FURUKAWA ELECTRIC CO., LTD.

Open Networking Conference 2019

あなた好みの中継ルータ

2019/10/31

古河ネットワークソリューション株式会社
羽田野　文也

自己紹介

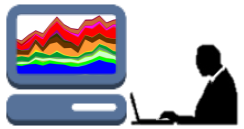
- 氏名
  - 羽田野　文也
- 所属
  - FNS 開発本部 技術企画開発部 先端技術開発グループ
  - 色々と新しいことをやっています
- 主に今までやってきたこと
  - ルータ装置の運用関連機能の開発
  - 構成管理用サーバシステムの開発/運用管理

ルータ上でFLOSSなアプリを使うメリット

■ ユーザ価値の観点から考えると…

価値	アプリの種類	具体的なメリット
設備コスト低減	•システム管理ツール •FW	•TCOを下げられる。機器代・ソフト代に加え電気代・ラック費用・保守費等 •管理者が少なくて済む
運用コスト低減	•構成管理ツール •ZTP •IDS	•自動化して人員を減らせる •人的ミスを減らせる •セキュリティ対策が可能
トラブル防止	•ネットワークフロー解析 •キャプチャ	•現地に行かずトラブルシュートが可能 •回線利用状況が把握できる •EUに報告しやすい
カスタマイズ	•見える化 •SE/キャリア開発アプリ	•より高度なサービスを提供できる •機器を追加することなくカスタマイズ

- **どんなアプリを動かしてみたいですか？**
- **どんなことをルータにやらせてみたいですか？**



いろいろなアプリケーションを動かしてみました

- パケットキャプチャ
  - tcpdump
- ネットワークフロー解析
  - ntopng, sflowd, hsflowd, packetbeat
- プロキシサーバ
  - squid
- IDS
  - snort
- ログ解析
  - filebeat

コンテナ型仮想環境の操作

Ubuntuベースのシステムコンテナ

```
Router#container attach
root@container:~#
root@container:~# cat /etc/lsb-release
ÐISTRIB_IÐ=Ubuntu
ÐISTRIB_RELEASE=18.04
ÐISTRIB_COÐENAME=bionic
ÐISTRIB_ÐESCRIPTION="Ubuntu 18.04.3 LTS"
root@container:~# pstree -A
systemd-+-accounts-daemon---2*[{accounts-daemon}]
        |-agetty
        |-atd
        |-chronyd
        |-cron
        |-dbus-daemon
        |-filebeat---8*[{filebeat}]
        |-hsflowd---{hsflowd}
        |-networkd-dispat---{networkd-dispat}
        |-polkitd---2*[{polkitd}]
        |-redis-server---3*[{redis-server}]
        |-rsyslogd---2*[{rsyslogd}]
        |-snort---{snort}
        |-softflowd
        |-squid---squid-+-log_file_daemon
        |               |-5*[security_file_c]
        |               `-unlinkd
        |-sshd
        |-systemd-journal
        |-systemd-logind
        |-systemd-network
        |-systemd-resolve
        |-systemd-udevd
        `-unattended-upgr---{unattended-upgr}
root@container:~#
```

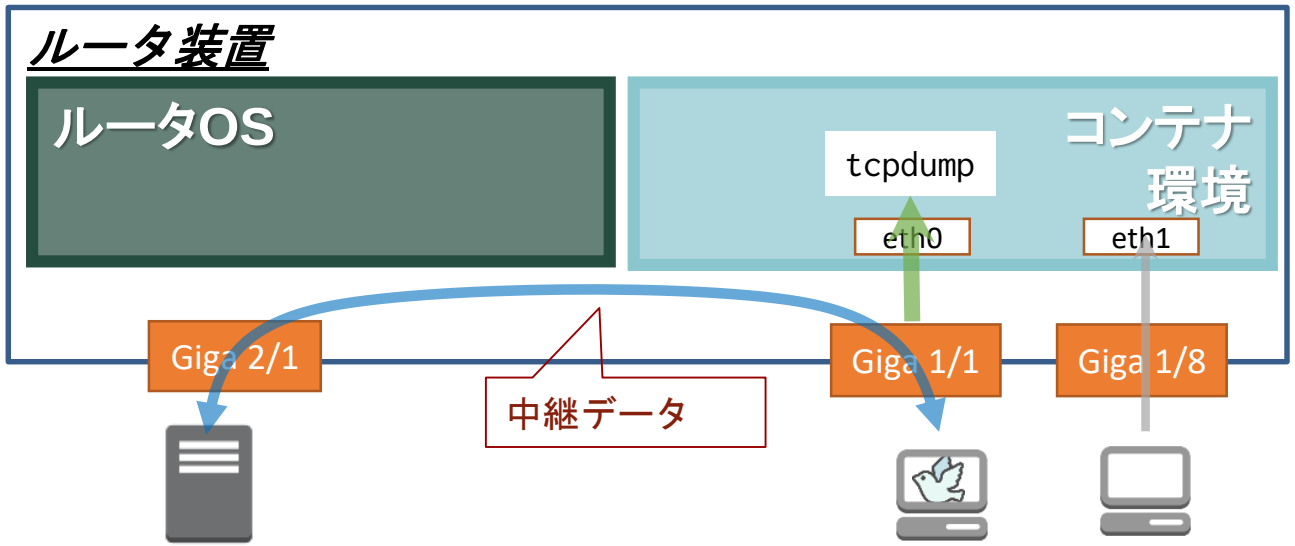
initデーモンを含めたOS全体が動作

複数のデーモンプロセスを同じコンテナ内で動作させる

パケットキャプチャ

- ルータのポートモニタリング機能を利用すれば中継データのキャプチャも可能

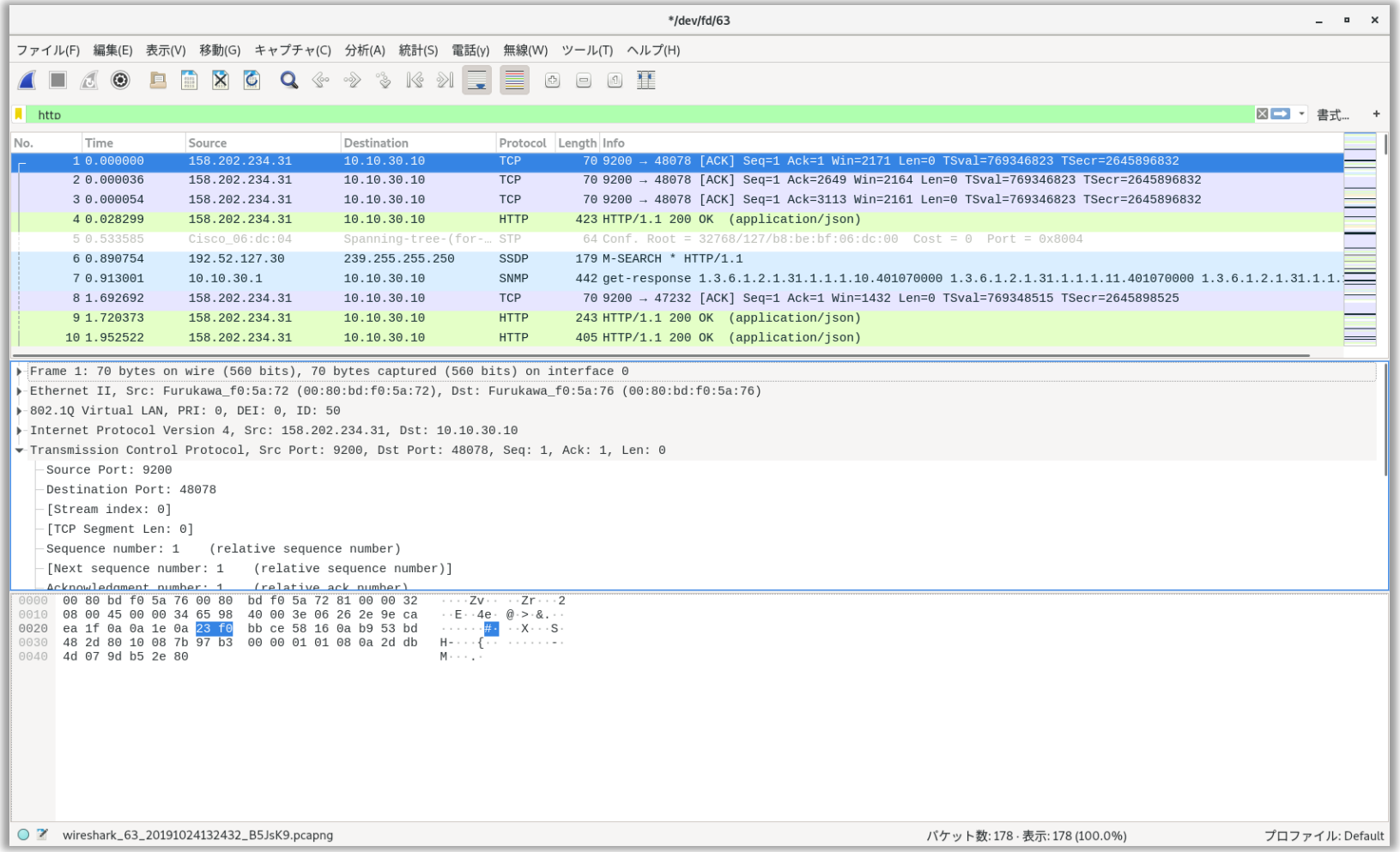
```
root@container:~# tcpdump -i eth0
06:29:48.611087 STP 802.1d, Config, Flags [none], bridge-id 807f.b8:be:bf:06:dc:00.8004,
length 43
06:29:49.691734 IP 192.168.127.30.65474 > nrt12s13-in-f195.1e100.net.http: Flags [R.], seq
1679839507, ack 3054297526, win 0, length 0
06:29:49.691797 IP 192.168.127.30.65457 > 117.18.237.29.http: Flags [R.], seq 0, ack 1, win 0,
length 0
06:29:49.691911 IP 192.168.127.30.65455 > 151.139.128.14.http: Flags [R.], seq 4273249763, ack
2972801286, win 0, length 0
06:29:50.610004 STP 802.1d, Config, Flags [none], bridge-id 807f.b8:be:bf:06:dc:00.8004,
length 43
:
```



