Nginx

Nginx - Background

Nginx merupakan sebuah perangkat lunak server yang sangat terkenal dalam dunia komputasi server. Diciptakan pertama kali oleh Igor Sysoev pada tahun 2004, Nginx awalnya dikembangkan untuk mengatasi masalah skala besar dalam penanganan koneksi bersamaan yang tinggi pada server web tradisional seperti Apache. Nama "Nginx" sendiri merupakan singkatan dari "Engine-X", yang mencerminkan fokusnya pada performa dan kecepatan.

Sejak diluncurkan, Nginx telah mengalami pertumbuhan yang pesat dan mendapatkan reputasi sebagai salah satu server web terkemuka di dunia. Hal ini terutama disebabkan oleh beberapa keunggulan teknisnya:

1. Performa Tinggi : Nginx didesain untuk menjadi ringan dan efisien dalam penggunaan sumber daya, sehingga mampu menangani banyak koneksi secara simultan dengan baik tanpa mengorbankan kecepatan.
2. Skalabilitas : Kemampuannya dalam menangani beban lalu lintas yang tinggi membuatnya sangat cocok untuk aplikasi dan situs web yang memerlukan skalabilitas horizontal.

3. Proxying dan Load Balancing : Nginx tidak hanya berfungsi sebagai server web, tetapi juga dapat berperan sebagai reverse proxy dan load balancer. Ini memungkinkan Nginx untuk mendistribusikan lalu lintas web ke beberapa server backend, meningkatkan efisiensi dan keandalan aplikasi.

4. Caching : Nginx memiliki fitur caching yang kuat, memungkinkan penyimpanan sementara konten statis seperti gambar, CSS, dan JavaScript. Hal ini tidak hanya mempercepat waktu respon, tetapi juga mengurangi beban pada server backend.

5. Konfigurasi yang Fleksibel : Konfigurasi Nginx menggunakan file teks yang mudah dimengerti, memungkinkan administrator sistem untuk menyesuaikan pengaturan server sesuai kebutuhan spesifik aplikasi atau lingkungan operasional.

Nginx telah menjadi pilihan utama untuk banyak organisasi besar dan start-up teknologi karena kombinasi fitur-fitur canggihnya, performa yang dapat diandalkan, serta kemudahan dalam konfigurasi dan skalabilitasnya. Dengan komunitas pengguna yang besar dan aktif, Nginx terus mengalami perkembangan dan pembaruan, menjaga posisinya sebagai salah satu solusi terbaik dalam infrastruktur web modern.

Nginx - Install Nginx Ubuntu

Perbarui Indeks Repository

```
apt update && apt upgrade -y
```

Syarat Instalasi

```
sudo apt install curl gnupg2 ca-certificates lsb-release ubuntu-keyring -y
```

Import official nginx signing key

```
curl https://nginx.org/keys/nginx_signing.key | gpg --dearmor \
    | sudo tee /usr/share/keyrings/nginx-archive-keyring.gpg >/dev/null
```

Verifikasi Key

```
gpg --dry-run --quiet --no-keyring --import --import-options import-show /usr/share/keyrings/nginx-archive-keyring.
```

Output Fingerprint yang dihasilkan

```
pub   rsa2048 2011-08-19 [SC] [expires: 2024-06-14]
      573BFD6B3D8FBC641079A6ABABF5BD827BD9BF62
uid                 nginx signing key <signing-key@nginx.com>
```

Note : **Jika output fingerprint yang dihasilkan berbeda, hapus file tersebut**

Untuk instalasi paket Nginx yang stabil (Stable), Jalankan perintah dibawah ini

```
echo "deb [signed-by=/usr/share/keyrings/nginx-archive-keyring.gpg] \
http://nginx.org/packages/ubuntu `lsb_release -cs` nginx" \
    | sudo tee /etc/apt/sources.list.d/nginx.list
```

