Basic Configuration

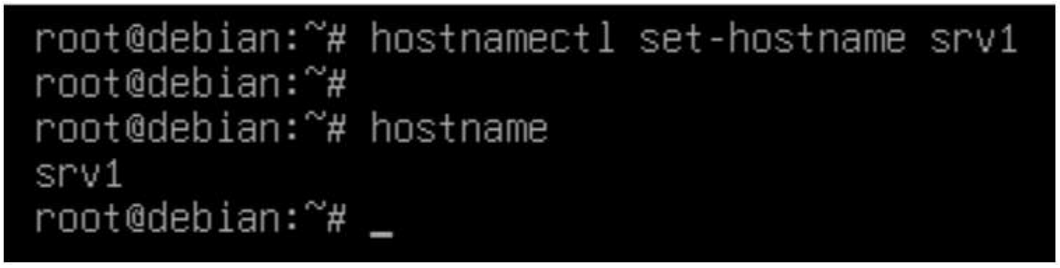
Pada dokumentasi ini akan dijelaskan beberapa *basic configuration* yang dapat dilakukan pada sistem operasi Linux Debian. Terdiri dari konfigurasi Hostname, Domain Name, Timezone, dan Basic Command. Sebelum melanjutkan praktikum pada materi selanjutnya, pemahaman akan *basic configuration* menjadi hal utama yang harus dipahami terlebih dahulu.

Konfigurasi Hostname

Digunakan untuk mengidentifikasi Linux server yang ada. Salah satu cara untuk memodifikasi hostname pada Linux server adalah menggunakan perintah `hostnamectl` yang memiliki sintaks seperti berikut:

```
hostnamectl set-hostname <NAMA_HOSTNAME>
```

Contoh konfigurasi hostname menggunakan `hostnamectl` pada **srv1**.



Dengan menggunakan perintah yang sama, lakukan konfigurasi hostname pada **srv2**

Domain name

IB

Linux Environment > Refresh Basic Configuration > Domain name

Step 1 of 1: Domain name

Konfigurasi Domain Name

Domain name digunakan sebagai identifikasi spesifik yang ditambahkan pada hostnamem, umumnya digunakan untuk menjelaskan lokasi atau grup server Linux berada. Gabungan antara Hostname dan Domain Name disebut sebagai Fully Qualified Domain Name.

Proses konfigurasi domain name dapat dilakukan dengan cara menambahkannya pada file /etc/hosts. Konfigurasi domain name yang ditambahkan memiliki format seperti berikut:

```
127.0.0.1  <NAMA_HOSTNAME>.<DOMAIN_NAME>  <NAMA_HOSTNAME>
```

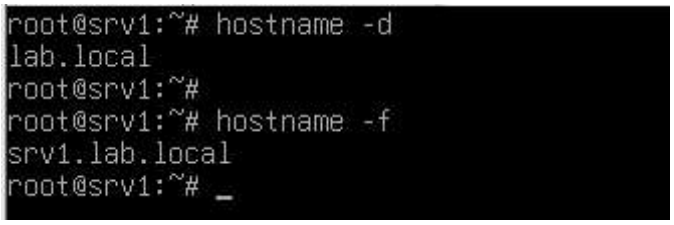
Konfigurasi pada srv1

Berikut contoh konfigurasi domain name **srv1** yang ditambahkan pada file /etc/hosts :

```
127.0.0.1       srv1.lab.local  srv1
```

```
# The following lines are desirable for IPv6 capable hosts
::1     localhost ip6-localhost ip6-loopback
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
127.0.0.1       srv1.lab.local srv1_
```

Untuk mengecek apakah domain name telah berhasil dikonfigurasi, gunakan perintah hostname -d sedangkan untuk mengetahui fully qualified domain name gunakan perintah hostname -f



Konfigurasi pada srv2

Untuk **srv2** , tambahkan konfigurasi berikut pada file /etc/hosts

```
127.0.0.1       srv2.lab.local  srv2
```

Lalu, check menggunakan perintah:

```
hostname -f
```

Konfigurasi timezone

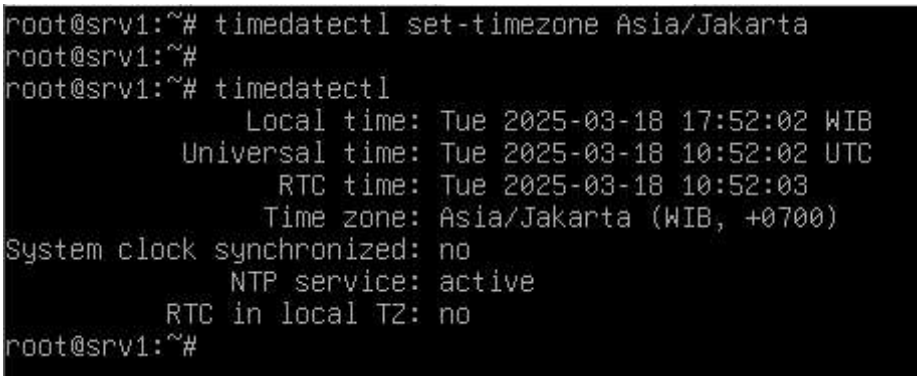
Pada Linux, konfigurasi zone waktu dapat dilakukan dengan menggunakan perintah timedatectl yang memiliki sintaks seperti berikut:

```
timedatectl set-timezone <NAMA_TIMEZONE>
```

Berikut konfigurasi timezone yang diset menjadi `Asia/Jakarta` :

```
timedatectl set-timezone Asia/Jakarta
```

Berikut contoh eksekusinya:



Untuk mengetahui nama zone waktu yang tersedia, gunakan perintah `timedatectl list-timezones`

Basic Command

Berikut adalah beberapa perintah dasar yang umum digunakan pada Linux:

1. Perintah `cd`
   Digunakan untuk pindah direktori, contohnya seperti berikut:

```
root@srv1:~# cd /srv
root@srv1:/srv#
root@srv1:/srv# cd /tmp
root@srv1:/tmp#
```

2. Perintah `ls`
   Digunakan untuk melihat list file yang ada pada direktori, contohnya seperti berikut:

```
root@srv1:/tmp# ls
systemd-private-e72a06a7d30547e3978b5b990c521306-systemd-logind.service-HhOrBY
systemd-private-e72a06a7d30547e3978b5b990c521306-systemd-timesyncd.service-0xt04o
vmware-root_316-868458717
root@srv1:/tmp# _
```

3. Perintah `mkdir`
   Digunakan untuk membuat direktori, contohnya seperti berikut:

```
root@srv1:/tmp# mkdir test
root@srv1:/tmp# cd test
root@srv1:/tmp/test# _
```

4. Perintah `touch`
   Digunakan untuk membuat direktori, contohnya seperti berikut:

```
root@srv1:/tmp/test# touch test.txt
root@srv1:/tmp/test#
```

5. Perintah `cat`
   Digunakan untuk melihat konten dari suatu file, contohnya seperti berikut:

```
root@srv1:/tmp/test# cat test.txt
root@srv1:/tmp/test# _
```

6. Perintah `cp` dan `mv`
   Perintah `cp` digunakan untuk mengcopy file, sedangkan `mv` dapat digunakan untuk memindahkan dan mengganti name suatu file. Kedua perintah ini memiliki argumen sintaks seperti berikut:

   ```
   cp <TARGET_FILE> <COPIED_FILE>
   mv <TARGET_FILE> <NEW_FILE/NEW_NAME_FILE>
   ```

   Berikut contohnya:

   ```
   root@srv1:/tmp/test# cp test.txt test_cp.txt
   root@srv1:/tmp/test#
   root@srv1:/tmp/test# ls
   test_cp.txt  test.txt
   root@srv1:/tmp/test#
   root@srv1:/tmp/test#
   root@srv1:/tmp/test# mv test_cp.txt test_mv.txt
   root@srv1:/tmp/test#
   root@srv1:/tmp/test#
   root@srv1:/tmp/test# ls
   test_mv.txt  test.txt
   root@srv1:/tmp/test#
   ```

7. Perintah `echo`
   Digunakan untuk menulis suatu teks pada terminal. Bisa juga digunakan untuk menulis teks ke dalam suatu file dengan ditambahkan ouput redirection.

   ```
   root@srv1:/tmp/test# echo test
   test
   root@srv1:/tmp/test#
   ```

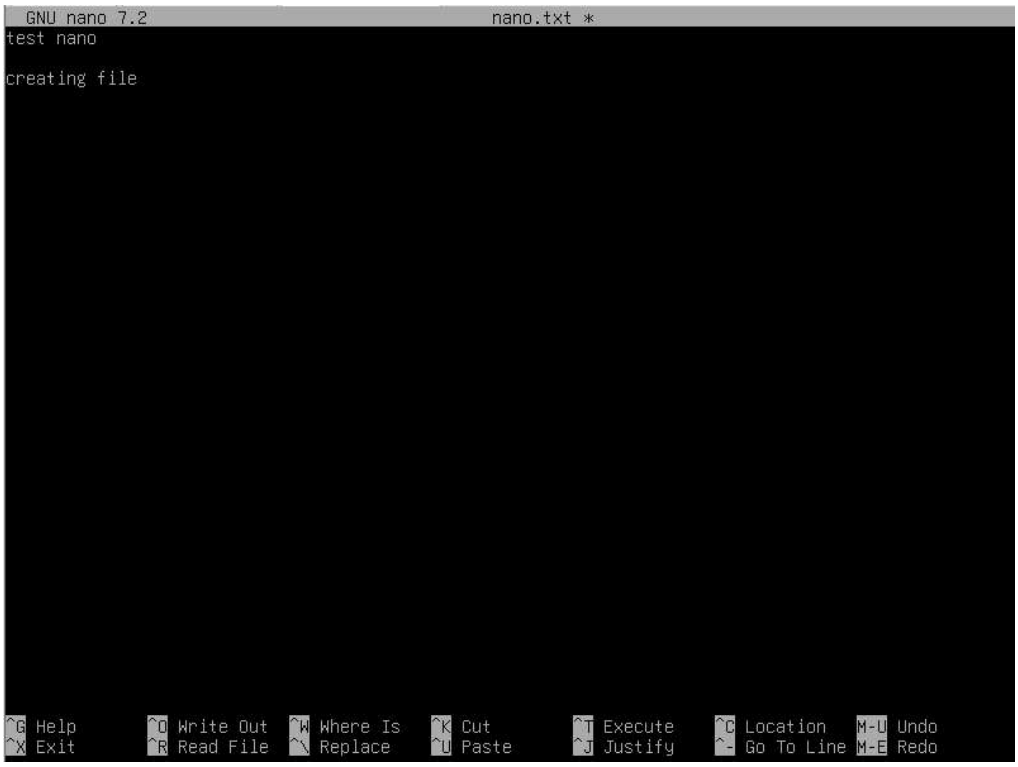
8. Output redirection
   Digunakan untuk mengarahkan atau menulis hasil eksekusi suatu perintah ke dalam suatu file dengan cara menambahkan tanda `>` setelah perintah yang dieksekusi. Contohnya seperti berikut:

   ```
   root@srv1:/tmp/test# echo test > test_output.txt
   root@srv1:/tmp/test#
   root@srv1:/tmp/test# cat test_output.txt
   test
   ```

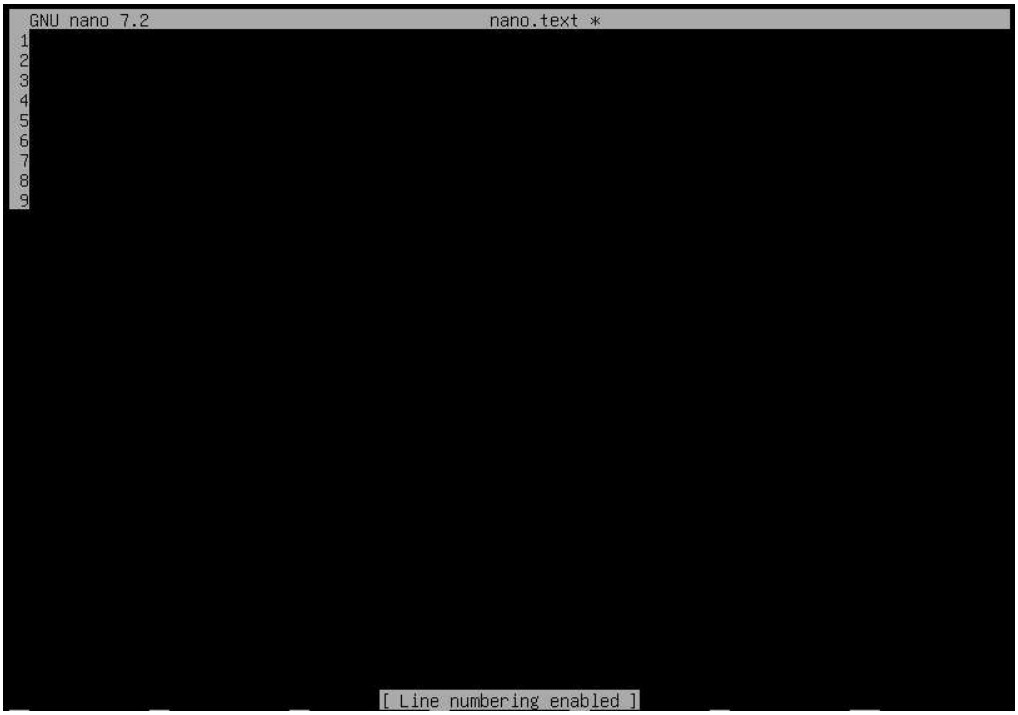
   Untuk menambahkan teks baru ke dalam suatu file yang didalamnya sudah terdapat teks atau konten, gunakan tanda `>>` . Contohnya seperti berikut:

```
root@srv1:/tmp/test# echo output >> test_output.txt
root@srv1:/tmp/test# cat test_output.txt
test
output
root@srv1:/tmp/test#
```

9. Perintah nano
   Merupakan salah satu text editor yang ada pada Linux. Untuk menjalankannya, cukup mengetik nano diikuti dengan nama file. Berikut contoh tampilannya:



   Setelah selesai menggunakan nano, untuk menyimpan file yang diakses tekan **Ctrl + S** dilanjut dengan **Ctrl + X**
   Untuk mengaktifkan line number pada nano, tekan **Alt + N** saat membuka nano



10. Konfigurasi IP address
    Konfigurasi IP address dapat dilakukan pada file `/etc/network/interfaces` contohnya seperti konfigurasi pada srv1:

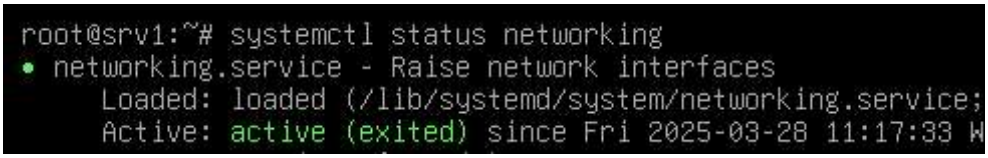
```
auto ens33
iface ens33 inet static
address 192.168.192.10/24


auto ens36
iface ens36 inet dhcp
```

    Sesuaikan nama interface. Jika pada VM nama interface **ens192** , maka ganti menjadi **ens192**
11. Perintah `systemctl`
    Systemctl digunakan untuk mengatur services yang ada pada Linux. Berikut beberapa contoh penggunaannya:
    Restart service menggunakan `systemctl restart` atau bisa juga menggunakan `systemctl reload`

```
root@srv1:~# systemctl restart networking
root@srv1:~#
```

    Untuk mengetahui status dari suatu service, gunakan perintah `systemctl status`

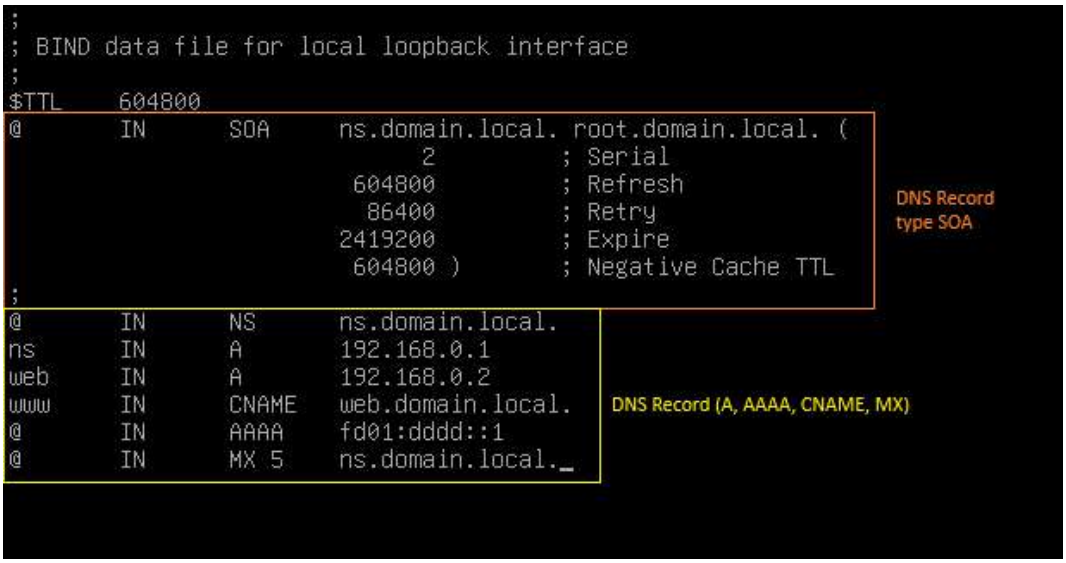

root@srv1:~# systemctl status networking
● networking.service - Raise network interfaces
Loaded: loaded (/lib/systemd/system/networking.service;
Active: active (exited) since Fri 2025-03-28 11:17:33 W

Intro to DNS

DNS adalah protokol yang digunakan untuk menerjemahkan nama domain (seperti www.example.com) menjadi alamat IP address (seperti 192.168.0.1). Hasil DNS yang berupa IP address akan digunakan oleh komputer untuk saling berkomunikasi.

Server yang menyediakan layanan DNS disebut sebagai **DNS server** atau **Nameserver**. Proses yang dilakukan oleh client untuk meminta informasi suatu domain disebut sebagai **DNS query** atau **DNS request**.

Selain IP address, DNS server juga dapat menjawab suatu DNS query dengan bentuk lain seperti alias suatu domain. Hal tersebut terjadi karena DNS memiliki **DNS record type** seperti `A`, `PTR`, `CNAME` dan `MX`. Setiap DNS record type tadi akan ditulis pada suatu file yang bernama **DNS zone**. Dengan kata lain, **DNS zone** merupakan file yang didalamnya terdapat banyak DNS record dari suatu domain. Satu nameserver bisa memuat banyak DNS zone dari domain yang berbeda – beda. Pada dokumentasi ini, layanan DNS akan menggunakan package Bind9. Berikut contoh dari file DNS zone yang digunakan pada aplikasi bind9.



Pada DNS zone memuat beberapa DNS record yang memiliki berbagai macam jenis seperti `SOA`, `A`, `AAAA`, `CNAME` dan `MX`. Setiap jenisnya

memiliki fungsi masing – masing dan perlu diingat **pada suatu DNS zone hanya ada satu SOA DNS record untuk satu DNS zone**.

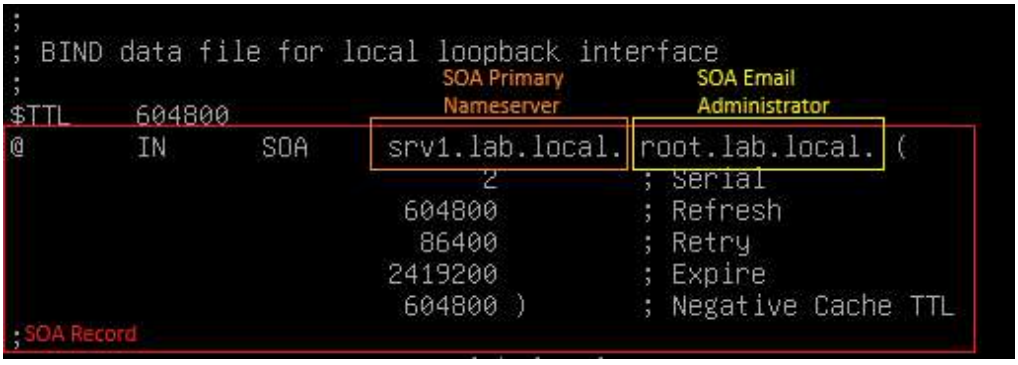
DNS zone hanya dimuat oleh satu nameserver utama (Primary Nameserver) yang dimana domain dari nameserver tersebut akan dimuat pada SOA Primary Nameserver. Untuk membuat DNS zone dapat dimuat pada banyak nameserver, maka proses **DNS Zone Transfer** perlu dilakukan.

DNS Record Type

DNS memiliki beberapa jenis tipe record. Setiap tipe record dimuat pada jenis zone file yang berbeda. Zone file ini umumnya terdiri dari dua jenis, yaitu **forward lookup zone** dan **reverse lookup zone**.

Forward lookup zone memuat DNS record yang meresolve suatu domain ke IP address, sedangkan reverse lookup zone memuat DNS record yang meresolve IP address ke suatu domain. Pada forward lookup dan reverse lookup zone terdapat beberapa jenis record type diantaranya:

1. **SOA Record**
   Start of Authority DNS record berisikan informasi tentang domain seperti Primary Nameserver dan Email Administrator, Serial, Refresh, Retry, Expire dan Negative Cache TTL. Perlu diingat **satu domain hanya memiliki satu primary nameserver** dan nama dari nameserver tersebut tertulis pada **SOA Primary Nameserver**:

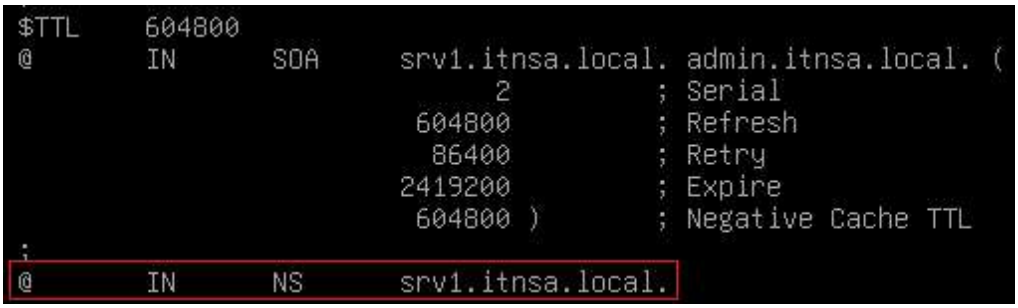


2. **A record**
   Record type yang dimuat pada forward lookup zone. Digunakan untuk membuat mapping dari suatu domain ke host IPv4 address. Contohnya seperti DNS record berikut yang ada pada zone file domain itnsa.local:

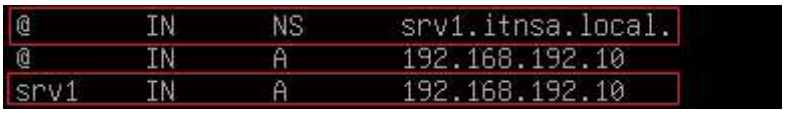
```
@       IN      A       192.168.192.10
srv1    IN      A       192.168.192.10
```

3. **NS record**

Record type yang memuat nama nameserver dari suatu domain. Contohnya seperti berikut:

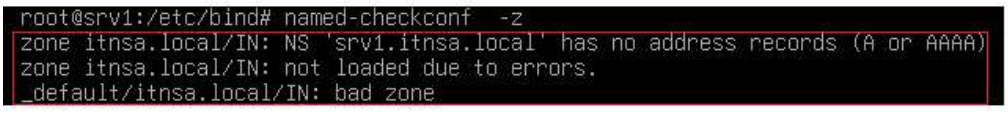


Gambar diatas dapat diartikan nameserver dari domain itnsa.local (sinonimnya **@**) adalah server yang memiliki domain name srv1.itnsa.local. Jangan lupa untuk mendefinisikan domain name dari Primary Nameserver dalam SOA Primary Nameserver. Dalam contoh diatas srv1.itnsa.local merupakan Primary Nameserver dari domain itnsa.local.
Lalu, pastikan sub-domain dari nameserver tersebut sudah didefinisikan pada zone file. Contohnya seperti berikut:



Jika sub-domain dari nameserver belum didefinisikan, maka akan terjadi error pada proses testing konfigurasi menggunakan perintah named-checkconf -z





4. **CNAME record**
   Disebut juga sebagai alias yang digunakan untuk memberikan nama lain dari suatu DNS record. Contohnya seperti berikut:

```
;
@       IN      NS      srv1.itnsa.local.
@       IN      A       192.168.192.10
srv1    IN      A       192.168.192.10
www     IN      CNAME   srv1.itnsa.local.
```

Gambar diatas dapat diartikan www.itnsa.local merupakan nama lain dari domain srv1.itnsa.local

5. **MX record**
   Digunakan untuk memuat domain name suatu server yang digunakan sebagai Mail Server suatu domain. Sebelum membuatnya, pastikan domain name server tersebut memiliki DNS record bertipe A. Contohnya seperti berikut:

```
mail    IN      A       192.168.192.20
@       IN      MX 10   mail.itnsa.local.
```

   Gambar diatas dapat diartikan bahwa server dengan domain name mail.itnsa.local merupakan Mail Server untuk domain itnsa.local ('@' sinonim dari itnsa.local) dengan priority 10. Nilai priority digunakan untuk menentukan prioritas Mail Server yang akan digunakan. Nilai priority pada MX record yang paling rendah akan lebih dipilih dibandingkan MX record yang memiliki priority tinggi.

