Check for updates

RESEARCH ARTICLE

A minimal, deployment-agnostic zero trust cybersecurity framework for federated research infrastructures

[version 1; peer review: awaiting peer review]

Evangelos Raptis[1], Konstantinos Rallis[2,3], Evangelos Tsipas[2,3], Ioannis Liliopoulos[2,3], Georgios D. Varsamis[2,3], Georgios Ch. Sirakoulis[2], Panagiotis Dimitrakis[3]

[1]AEGIS IT Research GmbH, Braunschweig, Germany
[2]Electrical and Computer Engineering, Democritus University of Thrace, Xanthi, Greece
[3]NCSR "Demokritos", Agia Paraskevi, Greece

v1 **First published:** 09 Oct 2026, **6**:408
https://doi.org/10.12688/openreseurope.24088.1
Latest published: 09 Oct 2026, **6**:408
https://doi.org/10.12688/openreseurope.24088.1

Abstract

Modern research infrastructures increasingly span heterogeneous environments, including cloud computing, high-performance computing (HPC), edge computing nodes, and most recently quantum computers (QC). This diversity challenges traditional perimeter-based security models and demands more agile, robust cybersecurity solutions. In response, we present a minimal yet comprehensive cybersecurity architecture built around Zero Trust principles and darknet-inspired principles. Our solution adopts a deployment-agnostic approach, abstracting away environmental differences through encrypted overlay networks and identity-driven authentication and authorization mechanisms. The framework is instantiated as an open-source reference implementation, validated through a pilot deployment using Nebula, Keycloak, Open Policy Agent, and Traefik as its core components. Drawing from practical experience, we demonstrate the logical design, methodological approach, and architectural patterns needed for achieving secure, scalable, and user-friendly collaboration across organizational boundaries. A mixed evaluation that combines qualitative analysis, policy-correctness tests and targeted performance benchmarks, highlights how the proposed framework significantly reduces complexity, enhances security, and maintains flexibility in federated research scenarios. Finally, we outline a roadmap for extending the framework toward quantum resilience through the incremental adoption of post-quantum cryptographic mechanisms.

Open Peer Review

Approval Status *AWAITING PEER REVIEW*

Any reports and responses or comments on the article can be found at the end of the article.

Keywords
Zero Trust Architecture, federated research infrastructures, encrypted overlay networks, identity federation, policy-as-code, post-quantum cryptography

This article is included in the Horizon Europe gateway.

Corresponding authors: Evangelos Raptis (e.raptis@aegisresearch.eu), Konstantinos Rallis (krallis@ee.duth.gr)

Author roles: Raptis E: Conceptualization, Formal Analysis, Investigation, Methodology, Software, Writing – Original Draft Preparation, Writing – Review & Editing; **Rallis K**: Supervision, Validation, Visualization, Writing – Original Draft Preparation, Writing – Review & Editing; **Tsipas E**: Validation; **Liliopoulos I**: Validation; **Varsamis GD**: Validation; **Sirakoulis GC**: Supervision, Validation; **Dimitrakis P**: Funding Acquisition, Supervision, Validation, Visualization, Writing – Original Draft Preparation, Writing – Review & Editing

Competing interests: No competing interests were disclosed.

Grant information: This project has received funding from the European Union's Horizon Europe research and innovation programme under grant agreement No [101135927] (NOUS: A catalyst for EuropeaN ClOUd Services in the era of data spaces, high-performance and edge computing).
The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Copyright: © 2026 Raptis E *et al*. This is an open access article distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

How to cite this article: Raptis E, Rallis K, Tsipas E *et al.* **A minimal, deployment-agnostic zero trust cybersecurity framework for federated research infrastructures [version 1; peer review: awaiting peer review]** Open Research Europe 2026, **6**:408 https://doi.org/10.12688/openreseurope.24088.1

First published: 09 Oct 2026, **6**:408 https://doi.org/10.12688/openreseurope.24088.1

1. Introduction

The field of cybersecurity in research infrastructures has undergone significant transformation due to the rapid adoption of cloud computing, the wide spread of Internet of Things (IoT) devices, and the increasing sophistication of cyber threats. Modern research environments are now highly interconnected, distributed, and dynamic, making them more vulnerable to a wide range of attacks. In addition, a great challenge is the protection of long-lived workflows, distributed infrastructure components and critical systems.[1]

The traditional approach to cybersecurity, which relies on perimeter-based models, is increasingly unable to address these new challenges. Those security models are built on the assumption that threats originate outside a clearly defined network boundary and that everything inside the perimeter can be trusted. However, the widespread use of cloud services, remote access, and mobile devices has blurred the boundaries of organizational networks, making it difficult to establish a fixed perimeter. As a result, attackers can exploit vulnerabilities both outside and inside the network, and insider threats have become a significant concern. Additionally, IoT devices and distributed computing architectures introduce new attack surfaces that perimeter defenses cannot adequately protect.[2,3,4,5]

The dynamic and evolving nature of cyber threats exposes further the limitations of perimeter-based models. Advanced persistent threats, zero-day vulnerabilities, and AI-driven attacks can bypass traditional defenses, targeting internal systems and user credentials.[6] Static rule-based security measures struggle to keep up with these sophisticated attack vectors, leading to increased risk for research infrastructures.[7,8,9,10] Furthermore, the complexity and scale of modern digital ecosystems, their dynamic and extensible nature with distributed and diverse resources, require security solutions that can adapt in real time and provide granular, context-aware protection, without disrupting their operation.

The progress of quantum computing technology at both the algorithmic and hardware levels complicates cybersecurity even more. A true implementation of Shor's quantum algorithms on a quantum computer with hundreds of thousands or millions of error-free qubits may be realized within the next decade. This makes quantum-accelerated cryptanalysis an actual threat (often referred to as Q-day).[11] While the exact timeline for this event remains debated, proactive architectural preparation is considered necessary. The solution on facing this threat is called post-quantum cryptography (PQC) - according to the National Institute of Standards and Technology (NIST) - which is a set of cryptographic algorithms designed to remain secure against both classical and quantum computer attacks.[12,13] Post quantum security support must provide secure boot, secure device authentication, and secure channels.[14,15]

To address these challenges, research and practice are shifting toward Zero Trust Architecture (ZTA) frameworks. Zero Trust operates on the principle of "never trust, always verify," enforcing continuous authentication, dynamic trust evaluation, and least-privilege access regardless of network location. This paper presents a minimal yet comprehensive Zero Trust cybersecurity architecture tailored for federated research collaborations. The framework builds on encrypted, darknet-inspired overlay networks for secure connectivity, identity-driven trust establishment, and policy-as-code enforcement to ensure consistency, scalability, and robustness across heterogeneous environments. Drawing from practical implementation experience, we demonstrate how the proposed logical model can be instantiated using open-source components and how it significantly reduces integration complexity while improving usability, scalability, and its security posture. The possible integration of this framework with PQC techniques is also investigated. The rest of this paper is structured as follows: Section 2 details the conceptual underpinnings of zero trust, and the current SotA, Section 3 analyses the core design philosophy of the proposed framework and Section 4 describes the followed deployment-agnostic architectural patterns. Section 5 presents the open-source reference implementation and pilot deployment experience. Section 6 reports the framework evaluation, including policy-correctness tests and targeted performance benchmarks. Finally, Section 7 describes the roadmap for extending the framework toward quantum resilience through post-quantum cryptographic mechanisms, and Section 8 draws the final conclusions.

2. Background and related work

Zero Trust is a cybersecurity paradigm that differentiates from the common perimeter-based security practices. It rejects implicit trust, requiring continuous verification, strict access controls, and operating under the assumption that breaches are inevitable. Its core principles as presented below, form a robust framework for defending against evolving threats.

2.1 Core principles of zero trust

1. **Verify explicitly:** Zero Trust mandates that every user, device, and application must be authenticated and authorized before accessing resources, regardless of their location or network position. This is achieved through rigorous identity and access management (IAM), multi-factor authentication (MFA), and continuous monitoring of contextual factors such as device health, user behaviour, and data sensitivity. Trust is never assumed; it is dynamically evaluated and granted only when all security requirements are met.[4,16–19]

2. **Assume breach:** Zero Trust operates on the premise that threats can originate from both inside and outside the network. Organizations must presume that breaches are possible or ongoing, leading to the implementation of real-time monitoring, threat detection, and rapid incident response. This mindset drives the adoption of micro-segmentation, continuous authentication, and comprehensive logging to contain and minimize the impact of potential breaches.[4,19,20,21,22,23]

3. **Least privilege:** Access is restricted to the minimum permissions necessary for users, devices, or applications to perform their tasks. Fine-grained, dynamic access controls ensure that even if credentials are compromised, attackers cannot move laterally or access critical assets. This principle reduces the attack surface and limits the scope of potential damage from insider threats or external attack.[4,18,19,20,22,23]

2.2 Darknet and overlay networks vs. traditional network-level security

Darknets and overlay networks represent a shift from trying to secure the underlying IP fabric itself to building secure, logically separate networks on top of it. An overlay is a virtual network whose nodes are connected by tunnels over an existing IP infrastructure; it can implement its own addressing, routing, and security policies independent of the physical topology below. Modern IP overlays commonly use virtual private network (VPN) technologies such as WireGuard[24] plus a control plane that automatically distributes keys and manages mappings from overlay to underlay addresses, avoiding reliance on external public key infrastructure (PKI) or proprietary systems.[25] This enables enterprises to build encrypted meshes that provide confidentiality, mobility, and fine-grained access control for endpoints, whether on-premises or remote, without re-architecting their underlying routing or segmentation.[25,26] In cloud and distributed environments, this model aligns naturally with Zero Trust architectures, which assume that no part of the underlying network is trusted and require continuous authentication and authorization for every connection.[27,28] Security is enforced at overlay edges rather than at a perimeter firewall, decoupling protection from physical placement and traditional IP boundaries.

