

Ghid detaliat pentru Configurarea Mașinii Virtuale

Acest ghid descrie pas cu pas procesul de configurare a unei mașini virtuale pentru rularea unei aplicații web cu NGINX, PHP, MySQL și un backend Node.js. Ghidul include și recomandări de securitate, administrare și bune practici.

1. Actualizarea sistemului

Primul pas este să ne asigurăm că sistemul de operare este actualizat la zi. Aceasta reduce vulnerabilitățile de securitate și garantează că toate pachetele folosite sunt stabile.

Comandă:

```
dnf update -y
```

2. Crearea unui utilizator administrator

Se recomandă evitarea utilizării directe a utilizatorului root. În schimb, se creează un utilizator obișnuit, care primește drepturi administrative prin grupul wheel.

Comenzi:

```
useradd [nume]
```

```
passwd [nume]
```

```
usermod -aG wheel [nume]
```

3. Instalarea unui editor de text

Pentru editarea fișierelor de configurare, putem folosi editorul nano sau vim.

Exemplu cu nano:

```
dnf install nano -y
```

4. Configurarea firewall-ului

Firewall-ul este esențial pentru securitatea serverului. Se recomandă activarea serviciului firewalld și permiterea doar a porturilor necesare (80, 443, 3306).

Comenzi:

Trecerea in root:

```
$ sudo -i
```

Verificarea firewall-ului

```
# systemctl status firewalld
```

Definirea serviciului firewalld ca fiind enabled

```
# systemctl enable firewalld
```

Pornirea serviciului firewalld

```
# systemctl start firewalld
```

Oprirea serviciului firewalld

```
# systemctl stop firewalld
```

Deschiderea porturilor necesare funcționării serverului (80 și 443)

```
firewall-cmd --zone=public --permanent --add-service=http  
firewall-cmd --zone=public --permanent --add-service=https  
firewall-cmd --zone=public --permanent --add-service=mysql  
firewall-cmd --reload
```

5. Instalarea și configurarea MySQL

MySQL este sistemul de baze de date utilizat pentru aplicație.

Instalare și pornire:

```
dnf install mysql-server -y
```

```
systemctl start mysqld
```

```
systemctl enable mysqld
```

Securizare:

```
mysql_secure_installation (setare parolă root, eliminare useri anonimi, dezactivare  
login remote root)
```

Test conectare:

```
mysql -u root -p
```

6. Instalarea PHP 8.3

Se folosesc depozitele EPEL și Remi pentru a instala o versiune recentă de PHP.

```
dnf install epel-release -y  
dnf install -y https://rpms.remirepo.net/enterprise/remi-release-9.2.rpm  
dnf module install -y php:remi-8.3  
dnf install -y php php-mysqlnd php-cli php-fpm php-json php-common
```

Testare:

```
php -v
```

7. Instalarea și configurarea NGINX

NGINX este serverul web care va livra fișierele statice și va face reverse proxy către backend.

Instalare:

```
dnf install nginx git curl -y
```

Dacă Apache rulează, acesta trebuie oprit:

```
systemctl stop httpd  
systemctl disable httpd
```

Pornire și activare NGINX:

```
systemctl enable nginx  
systemctl start nginx
```

8. Instalarea Node.js și configurarea backend-ului cu PM2

Pentru a rula backend-ul Node.js se folosește PM2 (manager de procese).

```
cd /var/www/api  
npm init -y  
npm install express cors  
npm install pm2@latest -g  
pm2 start server.js --name my-api
```

Creare în /var/www/api/ unui fisier denumit server.js care:

- ascultă pe portul 3000
- răspunde la /api/

- returnează JSON de test (poți adapta după nevoie)
- gestionează CORS (dacă frontend-ul și backend-ul sunt pe același domeniu, nu va fi nevoie de setări complicate)

server.js :

```
// Importă modulele necesare
const express = require('express');
const path = require('path');
const cors = require('cors');

const app = express();
const PORT = process.env.PORT || 3000;

// Middleware pentru JSON și CORS
app.use(express.json());
app.use(cors());

// Exemplu endpoint API
app.get('/api', (req, res) => {
  res.json({ status: 'success', message: 'API-ul Express funcționează!' });
});

// Dacă ai endpoint-uri suplimentare:
/*
app.post('/api/data', (req, res) => {
  const received = req.body;
  res.json({ status: 'ok', received });
});
*/

// Pornirea serverului
app.listen(PORT, () => {
  console.log(`Server Express pornit pe portul ${PORT}`);
});
```

pm2 save

pm2 startup system

9. Configurarea NGINX pentru frontend și backend

Presupunem că:

- fișierele statice (HTML, CSS, JS, assets) sunt în /var/www/html
- backend-ul Express rulează pe portul **3000**

