

# An Interoperable Zero Trust Federated Architecture for Tactical Systems

Alexandre Poirrier<sup>1,2</sup>, Laurent Cailleux<sup>2</sup> and Thomas Heide Clausen<sup>1</sup>

<sup>1</sup> *École polytechnique, Palaiseau, France*

<sup>2</sup> *Direction Générale de l'Armement, Rennes, France*

## Abstract

Military and tactical systems are heterogeneous, encompassing devices with low computing power and network capacity. Such networks can be secured by following the zero trust paradigm: every access request to resources is verified, without relying on inherent trust between the requester and the resource. However, operational needs can require different domains, such as different nations in a coalition, to federate, to enable sharing of resources between domains. This contradicts the principle of zero trust, as information on the requester cannot be verified by the domain offering the resource, and therefore access inherently relies on trust between domains.

This paper explores a solution for federating tactical network architectures, while following the zero trust paradigm. In particular, due to the power constraints on devices composing tactical architectures, the presented solution does not require invasive software to be installed in requester devices.

## Keywords

Federation, Internet of Military Things, Software-Defined Perimeters, Zero Trust

## 1. Introduction

Tactical networks for military operations are heterogeneous: They have different objectives, *e.g.*, surveillance, reconnaissance, or tactical mission execution, and are composed of different devices, such as satellites, sensors, Unmanned Aerial Vehicles (UAV), or battleships. They also carry heterogeneous communication, such as voice, video, or text [1]. They combine different technologies, such as combat-net radio (CNR), mobile networks (LTE), or satellite networks reaching Low Earth Orbit (LEO) satellites.

### A Need for Interoperability


During the Afghanistan intervention, the necessity for a coalition mission network to share information between allied nations arose. A first level of interoperability was reached with the Afghanistan Mission Network (AMN), facilitating decision making by enabling access to more complete information [2]. In 2012, the NATO Military Committee proposed an approach


---

*C&ESAR'23: Computer & Electronics Security Application Rendezvous, Nov. 21-22, 2023, Rennes, France*

✉ alexandre.poirrier@polytechnique.org (A. Poirrier); laurent.cailleux@intradef.gouv.fr (L. Cailleux); thomas.clausen@polytechnique.edu (T. H. Clausen)

🆔 0000-0001-7601-3199 (A. Poirrier); 0000-0002-7400-8887 (T. H. Clausen)

 © 2023 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

to improve the level of interoperability provided by the AMN, called Federated Mission Network (FMN) [3]. Widely accepted by NATO nations, the NATO FMN Implementation Plan (NFIP) was designed, and the fourth version endorsed in 2015 [4]. The need for interoperability has been confirmed to be fundamental for armed forces by NATO leaders in 2016 [5].

FMN aims at creating policies, procedures, and components for sharing data and applications, between Alliance nations and partners, and amongst communities of interest. The FMN consists of three parts: a framework, serving as a template for mission networks, a number of mission network instances, and a governance overseeing the framework and the specific mission network instances. The NFIP describes how to have interoperability in military networks, by proposing a service-oriented architecture. Development is staggered into several ‘spirals’, which add more capabilities providing interoperability between nations at operative and tactical levels [4].

Moreover, a need for Multi-Domain Operations (MDO) has been identified by NATO, which expands the requirements for interoperability across the five operational domains and between nations [6].

### A Need for Security

Perimetric security is vulnerable to insider threats, and to lateral movements. These considerations, and the plethora of breaches from the 2010s, gave rise to the concept of ‘deperimeterization’, from the Jericho Forum in 2004 [7], and then the zero trust architecture, introduced by Forrester in 2010 [8]. This paradigm change has been adopted by the U.S. Department of Defense (DoD) with the ‘Black Core’ architecture [9] in 2007.

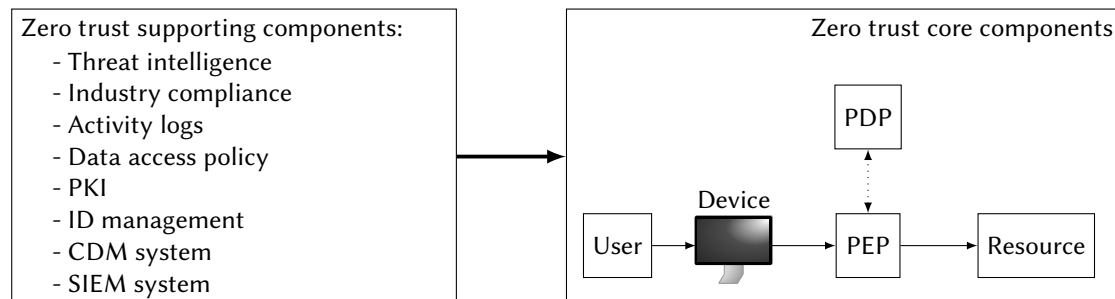
Following the Solarwinds attacks in 2020 [10], the U.S. government has made the transition to zero trust compulsory [11, 12], requiring federal agencies to meet zero trust standards by the end of the year 2024.

The goals of zero trust architectures are to secure and protect information, systems, and infrastructures against malicious activities, including organization information on non-owned networks [13]. This implies that several principles are to be followed by zero trust architectures [14, 15]:

1. **Authentication:** every entity (user, device, application, etc.) needs to be authenticated;
2. **Least-privilege, per-session, and dynamic authorization:** access to resources is granted following a least privilege policy, for a limited time, with authorization taking into account contextual and environmental information;
3. **Segmentation of access and encryption:** access to resources is evaluated and authorized for smallest possible pieces, and communication to resources is always secured.
4. **Monitoring:** the infrastructure, entities, and resources are constantly monitored.

According to [14], every resource is to be protected by a Policy Enforcement Point (PEP). As depicted in figure 1, every connection request is evaluated by a Policy Decision Point (PDP), which grants or denies access to the resource, by considering information on the requester, the environment, and threats. The decision of the PDP is enforced by the PEP.

