NIST Interagency Report
NIST IR 8625

# Open Radio Access Network (O-RAN) Near-Real Time RAN Intelligent Controller (RIC) Security

*Considerations for achieving Zero Trust RIC Security*

Scott Rose
Oliver Borchert
Simeon Wuthier
Douglas Montgomery

This publication is available free of charge from:
https://doi.org/10.6028/NIST.IR.8625

NIST NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY
U.S. DEPARTMENT OF COMMERCE

**NIST Interagency Report**
**NIST IR 8625**

# Open Radio Access Network (O-RAN) Near-Real Time RAN Intelligent Controller (RIC) Security

*Considerations for achieving Zero Trust RIC Security*

Scott Rose
Oliver Borchert
Simeon Wuthier
Douglas Montgomery
*Wireless Networks Division*
*Communication Technology Lab*

This publication is available free of charge from:
https://doi.org/10.6028/NIST.IR.8625

September 2026


U.S. Department of Commerce
*Howard Lutnick, Secretary*

National Institute of Standards and Technology
*Arvind Raman, Under Secretary of Commerce for Standards and Technology and NIST Director*

**NIST Technical Series Policies**
Copyright, Use, and Licensing Statements
NIST Technical Series Publication Identifier Syntax

**Publication History**
Approved by the NIST Editorial Review Board on 2026-09-23

**How to Cite this NIST Technical Series Publication**
Rose S, Borchert O, Wuthier S, and Montgomery D (2026) Open Radio Access Network (O-RAN) Near-Real Time RAN Intelligent Controller (RIC) Security. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Internal Report (NIST IR) NIST IR 8625. https://doi.org/10.6028/NIST.IR.8625

**Author ORCID iDs**
Scott Rose: 0000-0002-3105-7427
Oliver Borchert: 0009-0006-1880-0542
Simeon Wuthier: 0000-0003-4088-7518
Douglas Montgomery: 0000-0002-5364-9474

**Contact Information**
oran-cybersecurity@nist.gov

National Institute of Standards and Technology
Attn: Wireless Networks Division, Communications Technology Laboratory
100 Bureau Drive (Mail Stop 8920) Gaithersburg, MD 20899-8920

## Abstract

Radio Access Networks (RANs) are used to connect mobile wireless devices to a mobile network core and the internet. Open RAN (O-RAN) is an effort to standardize the architecture and interfaces of RAN technologies, so that a mobile network operator can deploy and operate a network composed of (potentially) multiple vendor components. O-RAN also seeks to specify the RAN control plane as a set of services. This allows O-RAN network management functions to operate on any virtualization platform using a service-based architecture. A new element of RAN architecture, called the RAN Intelligent Controller (RIC), provides a framework to deploy new applications and services to instantiate, monitor and manage the operation of the RAN. The evolution from legacy monolithic RANs to service-based, open RANs with RICs introduces new security threats for mobile network operators. To address the increase complexity and openness of emerging O-RAN architectures, the industry is adopting a zero trust security architecture for O-RAN. This report discusses the status of zero trust architecture in O-RAN standards and additional considerations for fully realizing a zero trust architecture for O-RAN.

## Keywords

O-RAN; Open RAN; RAN Intelligent Controller (RIC); rApp; xApp; zero trust.

## Table of Contents



## List of Tables



## List of Figures

## Preface

This report discusses the security specifications for the RAN Intelligent Controller (RIC) as standardized by the O-RAN ALLIANCE. There is also discussion of considerations when taking a cloud native approach to Open RAN deployments, including the use of alternative, cloud-native security technologies to fulfill O-RAN security requirements. At the time of writing, these alternative cloud native technologies are not specified for use by the O-RAN ALLIANCE and should not be seen as NIST guidance on how to implement O-RAN ALLIANCE security specifications.

## Acknowledgments

This report is based on work that was funded in part by the Department of Homeland Security (DHS) Science & Technology Directorate.

## 1. Introduction

Modern mobile telecommunication networks are divided into three logical segments: the Radio Access Network (RAN), the end device referred to as the User Equipment (UE) and the mobile core. The RAN is responsible for managing the connection of User Equipment (UEs) to the network via a radio link to the core network (and ultimately the Internet). This includes functions like network resource access, management and mobility. With the advent of 5G mobile networks, there has been an effort to redefine the RAN as a set of disaggregated components with standardized interfaces referred to as an Open RAN[1] (O-RAN). O-RAN can enable mobile network operators (MNOs) to build infrastructure using interoperable components from different vendors. The O-RAN ALLIANCE is the organization formed to define the components and interfaces for Open RAN as shown in Fig 1 below:

