

FITELnet: F70/F71/F220/F221/F225/F310/ F220 EX/F221 EX

コンテナ型仮想環境の使用方法

古河電気工業株式会社

目次

はじめに	1
本書の構成	1
マークについて	1
注意事項	2
補足情報	2
参考資料	2
1. コンテナ環境のセットアップ	3
1.1. コンテナ環境のリソース	4
1.2. ルータOSの設定(CLI)	5
1.2.1. コンテナ環境生成のための設定	5
1.2.2. インタフェース設定例	6
1.3. コンテナ環境のネットワーク設定	8
1.4. 時刻同期のための設定	9
1.5. コンテナ環境の起動	10
1.6. コンテナ環境への接続	12
1.6.1. シェル起動	14
1.6.2. コンソール接続	15
1.6.3. ネットワーク接続	16
1.7. コンテナ環境の停止	18
1.8. 設定例	19
1.9. 参考資料	22
2. アプリケーションの実行	23
2.1. Alpine Linuxが提供するアプリケーション	24
2.1.1. SSHサーバ	25
2.1.2. パケットキャプチャ	26
2.1.3. SNMPエージェント	30
2.1.4. packetbeat	31
2.1.5. NetFlow(softflowd)	33
2.2. 参考資料	34
付録 A: Elastic Stackを使用したデータの可視化	35
A-1: サーバのセットアップ	36
A-1-1: Elasticsearchの設定	36
A-1-2: Kibanaの設定	37
A-1-3: 時刻同期の設定	38
A-2: ネットワーク解析データの可視化	39
A-2-1: packetbeat	39
A-3: インタフェースの統計情報	44
A-4: システムログ	50
A-5: 参考資料	51
付録 B: USBフラッシュメモリの使用	52

付録 C: ポートモニタリング機能の使用	53
----------------------------	----

はじめに

F70/F71/F220/F221/F225/F310/F220 EX/F221 EX（以下、ルータ装置）では、コンテナ型仮想環境（以下、コンテナ環境）を利用することができます。この環境では、Alpine Linuxベースのシステム全体を一つのコンテナ内で動かすことができます（システムコンテナ）。このため、initも含めた様々なアプリケーションが起動されます。apkコマンドなどを使用することでソフトウェアのインストールやアップグレードも可能です。

本書の構成

本書では、下記内容についての説明を記載しています。

- コンテナ環境のセットアップ
- コンテナ環境でのアプリケーションの実行

マークについて



注意事項を記載しています。



重要事項を記載しています。



補足情報を記載しています。

注意事項



- 当社では、Alpine Linux 3.12, 3.13, 3.20のクラウドイメージを使用したコンテナ環境の基本的な動作確認を行っております。全てのアプリケーションについて動作確認、及び、動作保証しているものではありませんので、予めご了承ください。
- 本書は、[FITELnet LXCアプリケーション^{\[1\]}](https://www.furukawa.co.jp/fitelnet/product/container/lxc/index.html)のページで公開している、Alpine Linux イメージファイル3.20_20240704の利用を前提としています。以下よりも古いイメージファイルの場合は、デフォルトでSSHアクセス可能なアカウント（ユーザ名：operator、パスワード：fitelnet-container）はございませんので、ご注意ください。
- コンテナ環境に対するセキュリティ対策は、物理マシンと同様の対策を施してください。セキュリティ対策を怠ったことによる故障や損害などに関しては、当社は一切の責任を負いかねますので、予めご了承ください。
- コンテナ環境の使用に起因する故障や損害などに関して、当社は一切の責任を負いかねますので、予めご了承ください。

補足情報

Alpine Linuxのアプリケーションやシステムの使い方に関しては、下記のサイトなどを参考にしてください。

- [Welcome to Alpine Linux Wiki^{\[2\]}](https://wiki.alpinelinux.org/wiki/Main_Page)



ルータ装置のマニュアルもあわせて参照してください。

- [コマンドリファレンス-構成定義編^{\[3\]}](https://www.furukawa.co.jp/fitelnet/f/man/70_220/pdf/cmd_refe_config.pdf) 「コンテナ機能の設定」の章にルータOSの設定コマンドの説明がございます。
- [コマンドリファレンス-運用管理編^{\[4\]}](https://www.furukawa.co.jp/fitelnet/f/man/70_220/pdf/cmd_refe_ope.pdf) 「コンテナ機能関連」の章にコンテナ環境の状態確認やバックアップ方法の記載がございます。

参考資料

[1] FITELnet LXCアプリケーション <https://www.furukawa.co.jp/fitelnet/product/container/lxc/index.html>

[2] Welcome to Alpine Linux Wiki https://wiki.alpinelinux.org/wiki/Main_Page

[3] コマンドリファレンス-構成定義編 https://www.furukawa.co.jp/fitelnet/f/man/70_220/pdf/cmd_refe_config.pdf

[4] コマンドリファレンス-運用管理編 https://www.furukawa.co.jp/fitelnet/f/man/70_220/pdf/cmd_refe_ope.pdf

Chapter 1. コンテナ環境のセットアップ

コンテナ環境は、ルータOS（ルータ装置のOS）とは独立した環境となります。コンテナ環境のアプリケーションは、ルータOSのCLIではなく、コンテナ環境にログインして実行いただく必要があります。

コンテナ環境を使用するために必要な設定方法について説明します。主に下記の設定が必要となります。

- ルータOSの設定
 - 設定内容：コンテナ環境生成、インタフェース設定など
 - 設定方法：ルータOSにログインして設定します
- コンテナ環境側の設定
 - 設定内容：ユーザアカウント、ネットワークなどのシステム関連の設定、各アプリケーションの設定など
 - 設定方法：コンテナ環境内部にログインして設定します

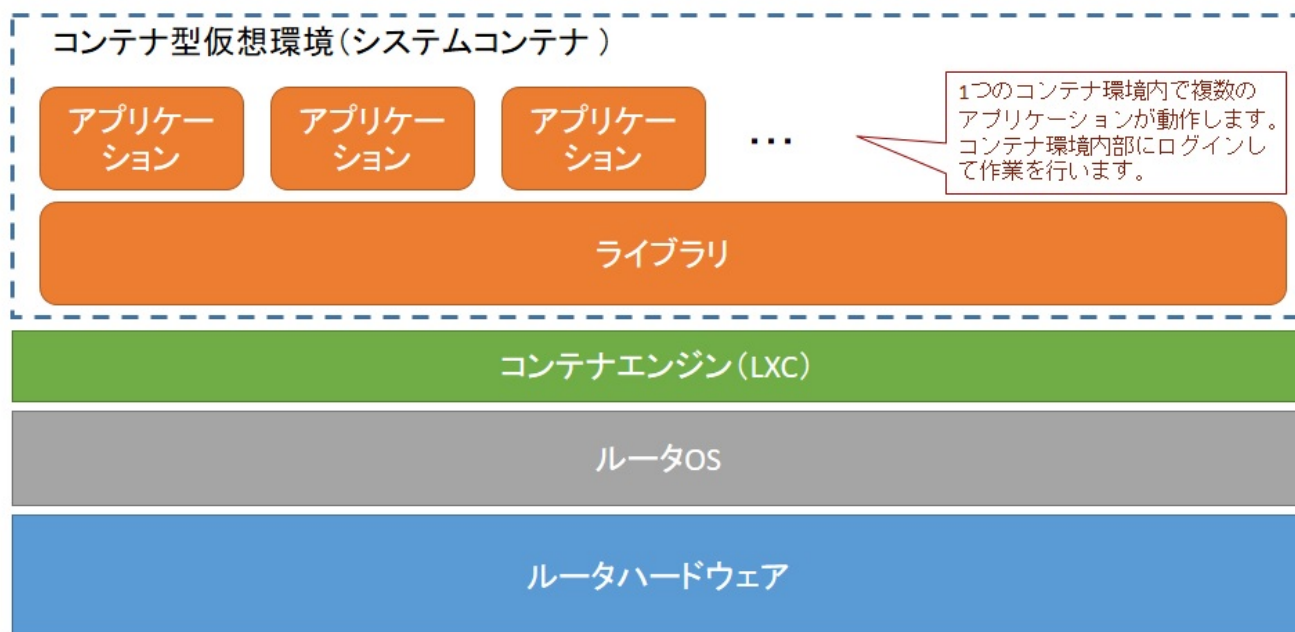


図 1. F70/F71/F220/F221/F225/F310/F220 EX/F221 EXのコンテナ環境

1.1. コンテナ環境のリソース

F70/F71/F220/F221/F225/F310/F220 EX/F221 EXでは、コンテナ環境（システムコンテナ）を一つのみ作成することが可能です。コンテナ環境には、ルータ装置の下記リソースが割り当てられます。

- CPU
 - 2コア（F310はデフォルトでは1コアとなります）
 - 2コア割り当ての場合に、アプリケーションが使用可能なコアは1つのみです。もう1つは主にカーネルが使用します。
 - ルータ装置のプラットフォームは、ARMマルチコア（計4コア）です。
- メモリ
 - F70/F71：最大0.7GB（デフォルトは0.5GB）
 - F220/F221/F225/F310/F220 EX/F221 EX：最大2.0GB（デフォルトは1.5GB）
- ストレージ
 - F70/F71/F220/F221/F225/F310/F220 EX/F221 EX：2.0GB（ユーザ利用領域は1.4GB）
 - F310：4.0GB（ユーザ利用領域は3.2GB）
- ネットワークインタフェース
 - 1Gbps（ルータ装置内部のスイッチのポートがコンテナ専用として1つ割り当てられています。）



- コンテナ環境で保存したファイルはルータ装置内部のストレージに保存されます。お客様が作成したデータの保管およびバックアップ作業は、お客様の責任で行ってください。コンテナ環境に保存したデータの保管に関しては、当社では一切の責任を負いかねますので、予めご了承ください。
- コンテナ環境を利用している最中に電源OFF等でルータ装置が停止した場合は、コンテナ環境のデータが消失する可能性があります。必要に応じて、停電対策やコンテナ環境の定期的なバックアップなどを行ってください。
- コンテナ環境に保存できるデータの容量には限りがあるので、サイズの大きなデータを保存する場合は、必要に応じて、コンテナ環境外部のディスクを使用してください。



- メモリ容量のサイズは、**container limits memory** コマンドで設定可能です。詳細は、[コマンドリファレンス-構成定義編^{\[1\]}](#)を参照してください。
- コンテナ環境は、非特権コンテナとして動作します。

1.2. ルータOSの設定(CLI)

コンテナ環境をネットワークに接続して利用するためには、ルータOSにて下記の設定を行う必要があります。

- コンテナ環境生成
- インタフェース有効化
- 時刻同期

1.2.1. コンテナ環境生成のための設定

グローバルモードで下記の設定を行います。

```
Router(config)#container enable
```



はじめてコンテナを生成する場合は設定反映(refresh)に1～2分を要することがあります。

1.2.2. インタフェース設定例

コンテナ環境で使用するLAN側インタフェース(F70/F71:Giga 1/1～1/4, F220/F221/F225/F220 EX/F221 EX:Giga 1/1～1/8, F310:Giga 1/1～1/5)の設定を行います。コンテナ環境に追加したいインタフェースを **container-use** コマンド(インタフェース設定モード)で設定してください。例えば、Giga 1/3のインタフェースに設定する場合は、次のように入力します。

```
Router(config-if-ge 1/3)#container-use
```

Giga 1/3のインタフェースを次のように設定した場合、bridge-group 30 とL2接続可能なインタフェース (VLAN-ID:50) がコンテナ環境に生成されます。同様の設定を別のGigaインタフェースもしくはサブインタフェースに対して設定すると、コンテナ環境には複数のインタフェースが作成されます。

リスト 1. コンテナ環境で使用するインターフェースの設定例

```
interface GigaEthernet 1/3
  vlan-id 50
  bridge-group 30
  channel-group 30
  container-use
exit
!
interface Port-channel 30
  ip address 10.10.30.1 255.255.255.0
exit
```

コンテナ環境側のネットワーク設定については、[コンテナ環境のネットワーク設定](#)の章にてご説明します。



- コンテナ環境では、"eth + brdige-group番号"が名前となるインタフェースが生成されます（上記の例では、"eth30"が生成されます）。
- 上記例の "ip address 10.10.30.1 255.255.255.0" は、ルータOS側のアドレスとなります。コンテナ環境にIPアドレスを付与するためには、コンテナ環境にログインして設定することが必要です（[コンテナ環境のネットワーク設定](#)の章を参照してください）。
- プライベートIPアドレスを使用する場合、コンテナ環境から外部ネットワークへ接続するためにはNATの設定が必要となります。必要に応じてルータOS側で設定してください（詳細は [コマンドリファレンス-構成定義編^{\[1\]}](#)のNATの章を参照してください）。
- 上記の設定を反映するとルータ側ではVLANのインタフェースとして設定されますが、コンテナ環境内では通常のインタフェースとして見えます。IPアドレス等の設定はコンテナ環境内にアクセスして設定を行う必要があります。

