【Practical Tips for SPARC】

第 7 回：SPARC Servers / Solaris 上で IoT 分析環境構築 (3/5)
～ファイヤーウォールゾーンとアプリケーションサーバゾーンの構築～

2018 年 5 月

今回は作成した dmz ドメインにアプリケーションサーバのインストールと実際のアプリケーションを動作させます。
先に DMZ 領域とイントラネット領域の通信を行うために、ファイヤーウォールゾーンを作成し、その後、アプリケーションサーバゾーンを作成していきます。

ファイヤーウォールゾーンの構築

アプリケーションサーバゾーンを構築する前に、ファイヤーウォールゾーンを作成しておきます。

ファイヤーウォールゾーンの構成

構成は以下の通りです。
ファイヤーウォールは ILB でも使用した VRRP 機能を使用して 3 ノード間で冗長化を行います。

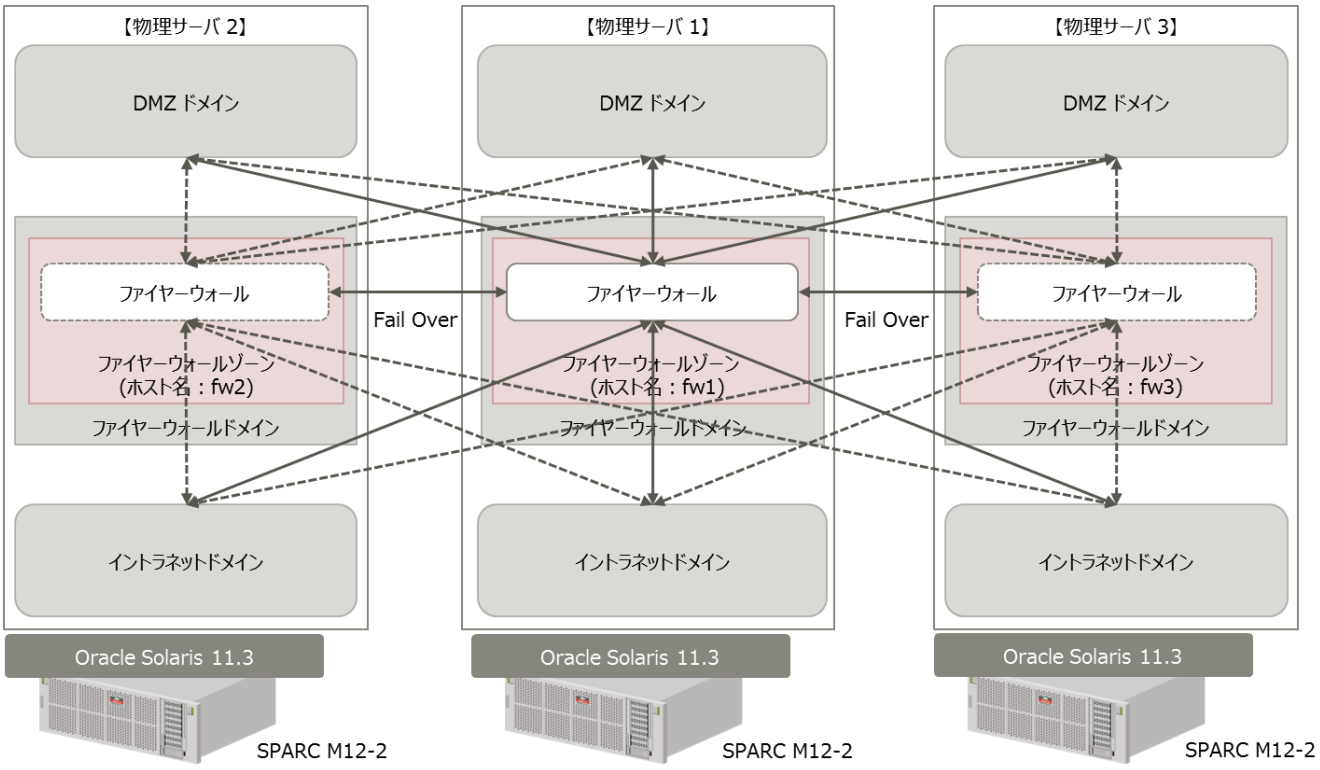

図1 ファイヤーウォール論理構成イメージ

図2 ファイヤーウォールゾーン構成図

ファイヤーウォールゾーンのインストール

fw ドメイン上に root ユーザーでログインします。特に断りが無い限り、同じ処理をすべてのノードで行います。
物理ポートとリンク名の確認を行います。ここで表示される物理ポートの vnet0 と vnet1 は、fw ドメイン作成時に追加した vnet ではなく、fw ドメインに追加された順番に OS によって指定される物理ポートの名前です。

```
# dladm show-phys
LINK              MEDIA             STATE     SPEED  DUPLEX    DEVICE
net1              Ethernet          up        0      unknown   vnet1
net0              Ethernet          up        0      unknown   vnet0
```