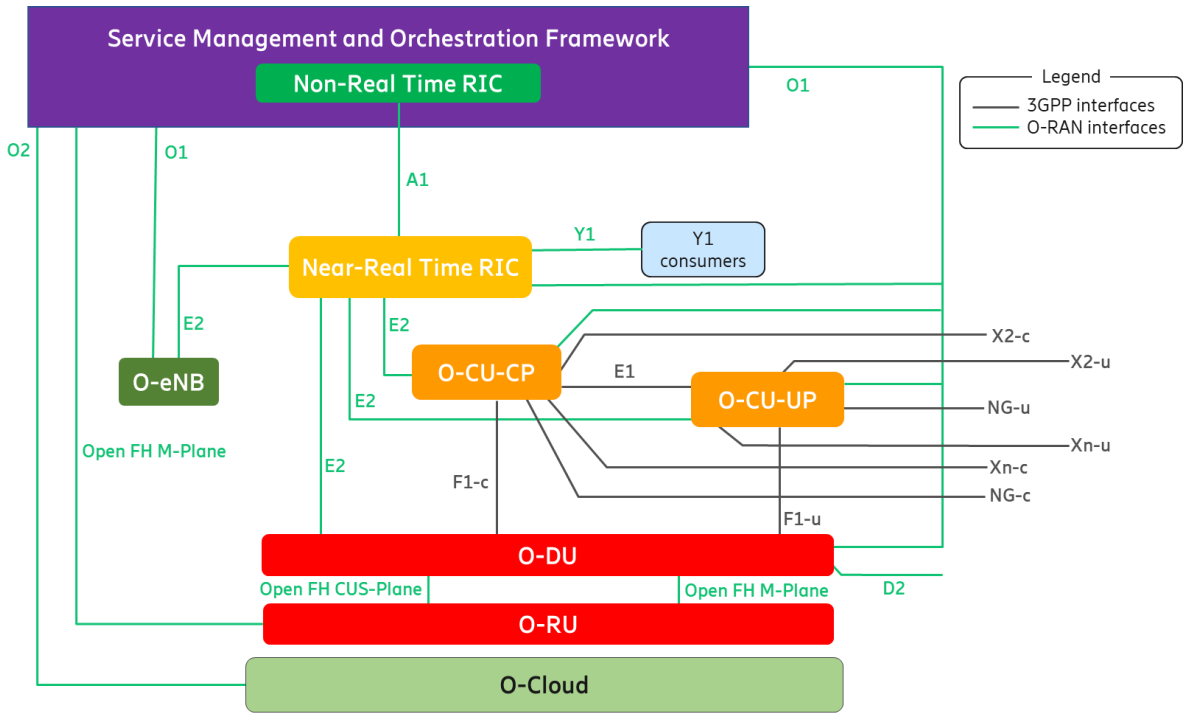


**Figure 1: O-RAN Logical Architecture (from [1])**

The disaggregation of RAN components allows for many of the functions and services to be implemented entirely in software, often running on virtualized compute platforms hosted on commodity IT infrastructure. This can reduce cost and complexity and can increase reliability as components can be operated as cloud native deployments. This also enables the RAN to be updated or modified more quickly than with dedicated hardware or physical appliances used to support the network infrastructure.

[1] This document uses "O-RAN" to denote the general concept of open radio access networks and "O-RAN ALLIANCE" for the O-RAN ALLIANCE standards body that develops specifications for Open RAN.

This flexibility also enables third parties to develop custom monitoring and control software for different operational use cases. This includes the ability to dynamically reconfigure or change the behavior of the RAN in response to changing environment or traffic characteristics. These configuration changes could be triggered by network measurements, security events, or other factors. Such RAN monitoring and management have traditionally been enacted by human administrators, but increasingly these tasks are executed by additional software applications operating through the RAN Intelligent Controller (RIC).

### 1.1. RAN Intelligent Controllers

A RIC is an O-RAN ALLIANCE defined component used to monitor and manage RAN operation. RICs allow the RAN deploy individual applications to optimize for specific use cases or rapidly respond to changes in the RAN. RICs allow MNOs to customize solutions to meet a customer's specific use case, and also provide additional functionality not defined by the O-RAN ALLIANCE for proprietary MNO operations.

RIC functions can be divided into two logical components based on the granularity and responsiveness of RAN measurements and control logic. The Near-Real Time RIC (Near-RT RIC) is assumed to address a subset of cells in a RAN infrastructure and support operational response times under a second. The Non-Real Time RIC (Non-RT RIC) gathers data from the entire RAN and implements macro control logic operating on slower time scales. The Near-RT RIC is designed to enable more direct monitoring and control of individual O-RAN components while the Non-RT RIC is to perform more data and time intensive tasks and coordinate across multiple Non-RT RICs as needed. Near-RT RIC monitoring and control is done through the E2 interface [2] (see Fig 1 and Fig 2). The Near-RT RIC communicates with the Non-RT RIC through the A1 interface [3]. The A1 interface provides a means to coordinate the actions between the Non-RT and one or more Near-RT RICs that each control a localized radio network. There is another interface called the R1 interface that is used by control applications that perform the actual configuration or modification of the RAN.

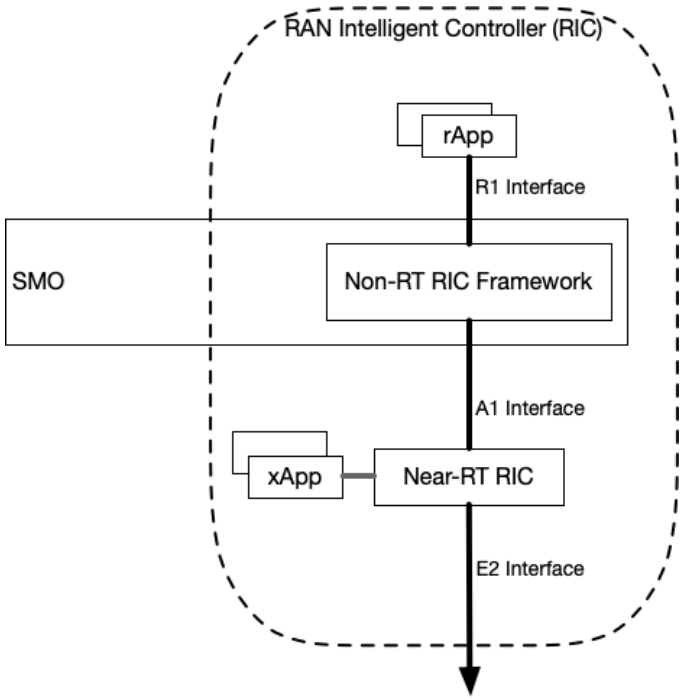


**Figure 2: The RAN Intelligent Controller Architecture**

## 1.2. rApps and xApps

The RIC's purpose is to allow for dynamic access and modification of the control logic for a RAN. The RIC is a platform that can support special purpose applications that can draw information from, or push changes to, an operating RAN configuration. These apps are referred to as rApps, if they operate in the Non-RT RIC, and xApps if they operate in the Near-RT RIC. xApps and rApps could also be deployed in coordination for certain tasks. For example, an rApp may analyze metrics from multiple RAN cell locations, then provide network policies or configuration rules to xApps operating on the Near-RT RIC platform. The xApps would then perform more time sensitive tuning of the RAN based on those polices.

xApps and rApps do not communicate with other O-RAN components directly, but through the RIC platform via O-RAN specified APIs. The RIC platform then forwards messages to other O-RAN components through the O-RAN defined interfaces. These APIs and interfaces have specified protocol stacks and structure for messages [4][5].

xApps and rApps can be hosted on different architectures but are designed with cloud native architecture in mind. This allows the RIC (and O-RAN) to be more responsive to changes or network conditions. This also brings new threats and security requirements that come with cloud-based architectures that were not previously present in legacy mobile networks. Security requirements for O-RAN are maintained by the O-RAN ALLIANCE Security Focus Group (now Working Group 11).

## 1.3. O-RAN ALLIANCE Security Working Group

In the O-RAN ALLIANCE, security specifications are developed in Working Group 11 (WG11). WG11 does not work separately from other working groups in the O-RAN ALLIANCE but instead provides security recommendations for specifications produced by other working groups. WG11 begins defining security requirements for an O-RAN feature or asset once the other working group responsible for that feature/asset signals that their specifications have stabilized. WG11 then begins a threat analysis using the STRIDE model (see Section 2) and introduces any necessary security requirements to mitigate identified threats. A Technical Report (TR) is often produced first, documenting the threat analysis, while related security requirements, changes, and tests are published in existing WG11 Technical Specification (TS) documents.