コンテナ環境から外部のネットワークへのアクセスが必要になる場合があります。このため、ルータOS側で外部のネットワークに接続するためのGigaインタフェース(2/1もしくは3/1)の設定を次のように行います（Giga 3/1はF220/F221のみ）。

```
interface GigaEthernet 2/1
  vlan-id 31
  bridge-group 31
  channel-group 31
exit
!
interface Portchannel 31
  ip address xxx.xxx.xxx.xxx yyy.yyy.yyy.yyy
exit
```



コンテナ環境はルータOSのデータプレーン経由で外部のネットワークへ接続します。

1.3. コンテナ環境のネットワーク設定

CLIコマンドにより設定可能です。以下に設定例を示します。詳細は [コマンドリファレンス-構成定義編^{\[2\]}](#) をご参照ください。

```
container configuration
dns x.x.x.x y.y.y.y ← お客様の環境に合わせて設定してください
hostname F221-Container
!
interface 1
bridge-group 1 ← ルータOS側のLANインタフェースのブリッジグループ番号に合わせてください
ip address 192.168.127.21 255.255.255.0
exit
!
interface 2
bridge-group 30 ← ルータOS側のLANインタフェースのブリッジグループ番号に合わせてください
ip address 10.10.30.10 255.255.255.0
ip gateway 10.10.30.1
exit
!
exit
```



[インタフェース設定例](#)の章で述べた通り、コンテナ環境側のインタフェースの実体はルータOS側のVLANインターフェースとなります。コンテナ環境側でVLANインタフェースの設定を行っても動作しません。

設定が完了したら、**refresh** コマンドを実行して設定を反映します。



ルータOS側のLANインタフェース設定にてブリッジグループ番号やネットワークアドレスを変更した場合は、併せてコンテナ側のインタフェース設定を変更する必要があります。

1.4. 時刻同期のための設定

コンテナ環境を外部のネットワークに接続したオンラインの環境で使用する場合は、時刻同期の設定を行ってください。時刻が同期されていないと、パッケージの更新等でエラーとなる可能性があります。

グローバルモードで下記の設定を行います。NTPサーバのアドレスはご利用の環境にあわせて設定してください。

リスト 2. 時刻同期の設定例

```
Router(config)#ntp server 192.168.1.10
```



付録： [Elastic Stackを使用したデータの可視化](#)を行う場合は、時刻同期の設定が必須となります。

時間のリソースに関してはコンテナ環境内で仮想化することができません。ルータOS側の時刻リソースと共通となるため、コンテナ環境側で時刻同期の設定を行う必要はありません。



コンテナ環境側では、時刻同期を行うためのデーモンプロセス(**systemd-timesyncd**)がインストールされていますが、下記のようにサービスを起動することはできません。必ずルータOS側で時刻同期を行うように設定してください。

```
root@container:~# systemctl status systemd-timesyncd
systemd-timesyncd.service - Network Time Synchronization
   Loaded: loaded (/lib/systemd/system/systemd-timesyncd.service; enabled; vendor preset:
   enabled)
   Active: inactive (dead)
 Condition: start condition failed at Mon 2019-12-02 06:16:35 UTC; 1min 6s ago
           mq ConditionVirtualization=!container was not met
   Docs: man:systemd-timesyncd.service(8)
```

1.5. コンテナ環境の起動

コンテナ環境を起動するには、ルータOSのCLIから **container start** コマンドを実行してください。

```
Router# container start
start ok?[y/N]:yes
```

```
Router#
```



起動コンフィグ(`boot.cfg`)に **container enable** コマンドを設定した状態でルータ装置を起動した場合は、自動的にコンテナ環境の起動が行われます。

コンテナ環境起動後、コンテナに設定を行っていない状態では、`show container list` および `show container information` にて次のような表示が確認されます。参考：[コマンドリファレンス-運用管理編](#)^[3]

```
Router#show container list
```

```
-----+
|  NAME    | BASE IMAGE |  IMAGE VERSION  |  STATE | IPV4 | IPV6 |
-----+
| container | 5c4e1566dd | 3.12 20201012_15:00 | RUNNING |      |      |
-----+
```

```
Router#show container information
```

```
Name: container
Remote: unix://
Architecture: aarch64
Created: 2019/10/08 02:09 UTC
Status: Running
Type: persistent
Profiles: default
Pid: 3602
Ips:
  lo:  inet    127.0.0.1
  lo:  inet6   ::1
Resources:
  Processes: 59
  CPU usage:
    CPU usage (in seconds): 18
  Memory usage:
    Memory (current): 30.74MB
    Memory (peak): 33.24MB
  Network usage:
    lo:
      Bytes received: 237.99kB
      Bytes sent: 237.99kB
      Packets received: 2858
      Packets sent: 2858
    sit0:
      Bytes received: 0B
      Bytes sent: 0B
```

```
Packets received: 0
Packets sent: 0
eth30:
  Bytes received: 0B
  Bytes sent: 0B
  Packets received: 0
  Packets sent: 0
```

```
Router#
```

1.6. コンテナ環境への接続

コンテナ環境への接続方法は下記の3つがあります。

- シェル起動
 - ルータOSのCLIからコンテナ環境内のシェルを起動してアクセスします。パスワード入力なしでrootユーザとしてログイン可能です。
- コンソール接続
 - ルータOSのCLIからコンテナ環境内のコンソールを起動してアクセスします。ログインするためにはパスワードが必要です。
- ネットワーク接続
 - ルータ装置外部からコンテナ環境へSSH接続してアクセスします。デフォルトでは、ユーザ名：operator、パスワード：fitelnet-containerにて、ログイン可能です。



- 初回アクセス後、ユーザ名/パスワードは、デフォルトから変更してご利用いただくことを推奨します。
- シェル起動によるコンテナ環境への接続は、ルータOS側の特権ユーザであれば誰でもアクセス可能です。特権ユーザの管理は、お客様の責任で適切に行ってください。
- コンテナ環境には任意のユーザアカウントを作成可能です。ユーザの管理は、お客様の責任で適切に行ってください。

コンテナ環境にログインするためのアカウントを、デフォルトから変更する方法を説明します。下記のように、LAN側の端末等コンテナへ到達可能な環境からSSHログインして、新しいアカウントを `useradd/passwd` コマンドにより作成してください。

```
$ ssh operator@10.10.30.10
operator@10.10.30.10's password:  [デフォルトのパスワード "fitelnet-container" を入力]
Welcome to Alpine!

The Alpine Wiki contains a large amount of how-to guides and general
information about administrating Alpine systems.
See .

You can setup the system with the command: setup-alpine

You may change this message by editing /etc/motd.

container:~$ sudo su
[sudo] password for operator:  [デフォルトのパスワード "fitelnet-container" を入力]
/home/operator # useradd -m furukawa
/home/operator # passwd furukawa
New password:
Retype new password:
```

```
passwd: password updated successfully
/home/operator # su - furukawa
container:~$ id
uid=1000(furukawa) gid=1000(furukawa) groups=1000(furukawa)
container:~$ exit
/home/operator #
```

もしくは下記のように、ルータOSにてシェル起動によりコンテナ環境にログインしてから、新しいアカウントを `useradd/passwd` コマンドにより作成してください。

```
F220#container attach
~ # useradd -m furukawa
~ # passwd furukawa
New password:
Retype new password:
passwd: password updated successfully
~ # su - furukawa
container:~$ id
uid=1000(furukawa) gid=1000(furukawa) groups=1000(furukawa)
container:~$ exit
~ #
```

新しく作成したアカウントにroot権限を付与する場合は、以下(1),(2)を実行してください。

- (1) 「`sed '/^root/a <ユーザ名> ALL=(ALL:ALL) ALL' /etc/sudoers | EDITOR=tee visudo > /dev/null`」を実行して、<ユーザ名>にsudo権限を付与
- (2) 「`cat /etc/sudoers | grep -A 1 ^root`」を実行して、<ユーザ名>にsudo権限付与されたことを確認（以下のように表示されればOKです）

```
~ # sed '/^root/a furukawa ALL=(ALL:ALL) ALL' /etc/sudoers | EDITOR=tee visudo > /dev/null
~ #
~ # cat /etc/sudoers | grep -A 1 ^root
root ALL=(ALL:ALL) ALL
furukawa ALL=(ALL:ALL) ALL
~ #
```

上記実施後、デフォルトのアカウントを削除してください。

```
~ # userdel operator
~ # exit
F220#
```



container attach でコンテナ環境にアクセスした場合は、rootユーザとしてログインします。rootユーザのパスワードも適切に設定してください。

```
~ # id
uid=0(root) gid=0(root)
~ # passwd
New password:
Retype new password:
passwd: password updated successfully
~ #
```

以下にアクセス例を示します。

1.6.1. シェル起動

ルータOSのCLI上で **container attach** を実行します。

```
Router#container attach
~ # uname
Linux
~ # exit
exit

Router#
```



exitを入力するとルータOSのCLIに戻ります。

1.6.2. コンソール接続

ルータOSのCLI上で **container attach console** を実行します。コンソール接続を行うと、起動時やシャットダウン時のログ情報が端末に出力されます。

```
Router#container attach console
To detach from the console, press: <ctrl>+a q

Welcome to Alpine Linux 3.12
Kernel 4.14.47 on an aarch64 (/dev/console)

container login: root
Password:
Welcome to Alpine!

The Alpine Wiki contains a large amount of how-to guides and general
information about administrating Alpine systems.
See http://wiki.alpinelinux.org/.

You can setup the system with the command: setup-alpine

You may change this message by editing /etc/motd.

container:~# uname -snm
Linux container aarch64
container:~# exit

Welcome to Alpine Linux 3.12
Kernel 4.14.47 on an aarch64 (/dev/console)

container login:
Router#
```



ルータOSのCLIに戻るには、<ctrl>+a q を入力してください。

1.6.3. ネットワーク接続

ルータ装置外部の端末からSSHアクセスを行います。

```
root@remote:~# ssh 10.10.30.10
root@10.10.30.10's password:
Welcome to Alpine!

The Alpine Wiki contains a large amount of how-to guides and general
information about administrating Alpine systems.
See http://wiki.alpinelinux.org/.

You can setup the system with the command: setup-alpine

You may change this message by editing /etc/motd.

container:~#
container:~# cat /etc/os-release
NAME="Alpine Linux"
ID=alpine
VERSION_ID=3.11.0
PRETTY_NAME="Alpine Linux v3.11"
HOME_URL="https://alpinelinux.org/"
BUG_REPORT_URL="https://bugs.alpinelinux.org/"
container:~# exit
Connection to 10.10.30.10 closed.
root@remote:~#
```



- コンテナ環境にて、SSH接続先となるIPアドレスの設定が必要です。[コンテナ環境のネットワーク設定](#)の章を参照してください。
- デフォルトではSSHサービスのパスワード認証は無効となっています。設定変更する場合は、[SSHサーバ](#)の章を参照してください。公開鍵認証を使用する場合は、パスワードの入力は必要ありません。

公開鍵認証でログインする場合は、予めクライアント側で認証鍵を生成してコンテナ環境側に公開鍵の情報を登録しておく必要があります。認証鍵は下記のコマンドで作成可能です。作成した公開鍵は、該当するユーザのホームディレクトリ配下にあるauthorized_keys（~/ssh/authorized_keys）に登録してください。詳細はオンラインマニュアル等を参照してください。

リスト 3. SSH認証鍵の作成例

```
root@remote:~# ssh-keygen
```

ssh-copy-id コマンドが使用できる環境では、下記のようにリモートの端末側から公開鍵の登録が可能です。

リスト 4. SSH公開鍵の登録例

```
root@remote:~# ssh-copy-id root@192.168.127.21
```



SSHサーバのパスワード認証を有効にしておく必要があります。登録が完了したら、パスワード認証を無効にしておくことをお勧めします。

1.7. コンテナ環境の停止

コンテナ環境を停止する場合は、コンテナ環境内で **poweroff** コマンド等によりシャットダウンを行ってください。

```
container:~# poweroff
```

もしくは、ルータOSのCLI側から **container stop** コマンドを実行してください。

```
Router#container stop
stop ok?[y/N]:yes

Router#
```



コンテナ環境を再度起動する場合は、ルータOSのCLI側から **container start** を実行してください。コンテナ環境内でrebootコマンドを実行した場合は、コンテナ環境が再起動するので **container start** を実行する必要はありません。



