

UNIVERSIDADE FEDERAL DE CAMPINA GRANDE
CENTRO DE ENGENHARIA ELÉTRICA E INFORMÁTICA
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

DAVI ANDRADE PONTES

SPIRE-Based Remote Attestation for Secure VPN Access

CAMPINA GRANDE - PB

2025

DAVI ANDRADE PONTES

SPIRE-Based Remote Attestation for Secure VPN Access

Trabalho de Conclusão Curso apresentado ao Curso Bacharelado em Ciência da Computação do Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande, como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

Orientador : Andrey Elision Monteiro Brito

CAMPINA GRANDE - PB

2025

P814s Pontes, Davi Andrade.
Spire-based remote attestation for secure VPN access / Davi Andrade Pontes. – Campina Grande, 2025.
8 f. : il color.

Artigo (Bacharelado em Ciência da Computação) – Universidade Federal de Campina Grande, Centro de Engenharia Elétrica e Informática, 2025.
"Orientação: Prof. Dr. Andrey Elisio Monteiro Brito".
Referências.

1. Device Attestation. 2. User Authentication. 3. Zero Trust Security. 4. Virtual Private Network (VPN). 5. Remote Attestation. 6. SPIFFE. 7. SPIRE. 8. Multi-Factor Authentication (MFA). I. Brito, Andrey Elisio Monteiro. II. Título.

CDU 004.056.53(04)

FICHA CATALOGRAFICA ELABORADA PELA BIBLIOTECARIA SEVERINA SUELI DA SILVA OLIVEIRA CRB-15/225

DAVI ANDRADE PONTES

SPIRE-Based Remote Attestation for Secure VPN Access

Trabalho de Conclusão Curso apresentado ao Curso Bacharelado em Ciência da Computação do Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande, como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

BANCA EXAMINADORA:

Andrey Elisio Monteiro Brito

Orientador – UASC/CEEI/UFCG

Thiago Emmanuel Pereira da Cunha Silva

Examinador – UASC/CEEI/UFCG

Everton Leandro Galdino Alves
Examinador – UASC/CEEI/UFCG

Trabalho aprovado em: 10 de MAIO de 2025.

CAMPINA GRANDE - PB

RESUMO

Organizações ao redor do mundo adotaram amplamente Redes Privadas Virtuais (VPNs) para proteger recursos internos contra acessos não autorizados. No entanto, essas soluções apresentam problemas críticos de segurança, pois a organização não tem controle sobre os dispositivos que acessam a VPN. Uma vez que um dispositivo ou credencial é comprometido, é difícil para a organização identificar o problema e remover o acesso do invasor. As soluções de Confiança Zero (Zero Trust, ou ZT) surgiram para resolver esses problemas de segurança, adotando verificações constantes do usuário e do hardware, além de proteção em cada ponto final da infraestrutura, permitindo um controle mais granular sobre os acessos. No entanto, a migração de VPNs tradicionais para Arquiteturas de Confiança Zero (ZTA) pode ser desafiadora devido às mudanças significativas exigidas na infraestrutura, nas políticas de controle de acesso e nos modelos de segurança. Este trabalho propõe uma solução intermediária, aprimorando a parte de autenticação das VPNs para ajudar a alcançar alguns dos benefícios da abordagem ZT sem a necessidade de modificar a infraestrutura da VPN. A proposta utiliza o padrão open source da Cloud Native Computing Foundation, o Secure Production Identity Framework for Everyone (SPIFFE), e sua implementação de referência, o SPIFFE Runtime Environment (SPIRE), para construir uma autorização baseada em atestação no acesso à VPN. O trabalho propõe novos validadores de nós e cargas de trabalho para o SPIRE, permitindo verificar a identidade do usuário e da máquina antes de conceder acesso à VPN. O SPIRE também fornece rotação automática de credenciais, evitando o uso de credenciais de longo prazo e minimizando o impacto em caso de roubo dessas credenciais. Por fim, o artigo apresenta uma visão geral de alguns problemas de segurança nos métodos tradicionais de autenticação em VPNs e como a solução proposta os mitiga.

SUMMARY

ABSTRACT……... 5

1. INTRODUCTION…….. 5
2. BACKGROUND…… 6
   - 2.1. Zero Trust Architecture and Virtual Private Networks……... 6
   - 2.2. SPIFFE and SPIRE……. 6
   - 2.3. Single Sign On and OpenID Connect……. 7
   - 2.4. Trusted Platform Module……..... 8
3. PROBLEM DEFINITION AND SOLUTION REQUIREMENTS……... 8
4. PROPOSAL…… 9
   - 4.1. Overview……..... 9
   - 4.2. Implementation……... 9
5. EVALUATION……... 10
   - 5.1. Requirements Satisfaction…….. 10
   - 5.2. Resources Consumption……. 11
6. RELATED WORK……. 11
7. CONCLUDING REMARKS……..... 11

ACKNOWLEDGEMENTS……... 12

REFERENCES…….. 12

SPIRE-Based Remote Attestation for Secure VPN Access

Davi Pontes
Federal University of Campina Grande
Campina Grande, Paraíba, Brazil
davi.pontes@ccc.ufcg.edu.br

Andrey Brito
Federal University of Campina Grande
Campina Grande, Paraíba, Brazil
andrey@computacao.ufcg.edu.br