## 1.4. Zero Trust Architecture and the RIC in Open RAN

The O-RAN ALLIANCE WG11 has stated the goal of implementing a zero trust architecture (ZTA) for O-RAN [6]. Zero trust is a security strategy where every resource request is authenticated and authorized regardless of requestor and resource location. NIST SP 800-207 *Zero Trust Architecture* [7] describes the zero trust tenets adopted by WG11 as the framework for O-RAN ZTA. As zero trust includes aspects of both security architecture and operational practices, the O-RAN ALLIANCE cannot mandate an organization to implement a complete zero trust architecture. The O-RAN ALLIANCE can only specify a set of security requirements for components that would allow an operator to implement a complete zero trust architecture. It would still be possible to deploy and operate O-RAN ALLIANCE specified components in a “non-zero trust” manner, as with any security focused specifications. Zero trust incorporates strategic operational aspects and not a single technological solution.

Zero trust is not a single standard or protocol, but a strategy based on a set of security principles. There is a tension between making specifications too rigid so that only one architecture is possible or having specifications be too vague so that interoperability issues emerge.

For RAN intelligent control, zero trust principles include security requirements for the RIC itself (i.e. Non-RT and Near-RT components) as well as the rApps/xApps that operate on the RIC platform. Each of these components in the system have a set of fine grain resource access policies that authorize the necessary communications during operation.

This report looks at the current state of the O-RAN ALLIANCE RIC security requirements and specifications as it relates to zero trust architecture. As zero trust architectures encompasses aspects that are beyond the normal scope of SDOs aspects like cyber hygiene, operational issues and environmental factors cannot be addressed. Instead, this report will focus on specifications around communication and specifically how identity, authentication and authorization are addressed.

## 2. Security Threats for the RIC, rApps and xApps

The O-RAN ALLIANCE has documented the threat landscape to O-RAN in a Technical Report [8]. This report details the threats to security, privacy and availability to each O-RAN ALLIANCE defined asset (O-RAN software component, data, or interface). The threats to the RIC are similar to the threats to other architectural elements in O-RAN, with additional threats that the usage of xApps/rApps brings. These apps can change the behavior of the RAN and may have communication patterns different than standardized mobile communication workflows. Third-party developed xApps/rApps may also be collecting and transmitting information about the RAN to other components (including external, non-RAN components). The threats can be broken down into a set of categories using the STRIDE methodology [9]:

- **Spoofing (S)**: Threats against authentication. Includes attacks that impersonate network functions, xApps/rApps, etc.
- **Tampering (T):** Threats against integrity. Includes attacks against data as well as components; supply chain attacks that add malware to an xApp for example.
- **Repudiation (R):** Threats against non-repudiation. Attacks that replay commands/requests.
- **Information Disclosure (I):** Threats against confidentiality. This includes attacks like attacker-in-the-middle, passive monitoring of traffic over an O-RAN interface, etc.
- **Denial of Service (DoS) (D):** Threats against availability. This can cover a variety of DoS attacks from external sources or internal (e.g. rouge or misconfigured xApp sending too much traffic leading to resource exhaustion).
- **Elevation of Privilege (E):** Threats against authorization. This included attacks like token theft/hijacking, or a subverted xApp/rApp attempting to access an NF or interface it is not authorized to access.

In response to these threats, the O-RAN ALLIANCE publishes a set of security requirements and controls [10]. Some of the listed security requirements apply to all assets that have a given functionality (such as requirements for API security), while other requirements are specific to particular assets, including those specific to the Near-RT RIC, Non-RT RIC and xApps/rApps. For some security requirements, there are security controls that give more guidance on the technologies to be used to implement the requirement, or additional guidance on how a requirement is to be fulfilled. O-RAN security requirements can be thought of as the necessary ability or feature, and the O-RAN controls are thought of as the technology to be implemented to provide that feature.

The O-RAN ALLIANCE also develops security test specifications[11]. The goal is to have conformance tests for each security controls defined in relevant security requirements and controls specification [10]. The set of tests in the test specification are not applicable to all possible architecture solutions available for deployments. For example, common security architectures and technologies used in cloud-native deployments may address O-RAN security

requirements but not in the manner assumed by O-RAN security controls and corresponding tests. Such deployments could use the underlying virtualization platform to provide the security features. This decouples security functionality from the application code (i.e., business logic). This makes security testing a potentially confusing issue, as it is not the application code (or the O-RAN ALLIANCE defined asset) that is responsible for meeting security requirements, but the underlying virtualization platform.

### 2.1. RIC App Communication Threats

The O-RAN ALLIANCE Technical Report [8] lists identified threats against the RIC and the xApps/rApps that operate within the RIC. The network-based threats are summarized below based on the resulting compromised feature. These threats depend on operational configurations and are not always malicious. For example, some large volumes of legitimate requests/usage can appear very similar to a DoS attack. These threats include:

- Denial of Service (DoS): This is an attack against availability. This could be an rApp/xApp attacking the RIC platform, another rApp/xApp, or another RIC component attacking one or more apps. There could even be an DDoS attack across the A1 interface between the Near-RT and Non-RT RIC elements. These attacks may not purely based on volume (i.e. increased traffic to overwhelm the victim), but attacks that cause increased resource consumption or that prevent some aspect of operation. For example, a malicious app could send messages to delete policy information created by other apps, which negatively impacts RAN operation.
- Spoofed access request: This is an attack against authentication, but can be followed by tampering, escalation of privilege, or information disclosure. Such attacks could come from malicious code or clients that have been able to access and send requests to O-RAN components, claiming to be legitimate parts of the mobile network. This vulnerability could be the result of excessively lax access rules (e.g., firewalls with too permissive rules) or a breach that allowed an attacker to run unapproved (and potentially malicious) code.
- Unauthorized access request: This is similar to spoofed access requests in that it often results in data tampering or disclosure. Unauthorized requests differ from spoofed requests in that these requests could come from an approved component, but the request is outside of its normal operating workflows. One example of this would be a supply chain attack (or other attack) that adds code to an xApp/rApp to request and exfiltrate information that the xApp/rApp would not need for its normal operations but is of interest to the attacker. The application may have credentials that were issued by the virtualization platform during operation that can be validated by other components, but should not make a given request, as it would normally be unnecessary for its operation.

In addition to the set of APIs defined by the O-RAN ALLIANCE for the Near-RT RIC, there is an option of allowing for proprietary APIs [3]. Since the realm of possible proprietary solutions does not have a single specification, there is no way to fully know if new or different threats

exist to these proprietary solutions. It would be up to the operator to work with the vendors using these proprietary APIs to do a threat analysis. While some O-RAN ALLIANCE security requirements [10] would translate to this scenario, it is possible that the controls may not, and that a more tailored assessment would be needed.

