クラウド基盤構築演習

第一部
クラウド基盤を支えるインフラ技術
～ 第4回 Linuxネットワーク管理演習

ver1.1 2012/05/01

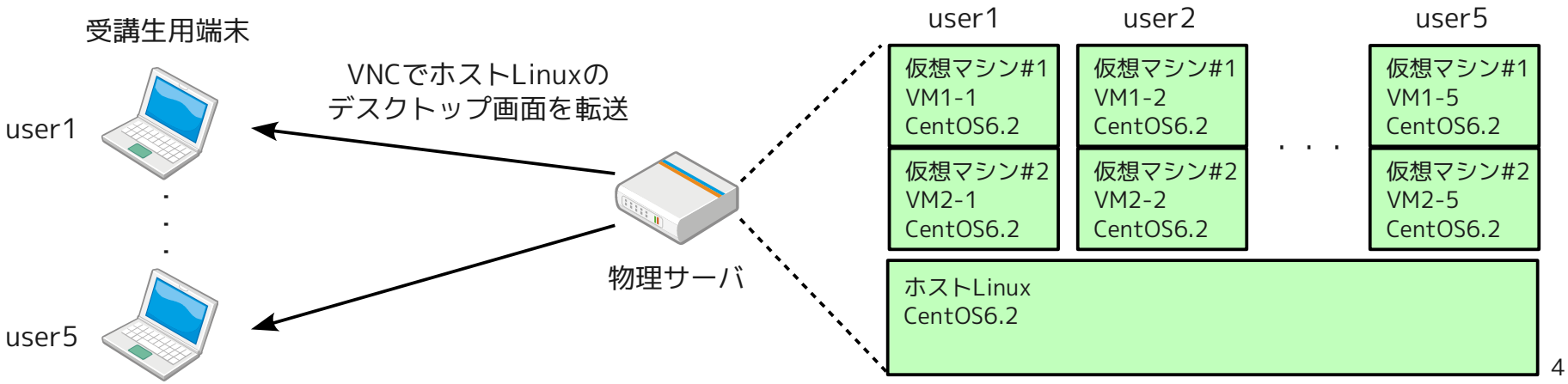
目次

- 演習環境の説明
- iptables設定演習
- SSH公開鍵認証演習
- VLANデバイス設定演習

演習環境の説明

演習環境 (1)

- 受講生（最大）5名ごとに演習用の物理サーバが割り当てられています。
  - 各受講生は、自分に割り当てられた「物理サーバ（IPアドレス）」と「ログインユーザ（user1～user5）」を確認してください。
- 各物理サーバには、ホストLinuxとして、CentOS6.2が導入されています。このホストLinuxのデスクトップ画面をVNCで受講生用端末に表示して演習を行います。
  - VNC接続の方法は、別途インストラクタよりガイドがあります。
- この演習では、Linux KVMによる仮想化環境を利用して、CentOS6.2をゲストOSとする仮想マシンを「受講生1名につき2台」作成します。
  - 各受講生は自分が作成する仮想マシンについて、「仮想マシン名、ホストネーム、IPアドレス」の割り当てルール（次ページ参照）を確認してください。



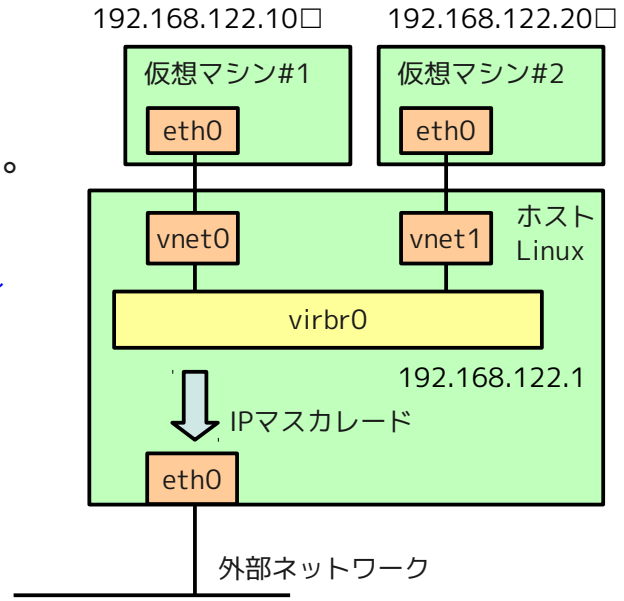
演習環境 (2)

- 演習で作成する仮想マシンは、ホストLinux上の仮想ブリッジによるプライベートネットワークに接続されます。
  - 仮想マシンから外部ネットワークには、IPマスカレードで接続します。外部ネットワークから仮想マシンに接続することはできません。
  - ホストLinuxから仮想マシンにログインすることは可能です。
- 仮想マシンを使用する際は、次のどちらかで接続します。
  - ホストLinuxで「virt-manager」を起動して、仮想マシンのコンソール画面を開く。
  - ホストLinuxから仮想マシンにSSHでログインする
  - 仮想マシン名、ホストネーム、IPアドレスは下表を使用します。
    □には、割り当てられたユーザ番号（1～5）が入ります。

※演習手順において、□で示された部分も同様にユーザ番号（1～5）を入れてください。

仮想マシン名	ホストネーム	IPアドレス /ネットマスク	デフォルトゲートウェイ
仮想マシン#1 VM1-□	vm1-□	192.168.122.10□ /255.255.255.0	192.168.122.1
仮想マシン#2 VM2-□	vm2-□	192.168.122.20□ /255.255.255.0	192.168.122.1

仮想ネットワークの構成



演習環境 (3)

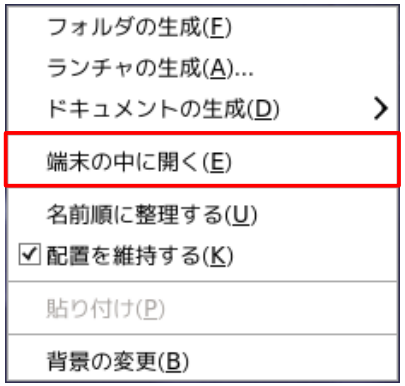
- ホストLinuxでコマンド端末を開くには、デスクトップを右クリックして「端末の中を開く」を選択します。

- 「virt-manager」を起動するには、コマンド端末で「virt-manager」を実行するか、デスクトップ左上の「アプリケーション」メニューから「システムツール→仮想マシンマネージャー」を選択します。

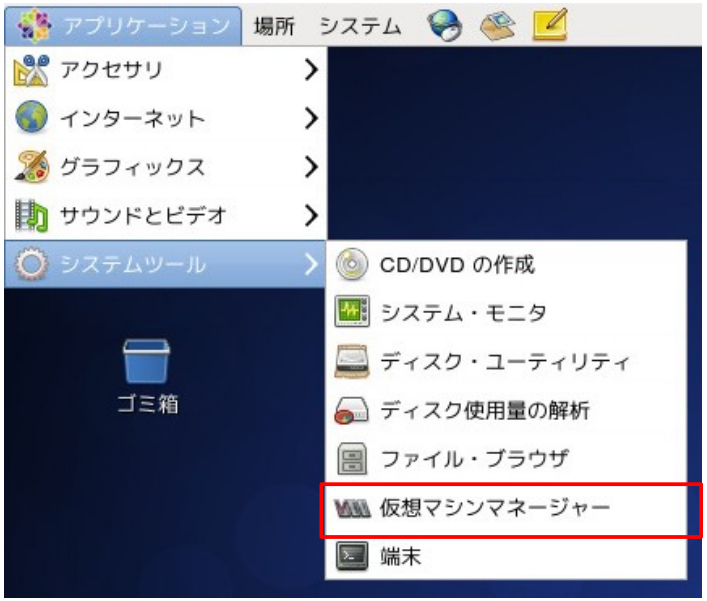
- 「Firefox」を起動するには、コマンド端末で「firefox」を実行するか、デスクトップ上部のアイコン（「システム」メニューの右横）をクリックします。

- ホストLinux上では、CentOS6.2のインストールメディアの内容がHTTPで公開されています。ホストLinuxのFirefoxから次のURLにアクセスして、内容を確認してください。
  - http://192.168.122.1/repo

