Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 198 (2022) 632–637

Procedia
Computer Science

www.elsevier.com/locate/procedia

The second International Workshop of Innovation and Technologies

(IWIT 2021)
November 2-5, 2020, Madeira, Portugal

Automated VPN configuration using DevOps

Lotfi Firdaouss*, Bahnasse Ayoub, Belkadi Manal, Yazidi Ikrame

ENSAM Casablanca, university Hassan II of Casablanca, Morocco

Abstract

Enterprise networks are becoming increasingly sophisticated and large in scale due to the critical need for interconnectivity. For the interconnection of sites, VPN technology is essential. Indeed, this technology allows a partially or completely meshed connection between the various sites in a secure way. IPsec is one of the most widely deployed VPN technologies due to its many advantages, including data confidentiality, integrity and authentication. However, implementing this technology requires considerable technical expertise given the diversity of gateway manufacturers that a company may have, advanced engineering given the set of technical parameters that a VPN tunnel may have for its proper functioning, and caution when setting up a large-scale network given that a simple error may prevent the creation of tunnels. Taking these limitations into account, the automation of IT infrastructures has become indispensable, known as DevOps, which promotes continuous communication, collaboration, integration, visibility and transparency between the teams responsible for application development (Dev) and those responsible for IT operations (Ops). With infrastructure automation, networks are becoming easier to manage, diagnose and configure. This paper proposes a new architecture that automates the deployment of VPN tunnels via a web-based graphical interface. This architecture is adapted with a variety of equipment manufacturers and delivers configurations generated via an SSH channel in an automatic way.

Keywords: Devops, VPN, Agile, Automation, Security, Waterfall model

* Corresponding author. Tel.: +212-6-17-24-25-08;
E-mail address: lotfi.firdaouss@ensam-casa.ma

10.1016/j.procs.2021.12.298

1. Introduction

Before DevOps, the model used for software development was the “WATERFALL” model [1]. This model is best suited when all the requirements are present beforehand. Hence, after the product planning phase, we move to the product development phase and then the testing phase and then the product building phase without any parallel work. Many issues were encountered during the development phase of the software, since the development and operation teams worked independently and a lot of time got wasted unnecessarily.

That is until the philosophy of DevOps [2] has seen the light of the day. DevOps as its name defines it consists of two main parts; “Dev” for development and “Ops” for operations. In short, DEVOPS is a set of practices which combines the software development along with the operations which result in better and faster software development cycle with high software quality and enables Agile Development. In fact, DevOps influences the application lifecycle throughout its plan, develop, deliver, and operate phases. Each phase relies on the others, and the phases are not role-specific. In a true DevOps culture, each role is involved in each phase to some extent.

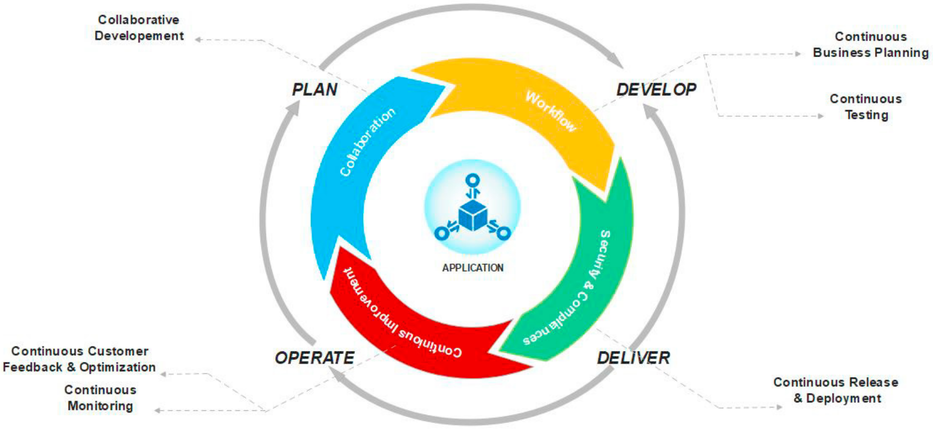


Figure 1 Application life cycle with DevOps

In the planning phase, DevOps teams ideate, define, and describe features and capabilities of the applications and systems they are building. As for the development phase, the latter involves the actual coding of the software as well as the testing, reviewing, and the integration of code by team members. In the delivery phase, teams define a release management process and set automated gates that move applications between stages until they’re made available to customers. Automating these processes makes them scalable, repeatable, and controlled. The operating phase involves maintaining, monitoring, and troubleshooting applications in production environments, in order to ensure system reliability, high availability, and reinforce security and governance.