6. **PTR record**
   Dimuat didalam reverse lookup zone, DNS record type ini digunakan untuk meresolve IP address ke suatu domain. Untuk penulisannya, perlu diperhatikan bagaimana zone tersebut diload. Contohnya seperti berikut:

```
zone "168.192.in-addr.arpa" {
        type master;
        file "/etc/bind/db.192";
};
```

   Gambar diatas dapat diartikan bahwa zone file /etc/bind/db.192 digunakan untuk meresolve IP address yang diawali dengan octet **192.168**. Penulisan zone tersebut harus dilakukan secara terbalik. Jika tidak ditulis secara terbalik, contohnya seperti zone "192.168.in-addr.arpa", maka nameserver akan meresolve IP address yang diawali octet **168.192** bukan octet **192.168**
   Untuk penulisan DNS PTR record pada zone file:

```
;
@       IN      NS      srv1.itnsa.local.
10.192  IN      PTR     srv1.itnsa.local._
```

Pada zone file, hanya perlu mendefinisikan sisa IP address atau host portion dari IP address suatu domain yang dituju. Penulisannya juga harus terbalik. Contoh proses resolve:

```
root@srv1:/etc/bind# nslookup
> server localhost
Default server: localhost
Address: ::1#53
Default server: localhost
Address: 127.0.0.1#53
> 192.168.192.10
10.192.168.192.in-addr.arpa     name = srv1.itnsa.local.
> _
```

DNS Zone Type

Setelah DNS zone file dibuat, langkah selanjutnya adalah proses pemuatan DNS zone suatu domain. Berikut beberapa jenis tipe DNS zone:

1. **Primary / Master Zone**
   Merupakan DNS Zone yang diload oleh suatu nameserver, yang dimana nameserver tersebut berperan sebagai Primary Nameserver suatu domain. Dalam konteks DNS, Primary merupakan sinonim dari Master. Berikut contohnya:

```
zone "itnsa.local" {
        type master;
        file "/etc/bind/db.itnsa.local";
};
```

```
zone "itnsa.local" {
        type primary;
        file "/etc/bind/db.itnsa.local";
};
```

   Kedua penulisan zone diatas memiliki makna yang sama

2. **Secondary / Slave Zone**
   Merupakan DNS Zone yang dikonfigurasi oleh suatu nameserver yang berperan sebagai Secondary/Slave Nameserver suatu domain. Slave Nameserver akan mereplika konten dari domain zone file yang dimiliki oleh Master Nameserver. Akan dijelaskan lebih lanjut pada dokumentasi selanjutnya. Contoh konfigurasinya seperti berikut:

```
zone "lab.local" {
        type secondary;
        masters { 192.168.192.10; };
};
```

```
zone "lab.local" {
        type slave;
        masters { 192.168.192.10; };
};
```

Kedua penulisan zone diatas memiliki makna yang sama. masters digunakan untu mendefinisikan master nameserver dari domain

3. **Forward Zone**
   Merupakan DNS Zone yang digunakan untuk meneruskan request domain yang spesifik ke suatu nameserver. Contohnya seperti berikut:

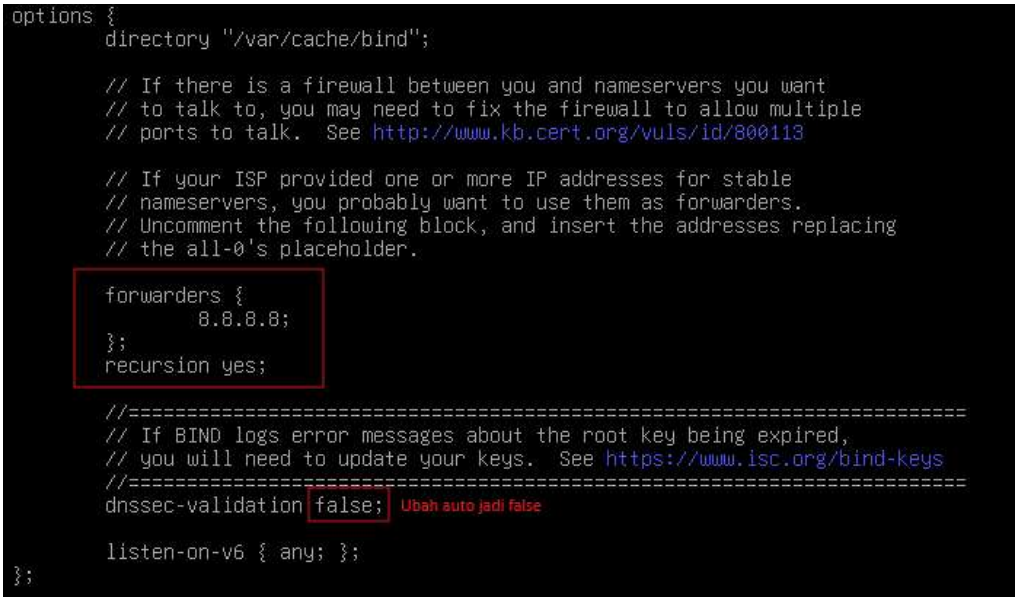
```
zone "lab.local" {
        type forward;
        forwarders { 192.168.192.20; };
};
```

Konfigurasi diatas dapat diartikan bahwa setiap request domain **lab.local** akan diteruskan ke 192.168.192.20.

DNS Server Type

Konfigurasi nameserver secara umum terbagi menjadi dua berdasarkan fungsinya, yaitu dan Recursive Nameserver dan Authoritative Nameserver. Berikut penjelasannya:

1. **Recursive Nameserver**
   Dapat disebut juga sebagai cache nameserver. Tipe nameserver ini tidak berperan sebagai Primary Nameserver atau Secondary Nameserver dari suatu domain tetapi digunakan sebagai *middleman* antara client dengan DNS server yang lain. Pada bind9, untuk membuat Recursive Nameserver lakukan beberapa konfigurasi berikut pada file `/etc/bind/named.conf.options` didalam `options` seperti contoh berikut:



   Selanjutnya proses unload default zone file yang ada pada `/etc/bind/named.conf.default-zones` dengan memberikan komentar pada zone yang diunload menggunakan tanda # seperti berikut:

```
#zone "localhost" {
#       type master;
#       file "/etc/bind/db.local";
#};
#
#zone "127.in-addr.arpa" {
#       type master;
#       file "/etc/bind/db.127";
#};
#
#zone "0.in-addr.arpa" {
#       type master;
#       file "/etc/bind/db.0";
#};
#
#zone "255.in-addr.arpa" {
#       type master;
#       file "/etc/bind/db.255";
#};
```

Untuk memastikan bahwa nameserver tidak meload zone file apapun, ketik perintah named-checkconf -l

```
root@srv1:/etc/bind# named-checkconf  -l
. IN _default hint
root@srv1:/etc/bind# _
```

Reload bind9. Test Recursion Nameserver dengan cara meresolve public domain seperti google.com menggunakan perintah dig @localhost google.com

```
root@srv1:~# dig @localhost google.com

; <<>> DiG 9.18.19-1~deb12u1-Debian <<>> @localhost google.com
; (2 servers found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 17871
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: dd1f8c6a57915118010000006766eb7c1e3353aa53ac20eb (good)
;; QUESTION SECTION:
;google.com.                    IN      A

;; ANSWER SECTION:
google.com.             218     IN      CNAME   forcesafesearch.google.com.
forcesafesearch.google.com. 16278 IN    A       216.239.38.120

;; Query time: 4 msec
;; SERVER: ::1#53(localhost) (UDP)
;; WHEN: Sat Dec 21 17:23:24 CET 2024
;; MSG SIZE  rcvd: 113
```

Pada response diatas, terdapat flags qr (query) rd (recursion desired) ra (recursion available) dan terdapat 2 jawaban dari query google.com. Meskipun nameserver yang dikonfigurasi tidak meload domain zone apapun, tetapi query google.com dapat dijawab oleh nameserver.

Kenapa Recursive Nameserver bisa mengetahui domain google.com padahal nameserver tidak meload zone apapun? Itu terjadi karena konfigurasi forwarders. Secara sederhana, Recursive Nameserver akan meneruskan (forward) query domain apapun ke IP address 8.8.8.8

2. **Authoritative Nameserver**
   Merupakan tipe nameserver yang berperan sebagai Primary Nameserver atau Secondary/Slave Nameserver suatu domain. Suatu nameserver dapat berperan sebagai Authoritative Nameserver dari banyak domain. Salah satu contohnya adalah konfigurasi domain **itnsa.local** berikut. Hal yang harus dilakukan pertama kali adalah membuat zone file terlebih dahulu, contohnya seperti berikut:

```
;
; BIND data file for local loopback interface
;
$TTL    604800
@       IN      SOA     srv1.itnsa.local. admin.itnsa.local. (
                              2         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
;
@       IN      NS      srv1.itnsa.local.
@       IN      A       192.168.192.10
srv1    IN      A       192.168.192.10
www     IN      CNAME   srv1.itnsa.local.
```

Selanjutnya load zone file dari domain itnsa.local. Konfigurasi load zone dapat dilakukan pada file /etc/bind/named.conf.local atau /etc/bind/named.conf.default-zones

```
 GNU nano 7.2                                  named.conf.local
//
// Do any local configuration here
//

// Consider adding the 1918 zones here, if they are not used in your
// organization
//include "/etc/bind/zones.rfc1918";

zone "itnsa.local" {
        type master;
        file "/etc/bind/db.itnsa.local";
};
```

Pastikan zone file sudah berhasil diload dengan menggunakan perintah named-checkconf -l

```
root@srv1:/etc/bind# named-checkconf  -l
itnsa.local IN _default master
. IN _default hint
root@srv1:/etc/bind# _
```

Untuk memastikan bahwa nameserver berperan menjadi Authoritative Nameserver dari suatu domain, lakukan konfigurasi berikut pada file /etc/bind/named.conf.options

```
options {
        directory "/var/cache/bind";

        // If there is a firewall between you and nameservers you want
        // to talk to, you may need to fix the firewall to allow multiple
        // ports to talk.  See http://www.kb.cert.org/vuls/id/800113

        // If your ISP provided one or more IP addresses for stable
        // nameservers, you probably want to use them as forwarders.
        // Uncomment the following block, and insert the addresses replacing
        // the all-0's placeholder.

#       forwarders {
#               8.8.8.8; Forwarder dikomentari
#       };

        recursion no; Menonaktifkan recursion
```

Reload bind9. Untuk proses testing, gunakan perintah dig @localhost srv1.itnsa.local

```
root@srv1:/etc/bind# dig @localhost srv1.itnsa.local

; <<>> DiG 9.18.19-1~deb12u1-Debian <<>> @localhost srv1.itnsa.local
; (2 servers found)
;; global options: +cmd
;; Got answer:
;; WARNING: .local is reserved for Multicast DNS
;; You are currently testing what happens when an mDNS query is leaked to DNS
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 56400
;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: 8cd4401a006d12c0010000006766f1c01d9e2f7db62d8f13 (good)
;; QUESTION SECTION:
;srv1.itnsa.local.              IN      A

;; ANSWER SECTION:
srv1.itnsa.local.       604800  IN      A       192.168.192.10

;; Query time: 0 msec
;; SERVER: ::1#53(localhost) (UDP)
;; WHEN: Sat Dec 21 17:50:08 CET 2024
;; MSG SIZE  rcvd: 89
```

Pada response query diatas, terdapat flag aa (authoritative answer) yang menandakan bahwa response query dari domain srv1.itnsa.local berasal dari authoritative nameserver domain tersebut atau secara sederhana response query berasal dari nameserver yang memiliki dan meload zone file domain itnsa.local.

Pada response query terdapat keterangan `WARNING: recursion requested but not available`, hal tersebut menandakan bahwa nameserver yang dituju yaitu localhost tidak berperan sebagai Recursive Nameserver.

Authoritative Nameserver juga bisa berperan sebagai Recursive Nameserver dengan cara mengaktifkan kembali `forwarders` dan `recursion`

DNS Configuration

Konfigurasi berikut dilakukan pada srv1 yang berperan sebagai master
Proses instalasi package bind9 dapat dilakukan dengan menggunakan perintah berikut:

```
apt install -y bind9
```

Untuk mengkonfigurasi DNS server, hal pertama yang harus dilakukan adalah membuat domain zone file. File ini digunakan untuk memuat DNS record dari suatu domain. Buat file `/etc/bind/db.lab` dan `/etc/bind/db.192` pada srv1 dengan konten seperti berikut:

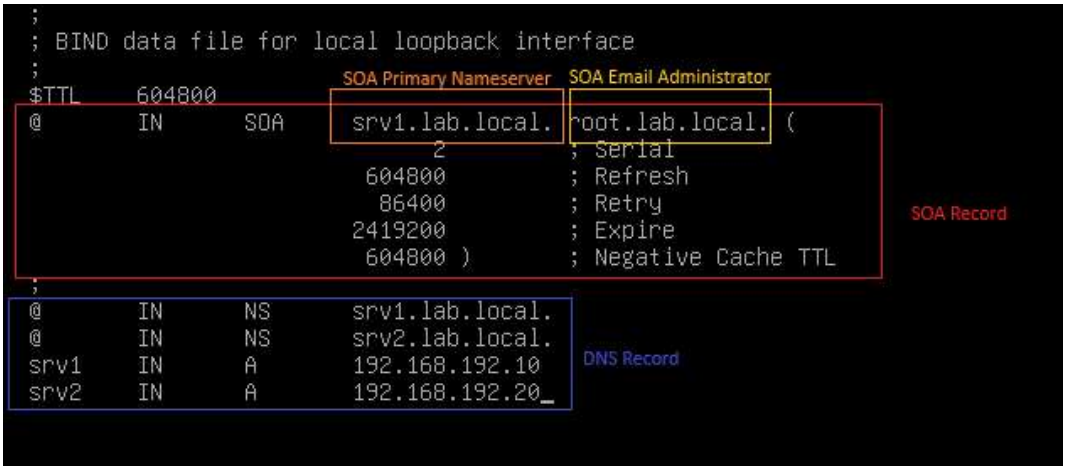
Lakukan konfigurasi berikut untuk mengisi file `/etc/bind/db.lab` dengan DNS record yang sesuai seperti contoh berikut:

```
nano /etc/bind/db.lab
```

```
;
; BIND data file for local loopback interface
;
$TTL    604800
@       IN      SOA     srv1.lab.local. root.lab.local. (
                              1             ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache T
;
@       IN      NS      srv1.lab.local.
@       IN      NS      srv2.lab.local.
```

```
srv1    IN      A       192.168.192.10
srv2    IN      A       192.168.192.20
```

Contoh:



Buat file /etc/bind/db.192 lalu isi file tersebut dengan DNS record yang sesuai seperti contoh berikut:

```
nano /etc/bind/db.192
```

```
;
; BIND reverse data file for local loopback interface
;
$TTL    604800
@       IN      SOA     srv1.lab.local. root.lab.local. (
                              1         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache T
;
@       IN      NS      srv1.lab.local.
@       IN      NS      srv2.lab.local.
10.192  IN      PTR     srv1.lab.local.
20.192  IN      PTR     srv2.lab.local.
```

Contoh:

```
;
; BIND reverse data file for local loopback interface
;
$TTL    604800
@       IN      SOA     srv1.lab.local. root.lab.local. (
                              1         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
;
@       IN      NS      srv1.lab.local.
@       IN      NS      srv2.lab.local.
10.192  IN      PTR     srv1.lab.local.
20.192_ IN      PTR     srv2.lab.local.
```

Setelah zone file dibuat, selanjutnya adalah proses memuat domain zone file dengan cara menulis konfigurasi berikut pada file /etc/bind/named.conf.local

```
nano /etc/bind/named.conf.local
```

```
// Consider adding the 1918 zones here, if they are not use
// organization
// #include "/etc/bind/zones.rfc1918";

zone "lab.local" {
    type master;
    file "/etc/bind/db.lab";
};

zone "168.192.in-addr.arpa" {
    type master;
    file "/etc/bind/db.192";
};
```

```
// Consider adding the 1918 zones here, if they are not used in your
// organization
//include "/etc/bind/zones.rfc1918";

zone "lab.local" {
        type master;
        file "/etc/bind/db.lab";
};

zone "168.192.in-addr.arpa" {
        type master;
        file "/etc/bind/db.192";
};
```

Untuk memastikan bahwa zone file untuk domain lab.local sudah dimuat, ketik perintah named-checkconf -z dan pastikan zone file untuk domain lab.local berstatus **loaded**.

```
root@srv1:/etc/bind# named-checkconf  -z
zone lab.local/IN: loaded serial 2
zone 168.192.in-addr.arpa/IN: loaded serial 1
```

Jika terjadi kesalahan pada konfigurasi zone file, maka zone file tersebut tidak akan diload sehingga perlu diperbaiki terlebih dahulu. Contohnya seperti berikut:

```
root@dns1:/etc/bind# named-checkconf  -z
dns_master_load: /etc/bind/db.lab:16: unexpected end of line
dns_master_load: /etc/bind/db.lab:16: unexpected end of input
/etc/bind/db.lab: file does not end with newline
zone lab.local/IN: loading from master file /etc/bind/db.lab failed: unexpected end of input
zone lab.local/IN: not loaded due to errors.
```

Terakhir reload Bind9 dengan perintah rndc reload dan tes nameserver tersebut menggunakan dig atau nslookup

```
rndc reload
```

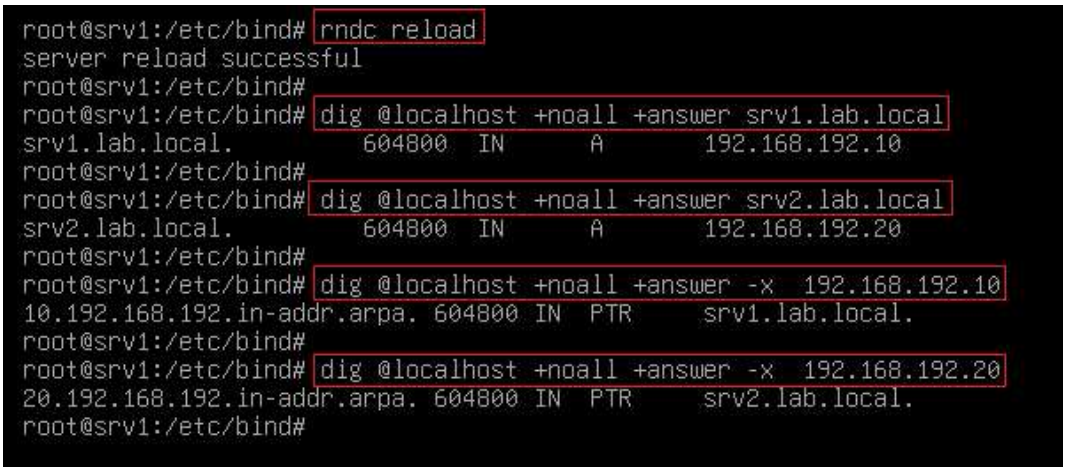
```
dig @localhost +noall +answer srv1.lab.local
```

```
dig @localhost +noall +answer srv2.lab.local
```

```
dig @localhost +noall +answer -x 192.168.192.10
```

```
dig @localhost +noall +answer -x 192.168.192.20
```

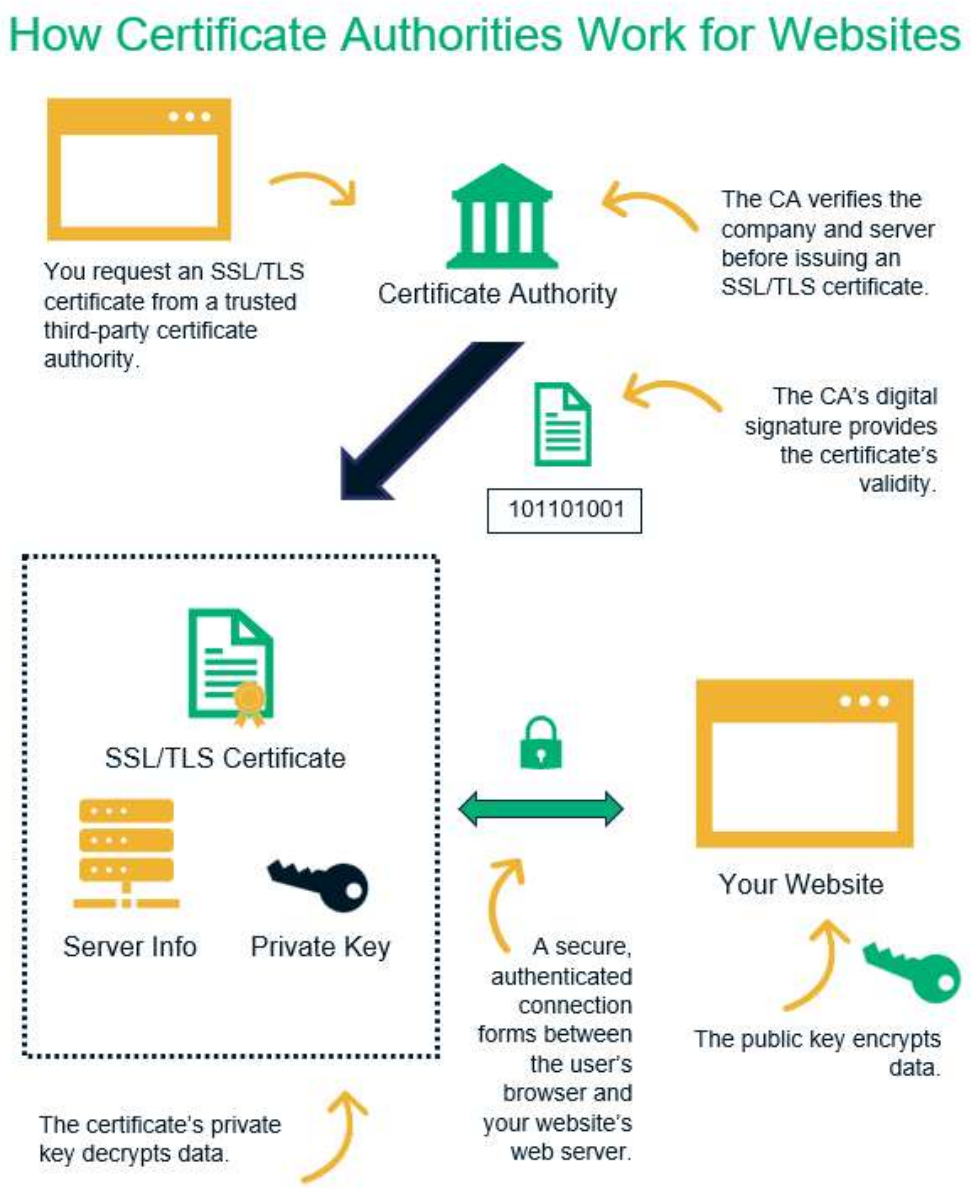
Pastikan outputnya seperti berikut:



Intro to Certificate Authority

Certificate Authority merupakan service yang dikonfigurasi pada server, dimana server tersebut digunakan untuk membuat SSL certificate yang akan di-sign atau di-approve sebagai tanda bahwa SSL certificate tersebut dibuat oleh Certificate Authority. **SSL certificate memerlukan suatu file yang bernama private key**. Kedua file tersebut akan digunakan secara inklusif untuk mengamankan suatu service seperti Web dan Mail services. Pada SSL certificate terdapat suatu atribut yang mengidentifikasi server atau service, atribut tersebut dinamakan ***Common Name***.

Dalam membuat Certificate Authority, diperlukan suatu SSL certificate yang bertipe self-signed, SSL certificate tersebut **dibuat lalu di-sign atau di-approve oleh server itu sendiri**. Sebenarnya *self-signed certificate* sudah dapat digunakan untuk mengamankan suatu service, tetapi saat client mengakses service yang diamankan oleh *self-signed certificate* tersebut, **client tidak akan percaya pada self-signed certificate karena certificate tersebut TIDAK di-sign atau di-approve oleh CA yang dipercayai oleh client**.



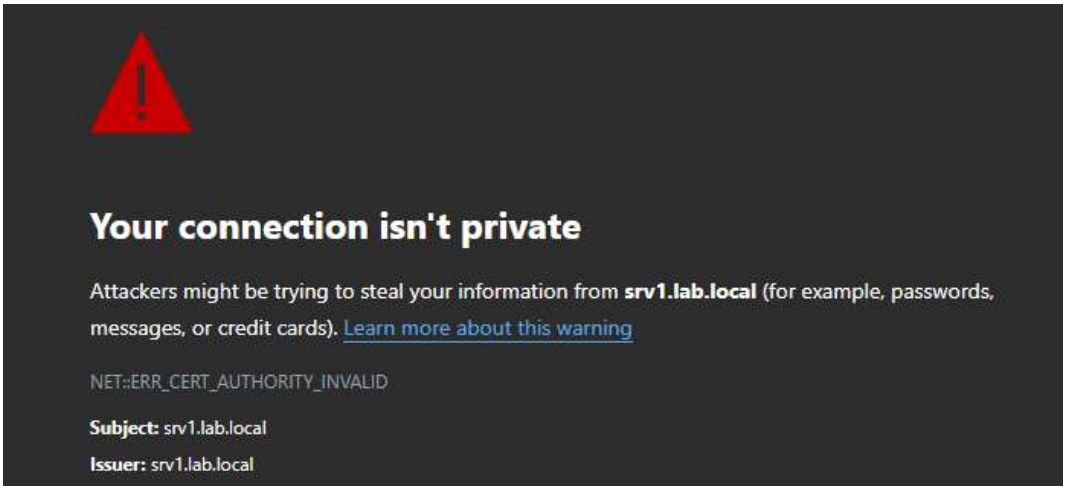
Sumber: https://sectigostore.com/blog/what-is-a-certificate-authority-certification-authorities-explained/

Analogi sederhananya seperti Paspor yang dibuat oleh suatu negara dan Kartu Nama yang dibuat oleh diri sendiri. Setiap Paspor dibuat oleh pihak yang berwenang dan telah diakui contohnya di Indonesia pembuatan paspor adalah tanggung jawab dari bagian Imigrasi. Paspor tersebut dibuat sebagai identitas satu orang dan hanya dibuat oleh Imigrasi tadi.

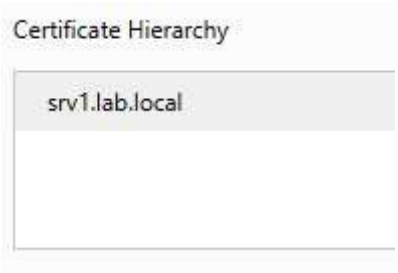
Berdasarkan analogi diatas, dalam konteks CA, **bagian Imigrasi adalah Certificate Authority-nya** dan **Paspor adalah SSL certificate yang digunakan sebagai identitas**. Sedangkan untuk Kartu Nama yang dibuat sendiri dapat dianalogikan sebagai ***self-signed certificate*** dan pada kartu tersebut terdapat identitas dari pembuatnya.

Pada Paspor dan Kartu Nama, bisa saja memiliki beberapa atribut yang mengidentifikasi identitas suatu pihak. Contohnya pada Paspor terdapat Nomor Paspor dan Nama yang digunakan sebagai identitas, lalu pada Kartu Nama terdapat Nama atau Nomor Telepon sebagai identitasnya. Sama halnya seperti Paspor atau Kartu Nama, **certificate dapat memiliki identitas lebih dari satu yang dimana identitas tersebut dimuat pada atribut *Subject Alternative Name***.

Contohnya seperti kasus berikut, saat client mengakses service yang diamankan oleh *self-signed certificate*. Maka, client akan mendapatkan peringatan atau error seperti berikut:



Pada gambar diatas saat client mengakses layanan HTTPS, *self-signed* certificate digunakan karena *Subject* and *Issuer* pada certificate memiliki value yang sama. Jika dilihat secara mendetail bagaimana hirarki self-signed certificate tersebut dibuat, maka hanya akan muncul satu hirarki seperti berikut.