デスクトップの
右クリックメニュー



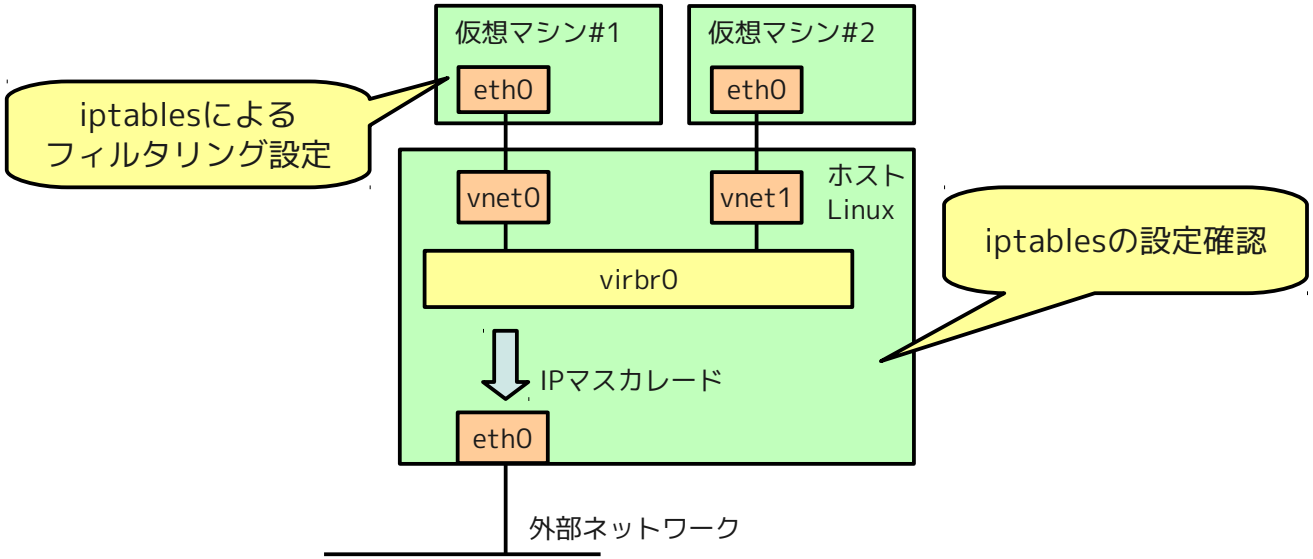
デスクトップのアプリケーションメニュー



iptables設定演習

演習内容

- この演習では、次の作業を行います。
  - ホストLinuxに設定されているiptablesについて次の内容を確認します。
    - 外部ネットワーク、および仮想マシンからホストLinuxに対するアクセス制限の設定。
    - 仮想マシンからホストLinuxを経由して、外部ネットワークにパケットを転送する設定。
  - 仮想マシン#1（VM1-□）において、iptablesによるフィルタリングの設定を行います。
    - はじめに、外部からのSSH接続のみを許可するように、iptablesコマンドで設定を行います。
    - つづいて、外部からのHTTP接続も許可するように、設定ファイルを編集して、設定を行います。
    - 最後にユーザ定義チェーンを定義して、iptablesが拒否したパケットをログ出力するように設定します。



事前準備 (1)

- 演習の事前準備として、VM1-□にHTTPサーバを構成します。
  - ホストLinuxのコマンド端末からVM1-□にログインします。

```
# ssh root@192.168.122.10□    ← ホストLinuxのコマンド端末で実行
```

  - httpdパッケージを導入します。

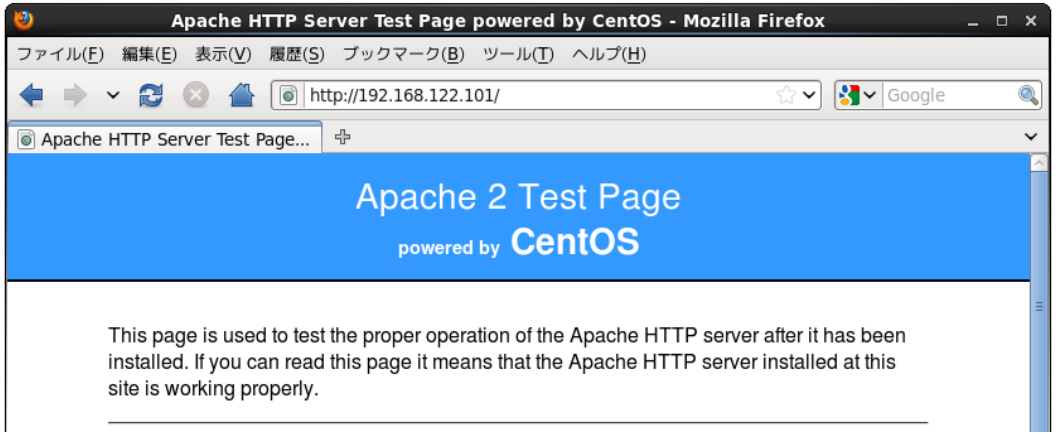
```
[root@vm1-□ ~]# yum install httpd
・・・
================================================================================
 Package             Arch         Version                       Repository  Size
================================================================================
Installing:
 httpd               x86_64       2.2.15-15.el6.centos          base       809 k
Installing for dependencies:
 apr                 x86_64       1.3.9-3.el6_1.2               base       123 k
 apr-util            x86_64       1.3.9-3.el6_0.1               base        87 k
 apr-util-ldap       x86_64       1.3.9-3.el6_0.1               base        15 k
 httpd-tools         x86_64       2.2.15-15.el6.centos          base        70 k

Transaction Summary
================================================================================
Install       5 Package(s)

Total download size: 1.1 M
Installed size: 3.5 M
Is this ok [y/N]: y
・・・
Complete!
```

事前準備 (2)

- httpdサービスを起動します。

```
[root@vm1-□ ~]# chkconfig httpd on
[root@vm1-□ ~]# service httpd start
httpd を起動中: httpd: apr_sockaddr_info_get() failed for vm01
httpd: Could not reliably determine the server's fully qualified domain name, using
127.0.0.1 for ServerName
                                                           [  OK  ]
```

- ホストLinuxでFirefoxを起動して「http://192.168.122.10□」に接続します。下記のテストページが表示されることを確認します。



- 以上で「事前準備」は完了です。

ホストLinuxのiptables設定確認 (1)

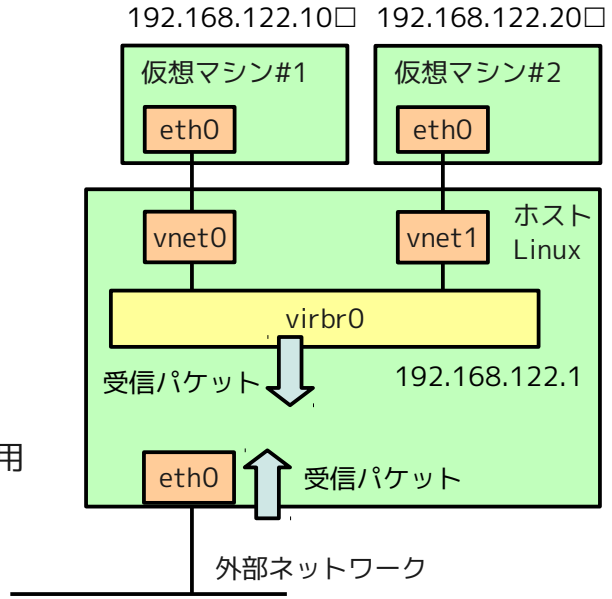
- ホストLinuxのiptablesの設定内容を確認します。
- 質問 1 : ホストLinuxでrootユーザから次のコマンドを実行します。その結果を見て、次の質問に答えてください。

```
# iptables -L INPUT -nv
```

  - 外部ネットワークからホストLinuxに対して、どのような接続が許可されていますか？
  - 仮想マシンからホストLinuxに対して、どのような接続が許可されていますか？

- ヒント:
  - 「virbr0」はホストLinux内部の仮想ブリッジ（仮想スイッチ）です。仮想マシンからのIPパケットは、この仮想ブリッジを通って、ホストLinuxに到達します。
  - TCP/UDP53番ポートは、DNSサーバとの通信に使用します。
  - TCP/UDP67番ポートは、DHCPサーバとの通信に使用します。
  - TCP22番ポートは、SSH接続に使用します。
  - TCP80番ポートは、HTTP接続に使用します。
  - TCP5001～5999番ポートは、VNC接続に使用します。
  - TCP4567番ポートは、本環境に固有の通信（クラウド管理）に使用します。
- 解説は次ページから記載しています。

仮想ネットワークの構成



ホストLinuxのiptables設定確認 (2)

