DMZ セグメントを備えたサブネットの構築

名古屋大学情報学部コンピュータ科学科 3 年
堀内颯太 (学籍番号 : 101730331)
horiuchi.sota@e.mbox.nagoya-u.ac.jp

実験日 2019 / 5 / 16 , 5/ 23

目次

1 はじめに

1.1 概要

本レポートでは、DMZ ネットワーク・LAN(ローカルネットワーク) を備えるサブネットワークを構築行い、Linux におけるネットワーク構成方法やファイアウォールの設置方法についての知見を深めるとともに TCP/IP ネットワークにおける各種サービスの提供・利用のために必要な基礎知識を身に着ける。そこで本レポートでは、まず初めに、すべての実験で使用するルータと WWW サーバの初期環境の確認、設定を行う (実験 1)。次に、nmcli コマンドを利用することでネットワークインターフェイスの接続設定を行う (実験 2)。その後、DHCP サーバーの設定を行い、実際にサーバーを起動する (実験 3)。レポートの後半ではルータ及び DMZ 用 PC のファイアウォール設定を行い、内外部端末両方からの仕様の確認を行う (実験 4)。そして、WWW サーバを DMZ 用 PC で動作させる (実験 5)。最後にすべての実験を総括し、ネットワーク構築についてまとめる。

1.2 実験機器

本レポートのすべての実験では以下の機器を使用した。

- ルータ
- WWW サーバ
- PC
- PC 切り替え機
- モニタ
- キーボード・マウス
- Ethernet クロスケーブル
- Ethernet ストレートケーブル

ルータ (以後 machine1)、WWW サーバ (以後　machine 2)、PC(以後 machine 3) は PC 切替器を介してキーボード及びマウスと接続した。キーボードは PC 切替器のキーボード用端子に、マウスは PC 切替器のマウス用接続端子に接続した。PC 切替器と machine 1 及び machine 2、machine 3 は RGB ケーブルと USB ケーブルを用いて接続した。PC 切替器とモニタを RGB ケーブルを用いて接続した。Ethernet クロスケーブルを用いて machine 1 と machine 2、machine 1 と machine 3 の LAN インターフェース間を直接接続した。machine 1 と上位ネットワークは HUB を介して接続した。

1.3 DMZ ネットワーク・LAN

DMZ(DeMilitarized Zone: 非武装地帯) とはセキュリティゾーンのうちの 1 つである。セキュリティゾーンとは同セキュリティレベルをを持つネットワークの集合のことを指し、一般的なシステムにはファイアウォールを中心として DMZ のほかに、Untrust ゾーン、Trust ゾーン、WAN ゾーンの 4 種類のセキュリティゾーンが存在する。Untrust ゾーンはファイアウォールの外側に配置するシステムにとって信頼できない存在であり、セキュリティレベルは最も低く、ファイアウォールは Untrust ゾーンからの脅威に備える。イン

ターネットに接続している環境であればインターネットが Untrust ゾーンにあたる。Trust ゾーンはファイアウォールの内側に配置するシステムにとって信頼できるゾーンであり、セキュリティレベルが最も高く、絶対に死守する必要がある。尚、一般的に一つの拠点内の複数の機器間のデータ通信が可能なネットワークのことである LAN はこの Trust ゾーンと同義である。Trust ゾーンにドメインコントローラーやファイルサーバといったインターネットに公開しない社内サーバーや社内ユーザーを配置する。Trust ゾーンは基本、他ゾーンからの通信を基本拒否し他ゾーンへの通信を許可する。WAN ゾーンは Trust ゾーンと同等のセキュリティレベルをもち、VPN を利用して他拠点と接続するゾーンである。DMZ は Untrust ゾーンと Trust ゾーンの緩衝材の役割を果たすゾーンであり、セキュリティレベルは Untrust ゾーンより高く、Trust ゾーンより低くちょうど中間に位置する。DMZ には、Web サーバーや DNS サーバ、プロシキサーバーなどの Untrust ゾーンと直接的にやりとりする公開サーバを配置する。公開サーバーは不特定多数のクライアントがアクセスするため、セキュリティ的に最も危険なサーバであり、インターネットからのサイバー攻撃に耐えるために他ゾーンからの通信は最小限にする必要がある。[2]

1.4 ファイアウォール

ファイアウォールとはインターネットに公開するサーバーを保護するネットワーク機器である。主に IP アドレスやポート番号をもとにして通信の拒否、許可を決定する。この通信の拒否、許可のことをセキュリティポリシーと呼び、インターネットから公開サーバに対する通信であるインバウンド通信と LAN からインターネットに対する通信であるアウトバウンド通信に分けて実現する。基本的にインバウンド通信では最低限の通信のみを許可しそれ以外は拒否をするよう設定してサーバーのセキュリティを保つ。アウトバウンド通信では対照的に概ねの通信を許可し、一部の通信のみを拒否することでユーザーの利用可能な通信を確保する。現在存在するファイアウォールはトラディショナルファイアウォール、UTM(Unified Threat Management)、次世代ファイアウォール、WAF(ウェブアプリケーションファイアウォール) の 4 種類に分けることができる。トラディショナルファイアウォールは必要最低限のセキュリティレベルであり、IP アドレスとポート番号での通信制御を行う。単純サーバーう劇にしか対応できないが、安価であり手軽に導入することができる。UTM はこれまで別々の機器で行っていたアンチウイルス機能やアンチスパム機能、VPN 機能、IPS/IDS 機器などの様々なセキュリティ機能を一台にまとめた機器であり、管理台数が一台になるため管理が容易になる。次世代ファイアウォールはインターネットに対するユーザートラフィックをアプリケーションレベルで細かく制御することができる。UTM にアプリケーション識別、可視化などの新しい機能を加えた機器である。通信のふるまいを監視することでアプリケーションを識別しそれをルールとして設定する。また、そのトラフィックをグラフや表にすることで統計的に管理し、可視化する。WAF はクロスサイトスクリプティングや SQL インジェクション、クロスサイトリクエストフォージェリといったような Web アプリケーションの脆弱性を狙う攻撃を検知し、ブロックするファイアウォールである。Web ブラウザと Web サーバ間のすべてのやり取りを監視することでアプリケーションレベルでの制御を行う。[2]

1.5 TCP/IP

TCP/IP とは Transmission Control Protcol/Internet Protocol の略称であり、コンピュータ同士がコンピュータネットワークなどの通信回線を介してデータ通信を行う際に使用される通信規約であるプロトコルの集まりのことを指すネットワークアーキテクチャの一つである。現在、最も広く利用されているネットワーク

アーキテクチャであり、ネットワーク通信に必要な機能をアプリケーション層、トランスポート層、インターネット層、ネットワークインターフェイス層といった 4 つの階層にわけて考え、各階層の通信プロトコルを組み合わせて通信を実現する。TCP/IP のプロトコルはネットワークインターフェース層以外の層のプロトコル群であり、これらの使用は RFC(Request For Comment) により表されている。なお、ネットワークインターフェース層におけるプロトコルは LAN や WAN でのデータ通信を行うためのプロトコルである。通信を行う PC やサーバは LAN に接続されており、遠距離の LAN 同士を接続するために WAN を利用する。これらのデータ転送に使われるプロトコルにはさまざまなものが存在する。TCP/IP ではこれらのプロトコルの規定していないため、TCP/IP で通信する際、ネットワークインターフェース層のプロトコルは自由であり、どのような LAN、WAN のネットワークであっても TCP/IP によって実現することができる。

2 実験 1: 初期環境の確認

2.1 実験目的・概要

本実験では、PC を用いてネットワーク構築をする際に必要な情報を獲得するために、使用する PC の初期環境の確認や設定を行う。

2.2 実験方法

コンソールより、machine1、machine2 の両方で以下の設定を行った。

2.2.1 Linux カーネルリリース番号の確認

以下のコマンドを実行し、現在動作している Linux カーネルを確認した。

```
uname -r
```

2.2.2 ファイルシステムの確認

1. 以下のコマンドを実行し、ファイルシステムのマウントポイントを記述した設定ファイル /etc/fatab の内容を確認した。

   ```
   cat /etc/fstab
   ```

2. 以下のコマンドを実行し、ディスクパーティションとマウントされているファイルシステムを確認した。

   ```
   fdisk -l
   df
   ```

3. 以下のコマンドを実行し、論理ボリュームの内容を確認した。

   ```
   lvdisplay
   ```

2.2.3 ホスト名の設定

machine1 と machine2 で異なる設定をした。

- machine1

  以下のコマンドを実行しホスト名を設定し、設定結果を確認した。

```
hostname icesc12.ice.nuie.nagoya-u.ac.jp
hostname
```

  設定ファイル/etc/hostname の内容を以下に変更し、ホスト名の恒久的変更をした。

```
icesc12.ice.nuie.nagoya-u.ac.jp
```

- machine2

  以下のコマンドを実行しホスト名を設定し、設定結果を確認した。

```
hostname www2.ice.nuie.nagoya-u.ac.jp
hostname
```

  vi エディタで設定ファイル/etc/hostname の内容を以下のように変更し、ホスト名の恒久的変更をした。

```
www2.ice.nuie.nagoya-u.ac.jp
```

2.2.4 SELinux の状態確認と無効化

1. 現在の設定状態の確認

   以下のコマンドを実行し、現在の SELinux の設定状態の確認をした。

```
cat /etc/sysconfig/selinux
getenforce
```

2. SELinux の設定を Permissive モードに変更

   以下のコマンドを実行し、SELinux の設定を permissive モードの変更した。

```
setenforce 0
```

   SELinux の設定ファイル/etc/sysconfig/selinux の内容を以下のように変更し、SELinux の設定を恒久的にした。

```
SELINUX=permissive
```

2.3 実験結果

2.3.1 Linux カーネルリリース番号の確認

Linux カーネルリリース番号はそれぞれ以下の通りであった。

- machine1

```
[root@localhost ~]# uname -r\\
3.10.0-862.el7.x86_64
```

- machine2

```
[root@localhost ~]# uname -r\\
3.10.0-862.el7.x86_64
```

それぞれのカーネルバージョンは同じで 3.10.0-862.el7.x86_64 であることがわかった。

2.3.2 ファイルシステムの確認

1. ファイルシステムのマウントポイントを記述した設定ファイル/etc/fstab の内容

   - machine1

```
[root@localhost ~]# cat /etc/fstab

#
# /etc/fstab
# Created by anaconda on Fri Mar 22 12:04:29 2019
#
# Accessible filesystems, by reference, are maintained under '/dev/disk'
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info
#
/dev/mapper/centos-root /                       xfs     defaults        0 0
UUID=aee50255-ddf9-4231-babe-4df05beb2989 /boot xfs     defaults        0 0
/dev/mapper/centos-home /home                   xfs     defaults        0 0
/dev/mapper/centos-swap swap                    swap    defaults
```

   - machine2

```
[root@localhost ~]# cat /etc/fstab

#
# /etc/fstab
# Created by anaconda on Sat Apr  6 00:42:18 2019
#
# Accessible filesystems, by reference, are maintained under '/dev/disk'
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info
#
/dev/mapper/centos-root /                       xfs     defaults        0 0
UUID=5eb200c5-354e-419c-9306-bdcc507c72c5 /boot xfs     defaults        0 0
/dev/mapper/centos-home /home                   xfs     defaults        0 0
/dev/mapper/centos-swap swap                    swap    defaults        0 0
```

   #以外の部分が fstab の内容である。fstab はディスクパーティションや様々なブロックデバイス、リモートファイルをどのようにしてファイルシステムにマウントするかを記述している。1 列目はマウントされるパーティションやストレージデバイスのデバイス名、2 列目が同行 1 列目のデバイスがマウントされるマウントポイントを示している。つまり例えば 1 行目は、/が/dev/mapper/centos-root としてマウントされている。3 列目はマウントされるパーティションやストレージデバイスのファイルシステムタイプ名、4 列目は使用されるファイルシステムのマウントオプションを示す。5 列目は Dump が

エントリをチェックし、ファイルシステムがバックアップされるべきかどうか決定するための真理値、6 列目はファイルシステムをチェックする順番を示す。そもそも、Linux は、Windows と異なり、ハード・ディスクや USB メモリー、光学ドライブといったストレージ・デバイスを接続しただけではファイルの読み書きはできない。これらのデバイスを Linux システムに認識させ、利用できるようにする作業をマウントと呼び、fstab はそのマウントの方法について記したものである。machine1、machine2 ではそれぞれ 4 つの記憶装置が認識されていることがわかった。/boot がデバイス名指定でなく UUID であるのは、ファイルシステム起動時にほかファイルシステムと認識される順番によるデバイス名の変化に対応するためである。

2. ディスクパーティション確認
   - machine1

```
[root@localhost ~]# fdisk -l

Disk /dev/sda: 250.1 GB, 250059350016 bytes, 488397168 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: dos
Disk identifier: 0x00016021

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *        2048     2099199     1048576   83  Linux
/dev/sda2         2099200   488396799   243148800   8e  Linux LVM

Disk /dev/mapper/centos-root: 53.7 GB, 53687091200 bytes, 104857600 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes


Disk /dev/mapper/centos-swap: 8321 MB, 8321499136 bytes, 16252928 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes


Disk /dev/mapper/centos-home: 187.0 GB, 186969489408 bytes, 365174784 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
[root@localhost ~]# df
Filesystem              1K-blocks    Used Available Use% Mounted on
/dev/mapper/centos-root  52403200 1003696  51399504   2% /
devtmpfs                  3936440       0   3936440   0% /dev
tmpfs                     3949112       0   3949112   0% /dev/shm
tmpfs                     3949112    8860   3940252   1% /run
tmpfs                     3949112       0   3949112   0% /sys/fs/cgroup
/dev/sda1                 1038336  145900    892436  15% /boot
/dev/mapper/centos-home 182498240   32944 182465296   1% /home
tmpfs                      789824       0    789824   0% /run/user/0
```

   - machine2

```
[root@localhost ~]# fdisk -l

Disk /dev/sda: 250.1 GB, 250059350016 bytes, 488397168 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: dos
Disk identifier: 0x0002caf3

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *        2048     2099199     1048576   83  Linux
/dev/sda2         2099200   488396799   243148800   8e  Linux LVM

Disk /dev/mapper/centos-root: 53.7 GB, 53687091200 bytes, 104857600 sectors
Units = sectors of 1 * 512 = 512 bytes
```

```
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes


Disk /dev/mapper/centos-swap: 8321 MB, 8321499136 bytes, 16252928 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes


Disk /dev/mapper/centos-home: 187.0 GB, 186969489408 bytes, 365174784 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
[root@localhost ~]# df
Filesystem              1K-blocks    Used Available Use% Mounted on
/dev/mapper/centos-root  52403200 1007240  51395960   2% /
devtmpfs                  3913236       0   3913236   0% /dev
tmpfs                     3925932       0   3925932   0% /dev/shm
tmpfs                     3925932    9012   3916920   1% /run
tmpfs                     3925932       0   3925932   0% /sys/fs/cgroup
/dev/sda1                 1038336  146056    892280  15% /boot
/dev/mapper/centos-home 182498240   32944 182465296   1% /home
tmpfs                      785188       0    785188   0% /run/user/0
```

fdisk -l は記憶装置を論理的に分割した際の領域 (パーティション) の位置や長さなどの管理情報を一覧表示するコマンドである。machine1、machine2 では/etc/fstab で確認した通りそれぞれ 4 つの記憶装置が読み込まれていることがわかった。/dev/ada には 2 つのパーティションが作成されていて、sda2 が論理ボリュームマネージャであることがわかった。df はマウントされているディスクの空き容量を示すコマンドであり、1 列目からデバイス名、ディスク容量、使用容量、空き容量、使用率、マウントポイントを表している。それぞれの空き容量を確認した。