ルータOS側で **container enable** コマンドを削除してコンテナ機能を無効化した場合は、自動的にコンテナ環境もシャットダウンされます。



コンテナ環境を利用している最中に電源OFF等でルータ装置を停止した場合は、コンテナ環境のデータが消失する可能性があります。ルータ装置をシャットダウンする際は、ルータOS側で **container enable** コマンドを削除してコンテナ機能を無効化してください。

コンテナ環境停止後、show container list および show container information にて次のような表示が確認されます。参考：[コマンドリファレンス-運用管理編](#)^[3]

```
Router#show container list
```

```
-----+
|  NAME   | BASE IMAGE |  IMAGE VERSION  |  STATE  | IPV4 | IPV6 |
-----+
| container | 5c4e1566dd | 3.12 20201012_15:00 | STOPPED |      |      |
-----+
```

```
Router#show container information
```

```
Name: container
Remote: unix://
Architecture: aarch64
Created: 2019/10/08 02:09 UTC
Status: Stopped
Type: persistent
Profiles: default
Router#
```

1.8. 設定例

コンテナ環境を利用するための、ネットワーク構成の設定例をご説明します。

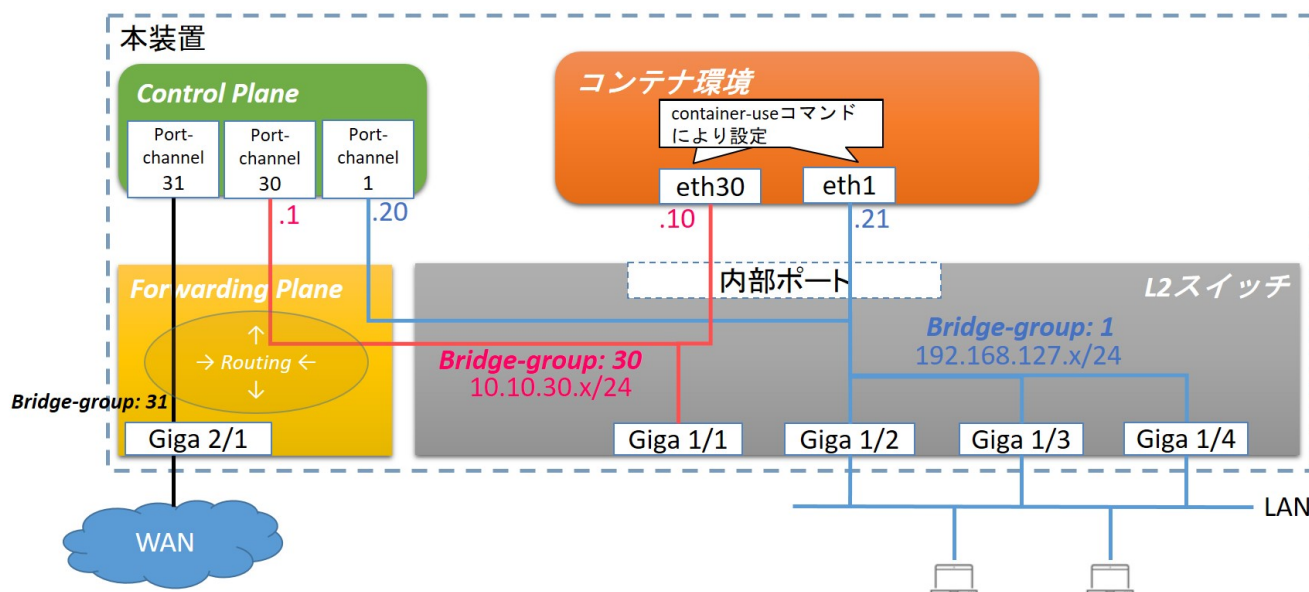


図 2. コンテナ環境のネットワーク構成例(2つのインタフェースを追加した場合)

参考：機能説明書^[4] 2.19.3 ブリッジグループの装置内部構成

CLI設定は次の通りです。

リスト 5. コンテナ環境のネットワーク構成例(2つのインタフェースを追加した場合)のCLI設定

```
container configuration
dns x.x.x.x y.y.y.y ← お客様の環境に合わせて設定してください
hostname F221-Container
!
interface 1
bridge-group 1
ip address 192.168.127.21 255.255.255.0
exit
!
interface 2
bridge-group 30
ip address 10.10.30.10 255.255.255.0
ip gateway 10.10.30.1
exit
!
exit
!
ntp server xxx.xxx.xxx.xxx
!
interface GigaEthernet 1/1
vlan-id 50
bridge-group 30
channel-group 30
```

```

container-use
exit
!
interface GigaEthernet 1/2
vlan-id 1
bridge-group 1
channel-group 1
container-use
exit
!
interface GigaEthernet 1/3
vlan-id 1
bridge-group 1
channel-group 1
container-use
exit
!
interface GigaEthernet 1/4
vlan-id 1
bridge-group 1
channel-group 1
container-use
exit
!
interface GigaEthernet 2/1
vlan-id 31
bridge-group 31
channel-group 31
exit
!
interface Port-channel 1
ip address 192.168.127.20 255.255.255.0
exit
!
interface Port-channel 30
ip address 10.10.30.1 255.255.255.0
exit
!
interface Port-channel 31
ip address xxx.xxx.xxx.xxx yyy.yyy.yyy.yyy
exit
!

```

show container list および show container information は次のように表示されます。

```
Router#show container list
```

NAME	BASE IMAGE	IMAGE VERSION	STATE	IPV4	IPV6
container	5c4e1566dd	3.12 20201012_15:00	RUNNING	192.168.127.21 (eth1)	
				10.10.30.10 (eth30)	

```
Router#show container info
```

```
Name: container
Remote: unix://
Architecture: aarch64
Created: 2019/10/08 07:46 UTC
Status: Running
Type: persistent
Profiles: default
Pid: 23852
Ips:
  lo:   inet    127.0.0.1
  lo:   inet6   ::1
  eth1: inet    192.168.127.21   sit0
  eth1: inet6   fe80::280:bdff:fef0:5a76   sit0
  eth30:        inet    10.10.30.10   sit0
  eth30:        inet6   fe80::280:bdff:fef0:5a76   sit0
Resources:
  Processes: 24
  CPU usage:
    CPU usage (in seconds): 15
  Memory usage:
    Memory (current): 4.98MB
    Memory (peak): 8.13MB
  Network usage:
    eth1:
      Bytes received: 3.89kB
      Bytes sent: 908B
      Packets received: 68
      Packets sent: 12
    eth30:
      Bytes received: 170.38kB
      Bytes sent: 9.82kB
      Packets received: 100
      Packets sent: 120
    lo:
      Bytes received: 1.06kB
      Bytes sent: 1.06kB
      Packets received: 12
      Packets sent: 12
    sit0:
      Bytes received: 0B
      Bytes sent: 0B
      Packets received: 0
      Packets sent: 0
```

1.9. 参考資料

- [1] コマンドリファレンス-構成定義編 https://www.furukawa.co.jp/fitelnet/f/man/70_220/pdf/cmd_refe_config.pdf
- [2] コマンドリファレンス-構成定義編 https://www.furukawa.co.jp/fitelnet/f/man/70_220/pdf/cmd_refe_config.pdf
- [3] コマンドリファレンス-運用管理編 https://www.furukawa.co.jp/fitelnet/f/man/70_220/pdf/cmd_refe_ope.pdf
- [4] 機能説明書 https://www.furukawa.co.jp/fitelnet/f/man/70_220/pdf/kinou.pdf

Chapter 2. アプリケーションの実行

コンテナ環境では、Alpine Linuxベースのシステムが起動します。Alpine Linuxが提供するソフトウェアの他にサードパーティ製のソフトウェアなども自由にインストールして動作させることが可能です。

本章では、下記のソフトウェアについて、コンテナ環境内での実行例を示します。

- SSHサーバ
- パケットキャプチャ(tcpdump)
- SNMPエージェント
- ネットワークパケットアナライザー(packetbeat)
- ログ送信ツール(filebeat)



各ソフトウェアの使用条件やライセンスを適切に守って使用してください。

2.1. Alpine Linuxが提供するアプリケーション

コンテナ環境では、Alpine Linuxのイメージに含まれているソフトウェアを使用することができます。また、[apk^{\[1\]}](#)コマンドを使用してAlpine Linuxが提供するソフトウェアのインストールやアップグレードも可能です（但し、この場合はコンテナ環境が外部のネットワークに接続している必要があります）。

aptコマンドを使用してソフトウェアをインストールする場合は、予めパッケージリストを更新しておく必要があります。更新する場合は、下記のようにコマンドを実行してください。

```
root@container:~# apk update
```

インストール済みのパッケージは、下記のコマンドで確認できます。

```
root@container:~# apk list --installed
```



インストール済みパッケージのアップデートもapkコマンドで実行できます。適宜アップデートを行ってください。

IMPORTANT:インストールサイズの大きなパッケージや多数のパッケージ追加などを行うと、コンテナ環境のディスク容量が不足する原因となります。不要なパッケージなどは削除するなどして適切なパッケージ数で運用することをお勧めします。

Alpine Linuxのアプリケーションやシステムの使い方に関しては、下記のドキュメントなどを参考にしてください。

- [Alpine Linux package management^{\[2\]}](#)

2.1.1. SSHサーバ

OpenSSH Server^[3]が予めインストールされています。以下に実行例を示します。

リスト 6. SSHサーバの起動例

```
container:~# rc-service sshd start
* Caching service dependencies ... [ ok ]
ssh-keygen: generating new host keys: RSA DSA ECDSA ED25519
* Starting sshd ... [ ok ]
container:~# rc-service sshd status
* status: started
container:~#
```

コンテナ環境起動時に自動的にサービスを起動させたい場合は、下記のように設定を行ってください。

リスト 7. SSHサーバの自動起動例

```
container:~# rc-update add sshd
* service sshd added to runlevel default
container:~# rc-status
Runlevel: default
networking [ started ]
monit [ started 00:48:50 (0) ]
chronyd [ started ]
crond [ started ]
sshd [ started ]
Dynamic Runlevel: hotplugged
Dynamic Runlevel: needed/wanted
Dynamic Runlevel: manual
container:~#
```



- rootユーザのパスワード認証を有効にするためには、さらに PermitRootLogin の設定項目を有効にする必要があります。詳細は、オンラインドキュメント等を参照してください。
- rootユーザのパスワード認証を許可するとセキュリティ的に脆弱となりますので、十分ご注意ください。
- コンテナ環境に予めインストールされているソフトウェアのうち、バックグラウンドで動作して機能を提供するプログラム（デーモン）は、rc-serviceコマンドで管理できます。詳細はオンラインドキュメント等を参照してください。

クライアント側からの接続は、[ネットワーク接続](#)の章を参照してください。

2.1.2. パケットキャプチャ

次のように、tcpdumpによりパケットキャプチャを行うことができます。

リスト 8. パケットキャプチャの実行例

```
container:~# tcpdump -i eth1 -vvv
tcpdump: listening on eth1, link-type EN10MB (Ethernet), capture size 262144 bytes
05:07:55.996932 STP 802.1d, Config, Flags [none], bridge-id 807f.b8:be:bf:06:dc:00.8004, length
43
    message-age 0.00s, max-age 20.00s, hello-time 2.00s, forwarding-delay 15.00s
    root-id 807f.b8:be:bf:06:dc:00, root-pathcost 0
05:07:56.780103 IP6 (flowlabel 0x489cc, hlim 255, next-header ICMPv6 (58) payload length: 16)
container > ip6-allrouters: [icmp6 sum ok] ICMP6, router solicitation, length 16
    source link-address option (1), length 8 (1): 00:80:bd:f0:5a:76
    0x0000: 0080 bdf0 5a76
05:07:57.998935 STP 802.1d, Config, Flags [none], bridge-id 807f.b8:be:bf:06:dc:00.8004, length
43
    message-age 0.00s, max-age 20.00s, hello-time 2.00s, forwarding-delay 15.00s
    root-id 807f.b8:be:bf:06:dc:00, root-pathcost 0
^C
3 packets captured
3 packets received by filter
0 packets dropped by kernel
```

キャプチャ結果をファイルに保存することも可能です。

```
container:~# tcpdump -i eth1 -w packet.cap
tcpdump: listening on eth1, link-type EN10MB (Ethernet), capture size 262144 bytes
^C1 packet captured
1 packet received by filter
0 packets dropped by kernel
root@container:~# ls -l packet.cap
-rw-r--r-- 1 root root 100 Oct  8 05:08 packet.cap
```



- ファイルに保存する場合は、ディスク容量を消費します。キャプチャするデータ量が多い場合はご注意ください。
- データサイズの大きなファイルを保存する場合や、ディスクI/Oの負荷が高い書き込み処理を行う場合などは、USBフラッシュメモリなどの外部記憶ディスクへ保存することをお勧めします。