Additional information considered by the PDP is provided by supporting components. Authentication is performed with an ID management (IdM) system, responsible for the identity of every entity and device in the organization. Identity refers to the set of attributes that describe



**Figure 1:** Zero trust architecture [14].

entities and devices within a given context [16]. IdM systems can rely on other systems, such as a Public Key Infrastructure (PKI), to associate artifacts, such as certificates, with entity identities.

Moreover, every asset and device of the architecture is continuously monitored. This can be performed using a continuous diagnostics and mitigation (CDM) system [17], which collects security information on devices in the infrastructure, and a security information and event management (SIEM) system [18], which analyses security events collected by the CDM [14].

## Problem Statement

Zero trust implies that every access to a resource needs to be verified, and that access must not be granted based on implicit trust.

However, in a federation, every domain is responsible for authenticating and monitoring its own entities and devices. Therefore, if an entity requests access to a federated resource, information describing this entity, its devices, and its context, is provided by the requester domain, and the resource domain needs to trust that information. This weakens zero trust security guarantees, as granting access requires implicit trust between federation members [19].

Thus, this paper explores how it is possible to create a zero trust federation, in which every access to a resource is explicitly verified, without implicitly trusting federation partners.

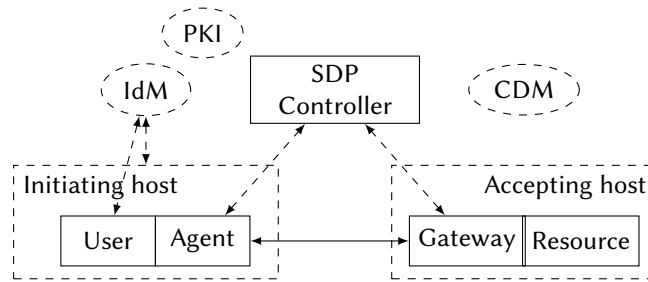
## 1.1. Related Work

### Interoperability of Tactical Systems

Interoperability between domains is ensured by standardization programs such as the NATO Interoperability Standards and Profiles (NISP) [20] and the Multilateral Interoperability Program (MIP) [21].

For enabling legacy systems to meet federation standards, legacy traffic can be encapsulated or converted by middle components. This allows minimal changes from original architectures, at the expense of higher operational latencies, and may increase the attack surface of the legacy device [22].

Federated Identity Management associates several service providers, and identity providers, for authentication and authorization [23]. It enables users to connect to different service



**Figure 2:** Architecture of SDP [25]. Dotted lines are control channels, and solid lines are data channels.

providers, without authenticating to each service. Service providers delegate identity management to identity providers, which offers flexibility and scalability. However, trust in providers is required [24].

### Software-Defined Perimeters [25]

Software-Defined Perimeters (SDP) are an abstract zero trust architecture which can be used for tactical networks. SDP-based solutions are recommended for agencies having many field agents who do not have continuous internet access, and utilize many sensors and IoT devices [26]. Moreover, SDP is based on Software-Defined Networking (SDN), which offers advanced management features that can be used for interoperability [27].

The components of an SDP architecture are depicted in figure 2. An SDP architecture creates an overlay network on top of the existing network infrastructure. A central component, called the SDP Controller, oversees the network from the control plane, and grants or denies access to resources. Every device contains a device agent, which is in charge of monitoring the device and of establishing connections with the SDP controller and SDP resources. The terminology for entities and devices trying to access an SDP resource is ‘Initiating Host’ (IH). SDP resources are protected by a gateway, which filters every communication with the resource, and communicates with the SDP Controller. By default, the gateway blocks all traffic. The gateway and the resource form an ‘Accepting Host’ (AH). Secure communication channels in SDP are composed of two mechanisms: first Single Packet Authorization (SPA), which ensures only pre-authorized entities can create the channel, and then mutual TLS (mTLS), to create end-to-end encrypted and authenticated channels.

SDP can be specialized for Internet-of-Things (IoT) applications [28]. Replacing the mTLS connection with a secure communication based on lightweight cryptography reduces the computing load of resource-constrained IoT devices [29]. Keys used in lightweight cryptography can be derived from Physically Unclonable Functions (PUF), which use unforgeable characteristics of the physical channel on which IoT devices communicate to establish security [30]. Comparing the actual behavior of IoT devices, with their expected behavior as defined by manufacturers, evaluates the security posture of IoT devices [31]. Machine learning techniques further automatize anomaly detection [32].

## **Zero Trust Federations**

The possibility to create a zero trust architecture that can be federated with other architectures, zero trust or not, is evaluated in [33] for the U.S. Air Force. Six solutions are proposed. In the first four (ZTE Federation, ZTE-like Federation, Identity Credential Federation, and Weak Identity solutions), the information describing the requester and its device are provided by the requester domain, and trusted by the resource domain. The difference between those four architectures is the zero trust maturity of the federated architecture: a more advanced maturity provides higher security guarantees. The Ad Hoc Federation involves a central source, which determines which resource can be shared with which entity. Similarly, the Person-to-Person sharing enables users to share data with other users following a chain of command.

There are three solutions for providing zero trust in a federation, without trusting federated members [34]:

1. Installation of a trusted component in every device accessing resources, including devices from partners, to evaluate the security posture of devices.
2. Standardized hierarchical trust architecture: trust in other domains comes from trust in a supervising organization.
3. Third-party negotiation: a trusted third-party collects and shares information on every architecture.

A federated zero trust architecture is proposed by [35] following the third of the solutions. In this architecture, a third party, called the Context Attribute Provider (CAP), installs an agent on every device of federated architectures, and monitors those devices. The CAP can then be used by PDP to provide contextual information in access requests. The CAP can be split into two components, one component responsible for contextual information collection, and another component for contextual information exploitation [36].