以下のようにファイヤーウォールゾーンを構成します。zone 名は fw とします。ネットワーク設定は anet で GZ 上のデータリンクとファイヤーウォールゾーンのデータリンクを結び付けています。

```
# zonecfg -z fw
zonecfg:ilb> create -b
zonecfg:ilb> set brand=solaris
zonecfg:ilb> set zonepath=/zones/fw
zonecfg:ilb> set ip-type=exclusive
zonecfg:ilb> set autoboot=false
zonecfg:ilb> add dedicated-cpu
zonecfg:ilb> set ncpus=16
zonecfg:ilb> end
zonecfg:ilb> add anet
zonecfg:ilb:anet> set linkname=net0
zonecfg:ilb:anet> set lower-link=net1
```

```
zonecfg:ilb:anet> set configure-allowed-address=true
zonecfg:ilb:anet> set link-protection=mac-nospoof
zonecfg:ilb:anet> set mac-address=auto
zonecfg:ilb:anet> end
zonecfg:ilb> add anet
zonecfg:ilb:anet> set linkname=net1
zonecfg:ilb:anet> set lower-link=net2
zonecfg:ilb:anet> set configure-allowed-address=true
zonecfg:ilb:anet> set link-protection=mac-nospoof
zonecfg:ilb:anet> set mac-address=auto
zonecfg:ilb:anet> end
zonecfg:ilb> commit
zonecfg:ilb> exit
```

ファイヤーウォールゾーンのインストールを行います。

```
# zoneadm -z ilb install
```

ファイヤーウォールゾーンをブートします。

```
# zoneadm -z ilb boot
```

ファイヤーウォールゾーンのコンソールにログインして OS の初期設定を行います。設定方法は以下を参照してください。
http://www.fujitsu.com/jp/sparc-technical/document/solaris/index.html#solaris-zone

ホスト名は図 1 を参考にしてください。
ネットワークは net0 側が DMZ 向け LAN、net1 側がイントラネット向け LAN となるように設定します。初期設定時に net0 を設定した場合は、net1 の IP インターフェースの作成から行います。
以下は初期設定時に net0 を設定した場合のネットワーク設定例です。

```
root@fw1:~# ipadm
NAME              CLASS/TYPE STATE        UNDER      ADDR
lo0               loopback   ok           --         --
   lo0/v4         static     ok           --         127.0.0.1/8
   lo0/v6         static     ok           --         ::1/128
net0              ip         ok           --         --
   net0/v4        static     ok           --         <DMZ LAN 側 IP アドレス>/<ネットマスク>
   net0/v6        addrconf   ok           --         fe80::214:4fff:fefa:b8e7/10
root@fw1:~# ipadm create-ip net1
root@fw1:~# ipadm create-addr -T static -a <イントラネット LAN 側 IP アドレス>/<ネットマスク> net1/v4
root@fw1:~# ipadm
NAME              CLASS/TYPE STATE        UNDER      ADDR
lo0               loopback   ok           --         --
   lo0/v4         static     ok           --         127.0.0.1/8
   lo0/v6         static     ok           --         ::1/128
net0              ip         ok           --         --
   net0/v4        static     ok           --         <DMZ LAN 側 IP アドレス>/<ネットマスク>
   net0/v6        addrconf   ok           --         fe80::214:4fff:fefa:b8e7/10
net1              ip         ok           --         --
   net1/v4        static     ok           --         <イントラネット LAN 側 IP アドレス>/<ネットマスク>
```

/etc/inet/hosts に通信するノードの IP アドレスを追加します。

必要なパッケージのインストール

ファイヤーウォールゾーンに以下のパッケージをインストールします。network/firewall はグローバルゾーンにもインストールしておきます。

```
root@fw1:~# pkg install system/network/routing
root@fw1:~# pkg install system/network/routing/vrrp
root@fw1:~# pkg install network/firewall
```

ipforward を設定します。

```
root@fw1:~# ipadm set-prop -p forwarding=on ipv4
```

ipforward の状態を確認する。IPv4 forwarding が enabled になっていることを確認します。

```
root@fw1:~# routeadm
             Configuration   Current             Current
                    Option   Configuration       System State
---------------------------------------------------------------
              IPv4 routing   disabled            disabled
              IPv6 routing   disabled            disabled
            IPv4 forwarding   enabled             enabled
            IPv6 forwarding   disabled            disabled

          Routing services   "route:default ripng:default"

Routing daemons:

                     STATE   FMRI
                    online   svc:/network/routing/ndp:default
                  disabled   svc:/network/routing/rdisc:default
                  disabled   svc:/network/routing/route:default
                  disabled   svc:/network/routing/ripng:default
                  disabled   svc:/network/routing/legacy-routing:ipv6
                  disabled   svc:/network/routing/legacy-routing:ipv4
```