Se creează un fișier de configurare în /etc/nginx/conf.d/myapp.conf cu următorul conținut:

```
# === Domeniul pe HTTPS ===
server {
    server_name smartgreenhouse.online www.smartgreenhouse.online;

    root /var/www/html;
    index index.html;

    location /phpmyadmin/ {
        root /var/www/html;
        index index.php index.html index.htm;
    }

    location ~ \.php$ {
        include fastcgi_params;
        fastcgi_pass unix:/run/php-fpm/www.sock;
        fastcgi_index index.php;
        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
    }

    location / {
        # încearcă fișier static (ex: index.html)
        try_files $uri $uri/ @backend;
    }
}

# === Backend Node pe radacina ===
location @backend {
    proxy_pass http://127.0.0.1:5000;
    proxy_http_version 1.1;
    proxy_set_header Host $host;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
```

```
proxy_set_header X-Forwarded-Proto $scheme;
proxy_set_header Upgrade $http_upgrade;
proxy_set_header Connection "upgrade";
proxy_read_timeout 60s;
}

listen 443 ssl; # managed by Certbot
ssl_certificate /etc/letsencrypt/live/smartgreenhouse.online/fullchain.pem; #
managed by Certbot
ssl_certificate_key /etc/letsencrypt/live/smartgreenhouse.online/privkey.pem; #
managed by Certbot
include /etc/letsencrypt/options-ssl-nginx.conf; # managed by Certbot
ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by Certbot
}

# === Domeniul pe HTTP ? redirect la HTTPS ===
server {
    listen 80;
    server_name smartgreenhouse.online www.smartgreenhouse.online;

    return 301 https://$host$request_uri;
}

# === Acces prin IP (fara redirect) ===
server {
    listen 80;
    listen 443 ssl;
    server_name 144.91.100.248;

    # dummy SSL pentru ca sa poata porni pe 443
    ssl_certificate /etc/ssl/certs/ssl-cert-snakeoil.pem;
    ssl_certificate_key /etc/ssl/private/ssl-cert-snakeoil.key;

    location / {
        try_files $uri $uri/ @backend;
    }

    location @backend {
        proxy_pass http://127.0.0.1:5000;
        proxy_http_version 1.1;
    }
}
```

```
proxy_set_header Host $host;
proxy_set_header X-Real-IP $remote_addr;
proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
proxy_set_header X-Forwarded-Proto $scheme;
proxy_set_header Upgrade $http_upgrade;
proxy_set_header Connection "upgrade";
proxy_read_timeout 60s;
}
}
```

Comenzi utile:

```
nginx -t (test sintaxă)
systemctl reload nginx
```

10. Configurarea DNS și SSL

În Namecheap se configurează A Record pentru domeniu și subdomeniu www, care să pointeze spre IP-ul public al serverului.

Pașii în Namecheap Dashboard

1. Loghează-te în contul tău Namecheap → <https://ap.www.namecheap.com>.
2. În meniul din stânga mergi la Domain List și apasă pe butonul Manage din dreptul domeniului cumpărat.
3. În tab-ul Domain găsești secțiunea Nameservers:
 - Dacă vrei să folosești DNS-ul implicit de la Namecheap → selectează Namecheap BasicDNS.
 - (Dacă folosești Cloudflare sau alți nameserveri, setările trebuie făcute acolo, nu în Namecheap).
4. Mergi la tab-ul Advanced DNS.
5. Aici trebuie să adaugi/editezi următoarele records:
 - A Record → pentru domeniu:
 - Host: @
 - Value: [IP-ul public al serverului tău]
 - TTL: Automatic (sau 30 min)

- A Record → pentru www:
- Host: www
- Value: [IP-ul public al serverului tău]
- TTL: Automatic

Rezultatul final ar trebui să arate aşa:

| Type | | Host | | Value | | TTL |
|---|--|------|--|--------------|--|-----------|
| ----- ----- ----- ----- ----- ----- ----- | | | | | | |
| A | | @ | | 123.45.67.89 | | Automatic |
| A | | www | | 123.45.67.89 | | Automatic |

6. Apasă pe Save All Changes (foarte important, altfel nu se aplică).

Pentru securizare, se instalează Certbot și se configurează SSL automat:

```
dnf install certbot python3-certbot-nginx -y
certbot --nginx -d domeniu.ro -d www.domeniu.ro
```

Se recomandă configurarea auto-renew:

```
systemctl list-timers | grep certbot
```

11. Recomandări suplimentare de securitate și bune practici

- Folosește autentificare cu chei SSH în loc de parolă.
- Monitorizează logurile cu journalctl și fail2ban.
- Creează backup-uri regulate pentru bazele de date și fișiere.
- Folosește un serviciu de monitorizare uptime (ex: UptimeRobot).
- Configurează un serviciu de alertare prin email în caz de erori majore.