World Journal of Advanced Engineering Technology and Sciences

eISSN: 2582-8266

Cross Ref DOI: 10.30574/wjaets

Journal homepage: https://wjaets.com/

(Review Article)

Implementing a multi-tenant FIDO relying party server: Architecture, security, and scalability considerations

Ravikanth Reddy Gudipati *

University at Buffalo, USA.

World Journal of Advanced Engineering Technology and Sciences, 2025, 15(01), 765-773

Publication history: Received on 01 March 2025; revised on 07 April 2025; accepted on 10 April 2025

Article DOI: https://doi.org/10.30574/wjaets.2025.15.1.0304

Abstract

The transition to passwordless authentication using FIDO standards marks a transformative shift in modern security architecture, particularly in multi-tenant environments. This technical exploration delves into the comprehensive implementation considerations for building robust multi-tenant FIDO Relying Party (RP) servers. The content addresses key architectural components, including WebAuthn and CTAP protocol integration, database architecture models, tenant isolation strategies, and identity federation mechanisms. Critical security aspects, including cross-tenant protection and audit logging, are examined alongside cloud-native deployment considerations. The discussion encompasses essential elements of scalability, maintainability, and security controls required for successful FIDO-based passwordless authentication in multi-tenant environments.

Keywords: Multi-Tenant Authentication; FIDO Implementation; Passwordless Security; Cloud-Native Architecture; Identity Federation

1. Introduction

The transition to passwordless authentication represents a fundamental shift in modern security architecture, with FIDO (Fast Identity Online) standards leading this transformation. According to Axiomatics' State of Authorization Report 2023, 74% of organizations consider identity and access management modernization a critical priority, with 67% actively working to implement advanced authentication methods, including FIDO standards [1]. This significant focus on authentication modernization reflects the growing understanding that traditional access control methods no longer meet the security demands of modern enterprises, particularly as organizations continue their digital transformation journeys.

The emergence of multi-tenant architectures in FIDO implementations has become increasingly crucial as organizations expand their digital footprints. The 2023 State of SaaSOps Report reveals that IT teams now manage an average of 130 SaaS applications per organization, representing an 18% increase from the previous year, with 62% of organizations reporting that their SaaS stack has grown significantly in the past three years [2]. This exponential growth in SaaS adoption has created a pressing need for scalable, multi-tenant authentication solutions to efficiently manage authentication services across multiple business units, customer segments, and partner organizations while maintaining strict security boundaries.

The complexity of modern enterprise environments has further emphasized the importance of robust multi-tenant FIDO Relying Party (RP) servers. With 71% of organizations citing security and compliance as their top SaaS management challenge [2] and 63% of enterprises struggling with authorization scalability across their growing application landscape [1], the need for sophisticated authentication architectures has never been more apparent. Multi-tenant FIDO

* Corresponding author: Ravikanth Reddy Gudipati.

implementations must address these challenges while providing the flexibility to support diverse authentication requirements across different organizational units and user populations.

This article provides a comprehensive examination of the architectural considerations and implementation strategies for building robust multi-tenant FIDO RP servers. By incorporating the findings that 82% of organizations plan to increase their investment in authorization and authentication technologies over the next year [1], we explore how these systems can be designed to meet both current and future security demands. The discussion encompasses critical aspects of security, scalability, and maintainability that organizations must consider when deploying FIDO-based passwordless authentication in multi-tenant environments, especially given that 58% of organizations report struggling with maintaining consistent security policies across their SaaS applications [2].

Table 1 Enterprise Security Challenges and Adoption Trends [1,2]

Metric	Percentage
Organizations prioritizing IAM modernization	74%
Organizations implementing advanced authentication methods	67%
Organizations reporting significant SaaS stack growth (past 3 years)	62%
Organizations citing security/compliance as a top challenge	71%
Organizations struggling with authorization scalability	63%
Organizations planning to increase authentication investment	82%
Organizations struggling with consistent security policies	58%

2. Fido authentication fundamentals

2.1. WebAuthn and CTAP Protocol Stack

The FIDO2 framework represents a significant advancement in authentication technology, built upon the foundational components of Web Authentication (WebAuthn) API and the Client to Authenticator Protocol (CTAP). WebAuthn has gained substantial traction, with over 85% of modern browsers now supporting the protocol and major platforms like Windows Hello, Apple's Touch ID, and Android's biometric systems offering native integration [3]. The WebAuthn API's capability to leverage platform authenticators and external security keys has proven particularly valuable for enterprise deployments, where the elimination of password-based vulnerabilities has become a critical security objective.

