2025/11/13 22:04 1/5 Switche versenken

Switche versenken

Switche versenken ist eine Variante von Schiffe versenken, bei der zunächst virtuelle Schiffe gesetzt und dann durch Kabelstecken versenkt werden müssen.



Das hier ist eine **neue** Version des Switche versenken Spieles. Die alte Version befindet sich hier.

Hardwarevoraussetzungen

- Beliebige HP-Switches (die Software ist getestet mit mehreren 2650er Switchen oder jeweils einem 54er Switch)
- Ein Server zum Hosten des Spiels und zum Abfangen der SNMP-Traps (ein einfacher Raspberry Pi zum Beispiel)
- 2 Laptops für die Spieler
- Einige Patchkabel
- Ein weiterer Switch, um alle Switche und die Laptops und den Server miteinander zu verbinden

Einrichtung der Switche

Allgemein ist es vorgesehen, dass das Spielfeld jedes Spielers aus einer 12×6 Matrix aus Switchports besteht. Bei einem 54er Switch sind das beispielsweise 3 übereinander liegende Module (in diesem Beispiel jeweils immer die Module A, C und E) oder falls keine 54er Switche zur Verfügung stehen, können auch mehrere 1HE-Switche mit mindestens 24 Ports gestacked werden. Für die Einrichtung beider Szenarien folgt nun eine Anleitung:

Mit jeweils 3 1HE Switchen pro Spieler

- 1. Zunächst sollten die Switche alle eine statische IP im Netz des Servers zugeordnet bekommen
 - ∘ In diesem Beispiel wird nun davon ausgegangen, dass die Switche die IPs 10.135.1.20/24-10.135.1.25/24 haben
 - Weiterhin hat in diesem Beispiel der Server die IP 10.135.1.5/24 und die SNMP-Community heißt swvcom (diese wird später bei der Einrichtung des Servers noch auf diesen Wert gesetzt)
- 2. Nun sollte noch die maximale VLAN-Grenze der Switche auf 100 erhöht werden, da jeder Spiele-Port ein eigenes VLAN bekommt, damit es nicht schlimm ist, wenn Loops gesteckt werden
 - Dies kann man am einfachsten tun, indem man menu in der Switch-CLI eingibt und es dort in den VLAN-Einstellungen ändert
- 3. Wenn das getan ist, muss der Switch einmal rebooten, um das VLAN-Limit zu erhöhen
- 4. Wenn er wieder an ist, kann die restliche Konfiguration automatisch erledigt werden, da es sonst sehr mühsam ist, an 3*24 Ports ein eigenes VLAN anzulegen
 - Dafür gibt es dieses Python-Script, welches die VLANs anlegt und auch gleich den SNMP-

Server einrichtet

- Zum Benutzen des Scripts muss der ausführende Rechner die Switche erreichen können und die Software expect muss installiert sein
- Wenn diese Voraussetzungen erfüllt sind, kann das Script mit ./configure_switch.py
 [switch_ip] [server_ip] [snmp_community]
 [anzahl switchports zum isolieren] ausgeführt werden
 - Für einen 2650er 50 Port Switch, bei dem die Ports 1-24 als Spielfeld und 25-48 als Linkerzeuger dienen und 49-50 zum Uplink vorgesehen sind, kann mit oben erwähnten Beispiel-Werten Switch 1 konfiguriert werden:
 - ./configure_switch.py 10.135.1.20 10.135.1.5 swvcom 48 (das muss jetzt noch für die Switche 2-6 ausgeführt werden und ja 24 isolierte Ports würden prinzipiell auch reichen)