ルータOSのポートモニタリング機能を使用すると、コンテナ環境側でルータ装置の中継データをキャプチャすることが可能です。この場合、コンテナ環境側では次のように **eth0** のインターフェースに対してキャプチャを行います。

リスト 9. ルータOSの中継データのキャプチャ実行例

```
container:~# tcpdump -i eth0 -vvv
```



ポートモニタリング機能の詳細と注意点は、付録：ポートモニタリング機能の使用を参照してください。



高負荷な中継データをキャプチャする場合は、ルータ装置の中継性能に影響を与える可能性があるので、ご注意ください。

コンテナ環境が外部のネットワークに接続している場合は、ルータ装置外部の端末上でキャプチャ結果を表示することも可能です。Wiresharkのリモートキャプチャ機能を使用した例を以下に示します。

まず、リモートの端末上でWiresharkを起動してSSH remote capture のインタフェースオプションを選択してください。

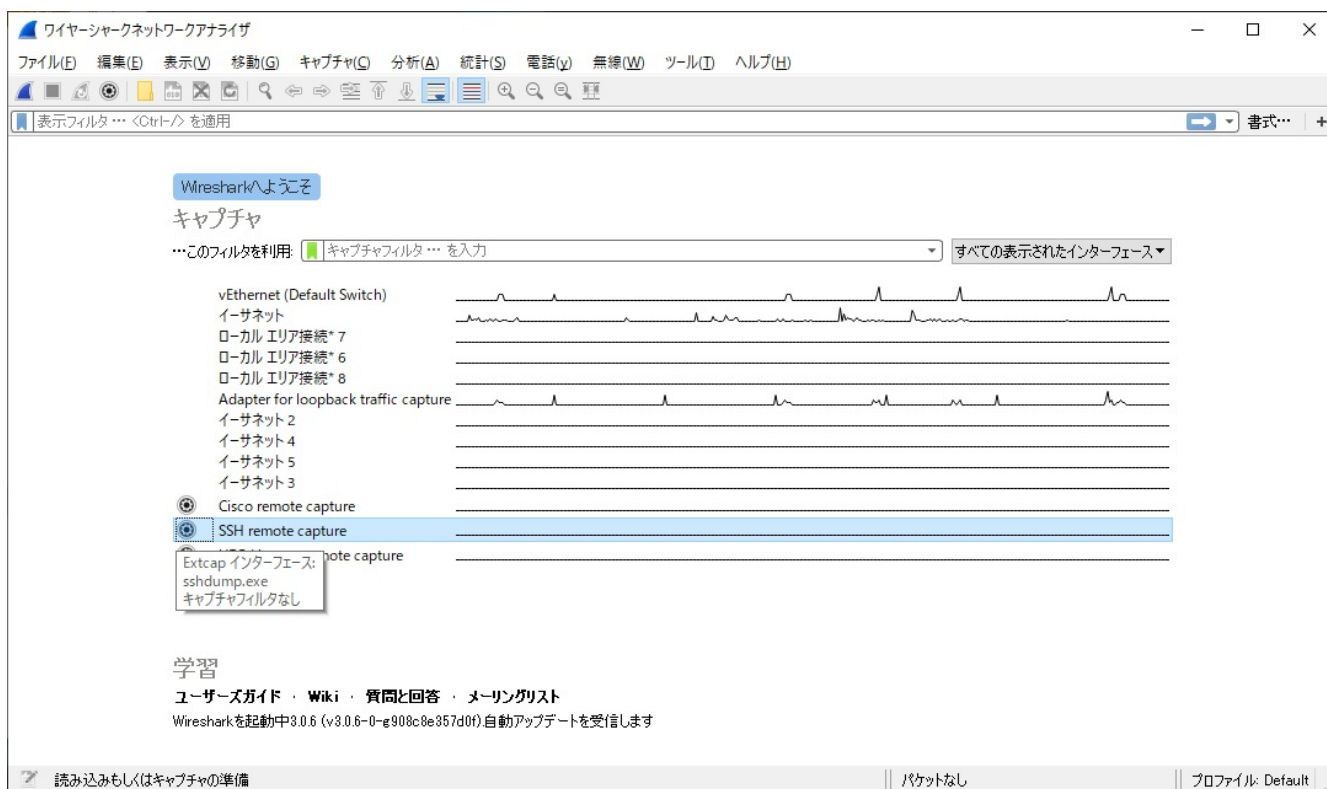


図 3. Wiresharkのsshdumpオプション



該当するオプションが表示されていない場合は、**sshdump** のツールがインストールされているかどうかを確認してください。

下記のように実行すると、GUI上でキャプチャ結果をリアルタイムに表示することができます。

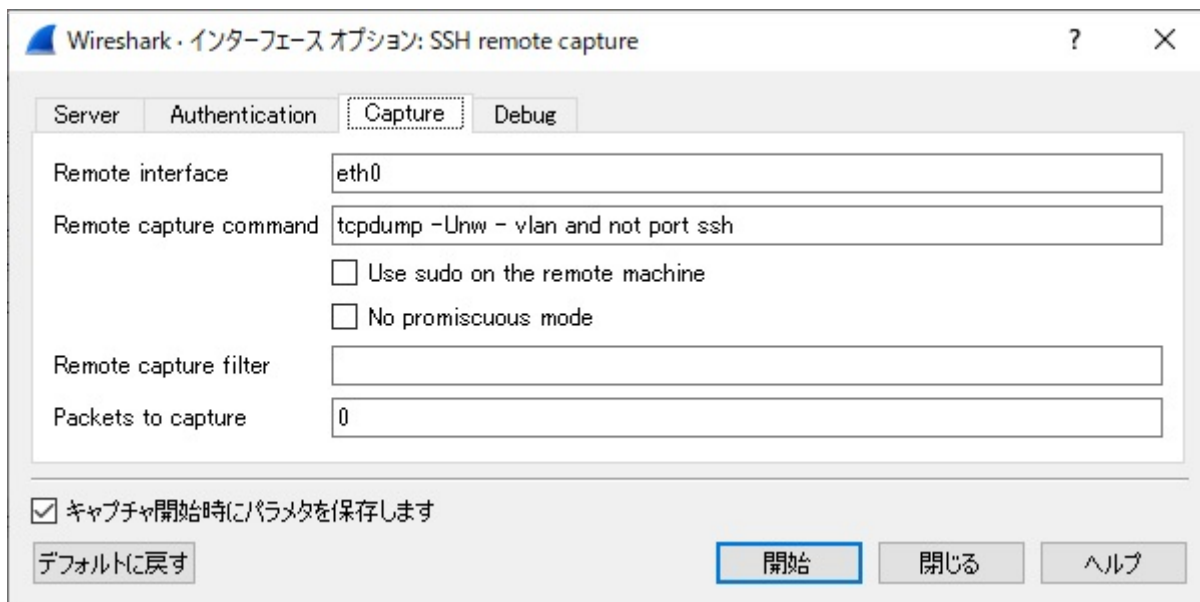


図 4. Wiresharkのリモートキャプチャ実行例(GUIの場合)

各欄には、下記の値を入力してください。

- **Remote interface** : コンテナ環境上のインタフェース名を入力してください（ルータ装置の中継データをキャプチャする場合は eth0 を入力してください）
- **Remote capture command** : tcpdump -Unw - vlan and not port ssh
- **Remote capture filter** : 必要に応じて入力してください

コンテナ環境のIPアドレスは、下記の画面で入力できます。ご利用の環境にあわせて入力してください。

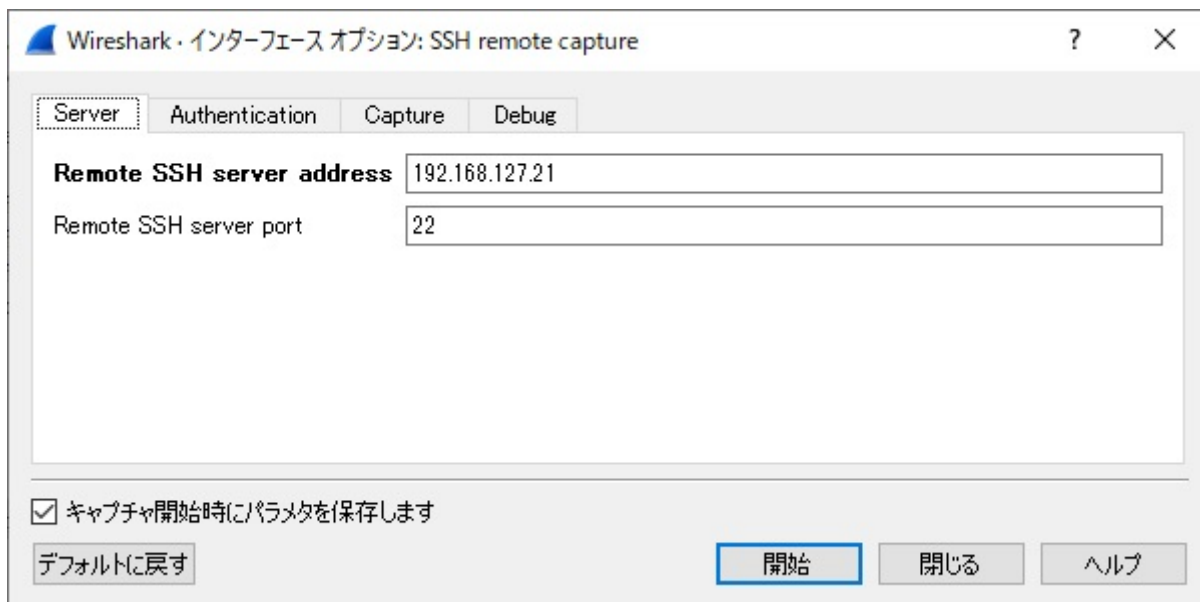


図 5. Wiresharkのリモートサーバの設定例

rootユーザでログインする必要があります。下記のように指定してください。

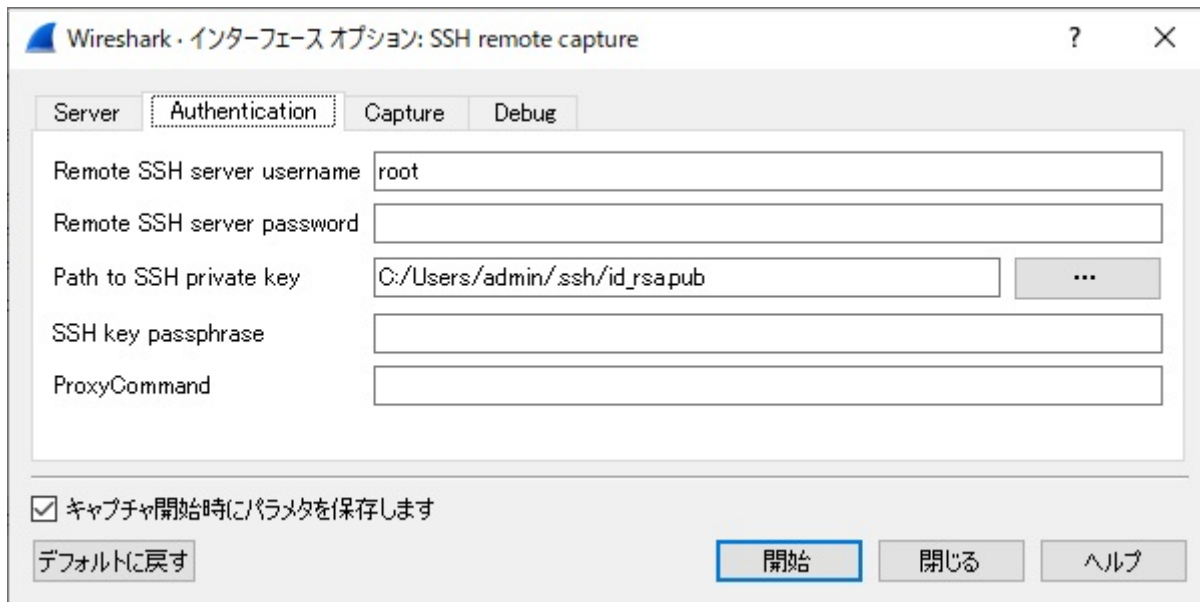


図 6. Wiresharkのリモートサーバの認証設定例



上記は公開鍵認証でログインする例です。SSHによるrootユーザのログインについては、[SSHサーバ](#)の章を参照してください。

コマンドラインからも同様に実行することが可能です。下記のように実行してください。

リスト 10. Wiresharkのリモートキャプチャ実行例(コマンドの場合)

