Prepare Your System

Preparing Openemail Docker Host

Before you run **openemail**, there are a few requirements that you should check:

> **Warning**
>
> When running openemail on a Debian 8 (Jessie) box, you should switch to kernel 4.9 from Jessie backports to avoid a bug when running Docker containers with *healthchecks*! For more details read: github.com/docker/docker/issues/30402

> **Info**
>
> - openemail requires some ports to be open for incoming connections, so make sure that your firewall is not blocking these port
> - Make sure that no other application is interfering with openemail's configuration, such as another mail service
> - A correct DNS setup is crucial to every good mailserver setup, so please make sure you got at least the basics DNS setup covered before you begin!
> - Make sure that your system has a correct date and time setup. This is crucial for stuff like two factor TOTP authentication.

Choosing a Linux Distribution for the Docker Host

During the creation of this installation guide I used Ubuntu 18.04.2 LTS (Bionic Beaver). Some of the examples shown in this document may largely depend on this Linux distribution. But with a minor adjustment on it you can make it prepare to get working in your Linux distribution of choice as your docker host. It may be Ubuntu, Debian, CentOS, or an another.

Minimum System Resources

Please make sure that your system has at least the following resources:

Resource	Openemail
CPU	1 GHz
RAM	2 GiB + Swap (better: 4 GiB + Swap)
Disk	5 GiB (without emails)
System Type	x86_64

ClamAV and Solr are greedy RAM munchers. You can disable them in `openemail.conf` by settings SKIP_CLAMD=y and SKIP_SOLR=y.

Firewall and Ports

Please check if any of openemail's standard ports are open and not in use by other applications:

```
# netstat -tulpn | grep -E -w '25|80|110|143|443|465|587|993|995'
```

Warning

There are several problems with running Openemail on a firewalld/ufw enabled system. You should disable it (if possible) and move your ruleset to the DOCKER-USER chain, which is not cleared by a Docker service restart, instead. See this blog post for information about how to use iptables-persistent with the DOCKER-USER chain. As openemail runs dockerized, INPUT rules have no effect on restricting access to openemail. Use the FORWARD chain instead.

If this command returns any results please remove or stop the application running on that port. You may also adjust openemail ports via the openemail.conf` configuration file.

Openemail Default Ports

If you have a firewall in front of openemail docker host, please make sure that these ports are open for incoming connections:

Service	Protocol	Port	Container	Variable
Postfix SMTP	TCP	25	postfix-openemail	`${SMTP_PORT}`
Postfix SMTPS	TCP	465	postfix-openemail	`${SMTPS_PORT}`
Postfix Submission	TCP	587	postfix-openemail	`${SUBMISSION_PORT}`
Dovecot IMAP	TCP	143	dovecot-openemail	`${IMAP_PORT}`
Dovecot IMAPS	TCP	993	dovecot-openemail	`${IMAPS_PORT}`
Dovecot POP3	TCP	110	dovecot-openemail	`${POP_PORT}`
Dovecot POP3S	TCP	995	dovecot-openemail	`${POPS_PORT}`
Dovecot ManageSieve	TCP	4190	dovecot-openemail	`${SIEVE_PORT}`
HTTP(S)	TCP	80/443	nginx-openemail	`${HTTP_PORT}` / `${HTTPS_PORT}`

To bind a service to an IP address, you can prepend the IP like this: `SMTP_PORT=1.2.3.4:25`

Important: You cannot use IP:PORT bindings in HTTP_PORT and HTTPS_PORT. Please use `HTTP_PORT=1234` and `HTTP_BIND=1.2.3.4` instead.

Setting Date and Time

You need to ensure that date and time is accurate. This is required for the operation of openemail as well and accurate system logging.

To check your current time run:

```
$ timedatectl status
```

You will get an output like below

```
                   Local time: Tue 2019-03-05 09:17:54 UTC
               Universal time: Tue 2019-03-05 09:17:54 UTC
                     RTC time: Tue 2019-03-05 09:17:55
                    Time zone: Etc/UTC (UTC, +0000)
      System clock synchronized: yes
                   Local time: Tue 2019-03-05 09:17:54 UTC
               Universal time: Tue 2019-03-05 09:17:54 UTC
                     RTC time: Tue 2019-03-05 09:17:55
```

```
                      Time zone: Etc/UTC (UTC, +0000)
       System clock synchronized: yes
systemd-timesyncd.service active: no
                 RTC in local TZ: no
                 RTC in local TZ: no
```

The value `systemd-timesyncd.service active: no` means your system hasn't been configured get it time synced using `systemd-timesyncd`