Darknets, in the sense of anonymity overlay networks such as Tor, I2P, and Freenet, go further by making endpoints and paths hard to discover, instead of just encrypting content. These overlays require specific software and configurations and use multi-layer encryption, onion or tunnel routing, and peer-to-peer topologies to hide the relationship between source, destination, and route.[29] Freenet's "darknet" mode, for example, restricts links to trusted peers only, so a node reveals its identity only to contacts it already trusts, and there is no single central party that everyone must trust.[30] From a security-architecture perspective, this demonstrates an alternative primitive: instead of exposing services on globally visible IP addresses and then filtering access, services can be reachable only through authenticated or trust-based overlay paths. Work on anonymous transmission based on identity mapping explicitly combines multi-layer encryption with overlay concepts to enhance privacy and integrity without modifying the underlying network structure.[31] The topology that results from these anonymity-driven design choices is structurally different from the public Internet: measurements show darknets tend to be scale-free, highly clustered small-world networks without a highly connected "rich core," which makes them significantly more resilient than the Internet to random failures, targeted attacks, and cascading failures.[32] In effect, darknet and overlay networking treat the underlay as untrusted transport and move the main security properties, including confidentiality, endpoint hiding, access control, and even resilience, into a logically separate, cryptographically protected layer that can be deployed, evolved, and reasoned about independently of traditional network-level mechanisms such as perimeter firewalls and static VPN site-to-site links.

2.3 Current state of the art

Academic and industrial efforts around Zero Trust in advanced computing environments are now substantial, whereas explicit adoption of darknet-style overlays (e.g., Tor-like hidden services) in cloud, HPC, or quantum computing contexts remains virtually absent in the available literature. Zero Trust work is driven by the erosion of perimeter-based security in highly distributed cloud and multi-cloud infrastructures, emphasizing continuous verification, strict identity and access management, micro-segmentation, and pervasive telemetry as foundational principles for securing data and services regardless of network location. Comparative reviews and surveys of Zero Trust in cloud networks converge on the view that Zero Trust, when properly implemented, can significantly reduce successful breaches and mitigate lateral movement and insider threats, though it introduces complexity, performance overhead, and substantial organizational change requirements.[23,28,33] Empirical and case-based studies report substantial reductions in security incidents after Zero Trust deployment, particularly in cloud-centric enterprise environments.[23] At the same time, critical analyses underline that large-scale Zero Trust implementations in cloud settings face unresolved challenges in scalability, integration with legacy systems, economics for small and medium-sized organizations, and compliance with diverse regulatory regimes.[23,33]

Recent research highlights how emerging technologies are reshaping Zero Trust design and deployment in cloud, edge, and next-generation networks. Multiple studies examine the integration of artificial intelligence and machine learning

into Zero Trust architectures to provide behavior-based trust scoring, anomaly detection, and predictive analytics for adaptive policy enforcement in dynamic cloud environments.[34,35,36] Proposed frameworks combine supervised and unsupervised learning with frequent pattern analysis or deep reinforcement learning to anticipate security breaches and update access rules in near real time.[34,37] In parallel, there is growing attention to quantum-related risks and opportunities. Several works argue that Zero Trust must be made "quantum-resilient" through the incorporation of post-quantum cryptography (e.g., CRYSTALS-Kyber, Dilithium) and even quantum machine learning to enhance detection and response in high-throughput cloud networks.[34,35,38] Related work on blockchain-enabled Zero Trust augments cloud and IoT environments with decentralized identity, tamper-evident logs, and zero-knowledge proofs for privacy-preserving authentication, explicitly targeting quantum-resilient keys and multi-factor, least-privilege access control at scale.[34,37] Across these streams, the cloud remains the primary deployment substrate, with industrial case studies referencing large vendors and government programs, including Google BeyondCorp,[39] major hyperscalers' Zero Trust offerings, and U.S. federal Zero Trust mandates, as reference implementations that demonstrate both feasibility and operational trade-offs.[20,28,33,37,40]

Within high-performance computing (HPC) and quantum computing (QC) specifically, the literature is thinner but emerging. One dedicated study assesses the compatibility of the U.S. Cybersecurity and Infrastructure Security Agency's Zero Trust Maturity Model with High Performance Computing Infrastructure, aligning it with NIST SP 800–223's zone-based reference architecture and concluding that only a relatively conservative "Traditional" Zero Trust maturity level is currently practical, given the stringent performance and operational constraints of HPC workloads.[33] This work underscores the difficulty of applying fully dynamic, highly granular Zero Trust controls to tightly coupled, latency-sensitive scientific and AI computations, where excessive segmentation and continuous verification can undermine throughput and job scheduling efficiency. In quantum networking and next-generation communication systems, conceptual frameworks propose quantum-enhanced Zero Trust, combining policy enforcement points, identity and access management to build more robust trust anchors for critical network infrastructure.[35,41] Parallel analyses of quantum computing adoption in industry highlight that, beyond cryptographic disruption, barriers such as lack of secure communication standards, limited technical expertise, and immature tooling slow integration of quantum technologies into production security architectures, including Zero Trust designs.[42,43]

In summary, current literature firmly establishes Zero Trust as the reference model for cloud and emerging advanced computing architectures. However, the explicit application of darknet-style hidden-service paradigms remains under-explored. Instead, current architectures attempt to achieve non-enumerability and minimized attack surfaces through other localized and possibly fragmented implementations. This absence of a unified, software-defined overlay fabric, creates a clear opportunity for deployment-agnostic architectures that bridge the gap between abstract identity policies and untrusted network transport.

3. Logical design and methodology

Translating the theoretical intersection of Zero Trust identity verification and non-enumerable network overlays into an operational architecture requires a functional design model that remains independent of specific software stacks. Instead of binding our methodology to a particular infrastructure substrate or vendor implementation, we first establish the invariant security conditions that must govern the platform.

In order to derive framework requirements, we model a typical modern organization where employees work remotely, applications run in the cloud, and the network is assumed to be compromised. A modern cybersecurity framework has to make sure that, at each step where a resource is accessed, four logical functions are always happening in the background: authentication, authorization, encryption, and segmentation. Zero Trust architectures and NIST-style frameworks can be seen largely as different ways of orchestrating these same functions continuously and consistently across people, processes, and technology.[16,21,44]

In practice, authentication means the system must reliably know who or what it is dealing with at any moment. When an employee logs in to a HR portal or an application programming interface (API) calls a microservice, the framework requires strong proof of identity: multi-factor authentication, device health checks, and, when required, behavioral signals (unusual time, location, or device).[21] In Zero Trust, this is not a one-time front-door event, the identity of users, devices, and workloads is re-verified as they move between services, access more sensitive data, or their context changes.[16,21]

Once identity is known, authorization decides what that identity is allowed to do. This becomes role- or attribute-based policies such as: "Finance-analyst role may read, but not export, payroll data from this subnet and SaaS app, only from corporate-managed devices".[16,44] Frameworks insist on least privilege and dynamic decisions: access may be tightened if the user is off-network, using a new device, or requesting unusually large data volumes. Identity and Access Management

(IAM) platforms and policy engines are the primary tools here, tightly integrated with applications and network controls.[16,21]

Encryption protects data wherever those identities and permissions are not enough by themselves. That means Transport Layer Security (TLS) for all service-to-service and user-to-service traffic, VPNs or secure tunnels for admin access, and disk/database encryption for stored HR files, logs, and backups. In a Zero Trust or cloud environment, the assumption is that networks and storage can be observed, so the framework mandates strong cryptography and sound key management so that even if traffic or stored data are stolen, they remain unreadable.[16,21,45]

Finally, segmentation determines where things can move. Instead of a flat internal network, a concrete design breaks systems into zones and micro-segments: HR, finance, OT, dev/test, individual microservices, and even per-workload segments in virtual private clouds (VPCs). Firewalls, software-defined networking (SDN) policies, and identity-aware proxies then strictly control lateral movement. For example, a compromised laptop cannot directly talk to the database subnet, and a breached microservice can talk only to the specific APIs defined in policy.[16]

Putting all together, a concrete request flow looks like this: a user or service authenticates, a policy engine authorizes a specific action with least privilege, all communication is encrypted and the request is only allowed to traverse pre-defined segments relevant to that action. Telemetry from these controls then feeds detection and incident-response functions in broader frameworks such as NIST CSF. Figure 1 illustrates how these four functions compose into a complete request lifecycle.