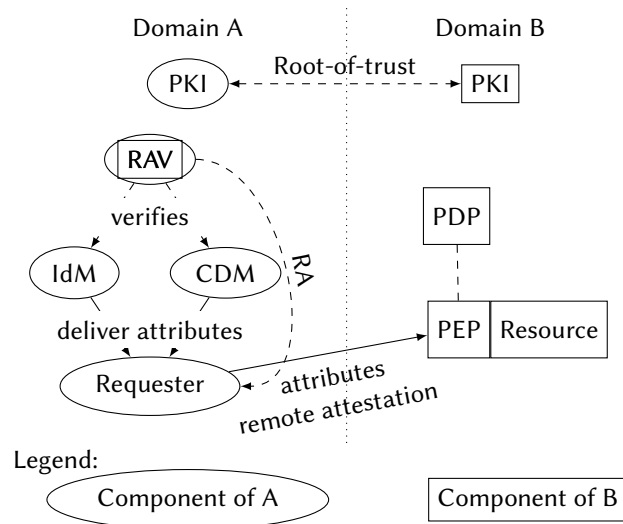
### **1.2. Statement of Purpose**

Existing solutions for federating zero trust architectures either require trust in a third party or in partners, or employ invasive techniques such as the installation of monitoring software on every device. The former solutions imply an inherent trust, which is never challenged during the execution of the mission. This is in opposition to the zero trust principles, stating that access requests need to be verified. Moreover, in the military context of the FMN, the need for sovereignty and control of equipment prevents the use of intrusive techniques and the establishment of trust between nations, or in a third party.

This paper presents a non-intrusive federation of architectures, which ensures interoperability between federated domains while following zero trust principles.

### **1.3. Paper Outline**

The remainder of this paper is organized as follows. Section 2 presents an abstract solution to federate zero trust architectures, which uses a component called remote attestation. Section 3 presents a proof-of-concept deployment of a zero trust federation for SDP architectures. Section 4 concludes this paper.



**Figure 3:** Proposed zero trust federated architecture.

## 2. A Proposed Federated Zero Trust Architecture

This section proposes a framework for how several architectures, following zero trust principles, as presented in figure 1, can federate. The basic idea is as follows: Each domain in the federation operates independently, and manages the identity, authentication, and monitoring of their infrastructure, devices, and entities, and control access to their resources.

Moreover, if an entity, called the *requester*, requires access to a resource from another domain, the domain of the resource can authorize or deny access based on *verified* information about the requester and its device, without inherently trusting information provided by the domain of the requester, relying on a trusted third-party, or installing intrusive software on the device of the requester. Verified information includes the identity of the requester, information on its device, and contextual information, and is produced by the requester architecture.

The resulting federated architecture is depicted in figure 3. Its components are detailed in the following sections.

### 2.1. Interoperability of Identity and Monitoring Information

*Identity* refers to an attribute, or a set of attributes, that uniquely describe an entity (or device) within a given context. Authentication is the process of establishing confidence in user identities [16].

Identity attributes (e.g., country of origin, clearance level, military rank, etc.), provided by the IdM, are the basis for attribute-based access control [37]. Attributes can also represent contextual information such as geolocalization, behavior evaluation, or time of day. FMN and MIP define standards for attributes. For example, STANAG 1059 defines country codes, and STANAG 2116 defines ranks for military personnel. Those standards are used by NATO nations to understand each other.

An alternative to standards, *e.g.*, if a specific standard does not exist, is to create agreements between members in the federation, similar to the identity mappings proposed by [33].

## 2.2. Root-of-trust Establishment

In a zero trust architecture, trust is established between entities, devices, and the architecture when entities and devices are on-boarded. The root-of-trust in this relationship is implementation dependent. Examples of root-of-trust include a secret key linked to a certificate uploaded on a device, a password given to a user, or a secret shared between the device of a user and the IdM for multi-factor authentication.

Moreover, there is necessarily trust in the supply chain, as the hardware used is not always created or managed by the organization. Such trust can be based on certification, and does not exclude continuous verification.

Zero trust federation requires a similar root-of-trust, exchanged when domains are federating. For the federation framework proposed in this paper, master certificates for each domain are exchanged to authenticate information produced by each domain. Similarly to certificates and keys shared with internal entities and devices, this root-of-trust shared between domains does not imply trust. Thus, additional verifications are required to grant access to resources.

Further, a new component, called the *remote attestation verifier* (RAV), described in section 2.3, is deployed in requester domains. This component is an element of trust between members of the federation: it monitors components of the requester domain responsible for authentication and context evaluation, while being managed by the resource domain, with an identity provided by the resource domain.

To ease the certification and deployment process in the requester domain, the RAV can be a component designed and implemented by the organization of the requester. The resource organization can verify its trustworthiness before federating, similarly to other supply chain components used in the architecture.

## 2.3. Attribute Verification

When a requester requests access to a resource from another domain in the federation, attributes representing the identity of the requester, its device, and the access request context are provided to the resource domain.

Those attributes are provided, and authenticated, by the requester domain.

However, there is no inherent trust between the requester and the resource domains. The root-of-trust does not guarantee trust between the domains, as components from the requester architecture can be compromised during the mission. Thus, the resource domain needs to continuously verify the integrity of the provided attributes.

### 2.3.1. Remote Attestation

Continuous and non-invasive verification is performed with remote attestation.

Remote ATtestation procedureS (RATS) [38] produces information on an evaluated system, the *attester*, to another remote system, the *relying party*. This information can be used by the

relying party to evaluate if the attester is trustworthy [39, 40]. Remote attestation involves a component called the *verifier*, which appraises the evidence submitted by the attester [40].

Remote attestation has been evaluated by the European commission for usage in health-care [41]. Depending on the constraints on the requester domain, remote attestation can either rely on dedicated hardware, such as a Trusted Platform Module (TPM) [42], or be software-based without requiring special hardware [43].

While remote attestation can be used for IoT systems [44], this is not required in the solution proposed in this paper, as the security posture of only control plane systems, such as the identity provider, is attested.

