

## 「サーバ構築・運用演習 I B」まとめ

学習内容： DNS、WEB、FILE、MAIL サーバの仕組みを理解し、AWS での設定方法を習得する

テキスト： CentOS 徹底入門第 4 版（翔泳社）ほか自作教材

### 0. 予備知識

AWS でのサーバ構築を学習する前に、Linux およびネットワークに関する基本内容を学習しておいた方がよい

#### ■Linux

Linux はオープンソースの OS で、主にサーバ構築でよく利用されている

CUI 環境でコマンドを使用して実行させる

#### ■主な Linux コマンド一覧

cat	ファイル内容表示
cd	カレント・ディレクトリの移動 引数なしでホームディレクトリ 「/」 ルート 「..」 1つ上
chgrp	ファイル・ディレクトリの所有グループを変更
chmod	ファイル・ディレクトリのパーミッションを変更
chown	ファイル・ディレクトリの所有者を変更
cp	ファイルのコピー
date	日時の表示
diff	二つのファイルの差分を出力
dig	ドメイン名から IP アドレスを調べる
echo	表示する リダイレクトするには「>」上書き 「>>」追記
exit	ログアウトする。プロセスを終了する
find	ファイル・ディレクトリの検索
grep	文字列を検索する ワイルドカード 1 文字「?」任意の文字列「*」
hostname	ホスト名を表示、設定
ls	ファイル一覧の表示 「-l」詳細表示 「-a」隠しファイルも表示
mkdir	新規ディレクトリの作成
mv	ファイル・ディレクトリの移動、リネーム
nslookup	ドメイン名から IP アドレスを調べる
passwd	ユーザのパスワードを変更
pwd	カレント・ディレクトリの絶対パスの表示
rm	ファイル・ディレクトリの削除
sed	文字列置換・行の削除
su	ユーザーを切り替える
sudo	指定したユーザーでコマンドを実行 root ユーザー（管理者ユーザー）
tail	ファイルの末尾を表示
vi	ファイルの編集 「i」文字入力「Esc」コメント入力「:wq」保存終了「/」検索
yum	パッケージの操作・管理

## 1. AWS 基礎知識

### ■AWS (Amazon Web Services)

アマゾンが提供しているクラウドコンピュータサービス  
 すぐに使える、サービスを選べる、従量課金、CUI 環境  
 1 年間無料枠あり（メールアドレス、クレジットカード必要）

### ■AmazonVPC (Virtual Private Cloud)

AWS 上での仮想的なネットワークを構築できるサービス  
 独自の IP アドレス範囲の選択、サブネットの作成、ルートテーブルやネットワークゲートウェイの設定など、仮想ネットワーク環境を完全に制御できる

### ■AmazonEC2 (Elastic Compute Cloud)

AWS 上で仮想サーバを構築・利用できるサービス  
 必要に応じて複数のインスタンスを立ち上げることで、柔軟なインフラ構築ができる  
 EC2 では仮想サーバのことを「インスタンス」という単位で扱う

### ■インスタンス

インスタンスとは AmazonEC2 で作成した仮想的なサーバのこと  
 インスタンスはプライベート IP アドレスと動的に割り当てられるパブリック IP アドレスの 2 つを持つ  
 インスタンスを利用しているときは課金対象となるので、利用しないときは停止する  
 一度削除したインスタンスを復活する方法はないので注意（利用時は開始と停止）  
 今回は 1 年間無償利用範囲に含まれる「t2.micro」タイプを利用

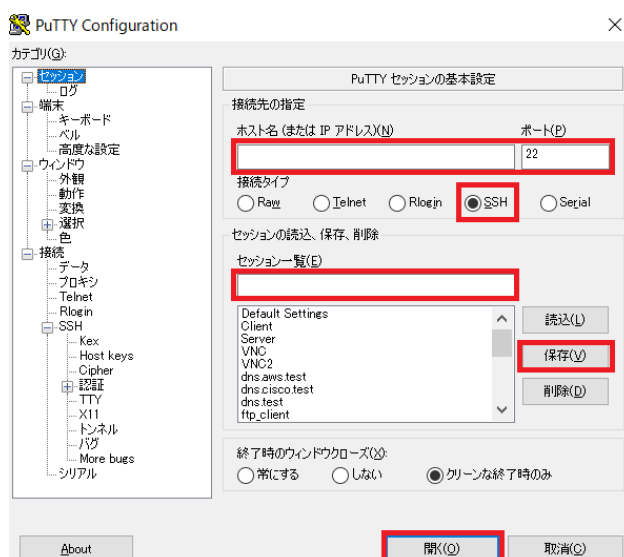
### ■セキュリティグループ

仮想的なファイアウォール機能  
 インスタンスへのアクセスを許可し、トラフィック（ネットワークを流れる情報）を制御する

### ■SSH (Secure SHell)

サーバにリモートでログインして各種遠隔操作をするためのプロトコルで、ポート番号は 22  
 Windows では TeraTerm や PuTTY などのソフトを使う  
 「ec2-user」というデフォルトユーザでログイン、キーペアファイルで認証

### PuTTY 画面



ホスト名にパブリック IP アドレス

セッション名は任意文字列

保存しておくと次回便利

サーバに接続する



ダウンロードしたキーペアを選択

## ■キーペア

SSH 接続において、ログインする際の認証に使用する公開鍵と秘密鍵のペア  
「.pem」ファイルとして1回だけダウンロードできる  
紛失するとインスタンスにログインできなくなるので、要管理  
2台目以降のインスタンスにも使うことができる

## ■ユーザ

「ec2-user」 : インスタンスを起動時のデフォルトユーザ、先頭に「\$」  
「root」ユーザ: 管理者ユーザ、ec2-user から「su -」で変更、先頭に「#」  
ec2-user から「sudo」コマンドで実行してもよい

## ■Elastic IP アドレス

パブリック IP アドレスを固定化することができる  
インスタンスを停止して起動し直してもパブリック IP アドレスは変わらない  
起動している EC2 インスタンスで、EIP を1つだけ使用している場合には料金が発生しない  
使わなくなった Elastic IP は解放する

## ■AWS Educate

14 歳以上の学生、教員が自由に活用できる無料のクラウド学習プログラム  
スターターアカウントではクレジットカード登録が不要で、教員からの招待で参加できる  
制限はあるが主要なサービスはカバーしている  
学校で機関加盟登録することでさらに無料クレジットが増える

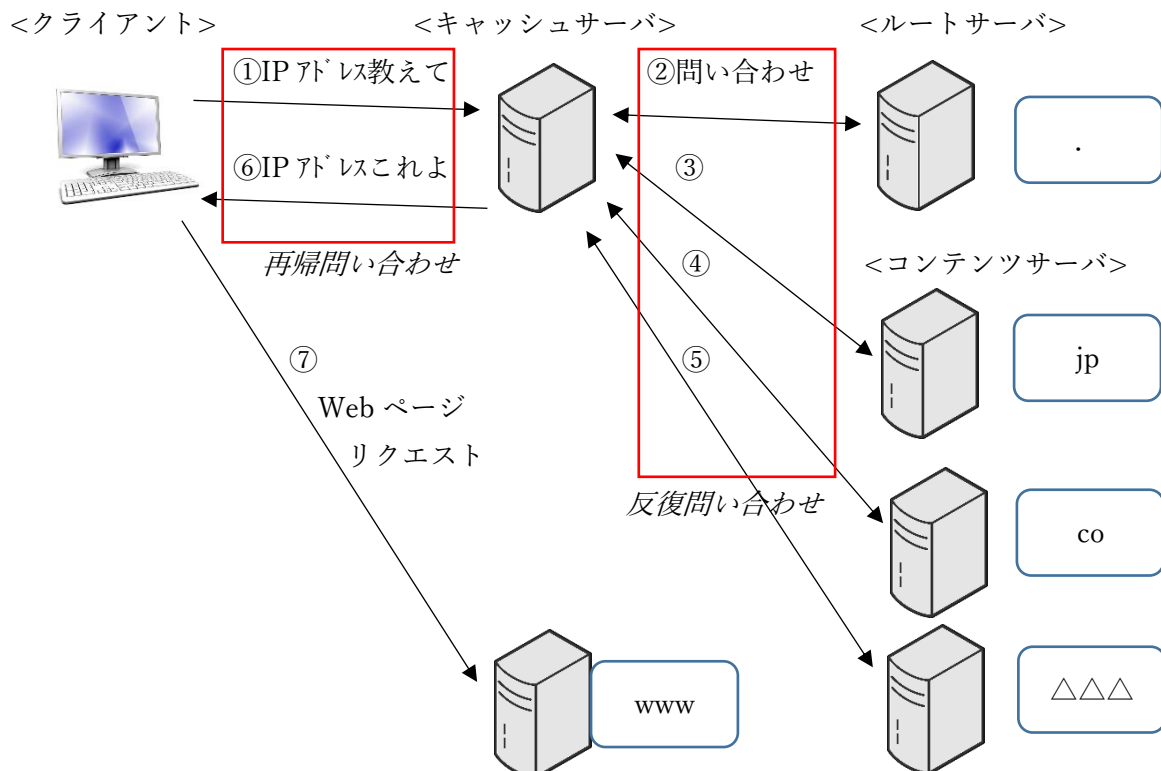
## 2. DNS サーバ構築

### ■DNS 基礎知識

- ・ DNS (Domain Name System)  
IP アドレスとドメイン名の対応情報を管理し、名前解決を行う
- ・ FQDN (完全修飾ドメイン名)  
ドメイン名を表記する際に、最後の「.」(ルート)を省略せずに記述した形式
- ・ 正引きと逆引き  
ドメイン名から IP アドレスに変換することを正引き、反対の変換を逆引きという
- ・ 再帰問い合わせ  
DNS クライアントからの名前解決要求に対し、最終的な結果を返す問い合わせ
- ・ 反復問い合わせ  
最終的な答えが得られるまで繰り返し他の DNS サーバへ行う問い合わせ
- ・ ルートサーバ  
階層構造を持つ DNS の最上位に存在するサーバで、世界に 13 個ある
- ・ キャッシュサーバ  
DNS クライアントからの名前解決を代行して、他の DNS サーバに問い合わせを行うサーバ  
結果をキャッシュしておき、次回問い合わせ時にキャッシュ情報を返す
- ・ コンテンツサーバ (権威サーバ)  
ドメイン名と IP アドレスの対応表を管理するサーバ  
外部からの問い合わせに対し、自身が管理するゾーン (ドメイン) 情報を回答する

### ■名前解決の方法

例えば、www.△△△.co.jp の名前解決をする場合



## ■BIND

Linux での DNS サーバソフトウェア

- ・パッケージ名 bind
- ・インストール
 

```
$ su -
```

```
# yum -y install bind
```
- ・設定ファイル
 

```
/etc/named.conf
```
- ・サービス起動（実行は 1 回で OK）
 

```
# systemctl start named
```
- ・自動起動 ON（実行は 1 回で OK）
 

```
# systemctl enable named
```
- ・サービス再起動（設定ファイルを書き換えた後などに実行する）
 

```
# systemctl restart named
```
- ・設定ファイルのバックアップ（大事なファイルを変更する前には必ずバックアップする）
 

```
# cp -p /etc/named.conf /etc/named.conf_bk1
```