VRRP の設定

VRRP ルーターを作成します。ILB と異なり両方のネットワークインターフェースに対して作成します。
VRRP を作成する vrrpadm コマンドの詳細は前回を参照してください。
今回も ipmp を考慮して L3 タイプとします。仮想ルーター識別子(VRID)は DMZ 側が 2、イントラネット側が 3 とします。

冗長化に関しては fw1 を MASTER とし、fw2 と fw3 を BACKUP とする 3 台構成です。fw1 -> fw2 -> fw3 の順番で切り替わります。

<fw1 の場合>

```
root@fw1:~# vrrpadm create-router -V 2 -I net0 -p 255 -A inet -T l3 vrrp2
root@fw1:~# vrrpadm create-router -V 3 -I net1 -p 255 -A inet -T l3 vrrp3
```

＜fw2 の場合＞

```
root@fw2:~# vrrpadm create-router -V 2 -I net0 -p 150 -A inet -T l3 vrrp2
root@fw2:~# vrrpadm create-router -V 3 -I net1 -p 150 -A inet -T l3 vrrp3
```

＜fw3 の場合＞

```
root@fw3:~# vrrpadm create-router -V 2 -I net0 -p 100 -A inet -T l3 vrrp2
root@fw3:~# vrrpadm create-router -V 3 -I net1 -p 100 -A inet -T l3 vrrp3
```

ルーターが使用する仮想 IP アドレスを VRRP ルーターに指定します。

```
root@fw1:~# ipadm create-addr -T vrrp -n vrrp2 -a <DMZ LAN 仮想 IP アドレス>/<ネットマスク> net0/vrrp
root@fw1:~# ipadm create-addr -T vrrp -n vrrp3 -a <イントラネット仮想 IP アドレス>/<ネットマスク> net1/vrrp
```

VRRP サービスを online にします。

```
root@fw1:~# svcadm enable vrrp
root@fw1:~# svcs vrrp
STATE          STIME    FMRI
online         15:45:52 svc:/network/vrrp:default
```

ルーターの確認をします。

<fw1 の場合>

```
root@fw1:~# vrrpadm show-router
NAME    VRID TYPE IFNAME  AF   PRIO ADV_INTV MODE  STATE VNIC
vrrp3   3    L3   net1    IPv4 100  1000     eopa- MASTER --
vrrp2   2    L3   net0    IPv4 100  1000     eopa- MASTER --
```

<fw2 の場合>

```
root@fw2:~# vrrpadm show-router
NAME    VRID TYPE IFNAME  AF   PRIO ADV_INTV MODE  STATE VNIC
vrrp3   3    L3   net1    IPv4 150  1000     e-pa- BACKUP --
vrrp2   2    L3   net0    IPv4 150  1000     e-pa- BACKUP --
```

<fw3 の場合>

```
root@fw3:~# vrrpadm show-router
NAME    VRID TYPE IFNAME  AF   PRIO ADV_INTV MODE  STATE VNIC
vrrp3   3    L3   net1    IPv4 100  1000     e-pa- BACKUP --
vrrp2   2    L3   net0    IPv4 100  1000     e-pa- BACKUP -
```

Packet Fileter の設定

セキュリティのためにファイヤーウォールの設定を行い、不要なポートを閉じます。
まず、SMF を Online にします。

```
root@fw1:~# svcadm enable svc:/network/firewall:default
```

"pfconf"コマンドを使用して、ファイヤーウォールを設定します。以下の例は 8080 番と 443 番(https)のポートのみを開放した場合です。
"224.0.0.18/32"は VRRP が相互にハートビート監視を行うためのブロードキャストアドレスです。VRRP を構成するすべてのノード間でこのアドレスへの通信を許可します。

```
root@fw1:~# pfconf
block in on net0 all
pass in on net1 all
pass out all
pass in on net0 proto tcp from any to any port = 8080
pass in on net0 proto tcp from any to any port = 443
pass in on net0 proto icmp from any to any
pass out quick on net0 from <fw1 の DMZ LAN 側 IP アドレス>/32 to 224.0.0.18/32
pass in quick on net0 from <fw1 の DMZ LAN 側 IP アドレス>/32 to 224.0.0.18/32
pass out quick on net0 from <fw2 の DMZ LAN 側 IP アドレス>/32 to 224.0.0.18/32
pass in quick on net0 from <fw2 の DMZ LAN 側 IP アドレス>/32 to 224.0.0.18/32
pass out quick on net0 from <fw3 の DMZ LAN 側 IP アドレス>/32 to 224.0.0.18/32
pass in quick on net0 from <fw3 の DMZ LAN 側 IP アドレス>/32 to 224.0.0.18/32
```