ABSTRACT

Organizations globally have adopted Virtual Private Networks (VPNs) extensively to safeguard internal resources against unauthorized access. However, these solutions have critical security problems, as the organization has no control over the devices accessing the VPN. Once a device or a credential is compromised, it is hard for the organization to identify the problem and remove the attacker's access. Zero Trust (ZT) solutions came to solve these security problems by adopting constant verifications on the user and the hardware and protection on each endpoint on the infrastructure, allowing more granular control over access. However, migrating from traditional VPNs to Zero Trust Architectures (ZTA) can be challenging due to the significant changes required in infrastructure, access control policies, and security models. This work proposes an intermediate solution, enhancing the authentication part of VPNs to help achieve some of the benefits of ZT without needing modifications to the VPN infrastructure. Leveraging the Cloud Native Computing Foundation's open-source standard, Secure Production Identity Framework for Everyone (SPIFFE), and its reference implementation, the SPIFFE Runtime Environment (SPIRE), to build an attestation-based authorization on VPN access. The work proposes a new node and workload attestors for SPIRE, enabling the verification of user and machine identity before granting access to the VPN. SPIRE also provides automatic credential rotation, avoiding long-term credentials and minimizing the impact if the credentials are stolen. Finally, the paper presents an overview of some security problems with traditional VPN authentication methods and how the proposed solution mitigates them.

CCS CONCEPTS

• **Security services → Access control**; • **Security protocols → Authentication**; • **Network architectures → Virtual Private Networks**; • **User authentication → Multi-factor authentication**.

KEYWORDS

Device Attestation, User Authentication, Zero Trust Security, Virtual Private Network (VPN), Remote Attestation, SPIFFE, SPIRE, Multi-Factor Authentication (MFA)

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
Conference'17, July 2017, Washington, DC, USA
© 2025 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

ACM Reference Format:
Davi Pontes and Andrey Brito. 2025. SPIRE-Based Remote Attestation for Secure VPN Access. In . ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

After the COVID-19 pandemic, organizations worldwide had to adopt the home-office system, forcing many employees to use their personal computers (PCs) to work from home. This shift introduced significant security concerns. Previously, organizations maintained a well-defined security perimeter around their managed workstations. However, after this change, the workstations were widely distributed, often beyond the organization's control, making it difficult to verify who accessed these machines and ensure a secure work environment. Even with the end of the pandemic, many of these companies have maintained this work model.

A virtual private network (VPN) helps solve some security issues associated with remote work. With it, the organization can protect sensitive resources within an emulation of a private network built upon a potentially unprotected network infrastructure, such as the Internet, granting access to these resources only to authenticated and authorized users. This solution gives the organization more control over its resource access. However, they still have problems with the security of the employees' PCs.

In the old, physical-office habits, the organizations had control over the workstations' physical and logical security, monitoring who had physical access to the machines and constantly applying security patches and critical updates on the Operating System (OS), but with the home office and the usage of PCs for work, this control has been lost. Although VPNs can help protect the communication channel between the employee's PC and the company's servers, they do not guarantee that the person using the credentials or the computer accessing the private resources is authorized to do so.

In addition to VPNs, Zero Trust Architectures (ZTAs) [2] also help solve the remote access to restrict resources. Unlike VPNs, ZT network security models don't emulate a private network over the original infrastructure; instead, they verify every access request, regardless of location, before granting access to resources. To do so, every component accessing the infrastructure must have its identity. With this, identity and permission verifications can be performed before granting access to any resources within the infrastructure. In this model, the components must be constantly verified, and one way to achieve this is through an attestation process; the component must present proof of identity and current state to obtain the necessary credentials. Although ZTAs solve the identity verification issues that VPNs face, they also require significant modifications to resource management and protection, making ZTA implementations more complex than VPN implementations.

This work proposes a solution to address the VPN user's identity verification problem by ensuring constant verification and combining the simplicity of VPNs with the robustness of continuous verification. It leverages open-source projects to build a reliable and scalable way to apply constant verifications on users and users' PCs. This identity verification allows the user to obtain the credentials to access a VPN, avoiding static, long-term credentials that could be stolen and misused. In the proposed solution, access to the VPN is attached to the machine, and multi-factor authentications are applied to verify the user's identity. All this, combined with the constant verification, creates a robust system for making VPN access more secure. Furthermore, the solution can also be applied to ZTA, providing the necessary credentials for users to access resources within the architecture.

This document is organized as follows. Section 2 defines key concepts necessary to understand the problem and the solution. Section 3 refines the problem, the threat model, and the requirements that solutions must satisfy. Section 4 defines the solution proposal, implementation, and how it solves the problem. Section 5 describes how the solution meets the problem's requirements, compares it with other existing solutions, and examines resource consumption metrics, such as CPU and RAM. Section 6 describes some works related to this one. Finally, Section 7 summarizes the main findings and describes some possible future works.

2 BACKGROUND

2.1 Zero Trust Architecture and Virtual Private Networks

2.1.1 Virtual Private Network. The Virtual Private Networks (VPN) architecture emulates private, secure networks over a public or less secure network, such as the Internet. The main security features of a VPN are the encryption and authentication of data traffic by employing tunneling mechanisms, thereby protecting data transmitted over potentially untrusted infrastructures. With this, an organization can protect and guarantee access to its infrastructure for devices authenticated in the VPN.

The main VPN implementation currently used is OpenVPN (OVPN) [8]. It allows remote clients to access resources protected inside the VPN. The advantage of OVPN over other protocols, such as Point-to-Point Tunneling Protocol (PPTP), which was used on past VPNs, is the support for Network Address Translation (NAT), firewalls, and modern encryption and authentication methods. OVPN also supports modern user authentication methods through user/-password or public/private certificates to authenticate clients on the OVPN Server.