## 2.2. RIC Implementation and Supply Chain Threats

There is the threat that vulnerabilities exist, or are introduced (i.e., supply chain attack) into RIC components. This could result in information disclosure, tampering, repudiation or lead to further attacks. O-RAN ALLIANCE security requirements for secure software development are in line with larger industry wide efforts [12], but operators will still need security measures in place to mitigate the threat a compromised component could introduce to an operational deployment. This follows the basic principles of zero trust to assume that any component of a system could already be compromised. The operator may not be able to prevent the specific compromise, but it can limit or prevent the attack from further compromising the deployment through detection and strict access controls.

Mitigations against a compromised component involve detection as well as enforcing strict access controls [12]. Issues involving access control in the RIC are discussed in Sections 3.1 and 3.2. There are requirements for security logging and monitoring in O-RAN [10], but this may not be sufficient for zero trust deployments. In a zero trust architecture, all activity should be logged, to aid in incident response to an attack that has not been detected yet. This includes non-RAN related communication involving O-RAN components. In ZTA, any communication between components is to be denied unless explicitly approved by policy. This is to prevent a compromised component access to external command & control nodes and data exfiltration.

## 3. Considerations for a Zero Trust RIC Architecture

Every system deployment involves decisions that will impact security and these decisions involve trade-offs in complexity and flexibility. These decisions include technology choices as well as the policies implemented with these technologies. While every system should have security requirements, not every requirement demands a single solution. Architects and operators may consider multiple factors when deciding which components to use for a system and how those components will be deployed, configured and operated.

In the considerations below, the O-RAN ALLIANCE security requirements are taken as the basis for discussions. However, the accompanying security controls are not assumed as the only available solution. Other security technologies, or implementation models, used in cloud native applications may have advantages over those specified in the security controls. As such, the requirement should be considered technology agnostic. This also allows for security features to be implemented in an underlying virtualization platform and not in the O-RAN ALLIANCE defined architectural elements.

The discussions below do not exhaustively cover all aspects of O-RAN deployments but provide some security considerations for architects, developers, and operators. The considerations focus on software component development, hosting, and communication policies for those components.

### 3.1. RIC Communication

There are multiple aspects to communication in a ZTA, but for architects and developers, the main techniques to achieve secure communication are authentication, authorization, and confidentiality protection. The goal is to ensure that the only instances of communication that are permitted are explicitly enabled by organizational policy, and all others are denied.

#### 3.1.1. Authentication Considerations

Authentication is defined as “verifying the identity of a user, process, or device, often as a prerequisite to allowing access to resources in a system” [13]. Authentication is important in zero trust, as a subject should only be allowed to interact with resources that are necessary to complete its work (tenet 6 in SP 800-207 [7]). For the RIC, these subjects and resources are the various components (i.e., defined interfaces and services) that communicate over the A1 and E2 interfaces (see Figure 1).

Authentication in a ZTA should be based on a unique identity for every component (subject or resource). This identity is usually associated with a cryptographic artifact such as a cryptographic key that is part of a credential. In O-RAN, this credential is an X.509 digital certificate or a shared secret (for limited uses). This is common for modern architectures but there are different ways these certificates can be issued and the different identity formats used.

There is a difference between the O-RAN ALLIANCE specified Non-RT RIC's and Near-RT RIC's identity controls and cloud native solutions. Currently, the Near-RT RIC uses Universal Unique IDentifiers (UUID) [14] as the namespace for workloads. Individual workloads either generate their own unique string for their UUID or have it assigned during a registration process. This UUID is encoded in a digital certificate that the service uses to authenticate itself when communicating. The Non-RT RIC does not specify a single identity method, with only the requirement that that rApp IDs must be sufficiently random. Using this approach rApps and xApps will have unique identities, but as they are random and may be generated during runtime, additional mappings are required to establish identity that can be used in policy statements. Also, if identities are generated during runtime, these workload identities may change when a new instance is created. This complicates mapping the service in historic logs, as the same instance may have different identities during different operating periods, depending on how they are generated.

A different identity solution is gaining traction in cloud-native deployments. This is the Secure Production Identity Framework For Everyone (SPIFFE). SPIFFE uses a centralized service to issue identities and credentials with a Uniform Resource Indicator (URI) that can include more organizationally defined, deployment relevant strings to link a SPIFFE ID within an administrative scope (e.g., an organization or unique deployment). This identity URI can be maintained across each lifetime of the workload, so that the identity is maintained when the service is restarted/updated, etc. This makes it easier to generate policy as code, as the identity can be scoped beforehand by the administrator, who then can create access policies using the assigned identities. Individual workloads interact with a SPIFFE agent to obtain a keypair and credentials. This technically means that the keypair is generated by the agent and not the actual workload, which introduces another threat vector that could expose the private keys of workloads.

Both systems provide a way to issue credentials with unique identities that can be used to authenticate a workload as a requester or resource. The main difference is the format of these identities and how and when they are generated. A comparison of these methods and how they are specified in O-RAN is given in Table 1:

**Table 1: Comparison of UUIDv4 to SPIFFE**

| | **UUIDv4** | **SPIFFE** |
|---|---|---|
| **Specifying O-RAN Component** | Near-RT RIC, xApps | N/A |
| **Format** | 128-bit string with defined sections | URI with the format "spiffe://<trust-domain>/<service-name>" |
| **Generated/Issued by** | Self, or centralized registration service | Centralized registration service |
| **How Credentials are Issued** | None built in, use of CMPv2 specified | Built-in, can issue X.509 certificate or JSON Web Token (JWT) |
| **Advantages** | Allows for distributed, autonomous identity generation | Allows for identity scoped to administrative domains and service types. |

| | | |
|---|---|---|
| **Disadvantages** | Consistency across instances, mapping to service policies | Private keys generated by third party |

It is possible for a Non-RT RIC and a Near-RT RIC using different identity systems to interoperate, as long as they both ultimately use standardized credentials. The Non-RT and Near-RT portions of the RIC communicate over the O-RAN A1 interface. However, the Common Name and Subject Alternative Name in the certificates used by subjects may be a randomly generated string (e.g. UUID), assigned by SPIFFE, or use some other agreed upon naming convention. However, whatever method is used to assign identities should allow for some degree of policy enforcement based on the identity of the subject. This could include policies that apply to architectural element groups (i.e., all rApps) or individual instances of an architectural element (i.e., a single rApp).