正しく設定が行われていることを確認します。

```
root@fw1:~# pfctl -s rules
block drop in on net0 all
pass in on net0 proto tcp from any to any port = 8080 flags S/SA
pass in on net0 proto tcp from any to any port = 443 flags S/SA
pass in on net0 proto icmp all
pass in on net1 all flags S/SA
pass out all flags S/SA
pass out quick on net0 inet from < fw1 の IP アドレス> to 224.0.0.18 flags S/SA
pass out quick on net0 inet from <fw2 の IP アドレス> to 224.0.0.18 flags S/SA
pass out quick on net0 inet from <fw3 の IP アドレス> to 224.0.0.18 flags S/SA
pass in quick on net0 inet from <fw1 の IP アドレス> to 224.0.0.18 flags S/SA
pass in quick on net0 inet from <fw2 の IP アドレス> to 224.0.0.18 flags S/SA
pass in quick on net0 inet from <fw3 の IP アドレス> to 224.0.0.18 flags S/SA
```

以上でファイヤーウォールゾーンの設定は完了です。

アプリケーションサーバを Solaris で効率的に動作させるワザ

構築の前にアプリケーションサーバを Solaris で効率的に動作させるワザについて解説します。
CPU のメニーコア化に伴って、ひとつのノードで多くのプロセス/スレッドを起動することができるようになってきました。アプリケーションサーバも処理の増大に対応するために複数のインスタンスやスレッドを起動するようになってきました。
その場合、クロスコールと呼ばれる各 CPU 間での割り込み処理が増大し、単純にコアを追加しても性能があまり上がらなくなります。
Solaris では Zone で使用する CPU を指定することにより、プロセスの CPU への割り当てを効率化しクロスコールの発生を抑えることができます。
さらに Zone が使用する CPU に対して OS の処理が割り当たらないようにし、該当の CPU をアプリケーションサーバの処理が可能性です。
そうすることで不要な処理が減り、CPU パワーを有効に活用することが可能となります。