Mit jeweils einem 54er Switch pro Spieler

- 1. Als erstes wird hierbei vorausgesetzt, dass als Spielfeld die Module A, C und E in den Switchen verwendet werden. Sind noch mehr Module im Switch können diese als Linkerzeuger genutzt werden (oder auch für den Uplink)
- 2. Zunächst sollten die Switche alle eine statische IP im Netz des Servers zugeordnet bekommen
 - In diesem Beispiel wird nun davon ausgegangen, dass die Switche die IPs 10.135.1.20/24 und 10.135.1.30/24 haben
 - Weiterhin hat in diesem Beispiel der Server die IP 10.135.1.5/24 und die SNMP-Community heißt swvcom (diese wird später bei der Einrichtung des Servers noch auf diesen Wert gesetzt)
- 3. Nun sollte noch die maximale VLAN-Grenze der Switche auf 100 erhöht werden, da jeder Spiele-Port ein eigenes VLAN bekommt, damit es nicht schlimm ist, wenn Loops gesteckt werden
 - Dies kann man am einfachsten tun, indem man menu in der Switch-CLI eingibt und es dort in den VLAN-Einstellungen ändert
- 4. Wenn das getan ist, muss der Switch einmal rebooten, um das VLAN-Limit zu erhöhen
- 5. Wenn er wieder an ist, kann die restliche Konfiguration automatisch erledigt werden, da es sonst sehr mühsam ist, an 3*24 Ports ein eigenes VLAN anzulegen
 - Dafür gibt es dieses Python-Script, welches die VLANs anlegt und auch gleich den SNMP-Server einrichtet
 - Zum Benutzen des Scripts muss der ausführende Rechner die Switche erreichen können und die Software expect muss installiert sein
 - Wenn diese Voraussetzungen erfüllt sind, kann das Script mit
 ./configure_switch_54.py [switch_ip] [server_ip] [snmp_community]
 ausgeführt werden
 - Nun kann mit oben genannten Beispielwerten der erste Switch so konfiguriert werden:
 - ./configure_switch.py 10.135.1.20 10.135.1.5 swvcom (das muss jetzt noch für den anderen Switch mit der anderen IP ausgeführt werden)

Einrichtung des Servers

Für die Erklärung wird von folgenden Beispielwerten ausgegangen (Erläuterungen folgen unten):

- Der Server hat die IP 10.135.1.5/24
- Die SNMP-Community lautet swvcom
- Der SNMP-Trap-Handler liegt unter /etc/snmp/handle.py
- Das Frontend wird unter /var/www/swv/frontend ausgecheckt
- Das Backend wird unter /var/www/swv/backend/files ausgecheckt
- Das venv für das Backend liegt unter /var/www/swv/backend/venv
- Der WSGI-Socket liegt unter /var/www/swv/backend/socket/wsgi.sock
- Die systemd-Unit für das Backend liegt unter /etc/systemd/system/swvbackend.service

Die Einrichtung erfolgt nun so:

- 1. Software installieren:
 - apt install python3-dev default-libmysqlclient-dev libssl-dev python3 python3-venv mariadb-server nginx snmptrapd nodejs npm
- 2. snmptrapd einrichten:
 - Das passende Handle-Script aus dem Repo nach Beispielsweise /etc/snmp/handle.py kopieren
 - Hier liegen aktuell Handle-Scripts für 2650er und 54er Switche vor. Falls andere Switche verwendet werden, kann mithilfe des dev-handle-Scripts nachgeschaut werden, wie die OIDs für das Linkup-Event und die Portnummer heißen und dann kann entsprechend ein Script abgewandelt werden.
 - 2. Die snmptrapd.conf unter /etc/snmp/snmptrapd.conf kann dann mit unseren Beispielwerten so aussehen:
 - /etc/snmp/snmptrapd.conf

```
authCommunity log,execute,net swvcom
traphandle default /etc/snmp/handle.py
```