Jika ingin menggunakan paket Nginx Mainline, Jalankan perintah dibawah ini

```
echo "deb [signed-by=/usr/share/keyrings/nginx-archive-keyring.gpg] \
http://nginx.org/packages/mainline/ubuntu `lsb_release -cs` nginx" \
    | sudo tee /etc/apt/sources.list.d/nginx.list
```

Jalankan juga perintah dibawah ini

```
echo -e "Package: *\nPin: origin nginx.org\nPin: release o=nginx\nPin-Priority: 900\n" \
    | sudo tee /etc/apt/preferences.d/99nginx
```

Untuk instalasi nginx jalankan perintah dibawah ini

```
sudo apt update
sudo apt install nginx
```

Cek versi Nginx menggunakan perintah dibawah ini

```
nginx -version
```

Nginx - Web Service

1. Buat file konfigurasi baru pada direktori conf.d :

```
sudo nano /etc/nginx/conf.d/kevin.conf
```

2. Tambahkan konfigurasi dasar pada file tersebut :

```
server {
    listen 80;
    server_name kevin.com;
    root /var/www/kevin;
    index index.html;
}
```

3. Konfigurasi di atas membuat Nginx untuk :

- Menjalankan server block pada port 80
- Mengarahkan semua request dengan server_name example.com ke direktori /var/www/kevin
- Menggunakan index file index.html jika ada

Optional :

4. Tambahkan direktif tambahan pada server block untuk meningkatkan performa atau fitur tambahan:

- Mematikan server signature :

```
server_tokens off;
```

- Menyediakan akses log :

```
access_log /var/log/nginx/kevin_access.log;
```

- Mengaktifkan kompresi gzip :

```
gzip on;
gzip_types text/plain text/css application/json application/javascript text/xml application/xml application/xml+rss
text/javascript;
```

- Mengatur cache untuk static file :

```
location /static/ {
    expires 7d;
    add_header Cache-Control "public, max-age=604800, immutable";
}
```

- Mengatur reverse proxy untuk mengakses aplikasi lain :

```
location /api/ {
    proxy_pass http://localhost:8000/;
    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;
}
```

- Restart Nginx untuk mengaktifkan konfigurasi :

```
sudo systemctl restart nginx
```

Konfigurasi di atas dapat diubah sesuai kebutuhan dan dapat digunakan sebagai dasar untuk mengatur server lain di direktori conf.d. Pastikan untuk melakukan testing setiap kali melakukan perubahan pada konfigurasi.

Nginx - Proxy Pass

Proxy pass di Nginx digunakan untuk mem-forward (meneruskan) permintaan (request) dari klien ke server backend, dan mengembalikan (return) respons dari server backend ke klien. Berikut adalah contoh penggunaan proxy pass di Nginx :

```
server {
    listen 80;
    server_name kevin.ilusidigital.com;

    location / {
        proxy_pass http://192.168.1.110:8080;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    }
}
```

Penjelasan dari konfigurasi tersebut :

- `listen 80;` mendefinisikan port yang akan didengarkan oleh server.
- `server_name kevin.ilusidigital.com;` mendefinisikan nama domain yang akan diproses oleh server.
- `location /` mendefinisikan lokasi dari permintaan yang akan diteruskan ke server backend.
- `proxy_pass http://backend_server:8080;` mendefinisikan alamat server backend yang akan diproses permintaan dari klien. Perhatikan bahwa protokol http harus didefinisikan dan port 8080 digunakan di sini.
- `proxy_set_header` mendefinisikan header tambahan yang akan disertakan dalam permintaan yang diteruskan. `Host` header digunakan untuk menentukan host tujuan, `X-Real-IP` digunakan untuk mendapatkan alamat IP asli dari klien, dan `X-Forwarded-For` digunakan untuk memberikan informasi tambahan tentang asal permintaan.

