|  |  |
| --- | --- |
| Project Name: | OpenVPN Implementation |
| Prepared by: | Thioro Fall |
| Date: | 3/28/2021 |

|  |
| --- |
| Overview |
| **OpenVPN** is one of the most popular open-source VPN clients. The VPN clients are built on the OpenVPN protocol, which is itself open source, has been audited several times, and is generally considered to be the most secure VPN protocol available. Even most closed source VPN clients tend to use OpenVPN as the default VPN protocol.  **OpenVPN** is a [virtual private network](https://en.wikipedia.org/wiki/Virtual_private_network) (VPN) system that implements techniques to create secure point-to-point or site-to-site connections in routed or bridged configurations and remote access facilities. It implements both [client and server](https://en.wikipedia.org/wiki/Client-server_architecture) applications.  OpenVPN Access Server is a VPN solution designed specifically for businesses, built on the open-source project. OpenVPN Access Server consists of three major components:   1. OpenVPN Server 2. Admin Web UI 3. OpenVPN Connect  OpenVPN Server: the VPN server is the underlying component in OpenVPN Access Server. It handles all the background work: routing; tunneling; encryption; user management; authentication; etc.Admin Web UI: the Admin Web UI provides you with a powerful and easy-to-use web-based admin site. It makes VPN management and configuration simple enough for anybody (with or without Linux knowledge). You can manage options such as layer 2 or layer 3 routing, user permissions, server network settings, authentication, and web server certificates.OpenVPN Connect: OpenVPN Connect is the client interface to connect with the VPN server. Your server has pre-configured clients available for download. Or users can download Connect directly from our site and import profiles by simply entering a URL and their credentials. The main features of the Access Server are:   * OpenVPN Access Server installs on a Linux OS and lets clients connect from Windows, Macintosh, iOS, Android, and Linux systems. * Available as installation package for a number of Linux operating systems so you can install on your own hardware, or you can use prepared images for virtual machines and deploy there, or use cloud platforms like Amazon AWS with our official images that can be launched from the AWS Marketplace. * There is a built-in authentication system with web based management. External authentication systems like PAM, LDAP, or RADIUS, can also be used. The authentication system is extensible but this requires programming knowledge (Python). * Fully automated VPN certificate management and provisioning built-in. External PKI is also possible, so you have full control over your own already existing PKI and use our VPN solution with it. * VPN tunnels are secured with OpenVPN protocol with TLS authentication, credentials, certificates, and optional MAC address lock and multi-factor authentication. Additional checks can be added on using post\_auth plugins but this requires programming knowledge (Python). * Multi-factor authentication is supported in various forms. For example, Google Authenticator is built-in, and Duo Security can be added with a post\_auth plugin. * Access control rules let you specify which user or group has access to which IP addresses or subnets, and if VPN clients can contact each other or not. * Full-tunnel and split-tunnel redirection are possible (all VPN client Internet traffic goes through the VPN tunnel, or only specified traffic). * Professional support available through OpenVPN Inc. online support ticket system manned 24/7 by professionals around the world.   After launching your server, your end users can install pre-configured clients directly from your Access Server’s Client UI.  The first step is to choose the deployment platform you prefer. For example, for Linux Deployment Overview: (Linux)**1. Determine the network configuration and IP address for the server** **2. Obtain an activation key****3. Download and install the OpenVPN Access Server package file****4. Run the initial configuration**5. **Set a password for the OpenVPN admin account****6. Complete configuration in Admin Web UI****7. User authenticates to the Connect Client** **8. User connects to VPN**  You can connect via OpenVPN both over TCP and UDP:  -for OpenVPN TCP connections use port 443. -for OpenVPN UDP connections use port 1194.  The OpenVPN protocol itself functions best over just the UDP protocol. And by default, the connection profiles that you can download from the Access Server are preprogrammed to always first try UDP, and if that fails, then try TCP. |
| Hardware & Software Requirements, Possible Integrations |
| **Hardware requirements** Access Server hardware requirements are largely specific to your bandwidth utilization. For example, if you run a VPN server for the purpose of connecting to a single web server through the VPN tunnel, then the requirements are much less when compared to running a server that redirects all Internet traffic. Traffic passing through a VPN connection utilizes processing capacity for encrypting and decrypting on both the client and the server side. To correctly estimate the sizing of your Access Server, you must estimate how much sustained bandwidth you need to route through the VPN server, and estimate the CPU size accordingly. Memory size and disk space are more predictable. Processor Almost all modern CPUs support AES-NI to speed up AES processing. Access Server automatically uses AES-NI for the default AES-256 encryption. A non-AES-NI CPU severely lowers the speed of the encryption/decryption process. As a very rough estimate, you should quadruple your estimates for CPU sizing if AES-NI is not supported on your intended deployment platform.  