DevOps has outperformed Agile methodology by overcoming the disconnection between development and operations teams thus allowing the laters to manage code throughout the software development life cycle. Apart from this, continuity and automation are two big parts of DevOps including things like Continuous Integration, Continuous Delivery, and Continuous Testing. These include, for example Docker which is a Linux-based open-source platform that focuses on containers, meaning you package the software with its dependencies and ship it all at once. We can also mention Git, a highly popular open-source DevOps tool. It allows us to track the progress of the development work and coordinate work among team members.

The paper is organized as follows; the abstract is positioned in the first place, then come the introduction where we discussed concepts like the waterfall model, DevOps, its lifecycle and also some technologies used in DevOps. Moving on we’ll be treating the subject of VPN and its different protocols. In the next step, we’ll present you with related works. And afterwards, we’ll be introducing the proposed approach and explaining the latter to finally finish with a conclusion.

2. VPN

Since the first use of the Internet, there has been a movement to protect and encrypt internet browser data. Hence the emergence of VPN (Virtual Private Network) technology [3]. Over time, the unlimited ambition of hackers to infiltrate all available devices and connections have increasingly motivated Internet users to use VPNs. VPN describes the opportunity to establish a protected network connection when using public networks. VPN hides your internet protocol (IP) address by letting the network redirect it through a specially configured remote server run by a VPN host, so your online actions are virtually untraceable.

Nowadays, security [4] and privacy threats have become increasingly complicated, which lead to the continuous evolution of VPN technologies, as well as the growing sophistication of user demands. The main purpose of VPN became to provide various security elements such as authenticity to prevent unauthorized users from accessing the VPN, confidentiality such that even if the network traffic is sniffed at the packet level, only encrypted data is seen and data integrity to detect any instances of tampering with transmitted messages.

IPsec (Internet Protocol Security) is a set of tunneling protocols, which use algorithms to transport data over an IP network in a secure manner. The IETF (Internet Engineering Task Force) defines it as a framework of open standards to ensure private and secure communications over IP networks, through the use of cryptographic security services. IPsec is closely related to the IPv4 and IPv6 protocols. It allows authentication and encryption of data, so as to ensure confidentiality and integrity of data flows.

Companies tend to use VPN due to covid-19 teleworking becoming a necessity to ensure public safety. Indeed, remotely working requires a set of conditions and measures to ensure that exchanges will be performed meeting security needs. However, deploying VPN tunnels is a tedious task that consume a lot of time and energy, e.g., deploying VPN between 10 sites with a full meshed manner requires at least more than 3 hours of parametrizing taking into account the best practices. Thus, it is important to automate the deployment of VPN tunnels thanks to DevOps concepts. This will allow companies to scale better and to ensure that security policy will be respected among all tunnels and between all remote clients.

3. Related works

IT infrastructure automation is an active research area. Researchers have opted for this approach to improve existing networks. The scientific paper [5] proposes a new architecture to automate the scanning of network vulnerabilities and to propose countermeasures thus improving network stability. The authors used the Python language as SBI. Tim Lackorzynski et al [6] have shown that VPN deployment does not depend primarily on the security of the technology but also on the nature of the industry. For these reasons, mastering all configurations of all VPN technologies is a difficult task for an IT manager to master. Ayoub Bahnasse et al [7] proposed an architecture that generates several VPN topologies ready to be simulated. The adoption of an architecture in the form of agents or modules allows to guarantee an agility [8] of the development, however the proposed solution is not going to be useful for the production more than for the study and the evaluation of the performances.