Jika client mengakses service yang diamankan oleh certificate yang valid atau di-sign dan di-approve oleh CA yang dipercaya, maka tidak akan muncul error seperti pada contoh sebelumnya. Misal saat client mengakses https://google.com. Jika melihat informasi hirarki dari certificate yang digunakan, maka akan muncul beberapa hirarki seperti berikut.



Gambar diatas menjelaskan bahwa certificate google tersebut dibuat oleh CA yang bernama **WR2**, lalu WR2 dibuat oleh CA **GTS Root R1**. Hal ini disebut sebagai ***chain of trust***. Certificate yang digunakan memiliki identitas ***.google.com** yang artinya **semua layanan yang memiliki domain google.com dapat menggunakan certificate ini**. Tipe certificate yang identitasnya memiliki tanda bintang disebut sebagai ***wildcard certificate***.

Berikut adalah contoh value dari atribut ***Subject Alternative Name*** yang ada pada certificate:



Pada Linux, list CA yang sudah dipercaya dapat dilihat dengan menggunakan perintah `dpkg-reconfigure ca-certificates` .



Kesimpulannya:

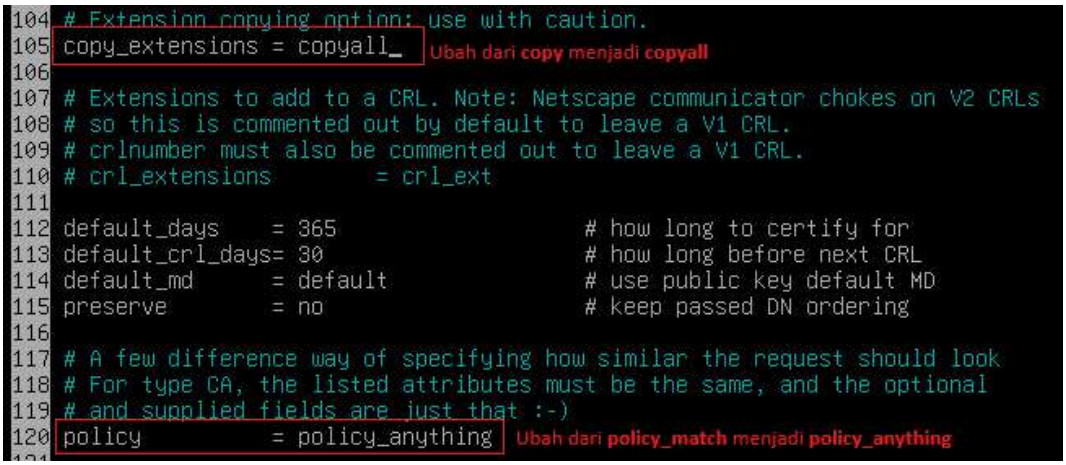
1. Certificate Authority adalah service pada server yang digunakan untuk membuat SSL certificate. CA membutuhkan *self-signed certificate*.

2. SSL certificate mengidentifikasi service suatu server. Pada *certificate* terdapat identitas yang dimuat pada atribut ***Common Name*** dan certificate bisa memiliki lebih dari satu identitas menggunakan ***Subject Alternative Name***.
3. SSL certificate memerlukan private key.
4. Agar tidak ada error saat client mengakses service yang diamankan oleh SSL certificate yang dibuat oleh suatu CA, maka CA certificate yang membuat SSL certificate tersebut harus dipercaya terlebih dahulu oleh client.

SSL Certificate Intro

Lakukan konfigurasi berikut pada srv1

Pada Linux, pembuatan SSL certificate dapat dibuat menggunakan perintah openssl. Sebelum itu, perlu beberapa konfigurasi yang harus dilakukan terkait pembuatan SSL certificate. Langkah pertama adalah melakukan konfigurasi baris `105` dan `120` pada file `/etc/ssl/openssl.cnf`

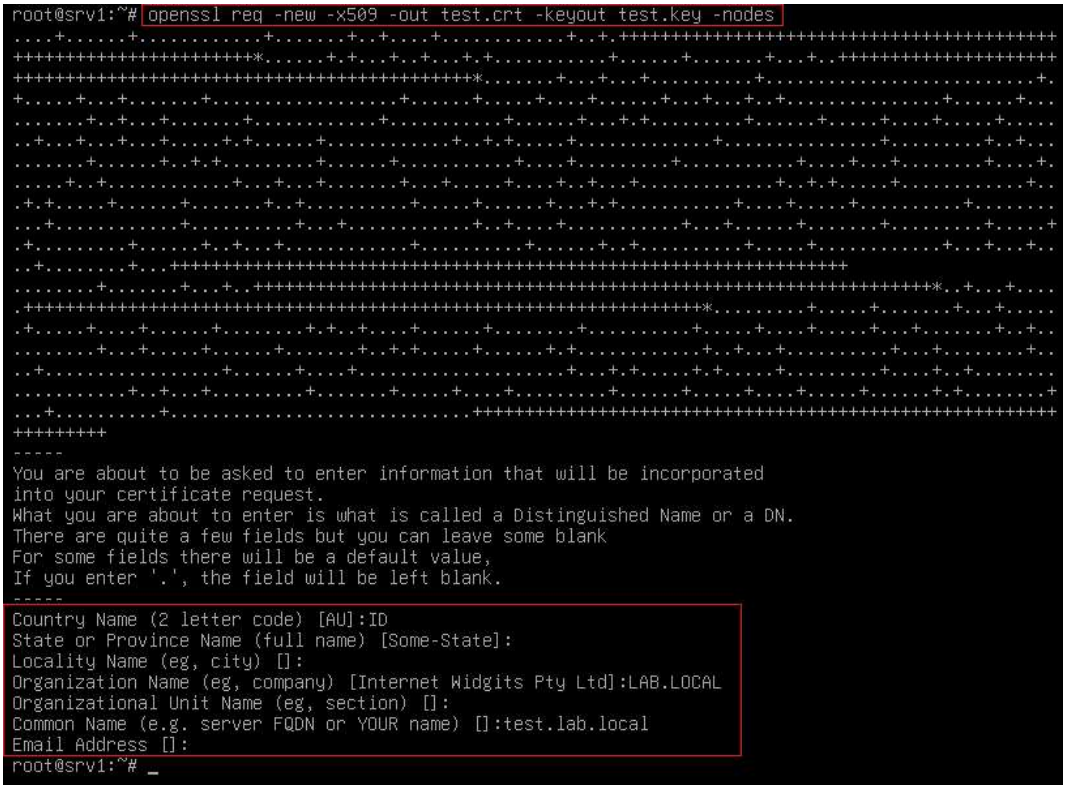


Pembuatan self-signed SSL certificate dapat dibuat dengan menggunakan perintah dengan sintaks seperti berikut:

```
openssl req -new -x509 -out <NAMA_SSL_CERTIFICATE> -keyout
```

Contohnya seperti berikut, membuat self-signed certificate untuk test.lab.local. File certificate akan diberi nama `test.crt` dan private key-nya `test.key` . Perintahnya adalah:

```
openssl req -new -x509 -out test.crt -keyout test.key -node
```

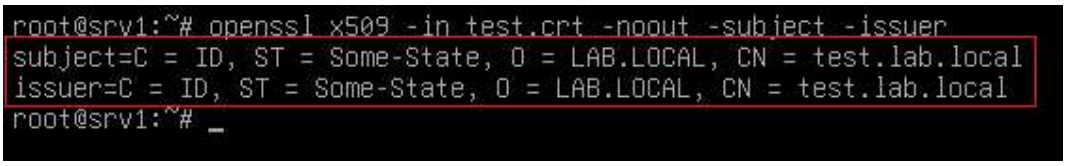


Saat membuat self-signed SSL certificate, user akan diminta beberapa pertanyaan terkait beberapa atribut tambahan pada certificate yang mencakup *Country Name, State, Locality, Organization, Organizational Unit* dan *Email Address* tetapi atribut yang paling terpenting adalah atribut **Common Name** yang didalamnya harus dimasukan identitas dari server atau domain dari service yang akan diamankan.

Untuk mengecek atribut dari SSL certificate yang telah dibuat, gunakan perintah berikut:

```
openssl x509 -in <NAMA_SSL_CERTIFICATE> -noout -subject -i
```

Contohnya seperti berikut. Jika value atribut *subject* dan *issuer* sama, artinya certificate tersebut bertipe self-signed.



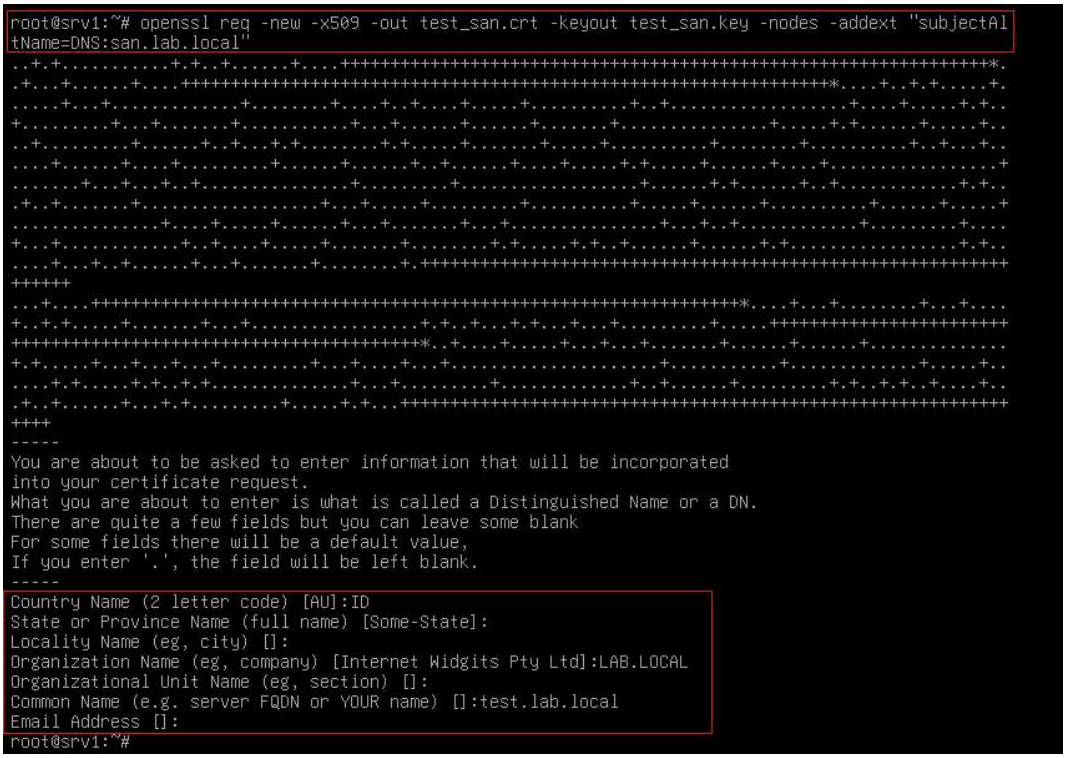
Certificate with Subject Alternative Name

Selanjutnya adalah proses pembuatan SSL certificate yang memiliki identitas lebih dari satu melalui atribute *Subject Alternative Name*. Prosesnya mirip seperti sebelumnya tetapi memiliki opsi tambahan. Berikut format perintahnya:

```
openssl req -new -x509 -out <SSL_CERTIFICATE> -keyout <PRI
```

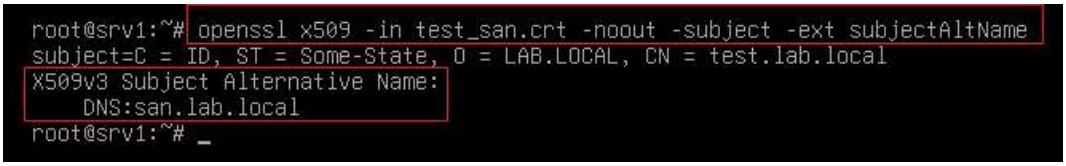
Contohnya perintah yang ada pada gambar dibawah ini:

```
openssl req -new -x509 -out test_san.crt -keyout  test_san
```



Untuk mengecek apakah Subject Alternative Name sudah ada pada certificate yang dibuat, gunakan perintah:

```
openssl x509 -in <NAMA_SSL_CERTIFICATE> -noout -subject -e
```

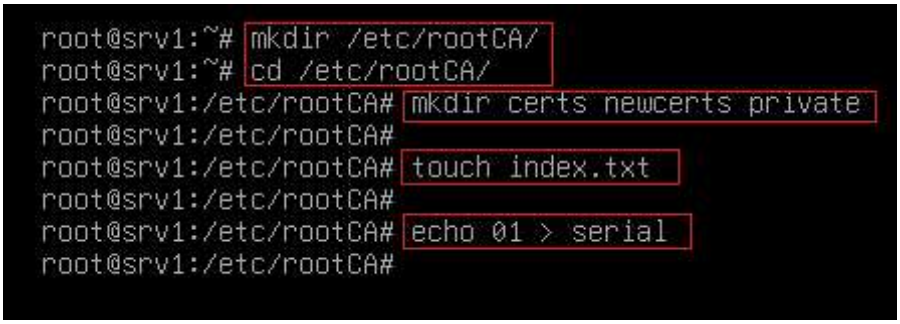

root@srv1:~# openssl x509 -in test_san.crt -noout -subject -ext subjectAltName
subject=C = ID, ST = Some-State, O = LAB.LOCAL, CN = test.lab.local
X509v3 Subject Alternative Name:
DNS:san.lab.local
root@srv1:~#

Certificate Authority Deployment

Lakukan konfigurasi berikut pada srv1
Untuk membuat Certificate Authority, langkah pertama yang harus dilakukan adalah mengkonfigurasi `/etc/ssl/openssl.cnf` untuk menentukan direktori yang akan digunakan oleh CA. Konfigurasi tersebut berada pada baris 82. Contohnya seperti berikut:



Setelah itu buat direktori yang digunakan oleh CA beserta direktori tambahan lainnya seperti direktori **certs**, **newcerts**, dan **private**. Lalu buat file kosong bernama **index.txt**, dilanjut membuat file serial yang didalamnya berisikan angka. Berikut contohnya:

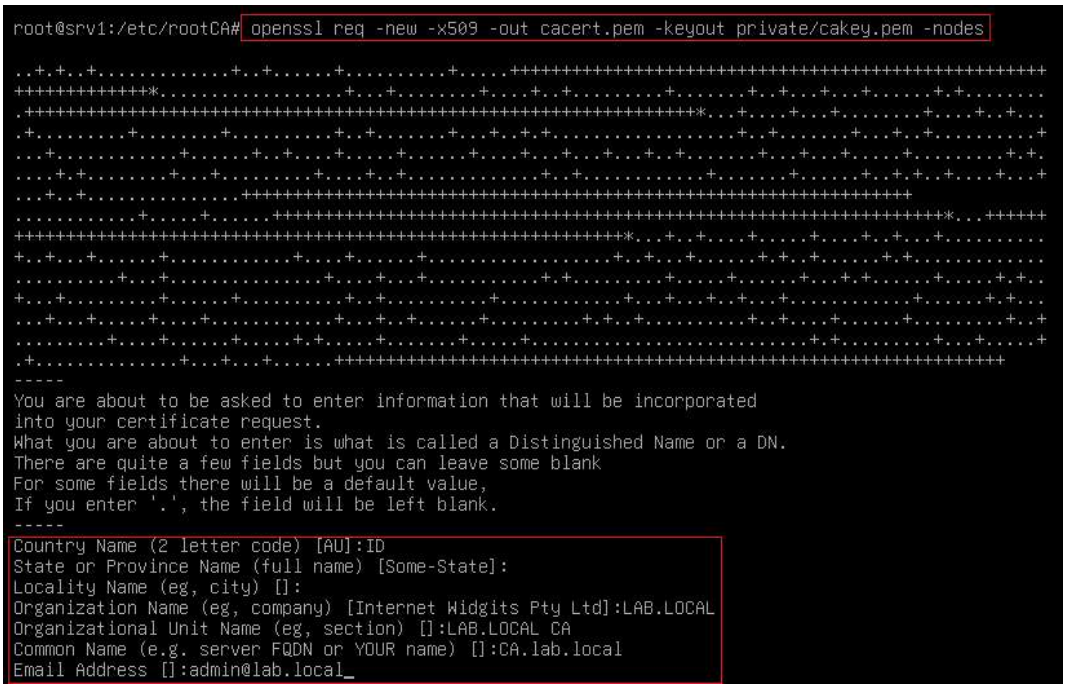


- Direktori **certs**, **newcerts**, **private** wajib dibuat supaya pada saat proses signing certificate, tidak terjadi error.
- File **index.txt** digunakan untuk menyimpan informasi list certificate yang dibuat dan di-sign oleh CA.
- File **serial** digunakan sebagai penanda nomor serial certificate. Setiap certificate yang di-sign oleh CA, nilai dari file serial akan berubah.

Lalu generate self-signed certificate menggunakan perintah yang sudah dijelaskan sebelumnya. Tetapi, pastikan nama file certificate yang

digenerate adalah **cacert.pem** dan private key disimpan pada direktori **private** sebagai **cakey.pem**. Berikut contohnya:

```
openssl req -new -x509 -out cacert.pem -keyout private/cake
```



Untuk mengecek apakah CA sudah siap digunakan, ketik perintah `openssl ca -verbose` Contohnya seperti berikut. Pastikan terdapat output generating index.

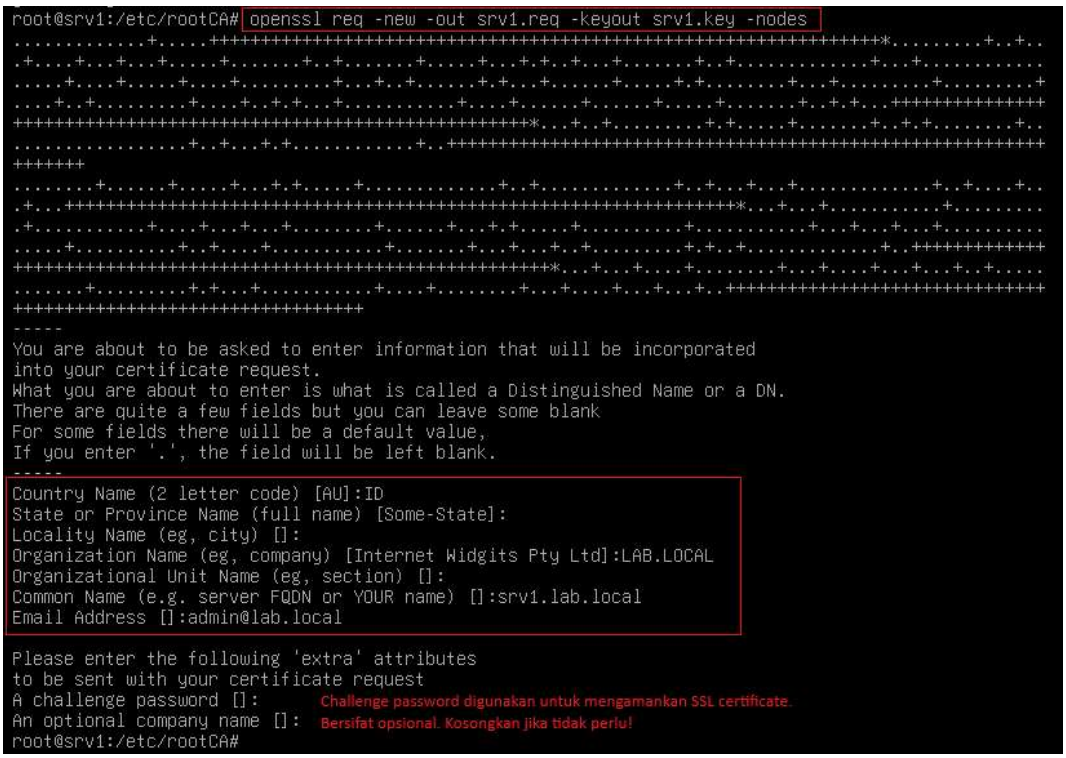
```
root@srv1:/etc/rootCA# openssl ca -verbose
Using configuration from /usr/lib/ssl/openssl.cnf
0 entries loaded from the database
generating index
root@srv1:/etc/rootCA#
```

Untuk proses testing CA, hal pertama yang harus digunakan adalah membuat certificate request beserta private key-nya. Perintahnya hampir sama seperti yang digunakan untuk membuat self-signed certificate, tetapi tidak menggunakan opsi `-x509` . Berikut sintaksnya

```
openssl req -new -out <CERTIFICATE_REQUEST> -keyout <PRIVA
```

Contohnya seperti berikut. Proses pembuatan certificate yang memiliki Common Name **srv1.lab.local**:

```
openssl req -new -out srv1.req -keyout srv1.key -nodes
```

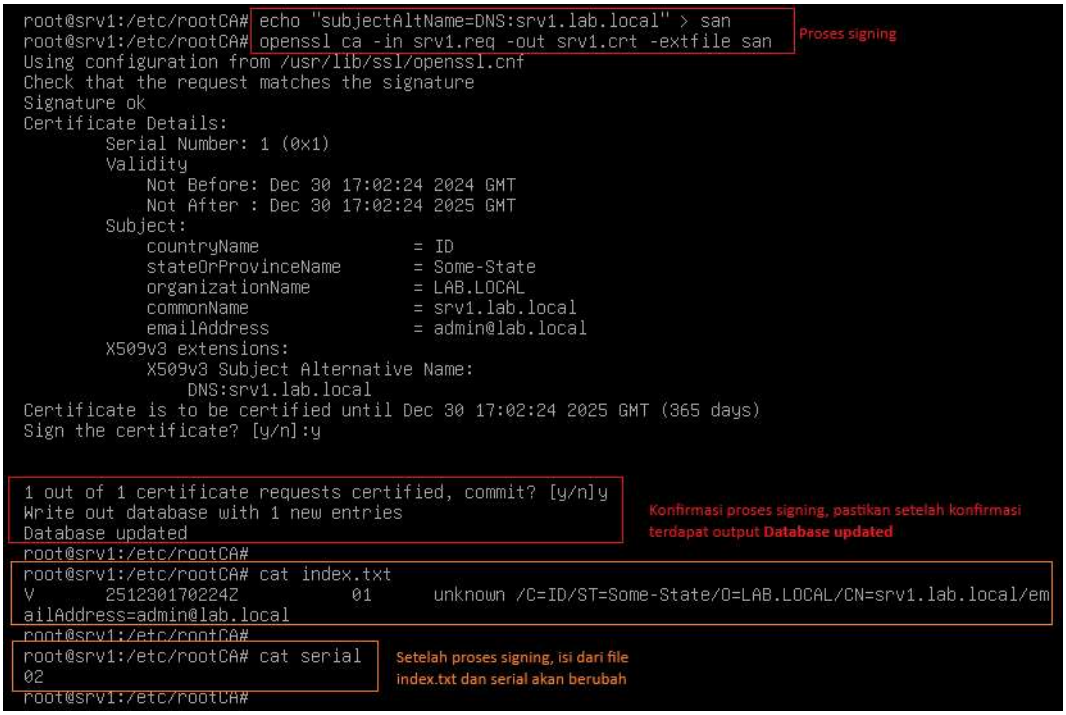


Proses selanjutnya adalah signing certificate request yang telah dibuat. Sebelum itu, pastikan **Subject Alternative Name** dimasukan ke dalam certificate yang akan di-sign oleh CA. Gunakan sintaks berikut untuk proses signing certificate beserta penambahan atribut Subject Alternative Name:

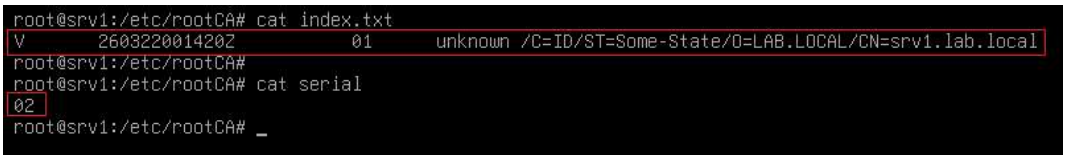
```
echo "subjectAltName=DNS:<DNS_NAME>" > san
openssl ca -in <CERTIFICATE_REQUEST> -out <CERTIFICATE_NAMI
```

Berikut contoh proses signing certificate request **srv1.lab.local**.

```
echo "subjectAltName=DNS:srv1.lab.local" > san
openssl ca -in srv1.req -out srv1.crt -extfile san
```

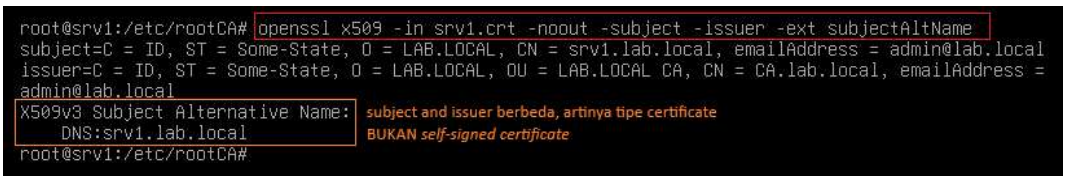


Sesudah melakukan proses signing, lihat isi dari file **index.txt** dan **serial**



Verifikasi certificate yang telah di-sign memiliki Subject Alternative Name menggunakan perintah berikut:

```
openssl x509 -in srv1.crt -noout -subject -issuer -ext subj
```



Trusting Certificate Authority

Agar client menganggap bahwa certificate yang dibuat dan di-sign oleh CA sebelumnya merupakan certificate yang valid, maka pada host client harus dilakukan proses trusting certificate authority. Pada Linux, prosesnya tergolong simple, hanya perlu mengcopy file self-signed certificate dari suatu CA ke dalam direktori `/usr/local/share/ca-certificates` lalu mengetik perintah `update-ca-certificates`. Tetapi, certificate tersebut harus berekstensi `.crt`. Berikut contohnya:



Previous Step

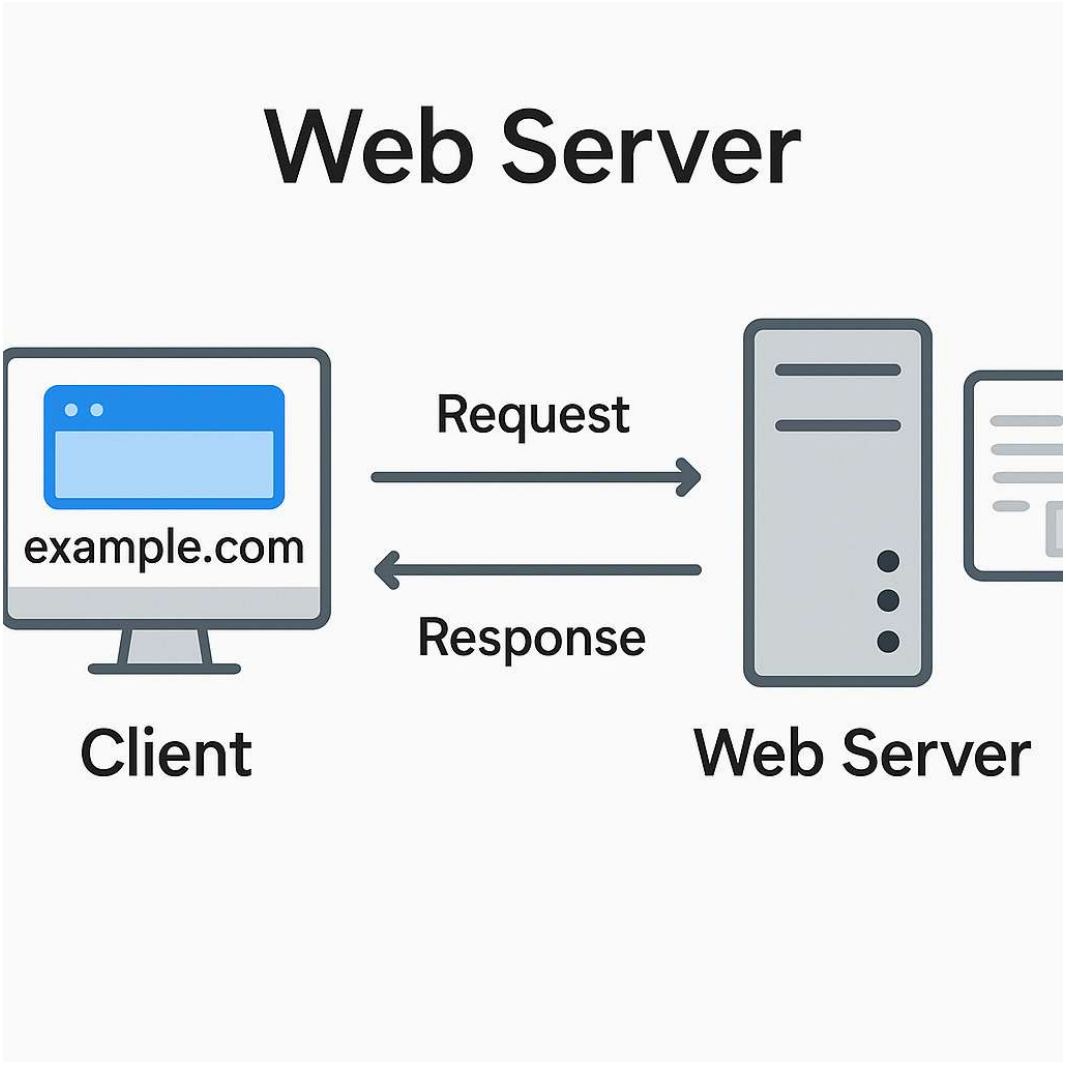
Next Step

Intro to Web Services

Web services adalah perangkat lunak (dan kadang juga perangkat keras) yang berfungsi untuk menerima permintaan (request) dari klien—biasanya melalui browser web—dan mengirimkan kembali respon berupa halaman web, gambar, file, atau data lainnya melalui protokol HTTP (80) atau HTTPS (443).
Server yang menjalankan layanan Web Services disebut sebagai **Web Server**. Secara default web services berjalan pada port TCP/80 (insecure) dan TCP/443 (secure). Perangkah lunak yang biasa digunakan adalah **Nginx** dan **Apache**.