These estimates are only a rough guide, because they don’t account for variations in capabilities between different CPUs. Nor do they account for throttling as a result of physical limitations such as overheating, or virtual limitations such as running on a shared platform (like AWS with burstable instances).  As a rule of thumb, you should assume that on a modern CPU with an AES-NI chipset you need approximately 12MHz for each megabit per second (Mbps) transferred in one direction. Access Server can use all available CPU cores on a system, so for example a modern 4-core system at 3GHz would count as 12,000MHz, which equates to approximately 1,000Mbps maximum throughput.  It is possible to use the OpenSSL program to run encryption tests to see what level of data encryption processing speed your CPU is capable of. However, this does not tell the full story because bottlenecks can exist elsewhere, for example in network connectivity, packet signing, packet loss, and others. Depending on what your OpenVPN client is capable of, AES-256-CBC and AES-256-GCM may be used. Note that GCM is easier on a CPU than CBC. To keep things simple, the estimates provided here are based on experience with standard encryption settings on somewhat recent systems. Memory Memory requirements are dependent on the number of connected devices and the level of NAT traffic your VPN server needs to process. At a minimum, you must start with 1GB of memory, and add approximately 1GB for each 150 connected devices. Again, note that this is a rough estimate but should serve as a basis for estimating memory size. Bandwidth Bandwidth requirements are completely dependent on how much total data you want to route through your VPN tunnels. If you have a server with a 1Gbps network connection, and you have 100 connections, that means 10Mbps per user if they all use the full potential bandwidth at the same time. Usually, however, not everyone requires that level of simultaneous bandwidth.  For example, if only half of the users are connected and the other half are idling, then that means the usage is about 20Mbps per user. Unfortunately, there is no way for us to estimate how your users will use the connection; you will have to make that estimation. Hard Disk Hard disk requirements are minimal. The only data that are necessary to store on disk are connection and program logs, and user certificates and settings. The logs may accumulate over time and should be rotated or cleaned using the instructions on our [logging and debugging options page](https://openvpn.net/vpn-server-resources/logging-and-debug-flag-options-for-access-server/). 16GB of disk space should be more than enough. **Software requirements** OpenVPN Access Server is compatible with these 64-bit Linux operating systems:   |  |  | | --- | --- | | Ubuntu\* | 20.04 LTS 18.04 LTS 16.04 LTS | | Debian | 10 9 | | Red Hat Enterprise Linux | 8 7 | | CentOS | 8 7 | | Amazon Linux 2 | Amazon Linux 2 |   \*Note: For Ubuntu, only long-term support (LTS) releases are supported.  One of the requirements for running Access Server is that you do not use a static-compiled kernel. Static-compiled kernels can’t load certain modules for network traffic management (mangling, iptables, NAT), which are necessary for Access Server to function. For example, OVH/Kimsufi servers can be provided with a static-compiled kernel on their Linux OS offerings, which prevents the Access Server web interface from functioning. Similarly, some VPS systems such as older OpenVZ implementations can deny access to certain kernel modules.  Otherwise, there are no particular software requirements. You need a server (virtual or on premise) with a supported Linux operating system. The server can be a minimal install, a server installs, or a full desktop installation. **Infrastructure recommendations** When you set up an on-premise Access Server with your hardware, you can make a fairly good estimation of how much bandwidth you can transfer through the VPN tunnels, and how many users can be connected at the same time. But with virtual systems, such as virtual machines and virtual private clouds (Google Cloud, Amazon AWS, Microsoft Azure, Oracle, Digital Ocean, and so on), it may be much more difficult to accurately determine how much processing power is available to you. Most cloud systems share their hardware in such a way that you may not have full access to the CPU. Your access speeds may be throttled and you may not realize it at first. Often, peak utilization is allowed for a short while and then after a time the speed is throttled down. |
| Additional Administrative Considerations |
