bwCAMPUSNETZ

Zukunftsfähige Konzepte für die Campusnetze
an Universitäten und Hochschulen

Zentrale und herstellerunabhängige VPN-Lösung für den Campus

eduVPN als OpenSource VPN-Lösung

Federführung bei der Erstellung dieses Dokuments: Universität Konstanz
Kontakt: team@bwcampusnetz.de

Inhalt

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

Im Rahmen des Projektes bwCampusnetz und dem daraus resultierenden Arbeitspaket 4 wurde das Problem erkannt, dass an vielen Hochschulen die VPN-Zugänge über mehrere VPN-Endpunkte (z.B. verschiedene Bereichs-Firewalls) verteilt sind. Die Benutzer werden an diesen Endpunkten teilweise manuell gepflegt, was zu einer inkonsistenten Benutzerverwaltung führt: Beispielsweise werden Benutzer beim Verlassen einer Einrichtung im Identity Management (IDM) gesperrt oder gelöscht - diese Änderung erreicht aber nicht immer den jeweiligen VPN-Endpunkt. Weiter besteht die Notwendigkeit, auch externe Personen - z.B. Mitarbeiter von Fremdfirmen - unabhängig von einer Föderation (wie eduroam) zu authentifizieren. Darüber hinaus besteht die Anforderung, nicht nur Benutzer, sondern auch Geräte (VR-Brillen, IoT-Geräte etc.) zu authentifizieren. Die mögliche Lösung sollte flexibel für alle Einrichtungen einsetzbar sein und idealerweise auf Open Source basieren.
Als mögliche Lösung wurde eduVPN betrachtet, eine Open-Source-Lösung, die als zentrales VPN dienen sollte.

1.1 Was ist eduVPN?

eduVPN - entwickelt und gepflegt von GÉANT, dem europäischen Forschungsnetzwerk - ist eine Open-Source-VPN-Lösung, die speziell auf die Bedürfnisse von Forschung und Lehre zugeschnitten ist. Grundsätzlich basiert eduVPN auf den Protokollen Wireguard und OpenVPN und beinhaltet eine Management-Plattform sowie weitere Tools. Es gibt eduVPN Clients für alle gängigen Betriebssysteme. Der gesamte Quellcode für die Serveranwendung und die Clients ist öffentlich verfügbar.
Das Ziel von eduVPN ist es, Studierenden, Lehrenden, Angestellten und Forschenden eine sichere Verbindung zu ihrem institutionellen Netzwerk über das Internet zu ermöglichen. eduVPN wurde von Anfang an mit Blick auf Datenschutz und Sicherheit entwickelt. eduVPN basiert auf den Prinzipien der Datenminimierung und Zweckbindung. Die eduVPN-App auf den Endgeräten wurde nach Privacy-by-Design-Prinzipien entwickelt und verwendet keine Tracker oder telemetrische Funktionen.[1]

[1] https://www.eduvpn.org/wp-content/uploads/2024/05/Privacy-overview-eduVPN-final.pdf

Es gibt zwei Möglichkeiten, wie die VPN-Lösung betrieben werden kann: einmal als eduVPN oder als "Let's Connect!". eduVPN wird - wie bereits erwähnt - verwendet, um eine VPN-Lösung für Bildungs- und Forschungseinrichtungen anzubieten. Dabei kann die Hochschule direkt in der eduVPN-Applikation ausgewählt werden. "Let's Connect!" bietet den Dienst „Secure Internet Access“: Mit der zunehmenden Verbreitung von öffentlichen WLANs greifen immer mehr Studierende und Mitarbeiter auf Informationen in potenziell unsicheren Netzwerken zu und riskieren dabei, dass ihre Daten abgefangen oder abgehört werden. Oder sie befinden sich in Ländern mit restriktivem Internetzugang und können deshalb bestimmte Seiten und Dienste nicht erreichen. Im Rahmen des „Secure Internet Access“ bieten Institutionen wie das DFN öffentliche eduVPN-Server an, über die sich die Nutzer sicher und anonym einwählen können. [2]
Um eduVPN zu betreiben, müssen der Server und die betreibende Einrichtung bei GÉANT registriert sein. Für den Dienst „Secure Internet Access“ ist eine separate Registrierung erforderlich.

1.2 Wie funktioniert eduVPN?

1.2.1 Server-Sicht

Die eduVPN-Software für den Server besteht aus den beiden Komponenten „Portal“ und dem „Node“. Die beiden Komponenten können gemeinsam auf einem Server oder einer virtuellen Maschine laufen, oder getrennt voneinander. Das Portal liefert die Weboberfläche, die Datenbank, die Nutzerauthentifizierung und die OAuth API für die VPN-Anwendungen und die Dienstkonfiguration. Der Node ist verantwortlich für die Konfiguration der VPN-Software, d.h. OpenVPN und WireGuard, die Kommunikation mit dem Portal und die Abwicklung der VPN-Verbindungen selbst. Es ist grundsätzlich möglich, mehrere Nodes mit einem Portal zu betreiben, um die Last der VPN-Verbindungen zu verteilen. So kann auch ein komplexeres Hochverfügarkeitssetup aufgebaut werden, mit mehreren Nodes und Portalen, um Ausfälle abzufedern und die Last zu verteilen.[3]

Es werden mehrere Authentifizierungsmechanismen von eduVPN unterstützt, wie SAML (Shibboleth), LDAP und RADIUS. Alternativ ist es auch möglich, eine lokale Datenbank mit

[2] https://www.eduvpn.org/secure-internet-access/
[3] https://docs.eduvpn.org/server/v3/ha.html

Nutzerdaten manuell zu pflegen. eduVPN bietet mit der Version 3.X selbst keinr Zwei-Faktor-Authentifizierung (2FA), die Funktion muss vom „Identity Provider“ (z.B. via Shibboleth) angeboten werden.[4]

eduVPN bietet die Möglichkeit, verschiedene Profile für Nutzer anzulegen. Diese Profile können auf unterschiedlichen Nodes enden und können auf Benutzergruppen oder einzelne Nutzer beschränkt werden. Zudem werden in dem Profil noch Routen, DNS-Server, IP-Adressbereiche und das Standard-VPN-Protokoll, die der Nutzer am Ende erhält, festgelegt. Die Konfiguration erfolgt auf dem Portal, das eine Art Verwaltungsfunktion übernimmt.[5]

1.2.2 Client-Sicht

Ein Nutzer kann einen der offiziellen eduVPN Clients verwenden, der nach Auswahl der Einrichtung und der Anmeldung des Nutzers die Konfiguration automatisch über die OAuth API des Portals abruft.[6] Alternativ kann sich der Nutzer über die Weboberfläche anmelden und manuell eine Wireguard- oder OpenVPN-Konfiguration erstellen lassen. Abhängig von der Auswahl des Profils (falls mehrere zur Auswahl stehen) wird der VPN-Zugang entsprechend konfiguriert. Das VPN-Protokoll, das als Standard festgelegt wurde, wird verwendet, um einen VPN-Tunnel mit Wireguard oder OpenVPN aufzubauen. In der App kann bei Problemen mit Wireguard die Einstellung TCP-Erzwingen aktiviert werden, wodurch OpenVPN über TCP verwendet wird. In Zukunft soll möglicherweise OpenVPN entfallen, wobei Wireguard über TCP möglich sein soll. Darüber hinaus bietet die App keine weiteren Einstellungsmöglichkeiten, was dem KISS-Prinzip[7] entspricht.