## ■設定ファイル(/etc/named.conf)

option ステートメント 全体で使用するオプション（デフォルト値）を指定する  
ゾーンファイルの場所、再帰問合せの受付可否など

zone ステートメント ゾーンごとのオプションを指定する  
タイプの指定(hint, master, slave)、ゾーンファイル名

## ■ゾーンファイル

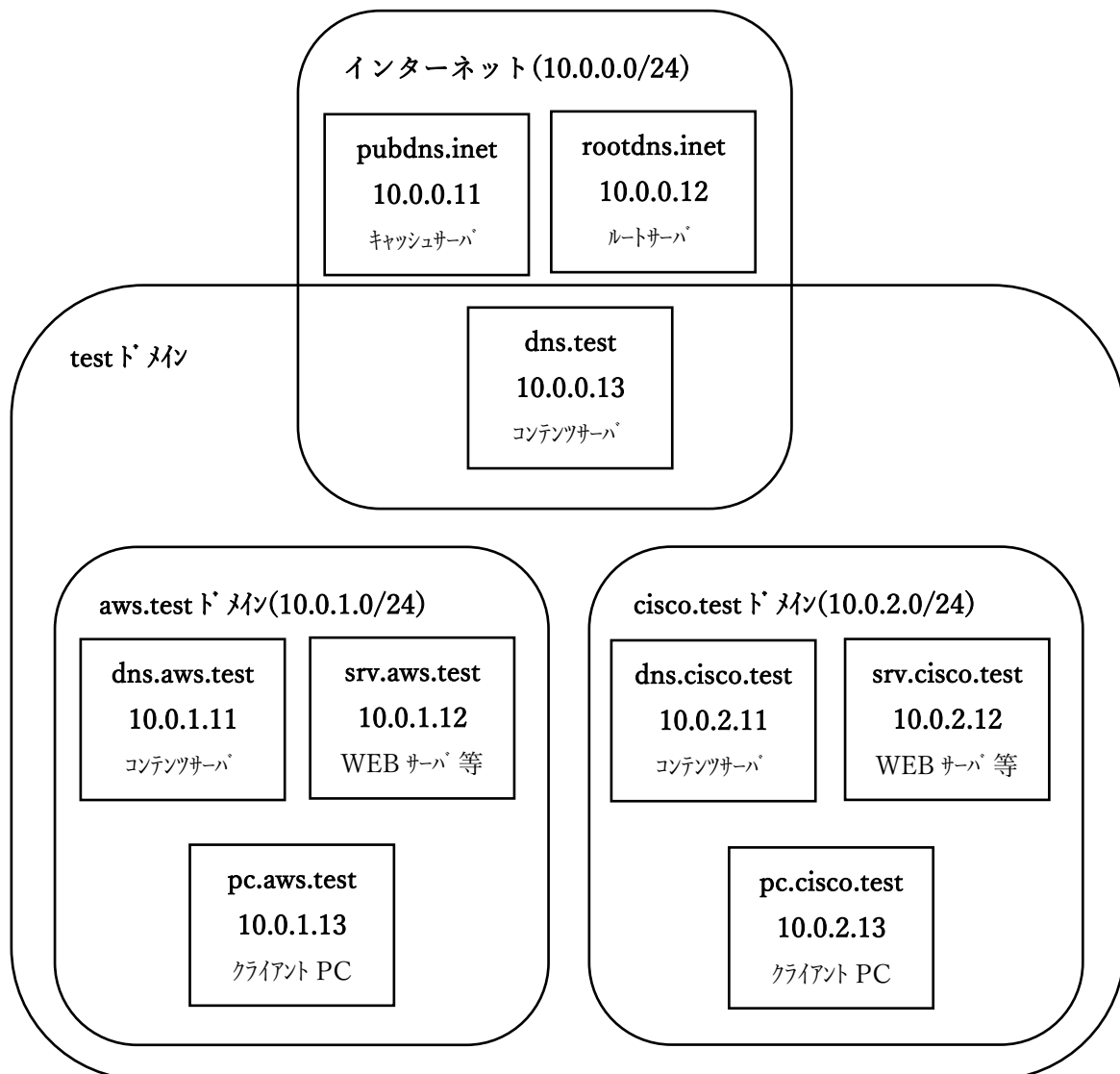
- ①SOA レコード ゾーン全体に関する情報を定義する
- ②NS レコード ゾーンの権威サーバに関する情報を定義する
- ③MX レコード メールサーバに関する情報を定義する
- ④A レコード ホスト名と IPv4 アドレスの関連付けを定義する
- ⑤CNAME レコード 別名を定義する
- ⑥PTR レコード 逆引きを定義する

キャッシュサーバはルートサーバの NS レコードと A レコードを記述

それ以外の DNS サーバは自己ドメインと配下ドメインの NS レコードと A レコードを記述

末端のコンテンツサーバは自己ドメインの NS、A、MX などすべてのレコードを記述

## ■DNS 演習構成図



## ■演習 2 - 1 「キャッシュサーバ設定①：インターネットの名前解決ができる」

【pubdns.inet】

[root@pubdns.inet ~]# cat /etc/named.conf → /etc/named.conf の中身を表示する

```
options{
    directory "/var/named";           →ゾーンファイルを置いているディレクトリ
    recursion yes;                     →再帰問い合わせに回答する
    allow-query { any; };              →問い合わせの送信元を限定しない
};

zone "." IN{                           →ルートゾーンステートメント
    type hint;                         →タイプはヒント (キャッシュサーバ)
    file "named.ca";                  →ゾーンファイルの名前
};                                     →ゾーンファイルは/var/named/named.ca
```

```
[root@pubdns.inet ~]# cat /var/named/named.ca    →ゾーンファイルの中身を表示
(一部省略)
.                518400  IN      NS      a.root-servers.net.    →ルートサーバ 名
.                518400  IN      NS      b.root-servers.net.
.                518400  IN      NS      c.root-servers.net.
a.root-servers.net. 518400  IN      A      198.41.0.4            →ルートサーバ アドレス(v4)
b.root-servers.net. 518400  IN      A      199.9.14.201
c.root-servers.net. 518400  IN      A      192.33.4.12
```

### 【pc.aws.test】

```
[root@pc.aws.test ~]# cat /etc/resolv.conf        →/etc/resolv.conf の中身を表示
nameserver 10.0.0.11                            →問い合わせ DNS サーバ (キャッシュサーバ) を指定
```

### 【検証】 dig コマンドで確認 (nslookup でもよい)

```
[root@pubdns.inet ~]# dig +short www.google.co.jp @10.0.0.11 →キャッシュサーバ にアドレス問合せ
172.253.63.94
[root@pc.aws.test ~]# dig +short www.yahoo.co.jp @10.0.0.11
edge12.g.yimg.jp.
182.22.25.252
```

## ■演習 2 - 2 「dns.aws.test の設定：aws.test ドメイン内の名前解決ができる」

(cisco.test ドメインも同様)

### 【dns.aws.test】

```
[root@dns.aws.test ~]# cat /etc/named.conf        →設定ファイル内容
options{
    directory "/var/named";                        →ゾーンファイルの場所
    // recursion no;                               →再帰問い合わせを受け付けない
    allow-query { any; };
};
zone "aws.test" IN{                                →aws.test ゾーンステートメント
    type master;                                   →タイプはマスタ (プライマリ)
    file "aws.test.zone";                         →ゾーンファイル名 /var/named/aws.test.zone
};
[root@dns.aws.test ~]# cat /var/named/aws.test.zone →ゾーンファイル内容
$TTL 3H                                             →問い合わせキャッシュ有効期間は 3 時間
@ IN SOA dns.aws.test. root.aws.test. (          →ネームサーバと管理者メールアドレス
    2020100101      ;serial      ゾーンファイルのバージョン
```

```

1D          ;refresh
1H          ;retry
1W          ;expire
3H )        ;minimum

```

```

aws.test. IN NS dns.aws.test.      →aws.test.のメインのネームサーバは dns.aws.test.
aws.test. IN MX 10 srv.aws.test.   →aws.test.のメインのメールサーバは srv.aws.test. 10は優先度
dns.aws.test. IN A 10.0.1.11      →dns.aws.test.のアドレスは 10.0.1.11
srv.aws.test. IN A 10.0.1.12      →srv.aws.test.のアドレスは 10.0.1.12

```

### 【検証】

```

[root@pc.aws.test ~]# dig +short SOA aws.test. @10.0.1.11      →SOA 情報
dns.aws.test. root.aws.test. 2020100101 86400 3600 604800 10800
[root@pc.aws.test ~]# dig +short NS aws.test. @10.0.1.11      →ネームサーバ名
dns.aws.test.
[root@pc.aws.test ~]# dig +short MX aws.test. @10.0.1.11      →メールサーバ名
10 srv.aws.test.
[root@pc.aws.test ~]# dig +short dns.aws.test. @10.0.1.11     →ネームサーバアドレス
10.0.1.11
[root@pc.aws.test ~]# dig +short srv.aws.test. @10.0.1.11     →メールサーバアドレス
10.0.1.12

```

### ■演習 2 - 3 「dns.test の設定：test ドメイン内の名前解決ができる」

#### 【dns.test】

```

[root@dns.test ~]# cat /etc/named.conf      →設定ファイル内容
options{
    directory "/var/named";
    // recursion no;
    allow-query { any; };
};
zone "test" IN{
    type master;
    file "test.zone";
};
[root@dns.test ~]# cat /var/named/test.zone  →ゾーンファイル内容
$TTL 3H
@ IN SOA dns.test. root.test. (
    2020101301      ;serial
    1D              ;refresh

```

```

1H          ;retry
1W          ;expire
3H )        ;minimum
test. IN NS dns.test.          →test ドメインのネームサーバ とアドレス
dns.test. IN A 10.0.0.13
aws.test. IN NS dns.aws.test.  →配下の aws.test ドメインのネームサーバ とアドレス
dns.aws.test. IN A 10.0.1.11
cisco.test. IN NS dns.cisco.test.  →配下の cisco.test ドメインのネームサーバ とアドレス
dns.cisco.test. IN A 10.0.2.11

```

【検証】 pc.aws.test から cisco.test ドメインの名前確認

```
[root@pc.aws.test ~]# nslookup -type=SOA cisco.test 10.0.0.13
```

```

cisco.test
    origin = dns.cisco.test
    mail addr = root.cisco.test
    serial = 2020100801
    refresh = 86400
    retry = 3600
    expire = 604800
    minimum = 10800