- 質問１の解説

```
# iptables -L INPUT -nv
Chain INPUT (policy DROP 6 packets, 451 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     udp  --  virbr0 *       0.0.0.0/0            0.0.0.0/0           udp dpt:53
    0     0 ACCEPT     tcp  --  virbr0 *       0.0.0.0/0            0.0.0.0/0           tcp dpt:53
    0     0 ACCEPT     udp  --  virbr0 *       0.0.0.0/0            0.0.0.0/0           udp dpt:67
    0     0 ACCEPT     tcp  --  virbr0 *       0.0.0.0/0            0.0.0.0/0           tcp dpt:67
  957 1081K ACCEPT     all  --  *      *       0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
    0     0 ACCEPT     icmp --  *      *       0.0.0.0/0            0.0.0.0/0
   10   600 ACCEPT     all  --  lo     *       0.0.0.0/0            0.0.0.0/0
    2   176 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:22
   12   720 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:80
    0     0 ACCEPT     tcp  --  eth0   *       0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:4567
    0     0 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0           state NEW tcp dpts:5901:5999
    5   391 LOG        all  --  *      *       0.0.0.0/0            0.0.0.0/0           limit: avg 3/hour burst 5 LOG
flags 0 level 4 prefix `[INPUT Dropped] '
```

- INPUTチェーンは、該当サーバ宛のパケットに対するフィルタリング処理を行います。
  - デフォルトポリシーがDROPに設定されているので、明示的に許可されたパケット以外は受信を拒否します。
- 1〜4は、入力インターフェース（in）がvirbr0なので仮想マシンからのパケットに対応します。
  - 1, 2は、ホストLinux上のDNSサーバに送られる宛先ポート53のTCP/UDPパケットの受信を許可します。
  - 3, 4は、ホストLinux上のDHCPサーバに送られる宛先ポート67のTCP/UDPパケットの受信を許可します。
- ５は、確立済みのセッションにおけるパケットの受信を許可します。
  - 8〜11の新規セッションを確立するためのパケットを許可する設定と対になって機能します。
- 6は、任意のICMPパケットの受信を許可します。

ホストLinuxのiptables設定確認 (3)

- 7は、ループバックインターフェース（127.0.0.1）宛の任意のパケットの受信を許可します。
  - ループバックインターフェースは、ホストLinuxから自身にアクセスするためのインターフェースです。
- 8～11は、新規のTCPセッションを確立するためのパケットの受信を許可します。
  - 8は、SSH接続のTCPパケットの受信を許可します。
  - 9は、HTTP接続のTCPパケットの受信を許可します。
  - 10は、本環境に固有のクラウド管理用TCPパケットの受信を許可します。
  - 11は、VNC接続のTCPパケットの受信を許可します。
- 12は、その他のパケットのログをシステムログに記録します。
  - 受信を拒否したパケットのログを記録することは、セキュリティ問題のチェックに役立ちます。

■ 以上をまとめると質問1の回答は以下になります。

- 仮想マシンから、ホストLinux上のDNSサーバ、DHCPサーバ機能へのアクセスが許可されます。
- 仮想マシン、および外部ネットワークから、ホストLinuxへのVNC接続、SSH接続、HTTP接続、ICMP通信、および本環境に固有のクラウド管理通信が許可されます。
- ループバックインターフェースへの任意の通信が許可されます。
- それ以外の接続は許可されません。

ホストLinuxのiptables設定確認 (4)

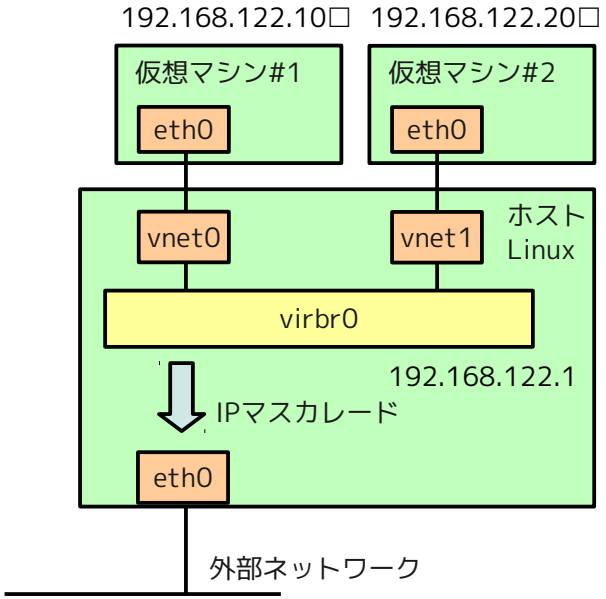
- 質問２：ホストLinuxで次のコマンドを実行して、その結果を見て次の質問に答えてください。

```
# iptables -L FORWARD -nv
```

  - 仮想マシン同士について、どのような通信が許可されていますか？
  - 仮想マシンから外部ネットワークに対して、どのような通信が許可されていますか？
  - 外部ネットワークから仮想マシンに対して、どのような通信が許可されていますか？

- ヒント:
  - 「virbr0」はホストLinux内部の仮想ブリッジ（仮想スイッチ）です。仮想マシンからのIPパケットは、この仮想ブリッジを通ってホストLinuxに到達した後、外部ネットワークに転送されるか、もしくは、再度、仮想ブリッジに戻って、他の仮想マシンに転送されます。
- 回答は次ページからです。

仮想ネットワークの構成



ホストLinuxのiptables設定確認 (5)

- 質問２の回答

```
# iptables -L FORWARD -nv
Chain FORWARD (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
1    0     0 ACCEPT     all  --  *      virbr0  0.0.0.0/0            192.168.122.0/24    state RELATED,ESTABLISHED
2    0     0 ACCEPT     all  --  virbr0 *       192.168.122.0/24     0.0.0.0/0
3    0     0 ACCEPT     all  --  virbr0 virbr0  0.0.0.0/0            0.0.0.0/0
4    0     0 REJECT     all  --  *      virbr0  0.0.0.0/0            0.0.0.0/0           reject-with icmp-port-unreachable
5    0     0 REJECT     all  --  virbr0 *       0.0.0.0/0            0.0.0.0/0           reject-with icmp-port-unreachable
```

- FORWARDチェーンは、該当サーバが中継するパケットに対するフィルタリング処理を行います。
  - デフォルトポリシーがDROPに設定されているので、明示的に許可されたパケット以外は受信を拒否します。
- 2,3,5は、入力インターフェース（in）がvirbr0なので、仮想マシンから他のネットワークに向けたパケットに対応します。
  - 3は、仮想ブリッジvirbr0に接続した仮想マシン間の通信を許可します。
  - 2は、仮想マシンから外部ネットワークへの通信を許可します。
  - 5は、仮想マシンからのその他の通信を拒否します。
- 1,4は、出力インターフェース（out）がvirbr0なので、仮想マシンに向けたパケットに対応します。
  - 1は、確立済みのセッションに対する通信を許可します。2,3で確立したセッションに対する返答パケットが該当します。
  - ４は、その他の仮想マシンに向けたパケットを拒否します。つまり、2,3以外の通信（外部ネットワークから仮想マシンへの接続など）は許可されないことになります。

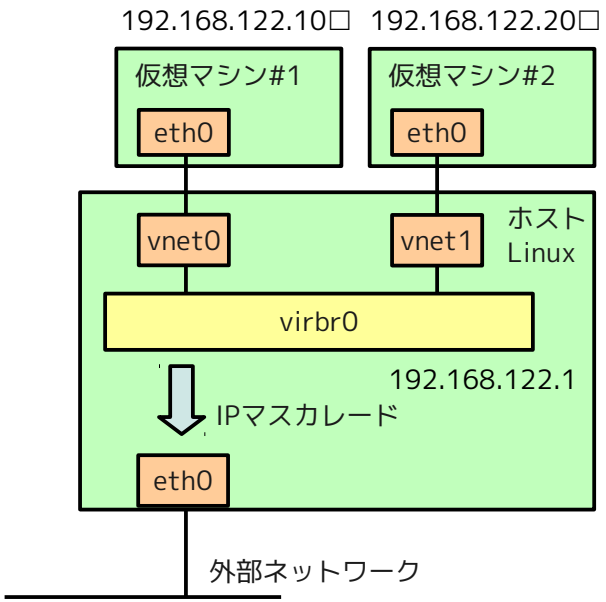
ホストLinuxのiptables設定確認 (6)

- 以上をまとめると質問2の回答は以下になります。
  - 仮想ブリッジvirbr0に接続された仮想マシン同士は任意の通信が許可されます。
  - 仮想マシンから外部ネットワークへの通信が許可されます。
  - それ以外の通信は許可されません。

ホストLinuxのiptables設定確認 (7)

- 質問 3 : 仮想マシンから外部ネットワークへの通信では、マスカレード処理が行われます。これは、iptablesのどのような設定から確認できるでしょうか？
  - 回答は次ページからです。

仮想ネットワークの構成



ホストLinuxのiptables設定確認 (8)