### 2.3.2. Access Requests

In the proposed federation framework, the resource architecture uses remote attestation to verify the integrity and the security posture of the IdM and CDM systems of the requester architecture. Remote attestation establishes trustworthiness in those systems, and by extension in the produced attributes.

Depending on the zero trust maturity of the resource architecture, the RAV can produce either a boolean value, indicating if the evaluated systems are in compliance with a security policy, or richer information, usable by the PDP for evaluating the access request.

## 3. Proof-of-Concept for a Federated Architecture

A proof-of-concept has been developed to illustrate how it is possible to federate two zero trust architectures.

The proof-of-concept is based on several building blocks:

1. A zero trust SDP architecture from Waverley Labs<sup>1</sup>, which has been extended to offer more functionalities and to provide an interface with the other components;
2. An identity provider, Keycloak<sup>2</sup>, to provide SAML authentication;
3. A proof-of-concept implementation of the Challenge-Response Remote Attestation<sup>3</sup> defined by the IETF RATS Working Group [45], based on simulated TPMs.

Every component in the proof-of-concept has been built as a Docker container, which simulates a server or device. Docker networks connect the containers to simulate network interactions. Figure 4 presents the complete architecture of the proof-of-concept, with details about each device, process, cryptographic keys and certificates held by each entity. The proof-of-concept presents two domains, *A* and *B*, each implementing an SDP zero trust architecture. The domains are presented asymmetrically: domain *A* hosts an Initiating Host needing to access a service from domain *B*.

More precisely, domain *A* is composed of an Initiating Host, *i.e.*, a device and a user, of a controller, of an identity provider, and of a service. Domain *B* is composed of a controller,

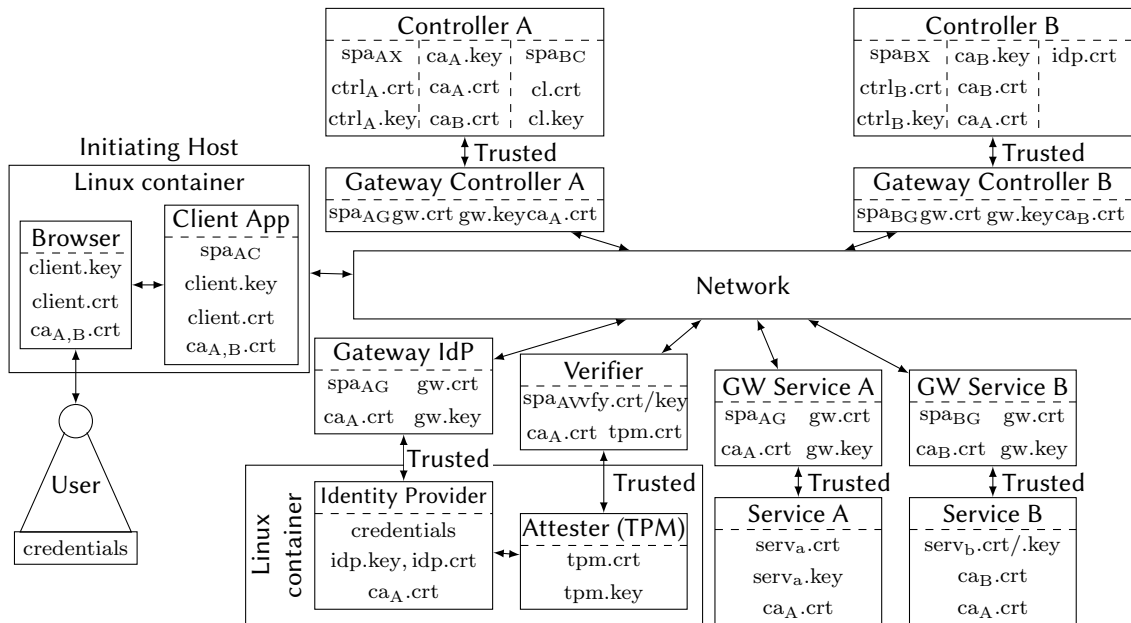
---

<sup>1</sup><https://github.com/WaverleyLabs/SDPcontroller>

<sup>2</sup><https://www.keycloak.org/>

<sup>3</sup><https://github.com/Fraunhofer-SIT/charra>





**Figure 4:** Architecture of the proof-of-concept. Every controller, service, or gateway is a Linux container. Keys held by each process are depicted.

and a service. As described in section 2, a verifier is also deployed in domain *A*, to guarantee the integrity of the identity provider. The user has been given credentials during on-boarding, which consist in this proof-of-concept of a username and a password. The user uses a device, and interacts with it through a browser. The browser connects to the client application, which is a web server installed on the device, and which is configured with cryptographic material for SDP operations.