パケットキャプチャ

- SSHサービスを利用すればWiresharkでみることも可能
  - コンテナ環境にGUIはないので外部端末から実行

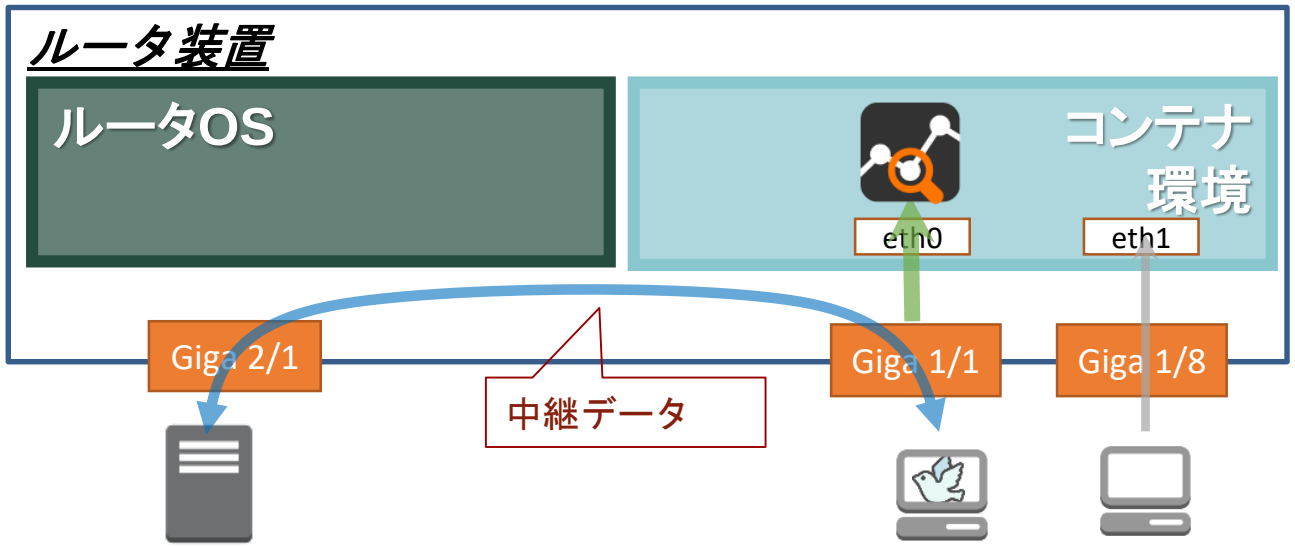
```
wireshark -k -i <(ssh root@192.168.127.21 "tcpdump -U -n -w - -i eth0 vlan and not port ssh")
```



ネットワークフロー解析(ntopng)

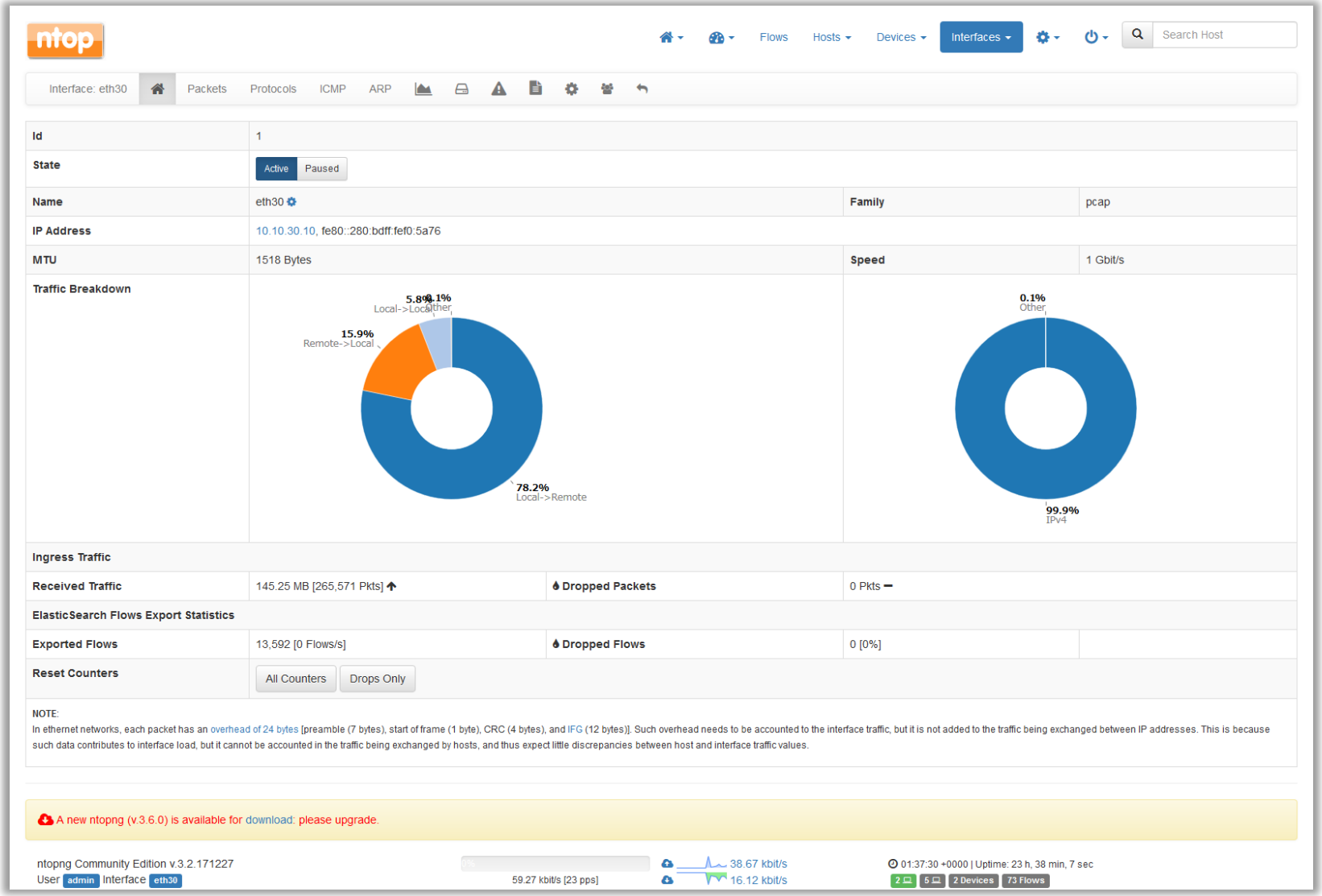
- DPI機能を使用して様々なネットワークプロトコルを判別可能

```
root@container:~# apt show ntopng
Package: ntopng
Version: 3.2+dfsg1-1
Priority: extra
Section: universe/net
Origin: Ubuntu
（省略）
root@container:~# apt install ntopng
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
（省略）
Setting up ntopng (3.2+dfsg1-1) ...
Processing triggers for libc-bin (2.27-3ubuntu1) ...
root@container:~# systemctl start ntopng
root@container:~#
```



ネットワークフロー解析(ntopng)