4. Proposed approach

The figure 2 contains the proposed architecture of the SDN Hybrid model allowing equipment from different providers to be connected through VPN tunnels configured automatically. The application Plane simply represents the web GUI provided to the end user, in order to fulfill details required for the deployment of the VPN tunnels. While the Control Plane defines the model operation. Different operating phases of our model are illustrated in the UML sequence diagram of Figure 3. These phases will be explained in more details later on. As for the data Plane, it's the layer that holds different physical equipment forming the necessary infrastructure to carry network traffic.

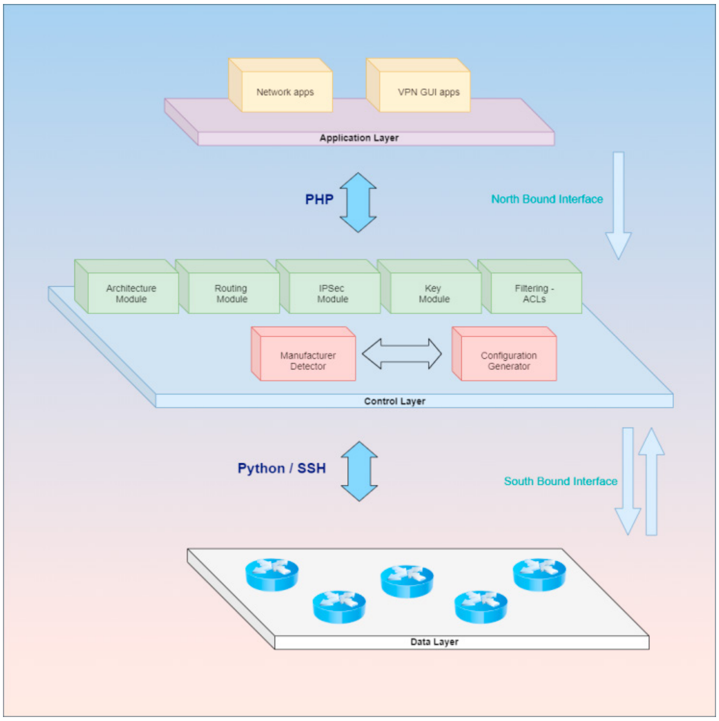


Figure 2 Hybrid SDN Architecture for VPN Tunnel Management

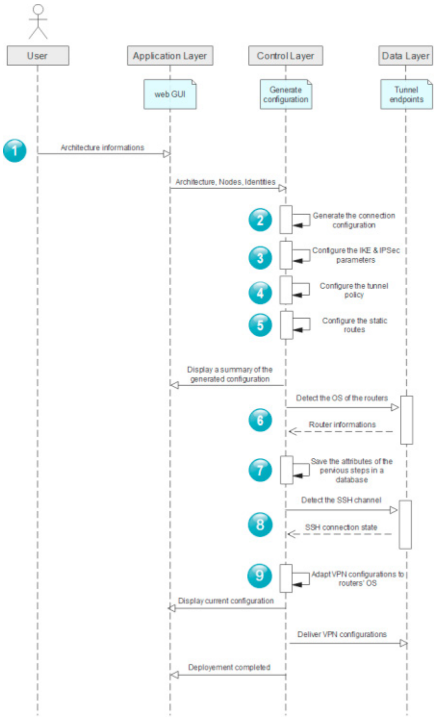

User
Application Layer
Control Layer
Data Layer
web GUI
Generate configuration
Tunnel endpoints
1
Architecture informations
Architecture, Nodes, Identities
2
Generate the connection configuration
3
Configure the IKE & IPSec parameters
4
Configure the tunnel policy
5
Configure the static routes
Display a summary of the generated configuration
Detect the OS of the routers
6
Router informations
7
Save the attributes of the perivous steps in a database
Detect the SSH channel
8
SSH connection state
9
Adapt VPN configurations to routers' OS
Display current configuration
Deliver VPN configurations
Deployement completed

Figure 3 UML sequence diagram of the proposed model

- *Phase 1: Sending architecture information*

The VPN architecture module allows the user to enter required architecture information for the deployment of the VPN tunnel.