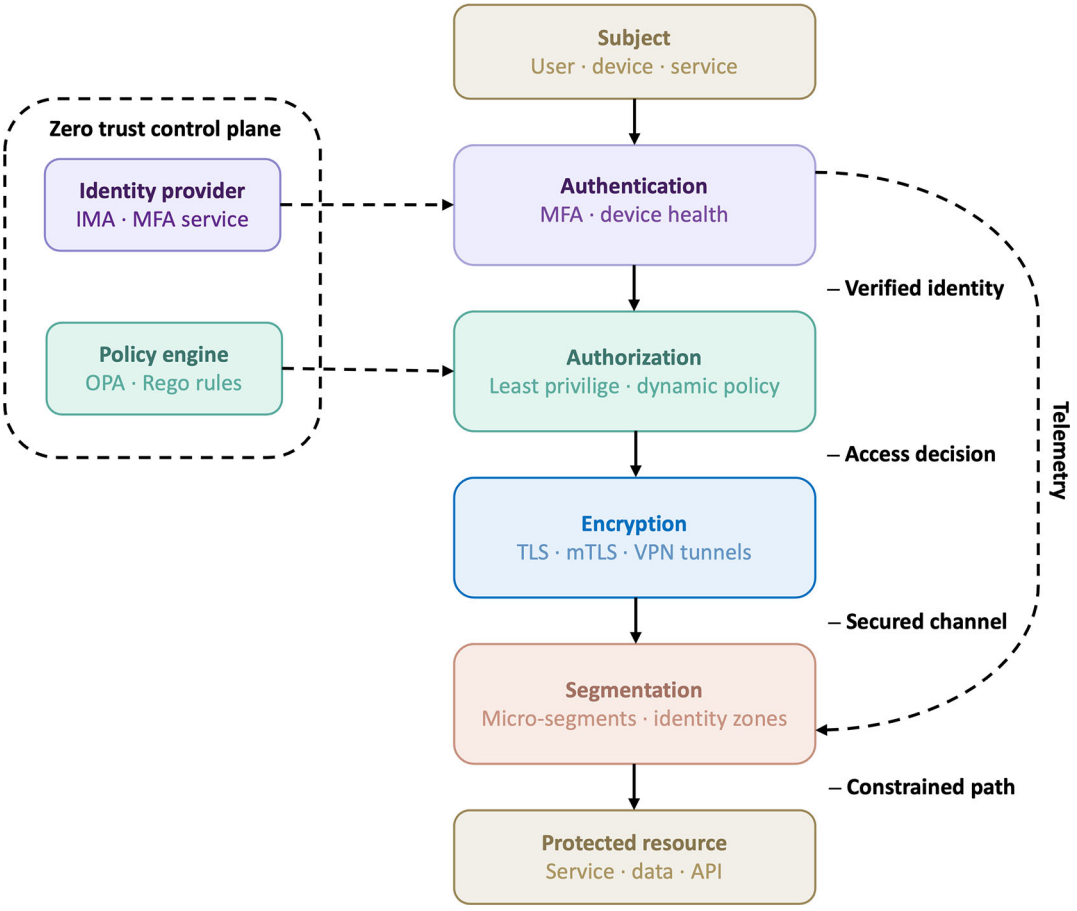

Figure 1. Zero Trust Architecture: four core logical functions of the request lifecycle. The control plane (left) provides identity verification via an Identity Provider (IAM) and policy decisions via a Policy Engine. A request from a subject traverses authentication, authorization, encryption, and segmentation before reaching the protected resource; transition labels denote the security guarantee produced at each stage.

4. Deployment-agnostic architectural patterns

The four logical functions described in Section 3 must operate consistently regardless of the physical environment in which they are deployed. Federated research infrastructures typically span cloud platforms, high-performance computing (HPC) clusters and edge devices, each with fundamentally different operational constraints, network topologies, and administrative boundaries. A cybersecurity framework that binds its security logic to any single deployment substrate cannot scale across such heterogeneous ecosystems. This section presents the abstract design patterns that decouple the framework's security guarantees from infrastructure specifics, and demonstrates how identical security logic can be instantiated across cloud, HPC, and edge environments without modification to the core architecture.

4.1 The case for deployment agnosticism

Traditional security architectures assume a relatively homogeneous infrastructure: a corporate data centre behind a firewall, or a single cloud provider's virtual private network. In federated research, however, a single workflow may originate at a researcher's browser, traverse a cloud-hosted API gateway, trigger a computation on an HPC supercomputer, and collect telemetry from edge sensors, all within a single session. Each of these environments imposes different constraints. Cloud platforms offer elastic provisioning and container orchestration but expose services on shared, multi-tenant networks. HPC clusters provide extreme computational throughput but operate under strict scheduling policies, tightly controlled network perimeters, and limited tolerance for additional per-packet overhead. Edge devices are resource-constrained, often behind Network Address Translation (NAT), and may connect intermittently over cellular or satellite links.

A deployment-agnostic framework addresses this diversity not by building environment-specific security modules, but by abstracting the security functions into logical layers that are instantiated identically everywhere. The physical environment supplies transport and the framework supplies trust. This separation yields three principal advantages: (i) security policies are defined once and enforced uniformly, eliminating the configuration drift that arises when each site maintains its own security stack, (ii) new environments can be on-boarded by provisioning credentials and installing lightweight agents, rather than redesigning the security architecture, and (iii) the operational burden on federated partners is minimised, since each site need only expose a single outbound UDP port to participate in the secure overlay.

4.2 Network abstraction via encrypted overlay

The foundational design pattern is network abstraction: the creation of an encrypted overlay network that treats all underlying physical networks, including cloud VPCs, HPC InfiniBand fabrics, campus LANs and public cellular networks, as untrusted transport. Every participating node, regardless of its physical location, is assigned a stable virtual address on the overlay and communicates exclusively through authenticated, encrypted tunnels.

This pattern draws on darknet-inspired principles discussed in Section 2. Rather than exposing services on globally routable IP addresses and filtering access with perimeter firewalls, services are reachable only through the overlay. A node that does not possess a valid cryptographic certificate signed by the federation's certificate authority (CA) cannot discover, connect to, or even detect other nodes on the overlay. The overlay thereby implements non-enumerability and thus, the attack surface is reduced to zero for any entity outside the trust boundary.

From the perspective of applications and services, the overlay presents a flat, routable address space. A cloud microservice communicates with an HPC scheduler or an edge data collector using the same addressing scheme and the same API calls, regardless of whether the peer is across the room or across the continent. This abstraction eliminates the need for environment-specific network configurations such as site-to-site VPN tunnels, cloud-provider peering agreements, or firewall rule negotiations with HPC administrators. The overlay consolidates all of these into a single, software-defined network layer.

The overlay also natively supports host-level micro-segmentation. Each node's certificate carries embedded attributes like role, group membership and environment tier, that are evaluated by a distributed firewall running on every host. Firewall rules are expressed in terms of cryptographic identity rather than IP addresses. For example, a database node may accept connections only from nodes whose certificates identify them as application servers, regardless of which cloud region, HPC cluster, or edge site those application servers inhabit. This identity-based segmentation is portable, as the same rule applies without modification whether the application server runs in a Kubernetes pod, on an HPC login node, or on an ARM-based edge gateway.

4.3 Identity federation and distributed authentication

The second core pattern is identity federation: a centralised identity provider (IdP) that serves as the single authoritative source for user and service identities across all deployment environments. Rather than maintaining separate user

databases at each site, which is a common and error-prone practice in federated research, the framework establishes one logical identity plane that all environments trust.

The IdP issues cryptographically signed tokens (following standards such as OAuth 2.0[46] and OpenID Connect (OIDC)[47] that encode a subject's identity, roles, group memberships, and other security-relevant attributes. Any service in any environment can validate these tokens locally by verifying the signature against the IdP's public key, without requiring a network call back to the IdP for each request. This distributed validation model is essential for environments like HPC clusters, where network connectivity to external services may be restricted or where latency-sensitive workloads cannot tolerate synchronous calls to a remote authentication server.

Identity federation also addresses the cross-organisational dimension of federated research. The IdP can act as an identity broker, accepting authentication assertions from external identity providers (i.e. university lightweight directory access protocol (LDAP) directories) and translating them into the framework's internal token format. A researcher at one institution can thus authenticate using their home credentials and receive a token that is understood and enforced by services running at any other partner site, on any infrastructure. This eliminates the need for researchers to manage separate accounts per site and ensures that access revocation at the home institution propagates immediately to all federated services.

The framework achieves a two-layer trust model by combining network-level identity (host certificates on the overlay) with user-level identity (IdP-issued tokens). A sensitive operation such as submitting a job to an HPC cluster or accessing a restricted dataset, requires both that the requesting host is a legitimate, certified member of the overlay and that the request carries a valid user token with the appropriate role claims. This layered verification provides compromise resilience, as a stolen user token cannot be exploited from an untrusted host, and a compromised host cannot perform privileged operations without a valid user credential.

4.4 Policy enforcement as an abstract service

Complementing these authentication mechanisms, the architecture externalizes policy enforcement delegating authorization to a dedicated policy engine that evaluates requests against declarative, centrally authored rules. Rather than embedding access-control logic in application code where it becomes fragmented, inconsistent, and difficult to audit, the framework treats policy as a service that any component in any environment can query.

The policy engine consumes structured inputs: the requesting identity (from the IdP token), the target resource, the requested action, and contextual attributes such as time, device posture, or data sensitivity classification. It evaluates these inputs against policies written in a declarative language, returning an allow or deny decision that the requesting service enforces. This separation of decision from enforcement means that the same policy is applied identically whether the data access request originates from a cloud-hosted analytics service, an HPC batch job, or an edge preprocessing pipeline.

To operationalize this uniformity, policies are managed as code, where rules are systematically versioned, tested, and deployed through the same continuous integration pipelines used for application software. This "policy-as-code" approach ensures auditability (every policy change is tracked), consistency (all enforcement points receive the same policy version), and agility (policy updates propagate to all environments without redeploying applications). For federated ecosystems where multiple organisations must agree on access rules, policy-as-code also provides a transparent, reviewable artefact that can serve as a basis for cross-organisational security governance.

The policy engine can be deployed flexibly to match environment constraints. In cloud-native settings with container orchestration, it may run as a sidecar alongside each microservice, ensuring sub-millisecond decision latency. In HPC environments where sidecar patterns are impractical, it can run as a system daemon on login or gateway nodes. On resource-constrained edge devices, a lightweight policy module with a cached subset of rules can operate autonomously, falling back to the central engine only for complex or novel requests. In each case, the logical function, context-aware, identity-driven authorisation, remains identical and only the physical deployment topology adapts.