[4] https://docs.eduvpn.org/server/v3/portal-config.html
[5] https://docs.eduvpn.org/server/v3/profile-config.html
[6] https://docs.eduvpn.org/server/v3/api.html
[7] Keep it Small & Simple

2 Konzept & Aufbau (Uni Konstanz)

In diesem Kapitel wird die Struktur von eduVPN an der Universität Konstanz betrachtet, die im Rahmen des Projektes bwCampusnetz aufgebaut wurde. Dabei werden die von eduVPN benötigten Kommunikationswege betrachtet und die Platzierung der einzelnen Komponenten nach dem Zonenkonzept des BSI festgelegt.

Grundsätzlich stellt sich zunächst die Frage, welche Benutzergruppe (z.B. Studierende, Administratoren) die Zielgruppe der VPN-Lösung ist und welche entsprechenden Anforderungen erfüllt werden müssen. An der Universität Konstanz sollte im ersten Schritt die alte Cisco AnyConnect-Lösung für Studierende und Mitarbeiter (ohne besondere Anforderungen bzw. ohne Zugriff auf sensible Bereiche) abgelöst werden. Im zweiten Schritt wurde das alte Admin-VPN mit Zugang zu sensiblen Bereichen abgelöst. Hier bestand zusätzlich die Anforderung, eine Multi-Faktor-Authentifizierung (MFA) sowie eine Autorisierung auf Benutzerebene (nicht auf Netzwerkbereichen) zu implementieren.
Die Authentifizierung erfolgt sowohl für Studierende als auch für Administratoren über Shibboleth. Für Administratoren besteht zusätzlich eine MFA-Pflicht - im Falle der Universität Konstanz werden dafür Hardware-Token (YubiKey) verwendet. Nach der einmaligen Anmeldung wird ein Profil/Key-Paar erstellt, das standardmäßig 30 Tage gültig ist.

Wie im vorherigen Kapitel erläutert, sind die Hauptkomponenten von eduVPN ein Portal und ein Node. Zusätzlich sollte/kann ein Proxy für das Portal eingerichtet werden, damit keine Daten aus dem unsicheren Internet direkt zum Portal fließen können. Weitere Informationen dazu finden sich im nächsten Kapitel.

2.1 eduVPN Kommunikationswege

2.1.1 Interne Kommunikation

Die Kommunikation zwischen dem Portal und den Nodes erfolgt über die API via HTTPS. So wird beispielsweise die Signalisierung von einem VPN-Verbindungsaufbau von einem Nutzer

auf diese Weise vom Node an das Portal übertragen. Das Portal verwendet dazu standardmäßig den Port 443, bei den Nodes kann jeweils ein beliebiger Port in der Konfiguration festgelegt werden.

2.1.2 Ersteinrichtung

Der Client verbindet sich via HTTPS (Standard Port) zum Portal und fordert ein Profil an. Das Portal leitet diese Anfrage über Shibboleth an das IDM weiter. Der Nutzer wird auf die Shibboleth-Seite weitergeleitet und authentisiert sich dort. Nach erfolgreicher Authentisierung erzeugt das Portal ein KeyPair/Zertifikat und überträgt dieses zusammen mit dem angefragten Profil an den Client.

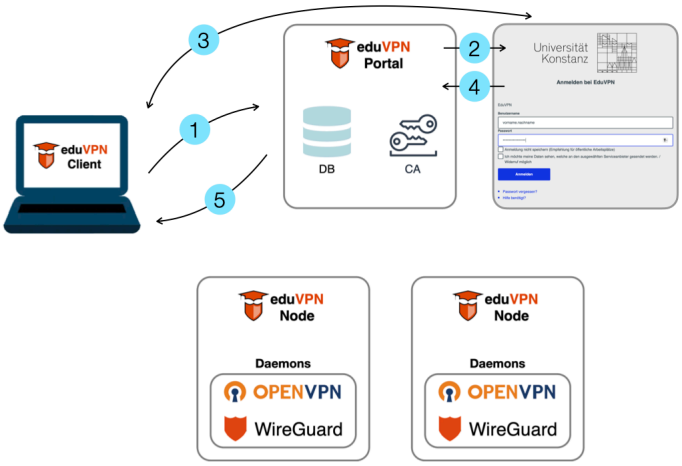

Abb. 2.1: eduVPN Ablauf - Profil erstellen

2.1.3 Aufbau VPN-Verbindung

Der Client verbindet sich mit dem im Profil angegebenen Node. Der Node prüft dann beim Portal, ob das KeyPair noch gültig ist. Ist ‘Live Permissions’ auf dem Portal konfiguriert, wird ein LDAP-Abruf durchgeführt, um zu prüfen, ob der Nutzer gesperrt wurde oder Rechte entzogen wurden. Der Node überträgt die Antwort vom Portal an den entsprechenden Daemon, der die Verbindung akzeptiert. Ohne die ‘Live Permissions’-Option wird nur die Gültigkeit des KeyPairs auf dem Portal geprüft; erst nach Ablauf der im Profil festgelegten Laufzeit ist eine erneute Authentifizierung wie zuvor beschrieben erforderlich.

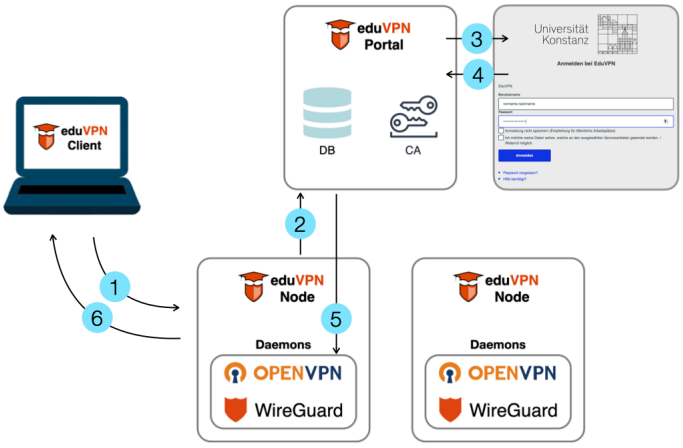

Abb. 2.2: eduVPN Ablauf - Aufbau VPN

2.2 Komponenten im Zonenkonzept

2.2.1 VPN-Zugang für Studenten und Mitarbeiter

Das eduVPN-Portal fungiert als Controller für das eduVPN. Auf ihm werden die Benutzerprofile konfiguriert, Benutzer authentifiziert und das Webportal zur Verwaltung bereitgestellt. Gemäß dem BSI-Zonenkonzept muss das Portal daher als schutzbedürftig eingestuft werden, was bedeutet, dass kein direkter Traffic aus dem unsicheren Internet zum Portal fließen darf. Um dies zu gewährleisten, haben wir einen Portal-Proxy in einer WebDMZ aufgestellt, der als Schnittstelle zum Internet und dem Portal dient.