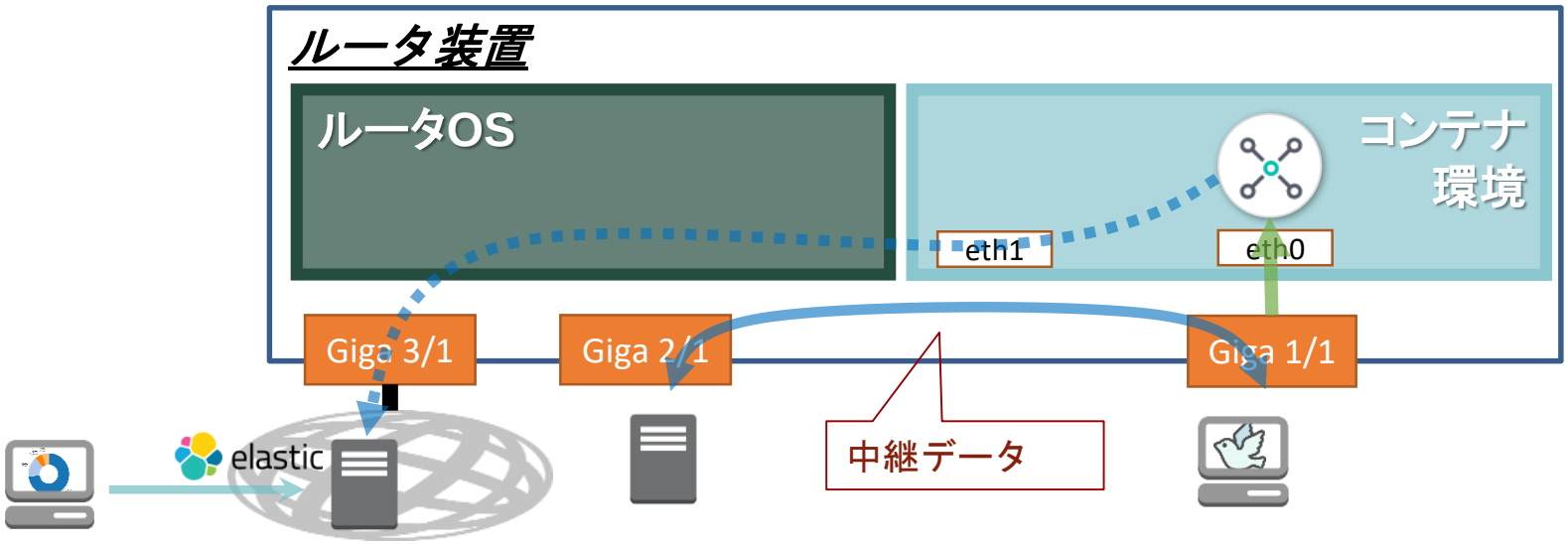
- NtopngのWebUIで解析結果を確認



ネットワークフローの見える化(packetbeat)

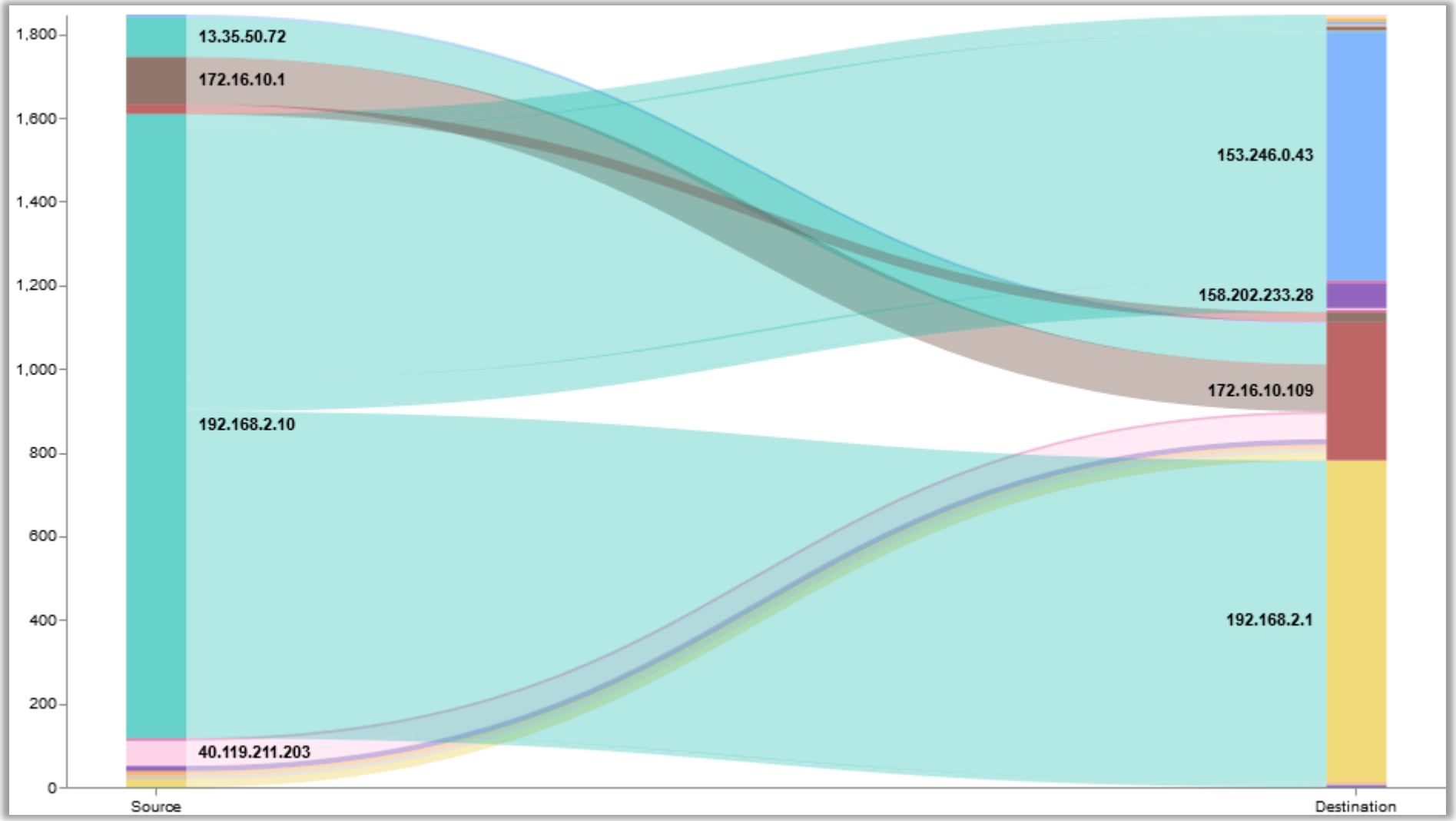
- Elastic社により開発されているパケットアナライザー
  - 外部サーバで動作するElasticsearchにデータを送信
  - Kibanaによるデータの可視化がリアルタイムに可能

```
root@container:~# systemctl start packetbeat
root@container:~# systemctl status packetbeat
  packetbeat.service - Real-Time Packet Analyzer
   Loaded: loaded (/usr/lib/systemd/system/packetbeat.service; disabled; vendor preset:
enabled)
   Active: active (running) since Sun 2019-10-20 09:04:14 UTC; 23h ago
     Docs: https://www.elastic.co/guide/en/beats/packetbeat/current/index.html
 Main PID: 7256 (packetbeat)
    Tasks: 9 (limit: 4275)
   CGroup: /system.slice/packetbeat.service
           mq7256 /usr/bin/packetbeat -c /etc/packetbeat/packetbeat.yml -path.home
/usr/share/packetbeat -path.config /etc/packetbeat -path.data /var/lib/packetbeat -path.logs
/var/log/packetbeat
```