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



パスワード認証でログインする場合は、パスワードの入力タイミングによってはエラーとなる可能性があります。この場合は、公開鍵認証によりログインして実行してください。

2.1.3. SNMPエージェント

SNMPエージェントのアプリケーションを実行すると、コンテナ環境のSNMP MIBの情報を取得することができます。適切に設定を行った上で、下記のように実行してください。



コンテナ環境が外部ネットワークに接続されている必要があります。

自動的にサービスを起動させるようにしたい場合は、下記のように設定してください。

リスト 11. SNMPエージェントの起動例

```
container:~# rc-update add snmpd
* service snmpd added to runlevel default
```

コンテナ環境、もしくは、ルータ装置外部の端末から下記のようにMIBを取得できます。

```
# snmpwalk -v2c -c public -OSX -IR 192.168.127.21 host
```



MIB定義ファイルを保存しているディレクトリが不明な場合、Symbol OID（上記の例では、"host"）の指定ができない場合があります。該当するディレクトリを明示的に指定するか、Numeric OIDを指定してください。詳細はNet-SNMPのオンラインドキュメント等を参照してください。

ルータOSでSNMP機能を有効にしていれば、コンテナ環境からルータOSのMIB情報を取得することも可能です。

```
root@container:~# snmpbulkwalk -v2c -c public -OSX -IR 192.168.127.20 system
```



ルータOSのIPアドレスを指定する必要があります。

2.1.4. packetbeat

[packetbeat](#)^[4]は、Elastic社により開発されているネットワークパケットアナライザーです。Elastic社の製品である、[Elasticsearch](#)^[5]や[Logstash](#)^[6]にデータを送信することで、解析結果をリアルタイムに可視化することが可能です。



データの可視化については、付録：[Elastic Stackを使用したデータの可視化](#)を参照してください。

まず、下記の設定ファイルを編集します。

- `/etc/packetbeat/packetbeat.yml`

設定ファイル中の、下記(1),(2),(2-1)は設定必須です。[Packetbeat Reference](#)^[7]を参考にして設定を行ってください。

- (1) `packetbeat.interfaces.device`
 - ルータ装置の中継データを解析したい場合は、ルータOS側でポートモニタリング機能を有効にした上で、`eth0` を指定してください。
- (2) `output.elasticsearch`
 - データの送信先を指定します。logstashに送信したい場合は、`output.logstash` に指定します。elasticsearchとlogstashの両方を指定することはできません（どちらか一つのみ）。
 - データの受信側(elasticsearchなど）で認証を行うように設定している場合は、`username` や`password`を適切に設定してください。
 - (2-1) `hosts`：出力先のIPアドレス、もしくは、ホスト名を指定します。複数の出力先を指定することも可能です。

下記(3)は、ログファイルに関する設定となります（設定は任意です）。コンテナ環境のディスク容量を消費することになるので、必要に応じて設定してください。

- (3) `logging.files`

リスト 12. packetbeatの設定例(/etc/packetbeat/packetbeat.yml抜粋)

```
packetbeat.interfaces.device: eth0    (1)

output.elasticsearch:                (2)
  # Array of hosts to connect to.
  hosts: ["xxx.xxx.xxx.xxx:9200"]    (2-1)

  # Optional protocol and basic auth credentials.
  #protocol: "https"
```

```
username: "elastic"
password: "secret"

logging.to_files: true
logging.files:           (3)
  path: /var/log/packetbeat
  name: packetbeat
  rotateeverybytes: 1048576
  keepfiles: 7
  permissions: 0644
```

設定ファイルの編集が完了したら、サービスを起動します。

リスト 13. packetbeatの起動例

```
container:~# vi /etc/packetbeat/packetbeat.yml
  (設定編集して保存...)
container:~# rc-service packetbeat start
* Starting packetbeat ...
```



コンテナ環境起動時に自動的にpacketbeatを起動するようにしたい場合は下記のように設定してください。インストール後は自動的に起動する設定にはなっていません。

```
container:~# rc-update add packetbeat
* service packetbeat added to runlevel default
```

2.1.5. NetFlow(softflowd)

ルータ装置の中継トラフィックの情報をNetFlowパケットとしてコレクタへ送信することが可能です。softflowdというアプリケーションを利用します。

softflowdの設定方法や使用方法については、[FITElnet LXCアプリケーション^{\[8\]}](#)のページの、「NetFlow(softflowd)使用方法」にまとめておりますので、ご参照ください。

2.2. 参考資料

- [1] apk https://wiki.alpinelinux.org/wiki/Alpine_Linux_package_management
- [2] Alpine Linux package management https://wiki.alpinelinux.org/wiki/Alpine_Linux_package_management
- [3] OpenSSH Server <https://www.openssh.com/portable.html>
- [4] packetbeat <https://www.elastic.co/jp/products/beats/packetbeat>
- [5] Elasticsearch <https://www.elastic.co/jp/products/elasticsearch>
- [6] Logstash <https://www.elastic.co/jp/products/logstash>
- [7] Packetbeat Reference <https://www.elastic.co/guide/en/beats/packetbeat/7.2/index.html>
- [8] FITELnet LXCアプリケーション <https://www.furukawa.co.jp/fitelnet/product/container/lxc/index.html>

付録 A: Elastic Stackを使用したデータの可視化

[Elastic Stack^{\[1\]}](#) は、Elastic社が提供する製品群です。下記の製品を使用すると、様々なデータの収集、分析、可視化などを行うことが可能です。

- Elasticsearch
- Logstash
- Beats
- Kibana

これらの製品はOSSとして利用可能です。

データの分析や解析などは多くのマシンリソースを必要とするので、ElasticsearchやKibanaなどのソフトウェアをルータ装置のコンテナ環境で動作させるにはリソースが不十分ですが、Beatsのような軽量のデータシッパーのソフトウェアであれば問題なく動作します。ルータ装置側ではデータ送信のみを行い、データ分析などはPCサーバのような高性能なマシン側で行うように役割を分担すれば、データの可視化をリアルタイムに行うことが可能です。

本章では、Elastic Stackの製品を使用してルータ装置のデータを可視化する例を示します。

- ネットワーク解析データ
- インタフェースの統計情報
- システムログ



当社では、バージョン7.2のElastic Stack製品を使用して確認しています。

A-1: サーバのセットアップ

下記の製品をPCサーバなどのサーバマシンにセットアップしてください。 [Elasticダウンロードサイト](#)^[2]からダウンロードすることが可能です。

- [Elasticsearch](#)^[3]
 - データの収集、検索、分析などを行います
- [Kibana](#)^[4]
 - Elasticsearchで解析した結果の可視化を行います

A-1-1: Elasticsearchの設定

主に下記項目の設定を行う必要があります。設定ファイル(elasticsearch.yml)を編集してからサービスを再起動してください。

- `network.host`
- `http.port`
- `discovery.seed_hosts`
- `cluster.initial_master_nodes`

詳細は、 [Important Elasticsearch configuration](#)^[5]を参照してください。認証設定などのセキュア設定を行いたい場合は、 [Secure settings](#)^[6]が参考になります。



- 非常に多くのデータを扱う場合は、elasticsearch.ymlに下記の設定を追加しておくことをお勧めします。数値は参考例です。お使いの環境に合わせて設定してください。

```
indices.query.bool.max_clause_count: 8192
search.max_buckets: 100000
```

- データの冗長化や可用性を高めるために、マルチノード上でクラスタを構築することも可能です。必要に応じて設定してください。本書では、シングルノード上でElasticsearchを動作させることを前提としています。

A-1-2: Kibanaの設定

主に下記項目の設定を行ってください。設定ファイル(kibana.yml)を編集してからサービスを再起動してください。

- `server.host`
- `elasticsearch.hosts`
- `elasticsearch.username`
- `elasticsearch.password`

詳細は、[Configuring Kibana^{\[7\]}](#)を参照してください。usernameとpasswordの設定は、Elasticsearch側でセキュリティ設定を行った場合に必要となります。

日本語表記にしたい場合は、`*i18n.locale*`の項目を設定してください。詳細は、[i18n settings in Kibana^{\[8\]}](#)を参照してください。

A-1-3: 時刻同期の設定

リアルタイムにデータの可視化を行うためには、コンテナ環境とサーバ側、及び、データを閲覧する端末側の時刻を同期させておく必要があります。コンテナ環境の時刻同期については、[時刻同期のための設定](#)の章を参照してください。



サーバ側とデータを閲覧する端末側の時刻同期は、お使いのシステムに従って設定を行ってください。

A-2: ネットワーク解析データの可視化

[packetbeat](#)を使用したネットワーク解析データの可視化例を示します。主に下記の設定を行う必要があります。

- アプリケーションのデータ送信設定(コンテナ環境)
- Elasticsearchのデータ受信設定(サーバ環境)
- Kibanaのグラフ作成(サーバ環境)

A-2-1: packetbeat

[packetbeat](#)の章を参考にして、コンテナ環境にpacketbeatをインストールして設定を行います。データの送信先は、Elasticsearchを指定します。送信先はセットアップしたサーバを指定してください。

Kibanaにアクセスして、Elasticsearchの設定とデータ可視化のためのグラフを作成します。サーバの該当するポート（デフォルトは5601番）にHTTP(もしくはHTTPS)アクセスしてください。セキュア設定を行っている場合は、下記のようなログイン画面が表示されます。

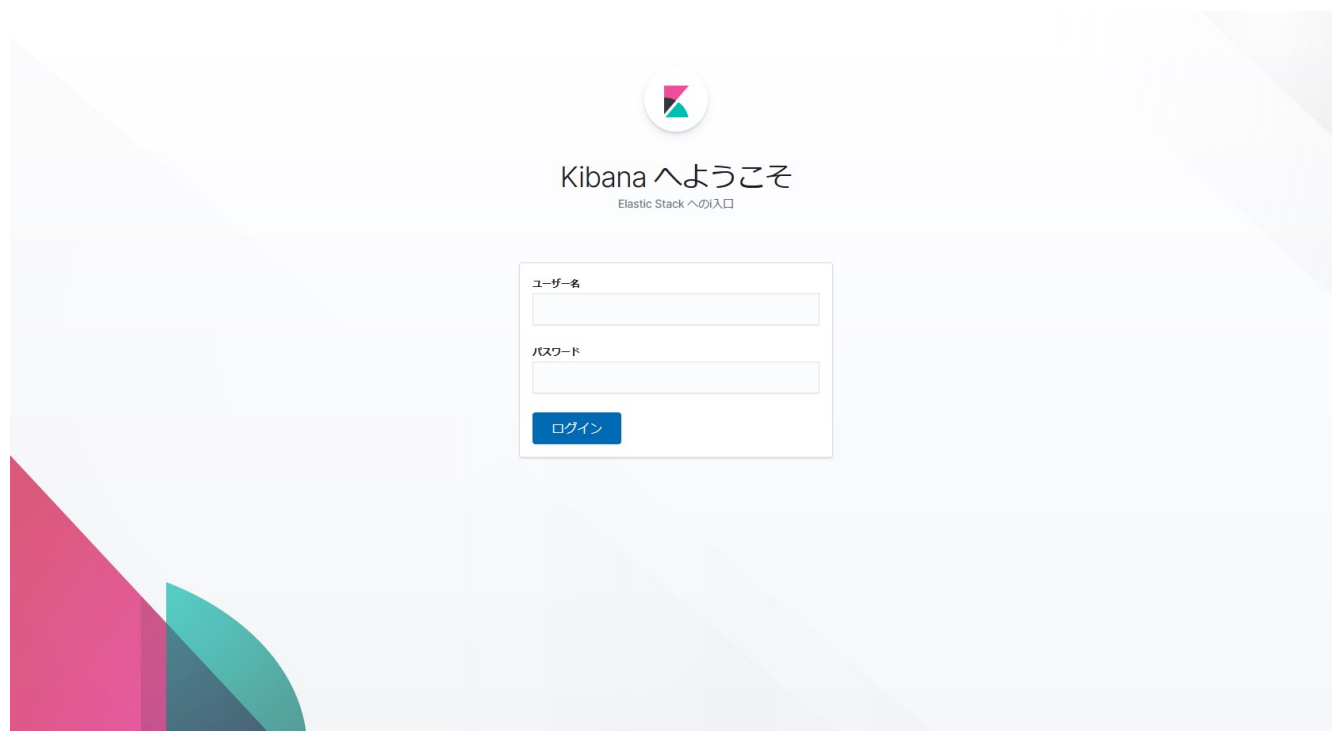


図 7. Kibanaのログイン画面

データを可視化するためには、Elasticsearch側で受信するデータのインデックス作成を行う必要があります（データベースのテーブルのようなもの）。REST APIやKibanaのWebインタフェースを使用して作成することができますが、packetbeatでは、グラフ作成と合わせて一括してセットアップできるコマンドが用意されています。サーバ環境上で下記のコマンドを実行してください。

```
# packetbeat setup --dashboards
```