Da die Endgeräte der Studenten nicht von uns verwaltet werden, klassifizieren wir diesen Traffic als ungesichert. Der Node wurde daher in eine DMZ an der Perimeter-Firewall positioniert. Über den Node gelangt der Nutzer in die Fachbereichs-Netze, jedoch nicht in schutzbedürftige Bereiche.

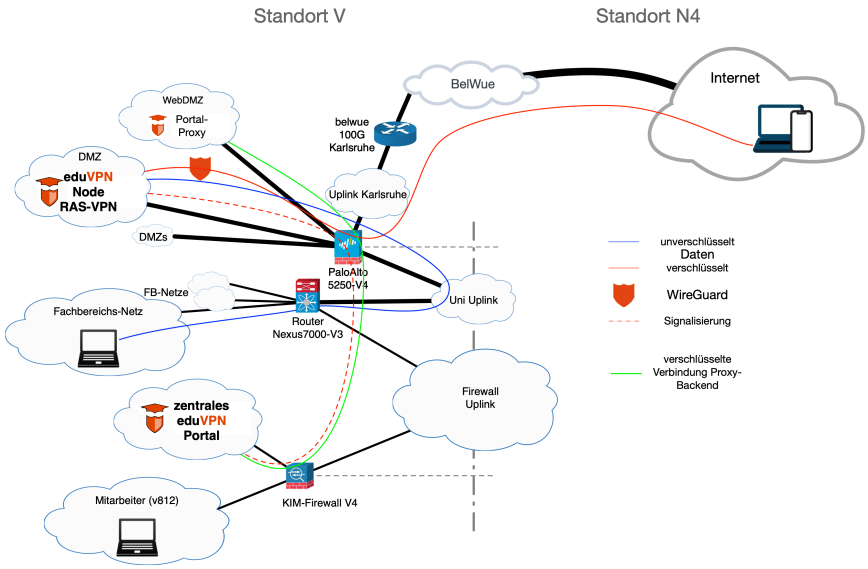

Abb. 2.3: eduVPN Setup - RemoteAccess-VPN

2.2.2 VPN-Zugang für Administratoren

Der VPN-Zugang für Administratoren nutzt das gleiche Portal und den dazugehörigen Proxy wie der für die Studenten. Der wesentliche Unterschied besteht in der Positionierung des Nodes. Im Gegensatz zu 2.2.1 wird dieser Traffic als vertrauenswürdig definiert. Der Node ist an der Internen Firewall positioniert und ermöglicht den Zugriff auf schutzbedürftige Bereiche anhand von individuellen Rechten. Jeder Administrator soll lediglich auf sein eigenes administriertes System zugreifen und keine allgemeinen Netzfreigaben nutzen.
Beim Erzeugen eines Profils bzw. Verbinden des VPNs wird diese Information inklusive der aktuellen IP-Adresse via Syslog an unsere Palo Alto Firewall übermittelt. Die Firewall erzeugt aus dieser Information eine UserID. Mit dieser UserID kann so der einzelne Nutzer oder Administrator gezielt berechtigt werden. Diese Nutzerberechtigung ist grundsätzlich auch mit anderen Firewall-Herstellern möglich, solange dieser z.B. eine Auswertung von Syslog oder entsprechende API-Calls zulässt. Dazu wird der “ScriptConnectionHook” verwendet, mit dem ein Script mit beliebiger Funktion aufgerufen wird, wenn der Nutzer sich mit dem VPN verbindet oder trennt.

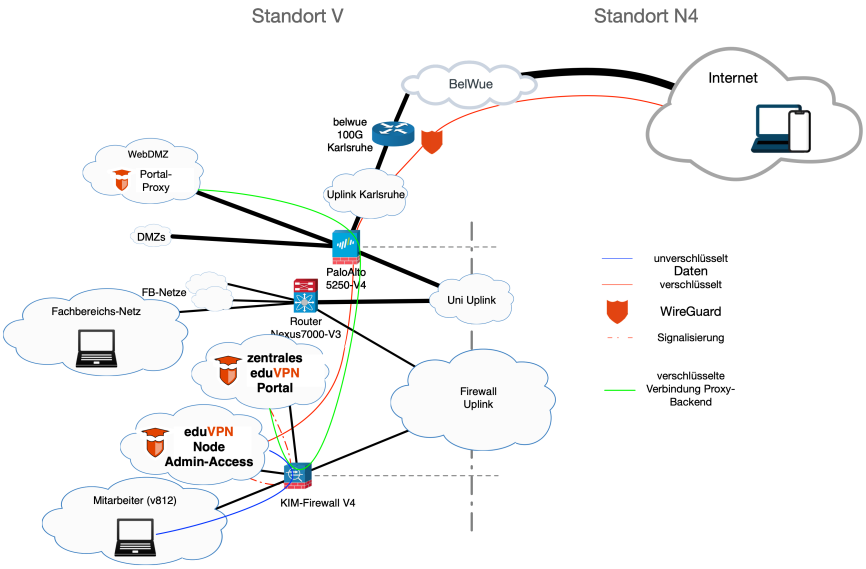

Abb. 2.4: eduVPN Setup - Admin-VPN

3 Installation & Inbetriebnahme (Uni Konstanz)

In diesem Kapitel wird die Installation und Inbetriebnahme des in dem vorherigen Kapitel beschriebenen eduVPN Aufbaus am Beispiel der Uni Konstanz erläutert. Die von eduVPN zur Verfügung gestellte Dokumentation auf der eigenen Webseite bietet grundsätzlich einen guten Start, um ein eigenes eduVPN-Setup aufzubauen. Daher ist die nachfolgende Installation und Inbetriebnahme als Ergänzung zur vorhandenen Dokumentation zu sehen.
eduVPN empfiehlt für die einzelnen Komponenten als Betriebssystem Debian, ob als Bare-Metal oder in einer Virtualisierung, ist schlussendlich dem Administrator überlassen. Für den Aufbau der Uni Konstanz wurde Debian als Betriebssystem in einer Virtualisierungsumgebung verwendet.

3.1 eduVPN-Portal

3.1.1 eduVPN-Portal Installation

Die Installation des eduVPN-Portals erfolgt unter Debian im Wesentlichen mit den folgenden Befehlen. Es wird ein Installationsscript aus der eduVPN-Repo heruntergeladen und ausgeführt, wobei unter anderem der Webserver Apache2 für das Portal installiert wird.

```
$ sudo apt -y install ca-certificates wget
$ wget https://codeberg.org/eduVPN/deploy/archive/v3.tar.gz
$ tar -xzf v3.tar.gz
$ cd deploy
$ sudo -s
# ./deploy_debian.sh
```

Die grundsätzliche Installationsanleitung für das eduVPN Portal ist:
`https://docs.eduvpn.org/server/v3/deploy-debian.html`

Nach der Installation ist das Web-Portal und eine VPN-Node (Node 0) auf dem Server installiert.