4.5 Consistent application across environments

To illustrate the deployment-agnostic nature of these patterns, we trace how the same security logic manifests in each target environment:

- **Cloud environment:** The cloud hosts the framework's control plane including the identity provider, the API gateway, the policy engine, and at least one overlay lighthouse for peer discovery. Services run as containers or virtual machines, each provisioned with an overlay certificate and connected to the encrypted mesh. External

user requests arrive at the API gateway, which enforces authentication via the IdP before routing traffic over the overlay to backend services. Each service queries the policy engine before processing requests. All inter-service communication traverses the overlay, ensuring encryption even within a single cloud provider's network, a critical property given that cloud-internal traffic has historically been assumed trustworthy but is increasingly targeted by lateral-movement attacks.[28]

- **HPC environment:** HPC clusters are integrated by installing the overlay agent on one or more gateway or login nodes. The agent initiates an outbound encrypted session to the cloud-hosted lighthouse, requiring no inbound firewall rules on the HPC side, a property that dramatically simplifies negotiation with HPC administrators, who are typically reluctant to open ports. Once connected, the HPC node appears as a peer on the overlay, reachable by cloud services and edge devices alike. User authentication is handled by the same IdP; the framework can federate with the HPC centre's existing LDAP or directory service, so researchers retain their institutional credentials. The overlay's micro-segmentation rules ensure that only authorised cloud services (e.g., an orchestration controller) can reach the HPC node, and only on specified ports (e.g., the job scheduler API). Internal HPC communication, message passing interface (MPI) traffic between compute nodes, for instance, remains on the high-speed local fabric, unaffected by the overlay, preserving the performance characteristics that justify HPC's existence.

- **Edge environment:** Edge devices such as sensors, IoT gateways and field servers, run the overlay agent and connect back to the lighthouse over whatever network is available (Wi-Fi, cellular, satellite). The overlay handles NAT traversal transparently, establishing direct encrypted tunnels to cloud or HPC peers without requiring static IP addresses or manual port forwarding. Edge nodes authenticate to platform services using machine credentials (e.g., client certificate grants from the IdP), and their overlay certificates restrict them to the "edge" group, limiting their communication to authorised data ingestion endpoints. The policy engine can enforce additional constraints, such as restricting an edge device to writing only to its own data partition. Because the edge node's overlay certificate and IdP credentials follow the same format and trust chain as those used in cloud and HPC, adding a new edge device requires only issuing a certificate and configuring the agent, without any additional changes to the security architecture itself.

5. Practical instantiation and experiences

The architectural patterns of Section 4 were instantiated as a working system within the NOUS federated research platform and released as open-source software on the project's public repository, as referenced in Section 9 (Data and Software Availability). This section briefly summarises the tool selection, the automation model, and the behaviour observed during the pilot deployment, highlighting faithfully the abstract logical model translated into operational code.

5.1 Open source component selection

Each of the four abstract design patterns introduced in Section 4 was realised through a mature, open-source, and standards-based component, selected for broad platform coverage, active maintenance, and independence from any single infrastructure provider:

- **Network abstraction** was instantiated with **Nebula**, a lightweight, certificate-based encrypted overlay originally released as open source by Slack.[48] Nebula was preferred over alternatives such as ZeroTier or traditional IPsec VPNs for three reasons: its flat, identity-driven addressing model maps cleanly to the host-level micro-segmentation pattern; its agent runs in user space on any Linux, Windows, or macOS host, including constrained ARM edge devices; and its lighthouse-based peer discovery traverses NAT without requiring inbound firewall rules on participating sites.

- **Identity federation** was instantiated with **Keycloak,**[49] deployed as a clustered identity provider backed by a PostgreSQL high-availability cluster. Keycloak supports the full set of standards relevant to federated research (OAuth 2.0, OpenID Connect, SAML 2.0, LDAP federation) and exposes identity brokering, letting partner organisations retain their existing authentication systems while participating in the federated platform.

- **Policy enforcement** was instantiated with **Open Policy Agent (OPA),**[50] configured with a deny-by-default Rego policy (`default allow := false`) and a centrally managed policy bundle. Authorisation rules are expressed in declarative YAML in the inventory and rendered into the OPA bundle at deployment time, so operators never edit Rego directly for routine access-control changes.

- **Secure ingress** was instantiated with **Traefik,**[51] chained with **oauth2-proxy** for OpenID Connect (OIDC) forward-authentication against Keycloak and with a small local authorisation bridge that normalises the request context before querying OPA's decision API.

Figure 2 shows how these aforementioned four components are composed into the complete NOUS platform architecture, illustrating the three-layer structure of control plane, application layer, and encrypted overlay across heterogeneous deployment environments.

All services except Nebula itself run as systemd-managed Podman containers, pinned to explicit image tags. Nebula runs directly on the host via systemd so that overlay connectivity is available before any container runtime starts. None of these

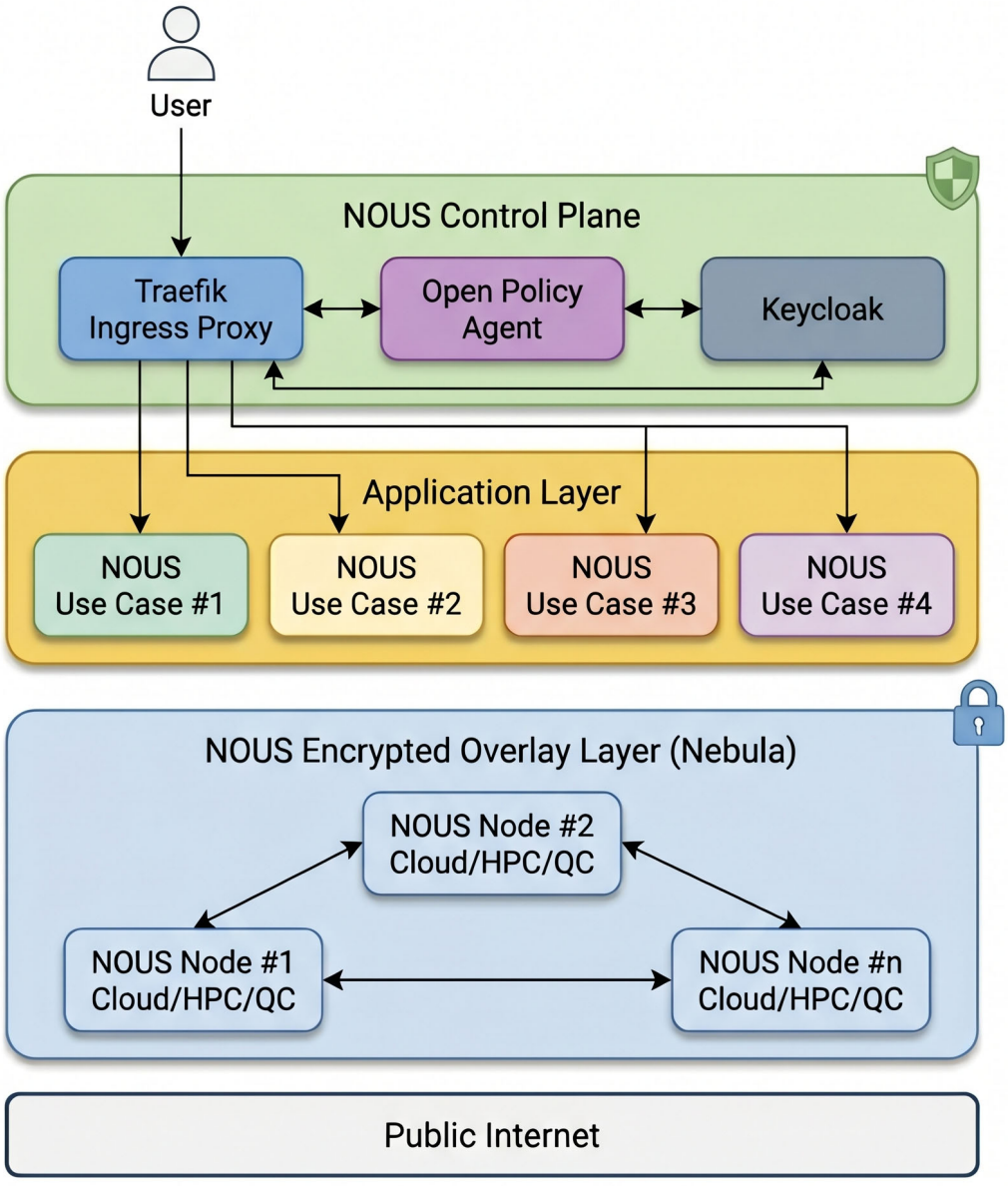

Figure 2. Complete toolchain of the proposed cybersecurity framework. The control plane (green) comprises Traefik as the ingress enforcement point, Keycloak as the identity provider, and Open Policy Agent for policy decisions. Protected use-case services (yellow) are exposed exclusively through the control plane. All nodes, spanning cloud, HPC, and edge computing environments, communicate over the Nebula encrypted overlay layer (blue).

choices is load-bearing at the architectural level as each component implements a well-defined abstract role, and any of them could be substituted with an equivalent tool (e.g., Tailscale in place of Nebula, Authentik in place of Keycloak) without altering the framework's logic. This substitutability was a deliberate design constraint, intended to protect the framework from long-term dependency on any single project's roadmap.