Setelah Anda mengonfigurasi proxy pass di Nginx, pastikan untuk menguji koneksi ke server backend dengan menjalankan perintah berikut :

```
curl http://localhost
```

Jika server backend merespons dengan benar, maka konfigurasi proxy pass di Nginx sudah berhasil.

Nginx - Load Balancer

Berikut adalah langkah-langkah konfigurasi load balancer Nginx di file `/etc/nginx/conf.d/loadbalancer.conf`:

- Buat file baru dengan perintah :

```
sudo nano /etc/nginx/conf.d/loadbalancer.conf
```

- Masukkan konfigurasi berikut :

```
upstream backend {
    server 10.0.0.1:80;
    server 10.0.0.2:80;
}

server {
    listen 80;
    server_name kevin.ilusidigital.com;
    location / {
        proxy_pass http://backend;
        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;
    }
}
```

- Konfigurasi ini mendefinisikan :

1. Upstream yang terdiri dari 2 server pada alamat IP 10.0.0.1 dan 10.0.0.2 pada port 80
2. Server yang mendengarkan pada port 80 dan server name kevin.cinema21.co.id
3. Lokasi root di-proxy ke backend, disertai dengan pengaturan header proxy

Pastikan Anda mengganti `kevin.ilusidigital.com` dengan nama domain Anda dan mengganti `backend` dengan nama upstream Anda yang diinginkan. Setelah selesai, simpan dan keluar dari editor dengan menekan `Ctrl+X`, `Y`, dan `Enter`.

- Verifikasi file konfigurasi dengan perintah :

```
sudo nginx -t
```

- Jika tidak ada error, restart Nginx dengan perintah :

```
sudo systemctl restart nginx
```

Load balancer Nginx sekarang sudah berjalan dan terhubung ke backend pada alamat IP yang telah didefinisikan. Anda dapat memeriksa dengan membuka browser dan mengunjungi nama domain atau alamat IP server Nginx.

Nginx - Amplify

Nginx Amplify

- Buat akun Nginx Amplify di https://amplify.nginx.com/signup/
- Setelah itu login ke akun tersebut
- Masuk ke server Nginx dan Buat config di **/etc/nginx/conf.d/**
- Buat file dengan nama **stub_status.conf**
- Copy config di bawah ini

```
server {
   listen 127.0.0.1:8080;
   server_name localhost;
   location /nginx_status {
      stub_status on;
      access_log off;
      allow 127.0.0.1;
      deny all;
   }
}
```

- Setelah itu restart / reload nginx
- Setelah itu copy script di bawah ini

```
curl -sS -L -O \
https://github.com/nginxinc/nginx-amplify-agent/raw/master/packages/install.sh && \
API_KEY='YOUR_API_KEY' sh ./install.sh
```