There are operational costs and administrative reasons to choose a common identity solution for different components of the RIC. For cloud-native operations where the workloads can be decoupled from the underlying platform, the actual workload identity framework in use could be opaque to the applications running the business logic. For example, an O-RAN deployment using a virtualization platform like Kubernetes [16] with a service mesh may have the identity management and authentication handled by the service mesh implementation and not the O-RAN components. NIST Special Publication 800-204A [17] provides general guidance for this scenario. NIST SP 800-207A provides guidance for a zero trust architecture approach to cloud native deployments that leverage SPIFFE credentials for workloads [18]. In these architectures, authentication can be checked by components lower in the technology stack so that the application components (in this case, O-RAN components like rApps/xApps and the RIC) do not need authentication logic, unless there is an overriding need to do authentication checks at the application layer.

### 3.1.2. Authorization Considerations

Authorization is "[a]n approval that is granted to a system entity to access a system resource [19]". Authorization is coupled to authentication in that, once the subject's identity is known and verified, authorization verifies the actions the subject is allowed to perform. In zero trust, access requests should be limited to only those actions necessary for a subject to perform its task (tenet 3 in NIST SP 800-207 [7]). This supports the concept of least privilege, so that a subject cannot perform any actions beyond what is needed for its approved workflow.

Authorization policies can vary in granularity and where in the technology stack the policy checks occur. Coarse-grain access policies usually require less information and can be enforced lower in the protocol stack than finer-grain policies. A comparison is given in Table 2 below, where example policies are listed in increasing complexity along with the technology layers at which they can be enforced. Generally, zero trust requires the use of more factors to validate access requests than what could be considered at low levels in the protocol stack. These factors could include environmental aspects (location, time of day, time zone of request) that are not traditional access control considerations.

**Table 2: Authentication Complexity Levels**

| Authorization complexity | Information typically used | Where policies can be applied | Example Policy |
|---|---|---|---|
| **None** | None | N/A | "Public webpage where any request is accepted" |
| **Basic** | IP address, source port, etc. | Network components, firewalls, etc. | "Only requests originating from 192.168.1.0/24 are accepted." |
| **Moderate** | Application protocol fields. | Application gateways, smart firewalls | "The given client may only issue HTTP GET requests to this API" |
| **Advanced** | Application protocol and other historic or environmental factors. | Application, specialized gateways | "The client may only delete data entries that it created." |

ZTA fine grain access controls require more complex logic so it may be advantageous to distribute the PEP across multiple systems to reduce latency and filter out request that fail basic checks before having those requests processed by application layer logic. For example, a ZTA may use a micro-segmentation solution to restrict communication between resources using network level policies and use other solutions with more complex application layer logic to enforce policies using application layer information. This would allow access requests that fail lower-level policy checks to be dropped before they reach the level of requiring more inspection at the application layer.

### 3.1.3. Using Tokens for Identity Over Certificate-based Systems

An alternative to encoding workload identity in a X.509 certificate is to use a JSON web token (JWT) or other application layer token format. JWT format is considered more web-friendly and thus better suited for where workloads communicate through APIs. JWTs also have the advantage of being easier to pass through gateways, load balancers and other supporting infrastructure that terminate a TLS session before the business logic of the service receives the API call. In many service mesh architectures, it is common to have a third-party component establish and manage the TLS connections between services. This means that the service mesh component terminates the TLS connection, and only the original API call is sent on to the business logic. The receiving resource has little knowledge of the identity of the original subject other than to infer it must have passed the authentication checks performed by the service mesh component.

Using token-based identity artifacts allow for authentication and even confidentiality protection to be performed at the application layer and not the lower network (i.e. TLS) layer. This has the advantage that the identity and confidentiality can survive traversing gateways, proxies, etc. by enforcing security at the data object level rather than the channel level. More discussion of channel security vs. object security can be found in Section 4.

The O-RAN ALLIANCE currently specifies the use of mTLS with X.509 certificates as the protocol to provide authentication [10]. However, for cloud-native deployments, an option to use token-based identity artifacts or a hybrid approach may become necessary. This would provide more flexibility to apply zero trust policy checks in a cloud-native O-RAN deployment.

## 3.2. O-RAN and OAuth2 Authorization

The O-RAN ALLIANCE specifies OAuth2 as the protocol used to assert and check authorization policies [10]. OAuth2 is protocol that allows an authenticated subject to indicate it is authorized by presenting an object (called a token) during a request [20]. OAuth2 relies on a trusted third-party authorization server to generate a token containing the access privileges and necessary additional information needed by a resource to ensure an access request from a specific subject is authorized according to policy. A resource can deny a request that fails to include a valid token or a token that does not include the necessary access privileges.

There are multiple authorization flows available within the OAuth2 specification and O-RAN specifies the use of OAuth2 access tokens for service-to-service authorization. In this model, a requesting subject first obtains a token, then includes that token in request to a resource. The high-level communication flow is given in Figure 3:

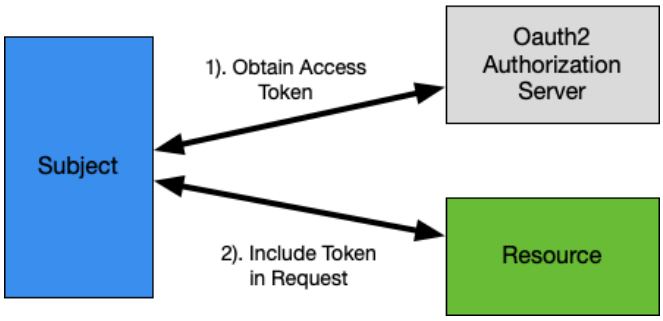


**Figure 3: OAuth2 Access Token Request and Usage**

In the above, the subject first sends a request to the authorization server for an access token. The authorization server authenticates the requester and responds with an access token that encodes the resource access privileges for the subject. The token incudes a digital signature generated by the authorization server that can be used to verify that the token was properly issued (usually an asymmetric digital signature keypair). Every service in the deployment would need the public portion of this token signing key to verify the signature over the token.

Once the requester has the token it can send an access request to the desired resource and include the token in that request. The resource (or some other on-path component) then validates the token by verifying the signatures (the resource must have the public key of the authorization server to do this) and checking the list of privileges (sometimes referred to as the "scope") of the token. If successful, the operation continues, otherwise the request is denied. The scope and its validation are workflow-defined so all parties in the transactions (subject, resource and authorization server) need to know the range of possible scopes and policies that apply to access requests. Like the example policies in Table 2, there is a range of possible granularities and policies that can be applied if all parties are aware of how to request, use and validate the resulting access tokens.