The CTAP protocol's role in facilitating communication between client platforms and external authenticators has become increasingly vital in enterprise environments. The protocol's support for both platform authenticators and roaming authenticators enables organizations to implement flexible authentication strategies while maintaining robust security standards. According to industry implementation data, WebAuthn's cryptographic challenge-response mechanism provides significantly stronger protection against phishing attacks compared to traditional two-factor authentication methods, with organizations reporting near-zero successful phishing attempts after implementation [3].

In multi-tenant contexts, these protocols require sophisticated isolation and identity management strategies. The implementation must maintain distinct cryptographic boundaries for each tenant while ensuring FIDO2 specification compliance. This includes the management of tenant-specific attestation requirements, credential storage systems, and authentication policies that can scale across diverse organizational needs while maintaining security integrity.

2.2. Passkey-Based Authentication

Passkeys have emerged as the next evolutionary step in FIDO credentials, fundamentally transforming the user authentication experience. Enterprise deployment studies indicate that FIDO2 implementations have shown particular success in large-scale environments, with organizations reporting up to a 50% reduction in authentication-related support tickets and significant improvements in user satisfaction scores [4]. The enhanced security model, combined with improved user experience, has made passkeys an increasingly attractive option for organizations seeking to modernize their authentication infrastructure.

The implementation of passkey management in multi-tenant environments requires careful consideration of various architectural aspects. Enterprise deployments have demonstrated that successful FIDO2 implementations typically require a phased approach, with initial deployments focusing on specific user groups or applications before expanding across the organization [4]. This methodical approach allows organizations to validate their implementation strategies and refine their deployment processes while maintaining strict tenant isolation.

Recovery procedures and credential management systems have proven particularly crucial in enterprise environments. Organizations implementing FIDO2 report that well-designed recovery mechanisms and clear administrative procedures are essential for maintaining both security and user satisfaction. The ability to manage credentials across tenant boundaries while maintaining strict isolation has become a key requirement for enterprise-scale deployments, particularly in organizations with complex organizational structures or regulatory requirements.

2.3. Multi-Tenancy Models and Architecture

The architectural design of a multi-tenant FIDO RP server's database infrastructure fundamentally shapes its security posture, scalability potential, and operational complexity. Research into multi-tenant security architectures indicates that organizations must carefully evaluate their specific requirements across three critical dimensions: security isolation, resource sharing, and operational efficiency [5]. The implementation of appropriate multi-tenant database models becomes particularly crucial as organizations scale their authentication services across multiple customer bases and business units.

2.3.1. Database Architecture Options

The selection of database architecture for multi-tenant FIDO RP servers represents a critical decision point that significantly impacts both security outcomes and operational efficiency. According to distributed database architecture studies, the choice of multi-tenant data architecture can lead to substantial variations in both maintenance overhead and security control implementation, with each model offering distinct advantages for different use cases [6].

2.3.2. Database-per-Tenant Model

The database-per-tenant model embodies the highest level of isolation available in multi-tenant architectures. This approach, where each tenant receives a completely separate database instance, provides the strongest possible security boundaries between tenants [5]. Each tenant maintains their own dedicated credential store, user mappings, and authentication logs, ensuring complete data isolation. This model has proven particularly effective for organizations with stringent compliance requirements or those operating in heavily regulated industries where data segregation is mandatory.

The practical implications of implementing a database-per-tenant model extend beyond security considerations. While this approach offers the clearest compliance story and simplest security architecture, it introduces significant operational overhead in terms of database maintenance, backup procedures, and resource allocation [6]. The model requires careful consideration of database provisioning processes, connection management, and resource optimization strategies to maintain efficient operations at scale.

2.3.3. Schema-per-Tenant Model

The schema-per-tenant model represents a middle-ground approach that balances security requirements with operational efficiency. This architecture utilizes separate schemas within a shared database infrastructure to maintain logical isolation between tenants while optimizing resource utilization [6]. The model has demonstrated particular effectiveness in environments where organizations need to maintain strong tenant isolation without incurring the full operational overhead of separate databases.

Implementation considerations for the schema-per-tenant model focus heavily on security control implementation and resource management. Organizations must carefully design their schema management processes, access control mechanisms, and monitoring systems to maintain effective tenant isolation [5]. The model requires robust automation for schema provisioning and management to handle tenant lifecycle operations efficiently while maintaining security boundaries.