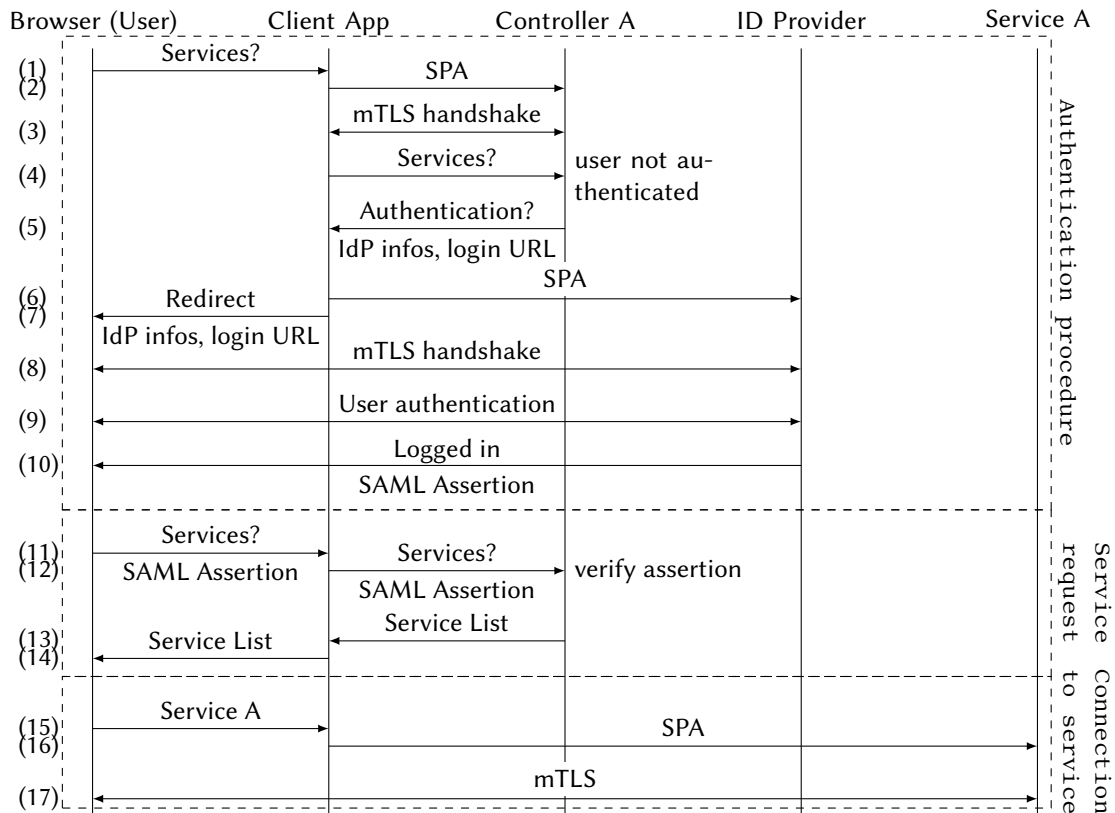
SDP controllers and services are protected by SDP gateways. Each controller is an SDP controller for their domain, communicating with clients and gateways to authorize connections. Moreover, they are the root certificate authority of the public key infrastructure for their domain. For users from domain *A* requiring access to services from domain *B*, Controller *A* acts as an Initiating Host for domain *B*.

The identity provider for domain *A* runs in a Linux container. In this container, a virtual TPM has been deployed and is used for the remote attestation procedure with the verifier, as described in section 3.2.

### 3.1. User Authentication

An SDP IH is authenticated using its private keys. Thus, only the device, which holds the private keys, is authenticated.

The SDP protocol has been extended to include the interaction between users and the identity provider to authenticate users. The protocol is depicted in figure 5, for an SDP architecture



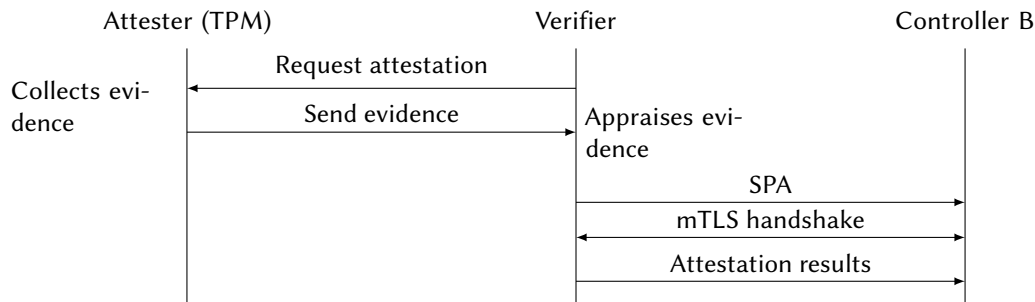
**Figure 5:** Extension of the Waverley Labs SDP architecture: service list, and integration of the identity provider. The first part of the sequence diagram shows the user authentication process. The second part shows the controller sending the service list to the client. The third part shows the client accessing the service.

with a single domain.

The first part of the figure shows the authentication procedure. The user requests a list of services it can access. The device authenticates itself to access the controller, using the SPA+mTLS SDP connection (arrows 2 and 3), and requests the list of available services. As the user is not yet authenticated, the controller answers the query with an authentication request, which contains information on how to reach the IdP (such as its IP address or domain name), and a login URL containing a SAML authentication request (arrow 5). Moreover, the controller informs the IdP gateway that the device of the client can access the IdP (which is not represented in the figure).

After receiving the authentication request, the device of the user sends an SPA packet to the IdP to authorize connections (arrow 6), and the user is redirected to the login URL (arrow 7). The user then authenticates themselves to the IdP, which returns a signed SAML assertion describing the user (arrow 10).

This SAML assertion is then forwarded to, and verified by, the controller (arrows 11 and 12), which answers with the list of services available to the user.



**Figure 6:** Remote attestation protocol, performed regularly.

Finally, the third part of the figure depicts the usual SDP connection process between the user and the federated service.

### 3.2. Remote Attestation

The remote attestation procedure follows the Challenge/Response interaction model [45]: An attester process runs continuously to attest the integrity of this server, and the verifier queries the attester regularly to collect evidence on the state of the device. The verifier then appraises the evidence to generate attestation results, which are sent to the controller of domain  $B$ . Those interactions are depicted in figure 6.

The remote attestation procedure follows the proof-of-concept implementation from the IETF Working Group on Remote Attestation. The verifier queries regularly the attester. Evidence appraised by the verifier are Platform Configuration Registers [46], which are read-only registers which record software state. In case of attestation failure, the access rights of the federated clients are revoked.