Packetbeatによるフローの見える化

- Packetbeatから受信したデータをもとに可視化

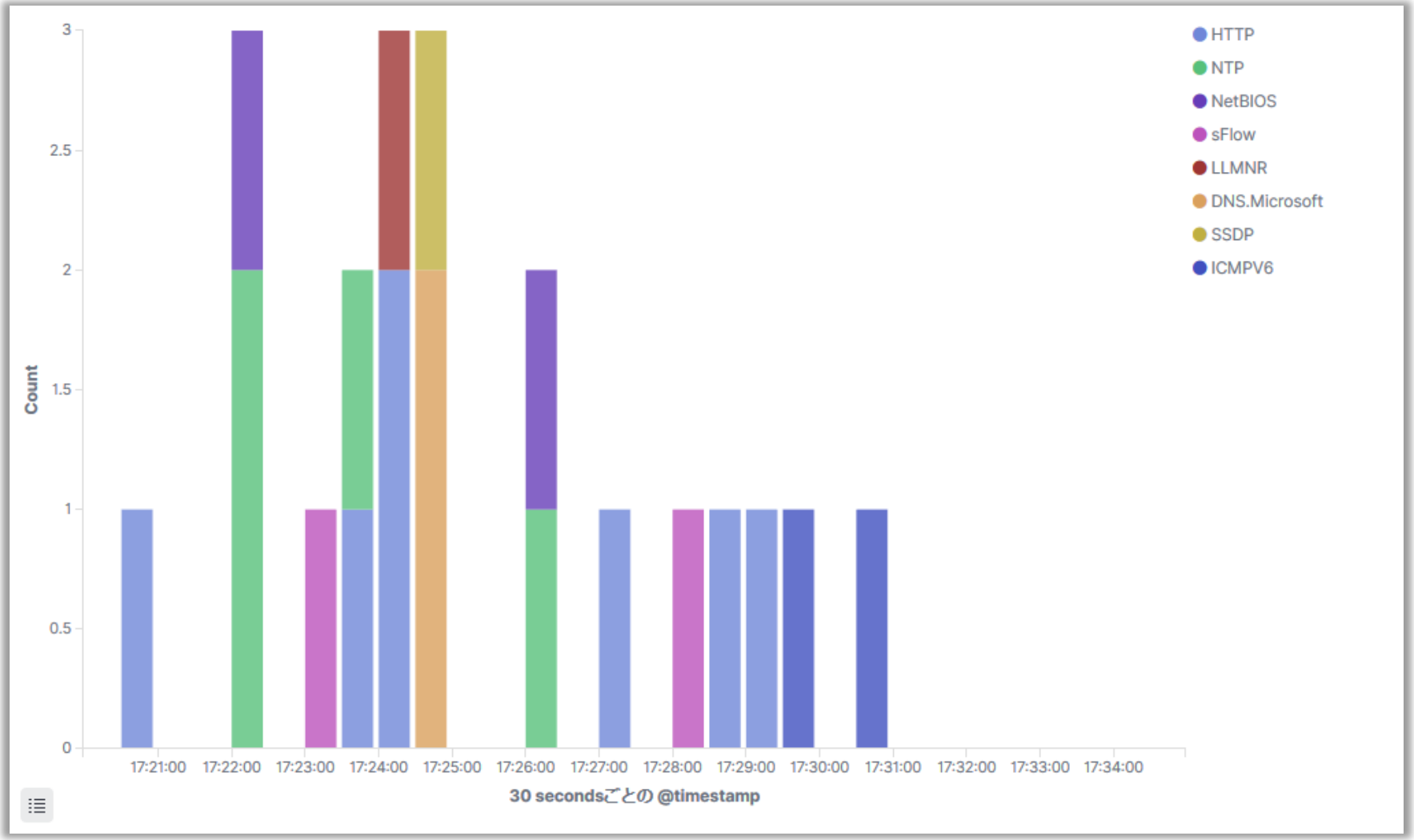


ntopngによるフローの見える化

- Ntopngの解析結果もKibanaで可視化

Dump expired flowsの設定

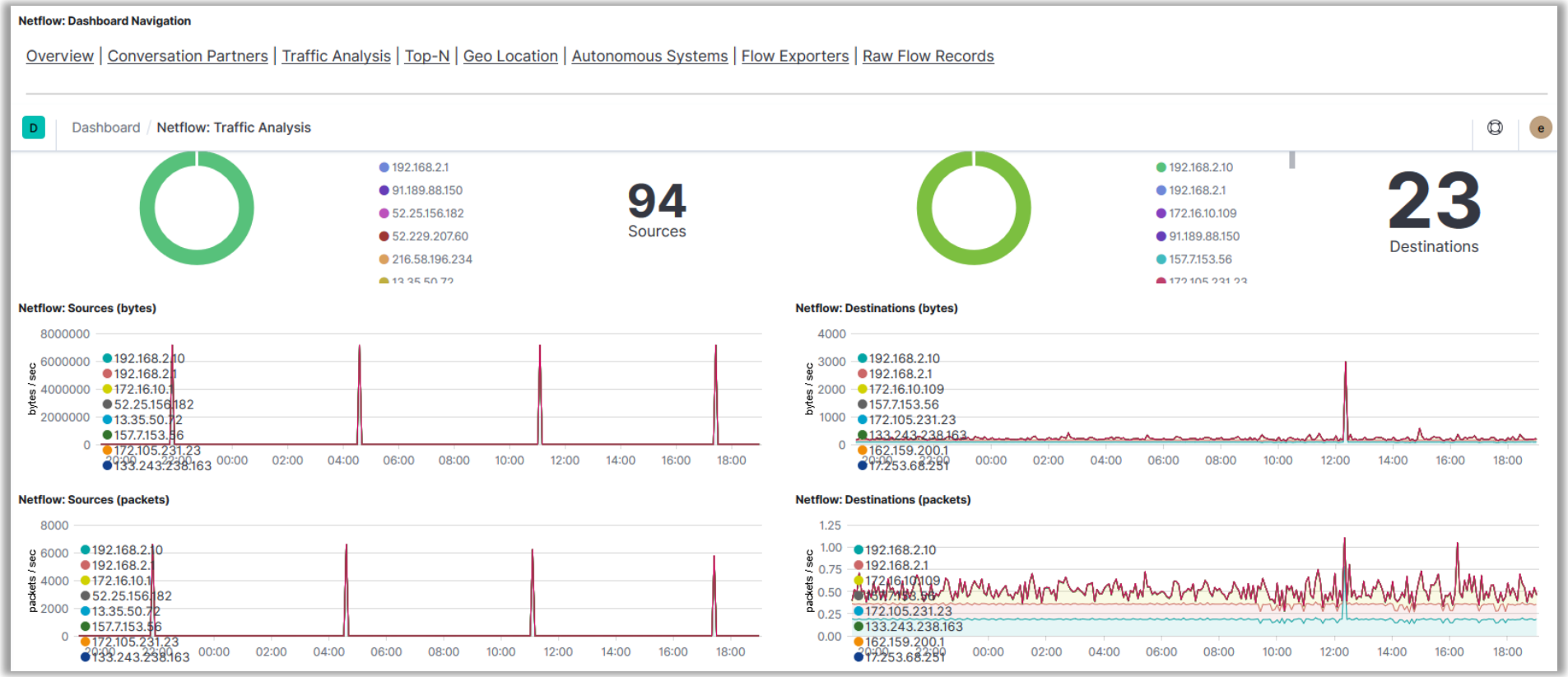
```
-F=es;_doc;ntopng-3.2;http://xxx.xxx.xxx.xxx:9200/_bulk;username:password
```



Netflowデータの可視化

- Softflowdを使用してNetflowのデータを外部サーバへ送信

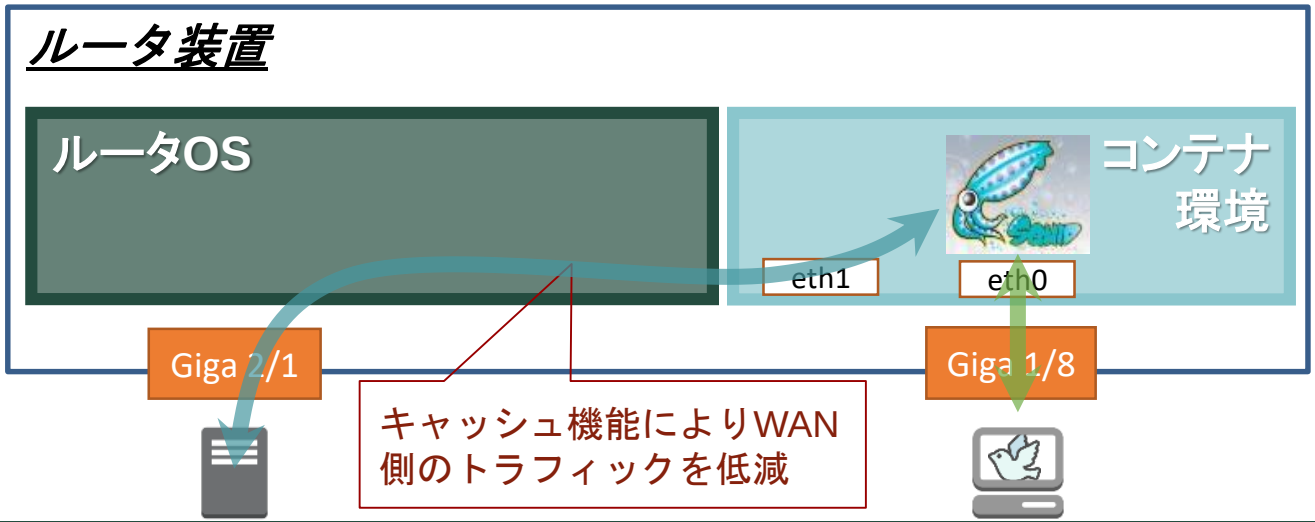
```
root@container:~# softflowd  -T ether -i eth0 -v 9 -n xxx.xxx.xxx.xxx:2055
```