```

```
[root@pc.aws.test ~]# dig +short NS cisco.test @10.0.0.13
```

```
dns.cisco.test.
```

```
[root@pc.aws.test ~]# dig +short MX cisco.test @10.0.0.13
```

```
10 srv.cisco.test.
```

```
[root@pc.aws.test ~]# dig +short dns.cisco.test @10.0.0.13
```

```
10.0.2.11
```

```
[root@pc.aws.test ~]# dig +short srv.cisco.test @10.0.0.13
```

```
10.0.2.12
```

## ■演習 2 - 4 「ルートサーバの設定：ルートサーバから test ドメイン内の名前解決ができる」

【rootdns.inet】

```
[root@rootdns.inet ~]# cat /etc/named.conf
```

→設定ファイル内容

```

options {
    directory "/var/named";
    allow-query { any; };
//      recursion no;
};

```

```
zone "." IN {
    type master;
    file "root.zone";
};
```

[root@rootdns.inet ~]# cat /var/named/root.zone →ゾーンファイル内容

```
$TTL 3H
@      IN      SOA      rootdns.inet.  root.inet. (
                        2020101501      ; serial
                        1D                ; refresh
                        1H                ; retry
                        1W                ; expire
                        3H )              ; minimum
.      IN      NS       rootdns.inet.    →ルートのネームサーバとアドレス
rootdns.inet.  IN      A       10.0.0.12
test.        IN      NS       dns.test.    →配下の test ドメインのネームサーバとアドレス
dns.test.    IN      A       10.0.0.13
```

【検証】 pc.aws.test から rootdns.inet へ cisco.test ドメインの問い合わせ

[root@pc.aws.test ~]# nslookup -type=SOA cisco.test 10.0.0.12

Non-authoritative answer:

```
cisco.test
    origin = dns.cisco.test
    mail addr = root.cisco.test
    serial = 2020100801
    refresh = 86400
    retry = 3600
    expire = 604800
    minimum = 10800
```

Authoritative answers can be found from:

```
cisco.test      nameserver = dns.cisco.test.
dns.cisco.test  internet address = 10.0.2.11
```

[root@pc.aws.test ~]# dig +short NS cisco.test @10.0.0.12

```
dns.cisco.test.
```

[root@pc.aws.test ~]# dig +short MX cisco.test @10.0.0.12

```
10 srv.cisco.test.
```

[root@pc.aws.test ~]# dig +short dns.cisco.test @10.0.0.12

```
10.0.2.11
```

[root@pc.aws.test ~]# dig +short srv.cisco.test @10.0.0.12

```
10.0.2.12
```

## ■演習 2 - 5 「キャッシュサーバ設定②：ルートサーバを rootdns.inet にして名前解決」

## 【pubdns.inet】

[root@pubdns.inet ~]# cat /etc/named.conf

→設定ファイル内容

```
options{
    directory "/var/named";
    recursion yes;
    allow-query { any; };
};
zone "." IN{
    type hint;
    file "named.hint";
};
```

[root@pubdns.inet ~]# cat /var/named/named.hint

→ゾーンファイル内容

```
$TTL 3H
@ IN SOA pubdns.inet. root.inet. (
    2020102001      ;serial
    1D              ;refresh
    1H              ;retry
    1W              ;expire
    3H )            ;minimum
```

. IN NS rootdns.inet.

→ルートサーバの NS レコードと A レコード

rootdns.inet. IN A 10.0.0.12

## 【検証】 pc.aws.test から pubdns.inet (キャッシュサーバ) へ名前解決

[root@pc.aws.test ~]# dig +short SOA aws.test. @10.0.0.11

dns.aws.test. root.aws.test. 2020100101 86400 3600 604800 10800

[root@pc.aws.test ~]# dig +short NS aws.test. @10.0.0.11

dns.aws.test.

[root@pc.aws.test ~]# dig +short MX aws.test. @10.0.0.11

10 srv.aws.test.

[root@pc.aws.test ~]# dig +short dns.aws.test. @10.0.0.11

10.0.1.11

[root@pc.aws.test ~]# dig +short srv.aws.test. @10.0.0.11

10.0.1.12

## ■ゾーンファイル省略形の書き方

書籍や検定問題などではゾーンファイルは省略して書くことが多い  
省略表記する場合は、SOA→NS→MX→A の順にまとめて記述する

\$ORIGIN を使うと、ドメイン名を「@」に置き換えることができる

「.」で終わらない部分はドメイン名を自動補完させる

先頭の「@」と\$ORIGIN は省略できる

【dns.aws.test】

[root@dns.aws.test ~]# cat /var/named/aws.test.zone

→ゾーンファイル（省略形）

\$TTL 3H

演習 2 と比較

\$ORIGIN aws.test.

@ IN SOA dns root (

2020100101 ;serial

1D ;refresh

1H ;retry

1W ;expire

3H ) ;minimum

IN NS dns

IN MX 10 srv

dns IN A 10.0.1.11

srv IN A 10.0.1.1

## ■逆引き

逆引きは必ず必要ではない

IP アドレスを反対の順番に並べる

先頭に逆引きのゾーンを表すサブドメイン名 (in-addr.arpa.) をつける

設定ファイルにゾーンステートメントを追加する

ゾーンファイルに SOA レコード、NS レコード、PTR レコードを記述

【dns.aws.test】

[root@dns.aws.test ~]# cat /etc/named.conf

→設定ファイルの内容

options{

directory "/var/named";

recursion no;

allow-query { any; };

};

zone "aws.test" IN{

type master;

file "aws.test.zone";

};

zone "1.0.10.in-addr.arpa." IN {

→逆引きのゾーンステートメントを追加

type master;

file "aws.test.rev";

};

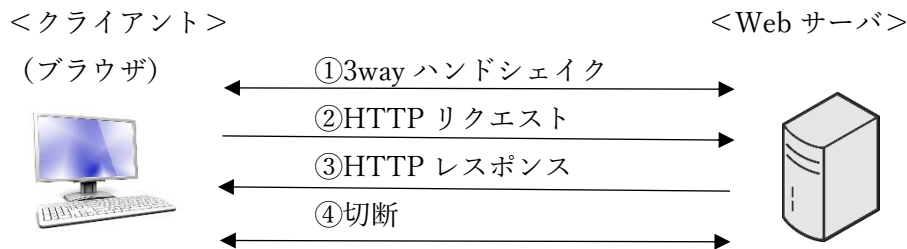
```
[root@dns.aws.test ~]# cat /var/named/aws.test.rev    →ゾーンファイルの内容
$TTL 3H
@ IN SOA dns.aws.test. root.aws.test. (              →SOA レコードはそのまま
    2020100101      ;serial
    1D              ;refresh
    1H              ;retry
    1W              ;expire
    3H )            ;minimum
1.0.10.in-addr.arpa. IN NS dns.aws.test.              →NS レコード記述
11.1.0.10.in-addr.arpa. IN PTR dns.aws.test.          →PTR レコード記述
12.1.0.10.in-addr.arpa. IN PTR srv.aws.test.
```

**【検証】**

```
[root@dns.aws.test ~]# dig +short SOA -x 10.0.1 @localhost
dns.aws.test. root.aws.test. 2020100101 86400 3600 604800 10800
[root@dns.aws.test ~]# dig +short NS -x 10.0.1 @localhost
dns.aws.test.
[root@dns.aws.test ~]# dig +short -x 10.0.1.11 @localhost
dns.aws.test.
[root@dns.aws.test ~]# dig +short -x 10.0.1.12 @localhost
srv.aws.test.
```

### 3. Web サーバ構築

## ■HTTP の仕組み（ステートレス通信）



■ Apache

Linux での Web サーバソフトウェアで、世界で最も使われている

- パッケージ名     httpd
- インストール  
\$ su -  
# yum -y install httpd
- 設定ファイル  
/etc/httpd/conf/httpd.conf
- 公開コンテンツ保存ディレクトリ  
/var/www/
- サービス起動（実行は 1 回で OK）  
# systemctl start httpd
- 自動起動 ON（実行は 1 回で OK）  
# systemctl enable httpd
- サービス再起動（設定ファイルを書き換えた後などに実行する）  
# systemctl restart httpd
- 設定ファイルのバックアップ（大事なファイルを変更する前には必ずバックアップする）  
# cp -p /etc/httpd/conf/httpd.conf /etc/httpd/conf/httpd.conf\_bk

## ■設定ファイル (/etc/httpd/conf/httpd.conf)

ディレクティブは1行に1つ記述

コメントは先頭に「#」（行の途中からコメント化はできない）

## <Directory “ディレクトリ”>

## ディレクティブ 設定内容

ディレクティブ 設定内容

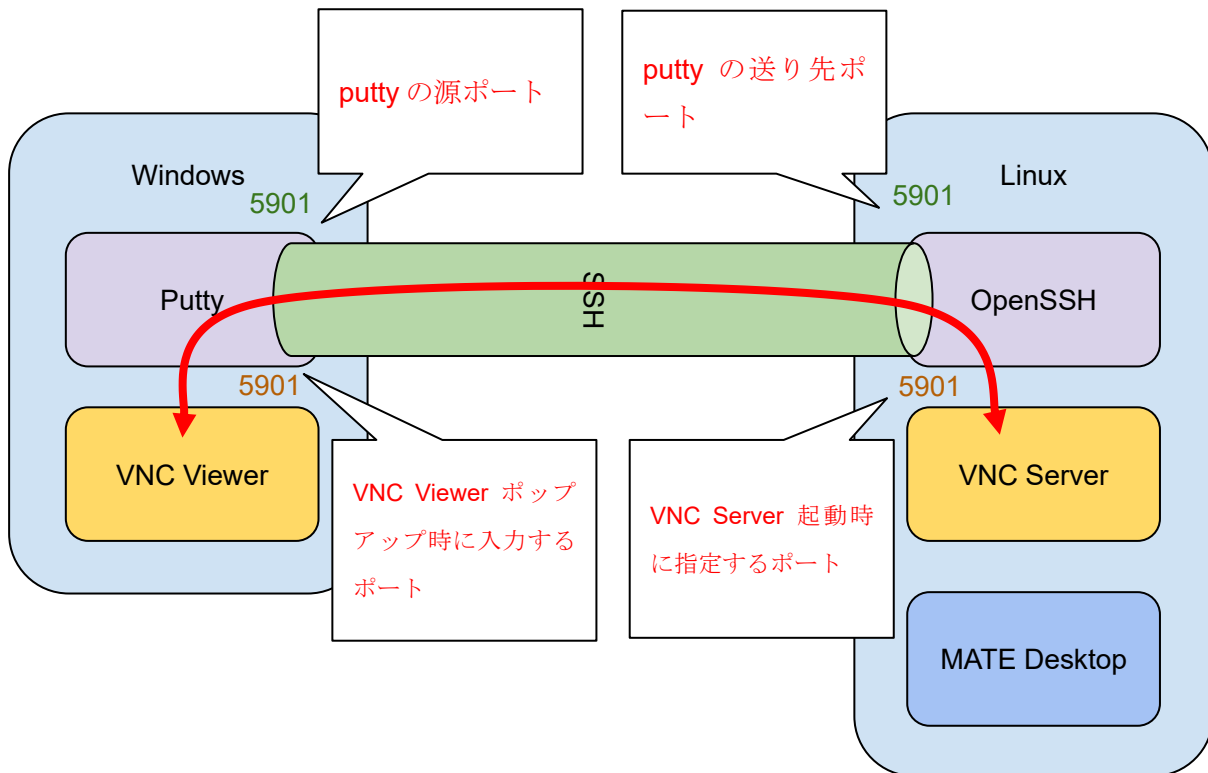
&lt;/Directory&gt;