- サーバ環境にもpacketbeatを予めインストールしておく必要があります。 [Step:1 Install Packetbeat^{\[9\]}](#)を参考にしてください。
- 設定ファイル(packetbeat.yml)の **setup.kibana.host** の項目を設定しておく必要があります（例: "**localhost:5601**" ）。

コマンドが正常に終了すると、Kibanaのダッシュボードにpacketbeatの項目が表示されます。ダッシュボードの画面を開いてpacketbeatを検索してください。



図 8. Kibanaのダッシュボード画面

複数のグラフがヒットします。 **[Packetbeat] Flows ECS** を開くと 下記のような画面が表示されます。

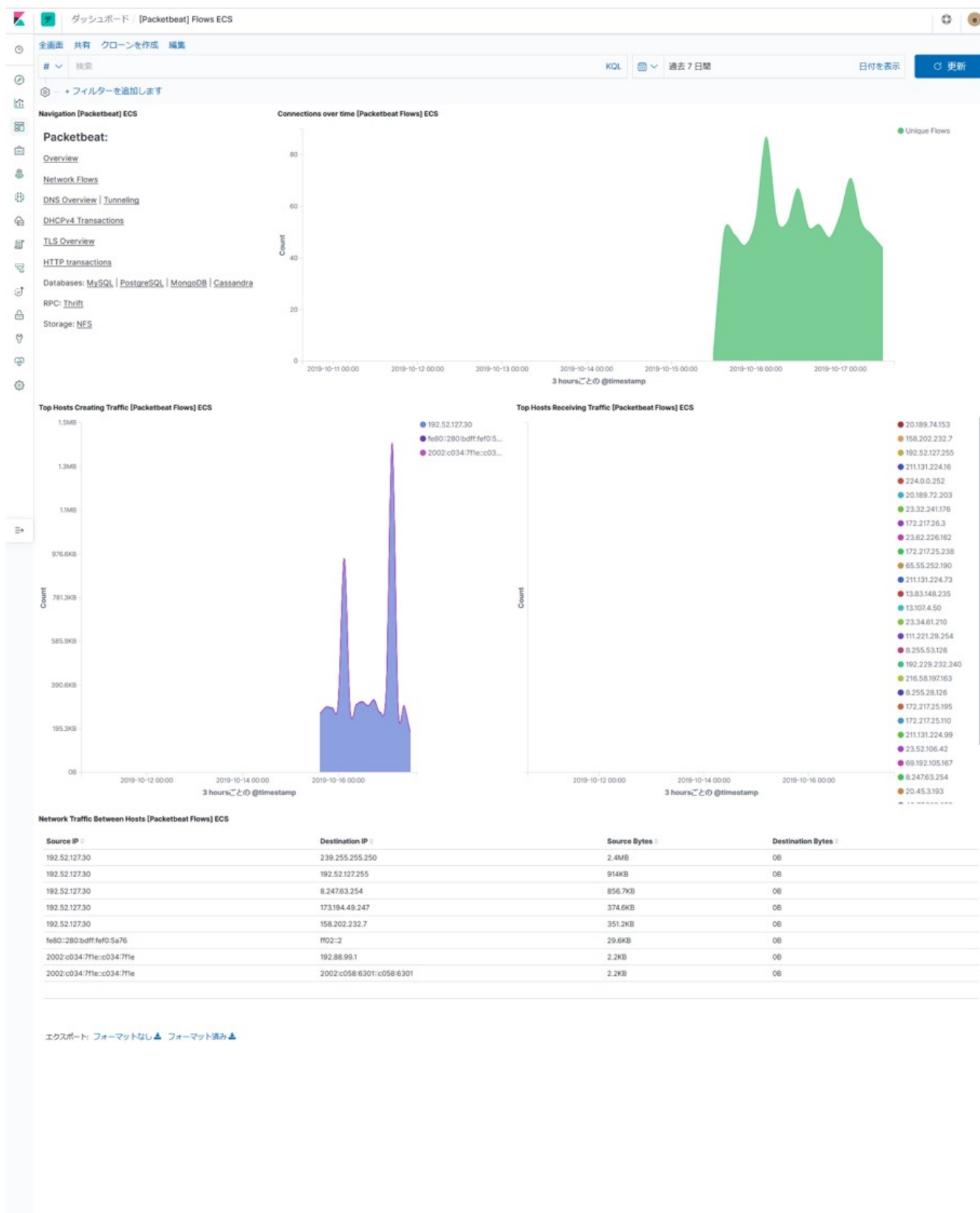


図 9. packetbeatによるネットワークフロー情報の可視化例



画面右上の箇所で時間表示範囲を変更することが可能です。また、グラフ上をドラックすることで時間範囲を指定することも可能です。

packetbeatが作成したグラフ以外にも、Kibana上で独自のグラフを作成することが可能です。 [Visualize your data^{\[10\]}](#)を参考にしてください。

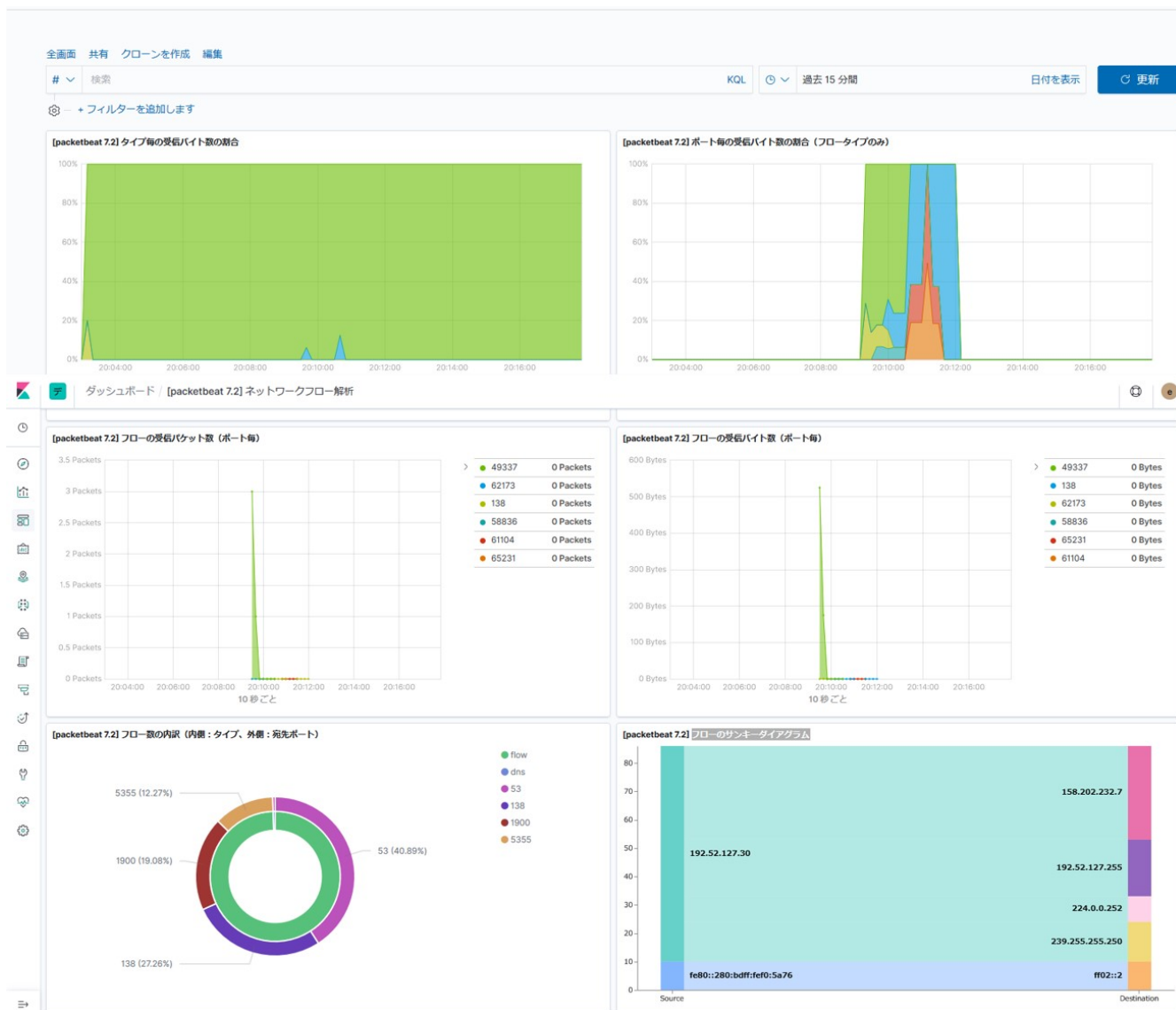


図 10. Kibana可視化

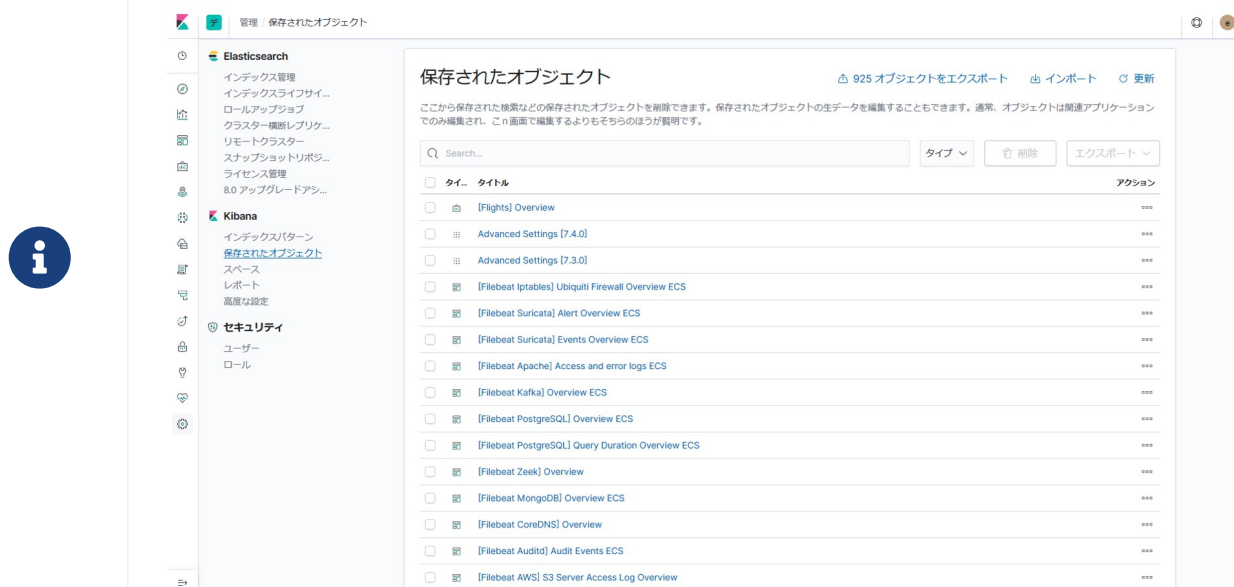
参考例として、下記のグラフを作成した例を示します。これらの画面を作成するための「packetbeat定義ファイル」をダウンロード/解凍して、Kibanaの管理画面（保存されたオブジェクト）からインポートしてください。[FITELnet LXCアプリケーション^{\[11\]}](#)の「各種ツール」の欄に用意しておりますので、ご使用ください。

- タイプ毎の受信バイト数の割合
- ポート毎の受信バイト数の割合（フロータイプのみ）
- フローの受信パケット数（ポート毎）
- フローの受信バイト数（ポート毎）
- フロー数の内訳
- フローのサンキーダイアグラム

ダッシュボードの画面から”[packetbeat 7.2] ネットワークフロー解析”を選択すると、作成したグラフの一覧が下記のように示されます。



定義ファイルのインポートについては、[Saved objects^{\[12\]}](#)を参照してください。下記の管理画面からインポートできます。



A-3: インタフェースの統計情報

[SNMPエージェント](#)の章で示したように、コンテナ環境からルータ装置のMIB情報を取得することが可能です。この情報をElasticsearchに送信して可視化することも可能です。ここでは、filebeatを使用してデータを送信する例を示します。インタフェース情報はIF-MIBのデータを使用します。

[filebeat](#)の章を参考にして、コンテナ環境にfilebeatをインストールして設定を行います。データの送信先は、Elasticsearchを指定します。送信先はセットアップしたサーバをoutput.elasticsearchに指定してください。内容はpacketbeatの場合と同様です。IPアドレスやポート番号、セキュア設定を必要に応じて行ってください。

また、取得したMIBデータをサーバ側へ送信するために、/etc/filebeat/filebeat.ymlに下記の設定を追加します。ファイル経由ではなく、TCPを使用してデータを送信します。