| **Secure the root user account** You should aim to have a situation where the root user can only be used directly on the console, and not over the network, and obviously with a very secure password. Additionally, you should have your own user account with a very secure password that you can use to log on over the network with, and has the ability to use sudo to run commands as root user. To make things even better you should set up an SSH key-pair for user login under your own user account, instead of simple username+password authentication. **Secure the OpenVPN administrative user account** By default, the OpenVPN Access Server comes configured with a user account called **OpenVPN** without a password set on it. That by itself is not immediately a security issue because an account without a password set on it normally cannot be used to log on at all, especially on the images we provide. You are expected to make your own password and set it on the OpenVPN user account to start logging in to the Admin UI and setting things up on the Access Server. So that is not the problem, but having an account with a predictable user name is of course not a good thing to have, especially when it’s facing the Internet. And the OpenVPN user account is also a bootstrap account meaning it has special access privileges. For example, it can bypass Google Authenticator and the authentication failure lockout policy. Therefore, we recommend that one of the first things you do after setting up the OpenVPN Access Server is to create a new user for yourself and give it admin privileges. That will then be your administrative user account from that moment on. You can do this from the Admin UI under User Permissions by adding a user there. If you use local authentication, you can set a password for the new account there as well. If you are using an external authentication system like PAM, RADIUS, or LDAP, remember to also add the account there as well so you can actually use it to log on to the Admin UI. Obviously test this first before proceeding with the next steps. **Installing an SSL certificate on the web interface** By default, the OpenVPN Access Server comes with a self-signed certificate to at least get things working. Such a self-signed certificate cannot be automatically verified by your web browser or an OpenVPN client program to check if the server it is contacting is really your server, and not some other server pretending to be. SSL certificates allow for the web browser to automatically verify if you are connecting to the real server, and to automatically trust the server so that the web interface will not show a warning message about not being able to validate the authenticity of the server, but instead show a nice green padlock icon in the address bar in the browser.  This requires that your OpenVPN Access Server is set up with an FQDN DNS name that points to the public IP address that the Access Server can be reached at from the Internet, and that this FQDN DNS name is configured correctly in the Admin UI under Server Network Settings in the Host name or IP address field. We recommend that you set up this FQDN DNS name in all cases, not only because it is required for an SSL certificate to function properly, but also because if ever in the future you change the IP address of your Access Server, for example if you move it to another Internet connection, then you need only update the DNS record and all clients will be able to find the server again. If however you configure it to IP basis only, then you will have to reinstall all your clients if you move your server to another public IP address. **Hardening the web server cipher suite string** The web server built into the Access Server by default uses HTTPS SSL encryption. This secures the connection between the web browser and the web server, so that any credentials you enter on the web interface cannot be intercepted by a “man-in-the-middle” attack or be seen in plain text on the network connection. Instead that information is all nicely encrypted. The cipher used to encrypt this information is one that is agreed upon by the web server and the web browser. The server offers a number of ciphers that it allows to be used, and the web browser then picks (usually) the best one of those that it can support and uses that to encrypt information. The list of ciphers that the web server allows is called the cipher suite string. By default the cipher suite string that the Access Server comes shipped with is reasonably secure, but not overly so. There are some older ciphers allowed to offer compatibility for older web browsers and operating systems, like Windows XP for example. In most cases though you will probably want to run the web server through its paces using an online SSL security checker like [Qualys SSL Labs SSL Server Test](https://www.ssllabs.com/ssltest/) to see what grade your current settings get and then adjust the cipher suite string to eliminate weak ciphers and thereby improve the grade and thus the security of your web server. This can have as consequence that older browsers and operating systems can’t connect to the web interface anymore, though. **Going beyond recommended security procedures** Some of our customers do not want the web services visible on the Internet, but only want the OpenVPN daemons reachable for VPN tunnel termination. We advise against doing this because of the fact that managing the Access Server without a web service makes things a lot more difficult. You would then have to rely on using the [command line interface tools](https://openvpn.net/vpn-server-resources/command-line/) to manage the Access Server settings, users, and certificates, and also the distribution of the required connection profiles to the users. Having the web services available makes this a lot easier. Furthermore, the OpenVPN Connect Client is tied into the web services of the Access Server using a secure XML-RPC connection over SSL. In short, this allows any user with valid credentials to log in with the OpenVPN Connect Client, instead of having to install separate user-locked connection profiles for each and every user that needs to log in from a client computer. Making the web services unreachable from the Internet breaks this functionality and forces you to use user-locked or auto-login profiles only. In short, you would end up breaking some of the designed functionality, and force you to do some extra work.  However, if you really want to, you can choose to for example only allow ports TCP 443 and UDP 1194 default ports for the OpenVPN daemons from the Internet through your firewalls to your Access Server installation, and then disable the **service forwarding** options for the client web UI and the admin web UI in the **Server Network Settings** page. Those two actions together will make the web interface unreachable from the Internet but still allow incoming user-locked and auto-login connection profile based OpenVPN tunnels to make a connection. But server-locked profiles will not be able to connect anymore. |