Figure 1 describes how OVPN architecture is organized. The organization will provide OVPN Access Servers where the clients will connect, and each client will be authenticated using one of the supported methods. After authentication, the OVPN Access Server will tunnel the client's communication, cryptographically protecting it using SSL/TLS. In addition to this protection, OVPN provides Internet Protocol Security (IPSec), providing privacy to the connected clients by protecting the Internet Protocol (IP) packet, masking the client's original IP, and emulating a private network.

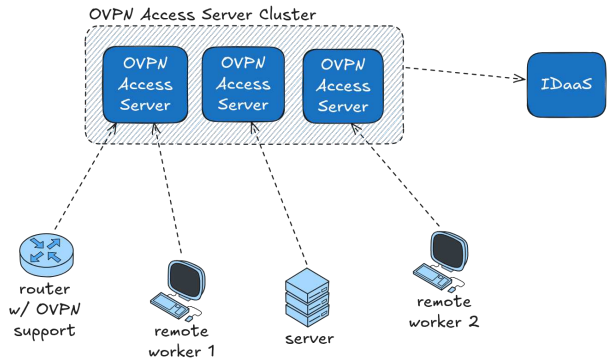


Figure 1: OpenVPN Remote Access Architecture. [9]

2.1.2 Zero Trust Architecture. Zero Trust (ZT) is a network security model based on principles that eliminate implicit trust in any user, application, or hardware without prior verification. It enforces the principle of least privilege by granting the minimum necessary access to all pieces of the system and continuously verifying and authenticating all communications [2].

Zero Trust Architecture (ZTA), in turn, is the implementation of Zero Trust (ZT) principles, applying them to design an organization's infrastructure and workflows [11]. ZTAs can also be used to secure communication between the devices and services of a single or different organization. As ZTA follows ZT principles, the idea is to have all the devices inside this architecture constantly being verified. The advantage of ZTA is that it does not have perimeter-based security, such as VPNs. All the communication will be authenticated and encrypted, removing the problem of having a single point of failure, as the communication is not concentrated in a single server or set of clusters.

2.2 SPIFFE and SPIRE

With modern automated infrastructure concepts and distributed services based on the cloud, and with the arrival of ZT [2] concepts, the need to provide identities to software components has increased. Moreover, generating these identities by hand poses some problems in terms of scalability, as it is impractical to create hundreds or maybe thousands of credentials by hand, and security, as the manual method implies having long-term credentials, which is undesirable in ZTAs.

2.2.1 SPIFFE Standard. The Secure Production Identity Framework for Everyone (SPIFFE) [3] is a specification that describes how to provide identities to software components in a platform—and technology-agnostic fashion. SPIFFE's primary objective is to avoid leaving a last credential unprotected. For that, SPIFFE issues identities based on the software's attestation process.

In SPIFFE's vocabulary, a piece of software is called a workload. After attestation, the workload receives its identity as an SPIFFE Verifiable Identity Document (SVID), a cryptographically verifiable object representing the workload identity. This object can be, for instance, an X.509 certificate or a JSON Web Token (JWT). The SVID also contains the SPIFFE ID, a string representing the workload's identity. The SPIFFE ID is a Uniform Resource Identifier (URI),

containing the trust domain and an identifier for the workload. This identifier could be any string that identifies the workload, such as a name, a hash, or a unique ID. A valid SPIFFE ID looks like *spiffe://example.org/my-example-workload*.

To obtain its identity, the workload communicates with an API called the Workload API. This API is responsible for giving identities to the workloads. To avoid the need for bootstrap credentials, the Workload API does not require authentication. As it has no authentication, it must collect information about granting the workload's identity from a reliable source, e.g., the Operating System. Attestation consists of acquiring information about the workload to derive its identity. The properties used to derive the identity are called selectors.

2.2.2 SPIFFE's Reference Implementation. SPIFFE Runtime Environment (SPIRE) is the reference implementation[1] of SPIFFE. SPIRE has two main components: the SPIRE Server and the SPIRE Agent. Figure 2 gives an overview of SPIRE's architecture.

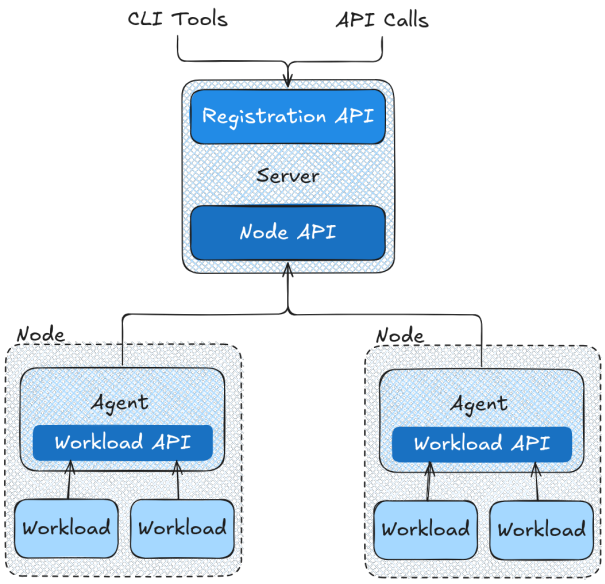


Figure 2: SPIRE Architecture.[10]

The SPIRE Server is the root of trust in SPIRE architecture and must run in a secure and reliable environment. It is responsible for managing all the identities, attesting the nodes that run the workloads, and issuing the SVIDs. It provides two APIs: the registration API, which manages the creation of identities of the workloads by setting the selectors, and the SPIFFEID and the Node API, which provides the interface to attest a node. Each identity registration is called an entry, and the Node API, which the Agents will communicate with to obtain their own identities, provides the Workload API and delivers the SVIDs to the workloads.