リスト 14. 設定例(/etc/filebeat/filebeat.yml抜粋)

```
filebeat.inputs:
- type: tcp
  host: "127.0.0.1:9006"
  fields:
    tags: "snmp_ifxtbl"
  pipeline: "snmp_ifxtbl"
```

ルータ装置側のGiga 1/3, Giga 1/4, Giga 2/1, Tunnel 1の各インターフェースに対して、下記のMIBデータを取得してElasticsearchへ送信する例を示します。



ルータOS側のSNMPサービスやインタフェースは適切に設定されていることを前提とします。詳細は、[コマンドリファレンス-構成定義編^{\[13\]}](#)を参照してください。

- ifHCOutOctets
- ifHCOutUcastPkts
- ifHCInOctets
- ifHCInUcastPkts

まず、データの取得については、コンテナ環境で下記のようなスクリプトを作成します。スクリプトのファイル名は任意でかまいませんが、本稿では「send_ifxtbl_counter.sh」として説明します。

リスト 15. IF-MIBのデータを取得するスクリプト(send_ifxtbl_counter.sh)例

```
#!/bin/bash
LANG=ja_JP.UTF-8
programe=$(basename $0)
server=127.0.0.1 ①
port=9006 ②
interval=5 ③
date=
mib="ALL"
dir="/usr/share/snmp/mibs:/usr/share/snmp/mibs/iana:/usr/share/snmp/mibs/ietf"
agent="10.10.30.1" ④
community="public" ⑤
ifidx_lan1="401030000" ⑥
ifidx_lan2="401040000" ⑦
ifidx_wan="402010000" ⑧
ifidx_tun="1000000001" ⑨

while true ; do
    date=$(date +%s | tr -d '\n')
    varbinds=$(snmpget -Ln -m $mib -M $dir -IR -Ov -v2c -c $community $agent
ifHCOutOctets.$ifidx_lan1 ifHCOutUcastPkts.$ifidx_lan1 ifHCInOctets.$ifidx_lan1
ifHCInUcastPkts.$ifidx_lan1 ifHCOutOctets.$ifidx_lan2 ifHCOutUcastPkts.$ifidx_la
n2 ifHCInOctets.$ifidx_lan2 ifHCInUcastPkts.$ifidx_lan2 ifHCOutOctets.$ifidx_wan
ifHCOutUcastPkts.$ifidx_wan ifHCInOctets.$ifidx_wan ifHCInUcastPkts.$ifidx_wan
ifHCOutOctets.$ifidx_tun ifHCOutUcastPkts.$ifidx_tun ifHCInOctets.$ifidx_tun
ifHCInUcastPkts.$ifidx_tun | cut -d : -f 2 | tr -d ' ' | tr '\n' ' ')
    [ -z "$varbinds" ] && continue

    varBindList=$(echo $varbinds)
    echo $varbinds | grep NoSuchObject
    [ $? -eq 0 ] && sleep 5 && continue ⑩

    varBindList=$(echo $varbinds)
    lan1_out_bytes=${varBindList[0]}
    lan1_out_pkts=${varBindList[1]}
    lan1_in_bytes=${varBindList[2]}
    lan1_in_pkts=${varBindList[3]}
    lan2_out_bytes=${varBindList[4]}
    lan2_out_pkts=${varBindList[5]}
    lan2_in_bytes=${varBindList[6]}
    lan2_in_pkts=${varBindList[7]}
    wan_out_bytes=${varBindList[8]}
    wan_out_pkts=${varBindList[9]}
    wan_in_bytes=${varBindList[10]}
    wan_in_pkts=${varBindList[11]}
    tun_out_bytes=${varBindList[12]}
    tun_out_pkts=${varBindList[13]}
    tun_in_bytes=${varBindList[14]}
    tun_in_pkts=${varBindList[15]}
```

リスト 16. IF-MIBのデータを取得するスクリプト(send_ifxtbl_counter.sh)例(続き)

```

cat << EOF > /dev/tcp/$server/$port ⑪
{ "timestamp": "$date", "lan1_out_bytes": "$lan1_out_bytes", "lan1_out_pkts": "$lan1_out_pkts",
  "lan1_in_bytes": "$lan1_in_bytes", "lan1_in_pkts": "$lan1_in_pkts",
  "lan2_out_bytes": "$lan2_out_bytes", "lan2_out_pkts": "$lan2_out_pkts", "lan2_in
  _bytes": "$lan2_in_bytes", "lan2_in_pkts": "$lan2_in_pkts", "wan_out_bytes": "$wan_out_bytes",
  "wan_out_pkts": "$wan_out_pkts", "wan_in_bytes": "$wan_in_bytes", "wan_in_pkts": "$wan_in_pkts",
  "tun_out_bytes": "$tun_out_bytes", "tun_out_pkts": "$
  tun_out_pkts", "tun_in_bytes": "$tun_in_bytes", "tun_in_pkts": "$tun_in_pkts" } ⑫
EOF

sleep $interval
done

```

- ① filebeatの設定項目 **filebeat.inputs** に設定したIPアドレスを指定
- ② filebeatの設定項目 **filebeat.inputs** に設定したポート番号を指定
- ③ MIBデータ取得のポーリング間隔を指定
- ④ SNMPエージェントが動作しているルータOS側のIPアドレスを指定
- ⑤ SNMPエージェントのコミュニティ名を指定（この例ではv2cを使用して取得）
- ⑥ Giga 1/3のifIndexを指定
- ⑦ Giga 1/4のifIndexを指定
- ⑧ Giga 2/1のifIndexを指定
- ⑨ Tunnel 1のifIndexを指定
- ⑩ MIBデータを全て取得できなかった場合はリトライを行う
- ⑪ bashのネットワーク機能を使用してfilebeatへ取得したMIBデータを送信する
- ⑫ 取得したMIBデータはJSON形式で送信します

上記のスクリプトでは、取得したMIBデータをJSONフォーマットの形式でfilebeatに送付しています。filebeatでは、このデータをElasticsearchの [Ingestノード](#)^[14] に対して送信します。

Ingestノードの設定はREST APIで行うことができます。以下に設定例を示します。サーバ環境、もしくは、サーバ環境にアクセス可能な端末から下記コマンドを実行してください。

リスト 17. REST APIによるIngestノードの設定例

```

# curl --user username:password -XPUT -H 'Content-Type: application/json'
  ${SERVER}:9200/_ingest/pipeline/snmp_ifxtbl -d @pipeline_snmp_ifxtbl.json

```



- \${SERVER}にはElasticsearchが動作しているサーバのIPアドレスもしくはホスト名を指定します。ポート番号も必要に応じて変更してください。
- サーバ側にセキュア設定を行っている場合は、usernameとpasswordの指定が必要です。必要に応じて設定してください。

pipeline_snmp_ifxtbl.jsonのファイルには、以下の内容を設定します。

リスト 18. Ingestノードの設定例(pipeline_snmp_ifxtbl.json)

```
{
  "description": "SNMP IF-MIB ifXTable",
  "processors": [
    {
      "json": { ①
        "field": "message",
        "add_to_root": true
      }
    },
    {
      "date": { ②
        "field": "timestamp",
        "formats": ["UNIX"]
      }
    },
    {
      "date_index_name": { ③
        "field": "@timestamp",
        "index_name_prefix": "snmp-ifxtbl-",
        "index_name_format": "yyyy.MM.dd",
        "date_rounding": "d"
      }
    },
    { "remove": { "field": "message" } },
    { "convert": { "field": "lan1_out_bytes", "type": "long" } }, ④
    { "convert": { "field": "lan1_out_pkts", "type": "long" } },
    { "convert": { "field": "lan1_in_bytes", "type": "long" } },
    { "convert": { "field": "lan1_in_pkts", "type": "long" } },
    { "convert": { "field": "lan2_out_bytes", "type": "long" } },
    { "convert": { "field": "lan2_out_pkts", "type": "long" } },
    { "convert": { "field": "lan2_in_bytes", "type": "long" } },
    { "convert": { "field": "lan2_in_pkts", "type": "long" } },
    { "convert": { "field": "wan_out_bytes", "type": "long" } },
    { "convert": { "field": "wan_out_pkts", "type": "long" } },
    { "convert": { "field": "wan_in_bytes", "type": "long" } },
    { "convert": { "field": "wan_in_pkts", "type": "long" } },
    { "convert": { "field": "tun_out_bytes", "type": "long" } },
    { "convert": { "field": "tun_out_pkts", "type": "long" } },
    { "convert": { "field": "tun_in_bytes", "type": "long" } },
    { "convert": { "field": "tun_in_pkts", "type": "long" } }
  ]
}
```

- ① [JSONプロセッサ](#)^[15]を使用。インデックステンプレートを予め登録していなくても、自動的にフィールド名とタイプが設定されます。
- ② 送信するデータのフォーマットを指定。詳細は、[Date Processor](#)^[16]を参照してください。
- ③ 作成されるインデックスのフォーマットを指定。詳細は、[Date Index Name Processor](#)^[17]を参照してください。
- ④ データのタイプをMIBカウンタ値のタイプにあわせて変更。

次に、ElasticsearchのインデックステンプレートとKibanaの画面を作成します。

「IFMIBカウンタ定義ファイル」をダウンロード/解凍して、Kibanaの管理画面（保存されたオブジェクト）からインポートしてください。[FITELnet LXCアプリケーション](#)^[11]の「各種ツール」の欄に用意しております。インデックステンプレートとKibanaのグラフがあわせて作成されます。



定義ファイルのインポートについては、[Saved Objects](#)^[18]を参照してください。

コンテナ環境でfilebeatを起動して、MIBデータ取得のためのスクリプト(send_ifxtbl_counter.sh)を実行するとリアルタイムにインタフェースカウンタのデータが可視化されます。

```
root@container:~# rc-service filebeat start
root@container:~# ./send_ifxtbl_counter.sh
```

本章で示した例では、LAN/WAN/TUNNELのカウンタ値としてグラフ上に表記されます。それぞれ、下記のインタフェースカウンタに該当します。

- **LAN**
 - Giga 1/3 と Giga 1/4 の合計
- **WAN**
 - Giga 2/1
- **TUNNEL**
 - Tunnel 1



Kibanaのグラフでは、それぞれのインタフェースカウンタ値の増分をグラフ上で表示しています。ifIndexの値を変更するとカウンタ値を取得するインタフェースを変更することが可能です。さらにインタフェースをLANに追加したい（もしくは削除したい）場合などは、スクリプト、Ingestノード、Kibanaのグラフのそれぞれの設定変更が必要となります。