3. 倫理ボリュームの内容確認
   - machine1

```
[root@localhost ~]# lvdisplay
  --- Logical volume ---
  LV Path                /dev/centos/swap
  LV Name                swap
  VG Name                centos
  LV UUID                5S8Xtv-OZ7e-3krn-HqEs-8xHA-CYFl-BfapS4
  LV Write Access        read/write
  LV Creation host, time localhost, 2019-03-22 12:04:24 +0900
  LV Status              available
  # open                 2
  LV Size                7.75 GiB
  Current LE             1984
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           253:1

  --- Logical volume ---
  LV Path                /dev/centos/home
  LV Name                home
  VG Name                centos
  LV UUID                vanhId-NRrd-icZs-ugHk-PTLw-F5zJ-XgnuQb
  LV Write Access        read/write
  LV Creation host, time localhost, 2019-03-22 12:04:25 +0900
  LV Status              available
  # open                 1
  LV Size                <174.13 GiB
  Current LE             44577
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           253:2

  --- Logical volume ---
```

```
  LV Path                /dev/centos/root
  LV Name                root
  VG Name                centos
  LV UUID                zNAdo6-xqs9-t9PR-iKrf-C855-jD6d-LvOl7q
  LV Write Access        read/write
  LV Creation host, time localhost, 2019-03-22 12:04:26 +0900
  LV Status              available
  # open                 1
  LV Size                50.00 GiB
  Current LE             12800
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           253:0
```

- machine2

```
[root@localhost ~]# lvdisplay
  --- Logical volume ---
  LV Path                /dev/centos/swap
  LV Name                swap
  VG Name                centos
  LV UUID                PohTZH-fMlu-Lf0h-Mdhh-Z2xs-I2QH-ZM8ush
  LV Write Access        read/write
  LV Creation host, time localhost, 2019-04-06 00:42:14 +0900
  LV Status              available
  # open                 2
  LV Size                7.75 GiB
  Current LE             1984
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           253:1

  --- Logical volume ---
  LV Path                /dev/centos/home
  LV Name                home
  VG Name                centos
  LV UUID                XYwPiS-jXOD-IKEh-Jzzo-NDvs-qfED-uV8ABr
  LV Write Access        read/write
  LV Creation host, time localhost, 2019-04-06 00:42:14 +0900
  LV Status              available
  # open                 1
  LV Size                <174.13 GiB
  Current LE             44577
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           253:2

  --- Logical volume ---
  LV Path                /dev/centos/root
  LV Name                root
  VG Name                centos
  LV UUID                Oz554A-DomL-OGcW-B3u7-yAir-UU6R-Fa3bE1
  LV Write Access        read/write
  LV Creation host, time localhost, 2019-04-06 00:42:16 +0900
  LV Status              available
  # open                 1
  LV Size                50.00 GiB
  Current LE             12800
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           253:0
```

lvdisplay では作成されている論理ボリュームを表示する。swap、home、root の三つの論理ボリュームが作成されていることが分かった。論理ボリュームは仮想的なパーティションであり、ボリュームグループ上に作成されており、これにより複数の HDD をまとめて 1 つのファイルシステムとして認識が

可能となる。

2.3.3 ホスト名の設定

- machine1

```
[root@localhost ~]# hostname
icesc12.ice.nuie.nagoya-u.ac.jp
```

- machine2

```
[root@localhost ~]# hostname
www2.ice.nuie.nagoya-u.ac.jp
```

hostname コマンドによりホスト名を設定し、設定結果を確認した。

2.3.4 SELinux の状態確認

- machine1

```
[root@icesc12 ~]# cat /etc/sysconfig/selinux

# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#     enforcing - SELinux security policy is enforced.
#     permissive - SELinux prints warnings instead of enforcing.
#     disabled - No SELinux policy is loaded.
SELINUX=enforcing
# SELINUXTYPE= can take one of three two values:
#     targeted - Targeted processes are protected,
#     minimum - Modification of targeted policy. Only selected processes are protected.
#     mls - Multi Level Security protection.
SELINUXTYPE=targeted


[root@icesc12 ~]# getenforce
Enforcing[root@icesc12 ~]# cat /etc/sysconfig/selinux

# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#     enforcing - SELinux security policy is enforced.
#     permissive - SELinux prints warnings instead of enforcing.
#     disabled - No SELinux policy is loaded.
SELINUX=enforcing
# SELINUXTYPE= can take one of three two values:
#     targeted - Targeted processes are protected,
#     minimum - Modification of targeted policy. Only selected processes are protected.
#     mls - Multi Level Security protection.
SELINUXTYPE=targeted

[root@icesc12 ~]# getenforce
Enforcing
```

- machine2

```
[root@www2 ~]# cat /etc/sysconfig/selinux

# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#     enforcing - SELinux security policy is enforced.
#     permissive - SELinux prints warnings instead of enforcing.
#     disabled - No SELinux policy is loaded.
SELINUX=enforcing
# SELINUXTYPE= can take one of three two values:
#     targeted - Targeted processes are protected,
#     minimum - Modification of targeted policy. Only selected processes are protected.
#     mls - Multi Level Security protection.
```

```
SELINUXTYPE=targeted

[root@www2 ~]# getenforce
Enforcing
```

SELinux が Enforcing(有効) になっていることがわかる。/etc/sysconfig/selinux では#下の SELINUX 及び SELINUXTYPE に設定することができる値の説明がされていて、enforsing では SELINUX が有効な状態である。

2.4 考察

SELinux を無効にした理由について考察する。そもそも SELinux は Linux カーネルのセキュリティ拡張モジュールである。大きな特徴として、root 権限を奪取された際の危険性を考え、各プロセスが最小限の権限で動くように設計されている。その中の機能に TE(Type Enforcement) というものがある。これはプロセス毎にアクセス制限をかける機能である。これが機能している状態のまま、サーバーの設定をするとプロセス毎にアクセス制御がかかり、アプリの軌道を妨げることがあり容易にサーバーを立てられないため、無効にしたと考えられる。

また、fdisk -l で得られた情報について考察する。/dev/sda1 と/dev/sda2 の容量が大きく異なる。これは sda2 が論理ボリュームマネージャであり、これのもとに仮想的なパーティションが作成されるためにこのように大きなサイズになっていると考えられる。

3 実験 2: ネットワーク設定

3.1 実験目的・概要

本実験では、machine1、machine2 について、nmcli コマンドを利用して、PC のネットワークインタフェースの接続確認を行い、ネットワーク接続を可能にする。

3.2 実験方法

3.2.1 現在の状況の確認

1. ネットワークデバイスの確認
   以下のコマンドを実行し、ネットワークデバイスの確認を行った。

   ```
   nmcli device status
   nmcli device show
   ```

2. ネットワーク接続の確認
   以下のコマンドを実行しネットワーク接続の確認を行った

   ```
   nmcli connection show –active
   ```

3.2.2　接続設定

下記コマンドを実行してネットワークの接続設定をした、

- machine1

```
nmcli c m enp1s0 ipv4.addresses 192.168.100.12/24
nmcli c m enp1s0 ipv4.dns 10.10.1.2
nmcli c m enp1s0 ipv4.dns-search ”ice.nuie.nagoya-u.ac.jp”
nmcli c m enp1s0 gateway 192.168.100.1
nmcli c m enp1s0 connection.autoconnect yes
nmcli c m enp1s0 ipv4.method manual
nmcli c m enp2s0 ipv4.addresses 192.168.150.1/24
nmcli c m enp2s0 ipv4.method manual
nmcli c m enp2s0 connection.autoconnect yes
nmcli c m enp3s0 ipv4.addresses 192.168.200.1/24
nmcli c m enp3s0 ipv4.method manual
nmcli c m enp3s0 connection.autoconnect yes
```

- machine2

```
nmcli c m enp3s0 ipv4.method auto
nmcli c m enp3s0 connection.autoconnect yes
```

3.2.3　ネットワークの接続の有効化

下記のコマンドを実行しそれぞれ機器のネットワーク接続を一度停止したのち開始した。

- machine1

```
nmcli c down enp1s0
nmcli c down enp2s0
nmcli c down enp3s0
nmcli c up enp1s0
nmcli c up enp2s0
nmcli c up enp3s0
```

- machine2

```
nmcli c down enp3s0
nmcli c up enp3s0
```

3.2.4 各種情報の確認

以下のコマンドを実行し、接続の確認を行った。

ip addr show
ip route

3.3 実験結果

3.3.1 現在の状況の確認

1. ネットワークデバイスの確認
   - machine1

```
[root@icesc12 ~]# nmcli device status
DEVICE  TYPE      STATE         CONNECTION
enp1s0  ethernet  disconnected  --
enp2s0  ethernet  disconnected  --
enp3s0  ethernet  disconnected  --
enp4s0  ethernet  unavailable   --
lo      loopback  unmanaged     --
[root@icesc12 ~]# nmcli device show
GENERAL.DEVICE:                         enp1s0
GENERAL.TYPE:                           ethernet
GENERAL.HWADDR:                         00:E0:67:12:2D:E4
GENERAL.MTU:                            1500
GENERAL.STATE:                          30 (disconnected)
GENERAL.CONNECTION:                     --
GENERAL.CON-PATH:                       --
WIRED-PROPERTIES.CARRIER:               on

GENERAL.DEVICE:                         enp2s0
GENERAL.TYPE:                           ethernet
GENERAL.HWADDR:                         00:E0:67:12:2D:E5
GENERAL.MTU:                            1500
GENERAL.STATE:                          30 (disconnected)
GENERAL.CONNECTION:                     --
GENERAL.CON-PATH:                       --
WIRED-PROPERTIES.CARRIER:               on

GENERAL.DEVICE:                         enp3s0
GENERAL.TYPE:                           ethernet
GENERAL.HWADDR:                         00:E0:67:12:2D:E6
GENERAL.MTU:                            1500
GENERAL.STATE:                          30 (disconnected)
GENERAL.CONNECTION:                     --
GENERAL.CON-PATH:                       --
WIRED-PROPERTIES.CARRIER:               on

GENERAL.DEVICE:                         enp4s0
GENERAL.TYPE:                           ethernet
GENERAL.HWADDR:                         00:E0:67:12:2D:E7
GENERAL.MTU:                            1500
GENERAL.STATE:                          20 (unavailable)
GENERAL.CONNECTION:                     --
GENERAL.CON-PATH:                       --
WIRED-PROPERTIES.CARRIER:               off

GENERAL.DEVICE:                         lo
GENERAL.TYPE:                           loopback
GENERAL.HWADDR:                         00:00:00:00:00:00
GENERAL.MTU:                            65536
GENERAL.STATE:                          10 (unmanaged)
GENERAL.CONNECTION:                     --
GENERAL.CON-PATH:                       --
IP4.ADDRESS[1]:                         127.0.0.1/8
IP4.GATEWAY:                            --
IP6.ADDRESS[1]:                         ::1/128
```

```
IP6.GATEWAY:                            --
```

- machine2

```
[root@www2 ~]# nmcli device status
DEVICE  TYPE      STATE         CONNECTION
enp3s0  ethernet  disconnected  --
lo      loopback  unmanaged     --
wlp2s0  wifi      unmanaged     --
[root@www2 ~]# nmcli device show
GENERAL.DEVICE:                         enp3s0
GENERAL.TYPE:                           ethernet
GENERAL.HWADDR:                         94:C6:91:A8:C6:B9
GENERAL.MTU:                            1500
GENERAL.STATE:                          30 (disconnected)
GENERAL.CONNECTION:                     --
GENERAL.CON-PATH:                       --
WIRED-PROPERTIES.CARRIER:               on

GENERAL.DEVICE:                         lo
GENERAL.TYPE:                           loopback
GENERAL.HWADDR:                         00:00:00:00:00:00
GENERAL.MTU:                            65536
GENERAL.STATE:                          10 (unmanaged)
GENERAL.CONNECTION:                     --
GENERAL.CON-PATH:                       --
IP4.ADDRESS[1]:                         127.0.0.1/8
IP4.GATEWAY:                            --
IP6.ADDRESS[1]:                         ::1/128
IP6.GATEWAY:                            --

GENERAL.DEVICE:                         wlp2s0
GENERAL.TYPE:                           wifi
GENERAL.HWADDR:                         18:56:80:A9:5C:5E
GENERAL.MTU:                            1500
GENERAL.STATE:                          10 (unmanaged)
GENERAL.CONNECTION:                     --
GENERAL.CON-PATH:                       --
IP4.GATEWAY:                            --
IP6.GATEWAY:                            --
```

numcli device status では 1 列目がデバイス名、2 列目がデバイスタイプ、3 列目がデバイスの状態、接続先を示している。machine1 では enp1s0、enp2s0、enp3s0 の 3 種類のネットワークデバイスが、machine2 では enp3s が使用できるが、接続されていないことがわかった。尚、lo はローカルループバックを表し、これは実際の別の機器との通信には使わない。また wlp2s0 は無線 LAN のデバイスであるが、現設定では使用できないため unmanaged になっていた。

nmcli device show では nmcli device status で確認したネットワークデバイスの詳細が表示される。GENERAL.DEVICE はデバイス名、GENERAL.TYPE はコネクションタイプ、GENERAL.HWADDR は MAC アド レス、GENERAL.MTU は一度に送信可能な最大データ量 (byte)、GENERAL.STATE は 接続状態をそれぞれ 100(接続済み)、30(未接続)、20(使用不可)、10(管理なし) で表示する。ローカルループバックアドレスは自分自身を表す IP アドレスであり 一般的に 127.0.0.1 が利用される。現時点では IP アドレスは与えられていないため IP アドレスは表示されなかった。また、有線でつながっているものについては WIRED-PROPERTIES.CARRIER がオンになっている。

3.3.2 ネットワーク接続確認

- machine1

```
[root@www2 ~]# nmcli connection show --active
NAME  UUID  TYPE  DEVICE
```

- machine2

```
[root@icesc12 ~]# nmcli connection show  --active
NAME  UUID  TYPE  DEVICE
```

現段階では、ネットワーク接続されたデバイスが存在しないため何も表示されなかった。

3.4 ネットワーク接続の有効化

- machine1

```
[root@icesc12 ~]# nmcli c down enp1s0
Connection 'enp1s0' successfully deactivated (D-Bus active path: /org/freedesktop/
    NetworkManager/ActiveConnection/13)
[root@icesc12 ~]# nmcli c down enp2s0
Connection 'enp2s0' successfully deactivated (D-Bus active path: /org/freedesktop/
    NetworkManager/ActiveConnection/20)
[root@icesc12 ~]# nmcli c down enp3s0
Connection 'enp3s0' successfully deactivated (D-Bus active path: /org/freedesktop/
    NetworkManager/ActiveConnection/21)
[root@icesc12 ~]# nmcli c up enp1s0
Connection successfully activated (D-Bus active path: /org/freedesktop/NetworkManager/
    ActiveConnection/24)
[root@icesc12 ~]# nmcli c up enp2s0
Connection successfully activated (D-Bus active path: /org/freedesktop/NetworkManager/
    ActiveConnection/25)
[root@icesc12 ~]# nmcli c up enp3s0
Connection successfully activated (D-Bus active path: /org/freedesktop/NetworkManager/
    ActiveConnection/26)
```

- machine2

```
[root@www2 ~]# nmcli c down enp3s0
Connection 'enp3s0' successfully deactivated (D-Bus active path: /org/freedesktop/
    NetworkManager/ActiveConnection/16)
[root@www2 ~]# nmcli c up enp3s0
Connection successfully activated (D-Bus active path: /org/freedesktop/NetworkManager/
    ActiveConnection/24)
```

デバイスを一度停止させ再び起動させることで設定内容を更新した。接続設定時に接続を指定したために D-Bus active path が与えられており、デバイスが接続されていることが確認できた。