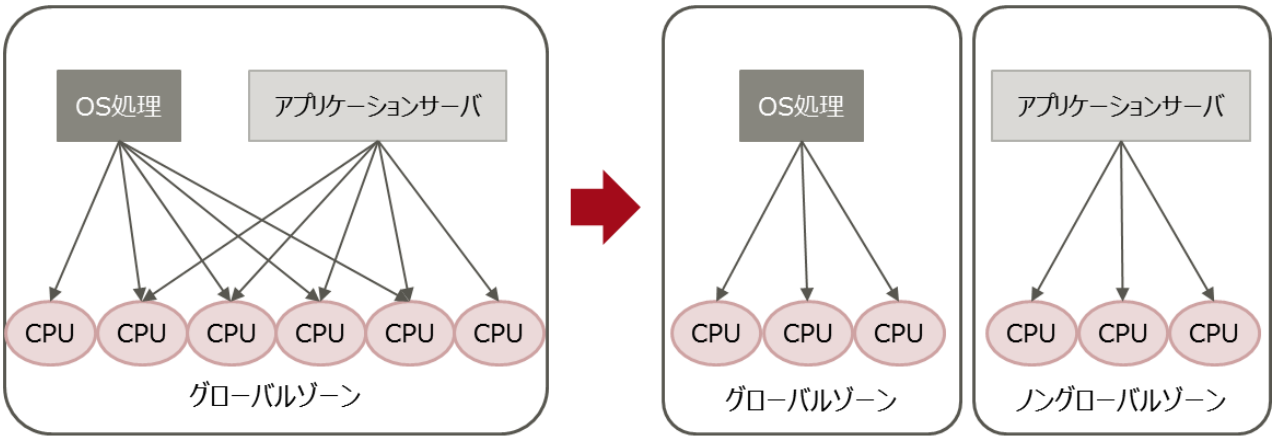

図3 処理イメージ

アプリケーションサーバゾーンの構成

構成は以下の通りです。
Payara と Kafka をインストールします。
今回は以下のバージョンで確認しています。

- payara　　4.1.2.173
- kafka　　2.12-1.0.0

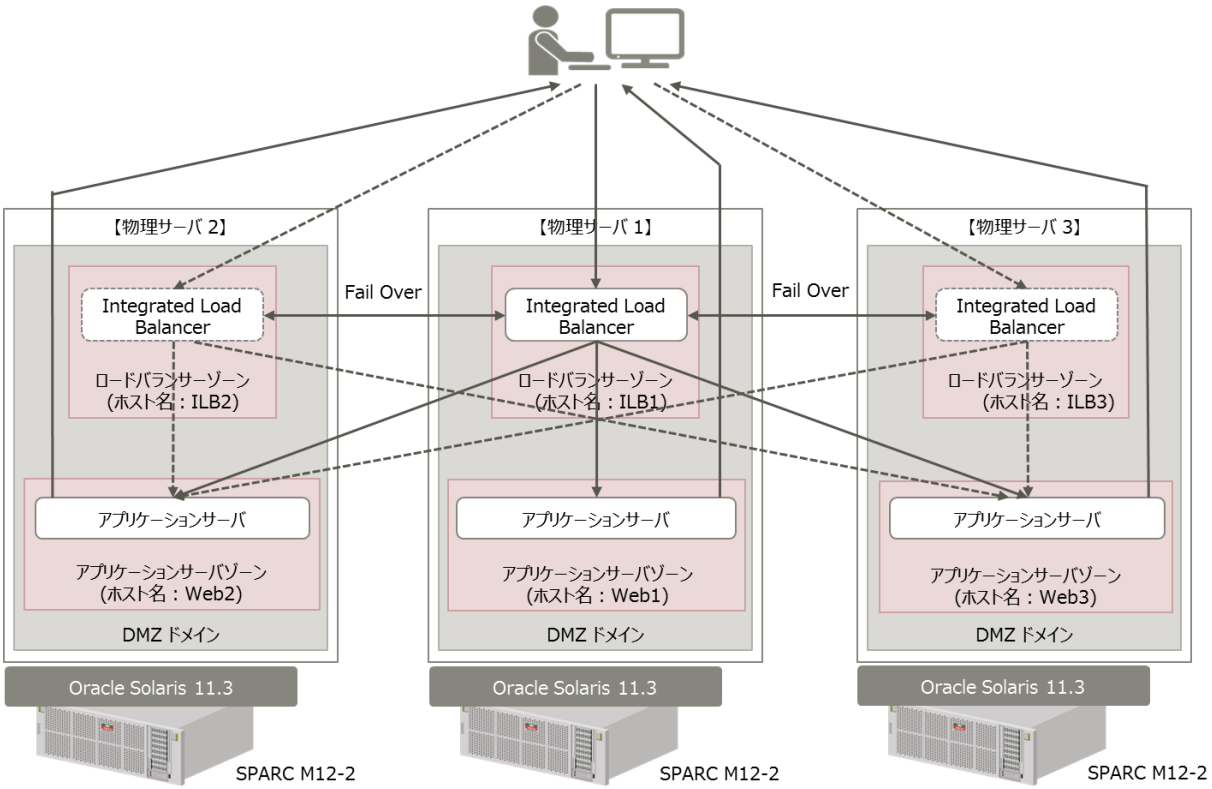

図4 論理構成イメージ

図5 アプリケーションサーバゾーン構成図

アプリケーションサーバゾーンの構築

構成は以下の通りです。

NGZ のインストール

dmz ドメイン上に root でログインします。
アプリケーションサーバゾーンに割り当てる CPU を確認します。

```
# psrinfo
0       on-line   since 01/05/2017 07:47:42
1       on-line   since 01/05/2017 07:47:43
2       on-line   since 01/05/2017 07:47:43

（省略）

21      on-line   since 01/05/2017 07:47:43
22      on-line   since 01/05/2017 07:47:43
23      on-line   since 01/05/2017 07:47:43
```

Global Zone 上でアプリケーションサーバゾーンが使用する CPU への I/O 割り込みを禁止する設定をします。以下の例では 8 番から 15 番の CPU に設定しています。

```
# psradm -i 16-23
```

設定を保存します。

```
# pooladm -s
```

物理ポートとリンク名の確認を行います。ここで表示される vnet は ldm コマンドで作成したものではなく、追加されたものから順番に vnet0, vnet1 と振られることに注意してください。

```
# dladm show-phys
LINK              MEDIA                STATE      SPEED  DUPLEX    DEVICE
net1              Ethernet             up         0      unknown   vnet1
net0              Ethernet             up         0      unknown   vnet0
```