Im Anschluss muss noch die Shibboleth Software installiert werden:

```
$ sudo apt install libapache2-mod-shib
$ sudo shib-keygen -n sp-encrypt
$ sudo shib-keygen -n sp-signing
```

3.1.2 Zertifikate erstellen und installieren

Es wird empfohlen, keine Let's Encrypt Zertifikate zu verwenden, sondern stattdessen ein Serverzertifikat der Hochschuleinrichtung zu nutzen. Wichtig: Als Subject Alternative Names müssen alle Namen des Portalservers, der Name des Proxys und der offizielle Dienstname eingetragen werden. Das hier erstellte Zertifikat wird auch auf dem Proxy und im Shibboleth SP installiert.

Installation des Zertifikats für das eduvpn Portal: Das Zertifikat, der Private Key und die Zertifikatskette wird nach /etc/shibboleth/certs/ kopiert.

Danach muss SSL mit den Zertifikaten auf dem Portal aktiviert werden: Dazu die Konfiguration des unter /etc/apache2/sites-available in einem Editor unter <VirtualHost *:443> folgende Zeilen eintragen:

```
    $ nano /etc/apache2/sites-available/eduvpn.example.org.conf
```

```
    SSLEngine on
SSLCertificateFile /etc/shibboleth/certs/eduvpn.pem
SSLCertificateKeyFile /etc/shibboleth/certs/eduvpn.key
SSLCertificateChainFile /etc/shibboleth/certs/eduvpn-chain.pem
```

Im Anschluss ein $ sudo systemctl restart apache2

3.1.3 Shibboleth Anbindung konfigurieren

Die Installation der Shibboleth Software erfolgte zu Beginn des Kapitels, nun erfolgt die Konfiguration des Shibboleth Anbindung. Die grundlegende Konfiguration befindet sich hier: `https://docs.eduvpn.org/server/v3/shibboleth-sp.html`

Anpassung am Portal Webserver für die Shibboleth-Anbindung:

Wie zuvor die Webserver Konfiguration in einem Editor öffnen und die Konfiguration folgendermaßen anpassen (Siehe /config Ordner). Wichtig als Servername die Dienst-URL die auf den Proxy zeigt eintragen!

Im Anschluss müssen noch die Metadaten für die Shibboleth-Anbindung konfiguriert werden: Dazu das file /etc/shibboleth/shibboleth2.xml öffnen und entsprechend der folgenden Konfiguration anpassen (Siehe /config Ordner).

ACHTUNG: Die die Entity ID die hier gesetzt wird, muss so auch im Antrag mitgegeben werden. Diese kann nicht mehr geändert werden!
Des Weiteren müssen hier wie in der config die im vorausgegangenen Schritt erstellten Zertifikate eingebunden werden, der IDP eingetragen und die DFN-AAI eingetragen werden.

Im Anschluss kann die Shibboleth-Anbindung an der jeweiligen Einrichtung beantragt werden.Dabei muss die Entity ID (der gesamte Pfad!) und das Zertifikat (Ohne Privatekey) mitgegeben werden. Es bietet sich an die Kompletten Metadaten unter `https://vpn.example.org/\Shibboleth.sso/Metadata` dem Antrag anzuhängen. Das erleichtert die Einrichtung.

Anschließend kann in der Portal-Konfiguration Shibboleth aktiviert werden: Dazu unter /etc/vpn-user-portal/config.php folgenden Abschnitt aktivieren:

```
        'authModule' => 'ShibAuthModule',       // SAML (Shibboleth)
```

```
        // ** SAML (Shibboleth) **

    'ShibAuthModule' => [
        'userIdAttribute' => 'cn',
        //'userIdAttribute' => 'eppn',
        // ** AUTHORIZATION | PERMISSIONS **
        'permissionAttributeList' => ['cn'],
        //'permissionAttributeList' => ['affiliation'],
],
```

ACHTUNG: das userIdAttribute muss auf das Feld gesetzt werden, das den Username enthält, wie hier der CN. Die permissionAttributeList enthält die Felder, anhand derer später Rechte vergeben werden sollen.
Wenn IDs wie cn verwendet werden, müssen diese zuvor in der attribute-map.xml definiert

werden. Es muss dazu die /etc/shibboleth/attribute-map.xml folgendermaßen angepasst werden (Siehe /config Ordner).

3.1.4 Konfiguration des eduVPN Portals

Alle Einstellungen inklusive aller Profile für den Controller/Portal befinden sich unter /etc/vpn-user-portal/config.php. Vorerst müssen hier keine weiteren Einstellungen getätigt werden.

ACHTUNG: Änderungen an der VPN-Konfiguration müssen durch einen Neustart des VPN-Dienstes übernommen werden:

```
sudo vpn-maint-apply-changes
```

→ Aktuelle Konfiguration im /config Ordner

3.1.5 Branding

Das Aussehen des Portals kann beliebig angepasst werden und gemäß dem Corporate Design der jeweiligen Einrichtung gestaltet werden. Die hierfür relevante Dokumentation befindet sich unter: `https://docs.eduvpn.org/server/v3/branding.html` & `https://docs.eduvpn.org/server/v3/custom-branding.html`

Wichtig ist: die Themes können durch Anlegen der entsprechenden Dateien und Ordner unter /etc/vpn-user-portal/views überschrieben werden. Nach Updates kann es notwendig sein das Branding anzupassen.

3.1.6 Firewall und Ports

Nach dem Ausführen des Installation-Scripts ist standardmäßig die Firewall aktiv und bietet einen Grundschutz an. Es können zusätzlich noch Anpassungen vorgenommen werden, wie den SSH-Zugang einzuschränken oder zusätzliche Ports für OpenVPN/Wireguard zu öffnen. Dazu ist eine ausführliche Anleitung auf `https://docs.eduvpn.org/server/v3/firewall.html` zu finden.

Der VPN Nodes muss auf folgenden Ports erreichbar sein:

- 22 SSH
- 80,443 HTTPS, HTTP
- OpenVPN Ports
- Wireguard Ports

```
    define EXTERNAL_IF = ens192

table inet filter {
    chain input {
            type filter hook input priority filter; policy drop;
            ct state vmap { invalid : drop, established : accept, related :
               ↪ accept }
            iif "lo" accept
            tcp dport { 22, 80, 443, 1194, 10050 } accept
            udp dport { 1194, 51820 } accept
            icmp type echo-request limit rate 5/second accept
            icmpv6 type { icmp, destination-unreachable, echo-reply,
               ↪ echo-request, nd-neighbor-solicit, nd-router-advert,
               ↪ nd-neighbor-a>
            ip6 ecn not-ect accept
    }

    chain forward {
            type filter hook forward priority filter; policy drop;
            tcp flags syn tcp option maxseg size set rt mtu
            ct state vmap { invalid : drop, established : accept, related :
               ↪ accept }
            iifname { "tun0", "tun1", "wg0" } oifname $EXTERNAL_IF accept
    }

    chain postrouting {
            type nat hook postrouting priority srcnat; policy accept;
            ip saddr { 10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16 } oifname
               ↪ $EXTERNAL_IF masquerade
            ip6 saddr fc00::/7 oifname $EXTERNAL_IF masquerade
    }
}
```