Note : **YOUR_API_KEY adalah API KEY yang didapat dari akun Nginx Amplify**

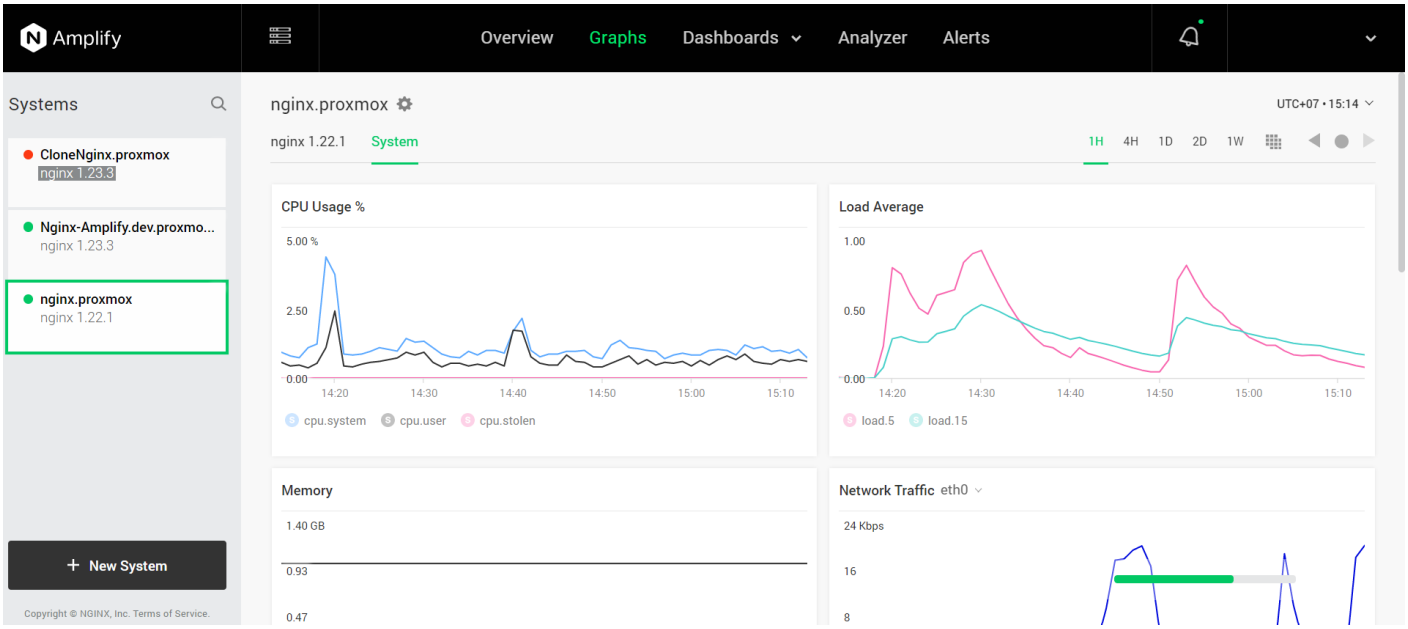
- Setelah selesai jalankan agent dari amplify

```
service amplify-agent start
```

- Setelah itu masuk ke lihat di akun nginx amplify kemudian klik continue

image.png
Image not found or type unknown

- Berikut adalah tampilan dashbord jika konfigurasi berhasil


Amplify
Overview
Graphs
Dashboards
Analyzer
Alerts
Systems
CloneNginx.proxmox
nginx 1.23.3
Nginx-Amplify.dev.proxmo...
nginx 1.23.3
nginx.proxmox
nginx 1.22.1
+ New System
Copyright © NGINX, Inc. Terms of Service.
nginx.proxmox
UTC+07 • 15:14
nginx 1.22.1
System
1H
4H
1D
2D
1W
CPU Usage %
5.00 %
2.50
0.00
14:20
14:30
14:40
14:50
15:00
15:10
cpu.system
cpu.user
cpu.stolen
Load Average
1.00
0.50
0.00
14:20
14:30
14:40
14:50
15:00
15:10
load.5
load.15
Memory
1.40 GB
0.93
0.47
Network Traffic
eth0
24 Kbps
16
8

Nginx - Anti DDos

- Aktifkan module "limit_req" dan "limit_conn" di konfigurasi Nginx :

```
http {
    # Aktifkan limit_req module untuk membatasi jumlah request dari satu alamat IP
    limit_req_zone $binary_remote_addr zone=req_limit_per_ip:10m rate=5r/s;

    # Aktifkan limit_conn module untuk membatasi jumlah koneksi dari satu alamat IP
    limit_conn_zone $binary_remote_addr zone=conn_limit_per_ip:10m;
}
```

- Batasi jumlah request dari satu alamat IP :

```
server {
    listen 80;
    server_name kevin.ilusidigital.com;

    location / {
        # Batasi jumlah request dari satu alamat IP menjadi 10 request per detik
        limit_req zone=req_limit_per_ip burst=15 nodelay;

        # Konfigurasi lainnya...
    }
}
```