3.4.1 各種情報の確認

ip addr では各デバイスの設定の確認を、ip route ではルーティング情報の確認を行った。

- ip addr
  - machine1

```
[root@icesc12 ~]# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group
     default qlen 1000
    link/ether 00:e0:67:12:2d:e4 brd ff:ff:ff:ff:ff:ff
    inet 192.168.100.12/24 brd 192.168.100.255 scope global noprefixroute enp1s0
       valid_lft forever preferred_lft forever
    inet6 fe80::3cc8:690f:c372:ceb4/64 scope link noprefixroute
```

```
       valid_lft forever preferred_lft forever
3: enp2s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group
    default qlen 1000
    link/ether 00:e0:67:12:2d:e5 brd ff:ff:ff:ff:ff:ff
    inet 192.168.150.1/24 brd 192.168.150.255 scope global noprefixroute enp2s0
       valid_lft forever preferred_lft forever
    inet6 fe80::5d6a:ded4:1f29:f568/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
4: enp3s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group
    default qlen 1000
    link/ether 00:e0:67:12:2d:e6 brd ff:ff:ff:ff:ff:ff
    inet 192.168.200.1/24 brd 192.168.200.255 scope global noprefixroute enp3s0
       valid_lft forever preferred_lft forever
    inet6 fe80::f5dc:32d5:88c5:e899/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
5: enp4s0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast state DOWN group
    default qlen 1000
    link/ether 00:e0:67:12:2d:e7 brd ff:ff:ff:ff:ff:ff
```

– machine2

```
[root@www2 ~]# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp3s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group
    default qlen 1000
    link/ether 94:c6:91:a8:c6:b9 brd ff:ff:ff:ff:ff:ff
    inet 192.168.150.2/24 brd 192.168.150.255 scope global noprefixroute dynamic enp3s0
       valid_lft 86387sec preferred_lft 86387sec
    inet6 fe80::ba59:99df:dba0:efc3/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
3: wlp2s0: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen 1000
    link/ether 18:56:80:a9:5c:5e brd ff:ff:ff:ff:ff:ff
```

– machine3

```
    [ice@icesc00 machine3]$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp3s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group
    default qlen 1000
    link/ether 94:c6:91:a8:ef:1d brd ff:ff:ff:ff:ff:ff
    inet 192.168.200.100/24 brd 192.168.200.255 scope global noprefixroute dynamic enp3s0
       valid_lft 3551sec preferred_lft 3551sec
    inet6 fe80::96c6:91ff:fea8:ef1d/64 scope link
       valid_lft forever preferred_lft forever
3: wlp2s0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc mq state DOWN group default
    qlen 1000
    link/ether ae:26:94:ba:2a:67 brd ff:ff:ff:ff:ff:ff
4: virbr0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group
    default qlen 1000
    link/ether 52:54:00:d9:de:53 brd ff:ff:ff:ff:ff:ff
    inet 192.168.122.1/24 brd 192.168.122.255 scope global virbr0
       valid_lft forever preferred_lft forever
5: virbr0-nic: <BROADCAST,MULTICAST> mtu 1500 qdisc pfifo_fast master virbr0 state DOWN
    group default qlen 1000
    link/ether 52:54:00:d9:de:53 brd ff:ff:ff:ff:ff:ff
```

ip addr はネットワークデバイスの IP アドレスを出力するコマンドであり、これで得られた情報については以下のとおりである。

– lo、enp1s0、enp2s0、enp3s0、enp4s0、wlp2s0、virbr0、virbr0-nic
  それぞれのデバイス名ををを表す

– LOOPBACK

デバイスがループバックに対応していることを表す
- BROADCAST
  ブロードキャストに対応していることを表す
- MULTICAST
  マルチキャストに対応していることを表す
- UP
  ネットワークインタフェースが有効ことを表す
- LOWER UP
  デバイスに有線ケーブルがつながっていることを表す
- NO-CARRIER
  デバイスが有線ケーブルにつながっていないことを表す
- mtu 1500
  一度に送信可能なデータの最大値を表す (byte)
  実例では 1500byte であり、これは Linux の標準値である
- qdisc pfo_fast
  ネットワーク送信を待つ間のデータの保存規則を表し、パケットの優先度を考慮した queue を使用することを表す。他に qdisc noqueue がありこれはデータの保存規則がないことを表し
  一度に大量のデータを自身に送信することが可能な lo で使われる
- state
  ネットワークインターフェイスの稼働状況を表し、UP なら稼働中、DOWN なら非稼働中を表す。UNKOWN は lo で使われる
- group default
  グループインタフェースを表す
- qlen 1000
  queue の長さを表す
- link/loopback 00:00:00:00:00:00
  ループバックのマックアドレスを表す
- link/either 00:e0:67:12:2d:e4
  それぞれの MAC アドレスを表す
- brd 00:00:00:00:00:00 brd ff:ff:ff:ff:ff:ff
  それぞれの MAC のブロードキャストアドレスを表す
- inet 192.168.150.2/24 brd 192.168.150.255
  inet が IPv4 アドレスを表し、brd がディレクティッドブロードキャストアドレスを表す。brd は XXX.XXX.XXX.255 で表され、lo には brd は無い
- scope global
  送信先指定を表すし、global では他ネットワークへの
  ゲートウェイを利用したグローバルな送信先を表す。scope link ではローカルネットワーク内、scope host では自分自身の身を表す
- noprefixroute enp3s0
  予め該当のネットワークの経路を固定しないことを表す

– valid_lft forever preferred_lft forever
  valid_lft は IPv4 及び IPv6 のアドレスの有効期限を表す。preferred_lft 適切な IPv4 及び IPv6 のアドレスの有効期限を表す
– inet6 fe80::5d6a:ded4:1f29:f568/64
  iPv6 のアドレスを表す

machine1 では enp1s0、enp2s0、enp3s0 がケーブルですでに繋がっており、IP アドレスがあり、例えば、enp1s0 は 192.168.199.12/24IP アドレスを有しておりそのマックアドレスは 00:e0:67:12:2d:e4 であり、使用可能なものになっていることが分かった。machine2 では enp3s0 がケーブルにつながっており、IP アドレスがあり、その値は 192.168.150.2/24 であることが分かった。尚 machine3 の virbr0 は仮想ブリッジであり、ユーザの設定に関係なく設定されている場合がある。

- ip route
  – machine1

```
[root@icesc12 ~]# ip route
default via 192.168.100.1 dev enp1s0 proto static metric 100
192.168.100.0/24 dev enp1s0 proto kernel scope link src 192.168.100.12 metric 100
192.168.150.0/24 dev enp2s0 proto kernel scope link src 192.168.150.1 metric 101
192.168.200.0/24 dev enp3s0 proto kernel scope link src 192.168.200.1 metric 102
```

  – machine2

```
[root@www2 ~]# ip route
default via 192.168.150.1 dev enp3s0 proto dhcp metric 100
192.168.150.0/24 dev enp3s0 proto kernel scope link src 192.168.150.2 metric 100
```

  – machine3

```
    [ice@icesc00 machine3]$ ip route
default via 192.168.200.1 dev enp3s0 proto dhcp metric 100
192.168.122.0/24 dev virbr0 proto kernel scope link src 192.168.122.1
192.168.200.0/24 dev enp3s0 proto kernel scope link src 192.168.200.100 metric 100
```

ip route は管理者が宛先ネットワークへの最適なルートを手動で設定したルートスタティックルートを出力するコマンドであり、これで得られた情報については以下のとおりである。

– via 192.168.100.1
  ネクストホップのルータを表している。この場合データは 192.168.100.1 デフォルトゲートウェイを経由することがわかる
– dev enp1s0
  対象デバイス名を指す。この場合 enp1s0 インターフェイスから送信されることを意味する
– proto kernel、proto dhcp
  何によって経路が生成されたかを表す。kernel ではカーネルが、dhcp では DHCP が自動生成した経路である
– metric 100
  ネットワーク間の仮想的な距離を表す
– scope link
  送信先を表す。scope link では unicast/broadcast 通信する経路指し、自身が属するネットワーク等が該当する。他に、scope host が自分自身への経路、scope global がゲートウェイを経由した unicast 通信による経路を表す。scope オプションがない時は global に該当する

– src

送信元の指定を表す

machine1 では 192.168.100.1 がネクストホッパーであり、例えば 192.168.100.12 から来たデータは 192.168.100.0/24 で受け取られ、192.168.100.1 を経由することになる。machine2 では 192.168.150.1 がネクストホップのルータであることがわかる。これらは dhcpd.conf に基づいて生成されている。

machine3 に与えられた IP アドレスは 192.168.200.100 であり、FW 構築実験用ネットワークでのアドレスが 192.168.100.12 であることが分かった。

3.5 考察

ネットワークデバイスの確認において numcli device status を表示した際 machine1 では、enp1s0、enp2s0、enp3s0 の 3 種類のネットワークデバイスが使用可能であるが、接続されていない disconnected の状態であった。通常このような状態の下では nmcli connection add でデバイスをしない限り設定は行えない。しかし、本実験では行うことができた。この理由について考察する。

[1] によると、

> nmcli connection down コマンドでは、デバイスからの接続は非アクティブ化されますが、その後デバイスが接続を自動的にアクティブ化することは禁じません。nmcli device disconnect コマンドでは、デバイスが切断され、手動の操作がない限りその後デバイスが接続を自動的にアクティブ化することはありません

とある。つまり、もともと NIC の設定は入っていて、nmcli device disconect で切断されていた可能性がある。この場合は nmclo device up を行うだけでインターフェイスが有効になるため nmcli connection add は必要としない。そのため今回は nmcli connection add をしなくても設定が行えたと考えられる。

4 実験 3: DHCP サービスの設定及び起動

4.1 実験目的・概要

本実験では、ルーターで DHCP サーバを動作させるために、DHCP サービスの各種設定を行い、実際に DHCP サーバが起動しているか確認を行う。DHCP とは TCP/IP 通信を行う際に必要な IP アドレスなどの設定を自動的にクライアントに設定するプロトコルであり、DHCP サーバはこれを利用するためのサーバであり、設定するアドレス等の範囲をあらかじめ設定する必要がある。

4.2 実験方法

本実験では machine1 で設定を行った。

4.2.1 稼働中サービスの確認

以下のコマンドを実行し、稼働中のサービスの確認を行った。

```
systemctl list-units –type=service
```

4.2.2 firewall の停止

以下のコマンドを実行し、一時的に firewall の停止を行った。

```
systemctl stop firewalld
systemctl status firewalld
```

4.2.3 DHCP サーバーのインストールと設定

1. DHCP 関連パッケージの確認と DHCP サーバのインストール以下のコマンドを実行し、DHCP 関連パッケージの確認と DHCP サーバのインストールを行った。

   ```
   yum list dhcp-
   yum info dhcp
   yum install dhcp
   ```

2. 設定ファイルの修正 DHCP サーバーの設定ファイル/etc/dhcp/dhcpd.conf を以下のように設定した。リース期間はデフォルト 1 時間 (3600 秒)、最大 1 日 (86400 秒)、machine2 のリース期間は 30 日 (2592000 秒) に設定した

```
default-lease-time 3600;
max-lease-time 86400;
option domain-name-servers 10.10.1.2;

subnet 192.168.150.0 netmask 255.255.255.0
range 192.168.150.100 192.168.150.250;
option routers 192.168.150.1;



subnet 192.168.200.0 netmask 255.255.255.0
range 192.168.200.100 192.168.200.250;
option routers 192.168.200.1;



host www2
hardware ethernet 94:C6:91:A8:C6:B9;
fixed-address 192.168.150.2;
default-lease-time 2592000;
```

4.2.4 DHCP サーバの再起動とブート時自動起動の設定

以下のコマンドを実行し、dhcp サーバの再起動及びブート時の自動起動の設定を行った。

```
systemctl restart dhcpd
systemctl enable dhcpd
```

4.3 実験結果

4.3.1 稼働中のサービスの確認

```
[root@icesc12 ~]# systemctl list-units --type=service
  UNIT                                      LOAD   ACTIVE SUB     DESCRIPTION
  auditd.service                            loaded active running Security Auditing Service
  crond.service                             loaded active running Command Scheduler
  dbus.service                              loaded active running D-Bus System Message Bus
  firewalld.service                         loaded active running firewalld - dynamic firewall
       daemon
  getty@tty1.service                        loaded active running Getty on tty1
  irqbalance.service                        loaded active running irqbalance daemon
  kdump.service                             loaded active exited  Crash recovery kernel arming
  kmod-static-nodes.service                 loaded active exited  Create list of required
       static device nodes for the current kernel
  lvm2-lvmetad.service                      loaded active running LVM2 metadata daemon
  lvm2-monitor.service                      loaded active exited  Monitoring of LVM2 mirrors,
       snapshots etc. using dmeventd or progress polling
```

```
  lvm2-pvscan@8:2.service                              loaded active exited  LVM2 PV scan on device 8:2
  network.service                                      loaded active exited  LSB: Bring up/down networking

NetworkManager-wait-online.service                   loaded failed failed  Network Manager Wait Online
  NetworkManager.service                               loaded active running Network Manager
  polkit.service                                       loaded active running Authorization Manager
  postfix.service                                      loaded active running Postfix Mail Transport Agent
  rhel-dmesg.service                                   loaded active exited  Dump dmesg to /var/log/dmesg
  rhel-domainname.service                              loaded active exited  Read and set NIS domainname
       from /etc/sysconfig/network
  rhel-import-state.service                            loaded active exited  Import network configuration
       from initramfs
  rhel-readonly.service                                loaded active exited  Configure read-only root
       support
  rsyslog.service                                      loaded active running System Logging Service
  sshd.service                                         loaded active running OpenSSH server daemon
  systemd-backlight@backlight:acpi_video0.service loaded active exited  Load/Save Screen Backlight
       Brightness of backlight:acpi_video0
  systemd-journal-flush.service                        loaded active exited  Flush Journal to Persistent
       Storage
  systemd-journald.service                             loaded active running Journal Service
  systemd-logind.service                               loaded active running Login Service
  systemd-random-seed.service                          loaded active exited  Load/Save Random Seed
  systemd-readahead-collect.service                    loaded active exited  Collect Read-Ahead Data
  systemd-readahead-replay.service                     loaded active exited  Replay Read-Ahead Data
  systemd-remount-fs.service                           loaded active exited  Remount Root and Kernel File
       Systems
  systemd-sysctl.service                               loaded active exited  Apply Kernel Variables
  systemd-tmpfiles-setup-dev.service                   loaded active exited  Create Static Device Nodes in
        /dev
  systemd-tmpfiles-setup.service                       loaded active exited  Create Volatile Files and
       Directories
  systemd-udev-trigger.service                         loaded active exited  udev Coldplug all Devices
  systemd-udevd.service                                loaded active running udev Kernel Device Manager
  systemd-update-utmp.service                          loaded active exited  Update UTMP about System Boot
       /Shutdown
  systemd-user-sessions.service                        loaded active exited  Permit User Sessions
  systemd-vconsole-setup.service                       loaded active exited  Setup Virtual Console
  tuned.service                                        loaded active running Dynamic System Tuning Daemon

LOAD   = Reflects whether the unit definition was properly loaded.
ACTIVE = The high-level unit activation state, i.e. generalization of SUB.
SUB    = The low-level unit activation state, values depend on unit type.

39 loaded units listed. Pass --all to see loaded but inactive units, too.
To show all installed unit files use 'systemctl list-unit-files'.
```

systemctl list-units –type=service は 1 列目に UNIT の名前、2 列目に LOAD されているか、3 列目にそれが active であるか、4 列目に UNIT のタイプによる active の状態、5 列目にその UNIT の説明がされている。NetworkManager-wait-online.service のみ active でなく他 38 個の UNIT はすべて actibe であることが分かった。また、firewalld.service loaded active running firewalld - dynamic firewall daemon により、ファイアウォールが起動していることが分かった。