2.3.4. Shared Database Model

The shared database model represents the most resource-efficient approach to multi-tenant data management, though it demands the most sophisticated security controls. This model utilizes row-level security and tenant context management to maintain isolation within a single shared schema [6]. While offering the highest level of resource efficiency, this approach requires careful attention to security implementation at the application and database levels to prevent unauthorized cross-tenant access.

Security implementations in shared database models must account for all potential access patterns and data isolation requirements. Organizations implementing this model must invest significantly in security control development and testing to ensure robust tenant isolation [5]. The architecture requires sophisticated query design patterns and careful attention to performance optimization to maintain both security and efficiency at scale.

Table 2 Comparative Analysis of Database Architecture Approaches [5,6]

Database Model	**Primary Strength**	**Key Security Feature**	**Main Operational Consideration**	**Resource Management**	**Implementation Focus**
Database-per-Tenant	Complete Isolation	Dedicated Instance	Heavy Maintenance Load	Individual Resources	Compliance & Security
Schema-per-Tenant	Balanced Approach	Schema Separation	Automated Management	Shared Infrastructure	Resource Optimization
Shared Database	Resource Efficiency	Row-Level Security	Performance Optimization	Unified Resources	Security Controls

3. Tenant isolation strategies

The implementation of robust tenant isolation in multi-tenant authentication systems requires a comprehensive approach that spans multiple security dimensions. According to NIST's Digital Identity Guidelines, authentication systems must implement appropriate security controls for each Authenticator Assurance Level (AAL), with AAL3 requiring multiple authentication factors and replay-resistant authenticators [7]. In multi-tenant environments, these requirements must be carefully implemented to maintain distinct security boundaries between tenants while ensuring compliance with NIST's specified authentication protocols and cryptographic standards.

3.1. Authentication Workflows

Authentication workflow isolation represents a fundamental security requirement in multi-tenant environments. According to multi-tenancy security best practices, the implementation of strong identity and access management (IAM) controls serves as the foundation for maintaining secure tenant boundaries [8]. Each tenant's authentication policies must be consistently enforced throughout the authentication lifecycle, with particular attention paid to session management and credential validation processes.

Session management in multi-tenant environments demands rigorous attention to tenant context preservation. NIST guidelines emphasize the importance of implementing robust session management protocols that maintain cryptographic binding between the authenticator output and the session being established [7]. This becomes particularly critical in multi-tenant systems where session contexts must be strictly isolated to prevent any potential cross-tenant access.

Credential validation pathways must incorporate comprehensive tenant isolation measures. NIST specifications require that authentication systems implement approved cryptographic protocols and validate all received authentication assertions [7]. In multi-tenant environments, these validation processes must be augmented with tenant-specific validation paths and ceremony bindings to ensure proper isolation between tenant authentication workflows.

Risk assessment and adaptive authentication policies require careful implementation in multi-tenant contexts. Security experts recommend implementing comprehensive monitoring and analysis capabilities to detect potential security threats and unauthorized access attempts across tenant boundaries [8]. These systems must maintain awareness of tenant-specific risk factors and security requirements while adapting authentication policies accordingly.

3.2. Cryptographic Operations

The management of cryptographic operations in multi-tenant environments requires meticulous attention to key material isolation and operational contexts. NIST guidelines specify strict requirements for cryptographic security, including the use of approved cryptographic algorithms and key sizes appropriate to the authentication assurance level [7]. In multi-tenant implementations, these requirements must be extended to ensure complete isolation of cryptographic materials between tenants.

Signing operations in multi-tenant environments necessitate careful implementation of tenant-specific contexts. Multi-tenancy security best practices emphasize the importance of maintaining strict separation of cryptographic operations between tenants [8]. This includes implementing separate trust roots for attestation, maintaining isolated signing contexts, and ensuring that all cryptographic operations are bound to their respective tenant contexts.

Key rotation and management procedures require careful consideration in multi-tenant systems. NIST guidelines specify requirements for key security and management, including provisions for secure key storage and rotation [7]. Organizations must implement these requirements while maintaining strict tenant boundaries, ensuring that key management procedures and cryptographic operation logs remain isolated between tenants. The implementation of comprehensive logging and monitoring systems remains crucial for maintaining security visibility across tenant boundaries while preserving isolation [8].