### 3.2.1. Zero Trust Considerations for Using OAuth2 Access Tokens

Zero trust principles call for fine-grained authorization for every resource request. The scope attribute in an OAuth2 access token is the field where policies can be encoded for PEPs to enforce. It is helpful if there is agreement among service producers, requesters, and operators around the policy elements that can be encoded in a token scope field so the token request, use and validation share a common understanding. Ultimately the access policies are decided by the enterprise and having authorization checks decoupled from the subject and resource is possible in cloud native deployment.

In a cloud-native deployment, platform-based components (e.g., service mesh) or middleware could provide authentication checks such that that the RIC application code does not need to implement these controls. For example, O-RAN security requirements include authentication (via mTLS) and authorization (via OAuth2) checks to be performed for all communication over the R1 interface between rApps and the Non-RT RIC. With the decoupling advantage of cloud-native deployment, these checks could be done by platform-dependent components and not the actual rAPP business logic components. Figure 4 below shows this advantage in a Kubernetes example. Here, the service mesh sidecar provides the security functionality before the business logic code receives the request. The token and scope formats can be opaque to the actual rApp and Non-RT RIC and be deployment specific if so desired. Different models of decoupled implementation of such security controls can be applied in other virtualization platforms [21].

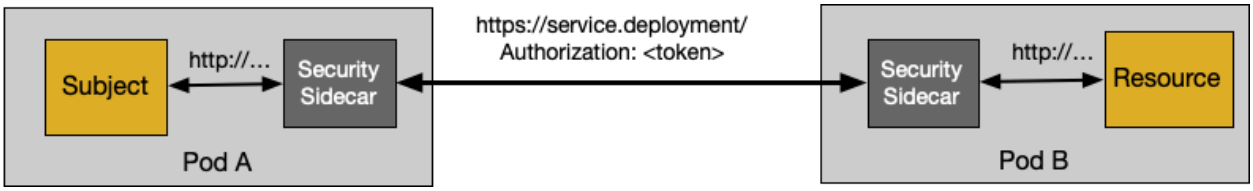


**Figure 4: Pods using Sidecars to Provide Security Services**

### 3.2.2. Scope Considerations in Authorization in O-RAN

The advantage of having the authorization decoupled from the business logic component is that the deploying organization can set the granularity. This could be as general or specific as desired to fit policies and changed when the ability or need to implement fine-grained access control increases.

Currently, the O-RAN ALLIANCE does not specify the format of the scoping in OAuth2, and there are different levels that could be applied. The level to apply depends on the organization's zero trust policies and operational maturity. Some high-level scoping factors are given in Table 3 below. This would be scoping at the asset level, to the API, and ultimately, the type of action.

**Table 3: Example O-RAN Authorization Scopes**

| Scope | Example Authorization Granularity |
|---|---|
| **O-RAN Defined Asset** | Near-RT RIC, xApp, rApp |
| **API** | A1 Interface policy API, but not A1 EI API |
| **Individual calls** | Authorized to read A1 policies, but not create or delete policies |

### 3.2.3. Other Information Sources in Access Control

A more advanced zero-trust architecture will have PEPs that use data sources and attributes beyond relying on the access token as the only authorization check. Information like the current security posture of the requester and resource, and behavioral access patterns cannot be encoded in an OAuth2 token. This access decision could also include information received from external threat intelligence or anomalous behavior seen that doesn't include authorization failure signaling. MNO wishing to move to more mature ZTA will need a comprehensive authorization framework that does not solely rely on the OAuth2 tokens but uses OAuth2 as a component.

Since many of these attributes may not be encoded in a token, it also means policy checks on these attributes may not be done using token validation logic. For authorization checks within a deployment (cloud-native or otherwise) it might be advantageous to send authorization checks to a centralized Policy Decision Point (PDP) that has access to information about various attributes used in policy checks. This means that authorization checks at the resource are reduced to a call to the central PDP, and the resource either continues operation or denies the request based on the PDP's response. This centralized PDP has the advantages when updating policies (only one component needs the update) and a centralized point for security data collection in that the PDP is the only component that needs advanced monitoring information about the infrastructure like access patterns, security posture of subject and resource, etc. The disadvantage is that this creates a single point of failure in that if the central PDP in unavailable, all access requests throughout the RAN cannot be authorized. Also, a centralized PDP for one organization would likely not have this information for subjects that come from different

administrative domains. A centralized PDP may work within an enterprise's infrastructure but may need other solutions for requests coming from outside of the enterprise. In O-RAN, this could arise from an rApp that is implemented in a software-as-a-service model instead of a software package that is operated on the operator's infrastructure.

### 3.3. Confidentiality Considerations and Quantum-Resistant Cryptography

Once a subject has successfully authenticated itself and the resource validates that the subject is authorized, communication can take place. This is where confidentiality must be protected. Zero trust architecture calls for all data to be protected in transit and at rest. O-RAN security standards address the transit portion by specifying the use of mTLS for encrypting communication between O-RAN components.

A TLS connection is used with one of the cryptographic algorithms defined for use with the protocol [22]. In the most common usage of TLS, the initial connection setup (referred to as the handshake) uses an asymmetric algorithm to authenticate the components in the transaction before establishing a shared session key that uses a more efficient symmetric cryptographic algorithm.

At present, the set of asymmetric algorithms defined for use in O-RAN [23] is believed to be strong enough to provide confidentiality protection. In the future, advances in quantum computing threaten this, as a powerful enough quantum computer could render commonly used asymmetric algorithms such as RSA and elliptic curve algorithms vulnerable to cryptoanalysis. This means that SDOs and MNOs need to develop plans to migrate to quantum resistant cryptographic algorithms, often referred to as post quantum cryptography (PQC) [24]. PQC algorithms are based on mathematical principles such that a quantum computer does not offer any advantages over a classical computer in cryptoanalysis. Asymmetric algorithms are considered the most vulnerable to a cryptographically relevant quantum computer CRQC, and the symmetric algorithms used with TLS session keys are not as vulnerable and any risks from a CRQC may be mitigated by using longer key lengths.

This migration to PQC algorithm use will take time as it is not simply an upgrade of the components that perform TLS. The underlying infrastructure that generates keypairs, digital certificates for PQC public keys, and associated network monitoring and logging tools must also be upgraded to understand the use of PQC algorithms. The operational cost of PQC over legacy algorithms may also become a factor in operations and should also be taken into consideration when migrating.

#### 3.3.1. State of PQC in O-RAN

The O-RAN ALLIANCE recognizes CRQCs as a future threat to O-RAN security. In response, the security working group in the O-RAN ALLIANCE has begun work on specifying PQC algorithms for use with O-RAN. The first step in this effort is an inventory of protocols used in O-RAN that specify algorithms that would be vulnerable to a CRQC [25]. Once the inventory is complete, replacement PQC candidate algorithms can be chosen and specified for use with O-RAN. This cannot fully be done in the O-RAN ALLIANCE but requires coordination with other SDOs that

specify the protocols used in O-RAN. For example, the Internet Engineering Task Force (IETF) develops the specifications for TLS and would be responsible for defining PQC use with TLS. O-RAN only specifies which of these algorithms are to be supported in O-RAN.