Achtung: es wird auf Debian eine nftables installiert und keine iptables wie noch in manchen Stellen der Doku beschrieben. Die Regeln müssen entsprechend angepasst werden. **Für IPv6 ist es wichtig in nftables ALLE ICMPv6 Typen zuzulassen!**

3.1.7 Netzwerk Konfiguration

Die Netzwerkkonfiguration auf dem Portal ist relativ einfach, da nur ein Netzwerk Interface konfiguriert werden muss. Daher erfolgt das einfach über die Datei /etc/network/interfaces:

```
source /etc/network/interfaces.d/*

    # The loopback network interface
auto lo
iface lo inet loopback

    # The primary network interface
auto ens192
iface ens192 inet static
   address 134.34.203.88/26
   gateway 134.34.203.65
   dns-nameservers 134.34.3.3 134.34.3.2
   dns-search kim.uni-konstanz.de uni-konstanz.de

iface ens192 inet6 static
   address 2001:7C0:2810:1111::31:2088/64
   gateway 2001:7C0:2810:1111::1
   dns-nameservers 2001:7c0:2800::3:2, 2001:7c0:2800::3:3
   dns-search kim.uni-konstanz.de uni-konstanz.de
   accept_ra 2
```

3.1.8 Script Connection Hook

Es besteht die Möglichkeit, ein (benutzerdefiniertes) Skript zu starten, wenn sich ein VPN-Client verbindet und/oder die Verbindung trennt. Dies kann beispielsweise verwendet werden, um zu prüfen, ob ein Benutzer noch im LDAP-Server vorhanden ist, oder um Firewall-Änderungen in einem entfernten System auszulösen. Wie genau die Funktionsweise von

dem script connection hook ist, kann hier nachgelesen werden: `https://docs.eduvpn.org/server/v3/script-connection-hook.html`

Für den VPN-Admin an der Uni Konstanz wird der hook verwendet, um einen Log-Eintrag zu generieren, aus dem wiederum unsere Palo Alto Firewall eine UserID generiert. Das Script sieht wie folgt aus:

```
#!/bin/bash

# Global Hook Script for eduVPN

#User-ID for VPN4
VPN4='vpn4'

if [[ "$VPN_PROFILE_ID" == *"$VPN4"*]] ; then

logger -n 134.34.24.1 -t eduvpn "Event: ${VPN_EVENT} User: ${VPN_USER_ID} IP:
    ↪ ${VPN_IP_FOUR} IPv6: ${VPN_IP_SIX} Profile: ${VPN_PROFILE_ID} Proto:
    ↪ ${VPN_PROTO}"

echo "$(date) Event: ${VPN_EVENT} User: ${VPN_USER_ID} IP: ${VPN_IP_FOUR}
    ↪ IPv6: ${VPN_IP_SIX} Profile: ${VPN_PROFILE_ID} Proto: ${VPN_PROTO}" >>
    ↪ /var/log/test.log

fi

exit 0
```

3.1.9 Reset des VPN Systems

Um alle Verbindungen zu entfernen, um zum Beispiel bei Änderung der Profillebenszeit die Laufzeit zu synchronisieren, gibt es einen Befehl:

```
sudo vpn-maint-reset-system
```

Dieser Befehl muss auf dem VPN-Portal und allen Nodes ausgeführt werden.

Danach werden insbesondere ALLE Keys vom Portal und den Nodes gelöscht, einschließlich der Keys der Nodes zum Portal. Diese müssen dann auf dem Portal neu erzeugt und in die Nodes neu eingepflegt werden. Anschließend müssen die SSL-Zertifikate für die Nodes wieder

auf dem Portal eingepflegt werden. Eventuell ist es erforderlich, die Nodes danach nochmals neuzustarten, um alle Services wieder sauber zu starten.

3.2 eduVPN-Portal Proxy

Der Proxy, der dem Portal vorgeschaltet ist, wird mittels Apache2 mit mod_proxy realisiert. Im Folgenden wird die Installation und Konfiguration des Web-Proxys erläutert. Dabei ist zu beachten, dass es einen Unterschied macht, ob Shibboleth verwendet wird oder nicht; die Proxy-Konfiguration unterscheidet sich entsprechend.

3.2.1 Installation Apache2

Auf einer separaten Maschine (VM oder Hardware), wird für den Web-Proxy ein Apache2 Webserver installiert. Wie zuvor wir als Betriebssystem Debian verwendet.

Installation des Apache2 mit folgendem Befehl:

```
sudo apt install apache2
```

Im Anschluss müssen noch die entsprechenden Module aktiviert werden:

```
sudo a2enmod proxy
sudo a2enmod proxy_html
sudo a2enmod proxy_http
```

Wichtig: auf dem Proxy müssen auch das Zertifikat und der Privatekey des Portals installiert werden unter: SSLCertificateFile /etc/apache2/ssl/eduvpn.pem und SSLCertificateKeyFile /etc/apache2/ssl/eduvpn.key

3.2.2 Konfiguration Apache2 Proxy

Die Konfiguration befindet sich unter /etc/apache2/sites-available/000-default.conf.
Diese muss folgendermaßen angepasst werden, dass Shibboleth und der Dienst funktionieren (Siehe /config Ordner).

3.3 eduVPN-Node

eduVPN bietet die Möglichkeit, mehrere Nodes (VPN-Endpunkte) zu betreiben. Diese können zum einen für Loadbalancing-Zwecke eingesetzt werden, aber auch, um verschiedene Profile auf unterschiedliche Nodes zu leiten.

3.3.1 Installation der VPN-Node

Für die Installation und das Einrichten des Nodes aus der eduVPN Repo das Installations-Script herunterladen und ausführen:

```
$ curl -L -O https://codeberg.org/eduVPN/deploy/archive/v3.tar.gz
$ tar -xzf v3.tar.gz
$ cd deploy
```

Dann die Node installieren:

```
$ sudo -s
# ./deploy_debian_node.sh
```

Weitere Information und Details sind in der offiziellen Dokumentation zu finden: `https://docs.eduvpn.org/server/v3/multi-node.html`

3.3.2 Erstellen der Node-Keys auf dem Portal

Damit sich die Nodes mit dem Portal verbinden kann, müssen auf dem Portal Keys erstellt werden, die dann dem Node mitgegeben werden. Dazu auf dem Portal folgenden Befehl ausführen:

```
sudo /usr/libexec/vpn-user-portal/generate-secrets --node 1
```

Die ”1”hinter –node muss durch die entsprechende Node Nummer ersetzt werden. Im Anschluss befindet sich der Node-Key auf dem Portal unter /etc/vpn-user-portal/keys/node.1.key Dieser lässt sich mit nachfolgendem Befehl auslesen und kann kopiert werden.

```
$ sudo less /etc/vpn-user-portal/keys/node.1.key
```

3.3.3 Installieren des Node-Keys auf der Node

Der oben kopierte Key muss jetzt auf der Node installiert werden. Dazu nutzt man am besten folgenden Befehl. Er verhindert, dass Whitespaces mit in den Key gelangen:

```
$ echo -n 'SECRET' | sudo tee /etc/vpn-server-node/keys/node.key
```