プロキシサーバ

- Squidを使用したプロキシ＆キャッシュサーバ

```
root@container:~# systemctl start squid
root@container:~# iptables -t nat -A PREROUTING -i eth1 -p tcp -m tcp --dport 80 -j
REĐIRECT --to-ports 3128
root@container:~# iptables -t nat -A PREROUTING -i eth1 -p tcp -m tcp --dport 443 -j
REĐIRECT --to-ports 3129
root@container:~# tail /var/log/squid/access.log
1572202844.819     60 192.168.127.30 NONE/200 0 CONNECT 172.217.161.36:443 -
ORIGINAL_ĐST/172.217.161.36 -
1572202844.823   1021 192.168.127.30 TCP_MISS/200 49334 GET
https://media.stsci.edu/uploads/image/thumbnail/4316/low_STScI-H-1912b-t407x400.png -
ORIGINAL_ĐST/130.167.167.5 image/png
1572202844.965     67 192.168.127.30 TCP_MISS/302 867 GET https://www.google.com/ads/ga-
audiences? - ORIGINAL_ĐST/172.217.161.36 text/html
1572202845.028    805 192.168.127.30 TCP_MISS/200 62557 GET
https://media.stsci.edu/uploads/image/thumbnail/4291/low_STSCI-H-p1856a-t-400x400.png -
ORIGINAL_ĐST/130.167.167.5 image/png
```



透過型プロキシ

- HTTPS通信はSquidとPC間でTLSセッションを確立

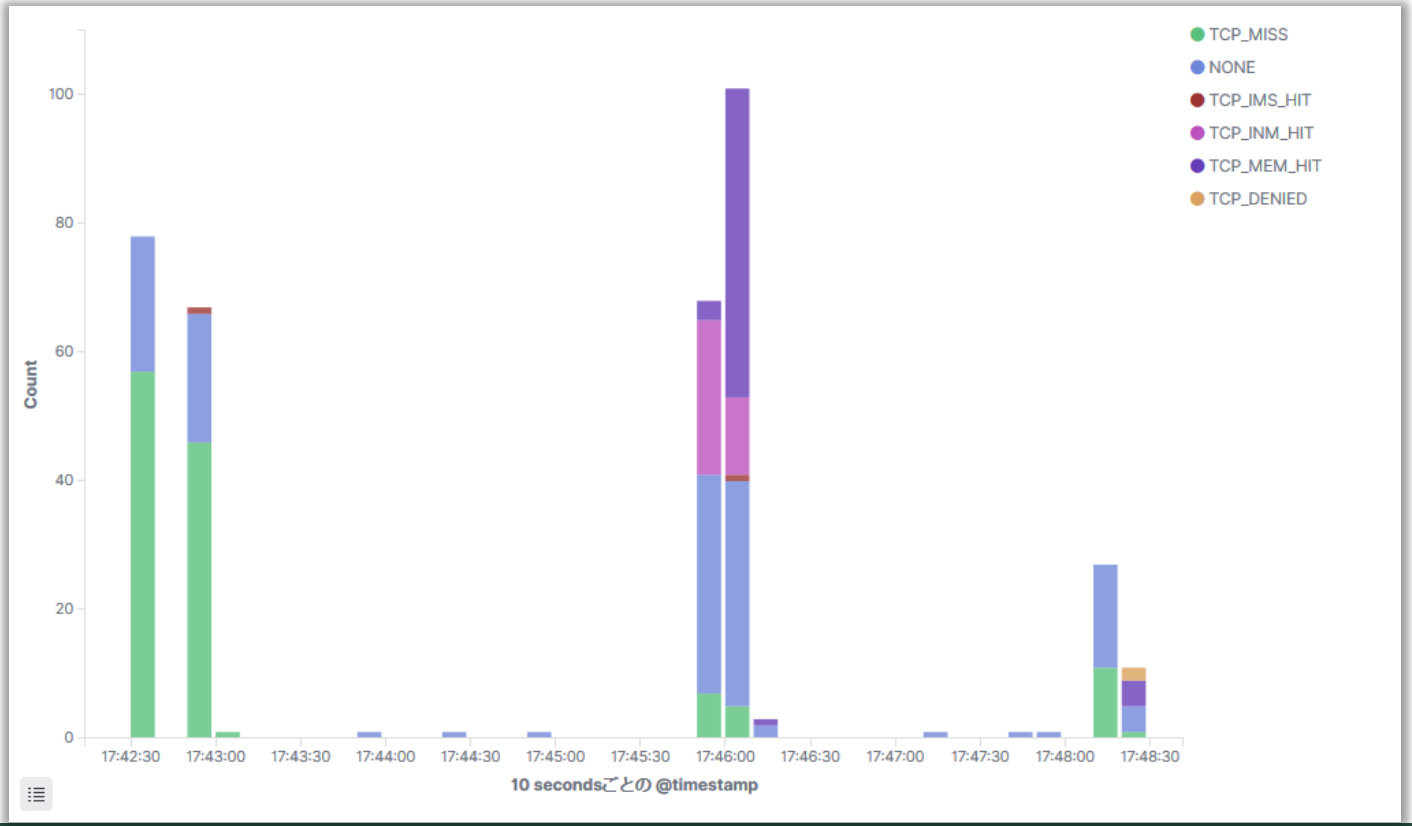
Squidのアクセスログを可視化

- Filebeatを使用してログファイルを外部サーバへ送信

```
root@container:~# systemctl start filebeat
```

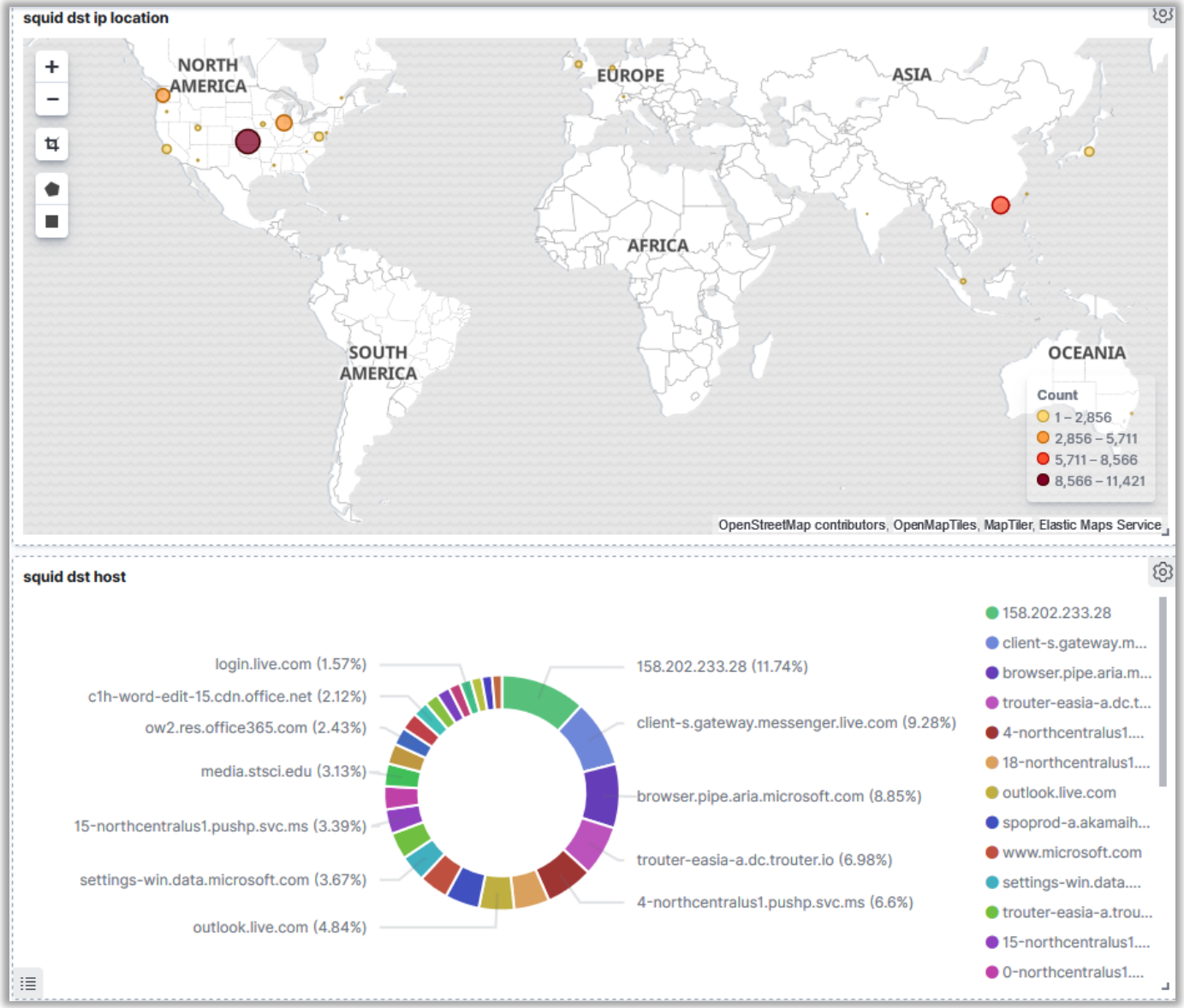
Grokフィルタの設定

```
"%{NUMBER:squid_date}%{SPACE}%{NUMBER:squid_res_time} (?:-
¦%{IP:squid_src_ip}) %{DATA:squid_req_stat}/%{NUMBER:squid_http_status} %{NUMBER:squid_repl
y_size} %{DATA:squid_http_method} %{DATA:squid_http_protocol}://%{IPORHOST:squid_dst_host}%
{NOTSPACE:squid_req_url} (?:-¦%{NOTSPACE:squid_user}) %{DATA:squid_hier_code}/(?:-
¦%{IP:squid_dst_ip}) %{NOTSPACE:squid_content_type}",
```