The SPIRE Agent, in turn, runs in each node where workloads are running. It is responsible for attesting the workload's identity by using a reliable source to obtain information about it, e.g., the node's operating system (OS). The agent itself must also be attested within the Server. After successful attestation, the Agent receives its SVID and can issue the SVIDs of all the associated workloads.

To provide various attestation capabilities, SPIRE follows a plugin-oriented architecture, which permits the easy implementation of different forms of attestation for the Nodes and the Workloads. Node Attestor plugins manage the process of attesting a Node, while Workload Attestor plugins enable the attestation of workloads. Each plugin will generate selectors, which are used to create the identities through the Registration API on the SPIRE Server. Figure 3 shows what a CLI command to create a node and a workload identity looks like.

```
root@spire-server:~# spire-server entry create \
-node \
-spiffeID spiffe://ufcg.edu.br/pc01 \
-selector tpm_devid:issuer:cn:issuer.org \
-selector tpm_devid:subject:cn:ufcg.edu.br
Entry ID         : 01769cd9-da3a-494c-8341-73f0ce0e929b
SPIFFE ID        : spiffe://ufcg.edu.br/pc01
Parent ID        : spiffe://ufcg.edu.br/spire/server
Revision         : 0
X509-SVID TTL    : default
JWT-SVID TTL     : default
Selector         : tpm_devid:issuer:cn:issuer.org
Selector         : tpm_devid:subject:cn:ufcg.edu.br

root@spire-server:~# spire-server entry create \
-parentID spiffe://ufcg.edu.br/pc01 \
-spiffeID spiffe://ufcg.edu.br/user/davi \
-selector unix:uid:1001 \
-selector unix:user:davi
Entry ID         : 1663e065-b803-40e1-9506-ef13ddd720a5
SPIFFE ID        : spiffe://ufcg.edu.br/user/davi
Parent ID        : spiffe://ufcg.edu.br/pc01
Revision         : 0
X509-SVID TTL    : default
JWT-SVID TTL     : default
Selector         : unix:uid:1001
Selector         : unix:user:davi
```

Figure 3: Entries registration CLI commands.

In figure 3, the first command refers to the creation of a node identity, as the *-node* flag suggests, with the SPIFFEID spiffe://ufcg.edu.br/pc01, which will be issued to the node with the Trusted Platform Module (TPM) device identity document with issuer common name equals to *issuer.org* and the subject common name equals to *ufcg.edu.br*, as the selectors describe.

The second command is for the workload entry registration. The *-parent* flag indicates that this workload identity is associated with Agent with the SPIFFEID spiffe://ufcg.edu.br/pc01, and must be delivered by it. The selectors describe that the workload must be a UNIX process owned by the user named *davi* with UID *1001*.

2.3 Single Sign On and OpenID Connect

Single Sign-On (SSO) is an authentication scheme that allows the user to authenticate once using a single set of credentials and access different systems using the same credentials. SSO uses the concept of Identity as a Service (IDaaS), relying on an Identity Provider (IdP)

[1]https://github.com/spiffe/spire

to manage users' logins. The idea of SSO is that the user logs into the IdP using a set of credentials and receives an access token. This token is sent to applications to authenticate the user, and the applications verify the token and get the users' information through the IdP. Using SSO simplifies the user's experience, avoiding the necessity of storing or remembering lots of credentials, and provides an easy way to implement Multi-Factor Authentication (MFA), which reduces the vulnerabilities related to credential leaks.

Open ID Connect (OIDC) is an authentication protocol based on OAuth2. It simplifies users' identity verification based on the authentication performed by an Authorization Service. OIDC can be used to enable SSO in an application by relying on an IdP, named OpenID Provider (OP) in the OIDC context, to manage the authentication process and users' credentials. OIDC workflow is described in figure 4.

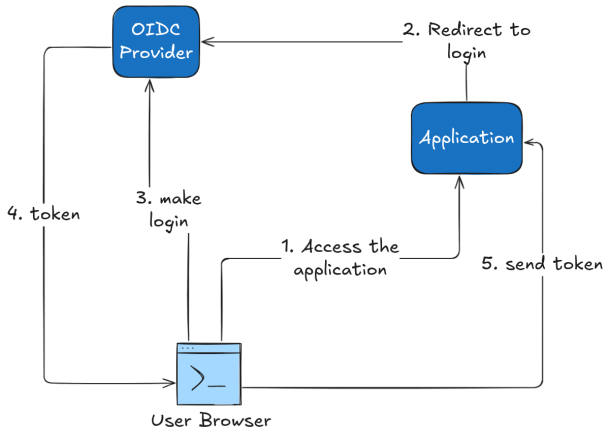


Figure 4: OIDC protocol workflow.

The user accesses the application, which will redirect it to the OIDC login page. The user then logs in using the credentials, receives the Access Token, and sends it to the application. The Access Token allows the client application to perform delegated authorization, granting access to a designated resource to carry out defined actions on behalf of the user after prior consent.

OIDC provides an easy way to implement SSO and make applications more secure since the developers do not need to worry about protecting the users' credentials and allowing easy implementation of modern authentication mechanisms, such as MFA, relying on an OIDC provider to handle all the complexities of it.