3.3.4 Konfiguration der Node

Zuerst müssen in der Konfiguration für den Node unter /etc/vpn-server-node/config.php die Node-Nummer, der Link zum Portal/Controller und die Liste der verfügbaren Profile eingestellt werden. Die Konfiguration für die Node 1 sieht zum Beispiel so aus:

```
    <?php

return [

   // Node API URL

   // DEFAULT: http://localhost/vpn-user-portal/node-api.php

   'apiUrl' =>
   ↪ 'https://eduvpn-portal.kim.uni-konstanz.de/vpn-user-portal/node-api.php',

   // specify the number of this node

   // DEFAULT: 0

   'nodeNumber' => 1,

   // Override whether to prefer AES over CHACHA with OpenVPN

   // DEFAULT: auto detect whether the CPU supports hardware accelerated AES,

   // if it does, prefer AES

   //'preferAes' => true,

   //'preferAes' => false,
```

```
    // specify the profiles that you want to load on this node, empty list,
    // i.e. [], means ALL profiles
    // DEFAULT: []
    //'profileIdList' => [],
    'profileIdList' => ['default'],
];
```

Es bietet sich an, mit curl zu testen, ob das Portal vom Node aus erreichbar ist, falls eine Firewall dazwischen sein sollte und ggf. noch Freigaben fehlen sollten:

```
$ curl https://vpn.example.org/vpn-user-portal/node-api.php
```

Im nächsten Schritt muss der VPN-Server auf dem Node noch konfiguriert werden:

Dazu muss in der Konfigurationsdatei unter /etc/default/vpn-daemon noch eingestellt werden, dass der Server auf den Controller-Port hört:

```
LISTEN=:41194
```

Des Weiteren muss noch Wireguard in derselben Konfigurationsdatei aktiviert werden mit:

```
WG_DEVICE=wg0
```

Die Konfiguration sollte dann so etwa aussehen:

```
# Listen Address
# OPTIONAL, DEFAULT = 127.0.0.1:41194
# OPTIONAL, DEFAULT = 127.0.0.1:41194
LISTEN=:41194
# The WireGuard Device
# OPTIONAL, DEFAULT = wg0
```

```
WG_DEVICE=wg0

#WG_DEVICE=wg5

# Path to "ca.crt", "server.crt" and "server.key" to enable TLS when

# "systemd credentials" is not used

# OPTIONAL, DEFAULT = no TLS, HTTP only

#CREDENTIALS_DIRECTORY=/etc/vpn-daemon
```

Im Anschluss den VPN-Service neu starten:

```
$ sudo systemctl restart vpn-daemon
```

3.3.5 Firewall und Ports

Der VPN Node muss auf folgenden Ports erreichbar sein:

- 22 SSH
- 41194 HTTPS, HTTP, OpenVPN
- OpenVPN Ports
- Wireguard Ports

Auf der VPN-Node läuft auch als Firewall ein nftables. Daher muss zum einen dort die Kommunikation vom Portal zur Node freigegeben werden dann muss hier auch ICMPv6 komplett freigegeben werden und das NAT muss eingerichtet werden. Die Konfiguration erfolgt in /etc/nftables.conf. Die Konfiguration kann so aussehen:

```
    define EXTERNAL_IF = ens192

table inet filter {

   chain input {
```

```
        type filter hook input priority filter; policy drop;

        ct state vmap { invalid : drop, established : accept, related : accept
↪ }

        iif "lo" accept

        tcp dport { 22, 1194 } accept

        udp dport { 1194, 51820 } accept

        # vpn-daemon

        ip saddr 134.34.0.0/16 tcp dport 41194 accept

        ip6 saddr 2001:7c0:2800::/40 tcp dport 41194 accept

        icmp type echo-request limit rate 5/second accept

        icmpv6 type { echo-request limit rate 5/second, nd-router-advert,
↪ nd-neighbor-solicit, nd-neighbor-advert } accept

    }

    chain forward {

        type filter hook forward priority filter; policy drop;

        tcp flags syn tcp option maxseg size set rt mtu

        ct state vmap { invalid : drop, established : accept, related : accept
↪ }

        iifname { "tun0", "tun1", "wg0" } oifname $EXTERNAL_IF accept

    }

    chain postrouting {

        type nat hook postrouting priority srcnat; policy accept;

        ip saddr { 10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16 } oifname ens224
```

```
    ↪ snat ip to 134.34.8.0/24

        ip6 saddr fc00::/7 oifname $EXTERNAL_IF masquerade

    }

}
```

Zu beachten ist hier auch, dass das NAT auf einem anderen Interface stattfindet als dem Externen, da der VPN Client Traffic auf das zweite Interface geroutet werden soll. Mehr dazu im nächsten Abschnitt.

3.3.6 Netzwerk Konfiguration

Die Netzwerk- und Routing-Konfiguration erfolgt mit dem Tool Netplan bzw. der dazugehörigen YAML-Files oder dem Standard IP-Kommando Tool unter Linux. Grundsätzliche Anleitung: `https://docs.eduvpn.org/server/v3/source-routing.html`

Hat die VPN-Node mehrere Netzwerkkarten und soll der Client Traffic über eine andere Netzwerkkarte/Route als die Default Route gehen, so muss zusätzlich das Routing geändert werden. Um die Konfiguration übersichtlich und persistent zu machen ist netplan zu empfehlen.

Installation unter Debian:

```
    $ sudo apt install netplan.io
```

Im Anschluss kann Netplan konfiguriert werden, indem eine Konfiguration unter /etc/netplan/00-netconfig.yaml angelegt wird. Die Konfiguration inklusive der Routen kann folgendermaßen aussehen:

```
network:
  version: 2
  renderer: networkd
  ethernets:
    ens192:
      dhcp4: no
      dhcp6: no
      accept-ra: true
```

```
      addresses:
        - 134.34.247.140/25
        - 2001:7C0:2880:200D::31:140/64
#       gateway4: 134.34.247.129
#       gateway6: 2001:7C0:2880:200D::1
      nameservers:
        search:
          - "kim.uni-konstanz.de"
          - "uni-konstanz.de"
        addresses:
          - 134.34.3.3
          - 134.34.3.2
          - 2001:7c0:2800::3:2
          - 2001:7c0:2800::3:3
      routes:
        - to: default
          via: 134.34.247.129
        - to: default
          via: 2001:7c0:2880:200D::1
      routing-policy:
        - from: 10.11.8.0/21
          table: 200
        - from: 10.11.0.0/21
          table: 200
        - from: 134.34.8.0/24
          table: 200
        - from: 2001:7c0:28E0:0102::/64
          table: 200
        - from: 2001:7c0:28E0:0103::/64
          table: 200
    ens224:
      dhcp4: no
      dhcp6: no
      accept-ra: no
      addresses:
        - 134.34.8.0/24
        - 192.168.1.5/27
        # - 2001:7c0:501:0305::5/64
        - 2001:7C0:28E0:4201::5/64
      routes:
        - to: default
          # via: 2001:7c0:501:0305::1
```

```
          via: 2001:7C0:28E0:4201::1
          table: 200
        - to: default
          via: 192.168.1.1
          table: 200
```