4.3.2 ファイアウォールの停止後の設定状況

```
[root@icesc12 ~]# systemctl status firewalld
firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; enabled; vendor preset: enabled)
   Active: inactive (dead) since Thu 2019-05-16 14:37:59 JST; 6s ago
     Docs: man:firewalld(1)
  Process: 706 ExecStart=/usr/sbin/firewalld --nofork --nopid $FIREWALLD_ARGS (code=exited, status=0/
       SUCCESS)
 Main PID: 706 (code=exited, status=0/SUCCESS)

May 16 14:25:52 icesc12.ice.nuie.nagoya-u.ac.jp systemd[1]: Starting firewalld - dynamic firewall
     daemon...
May 16 14:25:53 icesc12.ice.nuie.nagoya-u.ac.jp systemd[1]: Started firewalld - dynamic firewall
     daemon.
May 16 14:37:58 icesc12.ice.nuie.nagoya-u.ac.jp systemd[1]: Stopping firewalld - dynamic firewall
     daemon...
```

```
May 16 14:37:59 icesc12.ice.nuie.nagoya-u.ac.jp systemd[1]: Stopped firewalld - dynamic firewall
     daemon.
```

ファイアウォールのデーモンファイルが読み込まれてはいるが、Thu 2019-05-16 14:37:59 JST; 6s ago から inactive になっていることがわかった。これは先ほど起動したためである。

4.3.3 DHCP 関連パッケージの確認

```
[root@icesc12 ~]# yum info dhcp
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
 * base: ftp.jaist.ac.jp
 * extras: ftp.jaist.ac.jp
 * updates: ftp.jaist.ac.jp
Available Packages
Name        : dhcp
Arch        : x86_64
Epoch       : 12
Version     : 4.2.5
Release     : 68.el7.centos.1
Size        : 513 k
Repo        : base/7/x86_64
Summary     : Dynamic host configuration protocol software
URL         : http://isc.org/products/DHCP/
License     : ISC
Description : DHCP (Dynamic Host Configuration Protocol) is a protocol which allows
            : individual devices on an IP network to get their own network
            : configuration information (IP address, subnetmask, broadcast address,
            : etc.) from a DHCP server. The overall purpose of DHCP is to make it
            : easier to administer a large network.
            :
            : To use DHCP on your network, install a DHCP service (or relay agent),
            : and on clients run a DHCP client daemon.  The dhcp package provides
            : the ISC DHCP service and relay agent.
```

dhcp サーバのバージョンが 4.2.5 であり、size は 513 k のサーバファイルであることが分かった。

4.4 考察

faild になっていた NetworkManager-wait-online.service がどのようなときに使われうるのか考察した。NetworkManager-wait-online.service は設定されているタイムアウト時間 (デフォルト 30 秒) を超えても NetworkManager の準備が完了していない場合に待つという処理を行う。つまり、IP アドレスを所得できていない段階で起動しているためにエラーを出すサービスがある場合はこれを enable にする必要があると考えられる。

5 実験 4: ファイヤーウォールの設定

5.1 実験目的・概要

本実験では、サーバを外部の不正アクセスから保護ために、machine1、machine2 のファイアウォール設定をする。

5.2 実験方法

- machine1
  1. パケット転送の有効化

以下のコマンドを実行し設定ファイル/etc/sysctl.d/10-ipv4.conf を作成し内容を追加し、有効化した。

```
echo net.ipv4.ip_forward=1 ¿ /etc/sysctl.d/10-ipv4.conf
sysctl –system
reboot
```

2. firewalld から iptables-sevices への変更

   Netfilter のフロントエンドを以下のコマンドを実行し、firewalld から iptables に変更した。

```
systemctl stop firewalld
systemctl disable firewalld
yum install iptables-services
systemctl start iptables
systemctl status iptables
systemctl enable iptables
```

3. サンプルの実行

   以下のコマンドを実行し、ICE のホスト ssh のディレクトリにある iptables の設定サンプルファイルを scp でコピーし実行した。

```
scp @ssh.ice.nuie.nagoya-u.ac.jp:/pub1/jikken/cs-net/iptables-sanpke.sh
sh iptablse-sample.sh
```

4. iptables の設定状況を確認以下のコマンドを実行し、iptables の設定状況と現在のルールを確認した。

```
iptables -L -n
iptable-save
```

5. 設定スクリプトの修正

   iptables.sh を修正し以下のように設定スクリプトを修正した。

```
#!/bin/sh

PATH=/sbin:/bin:/usr/bin:/usr/sbin

## 変数の定義
EXTERNAL_INTERFACE="enp1s0"        # 外側インタフェースの名前
DMZ_INTERFACE="enp2s0"             # DMZ インタフェースの名前
INTERNAL_INTERFACE="enp3s0"        # 内側インタフェースの名前

# 外側インタフェースのアドレスIP
IPADDR=`ip addr show $EXTERNAL_INTERFACE | \
 sed -e 's/^.*inet \([^ \/]*\).*$/\1/p' -e d`
# 内部ネットワーク・アドレス
INTERNAL_LAN=`ip addr show $INTERNAL_INTERFACE | \
 sed -e 's/^.*inet \([^ ]*\).*$/\1/p' -e d`

# ネットワーク・アドレスDMZ
DMZ_LAN=`ip addr show $DMZ_INTERFACE | \
 sed -e 's/^.*inet \([^ ]*\).*$/\1/p' -e d`
```

```
ANYWHERE="0.0.0.0/0"

## 以下の設定を実行している間はパケットの転送を停止する
echo 0 > /proc/sys/net/ipv4/ip_forward

## すでに設定されているルールを消去する
iptables -F
iptables -F -t nat

## ポリシーの初期設定 -> しない場合の扱いmatch
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP

## ループバック・インタフェースの入出力を許可する
iptables -A INPUT -i lo -j ACCEPT
iptables -A OUTPUT -o lo -j ACCEPT

#####################################################################
##
## チェーンの設定（デフォルト拒否）INPUT
##

iptables -A INPUT -i $EXTERNAL_INTERFACE -p tcp -m state --state NEW -m tcp \
    --dport 22 -j ACCEPT # Ext -> Router: SSH

iptables -A INPUT -i $INTERNAL_INTERFACE -p tcp -m state --state NEW -m tcp \
    --dport 22 -j ACCEPT # Local -> Router: SSH
iptables -A INPUT -i $INTERNAL_INTERFACE -p udp -m state --state NEW -m udp \
    --dport 67 -j ACCEPT # Local -> Router: DHCP server
iptables -A INPUT -i $INTERNAL_INTERFACE -p icmp -j ACCEPT # Local -> Router Ping

iptables -A INPUT -i $DMZ_INTERFACE -p udp -m state --state NEW -m udp \
    --dport 67 -j ACCEPT # DMZ -> Router: DHCP server
iptables -A INPUT -i $DMZ_INTERFACE -p icmp -j ACCEPT # DMZ -> Router: Ping

iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT


#####################################################################
##
## チェーンの設定（デフォルト拒否）OUTPUT
##

iptables -A OUTPUT -o $EXTERNAL_INTERFACE -p tcp -m state --state NEW -m tcp \
    --dport 22 -j ACCEPT # Router -> Ext: SSH
iptables -A OUTPUT -o $EXTERNAL_INTERFACE -p tcp -m state --state NEW -m tcp \
    --dport 80 -j ACCEPT # Router -> Ext: HTTP
iptables -A OUTPUT -o $EXTERNAL_INTERFACE -p tcp -m state --state NEW -m tcp \
    --dport 443 -j ACCEPT # Router -> Ext: HTTPS
iptables -A OUTPUT -o $EXTERNAL_INTERFACE -p udp -m state --state NEW -m udp \
    --dport 53 -j ACCEPT # Router -> Ext: DNS
iptables -A OUTPUT -o $EXTERNAL_INTERFACE -p icmp -j ACCEPT # Router -> Ext: Ping

iptables -A OUTPUT -o $DMZ_INTERFACE -p tcp -m state --state NEW -m tcp \
    --dport 22 -j ACCEPT # Router -> DMZ: SSH
iptables -A OUTPUT -o $DMZ_INTERFACE -p udp -m state --state NEW -m udp \
    --dport 68 -j ACCEPT # Router -> DMZ: DHCP client

iptables -A OUTPUT -o $INTERNAL_INTERFACE -p udp -m state --state NEW -m udp \
    --dport 68 -j ACCEPT # Router -> Local: DHCP client
iptables -A OUTPUT -o $INTERNAL_INTERFACE -p icmp -j ACCEPT # Router -> Local: Ping

iptables -A OUTPUT -m state --state RELATED,ESTABLISHED -j ACCEPT


#####################################################################
##
## チェーンの設定（デフォルト拒否）FORWARD
##

iptables -A FORWARD -i $EXTERNAL_INTERFACE -o $DMZ_INTERFACE -p tcp -m tcp \
    --dport 80 -j ACCEPT # Ext -> DMZ: HTTP
iptables -A FORWARD -i $EXTERNAL_INTERFACE -o $DMZ_INTERFACE -p tcp -m tcp \
    --dport 443 -j ACCEPT # Ext -> DMZ: HTTPS
iptables -A FORWARD -i $DMZ_INTERFACE -o $EXTERNAL_INTERFACE \
    -m state --state RELATED,ESTABLISHED -j ACCEPT
```

```
iptables -A FORWARD -i $DMZ_INTERFACE -o $EXTERNAL_INTERFACE -p tcp -m tcp \
    --dport 22 -j ACCEPT # DMZ -> Ext: SSH
iptables -A FORWARD -i $DMZ_INTERFACE -o $EXTERNAL_INTERFACE -p tcp -m tcp \
    --dport 80 -j ACCEPT # DMZ -> Ext: HTTP
iptables -A FORWARD -i $DMZ_INTERFACE -o $EXTERNAL_INTERFACE -p tcp -m tcp \
    --dport 443 -j ACCEPT # DMZ -> Ext: HTTPS
iptables -A FORWARD -i $DMZ_INTERFACE -o $EXTERNAL_INTERFACE -p udp -m udp \
    --dport 53 -j ACCEPT # DMZ -> Ext: DNS
iptables -A FORWARD -i $DMZ_INTERFACE -o $EXTERNAL_INTERFACE -p icmp \
    -j ACCEPT # DMZ -> Ext: Ping
iptables -A FORWARD -i $EXTERNAL_INTERFACE -o $DMZ_INTERFACE \
    -m state --state RELATED,ESTABLISHED -j ACCEPT

iptables -A FORWARD -i $INTERNAL_INTERFACE -o $EXTERNAL_INTERFACE -p tcp \
    -m state --state NEW,ESTABLISHED -m tcp --dport 22 -j ACCEPT # Local -> Ext: SSH
iptables -A FORWARD -i $INTERNAL_INTERFACE -o $EXTERNAL_INTERFACE -p tcp \
    -m state --state NEW,ESTABLISHED -m tcp --dport 80 -j ACCEPT # Local -> Ext: HTTP
iptables -A FORWARD -i $INTERNAL_INTERFACE -o $EXTERNAL_INTERFACE -p tcp \
    -m state --state NEW,ESTABLISHED -m tcp --dport 443 -j ACCEPT # Local -> Ext: HTTPS
iptables -A FORWARD -i $INTERNAL_INTERFACE -o $EXTERNAL_INTERFACE -p udp \
    -m state --state NEW,ESTABLISHED -m udp --dport 53 -j ACCEPT # Local -> Ext: DNS
iptables -A FORWARD -i $INTERNAL_INTERFACE -o $EXTERNAL_INTERFACE -p icmp \
    -j ACCEPT # Local -> Ext: Ping
iptables -A FORWARD -i $EXTERNAL_INTERFACE -o $INTERNAL_INTERFACE \
    -m state --state RELATED,ESTABLISHED -j ACCEPT

iptables -A FORWARD -i $INTERNAL_INTERFACE -o $DMZ_INTERFACE -j ACCEPT # Local -> DMZ
iptables -A FORWARD -i $DMZ_INTERFACE -o $INTERNAL_INTERFACE \
    -m state --state RELATED,ESTABLISHED -j ACCEPT

#####################################################################
##
## の設定NAT
##

iptables -A POSTROUTING -t nat -s $INTERNAL_LAN -o $EXTERNAL_INTERFACE -j SNAT \
    --to-source $IPADDR
iptables -A POSTROUTING -t nat -s $DMZ_LAN -o $EXTERNAL_INTERFACE -j SNAT \
    --to-source $IPADDR

iptables -A PREROUTING -t nat -i $EXTERNAL_INTERFACE -p tcp -m tcp \
    --dport 80 -j DNAT --to-destination 192.168.150.2:80
iptables -A PREROUTING -t nat -i $EXTERNAL_INTERFACE -p tcp -m tcp \
    --dport 443 -j DNAT --to-destination 192.168.150.2:443

#####################################################################
##
## 設定の保存
##
#/etc/init.d/iptables save active

## パケットの転送を開始する
echo 1 > /proc/sys/net/ipv4/ip_forward

exit 0
```

6. 確認用ソフトウェアのインストール

   以下のコマンド machine を実行し、ファイアウォール設定も確認時に利用するクライアントソフトウェアとして、WWW コンソールブラウザ lynx、DNS クライアント nslookup、dig をインストールした。

   yum info lynx
   yum install lynx
   yum info bind-utils
   yum install bind-utils
   rpm -ql bind-utils — grep /usr/bin

7. ファイアウォールの動作状況の確認

machine1、2、3 にて外部ネットワークへの接続及び内部間での接続状況について以下のことを確認した。

- machine1 から外部ネットワークへの接続
- machine2 から外部ネットワークへの接続
- machine3 から外部ネットワークへの接続
- machine3 から machine1 への接続
- machine3 から machine2 への接続
- machine2 から machine3 への接続
- 外部ネットワークから machine1 への接続

8. 最終状態を保存

   以下のコマンドを実行し、最終状態の保存をした。

   ```
   iptables-save ¿ /etc/sysconfig/iptables
   ```

- machine2

1. firewalld の動作状態の確認

   以下のコマンドを実行し、firewalld の動作状態の確認をした

   ```
   systemctl status firewalld
   ```

2. 現在のゾーンの許可状態を確認

   以下のコマンドを実行し、現在のゾーンの許可状態を確認した。

   ```
   firewalld -cmd –get-active -zones
   ```

3. 必要なサービス許可設定の追加

   以下のコマンドを実行し、必要なサービスの許可設定の追加を行い、デバイスの再起動を行った。

   ```
   firewall-cmd –add-service=dhcpv6-client
   firewall-cmd –add-service=http
   firewall-cmd –add-service=https
   firewall-cmd –list-all
   nmcli con down enp3s0
   nmcli con up enp3s0
   ```

4. 動作確認

   SSH、HTTP/HTTPS、DHCPv6client、すべての ICMP パケットのみ許可し、その他の接続は全て拒否 されることを確認した。

5. 各設定の永続化

   以下のコマンドを実行し各設定の永続化を行った。

```
firewall-cmd –permanent –add-service=dhcpv6-client
firewall-cmd –permanent –add-service=http
firewall-cmd –permanent –add-service=https
```

6. ファイアウォールの再起動
   以下のコマンドを実行し、ファイアウォールの再起動を行った。

```
systemctl restart firewalld
```