- *Phase 2: Generate the connection configuration*

In this phase, the user provides the necessary details (Ip address, user name, password, interface) to establish a connection to a remote peer through SSH

- *Phase 3: Generate the IKE & IPsec parameters*

In this phase, firstly IKE parameters provided by the user such as the encryption algorithm, hashing algorithm, Diffie-Hellman group, and the lifetime are used to set up a secure, authenticated communications channel between the two concerned routers. In fact, IKE uses the Diffie–Hellman key exchange protocol to set up a shared session secret. Actually, the Internet Security Protocol (IPsec) which is responsible for negotiating security associations (SAs) between peers uses IKE to set up the channel mentioned earlier.

Secondly, the inputs provided by the user regarding the IPsec protocol (ESP/AH) and other relevant details (mutually agreed-upon keys and algorithms to be used by both parties) are used to prepare the SAs that are mandatory to establish a VPN connection/tunnel.

- *Phase 4: Configure the tunnel policy*

In this phase, the filtering module provides the user with the choice between different types of traffic (IP, HTTP, FTP, SMTP), or to specify manually the port number with the associated protocol (TCP/UDP) in order to define the tunnel policy. The latter will hold all necessary ACLs for VPN configuration.

- *Phase 5: Configure the static routes*

Static routes are being configured from provided private and public Ip addresses of the concerned routers.
And then a summary of all entered configurations is presented to the end user for verification purposes.

- *Phase 6: Detect the OS of the routers*

In our solution, the “Manufacturer detector” module is the one responsible for router’s OS detection. This module is mainly based on the SNMP protocol. The latter detects the OID of the concerned device -in our situation, the concerned router - which is its unique object identifier that provides the full name and version identification of the system’s hardware type, software operating system, and networking.

- *Phase 7: Save the attributes of the previous steps in a database.*

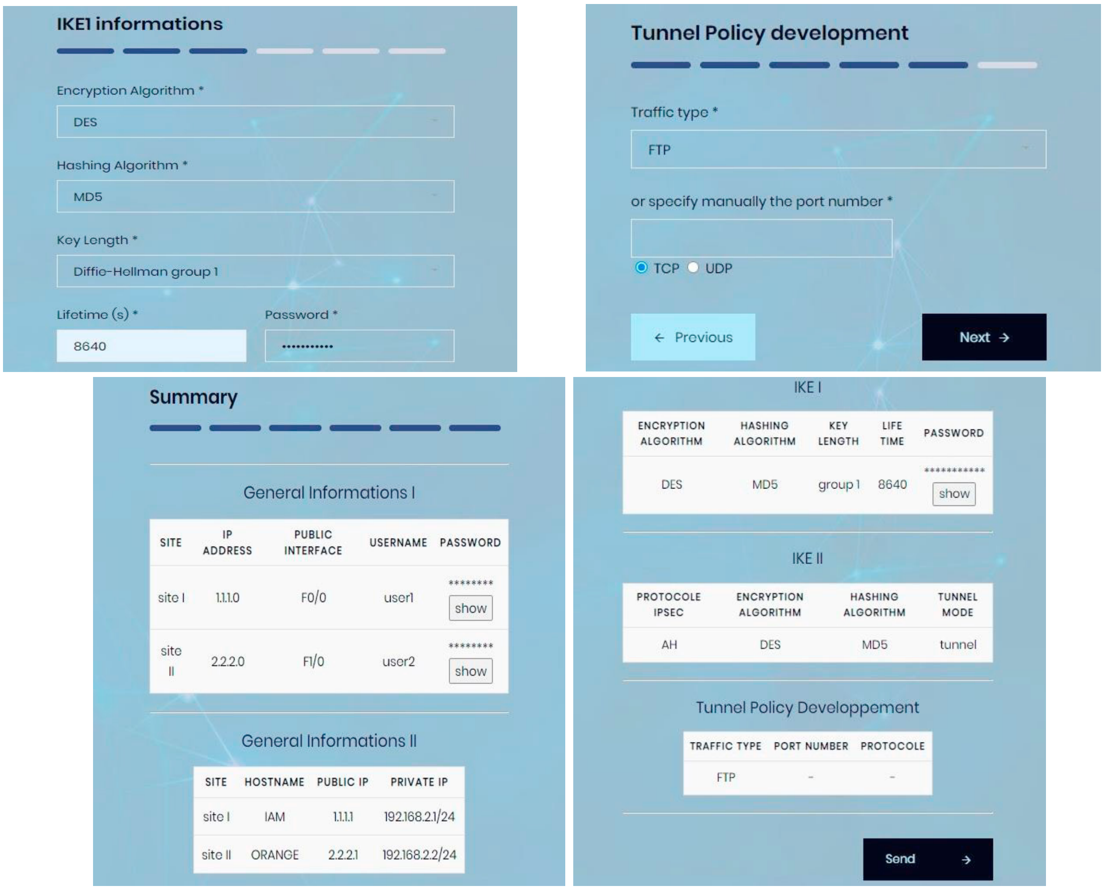
After determining all required parameters for setting up a VPN architecture between two peers, it is necessary to store these details in a relational database system for further usage. The model adopted in our solution is the one illustrated in the figure 5 shown below, the design of the latter was well-thought-out integrating all possible interactions between different entities, using the indexing of primary and foreign keys and the optimization of the entities number to facilitate future distribution or replication.