以下のように NGZ を構成します。zone 名は web とします。ネットワーク設定は anet で GZ 上のデータリンクと NGZ のデータリンクを結び付けています。
dedicated-cpu の cpus オプションで使用する CPU を 16 から 23 に指定しています。

```
# zonecfg -z web
zonecfg:ilb> create -b
zonecfg:ilb> set brand=solaris
zonecfg:ilb> set zonepath=/zones/web
zonecfg:ilb> set ip-type=exclusive
zonecfg:ilb> set autoboot=false
zonecfg:ilb> add dedicated-cpu
zonecfg:ilb> set cpus=16-23
zonecfg:ilb> end
zonecfg:ilb> add anet
zonecfg:ilb:anet> set linkname=net0
```

```
zonecfg:ilb:anet> set lower-link=net0
zonecfg:ilb:anet> set configure-allowed-address=true
zonecfg:ilb:anet> set link-protection=mac-nospoof
zonecfg:ilb:anet> set mac-address=auto
zonecfg:ilb:anet> end
zonecfg:ilb> add anet
zonecfg:ilb:anet> set linkname=net1
zonecfg:ilb:anet> set lower-link=net1
zonecfg:ilb:anet> set configure-allowed-address=true
zonecfg:ilb:anet> set link-protection=mac-nospoof
zonecfg:ilb:anet> set mac-address=auto
zonecfg:ilb:anet> end
zonecfg:ilb> commit
zonecfg:ilb> exit
```

アプリケーションサーバゾーンのインストールを行います。

```
# zoneadm -z web install
```

アプリケーションサーバゾーンをブートします。

```
# zoneadm -z web boot
```

アプリケーションサーバゾーンのコンソールにログインして OS の初期設定を行います。設定方法は以下を参照してください。

http://www.fujitsu.com/jp/sparc-technical/document/solaris/index.html#solaris-zone

ホスト名は図 2 を参考にしてください。

ネットワークは net0 側がインターネット向け LAN、net1 側が DMZ 向け LAN となるように設定します。初期設定時に net0 を設定した場合は、net1 の IP インターフェースの作成から行います。

以下は初期設定時に net0 を設定した場合のネットワーク設定例です。

```
root@fw1:~# ipadm
NAME              CLASS/TYPE STATE        UNDER      ADDR
lo0               loopback   ok           --         --
   lo0/v4         static     ok           --         127.0.0.1/8
   lo0/v6         static     ok           --         ::1/128
net0              ip         ok           --         --
   net0/v4        static     ok           --         <インターネット LAN 側 IP アドレス>/<ネットマスク>
   net0/v6        addrconf   ok           --         fe80::214:4fff:fef8:8f7b/10
root@fw1:~# ipadm create-ip net1
root@fw1:~# ipadm create-addr -T static -a <DMZ LAN 側 IP アドレス>/<ネットマスク> net1/v4
root@fw1:~# ipadm
NAME              CLASS/TYPE STATE        UNDER      ADDR
lo0               loopback   ok           --         --
   lo0/v4         static     ok           --         127.0.0.1/8
   lo0/v6         static     ok           --         ::1/128
net0              ip         ok           --         --
   net0/v4        static     ok           --         <インターネット LAN 側 IP アドレス>/<ネットマスク>
   net0/v6        addrconf   ok           --         fe80::214:4fff:fef8:8f7b/10
net1              ip         ok           --         --
   net1/v4        static     ok           --         <DMZ LAN 側 IP アドレス>/<ネットマスク>
```

DSR モードでシステムを運用する場合、ILB からアプリケーションサーバゾーン側に転送されるパケットの送信先 IP アドレスは仮想 IP アドレスとなるため、アプリケーションサーバゾーンではそのパケットが自ノード宛であると認識することができず、受け取ることができません。アプリケーションサーバゾーンに転送されたパケットを自ノード宛であると認識させるため、仮想ネットワークインターフェース(vni)を使用して、仮想 IP アドレスをアプリケーションサーバゾーンに設定します。
まず、vni を作成します。vni 名は任意です。

```
root@fw1:~# ipadm create-vni vni0
```

作成した vni に IP アドレスを設定します。

```
root@fw1:~# ipadm create-addr -T static -a <仮想 IP アドレス>/<NETMASK> vni0/vip
```

以下のように vni が設定されていることを確認します。

```
root@fw1:~# ipadm
NAME              CLASS/TYPE STATE        UNDER      ADDR
lo0               loopback   ok           --         --
   lo0/v4         static     ok           --         127.0.0.1/8
   lo0/v6         static     ok           --         ::1/128
net0              ip         ok           --         --
   net0/v4        static     ok           --         <インターネット LAN 側 IP アドレス>/<ネットマスク>
   net0/v6        addrconf   ok           --         fe80::214:4fff:fef8:8f7b/10
net1              ip         ok           --         --
   net1/v4        static     ok           --         <DMZ LAN 側 IP アドレス>/<ネットマスク>
vni0              vni        ok           --         --
   vni0/vip       static     ok           --         <仮想 IP アドレス>/<ネットマスク>
```

IP アドレスの設定が完了したら、Defaut Gateway を設定します。
net1 側のデフォルトゲートウェイはファイヤーウォールゾーンに作成した vrrp の仮想アドレスを指定します。

```
root@fw1:~# route -p add <イントラネットのネットワークアドレス> <ファイヤーウォールゾーンに作成する vrrp の仮想アドレス>
root@fw1:~# route -p show
persistent: route add -inet default <インターネット LAN 側デフォルトゲートウェー>
persistent: route add <イントラネットのネットワークアドレス> <ファイヤーウォールゾーンに作成する vrrp の仮想 IP アドレス>
```