Cara Kerja:

1. Pengguna mengetik alamat web di browser (misalnya example.com).
2. Browser mengirim permintaan HTTP ke web server yang meng-host situs tersebut.
3. Web server memproses permintaan tersebut.
4. Web server mengirim balik data (HTML, CSS, gambar, dll) ke browser.
5. Browser menampilkan hasilnya ke pengguna.


Web Server
example.com
Request
Response
Client
Web Server

Konfigurasi Web Server di Debian 12

Nginx

Sebelum memulai konfigurasi virtual host nginx, konfigurasi dns terlebih dahulu untuk menambahkan A record di file zona (misalnya: `/etc/bind/db.lab`):

```
vhost    IN    A    192.168.192.20
```

Setelah menambahkan record tersebut, pastikan untuk me-restart BIND9:

```
systemctl restart bind9
```

Pastikan juga domain `vhost.lab.local` dapat di-resolve dengan perintah berikut:

```
dig @localhost vhost.lab.local
```

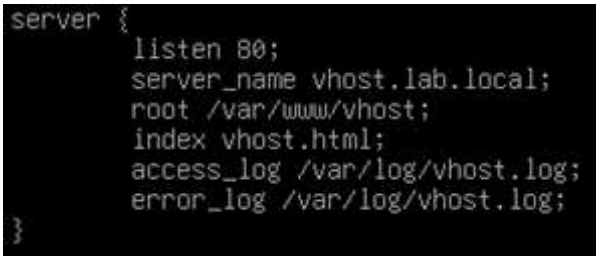
Install nginx

```
apt install nginx -y
```

File Konfigurasi Virtual Host

Lokasi file konfigurasi virtual host: `/etc/nginx/sites-available/vhost`

```
nano /etc/nginx/sites-available/vhost
```




```
server {
    listen 80;
    server_name vhost.lab.local;
    root /var/www/vhost;
    index vhost.html;
    access_log /var/log/vhost.log;
    error_log /var/log/vhost.log;
}
```

Penjelasan Direktif

- **listen**: Menentukan port tempat virtual host dijalankan.
- **server_name**: Menentukan domain name dari virtual host.
- **root**: Menentukan direktori root virtual host.
- **index**: Menentukan nama file index default.
- **access_log**: Menentukan lokasi penyimpanan log akses klien.
- **error_log**: Menentukan lokasi penyimpanan log error. Berguna untuk proses debugging.

Aktivasi Virtual Host

Untuk mengaktifkan konfigurasi virtual host, buat symbolic link ke direktori `sites-enabled`:




```
ln -s /etc/nginx/sites-available/vhost /etc/nginx/sites-ena
```

Membuat Root Directory dan File Index

Buat direktori root dan file index default:




```
mkdir /var/www/vhost
echo "Nginx Virtual Host" > /var/www/vhost/vhost.html
```

Mengecek Konfigurasi dan Reload Nginx

Lakukan pengecekan konfigurasi dan reload layanan Nginx:




```
nginx -t
systemctl reload nginx
```

Pengecekan Website

Setelah semua konfigurasi selesai dan layanan nginx telah di-reload

```
curl http://vhost.lab.local
```

Jika konfigurasi berhasil, maka akan tampil halaman dengan isi "Nginx Virtual Host".

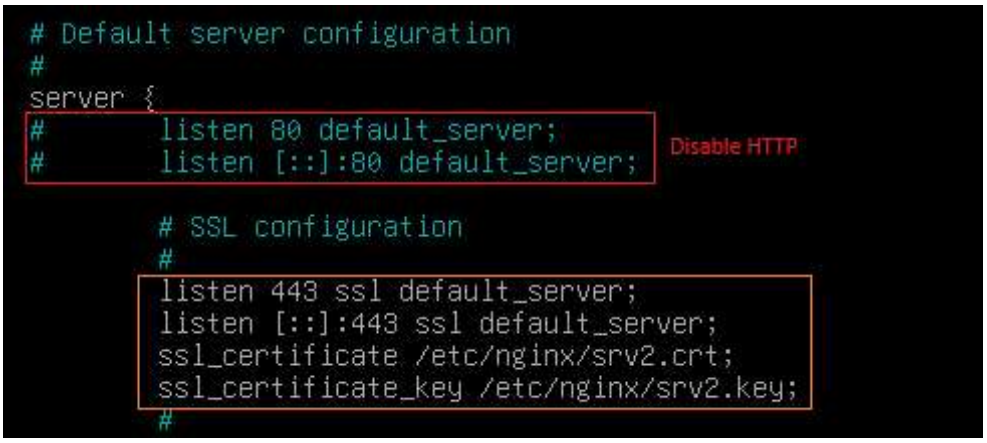
Fitur Web Server

1. Pengamanan dengan SSL Certificate

Untuk mengamankan website menggunakan protokol HTTPS, diperlukan konfigurasi SSL certificate.

Nginx

1. **Pindahkan file sertifikat**
   Pindahkan file `srv2.crt` dan `srv2.key` dari direktori `/srv` ke `/etc/nginx` .
2. **Tambahkan konfigurasi SSL pada virtual host**




```
# Default server configuration
#
server {
#       listen 80 default_server;
#       listen [::]:80 default_server;

        # SSL configuration
        #
        listen 443 ssl default_server;
        listen [::]:443 ssl default_server;
        ssl_certificate /etc/nginx/srv2.crt;
        ssl_certificate_key /etc/nginx/srv2.key;
        #
```

```
nano /etc/nginx/sites-available/vhost
```

```
listen 443 ssl default_server;
listen [::]:443 ssl default_server;
ssl_certificate /etc/nginx/srv2.crt;
ssl_certificate_key /etc/nginx/srv2.key;
```

3. **Cek konfigurasi dan reload nginx**




```
nginx -t
systemctl reload nginx
```

4. **Tes koneksi HTTPS**



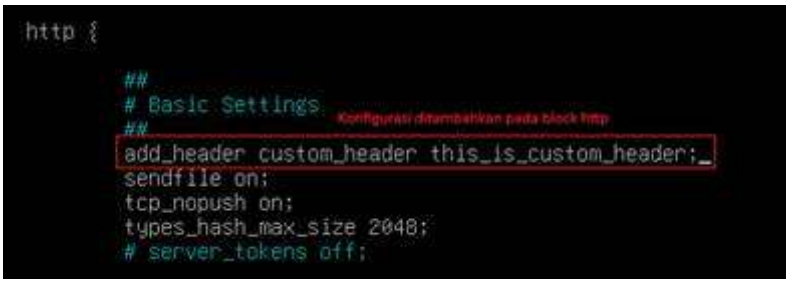

```
curl https://vhost.lab.local
```

2. HTTP Header

HTTP header digunakan untuk mengirim informasi tambahan dalam HTTP request/response. Contoh: header `Host` saat melakukan testing dengan `curl`.

Nginx

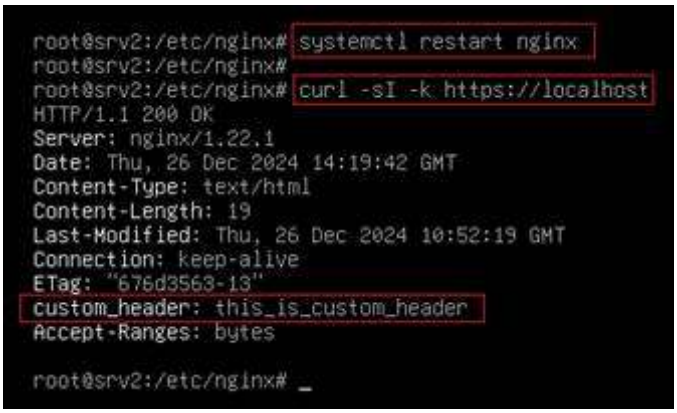
1. **Menambahkan custom header secara global**
   Tambahkan konfigurasi di `/etc/nginx/nginx.conf`:




```
nano /etc/nginx/nginx.conf
```

```
add_header custom_header this_is_custom_header;
```

2. **Restart nginx dan lakukan pengujian**




```
systemctl restart nginx
curl -SI -k https://vhost.lab.local
```

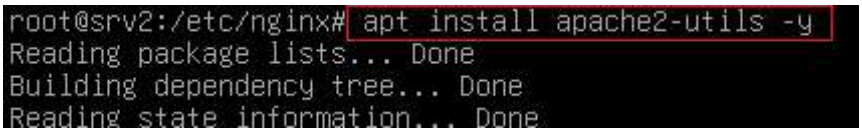
---

3. Basic Authentication

Apache2 dan Nginx mendukung autentikasi dasar, seperti autentikasi menggunakan file `.htpasswd` untuk membatasi akses ke website atau sub-folder tertentu.

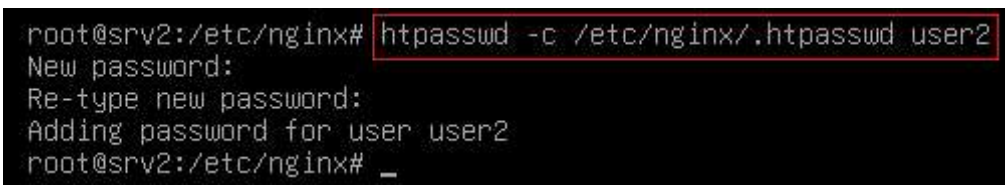
Nginx

1. **Instalasi paket `apache2-utils`**
   Agar bisa menggunakan perintah `htpasswd`.



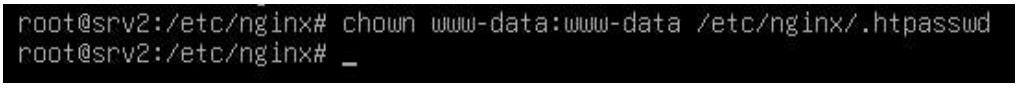

```
apt install apache2-utils -y
```

2. **Membuat file user-password**



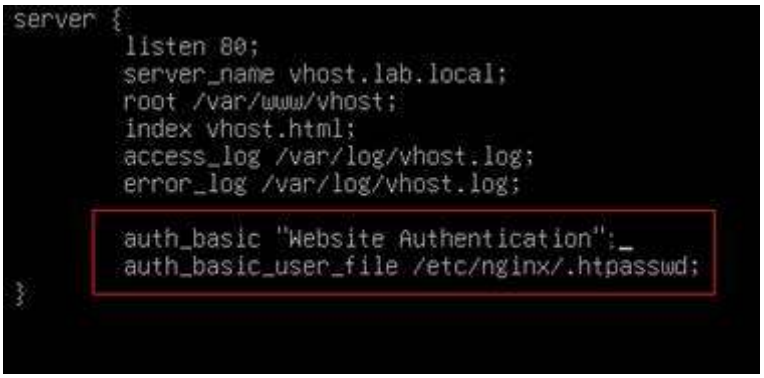
```
htpasswd -c /etc/nginx/.htpasswd user2
```

Pastikan file dimiliki oleh user dan grup www-data :




```
chown www-data:www-data /etc/nginx/.htpasswd
```

3. **Menambahkan konfigurasi autentikasi di virtual host**




```
server {
    listen 80;
    server_name vhost.lab.local;
    root /var/www/vhost;
    index vhost.html;
    access_log /var/log/vhost.log;
    error_log /var/log/vhost.log;

    auth_basic "Website Authentication";
    auth_basic_user_file /etc/nginx/.htpasswd;
}
```

- auth_basic : Memberikan nama autentikasi.
- auth_basic_user_file : Menentukan file yang digunakan untuk autentikasi.

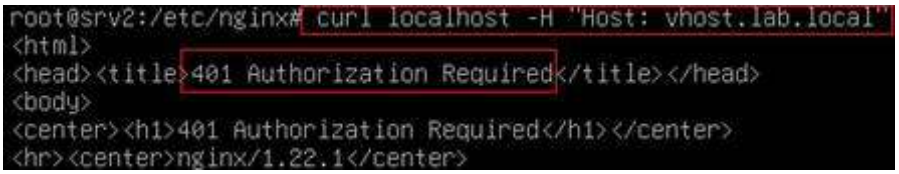
4. **Cek konfigurasi dan reload nginx**




```
nginx -t
systemctl reload nginx
```

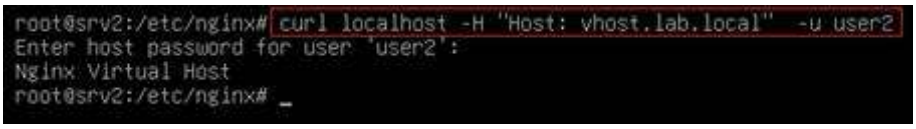
5. **Pengujian**

- Tes tanpa autentikasi (akan gagal dan menampilkan HTTP code 401):




```
curl localhost -H "Host: vhost.lab.local"
```

- Tes dengan autentikasi (akan berhasil):



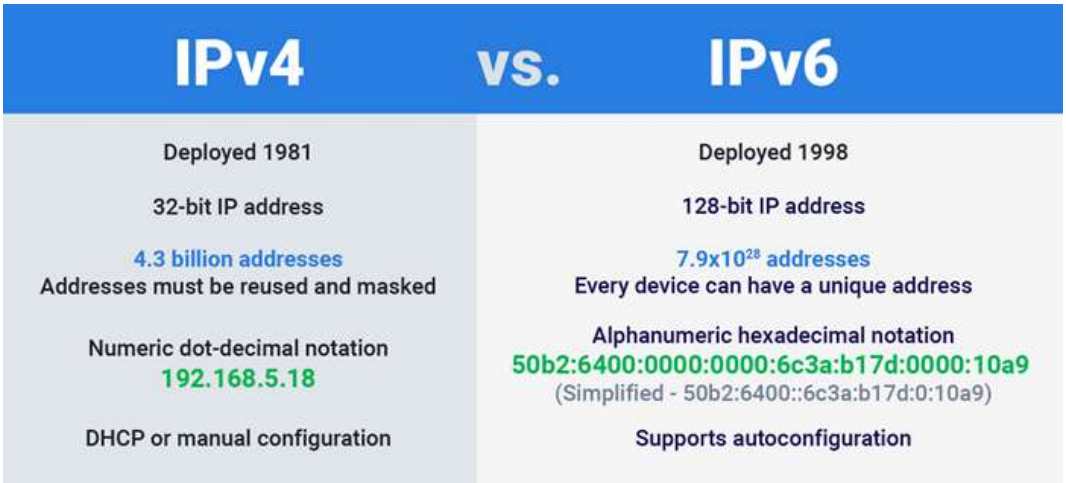

```
curl localhost -H "Host: vhost.lab.local" -u user2
```

Intro to IP Protocol

Pada dokumentasi basic configuration, sudah dijelaskan cara konfigurasi IPv4 address pada sistem operasi Linux Debian. Pada dokumentasi ini, akan dijelaskan lebih lanjut terkait IP protocol.

IP protocol merupakan protocol yang didefinisikan pada layer Network di dalam TCP/IP protocol suite. IP protocol digunakan untuk fungsi addressing dan routing. IP protocol ini merupakan fondasi dari Internet. IP protocol terdiri dari dua jenis yaitu IP protocol version 4 dan IP protocol version 6.

Secara sederhana perbedaan dari kedua jenis protokol tersebut adalah jumlah bit yang dipakai sehingga mempengaruhi jumlah alamat IP yang dapat digunakan. IPv6 memiliki jumlah bit yang lebih banyak sehingga alamatnya lebih banyak dibandingkan IPv4. Berikut adalah perbandingan antara IPv4 dan IPv6.



Sumber: https://www.avg.com/en/signal/ipv4-vs-ipv6

IP Address Configuration

Konfigurasi IP address pada Linux dapat dilakukan pada file `/etc/network/interfaces` seperti yang sudah dilakukan pada bagian basic configuration. Contohnya seperti berikut untuk konfigurasi static IPv4 address:

```
auto ens33
iface ens33 inet static
address 192.168.192.10/24
```

Untuk konfigurasi IPv4 address secara dinamis, ganti inet static menjadi inet dhcp contohnya seperti berikut:

```
auto ens36
iface ens36 inet dhcp
```

Untuk memastikan IPv4 address sudah dikonfigurasi, gunakan perintah `ip -4 addr`

```
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    altname enp2s1
    inet 192.168.192.10/24 brd 192.168.192.255 scope global ens33
       valid_lft forever preferred_lft forever
3: ens36: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    altname enp2s4
    inet 192.168.252.143/24 brd 192.168.252.255 scope global dynamic ens36
       valid_lft 1778sec preferred_lft 1778sec
```

Konfigurasi static IPv6 address dilakukan dengan cara yang sama, namun perbedaannya hanya ada pada sintaksnya seperti berikut:

```
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    altname enp2s1
    inet 192.168.192.10/24 brd 192.168.192.255 scope global ens33
       valid_lft forever preferred_lft forever
3: ens36: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    altname enp2s4
    inet 192.168.252.143/24 brd 192.168.252.255 scope global dynamic ens36
       valid_lft 1778sec preferred_lft 1778sec
```

Untuk memastikan IPv6 address sudah dikonfigurasi, gunakan perintah `ip -6 address`

```
auto ens33
iface ens33 inet6 static
address fd01::1/64
```

Basic DHCP Configuration

Pada dokumentasi ini, akan digunakan package `isc-dhcp-server` untuk DHCP service-nya. DHCP akan diinstall pada **srv1.lab.local**. Client **workstation.domain.local** akan digunakan sebagai service testing, untuk itu nyalakan terlebih dahulu VM tersebut. Konfigurasi DHCP yang akan dilakukan adalah client akan diberikan range IP address antara **192.168.192.100** hingga **192.168.192.200**.

1. Install package `isc-dhcp-server` menggunakan perintah `apt install isc-dhcp-server -y` . Pada saat proses instalasi, akan muncul error seperti berikut:



   Hal tersebut wajar terjadi karena `isc-dhcp-server` belum dikonfigurasi.

2. Tentukan interfaces mana yang satu network / menghadap ke DHCP client. Pada contoh yang dilakukan, interface ens33 merupakan interface yang menghadap ke client karena memiliki IP address yang termasuk bagian dari network 192.168.192.0/24.



   Pada contoh diatas, **ens33** merupakan interface yang satu network dengan client

3. Definisikan interface `ens33` pada `/etc/default/isc-dhcp-server` . Jika nama interface ens192, maka ganti ens33 menjadi `ens192`

```
# On what interfaces should the DHCP server (dhcpd) serve DHCP requests?
#       Separate multiple interfaces with spaces, e.g. "eth0 eth1".
INTERFACESv4="ens33"
INTERFACESv6=""
```

4. Konfigurasi subnet IP address serta range IP address yang akan diberikan pada client. Konfigurasi dilakukan pada file /etc/dhcp/dhcpd.conf . Berikut konfigurasinya:

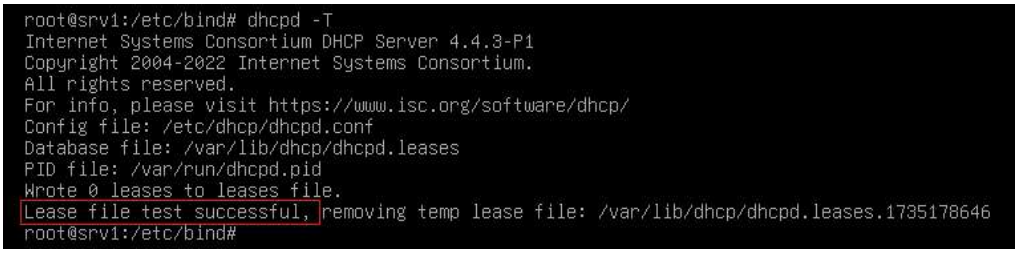
```
subnet 192.168.192.0 netmask 255.255.255.0 {
   range 192.168.192.100 192.168.192.200;
   option domain-name-servers 192.168.192.1, 192.168.192.10;
   option domain-name "domain.local";
   option routers 192.168.192.254;
   option broadcast-address 192.168.192.255;
   default-lease-time 600;
}
```


```
subnet 192.168.192.0 netmask 255.255.255.0 {
   range 192.168.192.100 192.168.192.200;
   option domain-name-servers 192.168.192.1, 192.168.192
   option domain-name "domain.local";
   option routers 192.168.192.254;
   option broadcast-address 192.168.192.255;
   default-lease-time 600;
}
```


Pada konfigurasi diatas, terdapat block subnet yang mendefinisikan network/subnet dimana isc-dhcp-server akan berjalan (listen). Lalu didalamnya terdapat beberapa konfigurasi seperti berikut:

- **range** digunakan untuk mendefinisikan range IP address untuk client
- **option domain-name-servers** digunakan untuk mendefinisikan DNS server yang akan digunakan oleh client
- **option domain-name** digunakan untuk mendefinisikan default domain name yang akan digunakan oleh client
- **option routers** digunakan untuk mendefinisikan default gateway untuk client
- **option broadcast-address** digunakan untuk mendefinisikan broadcast address pada subnet
- **default-lease-time** mendefinisikan waktu peminjaman IP address (lease time)

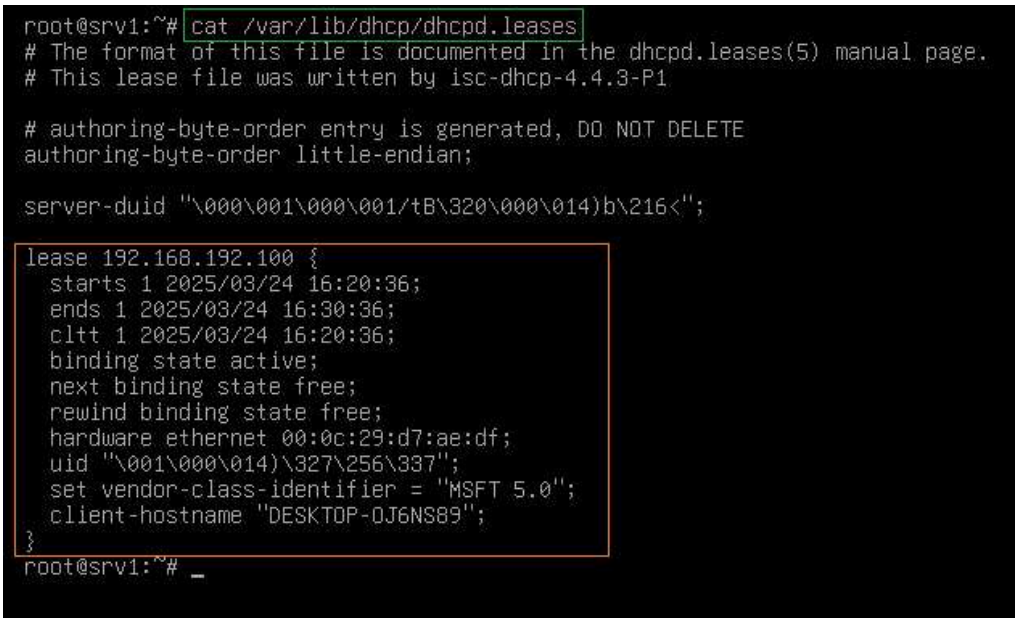
5. Setelah konfigurasi dilakukan, pastikan tidak terdapat error atau kesalahan konfigurasi dengan menggunakan perintah `dhcpd -T`



6. Restart isc-dhcp-server dengan menggunakan perintah:

```
systemctl restart isc-dhcp-server
```

7. Pada DHCP server, ketik perintah `cat /var/lib/dhcp/dhcpd.leases` untuk melihat DHCP lease secara lengkap pada DHCP server. Contohnya seperti berikut:



DHCP Static Lease

Static Lease artinya DHCP client akan selalu mendapatkan IP address yang sama. Static Lease akan menggunakan MAC address sebagai unique identifier DHCP client. Untuk mendapatkan MAC address dari client, lihat file `/var/lib/dhcp/dhcpd.leases`. Langkah konfigurasinya dilakukan pada `/etc/dhcp/dhcpd.conf`.