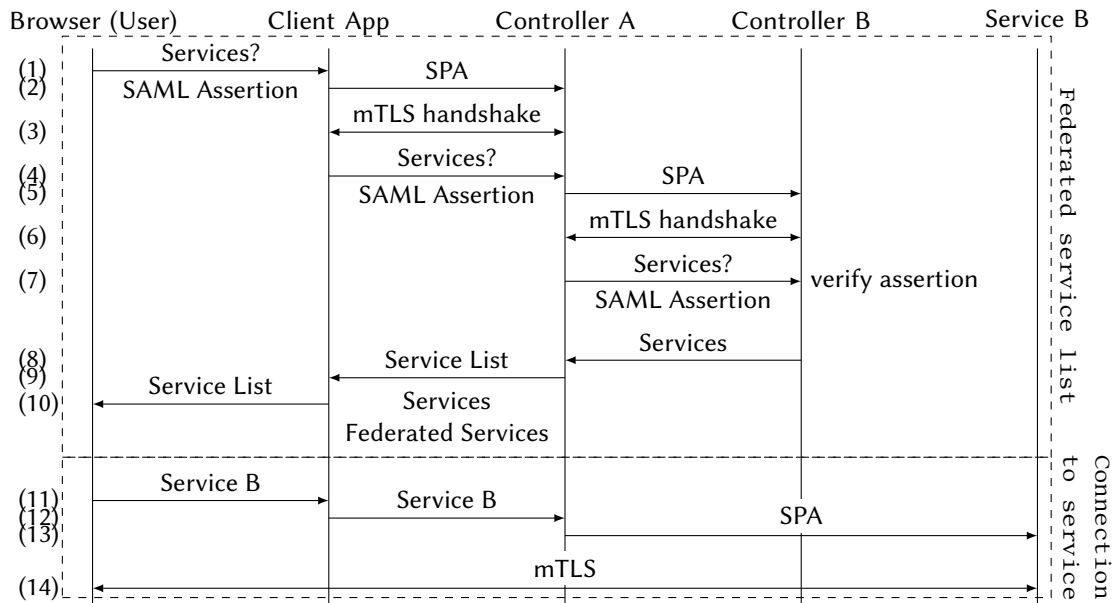
### 3.3. Federation Protocol

The remote attestation procedure presented in the previous section proves to controller  $B$  that attributes provided by the identity provider of  $A$  are correctly derived, and thus the SAML assertions are trustworthy.

Figure 7 presents how an authenticated user gets a list available services, including federated services, and how the user can access a federated service. In order to make the connection possible between the client and service  $B$ , both entities have been given the root certificates of both PKI, as depicted in figure 4.

When designing the architecture, two solutions have been considered. In the first solution, the client from domain  $A$  would directly contact controller  $B$  to get a list of federated services, and then connect to service  $B$  as an Initiating Host from domain  $B$ . However, this solution requires secrets to be shared beforehand between the user, its device, and domain  $B$ , which may not be possible. Thus, a second solution, presented afterwards, has been selected.

Similarly to the protocol with one domain, the user requests a list of available services to its controller through the client application, by forwarding the SAML assertion (arrows 1 to 4).



**Figure 7:** SDP Federation: list of federated services and access to a federated service. The first part describes the protocol for getting the list of federated services, including federated services, and the second part shows how the client connects to a federated service.

The controller verifies this SAML assertion and collects the list of services for its domain that the user can access. Moreover, it queries the federated controller for a list of services for this user. To do so, controller *A* acts as an Initiating Host for controller *B*, and initiates a connection using SPA and mTLS (arrows 5 and 6). Then, it sends a request for available services containing the SAML Assertion and the IP address of the device of the user (arrow 7). Controller *B* verifies the assertion using the public key of the identity provider of domain *A*, and replies with the federated services that the user is authorized to access (arrow 8). Moreover, controller *B* contacts the gateways of authorized services to add the IP address of the device of the client as an authorized IP.

Once the client gets the list of all services, it can connect to a federated service. Because no secrets have been exchanged between the client and domain *B*, the SPA packet is sent by controller *A* instead of the client (arrow 13). However, the mTLS connection can be performed directly between the user and the service, as they both have access to the root certificates.

### 3.4. Applications in Military Networks

The proof-of-concept presented in this section uses Docker containers networked together. The controllers, identity provider, their associated gateways, and the verifier, are part of the control plane, whereas the client, the services, and the gateways protecting the services are part of the data plane.

In this proof-of-concept, the client is a (human) user. It models for devices such as laptops, phones, or IoT devices embedding a human-machine interface. Accepting Hosts can be servers,

or IoT devices, such as a Raspberry Pi running both the gateway process and the service. Clients and services are deployed on the battlefield.

Two models have been considered for the control plane. In the first model, every component, including components from the control plane, are also deployed on the battlefield. This requires enough computing power, and available and stable bandwidth, for the controllers and identity providers. This solution can be supported by the MDO effort, for orchestrating activities across multiple domains in joint operations.

In the second model, the control plane is deployed in a combat cloud [47], which is a network dedicated to data distribution and information sharing within a battlefield, or deployed at the operational level at the edge of the tactical network. In this model, communication between controllers, verifier, and identity provider is more stable, and there are less resource constraints on those components. However, connectivity between the data plane and the control plane is less stable. This can be mitigated by increasing the lifetime of sessions. Authentication of users on their device can be performed before starting the operation and deploying users. Once deployed, clients and gateways need to contact their controller once to update policies, then sessions can be established between IHs and AHs without needing controllers. Contact with controllers are performed punctually to update access information.

## 4. Conclusion

The desire for cooperation between nations and operational domains creates a need for interoperable federated architectures. The zero trust paradigm provides principles to secure data and services from unauthorized access, with fine-grain compartmentalization and least privilege access policies.

Unfortunately, state-of-the-art zero trust federated architectures either require the installation of intrusive software in every devices, or assume an inherent trust between federation partners, or in a third party.

This paper has proposed a novel method for federating zero trust architectures, by leveraging remote attestation for continuous verification of the integrity and the authenticity of federated requester attributes. With this approach, the resource domain can verify the information provided by federation partners, instead of implicitly trusting this information. Thus, zero trust principles are preserved, without needing intrusive software to be installed on every requester device. This novel architecture has been illustrated with a proof-of-concept implementation and deployment, leveraging SDP architectures and TPM-based remote attestation.