- |                 |        |                                    |
|-----------------|--------|------------------------------------|
| ①DocumentRoot   | 119 行目 | 公開するホームページコンテンツを保存するディレクトリ指定       |
| ②ServerAdmin    | 86 行目  | クライアントに返すエラーメッセージ 内に記載する問い合わせ先アドレス |
| ③ServerName     | 95 行目  | サーバが自分自身を示すときに使うホスト名とポートを指定        |
| ④DirectoryIndex | 164 行目 | ディレクトリのみ指定されたときに使用するファイル名          |

## ■VNC(Virtual Network Computing)

ネットワーク上離れたコンピュータを GUI 遠隔操作するプロトコル  
Windows の場合はリモートデスクトップと呼ばれている

## ■VNC を使えるようにする手順



### 【VNC サーバ側】

```
sudo passwd root
→パスワード設定

su -
→特権モードへ

amazon-linux-extras install -y mate-desktop1.x
echo PREFERRED=/usr/bin/mate-session > /etc/sysconfig/desktop
→mate-desktop のインストールと初期設定

yum -y install tigervnc-server
→VNC Server インストール

mkdir /home/ec2-user/.vnc
chown ec2-user:ec2-user /home/ec2-user/.vnc
chmod 755 /home/ec2-user/.vnc
→VNC 用の隠しフォルダを作成し、所有者、パーミッションを変更

echo ×××× | vncpasswd -f > /home/ec2-user/.vnc/passwd
chown ec2-user:ec2-user /home/ec2-user/.vnc/passwd
```

```
chmod 600 /home/ec2-user/.vnc/passwd
```

→VNC 用のパスワードファイルを作成し、所有者、パーミッションを変更  
パスワードは「××××」

```
cp /lib/systemd/system/vncserver@.service /etc/systemd/system/vncserver@.service
```

```
sed -i 's/<USER>/ec2-user/' /etc/systemd/system/vncserver@.service
```

→VNC Server の初期設定を行う

```
systemctl daemon-reload
```

```
systemctl enable vncserver@:1
```

```
systemctl start vncserver@:1
```

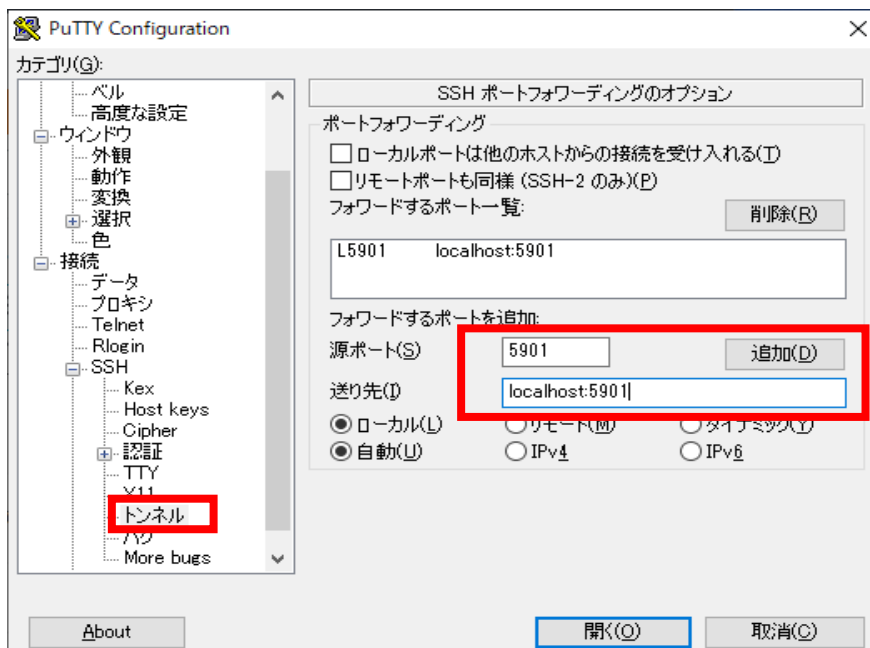
→VNC Server の自動起動 ON と起動を実行

起動時にポート番号 1 を指定（省略形なので本当は 5901）

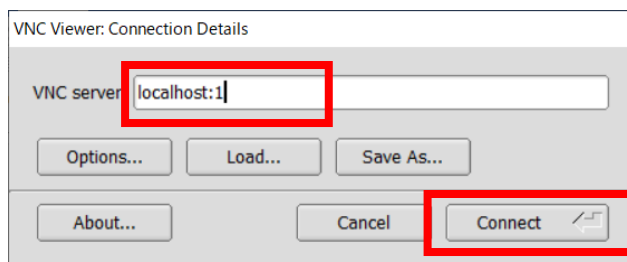
```
ss -atun | grep :5901
```

→5901 ポートがリッスンしてることを確認

## 【PuTTY】



## 【TigerVNC】



### ■演習 3 - 1 「srv.aws.test を Web サーバにする」

#### 【srv.aws.test】

[root@srv.aws.test ~]# cat /etc/httpd/conf/httpd.conf →設定ファイル内容 (一部)

31 行目 ServerRoot "/etc/httpd" →Apache がインストールされているディレクトリ

42 行目 Listen 80 →80 番ポートをリッスンしている

66 行目 User apache

67 行目 Group apache

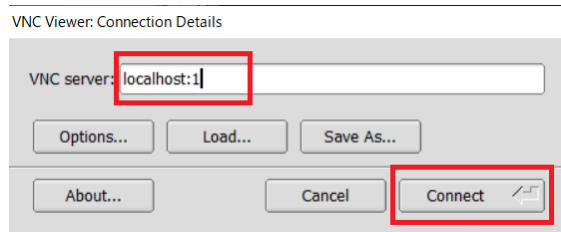
86 行目 ServerAdmin root@aws.test:80 →問い合わせ先メールアドレスを変更

95 行目 ServerName srv.aws.test:80 →ホスト名とポートを変更

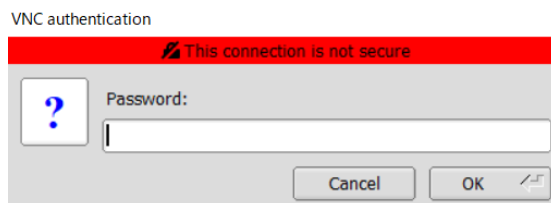
119 行目 DocumentRoot "/var/www/html" →トップページの html ファイルを保存するディレクトリ

164 行目 DirectoryIndex index.html →ディレクトリのみ指定された時のファイル名

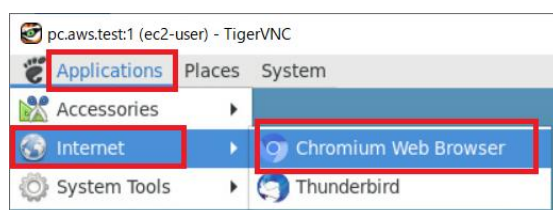
【検証】クライアント PC のブラウザから srv.aws.test にアクセスしてウェルカムページを表示  
(TigerVNC で接続し、サーバの IP アドレスにアクセス)



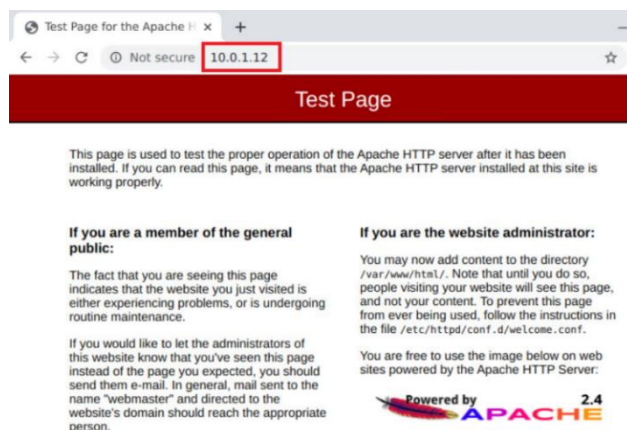
pc.aws.test に SSH 接続後、  
TigerVNC で接続



パスワード入力



ブラウザを開く



サーバの IP アドレス入力  
してウェルカムページ表示

## ■ ウェルカムページの変更

ウェルカムページをそのまま表示させると、サーバ情報が知られてしまう危険性がある

ウェルカムページ削除（今回は名前変更）

```
# mv /etc/httpd/conf.d/welcome.conf /etc/httpd/conf.d/welcome.conf_bk
```

代替りのデフォルトページを作成

```
# vi /var/www/html/index.html
```

※デフォルトページが存在しない場合、ディレクトリの階層が丸見えになり危険

/etc/httpd/conf/httpd.conf ファイル内の Options Indexes FollowSymLinks をコメント化(#)

## ■ ログ情報

イベントが発生するたびにログファイルに保存している

- ・アクセスログ     /var/log/httpd/access\_log
- ・エラーログ        /var/log/httpd/error\_log
- ・ログをリアルタイムに確認     tail -f /var/log/httpd/access\_log

## ■ 時刻合わせ

ログ管理では時刻が正しいことが前提である

- ①date コマンドで現在時刻を確認
- ②タイムゾーン設定ファイルのひな型を所定の場所にコピーする

```
cp -p /usr/share/zoneinfo/Japan /etc/localtime
```

- ③/etc/sysconfig/clock を編集

```
#ZONE="Asia/Tokyo"
```

```
#UTC=false
```

## ■ Apache 認証

認証を受けたユーザのみアクセスできるようにする

- ・Basic 認証
  - パスワードを暗号化しない
  - 機密性の高いデータ認証には適さない
- ・Digest 認証
  - パスワードを暗号化する
  - 対応するサーバとクライアント（Web ブラウザ）が必要

## ■ 認証手順

- ①/var/www/html に認証ディレクトリ作成
- ②①のディレクトリに index.html 作成
- ③/etc/httpd/conf に認証ファイル作成
  - (Basic は.htpasswd、Digest は.htdigest)
  - 認証用 ID とパスワードを登録するファイル
- ④/etc/httpd/conf.d に認証設定ファイル作成
  - (Basic は auth\_basic.conf、Digest は auth\_digest.conf)