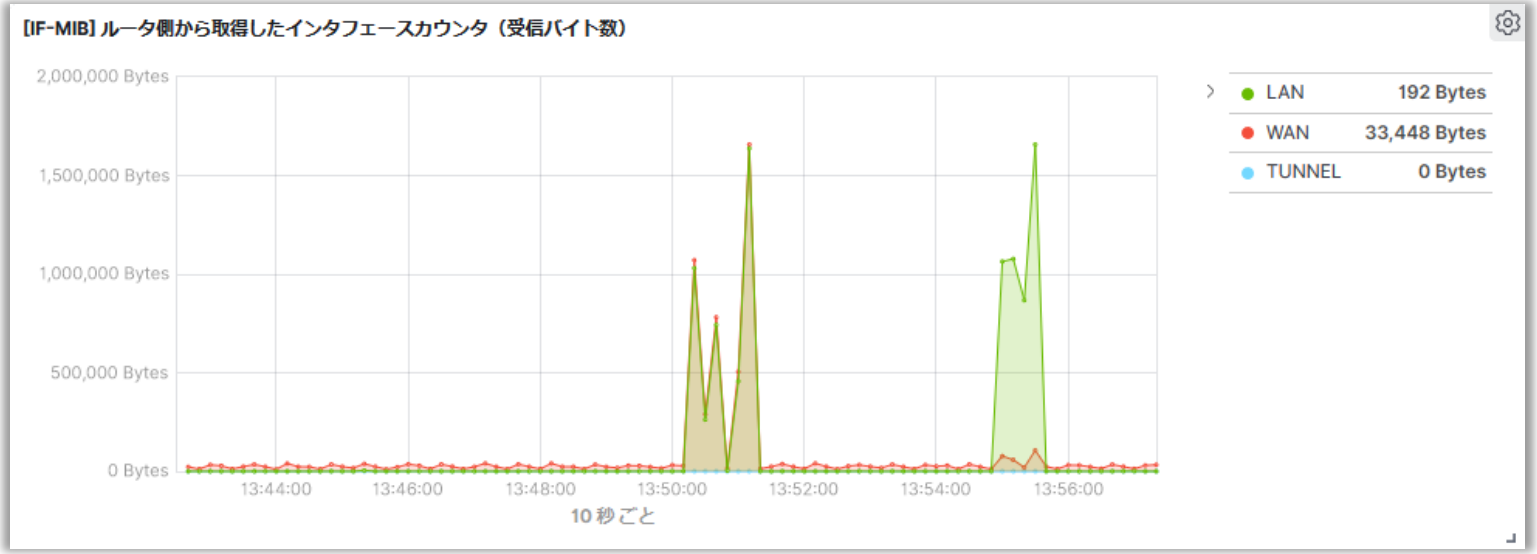
Squidのアクセス先を可視化

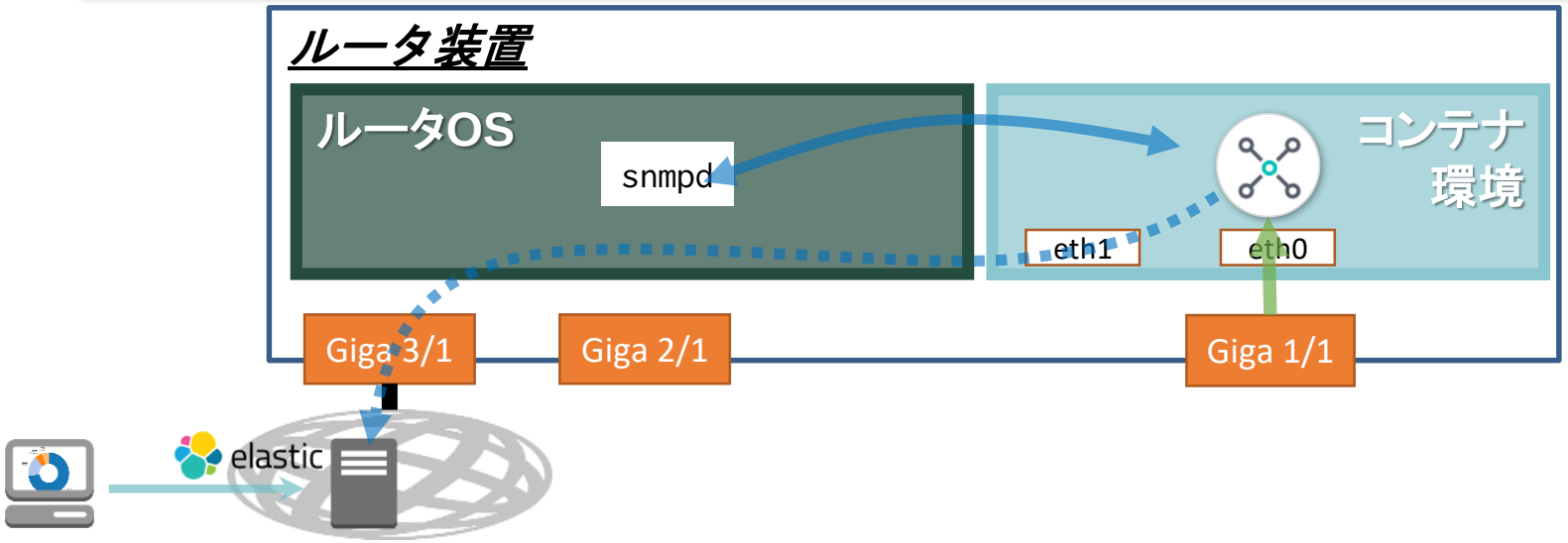
- GeoIPフィルタのプラグインを使用して地図上に可視化



インターフェースカウンタの可視化

- ルータOS側から取得したMIBデータをもとに可視化
  - データはfilebeatで送信





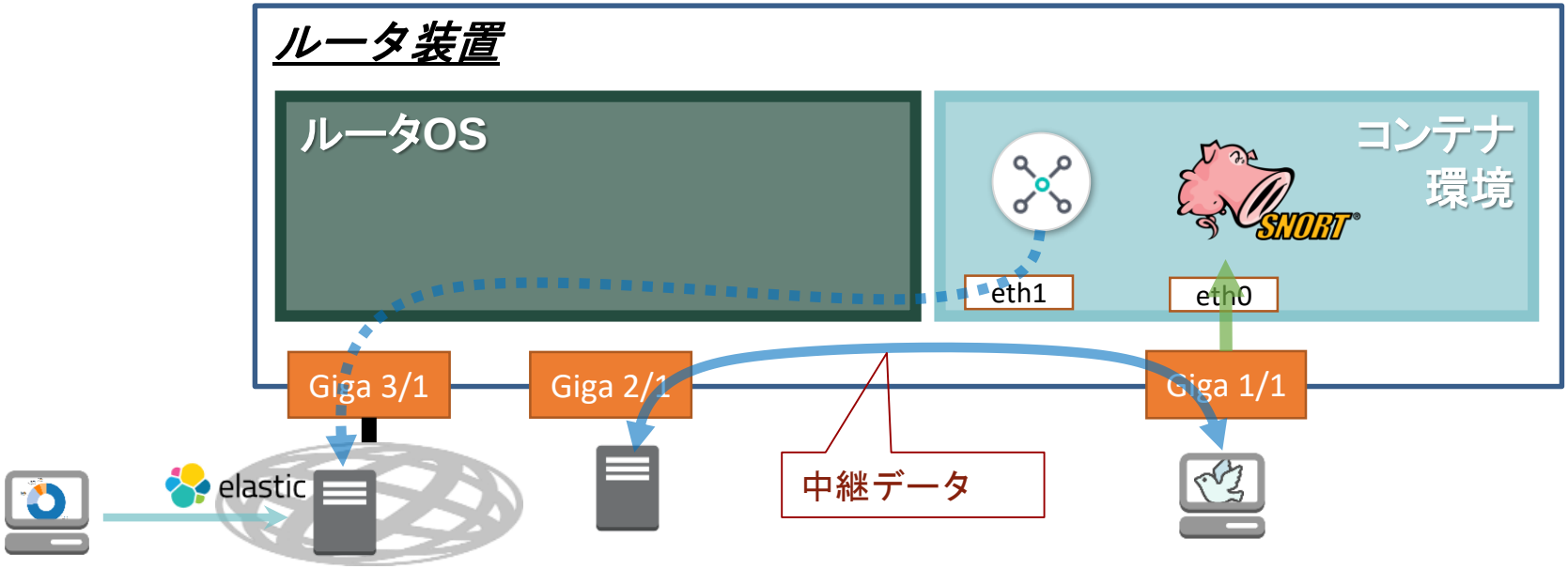
ネットワーク型IDS

- Snortによる中継データの解析、不正検知のアラート

```
root@container:~# snort -c /usr/local/etc/snort/snort.lua -R /usr/local/etc/snort3-
community.rules --plugin-path /usr/local/lib/snort_extra -i eth0 -H -y -k none
```

snort.luaの設定

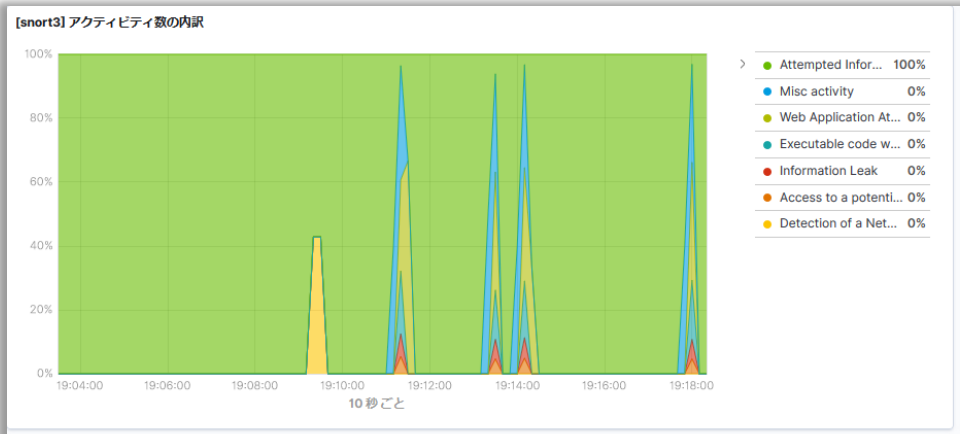
```
alert_json =
{
    file = true,
    fields = 'timestamp pkt_num proto pkt_gen pkt_len dir src_addr src_port dst_addr dst_port