- Batasi jumlah koneksi dari satu alamat IP :

```
server {
    listen 80;
    server_name kevin.ilusidigital.com;

    location / {
        # Batasi jumlah koneksi dari satu alamat IP menjadi 10 koneksi
        limit_conn conn_limit_per_ip 10;

        # Konfigurasi lainnya...
    }
```

```
}
```

- Aktifkan proteksi dari serangan SYN flood :

```
server {
    listen 80;
    server_name kevin.ilusidigital.com;

    location / {
        # Aktifkan proteksi dari serangan SYN flood
        limit_conn_zone $binary_remote_addr zone=syn_flood:10m;
        limit_conn syn_flood 10;

        # Konfigurasi lainnya...
    }
}
```

- Aktifkan proteksi dari serangan UDP flood :

```
stream {
    # Aktifkan proteksi dari serangan UDP flood
    limit_conn_zone $binary_remote_addr zone=udp_flood:10m;
    limit_conn udp_flood 10;

    # Konfigurasi lainnya...
}
```

- Aktifkan proteksi dari serangan HTTP flood :

```
server {
    listen 80;
    server_name kevin.ilusidigital.com;

    location / {
        # Aktifkan proteksi dari serangan HTTP flood
        limit_req_zone $binary_remote_addr zone=http_flood:10m rate=100r/s;
        limit_req zone=http_flood burst=200 nodelay;

        # Konfigurasi lainnya...
    }
}
```

Dengan konfigurasi ini, Anda dapat membatasi jumlah request, koneksi, dan mengaktifkan proteksi dari serangan DDoS di server Nginx Anda. Namun, perlu diingat bahwa konfigurasi ini hanya sebagai contoh dan dapat disesuaikan dengan kebutuhan Anda.

Nginx - WAF

- Install Nginx and ModSecurity.

```
sudo apt-get update
sudo apt-get install nginx -y
sudo apt-get install libnginx-mod-security -y
sudo apt-get intall git -y
```

- Enable ModSecurity module in Nginx configuration.

```
sudo sed -i 's/# include \/etc\/nginx\/modules-enabled\/\*\.conf;/include \/etc\/nginx\/modules-enabled\/\*\.conf;/' /etc/nginx/nginx.conf
```

- Configure ModSecurity rules

```
sudo mv /etc/nginx/mods-available/mod-security.conf /etc/nginx/mods-available/mod-security.conf.orig
sudo cp /usr/share/modsecurity-crs/modsecurity.conf-recommended /etc/nginx/mods-available/mod-security.conf
sudo mv /etc/nginx/mods-available/modsecurity.conf /etc/nginx/mods-available/modsecurity.conf.orig
sudo sed -i 's/SecRuleEngine DetectionOnly/SecRuleEngine On/' /etc/nginx/mods-available/modsecurity.conf
sudo sed -i 's/SecResponseBodyAccess On/SecResponseBodyAccess Off/' /etc/nginx/mods-available/modsecurity.conf
```

- Download and configure the OWASP Core Rule Set.

```
sudo git clone https://github.com/SpiderLabs/owasp-modsecurity-crs.git /usr/share/modsecurity-crs
sudo cp /usr/share/modsecurity-crs/crs-setup.conf.example /usr/share/modsecurity-crs/crs-setup.conf
sudo mv /etc/nginx/mods-available/modsecurity-crs.conf /etc/nginx/mods-available/modsecurity-crs.conf.orig
sudo cp /usr/share/modsecurity-crs/rules/*.conf /etc/nginx/mods-available/
sudo cp /usr/share/modsecurity-crs/rules/*.data /etc/nginx/mods-available/
sudo cp /usr/share/modsecurity-crs/rules/*.txt /etc/nginx/mods-available/
sudo cp /usr/share/modsecurity-crs/rules/*.sls /etc/nginx/mods-available/
sudo sed -i 's/Include \/etc\/nginx\/modsecurity.conf/Include \/etc\/nginx\/mods-available\/modsecurity.conf/' /etc/nginx/mods-available/modsecurity-crs.conf
```