認証ディレクトリやタイプ等を指定するファイル

⑤http://IP アドレス/ディレクトリにアクセスすると ID とパスワード入力を求められる

### ■演習 3 - 2 「Digest 認証：srv.aws.test (Web サーバ) に認証サイトを構築する」

#### 【srv.aws.test】

```
# mkdir /var/www/html/secretld      →Digest 認証ディレクトリ作成
# vi /var/www/html/secretld/index.html  →index.html 作成
<html>
  <body bgcolor="#e0ffff">
    AWS Digest Authentication Page
  </body>
</html>

# htdigest -c /etc/httpd/conf/.htdigest 'Digest-Auth' studentd  →認証ファイル作成
→2 回目以降は-c オプション不要      ↑ 認証ドメイン文字列と認証ユーザ ID
→パスワードを入力すると、ID とパスワードがセットで.htdigest ファイルに記録される

# vi/etc/httpd/conf.d/auth_digest.conf  →認証設定ファイルを新規作成
<Directory "/var/www/html/secretld">  →認証ディレクトリ (絶対パス)
  AuthType Digest                      →認証タイプ
  AuthName "Digest-Auth"              →認証ドメイン文字列
  AuthUserFile "/etc/httpd/conf/.htdigest"  →認証用ファイル (絶対パス)
  require valid-user                  →認証に成功した全ユーザを許可する
</Directory>
```

#### 【検証】

VNC 接続したクライアント PC から http://10.0.1.12/secretld にアクセス

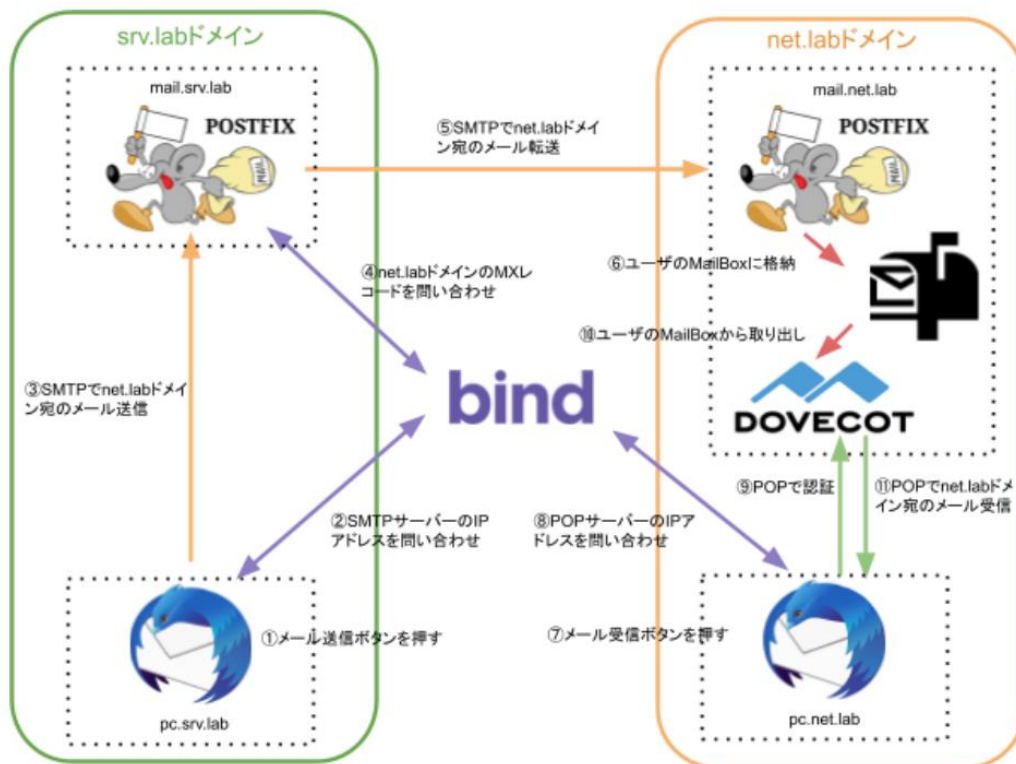
認証 ID とパスワードを入力するとページが表示される

## 4. MAIL サーバ構築

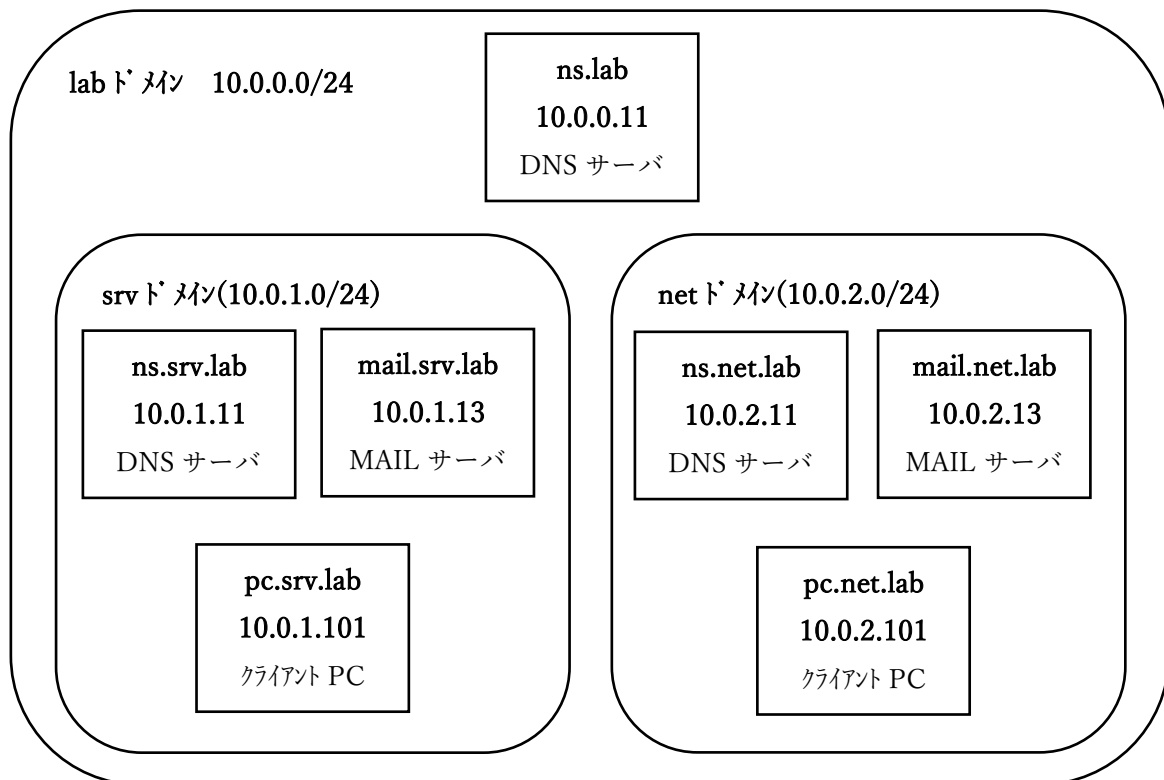
### ■メール関連ソフトウェア

- ・ Postfix  
オープンソースの送信用のメールサーバー用ソフトウェア
- ・ Dovecot  
オープンソースの受信用のメールサーバー用ソフトウェア
- ・ Thunderbird  
Mozilla 提供のオープンソースのクライアント用メールソフト

### ■メール配信の流れ



## ■演習構成図



■演習 4 - 1 「メールサーバ設定①：taro@srv.labo から自分宛にメールを送受信する」  
【mail.srv.lab】

su -

→特権モード（root ユーザ）に切り替え

yum -y install dovecot mailx

→dovecot,mailx をインストール

sed -i -e 's@nameserver 8.8.8.8@nameserver 10.0.0.11@' /etc/resolv.conf

→DNS を 10.0.0.11 に変更

dig +short pc.srv.lab

→「10.0.1.101」が表示されること

cp -p /etc/postfix/main.cf{,\_bk}

→設定ファイルバックアップ

sed -i -e 's@#myhostname = host.domain.tld@myhostname = mail.srv.lab@' /etc/postfix/main.cf

→ホスト名変更

sed -i -e 's@#mydomain = domain.tld@mydomain = srv.lab@' /etc/postfix/main.cf

→ドメイン名変更

sed -i -e 's@inet\_interfaces = localhost@inet\_interfaces = all@' /etc/postfix/main.cf

→ポート 25 を全 I/F でリッスン開始

sed -i -e 's@^mydestination = \$myhostname, localhost.\$mydomain, localhost@&, \$mydomain@'

```
/etc/postfix/main.cf
```

→宛先メールアドレスとしてドメイン (srv.lab) を許可

```
sed -i -e 's@#mynetworks = 168.100.189.0/28, 127.0.0.0/8@mynetworks = 10.0.0.0/16, 127.0.0.0/8@' /etc/postfix/main.cf
```

→転送元 NW として 10.0.0.0/16 を許可

```
sed -i -e 's@#home_mailbox = Maildir/@home_mailbox = Maildir/@' /etc/postfix/main.cf
```

→メールボックスの場所を設定

```
diff /etc/postfix/main.cf{,_bk}
```

→設定の差分を確認

```
systemctl restart postfix
```

→サービスを再起動し、設定を反映

```
cp -p /etc/dovecot/conf.d/10-mail.conf{,_bk}
```

→設定ファイルバックアップ

```
sed -i -e 's@#mail_location = maildir:~/Maildir@mail_location = maildir:~/Maildir@' /etc/dovecot/conf.d/10-mail.conf
```

→メールボックスの場所を設定

```
diff /etc/dovecot/conf.d/10-mail.conf{,_bk}
```

→設定の差分を確認

```
cp -p /etc/dovecot/conf.d/10-auth.conf{,_bk}
```

→設定ファイルバックアップ

```
sed -i -e 's@#disable_plaintext_auth = yes@disable_plaintext_auth = no@' /etc/dovecot/conf.d/10-auth.conf
```

→平文認証を許可

```
diff /etc/dovecot/conf.d/10-auth.conf{,_bk}
```

→設定の差分を確認

```
cp -p /etc/dovecot/conf.d/10-ssl.conf{,_bk}
```

→設定ファイルバックアップ

```
sed -i -e 's@ssl = required@ssl = no@' /etc/dovecot/conf.d/10-ssl.conf
```