3.3. Identity Federation and Enterprise Integration

The integration of identity federation and enterprise systems in multi-tenant FIDO RP servers represents a critical architectural consideration. According to the IETF OAuth 2.0 Security Best Current Practice guidelines, organizations implementing OAuth2 and OpenID Connect (OIDC) in multi-tenant environments must carefully design their federation architecture to prevent security vulnerabilities such as mix-up attacks, code injection, and cross-site request forgery [9]. The implementation must consider various attack vectors and incorporate appropriate countermeasures while maintaining clear tenant boundaries.

3.4. OAuth2 and OIDC Integration

Multi-tenant FIDO RP servers commonly integrate with OAuth2 and OpenID Connect for identity federation, requiring sophisticated configuration and management approaches. The OAuth 2.0 Security BCP emphasizes the importance of proper client authentication and the use of Proof Key for Code Exchange (PKCE) to prevent authorization code interception attacks [9]. In multi-tenant environments, these security measures must be implemented with careful consideration for tenant isolation, ensuring that authorization codes and tokens remain bound to their respective tenants.

The management of OAuth client credentials in multi-tenant environments requires particular attention to security considerations. The specification mandates the use of client authentication for confidential clients and recommends the implementation of client authentication even for public clients where possible [9]. Each tenant's OAuth client credentials must be managed separately, with proper validation of redirect URIs and implementation of state parameters to prevent cross-site request forgery attacks.

Session management in federated authentication scenarios presents unique challenges in multi-tenant environments. The OAuth 2.0 Security BCP provides specific guidance on token handling, emphasizing the importance of proper token validation and scope enforcement [9]. Organizations must implement these security controls while maintaining strict tenant boundaries, ensuring that access tokens and refresh tokens remain isolated within their intended tenant context.

3.5. Enterprise Integration and Implementation

The implementation of enterprise integration in multi-tenant environments requires careful consideration of organizational processes and security boundaries. According to enterprise implementation research, successful integration projects require a clear definition of security boundaries and careful attention to user management workflows [10]. The proper implementation of integration endpoints ensures a clear separation of management processes while maintaining security controls.

User management in multi-tenant enterprise implementations requires sophisticated coordination mechanisms. Enterprise integration studies emphasize the importance of establishing clear processes for user onboarding, attribute management, and access control [10]. These processes must be implemented with careful consideration for tenant boundaries, ensuring that user attributes and credentials remain properly isolated between tenants.

Access management coordination represents a critical aspect of enterprise integration in multi-tenant environments. Research indicates that successful enterprise implementations require careful attention to access control mechanisms and user lifecycle management [10]. The synchronization of user provisioning and de-provisioning processes across tenants must be carefully managed to maintain security boundaries while ensuring efficient operations.

Table 3 Enterprise Integration Components in Multi-Tenant Environments [9,10]

Component	Security Requirement	Implementation Focus	Integration Aspect
OAuth2/OIDC Federation	PKCE Implementation	Authorization Code Protection	Tenant Isolation
Client Authentication	Mandatory for Confidential Clients	Credential Management	Security Boundaries
Token Management	Scope Enforcement	Access Control	Context Isolation
Session Handling	Token Validation	State Management	Tenant Context
User Management	Attribute Isolation	Onboarding Workflow	Access Control
Integration Endpoints	Process Separation	Security Controls	Boundary Definition
Access Coordination	Lifecycle Management	Provisioning Process	Tenant Separation

4. Security considerations

Maintaining robust security in multi-tenant authentication systems requires a comprehensive approach to tenant isolation and activity monitoring. According to security research in multi-tenant cloud environments, organizations must implement multiple layers of security controls, including data discovery, classification, and access governance, to prevent unauthorized cross-tenant access and maintain proper audit trails [11]. The evolving landscape of multi-tenant deployments has made security considerations particularly critical, with proper implementation of security controls and monitoring capabilities being essential for maintaining system integrity.

4.1. Cross-Tenant Protection

The prevention of data leakage between tenants stands as a fundamental security requirement in multi-tenant systems. Research into cloud security architecture emphasizes the importance of implementing comprehensive data isolation strategies, including network segmentation, access controls, and encryption mechanisms [11]. These controls must span multiple layers of the authentication system, ensuring complete separation between tenant environments while maintaining operational efficiency.