- Backup Config Nginx Default.

```
sudo cp /etc/nginx/sites-available/default /etc/nginx/sites-available/default.orig
```

- Download and configure the OWASP Core Rule Set.

```
server {
    listen 80;
    listen [::]:80;
    server_name example.com;

    location / {
        proxy_pass http://localhost:8000;
        proxy_set_header Host \$host;
        proxy_set_header X-Real-IP \$remote_addr;
        proxy_set_header X-Forwarded-For \$proxy_add_x_forwarded_for;

        # ModSecurity settings
        modsecurity on;
        modsecurity_rules_file /etc/nginx/mods-available/modsecurity-crs.conf;
    }
}
```

- Restart Nginx to apply changes

```
sudo systemctl restart nginx
```

Nginx - All Prevention

Prevention secara singkat berarti **pencegahan** atau **tindakan untuk menghindari suatu risiko atau ancaman sebelum terjadi**.

Dalam konteks keamanan, **prevention** bertujuan untuk **melindungi sistem dari serangan, eksploitasi, atau penyalahgunaan** dengan menerapkan aturan atau mekanisme pengamanan

Berikut Konfigurasinya :

```
############################ XSS PREVENTION
##############################
        set $block_xss 0;
if ($query_string ~ "base64_(en|de)code\(.*\)") {
set $block_xss 1;
}
if ($request_uri ~ "base64_(en|de)code\(.*\)") {
        set $block_xss 1;
}
if ($query_string ~ "(<|%3C).*script.*(>|%3E)") {
        set $block_xss 1;
}
if ($request_uri ~ "(<|%3C).*script.*(>|%3E)") {
        set $block_xss 1;
}
if ($query_string ~ "(<|%3C).*iframe.*(>|%3E)") {
        set $block_xss 1;
}
if ($request_uri ~ "(<|%3C).*iframe.*(>|%3E)") {
        set $block_xss 1;
}
if ($query_string ~ "GLOBALS(=|\[|\%[0-9A-Z]{0,2})") {
        set $block_xss 1;
}
if ($query_string ~ "_REQUEST(=|\[|\%[0-9A-Z]{0,2})") {
        set $block_xss 1;
}
if ($block_xss = 1) {
```

```
        return 403;
}

## Block SQL injections
    set $block_sql_injections 0;
    if ($query_string ~ "union.*select.*\(") {
        set $block_sql_injections 1;
    }
    if ($query_string ~ "union.*all.*select.*") {
        set $block_sql_injections 1;
    }
    if ($query_string ~ "concat.*\(") {
        set $block_sql_injections 1;
    }
    if ($block_sql_injections = 1) {
        return 403;
    }

    ## Block file injections
    set $block_file_injections 0;
    if ($query_string ~ "[a-zA-Z0-9_]=http://") {
        set $block_file_injections 1;
    }
    if ($query_string ~ "[a-zA-Z0-9_]=(\.\.//?)+") {
        set $block_file_injections 1;
    }
    if ($query_string ~ "[a-zA-Z0-9_]=/([a-z0-9_.]//?)+") {
        set $block_file_injections 1;
    }
    if ($block_file_injections = 1) {
        return 403;
    }

    ## Block common exploits
    set $block_common_exploits 0;
    if ($query_string ~ "(<|%3C).*script.*(>|%3E)") {
        set $block_common_exploits 1;
    }
    if ($query_string ~ "GLOBALS(=|\[|\%[0-9A-Z]{0,2})") {
        set $block_common_exploits 1;
```

```
}
if ($query_string ~ "_REQUEST(=|\[|\%[0-9A-Z]{0,2})") {
    set $block_common_exploits 1;
}
if ($query_string ~ "proc/self/environ") {
    set $block_common_exploits 1;
}
if ($query_string ~ "mosConfig_[a-zA-Z_]{1,21}(=|\%3D)") {
    set $block_common_exploits 1;
}
if ($query_string ~ "base64_(en|de)code\(.*\)") {
    set $block_common_exploits 1;
}
if ($block_common_exploits = 1) {
    return 403;
}

## Block spam
set $block_spam 0;
if ($query_string ~ "\b(ultram|unicauca|valium|viagra|vicodin|xanax|ypxaieo)\b") {
    set $block_spam 1;
}
if ($query_string ~ "\b(erections|hoodia|huronriveracres|impotence|levitra|libido)\b") {
    set $block_spam 1;
}
if ($query_string ~ "\b(ambien|blue\spill|cialis|cocaine|ejaculation|erectile)\b") {
    set $block_spam 1;
}
if ($query_string ~ "\b(lipitor|phentermin|pro[sz]ac|sandyauer|tramadol|troyhamby)\b") {
    set $block_spam 1;
}
if ($block_spam = 1) {
    return 403;
}

## Block user agents
#SET $BLOCK_USER_AGENTS 0;

# Don't disable wget if you need it to run cron jobs!
#if ($http_user_agent ~ "Wget") {
```

```
#    set $block_user_agents 1;
#}

# Disable Akeeba Remote Control 2.5 and earlier
if ($http_user_agent ~ "Indy Library") {
    set $block_user_agents 1;
}

# Common bandwidth hoggers and hacking tools.
if ($http_user_agent ~ "libwww-perl") {
    set $block_user_agents 1;
}
if ($http_user_agent ~ "GetRight") {
    set $block_user_agents 1;
}
if ($http_user_agent ~ "GetWeb!") {
    set $block_user_agents 1;
}
if ($http_user_agent ~ "Go!Zilla") {
    set $block_user_agents 1;
}
if ($http_user_agent ~ "Download Demon") {
    set $block_user_agents 1;
}
if ($http_user_agent ~ "Go-Ahead-Got-It") {
    set $block_user_agents 1;
}
if ($http_user_agent ~ "TurnitinBot") {
    set $block_user_agents 1;
}
if ($http_user_agent ~ "GrabNet") {
    set $block_user_agents 1;
}

if ($block_user_agents = 1) {
    return 403;
}
```