- 質問３の回答
  - マスカレードの処理は、POSTROUTINGチェーンのnatテーブルで設定されます。

```
# iptables -t nat -L POSTROUTING -nv
Chain POSTROUTING (policy ACCEPT 108 packets, 7529 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 MASQUERADE  tcp  --  *      *       192.168.122.0/24    !192.168.122.0/24    masq ports: 1024-65535
    8   544 MASQUERADE  udp  --  *      *       192.168.122.0/24    !192.168.122.0/24    masq ports: 1024-65535
    1    40 MASQUERADE  all  --  *      *       192.168.122.0/24    !192.168.122.0/24
```

  - これらはすべて、送信元が「192.168.122.0/24」（仮想ブリッジvirbr0に接続した仮想マシン）で宛先が「192.168.122.0/24」以外（仮想ブリッジvirbr0に接続した仮想マシン以外）の通信に対するマスカレード処理になります。
  - 各行は、それぞれTCPパケット、UDPパケット、それ以外のパケット（ICMPパケット）に対する設定です。
  - 以上で「ホストLinuxのiptables設定確認」は完了です。

iptablesコマンドによる設定 (1)

- 仮想マシン#1（VM1-□）において、外部からのSSH接続、および外部のDNSサーバからの応答のみを受け付けるようにiptablesによるフィルタリング設定を行います。ここでは、iptablesコマンドによる設定を行います。
  - ホストLinuxのコマンド端末からVM1-□にログインします。

```
# ssh root@192.168.122.10□    ← ホストLinuxのコマンド端末で実行
```

  - 現在のiptablesの設定内容を表示して、何も設定されていないことを確認します。

```
[root@vm1-□ ~]# iptables -L -n
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

    - これらは「INPUT」「FORWARD」「OUTPUT」の各チェーンの「filter」ターゲットの内容です。

```
[root@vm1-□ ~]# iptables -t nat -L -n
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination

Chain POSTROUTING (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

    - これらは「PREROUTING」「POSTROUTING」「OUTPUT」の各チェーンの「nat」ターゲットの内容です。

iptablesコマンドによる設定 (2)

- INPUTチェーンのfilterテーブルにSSH接続（TCP22番ポート宛のパケット）の受信を許可する設定を追加します。

```
[root@vm1-□ ~]# iptables -A INPUT -p tcp --dport 22 -j ACCEPT
```

  - -tオプションでテーブルを指定しない場合は、デフォルトでfilterテーブルが選択されます。

- 同じくDNSサーバからの応答パケット（送信元がUDP/TCP53番ポート）の受信を許可する設定を追加します。

```
[root@vm1-□ ~]# iptables -A INPUT -p tcp --sport 53 -j ACCEPT
[root@vm1-□ ~]# iptables -A INPUT -p udp --sport 53 -j ACCEPT
```

  （次に設定するデフォルトアクションで破棄される）その他のパケットをシステムログに記録する設定を追加します。

```
[root@vm1-□ ~]# iptables -A INPUT -j LOG
```

- INPUTチェーンのデフォルトアクションを「DROP」に指定して、その他のパケットの受信を拒否します。

```
[root@vm1-□ ~]# iptables -P INPUT DROP
```

  - 「REJECT」アクションの場合は、パケットを拒否したことを通知するICMPパケットを返送しますが、「DROP」アクションでは、該当のパケットを黙って破棄します。

iptablesコマンドによる設定 (3)

- ここまでの設定内容を確認します。

```
[root@vm1-□ ~]# iptables -L INPUT -n
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:22
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp spt:53
ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           udp spt:53
LOG        all  --  0.0.0.0/0            0.0.0.0/0           LOG flags 0 level 4

[root@vm1-□ ~]# iptables -L INPUT
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:ssh
ACCEPT     tcp  --  anywhere             anywhere            tcp spt:domain
ACCEPT     udp  --  anywhere             anywhere            udp spt:domain
LOG        all  --  anywhere             anywhere            LOG level warning
```

- 「-n」オプションを省略すると、IPアドレスやポート番号が文字表記に変わります。

iptablesコマンドによる設定 (4)

- 現在の設定内容を設定ファイル「/etc/sysconfig/iptables」に反映して、サーバ起動時に再設定されるように、iptablesサービスの自動起動を有効にします。

```
[root@vm1-□ ~]# service iptables save
iptables: ファイアウォールのルールを /etc/sysconfig/iptable[  OK  ]中:
[root@vm1-□ ~]# chkconfig iptables on
```

- 設定ファイルに記録された内容を確認します。

natテーブルの設定

filterテーブルの設定

```
[root@vm1-□ ~]# cat /etc/sysconfig/iptables
# Generated by iptables-save v1.4.7 on Thu Jan 19 12:52:21 2012
*nat
:PREROUTING ACCEPT [0:0]
:POSTROUTING ACCEPT [1:420]
:OUTPUT ACCEPT [1:420]
COMMIT
# Completed on Thu Jan 19 12:52:21 2012
# Generated by iptables-save v1.4.7 on Thu Jan 19 12:52:21 2012
*filter
:INPUT DROP [1:32]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [109:13172]
-A INPUT -p tcp -m tcp --dport 22 -j ACCEPT
-A INPUT -p tcp -m tcp --sport 53 -j ACCEPT
-A INPUT -p udp -m udp --sport 53 -j ACCEPT
-A INPUT -j LOG
COMMIT
# Completed on Thu Jan 19 12:52:21 2012
```

iptablesコマンドによる設定 (5)

- VM1-□システムログの出力を確認しながら、ホストLinuxからpingを送ります。

```
[root@vm1-□ ~]# tail -f /var/log/messages
```

```
# ping 192.168.122.10□                                                    ← ホストLinuxで実行
PING 192.168.122.10□ (192.168.122.10□) 56(84) bytes of data.
^C                                                                        ← Ctrl+Cで停止
--- 192.168.122.10□ ping statistics ---
8 packets transmitted, 0 received, 100% packet loss, time 7506ms
```

  - iptablesによりICMPパケットの受信が拒否されるので、pingに対する応答はありません。

- VM1-□のシステムログにICMPパケットの受信ログが記録されることを確認します。

```
Jan 19 11:42:38 vm01 kernel: IN=eth0 OUT= MAC=52:54:00:47:02:46:52:54:00:18:89:ef:08:00
SRC=192.168.122.1 DST=192.168.122.10□ LEN=84 TOS=0x00 PREC=0x00 TTL=64 ID=0 DF PROTO=ICMP
TYPE=8 CODE=0 ID=65034 SEQ=1
Jan 19 11:42:39 vm01 kernel: IN=eth0 OUT= MAC=52:54:00:47:02:46:52:54:00:18:89:ef:08:00
SRC=192.168.122.1 DST=192.168.122.10□ LEN=84 TOS=0x00 PREC=0x00 TTL=64 ID=0 DF PROTO=ICMP
TYPE=8 CODE=0 ID=65034 SEQ=2
Jan 19 11:42:40 vm01 kernel: IN=eth0 OUT= MAC=52:54:00:47:02:46:52:54:00:18:89:ef:08:00
SRC=192.168.122.1 DST=192.168.122.10□ LEN=84 TOS=0x00 PREC=0x00 TTL=64 ID=0 DF PROTO=ICMP
TYPE=8 CODE=0 ID=65034 SEQ=3
```

- 以上で「iptablesコマンドによる設定」は完了です。

設定ファイル編集によるiptables設定 (1)

- 先の演習に続いて、HTTP接続を受け付けるようにiptablesによるフィルタリング設定を行います。ここでは、設定ファイルの編集による設定を行います。
  - ホストLinuxのFirefoxから「http://192.168.122.10□」にアクセスして、接続できないことを確認します。

- これは、先の演習で設定したフィルタリングでは、HTTP接続が許可されていないためです。

- これらは「PREROUTING」「POSTROUTING」「OUTPUT」の各チェーンの「nat」ターゲットの内容です。

設定ファイル編集によるiptables設定 (2)

- 設定ファイル「/etc/sysconfig/iptables」にHTTP接続を許可する設定を追加します。

/etc/sysconfig/iptables

```
# Generated by iptables-save v1.4.7 on Thu Jan 19 12:52:21 2012
*nat
:PREROUTING ACCEPT [0:0]
:POSTROUTING ACCEPT [1:420]
:OUTPUT ACCEPT [1:420]
COMMIT
# Completed on Thu Jan 19 12:52:21 2012
# Generated by iptables-save v1.4.7 on Thu Jan 19 12:52:21 2012
*filter
:INPUT DROP [1:32]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [109:13172]
-A INPUT -p tcp -m tcp --dport 22 -j ACCEPT
-A INPUT -p tcp -m tcp --dport 80 -j ACCEPT
-A INPUT -p tcp -m tcp --sport 53 -j ACCEPT
-A INPUT -p udp -m udp --sport 53 -j ACCEPT
-A INPUT -j LOG
COMMIT
# Completed on Thu Jan 19 12:52:21 2012
```