/etc/inet/hosts に通信するノードの IP アドレスを追加します。

必要なパッケージのインストール

アプリケーションサーバゾーンに以下のパッケージをインストールします。network/firewall はグローバルゾーンにもインストールしておきます。

```
# pkg install network/firewall
```

Payara のインストール

Payara をダウンロードし、インストールするノードに転送します。
https://www.payara.fish/downloads
Payara をインストールします。

```
# cd /opt
# <Payara のアーカイブの解凍>
# ln -s payara41 payara
```

管理者用のパスワードを設定します。

```
root@web1:~# /opt/payara/bin/asadmin change-admin-password
Enter admin user name [default: admin]>admin
Enter the admin password> (初回は何も入力しません)
Enter the new admin password>  (パスワードを入力)
Enter the new admin password again>  (パスワードを再入力)
Command change-admin-password executed successfully.
```

アプリケーションサーバのドメインを起動します。

```
root@web1:~# /opt/payara/bin/asadmin start-domain domain1
Waiting for domain1 to start .....
Successfully started the domain : domain1
domain  Location: /opt/payara41/glassfish/domains/domain1
Log File: /opt/payara41/glassfish/domains/domain1/logs/server.log
Admin Port: 4848
Command start-domain executed successfully.
```

GUI で管理ができるように設定を変更します。

```
root@web1:~# /opt/payara/bin/asadmin enable-secure-admin
Enter admin user name>  admin
Enter admin password for user "admin">
You must restart all running servers for the change in secure admin to take effect.
Command enable-secure-admin executed successfully.
```

設定を変更したらドメインを再起動します。

```
root@web1:~# /opt/payara/bin/asadmin restart-domain domain1
Successfully restarted the domain
Command restart-domain executed successfully.
```

Kafka のインストール

Apache Kafka をダウンロードし、インストールするノードに転送します。

https://kafka.apache.org/downloads

Apache Kafka をインストールします。

```
# cd /opt
# <Kafka のアーカイブの解凍>
# ln -s kafka_2.12-1.0.0 kafka
```

Packet Fileter の設定

セキュリティのためにファイヤーウォールの設定を行い、不要なポートを閉じます。
ファイヤーウォールゾーンと同様に、SMF を Online にします。

```
root@fw1:~# svcadm enable svc:/network/firewall:default
```

"pfconf"コマンドを使用して、ファイヤーウォールを設定します。
今回の構成ではアプリケーションサーバゾーンにはインターネット LAN から直接アクセスできないようにするため、net0 のポートをすべて閉じます。

```
root@web1:~# pfconf
block in on net0 all
pass out all
```

正しく設定が行われていることを確認します。

```
root@web1:~# pfctl -s rules
block drop in on net0 all
pass out all flags S/SA
```

サンプルアプリケーションのインストール

動作確認用サンプルとして、前回の ILB の設定でも触れた健全性検査用サーブレットアプリケーションと、クライアントからメッセージを受け取ってイントラネットに送信するメッセージングサーブレットアプリケーションをインストールします。

健全性検査用アプリケーションのインストール

健全性検査用アプリケーションのサンプルソースコードは以下のようになります。

```
package com.fujitsu.sparc.ilb;

import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class healthcheck extends HttpServlet {
  public void doGet (HttpServletRequest req, HttpServletResponse res)
    throws ServletException, IOException {
    PrintWriter out;

    res.setContentType("text/html; charset=UTF-8");
    out = res.getWriter();

    out.println("<html><body>");
    out.println("<h1>OK</h1>");
    out.println("</body></html>");
  }
}
```

web.xml は以下のようになります。

```
<?xml version="1.0" ?>
<!DOCTYPE web-app PUBLIC
  "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
  "http://java.sun.com/dtd/web-app_2_3.dtd">

<web-app>
  <servlet>
    <servlet-name>healthcheck</servlet-name>
    <servlet-class>com.fujitsu.sparc.ilb.healthcheck</servlet-class>
  </servlet>

  <servlet-mapping>
    <servlet-name>healthcheck</servlet-name>
    <url-pattern>/healthcheck</url-pattern>
  </servlet-mapping>
</web-app>
```