Tenant identification and validation mechanisms serve as the first line of defense against unauthorized access attempts. Security best practices emphasize the importance of implementing strong identity and access management controls, including role-based access control (RBAC) and proper authentication mechanisms for each tenant [11]. This includes the implementation of sophisticated tenant context management systems that maintain strict boundaries throughout the entire authentication process.

Credential storage and retrieval systems must maintain complete isolation between tenants to prevent unauthorized access. Multi-tenant security frameworks recommend implementing separate storage mechanisms for each tenant's credentials and sensitive data, with proper encryption and access controls in place [11]. The implementation of separate authentication state management for each tenant further enhances security, ensuring that session data and authentication states remain properly isolated.

4.2. Audit Logging

Comprehensive audit logging serves as a critical component of multi-tenant security infrastructure. According to IBM's multi-tenant management guidelines, organizations must implement detailed logging mechanisms that capture all security-relevant events while maintaining proper tenant context [12]. The logging system must be capable of tracking and recording all activities within the multi-tenant environment while maintaining clear separation between tenant data.

Authentication event logging must maintain complete tenant context to enable effective security monitoring. Multi-tenant management systems must implement comprehensive logging of all authentication-related activities, including successful and failed login attempts, password changes, and account lockouts [12]. The logging system should maintain proper tenant identification throughout all recorded events, enabling effective monitoring and analysis of security incidents.

Administrative action logging across tenant boundaries requires particular attention to detail. Security guidelines emphasize the importance of maintaining detailed audit trails of all administrative actions, including changes to tenant configurations, security policies, and access controls [12]. The logging system must be capable of tracking these activities while maintaining proper tenant context, enabling effective security monitoring and compliance reporting.

Credential lifecycle event logging must maintain detailed records of all credential-related activities. Multi-tenant systems must implement comprehensive logging of credential management activities, including the creation, modification, and deletion of authentication credentials [12]. Integration and federation activity logging should capture all relevant events related to identity federation and system integration, maintaining proper tenant context throughout all recorded activities.

5. Cloud-native implementation

The implementation of multi-tenant FIDO RP servers in cloud-native environments requires careful consideration of architecture and security factors. According to the Cloud Native Computing Foundation's security whitepaper, cloud-native security must be built-in throughout the application lifecycle, incorporating supply chain security, infrastructure security, and runtime security considerations [13]. The transition to cloud-native architectures necessitates a comprehensive approach to security that spans development, deployment, and operational phases.

5.1. Deployment Architecture

Modern multi-tenant FIDO RP servers increasingly leverage cloud-native architectures to enhance security and reliability. The CNCF emphasizes the importance of implementing security controls across four key cloud-native security layers: code, container, cluster, and cloud [13]. This layered approach ensures that security measures are implemented consistently throughout the application stack, from application code to cloud infrastructure.

Container security represents a fundamental aspect of cloud-native deployments. The CNCF security guidelines emphasize the importance of implementing proper container security measures, including image scanning, runtime protection, and supply chain security [13]. These security controls must be implemented while maintaining proper tenant isolation and ensuring that containerized workloads remain properly secured throughout their lifecycle.

Kubernetes orchestration has become a critical component for managing multi-tenant deployments effectively. According to research on cloud-native architectures, organizations must implement proper security controls at the cluster level, including network policies, pod security policies, and proper RBAC configurations [13]. These controls ensure that tenant workloads remain isolated while maintaining operational efficiency.

5.2. Scaling Considerations

Research into multi-tenancy in cloud-native architectures indicates that organizations must carefully consider both security and performance aspects when implementing scaling strategies. A systematic mapping study of cloud-native multi-tenancy identified several key architectural patterns for ensuring proper tenant isolation while maintaining system scalability [14]. These patterns must be carefully implemented to ensure both security and performance requirements are met.

The implementation of proper resource isolation mechanisms represents a critical consideration in cloud-native environments. Research has identified various approaches to resource isolation in cloud-native architectures, including namespace-based isolation, network isolation, and storage isolation [14]. These isolation mechanisms must be carefully implemented to maintain security boundaries while enabling efficient resource utilization.

Database management in cloud-native environments requires careful consideration of both scalability and security factors. The systematic study of cloud-native multi-tenancy patterns emphasizes the importance of implementing proper data isolation strategies, including separate storage volumes, dedicated database instances, or schema-level isolation [14]. These strategies must be carefully selected based on specific security and performance requirements.