- *Phase 8: Verify the availability of the SSH channel*

One last step is required before moving on to the deliverance of the VPN configurations which is the verifying of the SSH channel’s status.

- *Phase 9: Prepare and deliver VPN configurations*

Finally, the VPN configurations can be prepared based on all the previous phases, on information originating from the previously constructed database, and depending on the detected OS system of the routers. And lastly these configurations will be delivered to ensure the construction of the VPN tunnel.



IKE1 informations

Encryption Algorithm *: DES

Hashing Algorithm *: MD5

Key Length *: Diffie-Hellman group 1

Lifetime (s) *: 8640

Password *: ••••••••••

Tunnel Policy development

Traffic type *: FTP

or specify manually the port number *:

● TCP ○ UDP

Summary

General Informations I

SITE	IP ADDRESS	PUBLIC INTERFACE	USERNAME	PASSWORD
site I	1.1.1.0	F0/0	user1	******** show
site II	2.2.2.0	F1/0	user2	******** show

General Informations II

SITE	HOSTNAME	PUBLIC IP	PRIVATE IP
site I	IAM	1.1.1.1	192.168.2.1/24
site II	ORANGE	2.2.2.1	192.168.2.2/24

IKE I

ENCRYPTION ALGORITHM	HASHING ALGORITHM	KEY LENGTH	LIFE TIME	PASSWORD
DES	MD5	group 1	8640	*********** show

IKE II

PROTOCOLE IPSEC	ENCRYPTION ALGORITHM	HASHING ALGORITHM	TUNNEL MODE
AH	DES	MD5	tunnel

Tunnel Policy Developpement

TRAFFIC TYPE	PORT NUMBER	PROTOCOLE
FTP	-	-

Figure 4 Application Example

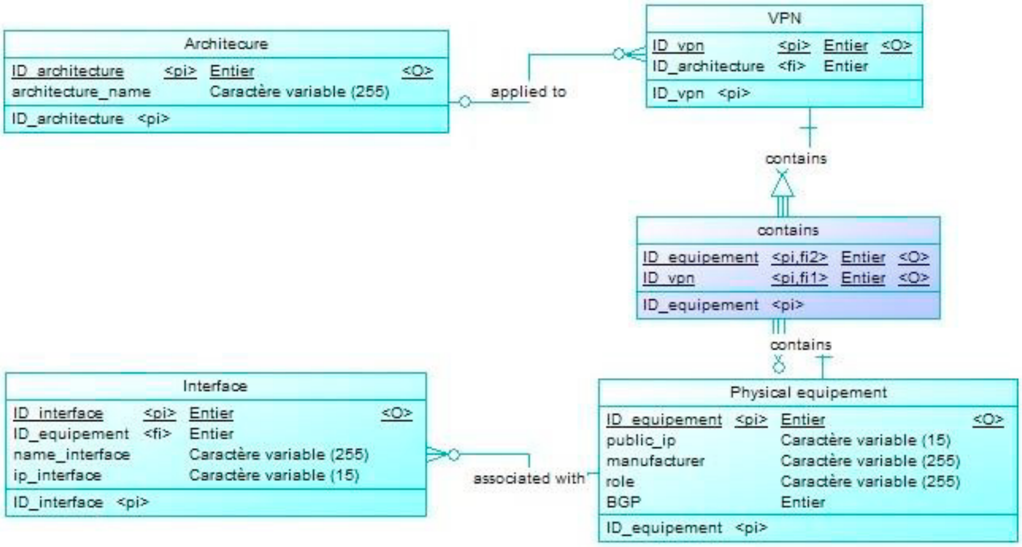

Architecure
ID_architecture <pi> Entier <O>
architecture_name Caractère variable (255)
ID_architecture <pi>
applied to
VPN
ID_vpn <pi> Entier <O>
ID_architecture <fi> Entier
ID_vpn <pi>
contains
contains
ID_equipement <pi,fi2> Entier <O>
ID_vpn <pi,fi1> Entier <O>
ID_equipement <pi>
contains
Interface
ID_interface <pi> Entier <O>
ID_equipement <fi> Entier
name_interface Caractère variable (255)
ip_interface Caractère variable (15)
ID_interface <pi>
associated with
Physical equipement
ID_equipement <pi> Entier <O>
public_ip Caractère variable (15)
manufacturer Caractère variable (255)
role Caractère variable (255)
BGP Entier
ID_equipement <pi>

Figure 5 The physical model of the data

5. Conclusion