→SSL ではない POP 接続を許可

```
diff /etc/dovecot/conf.d/10-ssl.conf{,_bk}
```

→設定の差分を確認

```
systemctl start dovecot
```

→サービス起動

```
systemctl enable dovecot
```

→サービス自動起動 ON

```
useradd taro
```

→taro ユーザ追加

```
passwd taro
```

→PW 設定「taro123」

## 【pc.srv.lab】

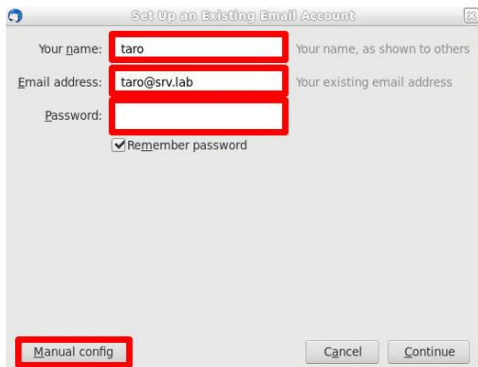
Putty のトンネルのローカル設定 L5901

Putty のトンネルのリモート設定 localhost:5901

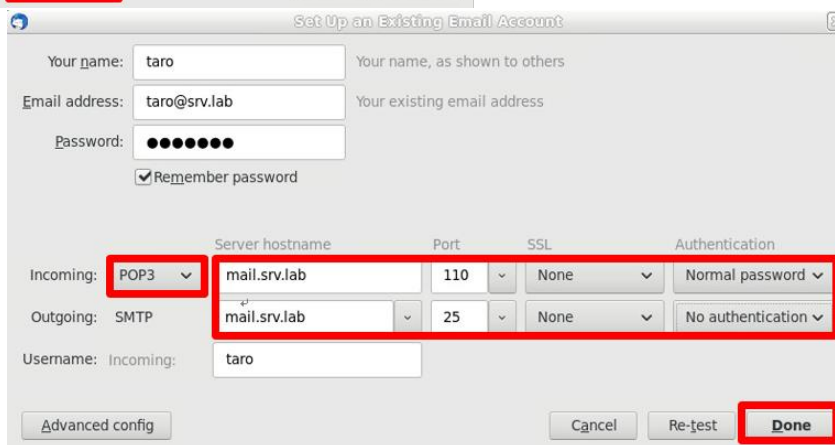
TigerVNC 接続設定 localhost:1

## 【検証】

pc.srv.lab に VNC 接続し、Thunderbird で taro@srv.lab 宛にメールを送る

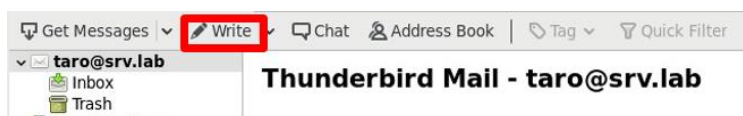


①ユーザ taro@srv.lab 設定

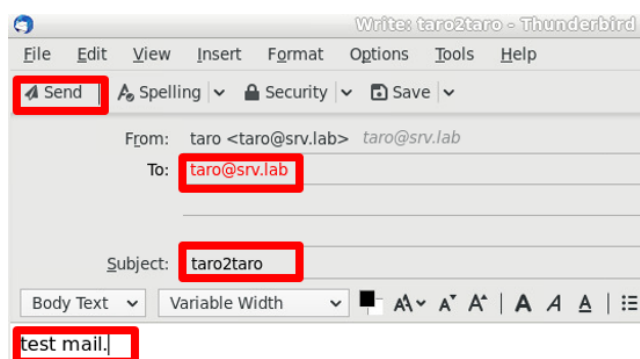


②サーバ名、ポート等  
設定

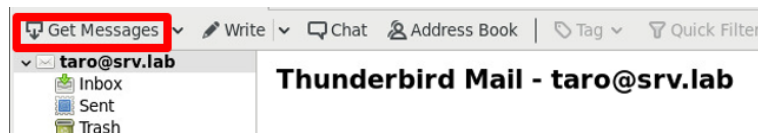
③設定についての Warning!の画面が出るが、チェックして Done



④Write



⑤自分宛メール送信



⑥メール受信

## ■演習 4 - 2 「メールサーバ設定②：taro@srv.lab から jiro@net.lab へメールを送受信する」 【mail.net.lab】

演習 4 - 1 と同じ内容を設定（ユーザ情報のみ以下）

useradd jiro →taro ユーザ追加

passwd jiro →PW 設定「jiro123」

### 【pc.net.lab】

Putty のトンネルのローカル設定 L5902

Putty のトンネルのリモート設定 localhost:5901

TigerVNC 接続設定 localhost:2

### 【検証】

pc.srv.lab に VNC 接続し、Thunderbird で jiro@net.lab 宛にメールを送る

pc.net.lab に VNC 接続し、Thunderbird でメールを受信する

## ■SASL（サスル）

認証フレームワークで、様々なプロトコルやソフトウェアに認証機能を提供

SMTP はデフォルトで認証に対応していない

SASL を使用して SMTP 認証を有効化すると、メール送信時にパスワードの入力が必要  
スパムメール対策となる

## ■演習 4 - 3 「SASL 設定：taro@srv.lab からメールを送るときにパスワード入力を求める」 【srv.net.lab】

su -

→特権モード（root ユーザ）に切り替え

cp -p /etc/postfix/main.cf{,\_bk2}

→main.cf を main.cf\_bk2 としてバックアップ

-----ここから-----

echo "

##smtpauth setting

#SASL 認証を有効にする

smtpd\_sasl\_auth\_enable = yes

#認証成功している場合許可、認証失敗は拒否

smtpd\_recipient\_restrictions = permit\_sasl\_authenticated, reject\_unauth\_destination

#Anonymous 認証を拒否

smtpd\_sasl\_security\_options = noanonymous

#outlook で SASL 認証を使えるようにする。

```
broken_sasl_auth_clients = yes
```

#SASL 認証で利用するドメインを指定

```
smtpd_sasl_local_domain = ¥$myhostname
```

#Dovecot-SASL を使用する

```
smtpd_sasl_type = dovecot
```

#Dovecot-SASL で使用するソケットファイル

```
smtpd_sasl_path = private/auth
```

#VRFY コマンドを無効化

```
disable_vrfy_command = yes" >> /etc/postfix/main.cf
```

-----ここまで 1 つのコマンド-----

→SASL 設定を main.cf に追加 (vi /etc/postfix/main.cf から追記してもよい)

```
diff /etc/postfix/main.cf{,_bk2}
```

→設定変更の差分を確認

```
systemctl restart postfix
```

→postfix を再起動

```
cp -p /etc/dovecot/conf.d/10-master.conf{,_bk}
```

→10-master.conf を 10-master.conf\_bk としてバックアップ

```
sed -i -e '96,98s@#@@@' /etc/dovecot/conf.d/10-master.conf
```

→96 行目、98 行目の#を削除

```
sed -i -e '98i ¥ user = postfix' /etc/dovecot/conf.d/10-master.conf
```

→98 行目に「user = postfix」を追加

```
sed -i -e '99i ¥ group = postfix' /etc/dovecot/conf.d/10-master.conf
```

→99 行目に「group = postfix」を追加

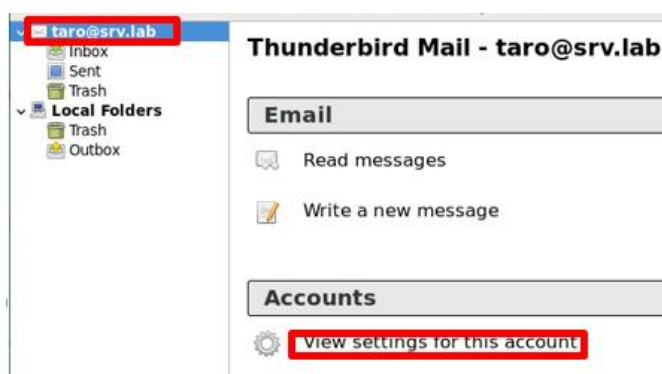
```
diff /etc/dovecot/conf.d/10-master.conf{,_bk}
```