Load balancing and service discovery mechanisms must be designed with multi-tenancy in mind. Research into cloud-native architectures has identified various patterns for implementing tenant-aware routing and load balancing, ensuring that tenant traffic remains properly isolated while maintaining system performance [14]. These mechanisms must be carefully implemented to prevent cross-tenant access while enabling efficient resource utilization.

Table 4 Multi-Tenant Cloud Architecture Security Controls [13, 14]

Security Layer	Primary Controls	Isolation Mechanism	Implementation Focus
Code Level	Supply Chain Security	Application Isolation	Development Security
Container Level	Image Scanning	Runtime Protection	Workload Security
Cluster Level	Network Policies	Pod Security	RBAC Configuration
Cloud Level	Infrastructure Security	Resource Isolation	Platform Security
Resource Management	Namespace Isolation	Storage Volumes	Resource Allocation
Network Management	Service Discovery	Load Balancing	Traffic Isolation
Database Management	Schema Isolation	Instance Separation	Data Security
Scaling Management	Tenant-aware Routing	Performance Controls	Resource Optimization

6. Conclusion

The implementation of multi-tenant FIDO RP servers represents a critical advancement in modern authentication architecture. The successful deployment requires careful consideration of multiple architectural layers, from database design to security controls, while maintaining strict tenant isolation. The integration of identity federation mechanisms and enterprise systems must be balanced with robust security measures and proper audit logging capabilities. Cloud-native architectures provide significant advantages for scalability and reliability, though they demand careful attention to resource isolation and security boundaries. As organizations continue to adopt passwordless authentication, the proper implementation of multi-tenant FIDO solutions becomes increasingly vital for maintaining secure and efficient authentication services across diverse organizational requirements.

References

[1] Axiomatics, "The State of Authorization Report 2023," [Online]. Available: https://axiomatics.com/resources/reports/state-of-authorization-report-2023

[2] BetterCloud, "2023 State of SaaSOps Report," 2022. Available: https://www.bettercloud.com/monitor/the-2023-state-of-saasops-report/

[3] Sharon Solomon, "WebAuthn: The Future of Passwordless Authentication," Frontegg, 2023. [Online]. Available: https://frontegg.com/blog/webauthn

[4] Dennis Okpara, "FIDO2 Deployment in the Enterprise," IDEE, 2023. [Online]. Available: https://www.getidee.com/blog/fido2-deployment-in-the-enterprise

[5] Qrvey, "What is Multi-Tenant Security? Definition, Risks, and Best Practices," [Online]. Available: https://qrvey.com/blog/multi-tenant-security/

[6] Frederik Chong et al., "Multi-Tenant Data Architecture," Citus Data. 2006. Available: https://docs.citusdata.com/en/v7.2/_static/mt-data-arch.pdf

[7] Paul A. Grassi et al., "Digital Identity Guidelines - Authentication and Lifecycle Management," NIST, 2017. Available: https://www.nist.gov/publications/digital-identity-guidelines-authentication-and-lifecycle-management

[8] Surya Kollimarla, "Best Practices to Secure Multi-Tenant Environments," ColorTokens, 2024 https://colortokens.com/blogs/multi-tenancy-best-practices/

[9] T. Lodderstedt et al., "Best Current Practice for OAuth 2.0 Security," Datatracker, 2025. https://datatracker.ietf.org/doc/html/rfc9700

[10] Andrew Filev, "What Is the Best Way to Enterprise 2.0 Implementation?" Wrike 2021. Available: https://www.wrike.com/blog/what-is-the-best-way-to-enterprise-2-0-implementation/

[11] Alexis Porter, "Maximizing Security in Multi-Tenant Cloud Environments," BigID, 2024. [Online]. Available: https://bigid.com/blog/maximizing-security-in-multi-tenant-cloud-environments/

[12] IBM, "Multi-tenant Management,", 2024. Available: https://www.ibm.com/docs/en/qsip/7.4?topic=administration-multitenant-management

[13] Brandon Krieger et al., "Cloud Native Security Whitepaper," CNCF, 2020. [Online]. Available: https://www.cncf.io/wp-content/uploads/2022/06/CNCF_cloud-native-security-whitepaper-May2022-v2.pdf

[14] Daniel Olabanji et al., "Multi-tenancy in Cloud-native Architecture: A Systematic Mapping Study," ResearchGate, 2023 [Online]. Available: https://www.researchgate.net/publication/369105350_Multi-tenancy_in_Cloud-native_Architecture_A_Systematic_Mapping_Study