- 設定変更を反映するためにiptablesサービスを再起動します。

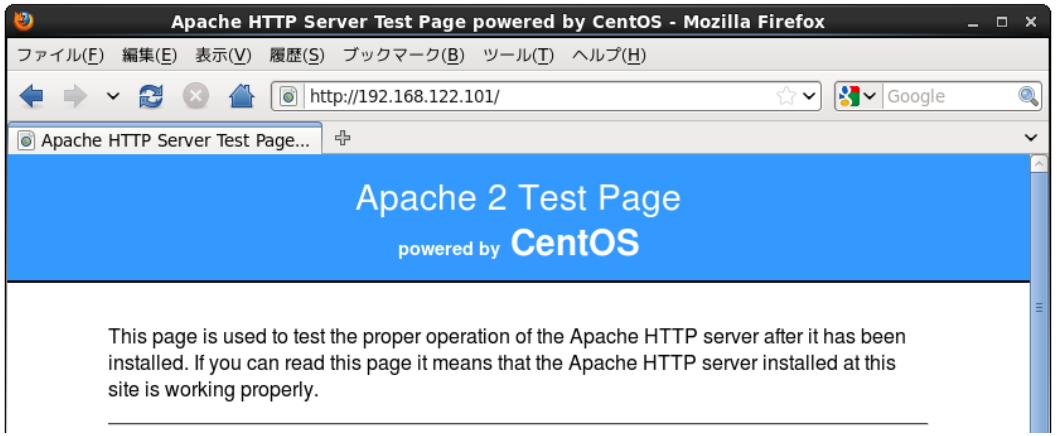
```
[root@vm1-□ ~]# service iptables restart
iptables: ファイアウォールルールを消去中:                    [  OK  ]
iptables: チェインをポリシー ACCEPT へ設定中filter nat     [  OK  ]
iptables: モジュールを取り外し中:                           [  OK  ]
iptables: ファイアウォールルールを適用中:                   [  OK  ]
```

設定ファイル編集によるiptables設定 (3)

- 設定変更が反映されたことを確認します。

```
[root@vm1-□ ~]# iptables -L -n
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:22
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:80
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp spt:53
ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           udp spt:53
LOG        all  --  0.0.0.0/0            0.0.0.0/0           LOG flags 0 level 4

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

- ホストLinuxのFirefoxから、再度、「http://192.168.122.10□」にアクセスして、今回は接続に成功することを確認します。



設定ファイル編集によるiptables設定 (4)

- 最後にiptablesサービスを停止して、すべてのフィルタリングを解除します。

```
[root@vm1-□ ~]# service iptables stop
iptables: ファイアウォールルールを消去中:                   [  OK  ]
iptables: チェインをポリシー ACCEPT へ設定中filter nat     [  OK  ]
iptables: モジュールを取り外し中:                           [  OK  ]
[root@vm1-□ ~]# chkconfig iptables off
```

- 以上で「設定ファイル編集によるiptables設定」は完了です。

- 以上で「iptables設定演習」は完了です。

メモとしてお使いください

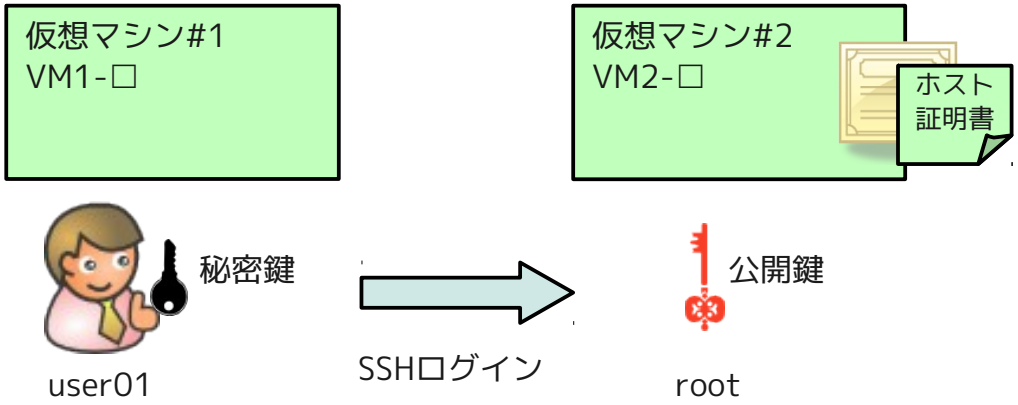
メモとしてお使いください

メモとしてお使いください

SSH公開鍵認証演習

演習内容

- この演習では、次の作業を行います。
  - 仮想マシン#1（VM1-□）と仮想マシン#2（VM2-□）の間で、公開鍵認証によるSSH接続の設定を行います。
    - 仮想マシン#1（VM1-□）のユーザ「user01」から、仮想マシン#2のrootユーザに公開鍵認証によるSSH接続ができるように設定します。
    - 仮想マシン#2（VM2-□）のホスト証明書を再作成して、公開鍵認証に与える影響を確認します。



SSH公開鍵認証の設定 (1)

- 仮想マシン#1（VM1-□）のユーザ「user01」から、仮想マシン#2のrootユーザに公開鍵認証によるSSH接続ができるように設定します。
  - ホストLinuxのコマンド端末からVM1-□にログインします。

```
# ssh root@192.168.122.10□    ← ホストLinuxのコマンド端末で実行
```

  - ユーザ「user01」を作成して、suコマンドでユーザを切り替えます。

```
[root@vm1-□ ~]# adduser user01
[root@vm1-□ ~]# su - user01
[user01@vm1-□ ~]$
```

  - user01の鍵ペアをパスフレーズなしで作成します。

```
[user01@vm1-□ ~]$ ssh-keygen -P "" -f ~/.ssh/mykey.rsa
Generating public/private rsa key pair.
Created directory '/home/user01/.ssh'.
Your identification has been saved in /home/user01/.ssh/mykey.rsa.
Your public key has been saved in /home/user01/.ssh/mykey.rsa.pub.
The key fingerprint is:
db:f7:b7:06:e3:d3:49:9c:34:77:c0:a3:7d:d2:3a:eb user01@vm1-□
The key's randomart image is:
+--[ RSA 2048]----+
|              .   |
|               +  |
|              o + |
|             . oo=|
|        S     o+=|
|         o   oo+ |
|        . . o =o.|
|            . +.+.|
|              .Eo.|
+-----------------+
```

SSH公開鍵認証の設定 (2)

- 作成された鍵ペアのファイルを確認します。

```
[user01@vm1-□ ~]$ ls -l ~/.ssh
合計 8
-rw-------. 1 user01 user01 1675  1月 19 14:20 2012 mykey.rsa
-rw-r--r--. 1 user01 user01  393  1月 19 14:20 2012 mykey.rsa.pub
```

  - 秘密鍵「mykey.rsa」と公開鍵「mykey.rsa.pub」でファイルのアクセス権が異なる点に注意してください。
    秘密鍵は所有者本人以外は読み取りできないようになっています。

- 鍵ファイルの内容を表示してみます。

```
[user01@vm1-□ ~]$ cat ~/.ssh/mykey.rsa
-----BEGIN RSA PRIVATE KEY-----
MIIEoQIBAAKCAQEApS++obe1U5MI5syxDPHm/cBJ8sEx1wAALvAzlgWh0rCTV9Hu
. . .
l5PLH40oVFCXEs607htNfEdTMZAtqTTefZpUz5fqdbO7Zz1U+MeQ+TOps4RkI70u
JVMlp8j/q757m4pNwWxG9aDWfKfV4t0YfVrCUvxo90Qg2iLZqg==
-----END RSA PRIVATE KEY-----

[user01@vm1-□ ~]$ cat ~/.ssh/mykey.rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAQEApS++obe1U5MI5syxDPHm/cBJ8sEx1wAALvAzlgWh0rCTV9Hur6MYoxNy
wQdT493sMlLbc8szD5bnRjCZWql0PHeJm/pVykJLBx1mEGGS+a73Mi5zlTeMGPToerWSTqqCwsLjV+AReSm5vF+ahe5F
/Zllbu0bjpn89eyrYD/uP6pwilUTHdQI0Fi+iDZ/2mmZ1+3mn5siTmXgNsZ/4A1h3/me/1xs86t6Vajc6tST1RbXMVON
GMJKvo1H2CNJcm7ceSixcfYt8dZZy4AQgLomG0/XzTgYUhcaw0fEwjxD5h9tA0d0Nh/xM8M/I8gnqkZ7yRsoEfla9t61
vs3CMAsPaw== user01@vm1-□
```

SSH公開鍵認証の設定 (3)

- 作成した公開鍵ファイルをVM2-□のrootユーザに対して登録します。

