The Networked Systems Security Solution for ACME

HAIDA LU | YANG YANG | ZIJIAN HUANG | ZHICHAO ZHOU

haida | yangy2 | zijianh | zhizho @kth.se

Group 10

March 8, 2021

1. Introduction

Based on the requirements of ACME, we achieve the following functions. First, there are two separate subnets, one stands for Stockholm and another stands for London. The communication between these two subnets is encrypted and authenticated by the VPN tunnel. A secure wireless connection is provided based on FreeRadius. The FreeRadius server is configured at the Stockholm headquarters, while the client is set on the router at the London branch. For employees of the company, they can access the internal servers by wireless connection and wired connection. At home, employees can use the remote VPN connection to access the company servers. In this way, they also need two-factor authentication based on Google Authentication. We also provide an IDS in the VPN Access Server, which will log the access to this server. Then we set up a web server that uses Apache2. On this web, we implement a file transfer system with OwnCloud. Employees can transfer their files to others on this platform as authenticated users.

2. Implementation

In this section, the specific implementation of our secure network system will be explained in detail. Our system is built on three virtual machines which are VM1, VM2, and VM3. Their specific functions are shown below:

Table 1 Virtual Machine Functions

Virtual Machine	Implementation Funciton
VM 1	OpenVPN Server, Firewall, IDS
VM 2	Web Server, FreeRadius Server
VM 3	OpenVPN Client (work as forward client)

2.1. OpenVPN

To realize the secure communication between Stockholm central subnet and London branch subnet, we settle VPN tunnels between these two subnets via OpenVPN.

With VPN all communications between Stockholm and London are encrypted and authenticated. OpenVPN also provides client GUI on almost all platforms, including Windows, macOS, Linux, Android, and iOS. Users can remote access corporate internal networks by importing personal ***.opvn** file (a kind of configuration file for OpenVPN) to the OpenVPN client GUI. Furthermore, OpenVPN supports Google's two-factor authentication (Google 2FA). After enabling this authentication, users should use the account and Google 2FA to log in.

2.1.1. OpenVPN Access Server

OpenVPN Access Server located in VM1 with IP address 192.168.1.151. OpenVPN Access server has an admin GUI web page. The administrator can access it via ***https://192.168.1.151:943/admin***.

The configurations of the access server are shown below:

Table 2 OpenVPN Access Server Configurations

Server Configuration	
Public Network	157.174.210.112
Vpn IP pool	172.27.224.0/20
TCP Port	443
UDP Port	1194
Web Server Port	943

2.1.2. Authentication

OpenVPN Access Server also acts as CA. The administrator should upload the server's private key and self-signature certificate. Also, the administrator can provide a concatenated list of CA certificates that validates the access server certificate. The access server will create certificates for each user. The certificate can be revoked on Access Server.

2.1.3. User management

OpenVPN Access Server GUI provides a user management page, where the administrator can change IP address, access control, VPN Gateway, and DMZ settings. Setting an initial username and password, the administrator could allow the user to change the password by themselves. Access server will generate a unique ****.ovpn*** configuration file for each user. Users can download it via the OpenVPN user webpage. In our system, it is ***https://192.168.1.151:943***.

2.1.4. OpenVPN Client

We settle the OpenVPN client on VM3 based on the Linux system. Importing the configuration file in the ***/etc/openvpn/*** directory, VM1 and VM3 can access each other securely via VPN.

2.1.5. Site-To-Site VPN Routing

To make the Stockholm subnet and the London subnet connected using one OpenVPN tunnel, we build a site-to-site VPN routing by implementing static routing. As VM1 and VM3 have already connected, we make them serve as a VPN gateway in their subnets.

On Router 1, we add such static router rules:

Network 172.27.224.0 with subnet mask 255.255.240.0 through gateway 192.168.1.151

Network 192.168.50.0 with subnet mask 255.255.255.0 through gateway 192.168.1.151

On Router 2, we add such static router rules:

Network 172.27.224.0 with subnet mask 255.255.240.0 through gateway 192.168.50.122

Network 192.168.1.0 with subnet mask 255.255.255.0 through gateway 192.168.50.122

Now, the Stockholm subnet and the London subnet are connected via the VPN tunnel.

2.1.6. Google Authenticator Multi-Factor Authentication

On concern about security, we enable the Google Authenticator Multi-Factor Authentication on OpenVPN Access Server. After that, employees will see a QR-code for the first time logging into the user GUI with their account. They should associate their account with Google Authenticator. And every time they try to log in to OpenVPN, the 6 bits code created by Google Authenticator is required.

2.1.7. Remote Access

As OpenVPN provides client GUI on various platforms if employees want to access corporate networks via personal devices at home. They can download OpenVPN GUI and import their ***.ovpn** file, type in their account, and the 6 bits code. The VPN connection will establish.

2.1.8. Firewall and IDS

OpenVPN also generates firewall rules automatically on VM1. Generally speaking, the rule sets chain INPUT and FORWARD to DROP by default, and accepts traffic for VPN tunnel communication and account authentication requests. The attackers cannot remote access the access server (e.g. access via ssh). And we also deploy the IDS to detect the intrusion. IDS is introduced in part 2.4.

2.2. Web Server and File Sharing

We have utilized OwnCloud, a comprehensive secure collaboration platform, to set up a Web server for data storage and file sharing, and transferring [4].

At first, we attempted to configure password authentication for Apache2, but it was difficult to create a Web server that stores files based on the default config. So we choose OwnCloud and directly configure it on Ubuntu on our VM2 as our Web server and file transferring tool instead of developing a new server or App by ourselves.