2.4 Trusted Platform Module

Trusted Platform Module (TPM) [1] is a security chip on a computer that stores secrets and verifies platform identity and integrity. To verify platform identity, TPM has an Endorsement Key (EK), stored in the TPM's hardware and with a private key not accessible by software. The EK is unique to each TPM chip and is signed by the vendor's certificate authority (CA) so it can be used to verify the platform's identity. The TPM also has registers that store hashes of measurements of software components running on the platform since its boot. These registers are called Platform Configuration Registers (PCRs), and there are 24 of them on the TPM chip. The PCRs can store hashes of the machine's boot process, such as kernel, Unified Extensible Firmware Interface (UEFI), and initial RAM file system (initramfs). The TPM can generate reports with the PCR values; these reports are signed by the Attestation Key (AK), a TPM's private key signed by the vendor's CA, and also protected on the TPM's private memory. This allows the verification of the reports given by the TPM.

3 PROBLEM DEFINITION AND SOLUTION REQUIREMENTS

This work considers the following initial scenario: an organization that provides remote access to specific private resources for its employees and selected clients through a VPN connection. The organization's VPN uses the OVPN protocol and supports X.509 certificates for user authentication, including its employees and some of its clients.

In such a scenario, the organization issues the X.509 certificates and delivers them to the employees through some secure channel to avoid leakage. The employees then store the certificates on their personal computers (PCs) to use them in the authentication process when connecting to the VPN. Once stored in the employees' PC, an attacker aiming to get access to the organization's infrastructure can focus on attacking the employees (e.g., through personalized social attacks), as they are way more unprotected than the organization's servers. Leveraging this, an attacker could steal employees' credentials and get access to the infrastructure, using this access as a door to start escalating privileges within the organization. Table 1 describes some of the attacks this scenario is vulnerable to.

Attack	Entry Point	Impact
Primary Key Theft	Malware, phishing, keyloggers	VPN access, impersonation
Physical Device Theft	Stolen laptop, unattended sessions	VPN access, further internal compromise
Session Hijacking	Weak TLS configs	Take over existing VPN session
Lateral Movement	Weak internal access control	Internal escalation, data breach
Persistent Backdoors	Malware on an employee PC	Long-term stealthy access & Endpoint detection

Table 1: Threat Model for VPN Access Protection

To mitigate the threats, a proposed solution must meet specific requirements to protect against each attack vector. (1) To prevent private key theft, the system should enforce device identity attestation to prevent credentials from being used on untrusted devices and Multi-Factor Authentication (MFA) to strengthen security by not trusting in only one factor to authenticate. (2) Countermeasures such as full disk encryption, session re-authentication, short-term credentials, and the ability to remotely revoke compromised devices' access can significantly reduce the risk if a physical device is stolen. (3) To defend against session hijacking, the system should

employ mutual TLS (mTLS) to ensure strong, bidirectional authentication and use short-lived credentials to minimize the usefulness of stolen sessions. (4) Addressing lateral movement within the infrastructure requires enforcing least-privilege access principles and implementing per-request authentication, ensuring users can only access necessary resources with re-verification. (5) Lastly, detecting and containing persistent backdoors requires continuous monitoring and automated access revocation mechanisms for identifying anomalies and cutting off compromised devices' access.

4 PROPOSAL

4.1 Overview

The solution proposed in this work aims to secure access to VPNs through attestation processes. The objective is to avoid static, long-term credentials and rely on previous verifications to guarantee access to the VPN without needing to change the original VPN infrastructure but the method to authenticate the users. The verifications are applied to the physical device to ensure its identity and integrity while verifying the user's identity. It is also important to tie the user to the physical device, preventing an authenticated user from using hardware he is not authorized to use to avoid internal impersonation.

The solution relies on hardware to attest to the device identity and integrity, as software is easily corrupted. Only after hardware verification is the user authentication performed, using MFA to rely on more than one factor to identify the user. After verifying the device and the user, the necessary credentials, in the form of X.509 certificates, are issued, and the user will be able to access the VPN using them. This credential will be automatically rotated constantly, performing the attestation process on every certificate rotation to guarantee constant verification. Note that the system will not provide the credentials if the user or the hardware fails the attestation, as both must be successfully attested to receive them. The next section details the implementation and technologies used in the solution.

4.2 Implementation

The implementation consists of SPIRE plugins capable of attesting the physical machine through TPM and the user identity through authentication into the Auth0[2] OIDC provider. In the solution, the infrastructure operator will create both the PC and the user entry on the SPIRE Server. The user's identity is related to the PC, creating a strong bond between them. The SPIRE Server uses the Trusted Platform Module (TPM) [6] to attest to the PC's identity. As the TPM identity is a hardware feature, it becomes harder for the attacker to corrupt it. After attestation, the SPIRE Agent will attest the user's identity through the User Attestation Module (UAM), built to allow attestation and store the SVID to be used on VPN access.

4.2.1 Attesting the Machine. To attest to the PC, a new SPIRE Node Attestor plugin was built, using TPM capabilities to attest to the device identity and boot integrity. This can ensure that the device receiving the identity is correct and in a trusted state (e.g., running a specific kernel version with disk encryption and secure boot enabled). Figure 5 describes the TPM Node attestation workflow.