```
[user01@vm1-□ ~]$ ssh-copy-id -i ~/.ssh/mykey.rsa.pub root@192.168.122.20□
The authenticity of host '192.168.122.20□ (192.168.122.20□)' can't be established.
RSA key fingerprint is 4a:39:f5:5e:57:5e:f4:9f:89:25:59:fc:39:d0:8f:cf.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.122.20□' (RSA) to the list of known hosts.
root@192.168.122.20□'s password:                    ← VM2-□のrootパスワード（edubase）を入力
Now try logging into the machine, with "ssh 'root@192.168.122.20□'", and check in:

  .ssh/authorized_keys

to make sure we haven't added extra keys that you weren't expecting.
```

  - 「yes」で返答している部分は、VM2-□のホスト証明書の受け入れの確認です。

- 受け入れたホスト証明書の内容を表示してみます。

```
[user01@vm1-□ ~]$ cat ~/.ssh/known_hosts
192.168.122.20□ ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAQEAwDZsHjhbmD1rjdCQ/UOZt1n9LMc8zfeSpkKKQBfX
EVkM+zPwmxGihCPUiCURj7OuVdMzehRWMFX2q/wX/GNC2kUBvfsqhtM1Pz1Xth5crt27MvvRzhUcT5EpRv0xGCtWTdX1
9PiQA7IRKjyYfN7G+xotXhHGzJQGTtEL42e/9QYnqlyohvQhQTAv8ZmRTxlVFYQSbHJQJLKNP/MM2O8tjEkEntH2WnUr
sWkAn5YMc8wVqV6C+hUbMVWkL9vUyJdn29Hbc3xnErAz9KfwJZYCdc7iFswgaMr/4BYg3iLV1qTOM5iyrg3YwbQimpba
i+8GpQxro5sTDKiplLYMaOi0Tw==
```

  - ホスト証明書は、対象サーバのIPアドレスに紐づけられていることが分かります。

- 秘密鍵を指定して、VM2-□にrootユーザとしてSSHログインします。

```
[user01@vm1-□ ~]$ ssh -i ~/.ssh/mykey.rsa root@192.168.122.20□
Last login: Thu Jan 19 13:01:05 2012 from 192.168.122.1
[root@vm2-□ ~]#
```

  - パスワード入力なしにログインに成功します。

SSH公開鍵認証の設定 (4)

- VM2-□のrootユーザに登録されている公開鍵を確認します。

```
[root@vm2-□ ~]# cat ~/.ssh/authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAQEApS++obe1U5MI5syxDPHm/cBJ8sEx1wAALvAzlgWh0rCTV9Hur6MYoxNy
wQdT493sMlLbc8szD5bnRjCZWql0PHeJm/pVykJLBx1mEGGS+a73Mi5zlTeMGPToerWSTqqCwsLjV+AReSm5vF+ahe5F
/Zllbu0bjpn89eyrYD/uP6pwilUTHdQI0Fi+iDZ/2mmZ1+3mn5siTmXgNsZ/4A1h3/me/1xs86t6Vajc6tST1RbXMVON
GMJKvo1H2CNJcm7ceSixcfYt8dZZy4AQgLomG0/XzTgYUhcaw0fEwjxD5h9tA0d0Nh/xM8M/I8gnqkZ7yRsoEfla9t61
vs3CMAsPaw== user01@vm1-□
```

  - 先に確認した公開鍵（mykey.rsa.pub）と同じ内容であることが分かります。

- VM2-□からログアウトします。

```
[root@vm2-□ ~]# exit
logout
Connection to 192.168.122.20□ closed.
```

- SSHでリモートコマンドが実行できることを確認します。

```
[user01@vm1-□ ~]$ ssh -i ~/.ssh/mykey.rsa root@192.168.122.20□ "date;hostname"
2012年  1月 19日 木曜日 14:50:18 JST
vm2-□
```

- 以上で「SSH公開鍵認証の設定」は完了です。

ホスト証明書変更の影響確認 (1)

- 仮想マシン#2（VM2-□）のホスト証明書を再作成して、公開鍵認証に与える影響を確認します。
  - ホストLinuxのコマンド端末からVM2-□にログインします。

```
# ssh root@192.168.122.20□    ← ホストLinuxのコマンド端末で実行
```

  - 既存のホスト証明書を削除します。

```
[root@vm2-□ ~]# cd /etc/ssh/
[root@vm2-□ ssh]# rm -f ssh_host_*
```

  - sshdサービスを再起動して、ホスト証明書を再作成します。

```
[root@vm2-□ ssh]# service sshd restart
sshd を停止中:                                             [  OK  ]
SSH1 RSA ホストキーを生成中:                               [  OK  ]
SSH2 RSA ホストキーを生成中:                               [  OK  ]
SSH2 DSAホストキーを生成中:                                [  OK  ]
sshd を起動中:                                             [  OK  ]
```

  - VM2-□から、一旦ログアウトします。

```
[root@vm2-□ ~]# exit
```

ホスト証明書変更の影響確認 (2)

- ホストLinuxのコマンド端末からVM1-□にログインします。

```
# ssh root@192.168.122.20□   ← ホストLinuxのコマンド端末で実行
```

- user01にユーザを切り替えます。

```
[root@vm1-□ ~]# su - user01
[user01@vm1-□ ~]$
```

- 先と同様に、VM2-□にrootユーザとしてSSHログインを行います。

```
[user01@vm1-□ ~]$ ssh -i ~/.ssh/mykey.rsa root@192.168.122.20□
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@    WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!     @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle attack)!
It is also possible that the RSA host key has just been changed.
The fingerprint for the RSA key sent by the remote host is
79:fd:48:c3:d9:eb:14:54:0f:e4:f2:03:56:4b:ee:8e.
Please contact your system administrator.
Add correct host key in /home/user01/.ssh/known_hosts to get rid of this message.
Offending key in /home/user01/.ssh/known_hosts:1
RSA host key for 192.168.122.20□ has changed and you have requested strict checking.
Host key verification failed.
```

- ホスト証明書が変更されているためログインに失敗します。

ホスト証明書変更の影響確認 (3)

- 「~/.ssh/known_hosts」に登録されているホスト証明書を削除します。

```
[user01@vm1-□ ~]$ rm ~/.ssh/known_hosts
```

  - 今回は、VM2-□のホスト証明書のみが登録されているので、証明書の登録ファイル全体を削除しています。
    複数のホスト証明書が登録されている場合は、該当するホスト証明書の行のみを削除します。

- 再度、VM2-□にrootユーザとしてSSHログインを行います。