Für das Routing der VPN-Clients wird die in der Tabelle 200 angegebene Default Route genutzt und das zweite Interface.
Die Konfiguration muss dann noch aktiviert werden:

```
$ sudo netplan generate
$ sudo netplan apply
```

3.4 eduVPN-Profile

Das eduVPN bietet die Möglichkeit, verschiedene Profile für Nutzer anzulegen. Diese Profile können auf unterschiedlichen Nodes enden und können auf Benutzergruppen oder einzelne Benutzer beschränkt werden. Grundsätzliche Dokumentation: `https://docs.eduvpn.org/server/v3/multi-profile.html`

3.4.1 Anlegen der Profile

Die Profile werden alle auf dem EduVPN Portal unter /etc/vpn-user-portal/config.php angelegt. Die Profile bestehen immer aus einem Allgemeinen Teil:

```
    // Die Profil-ID wird nicht angezeigt ist ein technischer Wert
'profileId' => 'vpn4',

// Der Anzeigename im Client bzw. im Portal des Profils
'displayName' => 'VPN4 Replacement',

// Die URL der Node auf dem das Profil laufen soll (optional falls es keine
    ↪ Nodes gibt
'nodeUrl' => 'https://eduvpn-node-kim-01.kim.uni-konstanz.de:41194',

// Die Node Nummer
'onNode' => 2,
```

```
//Hostname der Node
'hostName' => 'eduvpn-node-kim-01.kim.uni-konstanz.de',

//DNS Server die von den Clients verwendet werden sollen
'dnsServerList' => ['134.34.3.3', '134.34.3.2'],

//Beschränkung des Profils auf IDM/LDAP Gruppen oder User (optional)
'aclPermissionList' =>
    ↪ ['cn=orz_vpn-grp-all,ou=custom,ou=idm,ou=groups,o=universitaet
    ↪ konstanz,c=de',],

//Das bevorzugte Protokoll für das Profil
'preferredProto' => 'wireguard',
```

Anschließend kommt die spezifische Konfiguration von Wireguard und OpenVPN. Zu Beachten ist hier:

- Jedes Profil benötigt einen eindeutigen IPv4 und IPv6 Range für Wireguard
- 41194 HTTPS, HTTP, OpenVPN
- Der IPv4 Range muss mindestens ein /27 sein
- Der IPv6 Range muss mindestens ein /110 sein
- Jedes Profil benötigt einen eindeutigen Port (TCP und UDP) für OpenVPN
- Diese Ports müssen von außen auf der Node erreichbar sein

Konfiguration der Protkolle:

```
        // ********************************************************************
        // * WireGuard                                                        *
        // ********************************************************************

            'wRangeFour' => '10.7.2.32/27',
            'wRangeSix' => '2001:7C0:2810:109::/110',

        // ********************************************************************
        // * OpenVPN                                                          *
        // ********************************************************************
```

```
            'oRangeFour' => '10.7.2.0/27',
            'oRangeSix' => '2001:7C0:2810:109:8000::/110',
            'oUdpPortList' => [1196],
            'oTcpPortList' => [1196],
           // 'oExposedUdpPortList' => [443],
            // TCP ports used by the VPN client to connect to the OpenVPN
            // server. If missing, or empty uses tcpPortList
            // DEFAULT: []
           'oExposedTcpPortList' => [1196],
           'oExposedUdpPortList' => [1196],
```

Danach die Profile aktivieren mit *vpn-maint-apply-changes*

3.4.2 Anlegen der Profile auf den Nodes

Auf den Nodes müssen noch die Profile eingetragen werden, die auf diese Node verfügbar sein sollen. Dazu müssen die Profil-IDs unter /etc/vpn-server-node/config.php eintragen. Danach die Profile aktivieren mit *vpn-maint-apply-changes*

4 eduVPN Erweiterungen

4.1 Geräteauthentifizierung

In Fällen, in denen beispielsweise für besonders schutzbedürftige Bereiche eine Benutzerauthentifizierung über Benutzername und Passwort nicht ausreichend oder nicht gewollt ist, bietet eduVPN die Möglichkeit der Verwendung von X.509 Client-Zertifikat-Authentifizierung für das Portal. Die Portal-Oberfläche bietet jedoch dazu keine Option, die Konfiguration und das Anlegen des Clients erfolgen händisch.
Weiteres ist unter folgendem Link zu finde:
`https://docs.eduvpn.org/server/v3/client-cert-auth.html`

4.2 ProxyGuard (neu)

In öffentlichen Netzen oder Ländern mit restriktiven Internet-Zugang werden häufig VPN-Protokolle wie Wireguard blockiert. Abhilfe kann hier Proxy-Guard schaffen, das die UDP-Pakete über HTTP(S)-Proxys verschickt. Die Konfiguration sowohl auf dem Server als auch beim Client erfolgt aktuell noch händisch; eine Integration in den eduVPN-Client ist zum aktuellen Stand noch experimentell.
Allgemeine Dokumentation: `https://docs.eduvpn.org/server/v3/proxyguard.html`

4.3 Updates (automatisch) installieren

Wurde der eduVPN-Server über das offizielle Installations-Script installiert, kann das Betriebssystem und eduVPN einfach über nachfolgenden Befehl aktualisiert werden:

```
$ sudo vpn-maint-update-system
```

(je nachdem ist ein Neustart nötig)

Alternativ kann über ein Cronjob die Updates automatische installiert werden und Server Neugestatet werden. Neue Datei erstellen:

```
    nano /etc/cron.weekly/vpn-maint-update-system
```

Mit folgendem Inhalt:

```
    #!/bin/sh
/usr/sbin/vpn-maint-update-system && /usr/sbin/reboot
```

Und zum Schluss die Datei ausführbar machen:

```
    $ sudo chmod +x /etc/cron.weekly/vpn-maint-update-system
```

Der eduVPN-Server wird danach einmal wöchentlich geupdatet und neu gestartet.

5 Fazit und Ausblick

5.1 Migration & Betrieb

Die Migration des VPN-Zugang für Studenten und Mitarbeiter erfolgte im Wesentlichen in vier Schritten: Parallel zum alten Cisco AnyConnect VPN wurde die neue eduVPN-Lösung eingerichtet und abteilungsübergreifend getestet. Danach wurde im Cisco VPN-Client eine Bannermeldung eingeblendet, die die Abschaltung ankündigte und auf die neue eduVPN-Alternative hinwies. Drei Wochen später erfolgte eine Ankündigung im Uni-Newsletter und weitere drei Wochen später wurde die alte VPN-Lösung abgeschaltet. Die Ankündigung wurde jedoch von einigen Benutzern übersehen, was zu einem (erwartungsgemäß) erhöhten Supportaufkommen führte, das sich aber relativ schnell wieder legte. Probleme bei der Client-Installation, mit dem Client selbst oder fehlende Berechtigungen traten hauptsächlich in den ersten 2-3 Wochen auf. In dieser Zeit haben wir unsere FAQ und das Handbuch immer wieder angepasst, so dass die häufigsten Fragen weitgehend beantwortet werden konnten[1]. Die Probleme, die einige Nutzer mit Wireguard in Verbindung mit ihrem Internetprovider hatten, konnten wir durch eine Anpassung der MTU auf 1280 weitgehend lösen.