Apache2 HTTP server is installed first [5] since it is necessary to set up a management Web page for OwnCloud. Also, a new virtual host file is created for OwnCloud which contains the Server information about OwnCloud and is used to replace the default virtual host config file. The virtual host file is located in:

/etc/apache2/sites-available/fosslinuxowncloud.com.conf.

Besides Apache2, MySQL is required as well. We create a database and a user, granting all privileges on the database to the user. Then we visit the IP address of VM2 (which is set as the address of our OwnCloud server) and register an admin account based on the configuration above.

To connect to the Web server, users should connect to the internal networks at the Stockholm headquarters first. For users in the London branch or at home, they need to use the VPN.

Users can log in to the Web server by either directly visiting the IP address of our VM2 (i.e. ***192.168.1.16***) on browsers or have the client application of OwnCloud. No matter which method they choose, a user account created by the admin account is necessary. Files can be uploaded to the server by anyone logging in, but users themselves can decide who has access to the files. So this Web server can also work as an internal file sharing and transferring tool. The process is secure since only those who have access to the internal networks can upload or download files from the server.

2.3. FreeRadius

We set up a Radius server on our VM2 based on FreeRadius 3.0 [6], and simultaneously modify the encryption method of the router to WPA-EAP at the London branch.

To be exact, the FreeRadius packages are first installed on VM2. We modify the file ***/etc/freeradius/3.0/mods-available/eap*** to enable only the EAP-TLS protocol. Also, we add a new server config file ***/etc/freeradius/3.0/sites-enabled/mynetwork*** and set both the authorize section and authenticate section to contain EAP mode. It is also required to modify the file ***/etc/freeradius/3.0/clients.cnf*** on the FreeRadius server. We comment on the default config about the server localhost and add a new client entry for our router at the London branch. The router is regarded as a wireless access point (WAP) and has already been assigned a static IP address. So this IP address is also set in the client entry of the clients.conf file to indicate the router as our FreeRadius client. For the router, the encryption method is set to WPA2-EAP, and the server address is the IP address of our FreeRadius server VM2, i.e. ***192.168.1.16***.

Having all the configuraton above, we still need to create certificates for the secure connection, including CA certificates, server certificates, and user certificates. In particular, we also create Android-specific certificates (with a ***.p12*** suffix) for users who try to connect with an Android mobile phone. Users just need to install this certificate on their phone and connect the Wifi by TLS encryption with the certificate file if the FreeRadius server is running (with command ***freeradius -X*** on our VM2) and "ready to process requests". The connection test has been successfully done on the Redmi mobile phone.

2.4. Intrusion Detection System

There is an intrusion detection system (IDS) deployed in the VPN access server, monitoring traffic in the network so that when attackers try to infiltrate our corporate network, they can set off the alarm for intruders.

We have utilized Snort which can filter out using different kinds of policies and be able to identify specific attacks to implement our IDS. First, the snort is set to listen on the Stockholm interface enp0s3 and we configured the Stockholm address range for the local network. To run Snort, we have made some modifications. For instance, we replaced the HOME_NET "any" with the CIDR notation address range of the Stockholm network. Also, all the Snort rules should be updated. After running Snort, when attackers try to infiltrate (i.e. ping, nmap, etc.), it will scroll the output in the terminal and the information will be logged in a file too.

3. Topology

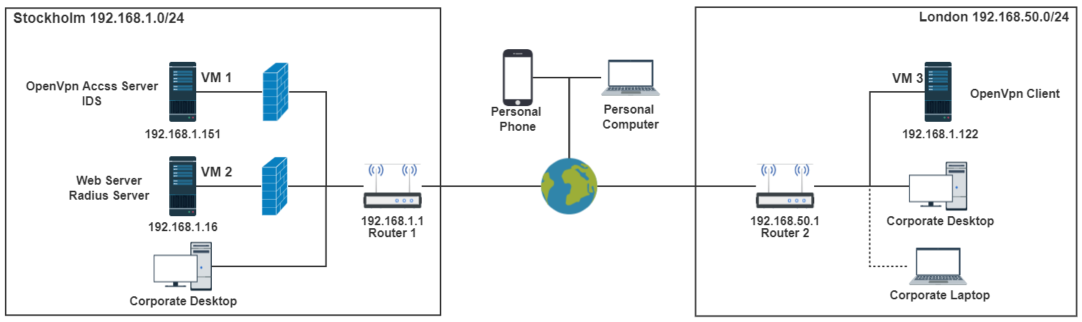

Figure 1 Network Topology

Appendix

GitHub Link: https://github.com/Lance-Huangzj/EP2520_ACME_Project

Reference

OpenVPN:

[1] https://openvpn.net/quick-start-guide/
[2] https://openvpn.net/linux-video-tutorials/
[3] https://openvpn.net/vpn-server-resources/site-to-site-routing-explained-in-detail/

FreeRadius:

[4] https://www.ossramblings.com/RADIUS-3.X-Server-on-Ubuntu-14.04-for-WIFI-Auth

Apache2:

[5] https://www.digitalocean.com/community/tutorials/how-to-install-the-apache-web-server-on-ubuntu-20-04

OwnCloud (Web Server):

[6] https://www.fosslinux.com/8797/how-to-install-and-configure-owncloud-on-ubuntu-18-04-lts.htm

IDS:

[7] https://upcloud.com/community/tutorials/install-snort-ubuntu/