→設定変更の差分を確認

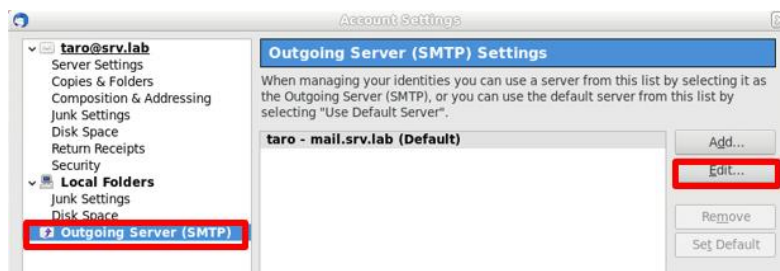
```
systemctl restart dovecot
```

→dovecot を再起動

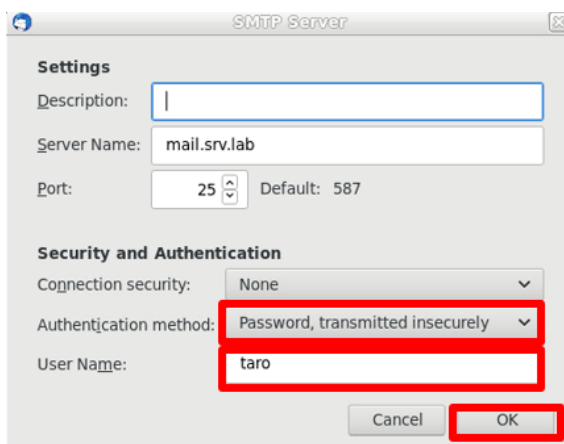
## 【pc.srv.lab】



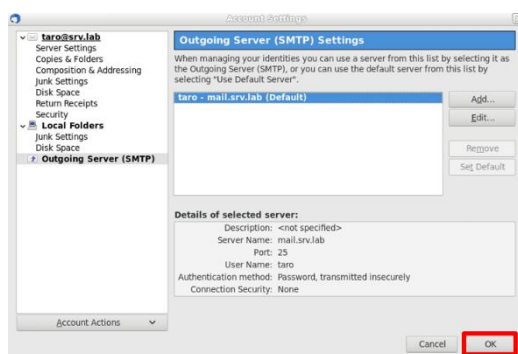
VNC 接続して Thunderbird 起動  
taro アカウントの設定



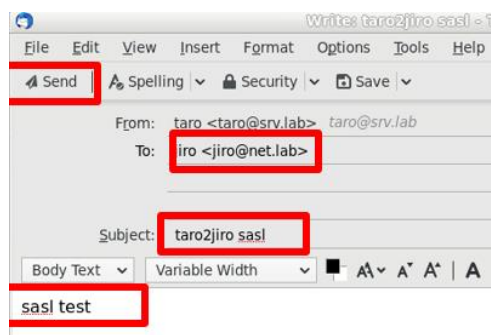
SMTP 編集



認証方法を指定する



【検証】 taro@srv.lab からメールを送るとパスワード入力を求められる

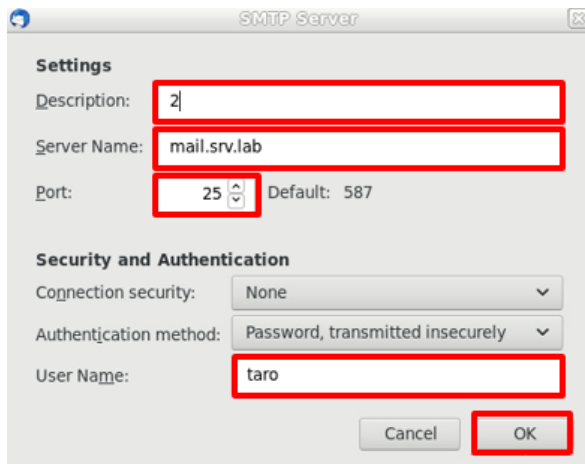


パスワード入力しないとメールが送れない

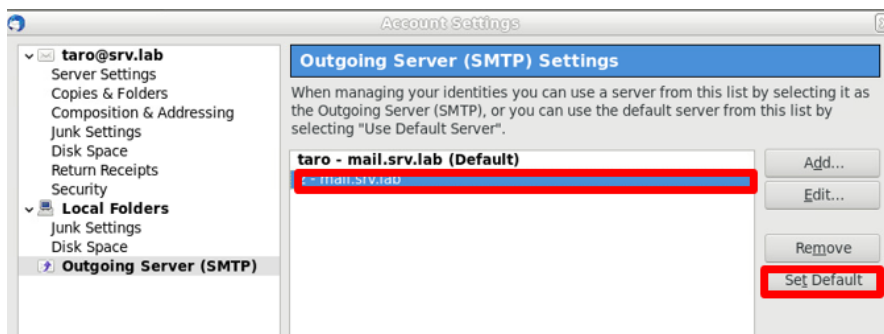
## ■SASL の送信 PW を削除する手順



SMTP 選択して Add



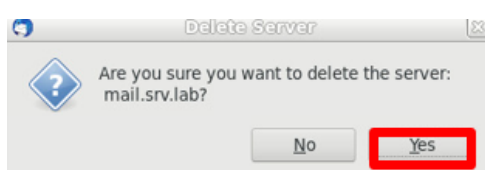
Description を任意の別文字にする



追加したものを  
SetDefault



以前からのものを  
Remove



## 5. ファイルサーバ構築

### ■Samba (サンバ)

Windows のファイル共有プロトコルである SMB(CIFS)を Linux で使用できるようにするソフト

Samba を使うと Windows のエクスプローラから Linux のファイルシステムにアクセスできる

### ■演習 5 - 1 「Samba 設定①：Samba インストールと Samba ユーザ追加」

【samba】（インスタンスを新規作成）

```
sudo passwd root
```

→root ユーザの PW 設定

```
su -
```

→root ユーザに変更

```
yum -y install samba
```

→Samba をインストール

```
systemctl start smb
```

→Samba を起動

```
systemctl enable smb
```

→Samba の自動起動を有効化

```
useradd taro
```

→Linux ユーザを追加 ※Samba ユーザと Linux ユーザは別だが、  
Samba ユーザと同一名の Linux ユーザがないと Samba ユーザは追加できない

```
passwd taro
```

→taro ユーザの PW 設定「taro123」

```
pdbedit -a taro
```

→samba ユーザ追加、PW「taro123」

【検証】

taro による SMB アクセス確認

<Windows>

エクスプローラーのアドレスバーに「¥¥<samba のパブリック IP>」と入力  
ID / PW 「taro / taro123」

taro フォルダが見えることを確認

taro フォルダに test フォルダを新規作成

<samba>

taro ユーザのホームディレクトリに test ディレクトリがあることを確認

```
ll /home/taro
```

## ■パーミッション

ディレクトリとファイルに対して、ユーザーとグループごとにアクセス権限を与えること

## ■パーミッションの種類

左の3文字：所有者のアクセス権限

中央3文字：所有グループのアクセス権限

右の3文字：その他ユーザ・グループのアクセス権限

r：読み込み可

w：書き込み可

x：実行可

-：アクセス権限なし

## ■パーミッションのパターン指定

使用例 `$ chmod a+w hello.txt` ※「すべてのユーザに、書き込みを追加する」

①誰に、②追加・削除する ③何の操作を

### ①【権限を変えるユーザー】

a：すべてのユーザー

u：所有者のみ

g：所有グループのみ

o：その他のユーザー

### ②【権限に対する操作】

＋：追加する

-：削除する

### ③【操作する権限】

r：読み込み権限

w：書き込み権限

x：実行権限

## ■パーミッションのパターン指定（数値）

使用例 `$ chmod 666 hello.txt`

※「666」はすべてのユーザに書込と読取を許可し、実行を許可しない

権限 2進数 10進数

--- 000 0

--x 001 1

-w- 010 2

-wx 011 3

r-- 100 4

r-x 101 5

rw- 110 6

rwx 111 7

## ■演習 5 - 2 「Samba 設定②：共有ディレクトリ作成（誰でもアクセス可）」

## 【samba】

```
mkdir /home/share
```

→共有ディレクトリ作成

```
chmod 777 /home/share
```

→パーミッションを 777 に変更

```
vi /etc/samba/smb.conf
```

→以下を最終行に追加

```
-----
```

```
[share]
```

```
#コメント
```

```
comment = Public Space
```

```
#ネットワーク上の表示を有効化
```

```
browseable = yes
```

```
#ファイル作成時のパーミッション設定、ファイルは最大 766
```

```
create mode = 0766
```

```
#ディレクトリ作成時のパーミッション設定、ディレクトリは最大 777
```

```
directory mode = 0777
```

```
#設定を適用するディレクトリのパス
```

```
path = /home/share
```

```
#パブリック（共有）設定を有効化
```

```
public = yes
```

```
#ゲスト設定を有効化、所有者が nobody となる
```

```
only guest = yes
```

```
#読込のみを無効化、書込 OK
```

```
read only = no
```

```
-----
```

→共有ディレクトリ用の設定を smb.conf に追加

```
systemctl restart smb
```

→Samba 再起動

## 【検証】

<Windows>

エクスプローラーで taro フォルダの横に share フォルダが見えることを確認

share フォルダに taro ディレクトリを作成

<samba>

```
ll /home/share
```

share ディレクトリに taro ディレクトリがあることを確認

パーミッションが 777 で所有者が nobody であることを確認

## ■ユーザの認証情報削除

Windows は一度アクセスしたら ID/PW を記憶してしまう

他のユーザのアクセスを試そうとした際に以前アクセスしたユーザで自動的にアクセスする

他のユーザでアクセスしたい場合、認証情報の削除を実行する必要がある

<Windows コマンドプロンプト> ※エクスプローラーを閉じて作業すること

認証情報の表示      net use

認証情報削除        net use <リモート名> /delete

## ■演習 3 – 「Samba 設定③：特定グループの共有ディレクトリ作成」

### 【samba】

groupadd srv

→srv グループ作成

usermod -G srv taro

→taro を srv グループに追加

usermod -G srv jiro

→jiro を srv グループに追加

useradd goro

→srv グループに所属しない goro ユーザを追加

passwd goro

→goro ユーザの PW 設定「goro123」

pdbedit -a goro

→samba ユーザ追加、PW「goro123」

mkdir /home/srv

→srv グループ用のディレクトリ作成

chmod 770 /home/srv

→所有ユーザと所有グループのみアクセス可能なパーミッション 770 に変更

chown :srv /home/srv

→所有グループ変更

vi /etc/samba/smb.conf

→最終行に追加

-----  
[srv]

#コメント

comment = Srv Group Space

#ネットワーク上の表示を有効化

browseable = yes

#ファイル作成時のパーミッション設定、ファイルは最大 666

create mode = 0660

#ディレクトリ作成時のパーミッション設定、ディレクトリは最大 777

directory mode = 0770

#設定を適用するディレクトリのパス

path = /home/srv

#srv グループのみ許可

valid users = @srv

#読込のみを無効化、書込 OK

read only = no

-----

→srv グループ用ディレクトリの設定を smb.conf に追加

systemctl restart smb

→Samba 再起動

## 【検証】

<Windows>

エクスプローラーを閉じ、認証情報を削除

taro ユーザでアクセス

エクスプローラーで taro フォルダの横の srv フォルダにアクセスできることを確認

srv フォルダに taro ディレクトリを作成

エクスプローラーを閉じ、認証情報を削除

jiro ユーザでアクセス

エクスプローラーで jiro フォルダの横の srv フォルダにアクセスできることを確認

srv フォルダに jiro ディレクトリを作成

エクスプローラーを閉じ、認証情報を削除

goro ユーザでアクセス

エクスプローラーで goro フォルダの横の srv フォルダにアクセスできないことを確認

<samba>

ll /home/srv

srv ディレクトリに taro ディレクトリ、jiro ディレクトがあることを確認

パーミッションが 770 で所有者がそれぞれのユーザであることを確認

## 6. FTP サーバ構築

### ■FTP (File Transfer Protocol) の仕組み

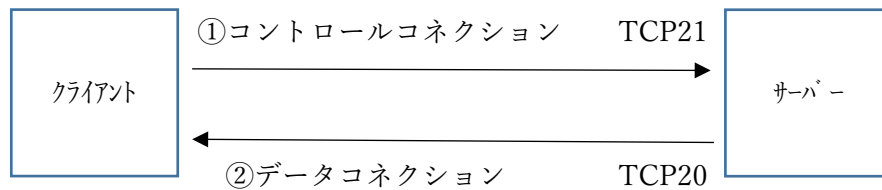
2つのポート番号を使い、離れた場所同士でファイルを送受信するためのプロトコル

TCP21：制御（ファイルを送受信するための命令用）

TCP20：データ（ファイル送受信用）

### ■アクティブモード

- ・アクティブモード：21 番クライアント→サーバー、20 番サーバー→クライアント



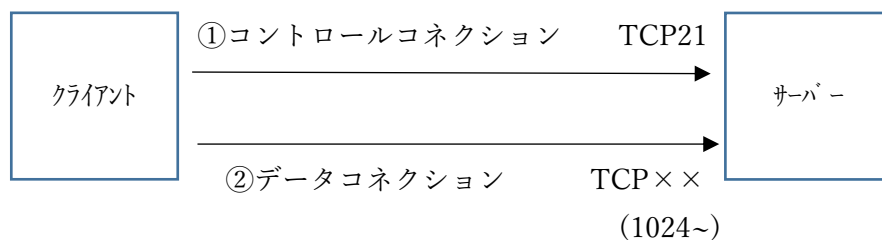
※クライアント側のファイアウォールで 20 番の通信が止められる問題がある

インターネットが通信の起点となる通信は基本的に止められる

（クライアント発の通信の返りは許可される）

### ■パッシブモード

- ・パッシブモード：21 番/20 番の代わりとなるポートともにクライアント→サーバー



※パッシブモードは、TCP20 ポートは使わない

一般的な環境ではパッシブモードでないとつながらない場合が多い

### ■FTP コマンド

FTP コマンドもしくは FTP ソフトを使ってファイルを転送

Windows では FTP コマンドをコマンドプロンプトから実行可能

ftp <FTP サーバーの IP or ホスト名>

→FTP サーバーに接続

※コマンドプロンプトやターミナルから FTP 接続した場合は、FTP 接続前の

クライアント側（ローカル）のカレントディレクトリを意識することが重要

ex) Windows の場合は「¥¥users¥<ユーザ>」がカレントディレクトリ

ダウンロードする場合、カレントディレクトリにファイルがコピーされる

アップロードする場合、カレントディレクトリのファイルをアップロードできる

※Linux に存在するユーザーでログイン可能

許可すれば匿名を意味する「anonymous」ユーザでもログイン可能で、PW 不要

ls

→FTP サーバー側（リモート）のファイル、ディレクトリ一覧を表示

!dir

→クライアント側（ローカル）のファイル、ディレクトリ一覧を表示

※「!」を付けるとクライアント側（ローカル）を意味する。

pwd

→FTP サーバー側（リモート）のカレントディレクトリを表示

※FTP 接続直後の FTP サーバー側（リモート）のカレントディレクトリは

ログインユーザのホームディレクトリ

ex) taro の場合「/home/taro」、anonymous の場合「/var/ftp」

cd <ディレクトリ>

→FTP サーバー側（リモート）のディレクトリ移動

lcd <ディレクトリ>

→クライアント側（ローカル）のディレクトリ移動

get <ファイル>

→FTP サーバー側（リモート）のファイルをダウンロード

put <ファイル>

→クライアント側（ローカル）のファイルをアップロード

delete <ファイル>

→FTP サーバー側（リモート）のファイルを削除

mkdir <ディレクトリ>

→FTP サーバー側（リモート）にディレクトリ作成

rmdir <ディレクトリ>

→FTP サーバー側（リモート）のディレクトリ削除

type

→現在の転送モード（ascii or binary）を表示

binary

→転送モードを binary に変更、データをそのまま転送するモード

ascii

→転送モードを ascii に変更、テキストデータを送るモード

改行コードを自動的に修正して転送するモード

※画像や実行ファイルを ascii モードで送ると壊れる可能性がある。

bye(quit)