5.3 実験結果

- machine1

5.3.1 パケット転送の有効化

```
[root@icesc12 ~]# sysctl --system
* Applying /usr/lib/sysctl.d/00-system.conf ...
* Applying /usr/lib/sysctl.d/10-default-yama-scope.conf ...
kernel.yama.ptrace_scope = 0
* Applying /etc/sysctl.d/10-ipv4.conf ...
net.ipv4.ip_forward = 1
* Applying /usr/lib/sysctl.d/50-default.conf ...
kernel.sysrq = 16
kernel.core_uses_pid = 1
net.ipv4.conf.default.rp_filter = 1
net.ipv4.conf.all.rp_filter = 1
net.ipv4.conf.default.accept_source_route = 0
net.ipv4.conf.all.accept_source_route = 0
net.ipv4.conf.default.promote_secondaries = 1
net.ipv4.conf.all.promote_secondaries = 1
fs.protected_hardlinks = 1
fs.protected_symlinks = 1
* Applying /etc/sysctl.d/99-sysctl.conf ...
* Applying /etc/sysctl.conf ...
```

より sysctl が/usr/lib/sysctl.d/00-system.conf 、/usr/lib/sysctl.d/10-default -yama -scope.conf、/etc/sysctl.d/10-ipv4.conf、 /etc/sysctl.d/10-ipv4.conf 、/usr/lib/sysctl.d/50-default.conf、/etc/sysctl.d/99-sysctl.conf、/etc/sysctl.conf から ip の設定を呼び出していて、net.ipv4.ip_forward = 1 が反映されていることが確認できた。

5.3.2 firewalld から iptables-services への変更

```
[root@icesc12 ~]# systemctl status iptables
 iptables.service - IPv4 firewall with iptables
   Loaded: loaded (/usr/lib/systemd/system/iptables.service; disabled; vendor preset: disabled)
   Active: active (exited) since Thu 2019-05-23 14:09:01 JST; 6s ago
  Process: 1575 ExecStart=/usr/libexec/iptables/iptables.init start (code=exited, status=0/
      SUCCESS)
 Main PID: 1575 (code=exited, status=0/SUCCESS)

May 23 14:09:00 icesc12.ice.nuie.nagoya-u.ac.jp systemd[1]: Starting IPv4 firewall with
     iptables...
May 23 14:09:01 icesc12.ice.nuie.nagoya-u.ac.jp iptables.init[1575]: iptables: Applying
     firewall rules: [  OK  ]
May 23 14:09:01 icesc12.ice.nuie.nagoya-u.ac.jp systemd[1]: Started IPv4 firewall with iptables
     .
```

より iptables が読み込まれていて、Thu 2019-05-23 14:09:01 JST; 6s ago から active になっていることが確認できた。

5.4 iptables の設定状況の確認

```
[root@icesc12 ~]# iptables -L -n
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:22
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:22
ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           state NEW udp dpt:67
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0
ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           state NEW udp dpt:67
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED

Chain FORWARD (policy DROP)
target     prot opt source               destination
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW,ESTABLISHED tcp dpt:22
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED

Chain OUTPUT (policy DROP)
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:22
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:80
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:443
ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           state NEW udp dpt:53
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:22
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
[root@icesc12 ~]# iptables-save
# Generated by iptables-save v1.4.21 on Thu May 23 14:11:54 2019
*nat
:PREROUTING ACCEPT [0:0]
:INPUT ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
:POSTROUTING ACCEPT [0:0]
-A POSTROUTING -s 192.168.200.0/24 -o enp1s0 -j SNAT --to-source 192.168.100.12
COMMIT
# Completed on Thu May 23 14:11:54 2019
# Generated by iptables-save v1.4.21 on Thu May 23 14:11:54 2019
*filter
:INPUT DROP [0:0]
:FORWARD DROP [0:0]
:OUTPUT DROP [0:0]
-A INPUT -i lo -j ACCEPT
-A INPUT -i enp1s0 -p tcp -m state --state NEW -m tcp --dport 22 -j ACCEPT
-A INPUT -i enp3s0 -p tcp -m state --state NEW -m tcp --dport 22 -j ACCEPT
-A INPUT -i enp3s0 -p udp -m state --state NEW -m udp --dport 67 -j ACCEPT
-A INPUT -i enp3s0 -p icmp -j ACCEPT
-A INPUT -i enp2s0 -p udp -m state --state NEW -m udp --dport 67 -j ACCEPT
-A INPUT -i enp2s0 -p icmp -j ACCEPT
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A FORWARD -i enp3s0 -o enp1s0 -p tcp -m state --state NEW,ESTABLISHED -m tcp --dport 22 -j
    ACCEPT
-A FORWARD -i enp3s0 -o enp1s0 -p icmp -j ACCEPT
-A FORWARD -i enp1s0 -o enp3s0 -m state --state RELATED,ESTABLISHED -j ACCEPT
-A OUTPUT -o lo -j ACCEPT
-A OUTPUT -o enp1s0 -p tcp -m state --state NEW -m tcp --dport 22 -j ACCEPT
-A OUTPUT -o enp1s0 -p tcp -m state --state NEW -m tcp --dport 80 -j ACCEPT
-A OUTPUT -o enp1s0 -p tcp -m state --state NEW -m tcp --dport 443 -j ACCEPT
-A OUTPUT -o enp1s0 -p udp -m state --state NEW -m udp --dport 53 -j ACCEPT
-A OUTPUT -o enp1s0 -p icmp -j ACCEPT
-A OUTPUT -o enp3s0 -p icmp -j ACCEPT
-A OUTPUT -o enp2s0 -p tcp -m state --state NEW -m tcp --dport 22 -j ACCEPT
-A OUTPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
COMMIT
```

iptables-sample.sh で設定されている仕様を確認した。

5.4.1 設定スクリプトの修正

```
#!/bin/sh

PATH=/sbin:/bin:/usr/bin:/usr/sbin

## 変数の定義
EXTERNAL_INTERFACE="enp1s0"        # 外側インタフェースの名前
DMZ_INTERFACE="enp2s0"             # DMZ インタフェースの名前
INTERNAL_INTERFACE="enp3s0"        # 内側インタフェースの名前

# 外側インタフェースのアドレスIP
IPADDR=`ip addr show $EXTERNAL_INTERFACE | \
 sed -e 's/^.*inet \([^ \/]*\).*$/\1/p' -e d`
# 内部ネットワーク・アドレス
INTERNAL_LAN=`ip addr show $INTERNAL_INTERFACE | \
 sed -e 's/^.*inet \([^ ]*\).*$/\1/p' -e d`

# ネットワーク・アドレスDMZ
DMZ_LAN=`ip addr show $DMZ_INTERFACE | \
 sed -e 's/^.*inet \([^ ]*\).*$/\1/p' -e d`

ANYWHERE="0.0.0.0/0"

## 以下の設定を実行している間はパケットの転送を停止する
echo 0 > /proc/sys/net/ipv4/ip_forward

## すでに設定されているルールを消去する
iptables -F
iptables -F -t nat

## ポリシーの初期設定 -> しない場合の扱いmatch
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP

## ループバック・インタフェースの入出力を許可する
iptables -A INPUT -i lo -j ACCEPT
iptables -A OUTPUT -o lo -j ACCEPT

################################################################
##
## チェーンの設定（デフォルト拒否）INPUT
##

iptables -A INPUT -i $EXTERNAL_INTERFACE -p tcp -m state --state NEW -m tcp \
    --dport 22 -j ACCEPT # Ext -> Router: SSH

iptables -A INPUT -i $INTERNAL_INTERFACE -p tcp -m state --state NEW -m tcp \
    --dport 22 -j ACCEPT # Local -> Router: SSH
iptables -A INPUT -i $INTERNAL_INTERFACE -p udp -m state --state NEW -m udp \
    --dport 67 -j ACCEPT # Local -> Router: DHCP server
iptables -A INPUT -i $INTERNAL_INTERFACE -p icmp -j ACCEPT # Local -> Router Ping

iptables -A INPUT -i $DMZ_INTERFACE -p udp -m state --state NEW -m udp \
    --dport 67 -j ACCEPT # DMZ -> Router: DHCP server
iptables -A INPUT -i $DMZ_INTERFACE -p icmp -j ACCEPT # DMZ -> Router: Ping

iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT


################################################################
##
## チェーンの設定（デフォルト拒否）OUTPUT
##

iptables -A OUTPUT -o $EXTERNAL_INTERFACE -p tcp -m state --state NEW -m tcp \
    --dport 22 -j ACCEPT # Router -> Ext: SSH
iptables -A OUTPUT -o $EXTERNAL_INTERFACE -p tcp -m state --state NEW -m tcp \
```

```
    --dport 80 -j ACCEPT # Router -> Ext: HTTP
iptables -A OUTPUT -o $EXTERNAL_INTERFACE -p tcp -m state --state NEW -m tcp \
    --dport 443 -j ACCEPT # Router -> Ext: HTTPS
iptables -A OUTPUT -o $EXTERNAL_INTERFACE -p udp -m state --state NEW -m udp \
    --dport 53 -j ACCEPT # Router -> Ext: DNS
iptables -A OUTPUT -o $EXTERNAL_INTERFACE -p icmp -j ACCEPT # Router -> Ext: Ping

iptables -A OUTPUT -o $DMZ_INTERFACE -p tcp -m state --state NEW -m tcp \
    --dport 22 -j ACCEPT # Router -> DMZ: SSH
iptables -A OUTPUT -o $DMZ_INTERFACE -p udp -m state --state NEW -m udp \
    --dport 68 -j ACCEPT # Router -> DMZ: DHCP client

iptables -A OUTPUT -o $INTERNAL_INTERFACE -p udp -m state --state NEW -m udp \
    --dport 68 -j ACCEPT # Router -> Local: DHCP client
iptables -A OUTPUT -o $INTERNAL_INTERFACE -p icmp -j ACCEPT # Router -> Local: Ping

iptables -A OUTPUT -m state --state RELATED,ESTABLISHED -j ACCEPT


############################################################
##
## チェーンの設定（デフォルト拒否）FORWARD
##

iptables -A FORWARD -i $EXTERNAL_INTERFACE -o $DMZ_INTERFACE -p tcp -m tcp \
    --dport 80 -j ACCEPT # Ext -> DMZ: HTTP
iptables -A FORWARD -i $EXTERNAL_INTERFACE -o $DMZ_INTERFACE -p tcp -m tcp \
    --dport 443 -j ACCEPT # Ext -> DMZ: HTTPS
iptables -A FORWARD -i $DMZ_INTERFACE -o $EXTERNAL_INTERFACE \
    -m state --state RELATED,ESTABLISHED -j ACCEPT

iptables -A FORWARD -i $DMZ_INTERFACE -o $EXTERNAL_INTERFACE -p tcp -m tcp \
    --dport 22 -j ACCEPT # DMZ -> Ext: SSH
iptables -A FORWARD -i $DMZ_INTERFACE -o $EXTERNAL_INTERFACE -p tcp -m tcp \
    --dport 80 -j ACCEPT # DMZ -> Ext: HTTP
iptables -A FORWARD -i $DMZ_INTERFACE -o $EXTERNAL_INTERFACE -p tcp -m tcp \
    --dport 443 -j ACCEPT # DMZ -> Ext: HTTPS
iptables -A FORWARD -i $DMZ_INTERFACE -o $EXTERNAL_INTERFACE -p udp -m udp \
    --dport 53 -j ACCEPT # DMZ -> Ext: DNS
iptables -A FORWARD -i $DMZ_INTERFACE -o $EXTERNAL_INTERFACE -p icmp \
    -j ACCEPT # DMZ -> Ext: Ping
iptables -A FORWARD -i $EXTERNAL_INTERFACE -o $DMZ_INTERFACE \
    -m state --state RELATED,ESTABLISHED -j ACCEPT

iptables -A FORWARD -i $INTERNAL_INTERFACE -o $EXTERNAL_INTERFACE -p tcp \
    -m state --state NEW,ESTABLISHED -m tcp --dport 22 -j ACCEPT # Local -> Ext: SSH
iptables -A FORWARD -i $INTERNAL_INTERFACE -o $EXTERNAL_INTERFACE -p tcp \
    -m state --state NEW,ESTABLISHED -m tcp --dport 80 -j ACCEPT # Local -> Ext: HTTP
iptables -A FORWARD -i $INTERNAL_INTERFACE -o $EXTERNAL_INTERFACE -p tcp \
    -m state --state NEW,ESTABLISHED -m tcp --dport 443 -j ACCEPT # Local -> Ext: HTTPS
iptables -A FORWARD -i $INTERNAL_INTERFACE -o $EXTERNAL_INTERFACE -p udp \
    -m state --state NEW,ESTABLISHED -m udp --dport 53 -j ACCEPT # Local -> Ext: DNS
iptables -A FORWARD -i $INTERNAL_INTERFACE -o $EXTERNAL_INTERFACE -p icmp \
    -j ACCEPT # Local -> Ext: Ping
iptables -A FORWARD -i $EXTERNAL_INTERFACE -o $INTERNAL_INTERFACE \
    -m state --state RELATED,ESTABLISHED -j ACCEPT

iptables -A FORWARD -i $INTERNAL_INTERFACE -o $DMZ_INTERFACE -j ACCEPT # Local -> DMZ
iptables -A FORWARD -i $DMZ_INTERFACE -o $INTERNAL_INTERFACE \
    -m state --state RELATED,ESTABLISHED -j ACCEPT

############################################################
##
## 設定NAT
##

iptables -A POSTROUTING -t nat -s $INTERNAL_LAN -o $EXTERNAL_INTERFACE -j SNAT \
    --to-source $IPADDR
iptables -A POSTROUTING -t nat -s $DMZ_LAN -o $EXTERNAL_INTERFACE -j SNAT \
    --to-source $IPADDR

iptables -A PREROUTING -t nat -i $EXTERNAL_INTERFACE -p tcp -m tcp \
    --dport 80 -j DNAT --to-destination 192.168.150.2:80
iptables -A PREROUTING -t nat -i $EXTERNAL_INTERFACE -p tcp -m tcp \
    --dport 443 -j DNAT --to-destination 192.168.150.2:443

############################################################
##
## 設定の保存
```

```
##
#/etc/init.d/iptables save active

## パケットの転送を開始する
echo 1 > /proc/sys/net/ipv4/ip_forward

exit 0
```

要求仕様ごとに iptables の設定を分解すると以下の通りになる。

```
iptables -P INPUT DROP\\
iptables -P OUTPUT DROP\\
iptables -P FORWARD DROP
```

これにより INPUT/OUTPUT/FORWARD チェーンの初期設定をすべて DROP にする。

```
iptables -A INPUT -i $EXTERNAL_INTERFACE -p tcp -m state --state NEW -m tcp \
    --dport 22 -j ACCEPT # Ext -> Router: SSH
```

これによりルータに入ってくる通信について、外部からの tcp をつかう SSH プロトコルで利用するポート番号 22 の通信について新しく設定した通信においては許可をする。

```
iptables -A INPUT -i $INTERNAL_INTERFACE -p tcp -m state --state NEW -m tcp \
    --dport 22 -j ACCEPT # Local -> Router: SSH
iptables -A INPUT -i $INTERNAL_INTERFACE -p udp -m state --state NEW -m udp \
    --dport 67 -j ACCEPT # Local -> Router: DHCP server
iptables -A INPUT -i $INTERNAL_INTERFACE -p icmp -j ACCEPT # Local -> Router Ping
```

これによりルータに入ってくる通信についてローカルネットワークからの tcp をつかうポート番号 22(SSH)、udp を使うポート番号 67(DHCP)、icmp を使う通信 (Ping) について許可をする。

```
iptables -A INPUT -i $DMZ_INTERFACE -p udp -m state --state NEW -m udp \
    --dport 67 -j ACCEPT # DMZ -> Router: DHCP server
iptables -A INPUT -i $DMZ_INTERFACE -p icmp -j ACCEPT # DMZ -> Router: Ping
```

これによりルータに入ってくる通信について DMZ ネットワークからの udp を使うポート番号 67(DHCP サーバ)、icmp を使う通信 (Ping) について許可をする。DHCP サーバがルータであるため 67 番を使う。

```
iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
```