5.2 Automation and unified configuration

Deployment-agnosticism at the architectural level must be matched by deployment-agnosticism at the operational level. Manually installing and configuring four distributed subsystems including a Nebula certificate authority, at least two lighthouses, a clustered identity provider over an high availability (HA) database, a policy engine with bundle-driven distribution, and a gateway with OIDC-plus-OPA middleware, across cloud, HPC, and edge sites would be error-prone, inconsistent, and fundamentally non-reproducible. Without automated orchestration, manual interventions inevitably introduce configuration drift, re-establishing the environment-specific customizations that the architecture aims to eliminate.

To address these operational challenges, the framework is packaged as an Ansible-based automation suite. Ansible was selected over container-orchestration-specific tooling, such as Helm charts, because it operates uniformly across bare-metal HPC login nodes, cloud virtual machines, and resource-constrained edge appliances using only standard SSH as its transport mechanism.[52]

Operational consistency is enforced by consolidating the platform state into a single inventory file that serves as a unified operator configuration. This file comprehensively maps host topologies, SSH parameters, Nebula IP plan and micro-segmentation rules, Keycloak realms with clients/roles/groups, OPA authorisation rules, and Traefik routes. Consequently, onboarding or adjusting a protected service requires modifications across four interdependent parameters within this single file: the host's overlay group and allowed inbound ports, the Keycloak client or service account, the Traefik route, and the matching OPA policy rule. Combining networking, identity, and access policy within a single source of truth mitigates isolation errors where a backend component initializes successfully but remains unreachable due to uncoordinated downstream proxy rules.

The deployment sequence utilizes a two-stage orchestration model featuring a strict local validation layer. A first playbook brings up only the Nebula overlay: CA, lighthouses, worker certificates, distributed firewall rules, and the systemd units. Operators then deploy their application workloads on top of the overlay. A second playbook first runs a wiring preflight on the control node, which checks, without touching any hosts, that every component host is also a Nebula participant, that the Traefik OIDC issuer URL points to the configured Keycloak realm, that the Traefik OIDC client exists in that realm with a matching secret, that every machine client referenced by OPA rules is a confidential Keycloak client with service accounts enabled, that Traefik targets the internal OPA endpoint on the overlay, and that the Keycloak login route itself is exempt from the OIDC/OPA middleware chain (to prevent the otherwise-circular situation of requiring a valid token in order to obtain one). Only if the preflight passes does the final stage deploy Keycloak, OPA, and Traefik, and it then runs live integration checks that Keycloak can reach PostgreSQL and Traefik can reach both Keycloak and OPA, all over the Nebula overlay. This preflight/verify pattern converts a class of subtle, late-binding failures like auth loops, 401 redirects, silent OPA denies and clustered database partitions, into fail-fast errors at inventory validation time, before any host state is changed.

5.3 Pilot deployment and observations

The framework was deployed in a pilot configuration consisting of a Nebula overlay spanning a control plane (two PostgreSQL nodes, two Keycloak nodes clustered over the overlay, a single OPA policy decision point, and a Traefik gateway with oauth2-proxy and the authorisation bridge), together with worker nodes that hosted a Flower-based federated learning pilot application. Traffic between Traefik and backend services, between Keycloak nodes, and between Keycloak and PostgreSQL all traversed the encrypted overlay; the underlying network was treated purely as transport. Several observations emerged from this deployment and informed the qualitative evaluation presented in Section 6.

From an operational perspective, once the inventory was populated, bringing up the full stack including certificate authority, overlay convergence, HA identity cluster, policy engine with initial bundle, and authenticated gateway, required only the two `ansible-playbook` invocations described above, and completed in the order of minutes. Re-running the playbooks was idempotent and safe, which made the automation usable not only for initial deployment but also for day-two operations such as adding hosts, rotating secrets, or publishing new routes.

This deployment and operation did not disrupt user workflows, as researchers using the platform did not perceive the presence of the cybersecurity component at all. They continued to access the pilot application through the same URLs they had used previously. The only visible change was a single redirection to a Keycloak login page on first access, after which single sign-on propagated to subsequent services. The mutual Nebula handshake on every service-to-service call, the host-level overlay firewall evaluating nebula-group membership on each connection attempt, and the policy decisions evaluated by OPA on each protected request all operated entirely below the user-visible surface. For HPC-side and site administrators, integration required no changes to the cluster's internal networking or user management: only an outbound UDP connection to the lighthouse, the installation of a lightweight agent, and the addition of the node to the inventory. Onboarding additional nodes followed the same procedure.

Adding to that, the framework's policy-as-code paradigm demonstrated strong practical utility. Authorisation rules are declared in the inventory in a compact form, for instance, a rule allowing only clients holding the `data_requester` role and the `orchestration-service` machine identity to `GET/POST` against the HPC results backend, and are rendered into an OPA bundle at deployment time. Changing an access rule requires no change to any application, no service restart, and no Keycloak reconfiguration; the policy bundle is pushed to OPA and the next request reflects the new decision. This separation of identity (Keycloak), host trust (Nebula), and authorisation decision (OPA) was visible end-to-end: a request authenticated by Keycloak still had to originate from an overlay host in the correct Nebula group and still had to satisfy the Rego rules in OPA before the backend saw it.

5.4 Reflection: Abstract logic as working system

Pilot operation exposed only minor, anticipated concerns including certificate lifecycle management, the ordering constraint that Nebula identities must be issued before a node can join the overlay, and the need to keep policy bundles synchronised across distributed OPA instances should multiple be deployed. Each one of those was addressed by automation (scheduled certificate rotation, bundle distribution via OPA's native pull mechanism) and not by architectural change of any kind. The operational difficulties that were encountered, for instance, the risk of protecting the Keycloak login route with the same authentication middleware it serves, or the coupling between a host's `nebula_ip` and its issued certificate, represent expected constraints when rigorously applying Zero Trust principles. These edge cases are now explicitly encoded into the preflight suite as mandatory validation checks to eliminate deployment errors.

More significantly, the pilot demonstrated that the four-function decomposition introduced in Section 3 mapped cleanly onto operational components without leaving logical gaps or introducing redundant enforcement. Keycloak owned authentication, OPA owned authorisation, Nebula owned encryption and segmentation, and Traefik provided the edge enforcement point that composed the first three for external traffic. Each function had exactly one primary component responsible for it, and the interfaces between components were those dictated by the logical model. The absence of ad-hoc glue layers or ambiguous component ownership indicates that the decoupled design remains structurally resilient when moving from abstract logic to the actual implementation at an operational environment.

6. Framework evaluation

Since the proposed framework is a logical, modular deployment-agnostic architecture rather than a fixed vendor product, a single end-to-end benchmark against one baseline would not capture its main contribution. We therefore adopt a mixed evaluation approach, assessing the framework in terms of security, usability, scalability, and operational complexity, based on its operation and on the qualitative observations reported in the pilot tests as described in Section 5. To provide concrete empirical validation for these dimensions, we complement this assessment with targeted, implementation-derived quantitative evidence. Specifically, we execute automated policy-correctness tests to validate Zero Trust authorization integrity, collect footprint metrics to verify viability across resource-constrained edge and performance-critical HPC nodes, and conduct isolated component benchmarks to characterize the performance scaling behavior and overhead of the underlying open-source stack.

6.1 Security

The security evaluation focuses on whether the reference implementation preserves the Zero Trust properties defined in the logical design while reducing the exposure typical of perimeter-based and federated VPN approaches.

This reduction in exposure is achieved primarily by placing protected services behind an authenticated Nebula overlay, allowing the framework to avoid exposing internal service endpoints directly on routable addresses. Only hosts holding valid certificates issued by the federation CA can participate in the overlay and attempt communication with other members. This reduces external service enumerability compared with conventional firewall-filtered deployments, where exposed endpoints remain discoverable even if access is later denied.[53] The existing framework practically narrows exposure to the overlay trust fabric, certificate lifecycle, and explicitly published ingress points.

Beyond minimizing the network-level attack surface, the framework structurally mitigates compromise propagation by enforcing multi-layered trust evaluation. By combining host-level cryptographic validity (Nebula) with user-level authorization tokens (Keycloak), the architecture ensures that the compromise of a single trust vector does not result in an immediate security breach. As a result, an attacker possessing leaked user credentials cannot exploit them from outside the encrypted overlay mesh, nor can a compromised host execute application transactions without an active, explicitly authorized user session.[16,21]

Within this framework, network-level segmentation is expressed through Nebula group-based firewall rules, while application-level authorization is enforced by OPA using deny-by-default policies. A request is permitted only when both the network-level and application-level conditions are satisfied. This reduces the risk of lateral movement and prevents newly introduced services from becoming accessible without explicit policy definition.[4,16]

6.2 Usability

The usability assessment considers whether the framework can enforce Zero Trust controls without imposing excessive friction on users and infrastructure administrators.

To address user-facing friction, Keycloak provides a common identity layer for the federation and supports integration with external LDAP, SAML, and OIDC providers where available. This allows institutional identities to be mapped into a common token format used consistently by protected services. In the reference implementation, role classes such as `admin`, `analyst`, `guest`, and `service` are mapped to OPA authorization rules, allowing access decisions to be made without requiring each service to implement its own identity logic.

From an application perspective, the overlay and gateway components are largely transparent. Services communicate using standard protocols, while authentication and authorization checks are handled at the ingress and policy layers. This reduces the need for application-specific security modifications and allows existing services to be placed behind the framework with limited adaptation.