→FTP 接続を終了

## ■演習 6 - 1 「FTP 設定①：VSFTPD インストールと anonymous 関連の設定」

前の実習で使った samba インスタンスを使用

### 【samba】

su -

yum -y install vsftpd

vi /etc/vsftpd/vsftpd.conf

→anonymous によるアップロードおよびディレクトリ作成を有効化、「#」を外す

```
-----
anon_upload_enable=YES
```

```
anon_mkdir_write_enable=YES
-----
```

systemctl start vsftpd

systemctl enable vsftpd

chmod 777 /var/ftp/pub

→/var/ftp/pub のパーミッションを 777 に設定

### 【検証】

<Windows コマンドプロンプト>

echo windows > ftp.txt

→「windows」という内容の ftp.txt を作成

ftp <samba のパブリック IP>

→taro/taro123 でログイン

pwd

→カレントディレクトリが/home/taroであることを確認

ls

→FTP サーバー側（リモート）の/home/taro のファイル、ディレクトリ一覧を表示

!dir

→クライアント側（ローカル）のカレントディレクトリのファイル、ディレクトリ一覧を表示

get test

→test ファイルをダウンロード

!dir

→クライアント側（ローカル）のカレントディレクトリのファイル、ディレクトリ一覧を表示

test が存在すること

put ftp.txt

→ftp.txt をアップロード

ls

→FTP サーバー側（リモート）の/home/taro のファイル、ディレクトリ一覧を表示

ftp.txt が存在すること

mkdir ftpdir

```

→FTP サーバー側（リモート）の/home/taro に ftpdir を作成
ls
→FTP サーバー側（リモート）の/home/taro のファイル、ディレクトリ一覧を表示
ftpdir が存在すること
bye
→FTP を切断
ftp <samba のパブリック IP>
→anonymous/pw 無しでログイン
pwd
→カレントディレクトリが「/」であることを確認（本当は/var/ftp）
ls
→FTP サーバー側（リモート）の/var/ftp のファイル、ディレクトリ一覧を表示
pub があることを確認
cd pub
→FTP サーバー側（リモート）のカレントディレクトリを pub に移動（本当は/var/ftp/pub）
put ftp.txt
→ftp.txt をアップロード
ls
→FTP サーバー側（リモート）の pub（本当は/var/ftp/pub）のファイル、ディレクトリ一覧を表示
ftp.txt が存在すること
mkdir ftpdir
→FTP サーバー側（リモート）の pub（本当は/var/ftp/pub）に ftpdir を作成
ls
→FTP サーバー側（リモート）の pub（本当は/var/ftp/pub）のファイル、ディレクトリ一覧を表示
ftpdir が存在すること
bye
→FTP を切断

```

## ■演習 6 - 2 「FTP 設定②：EC2 インスタンスを FTP クライアントにする」

samba を起動させてから作業すること

EC2→インスタンス→「samba」右クリック→Image and templates→同様のものを起動→起動

→キーペア「mail\_file」→☒→インスタンスの作成

新しいインスタンスの Name を「ftp\_client」に変更

「ftp\_client」クリック→「セキュリティ」タブ

→セキュリティグループ「xxxxxxx(samba-sg)」クリック→インバウンドルールを編集

→タイプ「すべてのトラフィック」→ソース「172.31.0.0/16」→ルールを保存

## 【ftp\_client】

```
sudo passwd root
```

→パスワード設定

```
su -
```

```
echo ftp_client > /etc/hostname
```

→再起動後も消えない名前を変更する

```
hostnamectl set-hostname ftp_client
```

→起動中に名前を変更する（暫定的なものなので、これだけでは再起動したら元に戻る）

```
exit
```

```
su -
```

→ログインし直すと名前が変わる

```
yum -y install ftp
```

→ftp コマンドをインストール

```
echo ftp_client > ftp_client_file
```

→「ftp\_client」という内容の ftp\_client\_file を作成

```
ftp <samba のプライベート IP>
```

→taro/taro123 でログイン

```
pwd
```

→カレントディレクトリが/home/taroであることを確認

```
ls
```

→FTP サーバー側（リモート）の/home/taro のファイル、ディレクトリ一覧を表示

```
!dir (!ls でも OK)
```

→クライアント側（ローカル）のカレントディレクトリのファイル、ディレクトリ一覧を表示

```
get test
```

→test ファイルをダウンロード ※test ファイルがない時は samba で「echo test > test」を実行

```
!dir (!ls でも OK)
```

→クライアント側（ローカル）のカレントディレクトリのファイル、ディレクトリ一覧を表示

test が存在すること

```
put ftp_client_file
```

→ftp\_client\_file をアップロード

```
ls
```

→FTP サーバー側（リモート）の/home/taro のファイル、ディレクトリ一覧を表示

ftp\_client\_file が存在すること

```
mkdir ftp_client_dir
```

→FTP サーバー側（リモート）の/home/taro に ftp\_client\_dir を作成

```
ls
```

→FTP サーバー側（リモート）の/home/taro のファイル、ディレクトリ一覧を表示

ftp\_client\_dir が存在すること

bye

→FTP を切断

## ■SFTP (SSH File Transfer Protocol)

SSH の TCP22 を使用してファイル転送を安全に行える。

sshd (SSH デモン) である OpenSSH を使用して利用可能

(OpenSSH は最初からインストールされている)

他の安全なファイル転送プロトコルとして SCP と FTPS がある

(SCP は今回の演習で触れるが、FTPS は証明書を使用するため今回は触れない)

SFTP と SCP は SSH が使用できる状態であれば使用できる

(SFTP は FTP コマンドに似ており、SCP は CP コマンドに似ている)

## ■SCP 書式

scp <コピー元> <コピー先>

公開鍵暗号方式を使用した場合は「-i」オプションで秘密鍵を指定する

scp -i <秘密鍵> <コピー元> <コピー先>

ex) 秘密鍵「mail\_file」を使う場合

scp -i mail\_file <コピー元> <コピー先>

<コピー先> がリモートの場合は「<ユーザ>@<IP アドレス>:<ファイル名>」という書式

scp -i <秘密鍵> <コピー元> <ユーザ>@<IP アドレス>:<ファイル名>

ex) 秘密鍵「mail\_file」を使い、カレントディレクトリのファイル「mail\_file」を、ユーザ「taro」

で IP 「1.2.3.4」にコピーする場合

scp -i mail\_file mail\_file taro@1.2.3.4:mail\_file

<コピー元> と <コピー先> でファイル名が同じで良い場合は「<ファイル名>」は省略可

scp -i mail\_file mail\_file taro@1.2.3.4:

## ■SFTP 書式

sftp <ユーザ>@<接続先 IP>

公開鍵暗号方式を使用した場合は「-i」オプションで秘密鍵を指定する。

sftp -i <秘密鍵> <ユーザ>@<接続先 IP>

ex) 秘密鍵「mail\_file」を使い、ユーザ「taro」で IP 「172.16.1.1」に接続する場合

sftp -i mail\_file taro@172.16.1.1

## ■演習 6 - 3 「SFTP 設定」

### 【samba】

su -

systemctl stop vsftpd

→SFTP には使用しないので、VSFTPD を停止する

(使用していないことを明確にするため、あえて停止する)

cp -rp /home/ec2-user/.ssh /home/taro/.ssh

→ec2-user のホームディレクトリにある「.ssh」を中身ごと taro のホームディレクトリにコピーする

#### ※SSH の鍵

- ・各ユーザの「.ssh」の中にある「authorized\_keys」に SSH 用の公開鍵が書き込んである
- ・ホームディレクトリに「.ssh/authorized\_keys」がなければ、そのユーザを使用して SSH ができない
- ・新しいキーペアを作った場合は「authorized\_keys」に公開鍵を追加すれば、そのキーペアを使用して SSH 可能
- ・「.ssh」と「authorized\_keys」はパーミッションに気をつける必要がある

.ssh = 700

authorized\_keys = 600

chown -R taro:taro /home/taro/.ssh

→パーミッションは上記の通りになっているが、所有者、所有グループが「ec2-user」であるため、「taro」に変更する

echo abcdef > /home/taro/sftpfile

→SFTP でダウンロードを試すためのファイル「sftpfile」を作成

#### <Windows コマンドプロンプト>

cd <mail\_file があるディレクトリ>

→mail\_file があるディレクトリに移動

scp -i mail\_file mail\_file ec2-user@<ftp\_client のパブリック IP>:

→秘密鍵を ftp\_client に SCP でコピーする

ftp\_client に秘密鍵を送ることで ftp\_client から samba に SFTP 接続できるようになる  
形式は PEM にすること

#### 【検証】

##### 【ftp\_client】

※「su」せずに実施する

pwd

→「/home/ec2-user」であること。

ll

→SCP でコピーした「mail\_file」があること。

chmod 600 mail\_file

→秘密鍵のパーミッションを「600」に変更

セキュリティのため、秘密鍵は「600」になっていないと使用できない

sftp -i mail\_file taro@<プライベート IP>

→秘密鍵「mail\_file」を使用し、ユーザ「taro」で<プライベート IP>に SFTP 接続  
「sftp>」のプロンプトが出ること。

※「sftp>」のプロンプトで実施。

pwd

→リモートのカレントディレクトリが「/home/taro」であること。

ls

→リモートに「sftpfile」があること。

get sftpfile

→ダウンロードできること。読込できることを確認。

!!s

→ローカルのカレントディレクトリに「sftpfile」があること。

mkdir sftpdire

→リモートのカレントディレクトリに「sftpdire」を作成。

ls

→リモートに「sftpdire」があること。書込できることを確認。

bye

→SFTP 接続を終了。