これによりルータに入ってくる通信について既に確立した、もしくは新しく許可されている通信に対しては許可をすることで一度許した通信については許可をする。

```
iptables -A OUTPUT -o $EXTERNAL_INTERFACE -p tcp -m state --state NEW -m tcp \
    --dport 22 -j ACCEPT # Router -> Ext: SSH
iptables -A OUTPUT -o $EXTERNAL_INTERFACE -p tcp -m state --state NEW -m tcp \
    --dport 80 -j ACCEPT # Router -> Ext: HTTP
iptables -A OUTPUT -o $EXTERNAL_INTERFACE -p tcp -m state --state NEW -m tcp \
    --dport 443 -j ACCEPT # Router -> Ext: HTTPS
iptables -A OUTPUT -o $EXTERNAL_INTERFACE -p udp -m state --state NEW -m udp \
    --dport 53 -j ACCEPT # Router -> Ext: DNS
iptables -A OUTPUT -o $EXTERNAL_INTERFACE -p icmp -j ACCEPT # Router -> Ext: Ping
```

これによりルータから出ていく通信について外部への通信は tcp をつかうポート番号 22(SSH)、80(HTTP)、443(HTTPS)、udp を使うポート番号 53(DNS)、icmp を使う通信 (Ping) ついて許可をする。

```
iptables -A OUTPUT -o $DMZ_INTERFACE -p tcp -m state --state NEW -m tcp \
    --dport 22 -j ACCEPT # Router -> DMZ: SSH
iptables -A OUTPUT -o $DMZ_INTERFACE -p udp -m state --state NEW -m udp \
    --dport 68 -j ACCEPT # Router -> DMZ: DHCP client
```

これによりルータから出ていく通信について DMZ への通信は tcp を使うポート番号 22(SSH)、udp を使うポート番号 68(DHCP クライアント) の通信について許可をする。DMZ はルータから DHCP により IP をもらうクライアントであるため 68 を使う。

```
iptables -A OUTPUT -o $INTERNAL_INTERFACE -p udp -m state --state NEW -m udp \
    --dport 68 -j ACCEPT # Router -> Local: DHCP client
iptables -A OUTPUT -o $INTERNAL_INTERFACE -p icmp -j ACCEPT # Router -> Local: Ping
```

これによりルータから出ていく通信についてローカルネットワークへの通信は udp を使うポート番号 68(DHCP クライアント)、icmp を使う通信 (Ping) について許可をする。ローカルネットワークはルータから DHCP により IP をもらうクライアントであるため 68 を使う。

```
iptables -A OUTPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
```

これによりルータから出ていく通信について既に確立した、もしくは新しく許可されている通信に対しては許可をすることで一度許した通信については許可をする。

```
iptables -A FORWARD -i $INTERNAL_INTERFACE -o $EXTERNAL_INTERFACE -p tcp \
    -m state --state NEW,ESTABLISHED -m tcp --dport 22 -j ACCEPT # Local -> Ext: SSH
iptables -A FORWARD -i $INTERNAL_INTERFACE -o $EXTERNAL_INTERFACE -p tcp \
    -m state --state NEW,ESTABLISHED -m tcp --dport 80 -j ACCEPT # Local -> Ext: HTTP
iptables -A FORWARD -i $INTERNAL_INTERFACE -o $EXTERNAL_INTERFACE -p tcp \
    -m state --state NEW,ESTABLISHED -m tcp --dport 443 -j ACCEPT # Local -> Ext: HTTPS
iptables -A FORWARD -i $INTERNAL_INTERFACE -o $EXTERNAL_INTERFACE -p udp \
    -m state --state NEW,ESTABLISHED -m udp --dport 53 -j ACCEPT # Local -> Ext: DNS
iptables -A FORWARD -i $INTERNAL_INTERFACE -o $EXTERNAL_INTERFACE -p icmp \
    -j ACCEPT # Local -> Ext: Ping
iptables -A FORWARD -i $EXTERNAL_INTERFACE -o $INTERNAL_INTERFACE \
    -m state --state RELATED,ESTABLISHED -j ACCEPT
```

これによりパケット通信について、ローカルネットワークから外部ネットワークへの通信において tcp をつかうポート番号 22(SSH)、80(HTTP)、443(HTTPS)、udp を使うポート番号 53(DNS)、icmp を使う通信 (Ping) ついて許可をする。また、受信返答などに用いられる通信においては外部からローカルネットワークへの通信について既に確立した、もしくは新しく許可されている通信に対しては許可をすることで一度許した通信については許可をする。

```
iptables -A FORWARD -i $DMZ_INTERFACE -o $EXTERNAL_INTERFACE -p tcp -m tcp \
    --dport 22 -j ACCEPT # DMZ -> Ext: SSH
iptables -A FORWARD -i $DMZ_INTERFACE -o $EXTERNAL_INTERFACE -p tcp -m tcp \
    --dport 80 -j ACCEPT # DMZ -> Ext: HTTP
iptables -A FORWARD -i $DMZ_INTERFACE -o $EXTERNAL_INTERFACE -p tcp -m tcp \
    --dport 443 -j ACCEPT # DMZ -> Ext: HTTPS
iptables -A FORWARD -i $DMZ_INTERFACE -o $EXTERNAL_INTERFACE -p udp -m udp \
    --dport 53 -j ACCEPT # DMZ -> Ext: DNS
iptables -A FORWARD -i $DMZ_INTERFACE -o $EXTERNAL_INTERFACE -p icmp \
    -j ACCEPT # DMZ -> Ext: Ping
iptables -A FORWARD -i $EXTERNAL_INTERFACE -o $DMZ_INTERFACE \
    -m state --state RELATED,ESTABLISHED -j ACCEPT
```

これによりパケット通信について、DMZ から外部ネットワークへの通信において tcp をつかうポート番号 22(SSH)、80(HTTP)、443(HTTPS)、udp を使うポート番号 53(DNS)、icmp を使う通信 (Ping) ついて許可をする。また、受信返答などに用いられる通信においては外部からローカルネットワークへ

の通信について既に確立した、もしくは新しく許可されている通信に対しては許可をすることで一度許した通信については許可をする。

```
iptables -A FORWARD -i $INTERNAL_INTERFACE -o $DMZ_INTERFACE -j ACCEPT # Local -> DMZ
iptables -A FORWARD -i $DMZ_INTERFACE -o $INTERNAL_INTERFACE \
    -m state --state RELATED,ESTABLISHED -j ACCEPT
```

これによりパケット通信においてネットワークから DMZ への通信において tcp をつかうポート番号 80(HTTP)、443(HTTPS) の通信を許可する。また、受信返答などに用いられる通信においては外部からローカルネットワークへの通信について既に確立した、もしくは新しく許可されている通信に対しては許可をすることで一度許した通信については許可をする。

```
iptables -A FORWARD -i $EXTERNAL_INTERFACE -o $DMZ_INTERFACE -p tcp -m tcp \
    --dport 80 -j ACCEPT # Ext -> DMZ: HTTP
iptables -A FORWARD -i $EXTERNAL_INTERFACE -o $DMZ_INTERFACE -p tcp -m tcp \
    --dport 443 -j ACCEPT # Ext -> DMZ: HTTPS
iptables -A FORWARD -i $DMZ_INTERFACE -o $EXTERNAL_INTERFACE \
    -m state --state RELATED,ESTABLISHED -j ACCEPT
```

これによりパケット通信において外部ネットワークから DMZ への通信において tcp をつかうポート番号 80(HTTP)、443(HTTPS) の通信を許可する。また、受信返答などに用いられる通信においては外部からローカルネットワークへの通信について既に確立した、もしくは新しく許可されている通信に対しては許可をすることで一度許した通信については許可をする。

```
iptables -A POSTROUTING -t nat -s $INTERNAL_LAN -o $EXTERNAL_INTERFACE -j SNAT \
    --to-source $IPADDR
iptables -A POSTROUTING -t nat -s $DMZ_LAN -o $EXTERNAL_INTERFACE -j SNAT \
    --to-source $IPADDR
```

これによりローカルネットワーク及び DMZ ネットワークから外部への接続は SNAT で行い、アドレスを送信時を変換する。

```
iptables -A PREROUTING -t nat -i $EXTERNAL_INTERFACE -p tcp -m tcp \
    --dport 80 -j DNAT --to-destination 192.168.150.2:80
iptables -A PREROUTING -t nat -i $EXTERNAL_INTERFACE -p tcp -m tcp \
    --dport 443 -j DNAT --to-destination 192.168.150.2:443
```

これにより machine2 の cp をつかうポート番号 80(HTTP)、443(HTTPS) の通信についてそれぞれルータの IP アドレスで公開する DNAT で変換する。

5.4.2 設定スクリプト修正後の設定状況の確認

```
[root@icesc12 ~]# iptables -L -n
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:22
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:22
ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           state NEW udp dpt:67
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0
ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           state NEW udp dpt:67
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED

Chain FORWARD (policy DROP)
target     prot opt source               destination
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 dpt:443
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
```

```
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 dpt:443
ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           udp dpt:53
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW,ESTABLISHED tcp dpt:22
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW,ESTABLISHED tcp dpt:80
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW,ESTABLISHED tcp dpt:443
ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           state NEW,ESTABLISHED udp dpt:53
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED

Chain OUTPUT (policy DROP)
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:22
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:80
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:443
ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           state NEW udp dpt:53
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:22
ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           state NEW udp dpt:68
ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           state NEW udp dpt:68
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
[root@icesc12 ~]# iptables-save
# Generated by iptables-save v1.4.21 on Thu May 23 14:33:39 2019
*nat
:PREROUTING ACCEPT [1:83]
:INPUT ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
:POSTROUTING ACCEPT [0:0]
-A PREROUTING -i enp1s0 -p tcp -m tcp --dport 80 -j DNAT --to-destination 192.168.150.2:80
-A PREROUTING -i enp1s0 -p tcp -m tcp --dport 443 -j DNAT --to-destination 192.168.150.2:443
-A POSTROUTING -s 192.168.200.0/24 -o enp1s0 -j SNAT --to-source 192.168.100.12
-A POSTROUTING -s 192.168.150.0/24 -o enp1s0 -j SNAT --to-source 192.168.100.12
COMMIT
# Completed on Thu May 23 14:33:39 2019
# Generated by iptables-save v1.4.21 on Thu May 23 14:33:39 2019
*filter
:INPUT DROP [0:0]
:FORWARD DROP [0:0]
:OUTPUT DROP [0:0]
-A INPUT -i lo -j ACCEPT
-A INPUT -i enp1s0 -p tcp -m state --state NEW -m tcp --dport 22 -j ACCEPT
-A INPUT -i enp3s0 -p tcp -m state --state NEW -m tcp --dport 22 -j ACCEPT
-A INPUT -i enp3s0 -p udp -m state --state NEW -m udp --dport 67 -j ACCEPT
-A INPUT -i enp3s0 -p icmp -j ACCEPT
-A INPUT -i enp2s0 -p udp -m state --state NEW -m udp --dport 67 -j ACCEPT
-A INPUT -i enp2s0 -p icmp -j ACCEPT
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A FORWARD -i enp1s0 -o enp2s0 -p tcp -m tcp --dport 80 -j ACCEPT
-A FORWARD -i enp1s0 -o enp2s0 -p tcp -m tcp --dport 443 -j ACCEPT
-A FORWARD -i enp2s0 -o enp1s0 -m state --state RELATED,ESTABLISHED -j ACCEPT
-A FORWARD -i enp2s0 -o enp1s0 -p tcp -m tcp --dport 22 -j ACCEPT
-A FORWARD -i enp2s0 -o enp1s0 -p tcp -m tcp --dport 80 -j ACCEPT
-A FORWARD -i enp2s0 -o enp1s0 -p tcp -m tcp --dport 443 -j ACCEPT
-A FORWARD -i enp2s0 -o enp1s0 -p udp -m udp --dport 53 -j ACCEPT
-A FORWARD -i enp2s0 -o enp1s0 -p icmp -j ACCEPT
-A FORWARD -i enp1s0 -o enp2s0 -m state --state RELATED,ESTABLISHED -j ACCEPT
-A FORWARD -i enp3s0 -o enp1s0 -p tcp -m state --state NEW,ESTABLISHED -m tcp --dport 22 -j
    ACCEPT
-A FORWARD -i enp3s0 -o enp1s0 -p tcp -m state --state NEW,ESTABLISHED -m tcp --dport 80 -j
    ACCEPT
-A FORWARD -i enp3s0 -o enp1s0 -p tcp -m state --state NEW,ESTABLISHED -m tcp --dport 443 -j
    ACCEPT
-A FORWARD -i enp3s0 -o enp1s0 -p udp -m state --state NEW,ESTABLISHED -m udp --dport 53 -j
    ACCEPT
-A FORWARD -i enp3s0 -o enp1s0 -p icmp -j ACCEPT
-A FORWARD -i enp1s0 -o enp3s0 -m state --state RELATED,ESTABLISHED -j ACCEPT
-A FORWARD -i enp3s0 -o enp2s0 -j ACCEPT
-A FORWARD -i enp2s0 -o enp3s0 -m state --state RELATED,ESTABLISHED -j ACCEPT
-A OUTPUT -o lo -j ACCEPT
-A OUTPUT -o enp1s0 -p tcp -m state --state NEW -m tcp --dport 22 -j ACCEPT
-A OUTPUT -o enp1s0 -p tcp -m state --state NEW -m tcp --dport 80 -j ACCEPT
-A OUTPUT -o enp1s0 -p tcp -m state --state NEW -m tcp --dport 443 -j ACCEPT
```

```
-A OUTPUT -o enp1s0 -p udp -m state --state NEW -m udp --dport 53 -j ACCEPT
-A OUTPUT -o enp1s0 -p icmp -j ACCEPT
-A OUTPUT -o enp2s0 -p tcp -m state --state NEW -m tcp --dport 22 -j ACCEPT
-A OUTPUT -o enp2s0 -p udp -m state --state NEW -m udp --dport 68 -j ACCEPT
-A OUTPUT -o enp3s0 -p udp -m state --state NEW -m udp --dport 68 -j ACCEPT
-A OUTPUT -o enp3s0 -p icmp -j ACCEPT
-A OUTPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
```

より、確かに設定が変更されていることが確認できた。

5.4.3 確認用ソフトウェアのインストール

```
[root@icesc12 machine1_2]# rpm -ql bind-utils
/etc/trusted-key.key
/usr/bin/dig
/usr/bin/host
/usr/bin/nslookup
/usr/bin/nsupdate
/usr/share/man/man1/dig.1.gz
/usr/share/man/man1/host.1.gz
/usr/share/man/man1/nslookup.1.gz
/usr/share/man/man1/nsupdate.1.gz
```

dig、及び nslookup がインストールされていることを確認できた。

- machine2

5.4.4 firewalld の動作状態の確認

```
[root@www2 ~]# systemctl status firewalld
firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; enabled; vendor preset: enabled)
   Active: active (running) since Thu 2019-05-23 14:07:59 JST; 42min ago
     Docs: man:firewalld(1)
 Main PID: 760 (firewalld)
   CGroup: /system.slice/firewalld.service └─
           760 /usr/bin/python -Es /usr/sbin/firewalld --nofork --nopid

May 23 14:07:58 www2.ice.nuie.nagoya-u.ac.jp systemd[1]: Starting firewalld - dynamic firewall
     daemon...
May 23 14:07:59 www2.ice.nuie.nagoya-u.ac.jp systemd[1]: Started firewalld - dynamic firewall
     daemon.
```

ファイアウォールが active で、Thu 2019-05-23 14:07:59 JST; 42min ago からファイアウォールが起動していることが分かった。

5.4.5 現在のゾーンの許可状態を確認

```
[root@www6 ~]# firewalld -cmd --get-active -zones
public
    interfaces: enp3s0
```