以下のディレクトリ構成で healthcheck.war を作成します。

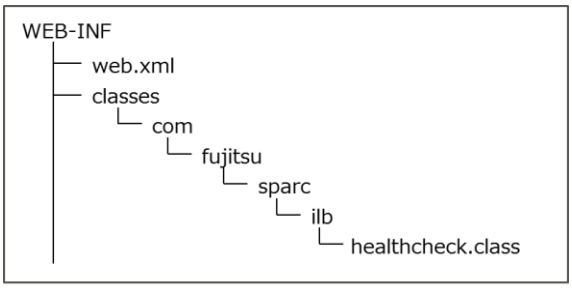

図6 healthcheck.war

healthcheck.war をデプロイします。

```
root@web:~# /opt/payara/bin/asadmin deploy healthcheck.war
Enter admin user name>  admin
Enter admin password for user "admin">
Application deployed with name healthcheck.
Command deploy executed successfully.
```

メッセージングアプリケーションのインストール

メッセージングアプリケーションのサンプルソースコードは以下のようになります。

```
package com.fujitsu.sparc.ilb;

import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.util.Properties;

import org.apache.kafka.clients.consumer.*;
import org.apache.kafka.clients.producer.*;
```

```
import org.apache.kafka.common.serialization.*;

public class MsgControl extends HttpServlet {

  public void doPost (HttpServletRequest req, HttpServletResponse res)
    throws ServletException, IOException {

    String      id, val, mbuf;
    PrintWriter out;

    id = req.getParameter("id");
    val = req.getParameter("value");
    mbuf = "{ " + "¥"ID¥"" + " : " + "¥"" + id + "¥"" + ", "
         + "¥"VALUE=¥"" + " : " + "¥"" + val + "¥"" + " }";

    Properties properties = new Properties();
    properties.put(ProducerConfig.BOOTSTRAP_SERVERS_CONFIG, "kafka1:9092, kafka2:9092, kafka3:9092");

    try (KafkaProducer<Integer, String> producer = new KafkaProducer<>(properties, new
 IntegerSerializer(), new StringSerializer())) {

      producer.send(new ProducerRecord<>("iot", 1, mbuf));
    }

    res.setContentType("text/html; charset=UTF-8");
    out = res.getWriter();

    out.println("<html><body>");
    out.println("<h1>"+id+"</h1>");
    out.println("<h1>"+val+"</h1>");
    out.println("</body></html>");
  }
}
```

web.xml は以下のようになります。

```
<?xml version="1.0" ?>
<!DOCTYPE web-app PUBLIC
  "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
  "http://java.sun.com/dtd/web-app_2_3.dtd">

<web-app>
  <servlet>
    <servlet-name>MsgControl</servlet-name>
    <servlet-class>com.fujitsu.sparc.ilb.MsgControl</servlet-class>
  </servlet>

  <servlet-mapping>
    <servlet-name>MsgControl</servlet-name>
    <url-pattern>/MsgControl</url-pattern>
  </servlet-mapping>
</web-app>
```

以下のディレクトリ構成で MsgControl.war を作成します。
WAR ファイルを作成する際に、以下の 3 つの Kafka 通信用のライブラリーを WEB-INF/lib に含めます。

```
/opt/kafka/libs/ kafka-clients-1.0.0.jar
/opt/kafka/libs/kafka_2.12-1.0.0.jar
/opt/kafka/libs/slf4j-api-1.7.25.jar
```

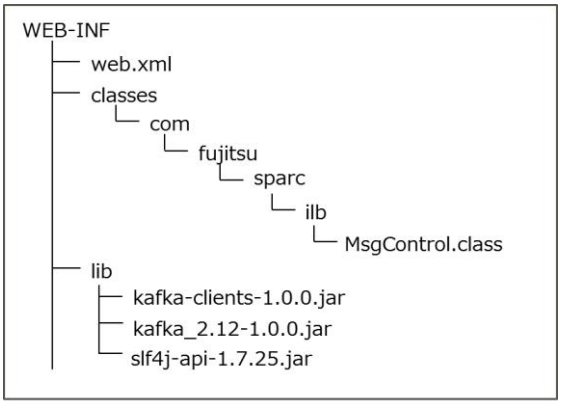

図7 MsgControl.war

MsgControl.war をデプロイします。

```
root@web:~# /opt/payara/bin/asadmin deploy MsgControl.war
Enter admin user name>  admin
Enter admin password for user "admin">
Application deployed with name MsgControl.
Command deploy executed successfully.
```

以上でアプリケーションサーバゾーンの設定は完了です。
次回は Kafka サーバの設定を行います。

【免責事項】

- 富士通(株) は、本コンテンツの内容について、妥当性や正確性について保証するものではなく、一切の責任を負い兼ねます。
- 本コンテンツや URL は、予告なしに変更または中止されることがあります。あらかじめご了承願います。
- 理由の如何に関わらず、情報の変更及び本コンテンツの掲載の中断または中止によって生じるいかなる損害についても責任を負うものではありません。

関連情報:http://www.fujitsu.com/jp/about/resources/terms/copyright/index.html