[2]https://auth0.com/

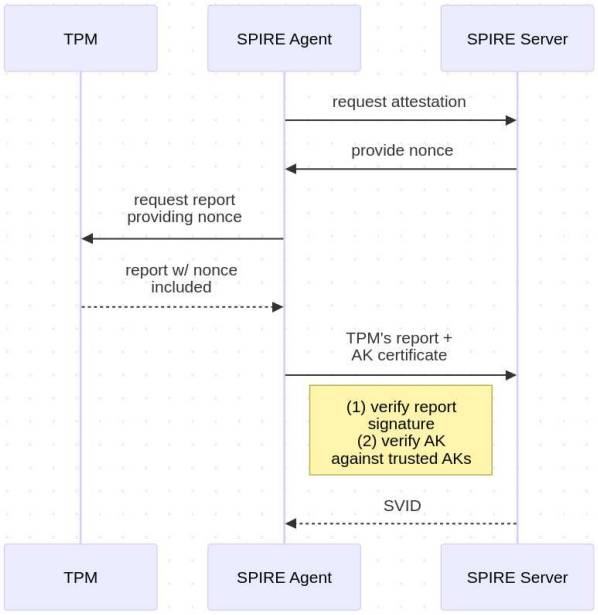


Figure 5: Machine Attestation Workflow.

```
NodeAttestor "tpm" {
    plugin_cmd = "/opt/spire/server/node_attestors/tpm_node_attestor"
    plugin_data = {
        trusted_aks_path = "/opt/spire/server/trusted_aks.bundle"
    }
}
```

Figure 6: TPM Node Attestor Configuration.

The SPIRE Agent starts requesting attestation to the SPIRE Server. The Server will then generate a nonce[3] and send it to the SPIRE Agent. The SPIRE Agent then asks for the TPM to emit a report containing information about the computer, such as PCR values and secure boot state. The TPM will include the nonce provided by the SPIRE Server in the report, alongside the information about the computer, and sign it with the AK private key. The Agent transmits the report to the SPIRE Server, which verifies whether the signature originates from one of the pre-configured trusted Attestation Keys (AKs) and if the nonce included on the report is the one sent before. If the nonce matches and the report was signed by a trusted AK, the SPIRE Server emits the SVID to the Agent.

On the plugin server-side configuration, the operator must provide the trusted AKs to manage which TPMs are trusted. As the AK is unique for each TPM, if an AK is set to trusted means that the identity of the machine that owns it is trusted. Figure 6 shows an example configuration on the server side.

After successful attestation, the Server generates selectors to identify the machine. These selectors describe information about the machine's identity and state. Table 2 shows and describes these selectors. The PCR selectors help identify important information about the machine, such as the UEFI, kernel version, and initramfs used in the boot process. This information gives an overview of

[3]The nonce is random data generated to guarantee the freshness of a report.

the machine's state on boot. According to the Trusted Computing Group (TCG) [12], the Platform Configuration Registers (PCRs) in a TPM are used to store integrity measurements of the system during the boot process. Specifically, PCR0 records information about the UEFI firmware from the system board ROM, while PCR1 reflects the host platform configuration. PCRs 2 and 3 capture measurements related to UEFI drivers, and PCRs 4 and 5 store data about the UEFI Boot Manager code and configuration and the GPT/partition table. PCR6 contains information about the platform manufacturer, and PCR7 is reserved for the system's Secure Boot policy. Finally, PCRs 8 through 15 are defined and used by the operating system for additional measurements. PCR16 is used for debugging, and 23 is for application support. The other PCRs can be extended as needed.

Selector	Description
ak_hash:<hash_value>	SHA256 of the AK certificate. It identifies the machine's identity.
pcr_[num]:<pcr_value>	The SHA256 value of PCR 0 to 23.
secure_boot:enabled	Indicates if Secure Boot is enabled on the machine. It is not present if Secure Boot is not enabled.
kernel_cmd:<value>	The kernel command line with blank spaces removed and lower case letters.

Table 2: SPIRE TPM selectors.

4.2.2 Attesting User's Identity. For user identity attestation, an SPIRE workload attestor plugin and a User Attestation Module (UAM) are used. The UAM is used to authenticate the user, this module is responsible for managing the user login on the Auth0 API. The Auth0 API can be configured as the system administrator prefers. It is strongly recommended to use MFA to enhance security by relying on different factors of the user to authenticate. The SPIRE workload attestor is responsible for communicating with the UAM to retrieve the auth tokens, validate them, and fetch user information to build the selectors to identify the user. Figure 7 describes the user authentication and UAM attestation workflow.

The user starts the login process within the UAM, and the UAM redirects the user to the Auth0 login web page for authentication. The user logs in with valid credentials. After successful authentication, Auth0 calls a callback endpoint on the UAM, passing the auth token. Upon receiving the token, the UAM requests attestation from the SPIRE Agent through the workload API. The SPIRE Agent retrieves the authentication token through a UNIX socket exposed by the UAM and verifies it within the Auth0 API. After validating the token and fetching the user's information, the SPIRE Agent generates selectors based on the user's attributes. If an identity is registered using these selectors and with the Agent as parent, the UAM receives the SVID and stores it for VPN authentication. Figure 8 shows the workload attestor configuration. The *auth0_domain* is the Auth0 endpoint used to verify the auth token and retrieve the user's information.

After attestation, the SPIRE Agent generates the selectors to identify the user. Table 3 lists these selectors.