現在 enp3s0 のインターフェースのみ許可していることが分かった。

5.4.6 必要なサービスの許可設定の追加

```
[root@www2 ~]# firewall-cmd --list-all
public (active)
  target: default
  icmp-block-inversion: no
```

```
interfaces: enp3s0
sources:
services: ssh dhcpv6-client http https
ports:
protocols:
masquerade: no
forward-ports:
source-ports:
icmp-blocks:
rich rules:
```

services の欄より、ssh のほか、dhcpv6-client、http、https がサービスとして許可されたことがわかった。

5.5 考察

ファイアウォールが起動していない場合のサーバの挙動について考察する。

通常ファイアウォールではパケットの IP アドレスとポート番号をもとに通信の許可及び拒否の決定を下す。これが働いていない場合外部からいかなるアプリケーションを呼び起こす通信でも受け付けてしまう。例えばこれがデータを何らかの形で破壊することができるアプリケーション間での通信であってもファイアウォールが起動していない場合ではブロックすることができないと考えられる。

6 実験 5: WWW サービスの設定及び起動

6.1 実験目的・概要

本実験では、内、外部で http 及び https でのネットワーク接続を可能とするため、machine2 にて WWWW サービスの設定及び起動を行い、WWW サーバを起動させる。

6.2 実験方法

6.2.1 ファイアウォールの停止

以下のコマンドを実行し、ファイアウォールを一時的に停止した。

```
systemctl stop firewalld
```

6.2.2 WWW サーバのインストール

以下のコマンドを実行し、ApacheWWW サーバと https 対応モジュールをインストールした。

```
yum info httpd
yum info mod_ssl
yum install httpd
yum install mod_ssl
```

6.2.3 SSL/TLS 自己署名証明書の作成

cd /etc/pki/tls/certs で/etc/pki/tls/certs に移動した。

1. 秘密鍵の作成

   以下のコマンドを実行し、パスフレーズを適当に入力し設定した。

   ```
   openssl genrsa -aes128 1024 ¿ server.key
   ```

2. パスフレーズの除去

   以下のコマンドを実行し、パスフレーズの除去を行った。

   ```
   openssl rsa -in server.key -out server.key
   ```

3. CSR の作成

   以下のコマンドを実行し、CSR の作成を行った。

   ```
   openssl req -utf8 -new -key server.key -out server.csr
   ```

4. 有効期間 1 年のサーバ証明書の作成

   以下のコマンドを実行し、有効期限 1 年のサーバ証明書の作成を行った。

   ```
   openssl x509 -in server.csr -out server.crt -days 365 -req -signkey server.key
   ```

6.2.4 WWW サーバの設定ファイルにサーバ証明書を指定

設定ファイル/etc/httpd/conf.d/ssl.conf の内容を以下のように変更した。

```
SSLProtocal +TLSv1.2
SSLCertificateFile /etc/pki/tls/certs/server.crt
SSLCertificateKeyFile /etc/pki/tils/certs/server.key
```

6.2.5 WWW サーバの主設定ファイルを修正

Apache WWW サーバ主設定ファイル/etc/httpd/conf/httpd.conf の ServerName ディレクティブを group2a に変更した

6.2.6 WWW サーバの起動

以下のコマンドを実行し、WWW サーバの起動を行った

```
systemctl restrat httpd
```

6.2.7　machine3 と外部端末から http、https 両方の動作を確認

machine3 と外部端末から http、https 両方の動作を確認を行った。machine3 と外部端末から http、https 両方の動作を確認した。

6.2.8　WWW サーバのブート時自動起動の設定

以下のコマンドを実行し、WWW サービスのブート時自動起動の設定を行った。

```
systemctl enable httpd
```

6.3　実験結果

6.3.1　WWW サーバのインストール

```
[root@www2 machine2_2]# yum info httpd
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
 * base: mirrors.cat.net
 * extras: mirrors.cat.net
 * updates: mirrors.cat.net
Available Packages
Name        : httpd
Arch        : x86_64
Version     : 2.4.6
Release     : 89.el7.centos
Size        : 2.7 M
Repo        : updates/7/x86_64
Summary     : Apache HTTP Server
URL         : http://httpd.apache.org/
License     : ASL 2.0
Description : The Apache HTTP Server is a powerful, efficient, and extensible
            : web server.

[root@www2 machine2_2]# yum info mod_ssl
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
 * base: mirrors.cat.net
 * extras: mirrors.cat.net
 * updates: mirrors.cat.net
Available Packages
Name        : mod_ssl
Arch        : x86_64
Epoch       : 1
Version     : 2.4.6
Release     : 89.el7.centos
Size        : 112 k
Repo        : updates/7/x86_64
Summary     : SSL/TLS module for the Apache HTTP Server
URL         : http://httpd.apache.org/
License     : ASL 2.0
Description : The mod_ssl module provides strong cryptography for the Apache Web
            : server via the Secure Sockets Layer (SSL) and Transport Layer
            : Security (TLS) protocols.
```

Summary の欄より、httpd では Apachehttp サーバがインストールされ、mod_ssl では the Apache HTTP サーバー用の SSL/TLS モジュールがインストールされたことがわかる。https が SSL/TLS により http を暗号化したプロトコルであるためこのようにして https に対応させた。

6.3.2　SSL/TLS 自己署名証明書の作成

```
[root@www2 certs]# openssl req -utf8 -new -key server.key -out server.csr
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [XX]:JP
State or Province Name (full name) []:Aichi
Locality Name (eg, city) [Default City]:Chikusa
Organization Name (eg, company) [Default Company Ltd]:Nagoya University
Organizational Unit Name (eg, section) []:School of Informatics
Common Name (eg, your name or your server's hostname) []:Group2a
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
```

自己署名として、国名、県名、区、所属機関、グループの番号を入力し、ほか部分は入力しなかった。

6.4 考察

本実験では http/https 通信両方でアクセス可能ではあるが、 https はブラウザは安全上の理由からアクセスを一度ブロックしている。これは本実験の証明書が自己署名証明書であるために起こっていると考えられる。今回はホワイトリストに追加して例外的にアクセスを許可したことにより、接続が可能になったと考えられる。

7 調査課題 1: TCP パケット及び IP パケットのヘッダ情報

データを送信する際、HTTP や SMPT などのアプリケーションプロトコルのデータに TCP ヘッダが付加される。この付与されたデータを全体として TCP セグメント又は TCP パケットと呼ぶ。TCP パケットヘッダフォーマットは図 1 である。

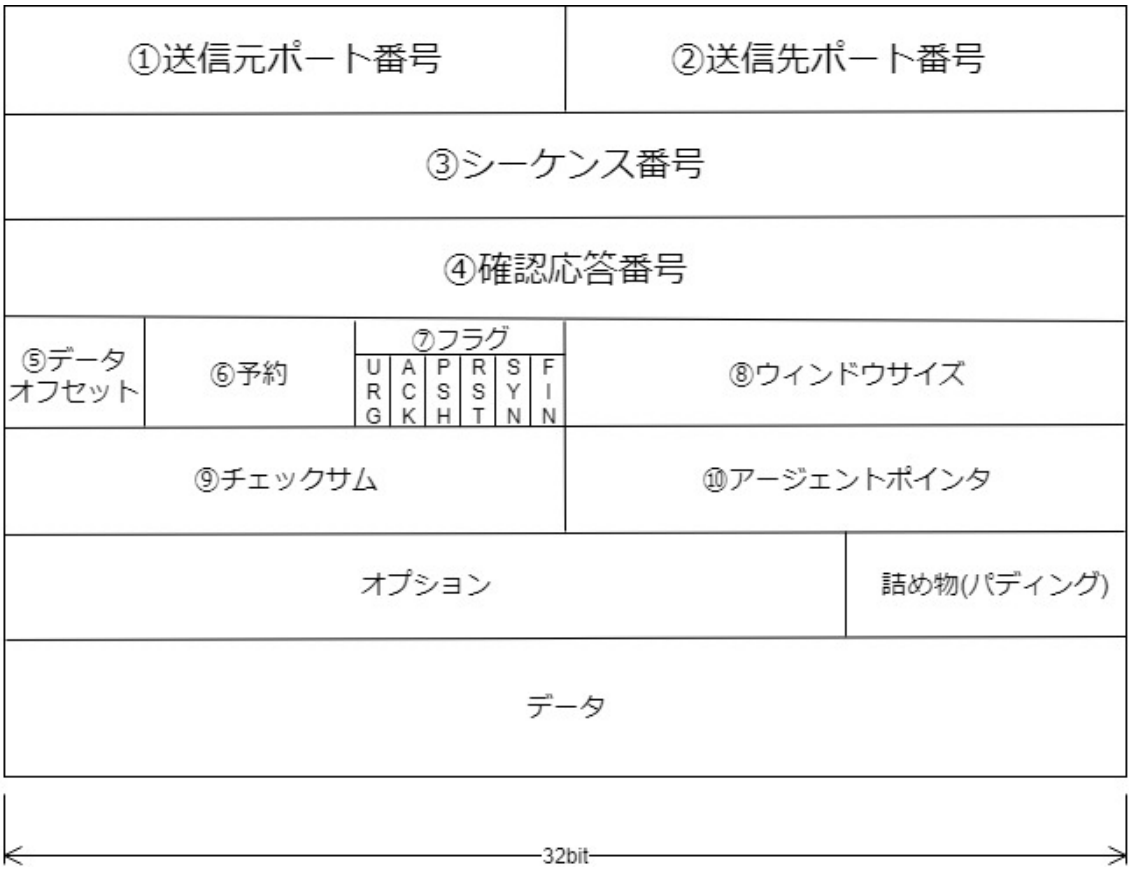

図 1　TCP ヘッダーフォーマット

1. 送信元ポート番号 (16bit) :
   送信元のポート番号。加えて、送信先のポート番号と、送信送信元とあて先の IP アドレスの 4 つの番号の組により、TCP のコネクションを識別する。パケットの返信時に送信元とあて先を入れ替えて、送信する。。TCP では必ず双方向に通信する必要があるため、送信元ポート番号を 0 にすることはできず、逆方向の TCP パケットにも、あて先となるポート番号がつける必要がある。
2. 送信先ポート番号 (16bit) :
   あて先となるアプリケーションが待ち受けしているポートの番号を表す。16bit 幅であるが、0 が予約済であるため、1〜65535 まで利用でき、目的別に利用可能な範囲が決められている。このうち 0 1023 までのポート番号はウェルノウンポートと呼ばれ主要なアプリケーションごとに割り振られている。例えばプロトコルごとで言えば、HTTP は 80、POP3 は 110、FTP data は 20 などを使っている。また、1024〜49151 は Registered Port と呼ばれ、登録されたサービスが利用するポートであり、以降の番号を Dynamic Port 又は Private Port と呼び、動的なアプリケーションなどで利用するポート番号である。
3. シーケンス番号 (32bit) :
   送信するデータの順序を示す 32bit の番号で、シーケンス番号により TCP セグメントの順序制御が可能である。初期値は 0 でなくてもよく、送信するデータ 1byte ごとに、シーケンス番号を 1 つずつ昇順に割り当て、どこまでデータを送信したかを指定する。このフィールドは 32bit 幅なので、2^32 =

4Gbytes 送信すると、また同じ番号に戻ってくる。

4. 確認応答番号 (32bit)：
   ACK(ACKnowledge) 番号とも呼び、データを受信した確認通知と、次に送信を期待するシーケンス番号を通知するために利用する。シーケンス番号と同様に 32bit 幅なので、4Gbytes ごとに同じ番号に戻ってくることになる。この番号は、受信が完了したデータ位置のシーケンス番号＋ 1 を返す。
5. データオフセット (4bit)：
   TCP のデータの始まり、つまり、TCP ヘッダのヘッダ長を示す。このフィールドは 4bit しかないため、0〜15 しか表すことができないが、1bit で表す値＝ 32bit（4bytes）単位で数える。よって、TCP ヘッダ・サイズは最大で 15 × 4 ＝ 60bytes までであり、またヘッダの最小サイズは 5 × 4=20bytes となる。
6. 予約 (6bit)：
   未使用領域。将来の拡張のために確保されている。
7. フラグ (6bit)：
   - URG（urgent）：ビットが 1 の場合は緊急データが含んでることを指す, がほとんどの場合では用いられない。
   - ACK：確認応答を行うセグメントであることを示し。コネクション確率時以外のパケットの場合はすべて 1 である。
   - PSH(push)：受け取ったデータをすぐにアプリケーションに渡す場合を表す。通常、TCP では送信するデータが膨大である場合、データを分割して送信する。その際、受信側が、受け取ったデータをバッファに保存して、すべてのデータを受け取ったのち、処理を行うと効率がいい場合と、送られてきたデータ毎で処理を行うほうが効率がいい場合両方に対処するために存在する。
   - RST(reset)：コネクションの強制終了を要求する。例えばサーバ側が要求に応えられない場合や、許可されていない IP アドレスからの接続要求を拒否したい場合に、この RST パケットが返されることがある。
   - SYS(synchronize)：コネクションの確率を要求する場合に利用する。TCP は双方向通信路であるため、双方から送信されるそれぞれの最初の接続要求パケットにこの SYN フラグがセットされ、2 番目以降のパケットにはセットされない。
   - FIN：コネクションの通常終了を要求する。双方から FIN が送られると TCP 接続が終了し、内部バッファなどのリソースが解放される。
8. ウィンドウサイズ (16bit)：
   受信側が一度に受信可能なデータ量を送信側に通知する。TCP の送信側は、相手から通知されたウィンドウ・サイズを見て、送信可能な最大のデータ量を判断する。
9. チェックサム (16bit)：
   TCP ヘッダとデータ部分のエラーチェックを行うための利用する。
10. アージェントポインタ：
    URG フラグが立っているときのみ有効で、緊急データの開始位置（サイズ）を表す数値を指定するを示す。

参照：[4]、[5]

また、データに上記のヘッダがついた TCP セグメントに IP ヘッダが付加されることで IP パケットとな

図 2　IPv4 ヘッダフォーマット

る。図 2 は IPv4 における IP ヘッダである。

1. バージョン (4bit) :
   IP プロトコルのバージョンを表現するために使われている。例えば今回のヘッダの情報は IPv4 のヘッダ情報であるため値は 0100_2 である。同じネットワーク媒体上に IPv4 とそれ以外のパケットが混在している際に、区別することができる。
2. ヘッダ長 (4bit) :
   IP ヘッダの長さが入る。ただし単位は 4bytes であり、標準の IP ヘッダは 20bytes であるためヘッダ長には 0101_2 が入る。確保されている幅は 4bit 幅であるため表せる最大長は 60bytes までである。
3. ToS(Type of Servise) (8bit) :
   IP パケットの優先順位を指定するために使用される。上位 3 ビットもしくは上位 6bit で優先順位を決定し、優先順位が高い場合はほかのパケットよりも優先してルーティング処理などを行う。しかし、現在の通信では ToS 指定はほとんど使用されておらず意味を持っていない場合が多い。
4. データグラム長 (16bit) :
   IP パケット全体のサイズを byte 単位で計測した本野が入る。生データのサイズを知りたい場合はデータグラム長-ヘッダ長から求められる。なお IP パケットの最大のサイズはここに収められるサイズでないといけないため、64Kbytes=65,535 までとなる。
5. 識別番号 (16bit) :
   IP パケットを複数のデータに分割して送信する際に用いる番号であり、受け取り側は識別番号が同じ