Pada dokumentasi ini, workstation akan dikonfigurasi sebagai Static Lease client dan akan selalu mendapatkan IP address 192.168.192.168/24. Tambahkan konfigurasi berikut pada file `/etc/dhcp/dhcpd.conf`

Tambahkan konfigurasi berikut pada file `/etc/dhcp/dhcpd.conf`:

```
host workstation.domain.local {
  hardware ethernet 00:12:34:56:78:90;
  fixed-address 192.168.192.168;
}
```

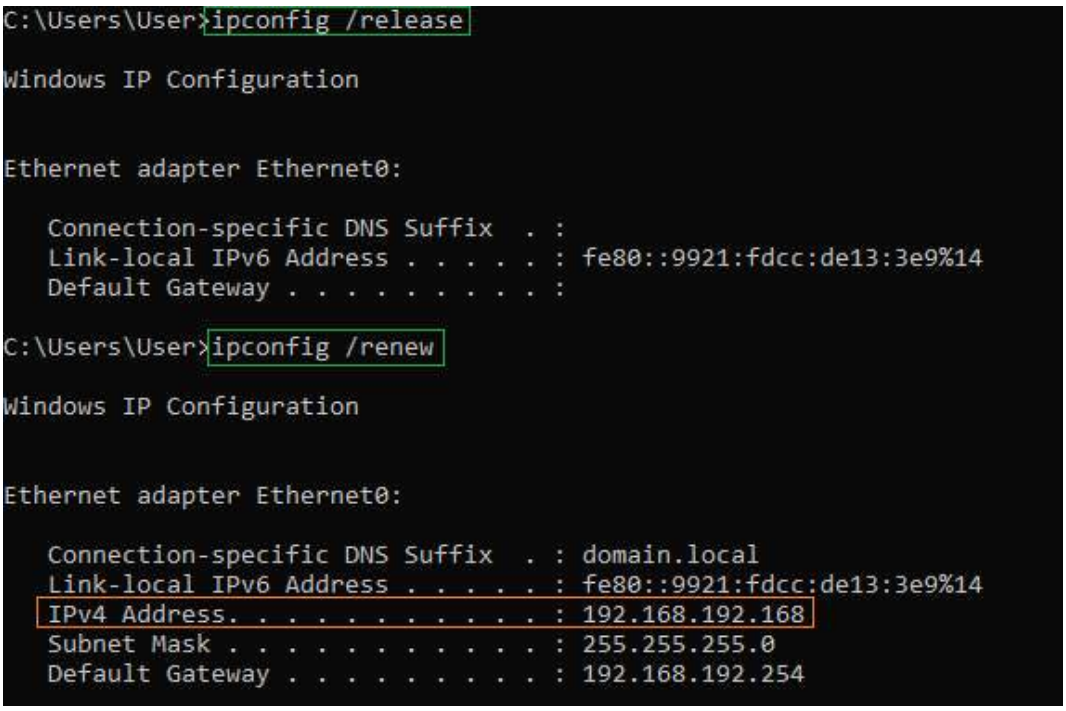
Memastikan tidak ada kesalahan pada konfigurasi menggunakan perintah `dhcpd -T`

```
root@srv1:/etc/bind# dhcpd -T
Internet Systems Consortium DHCP Server 4.4.3-P1
Copyright 2004-2022 Internet Systems Consortium.
All rights reserved.
For info, please visit https://www.isc.org/software/dhcp/
Config file: /etc/dhcp/dhcpd.conf
Database file: /var/lib/dhcp/dhcpd.leases
PID file: /var/run/dhcpd.pid
Wrote 0 leases to leases file.
Lease file test successful, removing temp lease file: /var/lib/dhcp/dhcpd.leases.1735178646
root@srv1:/etc/bind#
```

Selanjutnya restart isc-dhcp-server menggunakan perintah `systemctl restart isc-dhcp-server`

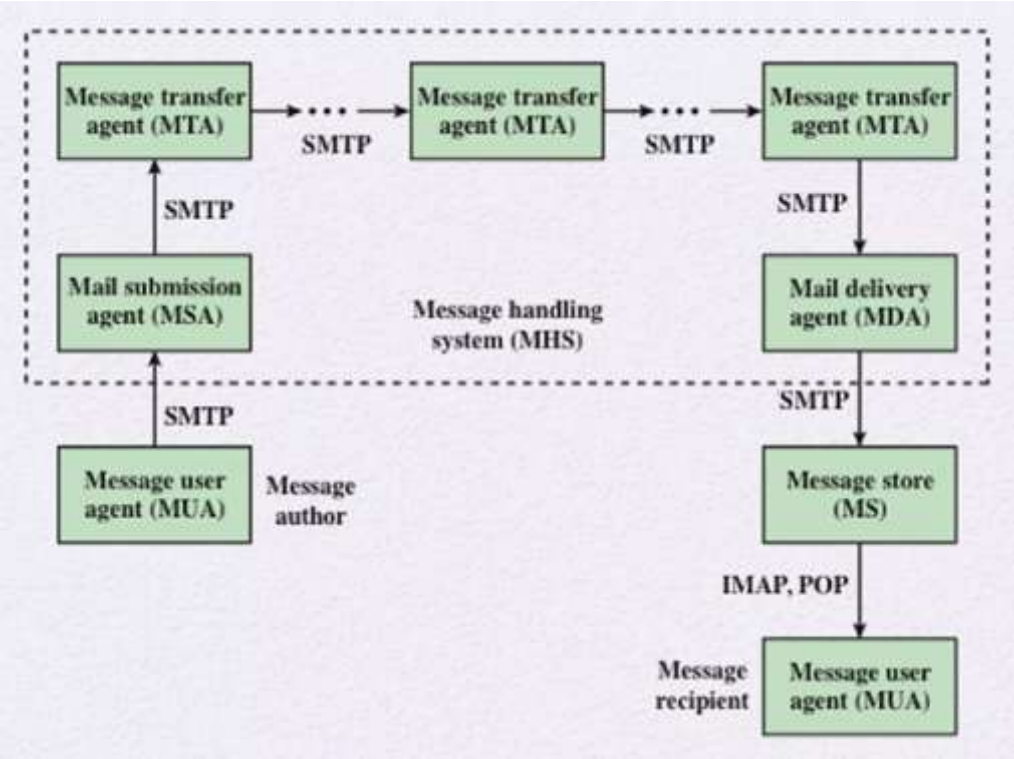
```
Lease file test successful, removing temp lease file: /var/lib/dhcp/dhcpd.leases.1735179165
root@srv1:/etc/bind# systemctl restart isc-dhcp-server
root@srv1:/etc/bind# _
```

Terakhir pada workstation, ketik perintah `ipconfig /release` lalu `ipconfig /renew` untuk mendapatkan IP address terbaru dari DHCP server. Pastikan workstation mendapatkan IP address terbaru.



Intro to Mail Services

Mail Services merupakan layanan yang digunakan untuk mengirim Email yang bersumber dari suatu domain ke Email domain tujuan. Perlu diingat, Mail Services bergantung pada DNS, khususnya DNS record bertipe MX. Berikut adalah gambar dari alur kerja Mail Services:



Sumber: https://www.slideshare.net/slideshow/electronic-mail-security-131902398/131902398
Dari gambar diatas, Mail Services terdiri dari beberapa komponen, yaitu MUA, MSA, MTA, MDA, dan MS. Berikut penjelasannya:

1. MUA (Mail User Agent)
   Merupakan komponen pada Mail Services yang digunakan pada sisi client. MUA digunakan oleh client untuk mengakses layanan Mail Services yang disediakan oleh suatu server. MUA dapat berupa application-based atau web-based. Contoh MUA Application-based adalah Windows Mail App dan Mozilla Thunderbird sedangkan web-based MUA contohnya Roundcube, Squirrelmail dan Gmail. **Intinya,**

MUA adalah komponen yang diakses oleh client untuk berkomunikasi dengan Mail Server.

2. MSA (Mail Submission Agent)
   Merupakan komponen yang menerima submisi email dari client yang dimana, client tersebut melakukan submisi /entry melalui MUA. Contoh aplikasi MSA adalah Postfix Submission dan Sendmail Submission. Port yang digunakan oleh pengguna untuk melakukan proses submisi email adalah port TCP/587. **Submission digunakan untuk agar Mail Server hanya mengirim email dari user yang telah terautentikasi**.
3. MTA (Mail Transfer Agent)
   Merupakan komponen utama dalam layanan Mail Services. MTA digunakan oleh Mail Server untuk saling berkomunikasi dengan Mail Server lainnya melalui port TCP/25. Setelah MSA menerima Email, proses selanjutnya adalah melakukan pengiriman Email ke domain yang dituju oleh MTA. Contoh aplikasi MTA adalah Postfix dan Sendmail.
4. MDA (Mail Delivery Agent)
   Merupakan komponen yang digunakan untuk mengirim email yang diterima dari MTA ke dalam Message Store setiap mail client. MDA ini bertanggung jawab dalam mensortir dan menyimpan email.
5. Message Store
   Merupakan komponen yang digunakan untuk menyimpan Email yang diterima oleh client dari MDA. Secara sederhana, Message Store adalah penyimpan email yang dimiliki oleh setiap mail user. Message Store akan diakses oleh MUA menggunakan protocol IMAP atau POP3.
   Saat client mendownload email dari Message Store menggunakan protokol IMAP, email tersebut masih tetap disimpan oleh mail server (backup email). Sedangkan protokol POP3, email yang didownload oleh client **tidak akan disimpan** pada mail server.

Dari penjelasan diatas, berikut adalah prosedur bagaimana client mengirim dan menerima suatu Email dari domain yang berbeda:

1. Client mengakses MUA (Mail User Agent) untuk berkomunikasi dengan Mail Server. Lalu membuat Email.
2. Setelah Email dibuat dan client menekan tombol kirim, Email akan diterima oleh MSA (Mail Submission Agent).
3. MSA akan mengirim Email yang disubmisi oleh client ke komponen MTA (Mail Transfer Agent)
4. MTA akan melihat domain dari destination address pada Email yang dikirim oleh MSA. Selanjutnya MTA akan melookup DNS MX record dari domain tersebut.

5. Setelah MTA mengetahui MX record dari domain, selanjutnya adalah proses pengiriman dari satu MTA ke MTA yang lain.
6. Saat Email berhasil terkirim ke MTA yang sesuai dengan domain pada Email destination address, selanjutnya MTA tersebut akan mengirim Email ke MDA (Mail Delivery Agent).
7. Setelah MDA menerima Email dari MTA, selanjutnya Email tersebut akan disimpan pada Message Store client sesuai dengan destination address yang ada pada Email.
8. Terakhir, Message Store akan mengirim Email yang diterima dari MDA ke MUA client menggunakan protocol IMAP/POP3

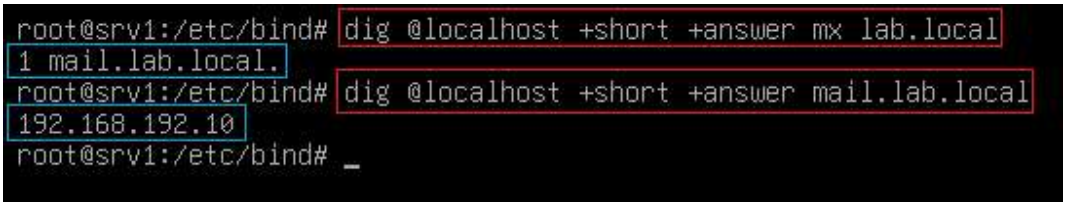
Basic Mail Services Configuration

Sebelum proses konfigurasi Mail Services, pada DNS zone lab.local perlu ditambahkan MX record. Tambahkan DNS record berikut pada file `/etc/bind/db.lab` yang ada pada srv1:

```
mail    IN  A       192.168.192.10
@       IN  MX 1    mail.lab.local.
```

Jika sudah, pastikan DNS record dapat diresolve seperti contoh berikut:

```
dig @localhost +short +answer mx lab.local
dig @localhost +short +answer mail.lab.local
```

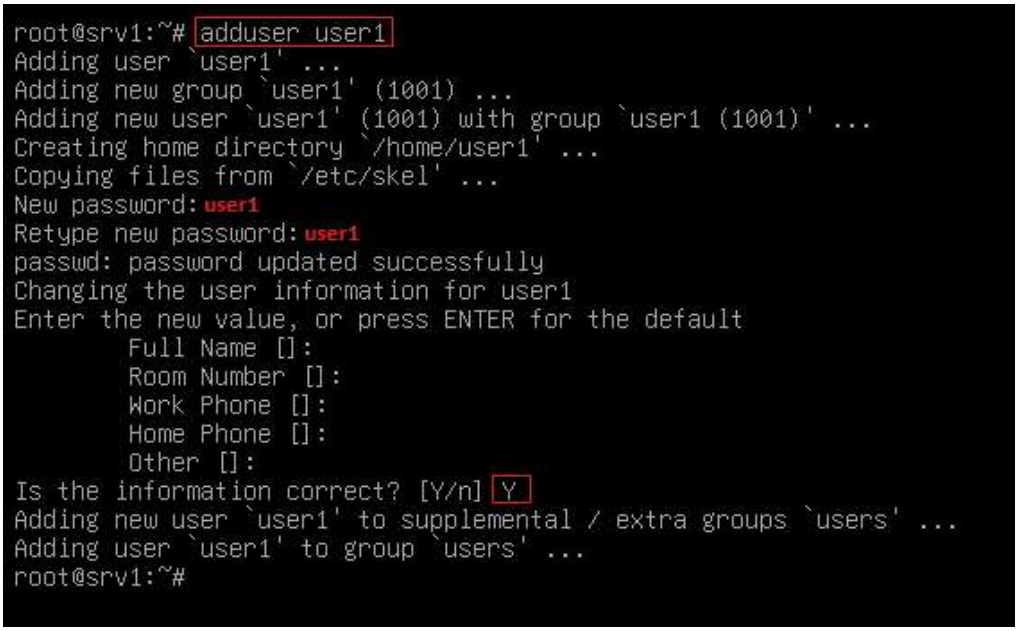


Konfigurasi Mail Services akan dilakukan pada **srv1**. Protocol digunakan adalah IMAP, POP3 dan Submission dengan SSL certificate yang telah disediakan pada direktori `/srv` (**mail.crt** dan **mail.key**). Berikut langkah – langkahnya:

1. Buat mail user terlebih dahulu menggunakan perintah `adduser` seperti berikut:

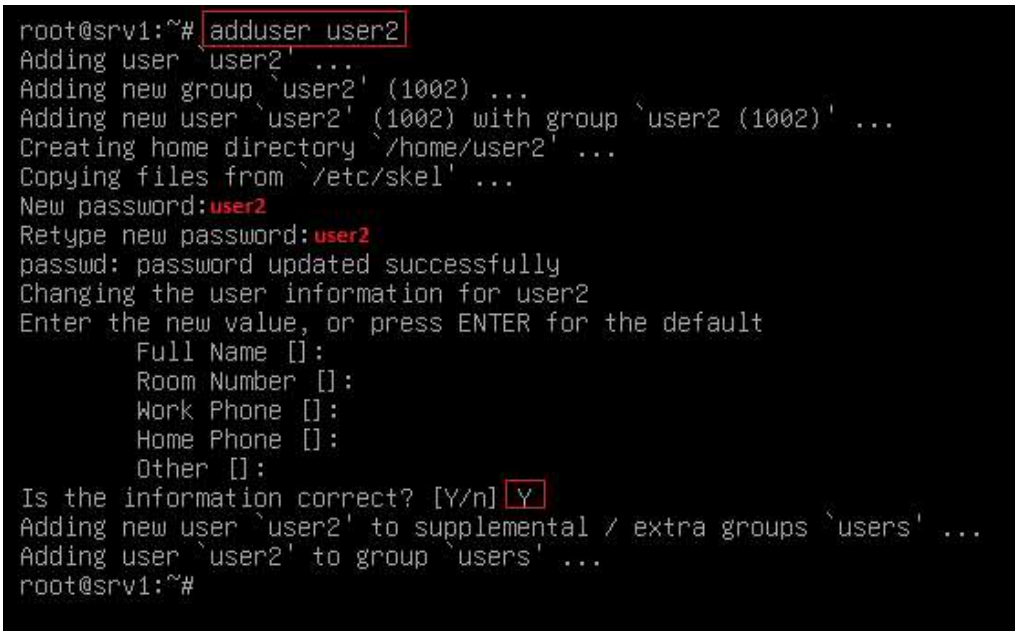
   ```
   adduser user1
   ```

   Contohnya:




```
adduser user2
```

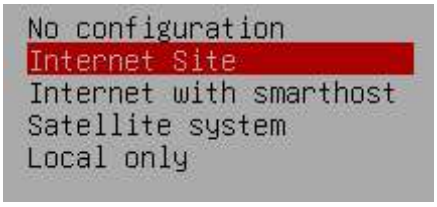
Contoh:



2. Install package postfix dan dovecot menggunakan perintah:

```
apt install postfix dovecot-imapd dovecot-pop3d -y
```

3. Saat proses instalasi, pilih **Internet Site**



4. Pada bagian system mail name, masukan nama domain name dari mail server. Contohnya **lab.local**



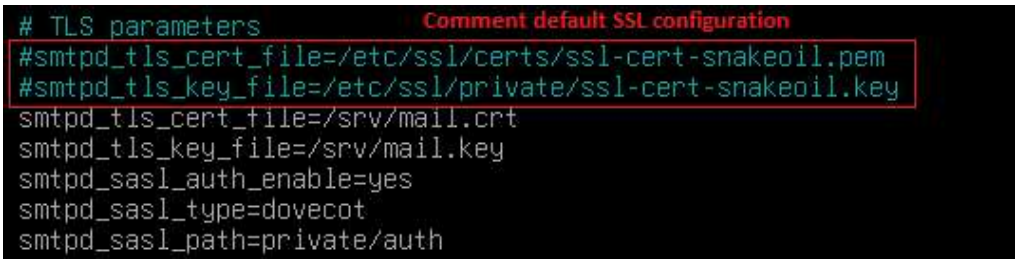
5. Lakukan konfigurasi postfix dengan perintah:

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

Lalu, tambahkan konfigurasi berikut dibawah # TLS:

```
smtpd_tls_cert_file=/srv/mail.crt
smtpd_tls_key_file=/srv/mail.key
smtpd_sasl_auth_enable=yes
smtpd_sasl_type=dovecot
smtpd_sasl_path=private/auth
```

Contohnya:



6. Selanjutnya mengaktifkan komponen submission (Mail Submission Agent) pada postfix yang akan digunakan oleh client untuk melakukan proses submisi email melalui port TCP/587. Caranya, hilangkan tanda pagar pada baris **19** sampai **24** pada file /etc/postfix/master.cf

```
#127.0.0.1:submission inet n -   y       -       -       smtpd
submission inet n       -       y       -       -       smtpd
  -o syslog_name=postfix/submission
  -o smtpd_tls_security_level=encrypt
  -o smtpd_sasl_auth_enable=yes
  -o smtpd_tls_auth_only=yes
  -o smtpd_reject_unlisted_recipient=no
#     Instead of specifying complex smtpd_<xxx>_restrictions here,
```

7. Cek konfigurasi postfix menggunakan perintah berikut dan pastikan tidak terdapat error apapun seperti **fatal error**.

   ```
   postconf
   ```

8. Selanjutnya ada proses konfigurasi komponen Message Store. Package yang digunakan adalah `dovecot`. Perintahnya adalah:

   ```
   nano /etc/dovecot/conf.d/10-master.conf
   ```

   Selanjutnya aktifkan konfigurasi authentication yang akan digunakan dengan cara menghilangkan tanda pagar dari baris **107** hingga **109**.

   ```
   # Postfix smtp-auth
   unix_listener /var/spool/postfix/private/auth {
     mode = 0666
   }
   ```
   Uncomment

9. Lakukan proses konfigurasi username format yang diterima oleh dovecot. Perintahnya seperti berikut:

   ```
   nano /etc/dovecot/conf.d/10-auth.conf
   ```

   Lalu, uncomment baris **51** dan lakukan konfigurasi berikut:

   ```
   # "-AT-". This translation is
   auth_username_format = %Ln
   ```

10. Konfigurasi SSL certificate beserta private key yang akan digunakan oleh dovecot. Caranya:

    ```
    nano /etc/dovecot/conf.d/10-ssl.conf
    ```

    Konfigurasi baris **12** dan **13** menjadi seperti berikut:

```
11 # certificate, just make sure to update the
12 ssl_cert = </srv/mail.crt
13 ssl_key = </srv/mail.key
```

11. Cek konfigurasi dovecot menggunakan perintah berikut. Jika sukses, restart service postfix dan dovecot.

```
doveconf 1> /dev/null && echo Success
systemctl restart postfix dovecot
```

```
root@srv1:/etc/dovecot# doveconf 1> /dev/null && echo Success
Success
root@srv1:/etc/dovecot#
root@srv1:/etc/dovecot# systemctl restart postfix dovecot
root@srv1:/etc/dovecot#
```

12. Terakhir, lakukan proses testing autentikasi menggunakan perintah berikut:

```
doveadm auth test user1@lab.local user1
doveadm auth test user2@lab.local user2
```

```
root@srv1:~# doveadm auth test user1@lab.local user1
passdb: user1@lab.local auth succeeded
extra fields:
  user=user1

  original_user=user1@lab.local
root@srv1:~#
root@srv1:~#
root@srv1:~# doveadm auth test user2@lab.local user2
passdb: user2@lab.local auth succeeded
extra fields:
  user=user2

  original_user=user2@lab.local
root@srv1:~#
```

Intro to Mail User Agent

Mail User Agent adalah salah satu komponen mail services yang digunakan oleh pengguna E-Mail untuk mengakses, mengirim dan menerima E-Mail dari mail server suatu domain. Mail User Agent terbagi menjadi dua yaitu *application-based* dan *web-based*.

MUA Application-based adalah MUA yang diinstall pada perangkat client sebagai *standalone application* untuk berkomunikasi dengan mail server, contohnya seperti **Windows Mail** dan **Mozilla Thunderbird**.

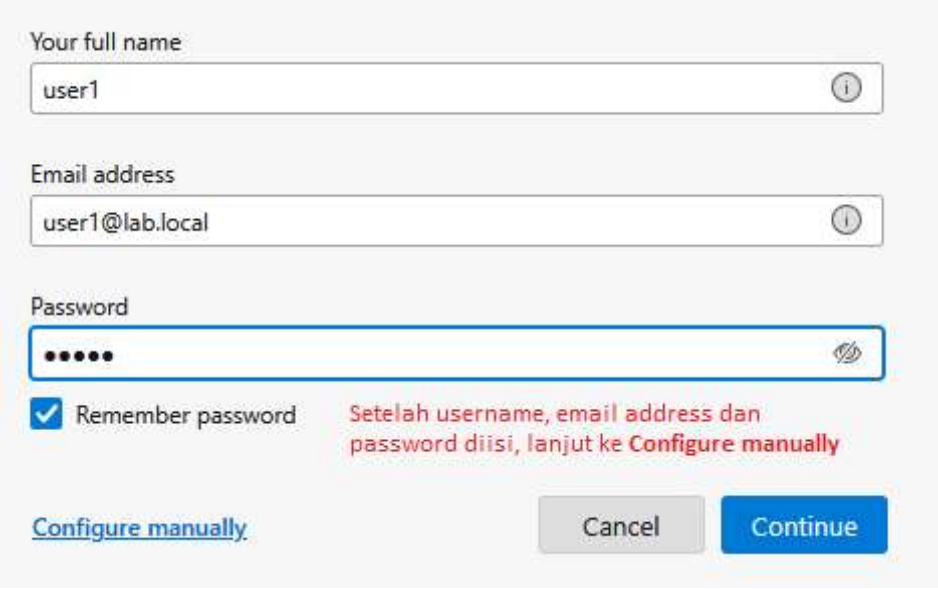
MUA Web-based merupakan MUA yang berbentuk aplikasi web dan diakses oleh client melalui browser sehingga pada client tidak perlu melakukan proses instalasi. Contoh MUA web-based antara lain **Gmail** dan **Roundcube**.

Application-based MUA

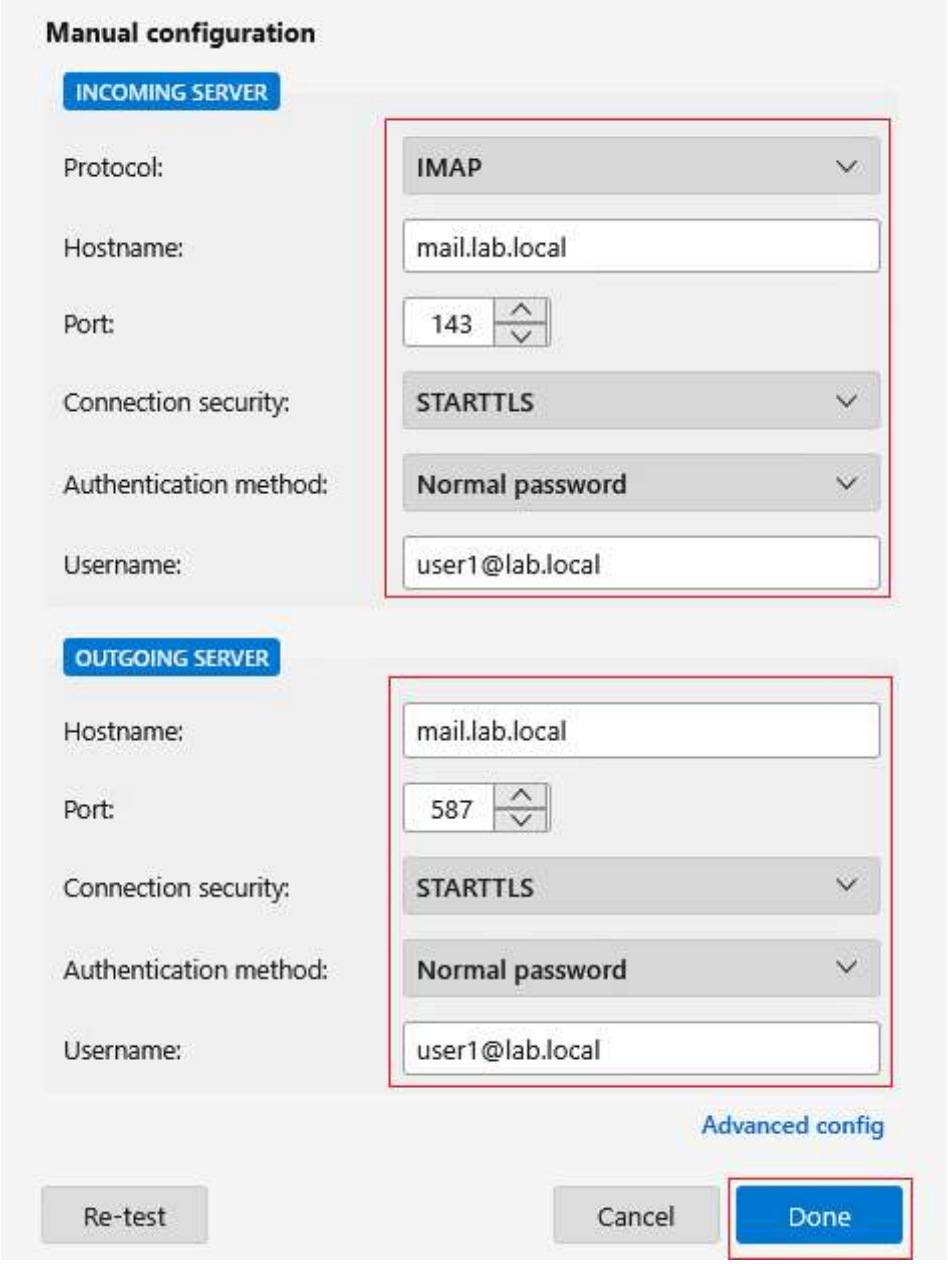
Seperti yang sudah disebutkan sebelumnya, application-based MUA adalah Mail User Agent berbasis aplikasi yang diinstall pada perangkat client sebagai *standalone application*. untuk berkomunikasi dengan mail server, contoh aplikasinya adalah Windows Mail dan Mozilla Thunderbird. Application-based MUA biasanya memerlukan konfigurasi ekstra pada saat proses setup. Contohnya seperti berikut yang dilakukan pada workstation.domain.local.