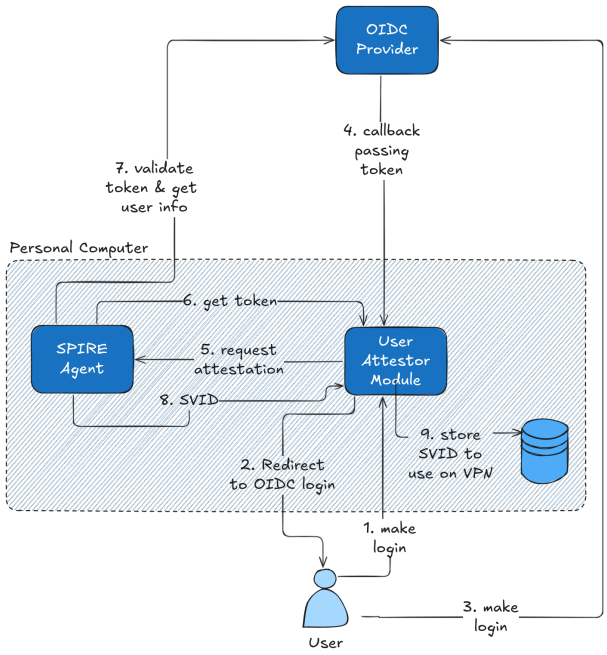


Figure 7: Personal Computer and User Attestation Workflow.

```
WorkloadAttestor "user" {
    plugin_cmd = "/home/davi/UFCG/SPIREPersonalComputing/dist/plugin"
    plugin_data {
        auth0_domain = "https://dev-j58fwup7mk575dp2.us.auth0.com/"
    }
}
```

Figure 8: SPRIE User Workload Attestor configuration.

Selector	Description
sub:<value>	Value unique for each user, can be used as an ID.
given_name:<value>	The first name of the user.
family_name:<value>	User's family name.
nickname:<value>	User's nickname.
name:<value>	User's full name.
email:<value>	User's email address.
email_verified:<true/false>	Indicates if the user has verified the e-mail address.

Table 3: SPIRE user selectors.

5 EVALUATION

This section evaluates the proposed solution by analysing how it fits the solution requirements described in section 3. It also analyzes resource consumption to see if the solution impacts PC performance.

5.1 Requirements Satisfaction

Section 3 has described the problems with VPN protection and enumerated some requirements that a solution must achieve to enhance the security of VPN authentication. Table 4 presents the

attacks identified in Section 3 and how the proposed solution helps mitigate them.

Attack	Mitigation Strategies
Private Key Theft	Device identity attestation, Multi-factor authentication (MFA), Constant certificate rotation
Physical Device Theft	Short-term credentials, Remote access revocation
Session Hijacking	Short-term credentials
Lateral Movement	—
Persistent Backdoors	Continuous attestation, Device integrity attestation

Table 4: Mitigation Strategies for VPN Threats

For Private Key Theft, the proposed solution provides MFA to strengthen the defense by requiring additional factors to authenticate the user, making it harder for an attacker to use stolen credentials. The device attestation guarantees that even if an attacker gains access to the user's credentials, they cannot authenticate into the VPN without access to the physical machine. The solution also relies on SPIRE to provide constant certificate rotation, reducing the damage in case of a successful private key theft.

In the case of Physical Device Theft, short-term credentials help prevent prolonged unauthorized access, as the machine will be valid, but the user credentials (e.g., MFA) will still be missing. SPIRE revocation of SVIDs allows administrators to disable access to the stolen device quickly.

To counter Session Hijacking, short-term credentials reduce the lifetime of active sessions. Any intercepted session tokens quickly become useless unless the attacker can continuously re-authenticate.

Lateral Movement attacks are out of the scope of the solution, as it aims to guarantee the security of the authentication process, not the authorization. Although the solution does not mitigate these attacks, good practices such as having least privilege access can help protect against them and can be implemented in the VPN access control along with the proposed solution.

Finally, for Persistent Backdoors, device integrity attestation verifies that the system has not been tampered with, thereby preventing malware-based persistence mechanisms from remaining undetected. The continuous attestation ensures that this verification is made on every certificate rotation.

5.2 Resources Consumption

To evaluate the resource consumption, tests were run using a PC with an Intel Core i5-13450HX processor and 16 GB of RAM, running the UAM and the SPIRE Agent. The machine had an Ubuntu 22.04.5 LTS (Jammy Jellyfish) installed and was running SPIRE Agent version 1.11.2.

The Linux *ps* command retrieved CPU and RAM usage. The command has been executed with the parameters *"-p <pid>"*, to identify the process, and *"-o %cpu,%mem"*, to retrieve CPU and RAM usage in percentage. The executions were monitored for 1 hour, and the results are: the SPIRE Agent consumed a mean of 0.1% of CPU and 0.2% of RAM (32 *MB*). The UAM consumed less than 0.01% of the CPU and 0.1% (16 *MB*) of RAM. This represents a total mean consumption of 48*MB* of RAM and less than 0.3% of the CPU consumption. This indicates that the solution can be run on lightweight machines without a problem.

6 RELATED WORK

Secure Access Service Edge (SASE), proposed by [5], also aims to protect remote access to private resources using ZTA principles, without distinction between public and private networks, but verifying on every access attempt. In [13], the authors gave an overview of SASE and proposed a deployment strategy to enhance SASE security, providing device attestation with identity and environment verification and MFA on user authentication. However, migrating from regular VPNs to SASE requires significant architectural changes, making it hard to implement and maintain. Although [13] attests the device identity and state, it relies on software to do so. The identity is verified through a certificate emitted by the organization and stored in the device's memory, while the state is verified through software installed on the device, making it easier to corrupt. The SPIRE-based solution proposed in this work integrates easily with regular VPNs, enhancing the security and threat model of the existing solutions without modifications to the original infrastructure. It also relies on hardware to guarantee a device's identity and state, making it harder to tamper.