場合複数のデータを結合し受信した IP パケットをもとの IP パケットとして再構成する。

6. フラグ (3bit) :
   分割されたデータが最後のデータであるかどうかという情報が入っている。3bit の幅が確保されているが、使用されているのは 2bit であり、これ以降のデータを結合する必要があるか表す部分と、これが最後のデータであるかを表す bit がそれぞれ確保されている。
7. フラグメントオフセット (13bit) :
   受信したデータが元のデータのどの位置にあったかを表す部分である。データの分割は 8bytes 単位で行われるため、13bit 幅でも十分にデータの復元が可能である。
8. TTL(Time To Live)(8bit) :
   パケットの寿命を表す部分である。この寿命とはパケットが何台のルータを経由することができるを表している。ルータを超えるごとに TTL の値を 1 減らし、0 になった段階でパケットが破棄される仕組みになっている。これにより、ルーターのルーティングテーブルに不具合があり、同ネットワーク内を延々とループすることを防ぐことができる。
9. プロトコル番号 (8bit) :
   IP の上位層に当たるトランスポート層のプロトコルの種類を識別するための部分である。1 番の ICMP、6 番の TCP、17 番の UDP、47 番の GRE などが、実際によく使われているプロトコルである。
10. ヘッダチェックサム (16bit) :
    IP ヘッダにエラーがないかをチェックするためのフィールドである。IP パケット全体のチェックは他のプロトコル層で検出が行われるためここでは行わない。
11. 送信元 IP アドレス (32bit) :
    送信元のコンピュータの IP アドレスを表す。送信だけを考えればここの部分は必要ないが、通信の相手先のコンピュータが、パケットを送り返すことができないので、この送信元 IP アドレスは絶対必要であるといえる。このアドレスは必ずユニキャストである。
12. 送信先 IP アドレス (32bit) :
    送信先の IP アドレスを表すフィールドである。

参照 : [4]、[6]

8 調査課題 2: TCP/IP 通信におけるブロードキャストの役割

ブロードキャストは 1 対多数の通信であり、送信先は同一ネットワーク上のすべてのホストである。TCP/IP 通信では、ブロードキャストアドレスを送信先 IP アドレスとして指定することでネットワーク上に接続されている機器にパケットを送信することができ、一斉に複数機器にデータを送信するために利用される。

ブロードキャストを行う際に使用するブロードキャストアドレスは 255.255.255.255 という IP アドレスの bit すべてを 1 にしたリミテッドブロードキャストアドレスと IP アドレスのホスト部をすべて 1 にしたアドレスである、ディレクテッドブロードキャストアドレスの 2 種類存在する。前者のアドレスに送ったパケットは同一イーサネット内のすべてのコンピュータに届くものの、ルータで接続した他のネットワークには送信されない。後者のアドレスに送ったパケットは必要に応じてルータを介して宛先のネットワークに送られ、そこにつながっているすべてのコンピュータに送られる。しかし、一般にルータでは後者の中継は禁止されること

が推奨されており、他のネットワークからブロードキャスト通信が行われることは通常無い。このブロードキャストアドレスは送信元 IP アドレスとして指定されることは無い。[3] ブロードキャストはその特徴を利用して、アドレス解決の際にルータが送信先の MAC アドレスを調べるときや、パソコンが起動時に同サブネットワーク内の機器に自分自身のコンピュータ名をブロードキャストして、ほかのコンピュータに対して自分自身の存在を知らせる際に使用する。

9　調査課題 3: TCP パケットの通信過程

まず送信元である PC1 のデフォルトゲートウェイであるルータ (192.168.200.1) にパケットは転送される。その際すでに IP パケットとして送信元及び送信先 IP の書かれた IP パケットはすでについている。ここに書かれているトランスポート層のプロトコルは 6 の TCP である。またルータに転送される前の現状の IP ヘッダは送信先アドレスと送信元アドレスの IP ヘッダの上に送信元アドレスはそのままの送信先アドレスがルータになっているヘッダがついている。その上のイーサネットヘッダに PC1 の MAC アドレスとルータの MAC アドレスが書かれている。

次にルータはルーティングテーブルに基づいて、受け取ったパケットの送信先 IP アドレスをキーにして検索をする。その際、ルーティングテーブルに送信先アドレスはなく、ネクストホップアドレスである 192.168.100.1 にパケットを転送する。その際 FW 構築実験用ルータ (192.168.100.1) の MAC アドレスを求めるためにブロードキャストで LAN1 に繋がっている機器にリクエストを送る。この際の送信先 MAC アドレスは FF:FF:FF:FF:FF:FF である。リクエストに応じて帰ってきた MAC アドレスをもとに IP ヘッダの上にイーサヘッダをつけ送信元 MAC アドレスにルータの MAC アドレス、送信先 MAC アドレスにリクエストで獲得した MAC アドレスを使用して、パケットを送信する。この際元の IP ヘッダの送信元 IP アドレスは NAT によりローカルアドレスから何らかの対応したグローバルアドレスに変換される.

最後に FW 構築実験用ルータがルーティングテーブルに基づいて、PC2(10.10.1.20) にパケットを送信することで通信が完了する。この際、上と同様にブロードキャストで MAC アドレスをリクエストして通信を行う。[4]

10　調査課題 4: 各種サービスの詳細

10.1　Unix/Linux システムのユーザ認証

一般に UNIX や Linux システムといった汎用コンピュータはマルチユーザシステムと呼ばれる機能を採用している。マルチユーザシステムは、Windows のマルチユーザ機能のように、複数のユーザで 1 台のシステムをユーザを切り替えて使用することと異なり、複数のユーザが同時に同じシステムにログインして使用できるシステムを指す。例として、PAM 及び NSS は Linux や UNIX におけるユーザの認証の代表的なシステムである。PAM は Pluggale Authentication Modules の略称で主に認証連携を実現するための機構である。NSS とは Name Service Switch の略で，本来は認証以外の一部情報も含め，Linux 上の設定ファイルの内容を参照する代わりに NSS モジュールが生成した情報を参照できるようにするための機構である。

ユーザを認証するにあたって、我々クライアントが行えることは、ユーザの追加、ユーザごとのパスワードの設定、ユーザのパスワードが無効になってから、そのユーザのアカウントが無効になるまでの期限の設定、ユーザアカウントの有効期限の設定などがある。また、ユーザは何らかのグループに追加することができ、グ

ループごとに対して実行権限を付与することができ、ディレクトリにアクセス制限をかけることができる。

パスワードの認証のほかにユーザは、次のような認証が利用できる。

- 証明書ベースの認証
  証明書をベースとしたクライアントの認証は、SSL プロトコルの一部である。クライアントはランダムに生成されたデータにデジタル処理で署名し、証明書と証明済みのデータをネットワーク経由で送信する。サーバーは署名を確認して証明書の有効性を確認する。
- Kerberos 認証
  Kerberos は ticket-granting tickets (TGT) と呼ばれる短期の認証情報システムを確立する。ユーザーがユーザー名とパスワードという認証情報を提示することでユーザーが特定され、このユーザーにチケットを発行可能であることをシステムに対して示す。TGT はその後、Web サイトや電子メールなどの他のサービスへのアクセスチケットを要求する際に繰り返し使用することができる。このように TGT を使用した認証では、ユーザーの認証プロセスは一度で済む。
- スマートカードベースの認証
  これは証明書ベースの認証とわずかに異なる。スマートカード (または トークン) にはユーザーの証明書が保存されている。ユーザーがトークンをシステムに挿入すると、システムは証明書を読み取り、アクセスを許可することができる。

[7] を参照

10.2 DNS

DNS はドメインネームシステムの略称で、ドメイン名 (co.jp など) から IP アドレスを返すシステムである。通常 www などのネットワークでは IP アドレスを利用して通信を実現しているが、IP アドレスは数字の羅列で構成されているため、人間が簡易的に理解し、使用するにはむつかしい。そこで、人でも分かりやすいドメイン名を用いてこれを元に検索を行い、対応する IP アドレスを返すという仕組みが使われている。この仕組みは名前解決と呼ばれ、DNS はこの名前解決を行うシステムである。名前解決は、インターネット上に存在する多数の DNS サーバーによりデータを分散することで実現している。

DNS サーバには 2 種類あり、権威 DNS サーバとキャッシュ DNS サーバがある。権威 DNS サーバは DNS コンテンツサーバーとも呼ばれ、ドメイン名に関する完全な情報を格納。対象となるドメイン名への問い合わせに回答する役割を担っている。ドメイン名の登録者や、その運用を預かるレンタルサーバー事業者などが運用している。キャッシュ DNS サーバー は再帰 DNS サーバーとも呼ばれ、権威 DNS サーバーに名前解決の問い合わせを行う。次に同じ問い合わせを受けた時に備え、データを一時保存するといった機能を有している。キャッシュ DNS サーバーは ISP(プロバイダ) などが会員への提供のためや、社内ユーザー用に企業内で運用されている。キャッシュ DNS サーバーは十分な管理・運用がされていないことが多く、攻撃者に狙われやすい傾向がある。

近年では DNS サーバとの通信を暗号化する DNS over TLS(DoT) および DNS over HTTPS (DoH) が開発されており、暗号化された通信路である TLS 及び HTTPS の上に DNS 通信を乗せることでクライアントと DNS サーバ間の通信経路の安全性を保つプロトコルが標準化されている。[8] を参照

10.3　メール配信サービス

一般にメール配信サービスではクライアントの情報を扱うクラウド (ASP・SaaS) サービスを提供する。セキュリティ面で重要な点はクラウドへのアクセスにおけるセキュリティとクラウドからクライアント送られるもしくはクライアントからクラウドへ送られる通信路でのセキュリティが求められる。通信路においてはSSL により暗号化されたものを使用することで通信路を暗号化することでセキュリティ面を保つ。ほかには、管理画面にアクセスできる IP を制限することでオフィス外からのアクセスを防止することや、管理権限の制限により個人情報へのアクセスを最小限にすることで個人情報の流失を防ぐことや、すべての操作ログにおいて IP 等の記録をとることでどのユーザがどのような問題行為を起こしたのかの原因究明を容易にすることでセキュリティ面を保つサービスが多く存在している。[9] を参照

11　調査課題 5: SSL/TLS を利用した通信路暗号化を行うプロトコル

SSL(Secure Sockets Layer)/TLS(Transport Layer Security) は多くの場合、コネクション型のトランスポート層プロトコル（通常は TCP）とアプリケーション層の間で使われる。主に HTTP での利用を意識して設計されているが、アプリケーション層の特定のプロトコルには依存せず、様々なアプリケーションにおいて使うことが可能である。UDP や DCCP といったデータグラム型プロトコル上でも実装されており、こちらは Datagram Transport Layer Security (DTLS) として独立した標準化がされている。SSL/TLS はアプリケーション層のプロトコルと組み合わせることで、HTTPS などセキュアな通信プロトコルを実現している。そのため、SSL/TLS を利用したプロトコルには元になるアプリケーション層が存在する。表 11 は代表的な https 以外の SSL/TLS を利用したプロトコルと元になったプロトコル及びその両方のポート番号を記したものである。

表 1　SSL/TLS を利用するプロトコル

プロトコル	ポート番号	元のプロトコル	ポート番号
SMTPS	465	SMTP	25
LDAPS	636	LDAP	389
FTPS(data)	989	FTP(data)	20
FTPS(control)	990	FTP(control)	21
IMAPS	993	IMAP	143
POP3S	995	POP3	110

SMTPS、IMAPS、POP3S はメール用プロトコルであり、ポート番号はそれぞれ 465、993、995 である。通信方式は、初めから通信を SSL/TLS で暗号化された状態で始める方式以外に、通常の SMTP で通信を開始し、双方のサーバーが通常の通信を SSL/TLS で暗号化された状態に切り替える STARTTLS の 2 種類の方式がある。後者ではメールサーバとメーラ両方が STARTTLS に対応していない場合は通常の暗号化されていない POP3、IMAP4、SMATP で通信する。POP3S、IMAPS、SMTPS によって暗号化されるのは、自分の PC のメーラおよび直接接続するメール・サーバ間の通信であり、メールそのものやメール・サーバ同士の配送及び、相手のメーラとメールサーバの通信路には関与しない。このためサーバ間の通信や、相手のメー

ラとメール・サーバの間の通信が暗号化されるかは、相手のメーラ／サーバなどに依存し、自分側で制御する術はない。もし自分のメーラから相手のメーラまで一貫した暗号化を行いたいのであれば、メールそのものを暗号化するなど別の手段を検討すべきである。[10]

LDAPS は情報ディレクトリに対するアクセスおよび管理を行うためのプロトコルで、ポート番号は 636 であり、ユーザ、グループ、ネットグループのようなオブジェクトを格納するための情報ディレクトリとして使用される LDPA を SSL/TLS により暗号化したものである。[11]

FTPS は SSL/TLS による暗号化・認証化により通信を暗号化し、データファイルのやり取りを行うプロトコルでポート番号は 989、及び 990 ある。FTPS には実行モードが 2 つ存在し、接続実行後に AUTH コマンドによって暗号化通信を実行するモードと、サーバーに接続後暗号化通信を実行するモードの 2 つである。FTPS は、アスキー/バイナリモードの転送がサポートされているため、転送先で、ファイルの改行コードによる文字化けのリスクを減らすことができる。また、Windows 用の FTP クライアントソフトである FFFTP や、WinSCP などを使用してサーバーと通信を行うことが可能である。[12]

12　まとめ

本レポートでは、DMZ セグメントを備えたサブネットの構築を行った。Linux では光学ドライブはマウントというシステムを用いており、そのデバイスは UUID という値で管理されていることが理解した。また、ネットワークインタフェースの設定を手動で行うことでそのルーティングテーブルやアドレスの管理について理解し、ネットワーク通信に必要なもの理解をした。そのため、DHCP やファイアウォール、WWW サーバの設定を行うことができた。

参考文献

[1] 第 2 章 IP ネットワークの設定 — Red Hat Customer Portalhttps://access.redhat.com/documentation/ja-jp/red_hat_enterprise_linux/7/html/networking_guide/ch-Configure_IP_Networking#sec-Using_the_NetworkManager_Command_Line_Tool_nmcli

[2] きはしまさひろ — ”イラスト図解式　この一冊で全部わかるサーバーの基本” — SB クリエイティブ株式会社 — 2016

[3] きはしまさひろ — ”イラスト図解式　この一冊で全部わかるネットワークの基本” — SB クリエイティブ株式会社 — 2016

[4] Gene— TCP/IP の基礎 — 株式会社毎日コミュニケーションズ — 2011

[5] 基礎から学ぶ Windows ネットワーク— 第 15 回　信頼性のある通信を実現する TCP プロトコル (2019/5/22)
https://www.atmarkit.co.jp/ait/articles/0401/29/news080_2.html

[6] 基礎から学ぶ Windows ネットワーク— 第 10 回　IP パケットの構造と IP フラグメンテーション (2019/5/22)
https://www.atmarkit.co.jp/ait/articles/0304/04/news001_2.html

[7] 第 1 章 システム認証について — Red Hat Customer Portal(2019/5/29) https://access.redhat.com/documentation/ja-jp/red_hat_enterprise_linux/7/html/system-level_authentication_guide/introduction

[8] DNS セキュリティの基礎を知る (1) 第 1 回〜深刻化する DNS への攻撃 (2019/5/29) https://news.mynavi.jp/article/dns_security-1/

[9] メール配信のセキュリティ｜メール配信システムの Cuenote FChttps://www.cuenote.jp/fc/security/

[10] メールの送受信を暗号化する POP3s ／ IMAP4s ／ SMTPs（over SSL）とは (2019/5/24)https://www.atmarkit.co.jp/ait/articles/0801/18/news126.html

[11] SSL/TLS 経由の LDAP の概念 - NetApp Support(2019/5/24)https://library.netapp.com/ecmdocs/ECMLP2372139/html/GUID-0E97E7F2-D46D-4883-B95B-A066B0D52B3D.html

[12] SFTP?FTPS?なにが違う？ — GMO クラウドアカデミー (2019/5/24)https://academy.gmocloud.com/keywords/20170308/4155