Administrative usability is achieved by converting complex, cross-institutional perimeter management into localized, automated deployment tasks. Traditional federated trust setups require extensive bilateral firewall negotiations and IP white-listing among institutional administrators.[21] In contrast, the pilot configuration demonstrates that onboarding a new node requires just three simple localized steps. More specifically it requires issuing an overlay certificate, assigning the host to its functional identity groups, and deploying the lightweight overlay agent. Because the network requirement is strictly limited to an automated, outbound-only UDP connection to a lighthouse, this approach entirely eliminates external network perimeter exposure and minimizes ongoing administrative overhead during federation expansion.

6.3 Scalability

The scalability assessment evaluates the framework's capacity to absorb an increasing volume of nodes, services, and multi-institutional users without generating management bottlenecks or performance degradation. Policy scalability is achieved by avoiding the pairwise configuration scaling curve typical of traditional cross-site VPN arrangements. As validated by the policy-as-code workflow in Section 5, adding a new service or organization requires only a localized, linear modification to a centralized inventory configuration, as optimized policy bundles are compiled and distributed out to OPA nodes automatically.[21]

Scalability is also related to the expansion of the existing system with additional nodes. For this, at the network layer, Nebula provides a peer-to-peer overlay model in which new nodes join by receiving certificates and overlay addresses. The lighthouse supports peer discovery but does not act as a central data-plane forwarding bottleneck. In addition, OIDC token validation can be performed locally using Keycloak's public keys, reducing dependence on synchronous identity-provider calls for every request.

6.4 Complexity

The complexity assessment considers whether the proposed framework remains operationally manageable despite enforcing multiple Zero Trust controls.

To achieve the required operational control, the implementation relies on a small set of open-source components: Nebula for encrypted overlay networking and host-level segmentation, Keycloak for identity management, OPA for policy decisions, and Traefik for secure ingress. This limited component set reduces architectural complexity and avoids dependence on proprietary appliances or provider-specific security services. This is also reinforced by the use of a single configuration file, as described above in Section 5.

Operationally, deployment is separated into an overlay-establishment phase and a final service-enablement phase. This allows connectivity and certificate distribution to be validated before identity, authorization, and gateway services are activated. Preflight checks further reduce operational risk by detecting wiring errors before changes are applied to live hosts.

Table 1 consolidates the qualitative comparison across the four dimensions above against two representative common alternatives: a traditional perimeter-based VPN[4,20,21,28] and a VPN mesh.[54] Across all four dimensions, the proposed framework improves on both baselines: it reduces attack surface through non-enumerability and deny-by-default enforcement, simplifies credential management through federation-wide single sign-on, scales node addition to a near-constant operational cost, and replaces bilateral negotiation with a four-component open-source stack governed by a single inventory.

6.5 Policy correctness and framework footprint

To complement the qualitative assessment, two concrete measurements were taken directly from the reference implementation.

First, the OPA authorization policy (`authz.rego`) was exercised with a purpose-built test suite using OPA v1.13.2, matching the version deployed in the pilot. The suite covers 16 distinct request scenarios spanning all security-relevant dimensions of the policy: deny-by-default enforcement, unauthenticated request rejection, role-based allow and deny, service-scoped and path-prefix-scoped access control, HTTP method filtering, machine-to-machine `client_id` enforcement, the Traefik `X-Forwarded-*` header fallback path, and the OIDC `azp` claim fallback for client identity. All 16 tests pass, and OPA's coverage report confirms 100% line coverage of `authz.rego`, meaning no reachable branch of the policy is left unverified.

Apart from validating the effectiveness of the policies, the framework was also examined regarding its footprint. Table 2 summarises selected non-arbitrary metrics derived from the reference implementation. The policy expresses 33 conditional rule heads across 7 independent matching dimensions, namely authentication state, service identity, HTTP method, URL path prefix, user role, group membership, and OAuth2 client identity. The Nebula overlay requires exactly one outbound UDP port (4242) regardless of the number of enrolled nodes. This is a hard constraint of the Nebula protocol,

Table 1. Qualitative comparison of security approaches across security, usability, scalability, and complexity dimensions.

Dimension	Perimeter VPN	Mesh VPN	Proposed Framework
Security	Perimeter trust; flat internal net	Bilateral tunnels; per-site policy	Non-enumerable Nebula overlay; OPA deny-by-default; Nebula + Keycloak layered trust
Usability	Single credential internally; multi-credential cross-site	Multiple credentials; per-site onboarding	Single Keycloak credential federation-wide; Traefik gateway transparent to apps
Scalability	Good within org; poor cross-org	Pairwise scaling curve	Linear node addition; OPA bundle sync; Keycloak HA
Complexity	Low internally; high at boundaries	High; bilateral negotiation per pair	4-component open-source stack; single inventory; staged deployment

Table 2. Architectural footprint metrics derived from the reference implementation.

Metric	Value
Conditional rule heads in policy	33
Matching dimensions enforced per request	7
OPA test cases	16
Policy line coverage	100%
External network ports required per node	1 (UDP 4242)

not a configuration choice, and directly supports the single-port onboarding claim made in Section 4. Together, these figures substantiate the "minimal" characterization attributed to the proposed framework.

6.6 Experimental benchmarks

Additionally, three targeted benchmarks were conducted on a local deployment using the same container images specified in the reference implementation. All experiments ran on a single host using Docker, the results quantify per-component overhead rather than end-to-end production throughput, and should be interpreted accordingly.

To evaluate internal component overhead in isolation, OPA was deployed with the production bundle and exercised via its HTTP API, which mirrors the ForwardAuth call Traefik issues on every protected request. Five hundred (500) requests were issued at 20 concurrent workers for two scenarios: an allow decision (authenticated analyst accessing `/datasets/`) and a deny decision (unauthenticated guest on the same path). Mean and tail (p99) decision latency were statistically indistinguishable between the two scenarios, with no divergence between the allow and deny paths. This confirms that the deny-by-default policy imposes no measurable additional cost relative to an explicit allow: the evaluation engine exhausts all rules at roughly equal speed regardless of outcome. In a production deployment, this overhead represents the marginal cost Traefik adds to each request via the ForwardAuth call to the OPA bridge.

Moving from isolated component evaluations to complete request lifecycles, a three-scenario latency comparison was conducted to quantify the overhead introduced by the complete NOUS security stack that consists of Traefik, oauth2-proxy, and OPA acting in sequence. All scenarios use the same Python `urllib` HTTP client and a controlled request rate of 5 req/s (well below the gateway rate limit of 100 req/s) to obtain uncontested measurements. Scenario A issues requests directly to the backend with no authentication or authorization (baseline). Scenario B issues requests through the full NOUS stack with a valid analyst token (authorised access). Scenario C issues requests with a guest token that OPA denies and no backend call is made. Results are summarised in Table 3.

The full security stack adds a mean overhead of 6.4 ms for authorised (ALLOW) requests and 3.6 ms for rejected (DENY) requests over the unprotected baseline (Table 3, rows B-A and C-A). This overhead jointly encompasses Traefik routing, oauth2-proxy JSON web token (JWT) validation, the authorisation-bridge ForwardAuth call, and OPA policy evaluation.

The deny path (Scenario C) completes 2.8 ms faster on average than the allow path (Scenario B). This is attributed to the fact that OPA evaluates the deny decision and returns HTTP 403 before the request is forwarded to the backend, so no backend round-trip is incurred. This demonstrates an operationally beneficial property of deny-by-default enforcement, where unauthorised requests are rejected early in the middleware chain, consuming fewer upstream resources than legitimate, fully-served requests. At the 95th percentile the gap widens further (+7.5 ms vs. +2.2 ms), indicating that the deny path remains consistently fast even under transient load spikes.

Preliminary trials at higher request rates (20 concurrent workers) confirmed that the gateway rate limiter configured at 100 req/s average, burst 200, in `nous-platform.yml`, activates under sustained concurrent load and returns HTTP 429 responses, as expected. The controlled 5 req/s rate was adopted specifically to obtain latency measurements free of rate-limiter interference.

In parallel with routing and proxy performance, the authentication plane was evaluated by exercising Keycloak with 300 `client_credentials` grant requests at 10 concurrent workers, simulating machine-to-machine service account authentication, the pattern used by the `orchestration-service` client in the reference deployment. Mean token issuance latency was 18.7 ms with a p99 of 81 ms, yielding a sustained throughput of approximately 536 tokens/s. Since

Table 3. End-to-end request latency and security stack overhead comparison (60 requests at 5 req/s) for baseline, authorized access (ALLOW), and short-circuited rejection (DENY) scenarios.

Scenario	Mean (ms)	p95 (ms)
A Baseline - direct backend, no auth	9.8	27.5
B Protected ALLOW - full NOUS stack	16.2	35.0
C Protected DENY - OPA short-circuits	13.4	29.7
Stack overhead ALLOW (B − A)	+6.4	+7.5
Stack overhead DENY (C − A)	+3.6	+2.2

tokens carry a configurable validity window (3600 s for access tokens in the reference configuration), clients reuse issued tokens across many requests. Thus, the measured throughput represents a worst-case ceiling applicable only to authentication request bursts or large-scale simultaneous cold-starts, not steady-state operation.

Finally, to validate the non-enumerability property claimed in Section 4, two OPA instances were deployed: one with its HTTP port exposed to the host (simulating a conventionally deployed service), and one on an isolated internal Docker network with no host-reachable ports (simulating a node enrolled only on the Nebula overlay). An `nmap` TCP scan of port 8181 on the exposed instance returned `8181/tcp open` within 1.4 s. The same scan against the isolated instance reported "Host seems down" with zero open ports discovered. To confirm the service was nonetheless fully operational, a curl request issued from a container running inside the isolated network received `HTTP 200` from OPA's health endpoint. This result demonstrates the core security that has been mentioned before, that a node without a valid overlay credential cannot discover or reach services that are bound exclusively to the overlay, regardless of the underlying IP infrastructure.

