pps": 7,  
"incoming_syn_tcp_traffic": 2906, "outgoing_syn_tcp_traffic": 159,  
"incoming_syn_tcp_pps": 6, "outgoing_syn_tcp_pps": 0, "incoming_udp_traffic":  
663951, "outgoing_udp_traffic": 25706, "incoming_udp_pps": 503,  
"outgoing_udp_pps": 267, "incoming_icmp_traffic": 0, "outgoing_icmp_traffic": 0,  
"incoming_icmp_pps": 0, "outgoing_icmp_pps": 0 }, "network_load": { "incoming  
traffic": 3685711, "outgoing traffic": 144473, "incoming pps": 2828, "outgoing pps":  
1514 }, "network_average_load": { "incoming traffic": 184833, "outgoing traffic": 8353,  
"incoming pps": 137, "outgoing pps": 73 } }
```

```
$ redis-cli get fastnetmon_203.0.113.165_packets_dump
```

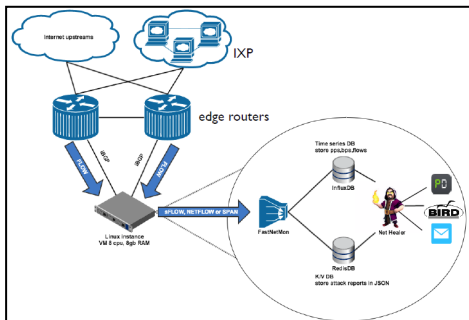
```
IP: 203.0.113.165  
Attack type: udp_flood  
Initial attack power: 510 packets per second  
Peak attack power: 510 packets per second  
Attack direction: incoming  
Attack protocol: udp  
Total incoming traffic: 4 mbps  
Total outgoing traffic: 0 mbps  
Total incoming pps: 510 packets per second  
Total outgoing pps: 310 packets per second  
Total incoming flows: 0 flows per second  
Total outgoing flows: 0 flows per second  
Average incoming traffic: 4 mbps  
Average outgoing traffic: 0 mbps  
Average incoming pps: 510 packets per second  
Average outgoing pps: 310 packets per second  
Average incoming flows: 0 flows per second  
Average outgoing flows: 0 flows per second
```

~ 省略 ~

```
Network: 203.0.113.160/29  
Network incoming traffic: 0 mbps  
Network outgoing traffic: 0 mbps  
Network incoming pps: 21 packets per second  
Network outgoing pps: 15 packets per second  
Average network incoming traffic: 1 mbps  
Average network outgoing traffic: 0 mbps  
Average network incoming pps: 151 packets per second  
Average network outgoing pps: 103 packets per second  
Average packet size for incoming traffic: 1102.8 bytes  
Average packet size for outgoing traffic: 124.6 bytes
```

```
2017-01-13 12:39:39.000000 216.58.197.193:443 > 203.0.113.165:61325 protocol: udp  
frag: 0 packets: 9 size: 11079 bytes ttl: 0 sample ratio: 1  
2017-01-13 12:39:39.000000 203.0.113.165:58099 > 172.217.27.66:443 protocol: udp  
frag: 0 packets: 6 size: 2542 bytes ttl: 0 sample ratio: 1  
2017-01-13 12:39:39.000000 172.217.27.66:443 > 203.0.113.165:58099 protocol: udp  
frag: 0 packets: 7 size: 3446 bytes ttl: 0 sample ratio: 1  
2017-01-13 12:39:39.000000 203.0.113.165:58411 > 174.129.255.59:443 protocol: tcp  
flags: syn,psh,ack frag: 0 packets: 4 size: 1414 bytes ttl: 0 sample ratio: 1  
2017-01-13 12:39:39.000000 203.0.113.165:58432 > 52.71.208.179:443 protocol: tcp  
flags: syn,psh,ack frag: 0 packets: 11 size: 12203 bytes ttl: 0 sample ratio: 1
```

つづ



まとめと所感

■ よいところ

- 高パフォーマンス
- 高速検知
- 検知時のアクションの作り込みが容易

■ もう少しなところ

- しきい値ベースの検知のみ
- TCP/UDPポート毎でのトラフィックの検知もしてほしいところ
- DBへ書き込む情報が少ない

(NetFlow v5 recordぶんかせめてTCP/UDPポート番号情報くらいはデータストアして欲しい)

■ 使い方としては..

- トラフィック可視化のデータ収集
 - いまいち..
- トラフィック異常検知
 - 異常検知のトリガーとしては使えそう
 - DDoS mitigationさせるためには他のコンポーネントと併用が良さそう
 - InfluxDB (Collection)
 - Chronograf (Visualization)
 - Kapacitor/Morgoth (Detection)
 - Relative value alert, Lossy Counting Algorithm(LSA)
 - BGP FlowSpec (Mitigation)

■ Chronograf / Kapacitor