|  |  |
| --- | --- |
| Project: | Duo 2FA Implementation |
| Prepared by: | THIIORO FALL |
| Date: | 4/2/2021 |

|  |
| --- |
| Overview |
| Duo 2FA prevent data breaches with strong authentication. Two-factor authentication is the best way to protect your sensitive data from theft.  Two-Factor Authentication (2FA) works by adding an additional layer of security to your online accounts. It requires an additional login credential – beyond just the username and password – to gain account access, and getting that second credential requires access to something that belongs to you.  Two-factor authentication describes an approach, not a method. Many different methods exist to secure your account with two-factor authentication.  There are three main types of two-factor authentication:   1. Additional login credentials only the account holder should know. This includes things like security question answers and PIN numbers. 2. Devices the account holder owns that receive additional login credentials. This most commonly takes the form of a security token, mobile phone app, or tablet device app. 3. Biometric login credentials unique to the account owner. This includes retina scans and fingerprints.   You can determine which 2FA method works best for you. Companies often prefer the device method, because employees may feel biometric options [violate their privacy](http://www.m2sys.com/blog/retail-point-of-sale/do-employees-have-a-right-to-refuse-enrollment-in-a-biometric-system/). Individuals often find it less cumbersome to secure devices they own with biometric methods, because they don’t require you to carry around multiple devices |
| Hardware & Software Requirements, Possible Integrations |
| **SMS 2FA** SMS two-factor authentication validates the identity of a user by texting a security code to their mobile device. The user then enters the code into the website or application to which they're authenticating. **TOTP 2FA** The Time-Based One Time Password (TOTP) 2FA method generates a key locally on the device a user is attempting to access. The security key is generally a QR code that the user scans with their mobile device to generate a series of numbers. The user then enters those numbers into the website or application to gain access. The passcodes generated by authenticators expire after a certain period of time, and a new one will be generated the next time a user logs in to an account. TOTP is part of the Open Authentication (OAUTH) security architecture. **Push-Based 2FA** Push-based 2FA improves on SMS and TOTP 2FA by adding additional layers of security, while improving ease of use for end users. Push-based 2FA confirms a user's identity with multiple factors of authentication that other methods cannot. Duo Security is the leading provider of push-based 2FA. **U2F Tokens** U2F tokens secure two-factor authentication by using a physical USB port to validate the location and identity of a user attempting to login. To use a U2F token, a user inserts the token into their device and presses the button located on the top of the device. Once the token is activated, the user enters their PIN and gains access to their accounts. **WebAuthn** Created by the FIDO (Fast IDentity Online) Alliance and W3C, the Web Authentication API is a specification that enables strong, public key cryptography registration and authentication. WebAuthn (Web Authentication API) allows third parties like Duo to tap into built-in capabilities on laptops, smartphones, and browsers, letting users authenticate quickly and with the tools they already have at their fingertips. |
| Additional Administrative Considerations |
| *Y*ou should never use just a username and password to protect your account. The number of corporate security breaches in recent memory proves it’s too easy for hackers to gain access to your accounts.  Using text messages, authenticator apps, or biometric methods are better than nothing, but you should also go beyond that and follow these 2FA best practices:   1. **Don’t use your personal phone number for text 2FA authentication.** Phone carriers are notorious for getting tricked into changing account details by clever hackers. Instead, set up a dedicated Google Voice number that you can always keep and that a phone carrier cannot change. 2. **Don’t use email-based account resets.** It’s convenient to reset your accounts by email. This is because it makes it very easy for a hacker to bypass other 2FA procedures you’ve put in place and get at the account with just a username and password. 3. **Use a combination of authentication methods.** You can secure many accounts with more than one 2FA method. And the more 2FA methods you use, the more secure your information is. |