Für den Admin-Remote-Zugriff war vorgesehen, dass sich die Admin-Benutzer über ihren Uni-Account mit dem Admin-VPN-Node verbinden können, wobei verschiedene Profile/Rollen zur Auswahl stehen. Hier waren aufgrund der noch fehlenden Multi-Faktor-Authentifizierung (MFA) bei Shibboleth temporär separate Accounts für die Admin-Nutzer bis zur Einführung der MFA notwendig (Vorgabe durch IT-Security). Dementsprechend mussten vor der Migration für alle Administratoren, die den Fernzugriff nutzen, separate Accounts ausgegeben werden. Mittlerweile wurde MFA ausgerollt und alle Admin-Benutzer können sich mit ihrem Benutzeraccount anmelden.
Im Gegensatz zum eduVPN für die Studierenden, bei dem einfach der komplette Adressbereich auf der Firewall freigeschaltet wurde, müssen beim Admin-VPN die einzelnen Benutzer/Gruppen freigeschaltet werden. Das Setup unterscheidet sich daher deutlich von der bereits im produktiven Betrieb befindlichen Lösung und dementsprechend wurden in einem Test-Setup die wesentlichen Funktionen getestet (Syslog an PaloAlto und Auswertung,

[1] https://www.kim.uni-konstanz.de/e-mail-und-internet/eduvpn/

UserID, LDAP). Das eigentliche produktive Setup konnte dann erst am Tag der Migration in vollem Umfang getestet werden.
Einige Admin-Benutzer haben über den (abgelösten) Cisco AnyConnect-Zugang eine feste IP-Adresse erhalten. Diese Speziallösung soll in Zukunft abgelöst werden, musste aber vorerst mit eduVPN gleichwertig abgebildet werden - dies geschieht mit einem Userprofil-basierten NAT auf der Firewall. Bei eduVPN, das erst nach der Migration vollständig getestet werden konnte, fiel auf, dass für die zusätzlichen „Fixed IP"-Profile jeweils ein Interface angelegt wurde, die entsprechende Anpassung in den nftables aber nicht vorgenommen wurde. Die Einträge mussten manuell vorgenommen werden. Dies ist zwar in der Dokumentation etwas versteckt erwähnt, ist aber bei uns im Eifer des Gefechts untergegangen. Zudem fehlten bei einigen eduVPN-Benutzern Berechtigungen, so dass diese noch in die fehlenden Benutzergruppen aufgenommen werden mussten.

5.2 Offene Punkte & Probleme

Ein offenes Problem bei eduVPN ist die Sperrung von Benutzern bzw. die Änderung der Gruppenzugehörigkeit von Benutzern. Sobald sich ein Benutzer über Shibboleth anmeldet und ein Wireguard/OpenVPN-Profil generiert - entweder per App oder manuell - ist dieses für die vordefinierte Laufzeit gültig (standardmässig: 30 Tage): wird ein Benutzer über das IDM gesperrt, könnte er theoretisch mit einem zuvor generierten VPN-Profil weiterhin den VPN-Zugang nutzen. Als Übergangslösung werden die zu sperrenden Benutzer im eduVPN-Portal manuell gesperrt und ihre zuvor generierten Profile ungültig gemacht. Langfristig wäre hier eine Lösung über die eduVPN-API und das IDM-Frontend denkbar, so dass Benutzer direkt über das IDM-Frontend gesperrt werden können.
Ein Punkt, der an sich kein Problem darstellt, aber beim Support beachtet werden muss, ist, dass die offiziellen eduVPN Apps von verschiedenen Entwicklern stammen und ähnlich aussehen, aber gewisse Unterschiede in der Funktionalität zueinander aufweisen. So verfügt die Linux-Applikation über einen automatischen Fallback auf OpenVPN-TCP, wenn eine Verbindung über Wireguard nicht hergestellt werden konnte.

Der eduVPN Client kann prinzipiell kein Start-before-Logon durchführen und diese Funktion lässt sich wohl auch nicht ohne Weiteres implementieren. Eine mögliche Lösung ist die Verwendung eines Skripts[2] im Hintergrund anstelle des Clients. Durch die Verwendung von

[2]https://github.com/FlorisHendriks/eduVPN-provisioning

Active Directory Certificate Services mit aktivierter automatischer Registrierung ruft jedes angeschlossene Gerät ein Maschinenzertifikat ab. Dieses Zertifikat wird zur Authentifizierung eines API-Aufrufs verwendet, bei dem eine WireGuard-Konfiguration abgerufen wird. Das Endgerät verwendet dann direkt die erhaltene WireGuard-Konfiguration für die VPN-Verbindung vor dem eigentlichen Login. Es sollte erwähnt werden, dass dies keine offizielle Lösung von eduVPN ist, sondern ein Workaround für das Problem, bis ggf. eine offizielle Lösung gefunden wurde.

5.3 Fazit

Es muss betont werden, dass die neue eduVPN-Lösung im Vergleich zur alten AnyConnect-Lösung aufgrund der zusätzlichen Abhängigkeiten (IDP, LDAP) und des separaten VPN-Endpunkts eine höhere Komplexität aufweist, d.h. die Implementierung und das initiale Troubleshooting sind im Vergleich zu 'out-of-the-box'-Lösungen deutlich aufwändiger. Andererseits skaliert die Benutzerverwaltung über IDM deutlich besser, und 'vergessene' VPN-Accounts (ausgeschiedene oder versetzte Mitarbeiter) gehören der Vergangenheit an. Grundsätzlich könnte die Komplexität reduziert werden, indem auf Shibboleth/SAML verzichtet wird und stattdessen z.B. ein (evtl. bereits vorhandener) RADIUS/LDAP-Server verwendet wird.
Nach mehrmonatigem Betrieb können wir feststellen, dass die Lösung zuverlässig und deutlich performanter ist als die bisherige Cisco AnyConnect-Lösung. Gleichzeitig hat sich der Supportaufwand auf der Client-Seite deutlich reduziert.Die größte Hürde bestand darin, die Studierenden und Mitarbeitenden zu erreichen und vor der Abschaltung vom alten AnyConnect-VPN auf eduVPN zu migrieren. Darüber hinaus verliefen die Migration und der Betrieb ohne größere Zwischenfälle. Vielmehr haben sich vielen Anwender positiv über den offenen VPN-Client und dessen Performance geäussert.
Das eduVPN bietet sowohl für die Studierenden als auch für die Admin-Zugänge eine flexible und herstellerunabhängige Lösung, die bei Bedarf an die eigenen Bedürfnisse angepasst werden kann - beispielsweise durch die Kommunikation mit der Firewall über Syslog oder API-Aufrufe. Obwohl derzeit kein kommerzieller Support angeboten wird, wird zeitnah über die Mailingliste oder Github/Gitlab Bugreports geholfen. Die Weiterentwicklung von eduVPN zeigt auch, dass die Entwickler auf Anfrage bereit sind, Probleme zu beheben und Funktionen zu implementieren, die ursprünglich nicht vorgesehen waren, wie z.B. die Möglichkeit, Profile nach einem erneuten Login zu verlängern.

Abkürzungsverzeichnis