Setup account using IMAP Protocol

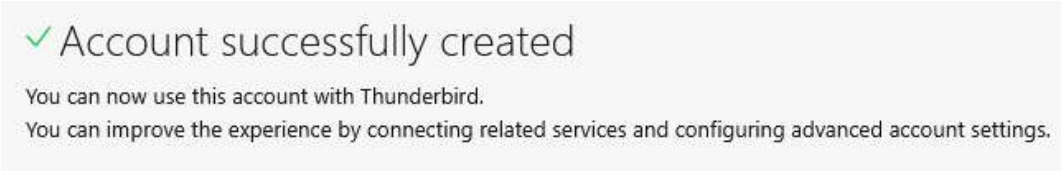
Buka Thunderbird, lalu setup **user1** seperti contoh berikut:



Pastikan protocol yang digunakan adalah **IMAP**

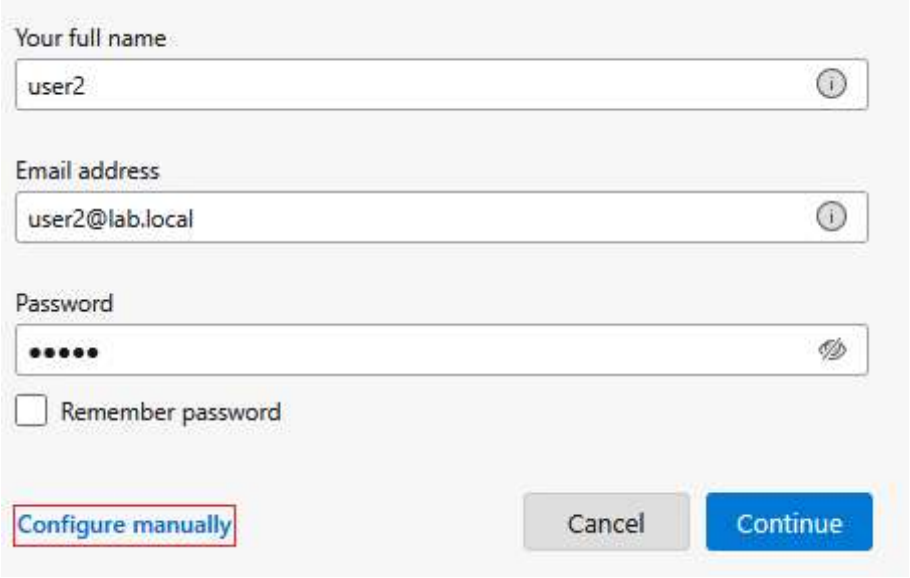


Setelah klik **Done**, pastikan terdapat notifikasi seperti berikut

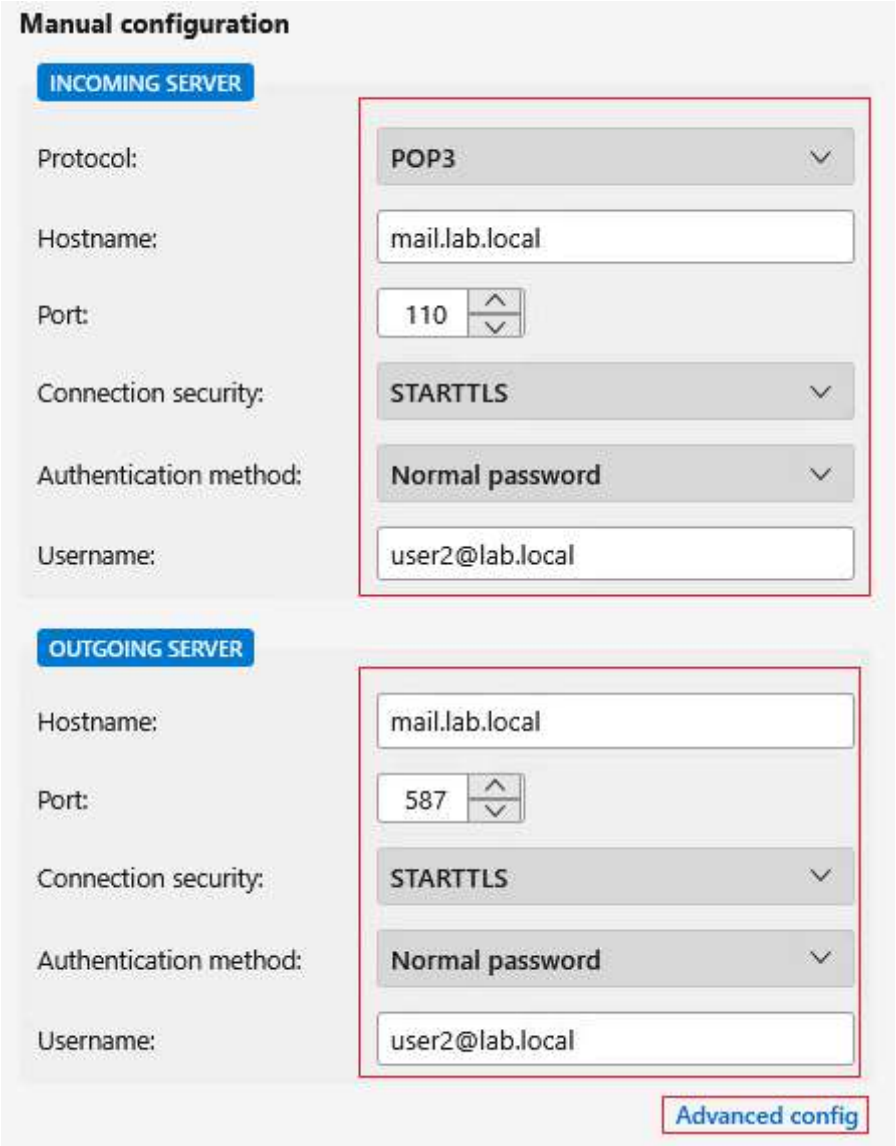


Setup account using POP3 Protocol

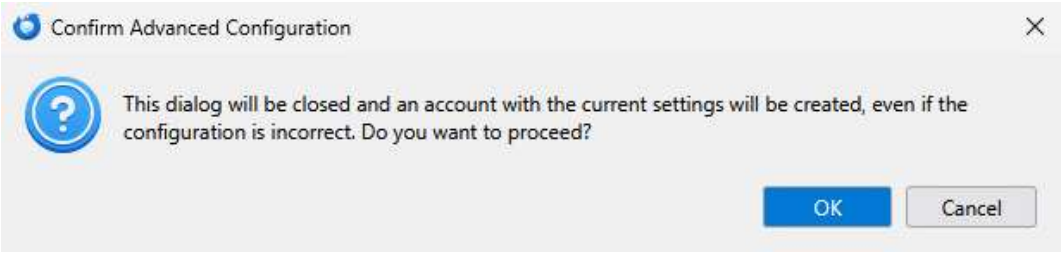
Setup kembali user2 yang akan menggunakan protocol POP3 pada Thunderbird seperti contoh berikut:



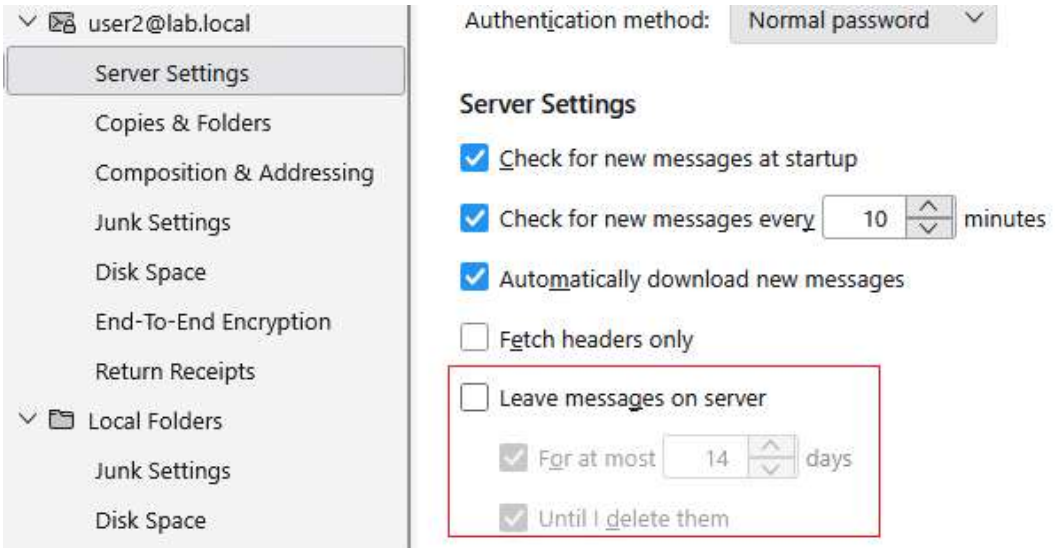
Pastikan protocol yang digunakan adalah POP3, lalu klik **Advanced Config**



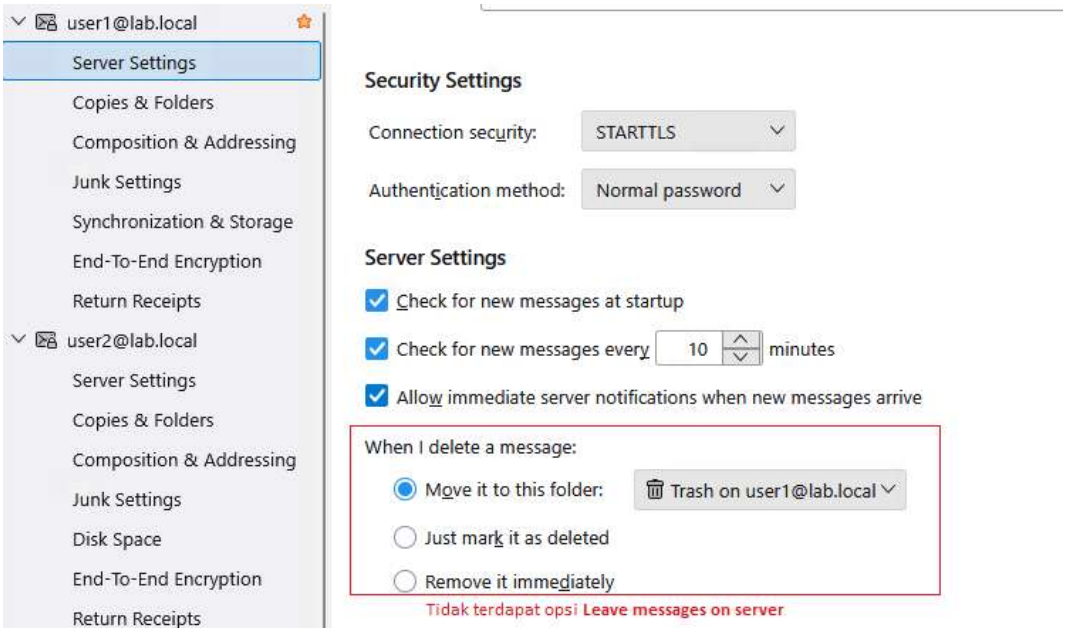
Pilih **OK** saat ada notifikasi seperti berikut:



Pada bagian **Server Settings user2**, nonaktifkan **Leave messages on server**



INFO:
Pada **Server Settings user1** yang menggunakan protocol IMAP, tidak tersedia opsi **Leave messages on server**

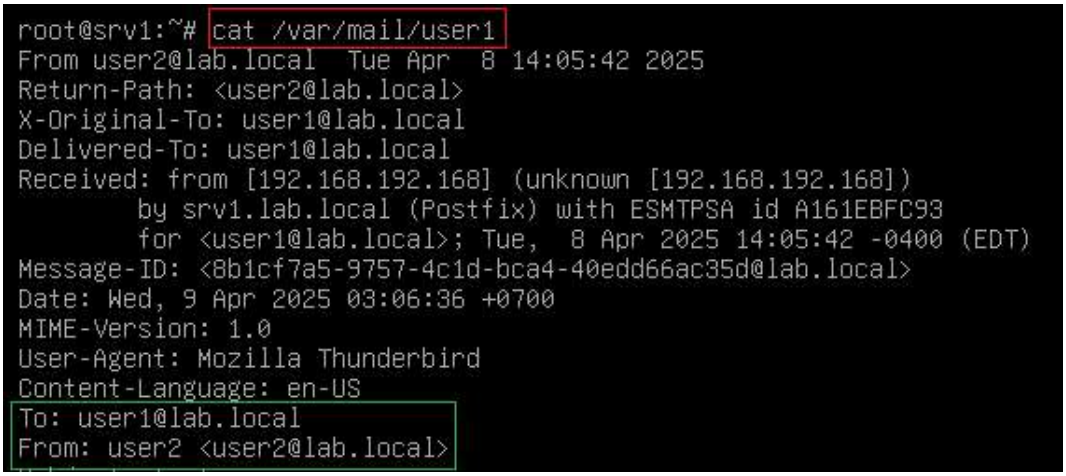


Testing Mail Services

IMAP

Pada workstation, buka Thunderbird dan kirim Email dari **user2** ke **user1**. Lalu, menuju ke **srv1** dan ketik perintah berikut untuk check apakah email berhasil dikirim atau tidak:

```
cat /var/mail/user1
```



Pada **workstation**, akan terdapat notifikasi bahwa ada email baru yang dikirim ke user1:



POP3

Selanjutnya proses testing user2 yang menggunakan protocol POP3. Kirim email dari **user1** ke **user2**. Lalu, menuju ke **srv1** dan ketik perintah berikut:

```
cat /var/mail/user2
```

```
root@srv1:~# cat /var/mail/user2
From user1@lab.local  Tue Apr  8 14:14:06 2025
Return-Path: <user1@lab.local>
X-Original-To: user2@lab.local
Delivered-To: user2@lab.local
Received: from [192.168.192.168] (unknown [192.168.192.168])
        by srv1.lab.local (Postfix) with ESMTPSA id 25928BFC93
        for <user2@lab.local>; Tue,  8 Apr 2025 14:14:06 -0400 (EDT)
Message-ID: <b2debe77-7f51-4f65-8031-807ae4436b89@lab.local>
Date: Wed, 9 Apr 2025 03:14:09 +0700
MIME-Version: 1.0
User-Agent: Mozilla Thunderbird
Content-Language: en-US
To: user2@lab.local
From: user1 <user1@lab.local>
Subject: test pop3
Content-Type: text/plain; charset=UTF-8; format=flowed
Content-Transfer-Encoding: 7bit

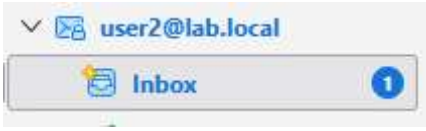
test pop3
```

Pada workstation, tidak terdapat notifikasi apapun pada **user2**. Untuk mendapatkan email dari mail server, buka profile **user2** dan tekan **Read messages**.



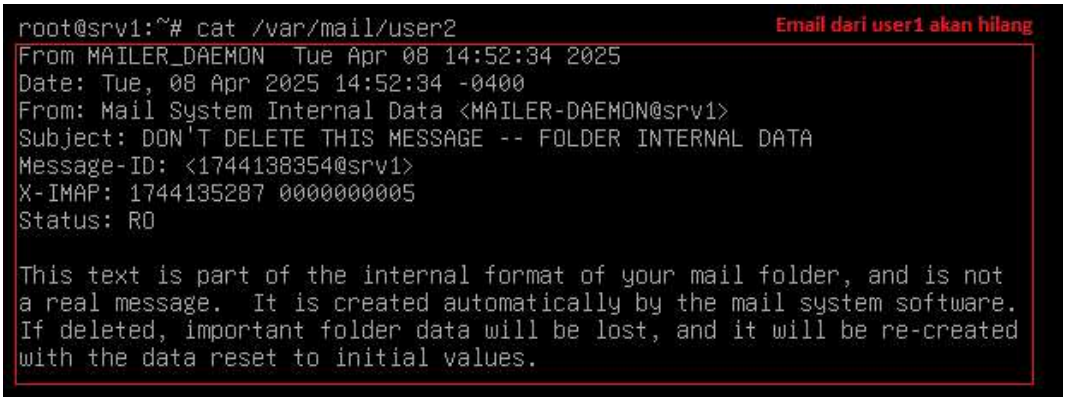


Setelah **Read Messages**, akan muncul notifikasi seperti berikut:



Lalu, kembali ke **srv1** dan ketik perintah:

```
cat /var/mail/user2
```



Previous Step

Next Step

Web-based MUA

Web-based MUA merupakan MUA yang berbentuk aplikasi web dan diakses oleh client melalui browser sehingga pada client tidak perlu melakukan proses instalasi. Contoh web-based MUA antara lain **Roundcube** dan **Gmail**. Proses setupnya tidak sebanyak Application-based MUA.
Pada praktikum berikut, akan dilakukan instalasi dan konfigurasi Web-based MUA pada **srv1.lab.local** menggunakan **Roundcube**. **Roundcube** memerlukan database sebagai tempat penyimpanan. Sebelum melakukan konfigurasi Roundcube, pastikan **srv1.lab.local** dapat melookup **mail.lab.local** seperti contoh berikut:

```
root@srv1:~# nslookup mail.lab.local
Server:         127.0.0.1
Address:        127.0.0.1#53

Name:   mail.lab.local
Address: 192.168.192.10
```

Roundcube secara default hanya support **IMAP** protocol. Lihat informasi pada link berikut.

Berikut langkah konfigurasi Roundcube yang dilakukan pada **srv1.lab.local**:

Roundcube website configuration

1. Install mariadb-server package menggunakan perintah berikut:

   ```
   apt install mariadb-server -y
   ```

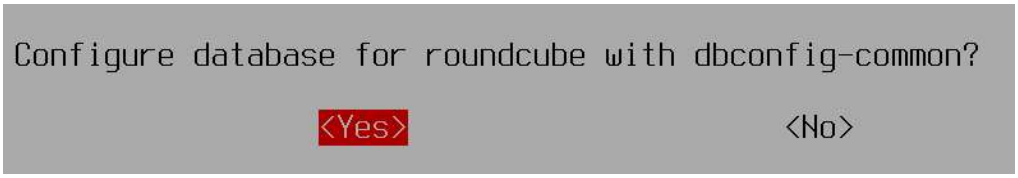
2. Buat user database pada mariadb server yang nantinya akan digunakan oleh Roundcube menggunakan perintah berikut:

```
mysql -u root -e'CREATE USER roundcube@"localhost" IDENT
```

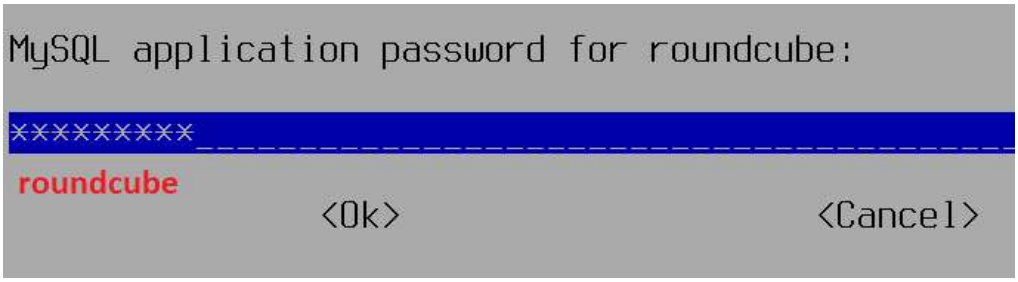
3. Selanjutnya install Roundcube menggunakan perintah berikut:

```
apt install roundcube -y
```

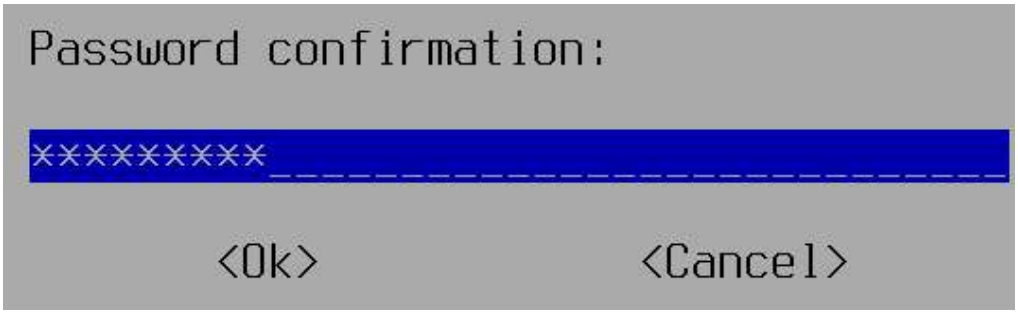
4. Pada bagian *configure database for roundcube*, pilih **Yes**



5. Pada bagian *MySQL application password for roundcube*, masukan password **roundcube**



6. Pada bagian konfirmasi password, masukan password **roundcube**



7. Selanjutnya, konfigurasi virtualhost untuk roundcube. Caranya buat file virtual host untuk roundcube dengan cara berikut:

```
nano /etc/apache2/sites-available/roundcube.conf
```

Isi file roundcube.conf dengan konten berikut:

```
<VirtualHost *:443>
ServerName mail.lab.local
```

```
SSLEngine on
SSLCertificateFile /srv/mail.crt
SSLCertificateKeyFile /srv/mail.key
DocumentRoot /var/lib/roundcube/public_html
</VirtualHost>
```

8. Check konfigurasi apache2 menggunakan perintah berikut:

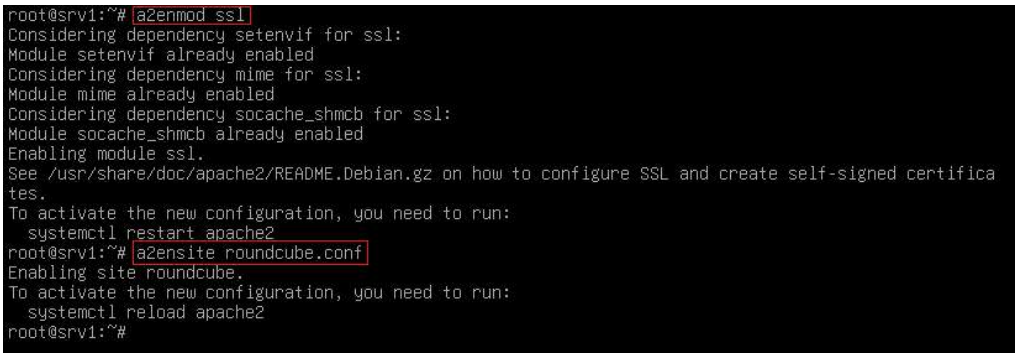
```
echo ServerName localhost >> /etc/apache2/apache2.conf
apachectl -t
```

Pastikan proses check konfigurasi berhasil seperti berikut:



9. Aktifkan module SSL dan virtualhost roundcube pada apache2 dengan menggunakan perintah berikut:

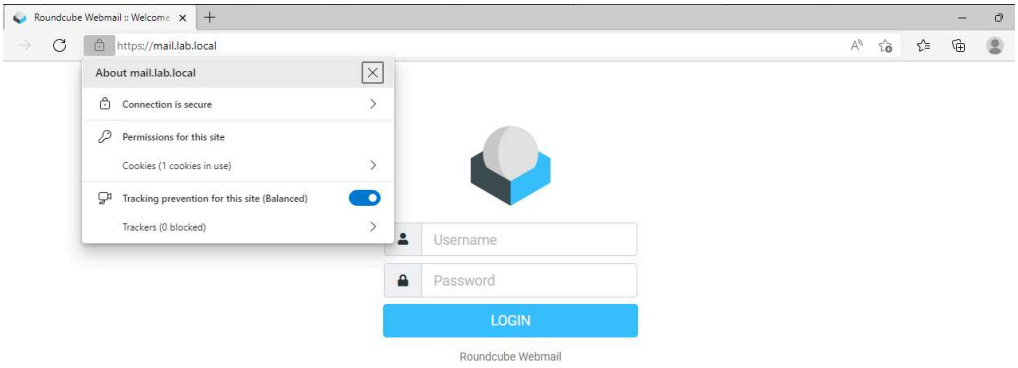
```
a2enmod ssl
a2ensite roundcube
```



Jika test sudah berhasil, restart apache2 menggunakan perintah berikut:

```
systemctl restart apache2
```

10. Buka **workstation.domain.local** lalu akses https://mail.lab.local menggunakan Microsoft Edge. Pastikan website roundcube dapat diakses seperti contoh berikut:



Roundcube mail configuration

1. Buka konfigurasi roundcube dengan perintah berikut:

   ```
   nano /etc/roundcube/config.inc.php
   ```

   Lakukan konfigurasi berikut:

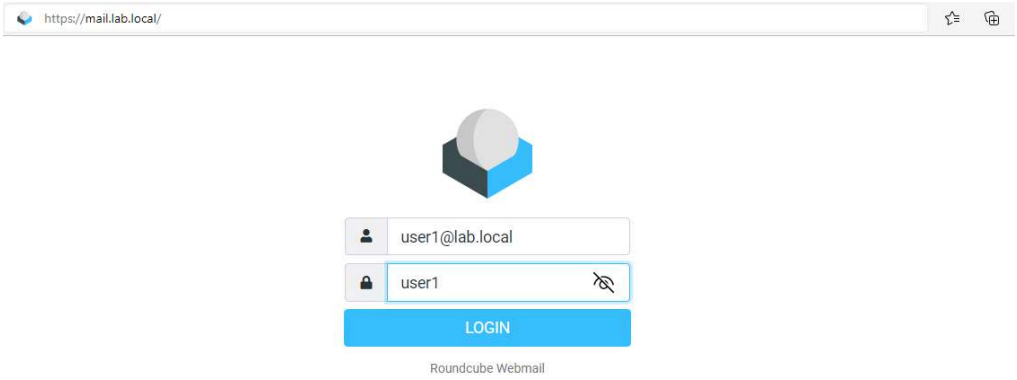
   ```
   // IMAP host chosen to perform the log-in.
   // See defaults.inc.php for the option description.
   $config['imap_host'] = ["tls://mail.lab.local:143"];

   // SMTP server host (for sending mails).
   // See defaults.inc.php for the option description.
   $config['smtp_host'] = 'tls://mail.lab.local:587';
   ```

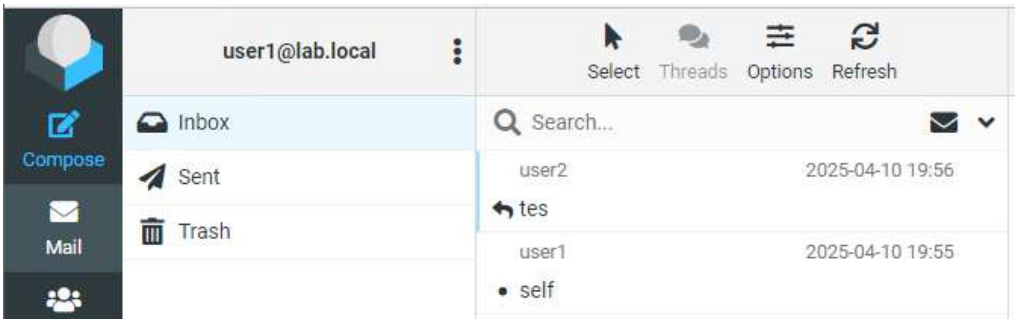
2. Restart apache2 menggunakan perintah berikut:

   ```
   systemctl restart apache2
   ```

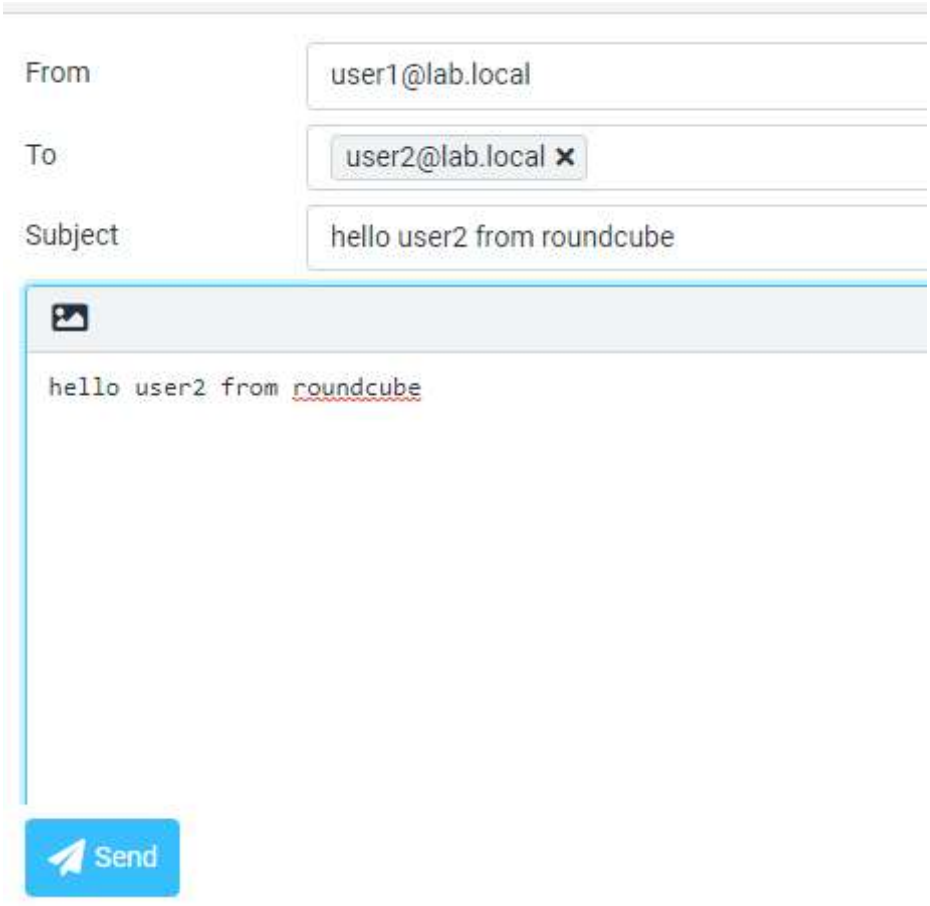
3. Testing roundcube dengan cara akses https://mail.lab.local lalu login sebagai **user1**:



Pastikan login berhasil seperti contoh berikut:

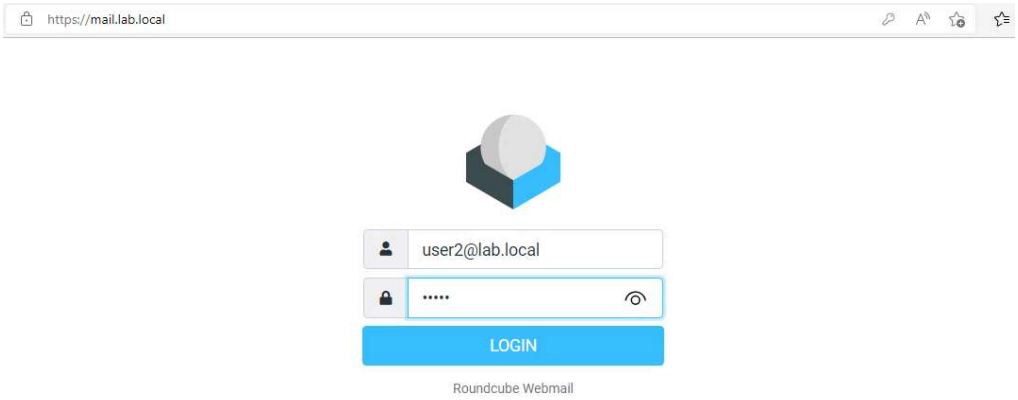


4. Lakukan testing mengirim email ke user1 ke user2

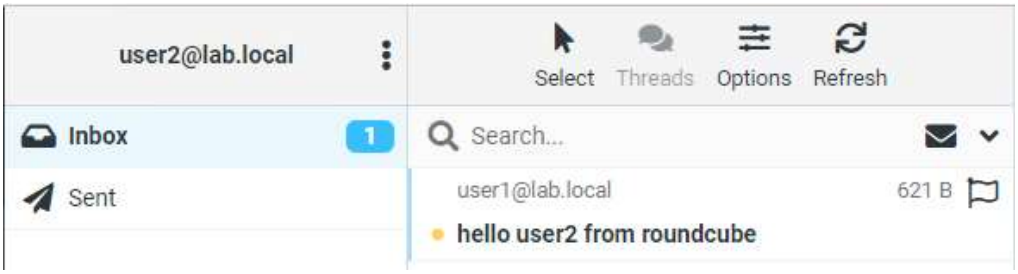


Lalu, setelah berhasil dikirim, logout akan **user1**

5. Pada roundcube, login sebagai user2



6. Setelah login, pastikan email dari user1 berhasil diterima oleh **user2**

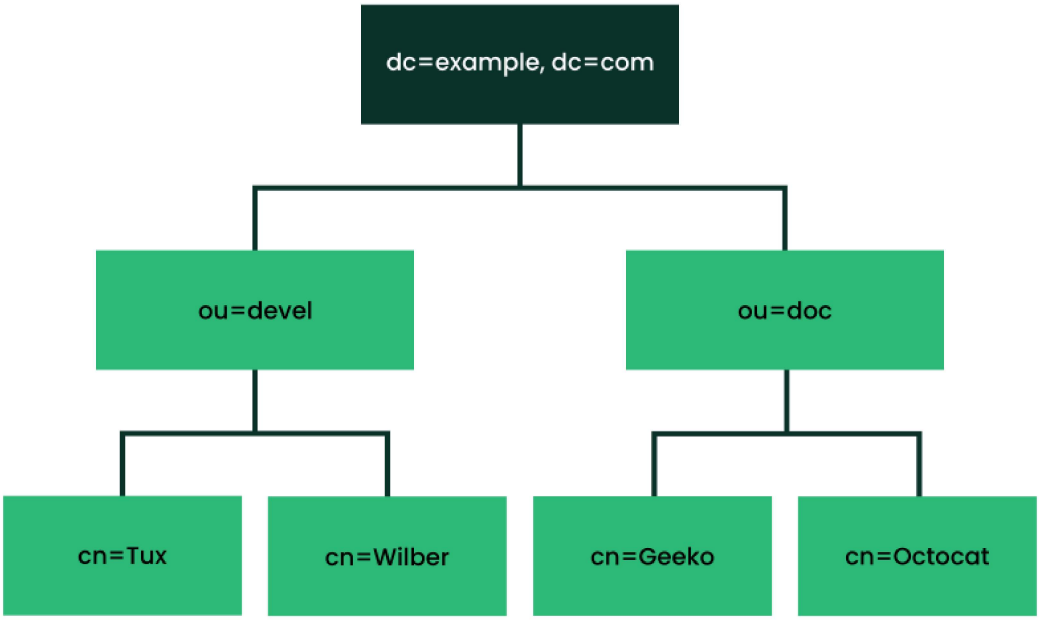


Intro to LDAP

LDAP merupakan services yang digunakan untuk menyimpan informasi seperti users dan groups. Users dan groups yang disimpan pada LDAP tadi dapat digunakan oleh host atau server lain yang diatur sebagai LDAP client. Suatu server yang dikonfigurasi sebagai LDAP server bisa dikonfigurasi sebagai LDAP client secara bersamaan. Tujuan utama penggunaan services LDAP ini adalah untuk autentikasi tersentral (centralized authentication).

Saat instalasi LDAP, perlu didefinisikan terlebih dahulu nama domain yang nantinya digunakan sebagai identifier LDAP database. Satu LDAP database memiliki satu domain name. Domain name ini akan dipecah menjadi beberapa bagian dan setiap bagian akan dimasukan ke dalam atribut LDAP yang bernama dc (Domain Component). Selain atribut dc, terdapat atribut LDAP lain seperti **ou (Organizational Unit)**, **cn (Common Name)**, **uid (User Identifier)**.

Pada LDAP, terdapat LDAP object yang memiliki atribut dan akan disimpan pada LDAP database. Contohnya seperti gambar LDAP database berikut yang menggunakan domain name **example.com**:



Sumber: https://documentation.suse.com/es-es/sled/15-SP5/html/SLED-all/cha-security-ldap.html

Setiap kata yang ada pada domain akan dipecah dan dimasukan ke dalam atribut **dc**. Lalu, dibawahnya terdapat atribut ou yang digunakan sebagai *container*. Di dalam setiap ou, terdapat LDAP object dengan atribut **cn**.

Di dalam LDAP database satu admin user yang dapat digunakan untuk memodifikasi services LDAP. User tersebut memiliki atribut **cn=admin** dan diikuti dengan atribut dc. Gabungan antara atribut suatu LDAP object dengan domain component disebut sebagai **Distinguished Name**.

Video tentang LDAP:
https://youtu.be/WBNV9VD1Oow

LDAP Configuration

Praktikum konfigurasi LDAP akan dilakukan pada srv1 dengan menggunakan domain **lab.local**. Lalu, server **srv1** dan **srv2** akan dijadikan sebagai LDAP client. Berikut langkah konfigurasinya:

LDAP Server

1. Install package slapd dan ldap-utils menggunakan perintah:

```
apt install slapd ldap-utils -y
```



2. Masukan password untuk user admin LDAP (**cn=admin**)



3. Cek LDAP database menggunakan perintah slapcat

```
root@srv1:/etc/rootCA# slapcat
dn: dc=lab,dc=local
objectClass: top
objectClass: dcObject
objectClass: organization
o: lab.local
dc: lab
structuralObjectClass: organization
entryUUID: 2ae392b6-9cf4-103f-8b30-0b9d7d26271d
creatorsName: cn=admin,dc=lab,dc=local
createTimestamp: 20250324120631Z
entryCSN: 20250324120631.328155Z#000000#000#000000
modifiersName: cn=admin,dc=lab,dc=local
modifyTimestamp: 20250324120631Z
```

4. Tambahkan konfigurasi LDAP berikut file `~/ldaprc` menggunakan perintah berikut:

```
nano ~/ldaprc
```

```
BASE dc=lab,dc=local
BINDDN cn=admin,dc=lab,dc=local
```

- **BASE** digunakan untuk menentukan LDAP database mana yang akan ditarget oleh perintah ldap (Ingat, 1 LDAP database memiliki 1 domain sebagai identifier)
- **BINDDN** digunakan untuk menentukan user admin yang akan digunakan saat mengakses LDAP

5. Lakukan test autentikasi sebagai user admin dengan menggunakan perintah `ldapwhoami -x -W`

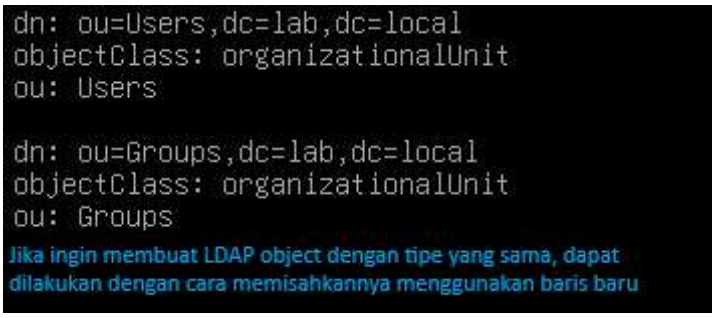
```
root@srv1:~# ldapwhoami -x -W
Enter LDAP Password:P@sswOrd
dn:cn=admin,dc=lab,dc=local
root@srv1:~# _
```

LDAP Object Creation

Terdapat berbagai macam tipe LDAP object yang dapat dibuat. Setiap tipe LDAP object dapat dibedakan melalui atribut objectClass. Pembuatan LDAP object dapat dilakukan dengan cara seperti berikut:

Pembuatan Organizational Unit

1. Buat file `ou.ldif` dengan konten seperti berikut



Perintahnya:

```
nano ou.ldif
```

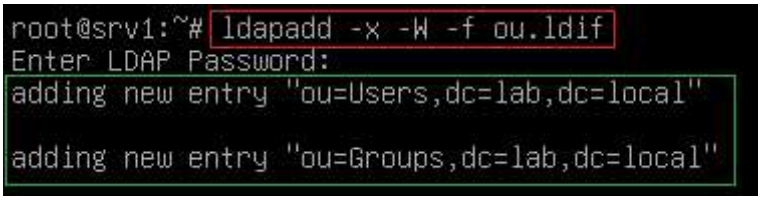
Tambahkan konfigurasi berikut dan pastikan spasi-nya:

```
dn: ou=Users,dc=lab,dc=local
objectClass: organizationalUnit
ou: Users

dn: ou=Groups,dc=lab,dc=local
objectClass: organizationalUnit
ou: Groups
```

2. Tambahkan LDAP object tersebut menggunakan perintah

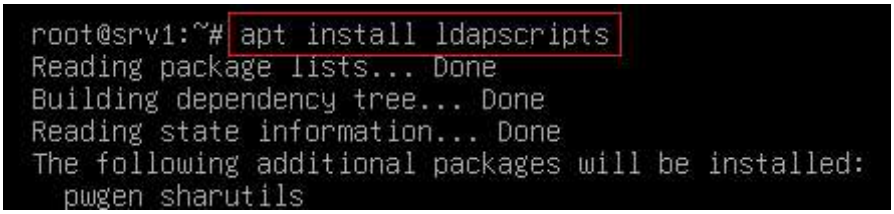
```
ldapadd -x -W -f ou.ldif
```



Pembuatan Group

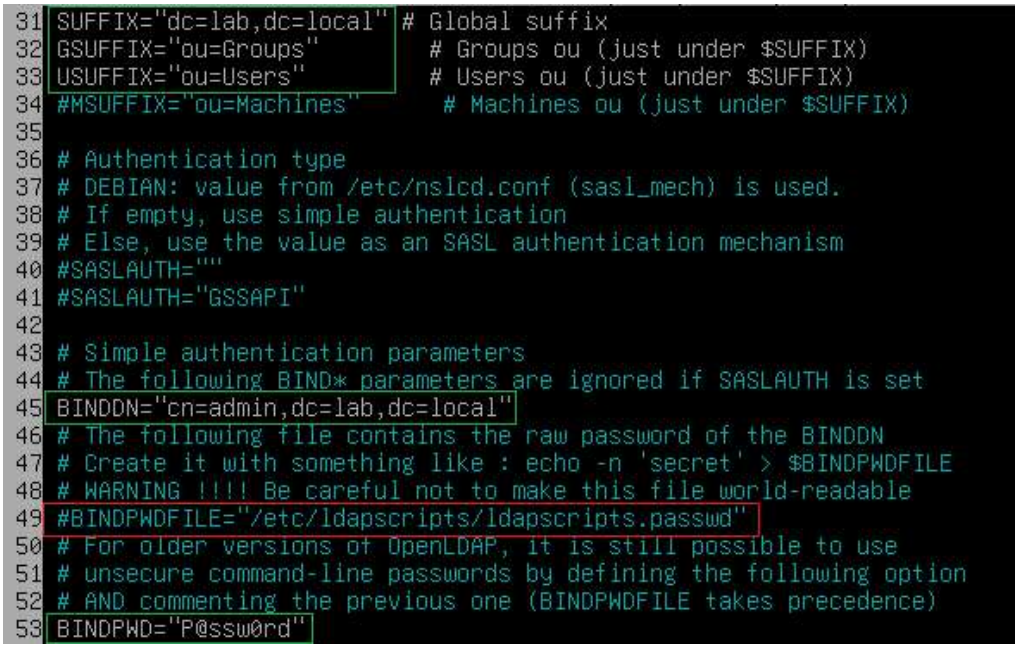
1. Install package ldapscripts terlebih dahulu

```
apt install ldapscripts -y
```



2. Lakukan konfigurasi berikut pada file /etc/ldapscripts/ldapscripts.conf

```
nano /etc/ldapscripts/ldapscripts.conf
```



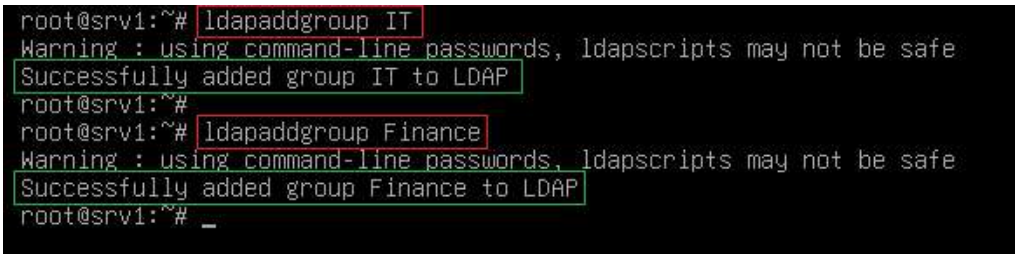
Masih di file yang sama, hapus komentar pada baris **89** dan komentari baris **90**

```
89 PASSWORDGEN="<ask>"
90 #PASSWORDGEN="pwgen"
```

3. Gunakan perintah `ldapaddgroup <NAMA_GROUP>` untuk membuat group. Contoh:

```
ldapaddgroup IT

ldapaddgroup Finance
```



Pembuatan User

Gunakan perintah `ldapadduser <NAMA_USER> <NAMA_GROUP>` . Contoh:

```
ldapadduser user1 IT
ldapadduser user2 Finance
```



Pastikan nama group yang akan dipakai sudah ada pada LDAP database. Jika tidak, maka akan terjadi error seperti berikut:

```
root@srv1:~# ldapadduser user1 test
Warning : using command-line passwords, ldapscripts may not be safe
Cannot resolve group test to gid : not found
root@srv1:~#
```

Checking

Untuk memastikan semua LDAP object telah dibuat, gunakan perintah

```
slapcat | grep dn
```

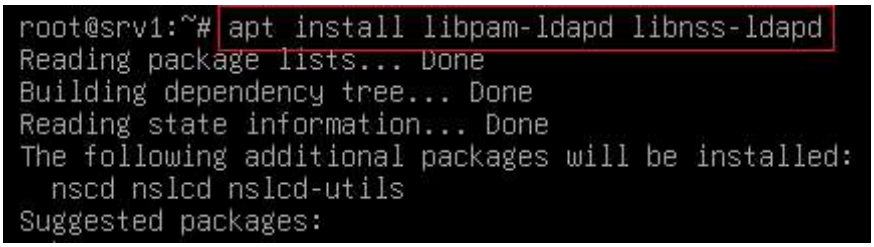
```
root@srv1:~# slapcat | grep dn
dn: dc=lab,dc=local
dn: ou=Users,dc=lab,dc=local
dn: ou=Groups,dc=lab,dc=local
dn: cn=IT,ou=Groups,dc=lab,dc=local
dn: cn=Finance,ou=Groups,dc=lab,dc=local
dn: uid=user1,ou=Users,dc=lab,dc=local
dn: uid=user2,ou=Users,dc=lab,dc=local
root@srv1:~#
```

LDAP Client

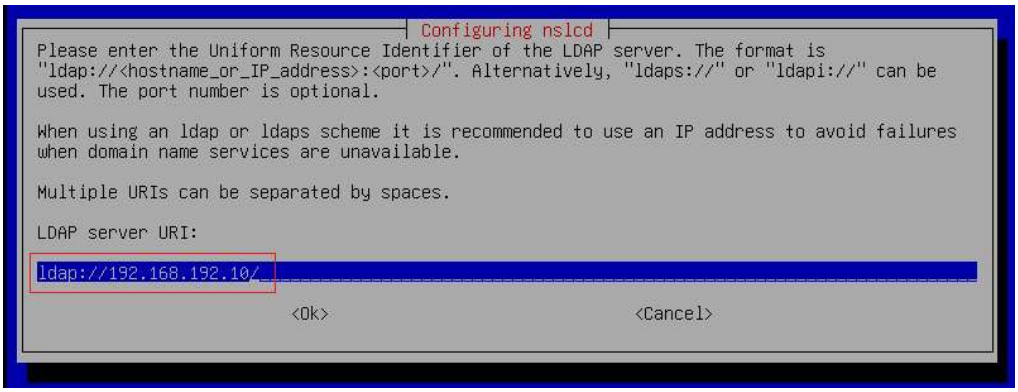
Lakukan konfigurasi LDAP client berikut pada **srv1** dan **srv2**

1. Install package `libnss-ldapd` dan `libpam-ldapd` menggunakan perintah:

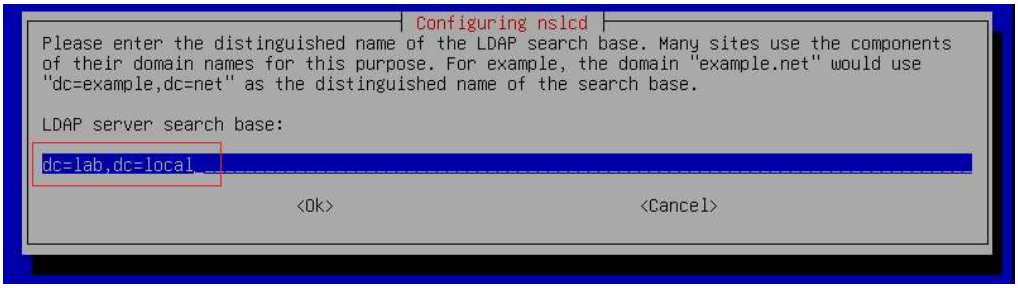
```
apt install libnss-ldapd libpam-ldapd
```



2. Masukan IP address **srv1** sebagai LDAP server



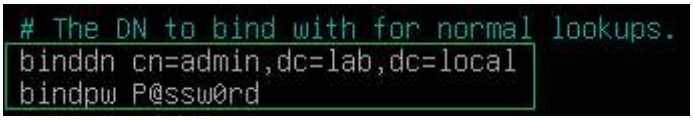
3. Pada bagian search base, pastikan atribut domain component memiliki nilai **dc=lab,dc=local**



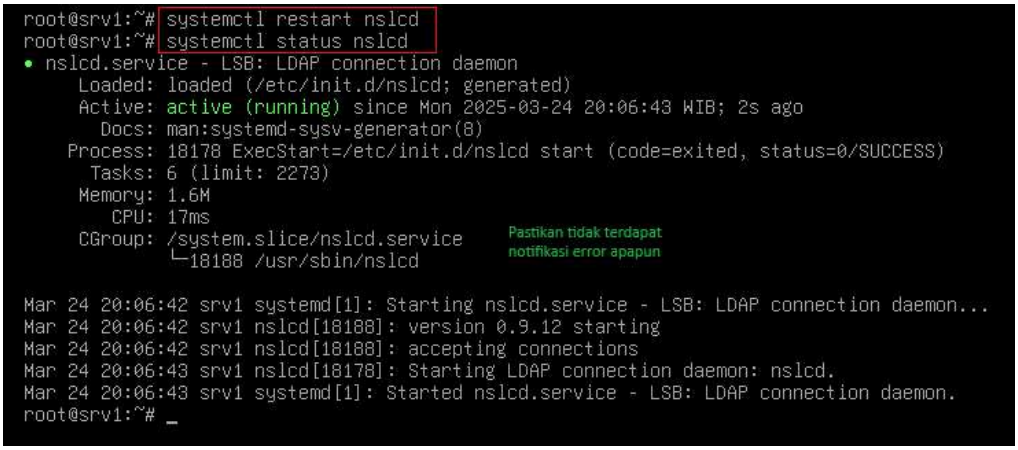
4. Pada bagian name services, pilih passwd , group dan shadow



5. Buka file /etc/nslcd.conf , tambahkan konfigurasi binding seperti berikut:



6. Restart layanan LDAP client menggunakan perintah systemctl restart nslcd dan check services tersebut menggunakan perintah systemctl status nslcd



7. Pastikan LDAP object dapat diakses oleh LDAP client menggunakan perintah:

```
getent passwd -s ldap
```

```
root@srv1:~# getent passwd -s ldap
user1:*:10000:10000:user1:/home/user1:/bin/bash
user2:*:10001:10001:user2:/home/user2:/bin/bash
root@srv1:~# _
```