## References

- [1] N. Suri, A. Hansson, J. Nilsson, P. Lubkowski, K. Marcus, M. Hauge, K. Lee, B. Buchin, L. Misirhoglu, M. Peuhkuri, A realistic military scenario and emulation environment for experimenting with tactical communications and heterogeneous networks, in: 2016 International Conference on Military Communications and Information Systems (ICMCIS), IEEE, 2016. doi:doi: 10.1109/icmcis.2016.7496568.

- [2] J. Stoltenberg, The secretary general's annual report 2014, 2014. URL: [https://www.nato.int/cps/en/natohq/opinions\\_116854.htm](https://www.nato.int/cps/en/natohq/opinions_116854.htm).
- [3] NATO, Federated mission networking, 2014. URL: <https://dnbl.ncia.nato.int/FMNPUBLIC/SitePages/Home.aspx>.
- [4] M. R. Brannsten, F. T. Johnsen, T. H. Bloebaum, K. Lund, Toward federated mission networking in the tactical domain, *IEEE Communications Magazine* 53 (2015) 52–58. doi:doi: 10.1109/mcom.2015.7295463.
- [5] W. B. King, Army Europe highlights interoperability at communications conference, U.S. Army (2016). URL: <https://www.army.mil/article/161837/>.
- [6] NATO, Multi-domain operations: Enabling NATO to out-pace and out-think its adversaries, 2022. URL: <https://www.act.nato.int/articles/multi-domain-operations-out-pacing-and-out-thinking-nato-adversaries>.
- [7] P. Simmonds, De-perimeterisation, 2004. URL: <https://www.blackhat.com/presentations/bh-usa-04/bh-us-04-simmonds.pdf>.
- [8] J. Kindervag, No More Chewy Centers: Introducing The Zero Trust Model Of Information Security, Technical Report, Forrester, 2010. URL: <https://media.paloaltonetworks.com/documents/Forrester-No-More-Chewy-Centers.pdf>.
- [9] J. G. Grimes, Vision for a Net-Centric, Service-Oriented DoD Enterprise, Technical Report, Department of Defense Global Information Grid, 2007. URL: <https://www.acqnotes.com/Attachments/DoDGIGArchitecturalVision,June07.pdf>.
- [10] J. Rundle, Solarwinds, microsoft hacks prompt focus on zero-trust security, *The Wall Street Journal* (2021). URL: <https://www.wsj.com/articles/solarwinds-microsoft-hacks-prompt-focus-on-zero-trust-security-11619429402>.
- [11] J. Biden, Improving the nation's cybersecurity, Executive order 14028, 2021. URL: <https://www.federalregister.gov/documents/2021/05/17/2021-10460/improving-the-nations-cybersecurity>.
- [12] S. D. Young, Moving the U.S. government toward zero trust cybersecurity principles, Memorandum M-22-09, 2022. URL: <https://www.whitehouse.gov/wp-content/uploads/2022/01/M-22-09.pdf>.
- [13] R. Freter, Zero Trust Reference Architecture, Technical Report, Department of Defense, 2022. URL: [https://dodcio.defense.gov/Portals/0/Documents/Library/\(U\)ZT\\_RA\\_v2.0\(U\)\\_Sep22.pdf](https://dodcio.defense.gov/Portals/0/Documents/Library/(U)ZT_RA_v2.0(U)_Sep22.pdf).
- [14] S. Rose, O. Borchert, S. Mitchell, S. Connelly, Zero Trust Architecture, Technical Report 800-207, National Institute of Standards and Technology, 2020. URL: <https://csrc.nist.gov/publications/detail/sp/800-207/final>. doi:doi: 10.6028/nist.sp.800-207.
- [15] CISA, Zero Trust Maturity Model, Technical Report, CISA, 2021. URL: <https://www.cisa.gov/zero-trust-maturity-model>.
- [16] P. A. Grassi, M. E. Garcia, J. L. Fenton, Digital identity guidelines: revision 3, Technical Report SP 800-63-3, National Institute of Standards and Technology, 2020. URL: <https://csrc.nist.gov/publications/detail/sp/800-63-3/final>. doi:doi: 10.6028/nist.sp.800-63-3.
- [17] Cybersecurity & Infrastructure Security Agency (CISA), Continuous diagnostics and mitigation (CDM) program, 2012. URL: <https://cisa.gov/cdm>.
- [18] S. Bhatt, P. K. Manadhata, L. Zomlot, The operational role of security information and event management systems, *IEEE Security & Privacy* 12 (2014) 35–41. doi:doi: 10.1109/

msp.2014.103.