service rule priority class action b64_data'
}
```

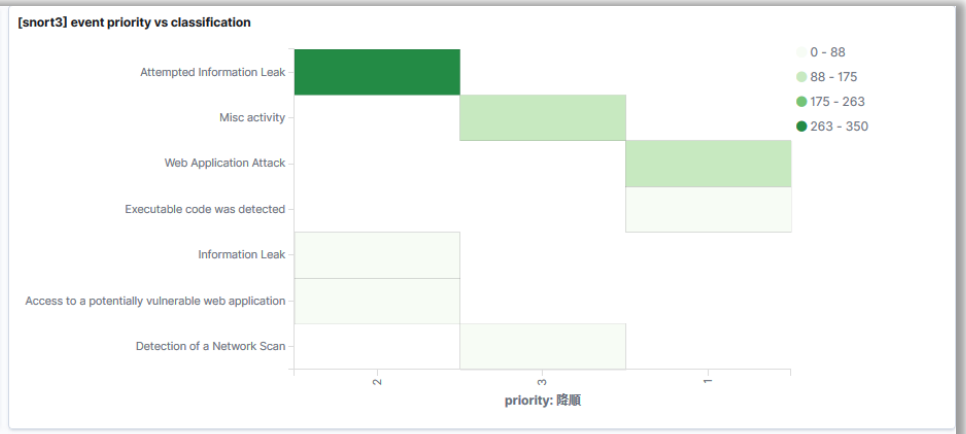


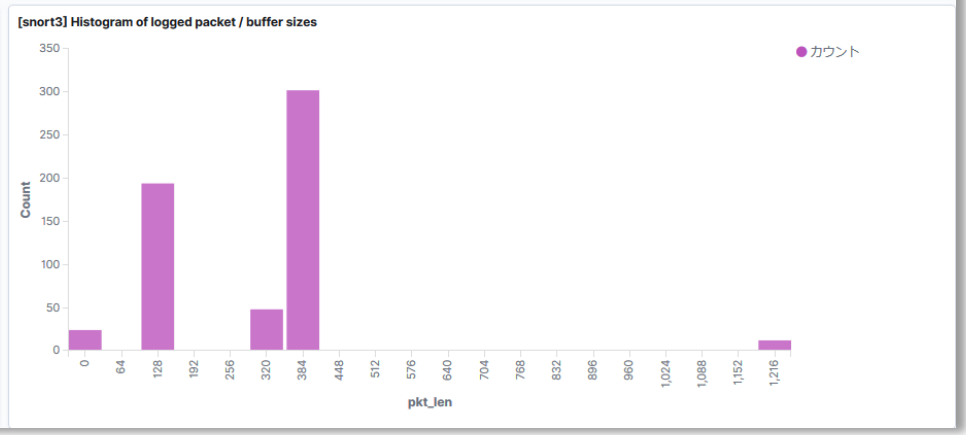
Snortの解析結果の可視化

- Snortのイベントログを Filebeatで外部サーバへ送信

```
root@container:~# tail /run/alert_json.txt
{ “timestamp” : “19/10/21-10:33:58.967749”, “pkt_num” : 16366, “proto” : “UDP”, “pkt_gen” :
“raw”, “pkt_len” : 395, “dir” : “C2S”, “src_addr” : “10.10.30.10”, “src_port” : 56888,
“dst_addr” : “10.10.30.1”, “dst_port” : 161, “service” : “unknown”, “rule” : “1:1411:19”,
“priority” : 2, “class” : “Attempted Information Leak”, “action” : “allow”, “b64_data” :
“MIIBawIBAQQGcHVibGljoIIBXAIESAj51gIBAAIBADCCA(中略)AAFAA==” }
```