```
[user01@vm1-□ ~]$ ssh -i ~/.ssh/mykey.rsa root@192.168.122.20□The authenticity of host
'192.168.122.20□ (192.168.122.20□)' can't be established.
RSA key fingerprint is 79:fd:48:c3:d9:eb:14:54:0f:e4:f2:03:56:4b:ee:8e.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.122.20□' (RSA) to the list of known hosts.
Last login: Thu Jan 19 15:30:32 2012 from 192.168.122.10□
[root@vm2-□ ~]#
```

  - 新しいホスト証明書の受け入れ確認が表示されるので、「yes」を入力して受け入れます。

- 以上で「ホスト証明書変更の影響確認」は完了です。

■ 以上で「SSH公開鍵認証演習」は完了です。

メモとしてお使いください

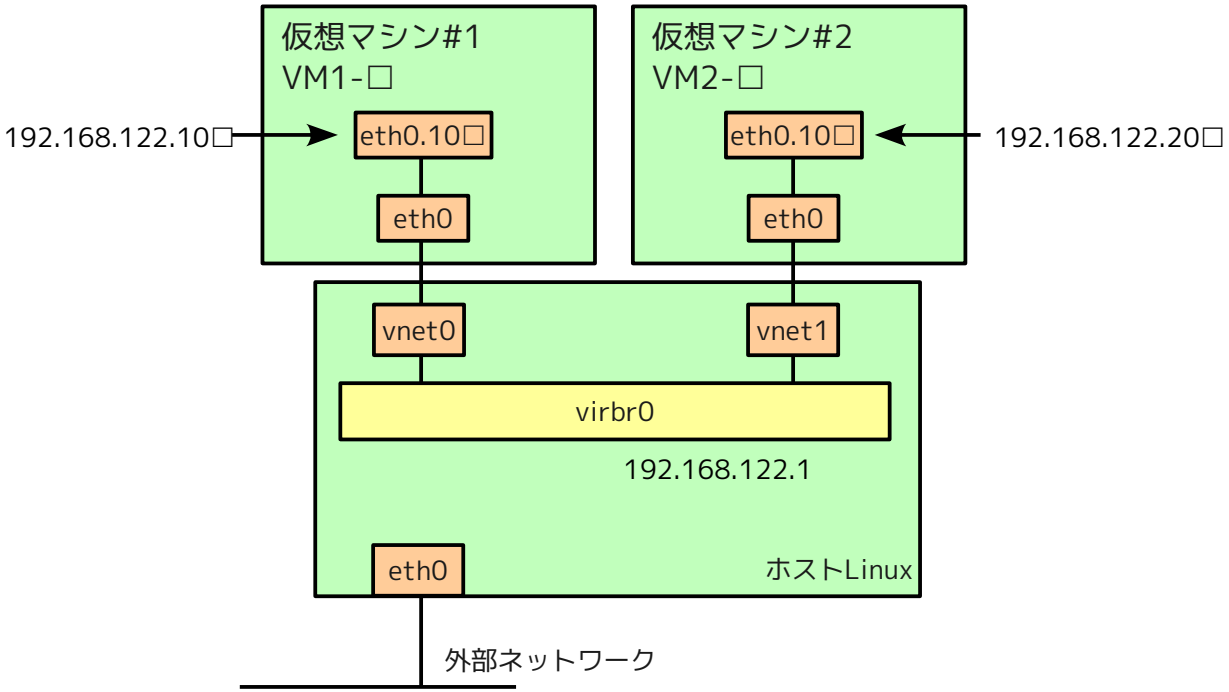
メモとしてお使いください

メモとしてお使いください

VLANデバイス設定演習

演習内容

- この演習では、次の作業を行います。
  - 仮想マシン#1（VM1-□）と仮想マシン#2（VM2-□）にVLANデバイスを構成します。
    - 同じVLAN IDのデバイスを構成した場合に、相互に通信できることを確認します。
    - 異なるVLAN IDのデバイスを構成した場合は、通信ができないことを確認します。



VLANデバイスの構成 (1)

- 仮想マシン#1（VM1-□）と仮想マシン#2（VM2-□）にVLAN ID 10□のVLANデバイスを構成します。
  - VLANデバイスの構成中にネットワーク接続を切断する必要があるため、virt-managerの仮想コンソール上で作業を行います。ホストLinuxでvirt-managerを起動します。

```
# virt-manager
```

  - 「VM1-□」をダブルクリックして仮想コンソールを開き、rootユーザでログインします。
  - 既存のネットワーク設定ファイルをコピーしてバックアップします。

```
[root@vm1-□ ~]# cp /etc/sysconfig/network-scripts/ifcfg-eth0 /root/work/
```

  - 設定ファイル「/etc/sysconfig/network」「/etc/sysconfig/network-scripts/ifcfg-eth0」「/etc/sysconfig/network-scripts/ifcfg-eth0.10□」を次のように修正／作成します。

/etc/sysconfig/network

```
NETWORKING=yes
HOSTNAME=vm1-□
VLAN=yes        ← この行を追加
```

/etc/sysconfig/network-scripts/ifcfg-eth0

```
DEVICE=eth0
BOOTPROTO=none
NM_CONTROLLED=no
ONBOOT=yes
```

/etc/sysconfig/network-scripts/ifcfg-eth0.10□

```
DEVICE=eth0.10□
BOOTPROTO=static
NM_CONTROLLED=no
ONBOOT=yes
IPADDR=192.168.122.10□
NETMASK=255.255.255.0
GATEWAY=192.168.122.1
```

VLANデバイスの構成 (2)

- networkサービスを再起動して、設定変更を反映します。

```
[root@vm1-□ ~]# service network restart
Shutting down interface eth0:                              [  OK  ]
Shutting down loopback interface:                          [  OK  ]
Bringing up loopback interface:                            [  OK  ]
Bringing up interface eth0:                                [  OK  ]
Bringing up interface eth0.10□:                            [  OK  ]
```

- VLANデバイス「eth0.10□」が構成されていることを確認します。

```
[root@vm1-□ ~]# ifconfig
eth0      Link encap:Ethernet  HWaddr 52:54:00:47:02:46
          inet6 addr: fe80::5054:ff:fe47:246/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:152834 errors:0 dropped:0 overruns:0 frame:0
          TX packets:23063 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:10737762 (10.2 MiB)  TX bytes:20725900 (19.7 MiB)

eth0.10□  Link encap:Ethernet  HWaddr 52:54:00:47:02:46
          inet addr:192.168.122.10□  Bcast:192.168.122.255  Mask:255.255.255.0
          inet6 addr: fe80::5054:ff:fe47:246/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:10 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 b)  TX bytes:600 (600.0 b)
...
```

VLANデバイスの構成 (3)

- procファイルシステムからVLANデバイスの構成情報を確認します。

```
[root@vm1-□ ~]# cat /proc/net/vlan/config
VLAN Dev name      | VLAN ID
Name-Type: VLAN_NAME_TYPE_RAW_PLUS_VID_NO_PAD
eth0.10□       | 10□  | eth0

[root@vm1-□ ~]# cat /proc/net/vlan/eth0.10□
eth0.10□  VID: 10□       REORDER_HDR: 1  dev->priv_flags: 1
         total frames received            0
          total bytes received            0
      Broadcast/Multicast Rcvd            0

      total frames transmitted           12
       total bytes transmitted          768
            total headroom inc            0
           total encap on xmit           12
Device: eth0
INGRESS priority mappings: 0:0  1:0  2:0  3:0  4:0  5:0  6:0 7:0
 EGRESS priority mappings:
```

- VM2-□に対するpingの疎通を確認します。

```
[root@vm1-□ ~]# ping 192.168.122.20□
PING 192.168.122.20□ (192.168.122.20□) 56(84) bytes of data.
^C                                                          ← Ctrl+Cで停止
--- 192.168.122.20□ ping statistics ---
2 packets transmitted, 0 received, 100% packet loss, time 1662ms
```

  - VM2-□ではVLANデバイスが構成されていないため、VM1-□からの（「VLAN ID 10□」のタグがついた）パケットを受け取れず、pingの応答がありません。

VLANデバイスの構成 (4)

- VM2-□に対してもVLANデバイスの設定を行います。virt-managerのウィンドウで「VM2-□」をダブルクリックして仮想コンソールを開き、rootユーザでログインします。
- 既存のネットワーク設定ファイルをコピーしてバックアップします。

```
[root@vm2-□ ~]# mkdir /root/work
[root@vm2-□ ~]# cp /etc/sysconfig/network-scripts/ifcfg-eth0 /root/work/
```

- 設定ファイル「/etc/sysconfig/network」「/etc/sysconfig/network-scripts/ifcfg-eth0」「/etc/sysconfig/network-scripts/ifcfg-eth0.10□」を次のように修正／作成します。

/etc/sysconfig/network

```
NETWORKING=yes
HOSTNAME=vm2-□
VLAN=yes        ← この行を追加
```

/etc/sysconfig/network-scripts/ifcfg-eth0

```
DEVICE=eth0
BOOTPROTO=none
NM_CONTROLLED=no
ONBOOT=yes
```

/etc/sysconfig/network-scripts/ifcfg-eth0.10□

```
DEVICE=eth0.10□
BOOTPROTO=static
NM_CONTROLLED=no
ONBOOT=yes
IPADDR=192.168.122.20□
NETMASK=255.255.255.0
GATEWAY=192.168.122.1
```

VLANデバイスの構成 (5)

- networkサービスを再起動して、設定変更を反映します。

```
[root@vm2-□ ~]# service network restart
Shutting down interface eth0:                              [  OK  ]
Shutting down loopback interface:                          [  OK  ]
Bringing up loopback interface:                            [  OK  ]
Bringing up interface eth0:                                [  OK  ]
Bringing up interface eth0.10□:                            [  OK  ]
```

- VLANデバイス「eth0.10□」が構成されていることを確認します。

```
[root@vm2-□ ~]# ifconfig
eth0      Link encap:Ethernet  HWaddr 52:54:00:EF:F5:C0
          inet6 addr: fe80::5054:ff:feef:f5c0/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:803 errors:0 dropped:0 overruns:0 frame:0
          TX packets:462 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:62759 (61.2 KiB)  TX bytes:42861 (41.8 KiB)

eth0.10□  Link encap:Ethernet  HWaddr 52:54:00:EF:F5:C0
          inet addr:192.168.122.20□  Bcast:192.168.122.255  Mask:255.255.255.0
          inet6 addr: fe80::5054:ff:feef:f5c0/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:12 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 b)  TX bytes:768 (768.0 b)
...
```

VLANデバイスの構成 (6)

- 再度、VM1-□から、VM2-□に対するpingの疎通を確認します。