- [19] K. Strandell, S. Mittal, Risks to zero trust in a federated mission partner environment, arXiv preprint arXiv:2211.17073 (2022). doi:doi: 10.48550/ARXIV.2211.17073. arXiv:2211.17073.
- [20] NATO Standardization Office, Nato interoperability standards and profiles, 2021. URL: <https://nhqc3s.hq.nato.int/Apps/Architecture/NISP/introduction.html>.
- [21] M. I. P. (MIP), MIP4-IES, 2021. URL: <https://www.mip4ies.com/>.
- [22] M. Pradhan, N. Suri, C. Fuchs, T. H. Bloebaum, M. Marks, Toward an architecture and data model to enable interoperability between federated mission networks and IoT-enabled smart city environments, IEEE Communications Magazine 56 (2018) 163–169. doi:doi: 10.1109/mcom.2018.1800305.
- [23] S. Shim, G. Bhalla, V. Pendyala, Federated identity management, Computer 38 (2005) 120–122. doi:doi: 10.1109/mc.2005.408.
- [24] D. W. Chadwick, Federated identity management, in: Foundations of Security Analysis and Design V, Springer Berlin Heidelberg, 2009, pp. 96–120. doi:doi: 10.1007/978-3-642-03829-7\_3.
- [25] J. Garbis, J. Koilpillai, Software-Defined Perimeter (SDP) Specification v2.0, Technical Report, Cloud Security Alliance, 2022. URL: <https://cloudsecurityalliance.org/artifacts/software-defined-perimeter-zero-trust-specification-v2/>.
- [26] K. Uttecht, Zero Trust (ZT) Concepts for Federal Government Architectures, Technical Report, MASSACHUSETTS INST OF TECH LEXINGTON, 2020. URL: <https://apps.dtic.mil/sti/citations/AD1106904>.
- [27] S. Khan, F. K. Hussain, Software-defined overlay network implementation and its use for interoperable mission network in military communications, in: Advanced Information Networking and Applications, Springer International Publishing, 2022, pp. 554–565. doi:doi: 10.1007/978-3-030-99584-3\_48.
- [28] A. Refaey, A. Sallam, A. Shami, On IoT applications: a proposed SDP framework for MQTT, Electronics Letters 55 (2019) 1201–1203. doi:doi: 10.1049/el.2019.2334.
- [29] S. W. Shah, N. F. Syed, A. Shaghaghi, A. Anwar, Z. Baig, R. Doss, LCDA: Lightweight continuous device-to-device authentication for a zero trust architecture (ZTA), Computers & Security 108 (2021) 102351. doi:doi: 10.1016/j.cose.2021.102351.
- [30] Z. Huang, Q. Wang, A PUF-based unified identity verification framework for secure IoT hardware via device authentication, World Wide Web 23 (2019) 1057–1088. doi:doi: 10.1007/s11280-019-00677-x.
- [31] E. Lear, R. Droms, D. Romascanu, Manufacturer Usage Description Specification, RFC 8520, 2019. URL: <https://www.rfc-editor.org/info/rfc8520>. doi:doi: 10.17487/RFC8520.
- [32] S. H. Haji, S. Y. Ameen, Attack and anomaly detection in IoT networks using machine learning techniques: A review, Asian Journal of Research in Computer Science (2021) 30–46. doi:doi: 10.9734/ajrcos/2021/v9i230218.
- [33] W. R. Simpson, K. E. Foltz, Maintaining zero trust with federation, International Journal of Emerging Technology and Advanced Engineering 11 (2021) 17–32. doi:doi: 10.46338/ijetae0521\_03.
- [34] K. Olson, E. Keller, Federating trust, in: Proceedings of the SIGCOMM '21 Poster and Demo Sessions, ACM, 2021. doi:doi: 10.1145/3472716.3472865.

- [35] K. Hatakeyama, D. Kotani, Y. Okabe, Zero trust federation: Sharing context under user control towards zero trust in identity federation, in: 2021 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (Per-Com Workshops), IEEE, IEEE, 2021, pp. 514–519. doi:doi: 10.1109/percomworkshops51409.2021.9431116.
- [36] M. Hirai, D. Kotani, Y. Okabe, Linking contexts from distinct data sources in zero trust federation, in: Lecture Notes in Computer Science, Springer Nature Switzerland, 2023, pp. 136–144. doi:doi: 10.1007/978-3-031-25467-3\_9.
- [37] B. Bezawada, K. Haefner, I. Ray, Securing home IoT environments with attribute-based access control, in: Proceedings of the Third ACM Workshop on Attribute-Based Access Control, ACM, 2018. doi:doi: 10.1145/3180457.3180464.
- [38] L. Gu, X. Ding, R. H. Deng, B. Xie, H. Mei, Remote attestation on program execution, in: Proceedings of the 3rd ACM workshop on Scalable trusted computing, ACM, 2008. doi:doi: 10.1145/1456455.1456458.
- [39] G. Coker, J. Guttman, P. Loscocco, A. Herzog, J. Millen, B. O’Hanlon, J. Ramsdell, A. Segall, J. Sheehy, B. Sniffen, Principles of remote attestation, International Journal of Information Security 10 (2011) 63–81. doi:doi: 10.1007/s10207-011-0124-7.
- [40] H. Birkholz, D. Thaler, M. Richardson, N. Smith, W. Pan, Remote ATtestation procedureS (RATS) Architecture, RFC 9334, 2023. URL: <https://www.rfc-editor.org/info/rfc9334>. doi:doi: 10.17487/RFC9334.
- [41] A. Vahidi, N. Paladi, Remote attestation of workloads in ITEEs, Technical Report D4.2, ASCLEPIOS, European Commission, 2020. URL: <https://ec.europa.eu/research/participants/documents/downloadPublic?documentIds=080166e5cfd3eb3b&appId=PPGMS>.
- [42] D. Schellekens, B. Wyseur, B. Preneel, Remote attestation on legacy operating systems with trusted platform modules, Science of Computer Programming 74 (2008) 13–22. doi:doi: 10.1016/j.scico.2008.09.005.
- [43] A. Seshadri, A. Perrig, L. van Doorn, P. Khosla, SWATT: software-based attestation for embedded devices, in: IEEE Symposium on Security and Privacy, 2004. Proceedings. 2004, IEEE, 2004. doi:doi: 10.1109/secpri.2004.1301329.
- [44] M. Conti, E. Dushku, L. V. Mancini, M. Rabbani, S. Ranise, Remote attestation as a service for IoT, in: 2019 Sixth International Conference on Internet of Things: Systems, Management and Security (IOTSMS), IEEE, 2019. doi:doi: 10.1109/iotsms48152.2019.8939224.
- [45] H. Birkholz, M. Eckel, W. Pan, E. Voit, Reference interaction models for remote attestation procedures, IETF, 2023. URL: <https://datatracker.ietf.org/doc/draft-ietf-rats-reference-interaction-models/>.
- [46] W. Arthur, D. Challener, K. Goldman, Platform configuration registers, in: A Practical Guide to TPM 2.0, Apress, 2015, pp. 151–161. doi:doi: 10.1007/978-1-4302-6584-9\_12.
- [47] A. Kiser, J. Hess, E. M. Bouhafa, S. Williams, The Combat Cloud: Enabling Multi-Domain Command and Control Across the Range of Military Operations, Technical Report AD1042210, AIR COMMAND AND STAFF COLLEGE, 2017. URL: <https://apps.dtic.mil/sti/tr/pdf/AD1042210.pdf>.