To set your system time to sync with `systemd-timesyncd, run:

```
$ sudo sudo timedatectl set-ntp on
```

Now run again run `timedatectl status`. You will observe that `systemd-timesyncd.service active: yes`

```
$ timedatectl status
```

You will see this time `systemd-timesyncd.service active: yes` as in the below output

```
                    Local time: Tue 2019-03-05 09:26:24 UTC
                Universal time: Tue 2019-03-05 09:26:24 UTC
                      RTC time: Tue 2019-03-05 09:26:25
                     Time zone: Etc/UTC (UTC, +0000)
       System clock synchronized: yes
systemd-timesyncd.service active: yes
```

To check to see whether `systemd-timesyncd` is running:

```
sudo systemctl status systemd-timesyncd
```

If it is not running you will get an output like below.

```
systemctl status systemd-timesyncd
● systemd-timesyncd.service - Network Time Synchronization
   Loaded: loaded (/lib/systemd/system/systemd-timesyncd.service; enabled; vendor
preset: enabled)
   Active: inactive (dead) since Wed 2019-03-06 03:18:40 UTC; 2s ago
     Docs: man:systemd-timesyncd.service(8)
  Process: 21020 ExecStart=/lib/systemd/systemd-timesyncd (code=exited, status=0/
SUCCESS)
 Main PID: 21020 (code=exited, status=0/SUCCESS)
   Status: "Idle."

Mar 05 09:26:13 mail.openemail.io systemd[1]: Starting Network Time
Synchronization...
Mar 05 09:26:13 mail.openemail.io systemd[1]: Started Network Time
Synchronization.
```

```
Mar 05 09:26:13 mail.openemail.io systemd-timesyncd[21020]: Synchronized to time
server 91.189.89.198:123 (ntp.ubuntu.com).
Mar 06 03:18:40 mail.openemail.io systemd[1]: Stopping Network Time
Synchronization...
Mar 06 03:18:40 mail.openemail.io systemd[1]: Stopped Network Time
Synchronization.
```

To start `systemd-timesyncd` run:

```
sudo systemctl start systemd-timesyncd
```

To see the status of `systemd-timesyncd` run:

```
sudo systemctl status systemd-timesyncd
```

Your output should be like below

```
● systemd-timesyncd.service - Network Time Synchronization
   Loaded: loaded (/lib/systemd/system/systemd-timesyncd.service; enabled; vendor
preset: enabled)
   Active: active (running) since Wed 2019-03-06 03:30:08 UTC; 8s ago
     Docs: man:systemd-timesyncd.service(8)
 Main PID: 29018 (systemd-timesyn)
   Status: "Synchronized to time server 91.189.89.198:123 (ntp.ubuntu.com)."
    Tasks: 2 (limit: 2361)
   CGroup: /system.slice/systemd-timesyncd.service
           └─29018 /lib/systemd/systemd-timesyncd

Mar 06 03:30:08 mail.openemail.io systemd[1]: Starting Network Time
Synchronization...
Mar 06 03:30:08 mail.openemail.io systemd[1]: Started Network Time
Synchronization.
Mar 06 03:30:08 mail.openemail.io systemd-timesyncd[29018]: Synchronized to time
server 91.189.89.198:123 (ntp.ubuntu.com).
```

Set Up Static IP

Ubuntu 18.04 has changed its network interface configuration subsystem with new netplan configuration. The yml syntaxes are used in network configuration.

To configure the network

```
sudo nano  /etc/netplan/50-cloud-init.yaml
```

Below is an example configuration. Adjust settings as per your network environment.

```
network:
    version: 2
    ethernets:
        eth0:
            addresses:
            - 178.128.57.210/20
            gateway4: 178.128.48.1
            match:
                macaddress: 0a:17:6b:4f:06:ed
            nameservers:
                addresses:
                - 67.207.67.2
                - 67.207.67.3
                search: []
            set-name: eth0
```

To update your settings run:

```
sudo netplan apply
```

To check your network configuration run

```
ip addr sh eth0
```

You will get an output like Below

```
eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group
default qlen 1000
   link/ether 0a:17:6b:4f:06:ed brd ff:ff:ff:ff:ff:ff
   inet 178.128.57.210/20 brd 178.128.63.255 scope global eth0
      valid_lft forever preferred_lft forever
   inet6 fe80::817:6bff:fe4f:6ed/64 scope link
      valid_lft forever preferred_lft forever
```

Setting MTU

If you are running Openemail in OpenStack environment: Check your MTU and set it accordingly in docker-compose.yml. See **4.1** in our installation docs.