7. Roadmap to a quantum-resilient architecture

As discussed in Section 2, federated research infrastructures face a concrete long-term cryptographic threat. Encrypted traffic or stored data collected today may face retroactive decryption under a "Harvest Now, Decrypt Later" (HNDL) attack model once cryptographically relevant quantum computers become available.[55] Integrating Post-Quantum Cryptography (PQC) into the proposed framework is therefore a natural evolution toward long-term resilience. PQC does not replace the Zero Trust model, rather it reinforces the cryptographic foundations on which identity proof, secure communication, credential protection, and data integrity depend. In the proposed modular and deployment-agnostic framework, post-quantum mechanisms can be introduced incrementally into the components that rely on public-key cryptography, without changing the overall security model.

Building on the role of PQC in preserving the cryptographic foundations of Zero Trust mechanisms,[56] we identify four integration points where PQC can be introduced into the proposed deployment-agnostic Zero Trust stack with limited architectural disruption. These points correspond to the framework components that depend most directly on public-key cryptography, including secure channel establishment, identity and certificate management, software and workflow integrity, and key-source abstraction.

The first primary integration point involves quantum-resistant session establishment for tunnels and mutual transport layer security (mTLS). The framework in its current form, already relies on authenticated, encrypted channels (e.g., overlay networks, service meshes, or mTLS). A PQC upgrade should primarily target key establishment during handshakes, since symmetric cryptography remains largely quantum-robust with appropriate key sizes. NIST standardized module-lattice-based key encapsulation mechanism (ML-KEM) (FIPS 203) as a general-purpose key-encapsulation mechanism for establishing shared secrets.[12] Likewise, ongoing work defines how ML-KEM can be negotiated in TLS 1.3, enabling post-quantum key establishment for widely deployed protocols.[57] During the transition period, hybrid key exchange, combining conventional asymmetric primitives with PQC mechanisms, can provide interoperability while reducing dependence on a single cryptographic assumption.[58]

Apart from securing the transport layer, post-quantum identities, credentials, and certificate chains constitute another PQC integration point. Because this framework is identity-driven, its security depends on robust digital signatures for device identities, workload identities, and federation trust anchors. NIST standardized module-lattice-based digital signature algorithm (ML-DSA) (FIPS 204)[59] and stateless hash-based digital signature algorithm (SLH-DSA) (FIPS 205)[60] for digital signatures.[59,60] A forward-compatible direction is to support PQC-capable PKI or credential formats such as X.509 profiles or equivalent identity objects, so that federation roots, intermediate CAs, and workload certificates can migrate to PQ signatures. This can be rolled out incrementally by introducing PQC at the federation trust-anchor and high-value service tiers first, while maintaining classical compatibility through dual-signature or parallel trust paths where required.

Beyond dynamic network sessions and identities, PQC signatures must be extended to guarantee the long-term integrity and provenance of software, containers, and workflows. Federated research infrastructures increasingly rely on container images, workflow descriptions, and distributed job submissions. PQC signatures should therefore be considered not only for network sessions but also for software and workflow integrity: signing of artifacts, container images, infrastructure-as-code, and policy bundles. This directly supports Zero Trust's continuous verification stance and reduces supply-chain risk, especially where artifacts may be validated years after publication.

Finally, achieving comprehensive resilience across these layers requires an underlying key-source abstraction mechanism. In a deployment-agnostic framework, cryptographic keys may be stored and managed through different backends

depending on the environment, such as software keystores, trusted platform modules (TPMs), hardware security modules (HSMs) or cloud key management services (KMS). A future enhancement is therefore to introduce a key-management abstraction layer that exposes a common interface for classical and post-quantum cryptographic operations. This would allow the framework to adopt PQC mechanisms without becoming dependent on a specific hardware module, cloud provider, or key-management service.

Based on these integration points, the migration path should be incremental: hybrid post-quantum key establishment for exposed communication paths, post-quantum-capable certificates and credentials for federation trust anchors and high-value services, and post-quantum signing for container images, deployment descriptors, workflow definitions, and OPA policy bundles. The migration should also remain algorithm-agile. Since PQC standardization is still evolving, the framework should avoid dependence on a single mathematical family. NIST's selection of hamming quasi-cyclic (HQC) as an additional KEM candidate, providing diversity relative to ML-KEM, supports this direction.[61]

8. Conclusions

This paper presented a minimal, deployment-agnostic Zero Trust cybersecurity framework for federated research infrastructures spanning cloud, HPC and edge environments. The proposed framework is based on four core logical functions: authentication, authorization, encryption, and segmentation. By separating these functions from any specific deployment substrate, the framework provides a reusable security model that can be applied consistently across heterogeneous infrastructures and administrative domains.

The framework was instantiated as an open-source reference implementation using Nebula for encrypted overlay networking and host-level segmentation, Keycloak for identity management, OPA for policy-as-code authorization, and Traefik for secure ingress enforcement. The pilot deployment demonstrated that these components map cleanly to the proposed logical model and can be coordinated through Ansible-based automation and a unified inventory.

The evaluation showed that the proposed approach reduces service exposure, supports layered trust enforcement, simplifies node onboarding, and limits configuration drift through centralized policy and inventory management. A qualitative comparison against perimeter-based VPN and VPN mesh deployments across four primary dimensions, namely security, usability, scalability, and operational complexity, confirms the distinct architectural advantages of this approach. These qualitative insights are robustly reinforced by implementation-derived experimental evidence: the OPA authorization policy achieves 100% line coverage across 16 test cases, the complete security stack adds a mean overhead of only 6.4 ms for authorized requests and 3.6 ms for denied requests over an unprotected baseline, and Keycloak sustains approximately 536 tokens/s for machine-to-machine authentication. Taken together, this body of evidence demonstrates that a minimal, logically coherent Zero Trust framework grounded in four core security functions is not just theoretical, but also operationally viable, scalable, and sustainable within highly distributed, multi-tenant scientific ecosystems.

Finally, the paper outlined a roadmap for extending the framework toward quantum resilience. Post-quantum techniques can be introduced incrementally into key establishment, identity and certificate management, signing mechanisms, and key-management backends without changing the overall Zero Trust control model. Future work will focus on validating these extensions in larger multi-site deployments, strengthening policy governance and assessing post-quantum performance and interoperability in operational federated infrastructures.

Ethics and consent

Ethical approval and consent were not required.

Data and software availability

Software availability

- Source code publicly available at: https://gitlab.eclipse.org/eclipse-research-labs/nous-project/common-administration-services/security-services/nous-cybersecurity-component
- Archived software version available at: https://doi.org/10.5281/zenodo.21337762[62]
- License: MIT License

References

1. Ani UPD, He H, Tiwari A: **Review of cybersecurity issues in industrial critical infrastructure: manufacturing in perspective.** *J. Cyber Secur. Technol.* 2017; **1**(1): 32–74.
Publisher Full Text

2. Patil DA, Shyamala G: **A comprehensive survey on securing the social internet of things: protocols, threat mitigation, technological integrations, tools, and performance metrics.** *Sci. Rep.* 2025; **15**(1): 40190.
PubMed Abstract | Publisher Full Text | Free Full Text

3. Gambo ML, Almulhem A: **Zero Trust Architecture: A Systematic Literature Review.** *J. Netw. Syst. Manag.* Nov. 2025; **34**(1).
Publisher Full Text

4. Kang H, Liu G, Wang Q, *et al.*: **Theory and application of zero trust security: A brief survey.** *Entropy.* 2023; **25**(12): 1595.
PubMed Abstract | Publisher Full Text | Free Full Text

5. Georgian I, Zamfirel TA, Goga N, *et al.*: **A Survey of Machine Learning Approaches to IoT Security.** *Algorithms.* 2026; **19**(5): 384.
Publisher Full Text

6. Sharma A, Gupta BB, Singh AK, *et al.*: **Advanced persistent threats (apt): evolution, anatomy, attribution and countermeasures.** *J. Ambient. Intell. Humaniz. Comput.* 2023; **14**(7): 9355–9381.
Publisher Full Text

7. Schmitt M: **Securing the digital world: Protecting smart infrastructures and digital industries with artificial intelligence (AI)-enabled malware and intrusion detection.** *J. Ind. Inf. Integr.* 2023; **36**: 100520.
Publisher Full Text

8. Ebad SA, Amara M: **The Principle of Least Privilege in Microservices: A Systematic Mapping Study.** *Appl. Sci.* 2026; **16**(3): 1495.
Publisher Full Text

9. Desai T, Pal RK: **Machine Learning in Cybersecurity: A Comprehensive Review of Threat Detection, Prevention, and Response Strategies.** *2025 4th International Conference on Computational Modelling, Simulation and Optimization (ICCMSO).* IEEE; 2025; pp. 148–153.

10. Mahida A: **An Intellectual Zero Trust Security Framework Using Deep Reinforcement Learning for Predictive Threat Mitigation in AI-Based Fraud Detection Systems.** *IEEE Access.* 2026; **14**: 24602–24617.
Publisher Full Text

11. Cain M, *et al.*: **Shor's algorithm is possible with as few as 10,000 reconfigurable atomic qubits.** *arXiv preprint arXiv:2603.28627.* 2026.