8. Uji coba akses login menggunakan LDAP user seperti contoh berikut:

```
root@srv1:~# login user1
Password:P@ssw0rd
Linux srv1 6.1.0-18-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.76-1 (2024-02-01) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Mon Mar 24 23:05:03 WIB 2025 on tty2
No directory, logging in with HOME=/
user1@srv1:/$
```

9. Agar LDAP user membuat home directory saat login secara otomatis, ketik perintah berikut:

   ```
   pam-auth-update --enable mkhomedir
   ```

   Lalu login kembali sebagai user LDAP

```
root@srv1:~# pam-auth-update --enable mkhomedir
root@srv1:~#
root@srv1:~# login user1
Password:
Linux srv1 6.1.0-18-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.76-1 (2024-02-01) x86_64

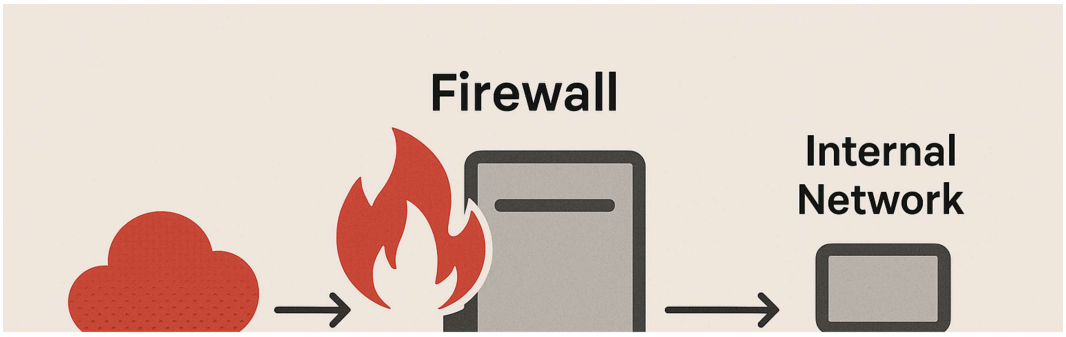
The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Mon Mar 24 23:06:05 WIB 2025 on tty2
Creating directory '/home/user1'.
user1@srv1:~$
```

Intro to Firewall

Firewall yang ada pada Linux merupakan fitur keamanan yang digunakan untuk memfilter traffic yang menuju ke dalam system, keluar dari system hingga traffic yang melewati sistem. Firewall ini dapat memfilter berbagai macam traffic yang dapat dikategorikan berdasarkan beberapa karakteristik seperti tipe protocol yang digunakan, source address, hingga destination address.
Implementasi firewall pada Linux dapat dilakukan menggunakan berbagai macam package seperti iptables, nftables, firewalld, dan ufw. Sejak Debian versi 10 (Buster), package firewall yang secara default sudah terinstall adalah nftables sehingga pada dokumentasi ini, package nftables akan digunakan untuk implementasi firewall.
Nftables merupakan pengganti iptables yang sebelumnya digunakan pada Linux Debian. Sintaks konfigurasi nftables lebih simple untuk ditulis daripada iptables tetapi konsepnya masih sama. Nftables secara default sudah memiliki chain **input**, **forward** dan **output** untuk melakukan filtering seperti iptables. Chain tersebut digunakan menyimpan rules atau peraturan terkait traffic yang akan difilter. Setiap chain memiliki jenis seperti berikut:

- Chain input digunakan untuk menyimpan rules traffic yang **MASUK** ke dalam sistem
- Chain forward digunakan untuk menyimpan rules traffic yang **MELEWATI** sistem
- Chain output digunakan untuk menyimpan rules traffic yang **KELUAR** dari sistem



Pada Debian 10 keatas, service nftables secara default sudah disediakan. Proses konfigurasinya dapat dilakukan pada file `/etc/nftables.conf` yang memiliki struktur seperti berikut:




```
#!/usr/sbin/nft -f

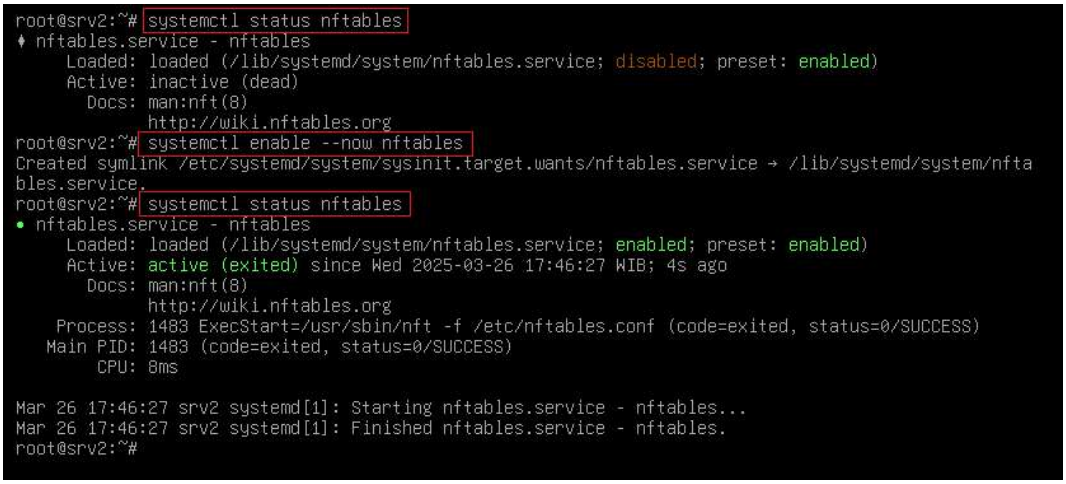
flush ruleset

table inet filter {
    chain input {
        type filter hook input priority filter;
    }
```

```
        chain forward {
            type filter hook forward priority filter;
        }

        chain output {
            type filter hook output priority filter;
        }
    }
```

Firewall rules untuk filter traffic yang masuk ke dalam sistem ditulis pada **chain input**, sedangkan untuk filter traffic yang keluar dari sistem ditulis pada **chain output**. Jika Linux dikonfigurasi sebagai router, chain forward dapat digunakan untuk menyimpan rules filter traffic yang melewati sistem.
Secara default services nftables belum berjalan. Untuk mengeceknya gunakan `systemctl status nftables` dan untuk mengaktifkannya gunakan perintah `systemctl enable --now nftables`

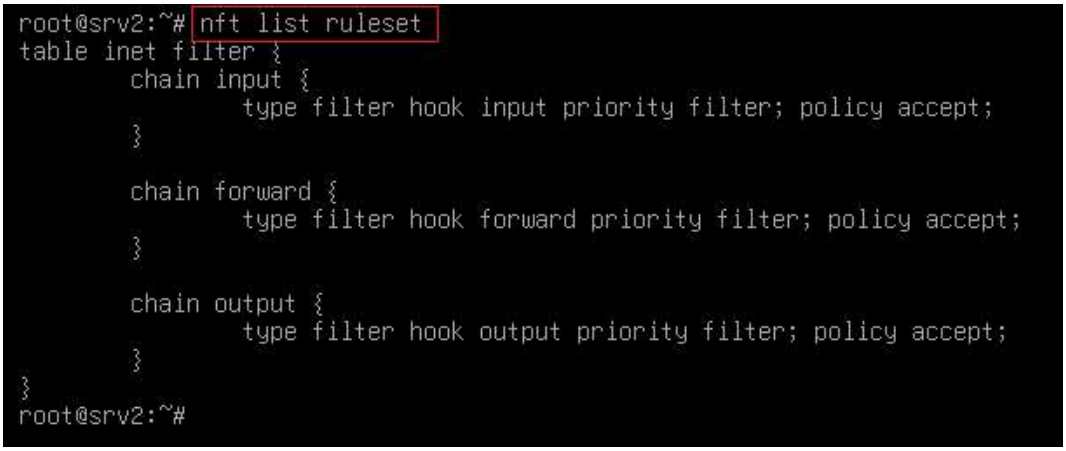



```
systemctl status nftables
```

```
systemctl enable --now nftables
```

```
systemctl status nftables
```

Untuk melihat firewall rules yang telah diaktifkan, gunakan `nft list ruleset`



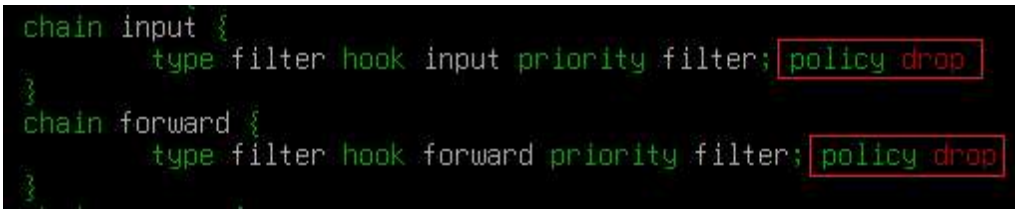

```
nft list ruleset
```

Setiap kali proses konfigurasi nftables dilakukan, reload service nftables menggunakan systemctl reload nftables

Firewall using Nftables

Seperti yang sudah dijelaskan sebelumnya, mulai dari versi Debian 10 Buster firewall yang digunakan oleh Debian adalah Nftables. Secara default, konfigurasi firewall nftables dapat dilakukan pada file `/etc/nftables.conf`. Untuk melihat konfigurasi nftables yang sudah diterapkan pada sistem, gunakan perintah `nft list ruleset`
Berikut beberapa sintaks konfigurasi terkait firewall menggunakan Nftables:

1. Policy definition
   Policy digunakan sebagai kebijakan yang akan dilakukan untuk traffic yang tidak tertulis pada suatu chain. Konfigurasi dilakukan pada suatu chain dengan contoh seperti berikut:



2. Connection tracking menggunakan conntrack
   Sintaks:

```
ct state {related, established} [comment …] [counter] ac
ct state new <RULES>
```

   Konfigurasi stateful connection umumnya ditambahkan pada baris pertama suatu chain. Seperti contoh berikut:



   Check connection tracking menggunakan perintah `nft list ruleset`

```
chain input {
        type filter hook input priority filter; policy drop;
        ct state { established, related } counter packets 44 bytes 19145 accept
        ct state new tcp dport { 22, 80, 389, 443 } counter packets 1 bytes 60 accept
```

3. Input traffic rules (TCP, UDP, ICMP)
   Sintaks:

```
ct state new <tcp|udp> dport <SERVICE_NAME/DEST_PORT> [c
ct state new icmp type echo-request accept
```

   Contoh:

```
chain input {
        type filter hook input priority filter; policy drop
        ct state {related,established} counter accept
        ct state new tcp dport {http, https, ldap, ssh} counter accept
        ct state new udp dport 53 accept comment "Permit DNS traffic"
        ct state new tcp dport 53 accept comment "Permit DNS zone transfer"
        ct state new icmp type echo-request accept
}
```

Rules pada input chain

4. Output traffic rules (TCP, UDP, ICMP)
   Sintaks:

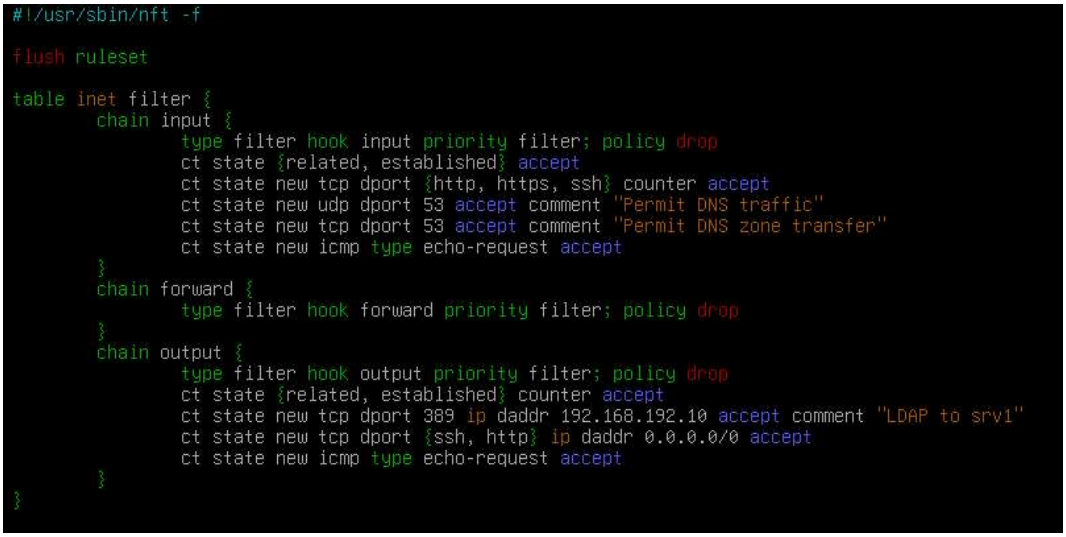
```
ct state new <tcp|udp> dport <SERVICE_NAME/DEST_PORT> [i
ct state new icmp type echo-request accept
```

```
chain output {
    type filter hook output priority filter; policy drop
    ct state {related, established} counter accept
    ct state new tcp dport 53 ip daddr 192.168.192.20 accept comment "Zone transfer to srv2"
    ct state new tcp dport {ssh, http} ip daddr any accept
    ct state new icmp type echo-request accept
}
```

Rules dibuat pada output chain

Example Configuration

Berikut adalah contoh nftables rules yang dapat diterapkan pada **srv2**



```
#!/usr/sbin/nft -f

flush ruleset

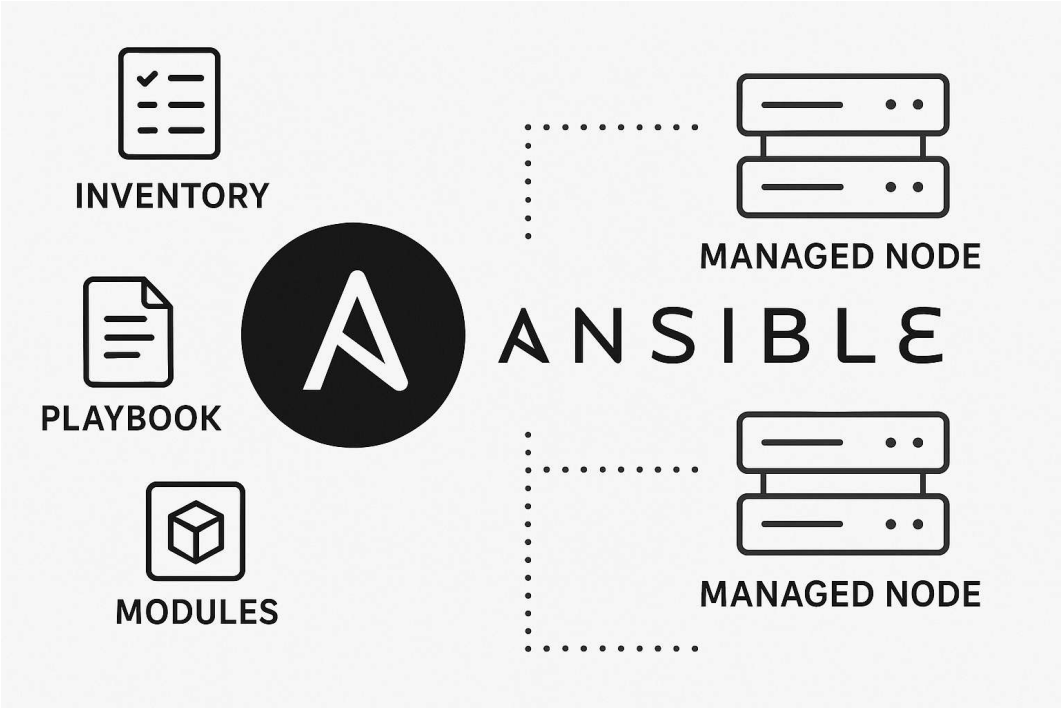
table inet filter {
        chain input {
                type filter hook input priority filter; policy drop
                ct state {related, established} accept
                ct state new tcp dport {http, https, ssh} counter accept
                ct state new udp dport 53 accept comment "Permit DNS traffic"
                ct state new tcp dport 53 accept comment "Permit DNS zone transfer"
                ct state new icmp type echo-request accept
        }
        chain forward {
                type filter hook forward priority filter; policy drop
        }
        chain output {
                type filter hook output priority filter; policy drop
                ct state {related, established} counter accept
                ct state new tcp dport 389 ip daddr 192.168.192.10 accept comment "LDAP to srv1"
                ct state new tcp dport {ssh, http} ip daddr 0.0.0.0/0 accept
                ct state new icmp type echo-request accept
        }
}
```

Rules:

```
table inet filter {
    chain input {
    type filter hook input priority filter; policy drop
        ct state {related, established} counter accept
        ct state new tcp dport {http, https, ssh} counter a
        ct state new udp dport 53 accept comment "Permit DI
        ct state new tcp dport 53 accept comment "Permit DI
        ct state new icmp type echo-request accept
    }
    chain forward {
    type filter hook forward priority filter; policy drop
    }
    chain input {
    type filter hook input priority filter; policy drop
        ct state {related, established} counter accept
        ct state new tcp dport 389 ip daddr 192.168.192.10
        ct state new tcp dport {ssh, http} ip daddr 0.0.0.(
        ct state new icmp type echo-request accept
    }
}
```

Intro to Ansible

Ansible merupakan automation tools yang dibuat menggunakan bahasa pemrograman Python untuk melakukan otomatisasi pekerjaan terkait IT seperti deployment, services configuration hingga security config. Ansible dapat menarget berbagai macam platform mulai dari server hingga network devices. Hal tersebut dapat dilakukan karena Ansible secara default menggunakan protocol SSH untuk berkomunikasi dengan target yang akan diotomatisasi. Untuk menggunakan ansible sebagai automation tools, pengguna perlu melakukan beberapa hal seperti melakukan ansible configuration, credentials configuration, dan inventory creation. Pada dokumentasi ini, srv1 akan digunakan sebagai Ansible Controller (Controller Node) sedangkan srv2 sebagai Ansible Target (Managed Node). Pada Ansible terdapat beberapa istilah yang harus diketahui seperti:



- **Ansible Controller / Controller Node**, merupakan host yang dipasang Ansible dan dipakai untuk otomatisasi Managed Node.
- **Ansible Target / Managed Node**, merupakan host yang ditarget oleh Ansible untuk dilakukan otomatisasi.

- **Ansible Inventory**, merupakan file yang didalamnya berisikan list Managed Node. Di dalamnya berisikan IP address dari Managed Node.
- **Ansible Configuration**, merupakan konfigurasi ansible yang mencakup beberapa hal seperti file inventory yang akan digunakan serta connection settings.
- **Playbook**, merupakan file yang didalamnya terdapat perintah Ansible untuk melakukan otomatisasi. Ditulis dalam format YAML.
- **Group Variables**, merupakan variable yang diterapkan pada group yang tertulis pada ansible inventory. Berisikan informasi seperti tipe koneksi untuk mengakses node serta username dan password sebagai credentials

Untuk mengetahui lebih lanjut tentang Ansible, kunjungi berikut https://docs.ansible.com/ansible/latest/getting_started/introduction.html

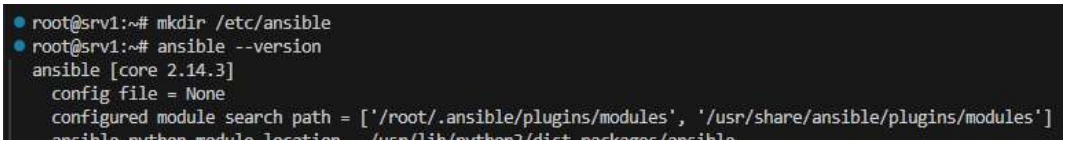
Vscode Remote-SSH for Ansible

Buka workstation.domain.local (Windows 11) dan lakukan langkah - langkah berikut

Pada dokumentasi ini, ansible controller akan diakses melalui **workstation.domain.local** menggunakan Visual Studio Code. Saat membuka vscode, lihat pada bagian pojok kiri bawah dan pastikan koneksi SSH berhasil dilakukan seperti berikut:



Pada vscode, tekan **Ctrl +** ` untuk membuka terminal, lalu buat direktori /etc/ansible. Terakhir ketik **ansible --version**



Install ansible

```
apt install ansible -y
```

Setup Linux for Ansible Target

Lakukan konfigurasi SSH berikut pada srv2

Sebelum melakukan konfigurasi ansible, pastikan terlebih dahulu koneksi SSH dapat dilakukan dari **Controller (srv1)** ke **Managed Node (srv2)**. Pada srv2, SSH sudah terinstall dan terkonfigurasi agar menerima login menggunakan user root. Untuk memastikannya lebih lanjut, tes SSH melalui terminal:

```
nano /etc/ssh/sshd_config
```

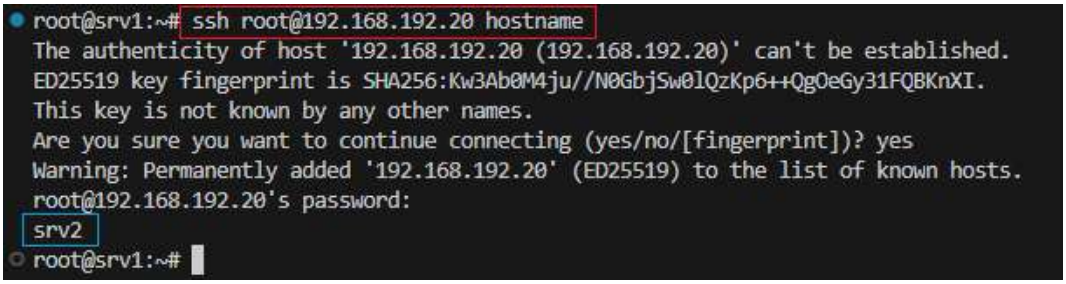
```
PermitRootLogin yes
UsePam yes
```

Setup Linux for Ansible Controller

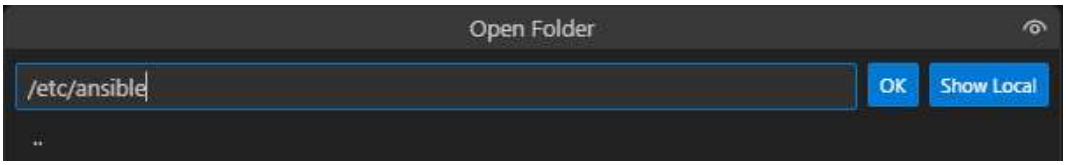
Lakukan langkah berikut pada workstation lalu buka vscode untuk mengakses srv1

Pada terminal yang ada di dalam vscode, install package `sshpass` terlebih dahulu menggunakan perintah:

```
apt install sshpass
```



Langkah selanjutnya adalah membuat ansible configuration yang disimpan pada file /etc/ansible/ansible.cfg . Pada vscode, tekan **Ctrl + K + O** lalu buka direktori /etc/ansible



Buat file ansible.cfg dengan konten seperti berikut:

```
[defaults]
inventory = /etc/ansible/inventory
host_key_checking = false
```

- inventory digunakan untuk menentukan file ansible inventory yang digunakan
- host_key_checking digunakan untuk menentukan apakah validasi host key akan dilakukan saat mengakses managed node menggunakan SSH Selanjutnya adalah membuat ansible inventory /etc/ansible/inventory dengan konten seperti berikut:

```
[linux]
srv2   ansible_host=192.168.192.20 ansible_user=root ansib
```

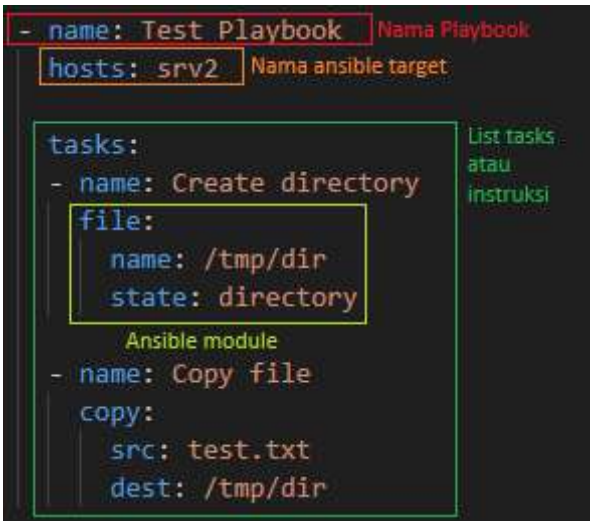
Konfigurasi file inventory diatas berisikan informasi managed node bernama srv2 yang memiliki IP address 192.168.192.20. Lalu, user root dengan password P@ssw0rd digunakan untuk mengakses server tersebut. Pada bagian paling atas, tulisan [linux] menjelaskan bahwa srv2 adalah anggota dari grup linux.
Untuk mengetahui lebih lanjut terkait pembuatan ansible inventory, kunjungi link berikut
https://docs.ansible.com/ansible/latest/getting_started/get_started_inventory.html

Terakhir, pastikan ansible bisa mengakses managed node menggunakan perintah ansible srv2 -m ping



Example Playbook

Ansible Playbook adalah file yang digunakan untuk menuliskan instruksi atau tasks yang untuk automation menggunakan ansible. Ansible Playbook memiliki struktur seperti berikut:



- name pada baris paling atas mendefinisikan nama playbook.
- hosts pada baris kedua mendefinisikan ansible target.
- tasks adalah block yang berisikan instruksi. Pada contoh diatas, terdapat 2 task dan setiap task akan menggunakan ansible module.

Ansible Module adalah modul atau plugin yang dapat digunakan pada Ansible Playbook. Setiap Ansible Module memiliki fungsinya masing - masing. Contohnya ansible module file yang digunakan untuk manipulasi direktori (create, update, delete). Untuk mengetahui ansible module yang tersedia, gunakan perintah ansible-doc -l
Untuk mengetahui lebih lanjut tentang penggunaan suatu ansible module, gunakan perintah:

```
ansible-doc <NAMA_MODULE>
```

Contoh playbook : Install NGINX dan Salin index.html

Buat file baru nginx-playbook.yaml

```
- name: Install NGINX dan Salin index.html ke direktori wel
  hosts: linux
  become: yes
  tasks:
    - name: Install NGINX
      apt:
        name: nginx
        state: present

    - name: Salin file index.html ke /var/www/html/index.ht
      copy:
        src: /etc/ansible/index.html
        dest: /var/www/html/index.html
        owner: www-data
        group: www-data
        mode: '0644'
```

Buat file /etc/ansible/index.html

```
nano /etc/ansible/index.html
```

```
srv2 nginx di konfigurasi menggunakan ansible
```

jalankan playbook nya

```
ansible-playbook nginx-playbook.yaml
```

Cek content nginx apakah sudah berubah

```
curl srv2.lab.local
```