As of the time of writing, PQC algorithm support exists in open-source cryptographic libraries [26]. Studies on the PQC algorithms' impact on operations in other areas could be used to infer the potential impact of PQC in O-RAN.

## 4. Channel Security Versus Object Level Security for the RIC

In zero trust, there is no single solution that applies to all security-related issues. There is also the question of where confidentiality is to be applied. There are two views of confidentiality that would apply to O-RAN: channel confidentiality and object confidentiality. They are not mutually exclusive, in that both can be applied as they address different threats:

- Channel security protects data transmitted between directly communicating components. This is to mitigate a potential attack on the communication path, either passive or active.
- Object security protects data end-to-end as it traverses the entire network from the originator to the ultimate consumer. This may be desired in architectures where the data producer and consumers do not directly interact but rely on one or more intermediate components to store or process data. This is very common in cloud native applications that use ephemeral microservices with centralized data repositories or message brokers to manage data producer/consumer communication streams. Object level security can also continue with the object as the data traverses gateways or proxies that terminate a link segment and the channel security that protected that link.

A service-based O-RAN architecture will likely have decoupling between application code and data storage. This means there is the threat of data manipulation ("Tampering" in the STRIDE model) to O-RAN data or policy statements. A malicious actor, including the data management service itself, might alter or modify data after it is stored by the producer and before it reaches the interested consumer(s).

### 4.1. Data Protection Requirements in O-RAN

There are requirements for data protection in O-RAN deployment and operations. The O-RAN ALLIANCE has requirements on integrity protection of software components when being deployed and upgraded [10]. This is to prevent supply chain attacks in which a legitimate version of a software component is replaced or modified to include malicious code. There are also requirements for protection of data at rest in components that store data. However, this does not require the original data producer to include any security artifacts such as digital signatures or to use end-to-end encryption. This model of data at rest security may not cover malicious insider attacks, but other mitigating factors, such as continuous monitoring, may address these risks.

#### 4.1.1. Advantages to Adding Object-Level Security to O-RAN

Adding security artifacts (e.g., digital signatures) directly to data objects to enable source authentication and integrity checks can protect data as it traverses the network from producer to consumer and any intermediary data storage or broker services. This enables data consumers to verify that the data was not tampered with in transit or intermediate storage, as well as identify the data producer. This also allows a given policy statement or data element to be tied to a producer and tracked throughout its lifecycle. This also enables a "layered" zero

trust architecture where data is protected as well as the components producing, consuming and storing a given data element.

### 4.1.2. Challenges to Adding Object Level Security to O-RAN

Adding object-level data security artifacts brings additional complexity to O-RAN operations for both data producers and consumers. Each will need to include new features and functionality to include and utilize these artifacts:

- Data producers will need to be able to sign and/or encrypt data before sending it. This means that the producer must be configured to use a key pair for these operations. This keypair may be different than the keypair used for channel level authentication as in mTLS. This may be because the data producer may not be the same component that sets up the protected channel to send the data. Data producers also need a way to distribute or make their public keys available to data consumers for decryption or signature validation.
- Data consumers will need to be able to validate signatures and/or decrypt data objects. This includes having a way to discover or otherwise obtain the public keys needed to validate signatures or decrypt data objects. Unless all the data objects can be specified with a common format, these operations may need to be done at the application layer by business logic components, as it may not be possible to do it in low-level software or on-path proxies, sidecars or similar support components often used in cloud-native infrastructure.

The key management challenges may be addressed by the virtualization infrastructure. Having data consumers initially configured with or allowed to discover and obtain a producer's key(s) from a central infrastructure avoids having the data producers also distribute their keys and maintains the asynchronous features of data publishes and consumers in cloud native applications.

## 5. Conclusion and Future Steps

O-RAN is an effort to create a disaggregated architecture and set of standardized interfaces and service descriptions for mobile network RANs. The O-RAN architecture and specifications enable vendors to implement using a service mesh architecture deployed in virtualized cloud infrastructures. These cloud-native deployments have advantages in flexibility and scalability but introduce new threats that were not present in legacy RAN implementations. Cloud-native applications have been increasingly common in IT operations, and new security concepts and technologies have been developed to deal with these new threats. One concept is the principles of zero trust, which is a strategy to plan, deploy and operate a digital infrastructure where security controls between individual components is strictly enforced.

The O-RAN ALLIANCE analyzed the new threats facing an open RAN [8] and possible solutions to address these new threats, by introducing zero trust principles. The O-RAN ALLIANCE has adopted zero trust as the desired security strategy for O-RAN. This entails following the zero trust tenets described in NIST SP 800-207 [7] to produce a set of security requirements [10] to provide authentication, authorization, and confidentiality to O-RAN operations. This is still ongoing work in the O-RAN ALLIANCE to address the unique implications of a cloud-native O-RAN deployments. These include:

- What security controls need to be implemented in O-RAN components, versus what security functionality can be provided by the underlying virtualization platform? Cloud-native applications decouple the application workload from the underlying infrastructure that provides compute, storage, and networking. This enables more flexibility in the structure of the overall application and allows the workload developers to focus on the business logic, while specialized components handle security, networking, storage, etc. This can make developing security specifications and testing difficult as it expands the solution set possible for deployments. Different cloud-native technologies and architectures could be used to meet the O-RAN security requirements but would fail to meet the specific O-RAN security controls.
- What level of authorization can be codified in specifications to enable zero trust, and how can it remain flexible to future use cases? Zero trust principles call for fine-grained access controls, but that can mean different things depending on what capabilities are available or possible in a given architecture. This level of authorization granularity may be difficult to include in specifications, as it may artificially limit architecture and deployment options.
- What risk does a CRQC present to O-RAN, and what specifications need to be updated to include quantum attack-resistant algorithms? The O-RAN ALLIANCE recognizes the threat a potential CRQC brings to RAN operations. In response, WG11 is attempting to produce a report on the key issues of introducing PQC algorithms into O-RAN and identifying the protocols that need to migrate to PQC algorithms [25]. The next phase is to identify PQCs to use with O-RAN and analyze the impact of migration on operations.
- Should security features be implemented to protect communication channels or implement object-level security for O-RAN operations? Legacy RAN deployments only

considered channel security, as most of the RAN control plane was a monolithic application. O-RAN architecture disaggregates the RAN control plane into a set of services and assets that allow for more cloud native deployment options. The cloud native approach seeks to decouple the application logic from the underlying infrastructure with a centralized data storage service being one architecture solution. This means that there is a third party (the data storage service) between data producers and consumers. There is the additional threat of the data being altered, or false data being injected into this architecture. This threat is present in the RIC, as there is a decoupling between the applications that operate through the RICs and the data sources they use in their workflows. This could be mitigated by introducing object-level (or data-level) security where the data object is digitally signed by the producer for integrity and authenticity and/or encrypted for confidentiality. This brings zero trust down to the data level, in that no resource (including the data storage service) is inherently trusted. Currently, O-RAN security requirements put more emphasis on channel security between O-RAN components. As O-RAN and mobile telecommunication networks move to a more cloud native posture, there should be an analysis of where object or data-level security is needed for the RIC to bring O-RAN further along the zero trust journey.