- 3. snmptrapd aktivieren und starten systemctl enable —now snmptrapd
- 3. Installieren und Konfigurieren des Backends
 - 1. Das Backend nach Beispielsweise /var/www/swv/backend/files auschecken
 - 2. Eine MySQL-Datenbank anlegen und mit dem Datenbank-Schema aus dem Backend-Repo befüllen und auch gleich einen MySQL-Benutzer mit Zugriff auf diese DB anlegen
 - 3. Ein venv für das Backend anlegen
 - cd /var/www/swv/backend
 - 2. python3 -m venv venv
 - 3. source venv/bin/activate
 - 4. pip install wheel
 - 5. pip install -r files/requirements.txt
 - 6. pip install uwsqi
 - 4. Unter Beispielsweise /var/www/swv/backend/files/wsgi.py eine WSGI-Config mit folgendem Inhalt anlegen:
 - wsgi.py

```
from flaskapp.app import app

if __name__ == "__main__":
    app.run()
```

- 5. Unter Beispielsweise /var/www/swv/backend/wsgi.ini die WSGI-INI mit folgendem Inhalt anlegen (Pfade bei Bedarf anpassen):
 - wsgi.ini

```
[uwsgi]
chdir = /var/www/swv/backend/files
wsgi-file = wsgi.py
callable = app

master = true
processes = 5

socket = /var/www/swv/backend/socket/wsgi.sock
chmod-socket = 660
vacuum = true

die-on-term = true
```

- 6. Socket-Verzeichnis unter Beispielsweise /var/www/swv/backend/socket/anlegen
- 7. Backend-Beispielconfig kopieren und anpassen
 - Die Config befindet sich in diesem Beispiel unter /var/www/swv/backend/files/flaskapp/config
 - Am Ende muss dort eine Datei namens secret.py liegen
 - Die secret.example.py ist das Beispiel für mehrere 1HE Switche und secret.example_54.py ist das Beispiel für 54er Switche (die Ports sollten jeweils immer so bleiben können)
- 8. Die systemd-Unit für das Backend anlegen (mit unseren Beispielpfaden):
 - /etc/systemd/system/swv-backend.service

```
[Unit]
Description=uWSGI Instanz für swv-backend
After=network.target

[Service]
User=www-data
Group=www-data
WorkingDirectory=/var/www/swv/backend/
Environment="PATH=/var/www/swv/backend/venv/bin"
ExecStart=/var/www/swv/backend/venv/bin/uwsgi --ini wsgi.ini -
-enable-threads

[Install]
WantedBy=multi-user.target
```

- 9. Die systemd-Unit aktivieren und starten:
 - systemctl enable -now swv-backend
- 4. Installieren und Konfigurieren des Frontends
 - 1. Das Frontend nach Beispielsweise /var/www/swv/frontend auschecken

- 2. Im ausgecheckten Repo unter src/config.ts die BASE_URL in unserem Beispiel auf http://10.135.1.5 abändern
- 3. Frontend bauen:
 - cd /var/www/swv/frontend
 - npm i
 - npm run build
- 5. Den nginx Konfigurieren:
 - Erneut mit unseren Beispielwerten:
 - /etc/nginx/sites-available/default

```
server {
    listen 80 default_server;
    listen [::]:80 default_server;
    root /var/www/swv/frontend/build;

    index index.html index.htm index.nginx-debian.html;

    server_name _;

    location / {
        try_files $uri $uri/ =404;
    }
    location /api/ {
        uwsgi_pass unix:///var/www/swv/backend/socket/wsgi.sock;
        include uwsgi_params;
    }
}
```

- 6. Den nginx reloaden:
 - systemctl reload nginx
- 7. Fertig (Wenn alles geklappt hat, dann sollte das Spiel nun von beiden Laptops aus unter http://10.135.1.5/ erreichbar sein)
- 8. Wenn eine Runde fertig ist, kann das Spiel händisch durch das aufrufen von http://10.135.1.5/api/reset zurückgesetzt werden (von einem beliebigen Laptop aus)

From

https://wiki.fem.tu-ilmenau.de/ - FeM-Wiki

Permanent link:

https://wiki.fem.tu-ilmenau.de/public/sonstiges/unterhaltung/spiele/switche versenker

Last update: 2021/10/14 20:34