```
[root@vm1-□ ~]# ping 192.168.122.20□
PING 192.168.122.20□ (192.168.122.20□) 56(84) bytes of data.
64 bytes from 192.168.122.20□: icmp_seq=1 ttl=64 time=0.117 ms
64 bytes from 192.168.122.20□: icmp_seq=2 ttl=64 time=0.159 ms
64 bytes from 192.168.122.20□: icmp_seq=3 ttl=64 time=0.166 ms
^C
--- 192.168.122.20□ ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2161ms
rtt min/avg/max/mdev = 0.117/0.147/0.166/0.023 ms
```

- procファイルシステムからVLANデバイスの送受信パケットが増加していることを確認します。

```
[root@vm1-□ ~]# cat /proc/net/vlan/eth0.10□
eth0.10□  VID: 10□       REORDER_HDR: 1  dev->priv_flags: 1
         total frames received            4
          total bytes received          296
      Broadcast/Multicast Rcvd            0

      total frames transmitted           16
       total bytes transmitted         1120
            total headroom inc            0
           total encap on xmit           16
Device: eth0
INGRESS priority mappings: 0:0  1:0  2:0  3:0  4:0  5:0  6:0 7:0
 EGRESS priority mappings:
```

- 以上で「VLANデバイスの構成」は完了です。

VLAN IDが異なる場合の動作確認 (1)

- VM2-□のVLANデバイスのVLAN IDを20□に変更します。
  - virt-managerのウィンドウで「VM2-□」をダブルクリックして仮想コンソールを開き、rootユーザでログインします。
  - networkサービスを停止して、NICインターフェースをダウンさせておきます。

```
[root@vm1-□ ~]# service network restart
Shutting down interface eth0.10□:                          [  OK  ]
Shutting down interface eth0:                              [  OK  ]
Shutting down loopback interface:                          [  OK  ]
```

  - 設定ファイル「/etc/sysconfig/network-scripts/ifcfg-eth0.10□」を「ifcfg-eth0.20□」にリネームします。

```
[root@vm2-□ ~]# cd /etc/sysconfig/network-scripts
[root@vm2-□ ~]# mv ifcfg-eth0.10□ ifcfg-eth0.20□
```

  - 設定フィアル「/etc/sysconfig/network-scripts/ifcfg-eth0.20□」を次のように変更します。

/etc/sysconfig/network-scripts/ifcfg-eth0.20□

```
DEVICE=eth0.20□      ← この行を変更
BOOTPROTO=static
NM_CONTROLLED=no
ONBOOT=yes
IPADDR=192.168.122.20□
NETMASK=255.255.255.0
GATEWAY=192.168.122.1
```

VLAN IDが異なる場合の動作確認 (2)

- networkサービスを起動します。

```
[root@vm2-□ ~]# service network start
Bringing up loopback interface:                            [  OK  ]
Bringing up interface eth0:                                [  OK  ]
Bringing up interface eth0.20□:                            [  OK  ]
```

- VLANデバイス「eth0.20□」が構成されていることを確認します。

```
[root@vm2-□ ~]# ifconfig
eth0      Link encap:Ethernet  HWaddr 52:54:00:EF:F5:C0
          inet6 addr: fe80::5054:ff:feef:f5c0/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:4414 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2892 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:367683 (359.0 KiB)  TX bytes:279995 (273.4 KiB)

eth0.20□  Link encap:Ethernet  HWaddr 52:54:00:EF:F5:C0
          inet addr:192.168.122.20□  Bcast:192.168.122.255  Mask:255.255.255.0
          inet6 addr: fe80::5054:ff:feef:f5c0/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:12 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 b)  TX bytes:768 (768.0 b)
. . .
```

VLAN IDが異なる場合の動作確認 (3)

- VM1-□に対するpingの疎通を確認します。

```
[root@vm2-□ ~]# ping 192.168.122.10□
PING 192.168.122.10□ (192.168.122.10□) 56(84) bytes of data.
^C                                                              ← Ctrl+Cで停止
--- 192.168.122.10□ ping statistics ---
2 packets transmitted, 0 received, 100% packet loss, time 1662ms
```

  - VM1-□とVM2-□では、VLAN idが異なるためpingの応答がありません。
- 以上で「VLAN IDが異なる場合の動作確認」は完了です。

ネットワーク構成の復旧 (1)

- VM1-□とVM2-□のネットワーク構成を元に戻します。
  - VM1-□でnetworkサービスを停止します。

```
[root@vm1-□ ~]# service network stop
Shutting down interface eth0.10□:                          [  OK  ]
Shutting down interface eth0:                              [  OK  ]
Shutting down loopback interface:                          [  OK  ]
```

  - VLANデバイスの設定ファイルを削除して、NIC（eth0）の設定ファイルをバックアップから戻します。

```
[root@vm1-□ ~]# rm /etc/sysconfig/network-scripts/ifcfg-eth0.10□
rm: remove regular file `/etc/sysconfig/network-scripts/ifcfg-eth0.10□'? y
[root@vm1-□ ~]# cp /root/work/ifcfg-eth0 /etc/sysconfig/network-scripts/
cp: overwrite `/etc/sysconfig/network-scripts/ifcfg-eth0'? y
```

  - Networkサービスを起動します。

```
[root@vm1-□ ~]# service network start
Bringing up loopback interface:                            [  OK  ]
Bringing up interface eth0:                                [  OK  ]
```

  - ホストLinuxのコマンド端末からVM1-□にログインできることを確認します。

```
# ssh root@192.168.122.10□    ← ホストLinuxのコマンド端末で実行
```

ネットワーク構成の復旧 (2)

- VM2-□でも同様の作業を行います。

```
[root@vm2-□ ~]# service network stop
Shutting down interface eth0:                              [  OK  ]
Shutting down interface eth0.20□:                          [  OK  ]
Shutting down loopback interface:                          [  OK  ]

[root@vm2-□ ~]# rm /etc/sysconfig/network-scripts/ifcfg-eth0.20□
rm: remove regular file `/etc/sysconfig/network-scripts/ifcfg-eth0.20□'? y
[root@vm2-□ ~]# cp /root/work/ifcfg-eth0 /etc/sysconfig/network-scripts/
cp: overwrite `/etc/sysconfig/network-scripts/ifcfg-eth0'? y

[root@vm2-□ ~]# service network start
Bringing up loopback interface:                            [  OK  ]
Bringing up interface eth0:                                [  OK  ]
```

- ホストLinuxのコマンド端末からVM2-□にログインできることを確認します。

```
# ssh root@192.168.122.20□    ← ホストLinuxのコマンド端末で実行
```

  - 先の演習でVM2-□のホスト証明書が変更されているため、ホストLinuxの「/home/user□/.ssh/known_hosts」から古いホスト証明書を削除しておく必要があります。

/home/user□/.ssh/known_hosts

```
192.168.122.10□ ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAQEAwDZsHjhbmD1rjdCQ/UOZt1n9LMc8zfeSpkKKQBfX
EVkM+zPwmxGihCPUiCURj7OuVdMzehRWMFX2q/wX/GNC2kUBvfsqhtM1Pz1Xth5crt27MvvRzhUcT5EpRv0xGCtWTdX1
9PiQA7IRKjyYfN7G+xotXhHGzJQGTtEL42e/9QYnqlyohvQhQTAv8ZmRTxlVFYQSbHJQJLKNP/MM2O8tjEkEntH2WnUr
sWkAn5YMc8wVqV6C+hUbMVWkL9vUyJdn29Hbc3xnErAz9KfwJZYCdc7iFswgaMr/4BYg3iLV1qTOM5iyrg3YwbQimpba
i+8GpQxro5sTDKiplLYMaOi0Tw==
192.168.122.20□ ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAQEAwDZsHjhbmD1rjdCQ/UOZt1n9LMc8zfeSpkKKQBfX
EVkM+zPwmxGihCPUiCURj7OuVdMzehRWMFX2q/wX/GNC2kUBvfsqhtM1Pz1Xth5crt27MvvRzhUcT5EpRv0xGCtWTdX1
9PiQA7IRKjyYfN7G+xotXhHGzJQGTtEL42e/9QYnqlyohvQhQTAv8ZmRTxlVFYQSbHJQJLKNP/MM2O8tjEkEntH2WnUr
sWkAn5YMc8wVqV6C+hUbMVWkL9vUyJdn29Hbc3xnErAz9KfwJZYCdc7iFswgaMr/4BYg3iLV1qTOM5iyrg3YwbQimpba
i+8GpQxro5sTDKiplLYMaOi0Tw==
```

この行を削除

ネットワーク構成の復旧 (3)

- 以上で「ネットワーク構成の復旧」は完了です。

- 以上で「VLANデバイス設定演習」は完了です。

メモとしてお使いください

メモとしてお使いください