O-RAN security is improving with the effort to enable a zero-trust architecture. There are still security considerations that must be addressed to improve security and face new threats that come with cloud-native architectures. However, there is a limit as to what can be done through specification and what is the responsibility of developers and operators. This may lead to new operational best practices and recommendations as O-RAN is deployed by MNOs.

## References

[1] O-RAN ALLIANCE. *O-RAN Architecture Description*. Versions R004-v14. June 2025. https://specifications.o-ran.org/specifications.

[2] O-RAN ALLIANCE. *Near-RT RIC Architecture*. Versions R004-v07. O-RAN ALLIANCE, February 2025. https://specifications.o-ran.org/specifications.

[3] O-RAN ALLIANCE. *Non-RT RIC: Architecture*. Versions R004-v07. O-RAN ALLIANCE, June 2025. https://specifications.o-ran.org/specifications.

[4] O-RAN ALLIANCE. *A1 Interface: General Aspects and Principles*. Versions R004-v05.01. O-RAN ALLIANCE, June 2025. https://specifications.o-ran.org/specifications.

[5] O-RAN ALLIANCE. *E2 General Aspects and Principles*. Versions R004-v07. O-RAN ALLIANCE, February 2025. https://specifications.o-ran.org/specifications.

[6] *Zero Trust Architecture for Secure O-RAN*. White Paper. O-RAN ALLIANCE WG11, 2024. https://mediastorage.o-ran.org/white-papers/O-RAN.WG11.ZTA for Secure O-RAN White Paper-2024-05.pdf.

[7] Rose, Scott, Oliver Borchert, Stu Mitchell, and Sean Connelly. *Zero Trust Architecture*. NIST Special Publication (SP) 800-207. National Institute of Standards and Technology, 2020. https://doi.org/10.6028/NIST.SP.800-207.

[8] *O-RAN Security Threat Modeling and Risk Assessment*. O-RAN ALLIANCE Technical Report, 2025. https://specifications.o-ran.org/specifications.

[9] "The STRIDE Threat Model." Accessed August 8, 2025. https://learn.microsoft.com/en-us/previous-versions/commerce-server/ee823878(v=cs.20).

[10] O-RAN ALLIANCE. *Security Requirements and Controls Specification*. Versions r004-v12. O-RAN ALLIANCE, June 2025. https://specifications.o-ran.org/specifications.

[11] O-RAN ALLIANCE. *O-RAN Security Test Specification*. Versions R004-v10.0. O-RAN ALLIANCE, n.d. https://specifications.o-ran.org/specifications.

[12] O-RAN ALLIANCE. *O-RAN Study on Security for Application Lifecycle Management*. O-RAN ALLIANCE Technical Report, 2025. https://specifications.o-ran.org/download?id=916

[13] National Institute of Standards and Technology (US). *Minimum Security Requirements for Federal Information and Information Systems*. (Department of Commerce, Washington, D.C.), Federal Information Processing Standards Publications (FIPS) 200, 2006. https://doi.org/10.6028/NIST.FIPS.200.

[14] Davis, Kyzer R., Brad Peabody, and P. Leach. *Universally Unique IDentifiers (UUIDs)*. Request for Comments RFC 9562. Internet Engineering Task Force, 2024. https://doi.org/10.17487/RFC9562.

[15] "SPIFFE – Secure Production Identity Framework for Everyone." Accessed August 1, 2025. https://spiffe.io/.

[16] Kubernetes. "Production-Grade Container Orchestration." Accessed August 11, 2025. https://kubernetes.io/.

[17] Chandramouli, Ramaswamy, and Zack Butcher. *Building Secure Microservices-Based Applications Using Service-Mesh Architecture*. NIST SP 800-204A. National Institute of Standards and Technology, 2020. https://doi.org/10.6028/NIST.SP.800-204a.

[18] Chandramouli, Ramaswamy. *A Zero Trust Architecture Model for Access Control in Cloud-Native Applications in Multi-Location Environments*. NIST SP 800-207A. National Institute of Standards and Technology, 2023. https://doi.org/10.6028/NIST.SP.800-207A.
[19] Shirey, Robert W. *Internet Security Glossary, Version 2*. Request for Comments RFC 4949. Internet Engineering Task Force, 2007. https://doi.org/10.17487/RFC4949.
[20] Hardt, Dick. *The OAuth 2.0 Authorization Framework*. Request for Comments RFC 6749. Internet Engineering Task Force, 2012. https://doi.org/10.17487/RFC6749.
[21] Chandramouli, Ramaswamy, Zack Butcher, and James Callaghan. *Service Mesh Proxy Models for Cloud-Native Applications*. NIST SP 800-233. National Institute of Standards and Technology (U.S.), 2024. https://doi.org/10.6028/NIST.SP.800-233.
[22] "Transport Layer Security (TLS) Parameters." Accessed August 11, 2025. https://www.iana.org/assignments/tls-parameters/tls-parameters.xhtml.
[23] O-RAN ALLIANCE. *Security Protocols Specifications*. Versions R004-v12.0. O-RAN ALLIANCE, June 2025. https://specifications.o-ran.org/specifications.
[24] Barker, Elaine, Lily Chen, David Cooper, et al. *Considerations for Achieving Cryptographic Agility: Strategies and Practices*. NIST CSWP 39 2pd. National Institute of Standards and Technology, 2025. https://doi.org/10.6028/NIST.CSWP.39.2pd.
[25] O-RAN ALLIANCE. *Study on Post-Quantum Cryptography*. O-RAN ALLIANCE Technical Report. Oct. 2025. https://specifications.o-ran.org/download?id=996
[26] OpenSSL Foundation "The Features of 3.5: Post-quantum cryptography" Accessed Sept 8, 2025. https://openssl-foundation.org/post/2025-04-22-pqc/index.html