In [4], the authors also proposed a way to control access to a network through verifications on the user identity and the device security posture. It also applies device verification using hardware capabilities to guarantee integrity. It also uses AI to implement behavior profiling of the device on the network and automatically block if any anomaly is detected. While the proposed Zero Trust Network Access Control (ZTNAC) system brings strong security improvements, its deployment is too complex and requires the deployment of many new components. Despite the SPIRE-based solution not offering the same level of security, it is an easy way to enhance VPN security with less effort. It can also reach high maturity levels if combined with other solutions such as Software Defined Perimeter (SDP) [7] to ensure the application of the ZT principles.

7 CONCLUDING REMARKS

This paper has presented a novel approach to mitigating security threats on regular VPNs by adopting ZT principles in VPN's authentication procedures. It leverages SPIRE, an open-source software, to provide attestation of personal computers and users' identities. The solution applies hardware verifications to ensure the reliability of physical machines and strong and modern user authentication methods. It can be easily implemented over a traditional VPN to produce a better security process, allowing the application of ZT principles without making changes to the VPN infrastructure.

Despite the benefits, the solution still does not provide some of the features that the modern ZTA solution has, such as behavior analysis for user profiling on the network and automated access revocation. VPNs still work on the idea of a well-defined security perimeter; once a device enters the VPN, it is treated as reliable. The present solution helps mitigate this by enforcing verifications on the

authentication process to make it more difficult for an attacker to gain access to the VPN. However, solutions such as SDP and SASE fit ZT principles by construction without needing a security perimeter, but applying a verification on each endpoint of the infrastructure, providing a more granular access control.

Finally, the approach relies on open-source software only and can be combined with other modern ZT solutions. As it focuses on providing identities to end-users, if combined with a more rigorous access control system such as SDP, using proxies to manage the connections instead of a VPN, it could leverage both the SPIRE robust attestation process and automatic identity rotation and the granular access control of SDP. A solution that automates an SVID revocation based on user behavior on the network is also possible, creating a strong mechanism to secure the infrastructure.

ACKNOWLEDGMENTS

I thank my advisor, Professor Andrey Elísio Monteiro Brito, for his guidance and support throughout the development of this work and my entire undergraduate journey. His mentorship has been invaluable in navigating this academic adventure. I also thank the Secure and Scalable Identity Provisioning (SSIP) project members; Their collaboration, encouragement, and companionship made this journey not only productive but also enjoyable. Thanks to my family and the 'Kajueiro' friends, who supported me throughout my undergraduate studies. Special thanks go to my parents, Luis Henrique Pontes and Flávia Helena da Silva Andrade, who always believed in me, and to my cousin, Mozart Grisi Correia Pontes, who made completing this course possible.

REFERENCES

[1] W. Arthur and D. Challener. 2015. *A Practical Guide to TPM 2.0: Using the Trusted Platform Module in the New Age of Security*. Apress.

[2] Mark Campbell. 2020. Beyond Zero Trust: Trust Is a Vulnerability. *Computer* 53, 10 (2020), 110–113. https://doi.org/10.1109/MC.2020.3011081

[3] Daniel Feldman, Emily Fox, Evan Gilman, Ian Haken, Frederick Kautz, Umair Khan, Max Lambrecht, Brandon Lum, Agustín M Fayó, Eli Nesterov, et al. 2020. Solving the Bottom Turtle: a SPIFFE way to establish trust in your infrastructure via universal identity. *Sprint Lab, Nova Zelândia* (2020).

[4] P. García-Teodoro, J. Camacho, G. Maciá-Fernández, J.A. Gómez-Hernández, and V.J. López-Marín. 2022. A novel zero-trust network access control scheme based on the security profile of devices and users. *Computer Networks* 212 (2022), 109068. https://doi.org/10.1016/j.comnet.2022.109068

[5] Gartner, Inc. 2019. The Future of Network Security Is in the Cloud. https://www.gartner.com/en/documents/3957375. Accessed: 2025-04-09.

[6] ISO. 2015. ISO/IEC 11889-1:2015 – Information technology – Trusted Platform Module Library – Part 1: Architecture. Accessed: April 9, 2025.

[7] Abdallah Moubayed, Ahmed Refaey, and Abdallah Shami. 2019. Software-defined perimeter (sdp): State of the art secure solution for modern networks. *IEEE network* 33, 5 (2019), 226–233.

[8] OpenVPN Project. n.d.. OpenVPN Protocol. https://openvpn.net/community-resources/openvpn-protocol/ Accessed: 19 Mar. 2025.

[9] OpenVPN Project. n.d.. Secure Remote Access - OpenVPN. https://openvpn.net/solutions/use-cases/secure-remote-access/ Accessed: 19 Mar. 2025.

[10] SPIFFE Project. n.d.. SPIRE Concepts - SPIFFE. https://spiffe.io/docs/latest/spire-about/spire-concepts/ Accessed: 19 Mar. 2025.

[11] V Stafford. 2020. Zero trust architecture. *NIST special publication* 800, 207 (2020), 800–207.

[12] Trusted Computing Group. 2018. TCG PC Client Platform Firmware Profile Specification Version 1.06 Revision 52. https://trustedcomputinggroup.org/wp-content/uploads/TCG-PC-Client-Platform-Firmware-Profile-Version-1.06-Revision-52_pub-3.pdf. Accessed: April 9, 2025.

[13] Silafu Yiliyaer and Yoohwan Kim. 2022. Secure access service edge: A zero trust based framework for accessing data securely. In *2022 IEEE 12th Annual Computing and Communication Workshop and Conference (CCWC)*. IEEE, 0586–0591.