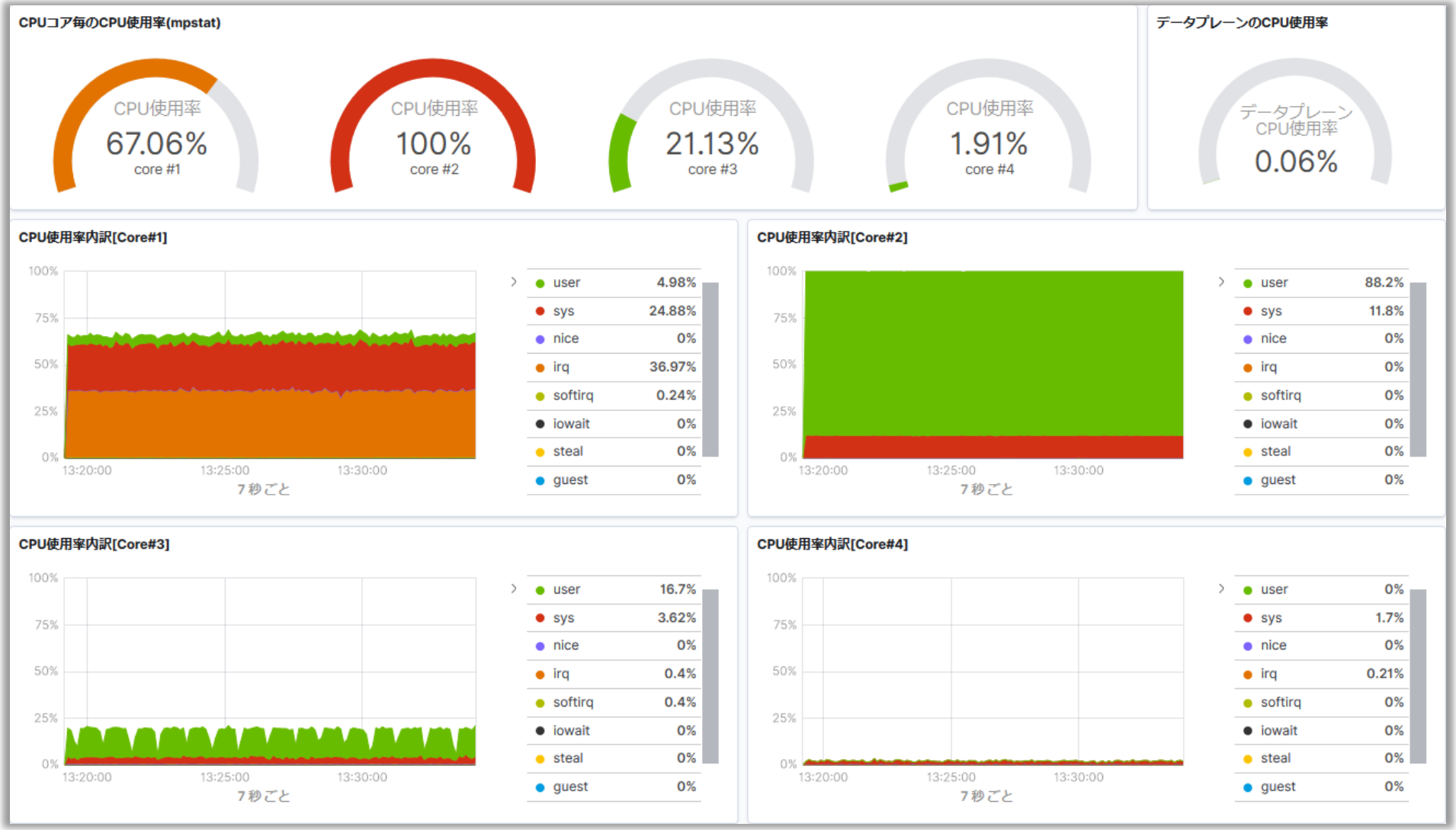






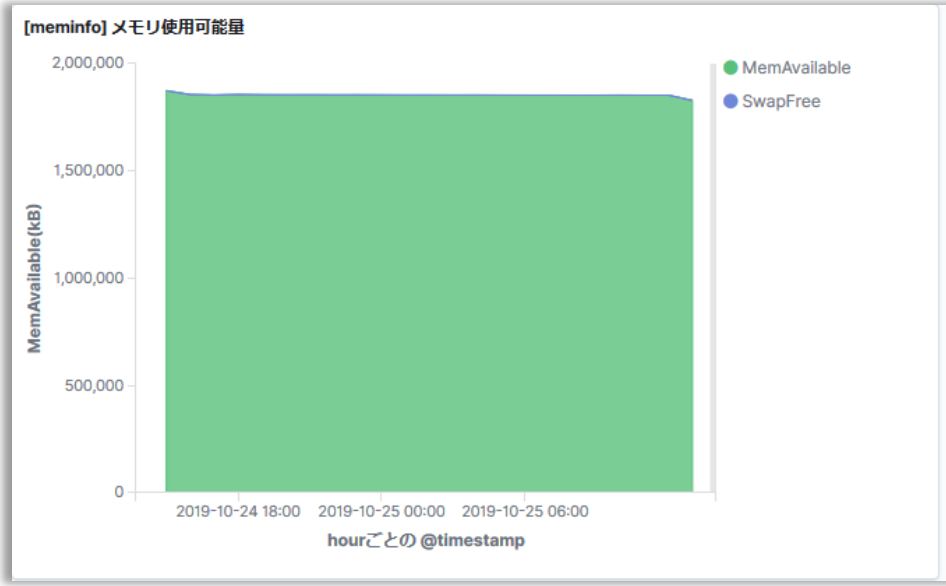
CPU使用率の可視化

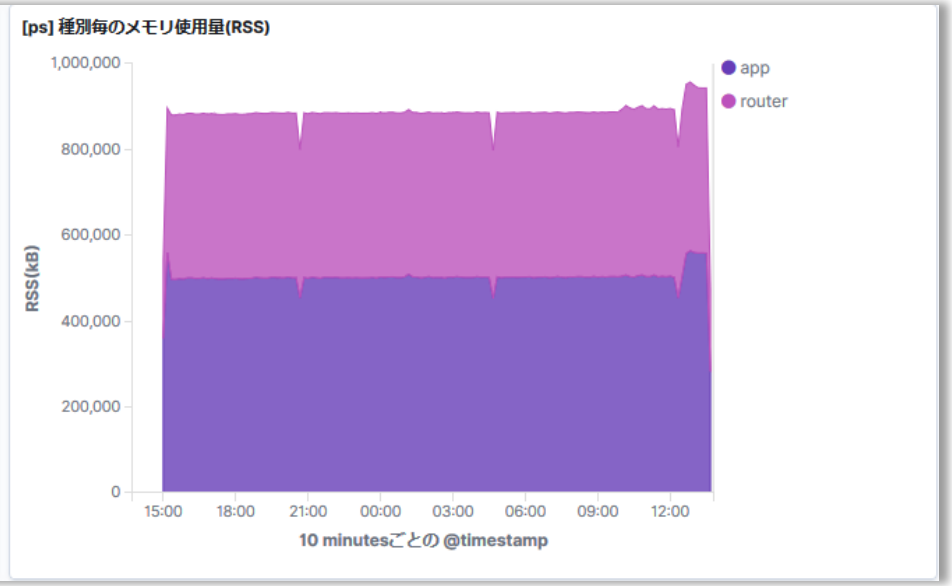
- ルータOS側で取得したデータをもとに可視化
  - データはfilebeatで送信

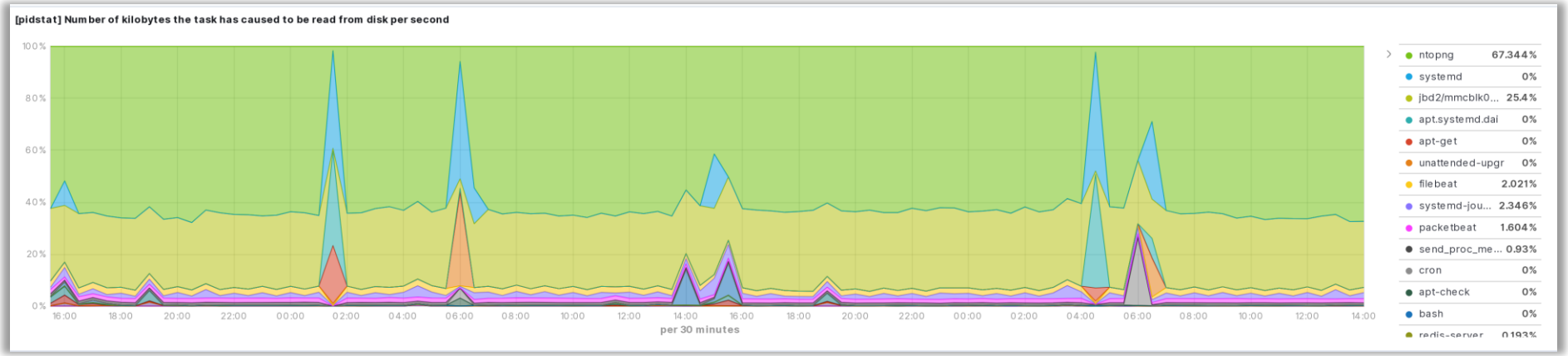


メモリ使用量やディスクI/Oの可視化

- 同様にメモリ使用量やディスクI/Oの可視化も可能







こんなこともできるかも（例）

- Webカメラ
  - USBタイプのカメラを装着してルータから画像を配信
- ソフトウェアAP
  - USBタイプの無線LAN親機を装着してルータを無線LANアクセスポイントにする
- ローカル5G
  - OSSのEPCを動作させてプライベートLTEを構築
- ChatOps
  - チャットサービスを利用したルータ装置の情報共有
- 構成管理自動化
  - Cloud-initやAnsibleなどと連携してルータやサーバ環境の構築を自動化
- 独自開発アプリのポーティング
  - Ubuntuベースのシステムで動作可能なら動くはず

皆様のご意見をお聞かせください！