12. Alagic G, *et al.*: **Module-Lattice-Based Key-Encapsulation Mechanism Standard.** 2024.

13. Truong QD, Nguyen H, Nguyen TT, *et al.*: **NIST Post-Quantum Cryptography Standards: A Comprehensive Review of Theoretical Foundations and Implementations.** *IEEE Access.* 2026; **14**: 14069–14097.
Publisher Full Text

14. D'Onghia G, Lioy A: **Towards Quantum-Resistant Trusted Computing: Architectures for Post-Quantum Integrity Verification Techniques.** *2025 IEEE Symposium on Computers and Communications (ISCC).* IEEE; 2025; pp. 1–6.

15. Kim H: **A Survey of Trust Chain in Secure Embedded Systems: From Secure Boot to TEE Applications.** *IEEE Access.* 2026; **14**: 59871–59897.
Publisher Full Text

16. Mushtaq S, Mohsin M, Mushtaq MM: **A Systematic Literature Review on the Implementation and Challenges of Zero Trust Architecture Across Domains.** *Sensors (Basel, Switzerland).* 2025; **25**.
PubMed Abstract | Publisher Full Text | Free Full Text

17. Azad MA, Abdullah S, Arshad J, *et al.*: **Verify and trust: A multidimensional survey of zero-trust security in the age of IoT.** *Internet Things.* 2024; **27**: 101227.
Publisher Full Text

18. Buck C, Olenberger C, Schweizer A, *et al.*: **Never trust, always verify: A multivocal literature review on current knowledge and research gaps of zero-trust.** *Comput. Secur.* 2021; **110**: 102436.
Publisher Full Text

19. Lund BD, Lee T-H, Wang Z, *et al.*: **Zero trust cybersecurity: procedures and considerations in context.** *Encyclopedia.* 2024; **4**(4): 1520–1533.
Publisher Full Text

20. Rose S, Borchert O, Mitchell S, *et al.*: **Zero trust architecture.** *NIST special publication.* 2020; **800**(207): 1–52.

21. Syed NF, Shah SW, Shaghaghi A, *et al.*: **Zero trust architecture (zta): A comprehensive survey.** *IEEE access.* 2022; **10**: 57143–57179.
Publisher Full Text

22. Ghasemshirazi S, Shirvani G, Alipour MA: **Zero Trust: Applications, Challenges, and Opportunities.** *ArXiv.* 2023; **abs/2309.03582**.
Publisher Full Text

23. Oladimeji G: **A Critical Analysis of Foundations, Challenges and Directions for Zero Trust Security in Cloud Environments.** *ArXiv.* 2024; **abs/2411.06139**.
Publisher Full Text

24. Donenfeld JA: **Wireguard: next generation kernel network tunnel.** *NDSS.* 2017; pp. 1–12.

25. Paillisse J, Barcia A, Lopez A, *et al.*: **A control plane for WireGuard.** *2021 International Conference on Computer Communications and Networks (ICCCN).* IEEE; 2021; pp. 1–8.

26. Gentile AF, Macri D, Greco E, *et al.*: **Overlay and Virtual Private Networks Security Performances Analysis with Open Source Infrastructure Deployment.** *Future Internet.* 2024; **16**: 283.
Publisher Full Text

27. He Y, Huang D, Chen L, *et al.*: **A Survey on Zero Trust Architecture: Challenges and Future Trends.** *Wirel. Commun. Mob. Comput.* 2022; **2022**.
Publisher Full Text

28. Sarkar S, Choudhary G, Shandilya SK, *et al.*: **Security of Zero Trust Networks in Cloud Computing: A Comparative Review.** *Sustainability.* 2022; **14**.
Publisher Full Text

29. Saleem J, Islam R, Islam MZ: **Darknet Traffic Analysis: A Systematic Literature Review.** *IEEE Access.* 2023; **12**: 42423–42452.
Publisher Full Text

30. Clarke I, Sandberg O, Toseland M, *et al.*: **Private communication through a network of trusted connections: The dark freenet.** *Network.* 2010; 1–19.

31. Wang C, Wang H: **Anonymous Network Transmission Method Based on Identity Mapping.** *2024 IEEE 4th International Conference on Information Technology, Big Data and Artificial Intelligence (ICIBA).* 2024; vol. **4**: pp. 1017–1021.
Publisher Full Text

32. Domenico M, Arenas A: **Modeling Structure and Resilience of the Dark Network.** *Phys. Rev. E.* 2016; **95**(2–1): 022313.
PubMed Abstract | Publisher Full Text

33. Macaulay T, Bhasker D: **High Performance Computing Infrastructure and Zero Trust Architecture.** *Pulse & Praxis: The Journal for Critical Infrastructure Protection, Security and Resilience.* 2024.
Publisher Full Text

34. Joshi H: **Emerging Technologies Driving Zero Trust Maturity Across Industries.** *IEEE Open Journal of the Computer Society.* 2025; **6**: 25–36.
Publisher Full Text

35. Weinberg AI, Cohen K: **Zero Trust Implementation in the Emerging Technologies Era: Survey.** *ArXiv.* 2024; **abs/2401.09575**.
Publisher Full Text

36. Ahmed S, Shihab IF, Khokhar A: **Quantum-driven zero trust architecture with dynamic anomaly detection in 7G technology: A neural network approach.** *Measurement: Digitalization.* 2025; **2**: 100005.

37. Aleisa MA: **Blockchain-enabled zero trust architecture for privacy-preserving cybersecurity in IoT environments.** *IEEE Access.* 2025; **13**: 18660–18676.
Publisher Full Text

38. Nalluri S, Malyala MM, Kandagiri H, *et al.*: **AI-Enhanced Zero Trust Architecture for Cloud Security with Quantum Resilience.** *2025 6th International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV).* 2025; pp. 1085–1092.
Publisher Full Text

39. Ward R, Beyer B: **Beyondcorp: A new approach to enterprise security.** *Log.* 2014; **39**(6): 6–11.

40. Phiayura P, Teerakanok S: **A comprehensive framework for migrating to zero trust architecture.** *Ieee Access.* 2023; **11**: 19487–19511.
Publisher Full Text

41. Lin J, Jiang Q, Zhang W, *et al.*: **Quantum-Enhanced Zero Trust Security: Evolution, Implementation, and Application.** *2024 International Conference on Quantum Communications, Networking,*

and Computing (QCNC). 2024; pp. 211–215.
Publisher Full Text

42. Awan U, Hannola L, Tandon A, *et al.*: **Quantum computing challenges in the software industry. A fuzzy AHP-based approach.** *Inf. Softw. Technol.* 2022; **147**: 106896.
Publisher Full Text

43. Bayerstadler A, *et al.*: **Industry quantum computing applications.** *EPJ Quantum Technol.* 2021; **8**.
Publisher Full Text

44. Melone M: *Designing secure systems.* CRC Press; 2021.
Publisher Full Text

45. Repetto M, Striccoli D, Piro G, *et al.*: **An Autonomous Cybersecurity Framework for Next-generation Digital Service Chains.** *J. Netw. Syst. Manag.* 2021; **29**.
Publisher Full Text

46. Hardt D: **The OAuth 2.0 authorization framework.** 2012.

47. Sakimura N, Bradley J, Jones M, *et al.*: **OpenID Connect Core 1.0 incorporating errata set 1.** *The OpenID Foundation, specification.* 2014; **335**.

48. Slack Technologies, Inc: **Nebula: An open-source, mutually authenticated overlay networking tool.** 2020.
Reference Source

49. Red Hat, Inc: **Keycloak: Open Source Identity and Access Management for Modern Applications.** 2026.
Reference Source

50. Cloud Native Computing Foundation: **Open Policy Agent (OPA): Policy-based control for cloud native environments.** 2026.
Reference Source

51. Traefik Labs: **Traefik: The Cloud Native Application Proxy.** 2026.
Reference Source

52. Red Hat, Inc: **Ansible Community Documentation.** 2024.
Reference Source

53. Abdelhay Z, Bello Y, Refaey A: **Toward zero-trust 6gc: A software defined perimeter approach with dynamic moving target defense mechanism.** *IEEE Wirel. Commun.* 2024; **31**(2): 74–80.
Publisher Full Text

54. Kjorveziroski V, Bernad C, Gilly K, *et al.*: **Full-mesh VPN performance evaluation for a secure edge-cloud continuum.** *Softw. Pract. Experience.* 2024; **54**(8): 1543–1564.

55. Kagai F, Branch P, But J, *et al.*: **Harvest-Now, Decrypt-Later: A Temporal Cybersecurity Risk in the Quantum Transition.** *Telecom.* MDPI; 2025; p. 100.

56. Aluri V: **Quantum-Resilient Access Control Protocols for Cloud-Native Infrastructures in Post-Quantum Security Contexts.** 2025.

57. Zheng J, Zhu H, Dong Y, *et al.*: **Faster Post-quantum TLS 1.3 Based on ML-KEM: Implementation and Assessment.** *European Symposium on Research in Computer Security.* Springer; 2024; pp. 123–143.
Publisher Full Text

58. Stebila D, Fluhrer S, Gueron S: *Hybrid key exchange in TLS 1.3.* IETF draft; 2020.

59. Dang T, *et al.*: *Module-lattice-based digital signature standard.* National Institute of Standards and Technology (NIST), Thinh Dang, Jacob; 2024.

60. Cooper D, *et al.*: *Stateless hash-based digital signature standard.* National Institute of Standards and Technology (NIST), David Cooper; 2024.

61. Alagic G, *et al.*: *Status report on the fourth round of the nist post-quantum cryptography standardization process.* US Department of Commerce, National Institute of Standards and Technology ...; 2025.

62. Raptis E: **NOUS Cybersecurity Component [Computer software].** *Zenodo.* 2026.
Publisher Full Text