Nginx - XSS Prevention

XSS (Cross-Site Scripting) Prevention adalah langkah-langkah untuk mencegah serangan di mana penyerang menyisipkan skrip berbahaya ke dalam halaman web yang kemudian dieksekusi di browser pengguna.

Berikut Konfigurasinya :

```
set $block_xss 0;


if ($query_string ~ "base64_(en|de)code\(.*\)") {
    set $block_xss 1;
}
if ($request_uri ~ "base64_(en|de)code\(.*\)") {
    set $block_xss 1;
}
if ($query_string ~ "(<|%3C).*script.*(>|%3E)") {
    set $block_xss 1;
}
if ($request_uri ~ "(<|%3C).*script.*(>|%3E)") {
    set $block_xss 1;
}
if ($query_string ~ "(<|%3C).*iframe.*(>|%3E)") {
    set $block_xss 1;
}
if ($request_uri ~ "(<|%3C).*iframe.*(>|%3E)") {
    set $block_xss 1;
}
if ($block_xss = 1) {
    return 403;
}
```

Nginx - SQL Injection Prevention

SQL Injection Prevention adalah langkah-langkah untuk mencegah serangan di mana penyerang menyisipkan SQL berbahaya ke dalam query database.

Berikut Konfigurasinya :

```
set $block_sql_injections 0;

if ($query_string ~ "union.*select.*\(") {
    set $block_sql_injections 1;
}
if ($query_string ~ "union.*all.*select.*") {
    set $block_sql_injections 1;
}
if ($query_string ~ "concat.*\(") {
    set $block_sql_injections 1;
}
if ($block_sql_injections = 1) {
    return 403;
}
```

Nginx - File Injection Prevention

File Injection Prevention adalah teknik keamanan yang digunakan untuk **mencegah penyisipan file berbahaya** ke dalam sistem melalui parameter URL atau input pengguna.