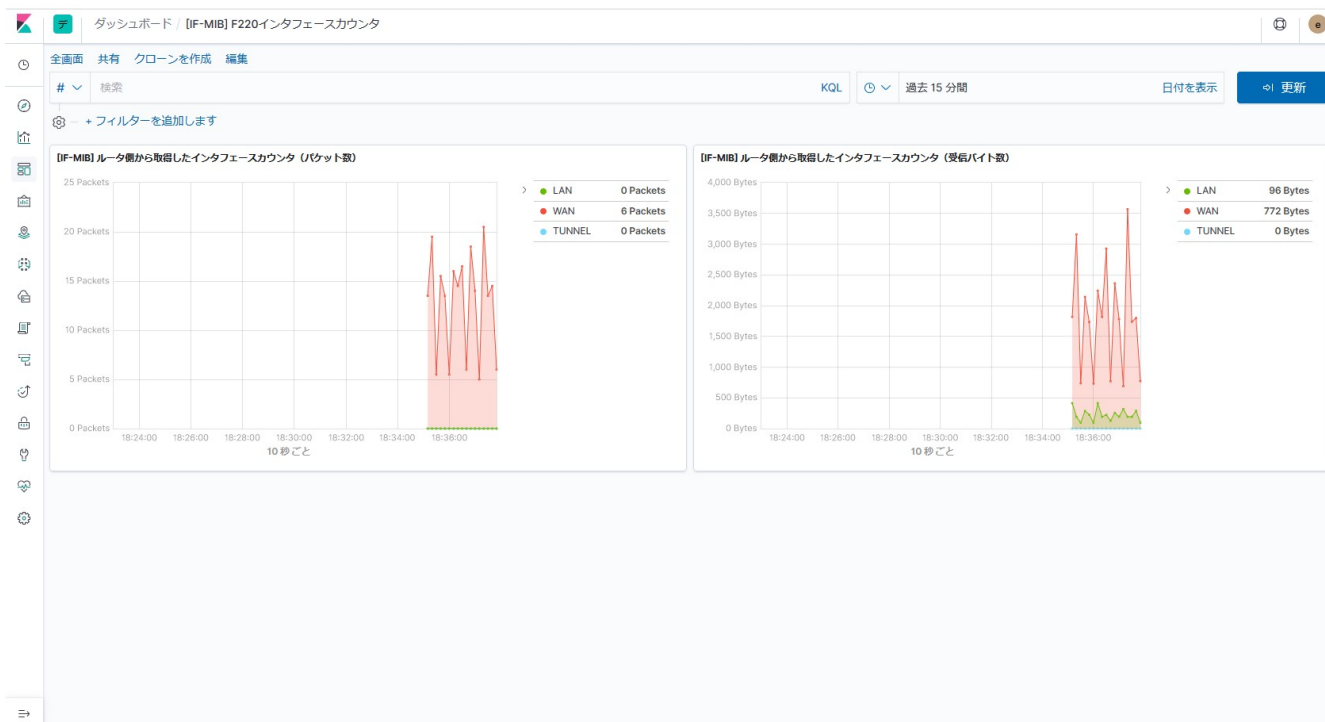


図 13. ルータ装置のインタフェースカウンタ値の可視化例

コンテナ環境から受信したデータの詳細は、Kibanaのディスカバリ画面で確認することができます。

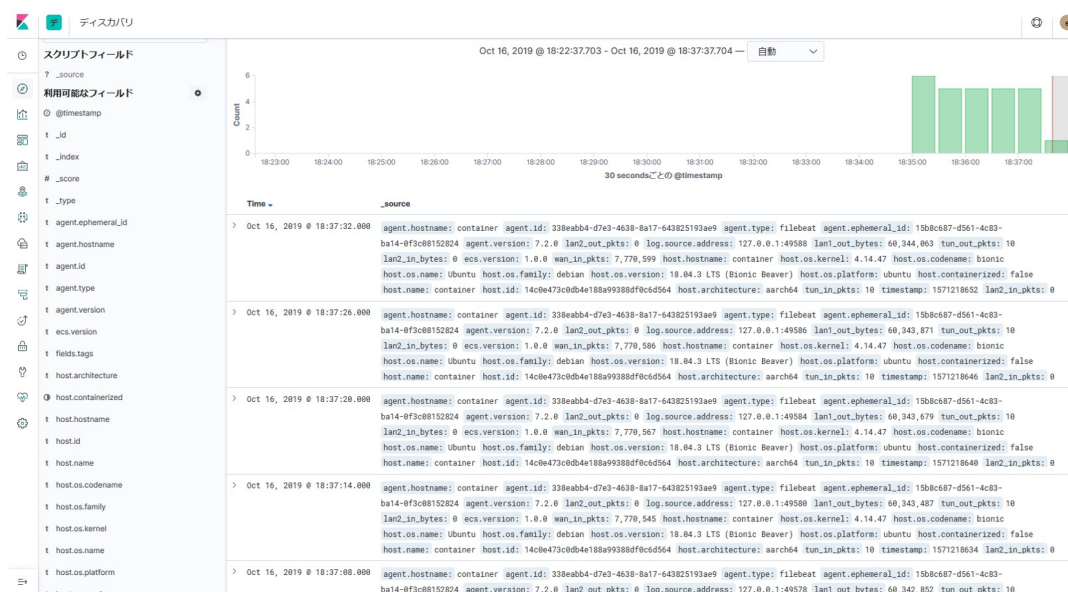


図 14. Kibanaのディスカバリ画面

A-4: システムログ

filebeatから送信したログデータは、Kibanaのログ画面で確認することができます。ElasticsearchやKibanaに追加の設定は必要ありません。

```
filebeat.inputs:
- type: log
  paths:
    - /var/log/syslog
  fields:
    tags: "system_log"
```

例えば、上記のようにsyslogファイルを送信するように設定した場合、syslogの内容を下記のようにリアルタイムで表示できます。

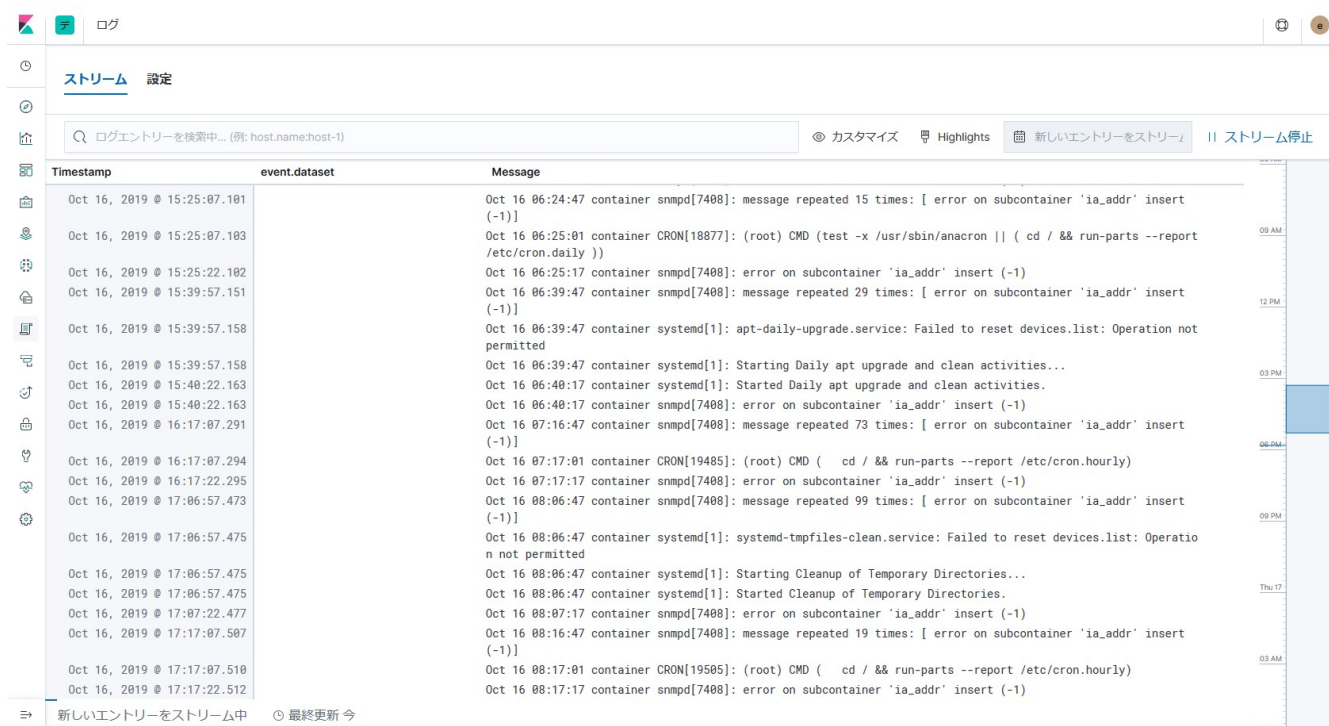


図 15. Kibanaのログ画面

tailコマンドのWebUI版のような画面になっており、ログの検索やフィルタリングを容易に行うことができます。複数のコンテナ環境からログを送信するようにしておけば、サーバ側でログの管理を一括して行うことが可能です。



[journalbeat^{\[19\]}](#)のようなログを可視化するBeatsファミリの製品もあります。こちらは、packetbeatと同様にテーブルやグラフを作成してログが表示されます。

A-5: 参考資料

- [1] Elastic Stack <https://www.elastic.co/jp/products/elastic-stack>
- [2] Elasticダウンロードサイト <https://www.elastic.co/jp/downloads/>
- [3] Elasticsearch <https://www.elastic.co/guide/en/elasticsearch/reference/7.2/setup.html>
- [4] Kibana <https://www.elastic.co/guide/en/kibana/7.2/setup.html>
- [5] Important Elasticsearch configuration <https://www.elastic.co/guide/en/elasticsearch/reference/7.x/important-settings.html>
- [6] Secure settings <https://www.elastic.co/guide/en/elasticsearch/reference/7.2/secure-settings.html>
- [7] Configuring Kibana <https://www.elastic.co/guide/en/kibana/7.x/settings.html>
- [8] i18n settings in Kibana <https://www.elastic.co/guide/en/kibana/7.x/i18n-settings-kb.html>
- [9] Step:1 Install Packetbeat <https://www.elastic.co/guide/en/beats/packetbeat/7.2/packetbeat-installation.html>
- [10] Visualize your data <https://www.elastic.co/guide/en/kibana/7.x/tutorial-visualizing.html>
- [11] FITELnet LXCアプリケーション <https://www.furukawa.co.jp/fitelnet/product/container/lxc/index.html>
- [12] Saved objects <https://www.elastic.co/guide/en/kibana/7.2/managing-saved-objects.html>
- [13] コマンドリファレンス-構成定義編 https://www.furukawa.co.jp/fitelnet/f/man/70_220/pdf/cmd_refe_config.pdf
- [14] Ingestノード <https://www.elastic.co/guide/en/elasticsearch/reference/7.x/ingest.html>
- [15] JSONプロセッサ <https://www.elastic.co/guide/en/elasticsearch/reference/7.x/json-processor.html>
- [16] Date Processor <https://www.elastic.co/guide/en/elasticsearch/reference/7.x/date-processor.html>
- [17] Date Index Name Processor <https://www.elastic.co/guide/en/elasticsearch/reference/7.x/date-index-name-processor.html>
- [18] Saved Objects <https://www.elastic.co/guide/en/kibana/7.2/managing-saved-objects.html>
- [19] journalbeat <https://www.elastic.co/guide/en/beats/journalbeat/7.x/journalbeat-overview.html>

付録 B: USBフラッシュメモリの使用

コンテナ環境でUSBフラッシュメモリを使用する場合は、下記のコマンドをルータOS側に設定する必要があります。

- container device disk



コマンドの詳細は、[コマンドリファレンス-構成定義編](#)を参照してください。

例えば、コンテナ環境内の /tmp/usb1 のディレクトリに対してマウントしたい場合、下記のように設定します。

```
Router(config)#container device disk /tmp/usb1 usb 1
```

上記の設定を反映すると、コンテナ環境では /tmp/usb1 のディレクトリが自動的に作成されます。このディレクトリ内に ファイルを保存すると、USBフラッシュメモリ内に保存されます。



- 上記の設定を行う前に、ルータOS側でUSB1にUSBモジュールをmountしておく必要があります。
- コンテナ環境側に作成されたディレクトリ（上記の例では /tmp/usb1）は、USBフラッシュメモリをumountしても自動的に削除されません。必要に応じて手動で削除してください。

付録 C: ポートモニタリング機能の使用

ルータOSのポートモニタリング機能を使用すると、ルータ装置の中継データをコンテナ環境へ出力することが可能です。この機能を利用することで、コンテナ環境で中継データの解析や可視化を行うことができます。



ポートモニタリング機能の設定については、[コマンドリファレンス-運用管理編](#)を参照してください。

リスト 19. ポートモニタリング機能の設定例

```
F220#port-monitor mirrored gigaethernet 2/1 in

F220#port-monitor monitor container

F220#show port-monitor

[Current state]
  mirrored port(ingress):
  mirrored port(egress) :
  monitor port: container
F220#
```

上記のようにポートモニタリング結果の出力先をコンテナ環境に設定すると、コンテナ環境側では **eth0** のインタフェースが自動的に追加されます。このインタフェースに対してIPアドレスの設定は必要ありません。**eth0** のインタフェースがダウンした状態のままである場合は、下記のようにインタフェースをアップしてからご利用ください。

```
root@container:~# ip link set eth0 up
```



ポートモニタリング機能を使用する場合は、下記の点にご注意ください。

- 高負荷な中継データをキャプチャする場合は、ルータ装置の中継性能に影響を与える可能性があります。また、全ての中継データをキャプチャできるとは限りません。
- eth0** 以外のインターフェースを追加している状態でrebootコマンドを実行すると、コンテナ環境の起動に失敗する可能性があります。rebootする場合は、ポートモニタの設定を削除（no port-monitor monitor container）してください。
 - もし起動が失敗した場合は、再度、ルータOSのCLIから [コンテナ環境を起動](#) してください。
 - ルータOSのCLIからコンテナ環境を再起動（container restart）する場合は問題ありません。
- コンテナ環境の再起動等で **eth0** が削除されてしまった場合は、再度、ポートモニタの設定（port-monitor monitor container）を行ってください。

FITELnet F70/F71/F220/F221/F225/F310/F220 EX/F221 EX コンテナ型仮想環境の使用方法

130-M0976-BS01-F

発行日 2025年8月

発行責任 古河電気工業株式会社

-
- 本書の一部または全部を無断で他に転載しないよう、お願いいたします。
 - 本書は、改善のために予告なしに変更することがあります。
 - 本書に記載されたデータの使用に起因する第三者の特許権、その他の権利、損害については、弊社はその責を負いません。