In this paper we have proposed a new architecture that allows the automation of VPN technology. The proposed architecture is entitled in the DevOps framework, the chosen VPN technology is justified by the growing need that companies express nowadays in security matters. The architecture is based on three layers; the application layer, control layer and data layer has been proposed. The south interface API used in our solution is Python plus SNMP. The solution is accompanied by a user-friendly web interface that is easy to handle by all categories of administrators.

References

[1] *Petersen, K., Wohlin, C., & Baca, D. (2009, June). The waterfall model in large-scale development. In the International Conference on Product-Focused Software Process Improvement (pp. 386-400). Springer, Berlin, Heidelberg.*

[2] *Lwakatare, L. E., Kuvaja, P., & Oivo, M. (2015, May). Dimensions of devops. In International conference on agile software development (pp. 212-217). Springer, Cham.*

[3] *Bahnasse, A., Talea, M., Badri, A., Louhab, F. E., & Laafar, S. (2020). Smart hybrid SDN approach for MPLS VPN management on digital environment. Telecommunication Systems, 73(2), 155-169.*

[4] *STEWART, Frances. Development and security. Conflict, Security & Development, 2004, vol. 4, no 3, p. 261-288.*

[5] *Mihăilă, P., Bălan, T., Curpen, R., & Sandu, F. (2017). Network Automation and Abstraction using Python Programming Methods. MACRO 2015, 2(1), 95-103.*

[6] *Lackorzynski, T., Köpsell, S., & Strufe, T. (2019, May). A comparative study on virtual private networks for future industrial communication systems. In 2019 15th IEEE International Workshop on Factory Communication Systems (WFCS) (pp. 1-8). IEEE.*

[7] *Bahnasse, A., Talea, M., Badri, A., & Louhab, F. E. (2018, April). New smart platform for automating MPLS virtual private network simulation. In 2018 International Conference on Advanced Communication Technologies and Networking (CommNet) (pp. 1-8). IEEE.*

[8] *Uludag, Ö., Kleehaus, M., Caprano, C. and Matthes, F., 2018, October. Identifying and structuring challenges in large-scale agile development based on a structured literature review. In 2018 IEEE 22nd International Enterprise Distributed Object Computing Conference (EDOC) (pp. 191-197). IEEE.*