Berikut Konfigurasinya :

```
set $block_file_injections 0;

if ($query_string ~ "[a-zA-Z0-9_]=http://") {
    set $block_file_injections 1;
}
if ($query_string ~ "[a-zA-Z0-9_]=(\.\.//?)+") {
    set $block_file_injections 1;
}
if ($query_string ~ "[a-zA-Z0-9_]=/([a-z0-9_.]//?)+") {
    set $block_file_injections 1;
}
if ($block_file_injections = 1) {
    return 403;
}
```

Nginx - Common Exploits Prevention

Common Exploits Prevention adalah upaya untuk **mencegah eksploitasi umum** yang sering digunakan oleh peretas untuk menyerang sistem.

Berikut Konfigurasinya :

```
set $block_common_exploits 0;

if ($query_string ~ "(<|%3C).*script.*(>|%3E)") {
    set $block_common_exploits 1;
}
if ($query_string ~ "GLOBALS(=|\[|\%[0-9A-Z]{0,2})") {
    set $block_common_exploits 1;
}
if ($query_string ~ "_REQUEST(=|\[|\%[0-9A-Z]{0,2})") {
    set $block_common_exploits 1;
}
if ($query_string ~ "proc/self/environ") {
    set $block_common_exploits 1;
}
if ($query_string ~ "mosConfig_[a-zA-Z_]{1,21}(=|\%3D)") {
    set $block_common_exploits 1;
}
if ($query_string ~ "base64_(en|de)code\(.*\)") {
    set $block_common_exploits 1;
}
if ($block_common_exploits = 1) {
    return 403;
}
```

Nginx - Spam Prevention

Spam Prevention adalah teknik untuk **mencegah pengiriman pesan atau request berisi spam** ke dalam sistem, seperti di formulir, komentar, atau URL.

Berikut Konfigurasinya :

```
set $block_spam 0;


if ($query_string ~ "\b(ultram|unicauca|valium|viagra|vicodin|xanax|ypxaieo)\b") {
    set $block_spam 1;
}
if ($query_string ~ "\b(erections|hoodia|huronriveracres|impotence|levitra|libido)\b") {
    set $block_spam 1;
}
if ($query_string ~ "\b(ambien|blue\spill|cialis|cocaine|ejaculation|erectile)\b") {
    set $block_spam 1;
}
if ($query_string ~ "\b(lipitor|phentermin|pro[sz]ac|sandyauer|tramadol|troyhamby)\b") {
    set $block_spam 1;
}
if ($block_spam = 1) {
    return 403;
}
```

Nginx - User Agent Blocking

User Agent Blocking Prevention adalah teknik untuk **memblokir akses dari user-agent yang mencurigakan** seperti bot, scraper, atau alat otomatis yang sering digunakan untuk serangan dan pencurian data.

Berikut Konfigurasinya :

```
set $block_user_agents 0;

if ($http_user_agent ~ "Indy Library") {
    set $block_user_agents 1;
}
if ($http_user_agent ~ "libwww-perl") {
    set $block_user_agents 1;
}
if ($http_user_agent ~ "GetRight") {
    set $block_user_agents 1;
}
if ($http_user_agent ~ "GetWeb!") {
    set $block_user_agents 1;
}
if ($http_user_agent ~ "Go!Zilla") {
    set $block_user_agents 1;
}
if ($http_user_agent ~ "Download Demon") {
    set $block_user_agents 1;
}
if ($http_user_agent ~ "Go-Ahead-Got-It") {
    set $block_user_agents 1;
}
if ($http_user_agent ~ "TurnitinBot") {
    set $block_user_agents 1;
}
if ($http_user_agent ~ "GrabNet") {
    set $block_user_agents 1;
}
```

```
if ($block_user_agents = 1) {
    return 403;
}
```